

KL03 Sub-Family Reference Manual

Supports: MKL03Z8VFG4(R), MKL03Z16VFG4(R),
MKL03Z32VFG4(R), MKL03Z32CAF4R, MKL03Z32CBF4R,
MKL03Z8VFK4(R), MKL03Z16VFK4(R), and MKL03Z32VFK4(R)

Document Number: KL03P24M48SF0RM
Rev. 5, July, 2017



Contents

| Section number | Title | Page |
|--|---------------------------------------|------|
| Chapter 1 About This Document | | |
| 1.1 | Overview..... | 27 |
| 1.1.1 | Purpose..... | 27 |
| 1.1.2 | Audience..... | 27 |
| 1.2 | Conventions..... | 27 |
| 1.2.1 | Numbering systems..... | 27 |
| 1.2.2 | Typographic notation..... | 28 |
| 1.2.3 | Special terms..... | 28 |
| Chapter 2 Introduction | | |
| 2.1 | KL03 Sub-family introduction..... | 29 |
| 2.2 | Module functional categories..... | 29 |
| 2.2.1 | ARM Cortex-M0+ core modules..... | 30 |
| 2.2.2 | System modules..... | 31 |
| 2.2.3 | Memories and memory interfaces..... | 32 |
| 2.2.4 | Clocks..... | 32 |
| 2.2.5 | Analog modules..... | 32 |
| 2.2.6 | Timer modules..... | 33 |
| 2.2.7 | Communication interfaces..... | 33 |
| 2.2.8 | Human-machine interfaces..... | 34 |
| 2.3 | Module to module interconnects..... | 34 |
| 2.3.1 | Interconnection overview..... | 34 |
| 2.3.2 | Analog reference options..... | 35 |
| 2.4 | Orderable part numbers..... | 36 |
| Chapter 3 Core Overview | | |
| 3.1 | ARM Cortex-M0+ core introduction..... | 37 |

| Section number | Title | Page |
|-----------------------|---|-------------|
| 3.1.1 | Buses, interconnects, and interfaces..... | 37 |
| 3.1.2 | System tick timer..... | 37 |
| 3.1.3 | Debug facilities..... | 37 |
| 3.1.4 | Core privilege levels..... | 38 |
| 3.2 | Nested vectored interrupt controller (NVIC) | 38 |
| 3.2.1 | Interrupt priority levels..... | 38 |
| 3.2.2 | Non-maskable interrupt..... | 38 |
| 3.2.3 | Interrupt channel assignments..... | 38 |
| 3.3 | AWIC introduction..... | 41 |
| 3.3.1 | Wake-up sources..... | 41 |

Chapter 4 Memory and Memory Map

| | | |
|-------|--|----|
| 4.1 | Flash memory..... | 43 |
| 4.1.1 | Flash memory map..... | 43 |
| 4.1.2 | Flash security..... | 44 |
| 4.1.3 | Flash modes..... | 44 |
| 4.1.4 | Erase all flash contents..... | 44 |
| 4.1.5 | FTFA_FOPT register..... | 44 |
| 4.2 | SRAM..... | 45 |
| 4.2.1 | SRAM sizes..... | 45 |
| 4.2.2 | SRAM ranges..... | 45 |
| 4.2.3 | SRAM retention in low power modes..... | 46 |
| 4.3 | System Register file..... | 46 |
| 4.4 | Memory map..... | 47 |
| 4.4.1 | Introduction..... | 47 |
| 4.4.2 | System memory map..... | 47 |
| 4.4.3 | Flash memory map..... | 48 |
| 4.4.4 | SRAM memory map..... | 48 |
| 4.4.5 | Bit Manipulation Engine..... | 49 |

| Section number | Title | Page |
|---|---|-------------|
| 4.4.6 | Peripheral bridge (AIPS-Lite) memory map..... | 49 |
| 4.4.7 | Private Peripheral Bus (PPB) memory map..... | 54 |
| Chapter 5 Clock Distribution | | |
| 5.1 | Introduction..... | 55 |
| 5.2 | Programming model..... | 55 |
| 5.3 | High-level device clocking diagram..... | 55 |
| 5.4 | Clock definitions..... | 56 |
| 5.4.1 | Device clock summary..... | 57 |
| 5.5 | Internal clocking requirements..... | 59 |
| 5.5.1 | Clock divider values after reset..... | 59 |
| 5.5.2 | VLPR mode clocking..... | 60 |
| 5.6 | Clock gating..... | 60 |
| 5.7 | Module clocks..... | 60 |
| 5.7.1 | PMC 1-kHz LPO clock..... | 61 |
| 5.7.2 | COP clocking..... | 62 |
| 5.7.3 | RTC clocking..... | 62 |
| 5.7.4 | RTC_CLKOUT and CLKOUT32K clocking..... | 63 |
| 5.7.5 | LPTMR clocking..... | 63 |
| 5.7.6 | TPM clocking..... | 64 |
| 5.7.7 | LPUART clocking..... | 64 |
| Chapter 6 Reset and Boot | | |
| 6.1 | Introduction..... | 67 |
| 6.2 | Reset..... | 67 |
| 6.2.1 | Power-on reset (POR)..... | 68 |
| 6.2.2 | System reset sources..... | 68 |
| 6.2.3 | MCU resets..... | 71 |
| 6.2.4 | RESET pin | 72 |

| Section number | Title | Page |
|---------------------------------------|---|-------------|
| 6.3 | Boot..... | 72 |
| 6.3.1 | Boot sources..... | 72 |
| 6.3.2 | FOPT boot options..... | 73 |
| 6.3.3 | Boot sequence..... | 75 |
| Chapter 7 Power Management | | |
| 7.1 | Introduction..... | 77 |
| 7.2 | Clocking modes..... | 77 |
| 7.2.1 | Partial Stop..... | 77 |
| 7.2.2 | Compute Operation..... | 78 |
| 7.2.3 | Peripheral Doze..... | 79 |
| 7.2.4 | Clock gating..... | 79 |
| 7.3 | Power modes..... | 80 |
| 7.4 | Entering and exiting power modes..... | 82 |
| 7.5 | Module operation in low-power modes..... | 82 |
| Chapter 8 Security | | |
| 8.1 | Introduction..... | 87 |
| 8.1.1 | Flash security..... | 87 |
| 8.1.2 | Security interactions with other modules..... | 87 |
| Chapter 9 Debug | | |
| 9.1 | Introduction..... | 89 |
| 9.2 | Debug port pin descriptions..... | 89 |
| 9.3 | SWD status and control registers..... | 90 |
| 9.3.1 | MDM-AP Control Register..... | 91 |
| 9.3.2 | MDM-AP Status Register..... | 92 |
| 9.4 | Debug resets..... | 94 |
| 9.5 | Micro Trace Buffer (MTB)..... | 94 |
| 9.6 | Debug in low-power modes..... | 95 |

| Section number | Title | Page |
|----------------|-------------------------|------|
| 9.7 | Debug and security..... | 96 |

Chapter 10 Signal Multiplexing and Signal Descriptions

| | | |
|--------|---|-----|
| 10.1 | Introduction..... | 97 |
| 10.2 | Signal multiplexing integration..... | 97 |
| 10.2.1 | I/O Port control and interrupt module features..... | 98 |
| 10.2.2 | Clock gating..... | 98 |
| 10.2.3 | Signal multiplexing constraints..... | 98 |
| 10.3 | Pinout..... | 98 |
| 10.3.1 | KL03 signal multiplexing and pin assignments..... | 98 |
| 10.3.2 | KL03 pinouts..... | 100 |
| 10.4 | Module Signal Description Tables..... | 101 |
| 10.4.1 | Core modules..... | 101 |
| 10.4.2 | System modules..... | 102 |
| 10.4.3 | Clock modules..... | 102 |
| 10.4.4 | Memories and memory interfaces..... | 103 |
| 10.4.5 | Analog..... | 103 |
| 10.4.6 | Timer Modules..... | 103 |
| 10.4.7 | Communication interfaces..... | 104 |
| 10.4.8 | Human-machine interfaces (HMI)..... | 105 |

Chapter 11 Kinetis ROM Bootloader

| | | |
|--------|--|-----|
| 11.1 | Chip-Specific Information..... | 107 |
| 11.2 | Introduction..... | 107 |
| 11.3 | Functional Description..... | 109 |
| 11.3.1 | Memory Maps..... | 109 |
| 11.3.2 | The Kinetis Bootloader Configuration Area (BCA)..... | 110 |
| 11.3.3 | Start-up Process..... | 111 |
| 11.3.4 | Clock Configuration..... | 113 |

| Section number | Title | Page |
|-----------------------|--|-------------|
| 11.3.5 | Bootloader Entry Point..... | 114 |
| 11.3.6 | Bootloader Protocol..... | 115 |
| 11.3.7 | Bootloader Packet Types..... | 118 |
| 11.3.8 | Bootloader Command API..... | 124 |
| 11.3.9 | Bootloader Exit state..... | 136 |
| 11.4 | Peripherals Supported..... | 137 |
| 11.4.1 | I2C Peripheral..... | 137 |
| 11.4.2 | SPI Peripheral..... | 139 |
| 11.4.3 | LPUART Peripheral..... | 141 |
| 11.5 | Get/SetProperty Command Properties..... | 143 |
| 11.5.1 | Property Definitions..... | 144 |
| 11.6 | Kinetis Bootloader Status Error Codes..... | 145 |

Chapter 12 **Port Control and Interrupts (PORT)**

| | | |
|--------|---|-----|
| 12.1 | Chip-specific PORT information..... | 147 |
| 12.1.1 | GPIO instantiation information..... | 147 |
| 12.1.2 | Port control and interrupt summary..... | 148 |
| 12.2 | Introduction..... | 149 |
| 12.3 | Overview..... | 149 |
| 12.3.1 | Features..... | 149 |
| 12.3.2 | Modes of operation..... | 149 |
| 12.4 | External signal description..... | 150 |
| 12.5 | Detailed signal description..... | 150 |
| 12.6 | Memory map and register definition..... | 151 |
| 12.6.1 | Pin Control Register n (PORTx_PCRn)..... | 154 |
| 12.6.2 | Global Pin Control Low Register (PORTx_GPCLR)..... | 156 |
| 12.6.3 | Global Pin Control High Register (PORTx_GPCHR)..... | 157 |
| 12.6.4 | Interrupt Status Flag Register (PORTx_ISFR)..... | 157 |
| 12.7 | Functional description..... | 158 |

| Section number | Title | Page |
|----------------|--------------------------|------|
| 12.7.1 | Pin control..... | 158 |
| 12.7.2 | Global pin control..... | 159 |
| 12.7.3 | External interrupts..... | 159 |

Chapter 13 General-Purpose Input/Output (GPIO)

| | | |
|--------|---|-----|
| 13.1 | Chip-specific GPIO information..... | 161 |
| 13.2 | Introduction..... | 161 |
| 13.2.1 | Features..... | 161 |
| 13.2.2 | Modes of operation..... | 162 |
| 13.2.3 | GPIO signal descriptions..... | 162 |
| 13.3 | Memory map and register definition..... | 163 |
| 13.3.1 | Port Data Output Register (GPIOx_PDOR)..... | 164 |
| 13.3.2 | Port Set Output Register (GPIOx_PSOR)..... | 165 |
| 13.3.3 | Port Clear Output Register (GPIOx_PCOR)..... | 165 |
| 13.3.4 | Port Toggle Output Register (GPIOx_PTOR)..... | 166 |
| 13.3.5 | Port Data Input Register (GPIOx_PDIR)..... | 166 |
| 13.3.6 | Port Data Direction Register (GPIOx_PDDR)..... | 167 |
| 13.4 | FGPIO memory map and register definition..... | 167 |
| 13.4.1 | Port Data Output Register (FGPIOx_PDOR)..... | 168 |
| 13.4.2 | Port Set Output Register (FGPIOx_PSOR)..... | 169 |
| 13.4.3 | Port Clear Output Register (FGPIOx_PCOR)..... | 169 |
| 13.4.4 | Port Toggle Output Register (FGPIOx_PTOR)..... | 170 |
| 13.4.5 | Port Data Input Register (FGPIOx_PDIR)..... | 170 |
| 13.4.6 | Port Data Direction Register (FGPIOx_PDDR)..... | 171 |
| 13.5 | Functional description..... | 171 |
| 13.5.1 | General-purpose input..... | 171 |
| 13.5.2 | General-purpose output..... | 171 |
| 13.5.3 | IOPORT..... | 172 |

Chapter 14

| Section number | Title | Page |
|----------------|--|------|
| | System Integration Module (SIM) | |
| 14.1 | Chip-specific COP information..... | 173 |
| 14.2 | COP clocks..... | 173 |
| 14.3 | COP watchdog operation..... | 173 |
| 14.4 | Introduction..... | 175 |
| | 14.4.1 Features..... | 175 |
| 14.5 | Memory map and register definition..... | 176 |
| | 14.5.1 System Options Register 1 (SIM_SOPT1)..... | 177 |
| | 14.5.2 System Options Register 2 (SIM_SOPT2)..... | 178 |
| | 14.5.3 System Options Register 4 (SIM_SOPT4)..... | 180 |
| | 14.5.4 System Options Register 5 (SIM_SOPT5)..... | 181 |
| | 14.5.5 System Options Register 7 (SIM_SOPT7)..... | 182 |
| | 14.5.6 System Device Identification Register (SIM_SDID)..... | 184 |
| | 14.5.7 System Clock Gating Control Register 4 (SIM_SCGC4)..... | 186 |
| | 14.5.8 System Clock Gating Control Register 5 (SIM_SCGC5)..... | 187 |
| | 14.5.9 System Clock Gating Control Register 6 (SIM_SCGC6)..... | 189 |
| | 14.5.10 System Clock Divider Register 1 (SIM_CLKDIV1)..... | 190 |
| | 14.5.11 Flash Configuration Register 1 (SIM_FCFG1)..... | 192 |
| | 14.5.12 Flash Configuration Register 2 (SIM_FCFG2)..... | 193 |
| | 14.5.13 Unique Identification Register Mid-High (SIM_UIDMH)..... | 194 |
| | 14.5.14 Unique Identification Register Mid Low (SIM_UIDML)..... | 194 |
| | 14.5.15 Unique Identification Register Low (SIM_UIDL)..... | 195 |
| | 14.5.16 COP Control Register (SIM_COPC)..... | 195 |
| | 14.5.17 Service COP (SIM_SRVCOP)..... | 197 |
| 14.6 | Functional description..... | 197 |

Chapter 15 System Mode Controller (SMC)

| | | |
|------|------------------------------------|-----|
| 15.1 | Chip-specific SMC information..... | 199 |
| 15.2 | Introduction..... | 199 |

| Section number | Title | Page |
|-----------------------|--|-------------|
| 15.3 | Modes of operation..... | 199 |
| 15.4 | Memory map and register descriptions..... | 201 |
| 15.4.1 | Power Mode Protection register (SMC_PMPROT)..... | 202 |
| 15.4.2 | Power Mode Control register (SMC_PMCTRL)..... | 203 |
| 15.4.3 | Stop Control Register (SMC_STOPCTRL)..... | 204 |
| 15.4.4 | Power Mode Status register (SMC_PMSTAT)..... | 206 |
| 15.5 | Functional description..... | 206 |
| 15.5.1 | Power mode transitions..... | 206 |
| 15.5.2 | Power mode entry/exit sequencing..... | 209 |
| 15.5.3 | Run modes..... | 211 |
| 15.5.4 | Wait modes..... | 212 |
| 15.5.5 | Stop modes..... | 213 |
| 15.5.6 | Debug in low power modes..... | 215 |

Chapter 16 Power Management Controller (PMC)

| | | |
|--------|--|-----|
| 16.1 | Introduction..... | 217 |
| 16.2 | Features..... | 217 |
| 16.3 | Low-voltage detect (LVD) system..... | 217 |
| 16.3.1 | LVD reset operation..... | 218 |
| 16.3.2 | LVD interrupt operation..... | 218 |
| 16.3.3 | Low-voltage warning (LVW) interrupt operation..... | 218 |
| 16.4 | I/O retention..... | 219 |
| 16.5 | Memory map and register descriptions..... | 219 |
| 16.5.1 | Low Voltage Detect Status And Control 1 register (PMC_LVDSC1)..... | 220 |
| 16.5.2 | Low Voltage Detect Status And Control 2 register (PMC_LVDSC2)..... | 221 |
| 16.5.3 | Regulator Status And Control register (PMC_REGSC)..... | 222 |

Chapter 17 Miscellaneous Control Module (MCM)

| | | |
|------|-------------------|-----|
| 17.1 | Introduction..... | 225 |
|------|-------------------|-----|

| Section number | Title | Page |
|-----------------------|--|-------------|
| 17.1.1 | Features..... | 225 |
| 17.2 | Memory map/register descriptions..... | 225 |
| 17.2.1 | Crossbar Switch (AXBS) Slave Configuration (MCM_PLASC)..... | 226 |
| 17.2.2 | Crossbar Switch (AXBS) Master Configuration (MCM_PLAMC)..... | 226 |
| 17.2.3 | Platform Control Register (MCM_PLACR)..... | 227 |
| 17.2.4 | Compute Operation Control Register (MCM_CPO)..... | 229 |

Chapter 18 **Crossbar Switch Lite (AXBS-Lite)**

| | | |
|--------|---------------------------------------|-----|
| 18.1 | Introduction..... | 231 |
| 18.1.1 | Features..... | 231 |
| 18.2 | Memory Map / Register Definition..... | 232 |
| 18.3 | Functional Description..... | 232 |
| 18.3.1 | General operation..... | 232 |

Chapter 19 **Low-Leakage Wakeup Unit (LLWU)**

| | | |
|--------|--|-----|
| 19.1 | Chip-specific LLWU information..... | 235 |
| 19.2 | Introduction..... | 235 |
| 19.2.1 | Features..... | 235 |
| 19.2.2 | Modes of operation..... | 236 |
| 19.2.3 | Block diagram..... | 237 |
| 19.3 | LLWU signal descriptions..... | 238 |
| 19.4 | Memory map/register definition..... | 238 |
| 19.4.1 | LLWU Pin Enable 1 register (LLWU_PE1)..... | 239 |
| 19.4.2 | LLWU Pin Enable 2 register (LLWU_PE2)..... | 240 |
| 19.4.3 | LLWU Module Enable register (LLWU_ME)..... | 241 |
| 19.4.4 | LLWU Flag 1 register (LLWU_F1)..... | 243 |
| 19.4.5 | LLWU Flag 3 register (LLWU_F3)..... | 244 |
| 19.4.6 | LLWU Pin Filter 1 register (LLWU_FILTER1)..... | 246 |
| 19.4.7 | LLWU Pin Filter 2 register (LLWU_FILTER2)..... | 247 |

| Section number | Title | Page |
|---|--|-------------|
| 19.5 | Functional description..... | 248 |
| 19.5.1 | VLLS modes..... | 249 |
| 19.5.2 | Initialization..... | 249 |
| Chapter 20 Peripheral Bridge (AIPS-Lite) | | |
| 20.1 | Introduction..... | 251 |
| 20.1.1 | Features..... | 251 |
| 20.1.2 | General operation..... | 251 |
| 20.2 | Functional description..... | 252 |
| 20.2.1 | Access support..... | 252 |
| Chapter 21 Reset Control Module (RCM) | | |
| 21.1 | Introduction..... | 253 |
| 21.2 | Reset memory map and register descriptions..... | 253 |
| 21.2.1 | System Reset Status Register 0 (RCM_SRS0)..... | 254 |
| 21.2.2 | System Reset Status Register 1 (RCM_SRS1)..... | 255 |
| 21.2.3 | Reset Pin Filter Control register (RCM_RPFC)..... | 256 |
| 21.2.4 | Reset Pin Filter Width register (RCM_RPFW)..... | 257 |
| 21.2.5 | Force Mode Register (RCM_FM)..... | 259 |
| 21.2.6 | Mode Register (RCM_MR)..... | 259 |
| 21.2.7 | Sticky System Reset Status Register 0 (RCM_SSRS0)..... | 260 |
| 21.2.8 | Sticky System Reset Status Register 1 (RCM_SSRS1)..... | 261 |
| Chapter 22 Bit Manipulation Engine (BME) | | |
| 22.1 | Introduction..... | 263 |
| 22.1.1 | Overview..... | 264 |
| 22.1.2 | Features..... | 264 |
| 22.1.3 | Modes of operation..... | 265 |
| 22.2 | Memory map and register definition..... | 265 |
| 22.3 | Functional description..... | 265 |

| Section number | Title | Page |
|-----------------------|--|-------------|
| 22.3.1 | BME decorated stores..... | 266 |
| 22.3.2 | BME decorated loads..... | 273 |
| 22.3.3 | Additional details on decorated addresses and GPIO accesses..... | 279 |
| 22.4 | Application information..... | 280 |

Chapter 23 Micro Trace Buffer (MTB)

| | | |
|--------|---|-----|
| 23.1 | Introduction..... | 283 |
| 23.1.1 | Overview..... | 283 |
| 23.1.2 | Features..... | 286 |
| 23.1.3 | Modes of operation..... | 287 |
| 23.2 | External signal description..... | 287 |
| 23.3 | Memory map and register definition..... | 288 |
| 23.3.1 | MTB_RAM Memory Map..... | 288 |
| 23.3.2 | MTB_DWT Memory Map..... | 300 |
| 23.3.3 | System ROM Memory Map..... | 310 |

Chapter 24 Multipurpose Clock Generator Lite (MCG_Lite)

| | | |
|--------|--|-----|
| 24.1 | Introduction | 315 |
| 24.1.1 | Features | 315 |
| 24.1.2 | Block diagram | 315 |
| 24.2 | Memory map and register definition..... | 316 |
| 24.2.1 | MCG Control Register 1 (MCG_C1)..... | 317 |
| 24.2.2 | MCG Control Register 2 (MCG_C2)..... | 318 |
| 24.2.3 | MCG Status Register (MCG_S)..... | 318 |
| 24.2.4 | MCG Status and Control Register (MCG_SC)..... | 319 |
| 24.2.5 | MCG Miscellaneous Control Register (MCG_MC)..... | 320 |
| 24.3 | Functional description..... | 320 |
| 24.3.1 | Clock mode switching | 320 |
| 24.3.2 | LIRC divider 1 | 322 |

| Section number | Title | Page |
|----------------|----------------------------------|------|
| 24.3.3 | LIRC divider 2 | 322 |
| 24.3.4 | Enable LIRC in Stop mode | 322 |
| 24.3.5 | MCG-Lite in Low-power mode | 322 |

Chapter 25 Oscillator (OSC)

| | | |
|--------|---|-----|
| 25.1 | OSC modes of operation with MCG_Lite and RTC..... | 323 |
| 25.2 | Introduction..... | 323 |
| 25.3 | Features and Modes..... | 323 |
| 25.4 | Block Diagram..... | 324 |
| 25.5 | OSC Signal Descriptions..... | 325 |
| 25.6 | External Crystal / Resonator Connections..... | 325 |
| 25.7 | External Clock Connections..... | 326 |
| 25.8 | Memory Map/Register Definitions..... | 327 |
| 25.8.1 | OSC Memory Map/Register Definition..... | 327 |
| 25.9 | Functional Description..... | 328 |
| 25.9.1 | OSC module states..... | 328 |
| 25.9.2 | OSC module modes..... | 330 |
| 25.9.3 | Counter..... | 331 |
| 25.9.4 | Reference clock pin requirements..... | 331 |
| 25.10 | Reset..... | 332 |
| 25.11 | Low power modes operation..... | 332 |
| 25.12 | Interrupts..... | 332 |

Chapter 26 Flash Memory Controller (FMC)

| | | |
|--------|----------------------------------|-----|
| 26.1 | Introduction..... | 333 |
| 26.1.1 | Overview..... | 333 |
| 26.1.2 | Features..... | 333 |
| 26.2 | Modes of operation..... | 334 |
| 26.3 | External signal description..... | 334 |

| Section number | Title | Page |
|---|--|-------------|
| 26.4 | Memory map and register descriptions..... | 334 |
| 26.5 | Functional description..... | 334 |
| Chapter 27 Flash Memory Module (FTFA) | | |
| 27.1 | Introduction..... | 337 |
| 27.1.1 | Features..... | 337 |
| 27.1.2 | Block Diagram..... | 338 |
| 27.1.3 | Glossary..... | 339 |
| 27.2 | External Signal Description..... | 340 |
| 27.3 | Memory Map and Registers..... | 340 |
| 27.3.1 | Flash Configuration Field Description..... | 340 |
| 27.3.2 | Program Flash IFR Map..... | 341 |
| 27.3.3 | Register Descriptions..... | 341 |
| 27.4 | Functional Description..... | 350 |
| 27.4.1 | Flash Protection..... | 350 |
| 27.4.2 | Interrupts..... | 351 |
| 27.4.3 | Flash Operation in Low-Power Modes..... | 352 |
| 27.4.4 | Functional Modes of Operation..... | 352 |
| 27.4.5 | Flash Reads and Ignored Writes..... | 352 |
| 27.4.6 | Read While Write (RWW)..... | 353 |
| 27.4.7 | Flash Program and Erase..... | 353 |
| 27.4.8 | Flash Command Operations..... | 353 |
| 27.4.9 | Margin Read Commands..... | 358 |
| 27.4.10 | Flash Command Description..... | 359 |
| 27.4.11 | Security..... | 373 |
| 27.4.12 | Reset Sequence..... | 375 |
| Chapter 28 Analog-to-Digital Converter (ADC) | | |
| 28.1 | Chip-specific ADC information..... | 377 |

| Section number | Title | Page |
|-----------------------|--|-------------|
| 28.1.1 | ADC0 connections/channel assignment..... | 377 |
| 28.1.2 | ADC analog supply and reference connections..... | 378 |
| 28.1.3 | ADC Reference Options..... | 378 |
| 28.1.4 | Alternate clock..... | 379 |
| 28.2 | Introduction..... | 379 |
| 28.2.1 | Features..... | 379 |
| 28.2.2 | Block diagram..... | 380 |
| 28.3 | ADC signal descriptions..... | 381 |
| 28.3.1 | Analog Power (VDDA)..... | 382 |
| 28.3.2 | Analog Ground (VSSA)..... | 382 |
| 28.3.3 | Analog Channel Inputs (ADx)..... | 382 |
| 28.4 | Memory map and register definitions..... | 382 |
| 28.4.1 | ADC Status and Control Registers 1 (ADC _x _SC1n)..... | 383 |
| 28.4.2 | ADC Configuration Register 1 (ADC _x _CFG1)..... | 387 |
| 28.4.3 | ADC Configuration Register 2 (ADC _x _CFG2)..... | 388 |
| 28.4.4 | ADC Data Result Register (ADC _x _Rn)..... | 389 |
| 28.4.5 | Compare Value Registers (ADC _x _CVn)..... | 390 |
| 28.4.6 | Status and Control Register 2 (ADC _x _SC2)..... | 391 |
| 28.4.7 | Status and Control Register 3 (ADC _x _SC3)..... | 393 |
| 28.4.8 | ADC Offset Correction Register (ADC _x _OFS)..... | 395 |
| 28.4.9 | ADC Plus-Side Gain Register (ADC _x _PG)..... | 395 |
| 28.4.10 | ADC Plus-Side General Calibration Value Register (ADC _x _CLPD)..... | 396 |
| 28.4.11 | ADC Plus-Side General Calibration Value Register (ADC _x _CLPS)..... | 396 |
| 28.4.12 | ADC Plus-Side General Calibration Value Register (ADC _x _CLP4)..... | 397 |
| 28.4.13 | ADC Plus-Side General Calibration Value Register (ADC _x _CLP3)..... | 397 |
| 28.4.14 | ADC Plus-Side General Calibration Value Register (ADC _x _CLP2)..... | 398 |
| 28.4.15 | ADC Plus-Side General Calibration Value Register (ADC _x _CLP1)..... | 398 |
| 28.4.16 | ADC Plus-Side General Calibration Value Register (ADC _x _CLP0)..... | 399 |
| 28.5 | Functional description..... | 399 |

| Section number | Title | Page |
|-----------------------|---|-------------|
| 28.5.1 | Clock select and divide control..... | 400 |
| 28.5.2 | Voltage reference selection..... | 401 |
| 28.5.3 | Hardware trigger and channel selects..... | 401 |
| 28.5.4 | Conversion control..... | 402 |
| 28.5.5 | Automatic compare function..... | 409 |
| 28.5.6 | Calibration function..... | 410 |
| 28.5.7 | User-defined offset function..... | 412 |
| 28.5.8 | Temperature sensor..... | 413 |
| 28.5.9 | MCU wait mode operation..... | 414 |
| 28.5.10 | MCU Normal Stop mode operation..... | 414 |
| 28.5.11 | MCU Low-Power Stop mode operation..... | 415 |
| 28.6 | Initialization information..... | 416 |
| 28.6.1 | ADC module initialization example..... | 416 |
| 28.7 | Application information..... | 418 |
| 28.7.1 | External pins and routing..... | 418 |
| 28.7.2 | Sources of error..... | 420 |

Chapter 29 Comparator (CMP)

| | | |
|--------|--------------------------------------|-----|
| 29.1 | Chip-specific CMP information..... | 425 |
| 29.1.1 | CMP input connections..... | 425 |
| 29.1.2 | CMP external references..... | 426 |
| 29.1.3 | CMP trigger mode..... | 426 |
| 29.2 | Introduction..... | 427 |
| 29.2.1 | CMP features..... | 427 |
| 29.2.2 | 6-bit DAC key features..... | 428 |
| 29.2.3 | ANMUX key features..... | 428 |
| 29.2.4 | CMP, DAC and ANMUX diagram..... | 428 |
| 29.2.5 | CMP block diagram..... | 429 |
| 29.3 | Memory map/register definitions..... | 430 |

| Section number | Title | Page |
|-----------------------|--|-------------|
| 29.3.1 | CMP Control Register 0 (CMP _x _CR0)..... | 431 |
| 29.3.2 | CMP Control Register 1 (CMP _x _CR1)..... | 432 |
| 29.3.3 | CMP Filter Period Register (CMP _x _FPR)..... | 433 |
| 29.3.4 | CMP Status and Control Register (CMP _x _SCR)..... | 434 |
| 29.3.5 | DAC Control Register (CMP _x _DACCR)..... | 435 |
| 29.3.6 | MUX Control Register (CMP _x _MUXCR)..... | 435 |
| 29.4 | Functional description..... | 436 |
| 29.4.1 | CMP functional modes..... | 436 |
| 29.4.2 | Power modes..... | 440 |
| 29.4.3 | Startup and operation..... | 441 |
| 29.4.4 | Low-pass filter..... | 441 |
| 29.5 | CMP interrupts..... | 443 |
| 29.6 | Digital-to-analog converter..... | 444 |
| 29.7 | DAC functional description..... | 444 |
| 29.7.1 | Voltage reference source select..... | 444 |
| 29.8 | DAC resets..... | 445 |
| 29.9 | DAC clocks..... | 445 |
| 29.10 | DAC interrupts..... | 445 |
| 29.11 | CMP Trigger Mode..... | 445 |

Chapter 30 Voltage Reference (VREF)

| | | |
|--------|---|-----|
| 30.1 | Chip specific VREF information..... | 447 |
| 30.1.1 | Clock Gating..... | 447 |
| 30.2 | Introduction..... | 447 |
| 30.2.1 | Overview..... | 448 |
| 30.2.2 | Features..... | 448 |
| 30.2.3 | Modes of Operation..... | 449 |
| 30.2.4 | VREF Signal Descriptions..... | 449 |
| 30.3 | Memory Map and Register Definition..... | 450 |

| Section number | Title | Page |
|-----------------------|---|-------------|
| 30.3.1 | VREF Trim Register (VREF_TRM)..... | 450 |
| 30.3.2 | VREF Status and Control Register (VREF_SC)..... | 451 |
| 30.4 | Functional Description..... | 452 |
| 30.4.1 | Voltage Reference Disabled, SC[VREFEN] = 0..... | 453 |
| 30.4.2 | Voltage Reference Enabled, SC[VREFEN] = 1..... | 453 |
| 30.4.3 | Internal voltage regulator..... | 454 |
| 30.5 | Initialization/Application Information..... | 455 |

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

| | | |
|--------|---|-----|
| 31.1 | Chip-specific TPM information..... | 457 |
| 31.1.1 | Clock options..... | 457 |
| 31.1.2 | Trigger options..... | 458 |
| 31.1.3 | Global timebase..... | 458 |
| 31.1.4 | TPM interrupts..... | 458 |
| 31.2 | Introduction..... | 459 |
| 31.2.1 | TPM Philosophy..... | 459 |
| 31.2.2 | Features..... | 459 |
| 31.2.3 | Modes of operation..... | 460 |
| 31.2.4 | Block diagram..... | 460 |
| 31.3 | TPM Signal Descriptions..... | 461 |
| 31.3.1 | TPM_EXTCLK — TPM External Clock..... | 461 |
| 31.3.2 | TPM_CHn — TPM Channel (n) I/O Pin..... | 462 |
| 31.4 | Memory Map and Register Definition..... | 462 |
| 31.4.1 | Status and Control (TPMx_SC)..... | 463 |
| 31.4.2 | Counter (TPMx_CNT)..... | 464 |
| 31.4.3 | Modulo (TPMx_MOD)..... | 465 |
| 31.4.4 | Channel (n) Status and Control (TPMx_CnSC)..... | 466 |
| 31.4.5 | Channel (n) Value (TPMx_CnV)..... | 467 |
| 31.4.6 | Capture and Compare Status (TPMx_STATUS)..... | 468 |

| Section number | Title | Page |
|-----------------------|---|-------------|
| 31.4.7 | Configuration (TPMx_CONF)..... | 470 |
| 31.5 | Functional description..... | 471 |
| 31.5.1 | Clock domains..... | 472 |
| 31.5.2 | Prescaler..... | 472 |
| 31.5.3 | Counter..... | 473 |
| 31.5.4 | Input Capture Mode..... | 476 |
| 31.5.5 | Output Compare Mode..... | 476 |
| 31.5.6 | Edge-Aligned PWM (EPWM) Mode..... | 478 |
| 31.5.7 | Center-Aligned PWM (CPWM) Mode..... | 479 |
| 31.5.8 | Registers Updated from Write Buffers..... | 481 |
| 31.5.9 | Output triggers..... | 482 |
| 31.5.10 | Reset Overview..... | 482 |
| 31.5.11 | TPM Interrupts..... | 482 |

Chapter 32 Low-Power Timer (LPTMR)

| | | |
|--------|--|-----|
| 32.1 | Chip-specific LPTMR information..... | 485 |
| 32.1.1 | LPTMR pulse counter input options..... | 485 |
| 32.1.2 | LPTMR prescaler/glitch filter clocking options..... | 485 |
| 32.2 | Introduction..... | 486 |
| 32.2.1 | Features..... | 486 |
| 32.2.2 | Modes of operation..... | 486 |
| 32.3 | LPTMR signal descriptions..... | 487 |
| 32.3.1 | Detailed signal descriptions..... | 487 |
| 32.4 | Memory map and register definition..... | 487 |
| 32.4.1 | Low Power Timer Control Status Register (LPTMR _x _CSR)..... | 488 |
| 32.4.2 | Low Power Timer Prescale Register (LPTMR _x _PSR)..... | 489 |
| 32.4.3 | Low Power Timer Compare Register (LPTMR _x _CMR)..... | 491 |
| 32.4.4 | Low Power Timer Counter Register (LPTMR _x _CNR)..... | 491 |
| 32.5 | Functional description..... | 492 |

| Section number | Title | Page |
|---|---|-------------|
| 32.5.1 | LPTMR power and reset..... | 492 |
| 32.5.2 | LPTMR clocking..... | 492 |
| 32.5.3 | LPTMR prescaler/glitch filter..... | 492 |
| 32.5.4 | LPTMR compare..... | 494 |
| 32.5.5 | LPTMR counter..... | 494 |
| 32.5.6 | LPTMR hardware trigger..... | 495 |
| 32.5.7 | LPTMR interrupt..... | 495 |
| Chapter 33 Real Time Clock (RTC) | | |
| 33.1 | Chip-specific RTC information..... | 497 |
| 33.1.1 | RTC_CLKOUT options..... | 497 |
| 33.2 | Introduction..... | 497 |
| 33.2.1 | Features..... | 497 |
| 33.2.2 | Modes of operation..... | 498 |
| 33.2.3 | RTC signal descriptions..... | 498 |
| 33.3 | Register definition..... | 498 |
| 33.3.1 | RTC Time Seconds Register (RTC_TSR)..... | 499 |
| 33.3.2 | RTC Time Prescaler Register (RTC_TPR)..... | 499 |
| 33.3.3 | RTC Time Alarm Register (RTC_TAR)..... | 500 |
| 33.3.4 | RTC Time Compensation Register (RTC_TCR)..... | 500 |
| 33.3.5 | RTC Control Register (RTC_CR)..... | 502 |
| 33.3.6 | RTC Status Register (RTC_SR)..... | 504 |
| 33.3.7 | RTC Lock Register (RTC_LR)..... | 505 |
| 33.3.8 | RTC Interrupt Enable Register (RTC_IER)..... | 506 |
| 33.4 | Functional description..... | 507 |
| 33.4.1 | Power, clocking, and reset..... | 507 |
| 33.4.2 | Time counter..... | 508 |
| 33.4.3 | Compensation..... | 508 |
| 33.4.4 | Time alarm..... | 509 |

| Section number | Title | Page |
|----------------|--------------------|------|
| 33.4.5 | Update mode..... | 509 |
| 33.4.6 | Register lock..... | 510 |
| 33.4.7 | Interrupt..... | 510 |

Chapter 34 Serial Peripheral Interface (SPI)

| | | |
|--------|--|-----|
| 34.1 | Chip-specific SPI information..... | 511 |
| 34.2 | Introduction..... | 511 |
| 34.2.1 | Features..... | 511 |
| 34.2.2 | Modes of operation..... | 512 |
| 34.2.3 | Block diagrams..... | 513 |
| 34.3 | External signal description..... | 515 |
| 34.3.1 | SPSCK — SPI Serial Clock..... | 515 |
| 34.3.2 | MOSI — Master Data Out, Slave Data In..... | 516 |
| 34.3.3 | MISO — Master Data In, Slave Data Out..... | 516 |
| 34.3.4 | SS — Slave Select..... | 516 |
| 34.4 | Memory map/register definition..... | 517 |
| 34.4.1 | SPI Status Register (SPIx_S)..... | 517 |
| 34.4.2 | SPI Baud Rate Register (SPIx_BR)..... | 518 |
| 34.4.3 | SPI Control Register 2 (SPIx_C2)..... | 519 |
| 34.4.4 | SPI Control Register 1 (SPIx_C1)..... | 521 |
| 34.4.5 | SPI Match Register (SPIx_M)..... | 522 |
| 34.4.6 | SPI Data Register (SPIx_D)..... | 523 |
| 34.5 | Functional description..... | 523 |
| 34.5.1 | General..... | 523 |
| 34.5.2 | Master mode..... | 524 |
| 34.5.3 | Slave mode..... | 525 |
| 34.5.4 | SPI clock formats..... | 527 |
| 34.5.5 | SPI baud rate generation..... | 530 |
| 34.5.6 | Special features..... | 530 |

| Section number | Title | Page |
|-----------------------|---|-------------|
| 34.5.7 | Error conditions..... | 532 |
| 34.5.8 | Low-power mode options..... | 533 |
| 34.5.9 | Reset..... | 534 |
| 34.5.10 | Interrupts..... | 535 |
| 34.6 | Initialization/application information..... | 536 |
| 34.6.1 | Initialization sequence..... | 536 |
| 34.6.2 | Pseudo-Code Example..... | 537 |

Chapter 35 Inter-Integrated Circuit (I2C)

| | | |
|---------|---|-----|
| 35.1 | Chip-specific I2C information..... | 541 |
| 35.2 | Introduction..... | 541 |
| 35.2.1 | Features..... | 541 |
| 35.2.2 | Modes of operation..... | 542 |
| 35.2.3 | Block diagram..... | 542 |
| 35.3 | I2C signal descriptions..... | 543 |
| 35.4 | Memory map/register definition..... | 544 |
| 35.4.1 | I2C Address Register 1 (I2Cx_A1)..... | 544 |
| 35.4.2 | I2C Frequency Divider register (I2Cx_F)..... | 545 |
| 35.4.3 | I2C Control Register 1 (I2Cx_C1)..... | 546 |
| 35.4.4 | I2C Status register (I2Cx_S)..... | 547 |
| 35.4.5 | I2C Data I/O register (I2Cx_D)..... | 549 |
| 35.4.6 | I2C Control Register 2 (I2Cx_C2)..... | 550 |
| 35.4.7 | I2C Programmable Input Glitch Filter Register (I2Cx_FLT)..... | 551 |
| 35.4.8 | I2C Range Address register (I2Cx_RA)..... | 552 |
| 35.4.9 | I2C SMBus Control and Status register (I2Cx_SMB)..... | 553 |
| 35.4.10 | I2C Address Register 2 (I2Cx_A2)..... | 554 |
| 35.4.11 | I2C SCL Low Timeout Register High (I2Cx_SLTH)..... | 555 |
| 35.4.12 | I2C SCL Low Timeout Register Low (I2Cx_SLTL)..... | 555 |
| 35.4.13 | I2C Status register 2 (I2Cx_S2)..... | 556 |

| Section number | Title | Page |
|-----------------------|---|-------------|
| 35.5 | Functional description..... | 556 |
| 35.5.1 | I2C protocol..... | 556 |
| 35.5.2 | 10-bit address..... | 562 |
| 35.5.3 | Address matching..... | 563 |
| 35.5.4 | System management bus specification..... | 564 |
| 35.5.5 | Resets..... | 567 |
| 35.5.6 | Interrupts..... | 567 |
| 35.5.7 | Programmable input glitch filter..... | 569 |
| 35.5.8 | Address matching wake-up..... | 570 |
| 35.5.9 | Double buffering mode..... | 570 |
| 35.6 | Initialization/application information..... | 571 |

Chapter 36 **Low Power Universal Asynchronous Receiver/Transmitter (LPUART)**

| | | |
|--------|---|-----|
| 36.1 | Chip-specific LPUART information..... | 575 |
| 36.2 | Introduction..... | 575 |
| 36.2.1 | Features..... | 575 |
| 36.2.2 | Modes of operation..... | 576 |
| 36.2.3 | Signal Descriptions..... | 577 |
| 36.2.4 | Block diagram..... | 577 |
| 36.3 | Register definition..... | 579 |
| 36.3.1 | LPUART Baud Rate Register (LPUART _x _BAUD)..... | 580 |
| 36.3.2 | LPUART Status Register (LPUART _x _STAT)..... | 582 |
| 36.3.3 | LPUART Control Register (LPUART _x _CTRL)..... | 586 |
| 36.3.4 | LPUART Data Register (LPUART _x _DATA)..... | 591 |
| 36.3.5 | LPUART Match Address Register (LPUART _x _MATCH)..... | 593 |
| 36.4 | Functional description..... | 593 |
| 36.4.1 | Baud rate generation..... | 593 |
| 36.4.2 | Transmitter functional description..... | 594 |
| 36.4.3 | Receiver functional description..... | 596 |

| Section number | Title | Page |
|----------------|----------------------------------|------|
| 36.4.4 | Additional LPUART functions..... | 601 |
| 36.4.5 | Interrupts and status flags..... | 603 |

Chapter 1

About This Document

1.1 Overview

1.1.1 Purpose

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

1.1.2 Audience

A reference manual is primarily for system architects and software application developers who are using or considering using a NXP product 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"> • A subset of a register's named field 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. • A continuous range of individual signals of a bus For example, XAD[7:0] refers to signals 7–0 of the XAD bus. |

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"> • An active-high signal is asserted when high (1). • An active-low signal is asserted when low (0). |
| deasserted | Refers to the state of a signal as follows: <ul style="list-style-type: none"> • An active-high signal is deasserted when low (0). • An active-low signal is deasserted when high (1). 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 KL03 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 features of the KL03 family derivatives are as follows.

- Core platform clock up to 48 MHz, bus clock up to 24 MHz
- Memory option is up to 32 KB flash, 2 KB RAM and 8 KB ROM with build-in boot loader
- Wide operating voltage ranges from 1.71–3.6 V with fully functional flash program/erase/read operations
- Multiple package options from 16-pin to 24-pin
- Ambient operating temperature ranges from –40 °C to 85 °C for WLCSP package and –40 °C to 105 °C for all the other packages.

The family acts as an ultra low-power, cost-effective microcontroller to provide developers an appropriate entry-level 32-bit solution. The family is the 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.

2.2 Module functional categories

The modules on this device are grouped into functional categories. Information found here describes 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">• 32-bit MCU core from ARM's Cortex-M class, 1.77 CoreMark®/MHz from single-cycle access memories, 48 MHz CPU frequency |
| System | <ul style="list-style-type: none">• System integration module• Power management and mode controllers |

Table continues on the next page...

Table 2-1. Module functional categories (continued)

| Module category | Description |
|--------------------------------|---|
| | <ul style="list-style-type: none"> • Multiple power modes available based on run, wait, stop, and power-down modes • Miscellaneous control module • Crossbar switch lite • Low-leakage wakeup unit • Peripheral bridge • COP watchdog |
| Memories | <ul style="list-style-type: none"> • Internal memories include: <ul style="list-style-type: none"> • Up to 32 KB flash memory • 2 KB SRAM • Up to 8 KB ROM • Up to 16 byte system register file |
| Clocks | <ul style="list-style-type: none"> • Multiple clock generation options available from internally- and externally-generated clocks <ul style="list-style-type: none"> • Low power 1 kHz RC oscillator for RTC and COP watchdog • System oscillator to provide clock source for the MCU |
| Security | <ul style="list-style-type: none"> • COP watchdog timer (COP) |
| Analog | <ul style="list-style-type: none"> • 12-bit analog-to-digital converters • Comparator (CMP) with internal 6-bit digital-to-analog converter (DAC) • Voltage reference (VREF) • High accuracy 1.2 V voltage reference to provide a stable reference for ADC |
| Timers | <ul style="list-style-type: none"> • Two 2-channel TPMs • Real time clock (RTC) • Low-power timer (LPTMR) • System tick timer |
| Communications | <ul style="list-style-type: none"> • One 8-bit serial peripheral interface (SPI) • One inter-integrated circuit (I^2C) module • One low power UART module (LPUART) |
| Human-Machine Interfaces (HMI) | <ul style="list-style-type: none"> • General purpose input/output controller(GPIO) |

2.2.1 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-vector interrupt controller (NVIC) implement a relocatable vector table supporting many external interrupts, a single non-maskable interrupt (NMI), and priority levels. |

Table continues on the next page...

Table 2-2. Core modules (continued)

| Module | Description |
|-----------------------|--|
| | 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"> Serial Wire Debug (SWD) |

2.2.2 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 (SMC) | 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. |
| Crossbar switch lite (AXBS-Lite) | The AXBS 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 (VLLS) through various internal peripheral and external pin sources. |
| Peripheral bridge (AIPS-Lite) | The peripheral bridge converts the crossbar switch interface to an interface to access a majority of peripherals on the device. |
| Computer operating properly watchdog (COP) | The COP monitors internal system operation and forces a reset in case of failure. It can run from an independent 1 kHz low power oscillator, 8/2 MHz internal oscillator, or external crystal oscillator with a programmable refresh window to detect deviations in program flow or system frequency. |

2.2.3 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 32 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 | 2 KB internal system RAM. |
| ROM | 8 KB ROM. |
| System register file | 16-byte register file. |

2.2.4 Clocks

The following clock modules are available on this device.

Table 2-5. Clock modules

| Module | Description |
|--|---|
| Multipurpose Clock Generator Lite (MCG-Lite) | MCG Lite module containing a 48 MHz and an 8 or 2 MHz internal reference clock source. |
| System oscillator (OSC) | The system oscillator, in conjunction with an external crystal or resonator, generates a reference clock for the MCU. |

2.2.5 Analog modules

The following analog modules are available on this device:

Table 2-6. Analog modules

| Module | Description |
|--|--|
| Analog-to-digital converters (ADC) | 12-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. |
| Voltage reference (VREF) | Supplies an accurate voltage output that is trimmable in 0.5 mV steps. The VREF can be used in medical applications, such as glucose meters, to provide a |

Table 2-6. Analog modules

| Module | Description |
|--------|--|
| | reference voltage to biosensors or as a reference to analog peripherals, such as the ADC, DAC, or CMP. |

2.2.6 Timer modules

The following timer modules are available on this device:

Table 2-7. Timer modules

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

2.2.7 Communication interfaces

The following communication interfaces are available on this device:

Table 2-8. Communication modules

| Module | Description |
|-----------------------------------|---|
| 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. |

Table continues on the next page...

Table 2-8. Communication modules (continued)

| Module | Description |
|---|--|
| Low power universal asynchronous receiver/transmitters (LPUART) | One low power UART module that retains functional in stop modes. |

2.2.8 Human-machine interfaces

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

Table 2-9. HMI modules

| Module | Description |
|-------------------------------------|---|
| General purpose input/output (GPIO) | Some general purpose input or output (GPIO) pins are capable of interrupt request generation. |

2.3 Module to module interconnects

2.3.1 Interconnection overview

The following table lists the module to module interconnections for this device.

Table 2-10. Module-to-module interconnects

| Peripheral | Signal | — | to Peripheral | Use Case | Control | Comment |
|------------|------------------|----|---------------|--------------------------|---|---|
| TPM1 | CH0F, CH1F | to | ADC (Trigger) | ADC Triggering (A AND B) | SIM_SOPT7[ADC0ALTTRGEN] = 0 | Ch0 is A, and Ch1 is B, selecting this ADC trigger is for supporting A and B triggering. In Stop and VLPS modes, the second trigger must be set to >10 µs after the first trigger |
| LPTMR | Hardware trigger | to | ADC (Trigger) | ADC Triggering (A or B) | SIM_SOPT7[ADC0TRGSEL] and SIM_SOPT7[ADC0PRETRGSEL] to select A or B | — |
| TPMx | TOF | to | ADC (Trigger) | ADC Triggering (A or B) | SIM_SOPT7[ADC0TRGSEL], SIM_SOPT7[ADC0PRETRGSEL] to select A or B | — |

Table continues on the next page...

Table 2-10. Module-to-module interconnects (continued)

| Peripheral | Signal | — | to Peripheral | Use Case | Control | Comment |
|------------|------------------|----|-----------------------|--|--|---------------|
| RTC | ALARM or SECONDS | to | ADC (Trigger) | ADC Triggering (A or B) | SIM_SOPT7[ADC0TRGSEL], SIM_SOPT7[ADC0PRETRGSEL] to select A or B | — |
| EXTRG_IN | EXTRG_IN | to | ADC (Trigger) | ADC Triggering (A or B) | SIM_SOPT7[ADC0TRGSEL], SIM_SOPT7[ADC0PRETRGSEL] to select A or B | — |
| CMP0 | CMP0_OUT | to | ADC (Trigger) | ADC Triggering (Aor B) | SIM_SOPT7[ADC0TRGSEL], SIM_SOPT7[ADC0PRETRGSEL] to select A or B | — |
| CMP0 | CMP0_OUT | to | LPTMR_ALT0 | Count CMP events | LPTMR_CSR[TPS] | — |
| CMP0 | CMP0_OUT | to | TPM1 CH0 | Input capture | SIM_SOPT4[TPM1CH0SRC] | — |
| CMP0 | CMP0_OUT | to | LPUART0_RX | IR interface | SIM_SOPT5[LPUART0RXSRC] | Uses for IrDA |
| LPTMR | Hardware trigger | to | CMPx | Low power triggering of the comparator | CMP_CR1[TRIGM] | — |
| LPTMR | Hardware trigger | to | TPMx | TPM Trigger input | TPMx_CONF[TRGSEL] (4-bit field) | — |
| TPMx | TOF | to | TPMx | TPM Trigger input | TPMx_CONF[TRGSEL] (4-bit field) | — |
| TPM1 | Timebase | to | TPMx | TPM Global timebase input | TPMx_CONF[GTBEEN] | — |
| RTC | ALARM or SECONDS | to | TPMx | TPM Trigger input | TPMx_CONF[TRGSEL] (4-bit field) | — |
| EXTRG_IN | EXTRG_IN | to | TPMx | TPM Trigger input | TPMx_CONF[TRGSEL] (4-bit field) | — |
| CMP0 | CMP0_OUT | to | TPMx | TPM Trigger input | TPMx_CONF[TRGSEL] (4-bit field) | — |
| LPUART0 | LPUART0_TX | to | Modulated by TPM1 CH0 | LPUART modulation | SIM_SOPT5[LPUART0TXSRC] | Uses for IrDA |

2.3.2 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 2-11. Analog reference options

| Module | Reference option | Comment/ Reference selection |
|----------------|------------------|------------------------------|
| 12-bit SAR ADC | 1 - VDD | Selected by ADCx_SC2[REFSEL] |

Table continues on the next page...

Table 2-11. Analog reference options (continued)

| Module | Reference option | Comment/ Reference selection |
|--------------------|--|-------------------------------|
| | 2 - VREF_OUT 3 - Reserved | |
| CMP with 6-bit DAC | Vin1 - VREF_OUT Vin2 - VDD ¹ | Selected by CMPx_DACCR[VRSEL] |

1. Use this option for the best ADC operation.

2.4 Orderable part numbers

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

Table 2-12. Orderable part numbers summary

| Part number | CPU frequency | Pin count | Package | Total flash memory | RAM | Temperature range |
|-----------------|---------------|-----------|--------------------|--------------------|------|-------------------|
| MKL03Z8VFG4(R) | 48 MHz | 16 | QFN | 8 KB | 2 KB | -40 to 105 °C |
| MKL03Z16VFG4(R) | 48 MHz | 16 | QFN | 16 KB | 2 KB | -40 to 105 °C |
| MKL03Z32VFG4(R) | 48 MHz | 16 | QFN | 32 KB | 2 KB | -40 to 105 °C |
| MKL03Z8VFK4(R) | 48 MHz | 24 | QFN | 8 KB | 2 KB | --40 to 105 °C |
| MKL03Z16VFK4(R) | 48 MHz | 24 | QFN | 16 KB | 2 KB | -40 to 105 °C |
| MKL03Z32VFK4(R) | 48 MHz | 24 | QFN | 32 KB | 2 KB | -40 to 105 °C |
| MKL03Z32CAF4R | 48 MHz | 20 | WLCSP | 32 KB | 2 KB | -40 to 85 °C |
| MKL03Z32CBF4R | 48 MHz | 20 | WLCSP (ultra thin) | 32 KB | 2 KB | -40 to 85 °C |

Chapter 3

Core Overview

3.1 ARM Cortex-M0+ core introduction

The enhanced ARM Cortex M0+ is the member of the Cortex-M Series of processors targeting microcontroller cores focused on very cost sensitive, low power applications. It has a single 32-bit AMBA AHB-Lite interface and includes an NVIC component. It also has hardware debug functionality including support for simple program trace capability. The processor supports the ARMv6-M instruction set (Thumb) architecture including all but three 16-bit Thumb opcodes (52 total) plus seven 32-bit instructions. It is upward compatible with other Cortex-M profile processors.

3.1.1 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 memory and RAM
- Single 32-bit I/O port bus interfacing to the GPIO with 1-cycle loads and stores

3.1.2 System tick timer

The CLKSOURCE field 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 field in the SysTick Calibration Value Register is always 0.

3.1.3 Debug facilities

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

3.1.4 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 |

3.2 Nested vectored interrupt controller (NVIC)

3.2.1 Interrupt priority levels

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

| | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
|----|------|----|----|----|----|----|----|------|----|----|----|----|----|----|----|------|----|----|----|----|----|---|---|------|---|---|---|---|---|---|---|---|
| 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 | IRQ3 | 0 | 0 | 0 | 0 | 0 | 0 | IRQ2 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | IRQ1 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | IRQ0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| W | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |

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.

3.2.3 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.

NOTE

The NVIC wake-up sources in the following table support only down to VLPS.

Table 3-2. Interrupt vector assignments

| Address | Vector | IRQ ¹ | NVIC IPR register number ² | Source module | Source description |
|--|--------|------------------|---------------------------------------|-------------------|--|
| ARM core system handler vectors | | | | | |
| 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 (PendableSrvReq) |
| 0x0000_003C | 15 | — | — | ARM core | System tick timer (SysTick) |
| Non-Core Vectors | | | | | |
| 0x0000_0040 | 16 | 0 | 0 | — | — |
| 0x0000_0044 | 17 | 1 | 0 | — | — |
| 0x0000_0048 | 18 | 2 | 0 | — | — |
| 0x0000_004C | 19 | 3 | 0 | — | — |
| 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 ² C0 | Status and Timeout and wakeup flags |
| 0x0000_0064 | 25 | 9 | 2 | — | — |
| 0x0000_0068 | 26 | 10 | 2 | SPI0 | Single interrupt vector for all sources |
| 0x0000_006C | 27 | 11 | 2 | — | — |
| 0x0000_0070 | 28 | 12 | 3 | LPUART0 | Status and error |
| 0x0000_0074 | 29 | 13 | 3 | — | — |
| 0x0000_0078 | 30 | 14 | 3 | — | — |

Table continues on the next page...

Table 3-2. Interrupt vector assignments (continued)

| Address | Vector | IRQ ¹ | NVIC IPR register number ² | Source module | Source description |
|-------------|--------|------------------|---------------------------------------|---------------------|---|
| 0x0000_007C | 31 | 15 | 3 | ADC0 | Conversion complete |
| 0x0000_0080 | 32 | 16 | 4 | CMP0 | Rising or falling edge of comparator output |
| 0x0000_0084 | 33 | 17 | 4 | TPM0 | Overflow or channel interrupt |
| 0x0000_0088 | 34 | 18 | 4 | TPM1 | Overflow or channel interrupt |
| 0x0000_008C | 35 | 19 | 4 | — | — |
| 0x0000_0090 | 36 | 20 | 5 | RTC | Alarm interrupt |
| 0x0000_0094 | 37 | 21 | 5 | RTC | Seconds interrupt |
| 0x0000_0098 | 38 | 22 | 5 | — | — |
| 0x0000_009C | 39 | 23 | 5 | — | — |
| 0x0000_00A0 | 40 | 24 | 6 | — | — |
| 0x0000_00A4 | 41 | 25 | 6 | — | — |
| 0x0000_00A8 | 42 | 26 | 6 | — | — |
| 0x0000_00AC | 43 | 27 | 6 | — | — |
| 0x0000_00B0 | 44 | 28 | 7 | LPTMR0 | LP Timer compare match |
| 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 B) |

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

3.2.3.1 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 3-3. Interrupt vector assignments

| Address | Vector | IRQ ¹ | NVIC IPR register number ² | 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 field location within these particular registers:

- NVICIPR2 field starting location = $8 * (\text{IRQ mod } 4) + 6 = 22$

Since the NVICIPR fields are 2-bit wide (4 priority levels), the NVICIPR2 field range is 22–23.

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

- NVICIPR2[23:22]

3.3 AWIC introduction

The primary function of the AWIC block 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.

3.3.1 Wake-up sources

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

Table 3-4. AWIC stop wake-up sources

| Wake-up source | Description |
|-------------------------|--|
| Available system resets | RESET pin when LPO is its clock source |
| Low-voltage detect | Power management controller—functional in Stop mode |
| Low-voltage warning | Power management controller—functional in Stop mode |
| 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 or external crystal clock |
| CMP0 | Interrupt in normal or trigger mode |
| I ² C0 | Address match wakeup |
| LPUART0 | Any enabled interrupt can be a source as long as the module remains clocked |
| RTC | Alarm or seconds interrupt |
| NMI | NMI pin |
| TPMx | Any enabled interrupt can be a source as long as the module remains clocked |
| LPTMR | Any enabled interrupt can be a source as long as the module remains clocked |
| SPIx | Slave mode interrupt |

Chapter 4

Memory and Memory Map

4.1 Flash memory

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:

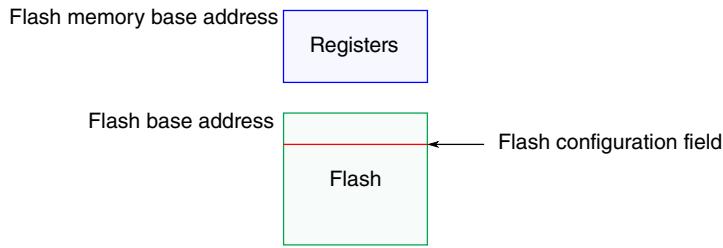
Table 4-1. KL03 flash memory size

| Part number | Program flash (KB) | Block 0 (P-Flash) address range |
|-----------------|--------------------|---------------------------------|
| MKL03Z8VFG4(R) | 8 KB | 0x0000_0000 – 0x0000_1FFF |
| MKL03Z16VFG4(R) | 16 KB | 0x0000_0000 – 0x0000_3FFF |
| MKL03Z32VFG4(R) | 32 KB | 0x0000_0000 – 0x0000_7FFF |
| MKL03Z8VFK4(R) | 8 KB | 0x0000_0000 – 0x0000_1FFF |
| MKL03Z16VFK4(R) | 16 KB | 0x0000_0000 – 0x0000_3FFF |
| MKL03Z32VFK4(R) | 32 KB | 0x0000_0000 – 0x0000_7FFF |
| MKL03Z32CAF4R | 32 KB | 0x0000_0000 – 0x0000_7FFF |
| MKL03Z32CBF4R | 32 KB | 0x0000_0000 – 0x0000_7FFF |

4.1.1 Flash memory map

The flash memory and the flash registers are located at different base addresses as shown in the figure found here.

The base address for each is specified in [System memory map](#).

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

The on-chip flash memory 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](#) for details of supported ranges.

Access 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.

4.1.2 Flash security

For information on how flash security is implemented on this device, see [Security](#).

4.1.3 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 must be ignored.

4.1.4 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.

4.1.5 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.

4.2 SRAM

4.2.1 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 4-2. KL03 SRAM memory size

| Device | SRAM(KB) |
|---|----------|
| MKL03Z8VFG4(R), MKL03Z16VFG4(R), MKL03Z32VFG4(R), MKL03Z8VFK4(R), MKL03Z16VFK4(R), MKL03Z32VFK4(R), MKL03Z32CAF4R | 2 |

4.2.2 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.

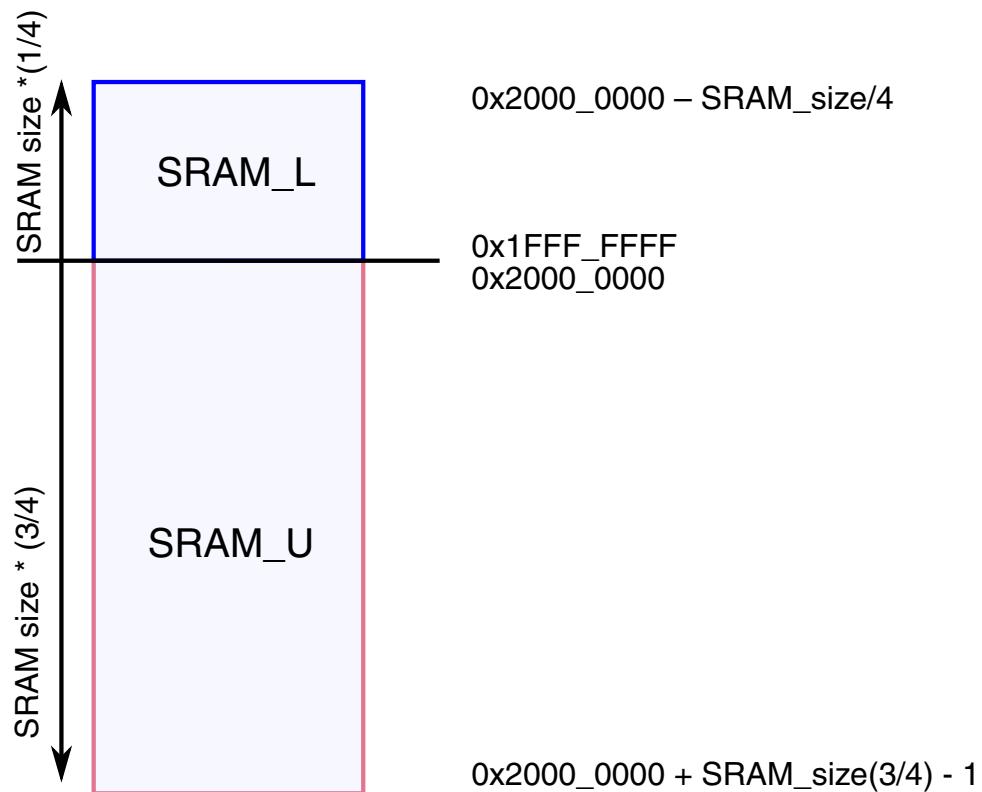
The on-chip RAM is implemented such that the SRAM_L and SRAM_U ranges form a contiguous block in the memory map. As such:

- 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 4-2. 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: $0x2000_0000 - 0x2000_2FFF$

4.2.3 SRAM retention in low power modes

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

4.3 System Register file

This device includes a 16-byte register file that is powered in all power modes.

Also, it retains contents during low-voltage detect (LVD) events and is only reset during a power-on reset.

4.4 Memory map

4.4.1 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.

4.4.2 System memory map

The table found here shows the high-level device memory map.

Table 4-3. System memory map

| System 32-bit address range | Destination slave | Access |
|--------------------------------------|---|-----------------|
| 0x0000_0000–0x07FF_FFFF ¹ | Program flash and read-only data (Includes exception vectors in first 196 bytes) | Cortex-M0+ core |
| 0x0800_0000–0x1BFF_FFFF | Reserved | — |
| 0x1C00_0000 – 0x1C00_1FFF | Boot ROM | Cortex-M0+ core |
| 0x1C00_2000 – 0x1FFF_FDFF | Reserved | — |
| 0xFFFF_FE00–0x1FFF_FFFF | SRAM_L: Lower SRAM | Cortex-M0+ core |
| 0x2000_0000–0x2000_05FF ² | SRAM_U: Upper SRAM | Cortex-M0+ core |
| 0x2000_0600–0x3FFF_FFFF | Reserved | — |
| 0x4000_0000–0x4007_FFFF | AIPS Peripherals | Cortex-M0+ core |
| 0x4008_0000–0x400F_EFFF | Reserved | — |
| 0x400F_F000–0x400F_FFFF | General-purpose input/output (GPIO) | Cortex-M0+ core |
| 0x4010_0000–0x43FF_FFFF | Reserved | — |
| 0x4400_0000–0x5FFF_FFFF | Bit Manipulation Engine (BME) access to AIPS Peripherals for slots 0-127 ³ | Cortex-M0+ core |
| 0x6000_0000–0xDFFF_FFFF | Reserved | — |
| 0xE000_0000–0xE00F_FFFF | Private Peripherals | Cortex-M0+ core |
| 0xE010_0000–0xEFFF_FFFF | Reserved | — |
| 0xF000_0000–0xF000_0FFF | Micro Trace Buffer (MTB) registers | Cortex-M0+ core |
| 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 |

Memory map

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](#) 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).

4.4.3 Flash memory map

The flash memory and the flash registers are located at different base addresses as shown in the figure found here.

The base address for each is specified in [System memory map](#).

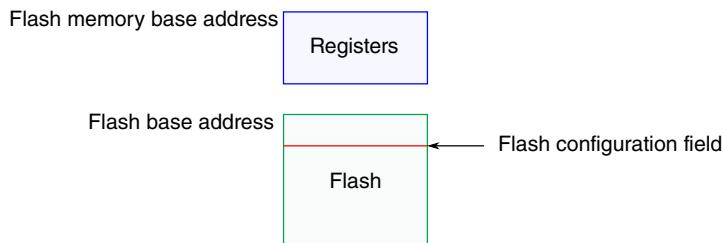


Figure 4-3. Flash memory map

The on-chip flash memory 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](#) for details of supported ranges.

Access 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.

4.4.4 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.

Access 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.

4.4.5 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.

4.4.6 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.

4.4.6.1 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, the 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.

4.4.6.2 Peripheral bridge (AIPS-Lite) memory map

Table 4-4. 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 | — |
| 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 | — |

Table continues on the next page...

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

| System 32-bit base address | Slot number | Module |
|----------------------------|-------------|-------------------------------------|
| 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 | — |
| 0x4002_2000 | 34 | — |
| 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 | — |
| 0x4003_8000 | 56 | Timer/PWM (LPTPM) 0 |
| 0x4003_9000 | 57 | Timer/PWM (LPTPM) 1 |
| 0x4003_A000 | 58 | — |
| 0x4003_B000 | 59 | Analog-to-digital converter 0(ADC0) |
| 0x4003_C000 | 60 | — |

Table continues on the next page...

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

| System 32-bit base address | Slot number | Module |
|----------------------------|-------------|---------------------------------|
| 0x4003_D000 | 61 | Real-time clock (RTC) |
| 0x4003_E000 | 62 | — |
| 0x4003_F000 | 63 | — |
| 0x4004_0000 | 64 | Low-power timer (LPTMR) |
| 0x4004_1000 | 65 | System register file |
| 0x4004_2000 | 66 | — |
| 0x4004_3000 | 67 | — |
| 0x4004_4000 | 68 | — |
| 0x4004_5000 | 69 | — |
| 0x4004_6000 | 70 | — |
| 0x4004_7000 | 71 | SIM low-power logic |
| 0x4004_8000 | 72 | System integration module (SIM) |
| 0x4004_9000 | 73 | Port A multiplexing control |
| 0x4004_A000 | 74 | Port B multiplexing control |
| 0x4004_B000 | 75 | — |
| 0x4004_C000 | 76 | — |
| 0x4004_D000 | 77 | — |
| 0x4004_E000 | 78 | — |
| 0x4004_F000 | 79 | — |
| 0x4005_0000 | 80 | — |
| 0x4005_1000 | 81 | — |
| 0x4005_2000 | 82 | — |
| 0x4005_3000 | 83 | — |
| 0x4005_4000 | 84 | LPUART0 |
| 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 | — |

Table continues on the next page...

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

| System 32-bit base address | Slot number | Module |
|----------------------------|-------------|---|
| 0x4006_4000 | 100 | Multi-purpose clock Generator Lite (MCG_Lite) |
| 0x4006_5000 | 101 | System oscillator (OSC) |
| 0x4006_6000 | 102 | I ² C0 |
| 0x4006_7000 | 103 | — |
| 0x4006_8000 | 104 | — |
| 0x4006_9000 | 105 | — |
| 0x4006_A000 | 106 | — |
| 0x4006_B000 | 107 | — |
| 0x4006_C000 | 108 | — |
| 0x4006_D000 | 109 | — |
| 0x4006_E000 | 110 | — |
| 0x4006_F000 | 111 | — |
| 0x4007_0000 | 112 | — |
| 0x4007_1000 | 113 | — |
| 0x4007_2000 | 114 | — |
| 0x4007_3000 | 115 | Analog comparator (CMP) / 6-bit digital-to-analog converter (DAC) |
| 0x4007_4000 | 116 | Voltage reference (VREF) |
| 0x4007_5000 | 117 | — |
| 0x4007_6000 | 118 | SPI0 |
| 0x4007_7000 | 119 | — |
| 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 |

4.4.6.3 Modules restricted access in user mode

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

4.4.7 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 4-5. PPB memory map

| System 32-bit Address Range | Resource | Additional Range Detail | Resource |
|-----------------------------|----------------------------|-------------------------|----------------------|
| 0xE000_0000–0xE000_DFFF | Reserved | | |
| 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 memory, and peripheral clocks can be configured independently. The clock distribution figure shows how clocks from the lite version of Multi Clock Generation (MCG-Lite) and OSC module are distributed to the microcontroller's other function units. Some modules in the microcontroller have selectable clock input.

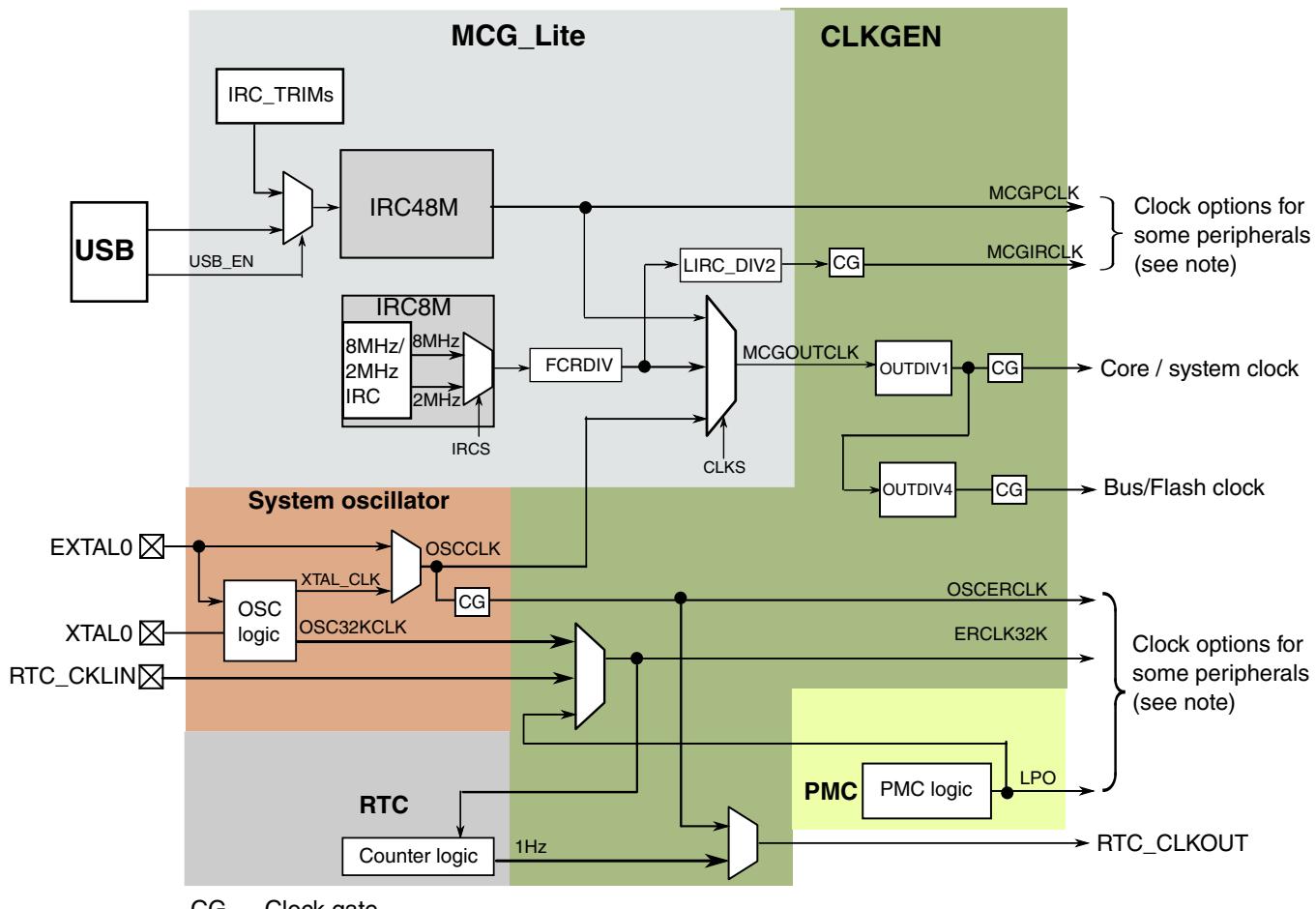
5.2 Programming model

The selection and multiplexing of system clock sources is controlled and programmed via the Clock Generation Module. The setting of clock dividers and module clock gating for the system are programmed via the SIM module. Refer to the [MCG_Lite](#) and [SIM](#) sections for detailed register and bit descriptions.

5.3 High-level device clocking diagram

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

| | OSC | MCG-Lite | SIM |
|--------------|--------|----------|----------------------|
| 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

Note1: See subsequent sections for details on where these clocks are used.

Note2: 48Mhz clock (IRC48M) control register is defined in either USB or MCG_Lite. In case USB is not available, IRC48M will be controlled by IRC_TRIMs in MCG_Lite module

Note3: FCRDIV support divider ratio 1,2,4,8,16, 32, 64, 128. LIRC_DIV2 provides the futher divide down for MCGIRCLK.

Figure 5-1. Clocking diagram

5.4 Clock definitions

The following table describes the clocks in the previous block diagram.

| Clock name | Description |
|----------------|--|
| Core clock | MCGOUTCLK divided by OUTDIV1 Clocks the ARM Cortex-M0+ core. |
| Platform clock | MCGOUTCLK divided by OUTDIV1 Clocks the crossbar switch and NVIC. |
| System clock | MCGOUTCLK divided by OUTDIV1 |

Table continues on the next page...

| Clock name | Description |
|-------------|---|
| | Clocks the bus masters directly . |
| Bus clock | System clock divided by OUTDIV4. Clocks the bus slaves and peripherals. |
| Flash clock | Flash memory clock On this device, it is the same as Bus clock. |
| MCGOUTCLK | MCG_Lite output of either IRC48M, IRC8M, MCG_Lite's external reference clock that sources the core, system, bus, and flash clock. |
| MCGIRCLK | MCG_Lite output of the slow (IRC8M) internal reference clock |
| MCGPCLK | MCG_Lite output of the fast (IRC48M) internal reference clock. This clock may clock some modules. In addition, this clock is used for LPUART0 and TPM modules |
| OSCCLK | System oscillator output of the internal oscillator or sourced directly from EXTAL. Used as MCG_Lite's external reference clock. |
| OSCRCLK | System oscillator output sourced from OSCCLK that may clock some on-chip modules |
| OSC32KCLK | System oscillator 32 kHz output |
| ERCLK32K | Clock source for some modules that is chosen as OSC32KCLK or RTC_CLKIN or LPO |
| LPO | PMC 1 kHz 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 clock frequency | VLPR mode clock frequency | Clock source | Clock is disabled when... |
|----------------|-----------------------------|------------------------------|-------------------------|--|
| MCGOUTCLK | Up to 48 MHz | Up to 8 MHz | MCG_Lite | In all stop modes except for partial stop modes. |
| MCGPCLK | Up to 48 MHz | N/A | MCG_Lite | MCG_Lite clock controls are not enabled and in all stop modes except for partial stop modes. |
| Core clock | Up to 48 MHz | Up to 4 MHz | MCGOUTCLK clock divider | In all wait and stop modes |
| 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 |

Table continues on the next page...

Table 5-1. Clock summary (continued)

| Clock name | Run mode clock frequency | VLPR mode clock frequency | Clock source | Clock is disabled when... |
|---|-------------------------------------|--|--------------------------------------|---|
| Bus clock | Up to 24 MHz | Up to 1 MHz | MCGOUTCLK clock divider | In all stop modes except for partial STOP2 mode, and Compute Operation |
| SWD Clock | Up to 24 MHz | Up to 1 MHz | SWD_CLK pin | In all stop modes |
| Flash clock | Up to 24 MHz | Up to 1 MHz in EXT and LIRC | MCGOUTCLK clock divider | In all stop modes except for partial STOP2 mode |
| Internal reference (MCGIRCLK) | 8/2MHz LIRC | 8/2MHz LIRC | MCG_Lite | MCG_C1[IRCLKEN] cleared, Stop/VLPS mode and MCG_C1[IREFSTEN] cleared, or VLLS mode |
| External reference (OSCERCLK) | Up to 48 MHz (bypass), 30–40 kHz | Up to 16 MHz (bypass), 30–40 kHz (low-range crystal) | System OSC | System OSC's OSC_CR[ERCLKEN] cleared, or Stop mode and OSC_CR[EREFSTEN] cleared or VLLS0 and oscillator not in external clock mode. |
| External reference 32kHz (ERCLK32K) | 30–40 kHz | 30–40 kHz | System OSC or RTC_CLKIN or LPO | System OSC's OSC_CR[ERCLKEN] cleared and RTC's RTC_CR[OSCE] cleared or VLLS0 and oscillator not in external clock mode. |
| RTC_CLKOUT | RTC 1Hz, OSCERCLK | RTC 1Hz, OSCERCLK | RTC 1Hz, OSCERCLK | Clock is disabled in VLLSx modes |
| CLKOUT32K | 32K | 32K | ERCLK32K | SIM_SOPT1[OSC3 2KOUT] not configured to drive ERCLK32K out. |
| LPO | 1 kHz | 1 kHz | PMC | in VLLS0, optional be disabled in VLLS1 and VLLS3 by SMC_STOPCTRL [LPOPO] |
| TPM clock | Up to 48 MHz | Up to 8 MHz | MCGIRCLK, MCGPCLK, OSCERCLK | SIM_SOPT2[TPMSRC]=00 selected clock source disabled |

Table continues on the next page...

Table 5-1. Clock summary (continued)

| Clock name | Run mode clock frequency | VLPR mode clock frequency | Clock source | Clock is disabled when... |
|---------------|-----------------------------|------------------------------|-----------------------------------|--|
| LPUART0 clock | Up to 48 MHz | Up to 8 MHz | MCGIRCLK, MCGPCLK, OSCERCLK | SIM_SOPT2[LPUART0 SRC]=00 selected clock source disabled |

5.5 Internal clocking requirements

The clock dividers are programmed via the SIM_CLKDIV1 register. The following requirements must be met when configuring the clocks for this device:

- 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.
- The frequency of bus clock and flash clock is divided by 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.
- MCGPCLK is used for peripheral which is fixed to 48 MHz.
- MCGIRCLK is also one of peripheral clock sources which is from IRC8M and can be divided down by a divider.

The following is a common clock configuration for this device:

| Clock | Max. Frequency |
|----------------|----------------|
| Core clock | 48 MHz |
| Platform clock | 48 MHz |
| System clock | 48 MHz |
| Bus clock | 24 MHz |
| Flash clock | 24 MHz |
| MCGIRCLK | 8 MHz |
| MCGPCLK | 48 MHz |

5.5.1 Clock divider values after reset

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

Clock gating

| FTFA_FOPT [4,0] | Core/system clock | Bus/Flash clock | Execution Mode |
|-----------------|-------------------|-------------------|----------------|
| 00 | 0x7 (divide by 8) | 0x1 (divide by 2) | VLPR |
| 01 | 0x3 (divide by 4) | 0x1 (divide by 2) | VLPR |
| 10 | 0x1 (divide by 2) | 0x1 (divide by 2) | RUN |
| 11 | 0x0 (divide by 1) | 0x1 (divide by 2) | RUN |

This gives the user flexibility in selecting between a lower frequency, low-power boot option and 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.

The default reset clock for core/system clock is 8 MHz from IRC8M.

5.5.2 VLPR mode clocking

The clock dividers cannot be changed while in VLPR mode. These dividers must be programmed prior to entering VLPR mode to guarantee operation. Maximum 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

5.6 Clock gating

The clock to each module can be individually gated on and off using bits of the SCGCx registers of the SIM module. 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 the SCGCx 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 |
|-------------------------------------|---------------------|--|------------------------|
| Core modules | | | |
| ARM Cortex-M0+ core | Platform clock | Core clock | — |
| NVIC | Platform clock | — | — |
| DAP | Platform clock | — | SWD_CLK |
| System modules | | | |
| 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 | — | — |
| COP watchdog | Bus clock | LPO, Bus Clock MCGIRCLK, OSCERCLK | — |
| Clocks | | | |
| MCG_Lite | Bus clock | MCGOUTCLK, MCGPCLK, MCGIRCLK, OSCERCLK, ERCLK32K | — |
| OSC | Bus clock | OSCERCLK | — |
| Memory and memory interfaces | | | |
| Flash Controller | Platform clock | Flash clock | — |
| Flash memory | Flash clock | — | — |
| Analog | | | |
| ADC | Bus clock | OSCERCLK | — |
| CMP | Bus clock | — | — |
| Internal Voltage Reference (VREF) | Bus clock | — | — |
| Timers | | | |
| TPM | Bus clock | TPM clock | TPM_CLKIN0, TPM_CLKIN1 |
| LPTMR | Bus clock | LPO, OSCERCLK, MCGPCLK, ERCLK32K | — |
| RTC | Bus clock | ERCLK32K | RTC_CLKOUT, RTC_CLKIN |
| Communication interfaces | | | |
| SPI0 | Bus clock | — | SPI0_SCK |
| I ² C0 | System Clock | — | I ² C0_SCL |
| LPUART0 | Bus clock | LPUART0 clock | — |
| Human-machine interfaces | | | |
| GPIO | Platform 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 four clock sources as shown in the following figure.

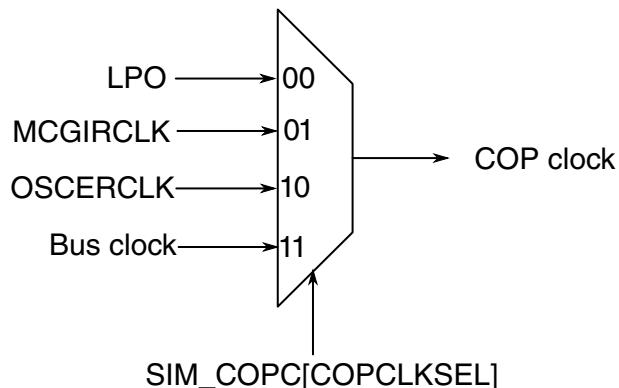


Figure 5-2. COP clock generation

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.

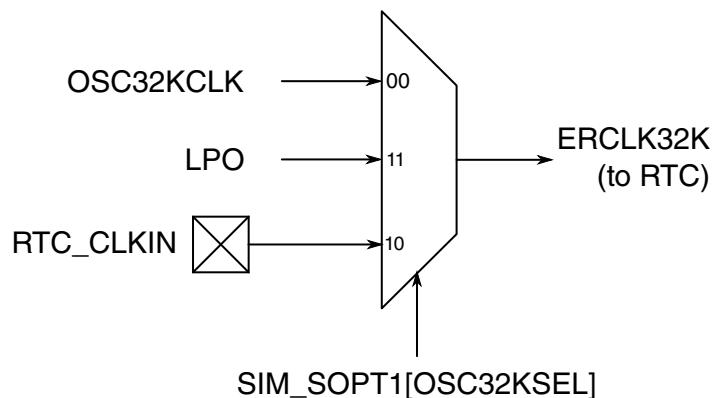


Figure 5-3. RTC clock generation

5.7.4 RTC_CLKOUT and CLKOUT32K clocking

When the RTC is enabled and the port control module selects the RTC_CLKOUT function, the RTC_CLKOUT signal, controlled from SIM_SOPT2[RTCCLKOUTSEL], outputs a 1 Hz or 32 kHz output derived from RTC oscillator as shown below and can be configured to drive to external pins via pin control configuration for the associated pin. It is also possible to drive CLKOUT32K on the same pins as controlled by SIM_SOPT1[OSC32KOUT] on the selected RTC_CLKOUT pins in all modes of operation (including VLLS and System Reset), overriding the existing pin mux configuration for that pin.

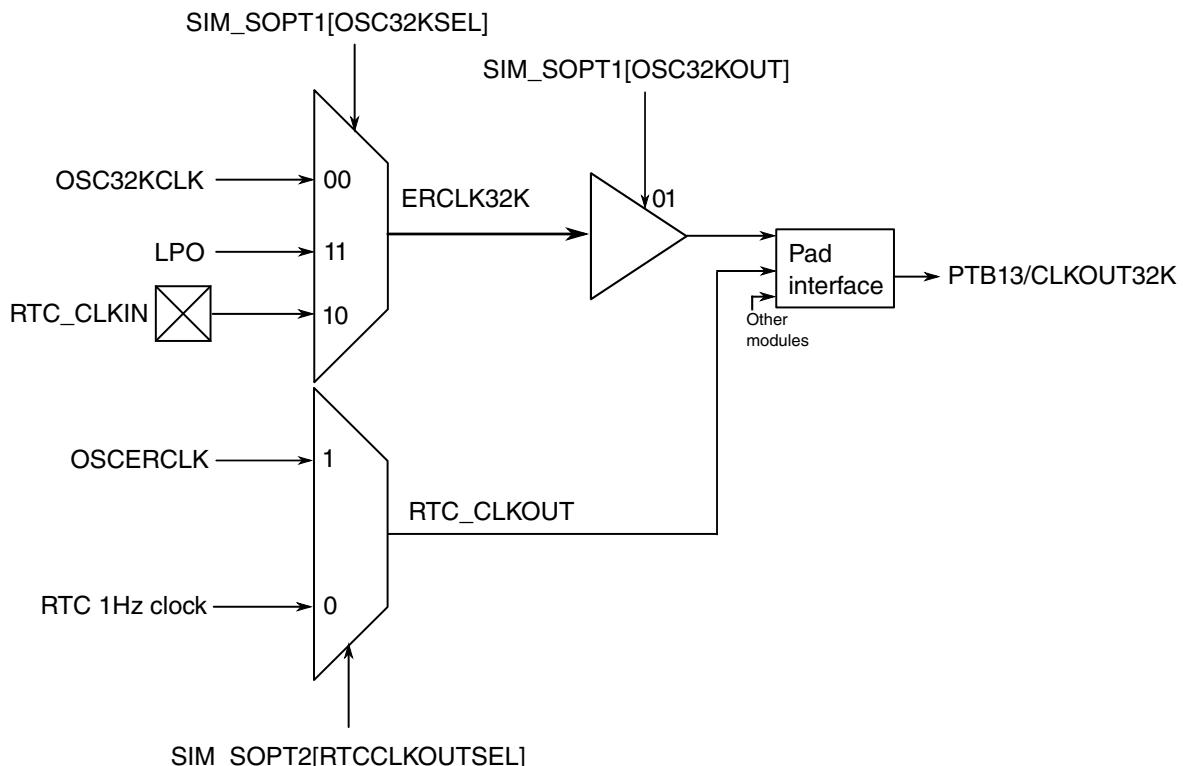


Figure 5-4. RTC_CLKOUT and CLKOUT32K generation

5.7.5 LPTMR clocking

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

NOTE

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

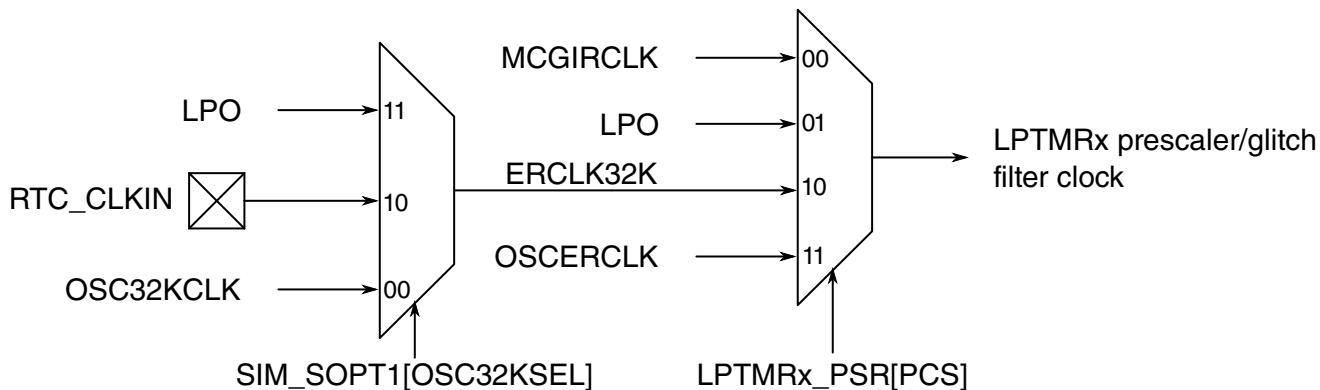


Figure 5-5. LPTMRx prescaler/glitch filter clock generation

5.7.6 TPM clocking

The counter for the TPM modules has 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.

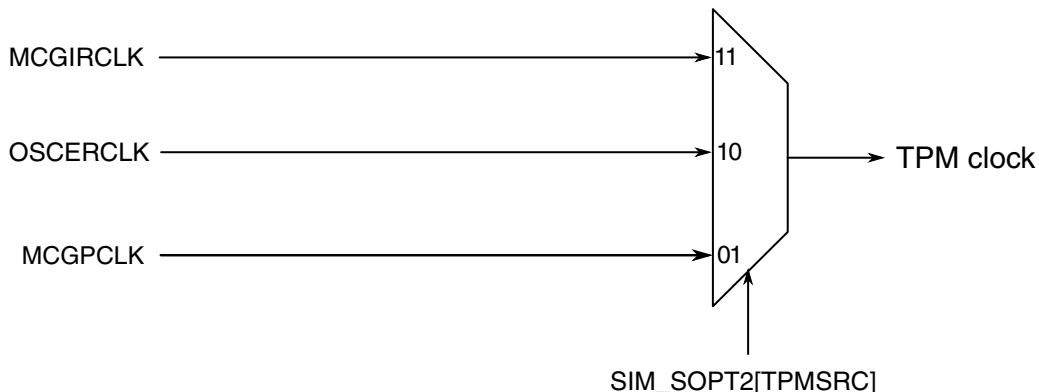


Figure 5-6. TPM clock generation

5.7.7 LPUART clocking

The LPUART0 has a selectable clock as shown in the following figure.

NOTE

The chosen clock must remain enabled if the LPUART0 is to continue operating in all required low-power modes.

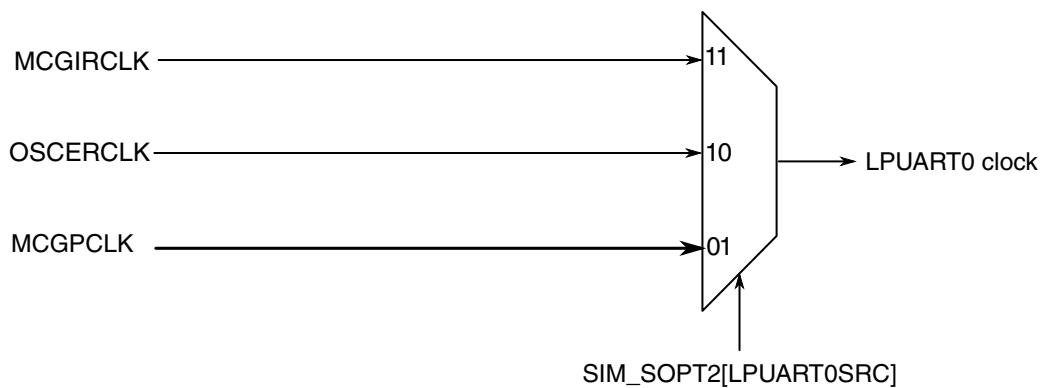


Figure 5-7. LPUART0 clock generation

Chapter 6

Reset and Boot

6.1 Introduction

The reset sources supported in this MCU are listed in the table found here.

Table 6-1. Reset sources

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

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

The information found here 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 fields in the 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 program counter (PC) from vector-table offset 4
- Link register (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 pulldown (PD)
- SWD_DIO in pullup (PU)

6.2.2.1 External pin reset (\overline{RESET})

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

The \overline{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.

6.2.2.1.1 **RESET** pin filter

The **RESET** pin filter supports filtering from both the 1 kHz LPO clock and the bus clock. RCM_RPFC[RSTFLTSS], RCM_RPFC[RSTFLTSRW], and RCM_RPFW[RSTFLTSEL] 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, VLLS3, and VLLS1 with SMC_STOPCTRL[LPOPO]=0), 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 or VLLSx modes.

The LVD can be configured to generate a reset upon detection of a low-voltage condition by setting PMC_LVDSC1[LVDRE] to 1. The low-voltage detection threshold is determined by PMC_LVDSC1[LVDV]. 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. RCM_SRS0[LVD] 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 RCM_SRS0[WDOG] to set.

6.2.2.4 Low leakage wakeup (LLWU)

The LLWU module provides the means for two external pins 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.

6.2.2.5 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.

