GR5525 Datasheet

Version: 1.2

Release Date: 2024-09-02

Shenzhen Goodix Technology Co., Ltd.

Contents

1 Product Overview..... 1
1.1 Features..... 1
1.2 Block Diagram..... 4
1.3 Applications..... 5
2 Pinout..... 6
2.1 QFN68..... 6
2.2 QFN56 (With Flash)..... 9
2.3 QFN56 (Without Flash)..... 12
2.4 Pin Properties..... 15
2.4.1 PMU Pin Properties..... 15
2.4.2 I/O Pin Properties..... 15
3 System..... 18
3.1 MCU Core..... 18
3.1.1 Functional Overview..... 18
3.1.2 Main Features..... 19
3.1.3 Performance..... 19
3.1.4 Interrupt..... 20
3.1.5 Debug Interfaces..... 23
3.2 Bus and Memory System..... 24
3.2.1 System Buses..... 24
3.2.1.1 AHB Connection Matrix..... 24
3.2.1.2 AHB Multilayer Arbitration..... 26
3.2.1.3 Bus Faults..... 27
3.2.2 Memory Map..... 27
3.2.3 Memory Subsystem..... 30
3.2.3.1 SRAM..... 31
3.2.3.1.1 Introduction..... 31
3.2.3.2 eFuse..... 31
3.2.3.2.1 Introduction..... 31
3.2.3.2.2 Main Features..... 32
3.2.3.2.3 Functional Description..... 32
3.2.3.2.3.1 Layout..... 33
3.2.3.2.3.2 Write/Read Instructions..... 36
3.2.3.2.3.3 Initial value check..... 37
3.2.3.2.3.4 CRC calculation..... 37
3.2.3.2.3.5 Write Keys to KEYRAM..... 37
3.2.3.3 Flash (Non-Volatile Memory)..... 38
3.2.3.3.1 Introduction..... 38
3.2.3.3.2 Main Features..... 38

3.2.3.3.3 Functional Description....39
3.2.3.3.3.1 Write and Erase....39
3.2.3.3.3.2 XQSPI Controller....39
3.2.3.3.3.3 Flash Operation Interfaces....42
3.2.3.3.4 Electrical Specifications....42
3.2.3.4 Cache....42
3.2.3.4.1 Introduction....42
3.2.3.4.2 Main Features....42
3.2.3.4.3 Block Diagram....43
3.2.3.4.4 Functional Description....43
3.2.3.4.4.1 Cacheable Range....43
3.2.3.4.4.2 Address Segmentation....44
3.2.3.4.4.3 Mapping Mode....45
3.2.3.4.4.4 Replacement Strategy....45
3.2.3.4.4.5 Miss Rate Monitor....45
3.3 PMU....45
3.3.1 Functional Overview....45
3.3.2 Power Module....47
3.3.2.1 DC-DC Converter....48
3.3.2.1.1 Introduction....48
3.3.2.1.2 Main Features....48
3.3.2.1.3 Regulation....48
3.3.2.1.4 Electrical Specifications....48
3.3.2.2 CORE_LDO....49
3.3.2.2.1 Introduction....49
3.3.2.2.2 Main Features....49
3.3.2.2.3 Enable and Disable....49
3.3.2.2.4 Regulation....49
3.3.2.2.5 Electrical Specifications....50
3.3.2.3 RET_LDO....50
3.3.2.3.1 Introduction....50
3.3.2.3.2 Main Features....50
3.3.2.3.3 Enable and Disable....50
3.3.2.3.4 Regulation....50
3.3.2.3.5 Electrical Specifications....51
3.3.2.4 AON LDO....51
3.3.2.4.1 Introduction....51
3.3.2.4.2 Main Features....51
3.3.2.4.3 Enable and Disable....51
3.3.2.4.4 Regulation....52
3.3.2.4.5 Electrical Specifications....52
3.3.2.5 ANA_IO_LDO....52

3.3.2.5.1 Introduction....52
3.3.2.5.2 Main Features....53
3.3.2.5.3 Enable and Disable....53
3.3.2.5.4 Regulation....53
3.3.2.5.5 Electrical Specifications....53
3.3.2.6 STB_IO_LDO....54
3.3.2.6.1 Introduction....54
3.3.2.6.2 Main Features....54
3.3.2.6.3 Enable and Disable....54
3.3.2.6.4 Regulation....54
3.3.2.6.5 Electrical Specifications....54
3.3.3 Power Management....55
3.3.3.1 Introduction....55
3.3.3.2 Power Mode....57
3.3.3.2.1 Active Mode....58
3.3.3.2.2 IDLE Mode....58
3.3.3.2.3 Sleep Mode....58
3.3.3.2.4 Ultra Deep Sleep Mode....59
3.3.3.3 Low Power Mode Entry....60
3.3.3.3.1 IDLE Mode Entry....60
3.3.3.3.2 Sleep Mode Entry....60
3.3.3.3.3 Ultra Deep Sleep Mode Entry....61
3.3.3.3.4 Timing Specifications....61
3.3.3.4 Low Power Mode Exit....61
3.3.4 Wakeup Source....62
3.3.4.1 Sleep Wakeup....62
3.3.4.2 Ultra Deep Sleep Wakeup....63
3.3.4.3 Programming....63
3.3.5 Brown-out Detector (BOD)....64
3.3.5.1 Introduction....64
3.3.5.2 Main Features....64
3.3.5.3 Functional Description....64
3.3.5.3.1 Architecture....64
3.3.5.3.2 Programming....65
3.3.5.4 Electrical Specifications....66
3.3.6 Digital Dynamic Voltage Scaling (DDVS)....66
3.3.6.1 Introduction....67
3.3.6.2 Main Features....67
3.3.6.3 Functional Description....67
3.3.6.3.1 Architecture....67
3.3.6.3.2 Programming....68
3.3.7 Power Mapping on Pin....69

3.3.7.1 Introduction....69
3.3.7.2 Main Features....69
3.3.7.3 Programming....69
3.3.8 Power Consumption....69
3.3.8.1 Electrical Specifications....70
3.3.8.1.1 Sleep Current Consumption....71
3.3.8.1.2 MCU Current Consumption....71
3.3.8.1.3 Radio Current Consumption....72
3.3.8.1.3.1 Radio Transmitting Current Consumption....72
3.3.8.1.3.2 Radio Receiving Current Consumption....72
3.3.8.1.4 ADC Current Consumption....72
3.4 Reset....73
3.4.1 Functional Overview....73
3.4.2 Power Reset....74
3.4.2.1 Power-on Reset....74
3.4.2.2 Brown-out Reset....75
3.4.3 Hardware Reset....75
3.4.3.1 External Reset Pin....75
3.4.3.2 Watchdog Reset....75
3.4.4 Software Reset....75
3.4.5 Reset Reason Recorder....76
3.4.5.1 Reset Reason Status....76
3.4.5.2 Programming....76
3.5 Clocks....76
3.5.1 Functional Overview....76
3.5.2 High Frequency Clocks....80
3.5.2.1 HFXO_32M....80
3.5.2.1.1 Introduction....80
3.5.2.1.2 Main Features....80
3.5.2.1.3 Enable and Disable....80
3.5.2.1.4 Regulation....81
3.5.2.1.5 Electrical Specifications....81
3.5.2.2 CPLL_192M....82
3.5.2.2.1 Introduction....82
3.5.2.2.2 Main Features....82
3.5.2.2.3 Enable and Disable....82
3.5.2.2.4 Regulation....83
3.5.2.2.5 Electrical Specifications....83
3.5.2.3 HFRC_192M....83
3.5.2.3.1 Introduction....83
3.5.2.3.2 Main Features....83
3.5.2.3.3 Enable and Disable....84

3.5.2.3.4 Regulation....84
3.5.2.3.5 Electrical Specifications....84
3.5.3 Low Frequency Clocks....85
3.5.3.1 RNG_OSC....85
3.5.3.1.1 Introduction....85
3.5.3.1.2 Main Features....85
3.5.3.1.3 Enable and Disable....85
3.5.3.1.4 Regulation....85
3.5.3.1.5 Electrical Specifications....86
3.5.3.2 LFXO_32K....86
3.5.3.2.1 Introduction....86
3.5.3.2.2 Main Features....86
3.5.3.2.3 Enable and Disable....86
3.5.3.2.4 Regulation....86
3.5.3.2.5 Electrical Specifications....86
3.5.3.3 LFRC_32K....87
3.5.3.3.1 Introduction....87
3.5.3.3.2 Main Features....87
3.5.3.3.3 Enable and Disable....87
3.5.3.3.4 Regulation....87
3.5.3.3.5 Electrical Specifications....87
3.5.4 Clock Gating....88
3.5.4.1 Architecture....88
3.5.4.2 Programming....89
3.5.5 Clock Calibration....90
3.5.5.1 Introduction....90
3.5.5.2 Main Features....90
3.5.5.3 Functional Description....90
3.5.5.3.1 Architecture....90
3.5.5.3.2 Programming....91
3.5.6 Clock Mapping to Pins....91
3.5.6.1 Introduction....91
3.5.6.2 Main Features....91
3.5.6.3 Programming....92
3.6 Startup Procedure....92
3.6.1 Introduction....92
3.6.2 ROM Boot....92
3.6.2.1 BL0....94
3.6.2.2 BL1....94
3.6.2.3 Boot Exception....95
3.7 Registers....96
3.7.1 Always-on Control....96

3.7.1.1 MCU_CLK_CTRL....96
3.7.1.2 MCU_MISC_CFG....98
3.7.1.3 XO_CTRL....98
3.7.1.4 FLASH_CACHE_CTRL0....99
3.7.1.5 FLASH_PSRAM_PAD_PWR....99
3.7.1.6 DIGIO_FST_CLK....100
3.7.1.7 AON_CLK....100
3.7.1.8 TPP_ANA....101
3.7.1.9 AON_PWR_SAVING....101
3.7.1.10 MCU_WAKEUP_CTRL....102
3.7.1.11 AON_SLP_EVENT....104
3.7.1.12 AON_SLP_EVENT_RAW....105
3.7.1.13 AON_IRQ....106
3.7.1.14 AON_IRQ_EN....106
3.7.1.15 AON_DBG_CTRL....108
3.7.1.16 MEM_MARGIN....108
3.7.1.17 EXT_WAKEUP_CTRL0....109
3.7.1.18 EXT_WAKEUP_CTRL1....110
3.7.1.19 EXT_WAKEUP_STAT....111
3.7.1.20 COMM_CTRL....112
3.7.1.21 BLE_MISC....113
3.7.1.22 COMM_TIMER_CFG0....113
3.7.1.23 COMM_TIMER_CFG1....114
3.7.1.24 BLE_DEEP_SLP_CORR_EN....114
3.7.1.25 BLE_DEEP_SLP_HW_TIMER_CORR....114
3.7.1.26 COMM_TIMER_STAT....115
3.7.1.27 PMU_COMP_GLITCH_REMOVE....115
3.7.1.28 SOFTWARE_REG0....115
3.7.1.29 SOFTWARE_REG1....116
3.7.1.30 SOFTWARE_REG2....116
3.7.1.31 DBG_REG0....117
3.7.1.32 DBG_REG_RST_SRC....117
3.7.1.33 MEM_BOND_OPT....118
3.7.2 Always-on Power Control....118
3.7.2.1 A_TIMING_CTRL0....118
3.7.2.2 A_TIMING_CTRL1....119
3.7.2.3 A_TIMING_CTRL2....119
3.7.2.4 A_TIMING_CTRL3....120
3.7.2.5 A_TIMING_CTRL4....120
3.7.2.6 MEM_PWR_BOOST....120
3.7.2.7 A_SLP_CFG....121
3.7.2.8 AON_START_CFG....122

3.7.2.9 DPAD_LE_CTRL.....123
3.7.2.10 AON_SLP_CTRL.....124
3.7.2.11 XO_PLL_SET.....124
3.7.2.12 XO_PLL_CLR.....125
3.7.2.13 XO_PLL_STAT.....125
3.7.2.14 PWR_SET.....126
3.7.2.15 PWR_CLR.....126
3.7.2.16 PWR_STAT.....126
3.7.2.17 MEM_PWR_SLP0.....127
3.7.2.18 MEM_PWR_SLP1.....129
3.7.2.19 MEM_PWR_WKUP0.....130
3.7.2.20 MEM_PWR_WKUP1.....132
3.7.2.21 MEM_PWR_APPLY.....134
3.7.2.22 MEM_PWR_STAT0.....134
3.7.2.23 MEM_PWR_STAT1.....136
3.7.2.24 MEM_PWR_DBG.....137
3.7.2.25 COMM_CORE_PWR_CTRL_SW.....138
3.7.2.26 COMM_CORE_PWR_CTRL_HW_CFG.....138
3.7.2.27 COMM_CORE_PWR_CTRL_HW_CTRL.....139
3.7.2.28 COMM_CORE_PWR_CTRL_HW_STAT.....140
3.7.2.29 COMM_CORE_PWR_CTRL_PD_DLY.....140
3.7.2.30 COMM_CORE_PWR_STAT.....141
3.7.2.31 COMM_TIMER_PWR_CTRL.....141
3.7.2.32 SOFTWARE_REG_3.....142
3.7.3 Always-on PMU Control.....142
3.7.3.1 RF_REG_0.....143
3.7.3.2 RF_REG_1.....143
3.7.3.3 RF_REG_2.....144
3.7.3.4 RF_REG_3.....145
3.7.3.5 RF_REG_4.....146
3.7.3.6 RC_RTC_REG_0.....146
3.7.3.7 DCDC_LDO_REG_0.....147
3.7.3.8 FS_REG_0.....147
3.7.3.9 FS_REG_1.....148
3.7.3.10 ADVS_DCDC.....148
3.7.3.11 ADVS_DIGCORE.....149
3.7.3.12 PMU_LPD_CFG.....150
3.7.3.13 PMU_TON_CFG.....150
3.7.3.14 PMU_CORE_LDO_VREF.....151
3.7.3.15 PMU_DCDC_VREF.....151
3.7.3.16 PMU_INTF_OVR_EN_0.....151
3.7.3.17 PMU_INTF_OVR_VAL_0.....152

3.7.3.18 PMU_INTF_OVR_RD_0 153
3.7.4 Always-on RF Control 153
3.7.4.1 RF5 153
3.7.4.2 RF6 154
3.7.4.3 RF7 155
3.7.4.4 RF8 155
3.7.4.5 RF9 157
3.7.4.6 RF10 157
3.7.4.7 RF_RD_REG_0 158
3.7.4.8 RF_XO_BIAS_VAL 158
3.7.4.9 RF_INTF_OVR_EN_0 159
3.7.4.10 RF_INTF_OVR_VAL_0 159
3.7.4.11 RF_INTF_OVR_RD_0 159
3.7.4.12 RF_INTF_OVR_EN_1 160
3.7.4.13 RF_INTF_OVR_VAL_1 160
3.7.4.14 RF_INTF_OVR_RD_1 161
3.7.5 Cache 161
3.7.5.1 CACHE_REGS 161
3.7.5.1.1 CTRL0 161
3.7.5.1.2 HIT_COUNT 163
3.7.5.1.3 MISS_COUNT 163
3.7.5.1.4 STAT 164
3.7.5.2 QSPI_REGS 164
3.7.5.2.1 TX_DATA 164
3.7.5.2.2 RX_DATA 164
3.7.5.2.3 CTRL 165
3.7.5.2.4 AUX_CTRL 166
3.7.5.2.5 STAT 168
3.7.5.2.6 SLAVE_SEL 169
3.7.5.2.7 SLAVE_SEL_POL 170
3.7.5.2.8 INTEN 171
3.7.5.2.9 INTSTAT 171
3.7.5.2.10 INTCLR 172
3.7.5.2.11 TX_FIFO_LVL 172
3.7.5.2.12 RX_FIFO_LVL 173
3.7.5.2.13 MSTR_IT_DELAY 173
3.7.5.2.14 SPIEN 173
3.7.5.2.15 SPI_GP_SET 174
3.7.5.2.16 SPI_GP_CLEAR 174
3.7.5.2.17 RX_DATA0_31 175
3.7.5.2.18 RX_DATA32_63 175
3.7.5.2.19 RX_DATA64_95 175

3.7.5.2.20 RX_DATA96_127....176
3.7.5.2.21 FLASH_WRITE....176
3.7.5.2.22 CS_IDLE_UNVLD_EN....176
3.7.5.3 XIP_REGS....177
3.7.5.3.1 CTRL0....177
3.7.5.3.2 CTRL1....178
3.7.5.3.3 CTRL2....179
3.7.5.3.4 CTRL3....180
3.7.5.3.5 STAT....180
3.7.5.3.6 INTEN....181
3.7.5.3.7 INTSTAT....181
3.7.5.3.8 INTREQ....182
3.7.5.3.9 INTSET....182
3.7.5.3.10 INTCLR....183
3.7.5.3.11 SOFT_RST....183
3.7.6 MCU Subsystem....183
3.7.6.1 MCU_SUB....184
3.7.6.1.1 CTE_FIFO_DATA....184
3.7.6.1.2 CTE_FIFO_THRESH....184
3.7.6.1.3 CTE_FIFO_STAT....184
3.7.6.1.4 CTE_CFG....185
3.7.6.1.5 QSPI_M_XIP....185
3.7.6.1.6 QSPI_M_HRESP_DBG....186
3.7.6.1.7 QSPI_M_CS_SETUP_DLY....187
3.7.6.1.8 QSPI_M_CS_RELEASE_DLY....187
3.7.6.1.9 DMA0_HS_SEL....188
3.7.6.1.10 DMA1_HS_SEL....189
3.7.6.1.11 PAD_IE_N_AUTO....190
3.7.6.1.12 BLE_FERP_CTL....191
3.7.6.1.13 SECURITY_RESET....192
3.7.6.1.14 PMU_ID....194
3.7.6.1.15 PWR_AVG_CTL_REG0....194
3.7.6.1.16 EFUSE_PWR_DELTA_0....195
3.7.6.1.17 EFUSE_PWR_DELTA_1....195
3.7.6.1.18 EFUSE_PWR_CTRL_0....195
3.7.6.1.19 EFUSE_PWR_CTRL_1....196
3.7.6.1.20 I2S_CLK_CFG....196
3.7.6.1.21 I2S_DMA_MODE....197
3.7.6.1.22 MCU_SUB_REG....197
3.7.6.1.23 MCU_NMI_CFG....198
3.7.6.1.24 CPLL_IRQ_CFG....199
3.7.6.1.25 AON_SW_RST....199

3.7.6.1.26 APB_TIMER_DBG 199
3.7.6.1.27 APB_MON_DBG 200
3.7.6.1.28 MCU_RELEASE 200
3.7.6.1.29 FPGA_CTRL 201
3.7.6.1.30 ST_CALIB 201
3.7.6.2 MCU_RET 201
3.7.6.2.1 MCU_SUBSYS_CG_CTRL[0] 201
3.7.6.2.2 MCU_SUBSYS_CG_CTRL[1] 203
3.7.6.2.3 MCU_SUBSYS_CG_CTRL[2] 204
3.7.6.2.4 MCU_PERIPH_PCLK_OFF 205
3.7.6.2.5 MCU_PERIPH_CG_LP_EN 209
3.7.6.2.6 MCU_PERIPH_CLK_SLP_OFF 209
3.7.6.2.7 SECU_CLK_CTRL 210
3.7.6.2.8 MCU_MISC_CLK 211
3.7.6.2.9 HFOSC_CLK_EN 212
3.7.6.2.10 CLK_FORCE_ON 213
3.7.7 Clock Calibration 213
3.7.7.1 SL_CLK_CTRL 213
3.7.7.2 SL_CLK_CNT 214
3.7.7.3 SL_CLK_STAT 214
3.7.7.4 SL_CLK_CNT0 215
3.7.7.5 SL_CLK_CNT1 215
3.7.7.6 SL_CLK_INT_EN 215
3.7.7.7 SL_CLK_INT_CLR 216
3.7.7.8 SL_CLK_SEL 216
3.7.7.9 HS_CLK_CTRL 217
3.7.7.10 HS_CLK_CNT 217
3.7.7.11 HS_CLK_STAT 217
3.7.7.12 HS_CLK_CNT0 218
3.7.7.13 HS_CLK_CNT1 218
3.7.7.14 HS_CLK_INT_EN 218
3.7.7.15 HS_CLK_INT_CLR 219
3.7.7.16 HS_CLK_SEL 219
3.7.8 DDVS 220
3.7.8.1 DDVS_EN 220
3.7.8.2 DDVS_CFG_1 220
3.7.8.3 DDVS_CFG_2 221
3.7.8.4 DDVS_RINGO_CNT_01 222
3.7.8.5 DDVS_RINGO_CNT_23 222
3.7.8.6 DDVS_FSM 223
3.7.8.7 DDVS_CLK_CTRL 223
3.7.9 eFuse 224

3.7.9.1 TPGM....224
3.7.9.2 PGENB....225
3.7.9.3 TEST_MODE....225
3.7.9.4 OPERATION....225
3.7.9.5 STAT....226
3.7.9.6 KEY_MASK....227
3.7.9.7 CRC_ADDR....227
3.7.9.8 CRC_OUTPUT....228
3.7.9.9 TRIM_ADDR....228
3.7.9.10 TRIM_LEN....228
3.7.9.11 TRIM[n]....229

4 Peripherals....230

4.1 Pin Mux....230
4.1.1 Introduction....230
4.1.2 Functional Description....230
4.1.3 Registers....233
4.1.3.1 DPAD_MUX_CTRL_00_07....233
4.1.3.2 DPAD_MUX_CTRL_08_15....234
4.1.3.3 DPAD_MUX_CTRL_16_23....234
4.1.3.4 DPAD_MUX_CTRL_24_31....235
4.1.3.5 AON_PAD_MUX_CTRL....235
4.1.3.6 MSIO_APAD_MUX_CTRL....236
4.1.3.7 DPAD_MUX_CTRL_32_33....236
4.2 I/O....237
4.2.1 GPIO....237
4.2.1.1 Introduction....237
4.2.1.2 Main Features....237
4.2.1.3 Functional Description....237
4.2.1.4 Registers....239
4.2.1.4.1 DPAD_R_EN_BUS_0....239
4.2.1.4.2 DPAD_RTYP_BUS_0....239
4.2.1.4.3 DPAD_IN_EN_BUS_0....239
4.2.1.4.4 DPAD_R_EN_BUS_1....240
4.2.1.4.5 DPAD_RTYP_BUS_1....240
4.2.1.4.6 DPAD_IN_EN_BUS_1....241
4.2.1.4.7 DPAD_R_EN_MISC....241
4.2.1.4.8 DPAD_RTYP_MISC....242
4.2.1.4.9 DPAD_IN_EN_MISC....242
4.2.1.4.10 GPIOx_DATA....243
4.2.1.4.11 GPIOx_DATA_OUT....243
4.2.1.4.12 GPIOx_OUT_EN....244
4.2.1.4.13 GPIOx_OUT_CLR....244

4.2.1.4.14 GPIOx_INT_EN.....245
4.2.1.4.15 GPIOx_INT_CLR.....245
4.2.1.4.16 GPIOx_INT_TYPE_EN.....246
4.2.1.4.17 GPIOx_INT_TYPE_CLR.....247
4.2.1.4.18 GPIOx_INT_POL_EN.....247
4.2.1.4.19 GPIOx_INT_POL_CLR.....248
4.2.1.4.20 GPIOx_INT_STAT.....248
4.2.1.5 Electrical Specifications.....249
4.2.2 Always-On I/O.....249
4.2.2.1 Introduction.....249
4.2.2.2 Functional Description.....250
4.2.2.3 Registers.....250
4.2.2.3.1 AON_PAD_CTRL0.....250
4.2.2.3.2 AON_PAD_CTRL1.....250
4.2.2.3.3 AON_PAD_CLK.....251
4.2.2.3.4 AON_PAD_MCU_OVR.....252
4.2.2.4 Electrical Specifications.....252
4.2.3 Mixed Signal I/O.....253
4.2.3.1 Introduction.....253
4.2.3.2 Functional Description.....253
4.2.3.3 Registers.....253
4.2.3.3.1 MSIO_APAD_CFG0.....253
4.2.3.3.2 MSIO_APAD_CFG1.....254
4.2.3.3.3 MSIO_APAD_MCU_OVR.....254
4.2.3.4 Electrical Specifications.....255
4.3 Timer Interfaces.....255
4.3.1 Timer.....255
4.3.1.1 Introduction.....255
4.3.1.2 Main Features.....256
4.3.1.3 Functional Description.....256
4.3.1.4 Registers.....256
4.3.1.4.1 CTRL.....256
4.3.1.4.2 VALUE.....257
4.3.1.4.3 RELOAD.....257
4.3.1.4.4 INTSTAT.....257
4.3.2 Dual Timer.....258
4.3.2.1 Introduction.....258
4.3.2.2 Main Features.....258
4.3.2.3 Functional Description.....258
4.3.2.4 Registers.....259
4.3.2.4.1 RELOAD.....259
4.3.2.4.2 VALUE.....259

4.3.2.4.3 CTRL.....260
4.3.2.4.4 INTCLR.....261
4.3.2.4.5 RAW_INTSTAT.....261
4.3.2.4.6 INTSTAT.....261
4.3.2.4.7 BG_LOAD.....262
4.3.3 PWM.....262
4.3.3.1 Introduction.....262
4.3.3.2 Main Features.....262
4.3.3.3 Block Diagram.....263
4.3.3.4 Functional Description.....263
4.3.3.4.1 Flicker Mode.....263
4.3.3.4.2 Breath Mode.....264
4.3.3.4.3 Alignment.....264
4.3.3.4.4 Active Update.....266
4.3.3.4.5 Duty Cycle Calculation.....267
4.3.3.4.6 Recommended Setting in Breath Mode.....267
4.3.3.5 Registers.....267
4.3.3.5.1 MODE.....267
4.3.3.5.2 UPDATE.....268
4.3.3.5.3 PRD.....270
4.3.3.5.4 CMPA0.....271
4.3.3.5.5 CMPA1.....271
4.3.3.5.6 CMPB0.....271
4.3.3.5.7 CMPB1.....271
4.3.3.5.8 CMPC0.....272
4.3.3.5.9 CMPC1.....272
4.3.3.5.10 AQCTRL.....272
4.3.3.5.11 BRPRD.....274
4.3.3.5.12 HOLD.....274
4.3.4 Always-on Watchdog (AON_WDT).....275
4.3.4.1 Introduction.....275
4.3.4.2 Main Features.....275
4.3.4.3 Functional Description.....275
4.3.4.3.1 Block Diagram.....275
4.3.4.3.2 Clock.....276
4.3.4.3.3 Enable/Disable and Timer Reload.....276
4.3.4.3.4 Alarm.....276
4.3.4.4 Registers.....276
4.3.4.4.1 CFG_0.....276
4.3.4.4.2 LOCK.....277
4.3.4.4.3 STAT.....277
4.3.4.4.4 CLK.....278

4.3.4.4.5 TIMER_WR....278
4.3.4.4.6 ALARM_WR....279
4.3.4.4.7 TIMER_RD....279
4.3.4.4.8 ALARM_RD....279
4.3.5 Real-Time Counter (RTC/Calendar)....280
4.3.5.1 Main Features....280
4.3.5.2 Functional Description....280
4.3.5.3 Registers....281
4.3.5.3.1 CTRL....281
4.3.5.3.2 CLK_DIV....282
4.3.5.3.3 IRQ_EN....282
4.3.5.3.4 IRQ_STAT....283
4.3.5.3.5 STAT....283
4.3.5.3.6 CLK_SEL....283
4.3.5.3.7 CMP_SET....284
4.3.5.3.8 TIMER_SET....284
4.3.5.3.9 TICK_SET....285
4.3.5.3.10 TIMER_RD....285
4.3.5.3.11 CMP_RD....285
4.3.5.3.12 TICK_RD....286
4.3.6 Sleep Timer (SLP TIMER)....286
4.3.6.1 Introduction....286
4.3.6.2 Main Features....286
4.3.6.3 Functional Description....286
4.3.6.4 Registers....287
4.3.6.4.1 CFG_0....287
4.3.6.4.2 STS....287
4.3.6.4.3 CLK....288
4.3.6.4.4 VAL_W....288
4.3.6.4.5 VAL_R....288
4.4 Communication Interfaces....289
4.4.1 UART....289
4.4.1.1 Introduction....289
4.4.1.2 Main Features....289
4.4.1.3 Block Diagram....289
4.4.1.4 Functional Description....291
4.4.1.4.1 UART (RS232) Serial Protocol....291
4.4.1.4.2 Fractional Baud Rate Support....292
4.4.1.4.3 FIFO Support....294
4.4.1.4.4 Clock Support....294
4.4.1.4.5 Interrupts....294
4.4.1.4.6 Auto Flow Control....296

4.4.1.4.7 Programmable THRE Interrupt....298
4.4.1.4.8 DMA Support....300
4.4.1.4.9 IrDA 1.0 SIR Protocol....300
4.4.1.5 Registers....302
4.4.1.5.1 RBR....302
4.4.1.5.2 DLL....302
4.4.1.5.3 THR....303
4.4.1.5.4 DLH....303
4.4.1.5.5 IER....304
4.4.1.5.6 FCR....305
4.4.1.5.7 IIR....307
4.4.1.5.8 LCR....308
4.4.1.5.9 MCR....310
4.4.1.5.10 LSR....312
4.4.1.5.11 MSR....313
4.4.1.5.12 SRBR....315
4.4.1.5.13 STHR....315
4.4.1.5.14 FAR....316
4.4.1.5.15 USR....317
4.4.1.5.16 TFL....318
4.4.1.5.17 RFL....318
4.4.1.5.18 SRR....319
4.4.1.5.19 SRTS....320
4.4.1.5.20 SBCR....320
4.4.1.5.21 SFE....321
4.4.1.5.22 SRT....322
4.4.1.5.23 STET....322
4.4.1.5.24 DLF....323
4.4.1.6 Electrical Specifications....323
4.4.2 SPI....324
4.4.2.1 Introduction....324
4.4.2.2 Main Features....324
4.4.2.3 Block Diagram....325
4.4.2.4 Functional Description....325
4.4.2.4.1 Motorola Serial Peripheral Interface....325
4.4.2.4.2 Transmit and Receive FIFO Buffers....327
4.4.2.4.3 SPI Interrupts....328
4.4.2.4.4 Transfer Modes....329
4.4.2.4.4.1 Transmit and Receive....329
4.4.2.4.4.2 Transmit Only....329
4.4.2.4.4.3 Receive Only....329
4.4.2.4.4.4 EEPROM Read....329

4.4.2.4.5 SPI Master Operation..... 330
4.4.2.4.6 SPI Slave Operation..... 330
4.4.2.5 Registers..... 331
4.4.2.5.1 CTRL0..... 331
4.4.2.5.2 CTRL1..... 335
4.4.2.5.3 SSI_EN..... 336
4.4.2.5.4 MW_CTRL..... 336
4.4.2.5.5 S_EN..... 337
4.4.2.5.6 BAUD..... 338
4.4.2.5.7 TX_FIFO_TL..... 339
4.4.2.5.8 RX_FIFO_TL..... 339
4.4.2.5.9 TX_FIFO_LV..... 340
4.4.2.5.10 RX_FIFO_LV..... 340
4.4.2.5.11 STAT..... 341
4.4.2.5.12 INT_MASK..... 343
4.4.2.5.13 INT_STAT..... 344
4.4.2.5.14 RAW_INT_STAT..... 345
4.4.2.5.15 TX_FIFO_OIC..... 346
4.4.2.5.16 RX_FIFO_OIC..... 347
4.4.2.5.17 RX_FIFO_UIC..... 347
4.4.2.5.18 MULTI_M_IC..... 348
4.4.2.5.19 INT_CLR..... 348
4.4.2.5.20 DMA_CTRL..... 349
4.4.2.5.21 DMA_TX_DL..... 349
4.4.2.5.22 DMA_RX_DL..... 350
4.4.2.5.23 DATA..... 350
4.4.2.5.24 RX_SMP_DLY..... 351
4.4.2.6 Electrical Specifications..... 351
4.4.2.6.1 SPIM Electrical Specifications..... 351
4.4.2.6.2 SPIS Electrical Specifications..... 352
4.4.3 I2C..... 353
4.4.3.1 Introduction..... 353
4.4.3.2 Main Features..... 353
4.4.3.3 Functional Description..... 354
4.4.3.3.1 I2C Operation..... 355
4.4.3.3.2 START and STOP Conditions..... 356
4.4.3.3.3 Addressing Slave..... 356
4.4.3.3.4 Transmitting and Receiving..... 358
4.4.3.3.5 SDA Hold Time..... 361
4.4.3.4 Registers..... 362
4.4.3.4.1 CTRL..... 363
4.4.3.4.2 TARGET_ADDR..... 366

4.4.3.4.3 S_ADDR....366
4.4.3.4.4 M_HS_ADDR....367
4.4.3.4.5 DATA_CMD....368
4.4.3.4.6 SS_CLK_HCNT....369
4.4.3.4.7 SS_CLK_LCNT....370
4.4.3.4.8 FS_CLK_HCNT....370
4.4.3.4.9 FS_CLK_LCNT....371
4.4.3.4.10 HS_CLK_HCNT....371
4.4.3.4.11 HS_CLK_LCNT....372
4.4.3.4.12 INT_STAT....373
4.4.3.4.13 INT_MASK....375
4.4.3.4.14 RAW_INT_STAT....377
4.4.3.4.15 RX_FIFO_THD....382
4.4.3.4.16 TX_FIFO_THD....383
4.4.3.4.17 CLR_INT....383
4.4.3.4.18 CLR_RX_UNDER....384
4.4.3.4.19 CLR_RX_OVER....384
4.4.3.4.20 CLR_TX_OVER....384
4.4.3.4.21 CLR_RD_REQ....385
4.4.3.4.22 CLR_TX_ABORT....385
4.4.3.4.23 CLR_RX_DONE....386
4.4.3.4.24 CLR_ACTIVITY....386
4.4.3.4.25 CLR_STOP_DET....387
4.4.3.4.26 CLR_START_DET....387
4.4.3.4.27 CLR_GEN_CALL....387
4.4.3.4.28 EN....388
4.4.3.4.29 STAT....389
4.4.3.4.30 TX_FIFO_LV....391
4.4.3.4.31 RX_FIFO_LV....392
4.4.3.4.32 SDA_HOLD....392
4.4.3.4.33 TX_ABORT_SRC....393
4.4.3.4.34 DMA_CTRL....399
4.4.3.4.35 DMA_TX_LV....399
4.4.3.4.36 DMA_RX_LV....400
4.4.3.4.37 SDA_SETUP....400
4.4.3.4.38 ACK_GEN_CALL....401
4.4.3.4.39 EN_STAT....401
4.4.3.4.40 FS_SPKLEN....403
4.4.3.4.41 HS_SPKLEN....404
4.4.3.4.42 SCL_STUCK_AT_LOW_TIMEOUT....405
4.4.3.4.43 SDA_STUCK_AT_LOW_TIMEOUT....405
4.4.3.4.44 CLR_SCL_STUCK_DET....406

4.4.3.5 Electrical Specifications....406
4.4.4 DSPI....407
4.4.4.1 Introduction....407
4.4.4.2 Main Features....407
4.4.4.3 Functional Description....408
4.4.4.3.1 Write Timing....408
4.4.4.3.1.1 3W1L Display Interface....409
4.4.4.3.1.2 4W1L Display Interface....409
4.4.4.3.1.3 4W2L Display Interface....409
4.4.4.3.2 Color Coding....410
4.4.4.3.2.1 3W1L Display Interface [A0+8-bit]....410
4.4.4.3.2.2 3W1L Display Interface [A0+9-bit]....410
4.4.4.3.2.3 3W1L Display Interface [A0+16-bit]....411
4.4.4.3.2.4 3W1L Display Interface [A0+18-bit]....411
4.4.4.3.2.5 3W1L Display Interface [A0+24-bit]....411
4.4.4.3.2.6 4W1L Display Interface [8-bit]....412
4.4.4.3.2.7 4W2L Display Interface [A0+8-bit]....412
4.4.4.3.2.8 4W2L Display Interface [A0+9-bit]....413
4.4.4.3.2.9 4W2L Display Interface [A0+12-bit]....414
4.4.4.3.3 Choice of DS and SPIMODE for Target Pixel Sizes....415
4.4.4.4 Registers....415
4.4.4.4.1 CTRL1....415
4.4.4.4.2 CTRL2....418
4.4.4.4.3 STAT....420
4.4.4.4.4 DATA....421
4.4.4.4.5 MODE....422
4.4.5 QSPI....423
4.4.5.1 Introduction....423
4.4.5.2 Main Features....423
4.4.5.3 Functional Description....424
4.4.5.3.1 Block Diagram....424
4.4.5.3.2 Clock Ratios....424
4.4.5.3.3 Transmit and Receive FIFO Buffers....425
4.4.5.3.4 QSPI Interrupts....425
4.4.5.3.5 Transfer Modes....426
4.4.5.3.5.1 Transmit and Receive....426
4.4.5.3.5.2 Transmit Only....426
4.4.5.3.5.3 Receive Only....427
4.4.5.3.5.4 EEPROM Read....427
4.4.5.3.6 DMA Controller Interface....427
4.4.5.3.6.1 DMA Operation....427
4.4.5.3.6.2 Transmit Watermark Level....428

4.4.5.3.6.3 Receive Watermark Level.... 428
4.4.5.3.6.4 DMA Usage Allocation.... 428
4.4.5.3.7 Clock Stretching in Enhanced SPI Transfers.... 428
4.4.5.3.8 Memory Mapped Mode (XIP Mode).... 429
4.4.5.3.8.1 Memory Mapped Read Model.... 429
4.4.5.3.8.2 Memory Mapped Write Model.... 429
4.4.5.3.8.3 Memory Mapped Space.... 430
4.4.5.3.8.4 Data Endian Mode for Memory Mapped Read Model.... 431
4.4.5.3.9 Enhanced Feature.... 431
4.4.5.3.9.1 Related Delay Between CS and SCK.... 431
4.4.5.4 Registers.... 432
4.4.5.4.1 CTRL0.... 432
4.4.5.4.2 CTRL1.... 435
4.4.5.4.3 QSPI_EN.... 436
4.4.5.4.4 MWC.... 436
4.4.5.4.5 SE.... 437
4.4.5.4.6 BAUD.... 438
4.4.5.4.7 TX_FTL.... 438
4.4.5.4.8 RX_FTL.... 439
4.4.5.4.9 TX_FL.... 440
4.4.5.4.10 RX_FL.... 440
4.4.5.4.11 STAT.... 441
4.4.5.4.12 INTMASK.... 442
4.4.5.4.13 INTSTAT.... 443
4.4.5.4.14 RAW_INTSTAT.... 445
4.4.5.4.15 TXEIC.... 446
4.4.5.4.16 RXOIC.... 446
4.4.5.4.17 RXUIC.... 447
4.4.5.4.18 MSTIC.... 447
4.4.5.4.19 INTCLR.... 448
4.4.5.4.20 DMAC.... 448
4.4.5.4.21 DMA_TDL.... 449
4.4.5.4.22 DMA_RDL.... 449
4.4.5.4.23 DATA.... 450
4.4.5.4.24 RX_SMP_DLY.... 450
4.4.5.4.25 SPI_CTRL0.... 451
4.4.5.4.26 XIP_MODE_BITS.... 452
4.4.5.4.27 XIP_INCR_INST.... 453
4.4.5.4.28 XIP_WRAP_INST.... 453
4.4.5.4.29 XIP_CTRL.... 454
4.4.5.4.30 XIP_SER.... 456
4.4.5.4.31 XIP_CNT_TIME_OUT.... 457

4.4.5.4.32 XIP_WR_INCR_INST 457
4.4.5.4.33 XIP_WR_WRAP_INST 458
4.4.5.4.34 XIP_WR_CTRL 458
4.4.5.4.35 QSPI_MODE_XIP 460
4.4.5.4.36 QSPI_CS_SETUP_DLY 462
4.4.5.4.37 QSPI_CS_RELEASE_DLY 463
4.4.5.5 Electrical Specifications 464
4.5 Audio Interface 464
4.5.1 PDM 465
4.5.1.1 Introduction 465
4.5.1.2 Main Features 465
4.5.1.3 Block Diagram 465
4.5.1.4 Protocols Introduction 466
4.5.1.4.1 Protocols Introduction 466
4.5.1.4.2 PCM Introduction 467
4.5.1.5 Functional Description 468
4.5.1.5.1 PDM Pin Mux 468
4.5.1.5.2 PDM Operation 468
4.5.1.5.3 PDM Gain 468
4.5.1.5.4 PDM Enable 469
4.5.1.5.5 PDM Interrupt 470
4.5.1.5.6 PDM DMA 470
4.5.1.5.7 Hardware Connection Example 470
4.5.1.6 Registers 471
4.5.1.6.1 EN_LEFT 471
4.5.1.6.2 PGA_CFG_LEFT 472
4.5.1.6.3 DATA_LEFT 472
4.5.1.6.4 INT_LEFT 473
4.5.1.6.5 EN_RIGHT 473
4.5.1.6.6 PGA_CFG_RIGHT 474
4.5.1.6.7 DATA_RIGHT 474
4.5.1.6.8 INT_RIGHT 475
4.5.1.6.9 DATA 475
4.5.1.6.10 CLK 475
4.5.1.7 Electrical Specifications 476
4.5.2 I2S 477
4.5.2.1 Introduction 477
4.5.2.2 Main Features 477
4.5.2.3 Protocols Introduction 478
4.5.2.3.1 Signals and Formats 478
4.5.2.3.2 Sampling and CLK frequency 479
4.5.2.4 Functional Description 479

4.5.2.4.1 I2S Pin Mux....479
4.5.2.4.2 Master and Slave....479
4.5.2.4.3 Clock Divider....480
4.5.2.4.4 I2S Enable....481
4.5.2.4.5 I2S as Transmitter....482
4.5.2.4.6 I2S as Receiver....482
4.5.2.4.7 I2S FIFO....483
4.5.2.4.8 I2S DMA....484
4.5.2.5 Registers....484
4.5.2.5.1 EN....484
4.5.2.5.2 RX_EN....484
4.5.2.5.3 TX_EN....485
4.5.2.5.4 CLK_EN....485
4.5.2.5.5 SCLK_CFG....486
4.5.2.5.6 RX_FIFO_RST....487
4.5.2.5.7 TX_FIFO_RST....487
4.5.2.5.8 LEFT_RX_BUF....488
4.5.2.5.9 LEFT_TX_HDG....488
4.5.2.5.10 RIGHT_RX_BUF....489
4.5.2.5.11 RIGHT_TX_HDG....490
4.5.2.5.12 RX_CH_EN....490
4.5.2.5.13 TX_CH_EN....491
4.5.2.5.14 RX_CFG....491
4.5.2.5.15 TX_CFG....492
4.5.2.5.16 INT_STAT....493
4.5.2.5.17 INT_MASK....494
4.5.2.5.18 RX_OVER....495
4.5.2.5.19 TX_OVER....495
4.5.2.5.20 RX_FIFO_CFG....496
4.5.2.5.21 TX_FIFO_CFG....497
4.5.2.5.22 RX_FIFO_FLUSH....498
4.5.2.5.23 TX_FIFO_FLUSH....498
4.5.2.5.24 RX_DMA....499
4.5.2.5.25 RST_RX_DMA....499
4.5.2.5.26 TX_DMA....500
4.5.2.5.27 RST_TX_DMA....500
4.5.2.5.28 CLK_CFG....501
4.5.2.6 Electrical Specifications....501
4.6 Security Interfaces....502
4.6.1 KEYRAM....502
4.6.2 Advanced Encryption Standard (AES)....502
4.6.2.1 Introduction....502

4.6.2.2 Main Features..... 503
4.6.2.3 Functional Description..... 503
4.6.2.3.1 Standard..... 503
4.6.2.3.2 Module Overview..... 503
4.6.2.3.3 Usage..... 504
4.6.2.4 Registers..... 505
4.6.2.4.1 CTRL..... 505
4.6.2.4.2 CFG..... 506
4.6.2.4.3 STAT..... 507
4.6.2.4.4 INT..... 508
4.6.2.4.5 XFER_SIZE..... 509
4.6.2.4.6 RD_START_ADDR..... 509
4.6.2.4.7 WR_START_ADDR..... 509
4.6.2.4.8 KEY_ADDR..... 510
4.6.2.4.9 DATA_OUT0..... 510
4.6.2.4.10 DATA_OUT1..... 510
4.6.2.4.11 DATA_OUT2..... 511
4.6.2.4.12 DATA_OUT3..... 511
4.6.2.4.13 KEY0..... 511
4.6.2.4.14 KEY1..... 512
4.6.2.4.15 KEY2..... 512
4.6.2.4.16 KEY3..... 512
4.6.2.4.17 KEY4..... 513
4.6.2.4.18 KEY5..... 513
4.6.2.4.19 KEY6..... 513
4.6.2.4.20 KEY7..... 514
4.6.2.4.21 INIT_SSI..... 514
4.6.2.4.22 INIT_SSO..... 514
4.6.2.4.23 MASK_SSI..... 515
4.6.2.4.24 MASK_SSO..... 515
4.6.2.4.25 INIT_V0..... 515
4.6.2.4.26 INIT_V1..... 516
4.6.2.4.27 INIT_V2..... 516
4.6.2.4.28 INIT_V3..... 516
4.6.2.4.29 DATA_IN0..... 517
4.6.2.4.30 DATA_IN1..... 517
4.6.2.4.31 DATA_IN2..... 517
4.6.2.4.32 DATA_IN3..... 518
4.6.2.4.33 KEYPORT_MASK..... 518
4.6.3 Hash-based Message Authentication Code (HMAC)..... 518
4.6.3.1 Introduction..... 518
4.6.3.2 Main Features..... 519

4.6.3.3 Functional Description..... 519
4.6.3.3.1 Standard.....519
4.6.3.3.2 Module Overview..... 519
4.6.3.3.3 Usage..... 520
4.6.3.4 Registers..... 521
4.6.3.4.1 CTRL..... 521
4.6.3.4.2 CFG.....521
4.6.3.4.3 STAT.....522
4.6.3.4.4 XFER_SIZE.....523
4.6.3.4.5 INT.....524
4.6.3.4.6 RD_START_ADDR.....524
4.6.3.4.7 WR_START_ADDR..... 525
4.6.3.4.8 USER_HASH_0.....525
4.6.3.4.9 USER_HASH_1.....525
4.6.3.4.10 USER_HASH_2.....526
4.6.3.4.11 USER_HASH_3.....526
4.6.3.4.12 USER_HASH_4.....526
4.6.3.4.13 USER_HASH_5.....527
4.6.3.4.14 USER_HASH_6.....527
4.6.3.4.15 USER_HASH_7.....528
4.6.3.4.16 DATA_OUT.....528
4.6.3.4.17 DATA_IN..... 528
4.6.3.4.18 KEY0..... 529
4.6.3.4.19 KEY1..... 529
4.6.3.4.20 KEY2..... 530
4.6.3.4.21 KEY3..... 530
4.6.3.4.22 KEY4..... 530
4.6.3.4.23 KEY5..... 531
4.6.3.4.24 KEY6..... 531
4.6.3.4.25 KEY7..... 531
4.6.3.4.26 KEY_ADDR.....532
4.6.3.4.27 KEYPORT_MASK.....532
4.6.4 Public Key Cryptography (PKC)..... 532
4.6.4.1 Introduction.....532
4.6.4.2 Main Features..... 533
4.6.4.3 Functional Description..... 533
4.6.4.3.1 Standard.....533
4.6.4.3.2 Module Overview..... 533
4.6.4.3.3 Usage..... 534
4.6.4.4 Registers..... 536
4.6.4.4.1 CTRL..... 536
4.6.4.4.2 CFG0.....536

4.6.4.4.3 CFG1....537
4.6.4.4.4 CFG2....537
4.6.4.4.5 CFG3....538
4.6.4.4.6 CFG4....538
4.6.4.4.7 CFG5....538
4.6.4.4.8 CFG6....539
4.6.4.4.9 CFG7....539
4.6.4.4.10 CFG8....540
4.6.4.4.11 CFG9....540
4.6.4.4.12 CFG10....541
4.6.4.4.13 CFG11....541
4.6.4.4.14 CFG12....541
4.6.4.4.15 CFG13....542
4.6.4.4.16 SW_CTRL....542
4.6.4.4.17 SW_CFG0....543
4.6.4.4.18 SW_CFG1....543
4.6.4.4.19 SW_CFG2....544
4.6.4.4.20 SW_CFG3....544
4.6.4.4.21 SW_CFG4....545
4.6.4.4.22 SW_CFG5....545
4.6.4.4.23 SW_CFG6....546
4.6.4.4.24 SW_CFG7....546
4.6.4.4.25 SW_CFG8....546
4.6.4.4.26 SW_CFG9....547
4.6.4.4.27 SW_CFG10....547
4.6.4.4.28 SW_CFG11....547
4.6.4.4.29 SW_CFG12....548
4.6.4.4.30 SW_CFG13....548
4.6.4.4.31 INT_STAT....549
4.6.4.4.32 INT_EN....549
4.6.4.4.33 STAT....550
4.6.5 True Random Number Generator (TRNG)....550
4.6.5.1 Introduction....550
4.6.5.2 Main Features....550
4.6.5.3 Functional Description....550
4.6.5.3.1 Standard....550
4.6.5.3.2 Module Overview....550
4.6.5.3.3 Usage....552
4.6.5.4 Registers....553
4.6.5.4.1 CTRL....553
4.6.5.4.2 STAT....553
4.6.5.4.3 DATA....554

4.6.5.4.4 LONG_RUN_STAT.....554
4.6.5.4.5 CFG.....554
4.6.5.4.6 SRC_CFG.....556
4.6.5.4.7 FRO_CFG.....557
4.6.5.4.8 USER_SEED.....557
4.6.5.4.9 LONG_RUN_CFG.....558
4.7 Analog Interfaces.....558
4.7.1 13-Bit SNSADC.....558
4.7.1.1 Introduction.....558
4.7.1.2 Main Features.....558
4.7.1.3 Block Diagram.....559
4.7.1.4 Resource Sharing.....559
4.7.1.5 Functional Description.....559
4.7.1.5.1 Raw Data Stream.....560
4.7.1.5.2 Sample Operation Mode.....560
4.7.1.5.3 Clock and Sampling Rate.....561
4.7.1.5.4 Considerations for Using SNSADC.....561
4.7.1.5.5 Measuring MSIO Input.....562
4.7.1.5.6 Measuring Battery Voltage.....563
4.7.1.5.7 Measuring Internal Temperature.....564
4.7.1.5.8 Calibration.....565
4.7.1.6 Registers.....567
4.7.1.6.1 SNSADC_CFG.....567
4.7.1.6.2 SENSE_ADC_FIFO.....569
4.7.1.6.3 SENSE_FF_THRESH.....569
4.7.1.6.4 SENSE_ADC_STAT.....570
4.7.1.6.5 SENSE_ADC_CLK.....570
4.7.1.6.6 SENSE_ADC_GET_TKN_HW.....571
4.7.1.6.7 SENSE_ADC_GET_TKN_SW.....571
4.7.1.6.8 SENSE_ADC_RET_TKN_HW.....572
4.7.1.6.9 SENSE_ADC_RET_TKN_SW.....572
4.7.1.6.10 SENSE_ADC_GET_TKN_STS.....572
4.7.1.7 Electrical Specifications.....573
4.7.2 Temperature Sensor.....574
4.7.2.1 Introduction.....574
4.7.2.2 Main Features.....574
4.7.2.3 Functional Description.....574
4.7.2.3.1 Acquisition Parameter Configuration.....574
4.7.2.3.2 Conversion Formula.....574
4.7.2.3.3 Measuring Internal Temperature.....574
4.7.2.4 Registers.....575
4.7.3 Comparator.....575

4.7.3.1 Introduction....575
4.7.3.2 Main Features....575
4.7.3.3 Functional Description....576
4.7.3.3.1 Overview....576
4.7.3.3.2 Hysteresis....578
4.7.3.3.3 Res Degeneration....578
4.7.3.3.4 CascRes Half....578
4.7.3.3.5 Bias Current Set....579
4.7.3.3.6 Glitch Remove....579
4.7.3.4 Registers....579
4.7.3.4.1 COMP_REG_0....579
4.7.3.4.2 COMP_REG_1....580
4.7.3.4.3 MCU_WKUP_CTRL....581
4.7.3.4.4 AON_SLP_EVENT....582
4.7.3.4.5 PMU_COMP_GLITCH_REMOVE....582
4.7.3.5 Electrical Specifications....582
4.8 DMA....583
4.8.1 Introduction....583
4.8.2 Main Features....583
4.8.3 Block Diagram....583
4.8.4 Functional Description....585
4.8.4.1 Transfer Type Flow Control....585
4.8.4.2 Transfer Width....585
4.8.4.3 Source and Destination Address Increment....585
4.8.4.4 Channel Priority....586
4.8.4.5 DMA Transfers....586
4.8.4.6 Gather and Scatter....587
4.8.4.7 Multi-block DMA Transfer....590
4.8.4.7.1 Auto-reloading....590
4.8.4.7.2 Linked Lists (Block Chaining)....591
4.8.4.8 DMA Interrupt....592
4.8.4.9 DMA Hardware Interface Assignment....593
4.8.5 Registers....594
4.8.5.1 Channel_x_Registers....594
4.8.5.1.1 SARx....594
4.8.5.1.2 DARx....595
4.8.5.1.3 LLPx....595
4.8.5.1.4 CTL_LOx....596
4.8.5.1.5 CTL_HIx....600
4.8.5.1.6 SSTATx....601
4.8.5.1.7 DSTATx....602
4.8.5.1.8 SSTATARx....603

4.8.5.1.9 DSTATARx....603
4.8.5.1.10 CFG_LOx....604
4.8.5.1.11 CFG_HIx....606
4.8.5.1.12 SGRx....609
4.8.5.1.13 DSRx....609
4.8.5.2 Interrupt_Registers....610
4.8.5.2.1 INT_RSTAT_TC....610
4.8.5.2.2 INT_RSTAT_BTC....610
4.8.5.2.3 INT_RSTAT_STC....611
4.8.5.2.4 INT_RSTAT_DTC....612
4.8.5.2.5 INT_RSTAT_ERR....612
4.8.5.2.6 INT_STAT_TC....613
4.8.5.2.7 INT_STAT_BTC....613
4.8.5.2.8 INT_STAT_STC....614
4.8.5.2.9 INT_STAT_DTC....614
4.8.5.2.10 INT_STAT_ERR....615
4.8.5.2.11 INT_MASK_TC....615
4.8.5.2.12 INT_MASK_BTC....616
4.8.5.2.13 INT_MASK_STC....617
4.8.5.2.14 INT_MASK_DTC....618
4.8.5.2.15 INT_MASK_ERR....618
4.8.5.2.16 INT_CLR_TC....619
4.8.5.2.17 INT_CLR_BTC....620
4.8.5.2.18 INT_CLR_STC....620
4.8.5.2.19 INT_CLR_DTC....621
4.8.5.2.20 INT_CLR_ERR....621
4.8.5.2.21 INT_STAT_ET....622
4.8.5.3 Software_Handshake_Registers....623
4.8.5.3.1 REQ_SST....623
4.8.5.3.2 REQ_DST....623
4.8.5.3.3 REQ_SGL_ST....624
4.8.5.3.4 REQ_SGL_DT....625
4.8.5.3.5 REQ_LST_ST....625
4.8.5.3.6 REQ_LST_DT....626
4.8.5.4 Miscellaneous_Registers....627
4.8.5.4.1 CFG....627
4.8.5.4.2 CH_EN....627

5 Communication Subsystem....629

5.1 Supported Features....629
5.2 Transceiver....629
5.3 Digital Bluetooth LE Subsystem....629
5.4 Electrical Specifications....630

5.4.1 Receiver Performance....630
5.4.2 Transmitter Performance....631

6 Package Information....632

6.1 QFN68....632
6.2 QFN56....634

7 Absolute Maximum Ratings....637

8 Recommended Operating Conditions....638

9 Ordering Codes....639

10 Tape and Reel Information....640

11 Glossary....642

12 Reference Documents....645

13 Legal and Contact Information....646

14 Revision History....647

1 Product Overview

The Goodix GR5525 series is a single-mode, low-power Bluetooth 5.3 System-on-Chip (SoC). It can be configured as a Broadcaster, an Observer, a Central, and a Peripheral and supports the combination of all the above roles, making it an ideal choice for Internet of Things (IoT) and smart wearable devices.

