

# KL25 Sub-Family Reference Manual

Supports: MKL25Z32VFM4, MKL25Z64VFM4, MKL25Z128VFM4,  
MKL25Z32VFT4, MKL25Z64VFT4, MKL25Z128VFT4,  
MKL25Z32VLH4, MKL25Z64VLH4, MKL25Z128VLH4,  
MKL25Z32VLK4, MKL25Z64VLK4, and MKL25Z128VLK4



Document Number: KL25P80M48SF0RM  
Rev. 3, September 2012





## Contents

| Section number                           | Title                               | Page |
|--|-------------------------------------|------|
| <b>Chapter 1<br/>About This Document</b> |                                     |      |
| 1.1                                      | Overview.....                       | 33   |
| 1.1.1                                    | Purpose.....                        | 33   |
| 1.1.2                                    | Audience.....                       | 33   |
| 1.2                                      | Conventions.....                    | 33   |
| 1.2.1                                    | Numbering systems.....              | 33   |
| 1.2.2                                    | Typographic notation.....           | 34   |
| 1.2.3                                    | Special terms.....                  | 34   |
| <b>Chapter 2<br/>Introduction</b>        |                                     |      |
| 2.1                                      | Overview.....                       | 35   |
| 2.2                                      | Kinetis L Series.....               | 35   |
| 2.3                                      | KL25 Sub-Family Introduction.....   | 38   |
| 2.4                                      | Module functional categories.....   | 39   |
| 2.4.1                                    | ARM® Cortex™-M0+ Core Modules.....  | 39   |
| 2.4.2                                    | System Modules.....                 | 40   |
| 2.4.3                                    | Memories and Memory Interfaces..... | 41   |
| 2.4.4                                    | Clocks.....                         | 41   |
| 2.4.5                                    | Security and Integrity modules..... | 42   |
| 2.4.6                                    | Analog modules.....                 | 42   |
| 2.4.7                                    | Timer modules.....                  | 42   |
| 2.4.8                                    | Communication interfaces.....       | 43   |
| 2.4.9                                    | Human-machine interfaces.....       | 44   |
| 2.5                                      | Orderable part numbers.....         | 44   |
| <b>Chapter 3<br/>Chip Configuration</b>  |                                     |      |
| 3.1                                      | Introduction.....                   | 45   |

| <b>Section number</b> | <b>Title</b>  | <b>Page</b> |
|-----------------------|---|-------------|
| 3.2                   | Module to Module Interconnects.....                                 | 45          |
| 3.2.1                 | Module to Module Interconnects.....                                 | 45          |
| 3.2.2                 | Analog reference options.....                                       | 48          |
| 3.3                   | Core Modules.....   | 48          |
| 3.3.1                 | ARM Cortex-M0+ Core Configuration.....                              | 48          |
| 3.3.2                 | Nested Vectored Interrupt Controller (NVIC) Configuration.....      | 51          |
| 3.3.3                 | Asynchronous wake-up interrupt controller (AWIC) configuration..... | 55          |
| 3.4                   | System Modules.....   | 56          |
| 3.4.1                 | SIM Configuration.....  | 56          |
| 3.4.2                 | System Mode Controller (SMC) Configuration.....                     | 57          |
| 3.4.3                 | PMC Configuration.....  | 57          |
| 3.4.4                 | Low-Leakage Wake-up Unit (LLWU) Configuration.....                  | 58          |
| 3.4.5                 | MCM Configuration.....  | 60          |
| 3.4.6                 | Crossbar-Light Switch Configuration.....                            | 61          |
| 3.4.7                 | Peripheral Bridge Configuration.....                                | 62          |
| 3.4.8                 | DMA request multiplexer configuration.....                          | 63          |
| 3.4.9                 | DMA Controller Configuration.....                                   | 66          |
| 3.4.10                | Computer Operating Properly (COP) Watchdog Configuration.....       | 67          |
| 3.5                   | Clock Modules.....  | 70          |
| 3.5.1                 | MCG Configuration.....  | 70          |
| 3.5.2                 | OSC Configuration.....  | 71          |
| 3.6                   | Memories and Memory Interfaces.....                                 | 72          |
| 3.6.1                 | Flash Memory Configuration.....                                     | 72          |
| 3.6.2                 | Flash Memory Controller Configuration.....                          | 74          |

| <b>Section number</b> | <b>Title</b>                                 | <b>Page</b> |
|-----------------------|--|-------------|
| 3.6.3                 | SRAM Configuration.....                      | 75          |
| 3.7                   | Analog.....                                  | 77          |
| 3.7.1                 | 16-bit SAR ADC Configuration.....            | 77          |
| 3.7.2                 | CMP Configuration.....                       | 81          |
| 3.7.3                 | 12-bit DAC Configuration.....                | 83          |
| 3.8                   | Timers.....                                  | 84          |
| 3.8.1                 | Timer/PWM Module Configuration.....          | 84          |
| 3.8.2                 | PIT Configuration.....                       | 87          |
| 3.8.3                 | Low-power timer configuration.....           | 88          |
| 3.8.4                 | RTC configuration.....                       | 90          |
| 3.9                   | Communication interfaces.....                | 91          |
| 3.9.1                 | Universal Serial Bus (USB) FS Subsystem..... | 91          |
| 3.9.2                 | SPI configuration.....                       | 96          |
| 3.9.3                 | I2C Configuration.....                       | 97          |
| 3.9.4                 | UART Configuration.....                      | 98          |
| 3.10                  | Human-machine interfaces (HMI).....          | 99          |
| 3.10.1                | GPIO Configuration.....                      | 99          |
| 3.10.2                | TSI Configuration.....                       | 101         |

## Chapter 4 Memory Map

|       |  |     |
|-------|--|-----|
| 4.1   | Introduction.....  | 105 |
| 4.2   | System memory map.....   | 105 |
| 4.3   | Flash Memory Map.....  | 106 |
| 4.3.1 | Alternate Non-Volatile IRC User Trim Description.....                          | 106 |
| 4.4   | SRAM memory map.....   | 107 |
| 4.5   | Bit Manipulation Engine.....   | 107 |
| 4.6   | Peripheral bridge (AIPS-Lite) memory map.....                                  | 108 |
| 4.6.1 | Read-after-write sequence and required serialization of memory operations..... | 108 |
| 4.6.2 | Peripheral Bridge (AIPS-Lite) Memory Map.....                                  | 109 |

| <b>Section number</b>                   | <b>Title</b>                                 | <b>Page</b> |
|---|--|-------------|
| 4.6.3                                   | Modules Restricted Access in User Mode.....  | 112         |
| 4.7                                     | Private Peripheral Bus (PPB) memory map..... | 112         |
| <b>Chapter 5<br/>Clock Distribution</b> |  |             |
| 5.1                                     | Introduction.....                            | 115         |
| 5.2                                     | Programming model.....                       | 115         |
| 5.3                                     | High-Level device clocking diagram.....      | 115         |
| 5.4                                     | Clock definitions.....                       | 116         |
| 5.4.1                                   | Device clock summary.....                    | 117         |
| 5.5                                     | Internal clocking requirements.....          | 119         |
| 5.5.1                                   | Clock divider values after reset.....        | 119         |
| 5.5.2                                   | VLPR mode clocking.....                      | 120         |
| 5.6                                     | Clock Gating.....                            | 121         |
| 5.7                                     | Module clocks.....                           | 121         |
| 5.7.1                                   | PMC 1-kHz LPO clock.....                     | 122         |
| 5.7.2                                   | COP clocking.....                            | 122         |
| 5.7.3                                   | RTC clocking.....                            | 123         |
| 5.7.4                                   | LPTMR clocking.....                          | 123         |
| 5.7.5                                   | TPM clocking.....                            | 124         |
| 5.7.6                                   | USB FS OTG Controller clocking.....          | 124         |
| 5.7.7                                   | UART clocking.....                           | 125         |
| <b>Chapter 6<br/>Reset and Boot</b>     |  |             |
| 6.1                                     | Introduction.....                            | 127         |
| 6.2                                     | Reset.....                                   | 127         |
| 6.2.1                                   | Power-on reset (POR).....                    | 128         |
| 6.2.2                                   | System reset sources.....                    | 128         |
| 6.2.3                                   | MCU Resets.....                              | 131         |
| 6.2.4                                   | Reset Pin .....                              | 133         |

| <b>Section number</b> | <b>Title</b>           | <b>Page</b> |
|-----------------------|------------------------|-------------|
| 6.2.5                 | Debug resets.....      | 133         |
| 6.3                   | Boot.....              | 134         |
| 6.3.1                 | Boot sources.....      | 134         |
| 6.3.2                 | FOPT boot options..... | 134         |
| 6.3.3                 | Boot sequence.....     | 135         |

## **Chapter 7 Power Management**

|       |  |     |
|-------|--|-----|
| 7.1   | Introduction.....                        | 137 |
| 7.2   | Clocking Modes.....                      | 137 |
| 7.2.1 | Partial Stop.....                        | 137 |
| 7.2.2 | DMA Wakeup.....                          | 138 |
| 7.2.3 | Compute Operation.....                   | 139 |
| 7.2.4 | Peripheral Doze.....                     | 140 |
| 7.2.5 | Clock Gating.....                        | 141 |
| 7.3   | Power modes.....                         | 141 |
| 7.4   | Entering and exiting power modes.....    | 143 |
| 7.5   | Module Operation in Low Power Modes..... | 143 |

## **Chapter 8 Security**

|       |   |     |
|-------|---|-----|
| 8.1   | Introduction.....                             | 149 |
| 8.2   | Flash Security.....                           | 149 |
| 8.3   | Security Interactions with other Modules..... | 149 |
| 8.3.1 | Security Interactions with Debug.....         | 150 |

## **Chapter 9 Debug**

|       |                                       |     |
|-------|---------------------------------------|-----|
| 9.1   | Introduction.....                     | 151 |
| 9.2   | Debug Port Pin Descriptions.....      | 151 |
| 9.3   | SWD status and control registers..... | 152 |
| 9.3.1 | MDM-AP Control Register.....          | 153 |
| 9.3.2 | MDM-AP Status Register.....           | 154 |

| <b>Section number</b> | <b>Title</b>                  | <b>Page</b> |
|-----------------------|-------------------------------|-------------|
| 9.4                   | Debug Resets.....             | 156         |
| 9.5                   | Micro Trace Buffer (MTB)..... | 157         |
| 9.6                   | Debug in Low Power Modes..... | 157         |
| 9.7                   | Debug & Security.....         | 157         |

## **Chapter 10 Signal Multiplexing and Signal Descriptions**

|        |   |     |
|--------|---|-----|
| 10.1   | Introduction.....                                 | 159 |
| 10.2   | Signal Multiplexing Integration.....              | 159 |
| 10.2.1 | Port control and interrupt module features.....   | 160 |
| 10.2.2 | Clock gating.....                                 | 161 |
| 10.2.3 | Signal multiplexing constraints.....              | 161 |
| 10.3   | Pinout.....                                       | 161 |
| 10.3.1 | KL25 Signal Multiplexing and Pin Assignments..... | 161 |
| 10.3.2 | KL25 Pinouts.....                                 | 164 |
| 10.4   | Module Signal Description Tables.....             | 168 |
| 10.4.1 | Core Modules.....                                 | 168 |
| 10.4.2 | System Modules.....                               | 169 |
| 10.4.3 | Clock Modules.....                                | 169 |
| 10.4.4 | Memories and Memory Interfaces.....               | 169 |
| 10.4.5 | Analog.....                                       | 169 |
| 10.4.6 | Timer Modules.....                                | 170 |
| 10.4.7 | Communication Interfaces.....                     | 171 |
| 10.4.8 | Human-Machine Interfaces (HMI).....               | 173 |

## **Chapter 11 Port control and interrupts (PORT)**

|        |                   |     |
|--------|-------------------|-----|
| 11.1   | Introduction..... | 175 |
| 11.2   | Overview.....     | 175 |
| 11.2.1 | Features.....     | 175 |

| <b>Section number</b> | <b>Title</b>   | <b>Page</b> |
|-----------------------|--|-------------|
| 11.2.2                | Modes of operation.....  | 176         |
| 11.3                  | External signal description.....                                   | 176         |
| 11.4                  | Detailed signal description.....                                   | 177         |
| 11.5                  | Memory map and register definition.....                            | 177         |
| 11.5.1                | Pin Control Register n (PORT <sub>x</sub> _PCR <sub>n</sub> )..... | 183         |
| 11.5.2                | Global Pin Control Low Register (PORT <sub>x</sub> _GPCLR).....    | 185         |
| 11.5.3                | Global Pin Control High Register (PORT <sub>x</sub> _GPCHR).....   | 186         |
| 11.5.4                | Interrupt Status Flag Register (PORT <sub>x</sub> _ISFR).....      | 186         |
| 11.6                  | Functional description.....  | 187         |
| 11.6.1                | Pin control.....   | 187         |
| 11.6.2                | Global pin control.....  | 188         |
| 11.6.3                | External interrupts.....   | 188         |

## Chapter 12 System integration module (SIM)

|         |   |     |
|---------|---|-----|
| 12.1    | Introduction.....                                       | 191 |
| 12.1.1  | Features.....   | 191 |
| 12.2    | Memory map and register definition.....                 | 191 |
| 12.2.1  | System Options Register 1 (SIM_SOPT1).....              | 193 |
| 12.2.2  | SOPT1 Configuration Register (SIM_SOPT1CFG).....        | 194 |
| 12.2.3  | System Options Register 2 (SIM_SOPT2).....              | 195 |
| 12.2.4  | System Options Register 4 (SIM_SOPT4).....              | 197 |
| 12.2.5  | System Options Register 5 (SIM_SOPT5).....              | 199 |
| 12.2.6  | System Options Register 7 (SIM_SOPT7).....              | 200 |
| 12.2.7  | System Device Identification Register (SIM_SDID).....   | 202 |
| 12.2.8  | System Clock Gating Control Register 4 (SIM_SCGC4)..... | 204 |
| 12.2.9  | System Clock Gating Control Register 5 (SIM_SCGC5)..... | 206 |
| 12.2.10 | System Clock Gating Control Register 6 (SIM_SCGC6)..... | 207 |
| 12.2.11 | System Clock Gating Control Register 7 (SIM_SCGC7)..... | 209 |
| 12.2.12 | System Clock Divider Register 1 (SIM_CLKDIV1).....      | 210 |

| <b>Section number</b> | <b>Title</b>   | <b>Page</b> |
|-----------------------|--|-------------|
| 12.2.13               | Flash Configuration Register 1 (SIM_FCFG1).....          | 211         |
| 12.2.14               | Flash Configuration Register 2 (SIM_FCFG2).....          | 213         |
| 12.2.15               | Unique Identification Register Mid-High (SIM_UIDMH)..... | 213         |
| 12.2.16               | Unique Identification Register Mid Low (SIM_UIDML).....  | 214         |
| 12.2.17               | Unique Identification Register Low (SIM_UIDL).....       | 214         |
| 12.2.18               | COP Control Register (SIM_COPC).....                     | 215         |
| 12.2.19               | Service COP Register (SIM_SRVCOP).....                   | 216         |
| 12.3                  | Functional description.....                              | 216         |

## **Chapter 13 System Mode Controller (SMC)**

|        |  |     |
|--------|--|-----|
| 13.1   | Introduction.....                                | 217 |
| 13.2   | Modes of operation.....                          | 217 |
| 13.3   | Memory map and register descriptions.....        | 219 |
| 13.3.1 | Power Mode Protection register (SMC_PMPROT)..... | 219 |
| 13.3.2 | Power Mode Control register (SMC_PMCTRL).....    | 221 |
| 13.3.3 | Stop Control Register (SMC_STOPCTRL).....        | 222 |
| 13.3.4 | Power Mode Status register (SMC_PMSTAT).....     | 223 |
| 13.4   | Functional description.....                      | 224 |
| 13.4.1 | Power mode transitions.....                      | 224 |
| 13.4.2 | Power mode entry/exit sequencing.....            | 227 |
| 13.4.3 | Run modes.....                                   | 229 |
| 13.4.4 | Wait modes.....                                  | 231 |
| 13.4.5 | Stop modes.....                                  | 232 |
| 13.4.6 | Debug in low power modes.....                    | 235 |

## **Chapter 14 Power Management Controller (PMC)**

|      |                   |     |
|------|-------------------|-----|
| 14.1 | Introduction..... | 237 |
| 14.2 | Features.....     | 237 |

| <b>Section number</b>                                      | <b>Title</b>   | <b>Page</b> |
|--|--|-------------|
| 14.3   | Low-voltage detect (LVD) system.....                               | 237         |
| 14.3.1   | LVD reset operation.....   | 238         |
| 14.3.2   | LVD interrupt operation.....                                       | 238         |
| 14.3.3   | Low-voltage warning (LVW) interrupt operation.....                 | 238         |
| 14.4   | I/O retention.....   | 239         |
| 14.5   | Memory map and register descriptions.....                          | 239         |
| 14.5.1   | Low Voltage Detect Status And Control 1 register (PMC_LVDSC1)..... | 240         |
| 14.5.2   | Low Voltage Detect Status And Control 2 register (PMC_LVDSC2)..... | 241         |
| 14.5.3   | Regulator Status And Control register (PMC_REGSC).....             | 242         |
| <b>Chapter 15</b><br><b>Low-Leakage Wakeup Unit (LLWU)</b> |  |             |
| 15.1   | Introduction.....  | 245         |
| 15.1.1   | Features.....  | 245         |
| 15.1.2   | Modes of operation.....  | 246         |
| 15.1.3   | Block diagram.....   | 247         |
| 15.2   | LLWU signal descriptions.....                                      | 248         |
| 15.3   | Memory map/register definition.....                                | 248         |
| 15.3.1   | LLWU Pin Enable 1 register (LLWU_PE1).....                         | 249         |
| 15.3.2   | LLWU Pin Enable 2 register (LLWU_PE2).....                         | 250         |
| 15.3.3   | LLWU Pin Enable 3 register (LLWU_PE3).....                         | 251         |
| 15.3.4   | LLWU Pin Enable 4 register (LLWU_PE4).....                         | 252         |
| 15.3.5   | LLWU Module Enable register (LLWU_ME).....                         | 253         |
| 15.3.6   | LLWU Flag 1 register (LLWU_F1).....                                | 255         |
| 15.3.7   | LLWU Flag 2 register (LLWU_F2).....                                | 257         |
| 15.3.8   | LLWU Flag 3 register (LLWU_F3).....                                | 258         |
| 15.3.9   | LLWU Pin Filter 1 register (LLWU_FILTER1).....                     | 260         |
| 15.3.10  | LLWU Pin Filter 2 register (LLWU_FILTER2).....                     | 261         |
| 15.4   | Functional description.....  | 262         |
| 15.4.1   | LLS mode.....  | 263         |

| Section number | Title               | Page |
|----------------|---------------------|------|
| 15.4.2         | VLLS modes.....     | 263  |
| 15.4.3         | Initialization..... | 263  |

**Chapter 16**  
**Reset Control Module (RCM)**

| <b>Section number</b>                                  | <b>Title</b>                                      | <b>Page</b> |
|--|---|-------------|
| 18.2.3   | Platform Control Register (MCM_PLACR).....        | 293         |
| 18.2.4   | Compute Operation Control Register (MCM_CPO)..... | 296         |
| <b>Chapter 19<br/>Micro Trace Buffer (MTB)</b>         |   |             |
| 19.1   | Introduction.....                                 | 299         |
| 19.1.1   | Overview.....                                     | 299         |
| 19.1.2   | Features.....                                     | 302         |
| 19.1.3   | Modes of Operation.....                           | 303         |
| 19.2   | External Signal Description.....                  | 303         |
| 19.3   | Memory Map and Register Definition.....           | 304         |
| 19.3.1   | MTB_RAM Memory Map.....                           | 304         |
| 19.3.2   | MTB_DWT Memory Map.....                           | 316         |
| 19.3.3   | System ROM Memory Map.....                        | 326         |
| <b>Chapter 20<br/>Crossbar Switch Lite (AXBS-Lite)</b> |   |             |
| 20.1   | Introduction.....                                 | 331         |
| 20.1.1   | Features.....                                     | 331         |
| 20.2   | Memory Map / Register Definition.....             | 331         |
| 20.3   | Functional Description.....                       | 332         |
| 20.3.1   | General operation.....                            | 332         |
| 20.3.2   | Arbitration.....                                  | 333         |
| 20.4   | Initialization/application information.....       | 334         |
| <b>Chapter 21<br/>Peripheral Bridge (AIPS-Lite)</b>    |   |             |
| 21.1   | Introduction.....                                 | 335         |
| 21.1.1   | Features.....                                     | 335         |
| 21.1.2   | General operation.....                            | 335         |
| 21.2   | Functional description.....                       | 336         |
| 21.2.1   | Access support.....                               | 336         |

| Section number | Title   | Page |
|----------------|---|------|
|                | <b>Chapter 22<br/>Direct Memory Access Multiplexer (DMAMUX)</b>   |      |
| 22.1           | Introduction.....   | .337 |
| 22.1.1         | Overview.....   | .337 |
| 22.1.2         | Features.....   | .338 |
| 22.1.3         | Modes of operation.....   | .338 |
| 22.2           | External signal description.....                                  | .339 |
| 22.3           | Memory map/register definition.....                               | .339 |
| 22.3.1         | Channel Configuration register (DMAMUX $x$ _CHCFG $n$ ).....      | .339 |
| 22.4           | Functional description.....                                       | .340 |
| 22.4.1         | DMA channels with periodic triggering capability.....             | .341 |
| 22.4.2         | DMA channels with no triggering capability.....                   | .343 |
| 22.4.3         | Always-enabled DMA sources.....                                   | .343 |
| 22.5           | Initialization/application information.....                       | .344 |
| 22.5.1         | Reset.....  | .344 |
| 22.5.2         | Enabling and configuring sources.....                             | .344 |
|                | <b>Chapter 23<br/>DMA Controller Module</b>                       |      |
| 23.1           | Introduction.....   | .349 |
| 23.1.1         | Overview.....   | .349 |
| 23.1.2         | Features.....   | .350 |
| 23.2           | DMA Transfer Overview.....  | .351 |
| 23.3           | Memory Map and Registers.....                                     | .352 |
| 23.3.1         | Source Address Register (DMA_SAR $n$ ).....                       | .353 |
| 23.3.2         | Destination Address Register (DMA_DAR $n$ ).....                  | .354 |
| 23.3.3         | DMA Status Register / Byte Count Register (DMA_DSR_BCR $n$ )..... | .355 |
| 23.3.4         | DMA Control Register (DMA_DCR $n$ ).....                          | .357 |
| 23.4           | Functional Description.....                                       | .361 |
| 23.4.1         | Transfer Requests (Cycle-Steal and Continuous Modes).....         | .361 |

| <b>Section number</b>  | <b>Title</b>   | <b>Page</b> |
|--|--|-------------|
| 23.4.2   | Channel Initialization and Startup.....                    | 361         |
| 23.4.3   | Dual-Address Data Transfer Mode.....                       | 363         |
| 23.4.4   | Advanced Data Transfer Controls: Auto-Alignment.....       | 364         |
| 23.4.5   | Termination.....   | 365         |
| <b>Chapter 24</b><br><b>Multipurpose Clock Generator (MCG)</b> |  |             |
| 24.1   | Introduction.....  | 367         |
| 24.1.1   | Features.....  | 367         |
| 24.1.2   | Modes of Operation.....                                    | 370         |
| 24.2   | External Signal Description.....                           | 371         |
| 24.3   | Memory Map/Register Definition.....                        | 371         |
| 24.3.1   | MCG Control 1 Register (MCG_C1).....                       | 372         |
| 24.3.2   | MCG Control 2 Register (MCG_C2).....                       | 373         |
| 24.3.3   | MCG Control 3 Register (MCG_C3).....                       | 374         |
| 24.3.4   | MCG Control 4 Register (MCG_C4).....                       | 374         |
| 24.3.5   | MCG Control 5 Register (MCG_C5).....                       | 376         |
| 24.3.6   | MCG Control 6 Register (MCG_C6).....                       | 377         |
| 24.3.7   | MCG Status Register (MCG_S).....                           | 378         |
| 24.3.8   | MCG Status and Control Register (MCG_SC).....              | 380         |
| 24.3.9   | MCG Auto Trim Compare Value High Register (MCG_ATCVH)..... | 381         |
| 24.3.10  | MCG Auto Trim Compare Value Low Register (MCG_ATCVL).....  | 381         |
| 24.3.11  | MCG Control 7 Register (MCG_C7).....                       | 382         |
| 24.3.12  | MCG Control 8 Register (MCG_C8).....                       | 382         |
| 24.3.13  | MCG Control 9 Register (MCG_C9).....                       | 383         |
| 24.3.14  | MCG Control 10 Register (MCG_C10).....                     | 383         |
| 24.4   | Functional Description.....                                | 384         |
| 24.4.1   | MCG mode state diagram.....                                | 384         |
| 24.4.2   | Low Power Bit Usage.....                                   | 388         |

| <b>Section number</b> | <b>Title</b>                                  | <b>Page</b> |
|-----------------------|---|-------------|
| 24.4.3                | MCG Internal Reference Clocks.....            | 388         |
| 24.4.4                | External Reference Clock.....                 | 389         |
| 24.4.5                | MCG Fixed frequency clock .....               | 389         |
| 24.4.6                | MCG PLL clock .....                           | 390         |
| 24.4.7                | MCG Auto TRIM (ATM).....                      | 390         |
| 24.5                  | Initialization / Application information..... | 391         |
| 24.5.1                | MCG module initialization sequence.....       | 391         |
| 24.5.2                | Using a 32.768 kHz reference.....             | 393         |
| 24.5.3                | MCG mode switching.....                       | 394         |

## **Chapter 25 Oscillator (OSC)**

|        |   |     |
|--------|---|-----|
| 25.1   | Introduction.....                             | 405 |
| 25.2   | Features and Modes.....                       | 405 |
| 25.3   | Block Diagram.....                            | 406 |
| 25.4   | OSC Signal Descriptions.....                  | 406 |
| 25.5   | External Crystal / Resonator Connections..... | 407 |
| 25.6   | External Clock Connections.....               | 408 |
| 25.7   | Memory Map/Register Definitions.....          | 409 |
| 25.7.1 | OSC Memory Map/Register Definition.....       | 409 |
| 25.8   | Functional Description.....                   | 410 |
| 25.8.1 | OSC Module States.....                        | 410 |
| 25.8.2 | OSC Module Modes.....                         | 412 |
| 25.8.3 | Counter.....                                  | 413 |
| 25.8.4 | Reference Clock Pin Requirements.....         | 413 |
| 25.9   | Reset.....                                    | 414 |
| 25.10  | Low Power Modes Operation.....                | 414 |
| 25.11  | Interrupts.....                               | 414 |

| <b>Section number</b>                                     | <b>Title</b>                               | <b>Page</b> |
|---|--|-------------|
| <b>Chapter 26</b><br><b>Flash Memory Controller (FMC)</b> |  |             |
| 26.1  | Introduction.....                          | 415         |
| 26.1.1  | Overview.....                              | 415         |
| 26.1.2  | Features.....                              | 415         |
| 26.2  | Modes of operation.....                    | 416         |
| 26.3  | External signal description.....           | 416         |
| 26.4  | Memory map and register descriptions.....  | 416         |
| 26.5  | Functional description.....                | 416         |
| <b>Chapter 27</b><br><b>Flash Memory Module (FTFA)</b>    |  |             |
| 27.1  | Introduction.....                          | 419         |
| 27.1.1  | Features.....                              | 420         |
| 27.1.2  | Block Diagram.....                         | 420         |
| 27.1.3  | Glossary.....                              | 421         |
| 27.2  | External Signal Description.....           | 422         |
| 27.3  | Memory Map and Registers.....              | 422         |
| 27.3.1  | Flash Configuration Field Description..... | 422         |
| 27.3.2  | Program Flash IFR Map.....                 | 423         |
| 27.3.3  | Register Descriptions.....                 | 424         |
| 27.4  | Functional Description.....                | 432         |
| 27.4.1  | Flash Protection.....                      | 433         |
| 27.4.2  | Interrupts.....                            | 433         |
| 27.4.3  | Flash Operation in Low-Power Modes.....    | 434         |
| 27.4.4  | Functional Modes of Operation.....         | 434         |
| 27.4.5  | Flash Reads and Ignored Writes.....        | 434         |
| 27.4.6  | Read While Write (RWW).....                | 435         |
| 27.4.7  | Flash Program and Erase.....               | 435         |

| <b>Section number</b> | <b>Title</b>                   | <b>Page</b> |
|-----------------------|--------------------------------|-------------|
| 27.4.8                | Flash Command Operations.....  | 435         |
| 27.4.9                | Margin Read Commands.....      | 440         |
| 27.4.10               | Flash Command Description..... | 441         |
| 27.4.11               | Security.....                  | 454         |
| 27.4.12               | Reset Sequence.....            | 456         |

## **Chapter 28** **Analog-to-Digital Converter (ADC)**

|         |  |     |
|---------|--|-----|
| 28.1    | Introduction.....  | 457 |
| 28.1.1  | Features.....  | 457 |
| 28.1.2  | Block diagram.....   | 458 |
| 28.2    | ADC Signal Descriptions.....   | 459 |
| 28.2.1  | Analog Power (VDDA).....   | 460 |
| 28.2.2  | Analog Ground (VSSA).....  | 460 |
| 28.2.3  | Voltage Reference Select.....  | 460 |
| 28.2.4  | Analog Channel Inputs (ADx).....   | 461 |
| 28.2.5  | Differential Analog Channel Inputs (DADx).....                                 | 461 |
| 28.3    | Register definition.....   | 461 |
| 28.3.1  | ADC Status and Control Registers 1 (ADC <sub>x</sub> _SC1 <sub>n</sub> ).....  | 462 |
| 28.3.2  | ADC Configuration Register 1 (ADC <sub>x</sub> _CFG1).....                     | 465 |
| 28.3.3  | ADC Configuration Register 2 (ADC <sub>x</sub> _CFG2).....                     | 467 |
| 28.3.4  | ADC Data Result Register (ADC <sub>x</sub> _R <sub>n</sub> ).....              | 468 |
| 28.3.5  | Compare Value Registers (ADC <sub>x</sub> _CV <sub>n</sub> ).....              | 469 |
| 28.3.6  | Status and Control Register 2 (ADC <sub>x</sub> _SC2).....                     | 470 |
| 28.3.7  | Status and Control Register 3 (ADC <sub>x</sub> _SC3).....                     | 472 |
| 28.3.8  | ADC Offset Correction Register (ADC <sub>x</sub> _OFS).....                    | 474 |
| 28.3.9  | ADC Plus-Side Gain Register (ADC <sub>x</sub> _PG).....                        | 474 |
| 28.3.10 | ADC Minus-Side Gain Register (ADC <sub>x</sub> _MG).....                       | 475 |
| 28.3.11 | ADC Plus-Side General Calibration Value Register (ADC <sub>x</sub> _CLPD)..... | 475 |
| 28.3.12 | ADC Plus-Side General Calibration Value Register (ADC <sub>x</sub> _CLPS)..... | 476 |

| <b>Section number</b> | <b>Title</b>   | <b>Page</b> |
|-----------------------|--|-------------|
| 28.3.13               | ADC Plus-Side General Calibration Value Register (ADCx_CLP4).....  | 476         |
| 28.3.14               | ADC Plus-Side General Calibration Value Register (ADCx_CLP3).....  | 477         |
| 28.3.15               | ADC Plus-Side General Calibration Value Register (ADCx_CLP2).....  | 477         |
| 28.3.16               | ADC Plus-Side General Calibration Value Register (ADCx_CLP1).....  | 478         |
| 28.3.17               | ADC Plus-Side General Calibration Value Register (ADCx_CLP0).....  | 478         |
| 28.3.18               | ADC Minus-Side General Calibration Value Register (ADCx_CLMD)..... | 479         |
| 28.3.19               | ADC Minus-Side General Calibration Value Register (ADCx_CLMS)..... | 479         |
| 28.3.20               | ADC Minus-Side General Calibration Value Register (ADCx_CLM4)..... | 480         |
| 28.3.21               | ADC Minus-Side General Calibration Value Register (ADCx_CLM3)..... | 480         |
| 28.3.22               | ADC Minus-Side General Calibration Value Register (ADCx_CLM2)..... | 481         |
| 28.3.23               | ADC Minus-Side General Calibration Value Register (ADCx_CLM1)..... | 481         |
| 28.3.24               | ADC Minus-Side General Calibration Value Register (ADCx_CLM0)..... | 482         |
| 28.4                  | Functional description.....  | 482         |
| 28.4.1                | Clock select and divide control.....                               | 483         |
| 28.4.2                | Voltage reference selection.....                                   | 483         |
| 28.4.3                | Hardware trigger and channel selects.....                          | 484         |
| 28.4.4                | Conversion control.....  | 485         |
| 28.4.5                | Automatic compare function.....                                    | 493         |
| 28.4.6                | Calibration function.....  | 494         |
| 28.4.7                | User-defined offset function.....                                  | 495         |
| 28.4.8                | Temperature sensor.....  | 497         |
| 28.4.9                | MCU wait mode operation.....                                       | 497         |
| 28.4.10               | MCU Normal Stop mode operation.....                                | 498         |
| 28.4.11               | MCU Low-Power Stop mode operation.....                             | 499         |

| <b>Section number</b>                  | <b>Title</b>                                    | <b>Page</b> |
|--|---|-------------|
| 28.5                                   | Initialization information.....                 | 499         |
| 28.5.1                                 | ADC module initialization example.....          | 499         |
| 28.6                                   | Application information.....                    | 501         |
| 28.6.1                                 | External pins and routing.....                  | 501         |
| 28.6.2                                 | Sources of error.....                           | 503         |
| <b>Chapter 29<br/>Comparator (CMP)</b> |   |             |
| 29.1                                   | Introduction.....                               | 509         |
| 29.2                                   | CMP features.....                               | 509         |
| 29.3                                   | 6-bit DAC key features.....                     | 510         |
| 29.4                                   | ANMUX key features.....                         | 511         |
| 29.5                                   | CMP, DAC and ANMUX diagram.....                 | 511         |
| 29.6                                   | CMP block diagram.....                          | 512         |
| 29.7                                   | Memory map/register definitions.....            | 514         |
| 29.7.1                                 | CMP Control Register 0 (CMPx_CR0).....          | 514         |
| 29.7.2                                 | CMP Control Register 1 (CMPx_CR1).....          | 515         |
| 29.7.3                                 | CMP Filter Period Register (CMPx_FPR).....      | 517         |
| 29.7.4                                 | CMP Status and Control Register (CMPx_SCR)..... | 517         |
| 29.7.5                                 | DAC Control Register (CMPx_DACCR).....          | 518         |
| 29.7.6                                 | MUX Control Register (CMPx_MUXCR).....          | 519         |
| 29.8                                   | Functional description.....                     | 520         |
| 29.8.1                                 | CMP functional modes.....                       | 520         |
| 29.8.2                                 | Power modes.....                                | 529         |
| 29.8.3                                 | Startup and operation.....                      | 530         |
| 29.8.4                                 | Low-pass filter.....                            | 531         |
| 29.9                                   | CMP interrupts.....                             | 533         |
| 29.10                                  | DMA support.....                                | 533         |
| 29.11                                  | CMP Asynchronous DMA support.....               | 534         |
| 29.12                                  | Digital-to-analog converter.....                | 534         |

| <b>Section number</b> | <b>Title</b>                         | <b>Page</b> |
|-----------------------|--------------------------------------|-------------|
| 29.13                 | DAC functional description.....      | 535         |
| 29.13.1               | Voltage reference source select..... | 535         |
| 29.14                 | DAC resets.....                      | 535         |
| 29.15                 | DAC clocks.....                      | 535         |
| 29.16                 | DAC interrupts.....                  | 536         |
| 29.17                 | CMP Trigger Mode.....                | 536         |

## **Chapter 30 12-bit Digital-to-Analog Converter (DAC)**

|        |   |     |
|--------|---|-----|
| 30.1   | Introduction.....                                     | 537 |
| 30.2   | Features.....   | 537 |
| 30.3   | Block diagram.....                                    | 537 |
| 30.4   | Memory map/register definition.....                   | 539 |
| 30.4.1 | DAC Data Low Register (DAC <sub>x</sub> _DATnL).....  | 539 |
| 30.4.2 | DAC Data High Register (DAC <sub>x</sub> _DATnH)..... | 540 |
| 30.4.3 | DAC Status Register (DAC <sub>x</sub> _SR).....       | 540 |
| 30.4.4 | DAC Control Register (DAC <sub>x</sub> _C0).....      | 541 |
| 30.4.5 | DAC Control Register 1 (DAC <sub>x</sub> _C1).....    | 542 |
| 30.4.6 | DAC Control Register 2 (DAC <sub>x</sub> _C2).....    | 542 |
| 30.5   | Functional description.....                           | 543 |
| 30.5.1 | DAC data buffer operation.....                        | 543 |
| 30.5.2 | DMA operation.....                                    | 544 |
| 30.5.3 | Resets.....   | 544 |
| 30.5.4 | Low-Power mode operation.....                         | 544 |

## **Chapter 31 Timer/PWM Module (TPM)**

|        |                         |     |
|--------|-------------------------|-----|
| 31.1   | Introduction.....       | 547 |
| 31.1.1 | TPM Philosophy.....     | 547 |
| 31.1.2 | Features.....           | 547 |
| 31.1.3 | Modes of Operation..... | 548 |

| Section number | Title   | Page |
|----------------|---|------|
| 31.1.4         | Block Diagram.....                              | 548  |
| 31.2           | TPM Signal Descriptions.....                    | 549  |
| 31.2.1         | TPM_EXTCLK — TPM External Clock.....            | 549  |
| 31.2.2         | TPM_CHn — TPM Channel (n) I/O Pin.....          | 550  |
| 31.3           | Memory Map and Register Definition.....         | 550  |
| 31.3.1         | Status and Control (TPMx_SC).....               | 552  |
| 31.3.2         | Counter (TPMx_CNT).....                         | 553  |
| 31.3.3         | Modulo (TPMx_MOD).....                          | 554  |
| 31.3.4         | Channel (n) Status and Control (TPMx_CnSC)..... | 555  |
| 31.3.5         | Channel (n) Value (TPMx_CnV).....               | 557  |
| 31.3.6         | Capture and Compare Status (TPMx_STATUS).....   | 557  |
| 31.3.7         | Configuration (TPMx_CONF).....                  | 559  |
| 31.4           | Functional Description.....                     | 561  |
| 31.4.1         | Clock Domains.....                              | 561  |
| 31.4.2         | Prescaler.....                                  | 562  |
| 31.4.3         | Counter.....                                    | 562  |
| 31.4.4         | Input Capture Mode.....                         | 564  |
| 31.4.5         | Output Compare Mode.....                        | 565  |
| 31.4.6         | Edge-Aligned PWM (EPWM) Mode.....               | 566  |
| 31.4.7         | Center-Aligned PWM (CPWM) Mode.....             | 568  |
| 31.4.8         | Registers Updated from Write Buffers.....       | 570  |
| 31.4.9         | DMA.....  | 570  |
| 31.4.10        | Reset Overview.....                             | 571  |
| 31.4.11        | TPM Interrupts.....                             | 571  |

## Chapter 32 Periodic Interrupt Timer (PIT)

|        |                    |     |
|--------|--------------------|-----|
| 32.1   | Introduction.....  | 573 |
| 32.1.1 | Block diagram..... | 573 |
| 32.1.2 | Features.....      | 574 |

| <b>Section number</b> | <b>Title</b>   | <b>Page</b> |
|-----------------------|--|-------------|
| 32.2                  | Signal description.....                              | 574         |
| 32.3                  | Memory map/register description.....                 | 575         |
| 32.3.1                | PIT Module Control Register (PIT_MCR).....           | 575         |
| 32.3.2                | PIT Upper Lifetime Timer Register (PIT_LTMR64H)..... | 577         |
| 32.3.3                | PIT Lower Lifetime Timer Register (PIT_LTMR64L)..... | 577         |
| 32.3.4                | Timer Load Value Register (PIT_LDVAL $n$ ).....      | 578         |
| 32.3.5                | Current Timer Value Register (PIT_CVAL $n$ ).....    | 578         |
| 32.3.6                | Timer Control Register (PIT_TCTRL $n$ ).....         | 579         |
| 32.3.7                | Timer Flag Register (PIT_TFLG $n$ ).....             | 580         |
| 32.4                  | Functional description.....                          | 580         |
| 32.4.1                | General operation.....                               | 580         |
| 32.4.2                | Interrupts.....                                      | 582         |
| 32.4.3                | Chained timers.....                                  | 582         |
| 32.5                  | Initialization and application information.....      | 582         |
| 32.6                  | Example configuration for chained timers.....        | 583         |
| 32.7                  | Example configuration for the lifetime timer.....    | 584         |

## Chapter 33 Low-Power Timer (LPTMR)

|        |   |     |
|--------|---|-----|
| 33.1   | Introduction.....   | 587 |
| 33.1.1 | Features.....   | 587 |
| 33.1.2 | Modes of operation.....                                   | 587 |
| 33.2   | LPTMR signal descriptions.....                            | 588 |
| 33.2.1 | Detailed signal descriptions.....                         | 588 |
| 33.3   | Memory map and register definition.....                   | 588 |
| 33.3.1 | Low Power Timer Control Status Register (LPTMRx_CSR)..... | 589 |
| 33.3.2 | Low Power Timer Prescale Register (LPTMRx_PSR).....       | 590 |
| 33.3.3 | Low Power Timer Compare Register (LPTMRx_CMR).....        | 592 |
| 33.3.4 | Low Power Timer Counter Register (LPTMRx_CNR).....        | 592 |

| <b>Section number</b>                             | <b>Title</b>                                  | <b>Page</b> |
|---|---|-------------|
| 33.4  | Functional description.....                   | 593         |
| 33.4.1  | LPTMR power and reset.....                    | 593         |
| 33.4.2  | LPTMR clocking.....                           | 593         |
| 33.4.3  | LPTMR prescaler/glitch filter.....            | 593         |
| 33.4.4  | LPTMR compare.....                            | 595         |
| 33.4.5  | LPTMR counter.....                            | 595         |
| 33.4.6  | LPTMR hardware trigger.....                   | 596         |
| 33.4.7  | LPTMR interrupt.....                          | 596         |
| <b>Chapter 34</b><br><b>Real Time Clock (RTC)</b> |   |             |
| 34.1  | Introduction.....                             | 597         |
| 34.1.1  | Features.....                                 | 597         |
| 34.1.2  | Modes of operation.....                       | 597         |
| 34.1.3  | RTC Signal Descriptions.....                  | 597         |
| 34.2  | Register definition.....                      | 598         |
| 34.2.1  | RTC Time Seconds Register (RTC_TSR).....      | 599         |
| 34.2.2  | RTC Time Prescaler Register (RTC_TPR).....    | 599         |
| 34.2.3  | RTC Time Alarm Register (RTC_TAR).....        | 600         |
| 34.2.4  | RTC Time Compensation Register (RTC_TCR)..... | 600         |
| 34.2.5  | RTC Control Register (RTC_CR).....            | 601         |
| 34.2.6  | RTC Status Register (RTC_SR).....             | 603         |
| 34.2.7  | RTC Lock Register (RTC_LR).....               | 604         |
| 34.2.8  | RTC Interrupt Enable Register (RTC_IER).....  | 605         |
| 34.3  | Functional description.....                   | 606         |
| 34.3.1  | Power, clocking, and reset.....               | 606         |
| 34.3.2  | Time counter.....                             | 607         |
| 34.3.3  | Compensation.....                             | 607         |
| 34.3.4  | Time alarm.....                               | 608         |
| 34.3.5  | Update mode.....                              | 608         |

| <b>Section number</b>  | <b>Title</b>  | <b>Page</b> |
|--|---|-------------|
| 34.3.6   | Register lock.....                                      | 609         |
| 34.3.7   | Interrupt.....  | 609         |
| <b>Chapter 35</b><br><b>Universal Serial Bus OTG Controller (USBOTG)</b> |   |             |
| 35.1   | Introduction.....                                       | 611         |
| 35.1.1   | USB.....  | 611         |
| 35.1.2   | USB On-The-Go.....                                      | 612         |
| 35.1.3   | USB-FS Features.....                                    | 613         |
| 35.2   | Functional description.....                             | 613         |
| 35.2.1   | Data Structures.....                                    | 613         |
| 35.3   | Programmers interface.....                              | 614         |
| 35.3.1   | Buffer Descriptor Table.....                            | 614         |
| 35.3.2   | RX vs. TX as a USB target device or USB host.....       | 615         |
| 35.3.3   | Addressing BDT entries.....                             | 616         |
| 35.3.4   | Buffer Descriptors (BDs).....                           | 616         |
| 35.3.5   | USB transaction.....                                    | 619         |
| 35.4   | Memory map/Register definitions.....                    | 621         |
| 35.4.1   | Peripheral ID register (USBx_PERID).....                | 623         |
| 35.4.2   | Peripheral ID Complement register (USBx_IDCOMP).....    | 624         |
| 35.4.3   | Peripheral Revision register (USBx_REV).....            | 624         |
| 35.4.4   | Peripheral Additional Info register (USBx_ADDINFO)..... | 625         |
| 35.4.5   | OTG Interrupt Status register (USBx_OTGISTAT).....      | 625         |
| 35.4.6   | OTG Interrupt Control Register (USBx_OTGICR).....       | 626         |
| 35.4.7   | OTG Status register (USBx_OTGSTAT).....                 | 627         |
| 35.4.8   | OTG Control register (USBx_OTGCTL).....                 | 628         |
| 35.4.9   | Interrupt Status register (USBx_ISTAT).....             | 629         |
| 35.4.10  | Interrupt Enable register (USBx_INTEN).....             | 630         |
| 35.4.11  | Error Interrupt Status register (USBx_ERRSTAT).....     | 631         |
| 35.4.12  | Error Interrupt Enable register (USBx_ERREN).....       | 632         |

| <b>Section number</b> | <b>Title</b>   | <b>Page</b> |
|-----------------------|--|-------------|
| 35.4.13               | Status register (USB <sub>x</sub> _STAT).....                          | 633         |
| 35.4.14               | Control register (USB <sub>x</sub> _CTL).....                          | 634         |
| 35.4.15               | Address register (USB <sub>x</sub> _ADDR).....                         | 635         |
| 35.4.16               | BDT Page Register 1 (USB <sub>x</sub> _BDTPAGE1).....                  | 636         |
| 35.4.17               | Frame Number Register Low (USB <sub>x</sub> _FRMNUML).....             | 636         |
| 35.4.18               | Frame Number Register High (USB <sub>x</sub> _FRMNUMH).....            | 637         |
| 35.4.19               | Token register (USB <sub>x</sub> _TOKEN).....                          | 637         |
| 35.4.20               | SOF Threshold Register (USB <sub>x</sub> _SOFTHLD).....                | 638         |
| 35.4.21               | BDT Page Register 2 (USB <sub>x</sub> _BDTPAGE2).....                  | 639         |
| 35.4.22               | BDT Page Register 3 (USB <sub>x</sub> _BDTPAGE3).....                  | 639         |
| 35.4.23               | Endpoint Control register (USB <sub>x</sub> _ENDPT <sub>n</sub> )..... | 639         |
| 35.4.24               | USB Control register (USB <sub>x</sub> _USBCTRL).....                  | 640         |
| 35.4.25               | USB OTG Observe register (USB <sub>x</sub> _OBSERVE).....              | 641         |
| 35.4.26               | USB OTG Control register (USB <sub>x</sub> _CONTROL).....              | 642         |
| 35.4.27               | USB Transceiver Control Register 0 (USB <sub>x</sub> _USBTRC0).....    | 642         |
| 35.4.28               | Frame Adjust Register (USB <sub>x</sub> _USBFRMADJUST).....            | 643         |
| 35.5                  | OTG and Host mode operation.....                                       | 644         |
| 35.6                  | Host Mode Operation Examples.....                                      | 644         |
| 35.7                  | On-The-Go operation.....   | 647         |
| 35.7.1                | OTG dual role A device operation.....                                  | 648         |
| 35.7.2                | OTG dual role B device operation.....                                  | 649         |

## Chapter 36 USB Voltage Regulator

|        |   |     |
|--------|---|-----|
| 36.1   | Introduction.....                                     | 651 |
| 36.1.1 | Overview.....   | 651 |
| 36.1.2 | Features.....   | 652 |
| 36.1.3 | Modes of Operation.....                               | 653 |
| 36.2   | USB Voltage Regulator Module Signal Descriptions..... | 653 |

| Section number | Title   | Page |
|----------------|---|------|
|                | <b>Chapter 37<br/>Serial Peripheral Interface (SPI)</b> |      |
| 37.1           | Introduction.....                                       | 655  |
| 37.1.1         | Features.....   | 655  |
| 37.1.2         | Modes of Operation.....                                 | 656  |
| 37.1.3         | Block Diagrams.....                                     | 657  |
| 37.2           | External Signal Description.....                        | 659  |
| 37.2.1         | SPSCK — SPI Serial Clock.....                           | 659  |
| 37.2.2         | MOSI — Master Data Out, Slave Data In.....              | 660  |
| 37.2.3         | MISO — Master Data In, Slave Data Out.....              | 660  |
| 37.2.4         | SS — Slave Select.....                                  | 660  |
| 37.3           | Memory Map and Register Descriptions.....               | 661  |
| 37.3.1         | SPI control register 1 (SPIx_C1).....                   | 661  |
| 37.3.2         | SPI control register 2 (SPIx_C2).....                   | 663  |
| 37.3.3         | SPI baud rate register (SPIx_BR).....                   | 664  |
| 37.3.4         | SPI status register (SPIx_S).....                       | 665  |
| 37.3.5         | SPI data register (SPIx_D).....                         | 667  |
| 37.3.6         | SPI match register (SPIx_M).....                        | 668  |
| 37.4           | Functional Description.....                             | 668  |
| 37.4.1         | General.....  | 668  |
| 37.4.2         | Master Mode.....  | 669  |
| 37.4.3         | Slave Mode.....   | 670  |
| 37.4.4         | SPI Transmission by DMA.....                            | 671  |
| 37.4.5         | SPI Clock Formats.....                                  | 673  |
| 37.4.6         | SPI Baud Rate Generation.....                           | 676  |
| 37.4.7         | Special Features.....                                   | 677  |
| 37.4.8         | Error Conditions.....                                   | 678  |
| 37.4.9         | Low Power Mode Options.....                             | 679  |
| 37.4.10        | Reset.....  | 681  |

| <b>Section number</b>                                      | <b>Title</b>  | <b>Page</b> |
|--|---|-------------|
| 37.4.11  | Interrupts.....   | 681         |
| 37.5   | Initialization/Application Information.....                   | 683         |
| 37.5.1   | Initialization Sequence.....                                  | 683         |
| 37.5.2   | Pseudo-Code Example.....                                      | 684         |
| <b>Chapter 38</b><br><b>Inter-Integrated Circuit (I2C)</b> |   |             |
| 38.1   | Introduction.....   | 687         |
| 38.1.1   | Features.....   | 687         |
| 38.1.2   | Modes of operation.....                                       | 688         |
| 38.1.3   | Block diagram.....  | 688         |
| 38.2   | I2C signal descriptions.....                                  | 689         |
| 38.3   | Memory map and register descriptions.....                     | 689         |
| 38.3.1   | I2C Address Register 1 (I2Cx_A1).....                         | 690         |
| 38.3.2   | I2C Frequency Divider register (I2Cx_F).....                  | 691         |
| 38.3.3   | I2C Control Register 1 (I2Cx_C1).....                         | 692         |
| 38.3.4   | I2C Status register (I2Cx_S).....                             | 694         |
| 38.3.5   | I2C Data I/O register (I2Cx_D).....                           | 695         |
| 38.3.6   | I2C Control Register 2 (I2Cx_C2).....                         | 696         |
| 38.3.7   | I2C Programmable Input Glitch Filter register (I2Cx_FLT)..... | 697         |
| 38.3.8   | I2C Range Address register (I2Cx_RA).....                     | 698         |
| 38.3.9   | I2C SMBus Control and Status register (I2Cx_SMB).....         | 699         |
| 38.3.10  | I2C Address Register 2 (I2Cx_A2).....                         | 701         |
| 38.3.11  | I2C SCL Low Timeout Register High (I2Cx_SLTH).....            | 701         |
| 38.3.12  | I2C SCL Low Timeout Register Low (I2Cx_SLTL).....             | 701         |
| 38.4   | Functional description.....                                   | 702         |
| 38.4.1   | I2C protocol.....   | 702         |
| 38.4.2   | 10-bit address.....   | 707         |
| 38.4.3   | Address matching.....   | 709         |

| <b>Section number</b> | <b>Title</b>                                | <b>Page</b> |
|-----------------------|---|-------------|
| 38.4.4                | System management bus specification.....    | 709         |
| 38.4.5                | Resets.....                                 | 712         |
| 38.4.6                | Interrupts.....                             | 712         |
| 38.4.7                | Programmable input glitch filter.....       | 714         |
| 38.4.8                | Address matching wakeup.....                | 715         |
| 38.4.9                | DMA support.....                            | 715         |
| 38.5                  | Initialization/application information..... | 716         |

## **Chapter 39 Universal Asynchronous Receiver/Transmitter (UART0)**

|         |  |     |
|---------|--|-----|
| 39.1    | Introduction.....  | 721 |
| 39.1.1  | Features.....  | 721 |
| 39.1.2  | Modes of operation.....                                      | 722 |
| 39.1.3  | Block diagram.....   | 722 |
| 39.2    | Register definition.....                                     | 724 |
| 39.2.1  | UART Baud Rate Register High (UART <sub>x</sub> _BDH).....   | 725 |
| 39.2.2  | UART Baud Rate Register Low (UART <sub>x</sub> _BDL).....    | 726 |
| 39.2.3  | UART Control Register 1 (UART <sub>x</sub> _C1).....         | 726 |
| 39.2.4  | UART Control Register 2 (UART <sub>x</sub> _C2).....         | 728 |
| 39.2.5  | UART Status Register 1 (UART <sub>x</sub> _S1).....          | 729 |
| 39.2.6  | UART Status Register 2 (UART <sub>x</sub> _S2).....          | 731 |
| 39.2.7  | UART Control Register 3 (UART <sub>x</sub> _C3).....         | 733 |
| 39.2.8  | UART Data Register (UART <sub>x</sub> _D).....               | 734 |
| 39.2.9  | UART Match Address Registers 1 (UART <sub>x</sub> _MA1)..... | 735 |
| 39.2.10 | UART Match Address Registers 2 (UART <sub>x</sub> _MA2)..... | 736 |
| 39.2.11 | UART Control Register 4 (UART <sub>x</sub> _C4).....         | 736 |
| 39.2.12 | UART Control Register 5 (UART <sub>x</sub> _C5).....         | 737 |
| 39.3    | Functional description.....                                  | 738 |
| 39.3.1  | Baud rate generation.....                                    | 738 |

| Section number | Title                                   | Page |
|----------------|---|------|
| 39.3.2         | Transmitter functional description..... | 738  |
| 39.3.3         | Receiver functional description.....    | 740  |
| 39.3.4         | Additional UART functions.....          | 743  |
| 39.3.5         | Interrupts and status flags.....        | 745  |

## Chapter 40 Universal Asynchronous Receiver/Transmitter (UART1 and UART2)

|        |   |     |
|--------|---|-----|
| 40.1   | Introduction.....   | 747 |
| 40.1.1 | Features.....   | 747 |
| 40.1.2 | Modes of operation.....                                     | 747 |
| 40.1.3 | Block diagram.....  | 748 |
| 40.2   | Register definition.....                                    | 750 |
| 40.2.1 | UART Baud Rate Register: High (UART <sub>x</sub> _BDH)..... | 751 |
| 40.2.2 | UART Baud Rate Register: Low (UART <sub>x</sub> _BDL).....  | 751 |
| 40.2.3 | UART Control Register 1 (UART <sub>x</sub> _C1).....        | 752 |
| 40.2.4 | UART Control Register 2 (UART <sub>x</sub> _C2).....        | 753 |
| 40.2.5 | UART Status Register 1 (UART <sub>x</sub> _S1).....         | 755 |
| 40.2.6 | UART Status Register 2 (UART <sub>x</sub> _S2).....         | 756 |
| 40.2.7 | UART Control Register 3 (UART <sub>x</sub> _C3).....        | 758 |
| 40.2.8 | UART Data Register (UART <sub>x</sub> _D).....              | 760 |
| 40.2.9 | UART Control Register 4 (UART <sub>x</sub> _C4).....        | 760 |
| 40.3   | Functional description.....                                 | 761 |
| 40.3.1 | Baud rate generation.....                                   | 761 |
| 40.3.2 | Transmitter functional description.....                     | 762 |
| 40.3.3 | Receiver functional description.....                        | 764 |
| 40.3.4 | Interrupts and status flags.....                            | 767 |
| 40.3.5 | DMA Operation.....  | 768 |

| Section number  | Title   | Page |
|---|---|------|
| 40.3.6  | Additional UART functions.....                  | 769  |
| <b>Chapter 41</b><br><b>General-Purpose Input/Output (GPIO)</b> |   |      |
| 41.1  | Introduction.....                               | 771  |
| 41.1.1  | Features.....                                   | 771  |
| 41.1.2  | Modes of operation.....                         | 771  |
| 41.1.3  | GPIO signal descriptions.....                   | 772  |
| 41.2  | Memory map and register definition.....         | 773  |
| 41.2.1  | Port Data Output Register (GPIOx_PDOR).....     | 775  |
| 41.2.2  | Port Set Output Register (GPIOx_PSOR).....      | 776  |
| 41.2.3  | Port Clear Output Register (GPIOx_PCOR).....    | 776  |
| 41.2.4  | Port Toggle Output Register (GPIOx_PTOR).....   | 777  |
| 41.2.5  | Port Data Input Register (GPIOx_PDIR).....      | 777  |
| 41.2.6  | Port Data Direction Register (GPIOx_PDDR).....  | 778  |
| 41.3  | FGPIO memory map and register definition.....   | 778  |
| 41.3.1  | Port Data Output Register (FGPIOx_PDOR).....    | 780  |
| 41.3.2  | Port Set Output Register (FGPIOx_PSOR).....     | 781  |
| 41.3.3  | Port Clear Output Register (FGPIOx_PCOR).....   | 781  |
| 41.3.4  | Port Toggle Output Register (FGPIOx_PTOR).....  | 782  |
| 41.3.5  | Port Data Input Register (FGPIOx_PDIR).....     | 782  |
| 41.3.6  | Port Data Direction Register (FGPIOx_PDDR)..... | 783  |
| 41.4  | Functional description.....                     | 783  |
| 41.4.1  | General-purpose input.....                      | 783  |
| 41.4.2  | General-purpose output.....                     | 783  |
| 41.4.3  | IOPORT.....                                     | 784  |
| <b>Chapter 42</b><br><b>Touch Sensing Input (TSI)</b>           |   |      |
| 42.1  | Introduction.....                               | 785  |
| 42.1.1  | Features.....                                   | 785  |

| <b>Section number</b> | <b>Title</b>  | <b>Page</b> |
|-----------------------|---|-------------|
| 42.1.2                | Modes of operation.....                                   | 785         |
| 42.1.3                | Block diagram.....  | 786         |
| 42.2                  | External signal description.....                          | 787         |
| 42.2.1                | TSI[15:0].....  | 787         |
| 42.3                  | Register definition.....                                  | 787         |
| 42.3.1                | TSI General Control and Status Register (TSIx_GENCS)..... | 787         |
| 42.3.2                | TSI DATA Register (TSIx_DATA).....                        | 792         |
| 42.3.3                | TSI Threshold Register (TSIx_TSHD).....                   | 793         |
| 42.4                  | Functional description.....                               | 793         |
| 42.4.1                | Capacitance measurement.....                              | 794         |
| 42.4.2                | TSI measurement result.....                               | 797         |
| 42.4.3                | Enable TSI module.....                                    | 797         |
| 42.4.4                | Software and hardware trigger.....                        | 797         |
| 42.4.5                | Scan times.....   | 798         |
| 42.4.6                | Clock setting.....  | 798         |
| 42.4.7                | Reference voltage.....                                    | 798         |
| 42.4.8                | Current source.....                                       | 799         |
| 42.4.9                | End of scan.....  | 799         |
| 42.4.10               | Out-of-range interrupt.....                               | 799         |
| 42.4.11               | Wake up MCU from low power modes.....                     | 800         |
| 42.4.12               | DMA function support.....                                 | 800         |
| 42.4.13               | Noise detection mode.....                                 | 800         |

# Chapter 1

## About This Document

### 1.1 Overview

#### 1.1.1 Purpose

This document describes the features, architecture, and programming model of the Freescale KL25 microcontroller.

#### 1.1.2 Audience

This document is primarily for system architects and software application developers who are using or considering using the KL25 microcontroller in a system.

### 1.2 Conventions

#### 1.2.1 Numbering systems

The following suffixes identify different numbering systems:

| This suffix | Identifies a   |
|-------------|--|
| b           | Binary number. For example, the binary equivalent of the number 5 is written 101b. In some cases, binary numbers are shown with the prefix 0b.                     |
| d           | Decimal number. Decimal numbers are followed by this suffix only when the possibility of confusion exists. In general, decimal numbers are shown without a suffix. |
| h           | Hexadecimal number. For example, the hexadecimal equivalent of the number 60 is written 3Ch. In some cases, hexadecimal numbers are shown with the prefix 0x.      |

## 1.2.2 Typographic notation

The following typographic notation is used throughout this document:

| Example                | Description  |
|------------------------|--|
| <i>placeholder</i> , x | Items in italics are placeholders for information that you provide. Italicized text is also used for the titles of publications and for emphasis. Plain lowercase letters are also used as placeholders for single letters and numbers.  |
| code                   | Fixed-width type indicates text that must be typed exactly as shown. It is used for instruction mnemonics, directives, symbols, subcommands, parameters, and operators. Fixed-width type is also used for example code. Instruction mnemonics and directives in text and tables are shown in all caps; for example, BSR.   |
| SR[SCM]                | A mnemonic in brackets represents a named field in a register. This example refers to the Scaling Mode (SCM) field in the Status Register (SR).  |
| REVNO[6:4], XAD[7:0]   | Numbers in brackets and separated by a colon represent either: <ul style="list-style-type: none"> <li>• A subset of a register's named field<br/>For example, REVNO[6:4] refers to bits 6–4 that are part of the COREREV field that occupies bits 6–0 of the REVNO register.</li> <li>• A continuous range of individual signals of a bus<br/>For example, XAD[7:0] refers to signals 7–0 of the XAD bus.</li> </ul> |

## 1.2.3 Special terms

The following terms have special meanings:

| Term       | Meaning  |
|------------|--|
| asserted   | Refers to the state of a signal as follows: <ul style="list-style-type: none"> <li>• An active-high signal is asserted when high (1).</li> <li>• An active-low signal is asserted when low (0).</li> </ul>   |
| deasserted | Refers to the state of a signal as follows: <ul style="list-style-type: none"> <li>• An active-high signal is deasserted when low (0).</li> <li>• An active-low signal is deasserted when high (1).</li> </ul> In some cases, deasserted signals are described as <i>negated</i> . |
| reserved   | Refers to a memory space, register, or field that is either reserved for future use or for which, when written to, the module or chip behavior is unpredictable.   |

# **Chapter 2**

## **Introduction**

### **2.1 Overview**

This chapter provides an overview of the Kinetis L series of ARM® Cortex™-M0+ MCUs and KL25 product family. It also presents high-level descriptions of the modules available on the devices covered by this document.

### **2.2 Kinetis L Series**

The Kinetis L series is the most scalable portfolio of ultra low-power, mixed-signal ARM Cortex-M0+ MCUs in the industry. The portfolio includes 5 MCU families that offer a broad range of memory, peripheral and package options. Kinetis L Series families share common peripherals and pin-counts allowing developers to migrate easily within an MCU family or between MCU families to take advantage of more memory or feature integration. This scalability allows developers to standardize on the Kinetis L Series for their end product platforms, maximising hardware and software reuse and reducing time-to-market.

Features common to all Kinetis L series families include:

- 48 MHz ARM Cortex-M0+ core
- High-speed 12/16-bit analog-to-digital converters
- 12-bit digital-to-analog converters for all series except for KLx4/KLx2 family
- High-speed analog comparators
- Low-power touch sensing with wake-up on touch from reduced power states for all series except for KLx4 family
- Powerful timers for a broad range of applications including motor control
- Low power focused serial communication interfaces such as low power UART, SPI, I2C etc.
- Single power supply: 1.71V - 3.6V with multiple low-power modes support single operation temperature: -40 ~ 105 °C (exclude CSP package)

Kinetis L series MCU families combine the latest low-power innovations with precision mixed-signal capability and a broad range of communication, connectivity, and human-machine interface peripherals. Each MCU family is supported by a market-leading enablement bundle from Freescale and numerous ARM 3rd party ecosystem partners. The KL0x family is the entry-point to the Kinetis L series and is pin compatible with the 8-bit S08PT family. The KL1x/2x/3x/4x families are compatible with each other and their equivalent ARM Cortex-M4 Kinetis K series families - K10/20/30/40.

| Family      | Program Flash | Packages | Key Features |
|-------------|---------------|----------|--------------|
| KL3x Family |               |          |              |
| KL2x Family |               |          |              |
| KL1x Family |               |          |              |
| KL0x Family |               |          |              |



Low power

- Optimized access to program memory: Accesses on alternate cycles reduces power consumption
- 100 percent compatible with ARM Cortex-M0 and a subset ARM Cortex-M3/M4: Reuse existing compilers and debug tools
- Simplified architecture: 56 instructions and 17 registers enables easy programming and efficient packaging of 8/16/32-bit data in memory
- Linear 4 GB address space removes the need for paging/banking, reducing software complexity
- ARM third-party ecosystem support: Software and tools to help minimize development time/cost
- Micro Trace Buffer: Lightweight trace solution allows fast bug identification and correction
- BME: Bit manipulation engine reduces code size and cycles for bit oriented operations to peripheral registers eliminating traditional methods where the core would need to perform read-modify-write operations.
- Up to 4-channel DMA for peripheral and memory servicing with minimal CPU intervention (feature not available on KL02 family)
- Ultra low-power:
  - Extreme dynamic efficiency: 32-bit ARM Cortex-M0+ core combined with Freescale 90 nm thin film storage flash technology delivers 50% energy savings per Coremark versus the closest 8/16-bit competitive solution
  - Multiple flexible low-power modes, including new operation clocking option which reduces dynamic power by shutting off bus and system clocks for lowest power core processing. Peripherals with an alternate asynchronous clock source can continue operation.
  - UART, SPI, I2C, ADC, DAC, TPM, LPT, and DMA support low-power mode operation without waking up the core
- Memory:
  - Scalable memory footprints from 8 KB flash / 1 KB SRAM to 256 KB flash / 32 KB SRAM
  - Embedded 64 B cache memory for optimizing bus bandwidth and flash execution performance (32 B cache on KL02 family)
- Mixed-signal analog:
  - Fast, high precision 16-, or 12-bit ADC with optional differential pairs, 12-bit DAC, high speed comparators. Powerful signal conditioning, conversion and analysis capability with reduced system cost (12-bit DAC not available on KL02 family)
- Human Machine Interface (HMI):
  - Optional capacitive Touch Sensing Interface with full low-power support and minimal current adder when enabled
  - Segment LCD controller

- Connectivity and Communications:
  - Up to three UARTs, all UARTs support DMA transfers, and can trigger when data on bus is detected, UART0 supports 4x to 32x over sampling ratio.  
Asynchronous transmit and receive operation for operating in STOP/VLPS modes.
  - Up to two SPIs
  - Up to two I<sup>2</sup>Cs
  - Full-speed USB OTG controller with on-chip transceiver
  - 5 V to 3.3 V USB on-chip regulator
  - One I<sup>2</sup>S
- Reliability, Safety and Security:
  - Internal watchdog with independent clock source
- Timing and Control:
  - Powerful timer modules which support general purpose, PWM, and motor control functions
  - Periodic Interrupt Timer for RTOS task scheduler time base or trigger source for ADC conversion and timer modules
- System:
  - GPIO with pin interrupt functionality
  - Wide operating voltage range from 1.71 V to 3.6 V with flash programmable down to 1.71 V with fully functional flash and analog peripherals
  - Ambient operating temperature ranges from -40 °C to 105 °C

## **2.3 KL25 Sub-Family Introduction**

The device is highly-integrated, market leading ultra low power 32-bit microcontroller based on the enhanced Cortex-M0+ (CM0+) core platform. The family derivatives feature:

- Core platform clock up to 48 MHz, bus clock up to 24 MHz
- Memory option is up to 128 KB Flash and 16 KB RAM
- Wide operating voltage ranges from 1.71V to 3.6V with full functional Flash program/erase/read operations
- Multiple package options from 32-pin to 80-pin
- Ambient operating temperature ranges from -40 °C to 105 °C

The family acts as an ultra low power, cost effective microcontroller to provide developers an appropriate entry-level 32-bit solution. The family is next generation MCU solution for low cost, low power, high performance devices applications. It's valuable for cost-sensitive, portable applications requiring long battery life-time.

## 2.4 Module functional categories

The modules on this device are grouped into functional categories. The following sections describe the modules assigned to each category in more detail.

**Table 2-1. Module functional categories**

| Module category                | Description  |
|--------------------------------|--|
| ARM Cortex-M0+ core            | <ul style="list-style-type: none"> <li>32-bit MCU core from ARM's Cortex-M class, 1.77 CoreMark®/MHz from single-cycle access memories, 48 MHz CPU frequency</li> </ul>  |
| System                         | <ul style="list-style-type: none"> <li>System integration module</li> <li>Power management and mode controllers <ul style="list-style-type: none"> <li>Multiple power modes available based on run, wait, stop, and power-down modes</li> </ul> </li> <li>Miscellaneous control module</li> <li>Low-leakage wakeup unit</li> <li>Peripheral bridge</li> <li>Direct memory access (DMA) controller with multiplexer to increase available DMA requests</li> <li>COP watchdog</li> </ul> |
| Memories                       | <ul style="list-style-type: none"> <li>Internal memories include: <ul style="list-style-type: none"> <li>Up to 128 KB flash memory</li> <li>up to 16 KB SRAM</li> </ul> </li> </ul>  |
| Clocks                         | <ul style="list-style-type: none"> <li>Multiple clock generation options available from internally- and externally-generated clocks <ul style="list-style-type: none"> <li>MCG module with FLL and PLL for systems and CPU clock sources</li> <li>Low power 1 kHz RC oscillator for RTC and COP watchdog</li> </ul> </li> <li>System oscillator to provide clock source for the MCU</li> </ul>   |
| Security                       | <ul style="list-style-type: none"> <li>COP watchdog timer (COP)</li> </ul>   |
| Analog                         | <ul style="list-style-type: none"> <li>16-bit analog-to-digital converters with DMA supported and four muxed differential pairs</li> <li>Comparator (CMP) with internal 6-bit digital-to-analog converter (DAC)</li> <li>12-bit DAC with DMA support and two 16-bit data buffer</li> </ul>   |
| Timers                         | <ul style="list-style-type: none"> <li>One 6-channel TPM</li> <li>Two 2-channel TPMs</li> <li>2-channel periodic interrupt timer</li> <li>Real time clock</li> <li>Low-power timer</li> <li>System tick timer</li> </ul>   |
| Communications                 | <ul style="list-style-type: none"> <li>Two 8-bit serial peripheral interface</li> <li>USB OTG controller with built-in FS/LS transceiver</li> <li>USB voltage regulator</li> <li>Two inter-integrated circuit (I<sup>2</sup>C) modules</li> <li>One low power UART module and two UART modules</li> </ul>  |
| Human-Machine Interfaces (HMI) | <ul style="list-style-type: none"> <li>General purpose input/output controller</li> <li>Capacitive touch sense input interface enabled in hardware</li> </ul>  |

## 2.4.1 ARM® Cortex™-M0+ Core Modules

The following core modules are available on this device.

**Table 2-2. Core modules**

| Module                | Description   |
|-----------------------|---|
| ARM® Cortex™-M0+      | The ARM® Cortex™-M0+ is the newest member of the Cortex M Series of processors targeting microcontroller applications focused on very cost sensitive, deterministic, interrupt driven environments. The Cortex M0+ processor is based on the ARMv6 Architecture and Thumb®-2 ISA and is 100% instruction set compatible with its predecessor, the Cortex-M0 core, and upward compatible to Cortex-M3 and M4 cores.  |
| NVIC                  | The ARMv6-M exception model and nested-vectored interrupt controller (NVIC) implement a relocatable vector table supporting many external interrupts, a single non-maskable interrupt (NMI), and priority levels.<br><br>The NVIC replaces shadow registers with equivalent system and simplified programmability. The NVIC contains the address of the function to execute for a particular handler. The address is fetched via the instruction port allowing parallel register stacking and look-up. The first sixteen entries are allocated to ARM internal sources with the others mapping to MCU-defined interrupts. |
| AWIC                  | The primary function of the Asynchronous Wake-up Interrupt Controller (AWIC) is to detect asynchronous wake-up events in stop modes and signal to clock control logic to resume system clocking. After clock restart, the NVIC observes the pending interrupt and performs the normal interrupt or event processing.  |
| Single-cycle I/O Port | For high-speed, single-cycle access to peripherals, the Cortex-M0+ processor implements a dedicated single-cycle I/O port.  |
| Debug interfaces      | Most of this device's debug is based on the ARM CoreSight™ architecture. One debug interface is supported: <ul style="list-style-type: none"> <li>Serial Wire Debug (SWD)</li> </ul>  |

## 2.4.2 System Modules

The following system modules are available on this device.

**Table 2-3. System modules**

| Module                             | Description   |
|------------------------------------|---|
| System integration module (SIM)    | The SIM includes integration logic and several module configuration settings.   |
| System mode controller             | The SMC provides control and protection on entry and exit to each power mode, control for the Power management controller (PMC), and reset entry and exit for the complete MCU.   |
| Power management controller (PMC)  | The PMC provides the user with multiple power options. Multiple modes are supported that allow the user to optimize power consumption for the level of functionality needed. Includes power-on-reset (POR) and integrated low voltage detect (LVD) with reset (brownout) capability and selectable LVD trip points. |
| Miscellaneous control module (MCM) | The MCM includes integration logic and details.   |

*Table continues on the next page...*

**Table 2-3. System modules (continued)**

| Module                                      | Description   |
|---|---|
| Crossbar switch (XBS)                       | The XBS connects bus masters and bus slaves, allowing all bus masters to access different bus slaves simultaneously and providing arbitration among the bus masters when they access the same slave.                                      |
| Low-leakage wakeup unit (LLWU)              | The LLWU module allows the device to wake from low leakage power modes (LLS and VLSS) through various internal peripheral and external pin sources.   |
| Peripheral bridge                           | The peripheral bridge converts the crossbar switch interface to an interface to access a majority of peripherals on the device.   |
| DMA multiplexer (DMAMUX)                    | The DMA multiplexer selects from many DMA requests down to 4 for the DMA controller.  |
| Direct memory access (DMA) controller       | The DMA controller provides programmable channels with transfer control descriptors for data movement via dual-address transfers for 8-, 16- and 32-bit data values.  |
| Computer operating properly watchdog (WDOG) | The WDOG monitors internal system operation and forces a reset in case of failure. It can run from an independent 1 kHz low power oscillator with a programmable refresh window to detect deviations in program flow or system frequency. |

## 2.4.3 Memories and Memory Interfaces

The following memories and memory interfaces are available on this device.

**Table 2-4. Memories and memory interfaces**

| Module                  | Description  |
|-------------------------|--|
| Flash memory            | Program flash memory — up to 128 KB of the non-volatile flash memory that can execute program code |
| Flash memory controller | Manages the interface between the device and the on-chip flash memory.                             |
| SRAM                    | Up to 16 KB internal system RAM.   |

## 2.4.4 Clocks

The following clock modules are available on this device.

**Table 2-5. Clock modules**

| Module                             | Description  |
|------------------------------------|--|
| Multipurpose Clock Generator (MCG) | MCG module containing a frequency-locked-loop (FLL) and phase-locked-loop (PLL) controlled by internal or external reference oscillator. |
| System oscillator                  | The system oscillator, in conjunction with an external crystal or resonator, generates a reference clock for the MCU.                    |

## 2.4.5 Security and Integrity modules

The following security and integrity modules are available on this device:

**Table 2-6. Security and integrity modules**

| Module                | Description  |
|-----------------------|--|
| Watchdog Timer (WDOG) | Watchdog Timer keeps a watch on the system functioning and resets it in case of its failure. |

## 2.4.6 Analog modules

The following analog modules are available on this device:

**Table 2-7. Analog modules**

| Module                                    | Description  |
|---|--|
| Analog-to-digital converters (ADC)        | 16-bit successive-approximation ADC module.  |
| Analog comparators                        | One comparator that compares two analog input voltages across the full range of the supply voltage and can trigger an ADC acquisition, TPM update, or CPU interrupt. |
| 6-bit digital-to-analog converters (DAC)  | 64-tap resistor ladder network which provides a selectable voltage reference for comparator.   |
| 12-bit digital-to-analog converters (DAC) | Low-power general-purpose DAC, whose output can be placed on an external pin or set as one of the inputs to the analog comparator or ADC.                            |

## 2.4.7 Timer modules

The following timer modules are available on this device:

**Table 2-8. Timer modules**

| Module                          | Description   |
|---------------------------------|---|
| Timer/PWM module (TPM)          | <ul style="list-style-type: none"> <li>Selectable TPM clock mode</li> <li>Prescaler divide-by 1, 2, 4, 8, 16, 32, 64, or 128</li> <li>16-bit free-running counter or modulo counter with counting be up or up-down</li> <li>Six configurable channels for input capture, output compare, or edge-aligned PWM mode</li> <li>Support the generation of an interrupt and/or DMA request per channel</li> <li>Support the generation of an interrupt and/or DMA request when the counter overflows</li> <li>Support selectable trigger input to optionally reset or cause the counter to start incrementing.</li> <li>Support the generation of hardware triggers when the counter overflows and per channel</li> </ul> |
| Periodic interrupt timers (PIT) | <ul style="list-style-type: none"> <li>One general purpose interrupt timer</li> <li>Interrupt timers for triggering ADC conversions</li> <li>32-bit counter resolution</li> <li>Clocked by bus clock frequency</li> <li>DMA support</li> </ul>  |
| Low power timer (LPTMR)         | <ul style="list-style-type: none"> <li>16-bit time counter or pulse counter with compare</li> <li>Configurable clock source for prescaler/glitch filter</li> <li>Configurable input source for pulse counter</li> </ul>   |
| Real-time counter (RTC)         | <ul style="list-style-type: none"> <li>16-bit up-counter           <ul style="list-style-type: none"> <li>16-bit modulo match limit</li> <li>Software controllable periodic interrupt on match</li> </ul> </li> <li>Software selectable clock sources for input to prescaler with programmable 16-bit prescaler           <ul style="list-style-type: none"> <li>XOSC 32.678 kHz nominal</li> <li>LPO (~1 kHz)</li> <li>External RTC_CLKIN</li> </ul> </li> </ul>   |

## 2.4.8 Communication interfaces

The following communication interfaces are available on this device:

**Table 2-9. Communication modules**

| Module  | Description  |
|---|--|
| USB OTG (low-/full-speed)                           | USB 2.0 compliant module with support for host, device, and On-The-Go modes. Includes an on-chip transceiver for full and low speeds.  |
| USB voltage regulator                               | Up to 5 V regulator input typically provided by USB VBUS power with 3.3 V regulated output that powers on-chip USB subsystem, capable of sourcing 120 mA to external board components. |
| Serial peripheral interface (SPI)                   | Synchronous serial bus for communication to an external device   |
| Inter-integrated circuit (I2C)                      | Allows communication between a number of devices. Also supports the System Management Bus (SMBus) Specification, version 2.  |
| Universal asynchronous receiver/transmitters (UART) | One low power UART module that retains functional in stop modes. Two UART modules.   |

## 2.4.9 Human-machine interfaces

The following human-machine interfaces (HMI) are available on this device:

**Table 2-10. HMI modules**

| Module                              | Description   |
|-------------------------------------|---|
| General purpose input/output (GPIO) | Some general purpose input or output (GPIO) pins are capable of interrupt and DMA request generation.                                 |
| Capacitive touch sense input (TSI)  | Contains up to 16 channel inputs for capacitive touch sensing applications. Operation is available in low-power modes via interrupts. |

## 2.5 Orderable part numbers

The following table summarizes the part numbers of the devices covered by this document.

**Table 2-11. Orderable part numbers summary**

| Freescale part number | CPU frequency | Pin count | Package | Total flash memory | RAM   | Temperature range |
|-----------------------|---------------|-----------|---------|--------------------|-------|-------------------|
| MKL25Z32VFM4          | 48 MHz        | 32        | QFN     | 32 KB              | 4 KB  | -40 to 105 °C     |
| MKL25Z64VFM4          | 48 MHz        | 32        | QFN     | 64 KB              | 8 KB  | -40 to 105 °C     |
| MKL25Z128VFM4         | 48 MHz        | 32        | QFN     | 128 KB             | 16 KB | -40 to 105 °C     |
| MKL25Z32VFT4          | 48 MHz        | 48        | QFN     | 32 KB              | 4 KB  | -40 to 105 °C     |
| MKL25Z64VFT4          | 48 MHz        | 48        | QFN     | 64 KB              | 8 KB  | -40 to 105 °C     |
| MKL25Z128VFT4         | 48 MHz        | 48        | QFN     | 128 KB             | 16 KB | -40 to 105 °C     |
| MKL25Z32VLH4          | 48 MHz        | 64        | LQFP    | 32 KB              | 4 KB  | -40 to 105 °C     |
| MKL25Z64VLH4          | 48 MHz        | 64        | LQFP    | 64 KB              | 8 KB  | -40 to 105 °C     |
| MKL25Z128VLH4         | 48 MHz        | 64        | LQFP    | 128 KB             | 16 KB | -40 to 105 °C     |
| MKL25Z32VLK4          | 48 MHz        | 80        | LQFP    | 32 KB              | 4 KB  | -40 to 105 °C     |
| MKL25Z64VLK4          | 48 MHz        | 80        | LQFP    | 64 KB              | 8 KB  | -40 to 105 °C     |
| MKL25Z128VLK4         | 48 MHz        | 80        | LQFP    | 128 KB             | 16 KB | -40 to 105 °C     |

# Chapter 3

## Chip Configuration

### 3.1 Introduction

This chapter provides details on the individual modules of the microcontroller. It includes:

- Module block diagrams showing immediate connections within the device
- Specific module-to-module interactions not necessarily discussed in the individual module chapters
- Links for more information

### Module to Module Interconnects

#### 3.2.1 Module to Module Interconnects

The below table captures the Module to module interconnections for this device.

**Table 3-1. Module to Module Interconnects**

| Peripheral | Signal     | —  | to Peripheral | Use Case                    | Control                   | Comment  |
|------------|------------|----|---------------|-----------------------------|---------------------------|--|
| TPM1       | CH0F, CH1F | to | ADC (Trigger) | ADC Triggering<br>(A AND B) | SOPT7_ADCAL<br>TTRGEN = 0 | Ch0 is A, and<br>Ch1 is B,<br>selecting this<br>ADC trigger is<br>for supporting A<br>and B triggering.<br>In Stop and<br>VLPS modes,<br>the second<br>trigger must be<br>set to >10us<br>after the first<br>trigger |

*Table continues on the next page...*

**Module to Module Interconnects**
**Table 3-1. Module to Module Interconnects (continued)**

| Peripheral | Signal           | —  | to Peripheral | Use Case                               | Control  | Comment |
|------------|------------------|----|---------------|--|--|---------|
| LPTMR      | Hardware trigger | to | ADC (Trigger) | ADC Triggering (A or B)                | SOPT7_ADC0T RGSEL (4 bit field), ADC0PRETRG SEL to select A or B       | —       |
| TPMx       | TOF              | to | ADC (Trigger) | ADC Triggering (A or B)                | SOPT7_ADC0T RGSEL (4 bit field), SOPT7_ADC0P RETRGSEL to select A or B | —       |
| PIT CHx    | TIF0, TIF1       | to | ADC (Trigger) | ADC Triggering (A or B)                | SOPT7_ADC0T RGSEL (4 bit field), ADC0PRETRG SEL to select A or B       | —       |
| RTC        | ALARM or SECONDS | to | ADC (Trigger) | ADC Triggering (A or B)                | SOPT7_ADC0T RGSEL (4 bit field) ADC0PRETRG SEL to select A or B        | —       |
| EXTRG_IN   | EXTRG_IN         | to | ADC (Trigger) | ADC Triggering (A or B)                | SOPT7_ADC0T RGSEL (4 bit field) ADC0PRETRG SEL to select A or B        | —       |
| CMP0       | CMP0_OUT         | to | ADC (Trigger) | ADC Triggering (A or B)                | SOPT7_ADC0T RGSEL (4 bit field) ADC0PRETRG SEL to select A or B        | —       |
| CMP0       | CMP0_OUT         | to | LPTMR_ALTO    | Count CMP events                       | LPTMR_CSR[T PS]  | —       |
| CMP0       | CMP0_OUT         | to | TPM1 CH0      | Input capture                          | SOPT4 TPM1C H0SRC  | —       |
| CMP0       | CMP0_OUT         | to | TPM2 CH0      | Input capture                          | SOPT4 TPM2C H0SRC  | —       |
| CMP0       | CMP0_OUT         | to | UART0_RX      | IR interface                           | SOPT5_UART0 RXSRC  | —       |
| CMP0       | CMP0_OUT         | to | UART1_RX      | IR Interface                           | SOPT5_UART1 SRC  | —       |
| LPTMR      | Hardware trigger | to | CMPx          | Low power triggering of the comparator | CMP_CR1[TRIG M]  | —       |

*Table continues on the next page...*

**Table 3-1. Module to Module Interconnects (continued)**

| Peripheral | Signal           | —  | to Peripheral         | Use Case                  | Control                          | Comment  |
|------------|------------------|----|-----------------------|---------------------------|----------------------------------|--|
| LPTMR      | Hardware trigger | to | TPMx                  | TPM Trigger input         | TPMx_CONF[T RGSEL] (4 bit field) | —  |
| TPMx       | TOF              | to | TPMx                  | TPM Trigger input         | TPMx_CONF[T RGSEL] (4 bit field) | —  |
| TPM1       | Timebase         | to | TPMx                  | TPM Global timebase input | TPMx_CONF[G TBEEN]               | —  |
| PIT CHx    | TIF0, TIF1       | to | TPMx                  | TPM Trigger input         | TPMx_CONF[T RGSEL] (4 bit field) | If PIT is triggering the TPM, the TPM clock must be faster than Bus clock. |
| RTC        | ALARM or SECONDS | to | TPMx                  | TPM Trigger input         | TPMx_CONF[T RGSEL] (4 bit field) | —  |
| EXTRG_IN   | EXTRG_IN         | to | TPMx                  | TPM Trigger input         | TPMx_CONF[T RGSEL] (4 bit field) | —  |
| CMP0       | CMP0_OUT         | to | TPMx                  | TPM Trigger input         | TPMx_CONF[T RGSEL] (4 bit field) | —  |
| LPTMR      | Hardware trigger | to | TSI                   | TSI triggering            | TSI selects HW trigger           | —  |
| UART0      | UART0_TX         | to | Modulated by TPM1 CH0 | UART modulation           | SOPT5_UART0 TXSRC                | —  |
| UART0      | UART0_TX         | to | Modulated by TPM2 CH0 | UART modulation           | SOPT5_UART0 TXSRC                | —  |
| UART1      | UART1_TX         | to | Modulated by TPM1 CH0 | UART modulation           | SOPT5_UART1 TXSRC                | —  |
| UART1      | UART1_TX         | to | Modulated by TPM2 CH0 | UART modulation           | SOPT5_UART1 TXSRC                | —  |
| PIT        | TIF0             | to | DAC                   | Advance DAC FIFO          | DAC HWTRG Select                 | —  |
| PIT        | TIF0             | to | DMA CH0               | DMA HW Trigger            | DMA MUX register option          | —  |
| PIT        | TIF1             | to | DMA CH1               | DMA HW Trigger            | DMA MUX register option          | —  |

### 3.2.2 Analog reference options

Several analog blocks have selectable reference voltages as shown in the below table. These options allow analog peripherals to share or have separate analog references. Care should be taken when selecting analog references to avoid cross talk noise.

**Table 3-2. Analog reference options**

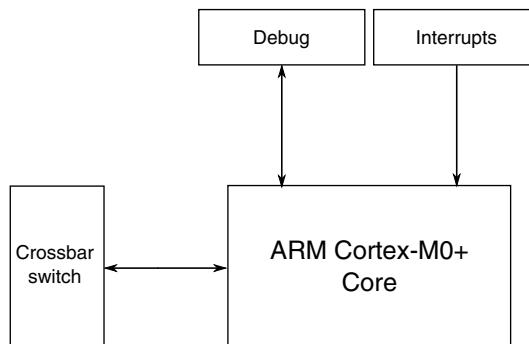
| Module             | Reference option                        | Comment/ Reference selection      |
|--------------------|---|-----------------------------------|
| 16-bit SAR ADC     | 1 - VREFH<br>2 - VDDA<br>3 - Reserved   | Selected by ADCx_SC2[REFSEL] bits |
| 12-bit DAC         | 1 - VREFH<br>2 - VDDA <sup>1</sup>      | Selected by DACx_C0[DACRFS] bit   |
| CMP with 6-bit DAC | Vin1 - VREFH<br>Vin2 - VDD <sup>1</sup> | Selected by CMPx_DACCR[VRSEL] bit |

1. Use this option for the best ADC operation.

## 3.3 Core Modules

### 3.3.1 ARM Cortex-M0+ Core Configuration

This section summarizes how the module has been configured in the chip. Full documentation for this module is provided by ARM and can be found at [www.arm.com](http://www.arm.com).

**Figure 3-1. Core configuration****Table 3-3. Reference links to related information**

| Topic                              | Related module                              | Reference   |
|------------------------------------|---|---|
| Full description                   | ARM Cortex-M0+ core, r0p0                   | <a href="#">ARM Cortex-M0+ Technical Reference Manual, r0p0</a> |
| System memory map                  |   | <a href="#">System memory map</a>                               |
| Clocking                           |   | <a href="#">Clock distribution</a>                              |
| Power management                   |   | <a href="#">Power management</a>                                |
| System/instruction/data bus module | Crossbar switch                             | <a href="#">Crossbar switch</a>                                 |
| Debug                              | Serial Wire Debug (SWD)                     | <a href="#">Debug</a>   |
| Interrupts                         | Nested Vectored Interrupt Controller (NVIC) | <a href="#">NVIC</a>  |
|                                    | Miscellaneous Control Module (MCM)          | <a href="#">MCM</a>   |

### 3.3.1.1 ARM Cortex M0+ Core

The ARM Cortex M0+ parameter settings are as follows:

**Table 3-4. Table 3. ARM Cortex-M0+ parameter settings**

| Parameter              | Verilog Name | Value       | Description  |
|------------------------|--------------|-------------|--|
| Arch Clock Gating      | ACG          | 1 = Present | Implements architectural clock gating                |
| DAP Slave Port Support | AHBSLV       | 1           | Support any AHB debug access port (like the CM4 DAP) |
| DAP ROM Table Base     | BASEADDR     | 0xF000_2003 | Base address for DAP ROM table                       |

*Table continues on the next page...*

**Table 3-4. Table 3. ARM Cortex-M0+ parameter settings (continued)**

| Parameter                    | Verilog Name | Value        | Description  |
|------------------------------|--------------|--------------|--|
| Endianess                    | BE           | 0            | Little endian control for data transfers                       |
| Breakpoints                  | BKPT         | 2            | Implements 2 breakpoints                                       |
| Debug Support                | DBG          | 1 = Present  |  |
| Halt Event Support           | HALTEV       | 1 = Present  |  |
| I/O Port                     | IOP          | 1 = Present  | Implements single-cycle ld/st accesses to special addr space   |
| IRQ Mask Enable              | IRQDIS       | 0x00000000   | Assume (for now) all 32 IRQs are used (set if IRQ is disabled) |
| Debug Port Protocol          | JTAGnSW      | 0 = SWD      | SWD protocol, not JTAG   |
| Core Memory Protection       | MPU          | 0 = Absent   | No MPU   |
| Number of IRQs               | NUMIRQ       | 32           | Assume full NVIC request vector                                |
| Reset all regs               | RAR          | 0 = Standard | Do not force all registers to be async reset                   |
| Multiplier                   | SMUL         | 0 = Fast Mul | Implements single-cycle multiplier                             |
| Multi-drop Support           | SWMD         | 0 = Absent   | Do not include serial wire support for multi-drop              |
| System Tick Timer            | SYST         | 1 = Present  | Implements system tick timer (for CM4 compatibility)           |
| DAP Target ID                | TARGETID     | 0            |  |
| User/Privileged              | USER         | 1 = Present  | Implements processor operating modes                           |
| Vector Table Offset Register | VTOR         | 1 = Present  | Implements relocation of exception vector table                |
| WIC Support                  | WIC          | 1 = Present  | Implements WIC interface                                       |
| WIC Requests                 | WICLINES     | 34           | Exact number of wakeup IRQs is 34                              |
| Watchpoints                  | WPT          | 2            | Implements 2 watchpoints                                       |

For details on the ARM Cortex-M0+ processor core see the ARM website: [arm.com](http://arm.com).

### 3.3.1.2 Buses, Interconnects, and Interfaces

The ARM Cortex-M0+ core has two bus interfaces:

- single 32-bit AMBA-3 AHB-Lite system interface that provides connections to peripherals and all system memory, which includes flash and RAM.
- single 32-bit I/O port bus interfacing to the GPIO with 1-cycle loads and stores.

### 3.3.1.3 System Tick Timer

The CLKSOURCE bit in SysTick Control and Status register selects either the core clock (when CLKSOURCE = 1) or a divide-by-16 of the core clock (when CLKSOURCE = 0). Because the timing reference is a variable frequency, the TENMS bit in the SysTick Calibration Value Register is always zero.

### 3.3.1.4 Debug Facilities

This device supports standard ARM 2-pin SWD debug port.

### 3.3.1.5 Core Privilege Levels

The Core on this device is implemented with both Privileged and Unprivileged levels. The ARM documentation uses different terms than this document to distinguish between privilege levels.

| If you see this term... | it also means this term... |
|-------------------------|----------------------------|
| Privileged              | Supervisor                 |
| Unprivileged or user    | User                       |

## 3.3.2 Nested Vectored Interrupt Controller (NVIC) Configuration

This section summarizes how the module has been configured in the chip. Full documentation for this module is provided by ARM and can be found at [www.arm.com](http://www.arm.com).

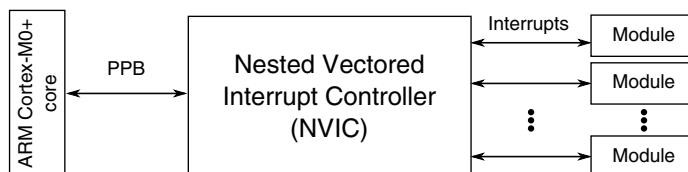


Figure 3-2. NVIC configuration

Table 3-5. Reference links to related information

| Topic            | Related module                              | Reference  |
|------------------|---|--|
| Full description | Nested Vectored Interrupt Controller (NVIC) | <a href="http://www.arm.com">ARM Cortex-M0+ Technical Reference Manual</a> |

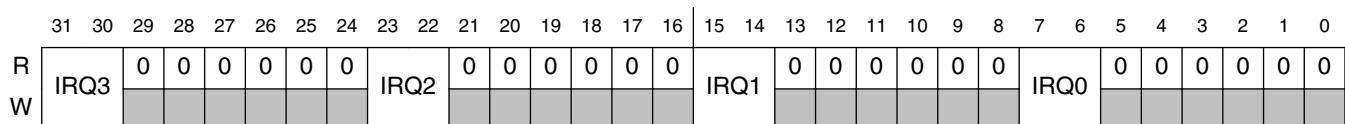
Table continues on the next page...

**Table 3-5. Reference links to related information (continued)**

| Topic                        | Related module      | Reference           |
|------------------------------|---------------------|---------------------|
| System memory map            |                     | System memory map   |
| Clocking                     |                     | Clock distribution  |
| Power management             |                     | Power management    |
| Private Peripheral Bus (PPB) | ARM Cortex-M0+ core | ARM Cortex-M0+ core |

### 3.3.2.1 Interrupt priority levels

This device supports 4 priority levels for interrupts. Therefore, in the NVIC each source in the IPR registers contains 2 bits. For example, IPR0 is shown below:



### 3.3.2.2 Non-maskable interrupt

The non-maskable interrupt request to the NVIC is controlled by the external  $\overline{\text{NMI}}$  signal. The pin the  $\overline{\text{NMI}}$  signal is multiplexed on, must be configured for the  $\overline{\text{NMI}}$  function to generate the non-maskable interrupt request.

### 3.3.2.3 Interrupt channel assignments

The interrupt vector assignments are defined in the following table.

- Vector number — the value stored on the stack when an interrupt is serviced.
- IRQ number — non-core interrupt source count, which is the vector number minus 16.

The IRQ number is used within ARM's NVIC documentation.

**Table 3-7. Interrupt vector assignments**

| Address                                | Vector | IRQ <sup>1</sup> | NVIC IPR register number <sup>2</sup> | Source module | Source description |
|--|--------|------------------|---------------------------------------|---------------|--------------------|
| <b>ARM Core System Handler Vectors</b> |        |                  |                                       |               |                    |

*Table continues on the next page...*

**Table 3-7. Interrupt vector assignments (continued)**

| Address                 | Vector | IRQ <sup>1</sup> | NVIC<br>IPR<br>register<br>number <sup>2</sup> | Source module     | Source description                                      |
|-------------------------|--------|------------------|--|-------------------|---|
| 0x0000_0000             | 0      | —                | —  | ARM core          | Initial Stack Pointer                                   |
| 0x0000_0004             | 1      | —                | —  | ARM core          | Initial Program Counter                                 |
| 0x0000_0008             | 2      | —                | —  | ARM core          | Non-maskable Interrupt (NMI)                            |
| 0x0000_000C             | 3      | —                | —  | ARM core          | Hard Fault  |
| 0x0000_0010             | 4      | —                | —  | —                 | —   |
| 0x0000_0014             | 5      | —                | —  | —                 | —   |
| 0x0000_0018             | 6      | —                | —  | —                 | —   |
| 0x0000_001C             | 7      | —                | —  | —                 | —   |
| 0x0000_0020             | 8      | —                | —  | —                 | —   |
| 0x0000_0024             | 9      | —                | —  | —                 | —   |
| 0x0000_0028             | 10     | —                | —  | —                 | —   |
| 0x0000_002C             | 11     | —                | —  | ARM core          | Supervisor call (SVCall)                                |
| 0x0000_0030             | 12     | —                | —  | —                 | —   |
| 0x0000_0034             | 13     | —                | —  | —                 | —   |
| 0x0000_0038             | 14     | —                | —  | ARM core          | Pendable request for system service<br>(PendableSrvReq) |
| 0x0000_003C             | 15     | —                | —  | ARM core          | System tick timer (SysTick)                             |
| <b>Non-Core Vectors</b> |        |                  |  |                   |   |
| 0x0000_0040             | 16     | 0                | 0  | DMA               | DMA channel 0 transfer complete and error               |
| 0x0000_0044             | 17     | 1                | 0  | DMA               | DMA channel 1 transfer complete and error               |
| 0x0000_0048             | 18     | 2                | 0  | DMA               | DMA channel 2 transfer complete and error               |
| 0x0000_004C             | 19     | 3                | 0  | DMA               | DMA channel 3 transfer complete and error               |
| 0x0000_0050             | 20     | 4                | 1  | —                 | —   |
| 0x0000_0054             | 21     | 5                | 1  | FTFA              | Command complete and read collision                     |
| 0x0000_0058             | 22     | 6                | 1  | PMC               | Low-voltage detect, low-voltage warning                 |
| 0x0000_005C             | 23     | 7                | 1  | LLWU              | Low Leakage Wakeup                                      |
| 0x0000_0060             | 24     | 8                | 2  | I <sup>2</sup> C0 |   |
| 0x0000_0064             | 25     | 9                | 2  | I <sup>2</sup> C1 |   |
| 0x0000_0068             | 26     | 10               | 2  | SPI0              | Single interrupt vector for all sources                 |
| 0x0000_006C             | 27     | 11               | 2  | SPI1              | Single interrupt vector for all sources                 |
| 0x0000_0070             | 28     | 12               | 3  | UART0             | Status and error  |
| 0x0000_0074             | 29     | 13               | 3  | UART1             | Status and error  |
| 0x0000_0078             | 30     | 14               | 3  | UART2             | Status and error  |
| 0x0000_007C             | 31     | 15               | 3  | ADC0              |   |
| 0x0000_0080             | 32     | 16               | 4  | CMP0              |   |
| 0x0000_0084             | 33     | 17               | 4  | TPM0              |   |
| 0x0000_0088             | 34     | 18               | 4  | TPM1              |   |
| 0x0000_008C             | 35     | 19               | 4  | TPM2              |   |

Table continues on the next page...

**Table 3-7. Interrupt vector assignments (continued)**

| Address     | Vector | IRQ <sup>1</sup> | NVIC IPR register number <sup>2</sup> | Source module       | Source description                       |
|-------------|--------|------------------|---------------------------------------|---------------------|--|
| 0x0000_0090 | 36     | 20               | 5                                     | RTC                 | Alarm interrupt                          |
| 0x0000_0094 | 37     | 21               | 5                                     | RTC                 | Seconds interrupt                        |
| 0x0000_0098 | 38     | 22               | 5                                     | PIT                 | Single interrupt vector for all channels |
| 0x0000_009C | 39     | 23               | 5                                     | —                   | —  |
| 0x0000_00A0 | 40     | 24               | 6                                     | USB OTG             |  |
| 0x0000_00A4 | 41     | 25               | 6                                     | DAC0                |  |
| 0x0000_00A8 | 42     | 26               | 6                                     | TSI0                |  |
| 0x0000_00AC | 43     | 27               | 6                                     | MCG                 |  |
| 0x0000_00B0 | 44     | 28               | 7                                     | LPTMR0              |  |
| 0x0000_00B4 | 45     | 29               | 7                                     | —                   |  |
| 0x0000_00B8 | 46     | 30               | 7                                     | Port control module | Pin detect (Port A)                      |
| 0x0000_00BC | 47     | 31               | 7                                     | Port control module | Pin detect ( Port D )                    |

1. Indicates the NVIC's interrupt source number.

2. Indicates the NVIC's IPR register number used for this IRQ. The equation to calculate this value is: IRQ div 4

### 3.3.2.3.1 Determining the bitfield and register location for configuring a particular interrupt

Suppose you need to configure the SPI0 interrupt. The following table is an excerpt of the SPI0 row from [Interrupt priority levels](#).

**Table 3-8. Interrupt vector assignments**

| Address     | Vector | IRQ <sup>1</sup> | NVIC IPR register number <sup>2</sup> | Source module | Source description                      |
|-------------|--------|------------------|---------------------------------------|---------------|---|
| 0x0000_0068 | 26     | 10               | 2                                     | SPI0          | Single interrupt vector for all sources |

1. Indicates the NVIC's interrupt source number.

2. Indicates the NVIC's IPR register number used for this IRQ. The equation to calculate this value is: IRQ div 4.

- The NVIC registers you would use to configure the interrupt are:
  - NVICIPR2
- To determine the particular IRQ's bitfield location within these particular registers:
  - NVICIPR2 bitfield starting location =  $8 * (\text{IRQ mod 4}) + 6 = 22$

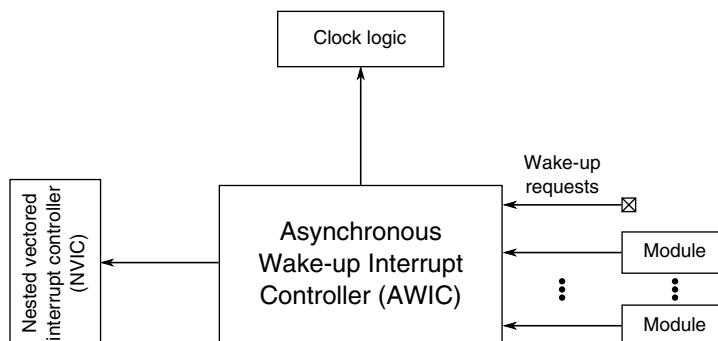
Since the NVICIPR bitfields are 2-bit wide (4 priority levels), the NVICIPR2 bitfield range is 22-23

Therefore, the following bitfield locations are used to configure the SPI0 interrupts:

- NVICIPR2[23:22]

### 3.3.3 Asynchronous wake-up interrupt controller (AWIC) configuration

This section summarizes how the module has been configured in the chip. Full documentation for this module is provided by ARM and can be found at [www.arm.com](http://www.arm.com).



**Figure 3-3. Asynchronous wake-up interrupt controller configuration**

**Table 3-9. Reference links to related information**

| Topic             | Related module                              | Reference                            |
|-------------------|---|--------------------------------------|
| System memory map |   | <a href="#">System memory map</a>    |
| Clocking          |   | <a href="#">Clock distribution</a>   |
| Power management  |   | <a href="#">Power management</a>     |
|                   | Nested vectored interrupt controller (NVIC) | <a href="#">NVIC</a>                 |
| Wake-up requests  |   | <a href="#">AWIC wake-up sources</a> |

#### 3.3.3.1 Wake-up sources

The device uses the following internal and external inputs to the AWIC module.

**Table 3-10. AWIC stop wake-up sources**

| Wake-up source          | Description   |
|-------------------------|---|
| Available system resets | RESET pin when LPO is its clock source  |
| Low-voltage detect      | Mode Controller   |
| Low-voltage warning     | Mode Controller   |
| Pin interrupts          | Port control module - Any enabled pin interrupt is capable of waking the system |
| ADC                     | The ADC is functional when using internal clock source                          |

*Table continues on the next page...*

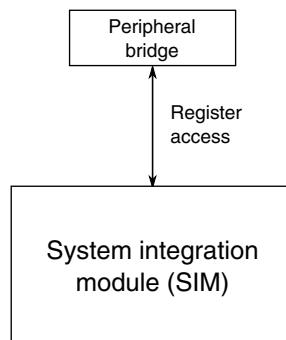
**Table 3-10. AWIC stop wake-up sources (continued)**

| Wake-up source    | Description                                  |
|-------------------|--|
| CMP0              | Interrupt in normal or trigger mode          |
| I <sup>2</sup> Cx | Address match wakeup                         |
| UART0             | Any interrupt provided clock remains enabled |
| UART1 and UART2   | Active edge on RXD                           |
| RTC               | Alarm or seconds interrupt                   |
| TSI               | Any interrupt                                |
| NMI               | NMI pin                                      |
| TPMx              | Any interrupt provided clock remains enabled |
| LPTMR             | Any interrupt provided clock remains enabled |
| SPI               | Slave mode interrupt                         |

## 3.4 System Modules

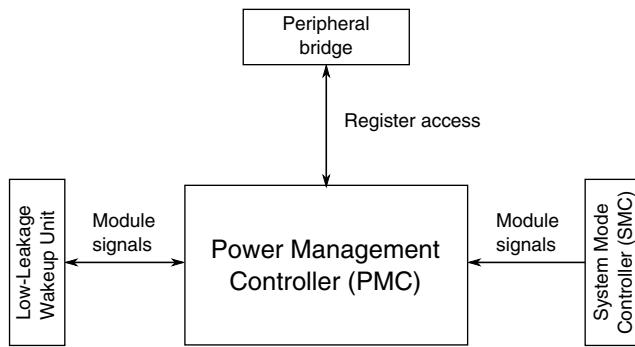
### 3.4.1 SIM Configuration

This section summarizes how the module has been configured in the chip. For a comprehensive description of the module itself, see the module's dedicated chapter.

**Figure 3-4. SIM configuration****Table 3-11. Reference links to related information**

| Topic             | Related module | Reference                          |
|-------------------|----------------|------------------------------------|
| Full description  | SIM            | <a href="#">SIM</a>                |
| System memory map |                | <a href="#">System memory map</a>  |
| Clocking          |                | <a href="#">Clock distribution</a> |
| Power management  |                | <a href="#">Power management</a>   |

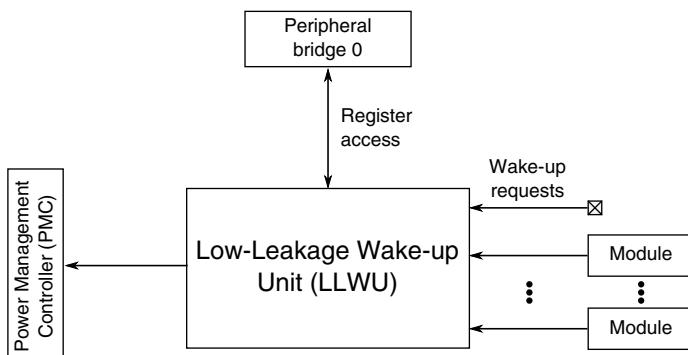
### **3.4.2**



**Figure 3-6. PMC configuration**

**Table 3-13. Reference links to related information**

| Topic                          | Related module               | Reference                              |
|--------------------------------|------------------------------|--|
| Full description               | PMC                          | <a href="#">PMC</a>                    |
| System memory map              |                              | <a href="#">System memory map</a>      |
| Power management               |                              | <a href="#">Power management</a>       |
| Full description               | System Mode Controller (SMC) | <a href="#">System Mode Controller</a> |
| Low-Leakage Wakeup Unit (LLWU) |                              | <a href="#">LLWU</a>                   |
| Reset Control Module (RCM)     |                              | <a href="#">Reset</a>                  |

**Figure 3-7. Low-Leakage Wake-up Unit configuration****Table 3-14. Reference links to related information**

| Topic             | Related module                    | Reference   |
|-------------------|-----------------------------------|---|
| Full description  | LLWU                              | <a href="#">LLWU</a>                              |
| System memory map |                                   | <a href="#">System memory map</a>                 |
| Clocking          |                                   | <a href="#">Clock distribution</a>                |
| Power management  |                                   | <a href="#">Power management chapter</a>          |
|                   | Power Management Controller (PMC) | <a href="#">Power Management Controller (PMC)</a> |
|                   | System Mode Controller (SMC)      | <a href="#">System Mode Controller</a>            |
| Wake-up requests  |                                   | <a href="#">LLWU wake-up sources</a>              |

### 3.4.4.1 LLWU interrupt

**NOTE**

Do not mask the LLWU interrupt when in LLS mode. Masking the interrupt prevents the device from exiting stop mode when a wakeup is detected.

### 3.4.4.2 Wake-up Sources

The device uses the following internal peripheral and external pin inputs as wakeup sources to the LLWU module. LLWU\_Px are external pin inputs, and LLWU\_M0IF-M7IF are connections to the internal peripheral interrupt flags.

**NOTE**

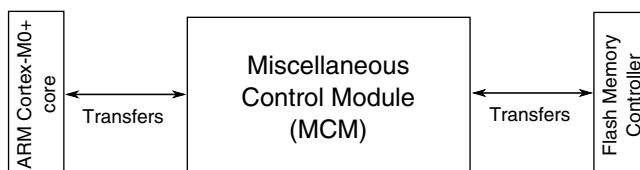
In addition to the LLWU wakeup sources, the device also wakes from low power modes when NMI or RESET pins are enabled and the respective pin is asserted.

**Table 3-15. Wakeup Sources**

| LLWU pin  | Module source or pin name |
|-----------|---------------------------|
| LLWU_P5   | PTB0                      |
| LLWU_P6   | PTC1                      |
| LLWU_P7   | PTC3                      |
| LLWU_P8   | PTC4                      |
| LLWU_P9   | PTC5                      |
| LLWU_P10  | PTC6                      |
| LLWU_P14  | PTD4                      |
| LLWU_P15  | PTD6                      |
| LLWU_M0IF | LPTMR0                    |
| LLWU_M1IF | CMP0                      |
| LLWU_M2IF | Reserved                  |
| LLWU_M3IF | Reserved                  |
| LLWU_M4IF | TSI0                      |
| LLWU_M5IF | RTC Alarm                 |
| LLWU_M6IF | Reserved                  |
| LLWU_M7IF | RTC Seconds               |

### 3.4.5 MCM Configuration

This section summarizes how the module has been configured in the chip. For a comprehensive description of the module itself, see the module's dedicated chapter.

**Figure 3-8. MCM configuration****Table 3-16. Reference links to related information**

| Topic                        | Related module                     | Reference                           |
|------------------------------|------------------------------------|-------------------------------------|
| Full description             | Miscellaneous control module (MCM) | <a href="#">MCM</a>                 |
| System memory map            |                                    | <a href="#">System memory map</a>   |
| Clocking                     |                                    | <a href="#">Clock distribution</a>  |
| Power management             |                                    | <a href="#">Power management</a>    |
| Private Peripheral Bus (PPB) | ARM Cortex-M0+ core                | <a href="#">ARM Cortex-M0+ core</a> |

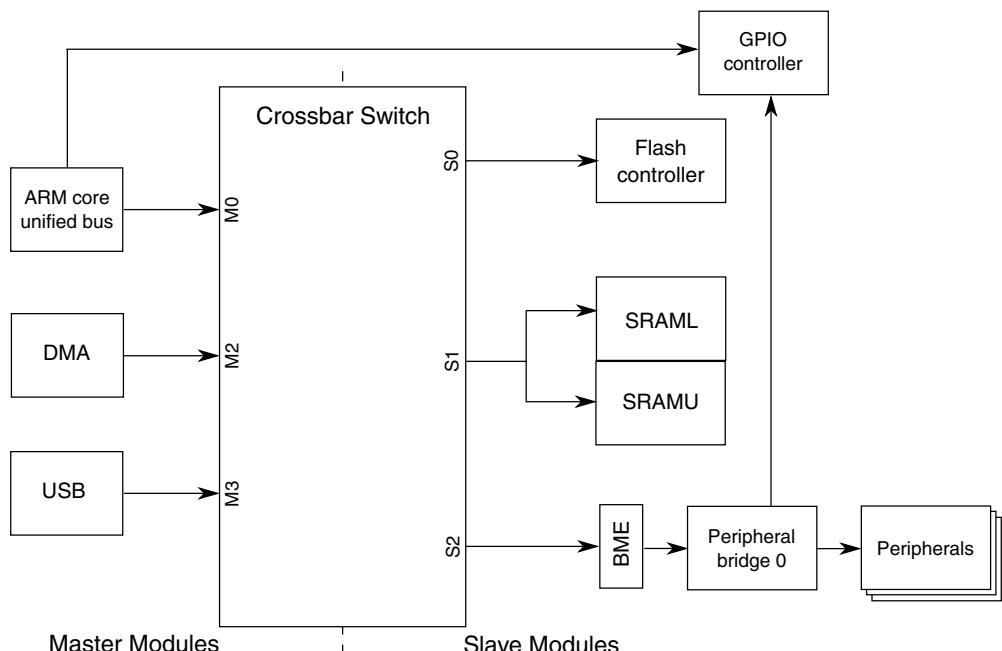
*Table continues on the next page...*

**Table 3-16. Reference links to related information (continued)**

| Topic    | Related module          | Reference                               |
|----------|-------------------------|---|
| Transfer | Flash memory controller | <a href="#">Flash memory controller</a> |

### 3.4.6 Crossbar-Light Switch Configuration

This section summarizes how the module has been configured in the chip. For a comprehensive description of the module itself, see the module's dedicated chapter.

**Figure 3-9. Crossbar-Light switch integration****Table 3-17. Reference links to related information**

| Topic                  | Related module          | Reference                               |
|------------------------|-------------------------|---|
| Full description       | Crossbar switch         | <a href="#">Crossbar Switch</a>         |
| System memory map      |                         | <a href="#">System memory map</a>       |
| Clocking               |                         | <a href="#">Clock Distribution</a>      |
| Crossbar switch master | ARM Cortex-M0+ core     | <a href="#">ARM Cortex-M0+ core</a>     |
| Crossbar switch master | DMA controller          | <a href="#">DMA controller</a>          |
| Crossbar switch master | USB FS/LS               | <a href="#">USB FS/LS</a>               |
| Crossbar switch slave  | Flash memory controller | <a href="#">Flash memory controller</a> |
| Crossbar switch slave  | SRAM controller         | <a href="#">SRAM configuration</a>      |

*Table continues on the next page...*

**Table 3-17. Reference links to related information (continued)**

| Topic                 | Related module    | Reference                         |
|-----------------------|-------------------|-----------------------------------|
| Crossbar switch slave | Peripheral bridge | <a href="#">Peripheral bridge</a> |
| 2-ported peripheral   | GPIO controller   | <a href="#">GPIO controller</a>   |

### 3.4.6.1 Crossbar-Light Switch Master Assignments

The masters connected to the crossbar switch are assigned as follows:

| Master module        | Master port number |
|----------------------|--------------------|
| ARM core unified bus | 0                  |
| DMA                  | 2                  |
| USB OTG              | 3                  |

### 3.4.6.2 Crossbar Switch Slave Assignments

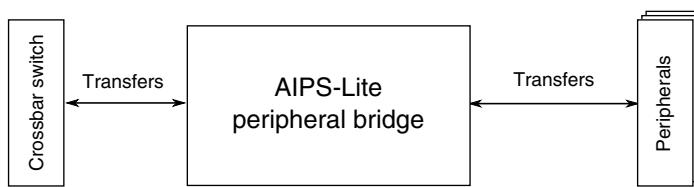
This device contains 3 slaves connected to the crossbar switch.

The slave assignment is as follows:

| Slave module            | Slave port number |
|-------------------------|-------------------|
| Flash memory controller | 0                 |
| SRAM controller         | 1                 |
| Peripheral bridge 0     | 2                 |

### 3.4.7 Peripheral Bridge Configuration

This section summarizes how the module has been configured in the chip. For a comprehensive description of the module itself, see the module's dedicated chapter.

**Figure 3-10. Peripheral bridge configuration****Table 3-18. Reference links to related information**

| Topic             | Related module                | Reference                                     |
|-------------------|-------------------------------|---|
| Full description  | Peripheral bridge (AIPS-Lite) | <a href="#">Peripheral bridge (AIPS-Lite)</a> |
| System memory map |                               | <a href="#">System memory map</a>             |
| Clocking          |                               | <a href="#">Clock Distribution</a>            |
| Crossbar switch   | Crossbar switch               | <a href="#">Crossbar switch</a>               |

### 3.4.7.1 Number of peripheral bridges

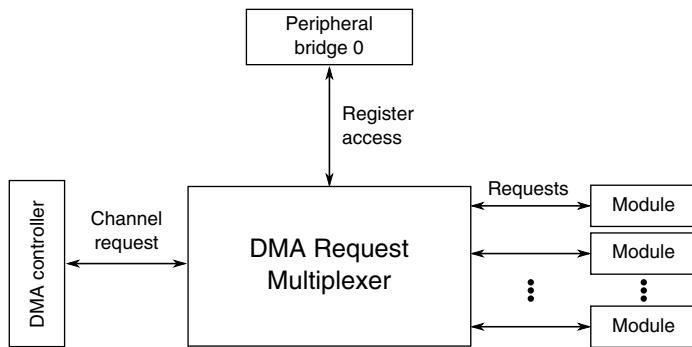
This device contains one peripheral bridge.

### 3.4.7.2 Memory maps

The peripheral bridges are used to access the registers of most of the modules on this device. See [AIPS0 Memory Map](#) for the memory slot assignment for each module.

## 3.4.8 DMA request multiplexer configuration

This section summarizes how the module has been configured in the chip. For a comprehensive description of the module itself, see the module's dedicated chapter.

**Figure 3-11. DMA request multiplexer configuration****Table 3-19. Reference links to related information**

| Topic             | Related module          | Reference                           |
|-------------------|-------------------------|-------------------------------------|
| Full description  | DMA request multiplexer | <a href="#">DMA Mux</a>             |
| System memory map |                         | <a href="#">System memory map</a>   |
| Clocking          |                         | <a href="#">Clock distribution</a>  |
| Power management  |                         | <a href="#">Power management</a>    |
| Channel request   | DMA controller          | <a href="#">DMA Controller</a>      |
| Requests          |                         | <a href="#">DMA request sources</a> |

### 3.4.8.1 DMA MUX Request Sources

This device includes a DMA request mux that allows up to 63 DMA request signals to be mapped to any of the 4 DMA channels. Because of the mux there is no hard correlation between any of the DMA request sources and a specific DMA channel. Some of the modules support Asynchronous DMA operation as indicated by the last column in the following DMA source assignment table.

**Table 3-20. DMA request sources - MUX 0**

| Source number | Source module | Source description            | Async DMA capable |
|---------------|---------------|-------------------------------|-------------------|
| 0             | —             | Channel disabled <sup>1</sup> |                   |
| 1             | Reserved      | Not used                      |                   |
| 2             | UART0         | Receive                       | Yes               |
| 3             | UART0         | Transmit                      | Yes               |
| 4             | UART1         | Receive                       |                   |
| 5             | UART1         | Transmit                      |                   |
| 6             | UART2         | Receive                       |                   |
| 7             | UART2         | Transmit                      |                   |

*Table continues on the next page...*

**Table 3-20. DMA request sources - MUX 0 (continued)**

| Source number | Source module     | Source description | Async DMA capable |
|---------------|-------------------|--------------------|-------------------|
| 8             | Reserved          | —                  |                   |
| 9             | Reserved          | —                  |                   |
| 10            | Reserved          | —                  |                   |
| 11            | Reserved          | —                  |                   |
| 12            | Reserved          | —                  |                   |
| 13            | Reserved          | —                  |                   |
| 14            | Reserved          | —                  |                   |
| 15            | Reserved          | —                  |                   |
| 16            | SPI0              | Receive            |                   |
| 17            | SPI0              | Transmit           |                   |
| 18            | SPI1              | Receive            |                   |
| 19            | SPI1              | Transmit           |                   |
| 20            | Reserved          | —                  |                   |
| 21            | Reserved          | —                  |                   |
| 22            | I <sup>2</sup> C0 | —                  |                   |
| 23            | I <sup>2</sup> C1 | —                  |                   |
| 24            | TPM0              | Channel 0          | Yes               |
| 25            | TPM0              | Channel 1          | Yes               |
| 26            | TPM0              | Channel 2          | Yes               |
| 27            | TPM0              | Channel 3          | Yes               |
| 28            | TPM0              | Channel 4          | Yes               |
| 29            | TPM0              | Channel 5          | Yes               |
| 30            | Reserved          | —                  |                   |
| 31            | Reserved          | —                  |                   |
| 32            | TPM1              | Channel 0          | Yes               |
| 33            | TPM1              | Channel 1          | Yes               |
| 34            | TPM2              | Channel 0          | Yes               |
| 35            | TPM2              | Channel 1          | Yes               |
| 36            | Reserved          | —                  |                   |
| 37            | Reserved          | —                  |                   |
| 38            | Reserved          | —                  |                   |
| 39            | Reserved          | —                  |                   |
| 40            | ADC0              | —                  | Yes               |
| 41            | Reserved          | —                  |                   |
| 42            | CMP0              | —                  | Yes               |
| 43            | Reserved          | —                  |                   |
| 44            | Reserved          | —                  |                   |
| 45            | DAC0              | —                  |                   |
| 46            | Reserved          | —                  |                   |

*Table continues on the next page...*

**Table 3-20. DMA request sources - MUX 0 (continued)**

| Source number | Source module       | Source description | Async DMA capable |
|---------------|---------------------|--------------------|-------------------|
| 47            | Reserved            | —                  |                   |
| 48            | Reserved            | —                  |                   |
| 49            | Port control module | Port A             | Yes               |
| 50            | Reserved            | —                  |                   |
| 51            | Reserved            | —                  |                   |
| 52            | Port control module | Port D             | Yes               |
| 53            | Reserved            | —                  |                   |
| 54            | TPM0                | Overflow           | Yes               |
| 55            | TPM1                | Overflow           | Yes               |
| 56            | TPM2                | Overflow           | Yes               |
| 57            | TSI                 | —                  | Yes               |
| 58            | Reserved            | —                  |                   |
| 59            | Reserved            | —                  |                   |
| 60            | DMA MUX             | Always enabled     |                   |
| 61            | DMA MUX             | Always enabled     |                   |
| 62            | DMA MUX             | Always enabled     |                   |
| 63            | DMA MUX             | Always enabled     |                   |

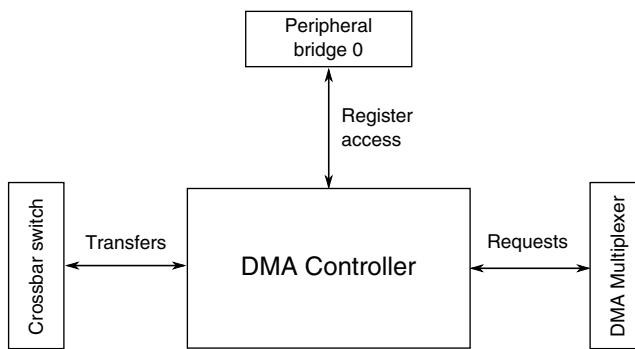
- Configuring a DMA channel to select source 0 or any of the reserved sources disables that DMA channel.

### 3.4.8.2 DMA transfers via PIT trigger

The PIT module can trigger a DMA transfer on the first two DMA channels. The assignments are detailed at [PIT/DMA Periodic Trigger Assignments](#).

### 3.4.9 DMA Controller Configuration

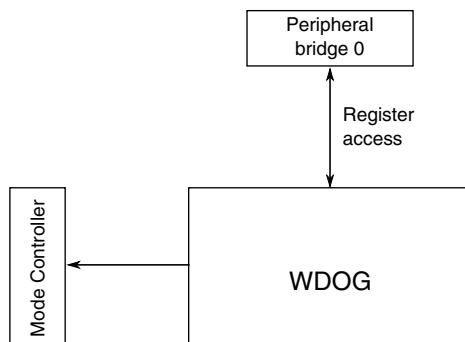
This section summarizes how the module has been configured in the chip. For a comprehensive description of the module itself, see the module's dedicated chapter.

**Figure 3-12. DMA Controller configuration****Table 3-21. Reference links to related information**

| Topic             | Related module  | Reference                           |
|-------------------|-----------------|-------------------------------------|
| Full description  | DMA controller  | <a href="#">DMA controller</a>      |
| System memory map |                 | <a href="#">System memory map</a>   |
| Clocking          |                 | <a href="#">Clock distribution</a>  |
| Power management  |                 | <a href="#">Power management</a>    |
| Crossbar switch   | Crossbar switch | <a href="#">Crossbar switch</a>     |
| Requests          |                 | <a href="#">DMA request sources</a> |

### 3.4.10 Computer Operating Properly (COP) Watchdog Configuration

This section summarizes how the module has been configured in the chip.

**Figure 3-13. COP watchdog configuration****Table 3-22. Reference links to related information**

| Topic            | Related module | Reference                          |
|------------------|----------------|------------------------------------|
| Clocking         |                | <a href="#">Clock distribution</a> |
| Power management |                | <a href="#">Power management</a>   |

*Table continues on the next page...*

**Table 3-22. Reference links to related information (continued)**

| Topic             | Related module                  | Reference           |
|-------------------|---------------------------------|---------------------|
| Programming model | System Integration Module (SIM) | <a href="#">SIM</a> |

### 3.4.10.1 COP clocks

The two clock inputs for the COP are the 1 kHz clock and the bus clock.

### 3.4.10.2 COP watchdog operation

The COP watchdog is intended to force a system reset when the application software fails to execute as expected. To prevent a system reset from the COP timer (when it is enabled), application software must reset the COP counter periodically. If the application program gets lost and fails to reset the COP counter before it times out, a system reset is generated to force the system back to a known starting point.

After any reset, the COP watchdog is enabled. If the COP watchdog is not used in an application, it can be disabled by clearing COPCTRL[COPT] in the SIM.

The COP counter is reset by writing 0x55 and 0xAA (in that order) to the address of the SIM's Service COP (SRVCOP) register during the selected timeout period. Writes do not affect the data in the SRVCOP register. As soon as the write sequence is complete, the COP timeout period is restarted. If the program fails to perform this restart during the timeout period, the microcontroller resets. Also, if any value other than 0x55 or 0xAA is written to the SRVCOP register, the microcontroller immediately resets.

The SIM's COPCTRL[COPCLKS] field selects the clock source used for the COP timer. The clock source options are either the bus clock or an internal 1 kHz clock source. With each clock source, there are three associated timeouts controlled by COPCTRL[COPT]. The following table summarizes the control functions of the COPCLKS and COPT bits. The COP watchdog defaults to operation from the 1 kHz clock source and the longest timeout for that clock source ( $2^{10}$  cycles).

**Table 3-23. COP configuration options**

| Control Bits     |               | Clock Source | COP Window Opens<br>(COPCTRL[COPW]=1) | COP Overflow Count   |
|------------------|---------------|--------------|---------------------------------------|----------------------|
| COPCTRL[COPCLKS] | COPCTRL[COPT] |              |                                       |                      |
| N/A              | 00            | N/A          | N/A                                   | COP is disabled      |
| 0                | 01            | 1 kHz        | N/A                                   | $2^5$ cycles (32 ms) |

*Table continues on the next page...*

**Table 3-23. COP configuration options (continued)**

| Control Bits     |               | Clock Source | COP Window Opens<br>(COPCTRL[COPW]=1) | COP Overflow Count        |
|------------------|---------------|--------------|---------------------------------------|---------------------------|
| COPCTRL[COPCLKS] | COPCTRL[COPT] |              |                                       |                           |
| 0                | 10            | 1 kHz        | N/A                                   | $2^8$ cycles (256 ms)     |
| 0                | 11            | 1 kHz        | N/A                                   | $2^{10}$ cycles (1024 ms) |
| 1                | 01            | Bus          | 6,144 cycles                          | $2^{13}$ cycles           |
| 1                | 10            | Bus          | 49,152 cycles                         | $2^{16}$ cycles           |
| 1                | 11            | Bus          | 196,608 cycles                        | $2^{18}$ cycles           |

After the bus clock source is selected, windowed COP operation is available by setting COPCTRL[COPW] in the SIM. In this mode, writes to the SRVCOP register to clear the COP timer must occur in the last 25% of the selected timeout period. A premature write immediately resets the chip. When the 1 kHz clock source is selected, windowed COP operation is not available.

The COP counter is initialized by the first writes to the SIM's COPCTRL register and after any system reset. Subsequent writes to the SIM's COPCTRL register have no effect on COP operation. Even if an application uses the reset default settings of the COPT, COPCLKS, and COPW bits, the user should write to the write-once COPCTRL register during reset initialization to lock in the settings. This approach prevents accidental changes if the application program becomes lost.

The write to the SRVCOP register that services (clears) the COP counter should not be placed in an interrupt service routine (ISR) because the ISR could continue to be executed periodically even if the main application program fails.

If the bus clock source is selected, the COP counter does not increment while the microcontroller is in debug mode or while the system is in stop (including VLPS or LLS) mode. The COP counter resumes when the microcontroller exits debug mode or stop mode.

If the 1 kHz clock source is selected, the COP counter is re-initialized to zero upon entry to either debug mode or stop (including VLPS or LLS) mode. The counter begins from zero upon exit from debug mode or stop mode.

Regardless of the clock selected, the COP is disabled when the chip enters a VLLSx mode. Upon a reset that wakes the chip from the VLLSx mode, the COP is re-initialized and enabled as for any reset.

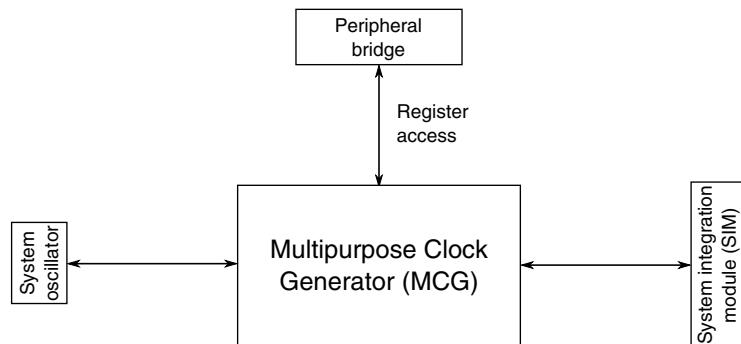
### 3.4.10.3 Clock Gating

This family of devices includes clock gating control for each peripheral, that is, the clock to each peripheral can explicitly be gated on or off, using clock-gate control bits in the SIM module.

## 3.5 Clock Modules

### 3.5.1 MCG Configuration

This section summarizes how the module has been configured in the chip. For a comprehensive description of the module itself, see the module's dedicated chapter.



**Figure 3-14. MCG configuration**

**Table 3-24. Reference links to related information**

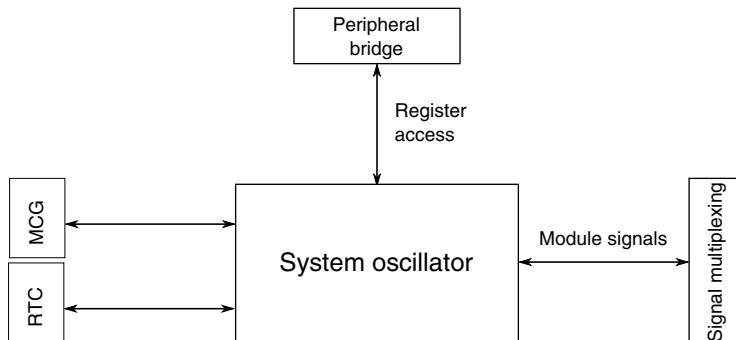
| Topic               | Related module | Reference                           |
|---------------------|----------------|-------------------------------------|
| Full description    | MCG            | <a href="#">MCG</a>                 |
| System memory map   |                | <a href="#">System memory map</a>   |
| Clocking            |                | <a href="#">Clock distribution</a>  |
| Power management    |                | <a href="#">Power management</a>    |
| Signal multiplexing | Port control   | <a href="#">Signal multiplexing</a> |

#### 3.5.1.1 MCG FLL modes

On L-series devices the MCGFLLCLK frequency is limited to 48 MHz max. The DCO is limited to the two lowest range settings (MCG\_C4[DRST\_DRST] must be set to either 0b00 or 0b01).

### 3.5.2 OSC Configuration

This section summarizes how the module has been configured in the chip. For a comprehensive description of the module itself, see the module's dedicated chapter.



**Figure 3-15. OSC configuration**

**Table 3-25. Reference links to related information**

| Topic               | Related module | Reference                           |
|---------------------|----------------|-------------------------------------|
| Full description    | OSC            | <a href="#">OSC</a>                 |
| System memory map   |                | <a href="#">System memory map</a>   |
| Clocking            |                | <a href="#">Clock distribution</a>  |
| Power management    |                | <a href="#">Power management</a>    |
| Signal multiplexing | Port control   | <a href="#">Signal multiplexing</a> |
| Full description    | MCG            | <a href="#">MCG</a>                 |

#### 3.5.2.1 OSC modes of operation with MCG and RTC

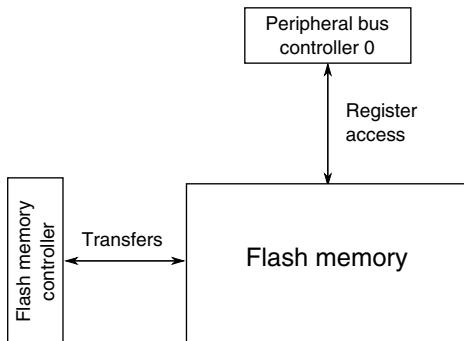
The most common method of controlling the OSC block is through MCG clock source selection MCG\_C1[CLKS] and the MCG\_C2 register bits to configure the oscillator frequency range, gain-mode, and for crystal or external clock operation. The OSC\_CR also provides control for enabling the OSC and configuring internal load capacitors for the EXTAL and XTAL pins. See the OSC and MCG chapters for more details.

The RTC\_CR[OSCE] bit has overriding control over the MCG and OSC\_CR enable functions. When RTC\_CR[OSCE] is set, the OSC is configured for low frequency, low power and the RTC\_CR[SCxP] bits override the OSC\_CR[SCxP] bits to control the internal capacitance configuration. See the RTC chapter for more details.

## 3.6 Memories and Memory Interfaces

### 3.6.1 Flash Memory Configuration

This section summarizes how the module has been configured in the chip. For a comprehensive description of the module itself, see the module's dedicated chapter.



**Figure 3-16. Flash memory configuration**

**Table 3-26. Reference links to related information**

| Topic             | Related module          | Reference                               |
|-------------------|-------------------------|---|
| Full description  | Flash memory            | <a href="#">Flash memory</a>            |
| System memory map |                         | <a href="#">System memory map</a>       |
| Clocking          |                         | <a href="#">Clock Distribution</a>      |
| Transfers         | Flash memory controller | <a href="#">Flash memory controller</a> |
| Register access   | Peripheral bridge       | <a href="#">Peripheral bridge</a>       |

#### 3.6.1.1 Flash Memory Sizes

The devices covered in this document contain 1 program flash block consisting of 1 KB sectors.

The amounts of flash memory for the devices covered in this document are:

**Table 3-27. KL25 flash memory size**

| Device       | Program flash (KB) | Block 0 (P-Flash) address range |
|--------------|--------------------|---------------------------------|
| MKL25Z32VFM4 | 32                 | 0x0000_0000 – 0x0000_7FFF       |
| MKL25Z64VFM4 | 64                 | 0x0000_0000 – 0x0000_FFFF       |

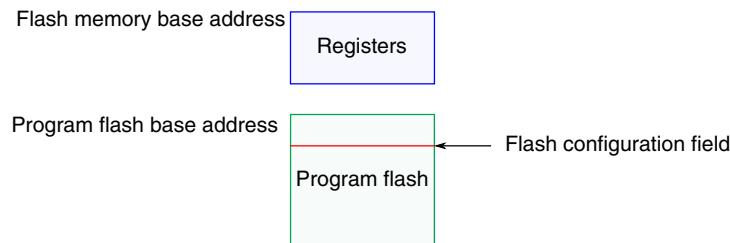
*Table continues on the next page...*

**Table 3-27. KL25 flash memory size (continued)**

| Device        | Program flash (KB) | Block 0 (P-Flash) address range |
|---------------|--------------------|---------------------------------|
| MKL25Z128VFM4 | 128                | 0x0000_0000 – 0x0001_FFFF       |
| MKL25Z32VFT4  | 32                 | 0x0000_0000 – 0x0000_7FFF       |
| MKL25Z64VFT4  | 64                 | 0x0000_0000 – 0x0000_FFFF       |
| MKL25Z128VFT4 | 128                | 0x0000_0000 – 0x0001_FFFF       |
| MKL25Z32VLH4  | 32                 | 0x0000_0000 – 0x0000_7FFF       |
| MKL25Z64VLH4  | 64                 | 0x0000_0000 – 0x0000_FFFF       |
| MKL25Z128VLH4 | 128                | 0x0000_0000 – 0x0001_FFFF       |
| MKL25Z32VLK4  | 32                 | 0x0000_0000 – 0x0000_7FFF       |
| MKL25Z64VLK4  | 64                 | 0x0000_0000 – 0x0000_FFFF       |
| MKL25Z128VLK4 | 128                | 0x0000_0000 – 0x0001_FFFF       |

### 3.6.1.2 Flash Memory Map

The flash memory and the flash registers are located at different base addresses as shown in the following figure. The base address for each is specified in [System memory map](#).



**Figure 3-17. Flash memory map**

contguous bBlock(in the memory ma bRegnnwing at address(0x0000\_0000) in thninwhiach flash memory isaccesssdnwhilg an

.

### 3.6.1.4 Flash Modes

The flash memory chapter defines two modes of operation - NVM normal and NVM special modes. On this device, The flash memory only operates in NVM normal mode. All references to NVM special mode should be ignored.

### 3.6.1.5 Erase All Flash Contents

In addition to software, the entire flash memory may be erased external to the flash memory via the SW-DP debug port by setting MDM-AP CONTROL[0]. MDM-AP STATUS[0] is set to indicate the mass erase command has been accepted. MDM-AP STATUS[0] is cleared when the mass erase completes.

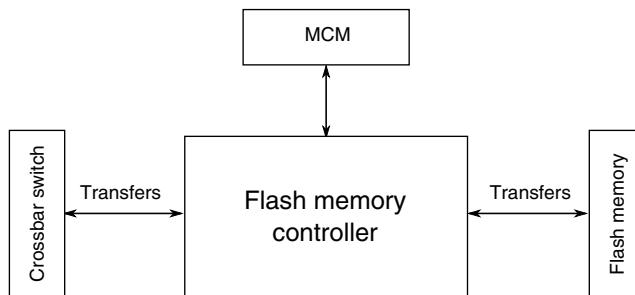
### 3.6.1.6 FTFA\_FOPT Register

The flash memory's FTFA\_FOPT register allows the user to customize the operation of the MCU at boot time. See [FOPT boot options](#) for details of its definition.

## 3.6.2 Flash Memory Controller Configuration

This section summarizes how the module has been configured in the chip. For a comprehensive description of the module itself, see the module's dedicated chapter.

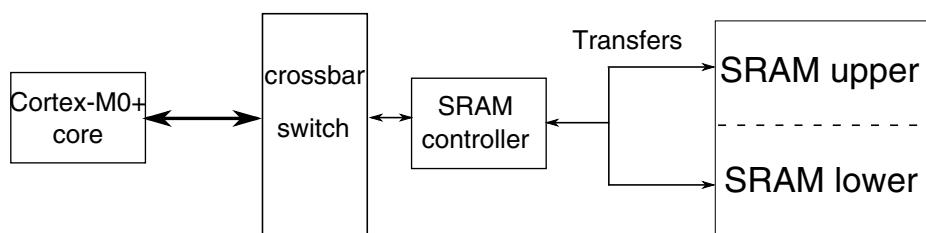
See MCM\_PLACR register description for details on the reset configuration of the FMC.

**Figure 3-18. Flash memory controller configuration****Table 3-28. Reference links to related information**

| Topic             | Related module          | Reference                               |
|-------------------|-------------------------|---|
| Full description  | Flash memory controller | <a href="#">Flash memory controller</a> |
| System memory map |                         | <a href="#">System memory map</a>       |
| Clocking          |                         | <a href="#">Clock Distribution</a>      |
| Transfers         | Flash memory            | <a href="#">Flash memory</a>            |
| Transfers         | Crossbar switch         | <a href="#">Crossbar Switch</a>         |
| Register access   | MCM                     | <a href="#">MCM</a>                     |

### 3.6.3 SRAM Configuration

This section summarizes how the module has been configured in the chip.

**Figure 3-19. SRAM configuration****Table 3-29. Reference links to related information**

| Topic               | Related module | Reference                           |
|---------------------|----------------|-------------------------------------|
| Full description    | SRAM           | <a href="#">SRAM</a>                |
| System memory map   |                | <a href="#">System memory map</a>   |
| Clocking            |                | <a href="#">Clock Distribution</a>  |
| ARM Cortex-M0+ core |                | <a href="#">ARM Cortex-M0+ core</a> |

### 3.6.3.1 SRAM Sizes

This device contains SRAM which could be accessed by bus masters through the cross-bar switch. The amount of SRAM for the devices covered in this document is shown in the following table.

**Table 3-30. KL25 SRAM memory size**

| Device        | SRAM (KB) |
|---------------|-----------|
| MKL25Z32VFM4  | 4         |
| MKL25Z64VFM4  | 8         |
| MKL25Z128VFM4 | 16        |
| MKL25Z32VFT4  | 4         |
| MKL25Z64VFT4  | 8         |
| MKL25Z128VFT4 | 16        |
| MKL25Z32VLH4  | 4         |
| MKL25Z64VLH4  | 8         |
| MKL25Z128VLH4 | 16        |
| MKL25Z32VLK4  | 4         |
| MKL25Z64VLK4  | 8         |
| MKL25Z128VLK4 | 16        |

### 3.6.3.2 SRAM Ranges

The on-chip SRAM is split into two ranges, 1/4 is allocated SRAM\_L and 3/4 is allocated to SRAM\_U.

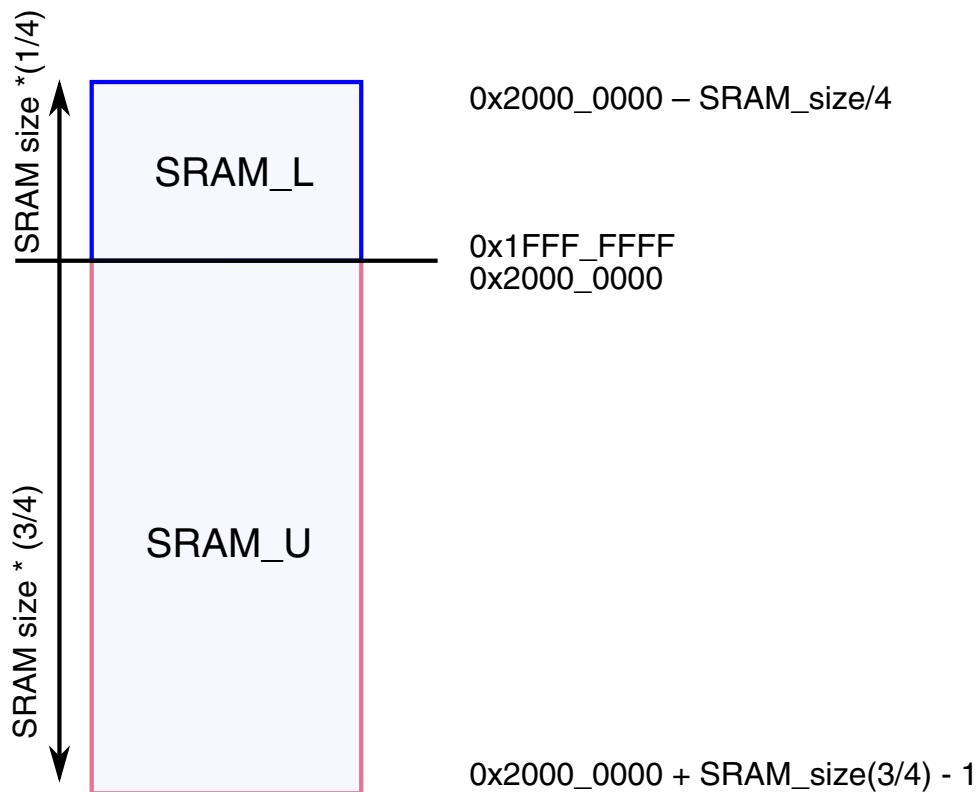
The on-chip RAM is implemented such that the SRAM\_L and SRAM\_U ranges form a contiguous block in the memory map. As such:

- SRAM\_L is anchored to 0x1FFF\_FFFF and occupies the space before this ending address.
- SRAM\_U is anchored to 0x2000\_0000 and occupies the space after this beginning address.

Valid address ranges for SRAM\_L and SRAM\_U are then defined as:

- SRAM\_L = [0x2000\_0000-(SRAM\_size/4)] to 0x1FFF\_FFFF
- SRAM\_U = 0x2000\_0000 to [0x2000\_0000+(SRAM\_size\*(3/4))-1]

This is illustrated in the following figure.



**Figure 3-20. SRAM blocks memory map**

For example, for a device containing 16 KB of SRAM the ranges are:

- SRAM\_L:  $0x1FFF\_F000 - 0x1FFF\_FFFF$
- SRAM\_U:  $0x2000\_0000 - 0x2000\_2FFF$

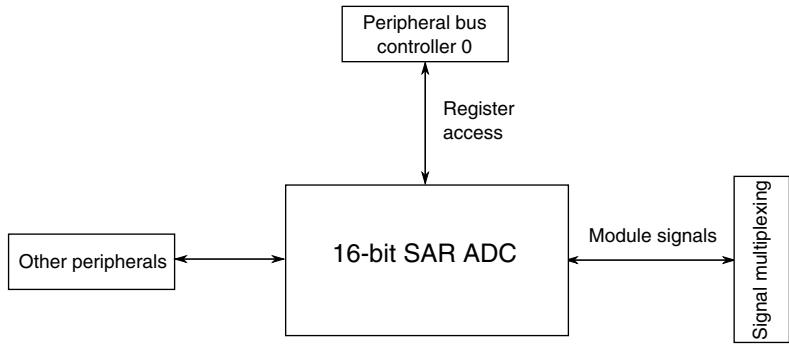
### 3.6.3.3 SRAM retention in low power modes

The SRAM is retained down to VLLS3 mode. In VLLS1 and VLLS0 no SRAM is retained.

## 3.7 Analog

### 3.7.1 16-bit SAR ADC Configuration

This section summarizes how the module has been configured in the chip. For a comprehensive description of the module itself, see the module's dedicated chapter.



**Figure 3-21. 16-bit SAR ADC configuration**

**Table 3-31. Reference links to related information**

| Topic               | Related module | Reference                           |
|---------------------|----------------|-------------------------------------|
| Full description    | 16-bit SAR ADC | <a href="#">16-bit SAR ADC</a>      |
| System memory map   |                | <a href="#">System memory map</a>   |
| Clocking            |                | <a href="#">Clock distribution</a>  |
| Power management    |                | <a href="#">Power management</a>    |
| Signal multiplexing | Port control   | <a href="#">Signal multiplexing</a> |

### 3.7.1.1 ADC Instantiation Information

This device contains one 16 -bit successive approximation ADC with up to 16-channel.

The ADC supports both software and hardware triggers. The hardware trigger sources are listed in the [Module-to-Module section](#).

The number of ADC channels present on the device is determined by the pinout of the specific device package and is shown in the following table.

**Table 3-32. Number of KL25 ADC channels**

**Table 3-32. Number of KL25 ADC channels (continued)**

| Device        | Number of ADC channels |
|---------------|------------------------|
| MKL25Z64VLK4  | 14                     |
| MKL25Z128VLK4 | 14                     |

### 3.7.1.2 DMA Support on ADC

Applications may require continuous sampling of the ADC that may have considerable load on the CPU. The ADC supports DMA request functionality for higher performance when the ADC is sampled at a very high rate. The ADC can trigger the DMA (via DMA req) on conversion completion.

### 3.7.1.3 ADC0 Connections/Channel Assignment

#### NOTE

As indicated by the following sections, each ADCx\_DP<sub>x</sub> input and certain ADCx\_DM<sub>x</sub> inputs may operate as single-ended ADC channels in single-ended mode.

#### 3.7.1.3.1 ADC0 Channel Assignment

| ADC Channel<br>(SC1n[ADCH]) | Channel | Input signal<br>(SC1n[DIFF]= 1) | Input signal<br>(SC1n[DIFF]= 0) |
|-----------------------------|---------|---------------------------------|---------------------------------|
| 00000                       | DAD0    | ADC0_DP0 and ADC0_DM0           | ADC0_DP0/ADC0_SE0               |
| 00001                       | DAD1    | ADC0_DP1 and ADC0_DM1           | ADC0_DP1/ADC0_SE1               |
| 00010                       | DAD2    | ADC0_DP2 and ADC0_DM2           | ADC0_DP2/ADC0_SE2               |
| 00011                       | DAD3    | ADC0_DP3 and ADC0_DM3           | ADC0_DP3/ADC0_SE3               |
| 00100 <sup>1</sup>          | AD4a    | Reserved                        | ADC0_DM0/ADC0_SE4a              |
| 00101 <sup>1</sup>          | AD5a    | Reserved                        | ADC0_DM1/ADC0_SE5a              |
| 00110 <sup>1</sup>          | AD6a    | Reserved                        | ADC0_DM2/ADC0_SE6a              |
| 00111 <sup>1</sup>          | AD7a    | Reserved                        | ADC0_DM3/ADC0_SE7a              |
| 00100 <sup>1</sup>          | AD4b    | Reserved                        | ADC0_SE4b                       |
| 00101 <sup>1</sup>          | AD5b    | Reserved                        | ADC0_SE5b                       |
| 00110 <sup>1</sup>          | AD6b    | Reserved                        | ADC0_SE6b                       |
| 00111 <sup>1</sup>          | AD7b    | Reserved                        | ADC0_SE7b                       |
| 01000                       | AD8     | Reserved                        | ADC0_SE8                        |
| 01001                       | AD9     | Reserved                        | ADC0_SE9                        |
| 01010                       | AD10    | Reserved                        | Reserved                        |

Table continues on the next page...

## Analog

| ADC Channel<br>(SC1n[ADCH]) | Channel | Input signal<br>(SC1n[DIFF]= 1) | Input signal<br>(SC1n[DIFF]= 0)  |
|-----------------------------|---------|---------------------------------|----------------------------------|
| 01011                       | AD11    | Reserved                        | ADC0_SE11                        |
| 01100                       | AD12    | Reserved                        | ADC0_SE12                        |
| 01101                       | AD13    | Reserved                        | ADC0_SE13                        |
| 01110                       | AD14    | Reserved                        | ADC0_SE14                        |
| 01111                       | AD15    | Reserved                        | ADC0_SE15                        |
| 10000                       | AD16    | Reserved                        | Reserved                         |
| 10001                       | AD17    | Reserved                        | Reserved                         |
| 10010                       | AD18    | Reserved                        | Reserved                         |
| 10011                       | AD19    | Reserved                        | Reserved                         |
| 10100                       | AD20    | Reserved                        | Reserved                         |
| 10101                       | AD21    | Reserved                        | Reserved                         |
| 10110                       | AD22    | Reserved                        | Reserved                         |
| 10111                       | AD23    | Reserved                        | 12-bit DAC0 Output/<br>ADC0_SE23 |
| 11000                       | AD24    | Reserved                        | Reserved                         |
| 11001                       | AD25    | Reserved                        | Reserved                         |
| 11010                       | AD26    | Temperature Sensor (Diff)       | Temperature Sensor (S.E)         |
| 11011                       | AD27    | Bandgap (Diff) <sup>2</sup>     | Bandgap (S.E) <sup>2</sup>       |
| 11100                       | AD28    | Reserved                        | Reserved                         |
| 11101                       | AD29    | -VREFH (Diff)                   | VREFH (S.E)                      |
| 11110                       | AD30    | Reserved                        | VREFL                            |
| 11111                       | AD31    | Module Disabled                 | Module Disabled                  |

1. ADCx\_CFG2[MUXSEL] bit selects between ADCx\_SEn channels a and b. Refer to MUXSEL description in ADC chapter for details.
2. This is the PMC bandgap 1V reference voltage. Prior to reading from this ADC channel, ensure that you enable the bandgap buffer by setting the PMC\_REGSC[BGBE] bit. Refer to the device data sheet for the bandgap voltage ( $V_{BG}$ ) specification.

### 3.7.1.4 ADC Analog Supply and Reference Connections

This device includes dedicated VDDA and VSSA pins.

This device contains separate VREFH and VREFL pins on 48-pin and higher devices. These pins are internally connected to VDDA and VSSA respectively, on 32-pin devices.

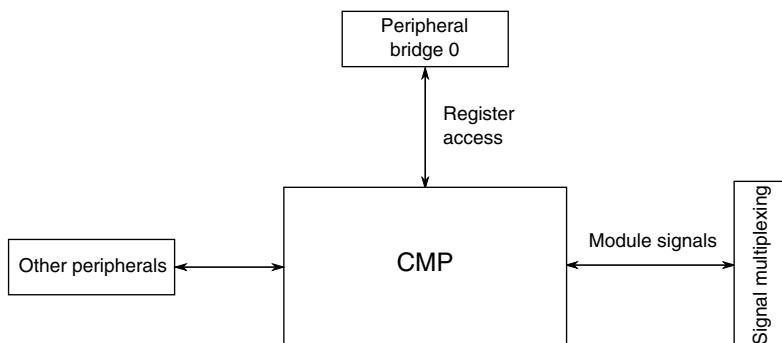
### 3.7.1.5 ADC Reference Options

The ADC supports the following references:

- VREFH/VREFL - connected as the primary reference option
- VDDA - connected as the  $V_{ALT}$  reference option

### 3.7.2 CMP Configuration

This section summarizes how the module has been configured in the chip. For a comprehensive description of the module itself, see the module's dedicated chapter.



**Figure 3-22. CMP configuration**

**Table 3-33. Reference links to related information**

| Topic               | Related module   | Reference                           |
|---------------------|------------------|-------------------------------------|
| Full description    | Comparator (CMP) | <a href="#">Comparator</a>          |
| System memory map   |                  | <a href="#">System memory map</a>   |
| Clocking            |                  | <a href="#">Clock distribution</a>  |
| Power management    |                  | <a href="#">Power management</a>    |
| Signal multiplexing | Port control     | <a href="#">Signal multiplexing</a> |

#### 3.7.2.1 CMP Instantiation Information

The device includes one high speed comparator and two 8-input multiplexors for both the inverting and non-inverting inputs of the comparator. Each CMP input channel connects to both muxes. Two of the channels are connected to internal sources, leaving resources to support up to 6 input pins. See the channel assignment table for a summary of CMP input connections for this device.

The CMP also includes one 6-bit DAC with a 64-tap resistor ladder network, which provides a selectable voltage reference for applications where voltage reference is needed for internal connection to the CMP.

The CMP can be optionally on in all modes except VLLS0.

The CMP has several module to module interconnects in order to facilitate ADC triggering, TPM triggering and UART IR interfaces. For complete details on the CMP module interconnects please refer to the [Module-to-Module section](#).

## Analog

The CMP does not support window compare function and CMP\_CR1[WE] must always be written to 0. The sample function has limited functionality since the SAMPLE input to the block is not connected to a valid input. Usage of sample operation is limited to a divided version of the bus clock (CMP\_CR1[SE] = 0).

Due to the pin number limitation, the CMP pass through mode is not supported by this device, so the CMPx\_MUXCR[PSTM] must be left as 0.

### 3.7.2.2 CMP input connections

The following table shows the fixed internal connections to the CMP.

**Table 3-34. CMP input connections**

| CMP Inputs | CMP0                            |
|------------|---------------------------------|
| IN0        | CMP0_IN0                        |
| IN1        | CMP0_IN1                        |
| IN2        | CMP0_IN2                        |
| IN3        | CMP0_IN3                        |
| IN4        | 12-bit DAC0 reference/ CMP0_IN4 |
| IN5        | CMP0_IN5                        |
| IN6        | Bandgap <sup>1</sup>            |
| IN7        | 6-bit DAC0 reference            |

1. This is the PMC bandgap 1V reference voltage. Prior to using as CMP input, ensure that you enable the bandgap buffer by setting the PMC\_REGSC[BGBE] bit. Refer to the device data sheet for the bandgap voltage ( $V_{BG}$ ) specification.

### 3.7.2.3 CMP external references

The 6-bit DAC sub-block supports selection of two references. For this device, the references are connected as follows:

- VREFH -  $V_{in1}$  input. When using VREFH, any ADC conversion using this same reference at the same time is negatively impacted.
- VDD -  $V_{in2}$  input

### 3.7.2.4 CMP trigger mode

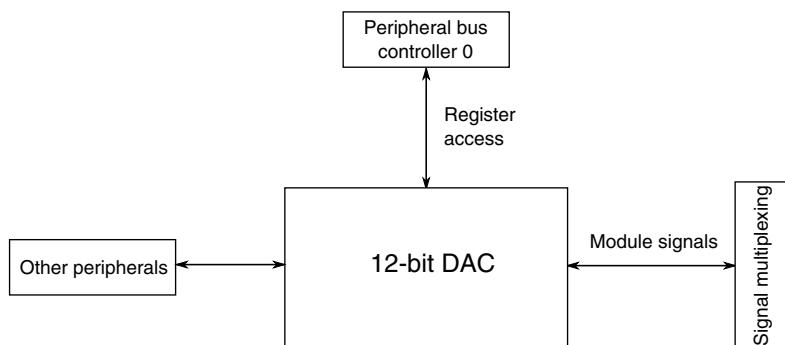
The CMP and 6-bit DAC sub-block supports trigger mode operation when the CMP\_CR1[TRIGM] is set. When trigger mode is enabled, the trigger event will initiate a compare sequence that must first enable the CMP and DAC prior to performing a CMP operation and capturing the output. In this device, control for this two staged sequencing

is provided from the LPTMR. The LPTMR triggering output is always enabled when the LPTMR is enabled. The first signal is supplied to enable the CMP and DAC and is asserted at the same time as the TCF flag is set. The delay to the second signal that triggers the CMP to capture the result of the compare operation is dependent on the LPTMR configuration. In Time Counter mode with prescaler enabled, the delay is 1/2 Prescaler output period. In Time Counter mode with prescaler bypassed, the delay is 1/2 Prescaler clock period.

The delay between the first signal from LPTMR and the second signal from LPTMR must be greater than the Analog comparator initialization delay as defined in the device datasheet.

### 3.7.3 12-bit DAC Configuration

This section summarizes how the module has been configured in the chip. For a comprehensive description of the module itself, see the module's dedicated chapter.



**Figure 3-23. 12-bit DAC configuration**

**Table 3-35. Reference links to related information**

| Topic               | Related module | Reference                           |
|---------------------|----------------|-------------------------------------|
| Full description    | 12-bit DAC     | <a href="#">12-bit DAC</a>          |
| System memory map   |                | <a href="#">System memory map</a>   |
| Clocking            |                | <a href="#">Clock distribution</a>  |
| Power management    |                | <a href="#">Power management</a>    |
| Signal multiplexing | Port control   | <a href="#">Signal multiplexing</a> |

#### 3.7.3.1 12-bit DAC Instantiation Information

This device contains one 12-bit digital-to-analog converter (DAC) with programmable reference generator output. The DAC includes a two word FIFO for DMA support.

### 3.7.3.2 12-bit DAC Output

The output of the DAC can be placed on an external pin or selected as an input to the analog comparator or ADC.

### 3.7.3.3 12-bit DAC Analog Supply and Reference Connections

This device includes dedicated VDDA and VSSA pins.

This device contains separate VREFH and VREFL pins on 48-pin and higher devices. These pins are internally connected to VDDA and VSSA respectively, on 32-pin devices.

### 3.7.3.4 12-bit DAC Reference

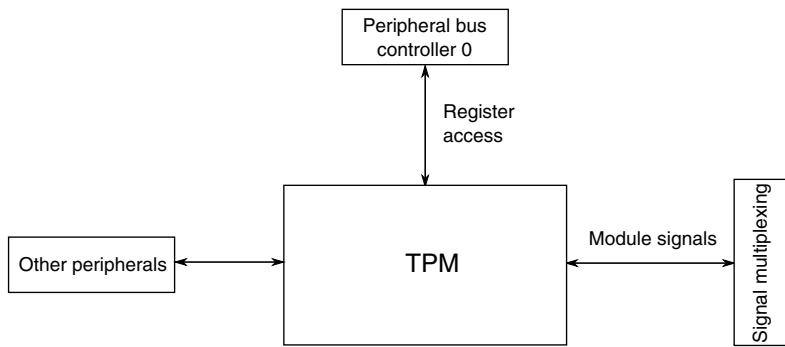
For this device VREFH and VDDA are selectable as the DAC reference. VREFH is connected to the DACREF\_1 input and VDDA is connected to the DACREF\_2 input. Use DACx\_C0[DACRFS] control bit to select between these two options.

Be aware that if the DAC and ADC use the same reference simultaneously, some degradation of ADC accuracy is to be expected due to DAC switching.

## 3.8 Timers

### 3.8.1 Timer/PWM Module Configuration

This section summarizes how the module has been configured in the chip. For a comprehensive description of the module itself, see the module's dedicated chapter.

**Figure 3-24. TPM configuration****Table 3-36. Reference links to related information**

| Topic               | Related module   | Reference                           |
|---------------------|------------------|-------------------------------------|
| Full description    | Timer/PWM Module | <a href="#">Timer/PWM Module</a>    |
| System memory map   |                  | <a href="#">System memory map</a>   |
| Clocking            |                  | <a href="#">Clock distribution</a>  |
| Power management    |                  | <a href="#">Power management</a>    |
| Signal multiplexing | Port control     | <a href="#">Signal multiplexing</a> |

### 3.8.1.1 TPM Instantiation Information

This device contains three Low Power TPM modules (TPM). All TPM modules in the device only are configured as basic TPM function, and no quadrature decoder function and all can be functional in Stop/VLPS mode. The clock source is either external or internal in Stop/VLPS mode.

The following table shows how these modules are configured.

**Table 3-37. TPM configuration**

| TPM instance | Number of channels | Features/usage                          |
|--------------|--------------------|---|
| TPM0         | 6                  | Basic TPM, functional in Stop/VLPS mode |
| TPM1         | 2                  | Basic TPM, functional in Stop/VLPS mode |
| TPM2         | 2                  | Basic TPM, functional in Stop/VLPS mode |

There are several connections to and from the TPMs in order to facilitate customer use cases. For complete details on the TPM module interconnects please refer to the [Module-to-Module section](#).

### 3.8.1.2 Clock Options

The TPM blocks are clocked from a single TPM clock that can be selected from OSCERCLK, MCGIRCLK, MCGPLLCLK/2, or MCGFLLCLK. The selected source is controlled by SIM\_SOPT2[TPMSRC] and SIM\_SOPT2[PLLFLSEL] control registers.

Each TPM also supports an external clock mode (TPM\_SC[CMOD]=1x) in which the counter increments after a synchronized (to the selected TPM clock source) rising edge detect of an external clock input. The available external clock (either TPM\_CLKIN0 or TPM\_CLKIN1) is selected by SIM\_SOPT4[TPMxCLKSEL] control register. To guarantee valid operation the selected external clock must be less than half the frequency of the selected TPM clock source.

### 3.8.1.3 Trigger Options

Each TPM has a selectable trigger input source controlled by the TPMx\_CONF[TRGSEL] field to use for starting the counter and/or reloading the counter. The options available are shown in the following table.

**Table 3-38. TPM trigger options**

| TPMx_CONF[TRGSEL] | Selected source                       |
|-------------------|---------------------------------------|
| 0000              | External trigger pin input (EXTRG_IN) |
| 0001              | CMP0 output                           |
| 0010              | Reserved                              |
| 0011              | Reserved                              |
| 0100              | PIT trigger 0                         |
| 0101              | PIT trigger 1                         |
| 0110              | Reserved                              |
| 0111              | Reserved                              |
| 1000              | TPM0 overflow                         |
| 1001              | TPM1 overflow                         |
| 1010              | TPM2 overflow                         |
| 1011              | Reserved                              |
| 1100              | RTC alarm                             |
| 1101              | RTC seconds                           |
| 1110              | LPTMR trigger                         |
| 1111              | Reserved                              |

### 3.8.1.4 Global Timebase

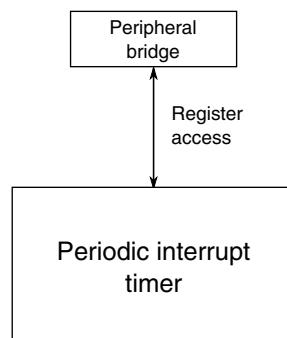
Each TPM has a global timebase feature controlled by the TPMx\_CONF[GTBEN] bit. TPM1 is configured as the global time when this option is enabled.

### 3.8.1.5 TPM Interrupts

The TPM has multiple sources of interrupt. However, these sources are OR'd together to generate a single interrupt request to the interrupt controller. When an TPM interrupt occurs, read the TPM status registers to determine the exact interrupt source.

## 3.8.2 PIT Configuration

This section summarizes how the module has been configured in the chip. For a comprehensive description of the module itself, see the module's dedicated chapter.



**Figure 3-25. PIT configuration**

**Table 3-39. Reference links to related information**

| Topic             | Related module | Reference                          |
|-------------------|----------------|------------------------------------|
| Full description  | PIT            | <a href="#">PIT</a>                |
| System memory map |                | <a href="#">System memory map</a>  |
| Clocking          |                | <a href="#">Clock Distribution</a> |
| Power management  |                | <a href="#">Power management</a>   |

### 3.8.2.1 PIT/DMA Periodic Trigger Assignments

The PIT generates periodic trigger events to the DMA channel mux as shown in the table below.

**Table 3-40. PIT channel assignments for periodic DMA triggering**

| PIT Channel   | DMA Channel Number |
|---------------|--------------------|
| PIT Channel 0 | DMA Channel 0      |
| PIT Channel 1 | DMA Channel 1      |

### 3.8.2.2 PIT/ADC Triggers

PIT triggers are selected as ADCx trigger sources using the SOPT7[ADCxTRGSEL] bits in the SIM module. For more details, refer to SIM chapter.

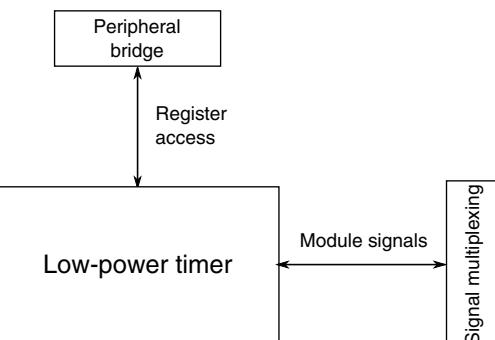
### 3.8.2.3 PIT/TPM Triggers

PIT triggers are selected as TPMx trigger sources using the TPMx\_CONF[TRGSEL] bits in the TPM module. For more details, refer to TPM chapter.

### 3.8.2.4 PIT/DAC Triggers

PIT Channel 0 is configured as the DAC hardware trigger source. For more details, refer to DAC chapter.

## 3.8.3 Low-power timer configuration

**Figure 3-26. LPT configuration****Table 3-41. Reference links to related information**

| Topic            | Related module  | Reference                       |
|------------------|-----------------|---------------------------------|
| Full description | Low-power timer | <a href="#">Low-power timer</a> |

*Table continues on the next page...*

**Table 3-41. Reference links to related information (continued)**

| Topic               | Related module | Reference                           |
|---------------------|----------------|-------------------------------------|
| System memory map   |                | <a href="#">System memory map</a>   |
| Clocking            |                | <a href="#">Clock Distribution</a>  |
| Power management    |                | <a href="#">Power management</a>    |
| Signal Multiplexing | Port control   | <a href="#">Signal Multiplexing</a> |

### 3.8.3.1 LPTMR Instantiation Information

The low-power timer (LPTMR) allows operation during all power modes. The LPTMR can operate as a real-time interrupt or pulse accumulator. It includes a 15-bit prescaler (real-time interrupt mode) or glitch filter (pulse accumulator mode).

The LPTMR can be clocked from the internal reference clock, the internal 1 kHz LPO, OSCERCLK, or an external 32.768 kHz crystal. In VLLS0 mode, the clocking option is limited to an external pin with the OSC configured for bypass (external clock) operation.

An interrupt is generated (and the counter may reset) when the counter equals the value in the 16-bit compare register.

### 3.8.3.2 LPTMR pulse counter input options

The LPTMR\_CSR[TPS] bitfield configures the input source used in pulse counter mode. The following table shows the chip-specific input assignments for this bitfield.

| LPTMR_CSR[TPS] | Pulse counter input number | Chip input     |
|----------------|----------------------------|----------------|
| 00             | 0                          | CMP0 output    |
| 01             | 1                          | LPTMR_ALT1 pin |
| 10             | 2                          | LPTMR_ALT2 pin |
| 11             | 3                          | LPTMR_ALT3 pin |

### 3.8.3.3 LPTMR prescaler/glitch filter clocking options

The prescaler and glitch filter of the LPTMR module can be clocked from one of four sources determined by the LPTMR0\_PSR[PCS] bitfield. The following table shows the chip-specific clock assignments for this bitfield.

**NOTE**

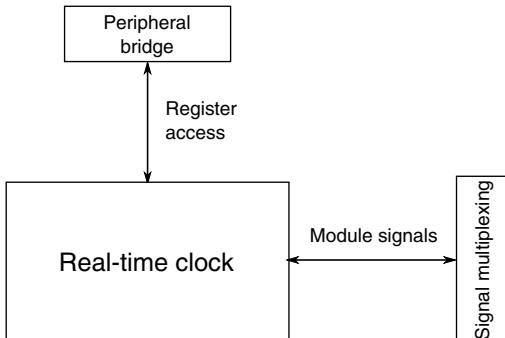
The chosen clock must remain enabled if the LPTMR is to continue operating in all required low-power modes.

| LPTMR0_PSR[PCS] | Prescaler/glitch filter clock number | Chip clock  |
|-----------------|--------------------------------------|---|
| 00              | 0                                    | MCGIRCLK — internal reference clock (not available in LLS and VLSS modes) |
| 01              | 1                                    | LPO — 1 kHz clock (not available in VLSS0 mode)                           |
| 10              | 2                                    | ERCLK32K (not available in VLSS0 mode when using 32 kHz oscillator)       |
| 11              | 3                                    | OSCERCLK — external reference clock (not available in VLSS0 mode)         |

See [Clock Distribution](#) for more details on these clocks.

### 3.8.4 RTC configuration

This section summarizes how the module has been configured in the chip. For a comprehensive description of the module itself, see the module's dedicated chapter.



**Figure 3-27. RTC configuration**

**Table 3-42. Reference links to related information**

| Topic             | Related module | Reference                          |
|-------------------|----------------|------------------------------------|
| Full description  | RTC            | <a href="#">RTC</a>                |
| System memory map |                | <a href="#">System memory map</a>  |
| Clocking          |                | <a href="#">Clock Distribution</a> |
| Power management  |                | <a href="#">Power management</a>   |

### 3.8.4.1 RTC Instantiation Information

RTC prescaler is clocked by ERCLK32K.

RTC is reset on POR Only.

RTC\_CR[OSCE] can override the configuration of the System OSC, configuring the OSC for 32kHz crystal operation in all power modes (except VLLS0) and through any System Reset. When OSCE is enabled, the RTC also overrides the capacitor configurations.

### 3.8.4.2 RTC\_CLKOUT options

RTC\_CLKOUT pin can be driven either with the RTC 1 Hz output or with the OSCERCLK on-chip clock source. Control for this option is through SIM\_SOPT2[RTCCLKOUTSEL] bit.

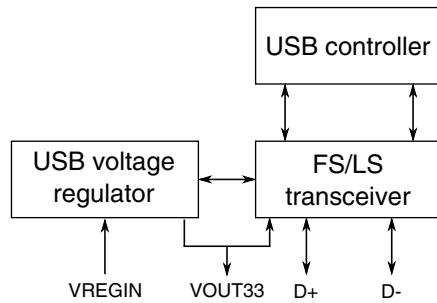
When RTCCLKOUTSEL=0, the RTC 1 Hz clock is output is selected on the RTC\_CLKOUT pin. When RTCCLKOUTSEL=1, OSCERCLK clock is output on the RTC\_CLKOUT pin.

## 3.9 Communication interfaces

### 3.9.1 Universal Serial Bus (USB) FS Subsystem

The USB FS subsystem includes these components:

- Dual-role USB OTG-capable (On-The-Go) controller that supports a full-speed (FS) device or FS/LS host. The module complies with the USB 2.0 specification.
- USB transceiver that includes internal 15 kΩ pulldowns on the D+ and D- lines for host mode functionality.
- A 3.3 V regulator.
- VBUS detect signal: To detect a valid VBUS in device mode, use a GPIO signal that can wake the chip in all power modes.

**Figure 3-28. USB Subsystem Overview**

### 3.9.1.1 USB Wakeup

When the USB detects that there is no activity on the USB bus for more than 3 ms, the INT\_STAT[SLEEP] bit is set. This bit can cause an interrupt and software decides the appropriate action.

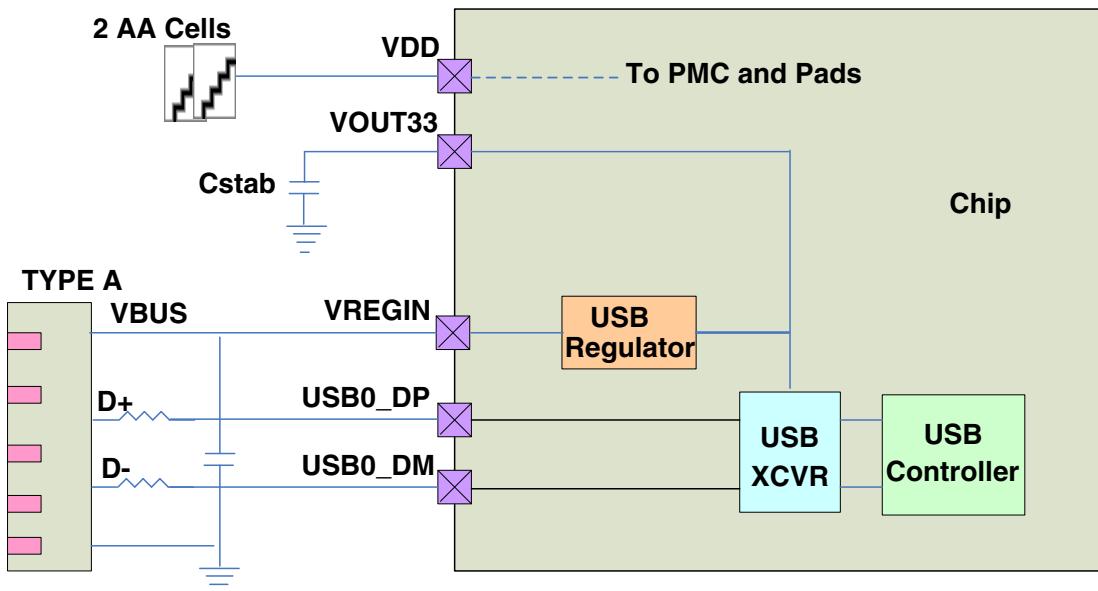
Waking from a low power mode (except in LLS/VLLS mode where USB is not powered) occurs through an asynchronous interrupt triggered by activity on the USB bus. Setting the USBTRC0[USBRESMEN] bit enables this function.

### 3.9.1.2 USB Power Distribution

This chip includes an internal 5 V to 3.3 V USB regulator that powers the USB transceiver or the MCU (depending on the application).

#### 3.9.1.2.1 AA/AAA cells power supply

The chip can be powered by two AA/AAA cells. In this case, the MCU is powered through VDD which is within the 1.8 to 3.0 V range. After USB cable insertion is detected, the USB regulator is enabled to power the USB transceiver.



**Figure 3-29. USB regulator AA cell usecase**

### 3.9.1.2.2 Li-Ion battery power supply

The chip can also be powered by a single Li-ion battery. In this case, VOUT33 is connected to VDD. The USB regulator must be enabled by default to power the MCU. When connected to a USB host, the input source of this regulator is switched to the USB bus supply from the Li-ion battery. To charge the battery, the MCU can configure the battery charger.

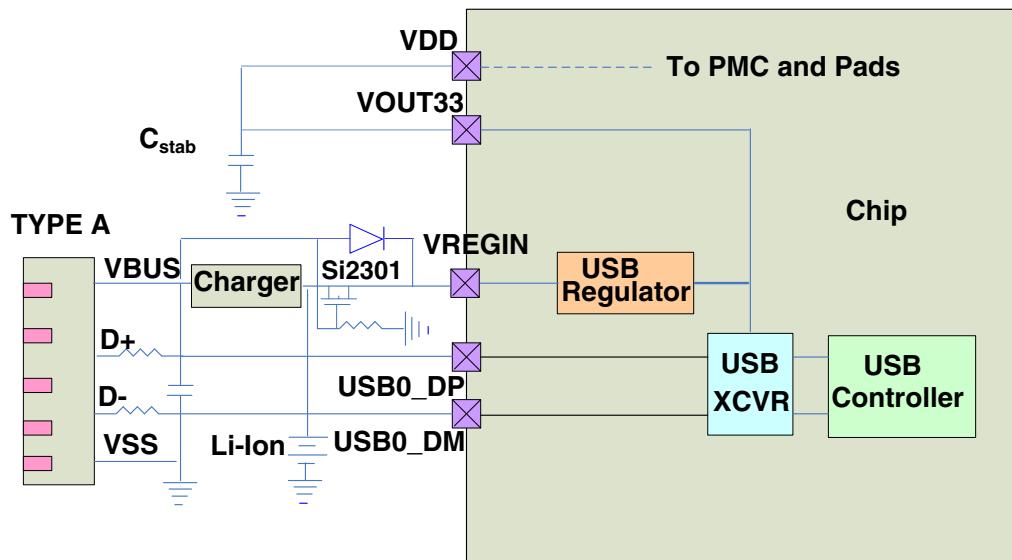


Figure 3-30. USB regulator Li-ion usecase

### 3.9.1.2.3 USB bus power supply

The chip can also be powered by the USB bus directly. In this case, VOUT33 is connected to VDD. The USB regulator must be enabled by default to power the MCU, then to power USB transceiver or external sensor.

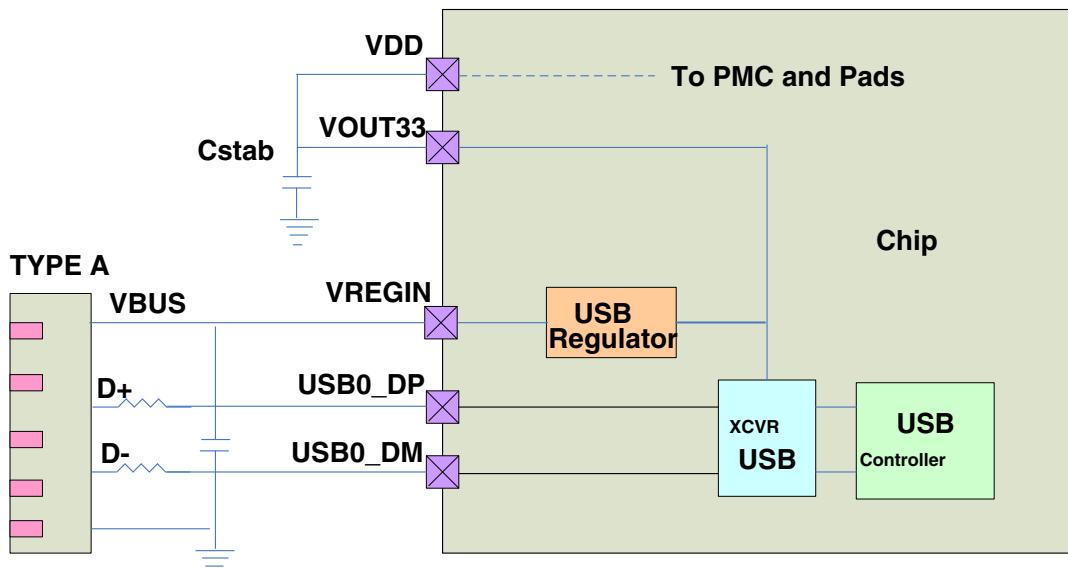


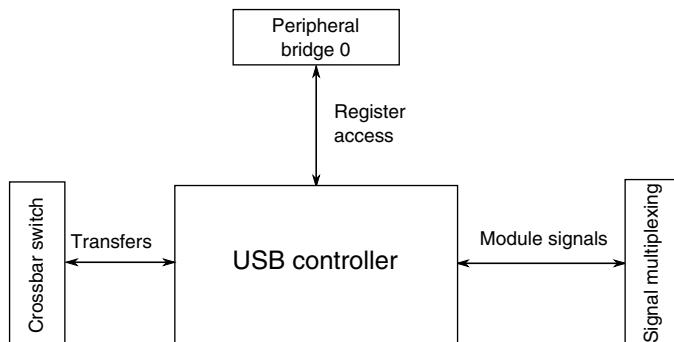
Figure 3-31. USB regulator bus supply

### 3.9.1.3 USB power management

The regulator should be put into STANDBY mode whenever the chip is in Stop mode.

### 3.9.1.4 USB controller configuration

This section summarizes how the module has been configured in the chip. For a comprehensive description of the module itself, see the module's dedicated chapter.



**Figure 3-32. USB controller configuration**

**Table 3-43. Reference links to related information**

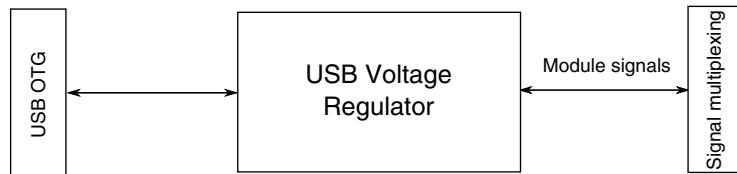
| Topic               | Related module  | Reference                           |
|---------------------|-----------------|-------------------------------------|
| Full description    | USB controller  | <a href="#">USB controller</a>      |
| System memory map   |                 | <a href="#">System memory map</a>   |
| Clocking            |                 | <a href="#">Clock Distribution</a>  |
| Transfers           | Crossbar switch | <a href="#">Crossbar switch</a>     |
| Signal Multiplexing | Port control    | <a href="#">Signal Multiplexing</a> |

#### NOTE

When USB is not used in the application, it is recommended that the USB regulator VREGIN and VOUT33 pins remain floating.

### 3.9.1.5 USB Voltage Regulator Configuration

This section summarizes how the module has been configured in the chip. For a comprehensive description of the module itself, see the module's dedicated chapter.

**Figure 3-33. USB Voltage Regulator configuration****Table 3-44. Reference links to related information**

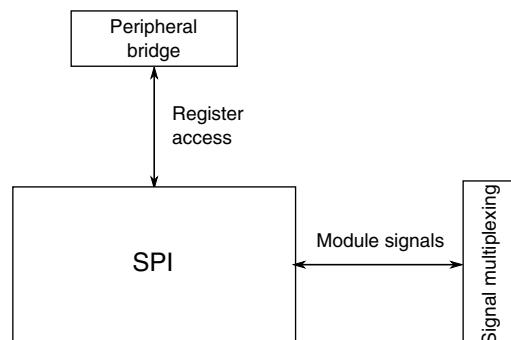
| Topic               | Related module        | Reference                             |
|---------------------|-----------------------|---------------------------------------|
| Full description    | USB Voltage Regulator | <a href="#">USB Voltage Regulator</a> |
| System memory map   |                       | <a href="#">System memory map</a>     |
| Clocking            |                       | <a href="#">Clock Distribution</a>    |
|                     | USB controller        | <a href="#">USB controller</a>        |
| Signal Multiplexing | Port control          | <a href="#">Signal Multiplexing</a>   |

**NOTE**

When USB is not used in the application, it is recommended that the USB regulator VREGIN and VOUT33 pins remain floating.

### 3.9.2 SPI configuration

This section summarizes how the module has been configured in the chip. For a comprehensive description of the module itself, see the module's dedicated chapter.

**Figure 3-34. SPI configuration****Table 3-45. Reference links to related information**

| Topic             | Related module | Reference                          |
|-------------------|----------------|------------------------------------|
| Full description  | SPI            | <a href="#">SPI</a>                |
| System memory map |                | <a href="#">System memory map</a>  |
| Clocking          |                | <a href="#">Clock Distribution</a> |

*Table continues on the next page...*

**Table 3-45. Reference links to related information (continued)**

| Topic               | Related module | Reference                           |
|---------------------|----------------|-------------------------------------|
| Signal Multiplexing | Port control   | <a href="#">Signal Multiplexing</a> |

### 3.9.2.1 SPI Instantiation Information

This device contains two SPI module that supports 8-bit data length.

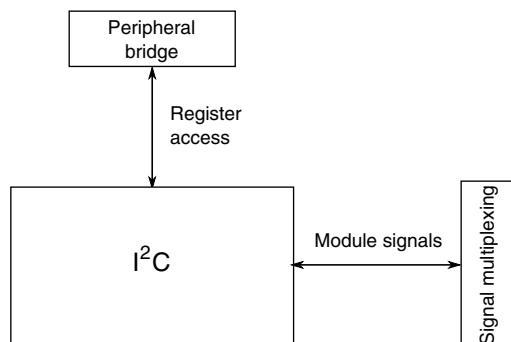
SPI0 is clocked on the bus clock. SPI1 is clocked from the system clock. SPI1 is therefore disabled in "Partial Stop Mode".

The SPI supports DMA request and can operate in VLPS mode. When the SPI is operating in VLPS mode, it will operate as a slave.

SPI can wakeup MCU from VLPS mode upon reception of SPI data in slave mode.

### 3.9.3 I2C Configuration

This section summarizes how the module has been configured in the chip. For a comprehensive description of the module itself, see the module's dedicated chapter.

**Figure 3-35. I2C configuration****Table 3-46. Reference links to related information**

| Topic               | Related module | Reference                           |
|---------------------|----------------|-------------------------------------|
| Full description    | I2C            | <a href="#">I2C</a>                 |
| System memory map   |                | <a href="#">System memory map</a>   |
| Clocking            |                | <a href="#">Clock Distribution</a>  |
| Power management    |                | <a href="#">Power management</a>    |
| Signal Multiplexing | Port control   | <a href="#">Signal Multiplexing</a> |

### 3.9.3.1 IIC Instantiation Information

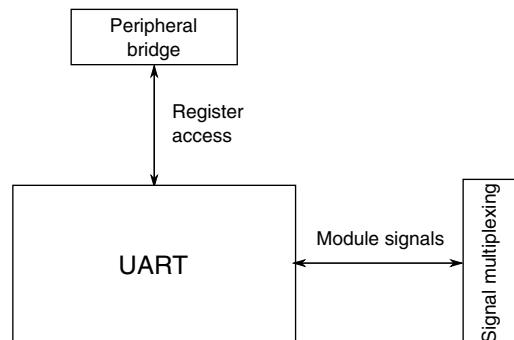
This device has two IIC module.

When the package pins associated with IIC have their mux select configured for IIC operation, the pins (SCL and SDA) are driven in a pseudo open drain configuration.

The digital glitch filter implemented in the IICx module, controlled by the I2Cx\_FLT[FLT] registers, is clocked from the bus clock and thus has filter granularity in bus clock cycle counts.

### 3.9.4 UART Configuration

This section summarizes how the module has been configured in the chip. For a comprehensive description of the module itself, see the module's dedicated chapter.



**Figure 3-36. UART configuration**

**Table 3-47. Reference links to related information**

| Topic               | Related module  | Reference                           |
|---------------------|-----------------|-------------------------------------|
| Full description    | UART1 and UART2 | <a href="#">UART</a>                |
| Full description    | UART0           | <a href="#">UART</a>                |
| System memory map   |                 | <a href="#">System memory map</a>   |
| Clocking            |                 | <a href="#">Clock Distribution</a>  |
| Power management    |                 | <a href="#">Power management</a>    |
| Signal Multiplexing | Port control    | <a href="#">Signal Multiplexing</a> |

### 3.9.4.1 UART0 overview

The UART0 module supports basic UART with DMA interface function, x4 to x32 oversampling of baud-rate.

This module supports LIN slave operation.

The module can remain functional in VLPS mode provided the clock it is using remains enabled.

ISO7816 protocol is intended to be handled in software for this product. To support smart card reading, TxD pin can be configured as pseudo open drain for 1-wire half-duplex like ISO7816 communication via the SIM\_SOPT5[UART0ODE] bit.

### 3.9.4.2 UART1 and UART2 Overview

This device contains two basic universal asynchronous receiver/transmitter (UART) modules with DMA function support. Generally, these modules are used in RS-232, RS-485, and other communications. This module supports LIN Slave operation.

## 3.10 Human-machine interfaces (HMI)

### 3.10.1 GPIO Configuration

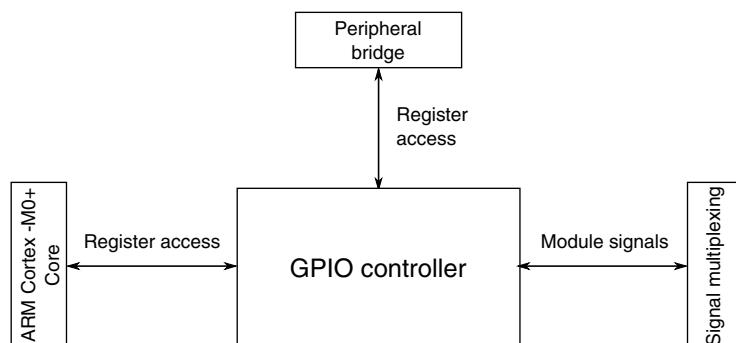


Figure 3-37. GPIO configuration

Table 3-48. Reference links to related information

| Topic             | Related module | Reference                          |
|-------------------|----------------|------------------------------------|
| Full description  | GPIO           | <a href="#">GPIO</a>               |
| System memory map |                | <a href="#">System memory map</a>  |
| Clocking          |                | <a href="#">Clock Distribution</a> |

Table continues on the next page...

**Table 3-48. Reference links to related information (continued)**

| Topic               | Related module  | Reference                           |
|---------------------|-----------------|-------------------------------------|
| Power management    |                 | <a href="#">Power management</a>    |
| Crossbar switch     | Crossbar switch | <a href="#">Crossbar switch</a>     |
| Signal Multiplexing | Port control    | <a href="#">Signal Multiplexing</a> |

### 3.10.1.1 GPIO Instantiation Information

The device includes four pins, PTB0, PTB1, PTD6, and PTD7, with high current drive capability. These pins can be used to drive LED or power MOSFET directly. The high drive capability applies to all functions which are multiplexed on these pins (UART, TPM, SPI...etc)

#### 3.10.1.1.1 Pull Devices and Directions

The pull devices are enabled out of POR only on RESET\_B, NMI\_b and respective SWD signals. Other pins can be enabled by writing to PORTx\_PCRn[PE] field.

All the pins are hard wired to be pullup except for SWD\_CLK. The state will be reflected in the PORTx\_PCRn[PS] field.

### 3.10.1.2 Port Control and Interrupt Summary

The following table provides more information regarding the Port Control and Interrupt configurations .

**Table 3-49. Ports Summary**

| Feature                  | Port A  | Port B   | Port C   | Port D   | Port E   |
|--------------------------|---|----------|----------|----------|----------|
| Pull Select control      | No  | No       | No       | No       | No       |
| Pull Select at reset     | PTA0=Pull down,<br>Others=Pull up                       | Pull up  | Pull up  | Pull up  | Pull up  |
| Pull Enable control      | Yes   | Yes      | Yes      | Yes      | Yes      |
| Pull Enable at reset     | PTA0/PTA3/PTA4/<br>RESET_b=Enabled<br>; Others=Disabled | Disabled | Disabled | Disabled | Disabled |
| Slew Rate Enable control | No  | No       | No       | No       | No       |

*Table continues on the next page...*

**Table 3-49. Ports Summary (continued)**

| Feature                                | Port A   | Port B   | Port C   | Port D  | Port E  |
|--|--|--|--|---|---|
| Slew Rate Enable at reset              | PTA3/PTA14/<br>PTA15/PTA16/<br>PTA17=Disabled;<br>Others=Enabled | PTB10/PTB11/<br>PTB16/PTB17 =<br>Disabled;<br>Others=Enabled | PTC3/PTC4/PTC5/<br>PTC6/<br>PTC7=Disabled;<br>Others=Enabled | PTD4/PTD5/PTD6/<br>PTD7=Disabled;<br>Others=Enabled | PTE16/PTE17/<br>PTE18/<br>PTE19=Disabled;<br>Others=Enabled |
| Passive Filter Enable control          | PTA4 and<br>RESET_b only   | No   | No   | No  | No  |
| Passive Filter Enable at reset         | RESET_b=Enabled ;<br>Others=Disabled                             | Disabled   | Disabled   | Disabled  | Disabled  |
| Open Drain Enable control <sup>1</sup> | No   | No   | No   | No  | No  |
| Open Drain Enable at reset             | Disabled   | Disabled   | Disabled   | Disabled  | Disabled  |
| Drive Strength Enable control          | No   | PTB0/PTB1 only   | No   | PTD6/PTD7 only                                      | No  |
| Drive Strength Enable at reset         | Disabled   | Disabled   | Disabled   | Disabled  | Disabled  |
| Pin Mux control                        | Yes  | Yes  | Yes  | Yes   | Yes   |
| Pin Mux at reset                       | PTA0/PTA3/<br>PTA4=ALT7;<br>Others=ALT0                          | ALT0   | ALT0   | ALT0  | ALT0  |
| Lock Bit                               | No   | No   | No   | No  | No  |
| Interrupt and DMA Request              | Yes  | No   | No   | Yes   | No  |
| Digital Glitch Filter                  | No   | No   | No   | No  | No  |

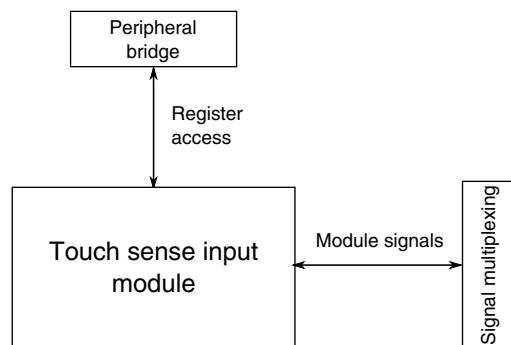
1. UART signals can be configured for open-drain using SIM\_SOPT5 register. IIC signals are automatically enabled for open drain when selected.

### 3.10.1.3 GPIO accessibility in the memory map

The GPIO is multi-ported and can be accessed directly by the core with zero wait states at base address 0xF800\_0000. It can also be accessed by the core and DMA masters through the cross bar/AIPS interface at 0x400F\_F000 and at an aliased slot (15) at address 0x4000\_F000. All BME operations to the GPIO space can be accomplished referencing the aliased slot (15) at address 0x4000\_F000. Only some of the BME operations can be accomplished referencing GPIO at address 0x400F\_F000.

### 3.10.2 TSI Configuration

This section summarizes how the module has been configured in the chip. For a comprehensive description of the module itself, see the module's dedicated chapter.

**Figure 3-38. TSI configuration****Table 3-50. Reference links to related information**

| Topic               | Related module | Reference                           |
|---------------------|----------------|-------------------------------------|
| Full description    | TSI            | <a href="#">TSI</a>                 |
| System memory map   |                | <a href="#">System memory map</a>   |
| Clocking            |                | <a href="#">Clock Distribution</a>  |
| Power management    |                | <a href="#">Power management</a>    |
| Signal Multiplexing | Port control   | <a href="#">Signal Multiplexing</a> |

### 3.10.2.1 TSI Instantiation Information

This device includes one TSI module containing the channels as shown in the following table. In Stop, VLPS, LLS and VLLSx modes any one channel can be enabled to be the wakeup source.

TSI hardware trigger is from the LPTMR. For complete details on the LPTMR module interconnects refer to the [Module-to-Module section](#).

The number of TSI channels present on the device is determined by the pinout of the specific device package and is shown in the following table.

**Table 3-51. Number of KL25 TSI channels**

| Device        | TSI channels |
|---------------|--------------|
| MKL25Z32VFM4  | 9            |
| MKL25Z64VFM4  | 9            |
| MKL25Z128VFM4 | 9            |
| MKL25Z32VFT4  | 14           |
| MKL25Z64VFT4  | 14           |
| MKL25Z128VFT4 | 14           |
| MKL25Z32VLH4  | 16           |
| MKL25Z64VLH4  | 16           |

*Table continues on the next page...*

**Table 3-51. Number of KL25 TSI channels (continued)**

| Device        | TSI channels |
|---------------|--------------|
| MKL25Z128VLH4 | 16           |
| MKL25Z32VLK4  | 16           |
| MKL25Z64VLK4  | 16           |
| MKL25Z128VLK4 | 16           |

### 3.10.2.2 TSI Interrupts

The TSI has multiple sources of interrupt requests. However, these sources are OR'd together to generate a single interrupt request. When a TSI interrupt occurs, read the TSI status register to determine the exact interrupt source.



# Chapter 4

## Memory Map

### 4.1 Introduction

This device contains various memories and memory-mapped peripherals which are located in a 4 GB memory space. This chapter describes the memory and peripheral locations within that memory space.

### 4.2 System memory map

The following table shows the high-level device memory map.

**Table 4-1. System memory map**

| System 32-bit Address Range          | Destination Slave   | Access                |
|--------------------------------------|---|-----------------------|
| 0x0000_0000–0x07FF_FFFF <sup>1</sup> | Program flash and read-only data<br>(Includes exception vectors in first 196 bytes)   | All masters           |
| 0x0800_0000–0x1FFF_EFFF              | Reserved  | —                     |
| 0x1FFF_F000–0x1FFF_FFFF <sup>2</sup> | SRAM_L: Lower SRAM  | All masters           |
| 0x2000_0000–0x2000_2FFF <sup>2</sup> | SRAM_U: Upper SRAM  | All masters           |
| 0x2000_3000–0x3FFF_FFFF              | Reserved  | —                     |
| 0x4000_0000–0x4007_FFFF              | AIPS Peripherals  | Cortex-M0+ core & DMA |
| 0x4008_0000–0x400F_EFFF              | Reserved  | —                     |
| 0x400F_F000–0x400F_FFFF              | General purpose input/output (GPIO)   | Cortex-M0+ core & DMA |
| 0x4010_0000–0x43FF_FFFF              | Reserved  | —                     |
| 0x4400_0000–0x5FFF_FFFF              | Bit Manipulation Engine (BME) access to AIPS Peripherals for slots 0–127 <sup>3</sup> | Cortex-M0+ core       |
| 0x6000_0000–0xDFFF_FFFF              | Reserved  | —                     |
| 0xE000_0000–0xE00F_FFFF              | Private Peripherals   | Cortex-M0+ core       |
| 0xE010_0000–0xFFFF_FFFF              | Reserved  | —                     |
| 0xF000_0000–0xF000_0FFF              | Micro Trace Buffer (MTB) registers  | Cortex-M0+ core       |

*Table continues on the next page...*

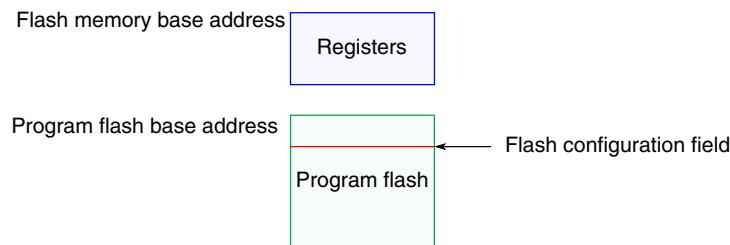
**Table 4-1. System memory map (continued)**

| System 32-bit Address Range | Destination Slave                                | Access          |
|-----------------------------|--|-----------------|
| 0xF000_1000–0xF000_1FFF     | MTB Data Watchpoint and Trace (MTBDWT) registers | Cortex-M0+ core |
| 0xF000_2000–0xF000_2FFF     | ROM table  | Cortex-M0+ core |
| 0xF000_3000–0xF000_3FFF     | Miscellaneous Control Module (MCM)               | Cortex-M0+ core |
| 0xF000_4000–0xF7FF_FFFF     | Reserved   | –               |
| 0xF800_0000–0xFFFF_FFFF     | IOPORT: GPIO (single cycle)                      | Cortex-M0+ core |

1. The program flash always begins at 0x0000\_0000 but the end of implemented flash varies depending on the amount of flash implemented for a particular device. See [Flash Memory Sizes](#) for details.
2. This range varies depending on SRAM sizes. See [SRAM Ranges](#) for details.
3. Includes BME operations to GPIO at slot 15 (based at 0x4000\_F000).

## 4.3 Flash Memory Map

The flash memory and the flash registers are located at different base addresses as shown in the following figure. The base address for each is specified in [System memory map](#).

**Figure 4-1. Flash memory map**

The on-chip Flash is implemented in a portion of the allocated Flash range to form a contiguous block in the memory map beginning at address 0x0000\_0000. See [Flash Memory Sizes](#) for details of supported ranges.

Accesses to the flash memory ranges outside the amount of Flash on the device causes the bus cycle to be terminated with an error followed by the appropriate response in the requesting bus master. Read collision events in which flash memory is accessed while a flash memory resource is being manipulated by a flash command also generates a bus error response.

### 4.3.1 Alternate Non-Volatile IRC User Trim Description

The following non-volatile locations (4 bytes) are reserved for custom IRC user trim supported by some development tools. An alternate IRC trim to the factory loaded trim can be stored at this location. To override the factory trim, user software must load new values into the MCG trim registers.

| Non-Volatile Byte Address | Alternate IRC Trim Value |
|---------------------------|--------------------------|
| 0x0000_03FC               | Reserved                 |
| 0x0000_03FD               | Reserved                 |
| 0x0000_03FE (bit 0)       | SCFTRIM                  |
| 0x0000_03FE (bit 4:1)     | FCTRIM                   |
| 0x0000_03FF               | SCTRIM                   |

## 4.4 SRAM memory map

The on-chip RAM is split between SRAM\_L and SRAM\_U. The RAM is also implemented such that the SRAM\_L and SRAM\_U ranges form a contiguous block in the memory map. See [SRAM Ranges](#) for details.

Accesses to the SRAM\_L and SRAM\_U memory ranges outside the amount of RAM on the device causes the bus cycle to be terminated with an error followed by the appropriate response in the requesting bus master.

## 4.5 Bit Manipulation Engine

The Bit Manipulation Engine (BME) provides hardware support for atomic read-modify-write memory operations to the peripheral address space. By combining the basic load and store instruction support in the Cortex-M instruction set architecture with the concept of decorated storage provided by the BME, the resulting implementation provides a robust and efficient read-modify-write capability to this class of ultra low-end microcontrollers. See the [Bit Manipulation Engine Block Guide \(BME\)](#) for a detailed description of BME functionality.

## 4.6 Peripheral bridge (AIPS-Lite) memory map

The peripheral memory map is accessible via one slave port on the crossbar in the 0x4000\_0000–0x400F\_FFFF region. The device implements one peripheral bridge that defines a 1024 KB address space.

The three regions associated with this space are:

- A 128 KB region, partitioned as 32 spaces, each 4 KB in size and reserved for on-platform peripheral devices. The AIPS controller generates unique module enables for all 32 spaces.
- A 384 KB region, partitioned as 96 spaces, each 4 KB in size and reserved for off-platform modules. The AIPS controller generates unique module enables for all 96 spaces.
- The last slot is a 4 KB region beginning at 0x400F\_F000 for accessing the GPIO module. The GPIO slot (slot 128) is an alias of slot 15. This block is also directly interfaced to the core and provides direct access without incurring wait states associated with accesses via the AIPS controller.

Modules that are disabled via their clock gate control bits in the SIM registers disable the associated AIPS slots. Access to any address within an unimplemented or disabled peripheral bridge slot results in a transfer error termination.

For programming model accesses via the peripheral bridges, there is generally only a small range within the 4 KB slots that is implemented. Accessing an address that is not implemented in the peripheral results in a transfer error termination.

### 4.6.1 Read-after-write sequence and required serialization of memory operations

In some situations, a write to a peripheral must be completed fully before a subsequent action can occur. Examples of such situations include:

- Exiting an interrupt service routine (ISR)
- Changing a mode
- Configuring a function

In these situations, application software must perform a read-after-write sequence to guarantee the required serialization of the memory operations:

1. Write the peripheral register.
2. Read the written peripheral register to verify the write.
3. Continue with subsequent operations.

## 4.6.2 Peripheral Bridge (AIPS-Lite) Memory Map

**Table 4-2. Peripheral bridge 0 slot assignments**

| System 32-bit base address | Slot number | Module                                   |
|----------------------------|-------------|--|
| 0x4000_0000                | 0           | —  |
| 0x4000_1000                | 1           | —  |
| 0x4000_2000                | 2           | —  |
| 0x4000_3000                | 3           | —  |
| 0x4000_4000                | 4           | —  |
| 0x4000_5000                | 5           | —  |
| 0x4000_6000                | 6           | —  |
| 0x4000_7000                | 7           | —  |
| 0x4000_8000                | 8           | DMA controller                           |
| 0x4000_9000                | 9           | —  |
| 0x4000_A000                | 10          | —  |
| 0x4000_B000                | 11          | —  |
| 0x4000_C000                | 12          | —  |
| 0x4000_D000                | 13          | —  |
| 0x4000_E000                | 14          | —  |
| 0x4000_F000                | 15          | GPIO controller (aliased to 0x400F_F000) |
| 0x4001_0000                | 16          | —  |
| 0x4001_1000                | 17          | —  |
| 0x4001_2000                | 18          | —  |
| 0x4001_3000                | 19          | —  |
| 0x4001_4000                | 20          | —  |
| 0x4001_5000                | 21          | —  |
| 0x4001_6000                | 22          | —  |
| 0x4001_7000                | 23          | —  |
| 0x4001_8000                | 24          | —  |
| 0x4001_9000                | 25          | —  |
| 0x4001_A000                | 26          | —  |
| 0x4001_B000                | 27          | —  |
| 0x4001_C000                | 28          | —  |
| 0x4001_D000                | 29          | —  |
| 0x4001_E000                | 30          | —  |
| 0x4001_F000                | 31          | —  |
| 0x4002_0000                | 32          | Flash memory                             |
| 0x4002_1000                | 33          | DMA channel multiplexer 0                |
| 0x4002_2000                | 34          | —  |

*Table continues on the next page...*

**Table 4-2. Peripheral bridge 0 slot assignments (continued)**

| System 32-bit base address | Slot number | Module                              |
|----------------------------|-------------|-------------------------------------|
| 0x4002_3000                | 35          | —                                   |
| 0x4002_4000                | 36          | —                                   |
| 0x4002_5000                | 37          | —                                   |
| 0x4002_6000                | 38          | —                                   |
| 0x4002_7000                | 39          | —                                   |
| 0x4002_8000                | 40          | —                                   |
| 0x4002_9000                | 41          | —                                   |
| 0x4002_A000                | 42          | —                                   |
| 0x4002_B000                | 43          | —                                   |
| 0x4002_C000                | 44          | —                                   |
| 0x4002_D000                | 45          | —                                   |
| 0x4002_E000                | 46          | —                                   |
| 0x4002_F000                | 47          | —                                   |
| 0x4003_0000                | 48          | —                                   |
| 0x4003_1000                | 49          | —                                   |
| 0x4003_2000                | 50          | —                                   |
| 0x4003_3000                | 51          | —                                   |
| 0x4003_4000                | 52          | —                                   |
| 0x4003_5000                | 53          | —                                   |
| 0x4003_6000                | 54          | —                                   |
| 0x4003_7000                | 55          | Periodic interrupt timers (PIT)     |
| 0x4003_8000                | 56          | Timer'/PWM (TPM) 0                  |
| 0x4003_9000                | 57          | Timer'/PWM (TPM) 1                  |
| 0x4003_A000                | 58          | Timer'/PWM (TPM) 2                  |
| 0x4003_B000                | 59          | Analog-to-digital converter (ADC) 0 |
| 0x4003_C000                | 60          | —                                   |
| 0x4003_D000                | 61          | Real-time clock (RTC)               |
| 0x4003_E000                | 62          | —                                   |
| 0x4003_F000                | 63          | DAC0                                |
| 0x4004_0000                | 64          | Low-power timer (LPTMR)             |
| 0x4004_1000                | 65          | —                                   |
| 0x4004_2000                | 66          | —                                   |
| 0x4004_3000                | 67          | —                                   |
| 0x4004_4000                | 68          | —                                   |
| 0x4004_5000                | 69          | Touch sense interface (TSI)         |
| 0x4004_6000                | 70          | —                                   |
| 0x4004_7000                | 71          | SIM low-power logic                 |
| 0x4004_8000                | 72          | System integration module (SIM)     |
| 0x4004_9000                | 73          | Port A multiplexing control         |

*Table continues on the next page...*

**Table 4-2. Peripheral bridge 0 slot assignments (continued)**

| System 32-bit base address | Slot number | Module                              |
|----------------------------|-------------|-------------------------------------|
| 0x4004_A000                | 74          | Port B multiplexing control         |
| 0x4004_B000                | 75          | Port C multiplexing control         |
| 0x4004_C000                | 76          | Port D multiplexing control         |
| 0x4004_D000                | 77          | Port E multiplexing control         |
| 0x4004_E000                | 78          | —                                   |
| 0x4004_F000                | 79          | —                                   |
| 0x4005_0000                | 80          | —                                   |
| 0x4005_1000                | 81          | —                                   |
| 0x4005_2000                | 82          | —                                   |
| 0x4005_3000                | 83          | —                                   |
| 0x4005_4000                | 84          | —                                   |
| 0x4005_5000                | 85          | —                                   |
| 0x4005_6000                | 86          | —                                   |
| 0x4005_7000                | 87          | —                                   |
| 0x4005_8000                | 88          | —                                   |
| 0x4005_9000                | 89          | —                                   |
| 0x4005_A000                | 90          | —                                   |
| 0x4005_B000                | 91          | —                                   |
| 0x4005_C000                | 92          | —                                   |
| 0x4005_D000                | 93          | —                                   |
| 0x4005_E000                | 94          | —                                   |
| 0x4005_F000                | 95          | —                                   |
| 0x4006_0000                | 96          | —                                   |
| 0x4006_1000                | 97          | —                                   |
| 0x4006_2000                | 98          | —                                   |
| 0x4006_3000                | 99          | —                                   |
| 0x4006_4000                | 100         | Multi-purpose Clock Generator (MCG) |
| 0x4006_5000                | 101         | System oscillator (OSC)             |
| 0x4006_6000                | 102         | I <sup>2</sup> C 0                  |
| 0x4006_7000                | 103         | I <sup>2</sup> C 1                  |
| 0x4006_8000                | 104         | —                                   |
| 0x4006_9000                | 105         | —                                   |
| 0x4006_A000                | 106         | UART 0                              |
| 0x4006_B000                | 107         | UART 1                              |
| 0x4006_C000                | 108         | UART 2                              |
| 0x4006_D000                | 109         | —                                   |
| 0x4006_E000                | 110         | —                                   |
| 0x4006_F000                | 111         | —                                   |
| 0x4007_0000                | 112         | —                                   |

Table continues on the next page...

**Table 4-2. Peripheral bridge 0 slot assignments (continued)**

| System 32-bit base address | Slot number | Module  |
|----------------------------|-------------|---|
| 0x4007_1000                | 113         | —   |
| 0x4007_2000                | 114         | USB OTG FS/LS   |
| 0x4007_3000                | 115         | Analog comparator (CMP) / 6-bit digital-to-analog converter (DAC) |
| 0x4007_4000                | 116         | —   |
| 0x4007_5000                | 117         | —   |
| 0x4007_6000                | 118         | SPI 0   |
| 0x4007_7000                | 119         | SPI 1   |
| 0x4007_8000                | 120         | —   |
| 0x4007_9000                | 121         | —   |
| 0x4007_A000                | 122         | —   |
| 0x4007_B000                | 123         | —   |
| 0x4007_C000                | 124         | Low-leakage wakeup unit (LLWU)                                    |
| 0x4007_D000                | 125         | Power management controller (PMC)                                 |
| 0x4007_E000                | 126         | System Mode controller (SMC)                                      |
| 0x4007_F000                | 127         | Reset Control Module (RCM)  |
| 0x400F_F000                | 128         | GPIO controller   |

### 4.6.3 Modules Restricted Access in User Mode

In user mode, for MCG, RCM, SIM (slot 71 and 72), SMC, LLWU, and PMC, reads are allowed, but writes are blocked and generate bus error.

## 4.7 Private Peripheral Bus (PPB) memory map

The PPB is part of the defined ARM bus architecture and provides access to select processor-local modules. These resources are only accessible from the core; other system masters do not have access to them.

**Table 4-3. PPB memory map**

| System 32-bit Address Range | Resource | Additional Range Detail | Resource |
|-----------------------------|----------|-------------------------|----------|
| 0xE000_0000–0xE000_DFFF     | Reserved |                         |          |

*Table continues on the next page...*

**Table 4-3. PPB memory map (continued)**

| System 32-bit Address Range | Resource                   | Additional Range Detail | Resource             |
|-----------------------------|----------------------------|-------------------------|----------------------|
| 0xE000_E000–0xE000_EFFF     | System Control Space (SCS) | 0xE000_E000–0xE000_E00F | Reserved             |
|                             |                            | 0xE000_E010–0xE000_E0FF | SysTick              |
|                             |                            | 0xE000_E100–0xE000_ECFF | NVIC                 |
|                             |                            | 0xE000_ED00–0xE000_ED8F | System Control Block |
|                             |                            | 0xE000_ED90–0xE000_EDEF | Reserved             |
|                             |                            | 0xE000_EDF0–0xE000_EEFF | Debug                |
|                             |                            | 0xE000_EF00–0xE000_EFFF | Reserved             |
| 0xE000_F000–0xE00F_EFFF     | Reserved                   |                         |                      |
| 0xE00F_F000–0xE00F_FFFF     | Core ROM Space (CRS)       |                         |                      |



# Chapter 5

## Clock Distribution

### 5.1 Introduction

This chapter presents the clock architecture for the device, the overview of the clocks and includes a terminology section.