6.2.2.6 Software reset (SW)

The SYSRESETREQ field 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 RCM_SRS1[SW] to set.

6.2.2.7 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 RCM_SRS1[LOCKUP] to set.

6.2.2.8 MDM-AP system reset request

Set the System Reset Request field 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 field is cleared.

Set the Core Hold Reset field 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-Lite.

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 $\overline{\text{RESET}}$ pin has also negated. It resets the remaining modules (the modules not reset by other reset types).

6.2.4 $\overline{\text{RESET}}$ pin

For all reset sources except a VLLS Wakeup that does not occur via the $\overline{\text{RESET}}$ pin, the $\overline{\text{RESET}}$ 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 $\overline{\text{RESET}}$ pin is released, and the internal Chip Reset negates after the $\overline{\text{RESET}}$ pin is pulled high. Keeping the $\overline{\text{RESET}}$ pin asserted externally delays the negation of the internal Chip Reset.

The $\overline{\text{RESET}}$ pin can be disabled by programming FTFA_FOPT[RESET_PIN_CFG] option bit to 0 (See [Table 6-2](#)). When this option is selected, there could be a short period of contention during a POR ramp where the device drives the pinout low prior to establishing the setting of this option and releasing the reset function on the pin.

6.3 Boot

The information found here 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
- boot ROM

This device supports booting from internal flash with the reset vectors located at addresses 0x0 (initial SP_main), 0x4 (initial PC).

The device is also able to boot from ROM. The ROM start address is from 0x1C00_0000. When boot from ROM, it remaps all vector fetch to ROM base address. ROM code start pointer locates in ROM vector table which address is 0x1C00_0000 where stack pointer is offset 0x0 and reset vector is offset 0x4. Vector table and stack pointer are valid out of reset. RCM mode register is cleared by software when Boot ROM completes, this disables remapping of vector fetches. Boot source can change between reset, but is always known before core reset negation. $\overline{\text{NMI}}$ input is disabled to platform when booting from ROM. See FOPT section and [Reset Control Module](#) for more detail options.

The boot options can be overridden by using RCM_FM[2:1] and RCM_MR[2:1] which can be written by software. The boot source remains set until the next System Reset or software can write logic one to clear one or both of the mode bits.

6.3.2 FOPT boot options

The Flash Option (FOPT) register in the Flash Memory module (FTFA_FOPT) 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 FTFA_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, see the flash memory chapter.

The MCU uses FTFA_FOPT to configure the device at reset as shown in the following table. An FTFA_FOPT value of 0x00 is invalid and will be ignored. The FOPT register is written to 0xFF if the contents of the Flash nonvolatile option are 0x00.

Table 6-2. Flash Option Register (FTFA_FOPT) definition

| Bit Num | Field | Value | Definition |
|---------|-------------|-------|--|
| 7-6 | BOOTSRC_SEL | | Boot Source Selection: these bits select the boot sources if boot pin option bit BOOTPIN_OPT = 1 |
| | | 00 | Boot from Flash |
| | | 01 | Reserved |
| | | 10 | Boot from ROM |

Table continues on the next page...

**Table 6-2. Flash Option Register (FTFA_FOPT) definition
(continued)**

| Bit Num | Field | Value | Definition |
|---------|---------------|-------|---|
| | | 11 | Boot from ROM |
| 5 | FAST_INIT | | Selects 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 | | Enables/disables control for the <u>RESET</u> pin. |
| | | 0 | <p><u>RESET</u> 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 pinout low prior to establishing the setting of this option and releasing the reset function on the pin.</p> <p>This bit is preserved through system resets and low-power modes. When <u>RESET</u> pin function is disabled, it cannot be used as a source for low-power mode wake-up.</p> <p>NOTE: When the reset pin has been disabled and security has been enabled by means of the FSEC register, a mass erase can be performed only by setting both the Mass Erase and System Reset Request fields in the MDM-AP register.</p> |
| | | 1 | <u>RESET</u> _b pin is dedicated. The port is configured with pullup enabled, open drain, passive filter enabled. |
| | | | Enables/disables control for the NMI function. |
| 2 | NMI_DIS | 0 | NMI interrupts are always blocked. The associated pin continues to default to <u>NMI</u> pin controls with internal pullup enabled. When <u>NMI</u> pin function is disabled, it cannot be used as a source for low-power mode wake-up. |
| | | 1 | NMI_b pin/interrupts reset default to enabled. |
| | | | External pin selects boot options |
| 1 | BOOTPIN_OPT | 0 | Force Boot from ROM if BOOTCFG0 asserted, where BOOTCFG0 is the boot config function which is muxed with <u>NMI</u> pin. <u>RESET</u> pin must be enabled (FOPT[RESET_PIN_CFG] = 1) when this option is selected. <u>NMI</u> pin is sampled at the end of reset (when reset pin negates). If BOOTCFG0 pin is not asserted, Boot source configured by FOPT[7:6] (BOOTSRC_SEL) bits. |
| | | 1 | Boot source configured by FOPT[7:6] (BOOTSRC_SEL) bits |
| 4,0 | LPBOOT | | Controls the reset value of OUTDIV1 value in SIM_CLKDIV1 register, and the state of the RUNM register in SMC_PMCTRL. 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. ¹ |
| | | 00 | Core and system clock divider (OUTDIV1) is 0x7 (divide by 8). Device is configured for VLPR mode on exit from reset. |
| | | 01 | Core and system clock divider (OUTDIV1) is 0x3 (divide by 4). Device is configured for VLPR mode on exit from reset. |
| | | 10 | Core and system clock divider (OUTDIV1) is 0x1 (divide by 2). Device is configured for RUN mode on exit from reset. |
| | | 11 | Core and system clock divider (OUTDIV1) is 0x0 (divide by 1). Device is configured for RUN mode on exit from reset. |

- Refer to [Clock divider values after reset](#) and RCM_FM, RCM_MR in the [Reset Control Module \(RCM\)](#) for details.

6.3.3 Boot sequence

The following figure is KL03 boot flow chart.

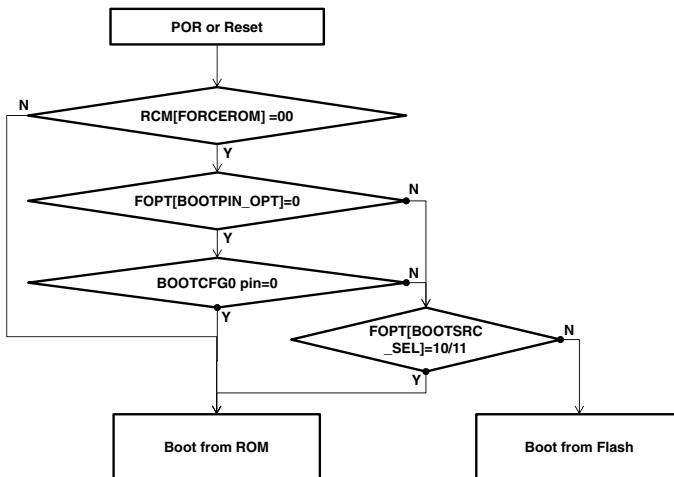


Figure 6-1. Chip boot flow chart

At power up, the on-chip regulator holds the system in a POR state until the input supply exceeds 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.

- A system reset is held on internal logic, the RESET pin is driven out low, and the MCG-Lite is enabled in its default clocking mode.
- Required clocks are enabled (system clock, flash clock, and any bus clocks that do not have clock gate control reset to disabled).
- 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.
- Early in reset sequencing, the NVM option byte is read and stored to the FOPT register of the Flash Memory module (FTFA_FOPT). If the bits associated with FTFA_FOPT[LPBOOT] are programmed for an alternate clock divider reset value, the system/core clock is switched to a slower clock speed. If FTFA_FOPT[FAST_INIT] is programmed clear, the flash initialization switches to slower clock resulting longer recovery times.
- 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 next sequence of events depends on the $\overline{\text{NMI}}$ /BOOTCFG0 input and FTFA_FOPT[NMI_DIS] and FTFA_FOPT[BOOTSRC_SEL] and FTFA_FOPT[BOOTPIN_OPT] as well as RCM_FM[FORCEROM] and RCM_MR[BOOTROM](See [Table 6-2](#) and RCM block guide) :

- If the $\overline{\text{NMI}}$ /BOOTCFG0 input is high or the NMI function is disabled in FTFA_FOPT, the CPU begins execution at the PC location.
- If the $\overline{\text{NMI}}$ /BOOTCFG0 input is low, the NMI function is enabled in FTFA_FOPT, and FTFA_FOPT[BOOTPIN_OPT] = 1, this results in an NMI interrupt. The processor executes an Exception Entry and reads the NMI interrupt handler address from vector-table offset 8. The CPU begins execution at the NMI interrupt handler.
- When FTFA_FOPT[BOOTPIN_OPT] = 0, it forces boot from ROM if $\overline{\text{NMI}}$ /BOOTCFG0 pin set to 0.

NOTE

When working in Debug mode, write 00b to the FTFA_FOPT[BOOTSRC_SEL] to configure boot source to internal flash, otherwise, code start pointer locates in ROM vector table.

If the NMI function is not required, either for an interrupt or wake up source, it is recommended that the NMI function be disabled by clearing NMI_DIS in the FOPT register.

Subsequent system resets follow this same reset flow.

Chapter 7

Power Management

7.1 Introduction

Information about the various chip power modes and functionality of the individual modules in these modes can be found here.

See [AN4503: Power Management for Kinetis and ColdFire+ MCUs](#) for further details on power management techniques.

7.2 Clocking modes

Information found here 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.

Clocking modes

PSTOP2 is functionally similar to WAIT mode, but offers additional power savings through the gating of the System clock. All the bus masters are disabled.

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.

PSTOP1 is functionally similar to Stop mode, but offers faster wake-up 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 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 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.

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 (MCM_CPO), which is only accessible to the CPU. Setting or clearing MCM_CPO[CPOREQ] 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 field in the CPO register of MCM module (MCM_CPO[CPOACK]) indicates when entry has completed.
- When exiting Compute Operation in Run mode, MCM_CPO[CPOACK] 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 that MCM_CPO[CPOACK] is polled to determine when the AIPS peripheral space can be accessed without generating a bus error.

7.2.3 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 memory 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.
- The CPU is in Compute Operation, including the entry sequence.

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.

If the flash memory 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 wake-up 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.

7.2.4 Clock gating

To conserve power, the clocks to most modules can be turned off using the SCGCx registers in the SIM module. The bits of these registers 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, see 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 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.

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. <ul style="list-style-type: none"> 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. <ul style="list-style-type: none"> 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. <ul style="list-style-type: none"> NVIC is disabled. AWIC is used to wake up from interrupt. Peripheral clocks are stopped. | Sleep Deep | Interrupt |

Table continues on the next page...

Table 7-1. Chip power modes (continued)

| Chip mode | Description | Core mode | Normal recovery method |
|-------------------------------------|---|------------|----------------------------|
| VLPR (Very Low-Power Run) | <p>On-chip voltage regulator is in a low-power mode that supplies only enough power to run the chip at a reduced frequency. Only MCG-Lite modes LIRC and EXT can be used in VLPR.</p> <ul style="list-style-type: none"> Reduced frequency Flash access mode (1 MHz) LVD off In LIRC clock mode, only the internal reference oscillator (LIRC8M) is available to provide a low power nominal 4 MHz source for the core with the nominal bus and flash clock required to be <1 MHz Alternatively, EXT 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 | <p>Same as VLPR but with the core in Sleep mode to further reduce power.</p> <ul style="list-style-type: none"> 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 | <p>Places chip in static state with LVD operation off. Lowest power mode with ADC and pin interrupts functional.</p> <ul style="list-style-type: none"> Peripheral clocks are stopped, but OSC, LPTMR, RTC, CMP can be used. LPUART and TPM 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 |
| VLLS3 (Very Low-Leakage Stop3) | <ul style="list-style-type: none"> Most peripherals are disabled (with clocks stopped), but OSC, LLWU, LPTMR, RTC, CMP can be used. NVIC is disabled; LLWU is used to wake up. SRAM_U and SRAM_L remain powered on (content retained and I/O states held). | Sleep Deep | Wake-up Reset |
| VLLS1 (Very Low-Leakage Stop1) | <ul style="list-style-type: none"> Most peripherals are disabled (with clocks stopped), but OSC, LLWU, LPTMR, RTC, CMP can be used. NVIC is disabled; LLWU is used to wake up. All of SRAM_U and SRAM_L are powered off. The 16-byte system register file remains powered for customer-critical data | Sleep Deep | Wake-up Reset ¹ |
| VLLS0 (Very Low-Leakage Stop 0) | <ul style="list-style-type: none"> Most peripherals are disabled (with clocks stopped), but LLWU, LPTMR, RTC can be used. NVIC is disabled; LLWU is used to wake up. All of SRAM_U and SRAM_L are powered off. The 16-byte system register file remains powered for customer-critical data LPO disabled, optional POR brown-out detection | Sleep Deep | Wake-up Reset ¹ |

- 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 VLLS modes, the wake-up sources are limited to LLWU generated wake-ups, LPTMR, CMP, SRTC, NMI_b pin, or RESET_b pin assertions. When the NMI_b pin or RESET_b pin have been disabled through associated FTFA_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 table found here 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).

Debug modules are discussed separately; see [Debug in low-power modes](#). Number ratings (such as 4 MHz and 1 Mbit/s) represent the maximum frequencies or maximum data rates per mode. Following is list of terms also used in the table.

- 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 wake-up. For clocks, OFF means disabled.
- wakeup = Modules can serve as a wake-up source for the chip.

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

| Modules | VLPR | VLPW | Stop | VLPS | VLLSx |
|-------------------------------------|--|--|--|--|---|
| Core modules | | | | | |
| NVIC | FF | FF | static | static | OFF |
| System modules | | | | | |
| Mode controller | FF | FF | FF | FF | FF |
| LLWU ¹ | static | static | static | static | FF ² |
| Regulator | low power | low power | ON | low power | low power in VLLS3, OFF in VLLS0/1 |
| LVD | disabled | disabled | ON | disabled | disabled |
| Brown-out detection | ON | ON | ON | ON | ON in VLLS1/3, optionally disabled in VLLS0 ³ |
| COP watchdog | FF static in CPO | FF | static FF in PSTOP2 | static | OFF |
| Clocks | | | | | |
| 1 kHz LPO | ON | ON | ON | ON | ON in VLLS1/3 , optional be disabled by SMC_STOPCTRL [LPOPO] , OFF in VLLS0 |
| System oscillator (OSC) | OSCERCLK low range/low power (30~40 kHz) | OSCERCLK low range/low power (30~40 kHz) | OSCERCLK optional | OSCERCLK low range/low power (30~40 kHz) | low range/low power in VLLS1/3, OFF in VLLS0 |
| MCG-Lite | 8 MHz IRC | 8 MHz IRC | static - LIRC optional | static - LIRC optional | OFF |
| Core clock | 4 MHz max | OFF | OFF | OFF | OFF |
| System clock | 4 MHz max OFF in CPO | 4 MHz max | OFF | OFF | OFF |
| Bus clock | 1 MHz max OFF in CPO | 1 MHz max | OFF 24 MHz max in PSTOP2 from RUN 1MHz max in PSTOP2 from VLPR | OFF | OFF |
| Memory and memory interfaces | | | | | |
| Flash | 1 MHz max access - no program | low power | low power | low power | 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 | VLLSx |
|---------------------------------|---|--|--|--------------------------------|-------------------------------------|
| | No register access in CPO | | | | |
| SRAM_U and SRAM_L | low power | low power | low power | low power | low power in VLLS3, OFF in VLLS0/1 |
| System Register File | Powered | Powered | Powered | Powered | Powered |
| Communication interfaces | | | | | |
| UART0 | 1 Mbps Async operation in CPO | 1 Mbit/s | Async operation FF in PSTOP2 | Async operation | OFF |
| SPI0 | master mode 500 kbit/s, slave mode 250 kbit/s static, slave mode receive in CPO | master mode 500 kbit/s, slave mode 250 kbit/s | static, slave mode receive FF in PSTOP2 | static, slave mode receive | OFF |
| I ² C0 | 100 kbps static, address match wakeup in CPO | 100 kbit/s | static, address match wakeup | static, address match wakeup | OFF |
| Timers | | | | | |
| TPM | FF Async operation in CPO | FF | Async operation FF in PSTOP2 | Async operation | OFF |
| LPTMR | FF static in CPO | FF | Async operation FF in PSTOP2 | Async operation | Async operation ⁴ |
| RTC | FF Async operation in CPO | FF | Async operation FF in PSTOP2 | Async operation | Async operation ⁵ |
| Analog | | | | | |
| 12-bit ADC | FF | FF | ADC internal clock only | ADC internal clock only | OFF |
| CMP ⁶ | FF HS or LS compare in CPO | FF | HS or LS compare FF in PSTOP2 | HS or LS compare | LS compare in VLLS1/3, OFF in VLLS0 |
| 6-bit DAC | FF static in CPO | FF | static FF in PSTOP2 | static | static, OFF in VLLS0 |
| 1.2V VREF | FF | FF | FF | FF | OFF |
| Human-machine interfaces | | | | | |
| GPIO | FF IOPORT write only in CPO | FF | static output, wakeup input FF in PSTOP2 | static output, wakeup input | OFF, pins latched |

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] field in the SMC module (SMC_STOPCTRL[PORPO]) 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 VLLSx only supports low speed external pin to pin or external pin to DAC compares.

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.

8.1.1 Flash security

The flash module provides security information to the MCU based on the state held by FTFA_FSEC[SEC]. The MCU, in turn, confirms the security request and limits access to flash resources. During reset, the flash module initializes FTFA_FSEC 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 FTFA_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 (FTFA_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.

8.1.2 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.

8.1.2.1 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 field of the MDM-AP Control register 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 debug of this device 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 figure found here.

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 this table.

Table 9-2. MDM-AP register summary

| Address | Register | Description |
|-------------|----------|--|
| 0x0100_0000 | Status | See MDM-AP Status Register |
| 0x0100_0004 | Control | See MDM-AP Control Register |
| 0x0100_00FC | IDR | Read-only identification register that always reads as 0x001C_0020 |

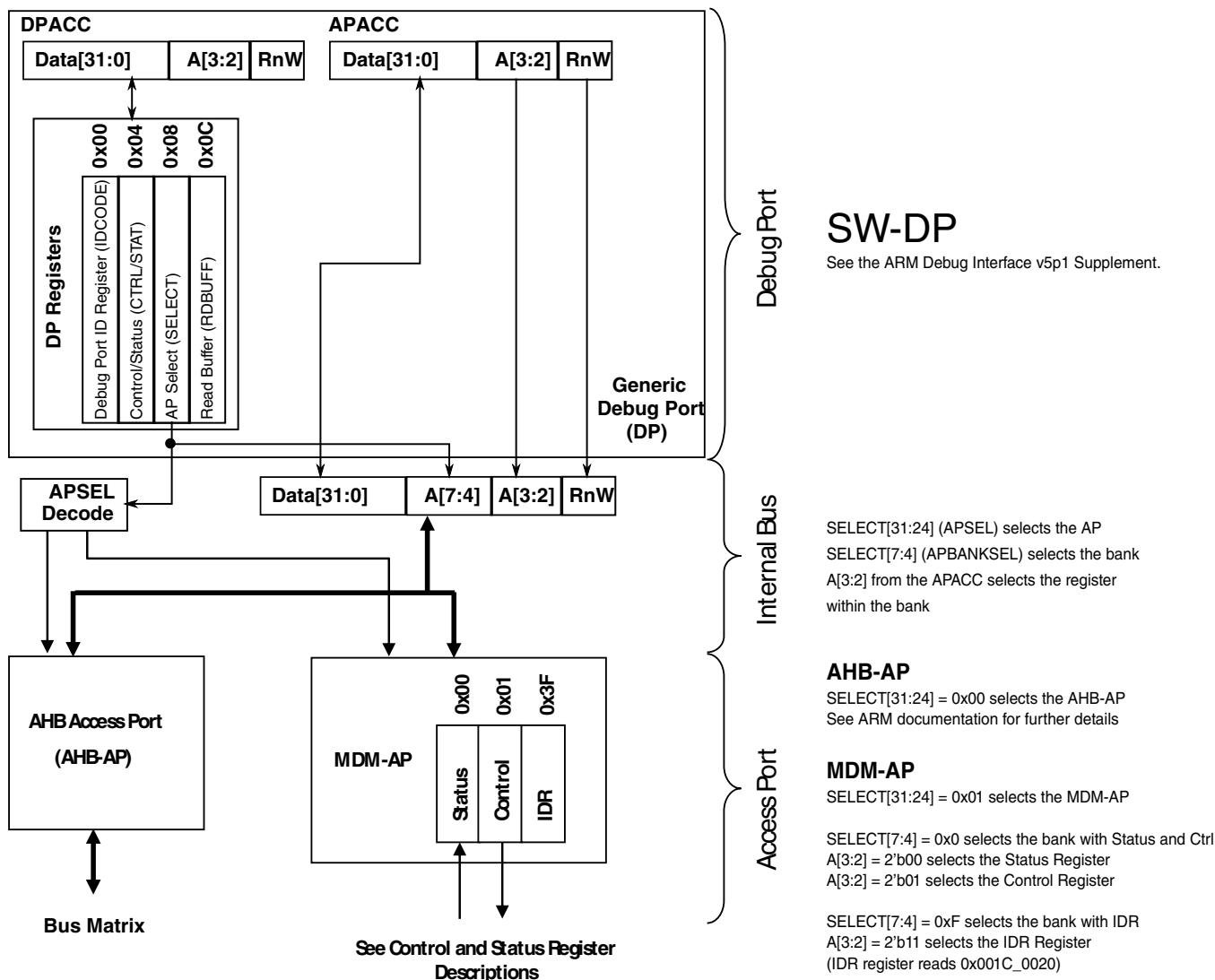


Figure 9-1. MDM AP addressing

9.3.1 MDM-AP Control Register

Table 9-3. MDM-AP Control register assignments

| Bit | Name | Secure ¹ | 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. |
| 2 | Debug Request | N | Set to force the core to halt. |

Table continues on the next page...

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

| Bit | Name | Secure ¹ | Description |
|--------|-------------------------------------|---------------------|---|
| | | | If the core is in a Stop or Wait mode, this bit can be used to wake the core and transition to a halted state. |
| 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 | Configuration bit to control core operation at the end of system reset sequencing. 0 Normal operation: Release the core from reset along with the rest of the system at the end of system reset sequencing. 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. |
| 5 | VLLSx Debug Request (VLLDBGREQ) | N | 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 reinitialize the debug modules. 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. 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. VLLDBGREQ clears automatically due to the POR reset generated as part of the VLLSx recovery. |
| 6 | VLLSx Debug Acknowledge (VLLDBGACK) | N | 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. VLLDBGACK 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. |
| 7 | VLLSx Status Acknowledge | N | Set this bit to acknowledge the DAP VLLS Status bits have been read. This acknowledge automatically clears the status bits. This bit is used by the debugger to clear the sticky VLLSx mode entry status bits. This bit is asserted and cleared by the debugger. |
| 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. 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 | Indicates 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. 0 System is in reset. 1 System is not in reset. |
| 4 | Reserved | |
| 5 | Mass Erase Enable | Indicates if the MCU can be mass erased or not 0 Mass erase is disabled. 1 Mass erase is enabled . |
| 6 | Backdoor Access Key Enable | Indicates if the MCU has the backdoor access key enabled. 0 Disabled 1 Enabled |
| 7 | LP Enabled | Decode of SMC_PMCTRL[STOPM] field to indicate that VLPS, or VLLSx are the selected power mode the next time the ARM Core enters Deep Sleep. 0 Low Power Stop Mode is not enabled. 1 Low Power Stop Mode is enabled. 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. This bit is used to throttle SWD_CLK frequency up/down. |
| 9 | | |
| 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. |

Table continues on the next page...

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

| Bit | Name | Description |
|---------|-------------------------|---|
| | | 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 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. 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:

- 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 field 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 be also 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.

Power mode entry logic monitors Debug Power Up and System Power Up signals from the debug port as indications that a debugger is active. These signals can be changed in RUN, VLPR, WAIT and VLPW. If the debug signal is active and the system attempts to enter Stop or VLPS, FCLK continues to run to support core register access. In these modes in which FCLK is left active the debug modules have access to core registers but not to system memory resources accessed via the crossbar.

With debug enabled, transitions from Run directly to VLPS result in the system entering Stop mode instead. Status bits within the MDM-AP Status register can be evaluated to determine this pseudo-VLPS state.

NOTE

With the debug enabled, transitions from Run --> VLPR --> VLPS are still possible.

In VLLS mode, all debug modules are powered off and reset at wakeup.

Going into a VLLSx mode causes all the debug controls and settings to be reset. To give time to the debugger to sync up with the HW, the MDM-AP Control register can be configured to hold the system in reset on recovery so that the debugger can regain control and reconfigure debug logic prior to the system exiting reset and resuming operation.

9.7 Debug and 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 has the capability of only 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. Information found here 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. Refer to that chapter to find which register controls the operation of a specific pin.

10.2 Signal multiplexing integration

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

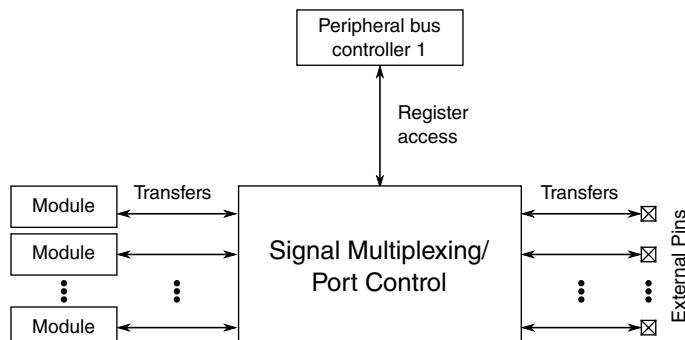


Figure 10-1. Signal multiplexing integration

Table 10-1. Reference links to related information

| Topic | Related module | Reference |
|-------------------|----------------|-----------------------------------|
| Full description | Port control | Port control |
| System memory map | | System memory map |

Table continues on the next page...

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

| Topic | Related module | Reference |
|-----------------|---------------------------|------------------------------------|
| Clocking | | Clock Distribution |
| Register access | Peripheral bus controller | Peripheral bridge |

10.2.1 I/O Port control and interrupt module features

- 32-bit ports

NOTE

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

- Each port is assigned one interrupt.

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 SIM_SCGC5[PORTx] to enable the clock. Before turning off the clock, make sure to disable the module. For more details, see 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 KL03 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.

NOTE

PTB3 and PTB4 are true open drain pins. The external pullup resistor must be added to make them output correct values in using I2C, GPIO, and LPUART0.

| 24 QFN | 20 WLC SP | 16 QFN | Pin Name | Default | ALT0 | ALT1 | ALT2 | ALT3 | ALT4 | ALT5 |
|--------|-----------|--------|----------------------------------|-----------------------|-----------------------|----------------------------------|------------|------------|------------|--------|
| 1 | — | — | PTB6/ IRQ_2/ LPTMR0_ALT3 | DISABLED | | PTB6/ IRQ_2/ LPTMR0_ALT3 | TPM1_CH1 | TPM_CLKIN1 | | |
| 2 | — | — | PTB7/ IRQ_3 | DISABLED | | PTB7/ IRQ_3 | TPM1_CH0 | | | |
| 3 | B5 | 1 | VDD | VDD | VDD | | | | | |
| 4 | C5 | 2 | VSS | VSS | VSS | | | | | |
| 5 | C4 | 3 | PTA3 | EXTAL0 | EXTAL0 | PTA3 | I2C0_SCL | I2C0_SDA | LPUART0_TX | |
| 6 | C3 | 4 | PTA4 | XTAL0 | XTAL0 | PTA4 | I2C0_SDA | I2C0_SCL | LPUART0_RX | CLKOUT |
| 7 | D3 | 5 | PTA5/ RTC_CLK_IN | DISABLED | | PTA5/ RTC_CLK_IN | TPM0_CH1 | SPI0_SS_b | | |
| 8 | D5 | 6 | PTA6 | DISABLED | | PTA6 | TPM0_CH0 | SPI0_MISO | | |
| 9 | — | — | PTB10 | DISABLED | | PTB10 | TPM0_CH1 | SPI0_SS_b | | |
| 10 | — | — | PTB11 | DISABLED | | PTB11 | TPM0_CH0 | SPI0_MISO | | |
| 11 | D4 | 7 | PTA7/ IRQ_4 | DISABLED | | PTA7/ IRQ_4 | SPI0_MISO | SPI0_MOSI | | |
| 12 | C1 | 8 | PTB0/ IRQ_5/ LLWU_P4 | ADC0_SE9 | ADC0_SE9 | PTB0/ IRQ_5/ LLWU_P4 | EXTRG_IN | SPI0_SCK | I2C0_SCL | |
| 13 | D1 | 9 | PTB1/ IRQ_6 | ADC0_SE8/ CMP0_IN3 | ADC0_SE8/ CMP0_IN3 | PTB1/ IRQ_6 | LPUART0_TX | LPUART0_RX | I2C0_SDA | |
| 14 | B1 | 10 | PTB2/ IRQ_7 | VREF_OUT/ CMP0_IN5 | VREF_OUT/ CMP0_IN5 | PTB2/ IRQ_7 | LPUART0_RX | LPUART0_TX | | |
| 15 | D2 | — | PTA8 | ADC0_SE3 | ADC0_SE3 | PTA8 | I2C0_SCL | SPI0_MOSI | | |
| 16 | C2 | — | PTA9 | ADC0_SE2 | ADC0_SE2 | PTA9 | I2C0_SDA | SPI0_SCK | | |
| 17 | A1 | 11 | PTB3/ IRQ_10 | DISABLED | | PTB3/ IRQ_10 | I2C0_SCL | LPUART0_TX | | |
| 18 | B2 | 12 | PTB4/ IRQ_11 | DISABLED | | PTB4/ IRQ_11 | I2C0_SDA | LPUART0_RX | | |
| 19 | A2 | 13 | PTB5/ IRQ_12 | NMI_b | ADC0_SE1/ CMP0_IN1 | PTB5/ IRQ_12 | TPM1_CH1 | NMI_b | | |
| 20 | B3 | — | PTA12/ IRQ_13/ LPTMR0_ALT2 | ADC0_SE0/ CMP0_IN0 | ADC0_SE0/ CMP0_IN0 | PTA12/ IRQ_13/ LPTMR0_ALT2 | TPM1_CH0 | TPM_CLKIN0 | | CLKOUT |

Pinout

| 24 QFN | 20 WLC SP | 16 QFN | Pin Name | Default | ALT0 | ALT1 | ALT2 | ALT3 | ALT4 | ALT5 |
|-----------|-----------------|-----------|--------------------------------|----------|------------------------|--------------------------------|------------|------------|------|------|
| 21 | A3 | — | PTB13/ CLKOUT32K | DISABLED | | PTB13/ CLKOUT32K | TPM1_CH1 | RTC_CLKOUT | | |
| 22 | A4 | 14 | PTA0/ IRQ_0/ LLWU_P7 | SWD_CLK | ADC0_SE15/ CMPO_IN2 | PTA0/ IRQ_0/ LLWU_P7 | TPM1_CH0 | SWD_CLK | | |
| 23 | B4 | 15 | PTA1/ IRQ_1/ LPTMR0_ALT1 | RESET_b | | PTA1/ IRQ_1/ LPTMR0_ALT1 | TPM_CLKIN0 | RESET_b | | |
| 24 | A5 | 16 | PTA2 | SWD_DIO | | PTA2 | CMP0_OUT | SWD_DIO | | |

10.3.2 KL03 pinouts

The following 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 [KL03 signal multiplexing and pin assignments](#).

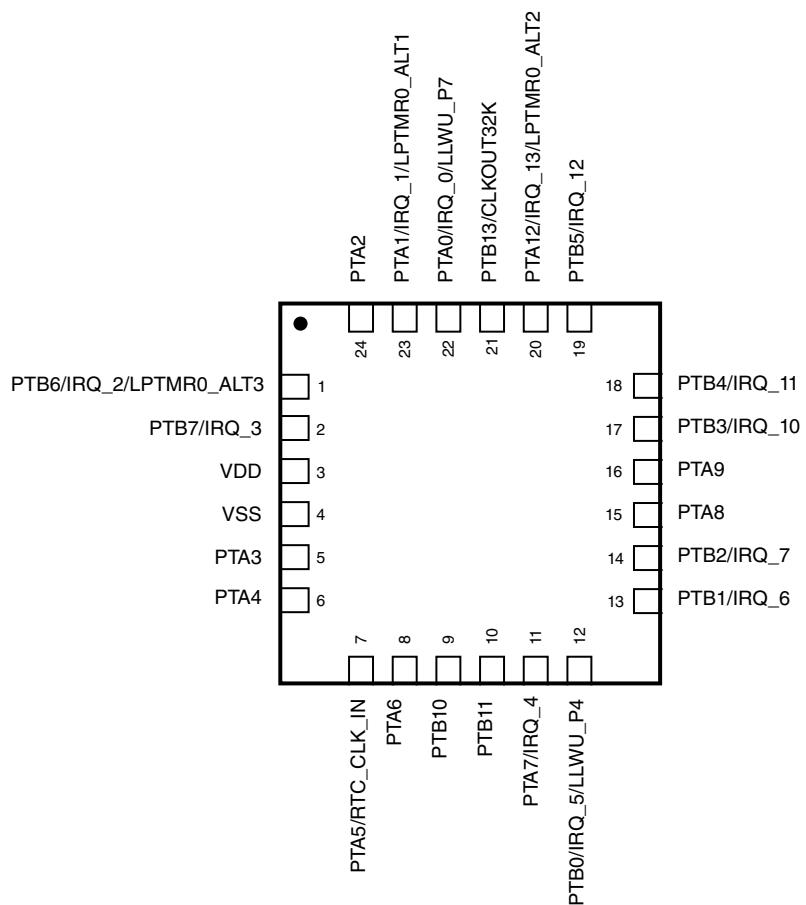
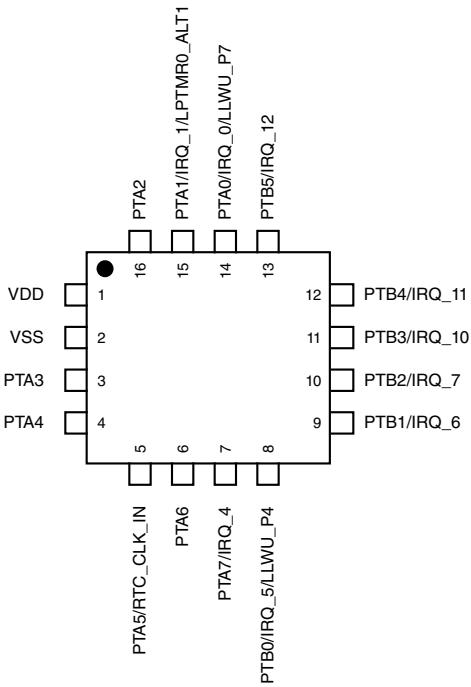


Figure 10-2. KL03 24-pin QFN pinout diagram

| | 1 | 2 | 3 | 4 | 5 |
|---|------|------|-------|------|------|
| A | PTB3 | PTB5 | PTB13 | PTA0 | PTA2 |
| B | PTB2 | PTB4 | PTA12 | PTA1 | VDD |
| C | PTB0 | PTA9 | PTA4 | PTA3 | VSS |
| D | PTB1 | PTA8 | PTA5 | PTA7 | PTA6 |

Figure 10-3. KL03 20-pin WLCSP pinout diagram**Figure 10-4. KL03 16-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 10-2. 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 10-3. System signal descriptions

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

Table 10-4. LLWU signal descriptions

| Chip signal name | Module signal name | Description | I/O |
|------------------|--------------------|--------------------------|-----|
| LLWU_Pn | LLWU_Pn | Wakeup inputs (n = 4, 7) | 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

This table presents the signal descriptions of the ADC0 module.

Table 10-6. ADC0 signal descriptions

| Chip signal name | Module signal name | Description | I/O |
|------------------|--------------------|------------------------------------|-----|
| ADC0_SEn | ADn | Single-Ended Analog Channel Inputs | I |
| VDD | V _{REFSH} | Voltage Reference Select High | I |
| VSS | V _{REFSL} | Voltage Reference Select Low | I |
| VDD | V _{DDA} | Analog Power Supply | I |
| VSS | V _{SSA} | Analog Ground | I |
| EXTRG_IN | ADHWT | Hardware trigger | I |

This table presents the signal descriptions of the CMP0 module.

Table 10-7. CMP0 signal descriptions

| Chip signal name | Module signal name | Description | I/O |
|------------------|--------------------|-----------------------|-----|
| CMP0_IN[3:0] | IN[3:0] | Analog voltage inputs | I |
| CMP0_OUT | CMPO | Comparator output | O |

Table 10-8. VREF signal descriptions

| Chip signal name | Module signal name | Description | I/O |
|------------------|--------------------|---|-----|
| VREF_OUT | VREF_OUT | Internally-generated voltage reference output | O |

10.4.6 Timer Modules

Table 10-9. TPM0 signal descriptions

| Chip signal name | Module signal name | Description | I/O |
|------------------|--------------------|---|-----|
| TPM_CLKIN[1:0] | 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 10-9. TPM0 signal descriptions (continued)

| Chip signal name | Module signal name | Description | I/O |
|------------------|--------------------|---|-----|
| TPM0_CH[1:0] | TPM_CHn | TPM channel (n = 1 to 0). A TPM channel pin is configured as output when configured in an output compare or PWM mode and the TPM counter is enabled, otherwise the TPM channel pin is an input. | I/O |

Table 10-10. TPM1 signal descriptions

| Chip signal name | Module signal name | Description | I/O |
|------------------|--------------------|---|-----|
| TPM_CLKIN[1:0] | 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 |
| TPM1_CH[1:0] | TPM_CHn | TPM channel (n = 1 to 0). A TPM channel pin is configured as output when configured in an output compare or PWM mode and the TPM counter is enabled, otherwise the TPM channel pin is an input. | I/O |

Table 10-11. LPTMR0 signal descriptions

| Chip signal name | Module signal name | Description | I/O |
|------------------|--------------------|-------------------------|-----|
| LPTMR0_ALT[3:1] | LPTMR_ALTn | Pulse Counter Input pin | I |

Table 10-12. RTC signal descriptions

| Chip signal name | Module signal name | Description | I/O |
|-------------------------|--------------------|-------------------------|-----|
| RTC_CLKOUT ¹ | 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 10-13. 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 10-14. I²C0 signal descriptions

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

Table 10-15. LPUART0 signal descriptions

| Chip signal name | Module signal name | Description | I/O |
|------------------|--------------------|---------------|-----|
| LPUART0_TX | TxD | Transmit data | I/O |
| LPUART0_RX | RxD | Receive data | I |

10.4.8 Human-machine interfaces (HMI)

Table 10-16. GPIO Signal Descriptions

| Chip signal name | Module signal name | Description | I/O |
|------------------------|--------------------|------------------------------|-----|
| PTA[31:0] | PORATA31–PORATA0 | General-purpose input/output | I/O |
| PTB[31:0] ¹ | PORTB31–PORTB0 | General-purpose input/output | I/O |

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

Chapter 11

Kinetis ROM Bootloader

11.1 Chip-Specific Information

This device has various peripherals (UART, I2C, SPI) supported by the Kinetis ROM Bootloader. To use an interface for bootloader communications, the peripheral must be enabled in the BCA, as shown in [Table 11-3](#). If the BCA is invalid (such as all 0xFF bytes), then all peripherals will be enabled by default. The next table shows the pads used by the Kinetis ROM Bootloader.

Table 11-1. Kinetis Bootloader Peripheral Pinmux

| Port | Signal |
|------|------------|
| PTB0 | SPI0_SCK |
| PTB1 | LPUART0_TX |
| PTB2 | LPUART0_RX |
| PTB3 | I2C0_SCL |
| PTB4 | I2C0_SDA |
| PTA5 | SPI0_SS_b |
| PTA6 | SPI0_MISO |
| PTA7 | SPI0_MOSI |

11.2 Introduction

The Kinetis bootloader is the program residing in the on-chip read-only memory (ROM) of a Kinetis microcontroller device. There is hardware logic in place at boot time that either starts execution of an embedded image available on the internal flash memory, or starts the execution of the Kinetis Bootloader from on-chip ROM.

Introduction

The Kinetis Bootloader's main task is to provision the internal flash memory with an embedded firmware image during manufacturing, or at any time during the life of the Kinetis device. The Kinetis Bootloader does the provisioning by acting as a slave device, and listening to various peripheral ports where a master can start communication.

For the Kinetis device, the Kinetis Bootloader can interface with I2C, SPI, and UART peripherals in slave mode and respond to the commands sent by a master (or host) communicating on one of those ports. The host/master can be a firmware-download application running on a PC or an embedded host communicating with the Kinetis Bootloader. Regardless of the host/master (PC or embedded host), the Kinetis Bootloader always uses a command protocol to communicate with that host/master. Commands are provided to write to memory (internal flash or RAM), erase flash, and get/set bootloader options and property values. The host application can query the set of available commands.

On start-up, the bootloader reads optional configuration parameters from a fixed area on flash called the bootloader configuration area (BCA). These parameters can be modified by the write memory command or by downloaded flash image. BCA parameters include configuration data such as enabled peripherals, peripheral-specific settings, etc.

This chapter describes Kinetis Bootloader features, functionality, command structure and which peripherals are supported.

Features supported by the Kinetis Bootloader in Kinetis ROM:

- Supports I2C, SPI, and UART peripheral interfaces
- Automatic detection of the active peripheral
- Ability to disable any peripheral
- UART peripheral implements autobaud
- Common packet-based protocol for all peripherals
- Packet error detection and retransmission
- Flash-resident configuration options
- Fully supports flash security, including ability to mass erase or unlock security via the backdoor key
- Protection of RAM used by the bootloader while it is running
- Provides command to read properties of the device, such as flash and RAM size
- Multiple options for executing the bootloader either at system start-up or under application control at runtime

Table 11-2. Commands supported by the Kinetis Bootloader in ROM

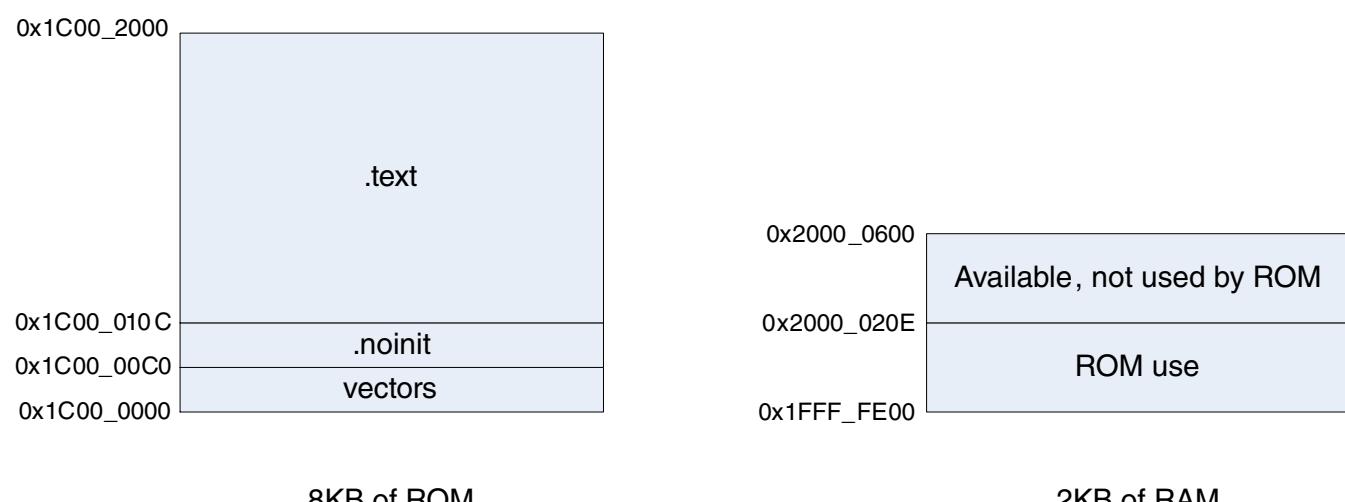
| Command | Description | When flash security is enabled, then this command is |
|-----------------------|--|--|
| Execute | Run user application code that never returns control to the bootloader | Not supported |
| FlashEraseAll | Erase the entire flash array | |
| FlashEraseRegion | Erase a range of sectors in flash | |
| WriteMemory | Write data to memory | |
| FlashSecurityDisable | Attempt to unlock flash security using the backdoor key | Supported |
| GetProperty | Get the current value of a property | |
| Reset | Reset the chip | |
| SetProperty | Attempt to modify a writable property | |
| FlashEraseAllUnsecure | Erase the entire flash array, including protected sectors | |

11.3 Functional Description

The following sub-sections describe the Kinetis Bootloader in KL03 ROM functionality.

11.3.1 Memory Maps

While executing, the Kinetis Bootloader uses ROM and RAM memory.

**Figure 11-1. Kinetis Bootloader ROM/RAM Memory Maps**

11.3.2 The Kinetis Bootloader Configuration Area (BCA)

The Kinetis Bootloader reads data from the Bootloader Configuration Area (BCA) to configure various features of the bootloader. The BCA resides in flash memory at offset 0x3C0, and provides all of the parameters needed to configure the Kinetis Bootloader operation. For uninitialized flash, the Kinetis Bootloader uses a predefined default configuration. A host application can use the Kinetis Bootloader to program the BCA for use during subsequent initializations of the bootloader.

Table 11-3. Configuration Fields for the Kinetis Bootloader

| Offset | Size (bytes) | Configuration Field | Description |
|-------------|--------------|----------------------------|--|
| 0x00 - 0x03 | 4 | tag | Magic number to verify bootloader configuration is valid. Must be set to 'kcfg'. |
| 0x04 - 0x07 | 4 | - | Reserved in KL03 |
| 0x08 - 0x0B | 4 | - | Reserved in KL03 |
| 0x0C - 0x0F | 4 | - | Reserved in KL03 |
| 0x10 | 1 | enabledPeripherals | Bitfield of peripherals to enable. bit 0 UART bit 1 I2C bit 2 SPI |
| 0x11 | 1 | i2cSlaveAddress | If not 0xFF, used as the 7-bit I2C slave address. |
| 0x12 - 0x13 | 2 | peripheralDetectionTimeout | Timeout in milliseconds for active peripheral detection |
| 0x14 - 0x15 | 2 | - | Reserved in KL03 |
| 0x16 - 0x17 | 2 | - | Reserved in KL03 |
| 0x18 - 0x1B | 4 | - | Reserved in KL03 |
| 0x1C | 1 | clockFlags | See Table 11-5 , clockFlags Configuration Field |
| 0x1D | 1 | clockDivider | Divider to use for core and bus clocks when in high speed mode |

The first configuration field 'tag' is a tag value or magic number. The tag value must be set to 'kcfg' for the bootloader configuration data to be recognized as valid. If tag-field verification fails, then the Kinetis Bootloader acts as if the configuration data is not present. The tag value is treated as a character string, so bytes 0-3 must be set as shown in the table.

Table 11-4. tag Configuration Field

| Offset | tag Byte Value |
|--------|----------------|
| 0 | 'k' (0x6B) |
| 1 | 'c' (0x63) |
| 2 | 'f' (0x66) |
| 3 | 'g' (0x67) |

The flags in the clockFlags configuration field are enabled if the corresponding bit is cleared (0).

Table 11-5. clockFlags Configuration Field

| Bit | Flag | Description |
|-------|-----------|--|
| 0 | HighSpeed | Enable high speed mode (i.e., 48 MHz). |
| 1 - 7 | Reserved | Not used in KL03 ROM |

11.3.3 Start-up Process

The following conditions will force the hardware to start the Kinetis Bootloader:

- BOOTSRC_SEL field of FOPT register is set to either 0b11 or 0b10. This forces the ROM to run out of reset.
- The BOOTCFG0 pin is asserted. The pin must be configured as BOOTCFG0 by setting the BOOTPIN_OPT bit of FOPT to 0.
- A user applications running on flash or RAM calls into the Kinetis Bootloader entry point address in ROM, to start Kinetis Bootloader execution.

The BOOTSRC_SEL bits (FOPT register, FOPT [7:6]) determine the boot source. The FOPT register is located in the flash configuration field at address 0x40D in the flash memory array. If BOOTSRC_SEL is set to 0b11, then the device will boot to ROM out of reset. Flash memory defaults to all 1s when erased, so a blank chip will automatically boot to ROM.

The BOOTCFG0 pin is shared with the NMI pin, with NMI being the default usage. Regardless of whether the NMI pin is enabled or not, the NMI functionality is disabled if the ROM is executed out of reset, for as long as the ROM is running.

Functional Description

When the ROM is executed out of reset, vector fetches from the CPU are redirected to the ROM's vector table in ROM memory at offset 0x1C00_0000. This ensures that any exceptions will be handled by the ROM.

After the Kinetis Bootloader has started, the following procedure starts bootloader operations:

1. The RCM_MR [FORCEROM] bits are set, so that the device will reboot back into the ROM if/when the device is reset.
2. Initializes the bootloader's .data and .bss sections.
3. Reads bootloader configuration data from flash at address 0x3C0. The configuration data is only used if the tag field is set to the expected 'kcfg' value. If the tag is incorrect, then the configuration values are set to default, as if the data was all 0xFF bytes.
4. Clocks are configured. See the [Clock Configuration](#) section.
5. Enabled peripherals are initialized.
6. The bootloader waits for communication to begin on a peripheral.
 - If detection times out, then the bootloader jumps to the user application in flash. See [Bootloader Exit state](#) section.
 - If communication is detected, then all inactive peripherals are shut down, and the command phase is entered.

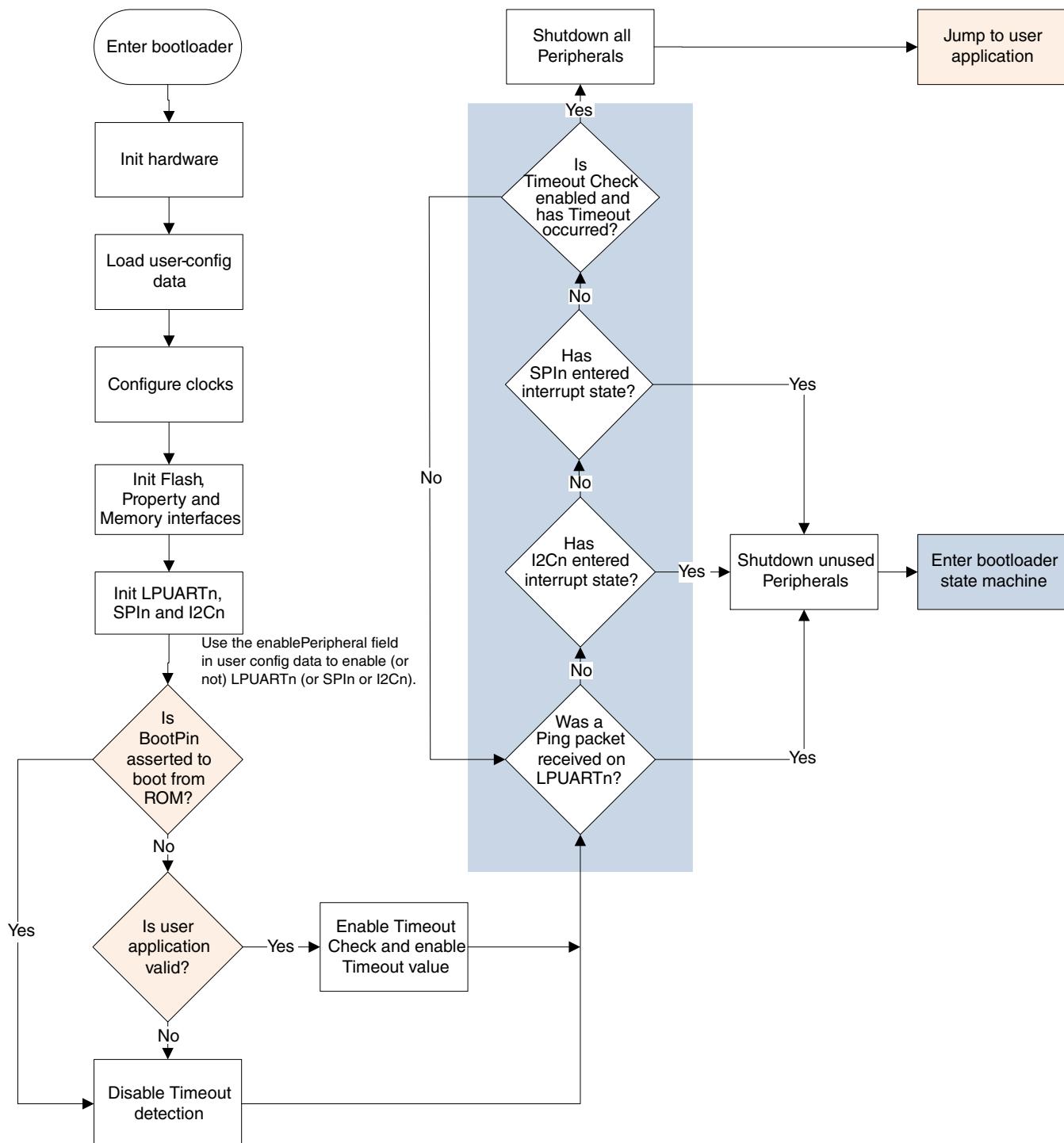


Figure 11-2. Kinetis Bootloader Start-up Flowchart

11.3.4 Clock Configuration

By default, the bootloader does not modify clocks. The Kinetis Bootloader in ROM will use the clock configuration of the chip out of reset unless the clock configuration bits in the FOPT register are cleared.

- Alternate clock configurations are supported, by setting fields in the Bootloader Configuration Area (BCA) shown in [Table 11-3](#).
- If the HighSpeed flag of the clockFlags configuration value is cleared, the bootloader will enable the internal 48 MHz reference clock.
- In high speed mode, the core and bus clock frequencies are determined by the clockDivider configuration value.
- The core clock divider is set directly from clockDivider.
- The bus clock divider is set to 1, unless the resulting bus clock frequency would be greater than the maximum supported value. In this case, the bus clock divider is increased until the bus clock frequency is at or below the maximum.
- Note that the maximum baud rate of serial peripherals is related to the core and bus clock frequencies. To achieve the desired baud rates, high speed mode should be enabled in BCA.

11.3.5 Bootloader Entry Point

The Kinetis Bootloader provides a function (`runBootloader`) that a user application can call, to run the bootloader.

To get the address of the entry point, the user application reads the word containing the pointer to the bootloader API tree at offset 0x1C of the bootloader's vector table. The vector table is placed at the base of the bootloader's address range, which for the ROM is 0x1C00_0000. Thus, the API tree pointer is at address 0x1C00_001C.

The bootloader API tree is a structure that contains pointers to other structures, which have the function and data addresses for the bootloader. The bootloader entry point is always the first word of the API tree.

The prototype of the entry point is:

```
void run_bootloader(void * arg);
```

The `arg` parameter is currently unused, and is intended for future expansion (for example, passing options to the bootloader). To ensure future compatibility, a value of `NULL` should be passed for `arg`.

Example code to get the entry pointer address from the ROM and start the bootloader:

```
// Variables
```

```

uint32_t runBootloaderAddress;
void (*runBootloader)(void * arg);

// Read the function address from the ROM API tree.
runBootloaderAddress = **(uint32_t **) (0x1c00001c);
runBootloader = (void (*) (void * arg)) runBootloaderAddress;

// Start the bootloader.
runBootloader(NULL);

```

11.3.6 Bootloader Protocol

This section explains the general protocol for the packet transfers between the host and the Kinetis Bootloader. The description includes the transfer of packets for different transactions, such as commands with no data phase and commands with incoming or outgoing data phase. The next section describes various packet types used in a transaction.

Each command sent from the host is replied to with a response command.

Commands may include an optional data phase:

- If the data phase is **incoming** (from host to bootloader), then the data phase is part of the **original command**.
- If the data phase is **outgoing** (from bootloader to host), then the data phase is part of the **response command**.

11.3.6.1 Command with no data phase

The protocol for a command with no data phase contains:

- Command packet (from host)
- Generic response command packet (to host)

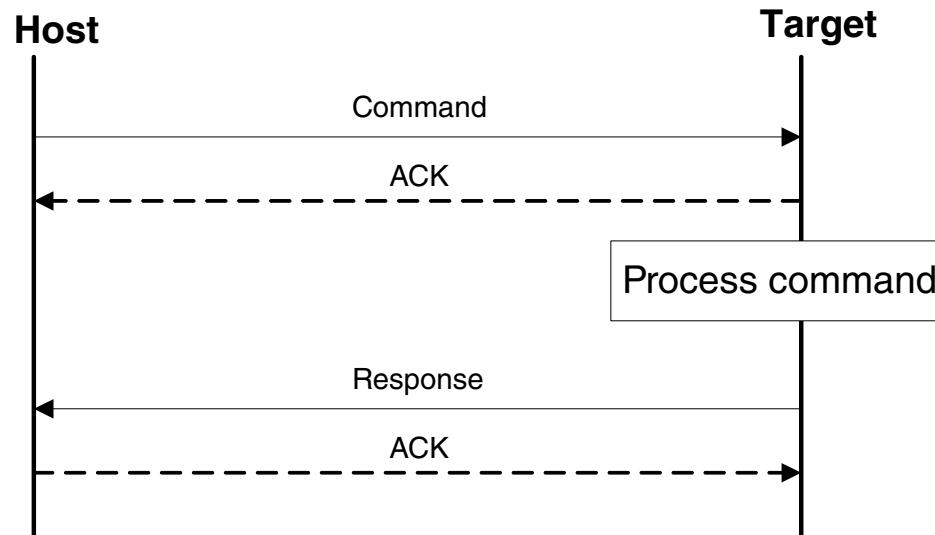


Figure 11-3. Command with No Data Phase

11.3.6.2 Command with incoming data phase

The protocol for a command with an incoming data phase contains:

- Command packet (from host)
- Generic response command packet (to host)
- Incoming data packets (from host)
- Generic response command packet (to host)

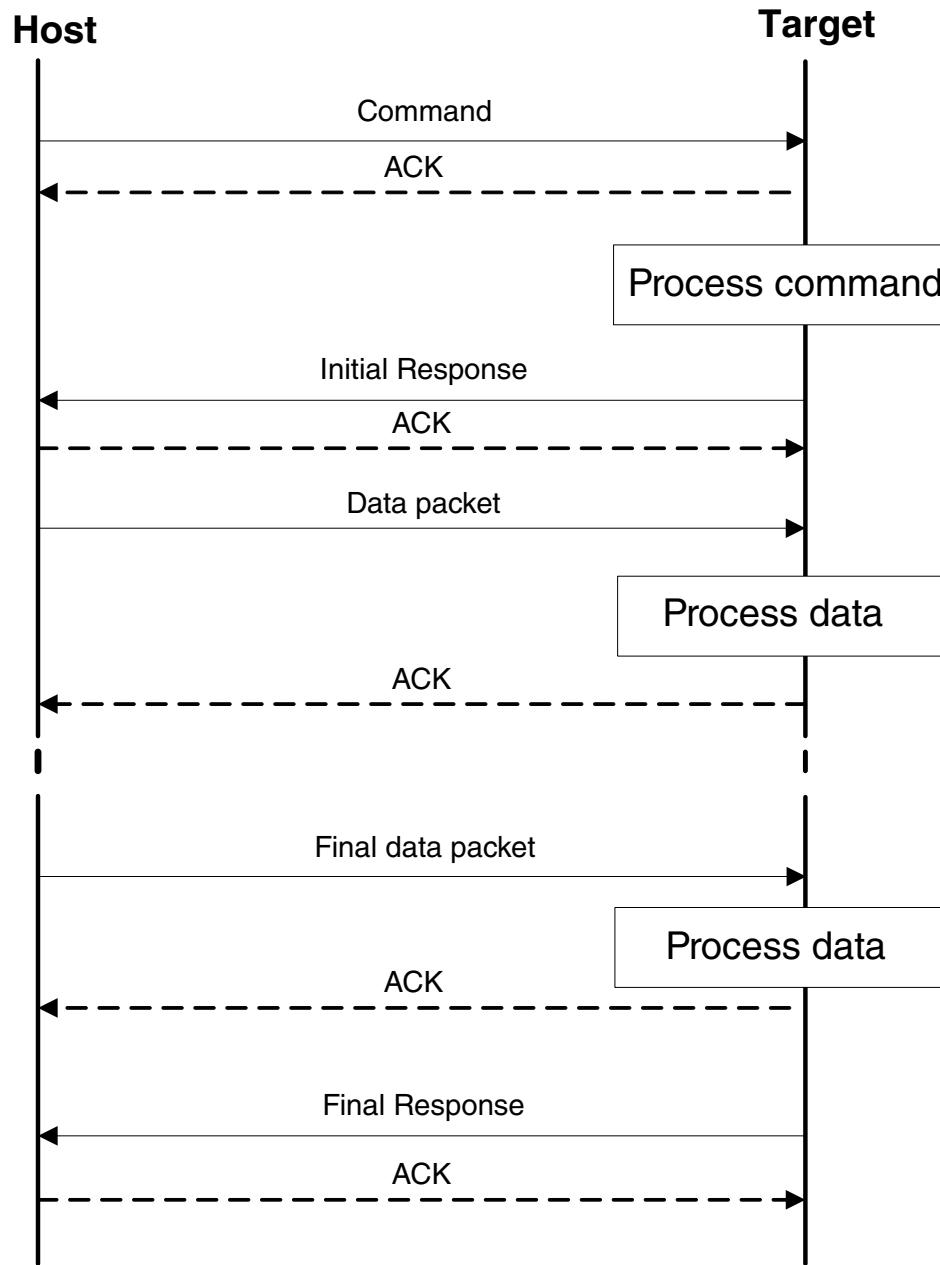


Figure 11-4. Command with incoming data phase

NOTE

- The host may not send any further packets while it (the host) is waiting for the response to a command.
- If the Generic Response packet prior to the start of the data phase does not have a status of kStatus_Success, then the data phase is aborted.
- Data phases may be aborted by the receiving side by sending the final Generic Response early with a status of

- kStatus_AbortDataPhase. The host may abort the data phase early by sending a zero-length data packet.
- The final Generic Response packet *sent after the data phase* includes the status for the entire operation.

11.3.6.3 Command with outgoing data phase

For KL03, there is no command available with an outgoing data phase.

11.3.7 Bootloader Packet Types

The Kinetis Bootloader device works in slave mode. All data communication is initiated by a host, which is either a PC or an embedded host. The Kinetis Bootloader device is the target, which receives a command or data packet. All data communication between host and target is packetized.

NOTE

The term "target" refers to the "Kinetis Bootloader device."

There are 6 types of packets used in the KL03 device:

- Ping packet
- Ping Response packet
- Framing packet
- Command packet
- Data packet
- Response packet

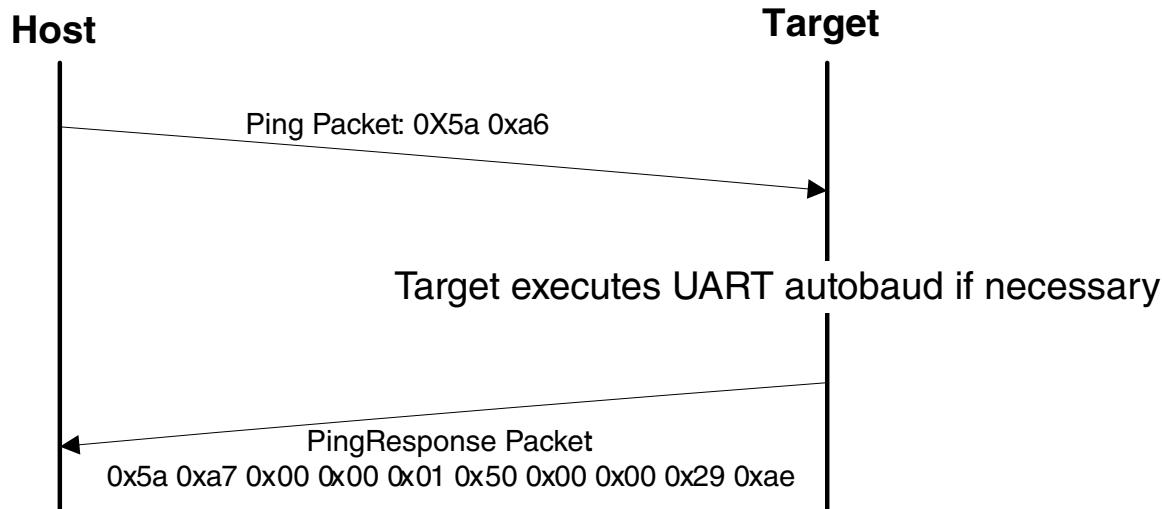
All fields in the packets are in little-endian byte order.

11.3.7.1 Ping packet

The Ping packet is the first packet sent from a host to the target (Kinetis Bootloader), to establish a connection on a selected peripheral. For a UART peripheral, the Ping packet is used to determine the baudrate. A Ping packet must be sent before any other communications. In response to a Ping packet, the target sends a Ping Response packet.

Table 11-6. Ping Packet Format

| Byte # | Value | Name |
|--------|-------|------------|
| 0 | 0x5A | start byte |
| 1 | 0xA6 | ping |

**Figure 11-5. Ping Packet Protocol Sequence**

11.3.7.2 Ping Response Packet

The target (Kinetis Bootloader) sends a Ping Response packet back to the host after receiving a Ping packet. If communication is over a UART peripheral, the target uses the incoming Ping pacaket to determine the baud rate before replying with the Ping Response packet. Once the Ping Response packet is received by the host, the connection is established, and the host starts sending commands to the target (Kinetis Bootloader).

Table 11-7. Ping Response Packet Format

| Byte # | Value | Parameter |
|--------|-------|----------------------------|
| 0 | 0x5A | start byte |
| 1 | 0xA7 | Ping response code |
| 2 | | Protocol bugfix |
| 3 | | Protocol minor |
| 4 | | Protocol major |
| 5 | | Protocol name = 'P' (0x50) |
| 6 | | Options low |
| 7 | | Options high |
| 8 | | CRC16 low |
| 9 | | CRC16 high |

The Serial Protocol Version number returned for the KL03 ROM is 1.0.01.1.0.

11.3.7.3 Framing Packet

The framing packet is used for flow control and error detection, and it (the framing packet) wraps command and data packets as well.

Table 11-8. Framing Packet Format

| Byte # | Value | Parameter | |
|-----------|-------|--------------------------------|---|
| 0 | 0x5A | start byte | |
| 1 | | packetType | |
| 2 | | length_low | |
| 3 | | length_high | Length is a 16-bit field that specifies the entire command or data packet size in bytes. |
| 4 | | crc16_low | |
| 5 | | crc16_high | This is a 16-bit field. The CRC16 value covers entire framing packet, including the start byte and command or data packets, but does not include the CRC bytes. See the CRC16 algorithm after this table. |
| 6 . . . n | | Command or Data packet payload | |

CRC16 algorithm:

```

uint16_t crc16_update(const uint8_t * src, uint32_t lengthInBytes)
{
    uint32_t crc = 0;
    uint32_t j;
    for (j=0; j < lengthInBytes; ++j)
    {
        uint32_t i;
        uint32_t byte = src[j];
        crc ^= byte << 8;
        for (i = 0; i < 8; ++i)
        {
            uint32_t temp = crc << 1;
            if (crc & 0x8000)
            {
                temp ^= 0x1021;
            }
            crc = temp;
        }
    }
}

```

```

    }
}

return crc;
}

```

A special framing packet that contains only a start byte and a packet type is used for synchronization between the host and target.

Table 11-9. Special Framing Packet Format

| Byte # | Value | Parameter |
|--------|-------|------------|
| 0 | 0x5A | start byte |
| 1 | 0xAn | packetType |

The Packet Type field specifies the type of the packet from one of the defined types (below):

Table 11-10. packetType Field

| packetType | Name | Description |
|------------|---------------------------------|--|
| 0xA1 | kFramingPacketType_Ack | The previous packet was received successfully; the sending of more packets is allowed. |
| 0xA2 | kFramingPacketType_Nak | The previous packet was corrupted and must be re-sent. |
| 0xA3 | kFramingPacketType_AckAbort | Data phase is being aborted. |
| 0xA4 | kFramingPacketType_Command | The framing packet contains a command packet payload. |
| 0xA5 | kFramingPacketType_Data | The framing packet contains a data packet payload. |
| 0xA6 | kFramingPacketType_Ping | Sent to verify the other side is alive. Also used for UART autobaud. |
| 0xA7 | kFramingPacketType_PingResponse | A response to Ping; contains the framing protocol version number and options. |

11.3.7.4 Command packet

The command packet carries a 32-bit command header and a list of 32-bit parameters.

Table 11-11. Command Packet Format

| Command Packet Format (32 bytes) | | | | | | | | | | |
|----------------------------------|--------|--------|-------------|--|-----------------|-----------------|-----------------|-----------------|-----------------|-----------------|
| Command Header (4 bytes) | | | | 28 bytes for Parameters (Max 7 parameters) | | | | | | |
| Tag | Flags | Rsvd | Param Count | Param1 (32-bit) | Param2 (32-bit) | Param3 (32-bit) | Param4 (32-bit) | Param5 (32-bit) | Param6 (32-bit) | Param7 (32-bit) |
| byte 0 | byte 1 | byte 2 | byte 3 | | | | | | | |

Table 11-12. Command Header Format

| Byte # | Command Header Field | |
|--------|---------------------------|--|
| 0 | Command or Response tag | The command header is 4 bytes long, with these fields. |
| 1 | Flags | |
| 2 | Reserved. Should be 0x00. | |
| 3 | ParameterCount | |

The header is followed by 32-bit parameters up to the value of the ParameterCount field specified in the header. Because a command packet is 32 bytes long, only 7 parameters can fit into the command packet.

Command packets are also used by the target to send responses back to the host. As mentioned earlier, command packets and data packets are embedded into framing packets for all of the transfers.

Table 11-13. Command Tags

| Command Tag | Name | |
|-------------|-----------------------|---|
| 0x01 | FlashEraseAll | The command tag specifies one of the commands supported by the Kinetis Bootloader in the KL03 ROM. The valid command tags for the Kinetis Bootloader in KL03 ROM are listed here. |
| 0x02 | FlashEraseRegion | |
| 0x04 | WriteMemory | |
| 0x05 | FillMemory | |
| 0x06 | FlashSecurityDisable | |
| 0x07 | GetProperty | |
| 0x09 | Execute | |
| 0x10 | FlashReadResource | |
| 0x0B | Reset | |
| 0x0C | SetProperty | |
| 0x0D | FlashEraseAllUnsecure | |
| 0x0E | FlashProgramOnce | |
| 0x0F | FlashReadOnce | |

Table 11-14. Response Tags

| Response Tag | Name | |
|--------------|--|---|
| 0xA0 | GenericResponse | The response tag specifies one of the responses the Kinetis Bootloader (target) returns to the host. The valid response tags are listed here. |
| 0xA7 | GetPropertyResponse (used for sending responses to GetProperty command only) | |

Flags: Each command packet contains a Flag byte. Only bit 0 of the flag byte is used. If bit 0 of the flag byte is set to 1, then data packets will follow in the command sequence. The number of bytes that will be transferred in the data phase is determined by a command-specific parameter in the parameters array.

ParameterCount: The number of parameters included in the command packet.

Parameters: The parameters are word-length (32 bits). With the default maximum packet size of 32 bytes, a command packet can contain up to 7 parameters.

11.3.7.5 Data packet

The data packet carries just the data, either host sending data to target, or target sending data to host. The data transfer direction is determined by the last command sent from the host. The data packet is also wrapped within a framing packet, to ensure the correct packet data is received.

The contents of a data packet are simply the data itself. There are no other fields, so that the most data per packet can be transferred. Framing packets are responsible for ensuring that the correct packet data is received.

11.3.7.6 Response packet

The responses are carried using the same command packet format wrapped with framing packet data. There are 2 types of responses:

- GenericResponse
- GetPropertyResponse

GenericResponse: After the Kinetis Bootloader has processed a command, the bootloader will send a generic response with status and command tag information to the host. The generic response is the last packet in the command protocol sequence. The generic response packet contains the framing packet data and the command packet data (with generic response tag = 0xA0) and a list of parameters (defined in the next section). The parameter count field in the header is always set to 2, for status code and command tag parameters.

Table 11-15. GenericResponse Parameters

| Byte # | Parameter | Description |
|--------|-------------|---|
| 0 - 3 | Status code | The Status codes are errors encountered during the execution of a command by the target (Kinetis Bootloader). If a command succeeds, then |

Table continues on the next page...

Table 11-15. GenericResponse Parameters (continued)

| Byte # | Parameter | Description |
|--------|-------------|--|
| | | a kStatus_Success code is returned. Table 11-37 , Kinetis Bootloader Status Error Codes, lists the status codes returned to the host by the Kinetis Bootloader for KL03 ROM. |
| 4 - 7 | Command tag | The Command tag parameter identifies the response to the command sent by the host. |

GetPropertyResponse: The GetPropertyResponse packet is sent by the target in response to the host query that uses the GetProperty command. The GetPropertyResponse packet contains the framing packet data and the command packet data, with the command/response tag set to a GetPropertyResponse tag value (0xA7).

The parameter count field in the header is set to greater than 1, to always include the status code and one or many property values.

Table 11-16. GetPropertyResponse Parameters

| Byte # | Value | Parameter |
|--------|-------|---|
| 0 - 3 | | Status code |
| 4 - 7 | | Property value |
| ... | | ... |
| | | Can be up to maximum 6 property values, limited to the size of the 32-bit command packet and property type. |

11.3.8 Bootloader Command API

All Kinetis Bootloader command APIs follow the command packet format that is wrapped by the framing packet, as explained in previous sections.

- For a list of commands supported by the Kinetis Bootloader in KL03 ROM, see [Table 11-2](#), Commands supported.
- For a list of status codes returned by the Kinetis Bootloader in KL03 ROM, see [Table 11-37](#), Kinetis Bootloader Status Error Codes.

11.3.8.1 GetProperty command

The GetProperty command is used to query the bootloader about various properties and settings. Each supported property has a unique 32-bit tag associated with it. The tag occupies the first parameter of the command packet. The target returns a GetPropertyResponse packet with the property values for the property identified with the tag in the GetProperty command.

Properties are the defined units of data that can be accessed with the GetProperty or SetProperty commands. Properties may be read-only or read-write. All read-write properties are 32-bit integers, so they can easily be carried in a command parameter.

For a list of properties and their associated 32-bit property tags supported by the Kinetis Bootloader in KL03 ROM, see [Table 11-33](#).

The 32-bit property tag is the only parameter required for GetProperty command.

Table 11-17. Parameters for GetProperty Command

| Byte # | Command |
|--------|--------------|
| 0 - 3 | Property tag |

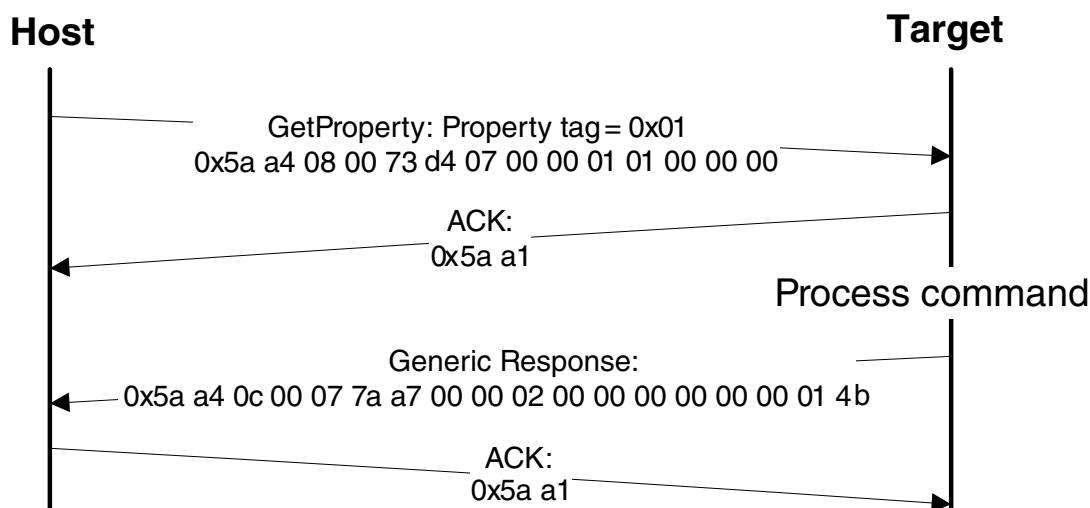


Figure 11-6. Protocol Sequence for GetProperty Command

Table 11-18. GetProperty Command Packet Format (Example)

| GetProperty | Parameter | Value |
|----------------|------------|----------------------------------|
| Framing packet | start byte | 0x5A |
| | packetType | 0xA4, kFramingPacketType_Command |
| | length | 0x08 0x00 |

Table continues on the next page...

Table 11-18. GetProperty Command Packet Format (Example) (continued)

| GetProperty | Parameter | Value |
|----------------|----------------|-----------------------------|
| | crc16 | 0x73 0xD4 |
| Command packet | commandTag | 0x07 – GetProperty |
| | flags | 0x00 |
| | reserved | 0x00 |
| | parameterCount | 0x01 |
| | propertyTag | 0x00000001 - CurrentVersion |

The GetProperty command has no data phase.

Response: In response to a GetProperty command, the target will send a GetPropertyResponse packet with the response tag set to 0xA7. The parameter count indicates the number of parameters sent for the property values, with the first parameter showing status code 0, followed by the property value(s). The next table shows an example of a GetPropertyResponse packet.

Table 11-19. GetProperty Response Packet Format (Example)

| GetPropertyResponse | Parameter | Value |
|---------------------|----------------|----------------------------------|
| Framing packet | start byte | 0x5A |
| | packetType | 0xA4, kFramingPacketType_Command |
| | length | 0x0c 0x00 (12 bytes) |
| | crc16 | 0x07 0x7a |
| Command packet | responseTag | 0xA7 |
| | flags | 0x00 |
| | reserved | 0x00 |
| | parameterCount | 0x02 |
| | status | 0x00000000 |
| | propertyValue | 0x0000014b - CurrentVersion |

11.3.8.2 SetProperty command

The SetProperty command is used to change or alter the values of the properties or options in the Kinetis Bootloader ROM. However, the SetProperty command can only change the value of properties that are writable—see [Table 11-33](#), Properties used by Get/SetProperty Commands. If you try to set a value for a read-only property, then the Kinetis Bootloader will return an error.

The property tag and the new value to set are the 2 parameters required for the SetProperty command.

Table 11-20. Parameters for SetProperty Command

| Byte # | Command |
|--------|----------------|
| 0 - 3 | Property tag |
| 4 - 7 | Property value |

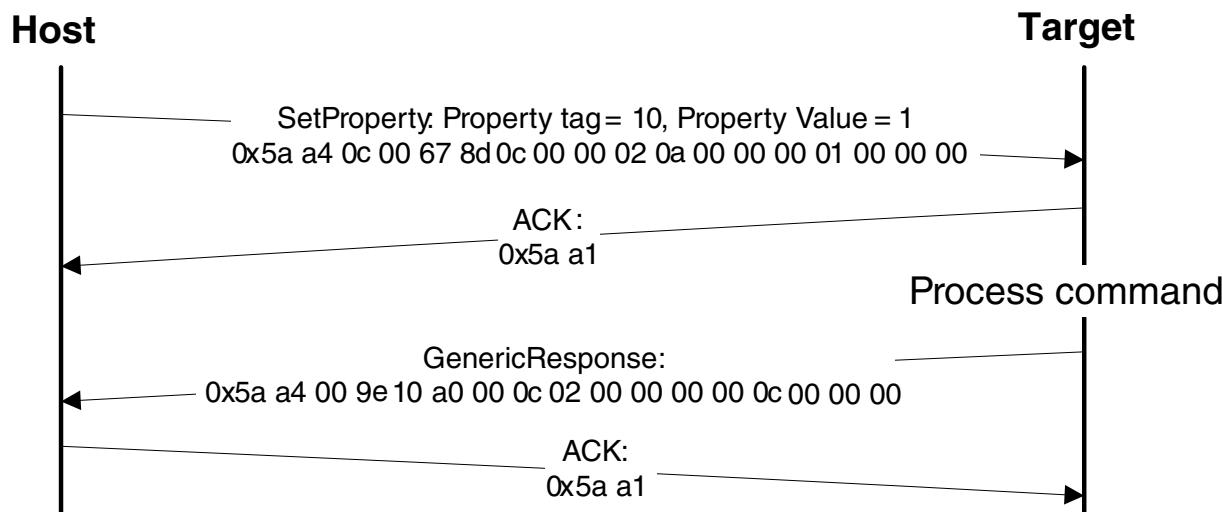


Figure 11-7. Protocol Sequence for SetProperty Command

Table 11-21. SetProperty Command Packet Format (Example)

| SetProperty | Parameter | Value |
|----------------|----------------|---|
| Framing packet | start byte | 0x5A |
| | packetType | 0xA4, kFramingPacketType_Command |
| | length | 0x0C 0x00 |
| | crc16 | 0x67 0x8D |
| Command packet | commandTag | 0x0C – SetProperty with property tag 10 |
| | flags | 0x00 |
| | reserved | 0x00 |
| | parameterCount | 0x02 |
| | propertyTag | 0x0000000A - VerifyWrites |
| | propertyValue | 0x00000001 |

The SetProperty command has no data phase.

Response: The target (Kinetis Bootloader) will return a GenericResponse packet with one of following status codes:

Table 11-22. SetProperty Response Status Codes

| Status Code |
|-------------------------|
| kStatus_Success |
| kStatus_ReadOnly |
| kStatus_UnknownProperty |
| kStatus_InvalidArgument |

11.3.8.3 FlashEraseAll command

The FlashEraseAll command performs an erase of the entire flash memory. If any flash regions are protected, then the FlashEraseAll command will fail and return an error status code. Executing the FlashEraseAll command will release flash security if it (flash security) was enabled, by setting the FTFA_FSEC register. However, the FSEC field of the flash configuration field is erased, so unless it is reprogrammed, the flash security will be re-enabled after the next system reset. The Command tag for FlashEraseAll command is 0x01 set in the commandTag field of the command packet.

The FlashEraseAll command requires no parameters.

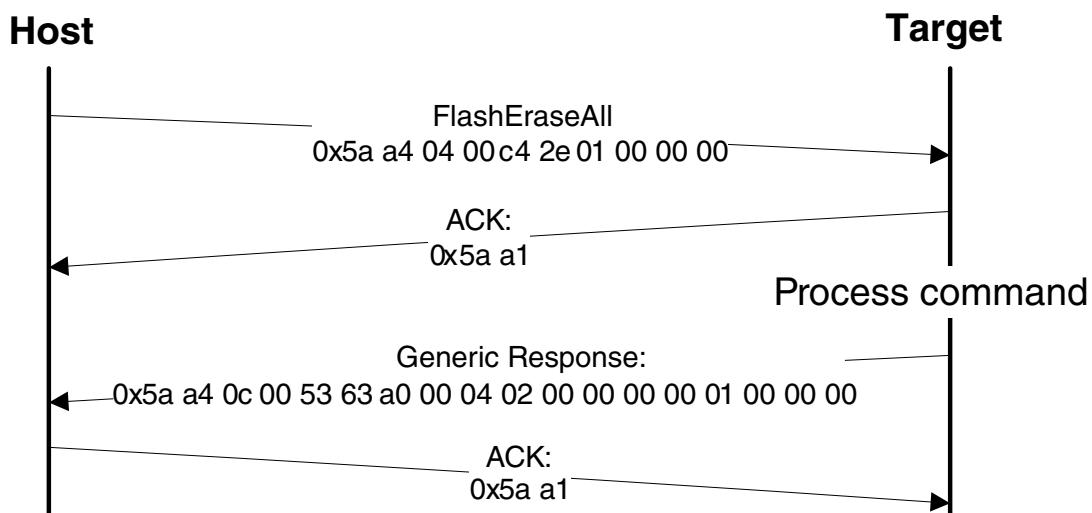


Figure 11-8. Protocol Sequence for FlashEraseAll Command

Table 11-23. FlashEraseAll Command Packet Format (Example)

| FlashEraseAll | Parameter | Value |
|----------------|----------------|----------------------------------|
| Framing packet | start byte | 0x5A |
| | packetType | 0xA4, kFramingPacketType_Command |
| | length | 0x04 0x00 |
| | crc16 | 0xC4 0x2E |
| Command packet | commandTag | 0x01 - FlashEraseAll |
| | flags | 0x00 |
| | reserved | 0x00 |
| | parameterCount | 0x00 |

The FlashEraseAll command has no data phase.

Response: The target (Kinetis Bootloader) will return a GenericResponse packet with status code either set to kStatus_Success for successful execution of the command, or set to an appropriate error status code.

11.3.8.4 FlashEraseRegion command

The FlashEraseRegion command performs an erase of one or more sectors of the flash memory.

The start address and number of bytes are the 2 parameters required for the FlashEraseRegion command. The start and byte count parameters must be 4-byte aligned ([1:0] = 00), or the FlashEraseRegion command will fail and return kStatus_FlashAlignmentError(101). If the region specified does not fit in the flash memory space, the FlashEraseRegion command will fail and return kStatus_FlashAddressError(102). If any part of the region specified is protected, the FlashEraseRegion command will fail and return kStatus_MemoryRangeInvalid(10200).

Table 11-24. Parameters for FlashEraseRegion Command

| Byte # | Parameter |
|--------|---------------|
| 0 - 3 | Start address |
| 4 - 7 | Byte count |

The FlashEraseRegion command has no data phase.

Response: The target (Kinetis Bootloader) will return a GenericResponse packet with one of following error status codes.

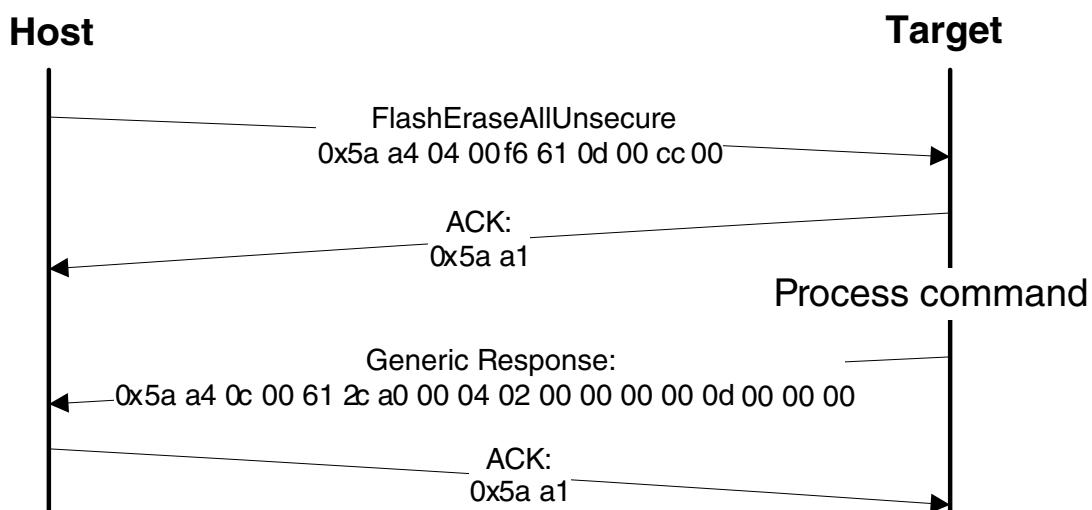
Table 11-25. FlashEraseRegion Response Status Codes

| Status Code |
|--|
| kStatus_Success (0) |
| kStatus_MemoryRangeInvalid (10200) |
| kStatus_FlashAlignmentError (101) |
| kStatus_FlashAddressError (102) |
| kStatus_FlashAccessError (103) |
| kStatus_FlashProtectionViolation (104) |
| kStatus_FlashCommandFailure (105) |

11.3.8.5 FlashEraseAllUnsecure command

The FlashEraseAllUnsecure command performs a mass erase of the flash memory, including protected sectors. Flash security is immediately disabled if it (flash security) was enabled, and the FSEC byte in the flash configuration field at address 0x40C is programmed to 0xFE. However, if the mass erase enable option in the FSEC field is disabled, then the FlashEraseAllUnsecure command will fail.

The FlashEraseAllUnsecure command requires no parameters.

**Figure 11-9. Protocol Sequence for FlashEraseAll Command****Table 11-26. FlashEraseAllUnsecure Command Packet Format (Example)**

| FlashEraseAllUnsecure | Parameter | Value |
|-----------------------|------------|-------|
| Framing packet | start byte | 0x5A |

Table continues on the next page...

Table 11-26. FlashEraseAllUnsecure Command Packet Format (Example) (continued)

| FlashEraseAllUnsecure | Parameter | Value |
|-----------------------|----------------|----------------------------------|
| | packetType | 0xA4, kFramingPacketType_Command |
| | length | 0x04 0x00 |
| | crc16 | 0xF6 0x61 |
| Command packet | commandTag | 0x0D - FlashEraseAllUnsecure |
| | flags | 0x00 |
| | reserved | 0x00 |
| | parameterCount | 0x00 |

The FlashEraseAllUnsecure command has no data phase.

Response: The target (Kinetis Bootloader) will return a GenericResponse packet with status code either set to kStatus_Success for successful execution of the command, or set to an appropriate error status code.

11.3.8.6 WriteMemory command

The WriteMemory command writes data provided in the data phase to a specified range of bytes in memory (flash or RAM). However, if flash protection is enabled, then writes to protected sectors will fail.

Special care must be taken when writing to flash.

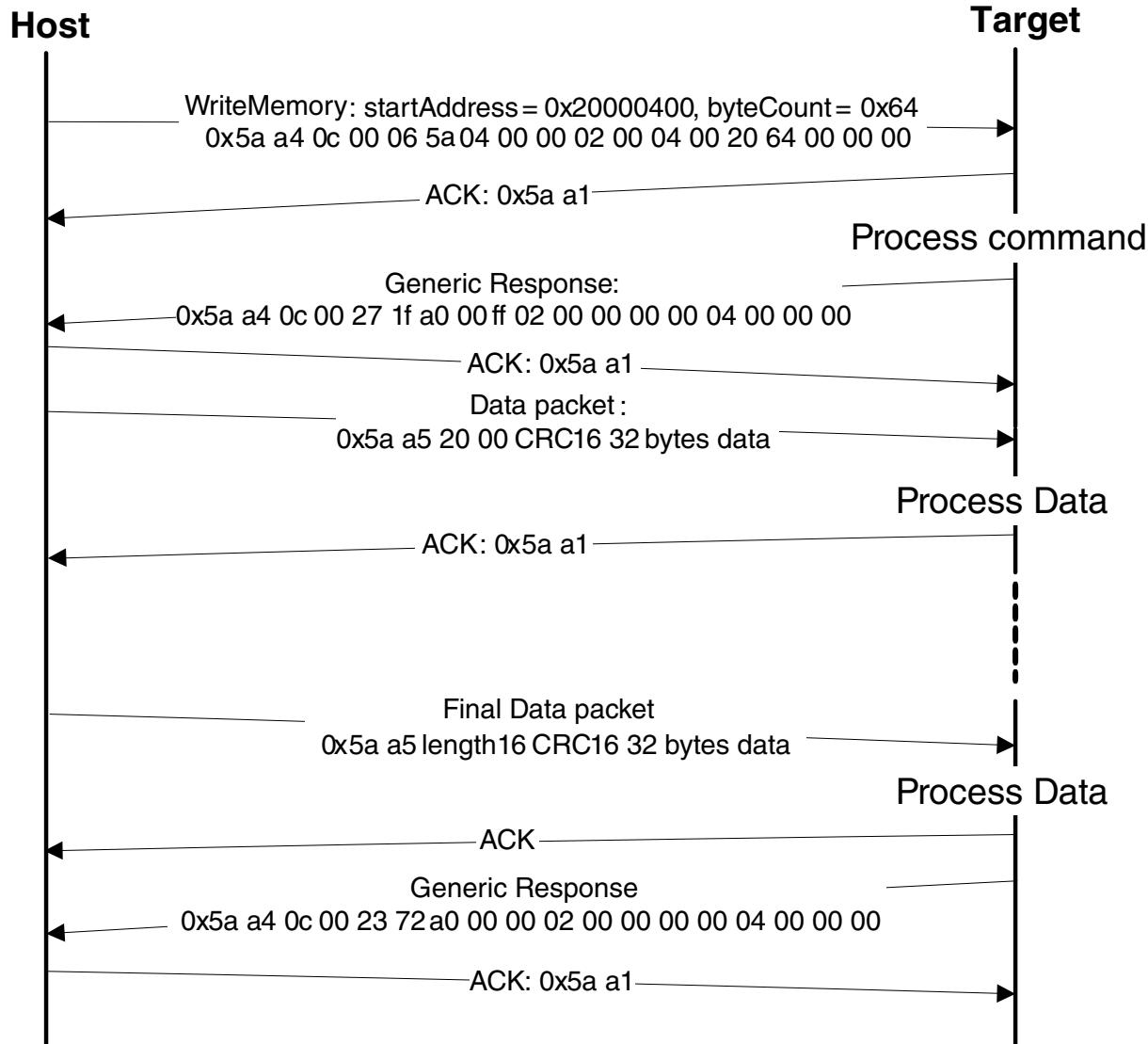
- First, any flash sector written to must have been previously erased with a FlashEraseAll, FlashEraseRegion, or FlashEraseAllUnsecure command.
- Writing to flash requires the start address to be 4-byte aligned ([1:0] = 00).
- The byte count will be rounded up to a multiple of 4, and the trailing bytes will be filled with the flash erase pattern (0xff).
- If the VerifyWrites property is set to true, then writes to flash will also perform a flash verify program operation.

When writing to RAM, the start address need not be aligned, and the data will not be padded.

The start address and number of bytes are the 2 parameters required for WriteMemory command.

Table 11-27. Parameters for WriteMemory Command

| Byte # | Command |
|--------|---------------|
| 0 - 3 | Start address |
| 4 - 7 | Byte count |

**Figure 11-10. Protocol Sequence for WriteMemory Command****Table 11-28. WriteMemory Command Packet Format (Example)**

| WriteMemory | Parameter | Value |
|----------------|----------------|----------------------------------|
| Framing packet | start byte | 0x5A |
| | packetType | 0xA4, kFramingPacketType_Command |
| | length | 0x0C 0x00 |
| | crc16 | 0x06 0x5A |
| Command packet | commandTag | 0x04 - writeMemory |
| | flags | 0x00 |
| | reserved | 0x00 |
| | parameterCount | 0x02 |

Table continues on the next page...

Table 11-28. WriteMemory Command Packet Format (Example) (continued)

| WriteMemory | Parameter | Value |
|-------------|--------------|------------|
| | startAddress | 0x20000400 |
| | byteCount | 0x00000064 |

Data Phase: The WriteMemory command has a data phase; the host will send data packets until the number of bytes of data specified in the byteCount parameter of the WriteMemory command are received by the target.

Response: The target (Kinetis Bootloader) will return a GenericResponse packet with a status code set to kStatus_Success upon successful execution of the command, or to an appropriate error status code.

11.3.8.7 FlashSecurityDisable command

The FlashSecurityDisable command performs the flash security disable operation, by comparing the 8-byte backdoor key (provided in the command) against the backdoor key stored in the flash configuration field (at address 0x400 in the flash).

The backdoor low and high words are the only parameters required for FlashSecurityDisable command.