Based on Arm® Cortex®-M4F CPU core running at 96 MHz, the GR5525 integrates a 2.4 GHz RF transceiver, Bluetooth LE 5.3 protocol stack, on-chip programmable Flash memory, 256 KB system SRAM, and a rich set of peripherals, enhanced I2C/UART port number for sensor applications, as well as expanded I/O functionality. The GR5525 MCU enhances QSPI interface to support external RAM (PSRAM) to accommodate display while still leaving plenty of space for wearable schemes.

GR5525 series are offered in both QFN68 and QFN56 packages. The table below lists the detailed package configurations.

Table 1-1 GR5525 series

Part Number	GR5525RGNI	GR5525IENI	GR5525I0NI
CPU	Cortex®-M4F	Cortex®-M4F	Cortex®-M4F
RAM	256 KB	256 KB	256 KB
SiP Flash	1 MB	512 KB	N/A
I/O number	50	39	39
I/O voltage	1.8 V–3.6 V	1.8 V–3.6 V	In line with Flash voltage
Package (mm)	QFN68 (7.0 x 7.0 x 0.85)	QFN56 (7.0 x 7.0 x 0.75)	QFN56 (7.0 x 7.0 x 0.75)

1.1 Features

- Bluetooth LE 5.3 transceiver
  - Supported data rates: 1 Mbps, 2 Mbps, and Long Range (500 kbps, 125 kbps)
  - TX power: –20 dBm to +7 dBm
  - RX sensitivity
    - –97 dBm sensitivity @ 1 Mbps mode
    - –93 dBm sensitivity @ 2 Mbps mode
    - –101 dBm sensitivity @ Long Range 500 kbps mode
    - –103 dBm sensitivity @ Long Range 125 kbps mode
  - Power consumption at 3.3 V VBAT input:
    - 6.3 mA TX current @ 0 dBm output power (64 MHz system clock)
    - 5.3 mA RX current @ 1 Mbps (64 MHz system clock)
- Arm® Cortex®-M4F 32-bit micro-processor with floating point support

Copyright © 2024 Shenzhen Goodix Technology Co., Ltd.

- Up to 96 MHz clock frequency
- Built-in Memory Protection Unit (MPU) supporting eight programmable regions
- Hardware Floating Point Unit (FPU)
- Built-in Nested Vectored Interrupt Controller (NVIC)
- Non-maskable Interrupt (NMI) input
- Serial Wire Debug (SWD) with 16 breakpoints, two watchpoints, and a debug timestamp counter
- 56 µA/MHz CoreMark running from Flash @ 3.3 V, 64 MHz from HFXO

- On-chip memory
  - 256 KB RAM data memory with retention capabilities
  - 8 KB cache RAM instruction memory with retention capabilities
  - Stack ROM (including boot ROM and Bluetooth LE Stack)
  - 1 MB internal QSPI Flash (512 KB for GR5525IENI, and external Flash for GR5525I0NI)
- Digital peripherals
  - 2 x general-purpose DMA engines with six channels and up to 16 programmable request/trigger sources
- Analog peripherals
  - 1 x 13-bit Sense ADC with sampling rate of 1 Msps, supporting up to eight external I/O channels and three internal signal channels
  - Built-in die temperature and voltage sensors
  - Low-power comparator, supporting wakeup from sleep mode
- Flexible serial peripherals
  - 4 x UART modules up to 2 Mbps with flow control and IrDA features
  - 4 x I2C modules for peripheral communication, up to 3.4 MHz, operating as either Master or Slave
  - 2 x I2S interfaces (1 x I2S master interface and 1 x I2S slave interface)
  - PDM interface with hardware sampling rate converter
  - 1 x 8-bit/16-bit/32-bit SPI master interface and 1 x SPI slave interface for host communication
  - 1 x Dual-lane SPI (DSPI) interface for display, with MIPI DBI Type-C support
  - 3 x Quad SPI (QSPI) interfaces, up to 48 MHz; supporting direct access via memory mapping when connecting with external NOR Flash
- Security
  - Complete secure computing engine:
    - AES 128-bit/192-bit/256-bit symmetric encryption (ECB, CBC)

    - Hash-based Message Authentication Code (HMAC-SHA256)
    - Public key cryptography (PKC)
    - True random number generator (TRNG)
  - Comprehensive security operation mechanism:
    - Secure boot
    - Encrypted firmware running directly from Flash
    - eFuse for encrypted key storage
    - Differentiate application data key and firmware key, supporting one data key per device/product
- I/O peripherals
  - Up to 50 multiplexed I/O pins in total
    - Up to 34 general-purpose I/O (GPIO) pins with configurable pull-up/pull-down resistors
    - Up to eight always-on I/O (AON I/O) pins, supporting wakeup from sleep mode
    - Up to eight mixed signal I/O (MSIO) pins, configurable to be digital/analog signal interfaces
- Timers
  - 2 x 32-bit general-purpose timers
  - 1 x dual timer with two programmable 32-bit or 16-bit down counters
  - 1 x sleep timer for waking the device up from sleep mode
  - 2 x 3-channel PWMs with edge-aligned and center-aligned modes
  - 2 x real-time counters (1 x Calendar, 1 x real-time counter)
- Power management
  - On-chip DC-DC to provide RF analog voltage and supply core LDO
  - On-chip I/O LDO to provide I/O voltage and supply external components
  - Programmable thresholds for brownout detector (BOD)
  - Supply voltage: 2.4 V–3.8 V
  - I/O voltage: 1.8 V–3.6 V
- Low-power consumption
  - Sleep mode: 7.3 µA (Typical) at 3.3 V VBAT input, wakeup sources from always-on domain, and LFXO_32K running
  - Ultra deep sleep mode: 5.0 µA (Typical), with no memory data in retention and wakeup sources from SLP Timer or AON I/Os

Copyright © 2024 Shenzhen Goodix Technology Co., Ltd.

- OFF mode: 200 nA (Typical), with system in reset mode

- Operating temperature range: –40°C to +85°C
- Packages
  - QFN68: 7.0 x 7.0 x 0.85 mm, 0.35 mm pitch
  - QFN56: 7.0 x 7.0 x 0.75 mm, 0.4 mm pitch

1.2 Block Diagram

The SoC block diagram is shown in the figure below.

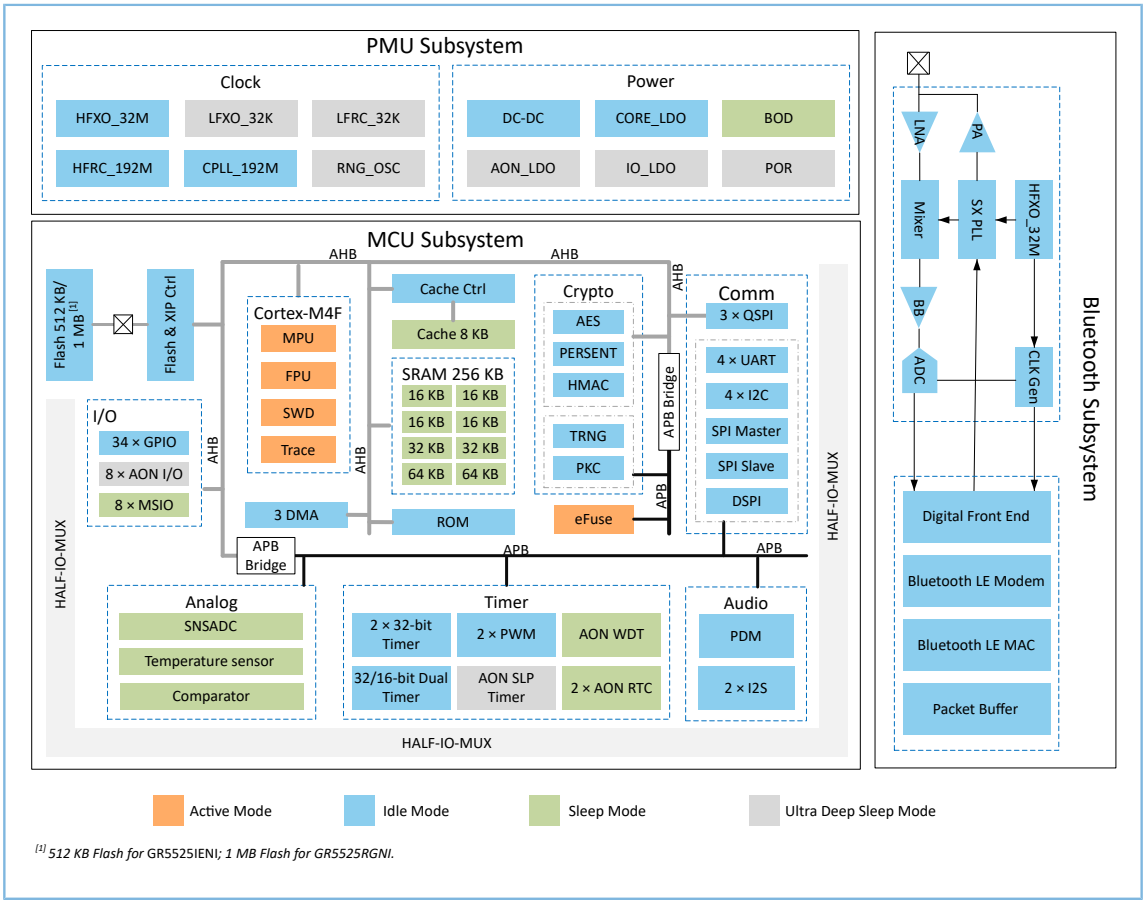

Figure 1-1 Block diagram

- Bluetooth subsystem
  - A 2.4 GHz transceiver and a digital communication core, both supporting Bluetooth LE 5.3
- MCU subsystem
  - An Arm® Cortex®-M4F with all the required memories and peripherals
  - Security cores that can be used for application security and secure boot implementation
- Power Management Unit (PMU) subsystem

- Power supply for the whole SoC, including internal modules and external peripherals
- Efficient and reliable SoC power management

- Clock Generation Unit (CGU) subsystem
  - Providing stable, low-power, and accurate clock signals
  - Optimizing SoC performance and minimizing power consumption, by adjusting clock frequency and selecting an appropriate clock

1.3 Applications

The device can be used in a rich set of applications.

- Advanced wearables
  - Sport bracelet
  - Smart watch
- Bluetooth HID devices
  - Voice remote control
  - Keyboard/Mouse
  - Gaming controller
  - Stylus pen
- IoT applications
  - Smart lock and smart home
  - Beacon and smart tracker
  - Electronic shelf label (ESL)
  - Mesh applications
  - Asset tracking

Copyright © 2024 Shenzhen Goodix Technology Co., Ltd.

2 Pinout

This chapter describes the pin assignment of the device and provides detailed information for each individual pin.

2.1 QFN68

The figure below shows the pin assignment for devices in QFN68 package.

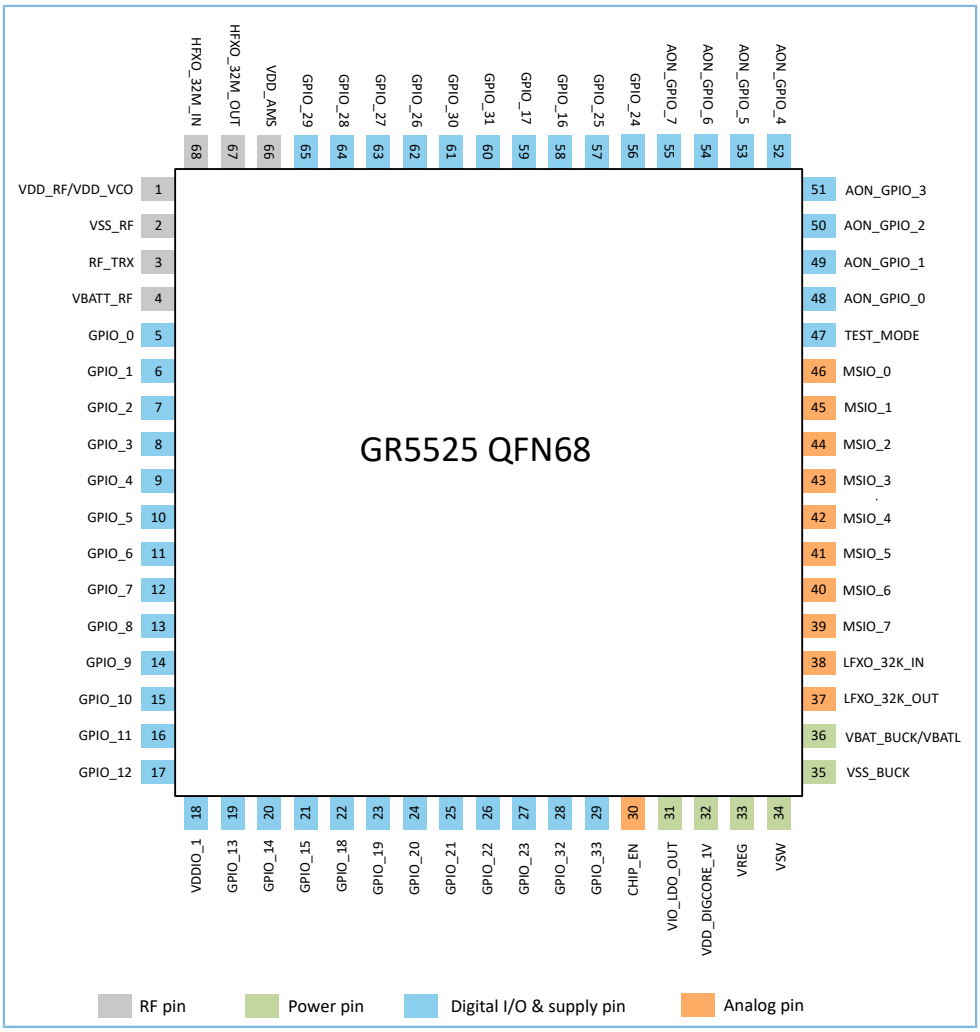

Figure 2-1 QFN68 pinout from top view

The following table provides the descriptions of pin functionality.

Table 2-1 Pin description

Pin #	Pin Name	Pin Type	Description/Default Function	Voltage Domain
1	VDD_RF/VDD_VCO	Analog/RF supply	Synthesizer VCO supply/RF supply: 1.1 V; connect to VREG	
2	VSS_RF	Analog/RF	RF GND; connect to GND	
3	RF_TRX	Analog/RF	RF transceiver RX input and TX output	
4	VBATT_RF	Analog/RF Supply	RF SPA and Bandgap supply; connect to VBATL	

Pin #	Pin Name	Pin Type	Description/Default Function	Voltage Domain
5	GPIO_0	Digital I/O	General purpose I/O; default: SWD_CLK	VDDIO1
6	GPIO_1	Digital I/O	General purpose I/O; default: SWD_IO	
7	GPIO_2	Digital I/O	General purpose I/O, configurable to be an SWO interface	
8	GPIO_3	Digital I/O	General purpose I/O	
9	GPIO_4	Digital I/O		
10	GPIO_5	Digital I/O		
11	GPIO_6	Digital I/O		
12	GPIO_7	Digital I/O		
13	GPIO_8	Digital I/O		
14	GPIO_9	Digital I/O		
15	GPIO_10	Digital I/O		
16	GPIO_11	Digital I/O		
17	GPIO_12	Digital I/O		
18	VDDIO_1	Digital I/O supply	Digital I/O supply input. Support external 1.8 V–3.3 V input voltage.	
19	GPIO_13	Digital I/O	General purpose I/O	
20	GPIO_14	Digital I/O		
21	GPIO_15	Digital I/O		
22	GPIO_18	Digital I/O		
23	GPIO_19	Digital I/O		
24	GPIO_20	Digital I/O		
25	GPIO_21	Digital I/O		
26	GPIO_22	Digital I/O		
27	GPIO_23	Digital I/O		
28	GPIO_32	Digital I/O		
29	GPIO_33	Digital I/O		
30	CHIP_EN	Analog/PMU	Master Enable for chip reset pin. The high level of CHIP_EN equals VBATL. Minimum value of high level for CHIP_EN: 1 V	
31	VIO_LDO_OUT	PMU	Output of on-chip I/O supply regulator, connect internally to VDDIO0. Typical output: 1.8 V	VDDIO0
32	VDD_DIGCORE_1V	Analog/PMU	On-chip LDO output for digital core	
33	VREG	Analog/PMU	DC-DC feedback pin of switch regulator	

Pin #	Pin Name	Pin Type	Description/Default Function	Voltage Domain
34	VSW	Analog/PMU	DC-DC converter switching node	
35	VSS_BUCK	Analog/PMU	DC-DC converter power GND	
36	VBAT_BUCK/VBATL	Analog/PMU	Power supply: 2.4 V to 3.8 V	
37	LFXO_32K_OUT	PMU	Output of inverting amplifier connected to 32.768 kHz crystal	
38	LFXO_32K_IN	PMU	Input of inverting amplifier connected to 32.768 kHz crystal	
39	MSIO_7	Mixed Signal I/O	Configurable mixed-signal I/O with digital I/O and analog I/O	VBATL
40	MSIO_6	Mixed Signal I/O		
41	MSIO_5	Mixed Signal I/O		
42	MSIO_4	Mixed Signal I/O		
43	MSIO_3	Mixed Signal I/O		
44	MSIO_2	Mixed Signal I/O		
45	MSIO_1	Mixed Signal I/O		
46	MSIO0	Mixed Signal I/O		
47	TEST_MODE	Analog/RF	Input pin, used for factory test mode selection • 1: factory test mode • 0: normal operation mode **Note:** In practice, the value is set to 0 and the pin is connected to GND by default.	
48	AON_GPIO_0	Digital I/O	Always-on GPIO; can wake up chip from sleep modes	VDDIO0
49	AON_GPIO_1	Digital I/O		
50	AON_GPIO_2	Digital I/O		
51	AON_GPIO_3	Digital I/O		
52	AON_GPIO_4	Digital I/O		
53	AON_GPIO_5	Digital I/O		
54	AON_GPIO_6	Digital I/O		
55	AON_GPIO_7	Digital I/O		
56	GPIO_24	Digital I/O	General purpose I/O	
57	GPIO_25	Digital I/O		
58	GPIO_16	Digital I/O		
59	GPIO_17	Digital I/O		
60	GPIO_31	Digital I/O		
61	GPIO_30	Digital I/O		
62	GPIO_26	Digital I/O		

Pin #	Pin Name	Pin Type	Description/Default Function	Voltage Domain
63	GPIO_27	Digital I/O		
64	GPIO_28	Digital I/O		
65	GPIO_29	Digital I/O		
66	VDD_AMS	Analog/RF	AMS supply: 1.1 V. Connect to VREG.	
67	HFXO_32M_OUT	Analog/RF	Output of inverting amplifier connected to 32 MHz crystal	
68	HFXO_32M_IN	Analog/RF	Input of inverting amplifier connected to 32 MHz crystal	

Note:

- The pad drive strength for GPIO and AON_GPIO pins can be configured to 4 mA under 3.3 V voltage.
- The pad drive strength for MSIO pins can be configured to 2 mA under 3.3 V voltage.

2.2 QFN56 (With Flash)

The figure below shows the pin assignment for devices in QFN56 package (with Flash).

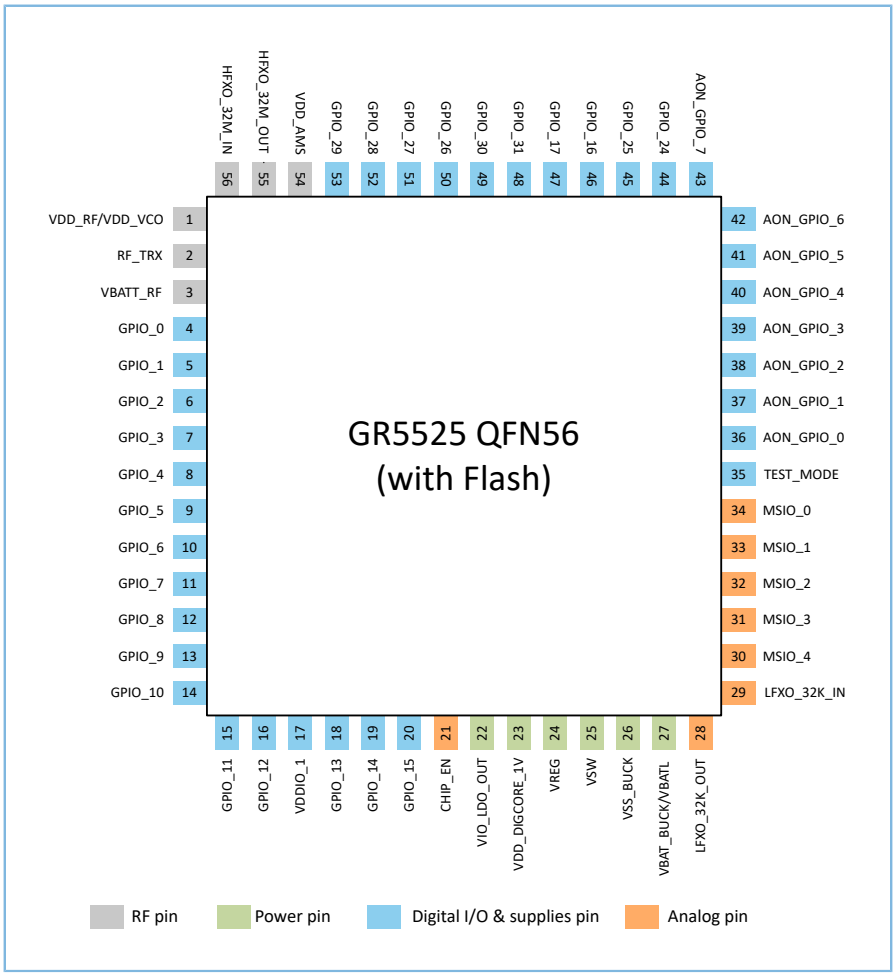

Figure 2-2 QFN56 (with Flash) pinout from top view

The following table provides the descriptions of pin functionality.

Table 2-2 Pin description

Pin #	Pin Name	Pin Type	Description/Default Function	Voltage Domain
1	VDD_RF/VDD_VCO	Analog/RF supply	Synthesizer VCO supply/RF supply: 1.1 V; connect to VREG	
2	RF_TRX	Analog/RF	RF transceiver RX input and TX output	
3	VBATT_RF	Analog/RF Supply	RF SPA and Bandgap supply; connect to VBATL	
4	GPIO_0	Digital I/O	General purpose I/O; default: SWD_CLK	VDDIO1
5	GPIO_1	Digital I/O	General purpose I/O; default: SWD_IO	
6	GPIO_2	Digital I/O	General purpose I/O	
7	GPIO_3	Digital I/O		
8	GPIO_4	Digital I/O		
9	GPIO_5	Digital I/O		
10	GPIO_6	Digital I/O		
11	GPIO_7	Digital I/O		
12	GPIO_8	Digital I/O		
13	GPIO_9	Digital I/O		
14	GPIO_10	Digital I/O		
15	GPIO_11	Digital I/O		
16	GPIO_12	Digital I/O		
17	VDDIO_1	Digital I/O supply	Digital I/O supply input. Support external 1.8 V–3.3 V input voltage.	
18	GPIO_13	Digital I/O	General purpose I/O	
19	GPIO_14	Digital I/O		
20	GPIO_15	Digital I/O		
21	CHIP_EN	Analog/PMU	Master Enable for chip reset pin. The high level of CHIP_EN equals VBATL. Minimum value of high level for CHIP_EN: 1 V	
22	VIO_LDO_OUT	PMU	Output of on-chip I/O supply regulator, connect internally to VDDIO0. Typical output: 1.8 V	VDDIO0
23	VDD_DIGCORE_1V	Analog/PMU	On-chip LDO output for digital core	
24	VREG	Analog/PMU	DC-DC feedback pin of switch regulator	
25	VSW	Analog/PMU	DC/DC converter switching node	
26	VSS_BUCK	Analog/PMU	DC/DC converter supply and general battery GND	
27	VBAT_BUCK/VBATL	Analog/PMU	Power supply: 2.4 V to 3.8 V	

Pin #	Pin Name	Pin Type	Description/Default Function	Voltage Domain
28	LFXO_32K_OUT	PMU	Output of inverting amplifier connected to 32.768 kHz crystal	
29	LFXO_32K_IN	PMU	Input of inverting amplifier connected to 32.768 kHz crystal	
30	MSIO_4	Mixed Signal I/O	Configurable mixed-signal I/O with digital I/O and analog I/O	VBATL
31	MSIO_3	Mixed Signal I/O		
32	MSIO_2	Mixed Signal I/O		
33	MSIO_1	Mixed Signal I/O		
34	MSIO_0	Mixed Signal I/O		
35	TEST_MODE	Analog/RF	Input pin, used for factory test mode selection • 1: factory test mode • 0: normal operation mode **Note:** In practice, the value is set to 0 and the pin is connected to GND by default.	
36	AON_GPIO_0	Digital I/O	Always-on GPIO; can wake up chip from sleep modes	VDDIO0
37	AON_GPIO_1	Digital I/O		
38	AON_GPIO_2	Digital I/O		
39	AON_GPIO_3	Digital I/O		
40	AON_GPIO_4	Digital I/O		
41	AON_GPIO_5	Digital I/O		
42	AON_GPIO_6	Digital I/O		
43	AON_GPIO_7	Digital I/O		
44	GPIO_24	Digital I/O	General purpose I/O	
45	GPIO_25	Digital I/O		
46	GPIO_16	Digital I/O		
47	GPIO_17	Digital I/O		
48	GPIO_31	Digital I/O		
49	GPIO_30	Digital I/O		
50	GPIO_26	Digital I/O		
51	GPIO_27	Digital I/O		
52	GPIO_28	Digital I/O		
53	GPIO_29	Digital I/O		
54	VDD_AMS	Analog/RF	AMS supply: 1.1 V. Connect to VREG.	
55	HFXO_32M_OUT	Analog/RF	Output of inverting amplifier connected to 32 MHz crystal	

Pin #	Pin Name	Pin Type	Description/Default Function	Voltage Domain
56	HFXO_32M_IN	Analog/RF	Input of inverting amplifier connected to 32 MHz crystal	

Note:

- The pad drive strength for GPIO and AON_GPIO pins can be configured to 4 mA under 3.3 V voltage.
- The pad drive strength for MSIO pins can be configured to 2 mA under 3.3 V voltage.

2.3 QFN56 (Without Flash)

The figure below shows the pin assignment for devices in QFN56 package (without Flash).

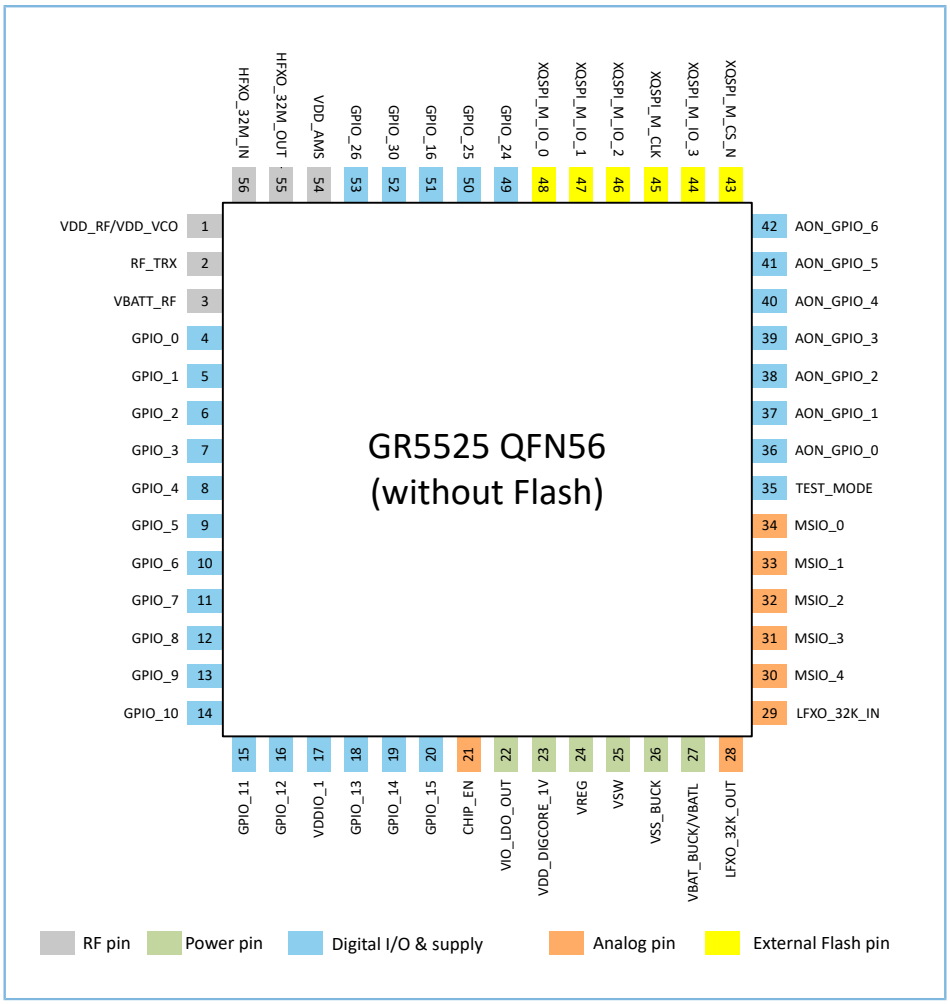

Figure 2-3 QFN56 (without Flash) pinout from top view

The following table provides the descriptions of pin functionality.

Table 2-3 Pin description

Pin #	Pin Name	Pin Type	Description/Default Function	Voltage Domain
1	VDD_RF/VDD_VCO	Analog/RF supply	Synthesizer VCO supply/RF supply: 1.1 V; connect to VREG	
2	RF_TRX	Analog/RF	RF transceiver RX input and TX output	
3	VBATT_RF	Analog/RF Supply	RF SPA and bandgap supply; connect to VBATL	
4	GPIO_0	Digital I/O	General purpose I/O; default: SWD_CLK	VDDIO1
5	GPIO_1	Digital I/O	General purpose I/O; default: SWD_IO	
6	GPIO_2	Digital I/O	General purpose I/O	
7	GPIO_3	Digital I/O		
8	GPIO_4	Digital I/O		
9	GPIO_5	Digital I/O		
10	GPIO_6	Digital I/O		
11	GPIO_7	Digital I/O		
12	GPIO_8	Digital I/O		
13	GPIO_9	Digital I/O		
14	GPIO_10	Digital I/O		
15	GPIO_11	Digital I/O		
16	GPIO_12	Digital I/O		
17	VDDIO_1	Digital I/O supply	Digital I/O supply input. Support external 1.8 V–3.3 V input voltage.	
18	GPIO_13	Digital I/O	General purpose I/O	
19	GPIO_14	Digital I/O		
20	GPIO_15	Digital I/O		
21	CHIP_EN	Analog/PMU	Master Enable for chip reset pin. The high level of CHIP_EN equals VBATL. Minimum value of high level for CHIP_EN: 1 V	
22	VIO_LDO_OUT	PMU	Output of on-chip I/O supply regulator, connect internally to VDDIO0. Typical output: 1.8 V	VDDIO0
23	VDD_DIGCORE_1V	Analog/PMU	On-chip LDO output for digital core	
24	VREG	Analog/PMU	DC-DC feedback pin of switch regulator	
25	VSW	Analog/PMU	DC-DC converter switching node	
26	VSS_BUCK	Analog/PMU	DC-DC converter supply and general battery GND	
27	VBAT_BUCK/VBATL	Analog/PMU	Power supply: 2.4 V to 3.8 V	
28	LFXO_32K_OUT	PMU	Output of inverting amplifier connected to 32.768 kHz crystal	

Pin #	Pin Name	Pin Type	Description/Default Function	Voltage Domain
29	LFXO_32K_IN	PMU	Input of inverting amplifier connected to 32.768 kHz crystal	
30	MSIO_4	Mixed Signal I/O	Configurable mixed-signal I/O with digital I/O and analog I/O	VBATL
31	MSIO_3	Mixed Signal I/O		
32	MSIO_2	Mixed Signal I/O		
33	MSIO_1	Mixed Signal I/O		
34	MSIO_0	Mixed Signal I/O		
35	TEST_MODE	Analog/RF	Input pin, used for factory test mode selection • 1: factory test mode • 0: normal operation mode **Note:** In practice, the value is set to 0 and the pin is connected to GND by default.	
36	AON_GPIO_0	Digital I/O	Always-on GPIO; can wake up chip from sleep modes	VDDIO0
37	AON_GPIO_1	Digital I/O		
38	AON_GPIO_2	Digital I/O		
39	AON_GPIO_3	Digital I/O		
40	AON_GPIO_4	Digital I/O		
41	AON_GPIO_5	Digital I/O		
42	AON_GPIO_6	Digital I/O		
43	XQSPI_M_CS_N	Digital I/O	Connected to an external Flash	
44	XQSPI_M_IO_3	Digital I/O		
45	XQSPI_M_CLK	Digital I/O		
46	XQSPI_M_IO_2	Digital I/O		
47	XQSPI_M_IO_1	Digital I/O		
48	XQSPI_M_IO_0	Digital I/O		
49	GPIO_24	Digital I/O	General purpose I/O	
50	GPIO_25	Digital I/O		
51	GPIO_16	Digital I/O		
52	GPIO_30	Digital I/O		
53	GPIO_26	Digital I/O		
54	VDD_AMS	Analog/RF	AMS supply: 1.1 V. Connect to VREG.	
55	HFXO_32M_OUT	Analog/RF	Output of inverting amplifier connected to 32 MHz crystal	
56	HFXO_32M_IN	Analog/RF	Input of inverting amplifier connected to 32 MHz crystal	

Note:

- The pad drive strength for GPIO and AON_GPIO pins can be configured to 4 mA under 3.3 V voltage.
- The pad drive strength for MSIO pins can be configured to 2 mA under 3.3 V voltage.

2.4 Pin Properties

2.4.1 PMU Pin Properties

Table 2-4 I/O properties

No.	Pin Name	Input/Output/GND	Min.	Typ.	Max.	Unit	Remark
1	VDD_RF/ VDD_VCO	Input	1	1.05	1.35	V	
2	VSS_RF	GND		0		V	
3	VBATT_RF	Input	2.4	3.3	3.8	V	
4	CHIP_EN	Input			3.8	V	
5	VIO_LDO_OUT	Output	1.6	1.8	3.6	V	
6	VDD_DIGCORE_1V	Output/Input	0.85	0.95	1.3	V	
7	VREG	Output	0.9	1.05	1.35	V	
8	VSW	Output				V	
9	VSS_BUCK	GND		0		V	
10	VBAT_BUCK/ VBATL	Input	2.4	3.3	3.8	V	
11	VDD_AMS	Input	1	1.05	1.35	V	
12	VDDIO_1	Input	1.8		3.3	V	
13	TEST_MODE	Output		0		V	

2.4.2 I/O Pin Properties

Table 2-5 I/O properties

No.	Pin Name	Default Status After POR	Pull-up/Pull-down Enable After POR	Pull-up/Pull-down Selection After POR	Pin Interrupt	Chip Wakeup	Fast Capability
1	GPIO_0	L	Y	PD	Y	N	Y
2	GPIO_1	L	Y	PD	Y	N	Y
3	GPIO_2	L	Y	PD	Y	N	Y
4	GPIO_3	L	Y	PD	Y	N	Y

No.	Pin Name	Default Status After POR	Pull-up/Pull-down Enable After POR	Pull-up/Pull-down Selection After POR	Pin Interrupt	Chip Wakeup	Fast Capability
5	GPIO_4	L	Y	PD	Y	N	Y
6	GPIO_5	L	Y	PD	Y	N	Y
7	GPIO_6	L	Y	PD	Y	N	Y
8	GPIO_7	L	Y	PD	Y	N	Y
9	GPIO_8	L	Y	PD	Y	N	Y
10	GPIO_9	L	Y	PD	Y	N	Y
11	GPIO_10	L	Y	PD	Y	N	Y
12	GPIO_11	L	Y	PD	Y	N	Y
13	GPIO_12	L	Y	PD	Y	N	Y
14	GPIO_13	L	Y	PD	Y	N	Y
15	GPIO_14	L	Y	PD	Y	N	Y
16	GPIO_15	L	Y	PD	Y	N	Y
17	GPIO_18	L	Y	PD	Y	N	Y
18	GPIO_19	L	Y	PD	Y	N	Y
19	GPIO_20	L	Y	PD	Y	N	Y
20	GPIO_21	L	Y	PD	Y	N	Y
21	GPIO_22	L	Y	PD	Y	N	Y
22	GPIO_23	L	Y	PD	Y	N	Y
23	GPIO_32	L	Y	PD	Y	N	Y
24	GPIO_33	L	Y	PD	Y	N	Y
25	MSIO_7	Hi-Z	N		N	N	Y
26	MSIO_6	Hi-Z	N		N	N	Y
27	MSIO_5	Hi-Z	N		N	N	Y
28	MSIO_4	Hi-Z	N		N	N	Y
29	MSIO_3	Hi-Z	N		N	N	Y
30	MSIO_2	Hi-Z	N		N	N	Y
31	MSIO_1	Hi-Z	N		N	N	Y
32	MSIO_0	Hi-Z	N		N	N	Y
33	AON_GPIO_0	L	Y	PD	Y	Y	Y
34	AON_GPIO_1	L	Y	PD	Y	Y	Y
35	AON_GPIO_2	L	Y	PD	Y	Y	Y
36	AON_GPIO_3	L	Y	PD	Y	Y	Y

No.	Pin Name	Default Status After POR	Pull-up/Pull-down Enable After POR	Pull-up/Pull-down Selection After POR	Pin Interrupt	Chip Wakeup	Fast Capability
37	AON_GPIO_4	L	Y	PD	Y	Y	Y
38	AON_GPIO_5	L	Y	PD	Y	Y	Y
39	AON_GPIO_6	L	Y	PD	Y	Y	Y
40	AON_GPIO_7	L	Y	PD	Y	Y	Y
41	GPIO_24	L	Y	PD	Y	N	Y
42	GPIO_25	L	Y	PD	Y	N	Y
43	GPIO_16	L	Y	PD	Y	N	Y
44	GPIO_17	L	Y	PD	Y	N	Y
45	GPIO_31	L	Y	PD	Y	N	Y
46	GPIO_30	L	Y	PD	Y	N	Y
47	GPIO_26	L	Y	PD	Y	N	Y
48	GPIO_27	L	Y	PD	Y	N	Y
49	GPIO_28	L	Y	PD	Y	N	Y
50	GPIO_29	L	Y	PD	Y	N	Y

Table 2-6 Abbreviations related to pin properties

Properties	Abbreviation	Description
Default status after POR	Hi-Z	High impedance
	H	High level
	L	Low level
Pull-up/Pull-down enable after POR	Y	Enabled
	N	Disabled
Pull-up/Pull-down selection after POR	PU	Pull-up
	PD	Pull-down
Pin interrupt	N	No
	Y	Yes
Fast capability	N	Fast capability is not supported.
	Y	Fast capability is supported.

3 System

3.1 MCU Core

3.1.1 Functional Overview

The device integrates a 32-bit Arm® Cortex®-M4F core as the processor with a floating-point unit (FPU) which maximizes code density and offers highly efficient processing with lower energy. The architecture is shown in the figure below.

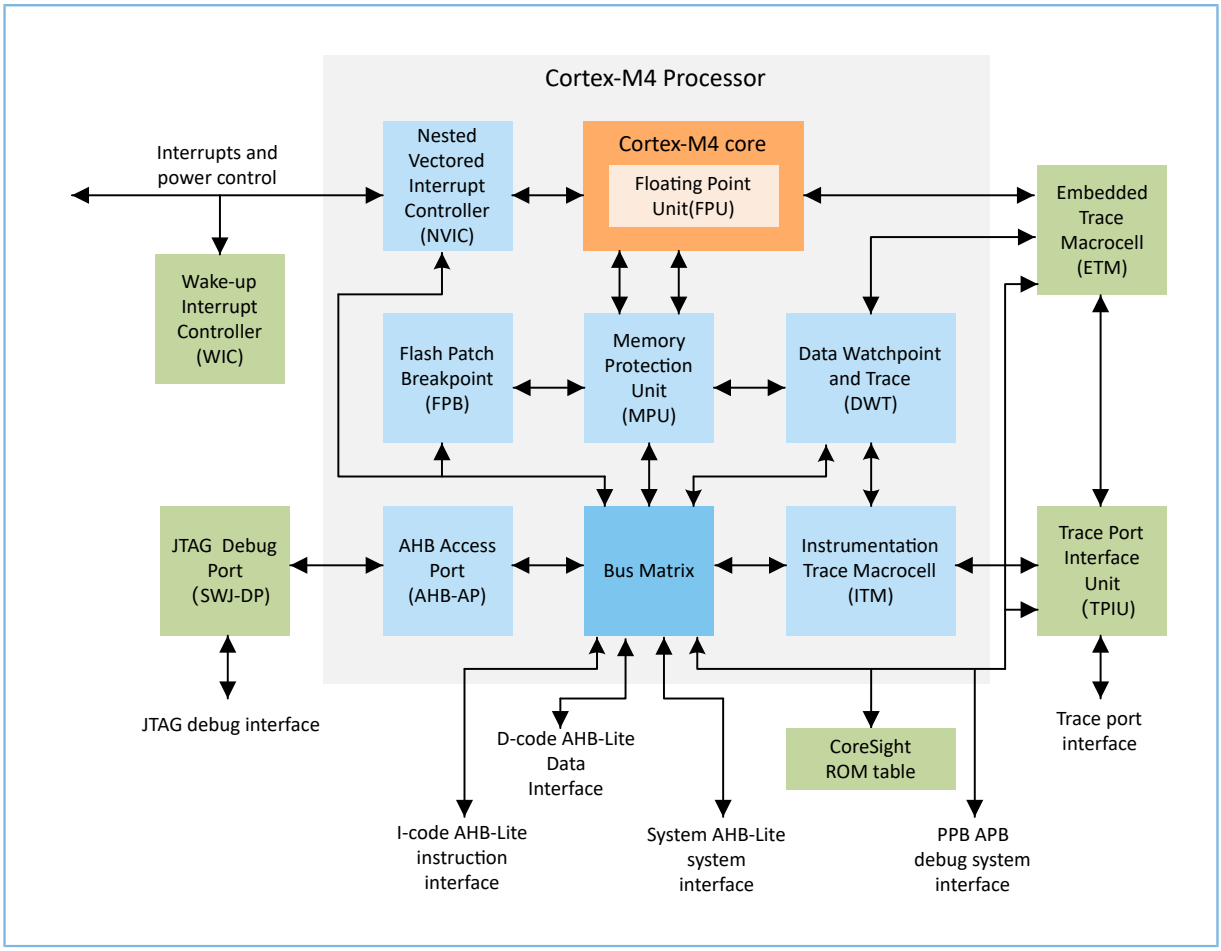

Figure 3-1 32-bit Arm ® Cortex®-M4F core

The Cortex®-M4F is built on a high-performance processor core with a 3-stage pipeline Harvard architecture, making it ideal for demanding embedded applications. The processor delivers exceptional power efficiency through an efficient instruction set and extensively optimized design, providing high-end processing hardware including optional IEEE754-compliant single-precision floating-point computation, a range of single-cycle and SIMD multiplication and multiply-with-accumulate capabilities, saturating arithmetic and dedicated hardware division.

The Cortex®-M4F processor implements a version of the Thumb® instruction set based on Thumb-2 technology, ensuring high code density and reduced program memory requirements. The Cortex®-M4F instruction set provides

exceptional performance expected of a modern 32-bit architecture, with the high code density of 8-bit and 16-bit microcontrollers (MCUs).

The Cortex®-M4F processor closely integrates a configurable Nested Vectored Interrupt Controller (NVIC), to deliver industry-leading interrupt performance. The NVIC includes a Non Maskable Interrupt (NMI) that can provide up to 256 interrupt priority levels. The tight integration of the processor core and NVIC provides fast execution of Interrupt Service Routines (ISRs), dramatically reducing the interrupt latency. This is achieved through the hardware stacking of registers, and the ability to suspend load-multiple and store-multiple operations. Interrupt handlers do not require wrapping in assembler code, or removing any code overhead from the ISRs. A tail-chain optimization also significantly reduces the overhead when switching from one ISR to another.

The Cortex®-M4F processor includes many debug features that allow developers to analyze design problems easily. In addition, its standard design features (such as halting and single stepping) that can be found in most microcontrollers allow developers to generate a trace to capture program flow, data changes, profiling information, and more.

3.1.2 Main Features

The Arm® Cortex®-M4F processor offers a number of CPU options and support modules implemented on the device.

Table 3-1 Support options/modules

No.	Option/Module	Description	Implemented
1	NVIC	Nested vector interrupt controller	66 vectors
2	PRIORITIES	Priority bits	8 bits
3	WIC	Wakeup interrupt controller	Yes
4	Endianness	Memory system endianness	Little endian
5	Bit-banding	Bit banded memory	Yes
6	DWT	Data watchpoint and trace	Yes
7	SysTick	System tick timer	Yes
8	FPU	Floating-point unit	Yes
9	MPU	Memory protection unit	Eight regions
10	DAP	Debug access port	Yes
11	ETM	Embedded trace macrocell	No
12	ITM	Instrumentation trace macrocell	Yes
13	TPIU	Trace port interface unit	Yes
14	ETB	Embedded trace buffer	No
15	FPB	Flash patch and breakpoint unit	16 addresses

3.1.3 Performance

Benefiting from the efficient design of each module, the device can easily reach a high level of operating efficiency. The CoreMark scores are shown in the table below.

Table 3-2 MCU CoreMark scores

Symbol	Description	Min.	Typ.	Max.	Unit
$CM_{FLASH,Direct}$	CoreMark, running from Flash, 96 MHz, direct map cache		316		CoreMark
$CM_{FLASH/MHz,Direct}$	CoreMark per MHz, running from Flash, 96 MHz, direct map cache		3.3		CoreMark/MHz
$CM_{FLASH,4\text{-}Way}$	CoreMark, running from Flash, 96 MHz, 4-way associative cache		261		CoreMark
$CM_{FLASH/MHz,\ 4\text{-}Way}$	CoreMark per MHz, running from Flash, 96 MHz, 4-way associative cache		2.7		CoreMark/MHz

3.1.4 Interrupt

In addition to supporting 16 system exceptions within the Arm® Cortex®-M4F, the device also supports 66 external interrupts with 256 programmable priority levels. In some cases, a single peripheral can generate multiple different interrupts (such as BOD/Bluetooth LE interrput). Most interrupts are connected directly to the NVIC, but all wakeup sources on the always-on domain are connected to the Power State Controller (PSC) outside of the Arm® Cortex®-M4F, allowing the interrupt sources to wake up the Arm® Cortex®-M4F core from the sleep mode. In most cases, developers do not need to change the interrupt source.

The Cortex®-M4F allows developers to assign various interrupts to different priority levels based on the application requirements. To maintain the stability of Bluetooth connection, it is recommended that the priority of Bluetooth interrupt should be set to the highest. In most cases, developers do not need to change priority levels, and the SoC has been configured with default priorities to meet most application requirements.

One additional feature of the Cortex®-M4F interrupt architecture is the ability to relocate the vector table to different addresses. The software can move the vector table into SRAM and reassign the interrupt service routine entry addresses on demand, but it should be noted that the remapping start address of the interrupt vector table should be aligned with **0x200**. For example, the vector table base address should be set as **0x00000000**, **0x00000200**, or **0x00000400**.

All non-programmable and programmable interrupts including those that can wake up the MCU from sleep mode are shown in Table 3-3 and Table 3-4. The width of interrupt priority control register is 8 bits, in which bit 7–bit 4 are used for preemption priority and bit 3–bit 0 are used for subpriority. A higher level corresponds to a lower priority, so level 0 is the highest interrupt priority.

Table 3-3 Non-programmable interrupts

No.	Name	Default Priority	Wake up MCU	Description
1	Reset_IRQ	-3		Reset exception
2	NMI_IRQn	-2		Non-maskable interrupt

No.	Name	Default Priority	Wake up MCU	Description
3	HardFault_IRQ	-1		All classes of fault. This interrupt occurs when the corresponding fault handler cannot be activated because it is disabled or masked by exception masking.

Table 3-4 Programmable interrupts

No.	Name	Default Preemption Priority	Default Subpriority	Wake up MCU	Description
4	MemManage_IRQ	-	-		Memory management fault; caused by MPU violation or invalid accesses
5	BusFault_IRQ	-	-		Error response received from the bus system
6	UsageFault_IRQ	-	-		Usage fault; typical causes are invalid instructions or invalid state transition attempts.
7–10	Reserved	-	-		Unused
11	SVCall_IRQ	0	0		Supervisor call via SVC instruction
12	DebugMonitor_IRQ	-	-		Debug monitor – for software-based debugging
13	Reserved	-	-		Unused
14	PendSV_IRQ	15	15		Pendable request for system service
15	SysTick_IRQ	15	15		System tick timer interrupt
16	WDT_IRQ	8	0		System watch dog timer interrupt
17	BLE_SDK_IRQ	15	15		Bluetooth LE software development kit (SDK) schedule interrupt
18	BLE_IRQ	2	0	√	Bluetooth LE interrupt
19	DMA0_IRQ	6	0		DMA0 interrupt
20	SPI_M_IRQ	8	0		SPI master interrupt
21	SPI_S_IRQ	8	0		SPI slave interrupt
22	EXT0_IRQ	8	0		GPIO0 interrupt
23	EXT1_IRQ	8	0		GPIO1 interrupt
24	TIMER0_IRQ	12	0		TIMER0 interrupt
25	TIMER1_IRQ	12	0		TIMER1 interrupt
26	DUAL_TIMER_IRQ	12	0		DUAL_TIMER interrupt
27	QSPI0_IRQ	8	0		QSPI0 interrupt
28	UART0_IRQ	10	0		UART0 interrupt

No.	Name	Default Preemption Priority	Default Subpriority	Wake up MCU	Description
29	UART1_IRQ	10	0		UART1 interrupt
30	I2C0_IRQ	8	0		I2C0 interrupt
31	I2C1_IRQ	8	0		I2C1 interrupt
32	AES_IRQ	8	0		AES interrupt
33	HMAC_IRQ	8	0		HMAC interrupt
34	EXT2_IRQ	8	0		GPIO2 interrupt
35	RNG_IRQ	8	0		TRNG interrupt
36	BOD_ASSERT_IRQ	8	0		Brown-out detect interrupt
37	PKC_IRQ	8	0		PKC interrupt
38	XQSPI_IRQ	8	0		XQSPI interrupt
39	QSPI1_IRQ	8	0		QSPI1 interrupt
40	RESERVED0_IRQ	8	0		Reserved interrupt
41	BLESLP_IRQ	2	0	√	Bluetooth LE timer done interrupt
42	SLPTIMER_IRQ	12	0	√	Sleep timer done interrupt
43	AON_EXT_IRQ	8	0	√	AON GPIO interrupt
44	AON_WDT_IRQ	8	0	√	AON watch dog timer interrupt
45	I2S_M_IRQ	8	0		I2S master interrupt
46	I2S_S_IRQ	8	0		I2S slave interrupt
47	RESERVED1_IRQ	8	0		Reserved interrupt
48	PRESENT_IRQ	8	0		Present done interrupt
49	RTC0_IRQ	8	0	√	Real-timer counter0 interrupt
50	COMM_CORE_IRQ	8	0		AHB/APB bus timeout interrupt
51	DMA1_IRQ	6	0		DMA1 interrupt
52	DMA2_IRQ	6	0		DMA2 interrupt
53	DSPI_IRQ	8	0		DSPI interrupt
54	AON_IRQ	8	0		AON domain interrupt
55	PDM_IRQ	8	0		PDM interrupt
56	VTTBL_IRQ	8	0		VT table interrupt
57	CTE_FULL_IRQ	8	0		CTE full interrupt
58	RESERVED2_IRQ	8	0		Reserved interrupt
59	RESERVED3_IRQ	8	0		Reserved interrupt
60	BOD_DEASSERT_IRQ	8	0	√	Brown out detect disappear interrupt

No.	Name	Default Preemption Priority	Default Subpriority	Wake up MCU	Description
61	COMP_IRQ	8	0	√	Comparator interrupt
62	RESERVED4_IRQ	8	0		Reserved interrupt
63	RESERVED5_IRQ	8	0		Reserved interrupt
64	RTC1_IRQ	8	0	√	Real-timer counter1 interrupt
65	CPLL_DRIFT_IRQ	8	0		CPLL drift interrupt
66	CLK_CALIB_IRQ	8	0		Clock calibration0/1 interrupt
67	RESERVED6_IRQ	8	0		Reserved interrupt
68	UART2_IRQ	10	0		UART2 interrupt
69	UART3_IRQ	10	0		UART3 interrupt
70	RESERVED7_IRQ	10	0		Reserved interrupt
71	RESERVED8_IRQ	10	0		Reserved interrupt
72	BLE_PWR_ON_IRQ	2	0		BLE sequencer power on done interrupt
73	BLE_PWR_DN_IRQ	8	0		BLE sequencer power off done interrupt
74	I2C2_IRQ	8	0		I2C2 interrupt
75	I2C3_IRQ	8	0		I2C3 interrupt
76	RESERVED9_IRQ	8	0		Reserved interrupt
77	RESERVED10_IRQ	8	0		Reserved interrupt
78	XO_BIAS_SW_CHG_IRQ	8	0		Crystal oscillator (XO) bias switch changed interrupt
79	DDVS_IRQ	8	0		DDVS control error interrupt
80	RESERVED11_IRQ	8	0		Reserved interrupt
81	QSPI2_IRQ	8	0		QSPI2 interrupt

3.1.5 Debug Interfaces

The device is built on a Cortex®-M4F with FPU core which contains hardware extensions for advanced debugging features. The debug extensions allow the core to be stopped either on a given instruction fetch (breakpoint) or a data access (watchpoint). When the core is stopped by connecting an external debugger to a serial wire debug (SWD) interface, the internal core state and the external system state can be examined. Once examination is completed, the core and the system can be restored, and program execution can be resumed.

It supports the following debug and trace features:

- Two-pin SWD interface
- Flash patch and breakpoint (FPB) unit support

- Two literal comparators
- Sixteen instruction comparators

- Data watchpoint trigger (DWT) unit
  - Instrumentation trace and macrocell (ITM)
  - Trace port interface unit (TPIU)
  - 4-bit parallel trace of ITM trace data
  - Serial wire output (SWO) trace of ITM data

3.2 Bus and Memory System

3.2.1 System Buses

The device contains a set of AMBA buses which move data between peripherals, memory, and the the Arm® Cortex®-M4F processor.

The Arm® Cortex®-M4F processor provides generic bus interfaces based on Advanced Microcontroller Bus Architecture (AMBA). The AMBA specification supports several bus protocols for communication between memories and peripherals. The main bus interface uses the AHB Lite protocol.

- I-Code bus interface: Primarily for program memory; fetches instructions from the 'Code' memory space (0x0 to 0x1FFFFFFF).
- D-Code bus interface: Primarily for program memory; supports data and debug accesses to the 'Code' memory space (0x0 to 0x1FFFFFFF).
- System bus interface: Primarily for RAM and peripherals; accesses the SRAM and other peripherals of the MCU.

Among them, I-CODE and D-CODE are independent from the system bus. They can realize parallel data access and value acquisition, which can effectively improve performance. The MCU maps the available SRAM onto an address space within the 'Code' memory space. This gives the application the opportunity to perform tasks efficiently because the system can quickly finish the task and go to sleep mode.

The infrequently accessed peripherals of the MCU are located on an AMBA advanced peripheral bus (APB). There is a bridge which translates accesses from the System AHB to the advanced high-performance bus (APB). Access to these peripherals will inject a single wait-state on the AHB during any access cycle.

In addition, it provides better performance in data access and higher flexibility. The AHB matrix ports support DMA and CPU access to different memories and peripherals at the same time. When multiple masters access the same memory or peripheral concurrently, the SoC supports setting the access priorities of the masters based on round-robin scheduling according to the access sequence, instead of using a fixed priority. After each response to the master request, the priority will be determined.

3.2.1.1 AHB Connection Matrix

AMBA supports a variety of bus protocols. For this device, the main bus interface uses the AHB lite protocol. The infrequently accessed peripherals of the MCU are located on an AMBA APB bus. There is a bridge which translates the accesses from System AHB to the APB. Access to these peripherals will inject a single wait-state on the AHB during any access cycle. The system bus diagram is shown in Figure 3-2.

At the same time, the AHB bus has matrix ports, which can support DMA and CPU to access different memories and peripherals at the same time, beneficial to improve the performance of data access. When multiple masters access the same memory or peripherals, The SoC uses a round-robin scheduling to configure the priorities of multiple masters.

After each response to the master's request, the priority will be determined according to the order of access. This method is more flexible than fixed priority.