The Cortex M0+ resides within a synchronous core platform, where the processor and bus masters, Flash and peripherals clocks can be configured independently. The clock distribution figure shows how clocks from the MCG and XOSC modules are distributed to the microcontroller's other function units. Some modules in the microcontroller have selectable clock input.

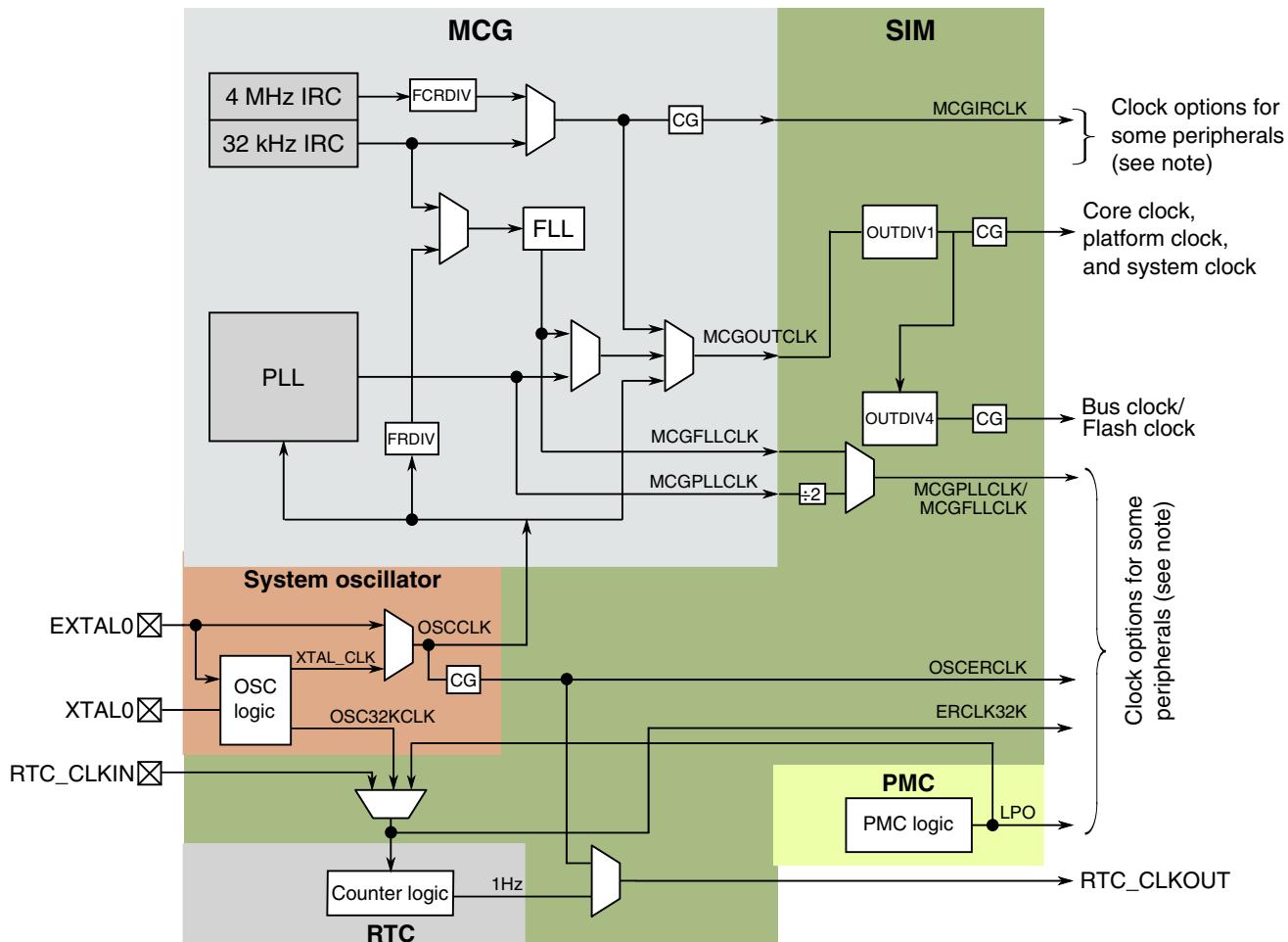
### 5.2 Programming model

The selection and multiplexing of system clock sources is controlled and programmed via the MCG module. The setting of clock dividers and module clock gating for the system are programmed via the SIM module. Reference those sections for detailed register and bit descriptions.

### 5.3 High-Level device clocking diagram

The following [system oscillator](#), [MCG](#), and [SIM](#) module registers control the multiplexers, dividers, and clock gates shown in the below figure:

|              | <b>OSC</b> | <b>MCG</b> | <b>SIM</b>           |
|--------------|------------|------------|----------------------|
| Multiplexers | MCG_Cx     | MCG_Cx     | SIM_SOPT1, SIM_SOPT2 |
| Dividers     | —          | MCG_Cx     | SIM_CLKDIVx          |
| Clock gates  | OSC_CR     | MCG_C1     | SIM_SCGCx            |



CG — Clock gate

**Note:** See subsequent sections for details on where these clocks are used.

| Clock name  | Description   |
|-------------|---|
| Flash clock | Flash memory clock. On this device it is the same as Bus clock.   |
| MCGIRCLK    | MCG output of the slow or fast internal reference clock   |
| MCGOUTCLK   | MCG output of either IRC, MCGFLLCLK, MCGPLLCLK, or MCG's external reference clock that sources the core, system, bus, and flash clock.  |
| MCGFLLCLK   | MCG output of the FLL. MCGFLLCLK or MCGPLLCLK may clock some modules. In addition, this clock is used for USBFS, UART0 and TPM modules. |
| MCGPLLCLK   | MCG output of the PLL. MCGFLLCLK or MCGPLLCLK may clock some modules. In addition, this clock is used for USBFS, UART0 and TPM modules. |
| OSCCLK      | System oscillator output of the internal oscillator or sourced directly from EXTAL. Used as MCG external reference clock.               |
| OSCERCLK    | System oscillator output sourced from OSCCLK that may clock some on-chip modules  |
| OSC32KCLK   | System oscillator 32kHz output  |
| ERCLK32K    | Clock source for some modules that is chosen as OSC32KCLK or RTC_CLKIN  |
| LPO         | PMC 1kHz output   |

## 5.4.1 Device clock summary

The following table provides more information regarding the on-chip clocks.

**Table 5-1. Clock Summary**

| Clock name | Run mode<br>clock frequency | VLPR mode<br>clock frequency | Clock source            | Clock is disabled<br>when...  |
|------------|-----------------------------|------------------------------|-------------------------|---|
| MCGOUTCLK  | Up to 100 MHz               | Up to 4 MHz                  | MCG                     | In all stop modes except for partial stop modes and during PLL locking when MCGOUTCLK derived from PLL. |
| MCGFLLCLK  | Up to 48 MHz                | N/A                          | MCG                     | MCG clock controls do not enable and in all stop modes  |
| MCGPLLCLK  | Up to 100 MHz               | N/A                          | MCG                     | MCG clock controls do not enable, in Stop mode but PLLSTEN=0, or in VLPS, LLS and VLLSx modes           |
| Core clock | Up to 48 MHz                | Up to 4 MHz                  | MCGOUTCLK clock divider | In all wait and stop modes  |

*Table continues on the next page...*

**Table 5-1. Clock Summary (continued)**

| Clock name                          | Run mode<br>clock frequency                                     | VLPR mode<br>clock frequency   | Clock source                                  | Clock is disabled<br>when...  |
|-------------------------------------|---|--|---|---|
| Platform clock                      | Up to 48 MHz  | Up to 4 MHz  | MCGOUTCLK clock divider                       | In all stop modes   |
| System clock                        | Up to 48 MHz  | Up to 4 MHz  | MCGOUTCLK clock divider                       | In all stop modes and Compute Operation   |
| Bus clock                           | Up to 24 MHz  | Up to 1 MHz <sup>1</sup>   | MCGOUTCLK clock divider                       | In all stop modes except for partial STOP2 mode, and Compute Operation  |
| SWD Clock                           | Up to 24MHz   | Up to 1MHz   | SWD_CLK pin                                   | In all stop modes   |
| Flash clock                         | Up to 24 MHz  | Up to 1 MHz in BLPE<br>Up to 800 kHz in BLPI   | MCGOUTCLK clock divider                       | In all stop modes except for partial STOP2 mode   |
| Internal reference (MCGIRCLK)       | 30-40 kHz Slow IRC or 4 MHz Fast IRC                            | 4 MHz Fast IRC only  | MCG   | MCG_C1[IRCLKEN] cleared,<br>Stop/VLPS mode and MCG_C1[IREFSTEN] cleared, or<br>LLS/VLLS mode  |
| External reference (OSCERCLK)       | Up to 48 MHz (bypass),<br>30-40 kHz<br>or<br>3-32 MHz (crystal) | Up to 16 MHz (bypass),<br>30-40 kHz (low-range crystal)<br>or<br>3-16 MHz (high-range crystal) | System OSC                                    | System OSC's OSC_CR[ERCLKEN] cleared, or<br>Stop mode and OSC_CR[EREFSTEN] cleared<br>or VLLS0 and oscillator not in external clock mode. |
| External reference 32kHz (ERCLK32K) | 30-40 kHz   | 30-40 kHz  | System OSC , or RTC_CLKIN                     | System OSC's OSC_CR[ERCLKEN] cleared<br>and RTC's RTC_CR[OSCE] cleared<br>or VLLS0 and oscillator not in external clock mode.             |
| RTC_CLKOUT                          | RTC 1Hz,<br>OSCERCLK  | RTC 1Hz,<br>OSCERCLK   | RTC 1Hz,<br>OSCERCLK                          | Clock is disabled in LLS and VLLSx modes  |
| LPO                                 | 1 kHz   | 1 kHz  | PMC   | in VLLS0  |
| TPM clock                           | Up to 48 MHz  | Up to 4 MHz  | MCGIRCLK, MCGPLLCLK/2, MCGFLLCLK, or OSCERCLK | SIM_SOPT2[TPMSRC ]=0 or selected clock source disabled.   |

Table continues on the next page...

**Table 5-1. Clock Summary (continued)**

| Clock name   | Run mode<br>clock frequency | VLPR mode<br>clock frequency | Clock source   | Clock is disabled<br>when...                                     |
|--------------|-----------------------------|------------------------------|--|--|
| UART0 clock  | Up to 48 MHz                | Up to 4 MHz                  | MCGIRCLK,<br>MCGPLLCLK/2,<br>MCGFLLCLK, or<br>OSCERCLK | SIM_SOPT2[UART0SR<br>C]=00 or selected clock<br>source disabled. |
| USB FS clock | 48 MHz                      | N/A                          | MCGPLLCLK/2 or<br>MCGFLLCLK, or<br>USB_CLKIN           | USB FS OTG is<br>disabled  |

1. If in BLPI mode, where clocking is derived from the fast internal reference clock, the Bus clock and flash clock frequency needs to be limited to 800 kHz if executing from flash.

## 5.5 Internal clocking requirements

The clock dividers are programmed via the SIM module's CLKDIV registers. The following requirements must be met when configuring the clocks for this device:

1. The core, platform, and system clock are programmable from a divide-by-1 through divide-by-16 setting. The core, platform, and system clock frequencies must be 48 MHz or slower.
2. The bus clock and flash clock frequency is divided from the system clock and is programmable from a divide-by-1 through divide-by-8 setting. The bus clock and flash clock must be programmed to 24 MHz or slower.

The following is a common clock configuration for this device:

| Clock          | Frequency |
|----------------|-----------|
| Core clock     | 48 MHz    |
| Platform clock | 48 MHz    |
| System clock   | 48 MHz    |
| Bus clock      | 24 MHz    |
| Flash clock    | 24 MHz    |

### 5.5.1 Clock divider values after reset

Each clock divider is programmed via the SIM module's CLKDIV1 registers. Two bits in the flash memory's FTFA\_FOPT register controls the reset value of the core clock, system clock, bus clock, and flash clock dividers as shown below:

| <b>FTFA_FOPT [4,0]</b> | <b>Core/system clock</b> | <b>Bus/Flash clock</b> | <b>Description</b> |
|------------------------|--------------------------|------------------------|--------------------|
| 00                     | 0x7 (divide by 8)        | 0x1 (divide by 2)      | Low power boot     |
| 01                     | 0x3 (divide by 4)        | 0x1 (divide by 2)      | Low power boot     |
| 10                     | 0x1 (divide by 2)        | 0x1 (divide by 2)      | Low power boot     |
| 11                     | 0x0 (divide by 1)        | 0x1 (divide by 2)      | Fast clock boot    |

This gives the user flexibility in selecting between a lower frequency, low-power boot option vs. higher frequency, higher power during and after reset.

The flash erased state defaults to fast clocking mode, since these bits reside in flash, which is logic 1 in the flash erased state. To enable a lower power boot option, program the appropriate bits in FTFA\_FOPT. During the reset sequence, if either of the control bits is cleared, the system is in a slower clock configuration. Upon any system reset, the clock dividers return to this configurable reset state.

### 5.5.2 VLPR mode clocking

The clock dividers cannot be changed while in VLPR mode. They must be programmed prior to entering VLPR mode to guarantee operation. Max frequency limitations for VLPR mode is as follows :

- the core/system clocks are less than or equal to 4 MHz, and
- the bus and flash clocks are less than or equal to 1 MHz

#### NOTE

When the MCG is in BLPI and clocking is derived from the

Fast IRC, the clock divider flashk d(e M\_SC[FCRDIV],TE) Tj 0 -16 TSIM\_CLKDIV1

## 5.6 Clock Gating

The clock to each module can be individually gated on and off using the SIM module's SCGC<sub>x</sub> registers. These bits are cleared after any reset, which disables the clock to the corresponding module to conserve power. Prior to initializing a module, set the corresponding bit in SCGC<sub>x</sub> register to enable the clock. Before turning off the clock, make sure to disable the module.

Any bus access to a peripheral that has its clock disabled generates an error termination.

## 5.7 Module clocks

The following table summarizes the clocks associated with each module.

**Table 5-2. Module clocks**

| Module                              | Bus interface clock | Internal clocks   | I/O interface clocks |
|-------------------------------------|---------------------|---|----------------------|
| <b>Core modules</b>                 |                     |   |                      |
| ARM Cortex-M0+ core                 | Platform clock      | Core clock  | —                    |
| NVIC                                | Platform clock      | —   | —                    |
| DAP                                 | Platform clock      | —   | SWD_CLK              |
| <b>System modules</b>               |                     |   |                      |
| DMA                                 | System clock        | —   | —                    |
| DMA Mux                             | Bus clock           | —   | —                    |
| Port control                        | Bus clock           | —   | —                    |
| Crossbar Switch                     | Platform clock      | —   | —                    |
| Peripheral bridges                  | System clock        | Bus clock   | —                    |
| LLWU, PMC, SIM, RCM                 | Bus clock           | LPO   | —                    |
| Mode controller                     | Bus clock           | —   | —                    |
| MCM                                 | Platform clock      | —   | —                    |
| Watchdog timer                      | Bus clock           | LPO   | —                    |
| <b>Clocks</b>                       |                     |   |                      |
| MCG                                 | Bus clock           | MCGOUTCLK, MCGPLLCLK,<br>MCGFLLCLK, MCGIRCLK,<br>OSCERCLK | —                    |
| OSC                                 | Bus clock           | OSCERCLK  | —                    |
| <b>Memory and memory interfaces</b> |                     |   |                      |
| Flash Controller                    | Platform clock      | Flash clock   | —                    |
| Flash memory                        | Flash clock         | —   | —                    |
| <b>Analog</b>                       |                     |   |                      |

*Table continues on the next page...*

**Table 5-2. Module clocks (continued)**

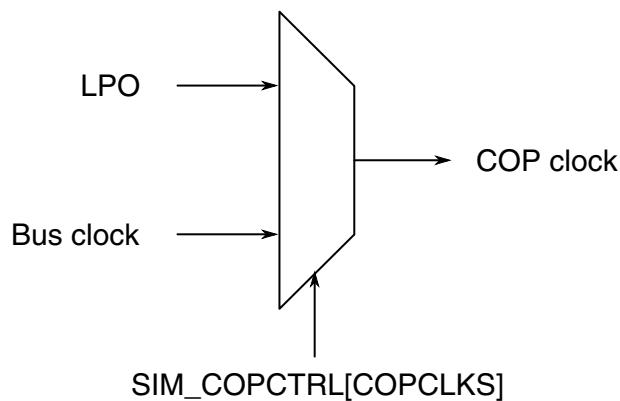
| Module                          | Bus interface clock | Internal clocks                      | I/O interface clocks   |
|---------------------------------|---------------------|--------------------------------------|------------------------|
| ADC                             | Bus clock           | OSCERCLK                             | —                      |
| CMP                             | Bus clock           | —                                    | —                      |
| DAC                             | Bus clock           | —                                    | —                      |
| <b>Timers</b>                   |                     |                                      |                        |
| TPM                             | Bus clock           | TPM clock                            | TPM_CLKIN0, TPM_CLKIN1 |
| PIT                             | Bus clock           | —                                    | —                      |
| LPTMR                           | Bus clock           | LPO, OSCERCLK,<br>MCGIRCLK, ERCLK32K | —                      |
| RTC                             | Bus clock           | ERCLK32K                             | RTC_CLKOUT             |
| <b>Communication interfaces</b> |                     |                                      |                        |
| USB FS OTG                      | System clock        | USB FS clock                         | —                      |
| SPI0                            | Bus clock           | —                                    | SPI0_SCK               |
| SPI1                            | System clock        | —                                    | SPI1_SCK               |
| I <sup>2</sup> C0               | Bus clock           | —                                    | I <sup>2</sup> C0_SCL  |
| I <sup>2</sup> C1               | Bus clock           | —                                    | I <sup>2</sup> C1_SCL  |
| UART0                           | Bus clock           | UART0 clock                          | —                      |
| UART1 , UART2                   | Bus clock           | —                                    | —                      |
| <b>Human-machine interfaces</b> |                     |                                      |                        |
| GPIO                            | Platform clock      | —                                    | —                      |
| TSI                             | Bus clock           | —                                    | —                      |

## 5.7.1 PMC 1-kHz LPO clock

The Power Management Controller (PMC) generates a 1-kHz clock that is enabled in all modes of operation, including all low power modes except VLLS0. This 1-kHz source is commonly referred to as LPO clock or 1-kHz LPO clock.

## 5.7.2 COP clocking

The COP may be clocked from two clock sources as shown in the following figure.

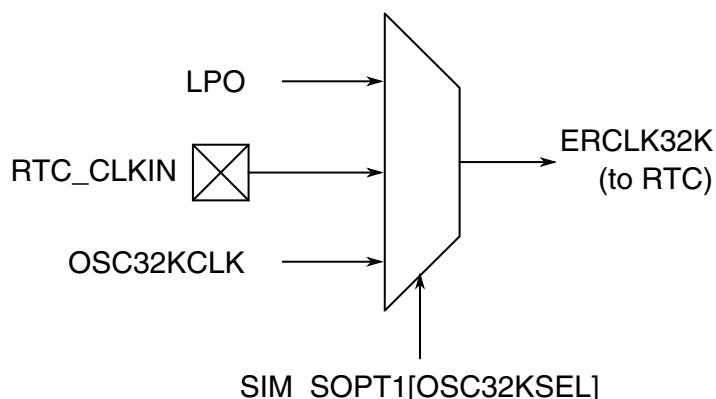
**Figure 5-2. COP clock generation**

### 5.7.3 RTC clocking

The RTC module can be clocked as shown in the following figure.

**NOTE**

The chosen clock must remain enabled if the RTC is to continue operating in all required low-power modes.

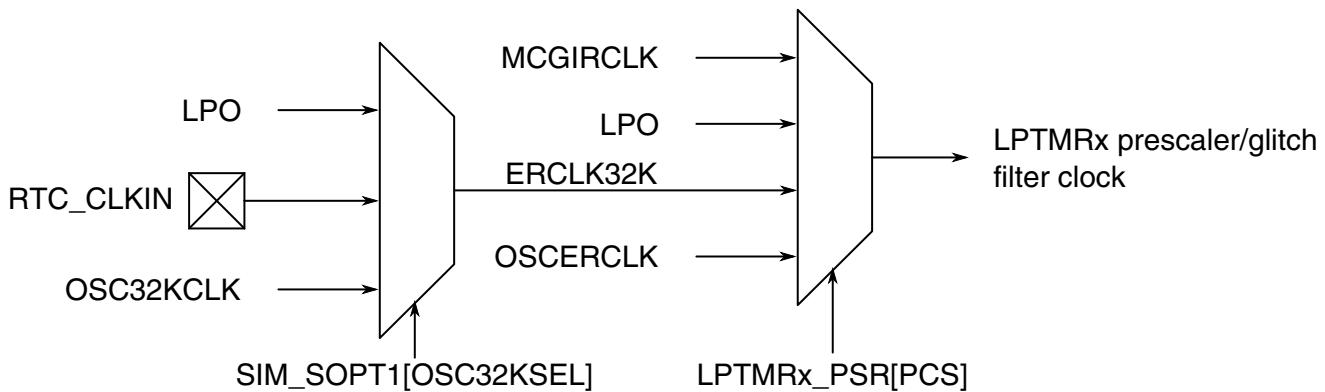
**Figure 5-3. RTC clock generation**

### 5.7.4 LPTMR clocking

The prescaler and glitch filters in each of the LPTMR<sub>x</sub> modules can be clocked as shown in the following figure.

**NOTE**

The chosen clock must remain enabled if the LPTMR<sub>x</sub> is to continue operating in all required low-power modes.



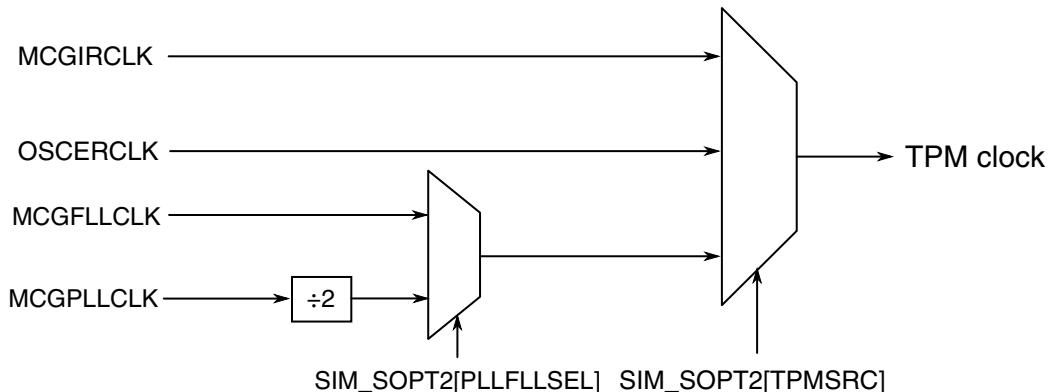
**Figure 5-4. LPTMRx prescaler/glitch filter clock generation**

## 5.7.5 TPM clocking

The counter for the TPM modules have a selectable clock as shown in the following figure.

### NOTE

The chosen clock must remain enabled if the TPM $x$  is to continue operating in all required low-power modes.



**Figure 5-5. TPM clock generation**

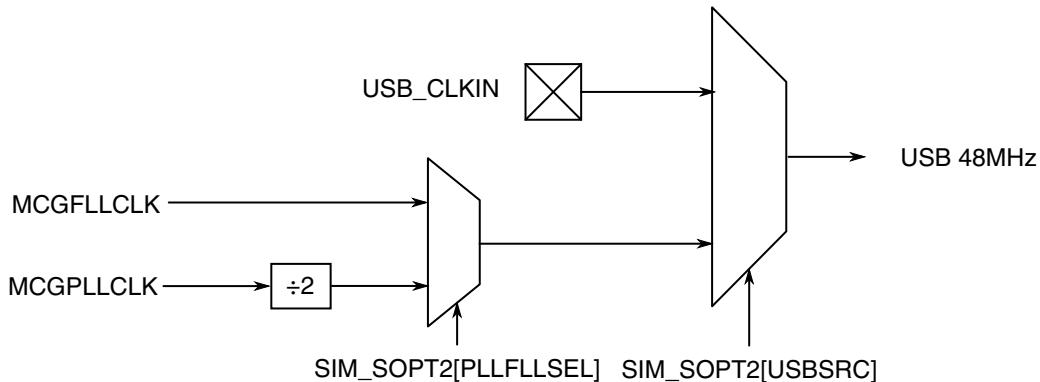
## 5.7.6 USB FS OTG Controller clocking

The USB FS OTG controller is a bus master attached to the crossbar switch. As such, its clock is connected to the system clock.

**NOTE**

For the USB FS OTG controller to operate, the minimum system clock frequency is 20 MHz.

The USB OTG controller also requires a 48 MHz clock. The clock source options are shown below.



**Figure 5-6. USB 48 MHz clock source**

**NOTE**

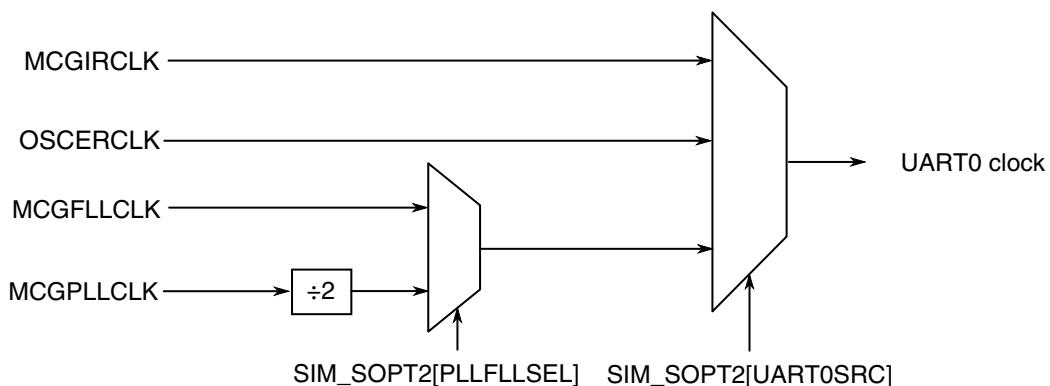
The MCGFLLCLK does not meet the USB jitter specifications for certification.

### 5.7.7 UART clocking

The UART0 module has a selectable clock as shown in the following figure. UART1 and UART2 modules operate from the bus clock.

**NOTE**

The chosen clock must remain enabled if the UART0 is to continue operating in all required low-power modes.



**Figure 5-7. UART0 clock generation**



# Chapter 6

## Reset and Boot

### 6.1 Introduction

The following reset sources are supported in this MCU:

**Table 6-1. Reset sources**

| Reset sources | Description  |
|---------------|--|
| POR reset     | <ul style="list-style-type: none"><li>• Power-on reset (POR)</li></ul>   |
| System resets | <ul style="list-style-type: none"><li>• External pin reset (PIN)</li><li>• Low-voltage detect (LVD)</li><li>• Computer operating properly (COP) watchdog reset</li><li>• Low leakage wakeup (LLWU) reset</li><li>• Multipurpose clock generator loss of clock (LOC) reset</li><li>• Multipurpose clock generator loss of lock (LOL) reset</li><li>• Stop mode acknowledge error (SACKERR)</li><li>• Software reset (SW)</li><li>• Lockup reset (LOCKUP)</li><li>• MDM DAP system reset</li></ul> |
| Debug reset   | <ul style="list-style-type: none"><li>• Debug reset</li></ul>  |

Each of the system reset sources has an associated bit in the system reset status (SRS) registers. See the [Reset Control Module](#) for register details.

The MCU can exit and reset in functional mode where the CPU is executing code (default) or the CPU is in a debug halted state. There are several boot options that can be configured. See [Boot information](#) for more details.

### 6.2 Reset

This section discusses basic reset mechanisms and sources. Some modules that cause resets can be configured to cause interrupts instead. Consult the individual peripheral chapters for more information.

### 6.2.1 Power-on reset (POR)

When power is initially applied to the MCU or when the supply voltage drops below the power-on reset re-arm voltage level ( $V_{POR}$ ), the POR circuit causes a POR reset condition.

As the supply voltage rises, the LVD circuit holds the MCU in reset until the supply has risen above the LVD low threshold ( $V_{LVDL}$ ). The POR and LVD bits in reset status register are set following a POR.

### 6.2.2 System reset sources

Resetting the MCU provides a way to start processing from a known set of initial conditions. System reset begins with the on-chip regulator in full regulation and system clocking generation from an internal reference. When the processor exits reset, it performs the following:

- Reads the start SP (SP\_main) from vector-table offset 0
- Reads the start PC from vector-table offset 4
- LR is set to 0xFFFF\_FFFF

The on-chip peripheral modules are disabled and the non-analog I/O pins are initially configured as disabled. The pins with analog functions assigned to them default to their analog function after reset.

During and following a reset, the SWD pins have their associated input pins configured as:

- SWD\_CLK in pull-down (PD)
- SWD\_DIO in pull-up (PU)

#### 6.2.2.1 External pin reset ( $\overline{\text{RESET}}$ )

This pin is open drain and has an internal pullup device. Asserting  $\overline{\text{RESET}}$  wakes the device from any mode.

The  $\overline{\text{RESET}}$  pin can be disabled by programming RESET\_PIN\_CFG option bit to 0. When this option selected, there could be a short period of contention during a POR ramp where the device drives the pin out low prior to establishing the setting of this option and releasing the RESET function on the pin.

### 6.2.2.1.1 Reset pin filter

The **RESET** pin filter supports filtering from both the 1 kHz LPO clock and the bus clock. The RPFC[RSTFLTSS], RPFC[RSTFLTSRW], and RPFW[RSTFLTSEL] fields in the reset control (RCM) register set control this functionality; see the RCM chapter. The filters are asynchronously reset by Chip POR. The reset value for each filter assumes the RESET pin is negated.

For all stop modes where LPO clock is still active (Stop, VLPS, LLS, VLLS3, and VLLS1), the only filtering option is the LPO based digital filter. The filtering logic either switches to bypass operation or has continued filtering operation depending on the filtering mode selected. When entering VLLS0, the RESET pin filter is disabled and bypassed.

The LPO filter has a fixed filter value of 3. Due to a synchronizer on the input data, there is also some associated latency (2 cycles). As a result, 5 cycles are required to complete a transition from low to high or high to low.

### 6.2.2.2 Low-voltage detect (LVD)

The chip includes a system for managing low voltage conditions to protect memory contents and control MCU system states during supply voltage variations. The system consists of a power-on reset (POR) circuit and an LVD circuit with a user-selectable trip voltage. The LVD system is always enabled in normal run, wait, or stop mode. The LVD system is disabled when entering VLPx, LLS, or VLLSx modes.

The LVD can be configured to generate a reset upon detection of a low voltage condition by setting the PMC's LVDSC1[LVDRE] bit to 1. The low voltage detection threshold is determined by the PMC's LVDSC1[LVDV] field. After an LVD reset has occurred, the LVD system holds the MCU in reset until the supply voltage has risen above the low voltage detection threshold. The RCM's SRS0[LVD] bit is set following either an LVD reset or POR.

### 6.2.2.3 Computer operating properly (COP) watchdog timer

The computer operating properly (COP) watchdog timer (WDOG) monitors the operation of the system by expecting periodic communication from the software. This communication is generally known as servicing (or refreshing) the COP watchdog. If this periodic refreshing does not occur, the watchdog issues a system reset. The COP reset causes the RCM's SRS0[WDOG] bit to set.

#### 6.2.2.4 Low leakage wakeup (LLWU)

The LLWU module provides the means for a number of external pins and a number of internal peripherals to wake the MCU from low leakage power modes. The LLWU module is functional only in low leakage power modes. In VLLSx modes, all enabled inputs to the LLWU can generate a system reset.

After a system reset, the LLWU retains the flags indicating the input source of the last wakeup until the user clears them.

##### NOTE

Some flags are cleared in the LLWU and some flags are required to be cleared in the peripheral module. Refer to the individual peripheral chapters for more information.

#### 6.2.2.5 Multipurpose clock generator loss-of-clock (LOC)

The MCG module supports an external reference clock.

If the C6[CME] bit in the MCG module is set, the clock monitor is enabled. If the external reference falls below  $f_{loc\_low}$  or  $f_{loc\_high}$ , as controlled by the C2[RANGE] field in the MCG module, the MCU resets. The RCM's SRS0[LOC] bit is set to indicate this reset source.

##### NOTE

To prevent unexpected loss of clock reset events, all clock monitors must be disabled before entering any low power modes, including VLPR and VLPW.

#### 6.2.2.6 MCG loss-of-lock (LOL) reset

The MCG includes a PLL loss-of-lock detector. The detector is enabled when configured for PEE and lock has been achieved. If the MCG\_C8[LOLRE] bit in the MCG module is set and the PLL lock status bit (MCG\_S[LOLS0]) becomes set, the MCU resets. The RCM\_SRS0[LOL] bit is set to indicate this reset source.

##### NOTE

This reset source does not cause a reset if the chip is in any stop mode.

### 6.2.2.7 Stop mode acknowledge error (SACKERR)

This reset is generated if the core attempts to enter stop mode or Compute Operation, but not all modules acknowledge stop mode within 1025 cycles of the 1 kHz LPO clock.

A module might not acknowledge the entry to stop mode if an error condition occurs. The error can be caused by a failure of an external clock input to a module.

### 6.2.2.8 Software reset (SW)

The SYSRESETREQ bit in the NVIC application interrupt and reset control register can be set to force a software reset on the device. (See ARM's NVIC documentation for the full description of the register fields, especially the VECTKEY field requirements.) Setting SYSRESETREQ generates a software reset request. This reset forces a system reset of all major components except for the debug module. A software reset causes the RCM's SRS1[SW] bit to set.

### 6.2.2.9 Lockup reset (LOCKUP)

The LOCKUP gives immediate indication of seriously errant kernel software. This is the result of the core being locked because of an unrecoverable exception following the activation of the processor's built in system state protection hardware.

The LOCKUP condition causes a system reset and also causes the RCM's SRS1[LOCKUP] bit to set.

### 6.2.2.10 MDM-AP system reset request

Set the system reset request bit in the MDM-AP control register to initiate a system reset. This is the primary method for resets via the SWD interface. The system reset is held until this bit is cleared.

Set the core hold reset bit in the MDM-AP control register to hold the core in reset as the rest of the chip comes out of system reset.

## 6.2.3 MCU Resets

A variety of resets are generated by the MCU to reset different modules.

### 6.2.3.1 POR Only

The POR Only reset asserts on the POR reset source only. It resets the PMC and RTC.

The POR Only reset also causes all other reset types to occur.

### 6.2.3.2 Chip POR not VLLS

The Chip POR not VLLS reset asserts on POR and LVD reset sources. It resets parts of the SMC and SIM. It also resets the LPTMR.

The Chip POR not VLLS reset also causes these resets to occur: Chip POR, Chip Reset not VLLS, and Chip Reset (including Early Chip Reset).

### 6.2.3.3 Chip POR

The Chip POR asserts on POR, LVD, and VLLS Wakeup reset sources. It resets the Reset Pin Filter registers and parts of the SIM and MCG.

The Chip POR also causes the Chip Reset (including Early Chip Reset) to occur.

### 6.2.3.4 Chip Reset not VLLS

The Chip Reset not VLLS reset asserts on all reset sources except a VLLS Wakeup that does not occur via the RESET pin. It resets parts of the SMC, LLWU, and other modules that remain powered during VLLS mode.

The Chip Reset not VLLS reset also causes the Chip Reset (including Early Chip Reset) to occur.

### 6.2.3.5 Early Chip Reset

The Early Chip Reset asserts on all reset sources. It resets only the flash memory module. It negates before flash memory initialization begins ("earlier" than when the Chip Reset negates).

### 6.2.3.6 Chip Reset

Chip Reset asserts on all reset sources and only negates after flash initialization has completed and the **RESET** pin has also negated. It resets the remaining modules (the modules not reset by other reset types).

### 6.2.4 Reset Pin

For all reset sources except a VLLS Wakeup that does not occur via the **RESET** pin, the **RESET** pin is driven low by the MCU for at least 128 bus clock cycles and until flash initialization has completed.

After flash initialization has completed, the **RESET** pin is released, and the internal Chip Reset negates after the **RESET** pin is pulled high. Keeping the **RESET** pin asserted externally delays the negation of the internal Chip Reset.

The **RESET** pin can be disabled by programming RESET\_PIN\_CFG option bit to 0. When this option is selected, there could be a short period of contention during a POR ramp where the device drives the pin out low prior to establishing the setting of this option and releasing the RESET function on the pin.

### 6.2.5 Debug resets

The following sections detail the debug resets available on the device.

#### 6.2.5.1 Resetting the Debug subsystem

Use the CDBGRSTREQ bit within the DP CTRL/STAT register to reset the debug modules. However, as explained below, using the CDBGRSTREQ bit does not reset all debug-related registers.

CDBGRSTREQ resets the debug-related registers within the following modules:

- SW-DP
- AHB-AP
- MDM-AP (MDM control and status registers)

CDBGRSTREQ does not reset the debug-related registers within the following modules:

- CM0+ core (core debug registers: DHCSR, DCRSR, DCRDR, DEMCR)
- BPU

- DWT
- NVIC
- Crossbar bus switch<sup>1</sup>
- AHB-AP<sup>1</sup>
- Private peripheral bus<sup>1</sup>

## 6.3 Boot

This section describes the boot sequence, including sources and options.

Some configuration information such as clock trim values stored in factory programmed flash locations is autoloaded.

### 6.3.1 Boot sources

The CM0+ core adds support for a programmable Vector Table Offset Register (VTOR) to relocate the exception vector table. This device supports booting from internal flash and RAM.

This device supports booting from internal flash with the reset vectors located at addresses 0x0 (initial SP\_main), 0x4 (initial PC), and RAM with relocating the exception vector table to RAM.

### 6.3.2 FOPT boot options

The flash option register (FOPT) in flash memory module (FTFA) allows the user to customize the operation of the MCU at boot time. The register contains read-only bits that are loaded from the NVM's option byte in the flash configuration field. The default setting for all values in the FOPT register is logic 1 since it is copied from the option byte residing in flash, which has all bits as logic 1 in the flash erased state. To configure for alternate settings, program the appropriate bits in the NVM option byte. The new settings will take effect on subsequent POR, VLLSx recoveries, and any system reset. For more details on programming the option byte, refer to the flash memory chapter.

---

1. CDBGRSTREQ does not affect AHB resources so that debug resources on the private peripheral bus are available during System Reset.

The MCU uses the FTFA\_FOPT register bits to configure the device at reset as shown in the following table.

**Table 6-2. Flash Option Register (FTFA\_FOPT) Bit Definitions**

| Bit Num | Field         | Value  | Definition  |
|---------|---------------|--|---|
| 7-6     | Reserved      | Reserved for future expansion.   |   |
| 5       | FAST_INIT     | Select initialization speed on POR, VLLSx, and any system reset .  |   |
|         |               | 0  | Slower initialization. The Flash initialization will be slower with the benefit of reduced average current during this time. The duration of the recovery will be controlled by the clock divider selection determined by the LPBOOT setting.   |
|         |               | 1  | Fast Initialization.The Flash has faster recoveries at the expense of higher current during these times.  |
| 3       | RESET_PIN_CFG | Enable/disable control for the RESET pin.  |   |
|         |               | 0  | RESET pin is disabled following a POR and cannot be enabled as RESET function. When this option is selected, there could be a short period of contention during a POR ramp where the device drives the pin out low prior to establishing the setting of this option and releasing the RESET function on the pin.<br><br>This bit is preserved through system resets and low power modes. When RESET pin function is disabled it cannot be used as a source for low power mode wakeup. |
|         |               | 1  | RESET pin is dedicated. The port is configured with pullup enabled, open drain, passive filter enabled.   |
| 2       | NMI_DIS       | Enable/disable control for the NMI function.   |   |
|         |               | 0  | NMI interrupts are always blocked. The associated pin continues to default to NMI pin controls with internal pullup enabled. When NMI pin function is disabled it cannot be used as a source for low power mode wakeup.   |
|         |               | 1  | NMI pin/interrupts reset default to enabled.  |
| 1       | Reserved      | Reserved for future expansion.   |   |
| 4,0     | LPBOOT        | Control the reset value of OUTDIV1 value in SIM_CLKDIV1 register. Larger divide value selections produce lower average power consumption during POR, VLLSx recoveries and reset sequencing and after reset exit. The recovery times are also extended if the FAST_INIT option is not selected. |   |
|         |               | 00   | Core and system clock divider (OUTDIV1) is 0x7 (divide by 8)  |
|         |               | 01   | Core and system clock divider (OUTDIV1) is 0x3 (divide by 4)  |
|         |               | 10   | Core and system clock divider (OUTDIV1) is 0x1 (divide by 2)  |
|         |               | 11   | Core and system clock divider (OUTDIV1) is 0x0 (divide by 1)  |

### 6.3.3 Boot sequence

At power up, the on-chip regulator holds the system in a POR state until the input supply is above the POR threshold. The system continues to be held in this static state until the internally regulated supplies have reached a safe operating voltage as determined by the LVD. The Reset Controller logic then controls a sequence to exit reset.

1. A system reset is held on internal logic, the RESET pin is driven out low, and the MCG is enabled in its default clocking mode.
2. Required clocks are enabled (System Clock, Flash Clock, and any Bus Clocks that do not have clock gate control reset to disabled).
3. The system reset on internal logic continues to be held, but the Flash Controller is released from reset and begins initialization operation while the Reset Control logic continues to drive the RESET pin out low.
4. Early in reset sequencing the NVM option byte is read and stored to FTFA\_FOPT. If the bits associated with LPBOOT are programmed for an alternate clock divider reset value, the system/core clock is switched to a slower clock speed. If the FAST\_INIT bit is programmed clear, the Flash initialization switches to slower clock resulting longer recovery times.
5. When Flash Initialization completes, the RESET pin is released. If RESET continues to be asserted (an indication of a slow rise time on the RESET pin or external drive in low), the system continues to be held in reset. Once the RESET pin is detected high, the Core clock is enabled and the system is released from reset.
6. When the system exits reset, the processor sets up the stack, program counter (PC), and link register (LR). The processor reads the start SP (SP\_main) from vector-table offset 0. The core reads the start PC from vector-table offset 4. LR is set to 0xFFFF\_FFFF. The CPU begins execution at the PC location.

Subsequent system resets follow this same reset flow.

# **Chapter 7**

## **Power Management**

### **7.1 Introduction**

This chapter describes the various chip power modes and functionality of the individual modules in these modes.

### **7.2 Clocking Modes**

This section describes the various clocking modes supported on this device.

#### **7.2.1 Partial Stop**

Partial Stop is a clocking option that can be taken instead of entering STOP mode and is configured in the SMC Stop Control Register (SMC\_STOPCTRL). The Stop mode is only partially entered, which leaves some additional functionality alive at the expense of higher power consumption. Partial Stop can be entered from either Run mode or VLP Run mode.

When configured for PSTOP2, only the core and system clocks are gated and the bus clock remains active. The bus masters and bus slaves clocked by the system clock enter Stop mode, but the bus slaves clocked by bus clock remain in Run (or VLP Run) mode. The clock generators in the MCG and the on-chip regulator in the PMC also remain in Run (or VLP Run) mode. Exit from PSTOP2 can be initiated by a reset, an asynchronous interrupt from a bus master or bus slave clocked by the system clock, or a synchronous interrupt from a bus slave clocked by the bus clock. If configured, a DMA request (using the asynchronous DMA wakeup) can also be used to exit Partial Stop for the duration of a DMA transfer before the device is transitioned back into PSTOP2.

## Clocking Modes

When configured for PSTOP1, both the system clock and bus clock are gated. All bus masters and bus slaves enter Stop mode, but the clock generators in the MCG and the on-chip regulator in the PMC remain in Run (or VLP Run) mode. Exit from PSTOP1 can be initiated by a reset or an asynchronous interrupt from a bus master or bus slave. If configured, an asynchronous DMA request can also be used to exit Partial Stop for the duration of a DMA transfer before the device is transitioned back into PSTOP1.

PSTOP1 is functionally similar to STOP mode, but offers faster wakeup at the expense of higher power consumption. Another benefit is that it keeps all of the MCG clocks enabled, which can be useful for some of the asynchronous peripherals that can remain functional in Stop modes.

### 7.2.2 DMA Wakeup

The DMA can be configured to wakeup the device on a DMA request whenever it is placed in stop mode. The wakeup is configured per DMA channel and is supported in Compute Operation, PSTOP, STOP and VLPS low power modes.

When a DMA wakeup is detected in PSTOP, STOP or VLPS then the device will initiate a normal exit from the low power mode. This can include restoring the on-chip regulator and internal power switches, enabling the clock generators in the MCG, enabling the system and bus clocks (but not the core clock) and negating the stop mode signal to the bus masters and bus slaves. The only difference is that the CPU will remain in the low power mode with the CPU clock disabled.

During Compute Operation, a DMA wakeup will initiate a normal exit from Compute Operation. This includes enabling the clocks and negating the stop mode signal to the bus masters and bus slaves. The core clock always remains enabled during Compute Operation.

Since the DMA wakeup will enable the clocks and negate the stop mode signals to all bus masters and slaves, software needs to ensure that bus masters and slaves that are not involved with the DMA wakeup and transfer remain in a known state. That can be accomplished by disabling the modules before entry into the low power mode or by setting the Doze enable bit in selected modules.

Once the DMA request that initiated the wakeup negates and the DMA completes the current transfer, the device will transition back into the original low power mode. This includes requesting all non-CPU bus masters to enter Stop mode and then requesting bus slaves to enter Stop mode. In STOP and VLPS modes the MCG and PMC would then also enter their appropriate modes.

**NOTE**

If the requested DMA transfer cannot cause the DMA request to negate then the device will remain in a higher power state until the low power mode is fully exited.

An enabled DMA wakeup can cause an aborted entry into the low power mode, if the DMA request asserts during the stop mode entry sequence (or reentry if the request asserts during a DMA wakeup) and can cause the SMC to assert its Stop Abort flag. Once the DMA wakeup completes, entry into the low power mode will restart.

An interrupt that occurs during a DMA wakeup will cause an immediate exit from the low power mode (this is optional for Compute Operation) without impacting the DMA transfer.

A DMA wakeup can be generated by either a synchronous DMA request or an asynchronous DMA request. Not all peripherals can generate an asynchronous DMA request in stop modes, although in general if a peripheral can generate synchronous DMA requests and also supports asynchronous interrupts in stop modes, then it can generate an asynchronous DMA request.

### 7.2.3 Compute Operation

Compute Operation is an execution or compute-only mode of operation that keeps the CPU enabled with full access to the SRAM and Flash read port, but places all other bus masters and bus slaves into their stop mode. Compute Operation can be enabled in either Run mode or VLP Run mode.

**NOTE**

Do not enter any stop mode without first exiting Compute Operation.

Because Compute Operation reuses the stop mode logic (including the staged entry with bus masters disabled before bus slaves), any bus master or bus slave that can remain functional in stop mode also remains functional in Compute Operation, including generation of asynchronous interrupts and DMA requests. When enabling Compute Operation in Run mode, module functionality for bus masters and slaves is the equivalent of STOP mode. When enabling Compute Operation in VLP Run mode, module functionality for bus masters and slaves is the equivalent of VLPS mode. The MCG, PMC, SRAM and Flash read port are not affected by Compute Operation, although the Flash register interface is disabled.

## Clocking Modes

During Compute Operation, the AIPS peripheral space is disabled and attempted accesses generate bus errors. The private peripheral space remains accessible during Compute Operation, including the MCM, NVIC, IOPORT and SysTick. Although access to the GPIO registers via the IOPORT is supported, the GPIO port data input registers do not return valid data since clocks are disabled to the Port Control and Interrupt modules. By writing to the GPIO port data output registers, it is possible to control those GPIO ports that are configured as output pins.

Compute Operation is controlled by the CPO register in the MCM, which is only accessible to the CPU. Setting or clearing the CPOREQ bit in the MCM initiates entry or exit into Compute Operation. Compute Operation can also be configured to exit automatically on detection of an interrupt, which is required in order to service most interrupts. Only the core system interrupts (exceptions, including NMI and SysTick) and any edge sensitive interrupts can be serviced without exiting Compute Operation.

When entering Compute Operation, the CPOACK status bit indicates when entry has completed. When exiting Compute Operation in Run mode, the CPOACK status bit negates immediately. When exiting Compute Operation in VLP Run mode, the exit is delayed to allow the PMC to handle the change in power consumption. This delay means the CPOACK bit is polled to determine when the AIPS peripheral space can be accessed without generating a bus error.

The DMA wakeup is also supported during Compute Operation and causes the CPOACK status bit to clear and the AIPS peripheral space to be accessible for the duration of the DMA wakeup. At the completion of the DMA wakeup, the device transitions back into Compute Operation.

### 7.2.4 Peripheral Doze

Several peripherals support a peripheral Doze mode, where a register bit can be used to disable the peripheral for the duration of a low power mode. The Flash can also be placed in a low power state during Peripheral Doze via a register bit in the SIM.

Peripheral Doze is defined to include all of the modes of operation listed below.

- The CPU is in wait mode.
- The CPU is in stop mode, including the entry sequence and for the duration of a DMA wakeup.
- The CPU is in Compute Operation, including the entry sequence and for the duration of a DMA wakeup.

Peripheral Doze can therefore be used to disable selected bus masters or slaves for the duration of WAIT or VLPW mode. It can also be used to disable selected bus slaves immediately on entry into any stop mode (or Compute Operation), instead of waiting for the bus masters to acknowledge the entry as part of the stop entry sequence. Finally, it can be used to disable selected bus masters or slaves that should remain inactive during a DMA wakeup.

If the Flash is not being accessed during WAIT and PSTOP modes, then the Flash Doze mode can be used to reduce power consumption, at the expense of a slightly longer wakeup when executing code and vectors from Flash. It can also be used to reduce power consumption during Compute Operation when executing code and vectors from SRAM.

## 7.2.5 Clock Gating

To conserve power, the clocks to most modules can be turned off using the SCGCx registers in the SIM module. These bits are cleared after any reset, which disables the clock to the corresponding module. Prior to initializing a module, set the corresponding bit in the SCGCx register to enable the clock. Before turning off the clock, make sure to disable the module. For more details, refer to the clock distribution and SIM chapters.

## 7.3 Power modes

The power management controller (PMC) provides multiple power options to allow the user to optimize power consumption for the level of functionality needed.

Depending on the stop requirements of the user application, a variety of stop modes are available that provide state retention, partial power down or full power down of certain logic and/or memory. I/O states are held in all modes of operation. The following table compares the various power modes available.

For each run mode there is a corresponding wait and stop mode. Wait modes are similar to ARM sleep modes. Stop modes (VLPS, STOP) are similar to ARM sleep deep mode. The very low power run (VLPR) operating mode can drastically reduce runtime power when the maximum bus frequency is not required to handle the application needs.

## Power modes

The three primary modes of operation are run, wait and stop. The WFI instruction invokes both wait and stop modes for the chip. The primary modes are augmented in a number of ways to provide lower power based on application needs.

**Table 7-1. Chip power modes**

| Chip mode                           | Description  | Core mode  | Normal recovery method        |
|-------------------------------------|--|------------|-------------------------------|
| Normal run                          | Allows maximum performance of chip. Default mode out of reset; on-chip voltage regulator is on.  | Run        | —                             |
| Normal Wait - via WFI               | Allows peripherals to function while the core is in sleep mode, reducing power. NVIC remains sensitive to interrupts; peripherals continue to be clocked.  | Sleep      | Interrupt                     |
| Normal Stop - via WFI               | Places chip in static state. Lowest power mode that retains all registers while maintaining LVD protection. NVIC is disabled; AWIC is used to wake up from interrupt; peripheral clocks are stopped.   | Sleep Deep | Interrupt                     |
| VLPR (Very Low Power Run)           | On-chip voltage regulator is in a low power mode that supplies only enough power to run the chip at a reduced frequency. Reduced frequency Flash access mode (1 MHz); LVD off; in BLPI clock mode, the fast internal reference oscillator is available to provide a low power nominal 4MHz source for the core with the nominal bus and flash clock required to be <800kHz; alternatively, BLPE clock mode can be used with an external clock or the crystal oscillator providing the clock source.                            | Run        | —                             |
| VLPW (Very Low Power Wait) -via WFI | Same as VLPR but with the core in sleep mode to further reduce power; NVIC remains sensitive to interrupts (FCLK = ON). On-chip voltage regulator is in a low power mode that supplies only enough power to run the chip at a reduced frequency.   | Sleep      | Interrupt                     |
| VLPS (Very Low Power Stop)-via WFI  | Places chip in static state with LVD operation off. Lowest power mode with ADC and pin interrupts functional. Peripheral clocks are stopped, but OSC, LPTMR, RTC, CMP, TSI can be used. TPM and UART can optionally be enabled if their clock source is enabled. NVIC is disabled (FCLK = OFF); AWIC is used to wake up from interrupt. On-chip voltage regulator is in a low power mode that supplies only enough power to run the chip at a reduced frequency. All SRAM is operating (content retained and I/O states held). | Sleep Deep | Interrupt                     |
| LLS (Low Leakage Stop)              | State retention power mode. Most peripherals are in state retention mode (with clocks stopped), but OSC, LLWU, LPTMR, RTC, CMP, TSI can be used. NVIC is disabled; LLWU is used to wake up.<br><br><b>NOTE:</b> The LLWU interrupt must not be masked by the interrupt controller to avoid a scenario where the system does not fully exit stop mode on an LLS recovery.<br>All SRAM is operating (content retained and I/O states held).  | Sleep Deep | Wakeup Interrupt <sup>1</sup> |
| VLLS3 (Very Low Leakage Stop3)      | Most peripherals are disabled (with clocks stopped), but OSC, LLWU, LPTMR, RTC, CMP, TSI can be used. NVIC is disabled; LLWU is used to wake up.<br><br>SRAM_U and SRAM_L remain powered on (content retained and I/O states held).  | Sleep Deep | Wakeup Reset <sup>2</sup>     |

*Table continues on the next page...*

**Table 7-1. Chip power modes (continued)**

| Chip mode                       | Description   | Core mode  | Normal recovery method    |
|---------------------------------|---|------------|---------------------------|
| VLLS1 (Very Low Leakage Stop1)  | Most peripherals are disabled (with clocks stopped), but OSC, LLWU, LPTMR, RTC, CMP, TSI can be used. NVIC is disabled; LLWU is used to wake up.<br><br>All of SRAM_U and SRAM_L are powered off.   | Sleep Deep | Wakeup Reset <sup>2</sup> |
| VLLS0 (Very Low Leakage Stop 0) | Most peripherals are disabled (with clocks stopped), but LLWU, LPTMR, RTC, TSI can be used. NVIC is disabled; LLWU is used to wake up.<br><br>All of SRAM_U and SRAM_L are powered off.<br><br>LPO disabled, optional POR brown-out detection | Sleep Deep | Wakeup Reset <sup>2</sup> |

1. Resumes normal run mode operation by executing the LLWU interrupt service routine.
2. Follows the reset flow with the LLWU interrupt flag set for the NVIC.

## 7.4 Entering and exiting power modes

The WFI instruction invokes wait and stop modes for the chip. The processor exits the low-power mode via an interrupt. For LLS and VLLS modes, the wakeup sources are limited to LLWU generated wakeups, NMI pin, or RESET pin assertions. When the NMI pin or RESET pin have been disabled through associated FOPT settings, then these pins are ignored as wakeup sources. The wake-up flow from VLLSx is always through reset.

### NOTE

The WFE instruction can have the side effect of entering a low-power mode, but that is not its intended usage. See ARM documentation for more on the WFE instruction.

On VLLS recoveries, the I/O pins continue to be held in a static state after code execution begins, allowing software to reconfigure the system before unlocking the I/O. RAM is retained in VLLS3 only.

## 7.5 Module Operation in Low Power Modes

The following table illustrates the functionality of each module while the chip is in each of the low power modes. The standard behavior is shown with some exceptions for Compute Operation (CPO) and Partial Stop2 (PSTOP2).

## Module Operation in Low Power Modes

(Debug modules are discussed separately; see [Debug in Low Power Modes](#).) Number ratings (such as 4 MHz and 1 Mbps) represent the maximum frequencies or maximum data rates per mode. Also, these terms are used:

- FF = Full functionality. In VLPR and VLPW the system frequency is limited, but if a module does not have a limitation in its functionality, it is still listed as FF.
- Async operation = Fully functional with alternate clock source, provided the selected clock source remains enabled
- static = Module register states and associated memories are retained.
- powered = Memory is powered to retain contents.
- low power = Memory is powered to retain contents in a lower power state
- OFF = Modules are powered off; module is in reset state upon wakeup. For clocks, OFF means disabled.
- wakeup = Modules can serve as a wakeup source for the chip.

**Table 7-2. Module operation in low power modes**

| Modules                 | VLPR                             | VLPW                             | Stop                   | VLPS                             | LLS                            | VLLSx  |
|-------------------------|----------------------------------|----------------------------------|------------------------|----------------------------------|--------------------------------|--|
| <b>Core modules</b>     |                                  |                                  |                        |                                  |                                |  |
| NVIC                    | FF                               | FF                               | static                 | static                           | static                         | OFF  |
| <b>System modules</b>   |                                  |                                  |                        |                                  |                                |  |
| Mode Controller         | FF                               | FF                               | FF                     | FF                               | FF                             | FF   |
| LLWU <sup>1</sup>       | static                           | static                           | static                 | static                           | FF                             | FF <sup>2</sup>  |
| Regulator               | low power                        | low power                        | ON                     | low power                        | low power                      | low power in VLLS3, OFF in VLLS0/1                       |
| LVD                     | disabled                         | disabled                         | ON                     | disabled                         | disabled                       | disabled   |
| Brown-out Detection     | ON                               | ON                               | ON                     | ON                               | ON                             | ON in VLLS1/3, optionally disabled in VLLS0 <sup>3</sup> |
| DMA                     | FF<br>Async operation in CPO     | FF                               | Async operation        | Async operation                  | static                         | OFF  |
| Watchdog                | FF<br>static in CPO              | FF                               | static<br>FF in PSTOP2 | static                           | static                         | OFF  |
| <b>Clocks</b>           |                                  |                                  |                        |                                  |                                |  |
| 1kHz LPO                | ON                               | ON                               | ON                     | ON                               | ON                             | ON in VLLS1/3, OFF in VLLS0                              |
| System oscillator (OSC) | OSCERCLK<br>max of 16MHz crystal | OSCERCLK<br>max of 16MHz crystal | OSCERCLK<br>optional   | OSCERCLK<br>max of 16MHz crystal | limited to low range/low power | limited to low range/low power in VLLS1/3, OFF in VLLS0  |

*Table continues on the next page...*

**Table 7-2. Module operation in low power modes (continued)**

| Modules                             | VLPR  | VLPW   | Stop  | VLPS                       | LLS                      | VLLSx                              |
|-------------------------------------|---|--|---|----------------------------|--------------------------|------------------------------------|
| MCG                                 | 4 MHz IRC   | 4 MHz IRC                                    | static - MCGIRCLK optional; PLL optional                              | static - MCGIRCLK optional | static - no clock output | OFF                                |
| Core clock                          | 4 MHz max   | OFF  | OFF   | OFF                        | OFF                      | OFF                                |
| Platform clock                      | 4 MHz max   | 4 MHz max                                    | OFF   | OFF                        | OFF                      | OFF                                |
| System clock                        | 4 MHz max<br>OFF in CPO   | 4 MHz max                                    | OFF   | OFF                        | OFF                      | OFF                                |
| Bus clock                           | 1 MHz max<br>OFF in CPO   | 1 MHz max                                    | OFF<br>24 MHz max in PSTOP2 from RUN<br>1 MHz max in PSTOP2 from VLPR | OFF                        | OFF                      | OFF                                |
| <b>Memory and memory interfaces</b> |   |  |   |                            |                          |                                    |
| Flash                               | 1 MHz max access - no program<br>No register access in CPO                        | low power                                    | low power   | low power                  | OFF                      | OFF                                |
| SRAM_U and SRAM_L                   | low power   | low power                                    | low power   | low power                  | low power                | low power in VLLS3, OFF in VLLS0/1 |
| <b>Communication interfaces</b>     |   |  |   |                            |                          |                                    |
| USB FS/LS                           | static  | static                                       | static  | static                     | static                   | OFF                                |
| USB Voltage Regulator               | optional  | optional                                     | optional  | optional                   | optional                 | optional                           |
| UART0                               | 1 Mbps<br>Async operation in CPO  | 1 Mbps                                       | Async operation FF in PSTOP2  | Async operation            | static                   | OFF                                |
| UART1 , UART2                       | 62.5 kbps<br>static, wakeup on edge in CPO  | 62.5 kbps                                    | static, wakeup on edge<br>FF in PSTOP2                                | static, wakeup on edge     | static                   | OFF                                |
| SPI0                                | master mode 500 kbps,<br>slave mode 250 kbps<br>static, slave mode receive in CPO | master mode 500 kbps,<br>slave mode 250 kbps | static, slave mode receive<br>FF in PSTOP2                            | static, slave mode receive | static                   | OFF                                |

Table continues on the next page...

## Module Operation in Low Power Modes

**Table 7-2. Module operation in low power modes (continued)**

| Modules                         | VLPR  | VLPW                                     | Stop                                      | VLPS                         | LLS                  | VLLSx                               |
|---------------------------------|---|--|---|------------------------------|----------------------|-------------------------------------|
| SPI1                            | master mode 2 Mbps,<br>slave mode 1 Mbps<br><br>static, slave mode receive in CPO | master mode 2 Mbps,<br>slave mode 1 Mbps | static, slave mode receive                | static, slave mode receive   | static               | OFF                                 |
| I <sup>2</sup> C0               | 50 kbps<br>static, address match wakeup in CPO                                    | 50 kbps                                  | static, address match wakeup FF in PSTOP2 | static, address match wakeup | static               | OFF                                 |
| I <sup>2</sup> C1               | 50 kbps<br>static, address match wakeup in CPO                                    | 50 kbps                                  | static, address match wakeup              | static, address match wakeup | static               | OFF                                 |
| <b>Timers</b>                   |   |  |   |                              |                      |                                     |
| TPM                             | FF<br>Async operation in CPO  | FF                                       | Async operation FF in PSTOP2              | Async operation              | static               | OFF                                 |
| PIT                             | FF<br>static in CPO   | FF                                       | static                                    | static                       | static               | OFF                                 |
| LPTMR                           | FF  | FF                                       | Async operation FF in PSTOP2              | Async operation              | Async operation      | Async operation <sup>4</sup>        |
| RTC                             | FF<br>Async operation in CPO  | FF                                       | Async operation FF in PSTOP2              | Async operation              | Async operation      | Async operation <sup>5</sup>        |
| <b>Analog</b>                   |   |  |   |                              |                      |                                     |
| 16-bit ADC                      | FF<br>ADC internal clock only in CPO  | FF                                       | ADC internal clock only FF in PSTOP2      | ADC internal clock only      | static               | OFF                                 |
| CMP <sup>6</sup>                | FF<br>HS or LS compare in CPO   | FF                                       | HS or LS compare FF in PSTOP2             | HS or LS compare             | LS compare           | LS compare in VLLS1/3, OFF in VLLS0 |
| 6-bit DAC                       | FF<br>static in CPO   | FF                                       | static FF in PSTOP2                       | static                       | static               | static, OFF in VLLS0                |
| 12-bit DAC                      | FF<br>static in CPO   | FF                                       | static FF in PSTOP2                       | static                       | static               | static                              |
| <b>Human-machine interfaces</b> |   |  |   |                              |                      |                                     |
| GPIO                            | FF<br>IOPORT write only in CPO  | FF                                       | static output, wakeup input FF in PSTOP2  | static output, wakeup input  | static, pins latched | OFF, pins latched                   |

*Table continues on the next page...*

**Table 7-2. Module operation in low power modes (continued)**

| Modules | VLPR                         | VLPW                         | Stop   | VLPS                         | LLS                          | VLLSx                        |
|---------|------------------------------|------------------------------|--|------------------------------|------------------------------|------------------------------|
| TSI     | FF<br>Async operation in CPO | Async operation <sup>7</sup> | Async operation <sup>7</sup><br>FF in PSTOP2 | Async operation <sup>7</sup> | Async operation <sup>7</sup> | Async operation <sup>7</sup> |

1. Using the LLWU module, the external pins available for this chip do not require the associated peripheral function to be enabled. It only requires the function controlling the pin (GPIO or peripheral) to be configured as an input to allow a transition to occur to the LLWU.
2. Since LPO clock source is disabled, filters will be bypassed during VLLS0.
3. The STOPCTRL[PORPO] bit in the SMC module controls this option.
4. LPO clock source is not available in VLLS0. Also, to use system OSC in VLLS0 it must be configured for bypass (external clock) operation. Pulse counting is available in all modes.
5. In VLLS0 the only clocking option is from RTC\_CLKIN.
6. CMP in stop or VLPS supports high speed or low speed external pin to pin or external pin to DAC compares. CMP in LLS or VLLSx only supports low speed external pin to pin or external pin to DAC compares. Windowed, sampled & filtered modes of operation are not available while in stop, VLPS, LLS, or VLLSx modes.
7. TSI wakeup from all low power modes is limited to a single selectable pin.



# Chapter 8

## Security

### 8.1 Introduction

This device implements security based on the mode selected from the flash module. The following sections provide an overview of flash security and details the effects of security on non-flash modules.

### 8.2 Flash Security

The flash module provides security information to the MCU based on the state held by the FSEC[SEC] bits. The MCU, in turn, confirms the security request and limits access to flash resources. During reset, the flash module initializes the FSEC register using data read from the security byte of the flash configuration field.

#### NOTE

The security features apply only to external accesses: debug. CPU accesses to the flash are not affected by the status of FSEC.

In the unsecured state all flash commands are available on the programming interfaces either from the debug port (SWD) or user code execution. When the flash is secured (FSEC[SEC] = 00, 01, or 11), the programmer interfaces are only allowed to launch mass erase operations. Additionally, in this mode, the debug port has no access to memory locations.

### 8.3 Security Interactions with other Modules

The flash security settings are used by the system to determine what resources are available. The following sections describe the interactions between modules and the flash security settings or the impact that the flash security has on non-flash modules.

### 8.3.1 Security Interactions with Debug

When flash security is active the SWD port cannot access the memory resources of the MCU.

Although most debug functions are disabled, the debugger can write to the Flash Mass Erase in Progress bit to trigger a mass erase (Erase All Blocks) command. A mass erase via the debugger is allowed even when some memory locations are protected.

When mass erase is disabled, mass erase via the debugger is blocked.

# Chapter 9

## Debug

### 9.1 Introduction

This device's debug is based on the ARM CoreSight™ architecture and is configured to provide the maximum flexibility as allowed by the restrictions of the pinout and other available resources.

It provides register and memory accessibility from the external debugger interface, basic run/halt control plus 2 breakpoints and 2 watchpoints.

Only one debug interface is supported:

- Serial Wire Debug (SWD)

### 9.2 Debug Port Pin Descriptions

The debug port pins default after POR to their SWD functionality.

**Table 9-1. Serial wire debug pin description**

| Pin Name | Type           | Description  |
|----------|----------------|--|
| SWD_CLK  | Input          | Serial Wire Clock. This pin is the clock for debug logic when in the Serial Wire Debug mode. This pin is pulled down internally.                               |
| SWD_DIO  | Input / Output | Serial wire debug data input/output. The SWD_DIO pin is used by an external debug tool for communication and device control. This pin is pulled up internally. |

## 9.3 SWD status and control registers

Through the ARM Debug Access Port (DAP), the debugger has access to the status and control elements, implemented as registers on the DAP bus as shown in the following figure. These registers provide additional control and status for low power mode recovery and typical run-control scenarios. The status register bits also provide a means for the debugger to get updated status of the core without having to initiate a bus transaction across the crossbar switch, thus remaining less intrusive during a debug session.

It is important to note that these DAP control and status registers are not memory mapped within the system memory map and are only accessible via the Debug Access Port using SWD. The MDM-AP is accessible as Debug Access Port 1 with the available registers shown in the table below.

**Table 9-2. MDM-AP Register Summary**

| Address     | Register | Description  |
|-------------|----------|--|
| 0x0100_0000 | Status   | See <a href="#">MDM-AP Status Register</a>                         |
| 0x0100_0004 | Control  | See <a href="#">MDM-AP Control Register</a>                        |
| 0x0100_00FC | IDR      | Read-only identification register that always reads as 0x001C_0020 |

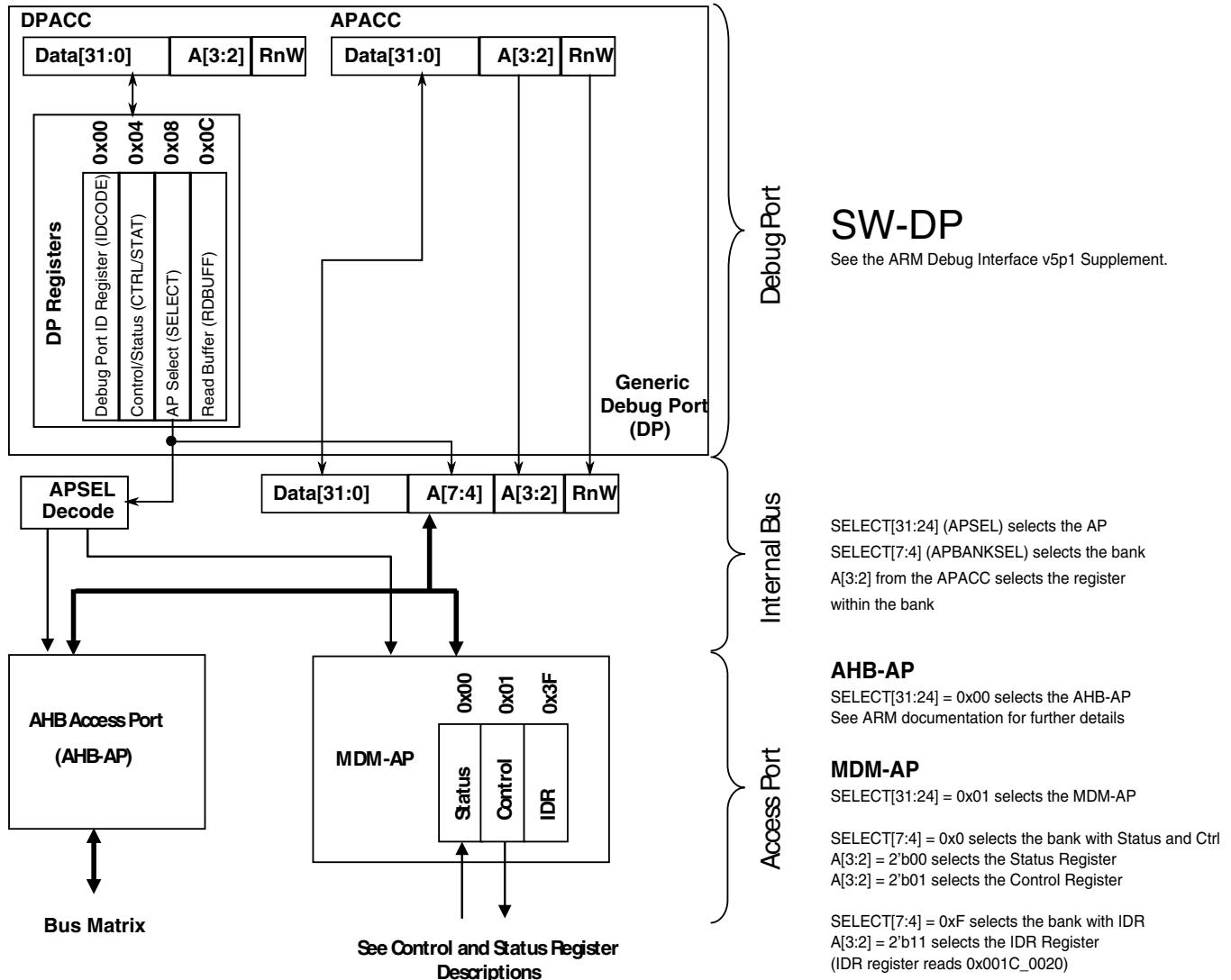


Figure 9-1. MDM AP Addressing

### 9.3.1 MDM-AP Control Register

Table 9-3. MDM-AP Control register assignments

| Bit | Name                         | Secure <sup>1</sup> | Description  |
|-----|------------------------------|---------------------|--|
| 0   | Flash Mass Erase in Progress | Y                   | <p>Set to cause mass erase. Cleared by hardware after mass erase operation completes.</p> <p>When mass erase is disabled (via MEEN and SEC settings), the erase request does not occur and the Flash Mass Erase in Progress bit continues to assert until the next system reset.</p> |
| 1   | Debug Disable                | N                   | Set to disable debug. Clear to allow debug operation. When set it overrides the C_DEBUGEN bit within the DHCSR and force disables Debug logic.   |

Table continues on the next page...

**Table 9-3. MDM-AP Control register assignments (continued)**

| Bit    | Name                                | Secure <sup>1</sup> | Description   |
|--------|-------------------------------------|---------------------|---|
| 2      | Debug Request                       | N                   | <p>Set to force the core to halt.</p> <p>If the core is in a stop or wait mode, this bit can be used to wakeup the core and transition to a halted state.</p>   |
| 3      | System Reset Request                | Y                   | Set to force a system reset. The system remains held in reset until this bit is cleared.  |
| 4      | Core Hold Reset                     | N                   | <p>Configuration bit to control core operation at the end of system reset sequencing.</p> <p>0 Normal operation - release the core from reset along with the rest of the system at the end of system reset sequencing.</p> <p>1 Suspend operation - hold the core in reset at the end of reset sequencing. Once the system enters this suspended state, clearing this control bit immediately releases the core from reset and CPU operation begins.</p>  |
| 5      | VLLSx Debug Request (VLLDBGREQ)     | N                   | <p>Set to configure the system to be held in reset after the next recovery from a VLLSx mode. This bit is ignored on a VLLS wakeup via the Reset pin. During a VLLS wakeup via the Reset pin, the system can be held in reset by holding the reset pin asserted allowing the debugger to re-initialize the debug modules.</p> <p>This bit holds the system in reset when VLLSx modes are exited to allow the debugger time to re-initialize debug IP before the debug session continues.</p> <p>The Mode Controller captures this bit logic on entry to VLLSx modes. Upon exit from VLLSx modes, the Mode Controller will hold the system in reset until VLLDBGACK is asserted.</p> <p>The VLLDBGREQ bit clears automatically due to the POR reset generated as part of the VLLSx recovery.</p> |
| 6      | VLLSx Debug Acknowledge (VLLDBGACK) | N                   | <p>Set to release a system being held in reset following a VLLSx recovery. This bit is used by the debugger to release the system reset when it is being held on VLLSx mode exit. The debugger re-initializes all debug IP and then assert this control bit to allow the Mode Controller to release the system from reset and allow CPU operation to begin.</p> <p>The VLLDBGACK bit is cleared by the debugger or can be left set because it clears automatically due to the POR reset generated as part of the next VLLSx recovery.</p>   |
| 7      | LLS, VLLSx Status Acknowledge       | N                   | <p>Set this bit to acknowledge the DAP LLS and VLLS Status bits have been read. This acknowledge automatically clears the status bits.</p> <p>This bit is used by the debugger to clear the sticky LLS and VLLSx mode entry status bits. This bit is asserted and cleared by the debugger.</p>  |
| 8 – 31 | Reserved for future use             | N                   |   |

1. Command available in secure mode

## 9.3.2 MDM-AP Status Register

**Table 9-4. MDM-AP Status register assignments**

| Bit | Name                         | Description  |
|-----|------------------------------|--|
| 0   | Flash Mass Erase Acknowledge | The Flash Mass Erase Acknowledge bit is cleared after any system reset. The bit is also cleared at launch of a mass erase command due to write of Flash Mass Erase in Progress bit in MDM AP Control Register. The Flash Mass Erase Acknowledge is set after Flash control logic has started the mass erase operation.<br><br>When mass erase is disabled (via MEEN and SEC settings), an erase request due to setting of Flash Mass Erase in Progress bit is not acknowledged.  |
| 1   | Flash Ready                  | Indicate Flash has been initialized and debugger can be configured even if system is continuing to be held in reset via the debugger.  |
| 2   | System Security              | Indicates the security state. When secure, the debugger does not have access to the system bus or any memory mapped peripherals. This bit indicates when the part is locked and no system bus access is possible.  |
| 3   | System Reset                 | Indicates the system reset state.<br><br>0 System is in reset<br><br>1 System is not in reset  |
| 4   | Reserved                     |  |
| 5   | Mass Erase Enable            | Indicates if the MCU can be mass erased or not<br><br>0 Mass erase is disabled<br><br>1 Mass erase is enabled  |
| 6   | Backdoor Access Key Enable   | Indicates if the MCU has the backdoor access key enabled.<br><br>0 Disabled<br><br>1 Enabled   |
| 7   | LP Enabled                   | Decode of SMC_PMCTRL[STOPM] field to indicate that VLPS, LLS, or VLLSx are the selected power mode the next time the ARM Core enters Deep Sleep.<br><br>0 Low Power Stop Mode is not enabled<br><br>1 Low Power Stop Mode is enabled<br><br>Usage intended for debug operation in which Run to VLPS is attempted. Per debug definition, the system actually enters the Stop state. A debugger should interpret deep sleep indication (with SLEEPDEEP and SLEEPING asserted), in conjunction with this bit asserted as the debugger-VLPS status indication. |
| 8   | Very Low Power Mode          | Indicates current power mode is VLPx. This bit is not 'sticky' and should always represent whether VLPx is enabled or not.<br><br>This bit is used to throttle SWD_CLK frequency up/down.  |

*Table continues on the next page...*

**Table 9-4. MDM-AP Status register assignments (continued)**

| Bit     | Name                    | Description   |
|---------|-------------------------|---|
| 9       | LLS Mode Exit           | This bit indicates an exit from LLS mode has occurred. The debugger will lose communication while the system is in LLS (including access to this register). Once communication is reestablished, this bit indicates that the system had been in LLS. Since the debug modules held their state during LLS, they do not need to be reconfigured.<br><br>This bit is set during the LLS recovery sequence. The LLS Mode Exit bit is held until the debugger has had a chance to recognize that LLS was exited and is cleared by a write of 1 to the LLS, VLLSx Status Acknowledge bit in MDM AP Control register.                    |
| 10      | VLLSx Modes Exit        | This bit indicates an exit from VLLSx mode has occurred. The debugger will lose communication while the system is in VLLSx (including access to this register). Once communication is reestablished, this bit indicates that the system had been in VLLSx. Since the debug modules lose their state during VLLSx modes, they need to be reconfigured.<br><br>This bit is set during the VLLSx recovery sequence. The VLLSx Mode Exit bit is held until the debugger has had a chance to recognize that a VLLS mode was exited and is cleared by a write of 1 to the LLS, VLLSx Status Acknowledge bit in MDM AP Control register. |
| 11 – 15 | Reserved for future use | Always read 0.  |
| 16      | Core Halted             | Indicates the Core has entered debug halt mode  |
| 17      | Core SLEEPDEEP          | Indicates the Core has entered a low power mode   |
| 18      | Core SLEEPING           | SLEEPING==1 and SLEEPDEEP==0 indicates wait or VLPW mode.<br>SLEEPING==1 and SLEEPDEEP==1 indicates stop or VLPS mode.  |
| 19 – 31 | Reserved for future use | Always read 0.  |

## 9.4 Debug Resets

The debug system receives the following sources of reset:

- Debug reset (CDBGRSTREQ bit within the DP CTRL/STAT register) that allows the debugger to reset the debug logic.
- System POR reset

Conversely the debug system is capable of generating system reset using the following mechanism:

- A system reset in the DAP control register which allows the debugger to hold the system in reset.
- SYSRESETREQ bit in the NVIC application interrupt and reset control register
- A system reset in the DAP control register which allows the debugger to hold the Core in reset.

## 9.5 Micro Trace Buffer (MTB)

The Micro Trace Buffer (MTB) provides a simple execution trace capability for the Cortex-M0+ processor. When enabled, the MTB records changes in program flow reported by the Cortex-M0+ processor, via the execution trace interface, into a configurable region of the SRAM. Subsequently an off-chip debugger may extract the trace information, which would allow reconstruction of an instruction flow trace. The MTB does not include any form of load/store data trace capability or tracing of any other information.

In addition to providing the trace capability, the MTB also operates as a simple AHB-Lite SRAM controller. The system bus masters, including the processor, have read/write access to all of the SRAM via the AHB-Lite interface, allowing the memory to also be used to store program and data information. The MTB simultaneously stores the trace information into an attached SRAM and allows bus masters to access the memory. The MTB ensures that trace information write accesses to the SRAM take priority over accesses from the AHB-Lite interface.

The MTB includes trace control registers for configuring and triggering the MTB functions. The MTB also supports triggering via TSTART and TSTOP control functions in the MTB DWT module.

## 9.6 Debug in Low Power Modes

In low power modes in which the debug modules are kept static or powered off, the debugger cannot gather any debug data for the duration of the low power mode. In the case that the debugger is held static, the debug port returns to full functionality as soon as the low power mode exits and the system returns to a state with active debug. In the case that the debugger logic is powered off, the debugger is reset on recovery and must be reconfigured once the low power mode is exited.

## 9.7 Debug & Security

When flash security is enabled, the debug port capabilities are limited in order to prevent exploitation of secure data. In the secure state the debugger still has access to the status register and can determine the current security state of the device. In the case of a secure device, the debugger only has the capability of performing a mass erase operation.



# Chapter 10

## Signal Multiplexing and Signal Descriptions

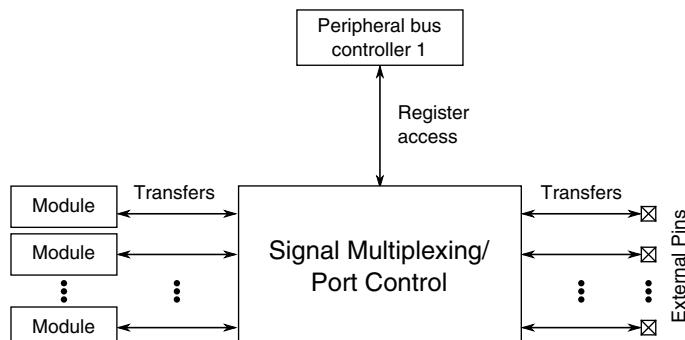
### 10.1 Introduction

To optimize functionality in small packages, pins have several functions available via signal multiplexing. This chapter illustrates which of this device's signals are multiplexed on which external pin.

The [Port Control](#) block controls which signal is present on the external pin. Reference that chapter to find which register controls the operation of a specific pin.

### 10.2 Signal Multiplexing Integration

This section summarizes how the module is integrated into the device. For a comprehensive description of the module itself, see the module's dedicated chapter.



**Figure 10-1. Signal multiplexing integration**

**Table 10-1. Reference links to related information**

| Topic             | Related module | Reference                         |
|-------------------|----------------|-----------------------------------|
| Full description  | Port control   | <a href="#">Port control</a>      |
| System memory map |                | <a href="#">System memory map</a> |

*Table continues on the next page...*

**Table 10-1. Reference links to related information (continued)**

| Topic           | Related module            | Reference                          |
|-----------------|---------------------------|------------------------------------|
| Clocking        |                           | <a href="#">Clock Distribution</a> |
| Register access | Peripheral bus controller | <a href="#">Peripheral bridge</a>  |

## 10.2.1 Port control and interrupt module features

- 32-pin ports

### NOTE

Not all pins are available on the device. See the following section for details.

- Port A and Port D each assigned one interrupt. For DMA requests, Port A and Port D each have a dedicated input to the DMA MUX.

The reset state and read/write characteristics of the bit fields within the PORTx\_PCRn registers is summarized in the table below.

**Table 10-2. Port control register configuration summary**

| This field of PORTx_PCRn | Generally resets to | Except for   | Resets to | Configurability   |
|--------------------------|---------------------|--|-----------|---|
| PS                       | 1                   | PTA0   | 0         | Fixed - All are read only   |
| PE                       | 0                   | PTA0 and PTA2  | 1         | Yes - All GPIO are configurable   |
| DSE                      | 0                   | No exceptions - all DSE are cleared on reset.  | —         | 4 pins are configurable for High Drive (PTB0 , PTB1, PTD6 , PTD7). All others are fixed for Normal Drive and the associated DSE bit is read only. |
| SRE                      | 1                   | PTA3, PTA14, PTA15, PTA16, PTA17, PTB10, PTB11, PTB16, PTB17, PTC3, PTC4, PTC5, PTC6, PTC7, PTD4, PTD5, PTD6, PTD7 | 0         | Fixed - All are read only   |
| MUX                      | 000                 | PTA0, PTA3 and PTA4  | 111       | Yes - All GPIO are configurable   |
| PFE                      | 0                   | No exceptions - all PFE are cleared on reset. <sup>1</sup>   | —         | The GPIO shared with NMI_b pin is configurable. All other GPIO is fixed and read only.  |
| IRQC                     | 000                 | No exceptions - all are cleared on reset.  | —         | Only implemented for ports that support interrupt and DMA functionality.  |

Table continues on the next page...

**Table 10-2. Port control register configuration summary (continued)**

| This field of PORTx_PC Rn | Generally resets to | Except for                                | Resets to | Configurability  |
|---------------------------|---------------------|---|-----------|--|
| ISF                       | 0                   | No exceptions - all are cleared on reset. | —         | Only implemented for ports that support interrupt and DMA functionality. |

1. The **RESET** pin has the passive analog filter fixed enabled when functioning as the **RESET** pin (FOPT[RESET\_PIN\_CFG] = 1) and fixed disabled when configured for other shared functions.

## 10.2.2 Clock gating

The clock to the port control module can be gated on and off using the SCGC5[PORTx] bits in the SIM module. These bits are cleared after any reset, which disables the clock to the corresponding module to conserve power. Prior to initializing the corresponding module, set SCGC5[PORTx] in the SIM module to enable the clock. Before turning off the clock, make sure to disable the module. For more details, refer to the clock distribution chapter.

## 10.2.3 Signal multiplexing constraints

1. A given peripheral function must be assigned to a maximum of one package pin. Do not program the same function to more than one pin.
2. To ensure the best signal timing for a given peripheral's interface, choose the pins in closest proximity to each other.

## 10.3 Pinout

### 10.3.1 KL25 Signal Multiplexing and Pin Assignments

The following table shows the signals available on each pin and the locations of these pins on the devices supported by this document. The Port Control Module is responsible for selecting which ALT functionality is available on each pin.

| 80 LQFP | 64 LQFP | 48 QFN | 32 QFN | Pin Name | Default  | ALT0 | ALT1 | ALT2      | ALT3     | ALT4       | ALT5      | ALT6     | ALT7 |
|---------|---------|--------|--------|----------|----------|------|------|-----------|----------|------------|-----------|----------|------|
| 1       | 1       | —      | 1      | PTE0     | DISABLED |      | PTE0 |           | UART1_TX | RTC_CLKOUT | CMP0_OUT  | I2C1_SDA |      |
| 2       | 2       | —      | —      | PTE1     | DISABLED |      | PTE1 | SPI1_MOSI | UART1_RX |            | SPI1_MISO | I2C1_SCL |      |

## Pinout

| 80<br>LQFP | 64<br>LQFP | 48<br>QFN | 32<br>QFN | Pin Name | Default                             | ALT0                                | ALT1  | ALT2      | ALT3     | ALT4       | ALT5      | ALT6 | ALT7    |
|------------|------------|-----------|-----------|----------|-------------------------------------|-------------------------------------|-------|-----------|----------|------------|-----------|------|---------|
| 3          | —          | —         | —         | PTE2     | DISABLED                            |                                     | PTE2  | SPI1_SCK  |          |            |           |      |         |
| 4          | —          | —         | —         | PTE3     | DISABLED                            |                                     | PTE3  | SPI1_MISO |          |            | SPI1_MOSI |      |         |
| 5          | —          | —         | —         | PTE4     | DISABLED                            |                                     | PTE4  | SPI1_PCS0 |          |            |           |      |         |
| 6          | —          | —         | —         | PTE5     | DISABLED                            |                                     | PTE5  |           |          |            |           |      |         |
| 7          | 3          | 1         | —         | VDD      | VDD                                 | VDD                                 |       |           |          |            |           |      |         |
| 8          | 4          | 2         | 2         | VSS      | VSS                                 | VSS                                 |       |           |          |            |           |      |         |
| 9          | 5          | 3         | 3         | USB0_DP  | USB0_DP                             | USB0_DP                             |       |           |          |            |           |      |         |
| 10         | 6          | 4         | 4         | USB0_DM  | USB0_DM                             | USB0_DM                             |       |           |          |            |           |      |         |
| 11         | 7          | 5         | 5         | VOUT33   | VOUT33                              | VOUT33                              |       |           |          |            |           |      |         |
| 12         | 8          | 6         | 6         | VREGIN   | VREGIN                              | VREGIN                              |       |           |          |            |           |      |         |
| 13         | 9          | 7         | —         | PTE20    | ADC0_DPO/<br>ADC0_SE0               | ADC0_DPO/<br>ADC0_SE0               | PTE20 |           | TPM1_CH0 | UART0_TX   |           |      |         |
| 14         | 10         | 8         | —         | PTE21    | ADC0_DM0/<br>ADC0_SE4a              | ADC0_DM0/<br>ADC0_SE4a              | PTE21 |           | TPM1_CH1 | UART0_RX   |           |      |         |
| 15         | 11         | —         | —         | PTE22    | ADC0_DP3/<br>ADC0_SE3               | ADC0_DP3/<br>ADC0_SE3               | PTE22 |           | TPM2_CH0 | UART2_TX   |           |      |         |
| 16         | 12         | —         | —         | PTE23    | ADC0_DM3/<br>ADC0_SE7a              | ADC0_DM3/<br>ADC0_SE7a              | PTE23 |           | TPM2_CH1 | UART2_RX   |           |      |         |
| 17         | 13         | 9         | 7         | VDDA     | VDDA                                | VDDA                                |       |           |          |            |           |      |         |
| 18         | 14         | 10        | —         | VREFH    | VREFH                               | VREFH                               |       |           |          |            |           |      |         |
| 19         | 15         | 11        | —         | VREFL    | VREFL                               | VREFL                               |       |           |          |            |           |      |         |
| 20         | 16         | 12        | 8         | VSSA     | VSSA                                | VSSA                                |       |           |          |            |           |      |         |
| 21         | 17         | 13        | —         | PTE29    | CMP0_IN5/<br>ADC0_SE4b              | CMP0_IN5/<br>ADC0_SE4b              | PTE29 |           | TPM0_CH2 | TPM_CLKIN0 |           |      |         |
| 22         | 18         | 14        | 9         | PTE30    | DAC0_OUT/<br>ADC0_SE23/<br>CMP0_IN4 | DAC0_OUT/<br>ADC0_SE23/<br>CMP0_IN4 | PTE30 |           | TPM0_CH3 | TPM_CLKIN1 |           |      |         |
| 23         | 19         | —         | —         | PTE31    | DISABLED                            |                                     | PTE31 |           | TPM0_CH4 |            |           |      |         |
| 24         | 20         | 15        | —         | PTE24    | DISABLED                            |                                     | PTE24 |           | TPM0_CH0 |            | I2C0_SCL  |      |         |
| 25         | 21         | 16        | —         | PTE25    | DISABLED                            |                                     | PTE25 |           | TPM0_CH1 |            | I2C0_SDA  |      |         |
| 26         | 22         | 17        | 10        | PTA0     | SWD_CLK                             | TSI0_CH1                            | PTA0  |           | TPM0_CH5 |            |           |      | SWD_CLK |
| 27         | 23         | 18        | 11        | PTA1     | DISABLED                            | TSI0_CH2                            | PTA1  | UART0_RX  | TPM2_CH0 |            |           |      |         |
| 28         | 24         | 19        | 12        | PTA2     | DISABLED                            | TSI0_CH3                            | PTA2  | UART0_TX  | TPM2_CH1 |            |           |      |         |
| 29         | 25         | 20        | 13        | PTA3     | SWD_DIO                             | TSI0_CH4                            | PTA3  | I2C1_SCL  | TPM0_CH0 |            |           |      | SWD_DIO |
| 30         | 26         | 21        | 14        | PTA4     | NMI_b                               | TSI0_CH5                            | PTA4  | I2C1_SDA  | TPM0_CH1 |            |           |      | NMI_b   |
| 31         | 27         | —         | —         | PTA5     | DISABLED                            |                                     | PTA5  | USB_CLKIN | TPM0_CH2 |            |           |      |         |
| 32         | 28         | —         | —         | PTA12    | DISABLED                            |                                     | PTA12 |           | TPM1_CH0 |            |           |      |         |
| 33         | 29         | —         | —         | PTA13    | DISABLED                            |                                     | PTA13 |           | TPM1_CH1 |            |           |      |         |
| 34         | —          | —         | —         | PTA14    | DISABLED                            |                                     | PTA14 | SPI0_PCS0 | UART0_TX |            |           |      |         |
| 35         | —          | —         | —         | PTA15    | DISABLED                            |                                     | PTA15 | SPI0_SCK  | UART0_RX |            |           |      |         |
| 36         | —          | —         | —         | PTA16    | DISABLED                            |                                     | PTA16 | SPI0_MOSI |          |            | SPI0_MISO |      |         |
| 37         | —          | —         | —         | PTA17    | DISABLED                            |                                     | PTA17 | SPI0_MISO |          |            | SPI0_MOSI |      |         |
| 38         | 30         | 22        | 15        | VDD      | VDD                                 | VDD                                 |       |           |          |            |           |      |         |

## Chapter 10 Signal Multiplexing and Signal Descriptions

| 80<br>LQFP | 64<br>LQFP | 48<br>QFN | 32<br>QFN | Pin Name                       | Default                 | ALT0                    | ALT1                           | ALT2      | ALT3            | ALT4       | ALT5      | ALT6            | ALT7 |
|------------|------------|-----------|-----------|--------------------------------|-------------------------|-------------------------|--------------------------------|-----------|-----------------|------------|-----------|-----------------|------|
| 39         | 31         | 23        | 16        | VSS                            | VSS                     |                         |                                |           |                 |            |           |                 |      |
| 40         | 32         | 24        | 17        | PTA18                          | EXTAL0                  | EXTAL0                  | PTA18                          |           | UART1_RX        | TPM_CLKIN0 |           |                 |      |
| 41         | 33         | 25        | 18        | PTA19                          | XTAL0                   | XTAL0                   | PTA19                          |           | UART1_TX        | TPM_CLKIN1 |           | LPTMR0_<br>ALT1 |      |
| 42         | 34         | 26        | 19        | RESET_b                        | RESET_b                 |                         | PTA20                          |           |                 |            |           |                 |      |
| 43         | 35         | 27        | 20        | PTB0/<br>LLWU_P5               | ADC0_SE8/<br>TSI0_CH0   | ADC0_SE8/<br>TSI0_CH0   | PTB0/<br>LLWU_P5               | I2C0_SCL  | TPM1_CH0        |            |           |                 |      |
| 44         | 36         | 28        | 21        | PTB1                           | ADC0_SE9/<br>TSI0_CH6   | ADC0_SE9/<br>TSI0_CH6   | PTB1                           | I2C0_SDA  | TPM1_CH1        |            |           |                 |      |
| 45         | 37         | 29        | —         | PTB2                           | ADC0_SE12/<br>TSI0_CH7  | ADC0_SE12/<br>TSI0_CH7  | PTB2                           | I2C0_SCL  | TPM2_CH0        |            |           |                 |      |
| 46         | 38         | 30        | —         | PTB3                           | ADC0_SE13/<br>TSI0_CH8  | ADC0_SE13/<br>TSI0_CH8  | PTB3                           | I2C0_SDA  | TPM2_CH1        |            |           |                 |      |
| 47         | —          | —         | —         | PTB8                           | DISABLED                |                         | PTB8                           |           | EXTRG_IN        |            |           |                 |      |
| 48         | —          | —         | —         | PTB9                           | DISABLED                |                         | PTB9                           |           |                 |            |           |                 |      |
| 49         | —          | —         | —         | PTB10                          | DISABLED                |                         | PTB10                          | SPI1_PCS0 |                 |            |           |                 |      |
| 50         | —          | —         | —         | PTB11                          | DISABLED                |                         | PTB11                          | SPI1_SCK  |                 |            |           |                 |      |
| 51         | 39         | 31        | —         | PTB16                          | TSI0_CH9                | TSI0_CH9                | PTB16                          | SPI1_MOSI | UART0_RX        | TPM_CLKIN0 | SPI1_MISO |                 |      |
| 52         | 40         | 32        | —         | PTB17                          | TSI0_CH10               | TSI0_CH10               | PTB17                          | SPI1_MISO | UART0_TX        | TPM_CLKIN1 | SPI1_MOSI |                 |      |
| 53         | 41         | —         | —         | PTB18                          | TSI0_CH11               | TSI0_CH11               | PTB18                          |           | TPM2_CH0        |            |           |                 |      |
| 54         | 42         | —         | —         | PTB19                          | TSI0_CH12               | TSI0_CH12               | PTB19                          |           | TPM2_CH1        |            |           |                 |      |
| 55         | 43         | 33        | —         | PTC0                           | ADC0_SE14/<br>TSI0_CH13 | ADC0_SE14/<br>TSI0_CH13 | PTC0                           |           | EXTRG_IN        |            | CMP0_OUT  |                 |      |
| 56         | 44         | 34        | 22        | PTC1/<br>LLWU_P6/<br>RTC_CLKIN | ADC0_SE15/<br>TSI0_CH14 | ADC0_SE15/<br>TSI0_CH14 | PTC1/<br>LLWU_P6/<br>RTC_CLKIN | I2C1_SCL  |                 | TPM0_CH0   |           |                 |      |
| 57         | 45         | 35        | 23        | PTC2                           | ADC0_SE11/<br>TSI0_CH15 | ADC0_SE11/<br>TSI0_CH15 | PTC2                           | I2C1_SDA  |                 | TPM0_CH1   |           |                 |      |
| 58         | 46         | 36        | 24        | PTC3/<br>LLWU_P7               | DISABLED                |                         | PTC3/<br>LLWU_P7               |           | UART1_RX        | TPM0_CH2   | CLKOUT    |                 |      |
| 59         | 47         | —         | —         | VSS                            | VSS                     | VSS                     |                                |           |                 |            |           |                 |      |
| 60         | 48         | —         | —         | VDD                            | VDD                     | VDD                     |                                |           |                 |            |           |                 |      |
| 61         | 49         | 37        | 25        | PTC4/<br>LLWU_P8               | DISABLED                |                         | PTC4/<br>LLWU_P8               | SPI0_PCS0 | UART1_TX        | TPM0_CH3   |           |                 |      |
| 62         | 50         | 38        | 26        | PTC5/<br>LLWU_P9               | DISABLED                |                         | PTC5/<br>LLWU_P9               | SPI0_SCK  | LPTMR0_<br>ALT2 |            |           | CMP0_OUT        |      |
| 63         | 51         | 39        | 27        | PTC6/<br>LLWU_P10              | CMP0_IN0                | CMP0_IN0                | PTC6/<br>LLWU_P10              | SPI0_MOSI | EXTRG_IN        |            | SPI0_MISO |                 |      |
| 64         | 52         | 40        | 28        | PTC7                           | CMP0_IN1                | CMP0_IN1                | PTC7                           | SPI0_MISO |                 |            | SPI0_MOSI |                 |      |
| 65         | 53         | —         | —         | PTC8                           | CMP0_IN2                | CMP0_IN2                | PTC8                           | I2C0_SCL  | TPM0_CH4        |            |           |                 |      |
| 66         | 54         | —         | —         | PTC9                           | CMP0_IN3                | CMP0_IN3                | PTC9                           | I2C0_SDA  | TPM0_CH5        |            |           |                 |      |
| 67         | 55         | —         | —         | PTC10                          | DISABLED                |                         | PTC10                          | I2C1_SCL  |                 |            |           |                 |      |
| 68         | 56         | —         | —         | PTC11                          | DISABLED                |                         | PTC11                          | I2C1_SDA  |                 |            |           |                 |      |
| 69         | —          | —         | —         | PTC12                          | DISABLED                |                         | PTC12                          |           |                 | TPM_CLKIN0 |           |                 |      |
| 70         | —          | —         | —         | PTC13                          | DISABLED                |                         | PTC13                          |           |                 | TPM_CLKIN1 |           |                 |      |

## Pinout

| 80<br>LQFP | 64<br>LQFP | 48<br>QFN | 32<br>QFN | Pin Name          | Default   | ALT0      | ALT1              | ALT2      | ALT3     | ALT4     | ALT5      | ALT6 | ALT7 |
|------------|------------|-----------|-----------|-------------------|-----------|-----------|-------------------|-----------|----------|----------|-----------|------|------|
| 71         | —          | —         | —         | PTC16             | DISABLED  |           | PTC16             |           |          |          |           |      |      |
| 72         | —          | —         | —         | PTC17             | DISABLED  |           | PTC17             |           |          |          |           |      |      |
| 73         | 57         | 41        | —         | PTD0              | DISABLED  |           | PTD0              | SPI0_PCS0 |          | TPM0_CH0 |           |      |      |
| 74         | 58         | 42        | —         | PTD1              | ADC0_SE5b | ADC0_SE5b | PTD1              | SPI0_SCK  |          | TPM0_CH1 |           |      |      |
| 75         | 59         | 43        | —         | PTD2              | DISABLED  |           | PTD2              | SPI0_MOSI | UART2_RX | TPM0_CH2 | SPI0_MISO |      |      |
| 76         | 60         | 44        | —         | PTD3              | DISABLED  |           | PTD3              | SPI0_MISO | UART2_TX | TPM0_CH3 | SPI0_MOSI |      |      |
| 77         | 61         | 45        | 29        | PTD4/<br>LLWU_P14 | DISABLED  |           | PTD4/<br>LLWU_P14 | SPI1_PCS0 | UART2_RX | TPM0_CH4 |           |      |      |
| 78         | 62         | 46        | 30        | PTD5              | ADC0_SE6b | ADC0_SE6b | PTD5              | SPI1_SCK  | UART2_TX | TPM0_CH5 |           |      |      |
| 79         | 63         | 47        | 31        | PTD6/<br>LLWU_P15 | ADC0_SE7b | ADC0_SE7b | PTD6/<br>LLWU_P15 | SPI1_MOSI | UART0_RX |          | SPI1_MISO |      |      |
| 80         | 64         | 48        | 32        | PTD7              | DISABLED  |           | PTD7              | SPI1_MISO | UART0_TX |          | SPI1_MOSI |      |      |

### 10.3.2 KL25 Pinouts

The below figures show the pinout diagrams for the devices supported by this document. Many signals may be multiplexed onto a single pin. To determine what signals can be used on which pin, see the previous section.

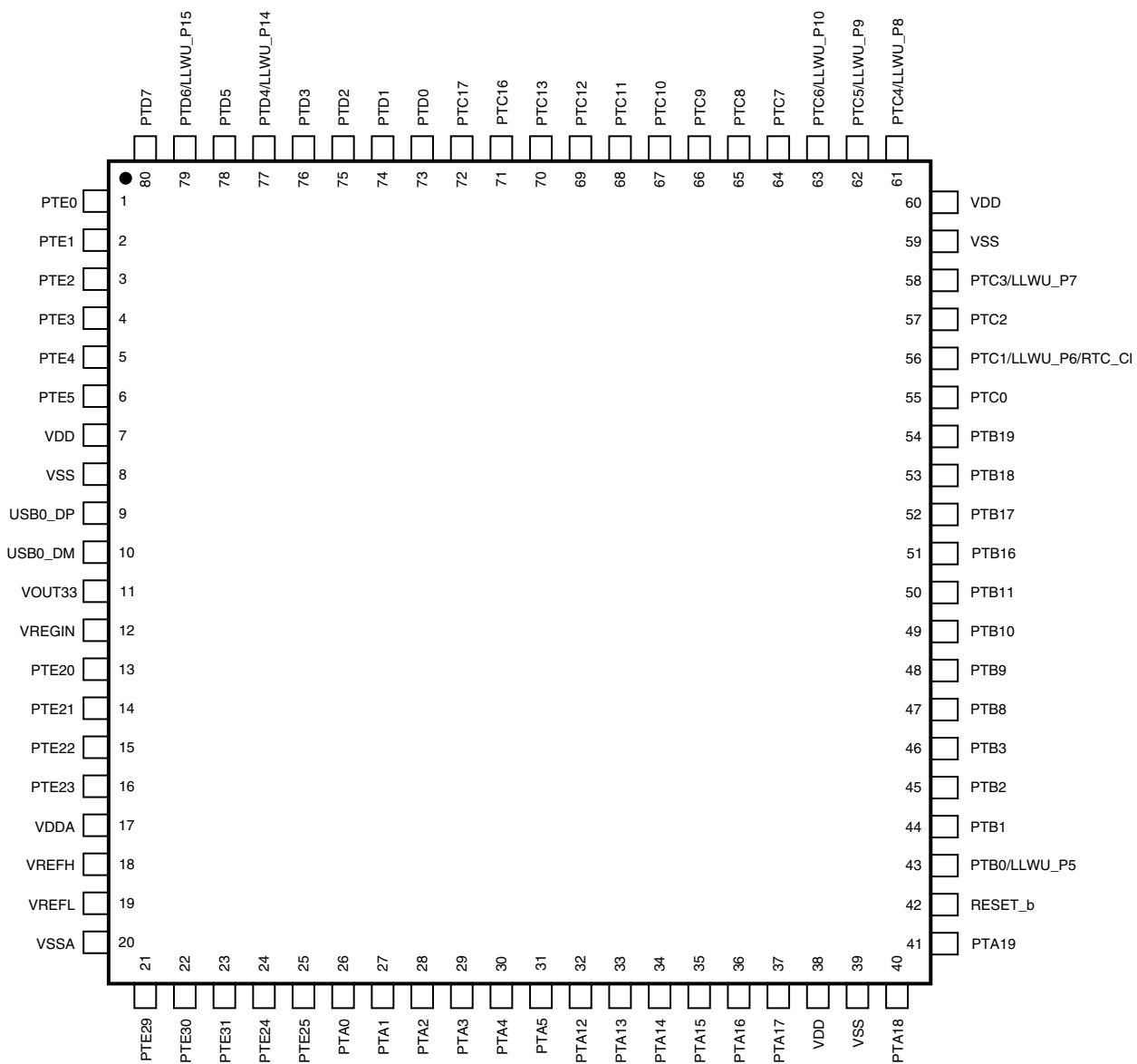
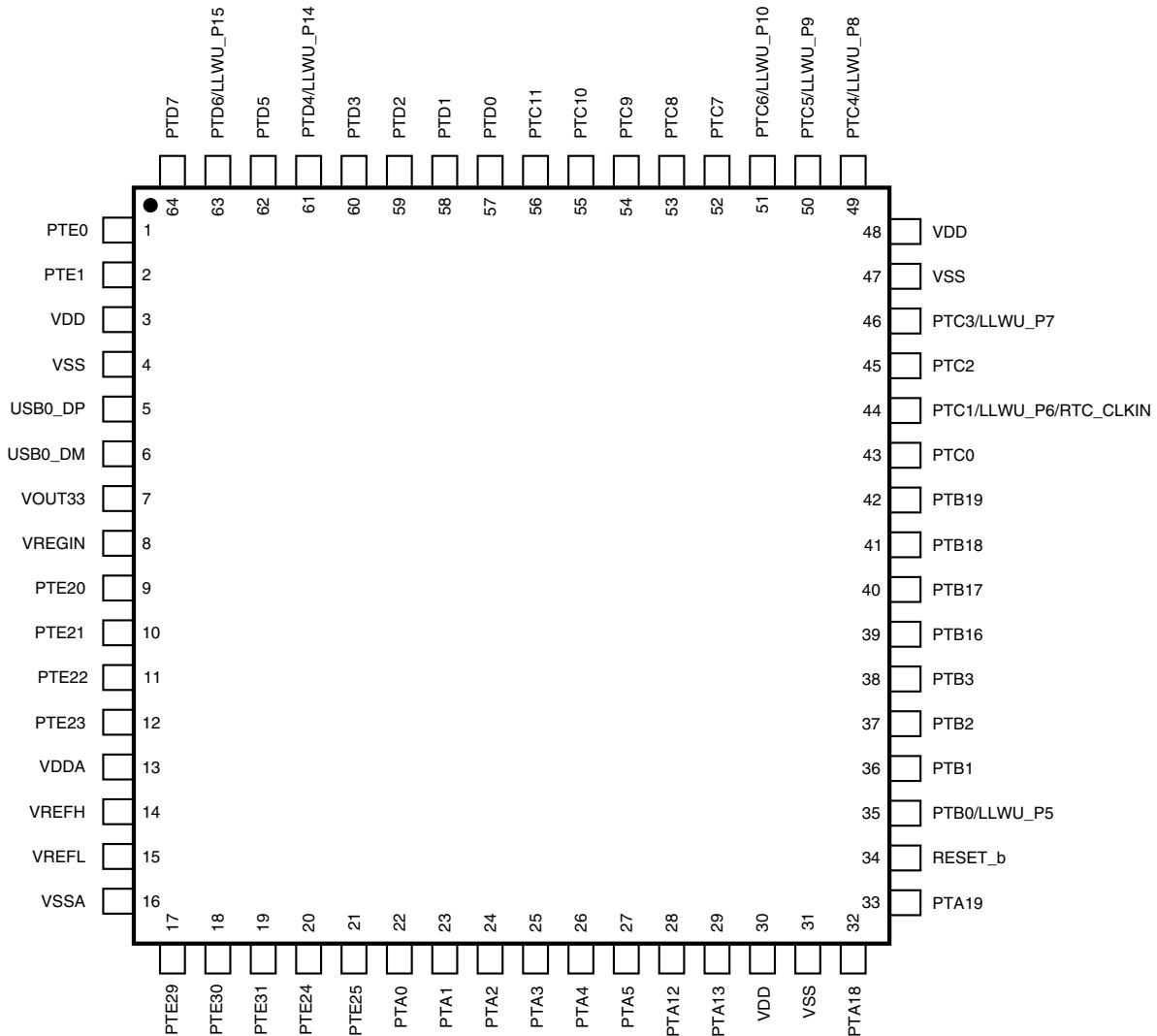
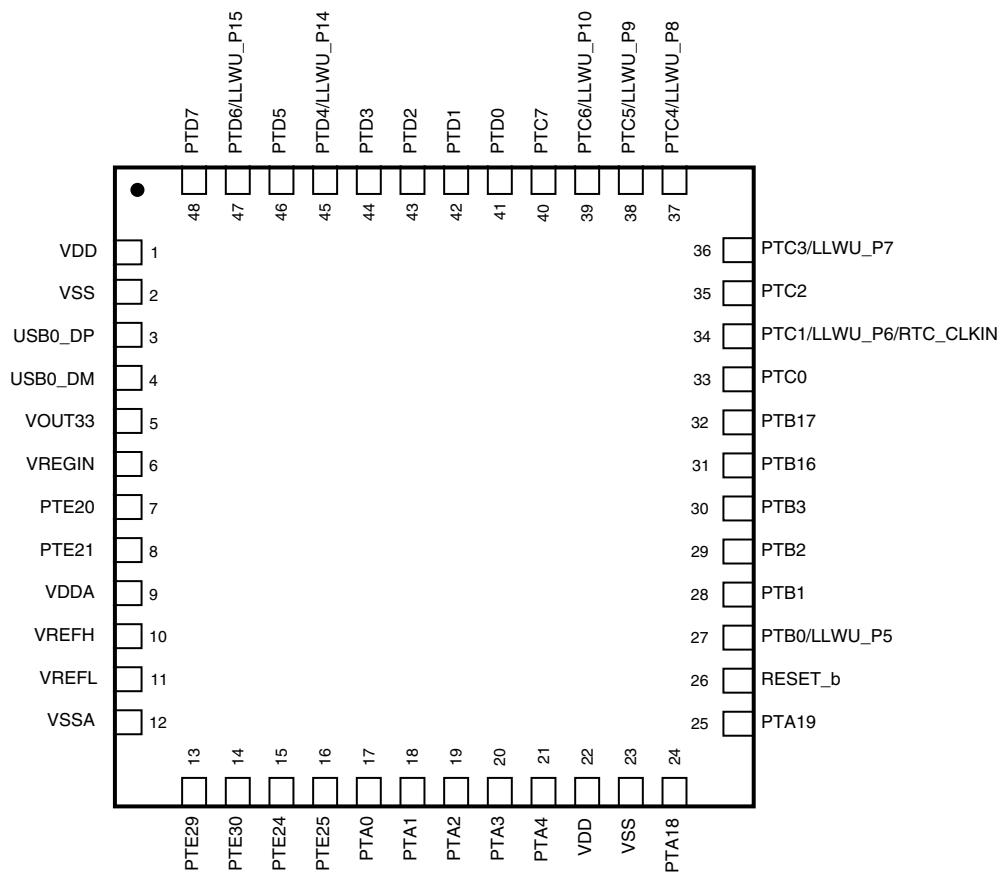


Figure 10-2. KL25 80-pin LQFP pinout diagram

## Pinout



**Figure 10-3. KL25 64-pin LQFP pinout diagram**

**Figure 10-4. KL25 48-pin QFN pinout diagram**

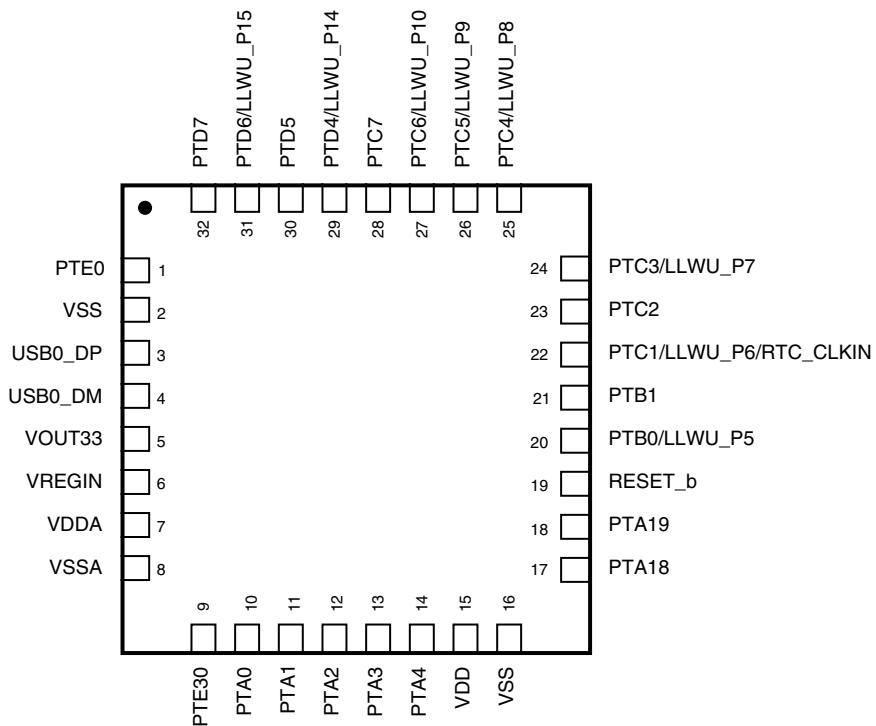


Figure 10-5. KL25 32-pin QFN pinout diagram

## 10.4 Module Signal Description Tables

The following sections correlate the chip-level signal name with the signal name used in the module's chapter. They also briefly describe the signal function and direction.

## 10.4.1 Core Modules

**Table 10-3. SWD Signal Descriptions**

| Chip signal name | Module signal name | Description  | I/O            |
|------------------|--------------------|--|----------------|
| SWD_DIO          | SWD_DIO            | Serial wire debug data input/output. The SWD_DIO pin is used by an external debug tool for communication and device control. This pin is pulled up internally. | Input / Output |
| SWD_CLK          | SWD_CLK            | Serial Wire Clock. This pin is the clock for debug logic when in the Serial Wire Debug mode. This pin is pulled down internally.                               | Input          |

## 10.4.2 System Modules

**Table 10-4. System Signal Descriptions**

| Chip signal name | Module signal name | Description  | I/O |
|------------------|--------------------|--|-----|
| NMI              | —                  | Non-maskable interrupt<br><br><b>NOTE:</b> Driving the NMI signal low forces a non-maskable interrupt, if the NMI function is selected on the corresponding pin. | I   |
| RESET            | —                  | Reset bi-directional signal  | I/O |
| VDD              | —                  | MCU power  | I   |
| VSS              | —                  | MCU ground   | I   |

## 10.4.3 Clock Modules

**Table 10-5. OSC Signal Descriptions**

| Chip signal name | Module signal name | Description                     | I/O |
|------------------|--------------------|---------------------------------|-----|
| EXTAL0           | EXTAL              | External clock/Oscillator input | I   |
| XTAL0            | XTAL               | Oscillator output               | O   |

## 10.4.4 Memories and Memory Interfaces

## 10.4.5 Analog

**Table 10-6. ADC 0 Signal Descriptions**

| Chip signal name | Module signal name | Description                        | I/O |
|------------------|--------------------|------------------------------------|-----|
| ADC0_DPn         | DADP3–DADP0        | Differential Analog Channel Inputs | I   |
| ADC0_DMn         | DADM3–DADM0        | Differential Analog Channel Inputs | I   |
| ADC0_SEn         | ADn                | Single-Ended Analog Channel Inputs | I   |
| VREFH            | V <sub>REFSH</sub> | Voltage Reference Select High      | I   |
| VREFL            | V <sub>REFSL</sub> | Voltage Reference Select Low       | I   |
| VDDA             | V <sub>DDA</sub>   | Analog Power Supply                | I   |
| VSSA             | V <sub>SSA</sub>   | Analog Ground                      | I   |

**Table 10-7. CMP 0 Signal Descriptions**

| Chip signal name | Module signal name | Description           | I/O |
|------------------|--------------------|-----------------------|-----|
| CMP0_IN[5:0]     | IN[5:0]            | Analog voltage inputs | I   |
| CMP0_OUT         | CMPO               | Comparator output     | O   |

**Table 10-8. DAC 0 Signal Descriptions**

| Chip signal name | Module signal name | Description | I/O |
|------------------|--------------------|-------------|-----|
| DAC0_OUT         | —                  | DAC output  | O   |

## 10.4.6 Timer Modules

**Table 10-9. TPM 0 Signal Descriptions**

| Chip signal name | Module signal name | Description  | I/O |
|------------------|--------------------|--|-----|
| TPM_CLKIN[2:0]   | EXTCLK             | External clock. FTM external clock can be selected to drive the FTM counter. | I   |
| TPM0_CH[5:0]     | CHn                | FTM channel (n), where n can be 7-0  | I/O |

**Table 10-10. TPM 1 Signal Descriptions**

| Chip signal name | Module signal name | Description   | I/O |
|------------------|--------------------|---|-----|
| TPM_CLKIN[2:0]   | TPM_EXTCLK         | External clock. TPM external clock can be selected to increment the TPM counter on every rising edge synchronized to the counter clock. | I   |

*Table continues on the next page...*

**Table 10-10. TPM 1 Signal Descriptions (continued)**

| Chip signal name | Module signal name | Description              | I/O |
|------------------|--------------------|--------------------------|-----|
| TPM1_CH[1:0]     | TPM_CHn            | TPM channel (n = 5 to 0) | I/O |

**Table 10-11. TPM 2 Signal Descriptions**

| Chip signal name | Module signal name | Description   | I/O |
|------------------|--------------------|---|-----|
| TPM_CLKIN[2:0]   | TPM_EXTCLK         | External clock. TPM external clock can be selected to increment the TPM counter on every rising edge synchronized to the counter clock. | I   |
| TPM2_CH[1:0]     | TPM_CHn            | TPM channel (n = 5 to 0)  | I/O |

**Table 10-12. LPTMR 0 Signal Descriptions**

| Chip signal name | Module signal name | Description             | I/O |
|------------------|--------------------|-------------------------|-----|
| LPTMR0_ALT[2:1]  | LPTMR_ALTn         | Pulse Counter Input pin | I   |

**Table 10-13. RTC Signal Descriptions**

| Chip signal name        | Module signal name | Description             | I/O |
|-------------------------|--------------------|-------------------------|-----|
| RTC_CLKOUT <sup>1</sup> | RTC_CLKOUT         | 1 Hz square-wave output | O   |

1. RTC\_CLKOUT can also be driven with OSCERCLK via SIM control bit SIM\_SOPT[RCTCLKOUTSEL]

## 10.4.7 Communication Interfaces

**Table 10-14. USB FS OTG Signal Descriptions**

| Chip signal name | Module signal name | Description                               | I/O |
|------------------|--------------------|---|-----|
| USB0_DM          | usb_dm             | USB D- analog data signal on the USB bus. | I/O |
| USB0_DP          | usb_dp             | USB D+ analog data signal on the USB bus. | I/O |
| USB_CLKIN        | —                  | Alternate USB clock input                 | I   |

**Table 10-15. USB VREG Signal Descriptions**

| Chip signal name | Module signal name | Description              | I/O |
|------------------|--------------------|--------------------------|-----|
| VOUT33           | reg33_out          | Regulator output voltage | O   |
| VREGIN           | reg33_in           | Unregulated power supply | I   |

**Table 10-16. SPI0 Signal Descriptions**

| Chip signal name | Module signal name | Description                    | I/O |
|------------------|--------------------|--------------------------------|-----|
| SPI0_MISO        | MISO               | Master Data In, Slave Data Out | I/O |
| SPI0_MOSI        | MOSI               | Master Data Out, Slave Data In | I/O |
| SPI0_SCLK        | SPSCK              | SPI Serial Clock               | I/O |
| SPI0_PCS0        | SS                 | Slave Select                   | I/O |

**Table 10-17. SPI1 Signal Descriptions**

| Chip signal name | Module signal name | Description                    | I/O |
|------------------|--------------------|--------------------------------|-----|
| SPI1_MISO        | MISO               | Master Data In, Slave Data Out | I/O |
| SPI1_MOSI        | MOSI               | Master Data Out, Slave Data In | I/O |
| SPI1_SCLK        | SPSCK              | SPI Serial Clock               | I/O |
| SPI1_PCS0        | SS                 | Slave Select                   | I/O |

**Table 10-18. I<sup>2</sup>C 0 Signal Descriptions**

| Chip signal name | Module signal name | Description   | I/O |
|------------------|--------------------|---|-----|
| I2C0_SCL         | SCL                | Bidirectional serial clock line of the I <sup>2</sup> C system. | I/O |
| I2C0_SDA         | SDA                | Bidirectional serial data line of the I <sup>2</sup> C system.  | I/O |

**Table 10-19. I<sup>2</sup>C 1 Signal Descriptions**

| Chip signal name | Module signal name | Description   | I/O |
|------------------|--------------------|---|-----|
| I2C1_SCL         | SCL                | Bidirectional serial clock line of the I <sup>2</sup> C system. | I/O |
| I2C1_SDA         | SDA                | Bidirectional serial data line of the I <sup>2</sup> C system.  | I/O |

**Table 10-20. UART 0 Signal Descriptions**

| Chip signal name | Module signal name | Description   | I/O |
|------------------|--------------------|---------------|-----|
| UART0_TX         | TXD                | Transmit data | O   |
| UART0_RX         | RXD                | Receive data  | I   |

**Table 10-21. UART 1 Signal Descriptions**

| Chip signal name | Module signal name | Description   | I/O |
|------------------|--------------------|---------------|-----|
| UART1_TX         | TXD                | Transmit data | O   |
| UART1_RX         | RXD                | Receive data  | I   |

**Table 10-22. UART 2 Signal Descriptions**

| Chip signal name | Module signal name | Description   | I/O |
|------------------|--------------------|---------------|-----|
| UART2_TX         | TXD                | Transmit data | O   |
| UART2_RX         | RXD                | Receive data  | I   |

## 10.4.8 Human-Machine Interfaces (HMI)

**Table 10-23. GPIO Signal Descriptions**

| Chip signal name       | Module signal name | Description                  | I/O |
|------------------------|--------------------|------------------------------|-----|
| PTA[31:0] <sup>1</sup> | PORATA31–PORATA0   | General-purpose input/output | I/O |
| PTB[31:0] <sup>1</sup> | PORB31–PORTB0      | General-purpose input/output | I/O |
| PTC[31:0] <sup>1</sup> | PORTC31–PORTC0     | General-purpose input/output | I/O |
| PTD[31:0] <sup>1</sup> | PORTD31–PORTD0     | General-purpose input/output | I/O |
| PTE[31:0] <sup>1</sup> | PORTE31–PORTE0     | General-purpose input/output | I/O |

1. The available GPIO pins depends on the specific package. See the signal multiplexing section for which exact GPIO signals are available.

**Table 10-24. TSI Signal Descriptions**

| Chip signal name | Module signal name | Description   | I/O |
|------------------|--------------------|---|-----|
| TSI0_CH[15:0]    | TSI[15:0]          | TSI capacitive pins. Switches driver that connects directly to the electrode pins TSI[15:0] can operate as GPIO pins. | I/O |



# Chapter 11

## Port control and interrupts (PORT)

### 11.1 Introduction

#### NOTE

For the chip-specific implementation details of this module's instances see the chip configuration information.

### 11.2 Overview

The port control and interrupt (PORT) module provides support for port control, and external interrupt functions. Most functions can be configured independently for each pin in the 32-bit port and affect the pin regardless of its pin muxing state.

There is one instance of the PORT module for each port. Not all pins within each port are implemented on a specific device.

#### 11.2.1 Features

The PORT module has the following features:

- Pin interrupt on selected pins
  - Interrupt flag and enable registers for each pin
  - Support for edge sensitive (rising, falling, both) or level sensitive (low, high) configured per pin
  - Support for interrupt or DMA request configured per pin
  - Asynchronous wakeup in Low-Power modes
  - Pin interrupt is functional in all digital Pin Muxing modes
- Port control

- Individual pull control fields with pullup, pulldown, and pull-disable support on selected pins
- Individual drive strength field supporting high and low drive strength on selected pins
- Individual slew rate field supporting fast and slow slew rates on selected pins
- Individual input passive filter field supporting enable and disable of the individual input passive filter on selected pins
- Individual mux control field supporting analog or pin disabled, GPIO, and up to six chip-specific digital functions
- Pad configuration fields are functional in all digital Pin Muxing modes

## **11.2.2 Modes of operation**

### **11.2.2.1 Run mode**

In Run mode, the PORT operates normally.

### **11.2.2.2 Wait mode**

In Wait mode, PORT continues to operate normally and may be configured to exit the Low-Power mode if an enabled interrupt is detected. DMA requests are still generated during the Wait mode, but do not cause an exit from the Low-Power mode.

### **11.2.2.3 Stop mode**

In Stop mode, the PORT can be configured to exit the Low-Power mode via an asynchronous wakeup signal if an enabled interrupt is detected.

### **11.2.2.4 Debug mode**

In Debug mode, PORT operates normally.

## **11.3 External signal description**

The following table describes the PORT external signal.

**Table 11-1. Signal properties**

| Name        | Function           | I/O | Reset | Pull |
|-------------|--------------------|-----|-------|------|
| PORTx[31:0] | External interrupt | I/O | 0     | -    |

**NOTE**

Not all pins within each port are implemented on each device.

## 11.4 Detailed signal description

The following table contains the detailed signal description for the PORT interface.

**Table 11-2. PORT interface—detailed signal description**

Signal



## PORT memory map (continued)

| Absolute address (hex) | Register name                                  | Width (in bits) | Access                | Reset value                 | Section/ page |
|------------------------|--|-----------------|-----------------------|-----------------------------|---------------|
| 4004_9020              | Pin Control Register n (PORTA_PCR8)            | 32              | R/W                   | <a href="#">See section</a> | 11.5.1/183    |
| 4004_9024              | Pin Control Register n (PORTA_PCR9)            | 32              | R/W                   | <a href="#">See section</a> | 11.5.1/183    |
| 4004_9028              | Pin Control Register n (PORTA_PCR10)           | 32              | R/W                   | <a href="#">See section</a> | 11.5.1/183    |
| 4004_902C              | Pin Control Register n (PORTA_PCR11)           | 32              | R/W                   | <a href="#">See section</a> | 11.5.1/183    |
| 4004_9030              | Pin Control Register n (PORTA_PCR12)           | 32              | R/W                   | <a href="#">See section</a> | 11.5.1/183    |
| 4004_9034              | Pin Control Register n (PORTA_PCR13)           | 32              | R/W                   | <a href="#">See section</a> | 11.5.1/183    |
| 4004_9038              | Pin Control Register n (PORTA_PCR14)           | 32              | R/W                   | <a href="#">See section</a> | 11.5.1/183    |
| 4004_903C              | Pin Control Register n (PORTA_PCR15)           | 32              | R/W                   | <a href="#">See section</a> | 11.5.1/183    |
| 4004_9040              | Pin Control Register n (PORTA_PCR16)           | 32              | R/W                   | <a href="#">See section</a> | 11.5.1/183    |
| 4004_9044              | Pin Control Register n (PORTA_PCR17)           | 32              | R/W                   | <a href="#">See section</a> | 11.5.1/183    |
| 4004_9048              | Pin Control Register n (PORTA_PCR18)           | 32              | R/W                   | <a href="#">See section</a> | 11.5.1/183    |
| 4004_904C              | Pin Control Register n (PORTA_PCR19)           | 32              | R/W                   | <a href="#">See section</a> | 11.5.1/183    |
| 4004_9050              | Pin Control Register n (PORTA_PCR20)           | 32              | R/W                   | <a href="#">See section</a> | 11.5.1/183    |
| 4004_9054              | Pin Control Register n (PORTA_PCR21)           | 32              | R/W                   | <a href="#">See section</a> | 11.5.1/183    |
| 4004_9058              | Pin Control Register n (PORTA_PCR22)           | 32              | R/W                   | <a href="#">See section</a> | 11.5.1/183    |
| 4004_905C              | Pin Control Register n (PORTA_PCR23)           | 32              | R/W                   | <a href="#">See section</a> | 11.5.1/183    |
| 4004_9060              | Pin Control Register n (PORTA_PCR24)           | 32              | R/W                   | <a href="#">See section</a> | 11.5.1/183    |
| 4004_9064              | Pin Control Register n (PORTA_PCR25)           | 32              | R/W                   | <a href="#">See section</a> | 11.5.1/183    |
| 4004_9068              | Pin Control Register n (PORTA_PCR26)           | 32              | R/W                   | <a href="#">See section</a> | 11.5.1/183    |
| 4004_906C              | Pin Control Register n (PORTA_PCR27)           | 32              | R/W                   | <a href="#">See section</a> | 11.5.1/183    |
| 4004_9070              | Pin Control Register n (PORTA_PCR28)           | 32              | R/W                   | <a href="#">See section</a> | 11.5.1/183    |
| 4004_9074              | Pin Control Register n (PORTA_PCR29)           | 32              | R/W                   | <a href="#">See section</a> | 11.5.1/183    |
| 4004_9078              | Pin Control Register n (PORTA_PCR30)           | 32              | R/W                   | <a href="#">See section</a> | 11.5.1/183    |
| 4004_907C              | Pin Control Register n (PORTA_PCR31)           | 32              | R/W                   | <a href="#">See section</a> | 11.5.1/183    |
| 4004_9080              | Global Pin Control Low Register (PORTA_GPCLR)  | 32              | W<br>(always reads 0) | 0000_0000h                  | 11.5.2/185    |
| 4004_9084              | Global Pin Control High Register (PORTA_GPCHR) | 32              | W<br>(always reads 0) | 0000_0000h                  | 11.5.3/186    |
| 4004_90A0              | Interrupt Status Flag Register (PORTA_ISFR)    | 32              | w1c                   | 0000_0000h                  | 11.5.4/186    |
| 4004_A000              | Pin Control Register n (PORTB_PCR0)            | 32              | R/W                   | <a href="#">See section</a> | 11.5.1/183    |
| 4004_A004              | Pin Control Register n (PORTB_PCR1)            | 32              | R/W                   | <a href="#">See section</a> | 11.5.1/183    |
| 4004_A008              | Pin Control Register n (PORTB_PCR2)            | 32              | R/W                   | <a href="#">See section</a> | 11.5.1/183    |
| 4004_A00C              | Pin Control Register n (PORTB_PCR3)            | 32              | R/W                   | <a href="#">See section</a> | 11.5.1/183    |
| 4004_A010              | Pin Control Register n (PORTB_PCR4)            | 32              | R/W                   | <a href="#">See section</a> | 11.5.1/183    |
| 4004_A014              | Pin Control Register n (PORTB_PCR5)            | 32              | R/W                   | <a href="#">See section</a> | 11.5.1/183    |
| 4004_A018              | Pin Control Register n (PORTB_PCR6)            | 32              | R/W                   | <a href="#">See section</a> | 11.5.1/183    |
| 4004_A01C              | Pin Control Register n (PORTB_PCR7)            | 32              | R/W                   | <a href="#">See section</a> | 11.5.1/183    |

Table continues on the next page...

## PORT memory map (continued)

| Absolute address (hex) | Register name                                  | Width (in bits) | Access                | Reset value                 | Section/ page |
|------------------------|--|-----------------|-----------------------|-----------------------------|---------------|
| 4004_A020              | Pin Control Register n (PORTB_PCR8)            | 32              | R/W                   | <a href="#">See section</a> | 11.5.1/183    |
| 4004_A024              | Pin Control Register n (PORTB_PCR9)            | 32              | R/W                   | <a href="#">See section</a> | 11.5.1/183    |
| 4004_A028              | Pin Control Register n (PORTB_PCR10)           | 32              | R/W                   | <a href="#">See section</a> | 11.5.1/183    |
| 4004_A02C              | Pin Control Register n (PORTB_PCR11)           | 32              | R/W                   | <a href="#">See section</a> | 11.5.1/183    |
| 4004_A030              | Pin Control Register n (PORTB_PCR12)           | 32              | R/W                   | <a href="#">See section</a> | 11.5.1/183    |
| 4004_A034              | Pin Control Register n (PORTB_PCR13)           | 32              | R/W                   | <a href="#">See section</a> | 11.5.1/183    |
| 4004_A038              | Pin Control Register n (PORTB_PCR14)           | 32              | R/W                   | <a href="#">See section</a> | 11.5.1/183    |
| 4004_A03C              | Pin Control Register n (PORTB_PCR15)           | 32              | R/W                   | <a href="#">See section</a> | 11.5.1/183    |
| 4004_A040              | Pin Control Register n (PORTB_PCR16)           | 32              | R/W                   | <a href="#">See section</a> | 11.5.1/183    |
| 4004_A044              | Pin Control Register n (PORTB_PCR17)           | 32              | R/W                   | <a href="#">See section</a> | 11.5.1/183    |
| 4004_A048              | Pin Control Register n (PORTB_PCR18)           | 32              | R/W                   | <a href="#">See section</a> | 11.5.1/183    |
| 4004_A04C              | Pin Control Register n (PORTB_PCR19)           | 32              | R/W                   | <a href="#">See section</a> | 11.5.1/183    |
| 4004_A050              | Pin Control Register n (PORTB_PCR20)           | 32              | R/W                   | <a href="#">See section</a> | 11.5.1/183    |
| 4004_A054              | Pin Control Register n (PORTB_PCR21)           | 32              | R/W                   | <a href="#">See section</a> | 11.5.1/183    |
| 4004_A058              | Pin Control Register n (PORTB_PCR22)           | 32              | R/W                   | <a href="#">See section</a> | 11.5.1/183    |
| 4004_A05C              | Pin Control Register n (PORTB_PCR23)           | 32              | R/W                   | <a href="#">See section</a> | 11.5.1/183    |
| 4004_A060              | Pin Control Register n (PORTB_PCR24)           | 32              | R/W                   | <a href="#">See section</a> | 11.5.1/183    |
| 4004_A064              | Pin Control Register n (PORTB_PCR25)           | 32              | R/W                   | <a href="#">See section</a> | 11.5.1/183    |
| 4004_A068              | Pin Control Register n (PORTB_PCR26)           | 32              | R/W                   | <a href="#">See section</a> | 11.5.1/183    |
| 4004_A06C              | Pin Control Register n (PORTB_PCR27)           | 32              | R/W                   | <a href="#">See section</a> | 11.5.1/183    |
| 4004_A070              | Pin Control Register n (PORTB_PCR28)           | 32              | R/W                   | <a href="#">See section</a> | 11.5.1/183    |
| 4004_A074              | Pin Control Register n (PORTB_PCR29)           | 32              | R/W                   | <a href="#">See section</a> | 11.5.1/183    |
| 4004_A078              | Pin Control Register n (PORTB_PCR30)           | 32              | R/W                   | <a href="#">See section</a> | 11.5.1/183    |
| 4004_A07C              | Pin Control Register n (PORTB_PCR31)           | 32              | R/W                   | <a href="#">See section</a> | 11.5.1/183    |
| 4004_A080              | Global Pin Control Low Register (PORTB_GPCLR)  | 32              | W<br>(always reads 0) | 0000_0000h                  | 11.5.2/185    |
| 4004_A084              | Global Pin Control High Register (PORTB_GPCHR) | 32              | W<br>(always reads 0) | 0000_0000h                  | 11.5.3/186    |
| 4004_A0A0              | Interrupt Status Flag Register (PORTB_ISFR)    | 32              | w1c                   | 0000_0000h                  | 11.5.4/186    |
| 4004_B000              | Pin Control Register n (PORTC_PCR0)            | 32              | R/W                   | <a href="#">See section</a> | 11.5.1/183    |
| 4004_B004              | Pin Control Register n (PORTC_PCR1)            | 32              | R/W                   | <a href="#">See section</a> | 11.5.1/183    |
| 4004_B008              | Pin Control Register n (PORTC_PCR2)            | 32              | R/W                   | <a href="#">See section</a> | 11.5.1/183    |
| 4004_B00C              | Pin Control Register n (PORTC_PCR3)            | 32              | R/W                   | <a href="#">See section</a> | 11.5.1/183    |
| 4004_B010              | Pin Control Register n (PORTC_PCR4)            | 32              | R/W                   | <a href="#">See section</a> | 11.5.1/183    |
| 4004_B014              | Pin Control Register n (PORTC_PCR5)            | 32              | R/W                   | <a href="#">See section</a> | 11.5.1/183    |
| 4004_B018              | Pin Control Register n (PORTC_PCR6)            | 32              | R/W                   | <a href="#">See section</a> | 11.5.1/183    |
| 4004_B01C              | Pin Control Register n (PORTC_PCR7)            | 32              | R/W                   | <a href="#">See section</a> | 11.5.1/183    |

Table continues on the next page...

## PORT memory map (continued)

| Absolute address (hex) | Register name                                  | Width (in bits) | Access                | Reset value                 | Section/ page |
|------------------------|--|-----------------|-----------------------|-----------------------------|---------------|
| 4004_B020              | Pin Control Register n (PORTC_PCR8)            | 32              | R/W                   | <a href="#">See section</a> | 11.5.1/183    |
| 4004_B024              | Pin Control Register n (PORTC_PCR9)            | 32              | R/W                   | <a href="#">See section</a> | 11.5.1/183    |
| 4004_B028              | Pin Control Register n (PORTC_PCR10)           | 32              | R/W                   | <a href="#">See section</a> | 11.5.1/183    |
| 4004_B02C              | Pin Control Register n (PORTC_PCR11)           | 32              | R/W                   | <a href="#">See section</a> | 11.5.1/183    |
| 4004_B030              | Pin Control Register n (PORTC_PCR12)           | 32              | R/W                   | <a href="#">See section</a> | 11.5.1/183    |
| 4004_B034              | Pin Control Register n (PORTC_PCR13)           | 32              | R/W                   | <a href="#">See section</a> | 11.5.1/183    |
| 4004_B038              | Pin Control Register n (PORTC_PCR14)           | 32              | R/W                   | <a href="#">See section</a> | 11.5.1/183    |
| 4004_B03C              | Pin Control Register n (PORTC_PCR15)           | 32              | R/W                   | <a href="#">See section</a> | 11.5.1/183    |
| 4004_B040              | Pin Control Register n (PORTC_PCR16)           | 32              | R/W                   | <a href="#">See section</a> | 11.5.1/183    |
| 4004_B044              | Pin Control Register n (PORTC_PCR17)           | 32              | R/W                   | <a href="#">See section</a> | 11.5.1/183    |
| 4004_B048              | Pin Control Register n (PORTC_PCR18)           | 32              | R/W                   | <a href="#">See section</a> | 11.5.1/183    |
| 4004_B04C              | Pin Control Register n (PORTC_PCR19)           | 32              | R/W                   | <a href="#">See section</a> | 11.5.1/183    |
| 4004_B050              | Pin Control Register n (PORTC_PCR20)           | 32              | R/W                   | <a href="#">See section</a> | 11.5.1/183    |
| 4004_B054              | Pin Control Register n (PORTC_PCR21)           | 32              | R/W                   | <a href="#">See section</a> | 11.5.1/183    |
| 4004_B058              | Pin Control Register n (PORTC_PCR22)           | 32              | R/W                   | <a href="#">See section</a> | 11.5.1/183    |
| 4004_B05C              | Pin Control Register n (PORTC_PCR23)           | 32              | R/W                   | <a href="#">See section</a> | 11.5.1/183    |
| 4004_B060              | Pin Control Register n (PORTC_PCR24)           | 32              | R/W                   | <a href="#">See section</a> | 11.5.1/183    |
| 4004_B064              | Pin Control Register n (PORTC_PCR25)           | 32              | R/W                   | <a href="#">See section</a> | 11.5.1/183    |
| 4004_B068              | Pin Control Register n (PORTC_PCR26)           | 32              | R/W                   | <a href="#">See section</a> | 11.5.1/183    |
| 4004_B06C              | Pin Control Register n (PORTC_PCR27)           | 32              | R/W                   | <a href="#">See section</a> | 11.5.1/183    |
| 4004_B070              | Pin Control Register n (PORTC_PCR28)           | 32              | R/W                   | <a href="#">See section</a> | 11.5.1/183    |
| 4004_B074              | Pin Control Register n (PORTC_PCR29)           | 32              | R/W                   | <a href="#">See section</a> | 11.5.1/183    |
| 4004_B078              | Pin Control Register n (PORTC_PCR30)           | 32              | R/W                   | <a href="#">See section</a> | 11.5.1/183    |
| 4004_B07C              | Pin Control Register n (PORTC_PCR31)           | 32              | R/W                   | <a href="#">See section</a> | 11.5.1/183    |
| 4004_B080              | Global Pin Control Low Register (PORTC_GPCLR)  | 32              | W<br>(always reads 0) | 0000_0000h                  | 11.5.2/185    |
| 4004_B084              | Global Pin Control High Register (PORTC_GPCHR) | 32              | W<br>(always reads 0) | 0000_0000h                  | 11.5.3/186    |
| 4004_B0A0              | Interrupt Status Flag Register (PORTC_ISFR)    | 32              | w1c                   | 0000_0000h                  | 11.5.4/186    |
| 4004_C000              | Pin Control Register n (PORTD_PCR0)            | 32              | R/W                   | <a href="#">See section</a> | 11.5.1/183    |
| 4004_C004              | Pin Control Register n (PORTD_PCR1)            | 32              | R/W                   | <a href="#">See section</a> | 11.5.1/183    |
| 4004_C008              | Pin Control Register n (PORTD_PCR2)            | 32              | R/W                   | <a href="#">See section</a> | 11.5.1/183    |
| 4004_C00C              | Pin Control Register n (PORTD_PCR3)            | 32              | R/W                   | <a href="#">See section</a> | 11.5.1/183    |
| 4004_C010              | Pin Control Register n (PORTD_PCR4)            | 32              | R/W                   | <a href="#">See section</a> | 11.5.1/183    |
| 4004_C014              | Pin Control Register n (PORTD_PCR5)            | 32              | R/W                   | <a href="#">See section</a> | 11.5.1/183    |
| 4004_C018              | Pin Control Register n (PORTD_PCR6)            | 32              | R/W                   | <a href="#">See section</a> | 11.5.1/183    |
| 4004_C01C              | Pin Control Register n (PORTD_PCR7)            | 32              | R/W                   | <a href="#">See section</a> | 11.5.1/183    |

Table continues on the next page...

## PORT memory map (continued)

| Absolute address (hex) | Register name                                  | Width (in bits) | Access                | Reset value                 | Section/ page |
|------------------------|--|-----------------|-----------------------|-----------------------------|---------------|
| 4004_C020              | Pin Control Register n (PORTD_PCR8)            | 32              | R/W                   | <a href="#">See section</a> | 11.5.1/183    |
| 4004_C024              | Pin Control Register n (PORTD_PCR9)            | 32              | R/W                   | <a href="#">See section</a> | 11.5.1/183    |
| 4004_C028              | Pin Control Register n (PORTD_PCR10)           | 32              | R/W                   | <a href="#">See section</a> | 11.5.1/183    |
| 4004_C02C              | Pin Control Register n (PORTD_PCR11)           | 32              | R/W                   | <a href="#">See section</a> | 11.5.1/183    |
| 4004_C030              | Pin Control Register n (PORTD_PCR12)           | 32              | R/W                   | <a href="#">See section</a> | 11.5.1/183    |
| 4004_C034              | Pin Control Register n (PORTD_PCR13)           | 32              | R/W                   | <a href="#">See section</a> | 11.5.1/183    |
| 4004_C038              | Pin Control Register n (PORTD_PCR14)           | 32              | R/W                   | <a href="#">See section</a> | 11.5.1/183    |
| 4004_C03C              | Pin Control Register n (PORTD_PCR15)           | 32              | R/W                   | <a href="#">See section</a> | 11.5.1/183    |
| 4004_C040              | Pin Control Register n (PORTD_PCR16)           | 32              | R/W                   | <a href="#">See section</a> | 11.5.1/183    |
| 4004_C044              | Pin Control Register n (PORTD_PCR17)           | 32              | R/W                   | <a href="#">See section</a> | 11.5.1/183    |
| 4004_C048              | Pin Control Register n (PORTD_PCR18)           | 32              | R/W                   | <a href="#">See section</a> | 11.5.1/183    |
| 4004_C04C              | Pin Control Register n (PORTD_PCR19)           | 32              | R/W                   | <a href="#">See section</a> | 11.5.1/183    |
| 4004_C050              | Pin Control Register n (PORTD_PCR20)           | 32              | R/W                   | <a href="#">See section</a> | 11.5.1/183    |
| 4004_C054              | Pin Control Register n (PORTD_PCR21)           | 32              | R/W                   | <a href="#">See section</a> | 11.5.1/183    |
| 4004_C058              | Pin Control Register n (PORTD_PCR22)           | 32              | R/W                   | <a href="#">See section</a> | 11.5.1/183    |
| 4004_C05C              | Pin Control Register n (PORTD_PCR23)           | 32              | R/W                   | <a href="#">See section</a> | 11.5.1/183    |
| 4004_C060              | Pin Control Register n (PORTD_PCR24)           | 32              | R/W                   | <a href="#">See section</a> | 11.5.1/183    |
| 4004_C064              | Pin Control Register n (PORTD_PCR25)           | 32              | R/W                   | <a href="#">See section</a> | 11.5.1/183    |
| 4004_C068              | Pin Control Register n (PORTD_PCR26)           | 32              | R/W                   | <a href="#">See section</a> | 11.5.1/183    |
| 4004_C06C              | Pin Control Register n (PORTD_PCR27)           | 32              | R/W                   | <a href="#">See section</a> | 11.5.1/183    |
| 4004_C070              | Pin Control Register n (PORTD_PCR28)           | 32              | R/W                   | <a href="#">See section</a> | 11.5.1/183    |
| 4004_C074              | Pin Control Register n (PORTD_PCR29)           | 32              | R/W                   | <a href="#">See section</a> | 11.5.1/183    |
| 4004_C078              | Pin Control Register n (PORTD_PCR30)           | 32              | R/W                   | <a href="#">See section</a> | 11.5.1/183    |
| 4004_C07C              | Pin Control Register n (PORTD_PCR31)           | 32              | R/W                   | <a href="#">See section</a> | 11.5.1/183    |
| 4004_C080              | Global Pin Control Low Register (PORTD_GPCLR)  | 32              | W<br>(always reads 0) | 0000_0000h                  | 11.5.2/185    |
| 4004_C084              | Global Pin Control High Register (PORTD_GPCHR) | 32              | W<br>(always reads 0) | 0000_0000h                  | 11.5.3/186    |
| 4004_C0A0              | Interrupt Status Flag Register (PORTD_ISFR)    | 32              | w1c                   | 0000_0000h                  | 11.5.4/186    |
| 4004_D000              | Pin Control Register n (PORTE_PCR0)            | 32              | R/W                   | <a href="#">See section</a> | 11.5.1/183    |
| 4004_D004              | Pin Control Register n (PORTE_PCR1)            | 32              | R/W                   | <a href="#">See section</a> | 11.5.1/183    |
| 4004_D008              | Pin Control Register n (PORTE_PCR2)            | 32              | R/W                   | <a href="#">See section</a> | 11.5.1/183    |
| 4004_D00C              | Pin Control Register n (PORTE_PCR3)            | 32              | R/W                   | <a href="#">See section</a> | 11.5.1/183    |
| 4004_D010              | Pin Control Register n (PORTE_PCR4)            | 32              | R/W                   | <a href="#">See section</a> | 11.5.1/183    |
| 4004_D014              | Pin Control Register n (PORTE_PCR5)            | 32              | R/W                   | <a href="#">See section</a> | 11.5.1/183    |
| 4004_D018              | Pin Control Register n (PORTE_PCR6)            | 32              | R/W                   | <a href="#">See section</a> | 11.5.1/183    |
| 4004_D01C              | Pin Control Register n (PORTE_PCR7)            | 32              | R/W                   | <a href="#">See section</a> | 11.5.1/183    |

Table continues on the next page...

## PORT memory map (continued)

| Absolute address (hex) | Register name                                  | Width (in bits) | Access                | Reset value                 | Section/ page |
|------------------------|--|-----------------|-----------------------|-----------------------------|---------------|
| 4004_D020              | Pin Control Register n (PORTE_PCR8)            | 32              | R/W                   | <a href="#">See section</a> | 11.5.1/183    |
| 4004_D024              | Pin Control Register n (PORTE_PCR9)            | 32              | R/W                   | <a href="#">See section</a> | 11.5.1/183    |
| 4004_D028              | Pin Control Register n (PORTE_PCR10)           | 32              | R/W                   | <a href="#">See section</a> | 11.5.1/183    |
| 4004_D02C              | Pin Control Register n (PORTE_PCR11)           | 32              | R/W                   | <a href="#">See section</a> | 11.5.1/183    |
| 4004_D030              | Pin Control Register n (PORTE_PCR12)           | 32              | R/W                   | <a href="#">See section</a> | 11.5.1/183    |
| 4004_D034              | Pin Control Register n (PORTE_PCR13)           | 32              | R/W                   | <a href="#">See section</a> | 11.5.1/183    |
| 4004_D038              | Pin Control Register n (PORTE_PCR14)           | 32              | R/W                   | <a href="#">See section</a> | 11.5.1/183    |
| 4004_D03C              | Pin Control Register n (PORTE_PCR15)           | 32              | R/W                   | <a href="#">See section</a> | 11.5.1/183    |
| 4004_D040              | Pin Control Register n (PORTE_PCR16)           | 32              | R/W                   | <a href="#">See section</a> | 11.5.1/183    |
| 4004_D044              | Pin Control Register n (PORTE_PCR17)           | 32              | R/W                   | <a href="#">See section</a> | 11.5.1/183    |
| 4004_D048              | Pin Control Register n (PORTE_PCR18)           | 32              | R/W                   | <a href="#">See section</a> | 11.5.1/183    |
| 4004_D04C              | Pin Control Register n (PORTE_PCR19)           | 32              | R/W                   | <a href="#">See section</a> | 11.5.1/183    |
| 4004_D050              | Pin Control Register n (PORTE_PCR20)           | 32              | R/W                   | <a href="#">See section</a> | 11.5.1/183    |
| 4004_D054              | Pin Control Register n (PORTE_PCR21)           | 32              | R/W                   | <a href="#">See section</a> | 11.5.1/183    |
| 4004_D058              | Pin Control Register n (PORTE_PCR22)           | 32              | R/W                   | <a href="#">See section</a> | 11.5.1/183    |
| 4004_D05C              | Pin Control Register n (PORTE_PCR23)           | 32              | R/W                   | <a href="#">See section</a> | 11.5.1/183    |
| 4004_D060              | Pin Control Register n (PORTE_PCR24)           | 32              | R/W                   | <a href="#">See section</a> | 11.5.1/183    |
| 4004_D064              | Pin Control Register n (PORTE_PCR25)           | 32              | R/W                   | <a href="#">See section</a> | 11.5.1/183    |
| 4004_D068              | Pin Control Register n (PORTE_PCR26)           | 32              | R/W                   | <a href="#">See section</a> | 11.5.1/183    |
| 4004_D06C              | Pin Control Register n (PORTE_PCR27)           | 32              | R/W                   | <a href="#">See section</a> | 11.5.1/183    |
| 4004_D070              | Pin Control Register n (PORTE_PCR28)           | 32              | R/W                   | <a href="#">See section</a> | 11.5.1/183    |
| 4004_D074              | Pin Control Register n (PORTE_PCR29)           | 32              | R/W                   | <a href="#">See section</a> | 11.5.1/183    |
| 4004_D078              | Pin Control Register n (PORTE_PCR30)           | 32              | R/W                   | <a href="#">See section</a> | 11.5.1/183    |
| 4004_D07C              | Pin Control Register n (PORTE_PCR31)           | 32              | R/W                   | <a href="#">See section</a> | 11.5.1/183    |
| 4004_D080              | Global Pin Control Low Register (PORTE_GPCLR)  |                 |                       | 0000_0000h                  | 11.5.2/185    |
| 4004_D084              | Global Pin Control High Register (PORTE_GPCHR) | 32              | W<br>(always reads 0) | 0000_0000h                  | 11.5.3/186    |
| 4004_D0A0              | Interrupt Status Flag Register (PORTE_ISFR)    | 32              | w1c                   | 0000_0000h                  | 11.5.4/186    |

## 11.5.1 Pin Control Register n (PORTx\_PCRn)

### NOTE

Refer to the Signal Multiplexing and Signal Descriptions chapter for the reset value of this device.

See the GPIO Configuration section for details on the available functions for each pin.

Do not modify pin configuration registers associated with pins not available in your selected package. All un-bonded pins not available in your package will default to DISABLE state for lowest power consumption.

Address: Base address + 0h offset + (4d × i), where i=0d to 31d

| Bit   | 31 | 30 | 29 | 28 | 27 | 26  | 25 | 24  | 23 | 22  | 21 | 20  | 19   | 18  | 17 | 16 |
|-------|----|----|----|----|----|-----|----|-----|----|-----|----|-----|------|-----|----|----|
| R     | 0  |    |    |    |    |     |    | ISF | 0  |     |    |     | IRQC |     |    |    |
| W     |    |    |    |    |    |     |    | w1c |    |     |    |     |      |     |    |    |
| Reset | 0  | 0  | 0  | 0  | 0  | 0   | 0  | 0   | 0  | 0   | 0  | 0   | 0    | 0   | 0  | 0  |
| Bit   | 15 | 14 | 13 | 12 | 11 | 10  | 9  | 8   | 7  | 6   | 5  | 4   | 3    | 2   | 1  | 0  |
| R     | 0  |    |    |    |    | MUX |    |     | 0  | DSE | 0  | PFE | 0    | SRE | PE | PS |
| W     |    |    |    |    |    |     |    |     |    |     |    |     |      |     |    |    |
| Reset | 0  | 0  | 0  | 0  | 0  | x*  | x* | x*  | 0  | x*  | 0  | x*  | 0    | x*  | x* | x* |

\* Notes:

- x = Undefined at reset.

### PORTx\_PCRn field descriptions

| Field             | Description  |
|-------------------|--|
| 31–25<br>Reserved | This field is reserved.<br>This read-only field is reserved and always has the value 0.  |
| 24<br>ISF         | Interrupt Status Flag<br><br>This bit is read only for pins that do not support interrupt generation.<br><br>The pin interrupt configuration is valid in all digital pin muxing modes.<br><br>0 Configured interrupt is not detected.<br>1 Configured interrupt is detected. If the pin is configured to generate a DMA request, then the corresponding flag will be cleared automatically at the completion of the requested DMA transfer. Otherwise, the flag remains set until a logic one is written to the flag. If the pin is configured for a level sensitive interrupt and the pin remains asserted, then the flag is set again immediately after it is cleared. |

Table continues on the next page...

**PORTx\_PCRn field descriptions (continued)**

| Field             | Description   |      |  |      |   |      |                                |      |                                |      |                                |      |                                |      |                                |      |                                |      |                           |        |           |
|-------------------|---|------|--|------|---|------|--------------------------------|------|--------------------------------|------|--------------------------------|------|--------------------------------|------|--------------------------------|------|--------------------------------|------|---------------------------|--------|-----------|
| 23–20<br>Reserved | This field is reserved.<br>This read-only field is reserved and always has the value 0.   |      |  |      |   |      |                                |      |                                |      |                                |      |                                |      |                                |      |                                |      |                           |        |           |
| 19–16<br>IRQC     | <p>Interrupt Configuration</p> <p>This field is read only for pins that do not support interrupt generation.</p> <p>The pin interrupt configuration is valid in all digital pin muxing modes. The corresponding pin is configured to generate interrupt/DMA request as follows:</p> <table> <tr><td>0000</td><td>Interrupt/DMA request disabled.</td></tr> <tr><td>0001</td><td>DMA request on rising edge.</td></tr> <tr><td>0010</td><td>DMA request on falling edge.</td></tr> <tr><td>0011</td><td>DMA request on either edge.</td></tr> <tr><td>1000</td><td>Interrupt when logic zero.</td></tr> <tr><td>1001</td><td>Interrupt on rising edge.</td></tr> <tr><td>1010</td><td>Interrupt on falling edge.</td></tr> <tr><td>1011</td><td>Interrupt on either edge.</td></tr> <tr><td>1100</td><td>Interrupt when logic one.</td></tr> <tr><td>Others</td><td>Reserved.</td></tr> </table> | 0000 | Interrupt/DMA request disabled.  | 0001 | DMA request on rising edge.   | 0010 | DMA request on falling edge.   | 0011 | DMA request on either edge.    | 1000 | Interrupt when logic zero.     | 1001 | Interrupt on rising edge.      | 1010 | Interrupt on falling edge.     | 1011 | Interrupt on either edge.      | 1100 | Interrupt when logic one. | Others | Reserved. |
| 0000              | Interrupt/DMA request disabled.   |      |  |      |   |      |                                |      |                                |      |                                |      |                                |      |                                |      |                                |      |                           |        |           |
| 0001              | DMA request on rising edge.   |      |  |      |   |      |                                |      |                                |      |                                |      |                                |      |                                |      |                                |      |                           |        |           |
| 0010              | DMA request on falling edge.  |      |  |      |   |      |                                |      |                                |      |                                |      |                                |      |                                |      |                                |      |                           |        |           |
| 0011              | DMA request on either edge.   |      |  |      |   |      |                                |      |                                |      |                                |      |                                |      |                                |      |                                |      |                           |        |           |
| 1000              | Interrupt when logic zero.  |      |  |      |   |      |                                |      |                                |      |                                |      |                                |      |                                |      |                                |      |                           |        |           |
| 1001              | Interrupt on rising edge.   |      |  |      |   |      |                                |      |                                |      |                                |      |                                |      |                                |      |                                |      |                           |        |           |
| 1010              | Interrupt on falling edge.  |      |  |      |   |      |                                |      |                                |      |                                |      |                                |      |                                |      |                                |      |                           |        |           |
| 1011              | Interrupt on either edge.   |      |  |      |   |      |                                |      |                                |      |                                |      |                                |      |                                |      |                                |      |                           |        |           |
| 1100              | Interrupt when logic one.   |      |  |      |   |      |                                |      |                                |      |                                |      |                                |      |                                |      |                                |      |                           |        |           |
| Others            | Reserved.   |      |  |      |   |      |                                |      |                                |      |                                |      |                                |      |                                |      |                                |      |                           |        |           |
| 15–11<br>Reserved | This field is reserved.<br>This read-only field is reserved and always has the value 0.   |      |  |      |   |      |                                |      |                                |      |                                |      |                                |      |                                |      |                                |      |                           |        |           |
| 10–8<br>MUX       | <p>Pin Mux Control</p> <p>Not all pins support all pin muxing slots. Unimplemented pin muxing slots are reserved and may result in configuring the pin for a different pin muxing slot.</p> <p>The corresponding pin is configured in the following pin muxing slot as follows:</p> <table> <tr><td>000</td><td>Pin disabled (analog).</td></tr> <tr><td>001</td><td>Alternative 1 (GPIO).</td></tr> <tr><td>010</td><td>Alternative 2 (chip-specific).</td></tr> <tr><td>011</td><td>Alternative 3 (chip-specific).</td></tr> <tr><td>100</td><td>Alternative 4 (chip-specific).</td></tr> <tr><td>101</td><td>Alternative 5 (chip-specific).</td></tr> <tr><td>110</td><td>Alternative 6 (chip-specific).</td></tr> <tr><td>111</td><td>Alternative 7 (chip-specific).</td></tr> </table>   | 000  | Pin disabled (analog).   | 001  | Alternative 1 (GPIO).   | 010  | Alternative 2 (chip-specific). | 011  | Alternative 3 (chip-specific). | 100  | Alternative 4 (chip-specific). | 101  | Alternative 5 (chip-specific). | 110  | Alternative 6 (chip-specific). | 111  | Alternative 7 (chip-specific). |      |                           |        |           |
| 000               | Pin disabled (analog).  |      |  |      |   |      |                                |      |                                |      |                                |      |                                |      |                                |      |                                |      |                           |        |           |
| 001               | Alternative 1 (GPIO).   |      |  |      |   |      |                                |      |                                |      |                                |      |                                |      |                                |      |                                |      |                           |        |           |
| 010               | Alternative 2 (chip-specific).  |      |  |      |   |      |                                |      |                                |      |                                |      |                                |      |                                |      |                                |      |                           |        |           |
| 011               | Alternative 3 (chip-specific).  |      |  |      |   |      |                                |      |                                |      |                                |      |                                |      |                                |      |                                |      |                           |        |           |
| 100               | Alternative 4 (chip-specific).  |      |  |      |   |      |                                |      |                                |      |                                |      |                                |      |                                |      |                                |      |                           |        |           |
| 101               | Alternative 5 (chip-specific).  |      |  |      |   |      |                                |      |                                |      |                                |      |                                |      |                                |      |                                |      |                           |        |           |
| 110               | Alternative 6 (chip-specific).  |      |  |      |   |      |                                |      |                                |      |                                |      |                                |      |                                |      |                                |      |                           |        |           |
| 111               | Alternative 7 (chip-specific).  |      |  |      |   |      |                                |      |                                |      |                                |      |                                |      |                                |      |                                |      |                           |        |           |
| 7<br>Reserved     | This field is reserved.<br>This read-only field is reserved and always has the value 0.   |      |  |      |   |      |                                |      |                                |      |                                |      |                                |      |                                |      |                                |      |                           |        |           |
| 6<br>DSE          | <p>Drive Strength Enable</p> <p>This bit is read only for pins that do not support a configurable drive strength.</p> <p>Drive strength configuration is valid in all digital pin muxing modes.</p> <table> <tr><td>0</td><td>Low drive strength is configured on the corresponding pin, if pin is configured as a digital output.</td></tr> <tr><td>1</td><td>High drive strength is configured on the corresponding pin, if pin is configured as a digital output.</td></tr> </table>   | 0    | Low drive strength is configured on the corresponding pin, if pin is configured as a digital output. | 1    | High drive strength is configured on the corresponding pin, if pin is configured as a digital output. |      |                                |      |                                |      |                                |      |                                |      |                                |      |                                |      |                           |        |           |
| 0                 | Low drive strength is configured on the corresponding pin, if pin is configured as a digital output.  |      |  |      |   |      |                                |      |                                |      |                                |      |                                |      |                                |      |                                |      |                           |        |           |
| 1                 | High drive strength is configured on the corresponding pin, if pin is configured as a digital output.   |      |  |      |   |      |                                |      |                                |      |                                |      |                                |      |                                |      |                                |      |                           |        |           |
| 5<br>Reserved     | This field is reserved.<br>This read-only field is reserved and always has the value 0.   |      |  |      |   |      |                                |      |                                |      |                                |      |                                |      |                                |      |                                |      |                           |        |           |
| 4<br>PFE          | <p>Passive Filter Enable</p> <p>This bit is read only for pins that do not support a configurable passive input filter.</p> <p>Passive filter configuration is valid in all digital pin muxing modes.</p>   |      |  |      |   |      |                                |      |                                |      |                                |      |                                |      |                                |      |                                |      |                           |        |           |

*Table continues on the next page...*

**PORTx\_PCRn field descriptions (continued)**

| Field      | Description   |
|------------|---|
|            | <p>0 Passive input filter is disabled on the corresponding pin.</p> <p>1 Passive input filter is enabled on the corresponding pin, if the pin is configured as a digital input. Refer to the device data sheet for filter characteristics.</p>  |
| 3 Reserved | This field is reserved.<br>This read-only field is reserved and always has the value 0.   |
| 2 SRE      | <p>Slew Rate Enable</p> <p>This bit is read only for pins that do not support a configurable slew rate.</p> <p>Slew rate configuration is valid in all digital pin muxing modes.</p> <p>0 Fast slew rate is configured on the corresponding pin, if the pin is configured as a digital output.</p> <p>1 Slow slew rate is configured on the corresponding pin, if the pin is configured as a digital output.</p>  |
| 1 PE       | <p>Pull Enable</p> <p>This bit is read only for pins that do not support a configurable pull resistor. Refer to the Chapter of Signal Multiplexing and Signal Descriptions for the pins that support a configurable pull resistor.</p> <p>Pull configuration is valid in all digital pin muxing modes.</p> <p>0 Internal pullup or pulldown resistor is not enabled on the corresponding pin.</p> <p>1 Internal pullup or pulldown resistor is enabled on the corresponding pin, if the pin is configured as a digital input.</p> |
| 0 PS       | <p>Pull Select</p> <p>This bit is read only for pins that do not support a configurable pull resistor direction.</p> <p>Pull configuration is valid in all digital pin muxing modes.</p> <p>0 Internal pulldown resistor is enabled on the corresponding pin, if the corresponding Port Pull Enable field is set.</p> <p>1 Internal pullup resistor is enabled on the corresponding pin, if the corresponding Port Pull Enable field is set.</p>  |

**11.5.2 Global Pin Control Low Register (PORTx\_GPCLR)**

Only 32-bit writes are supported to this register.

Address: Base address + 80h offset

| Bit   | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|-------|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|---|---|---|---|---|---|---|---|---|---|
| R     |    |    |    |    |    |    |    |    |    |    |    |    |    |    |    |    |    |    |    |    |    |    |   |   |   |   |   |   |   |   |   |   |
|       |    |    |    |    |    |    |    |    |    |    |    |    |    |    |    |    |    |    |    |    |    |    |   |   |   |   |   |   |   |   |   |   |
| W     |    |    |    |    |    |    |    |    |    |    |    |    |    |    |    |    |    |    |    |    |    |    |   |   |   |   |   |   |   |   |   |   |
| Reset | 0  | 0  | 0  | 0  | 0  | 0  | 0  | 0  | 0  | 0  | 0  | 0  | 0  | 0  | 0  | 0  | 0  | 0  | 0  | 0  | 0  | 0  | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |   |

**PORTx\_GPCLR field descriptions**

| Field      | Description   |
|------------|---|
| 31–16 GPWE | <p>Global Pin Write Enable</p> <p>Selects which Pin Control Registers (15 through 0) bits [15:0] update with the value in GPWD.</p> |

*Table continues on the next page...*

**PORTx\_GPCLR field descriptions (continued)**

| Field        | Description  |
|--------------|--|
|              | 0 Corresponding Pin Control Register is not updated with the value in GPWD.<br>1 Corresponding Pin Control Register is updated with the value in GPWD. |
| 15–0<br>GPWD | Global Pin Write Data<br><br>Write value that is written to all Pin Control Registers bits [15:0] that are selected by GPWE.                           |

**11.5.3 Global Pin Control High Register (PORTx\_GPCHR)**

Only 32-bit writes are supported to this register.

Address: Base address + 84h offset

| Bit   | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|-------|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|---|---|---|---|---|---|---|---|---|---|
| R     |    |    |    |    |    |    |    |    |    |    |    |    |    |    |    |    |    |    |    |    |    |    |   |   |   |   |   |   |   |   |   |   |
| W     |    |    |    |    |    |    |    |    |    |    |    |    |    |    |    |    |    |    |    |    |    |    |   |   |   |   |   |   |   |   |   |   |
| Reset | 0  | 0  | 0  | 0  | 0  | 0  | 0  | 0  | 0  | 0  | 0  | 0  | 0  | 0  | 0  | 0  | 0  | 0  | 0  | 0  | 0  | 0  | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |   |

**PORTx\_GPCHR field descriptions**

| Field         | Description   |
|---------------|---|
| 31–16<br>GPWE | Global Pin Write Enable<br><br>Selects which Pin Control Registers (31 through 16) bits [15:0] update with the value in GPWD.<br><br>0 Corresponding Pin Control Register is not updated with the value in GPWD.<br>1 Corresponding Pin Control Register is updated with the value in GPWD. |
| 15–0<br>GPWD  | Global Pin Write Data<br><br>Write value that is written to all Pin Control Registers bits [15:0] that are selected by GPWE.  |

**11.5.4 Interrupt Status Flag Register (PORTx\_ISFR)**

The corresponding bit is read only for pins that do not support interrupt generation.

The pin interrupt configuration is valid in all digital pin muxing modes. The Interrupt Status Flag for each pin is also visible in the corresponding Pin Control Register, and each flag can be cleared in either location.

Address: Base address + A0h offset

| Bit   | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|-------|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|---|---|---|---|---|---|---|---|---|---|
| R     |    |    |    |    |    |    |    |    |    |    |    |    |    |    |    |    |    |    |    |    |    |    |   |   |   |   |   |   |   |   |   |   |
| W     |    |    |    |    |    |    |    |    |    |    |    |    |    |    |    |    |    |    |    |    |    |    |   |   |   |   |   |   |   |   |   |   |
| Reset | 0  | 0  | 0  | 0  | 0  | 0  | 0  | 0  | 0  | 0  | 0  | 0  | 0  | 0  | 0  | 0  | 0  | 0  | 0  | 0  | 0  | 0  | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |   |

**PORTx\_ISFR field descriptions**

| Field       | Description   |
|-------------|---|
| 31–0<br>ISF | <p>Interrupt Status Flag</p> <p>Each bit in the field indicates the detection of the configured interrupt of the same number as the field.</p> <p>0 Configured interrupt is not detected.<br/>1 Configured interrupt is detected. If the pin is configured to generate a DMA request, then the corresponding flag will be cleared automatically at the completion of the requested DMA transfer. Otherwise, the flag remains set until a logic one is written to the flag. If the pin is configured for a level sensitive interrupt and the pin remains asserted, then the flag is set again immediately after it is cleared.</p> |

## 11.6 Functional description

### 11.6.1 Pin control

Each port pin has a corresponding pin control register, PORT\_PCRn, associated with it.

The upper half of the pin control register configures the pin's capability to either interrupt the CPU or request a DMA transfer, on a rising/falling edge or both edges as well as a logic level occurring on the port pin. It also includes a flag to indicate that an interrupt has occurred.

The lower half of the pin control register configures the following functions for each pin within the 32-bit port.

- Pullup or pulldown enable on selected pins
- Drive strength and slew rate configuration on selected pins
- Passive input filter enable on selected pins
- Pin Muxing mode

The functions apply across all digital Pin Muxing modes and individual peripherals do not override the configuration in the pin control register. For example, if an I<sup>2</sup>C function is enabled on a pin, that does not override the pullup configuration for that pin.

When the Pin Muxing mode is configured for analog or is disabled, all the digital functions on that pin are disabled. This includes the pullup and pulldown enables, and passive filter enable.

The configuration of each pin control register is retained when the PORT module is disabled.

## 11.6.2 Global pin control

The two global pin control registers allow a single register write to update the lower half of the pin control register on up to sixteen pins, all with the same value.

The global pin control registers are designed to enable software to quickly configure multiple pins within the one port for the same peripheral function. However, the interrupt functions cannot be configured using the global pin control registers.

The global pin control registers are write-only registers, that always read as zero.

## 11.6.3 External interrupts

The external interrupt capability of the PORT module is available in all digital pin muxing modes provided the PORT module is enabled.

Each pin can be individually configured for any of the following external interrupt modes:

- Interrupt disabled, default out of reset
- Active high level sensitive interrupt
- Active low level sensitive interrupt
- Rising edge sensitive interrupt
- Falling edge sensitive interrupt
- Rising and falling edge sensitive interrupt
- Rising edge sensitive DMA request
- Falling edge sensitive DMA request
- Rising and falling edge sensitive DMA request

The interrupt status flag is set when the configured edge or level is detected on the output of the pin. When not in Stop mode, the input is first synchronized to the bus clock to detect the configured level or edge transition.

The PORT module generates a single interrupt that asserts when the interrupt status flag is set for any enabled interrupt for that port. The interrupt negates after the interrupt status flags for all enabled interrupts have been cleared by writing a logic 0 to the ISF flag in the PORT\_PCRn register.

The PORT module generates a single DMA request that asserts when the interrupt status flag is set for any enabled DMA request in that port. The DMA request negates after the DMA transfer is completed, because that clears the interrupt status flags for all enabled DMA requests.

During Stop mode, the interrupt status flag for any enabled interrupt is asynchronously set if the required level or edge is detected. This also generates an asynchronous wakeup signal to exit the Low-Power mode.



# **Chapter 12**

## **System integration module (SIM)**

### **12.1 Introduction**

The system integration module (SIM) provides system control and chip configuration registers.

#### **12.1.1 Features**

- System clocking configuration
  - System clock divide values
  - Architectural clock gating control
  - ERCLK32K clock selection
  - USB clock selection
  - UART0 and TPM clock selection
- Flash and System RAM size configuration
- USB regulator configuration
- TPM external clock and input capture selection
- UART receive/transmit source selection/configuration

### **12.2 Memory map and register definition**

The SIM module contains many bitfields for selecting the clock source and dividers for various module clocks.

**NOTE**

The SIM registers can be written only in supervisor mode. In user mode, write accesses are blocked and will result in a bus error.

**NOTE**

The SIM\_SOPT1 and SIM\_SOPT1CFG registers are located at a different base address than the other SIM registers.

**SIM memory map**

| Absolute address (hex) | Register name                                       | Width (in bits) | Access | Reset value                 | Section/ page |
|------------------------|---|-----------------|--------|-----------------------------|---------------|
| 4004_7000              | System Options Register 1 (SIM_SOPT1)               | 32              | R/W    | <a href="#">See section</a> | 12.2.1/193    |
| 4004_7004              | SOPT1 Configuration Register (SIM_SOPT1CFG)         | 32              | R/W    | 0000_0000h                  | 12.2.2/194    |
| 4004_8004              | System Options Register 2 (SIM_SOPT2)               | 32              | R/W    | 0000_0000h                  | 12.2.3/195    |
| 4004_800C              | System Options Register 4 (SIM_SOPT4)               | 32              | R/W    | 0000_0000h                  | 12.2.4/197    |
| 4004_8010              | System Options Register 5 (SIM_SOPT5)               | 32              | R/W    | 0000_0000h                  | 12.2.5/199    |
| 4004_8018              | System Options Register 7 (SIM_SOPT7)               | 32              | R/W    | 0000_0000h                  | 12.2.6/200    |
| 4004_8024              | System Device Identification Register (SIM_SDID)    | 32              | R      | <a href="#">See section</a> | 12.2.7/202    |
| 4004_8034              | System Clock Gating Control Register 4 (SIM_SCGC4)  | 32              | R/W    | F000_0030h                  | 12.2.8/204    |
| 4004_8038              | System Clock Gating Control Register 5 (SIM_SCGC5)  | 32              | R/W    | 0000_0180h                  | 12.2.9/206    |
| 4004_803C              | System Clock Gating Control Register 6 (SIM_SCGC6)  | 32              | R/W    | 0000_0001h                  | 12.2.10/207   |
| 4004_8040              | System Clock Gating Control Register 7 (SIM_SCGC7)  | 32              | R/W    | 0000_0100h                  | 12.2.11/209   |
| 4004_8044              | System Clock Divider Register 1 (SIM_CLKDIV1)       | 32              | R/W    | <a href="#">See section</a> | 12.2.12/210   |
| 4004_804C              | Flash Configuration Register 1 (SIM_FCFG1)          | 32              | R/W    | <a href="#">See section</a> | 12.2.13/211   |
| 4004_8050              | Flash Configuration Register 2 (SIM_FCFG2)          | 32              | R      | <a href="#">See section</a> | 12.2.14/213   |
| 4004_8058              | Unique Identification Register Mid-High (SIM_UIDMH) | 32              | R      | <a href="#">See section</a> | 12.2.15/213   |
| 4004_805C              | Unique Identification Register Mid Low (SIM_UIDML)  | 32              | R      | <a href="#">See section</a> | 12.2.16/214   |
| 4004_8060              | Unique Identification Register Low (SIM_UIDL)       | 32              | R      | <a href="#">See section</a> | 12.2.17/214   |
| 4004_8100              | COP Control Register (SIM_COPC)                     | 32              | R/W    | 0000_000Ch                  | 12.2.18/215   |
| 4004_8104              | Service COP Register (SIM_SRVCOP)                   | 32              | W      | 0000_0000h                  | 12.2.19/216   |

## 12.2.1 System Options Register 1 (SIM\_SOPT1)

### NOTE

The SOPT1 register is only reset on POR or LVD.

Address: 4004\_7000h base + 0h offset = 4004\_7000h

| Bit   | 31       | 30       | 29       | 28 | 27 | 26 | 25 | 24 | 23 | 22 | 21 | 20 | 19        | 18 | 17 | 16 |
|-------|----------|----------|----------|----|----|----|----|----|----|----|----|----|-----------|----|----|----|
| R     | USBREGEN | USBSSTBY | USBVSTBY |    |    |    |    | 0  |    |    |    |    |           |    | 0  |    |
| W     |          |          |          |    |    |    |    |    |    |    |    |    | OSC32KSEL |    |    |    |
| Reset | 1        | 0        | 0        | 0  | 0  | 0  | 0  | 0  | 0  | 0  | 0  | 0  | 0         | 0  | 0  | 0  |
| Bit   | 15       | 14       | 13       | 12 | 11 | 10 | 9  | 8  | 7  | 6  | 5  | 4  | 3         | 2  | 1  | 0  |
| R     |          |          |          |    |    |    | 0  |    |    |    |    |    |           |    |    |    |
| W     |          |          |          |    |    |    |    |    |    |    |    |    |           |    |    |    |
| Reset | 0        | 0        | 0        | 0  | 0  | 0  | 0  | 0  | 0  | 0  | *  | *  | *         | *  | *  | *  |

\* Notes:

- Reserved field: Device specific value.

### SIM\_SOPT1 field descriptions

| Field             | Description   |
|-------------------|---|
| 31<br>USBREGEN    | USB voltage regulator enable<br><br>Controls whether the USB voltage regulator is enabled.<br><br>0 USB voltage regulator is disabled.<br>1 USB voltage regulator is enabled.   |
| 30<br>USBSSTBY    | USB voltage regulator in standby mode during Stop, VLPS, LLS and VLLS modes.<br><br>Controls whether the USB voltage regulator is placed in standby mode during Stop, VLPS, LLS and VLLS modes.<br><br>0 USB voltage regulator not in standby during Stop, VLPS, LLS and VLLS modes.<br>1 USB voltage regulator in standby during Stop, VLPS, LLS and VLLS modes. |
| 29<br>USBVSTBY    | USB voltage regulator in standby mode during VLPR and VLPW modes<br><br>Controls whether the USB voltage regulator is placed in standby mode during VLPR and VLPW modes.<br><br>0 USB voltage regulator not in standby during VLPR and VLPW modes.<br>1 USB voltage regulator in standby during VLPR and VLPW modes.  |
| 28–20<br>Reserved | This field is reserved.<br>This read-only field is reserved and always has the value 0.   |

Table continues on the next page...

**SIM\_SOPT1 field descriptions (continued)**

| Field              | Description   |
|--------------------|---|
| 19–18<br>OSC32KSEL | 32K oscillator clock select<br><br>Selects the 32 kHz clock source (ERCLK32K) for RTC and LPTMR. This bit is reset only on POR/LVD.<br><br>00 System oscillator (OSC32KCLK)<br>01 Reserved<br>10 RTC_CLKIN<br>11 LPO 1kHz |
| 17–6<br>Reserved   | This field is reserved.<br>This read-only field is reserved and always has the value 0.   |
| 5–0<br>Reserved    | This field is reserved.   |

**12.2.2 SOPT1 Configuration Register (SIM\_SOPT1CFG)****NOTE**

The SOPT1CFG register is reset on System Reset not VLLS.

Address: 4004\_7000h base + 4h offset = 4004\_7004h

| Bit   | 31 | 30 | 29 | 28 | 27 | 26 | 25    | 24 | 23    | 22 | 21   | 20 | 19 | 18 | 17 | 16 |
|-------|----|----|----|----|----|----|-------|----|-------|----|------|----|----|----|----|----|
| R     |    |    |    |    |    | 0  |       |    |       |    |      |    |    | 0  |    |    |
| W     |    |    |    |    |    |    | USSWE |    | UVSWE |    | URWE |    |    |    |    |    |
| Reset | 0  | 0  | 0  | 0  | 0  | 0  | 0     | 0  | 0     | 0  | 0    | 0  | 0  | 0  | 0  | 0  |
| Bit   | 15 | 14 | 13 | 12 | 11 | 10 | 9     | 8  | 7     | 6  | 5    | 4  | 3  | 2  | 1  | 0  |
| R     |    |    |    |    |    |    |       |    | 0     |    |      |    |    |    |    |    |
| W     |    |    |    |    |    |    |       |    |       |    |      |    |    |    |    |    |
| Reset | 0  | 0  | 0  | 0  | 0  | 0  | 0     | 0  | 0     | 0  | 0    | 0  | 0  | 0  | 0  | 0  |

**SIM\_SOPT1CFG field descriptions**

| Field             | Description  |
|-------------------|--|
| 31–27<br>Reserved | This field is reserved.<br>This read-only field is reserved and always has the value 0.  |
| 26<br>USSWE       | USB voltage regulator stop standby write enable<br><br>Writing one to the USSWE bit allows the SOPT1 USBSSTBY bit to be written. This register bit clears after a write to USBSSTBY. |

*Table continues on the next page...*

**SIM\_SOPT1CFG field descriptions (continued)**

| Field         | Description  |
|---------------|--|
|               | 0 SOPT1 USBSSTB cannot be written.<br>1 SOPT1 USBSSTB can be written.  |
| 25 UVSWE      | USB voltage regulator VLP standby write enable<br><br>Writing one to the UVSWE bit allows the SOPT1 USBVSTBY bit to be written. This register bit clears after a write to USBVSTBY.<br><br>0 SOPT1 USBVSTB cannot be written.<br>1 SOPT1 USBVSTB can be written. |
| 24 URWE       | USB voltage regulator enable write enable<br><br>Writing one to the URWE bit allows the SOPT1 USBREGEN bit to be written. This register bit clears after a write to USBREGEN.<br><br>0 SOPT1 USBREGEN cannot be written.<br>1 SOPT1 USBREGEN can be written.     |
| 23–0 Reserved | This field is reserved.<br>This read-only field is reserved and always has the value 0.  |

**12.2.3 System Options Register 2 (SIM\_SOPT2)**

SOPT2 contains the controls for selecting many of the module clock source options on this device. See the Clock Distribution chapter for more information including clocking diagrams and definitions of device clocks.

Address: 4004\_7000h base + 1004h offset = 4004\_8004h

| Bit   | 31 | 30 | 29 | 28 | 27 | 26       | 25 | 24     | 23 | 22        | 21 | 20             | 19 | 18     | 17 | 16       |
|-------|----|----|----|----|----|----------|----|--------|----|-----------|----|----------------|----|--------|----|----------|
| R     |    |    |    | 0  |    |          |    |        |    |           |    |                | 0  |        |    |          |
| W     |    |    |    |    |    | UART0SRC |    | TPMSRC |    |           |    |                |    | USBSRC | 0  | PLLFLSEL |
| Reset | 0  | 0  | 0  | 0  | 0  | 0        | 0  | 0      | 0  | 0         | 0  | 0              | 0  | 0      | 0  | 0        |
| Bit   | 15 | 14 | 13 | 12 | 11 | 10       | 9  | 8      | 7  | 6         | 5  | 4              | 3  | 2      | 1  | 0        |
| R     |    |    |    |    |    |          |    | 0      |    |           |    |                |    | 0      |    |          |
| W     |    |    |    |    |    |          |    |        |    | CLKOUTSEL |    | RTCCCLKOUTS_EL |    |        |    |          |
| Reset | 0  | 0  | 0  | 0  | 0  | 0        | 0  | 0      | 0  | 0         | 0  | 0              | 0  | 0      | 0  | 0        |

**SIM\_SOPT2 field descriptions**

| Field             | Description   |
|-------------------|---|
| 31–28<br>Reserved | This field is reserved.<br>This read-only field is reserved and always has the value 0.   |
| 27–26<br>UART0SRC | UART0 clock source select<br><br>Selects the clock source for the UART0 transmit and receive clock.<br><br>00 Clock disabled<br>01 MCGFLLCLK clock or MCGPLLCLK/2 clock<br>10 OSCERCLK clock<br>11 MCGIRCLK clock |
| 25–24<br>TPMSRC   | TPM clock source select<br><br>Selects the clock source for the TPM counter clock<br><br>00 Clock disabled<br>01 MCGFLLCLK clock or MCGPLLCLK/2<br>10 OSCERCLK clock<br>11 MCGIRCLK clock                         |
| 23–19<br>Reserved | This field is reserved.<br>This read-only field is reserved and always has the value 0.   |
| 18<br>USBSRC      | USB clock source select<br><br>Selects the clock source for the USB 48 MHz clock.<br><br>0 External bypass clock (USB_CLKIN).<br>1 MCGPLLCLK/2 or MCGFLLCLK clock   |
| 17<br>Reserved    | This field is reserved.<br>This read-only field is reserved and always has the value 0.   |
| 16<br>PLLFLSEL    | PLL/FLL clock select<br><br>Selects the MCGPLLCLK or MCGFLLCLK clock for various peripheral clocking options.<br><br>0 MCGFLLCLK clock<br>1 MCGPLLCLK clock with fixed divide by two                              |
| 15–8<br>Reserved  | This field is reserved.<br>This read-only field is reserved and always has the value 0.   |
| 7–5<br>CLKOUTSEL  | CLKOUT select<br><br>Selects the clock to output on the CLKOUT pin.<br><br>000 Reserved<br>001 Reserved<br>010 Bus clock<br>011 LPO clock (1 kHz)<br>100 MCGIRCLK<br>101 Reserved<br>110 OSCERCLK<br>111 Reserved |
| 4<br>RTCCLKOUTSEL | RTC clock out select  |

*Table continues on the next page...*

**SIM\_SOPT2 field descriptions (continued)**

| Field           | Description  |
|-----------------|--|
|                 | Selects either the RTC 1 Hz clock or the OSC clock to be output on the RTC_CLKOUT pin.<br>0 RTC 1 Hz clock is output on the RTC_CLKOUT pin.<br>1 OSCERCLK clock is output on the RTC_CLKOUT pin. |
| 3–0<br>Reserved | This field is reserved.<br>This read-only field is reserved and always has the value 0.  |

**12.2.4 System Options Register 4 (SIM\_SOPT4)**

Address: 4004\_7000h base + 100Ch offset = 4004\_800Ch

| Bit   | 31 | 30 | 29 | 28 | 27 | 26         | 25         | 24         | 23 | 22 | 21 | 20 | 19         | 18 | 17         | 16 |
|-------|----|----|----|----|----|------------|------------|------------|----|----|----|----|------------|----|------------|----|
| R     |    |    |    | 0  |    | TPM2CLKSEL | TPM1CLKSEL | TPMOCLKSEL |    |    | 0  |    | TPM2CH0SRC | 0  | TPM1CH0SRC | 0  |
| W     |    |    |    |    |    |            |            |            |    |    |    |    |            |    |            |    |
| Reset | 0  | 0  | 0  | 0  | 0  | 0          | 0          | 0          | 0  | 0  | 0  | 0  | 0          | 0  | 0          | 0  |
| Bit   | 15 | 14 | 13 | 12 | 11 | 10         | 9          | 8          | 7  | 6  | 5  | 4  | 3          | 2  | 1          | 0  |
| R     |    |    |    |    |    |            |            | 0          |    |    |    |    |            |    |            |    |
| W     |    |    |    |    |    |            |            |            |    |    |    |    |            |    |            |    |
| Reset | 0  | 0  | 0  | 0  | 0  | 0          | 0          | 0          | 0  | 0  | 0  | 0  | 0          | 0  | 0          | 0  |

**SIM\_SOPT4 field descriptions**

| Field             | Description  |
|-------------------|--|
| 31–27<br>Reserved | This field is reserved.<br>This read-only field is reserved and always has the value 0.  |
| 26<br>TPM2CLKSEL  | TPM2 External Clock Pin Select<br>Selects the external pin used to drive the clock to the TPM2 module.<br><b>NOTE:</b> The selected pin must also be configured for the TPM external clock function through the appropriate pin control register in the port control module.<br>0 TPM2 external clock driven by TPM_CLKIN0 pin.<br>1 TPM2 external clock driven by TPM_CLKIN1 pin. |
| 25<br>TPM1CLKSEL  | TPM1 External Clock Pin Select<br>Selects the external pin used to drive the clock to the TPM1 module.<br><b>NOTE:</b> The selected pin must also be configured for the TPM external clock function through the appropriate pin control register in the port control module.   |

*Table continues on the next page...*

**SIM\_SOPT4 field descriptions (continued)**

| Field             | Description   |
|-------------------|---|
|                   | <p>0 TPM1 external clock driven by TPM_CLKIN0 pin.<br/>1 TPM1 external clock driven by TPM_CLKIN1 pin.</p>  |
| 24<br>TPM0CLKSEL  | <p>TPM0 External Clock Pin Select<br/>Selects the external pin used to drive the clock to the TPM0 module.</p> <p><b>NOTE:</b> The selected pin must also be configured for the TPM external clock function through the appropriate pin control register in the port control module.</p> <p>0 TPM0 external clock driven by TPM_CLKIN0 pin.<br/>1 TPM0 external clock driven by TPM_CLKIN1 pin.</p> |
| 23–21<br>Reserved | <p>This field is reserved.<br/>This read-only field is reserved and always has the value 0.</p>   |
| 20<br>TPM2CH0SRC  | <p>TPM2 channel 0 input capture source select<br/>Selects the source for TPM2 channel 0 input capture.</p> <p><b>NOTE:</b> When TPM2 is not in input capture mode, clear this field.</p> <p>0 TPM2_CH0 signal<br/>1 CMP0 output</p>   |
| 19<br>Reserved    | <p>This field is reserved.<br/>This read-only field is reserved and always has the value 0.</p>   |
| 18<br>TPM1CH0SRC  | <p>TPM1 channel 0 input capture source select<br/>Selects the source for TPM1 channel 0 input capture.</p> <p><b>NOTE:</b> When TPM1 is not in input capture mode, clear this field.</p> <p>0 TPM1_CH0 signal<br/>1 CMP0 output</p>   |
| 17–0<br>Reserved  | <p>This field is reserved.<br/>This read-only field is reserved and always has the value 0.</p>   |

## 12.2.5 System Options Register 5 (SIM\_SOPT5)

Address: 4004\_7000h base + 1010h offset = 4004\_8010h

| Bit   | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 | 23 | 22         | 21 | 20         | 19 | 18 | 17         | 16         |
|-------|----|----|----|----|----|----|----|----|----|------------|----|------------|----|----|------------|------------|
| R     |    |    |    |    |    |    |    | 0  |    |            |    |            | 0  |    | UART2ODE   |            |
| W     |    |    |    |    |    |    |    |    |    |            |    |            |    |    | UART1ODE   | UART0ODE   |
| Reset | 0  | 0  | 0  | 0  | 0  | 0  | 0  | 0  | 0  | 0          | 0  | 0          | 0  | 0  | 0          | 0          |
| Bit   | 15 | 14 | 13 | 12 | 11 | 10 | 9  | 8  | 7  | 6          | 5  | 4          | 3  | 2  | 1          | 0          |
| R     |    |    |    |    |    |    |    | 0  |    | UART1RXSRC |    | UART1TXSRC | 0  |    | UART0RXSRC | UART0TXSRC |
| W     |    |    |    |    |    |    |    |    |    |            |    |            |    |    |            |            |
| Reset | 0  | 0  | 0  | 0  | 0  | 0  | 0  | 0  | 0  | 0          | 0  | 0          | 0  | 0  | 0          | 0          |

### SIM\_SOPT5 field descriptions

| Field             | Description   |
|-------------------|---|
| 31–20<br>Reserved | This field is reserved.<br>This read-only field is reserved and always has the value 0.                               |
| 19<br>Reserved    | This field is reserved.<br>This read-only field is reserved and always has the value 0.                               |
| 18<br>UART2ODE    | UART2 Open Drain Enable<br>0 Open drain is disabled on UART2<br>1 Open drain is enabled on UART2                      |
| 17<br>UART1ODE    | UART1 Open Drain Enable<br>0 Open drain is disabled on UART1<br>1 Open drain is enabled on UART1                      |
| 16<br>UART0ODE    | UART0 Open Drain Enable<br>0 Open drain is disabled on UART0<br>1 Open drain is enabled on UART0                      |
| 15–7<br>Reserved  | This field is reserved.<br>This read-only field is reserved and always has the value 0.                               |
| 6<br>UART1RXSRC   | UART1 receive data source select<br>Selects the source for the UART1 receive data.<br>0 UART1_RX pin<br>1 CMP0 output |
| 5–4<br>UART1TXSRC | UART1 transmit data source select<br>Selects the source for the UART1 transmit data.                                  |

Table continues on the next page...

**SIM\_SOPT5 field descriptions (continued)**

| Field             | Description  |
|-------------------|--|
|                   | 00 UART1_TX pin<br>01 UART1_TX pin modulated with TPM1 channel 0 output<br>10 UART1_TX pin modulated with TPM2 channel 0 output<br>11 Reserved   |
| 3<br>Reserved     | This field is reserved.<br>This read-only field is reserved and always has the value 0.  |
| 2<br>UART0RXSRC   | UART0 receive data source select<br><br>Selects the source for the UART0 receive data.<br><br>0 UART0_RX pin<br>1 CMP0 output  |
| 1–0<br>UART0TXSRC | UART0 transmit data source select<br><br>Selects the source for the UART0 transmit data.<br><br>00 UART0_TX pin<br>01 UART0_TX pin modulated with TPM1 channel 0 output<br>10 UART0_TX pin modulated with TPM2 channel 0 output<br>11 Reserved |

**12.2.6 System Options Register 7 (SIM\_SOPT7)**

Address: 4004\_7000h base + 1018h offset = 4004\_8018h

| Bit   | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 | 23            | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|-------|----|----|----|----|----|----|----|----|---------------|----|----|----|----|----|----|----|
| R     |    |    |    |    |    |    |    |    | 0             |    |    |    |    |    |    |    |
| W     |    |    |    |    |    |    |    |    |               |    |    |    |    |    |    |    |
| Reset | 0  | 0  | 0  | 0  | 0  | 0  | 0  | 0  | 0             | 0  | 0  | 0  | 0  | 0  | 0  | 0  |
| Bit   | 15 | 14 | 13 | 12 | 11 | 10 | 9  | 8  | 7             | 6  | 5  | 4  | 3  | 2  | 1  | 0  |
| R     |    |    |    |    |    |    |    |    | 0             |    | 0  | 0  | 0  | 0  | 0  | 0  |
| W     |    |    |    |    |    |    |    |    | ADC0ALTTRGE_N |    | 0  | 0  | 0  | 0  | 0  | 0  |
| Reset | 0  | 0  | 0  | 0  | 0  | 0  | 0  | 0  | 0             | 0  | 0  | 0  | 0  | 0  | 0  | 0  |

**SIM\_SOPT7 field descriptions**

| Field              | Description   |
|--------------------|---|
| 31–8<br>Reserved   | This field is reserved.<br>This read-only field is reserved and always has the value 0.   |
| 7<br>ADC0ALTTRGEN  | ADC0 alternate trigger enable<br><br>Enable alternative conversion triggers for ADC0.<br><br>0 TPM1 channel 0 (A) and channel 1 (B) triggers selected for ADC0.<br>1 Alternate trigger selected for ADC0.   |
| 6–5<br>Reserved    | This field is reserved.<br>This read-only field is reserved and always has the value 0.   |
| 4<br>ADC0PRETRGSEL | ADC0 pretrigger select<br><br>Selects the ADC0 pre-trigger source when alternative triggers are enabled through ADC0ALTTRGEN.<br><br>0 Pre-trigger A<br>1 Pre-trigger B   |
| 3–0<br>ADC0TRGSEL  | ADC0 trigger select<br><br>Selects the ADC0 trigger source when alternative triggers are functional in stop and VLPS modes..<br><br>0000 External trigger pin input (EXTRG_IN)<br>0001 CMP0 output<br>0010 Reserved<br>0011 Reserved<br>0100 PIT trigger 0<br>0101 PIT trigger 1<br>0110 Reserved<br>0111 Reserved<br>1000 TPM0 overflow<br>1001 TPM1 overflow<br>1010 TPM2 overflow<br>1011 Reserved<br>1100 RTC alarm<br>1101 RTC seconds<br>1110 LPTMR0 trigger<br>1111 Reserved |

## 12.2.7 System Device Identification Register (SIM\_SDID)

Address: 4004\_7000h base + 1024h offset = 4004\_8024h

| Bit   | 31 | 30    | 29 | 28       | 27 | 26       | 25 | 24       | 23 | 22    | 21 | 20    | 19 | 18 | 17 | 16    | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|-------|----|-------|----|----------|----|----------|----|----------|----|-------|----|-------|----|----|----|-------|----|----|----|----|----|----|---|---|---|---|---|---|---|---|---|---|
| R     |    | FAMID |    | SUBFAMID |    | SERIESID |    | SRAMSIZE |    | REVID |    | DIEID |    | 0  |    | PINID |    |    |    |    |    |    |   |   |   |   |   |   |   |   |   |   |
| W     |    |       |    |          |    |          |    |          |    |       |    |       |    |    |    |       |    |    |    |    |    |    |   |   |   |   |   |   |   |   |   |   |
| Reset | *  | *     | *  | *        | *  | *        | *  | *        | 0  | 0     | 0  | 1     | *  | *  | *  | *     | *  | *  | *  | 0  | 1  | 0  | 0 | 1 | 0 | 0 | 0 | * | * | * | * |   |

\* Notes:

- FAMID field: Device specific value.
- SUBFAMID field: Device specific value.
- SRAMSIZE field: Device specific value.
- REVID field: Device specific value.
- PINID field: Device specific value.

### SIM\_SDID field descriptions

| Field             | Description  |
|-------------------|--|
| 31–28<br>FAMID    | Kinetis family ID<br><br>Specifies the Kinetis family of the device.<br><br>0000 KL0x Family (low end)<br>0001 KL1x Family (basic)<br>0010 KL2x Family (USB)<br>0011 KL3x Family (Segment LCD)<br>0100 KL4x Family (USB and Segment LCD)           |
| 27–24<br>SUBFAMID | Kinetis Sub-Family ID<br><br>Specifies the Kinetis sub-family of the device.<br><br>0010 KLx2 Subfamily (low end)<br>0100 KLx4 Subfamily (basic analog)<br>0101 KLx5 Subfamily (advanced analog)<br>0110 KLx6 Subfamily (advanced analog with I2S) |
| 23–20<br>SERIESID | Kinetis Series ID<br><br>Specifies the Kinetis family of the device.<br><br>0001 KL family   |
| 19–16<br>SRAMSIZE | System SRAM Size<br><br>Specifies the size of the System SRAM<br><br>0000 0.5 KB<br>0001 1 KB<br>0010 2 KB<br>0011 4 KB<br>0100 8 KB<br>0101 16 KB   |

Table continues on the next page...

**SIM\_SDID field descriptions (continued)**

| Field           | Description   |
|-----------------|---|
|                 | 0110 32 KB<br>0111 64 KB  |
| 15–12<br>REVID  | Device revision number<br><br>Specifies the silicon implementation number for the device.   |
| 11–7<br>DIEID   | Device die number<br><br>Specifies the silicon implementation number for the device.  |
| 6–4<br>Reserved | This field is reserved.<br>This read-only field is reserved and always has the value 0.   |
| 3–0<br>PINID    | Pincount identification<br><br>Specifies the pincount of the device.<br><br>0000 16-pin<br>0001 24-pin<br>0010 32-pin<br>0011 Reserved<br>0100 48-pin<br>0101 64-pin<br>0110 80-pin<br>0111 Reserved<br>1000 100-pin<br>1001 Reserved<br>1010 Reserved<br>1011 Reserved<br>1100 Reserved<br>1101 Reserved<br>1110 Reserved<br>1111 Reserved |

## 12.2.8 System Clock Gating Control Register 4 (SIM\_SCGC4)

Address: 4004\_7000h base + 1034h offset = 4004\_8034h

| Bit   | 31 | 30 | 29 | 28    | 27    | 26    | 25 | 24 | 23   | 22   | 21 | 20 | 19     | 18 | 17 | 16 |
|-------|----|----|----|-------|-------|-------|----|----|------|------|----|----|--------|----|----|----|
| R     |    |    |    | 1     |       |       | 0  |    |      |      | 0  |    | CMP    |    | 0  |    |
| W     |    |    |    |       |       |       |    |    | SPI1 | SPI0 |    |    | USBOTG |    |    |    |
| Reset | 1  | 1  | 1  | 1     | 0     | 0     | 0  | 0  | 0    | 0    | 0  | 0  | 0      | 0  | 0  | 0  |
| Bit   | 15 | 14 | 13 | 12    | 11    | 10    | 9  | 8  | 7    | 6    | 5  | 4  | 3      | 2  | 1  | 0  |
| R     | 0  | 0  | 0  | UART2 | UART1 | UART0 | 0  | 0  | I2C1 | I2C0 | 1  |    | 0      |    |    |    |
| W     |    |    |    |       |       |       |    |    | 0    | 0    | 1  | 1  | 0      | 0  | 0  | 0  |
| Reset | 0  | 0  | 0  | 0     | 0     | 0     | 0  | 0  | 0    | 0    | 1  | 1  | 0      | 0  | 0  | 0  |

**SIM\_SCGC4 field descriptions**

| Field             | Description  |
|-------------------|--|
| 31–28<br>Reserved | This field is reserved.<br>This read-only field is reserved and always has the value 1.  |
| 27–24<br>Reserved | This field is reserved.<br>This read-only field is reserved and always has the value 0.  |
| 23<br>SPI1        | SPI1 Clock Gate Control<br><br>This bit controls the clock gate to the SPI1 module.<br><br>0 Clock disabled<br>1 Clock enabled             |
| 22<br>SPI0        | SPI0 Clock Gate Control<br><br>This bit controls the clock gate to the SPI0 module.<br><br>0 Clock disabled<br>1 Clock enabled             |
| 21–20<br>Reserved | This field is reserved.<br>This read-only field is reserved and always has the value 0.  |
| 19<br>CMP         | Comparator Clock Gate Control<br><br>This bit controls the clock gate to the comparator module.<br><br>0 Clock disabled<br>1 Clock enabled |
| 18<br>USBOTG      | USB Clock Gate Control<br><br>This bit controls the clock gate to the USB module.  |

*Table continues on the next page...*

**SIM\_SCGC4 field descriptions (continued)**

| Field             | Description   |
|-------------------|---|
|                   | <p>0 Clock disabled<br/>1 Clock enabled</p>   |
| 17–14<br>Reserved | <p>This field is reserved.<br/>This read-only field is reserved and always has the value 0.</p>   |
| 13<br>Reserved    | <p>This field is reserved.<br/>This read-only field is reserved and always has the value 0.</p>   |
| 12<br>UART2       | <p>UART2 Clock Gate Control<br/><br/>This bit controls the clock gate to the UART2 module.<br/><br/>0 Clock disabled<br/>1 Clock enabled</p>          |
| 11<br>UART1       | <p>UART1 Clock Gate Control<br/><br/>This bit controls the clock gate to the UART1 module.<br/><br/>0 Clock disabled<br/>1 Clock enabled</p>          |
| 10<br>UART0       | <p>UART0 Clock Gate Control<br/><br/>This bit controls the clock gate to the UART0 module.<br/><br/>0 Clock disabled<br/>1 Clock enabled</p>          |
| 9–8<br>Reserved   | <p>This field is reserved.<br/>This read-only field is reserved and always has the value 0.</p>   |
| 7<br>I2C1         | <p>I2C1 Clock Gate Control<br/><br/>This bit controls the clock gate to the I<sup>2</sup>C1 module.<br/><br/>0 Clock disabled<br/>1 Clock enabled</p> |
| 6<br>I2C0         | <p>I2C0 Clock Gate Control<br/><br/>This bit controls the clock gate to the I<sup>2</sup>C0 module.<br/><br/>0 Clock disabled<br/>1 Clock enabled</p> |
| 5–4<br>Reserved   | <p>This field is reserved.<br/>This read-only field is reserved and always has the value 1.</p>   |
| 3–0<br>Reserved   | <p>This field is reserved.<br/>This read-only field is reserved and always has the value 0.</p>   |

## 12.2.9 System Clock Gating Control Register 5 (SIM\_SCGC5)

Address: 4004\_7000h base + 1038h offset = 4004\_8038h

| Bit   | 31 | 30 | 29    | 28    | 27    | 26    | 25    | 24 | 23 | 22 | 21  | 20 | 19 | 18 | 17 | 16    |
|-------|----|----|-------|-------|-------|-------|-------|----|----|----|-----|----|----|----|----|-------|
| R     |    |    |       |       |       |       |       | 0  |    |    |     | 0  |    | 0  |    |       |
| W     |    |    |       |       |       |       |       |    |    |    |     |    |    |    |    |       |
| Reset | 0  | 0  | 0     | 0     | 0     | 0     | 0     | 0  | 0  | 0  | 0   | 0  | 0  | 0  | 0  | 0     |
| Bit   | 15 | 14 | 13    | 12    | 11    | 10    | 9     | 8  | 7  | 6  | 5   | 4  | 3  | 2  | 1  | 0     |
| R     | 0  |    | PORTE | PORTD | PORTC | PORTB | PORTA |    | 1  | 0  | TSI |    | 0  |    | 0  | LPTMR |
| W     |    |    |       |       |       |       |       |    |    |    |     |    |    |    |    |       |
| Reset | 0  | 0  | 0     | 0     | 0     | 0     | 0     | 1  | 1  | 0  | 0   | 0  | 0  | 0  | 0  | 0     |

### SIM\_SCGC5 field descriptions

| Field             | Description  |
|-------------------|--|
| 31–20<br>Reserved | This field is reserved.<br>This read-only field is reserved and always has the value 0.  |
| 19<br>Reserved    | This field is reserved.<br>This read-only field is reserved and always has the value 0.  |
| 18–14<br>Reserved | This field is reserved.<br>This read-only field is reserved and always has the value 0.  |
| 13<br>PORTE       | Port E Clock Gate Control<br><br>This bit controls the clock gate to the Port E module.<br><br>0 Clock disabled<br>1 Clock enabled |
| 12<br>PORTD       | Port D Clock Gate Control<br><br>This bit controls the clock gate to the Port D module.<br><br>0 Clock disabled<br>1 Clock enabled |
| 11<br>PORTC       | Port C Clock Gate Control<br><br>This bit controls the clock gate to the Port C module.<br><br>0 Clock disabled<br>1 Clock enabled |
| 10<br>PORTB       | Port B Clock Gate Control<br><br>This bit controls the clock gate to the Port B module.  |

*Table continues on the next page...*

**SIM\_SCGC5 field descriptions (continued)**

| Field        | Description   |
|--------------|---|
|              | 0 Clock disabled<br>1 Clock enabled   |
| 9 PORTA      | Port A Clock Gate Control<br><br>This bit controls the clock gate to the Port A module.<br><br>0 Clock disabled<br>1 Clock enabled                  |
| 8–7 Reserved | This field is reserved.<br>This read-only field is reserved and always has the value 1.   |
| 6 Reserved   | This field is reserved.<br>This read-only field is reserved and always has the value 0.   |
| 5 TSI        | TSI Access Control<br><br>This bit controls software access to the TSI module.<br><br>0 Access disabled<br>1 Access enabled                         |
| 4–2 Reserved | This field is reserved.<br>This read-only field is reserved and always has the value 0.   |
| 1 Reserved   | This field is reserved.<br>This read-only field is reserved and always has the value 0.   |
| 0 LPTMR      | Low Power Timer Access Control<br><br>This bit controls software access to the Low Power Timer module.<br><br>0 Access disabled<br>1 Access enabled |

**12.2.10 System Clock Gating Control Register 6 (SIM\_SCGC6)**

Address: 4004\_7000h base + 103Ch offset = 4004\_803Ch

| Bit   | 31   | 30 | 29  | 28 | 27   | 26   | 25   | 24   | 23  | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|-------|------|----|-----|----|------|------|------|------|-----|----|----|----|----|----|----|----|
| R     | DAC0 | 0  | RTC | 0  | ADC0 | TPM2 | TPM1 | TPMO | PIT |    |    |    |    | 0  |    |    |
| W     |      |    |     |    |      |      |      |      |     |    |    |    |    |    |    |    |
| Reset | 0    | 0  | 0   | 0  | 0    | 0    | 0    | 0    | 0   | 0  | 0  | 0  | 0  | 0  | 0  | 0  |
| Bit   | 15   | 14 | 13  | 12 | 11   | 10   | 9    | 8    | 7   | 6  | 5  | 4  | 3  | 2  | 1  | 0  |
| R     | 0    |    |     |    |      |      |      | 0    |     |    |    |    |    |    |    |    |
| W     |      |    |     |    |      |      |      |      |     |    |    |    |    |    |    |    |
| Reset | 0    | 0  | 0   | 0  | 0    | 0    | 0    | 0    | 0   | 0  | 0  | 0  | 0  | 0  | 0  | 1  |

**SIM\_SCGC6 field descriptions**

| Field             | Description  |
|-------------------|--|
| 31<br>DAC0        | DAC0 Clock Gate Control<br><br>This bit controls the clock gate to the DAC0 module.<br><br>0 Clock disabled<br>1 Clock enabled   |
| 30<br>Reserved    | This field is reserved.<br>This read-only field is reserved and always has the value 0.  |
| 29<br>RTC         | RTC Access Control<br><br>This bit controls software access and interrupts to the RTC module.<br><br>0 Access and interrupts disabled<br>1 Access and interrupts enabled |
| 28<br>Reserved    | This field is reserved.<br>This read-only field is reserved and always has the value 0.  |
| 27<br>ADC0        | ADC0 Clock Gate Control<br><br>This bit controls the clock gate to the ADC0 module.<br><br>0 Clock disabled<br>1 Clock enabled   |
| 26<br>TPM2        | TPM2 Clock Gate Control<br><br>This bit controls the clock gate to the TPM2 module.<br><br>0 Clock disabled<br>1 Clock enabled   |
| 25<br>TPM1        | TPM1 Clock Gate Control<br><br>This bit controls the clock gate to the TPM1 module.<br><br>0 Clock disabled<br>1 Clock enabled   |
| 24<br>TPM0        | TPM0 Clock Gate Control<br><br>This bit controls the clock gate to the TPM0 module.<br><br>0 Clock disabled<br>1 Clock enabled   |
| 23<br>PIT         | PIT Clock Gate Control<br><br>This bit controls the clock gate to the PIT module.<br><br>0 Clock disabled<br>1 Clock enabled   |
| 22–16<br>Reserved | This field is reserved.<br>This read-only field is reserved and always has the value 0.  |
| 15<br>Reserved    | This field is reserved.<br>This read-only field is reserved and always has the value 0.  |

*Table continues on the next page...*

**SIM\_SCGC6 field descriptions (continued)**

| Field            | Description   |
|------------------|---|
| 14–2<br>Reserved | This field is reserved.<br>This read-only field is reserved and always has the value 0.   |
| 1<br>DMAMUX      | DMA Mux Clock Gate Control<br><br>This bit controls the clock gate to the DMA Mux module.<br><br>0 Clock disabled<br>1 Clock enabled  |
| 0<br>FTF         | Flash Memory Clock Gate Control<br><br>This bit controls the clock gate to the flash memory. Flash reads are still supported while the flash memory is clock gated, but entry into low power modes is blocked.<br><br>0 Clock disabled<br>1 Clock enabled |

**12.2.11 System Clock Gating Control Register 7 (SIM\_SCGC7)**

Address: 4004\_7000h base + 1040h offset = 4004\_8040h

|       |    |    |    |    |    |    |    |    |   |     |    |    |    |    |    |    |    |
|-------|----|----|----|----|----|----|----|----|---|-----|----|----|----|----|----|----|----|
| Bit   | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |   | 23  | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
| R     |    |    |    |    |    |    |    |    | 0 |     |    |    |    |    |    |    |    |
| W     |    |    |    |    |    |    |    |    |   |     |    |    |    |    |    |    |    |
| Reset | 0  | 0  | 0  | 0  | 0  | 0  | 0  | 0  |   | 0   | 0  | 0  | 0  | 0  | 0  | 0  | 0  |
| Bit   | 15 | 14 | 13 | 12 | 11 | 10 | 9  | 8  |   | 7   | 6  | 5  | 4  | 3  | 2  | 1  | 0  |
| R     |    |    |    |    |    |    |    | 0  |   |     |    |    |    | 0  |    |    |    |
| W     |    |    |    |    |    |    |    |    |   | DMA |    |    |    |    |    |    |    |
| Reset | 0  | 0  | 0  | 0  | 0  | 0  | 0  | 1  |   | 0   | 0  | 0  | 0  | 0  | 0  | 0  | 0  |

**SIM\_SCGC7 field descriptions**

| Field            | Description  |
|------------------|--|
| 31–9<br>Reserved | This field is reserved.<br>This read-only field is reserved and always has the value 0.                                      |
| 8<br>DMA         | DMA Clock Gate Control<br><br>This bit controls the clock gate to the DMA module.<br><br>0 Clock disabled<br>1 Clock enabled |
| 7–0<br>Reserved  | This field is reserved.<br>This read-only field is reserved and always has the value 0.                                      |

## 12.2.12 System Clock Divider Register 1 (SIM\_CLKDIV1)

### NOTE

The CLKDIV1 register cannot be written to when the device is in VLPR mode.

### NOTE

Reset value loaded during System Reset from FTF\_FOPT[LPBOOT].