Table 11-29. Parameters for FlashSecurityDisable Command

| Byte # | Command |
|--------|------------------------|
| 0 - 3 | Backdoor key low word |
| 4 - 7 | Backdoor key high word |

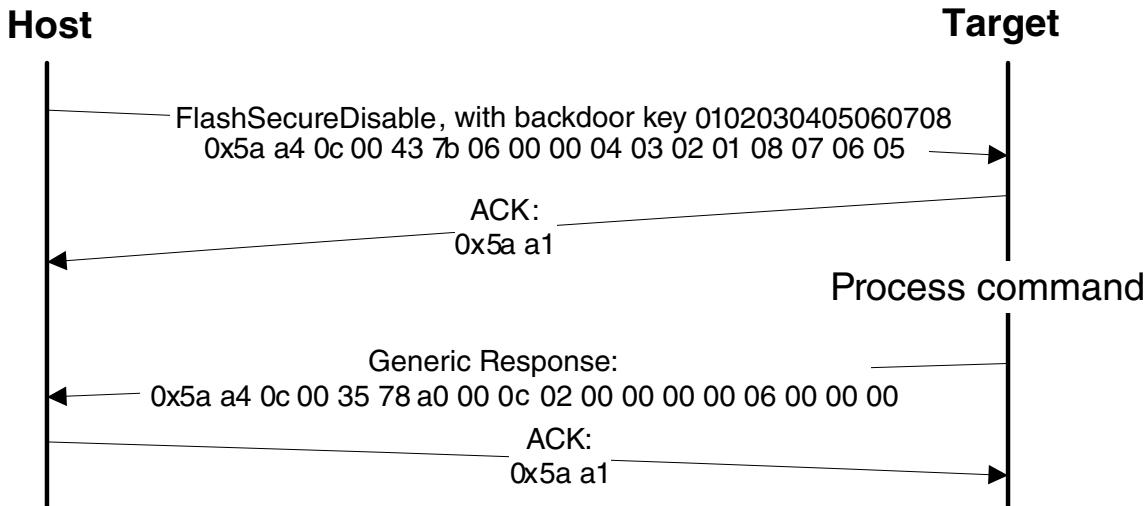


Figure 11-11. Protocol Sequence for FlashSecurityDisable Command

Table 11-30. FlashSecurityDisable Command Packet Format (Example)

| FlashSecurityDisable | Parameter | Value |
|----------------------|------------------|----------------------------------|
| Framing packet | start byte | 0x5A |
| | packetType | 0xA4, kFramingPacketType_Command |
| | length | 0x0C 0x00 |
| | crc16 | 0x43 0x7B |
| Command packet | commandTag | 0x06 - FlashSecurityDisable |
| | flags | 0x00 |
| | reserved | 0x00 |
| | parameterCount | 0x02 |
| | Backdoorkey_low | 0x04 0x03 0x02 0x01 |
| | Backdoorkey_high | 0x08 0x07 0x06 0x05 |

The FlashSecurityDisable command has no data phase.

Response: The target (Kinetis Bootloader) will return a GenericResponse packet with a status code either set to kStatus_Success upon successful execution of the command, or set to an appropriate error status code.

11.3.8.8 Execute command

The execute command results in the bootloader setting the program counter to the code at the provided jump address, R0 to the provided argument, and a Stack pointer to the provided stack pointer address. Prior to the jump, the system is returned to the reset state.

The Jump address, function argument pointer, and stack pointer are the parameters required for the Execute command.

Table 11-31. Parameters for Execute Command

| Byte # | Command |
|--------|-----------------------|
| 0 - 3 | Jump address |
| 4 - 7 | Argument word |
| 8 - 11 | Stack pointer address |

The Execute command has no data phase.

Response: Before executing the Execute command, the target (Kinetis Bootloader) will validate the parameters and return a GenericResponse packet with a status code either set to kStatus_Success or an appropriate error status code.

11.3.8.9 Reset command

The Reset command will result in bootloader resetting the chip.

The Reset command requires no parameters.

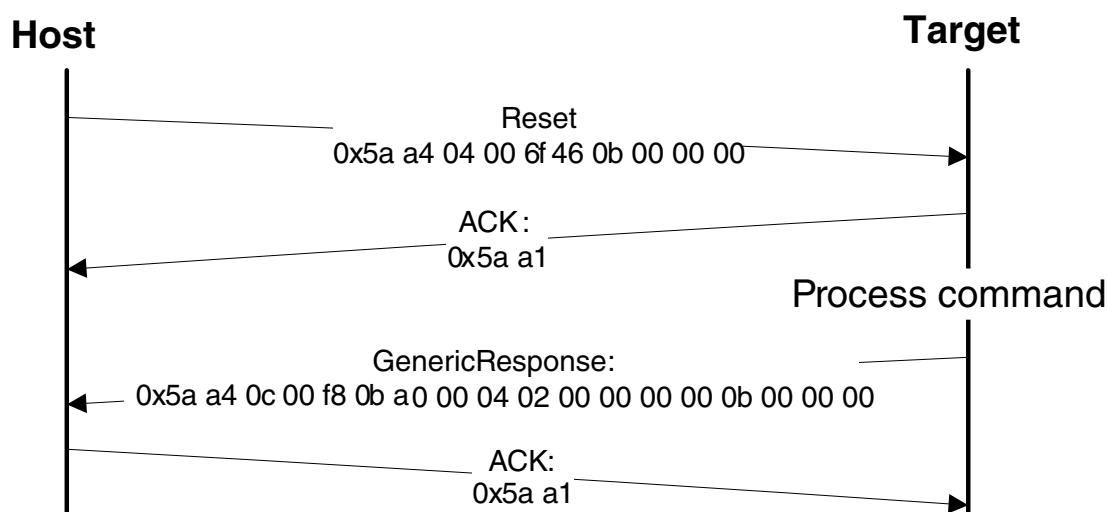


Figure 11-12. Protocol Sequence for Reset Command

Table 11-32. Reset Command Packet Format (Example)

| Reset | Parameter | Value |
|----------------|----------------|----------------------------------|
| Framing packet | start byte | 0x5A |
| | packetType | 0xA4, kFramingPacketType_Command |
| | length | 0x04 0x00 |
| | crc16 | 0x6F 0x46 |
| Command packet | commandTag | 0x0B - reset |
| | flags | 0x00 |
| | reserved | 0x00 |
| | parameterCount | 0x00 |

The Reset command has no data phase.

Response: The target (Kinetis Bootloader) will return a GenericResponse packet with status code set to kStatus_Success, before resetting the chip.

11.3.9 Bootloader Exit state

The Kinetis Bootloader tries to reconfigure the system back to the reset state in the following situations:

- After completion of an Execute command, but before jumping to the specified entry point.
- After a peripheral detection timeout, but before jumping to the application entry point.

In general, all peripherals are reset. However, the application must consider the following exceptions:

- I2C and SPI are restored by writing the control registers to reset values.
- LPUART is not restored.
- Peripheral pin mux is not reset to default, see the pin mux table ([Table 1](#), Kinetis Bootloader Peripheral Pinmux).

Additional considerations:

- If VLPR mode is active during a system boot, then the bootloader will exit VLPR mode on entry (to bootloader operation), and VLPR mode will not be restored after exiting from the bootloader.
- If high speed clocking is selected in the Bootloader Configuration Area, then high speed clocks will be retained after exiting from the bootloader.
- Upon exit from the bootloader, the bootloader leaves global interrupts disabled and restores the VTOR register to its default value (0x0).

NOTE

PORT clock gate, pin mux and peripheral registers are not reset to default values on Bootloader exit.

- Affected PORT clock gates: PORTA, PORTB(SIM_SCGC5_PORTA, SIM_SCGC5_PORTB are enabled)
- Affected pin mux:
 - LPUART0(PTB1, PTB2)
 - I2C0(PTB3, PTB4)
 - SPI0(PTA5, PTA6, PTA7, PTB0)
- Affected peripheral registers:
 - LPUART0 and LPUART0 clock source (SIM_SOPT2_LPUART0SRC = 11b)
 - SPI
 - I2C

You must re-configure the corresponding register to the expected value, instead of relying on the default value.

11.4 Peripherals Supported

This section describes the peripherals (UART, I2C, SPI) supported by the Kinetis ROM Bootloader. To use an interface for bootloader communications, the peripheral must be enabled in the BCA, as shown in [Table 11-3](#). If the BCA is invalid (such as all 0xFF bytes), then all peripherals will be enabled by default.

11.4.1 I2C Peripheral

The Kinetis Bootloader in KL03 ROM supports loading data into flash via the I2C peripheral, where the I2C peripheral serves as the I2C slave. A 7-bit slave address is used during the transfer.

Customizing an I2C slave address is also supported. This feature is enabled if the Bootloader Configuration Area (BCA) (shown in [Table 11-3](#)) is enabled (tag field is filled with ‘kcfg’) and the i2cSlaveAddress field is filled with a value other than 0xFF. Otherwise, 0x10 is used as the default I2C slave address.

The maximum supported I2C baud rate depends on corresponding clock configuration field in the BCA. Typical supported baud rate is 400 kbps with factory settings. Actual supported baud rate may be lower or higher than 400 kbps, depending on the actual value of the clockFlags and the clockDivider fields.

Peripherals Supported

Because the I2C peripheral serves as an I2C slave device, each transfer should be started by the host, and each outgoing packet should be fetched by the host.

- An incoming packet is sent by the host with a selected I2C slave address and the direction bit is set as write.
- An outgoing packet is read by the host with a selected I2C slave address and the direction bit is set as read.
- 0x00 will be sent as the response to host if the target is busy with processing or preparing data.

The following flow charts demonstrate the communication flow of how the host reads ping packet, ACK and response from the target.

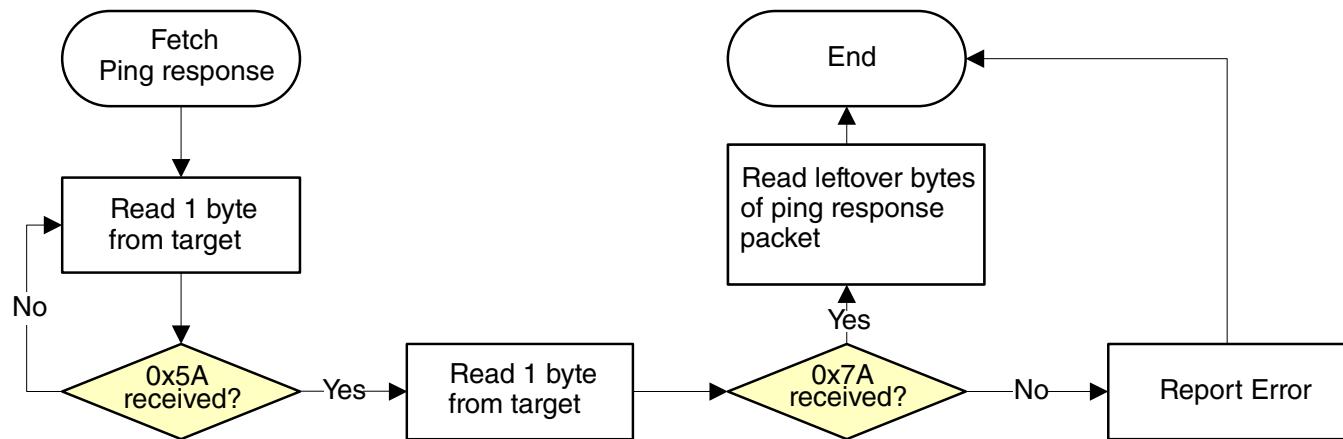


Figure 11-13. Host reads ping response from target via I2C

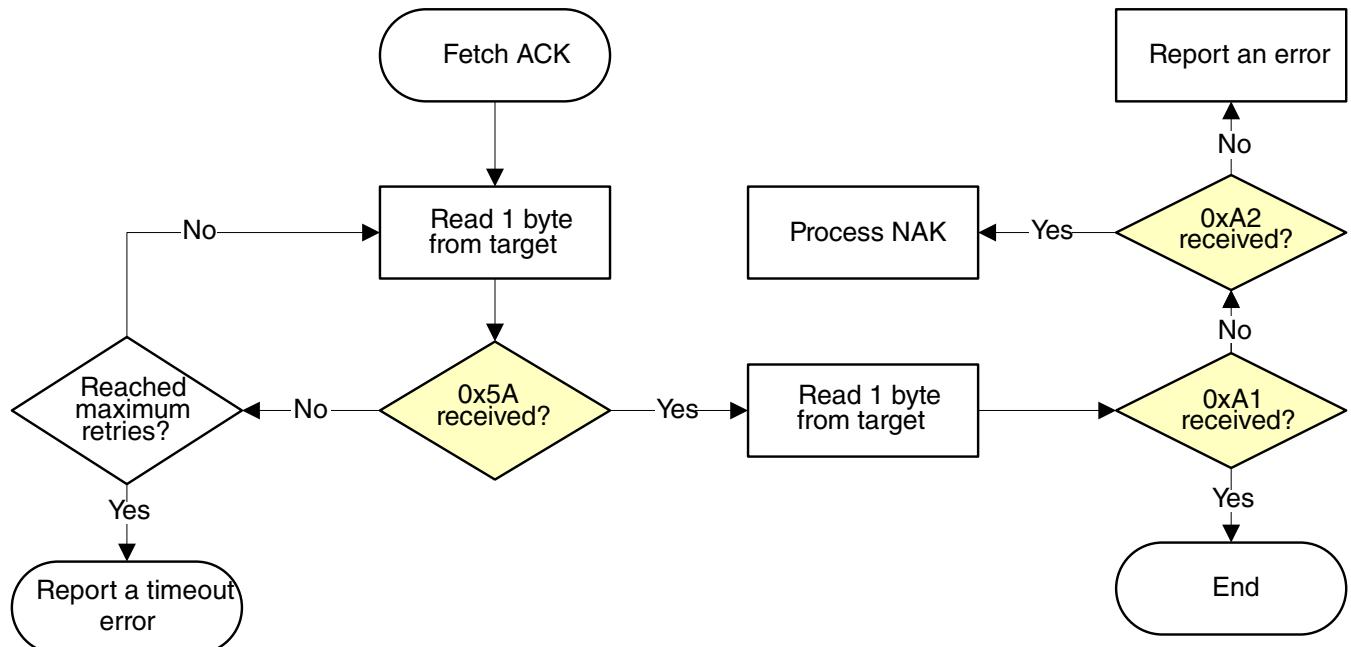


Figure 11-14. Host reads ACK packet from target via I2C

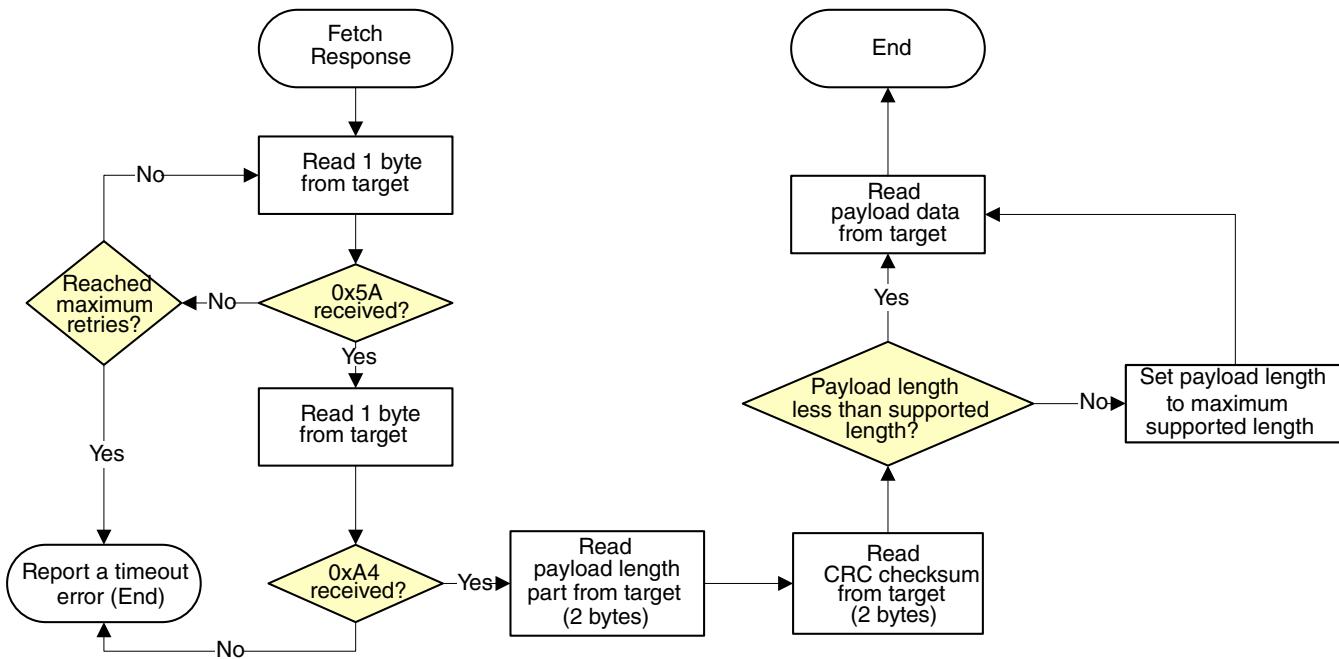


Figure 11-15. Host reads response from target via I2C

11.4.2 SPI Peripheral

The Kinetis Bootloader in KL03 ROM supports loading data into flash via the SPI peripheral, where the SPI peripheral serves as a SPI slave.

Maximum supported baud rate of SPI depends on the clock configuration fields in the Bootloader Configuration Area (BCA) shown in [Table 11-3](#). The typical supported baud rate is 400 kbps with the factory settings. The actual baud rate is lower or higher than 400 kbps, depending on the actual value of the `clockFlags` and `clockDivider` fields in the BCA.

Because the SPI peripheral in KL03 ROM serves as a SPI slave device, each transfer should be started by the host, and each outgoing packet should be fetched by the host.

The transfer on SPI is slightly different from I2C:

- Host will receive 1 byte after it sends out any byte.
- Received bytes should be ignored when host is sending out bytes to target
- Host starts reading bytes by sending 0x00s to target
- The byte 0x00 will be sent as response to host if target is under the following conditions:
 - Processing incoming packet
 - Preparing outgoing data
 - Received invalid data

Peripherals Supported

The following flowcharts demonstrate how the host reads a ping response, an ACK and a command response from target via SPI.

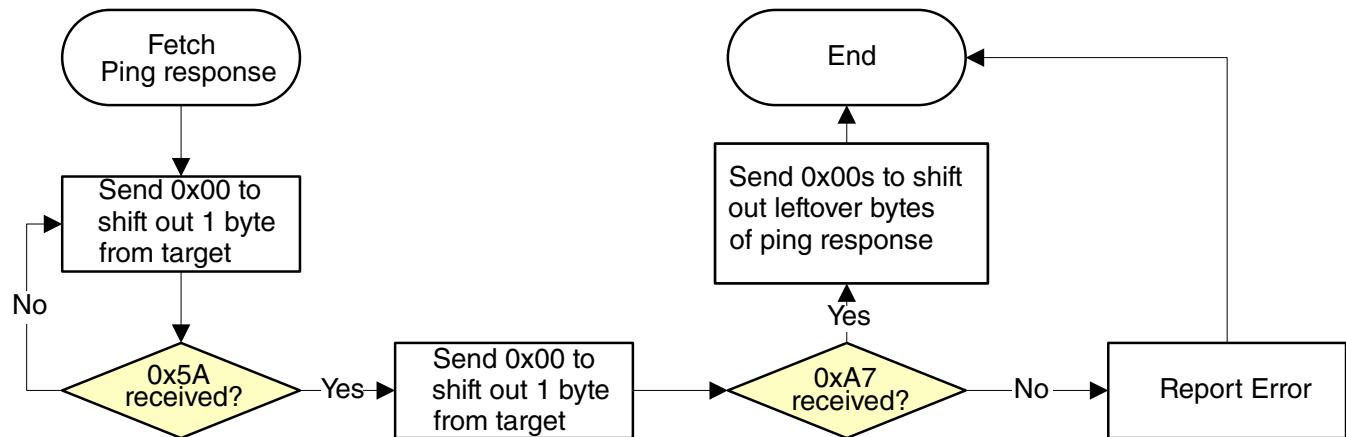


Figure 11-16. Host reads ping packet from target via SPI

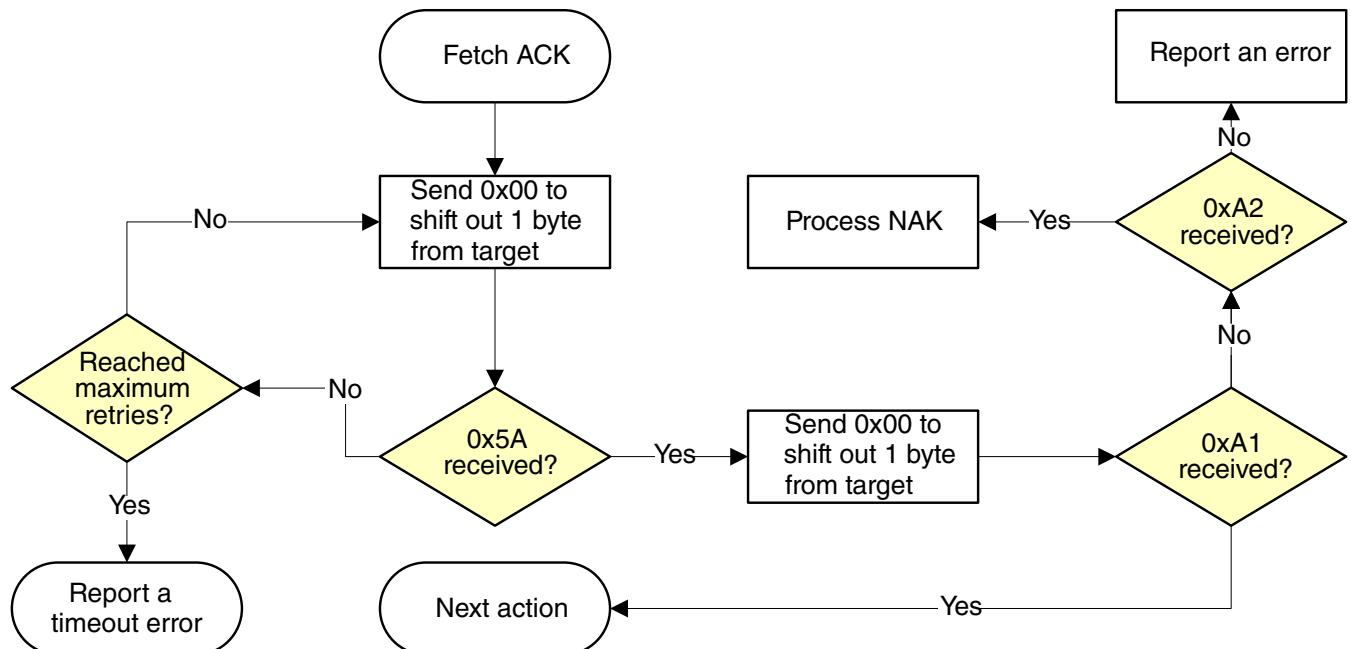


Figure 11-17. Host reads ACK from target via SPI

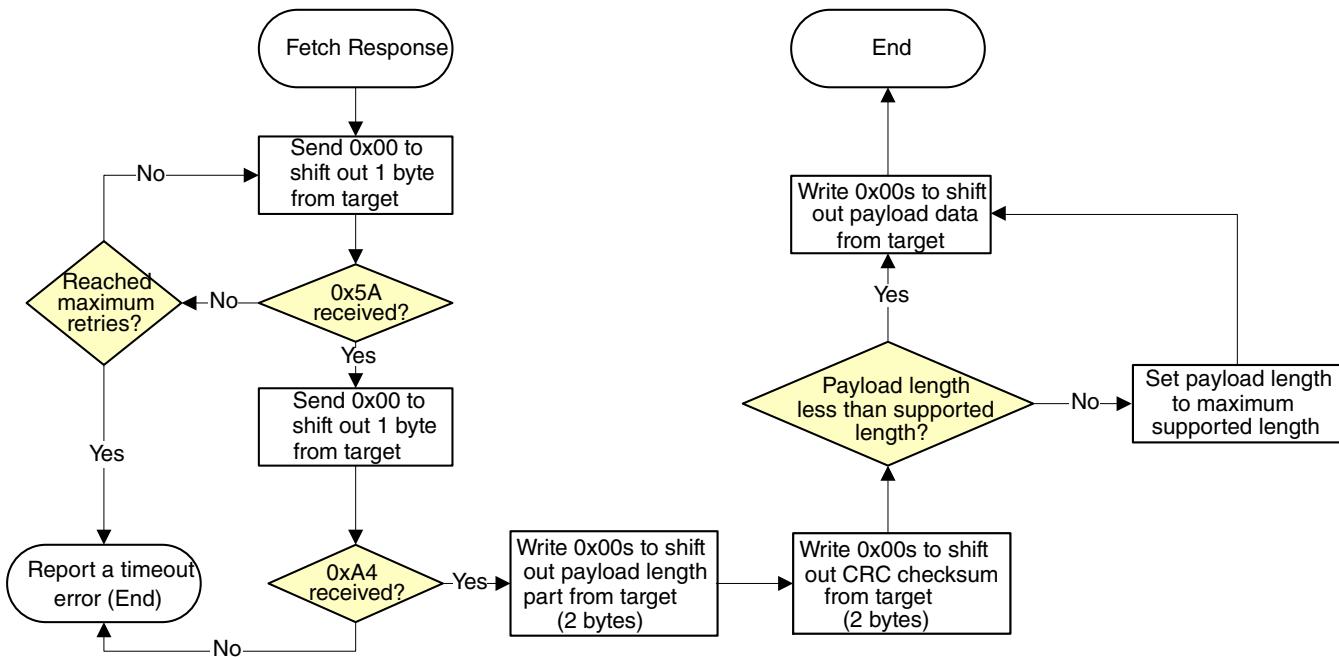


Figure 11-18. Host reads response from target via SPI

11.4.3 LPUART Peripheral

The Kinetis Bootloader integrates an autobaud detection algorithm for the LPUART peripheral, thereby providing flexible baud rate choices.

Autobaud feature: If LPUART n is used to connect to the bootloader, then the LPUART n _RX (PTB2) pin must be kept high and not left floating during the detection phase in order to comply with the autobaud detection algorithm. After the bootloader detects the ping packet (0x5A 0xA6) on LPUART n _RX, the bootloader firmware executes the autobaud sequence. If the baudrate is successfully detected, then the bootloader will send a ping packet response [(0x5A 0xA7), protocol version (4 bytes), protocol version options (2 bytes) and crc16 (2 bytes)] at the detected baudrate. The Kinetis Bootloader then enters a loop, waiting for bootloader commands via the LPUART peripheral.

NOTE

The data bytes of the ping packet must be sent continuously (with no more than 80 ms between bytes) in a fixed LPUART transmission mode (8-bit data, no parity bit and 1 stop bit). If the bytes of the ping packet are sent one-by-one with more than 80 ms delay between them, then the autobaud detection algorithm may calculate an incorrect baud rate. In this case, the autobaud detection state machine should be reset.

Supported baud rates: The baud rate is closely related to the MCU core and system clock frequencies. Typical baud rates supported are 9600, 19200, 38400, and 57600. Of course, to influence the performance of autobaud detection, the clock configuration in BCA can be changed.

Packet transfer: After autobaud detection succeeds, bootloader communications can take place over the LPUART peripheral. The following flow charts show:

- How the host detects an ACK from the target
- How the host detects a ping response from the target
- How the host detects a command response from the target

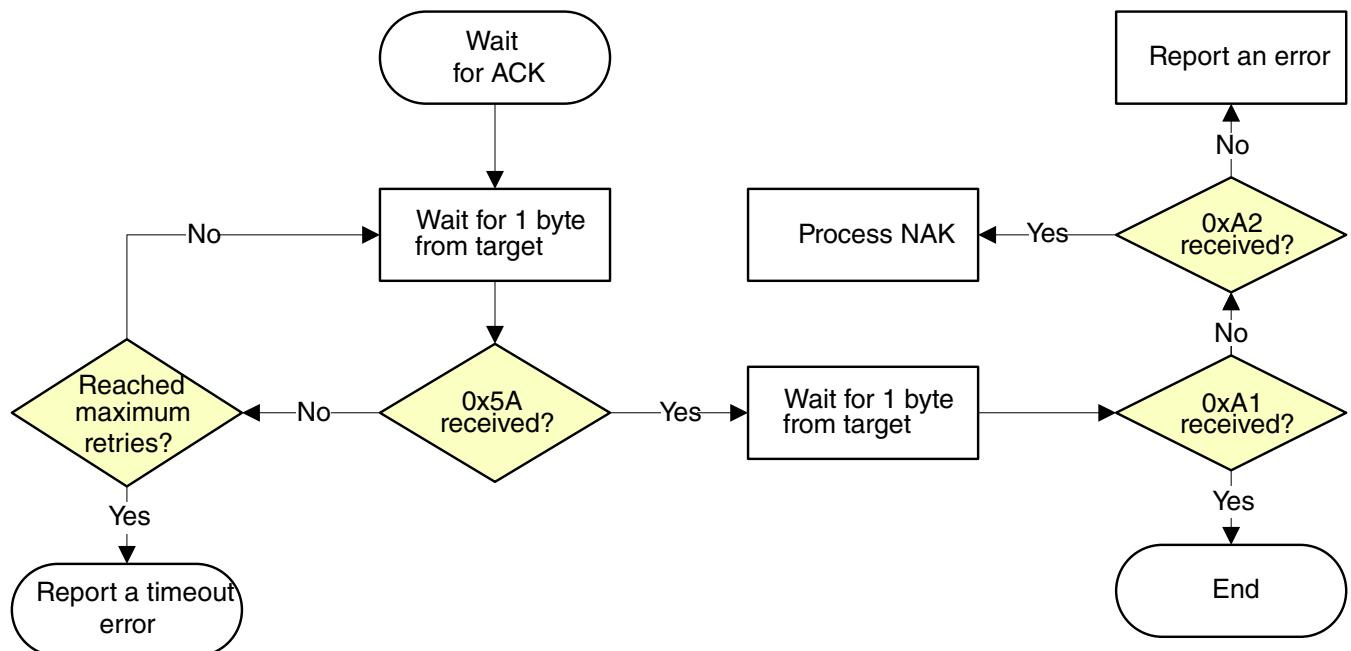


Figure 11-19. Host reads an ACK from target via LPUART

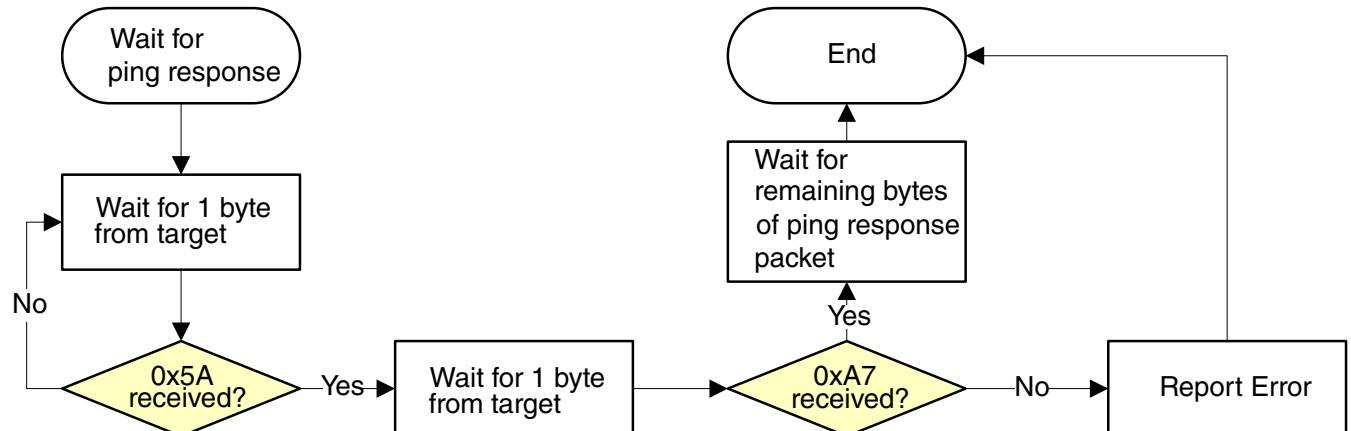


Figure 11-20. Host reads a ping response from target via LPUART

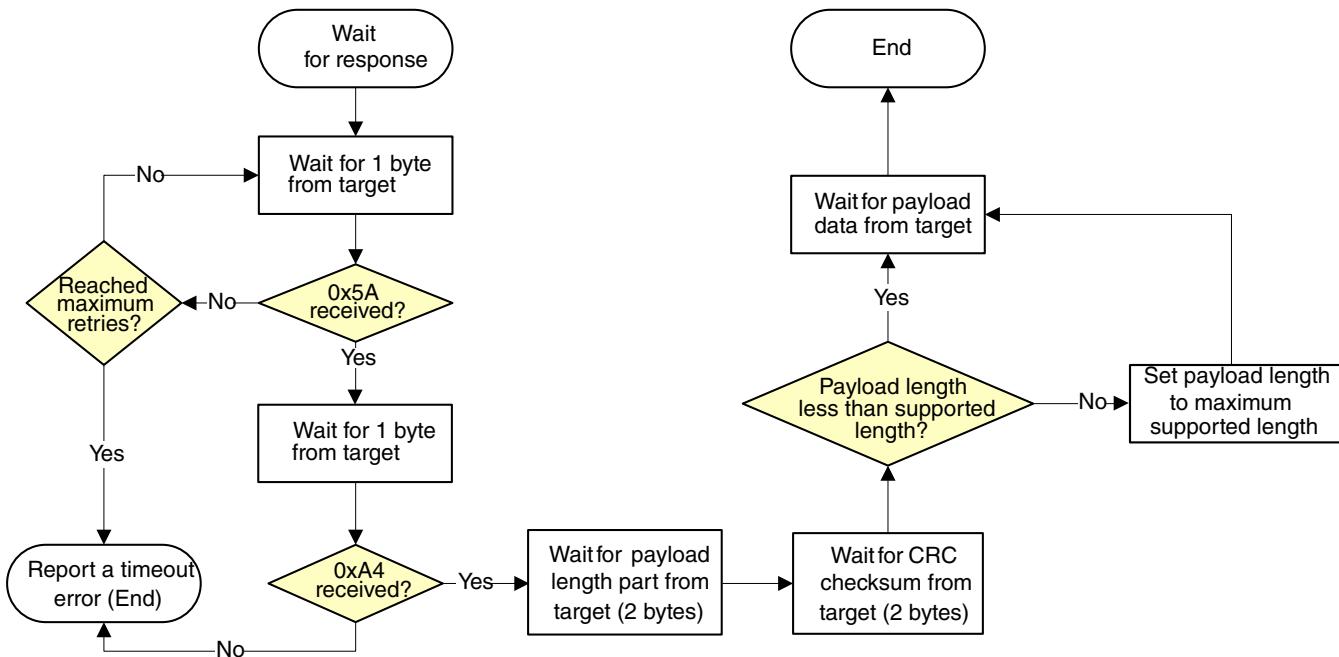


Figure 11-21. Host reads a command response from target via LPUART

11.5 Get/SetProperty Command Properties

This section lists the properties of the GetProperty and SetProperty commands.

Table 11-33. Properties used by Get/GetProperty Commands, sorted by Value

| Property | Writable | Tag Value | Size | Description |
|----------------------|----------|-----------|------|---|
| CurrentVersion | No | 01h | 4 | Current bootloader version. |
| AvailablePeripherals | No | 02h | 4 | The set of peripherals supported on this chip. |
| FlashStartAddress | No | 03h | 4 | Start address of program flash. |
| FlashSizeInBytes | No | 04h | 4 | Size in bytes of program flash. |
| FlashSectorSize | No | 05h | 4 | The size in bytes of one sector of program flash. This is the minimum erase size. |
| FlashBlockCount | No | 06h | 4 | Number of blocks in the flash array. |
| AvailableCommands | No | 07h | 4 | The set of commands supported by the bootloader. |
| VerifyWrites | Yes | 0Ah | 4 | Controls whether the bootloader will verify writes to flash. VerifyWrites feature is enabled by default. 0 - No verification is done. 1 - Enable verification. |
| MaxPacketSize | No | 0Bh | 4 | Maximum supported packet size for the currently active peripheral interface. |

Table continues on the next page...

Table 11-33. Properties used by Get/SetProperty Commands, sorted by Value (continued)

| Property | Writable | Tag Value | Size | Description |
|--------------------|----------|-----------|------|--|
| ReservedRegions | No | 0Ch | 16 | <p>List of memory regions reserved by the bootloader. Returned as value pairs (<start-address-of-region>, <end-address-of-region>).</p> <ul style="list-style-type: none"> If HasDataPhase flag is not set, then the Response packet parameter count indicates the number of pairs. If HasDataPhase flag is set, then the second parameter is the number of bytes in the data phase. |
| ValidateRegions | Yes | 0Dh | 4 | <p>Controls whether the bootloader will validate attempts to write to memory regions (i.e., check if they are reserved before attempting to write). ValidateRegions feature is enabled by default.</p> <p>0 - No validation is done 1 - Enable validation</p> |
| RAMStartAddress | No | 0Eh | 4 | Start address of RAM |
| RAMSizeInBytes | No | 0Fh | 4 | Size in bytes of RAM |
| SystemDeviceId | No | 10h | 4 | Value of the Kinetis System Device Identification register. |
| FlashSecurityState | No | 11h | 4 | <p>Indicates whether Flash security is enabled</p> <p>0 - Flash security is disabled 1 - Flash security is enabled</p> |

11.5.1 Property Definitions

Get/Set property definitions are provided in this section.

11.5.1.1 CurrentVersion Property

The value of this property is a 4-byte structure containing the current version of the bootloader.

Table 11-34. Bit ranges for the version components:

| Bits | [31:24] | [23:16] | [15:8] | [7:0] |
|-------|-------------------|---------------|---------------|----------------|
| Field | Name = 'K' (0x4B) | Major version | Minor version | Bugfix version |

The Kinetis Codebase Version number returned for the KL03 ROM is 1.1.0.

11.5.1.2 AvailablePeripherals Property

The value of this property is a bitfield that lists the peripherals supported by the bootloader and the hardware on which it is running.

Table 11-35. Peripheral bits:

| Bit | [31:7] | [6] | [5] | [4] | [3] | [2] | [1] | [0] |
|------------|----------|---------|---------|---------|-----|-----------|-----------|------|
| Peripheral | Reserved | USB DFU | USB CDC | USB HID | CAN | SPI Slave | I2C Slave | UART |

If the peripheral is available, then the corresponding bit will be set in the property value. All reserved bits must be set to 0.

11.5.1.3 AvailableCommands Property

This property value is a bitfield with set bits indicating the commands enabled in the bootloader. Only commands that can be sent from the host to the target are listed in the bitfield. Response commands such as GenericResponse are excluded.

The bit number that identifies whether a command is present is the command's tag value minus 1. 1 is subtracted from the command tag because the lowest command tag value is 0x01. To get the bit mask for a given command, use this expression:

```
mask = 1 << (tag - 1)
```

Table 11-36. Command bits:

| Bit | [31:16] | [15] | [14] | [13] | [12] | [11] | [10] | [9] | [8] | [7] | [6] | [5] | [4] | [3] | [2] | [1] | [0] |
|---------|----------|----------|----------|----------|-----------------------|-------------|-------|------|---------|---------------|-------------|----------------------|------------|-------------|------------|------------------|---------------|
| Command | Reserved | Reserved | Reserved | Reserved | FlashEraseAllUnsecure | SetProperty | Reset | Call | Execute | ReceiveSBFile | GetProperty | FlashSecurityDisable | FillMemory | WriteMemory | ReadMemory | FlashEraseRegion | FlashEraseAll |

11.6 Kinetis Bootloader Status Error Codes

This section describes the status error codes that the Kinetis Bootloader returns to the host.

Table 11-37. Kinetis Bootloader Status Error Codes, sorted by Value

| Error Code | Value | Description |
|----------------------------------|-------|---|
| kStatus_Success | 0 | Operation succeeded without error. |
| kStatus_Fail | 1 | Operation failed with a generic error. |
| kStatus_ReadOnly | 2 | Requested value cannot be changed because it is read-only. |
| kStatus_OutOfRange | 3 | Requested value is out of range. |
| kStatus_InvalidArgument | 4 | The requested command's argument is undefined. |
| kStatus_Timeout | 5 | A timeout occurred. |
| kStatus_FlashSizeError | 100 | Not used. |
| kStatus_FlashAlignmentError | 101 | Address or length does not meet required alignment. |
| kStatus_FlashAddressError | 102 | Address or length is outside addressable memory. |
| kStatus_FlashAccessError | 103 | The FTFA_FSTAT[ACCERR] bit is set. |
| kStatus_FlashProtectionViolation | 104 | The FTFA_FSTAT[FPVIOL] bit is set. |
| kStatus_FlashCommandFailure | 105 | The FTFA_FSTAT[MGSTAT0] bit is set. |
| kStatus_FlashUnknownProperty | 106 | Unknown Flash property. |
| kStatus_I2C_SlaveTxUnderrun | 200 | I2C Slave TX Underrun error. |
| kStatus_I2C_SlaveRxOverrun | 201 | I2C Slave RX Overrun error. |
| kStatus_I2C_ArbitrationLost | 202 | I2C Arbitration Lost error. |
| kStatus_SPI_SlaveTxUnderrun | 300 | SPI Slave TX Underrun error. |
| kStatus_SPI_SlaveRxOverrun | 301 | SPI Slave RX Overrun error. |
| kStatus_SPI_Timeout | 302 | SPI transfer timed out. |
| kStatus_SPI_Busy | 303 | SPI instance is already busy performing a transfer. |
| kStatus_SPI_NoTransferInProgress | 304 | Attempt to abort a transfer when no transfer was in progress. |
| kStatus_UnknownCommand | 10000 | The requested command value is undefined. |
| kStatus_SecurityViolation | 10001 | Command is disallowed because flash security is enabled. |
| kStatus_AbortDataPhase | 10002 | Abort the data phase early. |
| kStatusMemoryRangeInvalid | 10200 | Memory range conflicts with a protected region. |
| kStatus_UnknownProperty | 10300 | The requested property value is undefined. |
| kStatus_ReadOnlyProperty | 10301 | The requested property value cannot be written. |
| kStatus_InvalidPropertyValue | 10302 | The specified property value is invalid. |

Chapter 12

Port Control and Interrupts (PORT)

12.1 Chip-specific PORT information

This chip has only Port A and Port B.

Not all the pins in the Ports are available. Only the following pins are available and the correspondent registers are functional.

Table 12-1. KL03 pins and registers available

| Port | Pins available | Registers functional |
|--------|-------------------------------|---|
| Port A | PTA0-PTA9, PTA12 | PORTA_PCR[0-9,12] ¹ PORTA_GPCLR PORTA_ISFR |
| Port B | PTB0-PTB7, PTB10-PTB11, PTB13 | PORTB_PCR[0-7,10-11,13] ^{2,3} PORTB_GPCLR PORTB_ISFR |

1. DSE functions only in PORTA_PCR12.
2. PFE functions only in PORTB_PCR5, this field is reserved in all the other registers.
3. DSE functions only in PORTB_PCR0 and PORTB_PCR1.

Not all pins are able to generate interrupt request. See [Table 12-2](#) for detailed information.

12.1.1 GPIO instantiation information

The device includes a number of pins, PTB0, PTB1, and PTA12 , 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 (LPUART, TPM, SPI, I2C, CLK_OUT...etc)

Chip-specific PORT information

The device includes two pins, PTB3 and PTB4, with true open drain setting. These pins have the capability to support 5 V voltage input in 3.3 V systems.

12.1.1.1 Pull devices and directions

The pull devices are enabled out of POR only on $\overline{\text{RESET}}$, $\overline{\text{NMI}}$ and respective SWD signals. Other pins can be enabled by writing to PORTx_PCRn[PE] .

All the pins can be configured as pullup or pulldown through PORTx_PCRn[PS] .

12.1.2 Port control and interrupt summary

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

Table 12-2. Ports summary

| Feature | Port A | Port B |
|--|--|---|
| Pull select control | Yes | Yes |
| Pull select at reset | PTA0=Pull down, Others=Pull up | Pull up |
| Pull enable control | Yes | Yes |
| Pull enable at reset | PTA0/PTA2/ $\overline{\text{RESET_b}}$ =Enabled; Others=Disabled | PTB5=Enabled; Others=Disabled |
| Slew rate enable control | No | No |
| Slew rate enable at reset | PTA2/PTA4/PTA5/PTA6/PTA7/PTA8/ PTA9/PTA12=Disabled; Others=Enabled | PTB0/PTB10/PTB11/PTB13=Disabled; Others=Enabled |
| Passive filter enable control | No | PTB5 only |
| Passive filter enable at reset | Disabled | Disabled |
| Open drain enable control ¹ | No | No |
| Open drain enable at reset | Disabled | Disabled |
| Drive strength enable control | PTA12 only | PTB0/PTB1 only |
| Drive strength enable at reset | Disabled | Disabled |
| Pin mux control | Yes | Yes |
| Pin mux at reset | PTA0/PTA2=ALT3; Others=ALT0 | PTB5=ALT3; Others=ALT0 |
| Lock bit | No | No |
| Interrupt request | PTA0/PTA1/PTA7/PTA12 only | PTB0/PTB1/PTB2/PTB3/PTB4/PTB5/ PTB6/PTB7 only |
| Digital glitch filter | No | No |

1. PTB3/PTB4 are true open drain pad.

12.2 Introduction

12.3 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.

12.3.1 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 request configured per pin
 - Asynchronous wake-up 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 slew rate field supporting fast and slow slew rates on selected pins
 - Individual mux control field supporting analog or pin disabled, GPIO, and up to four chip-specific digital functions
 - Pad configuration fields are functional in all digital pin muxing modes.

12.3.2 Modes of operation

12.3.2.1 Run mode

In Run mode, the PORT operates normally.

12.3.2.2 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.

12.3.2.3 Stop mode

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

12.3.2.4 Debug mode

In Debug mode, PORT operates normally.

12.4 External signal description

The table found here describes the PORT external signal.

Table 12-3. 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.

12.5 Detailed signal description

The table found here contains the detailed signal description for the PORT interface.

Table 12-4. PORT interface—detailed signal description

| Signal | I/O | Description | |
|-------------|-----|---------------------|--|
| PORTx[31:0] | I/O | External interrupt. | |
| | | State meaning | Asserted—pin is logic 1. Negated—pin is logic 0. |
| | | Timing | Assertion—may occur at any time and can assert asynchronously to the system clock. |

Table 12-4. PORT interface—detailed signal description

| Signal | I/O | Description |
|--------|-----|---|
| | | Negation—may occur at any time and can assert asynchronously to the system clock. |

12.6 Memory map and register definition

Any read or write access to the PORT memory space that is outside the valid memory map results in a bus error. All register accesses complete with zero wait states.

PORT memory map

| Absolute address (hex) | Register name | Width (in bits) | Access | Reset value | Section/ page |
|------------------------|--------------------------------------|-----------------|--------|-----------------------------|---------------|
| 4004_9000 | Pin Control Register n (PORTA_PCR0) | 32 | R/W | See section | 12.6.1/154 |
| 4004_9004 | Pin Control Register n (PORTA_PCR1) | 32 | R/W | See section | 12.6.1/154 |
| 4004_9008 | Pin Control Register n (PORTA_PCR2) | 32 | R/W | See section | 12.6.1/154 |
| 4004_900C | Pin Control Register n (PORTA_PCR3) | 32 | R/W | See section | 12.6.1/154 |
| 4004_9010 | Pin Control Register n (PORTA_PCR4) | 32 | R/W | See section | 12.6.1/154 |
| 4004_9014 | Pin Control Register n (PORTA_PCR5) | 32 | R/W | See section | 12.6.1/154 |
| 4004_9018 | Pin Control Register n (PORTA_PCR6) | 32 | R/W | See section | 12.6.1/154 |
| 4004_901C | Pin Control Register n (PORTA_PCR7) | 32 | R/W | See section | 12.6.1/154 |
| 4004_9020 | Pin Control Register n (PORTA_PCR8) | 32 | R/W | See section | 12.6.1/154 |
| 4004_9024 | Pin Control Register n (PORTA_PCR9) | 32 | R/W | See section | 12.6.1/154 |
| 4004_9028 | Pin Control Register n (PORTA_PCR10) | 32 | R/W | See section | 12.6.1/154 |
| 4004_902C | Pin Control Register n (PORTA_PCR11) | 32 | R/W | See section | 12.6.1/154 |
| 4004_9030 | Pin Control Register n (PORTA_PCR12) | 32 | R/W | See section | 12.6.1/154 |
| 4004_9034 | Pin Control Register n (PORTA_PCR13) | 32 | R/W | See section | 12.6.1/154 |
| 4004_9038 | Pin Control Register n (PORTA_PCR14) | 32 | R/W | See section | 12.6.1/154 |
| 4004_903C | Pin Control Register n (PORTA_PCR15) | 32 | R/W | See section | 12.6.1/154 |
| 4004_9040 | Pin Control Register n (PORTA_PCR16) | 32 | R/W | See section | 12.6.1/154 |
| 4004_9044 | Pin Control Register n (PORTA_PCR17) | 32 | R/W | See section | 12.6.1/154 |
| 4004_9048 | Pin Control Register n (PORTA_PCR18) | 32 | R/W | See section | 12.6.1/154 |
| 4004_904C | Pin Control Register n (PORTA_PCR19) | 32 | R/W | See section | 12.6.1/154 |
| 4004_9050 | Pin Control Register n (PORTA_PCR20) | 32 | R/W | See section | 12.6.1/154 |
| 4004_9054 | Pin Control Register n (PORTA_PCR21) | 32 | R/W | See section | 12.6.1/154 |
| 4004_9058 | Pin Control Register n (PORTA_PCR22) | 32 | R/W | See section | 12.6.1/154 |
| 4004_905C | Pin Control Register n (PORTA_PCR23) | 32 | R/W | See section | 12.6.1/154 |
| 4004_9060 | Pin Control Register n (PORTA_PCR24) | 32 | R/W | See section | 12.6.1/154 |
| 4004_9064 | Pin Control Register n (PORTA_PCR25) | 32 | R/W | See section | 12.6.1/154 |

Table continues on the next page...

PORT memory map (continued)

| Absolute address (hex) | Register name | Width (in bits) | Access | Reset value | Section/ page |
|------------------------|--|-----------------|-----------------------|-----------------------------|---------------|
| 4004_9068 | Pin Control Register n (PORTA_PCR26) | 32 | R/W | See section | 12.6.1/154 |
| 4004_906C | Pin Control Register n (PORTA_PCR27) | 32 | R/W | See section | 12.6.1/154 |
| 4004_9070 | Pin Control Register n (PORTA_PCR28) | 32 | R/W | See section | 12.6.1/154 |
| 4004_9074 | Pin Control Register n (PORTA_PCR29) | 32 | R/W | See section | 12.6.1/154 |
| 4004_9078 | Pin Control Register n (PORTA_PCR30) | 32 | R/W | See section | 12.6.1/154 |
| 4004_907C | Pin Control Register n (PORTA_PCR31) | 32 | R/W | See section | 12.6.1/154 |
| 4004_9080 | Global Pin Control Low Register (PORTA_GPCLR) | 32 | W (always reads 0) | 0000_0000h | 12.6.2/156 |
| 4004_9084 | Global Pin Control High Register (PORTA_GPCHR) | 32 | W (always reads 0) | 0000_0000h | 12.6.3/157 |
| 4004_90A0 | Interrupt Status Flag Register (PORTA_ISFR) | 32 | w1c | 0000_0000h | 12.6.4/157 |
| 4004_A000 | Pin Control Register n (PORTB_PCR0) | 32 | R/W | See section | 12.6.1/154 |
| 4004_A004 | Pin Control Register n (PORTB_PCR1) | 32 | R/W | See section | 12.6.1/154 |
| 4004_A008 | Pin Control Register n (PORTB_PCR2) | 32 | R/W | See section | 12.6.1/154 |
| 4004_A00C | Pin Control Register n (PORTB_PCR3) | 32 | R/W | See section | 12.6.1/154 |
| 4004_A010 | Pin Control Register n (PORTB_PCR4) | 32 | R/W | See section | 12.6.1/154 |
| 4004_A014 | Pin Control Register n (PORTB_PCR5) | 32 | R/W | See section | 12.6.1/154 |
| 4004_A018 | Pin Control Register n (PORTB_PCR6) | 32 | R/W | See section | 12.6.1/154 |
| 4004_A01C | Pin Control Register n (PORTB_PCR7) | 32 | R/W | See section | 12.6.1/154 |
| 4004_A020 | Pin Control Register n (PORTB_PCR8) | 32 | R/W | See section | 12.6.1/154 |
| 4004_A024 | Pin Control Register n (PORTB_PCR9) | 32 | R/W | See section | 12.6.1/154 |
| 4004_A028 | Pin Control Register n (PORTB_PCR10) | 32 | R/W | See section | 12.6.1/154 |
| 4004_A02C | Pin Control Register n (PORTB_PCR11) | 32 | R/W | See section | 12.6.1/154 |
| 4004_A030 | Pin Control Register n (PORTB_PCR12) | 32 | R/W | See section | 12.6.1/154 |
| 4004_A034 | Pin Control Register n (PORTB_PCR13) | 32 | R/W | See section | 12.6.1/154 |
| 4004_A038 | Pin Control Register n (PORTB_PCR14) | 32 | R/W | See section | 12.6.1/154 |
| 4004_A03C | Pin Control Register n (PORTB_PCR15) | 32 | R/W | See section | 12.6.1/154 |
| 4004_A040 | Pin Control Register n (PORTB_PCR16) | 32 | R/W | See section | 12.6.1/154 |
| 4004_A044 | Pin Control Register n (PORTB_PCR17) | 32 | R/W | See section | 12.6.1/154 |
| 4004_A048 | Pin Control Register n (PORTB_PCR18) | 32 | R/W | See section | 12.6.1/154 |
| 4004_A04C | Pin Control Register n (PORTB_PCR19) | 32 | R/W | See section | 12.6.1/154 |
| 4004_A050 | Pin Control Register n (PORTB_PCR20) | 32 | R/W | See section | 12.6.1/154 |
| 4004_A054 | Pin Control Register n (PORTB_PCR21) | 32 | R/W | See section | 12.6.1/154 |
| 4004_A058 | Pin Control Register n (PORTB_PCR22) | 32 | R/W | See section | 12.6.1/154 |
| 4004_A05C | Pin Control Register n (PORTB_PCR23) | 32 | R/W | See section | 12.6.1/154 |
| 4004_A060 | Pin Control Register n (PORTB_PCR24) | 32 | R/W | See section | 12.6.1/154 |
| 4004_A064 | Pin Control Register n (PORTB_PCR25) | 32 | R/W | See section | 12.6.1/154 |

Table continues on the next page...

PORT memory map (continued)

| Absolute address (hex) | Register name | Width (in bits) | Access | Reset value | Section/page |
|------------------------|--|-----------------|-----------------------|-----------------------------|--------------|
| 4004_A068 | Pin Control Register n (PORTB_PCR26) | 32 | R/W | See section | 12.6.1/154 |
| 4004_A06C | Pin Control Register n (PORTB_PCR27) | 32 | R/W | See section | 12.6.1/154 |
| 4004_A070 | Pin Control Register n (PORTB_PCR28) | 32 | R/W | See section | 12.6.1/154 |
| 4004_A074 | Pin Control Register n (PORTB_PCR29) | 32 | R/W | See section | 12.6.1/154 |
| 4004_A078 | Pin Control Register n (PORTB_PCR30) | 32 | R/W | See section | 12.6.1/154 |
| 4004_A07C | Pin Control Register n (PORTB_PCR31) | 32 | R/W | See section | 12.6.1/154 |
| 4004_A080 | Global Pin Control Low Register (PORTB_GPCLR) | 32 | W (always reads 0) | 0000_0000h | 12.6.2/156 |
| 4004_A084 | Global Pin Control High Register (PORTB_GPCHR) | 32 | W (always reads 0) | 0000_0000h | 12.6.3/157 |
| 4004_A0A0 | Interrupt Status Flag Register (PORTB_ISFR) | 32 | w1c | 0000_0000h | 12.6.4/157 |

12.6.1 Pin Control Register n (PORTx_PCRn)

NOTE

See the Signal Multiplexing and Pin Assignment 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 unbonded 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 | | | | | | | | ISF | | | | | | | | |
| W | | | | | | | | w1c | | | | | | | | IRQC |
| 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 | | | | | | | | MUX | 0 | DSE | 0 | PFE | 0 | SRE | PE | PS |
| W | | | | | | | | | | | | | | | | |
| Reset | 0 | 0 | 0 | 0 | 0 | * | * | * | 0 | * | 0 | * | 0 | * | * | * |

* Notes:

- MUX field: Varies by port. See Signal Multiplexing and Signal Descriptions chapter for reset values per port.
- DSE field: Varies by port. See the Signal Multiplexing and Signal Descriptions chapter for reset values per port.
- PFE field: Varies by port. See Signal Multiplexing and Signal Descriptions chapter for reset values per port.
- SRE field: Varies by port. See Signal Multiplexing and Signal Descriptions chapter for reset values per port.
- PE field: Varies by port. See Signal Multiplexing and Signal Descriptions chapter for reset values per port.
- PS field: Varies by port. See Signal Multiplexing and Signal Descriptions chapter for reset values per port.

PORTx_PCRn field descriptions

| Field | Description |
|-------------------|--|
| 31–25 Reserved | This field is reserved. This read-only field is reserved and always has the value 0. |
| 24 ISF | Interrupt Status Flag This field is read-only for pins that do not support interrupt generation. The pin interrupt configuration is valid in all digital pin muxing modes. |

Table continues on the next page...

PORTx_PCRn field descriptions (continued)

| Field | Description |
|-------------------|--|
| | <p>0 Configured interrupt is not detected.</p> <p>1 Configured interrupt is detected. The flag remains set until a logic 1 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> |
| 23–20 Reserved | This field is reserved. This read-only field is reserved and always has the value 0. |
| 19–16 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 request as follows:</p> <ul style="list-style-type: none"> 0000 Flag is disabled. 0100 Reserved. 0101 Reserved. 0110 Reserved. 0111 Reserved. 1000 Flag and Interrupt when logic 0. 1001 Flag and Interrupt on rising-edge. 1010 Flag and Interrupt on falling-edge. 1011 Flag and Interrupt on either edge. 1100 Flag and Interrupt when logic 1. 1101 Reserved. 1110 Reserved. 1111 Reserved. |
| 15–11 Reserved | This field is reserved. This read-only field is reserved and always has the value 0. |
| 10–8 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> <ul style="list-style-type: none"> 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 Reserved | This field is reserved. This read-only field is reserved and always has the value 0. |
| 6 DSE | <p>Drive Strength Enable</p> <p>This field 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 continues on the next page...

PORTx_PCRn field descriptions (continued)

| Field | Description |
|------------|--|
| | <p>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.</p> |
| 5 Reserved | This field is reserved. This read-only field is reserved and always has the value 0. |
| 4 PFE | <p>Passive Filter Enable This field is read-only for pins that do not support a configurable passive input filter. Passive filter configuration is valid in all digital pin muxing modes.</p> <p>0 Passive input filter is disabled on the corresponding pin. 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. This read-only field is reserved and always has the value 0. |
| 2 SRE | <p>Slew Rate Enable This field is read-only for pins that do not support a configurable slew rate. 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. 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 This field 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. 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. 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 This bit is read only for pins that do not support a configurable pull resistor direction. 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 PE field is set. 1 Internal pullup resistor is enabled on the corresponding pin, if the corresponding PE field is set.</p> |

12.6.2 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 | 0 | | | | | | | | | | | | | | | | 0 | | | | | | | | | | | | | | | |
| W | GPWE | | | | | | | | | | | | | | | | GPWD | | | | | | | | | | | | | | | |
| 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 | | |

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

12.6.3 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 | | |

PORTx_GPCHR field descriptions

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

12.6.4 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.

Functional description

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 | | | | | | | | | | | | | | | | | ISF | | | | | | | | | | | | | | | | | |
| W | | | | | | | | | | | | | | | | | w1c | | | | | | | | | | | | | | | | | |
| 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 |
|-------|--|
| 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.</p> <p>1 Configured interrupt is detected. The flag remains set until a logic 1 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> |

12.7 Functional description

12.7.1 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 interrupt the CPU 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
- 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, except for SWD_CLK, SWD_IO, RESET and those input/output only peripherals pins (CLKOUT, LPUART0_RX, TPM_CLKIN1, etc). For example, if an I²C 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, output buffer enable, input buffer enable, and passive filter enable.

The configuration of each Pin Control register is retained when the PORT module is disabled.

Whenever a pin is configured in any digital pin muxing mode, the input buffer for that pin is enabled allowing the pin state to be read via the corresponding GPIO Port Data Input Register (GPIO_PDIR) or allowing a pin interrupt to be generated. If a pin is ever floating when its input buffer is enabled, then this can cause an increase in power consumption and must be avoided. A pin can be floating due to an input pin that is not connected or an output pin that has tristated (output buffer is disabled).

Enabling the internal pull resistor (or implementing an external pull resistor) will ensure a pin does not float when its input buffer is enabled; note that the internal pull resistor is automatically disabled whenever the output buffer is enabled allowing the Pull Enable bit to remain set. Configuring the Pin Muxing mode to disabled or analog will disable the pin's input buffer and results in the lowest power consumption.

12.7.2 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 16 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 0.

12.7.3 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

Functional description

The interrupt status flag is set when the configured edge or level is detected on 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 1 to the ISF flag in either the PORT_ISFR or PORT_PCRn registers.

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 wake-up signal to exit the Low-Power mode.

Chapter 13

General-Purpose Input/Output (GPIO)

13.1 Chip-specific GPIO information

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 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.

13.2 Introduction

The general-purpose input and output (GPIO) module is accessible via the peripheral bus and also communicates to the processor core via a zero wait state interface (IOPORT) 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.

13.2.1 Features

- Features of the GPIO module include:
 - Port Data Input register visible in all digital pin-multiplexing modes
 - Port Data Output register with corresponding set/clear/toggle registers

- Port Data Direction register
- Zero wait state access to GPIO registers through IOPORT

NOTE

The GPIO module is clocked by system clock.

13.2.2 Modes of operation

The following table depicts different modes of operation and the behavior of the GPIO module in these modes.

Table 13-1. 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. |

13.2.3 GPIO signal descriptions

Table 13-2. GPIO signal descriptions

| GPIO signal descriptions | Description | I/O |
|--------------------------|------------------------------|-----|
| PORTE31–PORTE0 | General-purpose input/output | 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.

13.2.3.1 Detailed signal description

Table 13-3. GPIO interface-detailed signal descriptions

| Signal | I/O | Description | |
|----------------|-----|------------------------------|--|
| PORTE31–PORTE0 | I/O | General-purpose input/output | |
| PORTE31–PORTE0 | | State meaning | Asserted: The pin is logic 1. Deasserted: The pin is logic 0. |

Table continues on the next page...

Table 13-3. GPIO interface-detailed signal descriptions (continued)

| Signal | I/O | Description | |
|--------|-----|-------------|--|
| | | Timing | 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. 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. |

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.

13.3 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.

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 | 13.3.1/164 |
| 400F_F004 | Port Set Output Register (GPIOA_PSOR) | 32 | W (always reads 0) | 0000_0000h | 13.3.2/165 |
| 400F_F008 | Port Clear Output Register (GPIOA_PCOR) | 32 | W (always reads 0) | 0000_0000h | 13.3.3/165 |
| 400F_F00C | Port Toggle Output Register (GPIOA_PTOR) | 32 | W (always reads 0) | 0000_0000h | 13.3.4/166 |
| 400F_F010 | Port Data Input Register (GPIOA_PDIR) | 32 | R | 0000_0000h | 13.3.5/166 |
| 400F_F014 | Port Data Direction Register (GPIOA_PDDR) | 32 | R/W | 0000_0000h | 13.3.6/167 |
| 400F_F040 | Port Data Output Register (GPIOB_PDOR) | 32 | R/W | 0000_0000h | 13.3.1/164 |

Table continues on the next page...

GPIO memory map (continued)

| Absolute address (hex) | Register name | Width (in bits) | Access | Reset value | Section/page |
|------------------------|---|-----------------|-----------------------|-------------|----------------------------|
| 400F_F044 | Port Set Output Register (GPIOB_PSOR) | 32 | W (always reads 0) | 0000_0000h | 13.3.2/165 |
| 400F_F048 | Port Clear Output Register (GPIOB_PCOR) | 32 | W (always reads 0) | 0000_0000h | 13.3.3/165 |
| 400F_F04C | Port Toggle Output Register (GPIOB_PTOR) | 32 | W (always reads 0) | 0000_0000h | 13.3.4/166 |
| 400F_F050 | Port Data Input Register (GPIOB_PDIR) | 32 | R | 0000_0000h | 13.3.5/166 |
| 400F_F054 | Port Data Direction Register (GPIOB_PDDR) | 32 | R/W | 0000_0000h | 13.3.6/167 |

13.3.1 Port Data Output Register (GPIO_x_PDOR)

This register configures the logic levels that are driven on each general-purpose output pins.

NOTE

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

Address: Base address + 0h offset

GPIOx PDOR field descriptions

| Field | Description |
|-------|--|
| PDO | <p>Port Data Output</p> <p>Register bits for unbonded pins return a undefined value when read.</p> <p>0 Logic level 0 is driven on pin, provided pin is configured for general-purpose output.</p> <p>1 Logic level 1 is driven on pin, provided pin is configured for general-purpose output.</p> |

13.3.2 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 |
|-------|---|
| 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"> 0 Corresponding bit in PDORn does not change. 1 Corresponding bit in PDORn is set to logic 1. |

13.3.3 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 |
|-------|---|
| 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"> 0 Corresponding bit in PDORn does not change. 1 Corresponding bit in PDORn is cleared to logic 0. |

13.3.4 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 | | |

GPIOx_PTOR field descriptions

| Field | Description |
|-------|--|
| 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"> 0 Corresponding bit in PDORn does not change. 1 Corresponding bit in PDORn is set to the inverse of its existing logic state. |

13.3.5 Port Data Input Register (GPIOx_PDIR)

NOTE

Do not modify pin configuration registers associated with pins not available in your selected package. All unbonded 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 | | |

GPIOx_PDIR field descriptions

| Field | Description |
|-------|---|
| 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"> 0 Pin logic level is logic 0, or is not configured for use by digital function. 1 Pin logic level is logic 1. |

13.3.6 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 |
|-------|--|
| PDD | Port Data Direction Configures individual port pins for input or output. 0 Pin is configured as general-purpose input, for the GPIO function. 1 Pin is configured as general-purpose output, for the GPIO function. |

13.4 FGPIO memory map and register definition

The GPIO registers are also aliased to the IOPORT interface on the Cortex-M0+ from address 0xF800_0000.

Accesses via the IOPORT interface occur in parallel with any instruction fetches and will therefore complete in a single cycle. This aliased Fast GPIO memory map is called FGPIO.

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.

FGPIO memory map

| Absolute address (hex) | Register name | Width (in bits) | Access | Reset value | Section/page |
|------------------------|--|-----------------|--------------------|-------------|----------------------------|
| F800_0000 | Port Data Output Register (FGPIOA_PDOR) | 32 | R/W | 0000_0000h | 13.4.1/168 |
| F800_0004 | Port Set Output Register (FGPIOA_PSOR) | 32 | W (always reads 0) | 0000_0000h | 13.4.2/169 |
| F800_0008 | Port Clear Output Register (FGPIOA_PCOR) | 32 | W (always reads 0) | 0000_0000h | 13.4.3/169 |

Table continues on the next page...

FGPIO memory map (continued)

| Absolute address (hex) | Register name | Width (in bits) | Access | Reset value | Section/page |
|------------------------|--|-----------------|-----------------------|-------------|----------------------------|
| F800_000C | Port Toggle Output Register (FGPIOA_PTOR) | 32 | W (always reads 0) | 0000_0000h | 13.4.4/170 |
| F800_0010 | Port Data Input Register (FGPIOA_PDIR) | 32 | R | 0000_0000h | 13.4.5/170 |
| F800_0014 | Port Data Direction Register (FGPIOA_PDDR) | 32 | R/W | 0000_0000h | 13.4.6/171 |
| F800_0040 | Port Data Output Register (FGPIOB_PDOR) | 32 | R/W | 0000_0000h | 13.4.1/168 |
| F800_0044 | Port Set Output Register (FGPIOB_PSOR) | 32 | W (always reads 0) | 0000_0000h | 13.4.2/169 |
| F800_0048 | Port Clear Output Register (FGPIOB_PCOR) | 32 | W (always reads 0) | 0000_0000h | 13.4.3/169 |
| F800_004C | Port Toggle Output Register (FGPIOB_PTOR) | 32 | W (always reads 0) | 0000_0000h | 13.4.4/170 |
| F800_0050 | Port Data Input Register (FGPIOB_PDIR) | 32 | R | 0000_0000h | 13.4.5/170 |
| F800_0054 | Port Data Direction Register (FGPIOB_PDDR) | 32 | R/W | 0000_0000h | 13.4.6/171 |

13.4.1 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

FGPIOx PDOR field descriptions

| Field | Description |
|-------|---|
| PDO | <p>Port Data Output</p> <p>Unimplemented pins for a particular device read as zero.</p> <p>0 Logic level 0 is driven on pin, provided pin is configured for general-purpose output.</p> <p>1 Logic level 1 is driven on pin, provided pin is configured for general-purpose output.</p> |

13.4.2 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 | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| 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_PSOR field descriptions

| Field | Description |
|-------|---|
| 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"> 0 Corresponding bit in PDORn does not change. 1 Corresponding bit in PDORn is set to logic 1. |

13.4.3 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 | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| 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_PCOR field descriptions

| Field | Description |
|-------|---|
| 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"> 0 Corresponding bit in PDORn does not change. 1 Corresponding bit in PDORn is cleared to logic 0. |

13.4.4 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 | | |

FGPIOx_PTOR field descriptions

| Field | Description |
|-------|--|
| 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"> 0 Corresponding bit in PDORn does not change. 1 Corresponding bit in PDORn is set to the inverse of its existing logic state. |

13.4.5 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 | | |

FGPIOx_PDIR field descriptions

| Field | Description |
|-------|---|
| 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"> 0 Pin logic level is logic 0, or is not configured for use by digital function. 1 Pin logic level is logic 1. |

13.4.6 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 |
|-------|--|
| PDD | Port Data Direction Configures individual port pins for input or output. 0 Pin is configured as general-purpose input, for the GPIO function. 1 Pin is configured as general-purpose output, for the GPIO function. |

13.5 Functional description

13.5.1 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.

13.5.2 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.

13.5.3 IOPORT

The GPIO registers are also aliased to the IOPORT interface on the Cortex-M0+ from address 0xF800_0000. Accesses via the IOPORT interface occur in parallel with any instruction fetches and will therefore complete in a single cycle.

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 14

System Integration Module (SIM)

14.1 Chip-specific COP information

14.2 COP clocks

The multiple clock inputs for the COP are:

- 1 kHz clock
- bus clock
- 8 MHz or 2 MHz internal reference clock
- external crystal

14.3 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), the 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 SIM_COPC[COPT].

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.

SIM_COPC[COPCLKS] and SIM_COPCTRL[COPCLKSEL] select the timeout duration and clock source used for the COP timer. The clock source options are either the bus clock, 8 MHz/2 MHz IRC, external crystal or an internal 1 kHz clock source. With each clock source, the associated timeouts are controlled by SIM_COPC[COPT] and SIM_COPC[COPCLKS]. The following table summarizes the control functions of SIM_COPCTRL[COPCLKS] and SIM_COPC[COPCLKSEL] and SIM_COPC[COPT] fields. The COP watchdog defaults to operation from the 1 kHz clock source and the longest timeout is 2^{10} cycles.

Table 14-1. COP configuration options

| Control bits | | | Clock source | COP window opens (SIM_COPC[COPW]=1) | COP overflow count |
|----------------------|--------------------|-----------------|--------------|--|----------------------------|
| SIM_COPC[COPCLK SEL] | SIM_COPC[COPCLK S] | SIM_COPC[COP T] | | | |
| N/A | N/A | 00 | N/A | N/A | COP is disabled. |
| 00 | 0 | 01 | 1 kHz | N/A | 2^5 cycles (32ms) |
| | 1 | | | 6,144 cycles | 2^{13} cycles (8192ms) |
| 00 | 0 | 10 | 1 kHz | N/A | 2^8 cycles (256ms) |
| | 1 | | | 49,152 cycles | 2^{16} cycles (65536ms) |
| 00 | 0 | 11 | 1 kHz | N/A | 2^{10} cycles (1024 ms) |
| | 1 | | | 196,608 cycles | 2^{18} cycles (262144ms) |
| 01 | 0 | 01 | 8/2 MHz | N/A | 2^5 cycles |
| | 1 | | | 6,144 cycles | 2^{13} cycles |
| 01 | 0 | 10 | 8/2 MHz | N/A | 2^8 cycles |
| | 1 | | | 49,152 cycles | 2^{16} cycles |
| 01 | 0 | 11 | 8/2 MHz | N/A | 2^{10} cycles |
| | 1 | | | 196,608 cycles | 2^{18} cycles |
| 10 | 0 | 01 | crystal | N/A | 2^5 cycles |
| | 1 | | | 6,144 cycles | 2^{13} cycles |
| 10 | 0 | 10 | crystal | N/A | 2^8 cycles |
| | 1 | | | 49,152 cycles | 2^{16} cycles |
| 10 | 0 | 11 | crystal | N/A | 2^{10} cycles |
| | 1 | | | 196,608 cycles | 2^{18} cycles |
| 01 | 0 | 01 | bus | N/A | 2^5 cycles |
| | 1 | | | 6,144 cycles | 2^{13} cycles |
| 01 | 0 | 10 | bus | N/A | 2^8 cycles |
| | 1 | | | 49,152 cycles | 2^{16} cycles |
| 01 | 0 | 11 | bus | N/A | 2^{10} cycles |
| | 1 | | | 196,608 cycles | 2^{18} cycles |

After the long timeout (COPCLKS = 1) is selected, windowed COP operation is available by setting SIM_COPC[COPW]. In this mode, writes to SIM_SRVCOP 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 short timeout (COPCLKS =0) is selected, windowed COP operation is not available.

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

The write to SIM_SRVCOP 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 selected clock is not the 1 kHz clock source, the COP counter does not increment while the microcontroller is in Debug mode or while the system is in Stop (including VLPS) mode. The COP counter resumes when the microcontroller exits Debug or Stop mode.

The COP counter is re-initialized to 0 upon entry to either Debug mode or Stop (including VLPS) mode. The counter begins from 0 upon exit from Debug mode or Stop mode.

The COP counter can also be configured to continue incrementing during Debug mode or Stop (including VLPS) mode if either COPDBGEN or COPSTPEN are set respectively. When the selected clock is the bus clock and COPSTEN bit is set, the COP counter cannot increment during Stop modes, however the COP counter is not reset to 0.

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 reinitialized and enabled as for any reset.

14.4 Introduction

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

14.4.1 Features

- System clocking configuration
 - System clock divide values
 - Architectural clock gating control
 - ERCLK32K clock selection
 - LPUART0 and TPM clock selection
- Flash and System RAM size configuration
- TPM external clock and input capture selection
- LPUART receive/transmit source selection/configuration

14.5 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 is located at a different base address than the other SIM registers.

4004_8008-4004_800B and 4004_8040-4004_8043 are reserved, any access to these addresses will not cause exception.

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 | 0000_0000h | 14.5.1/177 |
| 4004_8004 | System Options Register 2 (SIM_SOPT2) | 32 | R/W | 0000_0000h | 14.5.2/178 |
| 4004_800C | System Options Register 4 (SIM_SOPT4) | 32 | R/W | 0000_0000h | 14.5.3/180 |
| 4004_8010 | System Options Register 5 (SIM_SOPT5) | 32 | R/W | 0000_0000h | 14.5.4/181 |
| 4004_8018 | System Options Register 7 (SIM_SOPT7) | 32 | R/W | 0000_0000h | 14.5.5/182 |
| 4004_8024 | System Device Identification Register (SIM_SDID) | 32 | R | See section | 14.5.6/184 |
| 4004_8034 | System Clock Gating Control Register 4 (SIM_SCGC4) | 32 | R/W | F000_0030h | 14.5.7/186 |

Table continues on the next page...

SIM memory map (continued)

| Absolute address (hex) | Register name | Width (in bits) | Access | Reset value | Section/page |
|------------------------|---|-----------------|--------|-----------------------------|-----------------------------|
| 4004_8038 | System Clock Gating Control Register 5 (SIM_SCGC5) | 32 | R/W | 0000_0182h | 14.5.8/187 |
| 4004_803C | System Clock Gating Control Register 6 (SIM_SCGC6) | 32 | R/W | 0000_0001h | 14.5.9/189 |
| 4004_8044 | System Clock Divider Register 1 (SIM_CLKDIV1) | 32 | R/W | See section | 14.5.10/190 |
| 4004_804C | Flash Configuration Register 1 (SIM_FCFG1) | 32 | R/W | See section | 14.5.11/192 |
| 4004_8050 | Flash Configuration Register 2 (SIM_FCFG2) | 32 | R | See section | 14.5.12/193 |
| 4004_8058 | Unique Identification Register Mid-High (SIM_UIDMH) | 32 | R | See section | 14.5.13/194 |
| 4004_805C | Unique Identification Register Mid Low (SIM_UIDML) | 32 | R | See section | 14.5.14/194 |
| 4004_8060 | Unique Identification Register Low (SIM_UIDL) | 32 | R | See section | 14.5.15/195 |
| 4004_8100 | COP Control Register (SIM_COPC) | 32 | R/W | 0000_000Ch | 14.5.16/195 |
| 4004_8104 | Service COP (SIM_SRVCOP) | 32 | W | 0000_0000h | 14.5.17/197 |

14.5.1 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 | | | | | | | | 0 | | | | | | | | OSC32KSEL | OSC32KOUT |
| 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_SOPT1 field descriptions

| Field | Description |
|--------------------|---|
| 31–20 Reserved | This field is reserved. This read-only field is reserved and always has the value 0. |
| 19–18 OSC32KSEL | 32K Oscillator Clock Select Selects the 32 kHz clock source (ERCLK32K) for RTC and LPTMR. This field is reset only on POR/LVD. 00 System oscillator (OSC32KCLK) 01 Reserved 10 RTC_CLKIN 11 LPO 1kHz |

Table continues on the next page...

SIM_SOPT1 field descriptions (continued)

| Field | Description |
|--------------------|---|
| 17–16 OSC32KOUT | <p>32K oscillator clock output</p> <p>Outputs the ERCLK32K on the selected pin in all modes of operation (including VLLS and System Reset), overriding the existing pin mux configuration for that pin. This field is reset only on POR/LVD.</p> <p>00 ERCLK32K is not output. 01 ERCLK32K is output on PTB13. 10 Reserved. 11 Reserved.</p> |
| Reserved | This field is reserved. This read-only field is reserved and always has the value 0. |

14.5.2 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 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| W | | | | | LPUART0SRC | | TPMSRC | | | | | | | | | |
| 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 | | | | | | | | | | | | CLKOUTSEL | | 0 | | |
| W | | | | | | | | | | | | RTCCLKOUTS_E | | | | |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

SIM_SOPT2 field descriptions

| Field | Description |
|-------------------|---|
| 31–30 Reserved | This field is reserved. This read-only field is reserved and always has the value 0. |
| 29–28 Reserved | This field is reserved. This read-only field is reserved and always has the value 0. |

Table continues on the next page...

SIM_SOPT2 field descriptions (continued)

| Field | Description |
|---------------------|---|
| 27–26 LPUART0SRC | LPUART0 Clock Source Select Selects the clock source for the LPUART0 transmit and receive clock. 00 Clock disabled 01 IRC48M clock 10 OSCERCLK clock 11 MCGIRCLK clock |
| 25–24 TPMSRC | TPM Clock Source Select Selects the clock source for the TPM counter clock 00 Clock disabled 01 IRC48M clock 10 OSCERCLK clock 11 MCGIRCLK clock |
| 23–22 Reserved | This field is reserved. This read-only field is reserved and always has the value 0. |
| 21–18 Reserved | This field is reserved. This read-only field is reserved and always has the value 0. |
| 17–16 Reserved | This field is reserved. This read-only field is reserved and always has the value 0. |
| 15–8 Reserved | This field is reserved. This read-only field is reserved and always has the value 0. |
| 7–5 CLKOUTSEL | CLKOUT select Selects the clock to output on the CLKOUT pin. 000 Reserved 001 Reserved 010 Bus clock 011 LPO clock (1 kHz) 100 LIRC_CLK 101 Reserved 110 OSCERCLK 111 IRC48M clock (IRC48M clock can be output to PAD only when chip VDD is 2.7-3.6 V) |
| 4 RTCCLKOUTSEL | RTC Clock Out Select Selects either the RTC 1 Hz clock or the OSC clock to be output on the RTC_CLKOUT pin. 0 RTC 1 Hz clock is output on the RTC_CLKOUT pin. 1 OSCERCLK clock is output on the RTC_CLKOUT pin. |
| Reserved | This field is reserved. This read-only field is reserved and always has the value 0. |

14.5.3 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 | | | 0 | TPM1CLKSEL | TPM0CLKSEL | | 0 | | 0 | 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 Reserved | This field is reserved. This read-only field is reserved and always has the value 0. |
| 26 Reserved | This field is reserved. This read-only field is reserved and always has the value 0. |
| 25 TPM1CLKSEL | TPM1 External Clock Pin Select Selects the external pin used to drive the clock to the TPM1 module. NOTE: The selected pin must also be configured for the TPM external clock function through the appropriate pin control register in the port control module. 0 TPM1 external clock driven by TPM_CLKIN0 pin. 1 TPM1 external clock driven by TPM_CLKIN1 pin. |
| 24 TPM0CLKSEL | TPM0 External Clock Pin Select Selects the external pin used to drive the clock to the TPM0 module. NOTE: The selected pin must also be configured for the TPM external clock function through the appropriate pin control register in the port control module. 0 TPM0 external clock driven by TPM_CLKIN0 pin. 1 TPM0 external clock driven by TPM_CLKIN1 pin. |
| 23–21 Reserved | This field is reserved. This read-only field is reserved and always has the value 0. |
| 20 Reserved | This field is reserved. This read-only field is reserved and always has the value 0. |
| 19 Reserved | This field is reserved. This read-only field is reserved and always has the value 0. |

Table continues on the next page...

SIM_SOPT4 field descriptions (continued)

| Field | Description |
|------------------|--|
| 18 TPM1CH0SRC | <p>TPM1 Channel 0 Input Capture Source Select</p> <p>Selects the source for TPM1 channel 0 input capture.</p> <p>NOTE: When TPM1 is not in input capture mode, clear this field.</p> <p>0 TPM1_CH0 signal 1 CMP0 output</p> |
| Reserved | This field is reserved. This read-only field is reserved and always has the value 0. |

14.5.4 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 | 0 | 0 | LPUART0ODE | |
| 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 | LPUART0RXS RC | 0 | LPUART0TXS RC |
| 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 Reserved | This field is reserved. This read-only field is reserved and always has the value 0. |
| 19 Reserved | This field is reserved. This read-only field is reserved and always has the value 0. |
| 18 Reserved | This field is reserved. This read-only field is reserved and always has the value 0. |
| 17 Reserved | This field is reserved. This read-only field is reserved and always has the value 0. |
| 16 LPUART0ODE | LPUART0 Open Drain Enable 0 Open drain is disabled on LPUART0. 1 Open drain is enabled on LPUART0. |

Table continues on the next page...

SIM_SOPT5 field descriptions (continued)

| Field | Description |
|-------------------|---|
| 15–7 Reserved | This field is reserved. This read-only field is reserved and always has the value 0. |
| 6–4 Reserved | This field is reserved. This read-only field is reserved and always has the value 0. |
| 3 Reserved | This field is reserved. This read-only field is reserved and always has the value 0. |
| 2 LPUART0RXSRC | LPUART0 Receive Data Source Select Selects the source for the LPUART0 receive data. 0 LPUART_RX pin 1 CMP0 output |
| 1 Reserved | This field is reserved. This read-only field is reserved and always has the value 0. |
| 0 LPUART0TXSRC | LPUART0 transmit data source select Selects the source for the LPUART0 transmit data. 0 LPUART0_TX pin 1 LPUART0_TX pin modulated with TPM1 channel 0 output |

14.5.5 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 | | | | | | | | | ADC0ALTTRGE N | | 0 | | ADC0PRETRGS EL | | | ADC0TRGSEL |
| 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 |

SIM_SOPT7 field descriptions

| Field | Description |
|--------------------|---|
| 31–8 Reserved | This field is reserved. This read-only field is reserved and always has the value 0. |
| 7 ADC0ALTTRGEN | ADC0 Alternate Trigger Enable Enables alternative conversion triggers for ADC0. 0 ADC ADHWT trigger comes from TPM1 channel 0 and channel1. Prior to the assertion of TPM1 channel 0, a pre-trigger pulse will be sent to ADHWTSA to initiate an ADC acquisition using ADCx_SC1A configuration and store ADC conversion in ADCx_RA Register. Prior to the assertion of TPM1 channel 1 a pre-trigger pulse will be sent to ADHWTSB to initiate an ADC acquisition using ADCx_SC1B configuration and store ADC conversion in ADCx_RB Register. 1 ADC ADHWT trigger comes from a peripheral event selected by ADC0TRGSEL bits. ADC0PRETRGSEL bit will select the optional ADHWTSA or ADHWTSB select lines for choosing the ADCx_SC1x config and ADCx_Rx result register to store the ADC conversion. |
| 6–5 Reserved | This field is reserved. This read-only field is reserved and always has the value 0. |
| 4 ADC0PRETRGSEL | ADC0 Pretrigger Select Selects the ADC0 pre-trigger source when alternative triggers are enabled through ADC0ALTTRGEN. NOTE: The ADC0PRETRGSEL function is ignored if ADC0ALTTRGEN = 0. 0 Pre-trigger ADHDWTSA is selected, thus ADC0 will use ADC0_SC1A configuration for the next ADC conversion and store the result in ADC0_RA register. 1 Pre-trigger ADHDWTSB is selected, thus ADC0 will use ADC0_SC1B configuration for the next ADC conversion and store the result in ADC0_RB register. |
| ADC0TRGSEL | ADC0 Trigger Select Selects 1 of 16 peripherals to initiate an ADC conversion via the ADHWDT input, when ADC0ALTTRGEN =1, else is ignored by ADC0. 0000 External trigger pin input (EXTRG_IN) 0001 CMP0 output 0010 Reserved 0011 Reserved 0100 Reserved 0101 Reserved 0110 Reserved 0111 Reserved 1000 TPM0 overflow 1001 TPM1 overflow 1010 Reserved 1011 Reserved 1100 RTC alarm 1101 RTC seconds 1110 LPTMR0 trigger 1111 Reserved |

14.5.6 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 | 1 | 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 FAMID | Kinetis family ID Specifies the Kinetis family of the device. 0000 KL0x Family (low end) 0001 KL1x Family (basic) 0010 KL2x Family (USB) 0011 KL3x Family (Segment LCD) 0100 KL4x Family (USB and Segment LCD) |
| 27–24 SUBFAMID | Kinetis Sub-Family ID Specifies the Kinetis sub-family of the device. 0010 KLx2 Subfamily 0011 KLx3 Subfamily 0100 KLx4 Subfamily 0101 KLx5 Subfamily 0110 KLx6 Subfamily 0111 KLx7 Subfamily |
| 23–20 SERIESID | Kinetis Series ID Specifies the Kinetis family of the device. 0001 KL family |
| 19–16 SRAMSIZE | System SRAM Size Specifies the size of the System SRAM 0000 0.5 KB 0001 1 KB 0010 2 KB 0011 4 KB |

Table continues on the next page...

SIM_SDID field descriptions (continued)

| Field | Description |
|--------------|--|
| | 0100 8 KB 0101 16 KB 0110 32 KB 0111 64 KB |
| 15–12 REVID | Device Revision Number Specifies the silicon implementation number for the device. |
| 11–7 DIEID | Device Die Number Specifies the silicon implementation number for the device. |
| 6–4 Reserved | This field is reserved. This read-only field is reserved and always has the value 0. |
| PINID | Pincount Identification Specifies the pincount of the device. 0000 16-pin 0001 24-pin 0010 32-pin 0011 36-pin 0100 48-pin 0101 64-pin 0110 80-pin 0111 Reserved 1000 100-pin 1001 Reserved 1010 Reserved 1011 Custom pinout (WLCSP) 1100 Reserved 1101 Reserved 1110 Reserved 1111 Reserved |

14.5.7 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 | | | | | | | | | 0 | | 0 | | | 0 | | 0 |
| W | | | | | | | | | | SPI0 | | VREF | CMP0 | | | |
| 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 | 0 | 0 | 0 | 0 | 0 | 0 | | 1 | | | 0 | | |
| W | | | | | | | | | I2C0 | | | | | | | |
| 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 Reserved | This field is reserved. This read-only field is reserved and always has the value 1. |
| 27–24 Reserved | This field is reserved. This read-only field is reserved and always has the value 0. |
| 23 Reserved | This field is reserved. This read-only field is reserved and always has the value 0. |
| 22 SPI0 | SPI0 Clock Gate Control Controls the clock gate to the SPI0 module. 0 Clock disabled 1 Clock enabled |
| 21 Reserved | This field is reserved. This read-only field is reserved and always has the value 0. |
| 20 VREF | VREF Clock Gate Control Controls the clock gate to the VREF module. 0 Clock disabled 1 Clock enabled |
| 19 CMP0 | Comparator Clock Gate Control Controls the clock gate to the comparator module. 0 Clock disabled 1 Clock enabled |
| 18 Reserved | This field is reserved. This read-only field is reserved and always has the value 0. |

Table continues on the next page...

SIM_SCGC4 field descriptions (continued)

| Field | Description |
|-------------------|--|
| 17–14 Reserved | This field is reserved. This read-only field is reserved and always has the value 0. |
| 13 Reserved | This field is reserved. This read-only field is reserved and always has the value 0. |
| 12 Reserved | This field is reserved. This read-only field is reserved and always has the value 0. |
| 11 Reserved | This field is reserved. This read-only field is reserved and always has the value 0. |
| 10 Reserved | This field is reserved. This read-only field is reserved and always has the value 0. |
| 9 Reserved | This field is reserved. This read-only field is reserved and always has the value 0. |
| 8 Reserved | This field is reserved. This read-only field is reserved and always has the value 0. |
| 7 Reserved | This field is reserved. This read-only field is reserved and always has the value 0. |
| 6 I2C0 | I2C0 Clock Gate Control Controls the clock gate to the I ² C0 module. 0 Clock disabled 1 Clock enabled |
| 5–4 Reserved | This field is reserved. This read-only field is reserved and always has the value 1. |
| Reserved | This field is reserved. This read-only field is reserved and always has the value 0. |

14.5.8 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 | | LPUART0 | 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 | | 0 | | | | | | 1 | 0 | 0 | | 0 | | 1 | |
| W | | | | | | PORTB | PORTA | | | | | | | | | LPTMR |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 1 | 1 | 0 | 0 | 0 | 0 | 0 | 1 | 0 |

SIM_SCGC5 field descriptions

| Field | Description |
|-------------------|--|
| 31 Reserved | This field is reserved. This read-only field is reserved and always has the value 0. |
| 30–22 Reserved | This field is reserved. This read-only field is reserved and always has the value 0. |
| 21 Reserved | This field is reserved. This read-only field is reserved and always has the value 0. |
| 20 LPUART0 | LPUART0 Clock Gate Control This bit controls the clock gate to the LPUART0 module. 0 Clock disabled 1 Clock enabled |
| 19 Reserved | This field is reserved. This read-only field is reserved and always has the value 0. |
| 18–14 Reserved | This field is reserved. This read-only field is reserved and always has the value 0. |
| 13–11 Reserved | This field is reserved. This read-only field is reserved and always has the value 0. |
| 10 PORTB | Port B Clock Gate Control Controls the clock gate to the Port B module. 0 Clock disabled 1 Clock enabled |
| 9 PORTA | Port A Clock Gate Control Controls the clock gate to the Port A module. 0 Clock disabled 1 Clock enabled |
| 8–7 Reserved | This field is reserved. This read-only field is reserved and always has the value 1. |
| 6 Reserved | This field is reserved. This read-only field is reserved and always has the value 0. |
| 5 Reserved | This field is reserved. This read-only field is reserved and always has the value 0. |
| 4–2 Reserved | This field is reserved. This read-only field is reserved and always has the value 0. |
| 1 Reserved | This field is reserved. This read-only field is reserved and always has the value 1. |
| 0 LPTMR | Low Power Timer Access Control Controls software access to the Low Power Timer module. 0 Access disabled 1 Access enabled |

14.5.9 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 | 0 | 0 | | 0 | | 0 | | TPM0 | 0 | | 0 | | 0 | | 0 | |
| W | | | RTC | | ADC0 | | TPM1 | | | | | | | | | |
| 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 | | | |
| W | | | | | | | | | | | | | | | | FTF |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 1 |

SIM_SCGC6 field descriptions

| Field | Description |
|----------------|---|
| 31 Reserved | This field is reserved. This read-only field is reserved and always has the value 0. |
| 30 Reserved | This field is reserved. This read-only field is reserved and always has the value 0. |
| 29 RTC | RTC Access Control Controls software access and interrupts to the RTC module. 0 Access and interrupts disabled 1 Access and interrupts enabled |
| 28 Reserved | This field is reserved. This read-only field is reserved and always has the value 0. |
| 27 ADC0 | ADC0 Clock Gate Control Controls the clock gate to the ADC0 module. 0 Clock disabled 1 Clock enabled |
| 26 Reserved | This field is reserved. This read-only field is reserved and always has the value 0. |
| 25 TPM1 | TPM1 Clock Gate Control Controls the clock gate to the TPM1 module. 0 Clock disabled 1 Clock enabled |
| 24 TPM0 | TPM0 Clock Gate Control Controls the clock gate to the TPM0 module. |

Table continues on the next page...

SIM_SCGC6 field descriptions (continued)

| Field | Description |
|----------------|--|
| | 0 Clock disabled 1 Clock enabled |
| 23 Reserved | This field is reserved. This read-only field is reserved and always has the value 0. |
| 22–19 Reserved | This field is reserved. This read-only field is reserved and always has the value 0. |
| 18 Reserved | This field is reserved. This read-only field is reserved and always has the value 0. |
| 17–16 Reserved | This field is reserved. This read-only field is reserved and always has the value 0. |
| 15 Reserved | This field is reserved. This read-only field is reserved and always has the value 0. |
| 14–2 Reserved | This field is reserved. This read-only field is reserved and always has the value 0. |
| 1 Reserved | This field is reserved. This read-only field is reserved and always has the value 0. |
| 0 FTF | Flash Memory Clock Gate Control 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. 0 Clock disabled 1 Clock enabled |

14.5.10 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 FTFA_FOPT[LPBOOT] (See [Table 6-2](#)).

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 | 0 | 1 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | | |

* Notes:

- OUTDIV1 field: The reset value depends on the FTFA_FOPT[LPBOOT]. It is loaded with 0000 (divide by 1), 0001 (divide by 2), 0011 (divide by 4, or 0111 (divide by 8)).