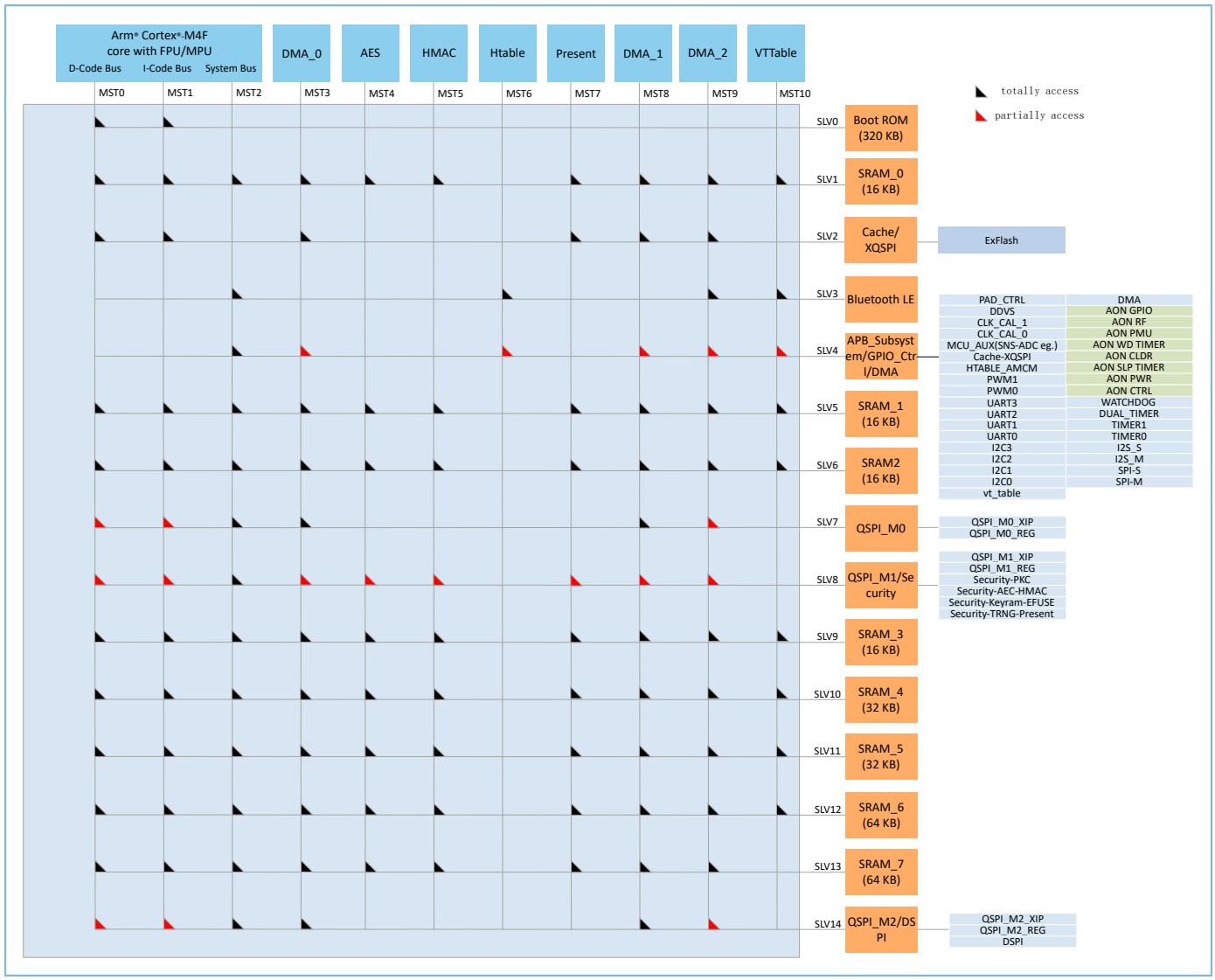

Figure 3-2 System bus

In total, 11 bus masters are implemented:

- MST0: D-Code bus of Cortex®-M4F CPU core
- MST1: I-Code bus of Cortex®-M4F CPU core
- MST2: System bus of Cortex®-M4F CPU core
- MST3: DMA_0 AHB master
- MST4: AES AHB master
- MST5: HMAC AHB master
- MST6: Hopping table master for AM mode. It fetches address and data from a memory bank for parameter register.
- MST7: Present AHB master
- MST8: DMA_1 AHB master
- MST9: DMA_2 AHB master
- MST10: VTTable AHB master

3.2.1.2 AHB Multilayer Arbitration

The round robin method is used when multiple masters access the same slave at the same time, during which each master connected to the multi-layer AHB is assigned a default natural priority based on its slave interface (SI) number. Specifically, the master with a lower SI number is given higher priority than a master with a higher SI number. The masters and their SI numbers are shown in the Table 3-5. When more than one master device initiates access to the same slave device simultaneously, the master device with the lowest SI number is granted access first, whereas other bus master devices with higher SI numbers need to wait until the higher-priority master device has completed its transaction.

Based on AHB protocol, AHB multilayer allows parallel access paths between multiple masters and slaves in a system. AHB multilayer provides a bus layer for each bus master, with each layer connected to one slave through a slave multiplexer, reducing latencies while increasing the bus bandwidth available to multi-master systems. The bus masters are assigned with priorities to help the SoC in granting access when two (or more) bus masters concurrently request access to the same slave device. In the process, the following rules apply:

- If two (or more) bus masters concurrently request access to the same slave device, the master with the highest priority is granted the access first.
- Bus masters with lower priorities need to wait until the higher-priority master has completed its transaction.
- If the higher-priority master pauses at any point during its transaction, the next lower priority master in queue is temporarily granted access to the slave device until the higher priority master resumes its activity.
- Bus masters that share the same priority are mutually exclusive, thus cannot be used concurrently.

Copyright © 2024 Shenzhen Goodix Technology Co., Ltd.

However, some peripherals do not have a safe stalling mechanism (unable to stop incoming data). Being a low priority master might cause a loss of data for such peripherals upon bus contention. To avoid AHB contention when multiple bus masters are used, apply one of the following guidelines:

- Avoid situations where more than one bus master accesses the same slave.
- If it cannot be avoided, make sure that the bus bandwidth is not exhausted before masters access the slave.

Each master connected to the AHB multilayer is assigned a default priority, indicated by the SI number, as provided in the table below.

Table 3-5 AHB master details

Bus Master Name	SI Number	In/Out
CPU D-Code	SI0	I/O
CPU I-Code	SI1	I/O
CPU System Bus	SI2	I/O
DMA_0	SI3	I/O
AES	SI4	I/O
HMAC	SI5	I/O
Htable	SI6	O
Present	SI7	I/O
DMA_1	SI8	I/O
DMA_2	SI9	I/O
VTTable	SI10	I/O

3.2.1.3 Bus Faults

The bus will receive a bus fault when:

- The core attempts to access an address that is illegal, which means the host access crosses address boundaries.
- The slave sends a transmission error message to the host with an ERROR response.

In addition, the bus fault interrupt can be enabled or disabled when any transfer error occurs on the bus.

3.2.2 Memory Map

Based on the storage architecture of the Cortex®-M4F processor, this SoC divides its memory into ROM, Flash, data memory, peripheral memory, and memory for debugging components. All the mapping spaces of the 4 GB SoC memory are shown in Figure 3-3.

- The contents of the ROM area have been preprogrammed and cannot be modified by developers. This area mainly stores the startup process and code related to Bluetooth LE protocol stack.
- The RAM Alias1 area is used to store special-purpose code segments that are generally used for Flash operations or to speed up code fetching. Due to the characteristics of the Cortex®-M4F bus architecture, there are

independent instruction bus (I-CODE) and data bus (D-CODE) in the address space of 0x0000 0000–0x1FFF FFFF. The two buses can be used to fetch instructions and data in parallel, and the address space of 0x20000000–0xDFFF FFFF has only one system bus (System) where fetching instructions and data can only be performed serially. Therefore, it is more efficient to put code segments and constants in the area of 0x0000 0000–0x1FFF FFFF.

- The ExFlash area is generally used to store user code. ExFlash area and ExFlash Alias area share same physical address. In unencrypted mode, the code data in the ExFlash area can also be read by the corresponding area of ExFlash Alias. In encrypted mode, firmware key is required to support reading the content of ExFlash, and data key is required to support reading the content of ExFlash Alias.
- The RAM area is generally used to store data and code copied from Flash. In the ARM® Cortex®-M4F architecture, the bit-band operations in the 0x2000 0000–0x200F FFFF and 0x4000 0000–0x400F FFFF are conducive to the atomic characteristics of data operations.
- QSPI0, QSPI1, and QSPI2 all support XIP mode. In this mode, the data space of Flash or PRSAM can be mapped to the memory, which allows for direct operations on the memory address. For external PSRAM solutions, the external PSRAM mounted on QSPI M1 XIP Alias1 can be combined with the memory of 0x2000 0000–0x2003 FFFF to form a continuous SRAM address space.

Copyright © 2024 Shenzhen Goodix Technology Co., Ltd.

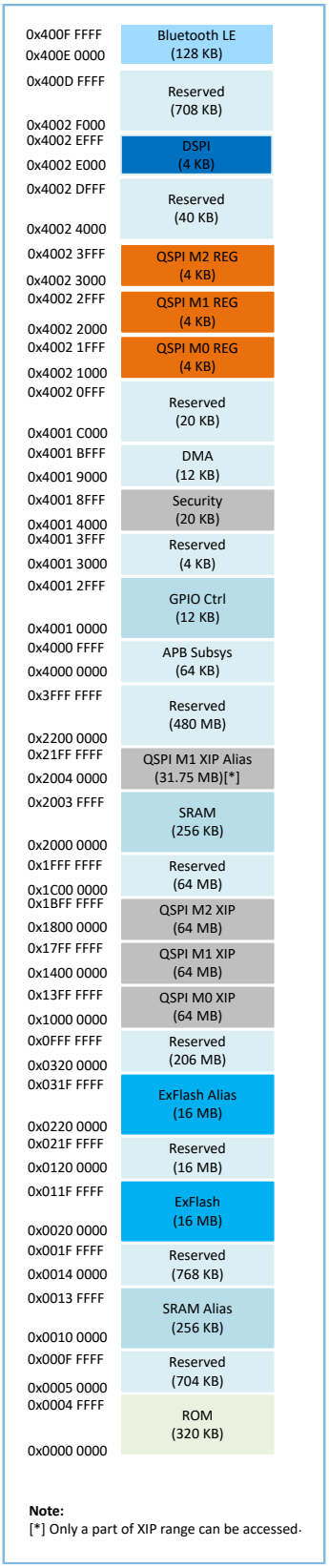

Figure 3-3 Memory map

Copyright © 2024 Shenzhen Goodix Technology Co., Ltd.

3.2.3 Memory Subsystem

The SoC integrates various kinds of memory, as described below.

- SRAM

  The main Static Random Access Memory (SRAM) is comprised of up to an on-chip static RAM with a capacity of up to 256 KB. It mainly stores application data, which can also be used to execute instructions. The DMA controller can transfer data between the SRAM, Flash and peripherals.

- Flash

  The non-volatile memory management system integrates an accessible Flash memory (512 KB Flash for GR5525IENI; 1 MB Flash for GR5525RGNI) and an on-chip XQSPI controller, supporting the XIP mechanism and allowing the SoC to store application data and code execution in Flash. The XIP mechanism allows the CPU to execute program code directly from the internal Flash by using QSPI. When a cache miss occurs, the XIP controller will automatically prefetch a certain number of program instructions from Flash and update them into the cache memory.

- ROM

  The 320 KB read-only memory (ROM) mainly stores the boot code and code related to Bluetooth LE protocol stack.

- eFuse

  The eFuse (512 bytes) stores factory information, such as the UID, Flash security register address, and memory information.

- Cache

  The 8 KB cache is a high-speed buffer memory between CPU and memory. The speed of CPU is much higher than that of memory. When CPU directly accesses data from memory, it has to wait for a certain period. However, cache can save the data that has just been used or repeatedly used by CPU. If CPU needs to use such data again, it can fetch the data directly from cache, which reduces delay by avoiding repeated access, and thus improves system efficiency.

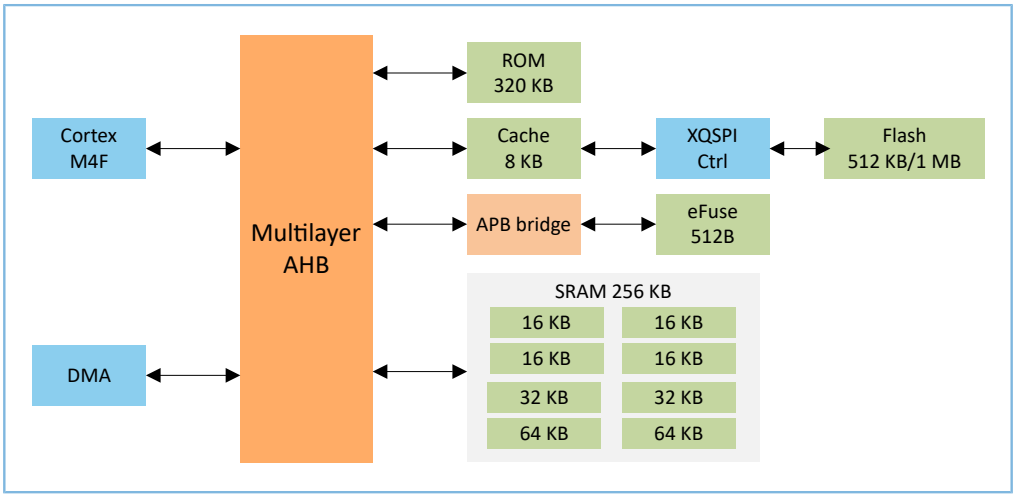

Figure 3-4 Memory subsystem

3.2.3.1 SRAM

3.2.3.1.1 Introduction

Although always contiguous on all devices, the SRAM instances are divided between eight AHB matrix ports. This allows user programs to potentially achieve better performance by dividing RAM usage among the ports. For example, simultaneous access to SRAM0 by the CPU and SRAM1 by the system DMA controller does not result in any bus stalls for either master. When multiple masters access the same SRAM, the device uses round-robin scheduling to configure the priorities of multiple hosts, ensuring the fairness of multi-master access.

Generally, the CPU will read or write all peripheral data at the same point, even when all such data is read from or sent to a peripheral by DMA. Therefore, minimizing stalls is likely to involve putting data to/from different peripherals in RAM on each port.

Alternatively, load sequences of data from the same peripheral can be alternated between RAM on each port. This can be helpful if DMA fills or empties a RAM buffer, and then signals the CPU before proceeding with a second buffer. The CPU will then tend to access the data while the DMA is using RAM on the other port.

At the same time, according to "Section 3.2.2 Memory Map", the device provides an Aliasing Memory with the start address of **0x0010_0000** for the RAM with the start address of **0x2000 0000**.

- The start address of the area is **0x2000 0000**, which supports bit-segment operations, and its corresponding bit-segment start address is **0x2200 0000**.
- Due to the characteristics of the Cortex® -M4F bus architecture, the access efficiency of the area with the start address of **0x0010 0000** is higher than that of other areas, so the running address of the RAM_CODE executable code is within the address range of the **0x0010_0000** area, which can speed up the operation. RW, ZI, HEAP, and STACK use RAM in the **0x2000 0000** area.

3.2.3.2 eFuse

3.2.3.2.1 Introduction

The eFuse (512 bytes) stores the product ID, TRIM, and security parameters. It consists of Main Region (240 bytes), Backup Region (240 bytes), and User Region (32 bytes). User Region is reserved for users to write and read by calling the eFuse driver. Every single byte of eFuse can only be written once. The following figure shows the structure of eFuse.

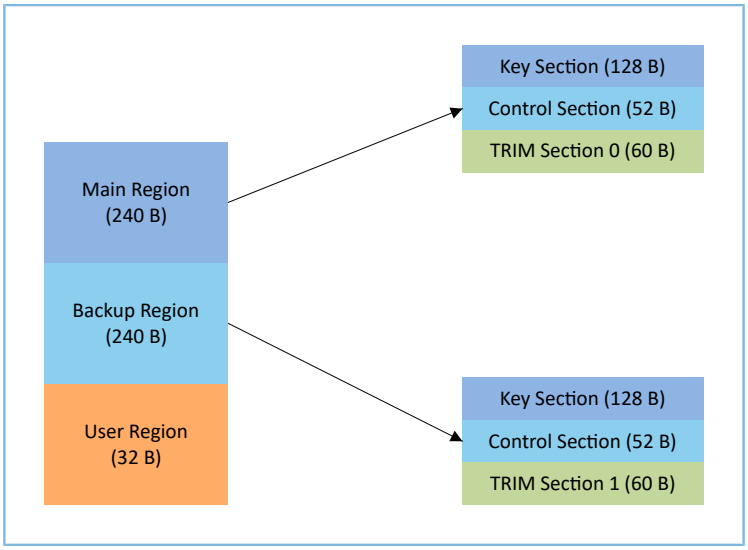

Figure 3-5 eFuse structure

Note:

The access to eFuse should be word-aligned, which means the offset address can only be 0x0, 0x4, …, 0x1C.

3.2.3.2.2 Main Features

- The initial eFuse bit is 0, and it can only be changed once from 0 to 1.
- Little-endian mode, the minimum unit of data reading or writing is 4 bytes.
- The Main Region has three sections: Key Section, Control Section and TRIM Section 0
  - Key Section stores all the keys needed for Secure Boot and SPI Flash encryption.
  - Control Section stores some parameters required by the boot loader for chip configuration.
  - TRIM Section 0 is used to save chip UID, TRIM parameters, and product configurations.
  - Users can read the chip UID from the address 0x40016110.
- The Backup Region has three sections: Key Section, Control Section and TRIM Section 1.
  - The values in Control Section and Key Section of the Backup Region are the same as those in the Main Region.
  - TRIM Section 1 is used to save Flash parameters as well as ADC and COMP calibration parameters.
- The User Region is reserved for users to write and read data by calling the eFuse driver.
  - The base address of User Region is 0x40016000.
  - Every single byte of eFuse can only be written once.
- Users can use GProgrammer to generate and download eFuse files.

3.2.3.2.3 Functional Description

3.2.3.2.3.1 Layout

The eFuse (512 bytes) consists of Main Region (240 bytes), Backup Region (240 bytes), and User Region (32 bytes). The layout of the three regions are as follows:

Table 3-6 eFuse Region Layout

Region	Section	Offset	Size (Byte)
User Region	User Region	0x000	32
Backup Region	TRIM Section 1	0x020	60
	Control Section	0x05C	52
	Key Section	0x090	128
Main Region	TRIM Section 0	0x110	60
	Control Section	0x14C	52
	Key Section	0x180	128

The information stored in Backup Region is shown in the following table.

Table 3-7 User Region Layout

Moudle	Symbol	Offset	Size (Byte)	Description
ADC	Trim	0x020	20	Trim value for ADC
FLASH	Trim	0x034	8	Trim value for FLASH
COMP	Trim	0x03C	4	Trim value for COMP
RSVD	Reserved	0x040	28	Reserved
Control Section	Configurations	0x05C	4	0bit: isp_uart_bypass (1: Not Support UART ISP Process, 0: Support UART ISP Process) 1bit: isp_usb_bypass (1;Not Support USB ISP Process, 0: Support USB ISP Process) 2bit: isp_jlink_bypass.(1: Not Support JLINK ISP Process, 0: Support JLINK ISP Process) 3 - 5bit: enc_boot_system_clk (0: XO-16M, 1: PLL-64M, 2: PLL-96M, 3: PLL-48M, 4: PLL-24M, 5: PLL-16M, 6: PLL-32M) 6 - 8bit: enc_boot_flash_clk (0: 16MHz, 1: 48MHz, 2: 32MHz, 3: 24MHz, 4:64MHz) 9 - 16bit: enc_boot_xip_read_cmd (0x03, 0x0B, 0x3B, 0xBB, 0x6B, 0xEB) 17-18bit: memory_power_size(0: 256KB, 1: 192KB, 2: 128KB)

Moudle	Symbol	Offset	Size (Byte)	Description
				19 - 31bit: Reserved
	SWDDisable	0x060	2	SWD enable (0: not disable, 1: disable)
	EncEnable	0x062	2	Encryption mode#0: unencrypted, 1: encrypted#
	CRC32	0x064	4	Key CRC32
	Chip ID	0x068	6	Chip ID
	Product ID	0x06E	2	Product ID
	FW public key HASH	0x070	16	Used in security application
	ROOT public key HASH	0x080	16	Used in security application
Key Section	ECC Key	0x090	32	Used in security application
	FW Key	0x0B0	32	Used in security application
	HMAC Key	0x0D0	32	Used in security application
	Data Key	0x0F0	32	Used in security application

The information stored in Main Region is shown in the following table.

Table 3-8 User Region Layout

Moudle	Symbol	Offset	Size (Byte)	Description
Version	Chip UID	0x110	16	0 - 2 Byte, 3Byte: FAB, eg.TSM 3 Byte, 1Byte:Year, the last two digits of year, eg.21 4 Byte, 1Byte:Month 5 - 12 Byte, 8Byte:LOT_ID, eg.P62G8501 13 Byte, 1Byte:Wafer_ID 14 Byte, 1Byte:Wafer_X 15 Byte, 1Byte:Wafer_Y
	ATE_PROG_VER	0x120	1	ATE program version
	Reserved	0x121	2	Reserved
USER	IO_LDO_SEL	0x123	1	0bit: IO_LDO selector (0: 1.8v, 1:3.3v) 1bit: IO_LDO_Bypass (0: no bypass, 1: bypass)
	BT_ADDR	0x124	6	Bluetooth device address
	XO_OFFSET	0x12A	2	XO calibration value
Verify	Pattern	0x12C	2	Trim pattern, fixed 0x4744

Moudle	Symbol	Offset	Size (Byte)	Description
	Trim_Sum	0x12E	2	Trim_sum include all Trim1 Section and a part of Trim 0 Section
Production	HW Version	0x130	2	Build code(hardware version). Encoded as ASCII.
	Chip ID	0x132	2	Part code. GR5525: 0x5525 unspecified : 0x0000
	Package	0x134	2	0x1000: BGA55 0x2000: QFN56 0x2100: QFN68 unspecified : 0x0000
	Flash Size	0x136	2	1024KB: 0x0400 512KB: 0x0200
	RAM Size	0x138	2	512KB: 0x0200, 256KB: 0x0100, 192KB: 0x00C0, 128KB: 0x0080
	Reserved	0x13A	2	Reserved
POWER	DCDC_1P05	0x13C	1	DCDC output 1.05V
	DCDC_1P15	0x13D	1	DCDC output 1.15V
	DIG_LDO_0P9	0x13E	1	Core LDO output 0.9V
	DIG_LDO_1P05	0x13F	1	Core LDO output 1.05V
	IO_LDO_1P8	0x140	1	IO LDO output 1.8V
	IO_LDO_3P0	0x252	1	IO LDO output 3.0V
	STB_IO_LDO_1P8	0x142	1	STB_IO_LDO 1.8V. STB IO LDO is used for sleep
	STB_IO_LDO_3P0	0x143	1	STB_IO_LDO 3.0V
	Reserved	0x144	4	Reserved
RF	TX Power	0x148	1	signed magnitude fixed point in 5.2
	RSSI CALI	0x149	1	signed int8_t variable, 2's complement format
	HP Gain_2M_offse	0x14A	1	signed int8_t variable, 2's complement format
	Reserved	0x14B	1	
Ctrl Section	Configurations		4	0bit: isp_uart_bypass (1: Not Support UART ISP Process, 0: Support UART ISP Process)

Moudle	Symbol	Offset	Size (Byte)	Description
				1bit: isp_usb_bypass (1;Not Support USB ISP Process, 0: Support USB ISP Process) 2bit: isp_jlink_bypass.(1: Not Support JLINK ISP Process, 0: Support JLINK ISP Process) 3~5bit: enc_boot_system_clk (0: XO-16M, 1: PLL-64M, 2: PLL-96M, 3: PLL-48M, 4: PLL-24M, 5: PLL-16M, 6: PLL-32M) 6 - 8bit: enc_boot_flash_clk (0: 16MHz, 1: 48MHz, 2: 32MHz, 3: 24MHz, 4:64MHz) 9 - 16bit: enc_boot_xip_read_cmd (0x03, 0x0B, 0x3B, 0xBB, 0x6B, 0xEB) 17-18bit: memory_power_size(0: 256KB, 1: 192KB, 2: 128KB) 19 - 31bit: Reserved
	SWDDisable		2	SWD enable (0: not disable, 1: disable)
	EncEnable		2	Encryption mode#0: unencrypted, 1: encrypted#
	CRC32		4	Key CRC32
	Chip ID		6	Chip ID
	Product ID		2	Product ID
	FW public key HASH		16	Used in security application
	ROOT public key HASH		16	Used in security application
Key Section	ECC Key		32	Used in security application
	FW Key		32	Used in security application
	HMAC Key		32	Used in security application
	Data Key		32	Used in security application

Note:

The data in the Key Section is encrypted.

3.2.3.2.3.2 Write/Read Instructions

- **Write flow:**
  1. Write 1 to RF_REG_2.[10]. Connect to eFuse supply voltage.
  2. Write 1 to RF_REG_2.[10-12]. Connect to eFuse program voltage.
  3. Configure the TPGM register.
  4. Write 0 to the PGENB.PGMEN_CTRL register.
  5. Software sends commands to write data to eFuse.
  6. If software has no data to write, write 1 to the PGENB.PGMEN_CTRL register.
  7. Write 0 to RF_REG_2.[10-12].
- **Read flow:**
  1. Write 1 to RF_REG_2.[10]. Connect to eFuse supply voltage.
  2. Write 0 to RF_REG_2.[11-12]. Disconnect from eFuse program voltage.
  3. Write 1 to the PGENB.PGMEN_CTRL register.
  4. Software sends commands to read data from eFuse.

3.2.3.2.3.3 Initial value check

The eFuse controller supports initial value check. When the efuse is all 0, the check returns success, otherwise it returns failure.

- Configure OPERATION.INIT_CHECK in order to trigger initial value check.
- Check STAT.INIT_DONE to get whether it is complete.
- Check STAT.INIT_SUCCESS to get result.

3.2.3.2.3.4 CRC calculation

The eFuse controller supports CRC calculation in order to ensure the integrity of the data stored in eFuse.

- Configure the start address and length in CRC_ADDR and TPGM.CRC_CHECK_LEN.
- Configure OPERATION.CRC_CHECK in order to trigger CRC calculation.
- Check STAT.CRC_CHECK_DONE to get whether it is complete.
- Get CRC result in CRC_OUTPUT.

3.2.3.2.3.5 Write Keys to KEYRAM

The data in the Key Section can only be written to the KEYRAM by hardware. The crypto engines can fetch these keys through KEYRAM.

- Configure OPERATION.WRITE_KEYRAM in order to trigger hardware to write keys from eFuse to KEYRAM.
- Check STAT.WRITE_KEYRAM_BUSY to get whether it is complete.

Copyright © 2024 Shenzhen Goodix Technology Co., Ltd.

3.2.3.3 Flash (Non-Volatile Memory)

3.2.3.3.1 Introduction

The non-volatile memory management system integrates an accessible Flash memory, and an on-chip QSPI controller that supports the XIP mechanism, allowing the device to be capable of storing application data and executing code in Flash address.

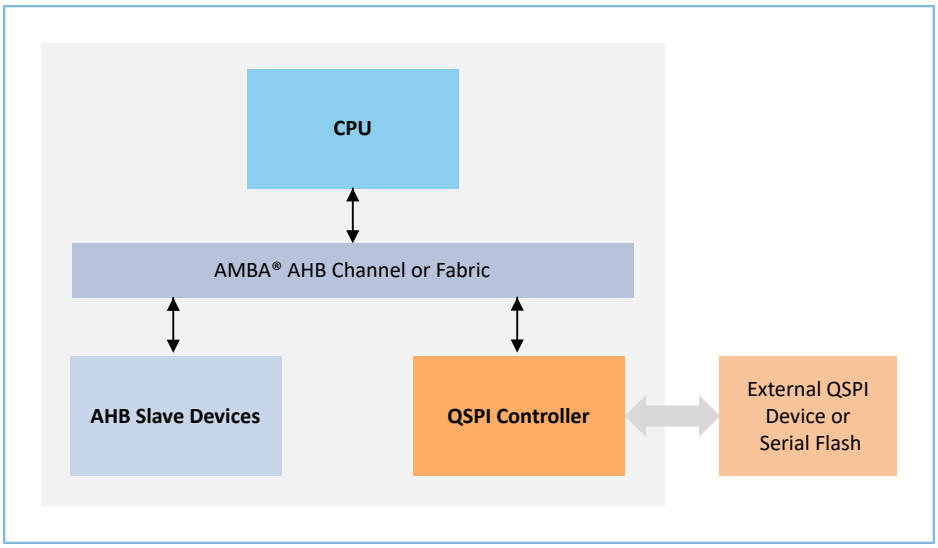

Figure 3-6 Flash structure

The XIP mechanism allows CPU to execute program code directly from the internal Flash memory by using the QSPI interface. When a cache miss occurs, the XIP controller will automatically prefetch a certain number of program instructions from Flash memory and update them into the cache memory.

3.2.3.3.2 Main Features

- Single 1.65 V to 3.60 V supply
- Industrial temperature range: -40°C to 85°C
- Serial peripheral interface (SPI) compatible: Mode 0 and Mode 3
- Single, dual, and quad I/O modes
- Flexible architecture for code and data storage
  - Uniform 256-byte page program
  - Uniform 256-byte page erase
  - Uniform 4-KB sector erase
  - Uniform 32-KB/64-KB block erase
  - Full chip erase
- Hardware controlled locking of protected sectors by the write protect (WP) pin

- Fast program and erase speed
  - 2 ms page program time
  - 16 ms page erase time
  - 16 ms 4-KB sector erase time
  - 16 ms 32-KB block erase time
  - 16 ms 64-KB block erase time
- Ultra low power consumption
  - 0.1 µA deep power-down current
  - 10 µA standby current
  - 2.5 mA active read current at 33 MHz
  - 3.0 mA active program or erase current
- High reliability
  - 100,000 program/erase cycles
  - 20-year data retention

3.2.3.3.3 Functional Description

3.2.3.3.3.1 Write and Erase

Flash memory can be read by CPU for a unlimited number of times through the XIP controller, but can only be written or erased for a limited number of times (up to 100000 cycles). The write approach is restricted.

Writing to Flash memory is managed by the on-chip XQSPI controller.

- Before performing a write/erase, the XIP controller should be disabled for writing and the on-chip XQSPI controller should be enabled.
- After writing/erasing, software needs to restore the XIP controller. Users must make sure that writing and erasing are not performed at the same time. If the CPU executes code from Flash while the on-chip XQSPI controller is writing to Flash, the CPU execution will be halted. In this case, use SRAM instead.

The Flash memory is only able to write “0” to erased bits, but it cannot write a bit back to 1. Therefore, a page action must be performed before a write to the same address in Flash.

After erasing a Flash page, all bits in the page are set to 1. The CPU will stop working if the CPU executes code from the Flash while the XQSPI controller performs erase operations.

3.2.3.3.3.2 XQSPI Controller

The XQSPI-AHB controller can be configured as a master or a slave by software. Reading from/writing to the core is done on the AMBA AHB interface. The core operates in various data modes from 4 bits to 32 bits (8 modes are

supported in multiples of 4 data bits). The data is then serialized and transmitted, either LSB or MSB first, using the standard 4-wire SPI bus interface or the extended Quad mode bus.

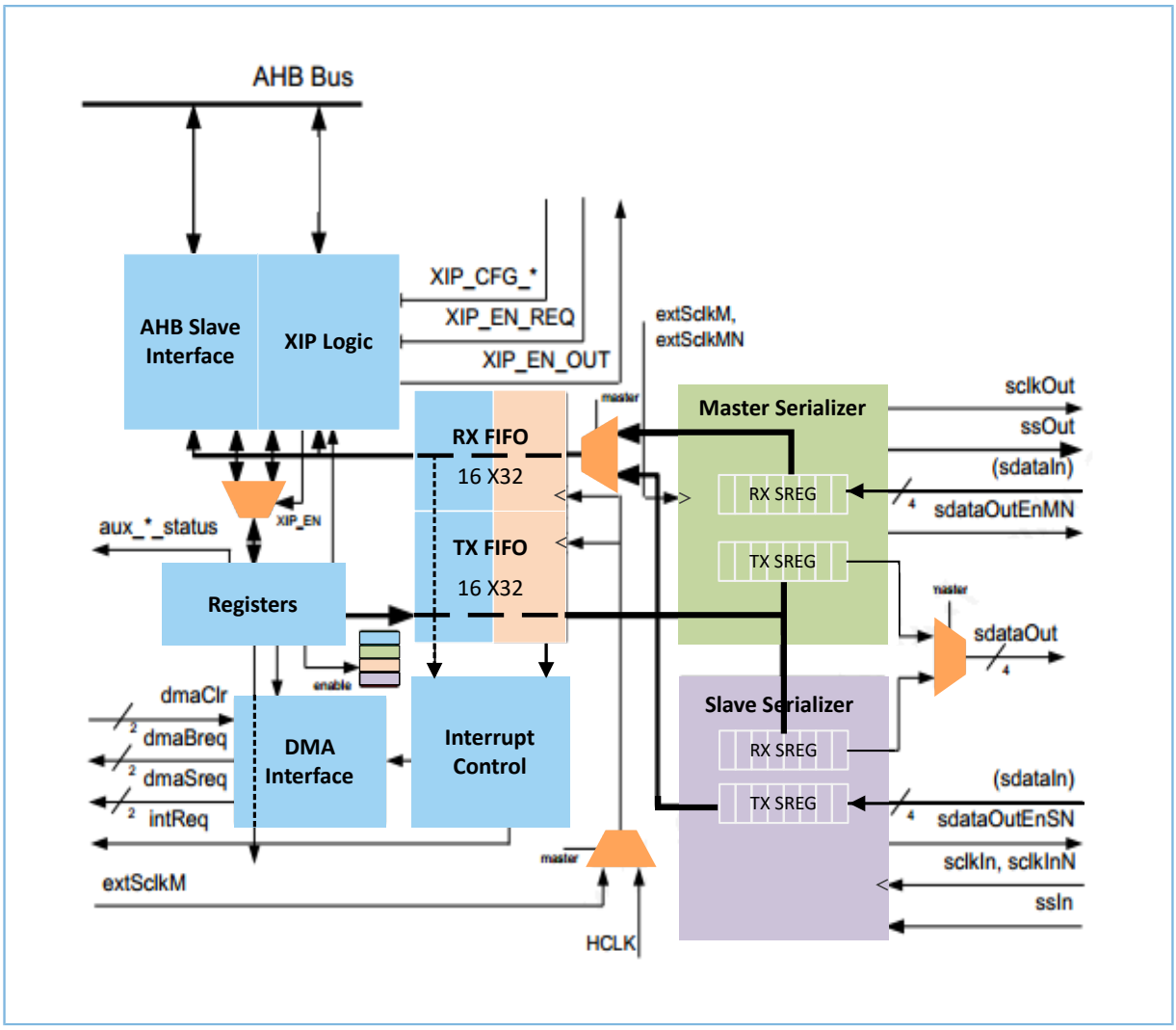

Figure 3-7 XQSPI controller clock domains

In the diagram, there are several clock domains represented: AHB clock (blue), External Master SCLK (green), and Slave SCLK (purple). The FIFOs serve as the clock boundary between the AHB clock and the SPI bus clock.

Production of the SPI bus clock varies from the Master/Slave mode:

- In Master mode, the QSPI core will produce the SPI bus clock directly from the External Master SCLK; in Slave mode the QSPI core will receive the SPI bus clock from another SPI Master.
- In Master mode, the Master Serializer and the right side of the FIFOs (purple) use the same clock; in Slave mode, the Slave Serializer (red) is clocked directly by the gapped SPI bus clock sclkIn/sclkInN, and the right side of the FIFOs (purple) is clocked by HCLK. Either ssIn or an internal signal that compares the receive data count against a terminal value is synchronized to the HCLK domain. FIFO pointers are advanced on the rising edge of this active-high, synchronized signal.

The master bit is responsible for directing much of the internal multiplexing. One other signal is also very important: enable. The enable bit is synchronized to the different clock domains (when appropriate). These synchronized enable signals, along with proper programming order of the AHB registers, allow logic on other domains to safely sample other AHB register values and control bits, since they are guaranteed to be stable once enable goes high. In short, the enable bit serves as a master synchronizer for the entire module.

SPI Data Link

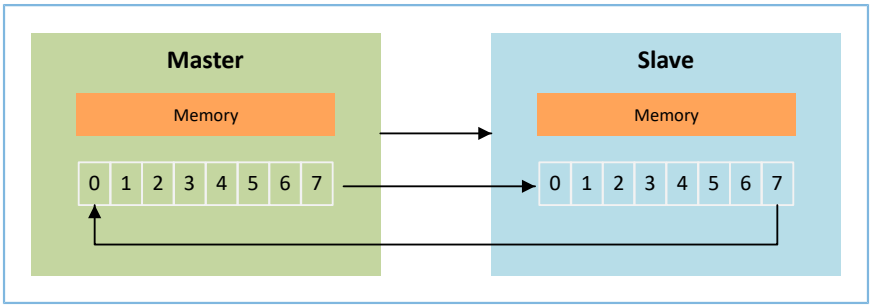

Figure 3-8 Master–Slave data transmission

Data is transmitted synchronously with MOSI (Master Out, Slave In) relative to the SCLK generated by the master device. The master also receives data on the MISO (Master In, Slave Out) signal in a full duplex fashion.

DMA Operation

The XQSPI-AHB module is compatible with various industry-standard DMA controllers. DMA operation in the IPC-QSPIAHB can be enabled to assist a DMA controller in the loading (writing) of the transmit FIFO, and the unloading (reading) of the receive FIFO. Note that dmaSreq indicates whether the receive FIFO is not empty, while dmaBreq indicates whether the receive FIFO level is above the programmed watermark level. It is the responsibility of the DMA Controller to implement the appropriate read transfer that does not underflow the RX FIFO.

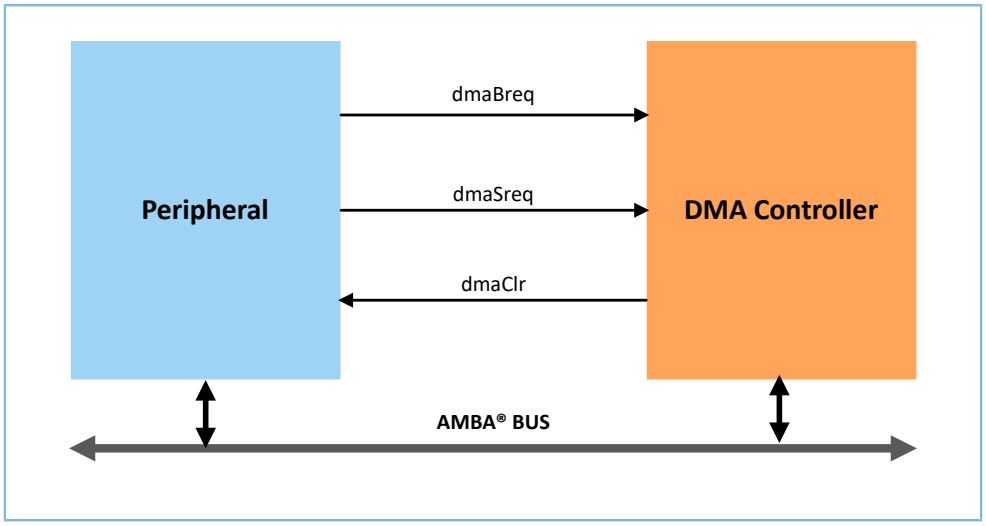

Figure 3-9 DMA operation

Execute-In-Place (XIP)

The Flash supports the Execute-in-place (XIP) functionality for Flash of several industry standards. The XIP mode allows an AHB Master to directly read contents from a device of one of the industry standards simply by reading from the

address space of the QSPI controller. This is useful in many situations. For example, when a processor executes boot instructions from a QSPI Flash device, or a DMA copy function where a DMA controller reads the contents of the QSPI Flash device then writes the data to a RAM elsewhere in the system.

Execute-In-Place Decryption (XIP-DEC)

The Execute-In-Place Decryption (XIP-DEC) is integrated into the original QSPI controller. With the XIP-DEC module, the code/data can be read on the fly decryption in the XIP mode.

3.2.3.3.3.3 Flash Operation Interfaces

All the Flash operation interfaces have been provided by the SDK package to ensure the safety of Flash operations. Users do not need to care about the implementation details of Flash operations.

3.2.3.3.4 Electrical Specifications

Table 3-9 Electrical specifications

Description	Min.	Typ.	Max.	Unit
Erase suspend latency			30	µs
Program suspend latency			30	µs
Latency between program resume and next suspend	0.3			µs
Latency between erase resume and next suspend	0.3			µs
Page program time (up to 256 bytes)		2	3	ms
Page erase time		8	20	ms
Sector erase time		8	20	ms
Chip erase time		8	20	ms
Program / Erase cycles		100000		Cycles

3.2.3.4 Cache

3.2.3.4.1 Introduction

A cache is a block of high-speed memory locations containing address information (known as a tag), and the associated data. The purpose of cache controller is to enhance system performance by speeding up CPU execution from Flash and reducing power consumption by minimizing Flash access. The Data RAM inside the cache controller stores instructions and data in blocks of cache-line size.

3.2.3.4.2 Main Features

- Direct mapping mode
- 4-way set associative mapping mode with LFU replacement policy

- 8 KB cache memory (Data RAM) and 16 bytes per cache line
- Auto clock gating for power saving
- Profiling counters

3.2.3.4.3 Block Diagram

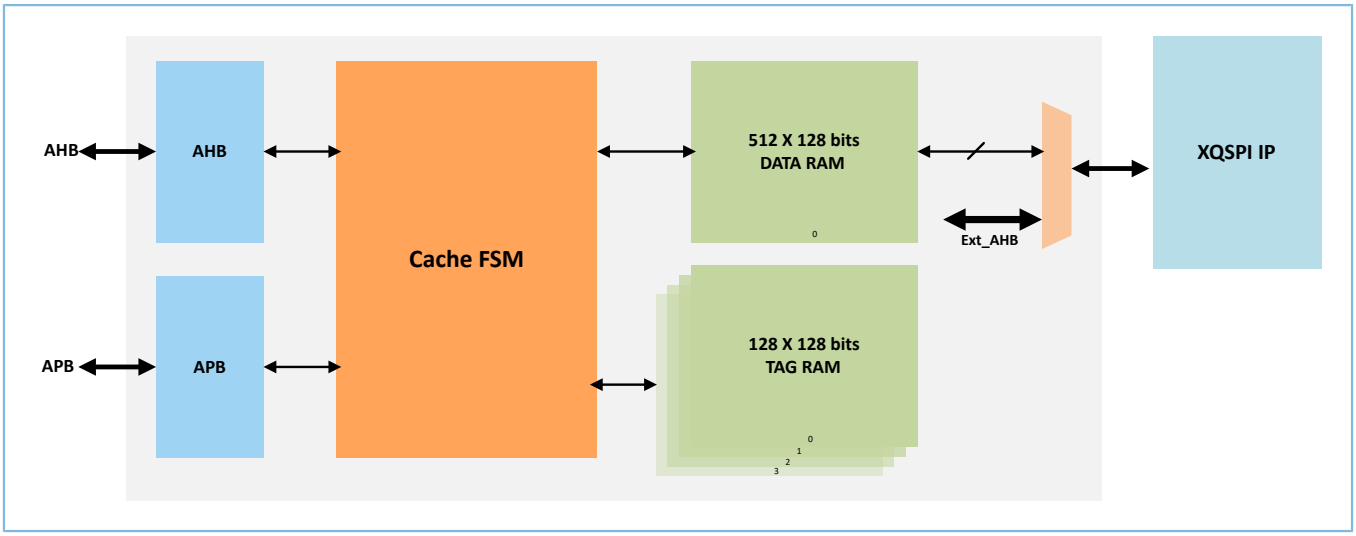

Figure 3-10 Cache block digram

3.2.3.4.4 Functional Description

3.2.3.4.4.1 Cacheable Range

The cache controller caches address range 0x20_0000 – 0x11F_FFFF (16 MB). There are two ways for CPU to read data from Flash:

- Cache mode: Cache controller sends request to XQSPI (green line way).
- Bypass mode: CPU directly sends request through AHB bus to XQSPI (blue line way).

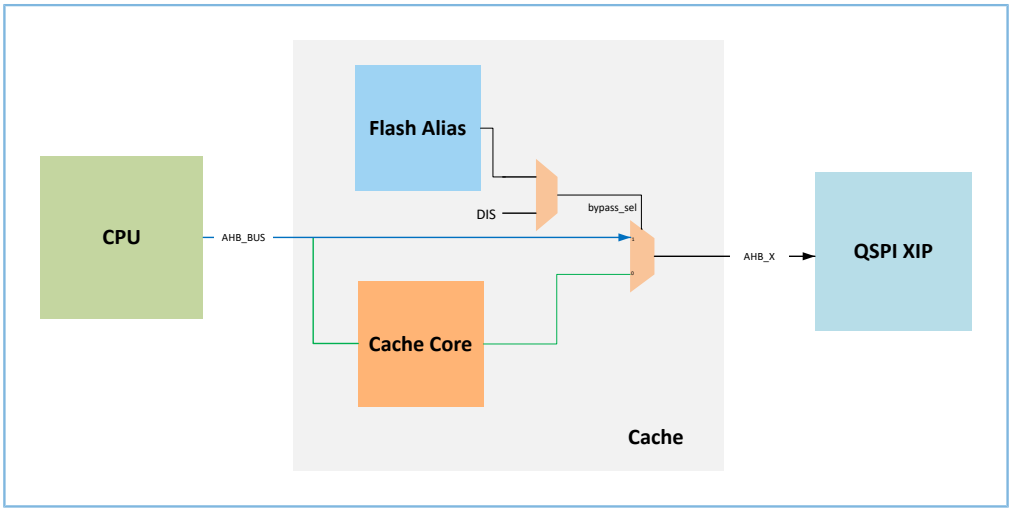

Figure 3-11 Two approaches for CPU reading data

There are also two ways to switch between two modes.

- Configure the DIS bit in the CTRL0 register of cache to control mode selection. When DIS = 1, the bypass mode is used; When DIS = 0, the cache mode is used.
- Automatic hardware switching according to address. When CPU reads Flash at Flash address, hardware automatically switches to cache mode. When CPU reads Flash at Flash alias address, hardware automatically switches to bypass mode. The following figure shows the allocation of Flash address and Flash alias address.

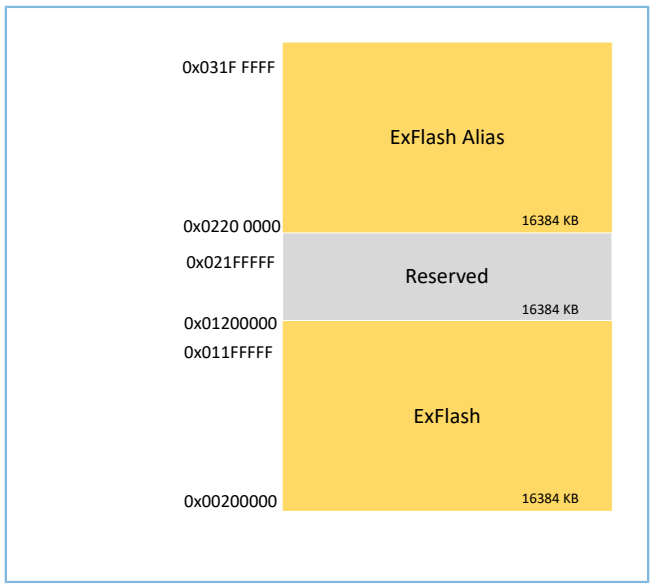

Figure 3-12 Flash/Flash alias address allocation

3.2.3.4.4.2 Address Segmentation

Flash main address is 0x20_0000–0x11F_FFFF. It is 16 MB.The address bit width is 24 bits. Cache size is 8 KB. Each cache way size is 2 KB. Cache line size is 128 bits. Address segmentation is shown below.

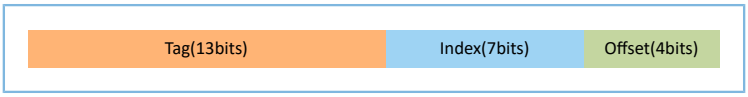

Figure 3-13 Address segmentation

3.2.3.4.4.3 Mapping Mode

To enable the associativity reconfiguration, the cache controller supports 4-way set associative cache and direct-mapped cache. Each cache way size is 2 KB in 4-way cache.

Configure the DIRECT_MAP_EN bit in the CTRL0 register of cache to select the cache mapping mode. When DIRECT_MAP_EN = 1, the direct-mapped cache is used; When DIRECT_MAP_EN = 0, the 4-way set associative cache mode is used.

3.2.3.4.4.4 Replacement Strategy

The cache controller uses the Least Frequency Used (LFU) algorithm in 4-way set associative cache mode. Each cache way in the controller has a hit_cnt that calculates the number of cache way hits. When a replacement is required, the way with the lowest hit_cnt will be replaced. In a case where the hit_cnt values of two or more ways equal, the substitution follows the priority: way3 > way2 > way1 > way0.

In direct-mapped cache mode, the controller directly replaces the entire cache when a cache miss occurs.

3.2.3.4.4.5 Miss Rate Monitor

The cache includes profiling counters. To measure the performance (hit-rate) of a code-section, the built-in profiling counters should be enabled. Start the profiling counters by resetting HITMISS bit in the CTRL0 register of cache. At the end of the section, the number of cache hits and cache misses for the section can be read from the HIT_COUNT and MISS_COUNT registers, and users can stop the counters by setting the HITMISS bit.

The cache hit-rate can be calculated based on the formula: HIT_COUNT/(HIT_COUNT + MISS_COUNT). The HIT_COUNT and MISS_COUNT registers do not wrap around after reaching the maximum value. If the maximum value is reached, consider profiling for a shorter duration to get correct numbers.

3.3 PMU

3.3.1 Functional Overview

The Power Management Unit (PMU), consisting of a DC-DC converter and a number of LDOs, is responsible for generating all required voltages for different blocks in the SoC.

In active mode, the DC-DC converter can generate voltage VDDRF/VDDVCO to power the RF domain. The CORE_LDO regulator generates the voltage DIGCORE for the digital logic blocks and can also power the memory and eFuse.

An AON_LDO regulator is used to supply its AON modules that stay ON when both the MCU subsystem and the Bluetooth LE subsystem are OFF. The RET_LDO regulator generates a lower voltage VRET for content retention to the memories where their content is needed after wakeup. Both ANA_IO_LDO and STB_IO_LDO supply power to the I/O domain, and also generate voltage VDD_EFLASH to supply power to Flash. ANA_IO_LDO provides I/O voltage to supply the I/O domain in active mode, and STB_IO_LDO is still working in sleep mode.

Copyright © 2024 Shenzhen Goodix Technology Co., Ltd.

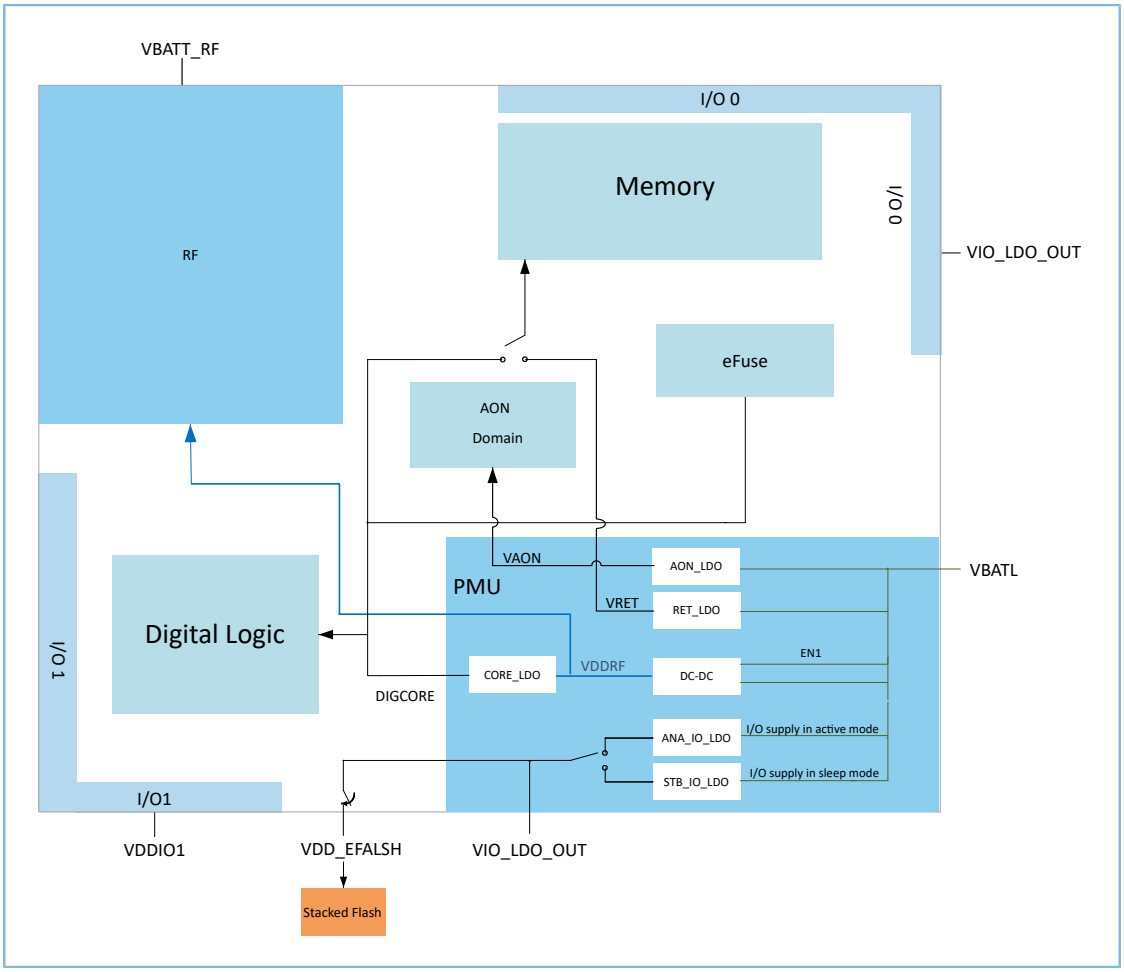

Figure 3-14 Power management block diagram

As shown in Figure 3-14, the power supply architecture is based on the PMU that generates the following supplies from an external supply VBATL.

- DC-DC: supply for RF domain and Core LDO (enabled by default)
- CORE_LDO: supply for digital logic blocks (enabled by default)
- AON_LDO: supply for AON blocks (enabled by default)
- RET_LDO: retention supply for memory instances (enabled by default)
- ANA_IO_LDO: Used to generate the I/O voltage to supply the pads of the SoC and external devices connecting to it. ANA_IO_LDO is also used to supply power to the stacked Flash (enabled by default).
- STB_IO_LDO: Used to replace ANA_IO_LDO during sleep; it can help reduce sleep current.
- VDDIO1: supply for I/O1 group; from external power voltage or VIO_LDO_OUT

Both VBATL and VBATT_RF come from the battery voltage VBAT. The VBATT_RF voltage is supplied to the RF domain only, and the VBATL voltage is the input voltage of all blocks.

As shown in the figure below, the startup sequence of each power module and clock in the SoC follows certain rules.