Address: 4004\_7000h base + 1044h offset = 4004\_8044h

| Bit | 31      | 30 | 29 | 28 | 27 | 26 | 25 | 24 | 23      | 22 | 21 | 20 | 19 | 18 | 17 | 16 | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |   |   |
|-----|---------|----|----|----|----|----|----|----|---------|----|----|----|----|----|----|----|----|----|----|----|----|----|---|---|---|---|---|---|---|---|---|---|---|---|
| R   | OUTDIV1 |    |    |    | 0  |    |    |    | OUTDIV4 |    |    |    | 0  |    |    |    |    |    |    |    |    |    |   |   |   |   |   |   |   |   |   |   |   |   |
| W   | *       | *  | *  | *  | *  | 0  | 0  | 0  | 0       | 0  | 0  | 0  | 0  | 0  | 0  | 1  | 0  | 0  | 0  | 0  | 0  | 0  | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

\* Notes:

- OUTDIV1 field: The reset value depends on the FTF\_FOPT[LPBOOT]. it is loaded with 0000 (divide by one), 0001 (divide by two), 0011 (divide by four), or 0111 (divide by eight).

### SIM\_CLKDIV1 field descriptions

| Field             | Description   |
|-------------------|---|
| 31–28<br>OUTDIV1  | <p>Clock 1 output divider value</p> <p>This field sets the divide value for the core/system clock, as well as the bus/flash clocks. At the end of reset, it is loaded with 0000 (divide by one), 0001 (divide by two), 0011 (divide by four), or 0111 (divide by eight) depending on the setting of the two FTF_FOPT[LPBOOT] configuration bits.</p> <p>0000 Divide-by-1.<br/>     0001 Divide-by-2.<br/>     0010 Divide-by-3.<br/>     0011 Divide-by-4.<br/>     0100 Divide-by-5.<br/>     0101 Divide-by-6.<br/>     0110 Divide-by-7.<br/>     0111 Divide-by-8.<br/>     1000 Divide-by-9.<br/>     1001 Divide-by-10.<br/>     1010 Divide-by-11.<br/>     1011 Divide-by-12.<br/>     1100 Divide-by-13.<br/>     1101 Divide-by-14.<br/>     1110 Divide-by-15.<br/>     1111 Divide-by-16.</p> |
| 27–19<br>Reserved | <p>This field is reserved.</p> <p>This read-only field is reserved and always has the value 0.</p>  |

Table continues on the next page...

**SIM\_CLKDIV1 field descriptions (continued)**

| Field            | Description   |
|------------------|---|
| 18–16<br>OUTDIV4 | <p>Clock 4 output divider value</p> <p>This field sets the divide value for the bus and flash clock and is in addition to the System clock divide ratio. At the end of reset, it is loaded with 0001 (divide by two).</p> <p>000 Divide-by-1.<br/>     001 Divide-by-2.<br/>     010 Divide-by-3.<br/>     011 Divide-by-4.<br/>     100 Divide-by-5.<br/>     101 Divide-by-6.<br/>     110 Divide-by-7.<br/>     111 Divide-by-8.</p> |
| 15–0<br>Reserved | <p>This field is reserved.</p> <p>This read-only field is reserved and always has the value 0.</p>  |

**12.2.13 Flash Configuration Register 1 (SIM\_FCFG1)**

Address: 4004\_7000h base + 104Ch offset = 4004\_804Ch

| Bit   | 31 | 30 | 29 | 28 | 27 | 26 | 25     | 24 | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|-------|----|----|----|----|----|----|--------|----|----|----|----|----|----|----|----|----|
| R     |    |    |    | 0  |    |    | PFSIZE |    |    |    |    |    | 0  |    |    |    |
| W     |    |    |    |    |    |    |        |    |    |    |    |    |    |    |    |    |
| Reset | 0  | 0  | 0  | 0  | *  | *  | *      | *  | 0  | 0  | 0  | 0  | 0  | 0  | 0  | 0  |
| Bit   | 15 | 14 | 13 | 12 | 11 | 10 | 9      | 8  | 7  | 6  | 5  | 4  | 3  | 2  | 1  | 0  |
| R     |    |    |    |    |    |    |        | 0  |    |    |    |    |    |    |    |    |
| W     |    |    |    |    |    |    |        |    |    |    |    |    |    |    |    |    |
| Reset | 0  | 0  | 0  | 0  | 0  | 0  | 0      | 0  | 0  | 0  | 0  | 0  | 0  | 0  | 0  | 0  |

\* Notes:

- PFSIZE field: Device specific value.

**SIM\_FCFG1 field descriptions**

| Field             | Description   |
|-------------------|---|
| 31–28<br>Reserved | This field is reserved.<br>This read-only field is reserved and always has the value 0.   |
| 27–24<br>PFSIZE   | Program flash size<br><br>This field specifies the amount of program flash memory available on the device . Undefined values are reserved.<br><br>0000 8 KB of program flash memory, 0.25 KB protection region<br>0001 16 KB of program flash memory, 0.5 KB protection region<br>0011 32 KB of program flash memory, 1 KB protection region<br>0101 64 KB of program flash memory, 2 KB protection region<br>0111 128 KB of program flash memory, 4 KB protection region<br>1001 256 KB of program flash memory, 8 KB protection region<br>1111 128 KB of program flash memory, 4 KB protection region |
| 23–2<br>Reserved  | This field is reserved.<br>This read-only field is reserved and always has the value 0.   |
| 1<br>FLASHDOZE    | Flash Doze<br><br>When set, Flash memory is disabled for the duration of Doze mode. This bit should be clear during VLP modes. The Flash will be automatically enabled again at the end of Doze mode so interrupt vectors do not need to be relocated out of Flash memory. The wakeup time from Doze mode is extended when this bit is set. An attempt by the DMA or other bus master to access the Flash when the Flash is disabled will result in a bus error.<br><br>0 Flash remains enabled during Doze mode<br>1 Flash is disabled for the duration of Doze mode                                   |
| 0<br>FLASHDIS     | Flash Disable<br><br>Flash accesses are disabled (and generate a bus error) and the Flash memory is placed in a low power state. This bit should not be changed during VLP modes. Relocate the interrupt vectors out of Flash memory before disabling the Flash.<br><br>0 Flash is enabled<br>1 Flash is disabled   |

## 12.2.14 Flash Configuration Register 2 (SIM\_FCFG2)

Address: 4004\_7000h base + 1050h offset = 4004\_8050h

| Bit   | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|-------|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|
| R     | 0  |    |    |    |    |    |    |    | 1  |    |    |    | 0  |    |    |    |
| W     |    |    |    |    |    |    |    |    |    |    |    |    |    |    |    |    |
| Reset | 0  | *  | *  | *  | *  | *  | *  | *  | 1  | 0  | 0  | 0  | 0  | 0  | 0  | 0  |
| Bit   | 15 | 14 | 13 | 12 | 11 | 10 | 9  | 8  | 7  | 6  | 5  | 4  | 3  | 2  | 1  | 0  |
| R     |    |    |    |    |    |    |    | 0  |    |    |    |    |    |    |    |    |
| W     |    |    |    |    |    |    |    |    |    |    |    |    |    |    |    |    |
| Reset | 0  | 0  | 0  | 0  | 0  | 0  | 0  | 0  | 0  | 0  | 0  | 0  | 0  | 0  | 0  | 0  |

\* Notes:

- MAXADDR0 field: Device specific value indicating amount of implemented flash.

### SIM\_FCFG2 field descriptions

| Field             | Description  |
|-------------------|--|
| 31<br>Reserved    | This field is reserved.<br>This read-only field is reserved and always has the value 0.  |
| 30–24<br>MAXADDR0 | Max address block<br><br>This field concatenated with leading zeros indicates the first invalid address of program flash.<br><br>For example, if MAXADDR0 = 0x10 the first invalid address of program flash is 0x0002_0000. This would be the MAXADDR0 value for a device with 128 KB program flash. |
| 23<br>Reserved    | This field is reserved.<br>This read-only field is reserved and always has the value 1.  |
| 22–16<br>Reserved | This field is reserved.<br>This read-only field is reserved and always has the value 0.  |
| 15–0<br>Reserved  | This field is reserved.<br>This read-only field is reserved and always has the value 0.  |

## 12.2.15 Unique Identification Register Mid-High (SIM\_UIDMH)

Address: 4004\_7000h base + 1058h offset = 4004\_8058h

| Bit   | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |   |
|-------|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|---|---|---|---|---|---|---|---|---|---|---|
| R     |    |    |    |    |    |    |    |    |    |    |    |    |    |    |    |    | 0  |    |    |    |    |    |   |   |   |   |   |   |   |   |   |   |   |
| W     |    |    |    |    |    |    |    |    |    |    |    |    |    |    |    |    |    |    |    |    |    |    |   |   |   |   |   |   |   |   |   |   |   |
| Reset | 0  | 0  | 0  | 0  | 0  | 0  | 0  | 0  | 0  | 0  | 0  | 0  | 0  | 0  | 0  | 0  | *  | *  | *  | *  | *  | *  | * | * | * | * | * | * | * | * | * | * | * |

\* Notes:

- UID field: Device specific value.

**SIM\_UIDMH field descriptions**

| Field             | Description   |
|-------------------|---|
| 31–16<br>Reserved | This field is reserved.<br>This read-only field is reserved and always has the value 0. |
| 15–0<br>UID       | Unique Identification<br>Unique identification for the device.                          |

**12.2.16 Unique Identification Register Mid Low (SIM\_UIDML)**

Address: 4004\_7000h base + 105Ch offset = 4004\_805Ch

| Bit   | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|-------|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|---|---|---|---|---|---|---|---|---|---|
| R     |    |    |    |    |    |    |    |    |    |    |    |    |    |    |    |    |    |    |    |    |    |    |   |   |   |   |   |   |   |   |   |   |
| W     |    |    |    |    |    |    |    |    |    |    |    |    |    |    |    |    |    |    |    |    |    |    |   |   |   |   |   |   |   |   |   |   |
| Reset | *  | *  | *  | *  | *  | *  | *  | *  | *  | *  | *  | *  | *  | *  | *  | *  | *  | *  | *  | *  | *  | *  | * | * | * | * | * | * | * | * | * |   |

\* Notes:

- UID field: Device specific value.

**SIM\_UIDML field descriptions**

| Field       | Description  |
|-------------|--|
| 31–0<br>UID | Unique Identification<br>Unique identification for the device. |

**12.2.17 Unique Identification Register Low (SIM\_UIDL)**

Address: 4004\_7000h base + 1060h offset = 4004\_8060h

| Bit   | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|-------|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|---|---|---|---|---|---|---|---|---|---|
| R     |    |    |    |    |    |    |    |    |    |    |    |    |    |    |    |    |    |    |    |    |    |    |   |   |   |   |   |   |   |   |   |   |
| W     |    |    |    |    |    |    |    |    |    |    |    |    |    |    |    |    |    |    |    |    |    |    |   |   |   |   |   |   |   |   |   |   |
| Reset | *  | *  | *  | *  | *  | *  | *  | *  | *  | *  | *  | *  | *  | *  | *  | *  | *  | *  | *  | *  | *  | *  | * | * | * | * | * | * | * | * | * |   |

\* Notes:

- UID field: Device specific value.

**SIM\_UIDL field descriptions**

| Field       | Description  |
|-------------|--|
| 31–0<br>UID | Unique Identification<br>Unique identification for the device. |

## 12.2.18 COP Control Register (SIM\_COPC)

All of the bits in this register can be written only once after a reset.

Address: 4004\_7000h base + 1100h offset = 4004\_8100h

| Bit   | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |   | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|-------|----|----|----|----|----|----|----|----|---|----|----|----|----|----|----|----|----|
| R     |    |    |    |    |    |    |    |    | 0 |    |    |    |    |    |    |    |    |
| W     |    |    |    |    |    |    |    |    |   |    |    |    |    |    |    |    |    |
| Reset | 0  | 0  | 0  | 0  | 0  | 0  | 0  | 0  | 0 | 0  | 0  | 0  | 0  | 0  | 0  | 0  | 0  |
| Bit   | 15 | 14 | 13 | 12 | 11 | 10 | 9  | 8  |   | 7  | 6  | 5  | 4  | 3  | 2  | 1  | 0  |
| R     |    |    |    |    |    |    |    |    | 0 |    |    |    |    |    |    |    |    |
| W     |    |    |    |    |    |    |    |    |   |    |    |    |    |    |    |    |    |
| Reset | 0  | 0  | 0  | 0  | 0  | 0  | 0  | 0  | 0 | 0  | 0  | 0  | 0  | 1  | 1  | 0  | 0  |

### SIM\_COPC field descriptions

| Field            | Description  |
|------------------|--|
| 31–4<br>Reserved | This field is reserved.<br>This read-only field is reserved and always has the value 0.  |
| 3–2<br>COPT      | COP Watchdog Timeout<br><br>These write-once bits select the timeout period of the COP. The COPT field along with the COPCLKS bit define the COP timeout period.<br><br>00 COP disabled<br>01 COP timeout after $2^5$ LPO cycles or $2^{13}$ bus clock cycles<br>10 COP timeout after $2^8$ LPO cycles or $2^{16}$ bus clock cycles<br>11 COP timeout after $2^{10}$ LPO cycles or $2^{18}$ bus clock cycles |
| 1<br>COPCLKS     | COP Clock Select<br><br>This write-once bit selects the clock source of the COP watchdog.<br><br>0 Internal 1 kHz clock is source to COP<br>1 Bus clock is source to COP   |
| 0<br>COPW        | COP windowed mode<br><br>Windowed mode is only supported when COP is running from the bus clock. The COP window is opened three quarters through the timeout period.<br><br>0 Normal mode<br>1 Windowed mode   |

## 12.2.19 Service COP Register (SIM\_SRVCOP)

Address: 4004\_7000h base + 1104h offset = 4004\_8104h

| Bit   | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|-------|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|---|---|---|---|---|---|---|---|---|---|
| R     |    |    |    |    |    |    |    |    |    |    |    |    |    |    |    |    |    |    |    |    |    |    |   |   |   |   |   |   |   |   |   |   |
| W     |    |    |    |    |    |    |    |    |    |    |    |    |    |    |    |    |    |    |    |    |    |    |   |   |   |   |   |   |   |   |   |   |
| Reset | 0  | 0  | 0  | 0  | 0  | 0  | 0  | 0  | 0  | 0  | 0  | 0  | 0  | 0  | 0  | 0  | 0  | 0  | 0  | 0  | 0  | 0  | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |   |   |

0

# Chapter 13

## System Mode Controller (SMC)

### 13.1 Introduction

#### NOTE

For the chip-specific implementation details of this module's instances see the chip configuration information.

The system mode controller (SMC) is responsible for sequencing the system into and out of all low power stop and run modes. Specifically, it monitors events to trigger transitions between power modes while controlling the power, clocks, and memories of the system to achieve the power consumption and functionality of that mode.

This chapter describes all the available low power modes, the sequence followed to enter/exit each mode, and the functionality available while in each of the modes.

The SMC is able to function during even the deepest low power modes.

### 13.2 Modes of operation

The ARM CPU has three primary modes of operation:

- Run
- Sleep
- Deep Sleep

The WFI or WFE instruction is used to invoke Sleep and Deep Sleep modes. Run, wait and stop are the common terms used for the primary operating modes of Freescale microcontrollers. The following table shows the translation between the ARM CPU modes and the Freescale MCU power modes.

## Modes of operation

| ARM CPU mode | MCU mode |
|--------------|----------|
| Sleep        | Wait     |
| Deep Sleep   | Stop     |

Accordingly, the ARM CPU documentation refers to sleep and deep sleep, while the Freescale MCU documentation normally uses wait and stop.

In addition, Freescale MCUs also augment stop, wait, and run modes in a number of ways. The power management controller (PMC) contains a run and a stop mode regulator. Run regulation is used in normal run, wait and stop modes. Stop mode regulation is used during all very low power and low leakage modes. During stop mode regulation, the bus frequencies are limited in the very low power modes.

The SMC provides the user with multiple power options. The Very Low Power Run (VLPR) mode can drastically reduce run time power when maximum bus frequency is not required to handle the application needs. From Normal Run mode, the Run Mode (RUNM) field can be modified to change the MCU into VLPR mode when limited frequency is sufficient for the application. From VLPR mode, a corresponding wait (VLPW) and stop (VLPS) mode can be entered.

Depending on the needs of the user application, a variety of stop modes are available that allow the state retention, partial power down or full power down of certain logic and/or memory. I/O states are held in all modes of operation. Several registers are used to configure the various modes of operation for the device.

The following table describes the power modes available for the device.

**Table 13-1. Power modes**

| Mode | Description   |
|------|---|
| RUN  | The MCU can be run at full speed and the internal supply is fully regulated, that is, in run regulation. This mode is also referred to as Normal Run mode.  |
| WAIT | The core clock is gated off. The system clock continues to operate. Bus clocks, if enabled, continue to operate. Run regulation is maintained.  |
| STOP | The core clock is gated off. System clocks to other masters and bus clocks are gated off after all stop acknowledge signals from supporting peripherals are valid.  |
| VLPR | The core, system, bus, and flash clock maximum frequencies are restricted in this mode. See the Power Management chapter for details about the maximum allowable frequencies.   |
| VLPW | The core clock is gated off. The system, bus, and flash clocks continue to operate, although their maximum frequency is restricted. See the Power Management chapter for details on the maximum allowable frequencies.  |
| VLPS | The core clock is gated off. System clocks to other masters and bus clocks are gated off after all stop acknowledge signals from supporting peripherals are valid.  |
| LLS  | The core clock is gated off. System clocks to other masters and bus clocks are gated off after all stop acknowledge signals from supporting peripherals are valid. The MCU is placed in a low leakage mode by reducing the voltage to internal logic. Internal logic states are retained. |

*Table continues on the next page...*

**Table 13-1. Power modes (continued)**

| Mode  | Description  |
|-------|--|
| VLLS3 | The core clock is gated off. System clocks to other masters and bus clocks are gated off after all stop acknowledge signals from supporting peripherals are valid. The MCU is placed in a low leakage mode by powering down the internal logic. All system RAM contents are retained and I/O states are held. Internal logic states are not retained.  |
| VLLS1 | The core clock is gated off. System clocks to other masters and bus clocks are gated off after all stop acknowledge signals from supporting peripherals are valid. The MCU is placed in a low leakage mode by powering down the internal logic and all system RAM. I/O states are held. Internal logic states are not retained.  |
| VLLS0 | The core clock is gated off. System clocks to other masters and bus clocks are gated off after all stop acknowledge signals from supporting peripherals are valid. The MCU is placed in a low leakage mode by powering down the internal logic and all system RAM. I/O states are held. Internal logic states are not retained. The 1kHz LPO clock is disabled and the power on reset (POR) circuit can be optionally enabled using STOPCTRL[PORPO]. |

## 13.3 Memory map and register descriptions

Details follow about the registers related to the system mode controller.

Different SMC registers reset on different reset types. Each register's description provides details. For more information about the types of reset on this chip, refer to the Reset section details.

### NOTE

The SMC registers can be written only in supervisor mode.

Write accesses in user mode are blocked and will result in a bus error.

### SMC memory map

| Absolute address (hex) | Register name                               | Width (in bits) | Access | Reset value | Section/ page |
|------------------------|---|-----------------|--------|-------------|---------------|
| 4007_E000              | Power Mode Protection register (SMC_PMPROT) | 8               | R/W    | 00h         | 13.3.1/219    |
| 4007_E001              | Power Mode Control register (SMC_PMCTRL)    | 8               | R/W    | 00h         | 13.3.2/221    |
| 4007_E002              | Stop Control Register (SMC_STOPCTRL)        | 8               | R/W    | 03h         | 13.3.3/222    |
| 4007_E003              | Power Mode Status register (SMC_PMSTAT)     | 8               | R      | 01h         | 13.3.4/223    |

### 13.3.1 Power Mode Protection register (SMC\_PMPROT)

This register provides protection for entry into any low-power run or stop mode. The enabling of the low-power run or stop mode occurs by configuring the Power Mode Control register (PMCTRL).

## Memory map and register descriptions

The PMPROT register can be written only once after any system reset.

If the MCU is configured for a disallowed or reserved power mode, the MCU remains in its current power mode. For example, if the MCU is in normal RUN mode and AVLP is 0, an attempt to enter VLPR mode using PMCTRL[RUNM] is blocked and the RUNM bits remain 00b, indicating the MCU is still in Normal Run mode.

### NOTE

This register is reset on Chip Reset not VLLS and by reset types that trigger Chip Reset not VLLS. It is unaffected by reset types that do not trigger Chip Reset not VLLS. See the Reset section details for more information.

Address: 4007\_E000h base + 0h offset = 4007\_E000h

| Bit   | 7 | 6 | 5    | 4 | 3    | 2 | 1     | 0 |
|-------|---|---|------|---|------|---|-------|---|
| Read  | 0 |   | AVLP | 0 | ALLS | 0 | AVLLS | 0 |
| Write |   | 0 |      | 0 | 0    | 0 | 0     | 0 |
| Reset | 0 | 0 | 0    | 0 | 0    | 0 | 0     | 0 |

### SMC\_PMPROT field descriptions

| Field           | Description   |
|-----------------|---|
| 7–6<br>Reserved | This field is reserved.<br>This read-only field is reserved and always has the value 0.   |
| 5<br>AVLP       | Allow Very-Low-Power Modes<br><br>Provided the appropriate control bits are set up in PMCTRL, this write-once bit allows the MCU to enter any very-low-power modes: VLPR, VLPW, and VLPS.<br><br>0 VLPR, VLPW and VLPS are not allowed<br>1 VLPR, VLPW and VLPS are allowed |
| 4<br>Reserved   | This field is reserved.<br>This read-only field is reserved and always has the value 0.   |
| 3<br>ALLS       | Allow Low-Leakage Stop Mode<br><br>This write once bit allows the MCU to enter any low-leakage stop mode (LLS), provided the appropriate control bits are set up in PMCTRL.<br><br>0 LLS is not allowed<br>1 LLS is allowed   |
| 2<br>Reserved   | This field is reserved.<br>This read-only field is reserved and always has the value 0.   |
| 1<br>AVLLS      | Allow Very-Low-Leakage Stop Mode<br><br>Provided the appropriate control bits are set up in PMCTRL, this write once bit allows the MCU to enter any very-low-leakage stop mode (VLLSx).<br><br>0 Any VLLSx mode is not allowed<br>1 Any VLLSx mode is allowed               |
| 0<br>Reserved   | This field is reserved.<br>This read-only field is reserved and always has the value 0.   |

### 13.3.2 Power Mode Control register (SMC\_PMCTRL)

The PMCTRL register controls entry into low-power run and stop modes, provided that the selected power mode is allowed via an appropriate setting of the protection (PMPROT) register.

#### NOTE

This register is reset on Chip POR not VLLS and by reset types that trigger Chip POR not VLLS. It is unaffected by reset types that do not trigger Chip POR not VLLS. See the Reset section details for more information.

Address: 4007\_E000h base + 1h offset = 4007\_E001h

| Bit   | 7 | 6    | 5 | 4 | 3 | 2     | 1     | 0 |
|-------|---|------|---|---|---|-------|-------|---|
| Read  | 0 | RUNM |   |   | 0 | STOPA | STOPM |   |
| Write |   |      |   |   |   |       |       |   |
| Reset | 0 | 0    | 0 | 0 | 0 | 0     | 0     | 0 |

#### SMC\_PMCTRL field descriptions

| Field         | Description   |
|---------------|---|
| 7<br>Reserved | This field is reserved.<br>This read-only field is reserved and always has the value 0.   |
| 6–5<br>RUNM   | Run Mode Control<br><br>When written, causes entry into the selected run mode. Writes to this field are blocked if the protection level has not been enabled using the PMPROT register. This field is cleared by hardware on any exit to normal RUN mode.<br><br><b>NOTE:</b> RUNM must be set to VLPR only when PMSTAT=RUN. After being written to VLPR, RUNM should not be written back to RUN until PMSTAT=VLPR.<br><br><b>NOTE:</b> RUNM must be set to RUN only when PMSTAT=VLPR. After being written to RUN, RUNM should not be written back to VLPR until PMSTAT=RUN.<br><br>00 Normal Run mode (RUN)<br>01 Reserved<br>10 Very-Low-Power Run mode (VLPR)<br>11 Reserved |
| 4<br>Reserved | This field is reserved.<br>This read-only field is reserved and always has the value 0.   |
| 3<br>STOPA    | Stop Aborted<br><br>When set, this read-only status bit indicates an interrupt or reset occurred during the previous stop mode entry sequence, preventing the system from entering that mode. This bit is cleared by hardware at the beginning of any stop mode entry sequence and is set if the sequence was aborted.<br><br>0 The previous stop mode entry was successful.<br>1 The previous stop mode entry was aborted.   |

Table continues on the next page...

**SMC\_PMCTRL field descriptions (continued)**

| Field        | Description   |     |                    |     |          |     |                            |     |                        |     |                               |     |          |     |         |     |          |
|--------------|---|-----|--------------------|-----|----------|-----|----------------------------|-----|------------------------|-----|-------------------------------|-----|----------|-----|---------|-----|----------|
| 2–0<br>STOPM | <p>Stop Mode Control</p> <p>When written, controls entry into the selected stop mode when Sleep-Now or Sleep-On-Exit mode is entered with SLEEPDEEP=1. Writes to this field are blocked if the protection level has not been enabled using the PMPROT register. After any system reset, this field is cleared by hardware on any successful write to the PMPROT register.</p> <p><b>NOTE:</b> When set to VLLSx, the VLLSM bits in the STOPCTRL register is used to further select the particular VLLS submode which will be entered.</p> <p><b>NOTE:</b> When set to STOP, the PSTOPO bits in the STOPCTRL register can be used to select a Partial Stop mode if desired.</p> <table> <tr><td>000</td><td>Normal Stop (STOP)</td></tr> <tr><td>001</td><td>Reserved</td></tr> <tr><td>010</td><td>Very-Low-Power Stop (VLPS)</td></tr> <tr><td>011</td><td>Low-Leakage Stop (LLS)</td></tr> <tr><td>100</td><td>Very-Low-Leakage Stop (VLLSx)</td></tr> <tr><td>101</td><td>Reserved</td></tr> <tr><td>110</td><td>Reseved</td></tr> <tr><td>111</td><td>Reserved</td></tr> </table> | 000 | Normal Stop (STOP) | 001 | Reserved | 010 | Very-Low-Power Stop (VLPS) | 011 | Low-Leakage Stop (LLS) | 100 | Very-Low-Leakage Stop (VLLSx) | 101 | Reserved | 110 | Reseved | 111 | Reserved |
| 000          | Normal Stop (STOP)  |     |                    |     |          |     |                            |     |                        |     |                               |     |          |     |         |     |          |
| 001          | Reserved  |     |                    |     |          |     |                            |     |                        |     |                               |     |          |     |         |     |          |
| 010          | Very-Low-Power Stop (VLPS)  |     |                    |     |          |     |                            |     |                        |     |                               |     |          |     |         |     |          |
| 011          | Low-Leakage Stop (LLS)  |     |                    |     |          |     |                            |     |                        |     |                               |     |          |     |         |     |          |
| 100          | Very-Low-Leakage Stop (VLLSx)   |     |                    |     |          |     |                            |     |                        |     |                               |     |          |     |         |     |          |
| 101          | Reserved  |     |                    |     |          |     |                            |     |                        |     |                               |     |          |     |         |     |          |
| 110          | Reseved   |     |                    |     |          |     |                            |     |                        |     |                               |     |          |     |         |     |          |
| 111          | Reserved  |     |                    |     |          |     |                            |     |                        |     |                               |     |          |     |         |     |          |

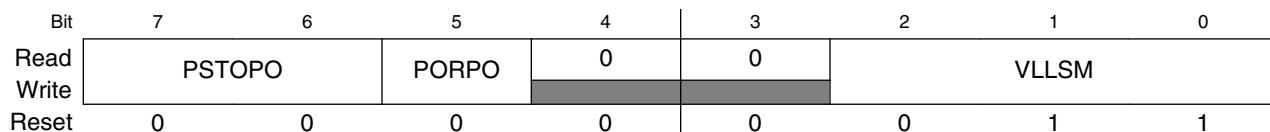
**13.3.3 Stop Control Register (SMC\_STOPCTRL)**

The STOPCTRL register provides various control bits allowing the user to fine tune power consumption during the stop mode selected by the STOPM field.

**NOTE**

This register is reset on Chip POR not VLLS and by reset types that trigger Chip POR not VLLS. It is unaffected by reset types that do not trigger Chip POR not VLLS. See the Reset section details for more information.

Address: 4007\_E000h base + 2h offset = 4007\_E002h

**SMC\_STOPCTRL field descriptions**

| Field         | Description  |
|---------------|--|
| 7–6<br>PSTOPO | <p>Partial Stop Option</p> <p>These bits control whether a Partial Stop mode is entered when STOPM=STOP. When entering a Partial Stop mode from RUN mode, the PMC, MCG and flash remain fully powered, allowing the device to wakeup almost instantaneously at the expense of higher power consumption. In PSTOP2, only system</p> |

*Table continues on the next page...*

**SMC\_STOPCTRL field descriptions (continued)**

| Field      | Description   |
|------------|---|
|            | <p>clocks are gated allowing peripherals running on bus clock to remain fully functional. In PSTOP1, both system and bus clocks are gated.</p> <p>00 STOP - Normal Stop mode<br/>     01 PSTOP1 - Partial Stop with both system and bus clocks disabled<br/>     10 PSTOP2 - Partial Stop with system clock disabled and bus clock enabled<br/>     11 Reserved</p> |
| 5 PORPO    | <p>POR Power Option</p> <p>This bit controls whether the POR detect circuit is enabled in VLLS0 mode.</p> <p>0 POR detect circuit is enabled in VLLS0<br/>     1 POR detect circuit is disabled in VLLS0</p>  |
| 4 Reserved | <p>This field is reserved.</p> <p>This read-only field is reserved and always has the value 0.</p>  |
| 3 Reserved | <p>This field is reserved.</p> <p>This read-only field is reserved and always has the value 0.</p>  |
| 2–0 VLLSM  | <p>VLLS Mode Control.</p> <p>This field controls which VLLS sub-mode to enter if STOPM=VLLS.</p> <p>000 VLLS0<br/>     001 VLLS1<br/>     010 Reserved<br/>     011 VLLS3<br/>     100 Reserved<br/>     101 Reserved<br/>     110 Reserved<br/>     111 Reserved</p>   |

**13.3.4 Power Mode Status register (SMC\_PMSTAT)**

PMSTAT is a read-only, one-hot register which indicates the current power mode of the system.

**NOTE**

This register is reset on Chip POR not VLLS and by reset types that trigger Chip POR not VLLS. It is unaffected by reset types that do not trigger Chip POR not VLLS. See the Reset section details for more information.

## Functional description

Address: 4007\_E000h base + 3h offset = 4007\_E003h

| Bit   | 7 | 6 | 5 | 4 | 3      | 2 | 1 | 0 |
|-------|---|---|---|---|--------|---|---|---|
| Read  | 0 |   |   |   | PMSTAT |   |   |   |
| Write |   |   |   |   |        |   |   |   |
| Reset | 0 | 0 | 0 | 0 | 0      | 0 | 0 | 1 |

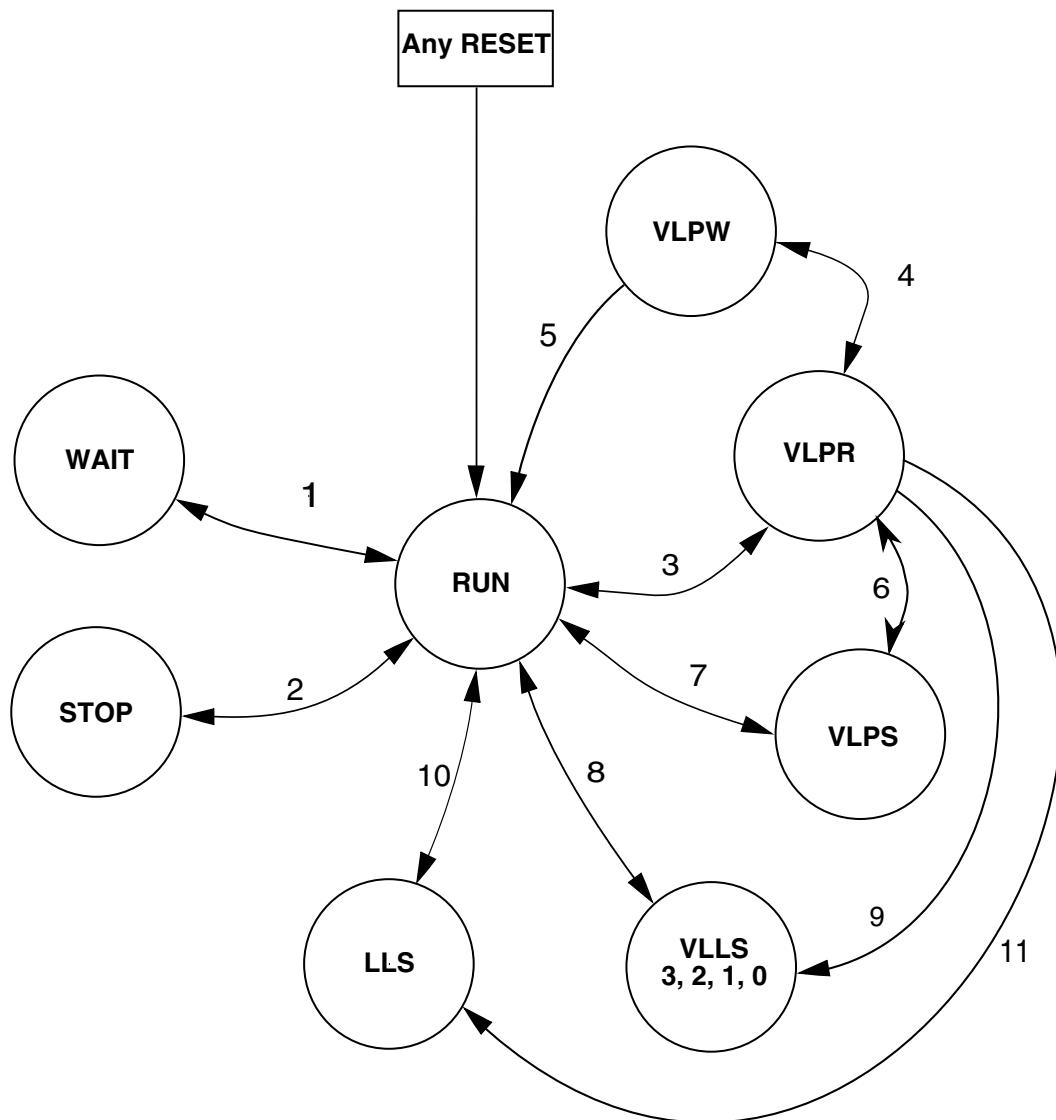
### SMC\_PMSTAT field descriptions

| Field         | Description  |
|---------------|--|
| 7<br>Reserved | This field is reserved.<br>This read-only field is reserved and always has the value 0.  |
| 6–0<br>PMSTAT | <b>NOTE:</b> When debug is enabled, the PMSTAT will not update to STOP or VLPS<br><b>NOTE:</b> When a PSTOP mode is enabled, the PMSTAT will not update to STOP or VLPS<br><br>000_0001 Current power mode is RUN<br>000_0010 Current power mode is STOP<br>000_0100 Current power mode is VLPR<br>000_1000 Current power mode is VLPW<br>001_0000 Current power mode is VLPS<br>010_0000 Current power mode is LLS<br>100_0000 Current power mode is VLLS |

## 13.4 Functional description

### 13.4.1 Power mode transitions

The following figure shows the power mode state transitions available on the chip. Any reset always brings the MCU back to the normal run state.

**Figure 13-5. Power mode state diagram**

The following table defines triggers for the various state transitions shown in the previous figure.

**Table 13-7. Power mode transition triggers**

| Transition # | From | To   | Trigger conditions  |
|--------------|------|------|---|
| 1            | RUN  | WAIT | Sleep-now or sleep-on-exit modes entered with SLEEPDEEP clear, controlled in System Control Register in ARM core.<br>See note. <sup>1</sup> |
|              | WAIT | RUN  | Interrupt or Reset  |

*Table continues on the next page...*

**Table 13-7. Power mode transition triggers (continued)**

| Transition # | From  | To    | Trigger conditions  |
|--------------|-------|-------|---|
| 2            | RUN   | STOP  | PMCTRL[RUNM]=00, PMCTRL[STOPM]=000 <sup>2</sup><br>Sleep-now or sleep-on-exit modes entered with SLEEPDEEP set, which is controlled in System Control Register in ARM core.<br>See note. <sup>1</sup> |
|              | STOP  | RUN   | Interrupt or Reset  |
| 3            | RUN   | VLPR  | The core, system, bus and flash clock frequencies are restricted in this mode. See the Power Management chapter for the maximum allowable frequencies.<br>Set PMPROT[AVLP]=1, PMCTRL[RUNM]=10.        |
|              | VLPR  | RUN   | Set PMCTRL[RUNM]=00 or Reset.   |
| 4            | VLPR  | VLPW  | Sleep-now or sleep-on-exit modes entered with SLEEPDEEP clear, which is controlled in System Control Register in ARM core.<br>See note. <sup>1</sup>  |
|              | VLPW  | VLPR  | Interrupt   |
| 5            | VLPW  | RUN   | Reset   |
| 6            | VLPR  | VLPS  | PMCTRL[STOPM]=000 <sup>3</sup> or 010,<br>Sleep-now or sleep-on-exit modes entered with SLEEPDEEP set, which is controlled in System Control Register in ARM core.<br>See note. <sup>1</sup>          |
|              | VLPS  | VLPR  | Interrupt<br><b>NOTE:</b> If VLPS was entered directly from RUN, hardware will not allow this transition and will force exit back to RUN  |
| 7            | RUN   | VLPS  | PMPROT[AVLP]=1, PMCTRL[STOPM]=010,<br>Sleep-now or sleep-on-exit modes entered with SLEEPDEEP set, which is controlled in System Control Register in ARM core.<br>See note. <sup>1</sup>              |
|              | VLPS  | RUN   | Interrupt and VLPS mode was entered directly from RUN or Reset  |
| 8            | RUN   | VLLSx | PMPROT[AVLLS]=1, PMCTRL[STOPM]=100, STOPCTRL[VLLSM]=x (VLLSx), Sleep-now or sleep-on-exit modes entered with SLEEPDEEP set, which is controlled in System Control Register in ARM core.               |
|              | VLLSx | RUN   | Wakeup from enabled LLWU input source or RESET pin  |
| 9            | VLPR  | VLLSx | PMPROT[AVLLS]=1, PMCTRL[STOPM]=100, STOPCTRL[VLLSM]=x (VLLSx), Sleep-now or sleep-on-exit modes entered with SLEEPDEEP set, which is controlled in System Control Register in ARM core.               |

*Table continues on the next page...*

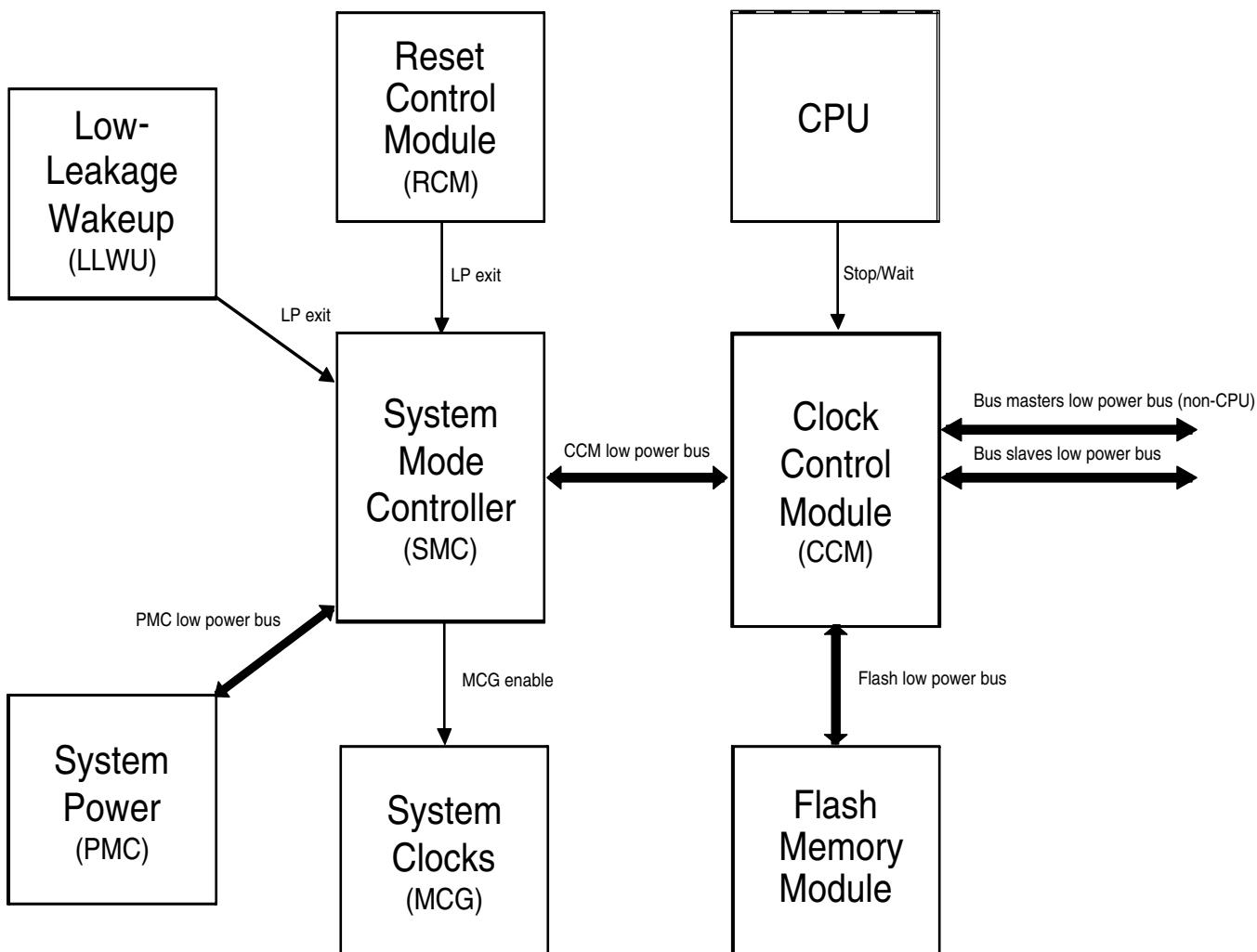
**Table 13-7. Power mode transition triggers (continued)**

| Transition # | From | To  | Trigger conditions   |
|--------------|------|-----|--|
| 10           | RUN  | LLS | PMPPROT[ALLS]=1, PMCTRL[STOPM]=011, Sleep-now or sleep-on-exit modes entered with SLEEPDEEP set, which is controlled in System Control Register in ARM core. |
|              | LLS  | RUN | Wakeup from enabled LLWU input source or RESET pin.  |
| 11           | VLPR | LLS | PMPPROT[ALLS]=1, PMCTRL[STOPM]=011, Sleep-now or sleep-on-exit modes entered with SLEEPDEEP set, which is controlled in System Control Register in ARM core. |

1. If debug is enabled, the core clock remains to support debug.
2. If PMCTRL[STOPM]=000 and STOPCTRL[PSTOPO]=01 or 10, then only a Partial Stop mode is entered instead of STOP
3. If PMCTRL[STOPM]=000 and STOPCTRL[PSTOPO]=00, then VLPS mode is entered instead of STOP. If PMCTRL[STOPM]=000 and STOPCTRL[PSTOPO]=01 or 10, then only a Partial Stop mode is entered instead of VLPS

### 13.4.2 Power mode entry/exit sequencing

When entering or exiting low-power modes, the system must conform to an orderly sequence to manage transitions safely. The SMC manages the system's entry into and exit from all power modes. The following diagram illustrates the connections of the SMC with other system components in the chip that are necessary to sequence the system through all power modes.



**Figure 13-6. Low-power system components and connections**

### 13.4.2.1 Stop mode entry sequence

Entry into a low-power stop mode (Stop, VLPS, LLS, VLLSx) is initiated by CPU execution of the WFI instruction. After the instruction is executed, the following sequence occurs:

1. The CPU clock is gated off immediately.
2. Requests are made to all non-CPU bus masters to enter Stop mode.
3. After all masters have acknowledged they are ready to enter Stop mode, requests are made to all bus slaves to enter Stop mode.
4. After all slaves have acknowledged they are ready to enter Stop mode, all system and bus clocks are gated off.
5. Clock generators are disabled in the MCG.
6. The on-chip regulator in the PMC and internal power switches are configured to meet the power consumption goals for the targeted low-power mode.

### 13.4.2.2 Stop mode exit sequence

Exit from a low-power stop mode is initiated either by a reset or an interrupt event. The following sequence then executes to restore the system to a run mode (RUN or VLPR):

1. The on-chip regulator in the PMC and internal power switches are restored.
2. Clock generators are enabled in the MCG.
3. System and bus clocks are enabled to all masters and slaves.
4. The CPU clock is enabled and the CPU begins servicing the reset or interrupt that initiated the exit from the low-power stop mode.

### 13.4.2.3 Aborted stop mode entry

If an interrupt or a reset occurs during a stop entry sequence, the SMC can abort the transition early and return to RUN mode without completely entering the stop mode. An aborted entry is possible only if the reset or interrupt occurs before the PMC begins the transition to stop mode regulation. After this point, the interrupt or reset is ignored until the PMC has completed its transition to stop mode regulation. When an aborted stop mode entry sequence occurs, the SMC's PMCTRL[STOPA] is set to 1.

### 13.4.2.4 Transition to wait modes

For wait modes (WAIT and VLPW), the CPU clock is gated off while all other clocking continues, as in RUN and VLPR mode operation. Some modules that support stop-in-wait functionality have their clocks disabled in these configurations.

### 13.4.2.5 Transition from stop modes to Debug mode

The debugger module supports a transition from STOP, WAIT, VLPS, and VLPW back to a Halted state when the debugger has been enabled, that is, ENBDM is 1. As part of this transition, system clocking is re-established and is equivalent to the normal RUN and VLPR mode clocking configuration.

## 13.4.3 Run modes

The device contains two different run modes:

- Run
- Very Low-Power Run (VLPR)

### 13.4.3.1 RUN mode

This is the normal operating mode for the device.

This mode is selected after any reset. When the ARM processor exits reset, it sets up the stack, program counter (PC), and link register (LR):

- The processor reads the start SP (SP\_main) from vector-table offset 0x000
- The processor reads the start PC from vector-table offset 0x004
- LR is set to 0xFFFF\_FFFF.

To reduce power in this mode, disable the clocks to unused modules using their corresponding clock gating control bits in the SIM's registers.

### 13.4.3.2 Very-Low Power Run (VLPR) mode

In VLPR mode, the on-chip voltage regulator is put into a stop mode regulation state. In this state, the regulator is designed to supply enough current to the MCU over a reduced frequency. To further reduce power in this mode, disable the clocks to unused modules using their corresponding clock gating control bits in the SIM's registers.

Before entering this mode, the following conditions must be met:

- The MCG must be configured in a mode which is supported during VLPR. See the Power Management details for information about these MCG modes.
- All clock monitors in the MCG must be disabled.
- The maximum frequencies of the system, bus, flash, and core are restricted. See the Power Management details about which frequencies are supported.
- Mode protection must be set to allow VLP modes, that is, PMPROT[AVLP] is 1.
- PMCTRL[RUNM] is set to 10b to enter VLPR.
- Flash programming/erasing is not allowed.

#### NOTE

Do not change the clock frequency while in VLPR mode, because the regulator is slow responding and cannot manage fast load transitions. In addition, do not modify the clock source in the MCG module, the module clock enables in the SIM, or any clock divider registers.

To reenter Normal Run mode, clear RUNM. The PMSTAT register is a read-only status register that can be used to determine when the system has completed an exit to RUN mode. When PMSTAT=RUN, the system is in run regulation and the MCU can run at full speed in any clock mode. If a higher execution frequency is desired, poll the PMSTAT register until it is set to RUN when returning from VLPR mode.

Any reset always causes an exit from VLPR and returns the device to RUN mode after the MCU exits its reset flow.

## 13.4.4 Wait modes

This device contains two different wait modes:

- Wait
- Very-Low Power Wait (VLPW)

### 13.4.4.1 WAIT mode

WAIT mode is entered when the ARM core enters the Sleep-Now or Sleep-On-Exit modes while SLEEPDEEP is cleared. The ARM CPU enters a low-power state in which it is not clocked, but peripherals continue to be clocked provided they are enabled. Clock gating to the peripheral is enabled via the SIM..

When an interrupt request occurs, the CPU exits WAIT mode and resumes processing in RUN mode, beginning with the stacking operations leading to the interrupt service routine.

A system reset will cause an exit from WAIT mode, returning the device to normal RUN mode.

### 13.4.4.2 Very-Low-Power Wait (VLPW) mode

VLPW is entered by entering the Sleep-Now or Sleep-On-Exit mode while SLEEPDEEP is cleared and the MCU is in VLPR mode.

In VLPW, the on-chip voltage regulator remains in its stop regulation state. In this state, the regulator is designed to supply enough current to the MCU over a reduced frequency. To further reduce power in this mode, disable the clocks to unused modules by clearing the peripherals' corresponding clock gating control bits in the SIM.

VLPR mode restrictions also apply to VLPW.

When an interrupt from VLPW occurs, the device returns to VLPR mode to execute the interrupt service routine.

A system reset will cause an exit from VLPW mode, returning the device to normal RUN mode.

### 13.4.5 Stop modes

This device contains a variety of stop modes to meet your application needs. The stop modes range from:

- a stopped CPU, with all I/O, logic, and memory states retained, and certain asynchronous mode peripherals operating

to:

- a powered down CPU, with only I/O and a small register file retained, very few asynchronous mode peripherals operating, while the remainder of the MCU is powered down.

The choice of stop mode depends upon the user's application, and how power usage and state retention versus functional needs may be traded off.

A module capable of providing an asynchronous interrupt to the device takes the device out of STOP mode and returns the device to normal RUN mode. Refer to the device's Power Management chapter for peripheral, I/O, and memory operation in STOP mode. When an interrupt request occurs, the CPU exits STOP mode and resumes processing, beginning with the stacking operations leading to the interrupt service routine.

A system reset will cause an exit from STOP mode, returning the device to normal RUN mode via an MCU reset.

### 13.4.5.2 Very-Low-Power Stop (VLPS) mode

VLPS mode can be entered in one of two ways:

- Entry into stop via the sleep-now or sleep-on-exit with the SLEEPDEEP bit set in the System Control Register in the ARM core while the MCU is in VLPR mode and STOPM=010 or 000 in the PMCTRL register.
- Entry into stop via the sleep-now or sleep-on-exit with the SLEEPDEEP bit set in the System Control Register in the ARM core while the MCU is in normal RUN mode and STOPM=010 in the PMCTRL register. When VLPS is entered directly from RUN mode, exit to VLPR is disabled by hardware and the system will always exit back to RUN.

In VLPS, the on-chip voltage regulator remains in its stop regulation state as in VLPR.

A module capable of providing an asynchronous interrupt to the device takes the device out of VLPS and returns the device to VLPR mode.

A system reset will also cause a VLPS exit, returning the device to normal RUN mode.

### 13.4.5.3 Low-Leakage Stop (LLS) mode

Low-Leakage Stop (LLS) mode can be entered from normal RUN or VLPR modes.

The MCU enters LLS mode if:

- In Sleep-Now or Sleep-On-Exit mode, SLEEPDEEP is set in the System Control Register in the ARM core, and
- The device is configured as shown in [Table 13-7](#).

In LLS, the on-chip voltage regulator is in stop regulation. Most of the peripherals are put in a state-retention mode that does not allow them to operate while in LLS.

Before entering LLS mode, the user should configure the low-leakage wakeup (LLWU) module to enable the desired wakeup sources. The available wakeup sources in LLS are detailed in the chip configuration details for this device.

After wakeup from LLS, the device returns to normal RUN mode with a pending LLWU module interrupt. In the LLWU interrupt service routine (ISR), the user can poll the LLWU module wakeup flags to determine the source of the wakeup.

**NOTE**

The LLWU interrupt must not be masked by the interrupt controller to avoid a scenario where the system does not fully exit stop mode on an LLS recovery.

An asserted RESET pin will cause an exit from LLS mode, returning the device to normal RUN mode. When LLS is exiting via the RESET pin, the PIN and WAKEUP bits are set in the SRS0 register of the reset control module (RCM).

#### 13.4.5.4 Very-Low-Leakage Stop (VLLSx) modes

This device contains these very low leakage modes:

- VLLS3
- VLLS1
- VLLS0

VLLSx is often used in this document to refer to all of these modes.

All VLLSx modes can be entered from normal RUN or VLPR modes.

The MCU enters the configured VLLS mode if:

- In Sleep-Now or Sleep-On-Exit mode, the SLEEPDEEP bit is set in the System Control Register in the ARM core, and
- The device is configured as shown in [Table 13-7](#).

In VLLS, the on-chip voltage regulator is in its stop-regulation state while most digital logic is powered off.

Before entering VLLS mode, the user should configure the low-leakage wakeup (LLWU) module to enable the desired wakeup sources. The available wakeup sources in VLLS are detailed in the chip configuration details for this device.

After wakeup from VLLS, the device returns to normal RUN mode with a pending LLWU interrupt. In the LLWU interrupt service routine (ISR), the user can poll the LLWU module wakeup flags to determine the source of the wakeup.

When entering VLLS, each I/O pin is latched as configured before executing VLLS. Because all digital logic in the MCU is powered off, all port and peripheral data is lost during VLLS. This information must be restored before the ACKISO bit in the PMC is set.

An asserted  $\overline{\text{RESET}}$  pin will cause an exit from any VLLS mode, returning the device to normal RUN mode. When exiting VLLS via the  $\overline{\text{RESET}}$  pin, the PIN and WAKEUP bits are set in the SRS0 register of the reset control module (RCM).

### 13.4.6 Debug in low power modes

When the MCU is secure, the device disables/limits debugger operation. When the MCU is unsecure, the ARM debugger can assert two power-up request signals:

- System power up, via SYSPWR in the Debug Port Control/Stat register
- Debug power up, via CDBGPWRUPREQ in the Debug Port Control/Stat register

When asserted while in RUN, WAIT, VLPR, or VLPW, the mode controller drives a corresponding acknowledge for each signal, that is, both CDBGPWRUPACK and CSYSPWRUPACK. When both requests are asserted, the mode controller handles attempts to enter STOP and VLPS by entering an emulated stop state. In this emulated stop state:

- the regulator is in run regulation,
- the MCG-generated clock source is enabled,
- all system clocks, except the core clock, are disabled,
- the debug module has access to core registers, and
- access to the on-chip peripherals is blocked.

No debug is available while the MCU is in LLS or VLLS modes. LLS is a state-retention mode and all debug operation can continue after waking from LLS, even in cases where system wakeup is due to a system reset event.

Entering into a VLLS mode causes all of the debug controls and settings to be powered off. To give time to the debugger to sync with the MCU, the MDM AP Control Register includes a Very-Low-Leakage Debug Request (VLLDBGREQ) bit that is set to configure the Reset Controller logic to hold the system in reset after the next recovery from a VLLS mode. This bit allows the debugger time to reinitialize the debug module before the debug session continues.

## Functional description

The MDM AP Control Register also includes a Very Low Leakage Debug Acknowledge (VLLDBGACK) bit that is set to release the ARM core being held in reset following a VLLS recovery. The debugger reinitializes all debug IP, and then asserts the VLLDBGACK control bit to allow the RCM to release the ARM core from reset and allow CPU operation to begin.

The VLLDBGACK bit is cleared by the debugger (or can be left set as is) or clears automatically due to the reset generated as part of the next VLLS recovery.

# Chapter 14

## Power Management Controller (PMC)

### 14.1 Introduction

#### NOTE

For the chip-specific implementation details of this module's instances see the chip configuration information.

The power management controller (PMC) contains the internal voltage regulator, power on reset (POR), and low voltage detect system.

### 14.2 Features

The PMC features include:

- Internal voltage regulator
- Active POR providing brown-out detect
- Low-voltage detect supporting two low-voltage trip points with four warning levels per trip point

### 14.3 Low-voltage detect (LVD) system

This device includes a system to guard against low-voltage conditions. This protects memory contents and controls MCU system states during supply voltage variations. The system is comprised of a power-on reset (POR) circuit and a LVD circuit with a user-selectable trip voltage: high ( $V_{LVDH}$ ) or low ( $V_{LVDL}$ ). The trip voltage is selected by the LVDSC1[LVDV] bits. The LVD is disabled upon entering VLPx, LLS, and VLLSx modes.

Two flags are available to indicate the status of the low-voltage detect system:

- The low voltage detect flag (LVDF) operates in a level sensitive manner. The LVDF bit is set when the supply voltage falls below the selected trip point (VLVD). The LVDF bit is cleared by writing one to the LVDACK bit, but only if the internal supply has returned above the trip point; otherwise, the LVDF bit remains set.
- The low voltage warning flag (LVWF) operates in a level sensitive manner. The LVWF bit is set when the supply voltage falls below the selected monitor trip point (VLVW). The LVWF bit is cleared by writing one to the LVWACK bit, but only if the internal supply has returned above the trip point; otherwise, the LVWF bit remains set.

### **14.3.1 LVD reset operation**

By setting the LVDRE bit, the LVD generates a reset upon detection of a low voltage condition. The low voltage detection threshold is determined by the LVDV bits. After an LVD reset occurs, the LVD system holds the MCU in reset until the supply voltage rises above this threshold. The LVD bit in the SRS register is set following an LVD or power-on reset.

### **14.3.2 LVD interrupt operation**

By configuring the LVD circuit for interrupt operation (LVDIE set and LVDRE clear), LVDSC1[LVDF] is set and an LVD interrupt request occurs upon detection of a low voltage condition. The LVDF bit is cleared by writing one to the LVDSC1[LVDACK] bit.

### **14.3.3 Low-voltage warning (LVW) interrupt operation**

The LVD system contains a low-voltage warning flag (LVWF) to indicate that the supply voltage is approaching, but is above, the LVD voltage. The LVW also has an interrupt, which is enabled by setting the LVDSC2[LVWIE] bit. If enabled, an LVW interrupt request occurs when the LVWF is set. LVWF is cleared by writing one to the LVDSC2[LVWACK] bit.

The LVDSC2[LVWV] bits select one of four trip voltages:

- Highest:  $V_{LVW4}$
- Two mid-levels:  $V_{LVW3}$  and  $V_{LVW2}$
- Lowest:  $V_{LVW1}$

## 14.4 I/O retention

When in LLS mode, the I/O pins are held in their input or output state. Upon wakeup, the PMC is re-enabled, goes through a power up sequence to full regulation, and releases the logic from state retention mode. The I/O are released immediately after a wakeup or reset event. In the case of LLS exit via a RESET pin, the I/O default to their reset state.

When in VLLS modes, the I/O states are held on a wakeup event (with the exception of wakeup by reset event) until the wakeup has been acknowledged via a write to the ACKISO bit. In the case of VLLS exit via a RESET pin, the I/O are released and default to their reset state. In this case, no write to the ACKISO is needed.

## 14.5 Memory map and register descriptions

PMC register details follow.

### NOTE

Different portions of PMC registers are reset only by particular reset types. Each register's description provides details. For more information about the types of reset on this chip, refer to the Reset section details.

The PMC registers can be written only in supervisor mode. Write accesses in user mode are blocked and will result in a bus error.

**PMC memory map**

| Absolute address (hex) | Register name   | Width (in bits) | Access | Reset value | Section/ page              |
|------------------------|---|-----------------|--------|-------------|----------------------------|
| 4007_D000              | Low Voltage Detect Status And Control 1 register (PMC_LVDSC1) | 8               | R/W    | 10h         | <a href="#">14.5.1/240</a> |
| 4007_D001              | Low Voltage Detect Status And Control 2 register (PMC_LVDSC2) | 8               | R/W    | 00h         | <a href="#">14.5.2/241</a> |
| 4007_D002              | Regulator Status And Control register (PMC_REGSC)             | 8               | R/W    | 04h         | <a href="#">14.5.3/242</a> |

### 14.5.1 Low Voltage Detect Status And Control 1 register (PMC\_LVDSC1)

This register contains status and control bits to support the low voltage detect function. This register should be written during the reset initialization program to set the desired controls even if the desired settings are the same as the reset settings.

While the device is in the very low power or low leakage modes, the LVD system is disabled regardless of LVDSC1 settings. To protect systems that must have LVD always on, configure the SMC's power mode protection register (PMPROT) to disallow any very low power or low leakage modes from being enabled.

See the device's data sheet for the exact LVD trip voltages.

#### NOTE

The LVDV bits are reset solely on a POR Only event. The register's other bits are reset on Chip Reset Not VLLS. For more information about these reset types, refer to the Reset section details.

Address: 4007\_D000h base + 0h offset = 4007\_D000h

| Bit   | 7    | 6      | 5     | 4     | 3 | 2 | 1    | 0 |
|-------|------|--------|-------|-------|---|---|------|---|
| Read  | LVDF | 0      | LVDIE | LVDRE | 0 |   |      |   |
| Write |      | LVDACK |       |       |   |   | LVDV |   |
| Reset | 0    | 0      | 0     | 1     | 0 | 0 | 0    | 0 |

#### PMC\_LVDSC1 field descriptions

| Field       | Description   |
|-------------|---|
| 7<br>LVDF   | Low-Voltage Detect Flag<br><br>This read-only status bit indicates a low-voltage detect event.<br><br>0 Low-voltage event not detected<br>1 Low-voltage event detected                      |
| 6<br>LVDACK | Low-Voltage Detect Acknowledge<br><br>This write-only bit is used to acknowledge low voltage detection errors. Write 1 to clear LVDF. Reads always return 0.                                |
| 5<br>LVDIE  | Low-Voltage Detect Interrupt Enable<br><br>Enables hardware interrupt requests for LVDF.<br><br>0 Hardware interrupt disabled (use polling)<br>1 Request a hardware interrupt when LVDF = 1 |

Table continues on the next page...

**PMC\_LVDSC1 field descriptions (continued)**

| Field           | Description  |
|-----------------|--|
| 4<br>LVDRE      | <p>Low-Voltage Detect Reset Enable</p> <p>This write-once bit enables LVDF events to generate a hardware reset. Additional writes are ignored.</p> <p>0 LVDF does not generate hardware resets<br/>1 Force an MCU reset when LVDF = 1</p>  |
| 3–2<br>Reserved | <p>This field is reserved.</p> <p>This read-only field is reserved and always has the value 0.</p>   |
| 1–0<br>LVDV     | <p>Low-Voltage Detect Voltage Select</p> <p>Selects the LVD trip point voltage (<math>V_{LVD}</math>).</p> <p>00 Low trip point selected (<math>V_{LVD} = V_{LVDL}</math>)<br/>01 High trip point selected (<math>V_{LVD} = V_{LVDH}</math>)<br/>10 Reserved<br/>11 Reserved</p> |

### 14.5.2 Low Voltage Detect Status And Control 2 register (PMC\_LVDSC2)

This register contains status and control bits to support the low voltage warning function.

While the device is in the very low power or low leakage modes, the LVD system is disabled regardless of LVDSC2 settings.

See the device's data sheet for the exact LVD trip voltages.

**NOTE**

The LVWV trip voltages depend on LVWV and LVDV bits.

**NOTE**

The LVWV bits are reset solely on a POR Only event. The register's other bits are reset on Chip Reset Not VLLS. For more information about these reset types, refer to the Reset section details.

Address: 4007\_D000h base + 1h offset = 4007\_D001h



**PMC\_LVDSC2 field descriptions**

| Field           | Description  |
|-----------------|--|
| 7<br>LVWF       | <p>Low-Voltage Warning Flag</p> <p>This read-only status bit indicates a low-voltage warning event. LVWF is set when <math>V_{Supply}</math> transitions below the trip point, or after reset and <math>V_{Supply}</math> is already below <math>V_{LVW}</math>. LVWF bit may be 1 after power on reset, therefore, to use LVW interrupt function, before enabling LVWIE, LVWF must be cleared by writing LVWACK first.</p> <ul style="list-style-type: none"> <li>0 Low-voltage warning event not detected</li> <li>1 Low-voltage warning event detected</li> </ul> |
| 6<br>LVWACK     | <p>Low-Voltage Warning Acknowledge</p> <p>This write-only bit is used to acknowledge low voltage warning errors. Write 1 to clear LVWF. Reads always return 0.</p>   |
| 5<br>LVWIE      | <p>Low-Voltage Warning Interrupt Enable</p> <p>Enables hardware interrupt requests for LVWF.</p> <ul style="list-style-type: none"> <li>0 Hardware interrupt disabled (use polling)</li> <li>1 Request a hardware interrupt when LVWF = 1</li> </ul>   |
| 4–2<br>Reserved | <p>This field is reserved.</p> <p>This read-only field is reserved and always has the value 0.</p>   |
| 1–0<br>LVWV     | <p>Low-Voltage Warning Voltage Select</p> <p>Selects the LVW trip point voltage (<math>V_{LVW}</math>). The actual voltage for the warning depends on LVDSC1[LVDV].</p> <ul style="list-style-type: none"> <li>00 Low trip point selected (<math>V_{LVW} = V_{LVW1}</math>)</li> <li>01 Mid 1 trip point selected (<math>V_{LVW} = V_{LVW2}</math>)</li> <li>10 Mid 2 trip point selected (<math>V_{LVW} = V_{LVW3}</math>)</li> <li>11 High trip point selected (<math>V_{LVW} = V_{LVW4}</math>)</li> </ul>  |

**14.5.3 Regulator Status And Control register (PMC\_REGSC)**

The PMC contains an internal voltage regulator. The voltage regulator design uses a bandgap reference that is also available through a buffer as input to certain internal peripherals, such as the CMP and ADC. The internal regulator provides a status bit (REGONS) indicating the regulator is in run regulation.

**NOTE**

This register is reset on Chip Reset Not VLLS and by reset types that trigger Chip Reset not VLLS. See the Reset section for more information.

Address: 4007\_D000h base + 2h offset = 4007\_D002h

| Bit   | 7 | 6 | 5        | 4    | 3      | 2      | 1        | 0    |
|-------|---|---|----------|------|--------|--------|----------|------|
| Read  | 0 |   | Reserved | BGEN | ACKISO | REGONS |          |      |
| Write |   |   |          |      | w1c    |        | Reserved | BGBE |
| Reset | 0 | 0 | 0        | 0    | 0      | 1      | 0        | 0    |

**PMC\_REGSC field descriptions**

| Field           | Description   |
|-----------------|---|
| 7–6<br>Reserved | This field is reserved.<br>This read-only field is reserved and always has the value 0.   |
| 5<br>Reserved   | This field is reserved.   |
| 4<br>BGEN       | Bandgap Enable In VLPx Operation<br><br>BGEN controls whether the bandgap is enabled in lower power modes of operation (VLPx, LLS, and VLLSx). When on-chip peripherals require the bandgap voltage reference in low power modes of operation, set BGEN to continue to enable the bandgap operation.<br><br><b>NOTE:</b> When the bandgap voltage reference is not needed in low power modes, clear BGEN to avoid excess power consumption.<br><br>0 Bandgap voltage reference is disabled in VLPx , LLS , and VLLSx modes<br>1 Bandgap voltage reference is enabled in VLPx , LLS , and VLLSx modes  |
| 3<br>ACKISO     | Acknowledge Isolation<br><br>Reading this bit indicates whether certain peripherals and the I/O pads are in a latched state as a result of having been in a VLLS mode. Writing one to this bit when it is set releases the I/O pads and certain peripherals to their normal run mode state.<br><br><b>NOTE:</b> After recovering from a VLLS mode, user should restore chip configuration before clearing ACKISO. In particular, pin configuration for enabled LLWU wakeup pins should be restored to avoid any LLWU flag from being falsely set when ACKISO is cleared.<br><br>0 Peripherals and I/O pads are in normal run state<br>1 Certain peripherals and I/O pads are in an isolated and latched state |
| 2<br>REGONS     | Regulator In Run Regulation Status<br><br>This read-only bit provides the current status of the internal voltage regulator.<br><br>0 Regulator is in stop regulation or in transition to/from it<br>1 Regulator is in run regulation  |
| 1<br>Reserved   | This field is reserved.<br><br><b>NOTE:</b> This reserved bit must remain cleared (set to 0).   |
| 0<br>BGBE       | Bandgap Buffer Enable<br><br>Enables the bandgap buffer.<br><br>0 Bandgap buffer not enabled<br>1 Bandgap buffer enabled  |



# Chapter 15

## Low-Leakage Wakeup Unit (LLWU)

### 15.1 Introduction

#### NOTE

For the chip-specific implementation details of this module's instances see the chip configuration information.

The LLWU module allows the user to select up to 16 external pin sources and up to 8 internal modules as a wakeup source from low-leakage power modes. The input sources are described in the device's chip configuration details. Each of the available wakeup sources can be individually enabled.

The LLWU module also includes two optional digital pin filters for the external wakeup pins.

#### 15.1.1 Features

The LLWU module features include:

- Support for up to 16 external input pins and up to 8 internal modules with individual enable bits
- Input sources may be external pins or from internal peripherals capable of running in LLS or VLLS. See the chip configuration information for wakeup input sources for this device.
- External pin wakeup inputs, each of which is programmable as falling-edge, rising-edge, or any change
- Wakeup inputs that are activated if enabled after MCU enters a low-leakage power mode
- Optional digital filters provided to qualify an external pin detect. When entering VLLS0, the filters are disabled and bypassed.

## 15.1.2 Modes of operation

The LLWU module becomes functional on entry into a low-leakage power mode. After recovery from LLS, the LLWU is immediately disabled. After recovery from VLLS, the LLWU continues to detect wakeup events until the user has acknowledged the wakeup via a write to the PMC\_REGSC[ACKISO] bit.

### 15.1.2.1 LLS mode

The LLWU module provides up to 16 external wakeup inputs and up to 8 internal module wakeup inputs.

Wakeup events due to external wakeup inputs and internal module wakeup inputs result in an interrupt flow when exiting LLS.

#### NOTE

The LLWU interrupt must not be masked by the interrupt controller to avoid a scenario where the system does not fully exit Stop mode on an LLS recovery.

### 15.1.2.2 VLLS modes

The LLWU module provides up to 16 external wakeup inputs and up to 8 internal module wakeup inputs. All wakeup events result in VLLS exit via a reset flow.

### 15.1.2.3 Non-low leakage modes

The LLWU is not active in all non-low leakage modes where detection and control logic are in a static state. The LLWU registers are accessible in non-low leakage modes and are available for configuring and reading status when bus transactions are possible.

When the wakeup pin filters are enabled, filter operation begins immediately. If a low leakage mode is entered within 5 LPO clock cycles of an active edge, the edge event will be detected by the LLWU.

### 15.1.2.4 Debug mode

When the chip is in Debug mode and then enters LLS or a VLLSx mode, no debug logic works in the fully-functional low-leakage mode. Upon an exit from the LLS or VLLSx mode, the LLWU becomes inactive.

### 15.1.3 Block diagram

The following figure is the block diagram for the LLWU module.

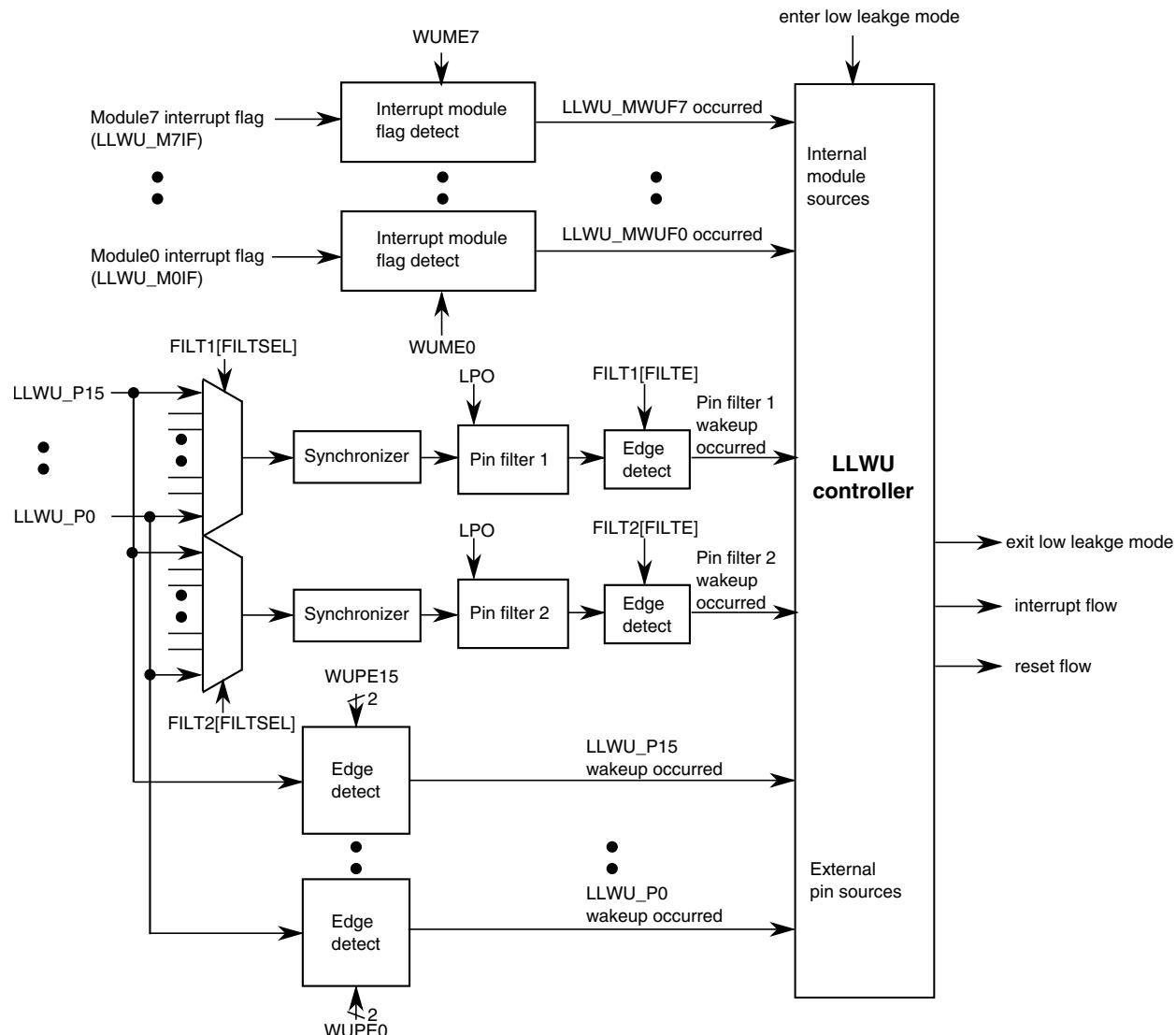


Figure 15-1. LLWU block diagram

## 15.2 LLWU signal descriptions

The signal properties of LLWU are shown in the following table. The external wakeup input pins can be enabled to detect either rising-edge, falling-edge, or on any change.

**Table 15-1. LLWU signal descriptions**

| Signal  | Description              | I/O |
|---------|--------------------------|-----|
| LLWU_Pn | Wakeup inputs (n = 0-15) | I   |

## 15.3 Memory map/register definition

The LLWU includes the following registers:

- Five 8-bit wakeup source enable registers
  - Enable external pin input sources
  - Enable internal peripheral sources
- Three 8-bit wakeup flag registers
  - Indication of wakeup source that caused exit from a low-leakage power mode includes external pin or internal module interrupt
- Two 8-bit wakeup pin filter enable registers

### NOTE

The LLWU registers can be written only in supervisor mode. Write accesses in user mode are blocked and will result in a bus error.

All LLWU registers are reset by Chip Reset not VLLS and by reset types that trigger Chip Reset not VLLS. Each register's displayed reset value represents this subset of reset types.

LLWU registers are unaffected by reset types that do not trigger Chip Reset not VLLS. For more information about the types of reset on this chip, refer to the [Introduction](#) details.

## LLWU memory map

| Absolute address (hex) | Register name                             | Width (in bits) | Access | Reset value | Section/ page               |
|------------------------|---|-----------------|--------|-------------|-----------------------------|
| 4007_C000              | LLWU Pin Enable 1 register (LLWU_PE1)     | 8               | R/W    | 00h         | <a href="#">15.3.1/249</a>  |
| 4007_C001              | LLWU Pin Enable 2 register (LLWU_PE2)     | 8               | R/W    | 00h         | <a href="#">15.3.2/250</a>  |
| 4007_C002              | LLWU Pin Enable 3 register (LLWU_PE3)     | 8               | R/W    | 00h         | <a href="#">15.3.3/251</a>  |
| 4007_C003              | LLWU Pin Enable 4 register (LLWU_PE4)     | 8               | R/W    | 00h         | <a href="#">15.3.4/252</a>  |
| 4007_C004              | LLWU Module Enable register (LLWU_ME)     | 8               | R/W    | 00h         | <a href="#">15.3.5/253</a>  |
| 4007_C005              | LLWU Flag 1 register (LLWU_F1)            | 8               | R/W    | 00h         | <a href="#">15.3.6/255</a>  |
| 4007_C006              | LLWU Flag 2 register (LLWU_F2)            | 8               | R/W    | 00h         | <a href="#">15.3.7/257</a>  |
| 4007_C007              | LLWU Flag 3 register (LLWU_F3)            | 8               | R      | 00h         | <a href="#">15.3.8/258</a>  |
| 4007_C008              | LLWU Pin Filter 1 register (LLWU_FILTER1) | 8               | R/W    | 00h         | <a href="#">15.3.9/260</a>  |
| 4007_C009              | LLWU Pin Filter 2 register (LLWU_FILTER2) | 8               | R/W    | 00h         | <a href="#">15.3.10/261</a> |

### 15.3.1 LLWU Pin Enable 1 register (LLWU\_PE1)

LLWU\_PE1 contains the field to enable and select the edge detect type for the external wakeup input pins LLWU\_P3-LLWU\_P0.

#### NOTE

This register is reset on Chip Reset not VLLS and by reset types that trigger Chip Reset not VLLS. It is unaffected by reset types that do not trigger Chip Reset not VLLS. See the [Introduction](#) details for more information.

Address: 4007\_C000h base + 0h offset = 4007\_C000h

| Bit        | 7 | 6     | 5 | 4     | 3 | 2     | 1 | 0     |
|------------|---|-------|---|-------|---|-------|---|-------|
| Read Write |   | WUPE3 |   | WUPE2 |   | WUPE1 |   | WUPE0 |
| Reset      | 0 | 0     | 0 | 0     | 0 | 0     | 0 | 0     |

#### LLWU\_PE1 field descriptions

| Field | Description                   | Table continues | 75.7-1.0.25 0 m -1.0 35.25 |
|-------|-------------------------------|-----------------|----------------------------|
| 7–6   | Wakeup Pin Enable For LLWU_P3 |                 |                            |

**WUPE3** Enables and configures the edge detection for the wakeup pin.

- 00 External input pin disabled as wakeup input
- 01 External input pin enabled with rising edge detection
- 10 External input pin enabled with falling edge detection
- 11 External input pin enabled with any change detection

**5–4 WUPE2** Wakeup Pin Enable For LLWU\_P2

Enables and configures the edge detection for the wakeup pin.

*Table continues on the next page...*

## LLWU\_PE1 field descriptions (continued)

| Field        | Description   |
|--------------|---|
|              | 00 External input pin disabled as wakeup input<br>01 External input pin enabled with rising edge detection<br>10 External input pin enabled with falling edge detection<br>11 External input pin enabled with any change detection  |
| 3–2<br>WUPE1 | <b>Wakeup Pin Enable For LLWU_P1</b><br>Enables and configures the edge detection for the wakeup pin.<br>00 External input pin disabled as wakeup input<br>01 External input pin enabled with rising edge detection<br>10 External input pin enabled with falling edge detection<br>11 External input pin enabled with any change detection |
| 1–0<br>WUPE0 | <b>Wakeup Pin Enable For LLWU_P0</b><br>Enables and configures the edge detection for the wakeup pin.<br>00 External input pin disabled as wakeup input<br>01 External input pin enabled with rising edge detection<br>10 External input pin enabled with falling edge detection<br>11 External input pin enabled with any change detection |

### 15.3.2 LLWU Pin Enable 2 register (LLWU\_PE2)

LLWU\_PE2 contains the field to enable and select the edge detect type for the external wakeup input pins LLWU\_P7-LLWU\_P4.

#### NOTE

This register is reset on Chip Reset not VLLS and by reset types that trigger Chip Reset not VLLS. It is unaffected by reset types that do not trigger Chip Reset not VLLS. See the [Introduction](#) details for more information.

Address: 4007\_C000h base + 1h offset = 4007\_C001h



**LLWU\_PE2 field descriptions (continued)**

| Field        | Description  |
|--------------|--|
|              | 10 External input pin enabled with falling edge detection<br>11 External input pin enabled with any change detection   |
| 5–4<br>WUPE6 | Wakeup Pin Enable For LLWU_P6<br><br>Enables and configures the edge detection for the wakeup pin.<br><br>00 External input pin disabled as wakeup input<br>01 External input pin enabled with rising edge detection<br>10 External input pin enabled with falling edge detection<br>11 External input pin enabled with any change detection |
| 3–2<br>WUPE5 | Wakeup Pin Enable For LLWU_P5<br><br>Enables and configures the edge detection for the wakeup pin.<br><br>00 External input pin disabled as wakeup input<br>01 External input pin enabled with rising edge detection<br>10 External input pin enabled with falling edge detection<br>11 External input pin enabled with any change detection |
| 1–0<br>WUPE4 | Wakeup Pin Enable For LLWU_P4<br><br>Enables and configures the edge detection for the wakeup pin.<br><br>00 External input pin disabled as wakeup input<br>01 External input pin enabled with rising edge detection<br>10 External input pin enabled with falling edge detection<br>11 External input pin enabled with any change detection |

**15.3.3 LLWU Pin Enable 3 register (LLWU\_PE3)**

LLWU\_PE3 contains the field to enable and select the edge detect type for the external wakeup input pins LLWU\_P11-LLWU\_P8.

**NOTE**

This register is reset on Chip Reset not VLLS and by reset types that trigger Chip Reset not VLLS. It is unaffected by reset types that do not trigger Chip Reset not VLLS. See the [Introduction](#) details for more information.

Address: 4007\_C000h base + 2h offset = 4007\_C002h

| Bit   | 7      | 6 | 5      | 4 | 3     | 2 | 1     | 0 |
|-------|--------|---|--------|---|-------|---|-------|---|
| Read  | WUPE11 |   | WUPE10 |   | WUPE9 |   | WUPE8 |   |
| Write | 0      | 0 | 0      | 0 | 0     | 0 | 0     | 0 |

**LLWU\_PE3 field descriptions**

| Field         | Description   |
|---------------|---|
| 7–6<br>WUPE11 | <p>Wakeup Pin Enable For LLWU_P11</p> <p>Enables and configures the edge detection for the wakeup pin.</p> <ul style="list-style-type: none"> <li>00 External input pin disabled as wakeup input</li> <li>01 External input pin enabled with rising edge detection</li> <li>10 External input pin enabled with falling edge detection</li> <li>11 External input pin enabled with any change detection</li> </ul> |
| 5–4<br>WUPE10 | <p>Wakeup Pin Enable For LLWU_P10</p> <p>Enables and configures the edge detection for the wakeup pin.</p> <ul style="list-style-type: none"> <li>00 External input pin disabled as wakeup input</li> <li>01 External input pin enabled with rising edge detection</li> <li>10 External input pin enabled with falling edge detection</li> <li>11 External input pin enabled with any change detection</li> </ul> |
| 3–2<br>WUPE9  | <p>Wakeup Pin Enable For LLWU_P9</p> <p>Enables and configures the edge detection for the wakeup pin.</p> <ul style="list-style-type: none"> <li>00 External input pin disabled as wakeup input</li> <li>01 External input pin enabled with rising edge detection</li> <li>10 External input pin enabled with falling edge detection</li> <li>11 External input pin enabled with any change detection</li> </ul>  |
| 1–0<br>WUPE8  | <p>Wakeup Pin Enable For LLWU_P8</p> <p>Enables and configures the edge detection for the wakeup pin.</p> <ul style="list-style-type: none"> <li>00 External input pin disabled as wakeup input</li> <li>01 External input pin enabled with rising edge detection</li> <li>10 External input pin enabled with falling edge detection</li> <li>11 External input pin enabled with any change detection</li> </ul>  |

**15.3.4 LLWU Pin Enable 4 register (LLWU\_PE4)**

LLWU\_PE4 contains the field to enable and select the edge detect type for the external wakeup input pins LLWU\_P15-LLWU\_P12.

**NOTE**

This register is reset on Chip Reset not VLLS and by reset types that trigger Chip Reset not VLLS. It is unaffected by reset types that do not trigger Chip Reset not VLLS. See the [Introduction](#) details for more information.

Address: 4007\_C000h base + 3h offset = 4007\_C003h

| Bit   | 7      | 6 | 5      | 4 | 3      | 2 | 1      | 0 |
|-------|--------|---|--------|---|--------|---|--------|---|
| Read  | WUPE15 |   | WUPE14 |   | WUPE13 |   | WUPE12 |   |
| Write | 0      | 0 | 0      | 0 | 0      | 0 | 0      | 0 |

### LLWU\_PE4 field descriptions

| Field         | Description   |
|---------------|---|
| 7–6<br>WUPE15 | <p>Wakeup Pin Enable For LLWU_P15</p> <p>Enables and configures the edge detection for the wakeup pin.</p> <ul style="list-style-type: none"> <li>00 External input pin disabled as wakeup input</li> <li>01 External input pin enabled with rising edge detection</li> <li>10 External input pin enabled with falling edge detection</li> <li>11 External input pin enabled with any change detection</li> </ul> |
| 5–4<br>WUPE14 | <p>Wakeup Pin Enable For LLWU_P14</p> <p>Enables and configures the edge detection for the wakeup pin.</p> <ul style="list-style-type: none"> <li>00 External input pin disabled as wakeup input</li> <li>01 External input pin enabled with rising edge detection</li> <li>10 External input pin enabled with falling edge detection</li> <li>11 External input pin enabled with any change detection</li> </ul> |
| 3–2<br>WUPE13 | <p>Wakeup Pin Enable For LLWU_P13</p> <p>Enables and configures the edge detection for the wakeup pin.</p> <ul style="list-style-type: none"> <li>00 External input pin disabled as wakeup input</li> <li>01 External input pin enabled with rising edge detection</li> <li>10 External input pin enabled with falling edge detection</li> <li>11 External input pin enabled with any change detection</li> </ul> |
| 1–0<br>WUPE12 | <p>Wakeup Pin Enable For LLWU_P12</p> <p>Enables and configures the edge detection for the wakeup pin.</p> <ul style="list-style-type: none"> <li>00 External input pin disabled as wakeup input</li> <li>01 External input pin enabled with rising edge detection</li> <li>10 External input pin enabled with falling edge detection</li> <li>11 External input pin enabled with any change detection</li> </ul> |

### 15.3.5 LLWU Module Enable register (LLWU\_ME)

LLWU\_ME contains the bits to enable the internal module flag as a wakeup input source for inputs MWUF7-MWUF0.

#### NOTE

This register is reset on Chip Reset not VLLS and by reset types that trigger Chip Reset not VLLS. It is unaffected by reset

## Memory map/register definition

types that do not trigger Chip Reset nor VLLS. See the [Introduction](#) details for more information.

Address: 4007\_C000h base + 4h offset = 4007\_C004h

| Bit           | 7     | 6     | 5     | 4     | 3     | 2     | 1     | 0     |
|---------------|-------|-------|-------|-------|-------|-------|-------|-------|
| Read<br>Write | WUME7 | WUME6 | WUME5 | WUME4 | WUME3 | WUME2 | WUME1 | WUME0 |
| Reset         | 0     | 0     | 0     | 0     | 0     | 0     | 0     | 0     |

### LLWU\_ME field descriptions

| Field      | Description   |
|------------|---|
| 7<br>WUME7 | <p>Wakeup Module Enable For Module 7</p> <p>Enables an internal module as a wakeup source input.</p> <p>0 Internal module flag not used as wakeup source<br/>1 Internal module flag used as wakeup source</p> |
| 6<br>WUME6 | <p>Wakeup Module Enable For Module 6</p> <p>Enables an internal module as a wakeup source input.</p> <p>0 Internal module flag not used as wakeup source<br/>1 Internal module flag used as wakeup source</p> |
| 5<br>WUME5 | <p>Wakeup Module Enable For Module 5</p> <p>Enables an internal module as a wakeup source input.</p> <p>0 Internal module flag not used as wakeup source<br/>1 Internal module flag used as wakeup source</p> |
| 4<br>WUME4 | <p>Wakeup Module Enable For Module 4</p> <p>Enables an internal module as a wakeup source input.</p> <p>0 Internal module flag not used as wakeup source<br/>1 Internal module flag used as wakeup source</p> |
| 3<br>WUME3 | <p>Wakeup Module Enable For Module 3</p> <p>Enables an internal module as a wakeup source input.</p> <p>0 Internal module flag not used as wakeup source<br/>1 Internal module flag used as wakeup source</p> |
| 2<br>WUME2 | <p>Wakeup Module Enable For Module 2</p> <p>Enables an internal module as a wakeup source input.</p> <p>0 Internal module flag not used as wakeup source<br/>1 Internal module flag used as wakeup source</p> |
| 1<br>WUME1 | <p>Wakeup Module Enable for Module 1</p> <p>Enables an internal module as a wakeup source input.</p> <p>0 Internal module flag not used as wakeup source<br/>1 Internal module flag used as wakeup source</p> |

*Table continues on the next page...*

**LLWU\_ME field descriptions (continued)**

| Field      | Description   |
|------------|---|
| 0<br>WUME0 | <p>Wakeup Module Enable For Module 0</p> <p>Enables an internal module as a wakeup source input.</p> <p>0 Internal module flag not used as wakeup source<br/>1 Internal module flag used as wakeup source</p> |

**15.3.6 LLWU Flag 1 register (LLWU\_F1)**

LLWU\_F1 contains the wakeup flags indicating which wakeup source caused the MCU to exit LLS or VLLS mode. For LLS, this is the source causing the CPU interrupt flow. For VLLS, this is the source causing the MCU reset flow.

The external wakeup flags are read-only and clearing a flag is accomplished by a write of a 1 to the corresponding WUFx bit. The wakeup flag (WUFx), if set, will remain set if the associated WUPEx bit is cleared.

**NOTE**

This register is reset on Chip Reset not VLLS and by reset types that trigger Chip Reset not VLLS. It is unaffected by reset types that do not trigger Chip Reset not VLLS. See the [Introduction](#) details for more information.

Address: 4007\_C000h base + 5h offset = 4007\_C005h

| Bit   | 7    | 6    | 5    | 4    | 3    | 2    | 1    | 0    |
|-------|------|------|------|------|------|------|------|------|
| Read  | WUF7 | WUF6 | WUF5 | WUF4 | WUF3 | WUF2 | WUF1 | WUF0 |
| Write | w1c  |
| Reset | 0    | 0    | 0    | 0    | 0    | 0    | 0    | 0    |

**LLWU\_F1 field descriptions**

| Field     | Description   |
|-----------|---|
| 7<br>WUF7 | <p>Wakeup Flag For LLWU_P7</p> <p>Indicates that an enabled external wakeup pin was a source of exiting a low-leakage power mode. To clear the flag write a one to WUF7.</p> <p>0 LLWU_P7 input was not a wakeup source<br/>1 LLWU_P7 input was a wakeup source</p> |
| 6<br>WUF6 | <p>Wakeup Flag For LLWU_P6</p> <p>Indicates that an enabled external wakeup pin was a source of exiting a low-leakage power mode. To clear the flag write a one to WUF6.</p>  |

*Table continues on the next page...*

**LLWU\_F1 field descriptions (continued)**

| Field     | Description   |
|-----------|---|
|           | <p>0 LLWU_P6 input was not a wakeup source<br/>1 LLWU_P6 input was a wakeup source</p>  |
| 5<br>WUF5 | <p>Wakeup Flag For LLWU_P5<br/><br/>Indicates that an enabled external wakeup pin was a source of exiting a low-leakage power mode. To clear the flag write a one to WUF5.</p> <p>0 LLWU_P5 input was not a wakeup source<br/>1 LLWU_P5 input was a wakeup source</p> |
| 4<br>WUF4 | <p>Wakeup Flag For LLWU_P4<br/><br/>Indicates that an enabled external wakeup pin was a source of exiting a low-leakage power mode. To clear the flag write a one to WUF4.</p> <p>0 LLWU_P4 input was not a wakeup source<br/>1 LLWU_P4 input was a wakeup source</p> |
| 3<br>WUF3 | <p>Wakeup Flag For LLWU_P3<br/><br/>Indicates that an enabled external wakeup pin was a source of exiting a low-leakage power mode. To clear the flag write a one to WUF3.</p> <p>0 LLWU_P3 input was not a wakeup source<br/>1 LLWU_P3 input was a wakeup source</p> |
| 2<br>WUF2 | <p>Wakeup Flag For LLWU_P2<br/><br/>Indicates that an enabled external wakeup pin was a source of exiting a low-leakage power mode. To clear the flag write a one to WUF2.</p> <p>0 LLWU_P2 input was not a wakeup source<br/>1 LLWU_P2 input was a wakeup source</p> |
| 1<br>WUF1 | <p>Wakeup Flag For LLWU_P1<br/><br/>Indicates that an enabled external wakeup pin was a source of exiting a low-leakage power mode. To clear the flag write a one to WUF1.</p> <p>0 LLWU_P1 input was not a wakeup source<br/>1 LLWU_P1 input was a wakeup source</p> |
| 0<br>WUF0 | <p>Wakeup Flag For LLWU_P0<br/><br/>Indicates that an enabled external wakeup pin was a source of exiting a low-leakage power mode. To clear the flag write a one to WUF0.</p> <p>0 LLWU_P0 input was not a wakeup source<br/>1 LLWU_P0 input was a wakeup source</p> |

### 15.3.7 LLWU Flag 2 register (LLWU\_F2)

LLWU\_F2 contains the wakeup flags indicating which wakeup source caused the MCU to exit LLS or VLLS mode. For LLS, this is the source causing the CPU interrupt flow. For VLLS, this is the source causing the MCU reset flow.

The external wakeup flags are read-only and clearing a flag is accomplished by a write of a 1 to the corresponding WUFx bit. The wakeup flag (WUFx), if set, will remain set if the associated WUPEx bit is cleared.

#### NOTE

This register is reset on Chip Reset not VLLS and by reset types that trigger Chip Reset not VLLS. It is unaffected by reset types that do not trigger Chip Reset not VLLS. See the [Introduction](#) details for more information.

Address: 4007\_C000h base + 6h offset = 4007\_C006h

| Bit   | 7     | 6     | 5     | 4     | 3     | 2     | 1    | 0    |
|-------|-------|-------|-------|-------|-------|-------|------|------|
| Read  | WUF15 | WUF14 | WUF13 | WUF12 | WUF11 | WUF10 | WUF9 | WUF8 |
| Write | w1c   | w1c   | w1c   | w1c   | w1c   | w1c   | w1c  | w1c  |
| Reset | 0     | 0     | 0     | 0     | 0     | 0     | 0    | 0    |

#### LLWU\_F2 field descriptions

| Field      | Description   |
|------------|---|
| 7<br>WUF15 | <p>Wakeup Flag For LLWU_P15</p> <p>Indicates that an enabled external wakeup pin was a source of exiting a low-leakage power mode. To clear the flag write a one to WUF15.</p> <p>0 LLWU_P15 input was not a wakeup source<br/>1 LLWU_P15 input was a wakeup source</p> |
| 6<br>WUF14 | <p>Wakeup Flag For LLWU_P14</p> <p>Indicates that an enabled external wakeup pin was a source of exiting a low-leakage power mode. To clear the flag write a one to WUF14.</p> <p>0 LLWU_P14 input was not a wakeup source<br/>1 LLWU_P14 input was a wakeup source</p> |
| 5<br>WUF13 | <p>Wakeup Flag For LLWU_P13</p> <p>Indicates that an enabled external wakeup pin was a source of exiting a low-leakage power mode. To clear the flag write a one to WUF13.</p> <p>0 LLWU_P13 input was not a wakeup source<br/>1 LLWU_P13 input was a wakeup source</p> |

Table continues on the next page...

**LLWU\_F2 field descriptions (continued)**

| Field      | Description   |
|------------|---|
| 4<br>WUF12 | <p>Wakeup Flag For LLWU_P12</p> <p>Indicates that an enabled external wakeup pin was a source of exiting a low-leakage power mode. To clear the flag write a one to WUF12.</p> <p>0 LLWU_P12 input was not a wakeup source<br/>1 LLWU_P12 input was a wakeup source</p> |
| 3<br>WUF11 | <p>Wakeup Flag For LLWU_P11</p> <p>Indicates that an enabled external wakeup pin was a source of exiting a low-leakage power mode. To clear the flag write a one to WUF11.</p> <p>0 LLWU_P11 input was not a wakeup source<br/>1 LLWU_P11 input was a wakeup source</p> |
| 2<br>WUF10 | <p>Wakeup Flag For LLWU_P10</p> <p>Indicates that an enabled external wakeup pin was a source of exiting a low-leakage power mode. To clear the flag write a one to WUF10.</p> <p>0 LLWU_P10 input was not a wakeup source<br/>1 LLWU_P10 input was a wakeup source</p> |
| 1<br>WUF9  | <p>Wakeup Flag For LLWU_P9</p> <p>Indicates that an enabled external wakeup pin was a source of exiting a low-leakage power mode. To clear the flag write a one to WUF9.</p> <p>0 LLWU_P9 input was not a wakeup source<br/>1 LLWU_P9 input was a wakeup source</p>     |
| 0<br>WUF8  | <p>Wakeup Flag For LLWU_P8</p> <p>Indicates that an enabled external wakeup pin was a source of exiting a low-leakage power mode. To clear the flag write a one to WUF8.</p> <p>0 LLWU_P8 input was not a wakeup source<br/>1 LLWU_P8 input was a wakeup source</p>     |

**15.3.8 LLWU Flag 3 register (LLWU\_F3)**

LLWU\_F3 contains the wakeup flags indicating which internal wakeup source caused the MCU to exit LLS or VLLS mode. For LLS, this is the source causing the CPU interrupt flow. For VLLS, this is the source causing the MCU reset flow.

For internal peripherals that are capable of running in a low-leakage power mode, such as iRTC or CMP modules, the flag from the associated peripheral is accessible as the MWUFx bit. The flag will need to be cleared in the peripheral instead of writing a 1 to the MWUFx bit.

**NOTE**

This register is reset on Chip Reset not VLLS and by reset types that trigger Chip Reset not VLLS. It is unaffected by reset types that do not trigger Chip Reset not VLLS. See the [Introduction](#) details for more information.

Address: 4007\_C000h base + 7h offset = 4007\_C007h

| Bit   | 7     | 6     | 5     | 4     | 3     | 2     | 1     | 0     |
|-------|-------|-------|-------|-------|-------|-------|-------|-------|
| Read  | MWUF7 | MWUF6 | MWUF5 | MWUF4 | MWUF3 | MWUF2 | MWUF1 | MWUF0 |
| Write |       |       |       |       |       |       |       |       |
| Reset | 0     | 0     | 0     | 0     | 0     | 0     | 0     | 0     |

**LLWU\_F3 field descriptions**

| Field      | Description  |
|------------|--|
| 7<br>MWUF7 | <p>Wakeup flag For module 7</p> <p>Indicates that an enabled internal peripheral was a source of exiting a low-leakage power mode. To clear the flag, follow the internal peripheral flag clearing mechanism.</p> <p>0 Module 7 input was not a wakeup source<br/>1 Module 7 input was a wakeup source</p> |
| 6<br>MWUF6 | <p>Wakeup flag For module 6</p> <p>Indicates that an enabled internal peripheral was a source of exiting a low-leakage power mode. To clear the flag, follow the internal peripheral flag clearing mechanism.</p> <p>0 Module 6 input was not a wakeup source<br/>1 Module 6 input was a wakeup source</p> |
| 5<br>MWUF5 | <p>Wakeup flag For module 5</p> <p>Indicates that an enabled internal peripheral was a source of exiting a low-leakage power mode. To clear the flag, follow the internal peripheral flag clearing mechanism.</p> <p>0 Module 5 input was not a wakeup source<br/>1 Module 5 input was a wakeup source</p> |
| 4<br>MWUF4 | <p>Wakeup flag For module 4</p> <p>Indicates that an enabled internal peripheral was a source of exiting a low-leakage power mode. To clear the flag, follow the internal peripheral flag clearing mechanism.</p> <p>0 Module 4 input was not a wakeup source<br/>1 Module 4 input was a wakeup source</p> |
| 3<br>MWUF3 | <p>Wakeup flag For module 3</p> <p>Indicates that an enabled internal peripheral was a source of exiting a low-leakage power mode. To clear the flag, follow the internal peripheral flag clearing mechanism.</p> <p>0 Module 3 input was not a wakeup source<br/>1 Module 3 input was a wakeup source</p> |
| 2<br>MWUF2 | <p>Wakeup flag For module 2</p>  |

*Table continues on the next page...*

**LLWU\_F3 field descriptions (continued)**

| Field   | Description  |
|---------|--|
|         | Indicates that an enabled internal peripheral was a source of exiting a low-leakage power mode. To clear the flag, follow the internal peripheral flag clearing mechanism.<br>0 Module 2 input was not a wakeup source<br>1 Module 2 input was a wakeup source                             |
| 1 MWUF1 | Wakeup flag For module 1<br>Indicates that an enabled internal peripheral was a source of exiting a low-leakage power mode. To clear the flag, follow the internal peripheral flag clearing mechanism.<br>0 Module 1 input was not a wakeup source<br>1 Module 1 input was a wakeup source |
| 0 MWUFO | Wakeup flag For module 0<br>Indicates that an enabled internal peripheral was a source of exiting a low-leakage power mode. To clear the flag, follow the internal peripheral flag clearing mechanism.<br>0 Module 0 input was not a wakeup source<br>1 Module 0 input was a wakeup source |

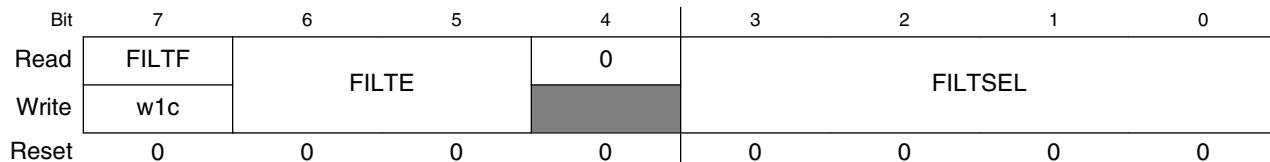
**15.3.9 LLWU Pin Filter 1 register (LLWU\_FILT1)**

LLWU\_FILT1 is a control and status register that is used to enable/disable the digital filter 1 features for an external pin.

**NOTE**

This register is reset on Chip Reset not VLLS and by reset types that trigger Chip Reset not VLLS. It is unaffected by reset types that do not trigger Chip Reset not VLLS. See the [Introduction](#) details for more information.

Address: 4007\_C000h base + 8h offset = 4007\_C008h

**LLWU\_FILT1 field descriptions**

| Field   | Description   |
|---------|---|
| 7 FILTF | Filter Detect Flag<br>Indicates that the filtered external wakeup pin, selected by FILTSEL, was a source of exiting a low-leakage power mode. To clear the flag write a one to FILTF. |

*Table continues on the next page...*

**LLWU\_FILT1 field descriptions (continued)**

| Field       | Description   |
|-------------|---|
|             | 0 Pin Filter 1 was not a wakeup source<br>1 Pin Filter 1 was a wakeup source  |
| 6–5 FILTE   | Digital Filter On External Pin<br><br>Controls the digital filter options for the external pin detect.<br><br>00 Filter disabled<br>01 Filter posedge detect enabled<br>10 Filter negedge detect enabled<br>11 Filter any edge detect enabled |
| 4 Reserved  | This field is reserved.<br>This read-only field is reserved and always has the value 0.   |
| 3–0 FILTSEL | Filter Pin Select<br><br>Selects 1 out of the 16 wakeup pins to be muxed into the filter.<br><br>0000 Select LLWU_P0 for filter<br>... ...<br>1111 Select LLWU_P15 for filter   |

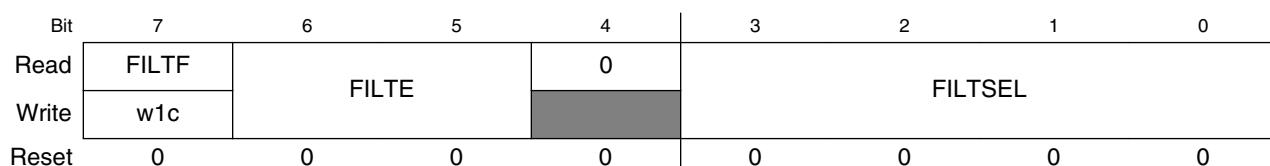
**15.3.10 LLWU Pin Filter 2 register (LLWU\_FILT2)**

LLWU\_FILT2 is a control and status register that is used to enable/disable the digital filter 2 features for an external pin.

**NOTE**

This register is reset on Chip Reset not VLLS and by reset types that trigger Chip Reset not VLLS. It is unaffected by reset types that do not trigger Chip Reset not VLLS. See the [Introduction](#) details for more information.

Address: 4007\_C000h base + 9h offset = 4007\_C009h

**LLWU\_FILT2 field descriptions**

| Field   | Description   |
|---------|---|
| 7 FILTF | Filter Detect Flag<br><br>Indicates that the filtered external wakeup pin, selected by FILTSEL, was a source of exiting a low-leakage power mode. To clear the flag write a one to FILTF. |

*Table continues on the next page...*

**LLWU\_FILT2 field descriptions (continued)**

| Field          | Description   |
|----------------|---|
|                | 0 Pin Filter 2 was not a wakeup source<br>1 Pin Filter 2 was a wakeup source  |
| 6–5<br>FILTE   | Digital Filter On External Pin<br><br>Controls the digital filter options for the external pin detect.<br><br>00 Filter disabled<br>01 Filter posedge detect enabled<br>10 Filter negedge detect enabled<br>11 Filter any edge detect enabled |
| 4<br>Reserved  | This field is reserved.<br>This read-only field is reserved and always has the value 0.   |
| 3–0<br>FILTSEL | Filter Pin Select<br><br>Selects 1 out of the 16 wakeup pins to be muxed into the filter.<br><br>0000 Select LLWU_P0 for filter<br>... ...<br>1111 Select LLWU_P15 for filter   |

## 15.4 Functional description

This on-chip peripheral module is called a low-leakage wakeup unit (LLWU) module because it allows internal peripherals and external input pins as a source of wakeup from low-leakage modes. It is operational only in LLS and VLLSx modes.

The LLWU module contains pin enables for each external pin and internal module. For each external pin, the user can disable or select the edge type for the wakeup. Type options are:

- Falling-edge
- Rising-edge
- Either-edge

When an external pin is enabled as a wakeup source, the pin must be configured as an input pin.

The LLWU implements optional 3-cycle glitch filters, based on the LPO clock. A detected external pin is required to remain asserted until the enabled glitch filter times out. Additional latency of up to 2 cycles is due to synchronization, which results in a total of up to 5 cycles of delay before the detect circuit alerts the system to the wakeup or reset event when the filter function is enabled. Two wakeup detect filters are available to detect up to two external pins. Glitch filtering is not provided on the internal modules.

For internal module wakeup operation, the WUMEx bit enables the associated module as a wakeup source.

### 15.4.1 LLS mode

Wakeup events triggered from either an external pin input or an internal module input result in a CPU interrupt flow to begin user code execution.

### 15.4.2 VLLS modes

In the case of a wakeup due to external pin or internal module wakeup, recovery is always via a reset flow and the RCM\_SRS[WAKEUP] is set indicating the low-leakage mode was active. State retention data is lost and I/O will be restored after PMC\_REGSC[ACKISO] has been written.

### 15.4.3 Initialization

For an enabled peripheral wakeup input, the peripheral flag must be cleared by software before entering LLS or VLLSx mode to avoid an immediate exit from the mode.

Flags associated with external input pins, filtered and unfiltered, must also be cleared by software prior to entry to LLS or VLLSx mode.

After enabling an external pin filter or changing the source pin, wait at least 5 LPO clock cycles before entering LLS or VLLSx mode to allow the filter to initialize.

#### NOTE

After recovering from a VLLS mode, user must restore chip configuration before clearing ACKISO. In particular, pin configuration for enabled LLWU wakeup pins must be restored to avoid any LLWU flag from being falsely set when ACKISO is cleared.

The signal selected as a wakeup source pin must be a digital pin, as selected in the pin mux control.



# Chapter 16

## Reset Control Module (RCM)

### 16.1 Introduction

This chapter describes the registers of the Reset Control Module (RCM). The RCM implements many of the reset functions for the chip. See the chip's reset chapter for more information.

### 16.2 Reset memory map and register descriptions

The Reset Control Module (RCM) registers provide reset status information and reset filter control.

#### NOTE

The RCM registers can be written only in supervisor mode. Write accesses in user mode are blocked and will result in a bus error.

**RCM memory map**

| Absolute address (hex) | Register name                                | Width (in bits) | Access | Reset value | Section/ page              |
|------------------------|--|-----------------|--------|-------------|----------------------------|
| 4007_F000              | System Reset Status Register 0 (RCM_SRS0)    | 8               | R      | 82h         | <a href="#">16.2.1/265</a> |
| 4007_F001              | System Reset Status Register 1 (RCM_SRS1)    | 8               | R      | 00h         | <a href="#">16.2.2/267</a> |
| 4007_F004              | Reset Pin Filter Control register (RCM_RPFC) | 8               | R/W    | 00h         | <a href="#">16.2.3/268</a> |
| 4007_F005              | Reset Pin Filter Width register (RCM_RPFW)   | 8               | R/W    | 00h         | <a href="#">16.2.4/269</a> |

#### 16.2.1 System Reset Status Register 0 (RCM\_SRS0)

This register includes read-only status flags to indicate the source of the most recent reset. The reset state of these bits depends on what caused the MCU to reset.

**NOTE**

The reset value of this register depends on the reset source:

- POR (including LVD) — 0x82
- LVD (without POR) — 0x02
- VLLS mode wakeup due to  $\overline{\text{RESET}}$  pin assertion — 0x41
- VLLS mode wakeup due to other wakeup sources — 0x01
- Other reset — a bit is set if its corresponding reset source caused the reset

Address: 4007\_F000h base + 0h offset = 4007\_F000h

| Bit   | 7   | 6   | 5    | 4 | 3   | 2   | 1   | 0      |
|-------|-----|-----|------|---|-----|-----|-----|--------|
| Read  | POR | PIN | WDOG | 0 | LOL | LOC | LVD | WAKEUP |
| Write |     |     |      |   |     |     |     |        |
| Reset | 1   | 0   | 0    | 0 | 0   | 0   | 1   | 0      |

**RCM\_SRS0 field descriptions**

| Field         | Description  |
|---------------|--|
| 7<br>POR      | <p>Power-On Reset</p> <p>Indicates a reset has been caused by the power-on detection logic. Because the internal supply voltage was ramping up at the time, the low-voltage reset (LVD) status bit is also set to indicate that the reset occurred while the internal supply was below the LVD threshold.</p> <p>0 Reset not caused by POR<br/>1 Reset caused by POR</p> |
| 6<br>PIN      | <p>External Reset Pin</p> <p>Indicates a reset has been caused by an active-low level on the external <math>\overline{\text{RESET}}</math> pin.</p> <p>0 Reset not caused by external reset pin<br/>1 Reset caused by external reset pin</p>   |
| 5<br>WDOG     | <p>Watchdog</p> <p>Indicates a reset has been caused by the watchdog timer Computer Operating Properly (COP) timing out. This reset source can be blocked by disabling the COP watchdog: write 00 to the SIM's COPC[COPT] field.</p> <p>0 Reset not caused by watchdog timeout<br/>1 Reset caused by watchdog timeout</p>  |
| 4<br>Reserved | <p>This field is reserved.</p> <p>This read-only field is reserved and always has the value 0.</p>   |
| 3<br>LOL      | <p>Loss-of-Lock Reset</p> <p>Indicates a reset has been caused by a loss of lock in the MCG PLL. See the MCG description for information on the loss-of-clock event.</p> <p>0 Reset not caused by a loss of lock in the PLL<br/>1 Reset caused by a loss of lock in the PLL</p>  |

*Table continues on the next page...*

**RCM\_SRS0 field descriptions (continued)**

| Field       | Description   |
|-------------|---|
| 2<br>LOC    | <p>Loss-of-Clock Reset</p> <p>Indicates a reset has been caused by a loss of external clock. The MCG clock monitor must be enabled for a loss of clock to be detected. Refer to the detailed MCG description for information on enabling the clock monitor.</p> <p>0 Reset not caused by a loss of external clock.<br/>1 Reset caused by a loss of external clock.</p>  |
| 1<br>LVD    | <p>Low-Voltage Detect Reset</p> <p>If the LVDRE bit is set and the supply drops below the LVD trip voltage, an LVD reset occurs. This bit is also set by POR.</p> <p>0 Reset not caused by LVD trip or POR<br/>1 Reset caused by LVD trip or POR</p>  |
| 0<br>WAKEUP | <p>Low Leakage Wakeup Reset</p> <p>Indicates a reset has been caused by an enabled LLWU module wakeup source while the chip was in a low leakage mode. In LLS mode, the RESET pin is the only wakeup source that can cause this reset. Any enabled wakeup source in a VLLSx mode causes a reset. This bit is cleared by any reset except WAKEUP.</p> <p>0 Reset not caused by LLWU module wakeup source<br/>1 Reset caused by LLWU module wakeup source</p> |

**16.2.2 System Reset Status Register 1 (RCM\_SRS1)**

This register includes read-only status flags to indicate the source of the most recent reset. The reset state of these bits depends on what caused the MCU to reset.

**NOTE**

The reset value of this register depends on the reset source:

- POR (including LVD) — 0x00
- LVD (without POR) — 0x00
- VLLS mode wakeup — 0x00
- Other reset — a bit is set if its corresponding reset source caused the reset

Address: 4007\_F000h base + 1h offset = 4007\_F001h

| Bit   | 7 | 6 | 5       | 4 | 3      | 2  | 1      | 0 |
|-------|---|---|---------|---|--------|----|--------|---|
| Read  | 0 | 0 | SACKERR | 0 | MDM_AP | SW | LOCKUP | 0 |
| Write |   |   |         |   |        |    |        |   |
| Reset | 0 | 0 | 0       | 0 | 0      | 0  | 0      | 0 |

**RCM\_SRS1 field descriptions**

| Field         | Description   |
|---------------|---|
| 7<br>Reserved | This field is reserved.<br>This read-only field is reserved and always has the value 0.   |
| 6<br>Reserved | This field is reserved.<br>This read-only field is reserved and always has the value 0.   |
| 5<br>SACKERR  | Stop Mode Acknowledge Error Reset<br><br>Indicates that after an attempt to enter Stop mode, a reset has been caused by a failure of one or more peripherals to acknowledge within approximately one second to enter stop mode.<br><br>0 Reset not caused by peripheral failure to acknowledge attempt to enter stop mode<br>1 Reset caused by peripheral failure to acknowledge attempt to enter stop mode |
| 4<br>Reserved | This field is reserved.<br>This read-only field is reserved and always has the value 0.   |
| 3<br>MDM_AP   | MDM-AP System Reset Request<br><br>Indicates a reset has been caused by the host debugger system setting of the System Reset Request bit in the MDM-AP Control Register.<br><br>0 Reset not caused by host debugger system setting of the System Reset Request bit<br>1 Reset caused by host debugger system setting of the System Reset Request bit  |
| 2<br>SW       | Software<br><br>Indicates a reset has been caused by software setting of SYSRESETREQ bit in Application Interrupt and Reset Control Register in the ARM core.<br><br>0 Reset not caused by software setting of SYSRESETREQ bit<br>1 Reset caused by software setting of SYSRESETREQ bit   |
| 1<br>LOCKUP   | Core Lockup<br><br>Indicates a reset has been caused by the ARM core indication of a LOCKUP event.<br><br>0 Reset not caused by core LOCKUP event<br>1 Reset caused by core LOCKUP event  |
| 0<br>Reserved | This field is reserved.<br>This read-only field is reserved and always has the value 0.   |

**16.2.3 Reset Pin Filter Control register (RCM\_RPFC)****NOTE**

The reset values of bits 2-0 are for Chip POR only. They are unaffected by other reset types.

**NOTE**

The bus clock filter is reset when disabled or when entering stop mode. The LPO filter is reset when disabled .

Address: 4007\_F000h base + 4h offset = 4007\_F004h

| Bit   | 7 | 6 | 5 | 4 | 3 | 2        | 1 | 0         |
|-------|---|---|---|---|---|----------|---|-----------|
| Read  |   |   | 0 |   |   | RSTFLTSS |   | RSTFLTSRW |
| Write | 0 | 0 | 0 | 0 | 0 | 0        | 0 | 0         |
| Reset |   |   |   |   |   |          |   |           |

**RCM\_RPFC field descriptions**

| Field            | Description   |
|------------------|---|
| 7–3<br>Reserved  | This field is reserved.<br>This read-only field is reserved and always has the value 0.   |
| 2<br>RSTFLTSS    | Reset Pin Filter Select in Stop Mode<br><br>Selects how the reset pin filter is enabled in Stop and VLPS modes , and also during LLS and VLLS modes. On exit from VLLS mode, this bit should be reconfigured before clearing ACKISO in the PMC.<br><br>0 All filtering disabled<br>1 LPO clock filter enabled |
| 1–0<br>RSTFLTSRW | Reset Pin Filter Select in Run and Wait Modes<br><br>Selects how the reset pin filter is enabled in run and wait modes.<br><br>00 All filtering disabled<br>01 Bus clock filter enabled for normal operation<br>10 LPO clock filter enabled for normal operation<br>11 Reserved                               |

**16.2.4 Reset Pin Filter Width register (RCM\_RPFW)****NOTE**

The reset values of the bits in the RSTFLTSEL field are for Chip POR only. They are unaffected by other reset types.

Address: 4007\_F000h base + 5h offset = 4007\_F005h

| Bit   | 7 | 6 | 5 | 4 | 3 | 2         | 1 | 0 |
|-------|---|---|---|---|---|-----------|---|---|
| Read  |   |   | 0 |   |   | RSTFLTSEL |   |   |
| Write | 0 | 0 | 0 | 0 | 0 | 0         | 0 | 0 |
| Reset |   |   |   |   |   |           |   |   |

**RCM\_RPFW field descriptions**

| Field            | Description   |
|------------------|---|
| 7–5<br>Reserved  | This field is reserved.<br>This read-only field is reserved and always has the value 0. |
| 4–0<br>RSTFLTSEL | Reset Pin Filter Bus Clock Select<br><br>Selects the reset pin bus clock filter width.  |

*Table continues on the next page...*

**RCM\_RPFW field descriptions (continued)**

| Field | Description                  |
|-------|------------------------------|
| 00000 | Bus clock filter count is 1  |
| 00001 | Bus clock filter count is 2  |
| 00010 | Bus clock filter count is 3  |
| 00011 | Bus clock filter count is 4  |
| 00100 | Bus clock filter count is 5  |
| 00101 | Bus clock filter count is 6  |
| 00110 | Bus clock filter count is 7  |
| 00111 | Bus clock filter count is 8  |
| 01000 | Bus clock filter count is 9  |
| 01001 | Bus clock filter count is 10 |
| 01010 | Bus clock filter count is 11 |
| 01011 | Bus clock filter count is 12 |
| 01100 | Bus clock filter count is 13 |
| 01101 | Bus clock filter count is 14 |
| 01110 | Bus clock filter count is 15 |
| 01111 | Bus clock filter count is 16 |
| 10000 | Bus clock filter count is 17 |
| 10001 | Bus clock filter count is 18 |
| 10010 | Bus clock filter count is 19 |
| 10011 | Bus clock filter count is 20 |
| 10100 | Bus clock filter count is 21 |
| 10101 | Bus clock filter count is 22 |
| 10110 | Bus clock filter count is 23 |
| 10111 | Bus clock filter count is 24 |
| 11000 | Bus clock filter count is 25 |
| 11001 | Bus clock filter count is 26 |
| 11010 | Bus clock filter count is 27 |
| 11011 | Bus clock filter count is 28 |
| 11100 | Bus clock filter count is 29 |
| 11101 | Bus clock filter count is 30 |
| 11110 | Bus clock filter count is 31 |
| 11111 | Bus clock filter count is 32 |

# Chapter 17

## Bit Manipulation Engine (BME)

### 17.1 Introduction

The Bit Manipulation Engine (BME) provides hardware support for atomic read-modify-write memory operations to the peripheral address space in Cortex-M0+ based microcontrollers. This architectural capability is also known as "decorated storage" as it defines a mechanism for providing additional semantics for load and store operations to memory-mapped peripherals beyond just the reading and writing of data values to the addressed memory locations. In the BME definition, the "decoration", that is, the additional semantic information, is encoded into the peripheral address used to reference the memory.

By combining the basic load and store instructions of the Cortex-M instruction set architecture (v6M, v7M) with the concept of decorated storage provided by the BME, the resulting implementation provides a robust and efficient read-modify-write capability to this class of ultra low-end microcontrollers. The resulting architectural capability defined by this core platform function is targeted at the manipulation of n-bit fields in peripheral registers and is consistent with I/O hardware addressing in the Embedded C standard. For most BME commands, a single core read or write bus cycle is converted into an atomic read-modify-write, that is, an indivisible "read followed by a write" bus sequence.

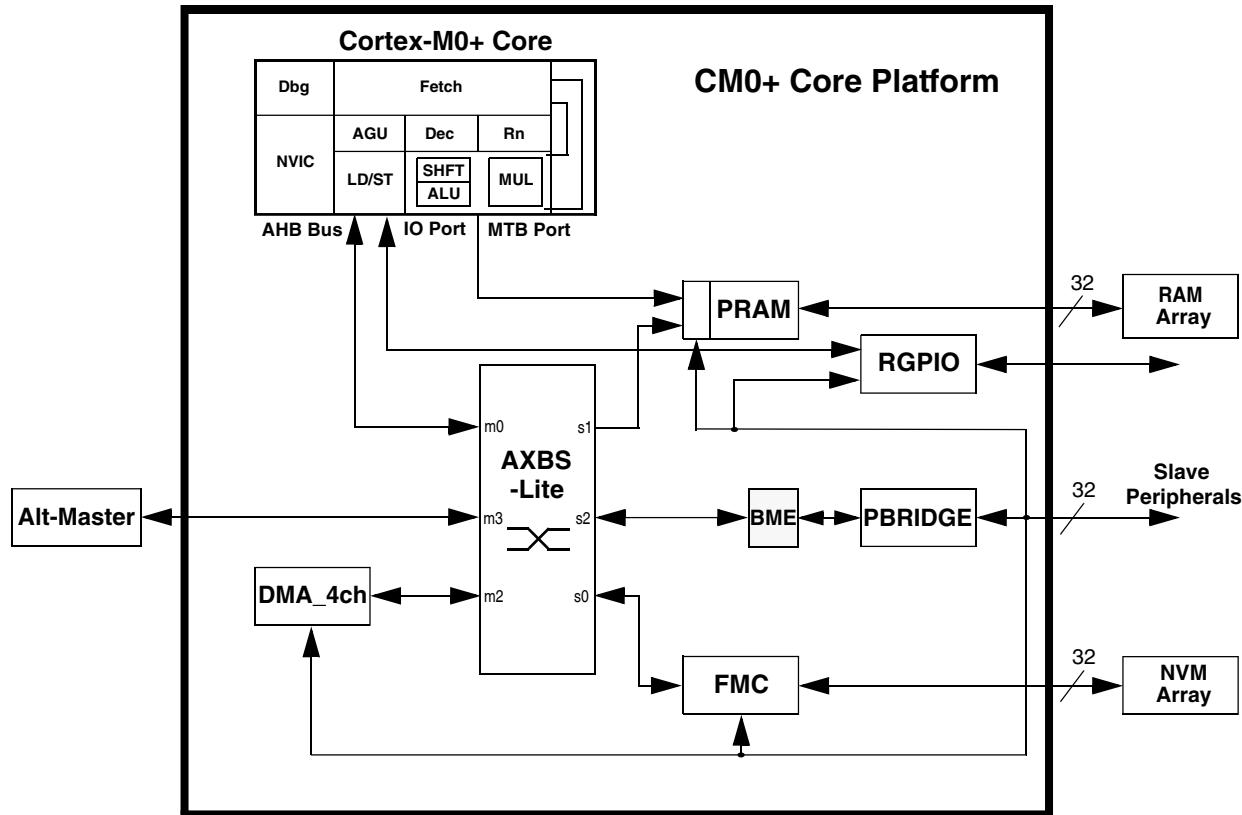
BME decorated references are only available on system bus transactions generated by the processor core and targeted at the standard 512 KB peripheral address space based at 0x4000\_0000<sup>1</sup>. The decoration semantic is embedded into address bits[28:19], creating a 448 MB space at addresses 0x4400\_0000 - 0x5FF\_FFFF; these bits are stripped out of the actual address sent to the peripheral bus controller and used by the BME to define and control its operation.

---

1. To be perfectly accurate, the peripheral address space occupies a 516 KB region: 512 KB based at 0x4000\_0000 plus a 4 KB space based at 0x400F\_F000 for GPIO accesses. This organization provides compatibility with the Kinetis K Family. Attempted accesses to the memory space located between 0x4008\_0000 - 0x400E\_FFFF are error terminated due to an illegal address.

### 17.1.1 Overview

The following figure is a generic block diagram of the processor core and platform for this class of ultra low-end microcontrollers.



**Figure 17-1. Generic Cortex-M0+ core platform block diagram**

As shown in the block diagram, the BME module interfaces to a crossbar switch AHB slave port as its primary input and sources an AHB bus output to the Peripheral Bridge (PBRIDGE) controller. The BME hardware microarchitecture is a 2-stage pipeline design matching the protocol of the AMBA-AHB system bus interfaces. The PBRIDGE module converts the AHB system bus protocol into the IPS/APB protocol used by the attached slave peripherals.

### 17.1.2 Features

The key features of the BME include:

- Lightweight implementation of decorated storage for peripheral address space
- Additional access semantics encoded into the reference address

- Resides between a crossbar switch slave port and a peripheral bridge bus controller
- 2-stage pipeline design matching the AHB system bus protocol
- Combinatorially passes non-decorated accesses to peripheral bridge bus controller
- Conversion of decorated loads and stores from processor core into atomic read-modify-writes
- Decorated loads support unsigned bit field extracts, load-and-{set,clear}-1bit operations
- Decorated stores support bit field inserts, logical AND, OR and XOR operations
- Support for byte, halfword and word-sized decorated operations
- Supports minimum signal toggling on AHB output bus to reduce power dissipation

### 17.1.3 Modes of Operation

The BME module does not support any special modes of operation. As a memory-mapped device located on a crossbar slave AHB system bus port, BME responds based strictly on memory addresses for accesses to the connected peripheral bridge bus controller.

All functionality associated with the BME module resides in the core platform's clock domain; this includes its connections with the crossbar slave port and the PBRIDGE bus controller.

## 17.2 External Signal Description

The BME module does not directly support any external interfaces.

The internal interfaces include two standard AHB buses with 32-bit datapath widths: the primary input from the appropriate crossbar slave port (`mx_h<signal>`) and the primary output to the PBRIDGE bus controller (`sx_h<signal>`).

Note the signal directions are defined by the BME's view and are labeled based on the dominant direction. Accordingly, the `mx_h<signal>` AHB bus is the primary input, even though there are certain data phase signals (`mx_h{rdata, ready, resp}`) which are outputs from BME. Likewise, the `sx_h<signal>` AHB bus is the primary output even though there are specific data phase signals (`sx_h{rdata, ready, resp}`) which are inputs to BME.

## 17.3 Memory Map and Register Definition

The BME module provides a memory-mapped capability and does not include any programming model registers. The exact set of functions supported by the BME are detailed in the [Functional Description](#).

The peripheral address space occupies a 516 KB region: 512 KB based at 0x4000\_0000 plus a 4 KB space based at 0x400F\_F000 for GPIO accesses; the decorated address space is mapped to the 448 MB region located at 0x4400\_0000 - 0x5FFF\_FFFF.

## 17.4 Functional Description

This section details the specific functions supported by the BME.

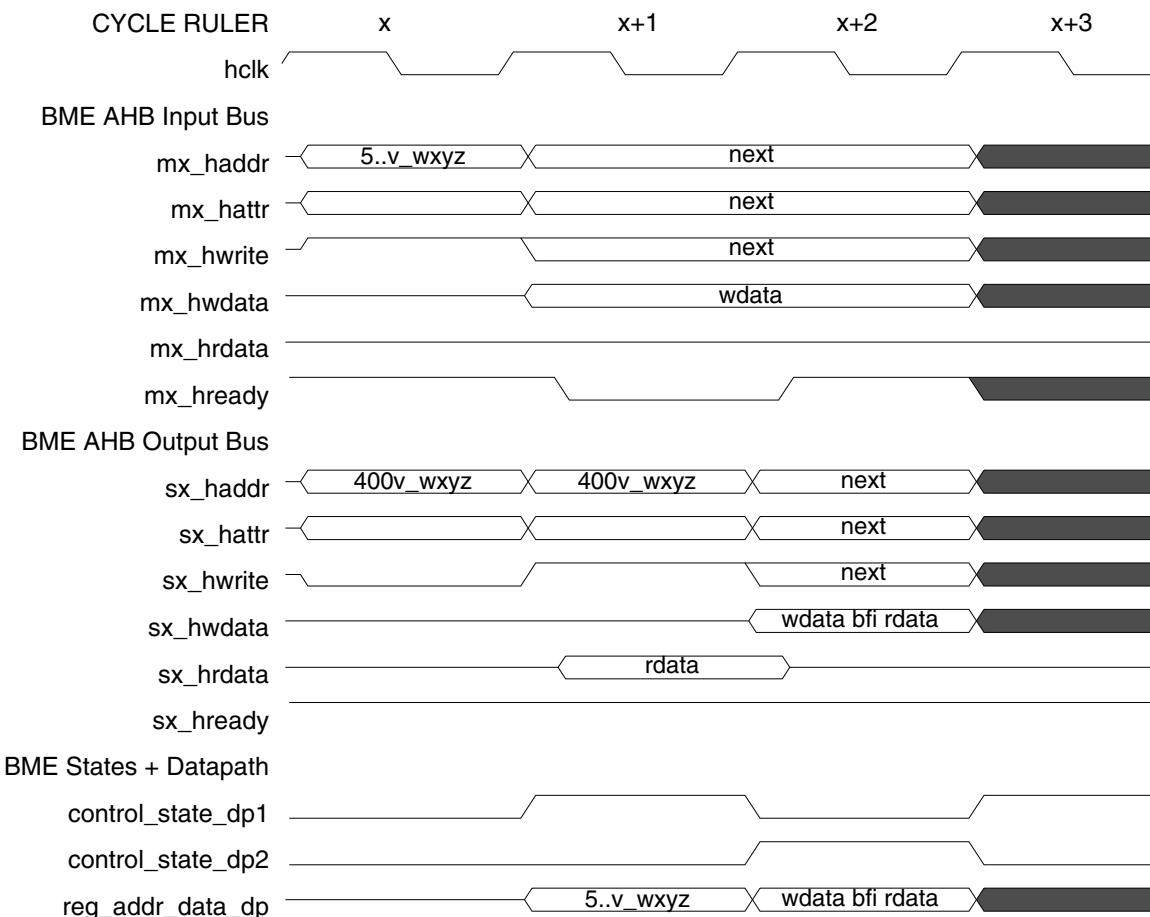
Recall the combination of the basic load and store instructions of the Cortex-M instruction set architecture (v6M, v7M) plus the concept of decorated storage provided by the BME, the resulting implementation provides a robust and efficient read-modify-write capability to this class of ultra low-end microcontrollers. The resulting architectural capability defined by this core platform function is targeted at the manipulation of n-bit fields in peripheral registers and is consistent with I/O hardware addressing in the Embedded C standard. For most BME commands, a single core read or write bus cycle is converted into an atomic read-modify-write, that is, an indivisible "read followed by a write" bus sequence.

Consider decorated store operations first, then decorated loads.

### 17.4.1 BME Decorated Stores

The functions supported by the BME's decorated stores include three logical operators (AND, OR, XOR) plus a bit field insert. For all these operations, BME converts a single decorated AHB store transaction into a 2-cycle atomic read-modify-write sequence, where the combined read-modify operation is performed in the first AHB data phase, and then the write is performed in the second AHB data phase.

A generic timing diagram of a decorated store showing a bit field insert operation is shown as follows:



**Figure 17-2. Decorated store: bit field insert timing diagram**

All the decorated store operations follow the same execution template shown in the figure above, a 2-cycle read-modify-write operation:

- Cycle x, 1st AHB address phase: Write from input bus (`mx_h<signal>`) is translated into a read operation on the output bus (`sx_h<signal>`) using the actual memory address (with the decoration removed) and then captured in a register (`reg_addr_data_dp`)
- Cycle x+1, 2nd AHB address phase: Write access with the registered (but actual) memory address is output (`sx_h<signal>`)
- Cycle x+1, 1st AHB data phase: Memory read data (`sx_hrdta`) is modified using the input bus write data (`mx_hwdata`) and the function defined by the decoration and captured in a data register (`reg_addr_data_dp`); the input bus cycle is stalled (`mx_hready = 0`)
- Cycle x+2, 2nd AHB data phase: Registered write data is sourced onto the output write data bus (`sx_hwdata`)

**NOTE**

Any wait states inserted by the peripheral slave device

**Table 17-1. Cycle definitions of decorated store: logical AND**

| Pipeline Stage | Cycle  |  |  |
|----------------|--|--|--|
|                | x  | x+1  | x+2  |
| BME AHB_ap     | Forward addr to memory;<br>Decode decoration; Convert<br>master_wt to slave_rd;<br>Capture address, attributes | Recirculate captured addr +<br>attr to memory as slave_wt                                | <next>   |
| BME AHB_dp     | <previous>   | Perform memory read; Form<br>(rdata & wdata) and capture<br>destination data in register | Perform write sending<br>registered data to memory |

### 17.4.1.2 Decorated Store Logical OR (OR)

This command performs an atomic read-modify-write of the referenced memory location. First, the location is read; it is then modified by performing a logical OR operation using the write data operand sourced for the system bus cycle; finally, the result of the OR operation is written back into the referenced memory location.

The data size is specified by the write operation and can be byte (8 bit), halfword (16 bit) or word (32 bit). The core performs the required write data lane replication on byte and halfword transfers.

| 31    | 30 | 29 | 28 | 27 | 26 | 25 | 24 | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2        | 1        | 0 |
|-------|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|---|---|---|---|---|---|---|----------|----------|---|
| ioorb | 0  | 1  | 0  | 0  | 1  | 0  | -  | -  | -  | -  | -  | -  | -  | -  | -  | -  | -  | -  | -  | -  | -  | - | - | - | - | - | - | - | -        | mem_addr |   |
| ioorh | 0  | 1  | 0  | 0  | 1  | 0  | -  | -  | -  | -  | -  | -  | -  | -  | -  | -  | -  | -  | -  | -  | -  | - | - | - | - | - | - | - | mem_addr | 0        |   |
| ioorw | 0  | 1  | 0  | 0  | 1  | 0  | -  | -  | -  | -  | -  | -  | -  | -  | -  | -  | -  | -  | -  | -  | -  | - | - | - | - | - | - | - | mem_addr | 0        | 0 |

**Figure 17-4. Decorated Address Store: Logical OR**

where  $\text{addr}[28:26] = 010$  specifies the OR operation, and  $\text{mem\_addr}[19:0]$  specifies the address offset into the peripheral space based at  $0x4000_0000$ . The "-" indicates an address bit "don't care".

The decorated OR write operation is defined in the following pseudo-code as:

```
ioor<sz>(accessAddress, wdata) // decorated store OR
tmp    = mem[accessAddress & 0xE00FFFF, size] // memory read
tmp    = tmp | wdata // modify
mem[accessAddress & 0xE00FFFF, size] = tmp // memory write
```

The cycle-by-cycle BME operations are detailed in the following table

**Table 17-2. Cycle definitions of decorated store: logical OR**

| Pipeline Stage | Cycle  |  |  |
|----------------|--|--|--|
|                | x  | x+1  | x+2  |
| BME AHB_ap     | Forward addr to memory;<br>Decode decoration; Convert<br>master_wt to slave_rd;<br>Capture address, attributes | Recirculate captured addr +<br>attr to memory as slave_wt                                | <next>   |
| BME AHB_dp     | <previous>   | Perform memory read; Form<br>(rdata   wdata) and capture<br>destination data in register | Perform write sending<br>registered data to memory |

### 17.4.1.3 Decorated Store: Logical XOR (XOR)

This command performs an atomic read-modify-write of the referenced memory location. First, the location is read; it is then modified by performing a logical XOR (exclusive-OR) operation using the write data operand sourced for the system bus cycle; finally, the result of the XOR operation is written back into the referenced memory location.

The data size is specified by the write operation and can be byte (8-bit), halfword (16-bit) or word (32-bit). The core performs the required write data lane replication on byte and halfword transfers.

|        | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 | 15 | 14 | 13 | 12 | 11       | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2   | 1 | 0 |
|--------|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----------|----|---|---|---|---|---|---|---|-----|---|---|
| ioxorb | 0  | 1  | 0  | 0  | 1  | 1  | -  | -  | -  | -  | -  | -  | -  | -  | -  | -  | -  | -  | -  | -  | mem_addr |    |   |   |   |   |   |   |   |     |   |   |
| ioxorh | 0  | 1  | 0  | 0  | 1  | 1  | -  | -  | -  | -  | -  | -  | -  | -  | -  | -  | -  | -  | -  | -  | mem_addr |    |   |   |   |   |   |   |   | 0   |   |   |
| ioxorw | 0  | 1  | 0  | 0  | 1  | 1  | -  | -  | -  | -  | -  | -  | -  | -  | -  | -  | -  | -  | -  | -  | mem_addr |    |   |   |   |   |   |   |   | 0 0 |   |   |

**Figure 17-5. Decorated Address Store: Logical XOR**

where  $\text{addr}[28:26] = 011$  specifies the XOR operation, and  $\text{mem\_addr}[19:0]$  specifies the address offset into the peripheral space based at  $0x4000_0000$ . The "-" indicates an address bit "don't care".

The decorated XOR write operation is defined in the following pseudo-code as:

```
ioxor<sz>(accessAddress, wdata) // decorated store XOR
tmp    = mem[accessAddress & 0xE00FFFFF, size] // memory read
tmp    = tmp ^ wdata // modify
mem[accessAddress & 0xE00FFFFF, size] = tmp // memory write
```

The cycle-by-cycle BME operations are detailed in the following table.

**Table 17-3. Cycle definitions of decorated store: logical XOR**

| Pipeline Stage | Cycle  |  |  |
|----------------|--|--|--|
|                | x  | x+1  | x+2  |
| BME AHB_ap     | Forward addr to memory;<br>Decode decoration; Convert<br>master_wt to slave_rd;<br>Capture address, attributes | Recirculate captured addr +<br>attr to memory as slave_wt                                | <next>   |
| BME AHB_dp     | <previous>   | Perform memory read; Form<br>(rdata ^ wdata) and capture<br>destination data in register | Perform write sending<br>registered data to memory |

#### 17.4.1.4 Decorated Store Bit Field Insert (BFI)

This command inserts a bit field contained in the write data operand, defined by LSB position (b) and the bit field width (w+1), into the memory "container" defined by the access size associated with the store instruction using an atomic read-modify-write sequence.

The data size is specified by the write operation and can be byte (8-bit), halfword (16-bit) or word (32-bit). Note for the word sized operation, the maximum bit field width is 16 bits. The core performs the required write data lane replication on byte and halfword transfers.

The BFI operation can be used to insert a single bit into a peripheral. For this case, the w field is simply set to 0, indicating a bit field width of 1.

|        | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0        |          |   |
|--------|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|---|---|---|---|---|---|---|---|---|----------|----------|---|
| iobfib | 0  | 1  | 0  | 1  | -  | -  | b  | b  | b  | b  | -  | w  | w  | w  |    |    |    |    |    |    |    |    |   |   |   |   |   |   |   |   |   |          | mem_addr |   |
| iobfih | 0  | 1  | 0  | 1  | -  | b  | b  | b  | b  | w  | w  | w  | w  |    |    |    |    |    |    |    |    |    |   |   |   |   |   |   |   |   |   | mem_addr | 0        |   |
| iobfiw | 0  | 1  | 0  | 1  | b  | b  | b  | b  | b  | w  | w  | w  | w  |    |    |    |    |    |    |    |    |    |   |   |   |   |   |   |   |   |   | mem_addr | 0        | 0 |

**Figure 17-6. Decorated address store: bit field insert**

where  $\text{addr}[28] = 1$  signals a BFI operation,  $\text{addr}[27:23]$  is "b", the LSB identifier,  $\text{addr}[22:19]$  is "w", the bit field width minus 1 identifier, and  $\text{addr}[18:0]$  specifies the address offset into the peripheral space based at 0x4000\_0000. The "-" indicates an address bit "don't care". Note, unlike the other decorated store operations, BFI uses  $\text{addr}[19]$  as the least significant bit in the "w" specifier and not as an address bit.

The decorated BFI write operation is defined in the following pseudo-code as:

```
iobfi<sz>(accessAddress, wdata) // decorated bit field insert
tmp   = mem[accessAddress & 0xE007FFFF, size] // memory read
mask  = ((1 << (w+1)) - 1) << b // generate bit mask
```

## Functional Description

```
tmp = tmp & ~mask // modify
      | wdata & mask
mem[accessAddress & 0xE007FFFF, size] = tmp // memory write
```

The write data operand (wdata) associated with the store instruction contains the bit field to be inserted. It must be properly aligned within a right-justified container, that is, within the lower 8 bits for a byte operation, the lower 16 bits for a halfword or the entire 32 bits for a word operation.

To illustrate, consider the following example of the insertion of the 3-bit field "xyz" into an 8-bit memory container, initially set to "abcd\_efgh". For all cases, w is 2, signaling a bit field width of 3.

```
if b = 0 and the decorated store (strb) Rt register[7:0] = ----_xyz,
  then destination is "abcd_efxyz"
if b = 1 and the decorated store (strb) Rt register[7:0] = ----_xyz-,
  then destination is "abcd_xyzh"
if b = 2 and the decorated store (strb) Rt register[7:0] = ---x_yz--,
  then destination is "abcx_yzgh"
if b = 3 and the decorated store (strb) Rt register[7:0] = --xy_z---,
  then destination is "abxy_zfgh"
if b = 4 and the decorated store (strb) Rt register[7:0] = -xyz_-----
  then destination is "axyz_efgh"
if b = 5 and the decorated store (strb) Rt register[7:0] = xyz-_-----
  then destination is "xyzd_efgh"
if b = 6 and the decorated store (strb) Rt register[7:0] = yz--_-----
  then destination is "yzcd_efgh"
if b = 7 and the decorated store (strb) Rt register[7:0] = z---_-----
  then destination is "zbcd_efgh"
```

Note from the example, when the starting bit position plus the field width exceeds the container size, only part of the source bit field is inserted into the destination memory location. Stated differently, if  $(b + w+1) > \text{container\_width}$ , only the low-order " $\text{container\_width} - b$ " bits are actually inserted.

The cycle-by-cycle BME operations are detailed in the following table.

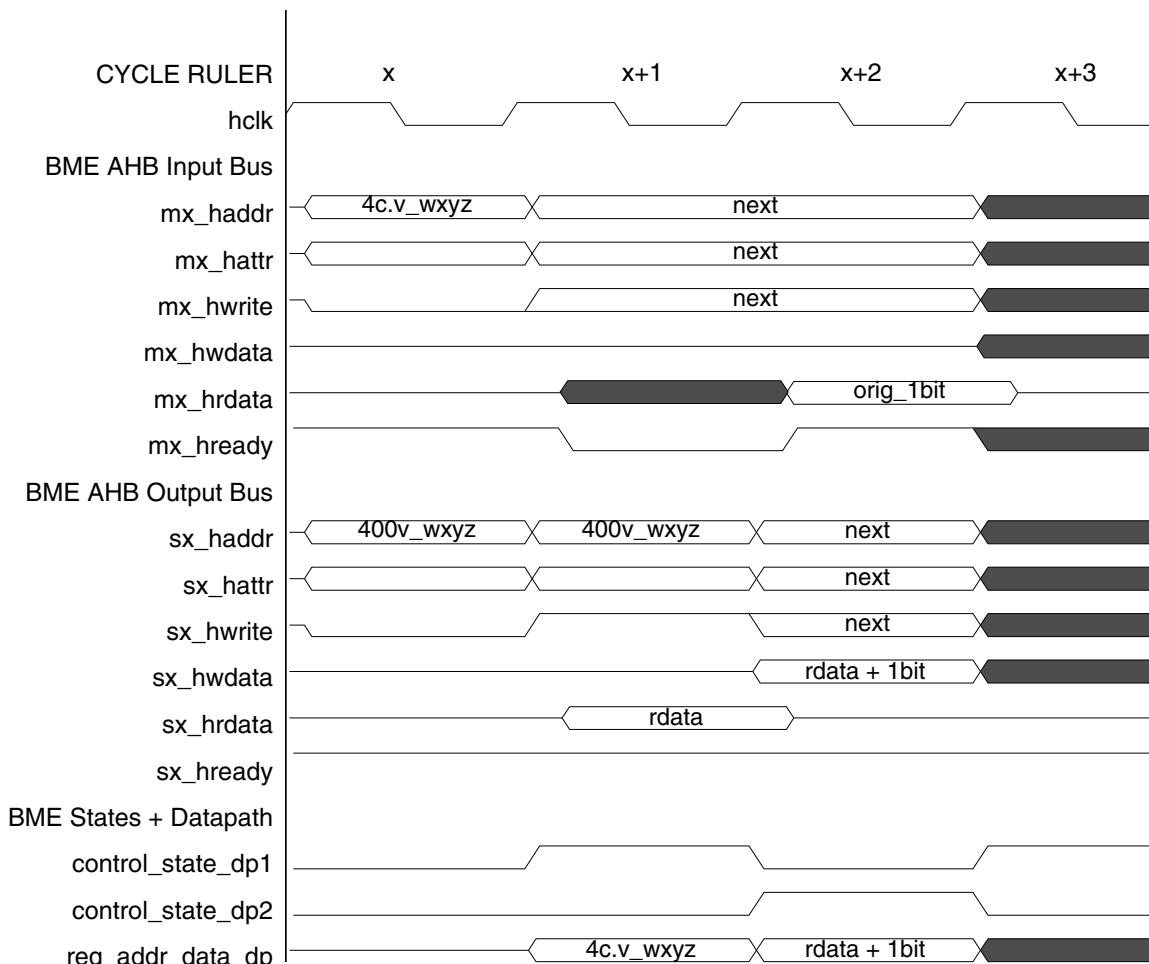
**Table 17-4. Cycle definitions of decorated store: bit field insert**

| Pipeline Stage | Cycle  |   |  |
|----------------|--|---|--|
|                | x  | x+1   | x+2  |
| BME AHB_ap     | Forward addr to memory;<br>Decode decoration; Convert<br>master_wt to slave_rd;<br>Capture address, attributes | Recirculate captured addr +<br>attr to memory as slave_wt   | <next>   |
| BME AHB_dp     | <previous>   | Perform memory read; Form<br>bit mask; Form bitwise<br>((mask) ? wdata : rdata)) and<br>capture destination data in<br>register | Perform write sending<br>registered data to memory |

## 17.4.2 BME Decorated Loads

The functions supported by the BME's decorated loads include two single-bit load-and-{set, clear} operators plus unsigned bit field extracts. For the two load-and-{set, clear} operations, BME converts a single decorated AHB load transaction into a 2-cycle atomic read-modify-write sequence, where the combined read-modify operations are performed in the first AHB data phase, and then the write is performed in the second AHB data phase as the original read data is returned to the processor core. For an unsigned bit field extract, the decorated load transaction is stalled for one cycle in the BME as the data field is extracted, then aligned and returned to the processor in the 2nd AHB data phase. This is the only decorated transaction that is not an atomic read-modify-write, as it is a simple data read.

A generic timing diagram of a decorated load showing a load-and-set 1-bit operation is shown as follows.



**Figure 17-7. Decorated load: load-and-set 1-bit field insert timing diagram**

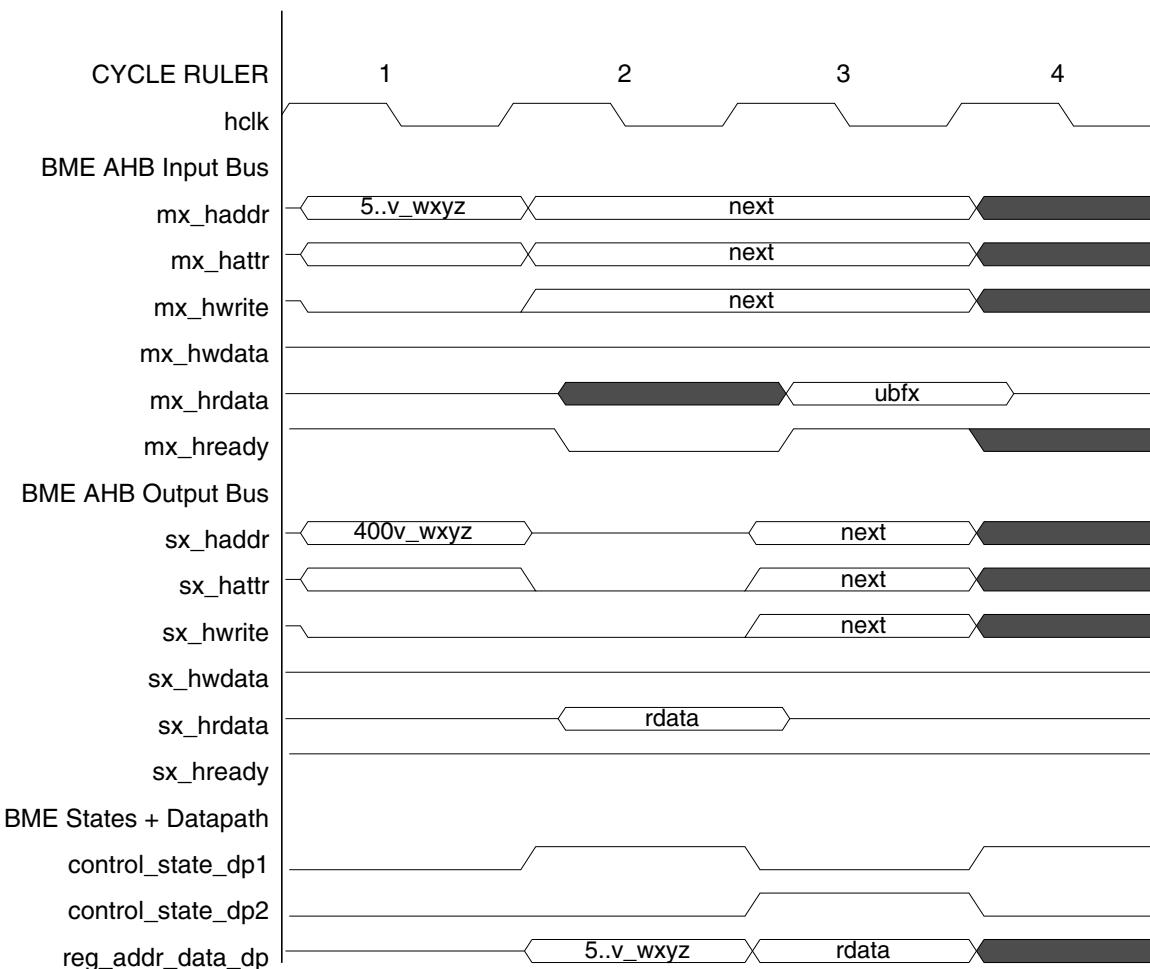
Decorated load-and-{set, clear} 1-bit operations follow the execution template shown in the above figure: a 2-cycle read-modify-write operation:

- Cycle x, 1st AHB address phase: Read from input bus is translated into a read operation on the output bus with the actual memory address (with the decoration removed) and then captured in a register
- Cycle x+1, 2nd AHB address phase: Write access with the registered (but actual) memory address is output
- Cycle x+1, 1st AHB data phase: The "original" 1-bit memory read data is captured in a register, while the 1-bit field is set or clear based on the function defined by the decoration with the modified data captured in a register; the input bus cycle is stalled
- Cycle x+2, 2nd AHB data phase: The selected original 1-bit is right justified, zero filled and then driven onto the input read data bus, while the registered write data is sourced onto the output write data bus

**NOTE**

Any wait states inserted by the peripheral slave device (`sx_hready = 0`) are simply passed through the BME back to the master input bus, stalling the AHB transaction cycle for cycle.

A generic timing diagram of a decorated load showing an unsigned bit field operation is shown in the following figure.



**Figure 17-8. Decorated load: unsigned bit field insert timing diagram**

The decorated unsigned bit field extract follows the same execution template shown in the above figure, a 2-cycle read operation:

- Cycle x, 1st AHB address phase: Read from input bus is translated into a read operation on the output bus with the actual memory address (with the decoration removed) and then captured in a register
- Cycle x+1, 2nd AHB address phase: Idle cycle
- Cycle x+1, 1st AHB data phase: A bit mask is generated based on the starting bit position and the field width; the mask is AND'ed with the memory read data to isolate the bit field; the resulting data is captured in a data register; the input bus cycle is stalled
- Cycle x+2, 2nd AHB data phase: Registered data is logically right shifted for proper alignment and driven onto the input read data bus

### NOTE

Any wait states inserted by the peripheral slave device ( $sx\_hready = 0$ ) are simply passed through the BME back to the master input bus, stalling the AHB transaction cycle for cycle.

### 17.4.2.1 Decorated Load Load-and-Clear 1 Bit (LAC1)

This command loads a 1-bit field defined by the LSB position (b) into the core's general purpose destination register (Rt) and zeroes the bit in the memory space after performing an atomic read-modify-write sequence.

The extracted one bit data field from the memory address is right justified and zero filled in the operand returned to the core.

The data size is specified by the read operation and can be byte (8-bit), halfword (16-bit) or word (32-bit).

|         | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---------|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|---|---|---|---|---|---|---|---|---|---|
| iolac1b | 0  | 1  | 0  | 0  | 1  | 0  | -  | -  | b  | b  | b  | -  |    |    |    |    |    |    |    |    |    |    |   |   |   |   |   |   |   |   |   |   |
| iolac1h | 0  | 1  | 0  | 0  | 1  | 0  | -  | b  | b  | b  | b  | -  |    |    |    |    |    |    |    |    |    |    |   |   |   |   |   |   |   | 0 |   |   |
| iolac1w | 0  | 1  | 0  | 0  | 1  | 0  | b  | b  | b  | b  | b  | -  |    |    |    |    |    |    |    |    |    |    |   |   |   |   |   |   | 0 | 0 |   |   |

**Figure 17-9. Decorated load address: load-and-clear 1 bit**

where  $\text{addr}[28:26] = 010$  specifies the load-and-clear 1 bit operation,  $\text{addr}[25:21]$  is "b", the bit identifier, and  $\text{mem\_addr}[19:0]$  specifies the address offset into the peripheral space based at 0x4000\_0000. The "-" indicates an address bit "don't care".

The decorated Load-and-Clear 1 Bit read operation is defined in the following pseudo-code as:

```
rdata = iolac1<sz>(accessAddress)           // decorated load-and-clear 1
tmp   = mem[accessAddress & 0xE00FFFFF, size] // memory read
mask  = 1 << b                            // generate bit mask
rdata = (tmp & mask) >> b                  // read data returned to core
tmp   = tmp & ~mask                          // modify
mem[accessAddress & 0xE00FFFFF, size] = tmp // memory write
```

The cycle-by-cycle BME operations are detailed in the following table.

**Table 17-5. Cycle definitions of decorated load: load-and-clear 1 bit**

| Pipeline Stage | Cycle   |   |  |
|----------------|---|---|--|
|                | x   | x+1   | x+2  |
| BME AHB_ap     | Forward addr to memory;<br>Decode decoration; Capture address, attributes | Recirculate captured addr + attr to memory as slave_wt  | <next>   |
| BME AHB_dp     | <previous>  | Perform memory read; Form bit mask; Extract bit from rdata; Form (rdata & ~mask) and capture destination data in register | Return extracted bit to master;<br>Perform write sending registered data to memory |

### 17.4.2.2 Decorated Load: Load-and-Set 1 Bit (LAS1)

This command loads a 1-bit field defined by the LSB position (b) into the core's general purpose destination register (Rt) and sets the bit in the memory space after performing an atomic read-modify-write sequence.

The extracted one bit data field from the memory address is right justified and zero filled in the operand returned to the core.

The data size is specified by the read operation and can be byte (8-bit), halfword (16-bit) or word (32-bit).

|         | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---------|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|---|---|---|---|---|---|---|---|---|---|
| iolas1b | 0  | 1  | 0  | 0  | 1  | 1  | -  | -  | b  | b  | b  | -  |    |    |    |    |    |    |    |    |    |    |   |   |   |   |   |   |   |   |   |   |
| iolas1h | 0  | 1  | 0  | 0  | 1  | 1  | -  | b  | b  | b  | b  | -  |    |    |    |    |    |    |    |    |    |    |   |   |   |   |   |   |   |   | 0 |   |
| iolas1w | 0  | 1  | 0  | 0  | 1  | 1  | b  | b  | b  | b  | b  | -  |    |    |    |    |    |    |    |    |    |    |   |   |   |   |   |   |   | 0 | 0 |   |

**Figure 17-10. Decorated load address: load-and-set 1 bit**

where  $\text{addr}[28:26] = 011$  specifies the load-and-set 1 bit operation,  $\text{addr}[25:21]$  is "b", the bit identifier, and  $\text{mem\_addr}[19:0]$  specifies the address offset into the peripheral space based at 0x4000\_0000. The "-" indicates an address bit "don't care".

The decorated Load-and-Set 1 Bit read operation is defined in the following pseudo-code as:

```
rdata = iolas1<sz>(accessAddress)           // decorated load-and-set 1
tmp   = mem[accessAddress & 0xE00FFFFF, size] // memory read
mask  = 1 << b                            // generate bit mask
rdata = (tmp & mask) >> b                  // read data returned to core
tmp   = tmp | mask                          // modify
mem[accessAddress & 0xE00FFFFF, size] = tmp // memory write
```

The cycle-by-cycle BME operations are detailed in the following table.

**Table 17-6. Cycle definitions of decorated load: load-and-set 1-bit**

| Pipeline Stage | Cycle   |  |  |
|----------------|---|--|--|
|                | x   | x+1  | x+2  |
| BME AHB_ap     | Forward addr to memory;<br>Decode decoration; Capture address, attributes | Recirculate captured addr + attr to memory as slave_wt   | <next>   |
| BME AHB_dp     | <previous>  | Perform memory read; Form bit mask; Extract bit from rdata; Form (rdata   mask) and capture destination data in register | Return extracted bit to master;<br>Perform write sending registered data to memory |

### 17.4.2.3 Decorated Load Unsigned Bit Field Extract (UBFX)

This command extracts a bit field defined by LSB position (b) and the bit field width (w+1) from the memory "container" defined by the access size associated with the load instruction using a 2-cycle read sequence.

The extracted bit field from the memory address is right justified and zero filled in the operand returned to the core. Recall this is the only decorated operation that does not perform a memory write, that is, UBX only performs a read.

The data size is specified by the write operation and can be byte (8-bit), halfword (16-bit) or word (32-bit). Note for the word sized operation, the maximum bit field width is 16 bits.

The use of a UBX operation is recommended to extract a single bit from a peripheral. For this case, the w field is simply set to 0, indicating a bit field width of 1.

|         | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---------|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|---|---|---|---|---|---|---|---|---|---|
| ioubfxb | 0  | 1  | 0  | 1  | -  | -  | b  | b  | b  | -  | w  | w  | w  |    |    |    |    |    |    |    |    |    |   |   |   |   |   |   |   |   |   |   |
| ioubfxh | 0  | 1  | 0  | 1  | -  | b  | b  | b  | b  | w  | w  | w  | w  |    |    |    |    |    |    |    |    |    |   |   |   |   |   |   |   |   | 0 |   |
| ioubfxw | 0  | 1  | 0  | 1  | b  | b  | b  | b  | b  | w  | w  | w  | w  |    |    |    |    |    |    |    |    |    |   |   |   |   |   |   |   | 0 | 0 |   |

Figure 17-11. Decorated load address: unsigned bit field extract

where addr[28] = 1 specifies the unsigned bit field extract operation, addr[27:23] is "b", the LSB identifier, addr[22:19] is "w", the bit field width minus 1 identifier, and mem\_addr[18:0] specifies the address offset into the peripheral space based at 0x4000\_0000. The "-" indicates an address bit "don't care". Note, unlike the other decorated load operations, UBX uses addr[19] as the least significant bit in the "w" specifier and not as an address bit.

The decorated unsigned bit field extract read operation is defined in the following pseudo-code as:

```
rdata = ioubfx<sz>(accessAddress) // unsigned bit field extract
tmp   = mem[accessAddress & 0xE007FFFF, size] // memory read
mask  = ((1 << (w+1)) - 1) << b // generate bit mask
rdata = (tmp & mask) >> b // read data returned to core
```

Like the BFI operation, when the starting bit position plus the field width exceeds the container size, only part of the source bit field is extracted from the destination memory location. Stated differently, if  $(b + w+1) > \text{container\_width}$ , only the low-order "container\_width - b" bits are actually extracted. The cycle-by-cycle BME operations are detailed in the following table.

**Table 17-7. Cycle definitions of decorated load: unsigned bit field extract**

| Pipeline Stage | Cycle  |   |   |
|----------------|--|---|---|
|                | x  | x+1   | x+2   |
| BME AHB_ap     | Forward addr to memory;<br>Decode decoration; Capture<br>address, attributes | Idle AHB address phase  | <next>  |
| BME AHB_dp     | <previous>   | Perform memory read; Form<br>bit mask; Form (rdata & mask)<br>and capture destination data<br>in register | Logically right shift registered<br>data; Return justified rdata to<br>master |

### 17.4.3 Additional Details on Decorated Addresses and GPIO Accesses

As previously noted, the peripheral address space occupies a 516 KB region: 512 KB based at 0x4000\_0000 plus a 4 KB space based at 0x400F\_F000 for GPIO accesses. This memory layout provides compatibility with the Kinetis K Family and provides 129 address "slots", each 4 KB in size.

The GPIO address space is multiply-mapped by the hardware: it appears at the "standard" system address 0x400F\_F000 and is physically located in the address slot corresponding to address 0x4000\_F000. Decorated loads and stores create a slight complication involving accesses to the GPIO. Recall the use of address[19] varies by decorated operation; for AND, OR, XOR, LAC1 and LAS1, this bit functions as a true address bit, while for BFI and UBFX, this bit defines the least significant bit of the "w" bit field specifier.

As a result, undecorated GPIO references and decorated AND, OR, XOR, LAC1 and LAS1 operations can use the standard 0x400F\_F000 base address, while decorated BFI and UBFX operations must use the alternate 0x4000\_F000 base address. Another implementation can simply use 0x400F\_F000 as the base address for all undecorated GPIO accesses and 0x4000\_F000 as the base address for all decorated accesses. Both implementations are supported by the hardware.

**Table 17-8. Decorated peripheral and GPIO address details**

Peripheral address space

**Table 17-8. Decorated peripheral and GPIO address details (continued)**

| Peripheral address space  | Description  |
|---------------------------|--|
| 0x4400_0000 - 0x4FFF_FFFF | Decorated AND, OR, XOR, LAC1, LAS1 references to peripherals and GPIO based at either 0x4000_F000 or 0x400F_F000 |
| 0x5000_0000 - 0x5FFF_FFFF | Decorated BFI, UBX references to peripherals and GPIO only based at 0x4000_F000                                  |

## 17.5 Application Information

In this section, GNU assembler macros with C expression operands are presented as examples of the required instructions to perform decorated operations. This section specifically presents a partial bme.h file defining the assembly language expressions for decorated logical stores: AND, OR and XOR. Comparable functions for BFI and the decorated loads are more complex and available in the complete BME header file.

These macros use the same function names presented in [Functional Description](#).

```
#define IOANDW(ADDR, WDATA) \
    __asm("ldr    r3, =(1<<26); " \
          "orr    r3, %[addr]; " \
          "mov    r2, %[wdata]; " \
          "str    r2, [r3]; " \
          :: [addr] "r" (ADDR), [wdata] "r" (WDATA) : "r2", "r3");

#define IOANDH(ADDR, WDATA) \
    __asm("ldr    r3, =(1<<26); " \
          "orr    r3, %[addr]; " \
          "mov    r2, %[wdata]; " \
          "strh   r2, [r3]; " \
          :: [addr] "r" (ADDR), [wdata] "r" (WDATA) : "r2", "r3");

#define IOANDB(ADDR, WDATA) \
    __asm("ldr    r3, =(1<<26); " \
          "orr    r3, %[addr]; " \
          "mov    r2, %[wdata]; " \
          "strb   r2, [r3]; " \
          :: [addr] "r" (ADDR), [wdata] "r" (WDATA) : "r2", "r3");

#define IOORW(ADDR, WDATA) \
    __asm("ldr    r3, =(1<<27); " \
          "orr    r3, %[addr]; " \
          "mov    r2, %[wdata]; " \
          "str    r2, [r3]; " \
          :: [addr] "r" (ADDR), [wdata] "r" (WDATA) : "r2", "r3");

#define IOORH(ADDR, WDATA) \
    __asm("ldr    r3, =(1<<27); " \
          "orr    r3, %[addr]; " \
          "mov    r2, %[wdata]; " \
          "strh   r2, [r3]; " \
          :: [addr] "r" (ADDR), [wdata] "r" (WDATA) : "r2", "r3");

#define IOORB(ADDR, WDATA) \
    __asm("ldr    r3, =(1<<27); " \
          "orr    r3, %[addr]; " \
          "mov    r2, %[wdata]; " \
          "strb   r2, [r3]; " \
          :: [addr] "r" (ADDR), [wdata] "r" (WDATA) : "r2", "r3");
```

```
:: [addr] "r" (ADDR), [wdata] "r" (WDATA) : "r2", "r3") ;\n\n#define IOXORW(ADDR,WDATA)          \
    __asm("ldr    r3, =(3<<26);"      \
          "orr    r3, %[addr];"        \
          "mov    r2, %[wdata];"       \
          "str    r2, [r3];"          \
    :: [addr] "r" (ADDR), [wdata] "r" (WDATA) : "r2", "r3") ;\n\n#define IOXORH(ADDR,WDATA)          \
    __asm("ldr    r3, =(3<<26);"      \
          "orr    r3, %[addr];"        \
          "mov    r2, %[wdata];"       \
          "strh   r2, [r3];"          \
    :: [addr] "r" (ADDR), [wdata] "r" (WDATA) : "r2", "r3") ;\n\n#define IOXORB(ADDR,WDATA)          \
    __asm("ldr    r3, =(3<<26);"      \
          "orr    r3, %[addr];"        \
          "mov    r2, %[wdata];"       \
          "strb   r2, [r3];"          \
    :: [addr] "r" (ADDR), [wdata] "r" (WDATA) : "r2", "r3");
```



# Chapter 18

## Miscellaneous Control Module (MCM)

### 18.1 Introduction

#### NOTE

For the chip-specific implementation details of this module's instances see the chip configuration information.

The Miscellaneous Control Module (MCM) provides a myriad of miscellaneous control functions.

#### 18.1.1 Features

The MCM includes the following features:

- Program-visible information on the platform configuration
- Crossbar master arbitration policy selection
- Flash controller speculation buffer and cache configurations

### 18.2 Memory map/register descriptions

The memory map and register descriptions below describe the registers using byte addresses.

**MCM memory map**

| Absolute address (hex) | Register name  | Width (in bits) | Access | Reset value | Section/page               |
|------------------------|--|-----------------|--------|-------------|----------------------------|
| F000_3008              | Crossbar Switch (AXBS) Slave Configuration (MCM_PLASC) | 16              | R      | 0007h       | <a href="#">18.2.1/292</a> |

*Table continues on the next page...*

**MCM memory map (continued)**

| Absolute address (hex) | Register name   | Width (in bits) | Access | Reset value | Section/page               |
|------------------------|---|-----------------|--------|-------------|----------------------------|
| F000_300A              | Crossbar Switch (AXBS) Master Configuration (MCM_PLAMC) | 16              | R      | 000Dh       | <a href="#">18.2.2/293</a> |
| F000_300C              | Platform Control Register (MCM_PLACR)                   | 32              | R/W    | 0000_0000h  | <a href="#">18.2.3/293</a> |
| F000_3040              | Compute Operation Control Register (MCM_CPO)            | 32              | R/W    | 0000_0000h  | <a href="#">18.2.4/296</a> |

**18.2.1 Crossbar Switch (AXBS) Slave Configuration (MCM\_PLASC)**

PLASC is a 16-bit read-only register identifying the presence/absence of bus slave connections to the device's crossbar switch.

Address: F000\_3000h base + 8h offset = F000\_3008h

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

**MCM\_PLASC field descriptions**

| Field            | Description   |
|------------------|---|
| 15–8<br>Reserved | This field is reserved.<br>This read-only field is reserved and always has the value 0.   |
| 7–0<br>ASC       | Each bit in the ASC field indicates whether there is a corresponding connection to the crossbar switch's slave input port.<br><br>0 A bus slave connection to AXBS input port <i>n</i> is absent<br>1 A bus slave connection to AXBS input port <i>n</i> is present |

## 18.2.2 Crossbar Switch (AXBS) Master Configuration (MCM\_PLAMC)

PLAMC is a 16-bit read-only register identifying the presence/absence of bus master connections to the device's crossbar switch.

Address: F000\_3000h base + Ah offset = F000\_300Ah

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

### MCM\_PLAMC field descriptions

| Field            | Description   |
|------------------|---|
| 15–8<br>Reserved | This field is reserved.<br>This read-only field is reserved and always has the value 0.   |
| 7–0<br>AMC       | Each bit in the AMC field indicates whether there is a corresponding connection to the AXBS master input port.<br><br>0 A bus master connection to AXBS input port <i>n</i> is absent<br>1 A bus master connection to AXBS input port <i>n</i> is present |

## 18.2.3 Platform Control Register (MCM\_PLACR)

The PLACR register selects the arbitration policy for the crossbar masters and configures the flash memory controller.

The speculation buffer and cache in the flash memory controller is configurable via MCM\_PLACR[15:10].

The speculation buffer is enabled only for instructions after reset. It is possible to have these states for the speculation buffer:

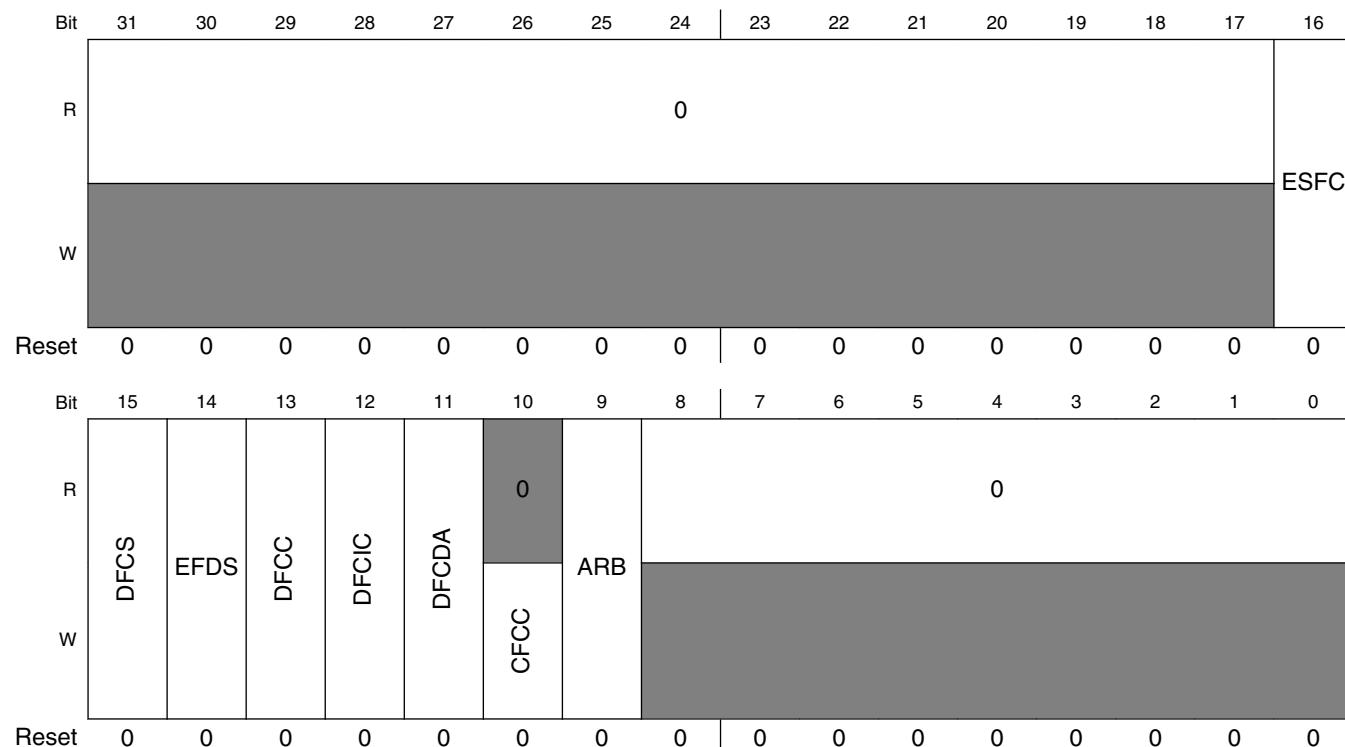
| DFCS | EFDS | Description  |
|------|------|--|
| 0    | 0    | Speculation buffer is on for instruction and off for data. |
| 0    | 1    | Speculation buffer is on for instruction and on for data.  |
| 1    | X    | Speculation buffer is off.                                 |

## Memory map/register descriptions

The cache in flash controller is enabled and caching both instruction and data type fetches after reset. It is possible to have these states for the cache:

| DFCC | DFCIC | DFCDA | Description                                   |
|------|-------|-------|---|
| 0    | 0     | 0     | Cache is on for both instruction and data.    |
| 0    | 0     | 1     | Cache is on for instruction and off for data. |
| 0    | 1     | 0     | Cache is off for instruction and on for data. |
| 0    | 1     | 1     | Cache is off for both instruction and data.   |
| 1    | X     | X     | Cache is off.                                 |

Address: F000\_3000h base + Ch offset = F000\_300Ch



## MCM\_PLACR field descriptions

| Field             | Description   |
|-------------------|---|
| 31–17<br>Reserved | This field is reserved.<br>This read-only field is reserved and always has the value 0.   |
| 16<br>ESFC        | Enable Stalling Flash Controller<br><br>This field is used to enable stalling flash controller when flash is busy.<br><br>0 Disable stalling flash controller when flash is busy.<br>1 Enable stalling flash controller when flash is busy. |

Table continues on the next page...

**MCM\_PLACR field descriptions (continued)**

| Field           | Description  |
|-----------------|--|
| 15<br>DFCS      | <p>Disable Flash Controller Speculation</p> <p>This field is used to disable flash controller speculation.</p> <p>0 Enable flash controller speculation.<br/>1 Disable flash controller speculation.</p>                                 |
| 14<br>EFDS      | <p>Enable Flash Data Speculation</p> <p>This field is used to enable flash data speculation.</p> <p>0 Disable flash data speculation.<br/>1 Enable flash data speculation.</p>   |
| 13<br>DFCC      | <p>Disable Flash Controller Cache</p> <p>This field is used to disable flash controller cache.</p> <p>0 Enable flash controller cache.<br/>1 Disable flash controller cache.</p>   |
| 12<br>DFCIC     | <p>Disable Flash Controller Instruction Caching</p> <p>This field is used to disable flash controller instruction caching.</p> <p>0 Enable flash controller instruction caching.<br/>1 Disable flash controller instruction caching.</p> |
| 11<br>DFCDA     | <p>Disable Flash Controller Data Caching</p> <p>This field is used to disable flash controller data caching.</p> <p>0 Enable flash controller data caching<br/>1 Disable flash controller data caching.</p>                              |
| 10<br>CFCC      | <p>Clear Flash Controller Cache</p> <p>Writing a 1 to this field clears the cache. Writing a 0 to this field is ignored. This field always reads as 0.</p>   |
| 9<br>ARB        | <p>Arbitration select</p> <p>0 Fixed-priority arbitration for the crossbar masters<br/>1 Round-robin arbitration for the crossbar masters</p>  |
| 8–0<br>Reserved | <p>This field is reserved.</p> <p>This read-only field is reserved and always has the value 0.</p>   |

## 18.2.4 Compute Operation Control Register (MCM\_CPO)

This register controls the Compute Operation.

Address: F000\_3000h base + 40h offset = F000\_3040h

| Bit   | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |   | 23 | 22 | 21 | 20 | 19 | 18    | 17     | 16     |
|-------|----|----|----|----|----|----|----|----|---|----|----|----|----|----|-------|--------|--------|
| R     |    |    |    |    |    |    |    |    | 0 |    |    |    |    |    |       |        |        |
| W     |    |    |    |    |    |    |    |    |   |    |    |    |    |    |       |        |        |
| Reset | 0  | 0  | 0  | 0  | 0  | 0  | 0  | 0  |   | 0  | 0  | 0  | 0  | 0  | 0     | 0      | 0      |
| Bit   | 15 | 14 | 13 | 12 | 11 | 10 | 9  | 8  |   | 7  | 6  | 5  | 4  | 3  | 2     | 1      | 0      |
| R     |    |    |    |    |    |    |    |    | 0 |    |    |    |    |    | CPOOI | CPOACK | CPOREQ |
| W     |    |    |    |    |    |    |    |    |   |    |    |    |    |    |       |        |        |
| Reset | 0  | 0  | 0  | 0  | 0  | 0  | 0  | 0  |   | 0  | 0  | 0  | 0  | 0  | 0     | 0      | 0      |

### MCM\_CPO field descriptions

| Field         | Description   |
|---------------|---|
| 31–3 Reserved | This field is reserved.<br>This read-only field is reserved and always has the value 0.   |
| 2 CPOWOI      | Compute Operation wakeup on interrupt<br>0 No effect.<br>1 When set, the CPOREQ is cleared on any interrupt or exception vector fetch.  |
| 1 CPOACK      | Compute Operation acknowledge<br>0 Compute operation entry has not completed or compute operation exit has completed.<br>1 Compute operation entry has completed or compute operation exit has not completed. |
| 0 CPOREQ      | Compute Operation request<br>This bit is auto-cleared by vector fetching if CPOWOI = 1.   |

*Table continues on the next page...*

**MCM\_CPO field descriptions (continued)**

| Field | Description                  |
|-------|------------------------------|
|       | 0 Request is cleared.        |
|       | 1 Request Compute Operation. |



# **Chapter 19**

## **Micro Trace Buffer (MTB)**

### **19.1 Introduction**

Microcontrollers using the Cortex-M0+ processor core include support for a CoreSight Micro Trace Buffer to provide program trace capabilities. The proper name for this function is the CoreSight Micro Trace Buffer for the Cortex-M0+ Processor; in this document, it is simply abbreviated as the MTB.

The simple program trace function creates instruction address change-of-flow data packets in a user-defined region of the system RAM. Accordingly, the system RAM controller manages requests from two sources:

- AMBA-AHB reads and writes from the system bus
- program trace packet writes from the processor

As part of the MTB functionality, there is a DWT (Data Watchpoint and Trace) module that allows the user to define watchpoint addresses, or optionally, an address and data value, that when triggered, can be used to start or stop the program trace recording.

This document details the functionality of both the MTB\_RAM and MTB\_DWT capabilities.

#### **19.1.1 Overview**

A generic block diagram of the processor core and platform for this class of ultra low-end microcontrollers is shown as follows:

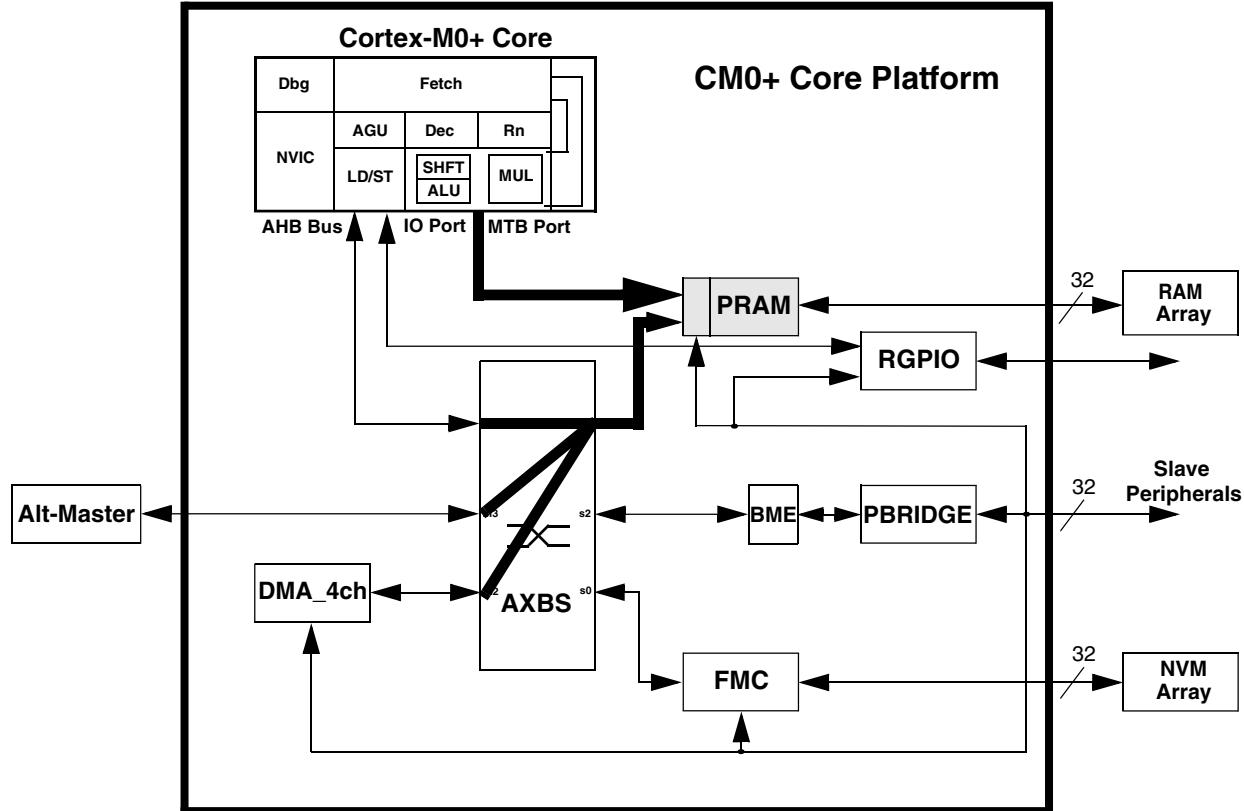


Figure 19-1. Generic Cortex-M0+ core platform block diagram

As shown in the block diagram, the platform RAM (PRAM) controller connects to two input buses:

- the crossbar slave port for system bus accesses
- a "private execution MTB port" from the core

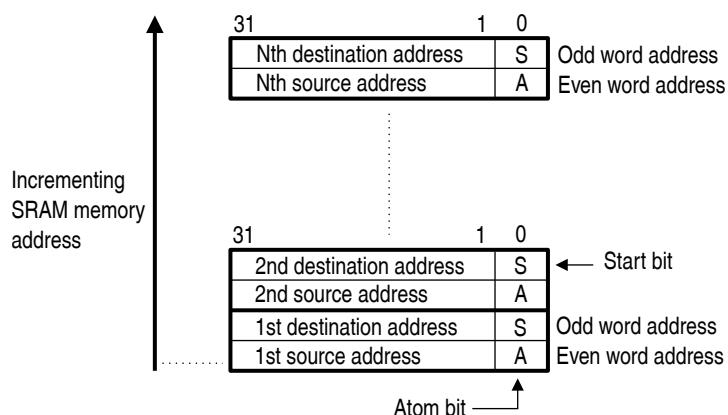
The logical paths from the crossbar master input ports to the PRAM controller are highlighted along with the private execution trace port from the processor core. The private MTB port signals the instruction address information needed for the 64-bit program trace packets written into the system RAM. The PRAM controller output interfaces to the attached RAM array. In this document, the PRAM controller is the MTB\_RAM controller.

The following information is taken from the ARM CoreSight Micro Trace Buffer documentation.

"The execution trace packet consists of a pair of 32-bit words that the MTB generates when it detects the processor PC value changes non-sequentially. A non-sequential PC change can occur during branch instructions or during exception entry."

The processor can cause a trace packet to be generated for any instruction.

The following figure shows how the execution trace information is stored in memory as a sequence of packets.



**Figure 19-2. MTB execution trace storage format**

The first, lower addressed, word contains the source of the branch, the address it branched from. The value stored only records bits[31:1] of the source address, because Thumb instructions are at least halfword aligned. The least significant bit of the value is the A-bit. The A-bit indicates the atomic state of the processor at the time of the branch, and can differentiate whether the branch originated from an instruction in a program, an exception, or a PC update in Debug state. When it is zero the branch originated from an instruction, when it is one the branch originated from an exception or PC update in Debug state. This word is always stored at an even word location.

The second, higher addressed word contains the destination of the branch, the address it branched to. The value stored only records bits[31:1] of the branch address. The least significant bit of the value is the S-bit. The S-bit indicates where the trace started. An S-bit value of 1 indicates where the first packet after the trace started and a value of 0 is used for other packets. Because it is possible to start and stop tracing multiple times in a trace session, the memory might contain several packets with the S-bit set to 1. This word is always stored in the next higher word in memory, an odd word address.

When the A-bit is set to 1, the source address field contains the architecturally-preferred return address for the exception. For example, if an exception was caused by an SVC instruction, then the source address field contains the address of the following instruction. This is different from the case where the A-bit is set to 0. In this case, the source address contains the address of the branch instruction.

For an exception return operation, two packets are generated:

- The first packet has the:
  - Source address field set to the address of the instruction that causes the exception return, BX or POP.

- Destination address field set to bits[31:1] of the EXC\_RETURN value. See the ARM v6-M Architecture Reference Manual.
- The A-bit set to 0.
- The second packet has the:
  - Source address field set to bits[31:1] of the EXC\_RETURN value.
  - Destination address field set to the address of the instruction where execution commences.
  - A-bit set to 1."

Given the recorded change-of-flow trace packets in system RAM and the memory image of the application, a debugger can read out the data and create an instruction-by-instruction program trace. In keeping with the low area and power implementation cost design targets, the MTB trace format is less efficient than other CoreSight trace modules, for example, the ETM (Embedded Trace Macrocell). Since each branch packet is 8 bytes in size, a 1 KB block of system RAM can contain 128 branches. Using the Dhrystone 2.1 benchmark's dynamic runtime as an example, this corresponds to about 875 instructions per KB of trace RAM, or with a zero wait state memory, this corresponds to approximately 1600 processor cycles per KB. This metric is obviously very sensitive to the runtime characteristics of the user code.

The MTB\_DWT function (not shown in the core platform block diagram) monitors the processor address and data buses so that configurable watchpoints can be detected to trigger the appropriate response in the MTB recording.

### 19.1.2 Features

The key features of the MTB\_RAM and MTB\_DWT include:

- Memory controller for system RAM and Micro Trace Buffer for program trace packets
- Read/write capabilities for system RAM accesses, write-only for program trace packets
- Supports zero wait state response to system bus accesses when no trace data is being written
- Can buffer two AHB address phases and one data write for system RAM accesses
- Supports 64-bit program trace packets including source and destination instruction addresses
- Program trace information in RAM available to MCU's application code or external debugger
- Program trace watchpoint configuration accessible by MCU's application code or debugger
- Location and size of RAM trace buffer is configured by software

- Two DWT comparators (addresses or address + data) provide programmable start/stop recording
- CoreSight compliant debug functionality

### 19.1.3 Modes of Operation

The MTB\_RAM and MTB\_DWT functions do not support any special modes of operation. The MTB\_RAM controller, as a memory-mapped device located on the platform's slave AHB system bus, responds based strictly on memory addresses for accesses to its attached RAM array. The MTB private execution bus provides program trace packet write information to the RAM controller. Both the MTB\_RAM and MTB\_DWT modules are memory-mapped so their programming models can be accessed.

All functionality associated with the MTB\_RAM and MTB\_DWT modules resides in the core platform's clock domain; this includes its connections with the RAM array.

## 19.2 External Signal Description

The MTB\_RAM and MTB\_DWT modules do not directly support any external interfaces.

The internal interfaces includes a standard AHB bus with a 32-bit datapath width from the appropriate crossbar slave port plus the private execution trace bus from the processor core. The signals in the private execution trace bus are detailed in the following table taken from the ARM CoreSight Micro Trace Buffer documentation. The signal direction is defined as viewed by the MTB\_RAM controller.

**Table 19-1. Private execution trace port from the core to MTB\_RAM**

| Signal     | Direction | Description  |
|------------|-----------|--|
| LOCKUP     | Input     | Indicates the processor is in the Lockup state. This signal is driven LOW for cycles when the processor is executing normally and driven HIGH for every cycle the processor is waiting in the Lockup state. This signal is valid on every cycle. |
| IAESEQ     | Input     | Indicates the next instruction address in execute, IAEX, is sequential, that is non-branching.   |
| IAEXEN     | Input     | IAEX register enable.  |
| IAEX[30:0] | Input     | Registered address of the instruction in the execution stage, shifted right by one bit, that is, PC >> 1.  |
| ATOMIC     | Input     | Indicates the processor is performing non-instruction related activities.  |
| EDBGRQ     | Output    | Request for the processor to enter the Debug state, if enabled, and halt.  |

In addition, there are two signals formed by the MTB\_DWT module and driven to the MTB\_RAM controller: TSTART (trace start) and TSTOP (trace stop). These signals can be configured using the trace watchpoints to define programmable addresses and data values to affect the program trace recording state.

## 19.3 Memory Map and Register Definition

The MTB\_RAM and MTB\_DWT modules each support a sparsely-populated 4 KB address space for their programming models. For each address space, there are a variety of control and configurable registers near the base address, followed by a large unused address space and finally a set of CoreSight registers to support dynamic determination of the debug configuration for the device.

Accesses to the programming model follow standard ARM conventions. Taken from the ARM CoreSight Micro Trace Buffer documentation, these are:

- Do not attempt to access reserved or unused address locations. Attempting to access these locations can result in UNPREDICTABLE behavior.
- The behavior of the MTB is UNPREDICTABLE if the registers with UNKNOWN reset values are not programmed prior to enabling trace.
- Unless otherwise stated in the accompanying text:
  - Do not modify reserved register bits
  - Ignore reserved register bits on reads
  - All register bits are reset to a logic 0 by a system or power-on reset
  - Use only word size, 32-bit, transactions to access all registers

### 19.3.1 MTB\_RAM Memory Map

**MTB memory map**

| Absolute address (hex) | Register name                                    | Width (in bits) | Access | Reset value | Section/ page               |
|------------------------|--|-----------------|--------|-------------|-----------------------------|
| F000_0000              | MTB Position Register (MTB_POSITION)             | 32              | R/W    | Undefined   | <a href="#">19.31.1/306</a> |
| F000_0004              | MTB Master Register (MTB_MASTER)                 | 32              | R/W    | 0000_0080h  | <a href="#">19.31.2/307</a> |
| F000_0008              | MTB Flow Register (MTB_FLOW)                     | 32              | R/W    | Undefined   | <a href="#">19.31.3/309</a> |
| F000_000C              | MTB Base Register (MTB_BASE)                     | 32              | R      | Undefined   | <a href="#">19.31.4/310</a> |
| F000_0F00              | Integration Mode Control Register (MTB_MODECTRL) | 32              | R      | 0000_0000h  | <a href="#">19.31.5/310</a> |
| F000_0FA0              | Claim TAG Set Register (MTB_TAGSET)              | 32              | R      | 0000_0000h  | <a href="#">19.31.6/311</a> |
| F000_0FA4              | Claim TAG Clear Register (MTB_TAGCLEAR)          | 32              | R      | 0000_0000h  | <a href="#">19.31.7/311</a> |
| F000_0FB0              | Lock Access Register (MTB_LOCKACCESS)            | 32              | R      | 0000_0000h  | <a href="#">19.31.8/312</a> |

*Table continues on the next page...*

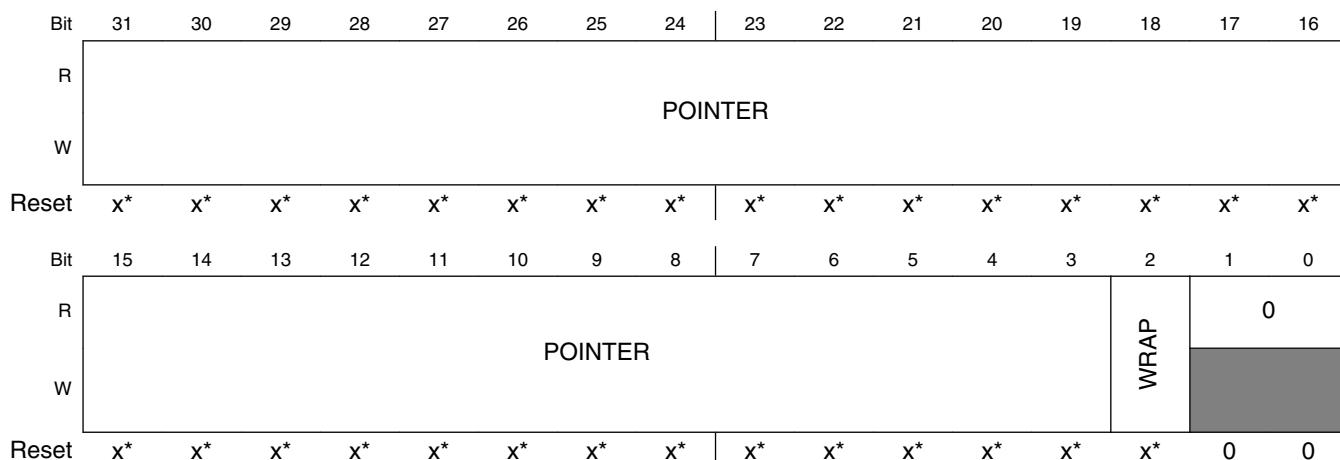
**MTB memory map (continued)**

| Absolute address (hex) | Register name                                     | Width (in bits) | Access | Reset value                 | Section/ page                |
|------------------------|---|-----------------|--------|-----------------------------|------------------------------|
| F000_0FB4              | Lock Status Register (MTB_LOCKSTAT)               | 32              | R      | 0000_0000h                  | <a href="#">19.31.9/312</a>  |
| F000_0FB8              | Authentication Status Register (MTB_AUTHSTAT)     | 32              | R      | 0000_0000h                  | <a href="#">19.31.10/313</a> |
| F000_0FBC              | Device Architecture Register (MTB_DEVICEARCH)     | 32              | R      | 4770_0A31h                  | <a href="#">19.31.11/314</a> |
| F000_0FC8              | Device Configuration Register (MTB_DEVICECFG)     | 32              | R      | 0000_0000h                  | <a href="#">19.31.12/314</a> |
| F000_0FCC              | Device Type Identifier Register (MTB_DEVICETYPID) | 32              | R      | 0000_0031h                  | <a href="#">19.31.13/315</a> |
| F000_0FD0              | Peripheral ID Register (MTB_PERIPHID4)            | 32              | R      | <a href="#">See section</a> | <a href="#">19.31.14/315</a> |
| F000_0FD4              | Peripheral ID Register (MTB_PERIPHID5)            | 32              | R      | <a href="#">See section</a> | <a href="#">19.31.14/315</a> |
| F000_0FD8              | Peripheral ID Register (MTB_PERIPHID6)            | 32              | R      | <a href="#">See section</a> | <a href="#">19.31.14/315</a> |
| F000_0FDC              | Peripheral ID Register (MTB_PERIPHID7)            | 32              | R      | <a href="#">See section</a> | <a href="#">19.31.14/315</a> |
| F000_0FE0              | Peripheral ID Register (MTB_PERIPHID0)            | 32              | R      | <a href="#">See section</a> | <a href="#">19.31.14/315</a> |
| F000_0FE4              | Peripheral ID Register (MTB_PERIPHID1)            | 32              | R      | <a href="#">See section</a> | <a href="#">19.31.14/315</a> |
| F000_0FE8              | Peripheral ID Register (MTB_PERIPHID2)            | 32              | R      | <a href="#">See section</a> | <a href="#">19.31.14/315</a> |
| F000_0FEC              | Peripheral ID Register (MTB_PERIPHID3)            | 32              | R      | <a href="#">See section</a> | <a href="#">19.31.14/315</a> |
| F000_0FF0              | Component ID Register (MTB_COMPID0)               | 32              | R      | <a href="#">See section</a> | <a href="#">19.31.15/316</a> |
| F000_0FF4              | Component ID Register (MTB_COMPID1)               | 32              | R      | <a href="#">See section</a> | <a href="#">19.31.15/316</a> |
| F000_0FF8              | Component ID Register (MTB_COMPID2)               | 32              | R      | <a href="#">See section</a> | <a href="#">19.31.15/316</a> |
| F000_0FFC              | Component ID Register (MTB_COMPID3)               | 32              | R      | <a href="#">See section</a> | <a href="#">19.31.15/316</a> |

### 19.31.1 MTB Position Register (MTB\_POSITION)

The MTB\_POSITION register is the trace write address pointer and wrap bit. This register can be modified by the explicit programming model writes. It is also automatically updated by the MTB hardware when trace packets are being recorded.

Address: F000\_0000h base + 0h offset = F000\_0000h



\* Notes:

- x = Undefined at reset.

#### MTB\_POSITION field descriptions

| Field           | Description   |
|-----------------|---|
| 31–3<br>POINTER | <p>Trace Packet Address Pointer</p> <p>Trace packet address pointer. Because a packet consists of two words, the POINTER field is the address of the first word of a packet. This field contains bits[31:3] of the RAM address that points to the next unused memory location for the trace data. This is an empty ascending location and is automatically updated.</p> <p>A debug agent can add the value of POINTER to the value of MTB_BASE to obtain the absolute pointer address as seen on the system AHB bus interface.</p> <p><b>NOTE:</b> The size of the RAM is parameterized and the most significant bits of the POINTER field are RAZ/WI.</p> <p>POSITION register bits greater than or equal to 15 are RAZ/WI, therefore, the active POINTER field bits are [11:0].</p> |
| 2<br>WRAP       | This bit is set to 1 automatically when the POINTER value wraps as determined by the MTB_MASTER[MASK] bit in the MASTER Trace Control Register. A debug agent can use the WRAP bit to determine whether the trace information above and below the pointer address is valid.   |
| 1–0<br>Reserved | This field is reserved.<br>This read-only field is reserved and always has the value 0.   |

## 19.31.2 MTB Master Register (MTB\_MASTER)

The MTB\_MASTER register contains the main program trace enable plus other trace controls. This register can be modified by the explicit programming model writes. MTB\_MASTER[EN] and MTB\_MASTER[HALTREQ] fields are also automatically updated by the MTB hardware.

Before the MTB\_MASTER[EN] or MTB\_MASTER[TSTARTEN] bits are set to 1, software must initialize the MTB\_POSITION and MTB\_FLOW registers.

If the MTB\_FLOW[WATERMARK] field is used to stop tracing or to halt the processor, the MTB\_MASTER[MASK] field must still be set to a value that prevents the MTB\_POSITION[POINTER] field from wrapping before it reaches the MTB\_FLOW[WATERMARK] value.

### NOTE

The format of this mask field is different than the MTBDWT\_MASKn[MASK].

Address: F000\_0000h base + 4h offset = F000\_0004h

| Bit       | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|-----------|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|
| R         |    |    |    |    |    |    |    |    | 0  |    |    |    |    |    |    |    |
| EN        |    |    |    |    |    |    |    |    |    |    |    |    |    |    |    |    |
| W         |    |    |    |    |    |    |    |    |    |    |    |    |    |    |    |    |
| Reset     | 0  | 0  | 0  | 0  | 0  | 0  | 0  | 0  | 0  | 0  | 0  | 0  | 0  | 0  | 0  | 0  |
| Bit       | 15 | 14 | 13 | 12 | 11 | 10 | 9  | 8  | 7  | 6  | 5  | 4  | 3  | 2  | 1  | 0  |
| R         |    |    |    |    |    |    |    |    | 0  |    |    |    |    |    |    |    |
|           |    |    |    |    |    |    |    |    |    |    |    |    |    |    |    |    |
| HALTREQ   |    |    |    |    |    |    |    |    |    |    |    |    |    |    |    |    |
| RAMPRIV   |    |    |    |    |    |    |    |    |    |    |    |    |    |    |    |    |
| SFRWPRIV  |    |    |    |    |    |    |    |    |    |    |    |    |    |    |    |    |
| TSTOPOPEN |    |    |    |    |    |    |    |    |    |    |    |    |    |    |    |    |
| TSTARTEN  |    |    |    |    |    |    |    |    |    |    |    |    |    |    |    |    |
| MASK      |    |    |    |    |    |    |    |    |    |    |    |    |    |    |    |    |
| Reset     | 0  | 0  | 0  | 0  | 0  | 0  | 0  | 0  | 1  | 0  | 0  | 0  | 0  | 0  | 0  | 0  |

### MTB\_MASTER field descriptions

| Field    | Description   |
|----------|---|
| 31<br>EN | Main trace enable bit<br><br>When this bit is 1, trace data is written into the RAM memory location addressed by MTB_POSITION[POINTER]. The MTB_POSITION[POINTER] value auto increments after the trace data packet is written.<br><br>The EN bit can be automatically set to 0 using the MTB_FLOW[WATERMARK] field and the MTB_FLOW[AUTOSTOP] bit.<br><br>The EN bit is automatically set to 1 if the TSTARTEN bit is 1 and the TSTART signal is HIGH.<br>The EN bit is automatically set to 0 if TSTOPOPEN bit is 1 and the TSTOP signal is HIGH. |

Table continues on the next page...

**MTB\_MASTER field descriptions (continued)**

| Field             | Description  |
|-------------------|--|
|                   | <b>NOTE:</b> If the EN bit is set to 0 because the MTB_FLOW[WATERMARK] field is set, then it is not automatically set to 1 if the TSTARTEN bit is 1 and the TSTART input is HIGH. In this case, tracing can only be restarted if the MTB_FLOW[WATERMARK] or MTB_POSITION[POSITION] value is changed by software.   |
| 30–10<br>Reserved | This field is reserved.<br>This read-only field is reserved and always has the value 0.  |
| 9<br>HALTREQ      | Halt request bit<br><br>This bit is connected to the halt request signal of the trace logic, EDBGREQ. When HALTREQ is set to 1, the EDBGREQ is asserted if DBGEN (invasive debug enable, one of the debug authentication interface signals) is also HIGH. The HALTREQ bit can be automatically set to 1 using the MTB_FLOW[WATERMARK] field.   |
| 8<br>RAMPRIV      | RAM privilege bit<br><br>If this bit is 0, then user or privileged AHB read and write accesses to the RAM are permitted. If this bit is 1, then only privileged AHB read and write accesses to the RAM are permitted and user accesses are RAZ/WI. The HPROT[1] signal determines if an access is a user or privileged mode reference.   |
| 7<br>SFRWPRI      | Special Function Register Write Privilege bit<br><br>If this bit is 0, then user or privileged AHB read and write accesses to the MTB_RAM Special Function Registers (programming model) are permitted. If this bit is 1, then only privileged write accesses are permitted; user write accesses are ignored. The HPROT[1] signal determines if an access is user or privileged. Note MTB_RAM SFR read access are not controlled by this bit and are always permitted.   |
| 6<br>TSTOPEN      | Trace stop input enable<br><br>If this bit is 1 and the TSTOP signal is HIGH, then the EN bit is set to 0. If a trace packet is being written to memory, the write is completed before tracing is stopped.   |
| 5<br>TSTARTEN     | Trace start input enable<br><br>If this bit is 1 and the TSTART signal is HIGH, then the EN bit is set to 1. Tracing continues until a stop condition occurs.  |
| 4–0<br>MASK       | Mask<br><br>This value determines the maximum size of the trace buffer in RAM. It specifies the most-significant bit of the MTB_POSITION[POSITION] field that can be updated by automatic increment. If the trace tries to advance past this power of two, the MTB_POSITION[WRAP] bit is set to 1, the MTB_POSITION[POSITION[MASK:0]] bits are set to zero, and the MTB_POSITION[POSITION[11:MASK+1]] bits remain unchanged.<br><br>This field causes the trace packet information to be stored in a circular buffer of size $2^{[MASK+4]}$ bytes, that can be positioned in memory at multiples of this size. Valid values of this field are zero to 11. Values greater than the maximum have the same effect as the maximum. |

### **19.31.3 MTB Flow Register (MTB\_FLOW)**

The MTB\_FLOW register contains the watermark address and the autostop/autohalt

### 19.31.4 MTB Base Register (MTB\_BASE)

The read-only MTB\_BASE Register indicates where the RAM is located in the processor memory map. This register is provided to enable auto discovery of the MTB RAM location, by a debug agent and is defined by a hardware design parameter. For these devices, the base address is defined by the expression:  $MTB\_BASE[BASEADDR] = 0x2000_0000 - (\text{RAM\_Size}/4)$

Address: F000\_0000h base + Ch offset = F000\_000Ch

| Bit | 31       | 30 | 29 | 28 | 27 | 26 | 25 | 24 | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|-----|----------|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|---|---|---|---|---|---|---|---|---|---|
| R   |          |    |    |    |    |    |    |    |    |    |    |    |    |    |    |    |    |    |    |    |    |    |   |   |   |   |   |   |   |   |   |   |
|     | BASEADDR |    |    |    |    |    |    |    |    |    |    |    |    |    |    |    |    |    |    |    |    |    |   |   |   |   |   |   |   |   |   |   |
| W   |          |    |    |    |    |    |    |    |    |    |    |    |    |    |    |    |    |    |    |    |    |    |   |   |   |   |   |   |   |   |   |   |

Reset x\* | x\* x\*

\* Notes:

- x = Undefined at reset.

#### MTB\_BASE field descriptions

| Field            | Description  |
|------------------|--|
| 31–0<br>BASEADDR | This value is defined with a hardwired signal and the expression: $0x2000_0000 - (\text{RAM\_Size}/4)$ . For example, if the total RAM capacity is 16 KB, this field is 0xFFFF_F000. |

### 19.31.5 Integration Mode Control Register (MTB\_MODECTRL)

This register enables the device to switch from a functional mode, or default behavior, into integration mode. It is hardwired to specific values used during the auto-discovery process by an external debug agent.

Address: F000\_0000h base + F00h offset = F000\_OF00h

| Bit | 31       | 30 | 29 | 28 | 27 | 26 | 25 | 24 | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|-----|----------|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|---|---|---|---|---|---|---|---|---|---|
| R   |          |    |    |    |    |    |    |    |    |    |    |    |    |    |    |    |    |    |    |    |    |    |   |   |   |   |   |   |   |   |   |   |
|     | MODECTRL |    |    |    |    |    |    |    |    |    |    |    |    |    |    |    |    |    |    |    |    |    |   |   |   |   |   |   |   |   |   |   |
| W   |          |    |    |    |    |    |    |    |    |    |    |    |    |    |    |    |    |    |    |    |    |    |   |   |   |   |   |   |   |   |   |   |

#### MTB\_MODECTRL field descriptions

| Field            | Description              |
|------------------|--------------------------|
| 31–0<br>MODECTRL | Hardwired to 0x0000_0000 |

### 19.31.6 Claim TAG Set Register (MTB\_TAGSET)

The Claim Tag Set Register returns the number of bits that can be set on a read, and enables individual bits to be set on a write. It is hardwired to specific values used during the auto-discovery process by an external debug agent.

Address: F000\_0000h base + FA0h offset = F000\_0FA0h

| Bit   | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|-------|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|---|---|---|---|---|---|---|---|---|---|
| R     |    |    |    |    |    |    |    |    |    |    |    |    |    |    |    |    |    |    |    |    |    |    |   |   |   |   |   |   |   |   |   |   |
| W     |    |    |    |    |    |    |    |    |    |    |    |    |    |    |    |    |    |    |    |    |    |    |   |   |   |   |   |   |   |   |   |   |
| Reset | 0  | 0  | 0  | 0  | 0  | 0  | 0  | 0  | 0  | 0  | 0  | 0  | 0  | 0  | 0  | 0  | 0  | 0  | 0  | 0  | 0  | 0  | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |   |

#### MTB\_TAGSET field descriptions

| Field          | Description              |
|----------------|--------------------------|
| 31–0<br>TAGSET | Hardwired to 0x0000_0000 |

### 19.31.7 Claim TAG Clear Register (MTB\_TAGCLEAR)

The read/write Claim Tag Clear Register is used to read the claim status on debug resources. A read indicates the claim tag status. Writing 1 to a specific bit clears the corresponding claim tag to 0. It is hardwired to specific values used during the auto-discovery process by an external debug agent.

Address: F000\_0000h base + FA4h offset = F000\_0FA4h

| Bit   | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|-------|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|---|---|---|---|---|---|---|---|---|---|
| R     |    |    |    |    |    |    |    |    |    |    |    |    |    |    |    |    |    |    |    |    |    |    |   |   |   |   |   |   |   |   |   |   |
| W     |    |    |    |    |    |    |    |    |    |    |    |    |    |    |    |    |    |    |    |    |    |    |   |   |   |   |   |   |   |   |   |   |
| Reset | 0  | 0  | 0  | 0  | 0  | 0  | 0  | 0  | 0  | 0  | 0  | 0  | 0  | 0  | 0  | 0  | 0  | 0  | 0  | 0  | 0  | 0  | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |   |

#### MTB\_TAGCLEAR field descriptions

| Field            | Description              |
|------------------|--------------------------|
| 31–0<br>TAGCLEAR | Hardwired to 0x0000_0000 |

### 19.31.8 Lock Access Register (MTB\_LOCKACCESS)

The Lock Access Register enables a write access to component registers. It is hardwired to specific values used during the auto-discovery process by an external debug agent.

Address: F000\_0000h base + FB0h offset = F000\_0FB0h

| Bit   | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|-------|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|---|---|---|---|---|---|---|---|---|---|
| R     |    |    |    |    |    |    |    |    |    |    |    |    |    |    |    |    |    |    |    |    |    |    |   |   |   |   |   |   |   |   |   |   |
| W     |    |    |    |    |    |    |    |    |    |    |    |    |    |    |    |    |    |    |    |    |    |    |   |   |   |   |   |   |   |   |   |   |
| Reset | 0  | 0  | 0  | 0  | 0  | 0  | 0  | 0  | 0  | 0  | 0  | 0  | 0  | 0  | 0  | 0  | 0  | 0  | 0  | 0  | 0  | 0  | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |   |

#### MTB\_LOCKACCESS field descriptions

| Field              | Description              |
|--------------------|--------------------------|
| 31–0<br>LOCKACCESS | Hardwired to 0x0000_0000 |

### 19.31.9 Lock Status Register (MTB\_LOCKSTAT)

The Lock Status Register indicates the status of the lock control mechanism. This register is used in conjunction with the Lock Access Register. It is hardwired to specific values used during the auto-discovery process by an external debug agent.

Address: F000\_0000h base + FB4h offset = F000\_0FB4h

| Bit   | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|-------|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|---|---|---|---|---|---|---|---|---|---|
| R     |    |    |    |    |    |    |    |    |    |    |    |    |    |    |    |    |    |    |    |    |    |    |   |   |   |   |   |   |   |   |   |   |
| W     |    |    |    |    |    |    |    |    |    |    |    |    |    |    |    |    |    |    |    |    |    |    |   |   |   |   |   |   |   |   |   |   |
| Reset | 0  | 0  | 0  | 0  | 0  | 0  | 0  | 0  | 0  | 0  | 0  | 0  | 0  | 0  | 0  | 0  | 0  | 0  | 0  | 0  | 0  | 0  | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |   |

#### MTB\_LOCKSTAT field descriptions

| Field            | Description              |
|------------------|--------------------------|
| 31–0<br>LOCKSTAT | Hardwired to 0x0000_0000 |

## 19.31.10 Authentication Status Register (MTB\_AUTHSTAT)

The Authentication Status Register reports the required security level and current status of the security enable bit pairs. Where functionality changes on a given security level, this change must be reported in this register. It is connected to specific signals used during the auto-discovery process by an external debug agent.

MTB\_AUTHSTAT[3:2] indicates if nonsecure, noninvasive debug is enabled or disabled, while MTB\_AUTHSTAT[1:0] indicates the enabled/disabled state of nonsecure, invasive debug. For both 2-bit fields, 0b10 indicates the functionality is disabled and 0b11 indicates it is enabled.

Address: F000\_0000h base + FB8h offset = F000\_0FB8h

| Bit   | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 | 23 | 22 | 21 | 20 | 19 | 18   | 17 | 16   |
|-------|----|----|----|----|----|----|----|----|----|----|----|----|----|------|----|------|
| R     |    |    |    |    |    |    |    |    | 0  |    |    |    |    |      |    |      |
| W     |    |    |    |    |    |    |    |    |    |    |    |    |    |      |    |      |
| Reset | 0  | 0  | 0  | 0  | 0  | 0  | 0  | 0  | 0  | 0  | 0  | 0  | 0  | 0    | 0  | 0    |
| Bit   | 15 | 14 | 13 | 12 | 11 | 10 | 9  | 8  | 7  | 6  | 5  | 4  | 3  | 2    | 1  | 0    |
| R     |    |    |    |    |    |    |    | 0  |    |    |    |    | 1  | BIT2 | 1  | BIT0 |
| W     |    |    |    |    |    |    |    |    |    |    |    |    |    |      |    |      |
| Reset | 0  | 0  | 0  | 0  | 0  | 0  | 0  | 0  | 0  | 0  | 0  | 0  | 0  | 0    | 0  | 0    |

### MTB\_AUTHSTAT field descriptions

| Field            | Description   |
|------------------|---|
| 31–4<br>Reserved | This field is reserved.<br>This read-only field is reserved and always has the value 0. |
| 3<br>Reserved    | This read-only field is reserved and always has the value 1.                            |
| 2<br>BIT2        | Connected to NIDEN or DBGEN signal.   |
| 1<br>Reserved    | This read-only field is reserved and always has the value 1.                            |
| 0<br>BIT0        | Connected to DBGEN.   |

### 19.31.11 Device Architecture Register (MTB\_DEVICEARCH)

This register indicates the device architecture. It is hardwired to specific values used during the auto-discovery process by an external debug agent.

Address: F000\_0000h base + FBCh offset = F000\_0FBCh

| Bit   | 31         | 30 | 29 | 28 | 27 | 26 | 25 | 24 | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|-------|------------|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|---|---|---|---|---|---|---|---|---|---|
| R     | DEVICEARCH |    |    |    |    |    |    |    |    |    |    |    |    |    |    |    |    |    |    |    |    |    |   |   |   |   |   |   |   |   |   |   |
| W     |            |    |    |    |    |    |    |    |    |    |    |    |    |    |    |    |    |    |    |    |    |    |   |   |   |   |   |   |   |   |   |   |
| Reset | 0          | 1  | 0  | 0  | 0  | 1  | 1  | 0  | 1  | 1  | 0  | 0  | 0  | 0  | 0  | 0  | 0  | 0  | 0  | 1  | 0  | 1  | 0 | 0 | 0 | 1 | 1 | 0 | 0 | 0 | 1 |   |

#### MTB\_DEVICEARCH field descriptions

| Field              | Description               |
|--------------------|---------------------------|
| 31–0<br>DEVICEARCH | Hardwired to 0x4770_0A31. |

### 19.31.12 Device Configuration Register (MTB\_DEVICECFG)

This register indicates the device configuration. It is hardwired to specific values used during the auto-discovery process by an external debug agent.