SIM_CLKDIV1 field descriptions

| Field | Description |
|-------------------|--|
| 31–28 OUTDIV1 | <p>Clock 1 Output Divider value</p> <p>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 FTFA_FOPT[LPBOOT] (See Table 6-2).</p> <p>0000 Divide-by-1. 0001 Divide-by-2. 0010 Divide-by-3. 0011 Divide-by-4. 0100 Divide-by-5. 0101 Divide-by-6. 0110 Divide-by-7. 0111 Divide-by-8. 1000 Divide-by-9. 1001 Divide-by-10. 1010 Divide-by-11. 1011 Divide-by-12. 1100 Divide-by-13. 1101 Divide-by-14. 1110 Divide-by-15. 1111 Divide-by-16.</p> |
| 27–19 Reserved | <p>This field is reserved.</p> <p>This read-only field is reserved and always has the value 0.</p> |
| 18–16 OUTDIV4 | <p>Clock 4 Output Divider value</p> <p>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 2).</p> <p>000 Divide-by-1. 001 Divide-by-2. 010 Divide-by-3. 011 Divide-by-4. 100 Divide-by-5. 101 Divide-by-6. 110 Divide-by-7. 111 Divide-by-8.</p> |
| Reserved | <p>This field is reserved.</p> <p>This read-only field is reserved and always has the value 0.</p> |

14.5.11 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 | | | | | | | | | | | | | | | FLASHDOZE | FLASHDIS |
| 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 Reserved | This field is reserved. This read-only field is reserved and always has the value 0. |
| 27–24 PFSIZE | Program Flash Size Specifies the amount of program flash memory available on the device . Undefined values are reserved. 0000 8 KB of program flash memory, 1 KB protection region 0001 16 KB of program flash memory, 1 KB protection region 0011 32 KB of program flash memory, 1 KB protection region 0101 64 KB of program flash memory, 2 KB protection region 0111 128 KB of program flash memory, 4 KB protection region 1001 256 KB of program flash memory, 8 KB protection region 1111 32 KB of program flash memory, 1 KB protection region |

Table continues on the next page...

SIM_FCFG1 field descriptions (continued)

| Field | Description |
|------------------|---|
| 23–2 Reserved | This field is reserved. This read-only field is reserved and always has the value 0. |
| 1 FLASHDOZE | Flash Doze When set, flash memory is disabled for the duration of Doze mode. This field must 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 wake-up time from Doze mode is extended when this field is set. 0 Flash remains enabled during Doze mode. 1 Flash is disabled for the duration of Doze mode. |
| 0 FLASHDIS | Flash Disable Flash accesses are disabled (and generate a bus error) and the flash memory is placed in a low-power state. This field should not be changed during VLP modes. Relocate the interrupt vectors out of Flash memory before disabling the Flash. 0 Flash is enabled. 1 Flash is disabled. |

14.5.12 Flash Configuration Register 2 (SIM_FCFG2)

This is read only register, any write to this register will cause transfer error.

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 Reserved | This field is reserved. This read-only field is reserved and always has the value 0. |

Table continues on the next page...

SIM_FCFG2 field descriptions (continued)

| Field | Description |
|-------------------|--|
| 30–24 MAXADDR0 | Max Address lock This field concatenated with 13 trailing zeros indicates the first invalid address of program flash. 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 Reserved | This field is reserved. This read-only field is reserved and always has the value 1. |
| 22–16 Reserved | This field is reserved. This read-only field is reserved and always has the value 0. |
| Reserved | This field is reserved. This read-only field is reserved and always has the value 0. |

14.5.13 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 | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | UID | |
| 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 Reserved | This field is reserved. This read-only field is reserved and always has the value 0. |
| UID | Unique Identification Unique identification for the device. |

14.5.14 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 | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | UID | |
| W | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| Reset | * | * | * | * | * | * | * | * | * | * | * | * | * | * | * | * | * | * | * | * | * | * | * | * | * | * | * | * | * | * | * | |

* Notes:

- UID field: Device specific value.

SIM_UIDML field descriptions

| Field | Description |
|-------|--|
| UID | Unique Identification Unique identification for the device. |

14.5.15 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 | UID | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| W | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| Reset | * | * | * | * | * | * | * | * | * | * | * | * | * | * | * | * | * | * | * | * | * | * | * | * | * | * | * | * | * | * | * | |

* Notes:

- UID field: Device specific value.

SIM_UIDL field descriptions

| Field | Description |
|-------|--|
| UID | Unique Identification Unique identification for the device. |

14.5.16 COP Control Register (SIM_COPC)

All of the bits in this register can be written only once after a reset, writing this register will also reset the COP counter.

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 |
| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 0 | | | | | | | |
| R | 0 | | | | | | | | COPCLKSEL | COPDBGEN | COPSTPEN | COPT | COPCLKS | COPW | | |
| W | | | | | | | | | 0 | 0 | 0 | 0 | 1 | 1 | 0 | 0 |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 1 | 1 | 0 | 0 |

SIM_COPC field descriptions

| Field | Description |
|------------------|--|
| 31–8 Reserved | This field is reserved. This read-only field is reserved and always has the value 0. |
| 7–6 COPCLKSEL | COP Clock Select This write-once field selects the clock source of the COP watchdog. 00 LPO clock (1 kHz) 01 MCGIRCLK 10 OSCERCLK 11 Bus clock |
| 5 COPDBGEN | COP Debug Enable 0 COP is disabled and the counter is reset in Debug mode 1 COP is enabled in Debug mode |
| 4 COPSTPEN | COP Stop Enable 0 COP is disabled and the counter is reset in Stop modes 1 COP is enabled in Stop modes |
| 3–2 COPT | COP Watchdog Timeout This write-once field selects the timeout period of the COP. COPT along with the COPCLKS field define the COP timeout period. 00 COP disabled 01 COP timeout after 2^5 cycles for short timeout or 2^{13} cycles for long timeout 10 COP timeout after 2^8 cycles for short timeout or 2^{16} cycles for long timeout 11 COP timeout after 2^{10} cycles for short timeout or 2^{18} cycles for long timeout |
| 1 COPCLKS | COP Clock Select This write-once field selects between a short timeout or a long timeout, the COP clock source is configured by COPCLKSEL. 0 COP configured for short timeout 1 COP configured for long timeout |
| 0 COPW | COP Windowed Mode Windowed mode is supported for all COP clock sources, but only when the COP is configured for a long timeout. The COP window is opened three quarters through the timeout period and will generate a system reset if the COP is serviced outside of that time. 0 Normal mode 1 Windowed mode |

14.5.17 Service COP (SIM_SRVCOP)

This is write only register, any read to this register will cause transfer error.

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 | Reserved | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| 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 | |

SIM_SRVCOP field descriptions

| Field | Description |
|------------------|--|
| 31–8 Reserved | This field is reserved. |
| SRVCOP | Service COP Register Write 0x55 and then 0xAA (in that order) to reset the COP timeout counter, writing any other value will generate a system reset. |

14.6 Functional description

See [Introduction](#) section.

Chapter 15

System Mode Controller (SMC)

15.1 Chip-specific SMC information

This device does not support VLLS2 power mode. Ignore the VLLS2 in the following sections.

15.2 Introduction

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.

See [AN4503: Power Management for Kinetis and ColdFire+ MCUs](#) for further details on using the SMC.

15.3 Modes of operation

The ARM CPU has three primary modes of operation:

- Run
- Sleep
- Deep Sleep

Modes of operation

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 Kinetis microcontrollers.

The following table shows the translation between the ARM CPU modes and the Kinetis MCU power modes.

| ARM CPU mode | MCU mode |
|--------------|----------|
| Sleep | Wait |
| Deep Sleep | Stop |

Accordingly, the ARM CPU documentation refers to sleep and deep sleep, while the Kinetis MCU documentation normally uses wait and stop.

In addition, Kinetis 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 15-1. 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. |

Table continues on the next page...

Table 15-1. Power modes (continued)

| Mode | Description |
|-------|--|
| 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. |
| 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]. |

15.4 Memory map and register descriptions

Information about the registers related to the system mode controller can be found here.

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.

NOTE

Before executing the WFI instruction, the last register written to must be read back. This ensures that all register writes associated with setting up the low power mode being entered have completed before the MCU enters the low power mode. Failure to do this may result in the low power mode not being entered correctly.

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 | 15.4.1/202 |
| 4007_E001 | Power Mode Control register (SMC_PMCTRL) | 8 | R/W | 00h | 15.4.2/203 |
| 4007_E002 | Stop Control Register (SMC_STOPCTRL) | 8 | R/W | 03h | 15.4.3/204 |
| 4007_E003 | Power Mode Status register (SMC_PMSTAT) | 8 | R | 01h | 15.4.4/206 |

15.4.1 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).

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 PMCTRL[RUNM] remains 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 | 0 | AVLP | 0 | 0 | 0 | AVLLS | 0 |
| Write | | | | | | | | |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

SMC_PMPROT field descriptions

| Field | Description |
|---------------|---|
| 7 Reserved | This field is reserved. This read-only field is reserved and always has the value 0. |
| 6 Reserved | This field is reserved. This read-only field is reserved and always has the value 0. |
| 5 AVLP | Allow Very-Low-Power Modes Provided the appropriate control bits are set up in PMCTRL, this write-once field allows the MCU to enter any very-low-power mode (VLPR, VLPW, and VLPS). |

Table continues on the next page...

SMC_PMPROT field descriptions (continued)

| Field | Description |
|------------|---|
| | 0 VLPR, VLPW, and VLPS are not allowed. 1 VLPR, VLPW, and VLPS are allowed. |
| 4 Reserved | This field is reserved. This read-only field is reserved and always has the value 0. |
| 3 Reserved | This field is reserved. This read-only field is reserved and always has the value 0. |
| 2 Reserved | This field is reserved. This read-only field is reserved and always has the value 0. |
| 1 AVLLS | Allow Very-Low-Leakage Stop Mode 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). 0 Any VLLSx mode is not allowed 1 Any VLLSx mode is allowed |
| 0 Reserved | This field is reserved. This read-only field is reserved and always has the value 0. |

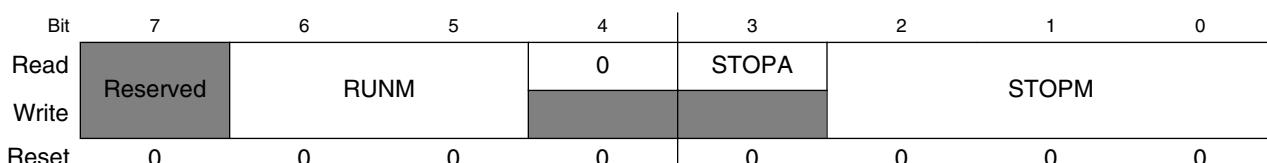
15.4.2 Power Mode Control register (SMC_PMCTRL)

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

**SMC_PMCTRL field descriptions**

| Field | Description |
|------------|---|
| 7 Reserved | This field is reserved. This bit is reserved for future expansion and should always be written zero. |
| 6–5 RUNM | Run Mode Control |

Table continues on the next page...

SMC_PMCTRL field descriptions (continued)

| Field | Description |
|------------|---|
| | <p>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.</p> <p>NOTE: RUNM may be set to VLPR only when PMSTAT=RUN. After being written to VLPR, RUNM should not be written back to RUN until PMSTAT=VLPR.</p> <p>00 Normal Run mode (RUN) 01 Reserved 10 Very-Low-Power Run mode (VLPR) 11 Reserved</p> |
| 4 Reserved | <p>This field is reserved. This read-only field is reserved and always has the value 0.</p> |
| 3 STOPA | <p>Stop Aborted</p> <p>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 field is cleared by hardware at the beginning of any stop mode entry sequence and is set if the sequence was aborted.</p> <p>0 The previous stop mode entry was successful. 1 The previous stop mode entry was aborted.</p> |
| 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>NOTE: When set to VLLSx, the VLLSM field in the STOPCTRL register is used to further select the particular VLLS submode which will be entered.</p> <p>NOTE: When set to STOP, the PSTOPO bits in the STOPCTRL register can be used to select a Partial Stop mode if desired.</p> <p>000 Normal Stop (STOP) 001 Reserved 010 Very-Low-Power Stop (VLPS) 011 Reserved 100 Very-Low-Leakage Stop (VLLSx) 101 Reserved 110 Reserved 111 Reserved</p> |

15.4.3 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 nor VLLS. See the Reset section details for more information.

Address: 4007_E000h base + 2h offset = 4007_E002h

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|-------|--------|-------|---|-------|-------|---|---|---|
| Read | PSTOPO | PORPO | 0 | LPOPO | VLLSM | | | |
| Write | 0 | 0 | 0 | 0 | 0 | 1 | 1 | 1 |
| Reset | | | | | | | | |

SMC_STOPCTRL field descriptions

| Field | Description |
|---------------|--|
| 7–6 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 clocks are gated allowing peripherals running on bus clock to remain fully functional. In PSTOP1, both system and bus clocks are gated.</p> <ul style="list-style-type: none"> 00 STOP - Normal Stop mode 01 PSTOP1 - Partial Stop with both system and bus clocks disabled 10 PSTOP2 - Partial Stop with system clock disabled and bus clock enabled 11 Reserved |
| 5 PORPO | <p>POR Power Option</p> <p>This bit controls whether the POR detect circuit is enabled in VLLS0 mode.</p> <ul style="list-style-type: none"> 0 POR detect circuit is enabled in VLLS0 1 POR detect circuit is disabled in VLLS0 |
| 4 Reserved | <p>This field is reserved.</p> <p>This read-only field is reserved and always has the value 0.</p> |
| 3 LPOPO | <p>LPO Power Option</p> <p>Controls whether the 1 kHz LPO clock is enabled in VLLSx modes.</p> <p>NOTE: During VLLS0 mode, the LPO clock is disabled by hardware and this bit has no effect.</p> <ul style="list-style-type: none"> 0 LPO clock is enabled in VLLSx 1 LPO clock is disabled in VLLSx |
| VLLSM | <p>VLLS Mode Control</p> <p>This field controls which VLLS sub-mode to enter if STOPM = VLLSx.</p> <ul style="list-style-type: none"> 000 VLLS0 001 VLLS1 010 Reserved 011 VLLS3 100 Reserved 101 Reserved 110 Reserved 111 Reserved |

15.4.4 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.

Address: 4007_E000h base + 3h offset = 4007_E003h

| Bit | 7 | 6 | 5 | 4 | | 3 | 2 | 1 | 0 |
|-------|--------|---|---|---|--|---|---|---|---|
| Read | PMSTAT | | | | | | | | |
| Write | | | | | | | | | |
| Reset | 0 | 0 | 0 | 0 | | 0 | 0 | 0 | 1 |

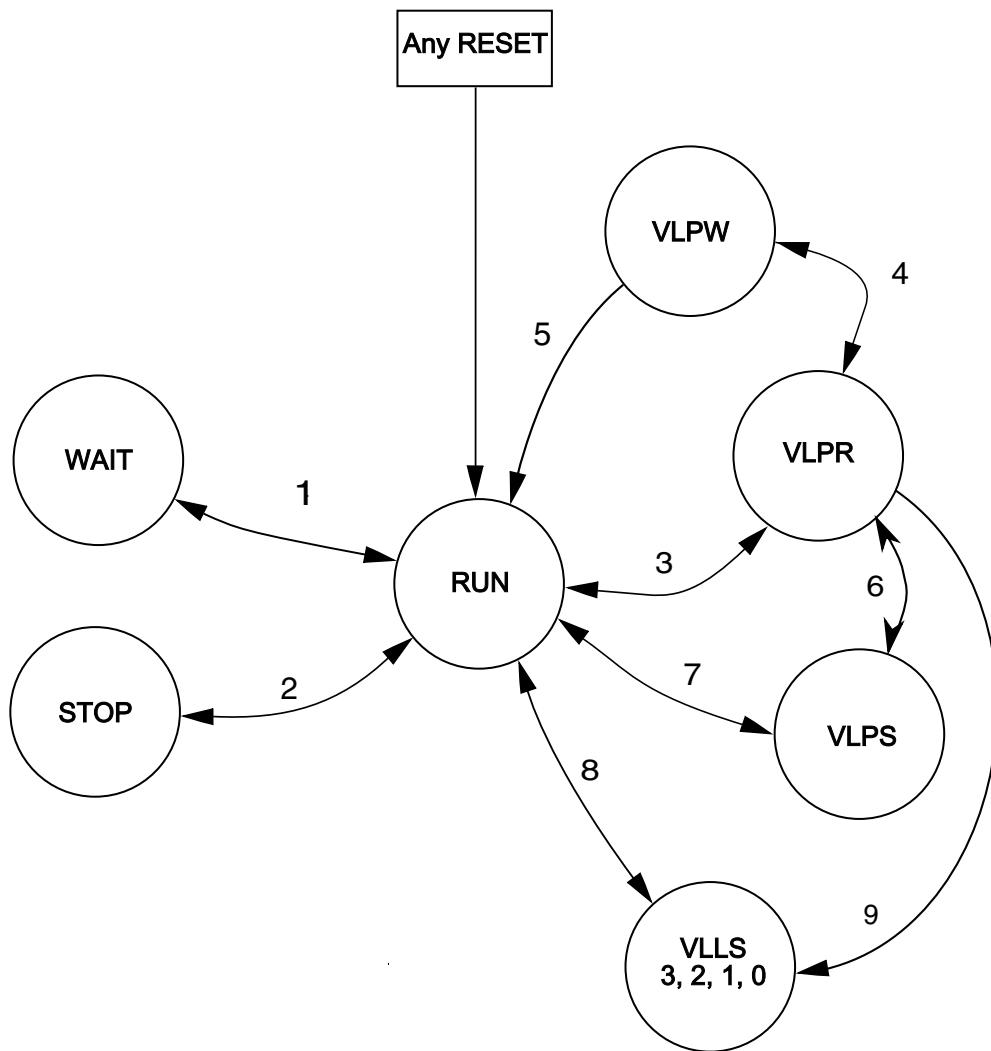
SMC_PMSTAT field descriptions

| Field | Description |
|--------|--|
| PMSTAT | <p>Power Mode Status</p> <p>NOTE: When debug is enabled, the PMSTAT will not update to STOP or VLPS</p> <p>NOTE: When a PSTOP mode is enabled, the PMSTAT will not update to STOP or VLPS</p> <p>0000_0001 Current power mode is RUN. 0000_0010 Current power mode is STOP. 0000_0100 Current power mode is VLPR. 0000_1000 Current power mode is VLPW. 0001_0000 Current power mode is VLPS. 0010_0000 Reserved 0100_0000 Current power mode is VLLS. 1000_0000 Reserved</p> |

15.5 Functional description

15.5.1 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 15-1. Power mode state diagram**

The following table defines triggers for the various state transitions shown in the previous figure.

Table 15-2. 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. See note. |
| | WAIT | RUN | Interrupt or Reset |
| 2 | RUN | STOP | PMCTRL[RUNM]=00, PMCTRL[STOPM]=000 ² |

Table continues on the next page...

Table 15-2. Power mode transition triggers (continued)

| Transition # | From | To | Trigger conditions |
|--------------|-------|-------|--|
| | | | Sleep-now or sleep-on-exit modes entered with SLEEPDEEP set, which is controlled in System Control Register in ARM core. See note. ¹ |
| | STOP | RUN | Interrupt or Reset |
| 3 | RUN | VLPR | The core, system, bus and flash clock frequencies and MCG clocking mode are restricted in this mode. See the Power Management chapter for the maximum allowable frequencies and MCG modes supported. Set PMPROT[AVLP]=1, PMCTRL[RUNM]=10. |
| | VLPR | RUN | Set PMCTRL[RUNM]=00 or Reset. |
| | VLPR | VLPW | Sleep-now or sleep-on-exit modes entered with SLEEPDEEP clear, which is controlled in System Control Register in ARM core. See note. ¹ |
| 4 | VLPW | VLPR | Interrupt |
| | VLPW | RUN | Reset |
| 5 | VLPS | VLPR | PMCTRL[STOPM]=000 ³ or 010, Sleep-now or sleep-on-exit modes entered with SLEEPDEEP set, which is controlled in System Control Register in ARM core. See note. ¹ |
| 6 | VLPS | VLPR | Interrupt NOTE: If VLPS was entered directly from RUN (transition #7), hardware forces exit back to RUN and does not allow a transition to VLPR. |
| | RUN | VLPS | PMPROT[AVLP]=1, PMCTRL[STOPM]=010, Sleep-now or sleep-on-exit modes entered with SLEEPDEEP set, which is controlled in System Control Register in ARM core. See note. ¹ |
| 7 | VLPS | RUN | Interrupt and VLPS mode was entered directly from RUN or Reset |
| | 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. |
| 8 | VLLSx | RUN | Wakeup from enabled LLWU input source or RESET pin |
| | 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. |

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

15.5.2 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. This 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.

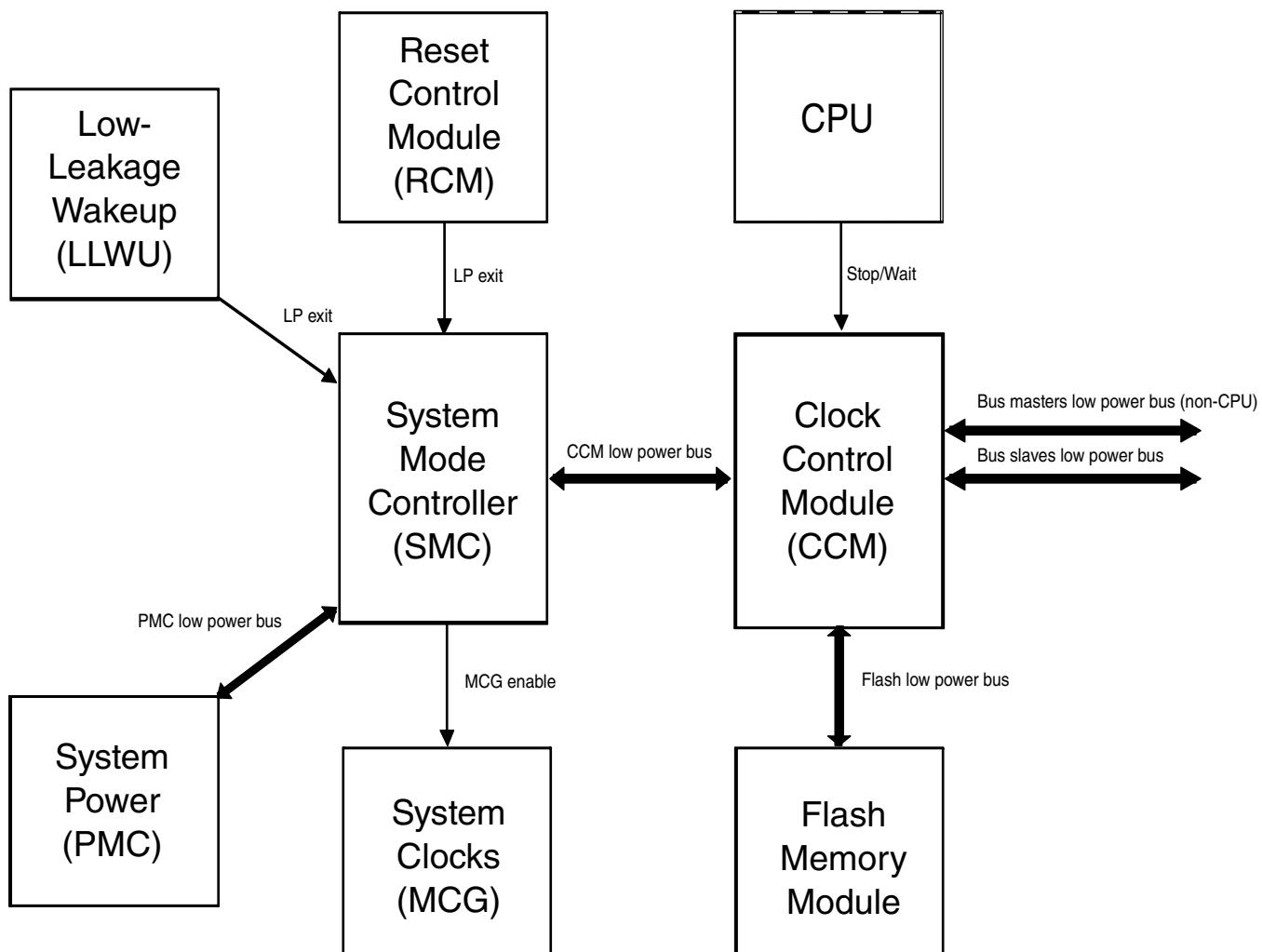


Figure 15-2. Low-power system components and connections

15.5.2.1 Stop mode entry sequence

Entry into a low-power stop mode (Stop, VLPS, 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.

15.5.2.2 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.

15.5.2.3 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, SMC_PMCTRL[STOPA] is set to 1.

15.5.2.4 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.

15.5.2.5 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. As part of this transition, system clocking is re-established and is equivalent to the normal RUN and VLPR mode clocking configuration.

15.5.3 Run modes

The run modes supported by this device can be found here.

- Run (RUN)
- Very Low-Power Run (VLPR)

15.5.3.1 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.

15.5.3.2 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 increase the clock frequency while in VLPR mode, because the regulator is slow in responding and cannot manage fast load transitions. In addition, do not modify the clock source in the MCG module or any clock divider registers. Module clock enables in the SIM can be set, but not cleared.

To reenter Normal Run mode, clear PMCTRL[RUNM]. PMSTAT 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 PMSTAT 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.

15.5.4 Wait modes

This device contains two different wait modes which are listed here.

- Wait
- Very-Low Power Wait (VLPW)

15.5.4.1 WAIT mode

WAIT mode is entered when the ARM core enters the Sleep-Now or Sleep-On-Exit modes while SLEEDEEP 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 module.

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.

15.5.4.2 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.

15.5.5 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 and recovery time may be traded off.

NOTE

All clock monitors must be disabled before entering these low-power modes: Stop, VLPS, VLPR, VLPW, and VLLSx.

The various stop modes are selected by setting the appropriate fields in PMPROT and PMCTRL. The selected stop mode is entered during the sleep-now or sleep-on-exit entry with the SLEEPDEEP bit set in the System Control Register in the ARM core.

The available stop modes are:

- Normal Stop (STOP)
- Very-Low Power Stop (VLPS)
- Very-Low-Leakage Stop (VLLSx)

15.5.5.1 STOP mode

STOP mode is entered via the sleep-now or sleep-on-exit with the SLEEPDEEP bit set in the System Control Register in the ARM core.

The MCG module can be configured to leave the reference clocks running.

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.

15.5.5.2 Very-Low-Power Stop (VLPS) mode

The two ways in which VLPS mode can be entered are listed here.

- 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 PMCTRL[STOPM] = 010 or 000.
- 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 PMCTRL[STOPM] = 010. 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.

15.5.5.3 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 15-2](#).

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 Wake-up (LLWU) module to enable the desired wakeup sources. The available wake-up 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 wake-up flags to determine the source of the wake-up.

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 PMC_REGSC[ACKISO] is set.

An asserted RESET pin will cause an exit from any VLLS mode, returning the device to normal RUN mode. When exiting VLLS via the RESET pin, RCM_SRS0[PIN] and RCM_SRS0[WAKEUP] are set.

15.5.6 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 CDBGWRUPREQ 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 CDBGWRUPACK 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:

Functional description

- 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 VLLS modes.

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.

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 16

Power Management Controller (PMC)

16.1 Introduction

The power management controller (PMC) contains the internal voltage regulator, power on reset (POR), and low voltage detect system.

See [AN4503: Power Management for Kinetis and ColdFire+ MCUs](#) for further details on using the PMC.

16.2 Features

A list of included PMC features can be found here.

- 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

16.3 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 LVDSC1[LVDV]. The LVD is disabled upon entering VLPx and VLLSx modes.

Two flags are available to indicate the status of the low-voltage detect system:

- The Low Voltage Detect Flag in the Low Voltage Status and Control 1 Register (LVDSC1[LVDF]) operates in a level sensitive manner. LVDSC1[LVDF] is set

when the supply voltage falls below the selected trip point (VLVD).

LVDSC1[LVDF] is cleared by writing 1 to LVDSC1[LVDACK], but only if the internal supply has returned above the trip point; otherwise, LVDSC1[LVDF] remains set.

- The Low Voltage Warning Flag (LVWF) in the Low Voltage Status and Control 2 Register (LVDSC2[LVWF]) operates in a level sensitive manner. LVDSC2[LVWF] is set when the supply voltage falls below the selected monitor trip point (VLVW). LVDSC2[LVWF] is cleared by writing one to LVDSC2[LVWACK], but only if the internal supply has returned above the trip point; otherwise, LVDSC2[LVWF] remains set.

16.3.1 LVD reset operation

By setting LVDSC1[LVDRE], the LVD generates a reset upon detection of a low-voltage condition. The low-voltage detection threshold is determined by LVDSC1[LVDV]. After an LVD reset occurs, the LVD system holds the MCU in reset until the supply voltage rises above this threshold. The LVD field in the SRS register of the RCM module (RCM_SRS0[LVD]) is set following an LVD or power-on reset.

16.3.2 LVD interrupt operation

By configuring the LVD circuit for interrupt operation (LVDSC1[LVDIE] set and LVDSC1[LVDRE] clear), LVDSC1[LVDF] is set and an LVD interrupt request occurs upon detection of a low voltage condition. LVDSC1[LVDF] is cleared by writing 1 to LVDSC1[LVDACK].

16.3.3 Low-voltage warning (LVW) interrupt operation

The LVD system contains a Low-Voltage Warning Flag (LVWF) in the Low Voltage Detect Status and Control 2 Register 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 LVDSC2[LVWIE]. If enabled, an LVW interrupt request occurs when LVDSC2[LVWF] is set. LVDSC2[LVWF] is cleared by writing 1 to LVDSC2[LVWACK].

LVDSC2[LVWV] selects one of the four trip voltages:

- Highest: V_{LVW4}

- Two mid-levels: V_{LVW3} and V_{LVW2}
- Lowest: V_{LVW1}

16.4 I/O retention

When in VLLS modes, the I/O states are held on a wake-up event (with the exception of wake-up by reset event) until the wake-up has been acknowledged via a write to REGSC[ACKISO]. 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 REGSC[ACKISO] is needed.

16.5 Memory map and register descriptions

Details about the PMC registers can be found here.

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 | 16.5.1/220 |
| 4007_D001 | Low Voltage Detect Status And Control 2 register (PMC_LVDSC2) | 8 | R/W | 00h | 16.5.2/221 |
| 4007_D002 | Regulator Status And Control register (PMC_REGSC) | 8 | R/W | 04h | 16.5.3/222 |

16.5.1 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 Power Mode Protection (PMPROT) register of the SMC module (SMC_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 LVDF | Low-Voltage Detect Flag This read-only status field indicates a low-voltage detect event. 0 Low-voltage event not detected 1 Low-voltage event detected |
| 6 LVDACK | Low-Voltage Detect Acknowledge This write-only field is used to acknowledge low voltage detection errors. Write 1 to clear LVDF. Reads always return 0. |
| 5 LVDIE | Low-Voltage Detect Interrupt Enable Enables hardware interrupt requests for LVDF. 0 Hardware interrupt disabled (use polling) 1 Request a hardware interrupt when LVDF = 1 |

Table continues on the next page...

PMC_LVDSC1 field descriptions (continued)

| Field | Description |
|-----------------|--|
| 4 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 1 Force an MCU reset when LVDF = 1</p> |
| 3–2 Reserved | <p>This field is reserved.</p> <p>This read-only field is reserved and always has the value 0.</p> |
| LVDV | <p>Low-Voltage Detect Voltage Select</p> <p>Selects the LVD trip point voltage (V_{LVD}).</p> <p>00 Low trip point selected ($V_{LVD} = V_{LVDL}$) 01 High trip point selected ($V_{LVD} = V_{LVDH}$) 10 Reserved 11 Reserved</p> |

16.5.2 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.

NOTE

LVWV is reset solely on a POR Only event. The other fields of the register 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 LVWF | <p>Low-Voltage Warning Flag</p> <p>This read-only status field indicates a low-voltage warning event. LVWF is set when V_{Supply} transitions below the trip point, or after reset and V_{Supply} is already below V_{LVW}. LVWF 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"> 0 Low-voltage warning event not detected 1 Low-voltage warning event detected |
| 6 LVWACK | <p>Low-Voltage Warning Acknowledge</p> <p>This write-only field is used to acknowledge low voltage warning errors. Write 1 to clear LVWF. Reads always return 0.</p> |
| 5 LVWIE | <p>Low-Voltage Warning Interrupt Enable</p> <p>Enables hardware interrupt requests for LVWF.</p> <ul style="list-style-type: none"> 0 Hardware interrupt disabled (use polling) 1 Request a hardware interrupt when LVWF = 1 |
| 4–2 Reserved | <p>This field is reserved.</p> <p>This read-only field is reserved and always has the value 0.</p> |
| LVWV | <p>Low-Voltage Warning Voltage Select</p> <p>Selects the LVW trip point voltage (V_{LVW}). The actual voltage for the warning depends on LVDSC1[LVDV].</p> <ul style="list-style-type: none"> 00 Low trip point selected ($V_{LVW} = V_{LVW1}$) 01 Mid 1 trip point selected ($V_{LVW} = V_{LVW2}$) 10 Mid 2 trip point selected ($V_{LVW} = V_{LVW3}$) 11 High trip point selected ($V_{LVW} = V_{LVW4}$) |

16.5.3 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 details for more information.

Address: 4007_D000h base + 2h offset = 4007_D002h

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|-------|---|---|----------|------|--------|--------|----------|------|
| Read | 0 | 0 | Reserved | BGEN | ACKISO | REGONS | Reserved | BGBE |
| Write | | | | | w1c | | | |
| Reset | 0 | 0 | 0 | 0 | 0 | 1 | 0 | 0 |

PMC_REGSC field descriptions

| Field | Description |
|---------------|---|
| 7 Reserved | This field is reserved. This read-only field is reserved and always has the value 0. |
| 6 Reserved | This field is reserved. This read-only field is reserved and always has the value 0. |
| 5 Reserved | This field is reserved. |
| 4 BGEN | Bandgap Enable In VLPx Operation BGEN controls whether the bandgap is enabled in lower power modes of operation (VLPx, 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. NOTE: When the bandgap voltage reference is not needed in low power modes, clear BGEN to avoid excess power consumption. 0 Bandgap voltage reference is disabled in VLPx , and VLLSx modes. 1 Bandgap voltage reference is enabled in VLPx , and VLLSx modes. |
| 3 ACKISO | Acknowledge Isolation Reading this field 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 1 to this field when it is set releases the I/O pads and certain peripherals to their normal run mode state. NOTE: 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. 0 Peripherals and I/O pads are in normal run state. 1 Certain peripherals and I/O pads are in an isolated and latched state. |
| 2 REGONS | Regulator In Run Regulation Status This read-only field provides the current status of the internal voltage regulator. 0 Regulator is in stop regulation or in transition to/from it 1 Regulator is in run regulation |
| 1 Reserved | This field is reserved. NOTE: This reserved bit must remain cleared (set to 0). |
| 0 BGBE | Bandgap Buffer Enable Enables the bandgap buffer. 0 Bandgap buffer not enabled 1 Bandgap buffer enabled |

Chapter 17

Miscellaneous Control Module (MCM)

17.1 Introduction

The Miscellaneous Control Module (MCM) provides a myriad of miscellaneous control functions.

17.1.1 Features

The MCM includes the following features:

- Program-visible information on the platform configuration
- Flash controller speculation buffer configurations

17.2 Memory map/register descriptions

The memory map and register descriptions found here describe the registers using byte addresses. The registers can be written only when in supervisor mode.

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 | 17.2.1/226 |
| F000_300A | Crossbar Switch (AXBS) Master Configuration (MCM_PLAMC) | 16 | R | 0001h | 17.2.2/226 |
| F000_300C | Platform Control Register (MCM_PLACR) | 32 | R/W | 0000_0000h | 17.2.3/227 |
| F000_3040 | Compute Operation Control Register (MCM_CPO) | 32 | R/W | 0000_0000h | 17.2.4/229 |

17.2.1 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 Reserved | This field is reserved. This read-only field is reserved and always has the value 0. |
| ASC | Each bit in the ASC field indicates whether there is a corresponding connection to the crossbar switch's slave input port. 0 A bus slave connection to AXBS input port <i>n</i> is absent. 1 A bus slave connection to AXBS input port <i>n</i> is present. |

17.2.2 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 | 0 | 0 | 0 | 1 |

MCM_PLAMC field descriptions

| Field | Description |
|------------------|--|
| 15–8 Reserved | This field is reserved. This read-only field is reserved and always has the value 0. |
| AMC | Each bit in the AMC field indicates whether there is a corresponding connection to the AXBS master input port. |

Table continues on the next page...

MCM_PLAMC field descriptions (continued)

| Field | Description |
|-------|---|
| 0 | A bus master connection to AXBS input port n is absent |
| 1 | A bus master connection to AXBS input port n is present |

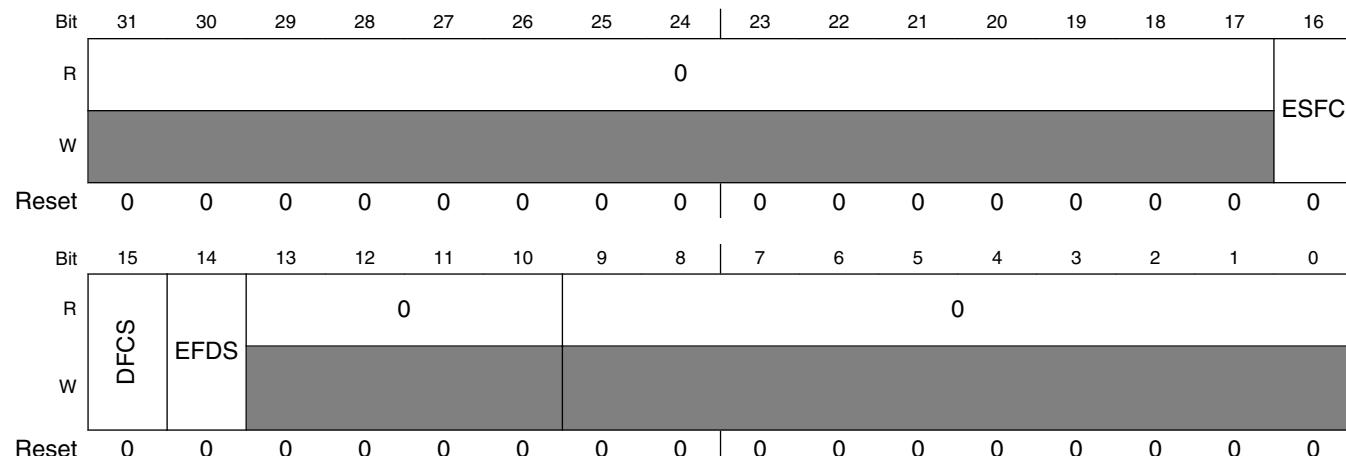
17.2.3 Platform Control Register (MCM_PLACR)

The speculation buffer in the flash memory controller is configurable via PLACR[15:14].

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. |

Address: F000_3000h base + Ch offset = F000_300Ch

**MCM_PLACR field descriptions**

| Field | Description |
|-------------------|---|
| 31–17 Reserved | This field is reserved. This read-only field is reserved and always has the value 0. |
| 16 ESFC | Enable Stalling Flash Controller Enables stalling flash controller when flash is busy. |

Table continues on the next page...

MCM_PLACR field descriptions (continued)

| Field | Description |
|-------------------|---|
| | <p>When software needs to access the flash memory while a flash memory resource is being manipulated by a flash command, software can enable a stall mechanism to avoid a read collision. The stall mechanism allows software to execute code from the same block on which flash operations are being performed. However, software must ensure the sector the flash operations are being performed on is not the same sector from which the code is executing.</p> <p>ESFC enables the stall mechanism. This bit must be set only just before the flash operation is executed and must be cleared when the operation completes.</p> <ul style="list-style-type: none"> 0 Disable stalling flash controller when flash is busy. 1 Enable stalling flash controller when flash is busy. |
| 15 DFCS | <p>Disable Flash Controller Speculation</p> <p>Disables flash controller speculation.</p> <ul style="list-style-type: none"> 0 Enable flash controller speculation. 1 Disable flash controller speculation. |
| 14 EFDS | <p>Enable Flash Data Speculation</p> <p>Enables flash data speculation.</p> <ul style="list-style-type: none"> 0 Disable flash data speculation. 1 Enable flash data speculation. |
| 13–10 Reserved | <p>This field is reserved.</p> <p>This read-only field is reserved and always has the value 0.</p> |
| Reserved | <p>This field is reserved.</p> <p>This read-only field is reserved and always has the value 0.</p> |

17.2.4 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. This read-only field is reserved and always has the value 0. |
| 2 CPOWOI | Compute Operation Wake-up on Interrupt 0 No effect. 1 When set, the CPOREQ is cleared on any interrupt or exception vector fetch. |
| 1 CPOACK | Compute Operation Acknowledge 0 Compute operation entry has not completed or compute operation exit has completed. 1 Compute operation entry has completed or compute operation exit has not completed. |
| 0 CPOREQ | Compute Operation Request 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 18

Crossbar Switch Lite (AXBS-Lite)

18.1 Introduction

The information found here 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.

NOTE

This device has only one master which has full control over the slaves.

18.1.1 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

18.2 Memory Map / Register Definition

This crossbar switch is designed for minimal gate count. It, therefore, has no memory-mapped configuration registers.

18.3 Functional Description

18.3.1 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.

Chapter 19

Low-Leakage Wakeup Unit (LLWU)

19.1 Chip-specific LLWU information

This device has no LLWU module wakeup source and uses the external pin inputs only of LLWU_P4 and LLWU_P7 as wakeup source to the LLWU module. The internal peripheral interrupt flags are connected to the platform directly.

Hence, only the following registers and fields are functional for this device:

- LLWU_PE2: [WUPE7] and [WUPE4]
- LLWU_F1: [WUF7] and [WUF4]
- LLWU_FILTER1

19.2 Introduction

The LLWU module allows the user to select up to 8 external pins and up to 8 internal modules as interrupt wake-up sources from low-leakage power modes. It also supports up to 8 internal modules as temporary DMA wake-up sources.

The input sources are described in the device's chip configuration details. Each of the available wake-up sources can be individually enabled.

The RESET pin is an additional source for triggering an exit from low-leakage power modes, and causes the MCU to exit VLLS through a reset flow.

The LLWU module also includes two optional digital pin filters for the external wakeup pins.

See [AN4503: Power Management for Kinetis and ColdFire+ MCUs](#) for further details on using the LLWU.

19.2.1 Features

The LLWU module features include:

- Support for up to 8 external input pins and up to 8 internal modules with individual enable bits for MCU interrupt from low leakage modes
- Support for up to 8 internal modules with individual enable bits for DMA wakeup from low leakage modes
- Input sources may be external pins or from internal peripherals capable of running in VLLS. See the chip configuration information for wakeup input sources for this device.
- External pin wake-up inputs, each of which is programmable as falling-edge, rising-edge, or any change
- Wake-up inputs that are activated after MCU enters a low-leakage power mode
- Optional digital filters provided to qualify an external pin detect. Note that when the LPO clock is disabled, the filters are disabled and bypassed.

19.2.2 Modes of operation

The LLWU module becomes functional on entry into a low-leakage power mode. After recovery from VLLS, the LLWU continues to detect wake-up events until the user has acknowledged the wake-up via a write to PMC_REGSC[ACKISO].

19.2.2.1 VLLS modes

All wakeup and reset events result in VLLS exit via a reset flow.

19.2.2.2 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 wake-up pin filters are enabled, filter operation begins immediately. If a low leakage mode is entered within five LPO clock cycles of an active edge, the edge event will be detected by the LLWU.

19.2.2.3 Debug mode

When the chip is in Debug mode and then enters a VLLSx mode, no debug logic works in the fully-functional low-leakage mode. Upon an exit from the VLLSx mode, the LLWU becomes inactive.

19.2.3 Block diagram

The following figure is the block diagram for the LLWU module.

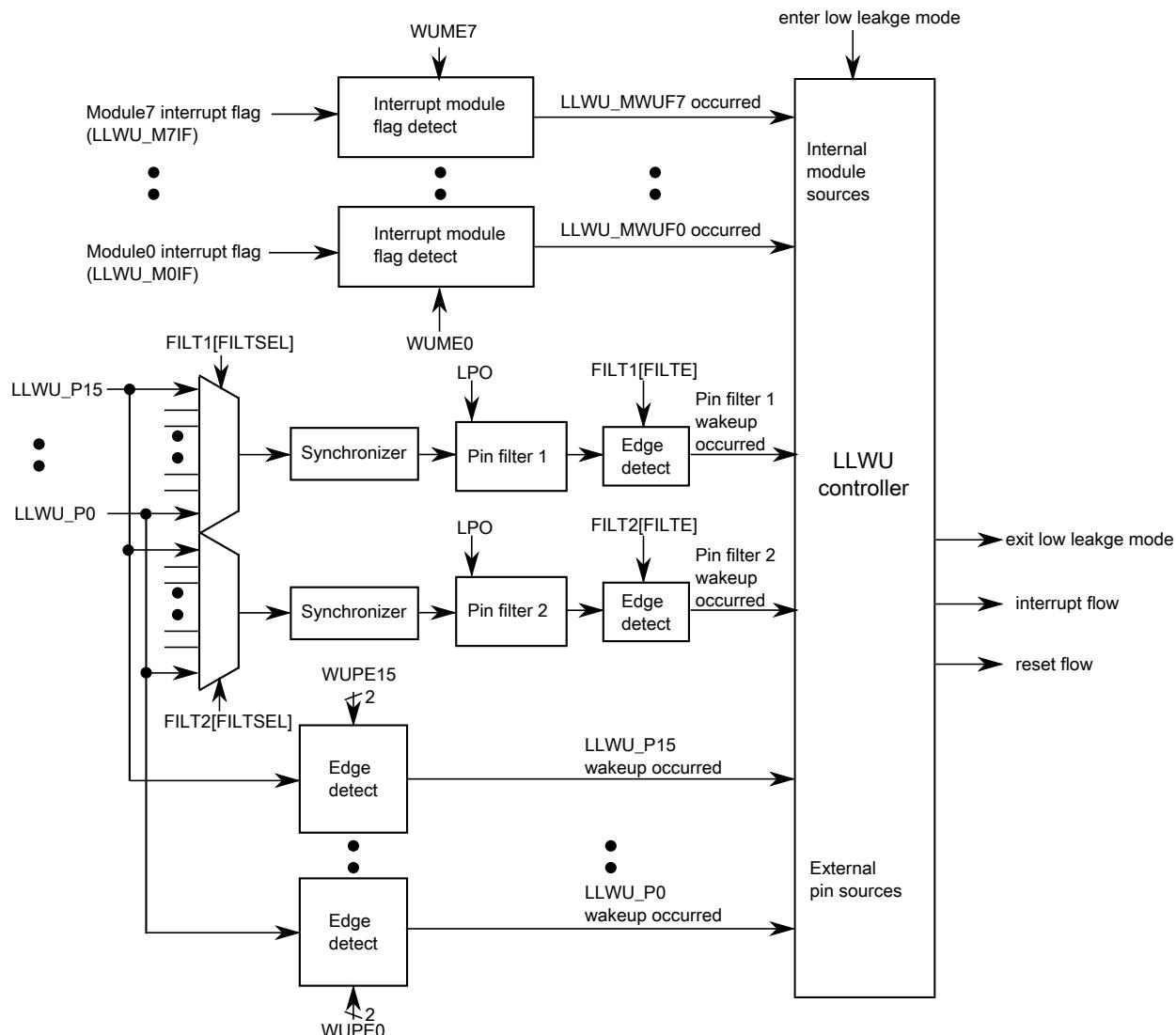


Figure 19-1. LLWU block diagram

19.3 LLWU signal descriptions

The signal properties of LLWU are shown in the table found here.

The external wakeup input pins can be enabled to detect either rising-edge, falling-edge, or on any change.

Table 19-1. LLWU signal descriptions

| Signal | Description | I/O |
|---------|--------------------------|-----|
| LLWU_Pn | Wakeup inputs (n = 0-7) | I |

19.4 Memory map/register definition

The LLWU includes the following registers:

- Wake-up source enable registers
 - Enable external pin input sources
 - Enable internal peripheral interrupt sources
 - Enable internal peripheral DMA sources
- Wake-up flag registers
 - Indication of wakeup source that caused exit from a low-leakage power mode
 - includes external pin or internal module interrupt
- Wake-up 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 | 19.4.1/239 |
| 4007_C001 | LLWU Pin Enable 2 register (LLWU_PE2) | 8 | R/W | 00h | 19.4.2/240 |
| 4007_C002 | LLWU Module Enable register (LLWU_ME) | 8 | R/W | 00h | 19.4.3/241 |
| 4007_C003 | LLWU Flag 1 register (LLWU_F1) | 8 | R/W | 00h | 19.4.4/243 |
| 4007_C004 | LLWU Flag 3 register (LLWU_F3) | 8 | R | 00h | 19.4.5/244 |
| 4007_C005 | LLWU Pin Filter 1 register (LLWU_FILTER1) | 8 | R/W | 00h | 19.4.6/246 |
| 4007_C006 | LLWU Pin Filter 2 register (LLWU_FILTER2) | 8 | R/W | 00h | 19.4.7/247 |

19.4.1 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 |
|--------------|--|
| 7–6 WUPE3 | <p>Wakeup Pin Enable For LLWU_P3</p> <p>Enables and configures the edge detection for the wakeup pin.</p> <p>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</p> |
| 5–4 WUPE2 | <p>Wakeup Pin Enable For LLWU_P2</p> <p>Enables and configures the edge detection for the wakeup pin.</p> <p>00 External input pin disabled as wakeup input 01 External input pin enabled with rising edge detection</p> |

Table continues on the next page...

LLWU_PE1 field descriptions (continued)

| Field | Description |
|--------------|--|
| | 10 External input pin enabled with falling edge detection 11 External input pin enabled with any change detection |
| 3–2 WUPE1 | Wakeup Pin Enable For LLWU_P1 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 |
| WUPE0 | Wakeup Pin Enable For LLWU_P0 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 |

19.4.2 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

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|------------|-------|---|-------|---|-------|---|-------|---|
| Read Write | WUPE7 | | WUPE6 | | WUPE5 | | WUPE4 | |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

LLWU_PE2 field descriptions

| Field | Description |
|--------------|--|
| 7–6 WUPE7 | Wakeup Pin Enable For LLWU_P7 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 |

Table continues on the next page...

LLWU_PE2 field descriptions (continued)

| Field | Description |
|--------------|---|
| | 10 External input pin enabled with falling edge detection 11 External input pin enabled with any change detection |
| 5–4 WUPE6 | Wakeup Pin Enable For LLWU_P6 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 |
| 3–2 WUPE5 | Wakeup Pin Enable For LLWU_P5 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 |
| WUPE4 | Wakeup Pin Enable For LLWU_P4 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 |

19.4.3 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 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 | WUME7 | WUME6 | WUME5 | WUME4 | WUME3 | WUME2 | WUME1 | WUME0 |
| Write | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

LLWU_ME field descriptions

| Field | Description |
|------------|---|
| 7 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 1 Internal module flag used as wakeup source</p> |
| 6 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 1 Internal module flag used as wakeup source</p> |
| 5 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 1 Internal module flag used as wakeup source</p> |
| 4 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 1 Internal module flag used as wakeup source</p> |
| 3 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 1 Internal module flag used as wakeup source</p> |
| 2 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 1 Internal module flag used as wakeup source</p> |
| 1 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 1 Internal module flag used as wakeup source</p> |
| 0 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 1 Internal module flag used as wakeup source</p> |

19.4.4 LLWU Flag 1 register (LLWU_F1)

LLWU_F1 contains the wakeup flags indicating which wakeup source caused the MCU to exit VLLS mode. 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 + 3h offset = 4007_C003h

| 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 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 1 to WUF7.</p> <p>0 LLWU_P7 input was not a wakeup source 1 LLWU_P7 input was a wakeup source</p> |
| 6 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 1 to WUF6.</p> <p>0 LLWU_P6 input was not a wakeup source 1 LLWU_P6 input was a wakeup source</p> |
| 5 WUF5 | <p>Wakeup Flag For LLWU_P5</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 1 to WUF5.</p> <p>0 LLWU_P5 input was not a wakeup source 1 LLWU_P5 input was a wakeup source</p> |
| 4 WUF4 | <p>Wakeup Flag For LLWU_P4</p> |

Table continues on the next page...

LLWU_F1 field descriptions (continued)

| Field | Description |
|-----------|--|
| | Indicates that an enabled external wake-up pin was a source of exiting a low-leakage power mode. To clear the flag, write a 1 to WUF4. 0 LLWU_P4 input was not a wakeup source 1 LLWU_P4 input was a wakeup source |
| 3 WUF3 | Wakeup Flag For LLWU_P3 Indicates that an enabled external wakeup pin was a source of exiting a low-leakage power mode. To clear the flag, write a 1 to WUF3. 0 LLWU_P3 input was not a wake-up source 1 LLWU_P3 input was a wake-up source |
| 2 WUF2 | Wakeup Flag For LLWU_P2 Indicates that an enabled external wakeup pin was a source of exiting a low-leakage power mode. To clear the flag, write a 1 to WUF2. 0 LLWU_P2 input was not a wakeup source 1 LLWU_P2 input was a wakeup source |
| 1 WUF1 | Wakeup Flag For LLWU_P1 Indicates that an enabled external wakeup pin was a source of exiting a low-leakage power mode. To clear the flag, write a 1 to WUF1. 0 LLWU_P1 input was not a wakeup source 1 LLWU_P1 input was a wakeup source |
| 0 WUF0 | Wakeup Flag For LLWU_P0 Indicates that an enabled external wake-up pin was a source of exiting a low-leakage power mode. To clear the flag, write a 1 to WUF0. 0 LLWU_P0 input was not a wakeup source 1 LLWU_P0 input was a wakeup source |

19.4.5 LLWU Flag 3 register (LLWU_F3)

LLWU_F3 contains the wakeup flags indicating which internal wakeup source caused the MCU to exit VLLS mode. 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 a real time clock module or CMP module, 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 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 | 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 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 1 Module 7 input was a wakeup source</p> |
| 6 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 1 Module 6 input was a wakeup source</p> |
| 5 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 1 Module 5 input was a wakeup source</p> |
| 4 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 1 Module 4 input was a wakeup source</p> |
| 3 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 1 Module 3 input was a wakeup source</p> |
| 2 MWUF2 | <p>Wakeup flag For module 2</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> |

Table continues on the next page...

LLWU_F3 field descriptions (continued)

| Field | Description |
|---------|--|
| | 0 Module 2 input was not a wakeup source 1 Module 2 input was a wakeup source |
| 1 MWUF1 | Wakeup flag For module 1 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. 0 Module 1 input was not a wakeup source 1 Module 1 input was a wakeup source |
| 0 MWUF0 | Wakeup flag For module 0 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. 0 Module 0 input was not a wakeup source 1 Module 0 input was a wakeup source |

19.4.6 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 + 5h offset = 4007_C005h

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|-------|-------|---|-------|---|---|---|---------|---|
| Read | FILTF | | | 0 | | | | |
| Write | w1c | | FILTE | | | | FILTSEL | |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

LLWU_FILT1 field descriptions

| Field | Description |
|---------|---|
| 7 FILTF | Filter Detect Flag 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. 0 Pin Filter 1 was not a wakeup source 1 Pin Filter 1 was a wakeup source |

Table continues on the next page...

LLWU_FILT1 field descriptions (continued)

| Field | Description |
|---------------|---|
| 6–5 FILTE | Digital Filter On External Pin Controls the digital filter options for the external pin detect. 00 Filter disabled 01 Filter posedge detect enabled 10 Filter negedge detect enabled 11 Filter any edge detect enabled |
| 4 Reserved | This field is reserved. This read-only field is reserved and always has the value 0. |
| FILTSEL | Filter Pin Select Selects 1 out of the 16 wakeup pins to be muxed into the filter. 0000 Select LLWU_P0 for filter 1111 Select LLWU_P15 for filter |

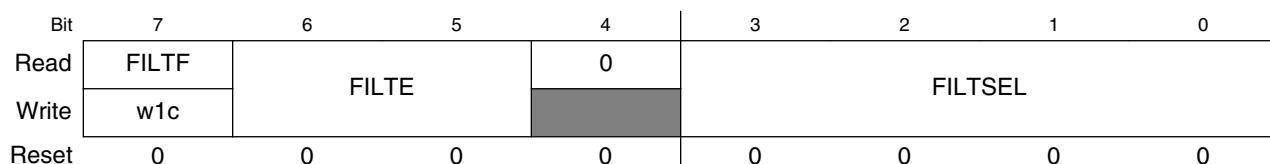
19.4.7 LLWU Pin Filter 2 register (LLWU_FILT2)

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 + 6h offset = 4007_C006h

**LLWU_FILT2 field descriptions**

| Field | Description |
|------------|---|
| 7 FILTF | Filter Detect Flag 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. 0 Pin Filter 2 was not a wakeup source 1 Pin Filter 2 was a wakeup source |

Table continues on the next page...

LLWU_FILT2 field descriptions (continued)

| Field | Description |
|---------------|---|
| 6–5 FILTE | Digital Filter On External Pin Controls the digital filter options for the external pin detect. 00 Filter disabled 01 Filter posedge detect enabled 10 Filter negedge detect enabled 11 Filter any edge detect enabled |
| 4 Reserved | This field is reserved. This read-only field is reserved and always has the value 0. |
| FILTSEL | Filter Pin Select Selects 1 out of the 16 wakeup pins to be muxed into the filter. 0000 Select LLWU_P0 for filter 1111 Select LLWU_P15 for filter |

19.5 Functional description

The low-leakage wakeup unit (LLWU) module allows internal peripherals and external input pins as a source of wakeup from low-leakage modes.

It is operational only in 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 with the following options:

- 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 for selected external pins. Glitch filtering is not provided on the internal modules.

For internal module interrupts, the WUMEx bit enables the associated module interrupt as a wakeup source.

For internal module DMA requests, the WUDEx bit enables the associated module DMA request as a temporary wakeup source.

19.5.1 VLLS modes

For any wakeup from VLLS, recovery is always via a reset flow and 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.

A VLLS exit event due to **RESET** pin assertion causes an exit via a system reset. State retention data is lost and the I/O states immediately return to their reset state. The RCM_SRS[WAKEUP] and RCM_SRS[PIN] bits are set and the system executes a reset flow before CPU operation begins with a reset vector fetch.

19.5.2 Initialization

For an enabled peripheral wakeup input, the peripheral flag must be cleared by software before entering 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 VLLSx mode.

After enabling an external pin filter or changing the source pin, wait at least five LPO clock cycles before entering VLLSx mode to allow the filter to initialize.

NOTE

After recovering from a VLLS mode, user must restore chip configuration before clearing PMC_REGSC[ACKISO]. In particular, pin configuration for enabled LLWU wake-up pins must be restored to avoid any LLWU flag from being falsely set when PMC_REGSC[ACKISO] is cleared.

The signal selected as a wake-up source pin must be a digital pin, as selected in the pin mux control.

Chapter 20

Peripheral Bridge (AIPS-Lite)

20.1 Introduction

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 occupies 64 MB of the address space, which is divided into peripheral slots of 4 KB. (It might be possible that all the peripheral slots are not used. See the memory map chapter for details on slot assignments.) The bridge includes separate clock enable inputs for each of the slots to accommodate slower peripherals.

20.1.1 Features

Key features of the peripheral bridge are:

- Supports peripheral slots with 8-, 16-, and 32-bit datapath width

20.1.2 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

Functional description

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 data width of accessed peripheral 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.

20.2 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.

20.2.1 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 21

Reset Control Module (RCM)

21.1 Introduction

Information found here 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.

See [AN4503: Power Management for Kinetis and ColdFire+ MCUs](#) for further details on using the RCM.

21.2 Reset memory map and register descriptions

The RCM Memory Map/Register Definition can be found [here](#).

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.

The addresses at 4007_F002, 4007_F003, 4007_F00A and 4007_F00B are reserved, any read and write to these addresses will not cause system 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 | 21.2.1/254 |

Table continues on the next page...

RCM memory map (continued)

| Absolute address (hex) | Register name | Width (in bits) | Access | Reset value | Section/page |
|------------------------|---|-----------------|--------|-----------------------------|----------------------------|
| 4007_F001 | System Reset Status Register 1 (RCM_SRS1) | 8 | R | 00h | 21.2.2/255 |
| 4007_F004 | Reset Pin Filter Control register (RCM_RPFC) | 8 | R/W | 00h | 21.2.3/256 |
| 4007_F005 | Reset Pin Filter Width register (RCM_RPFW) | 8 | R/W | 00h | 21.2.4/257 |
| 4007_F006 | Force Mode Register (RCM_FM) | 8 | R/W | 00h | 21.2.5/259 |
| 4007_F007 | Mode Register (RCM_MR) | 8 | R/W | See section | 21.2.6/259 |
| 4007_F008 | Sticky System Reset Status Register 0 (RCM_SSRS0) | 8 | R/W | 82h | 21.2.7/260 |
| 4007_F009 | Sticky System Reset Status Register 1 (RCM_SSRS1) | 8 | R/W | 00h | 21.2.8/261 |

21.2.1 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 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 | 0 | 0 | LVD | WAKEUP |
| Write | | | | | | | | |
| Reset | 1 | 0 | 0 | 0 | 0 | 0 | 1 | 0 |

RCM_SRS0 field descriptions

| Field | Description |
|----------|--|
| 7 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 1 Reset caused by POR</p> |

Table continues on the next page...

RCM_SRS0 field descriptions (continued)

| Field | Description |
|--------------|---|
| 6 PIN | <p>External Reset Pin</p> <p>Indicates a reset has been caused by an active-low level on the external <code>RESET</code> pin.</p> <p>0 Reset not caused by external reset pin 1 Reset caused by external reset pin</p> |
| 5 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 <code>SIM_COPCTRL[COPT]</code>.</p> <p>0 Reset not caused by watchdog timeout 1 Reset caused by watchdog timeout</p> |
| 4–3 Reserved | <p>This field is reserved.</p> <p>This read-only field is reserved and always has the value 0.</p> |
| 2 Reserved | <p>This field is reserved.</p> <p>This read-only field is reserved and always has the value 0.</p> |
| 1 LVD | <p>Low-Voltage Detect Reset</p> <p>If <code>PMC_LVDSC1[LVDRE]</code> is set and the supply drops below the LVD trip voltage, an LVD reset occurs. This field is also set by POR.</p> <p>0 Reset not caused by LVD trip or POR 1 Reset caused by LVD trip or POR</p> |
| 0 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. 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 1 Reset caused by LLWU module wakeup source</p> |

21.2.2 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

Reset memory map and register descriptions

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 Reserved | This field is reserved. This read-only field is reserved and always has the value 0. |
| 6 Reserved | This field is reserved. This read-only field is reserved and always has the value 0. |
| 5 SACKERR | Stop Mode Acknowledge Error Reset 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. 0 Reset not caused by peripheral failure to acknowledge attempt to enter stop mode 1 Reset caused by peripheral failure to acknowledge attempt to enter stop mode |
| 4 Reserved | This field is reserved. This read-only field is reserved and always has the value 0. |
| 3 MDM_AP | MDM-AP System Reset Request Indicates a reset has been caused by the host debugger system setting of the System Reset Request bit in the MDM-AP Control Register. 0 Reset not caused by host debugger system setting of the System Reset Request bit 1 Reset caused by host debugger system setting of the System Reset Request bit |
| 2 SW | Software Indicates a reset has been caused by software setting of SYSRESETREQ bit in Application Interrupt and Reset Control Register in the ARM core. 0 Reset not caused by software setting of SYSRESETREQ bit 1 Reset caused by software setting of SYSRESETREQ bit |
| 1 LOCKUP | Core Lockup Indicates a reset has been caused by the ARM core indication of a LOCKUP event. 0 Reset not caused by core LOCKUP event 1 Reset caused by core LOCKUP event |
| 0 Reserved | This field is reserved. This read-only field is reserved and always has the value 0. |

21.2.3 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 | | | | | | RSTFLTSS | | RSTFLTSRW | |
| Write | 0 | 0 | 0 | 0 | | 0 | 0 | 0 | 0 |
| Reset | 0 | 0 | 0 | 0 | | 0 | 0 | 0 | 0 |

RCM_RPFC field descriptions

| Field | Description |
|-----------------|--|
| 7–3 Reserved | This field is reserved. This read-only field is reserved and always has the value 0. |
| 2 RSTFLTSS | Reset Pin Filter Select in Stop Mode Selects how the reset pin filter is enabled in Stop and VLPS modes , and also during VLLS mode. On exit from VLLS mode, this bit should be reconfigured before clearing PMC_REGSC[ACKISO]. 0 All filtering disabled 1 LPO clock filter enabled |
| RSTFLTSRW | Reset Pin Filter Select in Run and Wait Modes Selects how the reset pin filter is enabled in run and wait modes. 00 All filtering disabled 01 Bus clock filter enabled for normal operation 10 LPO clock filter enabled for normal operation 11 Reserved |

21.2.4 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 | 0 | 0 | 0 | 0 | | 0 | 0 | 0 | 0 |

RCM_RPFW field descriptions

| Field | Description | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
|-----------------|--|-------|-----------------------------|-------|-----------------------------|-------|-----------------------------|-------|-----------------------------|-------|-----------------------------|-------|-----------------------------|-------|-----------------------------|-------|-----------------------------|-------|-----------------------------|-------|------------------------------|-------|------------------------------|-------|------------------------------|-------|------------------------------|-------|------------------------------|-------|------------------------------|-------|------------------------------|-------|------------------------------|-------|------------------------------|-------|------------------------------|-------|------------------------------|-------|------------------------------|-------|------------------------------|-------|------------------------------|-------|------------------------------|-------|------------------------------|-------|------------------------------|-------|------------------------------|-------|------------------------------|-------|------------------------------|-------|------------------------------|-------|------------------------------|-------|------------------------------|
| 7–5 Reserved | This field is reserved. This read-only field is reserved and always has the value 0. | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| RSTFLTSEL | <p>Reset Pin Filter Bus Clock Select</p> <p>Selects the reset pin bus clock filter width.</p> <table> <tbody> <tr><td>00000</td><td>Bus clock filter count is 1</td></tr> <tr><td>00001</td><td>Bus clock filter count is 2</td></tr> <tr><td>00010</td><td>Bus clock filter count is 3</td></tr> <tr><td>00011</td><td>Bus clock filter count is 4</td></tr> <tr><td>00100</td><td>Bus clock filter count is 5</td></tr> <tr><td>00101</td><td>Bus clock filter count is 6</td></tr> <tr><td>00110</td><td>Bus clock filter count is 7</td></tr> <tr><td>00111</td><td>Bus clock filter count is 8</td></tr> <tr><td>01000</td><td>Bus clock filter count is 9</td></tr> <tr><td>01001</td><td>Bus clock filter count is 10</td></tr> <tr><td>01010</td><td>Bus clock filter count is 11</td></tr> <tr><td>01011</td><td>Bus clock filter count is 12</td></tr> <tr><td>01100</td><td>Bus clock filter count is 13</td></tr> <tr><td>01101</td><td>Bus clock filter count is 14</td></tr> <tr><td>01110</td><td>Bus clock filter count is 15</td></tr> <tr><td>01111</td><td>Bus clock filter count is 16</td></tr> <tr><td>10000</td><td>Bus clock filter count is 17</td></tr> <tr><td>10001</td><td>Bus clock filter count is 18</td></tr> <tr><td>10010</td><td>Bus clock filter count is 19</td></tr> <tr><td>10011</td><td>Bus clock filter count is 20</td></tr> <tr><td>10100</td><td>Bus clock filter count is 21</td></tr> <tr><td>10101</td><td>Bus clock filter count is 22</td></tr> <tr><td>10110</td><td>Bus clock filter count is 23</td></tr> <tr><td>10111</td><td>Bus clock filter count is 24</td></tr> <tr><td>11000</td><td>Bus clock filter count is 25</td></tr> <tr><td>11001</td><td>Bus clock filter count is 26</td></tr> <tr><td>11010</td><td>Bus clock filter count is 27</td></tr> <tr><td>11011</td><td>Bus clock filter count is 28</td></tr> <tr><td>11100</td><td>Bus clock filter count is 29</td></tr> <tr><td>11101</td><td>Bus clock filter count is 30</td></tr> <tr><td>11110</td><td>Bus clock filter count is 31</td></tr> <tr><td>11111</td><td>Bus clock filter count is 32</td></tr> </tbody> </table> | 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 |
| 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 | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |

21.2.5 Force Mode Register (RCM_FM)

NOTE

The reset values of the bits in the FORCEROM field are for Chip POR only. They are unaffected by other reset types.

Address: 4007_F000h base + 6h offset = 4007_F006h

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|-------|---|---|---|---|---|---|----------|---|
| Read | | | 0 | | | | FORCEROM | 0 |
| Write | | | | | | | | |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

RCM_FM field descriptions

| Field | Description |
|-----------------|---|
| 7–3 Reserved | This field is reserved. This read-only field is reserved and always has the value 0. |
| 2–1 FORCEROM | Force ROM Boot When either bit is set, will force boot from ROM during all subsequent system resets. 00 No effect 01 Force boot from ROM with RCM_MR[1] set. 10 Force boot from ROM with RCM_MR[2] set. 11 Force boot from ROM with RCM_MR[2:1] set. |
| 0 Reserved | This field is reserved. This read-only field is reserved and always has the value 0. |

21.2.6 Mode Register (RCM_MR)

This register includes status flags to indicate the state of the mode pins during the last Chip Reset.

Address: 4007_F000h base + 7h offset = 4007_F007h

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|-------|---|---|---|---|---|---------|---|---|
| Read | | | 0 | | | BOOTROM | 0 | |
| Write | | | | | | w1c | | |
| Reset | 0 | 0 | 0 | 0 | 0 | * | * | 0 |

* Notes:

- BOOTROM field: The reset state of this register depends on the boot mode.

RCM_MR field descriptions

| Field | Description |
|-----------------|--|
| 7–3 Reserved | This field is reserved. This read-only field is reserved and always has the value 0. |
| 2–1 BOOTROM | Boot ROM Configuration Indicates the boot source, the boot source remains set until the next System Reset or software can write logic one to clear the corresponding mode bit. While either bit is set, the NMI input is disabled and the vector table is relocated to the ROM base address at 0x1C00_0000. These bits should be cleared by writing logic one before executing any code from either Flash or SRAM. 00 Boot from Flash 01 Boot from ROM due to BOOTCFG0 pin assertion 10 Boot from ROM due to FOPT[7] configuration 11 Boot from ROM due to both BOOTCFG0 pin assertion and FOPT[7] configuration |
| 0 Reserved | This field is reserved. This read-only field is reserved and always has the value 0. |

21.2.7 Sticky System Reset Status Register 0 (RCM_SSRS0)

This register includes status flags to indicate all reset sources since the last POR, LVD or VLLS Wakeup that have not been cleared by software. Software can clear the status flags by writing a logic one to a flag.

Address: 4007_F000h base + 8h offset = 4007_F008h

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|-------|------|------|-------|---|---|---|------|---------|
| Read | SPOR | SPIN | SWDOG | | 0 | 0 | SLVD | SWAKEUP |
| Write | w1c | w1c | w1c | | | | w1c | w1c |
| Reset | 1 | 0 | 0 | 0 | 0 | 0 | 1 | 0 |

RCM_SSRS0 field descriptions

| Field | Description |
|-----------|---|
| 7 SPOR | Sticky Power-On Reset 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. 0 Reset not caused by POR 1 Reset caused by POR |
| 6 SPIN | Sticky External Reset Pin Indicates a reset has been caused by an active-low level on the external RESET pin. |

Table continues on the next page...

RCM_SSRS0 field descriptions (continued)

| Field | Description |
|--------------|--|
| | 0 Reset not caused by external reset pin 1 Reset caused by external reset pin |
| 5 SWDOG | Sticky Watchdog 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 SIM_COPCTRL[COPT]. 0 Reset not caused by watchdog timeout 1 Reset caused by watchdog timeout |
| 4–3 Reserved | This field is reserved. This read-only field is reserved and always has the value 0. |
| 2 Reserved | This field is reserved. This read-only field is reserved and always has the value 0. |
| 1 SLVD | Sticky Low-Voltage Detect Reset If PMC_LVDSC1[LVDRE] is set and the supply drops below the LVD trip voltage, an LVD reset occurs. This field is also set by POR. 0 Reset not caused by LVD trip or POR 1 Reset caused by LVD trip or POR |
| 0 SWAKEUP | Sticky Low Leakage Wakeup Reset Indicates a reset has been caused by an enabled LLWU module wakeup source while the chip was in a low leakage mode. Any enabled wakeup source in a VLLSx mode causes a reset. 0 Reset not caused by LLWU module wakeup source 1 Reset caused by LLWU module wakeup source |

21.2.8 Sticky System Reset Status Register 1 (RCM_SSRS1)

This register includes status flags to indicate all reset sources since the last POR, LVD or VLLS Wakeup that have not been cleared by software. Software can clear the status flags by writing a logic one to a flag.

Address: 4007_F000h base + 9h offset = 4007_F009h

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|-------|---|---|----------|---|---------|-----|---------|---|
| Read | 0 | 0 | SSACKERR | 0 | SMDM_AP | SSW | SLOCKUP | 0 |
| Write | | | w1c | | w1c | w1c | w1c | |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

RCM_SSRS1 field descriptions

| Field | Description |
|---------------|--|
| 7 Reserved | This field is reserved. This read-only field is reserved and always has the value 0. |
| 6 Reserved | This field is reserved. This read-only field is reserved and always has the value 0. |
| 5 SSACKERR | Sticky Stop Mode Acknowledge Error Reset 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. 0 Reset not caused by peripheral failure to acknowledge attempt to enter stop mode 1 Reset caused by peripheral failure to acknowledge attempt to enter stop mode |
| 4 Reserved | This field is reserved. This read-only field is reserved and always has the value 0. |
| 3 SMDM_AP | Sticky MDM-AP System Reset Request Indicates a reset has been caused by the host debugger system setting of the System Reset Request bit in the MDM-AP Control Register. 0 Reset not caused by host debugger system setting of the System Reset Request bit 1 Reset caused by host debugger system setting of the System Reset Request bit |
| 2 SSW | Sticky Software Indicates a reset has been caused by software setting of SYSRESETREQ bit in Application Interrupt and Reset Control Register in the ARM core. 0 Reset not caused by software setting of SYSRESETREQ bit 1 Reset caused by software setting of SYSRESETREQ bit |
| 1 SLOCKUP | Sticky Core Lockup Indicates a reset has been caused by the ARM core indication of a LOCKUP event. 0 Reset not caused by core LOCKUP event 1 Reset caused by core LOCKUP event |
| 0 Reserved | This field is reserved. This read-only field is reserved and always has the value 0. |

Chapter 22

Bit Manipulation Engine (BME)

22.1 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.

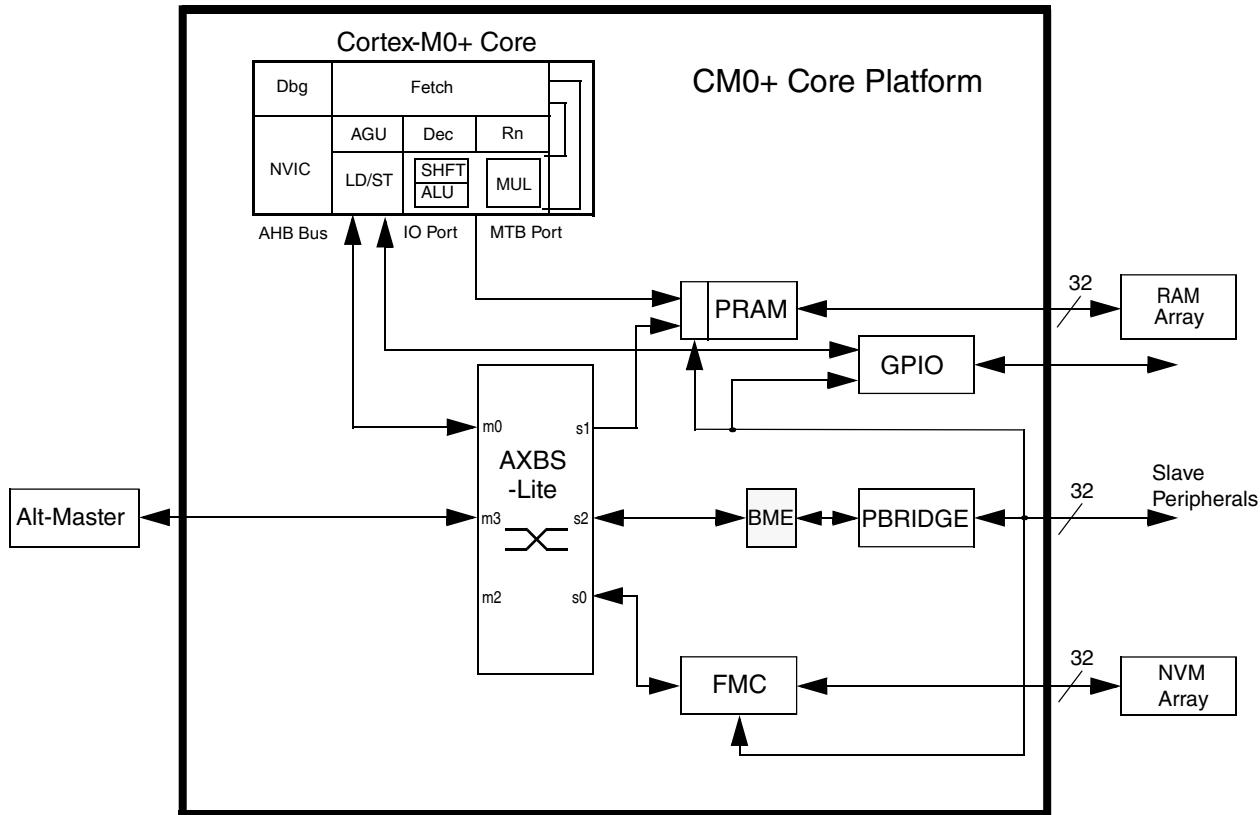
By combining the basic load and store instructions of the ARM Cortex-M instruction set architecture (v6M, v7M) with the concept of decorated storage provided by the BME, the resulting implementation provides a robust and efficient read-modify-write capability to this class of ultra low-end microcontrollers. 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_0000¹. The decoration semantic is embedded into address bits[28:19], creating a 448 MB space at addresses 0x4400_0000–0x5FFF_FFFF for AIPS; 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 - 0x400F_EFFF are error terminated due to an illegal address.

22.1.1 Overview

The following figure is a generic block diagram of the processor core and platform for this class of ultra low-end microcontrollers.



Note: BME can be accessed only by the core.

Figure 22-1. 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.

22.1.2 Features

The key features of the BME include:

- Lightweight implementation of decorated storage for selected address spaces

- Additional access semantics encoded into the reference address
- Resides between a crossbar switch slave port and a peripheral bridge bus controller
- Two-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} 1-bit 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

22.1.3 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 strictly on the basis of memory addresses for accesses to the 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.

22.2 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.

22.3 Functional description

Information found here 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.

22.3.1 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 peripheral bit field insert operation is shown as follows:

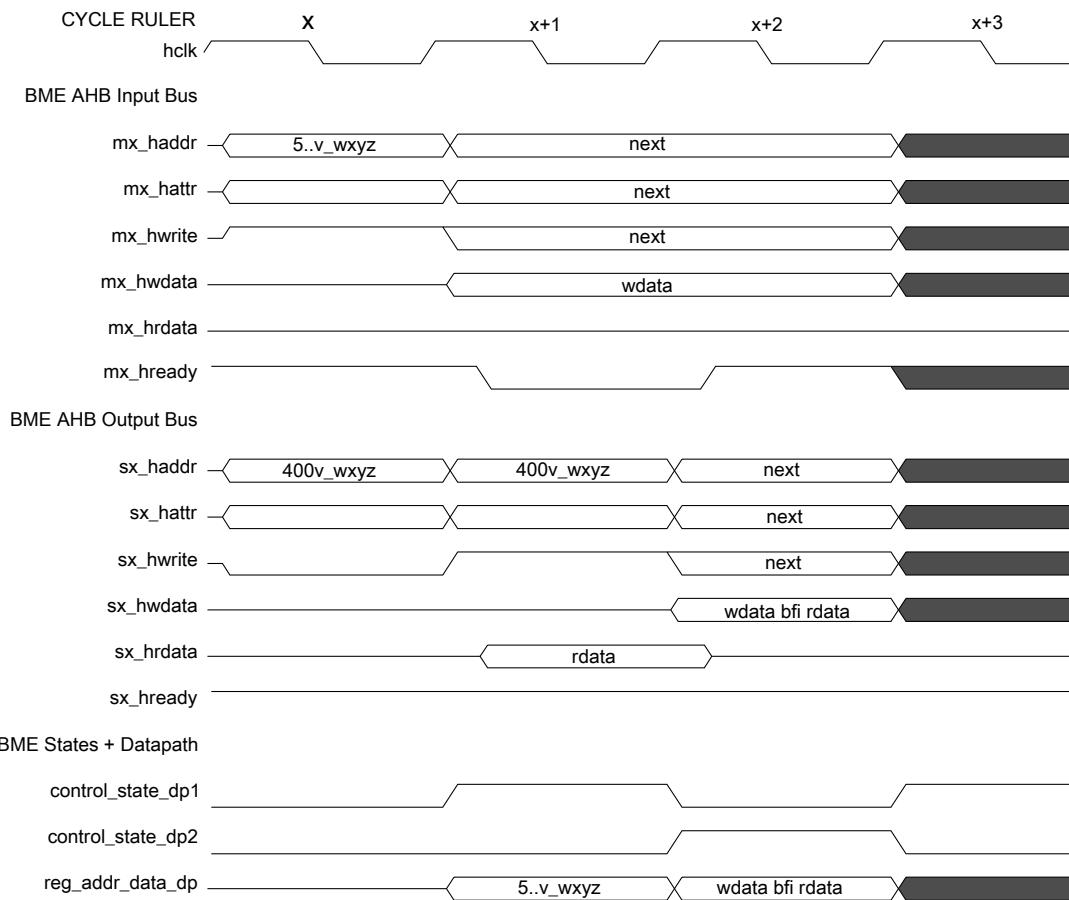


Figure 22-2. Decorated store: bit field insert timing diagram

All the decorated store operations follow the same execution template shown in [Figure 22-2](#), a two-cycle read-modify-write operation:

1. Cycle x, 1st AHB address phase: Write from input bus is translated into a read operation on the output bus using the actual memory address (with the decoration removed) and then captured in a register.
2. Cycle x+1, 2nd AHB address phase: Write access with the registered (but actual) memory address is output
3. Cycle x+1, 1st AHB data phase: Memory read data is modified using the input bus write data and the function defined by the decoration and captured in a data register; the input bus cycle is stalled.
4. Cycle x+2, 2nd AHB data phase: Registered write data is sourced onto the output write data bus.

NOTE

Any wait states inserted by the slave device are simply passed through the BME back to the master input bus, stalling the AHB transaction cycle for cycle.

22.3.1.1 Decorated store logical AND (AND)

This command performs an atomic read-modify-write of the referenced memory location.

1. First, the location is read;
2. It is then modified by performing a logical AND operation using the write data operand sourced for the system bus cycle
3. Finally, the result of the AND 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 |
|--------|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----------|----|---|---|---|---|---|---|---|---|---|---|
| ioandb | 0 | * | * | 0 | 0 | 1 | - | - | - | - | - | - | | | | | | | | | mem_addr | | | | | | | | | | | |
| ioandh | 0 | * | * | 0 | 0 | 1 | - | - | - | - | - | - | | | | | | | | | mem_addr | | | | | | | | | 0 | | |
| ioandw | 0 | * | * | 0 | 0 | 1 | - | - | - | - | - | - | | | | | | | | | mem_addr | | | | | | | | 0 | 0 | | |

Figure 22-3. Decorated store address: logical AND

See [Figure 22-3](#) where $\text{addr}[30:29] = 10$ for peripheral, $\text{addr}[28:26] = 001$ specifies the AND operation, and $\text{mem_addr}[19:0]$ specifies the address offset into the space based at $0x4000_0000$ for peripherals. The "-" indicates an address bit "don't care".

The decorated AND write operation is defined in the following pseudo-code as:

```
ioand<sz>(accessAddress, wdata)           // decorated store AND
tmp    = mem[accessAddress & 0xE00FFFFF, size] // memory read
tmp    = tmp & wdata                         // modify
mem[accessAddress & 0xE00FFFFF, size] = tmp // memory write
```

where the operand size $<\text{sz}>$ is defined as b(yte, 8-bit), h(alfword, 16-bit) and w(ord, 32-bit). This notation is used throughout the document.

In the cycle definition tables, the notations AHB_ap and AHB_dp refer to the address and data phases of the BME AHB transaction. The cycle-by-cycle BME operations are detailed in the following table.

Table 22-1. Cycle definitions of decorated store: logical AND

| Pipeline stage | Cycle | | |
|----------------|---|---|--------|
| | x | x+1 | x+2 |
| BME AHB_ap | Forward addr to memory; Decode decoration; Convert | Recirculate captured addr + attr to memory as slave_wt | <next> |

Table continues on the next page...

Table 22-1. Cycle definitions of decorated store: logical AND (continued)

| Pipeline stage | Cycle | | |
|----------------|---|--|--|
| | x | x+1 | x+2 |
| | master_wt to slave_rd; Capture address, attributes | | |
| BME AHB_dp | <previous> | Perform memory read; Form (rdata & wdata) and capture destination data in register | Perform write sending registered data to memory |

22.3.1.2 Decorated store logical OR (OR)

This command performs an atomic read-modify-write of the referenced memory location.

1. First, the location is read.
2. It is then modified by performing a logical OR operation using the write data operand sourced for the system bus cycle.
3. 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 | * | * | 0 | 1 | 0 | - | - | - | - | - | - | - | - | - | - | - | - | - | - | mem_addr | | | | | | | | | | | |
| ioorh | 0 | * | * | 0 | 1 | 0 | - | - | - | - | - | - | - | - | - | - | - | - | - | - | mem_addr | | | | | | | | | 0 | | |
| ioorw | 0 | * | * | 0 | 1 | 0 | - | - | - | - | - | - | - | - | - | - | - | - | - | - | mem_addr | | | | | | | | | 0 | | |

Figure 22-4. Decorated address store: logical OR

See [Figure 22-4](#), where $\text{addr}[30:29] = 10$ for peripheral, $\text{addr}[28:26] = 010$ specifies the OR operation, and $\text{mem_addr}[19:0]$ specifies the address offset into the space based at $0x4000_0000$ for peripherals. 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 & 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 22-2. Cycle definitions of decorated store: logical OR

| Pipeline stage | Cycle | | |
|----------------|--|--|--|
| | x | x+1 | x+2 |
| BME AHB_ap | Forward addr to memory; Decode decoration; Convert master_wt to slave_rd; Capture address, attributes | Recirculate captured addr + attr to memory as slave_wt | <next> |
| BME AHB_dp | <previous> | Perform memory read; Form (rdata wdata) and capture destination data in register | Perform write sending registered data to memory |

22.3.1.3 Decorated store logical XOR (XOR)

This command performs an atomic read-modify-write of the referenced memory location.

1. First, the location is read.
2. It is then modified by performing a logical XOR (exclusive-OR) operation using the write data operand sourced for the system bus cycle.
3. 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 | * | * | 0 | 1 | 1 | - | - | - | - | - | - | | | | | | | | | | | | | | | | | | | | |
| ioxorh | 0 | * | * | 0 | 1 | 1 | - | - | - | - | - | - | | | | | | | | | | | | | | | | | | 0 | | |
| ioxorw | 0 | * | * | 0 | 1 | 1 | - | - | - | - | - | - | | | | | | | | | | | | | | | | | | 0 | 0 | |

Figure 22-5. Decorated address store: logical XOR

See [Figure 22-5](#), where $\text{addr}[30:29] = 10$ for peripheral, $\text{addr}[28:26] = 011$ specifies the XOR operation, and $\text{mem_addr}[19:0]$ specifies the address offset into the peripheral space based at $0x4000_0000$ for peripherals. 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 22-3. Cycle definitions of decorated store: logical XOR

| Pipeline Stage | Cycle | | |
|----------------|--|--|--|
| | x | x+1 | x+2 |
| BME AHB_ap | Forward addr to memory; Decode decoration; Convert master_wt to slave_rd; Capture address, attributes | Recirculate captured addr + attr to memory as slave_wt | <next> |
| BME AHB_dp | <previous> | Perform memory read; Form (rdata ^ wdata) and capture destination data in register | Perform write sending registered data to memory |

22.3.1.4 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 | - | - | b | b | b | - | w | w | w | | | | | | | | | | | | | | | | | mem_addr | | |
| iobfih | 0 | * | * | 1 | - | b | b | b | b | w | w | w | w | | | | | | | | | | | | | | | | mem_addr | | 0 | |
| iobfiw | 0 | * | * | 1 | b | b | b | b | b | w | w | w | w | | | | | | | | | | | | | | | | mem_addr | | 0 | |

Figure 22-6. Decorated address store: bit field insert

where $\text{addr}[30:29] = 10$ for peripheral, $\text{addr}[28] = 1$ 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 for peripherals. 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.

Functional description

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
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-aligned 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 22-4. Cycle definitions of decorated store: bit field insert

| Pipeline stage | Cycle | | |
|----------------|--|---|--|
| | x | x+1 | x+2 |
| BME AHB_ap | Forward addr to memory; Decode decoration; Convert master_wt to slave_rd; Capture address, attributes | Recirculate captured addr + attr to memory as slave_wt | <next> |
| BME AHB_dp | <previous> | Perform memory read; Form bit mask; Form bitwise ((mask) ? wdata : rdata)) and capture destination data in register | Perform write sending registered data to memory |

22.3.2 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 two-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 second 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 peripheral load-and-set 1-bit operation is shown as follows.

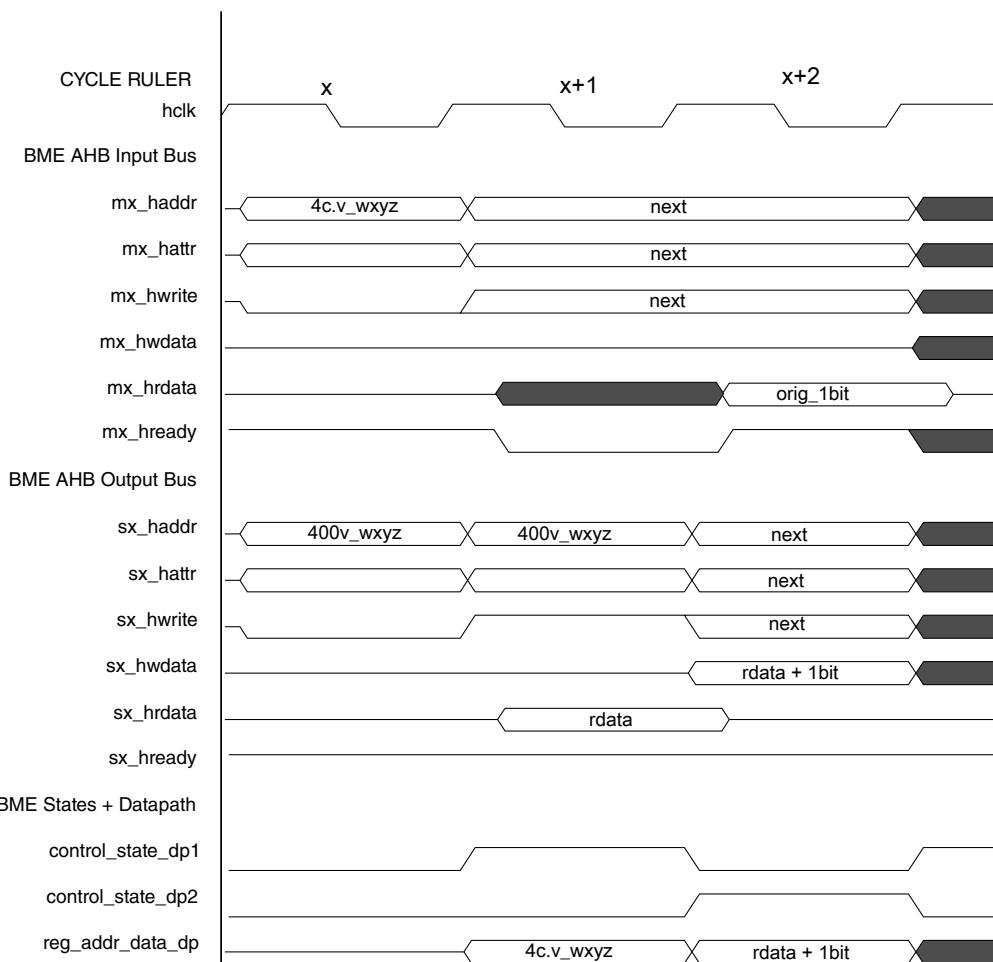


Figure 22-7. Decorated load: load-and-set 1-bit field insert timing diagram

Decorated load-and-{ set, clear } 1-bit operations follow the execution template shown in the above figure: a 2-cycle read-modify-write operation:

1. Cycle x, first 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
2. Cycle x+1, second AHB address phase: Write access with the registered (but actual) memory address is output
3. Cycle x+1, first 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
4. Cycle x+2, second 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 slave device 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 peripheral bit field operation is shown in the following figure.

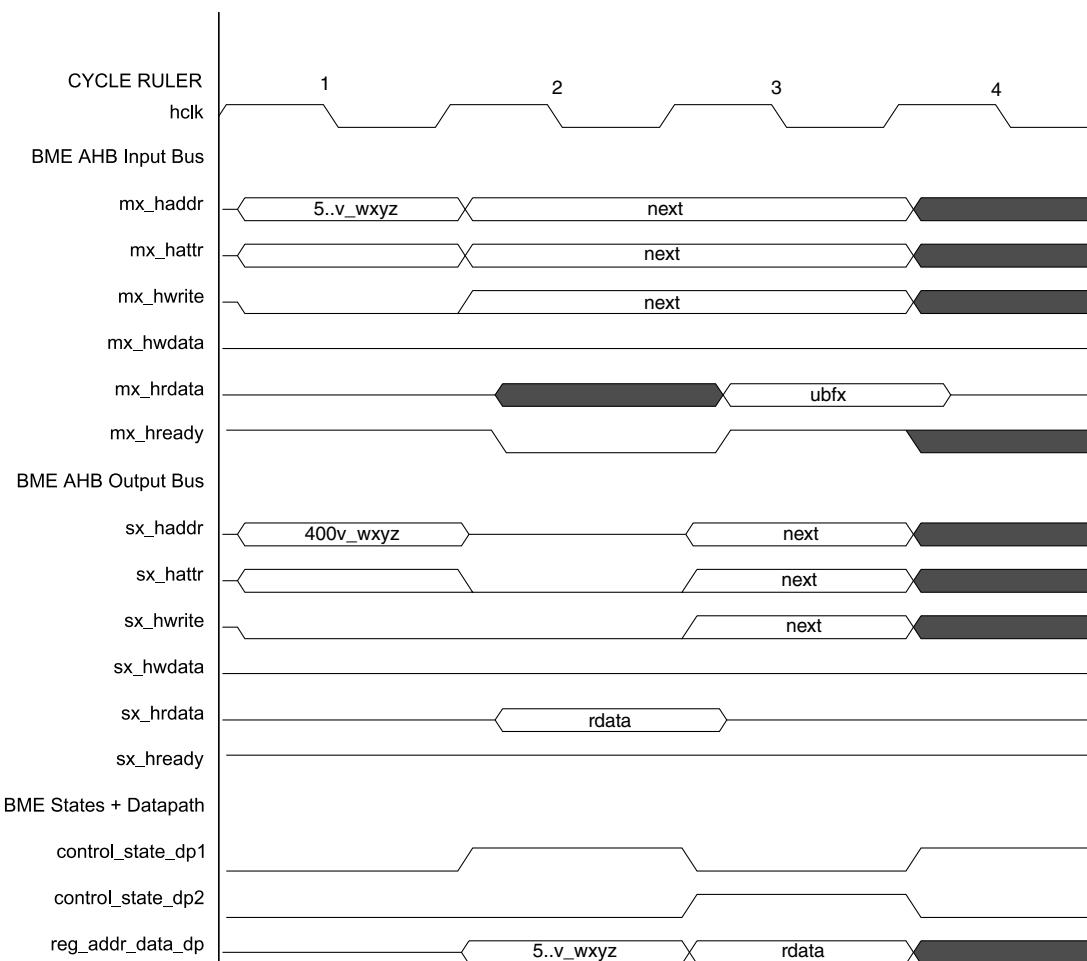


Figure 22-8. Decorated load: unsigned bit field insert timing diagram

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-aligned for proper alignment and driven onto the input read data bus

NOTE

Any wait states inserted by the slave device are simply passed through the BME back to the master input bus, stalling the AHB transaction cycle for cycle.