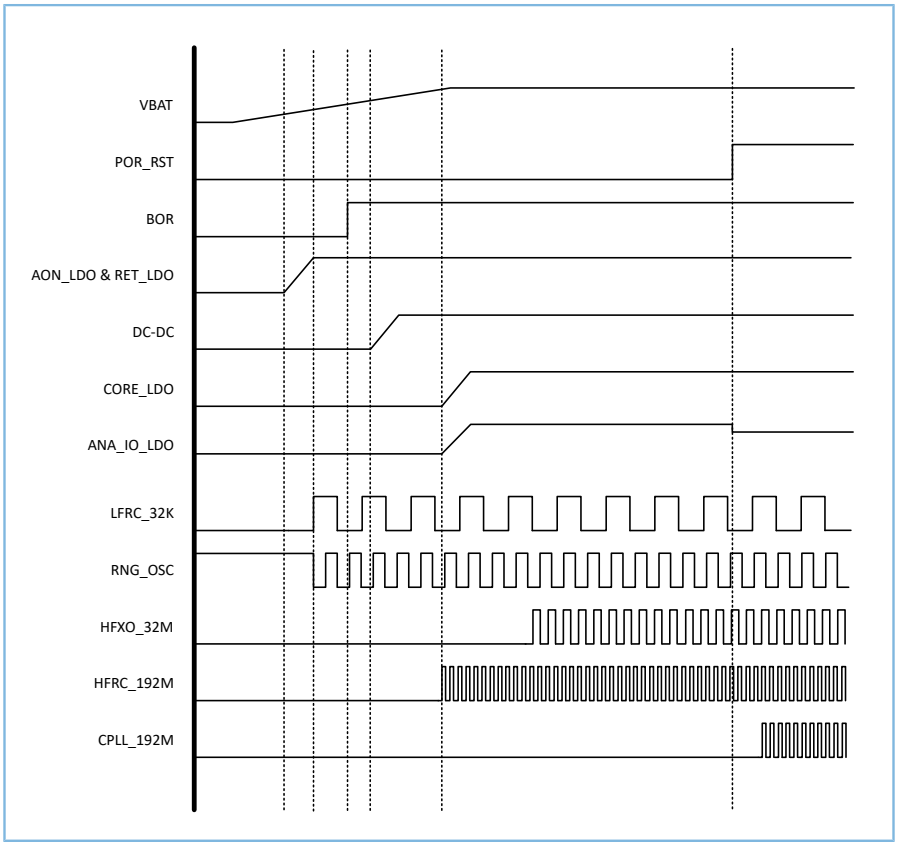

Figure 3-15 Power-up timing diagram

1. As the voltage of the VBAT gradually increases, when the corresponding voltage value is reached, AON_LDO and RET_LDO will be powered on first.
2. The Brown Out Reset (BOR) signal will be released after reaching the specified voltage. There is about 10 µs delay to generate a deglitch after releasing the BOR. Ensure that the AON_LDO and RET_LDO startup is complete.
3. The low frequency clock sources LFRC_32K and RNG_OSC start working after AON_LDO and RET_LDO startup.
4. Subsequently, the DC-DC converter begins to start.
5. After waiting for the DC-DC converter to stabilize for a period of time, CORE_LDO and ANA_IO_LDO will start to provide power to the digital domain and I/Os. At the same time, the high frequency clock sources HFXO_32M, HFRC_192M, and CPLL_192M start working. Since the crystal oscillator needs a certain amount of time to stabilize, the actual generation of the HFXO_32M clock will be slower than the HFRC_192M clock. Similarly, the CPLL_192M needs the HFXO_32M to get stable before it can work normally.
6. When both the basic power modules and the basic clock sources are working, the Power-on Reset (POR) signal is released, and the digital domain starts to work as well. Simultaneously, the ANA_IO_LDO output generates a voltage drop of 100 mV, indicating that the internal POR signal has been released. After completing a series of initialization of digital status, the CPU starts to work, allowing the software to start running.

3.3.2 Power Module

3.3.2.1 DC-DC Converter

3.3.2.1.1 Introduction

The DC-DC buck converter provides power to all the RF blocks, digital logic blocks, and memories.

3.3.2.1.2 Main Features

- Designed with soft start so that it can prevent inductor saturation
- Strong loading capability
- Precise Zero Current Detect

3.3.2.1.3 Regulation

The voltage of the DC-DC converter can be adjusted by configuring the field DCDC_REG3[7:3] in the RF_REG_1 register. The voltage step is determined by the MSB of the field DCDC_REG3[7:3].

- When the MSB of the field DCDC_REG3[7:3] is 0, the voltage step is 18 mV.
- When the MSB of the field DCDC_REG3[7:3] is 1, the voltage step is 25 mV.

Due to the deviation of the chip process, the voltage step might be slightly different. This norm is for reference only and the voltage step is subject to actual tests.

3.3.2.1.4 Electrical Specifications

Table 3-10 DC-DC converter specifications

Parameter	Symbol	Min.	Typ.	Max.	Unit	Remarks
Output voltage	V_{REG}	1.0	1.05	1.4	V	Programmable
Line regulation			1		mV/V	
Load current	I_{LOAD}			70	mA	
Load regulation			0.3		mV/mA	
Startup time	$T_{STARTUP}$		60		µs	
Voltage ripples	V_{RIPP}		16		mVpp	Peak-to-peak ripple voltage
Conversion efficiency	η_{DCDC}		81%			VBATL = 3.3 V, I_{LOAD} = 30 mA
Inductor	L		2.2		uH	I_{sat} ≥ 250 mA; 9.1 nH can be added for noise reduction.
Output capacitor	Cout		4.7		uF	

Note:

The parameter "Voltage ripples" is obtained based on L_{ext} = *2.2 µH + 9.1 nH* and C_{ext} = *4.7 µF*.

3.3.2.2 CORE_LDO

3.3.2.2.1 Introduction

CORE_LDO regulates the output of the DC-DC converter to supply power to all the digital logic blocks and memory blocks.

3.3.2.2.2 Main Features

- Low voltage drop out
- Supporting bypass mode to adapt to low voltage applications

3.3.2.2.3 Enable and Disable

The prerequisite for CORE_LDO to be enabled is to wait for the DC-DC converter to be powered on. When the system meets the following conditions, CORE_LDO will be enabled.

- When the system is being woken up, CORE_LDO is always enabled.
- If the PWR_SET.DIG_LDO_SET bit is set to 1, CORE_LDO will be enabled.

When the system meets the following conditions, CORE_LDO will be disabled.

- Before the system enters sleep mode, CORE_LDO will be disabled if A_SLP_CFG.TRN_OFF_DIG_LDO = 1.
- If the PWR_CLR.DIG_LDO_CLR bit is set to 1, CORE_LDO will be disabled.

By default, the system uses hardware to control signals. In order to improve flexibility, the SoC provides overriding registers to allow software to control signals. The current status of CORE_LDO can be configured by the overriding registers.

- Set the status to the PMU_INTF_OVR_VAL_0.DIG_LDO_EN bit.
- Enable the override function by setting the PMU_INTF_OVR_EN_0.DIG_LDO_EN bit to 1.
- The status of CORE_LDO can be obtained by reading the PMU_INTF_OVR_RD0.DIG_LDO_EN bit.

3.3.2.2.4 Regulation

The coarse voltage of CORE_LDO can be adjusted by configuring the field ANACORE_LDO_REG1[5:4] in the RF_REG_4 register. There are four options of the coarse voltage of CORE_LDO as shown in the table below.

Table 3-11 CORE_LDO coarse regulation

Voltage	RF_REG_4.ANACORE_LDO_REG1[5:4]
0.8 V	0x0
0.9 V	0x1
1.0 V	0x2 (default)
Bypass	0x3

The bypass mode can be used to bypass CORE_LDO, allowing the DC-DC converter to supply power directly. The RF_REG_4.ANACORE_LDO_REG1[6] can also be used to bypass CORE_LDO. However, If the bypass mode is enabled by setting the ANACORE_LDO_REG1[6] bit, the op-amp circuit of CORE_LDO will be powered off. Therefore, when CORE_LDO exits the bypass mode, it may take more time to restore CORE_LDO.

The voltage of CORE_LDO can also be automatically adjusted by the DDVS in order to meet the lowest power consumption. For detailed information of the DDVS, refer to "Section 3.3.6 Digital Dynamic Voltage Scaling (DDVS)".

3.3.2.2.5 Electrical Specifications

Table 3-12 CORE_LDO specifications

Parameter	Symbol	Min.	Typ.	Max.	Unit	Remarks
Supply voltage	V_{in}	1.05	1.15	1.4	V	DC-DC output
Output voltage	V_{core}	0.8	0.9	1.1	V	Programmable
Voltage drop	Vdrop		50		mV	
Load current range	I_{LOAD}	0.1		70	mA	VBAT = 3.3 V, voltage drop less than 5%
Load regulation			1.4		mV/mA	
Quiescent current	I_q		46		µA	
External load capacitance	C_L	0.1	1		µF	

3.3.2.3 RET_LDO

3.3.2.3.1 Introduction

RET_LDO provides the supply voltage to the retention instances of memory when CORE_LDO is disabled.

3.3.2.3.2 Main Features

- Extra low current consumption
- Stable and reliable for data saving and retention

3.3.2.3.3 Enable and Disable

In sleep mode, RET_LDO is used to provide power for retention of SRAM. Typically, SRAM is powered by CORE_LDO in active mode. When the system enters sleep mode, CORE_LDO will usually be turned off, and RET_LDO will be used to power SRAM. RET_LDO can be enabled or disabled by configuring the LPD_REG1[1] bit in the RF_REG_0 register. RET_LDO is enabled by default. For the power-on sequence of RET_LDO, refer to "Section 3.3.1 Functional Overview".

In ultra deep sleep mode, because SRAM does not need to be retained, RET_LDO can be turned off in order to further reduce power consumption.

3.3.2.3.4 Regulation

The voltage of RET_LDO can be adjusted by setting the field LPD_REG1[4:2] in the RF_REG_0 register. The voltage options of RET_LDO are shown in the table below.

Table 3-13 RET_LDO regulation

Voltage	RF_REG_0.LPD_REG1[4:2]
0.6 V	0x0
0.65 V	0x1
0.7 V	0x2
0.75 V	0x3 (default)
0.8 V	0x4
0.9 V	0x5
1.0 V	0x6
1.1 V	0x7

Appropriately reducing the voltage of RET_LDO can reduce the power consumption in sleep mode. Because RET_LDO is mainly used to maintain SRAM in sleep mode, the voltage of RET_LDO cannot be lower than the minimum retention voltage of the SRAM.

3.3.2.3.5 Electrical Specifications

Table 3-14 RET_LDO specifications

Parameter	Symbol	Min.	Typ.	Max.	Unit	Remarks
Supply voltage	V_{in}	2.0	3.3	3.6	V	
Output voltage	V_{ret}	0.65	0.8	1.0	V	Programmable
Load current range	I_{LOAD}	0.1		1	mA	
Quiescent current	I_q		60		nA	

3.3.2.4 AON LDO

3.3.2.4.1 Introduction

AON_LDO supplies voltage to the AON domain, including AON I/Os and digital logic blocks.

3.3.2.4.2 Main Features

- 1 mA load current
- Ultra low current consumption

3.3.2.4.3 Enable and Disable

AON_LDO is used to supply power to the AON domain. The voltage of AON_LDO can be configured by setting the field LPD_REG2[6:4] in the RF_REG_0 register, and it should be higher than the digital circuit workable voltage. AON_LDO is the basic supply of the AON domain which is constantly powered on, independently of active mode or any sleep mode. AON_LDO is powered on during the startup period and is unnecessary to be controlled. For the power-on sequence of AON_LDO, refer to "Section 3.3.1 Functional Overview".

3.3.2.4.4 Regulation

The voltage of AON_LDO can be adjusted by configuring the PMU_LPD_CFG register. The VAON_ACTIVE field controls the AON_LDO when MCU is ON. The VAON_SLEEP field controls the AON_LDO when MCU is OFF. The options of the voltage of the AON_LDO is shown in the table below.

Table 3-15 AON_LDO regulation

Voltage	PMU_LPD_CFG.VAON_SLEEP
0.75 V	0x0
0.8 V	0x1
0.85 V	0x2
0.9 V	0x3
0.95 V	0x4
1.0 V	0x5
1.05 V	0x6
1.1 V	0x7

Appropriately reducing the voltage of AON_LDO can reduce the power consumption in sleep mode. The voltage of AON_LDO should be higher than the always-on digital circuit workable voltage, in order to ensure normal work of the AON domain.

3.3.2.4.5 Electrical Specifications

Table 3-16 AON_LDO specifications

Parameter	Symbol	Min.	Typ.	Max.	Unit	Remarks
Supply voltage	V_{in}	2.0	3.3	3.6	V	
Output voltage	V_{aon}	0.8	1.0	1.1	V	Programmable
Load current range	I_{LOAD}	0.1		1	mA	
Quiescent current	I_q		60		nA	

3.3.2.5 ANA_IO_LDO

3.3.2.5.1 Introduction

ANA_IO_LDO provides the supply voltage to drive the I/O pads and Flash memories.

3.3.2.5.2 Main Features

- Strong load capability
- Programmable output range

3.3.2.5.3 Enable and Disable

When the system meets the following condition, ANA_IO_LDO will be enabled.

- When the system is being woken up, ANA_IO_LDO is always enabled.

When the system meets the following condition, ANA_IO_LDO will be disabled.

- Before the system enters sleep mode, ANA_IO_LDO is disabled and STB_IO_LDO has been enabled and stable, when A_SLP_CFG.TRN_OFF_IO_LDO = 1.

By default, the system uses hardware to control signals. In order to improve flexibility, the SoC provides overriding registers to allow software to control signals. The current status of ANA_IO_LDO can be controlled by the overriding registers.

- Set the status to the PMU_INTF_OVR_VAL_0.IO_LDO_EN bit.
- Enable the override function by setting the PMU_INTF_OVR_EN_0.IO_LDO_EN bit to 1.
- The status of ANA_IO_LDO can be obtained by reading the PMU_INTF_OVR_RD0.IO_LDO_EN bit.

3.3.2.5.4 Regulation

The voltage of ANA_IO_LDO can be adjusted by configuring the field IO_LDO_REG1[6:0] in the RF_REG_0 register. The default voltage of ANA_IO_LDO is set to 1.8 V.

3.3.2.5.5 Electrical Specifications

Table 3-17 ANA_IO_LDO specifications

Parameter	Symbol	Min.	Typ.	Max.	Unit	Remarks
Supply voltage	VBAT	2.0	3.3	3.6	V	
Output voltage	VIO_LDO_OUT	1.62	1.8	3.3	V	Programmable; please note that VBAT should be greater than the VIO_LDO output voltage plus the VIO_LDO voltage drop.
Voltage drop	Vdrop		100		mV	
Quiescent current	I_q		10		μA	
External load capacitance	C_L		1		μF	
Load current range	I_{LOAD}	30 mA	100 mA	160	mA	If VBAT = 3.0 V, I_{LOAD} = 100 mA; if VBAT = 2.0 V, I_{LOAD} = 30 mA

Parameter	Symbol	Min.	Typ.	Max.	Unit	Remarks
Load regulation			1.5		mV/mA	

3.3.2.6 STB_IO_LDO

3.3.2.6.1 Introduction

STB_IO_LDO provides the supply voltage to drive the I/O pads during sleep, helping reduce the sleep current.

3.3.2.6.2 Main Features

- Ultra low current consumption
- 1 mA load current
- Programmable output range

3.3.2.6.3 Enable and Disable

When the system meets one of the following conditions, STB_IO_LDO will be enabled.

- Before the system enters sleep mode, ANA_IO_LDO is disabled and STB_IO_LDO switch is selected, if A_SLP_CFG.TRN_OFF_IO_LDO = 1.
- If the PWR_SET.STB_IO_LDO_SET bit is set to 1, STB_IO_LDO will be enabled.

When the system meets one of the following condition, STB_IO_LDO will be disabled.

- When the system is being woken up, ANA_IO_LDO is enabled and stable.

3.3.2.6.4 Regulation

The coarse voltage of STB_IO_LDO can be adjusted by configuring the field IO_LDO_REG2[3:2] in the RF_REG_3 register. The options of the coarse voltage of STB_IO_LDO are shown in the table below. When the coarse voltage is selected, the fine voltage can also be configured by the field IO_LDO_REG2[7:4] in the RF_REG_3 register.

Table 3-18 STB_IO_LDO coarse regulation

Voltage	RF_REG_3.IO_LDO_REG2[3:2]
1.8 V	0x0
2.5 V	0x1
2.5 V	0x2
3.3 V	0x3

3.3.2.6.5 Electrical Specifications

Table 3-19 STB_IO_LDO specifications

Parameter	Symbol	Min.	Typ.	Max.	Unit	Remarks
Supply voltage	VBAT	2.2	3.3	3.6	V	
Output voltage	VIO_LDO_OUT	1.62	1.8	3.3	V	Programmable; please note that VBAT should be greater than the VIO_LDO output voltage plus the VIO_LDO voltage drop.
Voltage drop	Vdrop		100		mV	
Line regulation			2		mV/V	
Quiescent vurrent	I_q		0.1		μA	
External load capacitance	C_L		1		μF	
Load current range	I_{LOAD}			1	mA	

3.3.3 Power Management

3.3.3.1 Introduction

To minimize power consumption, the SoC supports power management in multiple power modes, including active mode, IDLE mode, sleep mode, and ultra deep sleep mode. The relationship between different power modes is shown in the figure below.

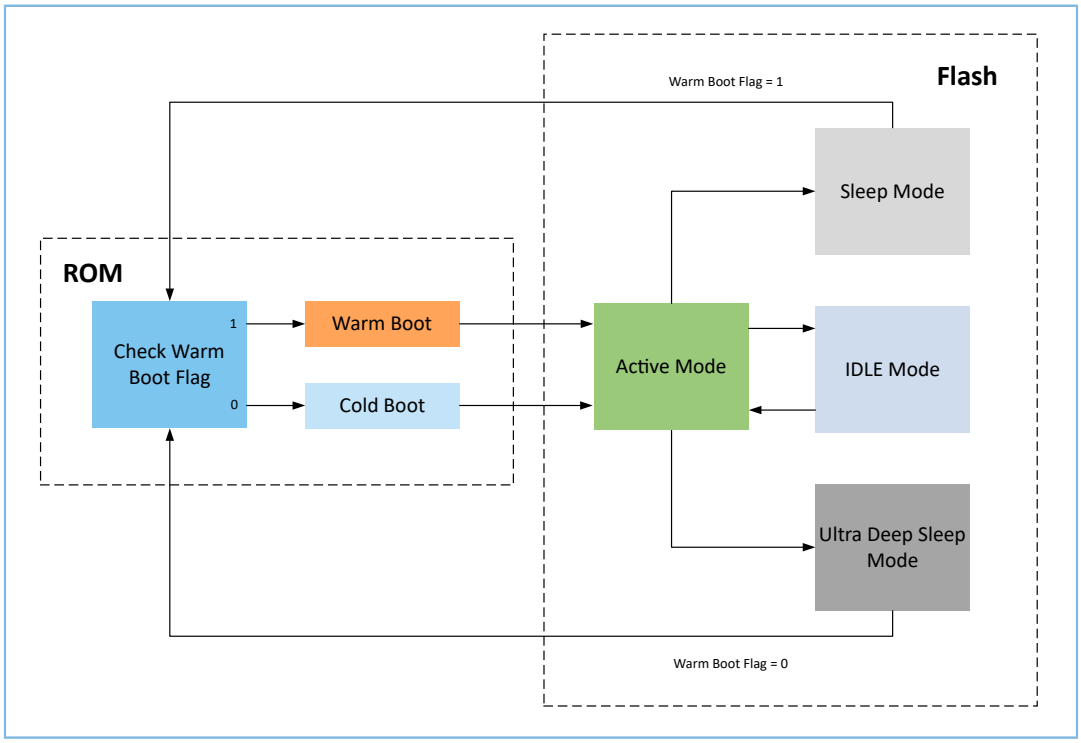

Figure 3-16 Power mode block diagram

- Cold Boot: Start the MCU from power-off or reset status. Current configuration and data are discarded. Program processing begins again with the initial values.
- Warm Boot: Restart the MCU from sleep status. Program processing starts once again where retentive configuration and data are retained.
- Check Warm Boot Flag: Whether to enter the warm boot can be determined by the warm boot flag in the SOFTWARE_REG0[8] register.

The possible transitions between different power modes are shown in the table below.

Table 3-20 Power mode transitions

Current Mode	Transition Action			
	Enter Active	Enter IDLE	Enter Sleep	Enter Ultra Deep Sleep
Active		WFI/WFE	AON_SLP_CTRL[0]	AON_SLP_CTRL[0]
IDLE	IRQ			
Sleep	Sleep Wakeup Source IRQ			
Ultra Deep Sleep	Ultra Deep Sleep Wakeup Source IRQ			

The table below displays an outline of the supported power modes and available features. In general, for each power mode, the lowest power configuration is set as the system default, with non-essential clocks and peripherals being deactivated. After that, software can selectively activate features as needed. The power management in different power modes for different modules is shown as follows.

Table 3-21 Power modes

Item	Software-configurable Power Modes				Holding Reset Pin
	Active	IDLE	Sleep	Ultra Deep Sleep	
CPU	Active	Off	Off	Off	Off
Flash	On	Standby	Off	Off	Off
SRAM	On	On	Retention	Off	Off
Radio	Available	Available	Off	Off	Off
Supply system	On	On			Off
Current			7.4 µA	5.0 µA	0.2 µA
Register retention	Full	Full	No	No	No
HFXO_32M	Available	Available	Off	Off	N/A
HFRC_192M	Available	Available	Off	Off	N/A
LFXO_32K	Available	Available	Available	Available	N/A
LFRC_32K	Available	Available	Available	Available	N/A
GPIO	Available	Available	Off	Off	Off

Copyright © 2024 Shenzhen Goodix Technology Co., Ltd.

Item	Software-configurable Power Modes				Holding Reset Pin
	Active	IDLE	Sleep	Ultra Deep Sleep	
AON IO	Available	Available	Available	Available	Off
MSIO	Available	Available	Available	Available	Off
I/O state retention	Available	Available	Available	Available	N/A
Timer	Available	Available	Off	Off	Off
Dual Timer	Available	Available	Off	Off	Off
PWM	Available	Available	Off	Off	Off
AON_WDT	Available	Available	Available	Available	Off
RTC	Available	Available	Available	Available	Off
SLP TIMER	Available	Available	Available	Available	Off
UART	Available	Available	Off	Off	Off
SPI	Available	Available	Off	Off	Off
I2C	Available	Available	Off	Off	Off
I2S	Available	Available	Off	Off	Off
PDM	Available	Available	Off	Off	Off
AES	Available	Available	Off	Off	Off
PKC	Available	Available	Off	Off	Off
HMAC	Available	Available	Off	Off	Off
TRNG	Available	Available	Off	Off	Off
ADC	Available	Available	Off	Off	Off
TEMP	Available	Available	Off	Off	Off
LPCOMP	Available	Available	Available	Available	Off
DMA	Available	Available	Off	Off	Off
Wakeup on Bluetooth LE	Available	Available	Available	Off	N/A
Wakeup on LPCOMP	Available	Available	Available	Off	N/A
Wakeup on RTC	Available	Available	Available	Off	N/A
Wakeup on SLP TIMER	Available	Available	Available	Available	N/A
Wakeup on AON IO	Available	Available	Available	Available	N/A
Wakeup on Reset Pin	Available	Available	Available	Available	Available
Brown Out Detector(BOD Interrupt)	Active	Active	Off	Off	N/A
Power On Reset (POR)	Active	Active	Active	Active	N/A

3.3.3.2 Power Mode

3.3.3.2.1 Active Mode

By default, the system enters active mode after power-on reset. In active mode, the Cortex®-M4F is actively executing code. Active mode supports normal operations of the processor and all of the peripherals that are currently available. The radio functionality is available. The system clock can be any available clock source (refer to Table 3-21). The power management by the PMU in active mode is shown in the figure below. The I/O supply in active mode is from ANA_IO_LDO in order to gain a stronger drive capability.

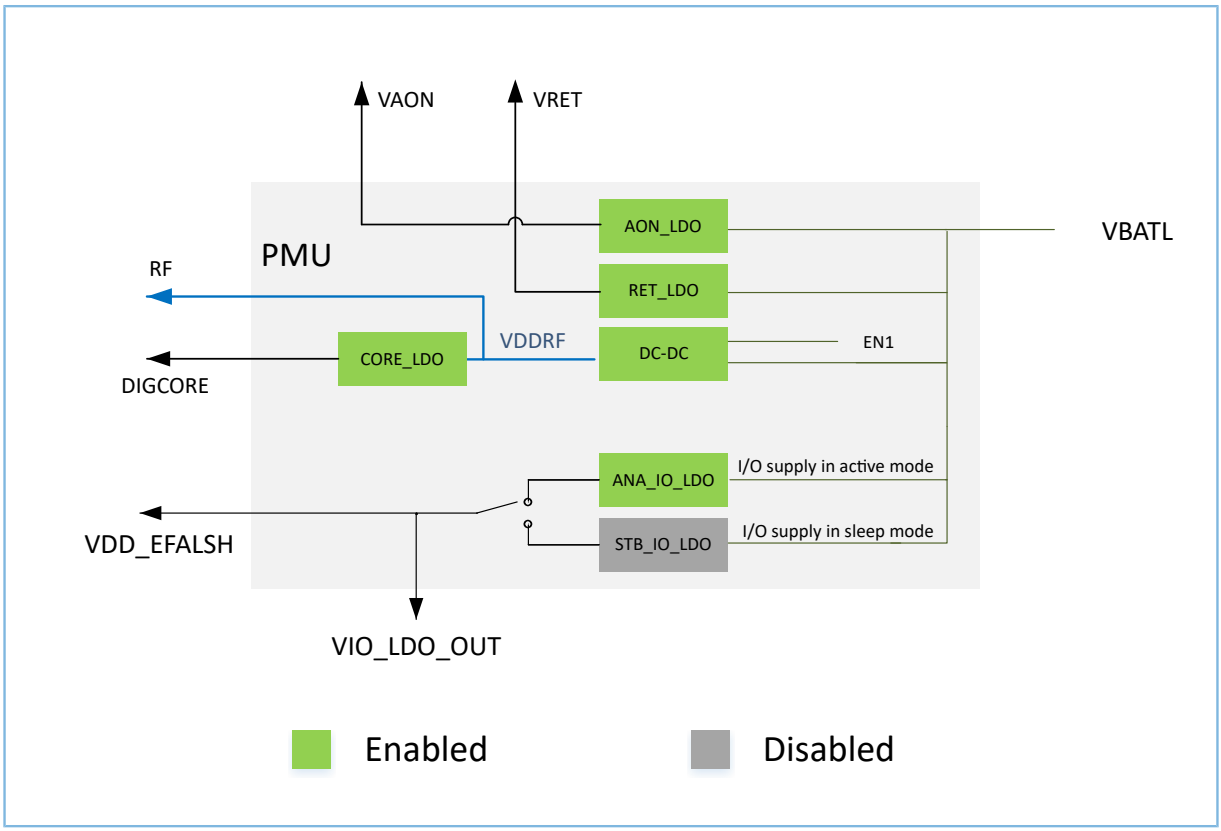

Figure 3-17 Power management in active mode

3.3.3.2.2 IDLE Mode

In IDLE mode, the Cortex®-M4F core is in sleep mode so that no code is executed. However, the remaining system is fully available. All peripheral functionalities and the radio functionality are still available. The system clock can be any available clock source (refer to Table 3-21). The power supply is still applied to the Cortex®-M4F core logic so that it can immediately become active on a wakeup event and begin executing instructions. The power management by the PMU in IDLE mode is similar to that in active mode, as shown in Figure 3-17.

3.3.3.2.3 Sleep Mode

In sleep mode, only the always-on (AON) domain is active. The Cortex®-M4F core is disabled and the radio functionality is not available. The AON domain peripherals are still available, such as AON_WDT, RTC, SLP TIMER, and LPCOMP. A part of the memory unit is used as retention for storing data that needs to be retained during sleep.

The AON power is supplied from AON_LDO, and the retention memory power is supplied from RET_LDO. The DC-DC convertor and CORE_LDO will be powered off. Only low frequency clocks are available in sleep mode. The I/O supply will be changed from ANA_IO_LDO to STB_IO_LDO in order to reduce power consumption during sleep mode.

The power management by the PMU in sleep mode is shown in the figure below.

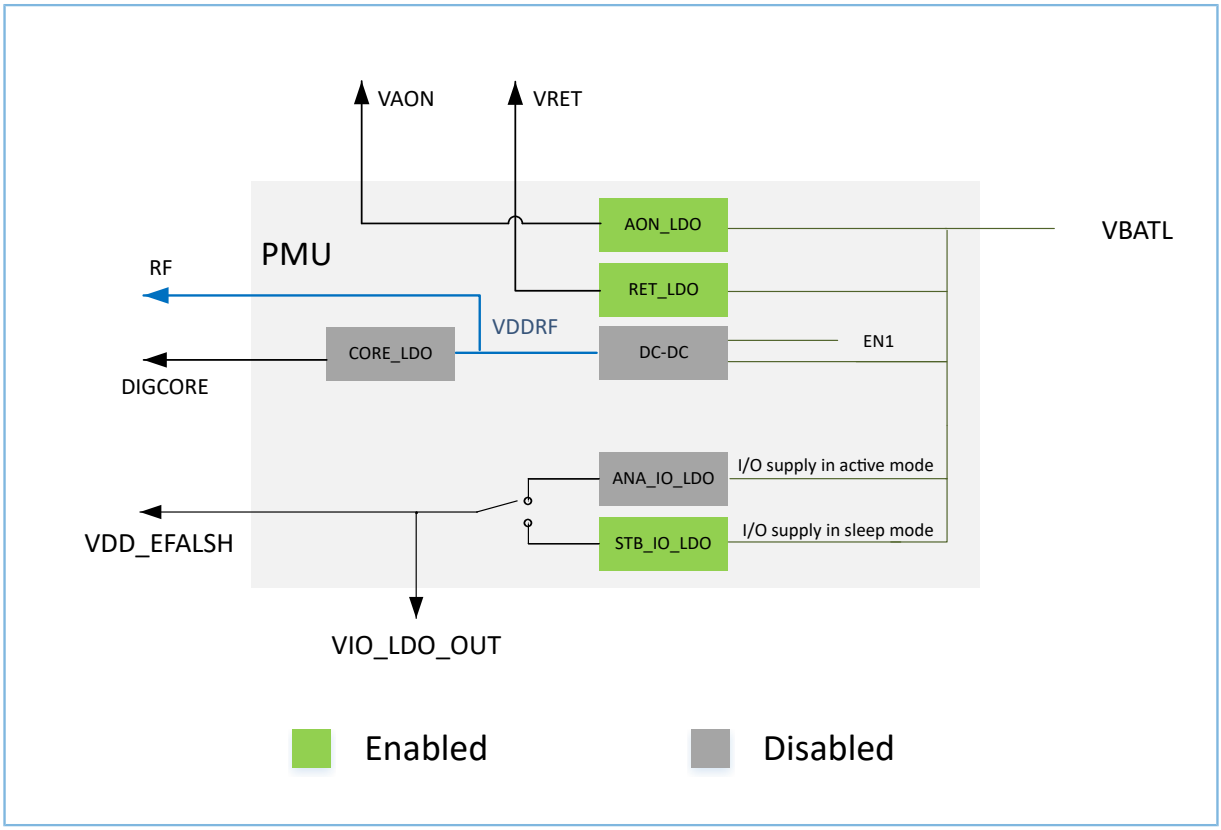

Figure 3-18 Power management in sleep mode

The I/Os are latched with the value they are in before the system enters sleep mode. This configuration can be achieved by configuring the DPAD_LE_CTRL register.

- When DPAD_LE_CTRL.SLEEP = 0 and DPAD_LE_CTRL.WAKEUP = 1, the I/Os are latched during sleep and automatically unlocked after wakeup.
- If DPAD_LE_CTRL.WAKEUP = 0, the I/Os can manually be unlocked after wakeup by writing 1 to DPAD_LE_CTRL.WR and DPAD_LE_CTRL.APPLY_WR in order.

This function has been implemented in GR5525 SDK. The default configuration in the SDK is DPAD_LE_CTRL.SLEEP = 0 and DPAD_LE_CTRL.WAKEUP = 0.

3.3.3.2.4 Ultra Deep Sleep Mode

Ultra deep sleep mode is the deepest power saving mode that the system can enter. In this mode, only the always-on (AON) domain is active with limited functionalities. The Cortex®-M4F core is disabled and the radio functionality is not available. All SRAMs are powered off, so the system restarts execution from cold boot when the system is woken up from ultra deep sleep mode. The power management by the PMU in ultra deep sleep mode is similar to that in

sleep mode, as shown in Figure 3-18. RET_LDO can be powered off in ultra deep sleep mode since all SRAMs are not retained.

The I/Os are latched with the value they are in before the system enters ultra deep sleep mode. This configuration is implemented in the same way as the sleep mode.

3.3.3.3 Low Power Mode Entry

3.3.3.3.1 IDLE Mode Entry

To transition from active mode to IDLE mode, a specific sequence of instructions is executed on the Cortex®-M4F core. First, specific bits in the Cortex®-M4F System Control Register must be set to determine the mode to enter. Refer to *The Definitive Guide to Arm Cortex-M3 and Cortex-M4 Processors*.

The Cortex®-M4F core can enter the IDLE status using one of the following two methods:

- Execute a Wait-For-Interrupt (WFI) instruction.
- Execute a Wait-For-Event (WFE) instruction.

3.3.3.3.2 Sleep Mode Entry

The sleep request can be initiated by configuring the AON_SLP_CTRL register. However, before the system enters sleep mode, it is necessary to ensure that the device is not in busy status and the remaining sleep time of the system is longer than the sleep threshold. When any high-speed peripheral/radio/DMA is working, or the Flash is being read/written/erased, it indicates the system is in busy status.

The integrated GR5525 SDK implements pre-sleep status checks and automatic backup of peripheral configurations that need to be retained before the system enters sleep mode. I/O and MCU peripherals with retention do not need to be reconfigured when being woken up again, and the CPU that continues execution from warm boot goes into the active mode.

A typical sleep mode entry procedure includes power mode check process and sleep enter process, as shown in the figure below. In the power mode check process, it is mainly to check whether the system can meet the conditions for entering the sleep mode. In the sleep enter process, the current system context should be saved, and then the system can request to enter the sleep mode.

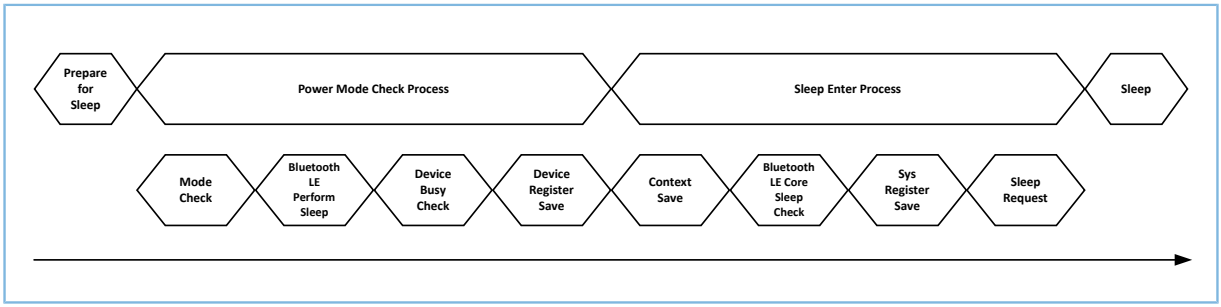

Figure 3-19 Typical sleep mode entry procedure

In the sleep enter process, the following operations will be performed:

1. When the PMU receives the sleep request, it will power off all the peripherals and Bluetooth LE Core. If the power-off is successful, the system is able to enter sleep mode. Otherwise, a sleep failure interrupt will be sent to the CPU.
2. Then the system clock is stopped, and both the I/O and SRAM will enter retention mode.
3. Subsequently, the MCU core is powered off. The power modules including the DC-DC convertor and CORE_LDO are turned off. High frequency clocks including HFXO_32M, CPLL_192M, and HFRC_192M are turned off. The Brown-out Detector (BOD) and Flash are disabled. The I/O supply is changed from ANA_IO_LDO to STB_IO_LDO. After that, ANA_IO_LDO is turned off.

The Cortex®-M4F processor provides a register called the System Control Register (SCR) to allow developers to select between the IDLE mode and sleep mode. This register is at an address of 0xE000ED10, and can be accessed in C programming using the “SCB->SCR” symbol. However, the sleep mode implementation of the integrated GR5525 SDK has nothing to do with this register. Arbitrary configuration of this register will cause unexpected errors.

3.3.3.3.3 Ultra Deep Sleep Mode Entry

The implementation of the system entering ultra deep sleep mode is similar to that in sleep mode. The difference is that more things will be closed in ultra deep sleep mode before requesting sleep than that in sleep mode. All SRAMs and peripherals will be powered off in order to minimize power consumption. In this mode, AON WDT and RTC will be disabled, and only the sleep timer and AON I/O can wake up the system. The slow frequency clock needs to choose RNG_OSC with div8 in order to minimize power consumption. These operations have been implemented in the GR5525 SDK. Since many states and registers are lost due to power loss, the system will wake up from a cold boot and reinitialize the whole system.

3.3.3.3.4 Timing Specifications

Table 3-22 Power mode entry specifications

Symbol	Description	Min.	Typ.	Max.	Unit
T_{ENT_IDLE}	Entry time to IDLE mode		4		Cycles
$T_{ENT_DEEP_SLEEP}$	Entry time to sleep mode		15		Cycles
$T_{ENT_ULTRA_DEEP_SLEEP}$	Entry time to ultra deep sleep mode		15		Cycles

3.3.3.4 Low Power Mode Exit

There are some events that will bring the processor back to the active mode from the IDLE mode. Because neither system state nor memory is lost, the system can quickly become active and continue to operate as before it went into the IDLE mode. The possible reasons to return to the active mode from the IDLE mode are:

- A reset
- An enabled interrupt is received by the NVIC.
- An event is received by the NVIC.
- A debug event is received from the DAP.

Due to the Cortex®-M4F processor will be powered off and all registers except the AON domain will be lost in sleep mode and ultra deep sleep mode, the processor will start executing from the zero address after the system wakes up. In sleep mode, the state of the system and the memory can be retained so that the system can go back to operate as before it went into the sleep mode through a warm boot. However, the system can only be re-initialized through a cold boot after waking up from the ultra deep sleep mode. For detailed description of the warm boot and the cold boot, refer to "Section 3.6 Startup Procedure".

A range of wake-up sources can bring the system back to the active mode from the sleep mode and the ultra deep sleep mode. The difference of the ultra deep sleep mode is that other wake-up sources will be blocked except for AON GPIO peripherals and the sleep timer. Only unmasked wakeup sources can bring the SoC back to the active mode. For detailed description of the wake-up sources, refer to "Section 3.3.4 Wakeup Source".

3.3.4 Wakeup Source

In order to reduce power consumption, the system will enter sleep mode when it is idle. In sleep mode, the MCU will be powered off and only the functional modules in the AON area work. In this case, the SoC needs a wakeup source to wake the system up when it needs to work.

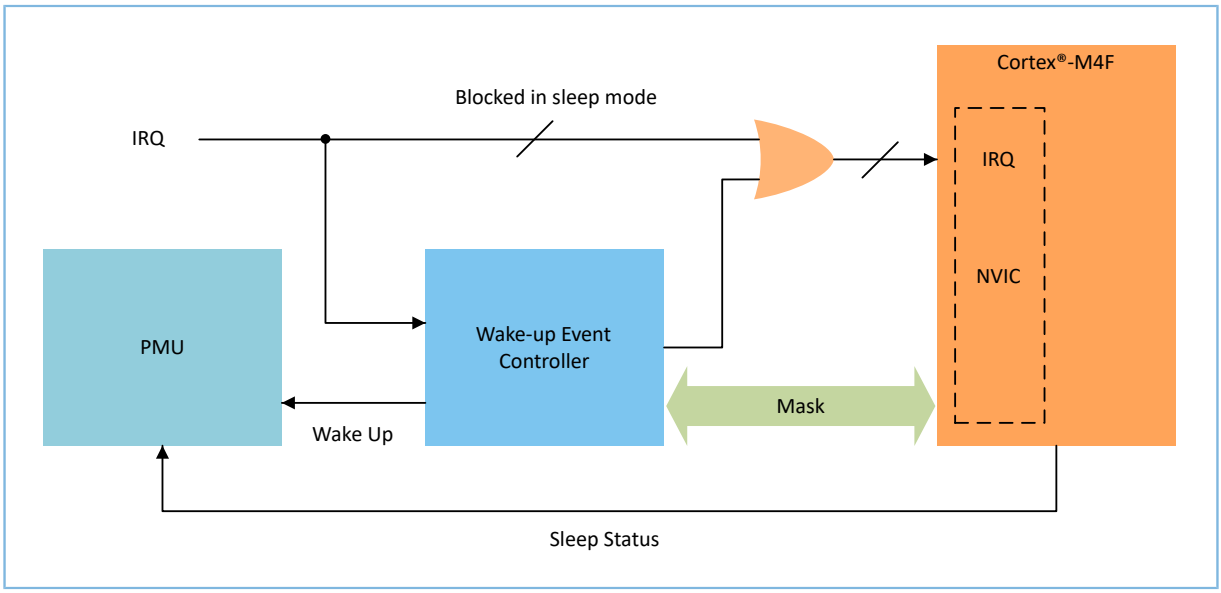

Figure 3-20 Wake-up control block diagram

Currently the SoC supports multiple wakeup sources to wake the system up from sleep. Each wakeup source belongs to a different function and is independent of each other.

3.3.4.1 Sleep Wakeup

In sleep mode, the device can be woken up by one of the following modules:

- Bluetooth LE controller
- LPCOMP module
- RTC counter

- Sleep timer
- AON GPIO peripheral
- BOD
- A reset

3.3.4.2 Ultra Deep Sleep Wakeup

In ultra deep sleep mode, the device can be woken up by one of the following modules:

- AON GPIO peripheral
- Sleep timer
- A reset

The system is reset when it is woken up from the ultra deep sleep mode.

3.3.4.3 Programming

The wakeup sources can be enabled by configuring the MCU_WAKEUP_CTRL register. The wakeup sources will wake the MCU up as a wakeup event after the MCU enters the sleep mode. There are nine wakeup sources, as shown in the table below.

Table 3-23 Wakeup source control register

Bits	Field Name	Description	IRQ No.
0	SLP_TIMER	Sleep Timer Wakeup	#42
1	EXT	AON IO Wakeup	#43
2	SMS_OSC	Bluetooth LE Timer Wakeup	#41
3	RTC0	RTC0 Wakeup	#49
4	BOD	BOD Wakeup	#60
5	COMP_RISE	LPCOMP Wakeup	#61
6	AON_WDT	AON WDT Wakeup	#44
7	RTC1	RTC1 Wakeup	#64
8	BLE_IRQ	Bluetooth LE IRQ Wakeup	#18

By reading the AON_SLP_EVENT register, developers can find out which wakeup source triggered the wakeup event. Writing 0 to the corresponding bit of the AON_SLP_EVENT register can clear the wakeup source event. For an external wakeup source event, developers need to disable the trigger source in the MCU_WAKEUP_CTRL register first, and then clear the event by writing 0 to the AON_SLP_EVENT register. Otherwise, because the trigger source is enabled, the event will be triggered again immediately after it is cleared.

Note that the AON_SLP_EVENT register should be cleared before entering sleep mode. Otherwise, the MCU cannot enter sleep mode.

Copyright © 2024 Shenzhen Goodix Technology Co., Ltd.

3.3.5 Brown-out Detector (BOD)

3.3.5.1 Introduction

Brown-out detector (BOD) is a voltage monitoring circuit whose functions include BOD Reset and BOD Interrupt. BOD Reset circuit puts the system into reset status when the supply falls below the Brown-Out Reset threshold. BOD Interrupt circuit is used to indicate the application when the supply voltage drops below the configured interrupt threshold. BOD Interrupt circuit will be prompted in two stages. When the supply voltage drops below the configured interrupt threshold, BOD_ASSERT_IRQn is triggered. When the voltage rises above the sum of the hysteresis voltage and the configured interrupt threshold, BOD_DEASSERT_IRQn is triggered.

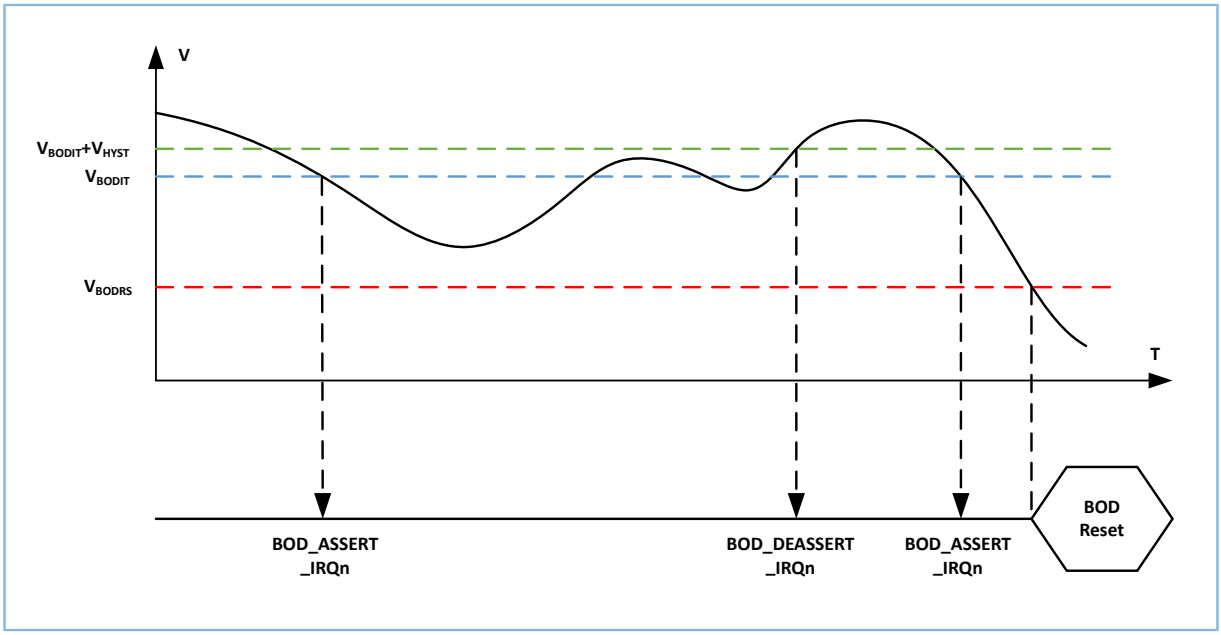

Figure 3-21 BOD Reset and BOD interrupt

3.3.5.2 Main Features

- Programmable BOD interrupt levels
- Hysteresis voltage prevents repeated interruptions.
- Voltage drops and rises with separate interruptions

3.3.5.3 Functional Description

3.3.5.3.1 Architecture

The core of the BOD lies in the use of two voltage comparators, as shown in the figure below, one for interrupts and one for resets. When the VBATL voltage is lower than the VREF voltage, a falling edge interrupt or reset will be generated. The BOD Reset is enabled by default. The BOD Interrupt is optional and the Interrupt threshold level is programmable, so that developers need to configure the BOD Interrupt through corresponding API integrated in the SDK. Developers can also configure the BOD through specific registers.

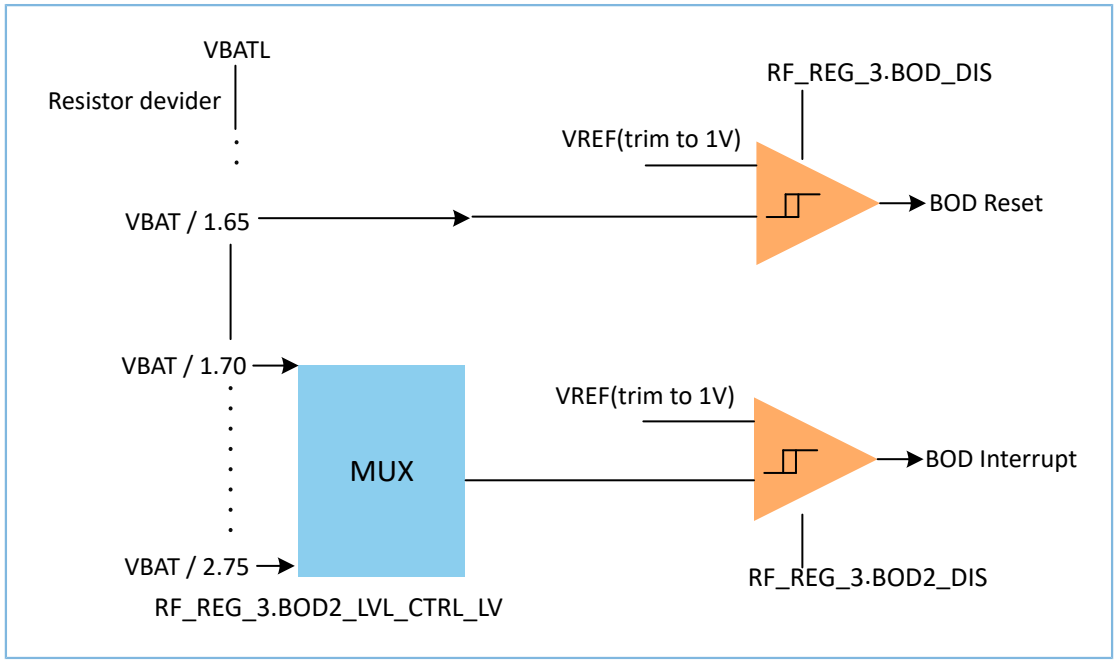

Figure 3-22 Brown-out detector block diagram

In sleep mode, BOD interrupt is not available, whereas BOD reset is available in any power modes. Refer to Table 3-21 for details.

A hysteresis voltage (V_{HYST}) of approximately 50 mV prevents voltage fluctuations from causing interrupts to be continuously triggered. Developers also need to configure appropriate interrupt threshold to prevent interrupts from being triggered frequently.

Note that the BOD Interrupt and Reset thresholds will not change in different PMU modes.

3.3.5.3.2 Programming

The BOD Reset can be enabled by configuring RF_REG_3.BOD_DIS. The BOD Reset threshold (V_{BODRS}) is fixed, and related parameters are listed in Table 3-24.

The BOD Interrupt can be enabled by configuring RF_REG_3.BOD2_DIS. The BOD Interrupt threshold (V_{BODIT}) is a variable to be configured through RF_REG_3.BOD2_LVL_CTRL_LV. The voltage range can be found in Table 3-25.

The BOD Interrupt is also generated when the supply voltage is below the threshold at the time that the BOD Interrupt is enabled, or if the threshold is reconfigured to a higher value than the current voltage. As shown in Figure 3-23, at T = 0, V_{BATL} has already fallen below V_{BODIT} (for example, when the startup voltage is stable and VBATL is below the interruption threshold), and then BOD_ASSERT_IRQn is generated. The BOD_ASSERT_IRQn will also be generated when the threshold is reset from V_{BODIT} to V_{BODIT2} above V_{BATL}.

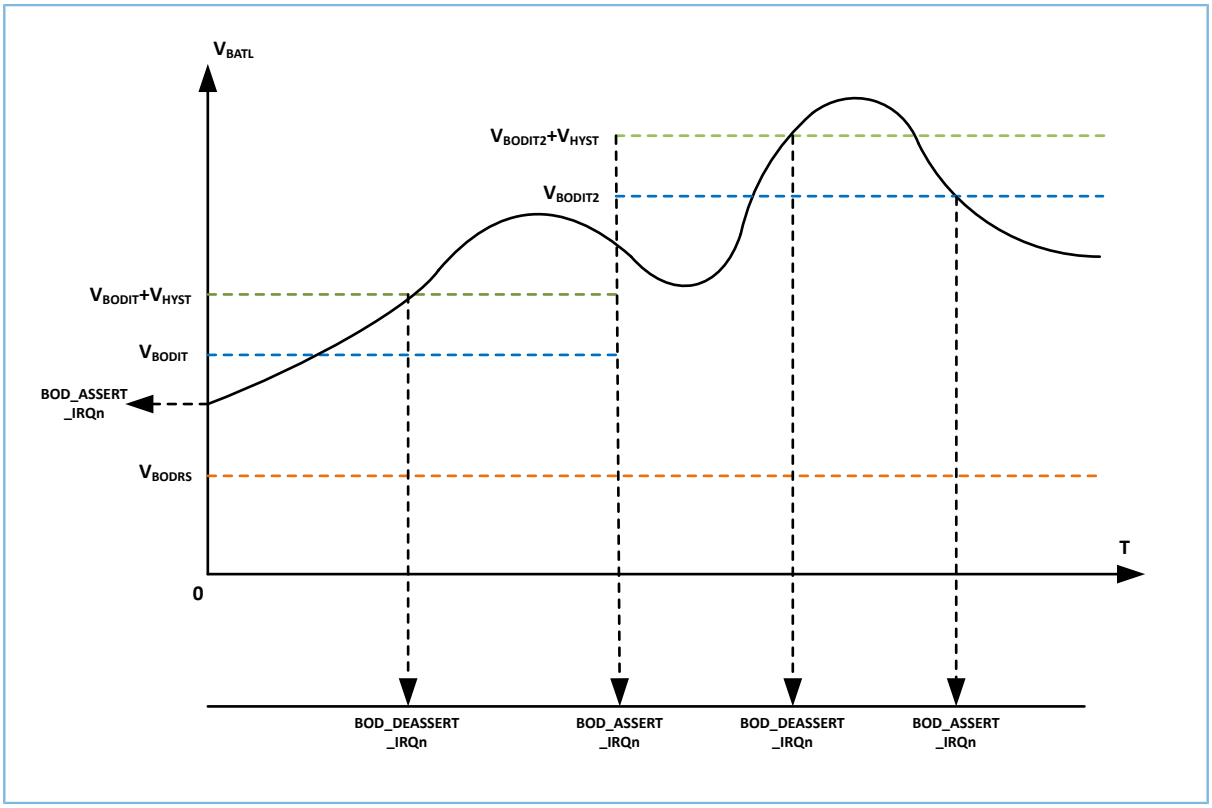

Figure 3-23 BOD Interrupt special case

3.3.5.4 Electrical Specifications

Table 3-24 BOD specifications

Parameter	Symbol	Min.	Typ.	Max.	Unit	Remarks
Brown-Out reset threshold	TH_BOD		1.60		V	

Table 3-25 BOD2 specifications

Parameter	Symbol	Min.	Typ. (BOD_LVL: Voltage Value)	Max.	Unit	Remarks
Brown-out interrupt threshold	TH_BOD2		0x0: 1.64 0x1: 1.73 0x2: 1.83 0x3: 1.94 0x4: 2.07 0x5: 2.24 0x6: 2.44 0x7: 2.67		V	

3.3.6 Digital Dynamic Voltage Scaling (DDVS)

3.3.6.1 Introduction

The performance of CMOS circuits is affected by PVT (Process, Voltage, Temperature) deviations. In the case of fixed P/T, Speed and Voltage are positively correlated within the allowable range of work. The dynamic power consumption of the circuit is positively related to the square of frequency and voltage:

$$P_{\mathrm{dyn}} = C_{avg} V_{DD}^{2} f$$

Digital Dynamic Voltage Scaling (DDVS) uses the correlation between voltage, delay, and power consumption to dynamically adjust the voltage of CORE_LDO, to achieve a balance and optimization of performance and power consumption on chips with different process variations.

3.3.6.2 Main Features

- Automatic mode and manual mode
- Ringo output frequency division
- Up to four Ringo inputs
- Supporting generating or masking error interrupt

3.3.6.3 Functional Description

3.3.6.3.1 Architecture

As shown in the figure below, DDVS supports up to four Ringo inputs. Each Ringo input is first counted according to the reference clock. Then the count results are compared with the lookup table (LUT) written by MCU in advance. Finally, the appropriate voltage value is obtained by calculation, and the voltage of CORE_LDO is adjusted through the PMU. Since the clock of the APB is asynchronous to the reference clock used inside the DDVS module, DDVS will perform clock synchronization processing.

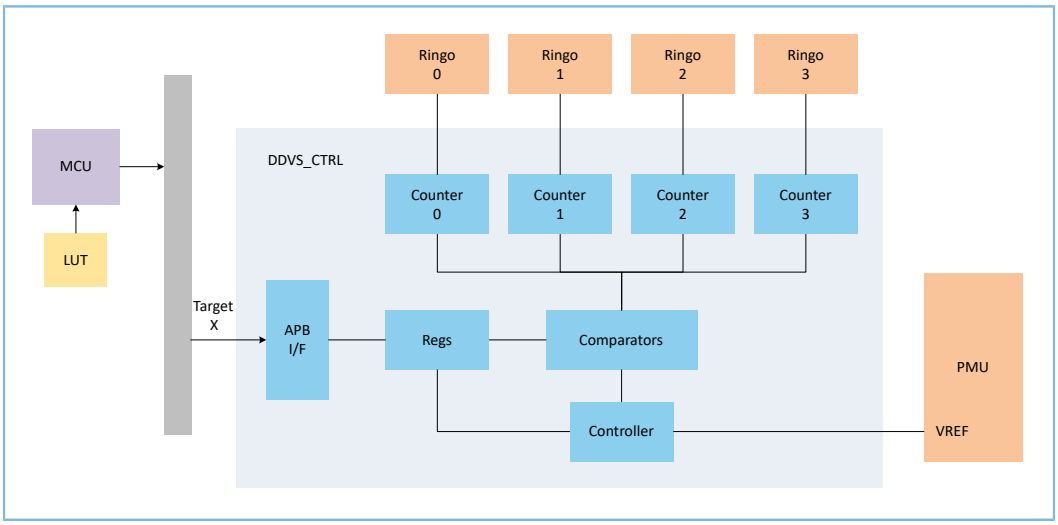

Figure 3-24 DDVS block diagram

3.3.6.3.2 Programming

DDVS can be enabled or disabled by configuring the DDVS_EN.DDVS_EN bit.