Address: F000\_0000h base + FC8h offset = F000\_0FC8h

| Bit   | 31        | 30 | 29 | 28 | 27 | 26 | 25 | 24 | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|-------|-----------|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|---|---|---|---|---|---|---|---|---|---|
| R     | DEVICECFG |    |    |    |    |    |    |    |    |    |    |    |    |    |    |    |    |    |    |    |    |    |   |   |   |   |   |   |   |   |   |   |
| W     |           |    |    |    |    |    |    |    |    |    |    |    |    |    |    |    |    |    |    |    |    |    |   |   |   |   |   |   |   |   |   |   |
| Reset | 0         | 0  | 0  | 0  | 0  | 0  | 0  | 0  | 0  | 0  | 0  | 0  | 0  | 0  | 0  | 0  | 0  | 0  | 0  | 0  | 0  | 0  | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |   |

#### MTB\_DEVICECFG field descriptions

| Field             | Description               |
|-------------------|---------------------------|
| 31–0<br>DEVICECFG | Hardwired to 0x0000_0000. |

### 19.31.13 Device Type Identifier Register (MTB\_DEVICETYPID)

This register indicates the device type ID. It is hardwired to specific values used during the auto-discovery process by an external debug agent.

Address: F000\_0000h base + FCCh offset = F000\_0FCCh

| Bit   | 31          | 30 | 29 | 28 | 27 | 26 | 25 | 24 | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|-------|-------------|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|---|---|---|---|---|---|---|---|---|---|
| R     | DEVICETYPID |    |    |    |    |    |    |    |    |    |    |    |    |    |    |    |    |    |    |    |    |    |   |   |   |   |   |   |   |   |   |   |
| W     |             |    |    |    |    |    |    |    |    |    |    |    |    |    |    |    |    |    |    |    |    |    |   |   |   |   |   |   |   |   |   |   |
| Reset | 0           | 0  | 0  | 0  | 0  | 0  | 0  | 0  | 0  | 0  | 0  | 0  | 0  | 0  | 0  | 0  | 0  | 0  | 0  | 0  | 0  | 0  | 0 | 0 | 0 | 0 | 1 | 1 | 0 | 0 | 0 | 1 |

#### MTB\_DEVICETYPID field descriptions

| Field               | Description               |
|---------------------|---------------------------|
| 31–0<br>DEVICETYPID | Hardwired to 0x0000_0031. |

### 19.31.14 Peripheral ID Register (MTB\_PERIPHIDn)

These registers indicate the peripheral IDs. They are hardwired to specific values used during the auto-discovery process by an external debug agent.

Address: F000\_0000h base + FD0h offset + (4d × i), where i=0d to 7d

| Bit   | 31       | 30 | 29 | 28 | 27 | 26 | 25 | 24 | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 | 15 | 14 | 13 | 12 | 11 | 10 | 9  | 8  | 7  | 6  | 5  | 4  | 3  | 2  | 1  | 0  |
|-------|----------|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|
| R     | PERIPHID |    |    |    |    |    |    |    |    |    |    |    |    |    |    |    |    |    |    |    |    |    |    |    |    |    |    |    |    |    |    |    |
| W     |          |    |    |    |    |    |    |    |    |    |    |    |    |    |    |    |    |    |    |    |    |    |    |    |    |    |    |    |    |    |    |    |
| Reset | x*       | x* | x* | x* | x* | x* | x* | x* | x* | x* | x* | x* | x* | x* | x* | x* | x* | x* | x* | x* | x* | x* | x* | x* | x* | x* | x* | x* | x* | x* | x* | x* |

\* Notes:

- See field descriptions for the reset values.x = Undefined at reset.

#### MTB\_PERIPHIDn field descriptions

| Field            | Description  |
|------------------|--|
| 31–0<br>PERIPHID | Peripheral ID4 is hardwired to 0x0000_0004; ID0 to 0x0000_0032; ID1 to 0x0000_00B9; ID2 to 0x0000_000B; and all the others to 0x0000_0000. |

## 19.31.15 Component ID Register (MTB\_COMPIDn)

These registers indicate the component IDs. They are hardwired to specific values used during the auto-discovery process by an external debug agent.

Address: F000\_0000h base + FF0h offset + (4d × i), where i=0d to 3d

| Bit | 31     | 30 | 29 | 28 | 27 | 26 | 25 | 24 | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|-----|--------|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|---|---|---|---|---|---|---|---|---|---|
| R   | COMPID |    |    |    |    |    |    |    |    |    |    |    |    |    |    |    |    |    |    |    |    |    |   |   |   |   |   |   |   |   |   |   |
| W   |        |    |    |    |    |    |    |    |    |    |    |    |    |    |    |    |    |    |    |    |    |    |   |   |   |   |   |   |   |   |   |   |

Reset x\* x\*

\* Notes:

- See field descriptions for the reset values.x = Undefined at reset.

### MTB\_COMPIDn field descriptions

| Field          | Description  |
|----------------|--|
| 31–0<br>COMPID | Component ID<br><br>Component ID0 is hardwired to 0x0000_000D; ID1 to 0x0000_0090; ID2 to 0x0000_0005; ID3 to 0x0000_00B1. |

## 19.3.2 MTB\_DWT Memory Map

The MTB\_DWT programming model supports a very simplified subset of the v7M debug architecture and follows the standard ARM DWT definition.

**MTBDWT memory map**

| Absolute address (hex) | Register name   | Width (in bits) | Access | Reset value | Section/ page               |
|------------------------|---|-----------------|--------|-------------|-----------------------------|
| F000_1000              | MTB DWT Control Register (MTBDWT_CTRL)                | 32              | R      | 2F00_0000h  | <a href="#">19.32.1/317</a> |
| F000_1020              | MTB_DWT Comparator Register (MTBDWT_COMP0)            | 32              | R/W    | 0000_0000h  | <a href="#">19.32.2/318</a> |
| F000_1024              | MTB_DWT Comparator Mask Register (MTBDWT_MASK0)       | 32              | R/W    | 0000_0000h  | <a href="#">19.32.3/319</a> |
| F000_1028              | MTB_DWT Comparator Function Register 0 (MTBDWT_FCT0)  | 32              | R/W    | 0000_0000h  | <a href="#">19.32.4/320</a> |
| F000_1030              | MTB_DWT Comparator Register (MTBDWT_COMP1)            | 32              | R/W    | 0000_0000h  | <a href="#">19.32.2/318</a> |
| F000_1034              | MTB_DWT Comparator Mask Register (MTBDWT_MASK1)       | 32              | R/W    | 0000_0000h  | <a href="#">19.32.3/319</a> |
| F000_1038              | MTB_DWT Comparator Function Register 1 (MTBDWT_FCT1)  | 32              | R/W    | 0000_0000h  | <a href="#">19.32.5/322</a> |
| F000_1200              | MTB_DWT Trace Buffer Control Register (MTBDWT_TBCTRL) | 32              | R/W    | 2000_0000h  | <a href="#">19.32.6/323</a> |
| F000_1FC8              | Device Configuration Register (MTBDWT_DEVICECFG)      | 32              | R      | 0000_0000h  | <a href="#">19.32.7/325</a> |

Table continues on the next page...

## MTBDWT memory map (continued)

| Absolute address (hex) | Register name  | Width (in bits) | Access | Reset value                 | Section/page                 |
|------------------------|--|-----------------|--------|-----------------------------|------------------------------|
| F000_1FCC              | Device Type Identifier Register (MTBDWT_DEVICETYPID) | 32              | R      | 0000_0004h                  | <a href="#">19.32.8/325</a>  |
| F000_1FD0              | Peripheral ID Register (MTBDWT_PERIPHID4)            | 32              | R      | <a href="#">See section</a> | <a href="#">19.32.9/326</a>  |
| F000_1FD4              | Peripheral ID Register (MTBDWT_PERIPHID5)            | 32              | R      | <a href="#">See section</a> | <a href="#">19.32.9/326</a>  |
| F000_1FD8              | Peripheral ID Register (MTBDWT_PERIPHID6)            | 32              | R      | <a href="#">See section</a> | <a href="#">19.32.9/326</a>  |
| F000_1FDC              | Peripheral ID Register (MTBDWT_PERIPHID7)            | 32              | R      | <a href="#">See section</a> | <a href="#">19.32.9/326</a>  |
| F000_1FE0              | Peripheral ID Register (MTBDWT_PERIPHID0)            | 32              | R      | <a href="#">See section</a> | <a href="#">19.32.9/326</a>  |
| F000_1FE4              | Peripheral ID Register (MTBDWT_PERIPHID1)            | 32              | R      | <a href="#">See section</a> | <a href="#">19.32.9/326</a>  |
| F000_1FE8              | Peripheral ID Register (MTBDWT_PERIPHID2)            | 32              | R      | <a href="#">See section</a> | <a href="#">19.32.9/326</a>  |
| F000_1FEC              | Peripheral ID Register (MTBDWT_PERIPHID3)            | 32              | R      | <a href="#">See section</a> | <a href="#">19.32.9/326</a>  |
| F000_1FF0              | Component ID Register (MTBDWT_COMPID0)               | 32              | R      | <a href="#">See section</a> | <a href="#">19.32.10/326</a> |
| F000_1FF4              | Component ID Register (MTBDWT_COMPID1)               | 32              | R      | <a href="#">See section</a> | <a href="#">19.32.10/326</a> |
| F000_1FF8              | Component ID Register (MTBDWT_COMPID2)               | 32              | R      | <a href="#">See section</a> | <a href="#">19.32.10/326</a> |
| F000_1FFC              | Component ID Register (MTBDWT_COMPID3)               | 32              | R      | <a href="#">See section</a> | <a href="#">19.32.10/326</a> |

### 19.32.1 MTB DWT Control Register (MTBDWTCtrl)

The MTBDWT\_CTRL register provides read-only information on the watchpoint configuration for the MTB\_DWT.

Address: F000 1000h base + 0h offset = F000 1000h

## MTBDWT CTRL field descriptions

| Field              | Description  |
|--------------------|--|
| 31–28<br>NUMCMP    | <p>Number of comparators</p> <p>The MTB_DWT implements two comparators.</p>  |
| 27–0<br>DWTCFGCTRL | <p>DWT configuration controls</p> <p>This field is hardwired to 0xF00_0000, disabling all the remaining DWT functionality. The specific fields and their state are:</p> <p>MTBDWT_CTRL[27] = NOTRCPKT = 1, trace sample and exception trace is not supported</p> |

*Table continues on the next page...*

## **MTBDWT\_CTRL field descriptions (continued)**

| Field | Description  |
|-------|--|
|       | MTBDWT_CTRL[26] = NOEXTTRIG = 1, external match signals are not supported<br>MTBDWT_CTRL[25] = NOCYCCNT = 1, cycle counter is not supported<br>MTBDWT_CTRL[24] = NOPRFCNT = 1, profiling counters are not supported<br>MTBDWT_CTRL[22] = CYCEBTENA = 0, no POSTCNT underflow packets generated<br>MTBDWT_CTRL[21] = FOLDEVVTENA = 0, no folded instruction counter overflow events<br>MTBDWT_CTRL[20] = LSUEVTENA = 0, no LSU counter overflow events<br>MTBDWT_CTRL[19] = SLEEPEVTENA = 0, no sleep counter overflow events<br>MTBDWT_CTRL[18] = EXCEVTENA = 0, no exception overhead counter events<br>MTBDWT_CTRL[17] = CPIEVVTENA = 0, no CPI counter overflow events<br>MTBDWT_CTRL[16] = EXCTRCENA = 0, generation of exception trace disabled<br>MTBDWT_CTRL[12] = PCSAMPLENA = 0, no periodic PC sample packets generated<br>MTBDWT_CTRL[11:10] = SYNCTAP = 0, no synchronization packets<br>MTBDWT_CTRL[9] = CYCTAP = 0, cycle counter is not supported<br>MTBDWT_CTRL[8:5] = POSTINIT = 0, cycle counter is not supported<br>MTBDWT_CTRL[4:1] = POSTPRESET = 0, cycle counter is not supported<br>MTBDWT_CTRL[0] = CYCCNTENA = 0, cycle counter is not supported |

### 19.32.2 MTB\_DWT Comparator Register (MTBDWT\_COMPn)

The MTBDWT\_COMPn registers provide the reference value for comparator n.

Address: F000\_1000h base + 20h offset + (16d × i), where i=0d to 1d

## **MTBDWT\_COMP*n* field descriptions**

| Field        | Description   |
|--------------|---|
| 31–0<br>COMP | <p>Reference value for comparison</p> <p>If MTBDWT_COMP0 is used for a data value comparator and the access size is byte or halfword, the data value must be replicated across all appropriate byte lanes of this register. For example, if the data is a byte-sized "x" value, then COMP[31:24] = COMP[23:16] = COMP[15:8] = COMP[7:0] = "x". Likewise, if the data is a halfword-size "y" value, then COMP[31:16] = COMP[15:0] = "y".</p> |

### 19.32.3 MTB\_DWT Comparator Mask Register (MTBDWT\_MASKn)

The MTBDWT\_MASKn registers define the size of the ignore mask applied to the reference address for address range matching by comparator n. Note the format of this mask field is different than the MTB\_MASTER[MASK].

Address: F000\_1000h base + 24h offset + (16d × i), where i=0d to 1d

| Bit   | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0    |
|-------|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|---|---|---|---|---|---|---|---|---|------|
| R     |    |    |    |    |    |    |    |    |    |    |    |    |    |    |    |    | 0  |    |    |    |    |    |   |   |   |   |   |   |   |   |   |      |
| W     |    |    |    |    |    |    |    |    |    |    |    |    |    |    |    |    |    |    |    |    |    |    |   |   |   |   |   |   |   |   |   | MASK |
| Reset | 0  | 0  | 0  | 0  | 0  | 0  | 0  | 0  | 0  | 0  | 0  | 0  | 0  | 0  | 0  | 0  | 0  | 0  | 0  | 0  | 0  | 0  | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |      |

#### MTBDWT\_MASKn field descriptions

| Field            | Description   |
|------------------|---|
| 31–5<br>Reserved | This field is reserved.<br>This read-only field is reserved and always has the value 0.   |
| 4–0<br>MASK      | MASK<br><br>The value of the ignore mask, 0-31 bits, is applied to address range matching. MASK = 0 is used to include all bits of the address in the comparison, except if MASK = 0 and the comparator is configured to watch instruction fetch addresses, address bit [0] is ignored by the hardware since all fetches must be at least halfword aligned. For MASK != 0 and regardless of watch type, address bits [x-1:0] are ignored in the address comparison.<br><br>Using a mask means the comparator matches on a range of addresses, defined by the unmasked most significant bits of the address, bits [31:x]. The maximum MASK value is 24, producing a 16 Mbyte mask. An attempted write of a MASK value > 24 is limited by the MTBDWT hardware to 24.<br><br>If MTBDWT_COMP0 is used as a data value comparator, then MTBDWT_MASK0 should be programmed to zero. |

### 19.32.4 MTB\_DWT Comparator Function Register 0 (MTBDWT\_FCT0)

The MTBDWT\_FCTn registers control the operation of comparator n.

Address: F000\_1000h base + 28h offset = F000\_1028h

| Bit   | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24         | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|-------|----|----|----|----|----|----|----|------------|----|----|----|----|----|----|----|----|
| R     |    |    |    |    |    |    |    | MATCHED    |    |    |    |    |    |    |    |    |
|       | 0  |    |    |    |    |    |    |            | 0  |    |    |    |    | 0  |    |    |
| W     |    |    |    |    |    |    |    |            |    |    |    |    |    |    |    |    |
| Reset | 0  | 0  | 0  | 0  | 0  | 0  | 0  | 0          | 0  | 0  | 0  | 0  | 0  | 0  | 0  | 0  |
| Bit   | 15 | 14 | 13 | 12 | 11 | 10 | 9  | 8          | 7  | 6  | 5  | 4  | 3  | 2  | 1  | 0  |
| R     |    |    |    |    |    |    |    | DATAVMATCH |    |    |    |    |    |    |    |    |
|       |    |    |    |    |    |    |    |            | 0  |    |    |    |    |    |    |    |
| W     |    |    |    |    |    |    |    |            |    |    |    |    |    |    |    |    |
| Reset | 0  | 0  | 0  | 0  | 0  | 0  | 0  | 0          | 0  | 0  | 0  | 0  | 0  | 0  | 0  | 0  |

**MTBDWT\_FCT0 field descriptions**

| Field             | Description  |
|-------------------|--|
| 31–25<br>Reserved | This field is reserved.<br>This read-only field is reserved and always has the value 0.  |
| 24<br>MATCHED     | Comparator match<br><br>If this read-only flag is asserted, it indicates the operation defined by the FUNCTION field occurred since the last read of the register. Reading the register clears this bit. |

*Table continues on the next page...*

**MTBDWT\_FCT0 field descriptions (continued)**

| Field               | Description  |
|---------------------|--|
|                     | <p>0 No match.<br/>1 Match occurred.</p>   |
| 23–20<br>Reserved   | This field is reserved.<br>This read-only field is reserved and always has the value 0.  |
| 19–16<br>Reserved   | This field is reserved.<br>This read-only field is reserved and always has the value 0.  |
| 15–12<br>DATAVADDR0 | <p>Data Value Address 0</p> <p>Since the MTB_DWT implements two comparators, the DATAVADDR0 field is restricted to values {0,1}. When the DATAVMATCH bit is asserted, this field defines the comparator number to use for linked address comparison.</p> <p>If MTBDWT_COMP0 is used as a data watchpoint and MTBDWT_COMP1 as an address watchpoint, DATAVADDR0 must be set.</p>  |
| 11–10<br>DATAVSIZE  | <p>Data Value Size</p> <p>For data value matching, this field defines the size of the required data comparison.</p> <p>00 Byte.<br/>01 Halfword.<br/>10 Word.<br/>11 Reserved. Any attempts to use this value results in UNPREDICTABLE behavior.</p>   |
| 9<br>Reserved       | This field is reserved.<br>This read-only field is reserved and always has the value 0.  |
| 8<br>DATAVMATCH     | <p>Data Value Match</p> <p>The assertion of this bit enables data value comparison. For this implementation, MTBDWT_COMP0 supports address or data value comparisons; MTBDWT_COMP1 only supports address comparisons.</p> <p>0 Perform address comparison.<br/>1 Perform data value comparison.</p>  |
| 7–4<br>Reserved     | This field is reserved.<br>This read-only field is reserved and always has the value 0.  |
| 3–0<br>FUNCTION     | <p>Function</p> <p>Selects the action taken on a comparator match. If MTBDWT_COMP0 is used for a data value and MTBDWT_COMP1 for an address value, then MTBDWT_FCT1[FUNCTION] must be set to zero. For this configuration, MTBDWT_MASK1 can be set to a non-zero value, so the combined comparators match on a range of addresses.</p> <p>0000 Disabled.<br/>0100 Instruction fetch.<br/>0101 Data operand read.<br/>0110 Data operand write.<br/>0111 Data operand (read + write).<br/>others Reserved. Any attempts to use this value results in UNPREDICTABLE behavior.</p> |

### 19.32.5 MTB\_DWT Comparator Function Register 1 (MTBDWT\_FCT1)

The MTBDWT\_FCTn registers control the operation of comparator n. Since the MTB\_DWT only supports data value comparisons on comparator 0, there are several fields in the MTBDWT\_FCT1 register that are RAZ/WI (bits 12, 11:10, 8).

Address: F000\_1000h base + 38h offset = F000\_1038h

| Bit   | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |         | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|-------|----|----|----|----|----|----|----|----|---------|----|----|----|----|----|----|----|----|
| R     |    |    |    |    |    |    |    | 0  | MATCHED |    |    |    |    | 0  |    |    |    |
| W     |    |    |    |    |    |    |    |    |         |    |    |    |    |    |    |    |    |
| Reset | 0  | 0  | 0  | 0  | 0  | 0  | 0  | 0  |         | 0  | 0  | 0  | 0  | 0  | 0  | 0  | 0  |
| Bit   | 15 | 14 | 13 | 12 | 11 | 10 | 9  | 8  |         | 7  | 6  | 5  | 4  | 3  | 2  | 1  | 0  |
| R     |    |    |    |    |    |    |    | 0  |         |    |    |    |    |    |    |    |    |
| W     |    |    |    |    |    |    |    |    |         |    |    |    |    |    |    |    |    |
| Reset | 0  | 0  | 0  | 0  | 0  | 0  | 0  | 0  |         | 0  | 0  | 0  | 0  | 0  | 0  | 0  | 0  |

#### MTBDWT\_FCT1 field descriptions

| Field             | Description  |
|-------------------|--|
| 31–25<br>Reserved | This field is reserved.<br>This read-only field is reserved and always has the value 0.  |
| 24<br>MATCHED     | Comparator match<br><br>If this read-only flag is asserted, it indicates the operation defined by the FUNCTION field occurred since the last read of the register. Reading the register clears this bit. |

*Table continues on the next page...*

## **MTBDWT\_FCT1 field descriptions (continued)**

| Field            | Description  |
|------------------|--|
|                  | 0 No match.<br>1 Match occurred.   |
| 23–4<br>Reserved | This field is reserved.<br>This read-only field is reserved and always has the value 0.  |
| 3–0<br>FUNCTION  | <p>Function</p> <p>Selects the action taken on a comparator match. If MTBDWT_COMP0 is used for a data value and MTBDWT_COMP1 for an address value, then MTBDWT_FCT1[FUNCTION] must be set to zero. For this configuration, MTBDWT_MASK1 can be set to a non-zero value, so the combined comparators match on a range of addresses.</p> <ul style="list-style-type: none"> <li>0000 Disabled.</li> <li>0100 Instruction fetch.</li> <li>0101 Data operand read.</li> <li>0110 Data operand write.</li> <li>0111 Data operand (read + write).</li> <li>others Reserved. Any attempts to use this value results in UNPREDICTABLE behavior.</li> </ul> |

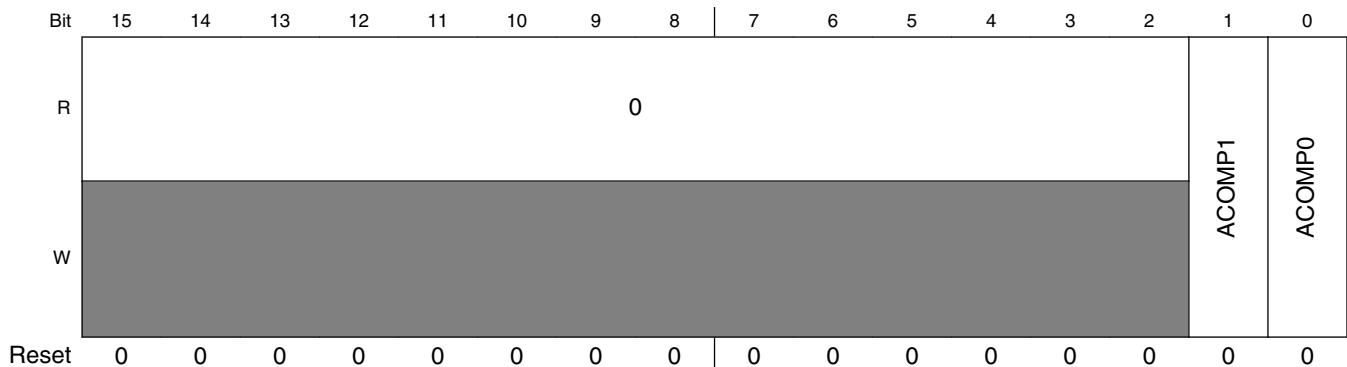
## 19.32.6 MTB\_DWT Trace Buffer Control Register (MTBDWT\_TBCTRL)

The MTBDWT\_TBCTRL register defines how the watchpoint comparisons control the actual trace buffer operation.

Recall the MTB supports starting and stopping the program trace based on the watchpoint comparisons signaled via TSTART and TSTOP. The watchpoint comparison signals are enabled in the MTB's control logic by setting the appropriate enable bits, MTB\_MASTER[TSTARTEN, TSTOPEN]. In the event of simultaneous assertion of both TSTART and TSTOP, TSTART takes priority.

Address: F000\_1000h base + 200h offset = F000\_1200h

## Memory Map and Register Definition



### MTBDWWT\_TBCTRL field descriptions

| Field            | Description   |
|------------------|---|
| 31–28<br>NUMCOMP | <p>Number of Comparators</p> <p>This read-only field specifies the number of comparators in the MTB_DWT. This implementation includes two registers.</p>  |
| 27–2<br>Reserved | <p>This field is reserved.</p> <p>This read-only field is reserved and always has the value 0.</p>  |
| 1<br>ACOMP1      | <p>Action based on Comparator 1 match</p> <p>When the MTBDWWT_FCT1[MATCHED] is set, it indicates MTBDWWT_COMP1 address compare has triggered and the trace buffer's recording state is changed.</p> <ul style="list-style-type: none"> <li>0 Trigger TSTOP based on the assertion of MTBDWWT_FCT1[MATCHED].</li> <li>1 Trigger TSTART based on the assertion of MTBDWWT_FCT1[MATCHED].</li> </ul>   |
| 0<br>ACOMP0      | <p>Action based on Comparator 0 match</p> <p>When the MTBDWWT_FCT0[MATCHED] is set, it indicates MTBDWWT_COMP0 address compare has triggered and the trace buffer's recording state is changed. The assertion of MTBDWWT_FCT0[MATCHED] is caused by the following conditions:</p> <ul style="list-style-type: none"> <li>• Address match in MTBDWWT_COMP0 when MTBDWWT_FCT0[DATAVMATCH] = 0</li> <li>• Data match in MTBDWWT_COMP0 when MTBDWWT_FCT0[DATAVMATCH, DATAVADDR0] = {1,0}</li> <li>• Data match in MTBDWWT_COMP0 and address match in MTBDWWT_COMP1 when MTBDWWT_FCT0[DATAVMATCH, DATAVADDR0] = {1,1}</li> </ul> <ul style="list-style-type: none"> <li>0 Trigger TSTOP based on the assertion of MTBDWWT_FCT0[MATCHED].</li> <li>1 Trigger TSTART based on the assertion of MTBDWWT_FCT0[MATCHED].</li> </ul> |

### 19.32.7 Device Configuration Register (MTBDWT\_DEVICECFG)

This register indicates the device configuration. It is hardwired to specific values used during the auto-discovery process by an external debug agent.

Address: F000\_1000h base + FC8h offset = F000\_1FC8h

| Bit   | 31        | 30 | 29 | 28 | 27 | 26 | 25 | 24 | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|-------|-----------|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|---|---|---|---|---|---|---|---|---|---|
| R     | DEVICECFG |    |    |    |    |    |    |    |    |    |    |    |    |    |    |    |    |    |    |    |    |    |   |   |   |   |   |   |   |   |   |   |
| W     |           |    |    |    |    |    |    |    |    |    |    |    |    |    |    |    |    |    |    |    |    |    |   |   |   |   |   |   |   |   |   |   |
| Reset | 0         | 0  | 0  | 0  | 0  | 0  | 0  | 0  | 0  | 0  | 0  | 0  | 0  | 0  | 0  | 0  | 0  | 0  | 0  | 0  | 0  | 0  | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |   |   |

#### MTBDWT\_DEVICECFG field descriptions

| Field             | Description               |
|-------------------|---------------------------|
| 31–0<br>DEVICECFG | Hardwired to 0x0000_0000. |

### 19.32.8 Device Type Identifier Register (MTBDWT\_DEVICETYPID)

This register indicates the device type ID. It is hardwired to specific values used during the auto-discovery process by an external debug agent.

Address: F000\_1000h base + FCCh offset = F000\_1FCCh

| Bit   | 31          | 30 | 29 | 28 | 27 | 26 | 25 | 24 | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|-------|-------------|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|---|---|---|---|---|---|---|---|---|---|
| R     | DEVICETYPID |    |    |    |    |    |    |    |    |    |    |    |    |    |    |    |    |    |    |    |    |    |   |   |   |   |   |   |   |   |   |   |
| W     |             |    |    |    |    |    |    |    |    |    |    |    |    |    |    |    |    |    |    |    |    |    |   |   |   |   |   |   |   |   |   |   |
| Reset | 0           | 0  | 0  | 0  | 0  | 0  | 0  | 0  | 0  | 0  | 0  | 0  | 0  | 0  | 0  | 0  | 0  | 0  | 0  | 0  | 0  | 0  | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 1 | 0 | 0 |

#### MTBDWT\_DEVICETYPID field descriptions

| Field               | Description               |
|---------------------|---------------------------|
| 31–0<br>DEVICETYPID | Hardwired to 0x0000_0004. |

### 19.32.9 Peripheral ID Register (MTBDWT\_PERIPHID $n$ )

These registers indicate the peripheral IDs. They are hardwired to specific values used during the auto-discovery process by an external debug agent.

Address: F000\_1000h base + FD0h offset + (4d × i), where i=0d to 7d

| Bit | 31       | 30 | 29 | 28 | 27 | 26 | 25 | 24 | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|-----|----------|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|---|---|---|---|---|---|---|---|---|---|
| R   | PERIPHID |    |    |    |    |    |    |    |    |    |    |    |    |    |    |    |    |    |    |    |    |    |   |   |   |   |   |   |   |   |   |   |
| W   |          |    |    |    |    |    |    |    |    |    |    |    |    |    |    |    |    |    |    |    |    |    |   |   |   |   |   |   |   |   |   |   |

Reset x\* | x\* x\*

\* Notes:

- See field descriptions for the reset values. x = Undefined at reset.

#### MTBDWT\_PERIPHID $n$ field descriptions

| Field            | Description  |
|------------------|--|
| 31–0<br>PERIPHID | Peripheral ID1 is hardwired to 0x0000_00E0; ID2 to 0x0000_0008; and all the others to 0x0000_0000. |

### 19.32.10 Component ID Register (MTBDWT\_COMPID $n$ )

These registers indicate the component IDs. They are hardwired to specific values used during the auto-discovery process by an external debug agent.

Address: F000\_1000h base + FF0h offset + (4d × i), where i=0d to 3d

| Bit | 31     | 30 | 29 | 28 | 27 | 26 | 25 | 24 | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|-----|--------|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|---|---|---|---|---|---|---|---|---|---|
| R   | COMPID |    |    |    |    |    |    |    |    |    |    |    |    |    |    |    |    |    |    |    |    |    |   |   |   |   |   |   |   |   |   |   |
| W   |        |    |    |    |    |    |    |    |    |    |    |    |    |    |    |    |    |    |    |    |    |    |   |   |   |   |   |   |   |   |   |   |

Reset x\* | x\* x\*

\* Notes:

- See field descriptions for the reset values.x = Undefined at reset.

#### MTBDWT\_COMPID $n$ field descriptions

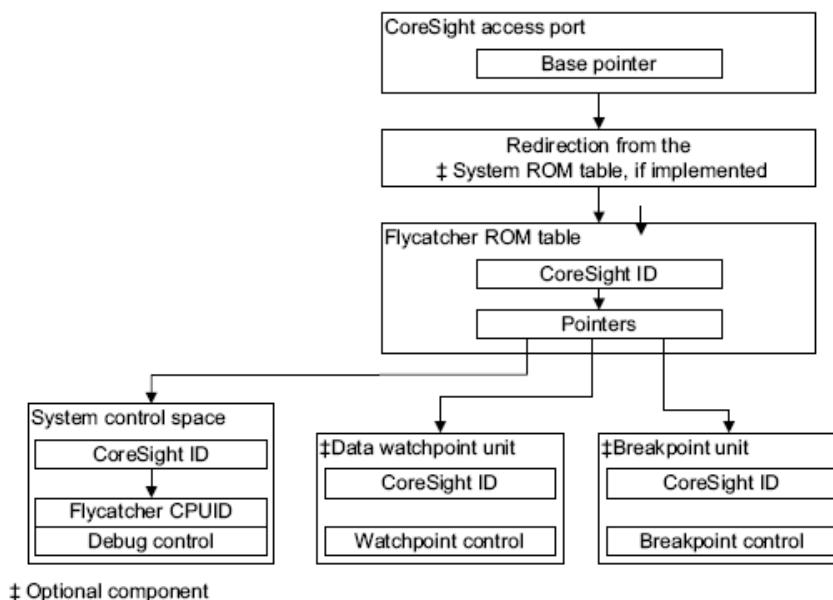
| Field          | Description  |
|----------------|--|
| 31–0<br>COMPID | Component ID0 is hardwired to 0x0000_000D; ID1 to 0x0000_0090; ID2 to 0x0000_0005; ID3 to 0x0000_00B1. |

### 19.3.3 System ROM Memory Map

The System ROM Table registers are also mapped into a sparsely-populated 4 KB address space.

For core configurations like that supported by Cortex-M0+, ARM recommends that a debugger identifies and connects to the debug components using the CoreSight debug infrastructure.

ARM recommends that a debugger follows the flow as shown in the following figure to discover the components in the CoreSight debug infrastructure. In this case a debugger reads the peripheral and component ID registers for each CoreSight component in the CoreSight system.



**Figure 19-56. CoreSight discovery process**

**ROM memory map**

| Absolute address (hex) | Register name                                | Width (in bits) | Access | Reset value                 | Section/ page |
|------------------------|--|-----------------|--------|-----------------------------|---------------|
| F000_2000              | Entry (ROM_ENTRY0)                           | 32              | R      | <a href="#">See section</a> | 19.33.1/328   |
| F000_2004              | Entry (ROM_ENTRY1)                           | 32              | R      | <a href="#">See section</a> | 19.33.1/328   |
| F000_2008              | Entry (ROM_ENTRY2)                           | 32              | R      | <a href="#">See section</a> | 19.33.1/328   |
| F000_200C              | End of Table Marker Register (ROM_TABLEMARK) | 32              | R      | 0000_0000h                  | 19.33.2/329   |
| F000_2FCC              | System Access Register (ROM_SYSACCESS)       | 32              | R      | 0000_0001h                  | 19.33.3/329   |
| F000_2FD0              | Peripheral ID Register (ROM_PERIPHID4)       | 32              | R      | <a href="#">See section</a> | 19.33.4/330   |
| F000_2FD4              | Peripheral ID Register (ROM_PERIPHID5)       | 32              | R      | <a href="#">See section</a> | 19.33.4/330   |
| F000_2FD8              | Peripheral ID Register (ROM_PERIPHID6)       | 32              | R      | <a href="#">See section</a> | 19.33.4/330   |

Table continues on the next page...

**ROM memory map (continued)**

| Absolute address (hex) | Register name                          | Width (in bits) | Access | Reset value                 | Section/page |
|------------------------|--|-----------------|--------|-----------------------------|--------------|
| F000_2FDC              | Peripheral ID Register (ROM_PERIPHID7) | 32              | R      | <a href="#">See section</a> | 19.33.4/330  |
| F000_2FE0              | Peripheral ID Register (ROM_PERIPHID0) | 32              | R      | <a href="#">See section</a> | 19.33.4/330  |
| F000_2FE4              | Peripheral ID Register (ROM_PERIPHID1) | 32              | R      | <a href="#">See section</a> | 19.33.4/330  |
| F000_2FE8              | Peripheral ID Register (ROM_PERIPHID2) | 32              | R      | <a href="#">See section</a> | 19.33.4/330  |
| F000_2FEC              | Peripheral ID Register (ROM_PERIPHID3) | 32              | R      | <a href="#">See section</a> | 19.33.4/330  |
| F000_2FF0              | Component ID Register (ROM_COMPID0)    | 32              | R      | <a href="#">See section</a> | 19.33.5/330  |
| F000_2FF4              | Component ID Register (ROM_COMPID1)    | 32              | R      | <a href="#">See section</a> | 19.33.5/330  |
| F000_2FF8              | Component ID Register (ROM_COMPID2)    | 32              | R      | <a href="#">See section</a> | 19.33.5/330  |
| F000_2FFC              | Component ID Register (ROM_COMPID3)    | 32              | R      | <a href="#">See section</a> | 19.33.5/330  |

**19.33.1 Entry (ROM\_ENTRYn)**

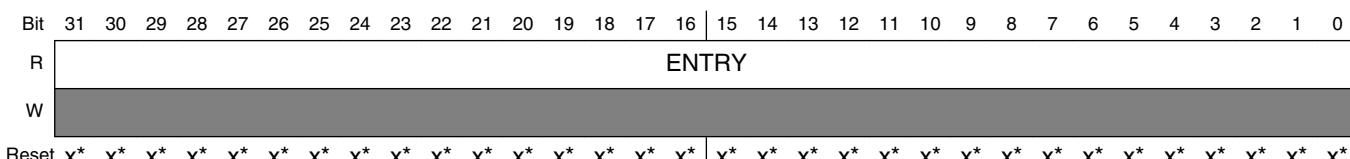
The System ROM Table begins with "n" relative 32-bit addresses, one for each debug component present in the device and terminating with an all-zero value signaling the end of the table at the "n+1"-th value.

**NOTE**

See Chip Configuration chapter for the debug components these registers point to.

It is hardwired to specific values used during the auto-discovery process by an external debug agent.

Address: F000\_2000h base + 0h offset + (4d × i), where i=0d to 2d



\* Notes:

- See field descriptions for reset values.x = Undefined at reset.

**ROM\_ENTRYn field descriptions**

| Field         | Description  |
|---------------|--|
| 31–0<br>ENTRY | Entry 0 (MTB) is hardwired to 0xFFFF_E003; Entry 1 (MTBDWT) to 0xFFFF_F003; Entry 2 (CM0+ ROM Table) to 0xF00F_D003. |

### 19.33.2 End of Table Marker Register (ROM\_TABLEMARK)

This register indicates end of table marker. It is hardwired to specific values used during the auto-discovery process by an external debug agent.

Address: F000\_2000h base + Ch offset = F000\_200Ch

| Bit   | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|-------|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|---|---|---|---|---|---|---|---|---|---|
| R     |    |    |    |    |    |    |    |    |    |    |    |    |    |    |    |    |    |    |    |    |    |    |   |   |   |   |   |   |   |   |   |   |
| W     |    |    |    |    |    |    |    |    |    |    |    |    |    |    |    |    |    |    |    |    |    |    |   |   |   |   |   |   |   |   |   |   |
| Reset | 0  | 0  | 0  | 0  | 0  | 0  | 0  | 0  | 0  | 0  | 0  | 0  | 0  | 0  | 0  | 0  | 0  | 0  | 0  | 0  | 0  | 0  | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |   |   |

#### ROM\_TABLEMARK field descriptions

| Field        | Description              |
|--------------|--------------------------|
| 31–0<br>MARK | Hardwired to 0x0000_0000 |

### 19.33.3 System Access Register (ROM\_SYSACCESS)

This register indicates system access. It is hardwired to specific values used during the auto-discovery process by an external debug agent.

Address: F000\_2000h base + FCCh offset = F000\_2FCCh

| Bit   | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|-------|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|---|---|---|---|---|---|---|---|---|---|
| R     |    |    |    |    |    |    |    |    |    |    |    |    |    |    |    |    |    |    |    |    |    |    |   |   |   |   |   |   |   |   |   |   |
| W     |    |    |    |    |    |    |    |    |    |    |    |    |    |    |    |    |    |    |    |    |    |    |   |   |   |   |   |   |   |   |   |   |
| Reset | 0  | 0  | 0  | 0  | 0  | 0  | 0  | 0  | 0  | 0  | 0  | 0  | 0  | 0  | 0  | 0  | 0  | 0  | 0  | 0  | 0  | 0  | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 1 |   |   |

#### ROM\_SYSACCESS field descriptions

| Field             | Description              |
|-------------------|--------------------------|
| 31–0<br>SYSACCESS | Hardwired to 0x0000_0001 |

### 19.33.4 Peripheral ID Register (ROM\_PERIPHID $n$ )

These registers indicate the peripheral IDs. They are hardwired to specific values used during the auto-discovery process by an external debug agent.

Address: F000\_2000h base + FD0h offset + (4d × i), where i=0d to 7d

| Bit | 31       | 30 | 29 | 28 | 27 | 26 | 25 | 24 | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|-----|----------|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|---|---|---|---|---|---|---|---|---|---|
| R   | PERIPHID |    |    |    |    |    |    |    |    |    |    |    |    |    |    |    |    |    |    |    |    |    |   |   |   |   |   |   |   |   |   |   |
| W   |          |    |    |    |    |    |    |    |    |    |    |    |    |    |    |    |    |    |    |    |    |    |   |   |   |   |   |   |   |   |   |   |

Reset x\* | x\* x\*

\* Notes:

- See field descriptions for reset values.x = Undefined at reset.

#### ROM\_PERIPHID $n$ field descriptions

| Field            | Description  |
|------------------|--|
| 31–0<br>PERIPHID | Peripheral ID1 is hardwired to 0x0000_00E0; ID2 to 0x0000_0008; and all the others to 0x0000_0000. |

### 19.33.5 Component ID Register (ROM\_COMPID $n$ )

These registers indicate the component IDs. They are hardwired to specific values used during the auto-discovery process by an external debug agent.

Address: F000\_2000h base + FF0h offset + (4d × i), where i=0d to 3d

| Bit | 31     | 30 | 29 | 28 | 27 | 26 | 25 | 24 | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|-----|--------|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|---|---|---|---|---|---|---|---|---|---|
| R   | COMPID |    |    |    |    |    |    |    |    |    |    |    |    |    |    |    |    |    |    |    |    |    |   |   |   |   |   |   |   |   |   |   |
| W   |        |    |    |    |    |    |    |    |    |    |    |    |    |    |    |    |    |    |    |    |    |    |   |   |   |   |   |   |   |   |   |   |

Reset x\* | x\* x\*

\* Notes:

- See field descriptions for reset values.x = Undefined at reset.

#### ROM\_COMPID $n$ field descriptions

| Field          | Description  |
|----------------|--|
| 31–0<br>COMPID | Component ID0 is hardwired to 0x0000_000D; ID1 to 0x0000_0010; ID2 to 0x0000_0005; ID3 to 0x0000_00B1. |

# **Chapter 20**

## **Crossbar Switch Lite (AXBS-Lite)**

### **20.1 Introduction**

#### **NOTE**

For the chip-specific implementation details of this module's instances see the chip configuration information.

This chapter provides information on the layout, configuration, and programming of the crossbar switch. The crossbar switch connects bus masters and bus slaves using a crossbar switch structure. This structure allows up to four bus masters to access different bus slaves simultaneously, while providing arbitration among the bus masters when they access the same slave.

#### **20.1.1 Features**

The crossbar switch includes these features:

- Symmetric crossbar bus switch implementation
  - Allows concurrent accesses from different masters to different slaves
- 32-bit data bus
- Operation at a 1-to-1 clock frequency with the bus masters
- Programmable configuration for fixed-priority or round-robin slave port arbitration

### **20.2 Memory Map / Register Definition**

This crossbar switch is designed for minimal gate count. It, therefore, has no memory-mapped configuration registers.

## 20.3 Functional Description

### 20.3.1 General operation

When a master accesses the crossbar switch the access is immediately taken. If the targeted slave port of the access is available, then the access is immediately presented on the slave port. Single-clock or zero-wait-state accesses are possible through the crossbar. If the targeted slave port of the access is busy or parked on a different master port, the requesting master simply sees wait states inserted until the targeted slave port can service the master's request. The latency in servicing the request depends on each master's priority level and the responding slave's access time.

Because the crossbar switch appears to be just another slave to the master device, the master device has no knowledge of whether it actually owns the slave port it is targeting. While the master does not have control of the slave port it is targeting, it simply waits.

A master is given control of the targeted slave port only after a previous access to a different slave port completes, regardless of its priority on the newly targeted slave port. This prevents deadlock from occurring when:

- A higher priority master has:
  - An outstanding request to one slave port that has a long response time and
  - A pending access to a different slave port, and
- A lower priority master is also making a request to the same slave port as the pending access of the higher priority master.

After the master has control of the slave port it is targeting, the master remains in control of the slave port until it relinquishes the slave port by running an IDLE cycle or by targeting a different slave port for its next access.

The master can also lose control of the slave port if another higher-priority master makes a request to the slave port.

The crossbar terminates all master IDLE transfers, as opposed to allowing the termination to come from one of the slave buses. Additionally, when no master is requesting access to a slave port, the crossbar drives IDLE transfers onto the slave bus, even though a default master may be granted access to the slave port.

When a slave bus is being idled by the crossbar, it remains parked with the last master to use the slave port. This is done to save the initial clock of arbitration delay that otherwise would be seen if the master had to arbitrate to gain control of the slave port.

## 20.3.2 Arbitration

The crossbar switch supports two arbitration algorithms:

- Fixed priority
- Round robin

The selection of the global slave port arbitration is controlled by MCM\_PLACR[ARB]. For fixed priority, set ARB to 0. For round robin, set ARB to 1. This arbitration setting applies to all slave ports.

### 20.3.2.1 Arbitration During Undefined Length Bursts

All lengths of burst accesses lock out arbitration until the last beat of the burst.

### 20.3.2.2 Fixed-priority operation

When operating in fixed-priority mode, each master is assigned a unique priority level with the highest numbered master having the highest priority (master 1 has lower priority than master 3). If two masters request access to the same slave port, the master with the highest priority gains control over the slave port.

#### NOTE

In this arbitration mode, a higher-priority master can monopolize a slave port, preventing accesses from any lower-priority master to the port.

When a master makes a request to a slave port, the slave port checks whether the new requesting master's priority level is higher than that of the master that currently has control over the slave port, unless the slave port is in a parked state. The slave port performs an arbitration check at every clock edge to ensure that the proper master, if any, has control of the slave port.

The following table describes possible scenarios based on the requesting master port:

**Table 20-1. How AXBS grants control of a slave port to a master**

| When   | Then AXBS grants control to the requesting master   |
|--|---|
| Both of the following are true: <ul style="list-style-type: none"> <li>• The current master is not running a transfer.</li> <li>• The new requesting master's priority level is higher than that of the current master.</li> </ul> | At the next clock edge  |
| The requesting master's priority level is lower than the current master.   | At the conclusion of one of the following cycles: <ul style="list-style-type: none"> <li>• An IDLE cycle</li> <li>• A non-IDLE cycle to a location other than the current slave port</li> </ul> |

### 20.3.2.3 Round-robin priority operation

When operating in round-robin mode, each master is assigned a relative priority based on the master port number. This relative priority is compared to the master port number (ID) of the last master to perform a transfer on the slave bus. The highest priority requesting master becomes owner of the slave bus at the next transfer boundary. Priority is based on how far ahead the ID of the requesting master is to the ID of the last master.

After granted access to a slave port, a master may perform as many transfers as desired to that port until another master makes a request to the same slave port. The next master in line is granted access to the slave port at the next transfer boundary, or possibly on the next clock cycle if the current master has no pending access request.

As an example of arbitration in round-robin mode, assume the crossbar is implemented with master ports 0, 1, 4, and 5. If the last master of the slave port was master 1, and master 0, 4 and 5 make simultaneous requests, they are serviced in the order 4, 5, and then 0.

The round-robin arbitration mode generally provides a more fair allocation of the available slave-port bandwidth (compared to fixed priority) as the fixed master priority does not affect the master selection.

## 20.4 Initialization/application information

No initialization is required for the crossbar switch. See the AXBS section of the configuration chapter for the reset state of the arbitration scheme.

# **Chapter 21**

## **Peripheral Bridge (AIPS-Lite)**

### **21.1 Introduction**

#### **NOTE**

For the chip-specific implementation details of this module's instances see the chip configuration information.

The peripheral bridge converts the crossbar switch interface to an interface that can access most of the slave peripherals on this chip.

The peripheral bridge supports up to 128 peripherals, each with a 4K-byte address space. (Not all peripheral slots might be used. See the chip configuration chapter and memory map chapter for details on slot assignment.) The bridge includes separate clock enable inputs for each of the slots to accommodate slower peripherals.

#### **21.1.1 Features**

Key features of the peripheral bridge are:

- Supports peripheral slots with 8-, 16-, and 32-bit datapath width
- Dedicated clock enables for independently configurable peripherals allow each on- or off-platform peripheral to operate at any integer-divisible speed less than or equal to the system clock frequency.

## 21.1.2 General operation

The slave devices connected to the peripheral bridge are modules which contain a programming model of control and status registers. The system masters read and write these registers through the peripheral bridge. The peripheral bridge performs a bus protocol conversion of the master transactions and generates the following as inputs to the peripherals:

- Module enables
- Module addresses
- Transfer attributes
- Byte enables
- Write data

The peripheral bridge selects and captures read data from the peripheral interface and returns it to the crossbar switch.

The register maps of the peripherals are located on 4-KB boundaries. Each peripheral is allocated one or more 4-KB block(s) of the memory map.

The AIPS-Lite module uses the accessed peripheral's data width to perform proper data byte lane routing; bus decomposition (bus sizing) is performed when the access size is larger than the peripheral's data width.

## 21.2 Functional description

The peripheral bridge functions as a bus protocol translator between the crossbar switch and the slave peripheral bus.

The peripheral bridge manages all transactions destined for the attached slave devices and generates select signals for modules on the peripheral bus by decoding accesses within the attached address space.

By default, reads and writes on the crossbar side of the peripheral bridge take two data-phase cycles. On the IPS side, accesses complete in one cycle. If wait states are inserted by the slave peripheral, access time will be extended accordingly.

### 21.2.1 Access support

All combinations of access size and peripheral data port width are supported. An access that is larger than the target peripheral's data width will be decomposed to multiple, smaller accesses. Bus decomposition is terminated by a transfer error caused by an access to an empty register area.

# **Chapter 22**

## **Direct Memory Access Multiplexer (DMAMUX)**

### **22.1 Introduction**

#### **NOTE**

For the chip-specific implementation details of this module's  
instae th

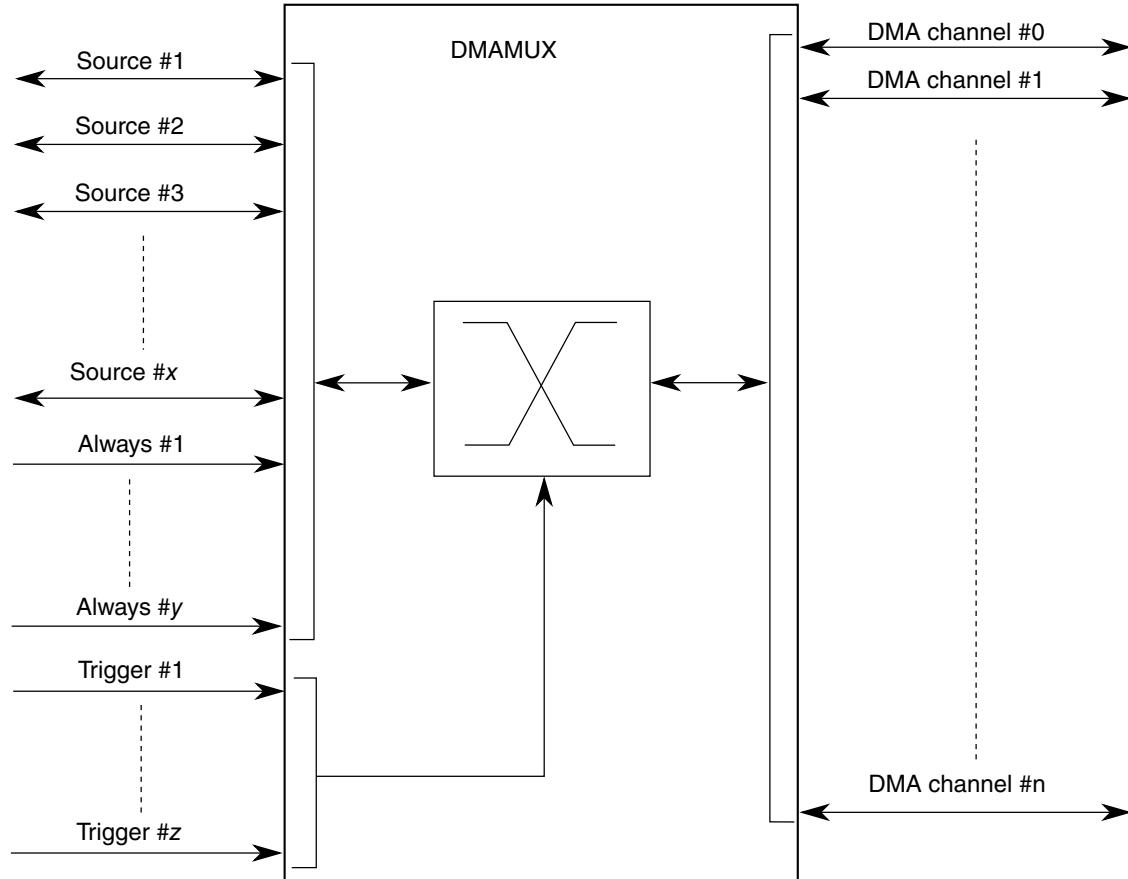


Figure 22-1. DMAMUX block diagram

## 22.1.2 Features

The DMAMUX module provides these features:

- 63 peripheral slots and 6 always-on slots can be routed to 4 channels.
- 4 independently selectable DMA channel routers.
  - The first 2 channels additionally provide a trigger functionality.
- Each channel router can be assigned to one of the 63 possible peripheral DMA slots or to one of the 6 always-on slots.

## 22.1.3 Modes of operation

The following operating modes are available:

- Disabled mode

In this mode, the DMA channel is disabled. Because disabling and enabling of DMA channels is done primarily via the DMA configuration registers, this mode is used mainly as the reset state for a DMA channel in the DMA channel MUX. It may also be used to temporarily suspend a DMA channel while reconfiguration of the system takes place, for example, changing the period of a DMA trigger.

- Normal mode

In this mode, a DMA source is routed directly to the specified DMA channel. The operation of the DMAMUX in this mode is completely transparent to the system.

- Periodic Trigger mode

In this mode, a DMA source may only request a DMA transfer, such as when a transmit buffer becomes empty or a receive buffer becomes full, periodically.

Configuration of the period is done in the registers of the periodic interrupt timer (PIT). This mode is available only for channels 0–1.

## 22.2 External signal description

The DMAMUX has no external pins.

## 22.3 Memory map/register definition

This section provides a detailed description of all memory-mapped registers in the DMAMUX.

**DMAMUX memory map**

| Absolute address (hex) | Register name                                   | Width (in bits) | Access | Reset value | Section/page               |
|------------------------|---|-----------------|--------|-------------|----------------------------|
| 4002_1000              | Channel Configuration register (DMAMUX0_CHCFG0) | 8               | R/W    | 00h         | <a href="#">22.3.1/339</a> |
| 4002_1001              | Channel Configuration register (DMAMUX0_CHCFG1) | 8               | R/W    | 00h         | <a href="#">22.3.1/339</a> |
| 4002_1002              | Channel Configuration register (DMAMUX0_CHCFG2) | 8               | R/W    | 00h         | <a href="#">22.3.1/339</a> |
| 4002_1003              | Channel Configuration register (DMAMUX0_CHCFG3) | 8               | R/W    | 00h         | <a href="#">22.3.1/339</a> |

### 22.3.1 Channel Configuration register (DMAMUXx\_CHCFGn)

Each of the DMA channels can be independently enabled/disabled and associated with one of the DMA slots (peripheral slots or always-on slots) in the system.

**NOTE**

Setting multiple CHCFG registers with the same Source value will result in unpredictable behavior.

**NOTE**

Before changing the trigger or source settings a DMA channel must be disabled via the CHCFGn[ENBL] bit.

Address: 4002\_1000h base + 0h offset + (1d × i), where i=0d to 3d

| Bit        | 7    | 6    | 5 | 4      | 3 | 2 | 1 | 0 |
|------------|------|------|---|--------|---|---|---|---|
| Read Write | ENBL | TRIG |   | SOURCE |   |   |   |   |
| Reset      | 0    | 0    | 0 | 0      | 0 | 0 | 0 | 0 |

**DMAMUXx\_CHCFGn field descriptions**

| Field         | Description   |
|---------------|---|
| 7<br>ENBL     | DMA Channel Enable<br><br>Enables the DMA channel.<br><br>0 DMA channel is disabled. This mode is primarily used during configuration of the DMA Mux. The DMA has separate channel enables/disables, which should be used to disable or re-configure a DMA channel.<br>1 DMA channel is enabled   |
| 6<br>TRIG     | DMA Channel Trigger Enable<br><br>Enables the periodic trigger capability for the triggered DMA channel.<br><br>0 Triggering is disabled. If triggering is disabled, and the ENBL bit is set, the DMA Channel will simply route the specified source to the DMA channel. (Normal mode)<br>1 Triggering is enabled. If triggering is enabled, and the ENBL bit is set, the DMAMUX is in Periodic Trigger mode. |
| 5–0<br>SOURCE | DMA Channel Source (Slot)<br><br>Specifies which DMA source, if any, is routed to a particular DMA channel. See your device's chip configuration details for further details about the peripherals and their slot numbers.  |

## 22.4 Functional description

The primary purpose of the DMAMUX is to provide flexibility in the system's use of the available DMA channels. As such, configuration of the DMAMUX is intended to be a static procedure done during execution of the system boot code. However, if the procedure outlined in [Enabling and configuring sources](#) is followed, the configuration of the DMAMUX may be changed during the normal operation of the system.

Functionally, the DMAMUX channels may be divided into two classes:

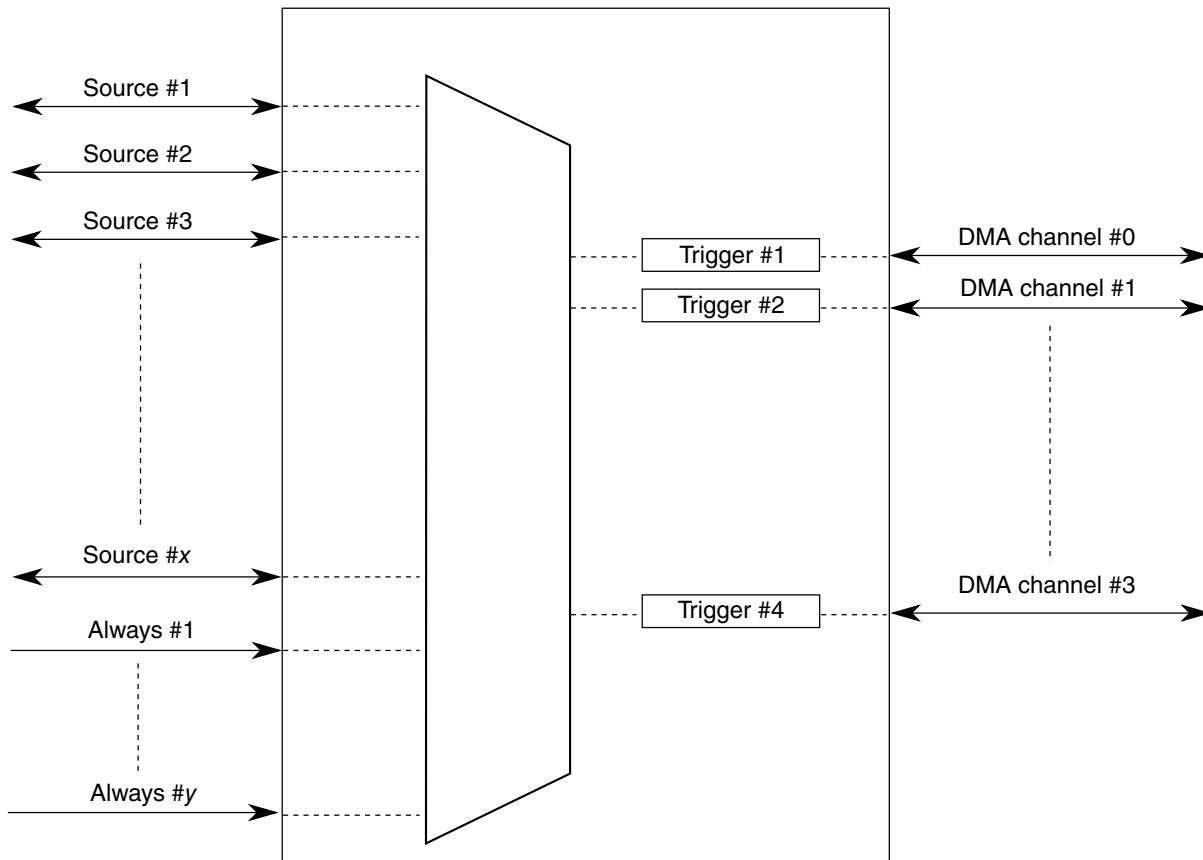
- Channels which implement the normal routing functionality plus periodic triggering capability
- Channels which implement only the normal routing functionality

### 22.4.1 DMA channels with periodic triggering capability

Besides the normal routing functionality, the first 2 channels of the DMAMUX provide a special periodic triggering capability that can be used to provide an automatic mechanism to transmit bytes, frames, or packets at fixed intervals without the need for processor intervention. The trigger is generated by the periodic interrupt timer (PIT); as such, the configuration of the periodic triggering interval is done via configuration registers in the PIT. See the section on periodic interrupt timer for more information on this topic.

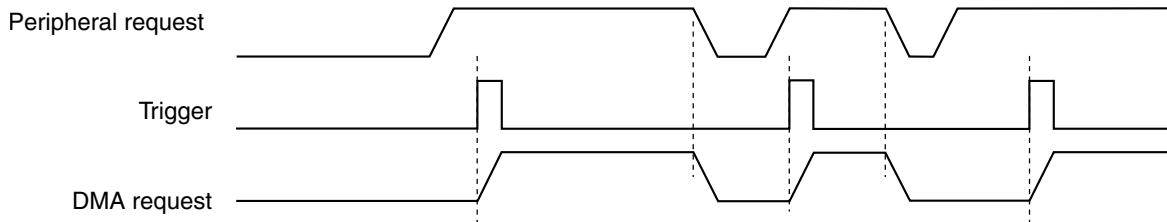
#### Note

Because of the dynamic nature of the system (due to DMA channel priorities, bus arbitration, interrupt service routine lengths, etc.), the number of clock cycles between a trigger and the actual DMA transfer cannot be guaranteed.



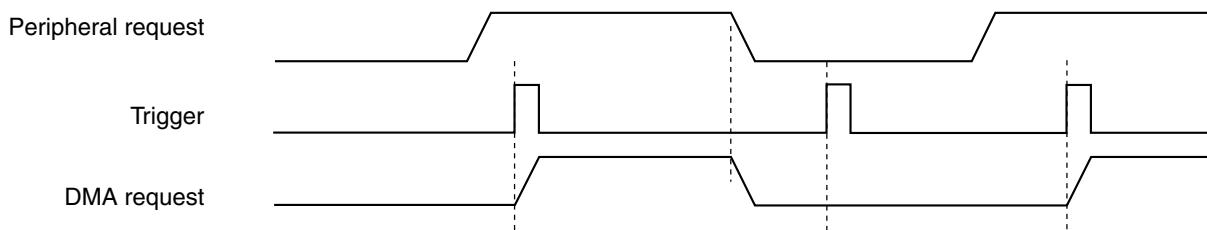
**Figure 22-12. DMAMUX triggered channels**

The DMA channel triggering capability allows the system to schedule regular DMA transfers, usually on the transmit side of certain peripherals, without the intervention of the processor. This trigger works by gating the request from the peripheral to the DMA until a trigger event has been seen. This is illustrated in the following figure.



**Figure 22-13. DMAMUX channel triggering: normal operation**

After the DMA request has been serviced, the peripheral will negate its request, effectively resetting the gating mechanism until the peripheral re-asserts its request AND the next trigger event is seen. This means that if a trigger is seen, but the peripheral is not requesting a transfer, then that trigger will be ignored. This situation is illustrated in the following figure.



**Figure 22-14. DMAMUX channel triggering: ignored trigger**

This triggering capability may be used with any peripheral that supports DMA transfers, and is most useful for two types of situations:

- Periodically polling external devices on a particular bus

As an example, the transmit side of an SPI is assigned to a DMA channel with a trigger, as described above. After it has been set up, the SPI will request DMA transfers, presumably from memory, as long as its transmit buffer is empty. By using a trigger on this channel, the SPI transfers can be automatically performed every 5 $\mu$ s (as an example). On the receive side of the SPI, the SPI and DMA can be configured to transfer receive data into memory, effectively implementing a method to periodically read data from external devices and transfer the results into memory without processor intervention.

- Using the GPIO ports to drive or sample waveforms

By configuring the DMA to transfer data to one or more GPIO ports, it is possible to create complex waveforms using tabular data stored in on-chip memory. Conversely, using the DMA to periodically transfer data from one or more GPIO ports, it is possible to sample complex waveforms and store the results in tabular form in on-chip memory.

A more detailed description of the capability of each trigger, including resolution, range of values, and so on, may be found in the periodic interrupt timer section.

## 22.4.2 DMA channels with no triggering capability

The other channels of the DMAMUX provide the normal routing functionality as described in [Modes of operation](#).

## 22.4.3 Always-enabled DMA sources

In addition to the peripherals that can be used as DMA sources, there are 6 additional DMA sources that are always enabled. Unlike the peripheral DMA sources, where the peripheral controls the flow of data during DMA transfers, the sources that are always enabled provide no such "throttling" of the data transfers. These sources are most useful in the following cases:

- Performing DMA transfers to/from GPIO—Moving data from/to one or more GPIO pins, either unthrottled (that is as fast as possible), or periodically (using the DMA triggering capability).
- Performing DMA transfers from memory to memory—Moving data from memory to memory, typically as fast as possible, sometimes with software activation.
- Performing DMA transfers from memory to the external bus, or vice-versa—Similar to memory to memory transfers, this is typically done as quickly as possible.
- Any DMA transfer that requires software activation—Any DMA transfer that should be explicitly started by software.

In cases where software should initiate the start of a DMA transfer, an always-enabled DMA source can be used to provide maximum flexibility. When activating a DMA channel via software, subsequent executions of the minor loop require that a new start event be sent. This can either be a new software activation, or a transfer request from the DMA channel MUX. The options for doing this are:

- Transfer all data in a single minor loop.

By configuring the DMA to transfer all of the data in a single minor loop (that is, major loop counter = 1), no reactivation of the channel is necessary. The disadvantage to this option is the reduced granularity in determining the load that the DMA transfer will impose on the system. For this option, the DMA channel must be disabled in the DMA channel MUX.

- Use explicit software reactivation.

In this option, the DMA is configured to transfer the data using both minor and major loops, but the processor is required to reactivate the channel by writing to the DMA registers *after every minor loop*. For this option, the DMA channel must be disabled in the DMA channel MUX.

- Use an always-enabled DMA source.

In this option, the DMA is configured to transfer the data using both minor and major loops, and the DMA channel MUX does the channel reactivation. For this option, the DMA channel should be enabled and pointing to an "always enabled" source. Note that the reactivation of the channel can be continuous (DMA triggering is disabled) or can use the DMA triggering capability. In this manner, it is possible to execute periodic transfers of packets of data from one source to another, without processor intervention.

## **22.5 Initialization/application information**

This section provides instructions for initializing the DMA channel MUX.

### **22.5.1 Reset**

The reset state of each individual bit is shown in [Memory map/register definition](#). In summary, after reset, all channels are disabled and must be explicitly enabled before use.

### **22.5.2 Enabling and configuring sources**

To enable a source with periodic triggering:

1. Determine with which DMA channel the source will be associated. Note that only the first 2 DMA channels have periodic triggering capability.
2. Clear the CHCFG[ENBL] and CHCFG[TRIG] fields of the DMA channel.

3. Ensure that the DMA channel is properly configured in the DMA. The DMA channel may be enabled at this point.
4. Configure the corresponding timer.
5. Select the source to be routed to the DMA channel. Write to the corresponding CHCFG register, ensuring that the CHCFG[ENBL] and CHCFG[TRIG] fields are set.

### NOTE

The following is an example. See chip configuration section for the number of this device's DMA channels that have triggering capability.

To configure source #5 transmit for use with DMA channel 2, with periodic triggering capability:

1. Write 0x00 to CHCFG2 (base address + 0x02).
2. Configure channel 2 in the DMA, including enabling the channel.
3. Configure a timer for the desired trigger interval.
4. Write 0xC5 to CHCFG2 (base address + 0x02).

The following code example illustrates steps 1 and 4 above:

```
In File registers.h:
#define DMAMUX_BASE_ADDR      0xFC084000/* Example only ! */
/* Following example assumes char is 8-bits */
volatile unsigned char *CHCONFIG0 = (volatile unsigned char *) (DMAMUX_BASE_ADDR+0x0000);
volatile unsigned char *CHCONFIG1 = (volatile unsigned char *) (DMAMUX_BASE_ADDR+0x0001);
volatile unsigned char *CHCONFIG2 = (volatile unsigned char *) (DMAMUX_BASE_ADDR+0x0002);
volatile unsigned char *CHCONFIG3 = (volatile unsigned char *) (DMAMUX_BASE_ADDR+0x0003);
volatile unsigned char *CHCONFIG4 = (volatile unsigned char *) (DMAMUX_BASE_ADDR+0x0004);
volatile unsigned char *CHCONFIG5 = (volatile unsigned char *) (DMAMUX_BASE_ADDR+0x0005);
volatile unsigned char *CHCONFIG6 = (volatile unsigned char *) (DMAMUX_BASE_ADDR+0x0006);
volatile unsigned char *CHCONFIG7 = (volatile unsigned char *) (DMAMUX_BASE_ADDR+0x0007);
volatile unsigned char *CHCONFIG8 = (volatile unsigned char *) (DMAMUX_BASE_ADDR+0x0008);
volatile unsigned char *CHCONFIG9 = (volatile unsigned char *) (DMAMUX_BASE_ADDR+0x0009);
volatile unsigned char *CHCONFIG10= (volatile unsigned char *) (DMAMUX_BASE_ADDR+0x000A);
volatile unsigned char *CHCONFIG11= (volatile unsigned char *) (DMAMUX_BASE_ADDR+0x000B);
volatile unsigned char *CHCONFIG12= (volatile unsigned char *) (DMAMUX_BASE_ADDR+0x000C);
volatile unsigned char *CHCONFIG13= (volatile unsigned char *) (DMAMUX_BASE_ADDR+0x000D);
volatile unsigned char *CHCONFIG14= (volatile unsigned char *) (DMAMUX_BASE_ADDR+0x000E);
volatile unsigned char *CHCONFIG15= (volatile unsigned char *) (DMAMUX_BASE_ADDR+0x000F);

In File main.c:
#include "registers.h"
:
:
*CHCONFIG2 = 0x00;
*CHCONFIG2 = 0xC5;
```

To enable a source without periodic triggering:

1. Determine with which DMA channel the source will be associated. Note that only the first 2 DMA channels have periodic triggering capability.
2. Clear the CHCFG[ENBL] and CHCFG[TRIG] fields of the DMA channel.

3. Ensure that the DMA channel is properly configured in the DMA. The DMA channel may be enabled at this point.
4. Select the source to be routed to the DMA channel. Write to the corresponding CHCFG register, ensuring that CHCFG[ENBL] is set while CHCFG[TRIG] is cleared.

### NOTE

The following is an example. See chip configuration section for the number of this device's DMA channels that have triggering capability.

To configure source #5 transmit for use with DMA channel 2, with no periodic triggering capability:

1. Write 0x00 to CHCFG2 (base address + 0x02).
2. Configure channel 2 in the DMA, including enabling the channel.
3. Write 0x85 to CHCFG2 (base address + 0x02).

The following code example illustrates steps 1 and 3 above:

In File registers.h:

```
#define DMAMUX_BASE_ADDR      0xFC084000/* Example only ! */
/* Following example assumes char is 8-bits */
volatile unsigned char *CHCONFIG0 = (volatile unsigned char *) (DMAMUX_BASE_ADDR+0x0000);
volatile unsigned char *CHCONFIG1 = (volatile unsigned char *) (DMAMUX_BASE_ADDR+0x0001);
volatile unsigned char *CHCONFIG2 = (volatile unsigned char *) (DMAMUX_BASE_ADDR+0x0002);
volatile unsigned char *CHCONFIG3 = (volatile unsigned char *) (DMAMUX_BASE_ADDR+0x0003);
volatile unsigned char *CHCONFIG4 = (volatile unsigned char *) (DMAMUX_BASE_ADDR+0x0004);
volatile unsigned char *CHCONFIG5 = (volatile unsigned char *) (DMAMUX_BASE_ADDR+0x0005);
volatile unsigned char *CHCONFIG6 = (volatile unsigned char *) (DMAMUX_BASE_ADDR+0x0006);
volatile unsigned char *CHCONFIG7 = (volatile unsigned char *) (DMAMUX_BASE_ADDR+0x0007);
volatile unsigned char *CHCONFIG8 = (volatile unsigned char *) (DMAMUX_BASE_ADDR+0x0008);
volatile unsigned char *CHCONFIG9 = (volatile unsigned char *) (DMAMUX_BASE_ADDR+0x0009);
volatile unsigned char *CHCONFIG10= (volatile unsigned char *) (DMAMUX_BASE_ADDR+0x000A);
volatile unsigned char *CHCONFIG11= (volatile unsigned char *) (DMAMUX_BASE_ADDR+0x000B);
volatile unsigned char *CHCONFIG12= (volatile unsigned char *) (DMAMUX_BASE_ADDR+0x000C);
volatile unsigned char *CHCONFIG13= (volatile unsigned char *) (DMAMUX_BASE_ADDR+0x000D);
volatile unsigned char *CHCONFIG14= (volatile unsigned char *) (DMAMUX_BASE_ADDR+0x000E);
volatile unsigned char *CHCONFIG15= (volatile unsigned char *) (DMAMUX_BASE_ADDR+0x000F);
```

In File main.c:

```
#include "registers.h"
:
:
*CHCONFIG2 = 0x00;
*CHCONFIG2 = 0x85;
```

To disable a source:

A particular DMA source may be disabled by not writing the corresponding source value into any of the CHCFG registers. Additionally, some module-specific configuration may be necessary. See the appropriate section for more details.

To switch the source of a DMA channel:

1. Disable the DMA channel in the DMA and re-configure the channel for the new source.
2. Clear the CHCFG[ENBL] and CHCFG[TRIG] bits of the DMA channel.
3. Select the source to be routed to the DMA channel. Write to the corresponding CHCFG register, ensuring that the CHCFG[ENBL] and CHCFG[TRIG] fields are set.

To switch DMA channel 8 from source #5 transmit to source #7 transmit:

1. In the DMA configuration registers, disable DMA channel 8 and reconfigure it to handle the transfers to peripheral slot 7. This example assumes channel 8 doesn't have triggering capability.
2. Write 0x00 to CHCFG8 (base address + 0x08).
3. Write 0x87 to CHCFG8 (base address + 0x08). (In this example, setting CHCFG[TRIG] would have no effect, due to the assumption that channel 8 does not support the periodic triggering functionality).

The following code example illustrates steps 2 and 3 above:

```
In File registers.h:
#define DMAMUX_BASE_ADDR      0xFC084000/* Example only ! */
/* Following example assumes char is 8-bits */
volatile unsigned char *CHCONFIG0 = (volatile unsigned char *) (DMAMUX_BASE_ADDR+0x0000);
volatile unsigned char *CHCONFIG1 = (volatile unsigned char *) (DMAMUX_BASE_ADDR+0x0001);
volatile unsigned char *CHCONFIG2 = (volatile unsigned char *) (DMAMUX_BASE_ADDR+0x0002);
volatile unsigned char *CHCONFIG3 = (volatile unsigned char *) (DMAMUX_BASE_ADDR+0x0003);
volatile unsigned char *CHCONFIG4 = (volatile unsigned char *) (DMAMUX_BASE_ADDR+0x0004);
volatile unsigned char *CHCONFIG5 = (volatile unsigned char *) (DMAMUX_BASE_ADDR+0x0005);
volatile unsigned char *CHCONFIG6 = (volatile unsigned char *) (DMAMUX_BASE_ADDR+0x0006);
volatile unsigned char *CHCONFIG7 = (volatile unsigned char *) (DMAMUX_BASE_ADDR+0x0007);
volatile unsigned char *CHCONFIG8 = (volatile unsigned char *) (DMAMUX_BASE_ADDR+0x0008);
volatile unsigned char *CHCONFIG9 = (volatile unsigned char *) (DMAMUX_BASE_ADDR+0x0009);
volatile unsigned char *CHCONFIG10= (volatile unsigned char *) (DMAMUX_BASE_ADDR+0x000A);
volatile unsigned char *CHCONFIG11= (volatile unsigned char *) (DMAMUX_BASE_ADDR+0x000B);
volatile unsigned char *CHCONFIG12= (volatile unsigned char *) (DMAMUX_BASE_ADDR+0x000C);
volatile unsigned char *CHCONFIG13= (volatile unsigned char *) (DMAMUX_BASE_ADDR+0x000D);
volatile unsigned char *CHCONFIG14= (volatile unsigned char *) (DMAMUX_BASE_ADDR+0x000E);
volatile unsigned char *CHCONFIG15= (volatile unsigned char *) (DMAMUX_BASE_ADDR+0x000F);

In File main.c:
#include "registers.h"
:
:
*CHCONFIG8 = 0x00;
*CHCONFIG8 = 0x87;
```



# Chapter 23

## DMA Controller Module

### 23.1 Introduction

#### NOTE

For the chip-specific implementation details of this module's instances see the chip configuration information.

This chapter describes the direct memory access (DMA) controller module. It provides an overview of the module and describes in detail its signals and programming model. The latter sections of this chapter describe operations, features, and supported data transfer modes in detail.

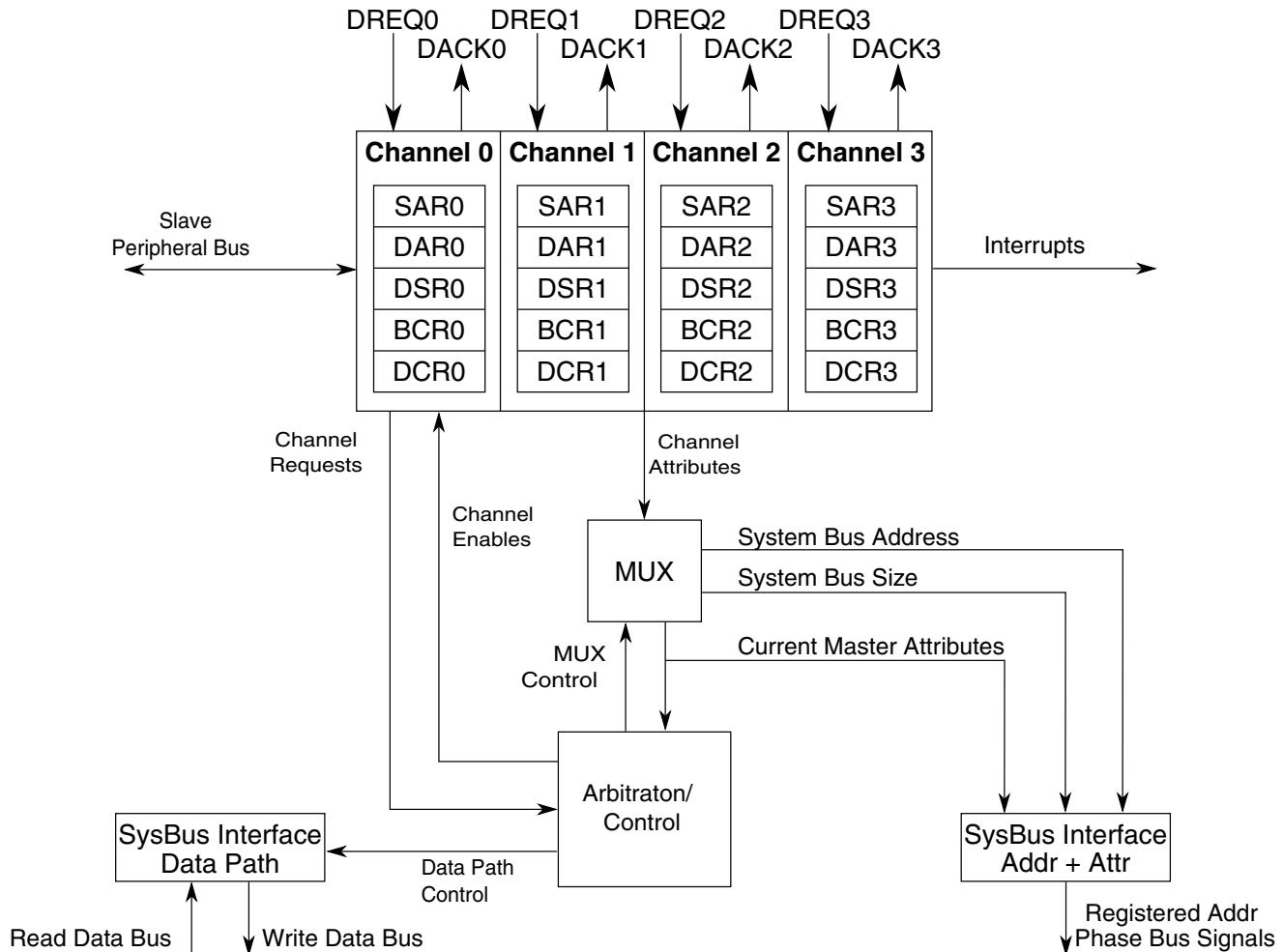
#### Note

The designation *n* is used throughout this section to refer to registers or signals associated with one of the four identical DMA channels: DMA0, DMA1, DMA2, or DMA3.

#### 23.1.1 Overview

The DMA controller module enables fast transfers of data, providing an efficient way to move blocks of data with minimal processor interaction. The DMA module, shown in the following figure, has four channels that allow 8-bit, 16-bit, or 32-bit data transfers. Each channel has a dedicated source address register (SAR<sub>*n*</sub>), destination address register (DAR<sub>*n*</sub>), status register (DSR<sub>*n*</sub>), byte count register (BCR<sub>*n*</sub>), and control register (DCR<sub>*n*</sub>). Collectively, the combined program-visible registers associated with each channel define a transfer control descriptor (TCD). All transfers are dual address, moving data from a source memory location to a destination memory location with the module operating as a 32-bit bus master connected to the system bus. The programming model is accessed through a 32-bit connection with the slave peripheral bus. DMA data transfers may be explicitly initiated by software or by peripheral hardware requests.

The following figure is a simplified block diagram of the 4-channel DMA controller.



**Figure 23-1. 4-Channel DMA Block Diagram**

The terms *peripheral request* and *DREQ* refer to a DMA request from one of the on-chip peripherals or package pins. The DMA provides hardware handshake signals: either a DMA acknowledge (DACK) or a done indicator back to the peripheral.

### 23.1.2 Features

The DMA controller module features:

- Four independently programmable DMA controller channels
- Dual-address transfers via 32-bit master connection to the system bus
- Data transfers in 8-, 16-, or 32-bit blocks

- Continuous-mode or cycle-steal transfers from software or peripheral initiation
- Automatic hardware acknowledge/done indicator from each channel
- Independent source and destination address registers
- Optional modulo addressing and automatic updates of source and destination addresses
- Independent transfer sizes for source and destination
- Optional auto-alignment feature for source or destination accesses
- Optional automatic single or double channel linking
- Programming model accessed via 32-bit slave peripheral bus
- Channel arbitration on transfer boundaries using fixed priority scheme

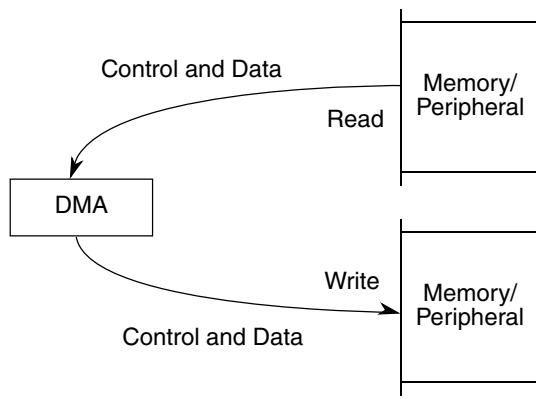
## 23.2 DMA Transfer Overview

The DMA module can move data within system memory (including memory and peripheral devices) with minimal processor intervention, greatly improving overall system performance. The DMA module consists of four independent, functionally equivalent channels, so references to DMA in this chapter apply to any of the channels. It is not possible to address all four channels at once.

As soon as a channel has been initialized, it may be started by setting DCRn[START] or a properly-selected peripheral DMA request, depending on the status of DCRn[ERQ].

The DMA controller supports dual-address transfers using its bus master connection to the system bus. The DMA channels support transfers up to 32 data bits in size and have the same memory map addressability as the processor.

- Dual-address transfers—A dual-address transfer consists of a read followed by a write and is initiated by a request using the DCRn[START] bit or by a peripheral DMA request. The read data is temporarily held in the DMA channel hardware until the write operation. Two types of single transfers occur: a read from a source address followed by a write to a destination address. See the following figure.

**Figure 23-2. Dual-Address Transfer**

Any operation involving a DMA channel follows the same three steps:

1. Channel initialization—The transfer control descriptor, contained in the channel registers, is loaded with address pointers, a byte-transfer count, and control information using accesses from the slave peripheral bus.
2. Data transfer—The DMA accepts requests for data transfers. Upon receipt of a request, it provides address and bus control for the transfers via its master connection to the system bus and temporary storage for the read data. The channel performs one or more source read and destination write data transfers.
3. Channel termination—Occurs after the operation is finished successfully or due to an error. The channel indicates the operation status in the channel's DSR, described in the definitions of the DMA Status Registers (DSRn) and Byte Count Registers (BCRn).

### 23.3 Memory Map and Registers

Descriptions of each register and its bit assignments follow. Modifying DMA control registers during a transfer can result in undefined operation. The following table shows the mapping of DMA controller registers. The DMA programming model is accessed via the slave peripheral bus. The concatenation of the source and destination address registers, the status and byte count register, and the control register create a 128-bit transfer control descriptor (TCD) that defines the operation of each DMA channel.

## DMA memory map

| Absolute address (hex) | Register name  | Width (in bits) | Access | Reset value | Section/page               |
|------------------------|--|-----------------|--------|-------------|----------------------------|
| 4000_8100              | Source Address Register (DMA_SAR0)                       | 32              | R/W    | 0000_0000h  | <a href="#">23.3.1/353</a> |
| 4000_8104              | Destination Address Register (DMA_DAR0)                  | 32              | R/W    | 0000_0000h  | <a href="#">23.3.2/354</a> |
| 4000_8108              | DMA Status Register / Byte Count Register (DMA_DSR_BCR0) | 32              | R/W    | 0000_0000h  | <a href="#">23.3.3/355</a> |
| 4000_810C              | DMA Control Register (DMA_DCR0)                          | 32              | R/W    | 0000_0000h  | <a href="#">23.3.4/357</a> |
| 4000_8110              | Source Address Register (DMA_SAR1)                       | 32              | R/W    | 0000_0000h  | <a href="#">23.3.1/353</a> |
| 4000_8114              | Destination Address Register (DMA_DAR1)                  | 32              | R/W    | 0000_0000h  | <a href="#">23.3.2/354</a> |
| 4000_8118              | DMA Status Register / Byte Count Register (DMA_DSR_BCR1) | 32              | R/W    | 0000_0000h  | <a href="#">23.3.3/355</a> |
| 4000_811C              | DMA Control Register (DMA_DCR1)                          | 32              | R/W    | 0000_0000h  | <a href="#">23.3.4/357</a> |
| 4000_8120              | Source Address Register (DMA_SAR2)                       | 32              | R/W    | 0000_0000h  | <a href="#">23.3.1/353</a> |
| 4000_8124              | Destination Address Register (DMA_DAR2)                  | 32              | R/W    | 0000_0000h  | <a href="#">23.3.2/354</a> |
| 4000_8128              | DMA Status Register / Byte Count Register (DMA_DSR_BCR2) | 32              | R/W    | 0000_0000h  | <a href="#">23.3.3/355</a> |
| 4000_812C              | DMA Control Register (DMA_DCR2)                          | 32              | R/W    | 0000_0000h  | <a href="#">23.3.4/357</a> |
| 4000_8130              | Source Address Register (DMA_SAR3)                       | 32              | R/W    | 0000_0000h  | <a href="#">23.3.1/353</a> |
| 4000_8134              | Destination Address Register (DMA_DAR3)                  | 32              | R/W    | 0000_0000h  | <a href="#">23.3.2/354</a> |
| 4000_8138              | DMA Status Register / Byte Count Register (DMA_DSR_BCR3) | 32              | R/W    | 0000_0000h  | <a href="#">23.3.3/355</a> |
| 4000_813C              | DMA Control Register (DMA_DCR3)                          | 32              | R/W    | 0000_0000h  | <a href="#">23.3.4/357</a> |

### 23.3.1 Source Address Register (DMA\_SAR $n$ )

## Restriction

For this register:

- Only 32-bit writes are allowed. 16-bit and 8-bit writes result in a bus error.
  - Only four values are allowed to be written to bits 31-20 of this register. A write of any other value to these bits causes a configuration error when the channel starts to execute.

For more information about the configuration error, see the description of the **CE** field of DSR.

Address: 4000 8000h base + 100h offset + (16d x i), where i=0d to 3d

**DMA\_SAR $n$  field descriptions**

| Field       | Description  |
|-------------|--|
| 31–0<br>SAR | <p>Each SAR contains the byte address used by the DMA controller to read data. The SAR<math>n</math> is typically aligned on a 0-modulo-ssize boundary—that is, on the natural alignment of the source data.</p> <p><b>Restriction:</b> Bits 31-20 of this register must be written with one of only four allowed values. Each of these four allowed values corresponds to a valid region of the device's memory map. The allowed values are:</p> <ul style="list-style-type: none"> <li>• 0x000x_xxxx</li> <li>• 0x1FFx_xxxx</li> <li>• 0x200x_xxxx</li> <li>• 0x400x_xxxx</li> </ul> <p>After being written with one of the allowed values, bits 31-20 read back as the written value. After being written with any other value, bits 31-20 read back as an indeterminate value.</p> |

**23.3.2 Destination Address Register (DMA\_DAR $n$ )****Restriction**

For this register:

- Only 32-bit writes are allowed. 16-bit and 8-bit writes result in a bus error.
  - Only four values are allowed to be written to bits 31-20 of this register. A write of any other value to these bits causes a configuration error when the channel starts to execute.
- For more information about the configuration error, see the description of the **CE** field of DSR.

Address: 4000\_8000h base + 104h offset + (16d × i), where i=0d to 3d

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|-----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|---|---|---|---|---|---|---|---|---|---|
| R   |    |    |    |    |    |    |    |    |    |    |    |    |    |    |    |    |    |    |    |    |    |    |   |   |   |   |   |   |   |   |   |   |
| W   |    |    |    |    |    |    |    |    |    |    |    |    |    |    |    |    |    |    |    |    |    |    |   |   |   |   |   |   |   |   |   |   |

Reset 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 | 0

**DMA\_DAR $n$  field descriptions**

| Field       | Description  |
|-------------|--|
| 31–0<br>DAR | <p>Each DAR contains the byte address used by the DMA controller to write data. The DAR<math>n</math> is typically aligned on a 0-modulo-dsize boundary—that is, on the natural alignment of the destination data.</p> <p><b>Restriction:</b> Bits 31-20 of this register must be written with one of only four allowed values. Each of these four allowed values corresponds to a valid region of the device's memory map. The allowed values are:</p> <ul style="list-style-type: none"> <li>• 0x000x_xxxx</li> <li>• 0x1FFx_xxxx</li> <li>• 0x200x_xxxx</li> <li>• 0x400x_xxxx</li> </ul> |

**DMA\_DARn field descriptions (continued)**

| Field | Description   |
|-------|---|
|       | After being written with one of the allowed values, bits 31-20 read back as the written value.<br>After being written with any other value, bits 31-20 read back as an indeterminate value. |

**23.3.3 DMA Status Register / Byte Count Register (DMA\_DSR\_BCRn)**

DSR and BCR are two logical registers that occupy one 32-bit address. DSRn occupies bits 31–24, and BCRn occupies bits 23–0. DSRn contains flags indicating the channel status, and BCRn contains the number of bytes yet to be transferred for a given block.

On the successful completion of the write transfer, BCRn decrements by 1, 2, or 4 for 8-bit, 16-bit, or 32-bit accesses, respectively. BCRn is cleared if a 1 is written to DSR[DONE].

In response to an event, the DMA controller writes to the appropriate DSRn bit. Only a write to DSRn[DONE] results in action. DSRn[DONE] is set when the block transfer is complete.

When a transfer sequence is initiated and BCRn[BCR] is not a multiple of 4 or 2 when the DMA is configured for 32-bit or 16-bit transfers, respectively, DSRn[CE] is set and no transfer occurs.

Address: 4000\_8000h base + 108h offset + (16d × i), where i=0d to 3d

| Bit   | 31 | 30 | 29  | 28  | 27 | 26  | 25  | 24   | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|-------|----|----|-----|-----|----|-----|-----|------|----|----|----|----|----|----|----|----|
| R     | 0  | CE | BES | BED | 0  | REQ | BSY | DONE |    |    |    |    |    |    |    |    |
| W     |    |    |     |     |    |     |     | w1c  |    |    |    |    |    |    |    |    |
| Reset | 0  | 0  | 0   | 0   | 0  | 0   | 0   | 0    | 0  | 0  | 0  | 0  | 0  | 0  | 0  | 0  |
| Bit   | 15 | 14 | 13  | 12  | 11 | 10  | 9   | 8    | 7  | 6  | 5  | 4  | 3  | 2  | 1  | 0  |
| R     |    |    |     |     |    |     |     |      |    |    |    |    |    |    |    |    |
| W     |    |    |     |     |    |     |     |      |    |    |    |    |    |    |    |    |
| Reset | 0  | 0  | 0   | 0   | 0  | 0   | 0   | 0    | 0  | 0  | 0  | 0  | 0  | 0  | 0  | 0  |

**DMA\_DSR\_BCRn field descriptions**

| Field          | Description   |
|----------------|---|
| 31<br>Reserved | This field is reserved.<br>This read-only field is reserved and always has the value 0.   |
| 30<br>CE       | Configuration error<br><br>Any of the following conditions causes a configuration error: <ul style="list-style-type: none"> <li>• BCR, SAR, or DAR does not match the requested transfer size.</li> <li>• A value greater than 0F_FFFFh is written to BCR.</li> <li>• Bits 31-20 of SAR or DAR are written with a value other than one of the allowed values. See <a href="#">SAR</a> and <a href="#">DAR</a>.</li> <li>• SSIZE or DSIZE is set to an unsupported value.</li> <li>• BCR equals 0 when the DMA receives a start condition.</li> </ul><br>CE is cleared at hardware reset or by writing a 1 to the DONE bit.<br><br>0 No configuration error exists.<br>1 A configuration error has occurred. |
| 29<br>BES      | Bus error on source<br><br>BES is cleared at hardware reset or by writing a 1 to the DONE bit.<br><br>0 No bus error occurred.<br>1 The DMA channel terminated with a bus error during the read portion of a transfer.  |
| 28<br>BED      | Bus error on destination<br><br>BED is cleared at hardware reset or by writing a 1 to the DONE bit.<br><br>0 No bus error occurred.<br>1 The DMA channel terminated with a bus error during the write portion of a transfer.  |
| 27<br>Reserved | This field is reserved.<br>This read-only field is reserved and always has the value 0.   |
| 26<br>REQ      | Request<br><br>0 No request is pending or the channel is currently active. Cleared when the channel is selected.<br>1 The DMA channel has a transfer remaining and the channel is not selected.   |
| 25<br>BSY      | Busy<br><br>0 DMA channel is inactive. Cleared when the DMA has finished the last transaction.<br>1 BSY is set the first time the channel is enabled after a transfer is initiated.   |
| 24<br>DONE     | Transactions done<br><br>Set when all DMA controller transactions complete as determined by transfer count, or based on error conditions. When BCR reaches zero, DONE is set when the final transfer completes successfully. DONE can also be used to abort a transfer by resetting the status bits. When a transfer completes, software must clear DONE before reprogramming the DMA.<br><br>0 DMA transfer is not yet complete. Writing a 0 has no effect.<br>1 DMA transfer completed. Writing a 1 to this bit clears all DMA status bits and should be used in an interrupt service routine to clear the DMA interrupt and error bits.  |
| 23–0<br>BCR    | This field contains the number of bytes yet to be transferred for a given block.<br><br><b>Restriction:</b> BCR must be written with a value equal to or less than 0F_FFFFh. After being written with a value in this range, bits 23-20 of BCR read back as 1110b. A write to BCR of a value  |

*Table continues on the next page...*

## DMA\_DSR\_BCR*n* field descriptions (continued)

| Field | Description   |
|-------|---|
|       | greater than 0F_FFFFh causes a configuration error when the channel starts to execute.<br>After being written with a value in this range, bits 23-20 of BCR read back as 1111b. |

**DMA\_DCRn field descriptions (continued)**

| Field             | Description   |
|-------------------|---|
| 29<br>CS          | Cycle steal<br><br>0 DMA continuously makes read/write transfers until the BCR decrements to 0.<br>1 Forces a single read/write transfer per request.   |
| 28<br>AA          | Auto-align<br><br>AA and SIZE bits determine whether the source or destination is auto-aligned; that is, transfers are optimized based on the address and size.<br><br>0 Auto-align disabled<br>1 If SSIZE indicates a transfer no smaller than DSIZE, source accesses are auto-aligned; otherwise, destination accesses are auto-aligned. Source alignment takes precedence over destination alignment. If auto-alignment is enabled, the appropriate address register increments, regardless of DINC or SINC. |
| 27–25<br>Reserved | This field is reserved.<br>This read-only field is reserved and always has the value 0.   |
| 24<br>Reserved    | This field is reserved.<br><br><b>CAUTION:</b> Must be written as zero; otherwise, undefined behavior results.  |
| 23<br>EADREQ      | Enable asynchronous DMA requests<br><br>Enables the channel to support asynchronous DREQs while the MCU is in Stop mode.<br><br>0 Disabled<br>1 Enabled   |
| 22<br>SINC        | Source increment<br><br>Controls whether the source address increments after each successful transfer.<br><br>0 No change to SAR after a successful transfer.<br>1 The SAR increments by 1, 2, 4 as determined by the transfer size.  |
| 21–20<br>SSIZE    | Source size<br><br>Determines the data size of the source bus cycle for the DMA controller.<br><br>00 32-bit<br>01 8-bit<br>10 16-bit<br>11 Reserved (generates a configuration error (DSRn[CE]) if incorrectly specified at time of channel activation)  |
| 19<br>DINC        | Destination increment<br><br>Controls whether the destination address increments after each successful transfer.<br><br>0 No change to the DAR after a successful transfer.<br>1 The DAR increments by 1, 2, 4 depending upon the size of the transfer.   |
| 18–17<br>DSIZE    | Destination size<br><br>Determines the data size of the destination bus cycle for the DMA controller.<br><br>00 32-bit<br>01 8-bit  |

*Table continues on the next page...*

**DMA\_DCRn field descriptions (continued)**

| Field      | Description  |
|------------|--|
|            | 10 16-bit<br>11 Reserved (generates a configuration error (DSRn[CE]) if incorrectly specified at time of channel activation)   |
| 16 START   | Start transfer<br>0 DMA inactive<br>1 The DMA begins the transfer in accordance to the values in the TCDn. START is cleared automatically after one module clock and always reads as logic 0.  |
| 15–12 SMOD | Source address modulo<br><p>Defines the size of the source data circular buffer used by the DMA Controller. If enabled (SMOD is non-zero), the buffer base address is located on a boundary of the buffer size. The value of this boundary is based upon the initial source address (SAR). The base address should be aligned to a 0-modulo-(circular buffer size) boundary. Misaligned buffers are not possible. The boundary is forced to the value determined by the upper address bits in the field selection.</p> <ul style="list-style-type: none"> <li>0000 Buffer disabled</li> <li>0001 Circular buffer size is 16 bytes</li> <li>0010 Circular buffer size is 32 bytes</li> <li>0011 Circular buffer size is 64 bytes</li> <li>0100 Circular buffer size is 128 bytes</li> <li>0101 Circular buffer size is 256 bytes</li> <li>0110 Circular buffer size is 512 bytes</li> <li>0111 Circular buffer size is 1 KB</li> <li>1000 Circular buffer size is 2 KB</li> <li>1001 Circular buffer size is 4 KB</li> <li>1010 Circular buffer size is 8 KB</li> <li>1011 Circular buffer size is 16 KB</li> <li>1100 Circular buffer size is 32 KB</li> <li>1101 Circular buffer size is 64 KB</li> <li>1110 Circular buffer size is 128 KB</li> <li>1111 Circular buffer size is 256 KB</li> </ul> |
| 11–8 DMOD  | Destination address modulo<br><p>Defines the size of the destination data circular buffer used by the DMA Controller. If enabled (DMOD value is non-zero), the buffer base address is located on a boundary of the buffer size. The value of this boundary depends on the initial destination address (DAR). The base address should be aligned to a 0-modulo-(circular buffer size) boundary. Misaligned buffers are not possible. The boundary is forced to the value determined by the upper address bits in the field selection.</p> <ul style="list-style-type: none"> <li>0000 Buffer disabled</li> <li>0001 Circular buffer size is 16 bytes</li> <li>0010 Circular buffer size is 32 bytes</li> <li>0011 Circular buffer size is 64 bytes</li> <li>0100 Circular buffer size is 128 bytes</li> <li>0101 Circular buffer size is 256 bytes</li> <li>0110 Circular buffer size is 512 bytes</li> <li>0111 Circular buffer size is 1 KB</li> <li>1000 Circular buffer size is 2 KB</li> <li>1001 Circular buffer size is 4 KB</li> </ul>  |

*Table continues on the next page...*

**DMA\_DCRn field descriptions (continued)**

| Field      | Description  |
|------------|--|
|            | <p>1010 Circular buffer size is 8 KB<br/>     1011 Circular buffer size is 16 KB<br/>     1100 Circular buffer size is 32 KB<br/>     1101 Circular buffer size is 64 KB<br/>     1110 Circular buffer size is 128 KB<br/>     1111 Circular buffer size is 256 KB</p>   |
| 7 D_REQ    | <p>Disable request</p> <p>DMA hardware automatically clears the corresponding DCRn[ERQ] bit when the byte count register reaches zero.</p> <p>0 ERQ bit is not affected.<br/>     1 ERQ bit is cleared when the BCR is exhausted.</p>  |
| 6 Reserved | <p>This field is reserved.</p> <p>This read-only field is reserved and always has the value 0.</p>   |
| 5–4 LINKCC | <p>Link channel control</p> <p>Allows DMA channels to have their transfers linked. The current DMA channel triggers a DMA request to the linked channels (LCH1 or LCH2) depending on the condition described by the LINKCC bits.</p> <p>If not in cycle steal mode (DCRn[CS]=0) and LINKCC equals 01 or 10, no link to LCH1 occurs.</p> <p>If LINKCC equals 01, a link to LCH1 is created after each cycle-steal transfer performed by the current DMA channel is completed. As the last cycle-steal is performed and the BCR reaches zero, then the link to LCH1 is closed and a link to LCH2 is created.</p> <p>00 No channel-to-channel linking<br/>     01 Perform a link to channel LCH1 after each cycle-steal transfer followed by a link to LCH2 after the BCR decrements to zero<br/>     10 Perform a link to channel LCH1 after each cycle-steal transfer<br/>     11 Perform a link to channel LCH1 after the BCR decrements to zero</p> |
| 3–2 LCH1   | <p>Link channel 1</p> <p>Indicates the DMA channel assigned as link channel 1. The link channel number cannot be the same as the currently executing channel, and generates a configuration error if this is attempted (DSRn[CE] is set).</p> <p>00 DMA Channel 0<br/>     01 DMA Channel 1<br/>     10 DMA Channel 2<br/>     11 DMA Channel 3</p>  |
| 1–0 LCH2   | <p>Link channel 2</p> <p>Indicates the DMA channel assigned as link channel 2. The link channel number cannot be the same as the currently executing channel, and generates a configuration error if this is attempted (DSRn[CE] is set).</p> <p>00 DMA Channel 0<br/>     01 DMA Channel 1<br/>     10 DMA Channel 2<br/>     11 DMA Channel 3</p>  |

## 23.4 Functional Description

In the following discussion, the term DMA request implies that  $DCRn[START]$  is set, or  $DCRn[ERQ]$  is set and then followed by assertion of the properly selected DMA peripheral request. The START bit is cleared when the channel is activated.

Before initiating a dual-address access, the DMA module verifies that  $DCRn[SSIZE]$  and  $DCRn[DSIZE]$  are consistent with the source and destination addresses. If they are not consistent, the configuration error bit,  $DSRn[CE]$ , is set. If misalignment is detected, no transfer occurs,  $DSRn[CE]$  is set, and, depending on the DCR configuration, an interrupt event may be issued. If the auto-align bit,  $DCRn[AA]$ , is set, error checking is performed on the appropriate registers.

A read/write transfer sequence reads data from the source address and writes it to the destination address. The number of bytes transferred is the largest of the sizes specified by  $DCRn[SSIZE]$  and  $DCRn[DSIZE]$  in the DMA Control Registers ( $DCRn$ ).

Source and destination address registers ( $SARn$  and  $DARn$ ) can be programmed in the  $DCRn$  to increment at the completion of a successful transfer.

### 23.4.1 Transfer Requests (Cycle-Steal and Continuous Modes)

The DMA channel supports software-initiated or peripheral-initiated requests. A request is issued by setting  $DCRn[START]$  or when the selected peripheral request asserts and  $DCRn[ERQ]$  is set. Setting  $DCRn[ERQ]$  enables recognition of the peripheral DMA requests. Selecting between cycle-steal and continuous modes minimizes bus usage for either type of request.

- Cycle-steal mode ( $DCRn[CS] = 1$ )—Only one complete transfer from source to destination occurs for each request. If  $DCRn[ERQ]$  is set, the request is peripheral initiated. A software-initiated request is enabled by setting  $DCRn[START]$ .
- Continuous mode ( $DCRn[CS] = 0$ )—After a software-initiated or peripheral request, the DMA continuously transfers data until  $BCRn$  reaches zero. The DMA performs the specified number of transfers, then retires the channel.

In either mode, the crossbar switch performs independent arbitration on each slave port after each transaction.

## 23.4.2 Channel Initialization and Startup

Before a data transfer starts, the channel's transfer control descriptor must be initialized with information describing configuration, request-generation method, and pointers to the data to be moved.

### 23.4.2.1 Channel Prioritization

The four DMA channels are prioritized based on number, with channel 0 having highest priority and channel 3 having the lowest, that is, channel 0 > channel 1 > channel 2 > channel 3.

Simultaneous peripheral requests activate the channels based on this priority order. Once activated, a channel runs to completion as defined by  $DCRn[CS]$  and  $BCRn$ .

### 23.4.2.2 Programming the DMA Controller Module

#### CAUTION

During a channel's execution, writes to programming model registers can corrupt the data transfer. The DMA module itself does not have a mechanism to prevent writes to registers during a channel's execution.

General guidelines for programming the DMA are:

- $TCDn$  is initialized.
- $SARn$  is loaded with the source (read) address. If the transfer is from a peripheral device to memory or to another peripheral, the source address is the location of the peripheral data register. If the transfer is from memory to a peripheral device or to memory, the source address is the starting address of the data block. This can be any appropriately aligned address.
- $DARn$  is initialized with the destination (write) address. If the transfer is from a peripheral device to memory, or from memory to memory,  $DARn$  is loaded with the starting address of the data block to be written. If the transfer is from memory to a peripheral device, or from a peripheral device to a peripheral device,  $DARn$  is loaded with the address of the peripheral data register. This address can be any appropriately aligned address.

- $SAR_n$  and  $DAR_n$  change after each data transfer depending on  $DCR_n[SSIZE]$ ,  $DSIZE$ ,  $SINC$ ,  $DINC$ ,  $SMOD$ ,  $DMOD$ ] and the starting addresses. Increment values can be 1, 2, or 4 for 8-bit, 16-bit, or 32-bit transfers, respectively. If the address register is programmed to remain unchanged, the register is not incremented after the data transfer.
- $BCR_n[BCR]$  must be loaded with the total number of bytes to be transferred. It is decremented by 1, 2, or 4 at the end of each transfer, depending on the transfer size.  $DSR_n[DONE]$  must be cleared for channel startup.
- After the channel has been initialized, it may be started by setting  $DCR_n[START]$  or a properly selected peripheral DMA request, depending on the status of  $DCR_n[ERQ]$ . For a software-initiated transfer, the channel can be started by setting  $DCR_n[START]$  as part of a single 32-bit write to the last 32 bits of the  $TCD_n$ ; that is, it is not required to write the  $DCR_n$  with  $START$  cleared and then perform a second write to explicitly set  $START$ .
- Programming the channel for a software-initiated request causes the channel to request the system bus and start transferring data immediately. If the channel is programmed for peripheral-initiated request, a properly selected peripheral DMA request must be asserted before the channel begins the system bus transfers.
- The hardware can automatically clear  $DCR_n[ERQ]$ , disabling the peripheral request, when  $BCR_n$  reaches zero by setting  $DCR_n[D\_REQ]$ .
- Changes to  $DCR_n$  are effective immediately while the channel is active. To avoid problems with changing a DMA channel setup, write a one to  $DSR_n[DONE]$  to stop the DMA channel.

### 23.4.3 Dual-Address Data Transfer Mode

Each channel supports dual-address transfers. Dual-address transfers consist of a source data read and a destination data write. The DMA controller module begins a dual-address transfer sequence after a DMA request. If no error condition exists,  $DSR_n[REQ]$  is set.

- Dual-address read—The DMA controller drives the  $SAR_n$  value onto the system address bus. If  $DCR_n[SINC]$  is set, the  $SAR_n$  increments by the appropriate number of bytes upon a successful read cycle. When the appropriate number of read cycles complete (multiple reads if the destination size is larger than the source), the DMA initiates the write portion of the transfer.

If a termination error occurs,  $DSR_n[BES, DONE]$  are set and DMA transactions stop.

- Dual-address write—The DMA controller drives the  $DAR_n$  value onto the system address bus. When the appropriate number of write cycles complete (multiple writes if the source size is larger than the destination),  $DAR_n$  increments by the appropriate number of bytes if  $DCR_n[DINC]$  is set.  $BCR_n$  decrements by the appropriate number of bytes.  $DSR_n[DONE]$  is set when  $BCR_n$  reaches zero. If the  $BCR_n$  is greater than zero, another read/write transfer is initiated if continuous mode is enabled ( $DCR_n[CS] = 0$ ).

If a termination error occurs,  $DSR_n[BED, DONE]$  are set and DMA transactions stop.

### 23.4.4 Advanced Data Transfer Controls: Auto-Alignment

Typically, auto-alignment for DMA transfers applies for transfers of large blocks of data. As a result, it does not apply for peripheral-initiated cycle-steal transfers.

Auto-alignment allows block transfers to occur at the optimal size based on the address, byte count, and programmed size. To use this feature,  $DCR_n[AA]$  must be set. The source is auto-aligned if  $DCR_n[SSIZE]$  indicates a transfer size larger than  $DCR_n[DSIZE]$ . Source alignment takes precedence over the destination when the source and destination sizes are equal. Otherwise, the destination is auto-aligned. The address register chosen for alignment increments regardless of the increment value. Configuration error checking is performed on registers not chosen for alignment.

If  $BCR_n$  is greater than 16, the address determines transfer size. Transfers of 8 bits, 16 bits, or 32 bits are transferred until the address is aligned to the programmed size boundary, at which time accesses begin using the programmed size. If  $BCR_n$  is less than 16 at the start of a transfer, the number of bytes remaining dictates transfer size.

Consider this example:

- AA equals 1.
- $SAR_n$  equals 0x2000\_0001.
- $BCR_n$  equals 0x00\_00F0.
- SSIZE equals 00 (32 bits).
- DSIZE equals 01 (8 bits).

Because  $SSIZE > DSIZE$ , the source is auto-aligned. Error checking is performed on destination registers. The access sequence is as follows:

1. Read 1 byte from 0x2000\_0001, increment  $SAR_n$ , write 1 byte (using  $DAR_n$ ).
2. Read 2 bytes from 0x2000\_0002, increment  $SAR_n$ , write 2 bytes.

3. Read 4 bytes from 0x2000\_0004, increment SAR $n$ , write 4 bytes.
4. Repeat 4-byte operations until SAR $n$  equals 0x2000\_00F0.
5. Read byte from 0x2000\_00F0, increment SAR $n$ , write byte.

If DSIZE is another size, data writes are optimized to write the largest size allowed based on the address, but not exceeding the configured size.

### 23.4.5 Termination

An unsuccessful transfer can terminate for one of the following reasons:

- Error conditions—When the DMA encounters a read or write cycle that terminates with an error condition, DSR $n$ [BES] is set for a read and DSR $n$ [BED] is set for a write before the transfer is halted. If the error occurred in a write cycle, data in the internal holding registers is lost.
- Interrupts—If DCR $n$ [EINT] is set, the DMA drives the appropriate interrupt request signal. The processor can read DSR $n$  to determine whether the transfer terminated successfully or with an error. DSR $n$ [DONE] is then written with a one to clear the interrupt, the DONE, and error status bits.



# **Chapter 24**

## **Multipurpose Clock Generator (MCG)**

### **24.1 Introduction**

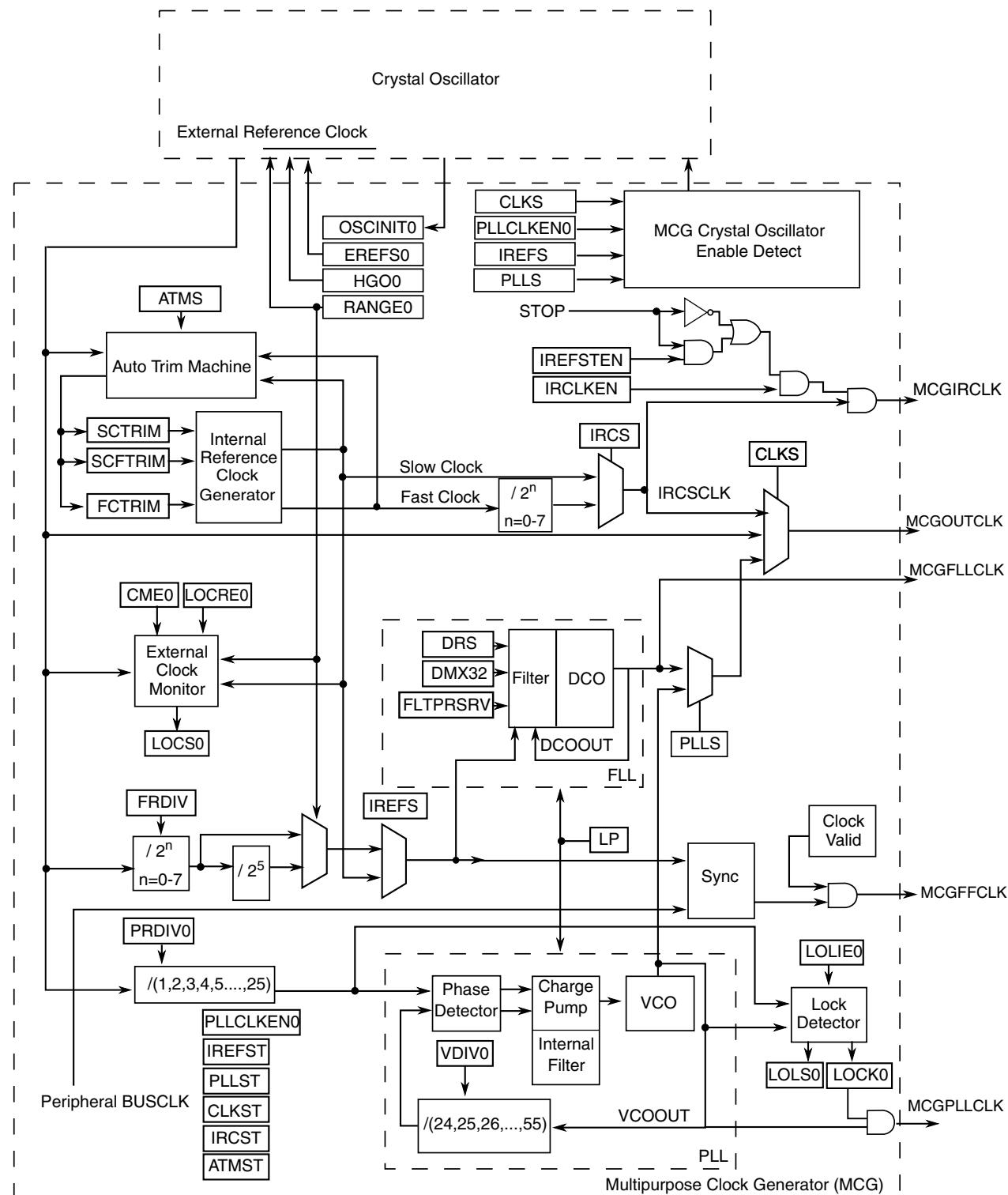
#### **NOTE**

For the chip-specific implementation details of this module's instances see the chip configuration information.

The multipurpose clock generator (MCG) module provides several clock source choices for the MCU. The module contains a frequency-locked loop (FLL) and a phase-locked

- Internal or external reference clock can be used as the FLL source.
- Can be used as a clock source for other on-chip peripherals.
- Phase-locked loop (PLL):
  - Voltage-controlled oscillator (VCO)
  - External reference clock is used as the PLL source.
  - Modulo VCO frequency divider
  - Phase/Frequency detector
  - Integrated loop filter
  - Can be used as a clock source for other on-chip peripherals.
- Internal reference clock generator:
  - Slow clock with nine trim bits for accuracy
  - Fast clock with four trim bits
  - Can be used as source clock for the FLL. In FEI mode, only the slow Internal Reference Clock (IRC) can be used as the FLL source.
  - Either the slow or the fast clock can be selected as the clock source for the MCU.
  - Can be used as a clock source for other on-chip peripherals.
- Control signals for the MCG external reference low power oscillator clock generators are provided:
  - HGO0, RANGE0, EREFS0
- External clock from the Crystal Oscillator :
  - Can be used as a source for the FLL and/or the PLL.
  - Can be selected as the clock source for the MCU.
- External clock monitor with reset and interrupt request capability to check for external clock failure when running in FBE, PEE, BLPE, or FEE modes
- Lock detector with interrupt request capability for use with the PLL
- Internal Reference Clocks Auto Trim Machine (ATM) capability using an external clock as a reference
- Reference dividers for both the FLL and the PLL are provided

- Reference dividers for the Fast Internal Reference Clock are provided
- MCG PLL Clock (MCGPLLCLK) is provided as a clock source for other on-chip peripherals
- MCG FLL Clock (MCGFLLCLK) is provided as a clock source for other on-chip peripherals
- MCG Fixed Frequency Clock (MCFFCLK) is provided as a clock source for other on-chip peripherals
- MCG Internal Reference Clock (MCGIRCLK) is provided as a clock source for other on-chip peripherals



**Figure 24-1. Multipurpose Clock Generator (MCG) block diagram**

## 24.1.2 Modes of Operation

The MCG has the following modes of operation: FEI, FEE, FBI, FBE, PBE, PEE, BLPI, BLPE, and Stop. For details, see [MCG modes of operation](#).

## 24.2 External Signal Description

There are no MCG signals that connect off chip.

## 24.3 Memory Map/Register Definition

This section includes the memory map and register definition.

The MCG registers can only be written when in supervisor mode. Write accesses when in user mode will result in a bus error. Read accesses may be performed in both supervisor and user mode.

**MCG memory map**

| Absolute address (hex) | Register name   | Width (in bits) | Access | Reset value | Section/page                |
|------------------------|---|-----------------|--------|-------------|-----------------------------|
| 4006_4000              | MCG Control 1 Register (MCG_C1)                       | 8               | R/W    | 04h         | <a href="#">24.3.1/372</a>  |
| 4006_4001              | MCG Control 2 Register (MCG_C2)                       | 8               | R/W    | 80h         | <a href="#">24.3.2/373</a>  |
| 4006_4002              | MCG Control 3 Register (MCG_C3)                       | 8               | R/W    | Undefined   | <a href="#">24.3.3/374</a>  |
| 4006_4003              | MCG Control 4 Register (MCG_C4)                       | 8               | R/W    | Undefined   | <a href="#">24.3.4/374</a>  |
| 4006_4004              | MCG Control 5 Register (MCG_C5)                       | 8               | R/W    | 00h         | <a href="#">24.3.5/376</a>  |
| 4006_4005              | MCG Control 6 Register (MCG_C6)                       | 8               | R/W    | 00h         | <a href="#">24.3.6/377</a>  |
| 4006_4006              | MCG Status Register (MCG_S)                           | 8               | R      | 10h         | <a href="#">24.3.7/378</a>  |
| 4006_4008              | MCG Status and Control Register (MCG_SC)              | 8               | R/W    | 02h         | <a href="#">24.3.8/380</a>  |
| 4006_400A              | MCG Auto Trim Compare Value High Register (MCG_ATCVH) | 8               | R/W    | 00h         | <a href="#">24.3.9/381</a>  |
| 4006_400B              | MCG Auto Trim Compare Value Low Register (MCG_ATCVL)  | 8               | R/W    | 00h         | <a href="#">24.3.10/381</a> |
| 4006_400C              | MCG Control 7 Register (MCG_C7)                       | 8               | R/W    | 00h         | <a href="#">24.3.11/382</a> |
| 4006_400D              | MCG Control 8 Register (MCG_C8)                       | 8               | R/W    | 80h         | <a href="#">24.3.12/382</a> |
| 4006_400E              | MCG Control 9 Register (MCG_C9)                       | 8               | R/W    | 00h         | <a href="#">24.3.13/383</a> |
| 4006_400F              | MCG Control 10 Register (MCG_C10)                     | 8               | R/W    | 00h         | <a href="#">24.3.14/383</a> |

## 24.3.1 MCG Control 1 Register (MCG\_C1)

Address: 4006\_4000h base + 0h offset = 4006\_4000h

| Bit        | 7    | 6 | 5     | 4 | 3     | 2       | 1        | 0 |
|------------|------|---|-------|---|-------|---------|----------|---|
| Read Write | CLKS |   | FRDIV |   | IREFS | IRCLKEN | IREFSTEN |   |
| Reset      | 0    | 0 | 0     | 0 | 0     | 1       | 0        | 0 |

### MCG\_C1 field descriptions

| Field         | Description  |
|---------------|--|
| 7–6<br>CLKS   | Clock Source Select<br><br>Selects the clock source for MCGOUTCLK .<br><br>00 Encoding 0 — Output of FLL or PLL is selected (depends on PLLS control bit).<br>01 Encoding 1 — Internal reference clock is selected.<br>10 Encoding 2 — External reference clock is selected.<br>11 Encoding 3 — Reserved.  |
| 5–3<br>FRDIV  | FLL External Reference Divider<br><br>Selects the amount to divide down the external reference clock for the FLL. The resulting frequency must be in the range 31.25 kHz to 39.0625 kHz (This is required when FLL/DCO is the clock source for MCGOUTCLK . In FBE mode, it is not required to meet this range, but it is recommended in the cases when trying to enter a FLL mode from FBE).<br><br>000 If RANGE 0 = 0 , Divide Factor is 1; for all other RANGE 0 values, Divide Factor is 32.<br>001 If RANGE 0 = 0 , Divide Factor is 2; for all other RANGE 0 values, Divide Factor is 64.<br>010 If RANGE 0 = 0 , Divide Factor is 4; for all other RANGE 0 values, Divide Factor is 128.<br>011 If RANGE 0 = 0 , Divide Factor is 8; for all other RANGE 0 values, Divide Factor is 256.<br>100 If RANGE 0 = 0 , Divide Factor is 16; for all other RANGE 0 values, Divide Factor is 512.<br>101 If RANGE 0 = 0 , Divide Factor is 32; for all other RANGE 0 values, Divide Factor is 1024.<br>110 If RANGE 0 = 0 , Divide Factor is 64; for all other RANGE 0 values, Divide Factor is 1280 .<br>111 If RANGE 0 = 0 , Divide Factor is 128; for all other RANGE 0 values, Divide Factor is 1536 . |
| 2<br>IREFS    | Internal Reference Select<br><br>Selects the reference clock source for the FLL.<br><br>0 External reference clock is selected.<br>1 The slow internal reference clock is selected.  |
| 1<br>IRCLKEN  | Internal Reference Clock Enable<br><br>Enables the internal reference clock for use as MCGIRCLK.<br><br>0 MCGIRCLK inactive.<br>1 MCGIRCLK active.   |
| 0<br>IREFSTEN | Internal Reference Stop Enable<br><br>Controls whether or not the internal reference clock remains enabled when the MCG enters Stop mode.<br><br>0 Internal reference clock is disabled in Stop mode.<br>1 Internal reference clock is enabled in Stop mode if IRCLKEN is set or if MCG is in FEI, FBI, or BLPI modes before entering Stop mode.   |

## 24.3.2 MCG Control 2 Register (MCG\_C2)

Address: 4006\_4000h base + 1h offset = 4006\_4001h

| Bit   | 7      | 6 | 5      | 4 | 3    | 2     | 1  | 0    |
|-------|--------|---|--------|---|------|-------|----|------|
| Read  | LOCRE0 | 0 | RANGE0 |   | HGO0 | EREF0 | LP | IRCS |
| Write |        |   |        |   |      |       |    |      |
| Reset | 1      | 0 | 0      | 0 | 0    | 0     | 0  | 0    |

**MCG\_C2 field descriptions**

| Field         | Description   |
|---------------|---|
| 7<br>LOCRE0   | <p>Loss of Clock Reset Enable</p> <p>Determines whether an interrupt or a reset request is made following a loss of OSC0 external reference clock. The LOCRE0 only has an affect when CME0 is set.</p> <ul style="list-style-type: none"> <li>0 Interrupt request is generated on a loss of OSC0 external reference clock.</li> <li>1 Generate a reset request on a loss of OSC0 external reference clock.</li> </ul>   |
| 6<br>Reserved | <p>This field is reserved.</p> <p>This read-only field is reserved and always has the value 0.</p>  |
| 5–4<br>RANGE0 | <p>Frequency Range Select</p> <p>Selects the frequency range for the crystal oscillator or external clock source. See the Oscillator (OSC) chapter for more details and the device data sheet for the frequency ranges used.</p> <ul style="list-style-type: none"> <li>00 Encoding 0 — Low frequency range selected for the crystal oscillator .</li> <li>01 Encoding 1 — High frequency range selected for the crystal oscillator .</li> <li>1X Encoding 2 — Very high frequency range selected for the crystal oscillator .</li> </ul> |
| 3<br>HGO0     | <p>High Gain Oscillator Select</p> <p>Controls the crystal oscillator mode of operation. See the Oscillator (OSC) chapter for more details.</p> <ul style="list-style-type: none"> <li>0 Configure crystal oscillator for low-power operation.</li> <li>1 Configure crystal oscillator for high-gain operation.</li> </ul>  |
| 2<br>EREF0    | <p>External Reference Select</p> <p>Selects the source for the external reference clock. See the Oscillator (OSC) chapter for more details.</p> <ul style="list-style-type: none"> <li>0 External reference clock requested.</li> <li>1 Oscillator requested.</li> </ul>  |
| 1<br>LP       | <p>Low Power Select</p> <p>Controls whether the FLL or PLL is disabled in BLPI and BLPE modes. In FBE or PBE modes, setting this bit to 1 will transition the MCG into BLPE mode; in FBI mode, setting this bit to 1 will transition the MCG into BLPI mode. In any other MCG mode, LP bit has no affect.</p> <ul style="list-style-type: none"> <li>0 FLL or PLL is not disabled in bypass modes.</li> <li>1 FLL or PLL is disabled in bypass modes (lower power)</li> </ul>   |
| 0<br>IRCS     | <p>Internal Reference Clock Select</p> <p>Selects between the fast or slow internal reference clock source.</p>   |

*Table continues on the next page...*

**MCG\_C2 field descriptions (continued)**

| Field | Description                               |
|-------|---|
|       | 0 Slow internal reference clock selected. |
|       | 1 Fast internal reference clock selected. |

**24.3.3 MCG Control 3 Register (MCG\_C3)**

Address: 4006\_4000h base + 2h offset = 4006\_4002h

| Bit        | 7      | 6  | 5  | 4  | 3  | 2  | 1  | 0  |
|------------|--------|----|----|----|----|----|----|----|
| Read Write | SCTRIM |    |    |    |    |    |    |    |
| Reset      | x*     | x* | x* | x* | x* | x* | x* | x* |

\* Notes:

- x = Undefined at reset.

**MCG\_C3 field descriptions**

| Field         | Description  |
|---------------|--|
| 7–0<br>SCTRIM | <p>Slow Internal Reference Clock Trim Setting</p> <p>SCTRIM <sup>1</sup> controls the slow internal reference clock frequency by controlling the slow internal reference clock period. The SCTRIM bits are binary weighted, that is, bit 1 adjusts twice as much as bit 0. Increasing the binary value increases the period, and decreasing the value decreases the period.</p> <p>An additional fine trim bit is available in C4 register as the SCFTRIM bit. Upon reset, this value is loaded with a factory trim value.</p> <p>If an SCTRIM value stored in nonvolatile memory is to be used, it is your responsibility to copy that value from the nonvolatile memory location to this register.</p> |

1. A value for SCTRIM is loaded during reset from a factory programmed location .

**24.3.4 MCG Control 4 Register (MCG\_C4)****NOTE**

Reset values for DRST and DMX32 bits are 0.

Address: 4006\_4000h base + 3h offset = 4006\_4003h

| Bit        | 7      | 6 | 5 | 4  | 3  | 2  | 1  | 0  |
|------------|--------|---|---|----|----|----|----|----|
| Read Write | FCTRIM |   |   |    |    |    |    |    |
| Reset      | 0      | 0 | 0 | x* | x* | x* | x* | x* |

\* Notes:

- x = Undefined at reset.
- A value for FCTRIM is loaded during reset from a factory programmed location . x = Undefined at reset.

**MCG\_C4 field descriptions**

| Field           | Description  |                   |            |            |  |          |       |                 |            |           |    |   |                   |     |           |   |            |     |        |    |   |                   |      |           |   |            |      |        |    |   |                   |      |           |   |            |      |        |    |   |                   |      |            |   |            |      |        |
|-----------------|--|-------------------|------------|------------|--|----------|-------|-----------------|------------|-----------|----|---|-------------------|-----|-----------|---|------------|-----|--------|----|---|-------------------|------|-----------|---|------------|------|--------|----|---|-------------------|------|-----------|---|------------|------|--------|----|---|-------------------|------|------------|---|------------|------|--------|
| 7<br>DMX32      | <p>DCO Maximum Frequency with 32.768 kHz Reference</p> <p>The DMX32 bit controls whether the DCO frequency range is narrowed to its maximum frequency with a 32.768 kHz reference.</p> <p>The following table identifies settings for the DCO frequency range.</p> <p><b>NOTE:</b> The system clocks derived from this source should not exceed their specified maximums.</p>  |                   |            |            |  |          |       |                 |            |           |    |   |                   |     |           |   |            |     |        |    |   |                   |      |           |   |            |      |        |    |   |                   |      |           |   |            |      |        |    |   |                   |      |            |   |            |      |        |
|                 | <table border="1"> <thead> <tr> <th>DRST_DRS</th><th>DMX32</th><th>Reference Range</th><th>FLL Factor</th><th>DCO Range</th></tr> </thead> <tbody> <tr> <td rowspan="2">00</td><td>0</td><td>31.25–39.0625 kHz</td><td>640</td><td>20–25 MHz</td></tr> <tr> <td>1</td><td>32.768 kHz</td><td>732</td><td>24 MHz</td></tr> <tr> <td rowspan="2">01</td><td>0</td><td>31.25–39.0625 kHz</td><td>1280</td><td>40–50 MHz</td></tr> <tr> <td>1</td><td>32.768 kHz</td><td>1464</td><td>48 MHz</td></tr> <tr> <td rowspan="2">10</td><td>0</td><td>31.25–39.0625 kHz</td><td>1920</td><td>60–75 MHz</td></tr> <tr> <td>1</td><td>32.768 kHz</td><td>2197</td><td>72 MHz</td></tr> <tr> <td rowspan="2">11</td><td>0</td><td>31.25–39.0625 kHz</td><td>2560</td><td>80–100 MHz</td></tr> <tr> <td>1</td><td>32.768 kHz</td><td>2929</td><td>96 MHz</td></tr> </tbody> </table> <p>0 DCO has a default range of 25%.</p> <p>1 DCO is fine-tuned for maximum frequency with 32.768 kHz reference.</p> |                   |            |            |  | DRST_DRS | DMX32 | Reference Range | FLL Factor | DCO Range | 00 | 0 | 31.25–39.0625 kHz | 640 | 20–25 MHz | 1 | 32.768 kHz | 732 | 24 MHz | 01 | 0 | 31.25–39.0625 kHz | 1280 | 40–50 MHz | 1 | 32.768 kHz | 1464 | 48 MHz | 10 | 0 | 31.25–39.0625 kHz | 1920 | 60–75 MHz | 1 | 32.768 kHz | 2197 | 72 MHz | 11 | 0 | 31.25–39.0625 kHz | 2560 | 80–100 MHz | 1 | 32.768 kHz | 2929 | 96 MHz |
| DRST_DRS        | DMX32  | Reference Range   | FLL Factor | DCO Range  |  |          |       |                 |            |           |    |   |                   |     |           |   |            |     |        |    |   |                   |      |           |   |            |      |        |    |   |                   |      |           |   |            |      |        |    |   |                   |      |            |   |            |      |        |
| 00              | 0  | 31.25–39.0625 kHz | 640        | 20–25 MHz  |  |          |       |                 |            |           |    |   |                   |     |           |   |            |     |        |    |   |                   |      |           |   |            |      |        |    |   |                   |      |           |   |            |      |        |    |   |                   |      |            |   |            |      |        |
|                 | 1  | 32.768 kHz        | 732        | 24 MHz     |  |          |       |                 |            |           |    |   |                   |     |           |   |            |     |        |    |   |                   |      |           |   |            |      |        |    |   |                   |      |           |   |            |      |        |    |   |                   |      |            |   |            |      |        |
| 01              | 0  | 31.25–39.0625 kHz | 1280       | 40–50 MHz  |  |          |       |                 |            |           |    |   |                   |     |           |   |            |     |        |    |   |                   |      |           |   |            |      |        |    |   |                   |      |           |   |            |      |        |    |   |                   |      |            |   |            |      |        |
|                 | 1  | 32.768 kHz        | 1464       | 48 MHz     |  |          |       |                 |            |           |    |   |                   |     |           |   |            |     |        |    |   |                   |      |           |   |            |      |        |    |   |                   |      |           |   |            |      |        |    |   |                   |      |            |   |            |      |        |
| 10              | 0  | 31.25–39.0625 kHz | 1920       | 60–75 MHz  |  |          |       |                 |            |           |    |   |                   |     |           |   |            |     |        |    |   |                   |      |           |   |            |      |        |    |   |                   |      |           |   |            |      |        |    |   |                   |      |            |   |            |      |        |
|                 | 1  | 32.768 kHz        | 2197       | 72 MHz     |  |          |       |                 |            |           |    |   |                   |     |           |   |            |     |        |    |   |                   |      |           |   |            |      |        |    |   |                   |      |           |   |            |      |        |    |   |                   |      |            |   |            |      |        |
| 11              | 0  | 31.25–39.0625 kHz | 2560       | 80–100 MHz |  |          |       |                 |            |           |    |   |                   |     |           |   |            |     |        |    |   |                   |      |           |   |            |      |        |    |   |                   |      |           |   |            |      |        |    |   |                   |      |            |   |            |      |        |
|                 | 1  | 32.768 kHz        | 2929       | 96 MHz     |  |          |       |                 |            |           |    |   |                   |     |           |   |            |     |        |    |   |                   |      |           |   |            |      |        |    |   |                   |      |           |   |            |      |        |    |   |                   |      |            |   |            |      |        |
| 6–5<br>DRST_DRS | <p>DCO Range Select</p> <p>The DRS bits select the frequency range for the FLL output, DCOOUT. When the LP bit is set, writes to the DRS bits are ignored. The DRST read field indicates the current frequency range for DCOOUT. The DRST field does not update immediately after a write to the DRS field due to internal synchronization between clock domains. See the DCO Frequency Range table for more details.</p> <p>00 Encoding 0 — Low range (reset default).<br/>     01 Encoding 1 — Mid range.<br/>     10 Encoding 2 — Mid-high range.<br/>     11 Encoding 3 — High range.</p>  |                   |            |            |  |          |       |                 |            |           |    |   |                   |     |           |   |            |     |        |    |   |                   |      |           |   |            |      |        |    |   |                   |      |           |   |            |      |        |    |   |                   |      |            |   |            |      |        |
| 4–1<br>FCTRIM   | <p>Fast Internal Reference Clock Trim Setting</p> <p>FCTRIM <sup>1</sup> controls the fast internal reference clock frequency by controlling the fast internal reference clock period. The FCTRIM bits are binary weighted, that is, bit 1 adjusts twice as much as bit 0. Increasing the binary value increases the period, and decreasing the value decreases the period.</p> <p>If an FCTRIM[3:0] value stored in nonvolatile memory is to be used, it is your responsibility to copy that value from the nonvolatile memory location to this register.</p>   |                   |            |            |  |          |       |                 |            |           |    |   |                   |     |           |   |            |     |        |    |   |                   |      |           |   |            |      |        |    |   |                   |      |           |   |            |      |        |    |   |                   |      |            |   |            |      |        |
| 0<br>SCFTRIM    | <p>Slow Internal Reference Clock Fine Trim</p> <p>SCFTRIM <sup>2</sup> controls the smallest adjustment of the slow internal reference clock frequency. Setting SCFTRIM increases the period and clearing SCFTRIM decreases the period by the smallest amount possible.</p> <p>If an SCFTRIM value stored in nonvolatile memory is to be used, it is your responsibility to copy that value from the nonvolatile memory location to this bit.</p>  |                   |            |            |  |          |       |                 |            |           |    |   |                   |     |           |   |            |     |        |    |   |                   |      |           |   |            |      |        |    |   |                   |      |           |   |            |      |        |    |   |                   |      |            |   |            |      |        |

1. A value for FCTRIM is loaded during reset from a factory programmed location .

2. A value for SCFTRIM is loaded during reset from a factory programmed location .

## 24.3.5 MCG Control 5 Register (MCG\_C5)

Address: 4006\_4000h base + 4h offset = 4006\_4004h

| Bit   | 7 | 6         | 5        | 4 | 3      | 2 | 1 | 0 |
|-------|---|-----------|----------|---|--------|---|---|---|
| Read  | 0 | PLLCLKEN0 | PLLSTEN0 |   | PRDIV0 |   |   |   |
| Write |   |           |          |   |        |   |   |   |
| Reset | 0 | 0         | 0        | 0 | 0      | 0 | 0 | 0 |

### MCG\_C5 field descriptions

| Field          | Description   |
|----------------|---|
| 7<br>Reserved  | This field is reserved.<br>This read-only field is reserved and always has the value 0.   |
| 6<br>PLLCLKEN0 | PLL Clock Enable<br><br>Enables the PLL independent of PLLS and enables the PLL clock for use as MCGPLLCLK. (PRDIV 0 needs to be programmed to the correct divider to generate a PLL reference clock in the range of 2 - 4 MHz range prior to setting the PLLCLKEN 0 bit). Setting PLLCLKEN 0 will enable the external oscillator if not already enabled. Whenever the PLL is being enabled by means of the PLLCLKEN 0 bit, and the external oscillator is being used as the reference clock, the OSCINIT 0 bit should be checked to make sure it is set.<br><br>0 MCGPLLCLK is inactive.<br>1 MCGPLLCLK is active. |
| 5<br>PLLSTEN0  | PLL Stop Enable<br><br>Enables the PLL Clock during Normal Stop. In Low Power Stop mode, the PLL clock gets disabled even if PLLSTEN 0 =1. All other power modes, PLLSTEN 0 bit has no affect and does not enable the PLL Clock to run if it is written to 1.<br><br>0 MCGPLLCLK is disabled in any of the Stop modes.<br>1 MCGPLLCLK is enabled if system is in Normal Stop mode.  |
| 4–0<br>PRDIV0  | PLL External Reference Divider<br><br>Selects the amount to divide down the external reference clock for the PLL. The resulting frequency must be in the range of 2 MHz to 4 MHz. After the PLL is enabled (by setting either PLLCLKEN 0 or PLLS), the PRDIV 0 value must not be changed when LOCK0 is zero.  |

**Table 24-7. PLL External Reference Divide Factor**

| PRDIV 0 | Divide Factor |
|---------|---------------|---------|---------------|---------|---------------|---------|---------------|
| 00000   | 1             | 01000   | 9             | 10000   | 17            | 11000   | 25            |
| 00001   | 2             | 01001   | 10            | 10001   | 18            | 11001   | Reserved      |
| 00010   | 3             | 01010   | 11            | 10010   | 19            | 11010   | Reserved      |
| 00011   | 4             | 01011   | 12            | 10011   | 20            | 11011   | Reserved      |

Table continues on the next page...

**MCG\_C5 field descriptions (continued)**

| Field   | Description |  |  |       |    |  |       |    |  |       |          |
|---|-------------|--|--|-------|----|--|-------|----|--|-------|----------|
| <b>Table 24-7. PLL External Reference Divide Factor (continued)</b> |             |  |  |       |    |  |       |    |  |       |          |
| 00100   | 5           |  |  | 01100 | 13 |  | 10100 | 21 |  | 11100 | Reserved |
| 00101   | 6           |  |  | 01101 | 14 |  | 10101 | 22 |  | 11101 | Reserved |
| 00110   | 7           |  |  | 01110 | 15 |  | 10110 | 23 |  | 11110 | Reserved |
| 00111   | 8           |  |  | 01111 | 16 |  | 10111 | 24 |  | 11111 | Reserved |

**24.3.6 MCG Control 6 Register (MCG\_C6)**

Address: 4006\_4000h base + 5h offset = 4006\_4005h

| Bit        | 7      | 6    | 5    | 4     | 3 | 2 | 1 | 0 |
|------------|--------|------|------|-------|---|---|---|---|
| Read Write | LOLIE0 | PLLS | CME0 | VDIV0 |   |   |   |   |
| Reset      | 0      | 0    | 0    | 0     | 0 | 0 | 0 | 0 |

**MCG\_C6 field descriptions**

| Field       | Description  |
|-------------|--|
| 7<br>LOLIE0 | <p>Loss of Lock Interrupt Enable</p> <p>Determines if an interrupt request is made following a loss of lock indication. This bit only has an effect when LOLS 0 is set.</p> <p>0 No interrupt request is generated on loss of lock.<br/>1 Generate an interrupt request on loss of lock.</p>   |
| 6<br>PLLS   | <p>PLL Select</p> <p>Controls whether the PLL or FLL output is selected as the MCG source when CLKS[1:0]=00. If the PLLS bit is cleared and PLLCLKEN 0 is not set, the PLL is disabled in all modes. If the PLLS is set, the FLL is disabled in all modes.</p> <p>0 FLL is selected.<br/>1 PLL is selected (PRDIV 0 need to be programmed to the correct divider to generate a PLL reference clock in the range of 2–4 MHz prior to setting the PLLS bit).</p>   |
| 5<br>CME0   | <p>Clock Monitor Enable</p> <p>Enables the loss of clock monitoring circuit for the OSC0 external reference mux select. The LOCRE0 bit will determine if a interrupt or a reset request is generated following a loss of OSC0 indication. The CME0 bit should only be set to a logic 1 when the MCG is in an operational mode that uses the external clock (FEE, FBE, PEE, PBE, or BLPE) . Whenever the CME0 bit is set to a logic 1, the value of the RANGE0 bits in the C2 register should not be changed. CME0 bit should be set to a logic 0 before the MCG enters</p> |

*Table continues on the next page...*

**MCG\_C6 field descriptions (continued)**

| Field        | Description   |  |  |  |  |  |  |  |  |  |
|--------------|---|--|--|--|--|--|--|--|--|--|
|              | any Stop mode. Otherwise, a reset request may occur while in Stop mode. CME0 should also be set to a logic 0 before entering VLPR or VLPW power modes if the MCG is in BLPE mode.   |  |  |  |  |  |  |  |  |  |
|              | 0 External clock monitor is disabled for OSC0.<br>1 External clock monitor is enabled for OSC0.   |  |  |  |  |  |  |  |  |  |
| 4–0<br>VDIV0 | VCO 0 Divider<br><br>Selects the amount to divide the VCO output of the PLL. The VDIV 0 bits establish the multiplication factor (M) applied to the reference clock frequency. After the PLL is enabled (by setting either PLLCLKEN 0 or PLLS), the VDIV 0 value must not be changed when LOCK 0 is zero. |  |  |  |  |  |  |  |  |  |

**Table 24-9. PLL VCO Divide Factor**

| VDIV 0 | Multiply Factor |
|--------|-----------------|--------|-----------------|--------|-----------------|--------|-----------------|
| 00000  | 24              | 01000  | 32              | 10000  | 40              | 11000  | 48              |
| 00001  | 25              | 01001  | 33              | 10001  | 41              | 11001  | 49              |
| 00010  | 26              | 01010  | 34              | 10010  | 42              | 11010  | 50              |
| 00011  | 27              | 01011  | 35              | 10011  | 43              | 11011  | 51              |
| 00100  | 28              | 01100  | 36              | 10100  | 44              | 11100  | 52              |
| 00101  | 29              | 01101  | 37              | 10101  | 45              | 11101  | 53              |
| 00110  | 30              | 01110  | 38              | 10110  | 46              | 11110  | 54              |
| 00111  | 31              | 01111  | 39              | 10111  | 47              | 11111  | 55              |

**24.3.7 MCG Status Register (MCG\_S)**

Address: 4006\_4000h base + 6h offset = 4006\_4006h

| Bit   | 7    | 6     | 5     | 4      | 3     | 2        | 1     | 0 |
|-------|------|-------|-------|--------|-------|----------|-------|---|
| Read  | LOLS | LOCK0 | PLLST | IREFST | CLKST | OSCINIT0 | IRCST |   |
| Write |      |       |       |        |       |          |       |   |
| Reset | 0    | 0     | 0     | 1      | 0     | 0        | 0     | 0 |

**MCG\_S field descriptions**

| Field     | Description  |
|-----------|--|
| 7<br>LOLS | Loss of Lock Status<br><br>This bit is a sticky bit indicating the lock status for the PLL. LOLS is set if after acquiring lock, the PLL output frequency has fallen outside the lock exit frequency tolerance, $D_{unl}$ . LOLIE determines whether an interrupt request is made when LOLS is set. LOLRE determines whether a reset request is made when LOLS is set. This bit is cleared by reset or by writing a logic 1 to it when set. Writing a logic 0 to this bit has no effect. |

Table continues on the next page...

**MCG\_S field descriptions (continued)**

| Field      | Description  |
|------------|--|
|            | <p>0 PLL has not lost lock since LOLS 0 was last cleared.<br/>1 PLL has lost lock since LOLS 0 was last cleared.</p>   |
| 6 LOCK0    | <p>Lock Status</p> <p>This bit indicates whether the PLL has acquired lock. Lock detection is only enabled when the PLL is enabled (either through clock mode selection or PLLCLKEN0=1 setting). While the PLL clock is locking to the desired frequency, the MCG PLL clock (MCGPLLCLK) will be gated off until the LOCK0 bit gets asserted. If the lock status bit is set, changing the value of the PRDIV0 [4:0] bits in the C5 register or the VDIV0[4:0] bits in the C6 register causes the lock status bit to clear and stay cleared until the PLL has reacquired lock. Loss of PLL reference clock will also cause the LOCK0 bit to clear until the PLL has reacquired lock. Entry into LLS, VLPS, or regular Stop with PLLSTEN=0 also causes the lock status bit to clear and stay cleared until the Stop mode is exited and the PLL has reacquired lock. Any time the PLL is enabled and the LOCK0 bit is cleared, the MCGPLLCLK will be gated off until the LOCK0 bit is asserted again.</p> <p>0 PLL is currently unlocked.<br/>1 PLL is currently locked.</p> |
| 5 PLLST    | <p>PLL Select Status</p> <p>This bit indicates the clock source selected by PLLS . The PLLST bit does not update immediately after a write to the PLLS bit due to internal synchronization between clock domains.</p> <p>0 Source of PLLS clock is FLL clock.<br/>1 Source of PLLS clock is PLL output clock.</p>  |
| 4 IREFST   | <p>Internal Reference Status</p> <p>This bit indicates the current source for the FLL reference clock. The IREFST bit does not update immediately after a write to the IREFS bit due to internal synchronization between clock domains.</p> <p>0 Source of FLL reference clock is the external reference clock.<br/>1 Source of FLL reference clock is the internal reference clock.</p>   |
| 3–2 CLKST  | <p>Clock Mode Status</p> <p>These bits indicate the current clock mode. The CLKST bits do not update immediately after a write to the CLKS bits due to internal synchronization between clock domains.</p> <p>00 Encoding 0 — Output of the FLL is selected (reset default).<br/>01 Encoding 1 — Internal reference clock is selected.<br/>10 Encoding 2 — External reference clock is selected.<br/>11 Encoding 3 — Output of the PLL is selected.</p>  |
| 1 OSCINIT0 | <p>OSC Initialization</p> <p>This bit, which resets to 0, is set to 1 after the initialization cycles of the crystal oscillator clock have completed. After being set, the bit is cleared to 0 if the OSC is subsequently disabled. See the OSC module's detailed description for more information.</p>  |
| 0 IRCST    | <p>Internal Reference Clock Status</p> <p>The IRCST bit indicates the current source for the internal reference clock select clock (IRCSCLK). The IRCST bit does not update immediately after a write to the IRCS bit due to internal synchronization between clock domains. The IRCST bit will only be updated if the internal reference clock is enabled, either by the MCG being in a mode that uses the IRC or by setting the C1[IRCLKEN] bit .</p>  |

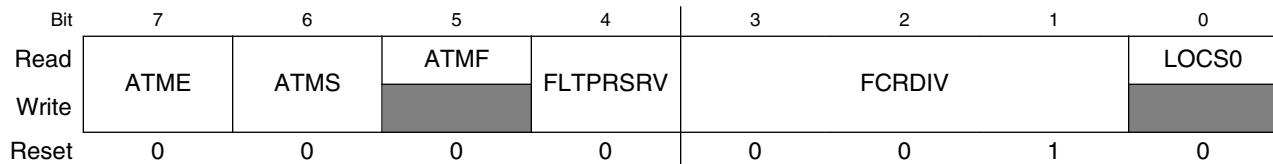
*Table continues on the next page...*

**MCG\_S field descriptions (continued)**

| Field | Description  |
|-------|--|
| 0     | Source of internal reference clock is the slow clock (32 kHz IRC). |
| 1     | Source of internal reference clock is the fast clock (4 MHz IRC).  |

**24.3.8 MCG Status and Control Register (MCG\_SC)**

Address: 4006\_4000h base + 8h offset = 4006\_4008h

**MCG\_SC field descriptions**

| Field         | Description  |
|---------------|--|
| 7<br>ATME     | Automatic Trim Machine Enable<br><br>Enables the Auto Trim Machine to start automatically trimming the selected Internal Reference Clock.<br><br><b>NOTE:</b> ATME deasserts after the Auto Trim Machine has completed trimming all trim bits of the IRCS clock selected by the ATMS bit.<br>Writing to C1, C3, C4, and SC registers or entering Stop mode aborts the auto trim operation and clears this bit.<br><br>0 Auto Trim Machine disabled.<br>1 Auto Trim Machine enabled.  |
| 6<br>ATMS     | Automatic Trim Machine Select<br><br>Selects the IRCS clock for Auto Trim Test.<br><br>0 32 kHz Internal Reference Clock selected.<br>1 4 MHz Internal Reference Clock selected.   |
| 5<br>ATMF     | Automatic Trim Machine Fail Flag<br><br>Fail flag for the Automatic Trim Machine (ATM). This bit asserts when the Automatic Trim Machine is enabled, ATME=1, and a write to the C1, C3, C4, and SC registers is detected or the MCG enters into any Stop mode. A write to ATMF clears the flag.<br><br>0 Automatic Trim Machine completed normally.<br>1 Automatic Trim Machine failed.  |
| 4<br>FLTPRSRV | FLL Filter Preserve Enable<br><br>This bit will prevent the FLL filter values from resetting allowing the FLL output frequency to remain the same during clock mode changes where the FLL/DCO output is still valid. (Note: This requires that the FLL reference frequency to remain the same as what it was prior to the new clock mode switch. Otherwise FLL filter and frequency values will change.)<br><br>0 FLL filter and FLL frequency will reset on changes to correct clock mode.<br>1 FLL filter and FLL frequency retain their previous values during new clock mode change. |

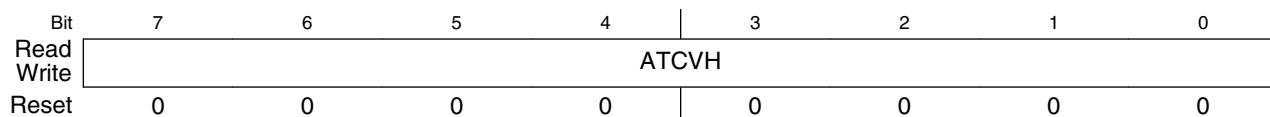
*Table continues on the next page...*

**MCG\_SC field descriptions (continued)**

| Field         | Description  |     |                                |     |                            |     |                     |     |                     |     |                     |     |                     |     |                     |     |                       |
|---------------|--|-----|--------------------------------|-----|----------------------------|-----|---------------------|-----|---------------------|-----|---------------------|-----|---------------------|-----|---------------------|-----|-----------------------|
| 3–1<br>FCRDIV | <p>Fast Clock Internal Reference Divider</p> <p>Selects the amount to divide down the fast internal reference clock. The resulting frequency will be in the range 31.25 kHz to 4 MHz (Note: Changing the divider when the Fast IRC is enabled is not supported).</p> <table> <tr><td>000</td><td>Divide Factor is 1</td></tr> <tr><td>001</td><td>Divide Factor is 2.</td></tr> <tr><td>010</td><td>Divide Factor is 4.</td></tr> <tr><td>011</td><td>Divide Factor is 8.</td></tr> <tr><td>100</td><td>Divide Factor is 16</td></tr> <tr><td>101</td><td>Divide Factor is 32</td></tr> <tr><td>110</td><td>Divide Factor is 64</td></tr> <tr><td>111</td><td>Divide Factor is 128.</td></tr> </table> | 000 | Divide Factor is 1             | 001 | Divide Factor is 2.        | 010 | Divide Factor is 4. | 011 | Divide Factor is 8. | 100 | Divide Factor is 16 | 101 | Divide Factor is 32 | 110 | Divide Factor is 64 | 111 | Divide Factor is 128. |
| 000           | Divide Factor is 1   |     |                                |     |                            |     |                     |     |                     |     |                     |     |                     |     |                     |     |                       |
| 001           | Divide Factor is 2.  |     |                                |     |                            |     |                     |     |                     |     |                     |     |                     |     |                     |     |                       |
| 010           | Divide Factor is 4.  |     |                                |     |                            |     |                     |     |                     |     |                     |     |                     |     |                     |     |                       |
| 011           | Divide Factor is 8.  |     |                                |     |                            |     |                     |     |                     |     |                     |     |                     |     |                     |     |                       |
| 100           | Divide Factor is 16  |     |                                |     |                            |     |                     |     |                     |     |                     |     |                     |     |                     |     |                       |
| 101           | Divide Factor is 32  |     |                                |     |                            |     |                     |     |                     |     |                     |     |                     |     |                     |     |                       |
| 110           | Divide Factor is 64  |     |                                |     |                            |     |                     |     |                     |     |                     |     |                     |     |                     |     |                       |
| 111           | Divide Factor is 128.  |     |                                |     |                            |     |                     |     |                     |     |                     |     |                     |     |                     |     |                       |
| 0<br>LOCS0    | <p>OSC0 Loss of Clock Status</p> <p>The LOCS0 indicates when a loss of OSC0 reference clock has occurred. The LOCS0 bit only has an effect when CME0 is set. This bit is cleared by writing a logic 1 to it when set.</p> <table> <tr><td>0</td><td>Loss of OSC0 has not occurred.</td></tr> <tr><td>1</td><td>Loss of OSC0 has occurred.</td></tr> </table>   | 0   | Loss of OSC0 has not occurred. | 1   | Loss of OSC0 has occurred. |     |                     |     |                     |     |                     |     |                     |     |                     |     |                       |
| 0             | Loss of OSC0 has not occurred.   |     |                                |     |                            |     |                     |     |                     |     |                     |     |                     |     |                     |     |                       |
| 1             | Loss of OSC0 has occurred.   |     |                                |     |                            |     |                     |     |                     |     |                     |     |                     |     |                     |     |                       |

**24.3.9 MCG Auto Trim Compare Value High Register (MCG\_ATCVH)**

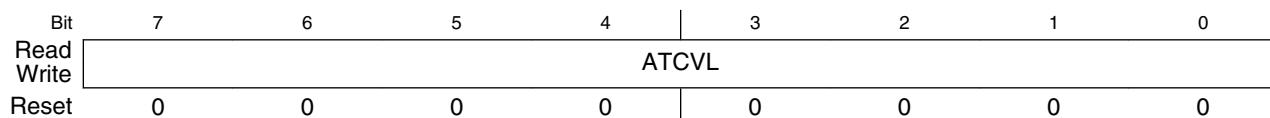
Address: 4006\_4000h base + Ah offset = 4006\_400Ah

**MCG\_ATCVH field descriptions**

| Field        | Description   |
|--------------|---|
| 7–0<br>ATCVH | <p>ATM Compare Value High</p> <p>Values are used by Auto Trim Machine to compare and adjust Internal Reference trim values during ATM SAR conversion.</p> |

**24.3.10 MCG Auto Trim Compare Value Low Register (MCG\_ATCVL)**

Address: 4006\_4000h base + Bh offset = 4006\_400Bh



**MCG\_ATCVL field descriptions**

| Field        | Description   |
|--------------|---|
| 7–0<br>ATCVL | ATM Compare Value Low<br><br>Values are used by Auto Trim Machine to compare and adjust Internal Reference trim values during ATM SAR conversion. |

**24.3.11 MCG Control 7 Register (MCG\_C7)**

Address: 4006\_4000h base + Ch offset = 4006\_400Ch

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

**MCG\_C7 field descriptions**

| Field           | Description   |
|-----------------|---|
| 7–6<br>Reserved | This field is reserved.<br>This read-only field is reserved and always has the value 0.                 |
| 5–1<br>Reserved | Reserved<br><br>This field is reserved.<br>This read-only field is reserved and always has the value 0. |
| 0<br>Reserved   | Reserved<br><br>This field is reserved.<br>This read-only field is reserved and always has the value 0. |

**24.3.12 MCG Control 8 Register (MCG\_C8)**

Address: 4006\_4000h base + Dh offset = 4006\_400Dh

| Bit   | 7 | 6     | 5 | 4 | 3 | 2 | 1 | 0 |
|-------|---|-------|---|---|---|---|---|---|
| Read  | 0 | LOLRE | 0 |   | 0 |   |   | 0 |
| Write | 1 | 0     | 0 | 0 | 0 | 0 | 0 | 0 |
| Reset | 1 | 0     | 0 | 0 | 0 | 0 | 0 | 0 |

**MCG\_C8 field descriptions**

| Field         | Description   |
|---------------|---|
| 7<br>Reserved | This field is reserved.<br>This read-only field is reserved and always has the value 0.                                 |
| 6<br>LOLRE    | PLL Loss of Lock Reset Enable<br><br>Determines if a interrupt or a reset request is made following a PLL loss of lock. |

*Table continues on the next page...*

**MCG\_C8 field descriptions (continued)**

| Field           | Description  |
|-----------------|--|
| 0               | Interrupt request is generated on a PLL loss of lock indication. The PLL loss of lock interrupt enable bit must also be set to generate the interrupt request. |
| 1               | Generate a reset request on a PLL loss of lock indication.   |
| 5<br>Reserved   | This field is reserved.<br>This read-only field is reserved and always has the value 0.  |
| 4–1<br>Reserved | This field is reserved.<br>This read-only field is reserved and always has the value 0.  |
| 0<br>Reserved   | This field is reserved.<br>This read-only field is reserved and always has the value 0.  |

**24.3.13 MCG Control 9 Register (MCG\_C9)**

Address: 4006\_4000h base + Eh offset = 4006\_400Eh

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

**MCG\_C9 field descriptions**

| Field           | Description   |
|-----------------|---|
| 7–4<br>Reserved | This field is reserved.<br>This read-only field is reserved and always has the value 0. |
| 3–0<br>Reserved | This field is reserved.<br>This read-only field is reserved and always has the value 0. |

**24.3.14 MCG Control 10 Register (MCG\_C10)**

Address: 4006\_4000h base + Fh offset = 4006\_400Fh

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

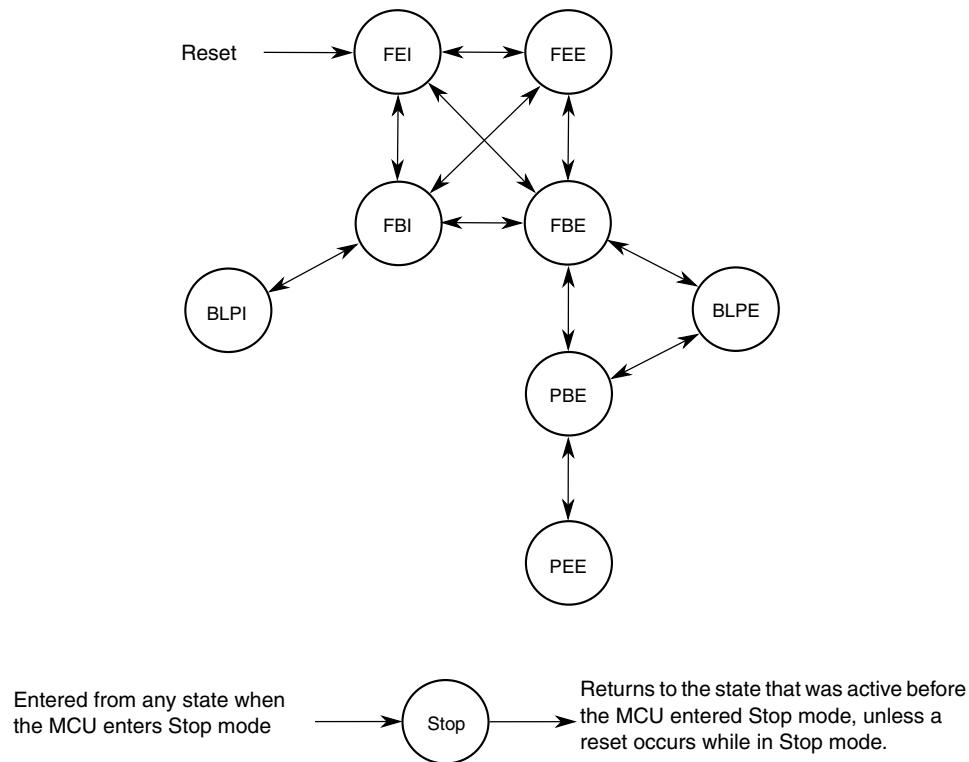
**MCG\_C10 field descriptions**

| Field           | Description   |
|-----------------|---|
| 7–4<br>Reserved | This field is reserved.<br>This read-only field is reserved and always has the value 0. |
| 3–0<br>Reserved | This field is reserved.<br>This read-only field is reserved and always has the value 0. |

## 24.4 Functional Description

### 24.4.1 MCG mode state diagram

The nine states of the MCG are shown in the following figure and are described in [Table 24-18](#). The arrows indicate the permitted MCG mode transitions.



**Figure 24-16. MCG mode state diagram**

#### NOTE

- During exits from LLS or VLPS when the MCG is in PEE mode, the MCG will reset to PBE clock mode and the C1[CLKS] and S[CLKST] will automatically be set to 2'b10.
- If entering Normal Stop mode when the MCG is in PEE mode with C5[PLLSTEN]=0, the MCG will reset to PBE clock mode and C1[CLKS] and S[CLKST] will automatically be set to 2'b10.

### 24.4.1.1 MCG modes of operation

The MCG operates in one of the following modes.

#### Note

The MCG restricts transitions between modes. For the permitted transitions, see [Figure 24-16](#).

**Table 24-18. MCG modes of operation**

| Mode                        | Description  |
|-----------------------------|--|
| FLL Engaged Internal (FEI)  | <p>FLL engaged internal (FEI) is the default mode of operation and is entered when all the following conditions occur:</p> <ul style="list-style-type: none"> <li>• C1[CLKS] bits are written to 00</li> <li>• C1[IREFS] bit is written to 1</li> <li>• C6[PLLS] bit is written to 0</li> </ul> <p>In FEI mode, MCGOUTCLK is derived from the FLL clock (DCOCLK) that is controlled by the 32 kHz Internal Reference Clock (IRC). The FLL loop will lock the DCO frequency to the FLL factor, as selected by C4[DRST_DRST] and C4[DMX32] bits, times the internal reference frequency. See the C4[DMX32] bit description for more details. In FEI mode, the PLL is disabled in a low-power state unless C5[PLLCLKEN0] is set.</p>  |
| FLL Engaged External (FEE)  | <p>FLL engaged external (FEE) mode is entered when all the following conditions occur:</p> <ul style="list-style-type: none"> <li>• C1[CLKS] bits are written to 00</li> <li>• C1[IREFS] bit is written to 0</li> <li>• C1[FRDIV] must be written to divide external reference clock to be within the range of 31.25 kHz to 39.0625 kHz</li> <li>• C6[PLLS] bit is written to 0</li> </ul> <p>In FEE mode, MCGOUTCLK is derived from the FLL clock (DCOCLK) that is controlled by the external reference clock. The FLL loop will lock the DCO frequency to the FLL factor, as selected by C4[DRST_DRST] and C4[DMX32] bits, times the external reference frequency, as specified by C1[FRDIV] and C2[RANGE0]. See the C4[DMX32] bit description for more details. In FEE mode, the PLL is disabled in a low-power state unless C5[PLLCLKEN0] is set.</p>  |
| FLL Bypassed Internal (FBI) | <p>FLL bypassed internal (FBI) mode is entered when all the following conditions occur:</p> <ul style="list-style-type: none"> <li>• C1[CLKS] bits are written to 01</li> <li>• C1[IREFS] bit is written to 1</li> <li>• C6[PLLS] is written to 0</li> <li>• C2[LP] is written to 0</li> </ul> <p>In FBI mode, the MCGOUTCLK is derived either from the slow (32 kHz IRC) or fast (4 MHz IRC) internal reference clock, as selected by the C2[IRCS] bit. The FLL is operational but its output is not used. This mode is useful to allow the FLL to acquire its target frequency while the MCGOUTCLK is driven from the C2[IRCS] selected internal reference clock. The FLL clock (DCOCLK) is controlled by the slow internal reference clock, and the DCO clock frequency locks to a multiplication factor, as selected by C4[DRST_DRST] and C4[DMX32] bits, times the internal reference frequency. See the C4[DMX32] bit description for more details. In FBI mode, the PLL is disabled in a low-power state unless C5[PLLCLKEN0] is set.</p> |

*Table continues on the next page...*

**Table 24-18. MCG modes of operation (continued)**

| Mode  | Description  |
|---|--|
| FLL Bypassed External (FBE)                     | <p>FLL bypassed external (FBE) mode is entered when all the following conditions occur:</p> <ul style="list-style-type: none"> <li>• C1[CLKS] bits are written to 10</li> <li>• C1[IREFS] bit is written to 0</li> <li>• C1[FRDIV] must be written to divide external reference clock to be within the range of 31.25 kHz to 39.0625 kHz.</li> <li>• C6[PLLS] bit is written to 0</li> <li>• C2[LP] is written to 0</li> </ul> <p>In FBE mode, the MCGOUTCLK is derived from the external reference clock. The FLL is operational but its output is not used. This mode is useful to allow the FLL to acquire its target frequency while the MCGOUTCLK is driven from the external reference clock. The FLL clock (DCOCLK) is controlled by the external reference clock, and the DCO clock frequency locks to a multiplication factor, as selected by C4[DRST_DRST] and C4[DMX32] bits, times the divided external reference frequency. See the C4[DMX32] bit description for more details. In FBI mode the PLL is disabled in a low-power state unless C5[PLLCLKEN0] is set.</p> |
| PLL Engaged External (PEE)                      | <p>PLL Engaged External (PEE) mode is entered when all the following conditions occur:</p> <ul style="list-style-type: none"> <li>• C1[CLKS] bits are written to 00</li> <li>• C1[IREFS] bit is written to 0</li> <li>• C6[PLLS] bit is written to 1</li> </ul> <p>In PEE mode, the MCGOUTCLK is derived from the PLL clock, which is controlled by the external reference clock. The PLL clock frequency locks to a multiplication factor, as specified by C6[VDIV0], times the external reference frequency, as specified by C5[PRDIV0]. The PLL's programmable reference divider must be configured to produce a valid PLL reference clock. The FLL is disabled in a low-power state.</p>   |
| PLL Bypassed External (PBE)                     | <p>PLL Bypassed External (PBE) mode is entered when all the following conditions occur:</p> <ul style="list-style-type: none"> <li>• C1[CLKS] bits are written to 10</li> <li>• C1[IREFS] bit is written to 0</li> <li>• C6[PLLS] bit is written to 1</li> <li>• C2[LP] bit is written to 0</li> </ul> <p>In PBE mode, MCGOUTCLK is derived from the external reference clock; the PLL is operational, but its output clock is not used. This mode is useful to allow the PLL to acquire its target frequency while MCGOUTCLK is driven from the external reference clock. The PLL clock frequency locks to a multiplication factor, as specified by its [VDIV], times the PLL reference frequency, as specified by its [PRDIV]. In preparation for transition to PEE, the PLL's programmable reference divider must be configured to produce a valid PLL reference clock. The FLL is disabled in a low-power state.</p>   |
| Bypassed Low Power Internal (BLPI) <sup>1</sup> | <p>Bypassed Low Power Internal (BLPI) mode is entered when all the following conditions occur:</p> <ul style="list-style-type: none"> <li>• C1[CLKS] bits are written to 01</li> <li>• C1[IREFS] bit is written to 1</li> <li>• C6[PLLS] bit is written to 0</li> <li>• C2[LP] bit is written to 1</li> </ul> <p>In BLPI mode, MCGOUTCLK is derived from the internal reference clock. The FLL is disabled and PLL is disabled even if the C5[PLLCLKEN0] is set to 1.</p>  |

Table continues on the next page...

**Table 24-18. MCG modes of operation (continued)**

| Mode                               | Description  |
|------------------------------------|--|
| Bypassed Low Power External (BLPE) | <p>Bypassed Low Power External (BLPE) mode is entered when all the following conditions occur:</p> <ul style="list-style-type: none"> <li>• C1[CLKS] bits are written to 10</li> <li>• C1[IREFS] bit is written to 0</li> <li>• C2[LP] bit is written to 1</li> </ul> <p>In BLPE mode, MCGOUTCLK is derived from the external reference clock. The FLL is disabled and PLL is disabled even if the C5[PLLCLKEN0] is set to 1.</p>  |
| Stop                               | <p>Entered whenever the MCU enters a Stop state. The power modes are chip specific. For power mode assignments, see the chapter that describes how modules are configured and MCG behavior during Stop recovery. Entering Stop mode, the FLL is disabled, and all MCG clock signals are static except in the following case:</p> <p>MCGPLLCLK is active in Normal Stop mode when PLLSTEN=1</p> <p>MCGIRCLK is active in Normal Stop mode when all the following conditions become true:</p> <ul style="list-style-type: none"> <li>• C1[IRCLKEN] = 1</li> <li>• C1[IREFSTEN] = 1</li> </ul> <p><b>NOTE:</b></p> <ul style="list-style-type: none"> <li>• In VLPS Stop Mode, the MCGIRCLK can be programmed to stay enabled and continue running if C1[IRCLKEN] = 1, C1[IREFSTEN]=1, and Fast IRC clock is selected (C2[IRCS] = 1)</li> </ul> <p><b>NOTE:</b></p> <ul style="list-style-type: none"> <li>• When entering Low Power Stop modes (LLS or VLPS) from PEE mode, on exit the MCG clock mode is forced to PBE clock mode. C1[CLKS] and S[CLKST] will be configured to 2'b10if entering from PEE mode or to 2'b01 if entering from PEI mode, C5[PLLSTEN0] will be force to 1'b0 and S[LOCK0] bit will be cleared without setting S[LOLS0].</li> <li>• When entering Normal Stop mode from PEE mode and if C5[PLLSTEN0]=0, on exit the MCG clock mode is forced to PBE mode, the C1[CLKS] and S[CLKST] will be configured to 2'b10 and S[LOCK0] bit will clear without setting S[LOLS0]. If C5[PLLSTEN0]=1, the S[LOCK0] bit will not get cleared and on exit the MCG will continue to run in PEE mode.</li> </ul> |

1. If entering VLPR mode, MCG has to be configured and enter BLPE mode or BLPI mode with the Fast IRC clock selected (C2[IRCS]=1). After it enters VLPR mode, writes to any of the MCG control registers that can cause an MCG clock mode switch to a non low power clock mode must be avoided.

### NOTE

For the chip-specific modes of operation, see the power management chapter of this MCU.

#### 24.4.1.2 MCG mode switching

The C1[IREFS] bit can be changed at any time, but the actual switch to the newly selected reference clocks is shown by the S[IREFST] bit. When switching between engaged internal and engaged external modes, the FLL will begin locking again after the switch is completed.

The C1[CLKS] bits can also be changed at any time, but the actual switch to the newly selected clock is shown by the S[CLKST] bits. If the newly selected clock is not available, the previous clock will remain selected.

The C4[DRST\_DRS] write bits can be changed at any time except when C2[LP] bit is 1. If the C4[DRST\_DRS] write bits are changed while in FLL engaged internal (FEI) or FLL engaged external (FEE), the MCGOUTCLK will switch to the new selected DCO range within three clocks of the selected DCO clock. After switching to the new DCO, the FLL remains unlocked for several reference cycles. DCO startup time is equal to the FLL acquisition time. After the selected DCO startup time is over, the FLL is locked. The completion of the switch is shown by the C4[DRST\_DRS] read bits.

## **24.4.2 Low Power Bit Usage**

The C2[LP] bit is provided to allow the FLL or PLL to be disabled and thus conserve power when these systems are not being used. The C4[DRST\_DRS] can not be written while C2[LP] bit is 1. However, in some applications, it may be desirable to enable the FLL or PLL and allow it to lock for maximum accuracy before switching to an engaged mode. Do this by writing C2[LP] to 0.

## **24.4.3 MCG Internal Reference Clocks**

This module supports two internal reference clocks with nominal frequencies of 32 kHz (slow IRC) and 4 MHz (fast IRC). The fast IRC frequency can be divided down by programming of the FCRDIV to produce a frequency range of 32 kHz to 4 MHz.

### **24.4.3.1 MCG Internal Reference Clock**

The MCG Internal Reference Clock (MCGIRCLK) provides a clock source for other on-chip peripherals and is enabled when C1[IRCLKEN]=1. When enabled, MCGIRCLK is driven by either the fast internal reference clock (4 MHz IRC which can be divided down by the FRDIV factors) or the slow internal reference clock (32 kHz IRC). The IRCS clock frequency can be re-targeted by trimming the period of its IRCS selected internal reference clock. This can be done by writing a new trim value to the C3[SCTRIM]:C4[SCFTRIM] bits when the slow IRC clock is selected or by writing a new trim value to the C4[FCTRIM] bits when the fast IRC clock is selected. The internal reference clock period is proportional to the trim value written.  
C3[SCTRIM]:C4[SCFTRIM] (if C2[IRCS]=0) and C4[FCTRIM] (if C2[IRCS]=1) bits

affect the MCGOUTCLK frequency if the MCG is in FBI or BLPI modes.

C3[SCTRIM]:C4[SCFTRIM] (if C2[IRCS]=0) bits also affect the MCGOUTCLK frequency if the MCG is in FEI mode.

Additionally, this clock can be enabled in Stop mode by setting C1[IRCLKEN] and C1[IREFSTEN], otherwise this clock is disabled in Stop mode.

#### 24.4.4 External Reference Clock

The MCG module can support an external reference clock in all modes. See the device datasheet for external reference frequency range. When C1[IREFS] is set, the external reference clock will not be used by the FLL or PLL. In these modes, the frequency can be equal to the maximum frequency the chip-level timing specifications will support.

If any of the CME bits are asserted the slow internal reference clock is enabled along with the enabled external clock monitor. For the case when C6[CME0]=1, a loss of clock is detected if the OSC0 external reference falls below a minimum frequency ( $f_{loc\_high}$  or  $f_{loc\_low}$  depending on C2[RANGE0]).

All clock monitors must be disabled before VLPR or VLPW power modes are entered.

Upon detect of a loss of clock event, the MCU generates a system reset if the respective LOCRE bit is set. Otherwise the MCG sets the respective LOCS bit and the MCG generates a LOCS interrupt request. In the case where a OSC0 loss of clock is detected, the PLL LOCK status bit is cleared if the OSC clock that is lost was selected as the PLL reference clock.

#### 24.4.5 MCG Fixed frequency clock

The MCG Fixed Frequency Clock (MCGFFCLK) provides a fixed frequency clock source for other on-chip peripherals; see the block diagram. This clock is driven by either the slow clock from the internal reference clock generator or the external reference clock from the Crystal Oscillator, divided by the FLL reference clock divider. The source of MCGFFCLK is selected by C1[IREFS].

This clock is synchronized to the peripheral bus clock and is valid only when its frequency is not more than 1/8 of the MCGOUTCLK frequency. When it is not valid, it is disabled and held high. The MCGFFCLK is not available when the MCG is in BLPI mode. This clock is also disabled in Stop mode. The FLL reference clock must be set within the valid frequency range for the MCGFFCLK.

## 24.4.6 MCG PLL clock

The MCG PLL Clock (MCGPLLCLK) is available depending on the device's configuration of the MCG module. For more details, see the clock distribution chapter of this MCU. The MCGPLLCLK is prevented from coming out of the MCG until it is enabled and S[LOCK0] is set.

## 24.4.7 MCG Auto TRIM (ATM)

The MCG Auto Trim (ATM) is a MCG feature that when enabled, it configures the MCG hardware to automatically trim the MCG Internal Reference Clocks using an external clock as a reference. The selection between which MCG IRC clock gets tested and enabled is controlled by the ATC[ATMS] control bit (ATC[ATMS]=0 selects the 32 kHz IRC and ATC[ATMS]=1 selects the 4 MHz IRC). If 4 MHz IRC is selected for the ATM, a divide by 128 is enabled to divide down the 4 MHz IRC to a range of 31.250 kHz.

When MCG ATM is enabled by writing ATC[ATME] bit to 1, The ATM machine will start auto trimming the selected IRC clock. During the autotrim process, ATC[ATME] will remain asserted and will deassert after ATM is completed or an abort occurs. The MCG ATM is aborted if a write to any of the following control registers is detected : C1, C3, C4, or ATC or if Stop mode is entered. If an abort occurs, ATC[ATMF] fail flag is asserted.

The ATM machine uses the bus clock as the external reference clock to perform the IRC auto-trim. Therefore, it is required that the MCG is configured in a clock mode where the reference clock used to generate the system clock is the external reference clock such as FBE clock mode. The MCG must not be configured in a clock mode where selected IRC ATM clock is used to generate the system clock. The bus clock is also required to be running with in the range of 8–16 MHz.

To perform the ATM on the selected IRC, the ATM machine uses the successive approximation technique to adjust the IRC trim bits to generate the desired IRC trimmed frequency. The ATM SARs each of the ATM IRC trim bits starting with the MSB. For each trim bit test, the ATM uses a pulse that is generated by the ATM selected IRC clock to enable a counter that counts number of ATM external clocks. At end of each trim bit, the ATM external counter value is compared to the ATCV[15:0] register value. Based on the comparison result, the ATM trim bit under test will get cleared or stay asserted. This is done until all trim bits have been tested by ATM SAR machine.

Before the ATM can be enabled, the ATM expected count needs to be derived and stored into the ATCV register. The ATCV expected count is derived based on the required target Internal Reference Clock (IRC) frequency, and the frequency of the external reference clock using the following formula:

$$\text{ATCV Expected Count Value} = 21 * (\text{Fe} / \text{Fr})$$

- Fr = Target Internal Reference Clock (IRC) Trimmed Frequency
- Fe = External Clock Frequency

If the auto trim is being performed on the 4 MHz IRC, the calculated expected count value must be multiplied by 128 before storing it in the ATCV register. Therefore, the ATCV Expected Count Value for trimming the 4 MHz IRC is calculated using the following formula.

$$\text{Expected Count Value} = (\text{Fe} / \text{Fr}) * 21 * (128)$$

## 24.5 Initialization / Application information

This section describes how to initialize and configure the MCG module in an application. The following sections include examples on how to initialize the MCG and properly switch between the various available modes.

### 24.5.1 MCG module initialization sequence

The MCG comes out of reset configured for FEI mode. The internal reference will stabilize in  $t_{irefsts}$  microseconds before the FLL can acquire lock. As soon as the internal reference is stable, the FLL will acquire lock in  $t_{fll\_acquire}$  milliseconds.

#### 24.5.1.1 Initializing the MCG

Because the MCG comes out of reset in FEI mode, the only MCG modes that can be directly switched to upon reset are FEE, FBE, and FBI modes (see [Figure 24-16](#)). Reaching any of the other modes requires first configuring the MCG for one of these three intermediate modes. Care must be taken to check relevant status bits in the MCG status register reflecting all configuration changes within each mode.

To change from FEI mode to FEE or FBE modes, follow this procedure:

1. Enable the external clock source by setting the appropriate bits in C2 register.

2. Write to C1 register to select the clock mode.

- If entering FEE mode, set C1[FRDIV] appropriately, clear the C1[IREFS] bit to switch to the external reference, and leave the C1[CLKS] bits at 2'b00 so that the output of the FLL is selected as the system clock source.
- If entering FBE, clear the C1[IREFS] bit to switch to the external reference and change the C1[CLKS] bits to 2'b10 so that the external reference clock is selected as the system clock source. The C1[FRDIV] bits should also be set appropriately here according to the external reference frequency to keep the FLL reference clock in the range of 31.25 kHz to 39.0625 kHz. Although the FLL is bypassed, it is still on in FBE mode.
- The internal reference can optionally be kept running by setting the C1[IRCLKEN] bit. This is useful if the application will switch back and forth between internal and external modes. For minimum power consumption, leave the internal reference disabled while in an external clock mode.

3. Once the proper configuration bits have been set, wait for the affected bits in the MCG status register to be changed appropriately, reflecting that the MCG has moved into the proper mode.

- If the MCG is in FEE, FBE, PEE, PBE, or BLPE mode, and C2[EREFS0] was also set in step 1, wait here for S[OSCINIT0] bit to become set indicating that the external clock source has finished its initialization cycles and stabilized.
- If in FEE mode, check to make sure the S[IREFST] bit is cleared before moving on.
- If in FBE mode, check to make sure the S[IREFST] bit is cleared and S[CLKST] bits have changed to 2'b10 indicating the external reference clock has been appropriately selected. Although the FLL is bypassed, it is still on in FBE mode.

4. Write to the C4 register to determine the DCO output (MCGFLLCLK) frequency range.

- By default, with C4[DMX32] cleared to 0, the FLL multiplier for the DCO output is 640. For greater flexibility, if a mid-low-range FLL multiplier of 1280 is desired instead, set C4[DRST\_DRS] bits to 2'b01 for a DCO output frequency of 40 MHz. If a mid high-range FLL multiplier of 1920 is desired instead, set the C4[DRST\_DRS] bits to 2'b10 for a DCO output frequency of 60 MHz. If a high-range FLL multiplier of 2560 is desired instead, set the C4[DRST\_DRS] bits to 2'b11 for a DCO output frequency of 80 MHz.

- When using a 32.768 kHz external reference, if the maximum low-range DCO frequency that can be achieved with a 32.768 kHz reference is desired, set C4[DRST\_DR] bits to 2'b00 and set C4[DMX32] bit to 1. The resulting DCO output (MCGOUTCLK) frequency with the new multiplier of 732 will be 24 MHz.
  - When using a 32.768 kHz external reference, if the maximum mid-range DCO frequency that can be achieved with a 32.768 kHz reference is desired, set C4[DRST\_DR] bits to 2'b01 and set C4[DMX32] bit to 1. The resulting DCO output (MCGOUTCLK) frequency with the new multiplier of 1464 will be 48 MHz.
  - When using a 32.768 kHz external reference, if the maximum mid high-range DCO frequency that can be achieved with a 32.768 kHz reference is desired, set C4[DRST\_DR] bits to 2'b10 and set C4[DMX32] bit to 1. The resulting DCO output (MCGOUTCLK) frequency with the new multiplier of 2197 will be 72 MHz.
  - When using a 32.768 kHz external reference, if the maximum high-range DCO frequency that can be achieved with a 32.768 kHz reference is desired, set C4[DRST\_DR] bits to 2'b11 and set C4[DMX32] bit to 1. The resulting DCO output (MCGOUTCLK) frequency with the new multiplier of 2929 will be 96 MHz.
5. Wait for the FLL lock time to guarantee FLL is running at new C4[DRST\_DR] and C4[DMX32] programmed frequency.

To change from FEI clock mode to FBI clock mode, follow this procedure:

1. Change C1[CLKS] bits in C1 register to 2'b01 so that the internal reference clock is selected as the system clock source.
2. Wait for S[CLKST] bits in the MCG status register to change to 2'b01, indicating that the internal reference clock has been appropriately selected.
3. Write to the C2 register to determine the IRCS output (IRCSCLK) frequency range.
  - By default, with C2[IRCS] cleared to 0, the IRCS selected output clock is the slow internal reference clock (32 kHz IRC). If the faster IRC is desired, set C2[IRCS] bit to 1 for a IRCS clock derived from the 4 MHz IRC source.

## 24.5.2 Using a 32.768 kHz reference

In FEE and FBE modes, if using a 32.768 kHz external reference, at the default FLL multiplication factor of 640, the DCO output (MCGFLLCLK) frequency is 20.97 MHz at low-range. If C4[DRST\_DR] bits are set to 2'b01, the multiplication factor is doubled to 1280, and the resulting DCO output frequency is 41.94 MHz at mid-low-range. If C4[DRST\_DR] bits are set to 2'b10, the multiplication factor is set to 1920, and the resulting DCO output frequency is 62.91 MHz at mid high-range. If C4[DRST\_DR] bits are set to 2'b11, the multiplication factor is set to 2560, and the resulting DCO output frequency is 83.89 MHz at high-range.

In FBI and FEI modes, setting C4[DMX32] bit is not recommended. If the internal reference is trimmed to a frequency above 32.768 kHz, the greater FLL multiplication factor could potentially push the microcontroller system clock out of specification and damage the part.

## 24.5.3 MCG mode switching

When switching between operational modes of the MCG, certain configuration bits must be changed in order to properly move from one mode to another. Each time any of these bits are changed (C6[PLLS], C1[IREFS], C1[CLKS], C2[IRCS], or C2[EREFS0]), the corresponding bits in the MCG status register (PLLST, IREFST, CLKST, IRCST, or OSCINIT) must be checked before moving on in the application software.

Additionally, care must be taken to ensure that the reference clock divider (C1[FRDIV] and C5[PRDIV0]) is set properly for the mode being switched to. For instance, in PEE mode, if using a 4 MHz crystal, C5[PRDIV0] must be set to 5'b000 (divide-by-1) or 5'b001 (divide -by-2) to divide the external reference down to the required frequency between 2 and 4 MHz.

In FBE, FEE, FBI, and FEI modes, at any time, the application can switch the FLL multiplication factor between 640, 1280, 1920, and 2560 with C4[DRST\_DR] bits. Writes to C4[DRST\_DR] bits will be ignored if C2[LP]=1.

The table below shows MCGOUTCLK frequency calculations using C1[FRDIV], C5[PRDIV0], and C6[VDIV0] settings for each clock mode.

**Table 24-19. MCGOUTCLK Frequency Calculation Options**

| Clock Mode                 | $f_{MCGOUTCLK}^1$        | Note  |
|----------------------------|--------------------------|---|
| FEI (FLL engaged internal) | $(f_{int} * F)$          | Typical $f_{MCGOUTCLK} = 21$ MHz immediately after reset.           |
| FEE (FLL engaged external) | $(f_{ext} / FLL\_R) * F$ | $f_{ext} / FLL\_R$ must be in the range of 31.25 kHz to 39.0625 kHz |

*Table continues on the next page...*

**Table 24-19. MCGOUTCLK Frequency Calculation Options  
(continued)**

| Clock Mode                         | $f_{MCGOUTCLK}^1$       | Note  |
|------------------------------------|-------------------------|---|
| FBE (FLL bypassed external)        | OSCCLK                  | OSCCLK / FLL_R must be in the range of 31.25 kHz to 39.0625 kHz |
| FBI (FLL bypassed internal)        | MCGIRCLK                | Selectable between slow and fast IRC                            |
| PEE (PLL engaged external)         | $(OSCCLK / PLL\_R) * M$ | OSCCLK / PLL_R must be in the range of 2 – 4 MHz                |
| PBE (PLL bypassed external)        | OSCCLK                  | OSCCLK / PLL_R must be in the range of 2 – 4 MHz                |
| BLPI (Bypassed low power internal) | MCGIRCLK                | Selectable between slow and fast IRC                            |
| BLPE (Bypassed low power external) | OSCCLK                  |   |

1. FLL\_R is the reference divider selected by the C1[FRDIV] bits, PLL\_R is the reference divider selected by C5[PRDIV0] bits, F is the FLL factor selected by C4[DRST\_DRS] and C4[DMX32] bits, and M is the multiplier selected by C6[VDIV0] bits.

This section will include three mode switching examples using an 4 MHz external crystal. If using an external clock source less than 2 MHz, the MCG must not be configured for any of the PLL modes (PEE and PBE).

#### 24.5.3.1 Example 1: Moving from FEI to PEE mode: External Crystal = 4 MHz, MCGOUTCLK frequency = 48 MHz

In this example, the MCG will move through the proper operational modes from FEI to PEE to achieve 48 MHz MCGOUTCLK frequency from 4 MHz external crystal reference. First, the code sequence will be described. Then there is a flowchart that illustrates the sequence.

1. First, FEI must transition to FBE mode:

a.  $C2 = 0x1C$

- C2[RANGE0] set to 2'b01 because the frequency of 4 MHz is within the high frequency range.
- C2[HGO0] set to 1 to configure the crystal oscillator for high gain operation.
- C2[EREFS0] set to 1, because a crystal is being used.

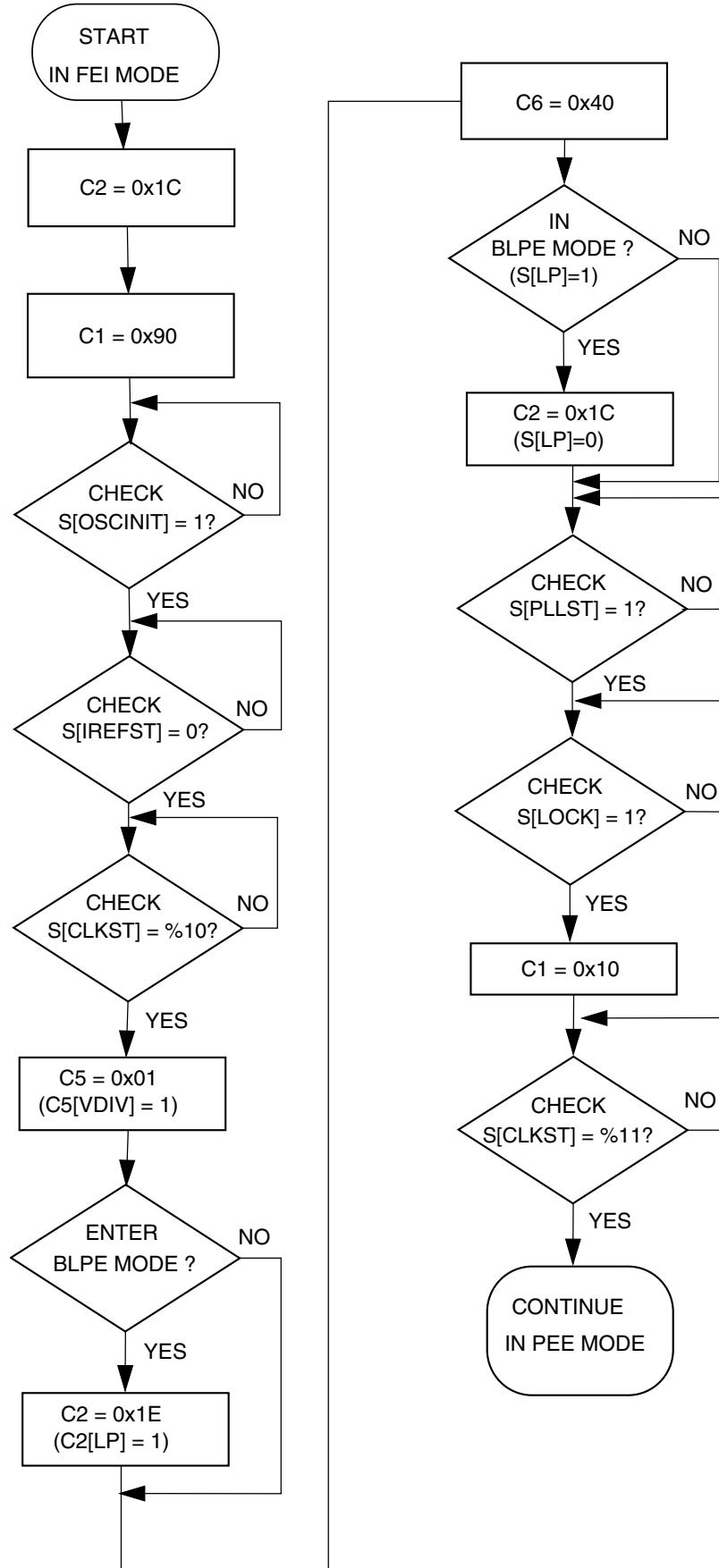
b.  $C1 = 0x90$

- C1[CLKS] set to 2'b10 to select external reference clock as system clock source

- C1[FRDIV] set to 3'b010, or divide-by-128 because  $4 \text{ MHz} / 128 = 31.25 \text{ kHz}$  which is in the 31.25 kHz to 39.0625 kHz range required by the FLL
  - C1[IREFS] cleared to 0, selecting the external reference clock and enabling the external oscillator.
- c. Loop until S[OSCINIT0] is 1, indicating the crystal selected by C2[EREFS0] has been initialized.
  - d. Loop until S[IREFST] is 0, indicating the external reference is the current source for the reference clock.
  - e. Loop until S[CLKST] is 2'b10, indicating that the external reference clock is selected to feed MCGOUTCLK.
2. Then configure C5[PRDIV0] to generate correct PLL reference frequency.
- a. C5 = 0x01
    - C5[PRDIV0] set to 5'b001, or divide-by-2 resulting in a pll reference frequency of  $4 \text{ MHz}/2 = 2 \text{ MHz}$ .
3. Then, FBE must transition either directly to PBE mode or first through BLPE mode and then to PBE mode:
- a. BLPE: If a transition through BLPE mode is desired, first set C2[LP] to 1.
  - b. BLPE/PBE: C6 = 0x40
    - C6[PLLS] set to 1, selects the PLL. At this time, with a C1[PRDIV] value of 2'b001, the PLL reference divider is 2 (see PLL External Reference Divide Factor table), resulting in a reference frequency of  $4 \text{ MHz}/2 = 2 \text{ MHz}$ . In BLPE mode, changing the C6[PLLS] bit only prepares the MCG for PLL usage in PBE mode.
    - C6[VDIV0] set to 5'b0000, or multiply-by-24 because  $2 \text{ MHz} \text{ reference} * 24 = 48 \text{ MHz}$ . In BLPE mode, the configuration of the VDIV bits does not matter because the PLL is disabled. Changing them only sets up the multiply value for PLL usage in PBE mode.
  - c. BLPE: If transitioning through BLPE mode, clear C2[LP] to 0 here to switch to PBE mode.
  - d. PBE: Loop until S[PLLST] is set, indicating that the current source for the PLLS clock is the PLL.
  - e. PBE: Then loop until S[LOCK0] is set, indicating that the PLL has acquired lock.

4. Lastly, PBE mode transitions into PEE mode:

- a. C1 = 0x10
  - C1[CLKS] set to 2'b00 to select the output of the PLL as the system clock source.
- b. Loop until S[CLKST] are 2'b11, indicating that the PLL output is selected to feed MCGOUTCLK in the current clock mode.
  - Now, with PRDIV0 of divide-by-2, and C6[VDIV0] of multiply-by-24,  
 $MCGOUTCLK = [(4 \text{ MHz} / 2) * 24] = 48 \text{ MHz}$ .



**Figure 24-17. Flowchart of FEI to PEE mode transition using an 4 MHz crystal**  
KL25 Sub-Family Reference Manual, Rev. 3, September 2012

### 24.5.3.2 Example 2: Moving from PEE to BLPI mode: MCGOUTCLK frequency =32 kHz

In this example, the MCG will move through the proper operational modes from PEE mode with a 4 MHz crystal configured for a 48 MHz MCGOUTCLK frequency (see previous example) to BLPI mode with a 32 kHz MCGOUTCLK frequency. First, the code sequence will be described. Then there is a flowchart that illustrates the sequence.

1. First, PEE must transition to PBE mode:
  - a. C1 = 0x90
    - C1[CLKS] set to 2'b10 to switch the system clock source to the external reference clock.
  - b. Loop until S[CLKST] are 2'b10, indicating that the external reference clock is selected to feed MCGOUTCLK.
2. Then, PBE must transition either directly to FBE mode or first through BLPE mode and then to FBE mode:
  - a. BLPE: If a transition through BLPE mode is desired, first set C2[LP] to 1.
  - b. BLPE/FBE: C6 = 0x00
    - C6[PLLS] clear to 0 to select the FLL. At this time, with C1[FRDIV] value of 3'b010, the FLL divider is set to 128, resulting in a reference frequency of  $4\text{ MHz} / 128 = 31.25\text{ kHz}$ . If C1[FRDIV] was not previously set to 3'b010 (necessary to achieve required 31.25–39.06 kHz FLL reference frequency with an 4 MHz external source frequency), it must be changed prior to clearing C6[PLLS] bit. In BLPE mode, changing this bit only prepares the MCG for FLL usage in FBE mode. With C6[PLLS] = 0, the C6[VDIV0] value does not matter.
    - c. BLPE: If transitioning through BLPE mode, clear C2[LP] to 0 here to switch to FBE mode.
    - d. FBE: Loop until S[PLLST] is cleared, indicating that the current source for the PLLS clock is the FLL.
3. Next, FBE mode transitions into FBI mode:
  - a. C1 = 0x54
    - C1[CLKS] set to 2'b01 to switch the system clock to the internal reference clock.

- C1[IREFS] set to 1 to select the internal reference clock as the reference clock source.
  - C1[FRDIV] remain unchanged because the reference divider does not affect the internal reference.
- b. Loop until S[IREFST] is 1, indicating the internal reference clock has been selected as the reference clock source.
- c. Loop until S[CLKST] are 2'b01, indicating that the internal reference clock is selected to feed MCGOUTCLK.
4. Lastly, FBI transitions into BLPI mode.
- a. C2 = 0x02
- C2[LP] is 1
  - C2[RANGE0], C2[HGO0], C2[EREFS0], C1[IRCLKEN], and C1[IREFSTEN] bits are ignored when the C1[IREFS] bit is set. They can remain set, or be cleared at this point.

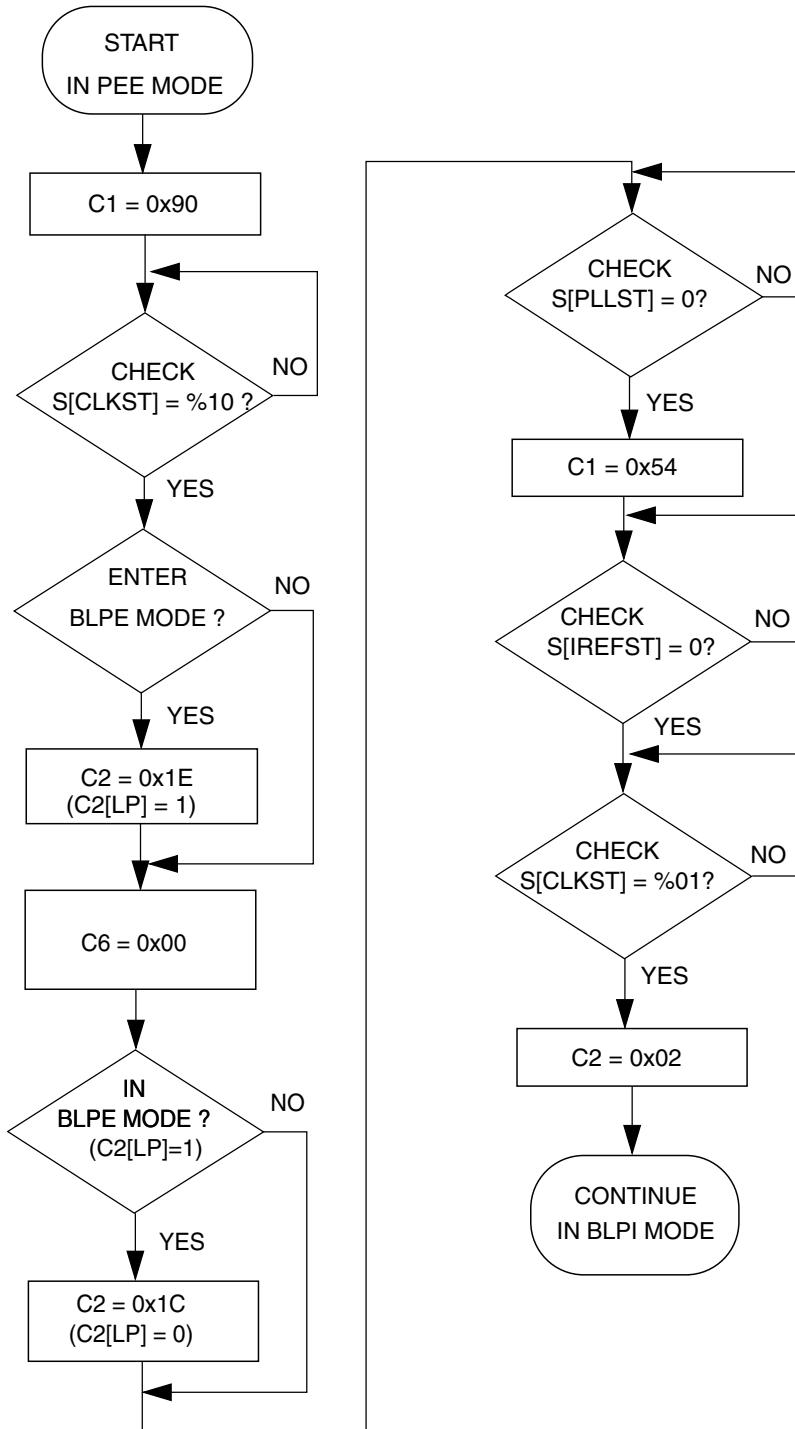


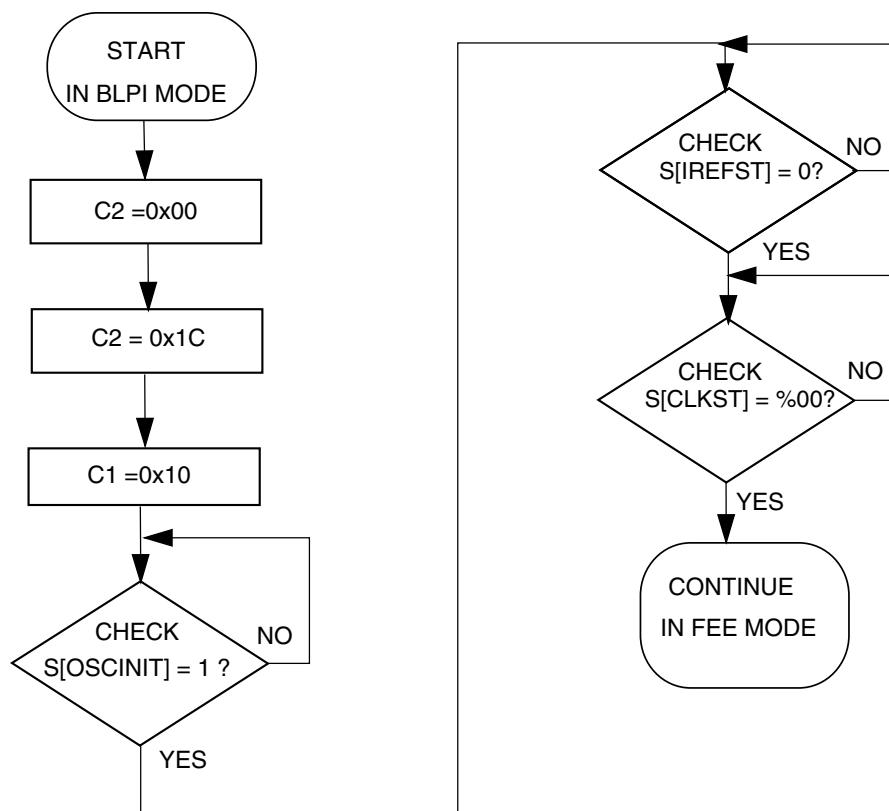
Figure 24-18. Flowchart of PEE to BLPI mode transition using a 4 MHz crystal

### 24.5.3.3 Example 3: Moving from BLPI to FEE mode

In this example, the MCG will move through the proper operational modes from BLPI mode at a 32 kHz MCGOUTCLK frequency running off the internal reference clock (see previous example) to FEE mode using a 4 MHz crystal configured for a 20 MHz MCGOUTCLK frequency. First, the code sequence will be described. Then there is a flowchart that illustrates the sequence.

1. First, BLPI must transition to FBI mode.
  - a.  $C2 = 0x00$ 
    - $C2[LP]$  is 0
2. Next, FBI will transition to FEE mode.
  - a.  $C2 = 0x1C$ 
    - $C2[RANGE0]$  set to 2'b01 because the frequency of 4 MHz is within the high frequency range.
    - $C2[HGO0]$  set to 1 to configure the crystal oscillator for high gain operation.
    - $C2[EREF0]$  set to 1, because a crystal is being used.
  - b.  $C1 = 0x10$ 
    - $C1[CLKS]$  set to 2'b00 to select the output of the FLL as system clock source.
    - $C1[FRDIV]$  remain at 3'b010, or divide-by-128 for a reference of 4 MHz / 128 = 31.25 kHz.
    - $C1[IREFS]$  cleared to 0, selecting the external reference clock.
  - c. Loop until  $S[OSCINIT0]$  is 1, indicating the crystal selected by the  $C2[EREF0]$  bit has been initialized.
  - d. Loop until  $S[IREFST]$  is 0, indicating the external reference clock is the current source for the reference clock.
  - e. Loop until  $S[CLKST]$  are 2'b00, indicating that the output of the FLL is selected to feed MCGOUTCLK.
  - f. Now, with a 31.25 kHz reference frequency, a fixed DCO multiplier of 640,  $MCGOUTCLK = 31.25 \text{ kHz} * 640 / 1 = 20 \text{ MHz}$ .
  - g. At this point, by default, the  $C4[DRST_DRS]$  bits are set to 2'b00 and  $C4[DMX32]$  is cleared to 0. If the MCGOUTCLK frequency of 40 MHz is desired instead, set the  $C4[DRST_DRS]$  bits to 0x01 to switch the FLL

multiplication factor from 640 to 1280. To return the MCGOUTCLK frequency to 20 MHz, set C4[DRST\_DRST] bits to 2'b00 again, and the FLL multiplication factor will switch back to 640.



**Figure 24-19. Flowchart of BLPI to FEE mode transition using an 4 MHz crystal**



# **Chapter 25**

## **Oscillator (OSC)**

### **25.1 Introduction**

#### **NOTE**

For the chip-specific implementation details of this module's instances see the chip configuration information.

The OSC module is a crystal oscillator. The module, in conjunction with an external crystal or resonator, generates a reference clock for the MCU.

### **25.2 Features and Modes**

Key features of the module are:

- Supports 32 kHz crystals (Low Range mode)
- Voltage and frequency filtering to guarantee clock frequency and stability
- Optionally external input bypass clock from EXTAL signal directly
- One clock for MCU clock system
- Two clocks for on-chip peripherals that can work in Stop modes

[Functional Description](#) describes the module's operation in more detail.

## 25.3 Block Diagram

The OSC module uses a crystal or resonator to generate three filtered oscillator clock signals. Three clocks are output from OSC module: OSCCLK for MCU system, OSCERCLK for on-chip peripherals, and OSC32KCLK. The OSCCLK can only work in run mode. OSCERCLK and OSC32KCLK can work in low power modes. For the clock source assignments, refer to the clock distribution information of this MCU.

Refer to the chip configuration chapter for the external reference clock source in this MCU.

The following figure shows the block diagram of the OSC module.

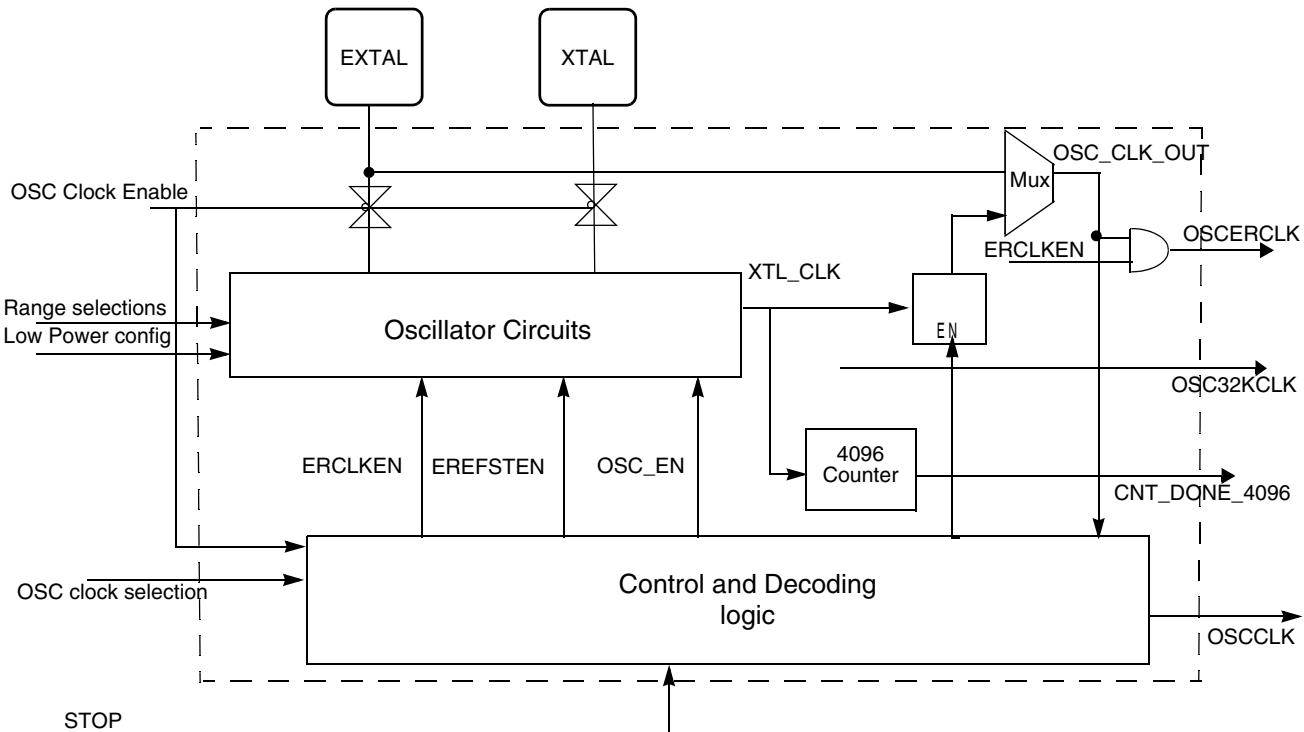


Figure 25-1. OSC Module Block Diagram

## 25.4 OSC Signal Descriptions

The following table shows the user-accessible signals available for the OSC module. Refer to signal multiplexing information for this MCU for more details.

**Table 25-1. OSC Signal Descriptions**

| Signal | Description                     | I/O |
|--------|---------------------------------|-----|
| EXTAL  | External clock/Oscillator input | I   |
| XTAL   | Oscillator output               | O   |

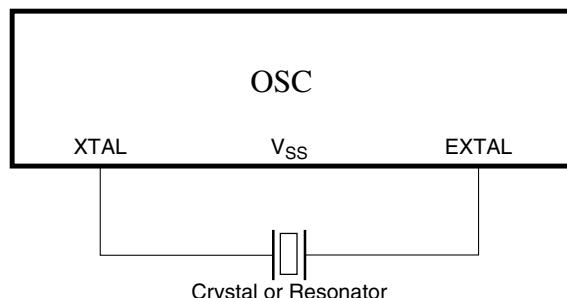
## 25.5 External Crystal / Resonator Connections

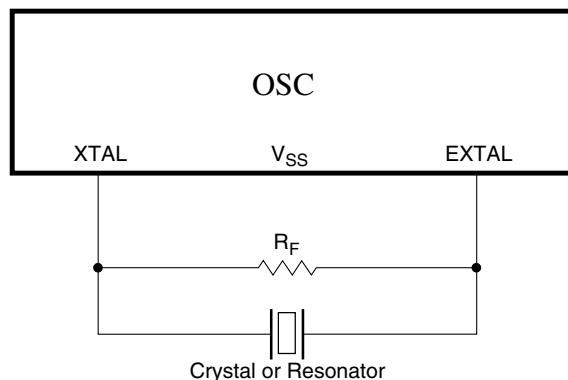
The connections for a crystal/resonator frequency reference are shown in the following figures. When using low-frequency, low-power mode, the only external component is the crystal or ceramic resonator itself. In the other oscillator modes, load capacitors ( $C_x$ ,  $C_y$ ) and feedback resistor ( $R_F$ ) are required. The following table shows all possible connections.

**Table 25-2. External Crystal/Resonator Connections**

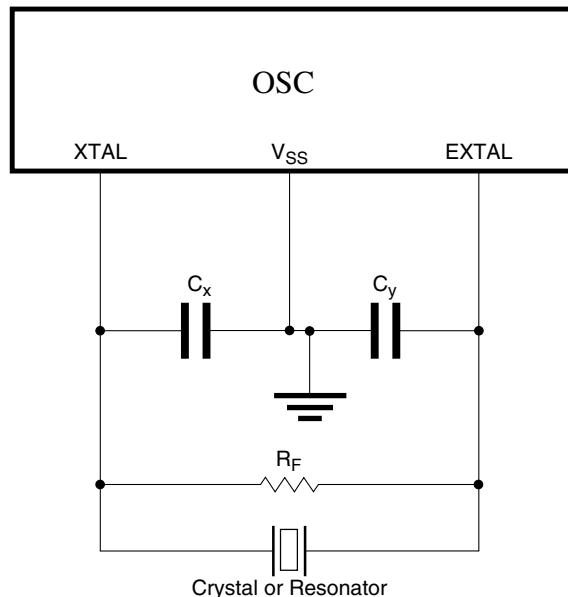
| Oscillator Mode                      | Connections                            |
|--------------------------------------|--|
| Low-frequency (32 kHz), low-power    | Connection 1 <sup>1</sup>              |
| Low-frequency (32 kHz), high-gain    | Connection 2/Connection 3 <sup>2</sup> |
| High-frequency (3~32 MHz), low-power | Connection 3 <sup>1</sup>              |
| High-frequency (3~32 MHz), high-gain | Connection 3                           |

- With the low-power mode, the oscillator has the internal feedback resistor  $R_F$ . Therefore, the feedback resistor must not be externally with the Connection 3.
- When the load capacitors ( $C_x$ ,  $C_y$ ) are greater than 30 pF, use Connection 3.