22.3.2.1 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 1-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 |
|---------|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|---|---|---|---|---|---|---|---|----------|---|
| ol ac1b | 0 | * | * | 0 | 1 | 0 | - | - | b | b | b | - | | | | | | | | | | | | | | | | | | | mem_addr | |
| ol ac1h | 0 | * | * | 0 | 1 | 0 | - | b | b | b | b | - | | | | | | | | | | | | | | | | | | 0 | mem_addr | |
| ol ac1w | 0 | * | * | 0 | 1 | 0 | b | b | b | b | b | - | | | | | | | | | | | | | | | | | 0 | 0 | mem_addr | |

Figure 22-9. Decorated load address: load-and-clear 1 bit

See [Figure 22-9](#) where $\text{addr}[30:29] = 10$ for peripheral, $\text{addr}[28:26] = 010$ 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 space based at 0x4000_0000 for peripheral. 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 = iolaci<sz>(accessAddress)           // decorated load-and-clear 1
tmp   = mem[accessAddress & 0xE00FFFF, size]  // memory read
mask  = 1 << b                            // generate bit mask
rdata = (tmp & mask) >> b                  // read data returned to core
tmp   = tmp & ~mask                          // modify
mem[accessAddress & 0xE00FFFF, size] = tmp    // memory write
```

The cycle-by-cycle BME operations are detailed in the following table.

Table 22-5. 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; 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; Perform write sending registered data to memory |

22.3.2.2 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 | |
|---------|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|---|---|---|---|---|---|---|---|---|---|---|
| iolaslb | 0 | * | * | 0 | 1 | 1 | - | - | b | b | b | - | | | | | | | | | | | | | | | | | | | | | |
| iolaslh | 0 | * | * | 0 | 1 | 1 | - | b | b | b | b | - | | | | | | | | | | | | | | | | | | | | | 0 |
| iolaslw | 0 | * | * | 0 | 1 | 1 | b | b | b | b | b | b | - | | | | | | | | | | | | | | | | | | 0 | 0 | |

Figure 22-10. Decorated load address: load-and-set 1 bit

where $\text{addr}[30:29] = 10$ for peripheral, $\text{addr}[28:26] = 011$ 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 space based at 0x4000_0000 for peripheral. 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 = iolasl<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
```

Functional description

```
tmp = tmp | mask           // modify
mem[accessAddress & 0xE00FFFFF, size] = tmp // memory write
```

The cycle-by-cycle BME operations are detailed in the following table.

Table 22-6. 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; 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; Perform write sending registered data to memory |

22.3.2.3 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 two-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, UBFX 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 UBFX operation is recommended to extract a single bit. 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 | - | - | b | b | b | - | w | w | w | | | | | | | | | | | | | | | | | | | |
| ioubfxh | 0 | * | * | 1 | - | b | b | b | b | w | w | w | w | | | | | | | | | | | | | | | | | | 0 | |
| ioubfxw | 0 | * | * | 1 | b | b | b | b | b | w | w | w | w | | | | | | | | | | | | | | | | | 0 | 0 | |

Figure 22-11. Decorated load address: unsigned bit field extract

See [Figure 22-11](#), where $\text{addr}[30:29] = 10$ for peripheral, $\text{addr}[28] = 1$ specifies the unsigned bit field extract operation, $\text{addr}[27:23]$ is "b", the LSB identifier, $\text{addr}[22:19]$ is "w", the bit field width minus 1 identifier, and $\text{mem_addr}[18:0]$ specifies the address

offset into the space based at 0x4000_0000 for peripheral. The "-" indicates an address bit "don't care". Note, unlike the other decorated load operations, UBFX 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 " $\text{container_width} - b$ " bits are actually extracted. The cycle-by-cycle BME operations are detailed in the following table.

Table 22-7. Cycle definitions of decorated load: unsigned bit field extract

| Pipeline Stage | Cycle | | |
|----------------|--|---|---|
| | x | x+1 | x+2 |
| BME AHB_ap | Forward addr to memory; Decode decoration; Capture address, attributes | Idle AHB address phase | <next> |
| BME AHB_dp | <previous> | Perform memory read; Form bit mask; Form (rdata & mask) and capture destination data in register | Logically right shift registered data; Return justified rdata to master |

22.3.3 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 UBXF 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 22-8. Decorated peripheral and GPIO address details

| Peripheral address space | Description |
|--------------------------|--|
| 0x4000_0000–0x4007_FFFF | Undecorated (normal) peripheral accesses |
| 0x4008_0000–0x400F_EFFF | Illegal addresses; attempted references are aborted and error terminated |
| 0x400F_F000–0x400F_FFFF | Undecorated (normal) GPIO accesses using standard address |
| 0x4010_0000–0x43FF_FFFF | Illegal addresses; attempted references are aborted and error terminated |
| 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, UBXF references to peripherals and GPIO only based at 0x4000_F000 |

22.4 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"); \
\
#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"); \
\
#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"); \
\
#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 23

Micro Trace Buffer (MTB)

23.1 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.

23.1.1 Overview

A generic block diagram of the processor core and platform for this class of ultra low-end microcontrollers is shown as follows:

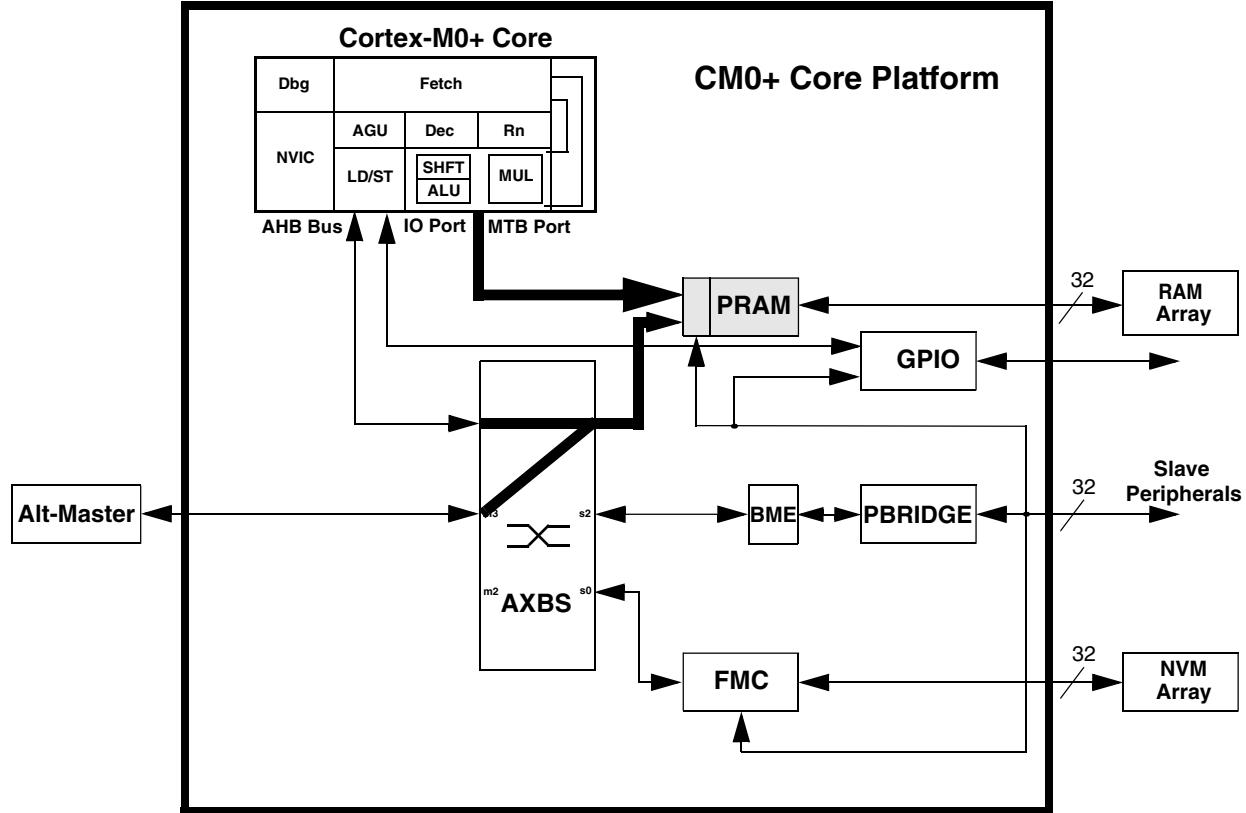


Figure 23-1. Generic Cortex-M0+ core platform block diagram

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.

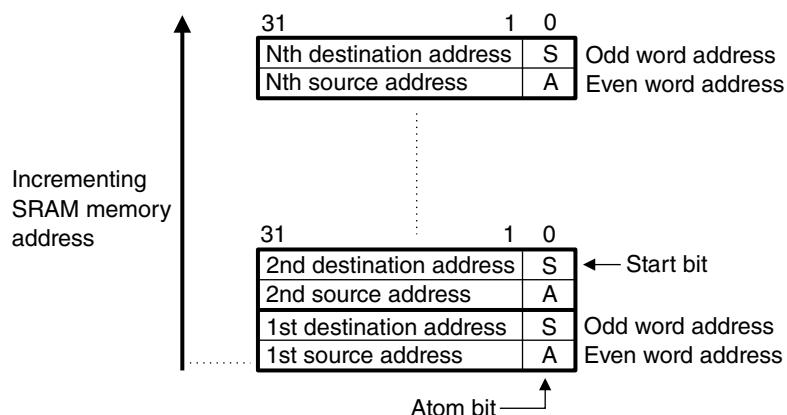


Figure 23-2. MTB execution trace storage format

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.

23.1.2 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

23.1.3 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 strictly on the basis of 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.

23.2 External signal description

The MTB_RAM and MTB_DWT modules do not directly support any external interfaces.

The internal interface 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 23-1. 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. |

Memory map and register definition

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.

23.3 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

23.3.1 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 | 23.3.1.1/ 290 |
| F000_0004 | MTB Master Register (MTB_MASTER) | 32 | R/W | See section | 23.3.1.2/ 291 |
| F000_0008 | MTB Flow Register (MTB_FLOW) | 32 | R/W | Undefined | 23.3.1.3/ 293 |
| F000_000C | MTB Base Register (MTB_BASE) | 32 | R | Undefined | 23.3.1.4/ 295 |
| F000_0F00 | Integration Mode Control Register (MTB_MODECTRL) | 32 | R | 0000_0000h | 23.3.1.5/ 295 |

Table continues on the next page...

MTB memory map (continued)

| Absolute address (hex) | Register name | Width (in bits) | Access | Reset value | Section/page |
|------------------------|---|-----------------|--------|-----------------------------|-------------------------------|
| F000_0FA0 | Claim TAG Set Register (MTB_TAGSET) | 32 | R | 0000_0000h | 23.3.1.6/296 |
| F000_0FA4 | Claim TAG Clear Register (MTB_TAGCLEAR) | 32 | R | 0000_0000h | 23.3.1.7/296 |
| F000_0FB0 | Lock Access Register (MTB_LOCKACCESS) | 32 | R | 0000_0000h | 23.3.1.8/297 |
| F000_0FB4 | Lock Status Register (MTB_LOCKSTAT) | 32 | R | 0000_0000h | 23.3.1.9/297 |
| F000_0FB8 | Authentication Status Register (MTB_AUTHSTAT) | 32 | R | 0000_0000h | 23.3.1.10/297 |
| F000_0FBC | Device Architecture Register (MTB_DEVICEARCH) | 32 | R | 4770_0A31h | 23.3.1.11/298 |
| F000_0FC8 | Device Configuration Register (MTB_DEVICECFG) | 32 | R | 0000_0000h | 23.3.1.12/299 |
| F000_0FCC | Device Type Identifier Register (MTB_DEVICETYPID) | 32 | R | 0000_0031h | 23.3.1.13/299 |
| F000_0FD0 | Peripheral ID Register (MTB_PERIPHID4) | 32 | R | See section | 23.3.1.14/300 |
| F000_0FD4 | Peripheral ID Register (MTB_PERIPHID5) | 32 | R | See section | 23.3.1.14/300 |
| F000_0FD8 | Peripheral ID Register (MTB_PERIPHID6) | 32 | R | See section | 23.3.1.14/300 |
| F000_0FDC | Peripheral ID Register (MTB_PERIPHID7) | 32 | R | See section | 23.3.1.14/300 |
| F000_0FE0 | Peripheral ID Register (MTB_PERIPHID0) | 32 | R | See section | 23.3.1.14/300 |
| F000_0FE4 | Peripheral ID Register (MTB_PERIPHID1) | 32 | R | See section | 23.3.1.14/300 |
| F000_0FE8 | Peripheral ID Register (MTB_PERIPHID2) | 32 | R | See section | 23.3.1.14/300 |
| F000_0FEC | Peripheral ID Register (MTB_PERIPHID3) | 32 | R | See section | 23.3.1.14/300 |
| F000_0FF0 | Component ID Register (MTB_COMPID0) | 32 | R | See section | 23.3.1.15/300 |
| F000_0FF4 | Component ID Register (MTB_COMPID1) | 32 | R | See section | 23.3.1.15/300 |
| F000_0FF8 | Component ID Register (MTB_COMPID2) | 32 | R | See section | 23.3.1.15/300 |
| F000_0FFC | Component ID Register (MTB_COMPID3) | 32 | R | See section | 23.3.1.15/300 |

23.3.1.1 MTB Position Register (MTB_POSITION)

The MTB_POSITION register contains the Trace Write Address Pointer and Wrap fields. 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.

The base address of the system RAM in the memory map dictates special consideration for the placement of the MTB. Consider the following guidelines:

For the standard configuration where the size of the MTB is $\leq 25\%$ of the total RAM capacity, it is recommended the MTB be based at the address defined by the MTB_BASE register. The read-only MTB_BASE register is defined by the expression $(0x2000_0000 - (\text{RAM_Size}/4))$. For this configuration, the MTB_POSITION register is initialized to MTB_BASE & 0x0000_7FF8.

If the size of the MTB is more than 25% but less than or equal to 50% of the total RAM capacity, it is recommended the MTB be based at address 0x2000_0000. In this configuration, the MTB_POSITION register is initialized to $(0x2000_0000 \& 0x0000_7FF8) = 0x0000_0000$.

Following these two suggested placements provides a full-featured circular memory buffer containing program trace packets.

In the unlikely event an even larger trace buffer is required, a write-once capacity of 75% of the total RAM capacity can be based at address 0x2000_0000. The MTB_POSITION register is initialized to $(0x2000_0000 \& 0x0000_7FF8) = 0x0000_0000$. However, this configuration cannot support operation as a circular queue and instead requires the use of the MTB_FLOW[WATERMARK] capability to automatically disable tracing or halting the processor as the number of packet writes approach the buffer capacity. See the MTB_FLOW register description for more details.

Address: F000_0000h base + 0h offset = F000_0000h

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

- * Notes:
- x = Undefined at reset.

MTB_POSITION field descriptions

| Field | Description |
|-----------------|---|
| 31–3 POINTER | <p>Trace Packet Address Pointer[28:0]</p> <p>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 where the next trace packet is written. Therefore, it points to an unused location and is automatically incremented.</p> <p>A debug agent can calculate the system memory map address for the current location in the MTB using the following "generic" equation:</p> <pre>Given mtb_size = 1 << (MTB_MASTER[MASK] + 4), systemAddress = MTB_BASE + (((MTB_POSITION & 0xFFFF_FFF8) + (mtb_size - (MTB_BASE & (mtb_size-1))) & 0x0000_7FF8);</pre> <p>For this device, a simpler expression also applies. See the following pseudo-code:</p> <pre>if ((MTB_POSITION >> 13) == 0x3) systemAddress = (0x1FFF << 16) + (0x1 << 15) + (MTB_POSITION & 0x7FF8); else systemAddress = (0x2000 << 16) + (0x0 << 15) + (MTB_POSITION & 0x7FF8);</pre> <p>NOTE: The size of the RAM is parameterized and the most significant bits of the POINTER field are RAZ/WI.</p> <p>For these devices, POSITION[31:15] == POSITION[POINTER[28:12]] are RAZ/WI. Therefore, the active bits in this field are POSITION[14:3] == POSITION[POINTER[11:0]].</p> |
| 2 WRAP | <p>WRAP</p> <p>This field is set to 1 automatically when the POINTER value wraps as determined by the MTB_MASTER[MASK] field in the MASTER Trace Control Register. A debug agent might use the WRAP field to determine whether the trace information above and below the pointer address is valid.</p> |
| Reserved | <p>This field is reserved.</p> <p>This read-only field is reserved and always has the value 0.</p> |

23.3.1.2 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 MTB_MASTER[EN] or MTB_MASTER[TSTARTEN] are set to 1, the software must initialize the MTB_POSITION and MTB_FLOW registers.

If MTB_FLOW[WATERMARK] is used to stop tracing or to halt the processor, MTB_MASTER[MASK] must still be set to a value that prevents MTB_POSITION[POINTER] from wrapping before it reaches the MTB_FLOW[WATERMARK] value.

NOTE

The format of this mask field is different than 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 | | SFRWPRI | TSTOPEN | TSTARTEN | | | MASK |
| W | | | | | | | | | | | | | | | | |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 1 | 0 | 0 | x* | x* | x* | x* | x* |

* Notes:

- x = Undefined at reset.

MTB_MASTER field descriptions

| Field | Description |
|-------------------|---|
| 31 EN | Main Trace Enable When this field is 1, trace data is written into the RAM memory location addressed by MTB_POSITION[POSITION]. The MTB_POSITION[POSITION] value auto increments after the trace data packet is written. EN can be automatically set to 0 using the MTB_FLOW[WATERMARK] field and the MTB_FLOW[AUTOSTOP] bit. EN is automatically set to 1 if TSTARTEN is 1 and the TSTART signal is HIGH. EN is automatically set to 0 if TSTOPEN is 1 and the TSTOP signal is HIGH. NOTE: If EN is set to 0 because MTB_FLOW[WATERMARK] is set, then it is not automatically set to 1 if TSTARTEN is 1 and the TSTART input is HIGH. In this case, tracing can only be restarted if MTB_FLOW[WATERMARK] or MTB_POSITION[POSITION] value is changed by software. |
| 30–10 Reserved | This field is reserved. This read-only field is reserved and always has the value 0. |
| 9 HALTREQ | Halt Request This field 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. HALTREQ can be automatically set to 1 using MTB_FLOW[WATERMARK]. |
| 8 RAMPRIV | RAM Privilege If this field is 0, then user or privileged AHB read and write accesses to the RAM are permitted. If this field 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. |

Table continues on the next page...

MTB_MASTER field descriptions (continued)

| Field | Description |
|---------------|--|
| 7 SFRWPRI | Special Function Register Write Privilege If this field is 0, then user or privileged AHB read and write accesses to the MTB_RAM Special Function Registers (programming model) are permitted. If this field 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 TSTOPEN | Trace Stop Input Enable If this field is 1 and the TSTOP signal is HIGH, then EN is set to 0. If a trace packet is being written to memory, the write is completed before tracing is stopped. |
| 5 TSTARTEN | Trace Start Input Enable If this field is 1 and the TSTART signal is HIGH, then EN is set to 1. Tracing continues until a stop condition occurs. |
| MASK | Mask 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 2, the MTB_POSITION[WRAP] bit is set to 1, the MTB_POSITION[MASK+3:3] == MTB_POSITION[POSITION[MASK:0]] bits are set to 0, and the MTB_POSITION[14:MASK+3] == MTB_POSITION[POSITION[11:MASK+1]] bits remain unchanged. 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. As detailed in the MTB_POSITION description, typical "upper limits" for the MTB size are RAM_Size/4 or RAM_Size/2. Values greater than the maximum have the same effect as the maximum. |

23.3.1.3 MTB Flow Register (MTB_FLOW)

The MTB_FLOW register contains the watermark address and the autostop/autohalt control bits.

If tracing is stopped using the watermark autostop feature, it cannot be restarted until software clears the watermark autostop. This can be achieved in one of the following ways:

- Changing the MTB_POSITION[POSITION] field value to point to the beginning of the trace buffer, or
- Setting MTB_FLOW[AUTOSTOP] = 0.

A debug agent can use MTB_FLOW[AUTOSTOP] to fill the trace buffer once only without halting the processor.

A debug agent can use MTB_FLOW[AUTOHALT] to fill the trace buffer once before causing the Cortex-M0+ processor to enter the Debug state. To enter Debug state, the Cortex-M0+ processor might have to perform additional branch type operations. Therefore, the MTB_FLOW[WATERMARK] field must be set below the final entry in the trace buffer region.

Memory map and register definition

Address: F000_0000h base + 8h offset = F000_0008h

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

* Notes:

- x = Undefined at reset.

MTB_FLOW field descriptions

| Field | Description |
|-------------------|--|
| 31–3 WATERMARK | WATERMARK[28:0] This field contains an address in the same format as the MTB_POSITION[POSITION] field. When MTB_POSITION[POSITION] matches the WATERMARK field value, actions defined by the AUTOHALT and AUTOSTOP bits are performed. |
| 2 Reserved | This field is reserved. This read-only field is reserved and always has the value 0. |
| 1 AUTOHALT | AUTOHALT If this field is 1 and WATERMARK is equal to MTB_POSITION[POSITION], then MTB_MASTER[HALTREQ] is automatically set to 1. If the DBGEN signal is HIGH, the MTB asserts this halt request to the Cortex-M0+ processor by asserting the EDBGREQ signal. |
| 0 AUTOSTOP | AUTOSTOP If this field is 1 and WATERMARK is equal to MTB_POSITION[POSITION], then MTB_MASTER[EN] is automatically set to 0. This stops tracing. |

23.3.1.4 MTB Base Register (MTB_BASE)

The read-only MTB_BASE Register indicates where the RAM is located in the system 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 this device, the base address is defined by the expression: MTB_BASE[BASEADDR] = 0x2000_0000 - (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*

* Notes:

- x = Undefined at reset.

MTB_BASE field descriptions

| Field | Description |
|----------|---|
| BASEADDR | BASEADDR This value is defined with a hardwired signal and the expression: 0x2000_0000 - (RAM_Size/4). For example, if the total RAM capacity is 16 KB, this field is 0xFFFF_F000. |

23.3.1.5 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 |
|----------|--|
| MODECTRL | MODECTRL Hardwired to 0x0000_0000 |

23.3.1.6 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_OF0A0h

| 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 |
|--------|------------------------------------|
| TAGSET | TAGSET Hardwired to 0x0000_0000 |

23.3.1.7 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_OF0A4h

| 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 |
|----------|--------------------------------------|
| TAGCLEAR | TAGCLEAR Hardwired to 0x0000_0000 |

23.3.1.8 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 |
|------------|--------------------------|
| LOCKACCESS | Hardwired to 0x0000_0000 |

23.3.1.9 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 |
|----------|--------------------------------------|
| LOCKSTAT | LOCKSTAT Hardwired to 0x0000_0000 |

23.3.1.10 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.

Memory map and register definition

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 Reserved | This field is reserved. This read-only field is reserved and always has the value 0. |
| 3 Reserved | BIT3 This read-only field is reserved and always has the value 1. |
| 2 BIT2 | BIT2 Connected to NIDEN or DBGEN signal. |
| 1 Reserved | BIT1 This read-only field is reserved and always has the value 1. |
| 0 BIT0 | Connected to DBGEN. |

23.3.1.11 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 | 1 | 0 | 1 | 1 | 1 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 1 | 0 | 0 | 0 | 1 | 1 | 0 | 0 | 0 | 1 | | |

MTB_DEVICEARCH field descriptions

| Field | Description |
|------------|---|
| DEVICEARCH | DEVICEARCH Hardwired to 0x4770_0A31. |

23.3.1.12 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 | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| 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 |
|-----------|--|
| DEVICECFG | DEVICECFG Hardwired to 0x0000_0000. |

23.3.1.13 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 | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| 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 | 1 | |

MTB_DEVICETYPID field descriptions

| Field | Description |
|-------------|--|
| DEVICETYPID | DEVICETYPID Hardwired to 0x0000_0031. |

23.3.1.14 Peripheral ID Register (MTB_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_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* |

MTB_PERIPHID n field descriptions

| Field | Description |
|----------|---|
| PERIPHID | <p>PERIPHID</p> <p>Peripheral ID4 is hardwired to 0x0000_0004; ID0 to 0x0000_0032; ID1 to 0x0000_00B9; ID2 to 0x0000_001B; and all the others to 0x0000_0000.</p> |

23.3.1.15 Component ID Register (MTB_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_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* | 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* | |

MTB_COMPID n field descriptions

| Field | Description |
|--------|---|
| COMPID | <p>Component ID</p> <p>Component ID0 is hardwired to 0x0000_000D; ID1 to 0x0000_0090; ID2 to 0x0000_0005; ID3 to 0x0000_00B1.</p> |

23.3.2 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 | 23.3.2.1/302 |
| F000_1020 | MTB_DWT Comparator Register (MTBDWT_COMP0) | 32 | R/W | 0000_0000h | 23.3.2.2/303 |
| F000_1024 | MTB_DWT Comparator Mask Register (MTBDWT_MASK0) | 32 | R/W | 0000_0000h | 23.3.2.3/303 |
| F000_1028 | MTB_DWT Comparator Function Register 0 (MTBDWT_FCT0) | 32 | R/W | 0000_0000h | 23.3.2.4/304 |
| F000_1030 | MTB_DWT Comparator Register (MTBDWT_COMP1) | 32 | R/W | 0000_0000h | 23.3.2.2/303 |
| F000_1034 | MTB_DWT Comparator Mask Register (MTBDWT_MASK1) | 32 | R/W | 0000_0000h | 23.3.2.3/303 |
| F000_1038 | MTB_DWT Comparator Function Register 1 (MTBDWT_FCT1) | 32 | R/W | 0000_0000h | 23.3.2.5/306 |
| F000_1200 | MTB_DWT Trace Buffer Control Register (MTBDWT_TBCTRL) | 32 | R/W | 2000_0000h | 23.3.2.6/307 |
| F000_1FC8 | Device Configuration Register (MTBDWT_DEVICECFG) | 32 | R | 0000_0000h | 23.3.2.7/309 |
| F000_1FCC | Device Type Identifier Register (MTBDWT_DEVICETYPID) | 32 | R | 0000_0004h | 23.3.2.8/309 |
| F000_1FD0 | Peripheral ID Register (MTBDWT_PERIPHID4) | 32 | R | See section | 23.3.2.9/310 |
| F000_1FD4 | Peripheral ID Register (MTBDWT_PERIPHID5) | 32 | R | See section | 23.3.2.9/310 |
| F000_1FD8 | Peripheral ID Register (MTBDWT_PERIPHID6) | 32 | R | See section | 23.3.2.9/310 |
| F000_1FDC | Peripheral ID Register (MTBDWT_PERIPHID7) | 32 | R | See section | 23.3.2.9/310 |
| F000_1FE0 | Peripheral ID Register (MTBDWT_PERIPHID0) | 32 | R | See section | 23.3.2.9/310 |
| F000_1FE4 | Peripheral ID Register (MTBDWT_PERIPHID1) | 32 | R | See section | 23.3.2.9/310 |
| F000_1FE8 | Peripheral ID Register (MTBDWT_PERIPHID2) | 32 | R | See section | 23.3.2.9/310 |
| F000_1FEC | Peripheral ID Register (MTBDWT_PERIPHID3) | 32 | R | See section | 23.3.2.9/310 |
| F000_1FF0 | Component ID Register (MTBDWT_COMPID0) | 32 | R | See section | 23.3.2.10/310 |
| F000_1FF4 | Component ID Register (MTBDWT_COMPID1) | 32 | R | See section | 23.3.2.10/310 |
| F000_1FF8 | Component ID Register (MTBDWT_COMPID2) | 32 | R | See section | 23.3.2.10/310 |
| F000_1FFC | Component ID Register (MTBDWT_COMPID3) | 32 | R | See section | 23.3.2.10/310 |

23.3.2.1 MTB DWT Control Register (MTBDWT_CTRL)

The MTBDWT_CTRL register provides read-only information on the watchpoint configuration for the MTB_DWT.

Address: F000_1000h base + 0h offset = F000_1000h

| 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 | NUMCMP | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| W | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| Reset | 0 | 0 | 1 | 0 | 1 | 1 | 1 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | | |

MTBDWT_CTRL field descriptions

| Field | Description |
|-----------------|---|
| 31–28 NUMCMP | Number of comparators The MTB_DWT implements two comparators. |
| DWTCFGCTRL | DWT configuration controls This field is hardwired to 0xF00_0000, disabling all the remaining DWT functionality. The specific fields and their state are: MTBDWT_CTRL[27] = NOTRCPKT = 1, trace sample and exception trace is not supported MTBDWT_CTRL[26] = NOEXTTRIG = 1, external match signals are not supported MTBDWT_CTRL[25] = NOCYCCNT = 1, cycle counter is not supported MTBDWT_CTRL[24] = NOPRFCNT = 1, profiling counters are not supported MTBDWT_CTRL[22] = CYCEBTENA = 0, no POSTCNT underflow packets generated MTBDWT_CTRL[21] = FOLDEVTENA = 0, no folded instruction counter overflow events MTBDWT_CTRL[20] = LSUEVTENA = 0, no LSU counter overflow events MTBDWT_CTRL[19] = SLEEPEVTENA = 0, no sleep counter overflow events MTBDWT_CTRL[18] = EXCEVTENA = 0, no exception overhead counter events MTBDWT_CTRL[17] = CPIEVTEVA = 0, no CPI counter overflow events MTBDWT_CTRL[16] = EXCTRCENA = 0, generation of exception trace disabled MTBDWT_CTRL[12] = PCSAMPLENA = 0, no periodic PC sample packets generated MTBDWT_CTRL[11:10] = SYNCTAP = 0, no synchronization packets MTBDWT_CTRL[9] = CYCTAP = 0, cycle counter is not supported MTBDWT_CTRL[8:5] = POSTINIT = 0, cycle counter is not supported MTBDWT_CTRL[4:1] = POSTPRESET = 0, cycle counter is not supported MTBDWT_CTRL[0] = CYCCNTENA = 0, cycle counter is not supported |

23.3.2.2 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

| 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

MTBDWT_COMPn field descriptions

| Field | Description |
|-------|--|
| COMP | Reference value for comparison 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". |

23.3.2.3 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 | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| W | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |

Reset 0

MTBDWT_MASKn field descriptions

| Field | Description |
|------------------|--|
| 31–5 Reserved | This field is reserved. This read-only field is reserved and always has the value 0. |
| MASK | MASK 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. 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. |

Table continues on the next page...

MTBDWT_MASKn field descriptions (continued)

| Field | Description |
|-------|---|
| | If MTBDWT_COMP0 is used as a data value comparator, then MTBDWT_MASK0 should be programmed to zero. |

23.3.2.4 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 | | | | | | | | | | | | | | | | |
| | | | | | | | | | | | | | | | | |
| | | | | | | | | | | | | | | | | |
| 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 | | | | | | | | | | | | | | | | |
| | | | | | | | | | | | | | | | | |
| | | | | | | | | | | | | | | | | |
| | | | | | | | | | | | | | | | | |
| 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 Reserved | This field is reserved. This read-only field is reserved and always has the value 0. |

Table continues on the next page...

MTBDWTFCT0 field descriptions (continued)

| Field | Description |
|---------------------|--|
| 24 MATCHED | <p>Comparator match</p> <p>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.</p> <ul style="list-style-type: none"> 0 No match. 1 Match occurred. |
| 23–20 Reserved | <p>This field is reserved.</p> <p>This read-only field is reserved and always has the value 0.</p> |
| 19–16 Reserved | <p>This field is reserved.</p> <p>This read-only field is reserved and always has the value 0.</p> |
| 15–12 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 MTBDWTFCT0 is used as a data watchpoint and MTBDWTFCT1 as an address watchpoint, DATAVADDR0 must be set.</p> |
| 11–10 DATAVSIZE | <p>Data Value Size</p> <p>For data value matching, this field defines the size of the required data comparison.</p> <ul style="list-style-type: none"> 00 Byte. 01 Halfword. 10 Word. 11 Reserved. Any attempts to use this value results in UNPREDICTABLE behavior. |
| 9 Reserved | <p>This field is reserved.</p> <p>This read-only field is reserved and always has the value 0.</p> |
| 8 DATAVMATCH | <p>Data Value Match</p> <p>When this field is 1, it enables data value comparison. For this implementation, MTBDWTFCT0 supports address or data value comparisons; MTBDWTFCT1 only supports address comparisons.</p> <ul style="list-style-type: none"> 0 Perform address comparison. 1 Perform data value comparison. |
| 7–4 Reserved | <p>This field is reserved.</p> <p>This read-only field is reserved and always has the value 0.</p> |
| FUNCTION | <p>Function</p> <p>Selects the action taken on a comparator match. If MTBDWTFCT0 is used for a data value and MTBDWTFCT1 for an address value, then MTBDWTFCT1[FUNCTION] must be set to zero. For this configuration, MTBDWTFCT1[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"> 0000 Disabled. 0100 Instruction fetch. 0101 Data operand read. 0110 Data operand write. 0111 Data operand (read + write). others Reserved. Any attempts to use this value results in UNPREDICTABLE behavior. |

23.3.2.5 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 | | | | | | 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 Reserved | This field is reserved. This read-only field is reserved and always has the value 0. |
| 24 MATCHED | Comparator match 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. 1 Match occurred. |
| 23–4 Reserved | This field is reserved. This read-only field is reserved and always has the value 0. |
| 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"> 0000 Disabled. 0100 Instruction fetch. 0101 Data operand read. 0110 Data operand write. 0111 Data operand (read + write). others Reserved. Any attempts to use this value results in UNPREDICTABLE behavior. |

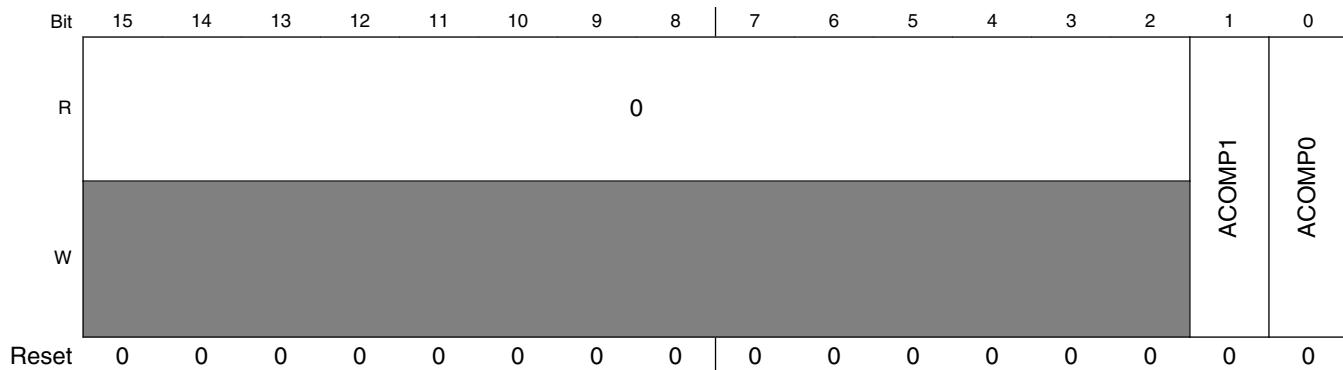
23.3.2.6 MTB_DWT Trace Buffer Control Register (MTBDWT_TBCTRL)

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 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 Reserved | <p>This field is reserved.</p> <p>This read-only field is reserved and always has the value 0.</p> |
| 1 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"> 0 Trigger TSTOP based on the assertion of MTBDWWT_FCT1[MATCHED]. 1 Trigger TSTART based on the assertion of MTBDWWT_FCT1[MATCHED]. |
| 0 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"> • Address match in MTBDWWT_COMP0 when MTBDWWT_FCT0[DATAVMATCH] = 0 • Data match in MTBDWWT_COMP0 when MTBDWWT_FCT0[DATAVMATCH, DATAVADDR0] = {1,0} • Data match in MTBDWWT_COMP0 and address match in MTBDWWT_COMP1 when MTBDWWT_FCT0[DATAVMATCH, DATAVADDR0] = {1,1} <ul style="list-style-type: none"> 0 Trigger TSTOP based on the assertion of MTBDWWT_FCT0[MATCHED]. 1 Trigger TSTART based on the assertion of MTBDWWT_FCT0[MATCHED]. |

23.3.2.7 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 |
|-----------|--|
| DEVICECFG | DEVICECFG Hardwired to 0x0000_0000. |

23.3.2.8 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 |
|-------------|--|
| DEVICETYPID | DEVICETYPID Hardwired to 0x0000_0004. |

23.3.2.9 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* | 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* | |

MTBDWT_PERIPHID n field descriptions

| Field | Description |
|----------|---|
| PERIPHID | PERIPHID Peripheral ID1 is hardwired to 0x0000_00E0; ID2 to 0x0000_0008; and all the others to 0x0000_0000. |

23.3.2.10 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* | 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* | | |

MTBDWT_COMPID n field descriptions

| Field | Description |
|--------|--|
| COMPID | Component ID Component ID0 is hardwired to 0x0000_000D; ID1 to 0x0000_0090; ID2 to 0x0000_0005; ID3 to 0x0000_00B1. |

23.3.3 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.

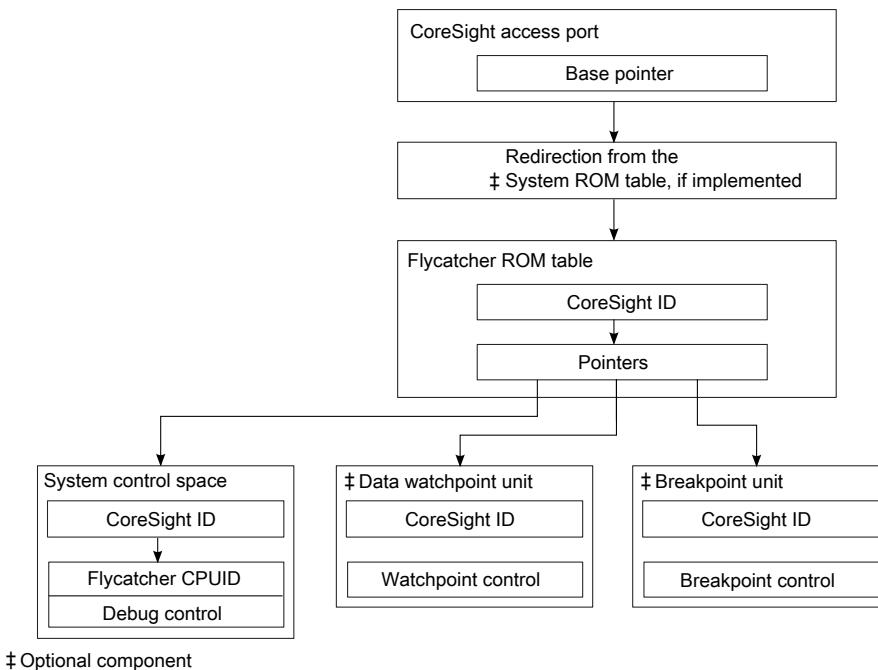


Figure 23-3. CoreSight discovery process

ROM memory map

| Absolute address (hex) | Register name | Width (in bits) | Access | Reset value | Section/ page |
|------------------------|--|-----------------|--------|-----------------------------|------------------|
| F000_2000 | Entry (ROM_ENTRY0) | 32 | R | See section | 23.3.3.1/ 312 |
| F000_2004 | Entry (ROM_ENTRY1) | 32 | R | See section | 23.3.3.1/ 312 |
| F000_2008 | Entry (ROM_ENTRY2) | 32 | R | See section | 23.3.3.1/ 312 |
| F000_200C | End of Table Marker Register (ROM_TABLEMARK) | 32 | R | 0000_0000h | 23.3.3.2/ 313 |
| F000_2FCC | System Access Register (ROM_SYSACCESS) | 32 | R | 0000_0001h | 23.3.3.3/ 313 |
| F000_2FD0 | Peripheral ID Register (ROM_PERIPHID4) | 32 | R | See section | 23.3.3.4/ 314 |

Table continues on the next page...

ROM memory map (continued)

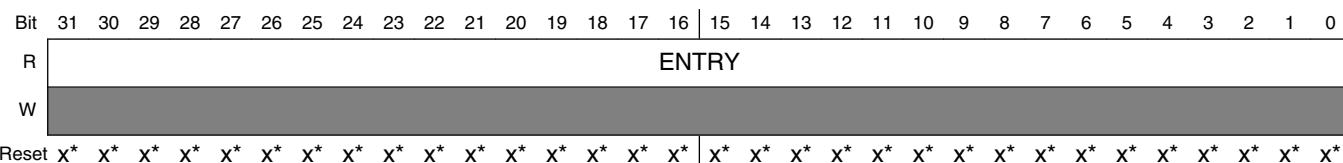
| Absolute address (hex) | Register name | Width (in bits) | Access | Reset value | Section/page |
|------------------------|--|-----------------|--------|-----------------------------|------------------|
| F000_2FD4 | Peripheral ID Register (ROM_PERIPHID5) | 32 | R | See section | 23.3.3.4/ 314 |
| F000_2FD8 | Peripheral ID Register (ROM_PERIPHID6) | 32 | R | See section | 23.3.3.4/ 314 |
| F000_2FDC | Peripheral ID Register (ROM_PERIPHID7) | 32 | R | See section | 23.3.3.4/ 314 |
| F000_2FE0 | Peripheral ID Register (ROM_PERIPHID0) | 32 | R | See section | 23.3.3.4/ 314 |
| F000_2FE4 | Peripheral ID Register (ROM_PERIPHID1) | 32 | R | See section | 23.3.3.4/ 314 |
| F000_2FE8 | Peripheral ID Register (ROM_PERIPHID2) | 32 | R | See section | 23.3.3.4/ 314 |
| F000_2FEC | Peripheral ID Register (ROM_PERIPHID3) | 32 | R | See section | 23.3.3.4/ 314 |
| F000_2FF0 | Component ID Register (ROM_COMPID0) | 32 | R | See section | 23.3.3.5/ 314 |
| F000_2FF4 | Component ID Register (ROM_COMPID1) | 32 | R | See section | 23.3.3.5/ 314 |
| F000_2FF8 | Component ID Register (ROM_COMPID2) | 32 | R | See section | 23.3.3.5/ 314 |
| F000_2FFC | Component ID Register (ROM_COMPID3) | 32 | R | See section | 23.3.3.5/ 314 |

23.3.3.1 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.

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

**ROM_ENTRYn field descriptions**

| Field | Description |
|-------|-------------|
| ENTRY | ENTRY |

ROM_ENTRYn field descriptions (continued)

| Field | Description |
|-------|--|
| | Entry 0 (MTB) is hardwired to 0xFFFF_E003; Entry 1 (MTBDWT) to 0xFFFF_F003; Entry 2 (CM0+ ROM Table) to 0xF00F_D003. |

23.3.3.2 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 | 0 | 0 | |

ROM_TABLEMARK field descriptions

| Field | Description |
|-------|----------------------------------|
| MARK | MARK Hardwired to 0x0000_0000 |

23.3.3.3 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 | 0 | 0 | 1 | |

ROM_SYSACCESS field descriptions

| Field | Description |
|-----------|---------------------------------------|
| SYSACCESS | SYSACCESS Hardwired to 0x0000_0001 |

23.3.3.4 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* | 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* | |

ROM_PERIPHID n field descriptions

| Field | Description |
|----------|---|
| PERIPHID | PERIPHID Peripheral ID1 is hardwired to 0x0000_00E0; ID2 to 0x0000_0008; and all the others to 0x0000_0000. |

23.3.3.5 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* | 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* | | |

ROM_COMPID n field descriptions

| Field | Description |
|--------|---|
| COMPID | Component ID Component ID0 is hardwired to 0x0000_000D; ID1 to 0x0000_0010; ID2 to 0x0000_0005; ID3 to 0x0000_00B1. |

Chapter 24

Multipurpose Clock Generator Lite (MCG_Lite)

24.1 Introduction

The Multipurpose Clock Generator Lite (MCG_Lite) module provides several clock source options for the MCU. This module contains one 48 MHz and one 8/2 MHz Internal Reference Clock (IRC) sources. The module selects one of IRCs or External Oscillator/Clock (EXT) as the MCU clock sources.

24.1.1 Features

The MCG_Lite module has the following features:

- High-frequency Internal Reference Clock (HIRC)
 - 48 MHz clock source
 - Support various trims to achieve target accuracy
- Low-frequency Internal Reference Clock (LIRC)
 - Selectable 8 MHz or 2 MHz clock source
 - Trim bit to ensure $\pm 3\%$ accuracy across PVT
- Glitchless clock switcher for internal clock sources (HIRC and LIRC) and external clock source (EXT)
- Dedicated high frequency clock sources output (MCGPCLK) for peripheral use
- Dedicated low frequency clock sources output (LIRC_CLK) for peripheral use
- Divider/prescaler (FCRDIV) for low frequency IRC to support /1, /2, /4, /8, /16, /32, /64, and /128 division factors
- Second divider (LIRC_DIV2) to support /1, /2, /4, /8, /16, /32, /64, and /128 division factors
- Control signal for the external clock/oscillator (EXT)
 - EREFS0
 - External clock from crystal oscillator can be used as system clock sources

24.1.2 Block diagram

The block diagram of MCG_Lite is as follows.

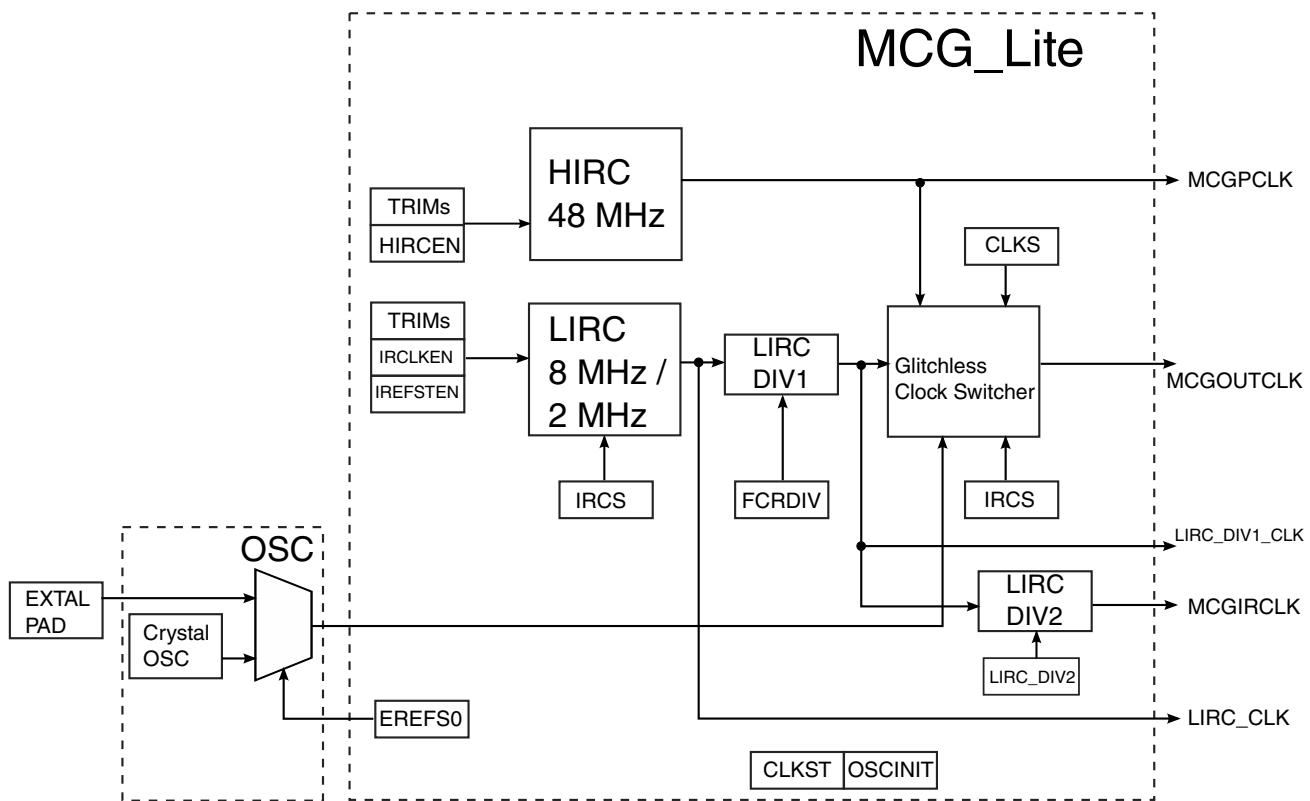


Figure 24-1. MCG_Lite block diagram

24.2 Memory map and register definition

The MCG_Lite module contains several fields for selecting the clock source and the dividers for various module clocks.

NOTE

The MCG_Lite registers can be written only in supervisor mode. Write accesses in user mode are blocked and will result in a bus error.

MCG memory map

| Absolute address (hex) | Register name | Width (in bits) | Access | Reset value | Section/ page |
|------------------------|---------------------------------|-----------------|--------|-------------|----------------------------|
| 4006_4000 | MCG Control Register 1 (MCG_C1) | 8 | R/W | 40h | 24.2.1/317 |

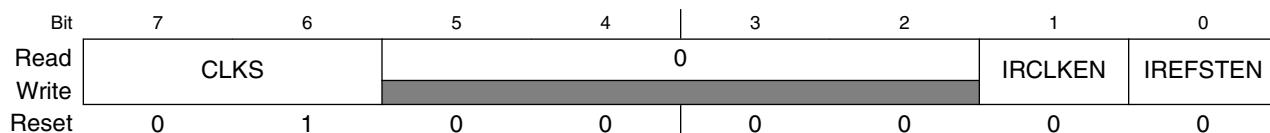
Table continues on the next page...

MCG memory map (continued)

| Absolute address (hex) | Register name | Width (in bits) | Access | Reset value | Section/ page |
|------------------------|---|-----------------|--------|-------------|----------------------------|
| 4006_4001 | MCG Control Register 2 (MCG_C2) | 8 | R/W | 01h | 24.2.2/318 |
| 4006_4006 | MCG Status Register (MCG_S) | 8 | R | 04h | 24.2.3/318 |
| 4006_4008 | MCG Status and Control Register (MCG_SC) | 8 | R/W | 00h | 24.2.4/319 |
| 4006_4018 | MCG Miscellaneous Control Register (MCG_MC) | 8 | R/W | 00h | 24.2.5/320 |

24.2.1 MCG Control Register 1 (MCG_C1)

Address: 4006_4000h base + 0h offset = 4006_4000h

**MCG_C1 field descriptions**

| Field | Description |
|-----------------|---|
| 7–6 CLKS | Clock Source Select Selects the clock source for MCGOUTCLK. 00 Selects HIRC clock as the main clock source. This is HIRC mode. 01 Selects LIRC clock as the main clock source. This is LIRC2M or LIRC8M mode. 10 Selects external clock as the main clock source. This is EXT mode. 11 Reserved. Writing 11 takes no effect. |
| 5–2 Reserved | This field is reserved. This read-only field is reserved and always has the value 0. |
| 1 IRCLKEN | Internal Reference Clock Enable Enables the IRC source. 0 LIRC is disabled. 1 LIRC is enabled. |
| 0 IREFSTEN | Internal Reference Stop Enable Controls whether the IRC source remains enabled when the MCG_Lite enters Stop mode. 0 LIRC is disabled in Stop mode. 1 LIRC is enabled in Stop mode, if IRCLKEN is set. |

24.2.2 MCG Control Register 2 (MCG_C2)

Address: 4006_4000h base + 1h offset = 4006_4001h

| Bit | 7 | 6 | 5 | 4 | | 3 | 2 | 1 | 0 |
|-------|---|---|---|---|--|---|--------|---|---|
| Read | | | | 0 | | | EREFS0 | 0 | |
| Write | | | | | | | | | |
| Reset | 0 | 0 | 0 | 0 | | 0 | 0 | 0 | 1 |

MCG_C2 field descriptions

| Field | Description |
|-----------------|---|
| 7–3 Reserved | This field is reserved. This read-only field is reserved and always has the value 0. |
| 2 EREFS0 | External Clock Source Select Selects the source for the external reference clock. See the Oscillator (OSC) chapter for more details. 0 External clock requested. 1 Oscillator requested. |
| 1 Reserved | This field is reserved. This read-only field is reserved and always has the value 0. |
| 0 IRCS | Low-frequency Internal Reference Clock Select Controls the LIRC to work at 2 MHz or 8 MHz mode. 0 LIRC is in 2 MHz mode. 1 LIRC is in 8 MHz mode. |

24.2.3 MCG Status Register (MCG_S)

Address: 4006_4000h base + 6h offset = 4006_4006h

| Bit | 7 | 6 | 5 | 4 | | 3 | 2 | 1 | 0 |
|-------|---|---|---|---|--|-------|---|----------|---|
| Read | | | | 0 | | CLKST | | OSCINIT0 | 0 |
| Write | | | | | | | | | |
| Reset | 0 | 0 | 0 | 0 | | 0 | 1 | 0 | 0 |

MCG_S field descriptions

| Field | Description |
|-----------------|--|
| 7–4 Reserved | This field is reserved. This read-only field is reserved and always has the value 0. |
| 3–2 CLKST | Clock Mode Status Indicates the current clock mode. This field does not update immediately after a write to MCG_C1[CLKS] due to internal synchronization between clock domains. |

Table continues on the next page...

MCG_S field descriptions (continued)

| Field | Description |
|---------------|---|
| | 00 HIRC clock is selected as the main clock source, and MCG_Lite works at HIRC mode. 01 LIRC clock is selected as the main clock source, and MCG_Lite works at LIRC2M or LIRC8M mode. 10 External clock is selected as the main clock source, and MCG_Lite works at EXT mode. 11 Reserved. |
| 1 OSCINIT0 | OSC Initialization Status This flag, which resets to 0, is set to 1 after the initialization cycles of the crystal oscillator clock are completed. After being set, the bit is cleared to 0 if the OSC is subsequently disabled. See the Oscillator (OSC) chapter for more information. 0 OSC is not ready. 1 OSC clock is ready. |
| 0 Reserved | This field is reserved. This read-only field is reserved and always has the value 0. |

24.2.4 MCG Status and Control Register (MCG_SC)

Address: 4006_4000h base + 8h offset = 4006_4008h

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|-------|---|---|---|---|---|--------|---|---|
| Read | | | 0 | | | | | 0 |
| Write | | | | | | FCRDIV | | |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

MCG_SC field descriptions

| Field | Description |
|-----------------|---|
| 7–4 Reserved | This field is reserved. This read-only field is reserved and always has the value 0. |
| 3–1 FCRDIV | Low-frequency Internal Reference Clock Divider Selects the factor value to divide the LIRC source. 000 Division factor is 1. 001 Division factor is 2. 010 Division factor is 4. 011 Division factor is 8. 100 Division factor is 16. 101 Division factor is 32. 110 Division factor is 64. 111 Division factor is 128. |
| 0 Reserved | This field is reserved. This read-only field is reserved and always has the value 0. |

24.2.5 MCG Miscellaneous Control Register (MCG_MC)

Address: 4006_4000h base + 18h offset = 4006_4018h

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|-------|--------|---|---|---|---|-----------|---|---|
| Read | HIRCEN | | 0 | | | LIRC_DIV2 | | |
| Write | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

MCG_MC field descriptions

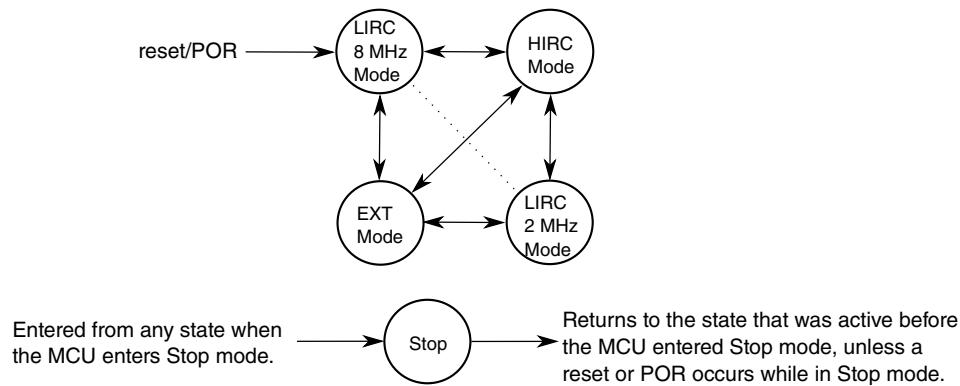
| Field | Description |
|-----------------|---|
| 7 HIRCEN | High-frequency IRC Enable Enables the HIRC, even when MCG_Lite is not working at HIRC mode. 0 HIRC source is not enabled. 1 HIRC source is enabled. |
| 6–3 Reserved | This field is reserved. This read-only field is reserved and always has the value 0. |
| LIRC_DIV2 | Second Low-frequency Internal Reference Clock Divider Selects the factor value to further divide the LIRC source. 000 Division factor is 1. 001 Division factor is 2. 010 Division factor is 4. 011 Division factor is 8. 100 Division factor is 16. 101 Division factor is 32. 110 Division factor is 64. 111 Division factor is 128. |

24.3 Functional description

This section presents the functional details of the MCG_Lite module.

24.3.1 Clock mode switching

Different states of the MCG_Lite module are shown in the following figure. The arrows indicate the permitted MCG_Lite mode transitions.

**Figure 24-2. MCG_Lite mode state diagram**

The MCG_Lite module does not support switch between LIRC 2 MHz and 8 MHz directly, because 2 MHz and 8 MHz clock generators share circuits and logics. To switch between each other, the module must be in HIRC or EXT clock mode.

If entering VLPR mode, MCG_Lite has to be configured and enter LIRC2M, LIRC8M or EXT mode, and MCG_MC[HIRCEN] must also be cleared. 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.

When power on or out of reset, LIRC is selected as the main clock source. To select other clock sources, the user must perform the following steps.

To enter HIRC mode:

1. Write 1b to MCG_MC[HIRCEN] to enable HIRC (optional).
2. Write 00b to MCG_C1[CLKS] to select HIRC clock source.
3. Check MCG_S[CLKST] to confirm HIRC clock source is selected.

To enter EXT mode:

1. Configure MCG_C2[EREFS0] for external clock source selection.
2. Write 10b to MCG_C1[CLKS] to select external clock source.
3. Check MCG_S[CLKST] to confirm external clock source is selected.

To enter LIRC2M mode from HIRC or EXT mode:

1. Write 0b to MCG_C2[IRCS] to select LIRC 2M.
2. Write 1b to MCG_C1[IRCLKEN] to enable LIRC clock (optional).
3. Write 01b to MCG_C1[CLKS] to select LIRC clock source.
4. Check MCG_S[CLKST] to confirm LIRC clock source is selected.

To enter LIRC8M mode from HIRC or EXT mode:

1. Write 1b to MCG_C2[IRCS] to select LIRC 8M.
2. Write 1b to MCG_C1[IRCLKEN] to enable LIRC clock (optional).
3. Write 01b to MCG_C1[CLKS] to select LIRC clock source.
4. Check MCG_S[CLKST] to confirm LIRC clock source is selected.

24.3.2 LIRC divider 1

In the MCG_Lite module, there is a divider for LIRC clock. The divider supports /1, /2, /4, /8, /16, /32, /64, and /128 division factors. For details, see the register field description of [MCG_SC\[FCRDIV\]](#). The divided clock of LIRC DIV1 is one of the inputs of clock select switch. It is the input for the 2nd LIRC DIV as well. See the Chip Configuration information for more details.

24.3.3 LIRC divider 2

In the MCG_Lite module, there is another divider to further divide the LIRC clock, named LIRC DIV2. This divider supports /1, /2, /4, /8, /16, /32, /64, and /128 division factors. For details, see the register field description of [MCG_MC\[LIRC_DIV2\]](#). The divided clock of LIRC DIV2 is MCGIRCLK, and it can be used as peripheral clock. See the Chip Configuration information for more details.

24.3.4 Enable LIRC in Stop mode

In Stop mode, HIRC is disabled to save power. For LIRC, by default it is disabled as well. To enable LIRC in Stop mode, write 1b to MCG_C1[IREFSTEN] and MCG_C1[IRCLKEN] before entering Stop mode.

24.3.5 MCG-Lite in Low-power mode

In Stop/VLPS mode, MCG-Lite is inactive, HIRC is disabled, and LIRC is disabled except that both MCG_C1[IREFSTEN] and MCG_C1[IRCLKEN] are set before entering the Stop/VLPS mode.

In VLLS mode, MCG-Lite is power down.

In VLPR/VLPW mode, MCG-Lite is in Low-power mode, HIRC is disabled, while LIRC can keep working.

Chapter 25

Oscillator (OSC)

25.1 OSC modes of operation with MCG_Lite and RTC

The most common method of controlling the OSC block is through MCG_C1[CLKS] and the fields of MCG_C2 register to configure for crystal or external clock operation. OSC_CR also provides control for enabling the OSC module and configuring internal load capacitors for the EXTAL and XTAL pins. See the [OSC](#) and [MCG_Lite](#) chapters for more details.

RTC_CR[OSCE] has overriding control over the MCG_Lite and OSC_CR enable functions. When RTC_CR[OSCE] is set, the OSC is configured for low frequency, low power and RTC_CR[SCxP] override OSC_CR[SCxP] to control the internal capacitance configuration. See the RTC chapter for more details.

25.2 Introduction

The OSC module is a crystal oscillator. The module, in conjunction with an external crystal or resonator, generates a reference clock for the MCU.

25.3 Features and Modes

Key features of the module are listed here.

- 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

Block Diagram

- 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.

25.4 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 details for the external reference clock source in this MCU.

The figure found here shows the block diagram of the OSC module.

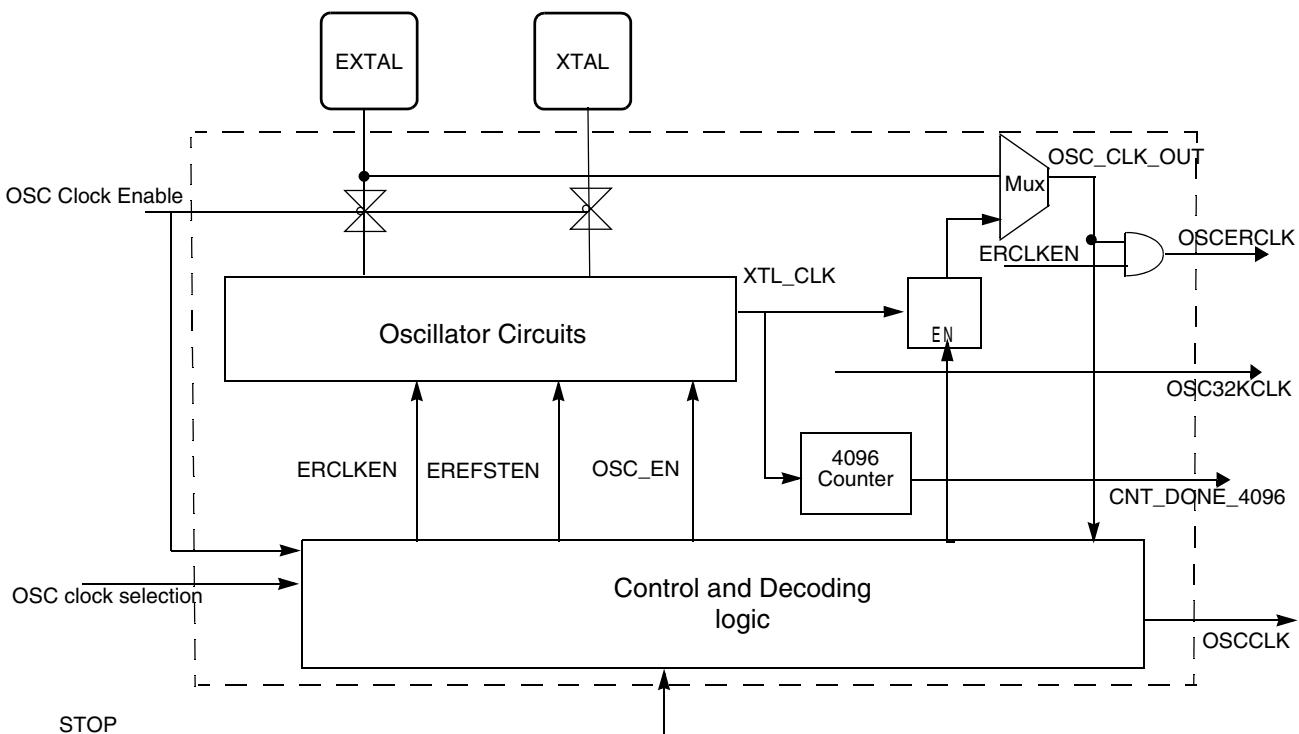


Figure 25-1. OSC Module Block Diagram

25.5 OSC Signal Descriptions

The table found here 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 |

25.6 External Crystal / Resonator Connections

The connections for a crystal/resonator frequency reference are shown in the figures found here.

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 |

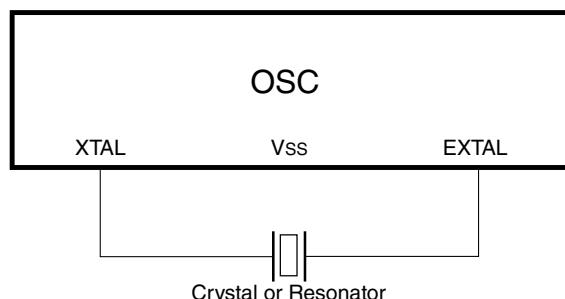
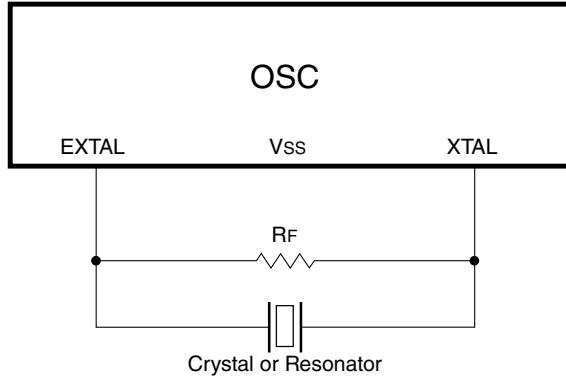
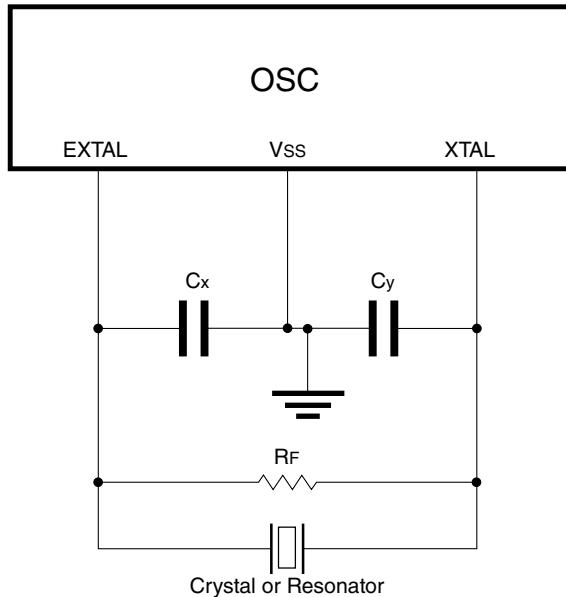


Figure 25-2. Crystal/Ceramic Resonator Connections - Connection 1

**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**

25.7 External Clock Connections

In external clock mode, the pins can be connected as shown in the figure found here.

NOTE

XTAL can be used as a GPIO when the GPIO alternate function is configured for it.

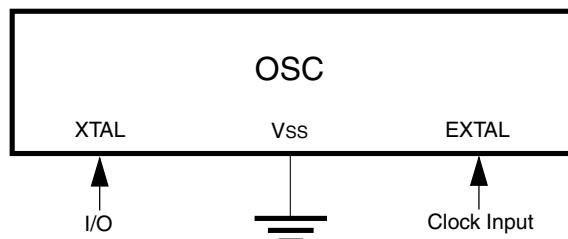


Figure 25-5. External Clock Connections

25.8 Memory Map/Register Definitions

Some oscillator module register bits are typically incorporated into other peripherals such as MCG or SIM.

25.8.1 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 (OSC_CR) | 8 | R/W | 00h | 25.8.1.1/ 327 |

25.8.1.1 OSC Control Register (OSC_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 | EREFTEN | 0 | SC2P | SC4P | SC8P | SC16P |
| Write | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

OSC_CR field descriptions

| Field | Description |
|---------------|--|
| 7 ERCLKEN | <p>External Reference Enable</p> <p>Enables external reference clock (OSCERCLK).</p> <p>0 External reference clock is inactive. 1 External reference clock is enabled.</p> |
| 6 Reserved | <p>This field is reserved.</p> <p>This read-only field is reserved and always has the value 0.</p> |
| 5 EREFSSEN | <p>External Reference Stop Enable</p> <p>Controls whether or not the external reference clock (OSCERCLK) remains enabled when MCU enters Stop mode.</p> <p>0 External reference clock is disabled in Stop mode. 1 External reference clock stays enabled in Stop mode if ERCLKEN is set before entering Stop mode.</p> |
| 4 Reserved | <p>This field is reserved.</p> <p>This read-only field is reserved and always has the value 0.</p> |
| 3 SC2P | <p>Oscillator 2 pF Capacitor Load Configure</p> <p>Configures the oscillator load.</p> <p>0 Disable the selection. 1 Add 2 pF capacitor to the oscillator load.</p> |
| 2 SC4P | <p>Oscillator 4 pF Capacitor Load Configure</p> <p>Configures the oscillator load.</p> <p>0 Disable the selection. 1 Add 4 pF capacitor to the oscillator load.</p> |
| 1 SC8P | <p>Oscillator 8 pF Capacitor Load Configure</p> <p>Configures the oscillator load.</p> <p>0 Disable the selection. 1 Add 8 pF capacitor to the oscillator load.</p> |
| 0 SC16P | <p>Oscillator 16 pF Capacitor Load Configure</p> <p>Configures the oscillator load.</p> <p>0 Disable the selection. 1 Add 16 pF capacitor to the oscillator load.</p> |

25.9 Functional Description

Functional details of the module can be found here.

25.9.1 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.

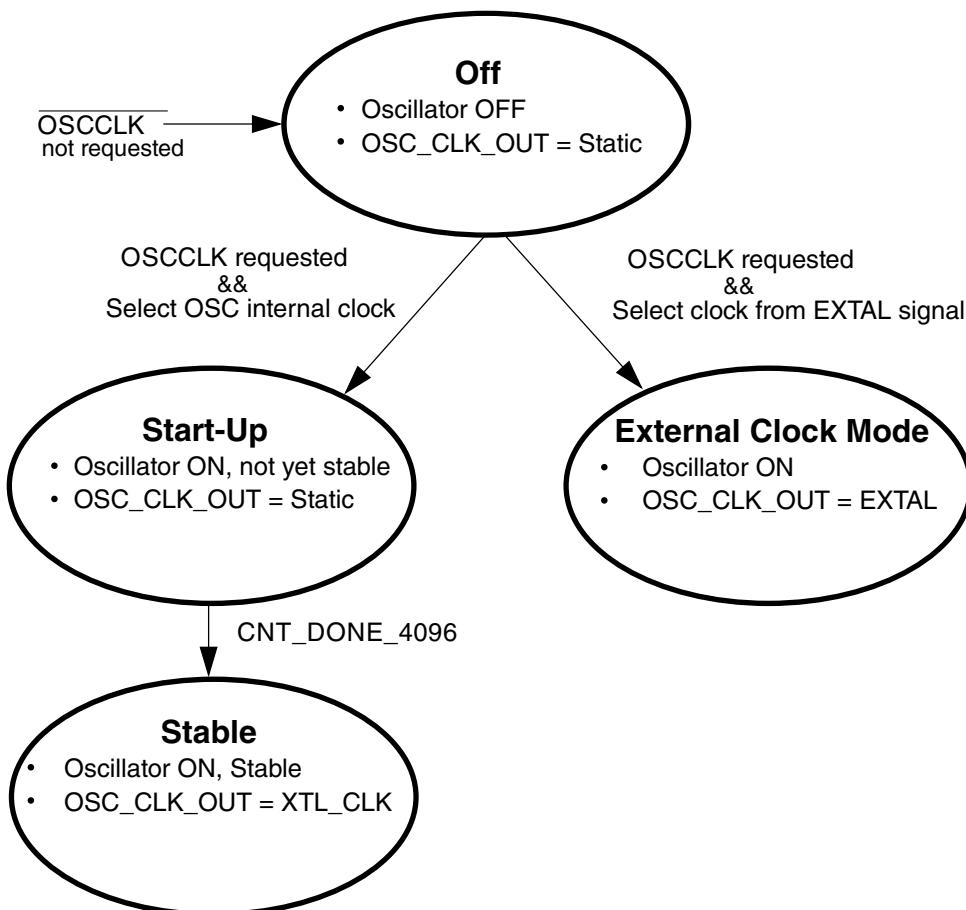


Figure 25-6. OSC Module state diagram

NOTE

XTL_CLK is the clock generated internally from OSC circuits.

25.9.1.1 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 details. 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.

25.9.1.2 Oscillator startup

The OSC enters startup 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.

25.9.1.3 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.

25.9.1.4 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, see the chip configuration details. 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.

25.9.2 OSC module modes

The OSC is a Pierce-type oscillator that supports external crystals or resonators operating over the frequency ranges shown in [Table 25-3](#). 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 25-3. Oscillator modes

| Mode | Frequency Range |
|--------------------------------|--|
| Low-frequency, low-power (VLP) | f_{osc_lo} (1 kHz) up to f_{osc_lo} (32.768 kHz) |

NOTE

For information about low power modes of operation used in this chip and their alignment with some OSC modes, see the chip's Power Management details.

25.9.2.1 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.

25.9.3 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 timeout is used to guarantee output clock stability.

25.9.4 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.

25.10 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.

25.11 Low power modes operation

When the MCU enters Stop modes, the OSC is functional depending on CR[ERCLKEN] and CR[EREFSETN] bit settings. If both these bits are set, the OSC is in operation.

After waking up from Very Low Leakage Stop (VLLSx) modes, all OSC register bits are reset and initialization is required through software.

25.12 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. A list of features provided by the FMC can be found here.

- an interface between bus masters and the 32-bit program flash memory.
- a buffer 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 a mechanism 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.

26.1.2 Features

The features of FMC module 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

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 buffer to provide single-cycle system clock data access times. Whenever a hit occurs for the prefetch speculation buffer, the requested data is transferred within a single system clock.

Upon system reset, the FMC is configured as follows:

- Instruction speculation is enabled.
- Data speculation is disabled.

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

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 KB
- 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.

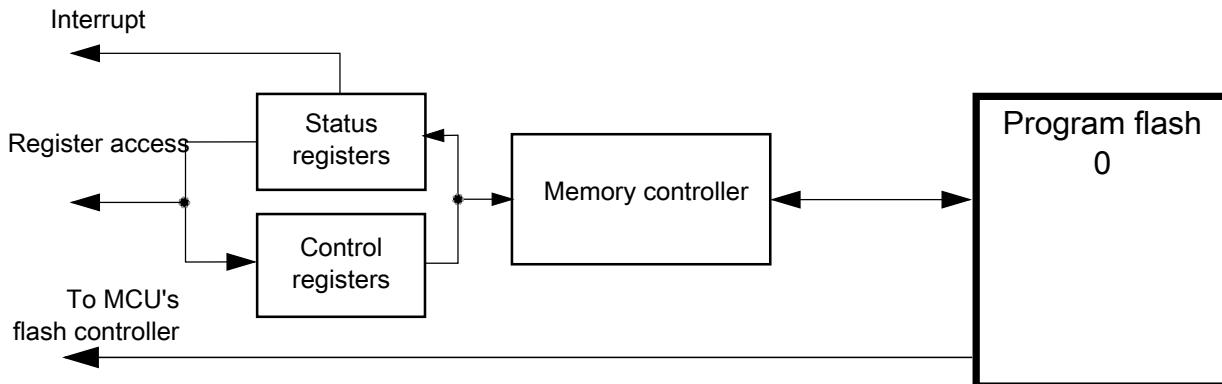


Figure 27-1. Flash Block Diagram

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.

Longword — 32 bits of data with an aligned longword having byte-address[1:0] = 00.

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.

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.

External Signal Description

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.

Word — 16 bits of data with an aligned word having byte-address[0] = 0.

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 Verify Backdoor Access Key Command and Unsecuring the Chip Using Backdoor Key Access . |
| 0x0_0408–0x0_040B | 4 | Program flash protection bytes. Refer to the description of the Program Flash Protection Registers (FPROT0-3). |

Table continues on the next page...

| Flash Configuration Field Byte Address | Size (Bytes) | Field Description |
|--|--------------|--|
| 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 table found here 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 | 27.3.3.1/ 343 |
| 4002_0001 | Flash Configuration Register (FTFA_FCNFG) | 8 | R/W | 00h | 27.3.3.2/ 344 |
| 4002_0002 | Flash Security Register (FTFA_FSEC) | 8 | R | Undefined | 27.3.3.3/ 346 |
| 4002_0003 | Flash Option Register (FTFA_FOPT) | 8 | R | Undefined | 27.3.3.4/ 347 |
| 4002_0004 | Flash Common Command Object Registers (FTFA_FCCOB3) | 8 | R/W | 00h | 27.3.3.5/ 348 |
| 4002_0005 | Flash Common Command Object Registers (FTFA_FCCOB2) | 8 | R/W | 00h | 27.3.3.5/ 348 |
| 4002_0006 | Flash Common Command Object Registers (FTFA_FCCOB1) | 8 | R/W | 00h | 27.3.3.5/ 348 |
| 4002_0007 | Flash Common Command Object Registers (FTFA_FCCOB0) | 8 | R/W | 00h | 27.3.3.5/ 348 |
| 4002_0008 | Flash Common Command Object Registers (FTFA_FCCOB7) | 8 | R/W | 00h | 27.3.3.5/ 348 |
| 4002_0009 | Flash Common Command Object Registers (FTFA_FCCOB6) | 8 | R/W | 00h | 27.3.3.5/ 348 |
| 4002_000A | Flash Common Command Object Registers (FTFA_FCCOB5) | 8 | R/W | 00h | 27.3.3.5/ 348 |
| 4002_000B | Flash Common Command Object Registers (FTFA_FCCOB4) | 8 | R/W | 00h | 27.3.3.5/ 348 |
| 4002_000C | Flash Common Command Object Registers (FTFA_FCCOB3) | 8 | R/W | 00h | 27.3.3.5/ 348 |
| 4002_000D | Flash Common Command Object Registers (FTFA_FCCOB2) | 8 | R/W | 00h | 27.3.3.5/ 348 |
| 4002_000E | Flash Common Command Object Registers (FTFA_FCCOB9) | 8 | R/W | 00h | 27.3.3.5/ 348 |
| 4002_000F | Flash Common Command Object Registers (FTFA_FCCOB8) | 8 | R/W | 00h | 27.3.3.5/ 348 |

Table continues on the next page...

FTFA memory map (continued)

| Absolute address (hex) | Register name | Width (in bits) | Access | Reset value | Section/page |
|------------------------|--|-----------------|--------|-------------|------------------------------|
| 4002_0010 | Program Flash Protection Registers (FTFA_FPROT3) | 8 | R/W | Undefined | 27.3.3.6/349 |
| 4002_0011 | Program Flash Protection Registers (FTFA_FPROT2) | 8 | R/W | Undefined | 27.3.3.6/349 |
| 4002_0012 | Program Flash Protection Registers (FTFA_FPROT1) | 8 | R/W | Undefined | 27.3.3.6/349 |
| 4002_0013 | Program Flash Protection Registers (FTFA_FPROT0) | 8 | R/W | Undefined | 27.3.3.6/349 |

27.3.3.1 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 CCIF | <p>Command Complete Interrupt Flag</p> <p>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>CCIF 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> <p>0 Flash command in progress 1 Flash command has completed</p> |

Table continues on the next page...

FTFA_FSTAT field descriptions (continued)

| Field | Description |
|-----------------|--|
| 6 RDCOLERR | <p>Flash Read Collision Error Flag</p> <p>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> <p>0 No collision error detected 1 Collision error detected</p> |
| 5 ACCERR | <p>Flash Access Error Flag</p> <p>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> <p>0 No access error detected 1 Access error detected</p> |
| 4 FPVIOL | <p>Flash Protection Violation Flag</p> <p>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> <p>0 No protection violation detected 1 Protection violation detected</p> |
| 3–1 Reserved | <p>This field is reserved.</p> <p>This read-only field is reserved and always has the value 0.</p> |
| 0 MGSTAT0 | <p>Memory Controller Command Completion Status Flag</p> <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 field 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> |

27.3.3.2 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 CCIE | <p>Command Complete Interrupt Enable</p> <p>Controls interrupt generation when a flash command completes.</p> <p>0 Command complete interrupt disabled</p> <p>1 Command complete interrupt enabled. An interrupt request is generated whenever the FSTAT[CCIF] flag is set.</p> |
| 6 RDCOLLIE | <p>Read Collision Error Interrupt Enable</p> <p>Controls interrupt generation when a flash memory read collision error occurs.</p> <p>0 Read collision error interrupt disabled</p> <p>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]).</p> |
| 5 ERSAREQ | <p>Erase All Request</p> <p>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>ERSAREQ 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> <p>0 No request or request complete</p> <p>1 Request to:</p> <ul style="list-style-type: none"> 1. run the Erase All Blocks command, 2. verify the erased state, 3. program the security byte in the Flash Configuration Field to the unsecure state, and 4. release MCU security by setting the FSEC[SEC] field to the unsecure state. |
| 4 ERSSUSP | <p>Erase Suspend</p> <p>Allows the user to suspend (interrupt) the Erase Flash Sector command while it is executing.</p> <p>0 No suspend requested</p> <p>1 Suspend the current Erase Flash Sector command execution.</p> |
| 3 Reserved | <p>This field is reserved.</p> <p>This read-only field is reserved and always has the value 0.</p> |
| 2 Reserved | <p>This field is reserved.</p> <p>This read-only field is reserved and always has the value 0.</p> |
| 1 Reserved | <p>This field is reserved.</p> <p>This read-only field is reserved and always has the value 0.</p> |
| 0 Reserved | <p>This field is reserved.</p> <p>This read-only field is reserved and always has the value 0.</p> |

27.3.3.3 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 KEYEN | Backdoor Key Security Enable Enables or disables backdoor key access to the flash memory module. 00 Backdoor key access disabled 01 Backdoor key access disabled (preferred KEYEN state to disable backdoor key access) 10 Backdoor key access enabled 11 Backdoor key access disabled |
| 5–4 MEEN | Mass Erase Enable Bits Enables and disables mass erase capability of the flash memory module. The state of this field is relevant only when SEC is set to secure outside of NVM Normal Mode. When SEC is set to unsecure, the MEEN setting does not matter. 00 Mass erase is enabled 01 Mass erase is enabled 10 Mass erase is disabled 11 Mass erase is enabled |
| 3–2 FSLACC | Freescale Failure Analysis Access Code Enables or disables 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. 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 SEC is set to secure. When SEC is set to unsecure, the FSLACC setting does not matter. |

Table continues on the next page...

FTFA_FSEC field descriptions (continued)

| Field | Description |
|-------|--|
| | 00 Freescale factory access granted 01 Freescale factory access denied 10 Freescale factory access denied 11 Freescale factory access granted |
| SEC | Flash Security Defines 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, SEC is forced to 10b. 00 MCU security status is secure. 01 MCU security status is secure. 10 MCU security status is unsecure. (The standard shipping condition of the flash memory module is unsecure.) 11 MCU security status is secure. |

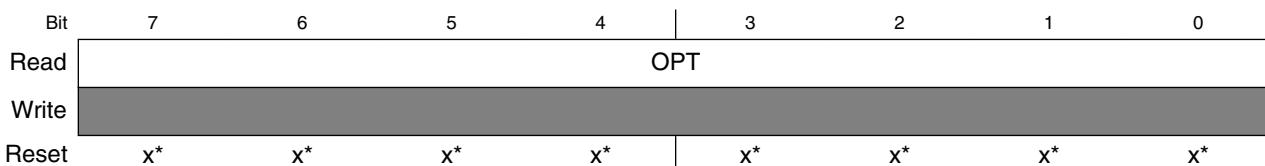
27.3.3.4 Flash Option Register (FTFA_FOPT)

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 |
|-------|--|
| OPT | Nonvolatile Option 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. |

27.3.3.5 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 | | | | | | | | | | | | | | | | | | | | | | | | | | |
|--------------|---|--------------|--|---|--|---|-----------------------|---|----------------------|---|---------------------|---|-------------|---|-------------|---|-------------|---|-------------|---|-------------|---|-------------|---|-------------|---|-------------|
| 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>NOTE: 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> <tr> <td>A</td> <td>Data Byte 6</td> </tr> <tr> <td>B</td> <td>Data Byte 7</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 | A | Data Byte 6 | B | Data Byte 7 |
| 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 | | | | | | | | | | | | | | | | | | | | | | | | | | |
| A | Data Byte 6 | | | | | | | | | | | | | | | | | | | | | | | | | | |
| B | Data Byte 7 | | | | | | | | | | | | | | | | | | | | | | | | | | |

FTFA_FCCOBn field descriptions (continued)

| Field | Description |
|-------|---|
| | <p>FCCOB Endianness and Multi-Byte Access :</p> <p>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).</p> |

27.3.3.6 Program Flash Protection Registers (FTFA_FPROTn)

The FPROT registers define which 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 KB of program flash where each assigned bit protects 1 KB . For configurations with 24 KB of program flash memory or less, FPROT0 is not used. For configurations with 16 KB of program flash memory or less, FPROT1 is not used. For configurations with 8 KB 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 |
| FPROT2 | 0x0009 |
| FPROT3 | 0x0008 |

Functional Description

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

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|------------|----|----|----|----|------|----|----|----|
| Read Write | | | | | PROT | | | |
| Reset | x* | x* | x* | x* | x* | x* | x* | x* |

* Notes:

- x = Undefined at reset.

FTFA_FPROTn field descriptions

| Field | Description |
|-------|---|
| 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>In NVM Normal mode: 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>In NVM Special mode: All bits of FPROT are writable without restriction. Unprotected areas can be protected and protected areas can be unprotected.</p> <p>Restriction: 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 KB. For configurations with less than 32 KB of program flash memory, each assigned bit represents 1 KB.</p> <p>0 Program flash region is protected. 1 Program flash region is not protected</p> |

27.4 Functional Description

The information found here describes 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** —
 - For 2^n program flash sizes, four registers typically protect 32 regions of the program flash memory as shown in the following figure

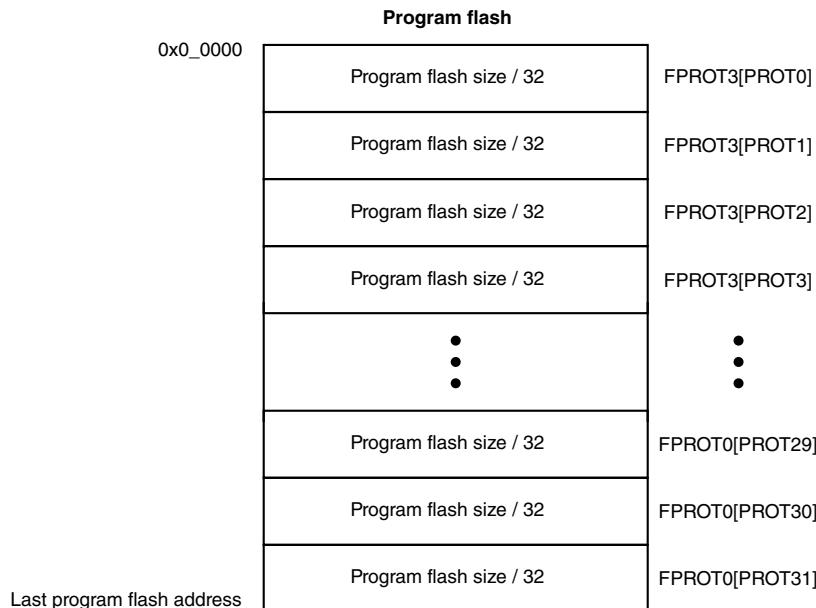


Figure 27-2. Program flash protection

NOTE

Flash protection features are discussed further in [AN4507: Using the Kinetis Security and Flash Protection Features](#). Not all features described in the application note are available on this device.

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 27-1. 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 27-2](#)). 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 27-3](#). 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.

Attempts to launch a flash command in VLP mode will be 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 FSTAT[CCIF] by writing a '1' to it. FSTAT[CCIF] remains 0 until the flash command completes.

The FSTAT register contains a blocking mechanism that prevents a new command from launching (can't clear FSTAT[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. FSTAT[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, 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 FSTAT[CCIF].

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 FSTAT[MGSTAT0]. 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 FSTAT[CCIF] signifying that the command has completed.

The flow for a generic command write sequence is illustrated in the following figure.

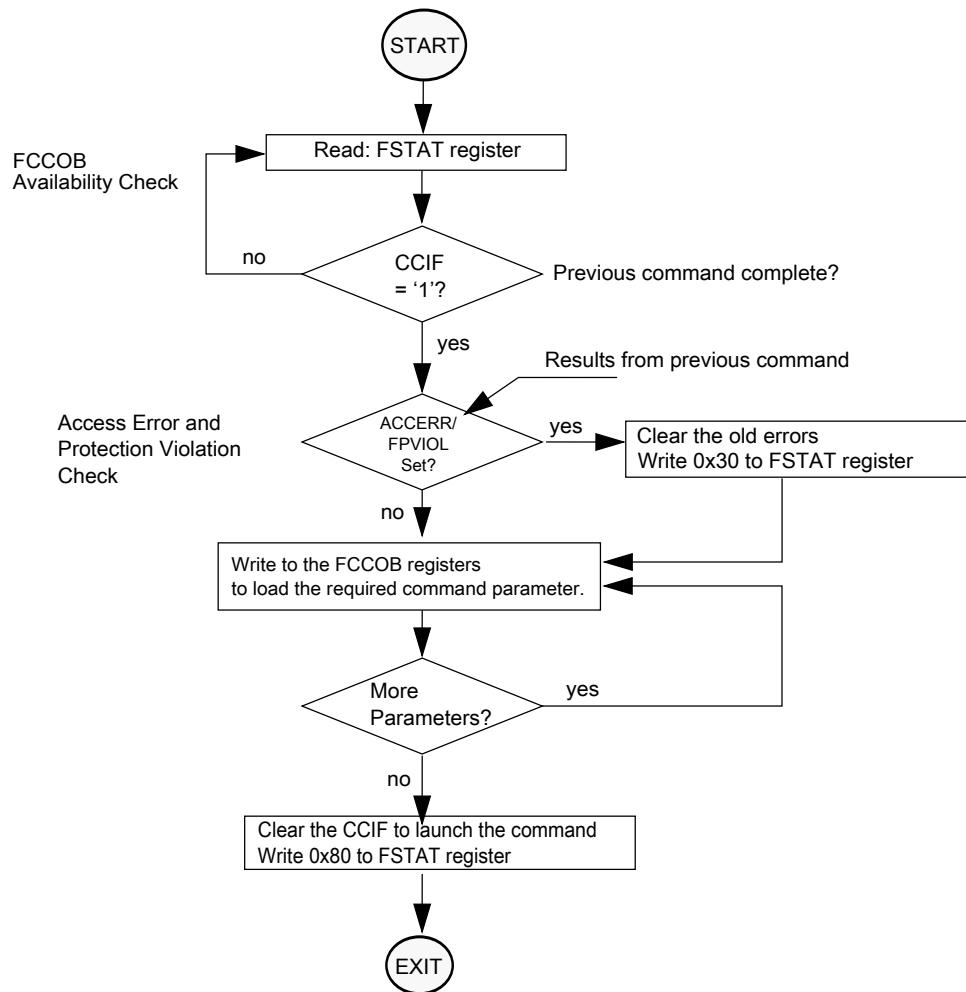


Figure 27-3. Generic flash command write sequence flowchart

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 | x | Erase all bytes in a program flash sector. |
| 0x40 | Read 1s All Blocks | x | 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 | x | Erase the program flash block, verify-erase and release MCU security. NOTE: An erase is only possible when all memory locations are unprotected. |
| 0x45 | Verify Backdoor Access Key | x | Release MCU security after comparing a set of user-supplied security keys to those stored in the program flash. |
| 0x49 | Erase All Blocks Unsecure | x | Erase the program flash block, verify-erase, program security byte to unsecure state, release MCU security. |

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 27-2. Flash Commands by Mode

| FCMD | Command | NVM Normal | | | NVM Special | | |
|------|--------------------|------------|--------|---------|-------------|--------|---------|
| | | Unsecure | Secure | MEEN=10 | Unsecure | Secure | MEEN=10 |
| 0x01 | Read 1s Section | x | x | x | x | — | — |
| 0x02 | Program Check | x | x | x | x | — | — |
| 0x03 | Read Resource | x | x | x | x | — | — |
| 0x06 | Program Longword | x | x | x | x | — | — |
| 0x09 | Erase Flash Sector | x | x | x | x | — | — |
| 0x40 | Read 1s All Blocks | x | x | x | x | x | — |
| 0x41 | Read Once | x | x | x | x | — | — |
| 0x43 | Program Once | x | x | x | x | — | — |
| 0x44 | Erase All Blocks | x | x | x | x | x | — |

Table continues on the next page...

Table 27-2. Flash Commands by Mode (continued)

| FCMD | Command | NVM Normal | | | NVM Special | | |
|------|----------------------------|------------|--------|---------|-------------|--------|---------|
| | | Unsecure | Secure | MEEN=10 | Unsecure | Secure | MEEN=10 |
| 0x45 | Verify Backdoor Access Key | × | × | × | × | — | — |
| 0x49 | Erase All Blocks Unsecure | × | × | × | × | × | — |

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 FSTAT[ACCERR] and FSTAT[FPVIOL] 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 (FSTAT[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 27-3. 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] ¹ 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).