DDVS supports two working modes: automatic mode and manual mode, which can be configured through the DDVS_CFG_1.CONF_DDVS_MODE bit.

Automatic mode

In automatic mode, the MCU finds the upper limit value in the lookup table which is generated from testing and estimation, and writes the value into the CONF_TARGET_CNT field in the DDVS_CFG_2 register of the module.

After the DDVS module is enabled, according to the current value of each Ringo, the voltage of CORE_LDO is automatically adjusted until the target is reached.

- When the chip speed is higher than the target, DDVS will reduce the voltage until it meets the performance requirements and LOCK to reduce power consumption.
- When the chip speed is lower than the target, DDVS will increase the voltage until it meets the performance requirements and LOCK.

When the voltage drops to the lowest level, DDVS will stop the adjustment and LOCK. When the voltage rises to the highest level and still fails to meet the target, DDVS will stop the adjustment and report an exception. The interrupt is controlled by the DDVS_CFG_1.CONF_INT_EN bit. The IRQ of DDVS error is #63 DDVS_ERR_IRQ.

When the speed exceeds the high threshold in LOCK mode, DDVS will re-enter the adjustment mode and increase the voltage. The high threshold can be configured through the field DDVS_CFG_2.CONF_THRESHOLD_FAST.

When the speed is lower than the low threshold in LOCK mode, DDVS will re-enter the adjustment mode and decrease the voltage. The low threshold can be configured through the field DDVS_CFG_1.CONF_THRESHOLD_SLOW.

Manual mode

In manual mode, the MCU adjusts the voltage by reading the real-time value of each Ringo and judging by the software algorithm. Usually, developers need to manually adjust the voltage of CORE_LDO, and then compare the Ringo count value obtained by DDVS with the expected value, to determine whether the current voltage of CORE_LDO is appropriate. The larger of the Ringo counter value, the slower the system, so that the voltage of CORE_LDO should be increased.

Each Ringo can be enabled separately through the field DDVS_CFG_1.CONF_RINGO_EN. When multiple Ringos are sampled at the same time, if the value of each counter is different, the maximum value is suitable for the calculation.

DDVS supports configuring the output frequency range of the Ringo counter through the field DDVS_CFG_1.CONF_DIV_FACTOR to avoid the Ringo counter value overflowing or being too small. When the DDVS error interrupt is enabled, the DDVS error interrupt can be cleared only through the DDVS_CFG_1.ERR_INT bit in manual mode. The current status of DDVS can be checked by the DDVS_FSM register.

Note:

The sampling clock for DDVS should be enabled through the DDVS_CLK_CTRL.CLK_EN bit. The clock source can be selected by the field DDVS_CLK_CTRL. CLK_SEL. Due to the sampling accuracy, the HFXO_32M based clock is generally selected as the sampling clock.

3.3.7 Power Mapping on Pin

3.3.7.1 Introduction

Voltages from power modules can be measured by extracting the voltages to MSIO_4. This function allows users to obtain the SoC internal voltage by measuring the voltage from an external pin. In specific occasions, the internal voltage of the SoC can be adjusted to a reasonable level.

3.3.7.2 Main Features

- Extracting DC-DC converter voltage to MSIO_4
- Extracting CORE_LDO voltage to MSIO_4
- Extracting ANA_IO_LDO voltage to MSIO_4
- Extracting STB_IO_LDO voltage to MSIO_4
- Extracting AON_LDO voltage to MSIO_4

3.3.7.3 Programming

The voltages of power modules (such as DC-DC converter, CORE_LDO, ANA_IO_LDO, and STB_IO_LDO) can be extracted to MSIO_4. The mapping can be achieved with software by:

- Configuring the RF_REG_1.DCDC_REG1 register to enable the voltage extraction function: DCDC_REG1 = 0x80
- Making sure that the MSIO_4 is in analog mode
- Configuring the RF_REG_2.MUX1_REG1 register to select the modules for voltage extraction

Table 3-26 Power module mapping to MSIO_4

Power Module	RF_REG_2.MUX1_REG1
DC-DC converter	0xD4
CORE_LDO	0xCC
ANA_IO_LDO	0xC4
STB_IO_LDO	0xC7
AON_LDO	0xFA

3.3.8 Power Consumption

The software constantly adjusts the power supplies of the power management modules (such as DC-DC/LDO) according to different temperatures and system clock sources, so estimating the current consumption of an application can be challenging when the measurements cannot be performed directly on different test conditions. To facilitate the estimation, a set of current consumption scenarios are provided to show the typical current drawn from the VBATL supply.

Each scenario specifies a set of operations and conditions to apply to the given scenario. All scenarios are listed in "Section 3.3.8.1 Electrical Specifications". The table below shows a set of common conditions used in all scenarios, unless otherwise stated in the description of a given scenario.

Table 3-27 Common conditions

Condition	State	Remarks
Supply	VBAT = 3.3 V	
Temperature	25°C	
CPU	WFI/WFE	
Peripheral	All idle	
Clock	System clock = 64 MHz@HFRC, low clock = 32.768 kHz@LFXO_32K	
Flash clock	64 MHz	
Core RAM	Full 256 KB retention	In active mode, full 256 KB RAM is powered; in sleep mode, full 256 KB RAM is retained.
Cache enabled	Yes, 4-way associative	Only applied when the CPU is running from Flash memory
Bluetooth LE Stack	Disabled	
Security Mode	Disabled	
32 MHz crystal	32 MHz	Only applied when the high-frequency crystal oscillator (HFXO_32M) is running, and HFXO_32M is used when the radio is running.
32 KHz crystal	32.768 KHz	Only applied when the low-frequency crystal oscillator (LFXO_32K) is running
BOD/IO_LDO	BOD Enabled/IO_LDO On	
VDDIO	1.8 V	
DC-DC/CORE_LDO	DC-DC = 1.1 V/CORE_LDO = 0.93 V	
Regulator	DC-DC	
Flash	Standby mode	In active mode, Flash is powered on; in sleep mode, Flash is powered off.
Compiler version	MDK-ARM Plus Version: 5.26.2.0	
Compiler flag	--C99 -c --cpu Cortex-M4F.fp -D__MICROLIB --li -g -O3 --apcs=interwork --split_sections -D__UVISION_VERSION="526"	

3.3.8.1 Electrical Specifications

The current consumption data listed in the following tables is applicable to all devices and is obtained under the common conditions in Table 3-27, unless otherwise stated.

The minimum and maximum values in the following tables represent the worst conditions across process variation at TA = 25°C.

3.3.8.1.1 Sleep Current Consumption

Table 3-28 Current consumption in sleep modes

Symbol	Condition 1	Condition 2	Condition 3	Min.	Typ.	Max.	Unit
$I_{UDS,RAM_OFF, RESET}$	Ultra deep sleep mode	No RAM retention	Woken up on reset		5.0		μA
$I_{UDS,RAM_OFF, AON_GPIO}$			Woken up on AON GPIO		5.0		μA
$I_{UDS,RAM_OFF,SLP_TIMER}$			Woken up on Sleep Timer		5.0		μA
$I_{UDS,RAM_OFF,SLP_TIMER_LFRC}$			Woken up on Sleep Timer (running from LFRC_32K clock)		5.4		μA
$I_{S,RAM256K_RET,BLE}$	Sleep mode	Full 256 KB RAM retention	Woken up on Bluetooth LE controller		7.3		μA
$I_{S,RAM256K_RET,BLE_LFRC}$			Woken up on Bluetooth LE controller (running from LFRC_32K clock)		7.9		μA
$I_{S,RAM256K_RET,SLP_TIMER}$			Woken up on sleep timer		7.4		μA
$I_{S,RAM256K_RET,COMP}$			Woken up on LPCOMP		7.8		μA
$I_{S,RAM256K_RET,SLP_TIMER_LFRC}$			Woken up on sleep timer (running from LFRC_32K clock)		8.0		μA
$I_{S,RAM256K_RET,RTC}$			Woken up on RTC		7.4		μA

3.3.8.1.2 MCU Current Consumption

Table 3-29 MCU current consumption

Symbol	Condition 1	Condition 2	Min.	Typ.	Max.	Unit
$I_{COREMARK}$	CPU running CoreMark from Flash			3.5		mA
$I_{COREMARK,96M_HFXO}$		System clock = 96 MHz from HFXO clock		5.1		mA
$I_{COREMARK,16M_HFRC}$		System clock = 16 MHz from HFRC clock		1.8		mA
I_{LOOP}	CPU running while loop from Flash			3.6		mA
$I_{LOOP,96M_HFXO}$		System clock = 96 MHz from HFXO clock		5.0		mA

Symbol	Condition 1	Condition 2	Min.	Typ.	Max.	Unit
I_{IDLE}	CPU idle			1.5		mA
$I_{IDLE,96M_HFXO}$		System clock = 96 MHz from HFXO clock		1.9		mA

3.3.8.1.3 Radio Current Consumption

3.3.8.1.3.1 Radio Transmitting Current Consumption

Table 3-30 Radio transmitting current consumption

Symbol	Condition 1	Condition 2	Min.	Typ.	Max.	Unit
$I_{TRANSMIT,0dBm,1Mbps,HFXO64M}$	1 Mbps Bluetooth LE	Radio transmitting @0 dBm output power		6.3		mA
$I_{TRANSMIT,0dBm,1Mbps,HFXO16M}$	1 Mbps Bluetooth LE, System clock = 16 MHz from HFXO clock			5.6		mA
$I_{TRANSMIT,0dBm,1Mbps,HFXO96M}$	1 Mbps Bluetooth LE, System clock = 96 MHz from HFXO clock			7.7		mA
$I_{TRANSMIT,7dBm,1Mbps,HFXO96M}$		Radio transmitting @7 dBm output power		14.3		mA
$I_{TRANSMIT,-20dBm,1Mbps,HFXO96M}$		Radio transmitting @-20 dBm output power		5.0		mA

3.3.8.1.3.2 Radio Receiving Current Consumption

Table 3-31 Radio receiving current consumption

Symbol	Condition 1	Condition 2	Min.	Typ.	Max.	Unit
$I_{RECEIVE,1Mbps,64M_HFXO}$	Radio receiving @ 1 Mbps Bluetooth LE mode			5.3		mA
$I_{RECEIVE,1Mbps,96M_HFXO}$		System clock = 96 MHz from HFXO clock		6.2		mA
$I_{RECEIVE,1Mbps,16M_HFXO}$		System clock = 16 MHz from HFXO clock		4.6		mA

3.3.8.1.4 ADC Current Consumption

Table 3-32 ADC current consumption

Symbol	Condition 1	Condition 2	Min.	Typ.	Max.	Unit
$I_{ADC,4KHz,128_POINTS}$	CPU running, acquiring 128 points	ADC sampling@4 kHz sample clock		200		µA

Copyright © 2024 Shenzhen Goodix Technology Co., Ltd.

Symbol	Condition 1	Condition 2	Min.	Typ.	Max.	Unit
$I_{ADC,16KHz,128_POINTS}$		ADC sampling@16 kHz sample clock		200		μA
$I_{ADC,1MHz,128_POINTS}$		ADC sampling@1 MHz sample clock		200		μA
$I_{ADC,16MHz,128_POINTS}$		ADC sampling@16 MHz sample clock		200		μA

3.4 Reset

3.4.1 Functional Overview

A reset is triggered by power, hardware or software in SoC.

- Power reset includes power-on reset and brown- out reset.
- Hardware reset includes external reset pin (CHIP_EN) and watchdog reset.
- Software reset refers to the NVIC System Reset Register (AIRCR) as defined in Cortex®-M4.

The reset structureis illustrated in the figure below.

The PMU, triggered by CHIP_EN pad, POR circuit or BOD reset circuit, generates an active low por_rst_n reset signal. The watchdog reset by the watchdog timer in the always-on power domain generates an active low aon_wdog_rst_n reset signal. Either of these two reset signals will cause an active low pwr_rst_n reset signal to the power state controller, resulting in generating a reset signal of the SoC.

The power state controller generates an active low MCU Core Reset (mcu_core_rst_n), which goes directly to Cortex®-M4F core reset input. In order to support CPU-initiated system reset, the System Reset Request output of Cortex®-M4F is combined with mcu_core_rst_n, and the resulting System Reset (sys_rst_n) is connected to MCU peripherals as well as the XF Cache and always-on peripherals.

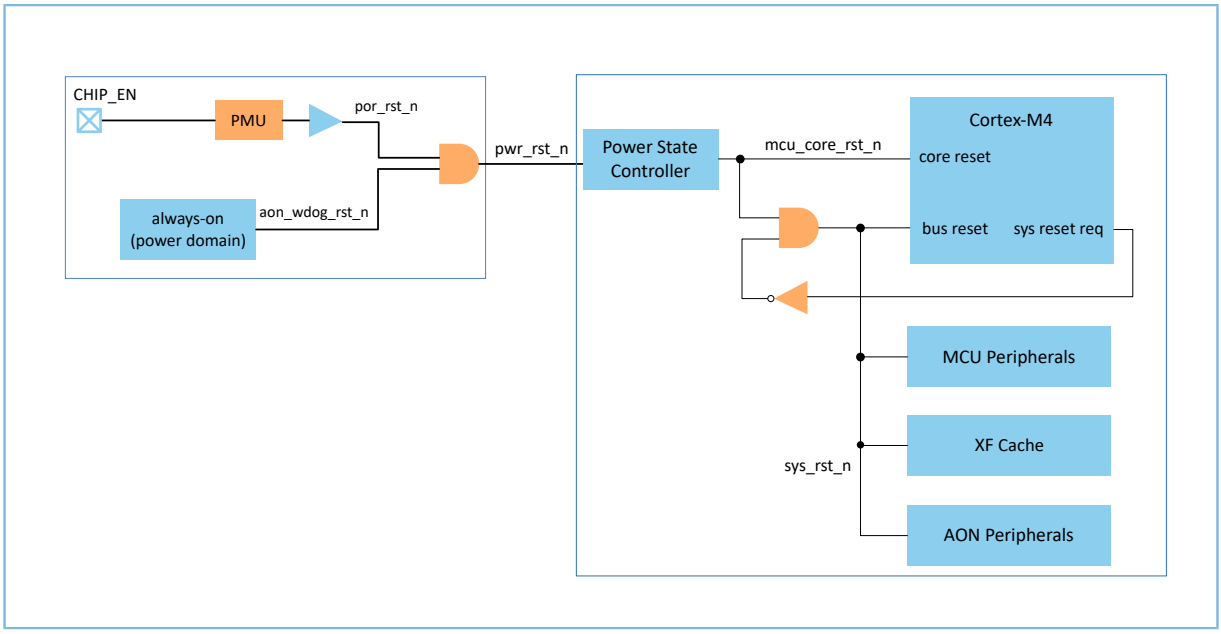

Figure 3-25 Reset structure

The reset table is illustrated as below:

Table 3-33 Impact of reset sources on modules

Reset Source	POR	BOD	Watchdog	SYSRESETREQ
Power state controller	Impacted	Impacted	Impacted	Not impacted
Always-on power control	Impacted	Impacted	Impacted	Not impacted
Always-on peripherals	Impacted	Impacted	Impacted	Impacted
Always-on memory control	Impacted	Impacted	Impacted	Impacted
Always-on others	Impacted	Impacted	Impacted	Impacted
Bluetooth LE timer	Impacted	Impacted	Impacted	Not impacted
Bluetooth LE core	Impacted	Impacted	Impacted	Not impacted
MCU subsystem	Impacted	Impacted	Impacted	Impacted
Cortex®-M4 core	Impacted	Impacted	Impacted	Impacted
Cortex®-M4 debug	Impacted	Impacted	Impacted	Not impacted

3.4.2 Power Reset

3.4.2.1 Power-on Reset

The Power-on Reset (POR) initializes the system when the supply voltage is above the power-on threshold. The system will be held in the reset phase until the supply voltage has reached the minimum operating voltage. The power-on threshold V_{work} of the device is about 1.4 V.

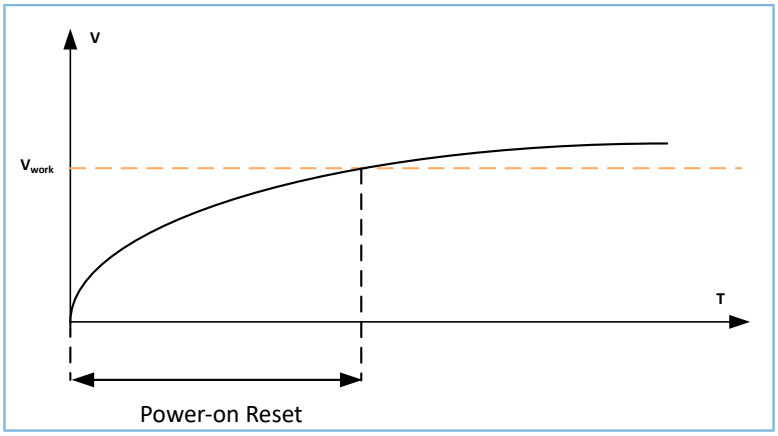

Figure 3-26 Power-on reset

3.4.2.2 Brown-out Reset

Brown-out Reset is dependent on the Brown-out detector (BOD). BOD reset circuit puts the system into reset state when the supply falls below the Brown-Out Reset threshold.

For detailed description of the BOD reset, refer to "Section 3.3.5 Brown-out Detector (BOD)".

3.4.3 Hardware Reset

3.4.3.1 External Reset Pin

The active-low CHIP_EN pin can be used to generate a POR reset by using an off-chip component. There is no internal pull-up resistor on this pin. The CHIP_EN pin is not maskable. The up voltage of the CHIP_EN pin is 1.1V, and the down voltage of the CHIP_EN pin is 0.8 V.

3.4.3.2 Watchdog Reset

A watchdog timer (WDT) reset is generated when the watchdog timer times out. The watchdog is on by default, and its time-out count value is 200,000 clock cycles of "aon_timer_clock", roughly 6 seconds by default. The software will re-perform cold boot after watchdog reset.

For detailed description of the watchdog timer, refer to "Section 4.3.4 Always-on Watchdog (AON_WDT)".

3.4.4 Software Reset

A reset may be generated via software by using the NVIC System Reset Register (AIRCR) defined in the Cortex®-M4. A software reset can be generated by controlling the SYSRESETREQ bit of this register. The software will re-perform cold boot after software reset.

For detailed description of the NVIC reset, refer to *The Definitive Guide to Arm Cortex-M3 and Cortex-M4 Processors*.

The SoC also provides software reset for the full digital logic by configuring the AON_SW_RST.FULL bit. However, the application still recommends using NVIC System Reset Register as the software reset in order to avoid unexpected exceptions.

3.4.5 Reset Reason Recorder

In the SoC, a reset reason recorder module is implemented in the always-on domain. When a reset is triggered, a specific flag register is set by hardware. The flag register is accumulative, which means it hold the status until a software clear is applied. The reset recorder will be helpful in debug when the system is reset accidentally.

3.4.5.1 Reset Reason Status

The reset reason status can be obtained through the DBG_REG_RST_SRC register. As shown in Figure 3-25, POR reset or watchdog reset will cause Power State Controller to reset, thus resulting in an active mcu_core_rst_n reset to the SoC. Using the digital logic software reset will record as system full reset in the reset reason recorder. When the digital logic power changes in POR reset, system full reset, AON Watchdog reset and sleep reset, the sleep reset will be recorded in the reset reason recorder.

Table 3-34 States of reset reason recorder

Item	**DBG_REG_RST_SRC [4:0]**				
Reset reson	**bit4**	**bit3**	**bit2**	**bit1**	**bit0**
POR reset	1	1	1	1	1
System full reset	1	1	1	1	1
System partial reset	1	0	0	0	0
AON Watchdog reset	1	1	1	1	1
System reset (SYSRESETREQ)	0	0	1	0	0

3.4.5.2 Programming

When the DBG_REG_RST_SRC register is used to clear flags. The following applies:

- Writing “1” to both bit[25] and bit[31] starts a clearing process.
- Asclock domain crossing takes some time, perform polling on bit[30] until the Reset Reason Recorder is not busy (low).
- Writing “0” to bit[25] and “1” to bit[31] exits the clearing process.

When the DBG_REG_RST_SRC register is used to enable reset recording, the following applies:

- Writing “1” to both bit[24] and bit[31] starts an enabling process.

Asclock domain crossing takes some time, perform polling on bit[16] until the Reset Reason Recorder block is ready (high).

3.5 Clocks

3.5.1 Functional Overview

Clock system is an important structure in the SoC. The clock sources can be derived from a series of internal or external high and low frequency oscillators. Different modules can use corresponding clock source according to their own requirements. The clock sources are divided into high-frequency clock and low-frequency clock ias below.

High-frequency clock sources:

- HFXO_32M (enabled by default)
- CPLL_192M (enabled by default)
- HFRC_192M (enabled by default)

Low-frequency clock sources:

- RNG_OSC (enabled by default)
- LFXO_32K
- LFRC_32K (enabled by default)

Figure 3-27 shows the basic clock structure of the device.

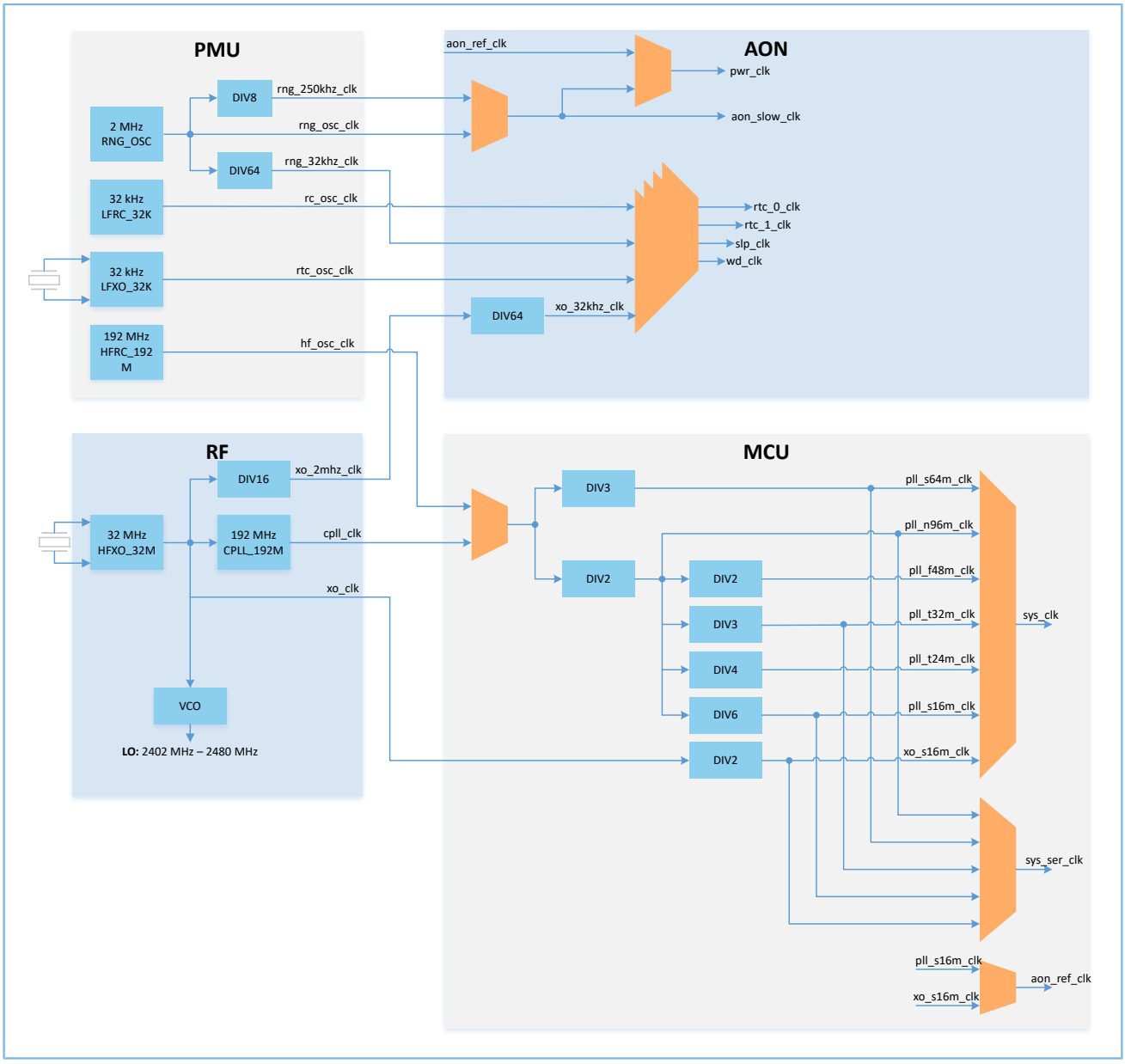

Figure 3-27 MCU clocks

MCU clocks are generated using a high-frequency clock (CPLL_192M) that requires a 32 MHz external crystal (HFXO_32M) or an internal high-frequency RC oscillator (HFRC_192M) when the wireless link is not used. During the MCU working stage, either CPLL_192M or HFRC_192M must be available. The external crystal is needed to meet the offset requirements of the Bluetooth LE specifications.

The system clock and system serial clock can be configured independently. In the SDK, the default value of the system serial clock is 96 MHz.

The AON Power State Controller clock comes from a 2 MHz ring oscillator (RNG_OSC) with a choice of divided by 250 kHz clock (rng_250khz_clk) or undivided 2 MHz clock (rng_osc_clk).

The clocks for the various slow AON timer modules can be generated from a choice of 4 sources: low-frequency RC oscillator (LFRC_32K), divided-down version of 2 MHz RNG_OSC, low-frequency crystal (LFXO_32K) based on 32

kHz crystal, and divided-down version of HFXO_32M clock (XO_32KHz_CLK). Due to the dependence between these modules and clock sources, once the clock source is selected for the module, this clock source should not be turned off. Since the clock source of the XO_32KHz_CLK is from HFXO_32M, the average power consumption of the system will increase, and the accuracy depends on the HFXO_32M in this situation.

A divided-down version of the HFXO_32M clock can be muxed out through one of the AON GPIO5 to provide an accurate 2 MHz clock output to an external peripheral.

For XO clock sources such as HFXO_32M and LFXO_32K, the device does not require external capacitors except the crystal itself. It is recommended to choose a crystal with low load capacitance (6 pF–8 pF). Low load capacitance will reduce startup time and current consumption. For specified parameters and recommended selection of external crystals, refer to *GR5525 Hardware Design Guidelines*.

To reduce power consumption, an integer clock supporting Phase-Locked Loop (CPLL_192M) is used to generate the main system clock to both RF and digital sections. Frequency dividers are used to provide slower frequencies for different blocks of the device. CPLL_192M is also used to tune the desired frequency hopping for each packet. The settling time of the CPLL_192M is designed to be within the time defined by the Bluetooth LE specifications to maintain the interframe space (IFS) timing.

Table 3-35 Clock sources with power management

Clock Source	Active Mode	Sleep Mode
HFXO_32M	ON (Conditionally OFF)	Conditionally OFF
CPLL_192M	ON (Conditionally OFF)	Normally OFF
HFRC_192M	ON (Conditionally OFF)	Normally OFF
RNG_OSC	ON	ON
LFXO_32K	ON (Conditionally OFF)	ON (Conditionally OFF)
LFRC_32K	ON (Conditionally OFF)	ON (Conditionally OFF)

The usage of each clock source in different power modes is described in Table 3-35. Developers can selectively turn off some clock sources in order to reduce the average power consumption of the system. Some clock sources are turned off conditionally, which can be found in the chapter description of each clock. It's worth noting that when the clock is turned on, it usually takes time for the crystal to start oscillating, and it also takes time for the output to fully stabilize. In order to prevent erroneous clocks, the SDK implements automatic calculation of the time required for different clocks to stabilize and delays until the clocks are stabilized before being fully activated.

For power optimization, a lot of clock gates (CG) are implemented.

The clock of the module can be controlled by the CGs. The CGs can be configured manually or controlled automatically by hardware, which improves the flexibility of power controlling. The clocks of each module can be forced off by controlling the *_force_off registers.

The module driver in the SDK has implemented that the corresponding clock is automatically turned on through CGs. Through the initialization API in the SDK, developers do not need to manually open the module clock.

The CG control management in sleep mode has been implemented in the SDK. Since the peripheral's control registers are not in the AON power domain, the configuration will be lost after entering sleep mode so that the critical register

configuration should been backed up before sleep. The clocks will be turned on through the CGs during the wake-up phase, and the critical register configuration will be restored automatically.

Developers is able to control the clock of each module through the CG register described in "Section 3.7 Registers".

3.5.2 High Frequency Clocks

3.5.2.1 HFXO_32M

3.5.2.1.1 Introduction

HFXO_32M is a crystal-based oscillator that connects to a 32 MHz external crystal. The startup time of HFXO_32M depends on the crystal type as well as the internal bias current. The bias current is programmable to reduce the power consumption when the HFXO_32M clock settles.

HFXO_32M is used in application scenarios that require high precision and temperature sensitivity, such as Bluetooth RF. During the processing of Bluetooth tasks, HFXO_32M should be turned on. The SoC can use xo16m_clk, which is generated by dividing HFXO_32M by two, as the system clock at the initial stage of power-on. After normal booting, it is able to switch to user-configured system clock.

3.5.2.1.2 Main Features

- Used to drive 32 MHz crystal
- Programmable on-chip capacitor array for frequency calibration
- Programmable bias current to optimize different crystals, supporting a wide range of equivalent series resistance (ESR) and equivalent series inductance (ESL)
- Fast setup mode disabled, can be enabled on demand
- Mapping divided HFXO_2M to pin

3.5.2.1.3 Enable and Disable

The prerequisite for HFXO_32M to be turned on is to wait for the DC-DC converter to be powered on.

When the system meets the following conditions, the HFXO_32M will be turned on.

- When the system meets one of the following conditions during the wakeup period, the hardware will automatically turn on HFXO_32M.
  - HFXO_32M is enabled when AON_START_CFG.XO_EN_PWR = 0x1.
  - CPLL_192M is enabled when AON_START_CFG.PLL_EN_PWR = 0x1.
  - HFXO_32M is selected as the system clock when MCU_CLK_CTRL.MCU_CLK_CTRL_SEL = 0x2.
  - CPLL_192M is selected as the system root working clock when AON_CLK.CAL_FST_CLK = 0x0.
- If one of the following registers is configured, HFXO_32M is forced to be turned on at any time.
  - MCU_CLK_CTRL.MCU_CLK_CTRL_SEL = 0x2

- AON_CLK.CAL_FST_CLK = 0x0

- If XO_PLL_SET.XO_SET= 1, HFXO_32M will be requested to be turned on.
- If the behavior of enabling CPLL_192M occurs, HFXO_32M will also be turned on.

If COMM_CORE_PWR_CTRL_HW_CFG.PLL_AUTO_ON = 1, both HFXO_32M and CPLL_192M will be automatically turned on when the Bluetooth LE Timer interrupt is triggered.

When the system meets the one of following conditions, the HFXO_32M will be turned off.

- Before the system enters sleep mode, both DC-DC converter is disabled when A_SLP_CFG.TRN_OFF_DCDC = 1.
- Auto HFXO_32M off function during sleep is enabled when A_SLP_CFG.TRN_OFF_XO = 1.
- If the bit XO_PLL_CLR.XO_CLR is set to 1, HFXO_32M will be requested to be turned off.

By default, the system uses hardware to control signals. In order to improve the flexibility, the SoC provides overriding registers to allow software to control signals. The state of HFXO_32M can be configured by the overriding registers.

- Set status to the RF_INTF_OVR_VAL_0.XO_EN bit.
- Enable override function by setting the RF_INTF_OVR_EN_0.XO_EN bit to 1.
- The status of HFXO_32M can be obtained by reading from the RF_INTF_OVR_RD_0.XO_EN bit.

3.5.2.1.4 Regulation

When HFXO_32M is running in steady operation state, it is possible to trim the frequency by modifying the tuning capacitance through the RF9[18:10] bits. Ideally, changing the RF9[18:10] bits by one LSB at one time will modify the frequency of the HFXO_32M. Sufficient time should elapse to wait for the HFXO_32M to stabilize before making frequency measurement.

The bias current of the HFXO_32M affects the power consumption. The bias current can be adjusted through the RF_XO_BIAS_VAL register after the DC-DC converter power on.

- When requesting to turn on HFXO_32M, the bias current of the HFXO_32M can be reduced by configuring the RF_XO_BIAS_VAL.BIAS_LO field.
- When requesting to turn off HFXO_32M, the bias current of the HFXO_32M can be increased by configuring the RF_XO_BIAS_VAL.BIAS_HI field.

Due to the deviation of the chip process, the most suitable bias current configuration needs to be obtained through actual test.

3.5.2.1.5 Electrical Specifications

Table 3-36 HFXO_32M crystal oscillator specifications

Parameter	Symbol	Min.	Typ.	Max.	Unit
HFXO_32M frequency	$HFXO_32M_{FREQ}$		32		MHz
Accuracy	$HFXO_32M_{ACC}$			100	ppm

Copyright © 2024 Shenzhen Goodix Technology Co., Ltd.

Parameter	Symbol	Min.	Typ.	Max.	Unit
Equivalent series resistance	$HFXO_32M_{ESR}$			80	Ω
Load capacitance	$HFXO_32M_{LCAP}$	6		8	pF
Startup time (without su)	$HFXO_32M_{STIM}$			1000	µs
Startup time (with su)	$HFXO_32M_{STIMSU}$			500	µs

3.5.2.2 CPLL_192M

3.5.2.2.1 Introduction

CPLL_192M is obtained by multiplying the HFXO_32M clock source by the phase-locked loop (CPLL) circuit. The multiplied clock is used to generate the main system clock for the RF and digital sections.

CPLL_192M is also used to tune frequency hopping to obtain the desired value for each packet. The settling time of CPLL_192M is designed to be within the time defined by the Bluetooth LE specifications to maintain the interframe space (IFS) timing.

3.5.2.2.2 Main Features

- Frequency locked at 192 MHz
- Low output jitter
- Lock detection module with flag bit
- Mapping divided clock to pin

3.5.2.2.3 Enable and Disable

The prerequisite for CPLL_192M to be turned on is to wait for HFXO_32M to be stable. When the system meets the following conditions, the CPLL_192M will be turned on.

- When the system meets one of the following conditions during the wakeup period, the CPLL_192M will be turned on.
  - CPLL_192M is enabled when AON_START_CFG.PLL_EN_PWR = 0x1
  - CPLL_192M is enabled when AON_CLK.CAL_FST_CLK = 0x0
- If the AON_CLK.CAL_FST_CLK bit is set to 0, CPLL_192M will be forced to be turned on at any time.
- If the COMM_CORE_PWR_CTRL_HW_CFG.PLL_AUTO_ON bit is set to 1, CPLL_192M will be automatically turned on when the Bluetooth LE Timer interrupt is triggerred.
- If the XO_PLL_SET.PLL_SET register is set to 1, CPLL_192M will be requested to be turned on.

When the system meets one of the following conditions, the CPLL_192M will be turned off.

- If the HFXO_32M is turned off, the CPLL_192M is turned off due to the loss of reference clock source.
- Auto CPLL_192M off function during sleep is enabled when A_SLP_CFG.TRN_OFF_PLL = 1
- If the XO_PLL_CLR.PLL_CLR bit is set to 1, CPLL_192M will be requested to be turned off.

By default, the system uses hardware to control signals. In order to improve the flexibility, the SoC provides overriding registers to allow software to control signals. The current state of CPLL_192M can be controlled by the overriding registers.

- Set status to the RF_INTF_OVR_VAL_0.CPLL_EN bit and the RF_INTF_OVR_VAL_0.RST_N bit.
- Enable override function by setting RF_INTF_OVR_EN_0.CPLL_EN = 1 and RF_INTF_OVR_EN_0.RST_N = 1.
- The status of CPLL_192M can be obtained by reading from the RF_INTF_OVR_RD_0.CPLL_EN bit and RF_INTF_OVR_RD_0.RST_N bit.

3.5.2.2.4 Regulation

The CPLL circuit is able to output other frequency by configuring the RF6 and RF7 registers. However, the SoC usually only uses the 192MHz frequency in application so that the CPLL will be locked at 192 MHz.

3.5.2.2.5 Electrical Specifications

Table 3-37 CPLL_192M specifications

Parameter	Symbol	Min.	Typ.	Max.	Unit
Output frequency	$CPLL_192M_{FREQ}$		192		MHz
Deviation across temperature	$CPLL_192M_{ACC}$	–500		500	ppm/°C

3.5.2.3 HFRC_192M

3.5.2.3.1 Introduction

The high-frequency ring oscillator (HFRC_192M) is an internal 192 MHz clock used for fast system start, providing system clock for MCU. HFRC_192M will be open during the initial stage of power-on by default. Due to the individual differences of the RC oscillator, the frequency of HFRC_192M may be different variable on different SoCs. Developers can adjust the frequency through specific registers to meet the clock frequency requirements of each module.

Due to the accuracy and temperature sensitivity, the high-frequency ring oscillator (HFRC_192M) is required to switch to HFXO_32M clock if Bluetooth LE connection is used. In applications where no high-precision clock is required, the high-frequency ring oscillator (HFRC_192M) can replace the high-frequency RC oscillator (CPLL_192M) in order to reduce the power consumption.

3.5.2.3.2 Main Features

- 192 MHz high frequency ring oscillator
- Low frequency-temperature coefficient with automatic adapt

- Low current consumption
- Fast startup
- Online calibration
- Mapping divided clock to a pin

3.5.2.3.3 Enable and Disable

When the system meets one of the following conditions, the HFRC_192M will be turned on.

- System needs to wait for DC-DC converter to complete power-on before HFRC_192M can be turned on during the wakeup period.
- If the XO_PLL_SET.HF_OSC_SET bit is set to 1, HFRC_192M will be requested to be turned on.

When the system meets one of the following conditions, the HFRC_192M will be turned off.

- Auto HFRC_192M off function during sleep is enabled when A_SLP_CFG.TRN_OFF_HF_OSC = 1
- On the premise that the system clock does not use HFRC_192M, the HFRC_192M can be turned off through the XO_PLL_CLR.HF_OSC_CLR bit.

By default, the system uses hardware to control signals. In order to improve the flexibility, the SoC provides overriding registers to allow software to control signals. The current state of HFRC_192M can be controlled by the overriding registers.

- Set status to the PMU_INTF_OVR_VAL_0.HF_OSC_EN bit.
- Enable override function by setting the PMU_INTF_OVR_EN_0.HF_OSC_EN bit to 1.
- The status of HFRC_192M can be obtained by reading from the PMU_INTF_OVR_RD0.HF_OSC_EN bit.

3.5.2.3.4 Regulation

Since the actual frequency of HFRC_192M will be variable due to process deviation, it is able to adjust the frequency through the FS_REG_0 and FS_REG_1 registers.

- FS_REG_0.FS_REG3 allows fine adjustment of the frequency of HFRC_192M.
- FS_REG_1.FS_REG4[3:0] allows coarse adjustment of the frequency of HFRC_192M.

Normally, the operation step is to adjust the frequency of HFRC_192M to the specified range through the coarse adjustment register first, and then adjust the frequency of the HFRC_192M through the fine adjustment register.

The SoC supports online calibration of the HFRC_192M through high precision crystal oscillator. For detailed description of the calibration, refer to "Section 3.5.5 Clock Calibration".

3.5.2.3.5 Electrical Specifications

Table 3-38 HFRC_192M specifications

Parameter	Symbol	Min.	Typ.	Max.	Unit
Output frequency	$HFRC_192M_{FREQ}$	183.3	192	196	MHz

Parameter	Symbol	Min.	Typ.	Max.	Unit
Deviation across temperature	$HFRC_192M_{ACC}$	–500	265	500	ppm/°C

3.5.3 Low Frequency Clocks

3.5.3.1 RNG_OSC

3.5.3.1.1 Introduction

The ring oscillator (RNG_OSC) clock is a low-frequency (2 MHz) and ultra-low power clock that is ON as long as the chip is enabled. This clock is used to clock the power sequencers at both the cold and warm boots of the SoC.

Due to high power consumption when the frequency is 2 MHz, in order to reduce the power consumption during sleep, the SoC will switch to rng_250khz_clock as the clock source during sleep. In active mode, it will switch back to 2 MHz frequency for faster response speed. If developers have no special requirements for the power consumption during sleep, the frequency during sleep can be kept at 2 MHz, which can improve the wake-up speed.

3.5.3.1.2 Main Features

- 2 MHz nominal frequency
- Frequency division to 250 kHz
- Low current consumption
- Mapping clock to a pin
- Online calibration

3.5.3.1.3 Enable and Disable

The RNG_OSC is mainly used for clocking power sequencers during boot process and clocking the AON domain. The RNG_OSC can be enabled or disabled through the MCU_CLK_CTRL.RNG_2MHZ_EN register. When the RNG_OSC is disabled, the AON domain needs other slow clock source. The slow clock source of the AON domain can be selected through the MCU_CLK_CTRL.SLOW_CLK_SEL register. Developer is able to switch the slow clock source of the AON domain to RNG in order to reduce the power consumption during sleep. However, the system wakeup and response speed will be reduced due to lower clock frequency.

By default, the system uses hardware to control signals. In order to improve the flexibility, the SoC provides overriding registers to allow software to control signals. The current state of RNG_OSC can be controlled by the overriding registers.

- Set status to the PMU_INTF_OVR_VAL_0. RNG_OSC_CLK_EN.
- Enable override function by setting PMU_INTF_OVR_EN_0. RNG_OSC_CLK_EN = 1.
- The status of RNG_OSC can be obtained by reading the PMU_INTF_OVR_RD0. RNG_OSC_CLK_EN bit.

3.5.3.1.4 Regulation

The RNG_OSC cannot be regulated.

The SoC supports mapping the RNG_OSC or the RNG_32K from the RNG_OSC on pin.

3.5.3.1.5 Electrical Specifications

Table 3-39 RNG_OSC specifications

Parameter	Symbol	Min.	Typ.	Max.	Unit
Output frequency	RNG_OSC_{FREQ}		2		MHz
Deviation across temperature	RNG_OSC_{ACC}	-500		500	ppm/°C

3.5.3.2 LFXO_32K

3.5.3.2.1 Introduction

LFXO_32K clock is a crystal-based accurate clock (32.768 kHz) that is more stable than the low-frequency RC oscillator (LFRC_32K) across voltage and temperature. This clock is used for accurate sleep and wakeup.

In some cost-sensitive scenarios, the external crystal can be removed. In this situation, the SoC will not have the clock source of LFXO_32K.

3.5.3.2.2 Main Features

- High precision clock
- Functional in sleep mode
- Mapping clock to a pin
- Online calibration

3.5.3.2.3 Enable and Disable

LFXO_32K can be enabled or disabled through the RF_REG_0[7] bit. The clock should not be used immediately after enabling. Sufficient time should elapse to wait for the LFXO_32K to stabilize.

3.5.3.2.4 Regulation

LFXO_32K can be trimmed by adjusting the tuning capacitor through the RC_RTC_REG_0[29:24] bits. Due to the parasitic capacitance in the circuit, the real frequency of LFXO_32K may not be the nominal frequency (32.768kHz). Usually the system does not need to adjust the LFXO_32K to the nominal frequency. The system only needs to obtain the current frequency of the LFXO_32K. The SoC supports online calibration of the LFXO_32K through high frequency and precision crystal oscillator. Through the calibration module, developers can obtain the real frequency of the current LFXO_32K clock. For detailed description of the calibration, refer to “Section 3.5.5 Clock Calibration”.

3.5.3.2.5 Electrical Specifications

Table 3-40 LFXO_32K specifications

Parameter	Symbol	Min.	Typ.	Max.	Unit
Output frequency	$LFXO_32K_{FREQ}$		32.768		kHz
Accuracy	$LFXO_32K_{ACC}$			100	ppm
Equivalent series resistance	$LFXO_32K_{ESR}$			100	kΩ
Load capacitance	$LFXO_32K_{LCAP}$	6		9	pF

3.5.3.3 LFRC_32K

3.5.3.3.1 Introduction

The low-frequency RC oscillator (LFRC_32K) is an internal 32 kHz clock used for generating various slow AON timer module clocks. Compared with LFXO_32K, LFRC_32K has lower precision.

In applications where no high-precision clock is required, the low-frequency RC oscillator (LFRC_32K) can replace the low-frequency crystal (LFXO_32K).

In sleep mode, LFRC_32K can be turned off unless it is used as the clock source for the AON timers.

3.5.3.3.2 Main Features

- 32 kHz low frequency RC oscillator
- Low frequency-temperature coefficient with automatic adaptation
- Low current consumption
- Fast startup
- Mapping clock to a pin
- Online calibration

3.5.3.3.3 Enable and Disable

LFRC_32K can be enabled or disabled through the RC_RTC_REG_0[15] bit. LFRC_32K has a fast startup time. It is not like LFXO_32K which needs to wait for stabilization after enabling. Since the RC oscillator is greatly affected by the environment, the actual frequency of LFRC_32K will fluctuate within a range.

3.5.3.3.4 Regulation

LFRC_32K can be trimmed by adjusting the RC_RTC_REG_0[10:8] bits. Usually the system does not need to adjust the LFRC_32K to the nominal frequency (32kHz). The system only needs to obtain the current frequency of the LFRC_32K. The SoC supports online calibration of the LFRC_32K through high frequency and precision crystal oscillator. Through the calibration module, developers can obtain the real frequency of the current LFRC_32K clock. For detailed description of the calibration, refer to “Section 3.5.5 Clock Calibration”.

3.5.3.3.5 Electrical Specifications

Table 3-41 LFRC_32K specifications

Parameter	Symbol	Min.	Typ.	Max.	Unit
Output frequency	$LFRC_32K_{FREQ}$	19	30.3	54	kHz
Deviation across temperature	$LFRC_32K_{ACC}$	–500	123	500	ppm/°C

3.5.4 Clock Gating

3.5.4.1 Architecture

Clock gating is a power management technique used in many digital designs for reducing dynamic power dissipation, by removing the clock signal when the circuit is not in use or ignores clock signal. Clock gating saves power by pruning the clock tree, at the cost of adding more logic to a circuit. When not being switched, the switching power consumption goes to zero, and only leakage currents are incurred.

In the SoC, a lot of clock gates (CG) are implemented, as shown in the figure below, in order to optimize power consumption. Clock gating supports mandatory control and dynamic control of the clock through software. Clock gates are divided according to functional categories, accurate to each system or peripheral module. Through the finely divided clock gates, developers can implement subtle digital circuit power control, thereby reducing the overall power consumption of the chip.

Copyright © 2024 Shenzhen Goodix Technology Co., Ltd.

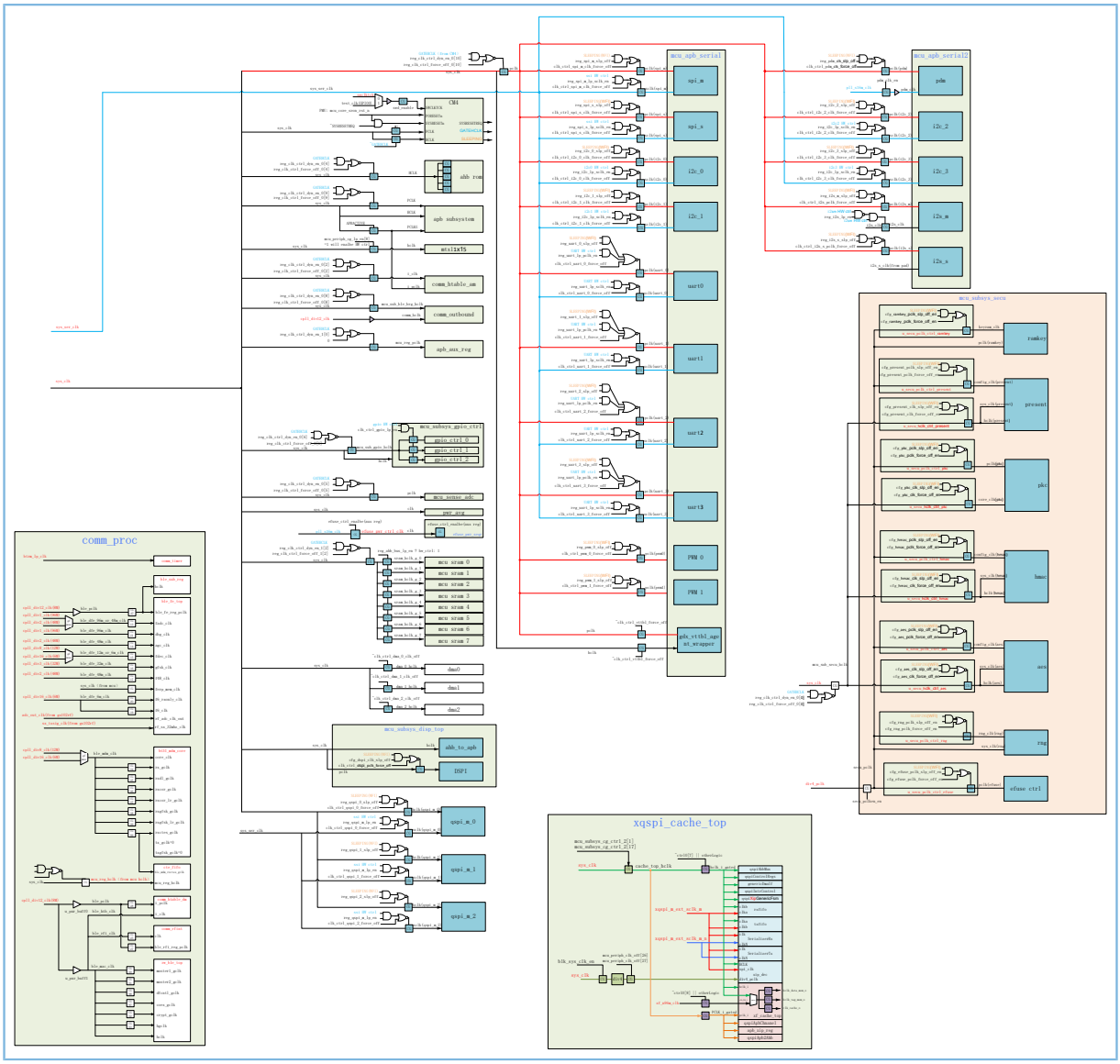

Figure 3-28 Clock gating of MCU subsystem

3.5.4.2 Programming

The clock of the system or peripheral module can be controlled by the CGs. The CGs can be configured into mandatory control mode or dynamic control mode, which improves the flexibility of power controlling.

The clocks of each module can be forced off by configuring the MCU_SUBSYS_CG_CTRL[1], MCU_SUBSYS_CG_CTRL[2], MCU_PERIPH_PCLK_OFF, SECU_CLK_CTRL and MCU_MISC_CLK registers.

Some of the modules support dynamic control during WFI/WFE. By configuring the MCU_SUBSYS_CG_CTRL[0], MCU_SUBSYS_CG_CTRL[2], MCU_PERIPH_CLK_SLP_OFF and SECU_CLK_CTRL registers, the clock of the module can be automatically turned off in CPU sleep mode. If the current clock gating of the module is forced off, the dynamic control will not take effect.

Developers is able to control the clock of each module through the CG register described in “Section 3.7 Registers”.

3.5.5 Clock Calibration

3.5.5.1 Introduction

The device provides a clock calibration module, which calibrates inaccurate clocks by using accurate clocks. One of these two clocks should be accurate, the other is to be calibrated. When a high-speed clock is used to calibrate a low-speed clock, the frequency difference between the two clocks should be large enough to obtain reasonable calibration results. When high-speed clock is calibrated with a low-speed clock, errors are inevitable in the result due to frequency limitations.

3.5.5.2 Main Features

- Slow clock calibration through HFXO_32M
- HFRC_192M calibration
- Custom calibration interval

3.5.5.3 Functional Description

3.5.5.3.1 Architecture

As shown in the figure below, the fast clock is used to sample the slow clock, generating a rise edge, which is used to enable the counter. By reading the result of sampling, the relationship of the two clocks can be calculated.

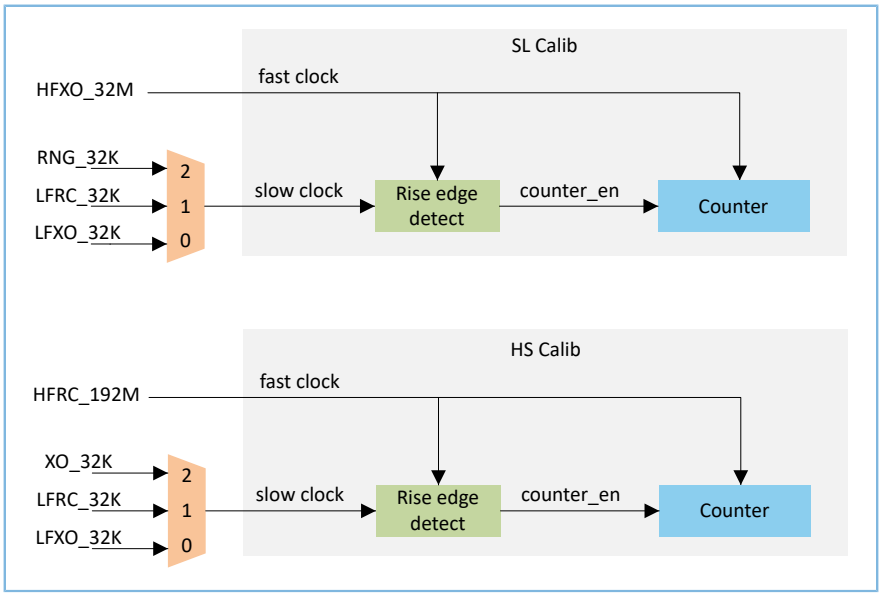

Figure 3-29 Clock calibration block diagram

The relationship of the fast clock (f_{fast}) and the slow clock (f_{slow}) can be calculated by the following equation:

$$\frac{f_{slow}}{N_{slow}} = \frac{f_{fast}}{N_{fast}}$$

The device provides two sets of registers for clock calibration, they both work the same way. The SL Calib registers are used to calibrate the kHz clock (low-frequency clock) such as LFXO_32K, LFRC_32K or RNG_32K from RNG_OSC with high-frequency clock (HFXO_32M). The HS Calib registers are used to calibrate the HFRC_192M clock with low-frequency clock such as LFXO_32K, LFRC_32K or XO 31.25 kHz from HFXO_32M.

Normally, it is recommended to use XO 31.25 kHz as slow clock source to calibrate HFRC_192M in order to get better calibration results. When using XO 31.25 kHz for calibration, HFXO_32M needs to be turned on. When using LFXO_32K or LFRC_32K as the low-speed clock source to calibrate HFRC_192M, LFXO_32K and LFRC_32K clock sources need to be calibrated by using the HFXO_32M clock sources first, so as to ensure that LFXO_32K or LFRC_32K as the reference clock source is valid.

3.5.5.3.2 Programming

The following operation flow applies to two sets of register:

1. Select a slow clock source. The SL_CLK_SEL register refers to the slow clock to be calibrated. The HS_CLK_SEL register refers to the slow clock used to be referenced.
2. Set the number of clock cycles of the slow clock to be calibrated (SL_CLK_CNT.COUNT or HS_CLK_CNT.COUNT = x, where final N = x + 1).
3. Enable the calibration module (SL_CLK_CTRL.EN or HS_CLK_CTRL.EN = 1).
4. Wait the module finishing counting (SL_CLK_STAT. DONE or HS_CLK_STAT.DONE = 1).
5. Read the number of reference fast clock cycles (SL_CLK_CNT0. VAL or HS_CLK_CNT0.VAL) for calculation.
   $calib_cal*f_{slow}=(cnt_val+1)*f_{fast}$
6. Disable the module (SL_CLK_CTRL.EN or HS_CLK_CTRL.EN = 0).

3.5.6 Clock Mapping to Pins

3.5.6.1 Introduction

Clocks can be mapped to external pins by configuration. This function allows developers to obtain the internal clock frequency of the chip. In specific occasions, developers can customize the clock calibration scheme through the clock drawn from the pins.

3.5.6.2 Main Features

- Mapping divided HFXO_32M to a pin
- Mapping divided CPLL_192M to a pin
- Mapping divided HFRC_192M to a pin
- Mapping LFXO_32K to a pin
- Mapping RNG_32K and RNG_OSC to a pin
- Mapping LFRC_32K to a pin

3.5.6.3 Programming

The low frequency clocks (such as RNG_32K, RNG_OSC, LFRC_32K, and LFXO_32K) generated from analog circuit can be mapped to the AON_GPIO_4 pin.