**Figure 25-2. Crystal/Ceramic Resonator Connections - Connection 1**

**Figure 25-3. Crystal/Ceramic Resonator Connections - Connection 2****NOTE**

Connection 1 and Connection 2 should use internal capacitors as the load of the oscillator by configuring the CR[SCxP] bits.

**Figure 25-4. Crystal/Ceramic Resonator Connections - Connection 3**

## 25.6 External Clock Connections

In external clock mode, the pins can be connected as shown below.

**NOTE**

XTAL can be used as a GPIO when the GPIO alternate function is configured for it.

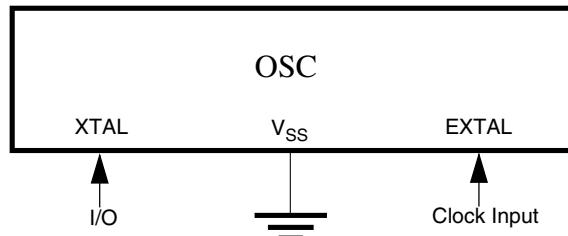


Figure 25-5. External Clock Connections

## 25.7 Memory Map/Register Definitions

Some oscillator module register bits are typically incorporated into other peripherals such as MCG or SIM.

### 25.7.1 OSC Memory Map/Register Definition

OSC memory map

| Absolute address (hex) | Register name                  | Width (in bits) | Access | Reset value | Section/page |
|------------------------|--------------------------------|-----------------|--------|-------------|--------------|
| 4006_5000              | OSC Control Register (OSC0_CR) | 8               | R/W    | 00h         | 25.71.1/409  |

### 25.7.1.1 OSC Control Register (OSCx\_CR)

#### NOTE

After OSC is enabled and starts generating the clocks, the configurations such as low power and frequency range, must not be changed.

Address: 4006\_5000h base + 0h offset = 4006\_5000h

| Bit   | 7       | 6 | 5        | 4 | 3    | 2    | 1    | 0     |
|-------|---------|---|----------|---|------|------|------|-------|
| Read  | ERCLKEN | 0 | EREFSTEN | 0 | SC2P | SC4P | SC8P | SC16P |
| Write | 0       | 0 | 0        | 0 | 0    | 0    | 0    | 0     |
| Reset | 0       | 0 | 0        | 0 | 0    | 0    | 0    | 0     |

#### OSCx\_CR field descriptions

| Field        | Description   |
|--------------|---|
| 7<br>ERCLKEN | External Reference Enable<br>Enables external reference clock (OSCERCLK). |

Table continues on the next page...

**OSC<sub>x</sub>\_CR field descriptions (continued)**

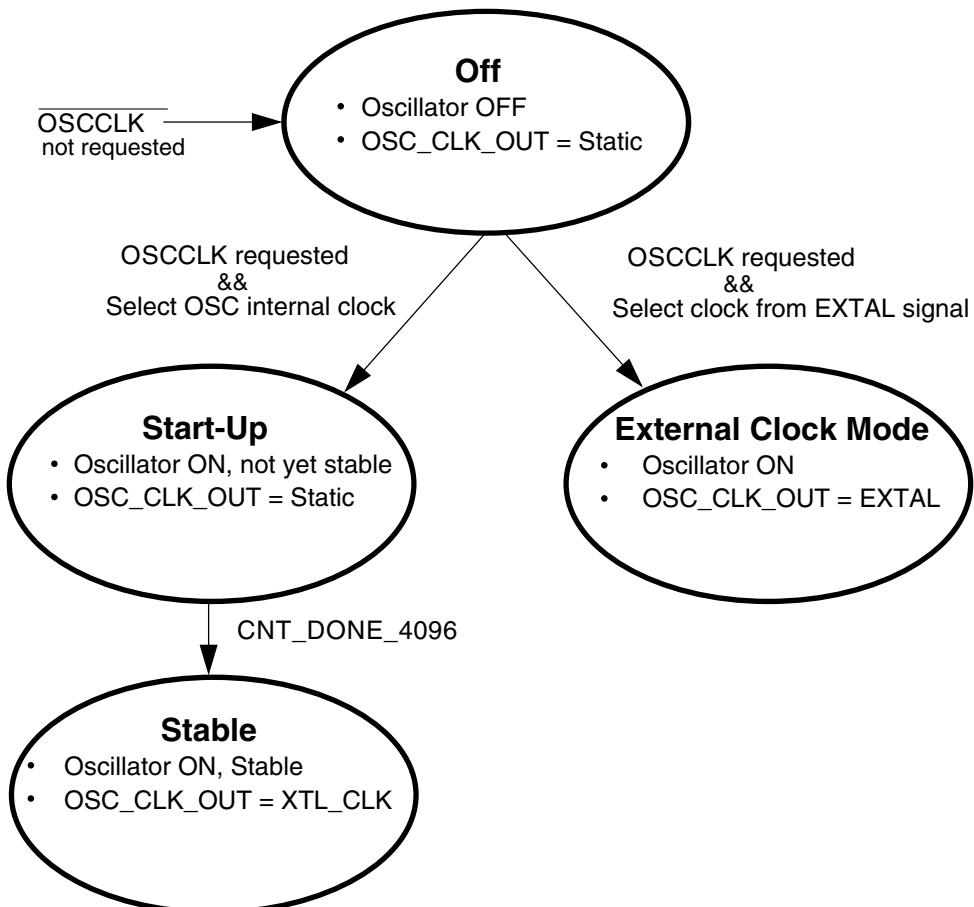
| Field      | Description   |
|------------|---|
|            | 0 External reference clock is inactive.<br>1 External reference clock is enabled.   |
| 6 Reserved | This field is reserved.<br>This read-only field is reserved and always has the value 0.   |
| 5 EREFSTEN | <b>External Reference Stop Enable</b><br>Controls whether or not the external reference clock (OSCERCLK) remains enabled when MCU enters Stop mode.<br>0 External reference clock is disabled in Stop mode.<br>1 External reference clock stays enabled in Stop mode if ERCLKEN is set before entering Stop mode. |
| 4 Reserved | This field is reserved.<br>This read-only field is reserved and always has the value 0.   |
| 3 SC2P     | <b>Oscillator 2 pF Capacitor Load Configure</b><br>Configures the oscillator load.<br>0 Disable the selection.<br>1 Add 2 pF capacitor to the oscillator load.  |
| 2 SC4P     | <b>Oscillator 4 pF Capacitor Load Configure</b><br>Configures the oscillator load.<br>0 Disable the selection.<br>1 Add 4 pF capacitor to the oscillator load.  |
| 1 SC8P     | <b>Oscillator 8 pF Capacitor Load Configure</b><br>Configures the oscillator load.<br>0 Disable the selection.<br>1 Add 8 pF capacitor to the oscillator load.  |
| 0 SC16P    | <b>Oscillator 16 pF Capacitor Load Configure</b><br>Configures the oscillator load.<br>0 Disable the selection.<br>1 Add 16 pF capacitor to the oscillator load.  |

## 25.8 Functional Description

This following sections provide functional details of the module.

### 25.8.1 OSC Module States

The states of the OSC module are shown in the following figure. The states and their transitions between each other are described in this section.

**Figure 25-8. OSC Module State Diagram****NOTE**

XTL\_CLK is the clock generated internally from OSC circuits.

**25.8.1.1 Off**

The OSC enters the Off state when the system does not require OSC clocks. Upon entering this state, XTL\_CLK is static unless OSC is configured to select the clock from the EXTAL pad by clearing the external reference clock selection bit. For details regarding the external reference clock source in this MCU, refer to the chip configuration chapter. The EXTAL and XTAL pins are also decoupled from all other oscillator circuitry in this state. The OSC module circuitry is configured to draw minimal current.

### 25.8.1.2 Oscillator Start-Up

The OSC enters start-up state when it is configured to generate clocks (internally the OSC\_EN transitions high) using the internal oscillator circuits by setting the external reference clock selection bit. In this state, the OSC module is enabled and oscillations are starting up, but have not yet stabilized. When the oscillation amplitude becomes large enough to pass through the input buffer, XTL\_CLK begins clocking the counter. When the counter reaches 4096 cycles of XTL\_CLK, the oscillator is considered stable and XTL\_CLK is passed to the output clock OSC\_CLK\_OUT.

### 25.8.1.3 Oscillator Stable

The OSC enters stable state when it is configured to generate clocks (internally the OSC\_EN transitions high) using the internal oscillator circuits by setting the external reference clock selection bit and the counter reaches 4096 cycles of XTL\_CLK (when CNT\_DONE\_4096 is high). In this state, the OSC module is producing a stable output clock on OSC\_CLK\_OUT. Its frequency is determined by the external components being used.

### 25.8.1.4 External Clock Mode

The OSC enters external clock state when it is enabled and external reference clock selection bit is cleared. For details regarding external reference clock source in this MCU, refer to the chip configuration chapter. In this state, the OSC module is set to buffer (with hysteresis) a clock from EXTAL onto the OSC\_CLK\_OUT. Its frequency is determined by the external clock being supplied.

## 25.8.2 OSC Module Modes

The OSC is a Pierce-type oscillator that supports external crystals or resonators operating over the frequency ranges shown in [Table 25-7](#). These modes assume the following conditions: OSC is enabled to generate clocks (OSC\_EN=1), configured to generate clocks internally (MCG\_C2[EREFS] = 1), and some or one of the other peripherals (MCG, Timer, and so on) is configured to use the oscillator output clock (OSC\_CLK\_OUT).

**Table 25-7. Oscillator Modes**

| Mode                            | Frequency Range   |
|---------------------------------|---|
| High-frequency mode1, high-gain | $f_{osc\_hi\_1}$ (3 MHz) up to $f_{osc\_hi\_1}$ (8 MHz) |
| High-frequency mode1, low-power |   |

**NOTE**

For information about low power modes of operation used in this chip and their alignment with some OSC modes, refer to the chip's Power Management details.

### 25.8.2.1 Low-Frequency, Low-Power Mode

In low-frequency, low-power mode, the oscillator uses a gain control loop to minimize power consumption. As the oscillation amplitude increases, the amplifier current is reduced. This continues until a desired amplitude is achieved at steady-state. This mode provides low pass frequency filtering as well as hysteresis for voltage filtering and converts the output to logic levels. In this mode, the internal capacitors could be used, the internal feedback resistor is connected, and no external resistor should be used.

In this mode, the amplifier inputs, gain-control input, and input buffer input are all capacitively coupled for leakage tolerance (not sensitive to the DC level of EXTAL).

Also in this mode, all external components except for the resonator itself are integrated, which includes the load capacitors and feedback resistor that biases EXTAL.

### 25.8.3 Counter

The oscillator output clock (OSC\_CLK\_OUT) is gated off until the counter has detected 4096 cycles of its input clock (XTL\_CLK). After 4096 cycles are completed, the counter passes XTL\_CLK onto OSC\_CLK\_OUT. This counting time-out is used to guarantee output clock stability.

### 25.8.4 Reference Clock Pin Requirements

The OSC module requires use of both the EXTAL and XTAL pins to generate an output clock in Oscillator mode, but requires only the EXTAL pin in External clock mode. The EXTAL and XTAL pins are available for I/O. For the implementation of these pins on this device, refer to the Signal Multiplexing chapter.

## 25.9 Reset

There is no reset state associated with the OSC module. The counter logic is reset when the OSC is not configured to generate clocks.

There are no sources of reset requests for the OSC module.

## 25.10 Low Power Modes Operation

When the MCU enters Stop modes, the OSC is functional depending on ERCLKEN and EREFSETN bit settings. If both these bits are set, the OSC is in operation. In Low Leakage Stop (LLS) modes, the OSC holds all register settings. If ERCLKEN and EREFSTEN bits are set before entry to Low Leakage Stop modes, the OSC is still functional in these modes. After waking up from Very Low Leakage Stop (VLLSx) modes, all OSC register bits are reset and initialization is required through software.

## 25.11 Interrupts

The OSC module does not generate any interrupts.

# **Chapter 26**

## **Flash Memory Controller (FMC)**

### **26.1 Introduction**

The Flash Memory Controller (FMC) is a memory acceleration unit that provides:

- an interface between bus masters and the 32-bit program flash memory.
- a buffer and a cache that can accelerate program flash memory data transfers.

#### **26.1.1 Overview**

The Flash Memory Controller manages the interface between bus masters and the 32-bit program flash memory. The FMC receives status information detailing the configuration of the flash memory and uses this information to ensure a proper interface. The FMC supports 8-bit, 16-bit, and 32-bit read operations from the program flash memory. A write operation to program flash memory results in a bus error.

In addition, the FMC provides two separate mechanisms for accelerating the interface between bus masters and program flash memory. A 32-bit speculation buffer can prefetch the next 32-bit flash memory location, and a 4-way, 4-set program flash memory cache can store previously accessed program flash memory data for quick access times.

#### **26.1.2 Features**

The FMC's features include:

- Interface between bus masters and the 32-bit program flash memory:
  - 8-bit, 16-bit, and 32-bit read operations to nonvolatile flash memory.
- Acceleration of data transfer from the program flash memory to the device:

- 32-bit prefetch speculation buffer for program flash accesses with controls for instruction/data access
- 4-way, 4-set, 32-bit line size program flash memory cache for a total of sixteen 32-bit entries with invalidation control

## 26.2 Modes of operation

The FMC operates only when a bus master accesses the program flash memory. In terms of chip power modes:

- The FMC operates only in run and wait modes, including VLPR and VLPW modes.
- For any power mode where the program flash memory cannot be accessed, the FMC is disabled.

## 26.3 External signal description

The FMC has no external (off-chip) signals.

## 26.4 Memory map and register descriptions

The MCM's programming model provides control and configuration of the FMC's features. For details, see the description of the MCM's Platform Control Register (PLACR).

## 26.5 Functional description

The FMC is a flash acceleration unit with flexible buffers for user configuration. Besides managing the interface between bus masters and the program flash memory, the FMC can be used to customize the program flash memory cache and buffer to provide single-cycle system clock data access times. Whenever a hit occurs for the prefetch speculation buffer or the cache (when enabled), the requested data is transferred within a single system clock.

Upon system reset, the FMC is configured as follows:

- Flash cache is enabled
- Instruction speculation and caching are enabled

- Data speculation is disabled
- Data caching is enabled

Though the default configuration provides flash acceleration, advanced users may desire to customize the FMC buffer configurations to maximize throughput for their use cases. For example, the user may adjust the controls to enable buffering per access type (data or instruction).

### **NOTE**

When reconfiguring the FMC, do not program the control and configuration inputs to the FMC while the program flash memory is being accessed. Instead, change them with a routine executing from RAM in supervisor mode.



# Chapter 27

## Flash Memory Module (FTFA)

### 27.1 Introduction

#### NOTE

For the chip-specific implementation details of this module's instances see the chip configuration information.

The flash memory module includes the following accessible memory regions:

- Program flash memory for vector space and code store

Flash memory is ideal for single-supply applications, permitting in-the-field erase and reprogramming operations without the need for any external high voltage power sources.

The flash memory module includes a memory controller that executes commands to modify flash memory contents. An erased bit reads '1' and a programmed bit reads '0'. The programming operation is unidirectional; it can only move bits from the '1' state (erased) to the '0' state (programmed). Only the erase operation restores bits from '0' to '1'; bits cannot be programmed from a '0' to a '1'.

#### CAUTION

A flash memory location must be in the erased state before being programmed. Cumulative programming of bits (back-to-back program operations without an intervening erase) within a flash memory location is not allowed. Re-programming of existing 0s to 0 is not allowed as this overstresses the device.

The standard shipping condition for flash memory is erased with security disabled. Data loss over time may occur due to degradation of the erased ('1') states and/or programmed ('0') states. Therefore, it is recommended that each flash block or sector be re-erased immediately prior to factory programming to ensure that the full data retention capability is achieved.

## 27.1.1 Features

The flash memory module includes the following features.

### NOTE

See the device's Chip Configuration details for the exact amount of flash memory available on your device.

### 27.1.1.1 Program Flash Memory Features

- Sector size of 1 Kbyte
- Program flash protection scheme prevents accidental program or erase of stored data
- Automated, built-in, program and erase algorithms with verify

### 27.1.1.2 Other Flash Memory Module Features

- Internal high-voltage supply generator for flash memory program and erase operations
- Optional interrupt generation upon flash command completion
- Supports MCU security mechanisms which prevent unauthorized access to the flash memory contents

## 27.1.2 Block Diagram

The block diagram of the flash memory module is shown in the following figure.

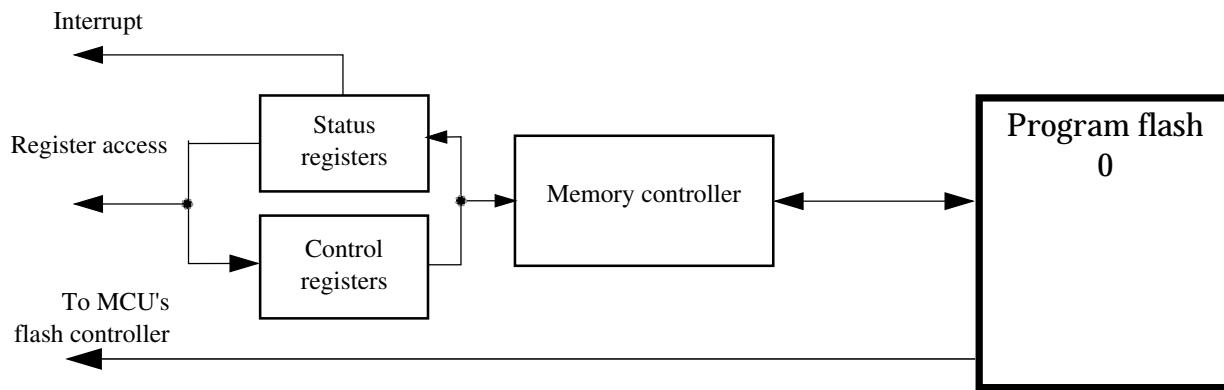


Figure 27-1. Flash Block Diagram

### 27.1.3 Glossary

**Command write sequence** — A series of MCU writes to the flash FCCOB register group that initiates and controls the execution of flash algorithms that are built into the flash memory module.

**Endurance** — The number of times that a flash memory location can be erased and reprogrammed.

**FCCOB (Flash Common Command Object)** — A group of flash registers that are used to pass command, address, data, and any associated parameters to the memory controller in the flash memory module.

**Flash block** — A macro within the flash memory module which provides the nonvolatile memory storage.

**Flash Memory Module** — All flash blocks plus a flash management unit providing high-level control and an interface to MCU buses.

**IFR** — Nonvolatile information register found in each flash block, separate from the main memory array.

**NVM** — Nonvolatile memory. A memory technology that maintains stored data during power-off. The flash array is an NVM using NOR-type flash memory technology.

**NVM Normal Mode** — An NVM mode that provides basic user access to flash memory module resources. The CPU or other bus masters initiate flash program and erase operations (or other flash commands) using writes to the FCCOB register group in the flash memory module.

**NVM Special Mode** — An NVM mode enabling external, off-chip access to the memory resources in the flash memory module. A reduced flash command set is available when the MCU is secured. See the Chip Configuration details for information on when this mode is used.

**Longword** — 32 bits of data with an aligned longword having byte-address[1:0] = 00.

**Word** — 16 bits of data with an aligned word having byte-address[0] = 0.

**Program flash** — The program flash memory provides nonvolatile storage for vectors and code store.

**Program flash Sector** — The smallest portion of the program flash memory (consecutive addresses) that can be erased.

**Retention** — The length of time that data can be kept in the NVM without experiencing errors upon readout. Since erased (1) states are subject to degradation just like programmed (0) states, the data retention limit may be reached from the last erase operation (not from the programming time).

**RWW** — Read-While-Write. The ability to simultaneously read from one memory resource while commanded operations are active in another memory resource.

**Secure** — An MCU state conveyed to the flash memory module as described in the Chip Configuration details for this device. In the secure state, reading and changing NVM contents is restricted.

## **27.2 External Signal Description**

The flash memory module contains no signals that connect off-chip.

## **27.3 Memory Map and Registers**

This section describes the memory map and registers for the flash memory module. Data read from unimplemented memory space in the flash memory module is undefined. Writes to unimplemented or reserved memory space (registers) in the flash memory module are ignored.

## 27.3.1 Flash Configuration Field Description

The program flash memory contains a 16-byte flash configuration field that stores default protection settings (loaded on reset) and security information that allows the MCU to restrict access to the flash memory module.

| Flash Configuration Field Byte Address | Size (Bytes) | Field Description  |
|--|--------------|--|
| 0x0_0400 - 0x0_0407                    | 8            | Backdoor Comparison Key. Refer to <a href="#">Verify Backdoor Access Key Command</a> and <a href="#">Unsecuring the Chip Using Backdoor Key Access</a> . |
| 0x0_0408 - 0x0_040B                    | 4            | Program flash protection bytes. Refer to the description of the Program Flash Protection Registers (FPROT0-3).   |
| 0x0_040F                               | 1            | Reserved   |
| 0x0_040E                               | 1            | Reserved   |
| 0x0_040D                               | 1            | Flash nonvolatile option byte. Refer to the description of the Flash Option Register (FOPT).   |
| 0x0_040C                               | 1            | Flash security byte. Refer to the description of the Flash Security Register (FSEC).   |

## 27.3.2 Program Flash IFR Map

The program flash IFR is nonvolatile information memory that can be read freely, but the user has no erase and limited program capabilities (see the Read Once, Program Once, and Read Resource commands in [Read Once Command](#), [Program Once Command](#) and [Read Resource Command](#)). The contents of the program flash IFR are summarized in the following table and further described in the subsequent paragraphs.

The program flash IFR is located within the program flash 0 memory block.

| Address Range | Size (Bytes) | Field Description  |
|---------------|--------------|--------------------|
| 0x00 – 0xBF   | 192          | Reserved           |
| 0xC0 – 0xFF   | 64           | Program Once Field |

### 27.3.2.1 Program Once Field

The Program Once Field in the program flash IFR provides 64 bytes of user data storage separate from the program flash main array. The user can program the Program Once Field one time only as there is no program flash IFR erase mechanism available to the

user. The Program Once Field can be read any number of times. This section of the program flash IFR is accessed in 4-Byte records using the Read Once and Program Once commands (see [Read Once Command](#) and [Program Once Command](#)).

### 27.3.3 Register Descriptions

The flash memory module contains a set of memory-mapped control and status registers.

#### NOTE

While a command is running (FSTAT[CCIF]=0), register writes are not accepted to any register except FCNFG and FSTAT. The no-write rule is relaxed during the start-up reset sequence, prior to the initial rise of CCIF. During this initialization period the user may write any register. All register writes are also disabled (except for registers FCNFG and FSTAT) whenever an erase suspend request is active (FCNFG[ERSSUSP]=1).

**FTFA memory map**

| Absolute address (hex) | Register name                                       | Width (in bits) | Access | Reset value | Section/ page               |
|------------------------|---|-----------------|--------|-------------|-----------------------------|
| 4002_0000              | Flash Status Register (FTFA_FSTAT)                  | 8               | R/W    | 00h         | <a href="#">27.33.1/425</a> |
| 4002_0001              | Flash Configuration Register (FTFA_FCNFG)           | 8               | R/W    | 00h         | <a href="#">27.33.2/426</a> |
| 4002_0002              | Flash Security Register (FTFA_FSEC)                 | 8               | R      | Undefined   | <a href="#">27.33.3/428</a> |
| 4002_0003              | Flash Option Register (FTFA_FOPT)                   | 8               | R      | Undefined   | <a href="#">27.33.4/429</a> |
| 4002_0004              | Flash Common Command Object Registers (FTFA_FCCOB3) | 8               | R/W    | 00h         | <a href="#">27.33.5/430</a> |
| 4002_0005              | Flash Common Command Object Registers (FTFA_FCCOB2) | 8               | R/W    | 00h         | <a href="#">27.33.5/430</a> |
| 4002_0006              | Flash Common Command Object Registers (FTFA_FCCOB1) | 8               | R/W    | 00h         | <a href="#">27.33.5/430</a> |
| 4002_0007              | Flash Common Command Object Registers (FTFA_FCCOB0) | 8               | R/W    | 00h         | <a href="#">27.33.5/430</a> |
| 4002_0008              | Flash Common Command Object Registers (FTFA_FCCOB7) | 8               | R/W    | 00h         | <a href="#">27.33.5/430</a> |
| 4002_0009              | Flash Common Command Object Registers (FTFA_FCCOB6) | 8               | R/W    | 00h         | <a href="#">27.33.5/430</a> |
| 4002_000A              | Flash Common Command Object Registers (FTFA_FCCOB5) | 8               | R/W    | 00h         | <a href="#">27.33.5/430</a> |
| 4002_000B              | Flash Common Command Object Registers (FTFA_FCCOB4) | 8               | R/W    | 00h         | <a href="#">27.33.5/430</a> |
| 4002_000C              | Flash Common Command Object Registers (FTFA_FCCOB3) | 8               | R/W    | 00h         | <a href="#">27.33.5/430</a> |

*Table continues on the next page...*

**FTFA memory map (continued)**

| Absolute address (hex) | Register name                                       | Width (in bits) | Access | Reset value | Section/page                |
|------------------------|---|-----------------|--------|-------------|-----------------------------|
| 4002_000D              | Flash Common Command Object Registers (FTFA_FCCOBA) | 8               | R/W    | 00h         | <a href="#">27.33.5/430</a> |
| 4002_000E              | Flash Common Command Object Registers (FTFA_FCCOB9) | 8               | R/W    | 00h         | <a href="#">27.33.5/430</a> |
| 4002_000F              | Flash Common Command Object Registers (FTFA_FCCOB8) | 8               | R/W    | 00h         | <a href="#">27.33.5/430</a> |
| 4002_0010              | Program Flash Protection Registers (FTFA_FPROT3)    | 8               | R/W    | Undefined   | <a href="#">27.33.6/431</a> |
| 4002_0011              | Program Flash Protection Registers (FTFA_FPROT2)    | 8               | R/W    | Undefined   | <a href="#">27.33.6/431</a> |
| 4002_0012              | Program Flash Protection Registers (FTFA_FPROT1)    | 8               | R/W    | Undefined   | <a href="#">27.33.6/431</a> |
| 4002_0013              | Program Flash Protection Registers (FTFA_FPROT0)    | 8               | R/W    | Undefined   | <a href="#">27.33.6/431</a> |

**27.33.1 Flash Status Register (FTFA\_FSTAT)**

The FSTAT register reports the operational status of the flash memory module.

The CCIF, RDCOLERR, ACCERR, and FPVIOL bits are readable and writable. The MGSTAT0 bit is read only. The unassigned bits read 0 and are not writable.

**NOTE**

When set, the Access Error (ACCERR) and Flash Protection Violation (FPVIOL) bits in this register prevent the launch of any more commands until the flag is cleared (by writing a one to it).

Address: 4002\_0000h base + 0h offset = 4002\_0000h

| Bit   | 7    | 6        | 5      | 4      | 3 | 2 | 1 | 0       |
|-------|------|----------|--------|--------|---|---|---|---------|
| Read  | CCIF | RDCOLERR | ACCERR | FPVIOL |   | 0 |   | MGSTAT0 |
| Write | w1c  | w1c      | w1c    | w1c    |   |   |   |         |
| Reset | 0    | 0        | 0      | 0      | 0 | 0 | 0 | 0       |

**FTFA\_FSTAT field descriptions**

| Field     | Description  |
|-----------|--|
| 7<br>CCIF | <p>Command Complete Interrupt Flag</p> <p>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 stays low until command completion or command violation.</p> <p>The CCIF bit is reset to 0 but is set to 1 by the memory controller at the end of the reset initialization sequence. Depending on how quickly the read occurs after reset release, the user may or may not see the 0 hardware reset value.</p> |

*Table continues on the next page...*

**FTFA\_FSTAT field descriptions (continued)**

| Field        | Description   |
|--------------|---|
|              | 0 Flash command in progress<br>1 Flash command has completed  |
| 6 RDCOLERR   | <b>Flash Read Collision Error Flag</b><br><p>The RDCOLERR error bit indicates that the MCU attempted a read from a flash memory resource that was being manipulated by a flash command (CCIF=0). Any simultaneous access is detected as a collision error by the block arbitration logic. The read data in this case cannot be guaranteed. The RDCOLERR bit is cleared by writing a 1 to it. Writing a 0 to RDCOLERR has no effect.</p> 0 No collision error detected<br>1 Collision error detected   |
| 5 ACCERR     | <b>Flash Access Error Flag</b><br><p>The ACCERR error bit indicates an illegal access has occurred to a flash memory resource caused by a violation of the 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 it. Writing a 0 to the ACCERR bit has no effect.</p> 0 No access error detected<br>1 Access error detected   |
| 4 FPVIOL     | <b>Flash Protection Violation Flag</b><br><p>The FPVIOL error bit indicates an attempt was made to program or erase an address in a protected area of program flash memory during a command write sequence . While FPVIOL is set, the CCIF flag cannot be cleared to launch a command. The FPVIOL bit is cleared by writing a 1 to it. Writing a 0 to the FPVIOL bit has no effect.</p> 0 No protection violation detected<br>1 Protection violation detected   |
| 3–1 Reserved | This field is reserved.<br>This read-only field is reserved and always has the value 0.   |
| 0 MGSTAT0    | <b>Memory Controller Command Completion Status Flag</b><br><p>The MGSTAT0 status flag is set if an error is detected during execution of a flash command or during the flash reset sequence. As a status flag, this bit cannot (and need not) be cleared by the user like the other error flags in this register.</p> <p>The value of the MGSTAT0 bit for "command-N" is valid only at the end of the "command-N" execution when CCIF=1 and before the next command has been launched. At some point during the execution of "command-N+1," the previous result is discarded and any previous error is cleared.</p> |

**27.33.2 Flash Configuration Register (FTFA\_FCNFG)**

This register provides information on the current functional state of the flash memory module.

The erase control bits (ERSAREQ and ERSSUSP) have write restrictions. The unassigned bits read as noted and are not writable.

Address: 4002\_0000h base + 1h offset = 4002\_0001h

| Bit   | 7    | 6        | 5       | 4       | 3 | 2 | 1 | 0 |
|-------|------|----------|---------|---------|---|---|---|---|
| Read  | CCIE | RDCOLLIE | ERSAREQ | ERSSUSP | 0 | 0 | 0 | 0 |
| Write |      |          |         |         |   |   |   |   |
| Reset | 0    | 0        | 0       | 0       | 0 | 0 | 0 | 0 |

### FTFA\_FCNFG field descriptions

| Field           | Description   |
|-----------------|---|
| 7<br>CCIE       | <p>Command Complete Interrupt Enable</p> <p>The CCIE bit controls interrupt generation when a flash command completes.</p> <ul style="list-style-type: none"> <li>0 Command complete interrupt disabled</li> <li>1 Command complete interrupt enabled. An interrupt request is generated whenever the FSTAT[CCIF] flag is set.</li> </ul>   |
| 6<br>RDCOLLIE   | <p>Read Collision Error Interrupt Enable</p> <p>The RDCOLLIE bit controls interrupt generation when a flash memory read collision error occurs.</p> <ul style="list-style-type: none"> <li>0 Read collision error interrupt disabled</li> <li>1 Read collision error interrupt enabled. An interrupt request is generated whenever a flash memory read collision error is detected (see the description of FSTAT[RDCOLERR]).</li> </ul>   |
| 5<br>ERSAREQ    | <p>Erase All Request</p> <p>This bit issues a request to the memory controller to execute the Erase All Blocks command and release security. ERSAREQ is not directly writable but is under indirect user control. Refer to the device's Chip Configuration details on how to request this command.</p> <p>The ERSAREQ bit sets when an erase all request is triggered external to the flash memory module and CCIF is set (no command is currently being executed). ERSAREQ is cleared by the flash memory module when the operation completes.</p> <ul style="list-style-type: none"> <li>0 No request or request complete</li> <li>1 Request to:           <ol style="list-style-type: none"> <li>1. run the Erase All Blocks command,</li> <li>2. verify the erased state,</li> <li>3. program the security byte in the Flash Configuration Field to the unsecure state, and</li> <li>4. release MCU security by setting the FSEC[SEC] field to the unsecure state.</li> </ol> </li> </ul> |
| 4<br>ERSSUSP    | <p>Erase Suspend</p> <p>The ERSSUSP bit allows the user to suspend (interrupt) the Erase Flash Sector command while it is executing.</p> <ul style="list-style-type: none"> <li>0 No suspend requested</li> <li>1 Suspend the current Erase Flash Sector command execution.</li> </ul>  |
| 3<br>Reserved   | <p>This field is reserved.</p> <p>This read-only field is reserved and always has the value 0.</p>  |
| 2–0<br>Reserved | <p>This field is reserved.</p> <p>This read-only field is reserved and always has the value 0.</p>  |

### 27.33.3 Flash Security Register (FTFA\_FSEC)

This read-only register holds all bits associated with the security of the MCU and flash memory module.

During the reset sequence, the register is loaded with the contents of the flash security byte in the Flash Configuration Field located in program flash memory. The flash basis for the values is signified by X in the reset value.

Address: 4002\_0000h base + 2h offset = 4002\_0002h

| Bit   | 7     | 6  | 5    | 4  | 3      | 2  | 1   | 0  |
|-------|-------|----|------|----|--------|----|-----|----|
| Read  | KEYEN |    | MEEN |    | FSLACC |    | SEC |    |
| Write |       |    |      |    |        |    |     |    |
| Reset | x*    | x* | x*   | x* | x*     | x* | x*  | x* |

\* Notes:

- x = Undefined at reset.

#### FTFA\_FSEC field descriptions

| Field         | Description  |
|---------------|--|
| 7–6<br>KEYEN  | <p>Backdoor Key Security Enable</p> <p>These bits enable and disable backdoor key access to the flash memory module.</p> <p>00 Backdoor key access disabled<br/>     01 Backdoor key access disabled (preferred KEYEN state to disable backdoor key access)<br/>     10 Backdoor key access enabled<br/>     11 Backdoor key access disabled</p>   |
| 5–4<br>MEEN   | <p>Mass Erase Enable Bits</p> <p>Enables and disables mass erase capability of the flash memory module. The state of the MEEN bits is only relevant when the SEC bits are set to secure outside of NVM Normal Mode. When the SEC field is set to unsecure, the MEEN setting does not matter.</p> <p>00 Mass erase is enabled<br/>     01 Mass erase is enabled<br/>     10 Mass erase is disabled<br/>     11 Mass erase is enabled</p>  |
| 3–2<br>FSLACC | <p>Freescale Failure Analysis Access Code</p> <p>These bits enable or disable access to the flash memory contents during returned part failure analysis at Freescale. When SEC is secure and FSLACC is denied, access to the program flash contents is denied and any failure analysis performed by Freescale factory test must begin with a full erase to unsecure the part.</p> <p>When access is granted (SEC is unsecure, or SEC is secure and FSLACC is granted), Freescale factory testing has visibility of the current flash contents. The state of the FSLACC bits is only relevant when the SEC bits are set to secure. When the SEC field is set to unsecure, the FSLACC setting does not matter.</p> |

Table continues on the next page...

**FTFA\_FSEC field descriptions (continued)**

| Field   | Description  |
|---------|--|
|         | 00 Freescale factory access granted<br>01 Freescale factory access denied<br>10 Freescale factory access denied<br>11 Freescale factory access granted   |
| 1–0 SEC | <b>Flash Security</b><br>These bits define the security state of the MCU. In the secure state, the MCU limits access to flash memory module resources. The limitations are defined per device and are detailed in the Chip Configuration details. If the flash memory module is unsecured using backdoor key access, the SEC bits are forced to 10b.<br>00 MCU security status is secure<br>01 MCU security status is secure<br>10 MCU security status is unsecure (The standard shipping condition of the flash memory module is unsecure.)<br>11 MCU security status is secure |

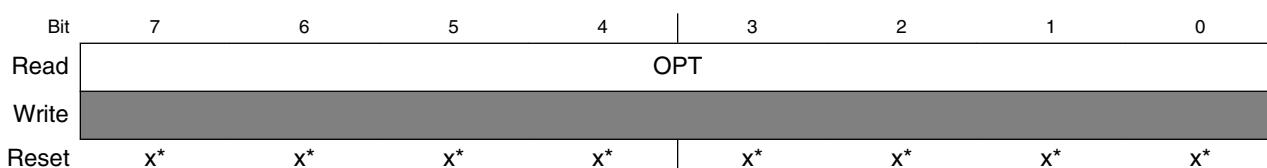
**27.33.4 Flash Option Register (FTFA\_FOPT)**

The flash option register allows the MCU to customize its operations by examining the state of these read-only bits, which are loaded from NVM at reset. The function of the bits is defined in the device's Chip Configuration details.

All bits in the register are read-only .

During the reset sequence, the register is loaded from the flash nonvolatile option byte in the Flash Configuration Field located in program flash memory. The flash basis for the values is signified by X in the reset value.

Address: 4002\_0000h base + 3h offset = 4002\_0003h



\* Notes:

- x = Undefined at reset.

**FTFA\_FOPT field descriptions**

| Field   | Description   |
|---------|---|
| 7–0 OPT | <b>Nonvolatile Option</b><br>These bits are loaded from flash to this register at reset. Refer to the device's Chip Configuration details for the definition and use of these bits. |

## 27.33.5 Flash Common Command Object Registers (FTFA\_FCCOBn)

The FCCOB register group provides 12 bytes for command codes and parameters. The individual bytes within the set append a 0-B hex identifier to the FCCOB register name: FCCOB0, FCCOB1, ..., FCCOBB.

Address: 4002\_0000h base + 4h offset + (1d × i), where i=0d to 11d

| Bit        | 7     | 6 | 5 | 4 |  | 3 | 2 | 1 | 0 |
|------------|-------|---|---|---|--|---|---|---|---|
| Read Write | CCOBn |   |   |   |  |   |   |   |   |
| Reset      | 0     | 0 | 0 | 0 |  | 0 | 0 | 0 | 0 |

### FTFA\_FCCOBn field descriptions

| Field        | Description   |              |  |   |  |   |                       |   |                      |   |                     |   |             |   |             |   |             |   |             |   |             |   |             |
|--------------|---|--------------|--|---|--|---|-----------------------|---|----------------------|---|---------------------|---|-------------|---|-------------|---|-------------|---|-------------|---|-------------|---|-------------|
| 7–0<br>CCOBn | <p>The FCCOB register provides a command code and relevant parameters to the memory controller. The individual registers that compose the FCCOB data set can be written in any order, but you must provide all needed values, which vary from command to command. First, set up all required FCCOB fields and then initiate the command's execution by writing a 1 to the FSTAT[CCIF] bit. This clears the CCIF bit, which locks all FCCOB parameter fields and they cannot be changed by the user until the command completes (CCIF returns to 1). No command buffering or queueing is provided; the next command can be loaded only after the current command completes.</p> <p>Some commands return information to the FCCOB registers. Any values returned to FCCOB are available for reading after the FSTAT[CCIF] flag returns to 1 by the memory controller.</p> <p>The following table shows a generic flash command format. The first FCCOB register, FCCOB0, always contains the command code. This 8-bit value defines the command to be executed. The command code is followed by the parameters required for this specific flash command, typically an address and/or data values.</p> <p><b>NOTE:</b> The command parameter table is written in terms of FCCOB Number (which is equivalent to the byte number). This number is a reference to the FCCOB register name and is not the register address.</p> <table border="1"> <thead> <tr> <th>FCCOB Number</th> <th>Typical Command Parameter Contents [7:0]</th> </tr> </thead> <tbody> <tr> <td>0</td> <td>FCMD (a code that defines the flash command)</td> </tr> <tr> <td>1</td> <td>Flash address [23:16]</td> </tr> <tr> <td>2</td> <td>Flash address [15:8]</td> </tr> <tr> <td>3</td> <td>Flash address [7:0]</td> </tr> <tr> <td>4</td> <td>Data Byte 0</td> </tr> <tr> <td>5</td> <td>Data Byte 1</td> </tr> <tr> <td>6</td> <td>Data Byte 2</td> </tr> <tr> <td>7</td> <td>Data Byte 3</td> </tr> <tr> <td>8</td> <td>Data Byte 4</td> </tr> <tr> <td>9</td> <td>Data Byte 5</td> </tr> </tbody> </table> | FCCOB Number | Typical Command Parameter Contents [7:0] | 0 | FCMD (a code that defines the flash command) | 1 | Flash address [23:16] | 2 | Flash address [15:8] | 3 | Flash address [7:0] | 4 | Data Byte 0 | 5 | Data Byte 1 | 6 | Data Byte 2 | 7 | Data Byte 3 | 8 | Data Byte 4 | 9 | Data Byte 5 |
| FCCOB Number | Typical Command Parameter Contents [7:0]  |              |  |   |  |   |                       |   |                      |   |                     |   |             |   |             |   |             |   |             |   |             |   |             |
| 0            | FCMD (a code that defines the flash command)  |              |  |   |  |   |                       |   |                      |   |                     |   |             |   |             |   |             |   |             |   |             |   |             |
| 1            | Flash address [23:16]   |              |  |   |  |   |                       |   |                      |   |                     |   |             |   |             |   |             |   |             |   |             |   |             |
| 2            | Flash address [15:8]  |              |  |   |  |   |                       |   |                      |   |                     |   |             |   |             |   |             |   |             |   |             |   |             |
| 3            | Flash address [7:0]   |              |  |   |  |   |                       |   |                      |   |                     |   |             |   |             |   |             |   |             |   |             |   |             |
| 4            | Data Byte 0   |              |  |   |  |   |                       |   |                      |   |                     |   |             |   |             |   |             |   |             |   |             |   |             |
| 5            | Data Byte 1   |              |  |   |  |   |                       |   |                      |   |                     |   |             |   |             |   |             |   |             |   |             |   |             |
| 6            | Data Byte 2   |              |  |   |  |   |                       |   |                      |   |                     |   |             |   |             |   |             |   |             |   |             |   |             |
| 7            | Data Byte 3   |              |  |   |  |   |                       |   |                      |   |                     |   |             |   |             |   |             |   |             |   |             |   |             |
| 8            | Data Byte 4   |              |  |   |  |   |                       |   |                      |   |                     |   |             |   |             |   |             |   |             |   |             |   |             |
| 9            | Data Byte 5   |              |  |   |  |   |                       |   |                      |   |                     |   |             |   |             |   |             |   |             |   |             |   |             |

**FTFA\_FCCOB $n$  field descriptions (continued)**

| Field   | Description  |  |
|---|--------------|--|
|   | FCCOB Number | Typical Command Parameter Contents [7:0] |
| A   |              | Data Byte 6                              |
| B   |              | Data Byte 7                              |
| <b>FCCOB Endianness and Multi-Byte Access :</b>   |              |  |
| The FCCOB register group uses a big endian addressing convention. For all command parameter fields larger than 1 byte, the most significant data resides in the lowest FCCOB register number. The FCCOB register group may be read and written as individual bytes, aligned words (2 bytes) or aligned longwords (4 bytes). |              |  |

**27.33.6 Program Flash Protection Registers (FTFA\_FPROT $n$ )**

The FPROT registers define which logical program flash regions are protected from program and erase operations. Protected flash regions cannot have their content changed; that is, these regions cannot be programmed and cannot be erased by any flash command. Unprotected regions can be changed by program and erase operations.

The four FPROT registers allow up to 32 protectable regions. Each bit protects a 1/32 region of the program flash memory except for memory configurations with less than 32 Kbytes of program flash where each assigned bit protects 1 Kbyte. For configurations with 24 Kbytes of program flash memory or less, FPROT0 is not used. For configurations with 16 Kbytes of program flash memory or less, FPROT1 is not used. For configurations with 8 Kbytes of program flash memory, FPROT2 is not used. The bitfields are defined in each register as follows:

| Program flash protection register | Program flash protection bits |
|-----------------------------------|-------------------------------|
| FPROT0                            | PROT[31:24]                   |
| FPROT1                            | PROT[23:16]                   |
| FPROT2                            | PROT[15:8]                    |
| FPROT3                            | PROT[7:0]                     |

During the reset sequence, the FPROT registers are loaded with the contents of the program flash protection bytes in the Flash Configuration Field as indicated in the following table.

| Program flash protection register | Flash Configuration Field offset address |
|-----------------------------------|--|
| FPROT0                            | 0x000B                                   |
| FPROT1                            | 0x000A                                   |

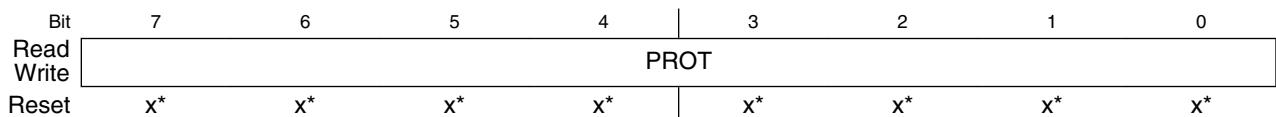
*Table continues on the next page...*

## Functional Description

| Program flash protection register | Flash Configuration Field offset address |
|-----------------------------------|--|
| FPROT2                            | 0x0009                                   |
| FPROT3                            | 0x0008                                   |

To change the program flash protection that is loaded during the reset sequence, unprotect the sector of program flash memory that contains the Flash Configuration Field. Then, reprogram the program flash protection byte.

Address: 4002\_0000h base + 10h offset + (1d × i), where i=0d to 3d



\* Notes:

- x = Undefined at reset.

### FTFA\_FPROTn field descriptions

| Field       | Description  |
|-------------|--|
| 7–0<br>PROT | <p>Program Flash Region Protect</p> <p>Each program flash region can be protected from program and erase operations by setting the associated PROT bit.</p> <p><b>In NVM Normal mode:</b> The protection can only be increased, meaning that currently unprotected memory can be protected, but currently protected memory cannot be unprotected. Since unprotected regions are marked with a 1 and protected regions use a 0, only writes changing 1s to 0s are accepted. This 1-to-0 transition check is performed on a bit-by-bit basis. Those FPROT bits with 1-to-0 transitions are accepted while all bits with 0-to-1 transitions are ignored.</p> <p><b>In NVM Special mode:</b> All bits of FPROT are writable without restriction. Unprotected areas can be protected and protected areas can be unprotected.</p> <p><b>Restriction:</b> The user must never write to any FPROT register while a command is running (CCIF=0). Trying to alter data in any protected area in the program flash memory results in a protection violation error and sets the FSTAT[FPVIOL] bit. A full block erase of a program flash block is not possible if it contains any protected region.</p> <p>Each bit in the 32-bit protection register represents 1/32 of the total program flash except for configurations where program flash memory is less than 32 Kbytes. For configurations with less than 32 Kbytes of program flash memory, each assigned bit represents 1 Kbyte.</p> <p>0 Program flash region is protected.<br/>1 Program flash region is not protected</p> |

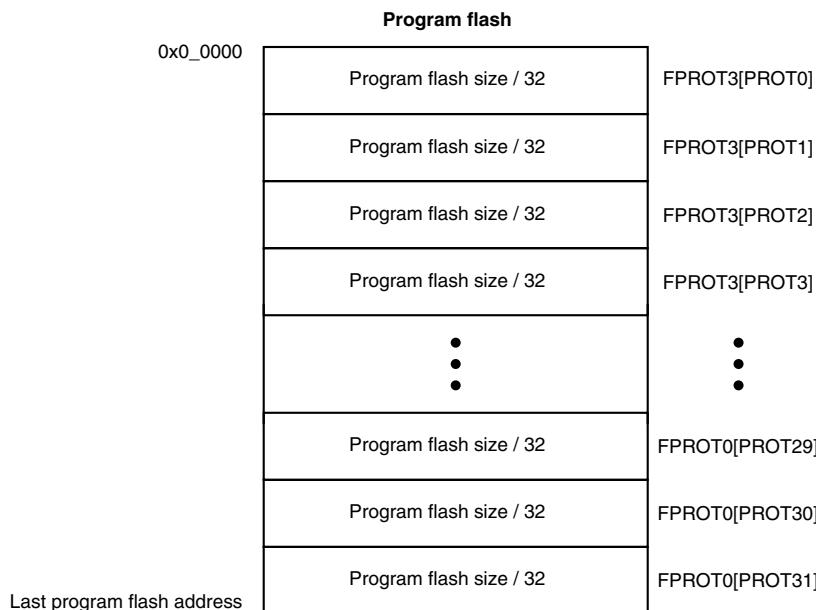
## 27.4 Functional Description

The following sections describe functional details of the flash memory module.

## 27.4.1 Flash Protection

Individual regions within the flash memory can be protected from program and erase operations. Protection is controlled by the following registers:

- **FPROT $n$**  — Four registers that protect 32 regions of the program flash memory as shown in the following figure



**Figure 27-24. Program flash protection**

## 27.4.2 Interrupts

The flash memory module can generate interrupt requests to the MCU upon the occurrence of various flash events. These interrupt events and their associated status and control bits are shown in the following table.

**Table 27-24. Flash Interrupt Sources**

| Flash Event                | Readable Status Bit | Interrupt Enable Bit |
|----------------------------|---------------------|----------------------|
| Flash Command Complete     | FSTAT[CCIF]         | FCNFG[CCIE]          |
| Flash Read Collision Error | FSTAT[RDCOLERR]     | FCNFG[RDCOLLIE]      |

**Note**

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

Some devices also generate a bus error response as a result of a Read Collision Error event. See the chip configuration information to determine if a bus error response is also supported.

## 27.4.3 Flash Operation in Low-Power Modes

### 27.4.3.1 Wait Mode

When the MCU enters wait mode, the flash memory module is not affected. The flash memory module can recover the MCU from wait via the command complete interrupt (see [Interrupts](#)).

### 27.4.3.2 Stop Mode

When the MCU requests stop mode, if a flash command is active ( $CCIF = 0$ ) the command execution completes before the MCU is allowed to enter stop mode.

**CAUTION**

The MCU should never enter stop mode while any flash command is running ( $CCIF = 0$ ).

**NOTE**

While the MCU is in very-low-power modes (VLPR, VLPW, VLPS), the flash memory module does not accept flash commands.

## 27.4.4 Functional Modes of Operation

The flash memory module has two operating modes: NVM Normal and NVM Special. The operating mode affects the command set availability (see [Table 27-25](#)). Refer to the Chip Configuration details of this device for how to activate each mode.

## 27.4.5 Flash Reads and Ignored Writes

The flash memory module requires only the flash address to execute a flash memory read.

The MCU must not read from the flash memory while commands are running (as evidenced by CCIF=0) on that block. Read data cannot be guaranteed from a flash block while any command is processing within that block. The block arbitration logic detects any simultaneous access and reports this as a read collision error (see the FSTAT[RDCOLERR] bit).

## 27.4.6 Read While Write (RWW)

The following simultaneous accesses are not allowed:

- Reading from program flash memory space while a flash command is active (CCIF=0).

## 27.4.7 Flash Program and Erase

All flash functions except read require the user to setup and launch a flash command through a series of peripheral bus writes. The user cannot initiate any further flash commands until notified that the current command has completed. The flash command structure and operation are detailed in [Flash Command Operations](#).

## 27.4.8 Flash Command Operations

Flash command operations are typically used to modify flash memory contents. The next sections describe:

- The command write sequence used to set flash command parameters and launch execution
- A description of all flash commands available

## 27.4.8.1 Command Write Sequence

Flash commands are specified using a command write sequence illustrated in [Figure 27-25](#). The flash memory module performs various checks on the command (FCCOB) content and continues with command execution if all requirements are fulfilled.

Before launching a command, the ACCERR and FPVIOL bits in the FSTAT register must be zero and the CCIF flag must read 1 to verify that any previous command has completed. If CCIF is zero, the previous command execution is still active, a new command write sequence cannot be started, and all writes to the FCCOB registers are ignored.

### 27.4.8.1.1 Load the FCCOB Registers

The user must load the FCCOB registers with all parameters required by the desired flash command. The individual registers that make up the FCCOB data set can be written in any order.

### 27.4.8.1.2 Launch the Command by Clearing CCIF

Once all relevant command parameters have been loaded, the user launches the command by clearing the FSTAT[CCIF] bit by writing a '1' to it. The CCIF flag remains zero until the flash command completes.

The FSTAT register contains a blocking mechanism that prevents a new command from launching (can't clear CCIF) if the previous command resulted in an access error (FSTAT[ACCERR]=1) or a protection violation (FSTAT[FPVIOL]=1). In error scenarios, two writes to FSTAT are required to initiate the next command: the first write clears the error flags, the second write clears CCIF.

### 27.4.8.1.3 Command Execution and Error Reporting

The command processing has several steps:

1. The flash memory module reads the command code and performs a series of parameter checks and protection checks, if applicable, which are unique to each command.

If the parameter check fails, the FSTAT[ACCERR] (access error) flag is set. ACCERR reports invalid instruction codes and out-of bounds addresses. Usually, access errors suggest that the command was not set-up with valid parameters in the FCCOB register group.

Program and erase commands also check the address to determine if the operation is requested to execute on protected areas. If the protection check fails, the FSTAT[FPVIOL] (protection error) flag is set.

Command processing never proceeds to execution when the parameter or protection step fails. Instead, command processing is terminated after setting the FSTAT[CCIF] bit.

2. If the parameter and protection checks pass, the command proceeds to execution. Run-time errors, such as failure to erase verify, may occur during the execution phase. Run-time errors are reported in the FSTAT[MGSTAT0] bit. A command may have access errors, protection errors, and run-time errors, but the run-time errors are not seen until all access and protection errors have been corrected.
3. Command execution results, if applicable, are reported back to the user via the FCCOB and FSTAT registers.
4. The flash memory module sets the FSTAT[CCIF] bit signifying that the command has completed.

The flow for a generic command write sequence is illustrated in the following figure.

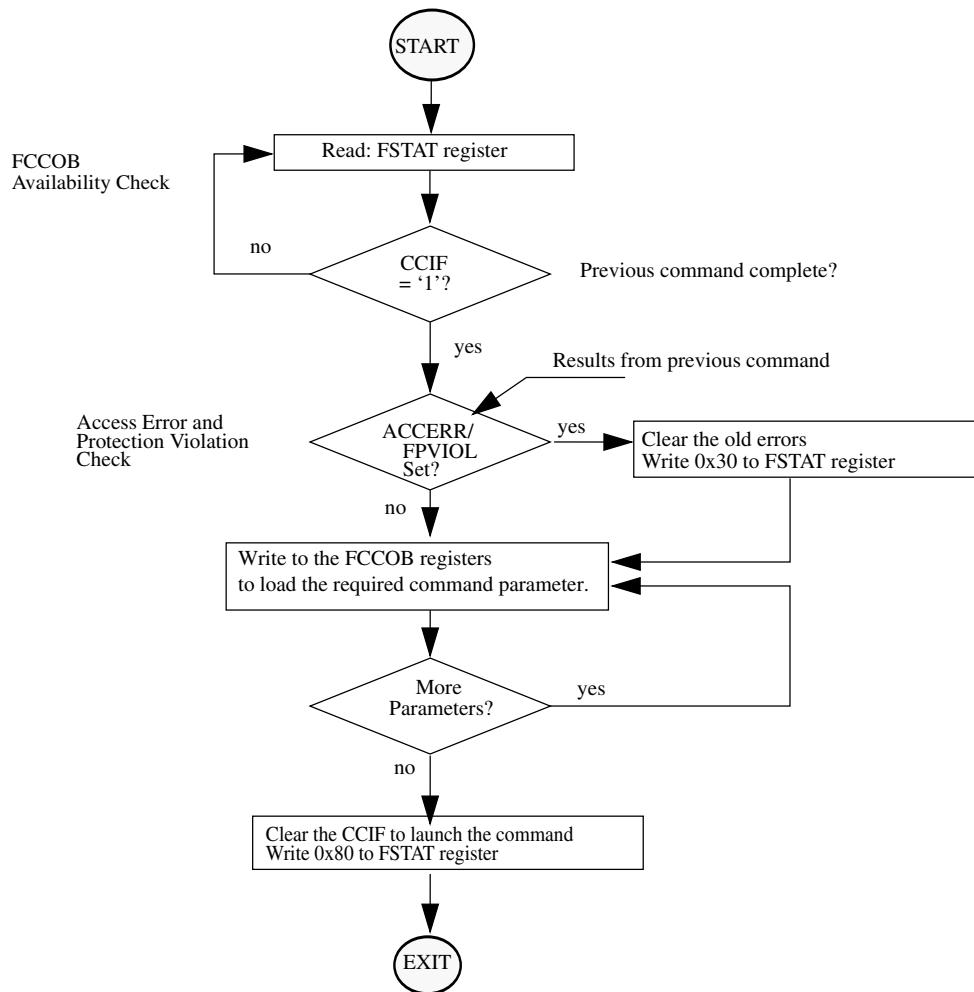


Figure 27-25. Generic Flash Command Write Sequence Flowchart

### 27.4.8.2 Flash Commands

The following table summarizes the function of all flash commands.

| FCMD | Command          | Program flash | Function  |
|------|------------------|---------------|---|
| 0x01 | Read 1s Section  | x             | Verify that a given number of program flash locations from a starting address are erased. |
| 0x02 | Program Check    | x             | Tests previously-programmed locations at margin read levels.                              |
| 0x03 | Read Resource    | IFR, ID       | Read 4 bytes from program flash IFR or version ID.  |
| 0x06 | Program Longword | x             | Program 4 bytes in a program flash block.   |

Table continues on the next page...

| FCMD | Command                    | Program flash | Function   |
|------|----------------------------|---------------|--|
| 0x09 | Erase Flash Sector         | ×             | Erase all bytes in a program flash sector.   |
| 0x40 | Read 1s All Blocks         | ×             | Verify that the program flash block is erased then release MCU security.   |
| 0x41 | Read Once                  | IFR           | Read 4 bytes of a dedicated 64 byte field in the program flash 0 IFR.  |
| 0x43 | Program Once               | IFR           | One-time program of 4 bytes of a dedicated 64-byte field in the program flash 0 IFR.   |
| 0x44 | Erase All Blocks           | ×             | Erase the program flash block, verify-erase and release MCU security.<br><br><b>NOTE:</b> An erase is only possible when all memory locations are unprotected. |
| 0x45 | Verify Backdoor Access Key | ×             | Release MCU security after comparing a set of user-supplied security keys to those stored in the program flash.  |

### 27.4.8.3 Flash Commands by Mode

The following table shows the flash commands that can be executed in each flash operating mode.

Table 27-25. Flash Commands by Mode

| FCMD | Command                    | NVM Normal |        |         | NVM Special |        |         |
|------|----------------------------|------------|--------|---------|-------------|--------|---------|
|      |                            | Unsecure   | Secure | MEEN=10 | Unsecure    | Secure | MEEN=10 |
| 0x01 | Read 1s Section            | ×          | ×      | ×       | ×           | —      | —       |
| 0x02 | Program Check              | ×          | ×      | ×       | ×           | —      | —       |
| 0x03 | Read Resource              | ×          | ×      | ×       | ×           | —      | —       |
| 0x06 | Program Longword           | ×          | ×      | ×       | ×           | —      | —       |
| 0x09 | Erase Flash Sector         | ×          | ×      | ×       | ×           | —      | —       |
| 0x40 | Read 1s All Blocks         | ×          | ×      | ×       | ×           | ×      | —       |
| 0x41 | Read Once                  | ×          | ×      | ×       | ×           | —      | —       |
| 0x43 | Program Once               | ×          | ×      | ×       | ×           | —      | —       |
| 0x44 | Erase All Blocks           | ×          | ×      | ×       | ×           | ×      | —       |
| 0x45 | Verify Backdoor Access Key | ×          | ×      | ×       | ×           | —      | —       |

## 27.4.9 Margin Read Commands

The Read-1s commands (Read 1s All Blocks and Read 1s Section) and the Program Check command have a margin choice parameter that allows the user to apply non-standard read reference levels to the program flash array reads performed by these commands. Using the preset 'user' and 'factory' margin levels, these commands perform their associated read operations at tighter tolerances than a 'normal' read. These non-standard read levels are applied only during the command execution. All simple (uncommanded) flash array reads to the MCU always use the standard, un-margined, read reference level.

Only the 'normal' read level should be employed during normal flash usage. The non-standard, 'user' and 'factory' margin levels should be employed only in special cases. They can be used during special diagnostic routines to gain confidence that the device is not suffering from the end-of-life data loss customary of flash memory devices.

Erased ('1') and programmed ('0') bit states can degrade due to elapsed time and data cycling (number of times a bit is erased and re-programmed). The lifetime of the erased states is relative to the last erase operation. The lifetime of the programmed states is measured from the last program time.

The 'user' and 'factory' levels become, in effect, a minimum safety margin; i.e. if the reads pass at the tighter tolerances of the 'user' and 'factory' margins, then the 'normal' reads have at least this much safety margin before they experience data loss.

The 'user' margin is a small delta to the normal read reference level. 'User' margin levels can be employed to check that flash memory contents have adequate margin for normal level read operations. If unexpected read results are encountered when checking flash memory contents at the 'user' margin levels, loss of information might soon occur during 'normal' readout.

The 'factory' margin is a bigger deviation from the norm, a more stringent read criteria that should only be attempted immediately (or very soon) after completion of an erase or program command, early in the cycling life. 'Factory' margin levels can be used to check that flash memory contents have adequate margin for long-term data retention at the normal level setting. If unexpected results are encountered when checking flash memory contents at 'factory' margin levels, the flash memory contents should be erased and reprogrammed.

### CAUTION

Factory margin levels must only be used during verify of the initial factory programming.

## 27.4.10 Flash Command Description

This section describes all flash commands that can be launched by a command write sequence. The flash memory module sets the FSTAT[ACCERR] bit and aborts the command execution if any of the following illegal conditions occur:

- There is an unrecognized command code in the FCCOB FCMD field.
- There is an error in a FCCOB field for the specific commands. Refer to the error handling table provided for each command.

Ensure that the ACCERR and FPVIOL bits in the FSTAT register are cleared prior to starting the command write sequence. As described in [Launch the Command by Clearing CCIF](#), a new command cannot be launched while these error flags are set.

Do not attempt to read a flash block while the flash memory module is running a command (CCIF = 0) on that same block. The flash memory module may return invalid data to the MCU with the collision error flag (FSTAT[RDCOLERR]) set.

### CAUTION

Flash data must be in the erased state before being programmed. Cumulative programming of bits (adding more zeros) is not allowed.

### 27.4.10.1 Read 1s Section Command

The Read 1s Section command checks if a section of program flash memory is erased to the specified read margin level. The Read 1s Section command defines the starting address and the number of longwords to be verified.

**Table 27-26. Read 1s Section Command FCCOB Requirements**

| FCCOB Number | FCCOB Contents [7:0]  |
|--------------|---|
| 0            | 0x01 (RD1SEC)   |
| 1            | Flash address [23:16] of the first longword to be verified            |
| 2            | Flash address [15:8] of the first longword to be verified             |
| 3            | Flash address [7:0] <sup>1</sup> of the first longword to be verified |
| 4            | Number of longwords to be verified [15:8]                             |
| 5            | Number of longwords to be verified [7:0]                              |
| 6            | Read-1 Margin Choice  |

1. Must be longword aligned (Flash address [1:0] = 00).

## Functional Description

Upon clearing CCIF to launch the Read 1s Section command, the flash memory module sets the read margin for 1s according to [Table 27-27](#) and then reads all locations within the specified section of flash memory. If the flash memory module fails to read all 1s (i.e. the flash section is not erased), the FSTAT[MGSTAT0] bit is set. The CCIF flag sets after the Read 1s Section operation completes.

**Table 27-27. Margin Level Choices for Read 1s Section**

| Read Margin Choice | Margin Level Description                              |
|--------------------|---|
| 0x00               | Use the 'normal' read level for 1s                    |
| 0x01               | Apply the 'User' margin to the normal read-1 level    |
| 0x02               | Apply the 'Factory' margin to the normal read-1 level |

**Table 27-28. Read 1s Section Command Error Handling**

| Error Condition                                      | Error Bit      |
|--|----------------|
| Command not available in current mode/security       | FSTAT[ACCERR]  |
| An invalid margin code is supplied                   | FSTAT[ACCERR]  |
| An invalid flash address is supplied                 | FSTAT[ACCERR]  |
| Flash address is not longword aligned                | FSTAT[ACCERR]  |
| The requested section crosses a Flash block boundary | FSTAT[ACCERR]  |
| The requested number of longwords is zero            | FSTAT[ACCERR]  |
| Read-1s fails  | FSTAT[MGSTAT0] |

### 27.4.10.2 Program Check Command

The Program Check command tests a previously programmed program flash longword to see if it reads correctly at the specified margin level.

**Table 27-29. Program Check Command FCCOB Requirements**

| FCCOB Number | FCCOB Contents [7:0]             |
|--------------|----------------------------------|
| 0            | 0x02 (PGMCHK)                    |
| 1            | Flash address [23:16]            |
| 2            | Flash address [15:8]             |
| 3            | Flash address [7:0] <sup>1</sup> |
| 4            | Margin Choice                    |
| 8            | Byte 0 expected data             |
| 9            | Byte 1 expected data             |
| A            | Byte 2 expected data             |
| B            | Byte 3 expected data             |

1. Must be longword aligned (Flash address [1:0] = 00).

Upon clearing CCIF to launch the Program Check command, the flash memory module sets the read margin for 1s according to [Table 27-30](#), reads the specified longword, and compares the actual read data to the expected data provided by the FCCOB. If the comparison at margin-1 fails, the FSTAT[MGSTAT0] bit is set.

The flash memory module then sets the read margin for 0s, re-reads, and compares again. If the comparison at margin-0 fails, the FSTAT[MGSTAT0] bit is set. The CCIF flag is set after the Program Check operation completes.

The supplied address must be longword aligned (the lowest two bits of the byte address must be 00):

- Byte 3 data is written to the supplied byte address ('start'),
- Byte 2 data is programmed to byte address start+0b01,
- Byte 1 data is programmed to byte address start+0b10,
- Byte 0 data is programmed to byte address start+0b11.

#### **NOTE**

See the description of margin reads, [Margin Read Commands](#)

**Table 27-30. Margin Level Choices for Program Check**

| Read Margin Choice | Margin Level Description                          |
|--------------------|---|
| 0x01               | Read at 'User' margin-1 and 'User' margin-0       |
| 0x02               | Read at 'Factory' margin-1 and 'Factory' margin-0 |

**Table 27-31. Program Check Command Error Handling**

| Error Condition   | Error Bit      |
|---|----------------|
| Command not available in current mode/security              | FSTAT[ACCERR]  |
| An invalid flash address is supplied                        | FSTAT[ACCERR]  |
| Flash address is not longword aligned                       | FSTAT[ACCERR]  |
| An invalid margin choice is supplied                        | FSTAT[ACCERR]  |
| Either of the margin reads does not match the expected data | FSTAT[MGSTAT0] |

### **27.4.10.3 Read Resource Command**

The Read Resource command allows the user to read data from special-purpose memory resources located within the flash memory module. The special-purpose memory resources available include program flash IFR space and the Version ID field. Each resource is assigned a select code as shown in [Table 27-33](#).

**Table 27-32. Read Resource Command FCCOB Requirements**

| FCCOB Number         | FCCOB Contents [7:0]                                    |
|----------------------|---|
| 0                    | 0x03 (RDRSRC)   |
| 1                    | Flash address [23:16]                                   |
| 2                    | Flash address [15:8]                                    |
| 3                    | Flash address [7:0] <sup>1</sup>                        |
| Returned Values      |   |
| 4                    | Read Data [31:24]                                       |
| 5                    | Read Data [23:16]                                       |
| 6                    | Read Data [15:8]  |
| 7                    | Read Data [7:0]   |
| User-provided values |   |
| 8                    | Resource Select Code (see <a href="#">Table 27-33</a> ) |

1. Must be longword aligned (Flash address [1:0] = 00).

**Table 27-33. Read Resource Select Codes**

| Resource Select Code | Description         | Resource Size | Local Address Range   |
|----------------------|---------------------|---------------|-----------------------|
| 0x00                 | Program Flash 0 IFR | 256 Bytes     | 0x00_0000 - 0x00_00FF |
| 0x01 <sup>1</sup>    | Version ID          | 8 Bytes       | 0x00_0000 - 0x00_0007 |

1. Located in program flash 0 reserved space.

## CAUTION

A flash memory location must be in the erased state before being programmed. Cumulative programming of bits (back-to-back program operations without an intervening erase) within a flash memory location is not allowed. Re-programming of existing 0s to 0 is not allowed as this overstresses the device.

**Table 27-35. Program Longword Command FCCOB Requirements**

| FCCOB Number | FCCOB Contents [7:0]             |
|--------------|----------------------------------|
| 0            | 0x06 (PGM4)                      |
| 1            | Flash address [23:16]            |
| 2            | Flash address [15:8]             |
| 3            | Flash address [7:0] <sup>1</sup> |
| 4            | Byte 0 program value             |
| 5            | Byte 1 program value             |
| 6            | Byte 2 program value             |
| 7            | Byte 3 program value             |

1. Must be longword aligned (Flash address [1:0] = 00).

Upon clearing CCIF to launch the Program Longword command, the flash memory module programs the data bytes into the flash using the supplied address. The targeted flash locations must be currently unprotected (see the description of the FPROT registers) to permit execution of the Program Longword operation.

The programming operation is unidirectional. It can only move NVM bits from the erased state ('1') to the programmed state ('0'). Erased bits that fail to program to the '0' state are flagged as errors in FSTAT[MGSTAT0]. The CCIF flag is set after the Program Longword operation completes.

The supplied address must be longword aligned (flash address [1:0] = 00):

- Byte 3 data is written to the supplied byte address ('start'),
- Byte 2 data is programmed to byte address start+0b01,
- Byte 1 data is programmed to byte address start+0b10, and
- Byte 0 data is programmed to byte address start+0b11.

**Table 27-36. Program Longword Command Error Handling**

| Error Condition                                | Error Bit     |
|--|---------------|
| Command not available in current mode/security | FSTAT[ACCERR] |
| An invalid flash address is supplied           | FSTAT[ACCERR] |
| Flash address is not longword aligned          | FSTAT[ACCERR] |
| Flash address points to a protected area       | FSTAT[FPVIOL] |

*Table continues on the next page...*

**Table 27-36. Program Longword Command Error Handling (continued)**

| Error Condition  | Error Bit      |
|--|----------------|
| Any errors have been encountered during the verify operation | FSTAT[MGSTAT0] |

### 27.4.10.5 Erase Flash Sector Command

The Erase Flash Sector operation erases all addresses in a flash sector.

**Table 27-37. Erase Flash Sector Command FCCOB Requirements**

| FCCOB Number | FCCOB Contents [7:0]  |
|--------------|---|
| 0            | 0x09 (ERSSCR)   |
| 1            | Flash address [23:16] in the flash sector to be erased            |
| 2            | Flash address [15:8] in the flash sector to be erased             |
| 3            | Flash address [7:0] <sup>1</sup> in the flash sector to be erased |

1. Must be longword aligned (flash address [1:0] = 00).

After clearing CCIF to launch the Erase Flash Sector command, the flash memory module erases the selected program flash sector and then verifies that it is erased. The Erase Flash Sector command aborts if the selected sector is protected (see the description of the FPROT registers). If the erase-verify fails the FSTAT[MGSTAT0] bit is set. The CCIF flag is set after the Erase Flash Sector operation completes. The Erase Flash Sector command is suspendable (see the FCNFG[ERSSUSP] bit and [Figure 27-26](#)).

**Table 27-38. Erase Flash Sector Command Error Handling**

| Error Condition  | Error Bit      |
|--|----------------|
| Command not available in current mode/security               | FSTAT[ACCERR]  |
| An invalid Flash address is supplied                         | FSTAT[ACCERR]  |
| Flash address is not longword aligned                        | FSTAT[ACCERR]  |
| The selected program flash sector is protected               | FSTAT[FPIOL]   |
| Any errors have been encountered during the verify operation | FSTAT[MGSTAT0] |

#### 27.4.10.5.1 Suspending an Erase Flash Sector Operation

To suspend an Erase Flash Sector operation set the FCNFG[ERSSUSP] bit (see [Flash Configuration Field Description](#)) when CCIF is clear and the CCOB command field holds the code for the Erase Flash Sector command. During the Erase Flash Sector operation (see [Erase Flash Sector Command](#)), the flash memory module samples the state of the ERSSUSP bit at convenient points. If the flash memory module detects that the

ERSSUSP bit is set, the Erase Flash Sector operation is suspended and the flash memory module sets CCIF. While ERSSUSP is set, all writes to flash registers are ignored except for writes to the FSTAT and FCNFG registers.

If an Erase Flash Sector operation effectively completes before the flash memory module detects that a suspend request has been made, the flash memory module clears the ERSSUSP bit prior to setting CCIF. When an Erase Flash Sector operation has been successfully suspended, the flash memory module sets CCIF and leaves the ERSSUSP bit set. While CCIF is set, the ERSSUSP bit can only be cleared to prevent the withdrawal of a suspend request before the flash memory module has acknowledged it.

#### **27.4.10.5.2 Resuming a Suspended Erase Flash Sector Operation**

If the ERSSUSP bit is still set when CCIF is cleared to launch the next command, the previous Erase Flash Sector operation resumes. The flash memory module acknowledges the request to resume a suspended operation by clearing the ERSSUSP bit. A new suspend request can then be made by setting ERSSUSP. A single Erase Flash Sector operation can be suspended and resumed multiple times.

There is a minimum elapsed time limit between the request to resume the Erase Flash Sector operation (CCIF is cleared) and the request to suspend the operation again (ERSSUSP is set). This minimum time period is required to ensure that the Erase Flash Sector operation will eventually complete. If the minimum period is continually violated, i.e. the suspend requests come repeatedly and too quickly, no forward progress is made by the Erase Flash Sector algorithm. The resume/suspend sequence runs indefinitely without completing the erase.

#### **27.4.10.5.3 Aborting a Suspended Erase Flash Sector Operation**

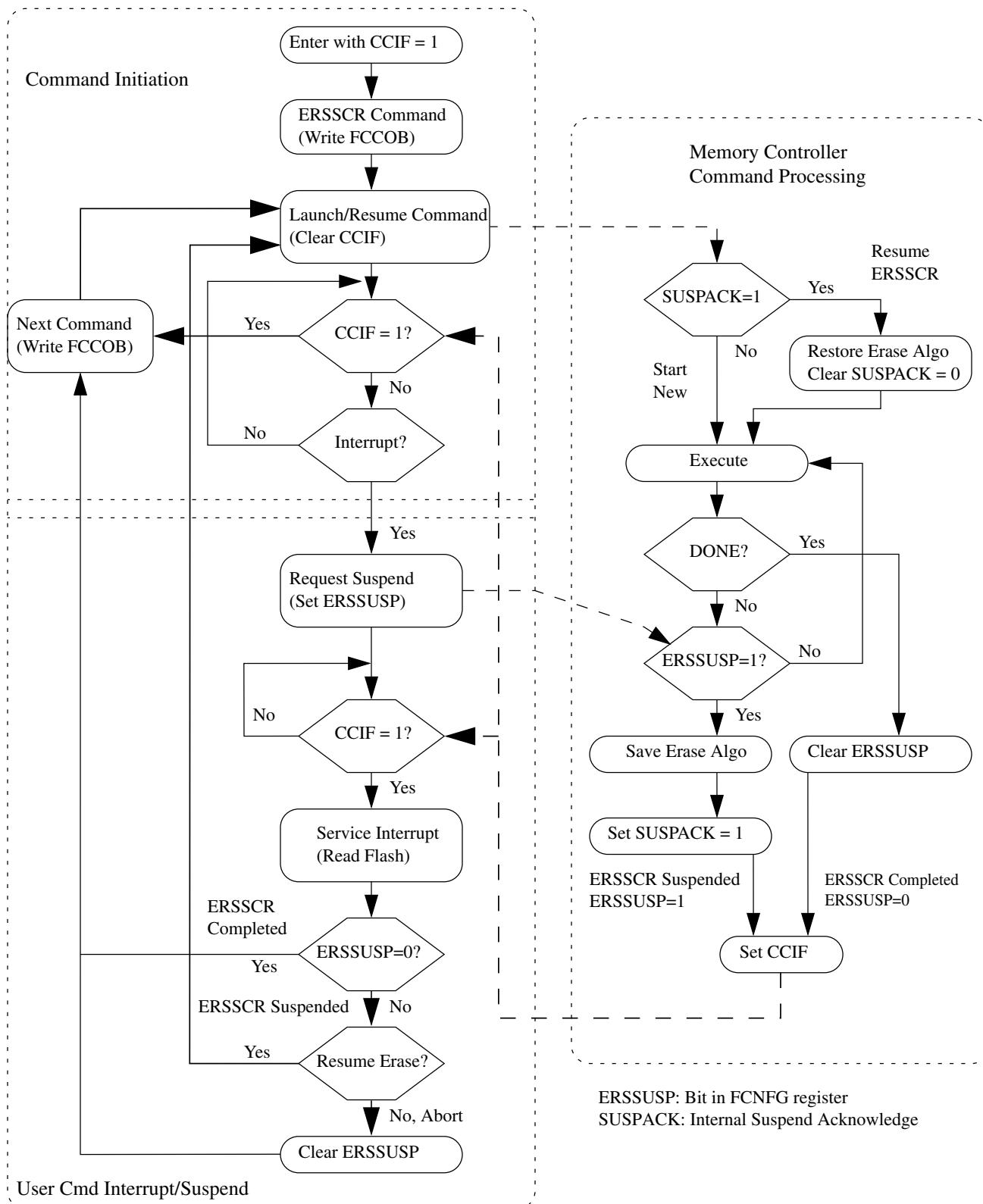
The user may choose to abort a suspended Erase Flash Sector operation by clearing the ERSSUSP bit prior to clearing CCIF for the next command launch. When a suspended operation is aborted, the flash memory module starts the new command using the new FCCOB contents.

##### **Note**

Aborting the erase leaves the bitcells in an indeterminate, partially-erased state. Data in this sector is not reliable until a new erase command fully completes.

The following figure shows how to suspend and resume the Erase Flash Sector operation.

## Functional Description



**Figure 27-26. Suspend and Resume of Erase Flash Sector Operation**

### 27.4.10.6 Read 1s All Blocks Command

The Read 1s All Blocks command checks if the program flash blocks have been erased to the specified read margin level, if applicable, and releases security if the readout passes, i.e. all data reads as '1'.

**Table 27-39. Read 1s All Blocks Command FCCOB Requirements**

| FCCOB Number | FCCOB Contents [7:0] |
|--------------|----------------------|
| 0            | 0x40 (RD1ALL)        |
| 1            | Read-1 Margin Choice |

After clearing CCIF to launch the Read 1s All Blocks command, the flash memory module :

- sets the read margin for 1s according to [Table 27-40](#),
- checks the contents of the program flash are in the erased state.

If the flash memory module confirms that these memory resources are erased, security is released by setting the FSEC[SEC] field to the unsecure state. The security byte in the flash configuration field (see [Flash Configuration Field Description](#)) remains unaffected by the Read 1s All Blocks command. If the read fails, i.e. all memory resources are not in the fully erased state, the FSTAT[MGSTAT0] bit is set.

The CCIF flag sets after the Read 1s All Blocks operation has completed.

**Table 27-40. Margin Level Choices for Read 1s All Blocks**

| Read Margin Choice | Margin Level Description                              |
|--------------------|---|
| 0x00               | Use the 'normal' read level for 1s                    |
| 0x01               | Apply the 'User' margin to the normal read-1 level    |
| 0x02               | Apply the 'Factory' margin to the normal read-1 level |

**Table 27-41. Read 1s All Blocks Command Error Handling**

| Error Condition                       | Error Bit      |
|---------------------------------------|----------------|
| An invalid margin choice is specified | FSTAT[ACCERR]  |
| Read-1s fails                         | FSTAT[MGSTAT0] |

### 27.4.10.7 Read Once Command

The Read Once command provides read access to a reserved 64-byte field located in the program flash 0 IFR (see [Program Flash IFR Map](#) and [Program Once Field](#)). Access to this field is via 16 records, each 4 bytes long. The Read Once field is programmed using the Program Once command described in [Program Once Command](#).

**Table 27-42. Read Once Command FCCOB Requirements**

| FCCOB Number    | FCCOB Contents [7:0]                 |
|-----------------|--------------------------------------|
| 0               | 0x41 (RDONCE)                        |
| 1               | Read Once record index (0x00 - 0x0F) |
| 2               | Not used                             |
| 3               | Not used                             |
| Returned Values |                                      |
| 4               | Read Once byte 0 value               |
| 5               | Read Once byte 1 value               |
| 6               | Read Once byte 2 value               |
| 7               | Read Once byte 3 value               |

After clearing CCIF to launch the Read Once command, a 4-byte Read Once record is read from the program flash IFR and stored in the FCCOB register. The CCIF flag is set after the Read Once operation completes. Valid record index values for the Read Once command range from 0x00 to 0x0F. During execution of the Read Once command, any attempt to read addresses within the program flash block containing this 64-byte field returns invalid data. The Read Once command can be executed any number of times.

**Table 27-43. Read Once Command Error Handling**

| Error Condition                                | Error Bit     |
|--|---------------|
| Command not available in current mode/security | FSTAT[ACCERR] |
| An invalid record index is supplied            | FSTAT[ACCERR] |

### 27.4.10.8 Program Once Command

The Program Once command enables programming to a reserved 64-byte field in the program flash 0 IFR (see [Program Flash IFR Map](#) and [Program Once Field](#)). Access to the Program Once field is via 16 records, each 4 bytes long. The Program Once field can be read using the Read Once command (see [Read Once Command](#)) or using the Read Resource command (see [Read Resource Command](#)). Each Program Once record can be programmed only once since the program flash 0 IFR cannot be erased.

**Table 27-44. Program Once Command FCCOB Requirements**

| FCCOB Number | FCCOB Contents [7:0]                    |
|--------------|---|
| 0            | 0x43 (PGMONCE)                          |
| 1            | Program Once record index (0x00 - 0x0F) |
| 2            | Not Used                                |
| 3            | Not Used                                |
| 4            | Program Once Byte 0 value               |
| 5            | Program Once Byte 1 value               |
| 6            | Program Once Byte 2 value               |
| 7            | Program Once Byte 3 value               |

After clearing CCIF to launch the Program Once command, the flash memory module first verifies that the selected record is erased. If erased, then the selected record is programmed using the values provided. The Program Once command also verifies that the programmed values read back correctly. The CCIF flag is set after the Program Once operation has completed.

The reserved program flash 0 IFR location accessed by the Program Once command cannot be erased and any attempt to program one of these records when the existing value is not Fs (erased) is not allowed. Valid record index values for the Program Once command range from 0x00 to 0x0F. During execution of the Program Once command, any attempt to read addresses within the program flash block containing this 64-byte field returns invalid data.

**Table 27-45. Program Once Command Error Handling**

| Error Condition   | Error Bit      |
|---|----------------|
| Command not available in current mode/security                                    | FSTAT[ACCERR]  |
| An invalid record index is supplied   | FSTAT[ACCERR]  |
| The requested record has already been programmed to a non-FFFF value <sup>1</sup> | FSTAT[ACCERR]  |
| Any errors have been encountered during the verify operation                      | FSTAT[MGSTAT0] |

1. If a Program Once record is initially programmed to 0xFFFF\_FFFF, the Program Once command is allowed to execute again on that same record.

### 27.4.10.9 Erase All Blocks Command

The Erase All Blocks operation erases all flash memory, verifies all memory contents, and releases MCU security.

**Table 27-46. Erase All Blocks Command FCCOB Requirements**

| FCCOB Number | FCCOB Contents [7:0] |
|--------------|----------------------|
| 0            | 0x44 (ERSALL)        |

After clearing CCIF to launch the Erase All Blocks command, the flash memory module erases all program flash memory, then verifies that all are erased.

If the flash memory module verifies that all flash memories were properly erased, security is released by setting the FSEC[SEC] field to the unsecure state. The Erase All Blocks command aborts if any flash region is protected. The security byte and all other contents of the flash configuration field (see [Flash Configuration Field Description](#)) are erased by the Erase All Blocks command. If the erase-verify fails, the FSTAT[MGSTAT0] bit is set. The CCIF flag is set after the Erase All Blocks operation completes.

**Table 27-47. Erase All Blocks Command Error Handling**

| Error Condition  | Error Bit      |
|--|----------------|
| Command not available in current mode/security               | FSTAT[ACCERR]  |
| Any region of the program flash memory is protected          | FSTAT[FPIOL]   |
| Any errors have been encountered during the verify operation | FSTAT[MGSTAT0] |

#### 27.4.10.9.1 Triggering an Erase All External to the Flash Memory Module

The functionality of the Erase All Blocks command is also available in an uncommanded fashion outside of the flash memory. Refer to the device's Chip Configuration details for information on this functionality.

Before invoking the external erase all function, the FSTAT[ACCERR and PVIOL] flags must be cleared and the FCCOB0 register must not contain 0x44. When invoked, the erase-all function erases all program flash memory regardless of the protection settings. If the post-erase verify passes, the routine then releases security by setting the FSEC[SEC] field register to the unsecure state. The security byte in the Flash Configuration Field is also programmed to the unsecure state. The status of the erase-all request is reflected in the FCNFG[ERSAREQ] bit. The FCNFG[ERSAREQ] bit is cleared once the operation completes and the normal FSTAT error reporting is available as described in [Erase All Blocks Command](#).

### 27.4.10.10 Verify Backdoor Access Key Command

The Verify Backdoor Access Key command only executes if the mode and security conditions are satisfied (see [Flash Commands by Mode](#)). Execution of the Verify Backdoor Access Key command is further qualified by the FSEC[KEYEN] bits. The Verify Backdoor Access Key command releases security if user-supplied keys in the FCCOB match those stored in the Backdoor Comparison Key bytes of the Flash Configuration Field (see [Flash Configuration Field Description](#)). The column labelled Flash Configuration Field offset address shows the location of the matching byte in the Flash Configuration Field.

**Table 27-48. Verify Backdoor Access Key Command FCCOB Requirements**

| FCCOB Number | FCCOB Contents [7:0] | Flash Configuration Field Offset Address |
|--------------|----------------------|--|
| 0            | 0x45 (VFYKEY)        |  |
| 1-3          | Not Used             |  |
| 4            | Key Byte 0           | 0x0_0000                                 |
| 5            | Key Byte 1           | 0x0_0001                                 |
| 6            | Key Byte 2           | 0x0_0002                                 |
| 7            | Key Byte 3           | 0x0_0003                                 |
| 8            | Key Byte 4           | 0x0_0004                                 |
| 9            | Key Byte 5           | 0x0_0005                                 |
| A            | Key Byte 6           | 0x0_0006                                 |
| B            | Key Byte 7           | 0x0_0007                                 |

After clearing CCIF to launch the Verify Backdoor Access Key command, the flash memory module checks the FSEC[KEYEN] bits to verify that this command is enabled. If not enabled, the flash memory module sets the FSTAT[ACCERR] bit and terminates. If the command is enabled, the flash memory module compares the key provided in FCCOB to the backdoor comparison key in the Flash Configuration Field. If the backdoor keys match, the FSEC[SEC] field is changed to the unsecure state and security is 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 immediately aborted and the FSTAT[ACCERR] bit is (again) set to 1 until a reset of the flash memory module module occurs. If the entire 8-byte key is all zeros or all ones, the Verify Backdoor Access Key command fails with an access error. The CCIF flag is set after the Verify Backdoor Access Key operation completes.

**Table 27-49. Verify Backdoor Access Key Command Error Handling**

| Error Condition                       | Error Bit     |
|---------------------------------------|---------------|
| The supplied key is all-0s or all-Fs  | FSTAT[ACCERR] |
| An incorrect backdoor key is supplied | FSTAT[ACCERR] |

*Table continues on the next page...*

**Table 27-49. Verify Backdoor Access Key Command Error Handling (continued)**

| Error Condition  | Error Bit     |
|--|---------------|
| Backdoor key access has not been enabled (see the description of the FSEC register)          | FSTAT[ACCERR] |
| This command is launched and the backdoor key has mismatched since the last power down reset | FSTAT[ACCERR] |

## 27.4.11 Security

The flash memory module provides security information to the MCU based on contents of the FSEC security register. The MCU then limits access to flash memory resources as defined in the device's Chip Configuration details. During reset, the flash memory module initializes the FSEC register using data read from the security byte of the Flash Configuration Field (see [Flash Configuration Field Description](#)).

The following fields are available in the FSEC register. The settings are described in the [Flash Security Register \(FTFA\\_FSEC\)](#) details.

**Table 27-50. FSEC register fields**

| FSEC field | Description              |
|------------|--------------------------|
| KEYEN      | Backdoor Key Access      |
| MEEN       | Mass Erase Capability    |
| FSLACC     | Freescale Factory Access |
| SEC        | MCU security             |

### 27.4.11.1 Flash Memory Access by Mode and Security

The following table summarizes how access to the flash memory module is affected by security and operating mode.

**Table 27-51. Flash Memory Access Summary**

| Operating Mode | Chip Security State |  |
|----------------|---------------------|--|
|                | Unsecure            | Secure   |
| NVM Normal     | Full command set    |  |
| NVM Special    | Full command set    | Only the Erase All Blocks and Read 1s All Blocks commands. |

## 27.4.11.2 Changing the Security State

The security state out of reset can be permanently changed by programming the security byte of the flash configuration field. This assumes that you are starting from a mode where the necessary program flash erase and program commands are available and that the region of the program flash containing the flash configuration field is unprotected. If the flash security byte is successfully programmed, its new value takes affect after the next chip reset.

### 27.4.11.2.1 Unsecuring the Chip Using Backdoor Key Access

The chip can be unsecured by using the backdoor key access feature, which requires knowledge of the contents of the 8-byte backdoor key value stored in the Flash Configuration Field (see [Flash Configuration Field Description](#)). If the FSEC[KEYEN] bits are in the enabled state, the Verify Backdoor Access Key command (see [Verify Backdoor Access Key Command](#)) can be run; it allows the user to present prospective keys for comparison to the stored keys. If the keys match, the FSEC[SEC] bits are changed to unsecure the chip. The entire 8-byte key cannot be all 0s or all 1s; that is, 0000\_0000\_0000\_0000h and FFFF\_FFFF\_FFFF\_FFFFh are not accepted by the Verify Backdoor Access Key command as valid comparison values. While the Verify Backdoor Access Key command is active, program flash memory is not available for read access and returns invalid data.

The user code stored in the program 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 bits are in the enabled state, the chip can be unsecured by the following backdoor key access sequence:

1. Follow the command sequence for the Verify Backdoor Access Key command as explained in [Verify Backdoor Access Key Command](#)
2. If the Verify Backdoor Access Key command is successful, the chip is unsecured and the FSEC[SEC] bits are forced to the unsecure state

An illegal key provided to the Verify Backdoor Access Key command prohibits further use of the Verify Backdoor Access Key command. A reset of the chip is the only method to re-enable the Verify Backdoor Access Key command when a comparison fails.

After the backdoor keys have been correctly matched, the chip is unsecured by changing the FSEC[SEC] bits. A successful execution of the Verify Backdoor Access Key command changes the security in the FSEC register only. It does not alter the security byte or the keys stored in the Flash Configuration Field ([Flash Configuration Field Description](#)). After the next reset of the chip, the security state of the flash memory

module reverts back to the flash security byte in the Flash Configuration Field. The Verify Backdoor Access Key command sequence has no effect on the program and erase protections defined in the program flash protection registers.

If the backdoor keys successfully match, the unsecured chip has full control of the contents of the Flash Configuration Field. The chip may erase the sector containing the Flash Configuration Field and reprogram the flash security byte to the unsecure state and change the backdoor keys to any desired value.

## **27.4.12 Reset Sequence**

On each system reset the flash memory module executes a sequence which establishes initial values for the flash block configuration parameters, FPROT, FOPT, and FSEC registers.

FSTAT[CCIF] is cleared throughout the reset sequence. The flash memory module holds off CPU access during the reset sequence. Flash reads are possible when the hold is removed. Completion of the reset sequence is marked by setting CCIF which enables flash user commands.

If a reset occurs while any flash command is in progress, that command is immediately aborted. The state of the word being programmed or the sector/block being erased is not guaranteed. Commands and operations do not automatically resume after exiting reset.

# **Chapter 28**

## **Analog-to-Digital Converter (ADC)**

### **28.1 Introduction**

#### **NOTE**

For the chip-specific implementation details of this module's instances see the chip configuration information.

The 16-bit analog-to-digital converter (ADC) is a successive approximation ADC designed for operation within an integrated microcontroller system-on-chip.

#### **NOTE**

For the chip specific modes of operation, see the power management information of the device.

#### **28.1.1 Features**

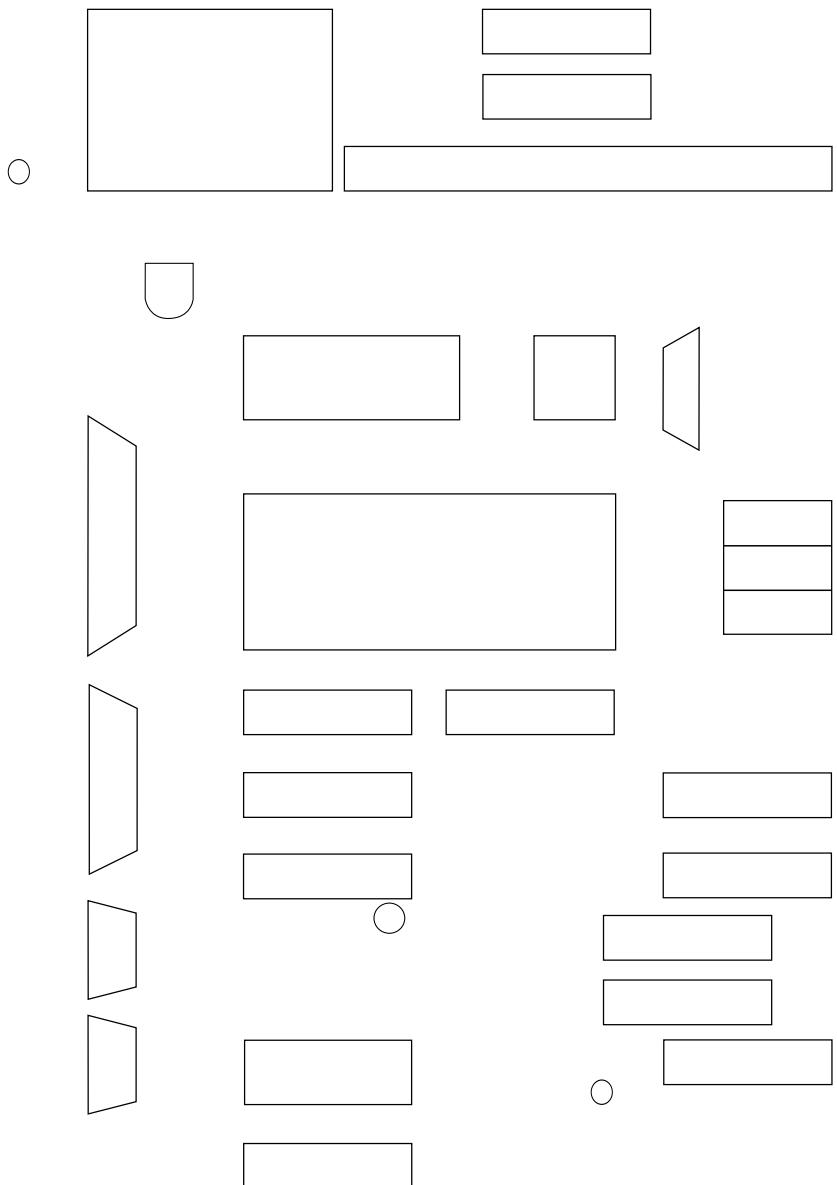
Features of the ADC module include:

- Linear successive approximation algorithm with up to 16-bit resolution
- Up to four pairs of differential and 24 single-ended external analog inputs
- Output modes:
  - differential 16-bit, 13-bit, 11-bit, and 9-bit modes
  - single-ended 16-bit, 12-bit, 10-bit, and 8-bit modes
- Output format in 2's complement 16-bit sign extended for differential modes
- Output in right-justified unsigned format for single-ended
- Single or continuous conversion, that is, automatic return to idle after single conversion

- Configurable sample time and conversion speed/power
- Conversion complete/hardware average complete flag and interrupt
- Input clock selectable from up to four sources
- Operation in Low-Power modes for lower noise
- Asynchronous clock source for lower noise operation with option to output the clock
- Selectable hardware conversion trigger with hardware channel select
- Automatic compare with interrupt for less-than, greater-than or equal-to, within range, or out-of-range, programmable value
- Temperature sensor
- Hardware average function
- Selectable voltage reference: external or alternate
- Self-Calibration mode

### **28.1.2 Block diagram**

The following figure is the ADC module block diagram.



**Table 28-1. ADC Signal Descriptions**

| Signal             | Description                        | I/O |
|--------------------|------------------------------------|-----|
| DADP3–DADP0        | Differential Analog Channel Inputs | I   |
| DADM3–DADM0        | Differential Analog Channel Inputs | I   |
| ADn                | Single-Ended Analog Channel Inputs | I   |
| V <sub>REFSH</sub> | Voltage Reference Select High      | I   |
| V <sub>REFSL</sub> | Voltage Reference Select Low       | I   |
| V <sub>DDA</sub>   | Analog Power Supply                | I   |
| V <sub>SSA</sub>   | Analog Ground                      | I   |

## 28.2.1 Analog Power (V<sub>DDA</sub>)

The ADC analog portion uses V<sub>DDA</sub> as its power connection. In some packages, V<sub>DDA</sub> is connected internally to V<sub>DD</sub>. If externally available, connect the V<sub>DDA</sub> pin to the same voltage potential as V<sub>DD</sub>. External filtering may be necessary to ensure clean V<sub>DDA</sub> for good results.

## 28.2.2 Analog Ground (V<sub>SSA</sub>)

The ADC analog portion uses V<sub>SSA</sub> as its ground connection. In some packages, V<sub>SSA</sub> is connected internally to V<sub>SS</sub>. If externally available, connect the V<sub>SSA</sub> pin to the same voltage potential as V<sub>SS</sub>.

## 28.2.3 Voltage Reference Select

V<sub>REFSH</sub> and V<sub>REFSL</sub> are the high and low reference voltages for the ADC module.

The ADC can be configured to accept one of two voltage reference pairs for V<sub>REFSH</sub> and V<sub>REFSL</sub>. Each pair contains a positive reference that must be between the minimum Ref Voltage High and V<sub>DDA</sub>, and a ground reference that must be at the same potential as V<sub>SSA</sub>. The two pairs are external (V<sub>REFH</sub> and V<sub>REFL</sub>) and alternate (V<sub>ALTH</sub> and V<sub>ALTL</sub>). These voltage references are selected using SC2[REFSEL]. The alternate V<sub>ALTH</sub> and V<sub>ALTL</sub> voltage reference pair may select additional external pins or internal sources depending on MCU configuration. See the chip configuration information on the Voltage References specific to this MCU.

In some packages, V<sub>REFH</sub> is connected in the package to V<sub>DDA</sub> and V<sub>REFL</sub> to V<sub>SSA</sub>. If externally available, the positive reference(s) may be connected to the same potential as V<sub>DDA</sub> or may be driven by an external source to a level between the minimum Ref Voltage High and the V<sub>DDA</sub> potential. V<sub>REFH</sub> must never exceed V<sub>DDA</sub>. Connect the ground references to the same voltage potential as V<sub>SSA</sub>.

## 28.2.4 Analog Channel Inputs (ADx)

The ADC module supports up to 24 single-ended analog inputs. A single-ended input is selected for conversion through the SC1[ADCH] channel select bits when SC1n[DIFF] is low.

## 28.2.5 Differential Analog Channel Inputs (DADx)

The ADC module supports up to four differential analog channel inputs. Each differential analog input is a pair of external pins, DADPx and DADMx, referenced to each other to provide the most accurate analog to digital readings. A differential input is selected for conversion through SC1[ADCH] when SC1n[DIFF] is high. All DADPx inputs may be used as single-ended inputs if SC1n[DIFF] is low. In certain MCU configurations, some DADMx inputs may also be used as single-ended inputs if SC1n[DIFF] is low. See the chip configuration chapter for ADC connections specific to this MCU.

## 28.3 Register definition

This section describes the ADC registers.

**ADC memory map**

| Absolute address (hex) | Register name                                  | Width (in bits) | Access | Reset value | Section/page               |
|------------------------|--|-----------------|--------|-------------|----------------------------|
| 4003_B000              | ADC Status and Control Registers 1 (ADC0_SC1A) | 32              | R/W    | 0000_001Fh  | <a href="#">28.3.1/462</a> |
| 4003_B004              | ADC Status and Control Registers 1 (ADC0_SC1B) | 32              | R/W    | 0000_001Fh  | <a href="#">28.3.1/462</a> |
| 4003_B008              | ADC Configuration Register 1 (ADC0_CFG1)       | 32              | R/W    | 0000_0000h  | <a href="#">28.3.2/465</a> |
| 4003_B00C              | ADC Configuration Register 2 (ADC0_CFG2)       | 32              | R/W    | 0000_0000h  | <a href="#">28.3.3/467</a> |
| 4003_B010              | ADC Data Result Register (ADC0_RA)             | 32              | R      | 0000_0000h  | <a href="#">28.3.4/468</a> |
| 4003_B014              | ADC Data Result Register (ADC0_RB)             | 32              | R      | 0000_0000h  | <a href="#">28.3.4/468</a> |
| 4003_B018              | Compare Value Registers (ADC0_CV1)             | 32              | R/W    | 0000_0000h  | <a href="#">28.3.5/469</a> |
| 4003_B01C              | Compare Value Registers (ADC0_CV2)             | 32              | R/W    | 0000_0000h  | <a href="#">28.3.5/469</a> |

*Table continues on the next page...*

## ADC memory map (continued)

| Absolute address (hex) | Register name   | Width (in bits) | Access | Reset value | Section/page                |
|------------------------|---|-----------------|--------|-------------|-----------------------------|
| 4003_B020              | Status and Control Register 2 (ADC0_SC2)                      | 32              | R/W    | 0000_0000h  | <a href="#">28.3.6/470</a>  |
| 4003_B024              | Status and Control Register 3 (ADC0_SC3)                      | 32              | R/W    | 0000_0000h  | <a href="#">28.3.7/472</a>  |
| 4003_B028              | ADC Offset Correction Register (ADC0_OFS)                     | 32              | R/W    | 0000_0004h  | <a href="#">28.3.8/474</a>  |
| 4003_B02C              | ADC Plus-Side Gain Register (ADC0_PG)                         | 32              | R/W    | 0000_8200h  | <a href="#">28.3.9/474</a>  |
| 4003_B030              | ADC Minus-Side Gain Register (ADC0_MG)                        | 32              | R/W    | 0000_8200h  | <a href="#">28.3.10/475</a> |
| 4003_B034              | ADC Plus-Side General Calibration Value Register (ADC0_CLPD)  | 32              | R/W    | 0000_000Ah  | <a href="#">28.3.11/475</a> |
| 4003_B038              | ADC Plus-Side General Calibration Value Register (ADC0_CLPS)  | 32              | R/W    | 0000_0020h  | <a href="#">28.3.12/476</a> |
| 4003_B03C              | ADC Plus-Side General Calibration Value Register (ADC0_CLP4)  | 32              | R/W    | 0000_0200h  | <a href="#">28.3.13/476</a> |
| 4003_B040              | ADC Plus-Side General Calibration Value Register (ADC0_CLP3)  | 32              | R/W    | 0000_0100h  | <a href="#">28.3.14/477</a> |
| 4003_B044              | ADC Plus-Side General Calibration Value Register (ADC0_CLP2)  | 32              | R/W    | 0000_0080h  | <a href="#">28.3.15/477</a> |
| 4003_B048              | ADC Plus-Side General Calibration Value Register (ADC0_CLP1)  | 32              | R/W    | 0000_0040h  | <a href="#">28.3.16/478</a> |
| 4003_B04C              | ADC Plus-Side General Calibration Value Register (ADC0_CLP0)  | 32              | R/W    | 0000_0020h  | <a href="#">28.3.17/478</a> |
| 4003_B054              | ADC Minus-Side General Calibration Value Register (ADC0_CLMD) | 32              | R/W    | 0000_000Ah  | <a href="#">28.3.18/479</a> |
| 4003_B058              | ADC Minus-Side General Calibration Value Register (ADC0_CLMS) | 32              | R/W    | 0000_0020h  | <a href="#">28.3.19/479</a> |
| 4003_B05C              | ADC Minus-Side General Calibration Value Register (ADC0_CLM4) | 32              | R/W    | 0000_0200h  | <a href="#">28.3.20/480</a> |
| 4003_B060              | ADC Minus-Side General Calibration Value Register (ADC0_CLM3) | 32              | R/W    | 0000_0100h  | <a href="#">28.3.21/480</a> |
| 4003_B064              | ADC Minus-Side General Calibration Value Register (ADC0_CLM2) | 32              | R/W    | 0000_0080h  | <a href="#">28.3.22/481</a> |
| 4003_B068              | ADC Minus-Side General Calibration Value Register (ADC0_CLM1) | 32              | R/W    | 0000_0040h  | <a href="#">28.3.23/481</a> |
| 4003_B06C              | ADC Minus-Side General Calibration Value Register (ADC0_CLM0) | 32              | R/W    | 0000_0020h  | <a href="#">28.3.24/482</a> |

### 28.3.1 ADC Status and Control Registers 1 (ADCx\_SC1n)

SC1A is used for both software and hardware trigger modes of operation.

To allow sequential conversions of the ADC to be triggered by internal peripherals, the ADC can have more than one status and control register: one for each conversion. The SC1B–SC1n registers indicate potentially multiple SC1 registers for use only in hardware

trigger mode. See the chip configuration information about the number of SC1n registers specific to this device. The SC1n registers have identical fields, and are used in a "ping-pong" approach to control ADC operation.

At any one point in time, only one of the SC1n registers is actively controlling ADC conversions. Updating SC1A while SC1n is actively controlling a conversion is allowed, and vice-versa for any of the SC1n registers specific to this MCU.

Writing SC1A while SC1A is actively controlling a conversion aborts the current conversion. In Software Trigger mode, when SC2[ADTRG]=0, writes to SC1A subsequently initiate a new conversion, if SC1[ADCH] contains a value other than all 1s.

Writing any of the SC1n registers while that specific SC1n register is actively controlling a conversion aborts the current conversion. None of the SC1B-SC1n registers are used for software trigger operation and therefore writes to the SC1B-SC1n registers do not initiate a new conversion.

Address: 4003\_B000h base + 0h offset + (4d × i), where i=0d to 1d

| Bit   | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 | 23   | 22   | 21   | 20 | 19 | 18 | 17 | 16 |
|-------|----|----|----|----|----|----|----|----|------|------|------|----|----|----|----|----|
| R     |    |    |    |    |    |    |    |    | 0    |      |      |    |    |    |    |    |
| W     |    |    |    |    |    |    |    |    |      |      |      |    |    |    |    |    |
| Reset | 0  | 0  | 0  | 0  | 0  | 0  | 0  | 0  | 0    | 0    | 0    | 0  | 0  | 0  | 0  | 0  |
| Bit   | 15 | 14 | 13 | 12 | 11 | 10 | 9  | 8  | 7    | 6    | 5    | 4  | 3  | 2  | 1  | 0  |
| R     |    |    |    |    |    |    |    |    | 0    |      |      |    |    |    |    |    |
| W     |    |    |    |    |    |    |    |    | COCO | AIEN | DIFF |    |    |    |    |    |
| Reset | 0  | 0  | 0  | 0  | 0  | 0  | 0  | 0  | 0    | 0    | 0    | 0  | 1  | 1  | 1  | 1  |

#### ADCx\_SC1n field descriptions

| Field            | Description   |
|------------------|---|
| 31–8<br>Reserved | This field is reserved.<br>This read-only field is reserved and always has the value 0. |
| 7<br>COCO        | Conversion Complete Flag  |

Table continues on the next page...

**ADCx\_SC1n field descriptions (continued)**

| Field    | Description   |
|----------|---|
|          | <p>This is a read-only field that is set each time a conversion is completed when the compare function is disabled, or SC2[ACFE]=0 and the hardware average function is disabled, or SC3[AVGE]=0. When the compare function is enabled, or SC2[ACFE]=1, COCO is set upon completion of a conversion only if the compare result is true. When the hardware average function is enabled, or SC3[AVGE]=1, COCO is set upon completion of the selected number of conversions (determined by AVGS). COCO in SC1A is also set at the completion of a calibration sequence. COCO is cleared when the respective SC1n register is written or when the respective Rn register is read.</p> <p>0 Conversion is not completed.<br/>1 Conversion is completed.</p>  |
| 6 AIEN   | <p>Interrupt Enable</p> <p>Enables conversion complete interrupts. When COCO becomes set while the respective AIEN is high, an interrupt is asserted.</p> <p>0 Conversion complete interrupt is disabled.<br/>1 Conversion complete interrupt is enabled.</p>   |
| 5 DIFF   | <p>Differential Mode Enable</p> <p>Configures the ADC to operate in differential mode. When enabled, this mode automatically selects from the differential channels, and changes the conversion algorithm and the number of cycles to complete a conversion.</p> <p>0 Single-ended conversions and input channels are selected.<br/>1 Differential conversions and input channels are selected.</p>   |
| 4–0 ADCH | <p>Input channel select</p> <p>Selects one of the input channels. The input channel decode depends on the value of DIFF. DAD0-DAD3 are associated with the input pin pairs DADPx and DADMx.</p> <p><b>NOTE:</b> Some of the input channel options in the bitfield-setting descriptions might not be available for your device. For the actual ADC channel assignments for your device, see the Chip Configuration details.</p> <p>The successive approximation converter subsystem is turned off when the channel select bits are all set, that is, ADCH = 11111. This feature allows explicit disabling of the ADC and isolation of the input channel from all sources. Terminating continuous conversions this way prevents an additional single conversion from being performed. It is not necessary to set ADCH to all 1s to place the ADC in a low-power state when continuous conversions are not enabled because the module automatically enters a low-power state when a conversion completes.</p> <p>00000 When DIFF=0, DADP0 is selected as input; when DIFF=1, DAD0 is selected as input.<br/>00001 When DIFF=0, DADP1 is selected as input; when DIFF=1, DAD1 is selected as input.<br/>00010 When DIFF=0, DADP2 is selected as input; when DIFF=1, DAD2 is selected as input.<br/>00011 When DIFF=0, DADP3 is selected as input; when DIFF=1, DAD3 is selected as input.<br/>00100 When DIFF=0, AD4 is selected as input; when DIFF=1, it is reserved.<br/>00101 When DIFF=0, AD5 is selected as input; when DIFF=1, it is reserved.<br/>00110 When DIFF=0, AD6 is selected as input; when DIFF=1, it is reserved.<br/>00111 When DIFF=0, AD7 is selected as input; when DIFF=1, it is reserved.<br/>01000 When DIFF=0, AD8 is selected as input; when DIFF=1, it is reserved.<br/>01001 When DIFF=0, AD9 is selected as input; when DIFF=1, it is reserved.<br/>01010 When DIFF=0, AD10 is selected as input; when DIFF=1, it is reserved.<br/>01011 When DIFF=0, AD11 is selected as input; when DIFF=1, it is reserved.</p> |

*Table continues on the next page...*

**ADCx\_SC1n field descriptions (continued)**

| Field | Description  |
|-------|--|
| 01100 | When DIFF=0, AD12 is selected as input; when DIFF=1, it is reserved.   |
| 01101 | When DIFF=0, AD13 is selected as input; when DIFF=1, it is reserved.   |
| 01110 | When DIFF=0, AD14 is selected as input; when DIFF=1, it is reserved.   |
| 01111 | When DIFF=0, AD15 is selected as input; when DIFF=1, it is reserved.   |
| 10000 | When DIFF=0, AD16 is selected as input; when DIFF=1, it is reserved.   |
| 10001 | When DIFF=0, AD17 is selected as input; when DIFF=1, it is reserved.   |
| 10010 | When DIFF=0, AD18 is selected as input; when DIFF=1, it is reserved.   |
| 10011 | When DIFF=0, AD19 is selected as input; when DIFF=1, it is reserved.   |
| 10100 | When DIFF=0, AD20 is selected as input; when DIFF=1, it is reserved.   |
| 10101 | When DIFF=0, AD21 is selected as input; when DIFF=1, it is reserved.   |
| 10110 | When DIFF=0, AD22 is selected as input; when DIFF=1, it is reserved.   |
| 10111 | When DIFF=0, AD23 is selected as input; when DIFF=1, it is reserved.   |
| 11000 | Reserved.  |
| 11001 | Reserved.  |
| 11010 | When DIFF=0, Temp Sensor (single-ended) is selected as input; when DIFF=1, Temp Sensor (differential) is selected as input.  |
| 11011 | When DIFF=0, Bandgap (single-ended) is selected as input; when DIFF=1, Bandgap (differential) is selected as input.  |
| 11100 | Reserved.  |
| 11101 | When DIFF=0, V <sub>REFSH</sub> is selected as input; when DIFF=1, -V <sub>REFSH</sub> (differential) is selected as input. Voltage reference selected is determined by SC2[REFSEL]. |
| 11110 | When DIFF=0, V <sub>REFSL</sub> is selected as input; when DIFF=1, it is reserved. Voltage reference selected is determined by SC2[REFSEL].  |
| 11111 | Module is disabled.  |

**28.3.2 ADC Configuration Register 1 (ADCx\_CFG1)**

The configuration Register 1 (CFG1) selects the mode of operation, clock source, clock divide, and configuration for low power or long sample time.

Address: 4003\_B000h base + 8h offset = 4003\_B008h

| Bit   | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 | 23    | 22   | 21     | 20   | 19     | 18 | 17 | 16 |
|-------|----|----|----|----|----|----|----|----|-------|------|--------|------|--------|----|----|----|
| R     |    |    |    |    |    |    |    |    | 0     |      |        |      |        |    |    |    |
| W     |    |    |    |    |    |    |    |    |       |      |        |      |        |    |    |    |
| Reset | 0  | 0  | 0  | 0  | 0  | 0  | 0  | 0  | 0     | 0    | 0      | 0    | 0      | 0  | 0  | 0  |
| Bit   | 15 | 14 | 13 | 12 | 11 | 10 | 9  | 8  | 7     | 6    | 5      | 4    | 3      | 2  | 1  | 0  |
| R     |    |    |    |    |    |    |    |    | ADLPC | ADIV | ADLSMP | MODE | ADICLK |    |    |    |
| W     |    |    |    |    |    |    |    |    | 0     | 0    | 0      | 0    | 0      | 0  | 0  | 0  |
| Reset | 0  | 0  | 0  | 0  | 0  | 0  | 0  | 0  | 0     | 0    | 0      | 0    | 0      | 0  | 0  | 0  |

**ADCx\_CFG1 field descriptions**

| Field            | Description   |
|------------------|---|
| 31–8<br>Reserved | This field is reserved.<br>This read-only field is reserved and always has the value 0.   |
| 7<br>ADLPC       | Low-Power Configuration<br><br>Controls the power configuration of the successive approximation converter. This optimizes power consumption when higher sample rates are not required.<br><br>0 Normal power configuration.<br>1 Low-power configuration. The power is reduced at the expense of maximum clock speed.   |
| 6–5<br>ADIV      | Clock Divide Select<br><br>ADIV selects the divide ratio used by the ADC to generate the internal clock ADCK.<br><br>00 The divide ratio is 1 and the clock rate is input clock.<br>01 The divide ratio is 2 and the clock rate is (input clock)/2.<br>10 The divide ratio is 4 and the clock rate is (input clock)/4.<br>11 The divide ratio is 8 and the clock rate is (input clock)/8.   |
| 4<br>ADLSMP      | Sample time configuration<br><br>ADLSMP selects between different sample times based on the conversion mode selected. This bit adjusts the sample period to allow higher impedance inputs to be accurately sampled or to maximize conversion speed for lower impedance inputs. Longer sample times can also be used to lower overall power consumption if continuous conversions are enabled and high conversion rates are not required. When ADLSMP=1, the long sample time select bits, (ADLSTS[1:0]), can select the extent of the long sample time.<br><br>0 Short sample time.<br>1 Long sample time.                              |
| 3–2<br>MODE      | Conversion mode selection<br><br>Selects the ADC resolution mode.<br><br>00 When DIFF=0:It is single-ended 8-bit conversion; when DIFF=1, it is differential 9-bit conversion with 2's complement output.<br>01 When DIFF=0:It is single-ended 12-bit conversion ; when DIFF=1, it is differential 13-bit conversion with 2's complement output.<br>10 When DIFF=0:It is single-ended 10-bit conversion ; when DIFF=1, it is differential 11-bit conversion with 2's complement output.<br>11 When DIFF=0:It is single-ended 16-bit conversion; when DIFF=1, it is differential 16-bit conversion with 2's complement output.           |
| 1–0<br>ADICLK    | Input Clock Select<br><br>Selects the input clock source to generate the internal clock, ADCK. Note that when the ADACK clock source is selected, it is not required to be active prior to conversion start. When it is selected and it is not active prior to a conversion start, when CFG2[ADACKEN]=0, the asynchronous clock is activated at the start of a conversion and deactivated when conversions are terminated. In this case, there is an associated clock startup delay each time the clock source is re-activated.<br><br>00 Bus clock<br>01 (Bus clock)/2<br>10 Alternate clock (ALTCLK)<br>11 Asynchronous clock (ADACK) |

### 28.3.3 ADC Configuration Register 2 (ADCx\_CFG2)

Configuration Register 2 (CFG2) selects the special high-speed configuration for very high speed conversions and selects the long sample time duration during long sample mode.

Address: 4003\_B000h base + Ch offset = 4003\_B00Ch

| Bit   | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |   | 23 | 22 | 21 | 20     | 19      | 18    | 17 | 16     |
|-------|----|----|----|----|----|----|----|----|---|----|----|----|--------|---------|-------|----|--------|
| R     |    |    |    |    |    |    |    |    | 0 |    |    |    |        |         |       |    |        |
| W     |    |    |    |    |    |    |    |    |   |    |    |    |        |         |       |    |        |
| Reset | 0  | 0  | 0  | 0  | 0  | 0  | 0  | 0  |   | 0  | 0  | 0  | 0      | 0       | 0     | 0  | 0      |
| Bit   | 15 | 14 | 13 | 12 | 11 | 10 | 9  | 8  |   | 7  | 6  | 5  | 4      | 3       | 2     | 1  | 0      |
| R     |    |    |    |    |    |    |    |    | 0 |    | 0  |    | MUXSEL | ADACKEN | ADHSC |    | ADLSTS |
| W     |    |    |    |    |    |    |    |    |   |    |    |    | 0      | 0       | 0     | 0  | 0      |
| Reset | 0  | 0  | 0  | 0  | 0  | 0  | 0  | 0  |   | 0  | 0  | 0  | 0      | 0       | 0     | 0  | 0      |

#### ADCx\_CFG2 field descriptions

| Field            | Description   |
|------------------|---|
| 31–8<br>Reserved | This field is reserved.<br>This read-only field is reserved and always has the value 0.   |
| 7–5<br>Reserved  | This field is reserved.<br>This read-only field is reserved and always has the value 0.   |
| 4<br>MUXSEL      | ADC Mux Select<br><br>Changes the ADC mux setting to select between alternate sets of ADC channels.<br><br>0 ADxxa channels are selected.<br>1 ADxxb channels are selected.   |
| 3<br>ADACKEN     | Asynchronous Clock Output Enable<br><br>Enables the asynchronous clock source and the clock source output regardless of the conversion and status of CFG1[ADICLK]. Based on MCU configuration, the asynchronous clock may be used by other modules. See chip configuration information. Setting this field allows the clock to be used even while the ADC is idle or operating from a different clock source. Also, latency of initiating a single or first-continuous conversion with the asynchronous clock selected is reduced because the ADACK clock is already operational.<br><br>0 Asynchronous clock output disabled; Asynchronous clock is enabled only if selected by ADICLK and a conversion is active.<br>1 Asynchronous clock and clock output is enabled regardless of the state of the ADC. |
| 2<br>ADHSC       | High-Speed Configuration  |

Table continues on the next page...

## ADCx\_CFG2 field descriptions (continued)

| Field         | Description   |
|---------------|---|
|               | <p>Configures the ADC for very high-speed operation. The conversion sequence is altered with 2 ADCK cycles added to the conversion time to allow higher speed conversion clocks.</p> <p>0 Normal conversion sequence selected.<br/>1 High-speed conversion sequence selected with 2 additional ADCK cycles to total conversion time.</p>  |
| 1–0<br>ADLSTS | <p>Long Sample Time Select</p> <p>Selects between the extended sample times when long sample time is selected, that is, when CFG1[ADLSMP]=1. This allows higher impedance inputs to be accurately sampled or to maximize conversion speed for lower impedance inputs. Longer sample times can also be used to lower overall power consumption when continuous conversions are enabled if high conversion rates are not required.</p> <p>00 Default longest sample time; 20 extra ADCK cycles; 24 ADCK cycles total.<br/>01 12 extra ADCK cycles; 16 ADCK cycles total sample time.<br/>10 6 extra ADCK cycles; 10 ADCK cycles total sample time.<br/>11 2 extra ADCK cycles; 6 ADCK cycles total sample time.</p> |

### 28.3.4 ADC Data Result Register (ADCx\_Rn)

The data result registers (Rn) contain the result of an ADC conversion of the channel selected by the corresponding status and channel control register (SC1A:SC1n). For every status and channel control register, there is a corresponding data result register.

Unused bits in R n are cleared in unsigned right-justified modes and carry the sign bit (MSB) in sign-extended 2's complement modes. For example, when configured for 10-bit single-ended mode, D[15:10] are cleared. When configured for 11-bit differential mode, D[15:10] carry the sign bit, that is, bit 10 extended through bit 15.

The following table describes the behavior of the data result registers in the different modes of operation.

Table 28-43. Data result register description

| Conversion mode     | D15 | D14 | D13 | D12 | D11 | D10 | D9 | D8 | D7 | D6 | D5 | D4 | D3 | D2 | D1 | D0 | Format |
|---------------------|-----|-----|-----|-----|-----|-----|----|----|----|----|----|----|----|----|----|----|--------|
| 16-bit differential | S   | D   | D   | D   | D   | D   | D  | D  | D  | D  | D  | D  | D  | D  | D  | D  | 16-bit |
| 16-bit single-ended | D   | D   | D   | D   | D   | D   | D  | D  | D  | D  | D  | D  | D  | D  | D  | D  | 16-bit |
| 13-bit differential | S   |     |     |     |     |     |    |    |    |    |    |    |    |    |    |    | 13-bit |
| 12-bit single-ended | 0   |     |     |     |     |     |    |    |    |    |    |    |    |    |    |    | 12-bit |
| 11-bit differential | S   |     |     |     |     |     |    |    |    |    |    |    |    |    |    |    | 11-bit |

Table continues on the next page...

**Table 28-43. Data result register description (continued)**

| Conversion mode     | D15 | D14 | D13 | D12 | D11 | D10 | D9 | D8 | D7 | D6 | D5 | D4 | D3 | D2 | D1 | D0 | Format                       |
|---------------------|-----|-----|-----|-----|-----|-----|----|----|----|----|----|----|----|----|----|----|------------------------------|
| 10-bit single-ended | 0   | 0   | 0   | 0   | 0   | 0   | D  | D  | D  | D  | D  | D  | D  | D  | D  | D  | Unsigned right-justified     |
| 9-bit differential  | S   | S   | S   | S   | S   | S   | S  | D  | D  | D  | D  | D  | D  | D  | D  | D  | Sign-extended 2's complement |
| 8-bit single-ended  | 0   | 0   | 0   | 0   | 0   | 0   | 0  | D  | D  | D  | D  | D  | D  | D  | D  | D  | Unsigned right-justified     |

**NOTE**

S: Sign bit or sign bit extension;

D: Data, which is 2's complement data if indicated

Address: 4003\_B000h base + 10h offset + (4d × i), where i=0d to 1d

| Bit   | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|-------|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|---|---|---|---|---|---|---|---|---|---|
| R     |    |    |    |    |    |    |    |    |    |    |    |    |    |    |    |    |    |    |    |    |    |    |   |   |   |   |   |   |   |   | D |   |
| W     |    |    |    |    |    |    |    |    |    |    |    |    |    |    |    |    |    |    |    |    |    |    |   |   |   |   |   |   |   |   |   |   |
| Reset | 0  | 0  | 0  | 0  | 0  | 0  | 0  | 0  | 0  | 0  | 0  | 0  | 0  | 0  | 0  | 0  | 0  | 0  | 0  | 0  | 0  | 0  | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |   |

**ADCx\_Rn field descriptions**

| Field             | Description   |
|-------------------|---|
| 31–16<br>Reserved | This field is reserved.<br>This read-only field is reserved and always has the value 0. |
| 15–0<br>D         | Data result   |

**28.3.5 Compare Value Registers (ADCx\_CVn)**

The compare value registers (CV1 and CV2) contain a compare value used to compare the conversion result when the compare function is enabled, that is, SC2[ACFE]=1. This register is formatted in the same way as the Rn registers in different modes of operation for both bit position definition and value format using unsigned or sign-extended 2's complement. Therefore, the compare function uses only the CVn fields that are related to the ADC mode of operation.

The compare value 2 register (CV2) is used only when the compare range function is enabled, that is, SC2[ACREN]=1.

## Register definition

Address: 4003\_B000h base + 18h offset + (4d × i), where i=0d to 1d

| Bit   | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|-------|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|---|---|---|---|---|---|---|---|---|---|
| R     |    |    |    |    |    |    |    |    |    |    |    |    |    |    |    |    |    |    |    |    |    |    |   |   |   |   |   |   |   |   |   |   |
| W     |    |    |    |    |    |    |    |    |    |    |    |    |    |    |    |    |    |    |    |    |    |    |   |   |   |   |   |   |   |   |   |   |
| Reset | 0  | 0  | 0  | 0  | 0  | 0  | 0  | 0  | 0  | 0  | 0  | 0  | 0  | 0  | 0  | 0  | 0  | 0  | 0  | 0  | 0  | 0  | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |   |   |

### ADCx\_CVn field descriptions

| Field             | Description   |
|-------------------|---|
| 31–16<br>Reserved | This field is reserved.<br>This read-only field is reserved and always has the value 0. |
| 15–0<br>CV        | Compare Value.  |

## 28.3.6 Status and Control Register 2 (ADCx\_SC2)

The status and control register 2 (SC2) contains the conversion active, hardware/software trigger select, compare function, and voltage reference select of the ADC module.

Address: 4003\_B000h base + 20h offset = 4003\_B020h

| Bit   | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 | 23    | 22    | 21   | 20    | 19    | 18    | 17 | 16     |
|-------|----|----|----|----|----|----|----|----|-------|-------|------|-------|-------|-------|----|--------|
| R     |    |    |    |    |    |    |    |    |       |       |      |       |       |       |    |        |
| W     |    |    |    |    |    |    |    |    |       |       |      |       |       |       |    |        |
| Reset | 0  | 0  | 0  | 0  | 0  | 0  | 0  | 0  | 0     | 0     | 0    | 0     | 0     | 0     | 0  | 0      |
| Bit   | 15 | 14 | 13 | 12 | 11 | 10 | 9  | 8  | 7     | 6     | 5    | 4     | 3     | 2     | 1  | 0      |
| R     |    |    |    |    |    |    |    |    | ADACT | ADTRG | ACFE | ACFGT | ACREN | DMAEN |    | REFSEL |
| W     |    |    |    |    |    |    |    |    | 0     | 0     | 0    | 0     | 0     | 0     | 0  |        |
| Reset | 0  | 0  | 0  | 0  | 0  | 0  | 0  | 0  | 0     | 0     | 0    | 0     | 0     | 0     | 0  | 0      |

**ADCx\_SC2 field descriptions**

| Field            | Description  |
|------------------|--|
| 31–8<br>Reserved | This field is reserved.<br>This read-only field is reserved and always has the value 0.  |
| 7<br>ADACT       | Conversion Active<br><br>Indicates that a conversion or hardware averaging is in progress. ADACT is set when a conversion is initiated and cleared when a conversion is completed or aborted.<br><br>0 Conversion not in progress.<br>1 Conversion in progress.  |
| 6<br>ADTRG       | Conversion Trigger Select<br><br>Selects the type of trigger used for initiating a conversion. Two types of trigger are selectable: <ul style="list-style-type: none"> <li>• Software trigger: When software trigger is selected, a conversion is initiated following a write to SC1A.</li> <li>• Hardware trigger: When hardware trigger is selected, a conversion is initiated following the assertion of the ADHWT input after a pulse of the ADHWTSn input.</li> </ul><br>0 Software trigger selected.<br>1 Hardware trigger selected. |
| 5<br>ACFE        | Compare Function Enable<br><br>Enables the compare function.<br><br>0 Compare function disabled.<br>1 Compare function enabled.  |
| 4<br>ACFGT       | Compare Function Greater Than Enable<br><br>Configures the compare function to check the conversion result relative to the CV1 and CV2 based upon the value of ACREN. ACFE must be set for ACFGT to have any effect.<br><br>0 Configures less than threshold, outside range not inclusive and inside range not inclusive; functionality based on the values placed in CV1 and CV2.<br>1 Configures greater than or equal to threshold, outside and inside ranges inclusive; functionality based on the values placed in CV1 and CV2.       |
| 3<br>ACREN       | Compare Function Range Enable<br><br>Configures the compare function to check if the conversion result of the input being monitored is either between or outside the range formed by CV1 and CV2 determined by the value of ACFGT. ACFE must be set for ACFGT to have any effect.<br><br>0 Range function disabled. Only CV1 is compared.<br>1 Range function enabled. Both CV1 and CV2 are compared.  |
| 2<br>DMAEN       | DMA Enable<br><br>0 DMA is disabled.<br>1 DMA is enabled and will assert the ADC DMA request during an ADC conversion complete event noted when any of the SC1n[COCO] flags is asserted.   |
| 1–0<br>REFSEL    | Voltage Reference Selection<br><br>Selects the voltage reference source used for conversions.<br><br>00 Default voltage reference pin pair, that is, external pins V <sub>REFH</sub> and V <sub>REFL</sub>   |

*Table continues on the next page...*

**ADCx\_SC2 field descriptions (continued)**

| Field | Description  |
|-------|--|
| 01    | Alternate reference pair, that is, $V_{ALTH}$ and $V_{ALTL}$ . This pair may be additional external pins or internal sources depending on the MCU configuration. See the chip configuration information for details specific to this MCU |
| 10    | Reserved   |
| 11    | Reserved   |

**28.3.7 Status and Control Register 3 (ADCx\_SC3)**

The Status and Control Register 3 (SC3) controls the calibration, continuous convert, and hardware averaging functions of the ADC module.

Address: 4003\_B000h base + 24h offset = 4003\_B024h

| Bit   | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 | 23 | 22   | 21 | 20 | 19   | 18   | 17 | 16   |
|-------|----|----|----|----|----|----|----|----|----|------|----|----|------|------|----|------|
| R     |    |    |    |    |    |    |    |    | 0  |      |    |    |      |      |    |      |
| W     |    |    |    |    |    |    |    |    |    |      |    |    |      |      |    |      |
| Reset | 0  | 0  | 0  | 0  | 0  | 0  | 0  | 0  | 0  | 0    | 0  | 0  | 0    | 0    | 0  | 0    |
| Bit   | 15 | 14 | 13 | 12 | 11 | 10 | 9  | 8  | 7  | 6    | 5  | 4  | 3    | 2    | 1  | 0    |
| R     |    |    |    |    |    |    |    |    | 0  | CALF |    | 0  | ADCO | AVGE |    | AVGS |
| W     |    |    |    |    |    |    |    |    |    |      |    |    |      |      |    |      |
| Reset | 0  | 0  | 0  | 0  | 0  | 0  | 0  | 0  | 0  | 0    | 0  | 0  | 0    | 0    | 0  | 0    |

**ADCx\_SC3 field descriptions**

| Field            | Description   |
|------------------|---|
| 31–8<br>Reserved | This field is reserved.<br>This read-only field is reserved and always has the value 0. |

*Table continues on the next page...*

**ADCx\_SC3 field descriptions (continued)**

| Field           | Description   |
|-----------------|---|
| 7<br>CAL        | Calibration<br><br>Begins the calibration sequence when set. This field stays set while the calibration is in progress and is cleared when the calibration sequence is completed. CALF must be checked to determine the result of the calibration sequence. Once started, the calibration routine cannot be interrupted by writes to the ADC registers or the results will be invalid and CALF will set. Setting CAL will abort any current conversion. |
| 6<br>CALF       | Calibration Failed Flag<br><br>Displays the result of the calibration sequence. The calibration sequence will fail if SC2[ADTRG] = 1, any ADC register is written, or any stop mode is entered before the calibration sequence completes. Writing 1 to CALF clears it.<br><br>0 Calibration completed normally.<br>1 Calibration failed. ADC accuracy specifications are not guaranteed.  |
| 5–4<br>Reserved | This field is reserved.<br>This read-only field is reserved and always has the value 0.   |
| 3<br>ADCO       | Continuous Conversion Enable<br><br>Enables continuous conversions.<br><br>0 One conversion or one set of conversions if the hardware average function is enabled, that is, AVGE=1, after initiating a conversion.<br>1 Continuous conversions or sets of conversions if the hardware average function is enabled, that is, AVGE=1, after initiating a conversion.  |
| 2<br>AVGE       | Hardware Average Enable<br><br>Enables the hardware average function of the ADC.<br><br>0 Hardware average function disabled.<br>1 Hardware average function enabled.   |
| 1–0<br>AVGS     | Hardware Average Select<br><br>Determines how many ADC conversions will be averaged to create the ADC average result.<br><br>00 4 samples averaged.<br>01 8 samples averaged.<br>10 16 samples averaged.<br>11 32 samples averaged.   |

### 28.3.8 ADC Offset Correction Register (ADCx\_OFS)

The ADC Offset Correction Register (OFS) contains the user-selected or calibration-generated offset error correction value. This register is a 2's complement, left-justified, 16-bit value. The value in OFS is subtracted from the conversion and the result is transferred into the result registers, Rn. If the result is greater than the maximum or less than the minimum result value, it is forced to the appropriate limit for the current mode of operation.

Address: 4003\_B000h base + 28h offset = 4003\_B028h

| Bit   | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|-------|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|---|---|---|---|---|---|---|---|---|---|
| R     |    |    |    |    |    |    |    |    |    |    |    |    |    |    |    | 0  |    |    |    |    |    |    |   |   |   |   |   |   |   |   |   |   |
| W     |    |    |    |    |    |    |    |    |    |    |    |    |    |    |    |    |    |    |    |    |    |    |   |   |   |   |   |   |   |   |   |   |
| Reset | 0  | 0  | 0  | 0  | 0  | 0  | 0  | 0  | 0  | 0  | 0  | 0  | 0  | 0  | 0  | 0  | 0  | 0  | 0  | 0  | 0  | 0  | 0 | 0 | 0 | 0 | 0 | 1 | 0 | 0 |   |   |

#### ADCx\_OFS field descriptions

| Field             | Description   |
|-------------------|---|
| 31–16<br>Reserved | This field is reserved.<br>This read-only field is reserved and always has the value 0. |
| 15–0<br>OFS       | Offset Error Correction Value   |

### 28.3.9 ADC Plus-Side Gain Register (ADCx\_PG)

The Plus-Side Gain Register (PG) contains the gain error correction for the plus-side input in differential mode or the overall conversion in single-ended mode. PG, a 16-bit real number in binary format, is the gain adjustment factor, with the radix point fixed between ADPG15 and ADPG14. This register must be written by the user with the value described in the calibration procedure. Otherwise, the gain error specifications may not be met.

Address: 4003\_B000h base + 2Ch offset = 4003\_B02Ch

| Bit   | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|-------|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|---|---|---|---|---|---|---|---|---|---|
| R     |    |    |    |    |    |    |    |    |    |    |    |    |    |    |    | 0  |    |    |    |    |    |    |   |   |   |   |   |   |   |   |   |   |
| W     |    |    |    |    |    |    |    |    |    |    |    |    |    |    |    |    |    |    |    |    |    |    |   |   |   |   |   |   |   |   |   |   |
| Reset | 0  | 0  | 0  | 0  | 0  | 0  | 0  | 0  | 0  | 0  | 0  | 0  | 0  | 0  | 0  | 0  | 1  | 0  | 0  | 0  | 0  | 1  | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |   |

**ADCx\_PG field descriptions**

| Field             | Description   |
|-------------------|---|
| 31–16<br>Reserved | This field is reserved.<br>This read-only field is reserved and always has the value 0. |
| 15–0<br>PG        | Plus-Side Gain  |

**28.3.10 ADC Minus-Side Gain Register (ADCx\_MG)**

The Minus-Side Gain Register (MG) contains the gain error correction for the minus-side input in differential mode. This register is ignored in single-ended mode. MG, a 16-bit real number in binary format, is the gain adjustment factor, with the radix point fixed between ADMG15 and ADMG14. This register must be written by the user with the value described in the calibration procedure. Otherwise, the gain error specifications may not be met.

Address: 4003\_B000h base + 30h offset = 4003\_B030h

| Bit   | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|-------|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|---|---|---|---|---|---|---|---|---|---|
| R     |    |    |    |    |    |    |    |    |    |    |    |    |    |    |    | 0  |    |    |    |    |    |    |   |   |   |   |   |   |   |   |   |   |
| W     |    |    |    |    |    |    |    |    |    |    |    |    |    |    |    |    |    |    |    |    |    |    |   |   |   |   |   |   |   |   |   |   |
| Reset | 0  | 0  | 0  | 0  | 0  | 0  | 0  | 0  | 0  | 0  | 0  | 0  | 0  | 0  | 0  | 0  | 1  | 0  | 0  | 0  | 0  | 0  | 1 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |   |

**ADCx\_MG field descriptions**

| Field             | Description   |
|-------------------|---|
| 31–16<br>Reserved | This field is reserved.<br>This read-only field is reserved and always has the value 0. |
| 15–0<br>MG        | Minus-Side Gain   |

**28.3.11 ADC Plus-Side General Calibration Value Register (ADCx\_CLPD)**

The Plus-Side General Calibration Value Registers (CLPx) contain calibration information that is generated by the calibration function. These registers contain seven calibration values of varying widths: CLP0[5:0], CLP1[6:0], CLP2[7:0], CLP3[8:0], CLP4[9:0], CLPS[5:0], and CLPD[5:0]. CLPx are automatically set when the self-calibration sequence is done, that is, CAL is cleared. If these registers are written by the user after calibration, the linearity error specifications may not be met.

## Register definition

Address: 4003\_B000h base + 34h offset = 4003\_B034h

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|-----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|---|---|---|---|---|---|---|---|---|---|
| R   |    |    |    |    |    |    |    |    |    |    |    |    |    |    |    |    | 0  |    |    |    |    |    |   |   |   |   |   |   |   |   |   |   |
| W   |    |    |    |    |    |    |    |    |    |    |    |    |    |    |    |    |    |    |    |    |    |    |   |   |   |   |   |   |   |   |   |   |

Reset 0 1 0 1 0

### ADCx\_CLPD field descriptions

| Field            | Description   |
|------------------|---|
| 31–6<br>Reserved | This field is reserved.<br>This read-only field is reserved and always has the value 0. |
| 5–0<br>CLPD      | Calibration Value   |

## 28.3.12 ADC Plus-Side General Calibration Value Register (ADCx\_CLPS)

For more information, see CLPD register description.

Address: 4003\_B000h base + 38h offset = 4003\_B038h

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|-----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|---|---|---|---|---|---|---|---|---|---|
| R   |    |    |    |    |    |    |    |    |    |    |    |    |    |    |    |    | 0  |    |    |    |    |    |   |   |   |   |   |   |   |   |   |   |
| W   |    |    |    |    |    |    |    |    |    |    |    |    |    |    |    |    |    |    |    |    |    |    |   |   |   |   |   |   |   |   |   |   |

Reset 0 1 0 0 0 0 0 0 0

### ADCx\_CLPS field descriptions

| Field            | Description   |
|------------------|---|
| 31–6<br>Reserved | This field is reserved.<br>This read-only field is reserved and always has the value 0. |
| 5–0<br>CLPS      | Calibration Value   |

## 28.3.13 ADC Plus-Side General Calibration Value Register (ADCx\_CLP4)

For more information, see CLPD register description.

Address: 4003\_B000h base + 3Ch offset = 4003\_B03Ch

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|-----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|---|---|---|---|---|---|---|---|---|---|
| R   |    |    |    |    |    |    |    |    |    |    |    |    |    |    |    |    | 0  |    |    |    |    |    |   |   |   |   |   |   |   |   |   |   |
| W   |    |    |    |    |    |    |    |    |    |    |    |    |    |    |    |    |    |    |    |    |    |    |   |   |   |   |   |   |   |   |   |   |

Reset 0 1 0 0 0 0 0 0 0

**ADCx\_CLP4 field descriptions**

| Field             | Description   |
|-------------------|---|
| 31–10<br>Reserved | This field is reserved.<br>This read-only field is reserved and always has the value 0. |
| 9–0<br>CLP4       | Calibration Value   |

**28.3.14 ADC Plus-Side General Calibration Value Register (ADCx\_CLP3)**

For more information, see CLPD register description.

Address: 4003\_B000h base + 40h offset = 4003\_B040h

| Bit   | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |  |  |  |
|-------|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|---|---|---|---|---|---|---|---|---|---|--|--|--|
| R     |    |    |    |    |    |    |    |    |    |    |    |    |    |    |    |    | 0  |    |    |    |    |    |   |   |   |   |   |   |   |   |   |   |  |  |  |
| W     |    |    |    |    |    |    |    |    |    |    |    |    |    |    |    |    |    |    |    |    |    |    |   |   |   |   |   |   |   |   |   |   |  |  |  |
| Reset | 0  | 0  | 0  | 0  | 0  | 0  | 0  | 0  | 0  | 0  | 0  | 0  | 0  | 0  | 0  | 0  | 0  | 0  | 0  | 0  | 0  | 0  | 0 | 0 | 1 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |  |  |  |

**ADCx\_CLP3 field descriptions**

| Field            | Description   |
|------------------|---|
| 31–9<br>Reserved | This field is reserved.<br>This read-only field is reserved and always has the value 0. |
| 8–0<br>CLP3      | Calibration Value   |

**28.3.15 ADC Plus-Side General Calibration Value Register (ADCx\_CLP2)**

For more information, see CLPD register description.

Address: 4003\_B000h base + 44h offset = 4003\_B044h

| Bit   | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |  |  |  |
|-------|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|---|---|---|---|---|---|---|---|---|---|--|--|--|
| R     |    |    |    |    |    |    |    |    |    |    |    |    |    |    |    |    | 0  |    |    |    |    |    |   |   |   |   |   |   |   |   |   |   |  |  |  |
| W     |    |    |    |    |    |    |    |    |    |    |    |    |    |    |    |    |    |    |    |    |    |    |   |   |   |   |   |   |   |   |   |   |  |  |  |
| Reset | 0  | 0  | 0  | 0  | 0  | 0  | 0  | 0  | 0  | 0  | 0  | 0  | 0  | 0  | 0  | 0  | 0  | 0  | 0  | 0  | 0  | 0  | 0 | 0 | 1 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |  |  |  |

**ADCx\_CLP2 field descriptions**

| Field            | Description   |
|------------------|---|
| 31–8<br>Reserved | This field is reserved.<br>This read-only field is reserved and always has the value 0. |
| 7–0<br>CLP2      | Calibration Value   |

### 28.3.16 ADC Plus-Side General Calibration Value Register (ADCx\_CLP1)

For more information, see CLPD register description.

Address: 4003 B000h base + 48h offset = 4003 B048h

## ADCx CLP1 field descriptions

| Field            | Description   |
|------------------|---|
| 31–7<br>Reserved | This field is reserved.<br>This read-only field is reserved and always has the value 0. |
| 6–0<br>CLP1      | Calibration Value   |

### 28.3.17 ADC Plus-Side General Calibration Value Register (ADCx\_CLP0)

For more information, see CLPD register description.

Address: 4003 B000h base + 4Ch offset = 4003 B04Ch

| Bit   | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |      |
|-------|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|---|---|---|---|---|---|---|---|---|---|------|
| R     |    |    |    |    |    |    |    |    |    |    |    |    |    |    |    | 0  |    |    |    |    |    |    |   |   |   |   |   |   |   |   |   |   |      |
| W     |    |    |    |    |    |    |    |    |    |    |    |    |    |    |    |    |    |    |    |    |    |    |   |   |   |   |   |   |   |   |   |   | CLP0 |
| Reset | 0  | 0  | 0  | 0  | 0  | 0  | 0  | 0  | 0  | 0  | 0  | 0  | 0  | 0  | 0  | 0  | 0  | 0  | 0  | 0  | 0  | 0  | 0 | 0 | 0 | 0 | 1 | 0 | 0 | 0 | 0 |   |      |

## ADCx CLP0 field descriptions

| Field            | Description   |
|------------------|---|
| 31–6<br>Reserved | This field is reserved.<br>This read-only field is reserved and always has the value 0. |
| 5–0<br>CLP0      | Calibration Value   |

### 28.3.18 ADC Minus-Side General Calibration Value Register (ADCx\_CLMD)

The Minus-Side General Calibration Value (CLMx) registers contain calibration information that is generated by the calibration function. These registers contain seven calibration values of varying widths: CLM0[5:0], CLM1[6:0], CLM2[7:0], CLM3[8:0], CLM4[9:0], CLMS[5:0], and CLMD[5:0]. CLMx are automatically set when the self-calibration sequence is done, that is, CAL is cleared. If these registers are written by the user after calibration, the linearity error specifications may not be met.

Address: 4003\_B000h base + 54h offset = 4003\_B054h

| Bit   | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|-------|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|---|---|---|---|---|---|---|---|---|---|
| R     |    |    |    |    |    |    |    |    |    |    |    |    |    |    |    | 0  |    |    |    |    |    |    |   |   |   |   |   |   |   |   |   |   |
| W     |    |    |    |    |    |    |    |    |    |    |    |    |    |    |    |    |    |    |    |    |    |    |   |   |   |   |   |   |   |   |   |   |
| Reset | 0  | 0  | 0  | 0  | 0  | 0  | 0  | 0  | 0  | 0  | 0  | 0  | 0  | 0  | 0  | 0  | 0  | 0  | 0  | 0  | 0  | 0  | 0 | 0 | 0 | 0 | 0 | 1 | 0 | 1 | 0 |   |

#### ADCx\_CLMD field descriptions

| Field            | Description   |
|------------------|---|
| 31–6<br>Reserved | This field is reserved.<br>This read-only field is reserved and always has the value 0. |
| 5–0<br>CLMD      | Calibration Value   |

## 28.3.20 ADC Minus-Side General Calibration Value Register (ADCx\_CLM4)

For more information, see CLMD register description.

Address: 4003\_B000h base + 5Ch offset = 4003\_B05Ch

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |  |  |
|-----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|---|---|---|---|---|---|---|---|---|---|--|--|
| R   |    |    |    |    |    |    |    |    |    |    |    |    |    |    |    | 0  |    |    |    |    |    |    |   |   |   |   |   |   |   |   |   |   |  |  |
| W   |    |    |    |    |    |    |    |    |    |    |    |    |    |    |    |    |    |    |    |    |    |    |   |   |   |   |   |   |   |   |   |   |  |  |

### ADCx\_CLM4 field descriptions

| Field             | Description   |
|-------------------|---|
| 31–10<br>Reserved | This field is reserved.<br>This read-only field is reserved and always has the value 0. |
| 9–0<br>CLM4       | Calibration Value   |

### 28.3.22 ADC Minus-Side General Calibration Value Register (ADCx\_CLM2)

For more information, see CLMD register description.

Address: 4003\_B000h base + 64h offset = 4003\_B064h

| Bit   | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |  |  |
|-------|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|---|---|---|---|---|---|---|---|---|---|--|--|
| R     |    |    |    |    |    |    |    |    |    |    |    |    |    |    |    |    | 0  |    |    |    |    |    |   |   |   |   |   |   |   |   |   |   |  |  |
| W     |    |    |    |    |    |    |    |    |    |    |    |    |    |    |    |    |    |    |    |    |    |    |   |   |   |   |   |   |   |   |   |   |  |  |
| Reset | 0  | 0  | 0  | 0  | 0  | 0  | 0  | 0  | 0  | 0  | 0  | 0  | 0  | 0  | 0  | 0  | 0  | 0  | 0  | 0  | 0  | 0  | 0 | 0 | 1 | 0 | 0 | 0 | 0 | 0 | 0 |   |  |  |

#### ADCx\_CLM2 field descriptions

| Field            | Description   |
|------------------|---|
| 31–8<br>Reserved | This field is reserved.<br>This read-only field is reserved and always has the value 0. |
| 7–0<br>CLM2      | Calibration Value   |

### 28.3.23 ADC Minus-Side General Calibration Value Register (ADCx\_CLM1)

For more information, see CLMD register description.

Address: 4003\_B000h base + 68h offset = 4003\_B068h

| Bit   | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |  |  |  |
|-------|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|---|---|---|---|---|---|---|---|---|---|--|--|--|
| R     |    |    |    |    |    |    |    |    |    |    |    |    |    |    |    |    | 0  |    |    |    |    |    |   |   |   |   |   |   |   |   |   |   |  |  |  |
| W     |    |    |    |    |    |    |    |    |    |    |    |    |    |    |    |    |    |    |    |    |    |    |   |   |   |   |   |   |   |   |   |   |  |  |  |
| Reset | 0  | 0  | 0  | 0  | 0  | 0  | 0  | 0  | 0  | 0  | 0  | 0  | 0  | 0  | 0  | 0  | 0  | 0  | 0  | 0  | 0  | 0  | 0 | 0 | 0 | 1 | 0 | 0 | 0 | 0 | 0 |   |  |  |  |

#### ADCx\_CLM1 field descriptions

| Field            | Description   |
|------------------|---|
| 31–7<br>Reserved | This field is reserved.<br>This read-only field is reserved and always has the value 0. |
| 6–0<br>CLM1      | Calibration Value   |

### 28.3.24 ADC Minus-Side General Calibration Value Register (ADCx\_CLM0)

For more information, see CLMD register description.

Address: 4003\_B000h base + 6Ch offset = 4003\_B06Ch

| Bit   | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |  |  |
|-------|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|---|---|---|---|---|---|---|---|---|---|--|--|
| R     |    |    |    |    |    |    |    |    |    |    |    |    |    |    |    | 0  |    |    |    |    |    |    |   |   |   |   |   |   |   |   |   |   |  |  |
| W     |    |    |    |    |    |    |    |    |    |    |    |    |    |    |    |    |    |    |    |    |    |    |   |   |   |   |   |   |   |   |   |   |  |  |
| Reset | 0  | 0  | 0  | 0  | 0  | 0  | 0  | 0  | 0  | 0  | 0  | 0  | 0  | 0  | 0  | 0  | 0  | 0  | 0  | 0  | 0  | 0  | 0 | 0 | 0 | 0 | 1 | 0 | 0 | 0 | 0 | 0 |  |  |

#### ADCx\_CLM0 field descriptions

| Field            | Description   |
|------------------|---|
| 31–6<br>Reserved | This field is reserved.<br>This read-only field is reserved and always has the value 0. |
| 5–0<br>CLM0      | Calibration Value   |

## 28.4 Functional description

The ADC module is disabled during reset, in Low-Power Stop mode, or when SC1n[ADCH] are all high; see the power management information for details. The module is idle when a conversion has completed and another conversion has not been initiated. When it is idle and the asynchronous clock output enable is disabled, or CFG2[ADACKEN]=0, the module is in its lowest power state. The ADC can perform an analog-to-digital conversion on any of the software selectable channels. All modes perform conversion by a successive approximation algorithm.

To meet accuracy specifications, the ADC module must be calibrated using the on-chip calibration function. See [Calibration function](#) for details on how to perform calibration.

When the conversion is completed, the result is placed in the Rn data registers. The respective SC1n[COCO] is then set and an interrupt is generated if the respective conversion complete interrupt has been enabled, or, when SC1n[AIEN]=1.

The ADC module has the capability of automatically comparing the result of a conversion with the contents of the CV1 and CV2 registers. The compare function is enabled by setting SC2[ACFE] and operates in any of the conversion modes and configurations.

The ADC module has the capability of automatically averaging the result of multiple conversions. The hardware average function is enabled by setting SC3[AVGE] and operates in any of the conversion modes and configurations.

### **NOTE**

For the chip specific modes of operation, see the power management information of this MCU.

#### **28.4.1 Clock select and divide control**

One of four clock sources can be selected as the clock source for the ADC module. This clock source is then divided by a configurable value to generate the input clock ADCK, to the module. The clock is selected from one of the following sources by means of CFG1[ADICLK].

- Bus clock. This is the default selection following reset.
- Bus clock divided by two. For higher bus clock rates, this allows a maximum divide-by-16 of the bus clock using CFG1[ADIV].
- ALTCLK: As defined for this MCU. See the chip configuration information.
- Asynchronous clock (ADACK): This clock is generated from a clock source within the ADC module. When the ADACK clock source is selected, it is not required to be active prior to conversion start. When it is selected and it is not active prior to a conversion start CFG2[ADACKEN]=0, ADACK is activated at the start of a conversion and deactivated when conversions are terminated. In this case, there is an associated clock startup delay each time the clock source is re-activated. To avoid the conversion time variability and latency associated with the ADACK clock startup, set CFG2[ADACKEN]=1 and wait the worst-case startup time of 5  $\mu$ s prior to initiating any conversions using the ADACK clock source. Conversions are possible using ADACK as the input clock source while the MCU is in Normal Stop mode. See [Power Control](#) for more information.

Whichever clock is selected, its frequency must fall within the specified frequency range for ADCK. If the available clocks are too slow, the ADC may not perform according to specifications. If the available clocks are too fast, the clock must be divided to the appropriate frequency. This divider is specified by CFG1[ADIV] and can be divide-by 1, 2, 4, or 8.

## 28.4.2 Voltage reference selection

The ADC can be configured to accept one of the two voltage reference pairs as the reference voltage ( $V_{REFSH}$  and  $V_{REFSL}$ ) used for conversions. Each pair contains a positive reference that must be between the minimum Ref Voltage High and  $V_{DDA}$ , and a ground reference that must be at the same potential as  $V_{SSA}$ . The two pairs are external ( $V_{REFH}$  and  $V_{REFL}$ ) and alternate ( $V_{ALTH}$  and  $V_{ALTL}$ ). These voltage references are selected using SC2[REFSEL]. The alternate ( $V_{ALTH}$  and  $V_{ALTL}$ ) voltage reference pair may select additional external pins or internal sources depending on MCU configuration. See the chip configuration information on the voltage references specific to this MCU.

## 28.4.3 Hardware trigger and channel selects

The ADC module has a selectable asynchronous hardware conversion trigger, ADHWT, that is enabled when SC2[ADTRG] is set and a hardware trigger select event, ADHWTSn, has occurred. This source is not available on all MCUs. See the Chip Configuration chapter for information on the ADHWT source and the ADHWTSn configurations specific to this MCU.

When an ADHWT source is available and hardware trigger is enabled, that is SC2[ADTRG]=1, a conversion is initiated on the rising-edge of ADHWT after a hardware trigger select event, that is, ADHWTSn, has occurred. If a conversion is in progress when a rising-edge of a trigger occurs, the rising-edge is ignored. In continuous convert configuration, only the initial rising-edge to launch continuous conversions is observed, and until conversion is aborted, the ADC continues to do conversions on the same SCn register that initiated the conversion. The hardware trigger function operates in conjunction with any of the conversion modes and configurations.

The hardware trigger select event, that is, ADHWTSn, must be set prior to the receipt of the ADHWT signal. If these conditions are not met, the converter may ignore the trigger or use the incorrect configuration. If a hardware trigger select event is asserted during a conversion, it must stay asserted until the end of current conversion and remain set until the receipt of the ADHWT signal to trigger a new conversion. The channel and status fields selected for the conversion depend on the active trigger select signal:

- ADHWTSA active selects SC1A
- ADHWTSn active selects SC1n

### Note

Asserting more than one hardware trigger select signal (ADHWTSn) at the same time results in unknown results. To avoid this, select only one hardware trigger select signal (ADHWTSn) prior to the next intended conversion.

When the conversion is completed, the result is placed in the Rn registers associated with the ADHWTSn received. For example:

- ADHWTSn active selects RA register
- ADHWTSn active selects Rn register

The conversion complete flag associated with the ADHWTsn received, that is, SC1n[COCO], is then set and an interrupt is generated if the respective conversion complete interrupt has been enabled, that is, SC1[AIEN]=1.

## 28.4.4 Conversion control

Conversions can be performed as determined by CFG1[MODE] and SC1n[DIFF] as shown in the description of CFG1[MODE].

Conversions can be initiated by a software or hardware trigger. In addition, the ADC module can be configured for:

- Low-power operation
- Long sample time
- Continuous conversion
- Hardware average
- Automatic compare of the conversion result to a software determined compare value

### 28.4.4.1 Initiating conversions

A conversion is initiated:

- Following a write to SC1A, with SC1n[ADCH] not all 1's, if software triggered operation is selected, that is, when SC2[ADTRG]=0.
- Following a hardware trigger, or ADHWT event, if hardware triggered operation is selected, that is, SC2[ADTRG]=1, and a hardware trigger select event, ADHWTsn, has occurred. The channel and status fields selected depend on the active trigger select signal:
  - ADHWTSn active selects SC1A
  - ADHWTSn active selects SC1n
  - if neither is active, the off condition is selected

**Note**

Selecting more than one ADHWT<sub>n</sub> prior to a conversion completion will result in unknown results. To avoid this, select only one ADHWT<sub>n</sub> prior to a conversion completion.

- Following the transfer of the result to the data registers when continuous conversion is enabled, that is, when ADCO=1.

If continuous conversions are enabled, a new conversion is automatically initiated after the completion of the current conversion, by: In software triggered operation, that is, when ADTRG=0, continuous conversions begin after SC1A is written and continue until aborted. In hardware triggered operation, that is, when ADTRG=1 and one ADHWT<sub>n</sub> event has occurred, continuous conversions begin after a hardware trigger event and continue until aborted.

If hardware averaging is enabled, a new conversion is automatically initiated after the completion of the current conversion until the correct number of conversions are completed. In software triggered operation, conversions begin after SC1A is written. In hardware triggered operation, conversions begin after a hardware trigger. If continuous conversions are also enabled, a new set of conversions to be averaged are initiated following the last of the selected number of conversions.

#### **28.4.4.2 Completing conversions**

A conversion is completed when the result of the conversion is transferred into the data result registers, R<sub>n</sub>. If the compare functions are disabled, this is indicated by setting of SC1n[COCO]. If hardware averaging is enabled, the respective SC1n[COCO] sets only if the last of the selected number of conversions is completed. If the compare function is enabled, the respective SC1n[COCO] sets and conversion result data is transferred only if the compare condition is true. If both hardware averaging and compare functions are enabled, then the respective SC1n[COCO] sets only if the last of the selected number of conversions is completed and the compare condition is true. An interrupt is generated if the respective SC1n[AIEN] is high at the time that the respective SC1n[COCO] is set.

#### **28.4.4.3 Aborting conversions**

Any conversion in progress is aborted when:

- Writing to SC1A while it is actively controlling a conversion, aborts the current conversion. In Software Trigger mode, when SC2[ADTRG]=0, a write to SC1A initiates a new conversion if SC1A[ADCH] is equal to a value other than all 1s. Writing to any of the SC1B–SC1n registers while that specific SC1B–SC1n register is actively controlling a conversion aborts the current conversion. The SC1(B-n) registers are not used for software trigger operation and therefore writes to the SC1(B-n) registers do not initiate a new conversion.
- A write to any ADC register besides the SC1A-SC1n registers occurs. This indicates that a change in mode of operation has occurred and the current conversion is therefore invalid.
- The MCU is reset or enters Low-Power Stop modes.
- The MCU enters Normal Stop mode with ADACK not enabled.

When a conversion is aborted, the contents of the data registers, R<sub>n</sub>, are not altered. The data registers continue to be the values transferred after the completion of the last successful conversion. If the conversion was aborted by a reset or Low-Power Stop modes, RA and R<sub>n</sub> return to their reset states.

#### 28.4.4.4 Power control

The ADC module remains in its idle state until a conversion is initiated. If ADACK is selected as the conversion clock source, but the asynchronous clock output is disabled, that is CFG2[ADACKEN]=0, the ADACK clock generator also remains in its idle state (disabled) until a conversion is initiated. If the asynchronous clock output is enabled, that is, CFG2[ADACKEN]=1, it remains active regardless of the state of the ADC or the MCU power mode.

Power consumption when the ADC is active can be reduced by setting CFG1[ADLPC]. This results in a lower maximum value for f<sub>ADCK</sub>.

#### 28.4.4.5 Sample time and total conversion time

For short sample, that is, when CFG1[ADLSMP]=0, there is a 2-cycle adder for first conversion over the base sample time of four ADCK cycles. For high speed conversions, that is, when CFG2[ADHSC]=1, there is an additional 2-cycle adder on any conversion. The table below summarizes sample times for the possible ADC configurations.

## Functional description

| ADC configuration |              |             | Sample time (ADCK cycles) |            |
|-------------------|--------------|-------------|---------------------------|------------|
| CFG1[ADLSMP]      | CFG2[ADLSTS] | CFG2[ADHSC] | First or Single           | Subsequent |
| 0                 | X            | 0           | 6                         | 4          |
| 1                 | 00           | 0           |                           | 24         |
| 1                 | 01           | 0           |                           | 16         |
| 1                 | 10           | 0           |                           | 10         |
| 1                 | 11           | 0           |                           | 6          |
| 0                 | X            | 1           | 8                         | 6          |
| 1                 | 00           | 1           |                           | 26         |
| 1                 | 01           | 1           |                           | 18         |
| 1                 | 10           | 1           |                           | 12         |
| 1                 | 11           | 1           |                           | 8          |

The total conversion time depends upon:

- The sample time as determined by CFG1[ADLSMP] and CFG2[ADLSTS]
- The MCU bus frequency
- The conversion mode, as determined by CFG1[MODE] and SC1n[DIFF]
- The high speed configuration, that is, CFG2[ADHSC]
- The frequency of the conversion clock, that is,  $f_{ADCK}$ .

CFG2[ADHSC] is used to configure a higher clock input frequency. This will allow faster overall conversion times. To meet internal ADC timing requirements, CFG2[ADHSC] adds additional ADCK cycles. Conversions with CFG2[ADHSC]=1 take two more ADCK cycles. CFG2[ADHSC] must be used when the ADCLK exceeds the limit for CFG2[ADHSC]=0.

After the module becomes active, sampling of the input begins.

1. CFG1[ADLSMP] and CFG2[ADLSTS] select between sample times based on the conversion mode that is selected.
2. When sampling is completed, the converter is isolated from the input channel and a successive approximation algorithm is applied to determine the digital value of the analog signal.
3. The result of the conversion is transferred to Rn upon completion of the conversion algorithm.

If the bus frequency is less than  $f_{ADCK}$ , precise sample time for continuous conversions cannot be guaranteed when short sample is enabled, that is, when CFG1[ADLSMP]=0.

The maximum total conversion time is determined by the clock source chosen and the divide ratio selected. The clock source is selectable by CFG1[ADICLK], and the divide ratio is specified by CFG1[ADIV].

The maximum total conversion time for all configurations is summarized in the equation below. See the following tables for the variables referenced in the equation.

$$\text{ConversionTime} = \text{SFCAdder} + \text{AverageNum} \times (\text{BCT} + \text{LSTAdder} + \text{HSCAdder})$$

**Figure 28-62. Conversion time equation**

**Table 28-70. Single or first continuous time adder (SFCAdder)**

| CFG1[AD LSMP] | CFG2[AD ACKEN] | CFG1[ADICLK] | Single or first continuous time adder (SFCAdder) |
|---------------|----------------|--------------|--|
| 1             | x              | 0x, 10       | 3 ADCK cycles + 5 bus clock cycles               |
| 1             | 1              | 11           | 3 ADCK cycles + 5 bus clock cycles <sup>1</sup>  |
| 1             | 0              | 11           | 5 µs + 3 ADCK cycles + 5 bus clock cycles        |
| 0             | x              | 0x, 10       | 5 ADCK cycles + 5 bus clock cycles               |
| 0             | 1              | 11           | 5 ADCK cycles + 5 bus clock cycles <sup>1</sup>  |
| 0             | 0              | 11           | 5 µs + 5 ADCK cycles + 5 bus clock cycles        |

- To achieve this time, CFG2[ADACKEN] must be 1 for at least 5 µs prior to the conversion is initiated.

**Table 28-71. Average number factor (AverageNum)**

| SC3[AVGE] | SC3[AVGS] | Average number factor (AverageNum) |
|-----------|-----------|------------------------------------|
| 0         | xx        | 1                                  |
| 1         | 00        | 4                                  |
| 1         | 01        | 8                                  |
| 1         | 10        | 16                                 |
| 1         | 11        | 32                                 |

**Table 28-72. Base conversion time (BCT)**

| Mode             | Base conversion time (BCT) |
|------------------|----------------------------|
| 8b single-ended  | 17 ADCK cycles             |
| 9b differential  | 27 ADCK cycles             |
| 10b single-ended | 20 ADCK cycles             |
| 11b differential | 30 ADCK cycles             |
| 12b single-ended | 20 ADCK cycles             |
| 13b differential | 30 ADCK cycles             |
| 16b single-ended | 25 ADCK cycles             |
| 16b differential | 34 ADCK cycles             |

**Table 28-73. Long sample time adder (LSTAdder)**

| CFG1[ADLSMP] | CFG2[ADLSTS] | Long sample time adder (LSTAdder) |
|--------------|--------------|-----------------------------------|
| 0            | xx           | 0 ADCK cycles                     |
| 1            | 00           | 20 ADCK cycles                    |
| 1            | 01           | 12 ADCK cycles                    |
| 1            | 10           | 6 ADCK cycles                     |
| 1            | 11           | 2 ADCK cycles                     |

**Table 28-74. High-speed conversion time adder (HSCAdder)**

| CFG2[ADHSC] | High-speed conversion time adder (HSCAdder) |
|-------------|---|
| 0           | 0 ADCK cycles                               |
| 1           | 2 ADCK cycles                               |

## Note

The ADCK frequency must be between  $f_{ADCK}$  minimum and  $f_{ADCK}$  maximum to meet ADC specifications.

### 28.4.4.6 Conversion time examples

The following examples use the [Figure 28-62](#), and the information provided in [Table 28-70](#) through [Table 28-74](#).

#### 28.4.4.6.1 Typical conversion time configuration

A typical configuration for ADC conversion is:

- 10-bit mode, with the bus clock selected as the input clock source
- The input clock divide-by-1 ratio selected
- Bus frequency of 8 MHz
- Long sample time disabled
- High-speed conversion disabled

The conversion time for a single conversion is calculated by using the [Figure 28-62](#), and the information provided in [Table 28-70](#) through [Table 28-74](#). The table below lists the variables of [Figure 28-62](#).

**Table 28-75. Typical conversion time**

| Variable | Time                               |
|----------|------------------------------------|
| SFCAdder | 5 ADCK cycles + 5 bus clock cycles |

*Table continues on the next page...*

**Table 28-75. Typical conversion time (continued)**

| Variable   | Time           |
|------------|----------------|
| AverageNum | 1              |
| BCT        | 20 ADCK cycles |
| LSTAdder   | 0              |
| HSCAdder   | 0              |

The resulting conversion time is generated using the parameters listed in the preceding table. Therefore, for a bus clock and an ADCK frequency equal to 8 MHz, the resulting conversion time is 3.75  $\mu$ s.

#### 28.4.4.6.2 Long conversion time configuration

A configuration for long ADC conversion is:

- 16-bit differential mode with the bus clock selected as the input clock source
- The input clock divide-by-8 ratio selected
- Bus frequency of 8 MHz
- Long sample time enabled
- Configured for longest adder
- High-speed conversion disabled
- Average enabled for 32 conversions

The conversion time for this conversion is calculated by using the [Figure 28-62](#), and the information provided in [Table 28-70](#) through [Table 28-74](#). The following table lists the variables of the [Figure 28-62](#).

**Table 28-76. Typical conversion time**

| Variable   | Time                               |
|------------|------------------------------------|
| SFCAdder   | 3 ADCK cycles + 5 bus clock cycles |
| AverageNum | 32                                 |
| BCT        | 34 ADCK cycles                     |
| LSTAdder   | 20 ADCK cycles                     |
| HSCAdder   | 0                                  |

The resulting conversion time is generated using the parameters listed in the preceding table. Therefore, for bus clock equal to 8 MHz and ADCK equal to 1 MHz, the resulting conversion time is 57.625  $\mu$ s, that is, AverageNum. This results in a total conversion time of 1.844 ms.

#### 28.4.4.6.3 Short conversion time configuration

A configuration for short ADC conversion is:

- 8-bit Single-Ended mode with the bus clock selected as the input clock source
- The input clock divide-by-1 ratio selected
- Bus frequency of 20 MHz
- Long sample time disabled
- High-speed conversion enabled

The conversion time for this conversion is calculated by using the [Figure 28-62](#), and the information provided in [Table 28-70](#) through [Table 28-74](#). The table below lists the variables of [Figure 28-62](#).

**Table 28-77. Typical conversion time**

| Variable   | Time                               |
|------------|------------------------------------|
| SFCAdder   | 5 ADCK cycles + 5 bus clock cycles |
| AverageNum | 1                                  |
| BCT        | 17 ADCK cycles                     |
| LSTAdder   | 0 ADCK cycles                      |
| HSCAdder   | 2                                  |

The resulting conversion time is generated using the parameters listed in the preceding table. Therefore, for bus clock and ADCK frequency equal to 20 MHz, the resulting conversion time is 1.45  $\mu$ s.

#### 28.4.4.7 Hardware average function

The hardware average function can be enabled by setting SC3[AVGE]=1 to perform a hardware average of multiple conversions. The number of conversions is determined by the AVGS[1:0] bits, which can select 4, 8, 16, or 32 conversions to be averaged. While the hardware average function is in progress, SC2[ADACT] will be set.

After the selected input is sampled and converted, the result is placed in an accumulator from which an average is calculated once the selected number of conversions have been completed. When hardware averaging is selected, the completion of a single conversion will not set SC1n[COCO].

If the compare function is either disabled or evaluates true, after the selected number of conversions are completed, the average conversion result is transferred into the data result registers, Rn, and SC1n[COCO] is set. An ADC interrupt is generated upon the setting of SC1n[COCO] if the respective ADC interrupt is enabled, that is, SC1n[AIEN]=1.

## Note

The hardware average function can perform conversions on a channel while the MCU is in Wait or Normal Stop modes. The ADC interrupt wakes the MCU when the hardware average is completed if SC1n[AIEN] was set.

### 28.4.5 Automatic compare function

The compare function can be configured to check whether the result is less than or greater-than-or-equal-to a single compare value, or, if the result falls within or outside a range determined by two compare values. The compare mode is determined by SC2[ACFGT], SC2[ACREN], and the values in the compare value registers, CV1 and CV2. After the input is sampled and converted, the compare values in CV1 and CV2 are used as described in the following table. There are six Compare modes as shown in the following table.

**Table 28-78. Compare modes**

| SC2[AC FGT] | SC2[AC REN] | ADCCV1 relative to ADCCV2 | Function                           | Compare mode description   |
|-------------|-------------|---------------------------|------------------------------------|--|
| 0           | 0           | —                         | Less than threshold                | Compare true if the result is less than the CV1 registers.   |
| 1           | 0           | —                         | Greater than or equal to threshold | Compare true if the result is greater than or equal to CV1 registers.  |
| 0           | 1           | Less than or equal        | Outside range, not inclusive       | Compare true if the result is less than CV1 <b>Or</b> the result is greater than CV2.                          |
| 0           | 1           | Greater than              | Inside range, not inclusive        | Compare true if the result is less than CV1 <b>And</b> the result is greater than CV2.                         |
| 1           | 1           | Less than or equal        | Inside range, inclusive            | Compare true if the result is greater than or equal to CV1 <b>And</b> the result is less than or equal to CV2. |
| 1           | 1           | Greater than              | Outside range, inclusive           | Compare true if the result is greater than or equal to CV1 <b>Or</b> the result is less than or equal to CV2.  |

If the condition selected evaluates true, SC1n[COCO] is set.

Upon completion of a conversion while the compare function is enabled, if the compare condition is not true, SC1n[COCO] is not set and the conversion result data will not be transferred to the result register, Rn. If the hardware averaging function is enabled, the compare function compares the averaged result to the compare values. The same compare function definitions apply. An ADC interrupt is generated when SC1n[COCO] is set and the respective ADC interrupt is enabled, that is, SC1n[AIEN]=1.

### Note

The compare function can monitor the voltage on a channel while the MCU is in Wait or Normal Stop modes. The ADC interrupt wakes the MCU when the compare condition is met.

## 28.4.6 Calibration function

The ADC contains a self-calibration function that is required to achieve the specified accuracy. Calibration must be run, or valid calibration values written, after any reset and before a conversion is initiated. The calibration function sets the offset calibration value, the minus-side calibration values, and the plus-side calibration values. The offset calibration value is automatically stored in the ADC offset correction register (OFS), and the plus-side and minus-side calibration values are automatically stored in the ADC plus-side and minus-side calibration registers, CLPx and CLMx. The user must configure the ADC correctly prior to calibration, and must generate the plus-side and minus-side gain calibration results and store them in the ADC plus-side gain register (PG) after the calibration function completes.

Prior to calibration, the user must configure the ADC's clock source and frequency, low power configuration, voltage reference selection, sample time, and high speed configuration according to the application's clock source availability and needs. If the application uses the ADC in a wide variety of configurations, the configuration for which the highest accuracy is required should be selected, or multiple calibrations can be done for the different configurations. For best calibration results:

- Set hardware averaging to maximum, that is, SC3[AVGE]=1 and SC3[AVGS]=11 for an average of 32
- Set ADC clock frequency  $f_{ADCk}$  less than or equal to 4 MHz
- $V_{REFH}=V_{DDA}$
- Calibrate at nominal voltage and temperature

The input channel, conversion mode continuous function, compare function, resolution mode, and differential/single-ended mode are all ignored during the calibration function.

To initiate calibration, the user sets SC3[CAL] and the calibration will automatically begin if the SC2[ADTRG] is 0. If SC2[ADTRG] is 1, SC3[CAL] will not get set and SC3[CALF] will be set. While calibration is active, no ADC register can be written and no stop mode may be entered, or the calibration routine will be aborted causing SC3[CAL] to clear and SC3[CALF] to set. At the end of a calibration sequence, SC1n[COCO] will be set. SC1n[AIEN] can be used to allow an interrupt to occur at the end of a calibration sequence. At the end of the calibration routine, if SC3[CALF] is not set, the automatic calibration routine is completed successfully.

To complete calibration, the user must generate the gain calibration values using the following procedure:

1. Initialize or clear a 16-bit variable in RAM.
2. Add the plus-side calibration results CLP0, CLP1, CLP2, CLP3, CLP4, and CLPS to the variable.
3. Divide the variable by two.
4. Set the MSB of the variable.
5. The previous two steps can be achieved by setting the carry bit, rotating to the right through the carry bit on the high byte and again on the low byte.
6. Store the value in the plus-side gain calibration register PG.
7. Repeat the procedure for the minus-side gain calibration value.

When calibration is complete, the user may reconfigure and use the ADC as desired. A second calibration may also be performed, if desired, by clearing and again setting SC3[CAL].

Overall, the calibration routine may take as many as 14k ADCK cycles and 100 bus cycles, depending on the results and the clock source chosen. For an 8 MHz clock source, this length amounts to about 1.7 ms. To reduce this latency, the calibration values, which are offset, plus-side and minus-side gain, and plus-side and minus-side calibration values, may be stored in flash memory after an initial calibration and recovered prior to the first ADC conversion. This method can reduce the calibration latency to 20 register store operations on all subsequent power, reset, or Low-Power Stop mode recoveries.

## 28.4.7 User-defined offset function

OFS contains the user-selected or calibration-generated offset error correction value. This register is a 2's complement, left-justified. The value in OFS is subtracted from the conversion and the result is transferred into the result registers, Rn. If the result is greater than the maximum or less than the minimum result value, it is forced to the appropriate limit for the current mode of operation.

The formatting of the OFS is different from the data result register, Rn, to preserve the resolution of the calibration value regardless of the conversion mode selected. Lower order bits are ignored in lower resolution modes. For example, in 8-bit single-ended mode, OFS[14:7] are subtracted from D[7:0]; OFS[15] indicates the sign (negative numbers are effectively added to the result) and OFS[6:0] are ignored. The same bits are used in 9-bit differential mode because OFS[15] indicates the sign bit, which maps to D[8]. For 16-bit differential mode, OFS[15:0] are directly subtracted from the conversion result data D[15:0]. In 16-bit single-ended mode, there is no field in the OFS corresponding to the least significant result D[0], so odd values, such as -1 or +1, cannot be subtracted from the result.

OFS is automatically set according to calibration requirements once the self-calibration sequence is done, that is, SC3[CAL] is cleared. The user may write to OFS to override the calibration result if desired. If the OFS is written by the user to a value that is different from the calibration value, the ADC error specifications may not be met. Storing the value generated by the calibration function in memory before overwriting with a user-specified value is recommended.

### Note

There is an effective limit to the values of offset that can be set by the user. If the magnitude of the offset is too high, the results of the conversions will cap off at the limits.

The offset calibration function may be employed by the user to remove application offsets or DC bias values. OFS may be written with a number in 2's complement format and this offset will be subtracted from the result, or hardware averaged value. To add an offset, store the negative offset in 2's complement format and the effect will be an addition. An offset correction that results in an out-of-range value will be forced to the minimum or maximum value. The minimum value for single-ended conversions is 0x0000; for a differential conversion it is 0x8000.

To preserve accuracy, the calibrated offset value initially stored in OFS must be added to the user-defined offset. For applications that may change the offset repeatedly during operation, store the initial offset calibration value in flash so it can be recovered and added to any user offset adjustment value and the sum stored in OFS.

## 28.4.8 Temperature sensor

The ADC module includes a temperature sensor whose output is connected to one of the ADC analog channel inputs. The following equation provides an approximate transfer function of the temperature sensor.

$$\text{Temp} = 25 - \left( \left( V_{\text{TEMP}} - V_{\text{TEMP25}} \right) \div m \right)$$

**Figure 28-63. Approximate transfer function of the temperature sensor**

where:

- $V_{\text{TEMP}}$  is the voltage of the temperature sensor channel at the ambient temperature.
- $V_{\text{TEMP25}}$  is the voltage of the temperature sensor channel at 25 °C.
- $m$  is referred as temperature sensor slope in the device data sheet. It is the hot or cold voltage versus temperature slope in V/°C.

For temperature calculations, use the  $V_{\text{TEMP25}}$  and temperature sensor slope values from the ADC Electricals table.

In application code, the user reads the temperature sensor channel, calculates  $V_{\text{TEMP}}$ , and compares to  $V_{\text{TEMP25}}$ . If  $V_{\text{TEMP}}$  is greater than  $V_{\text{TEMP25}}$  the cold slope value is applied in the preceding equation. If  $V_{\text{TEMP}}$  is less than  $V_{\text{TEMP25}}$ , the hot slope value is applied in the preceding equation. ADC Electricals table may only specify one temperature sensor slope value. In that case, the user could use the same slope for the calculation across the operational temperature range.

For more information on using the temperature sensor, see the application note titled *Temperature Sensor for the HCS08 Microcontroller Family* (document AN3031).

## 28.4.9 MCU wait mode operation

Wait mode is a lower-power consumption Standby mode from which recovery is fast because the clock sources remain active. If a conversion is in progress when the MCU enters Wait mode, it continues until completion. Conversions can be initiated while the MCU is in Wait mode by means of the hardware trigger or if continuous conversions are enabled.

The bus clock, bus clock divided by two, and ADACK are available as conversion clock sources while in Wait mode. The use of ALTCLK as the conversion clock source in Wait is dependent on the definition of ALTCLK for this MCU. See the Chip Configuration information on ALTCLK specific to this MCU.

If the compare and hardware averaging functions are disabled, a conversion complete event sets SC1n[COCO] and generates an ADC interrupt to wake the MCU from Wait mode if the respective ADC interrupt is enabled, that is, when SC1n[AIEN]=1. If the hardware averaging function is enabled, SC1n[COCO] will set, and generate an interrupt if enabled, when the selected number of conversions are completed. If the compare function is enabled, SC1n[COCO] will set, and generate an interrupt if enabled, only if the compare conditions are met. If a single conversion is selected and the compare trigger is not met, the ADC will return to its idle state and cannot wake the MCU from Wait mode unless a new conversion is initiated by the hardware trigger.

## **28.4.10 MCU Normal Stop mode operation**

Stop mode is a low-power consumption Standby mode during which most or all clock sources on the MCU are disabled.

### **28.4.10.1 Normal Stop mode with ADACK disabled**

If the asynchronous clock, ADACK, is not selected as the conversion clock, executing a stop instruction aborts the current conversion and places the ADC in its Idle state. The contents of the ADC registers, including Rn, are unaffected by Normal Stop mode. After exiting from Normal Stop mode, a software or hardware trigger is required to resume conversions.

### **28.4.10.2 Normal Stop mode with ADACK enabled**

If ADACK is selected as the conversion clock, the ADC continues operation during Normal Stop mode. See the chip configuration chapter for configuration information for this MCU.

If a conversion is in progress when the MCU enters Normal Stop mode, it continues until completion. Conversions can be initiated while the MCU is in Normal Stop mode by means of the hardware trigger or if continuous conversions are enabled.

If the compare and hardware averaging functions are disabled, a conversion complete event sets SC1n[COCO] and generates an ADC interrupt to wake the MCU from Normal Stop mode if the respective ADC interrupt is enabled, that is, when SC1n[AIEN]=1. The result register, R<sub>n</sub>, will contain the data from the first completed conversion that occurred during Normal Stop mode. If the hardware averaging function is enabled, SC1n[COCO] will set, and generate an interrupt if enabled, when the selected number of conversions are completed. If the compare function is enabled, SC1n[COCO] will set, and generate an interrupt if enabled, only if the compare conditions are met. If a single conversion is selected and the compare is not true, the ADC will return to its Idle state and cannot wake the MCU from Normal Stop mode unless a new conversion is initiated by another hardware trigger.

### 28.4.11 MCU Low-Power Stop mode operation

The ADC module is automatically disabled when the MCU enters Low-Power Stop mode. All module registers contain their reset values following exit from Low-Power Stop mode. Therefore, the module must be re-enabled and re-configured following exit from Low-Power Stop mode.

#### NOTE

For the chip specific modes of operation, see the power management information for the device.

## 28.5 Initialization information

This section gives an example that provides some basic direction on how to initialize and configure the ADC module. The user can configure the module for 16-bit, 12-bit, 10-bit, or 8-bit single-ended resolution or 16-bit, 13-bit, 11-bit, or 9-bit differential resolution, single or continuous conversion, and a polled or interrupt approach, among many other options. For information used in this example, refer to [Table 28-73](#), [Table 28-74](#), and [Table 28-75](#).

#### Note

Hexadecimal values are designated by a preceding 0x, binary values designated by a preceding %, and decimal values have no preceding character.

### 28.5.1 ADC module initialization example

### 28.5.1.1 Initialization sequence

Before the ADC module can be used to complete conversions, an initialization procedure must be performed. A typical sequence is:

1. Calibrate the ADC by following the calibration instructions in [Calibration function](#).
2. Update CFG to select the input clock source and the divide ratio used to generate ADCK. This register is also used for selecting sample time and low-power configuration.
3. Update SC2 to select the conversion trigger, hardware or software, and compare function options, if enabled.
4. Update SC3 to select whether conversions will be continuous or completed only once (ADCO) and whether to perform hardware averaging.
5. Update SC1:SC1n registers to select whether conversions will be single-ended or differential and to enable or disable conversion complete interrupts. Also, select the input channel which can be used to perform conversions.

### 28.5.1.2 Pseudo-code example

In this example, the ADC module is set up with interrupts enabled to perform a single 10-bit conversion at low-power with a long sample time on input channel 1, where ADCK is derived from the bus clock divided by 1.

**CFG1 = 0x98 (%10011000)**

|         |        |    |   |
|---------|--------|----|---|
| Bit 7   | ADLPC  | 1  | Configures for low power, lowers maximum clock speed.                       |
| Bit 6:5 | ADIV   | 00 | Sets the ADCK to the input clock $\div 1$ .                                 |
| Bit 4   | ADLSMP | 1  | Configures for long sample time.  |
| Bit 3:2 | MODE   | 10 | Selects the single-ended 10-bit conversion, differential 11-bit conversion. |
| Bit 1:0 | ADICLK | 00 | Selects the bus clock.  |

**SC2 = 0x00 (%00000000)**

|         |        |    |  |
|---------|--------|----|--|
| Bit 7   | ADACT  | 0  | Flag indicates if a conversion is in progress.   |
| Bit 6   | ADTRG  | 0  | Software trigger selected.   |
| Bit 5   | ACFE   | 0  | Compare function disabled.   |
| Bit 4   | ACFGT  | 0  | Not used in this example.  |
| Bit 3   | ACREN  | 0  | Compare range disabled.  |
| Bit 2   | DMAEN  | 0  | DMA request disabled.  |
| Bit 1:0 | REFSEL | 00 | Selects default voltage reference pin pair (External pins V <sub>REFH</sub> and V <sub>REFL</sub> ). |

**SC1A = 0x41 (%01000001)**

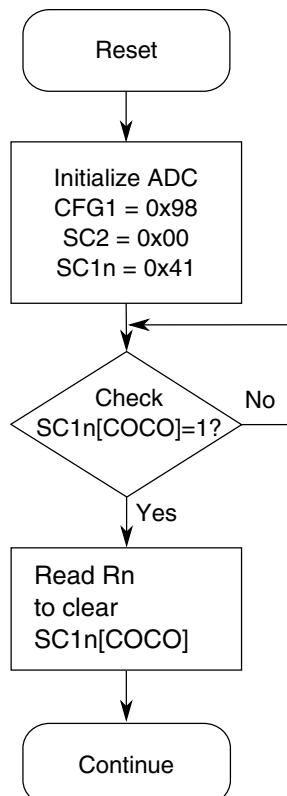
|         |      |       |  |
|---------|------|-------|--|
| Bit 7   | COCO | 0     | Read-only flag which is set when a conversion completes. |
| Bit 6   | AIEN | 1     | Conversion complete interrupt enabled.                   |
| Bit 5   | DIFF | 0     | Single-ended conversion selected.                        |
| Bit 4:0 | ADCH | 00001 | Input channel 1 selected as ADC input channel.           |

**RA = 0xxx**

Holds results of conversion.

**CV = 0xxx**

Holds compare value when compare function enabled.

**Figure 28-64. Initialization flowchart example**

## 28.6 Application information

The ADC has been designed to be integrated into a microcontroller for use in embedded control applications requiring an ADC.

### 28.6.1 External pins and routing

### 28.6.1.1 Analog supply pins

Depending on the device, the analog power and ground supplies,  $V_{DDA}$  and  $V_{SSA}$ , of the ADC module are available as:

- $V_{DDA}$  and  $V_{SSA}$  available as separate pins—When available on a separate pin, both  $V_{DDA}$  and  $V_{SSA}$  must be connected to the same voltage potential as their corresponding MCU digital supply,  $V_{DD}$  and  $V_{SS}$ , and must be routed carefully for maximum noise immunity and bypass capacitors placed as near as possible to the package.
- $V_{SSA}$  is shared on the same pin as the MCU digital  $V_{SS}$ .
- $V_{SSA}$  and  $V_{DDA}$  are shared with the MCU digital supply pins—in these cases, there are separate pads for the analog supplies bonded to the same pin as the corresponding digital supply so that some degree of isolation between the supplies is maintained.

If separate power supplies are used for analog and digital power, the ground connection between these supplies must be at the  $V_{SSA}$  pin. This must be the only ground connection between these supplies, if possible.  $V_{SSA}$  makes a good single point ground location.

### 28.6.1.2 Analog voltage reference pins

In addition to the analog supplies, the ADC module has connections for two reference voltage inputs used by the converter:

- $V_{REFSH}$  is the high reference voltage for the converter.
- $V_{REFSL}$  is the low reference voltage for the converter.

The ADC can be configured to accept one of two voltage reference pairs for  $V_{REFSH}$  and  $V_{REFSL}$ . Each pair contains a positive reference and a ground reference. The two pairs are external,  $V_{REFH}$  and  $V_{REFL}$  and alternate,  $V_{ALTH}$  and  $V_{ALTL}$ . These voltage references are selected using SC2[REFSEL]. The alternate voltage reference pair,  $V_{ALTH}$  and  $V_{ALTL}$ , may select additional external pins or internal sources based on MCU configuration. See the chip configuration information on the voltage references specific to this MCU.

In some packages, the external or alternate pairs are connected in the package to  $V_{DDA}$  and  $V_{SSA}$ , respectively. One of these positive references may be shared on the same pin as  $V_{DDA}$  on some devices. One of these ground references may be shared on the same pin as  $V_{SSA}$  on some devices.

If externally available, the positive reference may be connected to the same potential as  $V_{DDA}$  or may be driven by an external source to a level between the minimum Ref Voltage High and the  $V_{DDA}$  potential. The positive reference must never exceed  $V_{DDA}$ . If externally available, the ground reference must be connected to the same voltage potential as  $V_{SSA}$ . The voltage reference pairs must be routed carefully for maximum noise immunity and bypass capacitors placed as near as possible to the package.

AC current in the form of current spikes required to supply charge to the capacitor array at each successive approximation step is drawn through the  $V_{REFH}$  and  $V_{REFL}$  loop. The best external component to meet this current demand is a 0.1  $\mu$ F capacitor with good high-frequency characteristics. This capacitor is connected between  $V_{REFH}$  and  $V_{REFL}$  and must be placed as near as possible to the package pins. Resistance in the path is not recommended because the current causes a voltage drop that could result in conversion errors. Inductance in this path must be minimum, that is, parasitic only.

### 28.6.1.3 Analog input pins

The external analog inputs are typically shared with digital I/O pins on MCU devices.

Empirical data shows that capacitors on the analog inputs improve performance in the presence of noise or when the source impedance is high. Use of 0.01  $\mu$ F capacitors with good high-frequency characteristics is sufficient. These capacitors are not necessary in all cases, but when used, they must be placed as near as possible to the package pins and be referenced to  $V_{SSA}$ .

For proper conversion, the input voltage must fall between  $V_{REFH}$  and  $V_{REFL}$ . If the input is equal to or exceeds  $V_{REFH}$ , the converter circuit converts the signal to 0xFFFF, which is full scale 12-bit representation, 0x3FF, which is full scale 10-bit representation, or 0xFF, which is full scale 8-bit representation. If the input is equal to or less than  $V_{REFL}$ , the converter circuit converts it to 0x000. Input voltages between  $V_{REFH}$  and  $V_{REFL}$  are straight-line linear conversions. There is a brief current associated with  $V_{REFL}$  when the sampling capacitor is charging.

For minimal loss of accuracy due to current injection, pins adjacent to the analog input pins must not be transitioning during conversions.

## 28.6.2 Sources of error

### 28.6.2.1 Sampling error

For proper conversions, the input must be sampled long enough to achieve the proper accuracy.

$$RAS + RADIN = SC / (FMAX * NUMTAU * CADIN)$$

**Figure 28-65. Sampling equation**

Where:

RAS = External analog source resistance

SC = Number of ADCK cycles used during sample window

CADIN = Internal ADC input capacitance

NUMTAU =  $-\ln(\text{LSBERR} / 2^N)$

LSBERR = value of acceptable sampling error in LSBs

N = 8 in 8-bit mode, 10 in 10-bit mode, 12 in 12-bit mode or 16 in 16-bit mode

Higher source resistances or higher-accuracy sampling is possible by setting CFG1[ADLSMP] and changing CFG2[ADLSTS] to increase the sample window, or decreasing ADCK frequency to increase sample time.

### 28.6.2.2 Pin leakage error

Leakage on the I/O pins can cause conversion error if the external analog source resistance,  $R_{AS}$ , is high. If this error cannot be tolerated by the application, keep  $R_{AS}$  lower than  $V_{REFH} / (4 \times I_{LEAK} \times 2^N)$  for less than 1/4 LSB leakage error, where N = 8 in 8-bit mode, 10 in 10-bit mode, 12 in 12-bit mode, or 16 in 16-bit mode.

### 28.6.2.3 Noise-induced errors

System noise that occurs during the sample or conversion process can affect the accuracy of the conversion. The ADC accuracy numbers are guaranteed as specified only if the following conditions are met:

- There is a 0.1  $\mu\text{F}$  low-ESR capacitor from  $V_{REFH}$  to  $V_{REFL}$ .
- There is a 0.1  $\mu\text{F}$  low-ESR capacitor from  $V_{DDA}$  to  $V_{SSA}$ .
- If inductive isolation is used from the primary supply, an additional 1  $\mu\text{F}$  capacitor is placed from  $V_{DDA}$  to  $V_{SSA}$ .
- $V_{SSA}$ , and  $V_{REFL}$ , if connected, is connected to  $V_{SS}$  at a quiet point in the ground plane.
- Operate the MCU in Wait or Normal Stop mode before initiating (hardware-triggered conversions) or immediately after initiating (hardware- or software-triggered conversions) the ADC conversion.

- For software triggered conversions, immediately follow the write to SC1 with a Wait instruction or Stop instruction.
- For Normal Stop mode operation, select ADACK as the clock source. Operation in Normal Stop reduces  $V_{DD}$  noise but increases effective conversion time due to stop recovery.
- There is no I/O switching, input or output, on the MCU during the conversion.

There are some situations where external system activity causes radiated or conducted noise emissions or excessive  $V_{DD}$  noise is coupled into the ADC. In these situations, or when the MCU cannot be placed in Wait or Normal Stop mode, or I/O activity cannot be halted, the following actions may reduce the effect of noise on the accuracy:

- Place a 0.01  $\mu F$  capacitor ( $C_{AS}$ ) on the selected input channel to  $V_{REFL}$  or  $V_{SSA}$ . This improves noise issues, but affects the sample rate based on the external analog source resistance.
- Average the result by converting the analog input many times in succession and dividing the sum of the results. Four samples are required to eliminate the effect of a 1 LSB, one-time error.
- Reduce the effect of synchronous noise by operating off the asynchronous clock, that is, ADACK, and averaging. Noise that is synchronous to ADCK cannot be averaged out.

#### 28.6.2.4 Code width and quantization error

The ADC quantizes the ideal straight-line transfer function into 65536 steps in the 16-bit mode). Each step ideally has the same height, that is, 1 code, and width. The width is defined as the delta between the transition points to one code and the next. The ideal code width for an N-bit converter, where N can be 16, 12, 10, or 8, defined as 1 LSB, is:

$$1\text{LSB} = (V_{REFH}) / 2^N$$

**Figure 28-66. Ideal code width for an N-bit converter**

There is an inherent quantization error due to the digitization of the result. For 8-bit, 10-bit, or 12-bit conversions, the code transitions when the voltage is at the midpoint between the points where the straight line transfer function is exactly represented by the actual transfer function. Therefore, the quantization error will be  $\pm 1/2$  LSB in 8-bit, 10-bit, or 12-bit modes. As a consequence, however, the code width of the first (0x000) conversion is only 1/2 LSB and the code width of the last (0xFF or 0x3FF) is 1.5 LSB.

For 16-bit conversions, the code transitions only after the full code width is present, so the quantization error is -1 LSB to 0 LSB and the code width of each step is 1 LSB.

### **28.6.2.5 Linearity errors**

The ADC may also exhibit non-linearity of several forms. Every effort has been made to reduce these errors, but the system designers must be aware of these errors because they affect overall accuracy:

- Zero-scale error ( $E_{ZS}$ ), sometimes called offset: This error is defined as the difference between the actual code width of the first conversion and the ideal code width. This is 1/2 LSB in 8-bit, 10-bit, or 12-bit modes and 1 LSB in 16-bit mode. If the first conversion is 0x001, the difference between the actual 0x001 code width and its ideal (1 LSB) is used.
- Full-scale error ( $E_{FS}$ ): This error is defined as the difference between the actual code width of the last conversion and the ideal code width. This is 1.5 LSB in 8-bit, 10-bit, or 12-bit modes and 1 LSB in 16-bit mode. If the last conversion is 0x3FE, the difference between the actual 0x3FE code width and its ideal (1 LSB) is used.
- Differential non-linearity (DNL): This error is defined as the worst-case difference between the actual code width and the ideal code width for all conversions.
- Integral non-linearity (INL): This error is defined as the highest-value or absolute value that the running sum of DNL achieves. More simply, this is the worst-case difference of the actual transition voltage to a given code and its corresponding ideal transition voltage, for all codes.
- Total unadjusted error (TUE): This error is defined as the difference between the actual transfer function and the ideal straight-line transfer function and includes all forms of error.

### **28.6.2.6 Code jitter, non-monotonicity, and missing codes**

Analog-to-digital converters are susceptible to three special forms of error:

- Code jitter: Code jitter is when, at certain points, a given input voltage converts to one of the two values when sampled repeatedly. Ideally, when the input voltage is infinitesimally smaller than the transition voltage, the converter yields the lower code, and vice-versa. However, even small amounts of system noise can cause the converter to be indeterminate, between two codes, for a range of input voltages around the transition voltage.

This error may be reduced by repeatedly sampling the input and averaging the result. Additionally, the techniques discussed in [Noise-induced errors](#) reduces this error.

- Non-monotonicity: Non-monotonicity is defined as when, except for code jitter, the converter converts to a lower code for a higher input voltage.
- Missing codes: Missing codes are those values never converted for any input value.

In 8-bit or 10-bit mode, the ADC is guaranteed to be monotonic and have no missing codes.



# Chapter 29

## Comparator (CMP)

### 29.1 Introduction

#### NOTE

For the chip-specific implementation details of this module's instances see the chip configuration information.

The comparator (CMP) module provides a circuit for comparing two analog input voltages. The comparator circuit is designed to operate across the full range of the supply voltage, known as rail-to-rail operation.

The Analog MUX (ANMUX) provides a circuit for selecting an analog input signal from eight channels. One signal is provided by the 6-bit digital-to-analog converter (DAC). The mux circuit is designed to operate across the full range of the supply voltage.

The 6-bit DAC is 64-tap resistor ladder network which provides a selectable voltage reference for applications where voltage reference is needed. The 64-tap resistor ladder network divides the supply reference  $V_{in}$  into 64 voltage levels. A 6-bit digital signal input selects the output voltage level, which varies from  $V_{in}$  to  $V_{in}/64$ .  $V_{in}$  can be selected from two voltage sources,  $V_{in1}$  and  $V_{in2}$ . The 6-bit DAC from a comparator is available as an on-chip internal signal only and is not available externally to a pin.

### 29.2 CMP features

The CMP has the following features:

- Operational over the entire supply range
- Inputs may range from rail to rail
- Programmable hysteresis control

### 6-bit DAC key features

- Selectable interrupt on rising-edge, falling-edge, or both rising or falling edges of the comparator output
- Selectable inversion on comparator output
- Capability to produce a wide range of outputs such as:
  - Sampled
  - Windowed, which is ideal for certain PWM zero-crossing-detection applications
  - Digitally filtered:
    - Filter can be bypassed
    - Can be clocked via external SAMPLE signal or scaled bus clock
- External hysteresis can be used at the same time that the output filter is used for internal functions
- Two software selectable performance levels:
  - Shorter propagation delay at the expense of higher power
  - Low power, with longer propagation delay
- DMA transfer support
  - A comparison event can be selected to trigger a DMA transfer
- Functional in all modes of operation
- The window and filter functions are not available in the following modes:
  - Stop
  - VLPS
  - LLS
  - VLLSx

## 29.3 6-bit DAC key features

- 6-bit resolution
- Selectable supply reference source
- Power Down mode to conserve power when not in use
- Option to route the output to internal comparator input

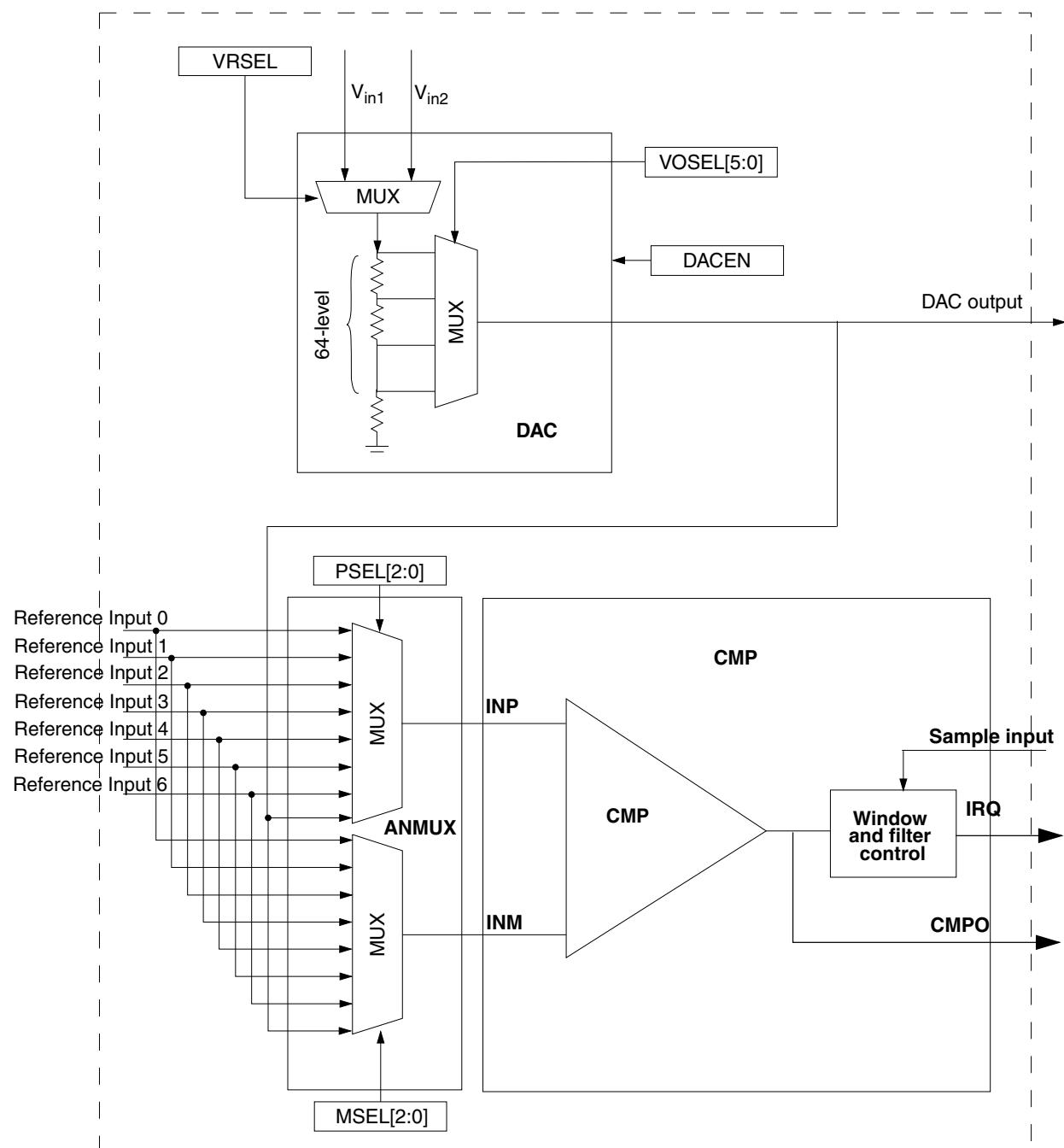
## 29.4 ANMUX key features

- Two 8-to-1 channel mux
- Operational over the entire supply range

## 29.5 CMP, DAC and ANMUX diagram

The following figure shows the block diagram for the High-Speed Comparator, DAC, and ANMUX modules.

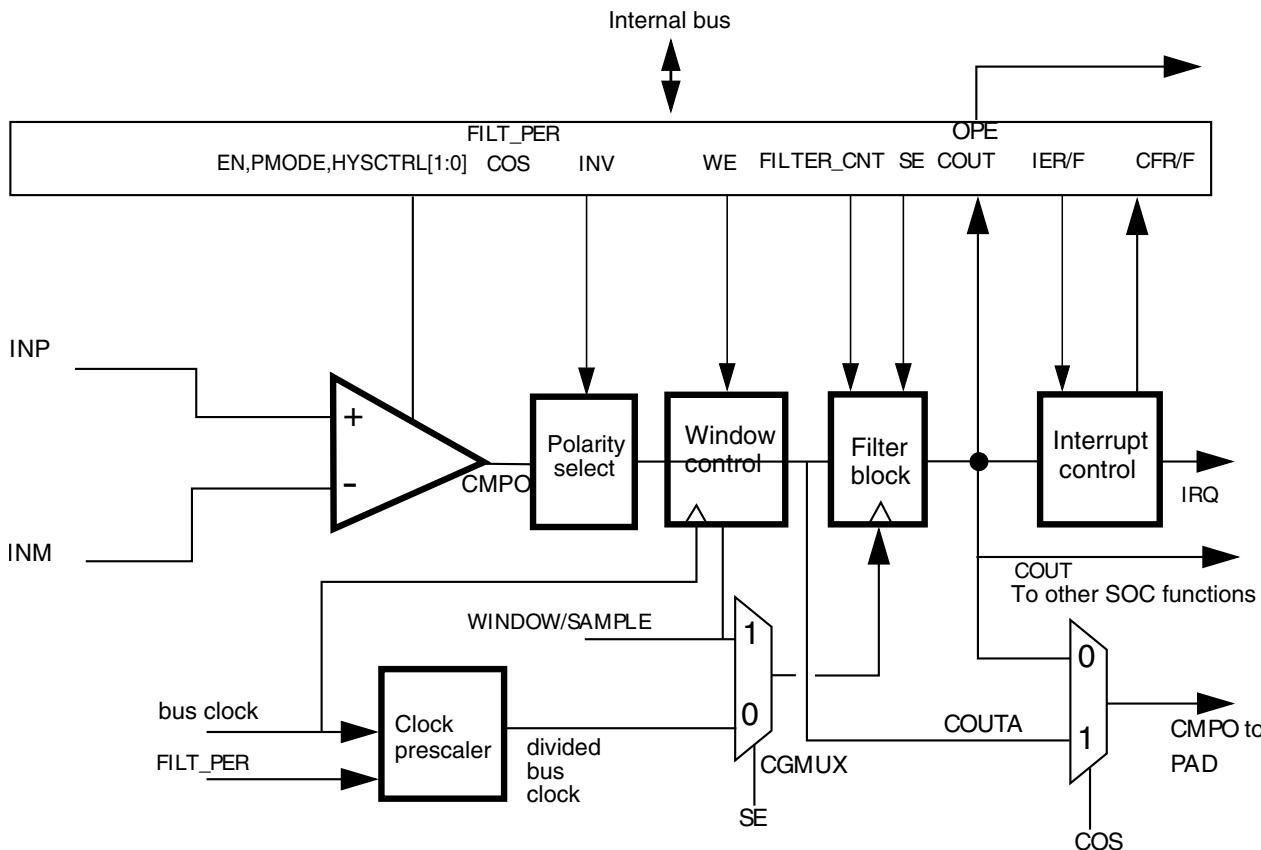
## CMP block diagram



**Figure 29-1. CMP, DAC and ANMUX block diagram**

## 29.6 CMP block diagram

The following figure shows the block diagram for the CMP module.

**Figure 29-2. Comparator module block diagram**

In the CMP block diagram:

- The Window Control block is bypassed when CR1[WE] = 0
- If CR1[WE] = 1, the comparator output will be sampled on every bus clock when WINDOW=1 to generate COUTA. Sampling does NOT occur when WINDOW = 0.
- The Filter block is bypassed when not in use.
- The Filter block acts as a simple sampler if the filter is bypassed and CR0[FILTER\_CNT] is set to 0x01.
- The Filter block filters based on multiple samples when the filter is bypassed and CR0[FILTER\_CNT] is set greater than 0x01.
  - If CR1[SE] = 1, the external SAMPLE input is used as sampling clock
  - IF CR1[SE] = 0, the divided bus clock is used as sampling clock

## Memory map/register definitions

- If enabled, the Filter block will incur up to one bus clock additional latency penalty on COUT due to the fact that COUT, which is crossing clock domain boundaries, must be resynchronized to the bus clock.
- CR1[WE] and CR1[SE] are mutually exclusive.

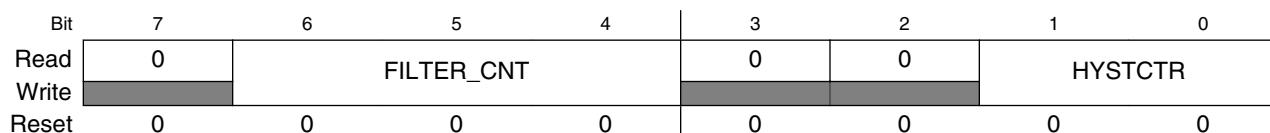
## 29.7 Memory map/register definitions

CMP memory map

| Absolute address (hex) | Register name                              | Width (in bits) | Access | Reset value | Section/page |
|------------------------|--|-----------------|--------|-------------|--------------|
| 4007_3000              | CMP Control Register 0 (CMP0_CR0)          | 8               | R/W    | 00h         | 29.7.1/514   |
| 4007_3001              | CMP Control Register 1 (CMP0_CR1)          | 8               | R/W    | 00h         | 29.7.2/515   |
| 4007_3002              | CMP Filter Period Register (CMP0_FPR)      | 8               | R/W    | 00h         | 29.7.3/517   |
| 4007_3003              | CMP Status and Control Register (CMP0_SCR) | 8               | R/W    | 00h         | 29.7.4/517   |
| 4007_3004              | DAC Control Register (CMP0_DACC0)          | 8               | R/W    | 00h         | 29.7.5/518   |
| 4007_3005              | MUX Control Register (CMP0_MUXCR)          | 8               | R/W    | 00h         | 29.7.6/519   |

### 29.7.1 CMP Control Register 0 (CMPx\_CR0)

Address: 4007\_3000h base + 0h offset = 4007\_3000h



CMPx\_CR0 field descriptions

| Field             | Description  |
|-------------------|--|
| 7<br>Reserved     | This field is reserved.<br>This read-only field is reserved and always has the value 0.  |
| 6–4<br>FILTER_CNT | Filter Sample Count<br><br>Represents the number of consecutive samples that must agree prior to the comparator output filter accepting a new output state. For information regarding filter programming and latency, see the <a href="#">Functional description</a> .<br><br>000 Filter is disabled. If SE = 1, then COUT is a logic 0. This is not a legal state, and is not recommended. If SE = 0, COUT = COUTA.<br>001 One sample must agree. The comparator output is simply sampled.<br>010 2 consecutive samples must agree.<br>011 3 consecutive samples must agree.<br>100 4 consecutive samples must agree. |

Table continues on the next page...

**CMPx\_CR0 field descriptions (continued)**

| Field       | Description   |
|-------------|---|
|             | 101 5 consecutive samples must agree.<br>110 6 consecutive samples must agree.<br>111 7 consecutive samples must agree.   |
| 3 Reserved  | This field is reserved.<br>This read-only field is reserved and always has the value 0.   |
| 2 Reserved  | This field is reserved.<br>This read-only field is reserved and always has the value 0.   |
| 1–0 HYSTCTR | Comparator hard block hysteresis control<br><br>Defines the programmable hysteresis level. The hysteresis values associated with each level are device-specific. See the Data Sheet of the device for the exact values.<br><br>00 Level 0<br>01 Level 1<br>10 Level 2<br>11 Level 3 |

**29.7.2 CMP Control Register 1 (CMPx\_CR1)**

Address: 4007\_3000h base + 1h offset = 4007\_3001h

| Bit        | 7  | 6  | 5     | 4     | 3   | 2   | 1   | 0  |
|------------|----|----|-------|-------|-----|-----|-----|----|
| Read Write | SE | WE | TRIGM | PMODE | INV | COS | OPE | EN |
| Reset      | 0  | 0  | 0     | 0     | 0   | 0   | 0   | 0  |

**CMPx\_CR1 field descriptions**

| Field   | Description   |
|---------|---|
| 7 SE    | Sample Enable<br><br>At any given time, either SE or WE can be set. It is mandatory request to not set SE and WE both at a given time.<br><br>0 Sampling mode is not selected.<br>1 Sampling mode is selected.                    |
| 6 WE    | Windowing Enable<br><br>At any given time, either SE or WE can be set. It is mandatory request to not set SE and WE both at a given time.<br><br>0 Windowing mode is not selected.<br>1 Windowing mode is selected.               |
| 5 TRIGM | Trigger Mode Enable<br><br>CMP and DAC are configured to CMP Trigger mode when CMP_CR1[TRIGM] is set to 1. In addition, the CMP should be enabled. If the DAC is to be used as a reference to the CMP, it should also be enabled. |

Table continues on the next page...