Upon clearing CCIF to launch the Read 1s Section command, the flash memory module sets the read margin for 1s according to [Table 27-4](#) and then reads all locations within the specified section of flash memory. If the flash memory module fails to read all 1s (that is, the flash section is not erased), FSTAT[MGSTAT0] is set. FSTAT[CCIF] sets after the Read 1s Section operation completes.

Table 27-4. 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 27-5. 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 0. | 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 27-6. 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] ¹ |
| 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 27-7](#), 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, FSTAT[MGSTAT0] 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, FSTAT[MGSTAT0] is set. FSTAT[CCIF] 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 27-7. 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 27-8. 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] |

Table continues on the next page...

Table 27-8. Program Check Command Error Handling (continued)

| Error Condition | Error Bit |
|---|----------------|
| 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 27-10](#).

Table 27-9. 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] ¹ |
| 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 Table 27-10) |

1. Must be longword aligned (Flash address [1:0] = 00).

Table 27-10. 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 ¹ | Version ID | 8 Bytes | 0x00_0000–0x00_0007 |

1. Located in program flash 0 reserved space.

After clearing CCIF to launch the Read Resource command, four consecutive bytes are read from the selected resource at the provided relative address and stored in the FCCOB register. The CCIF flag sets after the Read Resource operation completes. The Read Resource command exits with an access error if an invalid resource code is provided or if the address for the applicable area is out-of-range.

Table 27-11. Read Resource Command Error Handling

| Error Condition | Error Bit |
|--|---------------|
| Command not available in current mode/security | FSTAT[ACCERR] |
| An invalid resource code is entered | FSTAT[ACCERR] |
| Flash address is out-of-range for the targeted resource. | FSTAT[ACCERR] |
| Flash address is not longword aligned | FSTAT[ACCERR] |

27.4.10.4 Program Longword Command

The Program Longword command programs four previously-erased bytes in the program flash memory using an embedded algorithm.

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 27-12. 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] ¹ |
| 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).

Functional Description

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 27-13. 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] |
| 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 27-14. 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] ¹ 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 27-4](#)).

Table 27-15. 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[FPVIOL] |
| Any errors have been encountered during the verify operation ¹ | FSTAT[MGSTAT0] |

1. User margin read may be run using the Read 1s Section command to verify all bits are erased.

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.

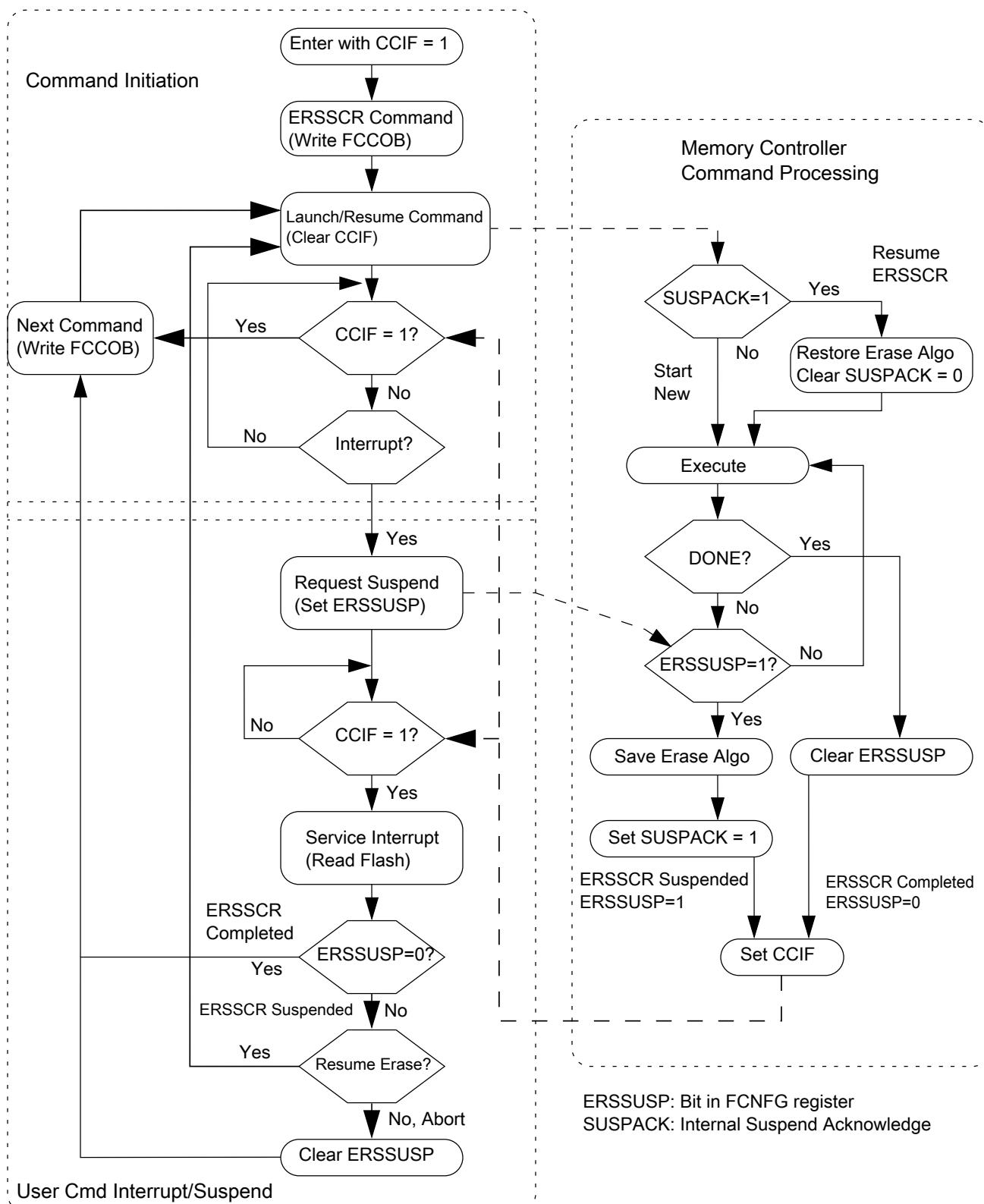


Figure 27-4. Suspend and Resume of Erase Flash Sector Operation

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 27-16. 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 27-17](#),
- 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 27-17. 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 27-18. 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 special 64-byte fields located 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 (index values 0x00 - 0x0F), each 4 bytes long. These fields are programmed using the Program Once command described in [Program Once Command](#).

Table 27-19. Read Once Command FCCOB Requirements

| FCCOB Number | FCCOB Contents [7:0] |
|-----------------|---|
| 0 | 0x41 (RDONCE) |
| 1 | Program Once record index (0x00 - 0x0F) |
| 2 | Not used |
| 3 | Not used |
| Returned Values | |
| 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 Read Once command, a 4-byte Program Once record is read 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 - 0x0F. During execution of the Read Once command, any attempt to read addresses within the program flash block containing the selected record index returns invalid data. The Read Once command can be executed any number of times.

Table 27-20. 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 special 64-byte fields 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 (index values 0x00 - 0x0F), each 4 bytes long. These records can be read using the Read Once command (see [Read Once Command](#)) or using the Read Resource command (see [Read Resource Command](#)). These records can be programmed only once since the program flash 0 IFR cannot be erased.

Table 27-21. 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.

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 - 0x0F. During execution of the Program Once command, any attempt to read addresses within the program flash block containing the selected record index returns invalid data.

Table 27-22. 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 ¹ | 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 27-23. 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 27-24. 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 ¹ | FSTAT[MGSTAT0] |

1. User margin read may be run using the Read 1s All Blocks command to verify all bits are erased.

27.4.10.9.1 Triggering an Erase All External to the Flash Memory Module

The functionality of the Erase All Blocks/Erase All Blocks Unsecure 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[EFSREQ] bit. The FCNFG[EFSREQ] 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

Functional Description

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 27-25. 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_0003 |
| 5 | Key Byte 1 | 0x0_0002 |
| 6 | Key Byte 2 | 0x0_0001 |
| 7 | Key Byte 3 | 0x0_0000 |
| 8 | Key Byte 4 | 0x0_0007 |
| 9 | Key Byte 5 | 0x0_0006 |
| A | Key Byte 6 | 0x0_0005 |
| B | Key Byte 7 | 0x0_0004 |

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 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 27-26. 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] |
| 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] |

27.4.10.11 Erase All Blocks Unsecure Command

The Erase All Blocks Unsecure operation erases all flash memory, verifies all memory contents, programs the security byte in the Flash Configuration Field to the unsecure state, and releases MCU security.

Table 27-27. Erase All Blocks Unsecure Command FCCOB Requirements

| FCCOB Number | FCCOB Contents [7:0] |
|--------------|----------------------|
| 0 | 0x49 (ERSALLU) |

After clearing CCIF to launch the Erase All Blocks Unsecure command, the flash memory module erases all program flash memory, then verifies that all are erased.

If the flash memory module verifies that all program flash memory was properly erased, security is released by setting the FSEC[SEC] field to the unsecure state, and the security byte (see [Flash Configuration Field Description](#)) is programmed to the unsecure state by the Erase All Blocks Unsecure command. If the erase or program verify fails, the FSTAT[MGSTAT0] bit is set. The CCIF flag is set after the Erase All Blocks Unsecure operation completes.

Table 27-28. Erase All Blocks Unsecure Command Error Handling

| Error Condition | Error Bit |
|--|----------------|
| Command not available in current mode/security | FSTAT[ACCERR] |
| Any errors have been encountered during erase or program verify operations | FSTAT[MGSTAT0] |

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.

Flash security features are discussed further in [AN4507: Using the Kinetis Security and Flash Protection Features](#). Note that not all features described in the application note are available on this device.

Table 27-29. 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 27-30. 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, Erase All Blocks Unsecure 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.

Functional Description

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 Chip-specific ADC information

This device contains one 12-bit successive approximation ADC with up to 7 channels.

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 28-1. Number of KL03 ADC channels

| Device | Number of ADC Channels | Package |
|---|------------------------|--------------|
| MKL03Z8VFG4(R) MKL03Z16VFG4(R) MKL03Z32VFG4(R) | 4 | 16-pin QFN |
| MKL03Z8VFK4(R) MKL03Z16VFK4(R) MKL03Z32VFK4(R) | 7 | 24-pin QFN |
| MKL03Z32CAF4R MKL03Z32CBF4R | 7 | 20-pin WLCSP |

28.1.1 ADC0 connections/channel assignment

Table 28-2. ADC0 channel assignment

| ADC Channel (SC1n[ADCH]) | Channel | Input signal |
|--------------------------|---------|--------------|
| 00000 | AD0 | ADC0_SE0 |
| 00001 | AD1 | ADC0_SE1 |
| 00010 | AD2 | ADC0_SE2 |
| 00011 | AD3 | ADC0_SE3 |
| 00100 | AD4 | Reserved |
| 00101 | AD5 | Reserved |
| 00110 | AD6 | Reserved |

Table continues on the next page...

Table 28-2. ADC0 channel assignment (continued)

| ADC Channel (SC1n[ADCH]) | Channel | Input signal |
|--------------------------|---------|----------------------------|
| 00111 | AD7 | Reserved |
| 01000 | AD8 | ADC0_SE8 |
| 01001 | AD9 | ADC0_SE9 |
| 01010 | AD10 | Reserved |
| 01011 | AD11 | Reserved |
| 01100 | AD12 | Reserved |
| 01101 | AD13 | Reserved |
| 01110 | AD14 | Reserved |
| 01111 | AD15 | ADC0_SE15 |
| 10000 | AD16 | Reserved |
| 10001 | AD17 | Reserved |
| 10010 | AD18 | Reserved |
| 10011 | AD19 | Reserved |
| 10100 | AD20 | Reserved |
| 10101 | AD21 | Reserved |
| 10110 | AD22 | Reserved |
| 10111 | AD23 | Reserved |
| 11000 | AD24 | Reserved |
| 11001 | AD25 | Reserved |
| 11010 | AD26 | Temperature Sensor (S.E) |
| 11011 | AD27 | Bandgap (S.E) ¹ |
| 11100 | AD28 | Reserved |
| 11101 | AD29 | VDD (S.E) |
| 11110 | AD30 | VSS |
| 11111 | AD31 | Module Disabled |

1. 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 (VBG) specification.

28.1.2 ADC analog supply and reference connections

Pin PTB2 can be configured to VREF_OUT, which needs to connect a capacitor to ground when 1.2 V VREF is enabled.

28.1.3 ADC Reference Options

The ADC supports the following references:

- VDD/VSS - connected as the primary reference option
- VREF_OUT - connected as the V_{ALT} reference option

28.1.4 Alternate clock

For this device, the alternate clock is connected to the external reference clock (OSCERCLK).

NOTE

This clock option is only usable when OSCERCLK is in the MHz range. A system with OSCERCLK in the kHz range has the optional clock source below minimum ADC clock operating frequency.

28.2 Introduction

The 12-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.

28.2.1 Features

Following are the features of the ADC module.

- Linear successive approximation algorithm with up to 12-bit resolution
- Up to 24 single-ended external analog inputs
- Output modes:
 - single-ended 12-bit, 10-bit, and 8-bit 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

28.2.2 Block diagram

The following figure is the ADC module block diagram.

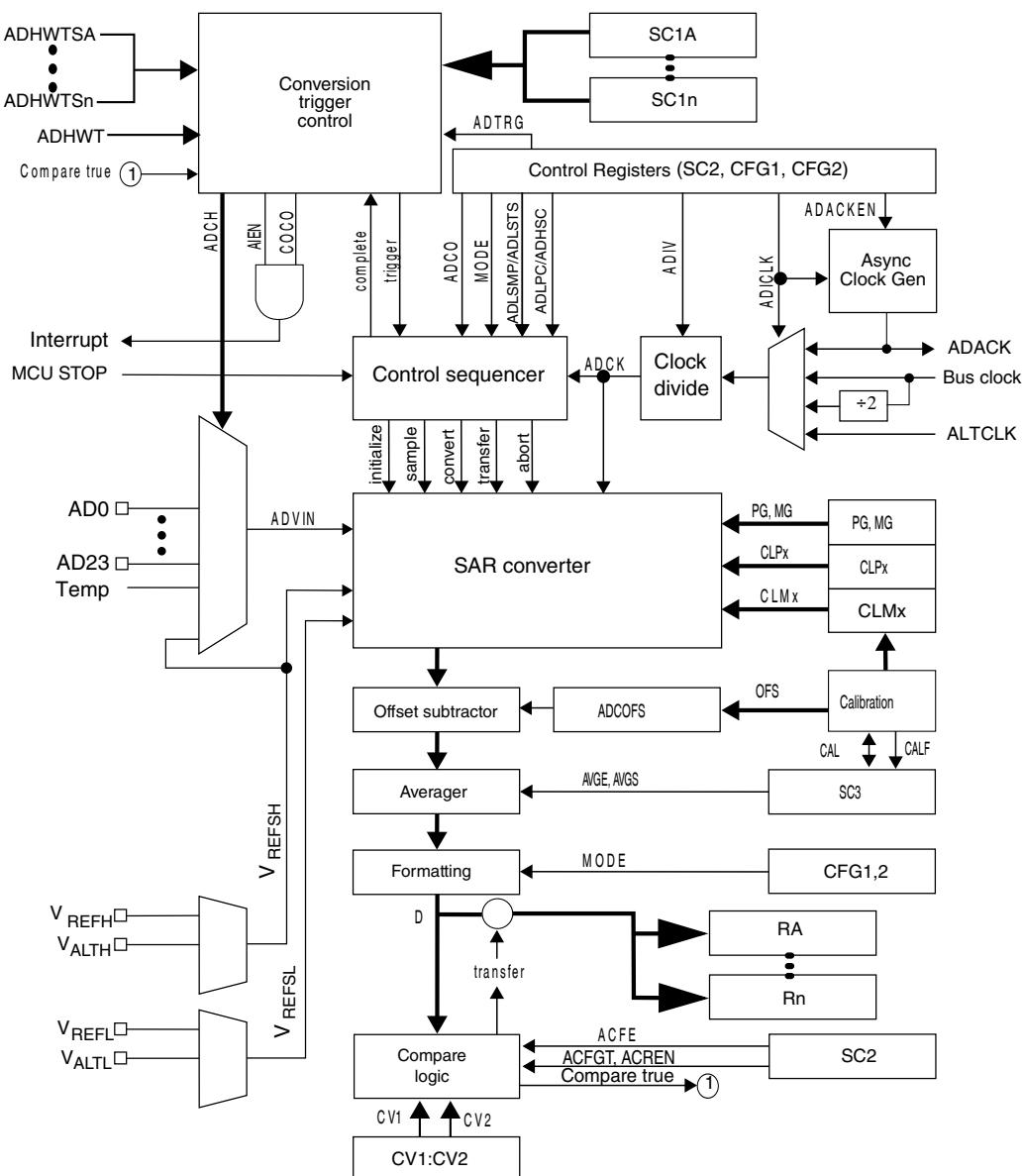


Figure 28-1. ADC block diagram

28.3 ADC signal descriptions

The ADC module supports up to 24 single-ended inputs.

The ADC also requires four supply/reference/ground connections.

NOTE

For the number of channels supported on this device, see the chip-specific ADC information.

Table 28-3. ADC signal descriptions

| Signal | Description | I/O |
|------------------|------------------------------------|-----|
| ADn | Single-Ended Analog Channel Inputs | I |
| V _{DDA} | Analog Power Supply | I |
| V _{SSA} | Analog Ground | I |

28.3.1 Analog Power (V_{DDA})

The ADC analog portion uses V_{DDA} as its power connection. In some packages, V_{DDA} is connected internally to V_{DD}. If externally available, connect the V_{DDA} pin to the same voltage potential as V_{DD}. External filtering may be necessary to ensure clean V_{DDA} for good results.

28.3.2 Analog Ground (V_{SSA})

The ADC analog portion uses V_{SSA} as its ground connection. In some packages, V_{SSA} is connected internally to V_{SS}. If externally available, connect the V_{SSA} pin to the same voltage potential as V_{SS}.

28.3.3 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.

28.4 Memory map and register definitions

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 | 28.4.1/383 |
| 4003_B004 | ADC Status and Control Registers 1 (ADC0_SC1B) | 32 | R/W | 0000_001Fh | 28.4.1/383 |
| 4003_B008 | ADC Configuration Register 1 (ADC0_CFG1) | 32 | R/W | 0000_0000h | 28.4.2/387 |
| 4003_B00C | ADC Configuration Register 2 (ADC0_CFG2) | 32 | R/W | 0000_0000h | 28.4.3/388 |

Table continues on the next page...

ADC memory map (continued)

| Absolute address (hex) | Register name | Width (in bits) | Access | Reset value | Section/page |
|------------------------|--|-----------------|--------|-------------|-----------------------------|
| 4003_B010 | ADC Data Result Register (ADC0_RA) | 32 | R | 0000_0000h | 28.4.4/389 |
| 4003_B014 | ADC Data Result Register (ADC0_RB) | 32 | R | 0000_0000h | 28.4.4/389 |
| 4003_B018 | Compare Value Registers (ADC0_CV1) | 32 | R/W | 0000_0000h | 28.4.5/390 |
| 4003_B01C | Compare Value Registers (ADC0_CV2) | 32 | R/W | 0000_0000h | 28.4.5/390 |
| 4003_B020 | Status and Control Register 2 (ADC0_SC2) | 32 | R/W | 0000_0000h | 28.4.6/391 |
| 4003_B024 | Status and Control Register 3 (ADC0_SC3) | 32 | R/W | 0000_0000h | 28.4.7/393 |
| 4003_B028 | ADC Offset Correction Register (ADC0_OFS) | 32 | R/W | 0000_0004h | 28.4.8/395 |
| 4003_B02C | ADC Plus-Side Gain Register (ADC0_PG) | 32 | R/W | 0000_8200h | 28.4.9/395 |
| 4003_B034 | ADC Plus-Side General Calibration Value Register (ADC0_CLPD) | 32 | R/W | 0000_000Ah | 28.4.10/396 |
| 4003_B038 | ADC Plus-Side General Calibration Value Register (ADC0_CLPS) | 32 | R/W | 0000_0020h | 28.4.11/396 |
| 4003_B03C | ADC Plus-Side General Calibration Value Register (ADC0_CLP4) | 32 | R/W | 0000_0200h | 28.4.12/397 |
| 4003_B040 | ADC Plus-Side General Calibration Value Register (ADC0_CLP3) | 32 | R/W | 0000_0100h | 28.4.13/397 |
| 4003_B044 | ADC Plus-Side General Calibration Value Register (ADC0_CLP2) | 32 | R/W | 0000_0080h | 28.4.14/398 |
| 4003_B048 | ADC Plus-Side General Calibration Value Register (ADC0_CLP1) | 32 | R/W | 0000_0040h | 28.4.15/398 |
| 4003_B04C | ADC Plus-Side General Calibration Value Register (ADC0_CLP0) | 32 | R/W | 0000_0020h | 28.4.16/399 |

28.4.1 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.

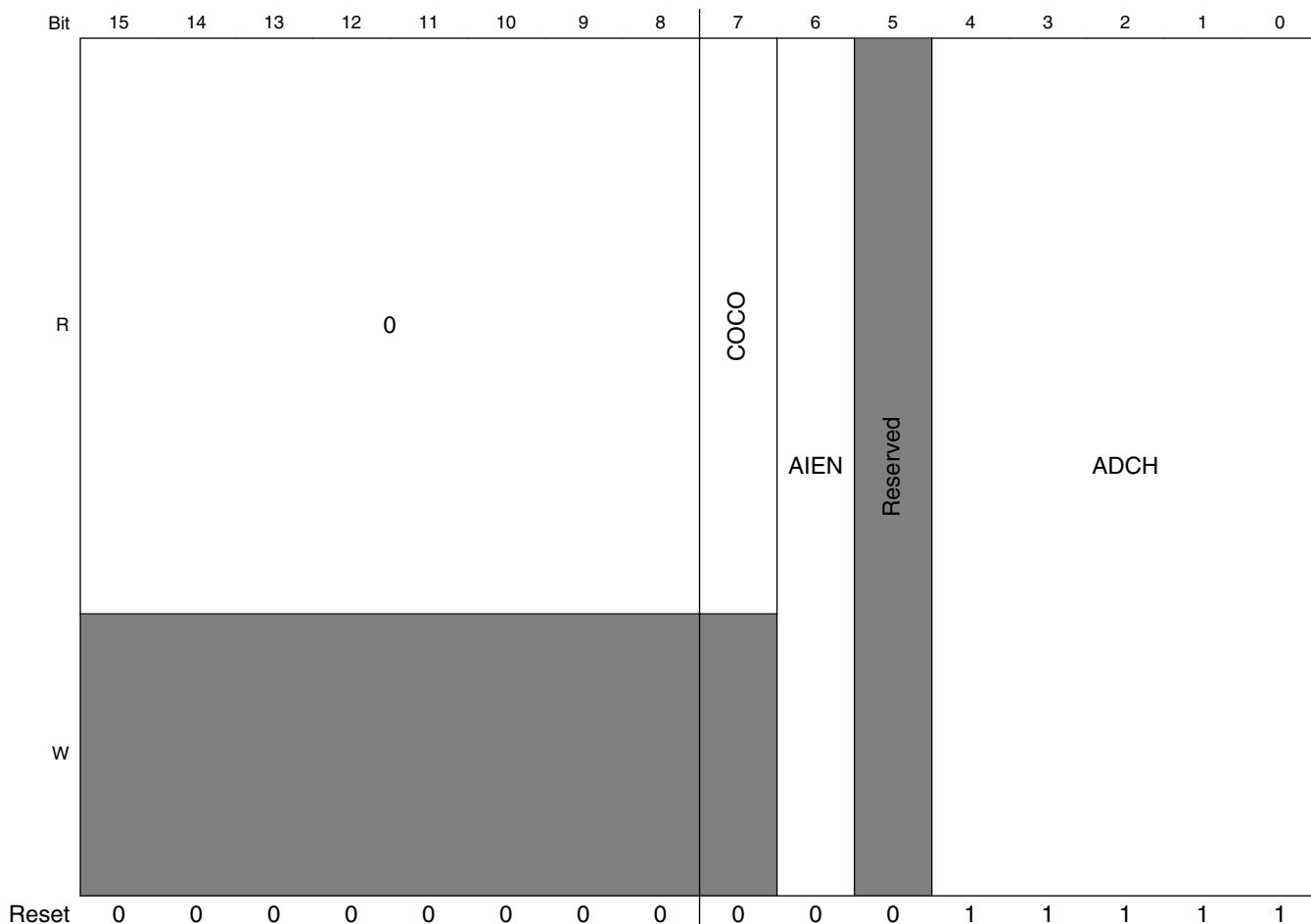
Writing SC1A while SC1A is actively controlling a conversion aborts the current conversion. In Software Trigger mode, when SC2[ADTRG]=0, writes to SC1A subsequently initiate a new conversion, if SC1[ADCH] contains a value other than all 1s.

Memory map and register definitions

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 |

**ADCx_SC1n field descriptions**

| Field | Description |
|------------------|--|
| 31–8 Reserved | This field is reserved. This read-only field is reserved and always has the value 0. |
| 7 COCO | Conversion Complete Flag 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. 0 Conversion is not completed. 1 Conversion is completed. |
| 6 AIEN | Interrupt Enable Enables conversion complete interrupts. When COCO becomes set while the respective AIEN is high, an interrupt is asserted. 0 Conversion complete interrupt is disabled. 1 Conversion complete interrupt is enabled. |

Table continues on the next page...

ADCx_SC1n field descriptions (continued)

| Field | Description |
|---------------|---|
| 5 Reserved | This field is reserved. This reserved bit should not be changed. |
| ADCH | <p>Input channel select</p> <p>Selects one of the input channels.</p> <p>NOTE: 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> <ul style="list-style-type: none"> 00000 AD0 is selected as input. 00001 AD1 is selected as input. 00010 AD2 is selected as input. 00011 AD3 is selected as input. 00100 AD4 is selected as input. 00101 AD5 is selected as input. 00110 AD6 is selected as input. 00111 AD7 is selected as input. 01000 AD8 is selected as input. 01001 AD9 is selected as input. 01010 AD10 is selected as input. 01011 AD11 is selected as input. 01100 AD12 is selected as input. 01101 AD13 is selected as input. 01110 AD14 is selected as input. 01111 AD15 is selected as input. 10000 AD16 is selected as input. 10001 AD17 is selected as input. 10010 AD18 is selected as input. 10011 AD19 is selected as input. 10100 AD20 is selected as input. 10101 AD21 is selected as input. 10110 AD22 is selected as input. 10111 AD23 is selected as input. 11000 Reserved. 11001 Reserved. 11010 Temp Sensor (single-ended) is selected as input. 11011 Bandgap (single-ended) is selected as input. 11100 Reserved. 11101 V_{REFSH} is selected as input. Voltage reference selected is determined by SC2[REFSEL]. 11110 V_{REFSL} is selected as input. Voltage reference selected is determined by SC2[REFSEL]. 11111 Module is disabled. |

28.4.2 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 | 0 | | | | | | | | | 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 Reserved | This field is reserved. This read-only field is reserved and always has the value 0. |
| 7 ADLPC | Low-Power Configuration Controls the power configuration of the successive approximation converter. This optimizes power consumption when higher sample rates are not required. 0 Normal power configuration. 1 Low-power configuration. The power is reduced at the expense of maximum clock speed. |
| 6–5 ADIV | Clock Divide Select Selects the divide ratio used by the ADC to generate the internal clock ADCK. 00 The divide ratio is 1 and the clock rate is input clock. 01 The divide ratio is 2 and the clock rate is (input clock)/2. 10 The divide ratio is 4 and the clock rate is (input clock)/4. 11 The divide ratio is 8 and the clock rate is (input clock)/8. |
| 4 ADLSMP | Sample Time Configuration Selects between different sample times based on the conversion mode selected. This field 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. |

Table continues on the next page...

ADCx_CFG1 field descriptions (continued)

| Field | Description |
|----------|---|
| | 0 Short sample time. 1 Long sample time. |
| 3–2 MODE | Conversion mode selection Selects the ADC resolution mode. 00 It is single-ended 8-bit conversion. 01 It is single-ended 12-bit conversion . 10 It is single-ended 10-bit conversion. 11 Reserved. Do not set the field to this value. |
| ADICLK | Input Clock Select 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. 00 Bus clock 01 Bus clock divided by 2(BUSCLK/DIV2) 10 Alternate clock (ALTCLK) 11 Asynchronous clock (ADACK) |

28.4.3 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 | | | | MUXSEL | ADACKEN | ADHSC | ADLSTS |
| W | | | | | | | | | | | | | | | | |
| 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 Reserved | This field is reserved. This read-only field is reserved and always has the value 0. |
| 7–5 Reserved | This field is reserved. This read-only field is reserved and always has the value 0. |
| 4 MUXSEL | ADC Mux Select Changes the ADC mux setting to select between alternate sets of ADC channels. 0 ADxxa channels are selected. 1 ADxxb channels are selected. |
| 3 ADACKEN | Asynchronous Clock Output Enable 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. 0 Asynchronous clock output disabled; Asynchronous clock is enabled only if selected by ADICLK and a conversion is active. 1 Asynchronous clock and clock output is enabled regardless of the state of the ADC. |
| 2 ADHSC | High-Speed Configuration 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. 0 Normal conversion sequence selected. 1 High-speed conversion sequence selected with 2 additional ADCK cycles to total conversion time. |
| ADLSTS | Long Sample Time Select 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. 00 Default longest sample time; 20 extra ADCK cycles; 24 ADCK cycles total. 01 12 extra ADCK cycles; 16 ADCK cycles total sample time. 10 6 extra ADCK cycles; 10 ADCK cycles total sample time. 11 2 extra ADCK cycles; 6 ADCK cycles total sample time. |

28.4.4 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-aligned modes and carry the sign bit (MSB) in sign-extended 2's complement modes.

Memory map and register definitions

The following table describes the behavior of the data result registers in the different modes of operation.

Table 28-4. Data result register description

| Conversion mode | D15 | D14 | D13 | D12 | D11 | D10 | D9 | D8 | D7 | D6 | D5 | D4 | D3 | D2 | D1 | D0 | Format |
|---------------------|-----|-----|-----|-----|-----|-----|----|----|----|----|----|----|----|----|----|--------------------------|--------|
| 12-bit single-ended | 0 | 0 | 0 | 0 | D | D | D | D | D | D | D | D | D | D | D | Unsigned right-justified | |
| 10-bit single-ended | 0 | 0 | 0 | 0 | 0 | 0 | D | D | D | D | D | D | D | D | D | Unsigned right-justified | |
| 8-bit single-ended | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 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 | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| 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 Reserved | This field is reserved. This read-only field is reserved and always has the value 0. |
| D | Data result |

28.4.5 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.

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 | | | | | | | | | | | | | | | | | 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 | | |

ADCx_CVn field descriptions

| Field | Description |
|-------------------|---|
| 31–16 Reserved | This field is reserved. This read-only field is reserved and always has the value 0. |
| CV | Compare Value. |

28.4.6 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 | | | | | | | | | | 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 | | | | | | | | | ADACT | ADTRG | ACFE | ACFGT | ACREN | 0 | | REFSEL |
| W | | | | | | | | | | | | | | | | |
| 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 Reserved | This field is reserved. This read-only field is reserved and always has the value 0. |
| 7 ADACT | Conversion Active 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. 0 Conversion not in progress. 1 Conversion in progress. |
| 6 ADTRG | Conversion Trigger Select Selects the type of trigger used for initiating a conversion. Two types of trigger are selectable: <ul style="list-style-type: none"> • Software trigger: When software trigger is selected, a conversion is initiated following a write to SC1A. • 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. 0 Software trigger selected. 1 Hardware trigger selected. |
| 5 ACFE | Compare Function Enable Enables the compare function. 0 Compare function disabled. 1 Compare function enabled. |
| 4 ACFGT | Compare Function Greater Than Enable 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. 0 Configures less than threshold, outside range not inclusive and inside range not inclusive; functionality based on the values placed in CV1 and CV2. 1 Configures greater than or equal to threshold, outside and inside ranges inclusive; functionality based on the values placed in CV1 and CV2. |
| 3 ACREN | Compare Function Range Enable 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. 0 Range function disabled. Only CV1 is compared. 1 Range function enabled. Both CV1 and CV2 are compared. |
| 2 Reserved | This field is reserved. This read-only field is reserved and always has the value 0. |
| REFSEL | Voltage Reference Selection Selects the voltage reference source used for conversions. 00 Default voltage reference pin pair, that is, external pins V_{REFH} and V_{REFL} 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 |

Table continues on the next page...

ADCx_SC2 field descriptions (continued)

| Field | Description |
|-------|-------------|
| | 10 Reserved |
| | 11 Reserved |

28.4.7 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 | | | | | | | | | CAL | 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 Reserved | This field is reserved. This read-only field is reserved and always has the value 0. |
| 7 CAL | Calibration 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 |

Table continues on the next page...

ADCx_SC3 field descriptions (continued)

| Field | Description |
|-----------------|--|
| | 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 CALF | <p>Calibration Failed Flag</p> <p>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.</p> <ul style="list-style-type: none"> 0 Calibration completed normally. 1 Calibration failed. ADC accuracy specifications are not guaranteed. |
| 5–4 Reserved | <p>This field is reserved.</p> <p>This read-only field is reserved and always has the value 0.</p> |
| 3 ADCO | <p>Continuous Conversion Enable</p> <p>Enables continuous conversions.</p> <ul style="list-style-type: none"> 0 One conversion or one set of conversions if the hardware average function is enabled, that is, AVGE=1, after initiating a conversion. 1 Continuous conversions or sets of conversions if the hardware average function is enabled, that is, AVGE=1, after initiating a conversion. |
| 2 AVGE | <p>Hardware Average Enable</p> <p>Enables the hardware average function of the ADC.</p> <ul style="list-style-type: none"> 0 Hardware average function disabled. 1 Hardware average function enabled. |
| AVGS | <p>Hardware Average Select</p> <p>Determines how many ADC conversions will be averaged to create the ADC average result.</p> <ul style="list-style-type: none"> 00 4 samples averaged. 01 8 samples averaged. 10 16 samples averaged. 11 32 samples averaged. |

28.4.8 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 | 0 | 1 | 0 | 0 | |

ADCx_OFS field descriptions

| Field | Description |
|-------------------|---|
| 31–16 Reserved | This field is reserved. This read-only field is reserved and always has the value 0. |
| OFS | Offset Error Correction Value |

28.4.9 ADC Plus-Side Gain Register (ADCx_PG)

The Plus-Side Gain Register (PG) contains the gain error correction for 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 | 0 | 0 | 1 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

ADCx_PG field descriptions

| Field | Description |
|-------------------|---|
| 31–16 Reserved | This field is reserved. This read-only field is reserved and always has the value 0. |
| PG | Plus-Side Gain |

28.4.10 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.

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 | 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_CLPD field descriptions

| Field | Description |
|------------------|---|
| 31–6 Reserved | This field is reserved. This read-only field is reserved and always has the value 0. |
| CLPD | Calibration Value Calibration Value |

28.4.11 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 | 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_CLPS field descriptions

| Field | Description |
|------------------|---|
| 31–6 Reserved | This field is reserved. This read-only field is reserved and always has the value 0. |
| CLPS | Calibration Value Calibration Value |

28.4.12 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 | 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 | 0 | | | |

ADCx_CLP4 field descriptions

| Field | Description |
|-------------------|---|
| 31–10 Reserved | This field is reserved. This read-only field is reserved and always has the value 0. |
| CLP4 | Calibration Value Calibration Value |

28.4.13 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 | 1 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | | | |

ADCx_CLP3 field descriptions

| Field | Description |
|------------------|---|
| 31–9 Reserved | This field is reserved. This read-only field is reserved and always has the value 0. |

Table continues on the next page...

ADCx_CLP3 field descriptions (continued)

| Field | Description |
|-------|-------------------|
| CLP3 | Calibration Value |
| | Calibration Value |

28.4.14 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 | 0 | 1 | 0 | 0 | 0 | 0 | 0 | | |

ADCx_CLP2 field descriptions

| Field | Description |
|------------------|---|
| 31–8 Reserved | This field is reserved. This read-only field is reserved and always has the value 0. |
| CLP2 | Calibration Value |
| | Calibration Value |

28.4.15 ADC Plus-Side General Calibration Value Register (ADCx_CLP1)

For more information, see CLPD register description.

Address: 4003_B000h base + 48h offset = 4003_B048h

| 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_CLP1 field descriptions

| Field | Description |
|------------------|---|
| 31–7 Reserved | This field is reserved. This read-only field is reserved and always has the value 0. |
| CLP1 | Calibration Value |
| | Calibration Value |

28.4.16 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 | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| 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_CLP0 field descriptions

| Field | Description |
|------------------|---|
| 31–6 Reserved | This field is reserved. This read-only field is reserved and always has the value 0. |
| CLP0 | Calibration Value Calibration Value |

28.5 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.

28.5.1 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. Conversions are possible using ALTCLK as the input clock source while the MCU is in Normal Stop mode.
- 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 µs 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.

28.5.2 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.

28.5.3 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-specific ADC information 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, 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:

- ADHWTSn active selects SC1A.
- ADHWTSn active selects SC1n.

Note

Asserting more than one hardware trigger select signal (ADHWT n) at the same time results in unknown results. To avoid this, select only one hardware trigger select signal (ADHWT n) prior to the next intended conversion.

When the conversion is completed, the result is placed in the R n registers associated with the ADHWT n received. For example:

- ADHWTSA active selects RA register
- ADHWT n active selects R n register

The conversion complete flag associated with the ADHWT n received, that is, SC1 n [COCO], is then set and an interrupt is generated if the respective conversion complete interrupt has been enabled, that is, SC1[AIEN]=1.

28.5.4 Conversion control

Conversions can be performed as determined by CFG1[MODE] 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

28.5.4.1 Initiating conversions

A conversion is initiated:

- Following a write to SC1A, with SC1 n [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, ADHWT n , has occurred. The channel and status fields selected depend on the active trigger select signal:
 - ADHWTSA active selects SC1A.

- ADHWT_n active selects SC1_n.
- if neither is active, the off condition is selected

Note

Selecting more than one ADHWT_n prior to a conversion completion will result in unknown results. To avoid this, select only one ADHWT_n prior to a conversion completion.

- Following the transfer of the result to the data registers when continuous conversion is enabled, that is, when SC3[ADCO] = 1.

If continuous conversions are enabled, a new conversion is automatically initiated after the completion of the current conversion. In software triggered operation, that is, when SC2[ADTRG] = 0, continuous conversions begin after SC1A is written and continue until aborted. In hardware triggered operation, that is, when SC2[ADTRG] = 1 and one ADHWT_n 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.

28.5.4.2 Completing conversions

A conversion is completed when the result of the conversion is transferred into the data result registers, R_n. If the compare functions are disabled, this is indicated by setting of SC1_n[COCO]. If hardware averaging is enabled, the respective SC1_n[COCO] sets only if the last of the selected number of conversions is completed. If the compare function is enabled, the respective SC1_n[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 SC1_n[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 SC1_n[AIEN] is high at the time that the respective SC1_n[COCO] is set.

28.5.4.3 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 or Alternate Clock Sources not enabled.

When a conversion is aborted, the contents of the data registers, R_n, 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_n return to their reset states.

28.5.4.4 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_{ADCK}.

28.5.4.5 Sample time and total conversion time

For short sample, that is, when $\text{CFG1[ADLSMP]}=0$, 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 $\text{CFG2[ADHSC]}=1$, there is an additional 2-cycle adder on any conversion. The table below summarizes sample times for the possible ADC configurations.

| 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]
- 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 $\text{CFG2[ADHSC]}=1$ take two more ADCK cycles. CFG2[ADHSC] must be used when the ADCLK exceeds the limit for $\text{CFG2[ADHSC]}=0$.

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 R_n upon completion of the conversion algorithm.

Functional description

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.

$$\text{ConversionTime} = \text{SFCAdder} + \text{AverageNum} \times (\text{BCT} + \text{LSTAdder} + \text{HSCAdder})$$

Equation 1. Conversion time equation

Table 28-5. Single or first continuous time adder (SFCAdder)

| CFG1[ADLSMP] | CFG2[ADACKEN] | 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 ¹ |
| 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 ¹ |
| 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 28-6. 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 28-7. Base conversion time (BCT)

| Mode | Base conversion time (BCT) |
|------------------|-----------------------------------|
| 8b single-ended | 17 ADCK cycles |
| 10b single-ended | 20 ADCK cycles |
| 12b single-ended | 20 ADCK cycles |

Table 28-8. 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 28-9. 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.

28.5.4.6 Conversion time examples

The following examples use the [Equation 1 on page 406](#), and the information provided in [Table 28-5](#) through [Table 28-9](#).

28.5.4.6.1 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 [Equation 1 on page 406](#), and the information provided in [Table 28-5](#) through [Table 28-9](#). The table below lists the variables of [Equation 1 on page 406](#).

Table 28-10. Typical conversion time

| Variable | Time |
|----------|------------------------------------|
| SFCAdder | 5 ADCK cycles + 5 bus clock cycles |

Table continues on the next page...

Table 28-10. 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 μ s.

28.5.4.6.2 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 [Equation 1 on page 406](#), and the information provided in [Table 28-5](#) through [Table 28-9](#). The table below lists the variables of [Equation 1 on page 406](#).

Table 28-11. 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 μ s.

28.5.4.7 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] is set.

28.5.5 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 28-12. 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. |

Table continues on the next page...

Table 28-12. Compare modes (continued)

| SC2[AC FGT] | SC2[AC REN] | ADCCV1 relative to ADCCV2 | Function | Compare mode description |
|--------------------|--------------------|----------------------------------|------------------------------|--|
| 0 | 1 | Less than or equal | Outside range, not inclusive | Compare true if the result is less than CV1 Or the result is greater than CV2. |
| 0 | 1 | Greater than | Inside range, not inclusive | Compare true if the result is less than CV1 And 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 And 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 Or the result is less than or equal to CV2. |

With SC2[ACREN] =1, and if the value of CV1 is less than or equal to the value of CV2, then setting SC2[ACFGT] will select a trigger-if-inside-compare-range inclusive-of-endpoints function. Clearing SC2[ACFGT] will select a trigger-if-outside-compare-range, not-inclusive-of-endpoints function.

If CV1 is greater than CV2, setting SC2[ACFGT] will select a trigger-if-outside-compare-range, inclusive-of-endpoints function. Clearing SC2[ACFGT] will select a trigger-if-inside-compare-range, not-inclusive-of-endpoints function.

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.

28.5.6 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 and the plus-side calibration values. The offset calibration value is automatically stored in the ADC offset correction register (OFS), and the plus-side calibration values are automatically stored in the ADC plus-side calibration registers, CLPx. The user must configure the ADC correctly prior to calibration, and must generate the plus-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 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.

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 gain, and plus-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.

Further information on the calibration procedure can be found in the Calibration section of [AN3949: ADC16 Calibration Procedure and Programmable Delay Block Synchronization](#).

28.5.7 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.

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.

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.

28.5.8 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.

$$\text{Temp} = 25 - \left((V_{\text{TEMP}} - V_{\text{TEMP25}}) \div m \right)$$

Equation 2. Approximate transfer function of the temperature sensor

where:

- V_{TEMP} is the voltage of the temperature sensor channel at the ambient temperature.
- V_{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_{TEMP25} and temperature sensor slope values from the ADC Electricals table.

In application code, the user reads the temperature sensor channel, calculates V_{TEMP} , and compares to V_{TEMP25} . If V_{TEMP} is greater than V_{TEMP25} the cold slope value is applied in the preceding equation. If V_{TEMP} is less than V_{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).

28.5.9 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.

28.5.10 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.

28.5.10.1 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.

28.5.10.2 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-specific ADC information for configuration information for this device.

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, Rn, 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.

28.5.11 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.

28.6 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 12-bit, 10-bit, or 8-bit single-ended resolution, single or continuous conversion, and a polled or interrupt approach, among many other options. For information used in this example, refer to [Table 28-8](#), [Table 28-9](#), and [Table 28-10](#).

Note

Hexadecimal values are designated by a preceding 0x, binary values designated by a preceding %, and decimal values have no preceding character.

28.6.1 ADC module initialization example

28.6.1.1 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 enable or disable conversion complete interrupts. Also, select the input channel which can be used to perform conversions.

28.6.1.2 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. |
| 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 1:0 | REFSEL | 00 | Selects default voltage reference pin pair (External pins V_{REFH} and V_{REFL}). |

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 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.

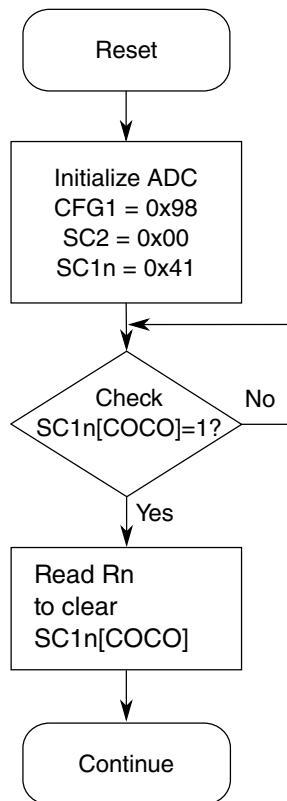


Figure 28-2. Initialization flowchart example

28.7 Application information

The ADC has been designed to be integrated into a microcontroller for use in embedded control applications requiring an ADC.

For guidance on selecting optimum external component values and converter parameters see [AN4373: Cookbook for SAR ADC Measurements](#).

28.7.1 External pins and routing

28.7.1.1 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.

28.7.1.2 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 μ 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.

28.7.1.3 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 μF 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.

28.7.2 Sources of error

28.7.2.1 Sampling error

For proper conversions, the input must be sampled long enough to achieve the proper accuracy.

$$RAS + RADIN = SC / (FMAX * NUMTAU * CADIN)$$

Figure 28-3. 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

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.

28.7.2.2 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.

28.7.2.3 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 μF low-ESR capacitor from V_{REFH} to V_{REFL} .
- There is a 0.1 μF low-ESR capacitor from V_{DDA} to V_{SSA} .
- If inductive isolation is used from the primary supply, an additional 1 μF 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 μF 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.

28.7.2.4 Code width and quantization error

The ADC quantizes the ideal straight-line transfer function into 4096 steps in the 12-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 12, 10, or 8, defined as 1 LSB, is:

$$1\text{LSB} = (V_{REFH}) / 2^N$$

Equation 3. 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.

28.7.2.5 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. 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. 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.

28.7.2.6 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.

Application information

- 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 Chip-specific CMP information

This device includes one high-speed comparator and two 8-input multiplexers 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 5 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 LPUART interfaces. For complete details on the CMP module interconnects, see the [Module-to-Module section](#).

The CMP does not support window compare function and a 0 must always be written to CMP_CR1[WE]. 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).

29.1.1 CMP input connections

The following table shows the fixed internal connections to the CMP0.

Table 29-1. CMP input connections

| CMP inputs | CMP0 |
|------------|----------|
| IN0 | CMP0_IN0 |

Table continues on the next page...

Table 29-1. CMP input connections (continued)

| CMP inputs | CMP0 |
|------------|----------------------|
| IN1 | CMP0_IN1 |
| IN2 | CMP0_IN2 |
| IN3 | CMP0_IN3 |
| IN4 | — |
| IN5 | 1.2V Verf /CMP0_IN5 |
| IN6 | Bandgap |
| IN7 | 6-bit DAC0 reference |

29.1.2 CMP external references

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

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

29.1.3 CMP trigger mode

The CMP and 6-bit DAC sub-block supports trigger mode operation when 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.

29.2 Introduction

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.

29.2.1 CMP features

The CMP has the following features:

- Operational over the entire supply range
- Inputs may range from rail to rail
- Programmable hysteresis control
- 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
 - Digitally filtered:
 - Filter can be bypassed
 - Can be clocked via 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
- Functional in all modes of operation except VLLS0
- The filter functions are not available in the following modes:
 - Stop
 - VLPS
 - VLLSx

29.2.2 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

29.2.3 ANMUX key features

- Two 8-to-1 channel mux
- Operational over the entire supply range

29.2.4 CMP, DAC and ANMUX diagram

The following figure shows the block diagram for the High-Speed Comparator, DAC, and ANMUX modules.

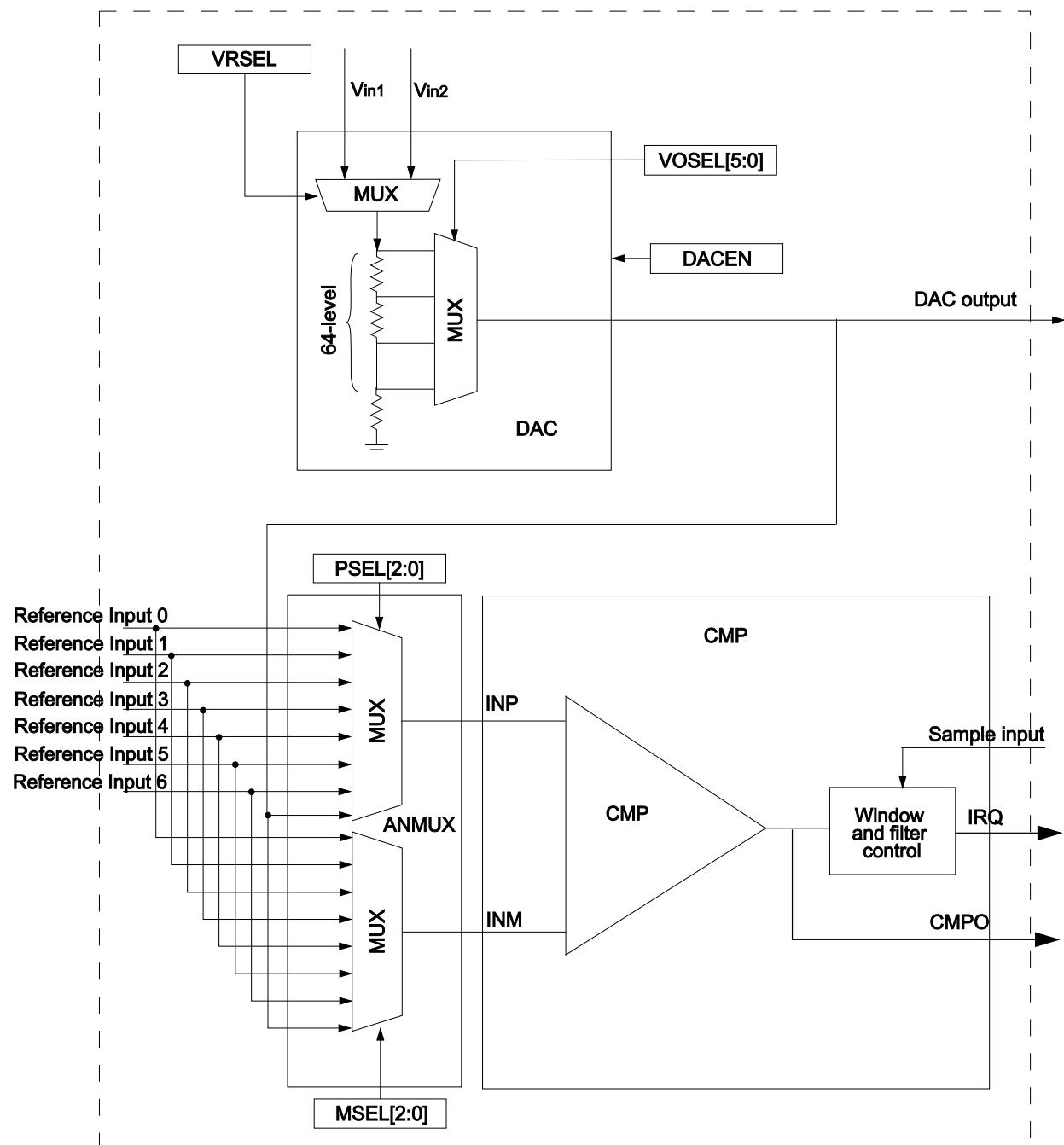


Figure 29-1. CMP, DAC and ANMUX block diagram

29.2.5 CMP block diagram

The following figure shows the block diagram for the CMP module.

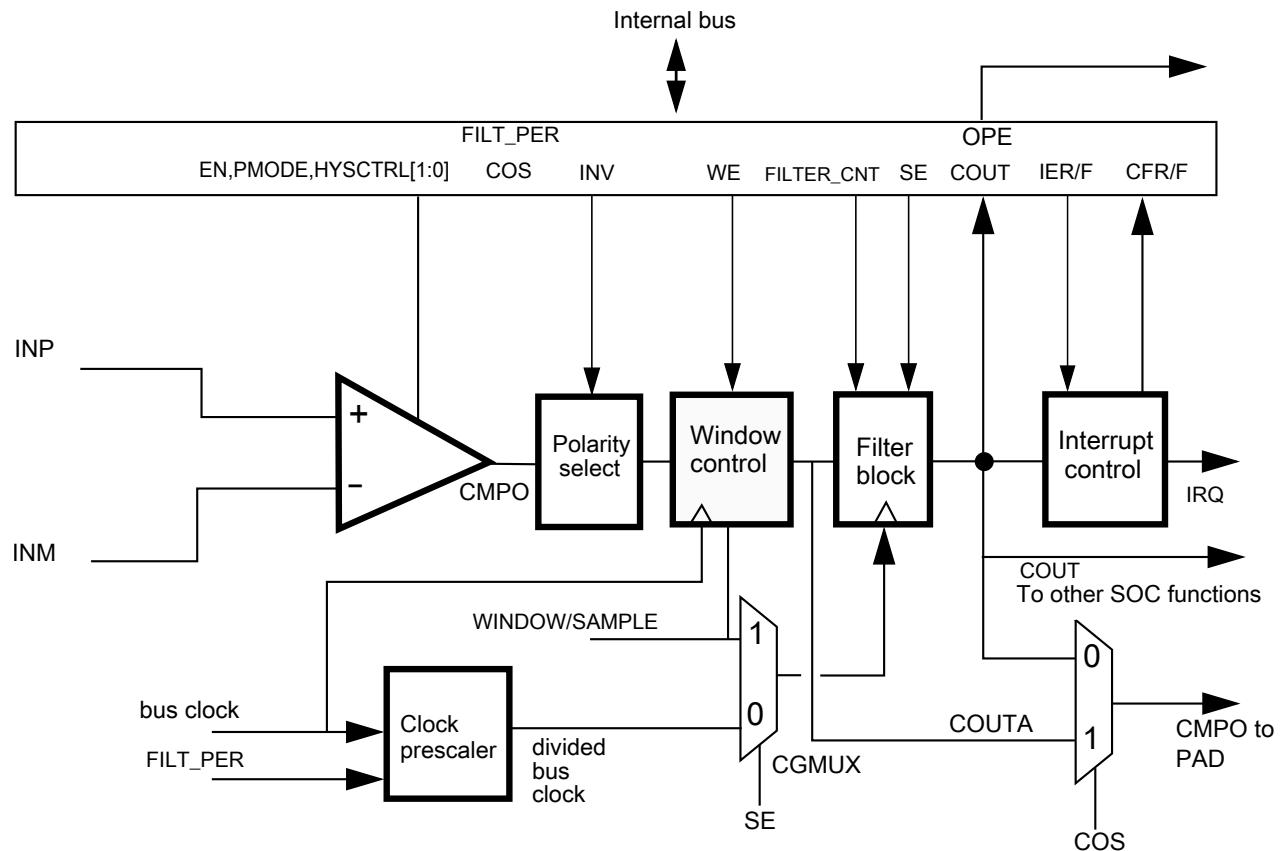


Figure 29-2. Comparator module block diagram

In the CMP block diagram:

- The Window Control block is bypassed when CR1[WE] = 0
- 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.
 - CR1[SE] = 0, the divided bus clock is used as sampling clock
- 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.

29.3 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 | 29.3.1/431 |
| 4007_3001 | CMP Control Register 1 (CMP0_CR1) | 8 | R/W | 00h | 29.3.2/432 |
| 4007_3002 | CMP Filter Period Register (CMP0_FPR) | 8 | R/W | 00h | 29.3.3/433 |
| 4007_3003 | CMP Status and Control Register (CMP0_SCR) | 8 | R/W | 00h | 29.3.4/434 |
| 4007_3004 | DAC Control Register (CMP0_DACCR) | 8 | R/W | 00h | 29.3.5/435 |
| 4007_3005 | MUX Control Register (CMP0_MUXCR) | 8 | R/W | 00h | 29.3.6/435 |

29.3.1 CMP Control Register 0 (CMPx_CR0)

Address: 4007_3000h base + 0h offset = 4007_3000h

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|-------|---|---|------------|---|---|---|---|---------|
| Read | 0 | | FILTER_CNT | | 0 | 0 | | HYSTCTR |
| Write | | | | | | | | |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

CMPx_CR0 field descriptions

| Field | Description |
|-------------------|--|
| 7 Reserved | This field is reserved. This read-only field is reserved and always has the value 0. |
| 6–4 FILTER_CNT | Filter Sample Count 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 Functional description . 000 Filter is disabled. SE = 0, COUT = COUTA. 001 One sample must agree. The comparator output is simply sampled. 010 2 consecutive samples must agree. 011 3 consecutive samples must agree. 100 4 consecutive samples must agree. 101 5 consecutive samples must agree. 110 6 consecutive samples must agree. 111 7 consecutive samples must agree. |
| 3 Reserved | This field is reserved. This read-only field is reserved and always has the value 0. |
| 2 Reserved | This field is reserved. This read-only field is reserved and always has the value 0. |
| HYSTCTR | Comparator hard block hysteresis control 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. 00 Level 0 |

Table continues on the next page...

CMPx_CR0 field descriptions (continued)

| Field | Description |
|-------|--|
| | 01 Level 1 10 Level 2 11 Level 3 |

29.3.2 CMP Control Register 1 (CMPx_CR1)

Address: 4007_3000h base + 1h offset = 4007_3001h

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|-------|----|----|-------|-------|-----|-----|-----|----|
| Read | SE | WE | TRIGM | PMODE | INV | COS | OPE | EN |
| Write | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| Reset | | | | | | | | |

CMPx_CR1 field descriptions

| Field | Description |
|------------|---|
| 7 SE | Sample Enable SE must be clear to 0 and usage of sample operation is limited to a divided version of the bus clock. 0 Sampling mode is not selected. 1 Sampling mode is selected. |
| 6 WE | Windowing Enable The CMP does not support window compare function and a 0 must always be written to WE. 0 Windowing mode is not selected. 1 Windowing mode is selected. |
| 5 TRIGM | Trigger Mode Enable 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. 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 for details about the external timer resource. 0 Trigger mode is disabled. 1 Trigger mode is enabled. |
| 4 PMODE | Power Mode Select See the electrical specifications table in the device Data Sheet for details. 0 Low-Speed (LS) Comparison mode selected. In this mode, CMP has slower output propagation delay and lower current consumption. 1 High-Speed (HS) Comparison mode selected. In this mode, CMP has faster output propagation delay and higher current consumption. |

Table continues on the next page...

CMPx_CR1 field descriptions (continued)

| Field | Description |
|----------|--|
| 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"> 0 Does not invert the comparator output. 1 Inverts the comparator output. |
| 2 COS | <p>Comparator Output Select</p> <ul style="list-style-type: none"> 0 Set the filtered comparator output (CMPO) to equal COUT. 1 Set the unfiltered comparator output (CMPO) to equal COUTA. |
| 1 OPE | <p>Comparator Output Pin Enable</p> <ul style="list-style-type: none"> 0 CMPO is not available on the associated CMPO output pin. If the comparator does not own the pin, this field has no effect. 1 CMPO is available on the associated CMPO output pin. <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"> 0 Analog Comparator is disabled. 1 Analog Comparator is enabled. |

29.3.3 CMP Filter Period Register (CMPx_FPR)

Address: 4007_3000h base + 2h offset = 4007_3002h

| Bit | 7 | 6 | 5 | 4 | | 3 | 2 | 1 | 0 |
|------------|----------|---|---|---|--|---|---|---|---|
| Read Write | FILT_PER | | | | | | | | |
| Reset | 0 | 0 | 0 | 0 | | 0 | 0 | 0 | 0 |

CMPx_FPR field descriptions

| Field | Description |
|----------|---|
| 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 Functional description.</p> |

29.3.4 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 | Reserved | 0 | IER | IEF | CFR | CFF | COUT |
| Write | | | | | w1c | w1c | | |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

CMPx_SCR field descriptions

| Field | Description |
|---------------|---|
| 7 Reserved | This field is reserved. This read-only field is reserved and always has the value 0. |
| 6 Reserved | This field is reserved. This bit must be written as 0. |
| 5 Reserved | This field is reserved. This read-only field is reserved and always has the value 0. |
| 4 IER | Comparator Interrupt Enable Rising Enables the CFR interrupt from the CMP. When this field is set, an interrupt will be asserted when CFR is set. 0 Interrupt is disabled. 1 Interrupt is enabled. |
| 3 IEF | Comparator Interrupt Enable Falling Enables the CFF interrupt from the CMP. When this field is set, an interrupt will be asserted when CFF is set. 0 Interrupt is disabled. 1 Interrupt is enabled. |
| 2 CFR | Analog Comparator Flag Rising 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 . 0 Rising-edge on COUT has not been detected. 1 Rising-edge on COUT has occurred. |
| 1 CFF | Analog Comparator Flag Falling 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 . 0 Falling-edge on COUT has not been detected. 1 Falling-edge on COUT has occurred. |
| 0 COUT | Analog Comparator Output 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. |

29.3.5 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 | <p>DAC Enable</p> <p>Enables the DAC. When the DAC is disabled, it is powered down to conserve power.</p> <p>0 DAC is disabled. 1 DAC is enabled.</p> |
| 6 VRSEL | <p>Supply Voltage Reference Source Select</p> <p>0 V_{in1} is selected as resistor ladder network supply reference. 1 V_{in2} is selected as resistor ladder network supply reference.</p> |
| VOSEL | <p>DAC Output Voltage Select</p> <p>Selects an output voltage from one of 64 distinct levels.</p> <p>$DACO = (V_{in} / 64) * (VOSEL[5:0] + 1)$, so the DACO range is from $V_{in}/64$ to V_{in}.</p> |

29.3.6 MUX Control Register (CMPx_MUXCR)

Address: 4007_3000h base + 5h offset = 4007_3005h

| Bit | 7 | 6 | 5 | 4 | | 3 | 2 | 1 | 0 |
|------------|----------|---|---|------|--|---|------|---|---|
| Read Write | Reserved | 0 | | PSEL | | | MSEL | | |
| Reset | 0 | 0 | 0 | 0 | | 0 | 0 | 0 | 0 |

CMPx_MUXCR field descriptions

| Field | Description |
|------------|---|
| 7 Reserved | <p>Bit can be programmed to zero only .</p> <p>This field is reserved.</p> |
| 6 Reserved | <p>This field is reserved.</p> <p>This read-only field is reserved and always has the value 0.</p> |
| 5–3 PSEL | <p>Plus Input Mux Control</p> <p>Determines which input is selected for the plus input of the comparator. For INx inputs, see CMP, DAC, and ANMUX block diagrams.</p> |

Table continues on the next page...

CMPx_MUXCR field descriptions (continued)

| Field | Description | | | | | | | | | | | | | | | | |
|-------|--|-----|-----|-----|-----|-----|-----|-----|-----|-----|-----|-----|-----|-----|-----|-----|-----|
| | <p>NOTE: 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 | | | | | | | | | | | | | | | | |
| MSEL | <p>Minus Input Mux Control</p> <p>Determines which input is selected for the minus input of the comparator. For INx inputs, see CMP, DAC, and ANMUX block diagrams.</p> <p>NOTE: 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 | | | | | | | | | | | | | | | | |

29.4 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].

29.4.1 CMP functional modes

There are the following main sub-blocks to the CMP module:

- The comparator itself
- The filter function

The filter, CR0[FILTER_CNT], can be clocked from an internal clock source only. 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 comparator filter and sampling features can be combined as shown in the following table. Individual modes are discussed below.

Table 29-2. Comparator sample/filter controls

| Mode # | CR1[EN] | CR1[WE] | CR1[SE] | CR0[FILTER_CNT] | FPR[FILT_PER] | Operation |
|--|---------|---------|---------|-----------------|---------------|--|
| 1 | 0 | X | X | X | X | Disabled See the Disabled mode (# 1). |
| 2A | 1 | 0 | 0 | 0x00 | X | Continuous Mode See the Continuous mode (#s 2A & 2B). |
| 2B | 1 | 0 | 0 | X | 0x00 | |
| 3B | 1 | 0 | 0 | 0x01 | > 0x00 | Sampled, Non-Filtered mode See the Sampled, Non-Filtered mode (#s 3B). |
| 4B | 1 | 0 | 0 | > 0x01 | > 0x00 | Sampled, Filtered mode See the Sampled, Filtered mode (#s 4B). |
| 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.

29.4.1.1 Disabled mode (# 1)

In Disabled mode, the analog comparator is non-functional and consumes no power. CMPO is 0 in this mode.

29.4.1.2 Continuous mode (#s 2A & 2B)

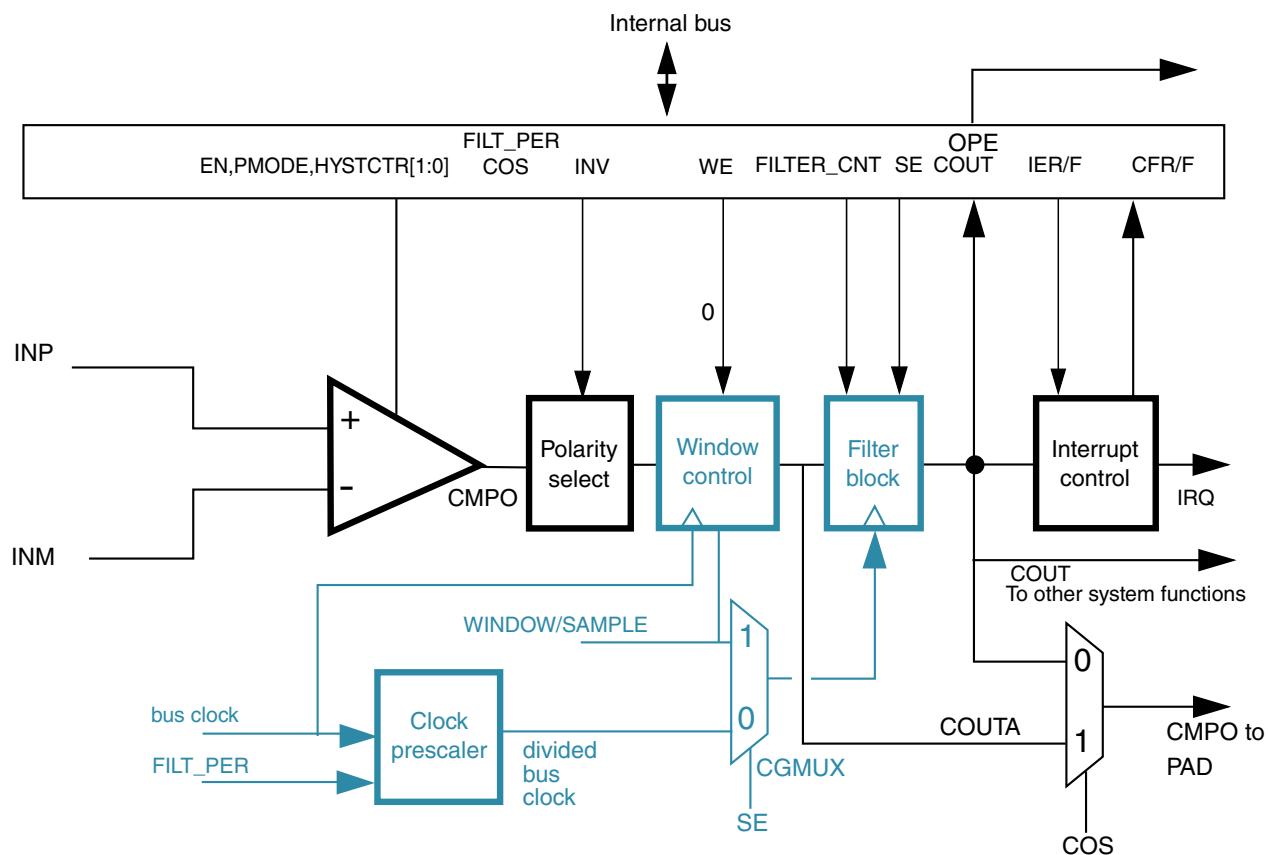


Figure 29-3. Comparator operation in Continuous mode

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.

29.4.1.3 Sampled, Non-Filtered mode (#s 3B)

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.

The comparator filter has no other function than sample/hold of the comparator output in this mode (# 3B).

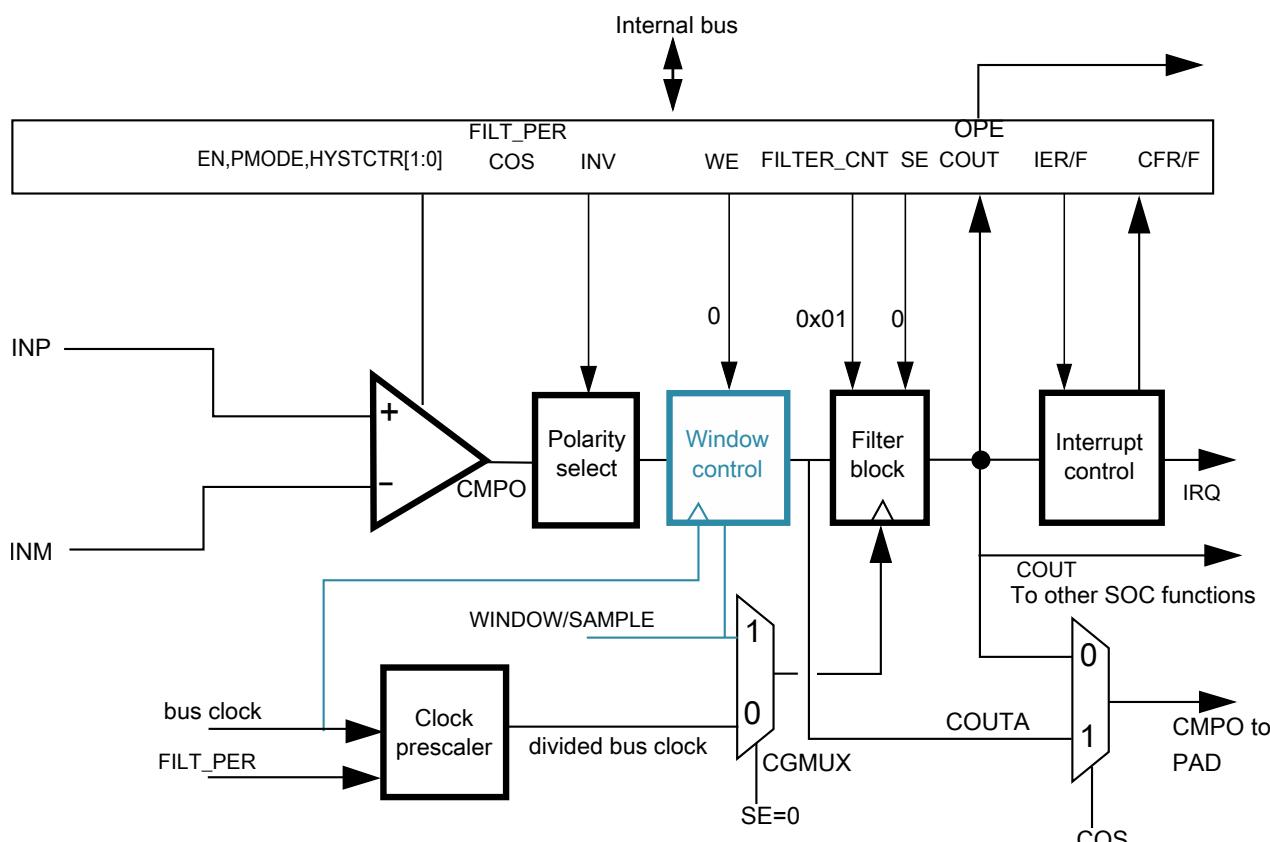


Figure 29-4. Sampled, Non-Filtered (# 3B): sampling interval internally derived

29.4.1.4 Sampled, Filtered mode (#s 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.

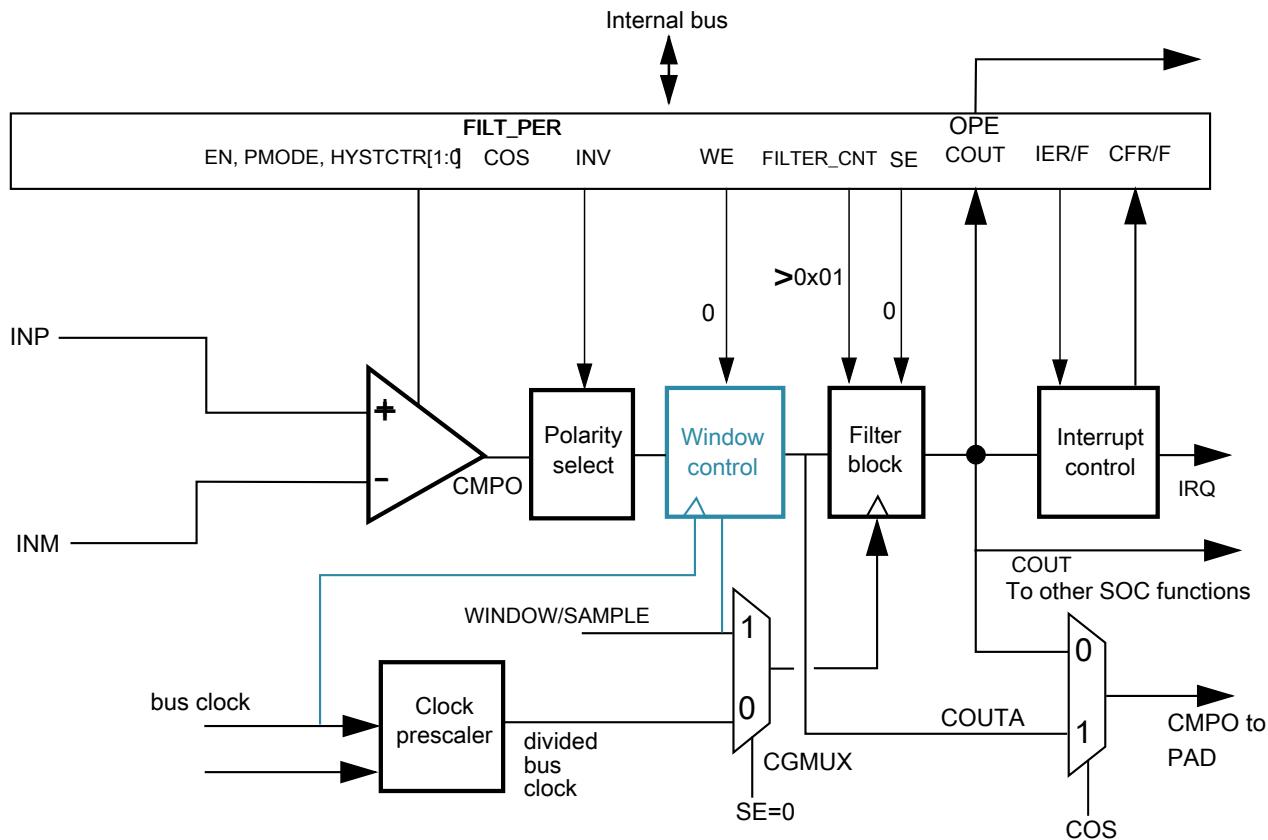


Figure 29-5. Sampled, Filtered (# 4B): sampling point internally derived

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.

29.4.2 Power modes

29.4.2.1 Wait mode operation

During Wait and VLPW modes, the CMP, if enabled, continues to operate normally and a CMP interrupt can wake the MCU.

29.4.2.2 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.

29.4.2.3 Background Debug Mode Operation

When the microcontroller is in active background debug mode, the CMP continues to operate normally.

29.4.3 Startup and operation

A typical startup sequence is listed here.

- 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 and filter. See the Data Sheets for power-on delays of the comparators. 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.

29.4.4 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 modes. Filtering can be performed using an internal timebase defined by FPR[FILT_PER] 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.

29.4.4.1 Enabling filter modes

Filter modes can be enabled by:

- Setting CR0[FILTER_CNT] > 0x01 and
- Setting FPR[FILT_PER] to a nonzero value

Using the divided bus clock to drive the filter, it will take samples of COUTA every FPR[FILT_PER] bus clock cycles.

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 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.

29.4.4.2 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 29-3. Comparator sample/filter maximum latencies

| Mode # | CR1[EN] | CR1[WE] | CR1[SE] | CR0[FILTER_CNT] | FPR[FILT_PER] | Operation | Maximum latency ¹ |
|--------|---------|---------|---------|-----------------|---------------|----------------------------|--|
| 1 | 0 | X | X | X | X | Disabled | N/A |
| 2A | 1 | 0 | 0 | 0x00 | X | Continuous Mode | T_{PD} |
| 2B | 1 | 0 | 0 | X | 0x00 | | |
| 3B | 1 | 0 | 0 | 0x01 | > 0x00 | Sampled, Non-Filtered mode | $T_{PD} + (FPR[FILT_PER] * T_{per}) + T_{per}$ |
| 4B | 1 | 0 | 0 | > 0x01 | > 0x00 | Sampled, Filtered mode | $T_{PD} + (CR0[FILTER_CNT] * FPR[FILT_PER] * T_{per}) + T_{per}$ |

1. T_{PD} represents the intrinsic delay of the analog component plus the polarity select logic. T_{per} is the period of the bus clock.

29.5 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 |

29.6 Digital-to-analog converter

The figure found here 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.

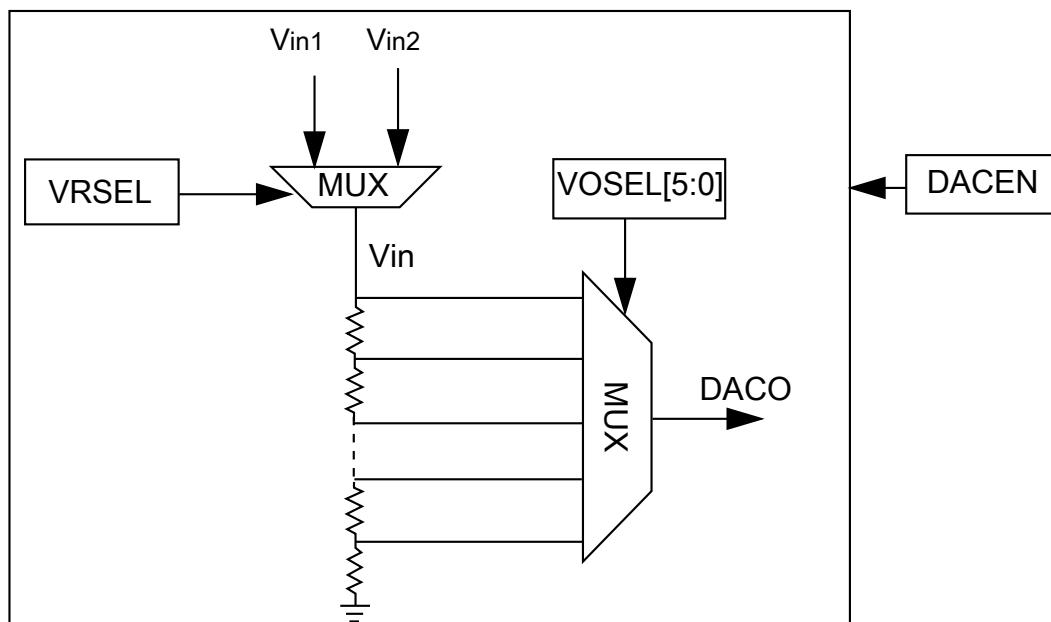


Figure 29-6. 6-bit DAC block diagram

29.7 DAC functional description

This section provides DAC functional description information.

29.7.1 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

29.8 DAC resets

This module has a single reset input, corresponding to the chip-wide peripheral reset.

29.9 DAC clocks

This module has a single clock input, the bus clock.

29.10 DAC interrupts

This module has no interrupts.

29.11 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.

Chapter 30

Voltage Reference (VREF)

30.1 Chip specific VREF information

This device includes a voltage reference (VREF) to supply an accurate 1.2 V voltage output.

The voltage reference can provide a reference voltage to external peripherals or a reference to analog peripherals, such as the ADC.

For either an internal or external reference if the VREF_OUT functionality is being used, VREF_OUT signal must be connected to an output load capacitor.

Refer the block guide for more details.

30.1.1 Clock Gating

The clock to the VREF module can be turned on or off using the SCGC4[VREF] 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 SCGC4[VREF] 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.

30.2 Introduction

The Voltage Reference (VREF) is intended to supply an accurate voltage output that can be trimmed in 0.5 mV steps. The VREF can be used in applications to provide a reference voltage to external devices or used internally as a reference to analog peripherals such as the ADC, or CMP. The voltage reference has three operating modes that provide different levels of supply rejection and power consumption.

The following figure is a block diagram of the Voltage Reference.

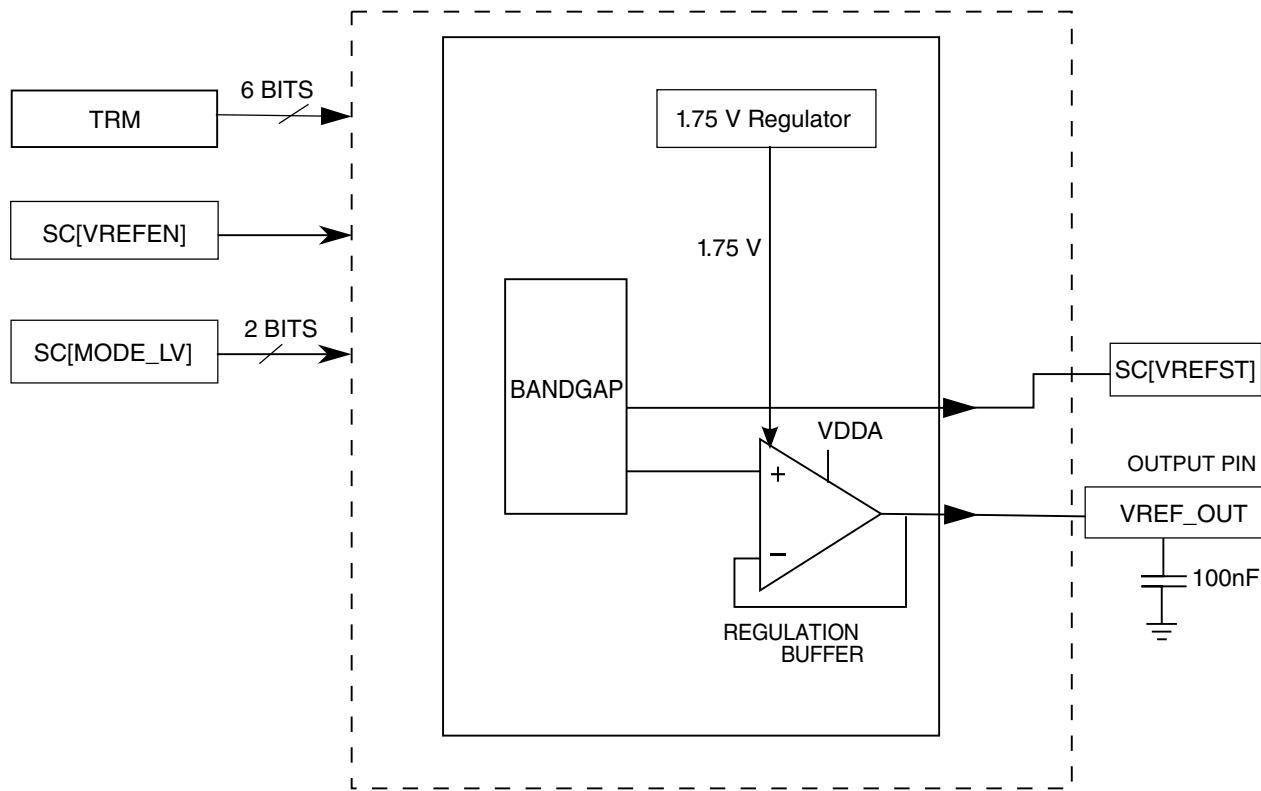


Figure 30-1. Voltage reference block diagram

30.2.1 Overview

The Voltage Reference provides a buffered reference voltage for use as an external reference. In addition, the buffered reference is available internally for use with on chip peripherals such as ADCs and DACs. Refer to the chip configuration details for a description of these options. The reference voltage signal is output when the VREF is enabled. The Voltage Reference output can be trimmed with a resolution of 0.5mV by means of the TRM register TRIM[5:0] bitfield.

30.2.2 Features

The Voltage Reference has the following features:

- Programmable trim register with 0.5 mV steps, automatically loaded with factory trimmed value upon reset
- Programmable buffer mode selection:
 - Off

- Bandgap enabled/standby (output buffer disabled)
- Low power buffer mode (output buffer enabled)
- High power buffer mode (output buffer enabled)
- 1.2 V output at room temperature
- VREF_OUT output signal

30.2.3 Modes of Operation

The Voltage Reference continues normal operation in Run, Wait, and Stop modes. The Voltage Reference can also run in Very Low Power Run (VLPR), Very Low Power Wait (VLPW) and Very Low Power Stop (VLPS). If it is desired to use the VREF regulator and/or the chop oscillator in the very low power modes, the system reference voltage (also referred to as the bandgap voltage reference) must be enabled in these modes. Refer to the chip configuration details for information on enabling this mode of operation. Having the VREF regulator enabled does increase current consumption. In very low power modes it may be desirable to disable the VREF regulator to minimize current consumption. Note however that the accuracy of the output voltage will be reduced (by as much as several mVs) when the VREF regulator is not used.

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.

30.2.4 VREF Signal Descriptions

The following table shows the Voltage Reference signals properties.

Table 30-1. VREF Signal Descriptions

| Signal | Description | I/O |
|----------|---|-----|
| VREF_OUT | Internally-generated Voltage Reference output | O |

NOTE

When the VREF output buffer is disabled, the status of the VREF_OUT signal is high-impedance.

30.3 Memory Map and Register Definition

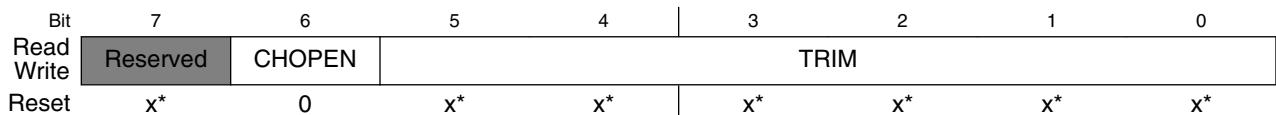
VREF memory map

| Absolute address (hex) | Register name | Width (in bits) | Access | Reset value | Section/ page |
|------------------------|--|-----------------|--------|-----------------------------|---------------|
| 4007_4000 | VREF Trim Register (VREF_TRM) | 8 | R/W | See section | 30.3.1/450 |
| 4007_4001 | VREF Status and Control Register (VREF_SC) | 8 | R/W | 00h | 30.3.2/451 |

30.3.1 VREF Trim Register (VREF_TRM)

This register contains bits that contain the trim data for the Voltage Reference.

Address: 4007_4000h base + 0h offset = 4007_4000h



* Notes:

- x = Undefined at reset.

VREF_TRM field descriptions

| Field | Description |
|---------------|---|
| 7 Reserved | This field is reserved. Upon reset this value is loaded with a factory trim value. |
| 6 CHOPEN | Chop oscillator enable. When set, internal chopping operation is enabled and the internal analog offset will be minimized. This bit is set during factory trimming of the VREF voltage. This bit should be written to 1 to achieve the performance stated in the data sheet. If the internal voltage regulator is being used (REGEN bit is set to 1), the chop oscillator must also be enabled. If the chop oscillator is to be used in very low power modes, the system (bandgap) voltage reference must also be enabled. See the chip-specific VREF information (also known as "chip configuration" details) for a description of how this can be achieved. 0 Chop oscillator is disabled. 1 Chop oscillator is enabled. |
| TRIM | Trim bits These bits change the resulting VREF by approximately ± 0.5 mV for each step. NOTE: Min = minimum and max = maximum voltage reference output. For minimum and maximum voltage reference output values, refer to the Data Sheet for this chip. |

Table continues on the next page...

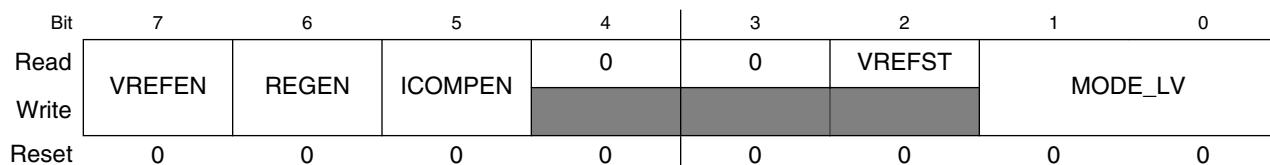
VREF_TRM field descriptions (continued)

| Field | Description | |
|-------|---------------------------------------|--|
| | 000000 Min 111111 Max | |

30.3.2 VREF Status and Control Register (VREF_SC)

This register contains the control bits used to enable the internal voltage reference and to select the buffer mode to be used.

Address: 4007_4000h base + 1h offset = 4007_4001h

**VREF_SC field descriptions**

| Field | Description |
|--------------|---|
| 7 VREFEN | <p>Internal Voltage Reference enable</p> <p>This bit is used to enable the bandgap reference within the Voltage Reference module.</p> <p>NOTE: After the VREF is enabled, turning off the clock to the VREF module via the corresponding clock gate register will not disable the VREF. VREF must be disabled via this VREFEN bit.</p> <p>0 The module is disabled. 1 The module is enabled.</p> |
| 6 REGEN | <p>Regulator enable</p> <p>This bit is used to enable the internal 1.75 V regulator to produce a constant internal voltage supply in order to reduce the sensitivity to external supply noise and variation. If it is desired to keep the regulator enabled in very low power modes, refer to the Chip Configuration details for a description on how this can be achieved.</p> <p>This bit should be written to 1 to achieve the performance stated in the data sheet.</p> <p>NOTE: See section "Internal voltage regulator" for details on the required sequence to enable the internal regulator.</p> <p>0 Internal 1.75 V regulator is disabled. 1 Internal 1.75 V regulator is enabled.</p> |
| 5 ICOMPEN | <p>Second order curvature compensation enable</p> <p>This bit should be written to 1 to achieve the performance stated in the data sheet.</p> |

Table continues on the next page...

VREF_SC field descriptions (continued)

| Field | Description |
|------------|--|
| | 0 Disabled 1 Enabled |
| 4 Reserved | This field is reserved. This read-only field is reserved and always has the value 0. |
| 3 Reserved | This field is reserved. This read-only field is reserved and always has the value 0. |
| 2 VREFST | Internal Voltage Reference stable This bit indicates that the bandgap reference within the Voltage Reference module has completed its startup and stabilization. NOTE: This bit is valid only when the chop oscillator is not being used. 0 The module is disabled or not stable. 1 The module is stable. |
| MODE_LV | Buffer Mode selection These bits select the buffer modes for the Voltage Reference module. 00 Bandgap on only, for stabilization and startup 01 High power buffer mode enabled 10 Low-power buffer mode enabled 11 Reserved |

30.4 Functional Description

The Voltage Reference is a bandgap buffer system. Unity gain amplifiers are used.

The VREF_OUT signal can be used by both internal and external peripherals in low and high power buffer mode. A 100 nF capacitor must always be connected between VREF_OUT and VSSA if the VREF is being used. This capacitor must be as close to VREF_OUT pin as possible.

The following table shows all possible function configurations of the Voltage Reference.

Table 30-2. Voltage Reference function configurations

| SC[VREFEN] | SC[MODE_LV] | Configuration | Functionality |
|------------|-------------|--|---|
| 0 | X | Voltage Reference disabled | Off |
| 1 | 00 | Voltage Reference enabled, bandgap on only | Startup and standby |
| 1 | 01 | Voltage Reference enabled, high-power buffer on | VREF_OUT available for internal and external use. 100 nF capacitor is required. |

Table continues on the next page...

Table 30-2. Voltage Reference function configurations (continued)

| SC[VREFEN] | SC[MODE_LV] | Configuration | Functionality |
|------------|-------------|---|---|
| 1 | 10 | Voltage Reference enabled, low power buffer on | VREF_OUT available for internal and external use. 100 nF capacitor is required. |
| 1 | 11 | Reserved | Reserved |

30.4.1 Voltage Reference Disabled, SC[VREFEN] = 0

When SC[VREFEN] = 0, the Voltage Reference is disabled, the VREF bandgap and the output buffers are disabled. The Voltage Reference is in off mode.

30.4.2 Voltage Reference Enabled, SC[VREFEN] = 1

When SC[VREFEN] = 1, the Voltage Reference is enabled, and different modes should be set by the SC[MODE_LV] bits.

30.4.2.1 SC[MODE_LV]=00

The internal VREF bandgap is enabled to generate an accurate 1.2 V output that can be trimmed with the TRM register's TRIM[5:0] bitfield. The bandgap requires some time for startup and stabilization. SC[VREFST] can be monitored to determine if the stabilization and startup is complete when the chop oscillator is not enabled.

If the chop oscillator is being used, the internal bandgap reference voltage settles within the chop oscillator start up time, Tchop_osc_stup.

The output buffer is disabled in this mode, and there is no buffered voltage output. The Voltage Reference is in standby mode. If this mode is first selected and the low power or high power buffer mode is subsequently enabled, there will be a delay before the buffer output is settled at the final value. This is the buffer start up delay (Tstup) and the value is specified in the appropriate device data sheet.

30.4.2.2 SC[MODE_LV] = 01

The internal VREF bandgap is on. The high power buffer is enabled to generate a buffered 1.2 V voltage to VREF_OUT. It can also be used as a reference to internal analog peripherals such as an ADC channel or analog comparator input.

If this mode is entered from the standby mode ($SC[MODE_LV] = 00$, $SC[VREFEN] = 1$) there will be a delay before the buffer output is settled at the final value. This is the buffer start up delay (T_{stup}) and the value is specified in the appropriate device data sheet. If this mode is entered when the VREF module is enabled then you must wait the longer of T_{stup} or until $SC[VREFST] = 1$ when the chop oscillator is not enabled. If the chop oscillator is being used, you must wait the time specified by $T_{chop_osc_stup}$ (chop oscillator start up time) to ensure the VREF output has stabilized.

In this mode, a 100 nF capacitor is required to connect between the VREF_OUT pin and VSSA.

30.4.2.3 SC[MODE_LV] = 10

The internal VREF bandgap is on. The low power buffer is enabled to generate a buffered 1.2 V voltage to VREF_OUT. It can also be used as a reference to internal analog peripherals such as an ADC channel or analog comparator input.

If this mode is entered from the standby mode ($SC[MODE_LV] = 00$, $SC[VREFEN] = 1$) there will be a delay before the buffer output is settled at the final value. This is the buffer start up delay (T_{stup}) and the value is specified in the appropriate device data sheet. If this mode is entered when the VREF module is enabled then you must wait the longer of T_{stup} or until $SC[VREFST] = 1$ when the chop oscillator is not enabled. If the chop oscillator is being used, you must wait the time specified by $T_{chop_osc_stup}$ (chop oscillator start up time) to ensure the VREF output has stabilized.

In this mode, a 100 nF capacitor is required to connect between the VREF_OUT pin and VSSA.

30.4.2.4 SC[MODE_LV] = 11

Reserved

30.4.3 Internal voltage regulator

The VREF module contains an internal voltage regulator that can be enabled to provide additional supply noise rejection. It is recommended that when possible, this regulator be enabled to provide the optimum VREF performance.