The mapping can be achieved with software by:

- Configure the field AON_PAD_CLK. CLK_OUT_SEL to select the mapped clock.
- Configure the AON_PAD_CLK. CLK_OUT_EN bit to enable the clock output via the AON_GPIO_4 pin.

Table 3-42 Mapping relationship on AON_GPIO4

Clock	AON_PAD_CLK.CLK_OUT_SEL
RNG_32K	0x0
RNG_OSC	0x1
LFRC_32K	0x2
LFXO_32K	0x3

Due to its high frequency, it is recommended to observe HFXO_32M by mapping HFXO_2M (generated by dividing HFXO_32M by 16) to the AON_GPIO_5 pin in an analog circuit. The mapping can be achieved with software by:

- Setting the XO_CTRL.2MHZ_ENA bit to enable HFXO_2M from RF (enabled by default)
- Setting the XO_CTRL.2MHZ_OUT bit to enable the clock output via the AON_GPIO_5 pin.

3.6 Startup Procedure

3.6.1 Introduction

The startup procedure will always be executed when a Power Reset, a Hardware Reset, a Software Reset or a wakeup event occurs. Before the main function is called, the device will do some initialization in order to bring the SoC to a more suitable state. The startup procedure can be divided into two parts. ROM Boot starts at the zero address, and its code is fixed in ROM and cannot be modified. Platform initialization starts at the beginning of the image and can be used to handle custom behaviors.

3.6.2 ROM Boot

ROM Boot is the initial Bootloader of the SoC. Its code is stored in ROM and cannot be modified by developers. Its main purpose is to boot the SoC and determine how the code is executed, so that the SoC can correctly jump to the application firmware.

As described in "Section 3.3.3 Power Management", there are two branches in the ROM Boot, which are the cold boot and the warm boot. The cold boot requires more time and resources than the warm boot because the entire system needs to be re-established.

The figure below shows the ROM boot process.

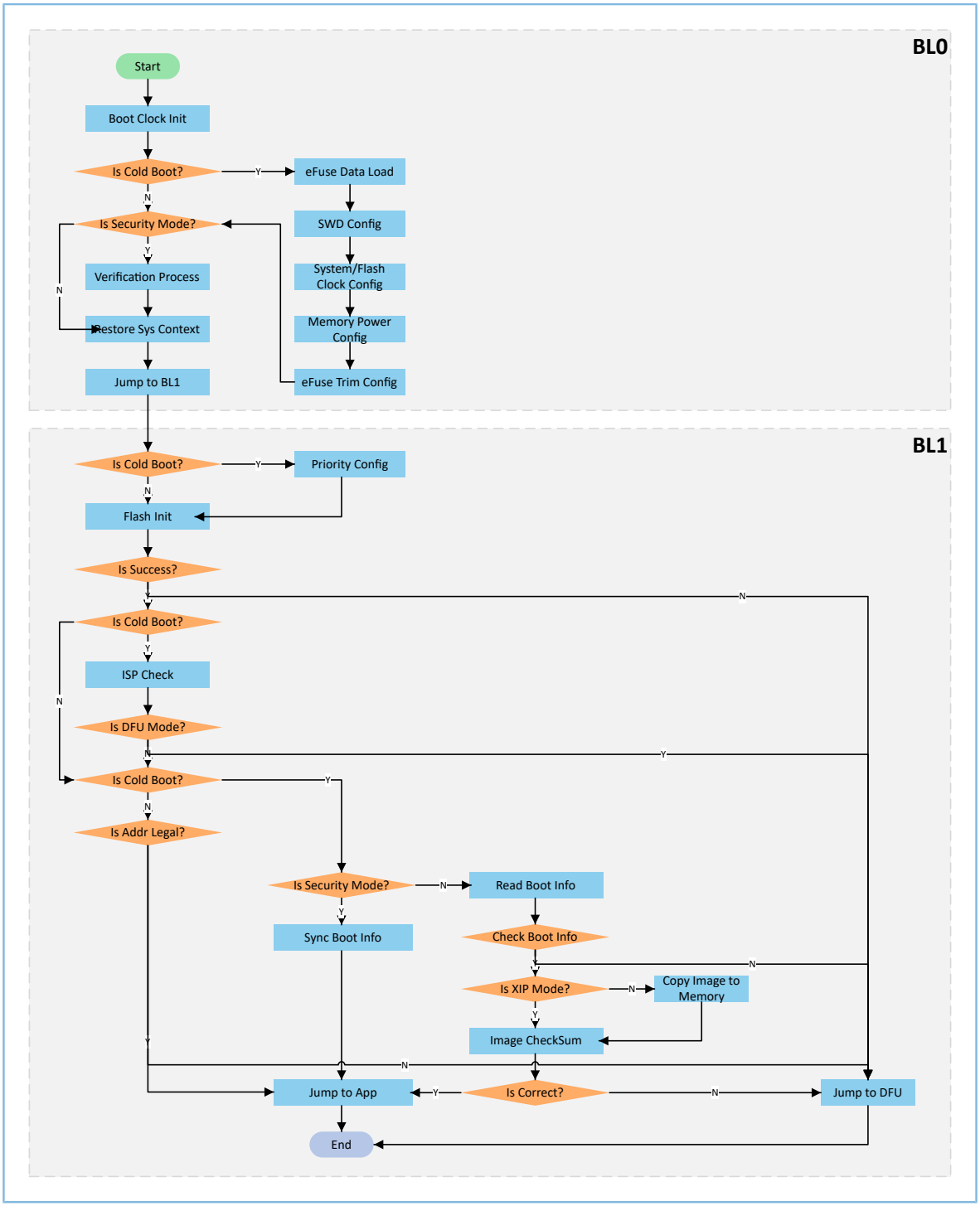

Figure 3-30 ROM boot flowchart

The ROM Boot process is composed of two separate phases:

- BL0
- BL1

If an exception occurs in ROM Boot, the SoC will enter the boot exception process, as described in "Section 3.6.2.3 Boot Exception".

3.6.2.1 BL0

The BL0 phase mainly serves for secure boot. The SoC needs to load the configuration information in eFuse to configure the system, such as whether to enter secure mode or whether SWD is disabled. The SoC also loads the trim information in eFuse.

If the SoC is set in secure mode, the verification process will be executed to perform a secure boot. The verification process includes Boot Info validation and firmware validation.

The Boot Info validation is mainly used to check and load Boot Info which stores the Flash system configuration.

The HMAC of the Boot Info is calculated through the HMAC hardware module which is based on the HMAC-SHA256 algorithm. The calculated HMAC will be compared with the HMAC which is stored at the end of the Boot Info. If the two HMAC values are equal, the Boot Info will be decrypted and the boot info will be updated to the main function subsequently.

The HMAC verification will be performed at most twice. The Boot Info in the main area is verified for the first time and the Boot Info in the backup area will be verified for the second time if the first verification fails. If both verifications fail, the SoC will eventually enter DFU mode.

The firmware verification mainly includes digital signature verification and key derivation.

In order to verify the public key, the HASH calculation which is based on the SHA256 HASH function is taken to the public key. By comparing the calculated public key HASH with the FW public key HASH which is stored in eFuse, the public key is able to be verified.

Since the public key cryptographic algorithm is slow, it is time-consuming to decrypt the entire message. Therefore, the HASH function can be used to calculate the digest of the firmware, and then the signature verification can be performed on the digest. Developers can choose the verification algorithm of the digital signature. Both RSASSA-PSS-2048 and ECDSA-P256 are supported by the SoC.

The encryption of the firmware adopts two kinds of encryption algorithm mixed calculation to ensure security. ECIES which is based on ECC with P-256 is used to calculate the key for the symmetric encryption of the firmware. The symmetric key will then be used by XIP-DEC for firmware on-the-fly decryption.

If there is an exception during the firmware verification process, the SoC will eventually enter DFU mode.

For more information about the secure mode, refer to *GR5xx Firmware Encryption Application Note*.

3.6.2.2 BL1

The BL1 phase eventually guides the operating mode of the code. In addition, if the SoC is not in secure mode, the image will be verified during the BL1 phase instead of the BL0 phase during cold boot.

There are three target jump modes in ROM Boot, which are XIP mode, Mirror mode, and DFU mode.

- In XIP mode, the ROM Boot will eventually jump to the Flash address and the code is executed in place.
- In Mirror mode, the ROM Boot will eventually jump to the SRAM address and the code is executed in SRAM.
- In DFU mode, the ROM Boot will eventually jump to the DFU service.

When the application firmware exists and no exception occurs, ROM Boot will usually jump to the XIP mode or the Mirror mode. Because the SWD interface will be lost during sleep mode, developers can hardly exit sleep mode through reset. Before ROM Boot jumps to the running address, there is a 1s delay to avoid the above situation. This delay can be turned on or off by controlling the parameters in Boot Info.

3.6.2.3 Boot Exception

Exceptions may occur during ROM Boot, causing the SoC to fail to jump to the application firmware. ROM Boot will record the current execution path to the SOFTWARE_REG0 register. When an exception occurs in ROM Boot, developers can read the SOFTWARE_REG0 register to analyze the cause of the exception. The recorded execution path is shown in the table below.

Table 3-43 Execution path in BL0

Path ID	Value
BL_INIT_SYSTEM	1<<0
BL_READ_EFUSE_INFO	1<<1
BL_READ_TEST_MODE	1<<2
BL_GENERATE_RANDOM_MASK	1<<3
BL_GENERATE_KEYRAM_KEY	1<<4
BL_CHECK_EFUSE_CRC32	1<<5
BL_TRIGGER_KEY_INTO_KEYRAM	1<<6
BL_INIT_FLASH	1<<7
BL_LOAD_INFO	1<<8
BL_INFO_BACKUP	1<<9
BL_FW_SIGNATURE	1<<10
BL_CHECK_RSA_PUBLIC_KEY_HASK	1<<11
BL_CHECK_RSA_SIGNATURE	1<<12
BL_CHECK_PRODUCT_ID	1<<13
BL_GENERATE_ECIES_KEY	1<<14
BL_MOVE_FW	1<<15
BL_JUMP_TO_APP	1<<16
BL_ENTER_UPGRADE	1<<31

Table 3-44 Execution path in BL1

Path ID	Value
BOOT_PATH_NONE	0x0
BOOT_PATH_JUMP_TO_APP	0x1
BOOT_PATH_FLASH_INIT	0x2

Path ID	Value
BOOT_PATH_FLASH_ERR_TO_DFU	0x3
BOOT_PATH_DBG_CHECK	0x4
BOOT_PATH_DBG_MODE_TO_DFU	0x5
BOOT_PATH_ISP_CHECK	0x6
BOOT_PATH_ISP_MODE_TO_DFU	0x7
BOOT_PATH_BOOTINFO_CHECK	0x8
BOOT_PATH_BOOT_ERR_TO_DFU	0x9
BOOT_PATH_IMAGE_CHECK	0xA
BOOT_PATH_IMAGE_ERR_TO_DFU	0xB
BOOT_PATH_WARM_BOOT_TO_APP	0xC
BOOT_PATH_WARM_BOOT_TO_DFU	0xD

When an exception occurs, Boot ROM will eventually jump into the DFU mode. For detailed information of the DFU mode, refer to *GR5xx DFU Development Application Note*.

3.7 Registers

3.7.1 Always-on Control

3.7.1.1 MCU_CLK_CTRL

- **Name:** MCU_CLK_CTRL Register
- **Description:** MCU_CLK_CTRL Register
- **Base Address:** 0x4000A000
- **Offset:** 0x0
- **Reset Value:** 0x42040202

Table 3-45 MCU_CLK_CTRL Register

Bits	Field Name	RW	Reset	Description
31	RSVD	R	0x0	Reserved bit
30	RNG_2MHZ_EN	RW	0x1	Enable/Disable RNG OSC2 MHz CLK in PMU. Will force enable when SLOW_CLK_SEL = 1.
29	RSVD	R	0x0	Reserved bit
28:27	SLOW_CLK_SEL	RW	0x0	Select the source of aon_slow CLK. Please allow about 10 µs for switching. **Value:** • 0x0: 250K RNG OSC CLK

Bits	Field Name	RW	Reset	Description
				• 0x1: 32K RNG OSC CLK
26:24	SYS_CLK_RD	R	0x2	Current MCU_CLK_CTRL_SEL value
23:19	RSVD	R	0x0	Reserved bits
18:16	SER_CLK_SEL	RW	0x4	Select the clock for serial modules (UART, I2C, SPI_M, and SPI_S). **Value:** • 0x0: 96M_clk PLL/HF_OSC 96 MHz CLK • 0x1: 64M_clk PLL/HF_OSC 64 MHz CLK • 0x2: 32M_clk PLL/HF_OSC 32 MHz CLK • 0x3: 16M_clk PLL/HF_OSC 16 MHz CLK • 0x4: XO_16M_clk XO 16 MHz CLK
15:13	RSVD	R	0x0	Reserved bits
12	WKUP_CLK_EN	RW	0x0	Apply WKUP_CLK_SEL to MCU_CLK_CTRL_SEL on wakeup.
11	RSVD	R	0x0	Reserved bit
10:8	WKUP_CLK_SEL	RW	0x2	Select system clock on wakeup. **Value:** • 0x0: 96M_clk PLL/HF_OSC 96 MHz CLK • 0x1: 64M_clk PLL/HF_OSC 64 MHz CLK • 0x2: XO_16M_clk XO 16 MHz CLK • 0x3: 48M_clk PLL/HF_OSC 48 MHz CLK • 0x4: 24M_clk PLL/HF_OSC 24 MHz CLK • 0x5: 16M_clk PLL/HF_OSC 16 MHz CLK • 0x6: 32M_clk PLL/HF_OSC 32 MHz CLK
7:3	RSVD	R	0x0	Reserved bits
2:0	MCU_CLK_CTRL_SEL	RW	0x2	Select the system clock for MCU subsystem. **Value:** • 0x0: 96M_clk PLL/HF_OSC 96 MHz CLK • 0x1: 64M_clk PLL/HF_OSC 64 MHz CLK • 0x2: XO_16M_clk XO 16 MHz CLK • 0x3: 48M_clk PLL/HF_OSC 48 MHz CLK • 0x4: 24M_clk PLL/HF_OSC 24 MHz CLK • 0x5: 16M_clk PLL/HF_OSC 16 MHz CLK • 0x6: 32M_clk PLL/HF_OSC 32 MHz CLK

3.7.1.2 MCU_MISC_CFG

- **Name:** MCU_MISC_CFG Register
- **Description:** MCU_MISC_CFG Register
- **Base Address:** 0x4000A000
- **Offset:** 0x4
- **Reset Value:** 0x00000000

Table 3-46 MCU_MISC_CFG Register

Bits	Field Name	RW	Reset	Description
31:9	RSVD	R	0x0	Reserved bits
8	SWD_ENABLE	RW	0x0	Enable SWD debugging.
7:0	RSVD	R	0x0	Reserved bits

3.7.1.3 XO_CTRL

- **Name:** XO_CTRL Register
- **Description:** XO_CTRL Register
- **Base Address:** 0x4000A000
- **Offset:** 0x8
- **Reset Value:** 0x00000000

Table 3-47 XO_CTRL Register

Bits	Field Name	RW	Reset	Description
31:25	RSVD	R	0x0	Reserved bits
24	BYP	RW	0x0	RF control register. Keep the default setting in test mode. **Value:** • 0x0: Use the analog XO module. • 0x1: Bypass the analog XO module. If the system uses XO or CPLL clock, an external oscillator shall be used.
23:9	RSVD	R	0x0	Reserved bits
8	2MHZ_ENA	RW	0x0	Enable XO 2 MHz from RF.
7:1	RSVD	R	0x0	Reserved bits
0	2MHZ_OUT	RW	0x0	**Value:** • 0x0: normal aon_gpio[5] • 0x1: aon_gpio[5] becomes xo_2mhz output.

3.7.1.4 FLASH_CACHE_CTRL0

- **Name:** FLASH_CACHE_CTRL0 Register
- **Description:** FLASH_CACHE_CTRL0 Register
- **Base Address:** 0x4000A000
- **Offset:** 0xC
- **Reset Value:** 0x00000014

Table 3-48 FLASH_CACHE_CTRL0 Register

Bits	Field Name	RW	Reset	Description
31:9	RSVD	R	0x0	Reserved bits
8	XF_TAG_RET	RW	0x0	This signal controls and disables flushing of the tag memory on power-on: **Value:** • 0x0: The tag memory will be flushed when mcu_core is powered on. • 0x1: Disable flushing the tag memory on mcu_core power-on.
7:5	RSVD	R	0x0	Reserved bits
4	XF_XO_DIV1	RW	0x1	Always need to be set to 1.
3	RSVD	R	0x0	Reserved bit
2:0	XF_SCK_CLK_SEL	RW	0x4	XQSPI clock **Value:** • 0x0: 64 MHz • 0x1: 48 MHz • 0x2: 32 MHz • 0x3: 24 MHz • 0x4: 16 MHz **Note:** 5 is forbidden! If this register is configured as 5, the clock will stop output.

3.7.1.5 FLASH_PSRAM_PAD_PWR

- **Name:** FLASH_PSRAM_PAD_PWR Register
- **Description:** FLASH_PSRAM_PAD_PWR Register

- **Base Address:** 0x4000A000
- **Offset:** 0x10
- **Reset Value:** 0x00000000

Table 3-49 FLASH_PSRAM_PAD_PWR Register

Bits	Field Name	RW	Reset	Description
31:1	RSVD	R	0x0	Reserved bits
0	PAD_EN	RW	0x0	eflash pad enable. Enable to provide POWER to External Flash

3.7.1.6 DIGIO_FST_CLK

- **Name:** DIGIO_FST_CLK Register
- **Description:** DIGIO_FST_CLK Register
- **Base Address:** 0x4000A000
- **Offset:** 0x14
- **Reset Value:** 0x00000000

Table 3-50 DIGIO_FST_CLK Register

Bits	Field Name	RW	Reset	Description
31:9	RSVD	R	0x0	Reserved bits
8	SEL	RW	0x0	Clock for digital IO LDO **Value:** • 0x0: 8 MHz • 0x1: 16 MHz
7:1	RSVD	R	0x0	Reserved bits
0	EN	RW	0x0	Enable digital IO LDO fast CLK.

3.7.1.7 AON_CLK

- **Name:** AON_CLK Register
- **Description:** AON_CLK Register
- **Base Address:** 0x4000A000
- **Offset:** 0x18
- **Reset Value:** 0x00000000

Table 3-51 AON_CLK Register

Bits	Field Name	RW	Reset	Description
31:9	RSVD	R	0x0	Reserved bits
8	WAKUP_FAST_CLK_SEL	RW	0x0	Select the source of the 192 MHz clock: **Value:** • 0x0: cpll_clk • 0x1: hf_osc_clk
7:5	RSVD	R	0x0	Reserved bits
4	WAKUP_CLK_EN	RW	0x0	Apply WAKUP_FAST_CLK_SEL value to CAL_FST_CLK on wakeup.
3:1	RSVD	R	0x0	Reserved bits
0	CAL_FST_CLK	RW	0x0	Select the source of the 192 MHz clock. **Value:** • 0x0: cpll_clk • 0x1: hf_osc_clk

3.7.1.8 TPP_ANA

- **Name:** TPP_ANA Register
- **Description:** TPP_ANA Register
- **Base Address:** 0x4000A000
- **Offset:** 0x20
- **Reset Value:** 0x00000101

Table 3-52 TPP_ANA Register

Bits	Field Name	RW	Reset	Description
31:9	RSVD	R	0x0	Reserved bits
8	EN_N_RD	R	0x1	Actual value of EN_N
7:1	RSVD	R	0x0	Reserved bits
0	EN_N	RW	0x1	This register only gets reset through POR.

3.7.1.9 AON_PWR_SAVING

- **Name:** AON_PWR_SAVING Register
- **Description:** AON_PWR_SAVING Register
- **Base Address:** 0x4000A000

- **Offset:** 0x24
- **Reset Value:** 0x00000000

Table 3-53 AON_PWR_SAVING Register

Bits	Field Name	RW	Reset	Description
31:1	RSVD	R	0x0	Reserved bits
0	EN	RW	0x0	Set it to 1 to save more AON power.

3.7.1.10 MCU_WAKEUP_CTRL

- **Name:** MCU_WAKEUP_CTRL Register
- **Description:** MCU_WAKEUP_CTRL Register
- **Base Address:** 0x4000A000
- **Offset:** 0x80
- **Reset Value**: 0x0000FFFF
- **Base Address**:
- **Offset**: 0x00

Note:

No IRQ is generated if the corresponding source is disabled.

Table 3-54 MCU_WAKEUP_CTRL Register

Bits	Field Name	RW	Reset	Description
31:16	RSVD	R	0x0	Reserved bits
15	RSVD	R	0x1	Reserved bits
14	RSVD	R	0x1	Reserved bits
13	RSVD	R	0x1	Reserved bits
12	RSVD	R	0x1	Reserved bits
11	COMP_FALL	RW	0x1	Select COMP_FALL IRQ as the wakeup event. **Value:** • 0x0: Disable IRQ. • 0x1: Enable IRQ.
10	BLE_IRQ	RW	0x1	Select BLE_IRQ IRQ as the wakeup event. **Value:** • 0x0: Disable IRQ. • 0x1: Enable IRQ.

Bits	Field Name	RW	Reset	Description
9	RTC1	RW	0x1	Select RTC1 IRQ as the wakeup event. **Value:** • 0x0: Disable IRQ. • 0x1: Enable IRQ.
8	RSVD	R	0x1	Reserved bits
7	RSVD	R	0x1	Reserved bits
6	AON_WDT	RW	0x1	Select AON_WDT IRQ as the wakeup event. **Value:** • 0x0: Disable IRQ. • 0x1: Enable IRQ.
5	COMP_RISE	RW	0x1	Select COMP_RISE IRQ as the wakeup event. **Value:** • 0x0: Disable IRQ. • 0x1: Enable IRQ.
4	PMU_BOD	RW	0x1	Select PMU_BOD IRQ as the wakeup event. **Value:** • 0x0: Disable IRQ. • 0x1: Enable IRQ.
3	RTC0	RW	0x1	Select RTC0 IRQ as the wakeup event. **Value:** • 0x0: Disable IRQ. • 0x1: Enable IRQ.
2	SMS_OSC	RW	0x1	Select SMS_OSC IRQ as the wakeup event. **Value:** • 0x0: Disable IRQ. • 0x1: Enable IRQ.
1	EXT	RW	0x1	Select EXT IRQ as the wakeup event. **Value:** • 0x0: Disable IRQ. • 0x1: Enable IRQ.
0	SLP_TIMER	RW	0x1	Select SLP_TIMER IRQ as the wakeup event. **Value:** • 0x0: Disable IRQ.

Bits	Field Name	RW	Reset	Description
				• 0x1: Enable IRQ.

3.7.1.11 AON_SLP_EVENT

- **Name:** AON_SLP_EVENT Register
- **Description:** AON_SLP_EVENT Register
- **Base Address**: 0x4000A000
- **Offset**: 0x84
- **Reset Value**: 0x00000000

Table 3-55 AON_SLP_EVENT Register

Bits	Field Name	RW	Reset	Description
31:12	RSVD	R	0x0	Reserved bits
11	COMP_FALL	WC	0x0	COMP_FALL event that happened during sleeping or in functional mode. Write 0 to clear.
10	BLE_IRQ	WC	0x0	BLE_IRQ event that happened during sleeping or in functional mode. Write 0 to clear.
9	RTC1	WC	0x0	RTC1 event that happened during sleeping or in functional mode. Write 0 to clear.
8:7	RSVD	R	0x0	Reserved bits
6	AON_WDT	WC	0x0	AON_WDT event that happened during sleeping or in functional mode. Write 0 to clear.
5	COMP_RISE	WC	0x0	COMP_RISE event that happened during sleeping or in functional mode. Write 0 to clear.
4	PMU_BOD	WC	0x0	PMU_BOD event that happened during sleeping or in functional mode. Write 0 to clear.
3	RTC0	WC	0x0	RTC0 event that happened during sleeping or in functional mode. Write 0 to clear.
2	SMS_OSC	WC	0x0	SMS_OSC event that happened during sleeping or in functional mode. Write 0 to clear.
1	EXT	WC	0x0	EXT event that happened during sleeping or in functional mode. Write 0 to clear.
0	SLP_TIMER	WC	0x0	SLP_TIMER event that happened during sleeping or in functional mode. Write 0 to clear.

3.7.1.12 AON_SLP_EVENT_RAW

- **Name:** AON_SLP_EVENT_RAW Register
- **Description:** AON_SLP_EVENT_RAW Register
- **Base Address:** 0x4000A000
- **Offset:** 0x8C
- **Reset Value:** 0x00000000

Table 3-56 AON_SLP_EVENT_RAW Register

Bits	Field Name	RW	Reset	Description
31:11	RSVD	R	0x0	Reserved bits
10	BLE_IRQ	R	0x0	AON sleep event status before masking (MCU_WAKEUP_CTRL). The bit order matches the AON_SLP_EVT register.
9	RTC1	R	0x0	AON Sleep event status before masking (MCU_WAKEUP_CTRL). The bit order matches the AON_SLP_EVT register.
8:7	RSVD	R	0x0	Reserved bits
6	AON_WDT	R	0x0	AON Sleep event status before masking (MCU_WAKEUP_CTRL). The bit order matches the AON_SLP_EVT register.
5	PMU_MSIO	R	0x0	AON Sleep event status before masking (MCU_WAKEUP_CTRL). The bit order matches the AON_SLP_EVT register.
4	PMU_BOD	R	0x0	AON Sleep event status before masking (MCU_WAKEUP_CTRL). The bit order matches the AON_SLP_EVT register.
3	RTC0	R	0x0	AON Sleep event status before masking (MCU_WAKEUP_CTRL). The bit order matches the AON_SLP_EVT register.
2	SMS_OSC	R	0x0	AON Sleep event status before masking (MCU_WAKEUP_CTRL). The bit order matches the AON_SLP_EVT register.
1	EXT	R	0x0	AON Sleep event status before masking (MCU_WAKEUP_CTRL). The bit order matches the AON_SLP_EVT register.

Bits	Field Name	RW	Reset	Description
0	SLP_TIMER	R	0x0	AON Sleep event status before masking (MCU_WAKEUP_CTRL). The bit order matches the AON_SLP_EVT register.

3.7.1.13 AON_IRQ

- **Name:** AON_IRQ Register
- **Description:** AON_IRQ Register
- **Base Address:** 0x4000A000
- **Offset:** 0x98
- **Reset Value:** 0x00000000

Table 3-57 AON_IRQ Register

Bits	Field Name	RW	Reset	Description
31:17	RSVD	R	0x0	Reserved bits
16	BLE_PWR_DN	WC	0x0	BLE_PWR_DN IRQ status. Write 0 to clear.
15:13	RSVD	R	0x0	Reserved bits
12	AONPLL_CHG	WC	0x0	AONPLL_CHG IRQ status. Write 0 to clear.
11:8	RSVD	R	0x0	Reserved bits
7	AONWDT_CFG_FAIL	WC	0x0	AONWDT_CFG_FAIL IRQ status. Write 0 to clear.
6	CLDR_CFG_FAIL	WC	0x0	CLDR_CFG_FAIL IRQ status. Write 0 to clear.
5	SLP_TIMER_CFG_FAIL	WC	0x0	SLP_TIMER_CFG_FAIL IRQ status. Write 0 to clear.
4	AON_SR_SLP_FAIL	WC	0x0	AON_SR_SLP_FAIL IRQ status. Write 0 to clear.
3	AON_DP_SLP_FAIL	WC	0x0	AON_DP_SLP_FAIL IRQ status. Write 0 to clear.
2	PMU_BOD	WC	0x0	PMU_BOD IRQ status. Write 0 to clear.
1	BLE_PWR_ON	WC	0x0	BLE_PWR_ON IRQ status. Write 0 to clear.
0	AON_WDT	WC	0x0	AON_WDT IRQ status. Write 0 to clear.

3.7.1.14 AON_IRQ_EN

- **Name:** AON_IRQ_EN Register
- **Description:** AON_IRQ_EN Register
- **Base Address:** 0x4000A000
- **Offset:** 0x9C
- **Reset Value:** 0x0003FFFF

Table 3-58 AON_IRQ_EN Register

Bits	Field Name	RW	Reset	Description
31:17	RSVD	R	0x0	Reserved bits
16	BLE_PWR_DN	RW	0x1	Enable BLE_PWR_DN status. **Value:** • 0x0: Disable IRQ. • 0x1: Enable IRQ.
15:13	RSVD	R	0x1	Reserved bits
12	AONPLL_CHG	RW	0x1	Enable AONPLL_CHG status. **Value:** • 0x0: Disable IRQ. • 0x1: Enable IRQ.
11:8	RSVD	R	0x1	Reserved bits
7	AONWDT_CFG_FAIL	RW	0x1	Enable AONWDT_CFG_FAIL status. **Value:** • 0x0: Disable IRQ. • 0x1: Enable IRQ.
6	RTC_CFG_FAIL	RW	0x1	Enable RTC_CFG_FAIL status. **Value:** • 0x0: Disable IRQ. • 0x1: Enable IRQ.
5	SLP_TIMER_CFG_FAIL	RW	0x1	Enable SLP_TIMER_CFG_FAIL IRQ. **Value:** • 0x0: Disable IRQ. • 0x1: Enable IRQ.
4	AON_SR_SLP_FAIL	RW	0x1	Enable AON_SR_SLP_FAIL IRQ. **Value:** • 0x0: Disable IRQ. • 0x1: Enable IRQ.
3	AON_DP_SLP_FAIL	RW	0x1	Enable AON_DP_SLP_FAIL IRQ. **Value:** • 0x0: Disable IRQ. • 0x1: Enable IRQ.
2	PMU_BOD	RW	0x1	Enable PMU_BOD IRQ. **Value:**

Bits	Field Name	RW	Reset	Description
				• 0x0: Disable IRQ. • 0x1: Enable IRQ.
1	BLE_PWR_ON	RW	0x1	Enable Bluetooth LE power-on done IRQ. **Value:** • 0x0: Disable IRQ. • 0x1: Enable IRQ.
0	AON_WDT	RW	0x1	Enable AON_WDT IRQ. **Value:** • 0x0: Disable IRQ. • 0x1: Enable IRQ.

3.7.1.15 AON_DBG_CTRL

- **Name:** AON_DBG_CTRL Register
- **Description:** AON_DBG_CTRL Register
- **Base Address:** 0x4000A000
- **Offset:** 0xA8
- **Reset Value:** 0x00000007

Table 3-59 AON_DBG_CTRL Register

Bits	Field Name	RW	Reset	Description
31:17	RSVD	R	0x0	Reserved bits
16	DBG_SLP	RW	0x0	Prevent sleep when JLINK is attached.
15:3	RSVD	R	0x0	Reserved bits
2	AON_WDT	RW	0x1	Set to stop the watchdog timer when MCU halts.
1	RTC	RW	0x1	Set to stop the calendar timer when MCU halts.
0	SLP_TIMER	RW	0x1	Set to stop the sleep timer when MCU halts.

3.7.1.16 MEM_MARGIN

- **Name:** MEM_MARGIN Register
- **Description:** MEM_MARGIN Register
- **Base Address:** 0x4000A000
- **Offset:** 0x1C0
- **Reset Value:** 0x0014A0A5

Table 3-60 MEM_MARGIN Register

Bits	Field Name	RW	Reset	Description
31:26	RSVD	R	0x0	Reserved bits
25:23	NON_CRITICAL_MEM_WPULSE	RW	0x0	Control of non-critical memory margin adjustment
22:21	NON_CRITICAL_MEM_RA	RW	0x0	Control of non-critical memory margin adjustment
20:18	NON_CRITICAL_MEM_WA	RW	0x5	Control of non-critical memory margin adjustment
17:14	NON_CRITICAL_MEM_RM	RW	0x2	Read margin of non-critical memory adjustment
13	NON_CRITICAL_MEM_RME	RW	0x1	Read margin enable of non-critical memory adjustment
12:10	CRITICAL_MEM_WPULSE	RW	0x0	Control of critical memory margin adjustment
9:8	CRITICAL_MEM_RA	RW	0x0	Control of critical memory margin adjustment
7:5	CRITICAL_MEM_WA	RW	0x5	Control of critical memory margin adjustment
4:1	CRITICAL_MEM_RM	RW	0x2	Read margin of critical memory adjustment
0	CRITICAL_MEM_RME	RW	0x1	Read margin enable of critical memory adjustment

3.7.1.17 EXT_WAKEUP_CTRL0

- **Name:** EXT_WAKEUP_CTRL0 Register
- **Description:** EXT_WAKEUP_CTRL0 Register
- **Base Address:** 0x4000A000
- **Offset:** 0x200
- **Reset Value:** 0x00000000

Table 3-61 EXT_WAKEUP_CTRL0 Register

Bits	Field Name	RW	Reset	Description
31:16	RSVD	R	0x0	Reserved bits
15:8	INVERT	RW	0x0	Invert external wakeup. • bit7: aon_gpio7 • bit6: aon_gpio6 • bit5: aon_gpio5 • bit4: aon_gpio4 • bit3: aon_gpio3 • bit2: aon_gpio2 • bit1: aon_gpio1 • bit0: aon_gpio0 **Value:**

Bits	Field Name	RW	Reset	Description
				• 0x0: Do not invert • 0x1: Invert
7:0	SRC_EN	RW	0x0	Enable external wakeup source. • bit7: aon_gpio7 • bit6: aon_gpio6 • bit5: aon_gpio5 • bit4: aon_gpio4 • bit3: aon_gpio3 • bit2: aon_gpio2 • bit1: aon_gpio1 • bit0: aon_gpio0 **Value:** • 0x0: Disable • 0x1: Enable

3.7.1.18 EXT_WAKEUP_CTRL1

- **Name:** EXT_WAKEUP_CTRL1 Register
- **Description:** EXT_WAKEUP_CTRL1 Register
- **Base Address:** 0x4000A000
- **Offset:** 0x204
- **Reset Value:** 0x00000000

Table 3-62 EXT_WAKEUP_CTRL1 Register

Bits	Field Name	RW	Reset	Description
31:24	RSVD	R	0x0	Reserved bits
23:16	EDGE_BOTH	RW	0x0	Valid when EDGE_EN = 1 • bit7: aon_gpio7 • bit6: aon_gpio6 • bit5: aon_gpio5 • bit4: aon_gpio4 • bit3: aon_gpio3 • bit2: aon_gpio2 • bit1: aon_gpio1

Bits	Field Name	RW	Reset	Description
				• bit0: aon_gpio0 **Value:** • 0x0: single edge • 0x1: both edges
15:8	EDGE_TYPE	RW	0x0	Valid when EDGE_EN = 1 & EDGE_BOTH = 0 • bit7: aon_gpio7 • bit6: aon_gpio6 • bit5: aon_gpio5 • bit4: aon_gpio4 • bit3: aon_gpio3 • bit2: aon_gpio2 • bit1: aon_gpio1 • bit0: aon_gpio0 **Value:** • 0x0: rising edge • 0x1: falling edge
7:0	EDGE_EN	RW	0x0	Enable edge detecting. • bit7: aon_gpio7 • bit6: aon_gpio6 • bit5: aon_gpio5 • bit4: aon_gpio4 • bit3: aon_gpio3 • bit2: aon_gpio2 • bit1: aon_gpio1 • bit0: aon_gpio0 **Value:** • 0x0: level signal • 0x1: Detect the edge.

3.7.1.19 EXT_WAKEUP_STAT

- **Name:** EXT_WAKEUP_STAT Register
- **Description:** EXT_WAKEUP_STAT Register

- **Base Address:** 0x4000A000
- **Offset:** 0x208
- **Reset Value:** 0x00000000

Table 3-63 EXT_WAKEUP_STAT Register

Bits	Field Name	RW	Reset	Description
31:8	RSVD	R	0x0	Reserved bits
7:0	STAT	WC	0x0	External wakeup status (after masking). Write 0 to clear the wakeup status (both level and pulse). • bit7: aon_gpio7 • bit6: aon_gpio6 • bit5: aon_gpio5 • bit4: aon_gpio4 • bit3: aon_gpio3 • bit2: aon_gpio2 • bit1: aon_gpio1 • bit0: aon_gpio0 **Value:** • 0x0: not triggered • 0x1: triggered

3.7.1.20 COMM_CTRL

- **Name:** COMM_CTRL Register
- **Description:** COMM_CTRL Register
- **Base Address:** 0x4000A000
- **Offset:** 0x280
- **Reset Value:** 0x00020000

Table 3-64 COMM_CTRL Register

Bits	Field Name	RW	Reset	Description
31:18	RSVD	R	0x0	Reserved bits
17:16	TIMER_CLK_SEL	RW	0x2	Select Comm timer clock. **Value:** • 0x0: rtc_osc_clk

Bits	Field Name	RW	Reset	Description
				• 0x1: rng_2_osc_clk (rc32k) • 0x2: rng_osc_clk
15:10	RSVD	R	0x0	Reserved bits
9	DEEPSLCNTL_EXTWKUPDSB	RW	0x0	Comm timer register
8	DEEPSLCNTL_SOFT_WAKEUP_REC	RW	0x0	Comm timer register
7	RSVD	R	0x0	Reserved bit
6	DEEPSLCNTL_DEEP_SLEEP_ON	RW	0x0	Comm timer register
5	DEEPSLCNTL_RADIO_SLEEP_EN	RW	0x0	Comm timer register
4	DEEPSLCNTL_OSC_SLEEP_EN	RW	0x0	Comm timer register
3	RSVD	R	0x0	Reserved bit
2	DEEPSLCNTL_DEEP_SLEEP_STAT	R	0x0	Comm timer register
1:0	RSVD	R	0x0	Reserved bits

3.7.1.21 BLE_MISC

- **Name:** BLE_MISC Register
- **Description:** BLE_MISC Register
- **Base Address:** 0x4000A000
- **Offset:** 0x284
- **Reset Value:** 0x00000000

Table 3-65 BLE_MISC Register

Bits	Field Name	RW	Reset	Description
31:1	RSVD	R	0x0	Reserved bits
0	SMC_WAKEUP_REQ	RW	0x0	Wake up SMC by MCU; it needs to be cleared by MCU as well.

3.7.1.22 COMM_TIMER_CFG0

- **Name:** COMM_TIMER_CFG0 Register
- **Description:** COMM_TIMER_CFG0 Register
- **Base Address:** 0x4000A000
- **Offset:** 0x288
- **Reset Value:** 0x00000000

Table 3-66 COMM_TIMER_CFG0 Register

Bits	Field Name	RW	Reset	Description
31:0	DEEPSLWKUP	RW	0x0	Used for Comm timer

3.7.1.23 COMM_TIMER_CFG1

- **Name:** COMM_TIMER_CFG1 Register
- **Description:** COMM_TIMER_CFG1 Register
- **Base Address:** 0x4000A000
- **Offset:** 0x28C
- **Reset Value:** 0x00000000

Table 3-67 COMM_TIMER_CFG1 Register

Bits	Field Name	RW	Reset	Description
31:0	ENBPRESET	RW	0x0	Used for Comm timer

3.7.1.24 BLE_DEEP_SLP_CORR_EN

- **Name:** BLE_DEEP_SLP_CORR_EN Register
- **Description:** BLE_DEEP_SLP_CORR_EN Register
- **Base Address:** 0x4000A000
- **Offset:** 0x290
- **Reset Value:** 0x00000001

Table 3-68 BLE_DEEP_SLP_CORR_EN Register

Bits	Field Name	RW	Reset	Description
31:2	RSVD	R	0x0	Reserved bits
1	HW_EN	RW	0x0	Enable HW Timer Correction Operation.
0	FW_EN	RW	0x1	Enable FW Timer Correction Operation.

3.7.1.25 BLE_DEEP_SLP_HW_TIMER_CORR

- **Name:** BLE_DEEP_SLP_HW_TIMER_CORR Register
- **Description:** BLE_DEEP_SLP_HW_TIMER_CORR Register
- **Base Address:** 0x4000A000
- **Offset:** 0x294

- **Reset Value:** 0x0F424000

Table 3-69 BLE_DEEP_SLP_HW_TIMER_CORR Register

Bits	Field Name	RW	Reset	Description
31:30	RSVD	R	0x0	Reserved bits
29:12	RTC_TOSC	RW	0XF424	Period of RTC clock in µs
11:10	RSVD	R	0x0	Reserved bits
9:0	WAIT	RW	0x0	Wait time in µs after wakeup before correction begins

3.7.1.26 COMM_TIMER_STAT

- **Name:** COMM_TIMER_STAT Register
- **Description:** COMM_TIMER_STAT Register
- **Base Address:** 0x4000A000
- **Offset:** 0x298
- **Reset Value:** 0x00000000

Table 3-70 COMM_TIMER_STAT Register

Bits	Field Name	RW	Reset	Description
31:0	DEEPSLSTAT	RW	0x0	Deep sleep status

3.7.1.27 PMU_COMP_GLITCH_REMOVE

- **Name:** PMU_COMP_GLITCH_REMOVE Register
- **Description:** PMU_COMP_GLITCH_REMOVE Register
- **Base Address:** 0x4000A000
- **Offset:** 0x2C0
- **Reset Value:** 0x00000000

Table 3-71 PMU_COMP_GLITCH_REMOVE Register

Bits	Field Name	RW	Reset	Description
31:3	RSVD	R	0x0	Reserved bits
2:0	CYCLE	RW	0x0	**Value:** • 0x0: no digital de-glitch • Others: 4 µs*N glitch removal

3.7.1.28 SOFTWARE_REG0

- **Name:** SOFTWARE_REG0 Register
- **Description:** SOFTWARE_REG0 Register
- **Base Address:** 0x4000A000
- **Offset:** 0x340
- **Reset Value:** 0x00000000

Table 3-72 SOFTWARE_REG0 Register

Bits	Field Name	RW	Reset	Description
31:0	SOFTWARE_REG0	RW	0x0	Used to save some values which can be used after waking up

3.7.1.29 SOFTWARE_REG1

- **Name:** SOFTWARE_REG1 Register
- **Description:** SOFTWARE_REG1 Register
- **Base Address:** 0x4000A000
- **Offset:** 0x344
- **Reset Value:** 0x00000000

Table 3-73 SOFTWARE_REG1 Register

Bits	Field Name	RW	Reset	Description
31:0	SOFTWARE_REG1	RW	0x0	Used to save some values which can be used after waking up

3.7.1.30 SOFTWARE_REG2

- **Name:** SOFTWARE_REG2 Register
- **Description:** SOFTWARE_REG2 Register
- **Base Address:** 0x4000A000
- **Offset:** 0x348
- **Reset Value:** 0x00000000

Table 3-74 SOFTWARE_REG2 Register

Bits	Field Name	RW	Reset	Description
31:0	SOFTWARE_REG2	RW	0x0	Used to save some values which can be used after waking up

3.7.1.31 DBG_REG0

- **Name:** DBG_REG0 Register
- **Description:** DBG_REG0 Register
- **Base Address:** 0x4000A000
- **Offset:** 0x380
- **Reset Value:** 0x00000000

Table 3-75 DBG_REG0 Register

Bits	Field Name	RW	Reset	Description
31:1	RSVD	R	0x0	Reserved bits
0	DBG_SYS_CLK_SEL	RW	0x0	0x1: Use the test logic to generate the final MCU_CLK_CTRL_SEL.

3.7.1.32 DBG_REG_RST_SRC

- **Name:** DBG_REG_RST_SRC Register
- **Description:** DBG_REG_RST_SRC Register
- **Base Address:** 0x4000A000
- **Offset:** 0x384
- **Reset Value:** 0x00000000

Table 3-76 DBG_REG_RST_SRC Register

Bits	Field Name	RW	Reset	Description
31	CFG	W	0x0	Write 1 to apply the bit [25:24].
30	BUSY	R	0x0	0x1: The block is being configured. Do not perform any write operation.
29:26	RSVD	R	0x0	Reserved bits
25	CLR	RW	0x0	1: Clear ALL the status
24	EN	RW	0x0	1: Enable reset source record.
23:15	RSVD	R	0x0	Reserved bits
16	READY	R	0x0	**Value:** • 0x0: The block is not ready to record. • 0x1: The block is ready to record.
15:5	RSVD	R	0x0	Reserved bits
4:0	SRC	WC	0x0	Reset season status:

Bits	Field Name	RW	Reset	Description
				• Bit 0: System reset. Clear the bit by writing 1 to it. • Bit 1: System watchdog reset. Clear the bit by writing 1 to it. • Bit 2: AON watchdog reset. Clear the bit by writing 1 to it. • Bit 3: System full reset. Clear the bit by writing 1 to it. • Bit 4: System partial reset. Clear the bit by writing 1 to it.

3.7.1.33 MEM_BOND_OPT

- **Name:** MEM_BOND_OPT Register
- **Description:** MEM_BOND_OPT Register
- **Base Address:** 0x4000A000
- **Offset:** 0x388
- **Reset Value:** 0x00000000

Table 3-77 MEM_BOND_OPT Register

Bits	Field Name	RW	Reset	Description
31:16	REG_MAGIC_WORD	W	0x0	The valid magic word is 16h5A5A. BOND_OPT cannot be written except when the magic word is valid.
15:2	RSVD	R	0x0	Reserved bits
1:0	BOND_OPT	RW	0x0	**Value:** • 0x0: 256 KB (SRAMs 0–7 ON) • 0x1: 192 KB (SRAMs 0–6 ON; SRAM 7 OFF) • 0x2: 128 KB (SRAMs 0–5 ON; SRAMs 6–7 OFF) • 0x3: RESERVE same as 0x0

3.7.2 Always-on Power Control

3.7.2.1 A_TIMING_CTRL0

- **Name:** A_TIMING_CTRL0 Register
- **Description:** A_TIMING_CTRL0 Register
- **Base Address:** 0x4000A400
- **Offset:** 0x0
- **Reset Value:** 0x00F900F9

Table 3-78 A_TIMING_CTRL0 Register

Bits	Field Name	RW	Reset	Description
31:25	RSVD	R	0x0	Reserved bits
24:16	DIG_LDO	RW	0xF9	DIG_LDO timing required for power-on
15:9	RSVD	R	0x0	Reserved bits
8:0	DCDC	RW	0xF9	DCDC timing required for power-on

3.7.2.2 A_TIMING_CTRL1

- **Name:** A_TIMING_CTRL1 Register
- **Description:** A_TIMING_CTRL1 Register
- **Base Address:** 0x4000A400
- **Offset:** 0x4
- **Reset Value:** 0x00050002

Table 3-79 A_TIMING_CTRL1 Register

Bits	Field Name	RW	Reset	Description
31:25	RSVD	R	0x0	Reserved bits
24:16	HF_OSC	RW	0x5	HF_OSC timing required for stability
15:0	RSVD	R	0x0	Reserved bits

3.7.2.3 A_TIMING_CTRL2

- **Name:** A_TIMING_CTRL2 Register
- **Description:** A_TIMING_CTRL2 Register
- **Base Address:** 0x4000A400
- **Offset:** 0x8
- **Reset Value:** 0x00F90018

Table 3-80 A_TIMING_CTRL2 Register

Bits	Field Name	RW	Reset	Description
31:25	RSVD	R	0x0	Reserved bits
24:16	PLL	RW	0xF9	PLL timing required for stability
15:9	RSVD	R	0x0	Reserved bits
8:0	PLL_LOCK	RW	0x18	PLL lock timing required for stability

3.7.2.4 A_TIMING_CTRL3

- **Name:** A_TIMING_CTRL3 Register
- **Description:** A_TIMING_CTRL3 Register
- **Base Address:** 0x4000A400
- **Offset:** 0xC
- **Reset Value:** 0x00F90002

Table 3-81 A_TIMING_CTRL3 Register

Bits	Field Name	RW	Reset	Description
31:25	RSVD	R	0x0	Reserved bits
24:16	XO	RW	0xF9	XO timing required for stability
15:9	RSVD	R	0x0	Reserved bits
8:0	PWR_SWITCH	RW	0x2	IO LDO or standard IO LDO power switch timing required for stability

3.7.2.5 A_TIMING_CTRL4

- **Name:** A_TIMING_CTRL4 Register
- **Description:** A_TIMING_CTRL4 Register
- **Base Address:** 0x4000A400
- **Offset:** 0x10
- **Reset Value:** 0x000000F9

Table 3-82 A_TIMING_CTRL4 Register

Bits	Field Name	RW	Reset	Description
31:9	RSVD	R	0x0	Reserved bits
8:0	XO_BIAS_SWITCH	RW	0xF9	XO bias timing required for stability

3.7.2.6 MEM_PWR_BOOST

- **Name:** MEM_PWR_BOOST Register
- **Description:** MEM_PWR_BOOST Register
- **Base Address:** 0x4000A400
- **Offset:** 0x14

- **Reset Value:**0x0000000B

Table 3-83 MEM_PWR_BOOST Register

Bits	Field Name	RW	Reset	Description
31:4	RSVD	R	0x0	Reserved bits
3:0	CTRL	RW	0xB	Provide memory power boost during memory power going retention for sleep. **Value:** • 0x0: Disable memory power boost • 0x1: 2 µs delay • 0x2: 4 µs delay • 0x3: 8 µs delay • 0x4: 16 µs delay • 0x5: 32 µs delay • 0x6: 64 µs delay • 0x7: 128 µs delay • 0x8: 256 µs delay • 0x9: 512 µs delay • 0xA: 1024 µs delay • 0xB: 2048 µs delay • 0xC: 4096 µs delay • 0xD: 8192 µs delay • 0xE: 16384 µs delay • 0xF: 32768 µs delay

3.7.2.7 A_SLP_CFG

- **Name:** A_SLP_CFG Register
- **Description:** A_SLP_CFG Register
- **Base Address:** 0x4000A400
- **Offset:** 0x20
- **Reset Value:** 0x00030017

Table 3-84 A_SLP_CFG Register

Bits	Field Name	RW	Reset	Description
31:18	RSVD	R	0x0	Reserved bits

Bits	Field Name	RW	Reset	Description
17	TRN_OFF_PLL	RW	0x1	PLL status in sleep mode **Value:** • 0x0: Keep the current status. • 0x1: Turn off
16	TRN_OFF_XO	RW	0x1	XO status in sleep mode **Value:** • 0x0: Keep the current status. • 0x1: Turn off
15:4	RSVD	R	0x0	Reserved bits
3	TRN_OFF_IO_LDO	RW	0x0	IO_LDO status in sleep mode **Value:** • 0x0: Keep the current status. • 0x1: Turn off
2	TRN_OFF_HF_OSC	RW	0x1	HF_OSC status in sleep mode **Value:** • 0x0: Keep the current status. • 0x1: Turn off
1	TRN_OFF_DIG_LDO	RW	0x1	DIG_LDO status in sleep mode **Value:** • 0x0: Keep the current status. • 0x1: Turn off
0	TRN_OFF_DCDC	RW	0x1	DC-DC status in sleep mode **Value:** • 0x0: Keep the current status. • 0x1: Turn off

3.7.2.8 AON_START_CFG

- **Name:** AON_START_CFG Register
- **Description:** AON_START_CFG Register
- **Base Address:** 0x4000A400
- **Offset:** 0x24
- **Reset Value:** 0x00000050

Table 3-85 AON_START_CFG Register

Bits	Field Name	RW	Reset	Description
31:7	RSVD	R	0x0	Reserved bits
6	PLL_EN_PWR	RW	0x1	1: Enable PLL after DC-DC gets ready.
5	RSVD	R	0x0	Reserved bit
4	XO_EN_PWR	RW	0x1	1: Enable XO after DC-DC gets ready.
3	RSVD	R	0x0	Reserved bit
2	MCU_PWR_TYPE	RW	0x0	Power mode: **Value:** • 0x0: (DCDC_MODE) Power MCU domain ON after dig_ldo being ready (Slow. ~3ms) Note: 1 is forbidden! Please keep this bit always 0
1:0	RSVD	R	0x0	Reserved bits

3.7.2.9 DPAD_LE_CTRL

- **Name:** DPAD_LE_CTRL Register
- **Description:** DPAD_LE_CTRL Register
- **Base Address:** 0x4000A400
- **Offset:** 0x28
- **Reset Value:** 0x00040000

Table 3-86 DPAD_LE_CTRL Register

Bits	Field Name	RW	Reset	Description
31:19	RSVD	R	0x0	Reserved bit
18	WAKEUP	RW	0x1	Set HW controlled dpad_le value after waking up.
17	RSVD	R	0x0	Reserved bit
16	SLEEP	RW	0x0	Set HW controlled dpad_le value during sleep.
15:11	RSVD	R	0x0	Reserved bit
10	BUSY	R	0x0	DPAD_LE_CTRL is applying a value. Do not apply a value again.
9	RSVD	R	0x0	Reserved bit
8	READ	R	0x0	The current DPAD Latch value
7:3	RSVD	R	0x0	Reserved bit
2	APPLY_WR	W	0x0	Write 1 to apply the value of WR to control signal.

Bits	Field Name	RW	Reset	Description
1	RSVD	R	0x0	Reserved bit
0	WR	RW	0x0	Enable DPAD Latch. **Value:** • 0x0: Latch the pad configurations. • 0x1: The DPAD latch is transparent.

3.7.2.10 AON_SLP_CTRL

- **Name:** AON_SLP_CTRL Register
- **Description:** AON_SLP_CTRL Register
- **Base Address:** 0x4000A400
- **Offset:** 0x3C
- **Reset Value:** 0x00000000

Table 3-87 AON_SLP_CTRL Register

Bits	Field Name	RW	Reset	Description
31:9	RSVD	R	0x0	Reserved bits
8	PRO	R	0x0	The power controller is processing a deep-sleep request.
7:1	RSVD	R	0x0	Reserved bits
0	REQ	W	0x0	Write 1 to send a sleep request.

3.7.2.11 XO_PLL_SET

- **Name:** XO_PLL_SET Register
- **Description:** XO_PLL_SET Register
- **Base Address:** 0x4000A400
- **Offset:** 0x40
- **Reset Value:** 0x00000000

Table 3-88 XO_PLL_SET Register

Bits	Field Name	RW	Reset	Description
31:3	RSVD	R	0x0	Reserved bits
2	HF_OSC_SET	W	0x0	Write 1 to enable HFOSC. This operation will turn DC-DC on if it is not.

Bits	Field Name	RW	Reset	Description
1	XO_SET	W	0x0	Write 1 to enable HFXO. This operation will turn DC-DC on if it is not.
0	PLL_SET	W	0x0	Write 1 to enable CPLL. This operation will turn DC-DC on if it is not.

3.7.2.12 XO_PLL_CLR

- **Name:** XO_PLL_CLR Register
- **Description:** XO_PLL_CLR Register
- **Base Address:** 0x4000A400
- **Offset:** 0x44
- **Reset Value:** 0x00000000

Table 3-89 XO_PLL_CLR Register

Bits	Field Name	RW	Reset	Description
31:3	RSVD	R	0x0	Reserved bits
2	HF_OSC_CLR	W	0x0	Write 1 to disable HFOSC.
1	XO_CLR	W	0x0	Write 1 to disable HFXO.
0	PLL_CLR	W	0x0	Write 1 to disable CPLL.

3.7.2.13 XO_PLL_STAT

- **Name:** XO_PLL_STAT Register
- **Description:** XO_PLL_STAT Register
- **Base Address:** 0x4000A400
- **Offset:** 0x48
- **Reset Value:** 0x00000000

Table 3-90 XO_PLL_STAT Register

Bits	Field Name	RW	Reset	Description
31:3	RSVD	R	0x0	Reserved bits
2	HF_STAT	R	0x0	HF_OSC stable state
1	XO_STAT	R	0x0	XO stable state
0	PLL_STAT	R	0x0	PLL stable state

Copyright © 2024 Shenzhen Goodix Technology Co., Ltd.

3.7.2.14 PWR_SET

- **Name:** PWR_SET Register
- **Description:** PWR_SET Register
- **Base Address:** 0x4000A400
- **Offset:** 0x4C
- **Reset Value:** 0x00000000

Table 3-91 PWR_SET Register

Bits	Field Name	RW	Reset	Description
31:4	RSVD	R	0x0	Reserved bits
3	SWC_DIG_LDO_SET	W	0x0	Switch to dig_ldo power request.
2	RSVD	R	0x0	Reserved bit
1	DIG_LDO_SET	W	0x0	Digital LDO power request
0	DCDC_SET	W	0x0	DC-DC power request

3.7.2.15 PWR_CLR

- **Name:** PWR_CLR Register
- **Description:** PWR_CLR Register
- **Base Address:** 0x4000A400
- **Offset:** 0x50
- **Reset Value:** 0x00000000

Table 3-92 PWR_CLR Register

Bits	Field Name	RW	Reset	Description
31:4	RSVD	R	0x0	Reserved bits
3	SWC_DIG_LDO_CLR	W	0x0	Disable Switch to dig_ldo power.
2	RSVD	R	0x0	Reserved bit
1	DIG_LDO_CLR	W	0x0	Disable digital LDO power.
0	DCDC_CLR	W	0x0	Disable DC-DC power.