**CMPx\_CR1 field descriptions (continued)**

| Field   | Description  |
|---------|--|
|         | <p>CMP Trigger mode depends on an external timer resource to periodically enable the CMP and 6-bit DAC in order to generate a triggered compare.</p> <p>Upon setting TRIGM, the CMP and DAC are placed in a standby state until an external timer resource trigger is received.</p> <p>See the chip configuration chapter for details about the external timer resource.</p> <ul style="list-style-type: none"> <li>0 Trigger mode is disabled.</li> <li>1 Trigger mode is enabled.</li> </ul> |
| 4 PMODE | <p>Power Mode Select</p> <p>See the electrical specifications table in the device Data Sheet for details.</p> <ul style="list-style-type: none"> <li>0 Low-Speed (LS) Comparison mode selected. In this mode, CMP has slower output propagation delay and lower current consumption.</li> <li>1 High-Speed (HS) Comparison mode selected. In this mode, CMP has faster output propagation delay and higher current consumption.</li> </ul>   |
| 3 INV   | <p>Comparator INVERT</p> <p>Allows selection of the polarity of the analog comparator function. It is also driven to the COUT output, on both the device pin and as SCR[COUT], when OPE=0.</p> <ul style="list-style-type: none"> <li>0 Does not invert the comparator output.</li> <li>1 Inverts the comparator output.</li> </ul>  |
| 2 COS   | <p>Comparator Output Select</p> <ul style="list-style-type: none"> <li>0 Set the filtered comparator output (CMPO) to equal COUT.</li> <li>1 Set the unfiltered comparator output (CMPO) to equal COUTA.</li> </ul>  |
| 1 OPE   | <p>Comparator Output Pin Enable</p> <ul style="list-style-type: none"> <li>0 CMPO is not available on the associated CMPO output pin. If the comparator does not own the pin, this field has no effect.</li> <li>1 CMPO is available on the associated CMPO output pin.</li> </ul> <p>The comparator output (CMPO) is driven out on the associated CMPO output pin if the comparator owns the pin. If the comparator does not own the field, this bit has no effect.</p>                       |
| 0 EN    | <p>Comparator Module Enable</p> <p>Enables the Analog Comparator module. When the module is not enabled, it remains in the off state, and consumes no power. When the user selects the same input from analog mux to the positive and negative port, the comparator is disabled automatically.</p> <ul style="list-style-type: none"> <li>0 Analog Comparator is disabled.</li> <li>1 Analog Comparator is enabled.</li> </ul>   |

### 29.7.3 CMP Filter Period Register (CMPx\_FPR)

Address: 4007\_3000h base + 2h offset = 4007\_3002h

| Bit   | 7 | 6 | 5 | 4 |          | 3 | 2 | 1 | 0 |
|-------|---|---|---|---|----------|---|---|---|---|
| Read  | 0 | 0 | 0 | 0 | FILT_PER | 0 | 0 | 0 | 0 |
| Write |   |   |   |   |          |   |   |   |   |
| Reset | 0 | 0 | 0 | 0 | 0        | 0 | 0 | 0 | 0 |

#### CMPx\_FPR field descriptions

| Field           | Description  |
|-----------------|--|
| 7–0<br>FILT_PER | <p>Filter Sample Period</p> <p>Specifies the sampling period, in bus clock cycles, of the comparator output filter, when CR1[SE]=0. Setting FILT_PER to 0x0 disables the filter. Filter programming and latency details appear in the <a href="#">Functional description</a>.</p> <p>This field has no effect when CR1[SE]=1. In that case, the external SAMPLE signal is used to determine the sampling period.</p> |

### 29.7.4 CMP Status and Control Register (CMPx\_SCR)

Address: 4007\_3000h base + 3h offset = 4007\_3003h

| Bit   | 7 | 6     | 5 | 4   |     | 3   | 2   | 1    | 0 |
|-------|---|-------|---|-----|-----|-----|-----|------|---|
| Read  | 0 | DMAEN | 0 | IER | IEF | CFR | CFF | COUT |   |
| Write |   |       |   |     | w1c | w1c |     |      |   |
| Reset | 0 | 0     | 0 | 0   | 0   | 0   | 0   | 0    | 0 |

#### CMPx\_SCR field descriptions

| Field         | Description  |
|---------------|--|
| 7<br>Reserved | This field is reserved.<br>This read-only field is reserved and always has the value 0.  |
| 6<br>DMAEN    | <p>DMA Enable Control</p> <p>Enables the DMA transfer triggered from the CMP module. When this field is set, a DMA request is asserted when CFR or CFF is set.</p> <p>0 DMA is disabled.<br/>1 DMA is enabled.</p> |
| 5<br>Reserved | This field is reserved.<br>This read-only field is reserved and always has the value 0.  |
| 4<br>IER      | <p>Comparator Interrupt Enable Rising</p> <p>Enables the CFR interrupt from the CMP. When this field is set, an interrupt will be asserted when CFR is set.</p>  |

Table continues on the next page...

**CMPx\_SCR field descriptions (continued)**

| Field  | Description  |
|--------|--|
|        | 0 Interrupt is disabled.<br>1 Interrupt is enabled.  |
| 3 IEF  | Comparator Interrupt Enable Falling<br><br>Enables the CFF interrupt from the CMP. When this field is set, an interrupt will be asserted when CFF is set.<br><br>0 Interrupt is disabled.<br>1 Interrupt is enabled.   |
| 2 CFR  | Analog Comparator Flag Rising<br><br>Detects a rising-edge on COUT, when set, during normal operation. CFR is cleared by writing 1 to it. During Stop modes, CFR is level sensitive .<br><br>0 Rising-edge on COUT has not been detected.<br>1 Rising-edge on COUT has occurred.     |
| 1 CFF  | Analog Comparator Flag Falling<br><br>Detects a falling-edge on COUT, when set, during normal operation. CFF is cleared by writing 1 to it. During Stop modes, CFF is level sensitive .<br><br>0 Falling-edge on COUT has not been detected.<br>1 Falling-edge on COUT has occurred. |
| 0 COUT | Analog Comparator Output<br><br>Returns the current value of the Analog Comparator output, when read. The field is reset to 0 and will read as CR1[INV] when the Analog Comparator module is disabled, that is, when CR1[EN] = 0. Writes to this field are ignored.                  |

**29.7.5 DAC Control Register (CMPx\_DACCR)**

Address: 4007\_3000h base + 4h offset = 4007\_3004h

| Bit        | 7     | 6     | 5 | 4 |  | 3     | 2 | 1 | 0 |
|------------|-------|-------|---|---|--|-------|---|---|---|
| Read Write | DACEN | VRSEL |   |   |  | VOSEL |   |   |   |
| Reset      | 0     | 0     | 0 | 0 |  | 0     | 0 | 0 | 0 |

**CMPx\_DACCR field descriptions**

| Field   | Description   |
|---------|---|
| 7 DACEN | DAC Enable<br><br>Enables the DAC. When the DAC is disabled, it is powered down to conserve power.<br><br>0 DAC is disabled.<br>1 DAC is enabled. |
| 6 VRSEL | Supply Voltage Reference Source Select  |

*Table continues on the next page...*

**CMPx\_DACCR field descriptions (continued)**

| Field        | Description   |
|--------------|---|
|              | 0 V is selected as resistor ladder network supply reference $V_{in1in}$<br>1 V is selected as resistor ladder network supply reference $V_{in2in}$  |
| 5–0<br>VOSEL | DAC Output Voltage Select<br><br>Selects an output voltage from one of 64 distinct levels.<br><br>$DACO = (V_{in} / 64) * (VOSEL[5:0] + 1)$ , so the DACO range is from $V_{in} / 64$ to $V_{in}$ . |

**29.7.6 MUX Control Register (CMPx\_MUXCR)**

Address: 4007\_3000h base + 5h offset = 4007\_3005h

| Bit   | 7    | 6 | 5 | 4    | 3 | 2 | 1    | 0 |
|-------|------|---|---|------|---|---|------|---|
| Read  | PSTM | 0 |   | PSEL |   |   | MSEL |   |
| Write | 0    | 0 | 0 | 0    | 0 | 0 | 0    | 0 |
| Reset |      |   |   |      |   |   |      |   |

**CMPx\_MUXCR field descriptions**

| Field         | Description  |
|---------------|--|
| 7<br>PSTM     | Pass Through Mode Enable<br><br>This bit is used to enable to MUX pass through mode. Pass through mode is always available but for some devices this feature must be always disabled due to the lack of package pins.<br><br>0 Pass Through Mode is disabled.<br>1 Pass Through Mode is enabled.   |
| 6<br>Reserved | This field is reserved.<br>This read-only field is reserved and always has the value 0.  |
| 5–3<br>PSEL   | Plus Input Mux Control<br><br>Determines which input is selected for the plus input of the comparator. For INx inputs, see CMP, DAC, and ANMUX block diagrams.<br><br><b>NOTE:</b> When an inappropriate operation selects the same input for both muxes, the comparator automatically shuts down to prevent itself from becoming a noise generator.<br><br>000 IN0<br>001 IN1<br>010 IN2<br>011 IN3<br>100 IN4<br>101 IN5<br>110 IN6<br>111 IN7 |
| 2–0<br>MSEL   | Minus Input Mux Control<br><br>Determines which input is selected for the minus input of the comparator. For INx inputs, see CMP, DAC, and ANMUX block diagrams.   |

*Table continues on the next page...*

**CMPx\_MUXCR field descriptions (continued)**

| Field | Description  |     |     |     |     |     |     |     |     |     |     |     |     |     |     |     |     |
|-------|--|-----|-----|-----|-----|-----|-----|-----|-----|-----|-----|-----|-----|-----|-----|-----|-----|
|       | <p><b>NOTE:</b> When an inappropriate operation selects the same input for both muxes, the comparator automatically shuts down to prevent itself from becoming a noise generator.</p> <table> <tr><td>000</td><td>IN0</td></tr> <tr><td>001</td><td>IN1</td></tr> <tr><td>010</td><td>IN2</td></tr> <tr><td>011</td><td>IN3</td></tr> <tr><td>100</td><td>IN4</td></tr> <tr><td>101</td><td>IN5</td></tr> <tr><td>110</td><td>IN6</td></tr> <tr><td>111</td><td>IN7</td></tr> </table> | 000 | IN0 | 001 | IN1 | 010 | IN2 | 011 | IN3 | 100 | IN4 | 101 | IN5 | 110 | IN6 | 111 | IN7 |
| 000   | IN0  |     |     |     |     |     |     |     |     |     |     |     |     |     |     |     |     |
| 001   | IN1  |     |     |     |     |     |     |     |     |     |     |     |     |     |     |     |     |
| 010   | IN2  |     |     |     |     |     |     |     |     |     |     |     |     |     |     |     |     |
| 011   | IN3  |     |     |     |     |     |     |     |     |     |     |     |     |     |     |     |     |
| 100   | IN4  |     |     |     |     |     |     |     |     |     |     |     |     |     |     |     |     |
| 101   | IN5  |     |     |     |     |     |     |     |     |     |     |     |     |     |     |     |     |
| 110   | IN6  |     |     |     |     |     |     |     |     |     |     |     |     |     |     |     |     |
| 111   | IN7  |     |     |     |     |     |     |     |     |     |     |     |     |     |     |     |     |

## 29.8 Functional description

The CMP module can be used to compare two analog input voltages applied to INP and INM. CMPO is high when the non-inverting input is greater than the inverting input, and is low when the non-inverting input is less than the inverting input. This signal can be selectively inverted by setting CR1[INV] = 1.

SCR[IER] and SCR[IEF] are used to select the condition which will cause the CMP module to assert an interrupt to the processor. SCR[CFF] is set on a falling-edge and SCR[CFR] is set on rising-edge of the comparator output. The optionally filtered CMPO can be read directly through SCR[COUT].

### 29.8.1 CMP functional modes

There are three main sub-blocks to the CMP module:

- The comparator itself
- The window function
- The filter function

The filter, CR0[FILTER\_CNT], can be clocked from an internal or external clock source. The filter is programmable with respect to the number of samples that must agree before a change in the output is registered. In the simplest case, only one sample must agree. In this case, the filter acts as a simple sampler.

The external sample input is enabled using CR1[SE]. When set, the output of the comparator is sampled only on rising edges of the sample input.

The "windowing mode" is enabled by setting CR1[WE]. When set, the comparator output is sampled only when WINDOW=1. This feature can be used to ignore the comparator output during time periods in which the input voltages are not valid. This is especially useful when implementing zero-crossing-detection for certain PWM applications.

The comparator filter and sampling features can be combined as shown in the following table. Individual modes are discussed below.

**Table 29-15. Comparator sample/filter controls**

| Mode #   | CR1[EN] | CR1[WE] | CR1[SE] | CR0[FILTER_CNT] | FPR[FILT_PER] | Operation  |
|--|---------|---------|---------|-----------------|---------------|--|
| 1  | 0       | X       | X       | X               | X             | <b>Disabled</b><br>See the <a href="#">Disabled mode (# 1).</a>  |
| 2A   | 1       | 0       | 0       | 0x00            | X             | <b>Continuous Mode</b>   |
| 2B   | 1       | 0       | 0       | X               | 0x00          | See the <a href="#">Continuous mode (#s 2A &amp; 2B).</a>  |
| 3A   | 1       | 0       | 1       | 0x01            | X             | <b>Sampled, Non-Filtered mode</b>  |
| 3B   | 1       | 0       | 0       | 0x01            | > 0x00        | See the <a href="#">Sampled, Non-Filtered mode (#s 3A &amp; 3B).</a>   |
| 4A   | 1       | 0       | 1       | > 0x01          | X             | <b>Sampled, Filtered mode</b>  |
| 4B   | 1       | 0       | 0       | > 0x01          | > 0x00        | See the <a href="#">Sampled, Filtered mode (#s 4A &amp; 4B).</a>   |
| 5A   | 1       | 1       | 0       | 0x00            | X             | <b>Windowed mode</b>   |
| 5B   | 1       | 1       | 0       | X               | 0x00          | Comparator output is sampled on every rising bus clock edge when SAMPLE=1 to generate COUTA.<br>See the <a href="#">Windowed mode (#s 5A &amp; 5B).</a>  |
| 6  | 1       | 1       | 0       | 0x01            | 0x01–0xFF     | <b>Windowed/Resampled mode</b><br>Comparator output is sampled on every rising bus clock edge when SAMPLE=1 to generate COUTA, which is then resampled on an interval determined by FILT_PER to generate COUT.<br>See the <a href="#">Windowed/Resampled mode (# 6).</a> |
| 7  | 1       | 1       | 0       | > 0x01          | 0x01–0xFF     | <b>Windowed/Filtered mode</b><br>Comparator output is sampled on every rising bus clock edge when SAMPLE=1 to generate COUTA, which is then resampled and filtered to generate COUT.<br>See the <a href="#">Windowed/Filtered mode (#7).</a>                             |
| All other combinations of CR1[EN], CR1[WE], CR1[SE], CR0[FILTER_CNT], and FPR[FILT_PER] are illegal. |         |         |         |                 |               |  |

For cases where a comparator is used to drive a fault input, for example, for a motor-control module such as FTM, it must be configured to operate in Continuous mode so that an external fault can immediately pass through the comparator to the target fault circuitry.

### Note

Filtering and sampling settings must be changed only after setting CR1[SE]=0 and CR0[FILTER\_CNT]=0x00. This resets the filter to a known state.

#### 29.8.1.1 Disabled mode (# 1)

In Disabled mode, the analog comparator is non-functional and consumes no power. CMPO is 0 in this mode.

#### 29.8.1.2 Continuous mode (#s 2A & 2B)

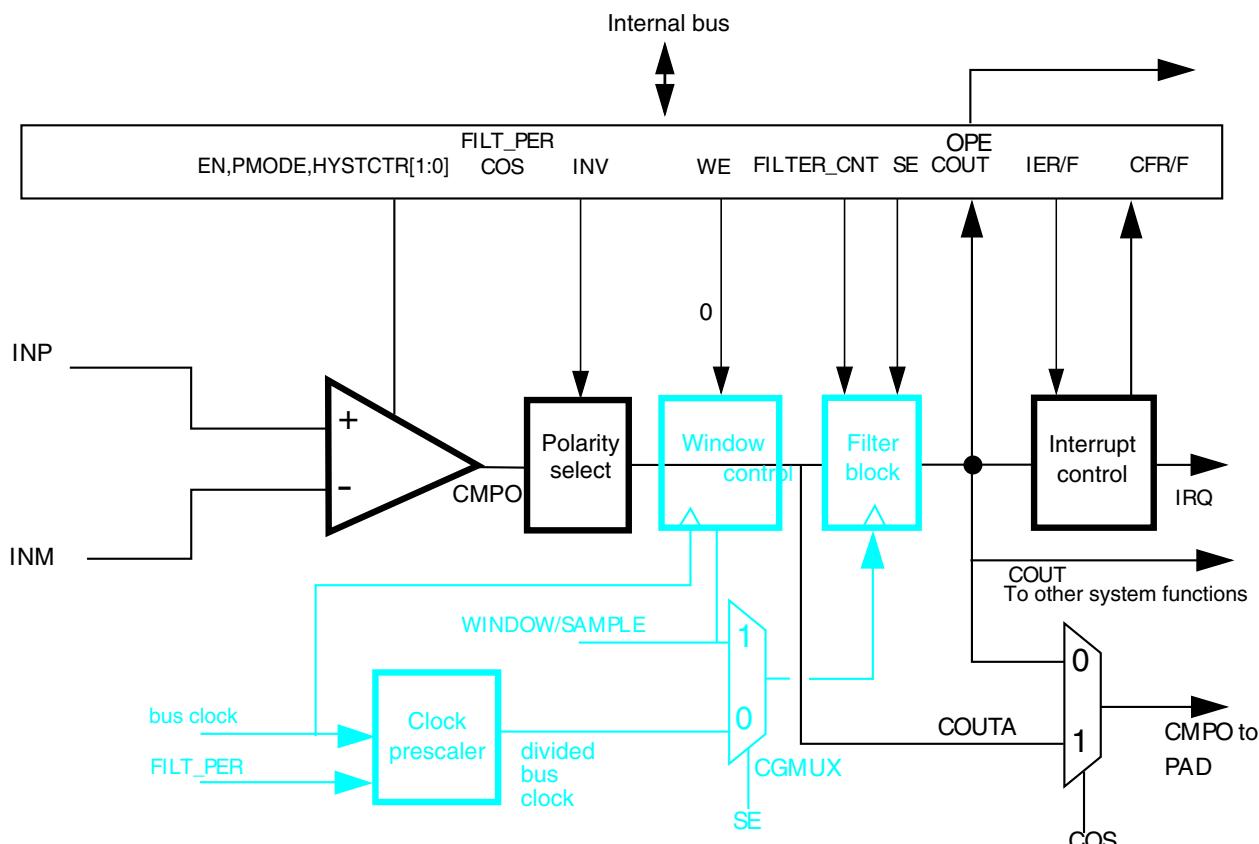


Figure 29-15. Comparator operation in Continuous mode

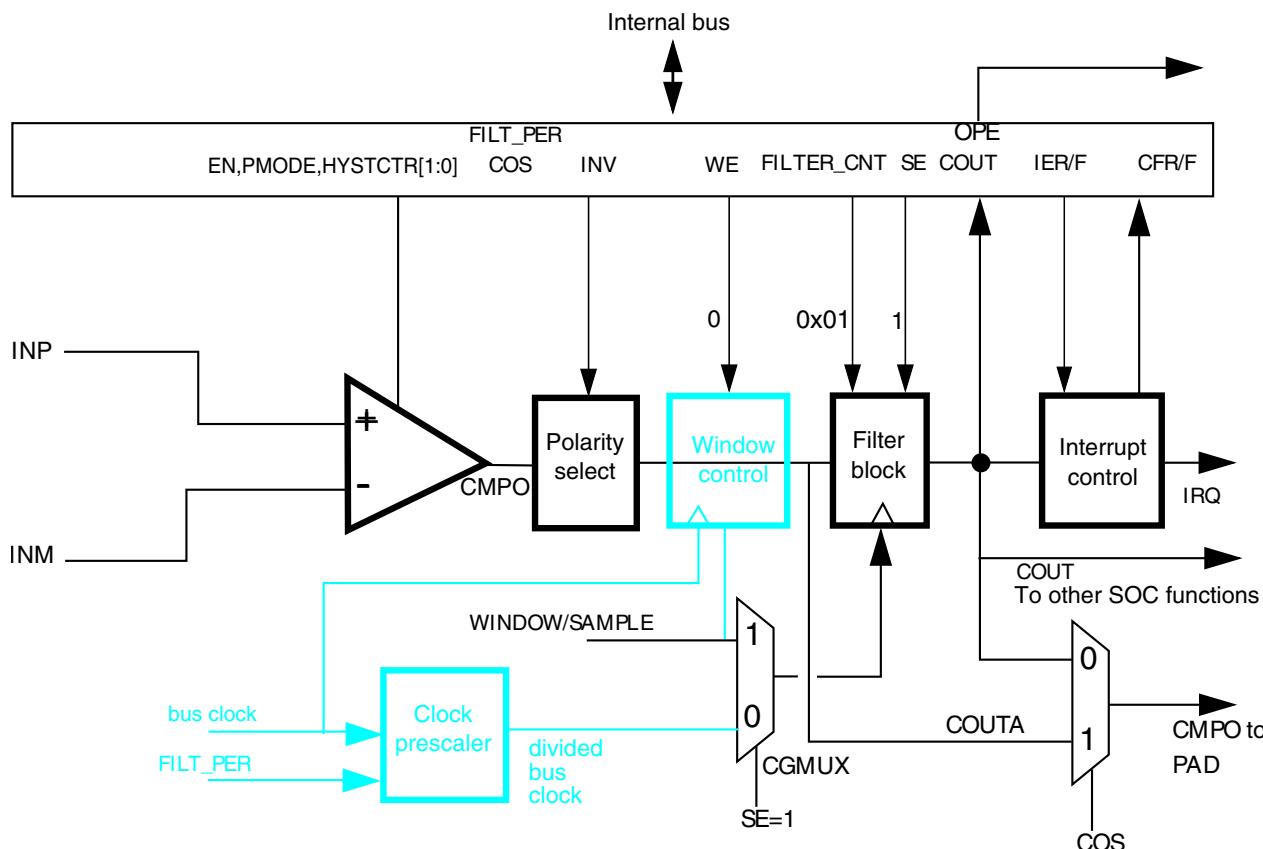
**NOTE**

See the chip configuration section for the source of sample/window input.

The analog comparator block is powered and active. CMPO may be optionally inverted, but is not subject to external sampling or filtering. Both window control and filter blocks are completely bypassed. SCR[COUT] is updated continuously. The path from comparator input pins to output pin is operating in combinational unclocked mode. COUT and COUTA are identical.

For control configurations which result in disabling the filter block, see the [Filter Block Bypass Logic](#) diagram.

### 29.8.1.3 Sampled, Non-Filtered mode (#s 3A & 3B)



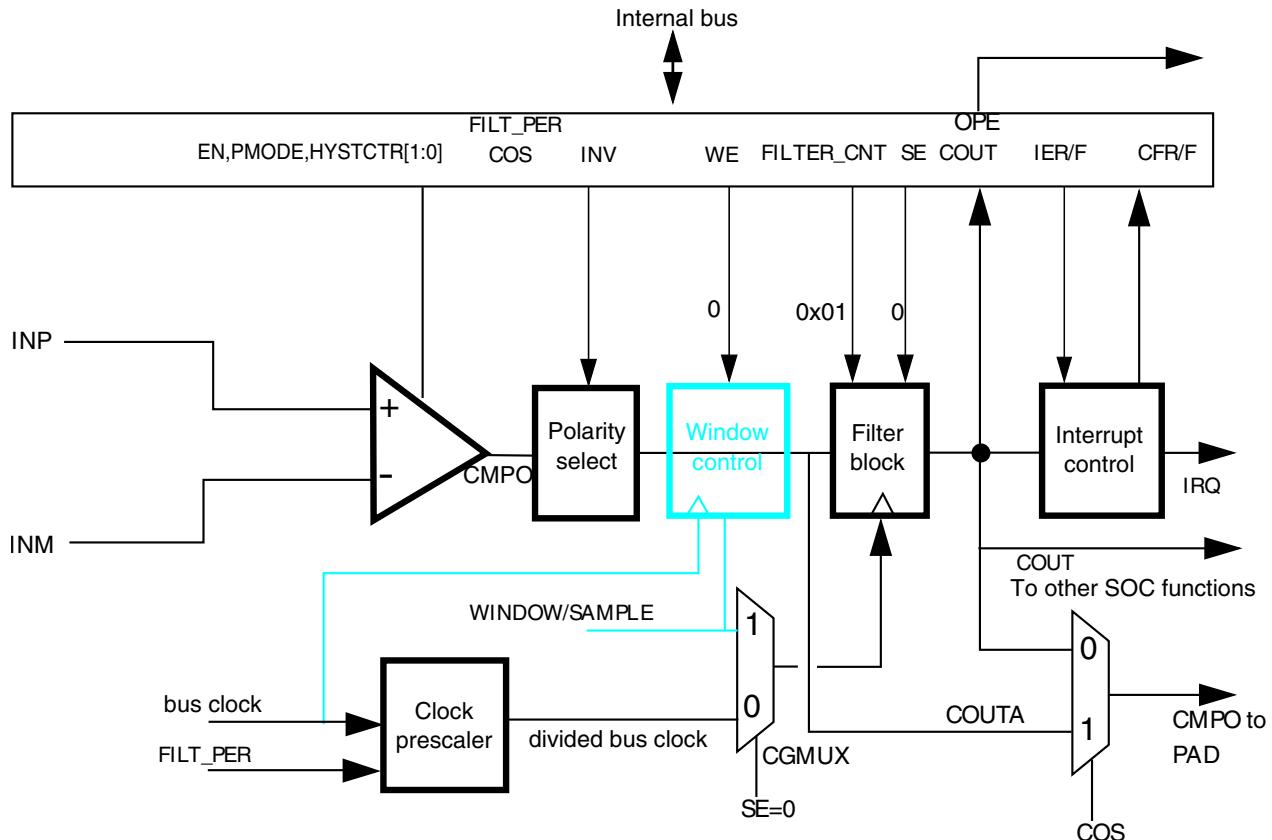
**Figure 29-16. Sampled, Non-Filtered (# 3A): sampling point externally driven**

In Sampled, Non-Filtered mode, the analog comparator block is powered and active. The path from analog inputs to COUTA is combinational unclocked. Windowing control is completely bypassed. COUTA is sampled whenever a rising-edge is detected on the filter block clock input.

## Functional description

The only difference in operation between Sampled, Non-Filtered (# 3A) and Sampled, Non-Filtered (# 3B) is in how the clock to the filter block is derived. In #3A, the clock to filter block is externally derived while in #3B, the clock to filter block is internally derived.

The comparator filter has no other function than sample/hold of the comparator output in this mode (# 3B).



**Figure 29-17. Sampled, Non-Filtered (# 3B): sampling interval internally derived**

### 29.8.1.4 Sampled, Filtered mode (#s 4A & 4B)

In Sampled, Filtered mode, the analog comparator block is powered and active. The path from analog inputs to COUTA is combinational unclocked. Windowing control is completely bypassed. COUTA is sampled whenever a rising edge is detected on the filter block clock input.

The only difference in operation between Sampled, Non-Filtered (# 3A) and Sampled, Filtered (# 4A) is that, now, CR0[FILTER\_CNT]>1, which activates filter operation.

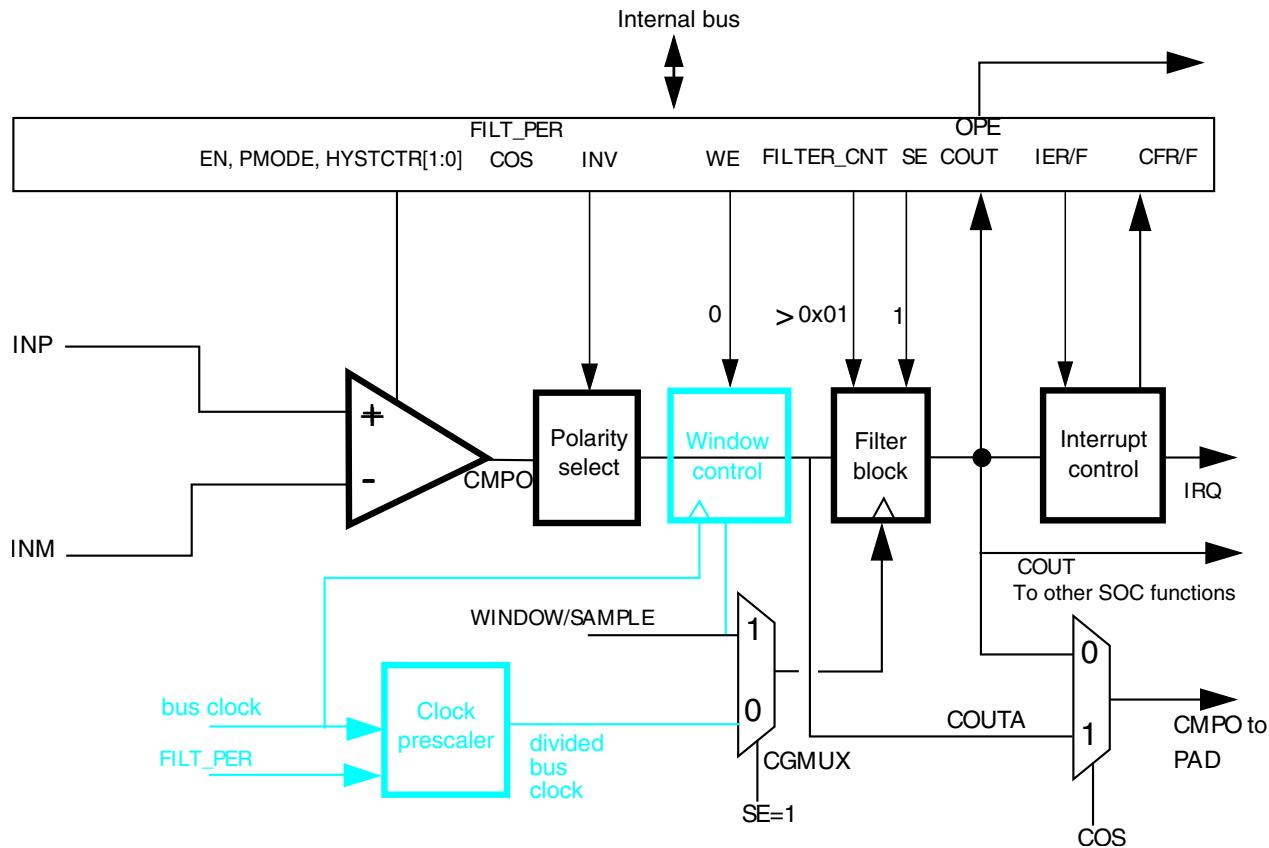
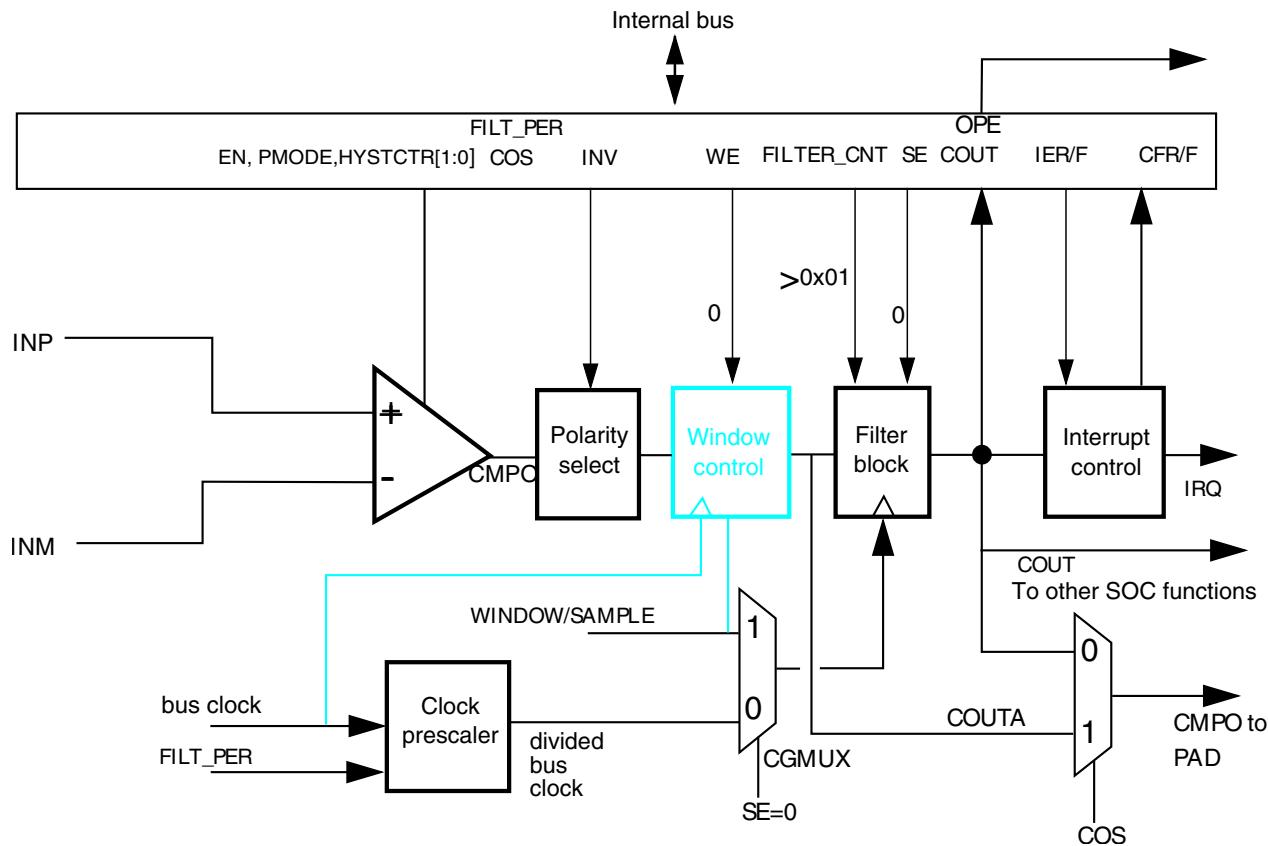


Figure 29-18. Sampled, Filtered (# 4A): sampling point externally driven



**Figure 29-19. Sampled, Filtered (# 4B): sampling point internally derived**

The only difference in operation between Sampled, Non-Filtered (# 3B) and Sampled, Filtered (# 4B) is that now, CR0[FILTER\_CNT]>1, which activates filter operation.

### 29.8.1.5 Windowed mode (#s 5A & 5B)

The following figure illustrates comparator operation in the Windowed mode, ignoring latency of the analog comparator, polarity select, and window control block. It also assumes that the polarity select is set to non-inverting state.

#### NOTE

The analog comparator output is passed to COUTA only when the WINDOW signal is high.

In actual operation, COUTA may lag the analog inputs by up to one bus clock cycle plus the combinational path delay through the comparator and polarity select logic.

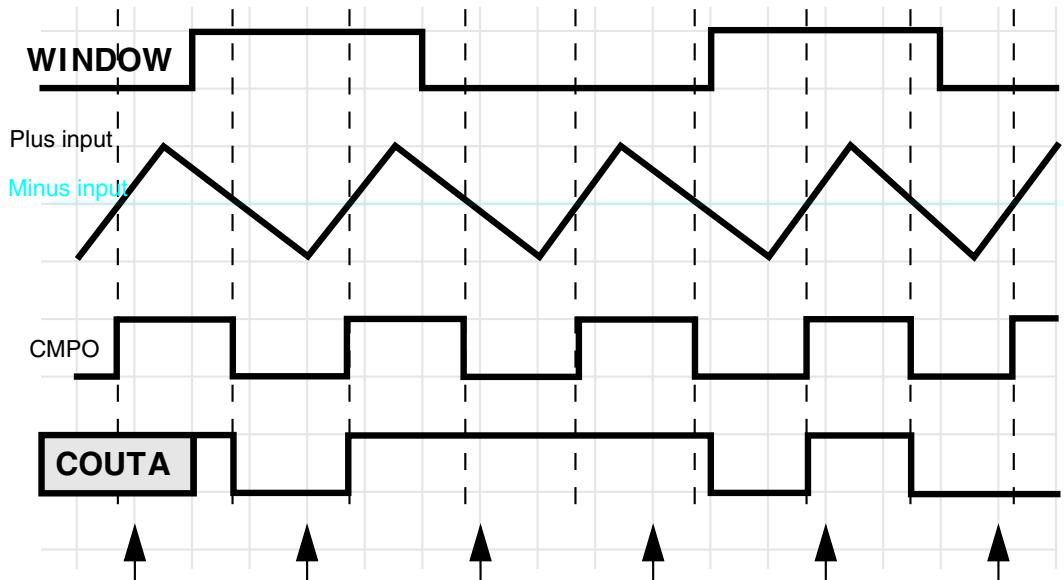


Figure 29-20. Windowed mode operation

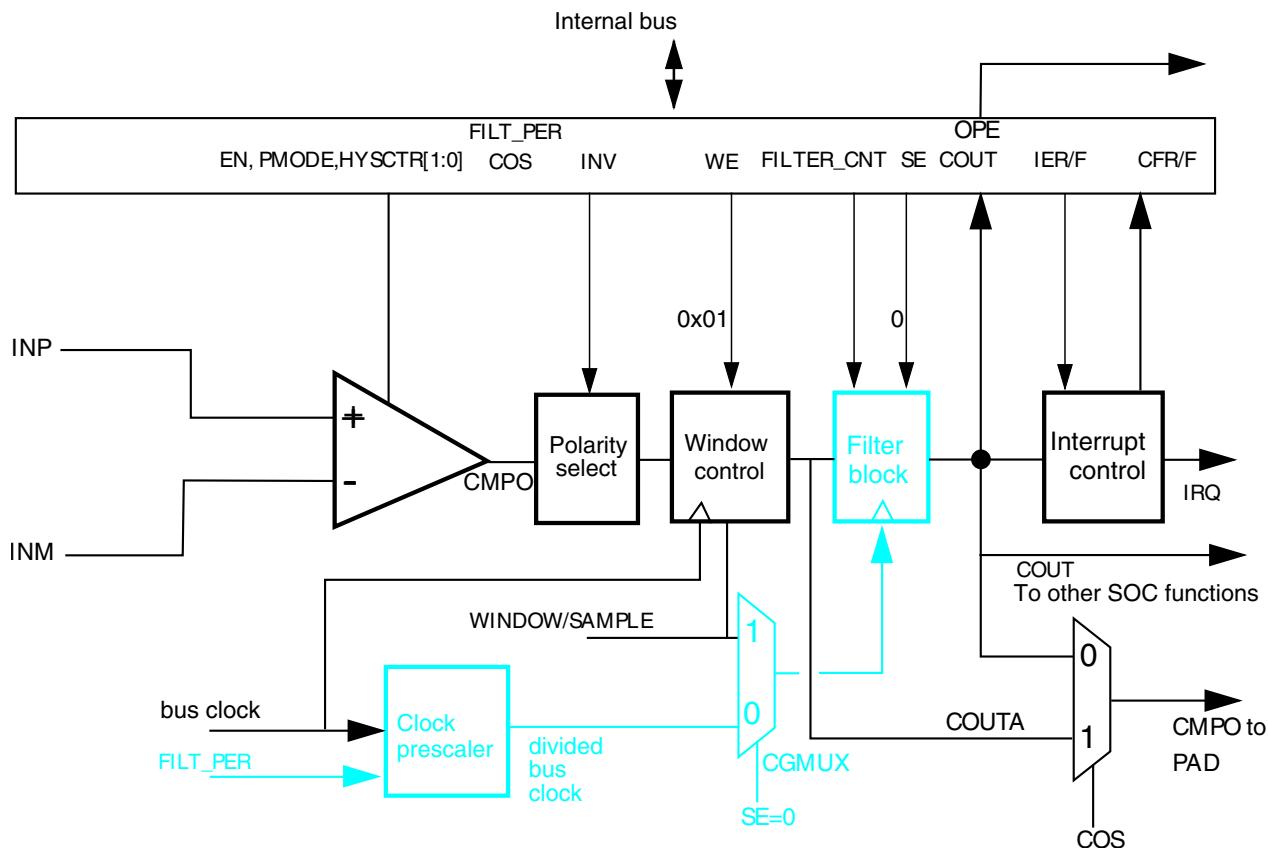


Figure 29-21. Windowed mode

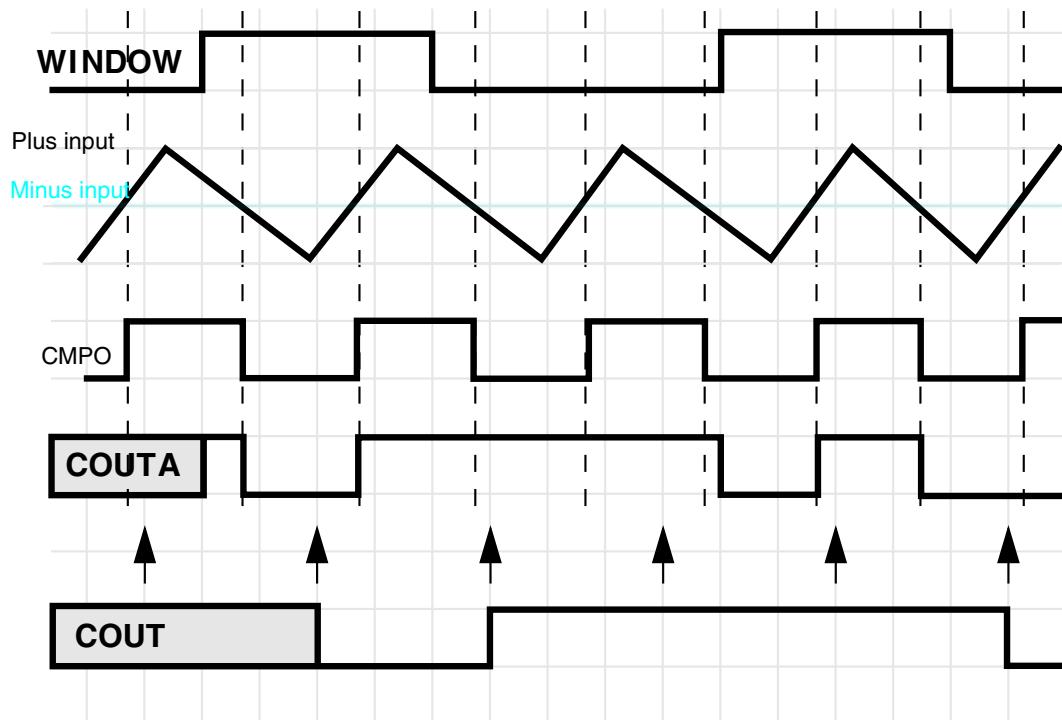
For control configurations which result in disabling the filter block, see [Filter Block Bypass Logic](#) diagram.

When any windowed mode is active, COUTA is clocked by the bus clock whenever WINDOW = 1. The last latched value is held when WINDOW = 0.

### 29.8.1.6 Windowed/Resampled mode (# 6)

The following figure uses the same input stimulus shown in [Figure 29-20](#), and adds resampling of COUTA to generate COUT. Samples are taken at the time points indicated by the arrows in the figure. Again, prop delays and latency are ignored for the sake of clarity.

This example was generated solely to demonstrate operation of the comparator in windowed/resampled mode, and does not reflect any specific application. Depending upon the sampling rate and window placement, COUT may not see zero-crossing events detected by the analog comparator. Sampling period and/or window placement must be carefully considered for a given application.



**Figure 29-22. Windowed/resampled mode operation**

This mode of operation results in an unfiltered string of comparator samples where the interval between the samples is determined by FPR[FILT\_PER] and the bus clock rate. Configuration for this mode is virtually identical to that for the Windowed/Filtered Mode shown in the next section. The only difference is that the value of CR0[FILTER\_CNT] must be 1.

### 29.8.1.7 Windowed/Filtered mode (#7)

This is the most complex mode of operation for the comparator block, as it uses both windowing and filtering features. It also has the highest latency of any of the modes. This can be approximated: up to 1 bus clock synchronization in the window function +  $((CR0[FILTER_CNT] * FPR[FILT_PER]) + 1) * \text{bus clock}$  for the filter function.

When any windowed mode is active, COUTA is clocked by the bus clock whenever WINDOW = 1. The last latched value is held when WINDOW = 0.

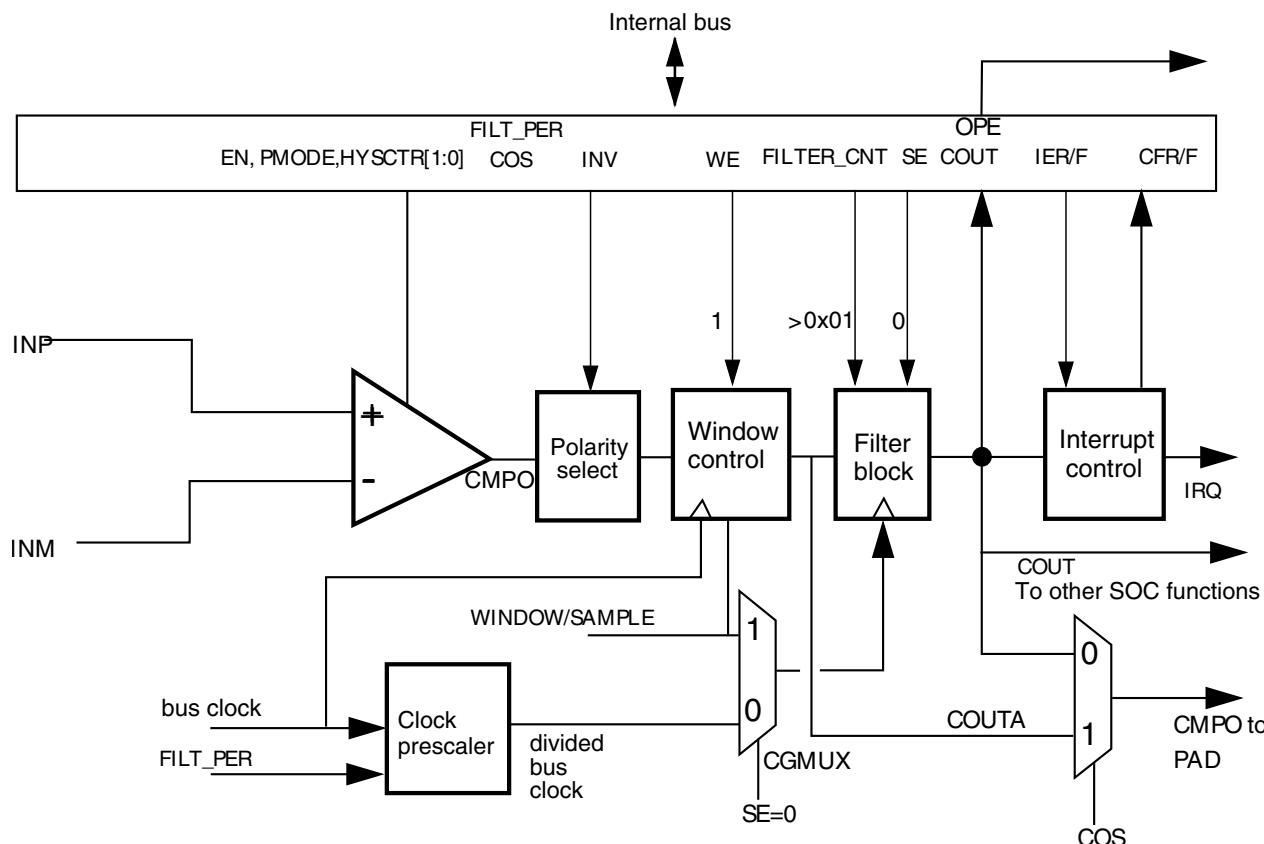


Figure 29-23. Windowed/Filtered mode

## 29.8.2 Power modes

### 29.8.2.1 Wait mode operation

During Wait and VLPW modes, the CMP, if enabled, continues to operate normally and a CMP interrupt can wake the MCU.

### 29.8.2.2 Stop mode operation

Depending on clock restrictions related to the MCU core or core peripherals, the MCU is brought out of stop when a compare event occurs and the corresponding interrupt is enabled. Similarly, if CR1[OPE] is enabled, the comparator output operates as in the normal operating mode and comparator output is placed onto the external pin. In Stop modes, the comparator can be operational in both:

- High-Speed (HS) Comparison mode when CR1[PMODE] = 1
- Low-Speed (LS) Comparison mode when CR1[PMODE] = 0

It is recommended to use the LS mode to minimize power consumption.

If stop is exited with a reset, all comparator registers are put into their reset state.

### 29.8.2.3 Low-Leakage mode operation

When the chip is in Low-Leakage modes:

- The CMP module is partially functional and is limited to Low-Speed mode, regardless of CR1[PMODE] setting
- Windowed, Sampled, and Filtered modes are not supported
- The CMP output pin is latched and does not reflect the compare output state.

The positive- and negative-input voltage can be supplied from external pins or the DAC output. The MCU can be brought out of the Low-Leakage mode if a compare event occurs and the CMP interrupt is enabled. After wakeup from low-leakage modes, the CMP module is in the reset state except for SCR[CFF] and SCR[CFR].

### 29.8.2.4 Background Debug Mode Operation

When the microcontroller is in active background debug mode, the CMP continues to operate normally.

## 29.8.3 Startup and operation

A typical startup sequence is as follows.

The time required to stabilize COUT will be the power-on delay of the comparators plus the largest propagation delay from a selected analog source through the analog comparator, windowing function and filter. See the Data Sheets for power-on delays of the comparators. The windowing function has a maximum of one bus clock period delay. The filter delay is specified in the [Low-pass filter](#).

During operation, the propagation delay of the selected data paths must always be considered. It may take many bus clock cycles for COUT and SCR[CFR]/SCR[CFF] to reflect an input change or a configuration change to one of the components involved in the data path.

When programmed for filtering modes, COUT will initially be equal to 0, until sufficient clock cycles have elapsed to fill all stages of the filter. This occurs even if COUTA is at a logic 1.

## 29.8.4 Low-pass filter

The low-pass filter operates on the unfiltered and unsynchronized and optionally inverted comparator output COUTA and generates the filtered and synchronized output COUT. Both COUTA and COUT can be configured as module outputs and are used for different purposes within the system.

Synchronization and edge detection are always used to determine status register bit values. They also apply to COUT for all sampling and windowed modes. Filtering can be performed using an internal timebase defined by FPR[FILT\_PER], or using an external SAMPLE input to determine sample time.

The need for digital filtering and the amount of filtering is dependent on user requirements. Filtering can become more useful in the absence of an external hysteresis circuit. Without external hysteresis, high-frequency oscillations can be generated at COUTA when the selected INM and INP input voltages differ by less than the offset voltage of the differential comparator.

### 29.8.4.1 Enabling filter modes

Filter modes can be enabled by:

- Setting CR0[FILTER\_CNT] > 0x01 and
- Setting FPR[FILT\_PER] to a nonzero value or setting CR1[SE]=1

If using the divided bus clock to drive the filter, it will take samples of COUTA every FPR[FILT\_PER] bus clock cycles.

## Functional description

The filter output will be at logic 0 when first initialized, and will subsequently change when all the consecutive CR0[FILTER\_CNT] samples agree that the output value has changed. In other words, SCR[COUT] will be 0 for some initial period, even when COUTA is at logic 1.

Setting both CR1[SE] and FPR[FILT\_PER] to 0 disables the filter and eliminates switching current associated with the filtering process.

### Note

Always switch to this setting prior to making any changes in filter parameters. This resets the filter to a known state. Switching CR0[FILTER\_CNT] on the fly without this intermediate step can result in unexpected behavior.

If CR1[SE]=1, the filter takes samples of COUTA on each positive transition of the sample input. The output state of the filter changes when all the consecutive CR0[FILTER\_CNT] samples agree that the output value has changed.

### 29.8.4.2 Latency issues

The value of FPR[FILT\_PER] or SAMPLE period must be set such that the sampling period is just longer than the period of the expected noise. This way a noise spike will corrupt only one sample. The value of CR0[FILTER\_CNT] must be chosen to reduce the probability of noisy samples causing an incorrect transition to be recognized. The probability of an incorrect transition is defined as the probability of an incorrect sample raised to the power of CR0[FILTER\_CNT].

The values of FPR[FILT\_PER] or SAMPLE period and CR0[FILTER\_CNT] must also be traded off against the desire for minimal latency in recognizing actual comparator output transitions. The probability of detecting an actual output change within the nominal latency is the probability of a correct sample raised to the power of CR0[FILTER\_CNT].

The following table summarizes maximum latency values for the various modes of operation *in the absence of noise*. Filtering latency is restarted each time an actual output transition is masked by noise.

**Table 29-16. Comparator sample/filter maximum latencies**

| Mode # | CR1[EN] | CR1[WE] | CR1[SE] | CR0[FILTER_CNT] | FPR[FILT_PER] | Operation | Maximum latency <sup>1</sup> |
|--------|---------|---------|---------|-----------------|---------------|-----------|------------------------------|
| 1      | 0       | X       | X       | X               | X             | Disabled  | N/A                          |

*Table continues on the next page...*

**Table 29-16. Comparator sample/filter maximum latencies (continued)**

| Mode # | CR1[EN] | CR1[WE] | CR1[SE] | CR0[FILTER_CNT] | FPR[FILT_PER] | Operation                  | Maximum latency <sup>1</sup>  |
|--------|---------|---------|---------|-----------------|---------------|----------------------------|---|
| 2A     | 1       | 0       | 0       | 0x00            | X             | Continuous Mode            | $T_{PD}$  |
| 2B     | 1       | 0       | 0       | X               | 0x00          |                            |   |
| 3A     | 1       | 0       | 1       | 0x01            | X             | Sampled, Non-Filtered mode | $T_{PD} + T_{SAMPLE} + T_{per}$                                     |
| 3B     | 1       | 0       | 0       | 0x01            | > 0x00        |                            | $T_{PD} + (FPR[FILT\_PER] * T_{per}) + T_{per}$                     |
| 4A     | 1       | 0       | 1       | > 0x01          | X             | Sampled, Filtered mode     | $T_{PD} + (CR0[FILTER\_CNT] * T_{SAMPLE}) + T_{per}$                |
| 4B     | 1       | 0       | 0       | > 0x01          | > 0x00        |                            | $T_{PD} + (CR0[FILTER\_CNT] * FPR[FILT\_PER] * T_{per}) + T_{per}$  |
| 5A     | 1       | 1       | 0       | 0x00            | X             | Windowed mode              | $T_{PD} + T_{per}$  |
| 5B     | 1       | 1       | 0       | X               | 0x00          |                            | $T_{PD} + T_{per}$  |
| 6      | 1       | 1       | 0       | 0x01            | 0x01 - 0xFF   | Windowed / Resampled mode  | $T_{PD} + (FPR[FILT\_PER] * T_{per}) + 2T_{per}$                    |
| 7      | 1       | 1       | 0       | > 0x01          | 0x01 - 0xFF   | Windowed / Filtered mode   | $T_{PD} + (CR0[FILTER\_CNT] * FPR[FILT\_PER] * T_{per}) + 2T_{per}$ |

1.  $T_{PD}$  represents the intrinsic delay of the analog component plus the polarity select logic.  $T_{SAMPLE}$  is the clock period of the external sample clock.  $T_{per}$  is the period of the bus clock.

## 29.9 CMP interrupts

The CMP module is capable of generating an interrupt on either the rising- or falling-edge of the comparator output, or both. The following table gives the conditions in which the interrupt request is asserted and deasserted.

| When   | Then                                |
|--|-------------------------------------|
| SCR[IER] and SCR[CFR] are set                                  | The interrupt request is asserted   |
| SCR[IEF] and SCR[CFF] are set                                  | The interrupt request is asserted   |
| SCR[IER] and SCR[CFR] are cleared for a rising-edge interrupt  | The interrupt request is deasserted |
| SCR[IEF] and SCR[CFF] are cleared for a falling-edge interrupt | The interrupt request is deasserted |

## 29.10 DMA support

Normally, the CMP generates a CPU interrupt if there is a change on the COUT. When DMA support is enabled by setting SCR[DMAEN] and the interrupt is enabled by setting SCR[IER], SCR[IEF], or both, the corresponding change on COUT forces a DMA

### CMP Asynchronous DMA support

transfer request rather than a CPU interrupt instead. When the DMA has completed the transfer, it sends a transfer completing indicator that deasserts the DMA transfer request and clears the flag to allow a subsequent change on comparator output to occur and force another DMA request.

The comparator can remain functional in STOP modes. When DMA support is enabled by setting SCR[DMAEN] and the interrupt is enabled by setting SCR[IER], SCR[IEF], or both, the corresponding change on COUT forces a DMA transfer request to wake up the system from STOP modes. After the data transfer has finished, system will go back to STOP modes. Refer to DMA chapters in the device reference manual for the asynchronous DMA function for details.

## 29.11 CMP Asynchronous DMA support

The comparator can remain functional in STOP modes. When DMA support is enabled by setting SCR[DMAEN] and the interrupt is enabled by setting SCR[IER], SCR[IEF], or both, the corresponding change on COUT forces a DMA transfer request to wake up the system from STOP modes. After the data transfer has finished, system will go back to STOP modes. Refer to DMA chapters in the device reference manual for the asynchronous DMA function for details.

## 29.12 Digital-to-analog converter

The following figure shows the block diagram of the DAC module. It contains a 64-tap resistor ladder network and a 64-to-1 multiplexer, which selects an output voltage from one of 64 distinct levels that outputs from DACO. It is controlled through the DAC Control Register (DCCR). Its supply reference source can be selected from two sources  $V_{in1}$  and  $V_{in2}$ . The module can be powered down or disabled when not in use. When in Disabled mode, DACO is connected to the analog ground.

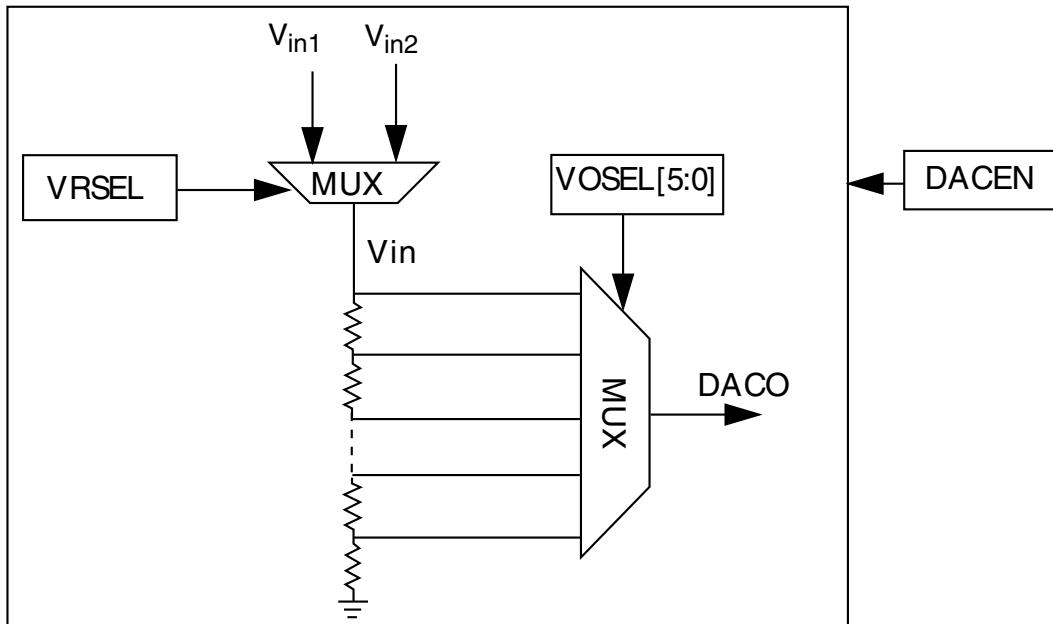


Figure 29-24. 6-bit DAC block diagram

## 29.13 DAC functional description

This section provides DAC functional description.

### 29.13.1 Voltage reference source select

- $V_{in1}$  connects to the primary voltage source as supply reference of 64 tap resistor ladder
- $V_{in2}$  connects to an alternate voltage source

## 29.14 DAC resets

This module has a single reset input, corresponding to the chip-wide peripheral reset.

## 29.15 DAC clocks

This module has a single clock input, the bus clock.

## 29.16 DAC interrupts

This module has no interrupts.

## 29.17 CMP Trigger Mode

CMP and DAC are configured to CMP Trigger mode when CMP\_CR1[TRIGM] is set to 1. In addition, the CMP must be enabled. If the DAC is to be used as a reference to the CMP, it must also be enabled.

CMP Trigger mode depends on an external timer resource to periodically enable the CMP and 6-bit DAC in order to generate a triggered compare.

Upon setting TRIGM, the CMP and DAC are placed in a standby state until an external timer resource trigger is received.

See the chip configuration chapter for details about the external timer resource.

# Chapter 30

## 12-bit Digital-to-Analog Converter (DAC)

### 30.1 Introduction

#### NOTE

For the chip-specific implementation details of this module's instances see the chip configuration information.

The 12-bit digital-to-analog converter (DAC) is a low-power general-purpose DAC. The output of this DAC can be placed on an external pin or set as one of the inputs to the analog comparator, analog-to-digital converter (ADC), or other peripherals.

### 30.2 Features

The features of the DAC module include:

- On-chip programmable reference generator output. The voltage output range is from  $1/4096 V_{in}$  to  $V_{in}$ , and the step is  $1/4096 V_{in}$ , where  $V_{in}$  is the input voltage.
- $V_{in}$  can be selected from two reference sources
- Static operation in Normal Stop mode
- 2-word data buffer supported with multiple operation modes
- DMA support

### 30.3 Block diagram

The block diagram of the DAC module is as follows:

Block diagram

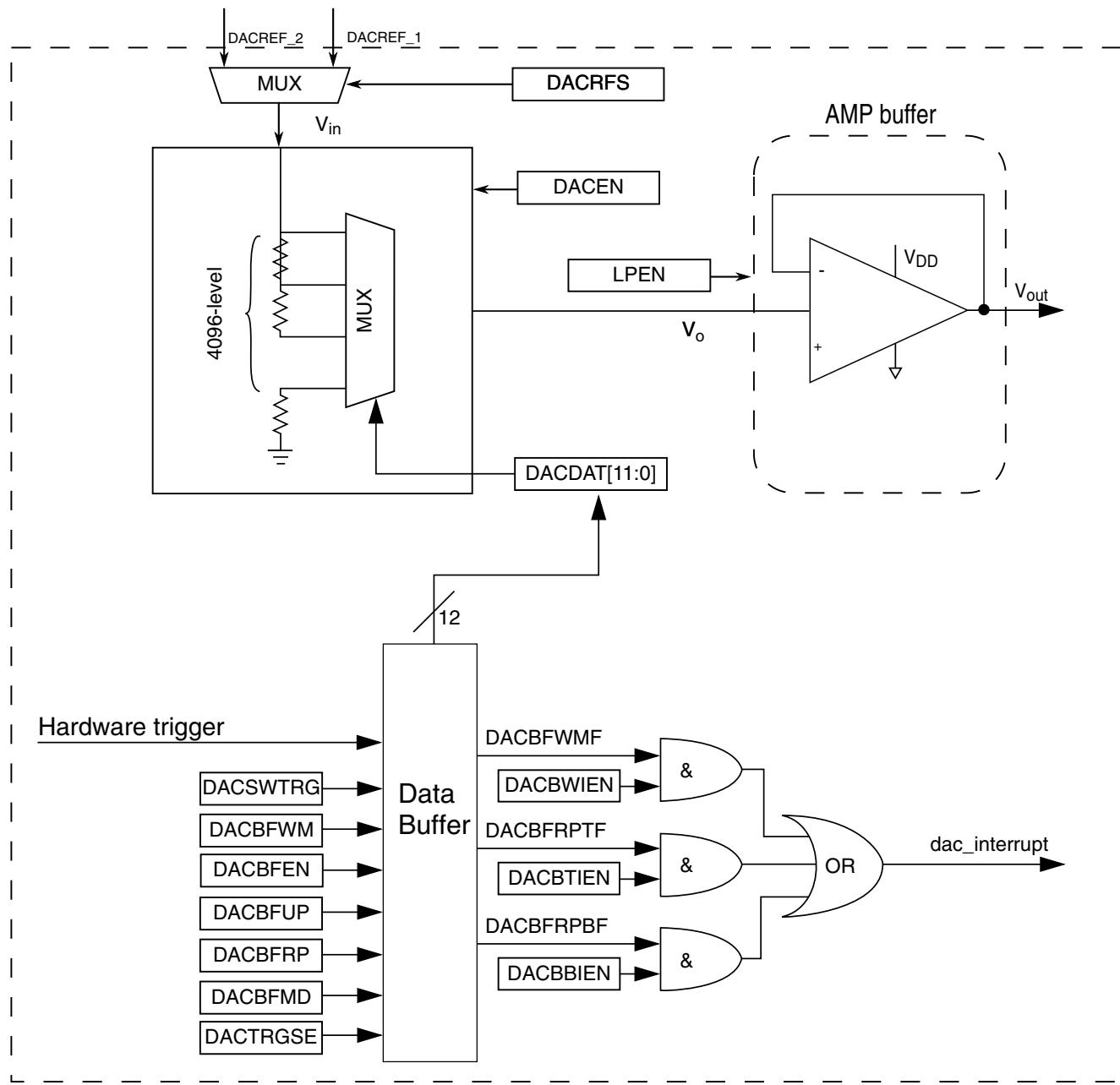


Figure 30-1. DAC block diagram

## 30.4 Memory map/register definition

The DAC has registers to control analog comparator and programmable voltage divider to perform the digital-to-analog functions.

The address of a register is the sum of a base address and an address offset. The base address is defined at the chip level. The address offset is defined at the module level.

**DAC memory map**

| Absolute address (hex) | Register name                       | Width (in bits) | Access | Reset value | Section/page               |
|------------------------|-------------------------------------|-----------------|--------|-------------|----------------------------|
| 4003_F000              | DAC Data Low Register (DAC0_DAT0L)  | 8               | R/W    | 00h         | <a href="#">30.4.1/539</a> |
| 4003_F001              | DAC Data High Register (DAC0_DAT0H) | 8               | R/W    | 00h         | <a href="#">30.4.2/540</a> |
| 4003_F002              | DAC Data Low Register (DAC0_DAT1L)  | 8               | R/W    | 00h         | <a href="#">30.4.1/539</a> |
| 4003_F003              | DAC Data High Register (DAC0_DAT1H) | 8               | R/W    | 00h         | <a href="#">30.4.2/540</a> |
| 4003_F020              | DAC Status Register (DAC0_SR)       | 8               | R/W    | 02h         | <a href="#">30.4.3/540</a> |
| 4003_F021              | DAC Control Register (DAC0_C0)      | 8               | R/W    | 00h         | <a href="#">30.4.4/541</a> |
| 4003_F022              | DAC Control Register 1 (DAC0_C1)    | 8               | R/W    | 00h         | <a href="#">30.4.5/542</a> |
| 4003_F023              | DAC Control Register 2 (DAC0_C2)    | 8               | R/W    | 01h         | <a href="#">30.4.6/542</a> |

### 30.4.1 DAC Data Low Register (DACx\_DATnL)

Address: 4003\_F000h base + 0h offset + (2d × i), where i=0d to 1d

| Bit        | 7 | 6 | 5 | 4 | 3     | 2 | 1 | 0 |
|------------|---|---|---|---|-------|---|---|---|
| Read Write |   |   |   |   | DATA0 |   |   |   |
| Reset      | 0 | 0 | 0 | 0 | 0     | 0 | 0 | 0 |

**DACx\_DATnL field descriptions**

| Field        | Description   |
|--------------|---|
| 7–0<br>DATA0 | When the DAC buffer is not enabled, DATA[11:0] controls the output voltage based on the following formula: $V_{out} = V_{in} * (1 + DACDAT0[11:0])/4096$<br>When the DAC buffer is enabled, DATA is mapped to the 16-word buffer. |

### 30.4.2 DAC Data High Register (DACx\_DATnH)

Address: 4003\_F000h base + 1h offset + (2d × i), where i=0d to 1d

| Bit   | 7 | 6 | 5 | 4 | 3 | 2 | 1     | 0 |
|-------|---|---|---|---|---|---|-------|---|
| Read  |   |   | 0 |   |   |   |       |   |
| Write |   |   |   |   |   |   | DATA1 |   |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0     | 0 |

#### DACx\_DATnH field descriptions

| Field           | Description   |
|-----------------|---|
| 7–4<br>Reserved | This field is reserved.<br>This read-only field is reserved and always has the value 0.   |
| 3–0<br>DATA1    | When the DAC Buffer is not enabled, DATA[11:0] controls the output voltage based on the following formula. $V_{out} = V_{in} * (1 + DACDAT0[11:0])/4096$<br>When the DAC buffer is enabled, DATA[11:0] is mapped to the 16-word buffer. |

### 30.4.3 DAC Status Register (DACx\_SR)

If DMA is enabled, the flags can be cleared automatically by DMA when the DMA request is done. Writing 0 to a field clears it whereas writing 1 has no effect. After reset, DACBFRPTF is set and can be cleared by software, if needed. The flags are set only when the data buffer status is changed.

Address: 4003\_F000h base + 20h offset = 4003\_F020h

| Bit   | 7 | 6 | 5 | 4 | 3 | 2 | 1         | 0         |
|-------|---|---|---|---|---|---|-----------|-----------|
| Read  |   |   | 0 |   |   |   |           |           |
| Write |   |   |   |   |   |   | DACBFRPTF | DACBFRPBF |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 1         | 0         |

#### DACx\_SR field descriptions

| Field           | Description  |
|-----------------|--|
| 7–2<br>Reserved | This field is reserved.<br>This read-only field is reserved and always has the value 0.  |
| 1<br>DACBFRPTF  | DAC Buffer Read Pointer Top Position Flag<br>0 The DAC buffer read pointer is not zero.<br>1 The DAC buffer read pointer is zero.                                    |
| 0<br>DACBFRPBF  | DAC Buffer Read Pointer Bottom Position Flag<br>0 The DAC buffer read pointer is not equal to C2[DACBFUP].<br>1 The DAC buffer read pointer is equal to C2[DACBFUP]. |

### 30.4.4 DAC Control Register (DACx\_C0)

Address: 4003\_F000h base + 21h offset = 4003\_F021h

| Bit   | 7     | 6      | 5                 | 4                 | 3    | 2 | 1            | 0            |
|-------|-------|--------|-------------------|-------------------|------|---|--------------|--------------|
| Read  | DACEN | DACRFS | DACTRGS<br>E<br>L | 0<br>DACS<br>WTRG | LPEN | 0 | DACBT<br>IEN | DACB<br>BIEN |
| Write | 0     | 0      | 0                 | 0                 | 0    | 0 | 0            | 0            |
| Reset | 0     | 0      | 0                 | 0                 | 0    | 0 | 0            | 0            |

#### DACx\_C0 field descriptions

| Field       | Description  |
|-------------|--|
| 7 DACEN     | DAC Enable<br>Starts the Programmable Reference Generator operation.<br>0 The DAC system is disabled.<br>1 The DAC system is enabled.  |
| 6 DACRFS    | DAC Reference Select<br>0 The DAC selects DACREF_1 as the reference voltage.<br>1 The DAC selects DACREF_2 as the reference voltage.   |
| 5 DACTRGSEL | DAC Trigger Select<br>0 The DAC hardware trigger is selected.<br>1 The DAC software trigger is selected.   |
| 4 DACSWTRG  | DAC Software Trigger<br>Active high. This is a write-only field, which always reads 0. If DAC software trigger is selected and buffer is enabled, writing 1 to this field will advance the buffer read pointer once.<br>0 The DAC soft trigger is not valid.<br>1 The DAC soft trigger is valid. |
| 3 LPEN      | DAC Low Power Control<br><b>NOTE:</b> See the 12-bit DAC electrical characteristics of the device data sheet for details on the impact of the modes below.<br>0 High-Power mode<br>1 Low-Power mode  |
| 2 Reserved  | This field is reserved.<br>This read-only field is reserved and always has the value 0.  |
| 1 DACBTIEN  | DAC Buffer Read Pointer Top Flag Interrupt Enable<br>0 The DAC buffer read pointer top flag interrupt is disabled.<br>1 The DAC buffer read pointer top flag interrupt is enabled.   |
| 0 DACBBIEN  | DAC Buffer Read Pointer Bottom Flag Interrupt Enable<br>0 The DAC buffer read pointer bottom flag interrupt is disabled.<br>1 The DAC buffer read pointer bottom flag interrupt is enabled.  |

### 30.4.5 DAC Control Register 1 (DACx\_C1)

Address: 4003\_F000h base + 22h offset = 4003\_F022h

| Bit   | 7     | 6 | 5 | 4 | 3       | 2 | 1       | 0 |
|-------|-------|---|---|---|---------|---|---------|---|
| Read  | DMAEN |   | 0 |   | DACBFMD | 0 | DACBFEN |   |
| Write | 0     | 0 | 0 | 0 | 0       | 0 | 0       | 0 |
| Reset | 0     | 0 | 0 | 0 | 0       | 0 | 0       | 0 |

**DACx\_C1 field descriptions**

| Field        | Description   |
|--------------|---|
| 7 DMAEN      | DMA Enable Select<br>0 DMA is disabled.<br>1 DMA is enabled. When DMA is enabled, the DMA request will be generated by original interrupts. The interrupts will not be presented on this module at the same time.   |
| 6–3 Reserved | This field is reserved.<br>This read-only field is reserved and always has the value 0.   |
| 2 DACBFMD    | DAC Buffer Work Mode Select<br>0 Normal mode<br>1 One-Time Scan mode  |
| 1 Reserved   | This field is reserved.<br>This read-only field is reserved and always has the value 0.   |
| 0 DACBFEN    | DAC Buffer Enable<br>0 Buffer read pointer is disabled. The converted data is always the first word of the buffer.<br>1 Buffer read pointer is enabled. The converted data is the word that the read pointer points to. It means converted data can be from any word of the buffer. |

### 30.4.6 DAC Control Register 2 (DACx\_C2)

Address: 4003\_F000h base + 23h offset = 4003\_F023h

| Bit   | 7 | 6 | 5       | 4 | 3 | 2       | 1 | 0 |
|-------|---|---|---------|---|---|---------|---|---|
| Read  | 0 |   | DACBFRP | 0 |   | DACBFUP |   |   |
| Write | 0 | 0 | 0       | 0 | 0 | 0       | 0 | 1 |
| Reset | 0 | 0 | 0       | 0 | 0 | 0       | 0 | 1 |

**DACx\_C2 field descriptions**

| Field        | Description   |
|--------------|---|
| 7–5 Reserved | This field is reserved.<br>This read-only field is reserved and always has the value 0. |
| 4 DACBFRP    | DAC Buffer Read Pointer<br>Keeps the current value of the buffer read pointer.          |

*Table continues on the next page...*

**DACx\_C2 field descriptions (continued)**

| Field           | Description  |
|-----------------|--|
| 3–1<br>Reserved | This field is reserved.<br>This read-only field is reserved and always has the value 0.                            |
| 0<br>DACBFUP    | DAC Buffer Upper Limit<br><br>Selects the upper limit of the DAC buffer. The buffer read pointer cannot exceed it. |

## 30.5 Functional description

The 12-bit DAC module can select one of the two reference inputs—DACREF\_1 and DACREF\_2 as the DAC reference voltage,  $V_{in}$  by C0[DACRFS]. See the module introduction for information on the source for DACREF\_1 and DACREF\_2.

When the DAC is enabled, it converts the data in DACDAT0[11:0] or the data from the DAC data buffer to a stepped analog output voltage. The output voltage range is from  $V_{in}$  to  $V_{in}/4096$ , and the step is  $V_{in}/4096$ .

### 30.5.1 DAC data buffer operation

When the DAC is enabled and the buffer is not enabled, the DAC module always converts the data in DAT0 to analog output voltage.

When both the DAC and the buffer are enabled, the DAC converts the data in the data buffer to analog output voltage. The data buffer read pointer advances to the next word whenever any hardware or software trigger event occurs. Refer to [Introduction](#) for the hardware trigger connection.

The data buffer can be configured to operate in Normal mode, Swing mode, or One-Time Scan mode. When the buffer operation is switched from one mode to another, the read pointer does not change. The read pointer can be set to any value between 0 and C2[DACBFUP] by writing C2[DACBFRP].

#### 30.5.1.1 DAC data buffer interrupts

There are several interrupts and associated flags that can be configured for the DAC buffer. SR[DACBFRPBF] is set when the DAC buffer read pointer reaches the DAC buffer upper limit, that is, C2[DACBFRP] = C2[DACBFUP]. SR[DACBFRPTF] is set when the DAC read pointer is equal to the start position, 0.

### 30.5.1.2 Modes of DAC data buffer operation

The following table describes the different modes of data buffer operation for the DAC module.

**Table 30-23. Modes of DAC data buffer operation**

| Modes                     | Description  |
|---------------------------|--|
| Buffer Normal mode        | This is the default mode. The buffer works as a circular buffer. The read pointer increases by one, every time the trigger occurs. When the read pointer reaches the upper limit, it goes to 0 directly in the next trigger event.   |
| Buffer One-time Scan mode | The read pointer increases by 1 every time the trigger occurs. When it reaches the upper limit, it stops there. If read pointer is reset to the address other than the upper limit, it will increase to the upper address and stop there again.<br><br><b>NOTE:</b> If the software set the read pointer to the upper limit, the read pointer will not advance in this mode. |

### 30.5.2 DMA operation

When DMA is enabled, DMA requests are generated instead of interrupt requests. The DMA Done signal clears the DMA request.

The status register flags are still set and are cleared automatically when the DMA completes.

### 30.5.3 Resets

During reset, the DAC is configured in the default mode and is disabled.

### 30.5.4 Low-Power mode operation

The following table shows the wait mode and the stop mode operation of the DAC module.

**Table 30-24. Modes of operation**

| Modes of operation | Description                                |
|--------------------|--|
| Wait mode          | The DAC will operate normally, if enabled. |

*Table continues on the next page...*

**Table 30-24. Modes of operation (continued)**

| Modes of operation | Description   |
|--------------------|---|
| Stop mode          | If enabled, the DAC module continues to operate in Normal Stop mode and the output voltage will hold the value before stop.<br>In low-power stop modes, the DAC is fully shut down. |

**NOTE**

The assignment of module modes to core modes is chip-specific. For module-to-core mode assignments, see the chapter that describes how modules are configured.



# **Chapter 31**

## **Timer/PWM Module (TPM)**

### **31.1 Introduction**

The TPM (Timer/PWM Module) is a two to eight channel timer which supports input capture, output compare, and the generation of PWM signals to control electric motor and power management applications. The counter, compare and capture registers are clocked by an asynchronous clock that can remain enabled in low power modes.

#### **31.1.1 TPM Philosophy**

The TPM is built upon a very simple timer (HCS08 Timer PWM Module – TPM) used for many years on Freescale's 8-bit microcontrollers. The TPM extends the functionality to support operation in low power modes by clocking the counter, compare and capture registers from an asynchronous clock that can remain functional in low power modes.

#### **31.1.2 Features**

The TPM features include:

- TPM clock mode is selectable
  - Can increment on every edge of the asynchronous counter clock
  - Can increment on rising edge of an external clock input synchronized to the asynchronous counter clock
- Prescaler divide-by 1, 2, 4, 8, 16, 32, 64, or 128
- TPM includes a 16-bit counter

- It can be a free-running counter or modulo counter
- The counting can be up or up-down
- Includes 6 channels that can be configured for input capture, output compare, or edge-aligned PWM mode
  - In input capture mode the capture can occur on rising edges, falling edges or both edges
  - In output compare mode the output signal can be set, cleared, pulsed, or toggled on match
  - All channels can be configured for center-aligned PWM mode
- Support the generation of an interrupt and/or DMA request per channel
- Support the generation of an interrupt and/or DMA request when the counter overflows
- Support selectable trigger input to optionally reset or cause the counter to start incrementing.
  - The counter can also optionally stop incrementing on counter overflow
- Support the generation of hardware triggers when the counter overflows and per channel

### **31.1.3 Modes of Operation**

During debug mode, the TPM can be configured to temporarily pause all counting until the core returns to normal user operating mode or to operate normally. When the counter is paused, trigger inputs and input capture events are ignored.

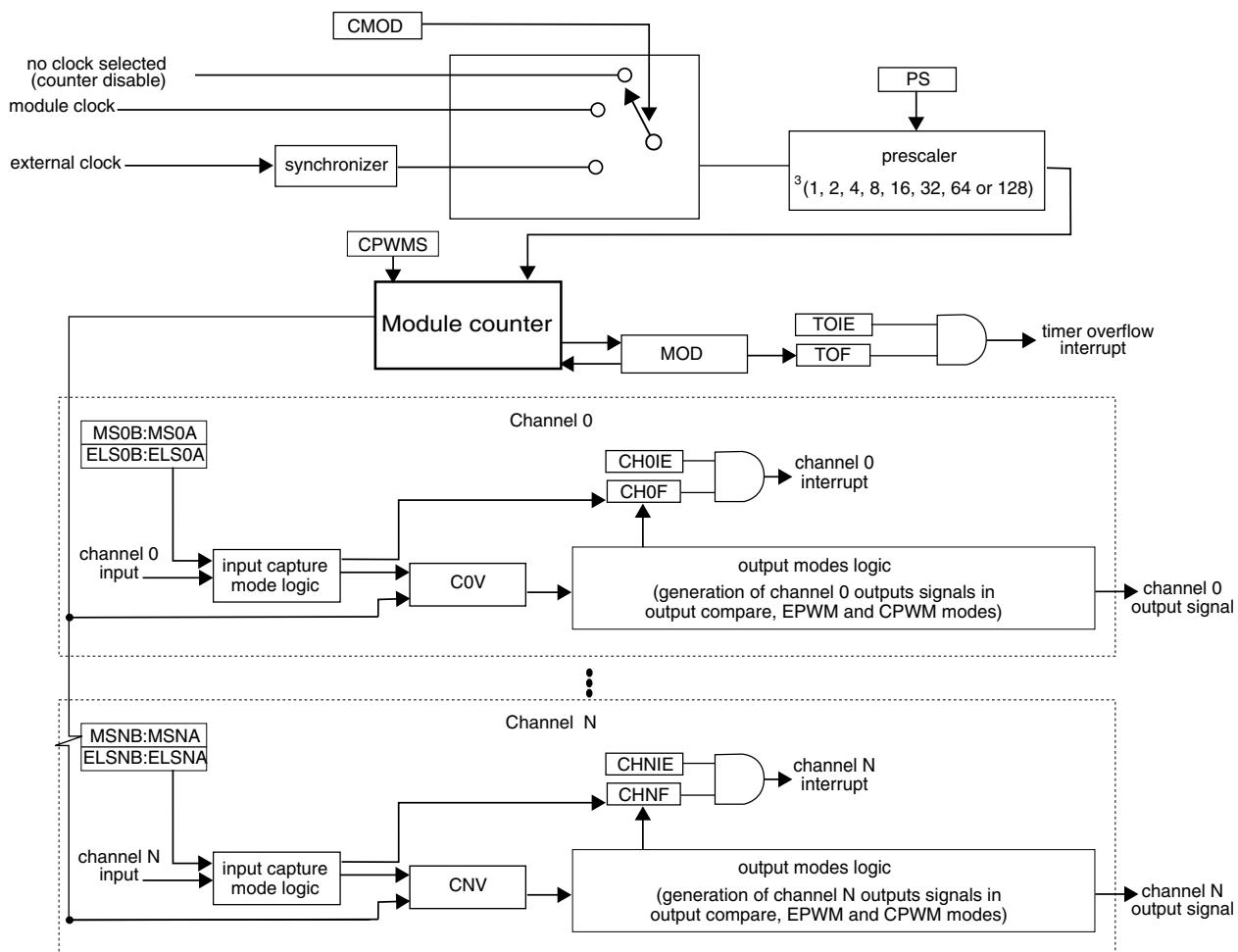
During doze mode, the TPM can be configured to operate normally or to pause all counting for the duration of doze mode. When the counter is paused, trigger inputs and input capture events are ignored.

During stop mode, the TPM counter clock can remain functional and the TPM can generate an asynchronous interrupt to exit the MCU from stop mode.

### **31.1.4 Block Diagram**

The TPM uses one input/output (I/O) pin per channel, CH<sub>n</sub> (TPM channel (n)) where n is the channel number.

The following figure shows the TPM structure. The central component of the TPM is the 16-bit counter with programmable final value and its counting can be up or up-down.



**Figure 31-1. TPM block diagram**

## 31.2 TPM Signal Descriptions

Table 31-1 shows the user-accessible signals for the TPM.

**Table 31-1. TPM signal descriptions**

| Signal     | Description   | I/O |
|------------|---|-----|
| TPM_EXTCLK | External clock. TPM external clock can be selected to increment the TPM counter on every rising edge synchronized to the counter clock. | I   |
| TPM_CHn    | TPM channel (n = 5 to 0)  | I/O |

### 31.2.1 TPM\_EXTCLK — TPM External Clock

The rising edge of the external input signal is used to increment the TPM counter if selected by CMOD[1:0] bits in the SC register. This input signal must be less than half of the TPM counter clock frequency. The TPM counter prescaler selection and settings are also used when an external input is selected.

### 31.2.2 TPM\_CHn — TPM Channel (n) I/O Pin

Each TPM channel can be configured to operate either as input or output. The direction associated with each channel, input or output, is selected according to the mode assigned for that channel.

## 31.3 Memory Map and Register Definition

This section provides a detailed description of all TPM registers.

Attempting to access a reserved register location in the TPM memory map will generate a bus error.

**TPM memory map**

| Absolute address (hex) | Register name                              | Width (in bits) | Access | Reset value | Section/ page              |
|------------------------|--|-----------------|--------|-------------|----------------------------|
| 4003_8000              | Status and Control (TPM0_SC)               | 32              | R/W    | 0000_0000h  | <a href="#">31.3.1/552</a> |
| 4003_8004              | Counter (TPM0_CNT)                         | 32              | R/W    | 0000_0000h  | <a href="#">31.3.2/553</a> |
| 4003_8008              | Modulo (TPM0_MOD)                          | 32              | R/W    | 0000_FFFFh  | <a href="#">31.3.3/554</a> |
| 4003_800C              | Channel (n) Status and Control (TPM0_C0SC) | 32              | R/W    | 0000_0000h  | <a href="#">31.3.4/555</a> |
| 4003_8010              | Channel (n) Value (TPM0_C0V)               | 32              | R/W    | 0000_0000h  | <a href="#">31.3.5/557</a> |
| 4003_8014              | Channel (n) Status and Control (TPM0_C1SC) | 32              | R/W    | 0000_0000h  | <a href="#">31.3.4/555</a> |
| 4003_8018              | Channel (n) Value (TPM0_C1V)               | 32              | R/W    | 0000_0000h  | <a href="#">31.3.5/557</a> |
| 4003_801C              | Channel (n) Status and Control (TPM0_C2SC) | 32              | R/W    | 0000_0000h  | <a href="#">31.3.4/555</a> |
| 4003_8020              | Channel (n) Value (TPM0_C2V)               | 32              | R/W    | 0000_0000h  | <a href="#">31.3.5/557</a> |
| 4003_8024              | Channel (n) Status and Control (TPM0_C3SC) | 32              | R/W    | 0000_0000h  | <a href="#">31.3.4/555</a> |
| 4003_8028              | Channel (n) Value (TPM0_C3V)               | 32              | R/W    | 0000_0000h  | <a href="#">31.3.5/557</a> |
| 4003_802C              | Channel (n) Status and Control (TPM0_C4SC) | 32              | R/W    | 0000_0000h  | <a href="#">31.3.4/555</a> |
| 4003_8030              | Channel (n) Value (TPM0_C4V)               | 32              | R/W    | 0000_0000h  | <a href="#">31.3.5/557</a> |
| 4003_8034              | Channel (n) Status and Control (TPM0_C5SC) | 32              | R/W    | 0000_0000h  | <a href="#">31.3.4/555</a> |
| 4003_8038              | Channel (n) Value (TPM0_C5V)               | 32              | R/W    | 0000_0000h  | <a href="#">31.3.5/557</a> |
| 4003_8050              | Capture and Compare Status (TPM0_STATUS)   | 32              | R/W    | 0000_0000h  | <a href="#">31.3.6/557</a> |
| 4003_8084              | Configuration (TPM0_CONF)                  | 32              | R/W    | 0000_0000h  | <a href="#">31.3.7/559</a> |

## TPM memory map (continued)

| Absolute address (hex) | Register name                              | Width (in bits) | Access | Reset value | Section/page               |
|------------------------|--|-----------------|--------|-------------|----------------------------|
| 4003_9000              | Status and Control (TPM1_SC)               | 32              | R/W    | 0000_0000h  | <a href="#">31.3.1/552</a> |
| 4003_9004              | Counter (TPM1_CNT)                         | 32              | R/W    | 0000_0000h  | <a href="#">31.3.2/553</a> |
| 4003_9008              | Modulo (TPM1_MOD)                          | 32              | R/W    | 0000_FFFFh  | <a href="#">31.3.3/554</a> |
| 4003_900C              | Channel (n) Status and Control (TPM1_C0SC) | 32              | R/W    | 0000_0000h  | <a href="#">31.3.4/555</a> |
| 4003_9010              | Channel (n) Value (TPM1_C0V)               | 32              | R/W    | 0000_0000h  | <a href="#">31.3.5/557</a> |
| 4003_9014              | Channel (n) Status and Control (TPM1_C1SC) | 32              | R/W    | 0000_0000h  | <a href="#">31.3.4/555</a> |
| 4003_9018              | Channel (n) Value (TPM1_C1V)               | 32              | R/W    | 0000_0000h  | <a href="#">31.3.5/557</a> |
| 4003_901C              | Channel (n) Status and Control (TPM1_C2SC) | 32              | R/W    | 0000_0000h  | <a href="#">31.3.4/555</a> |
| 4003_9020              | Channel (n) Value (TPM1_C2V)               | 32              | R/W    | 0000_0000h  | <a href="#">31.3.5/557</a> |
| 4003_9024              | Channel (n) Status and Control (TPM1_C3SC) | 32              | R/W    | 0000_0000h  | <a href="#">31.3.4/555</a> |
| 4003_9028              | Channel (n) Value (TPM1_C3V)               | 32              | R/W    | 0000_0000h  | <a href="#">31.3.5/557</a> |
| 4003_902C              | Channel (n) Status and Control (TPM1_C4SC) | 32              | R/W    | 0000_0000h  | <a href="#">31.3.4/555</a> |
| 4003_9030              | Channel (n) Value (TPM1_C4V)               | 32              | R/W    | 0000_0000h  | <a href="#">31.3.5/557</a> |
| 4003_9034              | Channel (n) Status and Control (TPM1_C5SC) | 32              | R/W    | 0000_0000h  | <a href="#">31.3.4/555</a> |
| 4003_9038              | Channel (n) Value (TPM1_C5V)               | 32              | R/W    | 0000_0000h  | <a href="#">31.3.5/557</a> |
| 4003_9050              | Capture and Compare Status (TPM1_STATUS)   | 32              | R/W    | 0000_0000h  | <a href="#">31.3.6/557</a> |
| 4003_9084              | Configuration (TPM1_CONF)                  | 32              | R/W    | 0000_0000h  | <a href="#">31.3.7/559</a> |
| 4003_A000              | Status and Control (TPM2_SC)               | 32              | R/W    | 0000_0000h  | <a href="#">31.3.1/552</a> |
| 4003_A004              | Counter (TPM2_CNT)                         | 32              | R/W    | 0000_0000h  | <a href="#">31.3.2/553</a> |
| 4003_A008              | Modulo (TPM2_MOD)                          | 32              | R/W    | 0000_FFFFh  | <a href="#">31.3.3/554</a> |
| 4003_A00C              | Channel (n) Status and Control (TPM2_C0SC) | 32              | R/W    | 0000_0000h  | <a href="#">31.3.4/555</a> |
| 4003_A010              | Channel (n) Value (TPM2_C0V)               | 32              | R/W    | 0000_0000h  | <a href="#">31.3.5/557</a> |
| 4003_A014              | Channel (n) Status and Control (TPM2_C1SC) | 32              | R/W    | 0000_0000h  | <a href="#">31.3.4/555</a> |
| 4003_A018              | Channel (n) Value (TPM2_C1V)               | 32              | R/W    | 0000_0000h  | <a href="#">31.3.5/557</a> |
| 4003_A01C              | Channel (n) Status and Control (TPM2_C2SC) | 32              | R/W    | 0000_0000h  | <a href="#">31.3.4/555</a> |
| 4003_A020              | Channel (n) Value (TPM2_C2V)               | 32              | R/W    | 0000_0000h  | <a href="#">31.3.5/557</a> |
| 4003_A024              | Channel (n) Status and Control (TPM2_C3SC) | 32              | R/W    | 0000_0000h  | <a href="#">31.3.4/555</a> |
| 4003_A028              | Channel (n) Value (TPM2_C3V)               | 32              | R/W    | 0000_0000h  | <a href="#">31.3.5/557</a> |
| 4003_A02C              | Channel (n) Status and Control (TPM2_C4SC) | 32              | R/W    | 0000_0000h  | <a href="#">31.3.4/555</a> |
| 4003_A030              | Channel (n) Value (TPM2_C4V)               | 32              | R/W    | 0000_0000h  | <a href="#">31.3.5/557</a> |
| 4003_A034              | Channel (n) Status and Control (TPM2_C5SC) | 32              | R/W    | 0000_0000h  | <a href="#">31.3.4/555</a> |
| 4003_A038              | Channel (n) Value (TPM2_C5V)               | 32              | R/W    | 0000_0000h  | <a href="#">31.3.5/557</a> |
| 4003_A050              | Capture and Compare Status (TPM2_STATUS)   | 32              | R/W    | 0000_0000h  | <a href="#">31.3.6/557</a> |
| 4003_A084              | Configuration (TPM2_CONF)                  | 32              | R/W    | 0000_0000h  | <a href="#">31.3.7/559</a> |

### 31.3.1 Status and Control (TPMx\_SC)

SC contains the overflow status flag and control bits used to configure the interrupt enable, module configuration and prescaler factor. These controls relate to all channels within this module.

Address: Base address + 0h offset

| Bit   | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 | 23  | 22   | 21    | 20   | 19 | 18 | 17 | 16 |
|-------|----|----|----|----|----|----|----|----|-----|------|-------|------|----|----|----|----|
| R     |    |    |    |    |    |    |    |    | 0   |      |       |      |    |    |    |    |
| W     |    |    |    |    |    |    |    |    |     |      |       |      |    |    |    |    |
| Reset | 0  | 0  | 0  | 0  | 0  | 0  | 0  | 0  | 0   | 0    | 0     | 0    | 0  | 0  | 0  | 0  |
| Bit   | 15 | 14 | 13 | 12 | 11 | 10 | 9  | 8  | 7   | 6    | 5     | 4    | 3  | 2  | 1  | 0  |
| R     |    |    |    |    |    |    |    |    | TOF |      |       |      |    |    |    |    |
| W     |    |    |    |    |    |    |    |    | DMA | TOIE | CPWMS | CMOD |    |    |    |    |
| Reset | 0  | 0  | 0  | 0  | 0  | 0  | 0  | 0  | w1c | 0    | 0     | 0    | 0  | 0  | 0  | 0  |

#### TPMx\_SC field descriptions

| Field            | Description  |
|------------------|--|
| 31–9<br>Reserved | This field is reserved.<br>This read-only field is reserved and always has the value 0.  |
| 8<br>DMA         | DMA Enable<br><br>Enables DMA transfers for the overflow flag.<br><br>0 Disables DMA transfers.<br>1 Enables DMA transfers.  |
| 7<br>TOF         | Timer Overflow Flag<br><br>Set by hardware when the LPTPM counter equals the value in the MOD register and increments. The TOF bit is cleared by writing a 1 to TOF bit. Writing a 0 to TOF has no effect.<br><br>If another LPTPM overflow occurs between the flag setting and the flag clearing, the write operation has no effect; therefore, TOF remains set indicating another overflow has occurred. In this case a TOF interrupt request is not lost due to a delay in clearing the previous TOF. |

*Table continues on the next page...*

**TPMx\_SC field descriptions (continued)**

| Field    | Description   |
|----------|---|
|          | <p>0 LPTPM counter has not overflowed.<br/>1 LPTPM counter has overflowed.</p>  |
| 6 TOIE   | <p>Timer Overflow Interrupt Enable<br/>Enables LPTPM overflow interrupts.</p> <p>0 Disable TOF interrupts. Use software polling or DMA request.<br/>1 Enable TOF interrupts. An interrupt is generated when TOF equals one.</p>   |
| 5 CPWMS  | <p>Center-aligned PWM Select<br/>Selects CPWM mode. This mode configures the LPTPM to operate in up-down counting mode.<br/>This field is write protected. It can be written only when the counter is disabled.</p> <p>0 LPTPM counter operates in up counting mode.<br/>1 LPTPM counter operates in up-down counting mode.</p>   |
| 4–3 CMOD | <p>Clock Mode Selection<br/>Selects the LPTPM counter clock modes. When disabling the counter, this field remains set until acknowledged in the LPTPM clock domain.</p> <p>00 LPTPM counter is disabled<br/>01 LPTPM counter increments on every LPTPM counter clock<br/>10 LPTPM counter increments on rising edge of LPTPM_EXTCLK synchronized to the LPTPM counter clock<br/>11 Reserved</p> |
| 2–0 PS   | <p>Prescale Factor Selection<br/>Selects one of 8 division factors for the clock mode selected by CMOD.<br/>This field is write protected. It can be written only when the counter is disabled.</p> <p>000 Divide by 1<br/>001 Divide by 2<br/>010 Divide by 4<br/>011 Divide by 8<br/>100 Divide by 16<br/>101 Divide by 32<br/>110 Divide by 64<br/>111 Divide by 128</p>                     |

### 31.3.2 Counter (TPMx\_CNT)

The CNT register contains the LPTPM counter value.

Reset clears the CNT register. Writing any value to COUNT also clears the counter.

When debug is active, the LPTPM counter does not increment unless configured otherwise.

## Memory Map and Register Definition

Reading the CNT register adds two wait states to the register access due to synchronization delays.

Address: Base address + 4h offset

| Bit   | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4     | 3 | 2 | 1 | 0 |
|-------|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|---|---|---|---|---|-------|---|---|---|---|
| R     |    |    |    |    |    |    |    |    |    |    |    |    |    |    |    | 0  |    |    |    |    |    |    |   |   |   |   |   |       |   |   |   |   |
| W     |    |    |    |    |    |    |    |    |    |    |    |    |    |    |    |    |    |    |    |    |    |    |   |   |   |   |   | COUNT |   |   |   |   |
| Reset | 0  | 0  | 0  | 0  | 0  | 0  | 0  | 0  | 0  | 0  | 0  | 0  | 0  | 0  | 0  | 0  | 0  | 0  | 0  | 0  | 0  | 0  | 0 | 0 | 0 | 0 | 0 | 0     | 0 | 0 |   |   |

### TPMx\_CNT field descriptions

| Field             | Description   |
|-------------------|---|
| 31–16<br>Reserved | This field is reserved.<br>This read-only field is reserved and always has the value 0. |
| 15–0<br>COUNT     | Counter value   |

### 31.3.3 Modulo (TPMx\_MOD)

The Modulo register contains the modulo value for the LPTPM counter. When the LPTPM counter reaches the modulo value and increments, the overflow flag (TOF) is set and the next value of LPTPM counter depends on the selected counting method (see [Counter](#) ).

Writing to the MOD register latches the value into a buffer. The MOD register is updated with the value of its write buffer according to [MOD Register Update](#) .

It is recommended to initialize the LPTPM counter (write to CNT) before writing to the MOD register to avoid confusion about when the first counter overflow will occur.

Address: Base address + 8h offset

| Bit   | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4   | 3 | 2 | 1 | 0 |
|-------|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|---|---|---|---|---|-----|---|---|---|---|
| R     |    |    |    |    |    |    |    |    |    |    |    |    |    |    |    | 0  |    |    |    |    |    |    |   |   |   |   |   |     |   |   |   |   |
| W     |    |    |    |    |    |    |    |    |    |    |    |    |    |    |    |    |    |    |    |    |    |    |   |   |   |   |   | MOD |   |   |   |   |
| Reset | 0  | 0  | 0  | 0  | 0  | 0  | 0  | 0  | 0  | 0  | 0  | 0  | 0  | 0  | 0  | 0  | 1  | 1  | 1  | 1  | 1  | 1  | 1 | 1 | 1 | 1 | 1 | 1   | 1 | 1 |   |   |

### TPMx\_MOD field descriptions

| Field             | Description  |
|-------------------|--|
| 31–16<br>Reserved | This field is reserved.<br>This read-only field is reserved and always has the value 0.  |
| 15–0<br>MOD       | Modulo value<br><br>When writing this field, all bytes must be written at the same time. |

### 31.3.4 Channel (n) Status and Control (TPMx\_CnSC)

CnSC contains the channel-interrupt-status flag and control bits used to configure the interrupt enable, channel configuration, and pin function. When switching from one channel mode to a different channel mode, the channel must first be disabled and this must be acknowledged in the LPTPM counter clock domain.

**Table 31-34. Mode, Edge, and Level Selection**

| CPWMS | MSnB:MSnA | ELSnB:ELSnA | Mode               | Configuration   |
|-------|-----------|-------------|--------------------|---|
| X     | 00        | 00          | None               | Channel disabled  |
| X     | 01/10/11  | 00          | Software compare   | Pin not used for LPTPM  |
| 0     | 00        | 01          | Input capture      | Capture on Rising Edge Only   |
|       |           | 10          |                    | Capture on Falling Edge Only  |
|       |           | 11          |                    | Capture on Rising or Falling Edge                                     |
|       | 01        | 01          | Output compare     | Toggle Output on match  |
|       |           | 10          |                    | Clear Output on match   |
|       |           | 11          |                    | Set Output on match   |
| 1     | 10        | 10          | Edge-aligned PWM   | High-true pulses (clear Output on match, set Output on reload)        |
|       |           | X1          |                    | Low-true pulses (set Output on match, clear Output on reload)         |
|       | 11        | 10          | Output compare     | Pulse Output low on match   |
|       |           | X1          |                    | Pulse Output high on match  |
|       | 10        | 10          | Center-aligned PWM | High-true pulses (clear Output on match-up, set Output on match-down) |
|       |           | X1          |                    | Low-true pulses (set Output on match-up, clear Output on match-down)  |

Address: Base address + Ch offset + (8d × i), where i=0d to 5d

| Bit   | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|-------|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|
| R     |    |    |    |    |    |    |    |    | 0  |    |    |    |    |    |    |    |
| W     |    |    |    |    |    |    |    |    |    |    |    |    |    |    |    |    |
| Reset | 0  | 0  | 0  | 0  | 0  | 0  | 0  | 0  | 0  | 0  | 0  | 0  | 0  | 0  | 0  | 0  |

## Memory Map and Register Definition

| Bit   | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8   | 7    | 6   | 5   | 4    | 3    | 2 | 1   | 0 |
|-------|----|----|----|----|----|----|---|-----|------|-----|-----|------|------|---|-----|---|
| R     | 0  |    |    |    |    |    |   | CHF | CHIE | MSB | MSA | ELSB | ELSA | 0 | DMA |   |
| W     |    |    |    |    |    |    |   | w1c |      |     |     |      |      |   |     |   |
| Reset | 0  | 0  | 0  | 0  | 0  | 0  | 0 | 0   | 0    | 0   | 0   | 0    | 0    | 0 | 0   | 0 |

## TPMx\_CnSC field descriptions

| Field            | Description   |
|------------------|---|
| 31–8<br>Reserved | This field is reserved.<br>This read-only field is reserved and always has the value 0.   |
| 7<br>CHF         | Channel Flag<br><br>Set by hardware when an event occurs on the channel. CHF is cleared by writing a 1 to the CHF bit. Writing a 0 to CHF has no effect.<br><br>If another event occurs between the CHF sets and the write operation, the write operation has no effect; therefore, CHF remains set indicating another event has occurred. In this case a CHF interrupt request is not lost due to the delay in clearing the previous CHF.<br><br>0 No channel event has occurred.<br>1 A channel event has occurred. |
| 6<br>CHIE        | Channel Interrupt Enable<br><br>Enables channel interrupts.<br><br>0 Disable channel interrupts.<br>1 Enable channel interrupts.  |
| 5<br>MSB         | Channel Mode Select<br><br>Used for further selections in the channel logic. Its functionality is dependent on the channel mode. When a channel is disabled, this bit will not change state until acknowledged in the LPTPM counter clock domain.   |
| 4<br>MSA         | Channel Mode Select<br><br>Used for further selections in the channel logic. Its functionality is dependent on the channel mode. When a channel is disabled, this bit will not change state until acknowledged in the LPTPM counter clock domain.   |
| 3<br>ELSB        | Edge or Level Select<br><br>The functionality of ELSB and ELSA depends on the channel mode. When a channel is disabled, this bit will not change state until acknowledged in the LPTPM counter clock domain.  |
| 2<br>ELSA        | Edge or Level Select<br><br>The functionality of ELSB and ELSA depends on the channel mode. When a channel is disabled, this bit will not change state until acknowledged in the LPTPM counter clock domain.  |
| 1<br>Reserved    | This field is reserved.<br>This read-only field is reserved and always has the value 0.   |
| 0<br>DMA         | DMA Enable<br><br>Enables DMA transfers for the channel.<br><br>0 Disable DMA transfers.<br>1 Enable DMA transfers.   |

### 31.3.5 Channel (n) Value (TPMx\_CnV)

These registers contain the captured LPTPM counter value for the input modes or the match value for the output modes.

In input capture mode, any write to a CnV register is ignored.

In compare modes, writing to a CnV register latches the value into a buffer. A CnV register is updated with the value of its write buffer according to [CnV Register Update](#).

Address: Base address + 10h offset + (8d × i), where i=0d to 5d

| Bit   | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|-------|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|---|---|---|---|---|---|---|---|---|---|
| R     |    |    |    |    |    |    |    |    |    |    |    |    |    |    |    |    |    |    |    |    |    |    |   |   |   |   |   |   |   |   |   |   |
| W     |    |    |    |    |    |    |    |    |    |    |    |    |    |    |    |    |    |    |    |    |    |    |   |   |   |   |   |   |   |   |   |   |
| Reset | 0  | 0  | 0  | 0  | 0  | 0  | 0  | 0  | 0  | 0  | 0  | 0  | 0  | 0  | 0  | 0  | 0  | 0  | 0  | 0  | 0  | 0  | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |   |   |

#### TPMx\_CnV field descriptions

| Field             | Description  |
|-------------------|--|
| 31–16<br>Reserved | This field is reserved.<br>This read-only field is reserved and always has the value 0.  |
| 15–0<br>VAL       | Channel Value<br>Captured LPTPM counter value of the input modes or the match value for the output modes. When writing this field, all bytes must be written at the same time. |

## Memory Map and Register Definition

Address: Base address + 50h offset

| Bit   | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24  |   | 23 | 22  | 21   | 20   | 19   | 18   | 17   | 16   |
|-------|----|----|----|----|----|----|----|-----|---|----|-----|------|------|------|------|------|------|
| R     |    |    |    |    |    |    |    |     | 0 |    |     |      |      |      |      |      |      |
| W     |    |    |    |    |    |    |    |     |   |    |     |      |      |      |      |      |      |
| Reset | 0  | 0  | 0  | 0  | 0  | 0  | 0  | 0   |   | 0  | 0   | 0    | 0    | 0    | 0    | 0    | 0    |
| Bit   | 15 | 14 | 13 | 12 | 11 | 10 | 9  | 8   |   | 7  | 6   | 5    | 4    | 3    | 2    | 1    | 0    |
| R     |    |    |    |    |    |    |    | TOF |   | 0  |     | CH5F | CH4F | CH3F | CH2F | CH1F | CH0F |
| W     |    |    |    |    |    |    |    | w1c |   |    | w1c | w1c  | w1c  | w1c  | w1c  | w1c  | w1c  |
| Reset | 0  | 0  | 0  | 0  | 0  | 0  | 0  | 0   |   | 0  | 0   | 0    | 0    | 0    | 0    | 0    | 0    |

### TPMx\_STATUS field descriptions

| Field            | Description   |
|------------------|---|
| 31–9<br>Reserved | This field is reserved.<br>This read-only field is reserved and always has the value 0.   |
| 8<br>TOF         | Timer Overflow Flag<br><br>See register description<br><br>0 LPTPM counter has not overflowed.<br>1 LPTPM counter has overflowed. |
| 7–6<br>Reserved  | This field is reserved.<br>This read-only field is reserved and always has the value 0.   |
| 5<br>CH5F        | Channel 5 Flag<br><br>See the register description.<br><br>0 No channel event has occurred.<br>1 A channel event has occurred.    |
| 4<br>CH4F        | Channel 4 Flag<br><br>See the register description.<br><br>0 No channel event has occurred.<br>1 A channel event has occurred.    |
| 3<br>CH3F        | Channel 3 Flag<br><br>See the register description.<br><br>0 No channel event has occurred.<br>1 A channel event has occurred.    |

Table continues on the next page...

**TPMx\_STATUS field descriptions (continued)**

| Field     | Description  |
|-----------|--|
| 2<br>CH2F | Channel 2 Flag<br><br>See the register description.<br><br>0 No channel event has occurred.<br>1 A channel event has occurred. |
| 1<br>CH1F | Channel 1 Flag<br><br>See the register description.<br><br>0 No channel event has occurred.<br>1 A channel event has occurred. |
| 0<br>CH0F | Channel 0 Flag<br><br>See the register description.<br><br>0 No channel event has occurred.<br>1 A channel event has occurred. |

**31.3.7 Configuration (TPMx\_CONF)**

This register selects the behavior in debug and wait modes and the use of an external global time base.

Address: Base address + 84h offset

|       |    |    |    |    |    |    |    |    |    |    |    |    |    |      |      |      |
|-------|----|----|----|----|----|----|----|----|----|----|----|----|----|------|------|------|
| Bit   | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 | 23 | 22 | 21 | 20 | 19 | 18   | 17   | 16   |
| R     | 0  |    |    |    |    |    |    |    | 0  |    |    |    |    | CROT | CSOO | CSOT |
| W     |    |    |    |    |    |    |    |    |    |    |    |    |    |      |      |      |
| Reset | 0  | 0  | 0  | 0  | 0  | 0  | 0  | 0  | 0  | 0  | 0  | 0  | 0  | 0    | 0    | 0    |
| Bit   | 15 | 14 | 13 | 12 | 11 | 10 | 9  | 8  | 7  | 6  | 5  | 4  | 3  | 2    | 1    | 0    |
| R     | 0  |    |    |    |    |    |    |    |    |    |    |    |    | 0    |      |      |
| W     |    |    |    |    |    |    |    |    |    |    |    |    |    |      |      |      |
| Reset | 0  | 0  | 0  | 0  | 0  | 0  | 0  | 0  | 0  | 0  | 0  | 0  | 0  | 0    | 0    | 0    |

**TPMx\_CONF field descriptions**

| Field             | Description   |
|-------------------|---|
| 31–28<br>Reserved | This field is reserved.<br>This read-only field is reserved and always has the value 0. |

*Table continues on the next page...*

## TPMx\_CONF field descriptions (continued)

| Field             | Description   |
|-------------------|---|
| 27–24<br>TRGSEL   | <p>Trigger Select</p> <p>Selects the input trigger to use for starting the counter and/or reloading the counter. This field should only be changed when the LPTPM counter is disabled. See Chip configuration section for available options.</p>  |
| 23–19<br>Reserved | <p>This field is reserved.</p> <p>This read-only field is reserved and always has the value 0.</p>  |
| 18<br>CROT        | <p>Counter Reload On Trigger</p> <p>When set, the LPTPM counter will reload with zero (and initialize PWM outputs to their default value) when a rising edge is detected on the selected trigger input.</p> <p>The trigger input is ignored if the LPTPM counter is paused during debug mode or doze mode. This field should only be changed when the LPTPM counter is disabled.</p> <p>0 Counter is not reloaded due to a rising edge on the selected input trigger<br/>1 Counter is reloaded when a rising edge is detected on the selected input trigger</p>   |
| 17<br>CSOO        | <p>Counter Stop On Overflow</p> <p>When set, the LPTPM counter will stop incrementing once the counter equals the MOD value and incremented (this also sets the TOF). Reloading the counter with zero due to writing to the counter register or due to a trigger input does not cause the counter to stop incrementing. Once the counter has stopped incrementing, the counter will not start incrementing unless it is disabled and then enabled again, or a rising edge on the selected trigger input is detected when CSOT set.</p> <p>This field should only be changed when the LPTPM counter is disabled.</p> <p>0 LPTPM counter continues incrementing or decrementing after overflow<br/>1 LPTPM counter stops incrementing or decrementing after overflow.</p>             |
| 16<br>CSOT        | <p>Counter Start on Trigger</p> <p>When set, the LPTPM counter will not start incrementing after it is enabled until a rising edge on the selected trigger input is detected. If the LPTPM counter is stopped due to an overflow, a rising edge on the selected trigger input will also cause the LPTPM counter to start incrementing again.</p> <p>The trigger input is ignored if the LPTPM counter is paused during debug mode or doze mode. This field should only be changed when the LPTPM counter is disabled.</p> <p>0 LPTPM counter starts to increment immediately, once it is enabled.<br/>1 LPTPM counter only starts to increment when a rising edge on the selected input trigger is detected, after it has been enabled or after it has stopped due to overflow.</p> |
| 15–10<br>Reserved | <p>This field is reserved.</p> <p>This read-only field is reserved and always has the value 0.</p>  |
| 9<br>GTBEEN       | <p>Global time base enable</p> <p>Configures the LPTPM to use an externally generated global time base counter. When an externally generated timebase is used, the internal LPTPM counter is not used by the channels but can be used to generate a periodic interrupt or DMA request using the Modulo register and timer overflow flag.</p> <p>0 All channels use the internally generated LPTPM counter as their timebase<br/>1 All channels use an externally generated global timebase as their timebase</p>  |
| 8<br>Reserved     | <p>This field is reserved.</p> <p>This read-only field is reserved and always has the value 0.</p>  |
| 7–6<br>DBGMODE    | Debug Mode  |

Table continues on the next page...

**TPMx\_CONF field descriptions (continued)**

| Field        | Description   |
|--------------|---|
|              | <p>Configures the LPTPM behavior in debug mode. All other configurations are reserved.</p> <p>00 LPTPM counter is paused and does not increment during debug mode. Trigger inputs and input capture events are also ignored.</p> <p>11 LPTPM counter continues in debug mode.</p> |
| 5 DOZEEN     | <p>Doze Enable</p> <p>Configures the LPTPM behavior in wait mode.</p> <p>0 Internal LPTPM counter continues in Doze mode.</p> <p>1 Internal LPTPM counter is paused and does not increment during Doze mode. Trigger inputs and input capture events are also ignored.</p>        |
| 4–0 Reserved | <p>This field is reserved.</p> <p>This read-only field is reserved and always has the value 0.</p>  |

## 31.4 Functional Description

The following sections describe the TPM features.

### 31.4.1 Clock Domains

The TPM module supports two clock domains.

The bus clock domain is used by the register interface and for synchronizing interrupts and DMA requests.

The TPM counter clock domain is used to clock the counter and prescaler along with the output compare and input capture logic. The TPM counter clock is considered asynchronous to the bus clock, can be a higher or lower frequency than the bus clock and can remain operational in Stop mode. Multiple TPM instances are all clocked by the same TPM counter clock in support of the external timebase feature.

#### 31.4.1.1 Counter Clock Mode

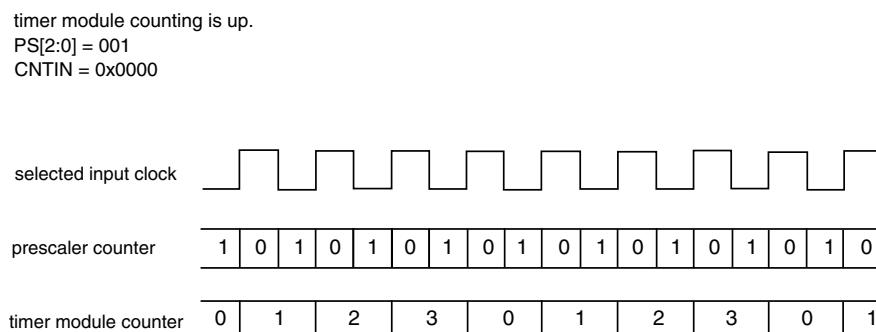
The CMOD[1:0] bits in the SC register either disable the TPM counter or select one of two possible clock modes for the TPM counter. After any reset, CMOD[1:0] = 0:0 so the TPM counter is disabled.

The CMOD[1:0] bits may be read or written at any time. Disabling the TPM counter by writing zero to the CMOD[1:0] bits does not affect the TPM counter value or other registers, but must be acknowledged by the TPM counter clock domain before they read as zero.

The external clock input passes through a synchronizer clocked by the TPM counter clock to assure that counter transitions are properly aligned to counter clock transitions. Therefore, to meet Nyquist criteria considering also jitter, the frequency of the external clock source must be less than half of the counter clock frequency.

### 31.4.2 Prescaler

The selected counter clock source passes through a prescaler that is a 7-bit counter. The value of the prescaler is selected by the PS[2:0] bits. The following figure shows an example of the prescaler counter and TPM counter.



**Figure 31-78. Example of the Prescaler Counter**

### 31.4.3 Counter

The TPM has a 16-bit counter that is used by the channels either for input or output modes. The counter updates from the selected clock divided by the prescaler.

The TPM counter has these modes of operation:

- up counting (see [Up Counting](#))
- up-down counting (see [Up-Down Counting](#))

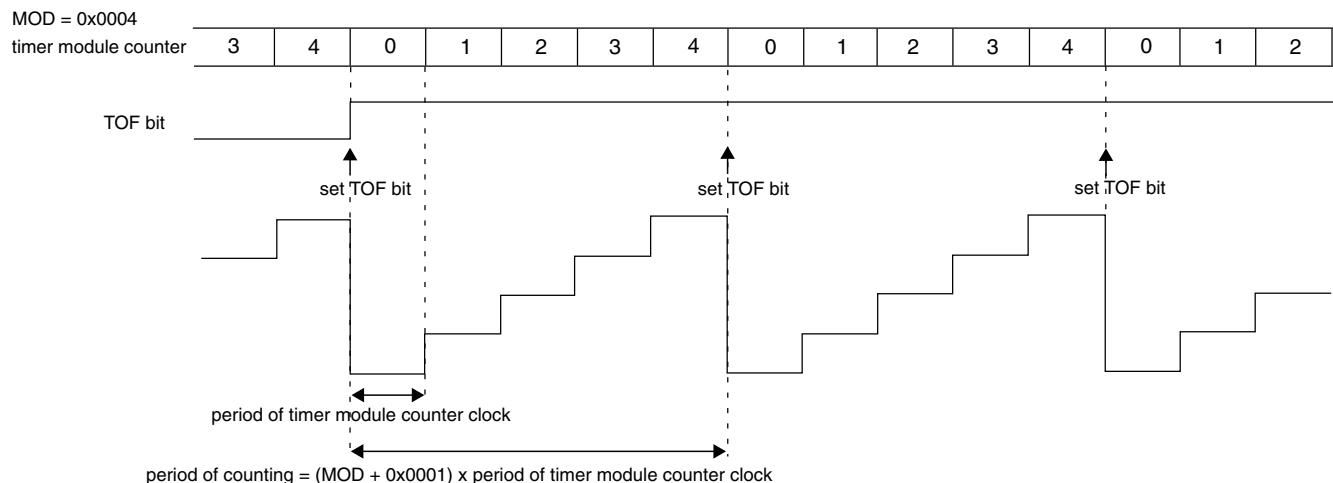
#### 31.4.3.1 Up Counting

Up counting is selected when (CPWMS = 0)

The value of zero is loaded into the TPM counter, and the counter increments until the value of MOD is reached, at which point the counter is reloaded with zero.

The TPM period when using up counting is  $(MOD + 0x0001) \times$  period of the TPM counter clock.

The TOF bit is set when the TPM counter changes from MOD to zero.



**Figure 31-79. Example of TPM Up Counting**

### Note

- MOD = 0000 is a redundant condition. In this case, the TPM counter is always equal to MOD and the TOF bit is set in each rising edge of the TPM counter clock.

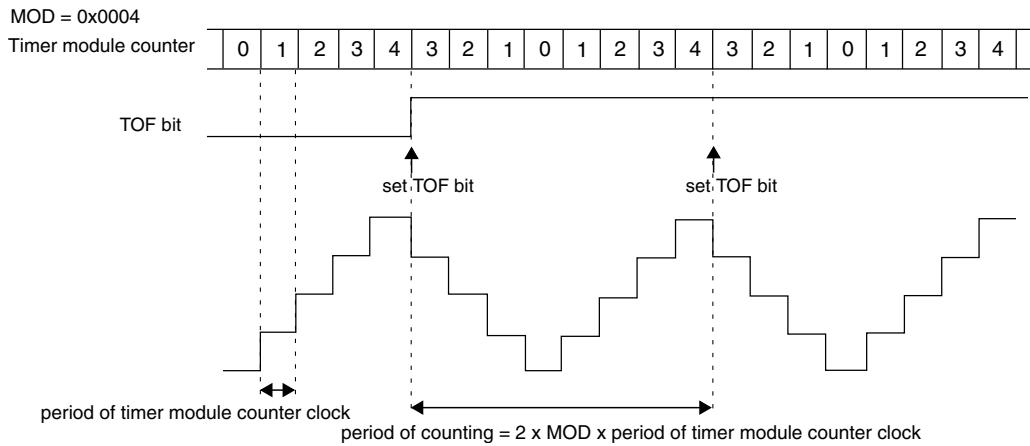
#### 31.4.3.2 Up-Down Counting

Up-down counting is selected when (CPWMS = 1). When configured for up-down counting, configuring MOD to less than 2 is not supported.

The value of zero is loaded into the TPM counter, and the counter increments until the value of MOD is reached, at which point the counter is decremented until it returns to zero and the up-down counting restarts.

The TPM period when using up-down counting is  $2 \times MOD \times$  period of the TPM counter clock.

The TOF bit is set when the TPM counter changes from MOD to (MOD – 1).



**Figure 31-80. Example of Up-Down Counting**

### 31.4.3.3 Counter Reset

Any write to CNT resets the TPM counter and the channel outputs to their initial values (except for channels in output compare mode).

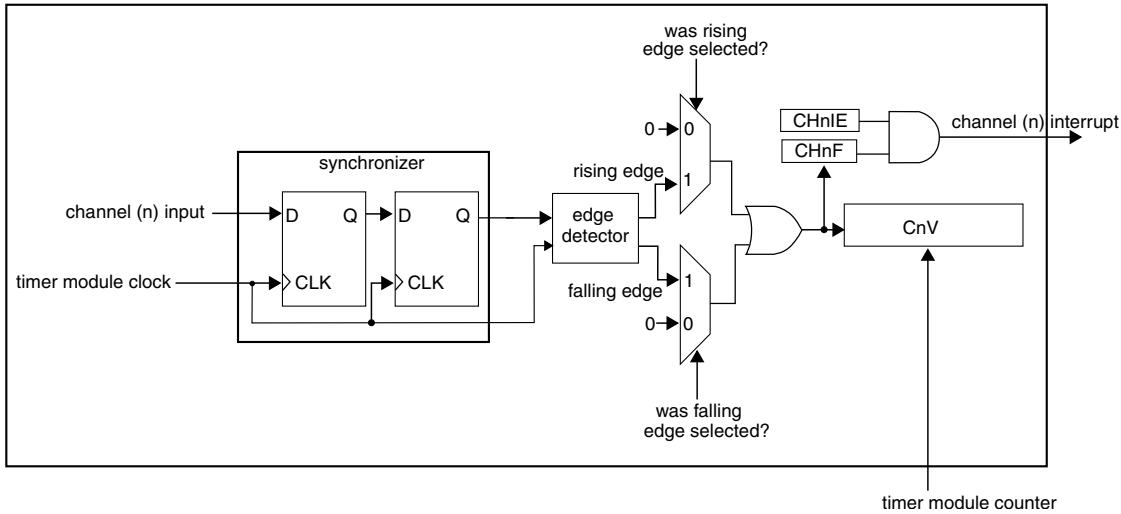
### 31.4.4 Input Capture Mode

The input capture mode is selected when (CPWMS = 0), (MSnB:MSnA = 0:0), and (ELSnB:ELSnA ≠ 0:0).

When a selected edge occurs on the channel input, the current value of the TPM counter is captured into the CnV register, at the same time the CHnF bit is set and the channel interrupt is generated if enabled by CHnIE = 1 (see the following figure).

When a channel is configured for input capture, the TPM\_CHn pin is an edge-sensitive input. ELSnB:ELSnA control bits determine which edge, falling or rising, triggers input-capture event. Note that the maximum frequency for the channel input signal to be detected correctly is counter clock divided by 4, which is required to meet Nyquist criteria for signal sampling.

Writes to the CnV register are ignored in input capture mode.



**Figure 31-81. Input capture mode**

The CHnF bit is set on the third rising edge of the counter clock after a valid edge occurs on the channel input.

### 31.4.5 Output Compare Mode

The output compare mode is selected when (CPWMS = 0), and (MSnB:MSnA = 0:1).

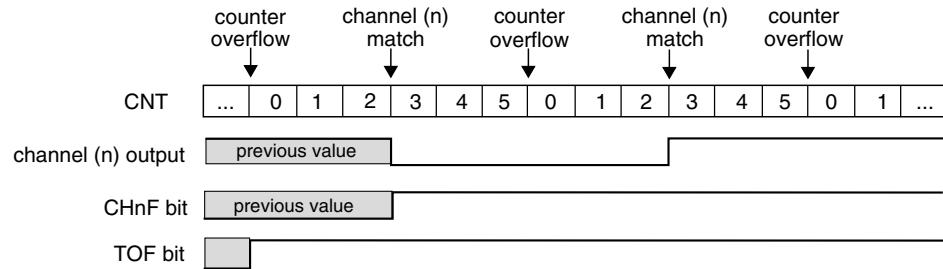
In output compare mode, the TPM can generate timed pulses with programmable position, polarity, duration, and frequency. When the counter matches the value in the CnV register of an output compare channel, the channel (n) output can be set, cleared, or toggled.

When a channel is initially configured to output compare mode, the channel output updates with its negated value (logic 0 for set/toggle/pulse high and logic one for clear/pulse low).

The CHnF bit is set and the channel (n) interrupt is generated (if CHnIE = 1) at the channel (n) match (TPM counter = CnV).

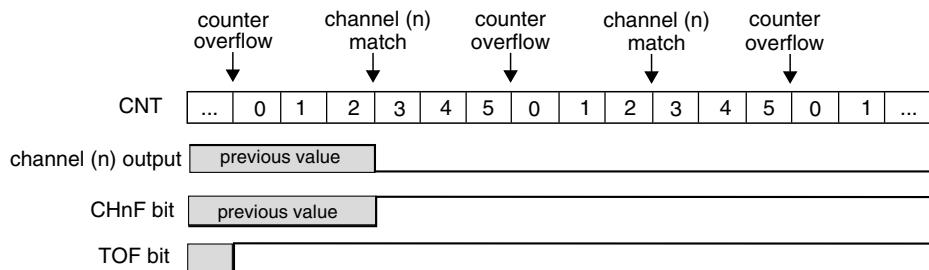
## Functional Description

MOD = 0x0005  
CnV = 0x0003



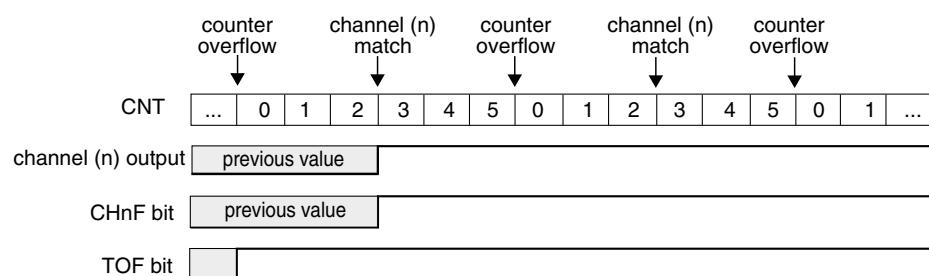
**Figure 31-82. Example of the output compare mode when the match toggles the channel output**

MOD = 0x0005  
CnV = 0x0003



**Figure 31-83. Example of the output compare mode when the match clears the channel output**

MOD = 0x0005  
CnV = 0x0003



**Figure 31-84. Example of the output compare mode when the match sets the channel output**

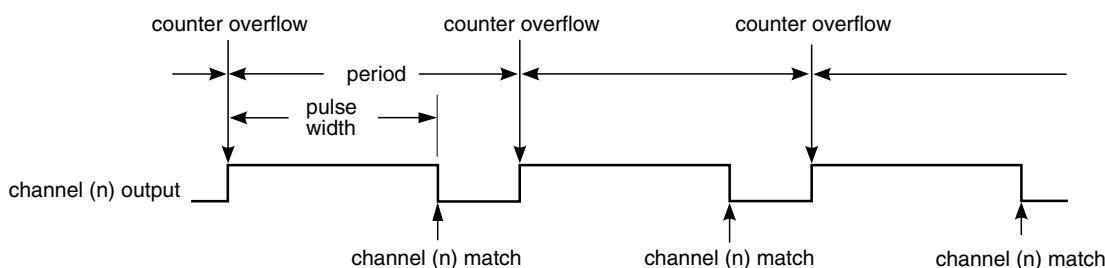
It is possible to use the output compare mode with (ELSnB:ELSnA = 0:0). In this case, when the counter reaches the value in the CnV register, the CHnF bit is set and the channel (n) interrupt is generated (if CHnIE = 1), however the channel (n) output is not modified and controlled by TPM.

### 31.4.6 Edge-Aligned PWM (EPWM) Mode

The edge-aligned mode is selected when ( $CPWMS = 0$ ), and ( $MSnB:MSnA = 1:0$ ). The EPWM period is determined by ( $MOD + 0x0001$ ) and the pulse width (duty cycle) is determined by  $CnV$ .

The  $CHnF$  bit is set and the channel (n) interrupt is generated (if  $CHnIE = 1$ ) at the channel (n) match (TPM counter =  $CnV$ ), that is, at the end of the pulse width.

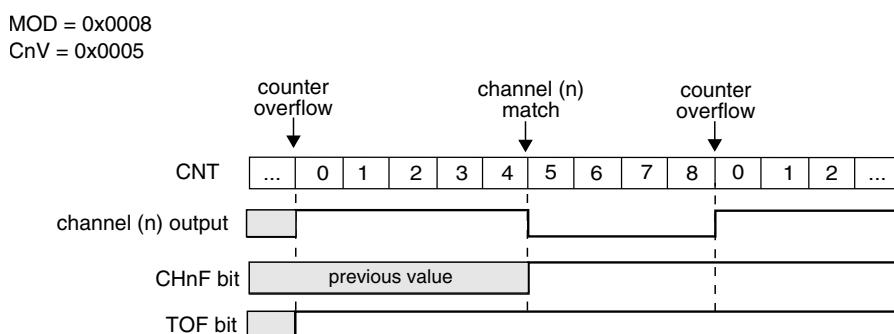
This type of PWM signal is called edge-aligned because the leading edges of all PWM signals are aligned with the beginning of the period, which is the same for all channels within an TPM.



**Figure 31-85. EPWM period and pulse width with  $ELSnB:ELSnA = 1:0$**

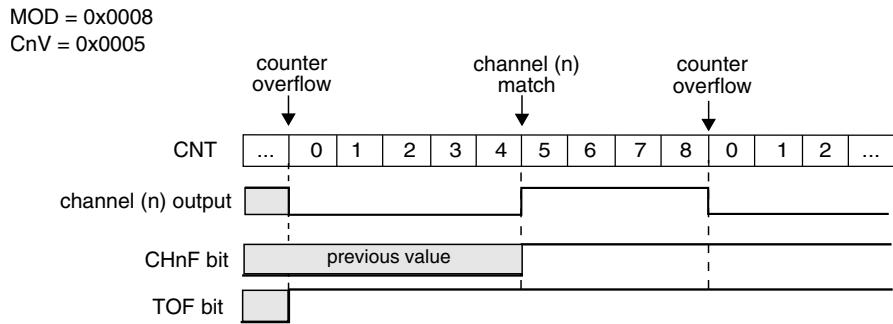
If ( $ELSnB:ELSnA = 0:0$ ) when the counter reaches the value in the  $CnV$  register, the  $CHnF$  bit is set and the channel (n) interrupt is generated (if  $CHnIE = 1$ ), however the channel (n) output is not controlled by TPM.

If ( $ELSnB:ELSnA = 1:0$ ), then the channel (n) output is forced high at the counter overflow (when the zero is loaded into the TPM counter), and it is forced low at the channel (n) match (TPM counter =  $CnV$ ) (see the following figure).



**Figure 31-86. EPWM signal with  $ELSnB:ELSnA = 1:0$**

If ( $ELSnB:ELSnA = X:1$ ), then the channel (n) output is forced low at the counter overflow (when zero is loaded into the TPM counter), and it is forced high at the channel (n) match (TPM counter =  $CnV$ ) (see the following figure).



**Figure 31-87. EPWM signal with ELSnB:ELSnA = X:1**

If ( $CnV = 0x0000$ ), then the channel (n) output is a 0% duty cycle EPWM signal. If ( $CnV > MOD$ ), then the channel (n) output is a 100% duty cycle EPWM signal and CHnF bit is not set since there is never a channel (n) match. Therefore, MOD must be less than 0xFFFF in order to get a 100% duty cycle EPWM signal.

### 31.4.7 Center-Aligned PWM (CPWM) Mode

The center-aligned mode is selected when ( $CPWMS = 1$ ) and ( $MSnB:MSnA = 1:0$ ).

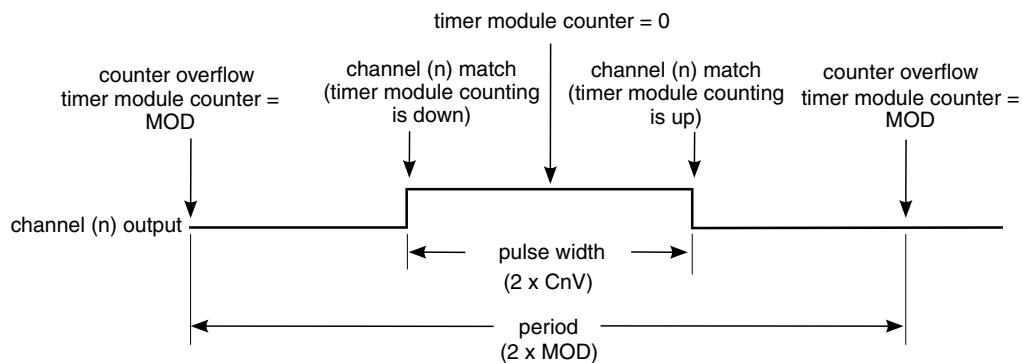
The CPWM pulse width (duty cycle) is determined by  $2 \times CnV$  and the period is determined by  $2 \times MOD$  (see the following figure). MOD must be kept in the range of 0x0001 to 0x7FFF because values outside this range can produce ambiguous results.

In the CPWM mode, the TPM counter counts up until it reaches MOD and then counts down until it reaches zero.

The CHnF bit is set and channel (n) interrupt is generated (if CHnIE = 1) at the channel (n) match (TPM counter = CnV) when the TPM counting is down (at the begin of the pulse width) and when the TPM counting is up (at the end of the pulse width).

This type of PWM signal is called center-aligned because the pulse width centers for all channels are when the TPM counter is zero.

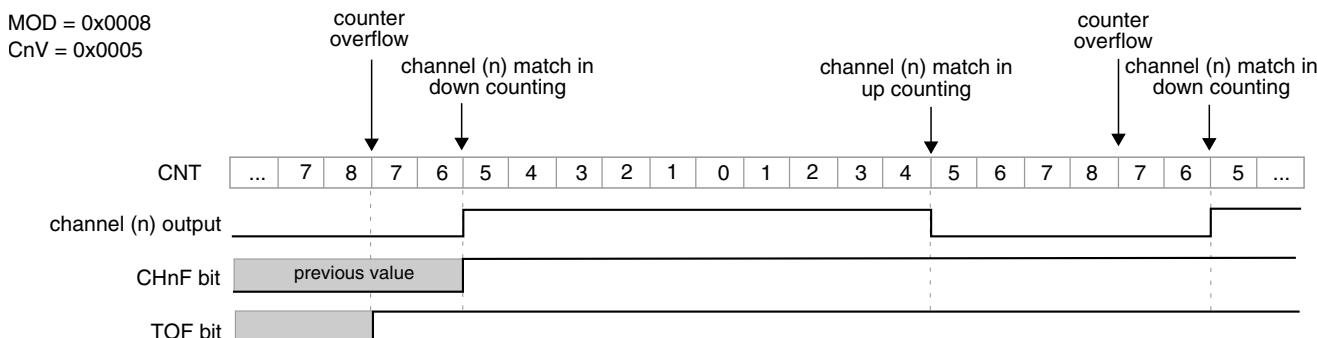
The other channel modes are not designed to be used with the up-down counter ( $CPWMS = 1$ ). Therefore, all TPM channels should be used in CPWM mode when ( $CPWMS = 1$ ).



**Figure 31-88. CPWM period and pulse width with ELSnB:ELSnA = 1:0**

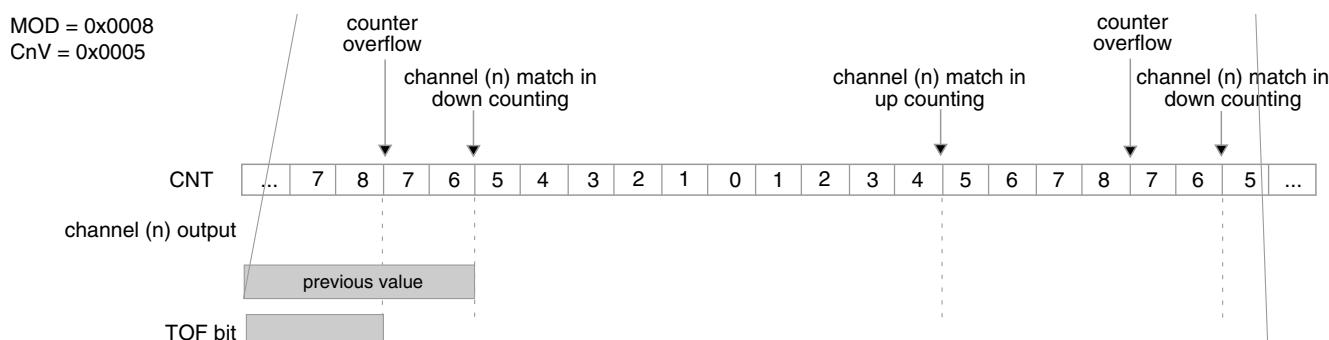
If (ELSnB:ELSnA = 0:0) when the TPM counter reaches the value in the CnV register, the CHnF bit is set and the channel (n) interrupt is generated (if CHnIE = 1), however the channel (n) output is not controlled by TPM.

If (ELSnB:ELSnA = 1:0), then the channel (n) output is forced high at the channel (n) match (TPM counter = CnV) when counting down, and it is forced low at the channel (n) match when counting up (see the following figure).



**Figure 31-89. CPWM signal with ELSnB:ELSnA = 1:0**

If (ELSnB:ELSnA = X:1), then the channel (n) output is forced low at the channel (n) match (TPM counter = CnV) when counting down, and it is forced high at the channel (n) match when counting up (see the following figure).



**Figure 31-90. CPWM signal with ELSnB:ELSnA = X:1**

If ( $CnV = 0x0000$ ) then the channel (n) output is a 0% duty cycle CPWM signal.

If ( $CnV > MOD$ ), then the channel (n) output is a 100% duty cycle CPWM signal, although the CHnF bit is set when the counter changes from incrementing to decrementing. Therefore, MOD must be less than 0xFFFF in order to get a 100% duty cycle CPWM signal.

## 31.4.8 Registers Updated from Write Buffers

### 31.4.8.1 MOD Register Update

If ( $CMOD[1:0] = 0:0$ ) then MOD register is updated when MOD register is written.

If ( $CMOD[1:0] \neq 0:0$ ), then MOD register is updated according to the CPWMS bit, that is:

- If the selected mode is not CPWM then MOD register is updated after MOD register was written and the TPM counter changes from MOD to zero.
- If the selected mode is CPWM then MOD register is updated after MOD register was written and the TPM counter changes from MOD to  $(MOD - 1)$ .

### 31.4.8.2 CnV Register Update

If ( $CMOD[1:0] = 0:0$ ) then CnV register is updated when CnV register is written.

If ( $CMOD[1:0] \neq 0:0$ ), then CnV register is updated according to the selected mode, that is:

- If the selected mode is output compare then CnV register is updated on the next TPM counter increment (end of the prescaler counting) after CnV register was written.
- If the selected mode is EPWM then CnV register is updated after CnV register was written and the TPM counter changes from MOD to zero.
- If the selected mode is CPWM then CnV register is updated after CnV register was written and the TPM counter changes from MOD to  $(MOD - 1)$ .

### 31.4.9 DMA

The channel generates a DMA transfer request according to DMA and CHnIE bits (see the following table).

**Table 31-110. Channel DMA Transfer Request**

| DMA | CHnIE | Channel DMA Transfer Request                                 | Channel Interrupt                                 |
|-----|-------|--|---|
| 0   | 0     | The channel DMA transfer request is not generated.           | The channel interrupt is not generated.           |
| 0   | 1     | The channel DMA transfer request is not generated.           | The channel interrupt is generated if (CHnF = 1). |
| 1   | 0     | The channel DMA transfer request is generated if (CHnF = 1). | The channel interrupt is not generated.           |
| 1   | 1     | The channel DMA transfer request is generated if (CHnF = 1). | The channel interrupt is generated if (CHnF = 1). |

If DMA = 1, the CHnF bit can be cleared either by channel DMA transfer done or writing a one to CHnF bit (see the following table).

**Table 31-111. Clear CHnF Bit**

| DMA | How CHnF Bit Can Be Cleared   |
|-----|---|
| 0   | CHnF bit is cleared by writing a 1 to CHnF bit.   |
| 1   | CHnF bit is cleared either when the channel DMA transfer is done or by writing a 1 to CHnF bit. |

### 31.4.10 Reset Overview

The TPM is reset whenever any chip reset occurs.

When the TPM exits from reset:

- the TPM counter and the prescaler counter are zero and are stopped (CMOD[1:0] = 0:0);
- the timer overflow interrupt is zero;
- the channels interrupts are zero;
- the channels are in input capture mode;
- the channels outputs are zero;
- the channels pins are not controlled by TPM (ELS(n)B:ELS(n)A = 0:0).

### 31.4.11 TPM Interrupts

This section describes TPM interrupts.

### 31.4.11.1 Timer Overflow Interrupt

The timer overflow interrupt is generated when (TOIE = 1) and (TOF = 1).

### 31.4.11.2 Channel (n) Interrupt

The channel (n) interrupt is generated when (CHnIE = 1) and (CHnF = 1).

# **Chapter 32**

## **Periodic Interrupt Timer (PIT)**

### **32.1 Introduction**

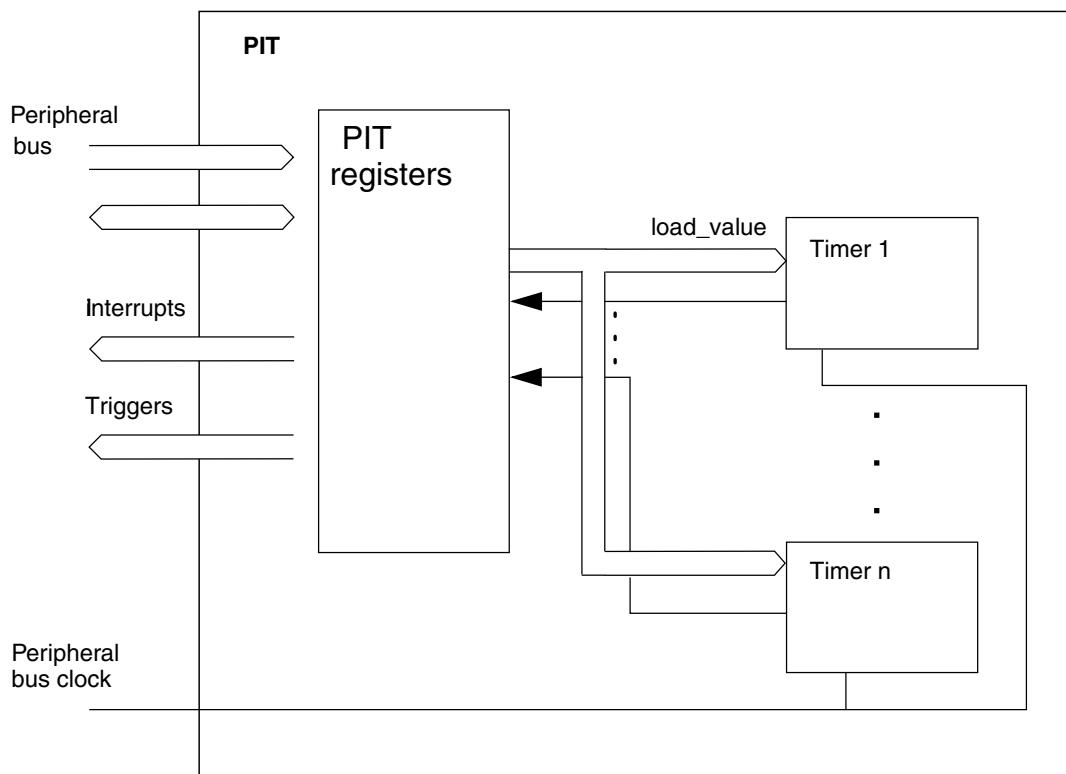
#### **NOTE**

For the chip-specific implementation details of this module's instances see the chip configuration information.

The PIT module is an array of timers that can be used to raise interrupts and trigger DMA channels.

#### **32.1.1 Block diagram**

The following figure shows the block diagram of the PIT module.



**Figure 32-1. Block diagram of the PIT**

### NOTE

See the chip configuration details for the number of PIT channels used in this MCU.

## 32.1.2 Features

The main features of this block are:

- Ability of timers to generate DMA trigger pulses
- Ability of timers to generate interrupts
- Maskable interrupts
- Independent timeout periods for each timer

## 32.2 Signal description

The PIT module has no external pins.

## 32.3 Memory map/register description

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

### NOTE

- Reserved registers will read as 0, writes will have no effect.
- See the chip configuration details for the number of PIT channels used in this MCU.

**Table 32-2. Timer Channel n / RTI Channel**

| Address Offset | Use                          | Access |
|----------------|------------------------------|--------|
| Channel + 0x00 | Timer Load Value Register    | R/W    |
| Channel + 0x04 | Current Timer Value Register | R      |
| Channel + 0x08 | Timer Control Register       | R/W    |
| Channel + 0x0C | Timer Flag Register          | R/W    |

### PIT memory map

| Absolute address (hex) | Register name                                   | Width (in bits) | Access | Reset value | Section/ page              |
|------------------------|---|-----------------|--------|-------------|----------------------------|
| 4003_7000              | PIT Module Control Register (PIT_MCR)           | 32              | R/W    | 0000_0002h  | <a href="#">32.3.1/575</a> |
| 4003_70E0              | PIT Upper Lifetime Timer Register (PIT_LTMR64H) | 32              | R      | 0000_0000h  | <a href="#">32.3.2/577</a> |
| 4003_70E4              | PIT Lower Lifetime Timer Register (PIT_LTMR64L) | 32              | R      | 0000_0000h  | <a href="#">32.3.3/577</a> |
| 4003_7100              | Timer Load Value Register (PIT_LDVAL0)          | 32              | R/W    | 0000_0000h  | <a href="#">32.3.4/578</a> |
| 4003_7104              | Current Timer Value Register (PIT_CVAL0)        | 32              | R      | 0000_0000h  | <a href="#">32.3.5/578</a> |
| 4003_7108              | Timer Control Register (PIT_TCTRL0)             | 32              | R/W    | 0000_0000h  | <a href="#">32.3.6/579</a> |
| 4003_710C              | Timer Flag Register (PIT_TFLG0)                 | 32              | R/W    | 0000_0000h  | <a href="#">32.3.7/580</a> |
| 4003_7110              | Timer Load Value Register (PIT_LDVAL1)          | 32              | R/W    | 0000_0000h  | <a href="#">32.3.4/578</a> |
| 4003_7114              | Current Timer Value Register (PIT_CVAL1)        | 32              | R      | 0000_0000h  | <a href="#">32.3.5/578</a> |
| 4003_7118              | Timer Control Register (PIT_TCTRL1)             | 32              | R/W    | 0000_0000h  | <a href="#">32.3.6/579</a> |
| 4003_711C              | Timer Flag Register (PIT_TFLG1)                 | 32              | R/W    | 0000_0000h  | <a href="#">32.3.7/580</a> |

### 32.3.1 PIT Module Control Register (PIT\_MCR)

This register enables or disables the PIT timer clocks and controls the timers when the PIT enters the Debug mode.

## Memory map/register description

Address: 4003\_7000h base + 0h offset = 4003\_7000h

| Bit   | 0  | 1  | 2  | 3  | 4  | 5  | 6  | 7  | 8  | 9  | 10 | 11 | 12 | 13       | 14   | 15  |
|-------|----|----|----|----|----|----|----|----|----|----|----|----|----|----------|------|-----|
| R     |    |    |    |    |    |    |    |    | 0  |    |    |    |    |          |      |     |
| W     |    |    |    |    |    |    |    |    |    |    |    |    |    |          |      |     |
| Reset | 0  | 0  | 0  | 0  | 0  | 0  | 0  | 0  | 0  | 0  | 0  | 0  | 0  | 0        | 0    | 0   |
| Bit   | 16 | 17 | 18 | 19 | 20 | 21 | 22 | 23 | 24 | 25 | 26 | 27 | 28 | 29       | 30   | 31  |
| R     |    |    |    |    |    |    |    |    | 0  |    |    |    |    | Reserved | MDIS | FRZ |
| W     |    |    |    |    |    |    |    |    |    |    |    |    |    |          |      |     |
| Reset | 0  | 0  | 0  | 0  | 0  | 0  | 0  | 0  | 0  | 0  | 0  | 0  | 0  | 0        | 1    | 0   |

## PIT\_MCR field descriptions

| Field            | Description  |
|------------------|--|
| 0–28<br>Reserved | This field is reserved.<br>This read-only field is reserved and always has the value 0.  |
| 29<br>Reserved   | This field is reserved.  |
| 30<br>MDIS       | Module Disable - (PIT section)<br><br>Disables the standard timers. The RTI timer is not affected by this field. This field must be enabled before any other setup is done.<br><br>0 Clock for standard PIT timers is enabled.<br>1 Clock for standard PIT timers is disabled. |
| 31<br>FRZ        | Freeze<br><br>Allows the timers to be stopped when the device enters the Debug mode.<br><br>0 Timers continue to run in Debug mode.<br>1 Timers are stopped in Debug mode.   |

### 32.3.2 PIT Upper Lifetime Timer Register (PIT\_LTMR64H)

This register is intended for applications that chain timer 0 and timer 1 to build a 64-bit lifetimer.

Address: 4003\_7000h base + E0h offset = 4003\_70E0h

| Bit   | 0 | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 | 10 | 11 | 12 | 13 | 14 | 15 | 16  | 17 | 18 | 19 | 20 | 21 | 22 | 23 | 24 | 25 | 26 | 27 | 28 | 29 | 30 | 31 |
|-------|---|---|---|---|---|---|---|---|---|---|----|----|----|----|----|----|-----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|
| R     |   |   |   |   |   |   |   |   |   |   |    |    |    |    |    |    | LTH |    |    |    |    |    |    |    |    |    |    |    |    |    |    |    |
| W     |   |   |   |   |   |   |   |   |   |   |    |    |    |    |    |    |     |    |    |    |    |    |    |    |    |    |    |    |    |    |    |    |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0  | 0  | 0  | 0  | 0  | 0  | 0   | 0  | 0  | 0  | 0  | 0  | 0  | 0  | 0  | 0  | 0  | 0  | 0  | 0  | 0  |    |

#### PIT\_LTMR64H field descriptions

| Field       | Description   |
|-------------|---|
| 0-31<br>LTH | Life Timer value<br><br>Shows the timer value of timer 1. If this register is read at a time t1, LTMR64L shows the value of timer 0 at time t1. |

### 32.3.3 PIT Lower Lifetime Timer Register (PIT\_LTMR64L)

This register is intended for applications that chain timer 0 and timer 1 to build a 64-bit lifetimer.

To use LTMR64H and LTMR64L, timer 0 and timer 1 need to be chained. To obtain the correct value, first read LTMR64H and then LTMR64L. LTMR64H will have the value of CVAL1 at the time of the first access, LTMR64L will have the value of CVAL0 at the time of the first access, therefore the application does not need to worry about carry-over effects of the running counter.

Address: 4003\_7000h base + E4h offset = 4003\_70E4h

| Bit   | 0 | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 | 10 | 11 | 12 | 13 | 14 | 15 | 16  | 17 | 18 | 19 | 20 | 21 | 22 | 23 | 24 | 25 | 26 | 27 | 28 | 29 | 30 | 31 |
|-------|---|---|---|---|---|---|---|---|---|---|----|----|----|----|----|----|-----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|
| R     |   |   |   |   |   |   |   |   |   |   |    |    |    |    |    |    | LTL |    |    |    |    |    |    |    |    |    |    |    |    |    |    |    |
| W     |   |   |   |   |   |   |   |   |   |   |    |    |    |    |    |    |     |    |    |    |    |    |    |    |    |    |    |    |    |    |    |    |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0  | 0  | 0  | 0  | 0  | 0  | 0   | 0  | 0  | 0  | 0  | 0  | 0  | 0  | 0  | 0  | 0  | 0  | 0  | 0  | 0  |    |

#### PIT\_LTMR64L field descriptions

| Field       | Description   |
|-------------|---|
| 0-31<br>LTL | Life Timer value<br><br>Shows the value of timer 0 at the time LTMR64H was last read. It will only update if LTMR64H is read. |

### 32.3.4 Timer Load Value Register (PIT\_LDVAL $n$ )

These registers select the timeout period for the timer interrupts.

Address: 4003\_7000h base + 100h offset + (16d × i), where i=0d to 1d

## PIT LDVAL $n$ field descriptions

| Field       | Description   |
|-------------|---|
| 0-31<br>TSV | <p>Timer Start Value</p> <p>Sets the timer start value. The timer will count down until it reaches 0, then it will generate an interrupt and load this register value again. Writing a new value to this register will not restart the timer; instead the value will be loaded after the timer expires. To abort the current cycle and start a timer period with the new value, the timer must be disabled and enabled again.</p> |

### 32.3.5 Current Timer Value Register (PIT\_CVALn)

These registers indicate the current timer position.

Address: 4003 7000h base + 104h offset + (16d x i), where i=0d to 1d

TVL

R

W

Reset 0

0 0

## PIT\_CVAL*n* field descriptions

| Field       | Description   |
|-------------|---|
| 0-31<br>TVL | <p>Current Timer Value</p> <p>Represents the current timer value, if the timer is enabled.</p> <p><b>NOTE:</b></p> <ul style="list-style-type: none"> <li>• If the timer is disabled, do not use this field as its value is unreliable.</li> <li>• The timer uses a downcounter. The timer values are frozen in Debug mode if MCR[FRZ] is set.</li> </ul> |

### 32.3.6 Timer Control Register (PIT\_TCTRL*n*)

These register contain the control bits for each timer.

Address: 4003\_7000h base + 108h offset + (16d × i), where i=0d to 1d

| Bit   | 0  | 1  | 2  | 3  | 4  | 5  | 6  | 7  | 8  | 9  | 10 | 11 | 12 | 13  | 14  | 15  |
|-------|----|----|----|----|----|----|----|----|----|----|----|----|----|-----|-----|-----|
| R     |    |    |    |    |    |    |    |    | 0  |    |    |    |    |     |     |     |
| W     |    |    |    |    |    |    |    |    |    |    |    |    |    |     |     |     |
| Reset | 0  | 0  | 0  | 0  | 0  | 0  | 0  | 0  | 0  | 0  | 0  | 0  | 0  | 0   | 0   | 0   |
| Bit   | 16 | 17 | 18 | 19 | 20 | 21 | 22 | 23 | 24 | 25 | 26 | 27 | 28 | 29  | 30  | 31  |
| R     |    |    |    |    |    |    |    |    | 0  |    |    |    |    | CHN | TIE | TEN |
| W     |    |    |    |    |    |    |    |    |    |    |    |    |    |     |     |     |
| Reset | 0  | 0  | 0  | 0  | 0  | 0  | 0  | 0  | 0  | 0  | 0  | 0  | 0  | 0   | 0   | 0   |

PIT\_TCTRL*n* field descriptions

| Field            | Description   |
|------------------|---|
| 0–28<br>Reserved | This field is reserved.<br>This read-only field is reserved and always has the value 0.   |
| 29<br>CHN        | Chain Mode<br><br>When activated, Timer n-1 needs to expire before timer n can decrement by 1.<br><br>Timer 0 can not be changed.<br><br>0 Timer is not chained.<br>1 Timer is chained to previous timer. For example, for Channel 2, if this field is set, Timer 2 is chained to Timer 1.  |
| 30<br>TIE        | Timer Interrupt Enable<br><br>When an interrupt is pending, or, TFLGn[TIF] is set, enabling the interrupt will immediately cause an interrupt event. To avoid this, the associated TFLGn[TIF] must be cleared first.<br><br>0 Interrupt requests from Timer n are disabled.<br>1 Interrupt will be requested whenever TIF is set. |
| 31<br>TEN        | Timer Enable<br><br>Enables or disables the timer.<br><br>0 Timer n is disabled.<br>1 Timer n is enabled.   |

### 32.3.7 Timer Flag Register (PIT\_TFLG $n$ )

These registers hold the PIT interrupt flags.

Address: 4003\_7000h base + 10Ch offset + (16d  $\times$  i), where i=0d to 1d

| Bit   | 0  | 1  | 2  | 3  | 4  | 5  | 6  | 7  | 8  | 9  | 10 | 11 | 12 | 13 | 14 | 15  |
|-------|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|-----|
| R     |    |    |    |    |    |    |    |    | 0  |    |    |    |    |    |    |     |
| W     |    |    |    |    |    |    |    |    |    |    |    |    |    |    |    |     |
| Reset | 0  | 0  | 0  | 0  | 0  | 0  | 0  | 0  | 0  | 0  | 0  | 0  | 0  | 0  | 0  | 0   |
| Bit   | 16 | 17 | 18 | 19 | 20 | 21 | 22 | 23 | 24 | 25 | 26 | 27 | 28 | 29 | 30 | 31  |
| R     |    |    |    |    |    |    |    |    | 0  |    |    |    |    |    |    | TIF |
| W     |    |    |    |    |    |    |    |    |    |    |    |    |    |    |    | w1c |
| Reset | 0  | 0  | 0  | 0  | 0  | 0  | 0  | 0  | 0  | 0  | 0  | 0  | 0  | 0  | 0  | 0   |

#### PIT\_TFLG $n$ field descriptions

| Field            | Description   |
|------------------|---|
| 0–30<br>Reserved | This field is reserved.<br>This read-only field is reserved and always has the value 0.   |
| 31<br>TIF        | Timer Interrupt Flag<br><br>Sets to 1 at the end of the timer period. Writing 1 to this flag clears it. Writing 0 has no effect. If enabled, or, when TCTRLn[TIE] = 1, TIF causes an interrupt request.<br><br>0 Timeout has not yet occurred.<br>1 Timeout has occurred. |

## 32.4 Functional description

This section provides the functional description of the module.

### 32.4.1 General operation

This section gives detailed information on the internal operation of the module. Each timer can be used to generate trigger pulses and interrupts. Each interrupt is available on a separate interrupt line.

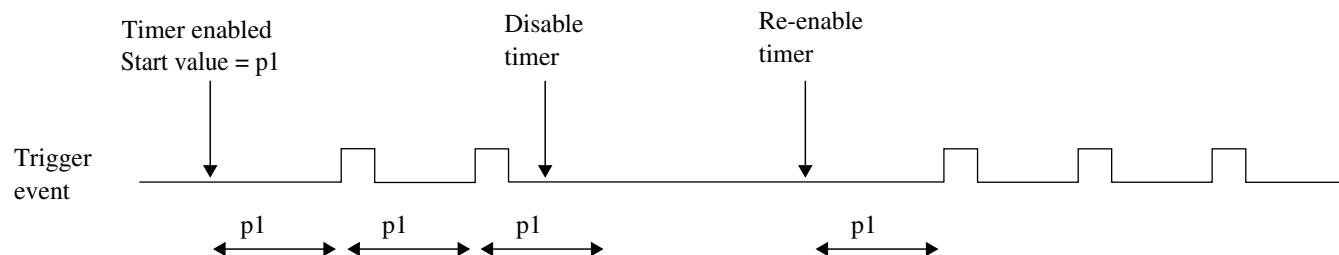
### 32.4.1.1 Timers

The timers generate triggers at periodic intervals, when enabled. The timers load the start values as specified in their LDVAL registers, count down to 0 and then load the respective start value again. Each time a timer reaches 0, it will generate a trigger pulse and set the interrupt flag.

All interrupts can be enabled or masked by setting TCTRLn[TIE]. A new interrupt can be generated only after the previous one is cleared.

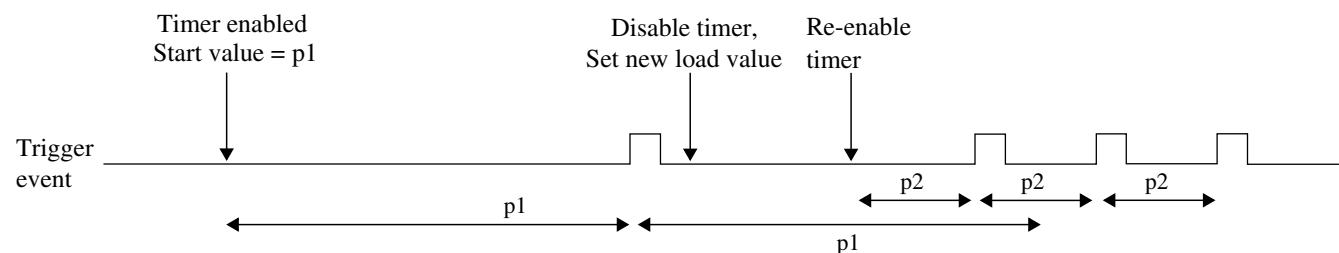
If desired, the current counter value of the timer can be read via the CVAL registers.

The counter period can be restarted, by first disabling, and then enabling the timer with TCTRLn[TEN]. See the following figure.



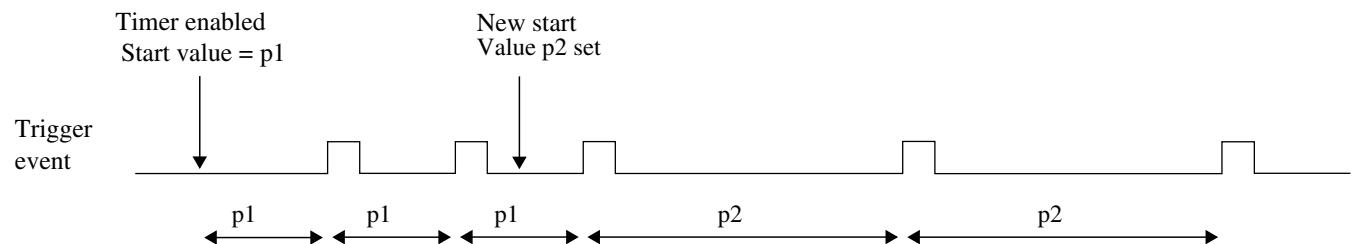
**Figure 32-17. Stopping and starting a timer**

The counter period of a running timer can be modified, by first disabling the timer, setting a new load value, and then enabling the timer again. See the following figure.



**Figure 32-18. Modifying running timer period**

It is also possible to change the counter period without restarting the timer by writing LDVAL with the new load value. This value will then be loaded after the next trigger event. See the following figure.



**Figure 32-19. Dynamically setting a new load value**

### 32.4.1.2 Debug mode

In Debug mode, the timers will be frozen based on MCR[FRZ]. This is intended to aid software development, allowing the developer to halt the processor, investigate the current state of the system, for example, the timer values, and then continue the operation.

### 32.4.2 Interrupts

All the timers support interrupt generation. See the MCU specification for related vector addresses and priorities.

Timer interrupts can be enabled by setting TCTRLn[TIE]. TFLGn[TIF] are set to 1 when a timeout occurs on the associated timer, and are cleared to 0 by writing a 1 to the corresponding TFLGn[TIF].

### 32.4.3 Chained timers

When a timer has chain mode enabled, it will only count after the previous timer has expired. So if timer n-1 has counted down to 0, counter n will decrement the value by one. This allows to chain some of the timers together to form a longer timer. The first timer (timer 0) cannot be chained to any other timer.

## 32.5 Initialization and application information

In the example configuration:

- The PIT clock has a frequency of 50 MHz.

- Timer 1 creates an interrupt every 5.12 ms.
- Timer 3 creates a trigger event every 30 ms.

The PIT module must be activated by writing a 0 to MCR[MDIS].

The 50 MHz clock frequency equates to a clock period of 20 ns. Timer 1 needs to trigger every  $5.12\text{ ms} / 20\text{ ns} = 256,000$  cycles and Timer 3 every  $30\text{ ms} / 20\text{ ns} = 1,500,000$  cycles. The value for the LDVAL register trigger is calculated as:

$$\text{LDVAL trigger} = (\text{period} / \text{clock period}) - 1$$

This means LDVAL1 and LDVAL3 must be written with 0x0003E7FF and 0x0016E35F respectively.

The interrupt for Timer 1 is enabled by setting TCTRL1[TIE]. The timer is started by writing 1 to TCTRL1[TEN].

Timer 3 shall be used only for triggering. Therefore, Timer 3 is started by writing a 1 to TCTRL3[TEN]. TCTRL3[TIE] stays at 0.

The following example code matches the described setup:

```
// turn on PIT
PIT_MCR = 0x00;

// Timer 1
PIT_LDVAL1 = 0x0003E7FF; // setup timer 1 for 256000 cycles
PIT_TCTRL1 = TIE; // enable Timer 1 interrupts
PIT_TCTRL1 |= TEN; // start Timer 1

// Timer 3
PIT_LDVAL3 = 0x0016E35F; // setup timer 3 for 1500000 cycles
PIT_TCTRL3 |= TEN; // start Timer 3
```

## 32.6 Example configuration for chained timers

In the example configuration:

- The PIT clock has a frequency of 100 MHz.
- Timers 1 and 2 are available.
- An interrupt shall be raised every 1 hour.

The PIT module needs to be activated by writing a 0 to MCR[MDIS].

### Example configuration for the lifetime timer

The 100 MHz clock frequency equates to a clock period of 10 ns, so the PIT needs to count for 6000 million cycles, which is more than a single timer can do. So, Timer 1 is set up to trigger every 6 s (600 million cycles). Timer 2 is chained to Timer 1 and programmed to trigger 10 times.

The value for the LDVAL register trigger is calculated as number of cycles-1, so LDVAL1 receives the value 0x23C345FF and LDVAL2 receives the value 0x00000009.

The interrupt for Timer 2 is enabled by setting TCTRL2[TIE], the Chain mode is activated by setting TCTRL2[CHN], and the timer is started by writing a 1 to TCTRL2[TEN]. TCTRL1[TEN] needs to be set, and TCTRL1[CHN] and TCTRL1[TIE] are cleared.

The following example code matches the described setup:

```
// turn on PIT
PIT_MCR = 0x00;

// Timer 2
PIT_LDVAL2 = 0x00000009; // setup Timer 2 for 10 counts
PIT_TCTRL2 = TIE; // enable Timer 2 interrupt
PIT_TCTRL2 |= CHN; // chain Timer 2 to Timer 1
PIT_TCTRL2 |= TEN; // start Timer 2

// Timer 1
PIT_LDVAL1 = 0x23C345FF; // setup Timer 1 for 600 000 000 cycles
PIT_TCTRL1 = TEN; // start Timer 1
```

## 32.7 Example configuration for the lifetime timer

To configure the lifetimer timer, channels 0 and 1 need to be chained together.

First the PIT module needs to be activated by writing a 0 to the MDIS bit in the CTRL register, then the LDVAL registers need to be set to the maximum value.

The timer is a downcounter.

The following example code matches the described setup:

```
// turn on PIT
PIT_MCR = 0x00;

// Timer 1
PIT_LDVAL1 = 0xFFFFFFFF; // setup timer 1 for maximum counting period
PIT_TCTRL1 = 0x0; // disable timer 1 interrupts
PIT_TCTRL1 |= CHN; // chain timer 1 to timer 0
PIT_TCTRL1 |= TEN; // start timer 1

// Timer 0
```

```
PIT_LDVAL0 = 0xFFFFFFFF; // setup timer 0 for maximum counting period  
PIT_TCTRL0 = TEN; // start timer 0
```

To access the lifetime, read first LTMR64H and then LTMR64L.

```
current_uptime = PIT_LTMR64H<<32;  
current_uptime = current_uptime + PIT_LTMR64L;
```



# Chapter 33

## Low-Power Timer (LPTMR)

### 33.1 Introduction

#### NOTE

For the chip-specific implementation details of this module's instances see the chip configuration information.

The low-power timer (LPTMR) can be configured to operate as a time counter with optional prescaler, or as a pulse counter with optional glitch filter, across all power modes, including the low-leakage modes. It can also continue operating through most system reset events, allowing it to be used as a time of day counter.

#### 33.1.1 Features

The features of the LPTMR module include:

- 16-bit time counter or pulse counter with compare
  - Optional interrupt can generate asynchronous wakeup from any low-power mode
  - Hardware trigger output
  - Counter supports free-running mode or reset on compare
- Configurable clock source for prescaler/glitch filter
- Configurable input source for pulse counter
  - Rising-edge or falling-edge

#### 33.1.2 Modes of operation

The following table describes the operation of the LPTMR module in various modes.

**Table 33-1. Modes of operation**

| Modes       | Description  |
|-------------|--|
| Run         | The LPTMR operates normally.   |
| Wait        | The LPTMR continues to operate normally and may be configured to exit the low-power mode by generating an interrupt request. |
| Stop        | The LPTMR continues to operate normally and may be configured to exit the low-power mode by generating an interrupt request. |
| Low-Leakage | The LPTMR continues to operate normally and may be configured to exit the low-power mode by generating an interrupt request. |
| Debug       | The LPTMR operates normally in Pulse Counter mode, but counter does not increment in Time Counter mode.                      |

## 33.2 LPTMR signal descriptions

**Table 33-2. LPTMR signal descriptions**

| Signal     | I/O | Description             |
|------------|-----|-------------------------|
| LPTMR_ALTn | I   | Pulse Counter Input pin |

### 33.2.1 Detailed signal descriptions

**Table 33-3. LPTMR interface—detailed signal descriptions**

| Signal     | I/O | Description  |   |
|------------|-----|--|---|
| LPTMR_ALTn | I   | Pulse Counter Input  |   |
|            |     | The LPTMR can select one of the input pins to be used in Pulse Counter mode. |   |
|            |     | State meaning  | Assertion—if configured for pulse counter mode with active-high input, then assertion causes the CNR to increment.<br>Deassertion—if configured for pulse counter mode with active-low input, then deassertion causes the CNR to increment. |
|            |     | Timing   | Assertion or deassertion may occur at any time; input may assert asynchronously to the bus clock.   |

## 33.3 Memory map and register definition

### LPTMR memory map

| Absolute address (hex) | Register name  | Width (in bits) | Access | Reset value | Section/ page              |
|------------------------|--|-----------------|--------|-------------|----------------------------|
| 4004_0000              | Low Power Timer Control Status Register (LPTMR0_CSR) | 32              | R/W    | 0000_0000h  | <a href="#">33.3.1/589</a> |
| 4004_0004              | Low Power Timer Prescale Register (LPTMR0_PSR)       | 32              | R/W    | 0000_0000h  | <a href="#">33.3.2/590</a> |
| 4004_0008              | Low Power Timer Compare Register (LPTMR0_CMR)        | 32              | R/W    | 0000_0000h  | <a href="#">33.3.3/592</a> |
| 4004_000C              | Low Power Timer Counter Register (LPTMR0_CNR)        | 32              | R      | 0000_0000h  | <a href="#">33.3.4/592</a> |

**LPTMRx\_CSR field descriptions (continued)**

| Field | Description  |
|-------|--|
|       | 01 Pulse counter input 1 is selected.<br>10 Pulse counter input 2 is selected.<br>11 Pulse counter input 3 is selected.  |
| 3 TPP | <b>Timer Pin Polarity</b><br>Configures the polarity of the input source in Pulse Counter mode. TPP must be changed only when the LPTMR is disabled.<br>0 Pulse Counter input source is active-high, and the CNR will increment on the rising-edge.<br>1 Pulse Counter input source is active-low, and the CNR will increment on the falling-edge. |
| 2 TFC | <b>Timer Free-Running Counter</b><br>When clear, TFC configures the CNR to reset whenever TCF is set. When set, TFC configures the CNR to reset on overflow. TFC must be altered only when the LPTMR is disabled.<br>0 CNR is reset whenever TCF is set.<br>1 CNR is reset on overflow.  |
| 1 TMS | <b>Timer Mode Select</b><br>Configures the mode of the LPTMR. TMS must be altered only when the LPTMR is disabled.<br>0 Time Counter mode.<br>1 Pulse Counter mode.  |
| 0 TEN | <b>Timer Enable</b><br>When TEN is clear, it resets the LPTMR internal logic, including the CNR and TCF. When TEN is set, the LPTMR is enabled. While writing 1 to this field, CSR[5:1] must not be altered.<br>0 LPTMR is disabled and internal logic is reset.<br>1 LPTMR is enabled.  |

**33.3.2 Low Power Timer Prescale Register (LPTMRx\_PSR)**

Address: 4004\_0000h base + 4h offset = 4004\_0004h

| Bit   | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 | 23 | 22 | 21 | 20 | 19       | 18   | 17  | 16 |
|-------|----|----|----|----|----|----|----|----|----|----|----|----|----------|------|-----|----|
| R     |    |    |    |    |    |    |    |    | 0  |    |    |    |          |      |     |    |
| W     |    |    |    |    |    |    |    |    |    |    |    |    |          |      |     |    |
| Reset | 0  | 0  | 0  | 0  | 0  | 0  | 0  | 0  | 0  | 0  | 0  | 0  | 0        | 0    | 0   | 0  |
| Bit   | 15 | 14 | 13 | 12 | 11 | 10 | 9  | 8  | 7  | 6  | 5  | 4  | 3        | 2    | 1   | 0  |
| R     |    |    |    |    |    |    |    |    | 0  |    |    |    | PRESCALE | PBYP | PCS |    |
| W     |    |    |    |    |    |    |    |    |    |    |    |    |          |      |     |    |
| Reset | 0  | 0  | 0  | 0  | 0  | 0  | 0  | 0  | 0  | 0  | 0  | 0  | 0        | 0    | 0   | 0  |

**LPTMRx\_PSR field descriptions**

| Field            | Description   |
|------------------|---|
| 31–7<br>Reserved | This field is reserved.<br>This read-only field is reserved and always has the value 0.   |
| 6–3<br>PRESCALE  | <p>Prescale Value</p> <p>Configures the size of the Prescaler in Time Counter mode or width of the glitch filter in Pulse Counter mode. PRESCALE must be altered only when the LPTMR is disabled.</p> <ul style="list-style-type: none"> <li>0000 Prescaler divides the prescaler clock by 2; glitch filter does not support this configuration.</li> <li>0001 Prescaler divides the prescaler clock by 4; glitch filter recognizes change on input pin after 2 rising clock edges.</li> <li>0010 Prescaler divides the prescaler clock by 8; glitch filter recognizes change on input pin after 4 rising clock edges.</li> <li>0011 Prescaler divides the prescaler clock by 16; glitch filter recognizes change on input pin after 8 rising clock edges.</li> <li>0100 Prescaler divides the prescaler clock by 32; glitch filter recognizes change on input pin after 16 rising clock edges.</li> <li>0101 Prescaler divides the prescaler clock by 64; glitch filter recognizes change on input pin after 32 rising clock edges.</li> <li>0110 Prescaler divides the prescaler clock by 128; glitch filter recognizes change on input pin after 64 rising clock edges.</li> <li>0111 Prescaler divides the prescaler clock by 256; glitch filter recognizes change on input pin after 128 rising clock edges.</li> <li>1000 Prescaler divides the prescaler clock by 512; glitch filter recognizes change on input pin after 256 rising clock edges.</li> <li>1001 Prescaler divides the prescaler clock by 1024; glitch filter recognizes change on input pin after 512 rising clock edges.</li> <li>1010 Prescaler divides the prescaler clock by 2048; glitch filter recognizes change on input pin after 1024 rising clock edges.</li> <li>1011 Prescaler divides the prescaler clock by 4096; glitch filter recognizes change on input pin after 2048 rising clock edges.</li> <li>1100 Prescaler divides the prescaler clock by 8192; glitch filter recognizes change on input pin after 4096 rising clock edges.</li> <li>1101 Prescaler divides the prescaler clock by 16,384; glitch filter recognizes change on input pin after 8192 rising clock edges.</li> <li>1110 Prescaler divides the prescaler clock by 32,768; glitch filter recognizes change on input pin after 16,384 rising clock edges.</li> <li>1111 Prescaler divides the prescaler clock by 65,536; glitch filter recognizes change on input pin after 32,768 rising clock edges.</li> </ul> |
| 2<br>PBYP        | <p>Prescaler Bypass</p> <p>When PBYP is set, the selected prescaler clock in Time Counter mode or selected input source in Pulse Counter mode directly clocks the CNR. When PBYP is clear, the CNR is clocked by the output of the prescaler/glitch filter. PBYP must be altered only when the LPTMR is disabled.</p> <ul style="list-style-type: none"> <li>0 Prescaler/glitch filter is enabled.</li> <li>1 Prescaler/glitch filter is bypassed.</li> </ul>   |
| 1–0<br>PCS       | <p>Prescaler Clock Select</p> <p>Selects the clock to be used by the LPTMR prescaler/glitch filter. PCS must be altered only when the LPTMR is disabled. The clock connections vary by device.</p> <p><b>NOTE:</b> See the chip configuration details for information on the connections to these inputs.</p>   |

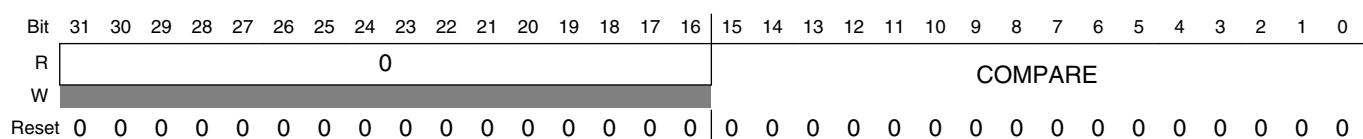
*Table continues on the next page...*

## LPTMRx\_PSR field descriptions (continued)

| Field | Description                               |
|-------|---|
| 00    | Prescaler/glitch filter clock 0 selected. |
| 01    | Prescaler/glitch filter clock 1 selected. |
| 10    | Prescaler/glitch filter clock 2 selected. |
| 11    | Prescaler/glitch filter clock 3 selected. |

### 33.3.3 Low Power Timer Compare Register (LPTMRx\_CMR)

Address: 4004 0000h base + 8h offset = 4004 0008h



## 33.4 Functional description

### 33.4.1 LPTMR power and reset

The LPTMR remains powered in all power modes, including low-leakage modes. If the LPTMR is not required to remain operating during a low-power mode, then it must be disabled before entering the mode.

The LPTMR is reset only on global Power On Reset (POR) or Low Voltage Detect (LVD). When configuring the LPTMR registers, the CSR must be initially written with the timer disabled, before configuring the PSR and CMR. Then, CSR[TIE] must be set as the last step in the initialization. This ensures the LPTMR is configured correctly and the LPTMR counter is reset to zero following a warm reset.

### 33.4.2 LPTMR clocking

The LPTMR prescaler/glitch filter can be clocked by one of the four clocks. The clock source must be enabled before the LPTMR is enabled.

#### NOTE

The clock source selected may need to be configured to remain enabled in low-power modes, otherwise the LPTMR will not operate during low-power modes.

In Pulse Counter mode with the prescaler/glitch filter bypassed, the selected input source directly clocks the CNR and no other clock source is required. To minimize power in this case, configure the prescaler clock source for a clock that is not toggling.

#### NOTE

The clock source or pulse input source selected for the LPTMR should not exceed the frequency  $f_{LPTMR}$  defined in the device datasheet.

### 33.4.3 LPTMR prescaler/glitch filter

The LPTMR prescaler and glitch filter share the same logic which operates as a prescaler in Time Counter mode and as a glitch filter in Pulse Counter mode.

**NOTE**

The prescaler/glitch filter configuration must not be altered when the LPTMR is enabled.

### 33.4.3.1 Prescaler enabled

In Time Counter mode, when the prescaler is enabled, the output of the prescaler directly clocks the CNR. When the LPTMR is enabled, the CNR will increment every  $2^2$  to  $2^{16}$  prescaler clock cycles. After the LPTMR is enabled, the first increment of the CNR will take an additional one or two prescaler clock cycles due to synchronization logic.

### 33.4.3.2 Prescaler bypassed

In Time Counter mode, when the prescaler is bypassed, the selected prescaler clock increments the CNR on every clock cycle. When the LPTMR is enabled, the first increment will take an additional one or two prescaler clock cycles due to synchronization logic.

### 33.4.3.3 Glitch filter

In Pulse Counter mode, when the glitch filter is enabled, the output of the glitch filter directly clocks the CNR. When the LPTMR is first enabled, the output of the glitch filter is asserted, that is, logic 1 for active-high and logic 0 for active-low. The following table shows the change in glitch filter output with the selected input source.

| If   | Then   |
|--|--|
| The selected input source remains deasserted for at least $2^1$ to $2^{15}$ consecutive prescaler clock rising edges | The glitch filter output will also deassert. |
| The selected input source remains asserted for at least $2^1$ to $2^{15}$ consecutive prescaler clock rising-edges   | The glitch filter output will also assert.   |

**NOTE**

The input is only sampled on the rising clock edge.

The CNR will increment each time the glitch filter output asserts. In Pulse Counter mode, the maximum rate at which the CNR can increment is once every  $2^2$  to  $2^{16}$  prescaler clock edges. When first enabled, the glitch filter will wait an additional one or two prescaler clock edges due to synchronization logic.

### 33.4.3.4 Glitch filter bypassed

In Pulse Counter mode, when the glitch filter is bypassed, the selected input source increments the CNR every time it asserts. Before the LPTMR is first enabled, the selected input source is forced to be asserted. This prevents the CNR from incrementing if the selected input source is already asserted when the LPTMR is first enabled.

### 33.4.4 LPTMR compare

When the CNR equals the value of the CMR and increments, the following events occur:

- CSR[TCF] is set.
- LPTMR interrupt is generated if CSR[TIE] is also set.
- LPTMR hardware trigger is generated.
- CNR is reset if CSR[TFC] is clear.

When the LPTMR is enabled, the CMR can be altered only when CSR[TCF] is set. When updating the CMR, the CMR must be written and CSR[TCF] must be cleared before the LPTMR counter has incremented past the new LPTMR compare value.

### 33.4.5 LPTMR counter

The CNR increments by one on every:

- Prescaler clock in Time Counter mode with prescaler bypassed
- Prescaler output in Time Counter mode with prescaler enabled
- Input source assertion in Pulse Counter mode with glitch filter bypassed
- Glitch filter output in Pulse Counter mode with glitch filter enabled

The CNR is reset when the LPTMR is disabled or if the counter register overflows. If CSR[TFC] is cleared, then the CNR is also reset whenever CSR[TCF] is set.

The CNR continues incrementing when the core is halted in Debug mode when configured for Pulse Counter mode, the CNR will stop incrementing when the core is halted in Debug mode when configured for Time Counter mode.

The CNR cannot be initialized, but can be read at any time. On each read of the CNR, software must first write to the CNR with any value. This will synchronize and register the current value of the CNR into a temporary register. The contents of the temporary register are returned on each read of the CNR.

When reading the CNR, the bus clock must be at least two times faster than the rate at which the LPTMR counter is incrementing, otherwise incorrect data may be returned.

### 33.4.6 LPTMR hardware trigger

The LPTMR hardware trigger asserts at the same time the CSR[TCF] is set and can be used to trigger hardware events in other peripherals without software intervention. The hardware trigger is always enabled.

| When  | Then   |
|---|--|
| The CMR is set to 0 with CSR[TFC] clear                   | The LPTMR hardware trigger will assert on the first compare and does not deassert.                         |
| The CMR is set to a nonzero value, or, if CSR[TFC] is set | The LPTMR hardware trigger will assert on each compare and deassert on the following increment of the CNR. |

### 33.4.7 LPTMR interrupt

The LPTMR interrupt is generated whenever CSR[TIE] and CSR[TCF] are set. CSR[TCF] is cleared by disabling the LPTMR or by writing a logic 1 to it.

CSR[TIE] can be altered and CSR[TCF] can be cleared while the LPTMR is enabled.

The LPTMR interrupt is generated asynchronously to the system clock and can be used to generate a wakeup from any low-power mode, including the low-leakage modes, provided the LPTMR is enabled as a wakeup source.

# **Chapter 34**

## **Real Time Clock (RTC)**

### **34.1 Introduction**

#### **NOTE**

For the chip-specific implementation details of this module's instances see the chip configuration information.

#### **34.1.1 Features**

The RTC module features include:

- 32-bit seconds counter with roll-over protection and 32-bit alarm
- 16-bit prescaler with compensation that can correct errors between 0.12 ppm and 3906 ppm
- Register write protection
  - Lock register requires POR or software reset to enable write access
- 1 Hz square wave output

#### **34.1.2 Modes of operation**

The RTC remains functional in all low power modes and can generate an interrupt to exit any low power mode.

### 34.1.3 RTC Signal Descriptions

Table 34-1. RTC signal descriptions

| Signal     | Description             | I/O |
|------------|-------------------------|-----|
| RTC_CLKOUT | 1 Hz square-wave output | O   |

#### 34.1.3.1 RTC clock output

The clock to the seconds counter is available on the RTC\_CLKOUT signal. It is a 1 Hz square wave output.

## 34.2 Register definition

All registers must be accessed using 32-bit writes and all register accesses incur three wait states.

Write accesses to any register by non-supervisor mode software, when the supervisor access bit in the control register is clear, will terminate with a bus error.

Read accesses by non-supervisor mode software complete as normal.

Writing to a register protected by the lock register does not generate a bus error, but the write will not complete.

RTC memory map

| Absolute address (hex) | Register name                            | Width (in bits) | Access | Reset value | Section/ page              |
|------------------------|--|-----------------|--------|-------------|----------------------------|
| 4003_D000              | RTC Time Seconds Register (RTC_TSR)      | 32              | R/W    | 0000_0000h  | <a href="#">34.2.1/599</a> |
| 4003_D004              | RTC Time Prescaler Register (RTC_TPR)    | 32              | R/W    | 0000_0000h  | <a href="#">34.2.2/599</a> |
| 4003_D008              | RTC Time Alarm Register (RTC_TAR)        | 32              | R/W    | 0000_0000h  | <a href="#">34.2.3/600</a> |
| 4003_D00C              | RTC Time Compensation Register (RTC_TCR) | 32              | R/W    | 0000_0000h  | <a href="#">34.2.4/600</a> |
| 4003_D010              | RTC Control Register (RTC_CR)            | 32              | R/W    | 0000_0000h  | <a href="#">34.2.5/601</a> |
| 4003_D014              | RTC Status Register (RTC_SR)             | 32              | R/W    | 0000_0001h  | <a href="#">34.2.6/603</a> |
| 4003_D018              | RTC Lock Register (RTC_LR)               | 32              | R/W    | 0000_00FFh  | <a href="#">34.2.7/604</a> |
| 4003_D01C              | RTC Interrupt Enable Register (RTC_IER)  | 32              | R/W    | 0000_0007h  | <a href="#">34.2.8/605</a> |

### 34.2.1 RTC Time Seconds Register (RTC\_TSR)

Address: 4003 D000h base + 0h offset = 4003 D000h

## **RTC\_TSR field descriptions**

| Field       | Description  |
|-------------|--|
| 31–0<br>TSR | <p>Time Seconds Register</p> <p>When the time counter is enabled, the TSR is read only and increments once a second provided SR[TOF] or SR[TIF] are not set. The time counter will read as zero when SR[TOF] or SR[TIF] are set. When the time counter is disabled, the TSR can be read or written. Writing to the TSR when the time counter is disabled will clear the SR[TOF] and/or the SR[TIF]. Writing to TSR with zero is supported, but not recommended because TSR will read as zero when SR[TIF] or SR[TOF] are set (indicating the time is invalid).</p> |

### 34.2.2 RTC Time Prescaler Register (RTC\_TPR)

Address: 4003 D000h base + 4h offset = 4003 D004h

## RTC TPR field descriptions

| Field             | Description   |
|-------------------|---|
| 31–16<br>Reserved | <p>This field is reserved.</p> <p>This read-only field is reserved and always has the value 0.</p>  |
| 15–0<br>TPR       | <p>Time Prescaler Register</p> <p>When the time counter is enabled, the TPR is read only and increments every 32.768 kHz clock cycle. The time counter will read as zero when SR[TOF] or SR[TIF] are set. When the time counter is disabled, the TPR can be read or written. The TSR[TSR] increments when bit 14 of the TPR transitions from a logic one to a logic zero.</p> |

### 34.2.3 RTC Time Alarm Register (RTC\_TAR)

Address: 4003 D000h base + 8h offset = 4003 D008h

## RTC\_TAR field descriptions

| Field       | Description  |
|-------------|--|
| 31–0<br>TAR | <p>Time Alarm Register</p> <p>When the time counter is enabled, the SR[TAF] is set whenever the TAR[TAR] equals the TSR[TSR] and the TSR[TSR] increments. Writing to the TAR clears the SR[TAF].</p> |

#### 34.2.4 RTC Time Compensation Register (RTC\_TCR)

Address: 4003 D000h base + Ch offset = 4003 D00Ch

| Bit | 31  | 30 | 29 | 28 | 27 | 26 | 25 | 24 | 23  | 22 | 21 | 20 | 19 | 18 | 17 | 16 | 15  | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7   | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|-----|-----|----|----|----|----|----|----|----|-----|----|----|----|----|----|----|----|-----|----|----|----|----|----|---|---|-----|---|---|---|---|---|---|---|
| R   | CIC |    |    |    |    |    |    |    | TCV |    |    |    |    |    |    |    | CIR |    |    |    |    |    |   |   | TCR |   |   |   |   |   |   |   |
| W   |     |    |    |    |    |    |    |    |     |    |    |    |    |    |    |    |     |    |    |    |    |    |   |   |     |   |   |   |   |   |   |   |

## RTC\_TCR field descriptions

| Field        | Description  |
|--------------|--|
| 31–24<br>CIC | <p>Compensation Interval Counter</p> <p>Current value of the compensation interval counter. If the compensation interval counter equals zero then it is loaded with the contents of the CIR. If the CIC does not equal zero then it is decremented once a second.</p>  |
| 23–16<br>TCV | <p>Time Compensation Value</p> <p>Current value used by the compensation logic for the present second interval. Updated once a second if the CIC equals 0 with the contents of the TCR field. If the CIC does not equal zero then it is loaded with zero (compensation is not enabled for that second increment).</p>  |
| 15–8<br>CIR  | <p>Compensation Interval Register</p> <p>Configures the compensation interval in seconds from 1 to 256 to control how frequently the TCR should adjust the number of 32.768 kHz cycles in each second. The value written should be one less than the number of seconds. For example, write zero to configure for a compensation interval of one second. This register is double buffered and writes do not take affect until the end of the current compensation interval.</p> |
| 7–0<br>TCR   | <p>Time Compensation Register</p> <p>Configures the number of 32.768 kHz clock cycles in each second. This register is double buffered and writes do not take affect until the end of the current compensation interval.</p>   |

*Table continues on the next page...*

**RTC\_TCR field descriptions (continued)**

| Field | Description |   |
|-------|-------------|---|
|       | 80h         | Time Prescaler Register overflows every 32896 clock cycles. |
|       | ...         | ...   |
|       | FFh         | Time Prescaler Register overflows every 32769 clock cycles. |
|       | 00h         | Time Prescaler Register overflows every 32768 clock cycles. |
|       | 01h         | Time Prescaler Register overflows every 32767 clock cycles. |
|       | ...         | ...   |
|       | 7Fh         | Time Prescaler Register overflows every 32641 clock cycles. |

**34.2.5 RTC Control Register (RTC\_CR)**

Address: 4003\_D000h base + 10h offset = 4003\_D010h

| Bit   | 31 | 30       | 29   | 28   | 27   | 26    | 25   | 24   | 23 | 22 | 21 | 20 | 19 | 18  | 17  | 16  |
|-------|----|----------|------|------|------|-------|------|------|----|----|----|----|----|-----|-----|-----|
| R     |    |          |      |      |      |       |      |      | 0  |    |    |    |    |     |     |     |
| W     |    |          |      |      |      |       |      |      |    |    |    |    |    |     |     |     |
| Reset | 0  | 0        | 0    | 0    | 0    | 0     | 0    | 0    | 0  | 0  | 0  | 0  | 0  | 0   | 0   | 0   |
| Bit   | 15 | 14       | 13   | 12   | 11   | 10    | 9    | 8    | 7  | 6  | 5  | 4  | 3  | 2   | 1   | 0   |
| R     | 0  | Reserved | SC2P | SC4P | SC8P | SC16P | CLKO | OSCE |    |    | 0  |    | UM | SUP | WPE | SWR |
| W     | 0  | 0        | 0    | 0    | 0    | 0     | 0    | 0    | 0  | 0  | 0  | 0  | 0  | 0   | 0   | 0   |
| Reset | 0  | 0        | 0    | 0    | 0    | 0     | 0    | 0    | 0  | 0  | 0  | 0  | 0  | 0   | 0   | 0   |

**RTC\_CR field descriptions**

| Field             | Description   |
|-------------------|---|
| 31–15<br>Reserved | This field is reserved.<br>This read-only field is reserved and always has the value 0. |

*Table continues on the next page...*

## RTC\_CR field descriptions (continued)

| Field           | Description  |
|-----------------|--|
| 14<br>Reserved  | This field is reserved.<br>It must always be written to 0.   |
| 13<br>SC2P      | Oscillator 2pF Load Configure<br><br>0 Disable the load.<br>1 Enable the additional load.  |
| 12<br>SC4P      | Oscillator 4pF Load Configure<br><br>0 Disable the load.<br>1 Enable the additional load.  |
| 11<br>SC8P      | Oscillator 8pF Load Configure<br><br>0 Disable the load.<br>1 Enable the additional load.  |
| 10<br>SC16P     | Oscillator 16pF Load Configure<br><br>0 Disable the load.<br>1 Enable the additional load.   |
| 9<br>CLKO       | Clock Output<br><br>0 The 32 kHz clock is output to other peripherals.<br>1 The 32 kHz clock is not output to other peripherals.   |
| 8<br>OSCE       | Oscillator Enable<br><br>0 32.768 kHz oscillator is disabled.<br>1 32.768 kHz oscillator is enabled. After setting this bit, wait the oscillator startup time before enabling the time counter to allow the 32.768 kHz clock time to stabilize.  |
| 7–4<br>Reserved | This field is reserved.<br>This read-only field is reserved and always has the value 0.  |
| 3<br>UM         | Update Mode<br><br>Allows SR[TCE] to be written even when the Status Register is locked. When set, the SR[TCE] can always be written if the SR[TIF] or SR[TOF] are set or if the SR[TCE] is clear.<br><br>0 Registers cannot be written when locked.<br>1 Registers can be written when locked under limited conditions. |
| 2<br>SUP        | Supervisor Access<br><br>0 Non-supervisor mode write accesses are not supported and generate a bus error.<br>1 Non-supervisor mode write accesses are supported.   |
| 1<br>WPE        | Wakeup Pin Enable<br><br>The wakeup pin is optional and not available on all devices.<br><br>0 Wakeup pin is disabled.<br>1 Wakeup pin is enabled and wakeup pin asserts if the RTC interrupt asserts or the wakeup pin is turned on.  |
| 0<br>SWR        | Software Reset   |

*Table continues on the next page...*

**RTC\_CR field descriptions (continued)**

| Field | Description   |
|-------|---|
|       | 0 No effect.<br>1 Resets all RTC registers except for the SWR bit . The SWR bit is cleared by POR and by software explicitly clearing it. |

**34.2.6 RTC Status Register (RTC\_SR)**

Address: 4003\_D000h base + 14h offset = 4003\_D014h

| Bit   | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 | 23 | 22 | 21  | 20 | 19  | 18  | 17  | 16 |
|-------|----|----|----|----|----|----|----|----|----|----|-----|----|-----|-----|-----|----|
| R     | 0  |    |    |    |    |    |    |    |    |    |     |    |     |     |     |    |
| W     |    |    |    |    |    |    |    |    |    |    |     |    |     |     |     |    |
| Reset | 0  | 0  | 0  | 0  | 0  | 0  | 0  | 0  | 0  | 0  | 0   | 0  | 0   | 0   | 0   | 0  |
| Bit   | 15 | 14 | 13 | 12 | 11 | 10 | 9  | 8  | 7  | 6  | 5   | 4  | 3   | 2   | 1   | 0  |
| R     | 0  |    |    |    |    |    |    |    |    |    | TCE | 0  | TAF | TOF | TIF |    |
| W     |    |    |    |    |    |    |    |    |    |    | 0   | 0  | 0   | 0   | 0   | 1  |
| Reset | 0  | 0  | 0  | 0  | 0  | 0  | 0  | 0  | 0  | 0  | 0   | 0  | 0   | 0   | 0   | 1  |

**RTC\_SR field descriptions**

| Field            | Description   |
|------------------|---|
| 31–5<br>Reserved | This field is reserved.<br>This read-only field is reserved and always has the value 0.   |
| 4<br>TCE         | Time Counter Enable<br><br>When time counter is disabled the TSR register and TPR register are writeable, but do not increment.<br>When time counter is enabled the TSR register and TPR register are not writeable, but increment.<br><br>0 Time counter is disabled.<br>1 Time counter is enabled.  |
| 3<br>Reserved    | This field is reserved.<br>This read-only field is reserved and always has the value 0.   |
| 2<br>TAF         | Time Alarm Flag<br><br>Time alarm flag is set when the TAR[TAR] equals the TSR[TSR] and the TSR[TSR] increments. This bit is cleared by writing the TAR register.<br><br>0 Time alarm has not occurred.<br>1 Time alarm has occurred.   |
| 1<br>TOF         | Time Overflow Flag<br><br>Time overflow flag is set when the time counter is enabled and overflows. The TSR and TPR do not increment and read as zero when this bit is set. This bit is cleared by writing the TSR register when the time counter is disabled.<br><br>0 Time overflow has not occurred.<br>1 Time overflow has occurred and time counter is read as zero. |

Table continues on the next page...

**RTC\_SR field descriptions (continued)**

| Field    | Description   |
|----------|---|
| 0<br>TIF | <p>Time Invalid Flag</p> <p>The time invalid flag is set on POR or software reset. The TSR and TPR do not increment and read as zero when this bit is set. This bit is cleared by writing the TSR register when the time counter is disabled.</p> <p>0 Time is valid.<br/>1 Time is invalid and time counter is read as zero.</p> |

**34.2.7 RTC Lock Register (RTC\_LR)**

Address: 4003\_D000h base + 18h offset = 4003\_D018h

| Bit   | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 | 23 | 22  | 21  | 20  | 19  | 18 | 17 | 16 |
|-------|----|----|----|----|----|----|----|----|----|-----|-----|-----|-----|----|----|----|
| R     |    |    |    |    |    |    |    |    | 0  |     |     |     |     |    |    |    |
| W     |    |    |    |    |    |    |    |    |    |     |     |     |     |    |    |    |
| Reset | 0  | 0  | 0  | 0  | 0  | 0  | 0  | 0  | 0  | 0   | 0   | 0   | 0   | 0  | 0  | 0  |
| Bit   | 15 | 14 | 13 | 12 | 11 | 10 | 9  | 8  | 7  | 6   | 5   | 4   | 3   | 2  | 1  | 0  |
| R     |    |    |    |    | 0  |    |    |    | 1  | LRL | SRL | CRL | TCL |    | 1  |    |
| W     |    |    |    |    |    |    |    |    |    |     |     |     |     |    |    |    |
| Reset | 0  | 0  | 0  | 0  | 0  | 0  | 0  | 0  | 1  | 1   | 1   | 1   | 1   | 1  | 1  | 1  |

**RTC\_LR field descriptions**

| Field            | Description  |
|------------------|--|
| 31–8<br>Reserved | This field is reserved.<br>This read-only field is reserved and always has the value 0.  |
| 7<br>Reserved    | This field is reserved.<br>This read-only field is reserved and always has the value 1.  |
| 6<br>LRL         | <p>Lock Register Lock</p> <p>After being cleared, this bit can be set only by POR or software reset.</p> <p>0 Lock Register is locked and writes are ignored.<br/>1 Lock Register is not locked and writes complete as normal.</p>       |
| 5<br>SRL         | <p>Status Register Lock</p> <p>After being cleared, this bit can be set only by POR or software reset.</p> <p>0 Status Register is locked and writes are ignored.<br/>1 Status Register is not locked and writes complete as normal.</p> |
| 4<br>CRL         | <p>Control Register Lock</p> <p>After being cleared, this bit can only be set by POR.</p> <p>0 Control Register is locked and writes are ignored.<br/>1 Control Register is not locked and writes complete as normal.</p>                |

Table continues on the next page...

**RTC\_LR field descriptions (continued)**

| Field           | Description  |
|-----------------|--|
| 3<br>TCL        | <p>Time Compensation Lock</p> <p>After being cleared, this bit can be set only by POR or software reset.</p> <p>0 Time Compensation Register is locked and writes are ignored.<br/>1 Time Compensation Register is not locked and writes complete as normal.</p> |
| 2–0<br>Reserved | <p>This field is reserved.</p> <p>This read-only field is reserved and always has the value 1.</p>   |

**34.2.8 RTC Interrupt Enable Register (RTC\_IER)**

Address: 4003\_D000h base + 1Ch offset = 4003\_D01Ch

|       |    |    |    |    |    |    |    |    |   |      |    |          |      |          |      |      |      |
|-------|----|----|----|----|----|----|----|----|---|------|----|----------|------|----------|------|------|------|
| Bit   | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |   | 23   | 22 | 21       | 20   | 19       | 18   | 17   | 16   |
| R     |    |    |    |    |    |    |    |    | 0 |      |    |          |      |          |      |      |      |
| W     |    |    |    |    |    |    |    |    |   |      |    |          |      |          |      |      |      |
| Reset | 0  | 0  | 0  | 0  | 0  | 0  | 0  | 0  |   | 0    | 0  | 0        | 0    | 0        | 0    | 0    | 0    |
| Bit   | 15 | 14 | 13 | 12 | 11 | 10 | 9  | 8  |   | 7    | 6  | 5        | 4    | 3        | 2    | 1    | 0    |
| R     |    |    |    |    |    |    |    |    |   | WPON |    | Reserved | TSIE | Reserved | TAIE | TOIE | TIIE |
| W     |    |    |    |    |    |    |    |    |   | 0    | 0  | 0        | 0    | 0        | 1    | 1    | 1    |
| Reset | 0  | 0  | 0  | 0  | 0  | 0  | 0  | 0  |   | 0    | 0  | 0        | 0    | 0        | 1    | 1    | 1    |

**RTC\_IER field descriptions**

| Field            | Description   |
|------------------|---|
| 31–8<br>Reserved | <p>This field is reserved.</p> <p>This read-only field is reserved and always has the value 0.</p>  |
| 7<br>WPON        | <p>Wakeup Pin On</p> <p>The wakeup pin is optional and not available on all devices. Whenever the wakeup pin is enabled and this bit is set, the wakeup pin will assert.</p> <p>0 No effect.<br/>1 If the wakeup pin is enabled, then the wakeup pin will assert.</p> |
| 6–5<br>Reserved  | <p>This field is reserved.</p>  |
| 4<br>TSIE        | <p>Time Seconds Interrupt Enable</p> <p>The seconds interrupt is an edge-sensitive interrupt with a dedicated interrupt vector. It is generated once a second and requires no software overhead (there is no corresponding status flag to clear).</p>                 |

*Table continues on the next page...*

**RTC\_IER field descriptions (continued)**

| Field      | Description  |
|------------|--|
|            | 0 Seconds interrupt is disabled.<br>1 Seconds interrupt is enabled.  |
| 3 Reserved | This field is reserved.  |
| 2 TAIE     | Time Alarm Interrupt Enable<br>0 Time alarm flag does not generate an interrupt.<br>1 Time alarm flag does generate an interrupt.          |
| 1 TOIE     | Time Overflow Interrupt Enable<br>0 Time overflow flag does not generate an interrupt.<br>1 Time overflow flag does generate an interrupt. |
| 0 TIIE     | Time Invalid Interrupt Enable<br>0 Time invalid flag does not generate an interrupt.<br>1 Time invalid flag does generate an interrupt.    |

## 34.3 Functional description

### 34.3.1 Power, clocking, and reset

The RTC is an always powered block that remains active in all low power modes.

The time counter within the RTC is clocked by a 32.768 kHz clock sourced from an external crystal using the oscillator.

The power-on-reset signal initializes all RTC registers to their default state. A software reset bit can also initialize all RTC registers.

#### 34.3.1.1 Oscillator control

The 32.768 kHz crystal oscillator is disabled at POR and must be enabled by software. After enabling the crystal oscillator, wait the oscillator startup time before setting SR[TCE] or using the oscillator clock external to the RTC.

The crystal oscillator includes tunable capacitors that can be configured by software. Do not change the capacitance unless the oscillator is disabled.

### 34.3.1.2 Software reset

Writing one to the CR[SWR] forces the equivalent of a POR to the rest of the RTC module. The CR[SWR] is not affected by the software reset and must be cleared by software.

### 34.3.1.3 Supervisor access

When the supervisor access control bit is clear, only supervisor mode software can write to the RTC registers, non-supervisor mode software will generate a bus error. Both supervisor and non-supervisor mode software can always read the RTC registers.

## 34.3.2 Time counter

The time counter consists of a 32-bit seconds counter that increments once every second and a 16-bit prescaler register that increments once every 32.768 kHz clock cycle.

The time seconds register and time prescaler register can be written only when SR[TCE] is clear. Always write to the prescaler register before writing to the seconds register, because the seconds register increments on the falling edge of bit 14 of the prescaler register.

The time prescaler register increments provided SR[TCE] is set, SR[TIF] is clear, SR[TOF] is clear, and the 32.768 kHz clock source is present. After enabling the oscillator, wait the oscillator startup time before setting SR[TCE] to allow time for the oscillator clock output to stabilize.

If the time seconds register overflows then the SR[TOF] will set and the time prescaler register will stop incrementing. Clear SR[TOF] by initializing the time seconds register. The time seconds register and time prescaler register read as zero whenever SR[TOF] is set.

SR[TIF] is set on POR and software reset and is cleared by initializing the time seconds register. The time seconds register and time prescaler register read as zero whenever SR[TIF] is set.

### 34.3.3 Compensation

The compensation logic provides an accurate and wide compensation range and can correct errors as high as 3906 ppm and as low as 0.12 ppm. The compensation factor must be calculated externally to the RTC and supplied by software to the compensation

register. The RTC itself does not calculate the amount of compensation that is required, although the 1 Hz clock is output to an external pin in support of external calibration logic.

Crystal compensation can be supported by using firmware and crystal characteristics to determine the compensation amount. Temperature compensation can be supported by firmware that periodically measures the external temperature via ADC and updates the compensation register based on a look-up table that specifies the change in crystal frequency over temperature.

The compensation logic alters the number of 32.768 kHz clock cycles it takes for the prescaler register to overflow and increment the time seconds counter. The time compensation value is used to adjust the number of clock cycles between -127 and +128. Cycles are added or subtracted from the prescaler register when the prescaler register equals 0x3FFF and then increments. The compensation interval is used to adjust the frequency at which the time compensation value is used, that is, from once a second to once every 256 seconds.

Updates to the time compensation register will not take effect until the next time the time seconds register increments and provided the previous compensation interval has expired. When the compensation interval is set to other than once a second then the compensation is applied in the first second interval and the remaining second intervals receive no compensation.

Compensation is disabled by configuring the time compensation register to zero.

### **34.3.4 Time alarm**

The time alarm register, SR[TAF], and IER[TAIE] allow the RTC to generate an interrupt at a predefined time. The 32-bit time alarm register is compared with the 32-bit time seconds register each time it increments. The SR[TAF] will set when the time alarm register equals the time seconds register and the time seconds register increments.

The time alarm flag is cleared by writing the time alarm register. This will usually be the next alarm value, although writing a value that is less than the time seconds register, such as zero, will prevent the time alarm flag from setting again. The time alarm flag cannot otherwise be disabled, although the interrupt it generates is enabled or disabled by IER[TAIE].

### 34.3.5 Update mode

The Update Mode bit in the Control register (CR[UM]) configures software write access to the Time Counter Enable (SR[TCE]) bit. When CR[UM] is clear, SR[TCE] can be written only when the LR[SRL] bit is set. When CR[UM] is set, the SR[TCE] can also be written when SR[TCE] is clear or when SR[TIF] or SR[TOF] are set. This allows the time seconds and prescaler registers to be initialized whenever time is invalidated, while preventing the time seconds and prescaler registers from being changed on the fly. When LR[SRL] is set, CR[UM] has no effect on SR[TCE].

### 34.3.6 Register lock

The lock register can be used to block write accesses to certain registers until the next POR or software reset. Locking the control register will disable the software reset. Locking the lock register will block future updates to the lock register.

Write accesses to a locked register are ignored and do not generate a bus error.

### 34.3.7 Interrupt

The RTC interrupt is asserted whenever a status flag and the corresponding interrupt enable bit are both set. It is always asserted on POR, and software reset. The RTC interrupt is enabled at the chip level by enabling the chip-specific RTC clock gate control bit. The RTC interrupt can be used to wakeup the chip from any low-power mode.

The optional RTC seconds interrupt is an edge-sensitive interrupt with a dedicated interrupt vector that is generated once a second and requires no software overhead (there is no corresponding status flag to clear). It is enabled in the RTC by the time seconds interrupt enable bit and enabled at the chip level by setting the chip-specific RTC clock gate control bit. This interrupt is optional and may not be implemented on all devices.



# **Chapter 35**

## **Universal Serial Bus OTG Controller (USBOTG)**

### **35.1 Introduction**

#### **NOTE**

For the chip-specific implementation details of this module's instances see the chip configuration information.

This section describes the USB. The OTG implementation in this module provides limited host functionality and device solutions for implementing a USB 2.0 full-speed/low-speed compliant peripheral. The OTG implementation supports the On-The-Go (OTG) addendum to the USB 2.0 Specification. Only one protocol can be active at any time. A negotiation protocol must be used to switch to a USB host functionality from a USB device. This is known as the Master Negotiation Protocol (MNP).

#### **35.1.1 USB**

The USB is a cable bus that supports data exchange between a host computer and a wide range of simultaneously accessible peripherals. The attached peripherals share USB bandwidth through a host-scheduled, token-based protocol. The bus allows peripherals to be attached, configured, used, and detached while the host and other peripherals are in operation.

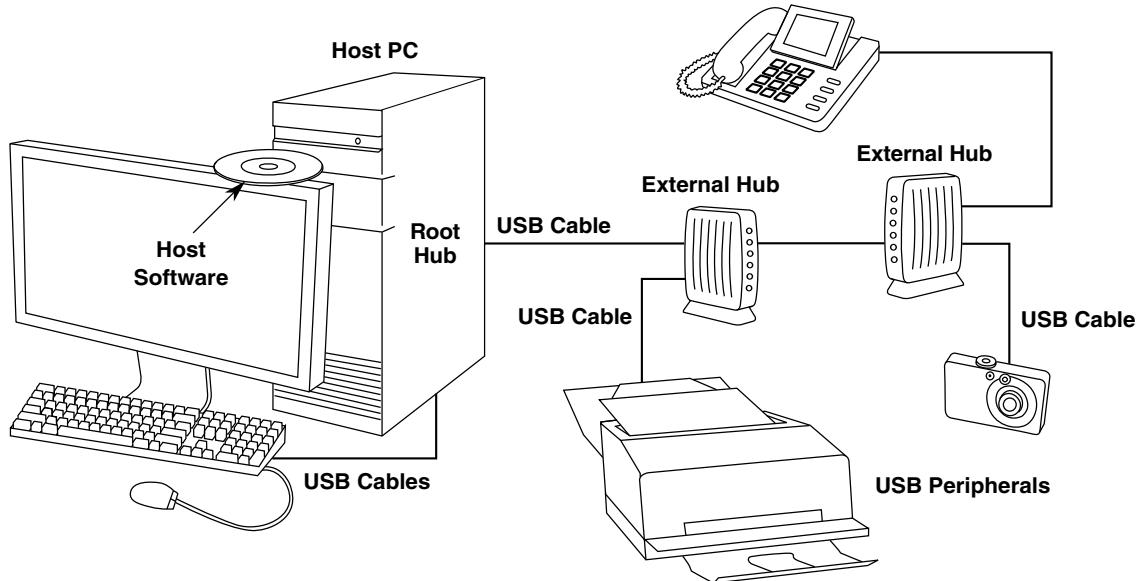
USB software provides a uniform view of the system for all application software, hiding implementation details making application software more portable. It manages the dynamic attach and detach of peripherals.

There is only one host in any USB system. The USB interface to the host computer system is referred to as the Host Controller.

There may be multiple USB devices in any system such as joysticks, speakers, printers, etc. USB devices present a standard USB interface in terms of comprehension, response, and standard capability.

The host initiates transactions to specific peripherals, whereas the device responds to control transactions. The device sends and receives data to and from the host using a standard USB data format. USB 2.0 full-speed /low-speed peripherals operate at 12Mbit/s or 1.5 Mbit/s.

For additional information, see the USB 2.0 specification.



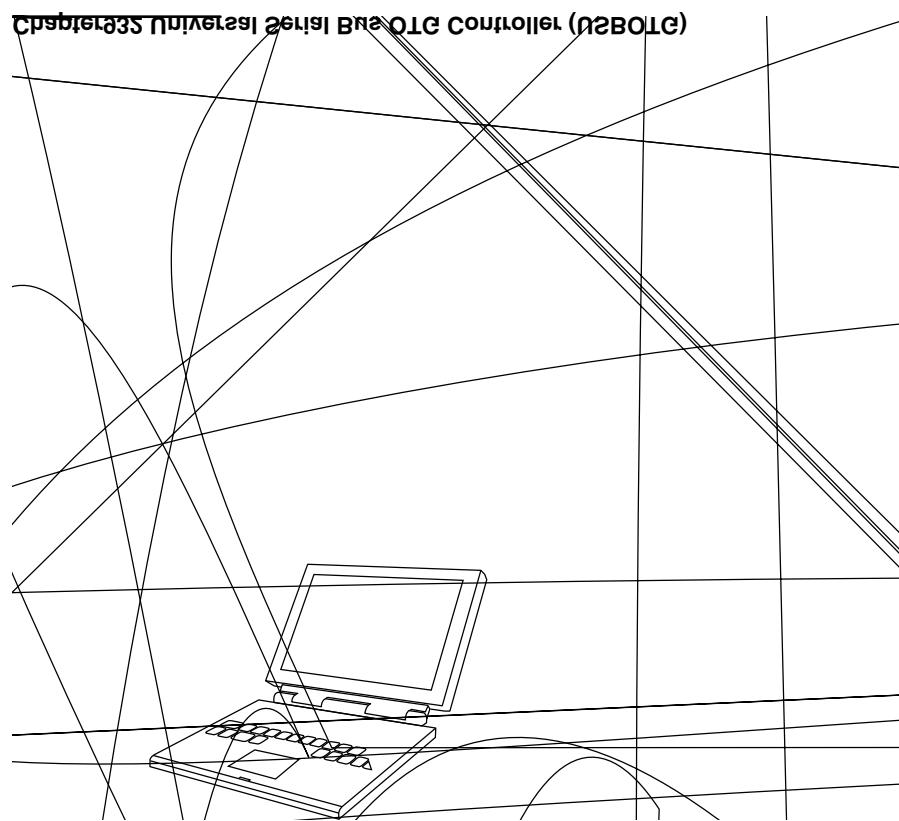
**Figure 35-1. Example USB 2.0 system configuration**

### 35.1.2 USB On-The-Go

USB is a popular standard for connecting peripherals and portable consumer electronic devices such as digital cameras and hand-held computers to host PCs. The On-The-Go (OTG) Supplement to the USB Specification extends USB to peer-to-peer application. Using USB OTG technology consumer electronics, peripherals, and portable devices can connect to each other to exchange data. For example, a digital camera can connect directly to a printer, or a keyboard can connect to a Personal Digital Assistant to exchange data.

With the USB On-The-Go product, you can develop a fully USB-compliant peripheral device that can also assume the role of a USB host. Software determines the role of the device based on hardware signals, and then initializes the device in the appropriate mode of operation (host or peripheral) based on how it is connected. After connecting the devices can negotiate using the OTG protocols to assume the role of host or peripheral based on the task to be accomplished.

For additional information, see the *On-The-Go Supplement to the USB 2.0 Specification*.