If the internal voltage regulator is being used, the chop oscillator must also be enabled. A specific sequence must be followed when enabling the internal regulator as follows:

1. Enable the chop oscillator (VREF_TRM[CHOPEN] = 1)
2. Configure the VREF_SC register to the desired settings with the internal regulator disabled, VREF_SC[REGEN] = 0
3. Wait > 300ns
4. Enable the internal regulator by setting VREF_SC[REGEN] to 1

30.5 Initialization/Application Information

The Voltage Reference requires some time for startup and stabilization. After SC[VREFEN] = 1, SC[VREFST] can be monitored to determine if the stabilization and startup is completed when the chop oscillator is not enabled. When the chop oscillator is enabled, the settling time of the internal bandgap reference is defined by Tchop_osc_stup (chop oscillator start up time). You must wait this time (Tchop_osc_stup) after the internal bandgap has been enabled to ensure the VREF internal reference voltage has stabilized.

When the Voltage Reference is already enabled and stabilized, changing SC[MODE_LV] will not clear SC[VREFST] but there will be some startup time before the output voltage at the VREF_OUT pin has settled. This is the buffer start up delay (Tstup) and the value is specified in the appropriate device data sheet. Also, there will be some settling time when a step change of the load current is applied to the VREF_OUT pin. When the 1.75V VREF regulator is disabled, the VREF_OUT voltage will be more sensitive to supply voltage variation. It is recommended to use this regulator to achieve optimum VREF_OUT performance.

The TRM[CHOPEN], SC[REGEN] and SC[ICOMPEN] bits must be written to 1 to achieve the performance stated in the device data sheet.

NOTE

See section "Internal voltage regulator" for details on the required sequence to enable the internal regulator.

Chapter 31

Timer/PWM Module (TPM)

31.1 Chip-specific TPM information

This device contains two low power TPM modules (TPM). All TPM modules in the device are configured only as basic TPM function, do not support 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 31-1. TPM configuration

| TPM instance | Number of channels | Features/usage |
|--------------|--------------------|---|
| TPMO | 2 | Basic TPM, functional in Stop/VLPS mode |
| TPM1 | 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](#).

31.1.1 Clock options

The TPM blocks are clocked from a single TPM clock that can be selected from OSCERCLK, MCGIRCLK, or MCGPCLK. The selected source is controlled by SIM_SOPT2[TPMSRC].

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.

31.1.2 Trigger options

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

Table 31-2. TPM external trigger options

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

31.1.3 Global timebase

Each TPM has a global timebase feature controlled by TPMx_CONF[GTBEEN]. TPM1 is configured as the global time when this option is enabled.

31.1.4 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.

31.2 Introduction

The TPM (Timer/PWM Module) is a 2- to 8-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. An example of using the TPM with the asynchronous DMA is described in [AN4631:Using the Asynchronous DMA features of the Kinetis L Series](#).

31.2.1 TPM Philosophy

The TPM is built upon a very simple timer (HCS08 Timer PWM Module – TPM) used for many years on NXP'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.

31.2.2 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 2 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 per channel
- Support the generation of an interrupt 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

31.2.3 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.

31.2.4 Block diagram

The TPM uses one input/output (I/O) pin per channel, CH_n (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.

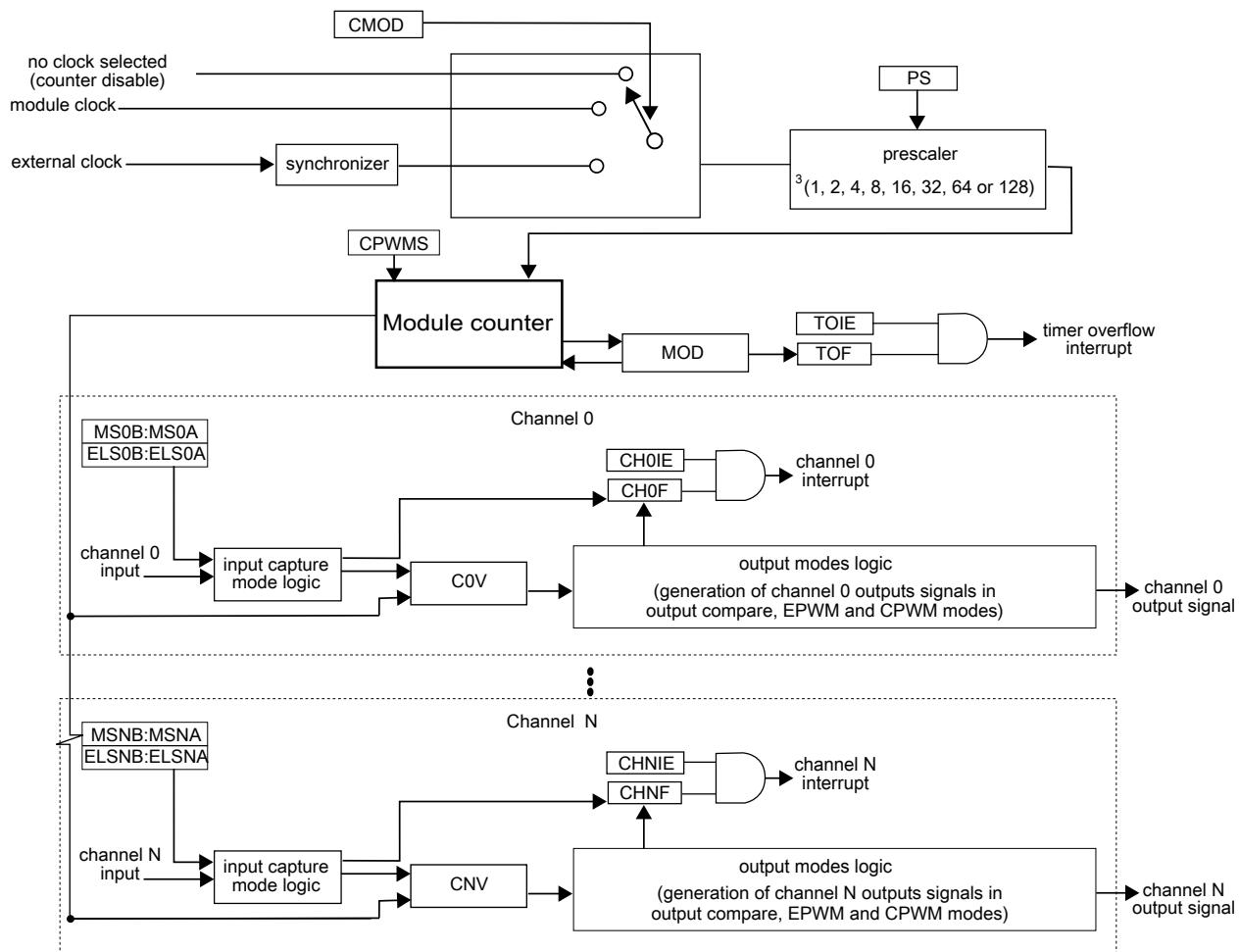


Figure 31-1. TPM block diagram

31.3 TPM Signal Descriptions

Table 31-3 shows the user-accessible signals for the TPM.

Table 31-3. 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 = 1 to 0). A TPM channel pin is configured as output when configured in an output compare or PWM mode and the TPM counter is enabled, otherwise the TPM channel pin is an input. | I/O |

31.3.1 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.

31.3.2 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.

31.4 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 | 31.4.1/463 |
| 4003_8004 | Counter (TPM0_CNT) | 32 | R/W | 0000_0000h | 31.4.2/464 |
| 4003_8008 | Modulo (TPM0_MOD) | 32 | R/W | 0000_FFFFh | 31.4.3/465 |
| 4003_800C | Channel (n) Status and Control (TPM0_C0SC) | 32 | R/W | 0000_0000h | 31.4.4/466 |
| 4003_8010 | Channel (n) Value (TPM0_C0V) | 32 | R/W | 0000_0000h | 31.4.5/467 |
| 4003_8014 | Channel (n) Status and Control (TPM0_C1SC) | 32 | R/W | 0000_0000h | 31.4.4/466 |
| 4003_8018 | Channel (n) Value (TPM0_C1V) | 32 | R/W | 0000_0000h | 31.4.5/467 |
| 4003_8050 | Capture and Compare Status (TPM0_STATUS) | 32 | R/W | 0000_0000h | 31.4.6/468 |
| 4003_8084 | Configuration (TPM0_CONF) | 32 | R/W | 0000_0000h | 31.4.7/470 |
| 4003_9000 | Status and Control (TPM1_SC) | 32 | R/W | 0000_0000h | 31.4.1/463 |
| 4003_9004 | Counter (TPM1_CNT) | 32 | R/W | 0000_0000h | 31.4.2/464 |
| 4003_9008 | Modulo (TPM1_MOD) | 32 | R/W | 0000_FFFFh | 31.4.3/465 |
| 4003_900C | Channel (n) Status and Control (TPM1_C0SC) | 32 | R/W | 0000_0000h | 31.4.4/466 |
| 4003_9010 | Channel (n) Value (TPM1_C0V) | 32 | R/W | 0000_0000h | 31.4.5/467 |
| 4003_9014 | Channel (n) Status and Control (TPM1_C1SC) | 32 | R/W | 0000_0000h | 31.4.4/466 |
| 4003_9018 | Channel (n) Value (TPM1_C1V) | 32 | R/W | 0000_0000h | 31.4.5/467 |

Table continues on the next page...

TPM memory map (continued)

| Absolute address (hex) | Register name | Width (in bits) | Access | Reset value | Section/page |
|------------------------|--|-----------------|--------|-------------|----------------------------|
| 4003_9050 | Capture and Compare Status (TPM1_STATUS) | 32 | R/W | 0000_0000h | 31.4.6/468 |
| 4003_9084 | Configuration (TPM1_CONF) | 32 | R/W | 0000_0000h | 31.4.7/470 |

31.4.1 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 | | | | | | | | | TOIE | | | | | | |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | w1c | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| | R | | | | | | | | | | CPWMS | | | | | |
| | W | | | | | | | | | | CMOD | | | | | |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | | | | | | | | |

TPMx_SC field descriptions

| Field | Description |
|------------------|--|
| 31–8 Reserved | This field is reserved. This read-only field is reserved and always has the value 0. |
| 7 TOF | Timer Overflow Flag Set by hardware when the TPM counter equals the value in the MOD register and increments. Writing a 1 to TOF clears it. Writing a 0 to TOF has no effect. |

Table continues on the next page...

TPMx_SC field descriptions (continued)

| Field | Description |
|-------------|---|
| | If another TPM 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. 0 TPM counter has not overflowed. 1 TPM counter has overflowed. |
| 6 TOIE | Timer Overflow Interrupt Enable Enables TPM overflow interrupts. 0 Disable TOF interrupts. Use software polling. 1 Enable TOF interrupts. An interrupt is generated when TOF equals one. |
| 5 CPWMS | Center-Aligned PWM Select Selects CPWM mode. This mode configures the TPM to operate in up-down counting mode. This field is write protected. It can be written only when the counter is disabled. 0 TPM counter operates in up counting mode. 1 TPM counter operates in up-down counting mode. |
| 4–3 CMOD | Clock Mode Selection Selects the TPM counter clock modes. When disabling the counter, this field remain set until acknowledged in the TPM clock domain. 00 TPM counter is disabled 01 TPM counter increments on every TPM counter clock 10 TPM counter increments on rising edge of TPM_EXTCLK synchronized to the TPM counter clock 11 Reserved |
| PS | Prescale Factor Selection Selects one of 8 division factors for the clock mode selected by CMOD. This field is write protected. It can be written only when the counter is disabled. 000 Divide by 1 001 Divide by 2 010 Divide by 4 011 Divide by 8 100 Divide by 16 101 Divide by 32 110 Divide by 64 111 Divide by 128 |

31.4.2 Counter (TPMx_CNT)

The CNT register contains the TPM counter value.

Reset clears the CNT register. Writing any value to COUNT also clears the counter.

When debug is active, the TPM counter does not increment unless configured otherwise.

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 | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| 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 | |

TPMx_CNT field descriptions

| Field | Description |
|-------------------|---|
| 31–16 Reserved | This field is reserved. This read-only field is reserved and always has the value 0. |
| COUNT | Counter value |

31.4.3 Modulo (TPMx_MOD)

The Modulo register contains the modulo value for the TPM counter. When the TPM counter reaches the modulo value and increments, the overflow flag (TOF) is set and the next value of TPM 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](#). Additional writes to the MOD write buffer are ignored until the register has been updated.

It is recommended to initialize the TPM 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 | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| 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 | 1 | |

TPMx_MOD field descriptions

| Field | Description |
|-------------------|--|
| 31–16 Reserved | This field is reserved. This read-only field is reserved and always has the value 0. |
| MOD | Modulo value When writing this field, all bytes must be written at the same time. |

31.4.4 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 TPM counter clock domain.

Table 31-4. Mode, Edge, and Level Selection

| CPWMS | MSnB:MSnA | ELSnB:ELSnA | Mode | Configuration |
|-------|-----------|-------------|--------------------|---|
| X | 00 | 00 | None | Channel disabled |
| X | 01 | 00 | Software compare | Pin not used for TPM |
| 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 |
| | 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 |
| | | 01 | | Pulse Output high on match |
| | 10 | 10 | Center-aligned PWM | High-true pulses (clear Output on match-up, set Output on match-down) |
| | | 01 | | Low-true pulses (set Output on match-up, clear Output on match-down) |

Address: Base address + Ch offset + (8d × 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 | | | | | | | CHF | CHIE | MSB | MSA | ELSB | ELSA | 0 | | |
| W | | | | | | | | w1c | 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 |

TPMx_CnSC field descriptions

| Field | Description |
|------------------|---|
| 31–8 Reserved | This field is reserved. This read-only field is reserved and always has the value 0. |
| 7 CHF | Channel Flag 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. 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. 0 No channel event has occurred. 1 A channel event has occurred. |
| 6 CHIE | Channel Interrupt Enable Enables channel interrupts. 0 Disable channel interrupts. 1 Enable channel interrupts. |
| 5 MSB | Channel Mode Select Used for further selections in the channel logic. Its functionality is dependent on the channel mode. When a channel is disabled, this field will not change state until acknowledged in the TPM counter clock domain. |
| 4 MSA | Channel Mode Select Used for further selections in the channel logic. Its functionality is dependent on the channel mode. When a channel is disabled, this field will not change state until acknowledged in the TPM counter clock domain. |
| 3 ELSB | Edge or Level Select The functionality of ELSB and ELSA depends on the channel mode. When a channel is disabled, this field will not change state until acknowledged in the TPM counter clock domain. |
| 2 ELSA | Edge or Level Select The functionality of ELSB and ELSA depends on the channel mode. When a channel is disabled, this field will not change state until acknowledged in the TPM counter clock domain. |
| Reserved | This field is reserved. This read-only field is reserved and always has the value 0. |

31.4.5 Channel (n) Value (TPMx_CnV)

These registers contain the captured TPM 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.

Memory Map and Register Definition

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](#). Additional writes to the CnV write buffer are ignored until the register has been updated.

Address: Base address + 10h offset + (8d × 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 | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| 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 | |

TPMx_CnV field descriptions

| Field | Description |
|-------------------|--|
| 31–16 Reserved | This field is reserved. This read-only field is reserved and always has the value 0. |
| VAL | Channel Value Captured TPM 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. |

31.4.6 Capture and Compare Status (TPMx_STATUS)

The STATUS register contains a copy of the status flag, CnSC[CHnF] for each TPM channel, as well as SC[TOF], for software convenience.

Each CHnF bit in STATUS is a mirror of CHnF bit in CnSC. All CHnF bits can be checked using only one read of STATUS. All CHnF bits can be cleared by writing all ones to STATUS.

Hardware sets the individual channel flags when an event occurs on the channel. Writing a 1 to CHF clears it. Writing a 0 to CHF has no effect.

If another event occurs between the flag setting 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 clearing sequence for a previous CHF.

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 | | CH1F | CH0F |
| W | | | | | | | | 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 Reserved | This field is reserved. This read-only field is reserved and always has the value 0. |
| 8 TOF | Timer Overflow Flag See register description 0 TPM counter has not overflowed. 1 TPM counter has overflowed. |
| 7–2 Reserved | This field is reserved. This read-only field is reserved and always has the value 0. |
| 1 CH1F | Channel 1 Flag See the register description. 0 No channel event has occurred. 1 A channel event has occurred. |
| 0 CH0F | Channel 0 Flag See the register description. 0 No channel event has occurred. 1 A channel event has occurred. |

31.4.7 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 | | | | | | | | | | | | | | | |
| 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 |

TPMx_CONF field descriptions

| Field | Description |
|-------------------|--|
| 31–28 Reserved | This field is reserved. This read-only field is reserved and always has the value 0. |
| 27–24 TRGSEL | Trigger Select Selects the input trigger to use for starting the counter and/or reloading the counter. This field should only be changed when the TPM counter is disabled. See Chip configuration section for available options. |
| 23–19 Reserved | This field is reserved. This read-only field is reserved and always has the value 0. |
| 18 CROT | Counter Reload On Trigger When set, the TPM counter will reload with 0 (and initialize PWM outputs to their default value) when a rising edge is detected on the selected trigger input. The trigger input is ignored if the TPM counter is paused during debug mode or doze mode. This field should only be changed when the TPM counter is disabled. 0 Counter is not reloaded due to a rising edge on the selected input trigger 1 Counter is reloaded when a rising edge is detected on the selected input trigger |
| 17 CSOO | Counter Stop On Overflow When set, the TPM counter will stop incrementing once the counter equals the MOD value and incremented (this also sets the TOF). Reloading the counter with 0 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. |

Table continues on the next page...

TPMx_CONF field descriptions (continued)

| Field | Description |
|----------------|---|
| | <p>This field should only be changed when the TPM counter is disabled.</p> <ul style="list-style-type: none"> 0 TPM counter continues incrementing or decrementing after overflow 1 TPM counter stops incrementing or decrementing after overflow. |
| 16 CSOT | <p>Counter Start on Trigger</p> <p>When set, the TPM counter will not start incrementing after it is enabled until a rising edge on the selected trigger input is detected. If the TPM counter is stopped due to an overflow, a rising edge on the selected trigger input will also cause the TPM counter to start incrementing again.</p> <p>The trigger input is ignored if the TPM counter is paused during debug mode or doze mode. This field should only be changed when the TPM counter is disabled.</p> <ul style="list-style-type: none"> 0 TPM counter starts to increment immediately, once it is enabled. 1 TPM 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. |
| 15–10 Reserved | <p>This field is reserved.</p> <p>This read-only field is reserved and always has the value 0.</p> |
| 9 GTBEEN | <p>Global time base enable</p> <p>Configures the TPM to use an externally generated global time base counter. When an externally generated timebase is used, the internal TPM counter is not used by the channels but can be used to generate a periodic interrupt using the Modulo register and timer overflow flag.</p> <ul style="list-style-type: none"> 0 All channels use the internally generated TPM counter as their timebase 1 All channels use an externally generated global timebase as their timebase |
| 8 Reserved | <p>This field is reserved.</p> <p>This read-only field is reserved and always has the value 0.</p> |
| 7–6 DBGMODE | <p>Debug Mode</p> <p>Configures the TPM behavior in debug mode. All other configurations are reserved.</p> <ul style="list-style-type: none"> 00 TPM counter is paused and does not increment during debug mode. Trigger inputs and input capture events are also ignored. 11 TPM counter continues in debug mode. |
| 5 DOZEEN | <p>Doze Enable</p> <p>Configures the TPM behavior in wait mode.</p> <ul style="list-style-type: none"> 0 Internal TPM counter continues in Doze mode. 1 Internal TPM counter is paused and does not increment during Doze mode. Trigger inputs and input capture events are also ignored. |
| Reserved | <p>This field is reserved.</p> <p>This read-only field is reserved and always has the value 0.</p> |

31.5 Functional description

The following sections describe the TPM features.

31.5.1 Clock domains

The TPM module supports two clock domains.

The bus clock domain is used by the register interface and for synchronizing interrupts.

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.

31.5.1.1 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.

31.5.2 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.

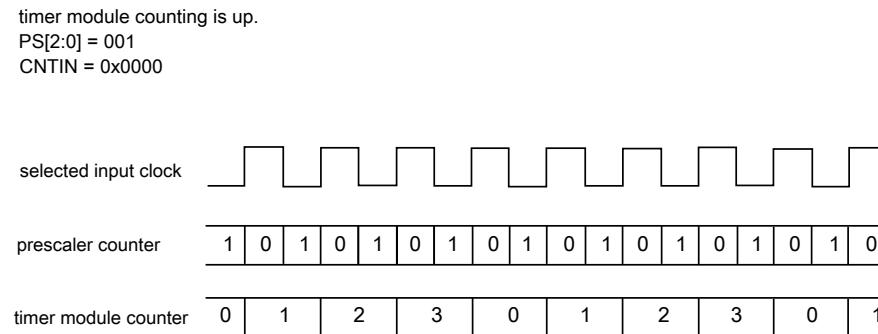


Figure 31-2. Example of the Prescaler Counter

31.5.3 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](#))

31.5.3.1 Up counting

Up counting is selected when SC[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.

Functional description

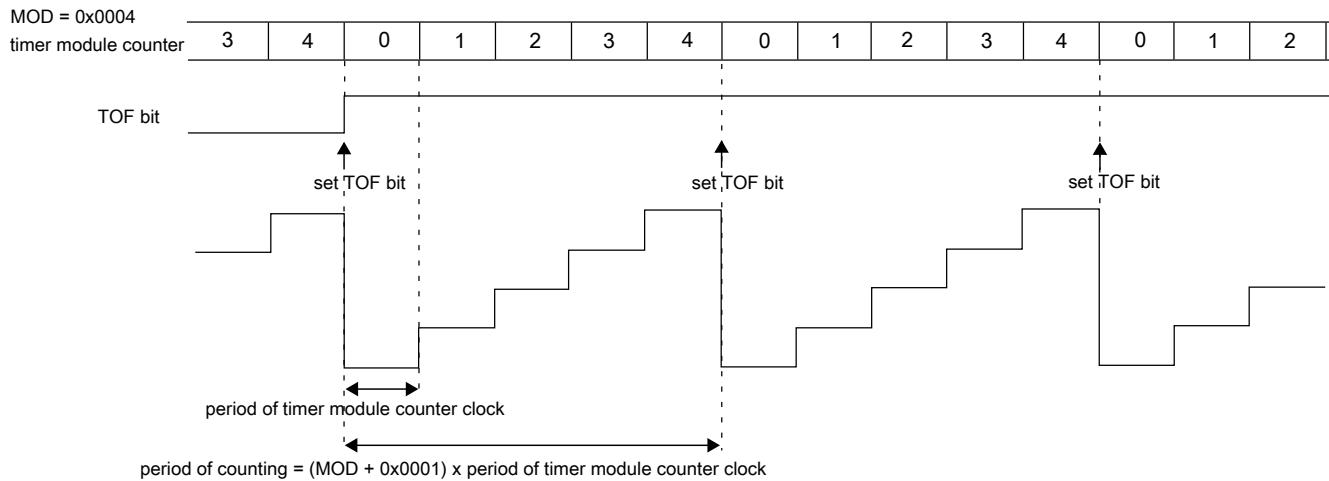


Figure 31-3. Example of TPM Up Counting

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.

31.5.3.2 Up-down counting

Up-down counting is selected when SC[CPWMS] = 1. When configured for up-down counting, configuring CONF[MOD] to less than 2 is not supported.

The value of 0 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 $2 \times \text{MOD} \times \text{period of the TPM counter clock}$.

The TOF bit is set when the TPM counter changes from MOD to (MOD – 1).

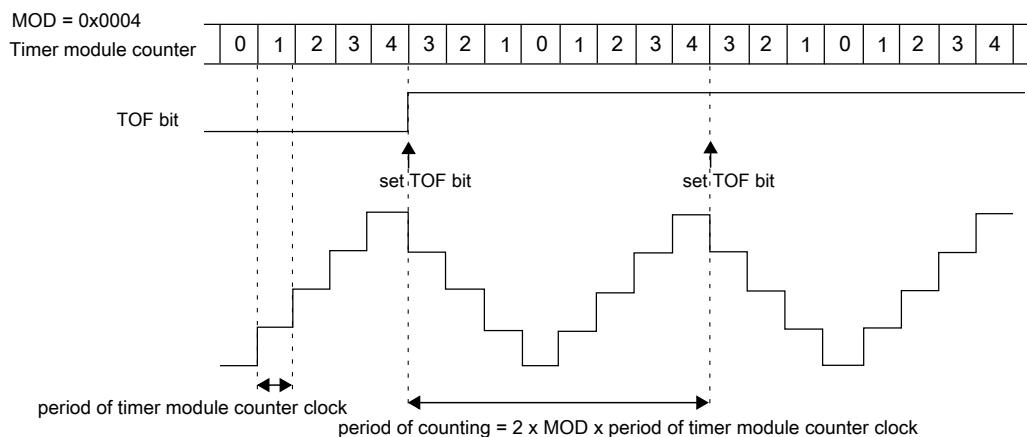


Figure 31-4. Example of up-down counting

31.5.3.3 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).

31.5.3.4 Global time base (GTB)

The global time base (GTB) is a TPM function that allows multiple TPM modules to share the same timebase. When the global time base is enabled (CONF[GTBEN] = 1), the local TPM channels use the counter value, counter enable and overflow indication from the TPM generating the global time base. If the local TPM counter is not generating the global time base, then it can be used as an independent counter or pulse accumulator.

31.5.3.5 Counter trigger

The TPM counter can be configured to start, stop or reset in response to a hardware trigger input. The trigger input is synchronized to the asynchronous counter clock, so there is a 3 counter clock delay between the trigger assertion and the counter responding.

- When (CSOT = 1), the counter will not start incrementing until a rising edge is detected on the trigger input.
- When (CSOO= 1), the counter will stop incrementing whenever the TOF flag is set. The counter does not increment again unless it is disabled, or if CSOT = 1 and a rising edge is detected on the trigger input.
- When (CROT= 1), the counter will reset to zero as if an overflow occurred whenever a rising edge is detected on the trigger input.

31.5.4 Input Capture Mode

The input capture mode is selected when ($\text{CPWMS} = 0$), ($\text{MSnB:MSnA} = 0:0$), and ($\text{ELSnB:ELSnA} \neq 0:0$).

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 $\text{CHnIE} = 1$ (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.

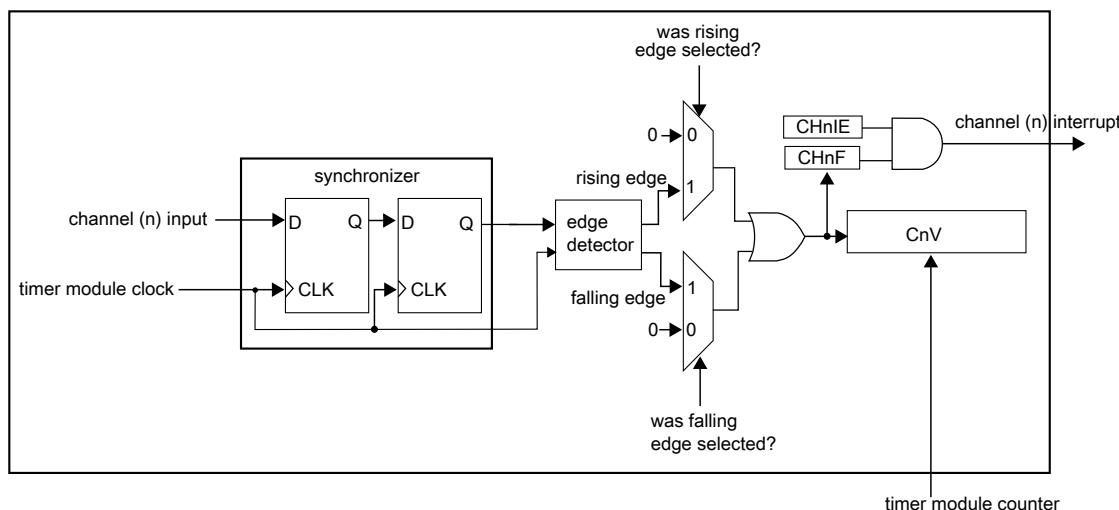


Figure 31-5. Input capture mode

The CHnF bit is set on the third rising edge of the counter clock after a valid edge occurs on the channel input.

31.5.5 Output Compare Mode

The output compare mode is selected when ($\text{CPWMS} = 0$), and ($\text{MSnB:MSnA} = X:1$).

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 if MSnB is clear. If MSnB is set then the channel (n) output is pulsed high or low for as long as the counter matches the value in the CnV register.

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).

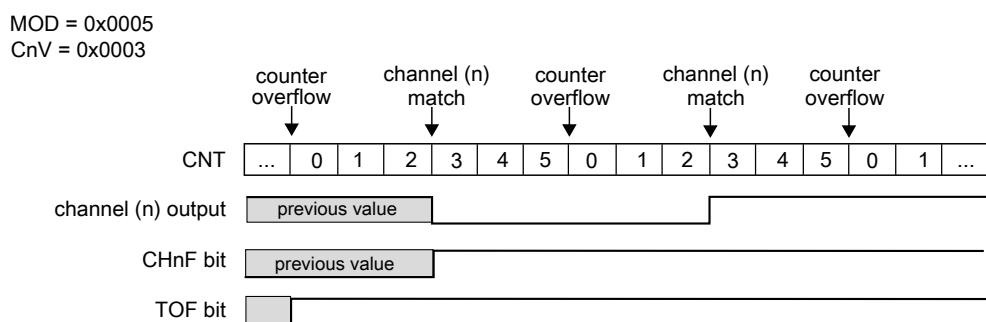


Figure 31-6. Example of the output compare mode when the match toggles the channel output

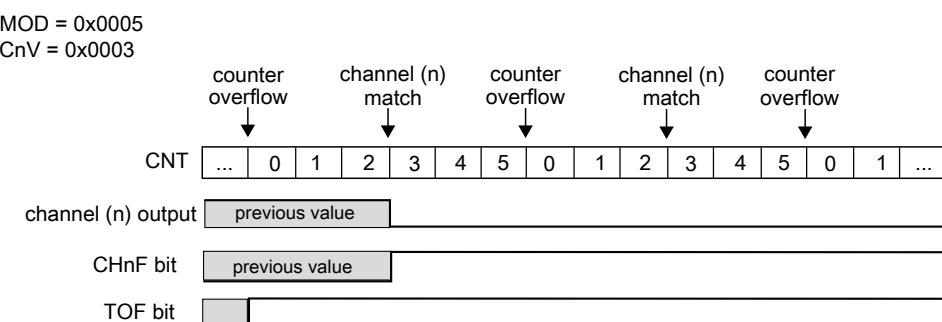


Figure 31-7. Example of the output compare mode when the match clears the channel output

MOD = 0x0005
CnV = 0x0003

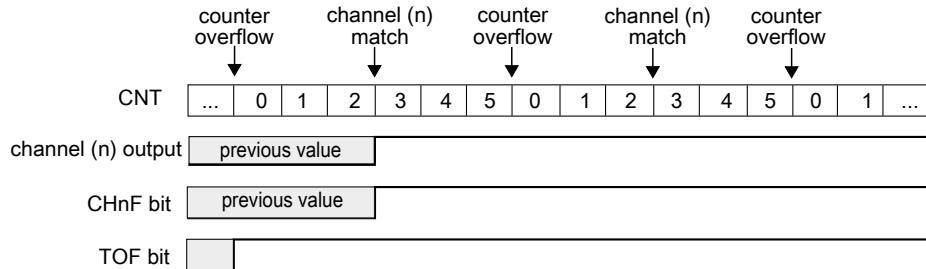


Figure 31-8. Example of the output compare mode when the match sets the channel output

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.

31.5.6 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.

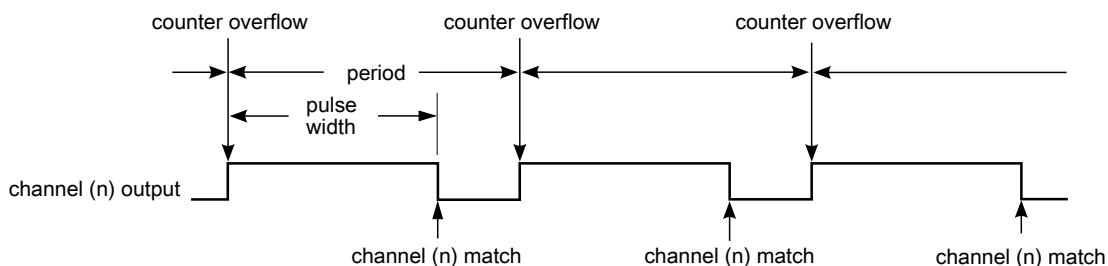


Figure 31-9. EPWM period and pulse width with ELSnB:ELSnA = 1:0

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).

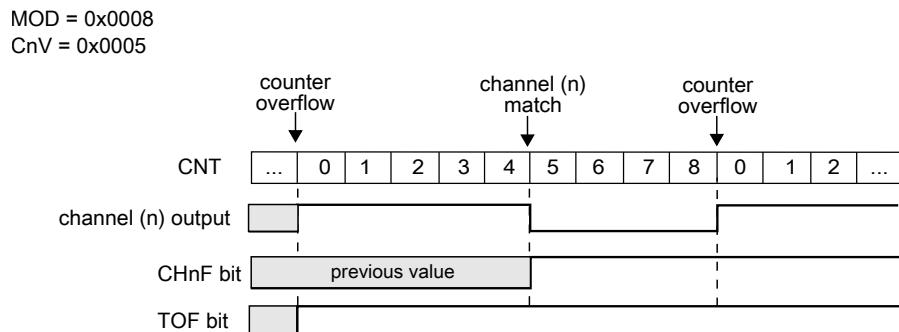


Figure 31-10. EPWM signal with ELSnB:ELSnA = 1:0

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).

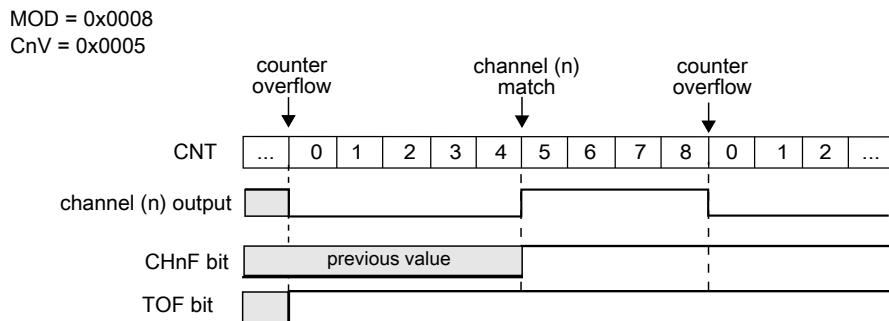


Figure 31-11. EPWM signal with ELSnB:ELSnA = X:1

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.

31.5.7 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 $2 \times \text{CnV}$ and the period is determined by $2 \times \text{MOD}$ (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.

Functional description

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).

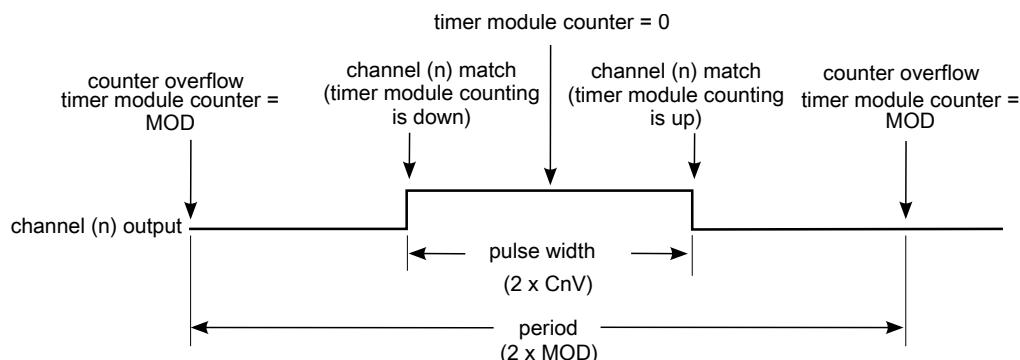


Figure 31-12. CPWM period and pulse width with ELSnB:ELSnA = 1:0

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).

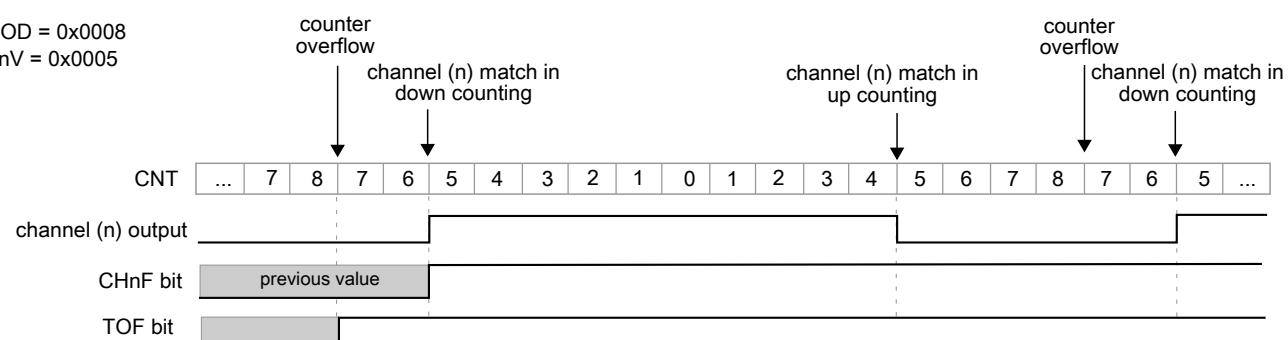
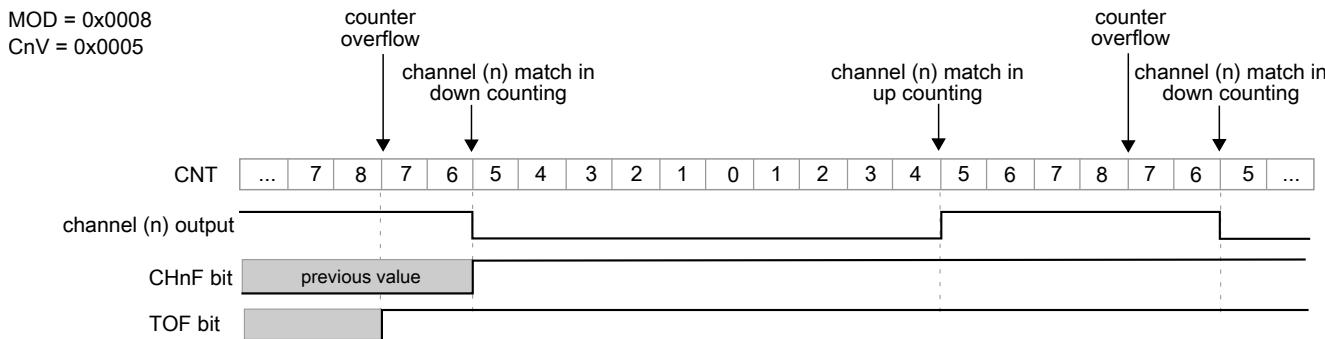


Figure 31-13. CPWM signal with ELSnB:ELSnA = 1:0

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 31-14. 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.

31.5.8 Registers Updated from Write Buffers

31.5.8.1 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 $(MOD - 1)$.

31.5.8.2 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).

31.5.9 Output triggers

The TPM generates output triggers for the counter and each channel that can be used to trigger events in other peripherals.

The counter trigger asserts whenever the TOF is set and remains asserted until the next increment.

Each TPM channel generates both a pre-trigger output and a trigger output. The pre-trigger output asserts whenever the CHnF is set, the trigger output asserts on the first counter increment after the pre-trigger asserts, and then both the trigger and pre-trigger negate on the first counter increment after the trigger asserts.

31.5.10 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).

31.5.11 TPM Interrupts

This section describes TPM interrupts.

31.5.11.1 Timer Overflow Interrupt

The timer overflow interrupt is generated when (TOIE = 1) and (TOF = 1).

31.5.11.2 Channel (n) Interrupt

The channel (n) interrupt is generated when (CHnIE = 1) and (CHnF = 1).

Chapter 32

Low-Power Timer (LPTMR)

32.1 Chip-specific LPTMR 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 2^N 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.

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

32.1.1 LPTMR pulse counter input options

LPTMR_CSR[TPS] configures the input source used in pulse counter mode. The following table shows the chip-specific input assignments for this field.

| 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 |

32.1.2 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 LPTMR0_PSR[PCS]. The following table shows the chip-specific clock assignments for this field.

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 VLLS modes) |
| 01 | 1 | LPO—1 kHz clock (not available in VLLS0 mode) |
| 10 | 2 | ERCLK32K (not available in VLLS0 mode when using 32 kHz oscillator) |
| 11 | 3 | OSCERCLK—external reference clock (not available in VLLS0 mode) |

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

32.2 Introduction

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.

32.2.1 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

32.2.2 Modes of operation

The following table describes the operation of the LPTMR module in various modes.

Table 32-1. 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. |

32.3 LPTMR signal descriptions

Table 32-2. LPTMR signal descriptions

| Signal | I/O | Description |
|------------|-----|-------------------------|
| LPTMR_ALTn | I | Pulse Counter Input pin |

32.3.1 Detailed signal descriptions

Table 32-3. 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. 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. |

32.4 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 | 32.4.1/488 |
| 4004_0004 | Low Power Timer Prescale Register (LPTMR0_PSR) | 32 | R/W | 0000_0000h | 32.4.2/489 |
| 4004_0008 | Low Power Timer Compare Register (LPTMR0_CMR) | 32 | R/W | 0000_0000h | 32.4.3/491 |
| 4004_000C | Low Power Timer Counter Register (LPTMR0_CNR) | 32 | R | 0000_0000h | 32.4.4/491 |

32.4.1 Low Power Timer Control Status Register (LPTMRx_CSR)

Address: 4004_0000h base + 0h offset = 4004_0000h

| | | | | | | | | | | | | | | | | | |
|-------|----|----|----|----|----|----|----|----|---|-----|-----|-----|-----|-----|-----|-----|----|
| 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 | | | | | | | | | | TCF | TIE | TPS | TPP | TFC | TMS | TEN | |
| W | | | | | | | | | | w1c | | | | | | | |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

LPTMRx_CSR field descriptions

| Field | Description |
|------------------|---|
| 31–8 Reserved | This field is reserved. This read-only field is reserved and always has the value 0. |
| 7 TCF | Timer Compare Flag TCF is set when the LPTMR is enabled and the CNR equals the CMR and increments. TCF is cleared when the LPTMR is disabled or a logic 1 is written to it. 0 The value of CNR is not equal to CMR and increments. 1 The value of CNR is equal to CMR and increments. |
| 6 TIE | Timer Interrupt Enable When TIE is set, the LPTMR Interrupt is generated whenever TCF is also set. 0 Timer interrupt disabled. 1 Timer interrupt enabled. |
| 5–4 TPS | Timer Pin Select Configures the input source to be used in Pulse Counter mode. TPS must be altered only when the LPTMR is disabled. The input connections vary by device. See the chip-specific LPTMR information for information on the connections to these inputs. 00 Pulse counter input 0 is selected. |

Table continues on the next page...

LPTMRx_CSR field descriptions (continued)

| Field | Description |
|-------|--|
| | 01 Pulse counter input 1 is selected. 10 Pulse counter input 2 is selected. 11 Pulse counter input 3 is selected. |
| 3 TPP | Timer Pin Polarity Configures the polarity of the input source in Pulse Counter mode. TPP must be changed only when the LPTMR is disabled. 0 Pulse Counter input source is active-high, and the CNR will increment on the rising-edge. 1 Pulse Counter input source is active-low, and the CNR will increment on the falling-edge. |
| 2 TFC | Timer Free-Running Counter 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. 0 CNR is reset whenever TCF is set. 1 CNR is reset on overflow. |
| 1 TMS | Timer Mode Select Configures the mode of the LPTMR. TMS must be altered only when the LPTMR is disabled. 0 Time Counter mode. 1 Pulse Counter mode. |
| 0 TEN | Timer Enable 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. 0 LPTMR is disabled and internal logic is reset. 1 LPTMR is enabled. |

32.4.2 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 Reserved | This field is reserved. This read-only field is reserved and always has the value 0. |
| 6–3 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"> 0000 Prescaler divides the prescaler clock by 2; glitch filter does not support this configuration. 0001 Prescaler divides the prescaler clock by 4; glitch filter recognizes change on input pin after 2 rising clock edges. 0010 Prescaler divides the prescaler clock by 8; glitch filter recognizes change on input pin after 4 rising clock edges. 0011 Prescaler divides the prescaler clock by 16; glitch filter recognizes change on input pin after 8 rising clock edges. 0100 Prescaler divides the prescaler clock by 32; glitch filter recognizes change on input pin after 16 rising clock edges. 0101 Prescaler divides the prescaler clock by 64; glitch filter recognizes change on input pin after 32 rising clock edges. 0110 Prescaler divides the prescaler clock by 128; glitch filter recognizes change on input pin after 64 rising clock edges. 0111 Prescaler divides the prescaler clock by 256; glitch filter recognizes change on input pin after 128 rising clock edges. 1000 Prescaler divides the prescaler clock by 512; glitch filter recognizes change on input pin after 256 rising clock edges. 1001 Prescaler divides the prescaler clock by 1024; glitch filter recognizes change on input pin after 512 rising clock edges. 1010 Prescaler divides the prescaler clock by 2048; glitch filter recognizes change on input pin after 1024 rising clock edges. 1011 Prescaler divides the prescaler clock by 4096; glitch filter recognizes change on input pin after 2048 rising clock edges. 1100 Prescaler divides the prescaler clock by 8192; glitch filter recognizes change on input pin after 4096 rising clock edges. 1101 Prescaler divides the prescaler clock by 16,384; glitch filter recognizes change on input pin after 8192 rising clock edges. 1110 Prescaler divides the prescaler clock by 32,768; glitch filter recognizes change on input pin after 16,384 rising clock edges. 1111 Prescaler divides the prescaler clock by 65,536; glitch filter recognizes change on input pin after 32,768 rising clock edges. |
| 2 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"> 0 Prescaler/glitch filter is enabled. 1 Prescaler/glitch filter is bypassed. |
| 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>NOTE: 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. |

32.4.3 Low Power Timer Compare Register (LPTMRx_CMR)

Address: 4004_0000h base + 8h offset = 4004_0008h

| 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 | | | |

LPTMRx_CMR field descriptions

| Field | Description |
|-------------------|---|
| 31–16 Reserved | This field is reserved. This read-only field is reserved and always has the value 0. |
| COMPARE | Compare Value When the LPTMR is enabled and the CNR equals the value in the CMR and increments, TCF is set and the hardware trigger asserts until the next time the CNR increments. If the CMR is 0, the hardware trigger will remain asserted until the LPTMR is disabled. If the LPTMR is enabled, the CMR must be altered only when TCF is set. |

32.4.4 Low Power Timer Counter Register (LPTMRx_CNR)

Address: 4004_0000h base + Ch offset = 4004_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 | | | | | | | | | | | | | | | | 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 | | | |

LPTMRx_CNR field descriptions

| Field | Description |
|-------------------|---|
| 31–16 Reserved | This field is reserved. This read-only field is reserved and always has the value 0. |
| COUNTER | Counter Value |

32.5 Functional description

32.5.1 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.

32.5.2 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.

32.5.3 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.

32.5.3.1 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.

32.5.3.2 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.

32.5.3.3 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.

32.5.3.4 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.

32.5.4 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.

32.5.5 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.

32.5.6 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. |

32.5.7 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 33

Real Time Clock (RTC)

33.1 Chip-specific RTC 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 32 kHz crystal operation in all power modes except VLLS0, and through any System Reset. When OSCE is enabled, the RTC also overrides the capacitor configurations.

33.1.1 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].

When SIM_SOPT2[RTCCLKOUTSEL] = 0, the RTC 1 Hz clock is output is selected on the RTC_CLKOUT pin. When SIM_SOPT2[RTCCLKOUTSEL] = 1, OSCERCLK clock is output on the RTC_CLKOUT pin.

33.2 Introduction

33.2.1 Features

The RTC module features include:

- 32-bit seconds counter with roll-over protection and 32-bit alarm

Register definition

- 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 with optional interrupt

33.2.2 Modes of operation

The RTC remains functional in all low power modes and can generate an interrupt to exit any low power mode.

33.2.3 RTC signal descriptions

Table 33-1. RTC signal descriptions

| Signal | Description | I/O |
|------------|-------------------------|-----|
| RTC_CLKOUT | 1 Hz square-wave output | O |

33.2.3.1 RTC clock output

The clock to the seconds counter is available on the RTC_CLKOUT signal. It is a 1 Hz square wave output.

33.3 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 | 33.3.1/499 |
| 4003_D004 | RTC Time Prescaler Register (RTC_TPR) | 32 | R/W | 0000_0000h | 33.3.2/499 |
| 4003_D008 | RTC Time Alarm Register (RTC_TAR) | 32 | R/W | 0000_0000h | 33.3.3/500 |
| 4003_D00C | RTC Time Compensation Register (RTC_TCR) | 32 | R/W | 0000_0000h | 33.3.4/500 |
| 4003_D010 | RTC Control Register (RTC_CR) | 32 | R/W | 0000_0000h | 33.3.5/502 |
| 4003_D014 | RTC Status Register (RTC_SR) | 32 | R/W | 0000_0001h | 33.3.6/504 |
| 4003_D018 | RTC Lock Register (RTC_LR) | 32 | R/W | 0000_00FFh | 33.3.7/505 |
| 4003_D01C | RTC Interrupt Enable Register (RTC_IER) | 32 | R/W | 0000_0007h | 33.3.8/506 |

33.3.1 RTC Time Seconds Register (RTC TSR)

Address: 4003 D000h base + 0h offset = 4003 D000h

RTC TSR field descriptions

| Field | Description |
|-------|--|
| 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> |

33.3.2 RTC Time Prescaler Register (RTC_TPR)

Address: 4003 D000h base + 4h offset = 4003 D004h

RTC_TPR field descriptions

| Field | Description |
|-------------------|--|
| 31–16 Reserved | This field is reserved. This read-only field is reserved and always has the value 0. |
| TPR | Time Prescaler Register 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. |

33.3.3 RTC Time Alarm Register (RTC_TAR)

Address: 4003_D000h base + 8h offset = 4003_D008h

| 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

RTC_TAR field descriptions

| Field | Description |
|-------|---|
| TAR | Time Alarm Register 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]. |

33.3.4 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 | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| W | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |

Reset 0

RTC_TCR field descriptions

| Field | Description |
|--------------|--|
| 31–24 CIC | Compensation Interval Counter 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. |
| 23–16 TCV | Time Compensation Value |

Table continues on the next page...

RTC_TCR field descriptions (continued)

| Field | Description | | | | | | | | | | | | | | |
|-------------|---|-----|---|-----|-----|-----|---|-----|---|-----|---|-----|-----|-----|---|
| | 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). | | | | | | | | | | | | | | |
| 15–8 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> | | | | | | | | | | | | | | |
| 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> <tbody> <tr> <td>80h</td> <td>Time Prescaler Register overflows every 32896 clock cycles.</td> </tr> <tr> <td>...</td> <td>...</td> </tr> <tr> <td>FFh</td> <td>Time Prescaler Register overflows every 32769 clock cycles.</td> </tr> <tr> <td>00h</td> <td>Time Prescaler Register overflows every 32768 clock cycles.</td> </tr> <tr> <td>01h</td> <td>Time Prescaler Register overflows every 32767 clock cycles.</td> </tr> <tr> <td>...</td> <td>...</td> </tr> <tr> <td>7Fh</td> <td>Time Prescaler Register overflows every 32641 clock cycles.</td> </tr> </tbody> </table> | 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. |
| 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. | | | | | | | | | | | | | | |

33.3.5 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 | | | | | 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 | | WPS | UM | SUP | WPE | SWR |
| W | | | | | | | | | | | | | | | | | |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | |

RTC_CR field descriptions

| Field | Description |
|-------------------|---|
| 31–24 Reserved | This field is reserved. This read-only field is reserved and always has the value 0. |
| 23–15 Reserved | This field is reserved. This read-only field is reserved and always has the value 0. |
| 14 Reserved | This field is reserved. It must always be written to 0. |
| 13 SC2P | Oscillator 2pF Load Configure 0 Disable the load. 1 Enable the additional load. |
| 12 SC4P | Oscillator 4pF Load Configure 0 Disable the load. 1 Enable the additional load. |

Table continues on the next page...

RTC_CR field descriptions (continued)

| Field | Description |
|-----------------|---|
| 11 SC8P | Oscillator 8pF Load Configure 0 Disable the load. 1 Enable the additional load. |
| 10 SC16P | Oscillator 16pF Load Configure 0 Disable the load. 1 Enable the additional load. |
| 9 CLKO | Clock Output 0 The 32 kHz clock is output to other peripherals. 1 The 32 kHz clock is not output to other peripherals. |
| 8 OSCE | Oscillator Enable 0 32.768 kHz oscillator is disabled. 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. |
| 7–5 Reserved | This field is reserved. This read-only field is reserved and always has the value 0. |
| 4 WPS | Wakeup Pin Select The wakeup pin is optional and not available on all devices. 0 Wakeup pin asserts (active low, open drain) if the RTC interrupt asserts or the wakeup pin is turned on. 1 Wakeup pin instead outputs the RTC 32kHz clock, provided the wakeup pin is turned on and the 32kHz clock is output to other peripherals. |
| 3 UM | Update Mode 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. 0 Registers cannot be written when locked. 1 Registers can be written when locked under limited conditions. |
| 2 SUP | Supervisor Access 0 Non-supervisor mode write accesses are not supported and generate a bus error. 1 Non-supervisor mode write accesses are supported. |
| 1 WPE | Wakeup Pin Enable The wakeup pin is optional and not available on all devices. 0 Wakeup pin is disabled. 1 Wakeup pin is enabled and wakeup pin asserts if the RTC interrupt asserts or the wakeup pin is turned on. |
| 0 SWR | Software Reset 0 No effect. 1 Resets all RTC registers except for the SWR bit . The SWR bit is cleared by POR and by software explicitly clearing it. |

33.3.6 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 | 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 | 1 |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 1 |

RTC_SR field descriptions

| Field | Description |
|------------------|---|
| 31–5 Reserved | This field is reserved. This read-only field is reserved and always has the value 0. |
| 4 TCE | Time Counter Enable When time counter is disabled the TSR register and TPR register are writeable, but do not increment. When time counter is enabled the TSR register and TPR register are not writeable, but increment. 0 Time counter is disabled. 1 Time counter is enabled. |
| 3 Reserved | This field is reserved. This read-only field is reserved and always has the value 0. |
| 2 TAF | Time Alarm Flag 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. 0 Time alarm has not occurred. 1 Time alarm has occurred. |
| 1 TOF | Time Overflow Flag 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. 0 Time overflow has not occurred. 1 Time overflow has occurred and time counter is read as zero. |
| 0 TIF | Time Invalid Flag 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. 0 Time is valid. 1 Time is invalid and time counter is read as zero. |

33.3.7 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 Reserved | This field is reserved. This read-only field is reserved and always has the value 0. |
| 7 Reserved | This field is reserved. This read-only field is reserved and always has the value 1. |
| 6 LRL | Lock Register Lock After being cleared, this bit can be set only by POR or software reset. 0 Lock Register is locked and writes are ignored. 1 Lock Register is not locked and writes complete as normal. |
| 5 SRL | Status Register Lock After being cleared, this bit can be set only by POR or software reset. 0 Status Register is locked and writes are ignored. 1 Status Register is not locked and writes complete as normal. |
| 4 CRL | Control Register Lock After being cleared, this bit can only be set by POR. 0 Control Register is locked and writes are ignored. 1 Control Register is not locked and writes complete as normal. |
| 3 TCL | Time Compensation Lock After being cleared, this bit can be set only by POR or software reset. 0 Time Compensation Register is locked and writes are ignored. 1 Time Compensation Register is not locked and writes complete as normal. |
| Reserved | This field is reserved. This read-only field is reserved and always has the value 1. |

33.3.8 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 | 0 | | | | | | | | 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 |
|-------------------|---|
| 31–16 Reserved | This field is reserved. This read-only field is reserved and always has the value 0. |
| 15–8 Reserved | This field is reserved. This read-only field is reserved and always has the value 0. |
| 7 WPON | Wakeup Pin On 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. 0 No effect. 1 If the wakeup pin is enabled, then the wakeup pin will assert. |
| 6–5 Reserved | This field is reserved. |
| 4 TSIE | Time Seconds Interrupt Enable 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). 0 Seconds interrupt is disabled. 1 Seconds interrupt is enabled. |
| 3 Reserved | This field is reserved. |
| 2 TAIE | Time Alarm Interrupt Enable 0 Time alarm flag does not generate an interrupt. 1 Time alarm flag does generate an interrupt. |
| 1 TOIE | Time Overflow Interrupt Enable |

Table continues on the next page...

RTC_IER field descriptions (continued)

| Field | Description |
|-----------|---|
| | 0 Time overflow flag does not generate an interrupt. 1 Time overflow flag does generate an interrupt. |
| 0 TIIE | Time Invalid Interrupt Enable 0 Time invalid flag does not generate an interrupt. 1 Time invalid flag does generate an interrupt. |

33.4 Functional description

33.4.1 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.

33.4.1.1 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.

33.4.1.2 Software reset

Writing 1 to CR[SWR] forces the equivalent of a POR to the rest of the RTC module. CR[SWR] is not affected by the software reset and must be cleared by software.

33.4.1.3 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.

33.4.2 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.

Reading the time counter (either seconds or prescaler) while it is incrementing may return invalid data due to synchronization of the read data bus. If it is necessary for software to read the prescaler or seconds counter when they could be incrementing, it is recommended that two read accesses are performed and that software verifies that the same data was returned for both reads.

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.

33.4.3 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.

33.4.4 Time alarm

The Time Alarm register (TAR), SR[TAF], and IER[TAIE] allow the RTC to generate an interrupt at a predefined time. The 32-bit TAR is compared with the 32-bit Time Seconds register (TSR) each time it increments. SR[TAF] will set when TAR equals TSR and TSR increments.

SR[TAF] is cleared by writing TAR. This will usually be the next alarm value, although writing a value that is less than TSR, such as 0, will prevent SR[TAF] from setting again. SR[TAF] cannot otherwise be disabled, although the interrupt it generates is enabled or disabled by IER[TAIE].

33.4.5 Update mode

The Update Mode field in the Control register (CR[UM]) configures software write access to the Time Counter Enable (SR[TCE]) field. When CR[UM] is clear, SR[TCE] can be written only when LR[SRL] is set. When CR[UM] is set, 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].

33.4.6 Register lock

The Lock register (LR) can be used to block write accesses to certain registers until the next POR or software reset. Locking the Control register (CR) will disable the software reset. Locking LR will block future updates to LR.

Write accesses to a locked register are ignored and do not generate a bus error.

33.4.7 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 34

Serial Peripheral Interface (SPI)

34.1 Chip-specific SPI information

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

SPI0 is clocked on the bus clock.

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

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

34.2 Introduction

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.

34.2.1 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

34.2.2 Modes of operation

The SPI functions in the following three modes.

- 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 C2[SPISWAI] is clear, the SPI operates like in Run mode. If C2[SPISWAI] 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](#).

34.2.3 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.

34.2.3.1 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.

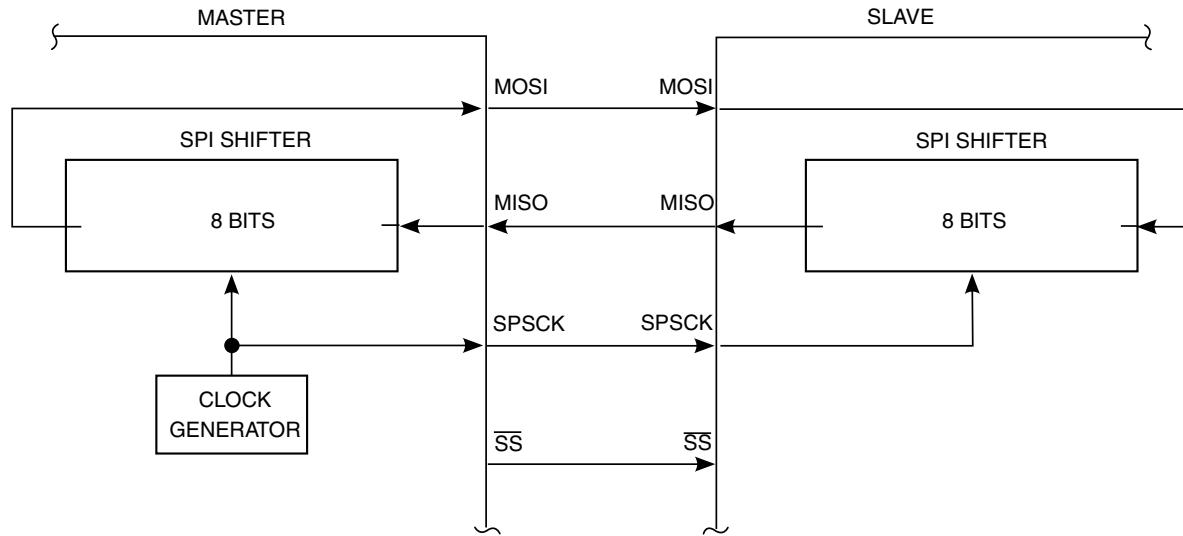


Figure 34-1. SPI system connections

34.2.3.2 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 SPI_x_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 SPI_x_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.

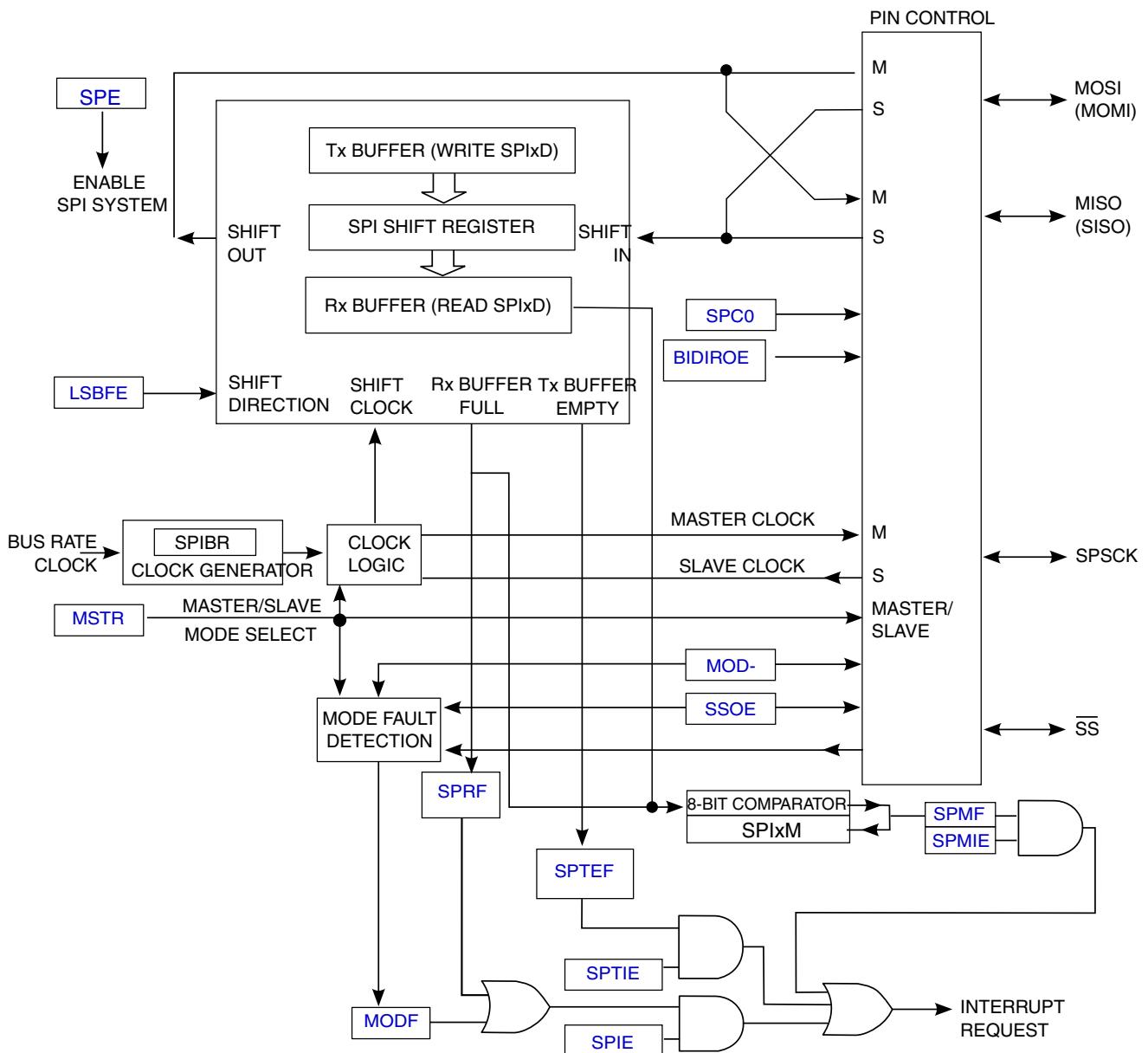


Figure 34-2. SPI module block diagram without FIFO

34.3 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).

34.3.1 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.

34.3.2 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).

34.3.3 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).

34.3.4 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).

34.4 Memory map/register definition

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 Status Register (SPI0_S) | 8 | R | 20h | 34.4.1/517 |
| 1 | 4007_6001 | SPI Baud Rate Register (SPI0_BR) | 8 | R/W | 00h | 34.4.2/518 |
| 2 | 4007_6002 | SPI Control Register 2 (SPI0_C2) | 8 | R/W | 00h | 34.4.3/519 |
| 3 | 4007_6003 | SPI Control Register 1 (SPI0_C1) | 8 | R/W | 04h | 34.4.4/521 |
| 4 | 4007_6004 | SPI Match Register (SPI0_M) | 8 | R/W | 00h | 34.4.5/522 |
| 6 | 4007_6006 | SPI Data Register (SPI0_D) | 8 | R/W | 00h | 34.4.6/523 |

34.4.1 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 + 0h offset = 4007_6000h

| 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 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. SPRF is cleared by reading SPRF while it is set and then reading the SPI data register.</p> <table> <tr> <td>0 No data available in the receive data buffer</td> </tr> <tr> <td>1 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 | | | |

Table continues on the next page...

SPI0_S field descriptions (continued)

| Field | Description |
|------------|--|
| 6 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> <p>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</p> |
| 5 SPTEF | <p>SPI Transmit Buffer Empty Flag</p> <p>This bit is set when the transmit data buffer is empty. 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. 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> <p>0 SPI transmit buffer not empty 1 SPI transmit buffer empty</p> |
| 4 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 C1[MSTR] is 1, C2[MODFEN] is 1, and C1[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> <p>0 No mode fault error 1 Mode fault error detected</p> |
| Reserved | <p>This field is reserved.</p> <p>This read-only field is reserved and always has the value 0.</p> |

34.4.2 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 + 1h offset = 4007_6001h

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|-------|---|---|-----------|---|---|----------|---|---|
| Read | 0 | | SPPR[2:0] | | | SPR[3:0] | | |
| Write | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

SPI0_BR field descriptions

| Field | Description | | | | | | | | | | | | | | | | | | | | |
|------------------|--|------|-----------------------------------|------|-----------------------------------|------|-----------------------------------|------|-----------------------------------|------|-----------------------------------|------|-----------------------------------|------|-----------------------------------|------|-----------------------------------|------|---------------------------|------------|----------|
| 7 Reserved | This field is reserved. This read-only field is reserved and always has the value 0. | | | | | | | | | | | | | | | | | | | | |
| 6–4 SPPR[2:0] | <p>SPI Baud Rate Prescale Divisor</p> <p>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.</p> <table> <tbody> <tr><td>000</td><td>Baud rate prescaler divisor is 1.</td></tr> <tr><td>001</td><td>Baud rate prescaler divisor is 2.</td></tr> <tr><td>010</td><td>Baud rate prescaler divisor is 3.</td></tr> <tr><td>011</td><td>Baud rate prescaler divisor is 4.</td></tr> <tr><td>100</td><td>Baud rate prescaler divisor is 5.</td></tr> <tr><td>101</td><td>Baud rate prescaler divisor is 6.</td></tr> <tr><td>110</td><td>Baud rate prescaler divisor is 7.</td></tr> <tr><td>111</td><td>Baud rate prescaler divisor is 8.</td></tr> </tbody> </table> | 000 | Baud rate prescaler divisor is 1. | 001 | Baud rate prescaler divisor is 2. | 010 | Baud rate prescaler divisor is 3. | 011 | Baud rate prescaler divisor is 4. | 100 | Baud rate prescaler divisor is 5. | 101 | Baud rate prescaler divisor is 6. | 110 | Baud rate prescaler divisor is 7. | 111 | Baud rate prescaler divisor is 8. | | | | |
| 000 | Baud rate prescaler divisor is 1. | | | | | | | | | | | | | | | | | | | | |
| 001 | Baud rate prescaler divisor is 2. | | | | | | | | | | | | | | | | | | | | |
| 010 | Baud rate prescaler divisor is 3. | | | | | | | | | | | | | | | | | | | | |
| 011 | Baud rate prescaler divisor is 4. | | | | | | | | | | | | | | | | | | | | |
| 100 | Baud rate prescaler divisor is 5. | | | | | | | | | | | | | | | | | | | | |
| 101 | Baud rate prescaler divisor is 6. | | | | | | | | | | | | | | | | | | | | |
| 110 | Baud rate prescaler divisor is 7. | | | | | | | | | | | | | | | | | | | | |
| 111 | Baud rate prescaler divisor is 8. | | | | | | | | | | | | | | | | | | | | |
| SPR[3:0] | <p>SPI Baud Rate Divider</p> <p>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.</p> <table> <tbody> <tr><td>0000</td><td>Baud rate divisor is 2.</td></tr> <tr><td>0001</td><td>Baud rate divisor is 4.</td></tr> <tr><td>0010</td><td>Baud rate divisor is 8.</td></tr> <tr><td>0011</td><td>Baud rate divisor is 16.</td></tr> <tr><td>0100</td><td>Baud rate divisor is 32.</td></tr> <tr><td>0101</td><td>Baud rate divisor is 64.</td></tr> <tr><td>0110</td><td>Baud rate divisor is 128.</td></tr> <tr><td>0111</td><td>Baud rate divisor is 256.</td></tr> <tr><td>1000</td><td>Baud rate divisor is 512.</td></tr> <tr><td>All others</td><td>Reserved</td></tr> </tbody> </table> | 0000 | Baud rate divisor is 2. | 0001 | Baud rate divisor is 4. | 0010 | Baud rate divisor is 8. | 0011 | Baud rate divisor is 16. | 0100 | Baud rate divisor is 32. | 0101 | Baud rate divisor is 64. | 0110 | Baud rate divisor is 128. | 0111 | Baud rate divisor is 256. | 1000 | Baud rate divisor is 512. | All others | Reserved |
| 0000 | Baud rate divisor is 2. | | | | | | | | | | | | | | | | | | | | |
| 0001 | Baud rate divisor is 4. | | | | | | | | | | | | | | | | | | | | |
| 0010 | Baud rate divisor is 8. | | | | | | | | | | | | | | | | | | | | |
| 0011 | Baud rate divisor is 16. | | | | | | | | | | | | | | | | | | | | |
| 0100 | Baud rate divisor is 32. | | | | | | | | | | | | | | | | | | | | |
| 0101 | Baud rate divisor is 64. | | | | | | | | | | | | | | | | | | | | |
| 0110 | Baud rate divisor is 128. | | | | | | | | | | | | | | | | | | | | |
| 0111 | Baud rate divisor is 256. | | | | | | | | | | | | | | | | | | | | |
| 1000 | Baud rate divisor is 512. | | | | | | | | | | | | | | | | | | | | |
| All others | Reserved | | | | | | | | | | | | | | | | | | | | |

34.4.3 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 + 2h offset = 4007_6002h

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|------------|-------|----------|----------|--------|---------|----------|---------|------|
| Read Write | SPMIE | Reserved | Reserved | MODFEN | BIDIROE | Reserved | 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) 1 When SPMF is 1, requests a hardware interrupt</p> |
| 6 Reserved | <p>This field is reserved. Do not write to this reserved bit.</p> |
| 5 Reserved | <p>This field is reserved. Do not write to this reserved bit.</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 \overline{SS} pin is the slave select input.) In master mode, this bit determines how the \overline{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 \overline{SS} pin reverts to general-purpose I/O not controlled by SPI 1 Mode fault function enabled, master \overline{SS} pin acts as the mode fault input or the slave select output</p> |
| 3 BIDIROE | <p>Bidirectional Mode Output Enable</p> <p>When bidirectional mode is enabled because SPI pin control 0 (SPC0) is set to 1, BIDIROE determines whether the SPI data output driver is enabled to the single bidirectional SPI I/O pin. Depending on whether the SPI is configured as a master or a slave, it uses 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 1 SPI I/O pin enabled as an output</p> |
| 2 Reserved | <p>This field is reserved. Do not write to this reserved bit.</p> |
| 1 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. 1 SPI clocks stop when the MCU enters Wait mode.</p> |
| 0 SPC0 | <p>SPI Pin Control 0</p> <p>Enables bidirectional pin configurations.</p> <p>0 SPI uses separate pins for data input and data output (pin mode is normal). In master mode of operation: MISO is master in and MOSI is master out. In slave mode of operation: MISO is slave out and MOSI is slave in. 1 SPI configured for single-wire bidirectional operation (pin mode is bidirectional). 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. 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> |

34.4.4 SPI Control Register 1 (SPlx_C1)

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

Address: 4007_6000h base + 3h offset = 4007_6003h

| 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 SPIE | SPI Interrupt Enable: for SPRF and MODF Enables the interrupt for SPI receive buffer full (SPRF) and mode fault (MODF) events. 0 Interrupts from SPRF and MODF are inhibited—use polling 1 Request a hardware interrupt when SPRF or MODF is 1 |
| 6 SPE | SPI System Enable 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. 0 SPI system inactive 1 SPI system enabled |
| 5 SPTIE | SPI Transmit Interrupt Enable 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). 0 Interrupts from SPTEF inhibited (use polling) 1 When SPTEF is 1, hardware interrupt requested |
| 4 MSTR | Master/Slave Mode Select Selects master or slave mode operation. 0 SPI module configured as a slave SPI device 1 SPI module configured as a master SPI device |
| 3 CPOL | Clock Polarity Selects an inverted or non-inverted SPI clock. To transmit data between SPI modules, the SPI modules must have identical CPOL values. 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. 0 Active-high SPI clock (idles low) 1 Active-low SPI clock (idles high) |

Table continues on the next page...

SPI0_C1 field descriptions (continued)

| Field | Description |
|------------|---|
| 2 CPHA | <p>Clock Phase</p> <p>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> <ul style="list-style-type: none"> 0 First edge on SPSCK occurs at the middle of the first cycle of a data transfer. 1 First edge on SPSCK occurs at the start of the first cycle of a data transfer. |
| 1 SSOE | <p>Slave Select Output Enable</p> <p>This bit is used in combination with the Mode Fault Enable (MODFEN) field in the C2 register and the Master/Slave (MSTR) control bit to determine the function of the \overline{SS} pin.</p> <ul style="list-style-type: none"> 0 When C2[MODFEN] is 0: In master mode, \overline{SS} pin function is general-purpose I/O (not SPI). In slave mode, \overline{SS} pin function is slave select input. When C2[MODFEN] is 1: In master mode, \overline{SS} pin function is \overline{SS} input for mode fault. In slave mode, \overline{SS} pin function is slave select input. 1 When C2[MODFEN] is 0: In master mode, \overline{SS} pin function is general-purpose I/O (not SPI). In slave mode, \overline{SS} pin function is slave select input. When C2[MODFEN] is 1: In master mode, \overline{SS} pin function is automatic \overline{SS} output. In slave mode: \overline{SS} pin function is slave select input. |
| 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> <ul style="list-style-type: none"> 0 SPI serial data transfers start with the most significant bit. 1 SPI serial data transfers start with the least significant bit. |

34.4.5 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 in the S register (S[SPMF]) sets.

Address: 4007_6000h base + 4h offset = 4007_6004h

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|------------|-----------|---|---|---|---|---|---|---|
| Read Write | Bits[7:0] | | | | | | | |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

SPI0_M field descriptions

| Field | Description |
|-----------|-----------------------------------|
| Bits[7:0] | Hardware compare value (low byte) |

34.4.6 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. The S register must be read when S[SPTEF] is set before writing to the SPI data register; otherwise, the write is ignored.

Data may be read from the SPI data register any time after S[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 + 6h offset = 4007_6006h

| Bit | 7 | 6 | 5 | 4 | | 3 | 2 | 1 | 0 |
|-------|---|---|---|---|-----------|---|---|---|---|
| Read | | | | | Bits[7:0] | | | | |
| Write | | | | | | | | | |
| Reset | 0 | 0 | 0 | 0 | | 0 | 0 | 0 | 0 |

SPI0_D field descriptions

| Field | Description |
|-----------|-----------------|
| Bits[7:0] | Data (low byte) |

34.5 Functional description

This section provides the functional description of the module.

34.5.1 General

The SPI system is enabled by setting the SPI enable (SPE) bit in SPI Control Register 1. While C1[SPE] 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 ($\text{SPI}_x\text{_S}$) when $S[\text{SPTEF}] = 1$ and then writing data to the transmit data buffer (write to SPI_xD). When a transfer is complete, received data is moved into the receive data buffer. The SPI_xD register acts as the SPI receive data buffer for reads and as the SPI transmit data buffer for writes.

The Clock Phase Control (CPHA) and Clock Polarity Control (CPOL) bits in the SPI Control Register 1 ($\text{SPI}_x\text{_C1}$) 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. $C1[\text{CPHA}]$ 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 $C1[\text{MSTR}]$ is clear, slave mode is selected.

34.5.2 Master mode

The SPI operates in master mode when $C1[\text{MSTR}]$ is set. Only a master SPI module can initiate transmissions. A transmission begins by reading the $\text{SPI}_x\text{_S}$ register while $S[\text{SPTEF}] = 1$ 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 C2[MODFEN] and C1[SSOE] 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 C2[MODFEN] is set and C1[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 C1[MSTR] 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 S[MODF] 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 C1[CPOL], C1[CPHA], C1[SSOE], C1[LSBFE], C2[MODFEN], C2[SPC0], C2[BIDIROE] with C2[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.

34.5.3 Slave mode

The SPI operates in slave mode when the MSTR bit in SPI Control Register 1 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 C1[CPHA] 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 C1[LSBFE].

When C1[CPHA] is set, the first edge is used to get the first data bit onto the serial data output pin. When C1[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 C2[BIDIROE] with C2[SPC0] set, C1[CPOL], C1[CPHA], C1[SSOE], C1[LSBFE], C2[MODFEN], and C2[SPC0] in slave mode will corrupt a transmission in progress and must be avoided.

34.5.4 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 in the Control Register 1 to select one of four clock formats for data transfers. C1[CPOL] selectively inserts an inverter in series with the clock. C1[CPHA] chooses between two different clock phase relationships between the clock and data.

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 C1[CPOL]. The SAMPLE IN waveform applies to the MOSI input of a slave or the MISO input of a master. The MOSI waveform applies to the MOSI output pin from a master and the MISO waveform applies to the MISO output from a slave. The \overline{SS} OUT waveform applies to the slave select output from a master (provided C2[MODFEN] and C1[SSOE] = 1). The master \overline{SS} output goes to active low one-half SPSCK cycle before the start of the transfer and goes back high at the end of the eighth bit time of the transfer. The \overline{SS} IN waveform applies to the slave select input of a slave.

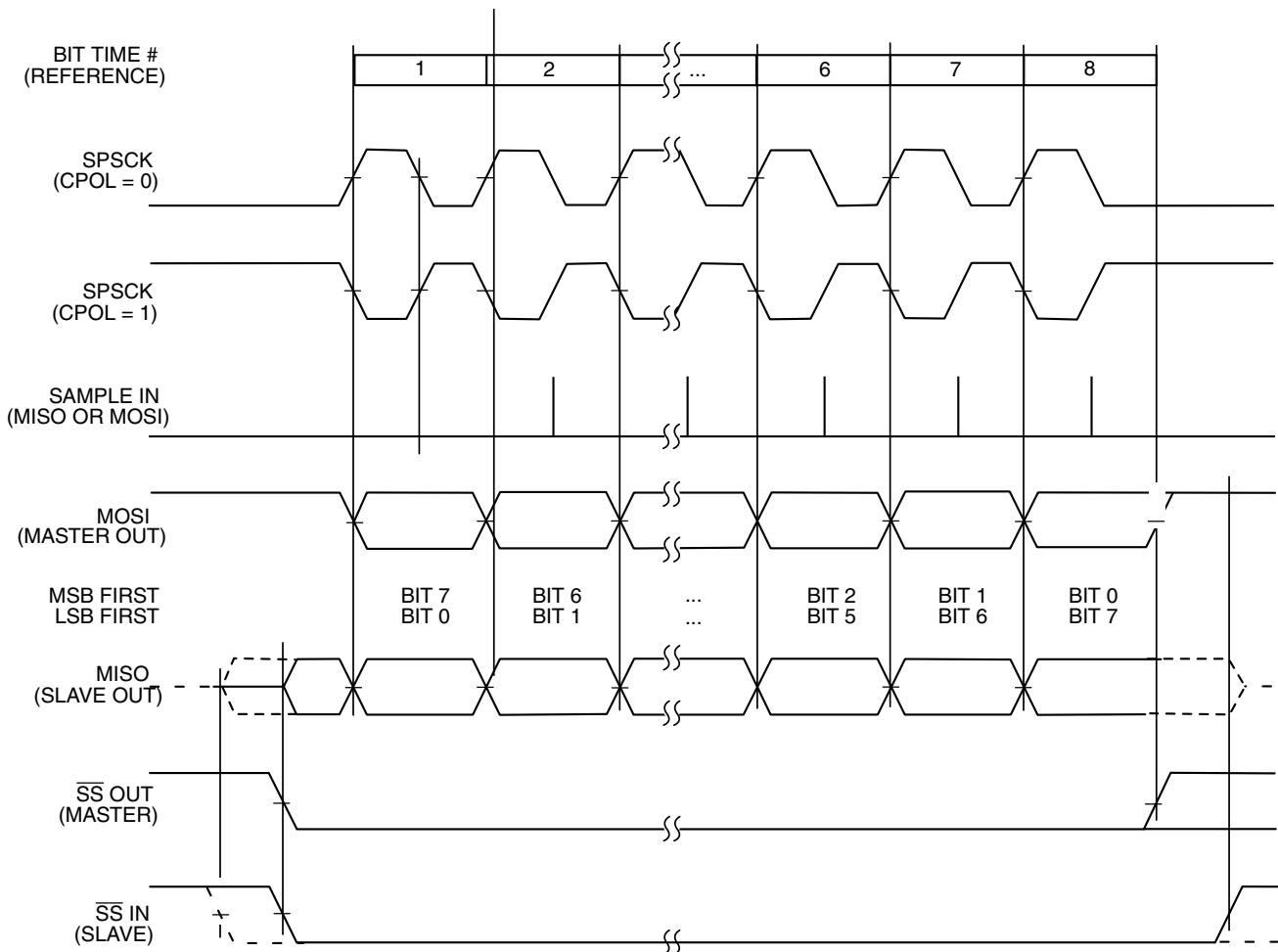


Figure 34-3. SPI clock formats (CPHA = 1)

When C1[CPHA] = 1, the slave begins to drive its MISO output when \overline{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 C1[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 C1[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 C2[MODFEN] and C1[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.

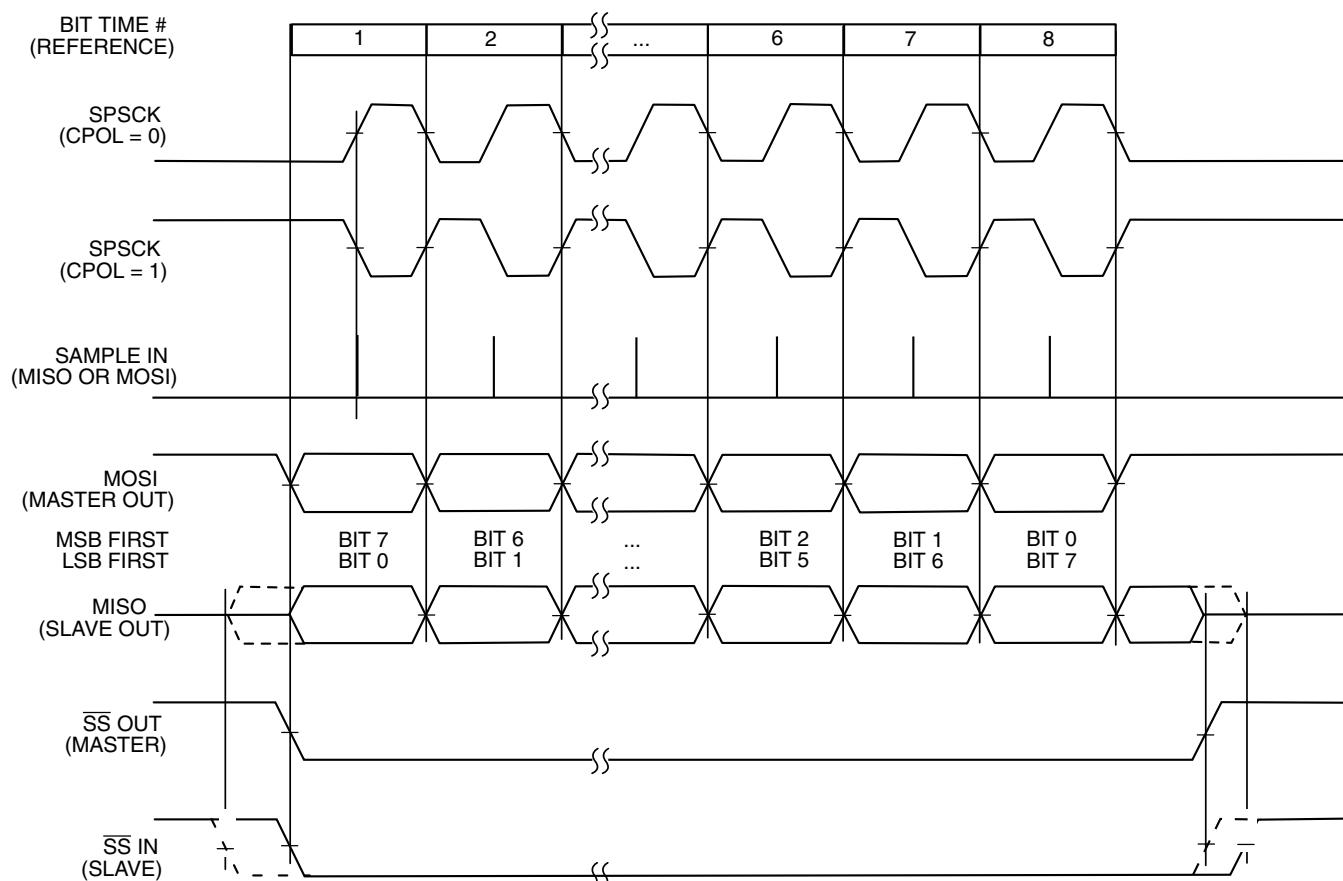


Figure 34-4. SPI clock formats (CPHA = 0)

When C1[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 C1[CPHA] = 0, the slave's SS input must go to its inactive high level between transfers.

34.5.5 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:

$$\text{BaudRate} = \text{BusClock} / \text{BaudRateDivisor}$$

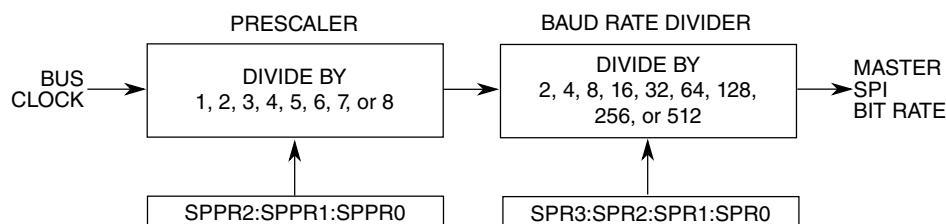


Figure 34-5. SPI baud rate generation

34.5.6 Special features

The following section describes the special features of SPI module.

34.5.6.1 $\overline{\text{SS}}$ 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 C1[SSOE] and C2[MODFEN] as shown in the description of C1[SSOE].

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.

34.5.6.2 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. C1[MSTR] 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 34-1. Normal Mode and Bidirectional Mode

| When SPE = 1 | Master Mode MSTR = 1 | Slave Mode MSTR = 0 |
|---------------------------------------|---|---|
| Normal Mode SPC0 = 0 | <pre> graph LR SPI[SPI] -- "Serial Out" --> MOSI[MOSI] SPI -- "Serial In" --> MISO[MISO] </pre> | <pre> graph LR SPI[SPI] -- "Serial In" --> MOSI[MOSI] SPI -- "Serial Out" --> MISO[MISO] </pre> |
| Bidirectional Mode SPC0 = 1 | <pre> graph LR SPI[SPI] -- "Serial Out" --> BIDIROE[BIDIROE] BIDIROE --> MOMI[MOMI] MOMI --> BIDIROE BIDIROE --> SerialIn[Serial In] </pre> | <pre> graph LR SPI[SPI] -- "Serial In" --> BIDIROE[BIDIROE] BIDIROE --> SISO[SISO] SISO --> BIDIROE </pre> |

The direction of each serial I/O pin depends on C2[BIDIROE]. 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.

34.5.7 Error conditions

The SPI module has one error condition: the mode fault error.

34.5.7.1 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 C2[MODFEN] is set.

In the special case where the SPI is in master mode and C2[MODFEN] 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.

34.5.8 Low-power mode options

This section describes the low-power mode options.

34.5.8.1 SPI in Run mode

In Run mode, with the SPI system enable (SPE) bit in the SPI Control Register 1 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.

34.5.8.2 SPI in Wait mode

SPI operation in Wait mode depends upon the state of the SPISWAI bit in SPI Control Register 2.

- If C2[SPISWAI] is clear, the SPI operates normally when the CPU is in Wait mode.
- If C2[SPISWAI] is set, SPI clock generation ceases and the SPI module enters a power conservation state when the CPU is in wait mode.
 - If C2[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 C2[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.

34.5.8.3 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 C2[SPISWAI]. 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 reinitialized.

34.5.9 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.

34.5.10 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).

34.5.10.1 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.

34.5.10.2 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.

34.5.10.3 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.

34.5.10.4 SPMF

SPMF occurs when the data in the receive data buffer is equal to the data in the SPI Match Register.

34.5.10.5 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. C1[SPIE] is set to 1.
2. The CPU is in Wait mode—in which case C2[SPISWAI] 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 signals. During the wakeup phase, a continuous transmission from a master would destroy the first received data.

34.6 Initialization/application information

This section discusses an example of how to initialize and use the SPI.

34.6.1 Initialization sequence

Before the SPI module can be used for communication, an initialization procedure must be carried out, as follows:

1. Update the 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 the 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 S[SPTEF] = 1, and then write to the transmit data register (SPIx_D) to begin transfer.

34.6.2 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.

| SPIx_C1=0x54(%01010100) | | | |
|--------------------------------|-------|-----|---|
| 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 |

| SPIx_C2 = 0x80(%10000000) | | | |
|----------------------------------|---------|-----|--------------------------------------|
| Bit 7 | SPMIE | = 1 | SPI hardware match interrupt enabled |
| Bit 6 | | = 0 | Unimplemented |
| Bit 5 | | = 0 | Reserved |
| Bit 4 | MODFEN | = 0 | Disables mode fault function |
| Bit 3 | BIDIROE | = 0 | SPI data I/O pin acts as input |

Table continues on the next page...

Initialization/application information

| SPIx_C2 = 0x80(%10000000) | | | |
|---|---------|------|--|
| Bit 2 | = | 0 | Reserved |
| Bit 1 | SPISWAI | = | SPI clocks operate in wait mode |
| Bit 0 | SPC0 | = | uses separate pins for data input and output |
| SPIx_BR = 0x00(%00000000) | | | |
| Bit 7 | = | 0 | Reserved |
| Bit 6:4 | = | 000 | Sets prescale divisor to 1 |
| Bit 3:0 | = | 0000 | Sets baud rate divisor to 2 |
| SPIx_S = 0x00(%00000000) | | | |
| Bit 7 | SPRF | = | Flag is set when receive data buffer is full |
| Bit 6 | SPMF | = | Flag is set when SPIx_M = receive data buffer |
| Bit 5 | SPTEF | = | Flag is set when transmit data buffer is empty |
| Bit 4 | MODF | = | 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. | | | |

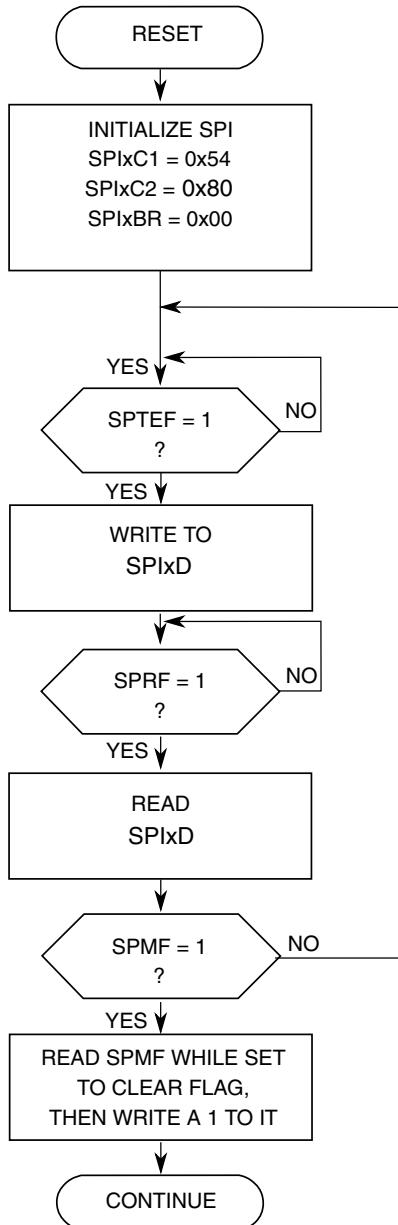


Figure 34-6. Initialization Flowchart Example for SPI Master Device

Chapter 35

Inter-Integrated Circuit (I²C)

35.1 Chip-specific I²C information

This device has one I²C modules. I²C0 is clocked by the system clock so it can support standard I²C communication rates of 100 kbit/s in VLPR mode.

When the package pins associated with I²C have their mux select configured for I²C operation, the pins (SCL and SDA) are driven either by true open drain or in a pseudo open drain configuration.

The digital glitch filter implemented in the I²C0 module, controlled by the I²C0_FLT[FLT] registers, is clocked from the core/system clock and thus has filter granularity in core/system clock cycle counts.

35.2 Introduction

The inter-integrated circuit (I²C, I²C, or IIC) module provides a method of communication between a number of devices.

The interface is designed to operate up to at least 400 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*.