3.7.2.16 PWR_STAT

- **Name:** PWR_STAT Register
- **Description:** PWR_STAT Register
- **Base Address:** 0x4000A400

- **Offset:** 0x54
- **Reset Value:** 0x00000000

Table 3-93 PWR_STAT Register

Bits	Field Name	RW	Reset	Description
31:2	RSVD	R	0x0	Reserved bits
1	DIG_LDO_AVL	R	0x0	Digital LDO power is available. The status is NOT valid if analog interface overriding is used.
0	DCDC_AVL	R	0x0	DCDC power is available. The status is NOT valid if analog interface overriding is used.

3.7.2.17 MEM_PWR_SLP0

- **Name:** MEM_PWR_SLP0 Register
- **Description:** MEM_PWR_SLP0 Register
- **Base Address:** 0x4000A400
- **Offset:** 0x80
- **Reset Value:** 0x0000FFFF

Table 3-94 MEM_PWR_SLP0 Register

Bits	Field Name	RW	Reset	Description
31:16	RSVD	R	0x0	Reserved bits
15:14	SET07	RW	0x3	Memory power state that is loaded in sleep phase **Value:** • 0x0: OFF • 0x1: light sleep • 0x2: full power • 0x3: retention
13:12	SET06	RW	0x3	Memory power state that is loaded in sleep phase **Value:** • 0x0: OFF • 0x1: light sleep • 0x2: full power • 0x3: retention
11:10	SET05	RW	0x3	Memory power state that is loaded in sleep phase **Value:**

Bits	Field Name	RW	Reset	Description
				• 0x0: OFF • 0x1: light sleep • 0x2: full power • 0x3: retention
9:8	SET04	RW	0x3	Memory power state that is loaded in sleep phase **Value:** • 0x0: OFF • 0x1: light sleep • 0x2: full power • 0x3: retention
7:6	SET03	RW	0x3	Memory power state that is loaded in sleep phase **Value:** • 0x0: OFF • 0x1: light sleep • 0x2: full power • 0x3: retention
5:4	SET02	RW	0x3	Memory power state that is loaded in sleep phase **Value:** • 0x0: OFF • 0x1: light sleep • 0x2: full power • 0x3: retention
3:2	SET01	RW	0x3	Memory power state that is loaded in sleep phase **Value:** • 0x0: OFF • 0x1: light sleep • 0x2: full power • 0x3: retention
1:0	SET00	RW	0x3	Memory power state that is loaded in sleep phase **Value:** • 0x0: OFF • 0x1: light sleep • 0x2: full power

Bits	Field Name	RW	Reset	Description
				• 0x3: retention

3.7.2.18 MEM_PWR_SLP1

- **Name:** MEM_PWR_SLP1 Register
- **Description:** MEM_PWR_SLP1 Register
- **Base Address:** 0x4000A400
- **Offset:** 0x84
- **Reset Value:** 0x000003FF

Table 3-95 MEM_PWR_SLP1 Register

Bits	Field Name	RW	Reset	Description
31:10	RSVD	R	0x0	
9:8	KEYRAM_SET	RW	0x3	Keyram memory sleep power. **Value:** • 0x0: OFF • 0x1: light sleep • 0x2: full power • 0x3: retention
7:6	PACKET_SET	RW	0x3	BLE packet memory sleep power **Value:** • 0x0: OFF • 0x1: light sleep • 0x2: full power • 0x3: retention
5:4	ICACHE_SET	RW	0x3	Cache memory sleep power Memory power states that is loaded in sleep phase **Value:** • 0x0: OFF • 0x1: light sleep • 0x2: full power • 0x3: retention
3:2	HTM_DM_SET	RW	0x3	Hopping table DM memory sleep power Memory power state that is loaded in sleep phase **Value:**

Bits	Field Name	RW	Reset	Description
				• 0x0: OFF • 0x1: light sleep • 0x2: full power • 0x3: retention
1:0	HTM_AM_SET	RW	0x3	Hopping table AM memory sleep power. Hopping table AM and DM are sharing the same VDM. So the final pwr settings are determined by both settings. **Value:** • 0x0: OFF • 0x1: light sleep • 0x2: full power • 0x3: retention DM AM Result OFF OFF OFF OFF RET RET OFF ON ON RET OFF RET RET RET RET RET ON ON ON OFF ON ON RET ON ON ON ON

3.7.2.19 MEM_PWR_WKUP0

- **Name:** MEM_PWR_WKUP0 Register
- **Description:** MEM_PWR_WKUP0 Register
- **Base Address:** 0x4000A400
- **Offset:** 0x88
- **Reset Value:** 0x0000AAAA

Table 3-96 MEM_PWR_WKUP0 Register

Bits	Field Name	RW	Reset	Description
31:16	RSVD	R	0x0	
15:14	SET07	RW	0x2	Memory power state that is loaded in wakeup phase

Bits	Field Name	RW	Reset	Description
				Value: • 0x0: OFF • 0x1: light sleep • 0x2: full power • 0x3: retention
13:12	SET06	RW	0x2	Memory power state that is loaded in wakeup phase **Value:** • 0x0: OFF • 0x1: light sleep • 0x2: full power • 0x3: retention
11:10	SET05	RW	0x2	Memory power state that is loaded in wakeup phase **Value:** • 0x0: OFF • 0x1: light sleep • 0x2: full power • 0x3: retention
9:8	SET04	RW	0x2	Memory power state that is loaded in wakeup phase **Value:** • 0x0: OFF • 0x1: light sleep • 0x2: full power • 0x3: retention
7:6	SET03	RW	0x2	Memory power state that is loaded in wakeup phase **Value:** • 0x0: OFF • 0x1: light sleep • 0x2: full power • 0x3: retention
5:4	SET02	RW	0x2	Memory power state that is loaded in wakeup phase **Value:** • 0x0: OFF • 0x1: light sleep

Bits	Field Name	RW	Reset	Description
				• 0x2: full power • 0x3: retention
3:2	SET01	RW	0x2	Memory power state that is loaded in wakeup phase **Value:** • 0x0: OFF • 0x1: light sleep • 0x2: full power • 0x3: retention
1:0	SET00	RW	0x2	Memory power state that is loaded in wakeup phase **Value:** • 0x0: OFF • 0x1: light sleep • 0x2: full power • 0x3: retention

3.7.2.20 MEM_PWR_WKUP1

- **Name:** MEM_PWR_WKUP1 Register
- **Description:** MEM_PWR_WKUP1 Register
- **Base Address:** 0x4000A400
- **Offset:** 0x8C
- **Reset Value:** 0x00000020

Table 3-97 MEM_PWR_WKUP1 Register

Bits	Field Name	RW	Reset	Description
31:10	RSVD	R	0x0	
9:8	KEYRAM	RW	0x0	Keyram memory sleep power. **Value:** • 0x0: OFF • 0x1: light sleep • 0x2: full power • 0x3: retention
7:6	PACKET	RW	0x0	Bluetooth LE packet memory sleep power **Value:** • 0x0: OFF

Bits	Field Name	RW	Reset	Description
				• 0x1: light sleep • 0x2: full power • 0x3: retention
5:4	ICACHE	RW	0x2	Cache memory sleep power Memory power state that is loaded on wakeup **Value:** • 0x0: OFF • 0x1: light sleep • 0x2: full power • 0x3: retention
3:2	HTM_DM	RW	0x0	Hopping table DM memory sleep power Memory power states that is loaded on wakeup **Value:** • 0x0: OFF • 0x1: light sleep • 0x2: full power • 0x3: retention
1:0	HTM_AM	RW	0x0	Hopping table AM memory sleep power. Hopping table AM and DM are sharing the same VDM. So the final pwr settings are determined by both settings. **Value:** • 0x0: OFF • 0x1: light sleep • 0x2: full power • 0x3: retention DM AM Result OFF OFF OFF OFF RET RET OFF ON ON RET OFF RET RET RET RET RET ON ON ON OFF ON ON RET ON

Bits	Field Name	RW	Reset	Description
				ON ON ON

3.7.2.21 MEM_PWR_APPLY

- **Name:** MEM_PWR_APPLY Register
- **Description:** MEM_PWR_APPLY Register
- **Base Address:** 0x4000A400
- **Offset:** 0x90
- **Reset Value:** 0x00000000

Table 3-98 MEM_PWR_APPLY Register

Bits	Field Name	RW	Reset	Description
31:17	RSVD	R	0x0	
16	BUSY	R	0x0	During the bit being 1, writing APPLY would not take any effect.
15:1	RSVD	R	0x0	
0	APPLY	W	0x0	Write 1 to apply the memory settings in MEM_PWR_WKUP manually.

3.7.2.22 MEM_PWR_STAT0

- **Name:** MEM_PWR_STAT0 Register
- **Description:** MEM_PWR_STAT0 Register
- **Base Address:** 0x4000A400
- **Offset:** 0x98
- **Reset Value:** 0x0000AAAA

Table 3-99 MEM_PWR_STAT0 Register

Bits	Field Name	RW	Reset	Description
31:16	RSVD	R	0x0	
15:14	SET07	R	0x2	Memory power states. Do NOT read when MEM_PWR_APPLY_BUSY is high. **Value:** • 0x0: OFF • 0x1: light sleep

Bits	Field Name	RW	Reset	Description
				• 0x2: full power • 0x3: retention
13:12	SET06	R	0x2	Memory power states. Do NOT read when MEM_PWR_APPLY_BUSY is high. **Value:** • 0x0: OFF • 0x1: light sleep • 0x2: full power • 0x3: retention
11:10	SET05	R	0x2	Memory power states. Do NOT read when MEM_PWR_APPLY_BUSY is high. **Value:** • 0x0: OFF • 0x1: light sleep • 0x2: full power • 0x3: retention
9:8	SET04	R	0x2	Memory power states. Do NOT read when MEM_PWR_APPLY_BUSY is high. **Value:** • 0x0: OFF • 0x1: light sleep • 0x2: full power • 0x3: retention
7:6	SET03	R	0x2	Memory power states. Do NOT read when MEM_PWR_APPLY_BUSY is high. **Value:** • 0x0: OFF • 0x1: light sleep • 0x2: full power • 0x3: retention
5:4	SET02	R	0x2	Memory power states. Do NOT read when MEM_PWR_APPLY_BUSY is high. **Value:** • 0x0: OFF

Bits	Field Name	RW	Reset	Description
				• 0x1: light sleep • 0x2: full power • 0x3: retention
3:2	SET01	R	0x2	Memory power states. Do NOT read when MEM_PWR_APPLY_BUSY is high. **Value:** • 0x0: OFF • 0x1: light sleep • 0x2: full power • 0x3: retention
1:0	SET00	R	0x2	Memory power states. Do NOT read when MEM_PWR_APPLY_BUSY is high. **Value:** • 0x0: OFF • 0x1: light sleep • 0x2: full power • 0x3: retention

3.7.2.23 MEM_PWR_STAT1

- **Name:** MEM_PWR_STAT1 Register
- **Description:** MEM_PWR_STAT1 Register
- **Base Address:** 0x4000A400
- **Offset:** 0x9C
- **Reset Value:** 0x00000000

Table 3-100 MEM_PWR_STAT1 Register

Bits	Field Name	RW	Reset	Description
31:10	RSVD	R	0x0	
9:8	KEYRAM_SET	RW	0x0	Memory power states. Do not read when MEM_PWR_APPLY_BUSY is high. **Value:** • 0x0: OFF • 0x1: light sleep • 0x2: full power

Bits	Field Name	RW	Reset	Description
				• 0x3: retention
7:6	PACKET_SET	RW	0x0	Memory power states. Do not read when MEM_PWR_APPLY_BUSY is high. **Value:** • 0x0: OFF • 0x1: light sleep • 0x2: full power • 0x3: retention
5:4	ICACHE_SET	R	0x0	Memory power states. Do not read when MEM_PWR_APPLY_BUSY is high. **Value:** • 0x0: OFF • 0x1: light sleep • 0x2: full power • 0x3: retention
3:2	HTM_DM_SET	R	0x0	Memory power states. Do not read when MEM_PWR_APPLY_BUSY is high. **Value:** • 0x0: OFF • 0x1: light sleep • 0x2: full power • 0x3: retention
1:0	HTM_AM_SET	R	0x0	Memory power states. Do not read when MEM_PWR_APPLY_BUSY is high. **Value:** • 0x0: OFF • 0x1: light sleep • 0x2: full power • 0x3: retention

3.7.2.24 MEM_PWR_DBG

- **Name:** MEM_PWR_DBG Register
- **Description:** MEM_PWR_DBG Register
- **Base Address:** 0x4000A400

- **Offset:** 0xBC
- **Reset Value:** 0x0000001C

Table 3-101 MEM_PWR_DBG Register

Bits	Field Name	RW	Reset	Description
31:5	RSVD	R	0x0	
4:2	RSVD	R	0x6F	
1	CLR_AONMASK	RW	0x0	Write 1, then write 0 to force clear the aon_mask
0	RESET_ON_SRAM	RW	0x0	1: When MCU issues a system-reset request, turn ON all SRAM power. Do not change this register when system-reset is ongoing.

3.7.2.25 COMM_CORE_PWR_CTRL_SW

- **Name:** COMM_CORE_PWR_CTRL_SW Register
- **Description:** COMM_CORE_PWR_CTRL_SW Register
- **Base Address:** 0x4000A400
- **Offset:** 0xC0
- **Reset Value:** 0x00000002

Table 3-102 COMM_CORE_PWR_CTRL_SW Register

Bits	Field Name	RW	Reset	Description
31:3	RSVD	R	0x0	
2	RST_N	RW	0x0	Comm proc reset active low When this bit is 0, Bluetooth LE is held in reset.
1	ISO_EN	RW	0x1	Enable comm_core power domain isolation. **Value:** • 0x0: NO isolation • 0x1: isolated
0	CORE_EN	RW	0x0	Enable comm_core power domain power. **Value:** • 0x0: OFF • 0x1: ON

3.7.2.26 COMM_CORE_PWR_CTRL_HW_CFG

- **Name:** COMM_CORE_PWR_CTRL_HW_CFG Register

- **Description:** COMM_CORE_PWR_CTRL_HW_CFG Register
- **Base Address:** 0x4000A400
- **Offset:** 0xC4
- **Reset Value:** 0x00020200

Table 3-103 COMM_CORE_PWR_CTRL_HW_CFG Register

Bits	Field Name	RW	Reset	Description
31:19	RSVD	R	0x0	
18	PD_HOLD_EN_RD	R	0x0	Real-time status of PD_HOLD_EN
17	AUTO_EN_RD	R	0x1	Real-time status of AUTO_EN
16	HW_EN_RD	R	0x0	Real-time status of HW_EN
15:12	RSVD	R	0x0	
11	PLL_AUTO_ON	RW	0x0	Waking Bluetooth LE timer up will enable PLL automatically.
10	PD_HOLD_EN	RW	0x0	Wait the Bluetooth LE timer flag and the delay counter before powering off Bluetooth LE.
9	AUTO_EN	RW	0x1	1: If HW sequencer is selected, set 1 to let HW automatically request power-on when a COMM Timer wakeup signal is received. Please allow 6 µs–12 µs for the configuration to be applied.
8	HW_EN	RW	0x0	Enable the hardware sequencer block. Please allow 6 µs–12 µs for the configuration to be applied.
7:1	RSVD	R	0x0	
0	CTRL_SEL	RW	0x0	**Value:** • 0x0: Use the register (COMM_CORE_PWR_CTRL_SW) to control comm_core power sequence. • 0x1: Use Hardware power sequencer.

3.7.2.27 COMM_CORE_PWR_CTRL_HW_CTRL

- **Name:** COMM_CORE_PWR_CTRL_HW_CTRL Register
- **Description:** COMM_CORE_PWR_CTRL_HW_CTRL Register
- **Base Address:** 0x4000A400
- **Offset:** 0xC8
- **Reset Value:** 0x00000000

Table 3-104 COMM_CORE_PWR_CTRL_HW_CTRL Register

Bits	Field Name	RW	Reset	Description
31:9	RSVD	R	0x0	
8	DIS	W	0x0	Write 1 to start comm_core power-off sequence. Writing 0 will take no effect.
7:1	RSVD	R	0x0	
0	REQ	W	0x0	Write 1 to start comm_core power-on sequence. Writing 0 will take no effect.

3.7.2.28 COMM_CORE_PWR_CTRL_HW_STAT

- **Name:** COMM_CORE_PWR_CTRL_HW_STAT Register
- **Description:** COMM_CORE_PWR_CTRL_HW_STAT Register
- **Base Address:** 0x4000A400
- **Offset:** 0xCC
- **Reset Value:** 0x00000000

Table 3-105 COMM_CORE_PWR_CTRL_HW_STAT Register

Bits	Field Name	RW	Reset	Description
31:3	RSVD	R	0x0	
2	HW_PD_WAIT	R	0x0	1: Power-off command received, waiting for PD condition
1	HW_PWR	R	0x0	**Value:** • 0x0: OFF state or going to be OFF state • 0x1: ON state or going to be ON state
0	BUSY	R	0x0	**Value:** • 0x0: HW power controller is ready to receive a command. • 0x1: HW power controller is working.

3.7.2.29 COMM_CORE_PWR_CTRL_PD_DLY

- **Name:** COMM_CORE_PWR_CTRL_PD_DLY Register
- **Description:** COMM_CORE_PWR_CTRL_PD_DLY Register
- **Base Address:** 0x4000A400
- **Offset:** 0xD0
- **Reset Value:** 0x00000000

Table 3-106 COMM_CORE_PWR_CTRL_PD_DLY Register

Bits	Field Name	RW	Reset	Description
31:17	RSVD	R	0x0	
16	PD_FLAG	RW	0x0	1: Do not wait for Bluetooth LE timer flag to power off the Bluetooth LE.
15:8	RSVD	R	0x0	
7:0	PD_DLY	RW	0x0	Delay after receiving Bluetooth LE timer flag • 0: no delay • >0: N x 4 µs

3.7.2.30 COMM_CORE_PWR_STAT

- **Name:** COMM_CORE_PWR_STAT Register
- **Description:** COMM_CORE_PWR_STAT Register
- **Base Address:** 0x4000A400
- **Offset:** 0xD4
- **Reset Value:** 0x00000002

Table 3-107 COMM_CORE_PWR_STAT Register

Bits	Field Name	RW	Reset	Description
31:3	RSVD	R	0x0	
2	RST_N_RD	R	0x0	Comm proc reset active low When this bit is 0, Bluetooth LE is held in reset.
1	ISO_EN_RD	R	0x1	Enable comm_core power domain isolation. **Value:** • 0x0: NO isolation • 0x1: isolated
0	CORE_EN_RD	R	0x0	Enable comm_core power domain power. **Value:** • 0x0: OFF • 0x1: ON

3.7.2.31 COMM_TIMER_PWR_CTRL

- **Name:** COMM_TIMER_PWR_CTRL Register
- **Description:** COMM_TIMER_PWR_CTRL Register

- **Base Address:** 0x4000A400
- **Offset:** 0xE0
- **Reset Value:** 0x00000002

Table 3-108 COMM_TIMER_PWR_CTRL Register

Bits	Field Name	RW	Reset	Description
31:3	RSVD	R	0x0	
2	RST_N	RW	0x0	Comm timer reset This bit is used to reset the Comm timer after it has been powered on. **Value:** • 0x0: Comm timer reset • 0x1: Comm timer reset released
1	ISO_EN	RW	0x1	Enable comm_timer power domain isolation. **Value:** • 0x0: NO isolation • 0x1: isolated
0	EN	RW	0x0	Enable comm_timer power domain power. **Value:** • 0x0: OFF • 0x1: ON

3.7.2.32 SOFTWARE_REG_3

- **Name:** SOFTWARE_REG_3 Register
- **Description:** SOFTWARE_REG_3 Register
- **Base Address:** 0x4000A400
- **Offset:** 0xE8
- **Reset Value:** 0x00000000

Table 3-109 SOFTWARE_REG_3 Register

Bits	Field Name	RW	Reset	Description
31:0	SOFTWARE_REG3	RW	0x0	Software can use this register to save data. It will not be reset by NVIC or in sleep.

3.7.3 Always-on PMU Control

3.7.3.1 RF_REG_0

- **Name:** RF_REG_0 Register
- **Description:** RF_REG_0 Register
- **Base Address:** 0x4000A800
- **Offset:** 0x4
- **Reset Value:** 0x0F02CD30

Table 3-110 RF_REG_0 Register

Bits	Field Name	RW	Reset	Description
31:24	IO_LDO_REG1	RW	0xF	[7] Enable Bypass IO LDO [6:0] Default is set to 1.8 V, LSB = 8.5 mV
23:16	LPD_REG2	RW	0x2	[7:6] NC [5:4] NC [3] 0: not allow to use boost set 1: use boost set [2] 0: close srpg_ldo [1:0] 1.1V LDO Vref trimming/output option(1.05/1.1/1.15/1.2)
15:8	LPD_REG1	RW	0xCD	[7] 1: operating at 2M mode 0: operating at low fequency mode [6:5] 00: 1.3M, 01: 1.7M, 10: 2M 11: 2.4M trimming [4:2] Retention level :0.6/0.65/0.7/0.75/0.8/0.9/1.0/1.1 [1] 0: not allow to use boost set 1: use boost set [0] 0: close ret_ldo
7:0	RTC_REG1	RW	0x30	[7] Enable RTC. [6] Enable Constant GM mode. [5] NC [4:0] change inverter width

3.7.3.2 RF_REG_1

- **Name:** RF_REG_1 Register
- **Description:** RF_REG_1 Register
- **Base Address:** 0x4000A800
- **Offset:** 0x8
- **Reset Value:** 0x91941F00

Table 3-111 RF_REG_1 Register

Bits	Field Name	RW	Reset	Description
31:24	DCDC_REG4	RW	0x91	[7] 1: Enable time-out circuit [6:4] ZCS_OCP_IN_2_0 [3:1] ZCS_OCN_IN_2_0 [0] 1: register can change DCDC Ton 0: cannot change
23:16	DCDC_REG3	RW	0x94	[7] NC [6:3] NC [2:0] NC
15:8	DCDC_REG2	RW	0x1F	[7] reset DCDC state machine [6] 1: use XO. 0: use internal OSC. [5] 1:testmode; 0:not testmode [4] 1: use VREG as DCDC power to save energy 0: use internal LDO as DCDC power [3:2] program the number of N-power switch [1:0] program the number of P-power switch
7:0	DCDC_REG1	RW	0x0	[7] MPX generate DIG signal [6] MPX generate ANA signal [5:3] select MPX`s dig signal [2] MPX not use OPAMP but bypass [1:0] NC

3.7.3.3 RF_REG_2

- **Name:** RF_REG_2 Register
- **Description:** RF_REG_2 Register
- **Base Address:** 0x4000A800
- **Offset:** 0xC
- **Reset Value:** 0x0000A000

Table 3-112 RF_REG_2 Register

Bits	Field Name	RW	Reset	Description
31:16	RSVD	R	0x0	Rserved bits
15:8	GP_REG1	RW	0xA0	[7:5] Reference control - default 101 [4] Delayed Enable signal for efuse_vddq (2.5 V) [3] Enable signal for efuse_vddq (2.5 V)

Bits	Field Name	RW	Reset	Description
				[2] Enable signal for efuse_vdd (1.1 V) [1] Bypass vddq to VBATL (when EN=1) [0] NC
7:0	MUX1_REG1	RW	0x00	[7] Enable MSIO path [6] Enable MUX [5:3] Signal Selection inside Block: refer to MUX tab [2:0] Block selection: refer to MUX tab

3.7.3.4 RF_REG_3

- **Name:** RF_REG_3 Register
- **Description:** RF_REG_3 Register
- **Base Address:** 0x4000A800
- **Offset:** 0x10
- **Reset Value:** 0x52000031

Table 3-113 RF_REG_3 Register

Bits	Field Name	RW	Reset	Description
31:24	IO_LDO_REG2	RW	0x52	[7:4] STB_IO_LDO Vref trimming [3:2] STB_IO_LDO output option (1.8/2.5/3.3) [1] STB_IO_LDO enable [0] rg_stb_ioldo_output
23:6	RSVD	R	0x0	Rserved bits
5	BOD_STATIC_EN	RW	0x1	The dynamic mode has lower power but lower accuracy. **Value:** • 0x0: Enable dynamic mode. • 0x1: Enable static mode.
4:2	BOD2_LVL_CTRL_LV	RW	0x4	BOD2 level control bits
1	BOD2_DIS	RW	0x0	**Value:** • 0x0: Disable BOD interrupt. • 0x1: Enable BOD interrupt.
0	BOD_DIS	RW	0x1	**Value:** • 0x0: Disable BOD reset. • 0x1: Enable BOD reset.

3.7.3.5 RF_REG_4

- **Name:** RF_REG_4 Register
- **Description:** RF_REG_4 Register
- **Base Address:** 0x4000A800
- **Offset:** 0x14
- **Reset Value:** 0x00185AC4

Table 3-114 RF_REG_4 Register

Bits	Field Name	RW	Reset	Description
31:24	RSVD	R	0x0	Rserved bits
23:16	CORE_LDO_REG1	RW	0x18	[7] 0: vddcore not equal to vddaon; 1: vddcore equal vddaon [6] Enables Bypass mode [5:4] NC [3:2] 00->11 for 0.8/.9/1.0/bypass [1:0] NC
15:8	DCDC_REG6	RW	0x5A	[7:3] scale DCDC`s internal clk [2:0] BBM_delay_1V_2_0
7:0	DCDC_REG5	RW	0xC4	[7] 1: enabes PMU BG independent of the DCDC [6:4] bandgap TC fine trim,3b'000: most negative, used to be R2_trim [3] NC [2:0] bandgap TC coarse trim,3b'000:most negative, used to be R1 trim

3.7.3.6 RC_RTC_REG_0

- **Name:** RC_RTC_REG_0 Register
- **Description:** RC_RTC_REG_0 Register
- **Base Address:** 0x4000A800
- **Offset:** 0x48
- **Reset Value:** 0x101014CB

Table 3-115 RC_RTC_REG_0 Register

Bits	Field Name	RW	Reset	Description
31:24	RTC_CAP	RW	0x10	[7:6] trimming the feedback resistor, default 00 is max [5:0] cap control

Bits	Field Name	RW	Reset	Description
23:16	RTC_CS	RW	0x10	[7] NC [6] NC [5] NC [4:0] CS
15:8	RC_REG2	RW	0x14	[7] Enable RCOSC. [6] Enable RCOSC delay. [5:3] RCOSC rdiv_delay_ctrl [2:0] RCOSC resn_ctrl
7:0	RC_REG1	RW	0xCB	[7:6] RCOSC bias control [5] cgm_byp [4] NC [3:0] NC

3.7.3.7 DCDC_LDO_REG_0

- **Name:** DCDC_LDO_REG_0 Register
- **Description:** DCDC_LDO_REG_0 Register
- **Base Address:** 0x4000A800
- **Offset:** 0x4C
- **Reset Value:** 0x8A00028A

Table 3-116 DCDC_LDO_REG_0 Register

Bits	Field Name	RW	Reset	Description
31:24	DIGIO_LDO_REG3	RW	0x8A	[7:0] NC
23:16	DIGIO_LDO_REG2	RW	0x0	[7:0] NC
15:8	DIGIO_LDO_REG1	RW	0x2	[7:0] NC
7:0	DCDC_REG7	RW	0x8A	[7:5] Controls the LPD BG temp Co. [4:2] prebias IPAT current trimming [1:0] NC

3.7.3.8 FS_REG_0

- **Name:** FS_REG_0 Register
- **Description:** FS_REG_0 Register
- **Base Address:** 0x4000A800
- **Offset:** 0x50

- **Reset Value:** 0x010029C6

Table 3-117 FS_REG_0 Register

Bits	Field Name	RW	Reset	Description
31:24	RSVD	R	0x0	Rserved bits
23:16	FS_REG3	RW	0x100	Fine code for hf_clk frequency adjustment. Used for background calibration. LSB ~ 1000 ppm
15:8	FS_REG2	RW	0x29	[7:6] NC [5:2] control input reference voltage. Hibit=0,810 mV-1020 mV; hibit=1, 1050 mV-1400 mV [1] sys ldo enable signal. 1: enabled [0] 1: enable with DCDC. 0: cntl separately
7:0	FS_REG1	RW	0xC6	[7:0] NC

3.7.3.9 FS_REG_1

- **Name:** FS_REG_1 Register
- **Description:** FS_REG_1 Register
- **Base Address:** 0x4000A800
- **Offset:** 0x54
- **Reset Value:** 0x0000542

Table 3-118 FS_REG_1 Register

Bits	Field Name	RW	Reset	Description
31:16	RSVD	R	0x0	Rserved bits
15:8	FS_REG5	RW	0x54	[7:0] NC
7:0	FS_REG4	RW	0x28	[7] Enables MUX path [6] NC [5:4] NC [3:0] coarse code that controls hf_clk frequency. Use default: 1000 (Need one-time calibration using the on-chip frequency calibration)

3.7.3.10 ADVS_DCDC

- **Name:** ADVS_DCDC Register
- **Description:** ADVS_DCDC Register

- **Base Address:** 0x4000A800
- **Offset:** 0x5C
- **Reset Value:** 0x00004340

Table 3-119 ADVS_DCDC Register

Bits	Field Name	RW	Reset	Description
31:16	RSVD	R	0x0	Rserved bits
15:8	DVS_DCDC_REG2	RW	0x43	[7] NC [6:4] Controls the default level value of the VT bias. (this should not be changed, rather the LDO/DCDC control bits should be used to set up the output to whatever value is needed) [3] NC [2:0] Programs the lower limit for the AVS output voltage (only when limiter function is enabled) this can be used to prevent the FFFF 105 corner from going too low if needed.
7:0	DVS_DCDC_REG1	RW	0x40	[7] NC [6] Enable AVS mode [5] Controls the slope (1 = higher slope) [4] Enables lower limiter function to prevent the AVS from dropping below a set limit [3:0] NC

3.7.3.11 ADVS_DIGCORE

- **Name:** ADVS_DIGCORE Register
- **Description:** ADVS_DIGCORE Register
- **Base Address:** 0x4000A800
- **Offset:** 0x60
- **Reset Value:** 0x00004340

Table 3-120 ADVS_DIGCORE Register

Bits	Field Name	RW	Reset	Description
31:16	RSVD	R	0x0	Rserved bits
15:8	DVS_DIGCORE_REG2	RW	0x43	[7:0] NC
7:0	DVS_DIGCORE_REG1	RW	0x40	[7] NC [6] Enable AVS mode

Bits	Field Name	RW	Reset	Description
				[5] Controls the slope (1 = higher slope) [4] Enables lower limiter function to prevent the AVS from dropping below a set limit [3:0] NC

3.7.3.12 PMU_LPD_CFG

- **Name:** PMU_LPD_CFG Register
- **Description:** PMU_LPD_CFG Register
- **Base Address:** 0x4000A800
- **Offset:** 0x90
- **Reset Value:** 0x00000027

Table 3-121 PMU_LPD_CFG Register

Bits	Field Name	RW	Reset	Description
31:7	RSVD	R	0x0	Rserved bits
6:4	VAON_SLEEP	RW	0x2	LPD value when MCU is OFF 0.75/.80/.85/.90/.95/1.0/1.05/1.1
3	RSVD	R	0x0	Rserved bit
2:0	VAON_ACTIVE	RW	0x7	LPD value when MCU is ON 0.75/.80/.85/.90/.95/1.0/1.05/1.1

3.7.3.13 PMU_TON_CFG

- **Name:** PMU_TON_CFG Register
- **Description:** PMU_TON_CFG Register
- **Base Address:** 0x4000A800
- **Offset:** 0x94
- **Reset Value:** 0x00000303

Table 3-122 PMU_TON_CFG Register

Bits	Field Name	RW	Reset	Description
31:19	RSVD	R	0x0	Rserved bits
11:8	OFF_VAL	RW	0x3	ton_ctrl value when TXEN = 0
7:4	RSVD	R	0x0	Rserved bits

Bits	Field Name	RW	Reset	Description
3:0	ON_VAL	RW	0x3	ton_ctrl value when TXEN = 1

3.7.3.14 PMU_CORE_LDO_VREF

- **Name:** PMU_CORE_LDO_VREF Register
- **Description:** PMU_CORE_LDO_VREF Register
- **Base Address:** 0x4000A800
- **Offset:** 0x98
- **Reset Value:** 0x00001801

Table 3-123 PMU_CORE_LDO_VREF Register

Bits	Field Name	RW	Reset	Description
31:13	RSVD	R	0x0	Rserved bits
12:8	VREF	RW	0x18	The core_ldo_vref value to PMU when SEL=1 4:1 trimming -8~+8% 1% per step (in 1V level 13mV/step)
7:1	RSVD	R	0x0	Rserved bits
0	SEL	RW	0x1	0: use the core_ldo_vref value from MCU domain, 1: use the register value

3.7.3.15 PMU_DCDC_VREF

- **Name:** PMU_DCDC_VREF Register
- **Description:** PMU_DCDC_VREF Register
- **Base Address:** 0x4000A800
- **Offset:** 0x9C
- **Reset Value:** 0x00000013

Table 3-124 PMU_DCDC_VREF Register

Bits	Field Name	RW	Reset	Description
31:5	RSVD	R	0x0	Rserved bits
4:0	REG_VAL	RW	0x13	The dcdc_vref value to PMU

3.7.3.16 PMU_INTF_OVR_EN_0

- **Name:** PMU_INTF_OVR_EN_0 Register
- **Description:** PMU_INTF_OVR_EN_0 Register

- **Base Address:** 0x4000A800
- **Offset:** 0xC0
- **Reset Value:** 0x00000000

Table 3-125 PMU_INTF_OVR_EN_0 Register

Bits	Field Name	RW	Reset	Description
31:8	RSVD	R	0x0	Rserved bits
7	RNG_OSC_CLK_EN	RW	0x0	Value for RNG_OSC_CLK overwrite interface
6	MEM_RET_BOOST	RW	0x0	Value for MEM_RET_BOOST overwrite interface
5	IO_LDO_EN	RW	0x0	Value for IO_LDO overwrite interface
4	DIG_PWR_SW	RW	0x0	Value for DIG_PWR_SW overwrite interface
3	DIG_LDO_EN	RW	0x0	Value for dig_ldo overwrite interface
2	HF_OSC_EN	RW	0x0	Value for hf_osc overwrite interface
1	RSVD	R	0x0	Rserved bit
0	DCDC_EN	RW	0x0	Value for dcdc overwrite interface

3.7.3.17 PMU_INTF_OVR_VAL_0

- **Name:** PMU_INTF_OVR_VAL_0 Register
- **Description:** PMU_INTF_OVR_VAL_0 Register
- **Base Address:** 0x4000A800
- **Offset:** 0xC4
- **Reset Value:** 0x00000000

Table 3-126 PMU_INTF_OVR_VAL_0 Register

Bits	Field Name	RW	Reset	Description
31:8	RSVD	R	0x0	Rserved bits
7	RNG_OSC_CLK_EN	RW	0x0	Value for RNG_OSC_CLK overwrite interface
6	MEM_RET_BOOST	RW	0x0	Value for MEM_RET_BOOST overwrite interface
5	IO_LDO_EN	RW	0x0	Value for io_ldo overwrite interface
4	DIG_PWR_SW	RW	0x0	Value for DIG_PWR_SW overwrite interface
3	DIG_LDO_EN	RW	0x0	Value for dig_ldo overwrite interface
2	HF_OSC_EN	RW	0x0	Value for hf_osc overwrite interface
1	RSVD	R	0x0	Rserved bit
0	DCDC_EN	RW	0x0	Value for dcdc overwrite interface

3.7.3.18 PMU_INTF_OVR_RD_0

- **Name:** PMU_INTF_OVR_RD_0 Register
- **Description:** PMU_INTF_OVR_RD_0 Register
- **Base Address:** 0x4000A800
- **Offset:** 0xE0
- **Reset Value:** 0x00000000

Table 3-127 PMU_INTF_OVR_RD_0 Register

Bits	Field Name	RW	Reset	Description
31:8	RSVD	R	0x0	Rserved bits
7	RNG_OSC_CLK_EN	R	0x0	Current RNG_OSC_CLK value
6	MEM_RET_BOOST	R	0x0	Current MEM_RET_BOOST value
5	IO_LDO_EN	R	0x0	Current io_ldo value
4	DIG_PWR_SW	R	0x0	Current DIG_PWR_SW value
3	DIG_LDO_EN	R	0x0	Current dig_ldo value
2	HF_OSC_EN	R	0x0	Current hf_osc_en value
1	RSVD	R	0x0	Rserved bit
0	DCDC_EN	R	0x0	Current dcdc value

3.7.4 Always-on RF Control

3.7.4.1 RF5

- **Name:** RF5 Register
- **Description:** RF5 Register
- **Base Address:** 0x4000A900
- **Offset:** 0x0
- **Reset Value:** 0x00000000

Table 3-128 RF5 Register

Bits	Field Name	RW	Reset	Description
31:24	RSVD	R	0x0	
23:16	MUX_REG2	RW	0x0	[7] if 0 --> 1MHz clock, if 1 --> 2MHz [6] 0: TPP 1: TPN (if tpc_ena = 0) , if tpc_ena = 1 --> dda_out [5] 1: sel ADC MUXin, (Note need to disable original MUX, mux_reg1<7> = 0)

Bits	Field Name	RW	Reset	Description
				[4] NC [3:2] adcmux_vref_prog 11,00: Vref=2V, 10:1V, 01:4V [1] tstmux_comp_en [0] ADCMUX_en
15:8	RSVD	R	0x0	
7:0	MUX_REG1	RW	0x0	[7] 1: Enable test MUX; 0: tri-state [6:4] block selection [3] 1: TPP; 0: TPN [2:0] signal selection inside block

3.7.4.2 RF6

- **Name:** RF6 Register
- **Description:** RF6 Register
- **Base Address:** 0x4000A900
- **Offset:** 0x4
- **Reset Value:** 0x384248B0

Table 3-129 RF6 Register

Bits	Field Name	RW	Reset	Description
31:0	CPLL_REG1	RW	0x384248B0	[31] pll_lock_det_en [30] NC [29:28] Pre division before CP (00- no division / 11 divide by 16) [27:23] CPLL loop filter third capacitor control - 2.5pF max [22:18] CPLL loop filter second capacitor control - 2.5pF max [17:13] CPLL loop filter first capacitor control - 40pF max [12:8] CPLL loop filter resistor control - 300kohm max [7:6] Clock freq. doubler delay control - to attain 50% duty cycle [5] current source select - 1= from bandgap / 0= from gm bias cell [4] current source select - 1= from bandgap / 0= from gm bias cell [3] Clock freq. doubler enable [2] CPLL VCO enable

Bits	Field Name	RW	Reset	Description
				[1] 1: cpll_rst_n = 0, 0: cpll_rst_n follows sequencer [0] MUX control select (0 = CPLL CLK out / 1=Doubler CLK out)

3.7.4.3 RF7

- **Name:** RF7 Register
- **Description:** RF7 Register
- **Base Address:** 0x4000A900
- **Offset:** 0x8
- **Reset Value:** 0x05808492

Table 3-130 RF7 Register

Bits	Field Name	RW	Reset	Description
31:0	CPLL_REG2	RW	0x05808492	[31:30] TH [29:28] TL [27:26] extra divider after the "m_div", to increase DPLL counting accuracy [25:18] DPLL ideal count [17] MUX control to select the source of the KVCO word (0 = DPLL engine; 1 = external word) [16:12] coarse tuning bits for VCO to set the mid frequency near the locking frequency [11:9] charge pump code programmability (000-5 µA to 111-40 µA); default: 15 µA [8:6] CPLL VCO KVCO control; default: 96 MHz/V [5:0] feedback divider; default: divided by 48

3.7.4.4 RF8

- **Name:** RF8 Register
- **Description:** RF8 Register
- **Base Address:** 0x4000A900
- **Offset:** 0xC
- **Reset Value:** 0x4F26F780

Table 3-131 RF8 Register

Bits	Field Name	RW	Reset	Description
31:0	XO_REG1	RW	0x4F26F780	[31:28] Comparator reference when to disable the Chirp noise injection [27:24] Control the strength of chirp buffers to limit crystal over drive [23:20] Chirp oscillator KVCO and center frequency control - 0001= highest freq and KVCO [19:17] Chirp oscillator ramp control , 111- slowest / 000- Fastest [16] doubles the current of the biasing circuit [15] Noise injection to be enabled in SU sequence [14] Neg cap generator to be enabled in SU sequence [13] needs to transition to high with xo_enable if you want fast SU [12:8] CK_LDO control bits [7] Use the CK_LDO as a switch (Bypass regulation) [6] Enables internal load for stability [5] doubles the current of the biasing circuit [4:0] NC

3.7.4.5 RF9

- **Name:** RF9 Register
- **Description:** RF9 Register
- **Base Address:** 0x4000A900
- **Offset:** 0x10
- **Reset Value:** 0x25B40096

Table 3-132 RF9 Register

Bits	Field Name	RW	Reset	Description
31:0	XO_REG2	RW	0x25B40096	[31] rc_cal_ena [30] rc_cal_reset_b [29:28] rc_cal_prediv [27:20] rc_cal_n_ideal_count [19] rc_cal_override [18:10] Cload programmability from 50 fF to 26 pF on each side [9:5] number of Xtal cycles for frequency settling after all SU sequences are done <11>: 128 µs; <10>: 64 µs; <9>: 32 µs; <8>: 16 µs; <7>: 8 µs; default: 32 µs [4:0] number of Xtal cycles before disconnecting negative cap <11>: 128 µs; <10>: 64 µs; <9>: 32 µs; <8>: 16 µs; <7>: 8 µs; default: 160 µs

3.7.4.6 RF10

- **Name:** RF10 Register
- **Description:** RF10 Register
- **Base Address:** 0x4000A900
- **Offset:** 0x14
- **Reset Value:** 0x00000004

Table 3-133 RF10 Register

Bits	Field Name	RW	Reset	Description
31:8	RSVD	R	0x0	Reserved bits

Bits	Field Name	RW	Reset	Description
7:0	CPLL_REG3	RW	0x4	[7:5] NC [4] su_vco_test_en [3:1] cpll_vco_bias_sel [0] cpll_kvco_dig_ctrl

3.7.4.7 RF_RD_REG_0

- **Name:** RF_RD_REG_0 Register
- **Description:** RF_RD_REG_0 Register
- **Base Address:** 0x4000A900
- **Offset:** 0x40
- **Reset Value:** 0x00000038

Table 3-134 RF_RD_REG_0 Register

Bits	Field Name	RW	Reset	Description
31:21	RSVD	R	0x0	Reserved bits
20:16	CPLL_CRSCDE	R	0x0	PLL frequency value
15:8	RSVD	R	0x0	Reserved bits
7:0	RF_ID	R	0x38	RF ID num

3.7.4.8 RF_XO_BIAS_VAL

- **Name:** RF_XO_BIAS_VAL Register
- **Description:** RF_XO_BIAS_VAL Register
- **Base Address:** 0x4000A900
- **Offset:** 0x68
- **Reset Value:** 0x40084008

Table 3-135 RF_XO_BIAS_VAL Register

Bits	Field Name	RW	Reset	Description
31	RSVD	R	0x0	Reserved bit
30:22	CAP_LO	RW	0x100	XO capacitance parameter
21:16	BIAS_LO	RW	0x8	XO bias parameter
15	RSVD	R	0x0	Reserved bit
14:6	CAP_HI	RW	0x100	XO capacitance parameter

Bits	Field Name	RW	Reset	Description
5:0	BIAS_HI	RW	0x8	XO bias parameter

3.7.4.9 RF_INTF_OVR_EN_0

- **Name:** RF_INTF_OVR_EN_0 Register
- **Description:** RF_INTF_OVR_EN_0 Register
- **Base Address:** 0x4000A900
- **Offset:** 0xC0
- **Reset Value:** 0x00000000

Table 3-136 RF_INTF_OVR_EN_0 Register

Bits	Field Name	RW	Reset	Description
31:24	RSVD	R	0x0	Reserved bits
3	RST_N	RW	0x0	Value for rst_n overwrite interface
2	CPLL_EN	RW	0x0	Value for cpll_en overwrite interface
1	TUNE_EN	RW	0x0	Value for tune_en overwrite interface
0	XO_EN	RW	0x0	Value for xo_en overwrite interface

3.7.4.10 RF_INTF_OVR_VAL_0

- **Name:** RF_INTF_OVR_VAL_0 Register
- **Description:** RF_INTF_OVR_VAL_0 Register
- **Base Address:** 0x4000A900
- **Offset:** 0xC4
- **Reset Value:** 0x00000000

Table 3-137 RF_INTF_OVR_VAL_0 Register

Bits	Field Name	RW	Reset	Description
31:24	RSVD	R	0x0	Reserved bits
3	RST_N	RW	0x0	Value for rst_n overwrite interface
2	CPLL_EN	RW	0x0	Value for cpll_en overwrite interface
1	TUNE_EN	RW	0x0	Value for tune_en overwrite interface
0	XO_EN	RW	0x0	Value for xo_en overwrite interface

3.7.4.11 RF_INTF_OVR_RD_0

- **Name:** RF_INTF_OVR_RD_0 Register
- **Description:** RF_INTF_OVR_RD_0 Register
- **Base Address:** 0x4000A900
- **Offset:** 0xC8
- **Reset Value:** 0x00000000

Table 3-138 RF_INTF_OVR_RD_0 Register

Bits	Field Name	RW	Reset	Description
31:24	RSVD	R	0x0	Reserved bits
3	RST_N	R	0x0	Current rst_n value. When RF_INTF_OVR_EN_0[3] = 0x1, the value equals RF_INTF_OVR_VAl_0[3]; when RF_INTF_OVR_EN_0[3] = 0x0, the value is controlled by software.
2	CPLL_EN	R	0x0	Current cpll_en value
1	TUNE_EN	R	0x0	Current tune_en value
0	XO_EN	R	0x0	Current xo_en value

3.7.4.12 RF_INTF_OVR_EN_1

- **Name:** RF_INTF_OVR_EN_1 Register
- **Description:** RF_INTF_OVR_EN_1 Register
- **Base Address:** 0x4000A900
- **Offset:** 0xCC
- **Reset Value:** 0x00000000

Table 3-139 RF_INTF_OVR_EN_1 Register

Bits	Field Name	RW	Reset	Description
31:1	RSVD	R	0x0	Reserved bits
0	XO_BIAS_SW	RW	0x0	Value for xo_bias overwrite interface

3.7.4.13 RF_INTF_OVR_VAL_1

- **Name:** RF_INTF_OVR_VAL_1 Register
- **Description:** RF_INTF_OVR_VAL_1 Register
- **Base Address:** 0x4000A900
- **Offset:** 0xD0

- **Reset Value:** 0x00000000

Table 3-140 RF_INTF_OVR_VAL_1 Register

Bits	Field Name	RW	Reset	Description
31:1	RSVD	R	0x0	Reserved bits
0	XO_BIAS_SW	RW	0x0	Value for xo_bias overwrite interface

3.7.4.14 RF_INTF_OVR_RD_1

- **Name:** RF_INTF_OVR_RD_1 Register
- **Description:** RF_INTF_OVR_RD_1 Register
- **Base Address:** 0x4000A900
- **Offset:** 0xD4
- **Reset Value:** 0x00000000

Table 3-141 RF_INTF_OVR_RD_1 Register

Bits	Field Name	RW	Reset	Description
31:1	RSVD	R	0x0	Reserved bits
0	XO_BIAS_SW	R	0x0	Value for xo_bias overwrite interface

3.7.5 Cache

3.7.5.1 CACHE_REGS

3.7.5.1.1 CTRL0

- **Name:** CTRL0 Register
- **Description:** CTRL0 Register
- **Base Address:** 0x4000D000
- **Offset:** 0x0
- **Reset Value:** 0x00000013

Table 3-142 CTRL0 Register

Bits	Field Name	RW	Reset	Description
31:12	RSVD	R	0x0	Reserved bits
11	DIRECT_MAP_EN	RW	0x0	Enable direct map. **Value:** • 0x0: Enable 4-way associative map.

Bits	Field Name	RW	Reset	Description
				• 0x1: Enable direct map (no effect for sequential access).
10	CLK_FORCE_EN_3	RW	0x0	Cache_FSM and cache memory clock gating **Value:** • 0x0: Forcibly disable gating. • 0x1: Enable gating dynamically.
9	CLK_FORCE_EN_2	RW	0x0	APB clock gating control **Value:** • 0x0: Forcibly disable gating. • 0x1: Enable gating dynamically.
8	CLK_FORCE_EN_1	RW	0x0	Cache CLK_96 gating control **Value:** • 0x0: Forcibly disable gating. • 0x1: Enable gating dynamically.
7	CLK_FORCE_EN_0	RW	0x0	XQSPI and cache HCLK gating control **Value:** • 0x0: Forcibly disable gating. • 0x1: Enable gating dynamically.
6	BUF_DIS	RW	0x0	Disable cache buffer preloading. **Value:** • 0x0: Enable cache preloading. • 0x1: Disable cache preloading.
5	DIS_SEQ	RW	0x0	Disable sequential cache access. **Value:** • 0x0: Enable cache sequential access. • 0x1: Disable cache sequential access.
4	HITMISS	RW	0x1	Clear hit/miss counters. **Value:** • 0x0: hit and miss counters in normal mode • 0x1: hit and miss counters in clear mode (default)
3	FIFO	RW	0x0	Clear LFU FIFO. **Value:** • 0x0: FIFO in normal mode (default) • 0x1: FIFO in clear mode
2	RSVD	R	0x0	Reserved bits

Bits	Field Name	RW	Reset	Description
1	FLUSH	RW	0x1	Enable tag memory flushing. Out of resetting tag memory will get flushed unless tag memory retention is enabled (tag_ret signal lives in CPU register space). **Value:** • 0x0: Tag memory flushing is disabled. • 0x1: Tag memory flushing is enabled.
0	DIS	RW	0x1	Disable cache. **Value:** • 0x0: Enable cache. • 0x1: Disable cache disabled (default).

3.7.5.1.2 HIT_COUNT

- **Name:** HIT_COUNT Register
- **Description:** HIT_COUNT Register
- **Base Address:** 0x4000D000
- **Offset:** 0x8
- **Reset Value:** 0x00000000

Table 3-143 HIT_COUNT Register

Bits	Field Name	RW	Reset	Description
31:0	HITCOUNT	R	0x0	Read cache hit count (if bit[4] of CTRL0 is 0).

3.7.5.1.3 MISS_COUNT

- **Name:** MISS_COUNT Register
- **Description:** MISS_COUNT Register
- **Base Address:** 0x4000D000
- **Offset:** 0xC
- **Reset Value:** 0x00000000

Table 3-144 MISS_COUNT Register

Bits	Field Name	RW	Reset	Description
31:0	MISSCOUNT	R	0x0	Read cache miss count (if bit[4] of CTRL0 is 0).

3.7.5.1.4 STAT

- **Name:** STAT Register
- **Description:** STAT Register
- **Base Address:** 0x4000D000
- **Offset:** 0x10
- **Reset Value:** 0x00000000

Table 3-145 STAT Register

Bits	Field Name	RW	Reset	Description
31:1	RSVD	R	0x0	Reserved bits
0	STAT	R	0x0	**Value:** • 0x0: tag flushing not busy • 0x1: tag flushing busy

3.7.5.2 QSPI_REGS

3.7.5.2.1 TX_DATA

- **Name:** TX_DATA Register
- **Description:** TX_DATA Register
- **Base Address:** 0x4000D400
- **Offset:** 0x0
- **Reset Value:** 0x00000000

Table 3-146 TX_DATA Register

Bits	Field Name	RW	Reset	Description
31:0	TX	W	0x0	Serial transmit data register No packing of transmit data is attempted. That is, the number of bits transmitted per FIFO element is determined by the “bitsize” value in the Control Register (e.g. bitsize = 010 means only 12 bits are used). In addition, the “msb1st” value in the Control Register determines whether the least-significant bits or the most-significant bits of the FIFO element are used.

3.7.5.2.2 RX_DATA

- **Name:** RX_DATA Register
- **Description:** RX_DATA Register
- **Base Address:** 0x4000D400
- **Offset:** 0x4
- **Reset Value:** 0x00000000

Table 3-147 RX_DATA Register

Bits	Field Name	RW	Reset	Description
31:0	RX	R	0x0	Serial receive data register No packing of receive data is attempted. That is, the number of bits stored per FIFO element is determined by the "bitsize" value in the Control Register (e.g. bitsize = 011 means only 16 bits are used).

3.7.5.2.3 CTRL

- **Name:** CTRL Register
- **Description:** CTRL Register
- **Base Address:** 0x4000D400
- **Offset:** 0xC
- **Reset Value:** 0x00000021

Table 3-148 CTRL Register

Bits	Field Name	RW	Reset	Description
31:16	RSVD	R	0x0	Reserved bits
15:14	TXWMARK	RW	0x0	**Value:** • 0x0: QSPI_FIFO_DEPTH/8 • 0x1: QSPI_FIFO_DEPTH/4 • 0x2: QSPI_FIFO_DEPTH/2 • 0x3: QSPI_FIFO_DEPTH/2 + QSPI_FIFO_DEPTH/4
13:12	RXWMARK	RW	0x0	**Value:** • 0x0: QSPI_FIFO_DEPTH/8 • 0x1: QSPI_FIFO_DEPTH/4 • 0x2: QSPI_FIFO_DEPTH/2 • 0x3: QSPI_FIFO_DEPTH/2 + QSPI_FIFO_DEPTH/4
11	MWAITEN	RW	0x0	**Value:**

Bits	Field Name	RW	Reset	Description
				• 0x0: Disable delay (full speed). • 0x1: Enable delay.
10	DMA	RW	0x0	Not used **Value:** • 0x0: Disable DMA mode. • 0x1: Enable DMA mode.
9:6	RSVD	R	0x0	Reserved bits
5	MASTER	R	0x1	qspi_master flag
4	CPOL	RW	0x0	**Value:** • 0x0: SPI clock rests low. • 0x1: SPI clock rests high.
3	CPHA	RW	0x0	**Value:** • 0x0: First bit(s) of sdataOut occur as SS is asserted. • 0x1: sdataOut occurs on first SPI clock edge after SS
2	MSB1ST	RW	0x0	MSB/LSB • 0x1: MSB first • 0x0: LSB first
1	RSVD	R	0x0	Reserved bits
0	CONTXFER	RW	0x1	Continuous Transfer bit. This is relevant for Master operation only. Note that this setting also affects the timing of the master output enable signal, sdataOutEnMN. **Value:** • 0x0: ssOut goes inactive between successive transfers. • 0x1: ssOut stays active until the TX FIFO is empty AND the contXferExtend bit (in the Auxiliary Control Register) is low.