### 35.2.1 Data Structures

The function of the device operation is to transfer a request in the memory image to and from the Universal Serial Bus. To efficiently manage USB endpoint communications the USB-FS implements a Buffer Descriptor Table (BDT) in system memory. See [Figure 35-3](#).

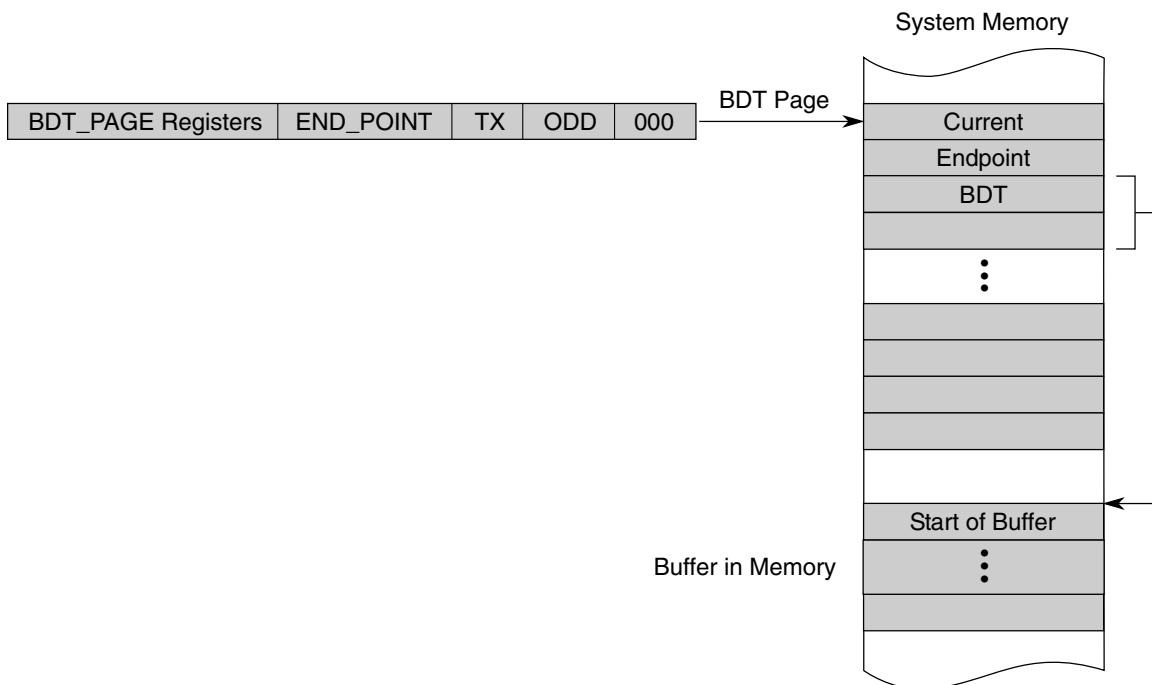
## 35.3 Programmers interface

This section discusses the major components of the programming model for the USB module.

### 35.3.1 Buffer Descriptor Table

To efficiently manage USB endpoint communications the USB-FS implements a Buffer Descriptor Table (BDT) in system memory. The BDT resides on a 512-byte boundary in system memory and is pointed to by the BDT Page Registers. Every endpoint direction requires two 8-byte Buffer Descriptor (BD) entries. Therefore, a system with 16 fully bidirectional endpoints would require 512 bytes of system memory to implement the BDT. The two BD entries allows for an EVEN BD and ODD BD entry for each endpoint direction. This allows the microprocessor to process one BD while the USB-FS is processing the other BD. Double buffering BDs in this way allows the USB-FS to transfer data easily at the maximum throughput provided by USB.

The software API intelligently manages buffers for the USB-FS by updating the BDT when needed. This allows the USB-FS to efficiently manage data transmission and reception, while the microprocessor performs communication overhead processing and other function dependent applications. Because the buffers are shared between the microprocessor and the USB-FS, a simple semaphore mechanism is used to distinguish who is allowed to update the BDT and buffers in system memory. A semaphore, the OWN bit, is cleared to 0 when the BD entry is owned by the microprocessor. The microprocessor is allowed read and write access to the BD entry and the buffer in system memory when the OWN bit is 0. When the OWN bit is set to 1, the BD entry and the buffer in system memory are owned by the USB-FS. The USB-FS now has full read and write access and the microprocessor must not modify the BD or its corresponding data buffer. The BD also contains indirect address pointers to where the actual buffer resides in system memory. This indirect address mechanism is shown in the following diagram.

**Figure 35-3. Buffer descriptor table**

### 35.3.2 RX vs. TX as a USB target device or USB host

The USB-FS core uses software control to switch between two modes of operation:

- USB target device
- USB hosts

In either mode, USB host or USB target device, the same data paths and buffer descriptors are used for the transmission and reception of data. For this reason, a USB-FS core centric nomenclature is used to describe the direction of the data transfer between the USB-FS core and the USB:

- RX (or receive) describes transfers that move data from the USB to memory.
- TX (or transmit) describes transfers that move data from memory to the USB.

The following table shows how the data direction corresponds to the USB token type in host and target device applications.

**Table 35-1. Data direction for USB host or USB target**

|        | RX           | TX           |
|--------|--------------|--------------|
| Device | OUT or SETUP | IN           |
| Host   | IN           | OUT or SETUP |

### 35.3.3 Addressing BDT entries

An understanding of the addressing mechanism of the Buffer Descriptor Table is useful when accessing endpoint data via the USB-FS or microprocessor. Some points of interest are:

- The BDT occupies up to 512 bytes of system memory.
- 16 bidirectional endpoints can be supported with a full BDT of 512 bytes.
- 16 bytes are needed for each USB endpoint direction.
- Applications with less than 16 endpoints require less RAM to implement the BDT.
- The BDT Page Registers (BDT\_PAGE) point to the starting location of the BDT.
- The BDT must be located on a 512-byte boundary in system memory.
- All enabled TX and RX endpoint BD entries are indexed into the BDT to allow easy access via the USB-FS or MCU core.

When a USB token on an enabled endpoint is received, the USB-FS uses its integrated DMA controller to interrogate the BDT. The USB-FS reads the corresponding endpoint BD entry to determine whether it owns the BD and corresponding buffer in system memory.

To compute the entry point into the BDT, the BDT\_PAGE registers are concatenated with the current endpoint and the TX and ODD fields to form a 32-bit address. This address mechanism is shown below:

**Table 35-2. BDT address calculation fields**

| Field     | Description   |
|-----------|---|
| BDT_PAGE  | BDT_PAGE registers in the Control Register Block  |
| END_POINT | END POINT field from the USB TOKEN  |
| TX        | 1 for transmit transfers and 0 for receive transfers  |
| ODD       | Maintained within the USB-FS SIE. It corresponds to the buffer currently in use. The buffers are used in a ping-pong fashion. |

### 35.3.4 Buffer Descriptors (BDs)

A buffer descriptor provides endpoint buffer control information for the USB-FS and processor. The Buffer Descriptors have different meaning based on whether it is the USB-FS or processor reading the BD in memory.

The USB-FS Controller uses the data stored in the BDs to determine:

- Who owns the buffer in system memory
- Data0 or Data1 PID
- Whether to release ownership upon packet completion
- No address increment (FIFO mode)
- Whether data toggle synchronization is enabled
- How much data is to be transmitted or received
- Where the buffer resides in system memory

While the processor uses the data stored in the BDs to determine:

- Who owns the buffer in system memory
- Data0 or Data1 PID
- The received TOKEN PID
- How much data was transmitted or received
- Where the buffer resides in system memory

The format for the BD is shown in the following figure.

**Table 35-3. Buffer descriptor format**

| 31:26                    | 25:16           | 15:8 | 7   | 6       | 5                   | 4                   | 3                  | 2                        | 1 | 0 |
|--------------------------|-----------------|------|-----|---------|---------------------|---------------------|--------------------|--------------------------|---|---|
| RSVD                     | BC<br>(10 bits) | RSVD | OWN | DATA0/1 | KEEP/<br>TOK_PID[3] | NINC/<br>TOK_PID[2] | DTS/<br>TOK_PID[1] | BDT_STALL/<br>TOK_PID[0] | 0 | 0 |
| Buffer Address (32-Bits) |                 |      |     |         |                     |                     |                    |                          |   |   |

**Table 35-4. Buffer descriptor fields**

| Field | Description  |
|-------|--|
| 31–26 | Reserved   |
| RSVD  |  |
| 25–16 | Byte Count   |
| BC    | Represents the 10-bit byte count. The USB-FS SIE changes this field upon the completion of a RX transfer with the byte count of the data received. |
| 15–8  | Reserved   |
| RSVD  |  |

*Table continues on the next page...*

**Table 35-4. Buffer descriptor fields (continued)**

| Field                        | Description   |
|------------------------------|---|
| 7<br>OWN                     | <p>Determines whether the processor or the USB-FS currently owns the buffer. Except when KEEP=1, the SIE hands ownership back to the processor after completing the token by clearing this bit.</p> <p>This must always be the last byte of the BD that the processor updates when it initializes a BD.</p> <p>0 The processor has exclusive access to the BD. The USB-FS ignores all other fields in the BD.</p> <p>1 USB-FS has exclusive access to the BD. After the BD has been assigned to the USB-FS, the processor should not change it in any way.</p>  |
| 6<br>DATA0/1                 | Defines whether a DATA0 field (DATA0/1=0) or a DATA1 (DATA0/1=1) field was transmitted or received. It is unchanged by the USB-FS.  |
| 5<br>KEEP/<br>TOK_PID[3]     | <p>Typically, this bit is 1 with ISO endpoints feeding a FIFO. The microprocessor is not informed that a token has been processed, the data is simply transferred to or from the FIFO. When KEEP is set, normally the NINC bit is also set to prevent address increment.</p> <p>0 Bit 3 of the current token PID is written back to the BD by the USB-FS. Allows the USB-FS to release the BD when a token has been processed.</p> <p>1 This bit is unchanged by the USB-FS. If the OWN bit also is set, the BD remains owned by the USB-FS forever.</p>  |
| 4<br>NINC/<br>TOK_PID[2]     | <p>No Increment (NINC)</p> <p>Disables the DMA engine address increment. This forces the DMA engine to read or write from the same address. This is useful for endpoints when data needs to be read from or written to a single location such as a FIFO. Typically this bit is set with the KEEP bit for ISO endpoints that are interfacing to a FIFO.</p> <p>0 The USB-FS writes bit 2 of the current token PID to the BD.</p> <p>1 This bit is unchanged by the USB-FS.</p>   |
| 3<br>DTS/<br>TOK_PID[1]      | <p>Setting this bit enables the USB-FS to perform Data Toggle Synchronization.</p> <ul style="list-style-type: none"> <li>If KEEP=0, bit 1 of the current token PID is written back to the BD.</li> <li>If KEEP=1, this bit is unchanged by the USB-FS.</li> </ul> <p>0 Data Toggle Synchronization is disabled.</p> <p>1 Enables the USB-FS to perform Data Toggle Synchronization.</p>  |
| 2<br>BDT_STALL<br>TOK_PID[0] | <p>Setting this bit causes the USB-FS to issue a STALL handshake if a token is received by the SIE that would use the BDT in this location. The BDT is not consumed by the SIE (the owns bit remains set and the rest of the BDT is unchanged) when a BDT-STALL bit is set.</p> <ul style="list-style-type: none"> <li>If KEEP=0, bit 0 of the current token PID is written back to the BD.</li> <li>If KEEP=1, this bit is unchanged by the USB-FS.</li> </ul> <p>0 No stall issued.</p> <p>1 The BDT is not consumed by the SIE (the OWN bit remains set and the rest of the BDT is unchanged).</p> |

*Table continues on the next page...*

**Table 35-4. Buffer descriptor fields (continued)**

| Field           | Description  |
|-----------------|--|
| TOK_PID[n]      | <p>Bits [5:2] can also represent the current token PID. The current token PID is written back in to the BD by the USB-FS when a transfer completes. The values written back are the token PID values from the USB specification:</p> <ul style="list-style-type: none"> <li>• 0x1h for an OUT token.</li> <li>• 0x9h for an IN token.</li> <li>• 0xDh for a SETUP token.</li> </ul> <p>In host mode, this field is used to report the last returned PID or a transfer status indication. The possible values returned are:</p> <ul style="list-style-type: none"> <li>• 0x3h DATA0</li> <li>• 0xBh DATA1</li> <li>• 0x2h ACK</li> <li>• 0xEh STALL</li> <li>• 0xAh NAK</li> <li>• 0x0h Bus Timeout</li> <li>• 0xFh Data Error</li> </ul> |
| 1–0<br>Reserved | Reserved, should read as zeroes.   |
| ADDR[31:0]      | <p>Address</p> <p>Represents the 32-bit buffer address in system memory. These bits are unchanged by the USB-FS.</p>   |

### 35.3.5 USB transaction

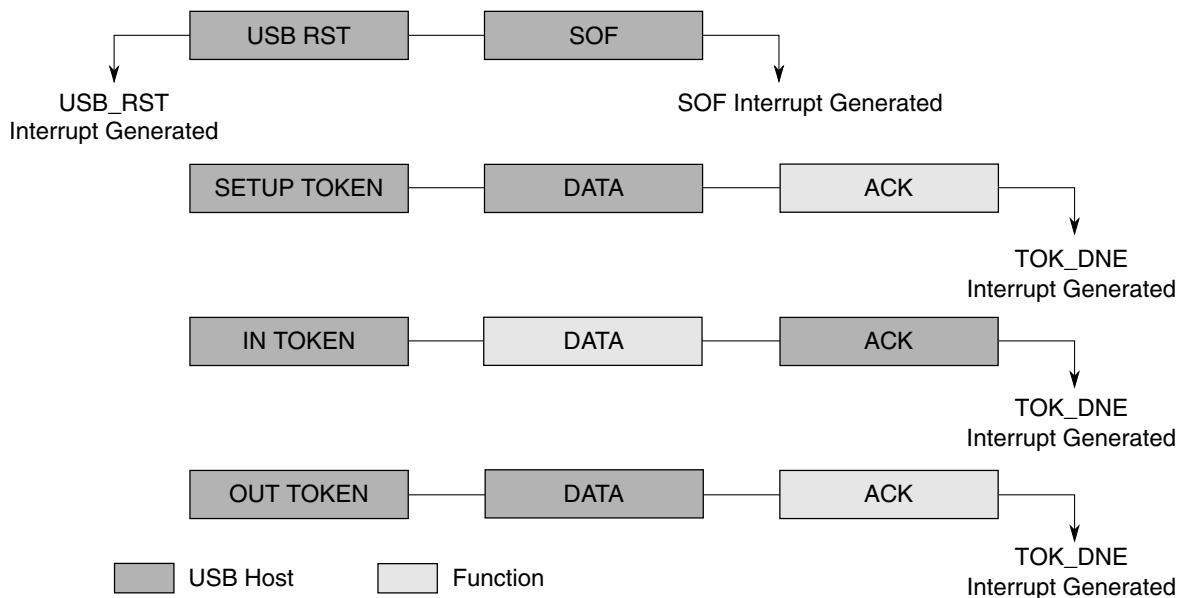
When the USB-FS transmits or receives data, it computes the BDT address using the address generation shown in "Addressing Buffer Descriptor Entries" table.

If OWN =1, the following process occurs:

1. The USB-FS reads the BDT.
2. The SIE transfers the data via the DMA to or from the buffer pointed to by the ADDR field of the BD.
3. When the TOKEN is complete, the USB-FS updates the BDT and, if KEEP=0, changes the OWN bit to 0.
4. The STAT register is updated and the TOK\_DNE interrupt is set.
5. When the processor processes the TOK\_DNE interrupt, it reads from the status register all the information needed to process the endpoint.
6. At this point, the processor allocates a new BD so that additional USB data can be transmitted or received for that endpoint, and then processes the last BD.

The following figure shows a timeline of how a typical USB token is processed after the BDT is read and OWN=1.

## Programmers interface



**Figure 35-4. USB token transaction**

The USB has two sources for the DMA overrun error:

### Memory Latency

The memory latency may be too high and cause the receive FIFO to overflow. This is predominantly a hardware performance issue, usually caused by transient memory access issues.

### Oversized Packets

The packet received may be larger than the negotiated *MaxPacket* size. Typically, this is caused by a software bug. For DMA overrun errors due to oversized data packets, the USB specification is ambiguous. It assumes correct software drivers on both sides. NAKing the packet can result in retransmission of the already oversized packet data. Therefore, in response to oversized packets, the USB core continues ACKing the packet for non-isochronous transfers.

**Table 35-5. USB responses to DMA overrun errors**

| Errors due to Memory Latency   | Errors due to Oversized Packets  |
|--|--|
| Non-Acknowledgment (NAK) or Bus Timeout (BTO) — See bit 4 in "Error Interrupt Status Register (ERRSTAT)" as appropriate for the class of transaction.  | Continues acknowledging (ACKing) the packet for non-isochronous transfers.   |
| —  | The data written to memory is clipped to the MaxPacket size so as not to corrupt system memory.  |
| The DMAERR bit is set in the ERRSTAT register for host and device modes of operation. Depending on the values of the INTENB and ERREN register, the core may assert an interrupt to notify the processor of the DMA error. | Asserts ERRSTAT[DMAERR], which can trigger an interrupt and TOKDNE interrupt fires. Note: The TOK_PID field of the BDT is not 1111 because the DMAERR is not due to latency. |

Table continues on the next page...

**Table 35-5. USB responses to DMA overrun errors (continued)**

| Errors due to Memory Latency   | Errors due to Oversized Packets   |
|--|---|
| <ul style="list-style-type: none"> <li>For host mode, the TOKDNE interrupt is generated and the TOK_PID field of the BDT is 1111 to indicate the DMA latency error. Host mode software can decide to retry or move to next scheduled item.</li> <li>In device mode, the BDT is not written back nor is the TOKDNE interrupt triggered because it is assumed that a second attempt is queued and will succeed in the future.</li> </ul> | The packet length field written back to the BDT is the MaxPacket value that represents the length of the clipped data actually written to memory. |
| From here, the software can decide an appropriate course of action for future transactions such as stalling the endpoint, canceling the transfer, disabling the endpoint, etc.   |   |

## 35.4 Memory map/Register definitions

This section provides the memory map and detailed descriptions of all USB interface registers.

**USB memory map**

| Absolute address (hex) | Register name                                      | Width (in bits) | Access | Reset value | Section/page |
|------------------------|--|-----------------|--------|-------------|--------------|
| 4007_2000              | Peripheral ID register (USB0_PERID)                | 8               | R      | 04h         | 35.4.1/623   |
| 4007_2004              | Peripheral ID Complement register (USB0_IDCOMP)    | 8               | R      | FBh         | 35.4.2/624   |
| 4007_2008              | Peripheral Revision register (USB0_REV)            | 8               | R      | 33h         | 35.4.3/624   |
| 4007_200C              | Peripheral Additional Info register (USB0_ADDINFO) | 8               | R      | 01h         | 35.4.4/625   |
| 4007_2010              | OTG Interrupt Status register (USB0_OTGISTAT)      | 8               | R/W    | 00h         | 35.4.5/625   |
| 4007_2014              | OTG Interrupt Control Register (USB0_OTGICR)       | 8               | R/W    | 00h         | 35.4.6/626   |
| 4007_2018              | OTG Status register (USB0_OTGSTAT)                 | 8               | R/W    | 00h         | 35.4.7/627   |
| 4007_201C              | OTG Control register (USB0_OTGCTL)                 | 8               | R/W    | 00h         | 35.4.8/628   |
| 4007_2080              | Interrupt Status register (USB0_ISTAT)             | 8               | R/W    | 00h         | 35.4.9/629   |
| 4007_2084              | Interrupt Enable register (USB0_INTEN)             | 8               | R/W    | 00h         | 35.4.10/630  |
| 4007_2088              | Error Interrupt Status register (USB0_ERRSTAT)     | 8               | R/W    | 00h         | 35.4.11/631  |
| 4007_208C              | Error Interrupt Enable register (USB0_ERREN)       | 8               | R/W    | 00h         | 35.4.12/632  |
| 4007_2090              | Status register (USB0_STAT)                        | 8               | R      | 00h         | 35.4.13/633  |
| 4007_2094              | Control register (USB0_CTL)                        | 8               | R/W    | 00h         | 35.4.14/634  |
| 4007_2098              | Address register (USB0_ADDR)                       | 8               | R/W    | 00h         | 35.4.15/635  |

*Table continues on the next page...*

**USB memory map (continued)**

| Absolute address (hex) | Register name                             | Width (in bits) | Access | Reset value | Section/page                 |
|------------------------|---|-----------------|--------|-------------|------------------------------|
| 4007_209C              | BDT Page Register 1 (USB0_BDTPAGE1)       | 8               | R/W    | 00h         | <a href="#">35.4.16/ 636</a> |
| 4007_20A0              | Frame Number Register Low (USB0_FRMNUML)  | 8               | R/W    | 00h         | <a href="#">35.4.17/ 636</a> |
| 4007_20A4              | Frame Number Register High (USB0_FRMNUMH) | 8               | R/W    | 00h         | <a href="#">35.4.18/ 637</a> |
| 4007_20A8              | Token register (USB0_TOKEN)               | 8               | R/W    | 00h         | <a href="#">35.4.19/ 637</a> |
| 4007_20AC              | SOF Threshold Register (USB0_SOFTHLD)     | 8               | R/W    | 00h         | <a href="#">35.4.20/ 638</a> |
| 4007_20B0              | BDT Page Register 2 (USB0_BDTPAGE2)       | 8               | R/W    | 00h         | <a href="#">35.4.21/ 639</a> |
| 4007_20B4              | BDT Page Register 3 (USB0_BDTPAGE3)       | 8               | R/W    | 00h         | <a href="#">35.4.22/ 639</a> |
| 4007_20C0              | Endpoint Control register (USB0_ENDPT0)   | 8               | R/W    | 00h         | <a href="#">35.4.23/ 639</a> |
| 4007_20C4              | Endpoint Control register (USB0_ENDPT1)   | 8               | R/W    | 00h         | <a href="#">35.4.23/ 639</a> |
| 4007_20C8              | Endpoint Control register (USB0_ENDPT2)   | 8               | R/W    | 00h         | <a href="#">35.4.23/ 639</a> |
| 4007_20CC              | Endpoint Control register (USB0_ENDPT3)   | 8               | R/W    | 00h         | <a href="#">35.4.23/ 639</a> |
| 4007_20D0              | Endpoint Control register (USB0_ENDPT4)   | 8               | R/W    | 00h         | <a href="#">35.4.23/ 639</a> |
| 4007_20D4              | Endpoint Control register (USB0_ENDPT5)   | 8               | R/W    | 00h         | <a href="#">35.4.23/ 639</a> |
| 4007_20D8              | Endpoint Control register (USB0_ENDPT6)   | 8               | R/W    | 00h         | <a href="#">35.4.23/ 639</a> |
| 4007_20DC              | Endpoint Control register (USB0_ENDPT7)   | 8               | R/W    | 00h         | <a href="#">35.4.23/ 639</a> |
| 4007_20E0              | Endpoint Control register (USB0_ENDPT8)   | 8               | R/W    | 00h         | <a href="#">35.4.23/ 639</a> |
| 4007_20E4              | Endpoint Control register (USB0_ENDPT9)   | 8               | R/W    | 00h         | <a href="#">35.4.23/ 639</a> |
| 4007_20E8              | Endpoint Control register (USB0_ENDPT10)  | 8               | R/W    | 00h         | <a href="#">35.4.23/ 639</a> |
| 4007_20EC              | Endpoint Control register (USB0_ENDPT11)  | 8               | R/W    | 00h         | <a href="#">35.4.23/ 639</a> |
| 4007_20F0              | Endpoint Control register (USB0_ENDPT12)  | 8               | R/W    | 00h         | <a href="#">35.4.23/ 639</a> |
| 4007_20F4              | Endpoint Control register (USB0_ENDPT13)  | 8               | R/W    | 00h         | <a href="#">35.4.23/ 639</a> |
| 4007_20F8              | Endpoint Control register (USB0_ENDPT14)  | 8               | R/W    | 00h         | <a href="#">35.4.23/ 639</a> |

*Table continues on the next page...*

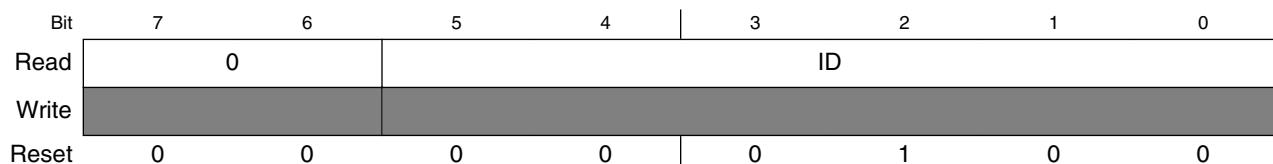
**USB memory map (continued)**

| Absolute address (hex) | Register name                                     | Width (in bits) | Access | Reset value | Section/page                 |
|------------------------|---|-----------------|--------|-------------|------------------------------|
| 4007_20FC              | Endpoint Control register (USB0_ENDPT15)          | 8               | R/W    | 00h         | <a href="#">35.4.23/ 639</a> |
| 4007_2100              | USB Control register (USB0_USBCTRL)               | 8               | R/W    | C0h         | <a href="#">35.4.24/ 640</a> |
| 4007_2104              | USB OTG Observe register (USB0_OBSERVE)           | 8               | R      | 50h         | <a href="#">35.4.25/ 641</a> |
| 4007_2108              | USB OTG Control register (USB0_CONTROL)           | 8               | R/W    | 00h         | <a href="#">35.4.26/ 642</a> |
| 4007_210C              | USB Transceiver Control Register 0 (USB0_USBTRC0) | 8               | R/W    | 00h         | <a href="#">35.4.27/ 642</a> |
| 4007_2114              | Frame Adjust Register (USB0_USBFRMADJUST)         | 8               | R/W    | 00h         | <a href="#">35.4.28/ 643</a> |

**35.4.1 Peripheral ID register (USBx\_PERID)**

Reads back the value of 0x04. This value is defined for the USB peripheral.

Address: 4007\_2000h base + 0h offset = 4007\_2000h

**USBx\_PERID field descriptions**

| Field           | Description   |
|-----------------|---|
| 7–6<br>Reserved | This field is reserved.<br>This read-only field is reserved and always has the value 0. |
| 5–0<br>ID       | Peripheral Identification<br><br>This field always reads 0x4h.                          |

### 35.4.2 Peripheral ID Complement register (USBx\_IDCOMP)

Reads back the complement of the Peripheral ID register. For the USB peripheral, the value is 0xFB.

Address: 4007\_2000h base + 4h offset = 4007\_2004h

| Bit   | 7 | 6 | 5 | 4 |  | 3   | 2 | 1 | 0 |
|-------|---|---|---|---|--|-----|---|---|---|
| Read  | 1 |   |   |   |  | NID |   |   |   |
| Write |   |   |   |   |  |     |   |   |   |
| Reset | 1 | 1 | 1 | 1 |  | 1   | 0 | 1 | 1 |

### 35.4.4 Peripheral Additional Info register (USBx\_ADDINFO)

Reads back the value of the fixed Interrupt Request Level (IRQNUM) along with the Host Enable bit.

Address: 4007\_2000h base + Ch offset = 4007\_200Ch

| Bit   | 7 | 6 | 5 | 4      | 3 | 2 | 1 | 0      |
|-------|---|---|---|--------|---|---|---|--------|
| Read  |   |   |   | IRQNUM |   | 0 |   | IEHOST |
| Write |   |   |   |        |   |   |   |        |
| Reset | 0 | 0 | 0 | 0      | 0 | 0 | 0 | 1      |

**USBx\_ADDINFO field descriptions**

| Field        | Description   |
|--------------|---|
| 7–3 IRQNUM   | Assigned Interrupt Request Number   |
| 2–1 Reserved | This field is reserved.<br>This read-only field is reserved and always has the value 0. |
| 0 IEHOST     | When this bit is set, the USB peripheral is operating in host mode.                     |

### 35.4.5 OTG Interrupt Status register (USBx\_OTGISTAT)

Records changes of the ID sense and VBUS signals. Software can read this register to determine the event that triggers interrupt. Only bits that have changed since the last software read are set. Writing a one to a bit clears the associated interrupt.

Address: 4007\_2000h base + 10h offset = 4007\_2010h

| Bit   | 7     | 6       | 5              | 4 | 3           | 2          | 1 | 0        |
|-------|-------|---------|----------------|---|-------------|------------|---|----------|
| Read  | IDCHG | ONEMSEC | LINE_STATE_CHG | 0 | SESSVLDC_HG | B_SESS_CHG | 0 | AVBUSCHG |
| Write |       |         |                |   |             |            |   |          |
| Reset | 0     | 0       | 0              | 0 | 0           | 0          | 0 | 0        |

**USBx\_OTGISTAT field descriptions**

| Field            | Description   |
|------------------|---|
| 7 IDCHG          | This bit is set when a change in the ID Signal from the USB connector is sensed.  |
| 6 ONEMSEC        | This bit is set when the 1 millisecond timer expires. This bit stays asserted until cleared by software. The interrupt must be serviced every millisecond to avoid losing 1msec counts. |
| 5 LINE_STATE_CHG | This bit is set when the USB line state changes. The interrupt associated with this bit can be used to detect Reset, Resume, Connect, and Data Line Pulse signaling                     |

*Table continues on the next page...*

**USBx\_OTGISTAT field descriptions (continued)**

| Field        | Description  |
|--------------|--|
| 4 Reserved   | This field is reserved.<br>This read-only field is reserved and always has the value 0.                    |
| 3 SESSVLDCHG | This bit is set when a change in VBUS is detected indicating a session valid or a session no longer valid. |
| 2 B_SESS_CHG | This bit is set when a change in VBUS is detected on a B device.   |
| 1 Reserved   | This field is reserved.<br>This read-only field is reserved and always has the value 0.                    |
| 0 AVBUSCHG   | This bit is set when a change in VBUS is detected on an A device.  |

**35.4.6 OTG Interrupt Control Register (USBx\_OTGICR)**

Enables the corresponding interrupt status bits defined in the OTG Interrupt Status Register.

Address: 4007\_2000h base + 14h offset = 4007\_2014h

| Bit   | 7    | 6             | 5               | 4 | 3             | 2       | 1 | 0       |
|-------|------|---------------|-----------------|---|---------------|---------|---|---------|
| Read  | IDEN | ONEMSEC<br>EN | LINESTATE<br>EN | 0 | SESSVLDE<br>N | BSESSEN | 0 | AVBUSEN |
| Write | 0    | 0             | 0               | 0 | 0             | 0       | 0 | 0       |
| Reset | 0    | 0             | 0               | 0 | 0             | 0       | 0 | 0       |

**USBx\_OTGICR field descriptions**

| Field         | Description   |
|---------------|---|
| 7 IDEN        | ID Interrupt Enable<br>0 The ID interrupt is disabled<br>1 The ID interrupt is enabled                                  |
| 6 ONEMSECEN   | One Millisecond Interrupt Enable<br>0 Disables the 1ms timer interrupt.<br>1 Enables the 1ms timer interrupt.           |
| 5 LINESTATEEN | Line State Change Interrupt Enable<br>0 Disables the LINE_STAT_CHG interrupt.<br>1 Enables the LINE_STAT_CHG interrupt. |
| 4 Reserved    | This field is reserved.<br>This read-only field is reserved and always has the value 0.                                 |
| 3 SESSVLDEN   | Session Valid Interrupt Enable<br>0 Disables the SESSVLDCHG interrupt.<br>1 Enables the SESSVLDCHG interrupt.           |
| 2 BSESSEN     | B Session END Interrupt Enable  |

*Table continues on the next page...*

**USBx\_OTGICR field descriptions (continued)**

| Field      | Description  |
|------------|--|
|            | 0 Disables the B_SESS_CHG interrupt.<br>1 Enables the B_SESS_CHG interrupt.                              |
| 1 Reserved | This field is reserved.<br>This read-only field is reserved and always has the value 0.                  |
| 0 AVBUSEN  | A VBUS Valid Interrupt Enable<br>0 Disables the AVBUSCHG interrupt.<br>1 Enables the AVBUSCHG interrupt. |

**35.4.7 OTG Status register (USBx\_OTGSTAT)**

Displays the actual value from the external comparator outputs of the ID pin and VBUS.

Address: 4007\_2000h base + 18h offset = 4007\_2018h

| Bit   | 7        | 6         | 5               | 4        |
|-------|----------|-----------|-----------------|----------|
| Read  | ID       | ONEMSECEN | LINESTATESTABLE | 0        |
| Write |          |           |                 |          |
| Reset | 0        | 0         | 0               | 0        |
| Bit   | 3        | 2         | 1               | 0        |
| Read  | SESS_VLD | BSESSEND  | 0               | AVBUSVLD |
| Write |          |           |                 |          |
| Reset | 0        | 0         | 0               | 0        |

**USBx\_OTGSTAT field descriptions**

| Field             | Description  |
|-------------------|--|
| 7 ID              | Indicates the current state of the ID pin on the USB connector<br>0 Indicates a Type A cable is plugged into the USB connector.<br>1 Indicates no cable is attached or a Type B cable is plugged into the USB connector.   |
| 6 ONEMSECEN       | This bit is reserved for the 1ms count, but it is not useful to software.  |
| 5 LINESTATESTABLE | Indicates that the internal signals that control the LINE_STATE_CHG field of OTGISTAT are stable for at least 1 millisecond. First read LINE_STATE_CHG field and then read this field. If this field reads as 1, then the value of LINE_STATE_CHG can be considered stable.<br>0 The LINE_STAT_CHG bit is not yet stable.<br>1 The LINE_STAT_CHG bit has been debounced and is stable. |
| 4 Reserved        | This field is reserved.<br>This read-only field is reserved and always has the value 0.  |
| 3 SESS_VLD        | Session Valid<br>0 The VBUS voltage is below the B session valid threshold<br>1 The VBUS voltage is above the B session valid threshold.   |

*Table continues on the next page...*

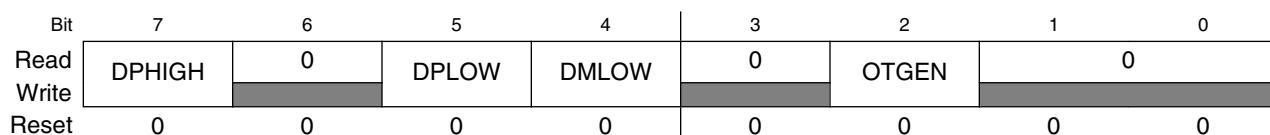
**USBx\_OTGSTAT field descriptions (continued)**

| Field         | Description   |
|---------------|---|
| 2<br>BSESEND  | B Session End<br><br>0 The VBUS voltage is above the B session end threshold.<br>1 The VBUS voltage is below the B session end threshold. |
| 1<br>Reserved | This field is reserved.<br>This read-only field is reserved and always has the value 0.   |
| 0<br>AVBUSVLD | A VBUS Valid<br><br>0 The VBUS voltage is below the A VBUS Valid threshold.<br>1 The VBUS voltage is above the A VBUS Valid threshold.    |

**35.4.8 OTG Control register (USBx\_OTGCTL)**

Controls the operation of VBUS and Data Line termination resistors.

Address: 4007\_2000h base + 1Ch offset = 4007\_201Ch

**USBx\_OTGCTL field descriptions**

| Field         | Description   |
|---------------|---|
| 7<br>DPHIGH   | D+ Data Line pullup resistor enable<br><br>0 D+ pullup resistor is not enabled<br>1 D+ pullup resistor is enabled   |
| 6<br>Reserved | This field is reserved.<br>This read-only field is reserved and always has the value 0.   |
| 5<br>DPLOW    | D+ Data Line pull-down resistor enable<br><br>This bit should always be enabled together with bit 4 (DMLOW)<br><br>0 D+ pulldown resistor is not enabled.<br>1 D+ pulldown resistor is enabled. |
| 4<br>DMLOW    | D- Data Line pull-down resistor enable<br><br>0 D- pulldown resistor is not enabled.<br>1 D- pulldown resistor is enabled.  |
| 3<br>Reserved | This field is reserved.<br>This read-only field is reserved and always has the value 0.   |
| 2<br>OTGEN    | On-The-Go pullup/pulldown resistor enable   |

*Table continues on the next page...*

**USBx\_OTGCTL field descriptions (continued)**

| Field        | Description   |
|--------------|---|
|              | 0 If USB_EN is 1 and HOST_MODE is 0 in the Control Register (CTL), then the D+ Data Line pull-up resistors are enabled. If HOST_MODE is 1 the D+ and D- Data Line pull-down resistors are engaged.<br>1 The pull-up and pull-down controls in this register are used. |
| 1–0 Reserved | This field is reserved.<br>This read-only field is reserved and always has the value 0.   |

**35.4.9 Interrupt Status register (USBx\_ISTAT)**

Contains fields for each of the interrupt sources within the USB Module. Each of these fields are qualified with their respective interrupt enable bits. All fields of this register are logically OR'd together along with the OTG Interrupt Status Register (OTGSTAT) to form a single interrupt source for the processor's interrupt controller. After an interrupt bit has been set it may only be cleared by writing a one to the respective interrupt bit. This register contains the value of 0x00 after a reset.

Address: 4007\_2000h base + 80h offset = 4007\_2080h

| Bit   | 7     | 6      | 5      | 4     | 3      | 2      | 1     | 0      |
|-------|-------|--------|--------|-------|--------|--------|-------|--------|
| Read  | STALL | ATTACH | RESUME | SLEEP | TOKDNE | SOFTOK | ERROR | USBRST |
| Write | w1c   | w1c    | w1c    | w1c   | w1c    | w1c    | w1c   | w1c    |
| Reset | 0     | 0      | 0      | 0     | 0      | 0      | 0     | 0      |

**USBx\_ISTAT field descriptions**

| Field    | Description  |
|----------|--|
| 7 STALL  | Stall Interrupt<br>In Target mode this bit is asserted when a STALL handshake is sent by the SIE.<br>In Host mode this bit is set when the USB Module detects a STALL acknowledge during the handshake phase of a USB transaction. This interrupt can be used to determine whether the last USB transaction was completed successfully or stalled. |
| 6 ATTACH | Attach Interrupt<br>This bit is set when the USB Module detects an attach of a USB device. This signal is only valid if HOSTMODEEN is true. This interrupt signifies that a peripheral is now present and must be configured.  |
| 5 RESUME | This bit is set depending upon the DP/DM signals, and can be used to signal remote wake-up signaling on the USB bus. When not in suspend mode this interrupt must be disabled.   |
| 4 SLEEP  | This bit is set when the USB Module detects a constant idle on the USB bus for 3 ms. The sleep timer is reset by activity on the USB bus.  |
| 3 TOKDNE | This bit is set when the current token being processed has completed. The processor must immediately read the STATUS (STAT) register to determine the EndPoint and BD used for this token. Clearing this bit (by writing a one) causes STAT to be cleared or the STAT holding register to be loaded into the STAT register.                        |
| 2 SOFTOK | This bit is set when the USB Module receives a Start Of Frame (SOF) token.   |

*Table continues on the next page...*

**USBx\_ISTAT field descriptions (continued)**

| Field       | Description   |
|-------------|---|
|             | In Host mode this field is set when the SOF threshold is reached, so that software can prepare for the next SOF.  |
| 1<br>ERROR  | This bit is set when any of the error conditions within Error Interrupt Status (ERRSTAT) register occur. The processor must then read the ERRSTAT register to determine the source of the error.  |
| 0<br>USBRST | This bit is set when the USB Module has decoded a valid USB reset. This informs the processor that it should write 0x00 into the address register and enable endpoint 0. USBRST is set after a USB reset has been detected for 2.5 microseconds. It is not asserted again until the USB reset condition has been removed and then reasserted. |

**35.4.10 Interrupt Enable register (USBx\_INTEN)**

Contains enable fields for each of the interrupt sources within the USB Module. Setting any of these bits enables the respective interrupt source in the ISTAT register. This register contains the value of 0x00 after a reset.

Address: 4007\_2000h base + 84h offset = 4007\_2084h

| Bit        | 7       | 6        | 5        | 4       | 3        | 2        | 1       | 0        |
|------------|---------|----------|----------|---------|----------|----------|---------|----------|
| Read Write | STALLEN | ATTACHEN | RESUMEEN | SLEEPEN | TOKDNEEN | SOFTOKEN | ERROREN | USBRSTEN |
| Reset      | 0       | 0        | 0        | 0       | 0        | 0        | 0       | 0        |

**USBx\_INTEN field descriptions**

| Field         | Description  |
|---------------|--|
| 7<br>STALLEN  | STALL Interrupt Enable<br><br>0 Disables the STALL interrupt.<br>1 Enables the STALL interrupt.    |
| 6<br>ATTACHEN | ATTACH Interrupt Enable<br><br>0 Disables the ATTACH interrupt.<br>1 Enables the ATTACH interrupt. |
| 5<br>RESUMEEN | RESUME Interrupt Enable<br><br>0 Disables the RESUME interrupt.<br>1 Enables the RESUME interrupt. |
| 4<br>SLEEPEN  | SLEEP Interrupt Enable<br><br>0 Disables the SLEEP interrupt.<br>1 Enables the SLEEP interrupt.    |
| 3<br>TOKDNEEN | TOKDNE Interrupt Enable<br><br>0 Disables the TOKDNE interrupt.<br>1 Enables the TOKDNE interrupt. |
| 2<br>SOFTOKEN | SOFTOK Interrupt Enable  |

*Table continues on the next page...*

**USBx\_INTEN field descriptions (continued)**

| Field         | Description  |
|---------------|--|
|               | 0 Disables the SOFTOK interrupt.<br>1 Enables the SOFTOK interrupt.                            |
| 1<br>ERROREN  | ERROR Interrupt Enable<br>0 Disables the ERROR interrupt.<br>1 Enables the ERROR interrupt.    |
| 0<br>USBRSTEN | USBRST Interrupt Enable<br>0 Disables the USBRST interrupt.<br>1 Enables the USBRST interrupt. |

**35.4.11 Error Interrupt Status register (USBx\_ERRSTAT)**

Contains enable bits for each of the error sources within the USB Module. Each of these bits are qualified with their respective error enable bits. All bits of this register are logically OR'd together and the result placed in the ERROR bit of the ISTAT register. After an interrupt bit has been set it may only be cleared by writing a one to the respective interrupt bit. Each bit is set as soon as the error conditions is detected. Therefore, the interrupt does not typically correspond with the end of a token being processed. This register contains the value of 0x00 after a reset.

Address: 4007\_2000h base + 88h offset = 4007\_2088h

| Bit   | 7      | 6 | 5      | 4      | 3    | 2     | 1       | 0      |
|-------|--------|---|--------|--------|------|-------|---------|--------|
| Read  | BTSERR | 0 | DMAERR | BTOERR | DFN8 | CRC16 | CRC5EOF | PIDERR |
| Write | w1c    |   | w1c    | w1c    | w1c  | w1c   | w1c     | w1c    |
| Reset | 0      | 0 | 0      | 0      | 0    | 0     | 0       | 0      |

**USBx\_ERRSTAT field descriptions**

| Field         | Description  |
|---------------|--|
| 7<br>BTSERR   | This bit is set when a bit stuff error is detected. If set, the corresponding packet is rejected due to the error.   |
| 6<br>Reserved | This field is reserved.<br>This read-only field is reserved and always has the value 0.  |
| 5<br>DMAERR   | This bit is set if the USB Module has requested a DMA access to read a new BDT but has not been given the bus before it needs to receive or transmit data. If processing a TX transfer this would cause a transmit data underflow condition. If processing a RX transfer this would cause a receive data overflow condition. This interrupt is useful when developing device arbitration hardware for the microprocessor and the USB module to minimize bus request and bus grant latency. This bit is also set if a data packet to or from the host is larger than the buffer size allocated in the BDT. In this case the data packet is truncated as it is put in buffer memory. |
| 4<br>BTOERR   | This bit is set when a bus turnaround timeout error occurs. The USB module contains a bus turnaround timer that keeps track of the amount of time elapsed between the token and data phases of a SETUP or  |

*Table continues on the next page...*

### USBx\_ERRSTAT field descriptions (continued)

| Field        | Description  |
|--------------|--|
|              | OUT TOKEN or the data and handshake phases of a IN TOKEN. If more than 16 bit times are counted from the previous EOP before a transition from IDLE, a bus turnaround timeout error occurs.  |
| 3<br>DFN8    | This bit is set if the data field received was not 8 bits in length. USB Specification 1.0 requires that data fields be an integral number of bytes. If the data field was not an integral number of bytes, this bit is set.   |
| 2<br>CRC16   | This bit is set when a data packet is rejected due to a CRC16 error.   |
| 1<br>CRC5EOF | <p>This error interrupt has two functions. When the USB Module is operating in peripheral mode (HOSTMODEEN=0), this interrupt detects CRC5 errors in the token packets generated by the host. If set the token packet was rejected due to a CRC5 error.</p> <p>When the USB Module is operating in host mode (HOSTMODEEN=1), this interrupt detects End Of Frame (EOF) error conditions. This occurs when the USB Module is transmitting or receiving data and the SOF counter reaches zero. This interrupt is useful when developing USB packet scheduling software to ensure that no USB transactions cross the start of the next frame.</p> |
| 0<br>PIDERR  | This bit is set when the PID check field fails.  |

### 35.4.12 Error Interrupt Enable register (USBx\_ERREN)

Contains enable bits for each of the error interrupt sources within the USB module. Setting any of these bits enables the respective interrupt source in ERRSTAT. Each bit is set as soon as the error conditions is detected. Therefore, the interrupt does not typically correspond with the end of a token being processed. This register contains the value of 0x00 after a reset.

Address: 4007\_2000h base + 8Ch offset = 4007\_208Ch



### USBx\_ERREN field descriptions

| Field         | Description  |
|---------------|--|
| 7<br>BTSERREN | <p>BTERR Interrupt Enable</p> <p>0 Disables the BTERR interrupt.<br/>1 Enables the BTERR interrupt.</p>    |
| 6<br>Reserved | <p>This field is reserved.</p> <p>This read-only field is reserved and always has the value 0.</p>         |
| 5<br>DMAERREN | <p>DMAERR Interrupt Enable</p> <p>0 Disables the DMAERR interrupt.<br/>1 Enables the DMAERR interrupt.</p> |

*Table continues on the next page...*

**USBx\_ERREN field descriptions (continued)**

| Field          | Description  |
|----------------|--|
| 4<br>BTOERREN  | BTOERR Interrupt Enable<br><br>0 Disables the BTOERR interrupt.<br>1 Enables the BTOERR interrupt.       |
| 3<br>DFN8EN    | DFN8 Interrupt Enable<br><br>0 Disables the DFN8 interrupt.<br>1 Enables the DFN8 interrupt.             |
| 2<br>CRC16EN   | CRC16 Interrupt Enable<br><br>0 Disables the CRC16 interrupt.<br>1 Enables the CRC16 interrupt.          |
| 1<br>CRC5EOFEN | CRC5/EOF Interrupt Enable<br><br>0 Disables the CRC5/EOF interrupt.<br>1 Enables the CRC5/EOF interrupt. |
| 0<br>PIDERREN  | PIDERR Interrupt Enable<br><br>0 Disables the PIDERR interrupt.<br>1 Enters the PIDERR interrupt.        |

**35.4.13 Status register (USBx\_STAT)**

Reports the transaction status within the USB module. When the processor's interrupt controller has received a TOKDNE, interrupt the Status Register must be read to determine the status of the previous endpoint communication. The data in the status register is valid when TOKDNE interrupt is asserted. The Status register is actually a read window into a status FIFO maintained by the USB module. When the USB module uses a BD, it updates the Status register. If another USB transaction is performed before the TOKDNE interrupt is serviced, the USB module stores the status of the next transaction in the STAT FIFO. Thus STAT is actually a four byte FIFO that allows the processor core to process one transaction while the SIE is processing the next transaction. Clearing the TOKDNE bit in the ISTAT register causes the SIE to update STAT with the contents of the next STAT value. If the data in the STAT holding register is valid, the SIE immediately reasserts to TOKDNE interrupt.

Address: 4007\_2000h base + 90h offset = 4007\_2090h

| Bit   | 7 | 6    | 5 | 4 | 3  | 2   | 1 | 0 |
|-------|---|------|---|---|----|-----|---|---|
| Read  |   | ENDP |   |   | TX | ODD |   | 0 |
| Write |   |      |   |   |    |     | 0 | 0 |
| Reset | 0 | 0    | 0 | 0 | 0  | 0   | 0 | 0 |

**USBx\_STAT field descriptions**

| Field           | Description   |
|-----------------|---|
| 7–4<br>ENDP     | This four-bit field encodes the endpoint address that received or transmitted the previous token. This allows the processor core to determine the BDT entry that was updated by the last USB transaction. |
| 3<br>TX         | Transmit Indicator<br>0 The most recent transaction was a receive operation.<br>1 The most recent transaction was a transmit operation.   |
| 2<br>ODD        | This bit is set if the last buffer descriptor updated was in the odd bank of the BDT.   |
| 1–0<br>Reserved | This field is reserved.<br>This read-only field is reserved and always has the value 0.   |

**35.4.14 Control register (USBx\_CTL)**

Provides various control and configuration information for the USB module.

Address: 4007\_2000h base + 94h offset = 4007\_2094h

| Bit   | 7          | 6      | 5                  | 4          |
|-------|------------|--------|--------------------|------------|
| Read  | JSTATE     | SE0    | TXSUSPENDTOKENBUSY | RESET      |
| Write |            |        |                    |            |
| Reset | 0          | 0      | 0                  | 0          |
| Bit   | 3          | 2      | 1                  | 0          |
| Read  | HOSTMODEEN | RESUME | ODDRST             | USBENSOFEN |
| Write |            |        |                    |            |
| Reset | 0          | 0      | 0                  | 0          |

**USBx\_CTL field descriptions**

| Field                   | Description  |
|-------------------------|--|
| 7<br>JSTATE             | Live USB differential receiver JSTATE signal<br>The polarity of this signal is affected by the current state of LSEN .   |
| 6<br>SE0                | Live USB Single Ended Zero signal  |
| 5<br>TXSUSPENDTOKENBUSY | In Host mode, TOKEN_BUSY is set when the USB module is busy executing a USB token. Software must not write more token commands to the Token Register when TOKEN_BUSY is set.. Software should check this field before writing any tokens to the Token Register to ensure that token commands are not lost.<br><br>In Target mode, TXD_SUSPEND is set when the SIE has disabled packet transmission and reception. Clearing this bit allows the SIE to continue token processing. This bit is set by the SIE when a SETUP Token is received allowing software to dequeue any pending packet transactions in the BDT before resuming token processing. |
| 4<br>RESET              | Setting this bit enables the USB Module to generate USB reset signaling. This allows the USB Module to reset USB peripherals. This control signal is only valid in Host mode (HOSTMODEEN=1). Software must set RESET to 1 for the required amount of time and then clear it to 0 to end reset signaling. For more information on reset signaling see Section 7.1.4.3 of the USB specification version 1.0.   |

Table continues on the next page...

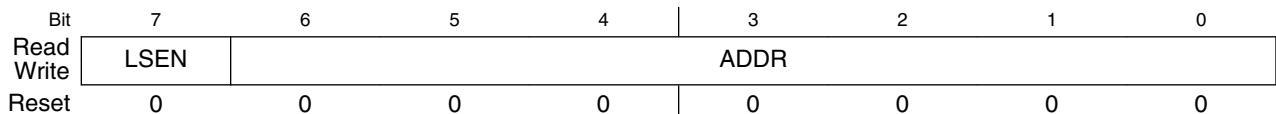
### USBx\_CTL field descriptions (continued)

| Field           | Description  |
|-----------------|--|
| 3<br>HOSTMODEEN | When set to 1, this bit enables the USB Module to operate in Host mode. In host mode, the USB module performs USB transactions under the programmed control of the host processor.   |
| 2<br>RESUME     | When set to 1 this bit enables the USB Module to execute resume signaling. This allows the USB Module to perform remote wake-up. Software must set RESUME to 1 for the required amount of time and then clear it to 0. If the HOSTMODEEN bit is set, the USB module appends a Low Speed End of Packet to the Resume signaling when the RESUME bit is cleared. For more information on RESUME signaling see Section 7.1.4.5 of the USB specification version 1.0. |
| 1<br>ODDRST     | Setting this bit to 1 resets all the BDT ODD ping/pong fields to 0, which then specifies the EVEN BDT bank.  |
| 0<br>USBENSOFEN | <p>USB Enable</p> <p>Setting this bit causes the SIE to reset all of its ODD bits to the BDTs. Therefore, setting this bit resets much of the logic in the SIE. When host mode is enabled, clearing this bit causes the SIE to stop sending SOF tokens.</p> <p>0 Disables the USB Module.<br/>1 Enables the USB Module.</p>  |

### 35.4.15 Address register (USBx\_ADDR)

Holds the unique USB address that the USB module decodes when in Peripheral mode (HOSTMODEEN=0). When operating in Host mode (HOSTMODEEN=1) the USB module transmits this address with a TOKEN packet. This enables the USB module to uniquely address an USB peripheral. In either mode, USB\_EN within the control register must be 1. The Address register is reset to 0x00 after the reset input becomes active or the USB module decodes a USB reset signal. This action initializes the Address register to decode address 0x00 as required by the USB specification.

Address: 4007\_2000h base + 98h offset = 4007\_2098h



### USBx\_ADDR field descriptions

### 35.4.16 BDT Page Register 1 (USBx\_BDTPAGE1)

Provides address bits 15 through 9 of the base address where the current Buffer Descriptor Table (BDT) resides in system memory. The 32-bit BDT Base Address is always aligned on 512-byte boundaries, so bits 8 through 0 of the base address are always zero.

Address: 4007\_2000h base + 9Ch offset = 4007\_209Ch

| Bit   | 7     | 6 | 5 | 4 |  | 3 | 2 | 1 | 0 |
|-------|-------|---|---|---|--|---|---|---|---|
| Read  | BDTBA |   |   |   |  |   |   |   | 0 |
| Write |       |   |   |   |  |   |   |   | 0 |
| Reset | 0     | 0 | 0 | 0 |  | 0 | 0 | 0 | 0 |

#### USBx\_BDTPAGE1 field descriptions

| Field         | Description   |
|---------------|---|
| 7–1<br>BDTBA  | Provides address bits 15 through 9 of the BDT base address.                             |
| 0<br>Reserved | This field is reserved.<br>This read-only field is reserved and always has the value 0. |

### 35.4.17 Frame Number Register Low (USBx\_FRMNUML)

Contains an 11-bit value used to compute the address where the current Buffer Descriptor Table (BDT) resides in system memory.

Address: 4007\_2000h base + A0h offset = 4007\_20A0h

| Bit   | 7        | 6 | 5 | 4 |  | 3 | 2 | 1 | 0 |
|-------|----------|---|---|---|--|---|---|---|---|
| Read  | FRM[7:0] |   |   |   |  |   |   |   | 0 |
| Write |          |   |   |   |  |   |   |   | 0 |
| Reset | 0        | 0 | 0 | 0 |  | 0 | 0 | 0 | 0 |

#### USBx\_FRMNUML field descriptions

| Field           | Description  |
|-----------------|--|
| 7–0<br>FRM[7:0] | This 8-bit field and the 3-bit field in the Frame Number Register High are used to compute the address where the current Buffer Descriptor Table (BDT) resides in system memory. |

### 35.4.18 Frame Number Register High (USBx\_FRMNUMH)

Contains an 11-bit value used to compute the address where the current Buffer Descriptor Table (BDT) resides in system memory.

Address: 4007\_2000h base + A4h offset = 4007\_20A4h

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

**USBx\_FRMNUMH field descriptions**

| Field            | Description   |
|------------------|---|
| 7–3<br>Reserved  | This field is reserved.<br>This read-only field is reserved and always has the value 0.   |
| 2–0<br>FRM[10:8] | This 3-bit field and the 8-bit field in the Frame Number Register Low are used to compute the address where the current Buffer Descriptor Table (BDT) resides in system memory. |

### 35.4.19 Token register (USBx\_TOKEN)

Used to initiate USB transactions when in host mode (HOSTMODEEN=1). When the software needs to execute a USB transaction to a peripheral, it writes the TOKEN type and endpoint to this register. After this register has been written, the USB module begins the specified USB transaction to the address contained in the address register. The processor core must always check that the TOKEN\_BUSY bit in the control register is not 1 before writing to the Token Register. This ensures that the token commands are not overwritten before they can be executed. The address register and endpoint control register 0 are also used when performing a token command and therefore must also be written before the Token Register. The address register is used to select the USB peripheral address transmitted by the token command. The endpoint control register determines the handshake and retry policies used during the transfer.

Address: 4007\_2000h base + A8h offset = 4007\_20A8h

| Bit   | 7 | 6 | 5        | 4 | 3 | 2          | 1 | 0 |
|-------|---|---|----------|---|---|------------|---|---|
| Read  |   |   | TOKENPID |   |   | TOKENENDPT |   |   |
| Write |   |   |          |   |   |            |   |   |
| Reset | 0 | 0 | 0        | 0 | 0 | 0          | 0 | 0 |

**USBx\_TOKEN field descriptions**

| Field           | Description   |
|-----------------|---|
| 7–4<br>TOKENPID | Contains the token type executed by the USB module. |

*Table continues on the next page...*

**USBx\_TOKEN field descriptions (continued)**

| Field             | Description  |
|-------------------|--|
|                   | 0001 OUT Token. USB Module performs an OUT (TX) transaction.<br>1001 IN Token. USB Module performs an In (RX) transaction.<br>1101 SETUP Token. USB Module performs a SETUP (TX) transaction |
| 3–0<br>TOKENENDPT | Holds the Endpoint address for the token command. The four bit value written must be a valid endpoint.   |

**35.4.20 SOF Threshold Register (USBx\_SOFTHLD)**

The SOF Threshold Register is used only in Host mode (HOSTMODEEN=1). When in Host mode, the 14-bit SOF counter counts the interval between SOF frames. The SOF must be transmitted every 1ms so therefore the SOF counter is loaded with a value of 12000. When the SOF counter reaches zero, a Start Of Frame (SOF) token is transmitted.

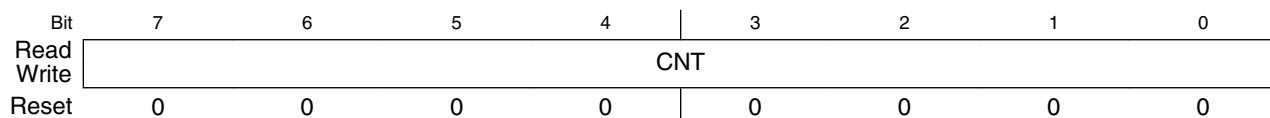
The SOF threshold register is used to program the number of USB byte times before the SOF to stop initiating token packet transactions. This register must be set to a value that ensures that other packets are not actively being transmitted when the SOF time counts to zero. When the SOF counter reaches the threshold value, no more tokens are transmitted until after the SOF has been transmitted.

The value programmed into the threshold register must reserve enough time to ensure the worst case transaction completes. In general the worst case transaction is an IN token followed by a data packet from the target followed by the response from the host. The actual time required is a function of the maximum packet size on the bus.

Typical values for the SOF threshold are:

- 64-byte packets=74;
- 32-byte packets=42;
- 16-byte packets=26;
- 8-byte packets=18.

Address: 4007\_2000h base + ACh offset = 4007\_20ACh

**USBx\_SOFTHLD field descriptions**

| Field      | Description                                       |
|------------|---|
| 7–0<br>CNT | Represents the SOF count threshold in byte times. |

### 35.4.21 BDT Page Register 2 (USBx\_BDTPAGE2)

Contains an 8-bit value used to compute the address where the current Buffer Descriptor Table (BDT) resides in system memory.

Address: 4007\_2000h base + B0h offset = 4007\_20B0h

| Bit   | 7 | 6 | 5 | 4 |       | 3 | 2 | 1 | 0 |
|-------|---|---|---|---|-------|---|---|---|---|
| Read  | 0 | 0 | 0 | 0 |       | 0 | 0 | 0 | 0 |
| Write |   |   |   |   | BDTBA |   |   |   |   |

Reset

## Memory map/Register definitions

In Host mode ENDPT0 is used to determine the handshake, retry and low speed characteristics of the host transfer. For Control, Bulk and Interrupt transfers, the EPHSHK bit should be 1. For Isochronous transfers it should be 0. Common values to use for ENDPT0 in host mode are 0x4D for Control, Bulk, and Interrupt transfers, and 0x4C for Isochronous transfers.

Address: 4007\_2000h base + C0h offset + (4d × i), where i=0d to 15d

| Bit   | 7             | 6        | 5 | 4        | 3      | 2      | 1       | 0      |
|-------|---------------|----------|---|----------|--------|--------|---------|--------|
| Read  | HOSTWOH<br>UB | RETRYDIS | 0 | EPCTLDIS | EPRXEN | EPTXEN | EPSTALL | EPHSHK |
| Write | 0             | 0        | 0 | 0        | 0      | 0      | 0       | 0      |
| Reset | 0             | 0        | 0 | 0        | 0      | 0      | 0       | 0      |

### USBx-ENDPTn field descriptions

| Field       | Description   |
|-------------|---|
| 7 HOSTWOHUB | This is a Host mode only field and is present in the control register for endpoint 0 (ENDPT0) only. When set this bit allows the host to communicate to a directly connected low speed device. When cleared, the host produces the PRE_PID. It then switches to low-speed signaling when sends a token to a low speed device as required to communicate with a low speed device through a hub.  |
| 6 RETRYDIS  | This is a Host mode only bit and is present in the control register for endpoint 0 (ENDPT0) only. When set this bit causes the host to not retry NAK'ed (Negative Acknowledgement) transactions. When a transaction is NAKed, the BDT PID field is updated with the NAK PID, and the TOKEN_DNE interrupt is set. When this bit is cleared NAKed transactions are retried in hardware. This bit must be set when the host is attempting to poll an interrupt endpoint. |
| 5 Reserved  | This field is reserved.<br>This read-only field is reserved and always has the value 0.   |
| 4 EPCTLDIS  | This bit, when set, disables control (SETUP) transfers. When cleared, control transfers are enabled. This applies if and only if the EPRXEN and EPTXEN bits are also set.   |
| 3 EPRXEN    | This bit, when set, enables the endpoint for RX transfers.  |
| 2 EPTXEN    | This bit, when set, enables the endpoint for TX transfers.  |
| 1 EPSTALL   | When set this bit indicates that the endpoint is stalled. This bit has priority over all other control bits in the EndPoint Enable Register, but it is only valid if EPTXEN=1 or EPRXEN=1. Any access to this endpoint causes the USB Module to return a STALL handshake. After an endpoint is stalled it requires intervention from the Host Controller.   |
| 0 EPHSHK    | When set this bit enables an endpoint to perform handshaking during a transaction to this endpoint. This bit is generally 1 unless the endpoint is Isochronous.   |

## 35.4.24 USB Control register (USBx\_USBCTRL)

Address: 4007\_2000h base + 100h offset = 4007\_2100h

| Bit   | 7    | 6   | 5 | 4 | 3 | 2 | 1 | 0 |
|-------|------|-----|---|---|---|---|---|---|
| Read  | SUSP | PDE |   |   | 0 |   |   |   |
| Write | 1    | 1   | 0 | 0 | 0 | 0 | 0 | 0 |
| Reset | 1    | 1   | 0 | 0 | 0 | 0 | 0 | 0 |

**USBx\_USBCTRL field descriptions**

| Field           | Description   |
|-----------------|---|
| 7<br>SUSP       | Places the USB transceiver into the suspend state.<br><br>0 USB transceiver is not in suspend state.<br>1 USB transceiver is in suspend state.      |
| 6<br>PDE        | Enables the weak pulldowns on the USB transceiver.<br><br>0 Weak pulldowns are disabled on D+ and D-.<br>1 Weak pulldowns are enabled on D+ and D-. |
| 5–0<br>Reserved | This field is reserved.<br>This read-only field is reserved and always has the value 0.   |

**35.4.25 USB OTG Observe register (USBx\_OBSERVE)**

Provides visibility on the state of the pull-ups and pull-downs at the transceiver. Useful when interfacing to an external OTG control module via a serial interface.

Address: 4007\_2000h base + 104h offset = 4007\_2104h

| Bit   | 7    | 6    | 5 | 4    | 3 | 2 | 1 | 0 |
|-------|------|------|---|------|---|---|---|---|
| Read  | DPPU | DPPD | 0 | DMPD | 0 | 0 | 0 | 0 |
| Write |      |      |   |      |   |   |   |   |
| Reset | 0    | 1    | 0 | 1    | 0 | 0 | 0 | 0 |

**USBx\_OBSERVE field descriptions**

| Field           | Description  |
|-----------------|--|
| 7<br>DPPU       | Provides observability of the D+ Pullup . signal output from the USB OTG module<br><br>0 D+ pullup disabled.<br>1 D+ pullup enabled.       |
| 6<br>DPPD       | Provides observability of the D+ Pulldown . signal output from the USB OTG module<br><br>0 D+ pulldown disabled.<br>1 D+ pulldown enabled. |
| 5<br>Reserved   | This field is reserved.<br>This read-only field is reserved and always has the value 0.  |
| 4<br>DMPD       | Provides observability of the D– Pulldown . signal output from the USB OTG module<br><br>0 D– pulldown disabled.<br>1 D– pulldown enabled. |
| 3–1<br>Reserved | This field is reserved.<br>This read-only field is reserved and always has the value 0.  |
| 0<br>Reserved   | This field is reserved.<br>This read-only field is reserved and always has the value 0.  |

### 35.4.26 USB OTG Control register (USBx\_CONTROL)

Address: 4007\_2000h base + 108h offset = 4007\_2108h

| Bit   | 7 | 6 | 5 | 4              |
|-------|---|---|---|----------------|
| Read  |   | 0 |   | DPPULLUPNONOTG |
| Write |   |   |   |                |
| Reset | 0 | 0 | 0 | 0              |
| Bit   | 3 | 2 | 1 | 0              |
| Read  |   | 0 |   |                |
| Write |   |   |   |                |
| Reset | 0 | 0 | 0 | 0              |

#### USBx\_CONTROL field descriptions

| Field               | Description   |
|---------------------|---|
| 7–5<br>Reserved     | This field is reserved.<br>This read-only field is reserved and always has the value 0.   |
| 4<br>DPPULLUPNONOTG | Provides control of the DP Pullup in the USB OTG module, if USB is configured in non-OTG device mode.<br><br>0 DP Pullup in non-OTG device mode is not enabled.<br>1 DP Pullup in non-OTG device mode is enabled. |
| 3–0<br>Reserved     | This field is reserved.<br>This read-only field is reserved and always has the value 0.   |

### 35.4.27 USB Transceiver Control Register 0 (USBx\_USBTRC0)

Address: 4007\_2000h base + 10Ch offset = 4007\_210Ch

| Bit   | 7        | 6 | 5         | 4              |
|-------|----------|---|-----------|----------------|
| Read  |          |   | USBRESMEN | 0              |
| Write | USBRESET | 1 |           |                |
| Reset | 0        | 0 | 0         | 0              |
| Bit   | 3        | 2 | 1         | 0              |
| Read  |          | 0 | SYNC_DET  | USB_RESUME_INT |
| Write |          |   |           |                |
| Reset | 0        | 0 | 0         | 0              |

#### USBx\_USBTRC0 field descriptions

| Field         | Description   |
|---------------|---|
| 7<br>USBRESET | USB Reset<br><br>Generates a hard reset to the USB_OTG module. After this bit is set and the reset occurs, this bit is automatically cleared. |

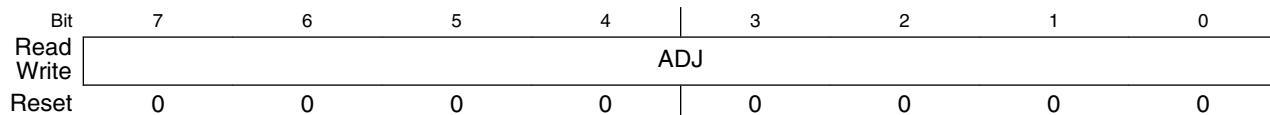
Table continues on the next page...

**USBx\_USBTRC0 field descriptions (continued)**

| Field               | Description  |
|---------------------|--|
|                     | <b>NOTE:</b> This bit is always read as zero. Wait two USB clock cycles after setting this bit.<br>0 Normal USB module operation.<br>1 Returns the USB module to its reset state.  |
| 6<br>Reserved       | This field is reserved.<br><br><b>NOTE:</b> Software must set this bit to 1b.  |
| 5<br>USBRESMEN      | Asynchronous Resume Interrupt Enable<br><br>This bit, when set, allows the USB module to send an asynchronous wakeup event to the MCU upon detection of resume signaling on the USB bus. The MCU then re-enables clocks to the USB module. It is used for low-power suspend mode when USB module clocks are stopped or the USB transceiver is in Suspend mode. Async wakeup only works in device mode.<br><br>0 USB asynchronous wakeup from suspend mode disabled.<br>1 USB asynchronous wakeup from suspend mode enabled. The asynchronous resume interrupt differs from the synchronous resume interrupt in that it asynchronously detects K-state using the unfiltered state of the D+ and D- pins. This interrupt should only be enabled when the Transceiver is suspended. |
| 4–2<br>Reserved     | This field is reserved.<br>This read-only field is reserved and always has the value 0.  |
| 1<br>SYNC_DET       | Synchronous USB Interrupt Detect<br><br>0 Synchronous interrupt has not been detected.<br>1 Synchronous interrupt has been detected.   |
| 0<br>USB_RESUME_INT | USB Asynchronous Interrupt<br><br>0 No interrupt was generated.<br>1 Interrupt was generated because of the USB asynchronous interrupt.  |

**35.4.28 Frame Adjust Register (USBx\_USBFRMADJUST)**

Address: 4007\_2000h base + 114h offset = 4007\_2114h

**USBx\_USBFRMADJUST field descriptions**

| Field      | Description   |
|------------|---|
| 7–0<br>ADJ | Frame Adjustment<br><br>In Host mode, the frame adjustment is a twos complement number that adjusts the period of each USB frame in 12-MHz clock periods. A SOF is normally generated every 12,000 12-MHz clock cycles. The Frame Adjust Register can adjust this by -128 to +127 to compensate for inaccuracies in the USB 48-MHz clock. Changes to the ADJ bit take effect at the next start of the next frame. |

## 35.5 OTG and Host mode operation

The Host mode logic allows devices such as digital cameras and palmtop computers to function as a USB Host Controller. The OTG logic adds an interface to allow the OTG Host Negotiation and Session Request Protocols (HNP and SRP) to be implemented in software. Host Mode allows a peripheral such as a digital camera to be connected directly to a USB compliant printer. Digital photos can then be easily printed without having to upload them to a PC. In the palmtop computer application, a USB compliant keyboard/mouse can be connected to the palmtop computer with the obvious advantages of easier interaction.

Host mode is intended for use in handheld-portable devices to allow easy connection to simple HID class devices such as printers and keyboards. It is not intended to perform the functions of a full OHCI or UHCI compatible host controller found on PC motherboards. The USB-FS is not supported by Windows 98 as a USB host controller. Host mode allows bulk, isochronous, interrupt and control transfers. Bulk data transfers are performed at nearly the full USB interface bandwidth. Support is provided for ISO transfers, but the number of ISO streams that can be practically supported is affected by the interrupt latency of the processor servicing the token during interrupts from the SIE. Custom drivers must be written to support Host mode operation.

Setting the HOST\_MODE\_EN bit in the CTL register enables Host mode. The USB-FS core can only operate as a peripheral device or in Host mode. It cannot operate in both modes simultaneously. When HOST\_MODE is enabled, only endpoint zero is used. All other endpoints should be disabled by software.

## 35.6 Host Mode Operation Examples

The following sections illustrate the steps required to perform USB host functions using the USB-FS core. While it is useful to understand the interaction of the hardware and the software at a detailed level, an understanding of the interactions at this level is not required to write host applications using the API software.

To enable host mode and discover a connected device:

1. Enable Host Mode (CTL[HOST\_MODE\_EN]=1). The pull-down resistors are enabled, and pull-up disabled. Start of Frame (SOF) generation begins. SOF counter loaded with 12,000. Disable SOF packet generation to eliminate noise on the USB by writing the USB enable bit to 0 (CTL[USB\_EN]=0).

2. Enable the ATTACH interrupt (INT\_ENB[ATTACH]=1).
3. Wait for ATTACH interrupt (INT\_STAT[ATTACH]). Signaled by USB Target pull-up resistor changing the state of DPLUS or DMINUS from 0 to 1 (SE0 to J or K state).
4. Check the state of the JSTATE and SE0 bits in the control register. If the connecting device is low speed (JSTATE bit is 0), set the low-speed bit in the address registers (ADDR[LS\_EN]=1) and the Host Without Hub bit in endpoint 0 register control (ENDPT0[HOSTTWOHUB]=1).
5. Enable RESET (CTL[RESET]=1) for 10 ms.
6. Enable SOF packet to keep the connected device from going to suspend (CTL[USB\_EN]=1).
7. Start enumeration by sending a sequence of device framework commands, device frame work packets to the default control pipe of the connected device. See the *Universal Serial Bus Revision 2.0 specification*, "Chapter 9 USB Device Framework" (<http://www.usb.org/developers/docs>).

To complete a control transaction to a connected device:

1. Complete all the steps to discover a connected device
2. Set up the endpoint control register for bidirectional control transfers ENDPT0[4:0] = 0x0d.
3. Place a copy of the device framework setup command in a memory buffer. See the *Universal Serial Bus Revision 2.0 specification*, "Chapter 9 USB Device Framework" (<http://www.usb.org/developers/docs>).
4. Initialize current even or odd TX EP0 BDT to transfer the 8 bytes of command data for a device framework command (for example, a GET DEVICE DESCRIPTOR).
  - Set the BDT command word to 0x00080080 –Byte count to 8, OWN bit to 1.
  - Set the BDT buffer address field to the start address of the 8 byte command buffer.
5. Set the USB device address of the target device in the address register (ADDR[6:0]). After the USB bus reset, the device USB address is zero. It is set to some other value usually 1 by the Set Address device framework command.
6. Write the token register with a SETUP to Endpoint 0 the target device default control pipe (TOKEN=0xD0). This initiates a setup token on the bus followed by a data packet. The device handshake is returned in the BDT PID field after the packets

complete. When the BDT is written, a token done (ISTAT[TOKDNE]) interrupt is asserted. This completes the setup phase of the setup transaction. See the *Universal Serial Bus Revision 2.0 specification*, "Chapter 9 USB Device Framework" (<http://www.usb.org/developers/docs>).

7. To initiate the data phase of the setup transaction (that is, get the data for the GET DEVICE DESCRIPTOR command), set up a buffer in memory for the data to be transferred.
8. Initialize the current even or odd TX EP0 BDT to transfer the data.
  - Set the BDT command word to 0x004000C0 – BC to 64 (the byte count of the data buffer in this case), OWN bit to 1, Data toggle to Data1.
  - Set the BDT buffer address field to the start address of the data buffer
9. Write the token register with a IN or OUT token to Endpoint 0 the target device default control pipe, an IN token for a GET DEVICE DESCRIPTOR command (TOKEN=0x90). This initiates an IN token on the bus followed by a data packet from the device to the host. When the data packet completes, the BDT is written and a token done (ISTAT[DNE]) interrupt is asserted. For control transfers with a single packet data phase this completes the data phase of the setup transaction. See the *Universal Serial Bus Revision 2.0 specification*, "Chapter 9 USB Device Framework" (<http://www.usb.org/developers/docs>).
10. To initiate the status phase of the setup transaction, set up a buffer in memory to receive or send the zero length status phase data packet.
11. Initialize the current even or odd TX EP0 BDT to transfer the status data.
  - Set the BDT command word to 0x00000080 – BC to 0 (the byte count of the data buffer in this case), OWN bit to 1, Data toggle to Data1.
  - Set the BDT buffer address field to the start address of the data buffer
12. Write the token register with a IN or OUT token to Endpoint 0 the target device default control pipe, an OUT token for a GET DEVICE DESCRIPTOR command (TOKEN=0x10). This initiates an OUT token on the bus followed by a zero length data packet from the host to the device. When the data packet completes, the BDT is written with the handshake from the device and a Token Done (ISTAT[TOKDNE]) interrupt is asserted. This completes the data phase of the setup transaction. See the *Universal Serial Bus Revision 2.0 specification*, "Chapter 9 USB Device Framework" (<http://www.usb.org/developers/docs>).

To send a full speed bulk data transfer to a target device:

1. Complete all steps to discover a connected device and to configure a connected device. Write the ADDR register with the address of the target device. Typically, there is only one other device on the USB bus in host mode so it is expected that the address is 0x01 and should remain constant.
2. Write 0x1D to ENDPT0 register to enable transmit and receive transfers with handshaking enabled.
3. Setup the even TX EP0 BDT to transfer up to 64 bytes.
4. Set the USB device address of the target device in the address register (ADDR[6:0]).
5. Write the TOKEN register with an OUT token to the desired endpoint. The write to this register triggers the USB-FS transmit state machines to begin transmitting the TOKEN and the data.
6. Setup the odd TX EP0 BDT to transfer up to 64 bytes.
7. Write the TOKEN register with an OUT token as in step 4. Two tokens can be queued at a time to allow the packets to be double buffered to achieve maximum throughput.
8. Wait for the TOKDNE interrupt. This indicates that one of the BDTs has been released back to the processor and the transfer has completed. If the target device asserts NAKs, the USB-FS continues to retry the transfer indefinitely without processor intervention unless the ENDPT0[RETRYDIS] is 1. If the retry disable field is set, the handshake (ACK, NAK, STALL, or ERROR (0xf)) is returned in the BDT PID field. If a stall interrupt occurs, the pending packet must be dequeued and the error condition in the target device cleared. If a Reset interrupt occurs (SE0 for more than 2.5  $\mu$ s), the target has detached.
9. After the TOK\_DNE interrupt occurs, the BDTs can be examined and the next data packet queued by returning to step 2.

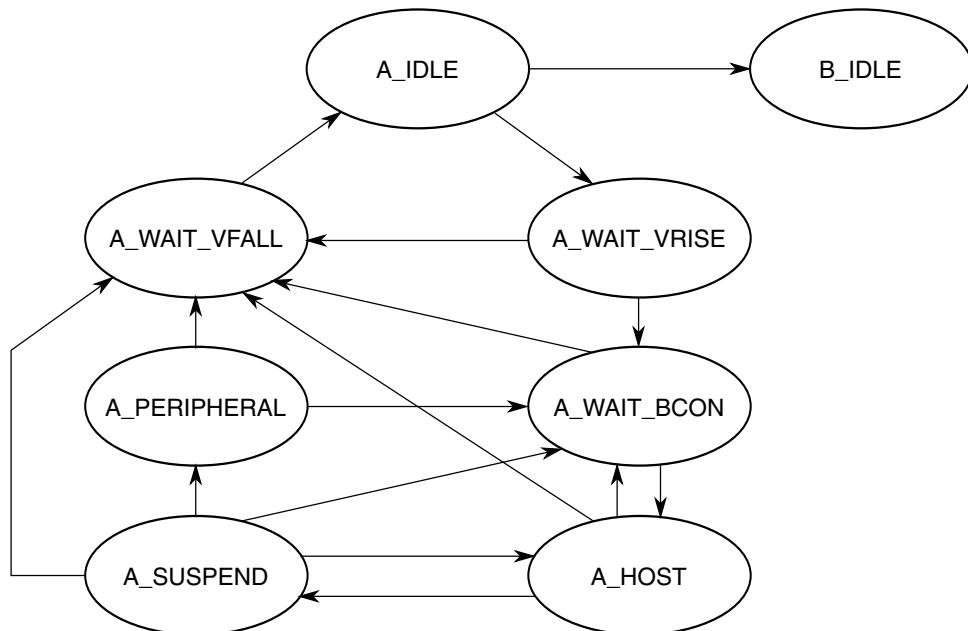
## 35.7 On-The-Go operation

The USB-OTG core provides sensors and controls to enable On-The-Go (OTG) operation. These sensors are used by the OTG API software to implement the Host Negotiation Protocol (HNP) and Session Request Protocol (SRP). API calls are provided to give access the OTG protocol control signals, and include the OTG capabilities in the device application. The following state machines show the OTG operations involved with HNP and SRP protocols from either end of the USB cable.

### 35.7.1 OTG dual role A device operation

A device is considered the A device because of the type of cable attached. If the USB Type A connector or the USB Type Mini A connector is plugged into the device, it is considered the A device.

A dual role A device operates as the following flow diagram and state description table illustrates.



**Figure 35-93. Dual role A device flow diagram**

**Table 35-96. State descriptions for the dual role A device flow**

| State        | Action   | Response                                |
|--------------|--|---|
| A_IDLE       | If ID Interrupt.<br>The cable has been un-plugged or a Type B cable has been attached. The device now acts as a Type B device.   | Go to B_IDLE                            |
|              | If the A application wants to use the bus or if the B device is doing an SRP as indicated by an A_SESS_VLD Interrupt or Attach or Port Status Change Interrupt check data line for 5 –10 msec pulsing. | Go to A_WAIT_VRISE<br>Turn on DRV_VBUS  |
| A_WAIT_VRISE | If ID Interrupt or if A_VBUS_VLD is false after 100 msec<br>The cable has been changed or the A device cannot support the current required from the B device.  | Go to A_WAIT_VFALL<br>Turn off DRV_VBUS |
|              | If A_VBUS_VLD interrupt  | Go to A_WAIT_BCON                       |
|              |  |   |

*Table continues on the next page...*

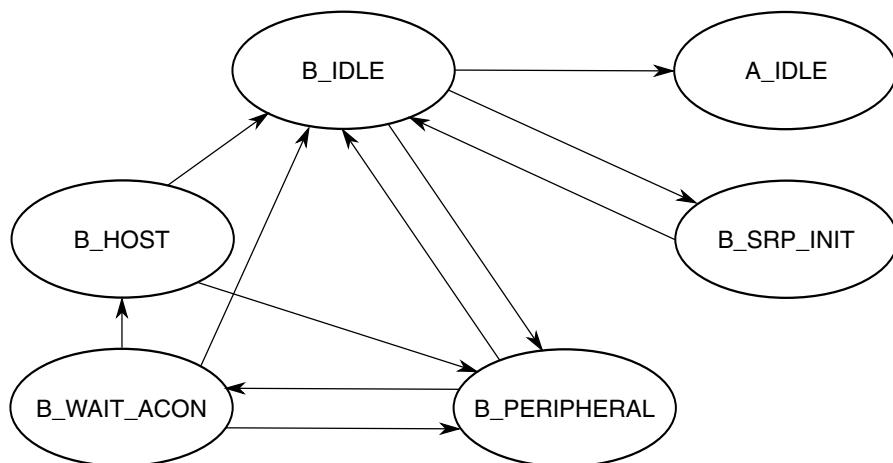
**Table 35-96. State descriptions for the dual role A device flow (continued)**

| State        | Action   | Response  |
|--------------|--|---|
| A_WAIT_BCON  | After 200 ms without Attach or ID Interrupt. (This could wait forever if desired.)                                       | Go to A_WAIT_FALL<br>Turn off DRV_VBUS                        |
|              | A_VBUS_VLD Interrupt and B device attaches   | Go to A_HOST<br>Turn on Host mode                             |
| A_HOST       | Enumerate Device determine OTG Support.  |   |
|              | If A_VBUS_VLD/ Interrupt or A device is done and doesn't think he wants to do something soon or the B device disconnects | Go to A_WAIT_VFALL<br>Turn off Host mode<br>Turn off DRV_VBUS |
|              | If the A device is finished with session or if the A device wants to allow the B device to take bus.                     | Go to A_SUSPEND   |
|              | ID Interrupt or the B device disconnects   | Go to A_WAIT_BCON   |
| A_SUSPEND    | If ID Interrupt, or if 150 ms B disconnect timeout (This timeout value could be longer) or if A_VBUS_VLD\ Interrupt      | Go to A_WAIT_VFALL<br>Turn off DRV_VBUS                       |
|              | If HNP enabled, and B disconnects in 150 ms then B device is becoming the host.  | Go to A_PERIPHERAL<br>Turn off Host mode                      |
|              | If A wants to start another session  | Go to A_HOST  |
| A_PERIPHERAL | If ID Interrupt or if A_VBUS_VLD interrupt   | Go to A_WAIT_VFALL<br>Turn off DRV_VBUS.                      |
|              | If 3 –200 ms of Bus Idle   | Go to A_WAIT_BCON<br>Turn on Host mode                        |
| A_WAIT_VFALL | If ID Interrupt or (A_SESS_VLD/ & b_conn/)   | Go to A_IDLE  |

### 35.7.2 OTG dual role B device operation

A device is considered a B device if it connected to the bus with a USB Type B cable or a USB Type Mini B cable.

A dual role B device operates as the following flow diagram and state description table illustrates.

**Figure 35-94. Dual role B device flow diagram****Table 35-97. State descriptions for the dual role B device flow**

| State        | Action  | Response  |
|--------------|---|---|
| B_IDLE       | If ID\ Interrupt.<br>A Type A cable has been plugged in and the device should now respond as a Type A device.<br><br>If B_SESS_VLD Interrupt.<br>The A device has turned on VBUS and begins a session.<br><br>If B application wants the bus and Bus is Idle for 2 ms and the B_SESS_END bit is set, the B device can perform an SRP. | Go to A_IDLE<br><br>Go to B_PERIPHERAL<br>Turn on DP_HIGH<br><br>Go to B_SRPI<br>Pulse CHRG_VBUS<br>Pulse DP_HIGH 5-10 ms |
| B_SRPI       | If ID\ Interrupt or SRP Done (SRP must be done in less than 100 ms.)  | Go to B_IDLE  |
| B_PERIPHERAL | If HNP enabled and the bus is suspended and B wants the bus, the B device can become the host.  | Go to B_WAIT_ACON<br>Turn off DP_HIGH   |
| B_WAIT_ACON  | If A connects, an attach interrupt is received<br><br>If ID\ Interrupt or B_SESS_VLD\ Interrupt<br>If the cable changes or if VBUS goes away, the host doesn't support us.<br>Go to B_IDLE<br><br>If 3.125 ms expires or if a Resume occurs   | Go to B_HOST<br>Turn on Host Mode<br><br>Go to B_IDLE<br><br>Go to B_PERIPHERAL   |
| B_HOST       | If ID\ Interrupt or B_SESS_VLD\ Interrupt<br>If the cable changes or if VBUS goes away, the host doesn't support us.<br><br>If B application is done or A disconnects   | Go to B_IDLE<br><br>Go to B_PERIPHERAL  |

# Chapter 36

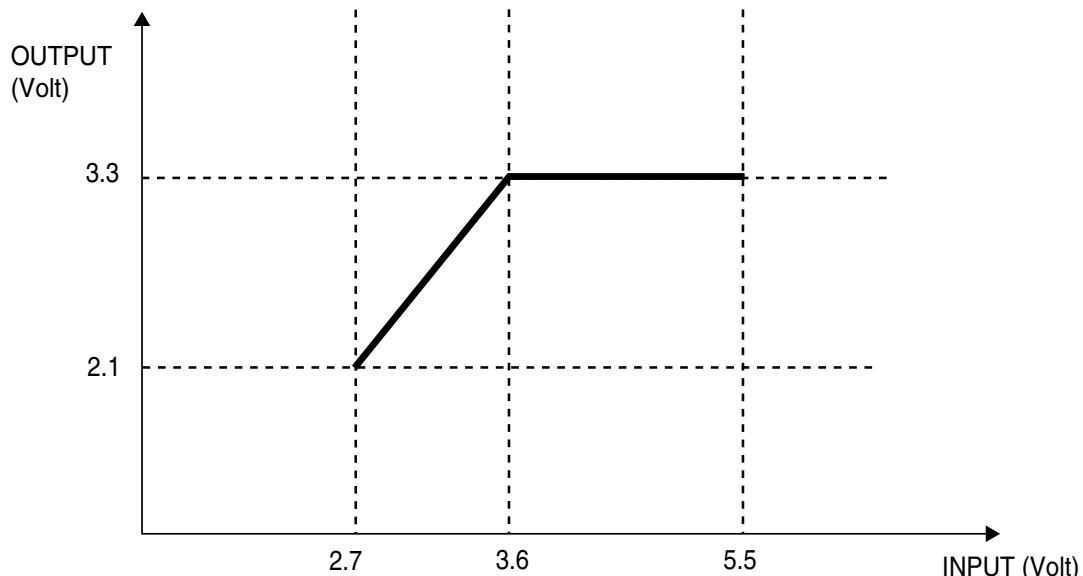
## USB Voltage Regulator

### 36.1 Introduction

#### NOTE

For the chip-specific implementation details of this module's instances see the chip configuration information.

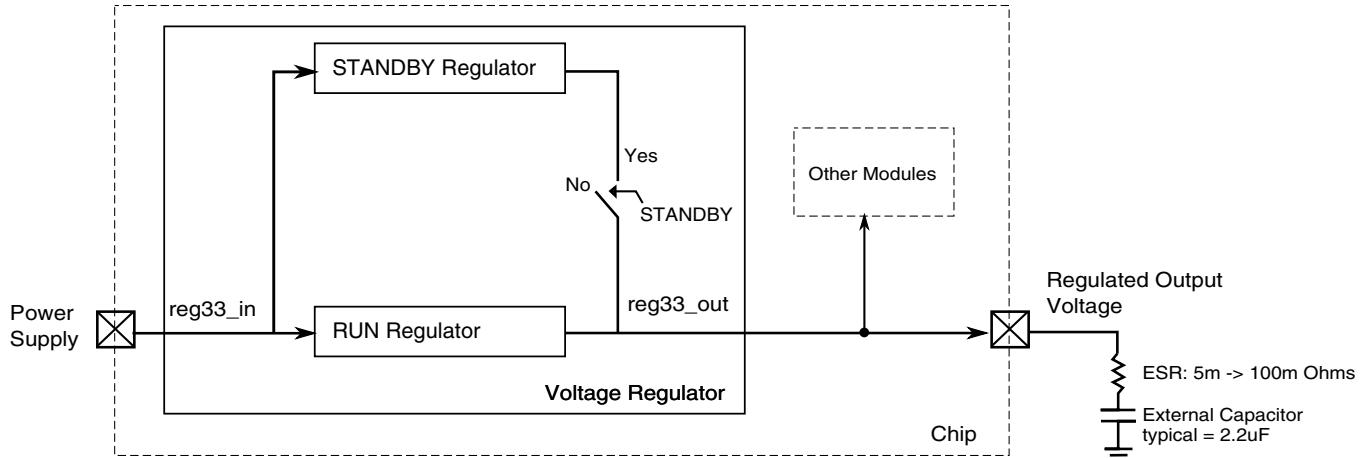
The USB Voltage Regulator module is a LDO linear voltage regulator to provide 3.3V power from an input power supply varying from 2.7 V to 5.5 V. It consists of one 3.3 V power channel. When the input power supply is below 3.6 V, the regulator goes to pass-through mode. The following figure shows the ideal relation between the regulator output and input power supply.



**Figure 36-1. Ideal Relation Between the Regulator Output and Input Power Supply**

### 36.1.1 Overview

A simplified block diagram for the USB Voltage Regulator module is shown below.



**Figure 36-2. USB Voltage Regulator Block Diagram**

This module uses 2 regulators in parallel. In run mode, the RUN regulator with the bandgap voltage reference is enabled and can provide up to 120 mA load current. In run mode, the STANDBY regulator and the low power reference are also enabled, but a switch disconnects its output from the external pin. In STANDBY mode, the RUN regulator is disabled and the STANDBY regulator output is connected to the external pin. Internal power mode signals control whether the module is in RUN or STANDBY mode.

### 36.1.2 Features

- Low drop-out linear voltage regulator with one power channel (3.3V).
- Low drop-out voltage: 300 mV.
- Output current: 120 mA.
- Three different power modes: RUN, STANDBY and SHUTDOWN.
- Low quiescent current in RUN mode.
  - Typical value is around 120 uA (one thousand times smaller than the maximum load current).
- Very low quiescent current in STANDBY mode.
  - Typical value is around 1 uA.
- Automatic current limiting if the load current is greater than 290 mA.

- Automatic power-up once some voltage is applied to the regulator input.
- Pass-through mode for regulator input voltages less than 3.6 V
- Small output capacitor: 2.2 uF
- Stable with aluminum, tantalum or ceramic capacitors.

### 36.1.3 Modes of Operation

The regulator has these power modes:

- RUN—The regulating loop of the RUN regulator and the STANDBY regulator are active, but the switch connecting the STANDBY regulator output to the external pin is open.
- STANDBY—The regulating loop of the RUN regulator is disabled and the standby regulator is active. The switch connecting the STANDBY regulator output to the external pin is closed.
- SHUTDOWN—The module is disabled.

The regulator is enabled by default. This means that once the power supply is provided, the module power-up sequence to RUN mode starts.

## 36.2 USB Voltage Regulator Module Signal Descriptions

The following table shows the external signals for the regulator.

**Table 36-1. USB Voltage Regulator Module Signal Descriptions**

| Signal    | Description              | I/O |
|-----------|--------------------------|-----|
| reg33_in  | Unregulated power supply | I   |
| reg33_out | Regulator output voltage | O   |



# Chapter 37

## Serial Peripheral Interface (SPI)

### 37.1 Introduction

#### NOTE

For the chip-specific implementation details of this module's instances see the chip configuration information.

The serial peripheral interface (SPI) module provides for full-duplex, synchronous, serial communication between the MCU and peripheral devices. These peripheral devices can include other microcontrollers, analog-to-digital converters, shift registers, sensors, and memories, among others.

The SPI runs at a baud rate up to the bus clock divided by two in master mode and up to the bus clock divided by four in slave mode. Software can poll the status flags, or SPI operation can be interrupt driven.

#### NOTE

For the actual maximum SPI baud rate, refer to the Chip Configuration details and to the device's Data Sheet.

The SPI also includes a hardware match feature for the receive data buffer.

The SPI includes an internal DMA interface to support continuous SPI transmission through an on-chip DMA controller instead of through the CPU. This feature decreases CPU loading, allowing CPU time to be used for other work.

#### 37.1.1 Features

The SPI includes these distinctive features:

- Master mode or slave mode operation
- Full-duplex or single-wire bidirectional mode

- Programmable transmit bit rate
- Double-buffered transmit and receive data register
- Serial clock phase and polarity options
- Slave select output
- Mode fault error flag with CPU interrupt capability
- Control of SPI operation during wait mode
- Selectable MSB-first or LSB-first shifting
- Receive data buffer hardware match feature
- Support transmission of both Transmit and Receive by DMA

### **37.1.2 Modes of Operation**

The SPI functions in three modes, run, wait, and stop.

- Run Mode

This is the basic mode of operation.

- Wait Mode

SPI operation in wait mode is a configurable low power mode, controlled by the SPISWAI bit located in the SPIx\_C2 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 enters run mode. If the SPI is configured as a slave, reception and transmission of a byte continues, so that the slave stays synchronized to the master.

- Stop Mode

To reduce power consumption, the SPI is inactive in stop modes where the peripheral bus clock is stopped but internal logic states are retained. If the SPI is configured as a master, any transmission in progress stops, but is resumed after the CPU enters run mode. If the SPI is configured as a slave, reception and transmission of a data continues, so that the slave stays synchronized to the master.

The SPI is completely disabled in stop modes where the peripheral bus clock is stopped and internal logic states are not retained. When the CPU wakes from these stop modes, all SPI register content is reset.

Detailed descriptions of operating modes appear in [Low Power Mode Options](#).

### 37.1.3 Block Diagrams

This section includes block diagrams showing SPI system connections, the internal organization of the SPI module, and the SPI clock dividers that control the master mode bit rate.

#### 37.1.3.1 SPI System Block Diagram

The following figure shows the SPI modules of two MCUs connected in a master-slave arrangement. The master device initiates all SPI data transfers. During a transfer, the master shifts data out (on the MOSI pin) to the slave while simultaneously shifting data in (on the MISO pin) from the slave. The transfer effectively exchanges the data that was in the SPI shift registers of the two SPI systems. The SPSCK signal is a clock output from the master and an input to the slave. The slave device must be selected by a low level on the slave select input (SS pin). In this system, the master device has configured its SS pin as an optional slave select output.

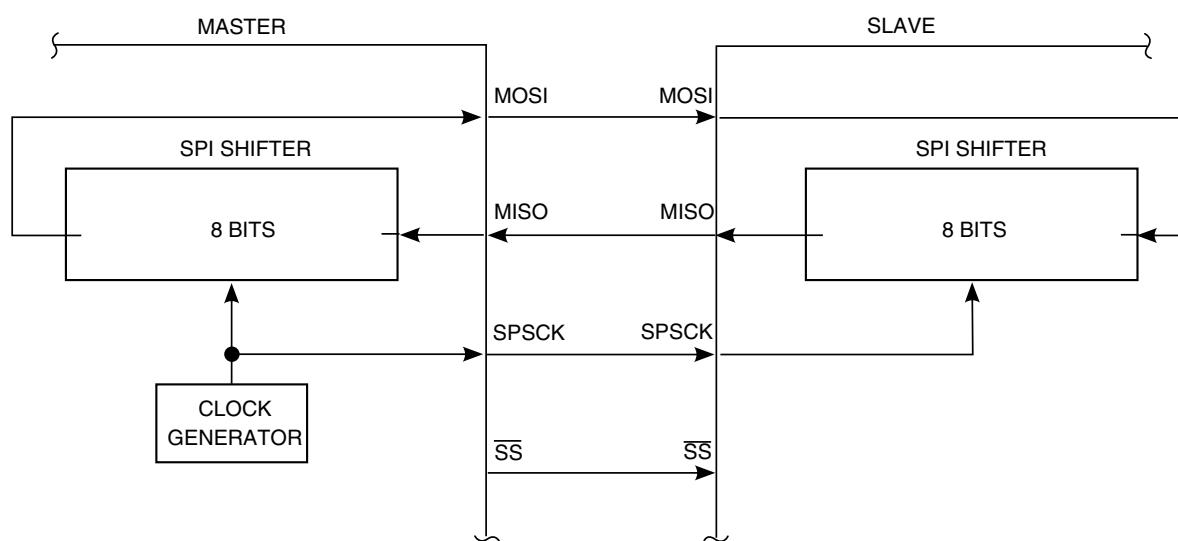


Figure 37-1. SPI System Connections

### 37.1.3.2 SPI Module Block Diagram

The following is a block diagram of the SPI module. The central element of the SPI is the SPI shift register. Data is written to the double-buffered transmitter (write to SPIx\_D) and gets transferred to the SPI shift register at the start of a data transfer. After shifting in 8 bits of data, the data is transferred into the double-buffered receiver where it can be read from SPIx\_D. Pin multiplexing logic controls connections between MCU pins and the SPI module.

When the SPI is configured as a master, the clock output is routed to the SPSCK pin, the shifter output is routed to MOSI, and the shifter input is routed from the MISO pin.

When the SPI is configured as a slave, the SPSCK pin is routed to the clock input of the SPI, the shifter output is routed to MISO, and the shifter input is routed from the MOSI pin.

In the external SPI system, simply connect all SPSCK pins to each other, all MISO pins together, and all MOSI pins together. Peripheral devices often use slightly different names for these pins.

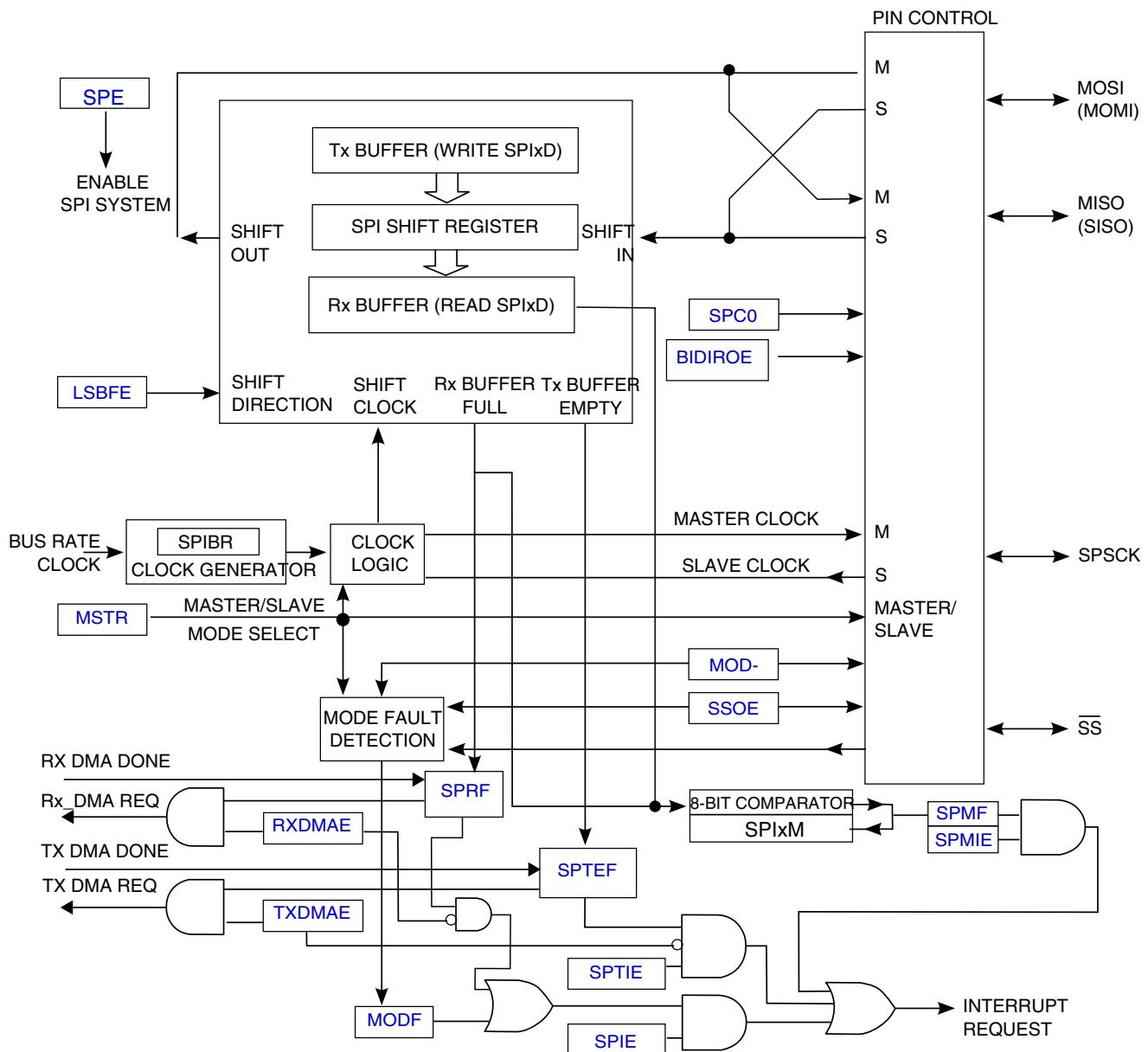


Figure 37-2. SPI Module Block Diagram without FIFO

## 37.2 External Signal Description

The SPI optionally shares four port pins. The function of these pins depends on the settings of SPI control bits. When the SPI is disabled ( $SPE = 0$ ), these four pins revert to other functions that are not controlled by the SPI (based on chip configuration).

### 37.2.1 SPSCK — SPI Serial Clock

When the SPI is enabled as a slave, this pin is the serial clock input. When the SPI is enabled as a master, this pin is the serial clock output.

### 37.2.2 MOSI — Master Data Out, Slave Data In

When the SPI is enabled as a master and SPI pin control zero (SPC0) is 0 (not bidirectional mode), this pin is the serial data output. When the SPI is enabled as a slave and SPC0 is 0, this pin is the serial data input. If SPC0 is 1 to select single-wire bidirectional mode, and master mode is selected, this pin becomes the bidirectional data I/O pin (MOMI). Also, the bidirectional mode output enable bit determines whether the pin acts as an input (BIDIROE is 0) or an output (BIDIROE is 1). If SPC0 is 1 and slave mode is selected, this pin is not used by the SPI and reverts to other functions (based on chip configuration).

### 37.2.3 MISO — Master Data In, Slave Data Out

When the SPI is enabled as a master and SPI pin control zero (SPC0) is 0 (not bidirectional mode), this pin is the serial data input. When the SPI is enabled as a slave and SPC0 is 0, this pin is the serial data output. If SPC0 is 1 to select single-wire bidirectional mode, and slave mode is selected, this pin becomes the bidirectional data I/O pin (SISO), and the bidirectional mode output enable bit determines whether the pin acts as an input (BIDIROE is 0) or an output (BIDIROE is 1). If SPC0 is 1 and master mode is selected, this pin is not used by the SPI and reverts to other functions (based on chip configuration).

### 37.2.4 SS — Slave Select

When the SPI is enabled as a slave, this pin is the low-true slave select input. When the SPI is enabled as a master and mode fault enable is off (MODFEN is 0), this pin is not used by the SPI and reverts to other functions (based on chip configuration). When the SPI is enabled as a master and MODFEN is 1, the slave select output enable bit determines whether this pin acts as the mode fault input (SSOE is 0) or as the slave select output (SSOE is 1).

## 37.3 Memory Map and Register Descriptions

The SPI has 8-bit registers to select SPI options, to control baud rate, to report SPI status, to hold an SPI data match value, and for transmit/receive data.

**SPI memory map**

| Address offset (hex) | Absolute address (hex) | Register name                    | Width (in bits) | Access | Reset value | Section/ page              |
|----------------------|------------------------|----------------------------------|-----------------|--------|-------------|----------------------------|
| 0                    | 4007_6000              | SPI control register 1 (SPI0_C1) | 8               | R/W    | 04h         | <a href="#">37.3.1/661</a> |
| 1                    | 4007_6001              | SPI control register 2 (SPI0_C2) | 8               | R/W    | 00h         | <a href="#">37.3.2/663</a> |
| 2                    | 4007_6002              | SPI baud rate register (SPI0_BR) | 8               | R/W    | 00h         | <a href="#">37.3.3/664</a> |
| 3                    | 4007_6003              | SPI status register (SPI0_S)     | 8               | R      | 20h         | <a href="#">37.3.4/665</a> |
| 5                    | 4007_6005              | SPI data register (SPI0_D)       | 8               | R/W    | 00h         | <a href="#">37.3.5/667</a> |
| 7                    | 4007_6007              | SPI match register (SPI0_M)      | 8               | R/W    | 00h         | <a href="#">37.3.6/668</a> |
| 0                    | 4007_7000              | SPI control register 1 (SPI1_C1) | 8               | R/W    | 04h         | <a href="#">37.3.1/661</a> |
| 1                    | 4007_7001              | SPI control register 2 (SPI1_C2) | 8               | R/W    | 00h         | <a href="#">37.3.2/663</a> |
| 2                    | 4007_7002              | SPI baud rate register (SPI1_BR) | 8               | R/W    | 00h         | <a href="#">37.3.3/664</a> |
| 3                    | 4007_7003              | SPI status register (SPI1_S)     | 8               | R      | 20h         | <a href="#">37.3.4/665</a> |
| 5                    | 4007_7005              | SPI data register (SPI1_D)       | 8               | R/W    | 00h         | <a href="#">37.3.5/667</a> |
| 7                    | 4007_7007              | SPI match register (SPI1_M)      | 8               | R/W    | 00h         | <a href="#">37.3.6/668</a> |

### 37.3.1 SPI control register 1 (SPIx\_C1)

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

Address: 4007\_6000h base + 0h offset = 4007\_6000h

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

**SPI0\_C1 field descriptions**

| Field     | Description  |
|-----------|--|
| 7<br>SPIE | SPI interrupt enable: for SPRF and MODF<br><br>This bit enables the interrupt for SPI receive buffer full (SPRF) and mode fault (MODF) events. |

*Table continues on the next page...*

**SPI0\_C1 field descriptions (continued)**

| Field   | Description  |
|---------|--|
|         | <p>0 Interrupts from SPRF and MODF are inhibited—use polling<br/>1 Request a hardware interrupt when SPRF or MODF is 1</p>   |
| 6 SPE   | <p>SPI system enable</p> <p>This bit enables the SPI system and dedicates the SPI port pins to SPI system functions. If SPE is cleared, the SPI is disabled and forced into an idle state, and all status bits in the S register are reset.</p> <p>0 SPI system inactive<br/>1 SPI system enabled</p>  |
| 5 SPTIE | <p>SPI transmit interrupt enable</p> <p>This is the interrupt enable bit for SPI transmit buffer empty (SPTEF). An interrupt occurs when the SPI transmit buffer is empty (SPTEF is set).</p> <p>0 Interrupts from SPTEF inhibited (use polling)<br/>1 When SPTEF is 1, hardware interrupt requested</p>   |
| 4 MSTR  | <p>Master/slave mode select</p> <p>This bit selects master or slave mode operation.</p> <p>0 SPI module configured as a slave SPI device<br/>1 SPI module configured as a master SPI device</p>  |
| 3 CPOL  | <p>Clock polarity</p> <p>This bit selects an inverted or non-inverted SPI clock. To transmit data between SPI modules, the SPI modules must have identical CPOL values.</p> <p>This bit effectively places an inverter in series with the clock signal either from a master SPI device or to a slave SPI device. Refer to the description of “SPI Clock Formats” for details.</p> <p>0 Active-high SPI clock (idles low)<br/>1 Active-low SPI clock (idles high)</p>   |
| 2 CPHA  | <p>Clock phase</p> <p>This bit selects one of two clock formats for different kinds of synchronous serial peripheral devices. Refer to the description of “SPI Clock Formats” for details.</p> <p>0 First edge on SPSCK occurs at the middle of the first cycle of a data transfer<br/>1 First edge on SPSCK occurs at the start of the first cycle of a data transfer</p>   |
| 1 SSOE  | <p>Slave select output enable</p> <p>This bit is used in combination with the mode fault enable (MODFEN) bit in the C2 register and the master/slave (MSTR) control bit to determine the function of the <math>\overline{SS}</math> pin.</p> <p>0 When MODFEN is 0: In master mode, <math>\overline{SS}</math> pin function is general-purpose I/O (not SPI). In slave mode, <math>\overline{SS}</math> pin function is slave select input.<br/>When MODFEN is 1: In master mode, <math>\overline{SS}</math> pin function is <math>\overline{SS}</math> input for mode fault. In slave mode, <math>\overline{SS}</math> pin function is slave select input.</p> <p>1 When MODFEN is 0: In master mode, <math>\overline{SS}</math> pin function is general-purpose I/O (not SPI). In slave mode, <math>\overline{SS}</math> pin function is slave select input.<br/>When MODFEN is 1: In master mode, <math>\overline{SS}</math> pin function is automatic <math>\overline{SS}</math> output. In slave mode: <math>\overline{SS}</math> pin function is slave select input.</p> |

*Table continues on the next page...*

**SPI0\_C1 field descriptions (continued)**

| Field   | Description  |
|---------|--|
| 0 LSBFE | <p>LSB first (shifter direction)</p> <p>This bit does not affect the position of the MSB and LSB in the data register. Reads and writes of the data register always have the MSB in bit 7.</p> <p>0 SPI serial data transfers start with most significant bit<br/>1 SPI serial data transfers start with least significant bit</p> |

**37.3.2 SPI control register 2 (SPIx\_C2)**

This read/write register is used to control optional features of the SPI system. Bit 6 is not implemented and always reads 0.

Address: 4007\_6000h base + 1h offset = 4007\_6001h

| Bit        | 7     | 6        | 5      | 4      | 3       | 2      | 1       | 0    |
|------------|-------|----------|--------|--------|---------|--------|---------|------|
| Read Write | SPMIE | Reserved | TXDMAE | MODFEN | BIDIROE | RXDMAE | SPISWAI | SPC0 |
| Reset      | 0     | 0        | 0      | 0      | 0       | 0      | 0       | 0    |

**SPI0\_C2 field descriptions**

| Field      | Description  |
|------------|--|
| 7 SPMIE    | <p>SPI match interrupt enable</p> <p>This is the interrupt enable bit for the SPI receive data buffer hardware match (SPMF) function.</p> <p>0 Interrupts from SPMF inhibited (use polling)<br/>1 When SPMF is 1, requests a hardware interrupt</p>  |
| 6 Reserved | <p>This field is reserved.<br/>Do not write to this reserved bit.</p>  |
| 5 TXDMAE   | <p>Transmit DMA enable</p> <p>This bit enables a transmit DMA request. When this bit is set to 1, a transmit DMA request is asserted when both SPTEF and SPE are set, and the interrupt from SPTEF is disabled.</p> <p>0 DMA request for transmit is disabled and interrupt from SPTEF is allowed<br/>1 DMA request for transmit is enabled and interrupt from SPTEF is disabled</p>   |
| 4 MODFEN   | <p>Master mode-fault function enable</p> <p>When the SPI is configured for slave mode, this bit has no meaning or effect. (The SS pin is the slave select input.) In master mode, this bit determines how the SS pin is used. For details, refer to the description of the SSOE bit in the C1 register.</p> <p>0 Mode fault function disabled, master SS pin reverts to general-purpose I/O not controlled by SPI<br/>1 Mode fault function enabled, master SS pin acts as the mode fault input or the slave select output</p> |

*Table continues on the next page...*

**SPI0\_C2 field descriptions (continued)**

| Field        | Description   |
|--------------|---|
| 3<br>BIDIROE | <p>Bidirectional mode output enable</p> <p>When bidirectional mode is enabled, because SPI pin control 0 (SPC0) is set to 1, the BIDIROE bit determines whether the SPI data output driver is enabled to the single bidirectional SPI I/O pin. Depending on whether the SPI is configured as a master or a slave, it uses the MOSI (MOMI) or MISO (SISO) pin, respectively, as the single SPI data I/O pin. When SPC0 is 0, BIDIROE has no meaning or effect.</p> <p>0 Output driver disabled so SPI data I/O pin acts as an input<br/>1 SPI I/O pin enabled as an output</p>   |
| 2<br>RXDMAE  | <p>Receive DMA enable</p> <p>This is the enable bit for a receive DMA request. When this bit is set to 1, a receive DMA request is asserted when both SPRF and SPE are set, and the interrupt from SPRF is disabled.</p> <p>0 DMA request for receive is disabled and interrupt from SPRF is allowed<br/>1 DMA request for receive is enabled and interrupt from SPRF is disabled</p>   |
| 1<br>SPISWAI | <p>SPI stop in wait mode</p> <p>This bit is used for power conservation while the device is in wait mode.</p> <p>0 SPI clocks continue to operate in wait mode<br/>1 SPI clocks stop when the MCU enters wait mode</p>  |
| 0<br>SPC0    | <p>SPI pin control 0</p> <p>This bit enables bidirectional pin configurations.</p> <p>0 SPI uses separate pins for data input and data output (pin mode is normal).<br/>In master mode of operation: MISO is master in and MOSI is master out.<br/>In slave mode of operation: MISO is slave out and MOSI is slave in.<br/>1 SPI configured for single-wire bidirectional operation (pin mode is bidirectional).<br/>In master mode of operation: MISO is not used by SPI; MOSI is master in when BIDIROE is 0 or master I/O when BIDIROE is 1.<br/>In slave mode of operation: MISO is slave in when BIDIROE is 0 or slave I/O when BIDIROE is 1; MOSI is not used by SPI.</p> |

**37.3.3 SPI baud rate register (SPIx\_BR)**

Use this register to set the prescaler and bit rate divisor for an SPI master. This register may be read or written at any time.

Address: 4007\_6000h base + 2h offset = 4007\_6002h

| Bit   | 7 | 6 | 5         | 4 | 3 | 2        | 1 | 0 |
|-------|---|---|-----------|---|---|----------|---|---|
| Read  | 0 |   | SPPR[2:0] |   |   | SPR[3:0] |   |   |
| Write |   |   |           |   |   |          |   |   |
| Reset | 0 | 0 | 0         | 0 | 0 | 0        | 0 | 0 |

**SPI0\_BR field descriptions**

| Field            | Description   |
|------------------|---|
| 7<br>Reserved    | This field is reserved.<br>This read-only field is reserved and always has the value 0.   |
| 6–4<br>SPPR[2:0] | SPI baud rate prescale divisor<br><br>This 3-bit field selects one of eight divisors for the SPI baud rate prescaler. The input to this prescaler is the bus rate clock (BUSCLK). The output of this prescaler drives the input of the SPI baud rate divider. Refer to the description of “SPI Baud Rate Generation” for details.<br><br>000 Baud rate prescaler divisor is 1<br>001 Baud rate prescaler divisor is 2<br>010 Baud rate prescaler divisor is 3<br>011 Baud rate prescaler divisor is 4<br>100 Baud rate prescaler divisor is 5<br>101 Baud rate prescaler divisor is 6<br>110 Baud rate prescaler divisor is 7<br>111 Baud rate prescaler divisor is 8 |
| 3–0<br>SPR[3:0]  | SPI baud rate divisor<br><br>This 4-bit field selects one of nine divisors for the SPI baud rate divider. The input to this divider comes from the SPI baud rate prescaler. Refer to the description of “SPI Baud Rate Generation” for details.<br><br>0000 Baud rate divisor is 2<br>0001 Baud rate divisor is 4<br>0010 Baud rate divisor is 8<br>0011 Baud rate divisor is 16<br>0100 Baud rate divisor is 32<br>0101 Baud rate divisor is 64<br>0110 Baud rate divisor is 128<br>0111 Baud rate divisor is 256<br>1000 Baud rate divisor is 512<br>All others Reserved  |

**37.3.4 SPI status register (SPIx\_S)**

This register contains read-only status bits. Writes have no meaning or effect.

**NOTE**

Bits 3 through 0 are not implemented and always read 0.

Address: 4007\_6000h base + 3h offset = 4007\_6003h

| Bit   | 7    | 6    | 5     | 4    | 3 | 2 | 1 | 0 |
|-------|------|------|-------|------|---|---|---|---|
| Read  | SPRF | SPMF | SPTEF | MODF |   |   | 0 |   |
| Write |      |      |       |      |   |   |   |   |
| Reset | 0    | 0    | 1     | 0    | 0 | 0 | 0 | 0 |

**SPI0\_S field descriptions**

| Field           | Description   |   |   |   |  |
|-----------------|---|---|---|---|--|
| 7<br>SPRF       | <p>SPI read buffer full flag</p> <p>SPRF is set at the completion of an SPI transfer to indicate that received data may be read from the SPI data (D) register. When the receive DMA request is disabled (RXDMAE is 0), SPRF is cleared by reading SPRF while it is set and then reading the SPI data register. When the receive DMA request is enabled (RXDMAE is 1), SPRF is automatically cleared when the DMA transfer for the receive DMA request is completed (RX DMA Done is asserted).</p> <table> <tr> <td>0</td><td>No data available in the receive data buffer</td></tr> <tr> <td>1</td><td>Data available in the receive data buffer</td></tr> </table>  | 0 | No data available in the receive data buffer                                | 1 | Data available in the receive data buffer                            |
| 0               | No data available in the receive data buffer  |   |   |   |  |
| 1               | Data available in the receive data buffer   |   |   |   |  |
| 6<br>SPMF       | <p>SPI match flag</p> <p>SPMF is set after SPRF is 1 when the value in the receive data buffer matches the value in the M register. To clear the flag, read SPMF when it is set and then write a 1 to it.</p> <table> <tr> <td>0</td><td>Value in the receive data buffer does not match the value in the M register</td></tr> <tr> <td>1</td><td>Value in the receive data buffer matches the value in the M register</td></tr> </table>   | 0 | Value in the receive data buffer does not match the value in the M register | 1 | Value in the receive data buffer matches the value in the M register |
| 0               | Value in the receive data buffer does not match the value in the M register   |   |   |   |  |
| 1               | Value in the receive data buffer matches the value in the M register  |   |   |   |  |
| 5<br>SPTEF      | <p>SPI transmit buffer empty flag</p> <p>This bit is set when the transmit data buffer is empty. When the transmit DMA request is disabled (TXDMAE is 0), SPTEF is cleared by reading the S register with SPTEF set and then writing a data value to the transmit buffer at D. The S register must be read with SPTEF set to 1 before writing data to the D register; otherwise, the D write is ignored. When the transmit DMA request is enabled (TXDMAE is 1), SPTEF is automatically cleared when the DMA transfer for the transmit DMA request is completed (TX DMA Done is asserted). SPTEF is automatically set when all data from the transmit buffer transfers into the transmit shift register. For an idle SPI, data written to D is transferred to the shifter almost immediately so that SPTEF is set within two bus cycles, allowing a second set of data to be queued into the transmit buffer. After completion of the transfer of the data in the shift register, the queued data from the transmit buffer automatically moves to the shifter, and SPTEF is set to indicate that room exists for new data in the transmit buffer. If no new data is waiting in the transmit buffer, SPTEF simply remains set and no data moves from the buffer to the shifter.</p> <p>If a transfer does not stop, the last data that was transmitted is sent out again.</p> <table> <tr> <td>0</td><td>SPI transmit buffer not empty</td></tr> <tr> <td>1</td><td>SPI transmit buffer empty</td></tr> </table> | 0 | SPI transmit buffer not empty   | 1 | SPI transmit buffer empty  |
| 0               | SPI transmit buffer not empty   |   |   |   |  |
| 1               | SPI transmit buffer empty   |   |   |   |  |
| 4<br>MODF       | <p>Master mode fault flag</p> <p>MODF is set if the SPI is configured as a master and the slave select input goes low, indicating some other SPI device is also configured as a master. The SS pin acts as a mode fault error input only when MSTR is 1, MODFEN is 1, and SSOE is 0; otherwise, MODF will never be set. MODF is cleared by reading MODF while it is 1 and then writing to the SPI control register 1 (C1).</p> <table> <tr> <td>0</td><td>No mode fault error</td></tr> <tr> <td>1</td><td>Mode fault error detected</td></tr> </table>   | 0 | No mode fault error   | 1 | Mode fault error detected  |
| 0               | No mode fault error   |   |   |   |  |
| 1               | Mode fault error detected   |   |   |   |  |
| 3–0<br>Reserved | <p>This field is reserved.</p> <p>This read-only field is reserved and always has the value 0.</p>  |   |   |   |  |

### 37.3.5 SPI data register (SPIx\_D)

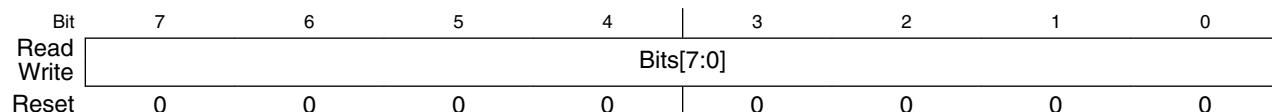
This register is both the input and output register for SPI data. A write to the register writes to the transmit data buffer, allowing data to be queued and transmitted.

When the SPI is configured as a master, data queued in the transmit data buffer is transmitted immediately after the previous transmission has completed.

The SPTEF bit in the S register indicates when the transmit data buffer is ready to accept new data. When the transmit DMA request is disabled (TXDMAE is 0): The S register must be read when SPTEF is set before writing to the SPI data register; otherwise, the write is ignored. When the transmit DMA request is enabled (TXDMAE is 1) when SPTEF is set, the SPI data register can be written automatically by DMA without reading the S register first.

Data may be read from the SPI data register any time after SPRF is set and before another transfer is finished. Failure to read the data out of the receive data buffer before a new transfer ends causes a receive overrun condition, and the data from the new transfer is lost. The new data is lost because the receive buffer still held the previous character and was not ready to accept the new data. There is no indication for a receive overrun condition, so the application system designer must ensure that previous data has been read from the receive buffer before a new transfer is initiated.

Address: 4007\_6000h base + 5h offset = 4007\_6005h



**SPI0\_D field descriptions**

| Field            | Description     |
|------------------|-----------------|
| 7–0<br>Bits[7:0] | Data (low byte) |

### 37.3.6 SPI match register (SPIx\_M)

This register contains the hardware compare value. When the value received in the SPI receive data buffer equals this hardware compare value, the SPI match flag (SPMF) sets.

Address: 4007\_6000h base + 7h offset = 4007\_6007h

| Bit   | 7 | 6 | 5 | 4 |           | 3 | 2 | 1 | 0 |
|-------|---|---|---|---|-----------|---|---|---|---|
| Read  | 0 | 0 | 0 | 0 | Bits[7:0] | 0 | 0 | 0 | 0 |
| Write |   |   |   |   |           |   |   |   |   |
| Reset | 0 | 0 | 0 | 0 | 0         | 0 | 0 | 0 | 0 |

#### SPI0\_M field descriptions

| Field            | Description                       |
|------------------|-----------------------------------|
| 7–0<br>Bits[7:0] | Hardware compare value (low byte) |

## 37.4 Functional Description

This section provides the functional description of the module.

### 37.4.1 General

The SPI system is enabled by setting the SPI enable (SPE) bit in SPI Control Register 1. While the SPE bit is set, the four associated SPI port pins are dedicated to the SPI function as:

- Slave select (SS)
- Serial clock (SPSCK)
- Master out/slave in (MOSI)
- Master in/slave out (MISO)

An SPI transfer is initiated in the master SPI device by reading the SPI status register (SPIx\_S) when SPTEF = 1 and then writing data to the transmit data buffer (write to SPIxD). When a transfer is complete, received data is moved into the receive data buffer. The SPIxD register acts as the SPI receive data buffer for reads and as the SPI transmit data buffer for writes.

The clock phase control bit (CPHA) and clock polarity control bit (CPOL) in the SPI Control Register 1 (SPIx\_C1) 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 SPSCK edges or on even numbered SPSCK edges.

The SPI can be configured to operate as a master or as a slave. When the MSTR bit in SPI Control Register 1 is set, master mode is selected; when the MSTR bit is clear, slave mode is selected.

### 37.4.2 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 reading the SPIx\_S register while SPTEF = 1 and writing to the master SPI data registers. If the shift register is empty, the byte immediately transfers to the shift register. The data begins shifting out on the MOSI pin under the control of the serial clock.

- SPSCK
  - The SPR3, 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 SPSCK pin is the SPI clock output. Through the SPSCK 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.
- $\overline{\text{SS}}$  pin
  - If MODFEN and SSOE bit are set, the SS pin is configured as slave select output. The SS output becomes low during each transmission and is high when the SPI is in idle state. If MODFEN is set and SSOE is cleared, the  $\overline{\text{SS}}$  pin is configured as input for detecting mode fault error. If the SS input becomes low this indicates a mode fault error where another master tries to drive the MOSI and SPSCK 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). As a result, all outputs are disabled, and SPSCK, 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 (SPIx\_S). If the SPI interrupt enable bit (SPIE) is set when the MODF flag gets set, then an SPI interrupt sequence is also requested. When a write to the SPI Data Register in the master occurs, there is a half SPSCK-cycle delay. After the delay, SPSCK 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 [SPI Clock Formats](#)).

### Note

A change of the bits CPOL, CPHA, SSOE, LSBFE, MODFEN, SPC0, BIDIROE with SPC0 set, SPPR2-SPPR0 and SPR3-SPR0 in master mode abort a transmission in progress and force the SPI into idle state. The remote slave cannot detect this, therefore the master has to ensure that the remote slave is set back to idle state.

### 37.4.3 Slave Mode

The SPI operates in slave mode when the MSTR bit in SPI Control Register1 is clear.

- SPSCK

In slave mode, SPSCK 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.

- SS pin

The SS pin is the slave select input. Before a data transmission occurs, the SS pin of the slave SPI must be low. SS must remain low until the transmission is complete. If SS goes high, the SPI is forced into an idle state.

The SS input also controls the serial data output pin. If SS is high (not selected), the serial data output pin is high impedance. If 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 (SS is high), then the SPSCK 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 SPSCK 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 SPSCK 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.

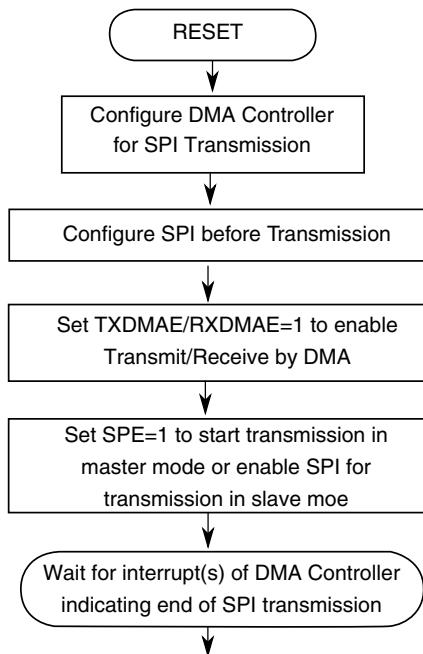
When CPHA is set, the first edge is used to get the first data bit onto the serial data output pin. When CPHA is clear and the SS input is low (slave selected), the first bit of the SPI data is driven out of the serial data output pin. After the eighth shift, the transfer is considered complete and the received data is transferred into the SPI Data register. To indicate transfer is complete, the SPRF flag in the SPI Status Register is set.

### Note

A change of the bits BIDIROE with SPC0 set, CPOL, CPHA, SSOE, LSBFE, MODFEN, and SPC0 in slave mode will corrupt a transmission in progress and must be avoided.

#### 37.4.4 SPI Transmission by DMA

SPI supports both Transmit and Receive by DMA. The basic flow of SPI transmission by DMA is as below.

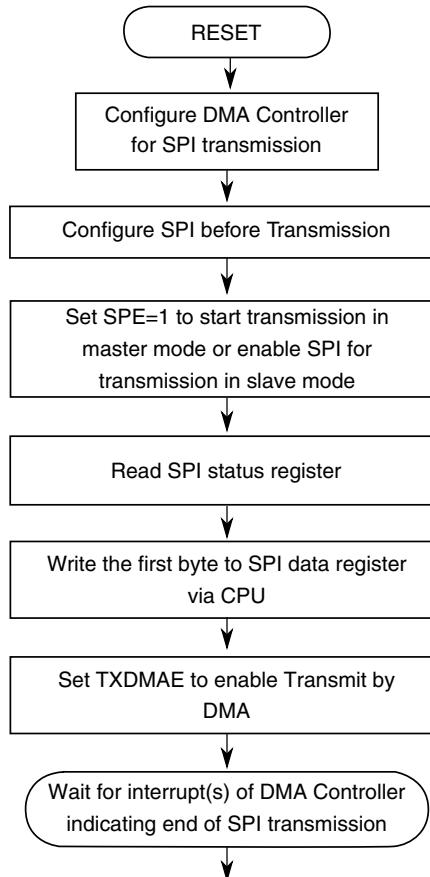


**Figure 37-21. Basic Flow of SPI Transmission by DMA**

#### 37.4.4.1 Transmit by DMA

Transmit by DMA is supported only when TXDMAE is set. A transmit DMA request is asserted when both SPE and SPTEF are set. Then the on-chip DMA controller detects this request and transfers data from memory into the SPI data register. After that, TX DMA DONE is asserted to clear SPTEF automatically. This process repeats until all data for transmission (the number is decided by the configuration register[s] of the DMA controller) is sent.

After DMA transfers the first byte to the SPI data register, the SPI pushes this data into the shifter, thereby making SPTEF high again. This generates another DMA request immediately, but the CPU lacks enough time to service the first DMA interrupt service request (ISR). The subsequent DMA request is paced at the SPI transfer rate. Manage this behavior during the first byte transfer through the DMA channel. Write the first byte to the SPI data register via the CPU. The other bytes are transmitted by the DMA.



**Figure 37-22. Recommended startup of SPI transmit by DMA**

### 37.4.4.2 Receive by DMA

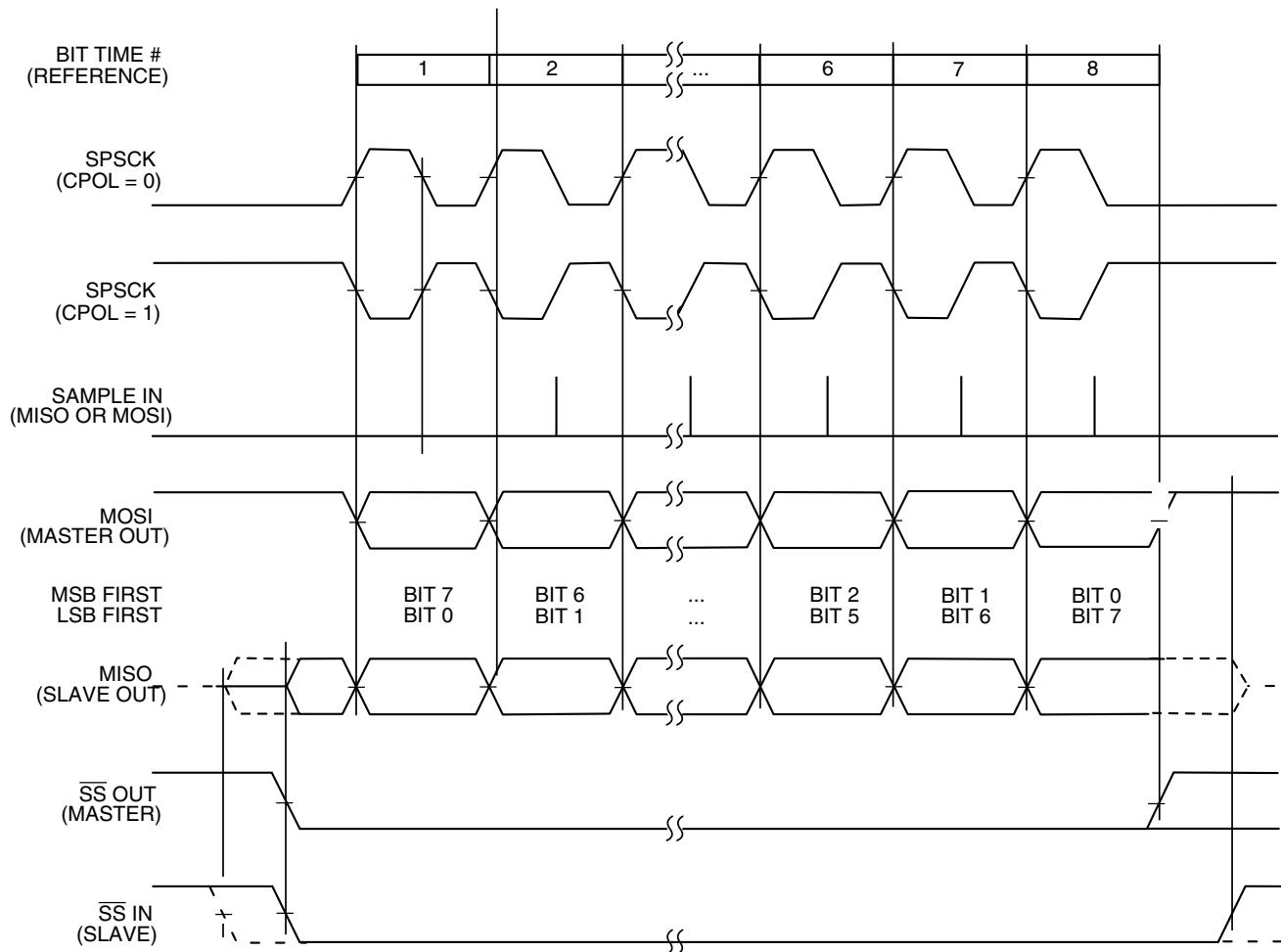
Receive by DMA is supported only when RXDMAE is set. A receive DMA request is asserted when both SPE and SPRF are set. Then the on-chip DMA controller detects this request and transfers data from the SPI data register into memory. After that, RX DMA DONE is asserted to clear SPRF automatically. This process repeats until all data to be received (the number is decided by configuration register[s] of the DMA controller) is received or no receive DMA request is generated again because the SPI transmission is finished.

### 37.4.5 SPI Clock Formats

To accommodate a wide variety of synchronous serial peripherals from different manufacturers, the SPI system has a clock polarity (CPOL) bit and a clock phase (CPHA) control bit to select one of four clock formats for data transfers. CPOL selectively inserts an inverter in series with the clock. CPHA chooses between two different clock phase relationships between the clock and data.

## Functional Description

The following figure shows the clock formats when CPHA = 1. At the top of the figure, the eight bit times are shown for reference with bit 1 starting at the first SPSCK edge and bit 8 ending one-half SPSCK cycle after the eighth SPSCK edge. The MSB first and LSB first lines show the order of SPI data bits depending on the setting in LSBFE. Both variations of SPSCK polarity are shown, but only one of these waveforms applies for a specific transfer, depending on the value in CPOL. The SAMPLE IN waveform applies to the MOSI input of a slave or the MISO input of a master. The MOSI waveform applies to the MOSI output pin from a master and the MISO waveform applies to the MISO output from a slave. The SS OUT waveform applies to the slave select output from a master (provided MODFEN and SSOE = 1). The master SS output goes to active low one-half SPSCK cycle before the start of the transfer and goes back high at the end of the eighth bit time of the transfer. The SS IN waveform applies to the slave select input of a slave.



**Figure 37-23. SPI Clock Formats (CPHA = 1)**

When CPHA = 1, the slave begins to drive its MISO output when SS goes to active low, but the data is not defined until the first SPSCK edge. The first SPSCK edge shifts the first bit of data from the shifter onto the MOSI output of the master and the MISO output

of the slave. The next SPSCK edge causes both the master and the slave to sample the data bit values on their MISO and MOSI inputs, respectively. At the third SPSCK edge, the SPI shifter shifts one bit position which shifts in the bit value that was just sampled, and shifts the second data bit value out the other end of the shifter to the MOSI and MISO outputs of the master and slave, respectively.

When CPHA = 1, the slave's  $\overline{SS}$  input is not required to go to its inactive high level between transfers. In this clock format, a back-to-back transmission can occur, as follows:

1. A transmission is in progress.
2. A new data byte is written to the transmit buffer before the in-progress transmission is complete.
3. When the in-progress transmission is complete, the new, ready data byte is transmitted immediately.

Between these two successive transmissions, no pause is inserted; the  $\overline{SS}$  pin remains low.

The following figure shows the clock formats when CPHA = 0. At the top of the figure, the eight bit times are shown for reference with bit 1 starting as the slave is selected (SS IN goes low), and bit 8 ends at the last SPSCK edge. The MSB first and LSB first lines show the order of SPI data bits depending on the setting in LSBFE. Both variations of SPSCK polarity are shown, but only one of these waveforms applies for a specific transfer, depending on the value in CPOL. The SAMPLE IN waveform applies to the MOSI input of a slave or the MISO input of a master. The MOSI waveform applies to the MOSI output pin from a master and the MISO waveform applies to the MISO output from a slave. The SS OUT waveform applies to the slave select output from a master (provided MODFEN and SSOE = 1). The master SS output goes to active low at the start of the first bit time of the transfer and goes back high one-half SPSCK cycle after the end of the eighth bit time of the transfer. The SS IN waveform applies to the slave select input of a slave.

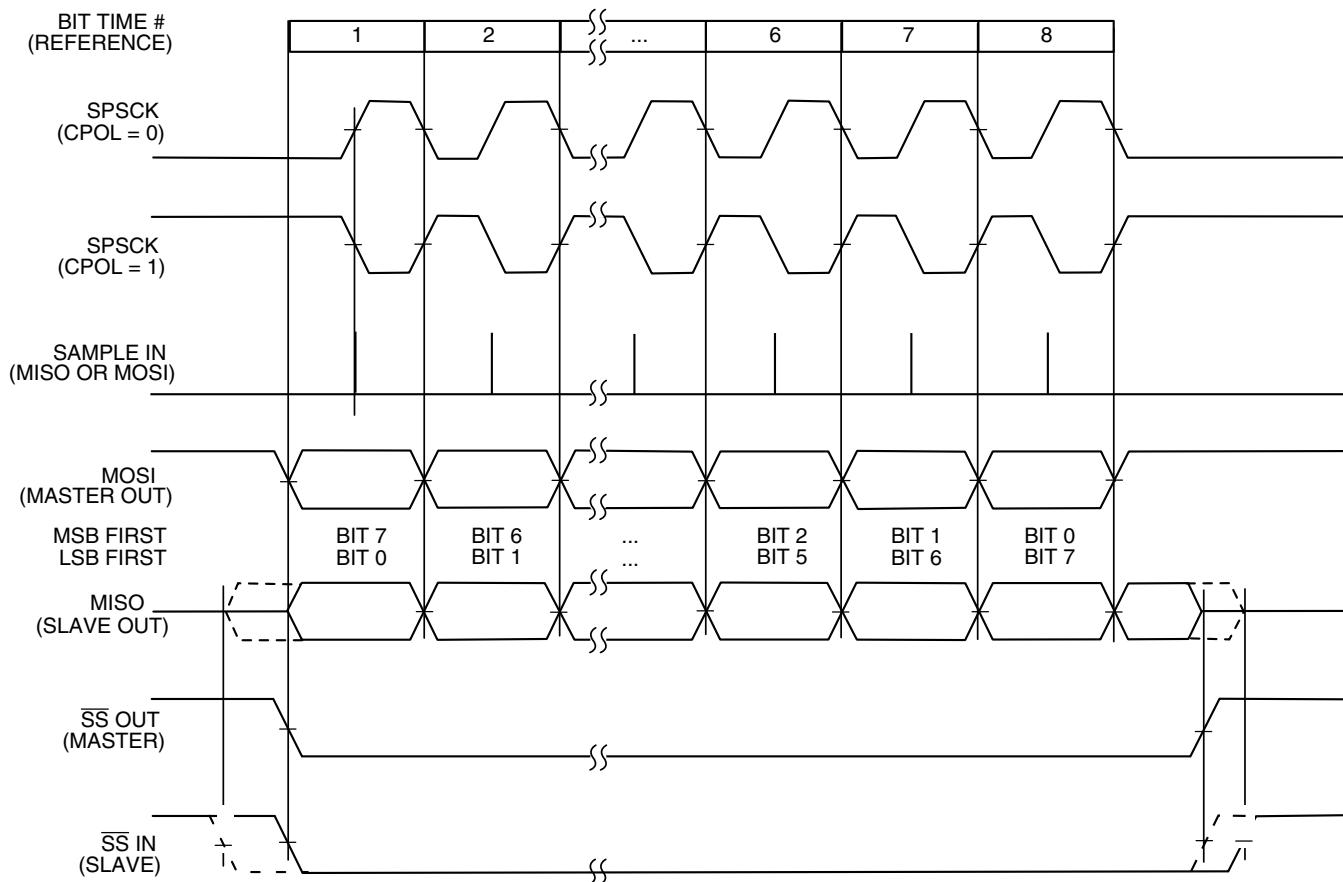


Figure 37-24. SPI Clock Formats (CPHA = 0)

When CPHA = 0, the slave begins to drive its MISO output with the first data bit value (MSB or LSB depending on LSBFE) when SS goes to active low. The first SPSCK edge causes both the master and the slave to sample the data bit values on their MISO and MOSI inputs, respectively. At the second SPSCK edge, the SPI shifter shifts one bit position which shifts in the bit value that was just sampled and shifts the second data bit value out the other end of the shifter to the MOSI and MISO outputs of the master and slave, respectively. When CPHA = 0, the slave's SS input must go to its inactive high level between transfers.

### 37.4.6 SPI Baud Rate Generation

As shown in the following figure, the clock source for the SPI baud rate generator is the bus clock. The three prescale bits (SPPR2:SPPR1:SPPR0) choose a prescale divisor of 1, 2, 3, 4, 5, 6, 7, or 8. The three rate select bits (SPR3:SPR2:SPR1:SPR0) divide the output of the prescaler stage by 2, 4, 8, 16, 32, 64, 128, 256, or 512 to get the internal SPI master mode bit-rate clock.

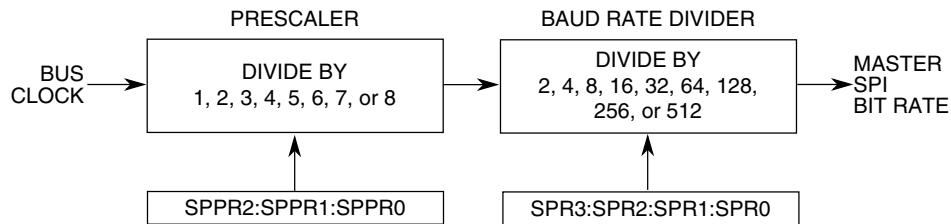
The baud rate generator is activated only when the SPI is in the master mode and a serial transfer is taking place. In the other cases, the divider is disabled to decrease  $I_{DD}$  current.

The baud rate divisor equation is as follows (except those reserved combinations in the SPI Baud Rate Divisor table).

$$\text{BaudRateDivisor} = (\text{SPPR} + 1) \times 2^{(\text{SPR} + 1)}$$

The baud rate can be calculated with the following equation:

$$\text{BaudRate} = \text{BusClock} / \text{BaudRateDivisor}$$



**Figure 37-25. SPI Baud Rate Generation**

## 37.4.7 Special Features

The following section shows the module special features.

### 37.4.7.1 SS Output

The  $\overline{\text{SS}}$  output feature automatically drives the  $\overline{\text{SS}}$  pin low during transmission to select external devices and drives the  $\overline{\text{SS}}$  pin high during idle to deselect external devices. When the  $\overline{\text{SS}}$  output is selected, the  $\overline{\text{SS}}$  output pin is connected to the  $\overline{\text{SS}}$  input pin of the external device.

The  $\overline{\text{SS}}$  output is available only in master mode during normal SPI operation by asserting the SSOE and MODFEN bits as shown in the description of the C1[SSOE] bit.

The mode fault feature is disabled while  $\overline{\text{SS}}$  output is enabled.

#### Note

Be careful when using the  $\overline{\text{SS}}$  output feature in a multimaster system because the mode fault feature is not available for detecting system errors between masters.

### 37.4.7.2 Bidirectional Mode (MOMI or SISO)

The bidirectional mode is selected when the SPC0 bit is set in SPI Control Register 2 (see the following table). In this mode, the SPI uses only one serial data pin for the interface with one or more external devices. 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 37-22. Normal Mode and Bidirectional Mode**

| When SPE = 1                          | Master Mode MSTR = 1 | Slave Mode MSTR = 0 |
|---------------------------------------|----------------------|---------------------|
| <b>Normal Mode</b><br>SPC0 = 0        |                      |                     |
| <b>Bidirectional Mode</b><br>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 SPSCK is an output for the master mode and an input for the slave mode.

$\overline{SS}$  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 SPSCK and  $\overline{SS}$  functions.

#### Note

In bidirectional master mode, with the mode fault feature 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. Consider this scenario if the MISO pin is used for another purpose.

## 37.4.8 Error Conditions

The SPI module has one error condition: the mode fault error.

### 37.4.8.1 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 SPSCK lines simultaneously. This condition is not permitted in normal operation, and it sets the MODF bit in the SPI status register automatically provided that the MODFEN bit is set.

In the special case where the SPI is in master mode and the 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. If the SPI system is configured as a slave, the  $\overline{SS}$  pin is a dedicated input pin. A mode fault error does not 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 the SPSCK, 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 an SPI system configured in master mode, the output enable of MOMI (MOSI in bidirectional mode) is cleared if it was set. No mode fault error occurs in the bidirectional mode for the 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.

## 37.4.9 Low Power Mode Options

This section describes the low power mode options.

### 37.4.9.1 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 can still be accessed, but clocks to the core of this module are disabled.

### 37.4.9.2 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 SPSCK continues to be driven from the master. This keeps the slave synchronized to the master and the SPSCK.

If the master transmits data while the slave is in wait mode, the slave continues to send data consistent with the operation mode at the start of wait mode (that is, if the slave is currently sending its SPIx\_D to the master, it continues to send the same byte. Otherwise, if the slave is currently sending the last data received byte from the master, it continues to send each previously received data from the master byte).

#### Note

Care must be taken when expecting data from a master while the slave is in a wait mode or a stop mode where the peripheral bus clock is stopped but internal logic states are retained. Even though the shift register continues to operate, the rest of the SPI is shut down (that is, an SPRF interrupt is not generated until an exit from stop or wait mode). Also, the data from the shift register is not copied into the SPIx\_D registers until after the slave SPI has exited wait or stop mode. An SPRF flag and SPIx\_D copy is only generated if wait mode is entered or exited during a transmission. If the slave enters wait mode in idle mode and exits wait mode in idle mode, neither an SPRF nor a SPIx\_D copy occurs.

### 37.4.9.3 SPI in Stop Mode

Operation in a stop mode where the peripheral bus clock is stopped but internal logic states are retained depends on the SPI system. The stop mode does not depend on the SPISWAI bit. Upon entry to this type of stop mode, the SPI module clock is disabled (held high or low).

- If the SPI is in master mode and exchanging data when the CPU enters the stop mode, the transmission is frozen until the CPU exits stop mode. After the exit from stop mode, data to and from the external SPI is exchanged correctly.
- In slave mode, the SPI remains synchronized with the master.

The SPI is completely disabled in a stop mode where the peripheral bus clock is stopped and internal logic states are not retained. After an exit from this type of stop mode, all registers are reset to their default values, and the SPI module must be re-initialized.

### 37.4.10 Reset

The reset values of registers and signals are described in the Memory Map and Register Descriptions content, which details the registers and their bitfields.

- If a data transmission occurs in slave mode after a reset without a write to SPIx\_D, the transmission consists of "garbage" or the data last received from the master before the reset.
- Reading from SPIx\_D after reset always returns zeros.

### 37.4.11 Interrupts

The SPI originates interrupt requests only when the SPI is enabled (the SPE bit in the SPIx\_C1 register is 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.

Four flag bits, three interrupt mask bits, and one interrupt vector are associated with the SPI system. The SPI interrupt enable mask (SPIE) enables interrupts from the SPI receiver full flag (SPRF) and mode fault flag (MODF). The SPI transmit interrupt enable mask (SPTIE) enables interrupts from the SPI transmit buffer empty flag (SPTEF). The SPI match interrupt enable mask bit (SPIMIE) enables interrupts from the SPI match flag (SPMF). When one of the flag bits is set, and the associated interrupt mask bit is set, a hardware interrupt request is sent to the CPU. If the interrupt mask bits are cleared, software can poll the associated flag bits instead of using interrupts. The SPI interrupt

service routine (ISR) should check the flag bits to determine which event caused the interrupt. The service routine should also clear the flag bit(s) before returning from the ISR (usually near the beginning of the ISR).

### **37.4.11.1 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 the description of the C1[SSOE] bit). Once MODF is set, the current transfer is aborted and the master (MSTR) bit in the SPIx\_C1 register resets to 0.

The MODF interrupt is reflected in the status register's MODF flag. Clearing the flag also clears the interrupt. This interrupt stays active while the MODF flag is set. MODF has an automatic clearing process that is described in the SPI Status Register.

### **37.4.11.2 SPRF**

SPRF occurs when new data has been received and copied to the SPI receive data buffer.

After SPRF is set, it does not clear until it is serviced. SPRF has an automatic clearing process that is described in the SPI Status Register details. If the SPRF is not serviced before the end of the next transfer (that is, SPRF remains active throughout another transfer), the subsequent transfers are ignored and no new data is copied into the Data register.

### **37.4.11.3 SPTEF**

SPTEF occurs when the SPI transmit buffer is ready to accept new data.

After SPTEF is set, it does not clear until it is serviced. SPTEF has an automatic clearing process that is described in the SPI Status Register details.

### **37.4.11.4 SPMF**

SPMF occurs when the data in the receive data buffer is equal to the data in the SPI match register.

### 37.4.11.5 Asynchronous interrupt in low power modes

When the CPU is in wait mode or stop mode and the SPI module receives a transmission, the SPI module can generate an asynchronous interrupt to wake the CPU from the low power mode. The module generates the asynchronous interrupt only when all of the following conditions apply:

1. The C1[SPIE] bit is set to 1.
2. The CPU is in wait mode—in which case the C2[SPISWAI] bit must be 1—or in stop mode where the peripheral bus clock is stopped but internal logic states are retained.
3. The SPI module is in slave mode.
4. The received transmission ends.

After the interrupt wakes the CPU and the peripheral bus clock is active again, the SPI module copies the received data from the shifter into the Data register and generates flags or DMA request signals. During the wakeup phase, a continuous transmission from a master would destroy the first received data.

## 37.5 Initialization/Application Information

This section discusses an example of how to initialize and use the SPI.

### 37.5.1 Initialization Sequence

Before the SPI module can be used for communication, an initialization procedure must be carried out, as follows:

1. Update control register 1 (SPIx\_C1) to enable the SPI and to control interrupt enables. This register also sets the SPI as master or slave, determines clock phase and polarity, and configures the main SPI options.
2. Update control register 2 (SPIx\_C2) to enable additional SPI functions such as the SPI match interrupt feature, the master mode-fault function, and bidirectional mode output as well as to control and other optional features.
3. Update the baud rate register (SPIx\_BR) to set the prescaler and bit rate divisor for an SPI master.
4. Update the hardware match register (SPIx\_M) with the value to be compared to the receive data register for triggering an interrupt if hardware match interrupts are enabled.

5. In the master, read SPIx\_S while SPTEF = 1, and then write to the transmit data register (SPIx\_D) to begin transfer.

### 37.5.2 Pseudo-Code Example

In this example, the SPI module is set up for master mode with only hardware match interrupts enabled. The SPI runs at a maximum baud rate of bus clock divided by 2. Clock phase and polarity are set for an active-high SPI clock where the first edge on SPSCK occurs at the start of the first cycle of a data transfer.

| <b>SPIx_C1=0x54(%01010100)</b> |       |       |     |   |
|--------------------------------|-------|-------|-----|---|
|                                | Bit 7 | SPIE  | = 0 | Disables receive and mode fault interrupts                |
|                                | Bit 6 | SPE   | = 1 | Enables the SPI system                                    |
|                                | Bit 5 | SPTIE | = 0 | Disables SPI transmit interrupts                          |
|                                | Bit 4 | MSTR  | = 1 | Sets the SPI module as a master SPI device                |
|                                | Bit 3 | CPOL  | = 0 | Configures SPI clock as active-high                       |
|                                | Bit 2 | CPHA  | = 1 | First edge on SPSCK at start of first data transfer cycle |
|                                | Bit 1 | SSOE  | = 0 | Determines SS pin function when mode fault enabled        |
|                                | Bit 0 | LSBFE | = 0 | SPI serial data transfers start with most significant bit |

| <b>SPIx_C2 = 0x80(%10000000)</b> |       |         |     |  |
|----------------------------------|-------|---------|-----|--|
|                                  | Bit 7 | SPMIE   | = 1 | SPI hardware match interrupt enabled         |
|                                  |       |         | = 0 | Unimplemented                                |
|                                  | Bit 5 | TXDMAE  | = 0 | DMA request disabled                         |
|                                  | Bit 4 | MODFEN  | = 0 | Disables mode fault function                 |
|                                  | Bit 3 | BIDIROE | = 0 | SPI data I/O pin acts as input               |
|                                  | Bit 2 | RXDMAE  | = 0 | DMA request disabled                         |
|                                  | Bit 1 | SPISWAI | = 0 | SPI clocks operate in wait mode              |
|                                  | Bit 0 | SPC0    | = 0 | uses separate pins for data input and output |

| <b>SPIx_BR = 0x00(%00000000)</b> |         |  |        |                             |
|----------------------------------|---------|--|--------|-----------------------------|
|                                  | Bit 7   |  | = 0    | Reserved                    |
|                                  | Bit 6:4 |  | = 000  | Sets prescale divisor to 1  |
|                                  | Bit 3:0 |  | = 0000 | Sets baud rate divisor to 2 |

| <b>SPIx_S = 0x00(%00000000)</b> |       |      |     |   |
|---------------------------------|-------|------|-----|---|
|                                 | Bit 7 | SPRF | = 0 | Flag is set when receive data buffer is full  |
|                                 | Bit 6 | SPMF | = 0 | Flag is set when SPIx_M = receive data buffer |

Table continues on the next page...

**SPIx\_S = 0x00(%00000000)**

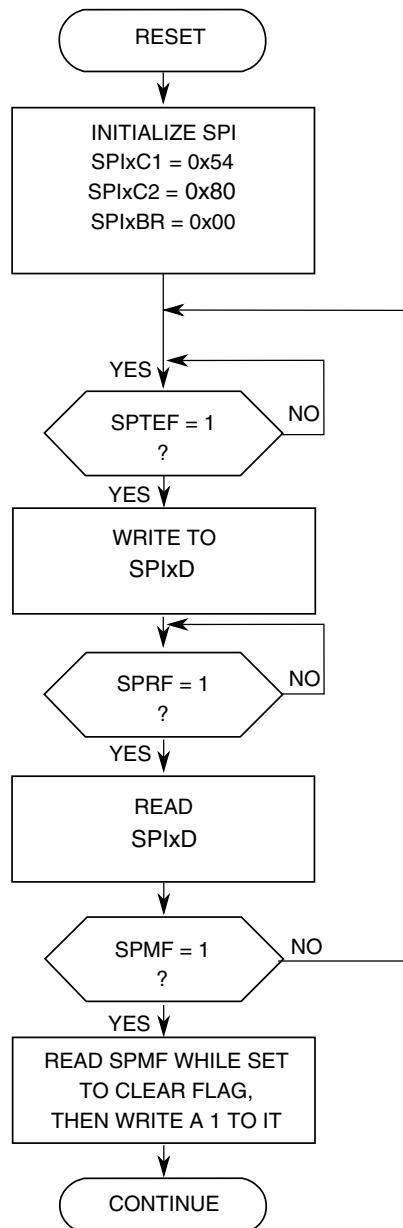
|  |         |       |   |   |  |
|--|---------|-------|---|---|--|
|  | Bit 5   | SPTEF | = | 0 | Flag is set when transmit data buffer is empty |
|  | Bit 4   | MODF  | = | 0 | Mode fault flag for master mode                |
|  | Bit 3:0 |       | = | 0 | Reserved                                       |

**SPIx\_M = 0xXX**

|  |  |
|--|--|
|  | Holds bits 0–7 of the hardware match buffer. |
|--|--|

**SPIx\_D = 0xxx**

|  |   |
|--|---|
|  | Holds bits 0–7 of the data to be transmitted by the transmit buffer and received by the receive buffer. |
|--|---|



**Figure 37-26. Initialization Flowchart Example for SPI Master Device**

# Chapter 38

## Inter-Integrated Circuit (I2C)

### 38.1 Introduction

#### NOTE

For the chip-specific implementation details of this module's instances see the chip configuration information.

The inter-integrated circuit ( $I^2C$ , I2C, or IIC) module provides a method of communication between a number of devices. The interface is designed to operate up to 100 kbit/s with maximum bus loading and timing. The I2C device is capable of operating at higher baud rates, up to a maximum of clock/20, with reduced bus loading. The maximum communication length and the number of devices that can be connected are limited by a maximum bus capacitance of 400 pF. The I2C module also complies with the *System Management Bus (SMBus) Specification, version 2*.

#### 38.1.1 Features

The I2C module has the following features:

- Compatible with *The I<sup>2</sup>C-Bus Specification*
- Multimaster operation
- Software programmable for one of 64 different serial clock frequencies
- Software-selectable acknowledge bit
- Interrupt-driven byte-by-byte data transfer
- Arbitration-lost interrupt with automatic mode switching from master to slave
- Calling address identification interrupt
- START and STOP signal generation and detection
- Repeated START signal generation and detection
- Acknowledge bit generation and detection
- Bus busy detection
- General call recognition

- 10-bit address extension
- Support for *System Management Bus (SMBus) Specification, version 2*
- Programmable glitch input filter
- Low power mode wakeup on slave address match
- Range slave address support
- DMA support

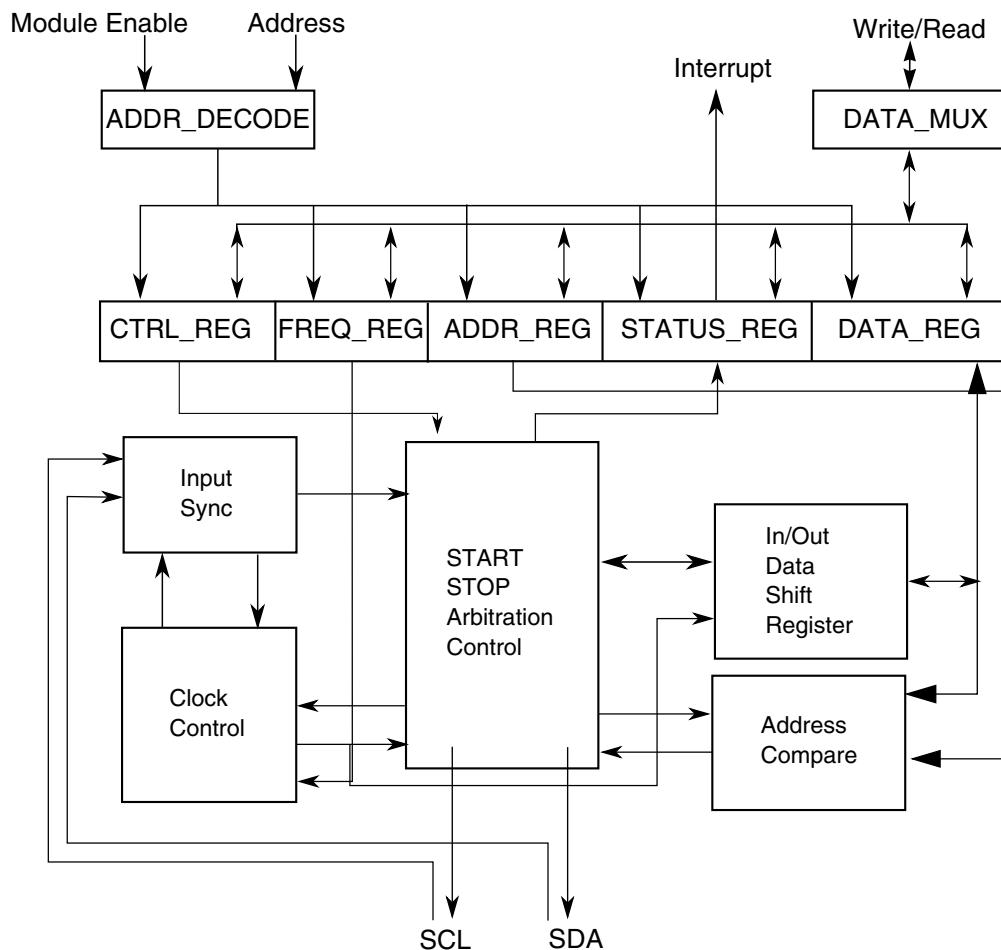
### **38.1.2 Modes of operation**

The I2C module's operation in various low power modes is as follows:

- Run mode: This is the basic mode of operation. To conserve power in this mode, disable the module.
- Wait mode: The module continues to operate when the core is in Wait mode and can provide a wakeup interrupt.
- Stop mode: The module is inactive in Stop mode for reduced power consumption, except that address matching is enabled in Stop mode. The STOP instruction does not affect the I2C module's register states.

### **38.1.3 Block diagram**

The following figure is a functional block diagram of the I2C module.

Figure 38-1. I<sup>2</sup>C Functional block diagram

## 38.2 I<sup>2</sup>C signal descriptions

The signal properties of I<sup>2</sup>C are shown in the following table.

Table 38-1. I<sup>2</sup>C signal descriptions

| Signal | Description   | I/O |
|--------|---|-----|
| SCL    | Bidirectional serial clock line of the I <sup>2</sup> C system. | I/O |
| SDA    | Bidirectional serial data line of the I <sup>2</sup> C system.  | I/O |

## 38.3 Memory map and register descriptions

This section describes in detail all I<sup>2</sup>C registers accessible to the end user.

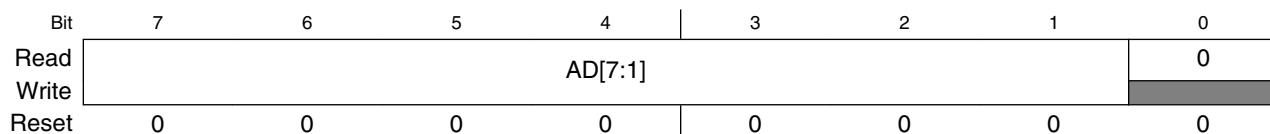
**I2C memory map**

| Absolute address (hex) | Register name  | Width (in bits) | Access | Reset value | Section/ page               |
|------------------------|--|-----------------|--------|-------------|-----------------------------|
| 4006_6000              | I2C Address Register 1 (I2C0_A1)                         | 8               | R/W    | 00h         | <a href="#">38.3.1/690</a>  |
| 4006_6001              | I2C Frequency Divider register (I2C0_F)                  | 8               | R/W    | 00h         | <a href="#">38.3.2/691</a>  |
| 4006_6002              | I2C Control Register 1 (I2C0_C1)                         | 8               | R/W    | 00h         | <a href="#">38.3.3/692</a>  |
| 4006_6003              | I2C Status register (I2C0_S)                             | 8               | R/W    | 80h         | <a href="#">38.3.4/694</a>  |
| 4006_6004              | I2C Data I/O register (I2C0_D)                           | 8               | R/W    | 00h         | <a href="#">38.3.5/695</a>  |
| 4006_6005              | I2C Control Register 2 (I2C0_C2)                         | 8               | R/W    | 00h         | <a href="#">38.3.6/696</a>  |
| 4006_6006              | I2C Programmable Input Glitch Filter register (I2C0_FLT) | 8               | R/W    | 00h         | <a href="#">38.3.7/697</a>  |
| 4006_6007              | I2C Range Address register (I2C0_RA)                     | 8               | R/W    | 00h         | <a href="#">38.3.8/698</a>  |
| 4006_6008              | I2C SMBus Control and Status register (I2C0_SMB)         | 8               | R/W    | 00h         | <a href="#">38.3.9/699</a>  |
| 4006_6009              | I2C Address Register 2 (I2C0_A2)                         | 8               | R/W    | C2h         | <a href="#">38.3.10/701</a> |
| 4006_600A              | I2C SCL Low Timeout Register High (I2C0_SLTH)            | 8               | R/W    | 00h         | <a href="#">38.3.11/701</a> |
| 4006_600B              | I2C SCL Low Timeout Register Low (I2C0_SLTL)             | 8               | R/W    | 00h         | <a href="#">38.3.12/701</a> |
| 4006_7000              | I2C Address Register 1 (I2C1_A1)                         | 8               | R/W    | 00h         | <a href="#">38.3.1/690</a>  |
| 4006_7001              | I2C Frequency Divider register (I2C1_F)                  | 8               | R/W    | 00h         | <a href="#">38.3.2/691</a>  |
| 4006_7002              | I2C Control Register 1 (I2C1_C1)                         | 8               | R/W    | 00h         | <a href="#">38.3.3/692</a>  |
| 4006_7003              | I2C Status register (I2C1_S)                             | 8               | R/W    | 80h         | <a href="#">38.3.4/694</a>  |
| 4006_7004              | I2C Data I/O register (I2C1_D)                           | 8               | R/W    | 00h         | <a href="#">38.3.5/695</a>  |
| 4006_7005              | I2C Control Register 2 (I2C1_C2)                         | 8               | R/W    | 00h         | <a href="#">38.3.6/696</a>  |
| 4006_7006              | I2C Programmable Input Glitch Filter register (I2C1_FLT) | 8               | R/W    | 00h         | <a href="#">38.3.7/697</a>  |
| 4006_7007              | I2C Range Address register (I2C1_RA)                     | 8               | R/W    | 00h         | <a href="#">38.3.8/698</a>  |
| 4006_7008              | I2C SMBus Control and Status register (I2C1_SMB)         | 8               | R/W    | 00h         | <a href="#">38.3.9/699</a>  |
| 4006_7009              | I2C Address Register 2 (I2C1_A2)                         | 8               | R/W    | C2h         | <a href="#">38.3.10/701</a> |
| 4006_700A              | I2C SCL Low Timeout Register High (I2C1_SLTH)            | 8               | R/W    | 00h         | <a href="#">38.3.11/701</a> |
| 4006_700B              | I2C SCL Low Timeout Register Low (I2C1_SLTL)             | 8               | R/W    | 00h         | <a href="#">38.3.12/701</a> |

**38.3.1 I2C Address Register 1 (I2Cx\_A1)**

This register contains the slave address to be used by the I2C module.

Address: Base address + 0h offset



**I2Cx\_A1 field descriptions**

| Field          | Description   |
|----------------|---|
| 7–1<br>AD[7:1] | Address<br><br>Contains the primary slave address used by the I2C module when it is addressed as a slave. This field is used in the 7-bit address scheme and the lower seven bits in the 10-bit address scheme. |
| 0<br>Reserved  | This field is reserved.<br>This read-only field is reserved and always has the value 0.   |

**38.3.2 I2C Frequency Divider register (I2Cx\_F)**

Address: Base address + 1h offset

| Bit        | 7    | 6 | 5 | 4 | 3   | 2 | 1 | 0 |
|------------|------|---|---|---|-----|---|---|---|
| Read Write | MULT |   |   |   | ICR |   |   |   |
| Reset      | 0    | 0 | 0 | 0 | 0   | 0 | 0 | 0 |

**I2Cx\_F field descriptions**

| Field       | Description  |
|-------------|--|
| 7–6<br>MULT | The MULT bits define the multiplier factor mul. This factor is used along with the SCL divider to generate the I2C baud rate.<br><br>00 mul = 1<br>01 mul = 2<br>10 mul = 4<br>11 Reserved   |
| 5–0<br>ICR  | ClockRate<br><br>Prescales the bus clock for bit rate selection. This field and the MULT field determine the I2C baud rate, the SDA hold time, the SCL start hold time, and the SCL stop hold time. For a list of values corresponding to each ICR setting, see <a href="#">I2C divider and hold values</a> .<br><br>The SCL divider multiplied by multiplier factor (mul) determines the I2C baud rate.<br><br>$\text{I2C baud rate} = \text{bus speed (Hz)} / (\text{mul} \times \text{SCL divider})$<br><br>The SDA hold time is the delay from the falling edge of SCL (I2C clock) to the changing of SDA (I2C data).<br><br>$\text{SDA hold time} = \text{bus period (s)} \times \text{mul} \times \text{SDA hold value}$<br><br>The SCL start hold time is the delay from the falling edge of SDA (I2C data) while SCL is high (start condition) to the falling edge of SCL (I2C clock).<br><br>$\text{SCL start hold time} = \text{bus period (s)} \times \text{mul} \times \text{SCL start hold value}$<br><br>The SCL stop hold time is the delay from the rising edge of SCL (I2C clock) to the rising edge of SDA (I2C data) while SCL is high (stop condition).<br><br>$\text{SCL stop hold time} = \text{bus period (s)} \times \text{mul} \times \text{SCL stop hold value}$<br><br>For example, if the bus speed is 8 MHz, the following table shows the possible hold time values with different ICR and MULT selections to achieve an I <sup>2</sup> C baud rate of 100 kbps. |

*Table continues on the next page...*

### I2Cx\_F field descriptions (continued)

| Field | Description |     |                 |           |          |
|-------|-------------|-----|-----------------|-----------|----------|
|       | MULT        | ICR | Hold times (μs) |           |          |
|       |             |     | SDA             | SCL Start | SCL Stop |
|       | 2h          | 00h | 3.500           | 3.000     | 5.500    |
|       | 1h          | 07h | 2.500           | 4.000     | 5.250    |
|       | 1h          | 0Bh | 2.250           | 4.000     | 5.250    |
|       | 0h          | 14h | 2.125           | 4.250     | 5.125    |
|       | 0h          | 18h | 1.125           | 4.750     | 5.125    |

### 38.3.3 I2C Control Register 1 (I2Cx\_C1)

Address: Base address + 2h offset

| Bit   | 7     | 6     | 5   | 4  | 3    | 2    | 1    | 0     |
|-------|-------|-------|-----|----|------|------|------|-------|
| Read  | IICEN | IICIE | MST | TX | TXAK | 0    | WUEN | DMAEN |
| Write | 0     | 0     | 0   | 0  | 0    | RSTA | 0    | 0     |

### I2Cx\_C1 field descriptions

| Field   | Description  |
|---------|--|
| 7 IICEN | I2C Enable<br>Enables I2C module operation.<br><br>0 Disabled<br>1 Enabled                                       |
| 6 IICIE | I2C Interrupt Enable<br>Enables I2C interrupt requests.<br><br>0 Disabled<br>1 Enabled                           |
| 5 MST   | Master Mode Select<br>When the MST bit is changed from a 0 to a 1, a START signal is generated on the bus and mt |

**I2Cx\_C1 field descriptions (continued)**

| Field   | Description  |
|---------|--|
|         | Selects the direction of master and slave transfers. In master mode this bit must be set according to the type of transfer required. Therefore, for address cycles, this bit is always set. When addressed as a slave this bit must be set by software according to the SRW bit in the status register.<br><br>0 Receive<br>1 Transmit   |
| 3 TXAK  | Transmit Acknowledge Enable<br><br>Specifies the value driven onto the SDA during data acknowledge cycles for both master and slave receivers. The value of the FACK bit affects NACK/ACK generation.<br><br><b>NOTE:</b> SCL is held low until TXAK is written.<br><br>0 An acknowledge signal is sent to the bus on the following receiving byte (if FACK is cleared) or the current receiving byte (if FACK is set).<br>1 No acknowledge signal is sent to the bus on the following receiving data byte (if FACK is cleared) or the current receiving data byte (if FACK is set).   |
| 2 RSTA  | Repeat START<br><br>Writing a one to this bit generates a repeated START condition provided it is the current master. This bit will always be read as zero. Attempting a repeat at the wrong time results in loss of arbitration.  |
| 1 WUEN  | Wakeup Enable<br><br>The I2C module can wake the MCU from low power mode with no peripheral bus running when slave address matching occurs.<br><br>0 Normal operation. No interrupt generated when address matching in low power mode.<br>1 Enables the wakeup function in low power mode.   |
| 0 DMAEN | DMA Enable<br><br>The DMAEN bit enables or disables the DMA function.<br><br>0 All DMA signalling disabled.<br>1 DMA transfer is enabled and the following conditions trigger the DMA request: <ul style="list-style-type: none"><li>• While FACK = 0, a data byte is received, either address or data is transmitted. (ACK/NACK automatic)</li><li>• While FACK = 0, the first byte received matches the A1 register or is general call address.</li></ul><br>If any address matching occurs, IAAS and TCF are set. If the direction of transfer is known from master to slave, then it is not required to check the SRW. With this assumption, DMA can also be used in this case. In other cases, if the master reads data from the slave, then it is required to rewrite the C1 register operation. With this assumption, DMA cannot be used.<br><br>When FACK = 1, an address or a data byte is transmitted. |

### 38.3.4 I2C Status register (I2Cx\_S)

Address: Base address + 3h offset

| Bit   | 7   | 6    | 5    | 4    | 3   | 2   | 1     | 0    |
|-------|-----|------|------|------|-----|-----|-------|------|
| Read  | TCF | IAAS | BUSY | ARBL | RAM | SRW | IICIF | RXAK |
| Write |     |      |      | w1c  |     |     | w1c   |      |
| Reset | 1   | 0    | 0    | 0    | 0   | 0   | 0     | 0    |

#### I2Cx\_S field descriptions

| Field     | Description  |
|-----------|--|
| 7<br>TCF  | <p>Transfer Complete Flag</p> <p>This bit sets on the completion of a byte and acknowledge bit transfer. This bit is valid only during or immediately following a transfer to or from the I2C module. The TCF bit is cleared by reading the I2C data register in receive mode or by writing to the I2C data register in transmit mode.</p> <p>0 Transfer in progress<br/>1 Transfer complete</p>   |
| 6<br>IAAS | <p>Addressed As A Slave</p> <p>This bit is set by one of the following conditions:</p> <ul style="list-style-type: none"> <li>The calling address matches the programmed slave primary address in the A1 register or range address in the RA register (which must be set to a nonzero value).</li> <li>GCAEN is set and a general call is received.</li> <li>SIICAEN is set and the calling address matches the second programmed slave address.</li> <li>ALERTEN is set and an SMBus alert response address is received</li> <li>RMEN is set and an address is received that is within the range between the values of the A1 and RA registers.</li> </ul> <p>This bit sets before the ACK bit. The CPU must check the SRW bit and set TX/RX accordingly. Writing the C1 register with any value clears this bit.</p> <p>0 Not addressed<br/>1 Addressed as a slave</p> |
| 5<br>BUSY | <p>Bus Busy</p> <p>Indicates the status of the bus regardless of slave or master mode. This bit is set when a START signal is detected and cleared when a STOP signal is detected.</p> <p>0 Bus is idle<br/>1 Bus is busy</p>  |
| 4<br>ARBL | <p>Arbitration Lost</p> <p>This bit is set by hardware when the arbitration procedure is lost. The ARBL bit must be cleared by software, by writing a one to it.</p> <p>0 Standard bus operation.<br/>1 Loss of arbitration.</p>   |
| 3<br>RAM  | <p>Range Address Match</p> <p>This bit is set to 1 by any of the following conditions:</p>   |

*Table continues on the next page...*

**I2Cx\_S field descriptions (continued)**

| Field   | Description  |
|---------|--|
|         | <ul style="list-style-type: none"> <li>Any nonzero calling address is received that matches the address in the RA register.</li> <li>The RMEN bit is set and the calling address is within the range of values of the A1 and RA registers.</li> </ul> <p><b>NOTE:</b> For the RAM bit to be set to 1 correctly, C1[IICIE] must be set to 1.<br/>Writing the C1 register with any value clears this bit to 0.</p> <p>0 Not addressed<br/>1 Addressed as a slave</p>   |
| 2 SRW   | <p>Slave Read/Write</p> <p>When addressed as a slave, SRW indicates the value of the R/W command bit of the calling address sent to the master.</p> <p>0 Slave receive, master writing to slave<br/>1 Slave transmit, master reading from slave</p>  |
| 1 IICIF | <p>Interrupt Flag</p> <p>This bit sets when an interrupt is pending. This bit must be cleared by software by writing a 1 to it, such as in the interrupt routine. One of the following events can set this bit:</p> <ul style="list-style-type: none"> <li>One byte transfer, including ACK/NACK bit, completes if FACK is 0. An ACK or NACK is sent on the bus by writing 0 or 1 to TXAK after this bit is set in receive mode.</li> <li>One byte transfer, excluding ACK/NACK bit, completes if FACK is 1.</li> <li>Match of slave address to calling address including primary slave address, range slave address, alert response address, second slave address, or general call address.</li> <li>Arbitration lost</li> <li>In SMBus mode, any timeouts except SCL and SDA high timeouts</li> <li>I2C bus stop detection if the STOPIE bit in the Input Glitch Filter register is 1</li> </ul> <p><b>NOTE:</b> To clear the I2C bus stop detection interrupt: In the interrupt service routine, first clear the STOPF bit in the Input Glitch Filter register by writing 1 to it, and then clear the IICIF bit. If this sequence is reversed, the IICIF bit is asserted again.</p> <p>0 No interrupt pending<br/>1 Interrupt pending</p> |
| 0 RXAK  | <p>Receive Acknowledge</p> <p>0 Acknowledge signal was received after the completion of one byte of data transmission on the bus<br/>1 No acknowledge signal detected</p>  |

**38.3.5 I2C Data I/O register (I2Cx\_D)**

Address: Base address + 4h offset

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

**I2Cx\_D field descriptions**

| Field       | Description   |
|-------------|---|
| 7–0<br>DATA | <p>Data</p> <p>In master transmit mode, when data is written to this register, a data transfer is initiated. The most significant bit is sent first. In master receive mode, reading this register initiates receiving of the next byte of data.</p> <p><b>NOTE:</b> When making the transition out of master receive mode, switch the I2C mode before reading the Data register to prevent an inadvertent initiation of a master receive data transfer.</p> <p>In slave mode, the same functions are available after an address match occurs.</p> <p>The C1[TX] bit must correctly reflect the desired direction of transfer in master and slave modes for the transmission to begin. For example, if the I2C module is configured for master transmit but a master receive is desired, reading the Data register does not initiate the receive.</p> <p>Reading the Data register returns the last byte received while the I2C module is configured in master receive or slave receive mode. The Data register does not reflect every byte that is transmitted on the I2C bus, and neither can software verify that a byte has been written to the Data register correctly by reading it back.</p> <p>In master transmit mode, the first byte of data written to the Data register following assertion of MST (start bit) or assertion of RSTA (repeated start bit) is used for the address transfer and must consist of the calling address (in bits 7–1) concatenated with the required R/W bit (in position bit 0).</p> |

**38.3.6 I2C Control Register 2 (I2Cx\_C2)**

Address: Base address + 5h offset

| Bit        | 7     | 6     | 5    | 4    | 3    | 2 | 1        | 0 |
|------------|-------|-------|------|------|------|---|----------|---|
| Read Write | GCAEN | ADEXT | HDRS | SBRC | RMEN |   | AD[10:8] |   |
| Reset      | 0     | 0     | 0    | 0    | 0    | 0 | 0        | 0 |

**I2Cx\_C2 field descriptions**

| Field      | Description   |
|------------|---|
| 7<br>GCAEN | <p>General Call Address Enable</p> <p>Enables general call address.</p> <p>0 Disabled<br/>1 Enabled</p>   |
| 6<br>ADEXT | <p>Address Extension</p> <p>Controls the number of bits used for the slave address.</p> <p>0 7-bit address scheme<br/>1 10-bit address scheme</p> |
| 5<br>HDRS  | <p>High Drive Select</p> <p>Controls the drive capability of the I2C pads.</p>  |

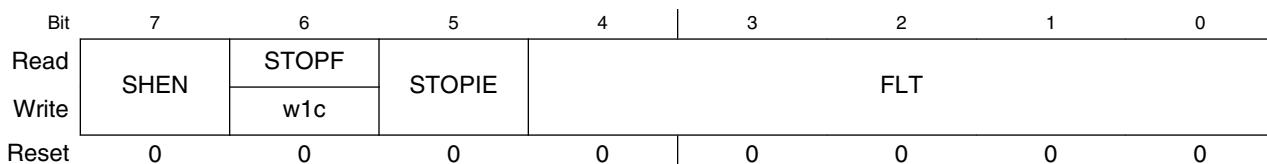
*Table continues on the next page...*

**I2Cx\_C2 field descriptions (continued)**

| Field        | Description   |
|--------------|---|
|              | 0 Normal drive mode<br>1 High drive mode  |
| 4 SBRC       | <b>Slave Baud Rate Control</b><br><p>Enables independent slave mode baud rate at maximum frequency, which forces clock stretching on SCL in very fast I2C modes. To a slave, an example of a "very fast" mode is when the master transfers at 40 kbps but the slave can capture the master's data at only 10 kbps.</p> 0 The slave baud rate follows the master baud rate and clock stretching may occur<br>1 Slave baud rate is independent of the master baud rate  |
| 3 RMEN       | <b>Range Address Matching Enable</b><br><p>This bit controls slave address matching for addresses between the values of the A1 and RA registers. When this bit is set, a slave address match occurs for any address greater than the value of the A1 register and less than or equal to the value of the RA register.</p> 0 Range mode disabled. No address match occurs for an address within the range of values of the A1 and RA registers.<br>1 Range mode enabled. Address matching occurs when a slave receives an address within the range of values of the A1 and RA registers. |
| 2–0 AD[10:8] | <b>Slave Address</b><br><p>Contains the upper three bits of the slave address in the 10-bit address scheme. This field is valid only while the ADEXT bit is set.</p>  |

**38.3.7 I2C Programmable Input Glitch Filter register (I2Cx\_FLT)**

Address: Base address + 6h offset

**I2Cx\_FLT field descriptions**

| Field  | Description  |
|--------|--|
| 7 SHEN | <b>Stop Hold Enable</b><br><p>Set this bit to hold off entry to stop mode when any data transmission or reception is occurring. The following scenario explains the holdoff functionality:</p> <ol style="list-style-type: none"> <li>1. The I2C module is configured for a basic transfer, and the SHEN bit is set to 1.</li> <li>2. A transfer begins.</li> <li>3. The MCU signals the I2C module to enter stop mode.</li> <li>4. The byte currently being transferred, including both address and data, completes its transfer.</li> <li>5. The I2C slave or master acknowledges that the in-transfer byte completed its transfer and acknowledges the request to enter stop mode.</li> <li>6. After receiving the I2C module's acknowledgment of the request to enter stop mode, the MCU determines whether to shut off the I2C module's clock.</li> </ol> |

*Table continues on the next page...*

**I2Cx\_FLT field descriptions (continued)**

| Field       | Description   |
|-------------|---|
|             | If the SHEN bit is set to 1 and the I2C module is in an idle or disabled state when the MCU signals to enter stop mode, the module immediately acknowledges the request to enter stop mode.<br><br>If SHEN is cleared to 0 and the overall data transmission or reception that was suspended by stop mode entry was incomplete: To resume the overall transmission or reception after the MCU exits stop mode, software must reinitialize the transfer by resending the address of the slave.<br><br>If the I2C Control Register 1's IICIE bit was set to 1 before the MCU entered stop mode, system software will receive the interrupt triggered by the I2C Status Register's TCF bit after the MCU wakes from the stop mode.<br><br>0 Stop holdoff is disabled. The MCU's entry to stop mode is not gated.<br>1 Stop holdoff is enabled. |
| 6<br>STOPF  | I2C Bus Stop Detect Flag<br><br>Hardware sets this bit when the I2C bus's stop status is detected. The STOPF bit must be cleared by writing 1 to it.<br><br>0 No stop happens on I2C bus<br>1 Stop detected on I2C bus  |
| 5<br>STOPIE | I2C Bus Stop Interrupt Enable<br><br>This bit enables the interrupt for I2C bus stop detection.<br><br><b>NOTE:</b> To clear the I2C bus stop detection interrupt: In the interrupt service routine, first clear the STOPF bit by writing 1 to it, and then clear the IICIF bit in the status register. If this sequence is reversed, the IICIF bit is asserted again.<br><br>0 Stop detection interrupt is disabled<br>1 Stop detection interrupt is enabled   |
| 4–0<br>FLT  | I2C Programmable Filter Factor<br><br>Controls the width of the glitch, in terms of bus clock cycles, that the filter must absorb. For any glitch whose size is less than or equal to this width setting, the filter does not allow the glitch to pass.<br><br>00h No filter/bypass<br>01-1Fh Filter glitches up to width of $n$ bus clock cycles, where $n=1-31d$  |

**38.3.8 I2C Range Address register (I2Cx\_RA)**

Address: Base address + 7h offset

| Bit   | 7 | 6 | 5 | 4   | 3 | 2 | 1 | 0 |
|-------|---|---|---|-----|---|---|---|---|
| Read  |   |   |   | RAD |   |   |   | 0 |
| Write |   |   |   |     |   |   |   |   |
| Reset | 0 | 0 | 0 | 0   | 0 | 0 | 0 | 0 |

**I2Cx\_RA field descriptions**

| Field      | Description         |
|------------|---------------------|
| 7–1<br>RAD | Range Slave Address |

Table continues on the next page...

**I2Cx\_RA field descriptions (continued)**

| Field         | Description   |
|---------------|---|
|               | This field contains the slave address to be used by the I2C module. The field is used in the 7-bit address scheme. Any nonzero write enables this register. This register's use is similar to that of the A1 register, but in addition this register can be considered a maximum boundary in range matching mode. |
| 0<br>Reserved | This field is reserved.<br>This read-only field is reserved and always has the value 0.   |

**38.3.9 I2C SMBus Control and Status register (I2Cx\_SMB)****NOTE**

When the SCL and SDA signals are held high for a length of time greater than the high timeout period, the SHTF1 flag sets. Before reaching this threshold, while the system is detecting how long these signals are being held high, a master assumes that the bus is free. However, the SHTF1 bit rises in the bus transmission process with the idle bus state.

**NOTE**

When the TCKSEL bit is set, there is no need to monitor the SHTF1 bit because the bus speed is too high to match the protocol of SMBus.

Address: Base address + 8h offset

| Bit   | 7    | 6       | 5       | 4      | 3    | 2     | 1     | 0       |
|-------|------|---------|---------|--------|------|-------|-------|---------|
| Read  | FACK | ALERTEN | SIICAEN | TCKSEL | SLTF | SHTF1 | SHTF2 | SHTF2IE |
| Write |      |         |         |        | w1c  |       | w1c   |         |
| Reset | 0    | 0       | 0       | 0      | 0    | 0     | 0     | 0       |

**I2Cx\_SMB field descriptions**

| Field        | Description   |
|--------------|---|
| 7<br>FACK    | Fast NACK/ACK Enable<br><br>For SMBus packet error checking, the CPU must be able to issue an ACK or NACK according to the result of receiving data byte.<br><br>0 An ACK or NACK is sent on the following receiving data byte<br>1 Writing 0 to TXAK after receiving a data byte generates an ACK. Writing 1 to TXAK after receiving a data byte generates a NACK. |
| 6<br>ALERTEN | SMBus Alert Response Address Enable<br><br>Enables or disables SMBus alert response address matching.<br><br><b>NOTE:</b> After the host responds to a device that used the alert response address, you must use software to put the device's address on the bus. The alert protocol is described in the SMBus specification.                                       |

*Table continues on the next page...*

## I2Cx\_SMB field descriptions (continued)

| Field     | Description  |
|-----------|--|
|           | <p>0 SMBus alert response address matching is disabled<br/>1 SMBus alert response address matching is enabled</p>  |
| 5 SIICAEN | <p>Second I2C Address Enable</p> <p>Enables or disables SMBus device default address.</p> <p>0 I2C address register 2 matching is disabled<br/>1 I2C address register 2 matching is enabled</p>  |
| 4 TCKSEL  | <p>Timeout Counter Clock Select</p> <p>Selects the clock source of the timeout counter.</p> <p>0 Timeout counter counts at the frequency of the bus clock / 64<br/>1 Timeout counter counts at the frequency of the bus clock</p>  |
| 3 SLTF    | <p>SCL Low Timeout Flag</p> <p>This bit is set when the SLT register (consisting of the SLTH and SLTL registers) is loaded with a non-zero value (LoValue) and an SCL low timeout occurs. Software clears this bit by writing a logic 1 to it.</p> <p><b>NOTE:</b> The low timeout function is disabled when the SLT register's value is zero.</p> <p>0 No low timeout occurs<br/>1 Low timeout occurs</p> |
| 2 SHTF1   | <p>SCL High Timeout Flag 1</p> <p>This read-only bit sets when SCL and SDA are held high more than <math>\text{clock} \times \text{LoValue} / 512</math>, which indicates the bus is free. This bit is cleared automatically.</p> <p>0 No SCL high and SDA high timeout occurs<br/>1 SCL high and SDA high timeout occurs</p>  |
| 1 SHTF2   | <p>SCL High Timeout Flag 2</p> <p>This bit sets when SCL is held high and SDA is held low more than <math>\text{clock} \times \text{LoValue}/512</math>. Software clears this bit by writing a 1 to it.</p> <p>0 No SCL high and SDA low timeout occurs<br/>1 SCL high and SDA low timeout occurs</p>  |
| 0 SHTF2IE | <p>SHTF2 Interrupt Enable</p> <p>Enables SCL high and SDA low timeout interrupt.</p> <p>0 SHTF2 interrupt is disabled<br/>1 SHTF2 interrupt is enabled</p>   |

### 38.3.10 I2C Address Register 2 (I2Cx\_A2)

Address: Base address + 9h offset

| Bit   | 7   | 6 | 5 | 4 |  | 3 | 2 | 1 | 0 |
|-------|-----|---|---|---|--|---|---|---|---|
| Read  | SAD |   |   |   |  |   |   | 0 |   |
| Write |     |   |   |   |  |   |   | 0 |   |
| Reset | 1   | 1 | 0 | 0 |  | 0 | 0 | 1 | 0 |

#### I2Cx\_A2 field descriptions

| Field         | Description   |
|---------------|---|
| 7–1<br>SAD    | SMBus Address<br><br>Contains the slave address used by the SMBus. This field is used on the device default address or other related addresses. |
| 0<br>Reserved | This field is reserved.<br>This read-only field is reserved and always has the value 0.   |

### 38.3.11 I2C SCL Low Timeout Register High (I2Cx\_SLTH)

Address: Base address + Ah offset

| Bit   | 7          | 6 | 5 | 4 |  | 3 | 2 | 1 | 0 |
|-------|------------|---|---|---|--|---|---|---|---|
| Read  | SSLT[15:8] |   |   |   |  |   |   | 0 |   |
| Write |            |   |   |   |  |   |   | 0 |   |
| Reset | 0          | 0 | 0 | 0 |  | 0 | 0 | 0 | 0 |

#### I2Cx\_SLTH field descriptions

| Field             | Description   |
|-------------------|---|
| 7–0<br>SSLT[15:8] | Most significant byte of SCL low timeout value that determines the timeout period of SCL low. |

### 38.3.12 I2C SCL Low Timeout Register Low (I2Cx\_SLTL)

Address: Base address + Bh offset

| Bit   | 7         | 6 | 5 | 4 |  | 3 | 2 | 1 | 0 |
|-------|-----------|---|---|---|--|---|---|---|---|
| Read  | SSLT[7:0] |   |   |   |  |   |   | 0 |   |
| Write |           |   |   |   |  |   |   | 0 |   |
| Reset | 0         | 0 | 0 | 0 |  | 0 | 0 | 0 | 0 |

#### I2Cx\_SLTL field descriptions

| Field            | Description  |
|------------------|--|
| 7–0<br>SSLT[7:0] | Least significant byte of SCL low timeout value that determines the timeout period of SCL low. |

## 38.4 Functional description

This section provides a comprehensive functional description of the I2C module.

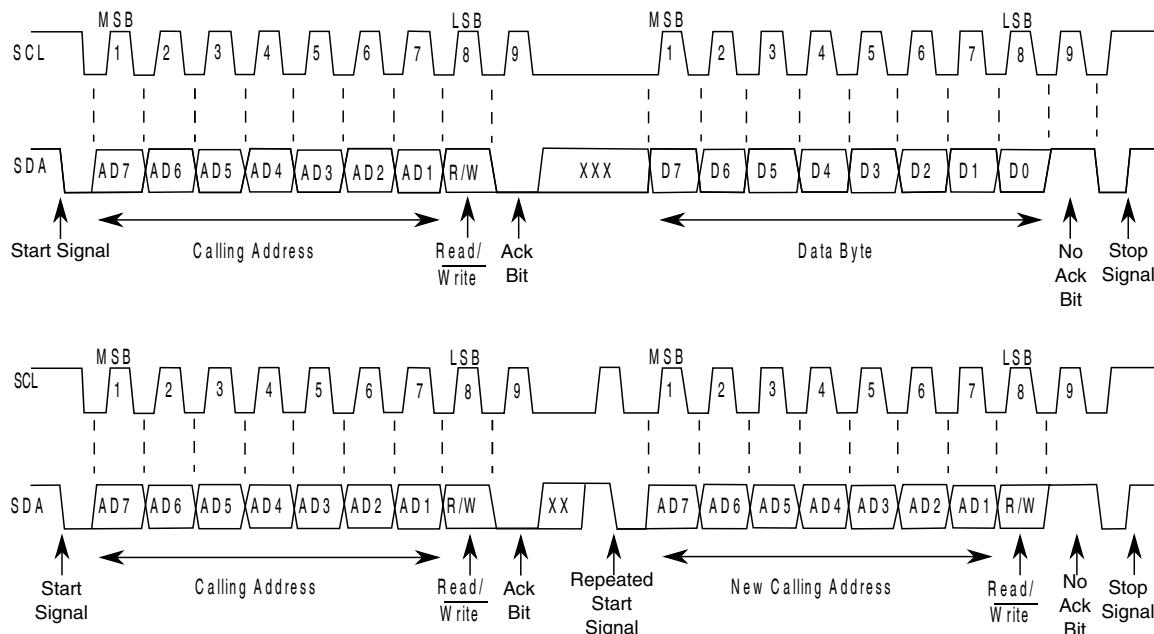
### 38.4.1 I2C protocol

The I2C bus system uses a serial data line (SDA) and a serial clock line (SCL) for data transfers. All devices connected to it must have open drain or open collector outputs. A logic AND function is exercised on both lines with external pull-up resistors. The value of these resistors depends on the system.

Normally, a standard instance of communication is composed of four parts:

1. START signal
2. Slave address transmission
3. Data transfer
4. STOP signal

The STOP signal should not be confused with the CPU STOP instruction. The following figure illustrates I2C bus system communication.



**Figure 38-38. I2C bus transmission signals**

### 38.4.1.1 START signal

The bus is free when no master device is engaging the bus (both SCL and SDA are high). When the bus is free, a master may initiate communication by sending a START signal. A START signal is defined as a high-to-low transition of SDA while SCL is high. This signal denotes the beginning of a new data transfer—each data transfer might contain several bytes of data—and brings all slaves out of their idle states.

### 38.4.1.2 Slave address transmission

Immediately after the START signal, the first byte of a data transfer is the slave address transmitted by the master. This address is a 7-bit calling address followed by an R/W bit. The R/W bit tells the slave the desired direction of data transfer.

- 1 = Read transfer: The slave transmits data to the master
- 0 = Write transfer: The master transmits data to the slave

Only the slave with a calling address that matches the one transmitted by the master responds by sending an acknowledge bit. The slave sends the acknowledge bit by pulling SDA low at the ninth clock.

No two slaves in the system can have the same address. If the I2C module is the master, it must not transmit an address that is equal to its own slave address. The I2C module cannot be master and slave at the same time. However, if arbitration is lost during an address cycle, the I2C module reverts to slave mode and operates correctly even if it is being addressed by another master.

### 38.4.1.3 Data transfers

When successful slave addressing is achieved, data transfer can proceed on a byte-by-byte basis in the direction specified by the R/W bit sent by the calling master.

All transfers that follow an address cycle are referred to as data transfers, even if they carry subaddress information for the slave device.

Each data byte is 8 bits long. Data may be changed only while SCL is low. Data must be held stable while SCL is high. There is one clock pulse on SCL for each data bit, and the MSB is transferred first. Each data byte is followed by a ninth (acknowledge) bit, which is signaled from the receiving device by pulling SDA low at the ninth clock. In summary, one complete data transfer needs nine clock pulses.

If the slave receiver does not acknowledge the master in the ninth bit, the slave must leave SDA high. The master interprets the failed acknowledgement as an unsuccessful data transfer.

If the master receiver does not acknowledge the slave transmitter after a data byte transmission, the slave interprets it as an end to data transfer and releases the SDA line.

In the case of a failed acknowledgement by either the slave or master, the data transfer is aborted and the master does one of two things:

- Relinquishes the bus by generating a STOP signal.
- Commences a new call by generating a repeated START signal.

#### **38.4.1.4 STOP signal**

The master can terminate the communication by generating a STOP signal to free the bus. A STOP signal is defined as a low-to-high transition of SDA while SCL is asserted.

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

#### **38.4.1.5 Repeated START signal**

The master may generate a START signal followed by a calling command without generating a STOP signal first. This action is called a repeated START. The master uses a repeated START to communicate with another slave or with the same slave in a different mode (transmit/receive mode) without releasing the bus.

#### **38.4.1.6 Arbitration procedure**

The I<sup>2</sup>C bus is a true multimaster bus that allows more than one master to be connected on it.

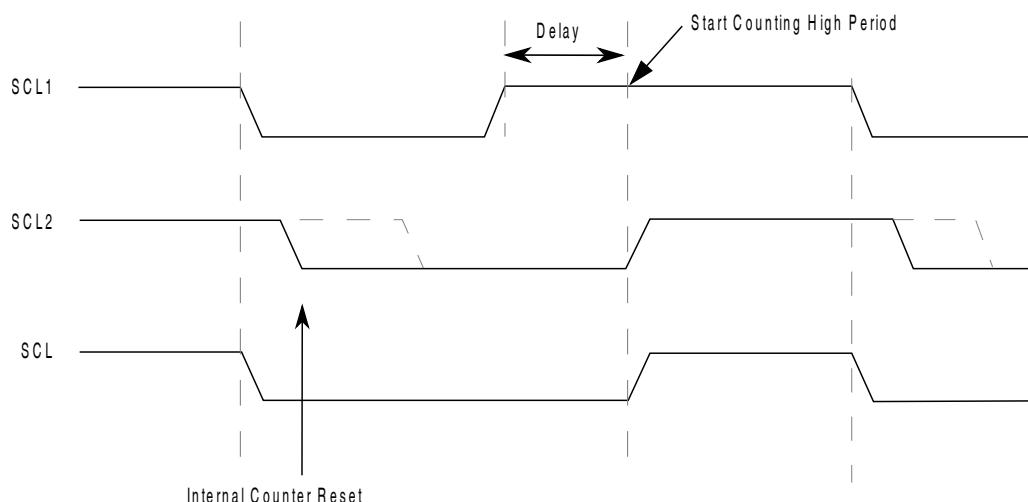
If two or more masters try to control the bus at the same time, a clock synchronization procedure determines the bus clock. The bus clock's low period is equal to the longest clock low period, and the high period is equal to the shortest one among the masters.

The relative priority of the contending masters is determined by a data arbitration procedure. A bus master loses arbitration if it transmits logic level 1 while another master transmits logic level 0. The losing masters immediately switch to slave receive mode and

stop driving SDA output. In this case, the transition from master to slave mode does not generate a STOP condition. Meanwhile, hardware sets a status bit to indicate the loss of arbitration.

### 38.4.1.7 Clock synchronization

Because wire AND logic is performed on SCL, a high-to-low transition on SCL affects all devices connected on the bus. The devices start counting their low period and, after a device's clock has gone low, that device holds SCL low until the clock reaches its high state. However, the change of low to high in this device clock might not change the state of SCL if another device clock is still within its low period. Therefore, the synchronized clock SCL is held low by the device with the longest low period. Devices with shorter low periods enter a high wait state during this time; see the following diagram. When all applicable devices have counted off their low period, the synchronized clock SCL is released and pulled high. Afterward there is no difference between the device clocks and the state of SCL, and all devices start counting their high periods. The first device to complete its high period pulls SCL low again.



**Figure 38-39. I2C clock synchronization**

### 38.4.1.8 Handshaking

The clock synchronization mechanism can be used as a handshake in data transfers. A slave device may hold SCL low after completing a single byte transfer (9 bits). In this case, it halts the bus clock and forces the master clock into wait states until the slave releases SCL.

### 38.4.1.9 Clock stretching

The clock synchronization mechanism can be used by slaves to slow down the bit rate of a transfer. After the master drives SCL low, a slave can drive SCL low for the required period and then release it. If the slave's SCL low period is greater than the master's SCL low period, the resulting SCL bus signal's low period is stretched. In other words, the SCL bus signal's low period is increased to be the same length as the slave's SCL low period.

### 38.4.1.10 I<sub>2</sub>C divider and hold values

#### NOTE

For some cases on some devices, the SCL divider value may vary by +/-2 or +/-4 when ICR's value ranges from 00h to 0Fh.

These potentially varying SCL divider values are highlighted in the following table. For the actual SCL divider values for your device, see the chip-specific details about the I<sub>2</sub>C module.

**Table 38-41. I<sub>2</sub>C divider and hold values**

| ICR<br>(hex) | SCL<br>divider | SDA hold<br>value | SCL hold<br>(start)<br>value | SCL hold<br>(stop)<br>value | ICR<br>(hex) | SCL<br>divider<br>(clocks) | SDA hold<br>(clocks) | SCL hold<br>(start)<br>value | SCL hold<br>(stop)<br>value |
|--------------|----------------|-------------------|------------------------------|-----------------------------|--------------|----------------------------|----------------------|------------------------------|-----------------------------|
| 00           | 20             | 7                 | 6                            | 11                          | 20           | 160                        | 17                   | 78                           | 81                          |
| 01           | 22             | 7                 | 7                            | 12                          | 21           | 192                        | 17                   | 94                           | 97                          |
| 02           | 24             | 8                 | 8                            | 13                          | 22           | 224                        | 33                   | 110                          | 113                         |
| 03           | 26             | 8                 | 9                            | 14                          | 23           | 256                        | 33                   | 126                          | 129                         |
| 04           | 28             | 9                 | 10                           | 15                          | 24           | 288                        | 49                   | 142                          | 145                         |
| 05           | 30             | 9                 | 11                           | 16                          | 25           | 320                        | 49                   | 158                          | 161                         |
| 06           | 34             | 10                | 13                           | 18                          | 26           | 384                        | 65                   | 190                          | 193                         |
| 07           | 40             | 10                | 16                           | 21                          | 27           | 480                        | 65                   | 238                          | 241                         |
| 08           | 28             | 7                 | 10                           | 15                          | 28           | 320                        | 33                   | 158                          | 161                         |
| 09           | 32             | 7                 | 12                           | 17                          | 29           | 384                        | 33                   | 190                          | 193                         |
| 0A           | 36             | 9                 | 14                           | 19                          | 2A           | 448                        | 65                   | 222                          | 225                         |
| 0B           | 40             | 9                 | 16                           | 21                          | 2B           | 512                        | 65                   | 254                          | 257                         |
| 0C           | 44             | 11                | 18                           | 23                          | 2C           | 576                        | 97                   | 286                          | 289                         |
| 0D           | 48             | 11                | 20                           | 25                          | 2D           | 640                        | 97                   | 318                          | 321                         |
| 0E           | 56             | 13                | 24                           | 29                          | 2E           | 768                        | 129                  | 382                          | 385                         |
| 0F           | 68             | 13                | 30                           | 35                          | 2F           | 960                        | 129                  | 478                          | 481                         |
| 10           | 48             | 9                 | 18                           | 25                          | 30           | 640                        | 65                   | 318                          | 321                         |
| 11           | 56             | 9                 | 22                           | 29                          | 31           | 768                        | 65                   | 382                          | 385                         |
| 12           | 64             | 13                | 26                           | 33                          | 32           | 896                        | 129                  | 446                          | 449                         |
| 13           | 72             | 13                | 30                           | 37                          | 33           | 1024                       | 129                  | 510                          | 513                         |

Table continues on the next page...

**Table 38-41. I2C divider and hold values (continued)**

| ICR<br>(hex) | SCL<br>divider | SDA hold<br>value | SCL hold<br>(start)<br>value | SCL hold<br>(stop)<br>value | ICR<br>(hex) | SCL<br>divider<br>(clocks) | SDA hold<br>(clocks) | SCL hold<br>(start)<br>value | SCL hold<br>(stop)<br>value |
|--------------|----------------|-------------------|------------------------------|-----------------------------|--------------|----------------------------|----------------------|------------------------------|-----------------------------|
| 14           | 80             | 17                | 34                           | 41                          | 34           | 1152                       | 193                  | 574                          | 577                         |
| 15           | 88             | 17                | 38                           | 45                          | 35           | 1280                       | 193                  | 638                          | 641                         |
| 16           | 104            | 21                | 46                           | 53                          | 36           | 1536                       | 257                  | 766                          | 769                         |
| 17           | 128            | 21                | 58                           | 65                          | 37           | 1920                       | 257                  | 958                          | 961                         |
| 18           | 80             | 9                 | 38                           | 41                          | 38           | 1280                       | 129                  | 638                          | 641                         |
| 19           | 96             | 9                 | 46                           | 49                          | 39           | 1536                       | 129                  | 766                          | 769                         |
| 1A           | 112            | 17                | 54                           | 57                          | 3A           | 1792                       | 257                  | 894                          | 897                         |
| 1B           | 128            | 17                | 62                           | 65                          | 3B           | 2048                       | 257                  | 1022                         | 1025                        |
| 1C           | 144            | 25                | 70                           | 73                          | 3C           | 2304                       | 385                  | 1150                         | 1153                        |
| 1D           | 160            | 25                | 78                           | 81                          | 3D           | 2560                       | 385                  | 1278                         | 1281                        |
| 1E           | 192            | 33                | 94                           | 97                          | 3E           | 3072                       | 513                  | 1534                         | 1537                        |
| 1F           | 240            | 33                | 118                          | 121                         | 3F           | 3840                       | 513                  | 1918                         | 1921                        |

## 38.4.2 10-bit address

For 10-bit addressing, 0x11110 is used for the first 5 bits of the first address byte. Various combinations of read/write formats are possible within a transfer that includes 10-bit addressing.

### 38.4.2.1 Master-transmitter addresses a slave-receiver

The transfer direction is not changed. When a 10-bit address follows a START condition, each slave compares the first 7 bits of the first byte of the slave address (11110XX) with its own address and tests whether the eighth bit ( $R/\bar{W}$  direction bit) is 0. It is possible that more than one device finds a match and generates an acknowledge (A1). Each slave that finds a match compares the 8 bits of the second byte of the slave address with its own address, but only one slave finds a match and generates an acknowledge (A2). The matching slave remains addressed by the master until it receives a STOP condition (P) or a repeated START condition (Sr) followed by a different slave address.

**Table 38-42. Master-transmitter addresses slave-receiver with a 10-bit address**

| S | Slave address first 7 bits<br>11110 + AD10 + AD9 | R/W 0 | A1 | Slave address second byte<br>AD[8:1] | A2 | Data | A | ... | Data | A/A | P |
|---|--|-------|----|--------------------------------------|----|------|---|-----|------|-----|---|
|---|--|-------|----|--------------------------------------|----|------|---|-----|------|-----|---|

After the master-transmitter has sent the first byte of the 10-bit address, the slave-receiver sees an I2C interrupt. User software must ensure that for this interrupt, the contents of the Data register are ignored and not treated as valid data.

### 38.4.2.2 Master-receiver addresses a slave-transmitter

The transfer direction is changed after the second R/W bit. Up to and including acknowledge bit A2, the procedure is the same as that described for a master-transmitter addressing a slave-receiver. After the repeated START condition (Sr), a matching slave remembers that it was addressed before. This slave then checks whether the first seven bits of the first byte of the slave address following Sr are the same as they were after the START condition (S), and it tests whether the eighth (R/W) bit is 1. If there is a match, the slave considers that it has been addressed as a transmitter and generates acknowledge A3. The slave-transmitter remains addressed until it receives a STOP condition (P) or a repeated START condition (Sr) followed by a different slave address.

After a repeated START condition (Sr), all other slave devices also compare the first seven bits of the first byte of the slave address with their own addresses and test the eighth (R/W) bit. However, none of them are addressed because R/W = 1 (for 10-bit devices), or the 11110XX slave address (for 7-bit devices) does not match.

**Table 38-43. Master-receiver addresses a slave-transmitter with a 10-bit address**

| S | Slave address first 7 bits<br>11110 + AD10 + AD9 | R/W 0 | A1 | Slave address second byte<br>AD[8:1] | A2 | Sr | Slave address first 7 bits<br>11110 + AD10 + AD9 | R/W 1 | A3 | Data | A | ... | Data | A | P |
|---|--|-------|----|--------------------------------------|----|----|--|-------|----|------|---|-----|------|---|---|
|---|--|-------|----|--------------------------------------|----|----|--|-------|----|------|---|-----|------|---|---|

After the master-receiver has sent the first byte of the 10-bit address, the slave-transmitter sees an I2C interrupt. User software must ensure that for this interrupt, the contents of the Data register are ignored and not treated as valid data.

### 38.4.3 Address matching

All received addresses can be requested in 7-bit or 10-bit address format.

- AD[7:1] in Address Register 1, which contains the I2C primary slave address, always participates in the address matching process. It provides a 7-bit address.
- If the ADEXT bit is set, AD[10:8] in Control Register 2 participates in the address matching process. It extends the I2C primary slave address to a 10-bit address.

Additional conditions that affect address matching include:

- If the GCAEN bit is set, general call participates the address matching process.
- If the ALERTEN bit is set, alert response participates the address matching process.
- If the SIICAEN bit is set, Address Register 2 participates in the address matching process.
- If the Range Address register is programmed to a nonzero value, the range address itself participates in the address matching process.
- If the RMEN bit is set, any address within the range of values of Address Register 1 and the Range Address register participates in the address matching process. The Range Address register must be programmed to a value greater than the value of Address Register 1.

When the I2C module responds to one of these addresses, it acts as a slave-receiver and the IAAS bit is set after the address cycle. Software must read the Data register after the first byte transfer to determine that the address is matched.

### 38.4.4 System management bus specification

SMBus provides a control bus for system and power management related tasks. A system can use SMBus to pass messages to and from devices instead of tripping individual control lines. Removing the individual control lines reduces pin count. Accepting messages ensures future expandability. With the system management bus, a device can provide manufacturer information, tell the system what its model/part number is, save its state for a suspend event, report different types of errors, accept control parameters, and return its status.

#### 38.4.4.1 Timeouts

The  $T_{TIMEOUT,MIN}$  parameter allows a master or slave to conclude that a defective device is holding the clock low indefinitely or a master is intentionally trying to drive devices off the bus. The slave device must release the bus (stop driving the bus and let SCL and

## Functional description

SDA float high) when it detects any single clock held low longer than  $T_{TIMEOUT,MIN}$ . Devices that have detected this condition must reset their communication and be able to receive a new START condition within the timeframe of  $T_{TIMEOUT,MAX}$ .

SMBus defines a clock low timeout,  $T_{TIMEOUT}$ , of 35 ms, specifies  $T_{LOW:SEXT}$  as the cumulative clock low extend time for a slave device, and specifies  $T_{LOW:MEXT}$  as the cumulative clock low extend time for a master device.

### 38.4.4.1.1 SCL low timeout

If the SCL line is held low by a slave device on the bus, no further communication is possible. Furthermore, the master cannot force the SCL line high to correct the error condition. To solve this problem, the SMBus protocol specifies that devices participating in a transfer must detect any clock cycle held low longer than a timeout value condition. Devices that have detected the timeout condition must reset the communication. When the I2C module is an active master, if it detects that SMBCLK low has exceeded the value of  $T_{TIMEOUT,MIN}$ , it must generate a stop condition within or after the current data byte in the transfer process. When the I2C module is a slave, if it detects the  $T_{TIMEOUT,MIN}$  condition, it resets its communication and is then able to receive a new START condition.

### 38.4.4.1.2 SCL high timeout

When the I2C module has determined that the SMBCLK and SMBDAT signals have been high for at least  $T_{HIGH:MAX}$ , it assumes that the bus is idle.

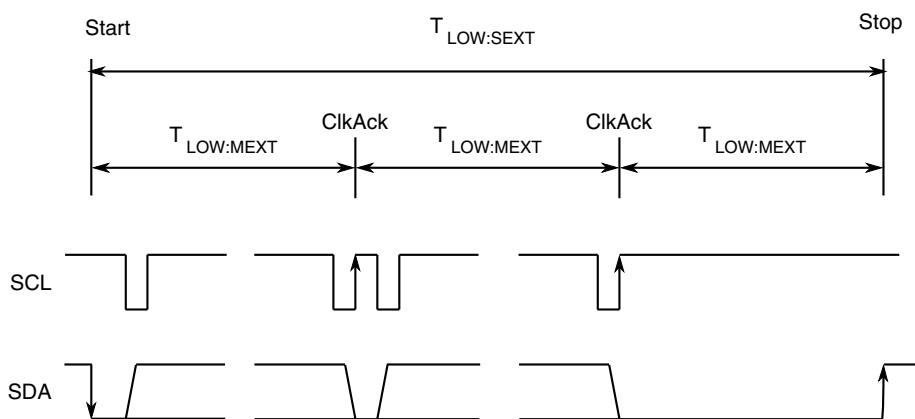
A HIGH timeout occurs after a START condition appears on the bus but before a STOP condition appears on the bus. Any master detecting this scenario can assume the bus is free when either of the following occurs:

- SHTF1 rises.
- The BUSY bit is high and SHTF1 is high.

When the SMBDAT signal is low and the SMBCLK signal is high for a period of time, another kind of timeout occurs. The time period must be defined in software. SHTF2 is used as the flag when the time limit is reached. This flag is also an interrupt resource, so it triggers IICIF.