35.2.1 Features

The I²C module has the following features:

- Compatible with *The I²C-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 input glitch filter
- Low power mode wakeup on slave address match
- Range slave address support
- Double buffering support to achieve higher baud rate

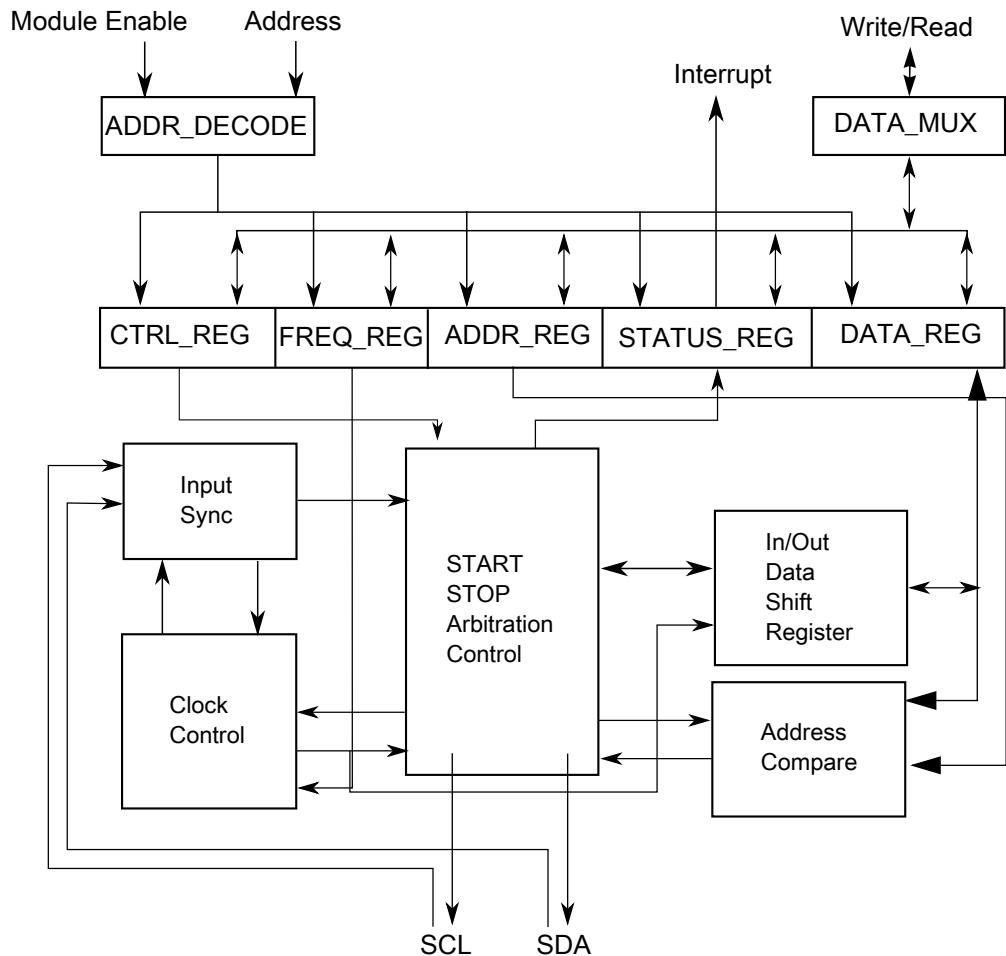
35.2.2 Modes of operation

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.

35.2.3 Block diagram

The following figure is a functional block diagram of the I2C module.

Figure 35-1. I²C Functional block diagram

35.3 I²C signal descriptions

The signal properties of I²C are shown in the table found here.

Table 35-1. I²C signal descriptions

| Signal | Description | I/O |
|--------|---|-----|
| SCL | Bidirectional serial clock line of the I ² C system. | I/O |
| SDA | Bidirectional serial data line of the I ² C system. | I/O |

35.4 Memory map/register definition

This section describes in detail all I²C registers accessible to the end user.

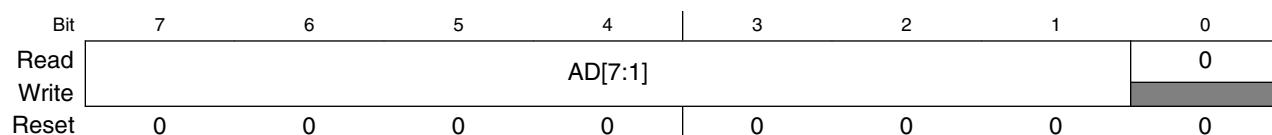
I²C memory map

| Absolute address (hex) | Register name | Width (in bits) | Access | Reset value | Section/page |
|------------------------|--|-----------------|--------|-------------|-----------------------------|
| 4006_6000 | I ² C Address Register 1 (I ² C0_A1) | 8 | R/W | 00h | 35.4.1/544 |
| 4006_6001 | I ² C Frequency Divider register (I ² C0_F) | 8 | R/W | 00h | 35.4.2/545 |
| 4006_6002 | I ² C Control Register 1 (I ² C0_C1) | 8 | R/W | 00h | 35.4.3/546 |
| 4006_6003 | I ² C Status register (I ² C0_S) | 8 | R/W | 80h | 35.4.4/547 |
| 4006_6004 | I ² C Data I/O register (I ² C0_D) | 8 | R/W | 00h | 35.4.5/549 |
| 4006_6005 | I ² C Control Register 2 (I ² C0_C2) | 8 | R/W | 00h | 35.4.6/550 |
| 4006_6006 | I ² C Programmable Input Glitch Filter Register (I ² C0_FLT) | 8 | R/W | 00h | 35.4.7/551 |
| 4006_6007 | I ² C Range Address register (I ² C0_RA) | 8 | R/W | 00h | 35.4.8/552 |
| 4006_6008 | I ² C SMBus Control and Status register (I ² C0_SMB) | 8 | R/W | 00h | 35.4.9/553 |
| 4006_6009 | I ² C Address Register 2 (I ² C0_A2) | 8 | R/W | C2h | 35.4.10/554 |
| 4006_600A | I ² C SCL Low Timeout Register High (I ² C0_SLTH) | 8 | R/W | 00h | 35.4.11/555 |
| 4006_600B | I ² C SCL Low Timeout Register Low (I ² C0_SLTL) | 8 | R/W | 00h | 35.4.12/555 |
| 4006_600C | I ² C Status register 2 (I ² C0_S2) | 8 | R/W | 01h | 35.4.13/556 |

35.4.1 I²C Address Register 1 (I²Cx_A1)

This register contains the slave address to be used by the I²C module.

Address: 4006_6000h base + 0h offset = 4006_6000h



I²Cx_A1 field descriptions

| Field | Description |
|----------------|-------------|
| 7–1 AD[7:1] | Address |

Table continues on the next page...

I2Cx_A1 field descriptions (continued)

| Field | Description |
|---------------|--|
| | 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 Reserved | This field is reserved. This read-only field is reserved and always has the value 0. |

35.4.2 I2C Frequency Divider register (I2Cx_F)

Address: 4006_6000h base + 1h offset = 4006_6001h

| 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 MULT | Multiplier Factor Defines the multiplier factor (mul). This factor is used along with the SCL divider to generate the I2C baud rate. 00 mul = 1 01 mul = 2 10 mul = 4 11 Reserved |
| ICR | ClockRate Prescales the I2C module 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 I2C divider and hold values . The SCL divider multiplied by multiplier factor (mul) determines the I2C baud rate. $\text{I2C baud rate} = \text{I2C module clock speed (Hz)} / (\text{mul} \times \text{SCL divider})$ The SDA hold time is the delay from the falling edge of SCL (I2C clock) to the changing of SDA (I2C data). $\text{SDA hold time} = \text{I2C module clock period (s)} \times \text{mul} \times \text{SDA hold value}$ 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). $\text{SCL start hold time} = \text{I2C module clock period (s)} \times \text{mul} \times \text{SCL start hold value}$ 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). $\text{SCL stop hold time} = \text{I2C module clock period (s)} \times \text{mul} \times \text{SCL stop hold value}$ For example, if the I2C module clock speed is 8 MHz, the following table shows the possible hold time values with different ICR and MULT selections to achieve an I ² C baud rate of 100 kbit/s. |

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 |
| | | | | | |

35.4.3 I2C Control Register 1 (I2Cx_C1)

Address: 4006_6000h base + 2h offset = 4006_6002h

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

I2Cx_C1 field descriptions

| Field | Description |
|------------|--|
| 7 IICEN | I2C Enable Enables I2C module operation. 0 Disabled 1 Enabled |
| 6 IICIE | I2C Interrupt Enable Enables I2C interrupt requests. 0 Disabled 1 Enabled |
| 5 MST | Master Mode Select When MST is changed from 0 to 1, a START signal is generated on the bus and master mode is selected. When this bit changes from 1 to 0, a STOP signal is generated and the mode of operation changes from master to slave. 0 Slave mode 1 Master mode |
| 4 TX | Transmit Mode Select |

Table continues on the next page...

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. 0 Receive 1 Transmit |
| 3 TXAK | Transmit Acknowledge Enable Specifies the value driven onto the SDA during data acknowledge cycles for both master and slave receivers. The value of SMB[FACK] affects NACK/ACK generation. NOTE: SCL is held low until TXAK is written. 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). 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 Writing 1 to this bit generates a repeated START condition provided it is the current master. This bit will always be read as 0. Attempting a repeat at the wrong time results in loss of arbitration. |
| 1 WUEN | Wakeup Enable The I2C module can wake the MCU from low power mode with no peripheral bus running when slave address matching occurs. 0 Normal operation. No interrupt generated when address matching in low power mode. 1 Enables the wakeup function in low power mode. |
| 0 Reserved | This field is reserved. This read-only field is reserved and always has the value 0. |

35.4.4 I2C Status register (I2Cx_S)

Address: 4006_6000h base + 3h offset = 4006_6003h

| 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 TCF | Transfer Complete Flag Acknowledges a byte transfer; TCF is set on the completion of a byte transfer. This bit is valid only during or immediately following a transfer to or from the I2C module. TCF is cleared by reading the I2C data register in receive mode or by writing to the I2C data register in transmit mode. |

Table continues on the next page...

I2Cx_S field descriptions (continued)

| Field | Description |
|--------|---|
| | <p>NOTE: In the buffer mode, TCF is cleared automatically by internal reading or writing the data register I2C_D, with no need waiting for manually reading/writing the I2C data register in Rx/Tx mode.</p> <p>0 Transfer in progress 1 Transfer complete</p> |
| 6 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"> The calling address matches the programmed primary slave address in the A1 register, or matches the range address in the RA register (which must be set to a nonzero value and under the condition I2C_C2[RMEN] = 1). C2[GCAEN] is set and a general call is received. SMB[SIICAEN] is set and the calling address matches the second programmed slave address. ALERTEN is set and an SMBus alert response address is received RMEN is set and an address is received that is within the range between the values of the A1 and RA registers. <p>IAAS 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 1 Addressed as a slave</p> |
| 5 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 1 Bus is busy</p> |
| 4 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 1 to it.</p> <p>0 Standard bus operation. 1 Loss of arbitration.</p> |
| 3 RAM | <p>Range Address Match</p> <p>This bit is set to 1 by any of the following conditions, if I2C_C2[RMEN] = 1:</p> <ul style="list-style-type: none"> Any nonzero calling address is received that matches the address in the RA register. The calling address is within the range of values of the A1 and RA registers. <p>NOTE: For the RAM bit to be set to 1 correctly, C1[IICIE] must be set to 1.</p> <p>Writing the C1 register with any value clears this bit to 0.</p> <p>0 Not addressed 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> |

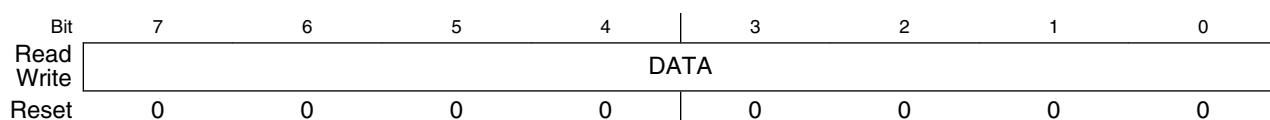
Table continues on the next page...

I2Cx_S field descriptions (continued)

| Field | Description |
|---------|--|
| | <p>0 Slave receive, master writing to slave 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 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"> • 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. • One byte transfer, excluding ACK/NACK bit, completes if FACK is 1. • Match of slave address to calling address including primary slave address, range slave address, alert response address, second slave address, or general call address. • Arbitration lost • In SMBus mode, any timeouts except SCL and SDA high timeouts • I2C bus stop or start detection if the SSIE bit in the Input Glitch Filter register is 1 <p>NOTE: To clear the I2C bus stop or start detection interrupt: In the interrupt service routine, first clear the STOPF or STARTF 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 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 1 No acknowledge signal detected</p> |

35.4.5 I2C Data I/O register (I2Cx_D)

Address: 4006_6000h base + 4h offset = 4006_6004h

**I2Cx_D field descriptions**

| Field | Description |
|-------|---|
| 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>NOTE: 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> |

I2Cx_D field descriptions (continued)

| Field | Description |
|-------|---|
| | <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> |

35.4.6 I2C Control Register 2 (I2Cx_C2)

Address: 4006_6000h base + 5h offset = 4006_6005h

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|-------|-------|-------|---|------|------|---|---|---|
| Read | GCAEN | ADEXT | 0 | SBRC | RMEN | | | |
| Write | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

I2Cx_C2 field descriptions

| Field | Description |
|------------|---|
| 7 GCAEN | <p>General Call Address Enable</p> <p>Enables general call address.</p> <p>0 Disabled 1 Enabled</p> |
| 6 ADEXT | <p>Address Extension</p> <p>Controls the number of bits used for the slave address.</p> <p>0 7-bit address scheme 1 10-bit address scheme</p> |
| 5 Reserved | <p>This field is reserved.</p> <p>This read-only field is reserved and always has the value 0.</p> |
| 4 SBRC | <p>Slave Baud Rate Control</p> <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 kbit/s but the slave can capture the master's data at only 10 kbit/s.</p> <p>0 The slave baud rate follows the master baud rate and clock stretching may occur 1 Slave baud rate is independent of the master baud rate</p> |
| 3 RMEN | <p>Range Address Matching Enable</p> <p>This bit controls the slave address matching for addresses between the values of the A1 and RA registers. When this bit is set, a slave address matching 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> |

Table continues on the next page...

I2Cx_C2 field descriptions (continued)

| Field | Description |
|----------|---|
| | <p>0 Range mode disabled. No address matching occurs for an address within the range of values of the A1 and RA registers.</p> <p>1 Range mode enabled. Address matching occurs when a slave receives an address within the range of values of the A1 and RA registers.</p> |
| AD[10:8] | <p>Slave Address</p> <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> |

35.4.7 I2C Programmable Input Glitch Filter Register (I2Cx_FLT)

Address: 4006_6000h base + 6h offset = 4006_6006h

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|-------|------|-------|------|--------|---|---|---|---|
| Read | SHEN | STOPF | SSIE | STARTF | | | | |
| Write | w1c | | | w1c | | | | |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

I2Cx_FLT field descriptions

| Field | Description |
|------------|---|
| 7 SHEN | <p>Stop Hold Enable</p> <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"> 1. The I2C module is configured for a basic transfer, and the SHEN bit is set to 1. 2. A transfer begins. 3. The MCU signals the I2C module to enter stop mode. 4. The byte currently being transferred, including both address and data, completes its transfer. 5. The I2C slave or master acknowledges that the in-transfer byte completed its transfer and acknowledges the request to enter stop mode. 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. <p>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.</p> <p>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.</p> <p>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.</p> <p>0 Stop holdoff is disabled. The MCU's entry to stop mode is not gated. 1 Stop holdoff is enabled.</p> |
| 6 STOPF | I2C Bus Stop Detect Flag |

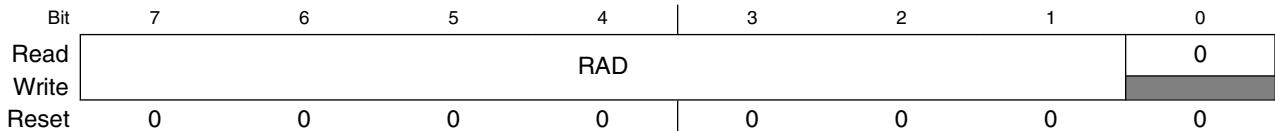
Table continues on the next page...

I2Cx_FLT field descriptions (continued)

| Field | Description |
|----------|--|
| | <p>Hardware sets this bit when the I2C bus's stop status is detected. The STOPF bit must be cleared by writing 1 to it.</p> <p>0 No stop happens on I2C bus 1 Stop detected on I2C bus</p> |
| 5 SSIE | <p>I2C Bus Stop or Start Interrupt Enable</p> <p>This bit enables the interrupt for I2C bus stop or start detection.</p> <p>NOTE: To clear the I2C bus stop or start detection interrupt: In the interrupt service routine, first clear the STOPF or STARTF 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.</p> <p>0 Stop or start detection interrupt is disabled 1 Stop or start detection interrupt is enabled</p> |
| 4 STARTF | <p>I2C Bus Start Detect Flag</p> <p>Hardware sets this bit when the I2C bus's start status is detected. The STARTF bit must be cleared by writing 1 to it.</p> <p>0 No start happens on I2C bus 1 Start detected on I2C bus</p> |
| FLT | <p>I2C Programmable Filter Factor</p> <p>Controls the width of the glitch, in terms of I2C module 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.</p> <p>0h No filter/bypass 1-Fh Filter glitches up to width of n I2C module clock cycles, where $n=1-15d$</p> |

35.4.8 I2C Range Address register (I2Cx_RA)

Address: 4006_6000h base + 7h offset = 4006_6007h

**I2Cx_RA field descriptions**

| Field | Description |
|------------|--|
| 7-1 RAD | <p>Range Slave Address</p> <p>This field contains the slave address to be used by the I2C module. The field is used in the 7-bit address scheme. If I2C_C2[RMEN] is set to 1, any nonzero value write enables this register. This register value can be considered as a maximum boundary in the range matching mode.</p> |
| 0 Reserved | <p>This field is reserved.</p> <p>This read-only field is reserved and always has the value 0.</p> |

35.4.9 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 is set to 1 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: 4006_6000h base + 8h offset = 4006_6008h

| 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 FACK | <p>Fast NACK/ACK Enable</p> <p>For SMBus packet error checking, the CPU must be able to issue an ACK or NACK according to the result of receiving data byte.</p> <p>0 An ACK or NACK is sent on the following receiving data byte 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.</p> |
| 6 ALERTEN | <p>SMBus Alert Response Address Enable</p> <p>Enables or disables SMBus alert response address matching.</p> <p>NOTE: 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.</p> <p>0 SMBus alert response address matching is disabled 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 1 I2C address register 2 matching is enabled</p> |

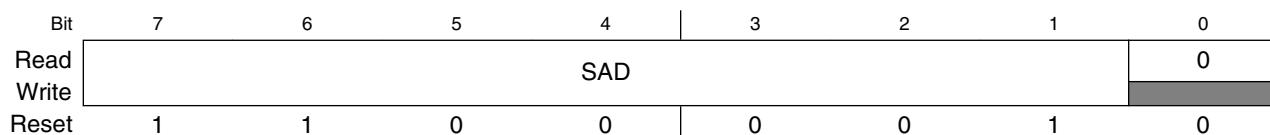
Table continues on the next page...

I2Cx_SMB field descriptions (continued)

| Field | Description |
|--------------|---|
| 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 I2C module clock / 64 1 Timeout counter counts at the frequency of the I2C module 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>NOTE: The low timeout function is disabled when the SLT register's value is 0.</p> <p>0 No low timeout occurs 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 $\text{clock} \times \text{LoValue} / 512$, which indicates the bus is free. This bit is cleared automatically.</p> <p>0 No SCL high and SDA high timeout occurs 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 $\text{clock} \times \text{LoValue} / 512$. Software clears this bit by writing 1 to it.</p> <p>0 No SCL high and SDA low timeout occurs 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 1 SHTF2 interrupt is enabled</p> |

35.4.10 I2C Address Register 2 (I2Cx_A2)

Address: 4006_6000h base + 9h offset = 4006_6009h

**I2Cx_A2 field descriptions**

| Field | Description |
|------------|---------------|
| 7–1 SAD | SMBus Address |

Table continues on the next page...

I2Cx_A2 field descriptions (continued)

| Field | Description |
|---------------|--|
| | Contains the slave address used by the SMBus. This field is used on the device default address or other related addresses. |
| 0 Reserved | This field is reserved. This read-only field is reserved and always has the value 0. |

35.4.11 I2C SCL Low Timeout Register High (I2Cx_SLTH)

Address: 4006_6000h base + Ah offset = 4006_600Ah

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|------------|------------|---|---|---|---|---|---|---|
| Read Write | SSLT[15:8] | | | | | | | |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

I2Cx_SLTH field descriptions

| Field | Description |
|------------|---|
| SSLT[15:8] | SSLT[15:8] Most significant byte of SCL low timeout value that determines the timeout period of SCL low. |

35.4.12 I2C SCL Low Timeout Register Low (I2Cx_SLTL)

Address: 4006_6000h base + Bh offset = 4006_600Bh

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|------------|-----------|---|---|---|---|---|---|---|
| Read Write | SSLT[7:0] | | | | | | | |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

I2Cx_SLTL field descriptions

| Field | Description |
|-----------|---|
| SSLT[7:0] | SSLT[7:0] Least significant byte of SCL low timeout value that determines the timeout period of SCL low. |

35.4.13 I2C Status register 2 (I2Cx_S2)

Address: 4006_6000h base + Ch offset = 4006_600Ch

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|-------|---|---|---|---|---|---|-------|-------|
| Read | 0 | 0 | 0 | 0 | 0 | 0 | ERROR | EMPTY |
| Write | | | | | | | w1c | |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 1 |

I2Cx_S2 field descriptions

| Field | Description |
|---------------|--|
| 7 Reserved | This field is reserved. This read-only field is reserved and always has the value 0. |
| 6 Reserved | This field is reserved. This read-only field is reserved and always has the value 0. |
| 5 Reserved | This field is reserved. This read-only field is reserved and always has the value 0. |
| 4 Reserved | This field is reserved. This read-only field is reserved and always has the value 0. |
| 3 Reserved | This field is reserved. This read-only field is reserved and always has the value 0. |
| 2 Reserved | This field is reserved. This read-only field is reserved and always has the value 0. |
| 1 ERROR | Error flag Indicates if there are read or write errors with the Tx and Rx buffers. 0 The buffer is not full and all write/read operations have no errors. 1 There are 3 or more write/read errors during the data transfer phase (when the Empty flag is not set and the buffer is busy). |
| 0 EMPTY | Empty flag Indicates if the Tx or Rx buffer is empty. 0 Tx or Rx buffer is not empty and cannot be written to, that is new data cannot be loaded into the buffer. 1 Tx or Rx buffer is empty and can be written to, that is new data can be loaded into the buffer. |

35.5 Functional description

This section provides a comprehensive functional description of the I2C module.

35.5.1 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.

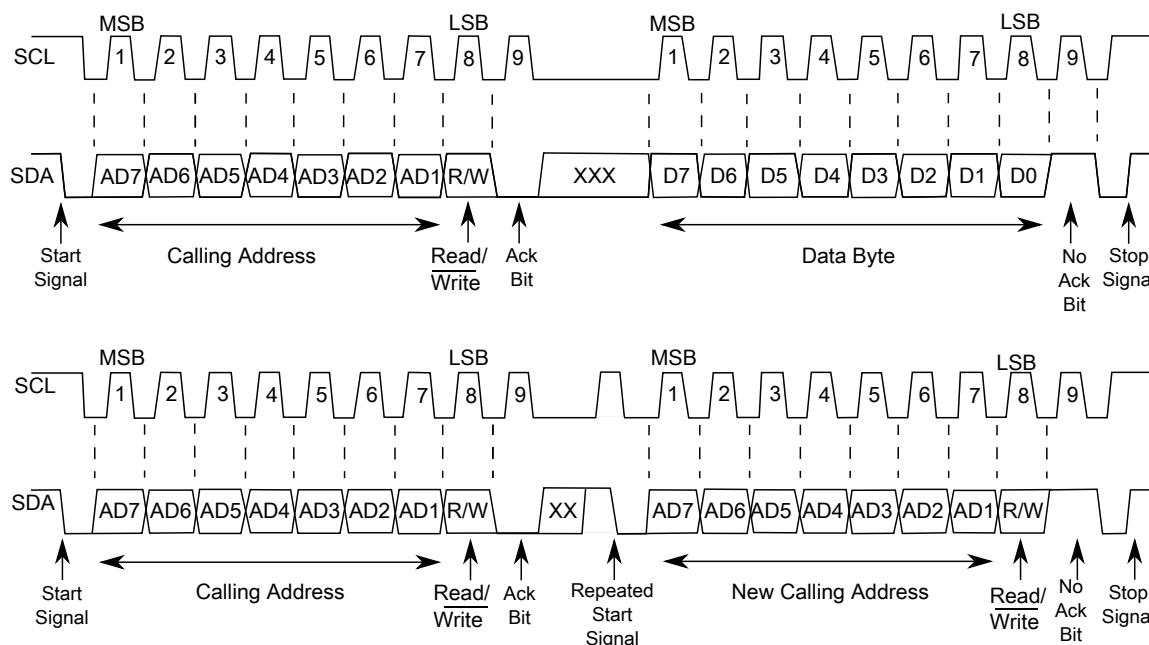


Figure 35-2. I2C bus transmission signals

35.5.1.1 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.

35.5.1.2 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.

35.5.1.3 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.

35.5.1.4 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.

35.5.1.5 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. The master needs to send a NACK signal before sending repeated-START in the buffering mode.

35.5.1.6 Arbitration procedure

The I2C 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.

35.5.1.7 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.

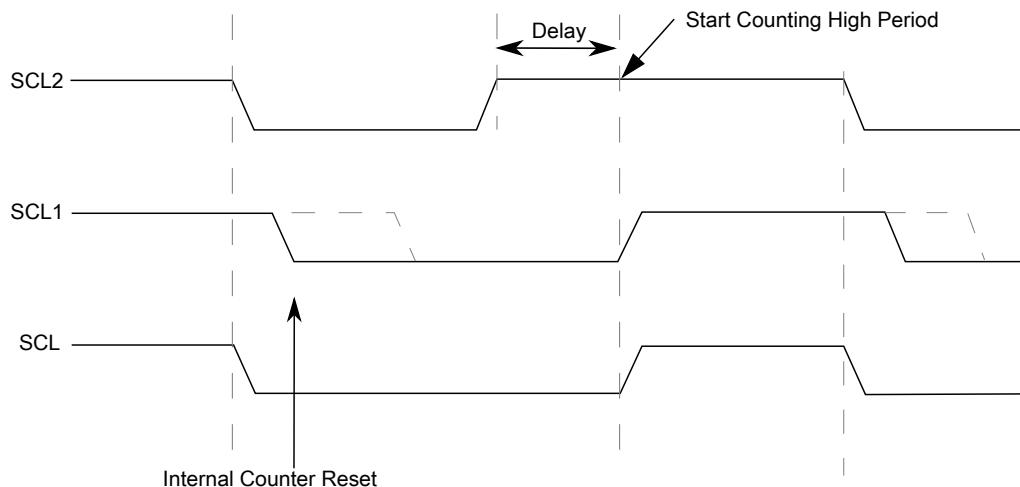


Figure 35-3. I2C clock synchronization

35.5.1.8 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.

35.5.1.9 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.

35.5.1.10 I2C 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 I2C module.

Table 35-2. I2C divider and hold values

| ICR (hex) | SCL divider | SDA hold value | SCL hold (start) value | SCL hold (stop) value | ICR (hex) | SCL divider (clocks) | SDA hold (clocks) | SCL hold (start) value | SCL hold (stop) 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 35-2. I2C divider and hold values (continued)

| ICR (hex) | SCL divider | SDA hold value | SCL hold (start) value | SCL hold (stop) value | ICR (hex) | SCL divider (clocks) | SDA hold (clocks) | SCL hold (start) value | SCL hold (stop) 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 |

35.5.2 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.

35.5.2.1 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/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 35-3. Master-transmitter addresses slave-receiver with a 10-bit address

| | | | | | | | | | | | |
|---|--|----------|----|--------------------------------------|----|------|---|-----|------|-----|---|
| S | Slave address first 7 bits 11110 + AD10 + AD9 | R/W 0 | A1 | Slave address second byte 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.

35.5.2.2 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 35-4. Master-receiver addresses a slave-transmitter with a 10-bit address

| S | Slave address first 7 bits 11110 + AD10 + AD9 | R/W 0 | A1 | Slave address second byte AD[8:1] | A2 | Sr | Slave address first 7 bits 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.

35.5.3 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 RMEN bit is set, when the Range Address register is programmed to a nonzero value, any address within the range of values of Address Register 1 (excluded) and the Range Address register (included) 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.

35.5.4 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.

35.5.4.1 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 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.

35.5.4.1.1 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.

35.5.4.1.2 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.

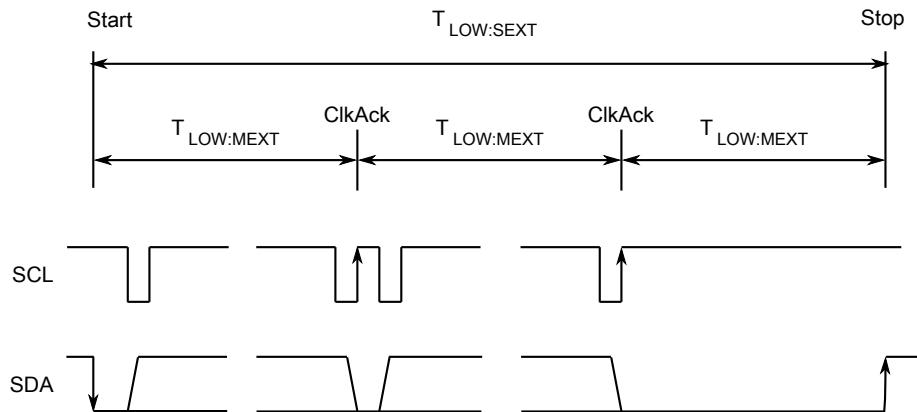
A HIGH timeout occurs after a START condition appears on the bus but before a STOP condition appears on the bus. Any master detecting this scenario can assume the bus is free when either of the following occurs:

- 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.

35.5.4.1.3 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 35-4. 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.

35.5.4.2 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.

35.5.5 Resets

The I2C module is disabled after a reset. The I2C module cannot cause a core reset.

35.5.6 Interrupts

The I2C module generates an interrupt when any of the events in the table found here 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 35-5. 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 ² C bus stop detection | STOPF | IICIF | IICIE & SSIE |
| I ² C bus start detection | STARTF | IICIF | IICIE & SSIE |
| 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 |

35.5.6.1 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.

35.5.6.2 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.

35.5.6.3 Stop Detect Interrupt

When the stop status is detected on the I²C bus, the SSIE bit is set to 1. The CPU is interrupted, provided the IICIE and SSIE bits are both set to 1.

35.5.6.4 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.

35.5.6.5 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.

35.5.6.6 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.

35.5.7 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) I2C module 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 I2C module clock cycles) for the filter to absorb and not pass.

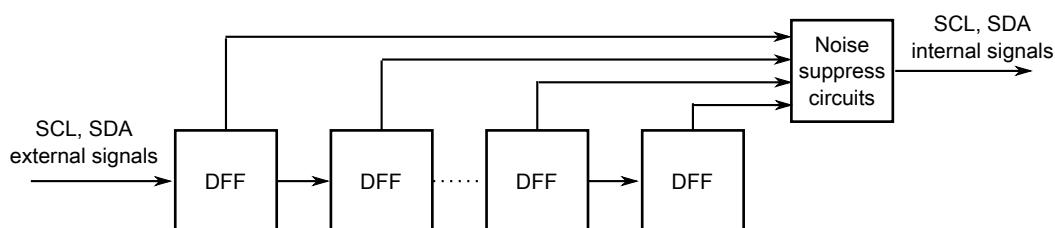


Figure 35-5. Programmable input glitch filter diagram

35.5.8 Address matching wake-up

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 where no peripheral bus is 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. To avoid I2C transfer problems resulting from the 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).

NOTE

After I2C address matching wake-up, the master must wait a time long enough for the slave ISR to finish running.

For the SRW bit to function properly, it only supports Address +Write to wake up by I2C address matching. Before entering the next low power mode, Address+Write must be sent to change the SRW status.

35.5.9 Double buffering mode

In the double buffering mode, the data transfer is processed byte by byte. However, the data can be transferred without waiting for the interrupt or the polling to finish. This means the write/read I2C_D operation will not block the data transfer, as the hardware has already finished the internal write or read. The benefit is that the baud rate is able to achieve higher speed.

There are several items to consider as follows:

- When initiating a double buffering transfer at Tx side, the user can write 2 values to the I2C_D buffer before transfer. However, that is allowed only at one time per package frame (due to the buffer depth, and because two-times writes in each ISR are

not allowed). The second write to the I2C_D buffer must wait for the Empty flag. On the other hand, at Rx side the user can read twice in a one-byte transfer (if needed).

NOTE

Check Empty flag before write to I2C_D.

Write twice to the I2C_D buffer ONLY after the address matching byte. Do not write twice (Address+Data) before START or at the beginning of I2C transfer, especially when the baud rate is very slow.

- To write twice in one frame, during the next-to-last ISR, do a dummy read from the I2C_D buffer at Tx side (or the TCF will stay high, because the TCF is cleared by write/read operation). In the next-to-last ISR, do not send data again (the buffer data will be under running).
- To keep new ISRs software-compatible with previous ISRs, the write/read I2C_D operation will not block the internal-hardware-released SCL/SDA signals. At the ACK phase, the bus is released to accept the next byte if the master can send the clock immediately.
- On the slave side, two-times writes to the I2C_D buffer may be limited by the master's clock and START/repeated-START signal. This is not currently supported, and the master's START/repeated-START signal will break data transfers. To release the bus, do a dummy read or write to the I2C_D buffer again. It is suggested to send repeated-START/START during intervals as before.
- The master receive should send a NACK in the next-to-last ISR, if it wants to do the STOP or the repeated-START work. The transmitting slave which receives the NACK, will switch to receive mode, and do a dummy read to release SCL and SDA signals.

35.6 Initialization/application information

Module Initialization (Slave)

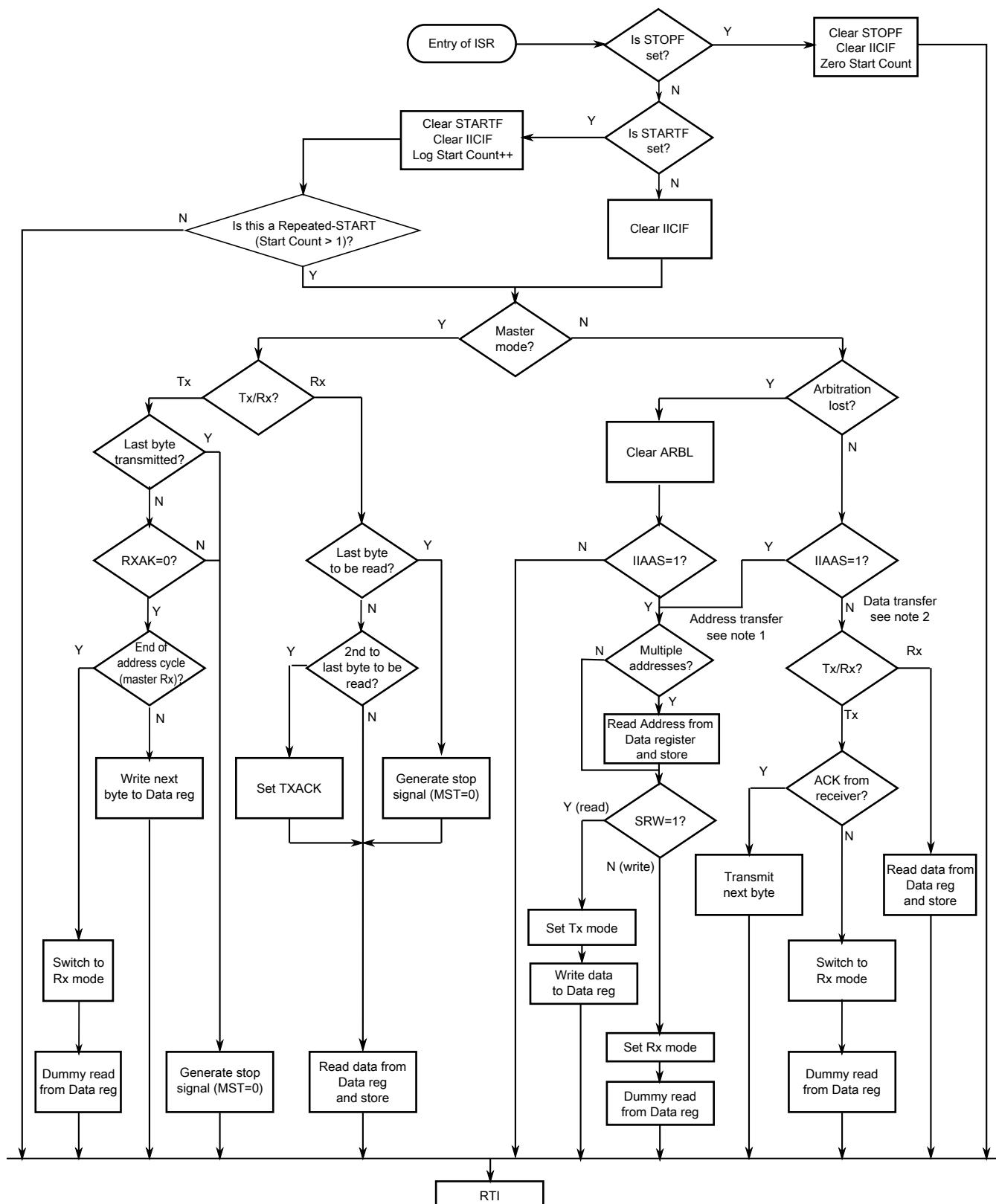
1. Write: Control Register 2
 - to enable or disable general call
 - to select 10-bit or 7-bit addressing mode
2. Write: Address Register 1 to set the slave address
3. Write: Control Register 1 to enable the I2C module and interrupts
4. Initialize RAM variables (IICEN = 1 and IICIE = 1) for transmit 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)

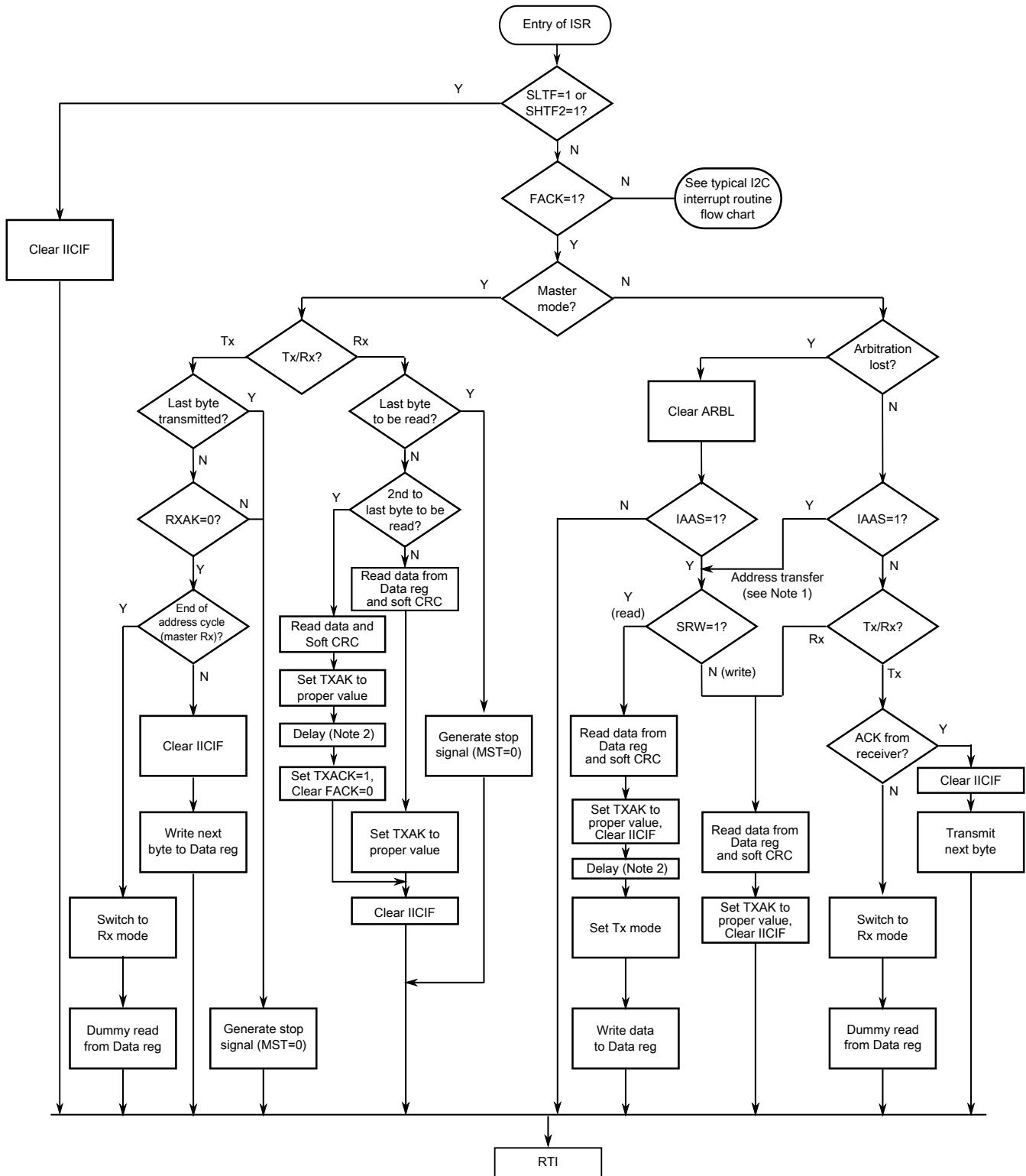
The routine shown in the following figure encompasses both master and slave I2C operations. For slave operation, an incoming I2C message that contains the proper address begins I2C communication. For master operation, communication must be initiated by writing the Data register. An example of an I2C driver which implements many of the steps described here is available in [AN4342: Using the Inter-Integrated Circuit on ColdFire+ and Kinetis](#).



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 35-6. Typical I2C interrupt routine



Notes:

- 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.
- In receive mode, one bit time delay may be needed before the first and second data reading, to wait for the possible longest time period (in worst case) of the 9th SCL cycle.

Figure 35-7. Typical I2C SMBus interrupt routine

Chapter 36

Low Power Universal Asynchronous Receiver/ Transmitter (LPUART)

36.1 Chip-specific LPUART information

The LPUART0 module supports basic UART, 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 SIM_SOPT5[LPUART0ODE].

36.2 Introduction

36.2.1 Features

Features of the LPUART module include:

- Full-duplex, standard non-return-to-zero (NRZ) format
- Programmable baud rates (13-bit modulo divider) with configurable oversampling ratio from 4x to 32x
- 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
- Interrupt 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
- Receive data match
- Hardware parity generation and checking
- Programmable 8-bit, 9-bit or 10-bit character length
- Programmable 1-bit or 2-bit stop bits
- Three receiver wakeup methods:
 - Idle line wakeup
 - Address mark wakeup
 - Receive data match
- Automatic address matching to reduce ISR overhead:
 - Address mark matching
 - Idle line address matching
 - Address match start, address match end
- Optional 13-bit break character generation / 11-bit break character detection
- Configurable idle length detection supporting 1, 2, 4, 8, 16, 32, 64 or 128 idle characters
- Selectable transmitter output and receiver input polarity

36.2.2 Modes of operation

36.2.2.1 Stop mode

The LPUART will remain functional during Stop mode, provided the asynchronous transmit and receive clock remains enabled. The LPUART can generate an interrupt to cause a wakeup from Stop mode.

NOTE

Before the MCU enters Stop, Wait, or VLPS power mode with the CTRL[DOZEEN]=1b or enters the LLS or VLLS power mode, enable the pullup resistor on the LPUART_TX pin to prevent the pin from floating. If the CTRL[TXINV]=1b, enable the pulldown resistor on the LPUART_TX pin.

36.2.2.2 Wait mode

The LPUART 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.

36.2.2.3 Debug mode

The LPUART remains functional in debug mode.

36.2.3 Signal Descriptions

| Signal | Description | I/O |
|-----------|---|-----|
| LPUART_TX | Transmit data. This pin is normally an output, but is an input (tristated) in single wire mode whenever the transmitter is disabled or transmit direction is configured for receive data. | I/O |
| LPUART_RX | Receive data. | I |

36.2.4 Block diagram

The following figure shows the transmitter portion of the LPUART.

Introduction

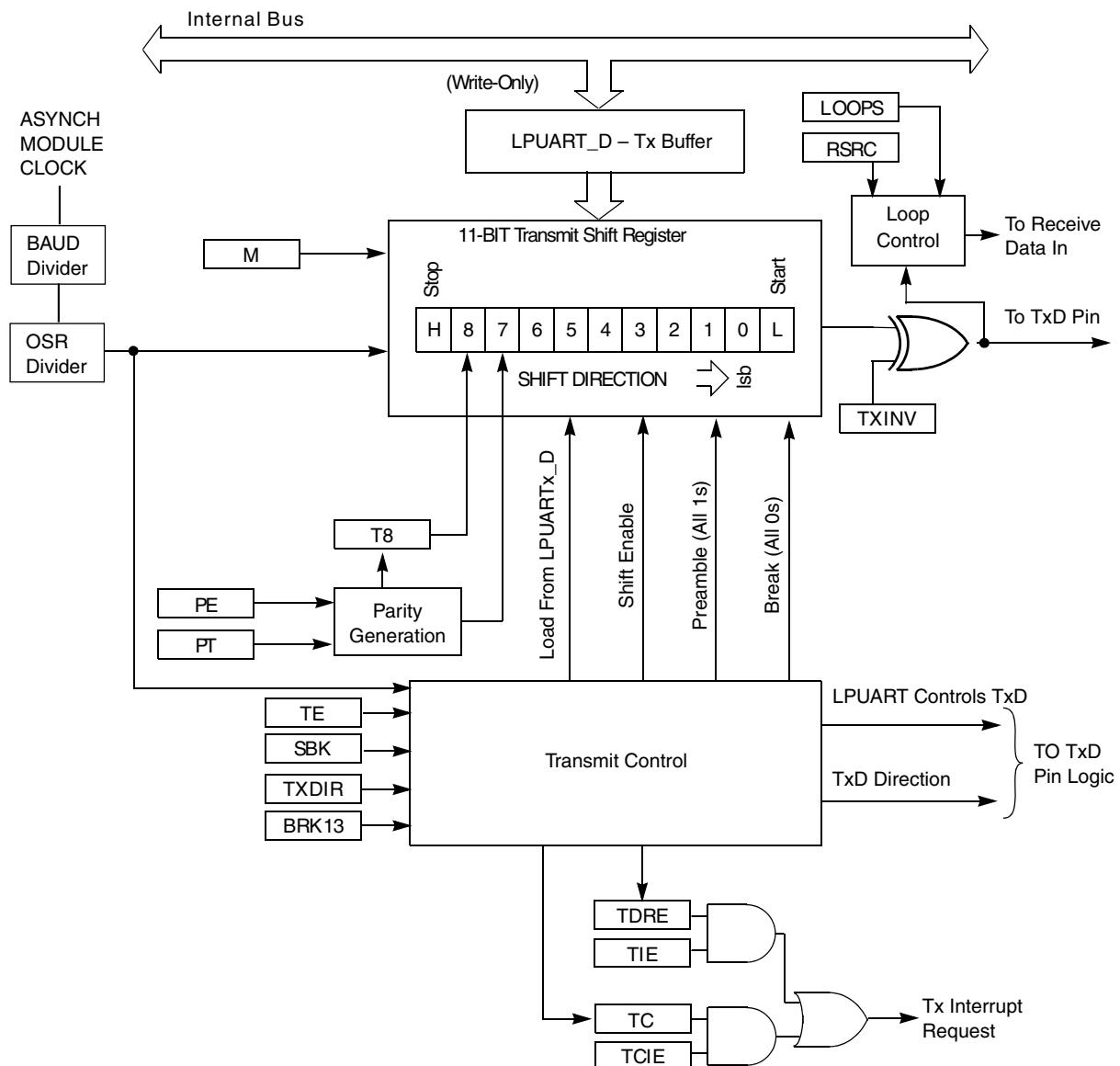


Figure 36-1. LPUART transmitter block diagram

The following figure shows the receiver portion of the LPUART.

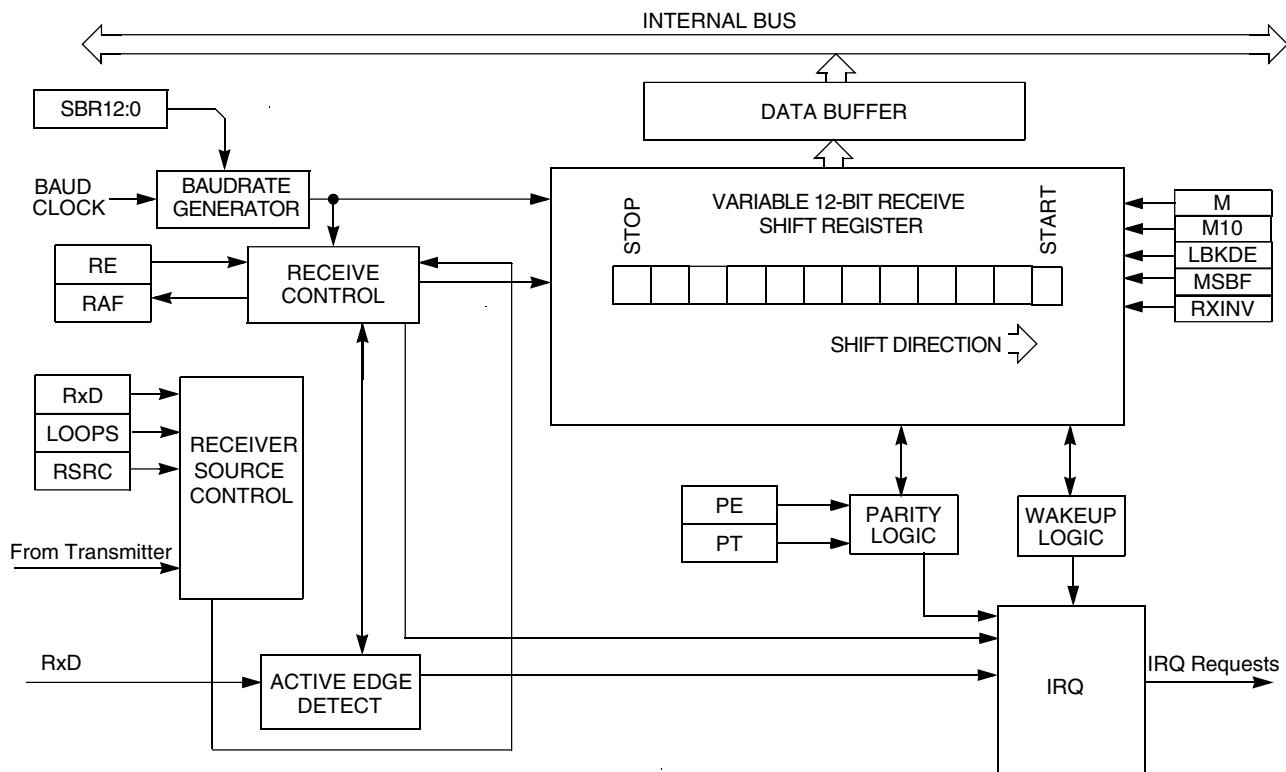


Figure 36-2. LPUART receiver block diagram

36.3 Register definition

The LPUART includes registers to control baud rate, select LPUART options, report LPUART status, and for transmit/receive data. Accesses to address outside the valid memory map will generate a bus error.

LPUART memory map

| Absolute address (hex) | Register name | Width (in bits) | Access | Reset value | Section/ page |
|------------------------|---|-----------------|--------|-------------|---------------|
| 4005_4000 | LPUART Baud Rate Register (LPUART0_BAUD) | 32 | R/W | 0F00_0004h | 36.3.1/580 |
| 4005_4004 | LPUART Status Register (LPUART0_STAT) | 32 | R/W | 00C0_0000h | 36.3.2/582 |
| 4005_4008 | LPUART Control Register (LPUART0_CTRL) | 32 | R/W | 0000_0000h | 36.3.3/586 |
| 4005_400C | LPUART Data Register (LPUART0_DATA) | 32 | R/W | 0000_1000h | 36.3.4/591 |
| 4005_4010 | LPUART Match Address Register (LPUART0_MATCH) | 32 | R/W | 0000_0000h | 36.3.5/593 |

36.3.1 LPUART Baud Rate Register (LPUARTx_BAUD)

Address: 4005_4000h base + 0h offset = 4005_4000h

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|-------|---------|----------|------|-----|----|----|----|-----|----|----|----|----|--------|----|-----------|----|
| R | MAEN1 | MAEN2 | M10 | OSR | | | | | 0 | 0 | 0 | 0 | MATCFG | | BOTHEdge | |
| W | | | | | | | | | | | | | | | RESYNCDis | |
| Reset | 0 | 0 | 0 | 0 | 1 | 1 | 1 | 1 | 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 | LBKDIEn | RXEDGEIE | SBNS | | | | | SBR | | | | | | | | |
| W | | | | | | | | | | | | | | | | |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 1 | 0 | 0 |

LPUARTx_BAUD field descriptions

| Field | Description |
|-------------|--|
| 31 MAEN1 | Match Address Mode Enable 1 0 Normal operation. 1 Enables automatic address matching or data matching mode for MATCH[MA1]. |
| 30 MAEN2 | Match Address Mode Enable 2 0 Normal operation. 1 Enables automatic address matching or data matching mode for MATCH[MA2]. |
| 29 M10 | 10-bit Mode select 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. 0 Receiver and transmitter use 8-bit or 9-bit data characters. 1 Receiver and transmitter use 10-bit data characters. |
| 28–24 OSR | Over Sampling Ratio 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. |
| 23 Reserved | This field is reserved. This read-only field is reserved and always has the value 0. |
| 22 Reserved | This field is reserved. This read-only field is reserved and always has the value 0. |
| 21 Reserved | This field is reserved. This read-only field is reserved and always has the value 0. |

Table continues on the next page...

LPUARTx_BAUD field descriptions (continued)

| Field | Description |
|-----------------|--|
| 20 Reserved | This field is reserved. This read-only field is reserved and always has the value 0. |
| 19–18 MATCFG | Match Configuration Configures the match addressing mode used. 00 Address Match Wakeup 01 Idle Match Wakeup 10 Match On and Match Off 11 Enables RWU on Data Match and Match On/Off for transmitter CTS input |
| 17 BOTHEDGE | Both Edge Sampling 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. 0 Receiver samples input data using the rising edge of the baud rate clock. 1 Receiver samples input data using the rising and falling edge of the baud rate clock. |
| 16 RESYNCDIS | Resynchronization Disable 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. 0 Resynchronization during received data word is supported 1 Resynchronization during received data word is disabled |
| 15 LBKDIIE | LIN Break Detect Interrupt Enable LBKDIIE enables the LIN break detect flag, LBKDIF, to generate interrupt requests. 0 Hardware interrupts from LPUART_STAT[LBKDIF] disabled (use polling). 1 Hardware interrupt requested when LPUART_STAT[LBKDIF] flag is 1. |
| 14 RXEDGIE | RX Input Active Edge Interrupt Enable Enables the receive input active edge, RXEDGIF, to generate interrupt requests. Changing CTRL[LOOP] or CTRL[RSRC] when RXEDGIE is set can cause the RXEDGIF to set. 0 Hardware interrupts from LPUART_STAT[RXEDGIF] disabled (use polling). 1 Hardware interrupt requested when LPUART_STAT[RXEDGIF] flag is 1. |
| 13 SBNS | Stop Bit Number Select 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. 0 One stop bit. 1 Two stop bits. |
| SBR | Baud Rate Modulo Divisor. The 13 bits in SBR[12:0] set the modulo divide rate for the baud rate generator. When SBR is 1 - 8191, the baud rate equals "baud clock / ((OSR+1) x SBR)". The 13-bit baud rate setting [SBR12:SBR0] must only be updated when the transmitter and receiver are both disabled (LPUART_CTRL[RE] and LPUART_CTRL[TE] are both 0). |

36.3.2 LPUART Status Register (LPUARTx_STAT)

Address: 4005_4000h base + 4h offset = 4005_4004h

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|-------|--------|---------|------|-------|-------|-------|-------|-----|------|----|------|------|-----|-----|-----|-----|
| R | LBKDIF | RXEDGIF | MSBF | RXINV | RWUID | BRK13 | LBKDE | RAF | TDRE | TC | RDRF | IDLE | OR | NF | FE | PF |
| W | w1c | w1c | | | | | | | | | | w1c | w1c | w1c | w1c | w1c |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 1 | 1 | 0 | 0 | 0 | 0 | 0 | 0 |
| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
| R | MA1F | MA2F | | | | | | | 0 | | | | | | | |
| W | w1c | w1c | | | | | | | | | | | | | | |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

LPUARTx_STAT field descriptions

| Field | Description |
|---------------|--|
| 31 LBKDIF | <p>LIN Break Detect Interrupt Flag</p> <p>LBKDIF is set when the LIN break detect circuitry is enabled and a LIN break character is detected. LBKDIF is cleared by writing a 1 to it.</p> <p>0 No LIN break character has been detected. 1 LIN break character has been detected.</p> |
| 30 RXEDGIF | <p>LPUART_RX Pin Active Edge Interrupt Flag</p> <p>RXEDGIF is set when an active edge, falling if RXINV = 0, rising if RXINV=1, on the LPUART_RX pin occurs. RXEDGIF is cleared by writing a 1 to it.</p> <p>0 No active edge on the receive pin has occurred. 1 An active edge on the receive pin has occurred.</p> |
| 29 MSBF | MSB First |

Table continues on the next page...

LPUARTx_STAT field descriptions (continued)

| Field | Description |
|----------|---|
| | <p>Setting this bit reverses the order of the bits that are transmitted and received on the wire. This bit does not affect the polarity of the bits, the location of the parity bit or the location of the start or stop bits. This bit should only be changed when the transmitter and receiver are both disabled.</p> <ul style="list-style-type: none"> 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. 1 MSB (bit9, bit8, bit7 or bit6) is the first bit that is transmitted following the start bit depending on the setting of CTRL[M], CTRL[PE] and BAUD[M10]. Further, the first bit received after the start bit is identified as bit9, bit8, bit7 or bit6 depending on the setting of CTRL[M] and CTRL[PE]. |
| 28 RXINV | <p>Receive Data Inversion</p> <p>Setting this bit reverses the polarity of the received data input.</p> <p>NOTE: Setting RXINV inverts the LPUART_RX input for all cases: data bits, start and stop bits, break, and idle.</p> <ul style="list-style-type: none"> 0 Receive data not inverted. 1 Receive data inverted. |
| 27 RWUID | <p>Receive Wake Up Idle Detect</p> <p>For RWU on idle character, RWUID controls whether the idle character that wakes up the receiver sets the IDLE bit. For address match wakeup, RWUID controls if the IDLE bit is set when the address does not match. This bit should only be changed when the receiver is disabled.</p> <ul style="list-style-type: none"> 0 During receive standby state (RWU = 1), the IDLE bit does not get set upon detection of an idle character. During address match wakeup, the IDLE bit does not get set when an address does not match. 1 During receive standby state (RWU = 1), the IDLE bit gets set upon detection of an idle character. During address match wakeup, the IDLE bit does get set when an address does not match. |
| 26 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> <ul style="list-style-type: none"> 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). 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). |
| 25 LBKDE | <p>LIN Break Detection Enable</p> <p>LBKDE selects a longer break character detection length. While LBKDE is set, receive data is not stored in the receive data buffer.</p> <ul style="list-style-type: none"> 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). 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). |
| 24 RAF | <p>Receiver Active Flag</p> <p>RAF is set when the receiver detects the beginning of a valid start bit, and RAF is cleared automatically when the receiver detects an idle line.</p> |

Table continues on the next page...

LPUARTx_STAT field descriptions (continued)

| Field | Description |
|------------|---|
| | <p>0 LPUART receiver idle waiting for a start bit. 1 LPUART receiver active (LPUART_RX input not idle).</p> |
| 23 TDRE | <p>Transmit Data Register Empty Flag TDRE will set when the transmit data register (LPUART_DATA) is empty. To clear TDRE, write to the LPUART data register (LPUART_DATA). TDRE is not affected by a character that is in the process of being transmitted, it is updated at the start of each transmitted character.</p> <p>0 Transmit data buffer full. 1 Transmit data buffer empty.</p> |
| 22 TC | <p>Transmission Complete Flag TC is cleared when there is a transmission in progress or when a preamble or break character is loaded. TC is set when the transmit buffer is empty and no data, preamble, or break character is being transmitted. When TC is set, the transmit data output signal becomes idle (logic 1). TC is cleared by writing to LPUART_DATA to transmit new data, queuing a preamble by clearing and then setting LPUART_CTRL[TE], queuing a break character by writing 1 to LPUART_CTRL[SBK].</p> <p>0 Transmitter active (sending data, a preamble, or a break). 1 Transmitter idle (transmission activity complete).</p> |
| 21 RDRF | <p>Receive Data Register Full Flag RDRF is set when the receive buffer (LPUART_DATA) is full. To clear RDRF, read the LPUART_DATA register. A character that is in the process of being received does not cause a change in RDRF until the entire character is received. Even if RDRF is set, the character will continue to be received until an overrun condition occurs once the entire character is received.</p> <p>0 Receive data buffer empty. 1 Receive data buffer full.</p> |
| 20 IDLE | <p>Idle Line Flag IDLE is set when the LPUART 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. 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 stored in the receive buffer or a LIN break character has set the LBKDIF flag . IDLE is set only once even if the receive line remains idle for an extended period.</p> <p>0 No idle line detected. 1 Idle line was detected.</p> |
| 19 OR | <p>Receiver Overrun Flag OR is set when software fails to prevent the receive data register from overflowing with data. The OR bit is set immediately after the stop bit has been completely received for the dataword that overflows the buffer and all the other error flags (FE, NF, and PF) are prevented from setting. The data in the shift register is</p> |

Table continues on the next page...

LPUARTx_STAT field descriptions (continued)

| Field | Description |
|------------|--|
| | <p>lost, but the data already in the LPUART data registers is not affected. If LBKDE is enabled and a LIN Break is detected, the OR field asserts if LBKDIF is not cleared before the next data character is received.</p> <p>While the OR flag is set, no additional data is stored in the data buffer even if sufficient room exists. To clear OR, write logic 1 to the OR flag.</p> <ul style="list-style-type: none"> 0 No overrun. 1 Receive overrun (new LPUART data lost). |
| 18 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 then noise is detected for that character. NF is set whenever the next character to be read from LPUART_DATA was received with noise detected within the character. To clear NF, write logic one to the NF.</p> <ul style="list-style-type: none"> 0 No noise detected. 1 Noise detected in the received character in LPUART_DATA. |
| 17 FE | <p>Framing Error Flag</p> <p>FE is set whenever the next character to be read from LPUART_DATA was received with logic 0 detected where a stop bit was expected. To clear NF, write logic one to the NF.</p> <ul style="list-style-type: none"> 0 No framing error detected. This does not guarantee the framing is correct. 1 Framing error. |
| 16 PF | <p>Parity Error Flag</p> <p>PF is set whenever the next character to be read from LPUART_DATA was received 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> <ul style="list-style-type: none"> 0 No parity error. 1 Parity error. |
| 15 MA1F | <p>Match 1 Flag</p> <p>MA1F is set whenever the next character to be read from LPUART_DATA matches MA1. To clear MA1F, write a logic one to the MA1F.</p> <ul style="list-style-type: none"> 0 Received data is not equal to MA1 1 Received data is equal to MA1 |
| 14 MA2F | <p>Match 2 Flag</p> <p>MA2F is set whenever the next character to be read from LPUART_DATA matches MA2. To clear MA2F, write a logic one to the MA2F.</p> <ul style="list-style-type: none"> 0 Received data is not equal to MA2 1 Received data is equal to MA2 |
| Reserved | <p>This field is reserved.</p> <p>This read-only field is reserved and always has the value 0.</p> |

36.3.3 LPUART Control Register (LPUARTx_CTRL)

This read/write register controls various optional features of the LPUART system. This register should only be altered when the transmitter and receiver are both disabled.

Address: 4005_4000h base + 8h offset = 4005_4008h

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|-------|-------|-------|-------|-------|------|---------|------|------|-------|--------|-------|------|------|-----|-----|-----|
| R | R8T9 | R9T8 | TXDIR | TXINV | ORIE | NEIE | FEIE | PEIE | TIE | TCIE | RIE | ILIE | TE | RE | RWU | SBK |
| 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 | MA1IE | MA2IE | 0 | | | IDLECFG | | | LOOPS | DOZEEN | RSR C | M | WAKE | ILT | PE | PT |
| W | | | | | | | | | | | | | | | | |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

LPUARTx_CTRL field descriptions

| Field | Description |
|-------------|---|
| 31 R8T9 | <p>Receive Bit 8 / Transmit Bit 9</p> <p>R8 is the ninth data bit received when the LPUART is configured for 9-bit or 10-bit data formats. When reading 9-bit or 10-bit data, read R8 before reading LPUART_DATA.</p> <p>T9 is the tenth data bit received when the LPUART is configured for 10-bit data formats. When writing 10-bit data, write T9 before writing LPUART_DATA. If T9 does not need to change from its previous value, such as when it is used to generate address mark or parity, they it need not be written each time LPUART_DATA is written.</p> |
| 30 R9T8 | <p>Receive Bit 9 / Transmit Bit 8</p> <p>R9 is the tenth data bit received when the LPUART is configured for 10-bit data formats. When reading 10-bit data, read R9 before reading LPUART_DATA</p> <p>T8 is the ninth data bit received when the LPUART is configured for 9-bit or 10-bit data formats. When writing 9-bit or 10-bit data, write T8 before writing LPUART_DATA. If T8 does not need to change from its previous value, such as when it is used to generate address mark or parity, they it need not be written each time LPUART_DATA is written.</p> |
| 29 TXDIR | <p>LPUART_TX Pin Direction in Single-Wire Mode</p> <p>When the LPUART is configured for single-wire half-duplex operation (LOOPS = RSRC = 1), this bit determines the direction of data at the LPUART_TX pin. When clearing TXDIR, the transmitter will finish receiving the current character (if any) before the receiver starts receiving data from the LPUART_TX pin.</p> <ul style="list-style-type: none"> 0 LPUART_TX pin is an input in single-wire mode. 1 LPUART_TX pin is an output in single-wire mode. |

Table continues on the next page...

LPUARTx_CTRL field descriptions (continued)

| Field | Description |
|-------------|--|
| 28 TXINV | <p>Transmit Data Inversion</p> <p>Setting this bit reverses the polarity of the transmitted data output.</p> <p>NOTE: Setting TXINV inverts the LPUART_TX output for all cases: data bits, start and stop bits, break, and idle.</p> <ul style="list-style-type: none"> 0 Transmit data not inverted. 1 Transmit data inverted. |
| 27 ORIE | <p>Overrun Interrupt Enable</p> <p>This bit enables the overrun flag (OR) to generate hardware interrupt requests.</p> <ul style="list-style-type: none"> 0 OR interrupts disabled; use polling. 1 Hardware interrupt requested when OR is set. |
| 26 NEIE | <p>Noise Error Interrupt Enable</p> <p>This bit enables the noise flag (NF) to generate hardware interrupt requests.</p> <ul style="list-style-type: none"> 0 NF interrupts disabled; use polling. 1 Hardware interrupt requested when NF is set. |
| 25 FEIE | <p>Framing Error Interrupt Enable</p> <p>This bit enables the framing error flag (FE) to generate hardware interrupt requests.</p> <ul style="list-style-type: none"> 0 FE interrupts disabled; use polling. 1 Hardware interrupt requested when FE is set. |
| 24 PEIE | <p>Parity Error Interrupt Enable</p> <p>This bit enables the parity error flag (PF) to generate hardware interrupt requests.</p> <ul style="list-style-type: none"> 0 PF interrupts disabled; use polling). 1 Hardware interrupt requested when PF is set. |
| 23 TIE | <p>Transmit Interrupt Enable</p> <p>Enables STAT[TDRE] to generate interrupt requests.</p> <ul style="list-style-type: none"> 0 Hardware interrupts from TDRE disabled; use polling. 1 Hardware interrupt requested when TDRE flag is 1. |
| 22 TCIE | <p>Transmission Complete Interrupt Enable for</p> <p>TCIE enables the transmission complete flag, TC, to generate interrupt requests.</p> <ul style="list-style-type: none"> 0 Hardware interrupts from TC disabled; use polling. 1 Hardware interrupt requested when TC flag is 1. |
| 21 RIE | <p>Receiver Interrupt Enable</p> <p>Enables STAT[RDRF] to generate interrupt requests.</p> <ul style="list-style-type: none"> 0 Hardware interrupts from RDRF disabled; use polling. 1 Hardware interrupt requested when RDRF flag is 1. |

Table continues on the next page...

LPUARTx_CTRL field descriptions (continued)

| Field | Description |
|-------------|--|
| 20 ILIE | <p>Idle Line Interrupt Enable</p> <p>ILIE enables the idle line flag, STAT[IDLE], to generate interrupt requests.</p> <p>0 Hardware interrupts from IDLE disabled; use polling. 1 Hardware interrupt requested when IDLE flag is 1.</p> |
| 19 TE | <p>Transmitter Enable</p> <p>Enables the LPUART transmitter. TE can also be used to queue an idle preamble by clearing and then setting TE. When TE is cleared, this register bit will read as 1 until the transmitter has completed the current character and the LPUART_TX pin is tristated.</p> <p>0 Transmitter disabled. 1 Transmitter enabled.</p> |
| 18 RE | <p>Receiver Enable</p> <p>Enables the LPUART receiver. When RE is written to 0, this register bit will read as 1 until the receiver finishes receiving the current character (if any).</p> <p>0 Receiver disabled. 1 Receiver enabled.</p> |
| 17 RWU | <p>Receiver Wakeup Control</p> <p>This field can be set to place the LPUART receiver in a standby state. RWU automatically clears when an RWU event occurs, that is, an IDLE event when CTRL[WAKE] is clear or an address match when CTRL[WAKE] is set with STAT[RWUID] is clear.</p> <p>NOTE: RWU must be set only with CTRL[WAKE] = 0 (wakeup on idle) if the channel is currently not idle. This can be determined by STAT[RAF]. If the flag is set to wake up an IDLE event and the channel is already idle, it is possible that the LPUART will discard data. This is because the data must be received or a LIN break detected after an IDLE is detected before IDLE is allowed to reasserted.</p> <p>0 Normal receiver operation. 1 LPUART receiver in standby waiting for wakeup condition.</p> |
| 16 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 LPUART_STATBRK13] is set, 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. 1 Queue break character(s) to be sent.</p> |
| 15 MA1IE | <p>Match 1 Interrupt Enable</p> <p>0 MA1F interrupt disabled 1 MA1F interrupt enabled</p> |
| 14 MA2IE | <p>Match 2 Interrupt Enable</p> <p>0 MA2F interrupt disabled 1 MA2F interrupt enabled</p> |

Table continues on the next page...

LPUARTx_CTRL field descriptions (continued)

| Field | Description |
|-------------------|--|
| 13–11 Reserved | This field is reserved. This read-only field is reserved and always has the value 0. |
| 10–8 IDLECFG | Idle Configuration Configures the number of idle characters that must be received before the IDLE flag is set. 000 1 idle character 001 2 idle characters 010 4 idle characters 011 8 idle characters 100 16 idle characters 101 32 idle characters 110 64 idle characters 111 128 idle characters |
| 7 LOOPS | Loop Mode Select When LOOPS is set, the LPUART_RX pin is disconnected from the LPUART and the transmitter output is internally connected to the receiver input. The transmitter and the receiver must be enabled to use the loop function. 0 Normal operation - LPUART_RX and LPUART_TX use separate pins. 1 Loop mode or single-wire mode where transmitter outputs are internally connected to receiver input (see RSRC bit). |
| 6 DOZEEN | Doze Enable 0 LPUART is enabled in Doze mode. 1 LPUART is disabled in Doze mode. |
| 5 RSRC | Receiver Source Select This field has no meaning or effect unless the LOOPS field is set. When LOOPS is set, the RSRC field determines the source for the receiver shift register input. 0 Provided LOOPS is set, RSRC is cleared, selects internal loop back mode and the LPUART does not use the LPUART_RX pin. 1 Single-wire LPUART mode where the LPUART_TX pin is connected to the transmitter output and receiver input. |
| 4 M | 9-Bit or 8-Bit Mode Select 0 Receiver and transmitter use 8-bit data characters. 1 Receiver and transmitter use 9-bit data characters. |
| 3 WAKE | Receiver Wakeup Method Select Determines which condition wakes the LPUART when RWU=1: <ul style="list-style-type: none">• Address mark in the most significant bit position of a received data character, or• An idle condition on the receive pin input signal. 0 Configures RWU for idle-line wakeup. 1 Configures RWU with address-mark wakeup. |
| 2 ILT | Idle Line Type Select |

Table continues on the next page...

LPUARTx_CTRL field descriptions (continued)

| Field | Description |
|---------|--|
| | <p>Determines when the receiver starts counting logic 1s as idle character bits. The count begins either after a valid start bit or after the stop bit. If the count begins after the start bit, then a string of logic 1s preceding the stop bit can cause false recognition of an idle character. Beginning the count after the stop bit avoids false idle character recognition, but requires properly synchronized transmissions.</p> <p>NOTE: In case the LPUART is programmed with ILT = 1, a logic 0 is automatically shifted after a received stop bit, therefore resetting the idle count.</p> <ul style="list-style-type: none"> 0 Idle character bit count starts after start bit. 1 Idle character bit count starts after stop bit. |
| 1 PE | <p>Parity Enable</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"> 0 No hardware parity generation or checking. 1 Parity enabled. |
| 0 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"> 0 Even parity. 1 Odd parity. |

36.3.4 LPUART Data Register (LPUARTx_DATA)

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 LPUART status flags.

Address: 4005_4000h base + Ch offset = 4005_400Ch

| 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 | NOISY | PARITYE | FRETSC | RXEMPT | IDLINE | 0 | R9T9 | R8T8 | R7T7 | R6T6 | R5T5 | R4T4 | R3T3 | R2T2 | R1T1 | R0T0 |
| W | | | | | | | | | | | | | | | | |
| Reset | 0 | 0 | 0 | 1 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

LPUARTx_DATA field descriptions

| Field | Description |
|-------------------|--|
| 31–16 Reserved | This field is reserved. This read-only field is reserved and always has the value 0. |
| 15 NOISY | The current received dataword contained in DATA[R9:R0] was received with noise. 0 The dataword was received without noise. 1 The data was received with noise. |

Table continues on the next page...

LPUARTx_DATA field descriptions (continued)

| Field | Description |
|----------------|---|
| 14 PARITYE | The current received dataword contained in DATA[R9:R0] was received with a parity error. 0 The dataword was received without a parity error. 1 The dataword was received with a parity error. |
| 13 FRETSC | Frame Error / Transmit Special Character For reads, indicates the current received dataword contained in DATA[R9:R0] was received with a frame error. For writes, indicates a break or idle character is to be transmitted instead of the contents in DATA[T9:T0]. T9 is used to indicate a break character when 0 and a idle character when 1, he contents of DATA[T8:T0] should be zero. 0 The dataword was received without a frame error on read, transmit a normal character on write. 1 The dataword was received with a frame error, transmit an idle or break character on transmit. |
| 12 RXEMPT | Receive Buffer Empty Asserts when there is no data in the receive buffer. This field does not take into account data that is in the receive shift register. 0 Receive buffer contains valid data. 1 Receive buffer is empty, data returned on read is not valid. |
| 11 IDLINE | Idle Line Indicates the receiver line was idle before receiving the character in DATA[9:0]. Unlike the IDLE flag, this bit can set for the first character received when the receiver is first enabled. 0 Receiver was not idle before receiving this character. 1 Receiver was idle before receiving this character. |
| 10 Reserved | This field is reserved. This read-only field is reserved and always has the value 0. |
| 9 R9T9 | Read receive data buffer 9 or write transmit data buffer 9. |
| 8 R8T8 | Read receive data buffer 8 or write transmit data buffer 8. |
| 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. |
| 4 R4T4 | Read receive data buffer 4 or write transmit data buffer 4. |
| 3 R3T3 | Read receive data buffer 3 or write transmit data buffer 3. |
| 2 R2T2 | Read receive data buffer 2 or write transmit data buffer 2. |
| 1 R1T1 | Read receive data buffer 1 or write transmit data buffer 1. |
| 0 R0T0 | Read receive data buffer 0 or write transmit data buffer 0. |

36.3.5 LPUART Match Address Register (LPUARTx_MATCH)

Address: 4005_4000h base + 10h offset = 4005_4010h

| 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 | | | | | | | | | | | | | | | | 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 | |

LPUARTx_MATCH field descriptions

| Field | Description |
|-------------------|--|
| 31–26 Reserved | This field is reserved. This read-only field is reserved and always has the value 0. |
| 25–16 MA2 | Match Address 2 The MA1 and MA2 registers are compared to input data addresses when the most significant bit is set and the associated BAUD[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 BAUD[MAEN] bit is clear. |
| 15–10 Reserved | This field is reserved. This read-only field is reserved and always has the value 0. |
| MA1 | Match Address 1 The MA1 and MA2 registers are compared to input data addresses when the most significant bit is set and the associated BAUD[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 BAUD[MAEN] bit is clear. |

36.4 Functional description

The LPUART 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 LPUART.

36.4.1 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 LPUART baud clock. The SBR bits are in the LPUART 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.

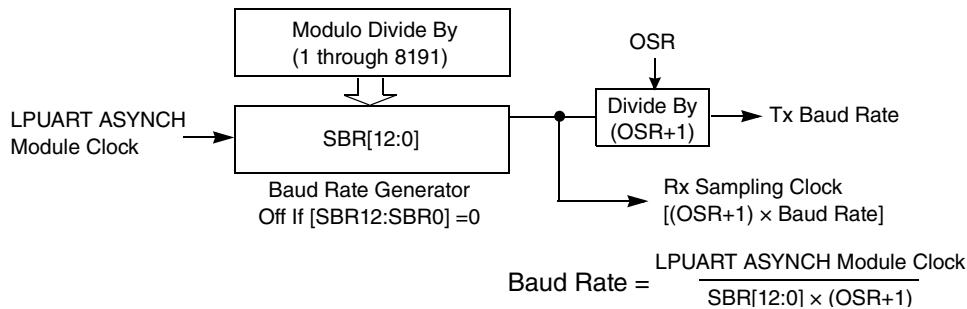


Figure 36-3. LPUART baud rate generation

Baud rate generation is subject to two sources of error:

- Integer division of the asynchronous LPUART baud clock may not give the exact target frequency.
- Synchronization with the asynchronous LPUART baud clock can cause phase shift.

36.4.2 Transmitter functional description

This section describes the overall block diagram for the LPUART transmitter, as well as specialized functions for sending break and idle characters.

The transmitter output (LPUART_TX) idle state defaults to logic high, CTRL[TXINV] is cleared following reset. The transmitter output is inverted by setting CTRL[TXINV]. The transmitter is enabled by setting the CTRL[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 LPUART data register.

The central element of the LPUART transmitter is the transmit shift register that is 10-bit to 13 bits long depending on the setting in the CTRL[M], BAUD[M10] and BAUD[SBNS] control bits. For the remainder of this section, assume CTRL[M], BAUD[M10] and BAUD[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 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 (STAT[TDRE]) status flag is set to indicate another character may be written to the transmit data buffer at LPUART_DATA.

If no new character is waiting in the transmit data buffer after a stop bit is shifted out the LPUART_TX pin, the transmitter sets the transmit complete flag and enters an idle mode, with LPUART_TX high, waiting for more characters to transmit.

Writing 0 to CTRL[TE] does not immediately disable the transmitter. The current transmit activity in progress must first be completed (that could include a data character, idle character or break character), although the transmitter will not start transmitting another character.

36.4.2.1 Send break and queued idle

The LPUART_CTRL[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 LPUART_STAT[BRK13]. Normally, a program would wait for LPUART_STAT[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 LPUART_CTRL[SBK] bit. This action queues a break character to be sent as soon as the shifter is available. If LPUART_CTRL[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 LPUART, the break characters are received as 0s in all data bits and a framing error (LPUART_STAT[FE] = 1) occurs.

A break character can also be transmitted by writing to the LPUART_DATA register with bit 13 set and the data bits clear. This supports transmitting the break character as part of the normal data stream

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 LPUART_STAT[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 LPUART_CTRL[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 LPUART_CTRL[TE] is cleared, the LPUART transmitter never actually releases control of the LPUART_TX pin.

An idle character can also be transmitted by writing to the LPUART_DATA register with bit 13 set and the data bits also set. This supports transmitting the idle character as part of the normal data stream.

The length of the break character is affected by the LPUART_STAT[BRK13], LPUART_CTRL[M], LPUART_BAUD[M10] and LPUART_BAUD[SNBS] bits as shown below.

Table 36-1. 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 | 13 bit times |
| 1 | 1 | 0 | 0 | 14 bit times |
| 1 | 1 | 0 | 1 | 14 bit times |
| 1 | X | 1 | 0 | 15 bit times |
| 1 | X | 1 | 1 | 15 bit times |

36.4.3 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, different variations of the receiver wakeup function are explained.

The receiver input is inverted by setting LPUART_STAT[RXINV]. The receiver is enabled by setting the LPUART_CTRL[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 LPUART 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 (LPUART_STAT[RDRF]) status flag is set. If LPUART_STAT[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 LPUART receiver is double-buffered, the program has one full character time after LPUART_STAT[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 (LPUART_STAT[RDRF] = 1), it gets the data from the receive data register by reading LPUART_DATA. Refer to [Interrupts and status flags](#) for details about flag clearing.

36.4.3.1 Data sampling technique

The LPUART receiver supports a configurable 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 LPUART_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 (LPUART_STAT[NF]) is set when the received character is transferred to the receive data buffer.

When the LPUART 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.

36.4.3.2 Receiver wakeup operation

Receiver wakeup and receiver address matching is a hardware mechanism that allows an LPUART receiver to ignore the characters in a message intended for a different receiver.

During receiver wakeup, 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(LPUART_CTRL[RWU]). When RWU bit is set, the status flags associated with the receiver, with the exception of the idle bit, IDLE, when LPUART_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 LPUART_CTRL[RWU] to 0 so all receivers wake up in time to look at the first character(s) of the next message.

During receiver address matching, the address matching is performed in hardware and the LPUART receiver will ignore all characters that do not meet the address match requirements.

Table 36-2. Receiver Wakeup Options

| RWU | MA1 MA2 | MATCFG | WAKE:RWUID | Receiver Wakeup |
|-----|-----------|--------|------------|--|
| 0 | 0 | X | X | Normal operation |
| 1 | 0 | 00 | 00 | Receiver wakeup on idle line, IDLE flag not set |
| 1 | 0 | 00 | 01 | Receiver wakeup on idle line, IDLE flag set |
| 1 | 0 | 00 | 10 | Receiver wakeup on address mark |
| 1 | 1 | 11 | X0 | Receiver wakeup on data match |
| 0 | 1 | 00 | X0 | Address mark address match, IDLE flag not set for discarded characters |
| 0 | 1 | 00 | X1 | Address mark address match, IDLE flag set for discarded characters |
| 0 | 1 | 01 | X0 | Idle line address match |
| 0 | 1 | 10 | X0 | Address match on and address match off, IDLE flag not set for discarded characters |
| 0 | 1 | 10 | X1 | Address match on and address match off, IDLE flag set for discarded characters |

36.4.3.2.1 Idle-line wakeup

When wake is cleared, the receiver is configured for idle-line wakeup. In this mode, LPUART_CTRL[RWU] is cleared automatically when the receiver detects a full character time of the idle-line level. The LPUART_CTRL[M] and LPUART_BAUD[M10] control bit selects 8-bit to 10-bit data mode and the LPUART_BAUD[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 LPUART_CTRL[RWU] is one and LPUART_STAT[RWUID] is zero, the idle condition that wakes up the receiver does not set the LPUART_STAT[IDLE] flag. The receiver wakes up and waits for the first data character of the next message that sets the LPUART_STAT[RDRF] flag and generates an interrupt if enabled. When LPUART_STAT[RWUID] is one, any idle condition sets the LPUART_STAT[IDLE] flag and generates an interrupt if enabled, regardless of whether LPUART_CTRL[RWU] is zero or one.

The idle-line type (LPUART_CTRL[ILT]) control bit selects one of two ways to detect an idle line. When LPUART_CTRL[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 LPUART_CTRL[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.

36.4.3.2.2 Address-mark wakeup

When LPUART_CTRL[WAKE] is set, the receiver is configured for address-mark wakeup. In this mode, LPUART_CTRL[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 LPUART_CTRL[RWU] bit before the stop bits are received and sets the LPUART_STAT[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.

36.4.3.2.3 Data match wakeup

When LPUART_CTRL[RWU] is set and LPUART_BAUD[MATCFG] equals 11, the receiver is configured for data match wakeup. In this mode, LPUART_CTRL[RWU] is cleared automatically when the receiver detects a character that matches MATCH[MA1] field when BAUD[MAEN1] is set, or that matches MATCH[MA2] when BAUD[MAEN2] is set.

36.4.3.2.4 Address Match operation

Address match operation is enabled when the LPUART_BAUD[MAEN1] or LPUART_BAUD[MAEN2] bit is set and LPUART_BAUD[MATCFG] is equal to 00. In this function, a character received by the LPUART_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 MATCH[MA1] or MATCH[MA2] field. The character is only transferred to the receive buffer, and LPUART_STAT[RDRF] is set, if the comparison matches. All subsequent characters 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 characters with logic zero in the bit position immediately preceding the stop bit are also discarded. If both the LPUART_BAUD[MAEN1] and LPUART_BAUD[MAEN2] bits are negated, the receiver operates normally and all data received is transferred to the receive data buffer.

Address match operation functions in the same way for both MATCH[MA1] and MATCH[MA2] fields.

- If only one of LPUART_BAUD[MAEN1] and LPUART_BAUD[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 LPUART_BAUD[MAEN1] and LPUART_BAUD[MAEN2] are asserted, a marked address is compared with both match registers and data is transferred only on a match with either register.

36.4.3.2.5 Idle Match operation

Idle match operation is enabled when the LPUART_BAUD[MAEN1] or LPUART_BAUD[MAEN2] bit is set and LPUART_BAUD[MATCFG] is equal to 01. In this function, the first character received by the LPUART_RX pin after an idle line condition is considered an address and is compared with the associated MA1 or MA2 register. The character is only transferred to the receive buffer, and LPUART_STAT[RDRF] is set, if the comparison matches. All subsequent characters are

considered to be data associated with the address and are transferred to the receive data buffer until the next idle line condition is detected. If no address match occurs then no transfer is made to the receive data buffer, and all following frames until the next idle condition are also discarded. If both the LPUART_BAUD[MAEN1] and LPUART_BAUD[MAEN2] bits are negated, the receiver operates normally and all data received is transferred to the receive data buffer.

Idle match operation functions in the same way for both MA1 and MA2 registers.

- If only one of LPUART_BAUD[MAEN1] and LPUART_BAUD[MAEN2] is asserted, the first character after an idle line is compared only with the associated match register and data is transferred to the receive data buffer only on a match.
- If LPUART_BAUD[MAEN1] and LPUART_BAUD[MAEN2] are asserted, the first character after an idle line is compared with both match registers and data is transferred only on a match with either register.

36.4.3.2.6 Match On Match Off operation

Match on, match off operation is enabled when both LPUART_BAUD[MAEN1] and LPUART_BAUD[MAEN2] are set and LPUART_BAUD[MATCFG] is equal to 10. In this function, a character received by the LPUART_RX pin that matches MATCH[MA1] is received and transferred to the receive buffer, and LPUART_STAT[RDRF] is set. All subsequent characters are considered to be data and are also transferred to the receive data buffer, until a character is received that matches MATCH[MA2] register. The character that matches MATCH[MA2] and all following characters are discarded, this continues until another character that matches MATCH[MA1] is received. If both the LPUART_BAUD[MAEN1] and LPUART_BAUD[MAEN2] bits are negated, the receiver operates normally and all data received is transferred to the receive data buffer.

Match on, match off operation requires both LPUART_BAUD[MAEN1] and LPUART_BAUD[MAEN2] to be asserted.

36.4.4 Additional LPUART functions

The following sections describe additional LPUART functions.

36.4.4.1 8-bit, 9-bit and 10-bit data modes

The LPUART transmitter and receiver can be configured to operate in 9-bit data mode by setting the LPUART_CTRL[M] or 10-bit data mode by setting LPUART_CTRL[M10]. In 9-bit mode, there is a ninth data bit in 10-bit mode there is a tenth data bit. For the transmit data buffer, these bits are stored in LPUART_CTRL[T8] and LPUART_CTRL[T9]. For the receiver, these bits are held in LPUART_CTRL[R8] and LPUART_CTRL[R9]. They are also accessible via 16-bit or 32-bit accesses to the LPUART_DATA register.

For coherent 8-bit writes to the transmit data buffer, write to LPUART_CTRL[T8] and LPUART_CTRL[T9] before writing to LPUART_DATA[7:0]. For 16-bit and 32-bit writes to the LPUART_DATA register all 10 transmit bits are written to the transmit data buffer at the same time.

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 LPUART_CTRL[T8] and LPUART_CTRL[T9] again. When data is transferred from the transmit data buffer to the transmit shifter, the value in LPUART_CTRL[T8] and LPUART_CTRL[T9] is copied at the same time data is transferred from LPUART_DATA[7:0] 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.

36.4.4.2 Idle length

An idle character is a character where the start bit, all data bits and stop bits are in the mark position. The CTRL[ILT] register can be configured to start detecting an idle character from the previous start bit (any data bits and stop bits count towards the idle character detection) or from the previous stop bit.

The number of idle characters that must be received before an idle line condition is detected can also be configured using the CTRL[IDLECFG] field. This field configures the number of idle characters that must be received before the STAT[IDLE] flag is set, the STAT[RAF] flag is cleared and the DATA[IDLINE] flag is set with the next received character.

Idle-line wakeup and idle match operation are also affected by the CTRL[IDLECFG] field. When address match or match on/off operation is enabled, setting the STAT[RWUID] bit will cause any discarded characters to be treated as if they were idle characters.

36.4.4.3 Loop mode

When LPUART_CTRL[LOOPS] is set, the LPUART_CTRL[RSRC] bit in the same register chooses between loop mode (LPUART_CTRL[RSRC] = 0) or single-wire mode (LPUART_CTRL[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 LPUART_RX pin is not used by the LPUART.

36.4.4.4 Single-wire operation

When LPUART_CTRL[LOOPS] is set, the RSRC bit in the same register chooses between loop mode (LPUART_CTRL[RSRC] = 0) or single-wire mode (LPUART_CTRL[RSRC] = 1). Single-wire mode implements a half-duplex serial connection. The receiver is internally connected to the transmitter output and to the LPUART_TX pin (the LPUART_RX pin is not used).

In single-wire mode, the LPUART_CTRL[TXDIR] bit controls the direction of serial data on the LPUART_TX pin. When LPUART_CTRL[TXDIR] is cleared, the LPUART_TX pin is an input to the receiver and the transmitter is temporarily disconnected from the LPUART_TX pin so an external device can send serial data to the receiver. When LPUART_CTRL[TXDIR] is set, the LPUART_TX 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.

36.4.5 Interrupts and status flags

The LPUART transmitter has two status flags that can optionally generate hardware interrupt requests. Transmit data register empty LPUART_STAT[TDRE]) indicates when there is room in the transmit data buffer to write another transmit character to LPUART_DATA. If the transmit interrupt enable LPUART_CTRL[TIE] bit is set, a hardware interrupt is requested when LPUART_STAT[TDRE] is set. Transmit complete (LPUART_STAT[TC]) indicates that the transmitter is finished transmitting all data, preamble, and break characters and is idle with LPUART_TX at the inactive level. This

Functional description

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 (LPUART_CTRL[TCIE]) bit is set, a hardware interrupt is requested when LPUART_STAT[TC] is set. Instead of hardware interrupts, software polling may be used to monitor the LPUART_STAT[TDRE] and LPUART_STAT[TC] status flags if the corresponding LPUART_CTRL[TIE] or LPUART_CTRL[TCIE] local interrupt masks are cleared.

When a program detects that the receive data register is full (LPUART_STAT[RDRF] = 1), it gets the data from the receive data register by reading LPUART_DATA. The LPUART_STAT[RDRF] flag is cleared by reading LPUART_DATA.

The IDLE status flag includes logic that prevents it from getting set repeatedly when the LPUART_RX line remains idle for an extended period of time. IDLE is cleared by writing 1 to the LPUART_STAT[IDLE] flag. After LPUART_STAT[IDLE] has been cleared, it cannot become set again until the receiver has received at least one new character and has set LPUART_STAT[RDRF].

If the associated error was detected in the received character that caused LPUART_STAT[RDRF] to be set, the error flags - noise flag (LPUART_STAT[NF]), framing error (LPUART_STAT[FE]), and parity error flag (LPUART_STAT[PF]) - are set at the same time as LPUART_STAT[RDRF]. These flags are not set in overrun cases.

If LPUART_STAT[RDRF] was already set when a new character is ready to be transferred from the receive shifter to the receive data buffer, the overrun (LPUART_STAT[OR]) flag is set instead of the data along with any associated NF, FE, or PF condition is lost.

If the received character matches the contents of MATCH[MA1] and/or MATCH[MA2] then the LPUART_STAT[MA1F] and/or LPUART_STAT[MA2F] flags are set at the same time that LPUART_STAT[RDRF] is set.

At any time, an active edge on the LPUART_RX serial data input pin causes the LPUART_STAT[RXEDGIF] flag to set. The LPUART_STAT[RXEDGIF] flag is cleared by writing a 1 to it. This function depends on the receiver being enabled (LPUART_CTRL[RE] = 1).

Appendix A

Changes between revisions 5 and 4

Table A-1. Changes between revisions 5 and 4

| Chapter | Description |
|---------------------------|--|
| Throughout the whole book | <ul style="list-style-type: none">Added new part information of MKL03Z32CBF4R. |



How to Reach Us:

Home Page:

nxp.com

Web Support:

nxp.com/support

Information in this document is provided solely to enable system and software implementers to use NXP products. There are no express or implied copyright licenses granted hereunder to design or fabricate any integrated circuits based on the information in this document. NXP reserves the right to make changes without further notice to any products herein.

NXP makes no warranty, representation, or guarantee regarding the suitability of its products for any particular purpose, nor does NXP assume any liability arising out of the application or use of any product or circuit, and specifically disclaims any and all liability, including without limitation consequential or incidental damages. "Typical" parameters that may be provided in NXP 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. NXP does not convey any license under its patent rights nor the rights of others. NXP sells products pursuant to standard terms and conditions of sale, which can be found at the following address: nxp.com/SalesTermsandConditions.

NXP, the NXP logo, NXP SECURE CONNECTIONS FOR A SMARTER WORLD, Freescale, the Freescale logo, and Kinetis are trademarks of NXP B.V. All other product or service names are the property of their respective owners. ARM, the ARM powered logo, and Cortex are registered trademarks of ARM Limited (or its subsidiaries) in the EU and/or elsewhere. All rights reserved.

©2014-2017 NXP B.V.

Document Number KL03P24M48SF0RM
Revision 5, July, 2017