### 38.4.4.1.3 CSMBCLK TIMEOUT MEXT and CSMBCLK TIMEOUT SEXT

The following figure illustrates the definition of the timeout intervals  $T_{LOW:SEXT}$  and  $T_{LOW:MEXT}$ . When in master mode, the I2C module must not cumulatively extend its clock cycles for a period greater than  $T_{LOW:MEXT}$  within a byte, where each byte is defined as START-to-ACK, ACK-to-ACK, or ACK-to-STOP. When CSMBCLK TIMEOUT MEXT occurs, SMBus MEXT rises and also triggers the SLTF.



**Figure 38-40. Timeout measurement intervals**

A master is allowed to abort the transaction in progress to any slave that violates the  $T_{LOW:SEXT}$  or  $T_{TIMEOUT,MIN}$  specifications. To abort the transaction, the master issues a STOP condition at the conclusion of the byte transfer in progress. When a slave, the I2C module must not cumulatively extend its clock cycles for a period greater than  $T_{LOW:SEXT}$  during any message from the initial START to the STOP. When CSMBCLK TIMEOUT SEXT occurs, SEXT rises and also triggers SLTF.

#### NOTE

CSMBCLK TIMEOUT SEXT and CSMBCLK TIMEOUT MEXT are optional functions that are implemented in the second step.

### 38.4.4.2 FAST ACK and NACK

To improve reliability and communication robustness, implementation of packet error checking (PEC) by SMBus devices is optional for SMBus devices but required for devices participating in and only during the address resolution protocol (ARP) process. The PEC is a CRC-8 error checking byte, calculated on all the message bytes. The PEC is appended to the message by the device that supplied the last data byte. If the PEC is present but not correct, a NACK is issued by the receiver. Otherwise an ACK is issued. To calculate the CRC-8 by software, this module can hold the SCL line low after

receiving the eighth SCL (8th bit) if this byte is a data byte. So software can determine whether an ACK or NACK should be sent to the bus by setting or clearing the TXAK bit if the FACK (fast ACK/NACK enable) bit is enabled.

SMBus requires a device always to acknowledge its own address, as a mechanism to detect the presence of a removable device (such as a battery or docking station) on the bus. In addition to indicating a slave device busy condition, SMBus uses the NACK mechanism to indicate the reception of an invalid command or invalid data. Because such a condition may occur on the last byte of the transfer, SMBus devices are required to have the ability to generate the not acknowledge after the transfer of each byte and before the completion of the transaction. This requirement is important because SMBus does not provide any other resend signaling. This difference in the use of the NACK signaling has implications on the specific implementation of the SMBus port, especially in devices that handle critical system data such as the SMBus host and the SBS components.

**NOTE**

In the last byte of master receive slave transmit mode, the master must send a NACK to the bus, so FACK must be switched off before the last byte transmits.

### **38.4.5 Resets**

The I2C module is disabled after a reset. The I2C module cannot cause a core reset.

### **38.4.6 Interrupts**

The I2C module generates an interrupt when any of the events in the following table occur, provided that the IICIE bit is set. The interrupt is driven by the IICIF bit (of the I2C Status Register) and masked with the IICIE bit (of the I2C Control Register 1). The IICIF bit must be cleared (by software) by writing 1 to it in the interrupt routine. The SMBus timeouts interrupt is driven by SLTF and masked with the IICIE bit. The SLTF bit must be cleared by software by writing 1 to it in the interrupt routine. You can determine the interrupt type by reading the Status Register.

**NOTE**

In master receive mode, the FACK bit must be set to zero before the last byte transfer.

**Table 38-44. Interrupt summary**

| Interrupt source                    | Status | Flag  | Local enable    |
|-------------------------------------|--------|-------|-----------------|
| Complete 1-byte transfer            | TCF    | IICIF | IICIE           |
| Match of received calling address   | IAAS   | IICIF | IICIE           |
| Arbitration lost                    | ARBL   | IICIF | IICIE           |
| I <sup>2</sup> C bus stop detection | STOPF  | IICIF | IICIE & STOPIE  |
| SMBus SCL low timeout               | SLTF   | IICIF | IICIE           |
| SMBus SCL high SDA low timeout      | SHTF2  | IICIF | IICIE & SHTF2IE |
| Wakeup from stop or wait mode       | IAAS   | IICIF | IICIE & WUEN    |

### 38.4.6.1 Byte transfer interrupt

The Transfer Complete Flag (TCF) bit is set at the falling edge of the ninth clock to indicate the completion of a byte and acknowledgement transfer. When FACK is enabled, TCF is then set at the falling edge of eighth clock to indicate the completion of byte.

### 38.4.6.2 Address detect interrupt

When the calling address matches the programmed slave address (I2C Address Register) or when the GCAEN bit is set and a general call is received, the IAAS bit in the Status Register is set. The CPU is interrupted, provided the IICIE bit is set. The CPU must check the SRW bit and set its Tx mode accordingly.

### 38.4.6.3 Stop Detect Interrupt

When the stop status is detected on the I<sup>2</sup>C bus, the STOPF bit is set to 1. The CPU is interrupted, provided the IICIE and STOPIE bits are both set to 1.

### 38.4.6.4 Exit from low-power/stop modes

The slave receive input detect circuit and address matching feature are still active on low power modes (wait and stop). An asynchronous input matching slave address or general call address brings the CPU out of low power/stop mode if the interrupt is not masked. Therefore, TCF and IAAS both can trigger this interrupt.

### 38.4.6.5 Arbitration lost interrupt

The I2C is a true multimaster bus that allows more than one master to be connected on it. If two or more masters try to control the bus at the same time, the relative priority of the contending masters is determined by a data arbitration procedure. The I2C module asserts the arbitration-lost interrupt when it loses the data arbitration process and the ARBL bit in the Status Register is set.

Arbitration is lost in the following circumstances:

1. SDA is sampled as low when the master drives high during an address or data transmit cycle.
2. SDA is sampled as low when the master drives high during the acknowledge bit of a data receive cycle.
3. A START cycle is attempted when the bus is busy.
4. A repeated START cycle is requested in slave mode.
5. A STOP condition is detected when the master did not request it.

The ARBL bit must be cleared (by software) by writing 1 to it.

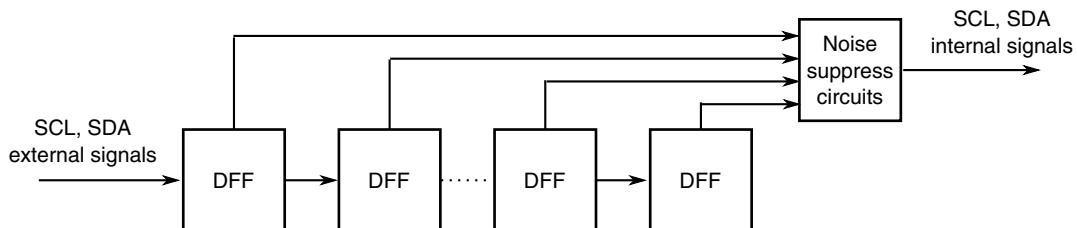
### 38.4.6.6 Timeout interrupt in SMBus

When the IICIE bit is set, the I2C module asserts a timeout interrupt (outputs SLTF and SHTF2) upon detection of any of the mentioned timeout conditions, with one exception. The SCL high and SDA high TIMEOUT mechanism must not be used to influence the timeout interrupt output, because this timeout indicates an idle condition on the bus. SHTF1 rises when it matches the SCL high and SDA high TIMEOUT and falls automatically just to indicate the bus status. The SHTF2's timeout period is the same as that of SHTF1, which is short compared to that of SLTF, so another control bit, SHTF2IE, is added to enable or disable it.

### 38.4.7 Programmable input glitch filter

An I2C glitch filter has been added outside legacy I2C modules but within the I2C package. This filter can absorb glitches on the I2C clock and data lines for the I2C module. The width of the glitch to absorb can be specified in terms of the number of (half) bus clock cycles. A single Programmable Input Glitch Filter control register is provided. Effectively, any down-up-down or up-down-up transition on the data line that

occurs within the number of clock cycles programmed in this register is ignored by the I2C module. The programmer must specify the size of the glitch (in terms of bus clock cycles) for the filter to absorb and not pass.



**Figure 38-41. Programmable input glitch filter diagram**

### 38.4.8 Address matching wakeup

When a primary, range, or general call address match occurs when the I2C module is in slave receive mode, the MCU wakes from a low power mode with no peripheral bus running. Data sent on the bus that is the same as a target device address might also wake the target MCU.

After the address matching IAAS bit is set, an interrupt is sent at the end of address matching to wake the core. The IAAS bit must be cleared after the clock recovery.

#### NOTE

After the system recovers and is in Run mode, restart the I2C module if it is needed to transfer packets. The SCL line is not held low until the I2C module resets after address matching.

The main purpose of this feature is to wake the MCU from a low power mode where no peripheral bus is running. When the MCU is in such a mode: addressing as a slave, slave read/write, and sending an acknowledge bit are not fully supported. To avoid I2C transfer problems resulting from this situation, firmware should prevent the MCU execution of a STOP instruction when the I2C module is in the middle of a transfer unless the Stop mode holdoff feature is used during this period (set FLT[SHEN] to 1).

### 38.4.9 DMA support

If the DMAEN bit is cleared and the IICIE bit is set, an interrupt condition generates an interrupt request. If the DMAEN bit is set and the IICIE bit is set, an interrupt condition generates a DMA request instead. DMA requests are generated by the transfer complete flag (TCF).

If the DMAEN bit is set, the only arbitration lost is to another I2C module (error), and SCL low timeouts (error) generate CPU interrupts. All other events initiate a DMA transfer.

#### NOTE

Before the last byte of master receive mode, TXAK must be set to send a NACK after the last byte's transfer. Therefore, the DMA must be disabled before the last byte's transfer.

#### NOTE

In 10-bit address mode transmission, the addresses to send occupy 2-3 bytes. During this transfer period, the DMA must be disabled because the C1 register is written to send a repeat start or to change the transfer direction.

## 38.5 Initialization/application information

### Module Initialization (Slave)

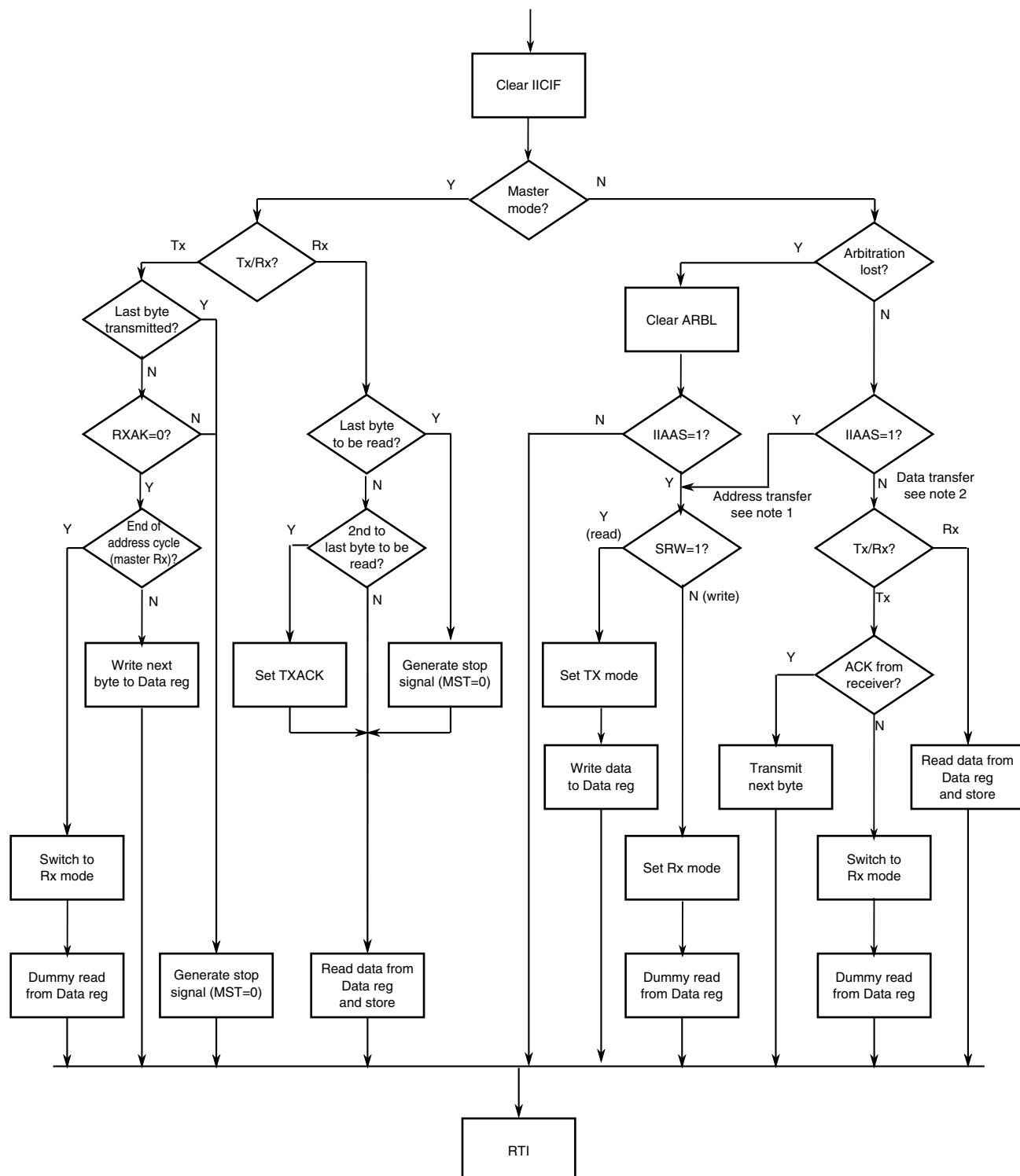
1. Write: Control Register 2
  - to enable or disable general call
  - to select 10-bit or 7-bit addressing mode
2. Write: Address Register 1 to set the slave address
3. Write: Control Register 1 to enable the I2C module and interrupts
4. Initialize RAM variables ( $IICEN = 1$  and  $IICIE = 1$ ) for transmit data
5. Initialize RAM variables used to achieve the routine shown in the following figure

### Module Initialization (Master)

1. Write: Frequency Divider register to set the I2C baud rate (see example in description of [ICR](#))
2. Write: Control Register 1 to enable the I2C module and interrupts
3. Initialize RAM variables ( $IICEN = 1$  and  $IICIE = 1$ ) for transmit data
4. Initialize RAM variables used to achieve the routine shown in the following figure
5. Write: Control Register 1 to enable TX

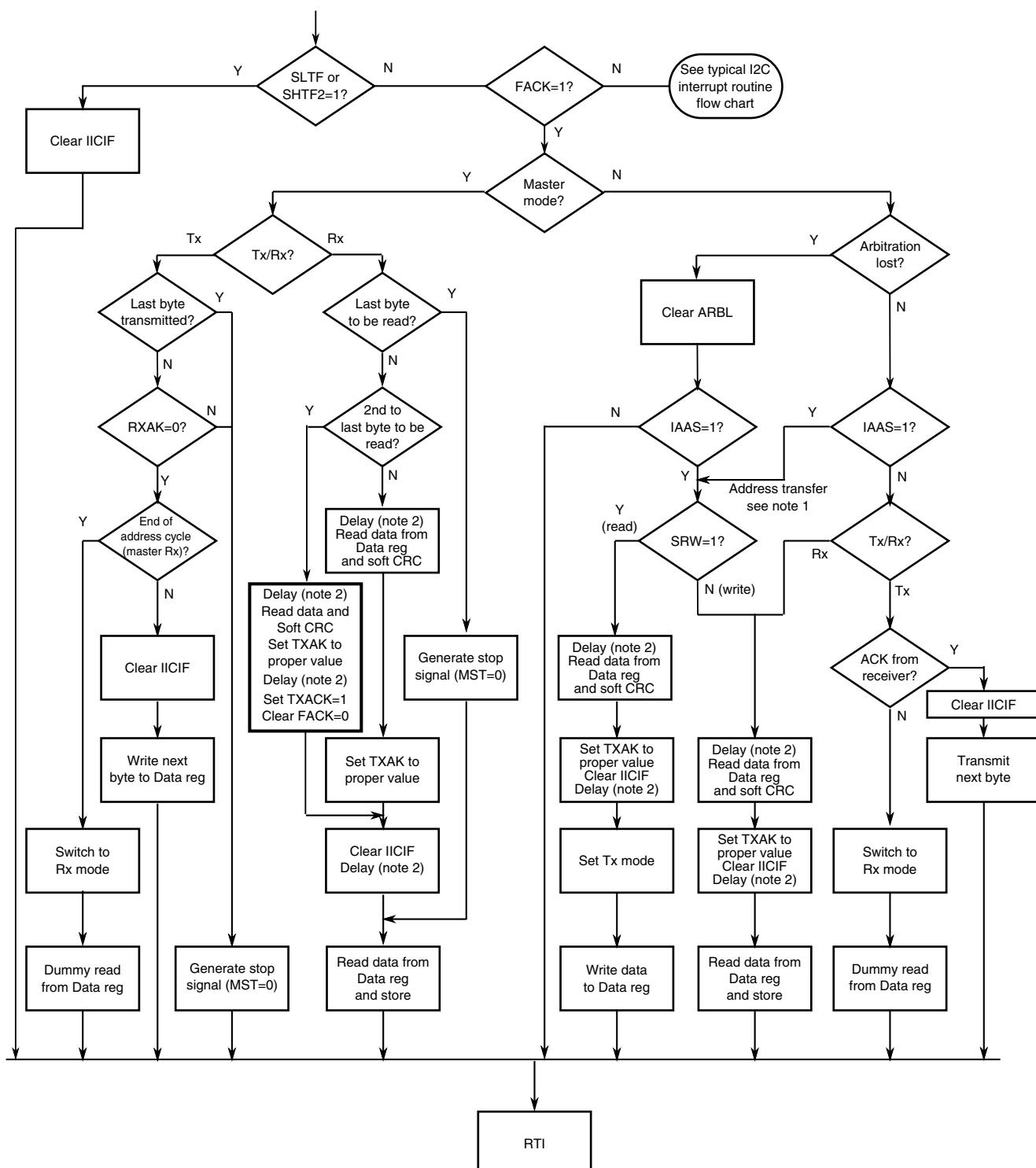
6. Write: Control Register 1 to enable MST (master mode)
7. Write: Data register with the address of the target slave (the LSB of this byte determines whether the communication is master receive or transmit)

The routine shown in the following figure encompasses both master and slave I2C operations. For slave operation, an incoming I2C message that contains the proper address begins I2C communication. For master operation, communication must be initiated by writing the Data register.

**Notes:**

- If general call is enabled, check to determine if the received address is a general call address (0x00).  
If the received address is a general call address, the general call must be handled by user software.
- When 10-bit addressing addresses a slave, the slave sees an interrupt following the first byte of the extended address.  
Ensure that for this interrupt, the contents of the Data register are ignored and not treated as a valid data transfer.

**Figure 38-42. Typical I2C interrupt routine**

**Notes:**

1. If general call or SIICAEN is enabled, check to determine if the received address is a general call address (0x00) or an SMBus device default address. In either case, they must be handled by user software.
2. In receive mode, one bit time delay may be needed before the first and second data reading.

**Figure 38-43. Typical I2C SMBus interrupt routine**



# **Chapter 39**

## **Universal Asynchronous Receiver/Transmitter (UART0)**

### **39.1 Introduction**

#### **39.1.1 Features**

Features of the UART module include:

- Full-duplex, standard non-return-to-zero (NRZ) format
- Double-buffered transmitter and receiver with separate enables
- Programmable baud rates (13-bit modulo divider)
- Transmit and receive baud rate can operate asynchronous to the bus clock:
  - Baud rate can be configured independently of the bus clock frequency
  - Supports operation in Stop modes
- Configurable receiver baud rate oversampling ratio from 4x to 32x
- Interrupt, DMA or polled operation:
  - Transmit data register empty and transmission complete
  - Receive data register full
  - Receive overrun, parity error, framing error, and noise error
  - Idle receiver detect
  - Active edge on receive pin
  - Break detect supporting LIN
- Hardware parity generation and checking
- Programmable 8-bit, 9-bit or 10-bit character length
- Programmable 1-bit or 2-bit stop bits
- Receiver wakeup by idle-line, address-mark or address match
- Optional 13-bit break character generation / 11-bit break character detection
- Selectable transmitter output and receiver input polarity

## 39.1.2 Modes of operation

### 39.1.2.1 Stop mode

The UART will remain functional during Stop mode, provided the asynchronous transmit and receive clock remains enabled. The UART can generate an interrupt or DMA request to cause a wakeup from Stop mode.

### 39.1.2.2 Wait mode

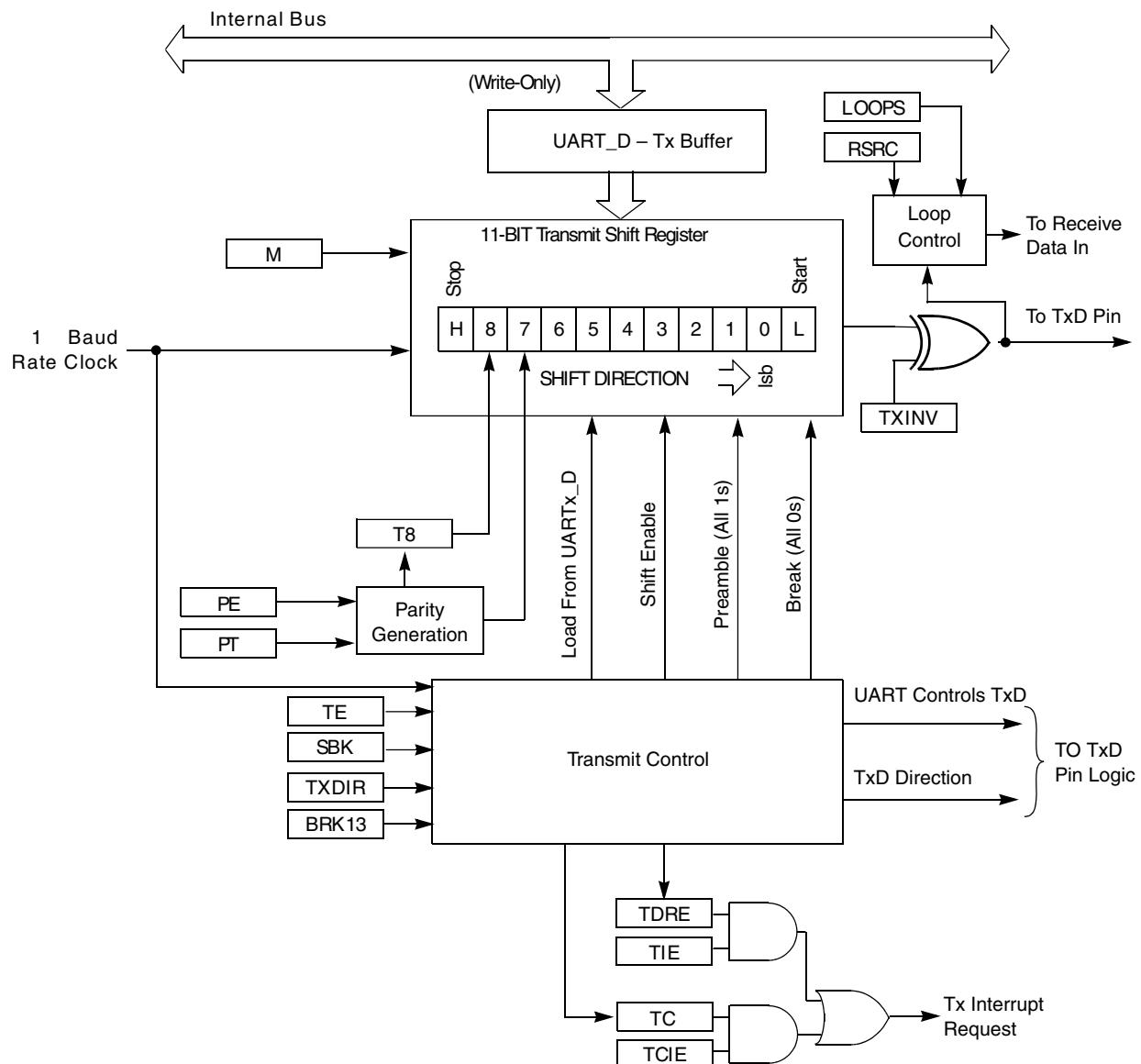
The UART can be configured to Stop in Wait modes, when the DOZEEN bit is set. The transmitter and receiver will finish transmitting/receiving the current word.

### 39.1.2.3 Debug mode

The UART remains functional in debug mode.

## 39.1.3 Block diagram

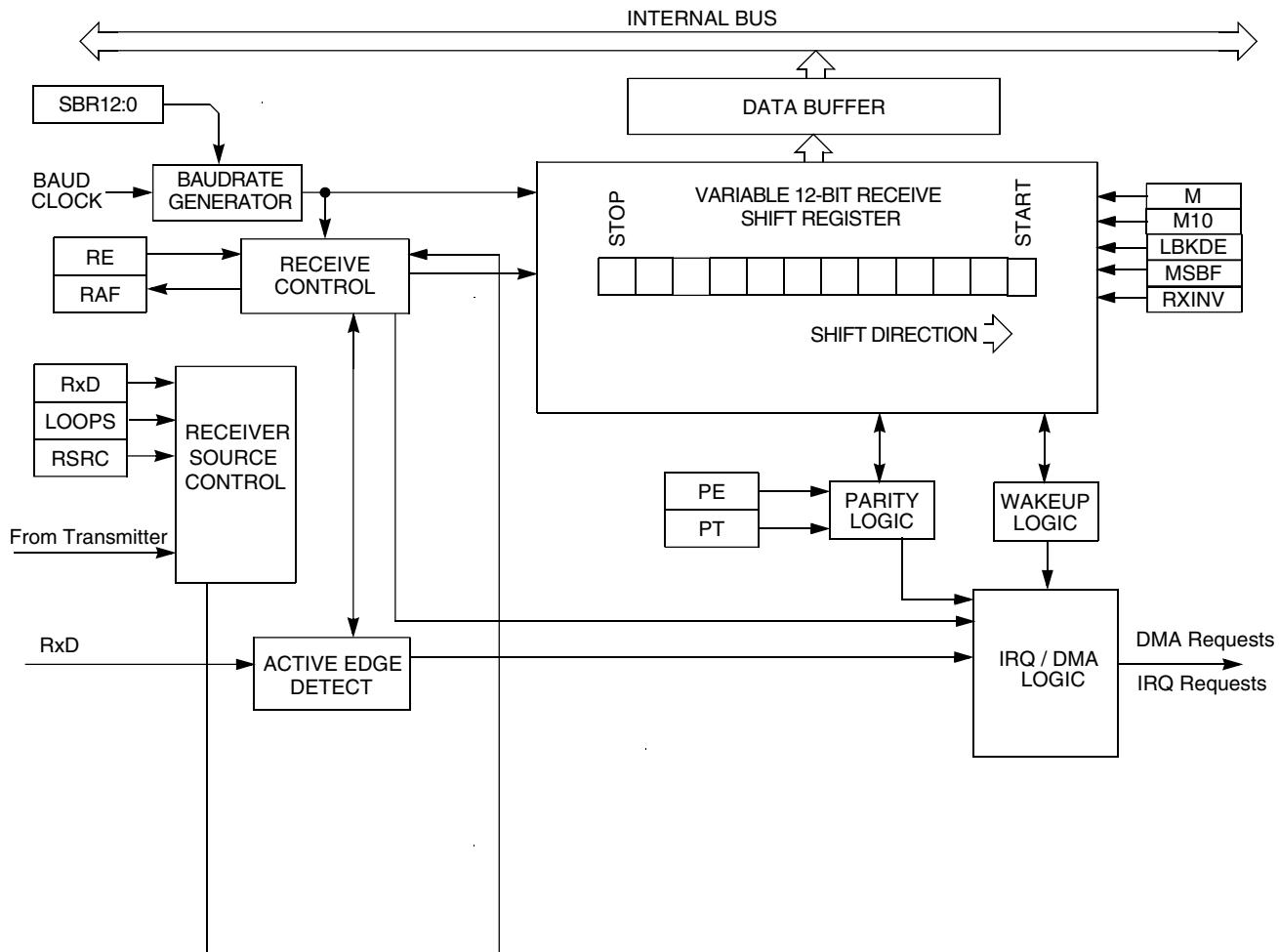
The following figure shows the transmitter portion of the UART.



**Figure 39-1. UART transmitter block diagram**

The following figure shows the receiver portion of the UART.

## Register definition



**Figure 39-2. UART receiver block diagram**

## 39.2 Register definition

The UART includes registers to control baud rate, select UART options, report UART status, and for transmit/receive data. Accesses to address outside the valid memory map will generate a bus error.

**UART memory map**

| Absolute address (hex) | Register name                            | Width (in bits) | Access | Reset value | Section/ page              |
|------------------------|--|-----------------|--------|-------------|----------------------------|
| 4006_A000              | UART Baud Rate Register High (UART0_BDH) | 8               | R/W    | 00h         | <a href="#">39.2.1/725</a> |
| 4006_A001              | UART Baud Rate Register Low (UART0_BDL)  | 8               | R/W    | 04h         | <a href="#">39.2.2/726</a> |
| 4006_A002              | UART Control Register 1 (UART0_C1)       | 8               | R/W    | 00h         | <a href="#">39.2.3/726</a> |
| 4006_A003              | UART Control Register 2 (UART0_C2)       | 8               | R/W    | 00h         | <a href="#">39.2.4/728</a> |
| 4006_A004              | UART Status Register 1 (UART0_S1)        | 8               | R/W    | C0h         | <a href="#">39.2.5/729</a> |

*Table continues on the next page...*

**UART memory map (continued)**

| Absolute address (hex) | Register name                              | Width (in bits) | Access | Reset value | Section/page                |
|------------------------|--|-----------------|--------|-------------|-----------------------------|
| 4006_A005              | UART Status Register 2 (UART0_S2)          | 8               | R/W    | 00h         | <a href="#">39.2.6/731</a>  |
| 4006_A006              | UART Control Register 3 (UART0_C3)         | 8               | R/W    | 00h         | <a href="#">39.2.7/733</a>  |
| 4006_A007              | UART Data Register (UART0_D)               | 8               | R/W    | 00h         | <a href="#">39.2.8/734</a>  |
| 4006_A008              | UART Match Address Registers 1 (UART0_MA1) | 8               | R/W    | 00h         | <a href="#">39.2.9/735</a>  |
| 4006_A009              | UART Match Address Registers 2 (UART0_MA2) | 8               | R/W    | 00h         | <a href="#">39.2.10/736</a> |
| 4006_A00A              | UART Control Register 4 (UART0_C4)         | 8               | R/W    | 0Fh         | <a href="#">39.2.11/736</a> |
| 4006_A00B              | UART Control Register 5 (UART0_C5)         | 8               | R/W    | 00h         | <a href="#">39.2.12/737</a> |

**39.2.1 UART Baud Rate Register High (UARTx\_BDH)**

This register, along with **UART\_BDL**, controls the prescale divisor for UART baud rate generation. The 13-bit baud rate setting [SBR12:SBR0] should only be updated when the transmitter and receiver are both disabled.

Address: Base address + 0h offset

| Bit   | 7     | 6       | 5    | 4 | 3 | 2   | 1 | 0 |
|-------|-------|---------|------|---|---|-----|---|---|
| Read  | LBKDI | RXEDGIE | SBNS |   |   | SBR |   |   |
| Write | 0     | 0       | 0    | 0 | 0 | 0   | 0 | 0 |

**UARTx\_BDH field descriptions**

| Field     | Description  |
|-----------|--|
| 7 LBKDI   | LIN Break Detect Interrupt Enable (for LBKDIF)<br>0 Hardware interrupts from <b>UART_S2[LBKDI]</b> disabled (use polling).<br>1 Hardware interrupt requested when <b>UART_S2[LBKDI]</b> flag is 1.                     |
| 6 RXEDGIE | RX Input Active Edge Interrupt Enable (for RXEDGIF)<br>0 Hardware interrupts from <b>UART_S2[RXEDGIF]</b> disabled (use polling).<br>1 Hardware interrupt requested when <b>UART_S2[RXEDGIF]</b> flag is 1.            |
| 5 SBNS    | Stop Bit Number Select<br>SBNS determines whether data characters are one or two stop bits. This bit should only be changed when the transmitter and receiver are both disabled.<br>0 One stop bit.<br>1 Two stop bit. |

*Table continues on the next page...*

## Register definition

### UARTx\_BDH field descriptions (continued)

| Field      | Description   |
|------------|---|
| 4–0<br>SBR | Baud Rate Modulo Divisor.<br><br>The 13 bits in SBR[12:0] are referred to collectively as BR, and they set the modulo divide rate for the baud rate generator. When BR is 1 - 8191, the baud rate equals baud clock / ((OSR+1) × BR). |

## 39.2.2 UART Baud Rate Register Low (UARTx\_BDL)

This register, along with UART\_BDH, control the prescale divisor for UART baud rate generation. The 13-bit baud rate setting [SBR12:SBR0] can only be updated when the transmitter and receiver are both disabled.

UART\_BDL is reset to a non-zero value, so after reset the baud rate generator remains disabled until the first time the receiver or transmitter is enabled; that is, UART\_C2[RE] or UART\_C2[TE] bits are written to 1.

Address: Base address + 1h offset

| Bit        | 7   | 6 | 5 | 4 |  | 3 | 2 | 1 | 0 |
|------------|-----|---|---|---|--|---|---|---|---|
| Read Write | SBR |   |   |   |  |   |   |   |   |
| Reset      | 0   | 0 | 0 | 0 |  | 0 | 1 | 0 | 0 |

### UARTx\_BDL field descriptions

| Field      | Description  |
|------------|--|
| 7–0<br>SBR | Baud Rate Modulo Divisor<br><br>These 13 bits in SBR[12:0] are referred to collectively as BR. They set the modulo divide rate for the baud rate generator. When BR is 1 - 8191, the baud rate equals baud clock/((OSR+1) × BR). |

## 39.2.3 UART Control Register 1 (UARTx\_C1)

This read/write register controls various optional features of the UART system. This register should only be altered when the transmitter and receiver are both disabled.

Address: Base address + 2h offset

| Bit        | 7     | 6      | 5    | 4 |  | 3    | 2   | 1  | 0  |
|------------|-------|--------|------|---|--|------|-----|----|----|
| Read Write | LOOPS | DOZEEN | RSRC | M |  | WAKE | ILT | PE | PT |
| Reset      | 0     | 0      | 0    | 0 |  | 0    | 0   | 0  | 0  |

## UARTx\_C1 field descriptions

| Field       | Description   |
|-------------|---|
| 7<br>LOOPS  | <p><b>Loop Mode Select</b></p> <p>Selects between loop back modes and normal 2-pin full-duplex modes. When LOOPS is set, the transmitter output is internally connected to the receiver input.</p> <ul style="list-style-type: none"> <li>0 Normal operation - UART_RX and UART_TX use separate pins.</li> <li>1 Loop mode or single-wire mode where transmitter outputs are internally connected to receiver input. (See RSRC bit.) UART_RX pin is not used by UART .</li> </ul>   |
| 6<br>DOZEEN | <p><b>Doze Enable</b></p> <ul style="list-style-type: none"> <li>0 UART is enabled in Wait mode.</li> <li>1 UART is disabled in Wait mode.</li> </ul>   |
| 5<br>RSRC   | <p><b>Receiver Source Select</b></p> <p>This bit has no meaning or effect unless the LOOPS bit is set to 1. When LOOPS is set, the receiver input is internally connected to the UART_TX pin and RSRC determines whether this connection is also connected to the transmitter output.</p> <ul style="list-style-type: none"> <li>0 Provided LOOPS is set, RSRC is cleared, selects internal loop back mode and the UART does not use the UART_RX pins.</li> <li>1 Single-wire UART mode where the UART_TX pin is connected to the transmitter output and receiver input.</li> </ul> |
| 4<br>M      | <p><b>9-Bit or 8-Bit Mode Select</b></p> <ul style="list-style-type: none"> <li>0 Receiver and transmitter use 8-bit data characters.</li> <li>1 Receiver and transmitter use 9-bit data characters.</li> </ul>   |
| 3<br>WAKE   | <p><b>Receiver Wakeup Method Select</b></p> <ul style="list-style-type: none"> <li>0 Idle-line wakeup.</li> <li>1 Address-mark wakeup.</li> </ul>   |
| 2<br>ILT    | <p><b>Idle Line Type Select</b></p> <p>Setting this bit to 1 ensures that the stop bits and logic 1 bits at the end of a character do not count toward the 10 to 13 bit times of logic high level needed by the idle line detection logic.</p> <ul style="list-style-type: none"> <li>0 Idle character bit count starts after start bit.</li> <li>1 Idle character bit count starts after stop bit.</li> </ul>  |
| 1<br>PE     | <p><b>Parity Enable</b></p> <p>Enables hardware parity generation and checking. When parity is enabled, the bit immediately before the stop bit is treated as the parity bit.</p> <ul style="list-style-type: none"> <li>0 No hardware parity generation or checking.</li> <li>1 Parity enabled.</li> </ul>   |
| 0<br>PT     | <p><b>Parity Type</b></p> <p>Provided parity is enabled (PE = 1), this bit selects even or odd parity. Odd parity means the total number of 1s in the data character, including the parity bit, is odd. Even parity means the total number of 1s in the data character, including the parity bit, is even.</p> <ul style="list-style-type: none"> <li>0 Even parity.</li> <li>1 Odd parity.</li> </ul>  |

### 39.2.4 UART Control Register 2 (UARTx\_C2)

This register can be read or written at any time.

Address: Base address + 3h offset

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

#### UARTx\_C2 field descriptions

| Field     | Description  |
|-----------|--|
| 7<br>TIE  | Transmit Interrupt Enable for TDRE<br><br>0 Hardware interrupts from TDRE disabled; use polling.<br>1 Hardware interrupt requested when TDRE flag is 1.  |
| 6<br>TCIE | Transmission Complete Interrupt Enable for TC<br><br>0 Hardware interrupts from TC disabled; use polling.<br>1 Hardware interrupt requested when TC flag is 1.   |
| 5<br>RIE  | Receiver Interrupt Enable for RDRF<br><br>0 Hardware interrupts from RDRF disabled; use polling.<br>1 Hardware interrupt requested when RDRF flag is 1.  |
| 4<br>ILIE | Idle Line Interrupt Enable for IDLE<br><br>0 Hardware interrupts from IDLE disabled; use polling.<br>1 Hardware interrupt requested when IDLE flag is 1.   |
| 3<br>TE   | Transmitter Enable<br><br>TE must be 1 to use the UART transmitter. When TE is set, the UART forces the UART _TX pin to act as an output for the UART system.<br><br>When the UART is configured for single-wire operation (LOOPS = RSRC = 1), TXDIR controls the direction of traffic on the single UART communication line ( UART _TX pin).<br><br>TE can also queue an idle character by clearing TE then setting TE while a transmission is in progress.<br><br>When TE is written to 0, the transmitter keeps control of the port UART _TX pin until any data, queued idle, or queued break character finishes transmitting before allowing the pin to tristate.<br><br>0 Transmitter disabled.<br>1 Transmitter enabled. |
| 2<br>RE   | Receiver Enable<br><br>When the UART receiver is off or LOOPS is set, the UART _RX pin is not used by the UART .<br><br>When RE is written to 0, the receiver finishes receiving the current character (if any).<br><br>0 Receiver disabled.<br>1 Receiver enabled.  |
| 1<br>RWU  | Receiver Wakeup Control  |

*Table continues on the next page...*

**UARTx\_C2 field descriptions (continued)**

| Field    | Description  |
|----------|--|
|          | <p>This bit can be written to 1 to place the UART receiver in a standby state where it waits for automatic hardware detection of a selected wakeup condition. The wakeup condition is an idle line between messages, WAKE = 0, idle-line wakeup, or a logic 1 in the most significant data bit in a character, WAKE = 1, address-mark wakeup. Application software sets RWU and, normally, a selected hardware condition automatically clears RWU.</p> <p>0 Normal UART receiver operation.<br/>1 UART receiver in standby waiting for wakeup condition.</p> |
| 0<br>SBK | <p>Send Break</p> <p>Writing a 1 and then a 0 to SBK queues a break character in the transmit data stream. Additional break characters of 10 to 13, or 13 to 16 if BRK13 = 1, bit times of logic 0 are queued as long as SBK is set. Depending on the timing of the set and clear of SBK relative to the information currently being transmitted, a second break character may be queued before software clears SBK.</p> <p>0 Normal transmitter operation.<br/>1 Queue break character(s) to be sent.</p>   |

**39.2.5 UART Status Register 1 (UARTx\_S1)**

Address: Base address + 4h offset

| Bit   | 7    | 6  | 5    | 4    | 3   | 2   | 1   | 0   |
|-------|------|----|------|------|-----|-----|-----|-----|
| Read  | TDRE | TC | RDRF | IDLE | OR  | NF  | FE  | PF  |
| Write |      |    |      | w1c  | w1c | w1c | w1c | w1c |
| Reset | 1    | 1  | 0    | 0    | 0   | 0   | 0   | 0   |

**UARTx\_S1 field descriptions**

| Field     | Description  |
|-----------|--|
| 7<br>TDRE | <p>Transmit Data Register Empty Flag</p> <p>TDRE is set out of reset and whenever there is room to write data to the transmit data buffer. To clear TDRE, write to the UART data register (UART_D).</p> <p>0 Transmit data buffer full.<br/>1 Transmit data buffer empty.</p>  |
| 6<br>TC   | <p>Transmission Complete Flag</p> <p>TC is set out of reset and when TDRE is set and no data, preamble, or break character is being transmitted.</p> <p>TC is cleared automatically by one of the following:</p> <ul style="list-style-type: none"> <li>• Write to the UART data register (UART_D) to transmit new data</li> <li>• Queue a preamble by changing TE from 0 to 1</li> <li>• Queue a break character by writing 1 to UART_C2[SBK]</li> </ul> <p>0 Transmitter active (sending data, a preamble, or a break).<br/>1 Transmitter idle (transmission activity complete).</p> |

*Table continues on the next page...*

**UARTx\_S1 field descriptions (continued)**

| Field     | Description   |
|-----------|---|
| 5<br>RDRF | <p>Receive Data Register Full Flag</p> <p>RDRF becomes set whenever the receive data buffer is full. To clear RDRF, read the UART data register (UART_D).</p> <p>0 Receive data buffer empty.<br/>1 Receive data buffer full.</p>   |
| 4<br>IDLE | <p>Idle Line Flag</p> <p>IDLE is set when the UART receive line becomes idle for a full character time after a period of activity. When ILT is cleared, the receiver starts counting idle bit times after the start bit. If the receive character is all 1s, these bit times and the stop bits time count toward the full character time of logic high, 10 to 13 bit times, needed for the receiver to detect an idle line. When ILT is set, the receiver doesn't start counting idle bit times until after the stop bits. The stop bits and any logic high bit times at the end of the previous character do not count toward the full character time of logic high needed for the receiver to detect an idle line.</p> <p>To clear IDLE, write logic 1 to the IDLE flag. After IDLE has been cleared, it cannot become set again until after a new character has been received and RDRF has been set. IDLE is set only once even if the receive line remains idle for an extended period.</p> <p>0 No idle line detected.<br/>1 Idle line was detected.</p> |
| 3<br>OR   | <p>Receiver Overrun Flag</p> <p>OR is set when a new serial character is ready to be transferred to the receive data buffer, but the previously received character has not been read from UART_D yet. In this case, the new character, and all associated error information, is lost because there is no room to move it into UART_D. To clear OR, write a logic 1 to the OR flag.</p> <p>0 No overrun.<br/>1 Receive overrun (new UART data lost).</p>   |
| 2<br>NF   | <p>Noise Flag</p> <p>The advanced sampling technique used in the receiver takes three samples in each of the received bits. If any of these samples disagrees with the rest of the samples within any bit time in the frame, the flag NF is set at the same time as RDRF is set for the character. To clear NF, write logic one to the NF.</p> <p>0 No noise detected.<br/>1 Noise detected in the received character in UART_D.</p>  |
| 1<br>FE   | <p>Framing Error Flag</p> <p>FE is set at the same time as RDRF when the receiver detects a logic 0 where a stop bit was expected. This suggests the receiver was not properly aligned to a character frame. To clear FE, write a logic one to the FE flag.</p> <p>0 No framing error detected. This does not guarantee the framing is correct.<br/>1 Framing error.</p>  |
| 0<br>PF   | <p>Parity Error Flag</p> <p>PF is set at the same time as RDRF when parity is enabled (PE = 1) and the parity bit in the received character does not agree with the expected parity value. To clear PF, write a logic one to the PF.</p>  |

*Table continues on the next page...*

### **UARTx\_S1 field descriptions (continued)**

| Field | Description      |
|-------|------------------|
| 0     | No parity error. |
| 1     | Parity error.    |

**UARTx\_S2 field descriptions (continued)**

| Field   | Description  |
|---------|--|
|         | <p>0 LSB (bit0) is the first bit that is transmitted following the start bit. Further, the first bit received after the start bit is identified as bit0.</p> <p>1 MSB (bit9, bit8, bit7 or bit6) is the first bit that is transmitted following the start bit depending on the setting of C1[M], C1[PE] and C4[M10]. Further, the first bit received after the start bit is identified as bit9, bit8, bit7 or bit6 depending on the setting of C1[M] and C1[PE].</p>   |
| 4 RXINV | <p>Receive Data Inversion</p> <p>Setting this bit reverses the polarity of the received data input.</p> <p><b>NOTE:</b> Setting RXINV inverts the UART _RXD input for all cases: data bits, start and stop bits, break, and idle.</p> <p>0 Receive data not inverted.<br/>1 Receive data inverted.</p>   |
| 3 RWUID | <p>Receive Wake Up Idle Detect</p> <p>RWUID controls whether the idle character that wakes up the receiver sets the IDLE bit. This bit should only be changed when the receiver is disabled.</p> <p>0 During receive standby state (RWU = 1), the IDLE bit does not get set upon detection of an idle character.<br/>1 During receive standby state (RWU = 1), the IDLE bit gets set upon detection of an idle character.</p>  |
| 2 BRK13 | <p>Break Character Generation Length</p> <p>BRK13 selects a longer transmitted break character length. Detection of a framing error is not affected by the state of this bit. This bit should only be changed when the transmitter is disabled.</p> <p>0 Break character is transmitted with length of 10 bit times (if M = 0, SBNS = 0) or 11 (if M = 1, SBNS = 0 or M = 0, SBNS = 1) or 12 (if M = 1, SBNS = 1 or M10 = 1, SNBS = 0) or 13 (if M10 = 1, SNBS = 1).<br/>1 Break character is transmitted with length of 13 bit times (if M = 0, SBNS = 0) or 14 (if M = 1, SBNS = 0 or M = 0, SBNS = 1) or 15 (if M = 1, SBNS = 1 or M10 = 1, SNBS = 0) or 16 (if M10 = 1, SNBS = 1).</p> |
| 1 LBKDE | <p>LIN Break Detection Enable</p> <p>LBKDE selects a longer break character detection length. While LBKDE is set, framing error (FE) and receive data register full (RDRF) flags are prevented from setting.</p> <p>0 Break character is detected at length 10 bit times (if M = 0, SBNS = 0) or 11 (if M = 1, SBNS = 0 or M = 0, SBNS = 1) or 12 (if M = 1, SBNS = 1 or M10 = 1, SNBS = 0) or 13 (if M10 = 1, SNBS = 1).<br/>1 Break character is detected at length of 11 bit times (if M = 0, SBNS = 0) or 12 (if M = 1, SBNS = 0 or M = 0, SBNS = 1) or 14 (if M = 1, SBNS = 1 or M10 = 1, SNBS = 0) or 15 (if M10 = 1, SNBS = 1).</p>   |
| 0 RAF   | <p>Receiver Active Flag</p> <p>RAF is set when the UART receiver detects the beginning of a valid start bit, and RAF is cleared automatically when the receiver detects an idle line.</p> <p>0 UART receiver idle waiting for a start bit.<br/>1 UART receiver active ( UART _RXD input not idle).</p>   |

### 39.2.7 UART Control Register 3 (UARTx\_C3)

Address: Base address + 6h offset

| Bit        | 7    | 6    | 5     | 4     | 3    | 2    | 1    | 0    |
|------------|------|------|-------|-------|------|------|------|------|
| Read Write | R8T9 | R9T8 | TXDIR | TXINV | ORIE | NEIE | FEIE | PEIE |
| Reset      | 0    | 0    | 0     | 0     | 0    | 0    | 0    | 0    |

#### UARTx\_C3 field descriptions

| Field      | Description  |
|------------|--|
| 7<br>R8T9  | <p>Receive Bit 8 / Transmit Bit 9</p> <p>When the UART is configured for 9-bit data (<math>M = 1</math>), R8 can be thought of as a ninth receive data bit to the left of the msb of the buffered data in the UART_D register. When reading 9-bit data, read R8 before reading UART_D because reading UART_D completes automatic flag clearing sequences that could allow R8 and UART_D to be overwritten with new data.</p> <p>When the UART is configured for 10-bit data (<math>M10 = 1</math>), T9 may be thought of as a tenth transmit data bit. When writing 10-bit data, the entire 10-bit value is transferred to the UART transmit buffer when UART_D is written so T9 and T8 should be written, if it needs to change from its previous value, before UART_D is written. If T9 and T8 do not need to change in the new value, such as when it is used to generate mark or space parity, they need not be written each time UART_D is written.</p> |
| 6<br>R9T8  | <p>Receive Bit 9 / Transmit Bit 8</p> <p>When the UART is configured for 9-bit data (<math>M = 1</math>), T8 may be thought of as a ninth transmit data bit to the left of the msb of the data in the UART_D register. When writing 9-bit data, the entire 9-bit value is transferred to the UART transmit buffer after UART_D is written so T8 should be written, if it needs to change from its previous value, before UART_D is written. If T8 does not need to change in the new value, such as when it is used to generate mark or space parity, it need not be written each time UART_D is written.</p> <p>When the UART is configured for 10-bit data (<math>M10 = 1</math>), R9 can be thought of as a tenth receive data bit. When reading 10-bit data, read R9 and R8 before reading UART_D because reading UART_D completes automatic flag clearing sequences that could allow R8, R9 and UART_D to be overwritten with new data.</p>             |
| 5<br>TXDIR | <p>UART_TX Pin Direction in Single-Wire Mode</p> <p>When the is configured for single-wire half-duplex operation (LOOPS = RSRC = 1), this bit determines the direction of data at the UART_TXD pin. When clearing TXDIR, the transmitter will finish receiving the current character (if any) before the receiver starts receiving data from the UART_TXD pin.</p> <p>0 UART_TXD pin is an input in single-wire mode.<br/>1 UART_TXD pin is an output in single-wire mode.</p>   |
| 4<br>TXINV | <p>Transmit Data Inversion</p> <p>Setting this bit reverses the polarity of the transmitted data output.</p> <p><b>NOTE:</b> Setting TXINV inverts the UART_TXD output for all cases: data bits, start and stop bits, break, and idle.</p> <p>0 Transmit data not inverted.<br/>1 Transmit data inverted.</p>  |
| 3<br>ORIE  | <p>Overrun Interrupt Enable</p> <p>This bit enables the overrun flag (OR) to generate hardware interrupt requests.</p>   |

Table continues on the next page...

**UARTx\_C3 field descriptions (continued)**

| Field  | Description   |
|--------|---|
|        | 0 OR interrupts disabled; use polling.<br>1 Hardware interrupt requested when OR is set.  |
| 2 NEIE | Noise Error Interrupt Enable<br><br>This bit enables the noise flag (NF) to generate hardware interrupt requests.<br><br>0 NF interrupts disabled; use polling.<br>1 Hardware interrupt requested when NF is set.           |
| 1 FEIE | Framing Error Interrupt Enable<br><br>This bit enables the framing error flag (FE) to generate hardware interrupt requests.<br><br>0 FE interrupts disabled; use polling.<br>1 Hardware interrupt requested when FE is set. |
| 0 PEIE | Parity Error Interrupt Enable<br><br>This bit enables the parity error flag (PF) to generate hardware interrupt requests.<br><br>0 PF interrupts disabled; use polling.<br>1 Hardware interrupt requested when PF is set.   |

**39.2.8 UART Data Register (UARTx\_D)**

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

Address: Base address + 7h offset

| Bit        | 7    | 6    | 5    | 4    | 3    | 2    | 1    | 0    |
|------------|------|------|------|------|------|------|------|------|
| Read Write | R7T7 | R6T6 | R5T5 | R4T4 | R3T3 | R2T2 | R1T1 | R0T0 |
| Reset      | 0    | 0    | 0    | 0    | 0    | 0    | 0    | 0    |

**UARTx\_D field descriptions**

| Field  | Description   |
|--------|---|
| 7 R7T7 | Read receive data buffer 7 or write transmit data buffer 7. |
| 6 R6T6 | Read receive data buffer 6 or write transmit data buffer 6. |
| 5 R5T5 | Read receive data buffer 5 or write transmit data buffer 5. |

*Table continues on the next page...*

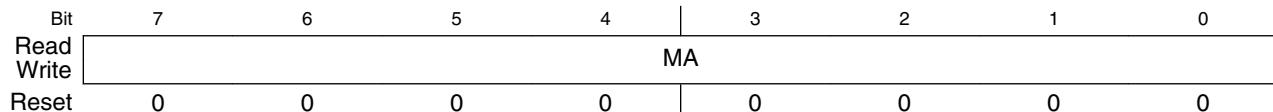
**UARTx\_D field descriptions (continued)**

| Field     | Description   |
|-----------|---|
| 4<br>R4T4 | Read receive data buffer 4 or write transmit data buffer 4. |
| 3<br>R3T3 | Read receive data buffer 3 or write transmit data buffer 3. |
| 2<br>R2T2 | Read receive data buffer 2 or write transmit data buffer 2. |
| 1<br>R1T1 | Read receive data buffer 1 or write transmit data buffer 1. |
| 0<br>R0T0 | Read receive data buffer 0 or write transmit data buffer 0. |

**39.2.9 UART Match Address Registers 1 (UARTx\_MA1)**

The MA1 and MA2 registers are compared to input data addresses when the most significant bit is set and the associated C4[MAEN] bit is set. If a match occurs, the following data is transferred to the data register. If a match fails, the following data is discarded. Software should only write a MA register when the associated C4[MAEN] bit is clear.

Address: Base address + 8h offset

**UARTx\_MA1 field descriptions**

| Field     | Description   |
|-----------|---------------|
| 7–0<br>MA | Match Address |

### 39.2.10 UART Match Address Registers 2 (UARTx\_MA2)

The MA1 and MA2 registers are compared to input data addresses when the most significant bit is set and the associated C4[MAEN] bit is set. If a match occurs, the following data is transferred to the data register. If a match fails, the following data is discarded. Software should only write a MA register when the associated C4[MAEN] bit is clear.

Address: Base address + 9h offset

| Bit           | 7  | 6 | 5 | 4 |   | 3 | 2 | 1 | 0 |
|---------------|----|---|---|---|---|---|---|---|---|
| Read<br>Write | MA |   |   |   |   |   |   |   |   |
| Reset         | 0  | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

#### UARTx\_MA2 field descriptions

| Field     | Description   |  |  |  |  |  |  |  |
|-----------|---------------|--|--|--|--|--|--|--|
| 7–0<br>MA | Match Address |  |  |  |  |  |  |  |

### 39.2.11 UART Control Register 4 (UARTx\_C4)

Address: Base address + Ah offset

| Bit           | 7     | 6     | 5   | 4 |     | 3 | 2 | 1 | 0 |
|---------------|-------|-------|-----|---|-----|---|---|---|---|
| Read<br>Write | MAEN1 | MAEN2 | M10 |   | OSR |   |   |   |   |
| Reset         | 0     | 0     | 0   | 0 | 1   | 1 | 1 | 1 | 1 |

#### UARTx\_C4 field descriptions

| Field      | Description   |  |  |  |  |  |  |  |
|------------|---|--|--|--|--|--|--|--|
| 7<br>MAEN1 | Match Address Mode Enable 1<br>Refer to <a href="#">Match address operation</a> for more information.<br>0 All data received is transferred to the data buffer if MAEN2 is cleared.<br>1 All data received with the most significant bit cleared, is discarded. All data received with the most significant bit set, is compared with contents of MA1 register. If no match occurs, the data is discarded. If match occurs, data is transferred to the data buffer. |  |  |  |  |  |  |  |
| 6<br>MAEN2 | Match Address Mode Enable 2<br>Refer to <a href="#">Match address operation</a> for more information.<br>0 All data received is transferred to the data buffer if MAEN1 is cleared.<br>1 All data received with the most significant bit cleared, is discarded. All data received with the most significant bit set, is compared with contents of MA2 register. If no match occurs, the data is discarded. If match occurs, data is transferred to the data buffer. |  |  |  |  |  |  |  |

Table continues on the next page...

**UARTx\_C4 field descriptions (continued)**

| Field      | Description  |
|------------|--|
| 5<br>M10   | <p>10-bit Mode select</p> <p>The M10 bit causes a tenth bit to be part of the serial transmission. This bit should only be changed when the transmitter and receiver are both disabled.</p> <p>0 Receiver and transmitter use 8-bit or 9-bit data characters.<br/>1 Receiver and transmitter use 10-bit data characters.</p> |
| 4–0<br>OSR | <p>Over Sampling Ratio</p> <p>This field configures the oversampling ratio for the receiver between 4x (00011) and 32x (11111). Writing an invalid oversampling ratio will default to an oversampling ratio of 16 (01111). This field should only be changed when the transmitter and receiver are both disabled.</p>        |

**39.2.12 UART Control Register 5 (UARTx\_C5)**

Address: Base address + Bh offset

**UARTx\_C5 field descriptions**

| Field           | Description   |
|-----------------|---|
| 7<br>TDMAE      | <p>Transmitter DMA Enable</p> <p>TDMAE configures the transmit data register empty flag, S1[TDRE], to generate a DMA request.</p> <p>0 DMA request disabled.<br/>1 DMA request enabled.</p>   |
| 6<br>Reserved   | <p>This field is reserved.</p> <p>This read-only field is reserved and always has the value 0.</p>  |
| 5<br>RDMAE      | <p>Receiver Full DMA Enable</p> <p>RDMAE configures the receiver data register full flag, S1[RDRF], to generate a DMA request.</p> <p>0 DMA request disabled.<br/>1 DMA request enabled.</p>  |
| 4–2<br>Reserved | <p>This field is reserved.</p> <p>This read-only field is reserved and always has the value 0.</p>  |
| 1<br>BOTHEDGE   | <p>Both Edge Sampling</p> <p>Enables sampling of the received data on both edges of the baud rate clock, effectively doubling the number of times the receiver samples the input data for a given oversampling ratio. This bit must be set for oversampling ratios between x4 and x7 and is optional for higher oversampling ratios. This bit should only be changed when the receiver is disabled.</p> |

*Table continues on the next page...*

**UARTx\_C5 field descriptions (continued)**

| Field          | Description   |
|----------------|---|
|                | <p>0 Receiver samples input data using the rising edge of the baud rate clock.<br/>1 Receiver samples input data using the rising and falling edge of the baud rate clock.</p>  |
| 0<br>RESYNCDIS | <p>Resynchronization Disable<br/><br/>When set, disables the resynchronization of the received data word when a data one followed by data zero transition is detected. This bit should only be changed when the receiver is disabled.</p> <p>0 Resynchronization during received data word is supported<br/>1 Resynchronization during received data word is disabled</p> |

## 39.3 Functional description

The UART supports full-duplex, asynchronous, NRZ serial communication and comprises a baud rate generator, transmitter, and receiver block. The transmitter and receiver operate independently, although they use the same baud rate generator. The following describes each of the blocks of the UART.

### 39.3.1 Baud rate generation

A 13-bit modulus counter in the baud rate generator derive the baud rate for both the receiver and the transmitter. The value from 1 to 8191 written to SBR[12:0] determines the baud clock divisor for the asynchronous UART baud clock. The SBR bits are in the UART baud rate registers, BDH and BDL. The baud rate clock drives the receiver, while the transmitter is driven by the baud rate clock divided by the over sampling ratio. Depending on the over sampling ratio, the receiver has an acquisition rate of 4 to 32 samples per bit time.

Baud rate generation is subject to two sources of error:

- Integer division of the module clock may not give the exact target frequency.
- Synchronization with the asynchronous UART baud clock can cause phase shift.

### 39.3.2 Transmitter functional description

This section describes the overall block diagram for the UART transmitter, as well as specialized functions for sending break and idle characters.

The transmitter output (UART\_TX) idle state defaults to logic high, C3[TXINV] is cleared following reset. The transmitter output is inverted by setting C3[TXINV]. The transmitter is enabled by setting the C2[TE] bit. This queues a preamble character that is one full character frame of the idle state. The transmitter then remains idle until data is available in the transmit data buffer. Programs store data into the transmit data buffer by writing to the UART data register.

The central element of the UART transmitter is the transmit shift register that is 10-bit to 13 bits long depending on the setting in the C1[M], C2[M10] and BDH[SBNS] control bits. For the remainder of this section, assume C1[M], C2[M10] and BDH[SBNS] are cleared, selecting the normal 8-bit data mode. In 8-bit data mode, the shift register holds a start bit, eight data bits, and a stop bit. When the transmit shift register is available for a new UART character, the value waiting in the transmit data register is transferred to the shift register, synchronized with the baud rate clock, and the transmit data register empty (S1[TDRE]) status flag is set to indicate another character may be written to the transmit data buffer at UART\_D.

If no new character is waiting in the transmit data buffer after a stop bit is shifted out the UART\_TX pin, the transmitter sets the transmit complete flag and enters an idle mode, with UART\_TX high, waiting for more characters to transmit.

Writing 0 to C2[TE] does not immediately disable the transmitter. The current transmit activity in progress must first be completed. This includes data characters in progress, queued idle characters, and queued break characters.

### 39.3.2.1 Send break and queued idle

The UART\_C2[SBK] bit sends break characters originally used to gain the attention of old teletype receivers. Break characters are a full character time of logic 0, 10-bit to 12-bit times including the start and stop bits. A longer break of 13-bit times can be enabled by setting UART\_S2[BRK13]. Normally, a program would wait for UART\_S1[TDRE] to become set to indicate the last character of a message has moved to the transmit shifter, write 1, and then write 0 to the UART\_C2[SBK] bit. This action queues a break character to be sent as soon as the shifter is available. If UART\_C2[SBK] remains 1 when the queued break moves into the shifter, synchronized to the baud rate clock, an additional break character is queued. If the receiving device is another Freescale Semiconductor UART, the break characters are received as 0s in all data bits and a framing error (UART\_S1[FE] = 1) occurs.

When idle-line wakeup is used, a full character time of idle (logic 1) is needed between messages to wake up any sleeping receivers. Normally, a program would wait for UART\_S1[TDRE] to become set to indicate the last character of a message has moved to

the transmit shifter, then write 0 and then write 1 to the UART\_C2[TE] bit. This action queues an idle character to be sent as soon as the shifter is available. As long as the character in the shifter does not finish while UART\_C2[TE] is cleared, the UART transmitter never actually releases control of the UART\_TX pin.

The length of the break character is affected by the UART\_S2[BRK13], UART\_C1[M] and UART\_C4[M10] bits as shown below.

**Table 39-27. Break character length**

| BRK13 | M | M10 | SBNS | Break character length |
|-------|---|-----|------|------------------------|
| 0     | 0 | 0   | 0    | 10 bit times           |
| 0     | 0 | 0   | 1    | 11 bit times           |
| 0     | 1 | 0   | 0    | 11 bit times           |
| 0     | 1 | 0   | 1    | 12 bit times           |
| 0     | X | 1   | 0    | 12 bit times           |
| 0     | X | 1   | 1    | 13 bit times           |
| 1     | 0 | 0   | 0    | 13 bit times           |
| 1     | 0 | 0   | 1    | 14 bit times           |
| 1     | 1 | 0   | 0    | 14 bit times           |
| 1     | 1 | 0   | 1    | 15 bit times           |
| 1     | X | 1   | 0    | 15 bit times           |
| 1     | X | 1   | 1    | 16 bit times           |

### 39.3.3 Receiver functional description

In this section, the receiver block diagram is a guide for the overall receiver functional description. Next, the data sampling technique used to reconstruct receiver data is described in more detail. Finally, two variations of the receiver wakeup function are explained.

The receiver input is inverted by setting UART\_S2[RXINV]. The receiver is enabled by setting the UART\_C2[RE] bit. Character frames consist of a start bit of logic 0, eight to ten data bits (msb or lsb first), and one or two stop bits of logic 1. For information about 9-bit or 10-bit data mode, refer to [8-bit, 9-bit and 10-bit data modes](#). For the remainder of this discussion, assume the UART is configured for normal 8-bit data mode.

After receiving the stop bit into the receive shifter, and provided the receive data register is not already full, the data character is transferred to the receive data register and the receive data register full (UART\_S1[RDRF]) status flag is set. If UART\_S1[RDRF] was already set indicating the receive data register (buffer) was already full, the overrun (OR)

status flag is set and the new data is lost. Because the UART receiver is double-buffered, the program has one full character time after `UART_S1[RDRF]` is set before the data in the receive data buffer must be read to avoid a receiver overrun.

When a program detects that the receive data register is full (`UART_S1[RDRF] = 1`), it gets the data from the receive data register by reading `UART_D`. Refer to [Interrupts and status flags](#) for details about flag clearing.

### 39.3.3.1 Data sampling technique

The UART receiver supports an oversampling rate of between  $4\times$  and  $32\times$  of the baud rate clock for sampling. The receiver starts by taking logic level samples at the oversampling rate times the baud rate to search for a falling edge on the `UART_RX` serial data input pin. A falling edge is defined as a logic 0 sample after three consecutive logic 1 samples. The oversampling baud rate clock divides the bit time into 4 to 32 segments from 1 to OSR (where OSR is the configured oversampling ratio). When a falling edge is located, three more samples are taken at  $(OSR/2)$ ,  $(OSR/2)+1$ , and  $(OSR/2)+2$  to make sure this was a real start bit and not merely noise. If at least two of these three samples are 0, the receiver assumes it is synchronized to a receive character. If another falling edge is detected before the receiver is considered synchronized, the receiver restarts the sampling from the first segment.

The receiver then samples each bit time, including the start and stop bits, at  $(OSR/2)$ ,  $(OSR/2)+1$ , and  $(OSR/2)+2$  to determine the logic level for that bit. The logic level is interpreted to be that of the majority of the samples taken during the bit time. If any sample in any bit time, including the start and stop bits, in a character frame fails to agree with the logic level for that bit, the noise flag (`UART_S1[NF]`) is set when the received character is transferred to the receive data buffer.

When the UART receiver is configured to sample on both edges of the baud rate clock, the number of segments in each received bit is effectively doubled (from 1 to  $OSR*2$ ). The start and data bits are then sampled at  $OSR$ ,  $OSR+1$  and  $OSR+2$ . Sampling on both edges of the clock must be enabled for oversampling rates of  $4\times$  to  $7\times$  and is optional for higher oversampling rates.

The falling edge detection logic continuously looks for falling edges. If an edge is detected, the sample clock is resynchronized to bit times (unless resynchronization has been disabled). This improves the reliability of the receiver in the presence of noise or mismatched baud rates. It does not improve worst case analysis because some characters do not have any extra falling edges anywhere in the character frame.

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

### **39.3.3.2 Receiver wakeup operation**

Receiver wakeup is a hardware mechanism that allows an UART receiver to ignore the characters in a message intended for a different UART receiver. In such a system, all receivers evaluate the first character(s) of each message, and as soon as they determine the message is intended for a different receiver, they write logic 1 to the receiver wake up control bit(UART\_C2[RWU]). When RWU bit is set, the status flags associated with the receiver, with the exception of the idle bit, IDLE, when UART\_S2[RWUID] bit is set, are inhibited from setting, thus eliminating the software overhead for handling the unimportant message characters. At the end of a message, or at the beginning of the next message, all receivers automatically force UART\_C2[RWU] to 0 so all receivers wake up in time to look at the first character(s) of the next message.

#### **39.3.3.2.1 Idle-line wakeup**

When wake is cleared, the receiver is configured for idle-line wakeup. In this mode, UART\_C2[RWU] is cleared automatically when the receiver detects a full character time of the idle-line level. The UART\_C1[M] and UART\_C4[M10] control bit selects 8-bit to 10-bit data mode and the UART\_BDH[SBNS] bit selects 1-bit or 2-bit stop bit number that determines how many bit times of idle are needed to constitute a full character time, 10 to 13 bit times because of the start and stop bits.

When UART\_C2[RWU] is one and UART\_S2[RWUID] is zero, the idle condition that wakes up the receiver does not set the UART\_S1[IDLE] flag. The receiver wakes up and waits for the first data character of the next message that sets the UART\_S1[RDRF] flag and generates an interrupt if enabled. When UART\_S2[RWUID] is one, any idle condition sets the UART\_S1[IDLE] flag and generates an interrupt if enabled, regardless of whether UART\_C2[RWU] is zero or one.

The idle-line type (UART\_C1[ILT]) control bit selects one of two ways to detect an idle line. When UART\_C1[ILT] is cleared, the idle bit counter starts after the start bit so the stop bit and any logic 1s at the end of a character count toward the full character time of idle. When UART\_C1[ILT] is set, the idle bit counter does not start until after the stop bit time, so the idle detection is not affected by the data in the last character of the previous message.

### 39.3.3.2.2 Address-mark wakeup

When wake is set, the receiver is configured for address-mark wakeup. In this mode, UART\_C2[RWU] is cleared automatically when the receiver detects a logic 1 in the most significant bit of a received character.

Address-mark wakeup allows messages to contain idle characters, but requires the msb be reserved for use in address frames. The logic 1 in the msb of an address frame clears the UART\_C2[RWU] bit before the stop bits are received and sets the UART\_S1[RDRF] flag. In this case, the character with the msb set is received even though the receiver was sleeping during most of this character time.

### 39.3.3.2.3 Match address operation

Match address operation is enabled when the UART\_C4[MAEN1] or UART\_C4[MAEN2] bit is set. In this function, a frame received by the UART\_RX pin with a logic 1 in the bit position immediately preceding the stop bit is considered an address and is compared with the associated MA1 or MA2 register. The frame is only transferred to the receive buffer, and UART\_S1[RDRF] is set, if the comparison matches. All subsequent frames received with a logic 0 in the bit position immediately preceding the stop bit are considered to be data associated with the address and are transferred to the receive data buffer. If no marked address match occurs then no transfer is made to the receive data buffer, and all following frames with logic zero in the bit position immediately preceding the stop bit are also discarded. If both the UART\_C4[MAEN1] and UART\_C4[MAEN2] bits are negated, the receiver operates normally and all data received is transferred to the receive data buffer.

Match Address operation functions in the same way for both MA1 and MA2 registers.

- If only one of UART\_C4[MAEN1] and UART\_C4[MAEN2] is asserted, a marked address is compared only with the associated match register and data is transferred to the receive data buffer only on a match.
- If UART\_C4[MAEN1] and UART\_C4[MAEN2] are asserted, a marked address is compared with both match registers and data is transferred only on a match with either register.

## 39.3.4 Additional UART functions

The following sections describe additional UART functions.

### 39.3.4.1 8-bit, 9-bit and 10-bit data modes

The UART system, transmitter and receiver, can be configured to operate in 9-bit data mode by setting the `UART_C1[M]` or 10-bit data mode by setting `UART_C4[M10]`. In 9-bit mode, there is a ninth data bit to the left of the msb of the UART data register, in 10-bit mode there is a tenth data bit. For the transmit data buffer, these bits are stored in `T8` and `T9` in `UART_C3`. For the receiver, these bits are held in `UART_C3[R8]` and `UART_C3[R9]`.

For coherent writes to the transmit data buffer, write to `UART_C3[T8]` and `UART_C3[T9]` before writing to `UART_D`.

If the bit values to be transmitted as the ninth and tenth bit of a new character are the same as for the previous character, it is not necessary to write to `T8` and `T9` again. When data is transferred from the transmit data buffer to the transmit shifter, the value in `T8` and `T9` is copied at the same time data is transferred from `UART_D` to the shifter.

The 9-bit data mode is typically used with parity to allow eight bits of data plus the parity in the ninth bit, or it is used with address-mark wakeup so the ninth data bit can serve as the wakeup bit. The 10-bit data mode is typically used with parity and address-mark wakeup so the ninth data bit can serve as the wakeup bit and the tenth bit as the parity bit. In custom protocols, the ninth and/or tenth bits can also serve as software-controlled markers.

### 39.3.4.2 Loop mode

When `UART_C1[LOOPS]` is set, the `UART_C1[RSRC]` bit in the same register chooses between loop mode (`UART_C1[RSRC] = 0`) or single-wire mode (`UART_C1[RSRC] = 1`). Loop mode is sometimes used to check software, independent of connections in the external system, to help isolate system problems. In this mode, the transmitter output is internally connected to the receiver input and the `UART_RX` pin is not used by the UART.

### 39.3.4.3 Single-wire operation

When `UART_C1[LOOPS]` is set, the `RSRC` bit in the same register chooses between loop mode (`UART_C1[RSRC] = 0`) or single-wire mode (`UART_C1[RSRC] = 1`). Single-wire mode implements a half-duplex serial connection. The receiver is internally connected to the transmitter output and to the `UART_TX` pin (the `UART_RX` pin is not used).

In single-wire mode, the UART\_C3[TXDIR] bit controls the direction of serial data on the UART\_TX pin. When UART\_C3[TXDIR] is cleared, the UART\_TX pin is an input to the UART receiver and the transmitter is temporarily disconnected from the UART\_TX pin so an external device can send serial data to the receiver. When UART\_C3[TXDIR] is set, the UART\_TxD pin is an output driven by the transmitter, the internal loop back connection is disabled, and as a result the receiver cannot receive characters that are sent out by the transmitter.

### 39.3.5 Interrupts and status flags

The UART system generates three separate interrupts to reduce the amount of software needed to isolate the cause of the interrupt. One interrupt is associated with the transmitter for TDRE and TC events. Another interrupt is associated with the receiver for RDRF, IDLE, RXEDGIF, and LBKDIF events. A third interrupt is used for OR, NF, FE, and PF error conditions. Each of these ten interrupt sources can be separately masked by local interrupt enable masks. The flags can be polled by software when the local masks are cleared to disable generation of hardware interrupt requests.

The UART transmitter has two status flags that can optionally generate hardware interrupt requests. Transmit data register empty (UART\_S1[TDRE]) indicates when there is room in the transmit data buffer to write another transmit character to UART\_D. If the transmit interrupt enable (UART\_C2[TIE]) bit is set, a hardware interrupt is requested when UART\_S1[TDRE] is set. Transmit complete (UART\_S1[TC]) indicates that the transmitter is finished transmitting all data, preamble, and break characters and is idle with UART\_TX at the inactive level. This flag is often used in systems with modems to determine when it is safe to turn off the modem. If the transmit complete interrupt enable (UART\_C2[TCIE]) bit is set, a hardware interrupt is requested when UART\_S1[TC] is set. Instead of hardware interrupts, software polling may be used to monitor the UART\_S1[TDRE] and UART\_S1[TC] status flags if the corresponding UART\_C2[TIE] or UART\_C2[TCIE] local interrupt masks are cleared.

When a program detects that the receive data register is full (UART\_S1[RDRF] = 1), it gets the data from the receive data register by reading UART\_D. The UART\_S1[RDRF] flag is cleared by reading UART\_D.

The IDLE status flag includes logic that prevents it from getting set repeatedly when the UART\_RX line remains idle for an extended period of time. IDLE is cleared by writing 1 to the UART\_S1[IDLE] flag. After UART\_S1[IDLE] has been cleared, it cannot become set again until the receiver has received at least one new character and has set UART\_S1[RDRF].

## Functional description

If the associated error was detected in the received character that caused UART\_S1[RDRF] to be set, the error flags - noise flag (UART\_S1[NF]), framing error (UART\_S1[FE]), and parity error flag (UART\_S1[PF]) - are set at the same time as UART\_S1[RDRF]. These flags are not set in overrun cases.

If UART\_S1[RDRF] was already set when a new character is ready to be transferred from the receive shifter to the receive data buffer, the overrun (UART\_S1[OR]) flag is set instead of the data along with any associated NF, FE, or PF condition is lost.

At any time, an active edge on the UART\_RX serial data input pin causes the UART\_S2[RXEDGIF] flag to set. The UART\_S2[RXEDGIF] flag is cleared by writing a 1 to it. This function depends on the receiver being enabled (UART\_C2[RE] = 1).

# **Chapter 40**

## **Universal Asynchronous Receiver/Transmitter (UART1 and UART2)**

### **40.1 Introduction**

#### **40.1.1 Features**

Features of UART module include:

- Full-duplex, standard non-return-to-zero (NRZ) format
- Double-buffered transmitter and receiver with separate enables
- Programmable baud rates (13-bit modulo divider)
- Interrupt-driven or polled operation:
  - Transmit data register empty and transmission complete
  - Receive data register full
  - Receive overrun, parity error, framing error, and noise error
  - Idle receiver detect
  - Active edge on receive pin
  - Break detect supporting LIN
- Hardware parity generation and checking
- Programmable 8-bit or 9-bit character length
- Programmable 1-bit or 2-bit stop bits
- Receiver wakeup by idle-line or address-mark
- Optional 13-bit break character generation / 11-bit break character detection
- Selectable transmitter output polarity
- 5-channel DMA interface

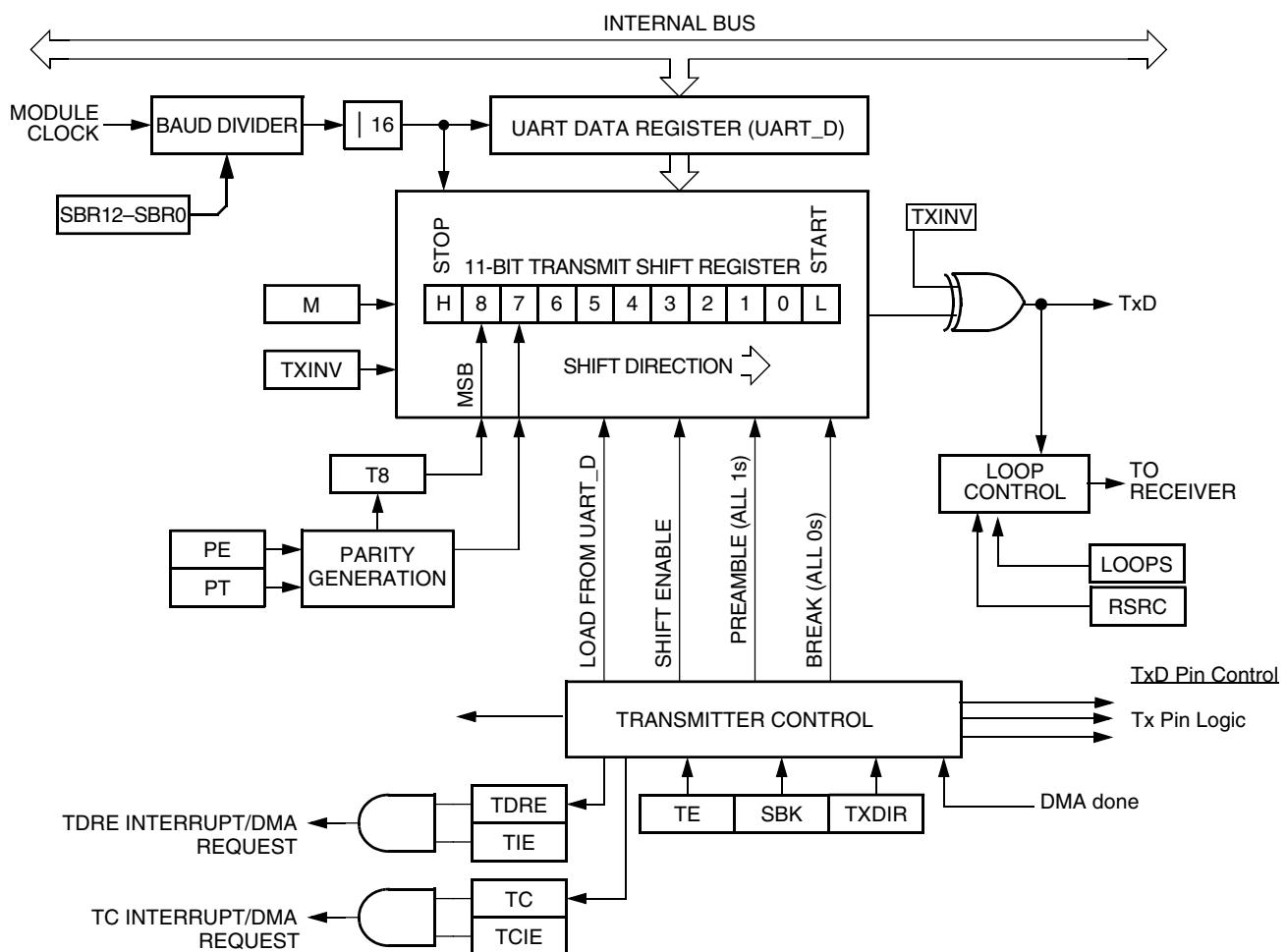
## 40.1.2 Modes of operation

See Section [Functional description](#) for details concerning UART operation in these modes:

- 8- and 9-bit data modes
- Stop mode operation
- Loop mode
- Single-wire mode

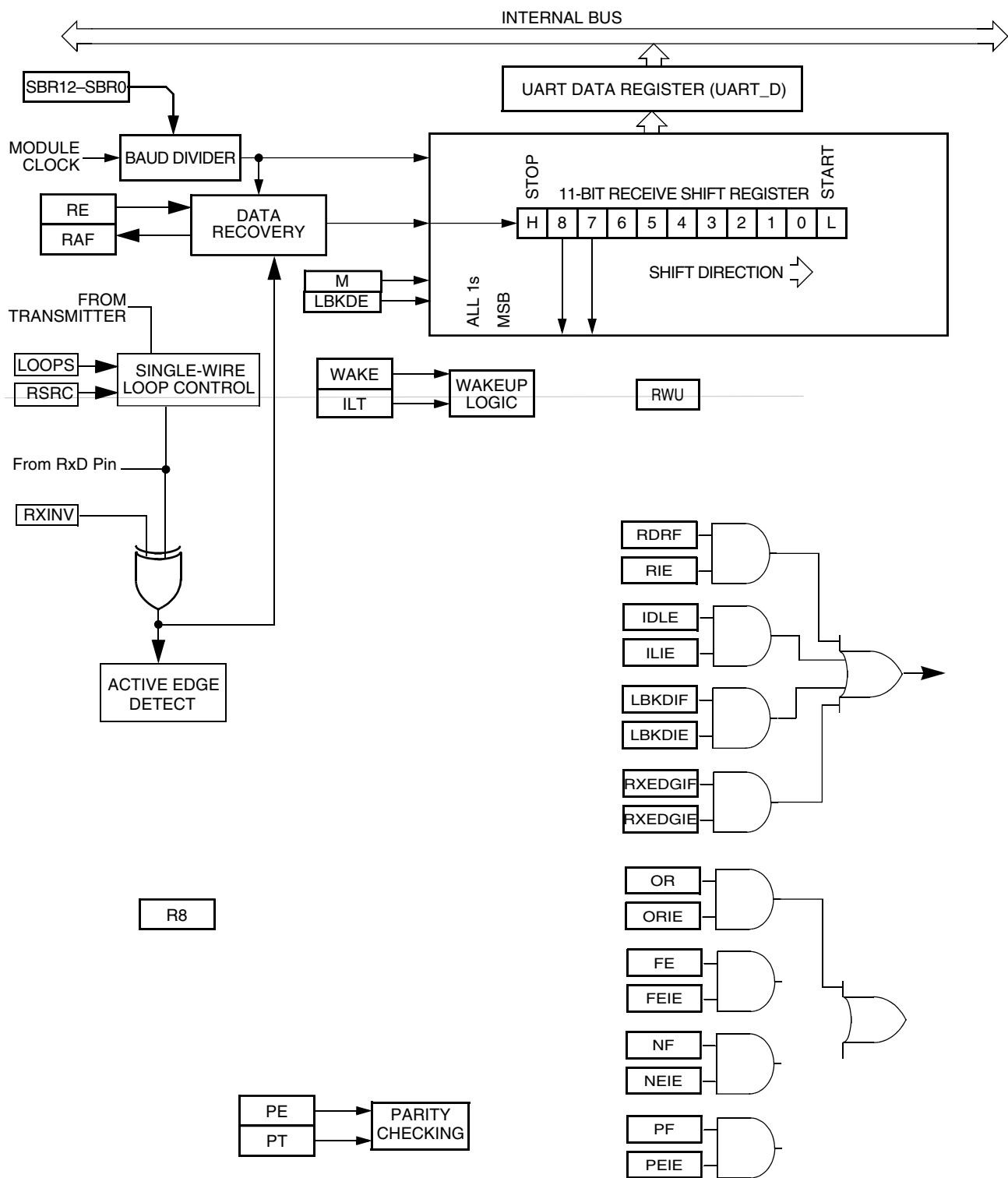
## 40.1.3 Block diagram

The following figure shows the transmitter portion of the UART.



**Figure 40-1. UART transmitter block diagram**

The following figure shows the receiver portion of the UART.



## 40.2 Register definition

The UART has 8-bit registers to control baud rate, select UART options, report UART status, select DMA options, and for transmit/receive data.

Refer to the direct-page register summary in the memory chapter of this document or the absolute address assignments for all UART registers. This section refers to registers and control bits only by their names. A Freescale-provided equate or header file is used to translate these names into the appropriate absolute addresses.

**UART memory map**

| Absolute address (hex) | Register name                             | Width (in bits) | Access | Reset value | Section/page               |
|------------------------|---|-----------------|--------|-------------|----------------------------|
| 4006_B000              | UART Baud Rate Register: High (UART1_BDH) | 8               | R/W    | 00h         | <a href="#">40.2.1/751</a> |
| 4006_B001              | UART Baud Rate Register: Low (UART1_BDL)  | 8               | R/W    | 04h         | <a href="#">40.2.2/751</a> |
| 4006_B002              | UART Control Register 1 (UART1_C1)        | 8               | R/W    | 00h         | <a href="#">40.2.3/752</a> |
| 4006_B003              | UART Control Register 2 (UART1_C2)        | 8               | R/W    | 00h         | <a href="#">40.2.4/753</a> |
| 4006_B004              | UART Status Register 1 (UART1_S1)         | 8               | R      | C0h         | <a href="#">40.2.5/755</a> |
| 4006_B005              | UART Status Register 2 (UART1_S2)         | 8               | R/W    | 00h         | <a href="#">40.2.6/756</a> |
| 4006_B006              | UART Control Register 3 (UART1_C3)        | 8               | R/W    | 00h         | <a href="#">40.2.7/758</a> |
| 4006_B007              | UART Data Register (UART1_D)              | 8               | R/W    | 00h         | <a href="#">40.2.8/760</a> |
| 4006_B008              | UART Control Register 4 (UART1_C4)        | 8               | R/W    | 00h         | <a href="#">40.2.9/760</a> |
| 4006_C000              | UART Baud Rate Register: High (UART2_BDH) | 8               | R/W    | 00h         | <a href="#">40.2.1/751</a> |
| 4006_C001              | UART Baud Rate Register: Low (UART2_BDL)  | 8               | R/W    | 04h         | <a href="#">40.2.2/751</a> |
| 4006_C002              | UART Control Register 1 (UART2_C1)        | 8               | R/W    | 00h         | <a href="#">40.2.3/752</a> |
| 4006_C003              | UART Control Register 2 (UART2_C2)        | 8               | R/W    | 00h         | <a href="#">40.2.4/753</a> |
| 4006_C004              | UART Status Register 1 (UART2_S1)         | 8               | R      | C0h         | <a href="#">40.2.5/755</a> |
| 4006_C005              | UART Status Register 2 (UART2_S2)         | 8               | R/W    | 00h         | <a href="#">40.2.6/756</a> |
| 4006_C006              | UART Control Register 3 (UART2_C3)        | 8               | R/W    | 00h         | <a href="#">40.2.7/758</a> |
| 4006_C007              | UART Data Register (UART2_D)              | 8               | R/W    | 00h         | <a href="#">40.2.8/760</a> |
| 4006_C008              | UART Control Register 4 (UART2_C4)        | 8               | R/W    | 00h         | <a href="#">40.2.9/760</a> |

## 40.2.1 UART Baud Rate Register: High (UARTx\_BDH)

This register, along with UART\_BDL, controls the prescale divisor for UART baud rate generation. To update the 13-bit baud rate setting [SBR12:SBR0], first write to UART\_BDH to buffer the high half of the new value and then write to UART\_BDL. The working value in UART\_BDH does not change until UART\_BDL is written.

Address: Base address + h offset

| Bit   | 7       | 6       | 5    | 4 |  | 3   | 2 | 1 | 0 |
|-------|---------|---------|------|---|--|-----|---|---|---|
| Read  | LBKDIIE | RXEDGIE | SBNS |   |  | SBR |   |   |   |
| Write | 0       | 0       | 0    | 0 |  | 0   | 0 | 0 | 0 |
| Reset | 0       | 0       | 0    | 0 |  | 0   | 0 | 0 | 0 |

### UARTx\_BDH field descriptions

| Field        | Description   |
|--------------|---|
| 7<br>LBKDIIE | LIN Break Detect Interrupt Enable (for LBKDIF)<br>0 Hardware interrupts from UART_S2[LBKDIF] disabled (use polling).<br>1 Hardware interrupt requested when UART_S2[LBKDIF] flag is 1.  |
| 6<br>RXEDGIE | RxD Input Active Edge Interrupt Enable (for RXEDGIF)<br>0 Hardware interrupts from UART_S2[RXEDGIF] disabled (use polling).<br>1 Hardware interrupt requested when UART_S2[RXEDGIF] flag is 1.  |
| 5<br>SBNS    | Stop Bit Number Select<br>SBNS determines whether data characters are one or two stop bits.<br>0 One stop bit.<br>1 Two stop bit.   |
| 4–0<br>SBR   | Baud Rate Modulo Divisor.<br>The 13 bits in SBR[12:0] are referred to collectively as BR, and they set the modulo divide rate for the UART baud rate generator. When BR is cleared, the UART baud rate generator is disabled to reduce supply current. When BR is 1 - 8191, the UART baud rate equals BUSCLK/(16×BR). |

## 40.2.2 UART Baud Rate Register: Low (UARTx\_BDL)

This register, along with UART\_BDH, control the prescale divisor for UART baud rate generation. To update the 13-bit baud rate setting [SBR12:SBR0], first write to UART\_BDH to buffer the high half of the new value and then write to UART\_BDL. The working value in UART\_BDH does not change until UART\_BDL is written.

UART\_BDL is reset to a non-zero value, so after reset the baud rate generator remains disabled until the first time the receiver or transmitter is enabled; that is, UART\_C2[RE] or UART\_C2[TE] bits are written to 1.

Address: Base address + h offset

| Bit        | 7   | 6 | 5 | 4 |  | 3 | 2 | 1 | 0 |
|------------|-----|---|---|---|--|---|---|---|---|
| Read Write | SBR |   |   |   |  |   |   |   |   |
| Reset      | 0   | 0 | 0 | 0 |  | 0 | 1 | 0 | 0 |

### UARTx\_BDL field descriptions

| Field      | Description   |
|------------|---|
| 7–0<br>SBR | Baud Rate Modulo Divisor<br><br>These 13 bits in SBR[12:0] are referred to collectively as BR. They set the modulo divide rate for the UART baud rate generator. When BR is cleared, the UART baud rate generator is disabled to reduce supply current. When BR is 1 - 8191, the UART baud equals BUSCLK/(16×BR). |

### 40.2.3 UART Control Register 1 (UARTx\_C1)

This read/write register controls various optional features of the UART system.

Address: Base address + h offset

| Bit        | 7     | 6        | 5    | 4 |  | 3     | 2   | 1  | 0  |
|------------|-------|----------|------|---|--|-------|-----|----|----|
| Read Write | LOOPS | UARTSWAI | RSRC | M |  | WAKEF | ILT | PF | PT |
| Reset      | 0     | 0        | 0    | 0 |  | 0     | 0   | 0  | 0  |

**UARTx\_C1 field descriptions (continued)**

| Field     | Description  |
|-----------|--|
|           | <p>This bit has no meaning or effect unless the LOOPS bit is set to 1. When LOOPS is set, the receiver input is internally connected to the TxD pin and RSRC determines whether this connection is also connected to the transmitter output.</p> <ul style="list-style-type: none"> <li>0 Provided LOOPS is set, RSRC is cleared, selects internal loop back mode and the UART does not use the RxD pins.</li> <li>1 Single-wire UART mode where the TxD pin is connected to the transmitter output and receiver input.</li> </ul> |
| 4<br>M    | <p>9-Bit or 8-Bit Mode Select</p> <ul style="list-style-type: none"> <li>0 Normal - start + 8 data bits (lsb first) + stop.</li> <li>1 Receiver and transmitter use 9-bit data characters start + 8 data bits (lsb first) + 9th data bit + stop.</li> </ul>  |
| 3<br>WAKE | <p>Receiver Wakeup Method Select</p> <ul style="list-style-type: none"> <li>0 Idle-line wakeup.</li> <li>1 Address-mark wakeup.</li> </ul>   |
| 2<br>ILT  | <p>Idle Line Type Select</p> <p>Setting this bit to 1 ensures that the stop bits and logic 1 bits at the end of a character do not count toward the 10 or 11 bit times of logic high level needed by the idle line detection logic.</p> <ul style="list-style-type: none"> <li>0 Idle character bit count starts after start bit.</li> <li>1 Idle character bit count starts after stop bit.</li> </ul>  |
| 1<br>PE   | <p>Parity Enable</p> <p>Enables hardware parity generation and checking. When parity is enabled, the most significant bit (msb) of the data character, eighth or ninth data bit, is treated as the parity bit.</p> <ul style="list-style-type: none"> <li>0 No hardware parity generation or checking.</li> <li>1 Parity enabled.</li> </ul>   |
| 0<br>PT   | <p>Parity Type</p> <p>Provided parity is enabled (PE = 1), this bit selects even or odd parity. Odd parity means the total number of 1s in the data character, including the parity bit, is odd. Even parity means the total number of 1s in the data character, including the parity bit, is even.</p> <ul style="list-style-type: none"> <li>0 Even parity.</li> <li>1 Odd parity.</li> </ul>  |

**40.2.4 UART Control Register 2 (UARTx\_C2)**

This register can be read or written at any time.

Address: Base address + h offset

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

**UARTx\_C2 field descriptions**

| Field     | Description   |
|-----------|---|
| 7<br>TIE  | Transmit Interrupt Enable for TDRE<br><br>0 Hardware interrupts from TDRE disabled; use polling.<br>1 Hardware interrupt requested when TDRE flag is 1.   |
| 6<br>TCIE | Transmission Complete Interrupt Enable for TC<br><br>0 Hardware interrupts from TC disabled; use polling.<br>1 Hardware interrupt requested when TC flag is 1.  |
| 5<br>RIE  | Receiver Interrupt Enable for RDRF<br><br>0 Hardware interrupts from RDRF disabled; use polling.<br>1 Hardware interrupt requested when RDRF flag is 1.   |
| 4<br>ILIE | Idle Line Interrupt Enable for IDLE<br><br>0 Hardware interrupts from IDLE disabled; use polling.<br>1 Hardware interrupt requested when IDLE flag is 1.  |
| 3<br>TE   | Transmitter Enable<br><br>TE must be 1 to use the UART transmitter. When TE is set, the UART forces the TxD pin to act as an output for the UART system.<br><br>When the UART is configured for single-wire operation (LOOPS = RSRC = 1), TXDIR controls the direction of traffic on the single UART communication line (TxD pin).<br><br>TE can also queue an idle character by clearing TE then setting TE while a transmission is in progress.<br><br>When TE is written to 0, the transmitter keeps control of the port TxD pin until any data, queued idle, or queued break character finishes transmitting before allowing the pin to revert to a general-purpose I/O pin.<br><br>0 Transmitter off.<br>1 Transmitter on. |
| 2<br>RE   | Receiver Enable<br><br>When the UART receiver is off, the RxD pin reverts to being a general-purpose port I/O pin. If LOOPS is set the RxD pin reverts to being a general-purpose I/O pin even if RE is set.<br><br>0 Receiver off.<br>1 Receiver on.   |
| 1<br>RWU  | Receiver Wakeup Control<br><br>This bit can be written to 1 to place the UART receiver in a standby state where it waits for automatic hardware detection of a selected wakeup condition. The wakeup condition is an idle line between messages, WAKE = 0, idle-line wakeup, or a logic 1 in the most significant data bit in a character, WAKE = 1, address-mark wakeup. Application software sets RWU and, normally, a selected hardware condition automatically clears RWU.<br><br>0 Normal UART receiver operation.<br>1 UART receiver in standby waiting for wakeup condition.   |
| 0<br>SBK  | Send Break<br><br>Writing a 1 and then a 0 to SBK queues a break character in the transmit data stream. Additional break characters of 10 or 11 or 12, 13 or 14 or 15 if BRK13 = 1, bit times of logic 0 are queued as long as SBK is set. Depending on the timing of the set and clear of SBK relative to the information currently being transmitted, a second break character may be queued before software clears SBK.  |

*Table continues on the next page...*

**UARTx\_C2 field descriptions (continued)**

| Field | Description   |
|-------|---|
|       | 0 Normal transmitter operation.<br>1 Queue break character(s) to be sent. |

**40.2.5 UART Status Register 1 (UARTx\_S1)**

This register has eight read-only status flags. Writes have no effect. Special software sequences, which do not involve writing to this register, clear these status flags.

Address: Base address + h offset

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

**UARTx\_S1 field descriptions**

| Field  | Description   |
|--------|---|
| 7 TDRE | Transmit Data Register Empty Flag<br><br>TDRE is set out of reset and when a transmit data value transfers from the transmit data buffer to the transmit shifter, leaving room for a new character in the buffer. To clear TDRE, read UART_S1 with TDRE set and then write to the UART data register (UART_D).<br><br>0 Transmit data register (buffer) full.<br>1 Transmit data register (buffer) empty.   |
| 6 TC   | Transmission Complete Flag<br><br>TC is set out of reset and when TDRE is set and no data, preamble, or break character is being transmitted.<br>TC is cleared automatically by reading UART_S1 with TC set and then doing one of the following: <ul style="list-style-type: none"><li>• Write to the UART data register (UART_D) to transmit new data</li><li>• Queue a preamble by changing TE from 0 to 1</li><li>• Queue a break character by writing 1 to UART_C2[SBK]</li></ul><br>0 Transmitter active (sending data, a preamble, or a break).<br>1 Transmitter idle (transmission activity complete). |
| 5 RDRF | Receive Data Register Full Flag<br><br>RDRF becomes set when a character transfers from the receive shifter into the receive data register (UART_D). To clear RDRF, read UART_S1 with RDRF set and then read the UART data register (UART_D).<br><br>0 Receive data register empty.<br>1 Receive data register full.  |
| 4 IDLE | Idle Line Flag<br><br>IDLE is set when the UART receive line becomes idle for a full character time after a period of activity. When ILT is cleared, the receiver starts counting idle bit times after the start bit. If the receive character is   |

*Table continues on the next page...*

**UARTx\_S1 field descriptions (continued)**

| Field   | Description  |
|---------|--|
|         | <p>all 1s, these bit times and the stop bits time count toward the full character time of logic high, 10 or 11 bit times depending on the M control bit, needed for the receiver to detect an idle line. When ILT is set, the receiver doesn't start counting idle bit times until after the stop bits. The stop bits and any logic high bit times at the end of the previous character do not count toward the full character time of logic high needed for the receiver to detect an idle line.</p> <p>To clear IDLE, read UART_S1 with IDLE set and then read the UART data register (UART_D). After IDLE has been cleared, it cannot become set again until after a new character has been received and RDRF has been set. IDLE is set only once even if the receive line remains idle for an extended period.</p> <ul style="list-style-type: none"> <li>0 No idle line detected.</li> <li>1 Idle line was detected.</li> </ul> |
| 3<br>OR | <p>Receiver Overrun Flag</p> <p>OR is set when a new serial character is ready to be transferred to the receive data register (buffer), but the previously received character has not been read from UART_D yet. In this case, the new character, and all associated error information, is lost because there is no room to move it into UART_D. To clear OR, read UART_S1 with OR set and then read the UART data register (UART_D).</p> <ul style="list-style-type: none"> <li>0 No overrun.</li> <li>1 Receive overrun (new UART data lost).</li> </ul>   |
| 2<br>NF | <p>Noise Flag</p> <p>The advanced sampling technique used in the receiver takes seven samples during the start bit and three samples in each data bit and the stop bits. If any of these samples disagrees with the rest of the samples within any bit time in the frame, the flag NF is set at the same time as RDRF is set for the character. To clear NF, read UART_S1 and then read the UART data register (UART_D).</p> <ul style="list-style-type: none"> <li>0 No noise detected.</li> <li>1 Noise detected in the received character in UART_D.</li> </ul>   |
| 1<br>FE | <p>Framing Error Flag</p> <p>FE is set at the same time as RDRF when the receiver detects a logic 0 where the stop bits was expected. This suggests the receiver was not properly aligned to a character frame. To clear FE, read UART_S1 with FE set and then read the UART data register (UART_D).</p> <ul style="list-style-type: none"> <li>0 No framing error detected. This does not guarantee the framing is correct.</li> <li>1 Framing error.</li> </ul>  |
| 0<br>PF | <p>Parity Error Flag</p> <p>PF is set at the same time as RDRF when parity is enabled (PE = 1) and the parity bit in the received character does not agree with the expected parity value. To clear PF, read UART_S1 and then read the UART data register (UART_D).</p> <ul style="list-style-type: none"> <li>0 No parity error.</li> <li>1 Parity error.</li> </ul>  |

**40.2.6 UART Status Register 2 (UARTx\_S2)**

This register contains one read-only status flag.

When using an internal oscillator in a LIN system, it is necessary to raise the break detection threshold one bit time. Under the worst case timing conditions allowed in LIN, it is possible that a 0x00 data character can appear to be 10.26 bit times long at a slave running 14% faster than the master. This would trigger normal break detection circuitry designed to detect a 10-bit break symbol. When the LBKDE bit is set, framing errors are inhibited and the break detection threshold changes from 10 bits to 11 bits, preventing false detection of a 0x00 data character as a LIN break symbol.

Address: Base address + h offset

| Bit   | 7      | 6       | 5 | 4     | 3     | 2     | 1     | 0   |
|-------|--------|---------|---|-------|-------|-------|-------|-----|
| Read  | LBKDIF | RXEDGIF | 0 | RXINV | RWUID | BRK13 | LBKDE | RAF |
| Write | 0      | 0       | 0 | 0     | 0     | 0     | 0     | 0   |
| Reset | 0      | 0       | 0 | 0     | 0     | 0     | 0     | 0   |

### UARTx\_S2 field descriptions

| Field         | Description  |
|---------------|--|
| 7<br>LBKDIF   | LIN Break Detect Interrupt Flag<br><br>LBKDIF is set when the LIN break detect circuitry is enabled and a LIN break character is detected.<br>LBKDIF is cleared by writing a 1 to it.<br><br>0 No LIN break character has been detected.<br>1 LIN break character has been detected.   |
| 6<br>RXEDGIF  | RxD Pin Active Edge Interrupt Flag<br><br>RXEDGIF is set when an active edge, falling if RXINV = 0, rising if RXINV=1, on the RxD pin occurs.<br>RXEDGIF is cleared by writing a 1 to it.<br><br>0 No active edge on the receive pin has occurred.<br>1 An active edge on the receive pin has occurred.  |
| 5<br>Reserved | This field is reserved.<br>This read-only field is reserved and always has the value 0.  |
| 4<br>RXINV    | Receive Data Inversion<br><br>Setting this bit reverses the polarity of the received data input.<br><br><b>NOTE:</b> Setting RXINV inverts the RxD input for all cases: data bits, start and stop bits, break, and idle.<br><br>0 Receive data not inverted.<br>1 Receive data inverted.   |
| 3<br>RWUID    | Receive Wake Up Idle Detect<br><br>RWUID controls whether the idle character that wakes up the receiver sets the IDLE bit.<br><br>0 During receive standby state (RWU = 1), the IDLE bit does not get set upon detection of an idle character.<br>1 During receive standby state (RWU = 1), the IDLE bit gets set upon detection of an idle character. |
| 2<br>BRK13    | Break Character Generation Length  |

Table continues on the next page...

**UARTx\_S2 field descriptions (continued)**

| Field      | Description   |
|------------|---|
|            | <p>BRK13 selects a longer transmitted break character length. Detection of a framing error is not affected by the state of this bit.</p> <p>0 Break character is transmitted with length of 10 bit times (if M = 0, SBNS = 0) or 11 (if M = 1, SBNS = 0 or M = 0, SBNS = 1) or 12 (if M = 1, SBNS = 1).</p> <p>1 Break character is transmitted with length of 13 bit times (if M = 0, SBNS = 0) or 14 (if M = 1, SBNS = 0 or M = 0, SBNS = 1) or 15 (if M = 1, SBNS = 1).</p>  |
| 1<br>LBKDE | <p>LIN Break Detection Enable</p> <p>LBKDE selects a longer break character detection length. While LBKDE is set, framing error (FE) and receive data register full (RDRF) flags are prevented from setting.</p> <p>0 Break character is detected at length 10 bit times (if M = 0, SBNS = 0) or 11 (if M = 1, SBNS = 0 or M = 0, SBNS = 1) or 12 (if M = 1, SBNS = 1).</p> <p>1 Break character is detected at length of 11 bit times (if M = 0, SBNS = 0) or 12 (if M = 1, SBNS = 0 or M = 0, SBNS = 1) or 13 (if M = 1, SBNS = 1).</p> |
| 0<br>RAF   | <p>Receiver Active Flag</p> <p>RAF is set when the UART receiver detects the beginning of a valid start bit, and RAF is cleared automatically when the receiver detects an idle line. This status flag can be used to check whether an UART character is being received before instructing the MCU to go to stop mode.</p> <p>0 UART receiver idle waiting for a start bit.</p> <p>1 UART receiver active (RxD input not idle).</p>   |

**40.2.7 UART Control Register 3 (UARTx\_C3)**

Address: Base address + h offset

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

**UARTx\_C3 field descriptions**

| Field   | Description   |
|---------|---|
| 7<br>R8 | <p>Ninth Data Bit for Receiver</p> <p>When the UART is configured for 9-bit data (M = 1), R8 can be thought of as a ninth receive data bit to the left of the msb of the buffered data in the UART_D register. When reading 9-bit data, read R8 before reading UART_D because reading UART_D completes automatic flag clearing sequences that could allow R8 and UART_D to be overwritten with new data.</p>  |
| 6<br>T8 | <p>Ninth Data Bit for Transmitter</p> <p>When the UART is configured for 9-bit data (M = 1), T8 may be thought of as a ninth transmit data bit to the left of the msb of the data in the UART_D register. When writing 9-bit data, the entire 9-bit value is transferred to the UART shift register after UART_D is written so T8 should be written, if it needs to change from its previous value, before UART_D is written. If T8 does not need to change in the new value,</p> |

*Table continues on the next page...*

**UARTx\_C3 field descriptions (continued)**

| Field   | Description  |
|---------|--|
|         | such as when it is used to generate mark or space parity, it need not be written each time UART_D is written.  |
| 5 TXDIR | <p>TxD Pin Direction in Single-Wire Mode</p> <p>When the UART is configured for single-wire half-duplex operation (LOOPS = RSRC = 1), this bit determines the direction of data at the TxD pin.</p> <p>0 TxD pin is an input in single-wire mode.<br/>1 TxD pin is an output in single-wire mode.</p>    |
| 4 TXINV | <p>Transmit Data Inversion</p> <p>Setting this bit reverses the polarity of the transmitted data output.</p> <p><b>NOTE:</b> Setting TXINV inverts the TxD output for all cases: data bits, start and stop bits, break, and idle.</p> <p>0 Transmit data not inverted.<br/>1 Transmit data inverted.</p> |
| 3 ORIE  | <p>Overrun Interrupt Enable</p> <p>This bit enables the overrun flag (OR) to generate hardware interrupt requests.</p> <p>0 OR interrupts disabled; use polling.<br/>1 Hardware interrupt requested when OR is set.</p>  |
| 2 NEIE  | <p>Noise Error Interrupt Enable</p> <p>This bit enables the noise flag (NF) to generate hardware interrupt requests.</p> <p>0 NF interrupts disabled; use polling.<br/>1 Hardware interrupt requested when NF is set.</p>  |
| 1 FEIE  | <p>Framing Error Interrupt Enable</p> <p>This bit enables the framing error flag (FE) to generate hardware interrupt requests.</p> <p>0 FE interrupts disabled; use polling.<br/>1 Hardware interrupt requested when FE is set.</p>  |
| 0 PEIE  | <p>Parity Error Interrupt Enable</p> <p>This bit enables the parity error flag (PF) to generate hardware interrupt requests.</p> <p>0 PF interrupts disabled; use polling.<br/>1 Hardware interrupt requested when PF is set.</p>  |

## 40.2.8 UART Data Register (UARTx\_D)

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

Address: Base address + h offset

| Bit   | 7    | 6    | 5    | 4    | 3    | 2    | 1    | 0    |
|-------|------|------|------|------|------|------|------|------|
| Read  | R7T7 | R6T6 | R5T5 | R4T4 | R3T3 | R2T2 | R1T1 | R0T0 |
| Reset | 0    | 0    | 0    | 0    | 0    | 0    | 0    | 0    |

**UARTx\_D field descriptions**

| Field     | Description   |
|-----------|---|
| 7<br>R7T7 | Read receive data buffer 7 or write transmit data buffer 7. |
| 6<br>R6T6 | Read receive data buffer 6 or write transmit data buffer 6. |
| 5<br>R5T5 | Read receive data buffer 5 or write transmit data buffer 5. |
| 4<br>R4T4 | Read receive data buffer 4 or write transmit data buffer 4. |
| 3<br>R3T3 | Read receive data buffer 3 or write transmit data buffer 3. |
| 2<br>R2T2 | Read receive data buffer 2 or write transmit data buffer 2. |
| 1<br>R1T1 | Read receive data buffer 1 or write transmit data buffer 1. |
| 0<br>R0T0 | Read receive data buffer 0 or write transmit data buffer 0. |

**UARTx\_C4 field descriptions**

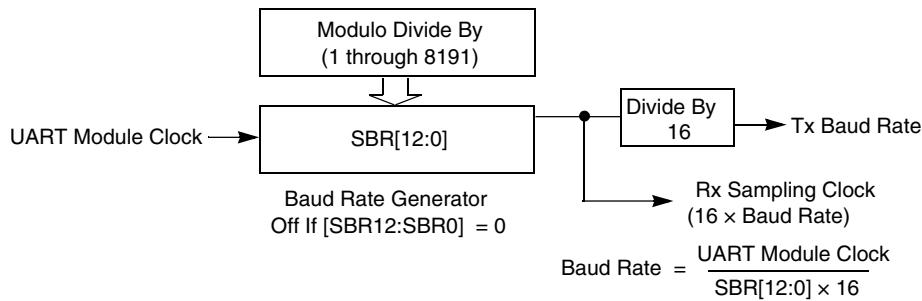
| Field           | Description   |
|-----------------|---|
| 7<br>TDMAS      | <p>Transmitter DMA Select</p> <p>TDMAS configures the transmit data register empty flag, TDRE, to generate interrupt or DMA requests if TIE is set.</p> <p><b>NOTE:</b> If UART_C2[TIE] is cleared, TDRE DMA and TDRE interrupt request signals are not asserted when the TDRE flag is set, regardless of the state of TDMAS.</p> <p>If UART_C2[TIE] and TDMAS are both set, then UART_C2[TCIE] must be cleared, and UART_D must not be written outside of servicing a DMA request.</p> <ul style="list-style-type: none"> <li>0 If TIE is set and the TDRE flag is set, the TDRE interrupt request signal is asserted to request interrupt service.</li> <li>1 If TIE is set and the TDRE flag is set, the TDRE DMA request signal is asserted to request a DMA transfer.</li> </ul> |
| 6<br>Reserved   | <p>This field is reserved.</p> <p>This read-only field is reserved and always has the value 0.</p>  |
| 5<br>RDMAS      | <p>Receiver Full DMA Select</p> <p>RDMAS configures the receiver data register full flag, RDRF, to generate interrupt or DMA requests if RIE is set.</p> <p><b>NOTE:</b> If RIE is cleared, the RDRF DMA and RDRF interrupt request signals are not asserted when the RDRF flag is set, regardless of the state of RDMAS.</p> <ul style="list-style-type: none"> <li>0 If RIE is set and the RDRF flag is set, the RDRF interrupt request signal is asserted to request interrupt service.</li> <li>1 If RIE is set and the RDRF flag is set, the RDRF DMA request signal is asserted to request a DMA transfer.</li> </ul>   |
| 4<br>Reserved   | <p>This field is reserved.</p> <p>This read-only field is reserved and always has the value 0.</p>  |
| 3<br>Reserved   | <p>This field is reserved.</p> <p>This read-only field is reserved and always has the value 0.</p>  |
| 2–0<br>Reserved | <p>This field is reserved.</p> <p>This read-only field is reserved and always has the value 0.</p>  |

## 40.3 Functional description

The UART allows full-duplex, asynchronous, NRZ serial communication among the MCU and remote devices, including other MCUs. The UART comprises a baud rate generator, transmitter, and receiver block. The transmitter and receiver operate independently, although they use the same baud rate generator. During normal operation, the MCU monitors the status of the UART, writes the data to be transmitted, and processes received data. The following describes each of the blocks of the UART.

### 40.3.1 Baud rate generation

As shown in the following figure, the clock source for the UART baud rate generator is the bus-rate clock.



**Figure 40-30. UART baud rate generation**

UART communications require the transmitter and receiver, which typically derive baud rates from independent clock sources, to use the same baud rate. Allowed tolerance on this baud frequency depends on the details of how the receiver synchronizes to the leading edge of the start bit and how bit sampling is performed.

The MCU resynchronizes to bit boundaries on every high-to-low transition. In the worst case, there are no such transitions in the full 10- or 11-bit or 12-bittime character frame so any mismatch in baud rate is accumulated for the whole character time. For a Freescale UART system whose bus frequency is driven by a crystal, the allowed baud rate mismatch is about  $\pm 4.5$  percent for 8-bit data format and about  $\pm 4$  percent for 9-bit data format. Although baud rate modulo divider settings do not always produce baud rates that exactly match standard rates, it is normally possible to get within a few percent, which is acceptable for reliable communications.

### 40.3.2 Transmitter functional description

This section describes the overall block diagram for the UART transmitter, as well as specialized functions for sending break and idle characters.

The transmitter output (Tx<sub>D</sub>) idle state defaults to logic high, UART\_C3[TXINV] is cleared following reset. The transmitter output is inverted by setting UART\_C3[TXINV]. The transmitter is enabled by setting the TE bit in UARTxC2. This queues a preamble character that is one full character frame of the idle state. The transmitter then remains idle until data is available in the transmit data buffer. Programs store data into the transmit data buffer by writing to the UART data register (UART\_D).

The central element of the UART transmitter is the transmit shift register that is 10 or 11 or 12 bits long depending on the setting in the UART\_C1[M] control bit and UART\_BDH[SBNS] bit. For the remainder of this section, assume UART\_C1[M] is

cleared, UART\_BDH[SBNS] is also cleared, selecting the normal 8-bit data mode. In 8-bit data mode, the shift register holds a start bit, eight data bits, and a stop bit. When the transmit shift register is available for a new UART character, the value waiting in the transmit data register is transferred to the shift register, synchronized with the baud rate clock, and the transmit data register empty (UART\_S1[TDRE]) status flag is set to indicate another character may be written to the transmit data buffer at UART\_D.

If no new character is waiting in the transmit data buffer after a stop bit is shifted out the TxD pin, the transmitter sets the transmit complete flag and enters an idle mode, with TxD high, waiting for more characters to transmit.

Writing 0 to UART\_C2[TE] does not immediately release the pin to be a general-purpose I/O pin. Any transmit activity in progress must first be completed. This includes data characters in progress, queued idle characters, and queued break characters.

#### 40.3.2.1 Send break and queued idle

The UART\_C2[SBK] bit sends break characters originally used to gain the attention of old teletype receivers. Break characters are a full character time of logic 0, 10 bit times including the start and stop bits. A longer break of 13 bit times can be enabled by setting UART\_S2[BRK13]. Normally, a program would wait for UART\_S1[TDRE] to become set to indicate the last character of a message has moved to the transmit shifter, write 1, and then write 0 to the UART\_C2[SBK] bit. This action queues a break character to be sent as soon as the shifter is available. If UART\_C2[SBK] remains 1 when the queued break moves into the shifter, synchronized to the baud rate clock, an additional break character is queued. If the receiving device is another Freescale Semiconductor UART, the break characters are received as 0s in all eight data bits and a framing error (UART\_S1[FE] = 1) occurs.

When idle-line wakeup is used, a full character time of idle (logic 1) is needed between messages to wake up any sleeping receivers. Normally, a program would wait for UART\_S1[TDRE] to become set to indicate the last character of a message has moved to the transmit shifter, then write 0 and then write 1 to the UART\_C2[TE] bit. This action queues an idle character to be sent as soon as the shifter is available. As long as the character in the shifter does not finish while UART\_C2[TE] is cleared, the UART transmitter never actually releases control of the TxD pin. If there is a possibility of the shifter finishing while UART\_C2[TE] is cleared, set the general-purpose I/O controls so the pin shared with TxD is an output driving a logic 1. This ensures that the TxD line looks like a normal idle line even if the UART loses control of the port pin between writing 0 and then 1 to UART\_C2[TE].

The length of the break character is affected by the UART\_S2[BRK13] and UART\_C1[M] bits as shown below.

**Table 40-31. Break character length**

| BRK13 | M | SBNS | Break character length |
|-------|---|------|------------------------|
| 0     | 0 | 0    | 10 bit times           |
| 0     | 0 | 1    | 11 bit times           |
| 0     | 1 | 0    | 11 bit times           |
| 0     | 1 | 1    | 12 bit times           |
| 1     | 0 | 0    | 13 bit times           |
| 1     | 0 | 1    | 14 bit times           |
| 1     | 1 | 0    | 14 bit times           |
| 1     | 1 | 1    | 15 bit times           |

### 40.3.3 Receiver functional description

In this section, the receiver block diagram is a guide for the overall receiver functional description. Next, the data sampling technique used to reconstruct receiver data is described in more detail. Finally, two variations of the receiver wakeup function are explained.

The receiver input is inverted by setting UART\_S2[RXINV]. The receiver is enabled by setting the UART\_C2[RE] bit. Character frames consist of a start bit of logic 0, eight (or nine) data bits (lsb first), and one (or two) stop bits of logic 1. For information about 9-bit data mode, refer to [8- and 9-bit data modes](#). For the remainder of this discussion, assume the UART is configured for normal 8-bit data mode.

After receiving the stop bit into the receive shifter, and provided the receive data register is not already full, the data character is transferred to the receive data register and the receive data register full (UART\_S1[RDRF]) status flag is set. If UART\_S1[RDRF] was already set indicating the receive data register (buffer) was already full, the overrun (OR) status flag is set and the new data is lost. Because the UART receiver is double-buffered, the program has one full character time after UART\_S1[RDRF] is set before the data in the receive data buffer must be read to avoid a receiver overrun.

When a program detects that the receive data register is full (UART\_S1[RDRF] = 1), it gets the data from the receive data register by reading UART\_D. The UART\_S1[RDRF] flag is cleared automatically by a two-step sequence normally satisfied in the course of the user's program that manages receive data. Refer to [Interrupts and status flags](#) for more details about flag clearing.

### 40.3.3.1 Data sampling technique

The UART receiver uses a 16× baud rate clock for sampling. The receiver starts by taking logic level samples at 16 times the baud rate to search for a falling edge on the RxD serial data input pin. A falling edge is defined as a logic 0 sample after three consecutive logic 1 samples. The 16× baud rate clock divides the bit time into 16 segments labeled `UART_D[RT1]` through `UART_D[RT16]`. When a falling edge is located, three more samples are taken at `UART_D[RT3]`, `UART_D[RT5]`, and `UART_D[RT7]` to make sure this was a real start bit and not merely noise. If at least two of these three samples are 0, the receiver assumes it is synchronized to a receive character.

The receiver then samples each bit time, including the start and stop bits, at `UART_D[RT8]`, `UART_D[RT9]`, and `UART_D[RT10]` to determine the logic level for that bit. The logic level is interpreted to be that of the majority of the samples taken during the bit time. In the case of the start bit, the bit is assumed to be 0 if at least two of the samples at `UART_D[RT3]`, `UART_D[RT5]`, and `UART_D[RT7]` are 0 even if one or all of the samples taken at `UART_D[RT8]`, `UART_D[RT9]`, and `UART_D[RT10]` are 1s. If any sample in any bit time, including the start and stop bits, in a character frame fails to agree with the logic level for that bit, the noise flag (`UART_S1[NF]`) is set when the received character is transferred to the receive data buffer.

The falling edge detection logic continuously looks for falling edges. If an edge is detected, the sample clock is resynchronized to bit times. This improves the reliability of the receiver in the presence of noise or mismatched baud rates. It does not improve worst case analysis because some characters do not have any extra falling edges anywhere in the character frame.

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

In the case of a framing error, the receiver is inhibited from receiving any new characters until the framing error flag is cleared. The receive shift register continues to function, but a complete character cannot transfer to the receive data buffer if `UART_S1[FE]` remains set.

### 40.3.3.2 Receiver wakeup operation

Receiver wakeup is a hardware mechanism that allows an UART receiver to ignore the characters in a message intended for a different UART receiver. In such a system, all receivers evaluate the first character(s) of each message, and as soon as they determine the message is intended for a different receiver, they write logic 1 to the receiver wake up control bit(UART\_C2[RWU]). When RWU bit is set, the status flags associated with the receiver, with the exception of the idle bit, IDLE, when UART\_S2[RWUID] bit is set, are inhibited from setting, thus eliminating the software overhead for handling the unimportant message characters. At the end of a message, or at the beginning of the next message, all receivers automatically force UART\_C2[RWU] to 0 so all receivers wake up in time to look at the first character(s) of the next message.

#### 40.3.3.2.1 Idle-line wakeup

When wake is cleared, the receiver is configured for idle-line wakeup. In this mode, UART\_C2[RWU] is cleared automatically when the receiver detects a full character time of the idle-line level. The UART\_C1[M] control bit selects 8-bit or 9-bit data mode and the UART\_BDH[SBNS] bit selects 1-bit or 2-bit stop bit number that determines how many bit times of idle are needed to constitute a full character time, 10 or 11 or 12 bit times because of the start and stop bits.

When UARTI\_C2[RWU] is one and UART\_S2[RWUID] is zero, the idle condition that wakes up the receiver does not set the UART\_S1[IDLE] flag. The receiver wakes up and waits for the first data character of the next message that sets the UART\_S1[RDRF] flag and generates an interrupt if enabled. When UART\_S2[RWUID] is one, any idle condition sets the UART\_S1[IDLE] flag and generates an interrupt if enabled, regardless of whether UART\_C2[RWU] is zero or one.

The idle-line type (UART\_C1[ILT]) control bit selects one of two ways to detect an idle line. When UART\_C1[ILT] is cleared, the idle bit counter starts after the start bit so the stop bit and any logic 1s at the end of a character count toward the full character time of idle. When UART\_C1[ILT] is set, the idle bit counter does not start until after a stop bit time, so the idle detection is not affected by the data in the last character of the previous message.

#### 40.3.3.2.2 Address-mark wakeup

When wake is set, the receiver is configured for address-mark wakeup. In this mode, UART\_C2[RWU] is cleared automatically when the receiver detects a, or two, if UART\_BDH[SBNS] = 1, logic 1 in the most significant bits of a received character, eighth bit when UART\_C1[M] is cleared and ninth bit when UART\_C1[M] is set.

Address-mark wakeup allows messages to contain idle characters, but requires the msb be reserved for use in address frames. The one, or two, if `UART_BDH[SBNS] = 1`, logic 1s msb of an address frame clears the `UART_C2[RWU]` bit before the stop bits are received and sets the `UART_S1[RDRF]` flag. In this case, the character with the msb set is received even though the receiver was sleeping during most of this character time.

#### 40.3.4 Interrupts and status flags

The UART system has three separate interrupt vectors to reduce the amount of software needed to isolate the cause of the interrupt. One interrupt vector is associated with the transmitter for `UART_S1[TDRE]` and `UART_S1[TC]` events. Another interrupt vector is associated with the receiver for `RDRF`, `IDLE`, `RXEDGIF`, and `LBKDIF` events. A third vector is used for `OR`, `NF`, `FE`, and `PF` error conditions. Each of these ten interrupt sources can be separately masked by local interrupt enable masks. The flags can be polled by software when the local masks are cleared to disable generation of hardware interrupt requests.

The UART transmitter has two status flags that can optionally generate hardware interrupt requests. Transmit data register empty (`UART_S1[TDRE]`) indicates when there is room in the transmit data buffer to write another transmit character to `UART_D`. If the transmit interrupt enable (`UART_C2[TIE]`) bit is set, a hardware interrupt is requested when `UART_S1[TDRE]` is set. Transmit complete (`UART_S1[TC]`) indicates that the transmitter is finished transmitting all data, preamble, and break characters and is idle with `TxD` at the inactive level. This flag is often used in systems with modems to determine when it is safe to turn off the modem. If the transmit complete interrupt enable (`UART_C2[TCIE]`) bit is set, a hardware interrupt is requested when `UART_S1[TC]` is set. Instead of hardware interrupts, software polling may be used to monitor the `UART_S1[TDRE]` and `UART_S1[TC]` status flags if the corresponding `UART_C2[TIE]` or `UART_C2[TCIE]` local interrupt masks are cleared.

When a program detects that the receive data register is full (`UART_S1[RDRF] = 1`), it gets the data from the receive data register by reading `UART_D`. The `UART_S1[RDRF]` flag is cleared by reading `UARTxS1` while `UART_S1[RDRF]` is set and then reading `UART_D`.

When polling is used, this sequence is naturally satisfied in the normal course of the user program. If hardware interrupts are used, `UARTxS1` must be read in the interrupt service routine (ISR). Normally, this is done in the ISR anyway to check for receive errors, so the sequence is automatically satisfied.

The IDLE status flag includes logic that prevents it from getting set repeatedly when the RxD line remains idle for an extended period of time. IDLE is cleared by reading UARTxS1 while UART\_S1[IDLE] is set and then reading UART\_D. After UART\_S1[IDLE] has been cleared, it cannot become set again until the receiver has received at least one new character and has set UART\_S1[RDRF].

If the associated error was detected in the received character that caused UART\_S1[RDRF] to be set, the error flags - noise flag (UART\_S1[NF]), framing error (UART\_S1[FE]), and parity error flag (UART\_S1[PF]) - are set at the same time as UART\_S1[RDRF]. These flags are not set in overrun cases.

If UART\_S1[RDRF] was already set when a new character is ready to be transferred from the receive shifter to the receive data buffer, the overrun (UART\_S1[OR]) flag is set instead of the data along with any associated NF, FE, or PF condition is lost.

At any time, an active edge on the RxD serial data input pin causes the UART\_S2[RXEDGIF] flag to set. The UART\_S2[RXEDGIF] flag is cleared by writing a 1 to it. This function depends on the receiver being enabled (UART\_C2[RE] = 1).

### 40.3.5 DMA Operation

In the transmitter, flags TDRE and TC can be configured to assert a DMA transfer request. In the receiver, flags RDRF, IDLE and LBKDIF can be configured to assert a DMA transfer request. The following table shows the configuration bit settings required to configure each flag for DMA operation.

**Table 40-32. DMA configuration**

| Flag   | Request enable bit | DMA select bit |
|--------|--------------------|----------------|
| TDRE   | TIE = 1            | TDMAS = 1      |
| TC     | TCIE = 1           | TCDMAS = 1     |
| RDRF   | RIE = 1            | RDMAS = 1      |
| IDLE   | ILIE = 1           | ILDMAS = 1     |
| LBKDIF | LBKDIE = 1         | LBKDDMAS = 1   |

When a flag is configured for a DMA request, its associated DMA request is asserted when the flag is set. When the RDRF or IDLE flag is configured as a DMA request, the clearing mechanism of reading UART\_S1 followed by reading UART\_D does not clear the associated flag. The DMA request remains asserted until an indication is received that the DMA transactions are done. When this indication is received, the flag bit and the associated DMA request are cleared. If the DMA operation failed to remove the situation that caused the DMA request another request will be issued.

## 40.3.6 Additional UART functions

The following sections describe additional UART functions.

### 40.3.6.1 8- and 9-bit data modes

The UART system, transmitter and receiver, can be configured to operate in 9-bit data mode by setting the `UART_C1[M]`. In 9-bit mode, there is a ninth data bit to the left of the msb of the UART data register. For the transmit data buffer, this bit is stored in `T8` in `UART_C3`. For the receiver, the ninth bit is held in `UART_C3[R8]`.

For coherent writes to the transmit data buffer, write to the `UART_C3[T8]` bit before writing to `UART_D`.

If the bit value to be transmitted as the ninth bit of a new character is the same as for the previous character, it is not necessary to write to `UART_C3[T8]` again. When data is transferred from the transmit data buffer to the transmit shifter, the value in `UART_C3[T8]` is copied at the same time data is transferred from `UART_D` to the shifter.

The 9-bit data mode is typically used with parity to allow eight bits of data plus the parity in the ninth bit, or it is used with address-mark wakeup so the ninth data bit can serve as the wakeup bit. In custom protocols, the ninth bit can also serve as a software-controlled marker.

### 40.3.6.2 Stop mode operation

During all stop modes, clocks to the UART module are halted.

No UART module registers are affected in stop3 mode.

Because the clocks are halted, the UART module resumes operation upon exit from stop, only in stop and VLPS mode. Software must ensure stop mode is not entered while there is a character being transmitted out of or received into the UART module.

### 40.3.6.3 Loop mode

When `UART_C1[LOOPS]` is set, the `UART_C1[RSRC]` bit in the same register chooses between loop mode (`UART_C1[RSRC] = 0`) or single-wire mode (`UART_C1[RSRC] = 1`). Loop mode is sometimes used to check software, independent of connections in the

external system, to help isolate system problems. In this mode, the transmitter output is internally connected to the receiver input and the RxD pin is not used by the UART, so it reverts to a general-purpose port I/O pin.

#### **40.3.6.4 Single-wire operation**

When `UART_C1[LOOPS]` is set, the `RSRC` bit in the same register chooses between loop mode (`UART_C1[RSRC] = 0`) or single-wire mode (`UART_C1[RSRC] = 1`).

Single-wire mode implements a half-duplex serial connection. The receiver is internally connected to the transmitter output and to the TxD pin. The RxD pin is not used and reverts to a general-purpose port I/O pin.

In single-wire mode, the `UART_C3[TXDIR]` bit controls the direction of serial data on the TxD pin. When `UART_C3[TXDIR]` is cleared, the TxD pin is an input to the UART receiver and the transmitter is temporarily disconnected from the TxD pin so an external device can send serial data to the receiver. When `UART_C3[TXDIR]` is set, the TxD pin is an output driven by the transmitter, the internal loop back connection is disabled, and as a result the receiver can not receive characters that are sent out by the transmitter.

# Chapter 41

## General-Purpose Input/Output (GPIO)

### 41.1 Introduction

#### NOTE

For the chip-specific implementation details of this module's instances see the chip configuration information.

The general-purpose input and output (GPIO) module communicates to the processor core via a zero wait state interface for maximum pin performance. The GPIO registers support 8-bit, 16-bit or 32-bit accesses.

The GPIO data direction and output data registers control the direction and output data of each pin when the pin is configured for the GPIO function. The GPIO input data register displays the logic value on each pin when the pin is configured for any digital function, provided the corresponding Port Control and Interrupt module for that pin is enabled.

Efficient bit manipulation of the general-purpose outputs is supported through the addition of set, clear, and toggle write-only registers for each port output data register.

#### 41.1.1 Features

- Features of the GPIO module include:
  - Pin input data register visible in all digital pin-multiplexing modes
  - Pin output data register with corresponding set/clear/toggle registers
  - Pin data direction register
  - Zero wait state access to GPIO registers through IOPORT

#### NOTE

GPIO module is clocked by system clock.

## 41.1.2 Modes of operation

The following table depicts different modes of operation and the behavior of the GPIO module in these modes.

**Table 41-1. Modes of operation**

| Modes of operation | Description                        |
|--------------------|------------------------------------|
| Run                | The GPIO module operates normally. |
| Wait               | The GPIO module operates normally. |
| Stop               | The GPIO module is disabled.       |
| Debug              | The GPIO module operates normally. |

## 41.1.3 GPIO signal descriptions

**Table 41-2. GPIO signal descriptions**

| GPIO signal descriptions | Description                  | I/O |
|--------------------------|------------------------------|-----|
| PORTE31–PORTE0           | General-purpose input/output | I/O |

### NOTE

Not all pins within each port are implemented on each device.  
See the chapter on signal multiplexing for the number of GPIO ports available in the device.

### 41.1.3.1 Detailed signal description

Table 41-3. GPIO interface-detailed signal descriptions

| Signal         | I/O | Description  |  |
|----------------|-----|--|--|
| PORTE31–PORTE0 | I/O | General-purpose input/output   |  |
| PORTE31–PORTE0 |     | State meaning<br>Asserted: The pin is logic 1.<br>Deasserted: The pin is logic 0.  |  |
| PORTE31–PORTE0 |     | Timing<br>Assertion: When output, this signal occurs on the rising-edge of the system clock. For input, it may occur at any time and input may be asserted asynchronously to the system clock.<br>Deassertion: When output, this signal occurs on the rising-edge of the system clock. For input, it may occur at any time and input may be asserted asynchronously to the system clock. |  |

## 41.2 Memory map and register definition

Any read or write access to the GPIO memory space that is outside the valid memory map results in a bus error.

### NOTE

For simplicity, each GPIO port's registers appear with the same width of 32 bits, corresponding to 32 pins. The actual number of pins per port (and therefore the number of usable control bits per port register) is chip-specific. Refer to the Chip Configuration chapter to see the exact control bits for the non-identical port instance.

### GPIO memory map

| Absolute address (hex) | Register name                          | Width (in bits) | Access                | Reset value | Section/ page              |
|------------------------|--|-----------------|-----------------------|-------------|----------------------------|
| 400F_F000              | Port Data Output Register (GPIOA_PDOR) | 32              | R/W                   | 0000_0000h  | <a href="#">41.2.1/775</a> |
| 400F_F004              | Port Set Output Register (GPIOA_PSOR)  | 32              | W<br>(always reads 0) | 0000_0000h  | <a href="#">41.2.2/776</a> |

Table continues on the next page...

## GPIO memory map (continued)

| Absolute address (hex) | Register name                             | Width (in bits) | Access                | Reset value | Section/ page |
|------------------------|---|-----------------|-----------------------|-------------|---------------|
| 400F_F008              | Port Clear Output Register (GPIOA_PCOR)   | 32              | W<br>(always reads 0) | 0000_0000h  | 41.2.3/776    |
| 400F_F00C              | Port Toggle Output Register (GPIOA_PTOR)  | 32              | W<br>(always reads 0) | 0000_0000h  | 41.2.4/777    |
| 400F_F010              | Port Data Input Register (GPIOA_PDIR)     | 32              | R                     | 0000_0000h  | 41.2.5/777    |
| 400F_F014              | Port Data Direction Register (GPIOA_PDDR) | 32              | R/W                   | 0000_0000h  | 41.2.6/778    |
| 400F_F040              | Port Data Output Register (GPIOB_PDOR)    | 32              | R/W                   | 0000_0000h  | 41.2.1/775    |
| 400F_F044              | Port Set Output Register (GPIOB_PSOR)     | 32              | W<br>(always reads 0) | 0000_0000h  | 41.2.2/776    |
| 400F_F048              | Port Clear Output Register (GPIOB_PCOR)   | 32              | W<br>(always reads 0) | 0000_0000h  | 41.2.3/776    |
| 400F_F04C              | Port Toggle Output Register (GPIOB_PTOR)  | 32              | W<br>(always reads 0) | 0000_0000h  | 41.2.4/777    |
| 400F_F050              | Port Data Input Register (GPIOB_PDIR)     | 32              | R                     | 0000_0000h  | 41.2.5/777    |
| 400F_F054              | Port Data Direction Register (GPIOB_PDDR) | 32              | R/W                   | 0000_0000h  | 41.2.6/778    |
| 400F_F080              | Port Data Output Register (GPIOC_PDOR)    | 32              | R/W                   | 0000_0000h  | 41.2.1/775    |
| 400F_F084              | Port Set Output Register (GPIOC_PSOR)     | 32              | W<br>(always reads 0) | 0000_0000h  | 41.2.2/776    |
| 400F_F088              | Port Clear Output Register (GPIOC_PCOR)   | 32              | W<br>(always reads 0) | 0000_0000h  | 41.2.3/776    |
| 400F_F08C              | Port Toggle Output Register (GPIOC_PTOR)  | 32              | W<br>(always reads 0) | 0000_0000h  | 41.2.4/777    |
| 400F_F090              | Port Data Input Register (GPIOC_PDIR)     | 32              | R                     | 0000_0000h  | 41.2.5/777    |
| 400F_F094              | Port Data Direction Register (GPIOC_PDDR) | 32              | R/W                   | 0000_0000h  | 41.2.6/778    |
| 400F_F0C0              | Port Data Output Register (GPIOD_PDOR)    | 32              | R/W                   | 0000_0000h  | 41.2.1/775    |
| 400F_F0C4              | Port Set Output Register (GPIOD_PSOR)     | 32              | W<br>(always reads 0) | 0000_0000h  | 41.2.2/776    |
| 400F_F0C8              | Port Clear Output Register (GPIOD_PCOR)   | 32              | W<br>(always reads 0) | 0000_0000h  | 41.2.3/776    |
| 400F_F0CC              | Port Toggle Output Register (GPIOD_PTOR)  | 32              | W<br>(always reads 0) | 0000_0000h  | 41.2.4/777    |
| 400F_F0D0              | Port Data Input Register (GPIOD_PDIR)     | 32              | R                     | 0000_0000h  | 41.2.5/777    |
| 400F_F0D4              | Port Data Direction Register (GPIOD_PDDR) | 32              | R/W                   | 0000_0000h  | 41.2.6/778    |

### GPIO memory map (continued)

| Absolute address (hex) | Register name                             | Width (in bits) | Access                | Reset value | Section/ page              |
|------------------------|---|-----------------|-----------------------|-------------|----------------------------|
| 400F_F100              | Port Data Output Register (GPIOE_PDOR)    | 32              | R/W                   | 0000_0000h  | <a href="#">41.2.1/775</a> |
| 400F_F104              | Port Set Output Register (GPIOE_PSOR)     | 32              | W<br>(always reads 0) | 0000_0000h  | <a href="#">41.2.2/776</a> |
| 400F_F108              | Port Clear Output Register (GPIOE_PCOR)   | 32              | W<br>(always reads 0) | 0000_0000h  | <a href="#">41.2.3/776</a> |
| 400F_F10C              | Port Toggle Output Register (GPIOE_PTOR)  | 32              | W<br>(always reads 0) | 0000_0000h  | <a href="#">41.2.4/777</a> |
| 400F_F110              | Port Data Input Register (GPIOE_PDIR)     | 32              | R                     | 0000_0000h  | <a href="#">41.2.5/777</a> |
| 400F_F114              | Port Data Direction Register (GPIOE_PDDR) | 32              | R/W                   | 0000_0000h  | <a href="#">41.2.6/778</a> |

## 41.2.2 Port Set Output Register (GPIOx\_PSOR)

This register configures whether to set the fields of the PDOR.

Address: Base address + 4h offset

| Bit   | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|-------|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|---|---|---|---|---|---|---|---|---|---|
| R     |    |    |    |    |    |    |    |    |    |    |    |    |    |    |    |    | 0  |    |    |    |    |    |   |   |   |   |   |   |   |   |   |   |
| W     |    |    |    |    |    |    |    |    |    |    |    |    |    |    |    |    |    |    |    |    |    |    |   |   |   |   |   |   |   |   |   |   |
| Reset | 0  | 0  | 0  | 0  | 0  | 0  | 0  | 0  | 0  | 0  | 0  | 0  | 0  | 0  | 0  | 0  | 0  | 0  | 0  | 0  | 0  | 0  | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |   |

### GPIOx\_PSOR field descriptions

| Field        | Description   |
|--------------|---|
| 31–0<br>PTSO | <p>Port Set Output</p> <p>Writing to this register will update the contents of the corresponding bit in the PDOR as follows:</p> <ul style="list-style-type: none"> <li>0 Corresponding bit in PDORn does not change.</li> <li>1 Corresponding bit in PDORn is set to logic 1.</li> </ul> |

## 41.2.3 Port Clear Output Register (GPIOx\_PCOR)

This register configures whether to clear the fields of PDOR.

Address: Base address + 8h offset

| Bit   | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|-------|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|---|---|---|---|---|---|---|---|---|---|
| R     |    |    |    |    |    |    |    |    |    |    |    |    |    |    |    |    | 0  |    |    |    |    |    |   |   |   |   |   |   |   |   |   |   |
| W     |    |    |    |    |    |    |    |    |    |    |    |    |    |    |    |    |    |    |    |    |    |    |   |   |   |   |   |   |   |   |   |   |
| Reset | 0  | 0  | 0  | 0  | 0  | 0  | 0  | 0  | 0  | 0  | 0  | 0  | 0  | 0  | 0  | 0  | 0  | 0  | 0  | 0  | 0  | 0  | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |   |

### GPIOx\_PCOR field descriptions

| Field        | Description   |
|--------------|---|
| 31–0<br>PTCO | <p>Port Clear Output</p> <p>Writing to this register will update the contents of the corresponding bit in the Port Data Output Register (PDOR) as follows:</p> <ul style="list-style-type: none"> <li>0 Corresponding bit in PDORn does not change.</li> <li>1 Corresponding bit in PDORn is cleared to logic 0.</li> </ul> |

## 41.2.4 Port Toggle Output Register (GPIOx\_PTOR)

Address: Base address + Ch offset

| Bit   | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 | 15   | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |  |
|-------|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|------|----|----|----|----|----|---|---|---|---|---|---|---|---|---|---|--|
| R     |    |    |    |    |    |    |    |    |    |    |    |    |    |    |    | 0  |      |    |    |    |    |    |   |   |   |   |   |   |   |   |   |   |  |
| W     |    |    |    |    |    |    |    |    |    |    |    |    |    |    |    |    | PTTO |    |    |    |    |    |   |   |   |   |   |   |   |   |   |   |  |
| Reset | 0  | 0  | 0  | 0  | 0  | 0  | 0  | 0  | 0  | 0  | 0  | 0  | 0  | 0  | 0  | 0  | 0    | 0  | 0  | 0  | 0  | 0  | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |   |  |

### GPIOx\_PTOR field descriptions

| Field        | Description  |
|--------------|--|
| 31–0<br>PTTO | <p>Port Toggle Output</p> <p>Writing to this register will update the contents of the corresponding bit in the PDOR as follows:</p> <ul style="list-style-type: none"> <li>0 Corresponding bit in PDORn does not change.</li> <li>1 Corresponding bit in PDORn is set to the inverse of its existing logic state.</li> </ul> |

## 41.2.5 Port Data Input Register (GPIOx\_PDIR)

### NOTE

Do not modify pin configuration registers associated with pins not available in your selected package. All un-bonded pins not available in your package will default to DISABLE state for lowest power consumption.

Address: Base address + 10h offset

| Bit   | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16  | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|-------|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|-----|----|----|----|----|----|----|---|---|---|---|---|---|---|---|---|---|
| R     |    |    |    |    |    |    |    |    |    |    |    |    |    |    |    | PDI |    |    |    |    |    |    |   |   |   |   |   |   |   |   |   |   |
| W     |    |    |    |    |    |    |    |    |    |    |    |    |    |    |    |     |    |    |    |    |    |    |   |   |   |   |   |   |   |   |   |   |
| Reset | 0  | 0  | 0  | 0  | 0  | 0  | 0  | 0  | 0  | 0  | 0  | 0  | 0  | 0  | 0  | 0   | 0  | 0  | 0  | 0  | 0  | 0  | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |   |

### GPIOx\_PDIR field descriptions

| Field       | Description   |
|-------------|---|
| 31–0<br>PDI | <p>Port Data Input</p> <p>Reads 0 at the unimplemented pins for a particular device. Pins that are not configured for a digital function read 0. If the Port Control and Interrupt module is disabled, then the corresponding bit in PDIR does not update.</p> <ul style="list-style-type: none"> <li>0 Pin logic level is logic 0, or is not configured for use by digital function.</li> <li>1 Pin logic level is logic 1.</li> </ul> |

### 41.2.6 Port Data Direction Register (GPIOx\_PDDR)

The PDDR configures the individual port pins for input or output.

Address: Base address + 14h offset

| Bit   | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 | 15  | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |  |
|-------|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|-----|----|----|----|----|----|---|---|---|---|---|---|---|---|---|---|--|
| R     |    |    |    |    |    |    |    |    |    |    |    |    |    |    |    |    | PDD |    |    |    |    |    |   |   |   |   |   |   |   |   |   |   |  |
| Reset | 0  | 0  | 0  | 0  | 0  | 0  | 0  | 0  | 0  | 0  | 0  | 0  | 0  | 0  | 0  | 0  | 0   | 0  | 0  | 0  | 0  | 0  | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |   |  |

#### GPIOx\_PDDR field descriptions

| Field       | Description  |
|-------------|--|
| 31–0<br>PDD | Port Data Direction<br><br>Configures individual port pins for input or output.<br><br>0 Pin is configured as general-purpose input, for the GPIO function.<br>1 Pin is configured as general-purpose output, for the GPIO function. |

### 41.3 FGPIO memory map and register definition

Any read or write access to the FGPIO memory space that is outside the valid memory map results in a bus error. All register accesses complete with zero wait states, except error accesses which complete with one wait state.

#### NOTE

For simplicity, each FGPIO port's registers appear with the same width of 32 bits, corresponding to 32 pins. The actual number of pins per port (and therefore the number of usable control bits per port register) is chip-specific. Refer to the Chip Configuration chapter to see the exact control bits for the non-identical port instance.

#### FGPIO memory map

| Absolute address (hex) | Register name                           | Width (in bits) | Access             | Reset value | Section/ page              |
|------------------------|---|-----------------|--------------------|-------------|----------------------------|
| F80F_F000              | Port Data Output Register (FGPIOA_PDOR) | 32              | R/W                | 0000_0000h  | <a href="#">41.3.1/780</a> |
| F80F_F004              | Port Set Output Register (FGPIOA_PSOR)  | 32              | W (always reads 0) | 0000_0000h  | <a href="#">41.3.2/781</a> |

Table continues on the next page...

**FGPIO memory map (continued)**

| Absolute address (hex) | Register name                              | Width (in bits) | Access                | Reset value | Section/page               |
|------------------------|--|-----------------|-----------------------|-------------|----------------------------|
| F80F_F008              | Port Clear Output Register (FGPIOA_PCOR)   | 32              | W<br>(always reads 0) | 0000_0000h  | <a href="#">41.3.3/781</a> |
| F80F_F00C              | Port Toggle Output Register (FGPIOA_PTOR)  | 32              | W<br>(always reads 0) | 0000_0000h  | <a href="#">41.3.4/782</a> |
| F80F_F010              | Port Data Input Register (FGPIOA_PDIR)     | 32              | R                     | 0000_0000h  | <a href="#">41.3.5/782</a> |
| F80F_F014              | Port Data Direction Register (FGPIOA_PDDR) | 32              | R/W                   | 0000_0000h  | <a href="#">41.3.6/783</a> |
| F80F_F040              | Port Data Output Register (FGPIOB_PDOR)    | 32              | R/W                   | 0000_0000h  | <a href="#">41.3.1/780</a> |
| F80F_F044              | Port Set Output Register (FGPIOB_PSOR)     | 32              | W<br>(always reads 0) | 0000_0000h  | <a href="#">41.3.2/781</a> |
| F80F_F048              | Port Clear Output Register (FGPIOB_PCOR)   | 32              | W<br>(always reads 0) | 0000_0000h  | <a href="#">41.3.3/781</a> |
| F80F_F04C              | Port Toggle Output Register (FGPIOB_PTOR)  | 32              | W<br>(always reads 0) | 0000_0000h  | <a href="#">41.3.4/782</a> |
| F80F_F050              | Port Data Input Register (FGPIOB_PDIR)     | 32              | R                     | 0000_0000h  | <a href="#">41.3.5/782</a> |
| F80F_F054              | Port Data Direction Register (FGPIOB_PDDR) | 32              | R/W                   | 0000_0000h  | <a href="#">41.3.6/783</a> |
| F80F_F080              | Port Data Output Register (FGPIOC_PDOR)    | 32              | R/W                   | 0000_0000h  | <a href="#">41.3.1/780</a> |
| F80F_F084              | Port Set Output Register (FGPIOC_PSOR)     | 32              | W<br>(always reads 0) | 0000_0000h  | <a href="#">41.3.2/781</a> |
| F80F_F088              | Port Clear Output Register (FGPIOC_PCOR)   | 32              | W<br>(always reads 0) | 0000_0000h  | <a href="#">41.3.3/781</a> |
| F80F_F08C              | Port Toggle Output Register (FGPIOC_PTOR)  | 32              | W<br>(always reads 0) | 0000_0000h  | <a href="#">41.3.4/782</a> |
| F80F_F090              | Port Data Input Register (FGPIOC_PDIR)     | 32              | R                     | 0000_0000h  | <a href="#">41.3.5/782</a> |
| F80F_F094              | Port Data Direction Register (FGPIOC_PDDR) | 32              | R/W                   | 0000_0000h  | <a href="#">41.3.6/783</a> |
| F80F_F0C0              | Port Data Output Register (FGPIOD_PDOR)    | 32              | R/W                   | 0000_0000h  | <a href="#">41.3.1/780</a> |
| F80F_F0C4              | Port Set Output Register (FGPIOD_PSOR)     | 32              | W<br>(always reads 0) | 0000_0000h  | <a href="#">41.3.2/781</a> |
| F80F_F0C8              | Port Clear Output Register (FGPIOD_PCOR)   | 32              | W<br>(always reads 0) | 0000_0000h  | <a href="#">41.3.3/781</a> |
| F80F_F0CC              | Port Toggle Output Register (FGPIOD_PTOR)  | 32              | W<br>(always reads 0) | 0000_0000h  | <a href="#">41.3.4/782</a> |
| F80F_F0D0              | Port Data Input Register (FGPIOD_PDIR)     | 32              | R                     | 0000_0000h  | <a href="#">41.3.5/782</a> |
| F80F_F0D4              | Port Data Direction Register (FGPIOD_PDDR) | 32              | R/W                   | 0000_0000h  | <a href="#">41.3.6/783</a> |

## FGPIO memory map (continued)

| Absolute address (hex) | Register name                              | Width (in bits) | Access                | Reset value | Section/ page              |
|------------------------|--|-----------------|-----------------------|-------------|----------------------------|
| F80F_F100              | Port Data Output Register (FGPIOE_PDOR)    | 32              | R/W                   | 0000_0000h  | <a href="#">41.3.1/780</a> |
| F80F_F104              | Port Set Output Register (FGPIOE_PSOR)     | 32              | W<br>(always reads 0) | 0000_0000h  | <a href="#">41.3.2/781</a> |
| F80F_F108              | Port Clear Output Register (FGPIOE_PCOR)   | 32              | W<br>(always reads 0) | 0000_0000h  | <a href="#">41.3.3/781</a> |
| F80F_F10C              | Port Toggle Output Register (FGPIOE_PTOR)  | 32              | W<br>(always reads 0) | 0000_0000h  | <a href="#">41.3.4/782</a> |
| F80F_F110              | Port Data Input Register (FGPIOE_PDIR)     | 32              | R                     | 0000_0000h  | <a href="#">41.3.5/782</a> |
| F80F_F114              | Port Data Direction Register (FGPIOE_PDDR) | 32              | R/W                   | 0000_0000h  | <a href="#">41.3.6/783</a> |

**41.3.1 Port Data Output Register (FGPIOx\_PDOR)**

This register configures the logic levels that are driven on each general-purpose output pins.

Address: Base address + 0h offset

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |  |
|-----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|---|---|---|---|---|---|---|---|---|---|--|
| R   |    |    |    |    |    |    |    |    |    |    |    |    |    |    |    |    |    |    |    |    |    |    |   |   |   |   |   |   |   |   |   |   |  |
| W   |    |    |    |    |    |    |    |    |    |    |    |    |    |    |    |    |    |    |    |    |    |    |   |   |   |   |   |   |   |   |   |   |  |

Reset 0

**FGPIOx\_PDOR field descriptions**

| Field       | Description  |
|-------------|--|
| 31–0<br>PDO | Port Data Output<br><br>Unimplemented pins for a particular device read as zero.<br><br>0 Logic level 0 is driven on pin, provided pin is configured for general-purpose output.<br>1 Logic level 1 is driven on pin, provided pin is configured for general-purpose output. |

### 41.3.2 Port Set Output Register (FGPIOx\_PSOR)

This register configures whether to set the fields of the PDOR.

Address: Base address + 4h offset

| Bit   | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 | 15 | 14   | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |  |
|-------|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|------|----|----|----|----|---|---|---|---|---|---|---|---|---|---|--|
| R     |    |    |    |    |    |    |    |    |    |    |    |    |    |    |    |    | 0  |      |    |    |    |    |   |   |   |   |   |   |   |   |   |   |  |
| W     |    |    |    |    |    |    |    |    |    |    |    |    |    |    |    |    |    | PTSO |    |    |    |    |   |   |   |   |   |   |   |   |   |   |  |
| Reset | 0  | 0  | 0  | 0  | 0  | 0  | 0  | 0  | 0  | 0  | 0  | 0  | 0  | 0  | 0  | 0  | 0  | 0    | 0  | 0  | 0  | 0  | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |   |   |  |

#### FGPIOx\_PSOR field descriptions

| Field        | Description   |   |   |   |   |
|--------------|---|---|---|---|---|
| 31–0<br>PTSO | <p>Port Set Output</p> <p>Writing to this register will update the contents of the corresponding bit in the PDOR as follows:</p> <table> <tr> <td>0</td> <td>Corresponding bit in PDORn does not change.</td> </tr> <tr> <td>1</td> <td>Corresponding bit in PDORn is set to logic 1.</td> </tr> </table> | 0 | Corresponding bit in PDORn does not change. | 1 | Corresponding bit in PDORn is set to logic 1. |
| 0            | Corresponding bit in PDORn does not change.   |   |   |   |   |
| 1            | Corresponding bit in PDORn is set to logic 1.   |   |   |   |   |

### 41.3.3 Port Clear Output Register (FGPIOx\_PCOR)

This register configures whether to clear the fields of PDOR.

Address: Base address + 8h offset

| Bit   | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 | 15 | 14   | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |  |
|-------|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|------|----|----|----|----|---|---|---|---|---|---|---|---|---|---|--|
| R     |    |    |    |    |    |    |    |    |    |    |    |    |    |    |    |    | 0  |      |    |    |    |    |   |   |   |   |   |   |   |   |   |   |  |
| W     |    |    |    |    |    |    |    |    |    |    |    |    |    |    |    |    |    | PTCO |    |    |    |    |   |   |   |   |   |   |   |   |   |   |  |
| Reset | 0  | 0  | 0  | 0  | 0  | 0  | 0  | 0  | 0  | 0  | 0  | 0  | 0  | 0  | 0  | 0  | 0  | 0    | 0  | 0  | 0  | 0  | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |   |   |  |

#### FGPIOx\_PCOR field descriptions

| Field        | Description   |   |   |   |   |
|--------------|---|---|---|---|---|
| 31–0<br>PTCO | <p>Port Clear Output</p> <p>Writing to this register will update the contents of the corresponding bit in the Port Data Output Register (PDOR) as follows:</p> <table> <tr> <td>0</td> <td>Corresponding bit in PDORn does not change.</td> </tr> <tr> <td>1</td> <td>Corresponding bit in PDORn is cleared to logic 0.</td> </tr> </table> | 0 | Corresponding bit in PDORn does not change. | 1 | Corresponding bit in PDORn is cleared to logic 0. |
| 0            | Corresponding bit in PDORn does not change.   |   |   |   |   |
| 1            | Corresponding bit in PDORn is cleared to logic 0.   |   |   |   |   |

### 41.3.4 Port Toggle Output Register (FGPIOx\_PTOR)

Address: Base address + Ch offset

| Bit   | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16   | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |  |
|-------|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|------|----|----|----|----|----|----|---|---|---|---|---|---|---|---|---|---|--|
| R     |    |    |    |    |    |    |    |    |    |    |    |    |    |    |    | 0    |    |    |    |    |    |    |   |   |   |   |   |   |   |   |   |   |  |
| W     |    |    |    |    |    |    |    |    |    |    |    |    |    |    |    | PTTO |    |    |    |    |    |    |   |   |   |   |   |   |   |   |   |   |  |
| Reset | 0  | 0  | 0  | 0  | 0  | 0  | 0  | 0  | 0  | 0  | 0  | 0  | 0  | 0  | 0  | 0    | 0  | 0  | 0  | 0  | 0  | 0  | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |   |  |

#### FGPIOx\_PTOR field descriptions

| Field        | Description  |
|--------------|--|
| 31–0<br>PTTO | <p>Port Toggle Output</p> <p>Writing to this register will update the contents of the corresponding bit in the PDOR as follows:</p> <ul style="list-style-type: none"> <li>0 Corresponding bit in PDORn does not change.</li> <li>1 Corresponding bit in PDORn is set to the inverse of its existing logic state.</li> </ul> |

### 41.3.5 Port Data Input Register (FGPIOx\_PDIR)

Address: Base address + 10h offset

| Bit   | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16  | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |  |
|-------|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|-----|----|----|----|----|----|----|---|---|---|---|---|---|---|---|---|---|--|
| R     |    |    |    |    |    |    |    |    |    |    |    |    |    |    |    | PDI |    |    |    |    |    |    |   |   |   |   |   |   |   |   |   |   |  |
| W     |    |    |    |    |    |    |    |    |    |    |    |    |    |    |    |     |    |    |    |    |    |    |   |   |   |   |   |   |   |   |   |   |  |
| Reset | 0  | 0  | 0  | 0  | 0  | 0  | 0  | 0  | 0  | 0  | 0  | 0  | 0  | 0  | 0  | 0   | 0  | 0  | 0  | 0  | 0  | 0  | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |   |  |

#### FGPIOx\_PDIR field descriptions

| Field       | Description   |
|-------------|---|
| 31–0<br>PDI | <p>Port Data Input</p> <p>Reads 0 at the unimplemented pins for a particular device. Pins that are not configured for a digital function read 0. If the Port Control and Interrupt module is disabled, then the corresponding bit in PDIR does not update.</p> <ul style="list-style-type: none"> <li>0 Pin logic level is logic 0, or is not configured for use by digital function.</li> <li>1 Pin logic level is logic 1.</li> </ul> |

### 41.3.6 Port Data Direction Register (FGPIOx\_PDDR)

The PDDR configures the individual port pins for input or output.

Address: Base address + 14h offset

| Bit   | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 | 15  | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |  |
|-------|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|-----|----|----|----|----|----|---|---|---|---|---|---|---|---|---|---|--|
| R     |    |    |    |    |    |    |    |    |    |    |    |    |    |    |    |    | PDD |    |    |    |    |    |   |   |   |   |   |   |   |   |   |   |  |
| Reset | 0  | 0  | 0  | 0  | 0  | 0  | 0  | 0  | 0  | 0  | 0  | 0  | 0  | 0  | 0  | 0  | 0   | 0  | 0  | 0  | 0  | 0  | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |   |  |

#### FGPIOx\_PDDR field descriptions

| Field       | Description  |
|-------------|--|
| 31–0<br>PDD | <p>Port Data Direction</p> <p>Configures individual port pins for input or output.</p> <p>0 Pin is configured as general-purpose input, for the GPIO function.<br/>1 Pin is configured as general-purpose output, for the GPIO function.</p> |

## 41.4 Functional description

### 41.4.1 General-purpose input

The logic state of each pin is available via the Port Data Input registers, provided the pin is configured for a digital function and the corresponding Port Control and Interrupt module is enabled.

### 41.4.2 General-purpose output

The logic state of each pin can be controlled via the port data output registers and port data direction registers, provided the pin is configured for the GPIO function. The following table depicts the conditions for a pin to be configured as input/output.

| If   | Then   |
|--|--|
| A pin is configured for the GPIO function and the corresponding port data direction register bit is clear. | The pin is configured as an input.   |
| A pin is configured for the GPIO function and the corresponding port data direction register bit is set.   | The pin is configured as an output and the logic state of the pin is equal to the corresponding port data output register. |

To facilitate efficient bit manipulation on the general-purpose outputs, pin data set, pin data clear, and pin data toggle registers exist to allow one or more outputs within one port to be set, cleared, or toggled from a single register write.

The corresponding Port Control and Interrupt module does not need to be enabled to update the state of the port data direction registers and port data output registers including the set/clear/toggle registers.

### **41.4.3 IOPORT**

The GPIO registers are also aliased to the IOPORT interface on the Cortex-M0+ from address \$F800\_0000. Accesses via the IOPORT interface occur in parallel with any instruction fetches and will therefore complete in a single cycle. If the DMA attempts to access the GPIO registers on the same cycle as an IOPORT access, then the DMA access will stall until any IOPORT accesses have completed.

During Compute Operation, the GPIO registers remain accessible via the IOPORT interface only. Since the clocks to the Port Control and Interrupt modules are disabled during Compute Operation, the Pin Data Input Registers do not update with the current state of the pins.

# **Chapter 42**

## **Touch Sensing Input (TSI)**

### **42.1 Introduction**

The touch sensing input (TSI) module provides capacitive touch sensing detection with high sensitivity and enhanced robustness. Each TSI pin implements the capacitive measurement by a current source scan, charging and discharging the electrode, once or several times. A reference oscillator ticks the scan time and stores the result in a 16-bit register when the scan completes. Meanwhile, an interrupt request is submitted to CPU for post-processing if TSI interrupt is enabled and DMA function is not selected. The TSI module can be periodically triggered to work in low power mode with ultra-low current adder and wake CPU at the end of scan or the conversion result is out of the range specified by TSI threshold. It provides a solid capacitive measurement module to the implementation of touch keyboard, rotaries and sliders.

#### **42.1.1 Features**

TSI features includes:

- Support up to 16 external electrodes
- Automatic detection of electrode capacitance across all operational power modes
- Internal reference oscillator for high-accuracy measurement
- Configurable software or hardware scan trigger
- Fully support Freescale touch sensing software (TSS) library, see [www.freescale.com/touchsensing](http://www.freescale.com/touchsensing).
- Capability to wake MCU from low power modes
- Compensate for temperature and supply voltage variations
- High sensitivity change with 16-bit resolution register
- Configurable up to 4096 scan times.
- Support DMA data transfer

## 42.1.2 Modes of operation

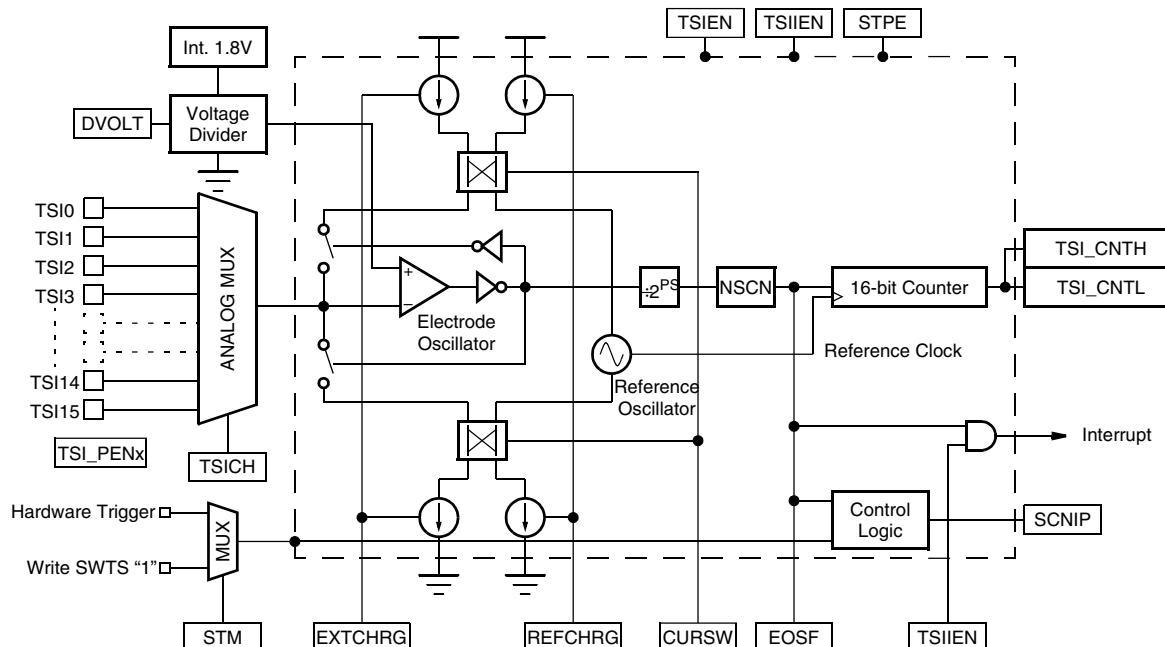
This module supports the following operation modes.

**Table 42-1. Operating modes**

| Mode                    | Description  |
|-------------------------|--|
| Stop and low power stop | TSI module is fully functional in all of the stop modes as long as TSI_GENCS[STPE] is set. The channel specified by TSI_DATA[TSICH] will be scanned upon the trigger. After scan finishes, either end-of-scan or out-of-range interrupt can be selected to bring MCU out of low power modes. |
| Wait                    | TSI module is fully functional in this mode. When a scan completes, TSI submits an interrupt request to CPU if the interrupt is enabled.   |
| Run                     | TSI module is fully functional in this mode. When a scan completes, TSI submits an interrupt request to CPU if the interrupt is enabled.   |

## 42.1.3 Block diagram

The following figure is a block diagram of the TSI module.



**Figure 42-1. TSI module block diagram**

## 42.2 External signal description

The TSI module contains up to 16 external pins for touch sensing. The following table describes each of the TSI external pins.

**Table 42-2. TSI signal description**

| Name      | Port | Direction | Function   | Reset state |
|-----------|------|-----------|--|-------------|
| TSI[15:0] | TSI  | I/O       | TSI capacitive pins.<br>Switches driver that connects directly to the electrode pins TSI[15:0] can operate as GPIO pins. | I/O         |

### 42.2.1 TSI[15:0]

When TSI functionality is enabled , the TSI analog portion uses the corresponding channel to connect external on-board touch capacitors. The PCB connection between the pin and the touch pad must be kept as short as possible to reduce distribution capacity on board.

## 42.3 Register definition

This section describes the memory map and control/status registers for the TSI module.

**TSI memory map**

| Absolute address (hex) | Register name  | Width (in bits) | Access | Reset value | Section/ page              |
|------------------------|--|-----------------|--------|-------------|----------------------------|
| 4004_5000              | TSI General Control and Status Register (TSI0_GENCS) | 32              | R/W    | 0000_0000h  | <a href="#">42.3.1/787</a> |
| 4004_5004              | TSI DATA Register (TSI0_DATA)                        | 32              | R/W    | 0000_0000h  | <a href="#">42.3.2/792</a> |
| 4004_5008              | TSI Threshold Register (TSI0_TSHD)                   | 32              | R/W    | 0000_0000h  | <a href="#">42.3.3/793</a> |

### 42.3.1 TSI General Control and Status Register (TSIx\_GENCS)

This control register provides various control and configuration information for the TSI module.

## **NOTE**

When TSI is working, the configuration bits (GENCS[TSIEN], GENCS[TSIIEN], and GENCS[STM]) must not be changed. The EOSF flag is kept until the software acknowledge it.

Address: 4004\_5000h base + 0h offset = 4004\_5000h

| Bit   | 31     | 30 | 29 | 28    | 27 | 26   | 25 | 24 | 23      | 22  | 21    | 20 | 19      | 18    | 17 | 16 |
|-------|--------|----|----|-------|----|------|----|----|---------|-----|-------|----|---------|-------|----|----|
| R     | OUTRGF | 0  |    | ESO R |    | MODE |    |    | REFCHRG |     | DVOLT |    | EXTCHRG |       |    |    |
| W     | w1c    |    |    |       |    |      |    |    |         |     |       |    |         |       |    |    |
| Reset | 0      | 0  | 0  | 0     | 0  | 0    | 0  | 0  | 0       | 0   | 0     | 0  | 0       | 0     | 0  | 0  |
| Bit   | 15     | 14 | 13 | 12    | 11 | 10   | 9  | 8  | 7       | 6   | 5     | 4  | 3       | 2     | 1  | 0  |
| R     |        |    |    |       |    |      |    |    | TSIEN   |     |       |    | SCNIP   | EOSF  |    | 0  |
| W     |        |    |    |       |    |      |    |    | STPEN   | STM |       |    |         | CURSW |    |    |
| Reset | 0      | 0  | 0  | 0     | 0  | 0    | 0  | 0  | 0       | 0   | 0     | 0  | 0       | w1c   |    | 0  |

**TSIx\_GENCS field descriptions (continued)**

| Field            | Description  |      |  |      |   |      |   |      |  |     |       |     |        |     |        |     |        |
|------------------|--|------|--|------|---|------|---|------|--|-----|-------|-----|--------|-----|--------|-----|--------|
| 27–24<br>MODE    | <p>TSI analog modes setup and status bits.</p> <p>Set up TSI analog modes, especially, setting MODE[3:2] to not 2'b00 will configure TSI to noise detection modes. MODE[1:0] take no effect on TSI operation mode and should always write to 2'b00 for setting up. When reading this field will return the analog status. Refer to chapter "Noise detection mode" for details.</p> <table> <tr><td>0000</td><td>Set TSI in capacitive sensing(non-noise detection) mode.</td></tr> <tr><td>0100</td><td>Set TSI analog to work in single threshold noise detection mode and the frequency limitation circuit is disabled.</td></tr> <tr><td>1000</td><td>Set TSI analog to work in single threshold noise detection mode and the frequency limitation circuit is enabled to work in higher frequencies operations.</td></tr> <tr><td>1100</td><td>Set TSI analog to work in automatic noise detection mode.</td></tr> </table> | 0000 | Set TSI in capacitive sensing(non-noise detection) mode.       | 0100 | Set TSI analog to work in single threshold noise detection mode and the frequency limitation circuit is disabled. | 1000 | Set TSI analog to work in single threshold noise detection mode and the frequency limitation circuit is enabled to work in higher frequencies operations. | 1100 | Set TSI analog to work in automatic noise detection mode.      |     |       |     |        |     |        |     |        |
| 0000             | Set TSI in capacitive sensing(non-noise detection) mode.   |      |  |      |   |      |   |      |  |     |       |     |        |     |        |     |        |
| 0100             | Set TSI analog to work in single threshold noise detection mode and the frequency limitation circuit is disabled.  |      |  |      |   |      |   |      |  |     |       |     |        |     |        |     |        |
| 1000             | Set TSI analog to work in single threshold noise detection mode and the frequency limitation circuit is enabled to work in higher frequencies operations.  |      |  |      |   |      |   |      |  |     |       |     |        |     |        |     |        |
| 1100             | Set TSI analog to work in automatic noise detection mode.  |      |  |      |   |      |   |      |  |     |       |     |        |     |        |     |        |
| 23–21<br>REFCHRG | <p>REFCHRG</p> <p>These bits indicate the reference oscillator charge and discharge current value.</p> <table> <tr><td>000</td><td>500 nA.</td></tr> <tr><td>001</td><td>1 µA.</td></tr> <tr><td>010</td><td>2 µA.</td></tr> <tr><td>011</td><td>4 µA.</td></tr> <tr><td>100</td><td>8 µA.</td></tr> <tr><td>101</td><td>16 µA.</td></tr> <tr><td>110</td><td>32 µA.</td></tr> <tr><td>111</td><td>64 µA.</td></tr> </table>   | 000  | 500 nA.  | 001  | 1 µA.   | 010  | 2 µA.   | 011  | 4 µA.  | 100 | 8 µA. | 101 | 16 µA. | 110 | 32 µA. | 111 | 64 µA. |
| 000              | 500 nA.  |      |  |      |   |      |   |      |  |     |       |     |        |     |        |     |        |
| 001              | 1 µA.  |      |  |      |   |      |   |      |  |     |       |     |        |     |        |     |        |
| 010              | 2 µA.  |      |  |      |   |      |   |      |  |     |       |     |        |     |        |     |        |
| 011              | 4 µA.  |      |  |      |   |      |   |      |  |     |       |     |        |     |        |     |        |
| 100              | 8 µA.  |      |  |      |   |      |   |      |  |     |       |     |        |     |        |     |        |
| 101              | 16 µA.   |      |  |      |   |      |   |      |  |     |       |     |        |     |        |     |        |
| 110              | 32 µA.   |      |  |      |   |      |   |      |  |     |       |     |        |     |        |     |        |
| 111              | 64 µA.   |      |  |      |   |      |   |      |  |     |       |     |        |     |        |     |        |
| 20–19<br>DVOLT   | <p>DVOLT</p> <p>These bits indicate the oscillator's voltage rails as below.</p> <table> <tr><td>00</td><td>DV = 1.03 V; V<sub>P</sub> = 1.33 V; V<sub>m</sub> = 0.30 V.</td></tr> <tr><td>01</td><td>DV = 0.73 V; V<sub>P</sub> = 1.18 V; V<sub>m</sub> = 0.45 V.</td></tr> <tr><td>10</td><td>DV = 0.43 V; V<sub>P</sub> = 1.03 V; V<sub>m</sub> = 0.60 V.</td></tr> <tr><td>11</td><td>DV = 0.29 V; V<sub>P</sub> = 0.95 V; V<sub>m</sub> = 0.67 V.</td></tr> </table>  | 00   | DV = 1.03 V; V <sub>P</sub> = 1.33 V; V <sub>m</sub> = 0.30 V. | 01   | DV = 0.73 V; V <sub>P</sub> = 1.18 V; V <sub>m</sub> = 0.45 V.  | 10   | DV = 0.43 V; V <sub>P</sub> = 1.03 V; V <sub>m</sub> = 0.60 V.  | 11   | DV = 0.29 V; V <sub>P</sub> = 0.95 V; V <sub>m</sub> = 0.67 V. |     |       |     |        |     |        |     |        |
| 00               | DV = 1.03 V; V <sub>P</sub> = 1.33 V; V <sub>m</sub> = 0.30 V.   |      |  |      |   |      |   |      |  |     |       |     |        |     |        |     |        |
| 01               | DV = 0.73 V; V <sub>P</sub> = 1.18 V; V <sub>m</sub> = 0.45 V.   |      |  |      |   |      |   |      |  |     |       |     |        |     |        |     |        |
| 10               | DV = 0.43 V; V <sub>P</sub> = 1.03 V; V <sub>m</sub> = 0.60 V.   |      |  |      |   |      |   |      |  |     |       |     |        |     |        |     |        |
| 11               | DV = 0.29 V; V <sub>P</sub> = 0.95 V; V <sub>m</sub> = 0.67 V.   |      |  |      |   |      |   |      |  |     |       |     |        |     |        |     |        |
| 18–16<br>EXTCHRG | <p>EXTCHRG</p> <p>These bits indicate the electrode oscillator charge and discharge current value.</p> <table> <tr><td>000</td><td>500 nA.</td></tr> <tr><td>001</td><td>1 µA.</td></tr> <tr><td>010</td><td>2 µA.</td></tr> <tr><td>011</td><td>4 µA.</td></tr> <tr><td>100</td><td>8 µA.</td></tr> <tr><td>101</td><td>16 µA.</td></tr> <tr><td>110</td><td>32 µA.</td></tr> <tr><td>111</td><td>64 µA.</td></tr> </table>   | 000  | 500 nA.  | 001  | 1 µA.   | 010  | 2 µA.   | 011  | 4 µA.  | 100 | 8 µA. | 101 | 16 µA. | 110 | 32 µA. | 111 | 64 µA. |
| 000              | 500 nA.  |      |  |      |   |      |   |      |  |     |       |     |        |     |        |     |        |
| 001              | 1 µA.  |      |  |      |   |      |   |      |  |     |       |     |        |     |        |     |        |
| 010              | 2 µA.  |      |  |      |   |      |   |      |  |     |       |     |        |     |        |     |        |
| 011              | 4 µA.  |      |  |      |   |      |   |      |  |     |       |     |        |     |        |     |        |
| 100              | 8 µA.  |      |  |      |   |      |   |      |  |     |       |     |        |     |        |     |        |
| 101              | 16 µA.   |      |  |      |   |      |   |      |  |     |       |     |        |     |        |     |        |
| 110              | 32 µA.   |      |  |      |   |      |   |      |  |     |       |     |        |     |        |     |        |
| 111              | 64 µA.   |      |  |      |   |      |   |      |  |     |       |     |        |     |        |     |        |
| 15–13<br>PS      | <p>PS</p> <p>These bits indicate the prescaler of the output of electrode oscillator.</p> <table> <tr><td>000</td><td>Electrode Oscillator Frequency divided by 1</td></tr> <tr><td>001</td><td>Electrode Oscillator Frequency divided by 2</td></tr> </table>   | 000  | Electrode Oscillator Frequency divided by 1                    | 001  | Electrode Oscillator Frequency divided by 2   |      |   |      |  |     |       |     |        |     |        |     |        |
| 000              | Electrode Oscillator Frequency divided by 1  |      |  |      |   |      |   |      |  |     |       |     |        |     |        |     |        |
| 001              | Electrode Oscillator Frequency divided by 2  |      |  |      |   |      |   |      |  |     |       |     |        |     |        |     |        |

*Table continues on the next page...*

**TSIx\_GENCS field descriptions (continued)**

| Field        | Description   |
|--------------|---|
|              | <p>010 Electrode Oscillator Frequency divided by 4<br/>     011 Electrode Oscillator Frequency divided by 8<br/>     100 Electrode Oscillator Frequency divided by 16<br/>     101 Electrode Oscillator Frequency divided by 32<br/>     110 Electrode Oscillator Frequency divided by 64<br/>     111 Electrode Oscillator Frequency divided by 128</p>  |
| 12–8<br>NSCN | <p>NSCN</p> <p>These bits indicate the scan number for each electrode. The scan number is equal to NSCN + 1, which allows the scan time ranges from 1 to 32. By default, NSCN is configured as 0, which asserts the TSI scans once on the selected electrode channel.</p> <p>00000 Once per electrode<br/>     00001 Twice per electrode<br/>     00010 3 times per electrode<br/>     00011 4 times per electrode<br/>     00100 5 times per electrode<br/>     00101 6 times per electrode<br/>     00110 7 times per electrode<br/>     00111 8 times per electrode<br/>     01000 9 times per electrode<br/>     01001 10 times per electrode<br/>     01010 11 times per electrode<br/>     01011 12 times per electrode<br/>     01100 13 times per electrode<br/>     01101 14 times per electrode<br/>     01110 15 times per electrode<br/>     01111 16 times per electrode<br/>     10000 17 times per electrode<br/>     10001 18 times per electrode<br/>     10010 19 times per electrode<br/>     10011 20 times per electrode<br/>     10100 21 times per electrode<br/>     10101 22 times per electrode<br/>     10110 23 times per electrode<br/>     10111 24 times per electrode<br/>     11000 25 times per electrode<br/>     11001 26 times per electrode<br/>     11010 27 times per electrode<br/>     11011 28 times per electrode<br/>     11100 29 times per electrode<br/>     11101 30 times per electrode<br/>     11110 31 times per electrode<br/>     11111 32 times per electrode</p> |
| 7<br>TSIEN   | <p>Touch Sensing Input Module Enable</p> <p>This bit enables TSI module.</p>  |

*Table continues on the next page...*

**TSIx\_GENCS field descriptions (continued)**

| Field         | Description  |
|---------------|--|
|               | <p>0 TSI module disabled.<br/>1 TSI module enabled.</p>  |
| 6<br>TSIIEN   | <p>Touch Sensing Input Interrupt Enable</p> <p>This bit enables TSI module interrupt request to CPU when the scan completes. The interrupt will wake MCU from low power mode if this interrupt is enabled.</p> <p>0 TSI interrupt is disabled.<br/>1 TSI interrupt is enabled.</p> |
| 5<br>STPE     | <p>TSI STOP Enable</p> <p>This bit enables TSI module function in low power modes (stop, VLPS, LLS and VLLS{3,2,1}).</p> <p>0 TSI is disabled when MCU goes into low power mode.<br/>1 Allows TSI to continue running in all low power modes.</p>                                  |
| 4<br>STM      | <p>Scan Trigger Mode</p> <p>This bit specifies the trigger mode. User is allowed to change this bit when TSI is not working in progress.</p> <p>0 Software trigger scan.<br/>1 Hardware trigger scan.</p>  |
| 3<br>SCNIP    | <p>Scan In Progress Status</p> <p>This read-only bit indicates if scan is in progress. This bit will get asserted after the analog bias circuit is stable after a trigger and it changes automatically by the TSI.</p> <p>0 No scan in progress.<br/>1 Scan in progress.</p>       |
| 2<br>EOSF     | <p>End of Scan Flag</p> <p>This flag is set when all active electrodes are finished scanning after a scan trigger. Write "1", when this flag is set, to clear it.</p> <p>0 Scan not complete.<br/>1 Scan complete.</p>   |
| 1<br>CURSW    | <p>CURSW</p> <p>This bit specifies if the current sources of electrode oscillator and reference oscillator are swapped.</p> <p>0 The current source pair are not swapped.<br/>1 The current source pair are swapped.</p>   |
| 0<br>Reserved | <p>This field is reserved.</p> <p>This read-only field is reserved and always has the value 0.</p>   |

## 42.3.2 TSI DATA Register (TSIx\_DATA )

Address: 4004\_5000h base + 4h offset = 4004\_5004h

| Bit   | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 | 23     | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|-------|----|----|----|----|----|----|----|----|--------|----|----|----|----|----|----|----|
| R     |    |    |    |    |    |    |    |    | 0      | 0  |    |    |    | 0  |    |    |
|       |    |    |    |    |    |    |    |    |        |    |    |    |    |    |    |    |
| W     |    |    |    |    |    |    |    |    |        |    |    |    |    |    |    |    |
| Reset | 0  | 0  | 0  | 0  | 0  | 0  | 0  | 0  | 0      | 0  | 0  | 0  | 0  | 0  | 0  | 0  |
| Bit   | 15 | 14 | 13 | 12 | 11 | 10 | 9  | 8  | 7      | 6  | 5  | 4  | 3  | 2  | 1  | 0  |
| R     |    |    |    |    |    |    |    |    | TSICNT |    |    |    |    |    |    |    |
|       |    |    |    |    |    |    |    |    |        |    |    |    |    |    |    |    |
| W     |    |    |    |    |    |    |    |    |        |    |    |    |    |    |    |    |
| Reset | 0  | 0  | 0  | 0  | 0  | 0  | 0  | 0  | 0      | 0  | 0  | 0  | 0  | 0  | 0  | 0  |

### TSIx\_DATA field descriptions

| Field          | Description   |
|----------------|---|
| 31–28<br>TSICH | <p>TSICH</p> <p>These bits specify current channel to be measured. In hardware trigger mode (TSI_GENCS[STM] = 1), the scan will not start until the hardware trigger occurs. In software trigger mode (TSI_GENCS[STM] = 0), the scan starts immediately when TSI_DATA[SWTS] bit is written by 1.</p> <p>0000 Channel 0.<br/>0001 Channel 1.<br/>0010 Channel 2.<br/>0011 Channel 3.<br/>0100 Channel 4.<br/>0101 Channel 5.<br/>0110 Channel 6.<br/>0111 Channel 7.<br/>1000 Channel 8.<br/>1001 Channel 9.<br/>1010 Channel 10.<br/>1011 Channel 11.<br/>1100 Channel 12.<br/>1101 Channel 13.</p> |

Table continues on the next page...

**TSIx\_DATA field descriptions (continued)**

| Field             | Description   |
|-------------------|---|
|                   | 1110 Channel 14.<br>1111 Channel 15.  |
| 27–24<br>Reserved | This field is reserved.<br>This read-only field is reserved and always has the value 0.   |
| 23<br>DMAEN       | DMA Transfer Enabled<br><br>This bit is used together with the TSI interrupt enable bits(TSIIIE, ESOR) to generate a DMA transfer request instead of an interrupt.<br><br>0 Interrupt is selected when the interrupt enable bit is set and the corresponding TSI events assert.<br>1 DMA transfer request is selected when the interrupt enable bit is set and the corresponding TSI events assert. |
| 22<br>SWTS        | Software Trigger Start<br><br>This write-only bit is a software start trigger. When STM bit is clear, write "1" to this bit will start a scan. The electrode channel to be scanned is determinated by TSI_DATA[TSICH] bits.<br><br>0 No effect.<br>1 Start a scan to determine which channel is specified by TSI_DATA[TSICH].   |
| 21–16<br>Reserved | This field is reserved.<br>This read-only field is reserved and always has the value 0.   |
| 15–0<br>TSICNT    | TSI Conversion Counter Value<br><br>These read-only bits record the accumulated scan counter value ticked by the reference oscillator.  |

**42.3.3 TSI Threshold Register (TSIx\_TSHD)**

Address: 4004\_5000h base + 8h offset = 4004\_5008h

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |  |
|-----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|---|---|---|---|---|---|---|---|---|---|--|
| R   |    |    |    |    |    |    |    |    |    |    |    |    |    |    |    |    |    |    |    |    |    |    |   |   |   |   |   |   |   |   |   |   |  |
| W   |    |    |    |    |    |    |    |    |    |    |    |    |    |    |    |    |    |    |    |    |    |    |   |   |   |   |   |   |   |   |   |   |  |

Reset 0

**TSIx\_TSHD field descriptions**

| Field           | Description   |
|-----------------|---|
| 31–16<br>THRESH | TSI Wakeup Channel High-threshold<br><br>This half-word specifies the high threshold of the wakeup channel. |
| 15–0<br>THRESL  | TSI Wakeup Channel Low-threshold<br><br>This half-word specifies the low threshold of the wakeup channel.   |

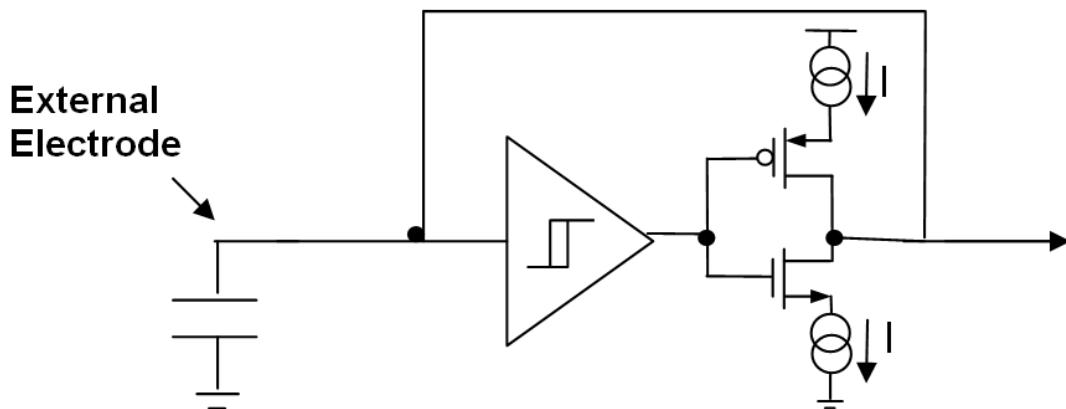
**42.4 Functional description**

## 42.4.1 Capacitance measurement

The electrode pin capacitance measurement uses a dual oscillator approach. The frequency of the TSI electrode oscillator depends on the external electrode capacitance and the TSI module configuration. After going to a configurable prescaler, the TSI electrode oscillator signal goes to the input of the module counter. The time for the module counter to reach its module value is measured using the TSI reference oscillator. The measured electrode capacitance is directly proportional to the time.

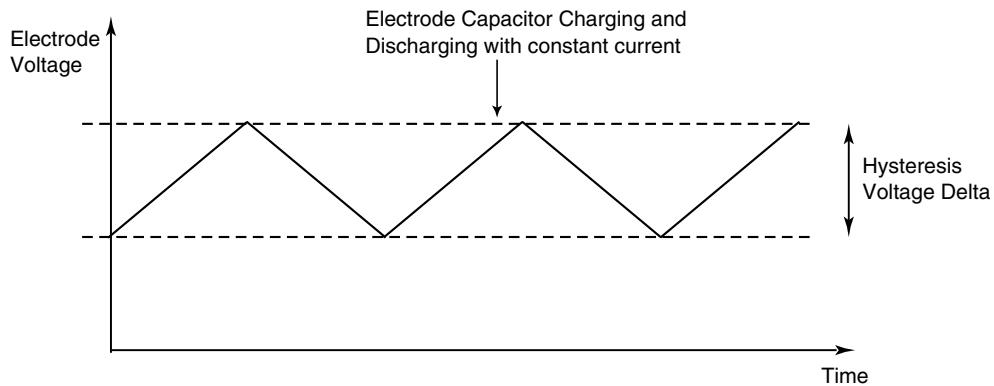
### 42.4.1.1 TSI electrode oscillator

The TSI electrode oscillator circuit is illustrated in the following figure. A configurable constant current source is used to charge and discharge the external electrode capacitance. A buffer hysteresis defines the oscillator delta voltage. The delta voltage defines the margin of high and low voltage which are the reference input of the comparator in different time.



**Figure 42-8. TSI electrode oscillator circuit**

The current source applied to the pad capacitance is controlled by the SCANC[EXTCHRG]. The hysteresis delta voltage is defined in the module electrical specifications present in the device Data Sheet. The figure below shows the voltage amplitude waveform of the electrode capacitance charging and discharging with a programmable current.

**Figure 42-9. TSI electrode oscillator chart**

The oscillator frequency is given by the following equation

$$F_{elec} = \frac{I}{2 * C_{elec} * \Delta V}$$

**Figure 42-10. Equation 1: TSI electrode oscillator frequency**

Where:

I: constant current

$C_{elec}$ : electrode capacitance

$\Delta V$ : Hysteresis delta voltage

So by this equation, for example, an electrode with  $C_{elec} = 20 \text{ pF}$ , with a current source of  $I = 16 \mu\text{A}$  and  $\Delta V = 600 \text{ mV}$  have the following oscillation frequency:

$$F_{elec} = \frac{16 \mu\text{A}}{2 * 20 \text{ pF} * 600 \text{ mV}} = 0.67 \text{ MHz}$$

**Figure 42-11. Equation 2: TSI electrode oscillator frequency**

The current source is used to accommodate the TSI electrode oscillator frequency with different electrode capacitance sizes.

#### 42.4.1.2 Electrode oscillator and counter module control

The TSI oscillator frequency signal goes through a prescaler defined by the GENCS[PS] and then enters in a modulus counter. The bit field GENCS[NSCN] defines the maximum count value of the modulus counter.

## Functional description

The pin capacitance sampling time is given by the time the module counter takes to go from zero to its maximum value, defined by NSCN. The electrode sample time is expressed by the following equation:

$$T_{cap\_samp} = \frac{PS * NSCN}{F_{elec}}$$

Using Equation 1.

$$T_{cap\_samp} = \frac{2 * PS * NSCN * C_{elec} * \Delta V}{I}$$

**Figure 42-12. Equation 3: Electrode sampling time**

Where:

PS: prescaler value

NSCN: module counter maximum value

I: constant current

C<sub>elec</sub>: electrode capacitance

ΔV: Hysteresis delta voltage

By this equation we have that an electrode with C = 20 pF, with a current source of I = 16 μA and ΔV = 600 mV, PS = 2 and NSCN = 16 have the following sampling time:

$$T_{cap\_samp} = \frac{2 * 2 * 16 * 20pF * 600mV}{16\mu A} = 48\mu s$$

### 42.4.1.3 TSI reference oscillator

The TSI reference oscillator has the same topology of the TSI electrode oscillator. The TSI reference oscillator instead of using an external capacitor for the electrode oscillator has an internal reference capacitor.

The TSI reference oscillator has an independent programmable current source controlled by the SCANC[REFCHRG].

The reference oscillator frequency is given by the following equation:

$$F_{ref\_osc} = \frac{I_{ref}}{2 * C_{ref} * \Delta V}$$

**Figure 42-13. Equation 4: TSI reference oscillator frequency**

Where:

$C_{ref}$ : Internal reference capacitor

$I_{ref}$ : Reference oscillator current source

$\Delta V$  : Hysteresis delta voltage

Considering  $C_{ref} = 1.0 \text{ pF}$ ,  $I_{ref} = 12 \mu\text{A}$  and  $\Delta V = 600 \text{ mV}$ , follows

$$F_{ref\_osc} = \frac{12\mu\text{A}}{2 * 1.0\text{pF} * 600\text{mV}} = 10.0\text{MHz}$$

## 42.4.2 TSI measurement result

The capacitance measurement result is defined by the number of TSI reference oscillator periods during the sample time and is stored in the TSICHnCNT register.

$$\text{TSICHnCNT} = T_{cap\_samp} * F_{ref\_osc}$$

Using Equation 2 and Equation 1 follows:

$$\text{TSICHnCNT} = \frac{I_{ref} * PS * NSCN}{C_{ref} * I_{elec}} * C_{elec}$$

**Figure 42-14. Equation 5: Capacitance result value**

In the example where  $F_{ref\_osc} = 10.0\text{MHz}$  and  $T_{cap\_samp} = 48 \mu\text{s}$ ,  $\text{TSICHnCNT} = 480$

## 42.4.3 Enable TSI module

The TSI module can be fully functional in run, wait and low power modes. The TSI\_GENCS[TSIEN] bit must be set to enable the TSI module in run and wait mode. When TSI\_GENCS[STPE] bit is set, it allows the TSI module to work in low power mode.

## 42.4.4 Software and hardware trigger

The TSI module allows a software or hardware trigger to start a scan. When a software trigger is applied ( TSI\_GENCS[STM] bit clear), the TSI\_GENCS[SWTS] bit must be written "1" to start the scan electrode channel that is identified by TSI\_DATA[TSICH]. When a hardware trigger is applied ( TSI\_GENCS[STM] bit set), the TSI will not start scanning until the hardware trigger arrives. The hardware trigger is different depending on the MCU configuration. Generally, it could be an event that RTC overflows. See chip configuration section for details.

#### 42.4.5 Scan times

The TSI provides multi-scan function. The number of scans is indicated by the TSI\_GENCS[NSCN] bits that allow the scan number from 1 to 32. When TSI\_GENCS[NSCN] is set to 0 (only once), the single scan is engaged. The 16-bit counter accumulates all scan results until the NSCN time scan completes, and users can read TSI\_DATA[TSICNT] to get this accumulation. When DMA transfer is enabled, the counter values can also be read out by DMA engine.

#### 42.4.6 Clock setting

TSI is built with dual oscillator architecture. In normal sensing application, the reference oscillator clock is the only clock source for operations. The reference clock is used to measure the electrode oscillator by ticking a 16-bit counter. The reference oscillator frequency depends on the current source setting. Please refer to the [Current source](#) for more details.

The output of electrode oscillator has several prescalers up to 128 indicated by TSI\_GENCS[PS]bits. This allows a flexible counter configuration for different electrode oscillator frequency.

#### 42.4.7 Reference voltage

The TSI module offers a internal reference voltage for both electrode oscillator and reference oscillator. The internal reference voltage can work in low power modes even when the MCU regulator is partially powered down, which is ideally for low-power touch detection.

The charge and discharge difference voltage is configurable upon the setting of TSI\_GENCS[DVOLT] bits. The following table shows the all the delta voltage configurations. Note this table doesn't apply to noise mode, where the details should be consult form noise modes chapters.

**Table 42-11. Delta voltage configuration**

| DVOLT | V <sub>p</sub> (V) | V <sub>m</sub> (V) | ΔV (V) |
|-------|--------------------|--------------------|--------|
| 00    | 1.328              | 0.302              | 1.026  |
| 01    | 1.111              | 0.519              | 0.592  |
| 10    | 0.986              | 0.644              | 0.342  |
| 11    | 0.914              | 0.716              | 0.198  |

## 42.4.8 Current source

The TSI module supports eight different current source power to increment from 500 nA to 64uA. The TSI\_GENCS[EXTCHRG] bits determine the current of electrode oscillator that charges and discharges external electrodes. The TSI\_GENCS[REFCHRG] bits determine the current of reference oscillator on which the internal reference clock depends. The lower current source takes more time for charge and discharge, which is useful to detect high-accuracy change. The higher current source takes less time, which can be used to charge a big electrode by less power consumption.

The TSI\_GENCS[CURSW] bit allows the current source to swap, so that the reference oscillator and electrode oscillator use the opposite current sources. When TSI\_GENCS[CURSW] is set and the current sources are swapped, the TSI\_GENCS[EXTCHRG] bits and TSI\_GENCS[REFCHRG] bits still control the corresponding current sources, that is, TSI\_GENCS[EXTCHRG] controls the reference oscillator current and TSI\_GENCS[REFCHRG] controls the electrode oscillator current.

## 42.4.9 End of scan

As a scan starts, [SCNIP] bit is set to indicate scan is in progress. When the scan completes, the [EOSF] bit is set. Before clearing the [EOSF] bit, the value in TSI\_DATA[TSICNT] must be read. If the TSI\_GENCS[TSIIEN] bit is set and TSI\_GENCS[DMAEN] is not set, an interrupt is submitted to CPU for post-processing immediately. The interrupt is also optional to wake MCU to execute ISR if it is in low power mode. When DMA function is enabled by setting TSI\_GENCS[TSIIEN] and TSI\_GENCS[ESOR], as soon as scan completes, a DMA transfer request is asserted to DMA controller for data movement, generally, DMA engine will fetch TSI conversion result from TSI\_DATA register, store it to other memory space and then refresh the TSI scan channel index(TSI\_DATA[TSICH]) for next loop. When DMA transfer is done, TSI\_GENCS[EOSF] is cleared automatically.

## 42.4.10 Out-of-range interrupt

If enabled, TSI will scan the electrode specified by TSI\_DATA[TSICH] as soon as the trigger arrives. The TSI\_GENCS[OUTRGF] flag generates a TSI interrupt request if the TSI\_GENCS[TSIIIE] bit is set and GENCS[ESOR] bit is cleared. With this configuration, after the end-of-electrode scan, the electrode capacitance will be converted and stored to the result register TSI\_DATA[TSICNT], the out-of-range interrupt is only requested if

there is a considerable capacitance change defined by the TSI\_TSHD. For instance, if in low power mode the electrode capacitance does not vary, the out-of-range interrupt does not interrupt the CPU. This interrupt will not happen in noise detection mode. It is worthy to note that when the counter value reaches 0xFFFF is treated as an extreme case the out-of-range will not happen. Also in noise detection mode, the out-of-range will not assert either.

#### **42.4.11 Wake up MCU from low power modes**

In low power modes, once enabled by TSI\_GENCS[STPE] and TSI\_GENCS[TSIIIE], TSI can bring MCU out of its low power modes(STOP, VLPS, VLLS,etc) by either end of scan or out of range interrupt, that is, if TSI\_GENCS[ESOR] is set, end of scan interrupt is selected and otherwise, out of range is selected.

#### **42.4.12 DMA function support**

Transmit by DMA is supported only when TSI\_DATA[DMAEN] is set. A DMA transfer request is asserted when all of TSI\_GENCS[EOSF], TSI\_GENCS[ESOR] and TSI\_GENCS[TSIIIE] are set. Then the on-chip DMA controller detects this request and transfers data between memory space and TSI register space. After the data tranfer, DMA DONE is asserted to clear TSI\_GENCS[EOSF] automatically. This function is normally used by DMA controller to get the conversion result from TSI\_DATA[TSICNT] upon a end-of-scan event and then refresh the channel index(TSI\_DATA[TSICH]) for next trigger. DMA function is not available when MCU is in stop modes.

#### **42.4.13 Noise detection mode**

The noise detection mode change the circuit configuration as shown in the following figure. With this configuration, it is possible to detect touch with high levels of EMC noise present. To enter the test mode, see TSIx\_GENCS[MODE] register field.

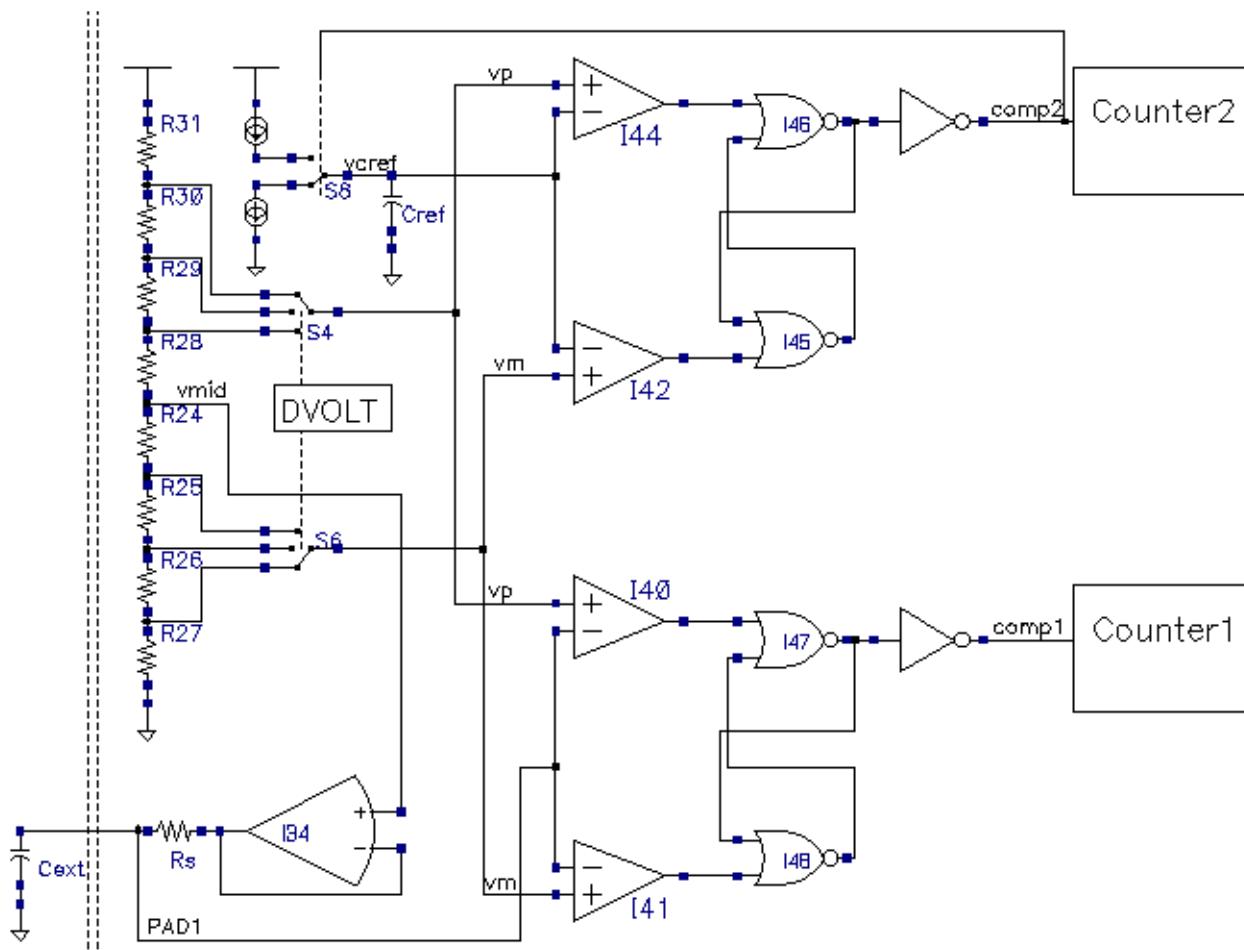
In noise detection mode the reference oscillator has the same configuration except the output goes to Counter2 and this counter will have the it's maximum count set by NSCN.  $2^{(PS)}$ . This means this oscillator will setup the noise detection mode sense duration.

The blocks of external oscillator is changed and instead of an oscillator the circuit implements an RF amplitude detection. The threshold for this amplitude detection is set by DVOLT register bits.

Also the external voltage is biased by vmid voltage with a Rs series resistance.

The  $v_{mid}$  voltage is defined as  $V(v_{mid}) = (V(vp) + V(vm))/2$ .

The  $R_s$  value is defined by i\_ext\_3v<2:0> register bits.



**Figure 42-15. TSI circuit in noise detection mode**

To determine the noise level the below algorithm can be used:

1. Initialize  $R_s = \text{maxrs}$ ;  $D_{volt} = \text{minD}_v$  (set other configurations also)
2. Perform a noise cycle.
3. If  $\text{TSI}_{\text{counter}} < 3$ , go to step 8
4. If  $R_s = \text{minrs}$ , go to step 6.
5. Reduce value of  $R_s$ . go to step 2
6. If  $D_{volt} = \text{maxD}_v$ , go to END
7. Increase value of  $D_{volt}$ . Set  $R_s = \text{maxrs}$ . go to step 2
8. If  $R_s > \text{minrs}$ , (Reduce value of  $R_s$ , go to END)
9.  $R_s = \text{maxrs}$ , reduce value of  $D_{volt}$ .
10. END Get value of  $R_s$  and  $D_{volt}$ .

## Functional description

One example of noise detection mode is shown in the following figure. in this figure the TSI is working in capacitive mode until 30 $\mu$ S when it is changed to noise detection mode. In noise detection mode the selected pad is biased with 0.815V and all AC waveform in this pad is caused by a noise source external to IC.

It is possible to observe in the following figure that, in noise detection mode, the clkref output has the peak detection and the number of detected peaks can be counted or used by digital block. The clkext output has the internal oscillator output and can be used to set the maximum noise detection time window.

The waveform of the following figure shows two operations during noise detection mode:

- The V(vp) and V(vm) thresholds are changed in 34.4  $\mu$ s.
- The Rs series resistance value is changed between 184 k $\Omega$  (iext<2:0>=011) and 32 k $\Omega$  (iext<2:0>=101). Because of this Rs change the amplitude of noise waveform change also.

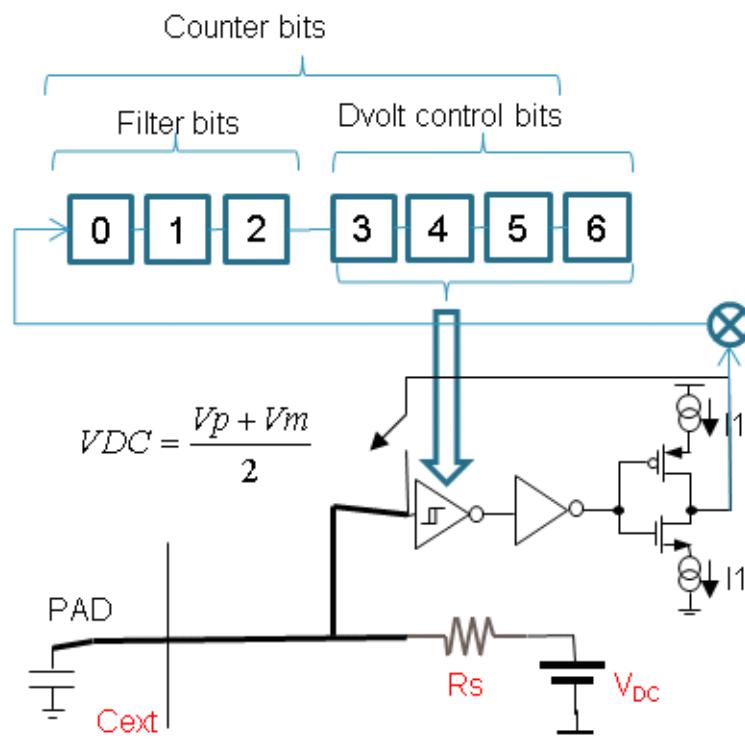


Figure 42-16. TSI noise detection mode waveform

### 42.4.13.1 Automatic noise mode

This mode is set by MODE[3:2] = 11 (noise mode 3). In this mode, the thresholds are incremented internally by the module until the point that there is no noise voltage trespassing the threshold.

The following diagram shows how it is done. The threshold comparator output goes to a counter and as the DVOLT control bits are increased the DVOLT thresholds are increased as well. The four bits are counted until 1111 (=15) and the counter is stop with this maximum value.



**Figure 42-17. Block diagram automatic noise threshold operation**

The signals that have different behavior in this noise mode (wrt capacitive mode) are shown in the following table.

**Table 42-12. Signal properties in automatic noise operation mode**

| Name      | Function   | I/O type | Power Up/Reset state |
|-----------|--|----------|----------------------|
| MODE[3:2] | 11 - Noise mode operation with frequency limitation and automatic threshold counter. | I        | 00                   |

*Table continues on the next page...*

**Table 42-12. Signal properties in automatic noise operation mode (continued)**

| Name         | Function   | I/O type | Power Up/Reset state |
|--------------|--|----------|----------------------|
| EXTCHRG[2:1] | In this operation mode these bits select the number of filter bits.<br>00 - 3 filter bits<br>01 - 2 filter bits<br>10 - 1 filter bit<br>11 - no filter bit   | I        | 00                   |
| EXTCHRG[0]   | In this operation mode this bits selects the series resistance.<br>0 - uses $R_s=32\text{ k}\Omega$<br>0 - uses $R_s=187\text{ k}\Omega$<br>Independent of this bit selection the threshold 15 is done with $R_s = 5.5\text{ k}\Omega$ | I        | 0                    |
| DVOLT[1:0]   | Select voltage rails of the internal oscillator  | I        | 00                   |
| MODE[3:0]    | DVOLT counter bits output.<br>This register content is reset always the MODE[3:2] is not 11, and after beginning of automatic noise mode operation.  | O        | 0000                 |

#### 42.4.13.2 Single threshold noise modes

These modes reset by MODE[3:2]=01 and 10. The difference between these two modes is that in mode 2 (MODE[3:2]=10) there is a frequency limitation circuit that enables the circuit to operate in higher frequencies. In mode 1 (MODE[3:2]=01) this frequency limitation circuit is not enabled.

In this mode the thresholds are set by user via register bits as described in the following table.

During these modes the internal oscillator rails are set to the maximum (equivalent to DVOLT[1:0]=00).

**Table 42-13. Signal properties in single noise modes (1,2)**

| Name      | Function   | I/O type | Power up / reset |
|-----------|--|----------|------------------|
| MODE[3:2] | 01 or 10- Single threshold noise mode operation. | I        | 00               |

*Table continues on the next page...*

**Table 42-13. Signal properties in single noise modes (1,2) (continued)**

| Name                        | Function  | I/O type | Power up / reset |
|-----------------------------|---|----------|------------------|
| DVOLT[1:0],<br>EXTCHRG[2:1] | <p>In this operation mode these 4 bits are used select the noise threshold.</p> <p>0000 - DVpm = 0.038 V, Vp = 0.834 V, Vm = 0.796 V<br/>           0001 - DVpm = 0.050 V, Vp = 0.830 V, Vm = 0.790 V<br/>           0010 - DVpm = 0.066 V, Vp = 0.848 V, Vm = 0.782 V<br/>           0011 - DVpm = 0.087 V, Vp = 0.858 V, Vm = 0.772 V<br/>           0100 - DVpm = 0.114 V, Vp = 0.872 V, Vm = 0.758 V<br/>           0101 - DVpm = 0.150 V, Vp = 0.890 V, Vm = 0.740 V<br/>           0110 - DVpm = 0.197 V, Vp = 0.914 V, Vm = 0.716 V<br/>           0111 - DVpm = 0.260 V, Vp = 0.945 V, Vm = 0.685 V<br/>           1000 - DVpm = 0.342 V, Vp = 0.986 V, Vm = 0.644 V<br/>           1001 - DVpm = 0.450 V, Vp = 1.040 V, Vm = 0.590 V<br/>           1010 - DVpm = 0.592 V, Vp = 1.111 V, Vm = 0.519 V<br/>           1011 - DVpm = 0.780 V, Vp = 1.205 V, Vm = 0.425 V<br/>           1100 - DVpm = 1.026 V, Vp = 1.328 V, Vm = 0.302 V<br/>           1101 - DVpm = 1.350 V, Vp = 1.490 V, Vm = 0.140 V<br/>           1110 - DVpm = 1.630 V, Vp = 1.630 V, Vm = 0 V<br/>           1111 - DVpm = 1.630 V, Vp = 1.630 V, Vm = 0 V</p> | I        | XXXX             |
| EXTCHRG[0]                  | <p>In this operation mode this bits selects the series resistance.</p> <p>0 - uses <math>R_s = 32\text{ k}\Omega</math>.</p> <p>1- uses <math>R_s = 187\text{ k}\Omega</math>.</p> <p>Independent of this bit selection the threshold 15 is done with <math>R_s = 5.5\text{ k}\Omega</math>.</p>  | I        | XX               |

# **Appendix A**

## **Revision History of this Document**

This appendix describes corrections to the this reference manual for convenience. Grammatical and formatting changes are not listed here unless the meaning of something changed.

### **A.1 Changes between revisions 3 and 2**

**Table A-1. Changes between revisions 3 and 2**

| <b>Chapter</b>                              | <b>Description</b>   |
|---|--|
| Flash Memory Module<br>(FTFA)               | <ul style="list-style-type: none"><li>• Updated register absolute address.</li></ul> |
| 12-bit Digital-to-Analog<br>Converter (DAC) | <ul style="list-style-type: none"><li>• Updated register absolute address.</li></ul> |

## **How to Reach Us:**

### **Home Page:**

[www.freescale.com](http://www.freescale.com)

### **Web Support:**

<http://www.freescale.com/support>

### **USA/Europe or Locations Not Listed:**

Freescale Semiconductor

Technical Information Center, EL516  
2100 East Elliot Road  
Tempe, Arizona 85284  
+1-800-521-6274 or +1-480-768-2130  
[www.freescale.com/support](http://www.freescale.com/support)

### **Europe, Middle East, and Africa:**

Freescale Halbleiter Deutschland GmbH

Technical Information Center

Schatzbogen 7  
81829 Muenchen, Germany  
+44 1296 380 456 (English)  
+46 8 52200080 (English)  
+49 89 92103 559 (German)  
+33 1 69 35 48 48 (French)  
[www.freescale.com/support](http://www.freescale.com/support)

### **Japan:**

Freescale Semiconductor Japan Ltd.

Headquarters

ARCO Tower 15F  
1-8-1, Shimo-Meguro, Meguro-ku,  
Tokyo 153-0064  
Japan  
0120 191014 or +81 3 5437 9125  
[support.japan@freescale.com](mailto:support.japan@freescale.com)

### **Asia/Pacific:**

Freescale Semiconductor China Ltd.

Exchange Building 23F

No. 118 Jianguo Road  
Chaoyang District  
Beijing 100022  
China  
+86 10 5879 8000  
[support.asia@freescale.com](mailto:support.asia@freescale.com)

Information in this document is provided solely to enable system and software implementers to use Freescale Semiconductors 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 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 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 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 claims alleges that Freescale Semiconductor was negligent regarding the design or manufacture of the part.

RoHS-compliant and/or Pb-free versions of Freescale products have the functionality and electrical characteristics as their non-RoHS-compliant and/or non-Pb-free counterparts. For further information, see <http://www.freescale.com> or contact your Freescale sales representative.

For information on Freescale's Environmental Products program, go to <http://www.freescale.com/epp>.

Freescale™ and the Freescale logo are trademarks of Freescale Semiconductor, Inc. All other product or service names are the property of their respective owners.

© 2012 Freescale Semiconductor, Inc.