3.7.5.2.4 AUX_CTRL

- **Name:** AUX_CTRL Register
- **Description:** AUX_CTRL Register
- **Base Address:** 0x4000D400
- **Offset:** 0x10
- **Reset Value:** 0x00000070

Table 3-149 AUX_CTRL Register

Bits	Field Name	RW	Reset	Description
31:8	RSVD	R	0x0	Reserved bits
7	CONTXFERX	RW	0x0	Continuous Transfer Extend Bit. This is relevant for Master operation only. Further, this bit has no effect if the contXfer bit is 0. This register bit is synchronized to the SCLK domain for use there. Note that this setting also affects the timing of the master output enable signal, sdataOutEnMN. **Value:** • 0x0: Set to 0 to have ssOut[x] become inactive when the TX FIFO is empty. • 0x1: Set to 1 to have ssOut[x] remain active even when the TX FIFO is empty.
6:4	BITSIZE	RW	0x7	**Value:** • 0x0: 4-bit serial mode • 0x1: 8-bit serial mode • 0x2: 12-bit serial mode • 0x3: 16-bit serial mode • 0x4: 20-bit serial mode • 0x5: 24-bit serial mode • 0x6: 28-bit serial mode • 0x7: 32-bit serial mode
3	INHIBITDIN	RW	0x0	Set to 1 to inhibit the SPI serializers from writing to the Read Data FIFO.
2	INHIBITDOUT	RW	0x0	Set to 1 to inhibit the SPI serializers from reading the Transmit FIFO. Note that this is intended to be used as a Slave mode configuration. In many cases, a Slave simply receives data from a Master; in these cases, setting this bit will allow the Slave to receive data without causing a TX FIFO underflow condition.
1:0	QMODE	RW	0x0	**Value:** • 0x0: normal SPI • 0x1: reserved • 0x2: dual SPI

Bits	Field Name	RW	Reset	Description
				• 0x3: quad SPI

3.7.5.2.5 STAT

- **Name:** STAT Register
- **Description:** STAT Register
- **Base Address:** 0x4000D400
- **Offset:** 0x14
- **Reset Value:** 0x0000002C

Table 3-150 STAT Register

Bits	Field Name	RW	Reset	Description
31:8	RSVD	R	0x0	Reserved bits
7	RXFULL	R	0x0	**Value:** • 0x0: RX FIFO is not full. • 0x1: RX FIFO is full.
6	RXWMARK	R	0x0	**Value:** • 0x0: The received data is less than the watermark. • 0x1: The received data is more than or equal to the watermark.
5	RXEMPTY	R	0x1	**Value:** • 0x0: RX FIFO is not empty. • 0x1: RX FIFO is empty.
4	TXFULL	R	0x0	**Value:** • 0x0: TX FIFO is not full. • 0x1: TX FIFO is full.
3	TXWMARK	R	0x1	**Value:** • 0x0: The data in TX FIFO is more than the watermark. • 0x1: The data in TX FIFO is less than or equal to the watermark.
2	TXEMPTY	R	0x1	**Value:** • 0x0: TX FIFO is not empty. • 0x1: TX FIFO is empty.
1	RSVD	R	0x0	Reserved bits
0	XFERIP	R	0x0	Transfer is in progress (ssOut/ssIn is active).

Bits	Field Name	RW	Reset	Description
				Value: • 0x0: CS is inactive. • 0x1: CS is active.

3.7.5.2.6 SLAVE_SEL

- **Name:** SLAVE_SEL Register
- **Description:** SLAVE_SEL Register
- **Base Address:** 0x4000D400
- **Offset:** 0x18
- **Reset Value:** 0x00000000

Table 3-151 SLAVE_SEL Register

Bits	Field Name	RW	Reset	Description
31:4	RSVD	R	0x0	Reserved bits
3	OUT3	RW	0x0	**Value:** • 0x0: Do not select Slave 1 for TX/RX. • 0x1: Select Slave 1 for TX/RX. **Note:** Only bit0 is used.
2	OUT2	RW	0x0	**Value:** • 0x0: Do not select Slave 1 for TX/RX. • 0x1: Select Slave 1 for TX/RX. **Note:** Only bit0 is used.
1	OUT1	RW	0x0	**Value:** • 0x0: Do not select Slave 1 for TX/RX. • 0x1: Select Slave 1 for TX/RX. **Note:** Only bit0 is used.
0	OUT0	RW	0x0	**Value:** • 0x0: Do not select Slave 1 for TX/RX. • 0x1: Select Slave 1 for TX/RX.

Bits	Field Name	RW	Reset	Description
				Note: Only bit0 is used.

3.7.5.2.7 SLAVE_SEL_POL

- **Name:** SLAVE_SEL_POL Register
- **Description:** SLAVE_SEL_POL Register
- **Base Address:** 0x4000D400
- **Offset:** 0x1C
- **Reset Value:** 0x00000000

Table 3-152 SLAVE_SEL_POL Register

Bits	Field Name	RW	Reset	Description
31:4	RSVD	R	0x0	Reserved bits
3	POL3	RW	0x0	**Value:** • 0x0: CS is an active low signal. • 0x1: CS is an active high signal. **Note:** Only bit0 is used.
2	POL2	RW	0x0	**Value:** • 0x0: CS is an active low signal. • 0x1: CS is an active high signal. **Note:** Only bit0 is used.
1	POL1	RW	0x0	**Value:** • 0x0: CS is an active low signal. • 0x1: CS is an active high signal. **Note:** Only bit0 is used.
0	POL0	RW	0x0	**Value:** • 0x0: CS is an active low signal. • 0x1: CS is an active high signal. **Note:**

Bits	Field Name	RW	Reset	Description
				Only bit0 is used.

3.7.5.2.8 INTEN

- **Name:** INTEN Register
- **Description:** INTEN Register
- **Base Address:** 0x4000D400
- **Offset:** 0x20
- **Reset Value:** 0x00000000

Table 3-153 INTEN Register

Bits	Field Name	RW	Reset	Description
31:7	RSVD	R	0x0	Reserved bits
6:0	INT_EN	RW	0x0	[0] TX FIFO empty interrupt [1] TX FIFO watermark interrupt [2] RX FIFO watermark interrupt [3] RX FIFO full interrupt [4] ro_xferip from high to low [5] no use [6] no use **Value:** • 0x0: Disable • 0x1: Enable

3.7.5.2.9 INTSTAT

- **Name:** INTSTAT Register
- **Description:** INTSTAT Register
- **Base Address:** 0x4000D400
- **Offset:** 0x24
- **Reset Value:** 0x00000000

Table 3-154 INTSTAT Register

Bits	Field Name	RW	Reset	Description
31:5	RSVD	R	0x0	Reserved bits
4:0	INT_STAT	R	0x0	[0] TX FIFO empty flag interrupt

Bits	Field Name	RW	Reset	Description
				[1] TX FIFO watermark interrupt [2] RX FIFO watermark interrupt [3] RX FIFO full interrupt [4] CS done interrupt **Value:** • 0x0: no interrupt • 0x1: Interrupt source is active.

3.7.5.2.10 INTCLR

- **Name:** INTCLR Register
- **Description:** INTCLR Register
- **Base Address:** 0x4000D400
- **Offset:** 0x28
- **Reset Value:** 0x00000000

Table 3-155 INTCLR Register

Bits	Field Name	RW	Reset	Description
31:5	RSVD	R	0x0	Reserved bits
4:0	INT_CLR	W	0x0	[0] Clear the TX FIFO empty flag interrupt. [1] Clear the TX FIFO watermark interrupt. [2] Clear the RX FIFO watermark interrupt. [3] Clear the RX FIFO full interrupt. [4] Clear the CS done interrupt. **Value:** • 0x0: no effect • 0x1: Clear the interrupt,

3.7.5.2.11 TX_FIFO_LVL

- **Name:** TX_FIFO_LVL Register
- **Description:** TX_FIFO_LVL Register
- **Base Address:** 0x4000D400
- **Offset:** 0x2C
- **Reset Value:** 0x00000000

Table 3-156 TX_FIFO_LVL Register

Bits	Field Name	RW	Reset	Description
31:5	RSVD	R	0x0	Reserved bits
4:0	TXFIFOLVL	R	0x0	Read the transmit FIFO current level (0–16).

3.7.5.2.12 RX_FIFO_LVL

- **Name:** RX_FIFO_LVL Register
- **Description:** RX_FIFO_LVL Register
- **Base Address:** 0x4000D400
- **Offset:** 0x30
- **Reset Value:** 0x00000000

Table 3-157 RX_FIFO_LVL Register

Bits	Field Name	RW	Reset	Description
31:5	RSVD	R	0x0	Reserved bits
4:0	RXFIFOLVL	R	0x0	Read the transmit FIFO current level (0–16).

3.7.5.2.13 MSTR_IT_DELAY

- **Name:** MSTR_IT_DELAY Register
- **Description:** MSTR_IT_DELAY Register
- **Base Address:** 0x4000D400
- **Offset:** 0x38
- **Reset Value:** 0x00000000

Table 3-158 MSTR_IT_DELAY Register

Bits	Field Name	RW	Reset	Description
31:8	RSVD	R	0x0	Reserved bits
7:0	MWAIT	RW	0x0	If master (Control Register) is 1, this register sets the inter-transfer delay for the master serializer. The delay is (mwait + 1) SCLK cycles.

3.7.5.2.14 SPIEN

- **Name:** SPIEN Register
- **Description:** SPIEN Register

- **Base Address:** 0x4000D400
- **Offset:** 0x3C
- **Reset Value:** 0x00000000

Table 3-159 SPIEN Register

Bits	Field Name	RW	Reset	Description
31:1	RSVD	R	0x0	Reserved bits
0	EN	RW	0x0	Enable SPI. **Value:** • 0x0: Disable SPI TX and RX; reset RX and TX FIFOs on the transition from 1 to 0 of this bit. Although SPI can be disabled at any time, it is recommended to disable SPI when FIFOs are empty. After writing 0 to the enable bit, read the enable bit back; only after the enable bit is read to be 0, should any further register configurations be performed. • 0x1: Enable SPI TX and RX.

3.7.5.2.15 SPI_GP_SET

- **Name:** SPI_GP_SET Register
- **Description:** SPI_GP_SET Register
- **Base Address:** 0x4000D400
- **Offset:** 0x40
- **Reset Value:** 0x00000000

Table 3-160 SPI_GP_SET Register

Bits	Field Name	RW	Reset	Description
31:0	RSVD	R	0x0	Reserved bits
7:0	SET	RW	0x0	Write: write 1's to set individual bits of spiGpo. Read: returns current state of spiGpo.

3.7.5.2.16 SPI_GP_CLEAR

- **Name:** SPI_GP_CLEAR Register
- **Description:** SPI_GP_CLEAR Register
- **Base Address:** 0x4000D400
- **Offset:** 0x44

- **Reset Value:** 0x00000000

Table 3-161 SPI_GP_CLEAR Register

Bits	Field Name	RW	Reset	Description
31:0	RSVD	R	0x0	Reserved bits
7:0	CLR	RW	0x0	Write: write 1's to clear individual bits of spiGpo. Read: returns current state of spiGpi[1:0]

3.7.5.2.17 RX_DATA0_31

- **Name:** RX_DATA0_31 Register
- **Description:** RX_DATA0_31 Register
- **Base Address:** 0x4000D400
- **Offset:** 0x48
- **Reset Value:** 0x00000000

Table 3-162 RX_DATA0_31 Register

Bits	Field Name	RW	Reset	Description
31:0	RX_DATA0	R	0x0	32-bit LSB word in RX 128-bit read mode, rx_data[31:0]

3.7.5.2.18 RX_DATA32_63

- **Name:** RX_DATA32_63 Register
- **Description:** RX_DATA32_63 Register
- **Base Address:** 0x4000D400
- **Offset:** 0x4C
- **Reset Value:** 0x00000000

Table 3-163 RX_DATA32_63 Register

Bits	Field Name	RW	Reset	Description
31:0	RX_DATA1	R	0x0	32-bit 2nd LSB word in RX 128-bit read mode, rx_data[63:32]

3.7.5.2.19 RX_DATA64_95

- **Name:** RX_DATA64_95 Register
- **Description:** RX_DATA64_95 Register
- **Base Address:** 0x4000D400

- **Offset:** 0x50
- **Reset Value:** 0x00000000

Table 3-164 RX_DATA64_95 Register

Bits	Field Name	RW	Reset	Description
31:0	RX_DATA2	R	0x0	32-bit 3rd LSB word in RX 128-bit read mode, rx_data[95:64]

3.7.5.2.20 RX_DATA96_127

- **Name:** RX_DATA96_127 Register
- **Description:** RX_DATA96_127 Register
- **Base Address:** 0x4000D400
- **Offset:** 0x54
- **Reset Value:** 0x00000000

Table 3-165 RX_DATA96_127 Register

Bits	Field Name	RW	Reset	Description
31:0	RX_DATA3	R	0x0	32-bit 4th LSB word in RX 128-bit read mode, rx_data[127:96]

3.7.5.2.21 FLASH_WRITE

- **Name:** FLASH_WRITE Register
- **Description:** FLASH_WRITE Register
- **Base Address:** 0x4000D400
- **Offset:** 0x5C
- **Reset Value:** 0x00000000

Table 3-166 FLASH_WRITE Register

Bits	Field Name	RW	Reset	Description
31:1	RSVD	R	0x0	Reserved bits
0	FLASH_WRITE	RW	0x0	**Value:** • 0x0: 128-bit mode • 0x1: 32-bit mode (used for Flash programming/write operation)

3.7.5.2.22 CS_IDLE_UNVLD_EN

- **Name:** CS_IDLE_UNVLD_EN Register
- **Description:** CS_IDLE_UNVLD_EN Register
- **Base Address:** 0x4000D400
- **Offset:** 0x78
- **Reset Value:** 0x00000001

Table 3-167 CS_IDLE_UNVLD_EN Register

Bits	Field Name	RW	Reset	Description
31:3	RSVD	R	0x0	Reserved bits
2	KEY_PULSE_DIS	RW	0x0	When CPU reads Flash alias address and Flash address, access to Flash alias address will flush XQSPI FIFO, thus resulting in low performance. **Value:** • 0x0: Enable key_pulse to interrupt rd_data state. • 0x1: Disable key_pulse to interrupt rd_data state.
1	1ST_PRETETCH_DIS	RW	0x0	When XIP_EN_REQ is set, XQSPI will prefetch 256 bytes of data from Flash. Clear this bit to disable this function. **Value:** • 0x0: Enable 1st prefecth function. • 0x1: Disable 1st prefetch function.
0	CS_IDLE_UNVLD_EN	RW	0x1	**Value:** • 0x0: CS keeps valid when the device is not reading data. • 0x1: CS keeps invalid when the device is not reading data.

3.7.5.3 XIP_REGS

3.7.5.3.1 CTRL0

- **Name:** CTRL0 Register
- **Description:** CTRL0 Register
- **Base Address:** 0x4000DC00
- **Offset:** 0x0
- **Reset Value:** 0x00000003

Table 3-168 CTRL0 Register

Bits	Field Name	RW	Reset	Description
31:8	RSVD	R	0x0	Reserved bits
7:0	CMD	RW	0x3	XIP configuration read command **Value:** • 0x03: read • 0x0B: fast read • 0x3B: fast dual out read • 0x6B: fast quad out read • 0xBB: fast dual I/O read • 0xEB: fast quad I/O read • All others: reserved

3.7.5.3.2 CTRL1

- **Name:** CTRL1 Register
- **Description:** CTRL1 Register
- **Base Address:** 0x4000DC00
- **Offset:** 0x4
- **Reset Value:** 0x00000108

Table 3-169 CTRL1 Register

Bits	Field Name	RW	Reset	Description
31:9	RSVD	R	0x0	Reserved bits
8	LE32	RW	0x1	XIP configuration 128-bit little endian arrangement of read data **Value:** • 0x0: The read data is in big endian mode. • 0x1: The read data is in 128-bit little endian mode.
7	ADDR4	RW	0x0	XIP configuration 4-byte address mode **Value:** • 0x0: XIP issues 3-byte address • 0x1: XIP issues 4-byte SPI address
6	CPOL	RW	0x0	XIP configuration CPOL mode **Value:**

Bits	Field Name	RW	Reset	Description
				• 0x0: CPOL mode 0 • 0x1: CPOL mode 1
5	CPHA	RW	0x0	XIP configuration CPHA mode **Value:** • 0x0: CPHA mode 0 • 0x1: CPHA mode 1
4:1	SS	RW	0x4	XIP configuration slave select **Value:** • 0x1: slave select 0 • 0x2: slave select 1 • 0x4: slave select 2 • 0x8: slave select 3 • All others: reserved
0	HPEN	RW	0x0	XIP configuration high performance mode **Value:** • 0x0: Disable HP mode. • 0x1: Enable HP mode.

3.7.5.3.3 CTRL2

- **Name:** CTRL2 Register
- **Description:** CTRL2 Register
- **Base Address:** 0x4000DC00
- **Offset:** 0x8
- **Reset Value:** 0x000000A3

Table 3-170 CTRL2 Register

Bits	Field Name	RW	Reset	Description
31:14	RSVD	R	0x0	Reserved bits
13:12	ENDDUMMY	RW	0x0	Configure the number of dummy cycles necessary to terminate high performance mode. This is used by the core when exiting XIP mode after performing high-performance transfers to ensure that FLASH device is not still in HP mode. For Quad mode, this value is multiplied by 2 to get the actual number of dummy cycles.

Bits	Field Name	RW	Reset	Description
11:8	DUMMYCYCLES	RW	0x0	Used to configure the number of dummy cycles used by the core for Fast Read, Fast Read Dual Output, and Fast Read Quad Output. • Fast Read Dual I/O: 4 x value + 4 • Fast Read Quad I/O: 2 x value + 2 • Fast Read Quad Output: 8 x value
7:0	HPMODE	RW	0xA3	XIP mode command byte. This byte is transferred from core to QSPI Flash memory. It is the value specified by different QSPI Flash memory vendors to enter into its status register to activate HP mode in dual I/O and Quad I/O accesses.

3.7.5.3.4 CTRL3

- **Name:** CTRL3 Register
- **Description:** CTRL3 Register
- **Base Address:** 0x4000DC00
- **Offset:** 0xC
- **Reset Value:** 0x00000000

Table 3-171 CTRL3 Register

Bits	Field Name	RW	Reset	Description
31:1	RSVD	R	0x0	Reserved bits
0	XIP_EN_REQ	RW	0x0	XIP enable request **Value:** • 0x0: Disable XIP mode. • 0x1: Enable XIP mode. When it is set to 1’b1, do not access QSPI address space. Wait until XIP_EN_OUT is 1’b1. When it is set to 1’b0, do not access QSPI address space. Wait until XIP_EN_OUT is 1’b0.

3.7.5.3.5 STAT

- **Name:** STAT Register
- **Description:** STAT Register
- **Base Address:** 0x4000DC00

- **Offset:** 0x10
- **Reset Value:** 0x00000000

Table 3-172 STAT Register

Bits	Field Name	RW	Reset	Description
31:1	RSVD	R	0x0	Reserved bits
0	XIP_EN_OUT	R	0x0	XIP enable status **Value:** • 0x0: XIP mode is disabled. • 0x1: XIP mode is enabled.

3.7.5.3.6 INTEN

- **Name:** INTEN Register
- **Description:** INTEN Register
- **Base Address:** 0x4000DC00
- **Offset:** 0x14
- **Reset Value:** 0x00000000

Table 3-173 INTEN Register

Bits	Field Name	RW	Reset	Description
31:1	RSVD	R	0x0	Reserved bits
0	XIP_INT_EN	R	0x0	Enable interrupt. **Value:** • 0x0: Disable interrupt. • 0x1: Enable interrupt.

3.7.5.3.7 INTSTAT

- **Name:** INTSTAT Register
- **Description:** INTSTAT Register
- **Base Address:** 0x4000DC00
- **Offset:** 0x18
- **Reset Value:** 0x00000000

Table 3-174 INTSTAT Register

Bits	Field Name	RW	Reset	Description
31:1	RSVD	R	0x0	Reserved bits
0	XIP_INT_STAT	R	0x0	Interrupt status **Value:** • 0x0: No interrupt occurred. • 0x1: An interrupt occurred.

3.7.5.3.8 INTREQ

- **Name:** INTREQ Register
- **Description:** INTREQ Register
- **Base Address:** 0x4000DC00
- **Offset:** 0x1C
- **Reset Value:** 0x00000000

Table 3-175 INTREQ Register

Bits	Field Name	RW	Reset	Description
31:1	RSVD	R	0x0	Reserved bits
0	XIP_INT_REQ	R	0x0	Output interrupt. **Value:** • 0x0: No interrupt occurred or interrupt is disabled. • 0x1: An interrupt occurred and interrupt is enabled.

3.7.5.3.9 INTSET

- **Name:** INTSET Register
- **Description:** INTSET Register
- **Base Address:** 0x4000DC00
- **Offset:** 0x20
- **Reset Value:** 0x00000000

Table 3-176 INTSET Register

Bits	Field Name	RW	Reset	Description
31:1	RSVD	R	0x0	Reserved bits
0	XIP_INT_SET	W	0x0	Interrupt enable set

Bits	Field Name	RW	Reset	Description
				Value: • 0x0: no action • 0x1: Set the related bit to 1.

3.7.5.3.10 INTCLR

- **Name:** INTCLR Register
- **Description:** INTCLR Register
- **Base Address:** 0x4000DC00
- **Offset:** 0x24
- **Reset Value:** 0x00000000

Table 3-177 INTCLR Register

Bits	Field Name	RW	Reset	Description
31:1	RSVD	R	0x0	Reserved bits
0	XIP_INT_CLR	W	0x0	Interrupt enable clear **Value:** • 0x0: no action • 0x1: Clear the related bit to 0.

3.7.5.3.11 SOFT_RST

- **Name:** SOFT_RST Register
- **Description:** SOFT_RST Register
- **Base Address:** 0x4000DC00
- **Offset:** 0x28
- **Reset Value:** 0x00000001

Table 3-178 SOFT_RST Register

Bits	Field Name	RW	Reset	Description
31:1	RSVD	R	0x0	Reserved bits
0	XIP_SOFT_RST	RW	0x0	Soft reset IP by writing 1’b0, and then internal logic generates active low reset for one HCLK cycle.

3.7.6 MCU Subsystem

3.7.6.1 MCU_SUB

3.7.6.1.1 CTE_FIFO_DATA

- **Name:** CTE_FIFO_DATA Register
- **Description:** CTE_FIFO_DATA Register
- **Base Address:** 0x4000E000
- **Offset:** 0x40
- **Reset Value:** 0x00000000

Table 3-179 CTE_FIFO_DATA Register

Bits	Field Name	RW	Reset	Description
31:0	OUT	R	0x0	Reading this register will “pop” an entry from the CTE FIFO

3.7.6.1.2 CTE_FIFO_THRESH

- **Name:** CTE_FIFO_THRESH Register
- **Description:** CTE_FIFO_THRESH Register
- **Base Address:** 0x4000E000
- **Offset:** 0x44
- **Reset Value:** 0x00000000

Table 3-180 CTE_FIFO_THRESH Register

Bits	Field Name	RW	Reset	Description
31:5	RSVD	R	0x0	Reserved bits
4:0	HOLD	RW	0x0	Used to set the threshold at which the DMA request is made to the hardware handshake mechanism

3.7.6.1.3 CTE_FIFO_STAT

- **Name:** CTE_FIFO_STAT Register
- **Description:** CTE_FIFO_STAT Register
- **Base Address:** 0x4000E000
- **Offset:** 0x48
- **Reset Value:** 0x00000000

Table 3-181 CTE_FIFO_STAT Register

Bits	Field Name	RW	Reset	Description
31:11	RSVD	R	0x0	Reserved bits
10	FULL	R	0x0	fifo full
9	RSVD	R	0x0	Reserved bits
8	VAL	R	0x0	When this bit is 1, the FIFO is not empty
7:6	RSVD	R	0x0	Reserved bits
5:0	COUNT	R	0x0	CTE ADC FIFO Count, number of 32 bits words in the FIFO.

3.7.6.1.4 CTE_CFG

- **Name:** CTE_CFG Register
- **Description:** CTE_CFG Register
- **Base Address:** 0x4000E000
- **Offset:** 0x4C
- **Reset Value:** 0x00000000

Table 3-182 CTE_CFG Register

Bits	Field Name	RW	Reset	Description
31:27	RSVD	R	0x0	Reserved bits
26:16	SAMP2CAP	RW	0x0	Number of CTE samples to capture
15:1	RSVD	R	0x0	Reserved bits
0	OVR_SAMP_ON	RW	0x0	1: oversampling is turned on

3.7.6.1.5 QSPI_M_XIP

- **Name:** QSPI_M_XIP Register
- **Description:** QSPI_M_XIP Register
- **Base Address:** 0x4000E000
- **Offset:** 0x180
- **Reset Value:** 0x00000000

Table 3-183 QSPI_M_XIP Register

Bits	Field Name	RW	Reset	Description
31	M2_DYNAMIC_LE_EN	RW	0x0	0 - read and write 32-bit without re-ordering in XIP mode

Bits	Field Name	RW	Reset	Description
				1 - Re-order the read and write data as [7:0], [15:8], [23:16], [31:24] when HSIZE is word. Re-order the read and write data as [23:16], [31:24], [7:0], [15:8] when HSIZE is half-word.
30	M1_DYNAMIC_LE_EN	RW	0x0	0 - read and write 32-bit without re-ordering in XIP mode 1 - Re-order the read and write data as [7:0], [15:8], [23:16], [31:24] when HSIZE is word. Re-order the read and write data as [23:16], [31:24], [7:0], [15:8] when HSIZE is half-word.
29	M0_DYNAMIC_LE_EN	RW	0x0	0 - read and write 32-bit without re-ordering in XIP mode 1 - Re-order the read and write data as [7:0], [15:8], [23:16], [31:24] when HSIZE is word. Re-order the read and write data as [23:16], [31:24], [7:0], [15:8] when HSIZE is half-word.
28:12	RSVD	R	0x0	Reserved bits
11:10	M2_ENDIAN_MODE	RW	0x0	0,3 - Read 32-bit without re-ordering 1 - Re-order the read data as [23:16], [31:24], [7:0], [15:8] 2 - Re-order the read data as [7:0], [15:8], [23:16], [31:24]
9	M2_XIP_EN_VAL	RW	0x0	When M2_XIP_EN_OVR=1: 0: normal mode, 1: XIP mode
8	M2_XIP_EN_OVR	RW	0x0	0: Automatically identify the XIP mode by address. 1: Configure the xip mode by M2_XIP_EN_VAL
7:6	M1_ENDIAN_MODE	RW	0x0	0,3 - Read 32-bit without re-ordering 1 - Re-order the read data as [23:16], [31:24], [7:0], [15:8] 2 - Re-order the read data as [7:0], [15:8], [23:16], [31:24]
5	M1_XIP_EN_VAL	RW	0x0	When M1_XIP_EN_OVR=1: 0: normal mode, 1: XIP mode
4	M1_XIP_EN_OVR	RW	0x0	0: Automatically identify the XIP mode by address. 1: Configure the xip mode by M1_XIP_EN_VAL
3:2	M0_ENDIAN_MODE	RW	0x0	0,3 - Read 32-bit without re-ordering 1 - Re-order the read data as [23:16], [31:24], [7:0], [15:8] 2 - Re-order the read data as [7:0], [15:8], [23:16], [31:24]
1	M0_XIP_EN_VAL	RW	0x0	When M0_XIP_EN_OVR=1: 0: normal mode, 1: XIP mode
0	M0_XIP_EN_OVR	RW	0x0	0: Automatically identify the XIP mode by address. 1: Configure the xip mode by M0_XIP_EN_VAL

3.7.6.1.6 QSPI_M_HRESP_DBG

- **Name:** QSPI_M_HRESP_DBG Register
- **Description:** QSPI_M_HRESP_DBG Register
- **Base Address:** 0x4000E000
- **Offset:** 0x184
- **Reset Value:** 0x00000001

Table 3-184 QSPI_M_HRESP_DBG Register

Bits	Field Name	RW	Reset	Description
31:1	RSVD	R	0x0	Reserved bits
0	ERR_MASK_EN	RW	0x1	1: force qspi_m hresp to 0 to avoid hard fault. 0: Will get hardfault when QSPI_M generates hresp=1

3.7.6.1.7 QSPI_M_CS_SETUP_DLY

- **Name:** QSPI_M_CS_SETUP_DLY Register
- **Description:** QSPI_M_CS_SETUP_DLY Register
- **Base Address:** 0x4000E000
- **Offset:** 0x188
- **Reset Value:** 0x00000000

Table 3-185 QSPI_M_CS_SETUP_DLY Register

Bits	Field Name	RW	Reset	Description
31:24	RSVD	R	0x0	Reserved bits
23:16	M2_CS_SETUP_DLY	RW	0x0	The number of clocks from cs negedge to sck
15:8	M1_CS_SETUP_DLY	RW	0x0	The number of clocks from cs negedge to sck
7:0	M0_CS_SETUP_DLY	RW	0x0	The number of clocks from cs negedge to sck

3.7.6.1.8 QSPI_M_CS_RELEASE_DLY

- **Name:** QSPI_M_CS_RELEASE_DLY Register
- **Description:** QSPI_M_CS_RELEASE_DLY Register
- **Base Address:** 0x4000E000
- **Offset:** 0x18C
- **Reset Value:** 0x00000000

Table 3-186 QSPI_M_CS_RELEASE_DLY Register

Bits	Field Name	RW	Reset	Description
31:24	RSVD	R	0x0	Reserved bits
23:16	M2_CS_RELEASE_DLY	RW	0x0	The number of clocks from sck to cs posedge
15:8	M1_CS_RELEASE_DLY	RW	0x0	The number of clocks from sck to cs posedge
7:0	M0_CS_RELEASE_DLY	RW	0x0	The number of clocks from sck to cs posedge

3.7.6.1.9 DMA0_HS_SEL

- **Name:** DMA0_HS_SEL Register
- **Description:** DMA0_HS_SEL Register
- **Base Address:** 0x4000E000
- **Offset:** 0x1C0
- **Reset Value:** 0x00000000

Table 3-187 DMA0_HS_SEL Register

Bits	Field Name	RW	Reset	Description
31:15	RSVD	R	0x0	Reserved bits
14	HS14_15_SEL	RW	0x0	Choice for handshake interface 14 & 15 0: choice 0 1: choice 1
13	RSVD	R	0x0	Reserved bits
12	HS12_13_SEL	RW	0x0	Choice for handshake interface 12 & 13 0: choice 0 1: choice 1
11	RSVD	R	0x0	Reserved bits
10	HS10_11_SEL	RW	0x0	Choice for handshake interface 10 & 11 0: choice 0 1: choice 1
9	RSVD	R	0x0	Reserved bits
8	HS8_9_SEL	RW	0x0	Choice for handshake interface 8 & 9 0: choice 0 1: choice 1
7	RSVD	R	0x0	Reserved bits
6	HS6_7_SEL	RW	0x0	Choice for handshake interface 6 & 7 0: choice 0

Bits	Field Name	RW	Reset	Description
				1: choice 1
5	RSVD	R	0x0	Reserved bits
4	HS4_5_SEL	RW	0x0	Choice for handshake interface 4 & 5 0: choice 0 1: choice 1
3	RSVD	R	0x0	Reserved bits
2	HS2_3_SEL	RW	0x0	Choice for handshake interface 2 & 3 0: choice 0 1: choice 1
1:0	RSVD	R	0x0	Reserved bits

3.7.6.1.10 DMA1_HS_SEL

- **Name:** DMA1_HS_SEL Register
- **Description:** DMA1_HS_SEL Register
- **Base Address:** 0x4000E000
- **Offset:** 0x1C4
- **Reset Value:** 0x00000000

Table 3-188 DMA1_HS_SEL Register

Bits	Field Name	RW	Reset	Description
31:15	RSVD	R	0x0	Reserved bits
14	HS14_15_SEL	RW	0x0	Choice for handshake interface 14 & 15 0: choice 0 1: choice 1
13:11	RSVD	R	0x0	Reserved bits
10	HS10_11_SEL	RW	0x0	Choice for handshake interface 10 & 11 0: choice 0 1: choice 1
9	RSVD	R	0x0	Reserved bits
8	HS8_9_SEL	RW	0x0	Choice for handshake interface 8 & 9 0: choice 0 1: choice 1
7	RSVD	R	0x0	Reserved bits
6	HS6_7_SEL	RW	0x0	Choice for handshake interface 6 & 7

Bits	Field Name	RW	Reset	Description
				0: choice 0 1: choice 1
5	RSVD	R	0x0	Reserved bits
4	HS4_5_SEL	RW	0x0	Choice for handshake interface 4 & 5 0: choice 0 1: choice 1
3	RSVD	R	0x0	Reserved bits
2	HS2_3_SEL	RW	0x0	Choice for handshake interface 2 & 3 0: choice 0 1: choice 1
1	RSVD	R	0x0	Reserved bits
0	HS0_1_SEL	RW	0x0	choice for handshake interface 0 & 1 0: choice 0 1: choice 1

3.7.6.1.11 PAD_IE_N_AUTO

- **Name:** PAD_IE_N_AUTO Register
- **Description:** PAD_IE_N_AUTO Register
- **Base Address:** 0x4000E000
- **Offset:** 0x220
- **Reset Value:** 0x00000000

Table 3-189 PAD_IE_N_AUTO Register

Bits	Field Name	RW	Reset	Description
31:4	RSVD	R	0x0	Reserved bits
3	AON_GPIO_EN	RW	0x0	(When MCU ON) 0: Use the value of AON_IE in AON 1: Automatically apply the invertion of the OE value.
2	MISC_EN	RW	0x0	0: Use the value of XQSPI & OSPI PADs, 1: Automatically apply the invertion of the OE value.
1	MSIO_EN	RW	0x0	(When MCU ON) 0: Use the value of MSIO_IE in AON 1: Automatically apply the invertion of the OEvalue.
0	DPAD_EN	RW	0x0	0: Use the value of DPAD_IE_BUS_0, 1: Automatically apply the invertion of the OE value.

3.7.6.1.12 BLE_FERP_CTL

- **Name:** BLE_FERP_CTL Register
- **Description:** BLE_FERP_CTL Register
- **Base Address:** 0x4000E000
- **Offset:** 0x224
- **Reset Value:** 0x00000000

Table 3-190 BLE_FERP_CTL Register

Bits	Field Name	RW	Reset	Description
31:24	RSVD	R	0x0	Reserved bits
23:20	MCU_DBG_FLAGSELH	RW	0x0	select of mcu_testbus[15:8] 1:mcu_testbus[15:8]= i2c_0_debug[15:8] 2:mcu_testbus[15:8]= i2c_1_debug[15:8] 3:mcu_testbus[15:8]= i2c_2_debug[15:8] 4:mcu_testbus[15:8]= i2c_3_debug[15:8] 5:mcu_testbus[15:8]= uart0_debug[15:8] 6:mcu_testbus[15:8]= uart1_debug[15:8] 7:mcu_testbus[15:8]= uart2_debug[15:8] 8:mcu_testbus[15:8]= uart3_debug[15:8] 9:mcu_testbus[15:8]= spi_m_debug[15:8] a:mcu_testbus[15:8]= spi_s_debug[15:8] b:mcu_testbus[7:0]= dma_dbg_0[15:8] c:mcu_testbus[7:0]= dma_dbg_1[15:8] d:mcu_testbus[7:0]= dma_dbg_2[15:8] e:mcu_testbus[7:0]= dma_dbg_3[15:8] f:mcu_testbus[7:0]= dma_dbg_4[15:8]
19:16	MCU_DBG_FLAGSELL	RW	0x0	select of mcu_testbus[7:0] 1:mcu_testbus[7:0]= i2c_0_debug[7:0] 2:mcu_testbus[7:0]= i2c_1_debug[7:0] 3:mcu_testbus[7:0]= i2c_2_debug[7:0] 4:mcu_testbus[7:0]= i2c_3_debug[7:0] 5:mcu_testbus[7:0]= uart0_debug[7:0] 6:mcu_testbus[7:0]= uart1_debug[7:0] 7:mcu_testbus[7:0]= uart2_debug[7:0] 8:mcu_testbus[7:0]= uart3_debug[7:0] 9:mcu_testbus[7:0]= spi_m_debug[7:0]

Bits	Field Name	RW	Reset	Description
				a:mcu_testbus[7:0]= spi_s_debug[7:0] b:mcu_testbus[7:0]= dma_dbg_0[7:0] c:mcu_testbus[7:0]= dma_dbg_1[7:0] d:mcu_testbus[7:0]= dma_dbg_2[7:0] e:mcu_testbus[7:0]= dma_dbg_3[7:0] f:mcu_testbus[7:0]= dma_dbg_4[7:0]
15	RSVD	R	0x0	Reserved bits
14	MCU_DBG_FLAGH_EN	RW	0x0	enable bit of mcu_testbus[15:8]
13	MCU_DBG_FLAGL_EN	RW	0x0	enable bit of mcu_testbus[7:0]
12	TESTBUS_REORDER_16_31	RW	0x0	reorder the bit field of TEST_BUS[31:16], 0: TEST_BUS[31:16] = {test_bus[31:24], test_bus[23:16]} 1: TEST_BUS[31:16]= {test_bus[23:16], test_bus[31:24]}
11	TESTBUS_REORDER_00_15	RW	0x0	reorder the bit field of TEST_BUS[15:0], 0: TEST_BUS[15:0] = {test_bus[15:8], test_bus[7:0]} 1: TEST_BUS[15:0]= {test_bus[7:0], test_bus[15:8]}
10:8	TESTBUS_SEL_16_31	RW	0x0	Select modes of test bus[31:16], repeat as testbus[15:0] 0: comm_proc[15:0] 1: xf_dbg[15:0] 2: clk_bus[15:0] 4:mcu_testbus[15:0]
7	RSVD	R	0x0	Reserved bits
6:4	TESTBUS_SEL_00_15	RW	0x0	Select modes of test bus[15:0] 0: comm_proc[15:0] 1: xf_dbg[15:0] 2: clk_bus[15:0], 3:mcu_testbus[15:0]
3:2	RSVD	R	0x0	Reserved bits
1:0	FERP_EN	RW	0x0	Enable ferp memory clock & ferp memory access in SoC side ferp_en[0] for 16KB sram1, i.e. addr range 0x00000000-0x00003fff from ferp engine ferp_en[1] for 16KB sram2, i.e. addr range 0x00004000-0x00007ffff from ferp engine

3.7.6.1.13 SECURITY_RESET

- **Name:** SECURITY_RESET Register

- **Description:** SECURITY_RESET Register
- **Base Address:** 0x4000E000
- **Offset:** 0x22C
- **Reset Value:** 0x0000007F

Table 3-191 SECURITY_RESET Register

Bits	Field Name	RW	Reset	Description
31:7	RSVD	R	0x0	Reserved bits
6	PRESENT	RW	0x1	PRESENT soft reset, low active **Value:** • 0x0: reset module • 0x1: normal work mode
5	TRNG	RW	0x1	TRNG soft reset, low active **Value:** • 0x0: reset module • 0x1: normal work mode
4	RAMKEY	RW	0x1	KEYRAM soft reset, low active **Value:** • 0x0: reset module • 0x1: normal work mode
3	EFUSE	RW	0x1	eFuse controller soft reset, low active **Value:** • 0x0: reset module • 0x1: normal work mode
2	PKC	RW	0x1	PKC soft reset, low active **Value:** • 0x0: reset module • 0x1: normal work mode
1	HMAC	RW	0x1	HMAC soft reset, low active **Value:** • 0x0: reset module • 0x1: normal work mode
0	AES	RW	0x1	AES soft reset, low active **Value:** • 0x0: reset module

Bits	Field Name	RW	Reset	Description
				• 0x1: normal work mode

3.7.6.1.14 PMU_ID

- **Name:** PMU_ID Register
- **Description:** PMU_ID Register
- **Base Address:** 0x4000E000
- **Offset:** 0x230
- **Reset Value:** 0x00000038

Table 3-192 PMU_ID Register

Bits	Field Name	RW	Reset	Description
31:8	RSVD	R	0x0	Reserved bits
7:0	PMU_ID	R	0x38	PMU chip ID

3.7.6.1.15 PWR_AVG_CTL_REG0

- **Name:** PWR_AVG_CTL_REG0 Register
- **Description:** PWR_AVG_CTL_REG0 Register
- **Base Address:** 0x4000E000
- **Offset:** 0x234
- **Reset Value:** 0x00000004

Table 3-193 PWR_AVG_CTL_REG0 Register

Bits	Field Name	RW	Reset	Description
31:24	TPA_ADC_OUT	R	0x0	Real-time value
18	AVG_PWR_ERR	R	0x0	Indicate ble_tx_en going off before outputting the result.
17	RSVD	R	0x0	Reserved bit
16	AVG_PWR_RDY	R	0x0	Indicate that the average value is ready to read.
15:8	AVG_PWR	R	0x0	Average power data
7:4	SAMPL_PWR	RW	0x0	Take 2 ^ n samples for averaging (0–15).
3	BLE_F_TX_EN	RW	0x0	Forcibly enable internal ble_tx_en.
2	ONESHOT_EN	RW	0x1	Enable one shot mode.
1	RSVD	R	0x0	Reserved bit
0	PWR_AVG_EN	RW	0x0	Enable the power average calculation block.

3.7.6.1.16 EFUSE_PWR_DELTA_0

- **Name:** EFUSE_PWR_DELTA_0 Register
- **Description:** EFUSE_PWR_DELTA_0 Register
- **Base Address:** 0x4000E000
- **Offset:** 0x254
- **Reset Value:** 0x0FFF00B4

Table 3-194 EFUSE_PWR_DELTA_0 Register

Bits	Field Name	RW	Reset	Description
31:16	DELTA1	RW	0xFFF	About 250 µs; an eFuse power controller timing parameter; based on 16 MHz clock
15:0	DELTA0	RW	0xB4	About 10 µs; an eFuse power controller timing parameter; based on 16 MHz clock

3.7.6.1.17 EFUSE_PWR_DELTA_1

- **Name:** EFUSE_PWR_DELTA_1 Register
- **Description:** EFUSE_PWR_DELTA_1 Register
- **Base Address:** 0x4000E000
- **Offset:** 0x258
- **Reset Value:** 0x000000B4

Table 3-195 EFUSE_PWR_DELTA_1 Register

Bits	Field Name	RW	Reset	Description
31:16	RSVD	R	0x0	Reserved bits
15:0	DELTA2	RW	0xB4	About 10 µs; an eFuse power controller timing parameter; based on 16 MHz clock

3.7.6.1.18 EFUSE_PWR_CTRL_0

- **Name:** EFUSE_PWR_CTRL_0 Register
- **Description:** EFUSE_PWR_CTRL_0 Register
- **Base Address:** 0x4000E000
- **Offset:** 0x260
- **Reset Value:** 0x00000000

Table 3-196 EFUSE_PWR_CTRL_0 Register

Bits	Field Name	RW	Reset	Description
31:5	RSVD	R	0x0	Reserved bits
4	STP	RW	0x0	Write 1 to stop the power sequencer.
3	RSVD	R	0x0	Reserved bit
2	BGN	RW	0x0	Write 1 to begin the power sequencer.
1	RSVD	R	0x0	Reserved bit
0	EN	RW	0x0	Enable the eFuse controller.

3.7.6.1.19 EFUSE_PWR_CTRL_1

- **Name:** EFUSE_PWR_CTRL_1 Register
- **Description:** EFUSE_PWR_CTRL_1 Register
- **Base Address:** 0x4000E000
- **Offset:** 0x264
- **Reset Value:** 0x00000000

Table 3-197 EFUSE_PWR_CTRL_1 Register

Bits	Field Name	RW	Reset	Description
31:5	RSVD	R	0x0	Reserved bits
4	DIS_DONE	R	0x0	eFuse power disable done. Clear on read.
3:1	RSVD	R	0x0	Reserved bits
0	EN_DONE	R	0x0	eFuse power enable done. Clear on read.

3.7.6.1.20 I2S_CLK_CFG

- **Name:** I2S_CLK_CFG Register
- **Description:** I2S_CLK_CFG Register
- **Base Address:** 0x4000E000
- **Offset:** 0x268
- **Reset Value:** 0x00040008

Table 3-198 I2S_CLK_CFG Register

Bits	Field Name	RW	Reset	Description
31:21	RSVD	R	0x0	Reserved bits
20	SRC_CLK_EN	RW	0x0	1: Enable I2S source clock

Bits	Field Name	RW	Reset	Description
19:18	SRC_CLK_SEL	RW	0x1	clock divider source select 0: 96M, 1: 64M, 2: 32M
17	RSVD	R	0x0	Reserved bits
16	CLK_DIV_EN	RW	0x0	1: enable i2s clock divider, 0: No clock output
15:12	RSVD	R	0x0	Reserved bits
11:0	DIV_CNT	RW	0x8	divide_ratio = 1/(target + 2). Duty cycle is not 50 if the number is odd. For example: target = 0, ratio = 1/2

3.7.6.1.21 I2S_DMA_MODE

- **Name:** I2S_DMA_MODE Register
- **Description:** I2S_DMA_MODE Register
- **Base Address:** 0x4000E000
- **Offset:** 0x26C
- **Reset Value:** 0x00000011

Table 3-199 I2S_DMA_MODE Register

Bits	Field Name	RW	Reset	Description
31:5	RSVD	R	0x0	Reserved bits
4	S_SET	RW	0x1	I2S_Slave DMA mode enable 0: disable 1: enable
3:1	RSVD	R	0x0	Reserved bits
0	SET	RW	0x1	I2S_Master DMA mode enable 0: disable 1: enable

3.7.6.1.22 MCU_SUB_REG

- **Name:** MCU_SUB_REG Register
- **Description:** MCU_SUB_REG Register
- **Base Address:** 0x4000E000
- **Offset:** 0x280
- **Reset Value:** 0x00000000

Table 3-200 MCU_SUB_REG Register

Bits	Field Name	RW	Reset	Description
31:14	RSVD	R	0x0	Reserved bits
13:12	GDX_REG_WAIT_STATE	RW	0x0	ROM AHB wait state number
11:6	RSVD	R	0x0	Reserved bits
5:4	MEM_BOND_OPT	R	0x0	Memory bond option: Available SRAM Size 0x0 : 256KB 0x1 : 192KB 0x2 : 128KB 0x3 : 256KB
3:0	RSVD	R	0x0	Reserved bits

3.7.6.1.23 MCU_NMI_CFG

- **Name:** MCU_NMI_CFG Register
- **Description:** MCU_NMI_CFG Register
- **Base Address:** 0x4000E000
- **Offset:** 0x284
- **Reset Value:** 0x00000000

Table 3-201 MCU_NMI_CFG Register

Bits	Field Name	RW	Reset	Description
31:10	RSVD	R	0x0	Reserved bits
9:0	MCU_NMI_SEL	RW	0x0	Select MCU NMI. bit9: IRQ#57 bit8: IRQ#56 bit7: IRQ#8 bit6: IRQ#48 bit5: IRQ#28 bit4: IRQ#25 bit3: IRQ#20 bit2: IRQ#2 bit1: IRQ#7 bit0: IRQ#6

3.7.6.1.24 CPLL_IRQ_CFG

- **Name:** CPLL_IRQ_CFG Register
- **Description:** CPLL_IRQ_CFG Register
- **Base Address:** 0x4000E000
- **Offset:** 0x288
- **Reset Value:** 0x00000000

Table 3-202 CPLL_IRQ_CFG Register

Bits	Field Name	RW	Reset	Description
31:3	RSVD	R	0x0	Reserved bits
2	DRIFT_IRQ_CLR	W	0x0	Write 1 to clear DRIFT_IRQ_STAT
1	DRIFT_IRQ_STAT	R	0x0	CPLL drift IRQ status
0	DRIFT_IRQ_EN	RW	0x0	Enable CPLL drift IRQ.

3.7.6.1.25 AON_SW_RST

- **Name:** AON_SW_RST Register
- **Description:** AON_SW_RST Register
- **Base Address:** 0x4000E000
- **Offset:** 0x28C
- **Reset Value:** 0x00000000

Table 3-203 AON_SW_RST Register

Bits	Field Name	RW	Reset	Description
31:16	SET	W	0x0	Write 0xC5A1 along with bit 0/1 to perform aon reset Write any other value will not apply the aon reset.
15:9	RSVD	R	0x0	Reserved bits
8	FULL	W	0x0	Reset all AON modules. This operation will cause the power-cycle.
7:1	RSVD	R	0x0	Reserved bits
0	PARTIAL	W	0x0	Reset AON modules except the power controller.

3.7.6.1.26 APB_TIMER_DBG

- **Name:** APB_TIMER_DBG Register
- **Description:** APB_TIMER_DBG Register

- **Base Address:** 0x4000E000
- **Offset:** 0x340
- **Reset Value:** 0x0000000F

Table 3-204 APB_TIMER_DBG Register

Bits	Field Name	RW	Reset	Description
31:4	RSVD	R	0x0	Reserved bits
3	WDT	RW	0x0	Write 1 to stop the watchdog timer during CPU HALT.
2	DUAL_TIMER	RW	0x0	Write 1 to stop the dual timer during CPU HALT.
1	TIMER1	RW	0x0	Write 1 to stop the timer1 during CPU HALT.
0	TIMER0	RW	0x0	Write 1 to stop the timer0 during CPU HALT.

3.7.6.1.27 APB_MON_DBG

- **Name:** APB_MON_DBG Register
- **Description:** APB_MON_DBG Register
- **Base Address:** 0x4000E000
- **Offset:** 0x344
- **Reset Value:** 0x00000000

Table 3-205 APB_MON_DBG Register

Bits	Field Name	RW	Reset	Description
31:1	RSVD	R	0x0	Reserved bits
0	BYPASS	RW	0x0	Write 1 to bypass AHB2APB bridge monitor.

3.7.6.1.28 MCU_RELEASE

- **Name:** MCU_RELEASE Register
- **Description:** MCU_RELEASE Register
- **Base Address:** 0x4000E000
- **Offset:** 0x380
- **Reset Value:** 0x00000000

Table 3-206 MCU_RELEASE Register

Bits	Field Name	RW	Reset	Description
31:0	MCU_RELEASE	R	0x0	Contain the RTL release date.

3.7.6.1.29 FPGA_CTRL

- **Name:** FPGA_CTRL Register
- **Description:** FPGA_CTRL Register
- **Base Address:** 0x4000E000
- **Offset:** 0x384
- **Reset Value:** 0x00000000

Table 3-207 FPGA_CTRL Register

Bits	Field Name	RW	Reset	Description
31:5	RSVD	R	0x0	Reserved bits
4	EXIST	R	0x0	This HW is 0: ASIC, 1: FPGA.
3:0	RSVD	R	0x0	Reserved bits

3.7.6.1.30 ST_CALIB

- **Name:** ST_CALIB Register
- **Description:** ST_CALIB Register
- **Base Address:** 0x4000E000
- **Offset:** 0x388
- **Reset Value:** 0x00000190

Table 3-208 ST_CALIB Register

Bits	Field Name	RW	Reset	Description
31:29	RSVD	R	0x0	Reserved bits
28	STCALIB_CLK	RW	0x0	mcu_stclk_sel Value: 0x0: rng_32khz_clk 0x1: xo_clk
27:26	RSVD	R	0x0	Reserved bits
25:0	STCALIB	RW	0x190	STCAILIB to ARM Core

3.7.6.2 MCU_RET

3.7.6.2.1 MCU_SUBSYS_CG_CTRL[0]

- **Name:** MCU_SUBSYS_CG_CTRL[0] Register
- **Description:** MCU_SUBSYS_CG_CTRL[0] Register

- **Base Address:** 0x4000EE00
- **Offset:** 0xA0
- **Reset Value:** 0x0000FFFF

Table 3-209 MCU_SUBSYS_CG_CTRL[0] Register

Bits	Field Name	RW	Reset	Description
31:16	RSVD	R	0x0	
15:11	RSVD	R	0x1F	
10	WFI_SERIAL_HCLK	RW	0x1	HCLK for serial blocks including I2C, UART, QSPI, and SPI(M & S) **Value:** • 0x0: HCLK is always on. • 0x1: HCLK is gated during WFI/WFE.
9	WFI_APB_SUB_HCLK	RW	0x1	HCLK for APB subsystem including watchdog and timer **Value:** • 0x0: HCLK is always on. • 0x1: HCLK is gated during WFI/WFE.
8	WFI_BLE_BRG_HCLK	RW	0x1	HCLK for Bluetooth LE MCU bridge **Value:** • 0x0: HCLK is always on. • 0x1: HCLK is gated during WFI/WFE.
7	RSVD	R	0x1	Reserved bits
6	WFI_GPIO_HCLK	RW	0x1	HCLK for GPIOs **Value:** • 0x0: HCLK is always on. • 0x1: HCLK is gated during WFI/WFE.
5	WFI_SNSADC_HCLK	RW	0x1	HCLK for sense ADC **Value:** • 0x0: HCLK is always on. • 0x1: HCLK is gated during WFI/WFE.
4	WFI_ROM_HCLK	RW	0x1	HCLK for ROM **Value:** • 0x0: HCLK is always on. • 0x1: HCLK is gated during WFI/WFE.
3	RSVD	R	0x1	Reserved bits

Bits	Field Name	RW	Reset	Description
2	WFI_HTB_HCLK	RW	0x1	HCLK for hopping table **Value:** • 0x0: HCLK is always on. • 0x1: HCLK is gated during WFI/WFE.
1	RSVD	R	0x1	Reserved bits
0	WFI_SECU_HCLK	RW	0x1	HCLK for all security blocks including eFuse and TRNG **Value:** • 0x0: HCLK is always on. • 0x1: HCLK is gated during WFI/WFE.

3.7.6.2.2 MCU_SUBSYS_CG_CTRL[1]

- **Name:** MCU_SUBSYS_CG_CTRL[1] Register
- **Description:** MCU_SUBSYS_CG_CTRL[1] Register
- **Base Address:** 0x4000EE00
- **Offset:** 0xA4
- **Reset Value:** 0x0000FFEF

Table 3-210 MCU_SUBSYS_CG_CTRL[1] Register

Bits	Field Name	RW	Reset	Description
31:16	RSVD	R	0x0	Reserved bits
15:11	RSVD	R	0x1F	Reserved bits
10	FORCE_SERIAL_HCLK	RW	0x1	HCLK for serial blocks including I2C, UART, QSPI, and SPI(M & S) **Value:** • 0x0: Force the clock to be ON. • 0x1: Force the clock to be OFF.
9	FORCE_APB_SUB_HCLK	RW	0x1	HCLK for APB subsystem including watchdog and timer **Value:** • 0x0: Force the clock to be ON. • 0x1: Force the clock to be OFF.
8	FORCE_BLE_BRG_HCLK	RW	0x1	HCLK for Bluetooth LE MCU bridge **Value:** • 0x0: Force the clock to be ON. • 0x1: Force the clock to be OFF.

Bits	Field Name	RW	Reset	Description
7	RSVD	R	0x1	Reserved bits
6	FORCE_GPIO_HCLK	RW	0x1	HCLK for GPIOs **Value:** • 0x0: Force the clock to be ON. • 0x1: Force the clock to be OFF.
5	FORCE_SNSADC_HCLK	RW	0x1	HCLK for sense ADC **Value:** • 0x0: Force the clock to be ON. • 0x1: Force the clock to be OFF.
4	FORCE_ROM_HCLK	RW	0x0	HCLK for ROM **Value:** • 0x0: Force the clock to be ON. • 0x1: Force the clock to be OFF.
3	RSVD	R	0x1	Reserved bits
2	FORCE_HTB_HCLK	RW	0x1	HCLK for hopping table **Value:** • 0x0: Force the clock to be ON. • 0x1: Force the clock to be OFF.
1	RSVD	R	0x1	Reserved bits
0	FORCE_SECU_HCLK	RW	0x1	HCLK for all security blocks including eFuse and TRNG **Value:** • 0x0: Force the clock to be ON. • 0x1: Force the clock to be OFF.

3.7.6.2.3 MCU_SUBSYS_CG_CTRL[2]

- **Name:** MCU_SUBSYS_CG_CTRL[2] Register
- **Description:** MCU_SUBSYS_CG_CTRL[2] Register
- **Base Address:** 0x4000EE00
- **Offset:** 0xA8
- **Reset Value:** 0x00020007

Table 3-211 MCU_SUBSYS_CG_CTRL[2] Register

Bits	Field Name	RW	Reset	Description
31:19	RSVD	R	0x0	Reserved bits

Bits	Field Name	RW	Reset	Description
18	FORCE_SRAM_HCLK	RW	0x0	SRAM HCLK clock **Value:** • 0x0: Force the clock to be ON. • 0x1: Force the clock to be OFF.
17	FORCE_XF_XQSPI_HCLK	RW	0x1	Cache HCLK clock **Value:** • 0x0: Force the clock to be ON. • 0x1: Force the clock to be OFF.
16	FORCE_AON_MCUSUB_HCLK	RW	0x0	hclk for mcu_aux_reg and always_on_reg **Value:** • 0x0: Force the clock to be ON. • 0x1: Force the clock to be OFF.
15:3	RSVD	R	0x0	Reserved bits
2	WFI_SRAM_HCLK	RW	0x1	SRAM HCLK clock during WFI **Value:** • 0x0: HCLK is always on. • 0x1: HCLK is gated during WFI/WFE.
1	WFI_XF_XQSPI_HCLK	RW	0x1	Cache HCLK clock during WFI **Value:** • 0x0: HCLK is always on. • 0x1: HCLK is gated during WFI/WFE.
0	WFI_AON_MCUSUB_HCLK	RW	0x1	hclk for mcu_aux_reg and always_on_reg during WFI **Value:** • 0x0: HCLK is always on. • 0x1: HCLK is gated during WFI/WFE.

3.7.6.2.4 MCU_PERIPH_PCLK_OFF

- **Name:** MCU_PERIPH_PCLK_OFF Register
- **Description:** MCU_PERIPH_PCLK_OFF Register
- **Base Address:** 0x4000EE00
- **Offset:** 0xAC
- **Reset Value:** 0x7F7FC3CF

Table 3-212 MCU_PERIPH_PCLK_OFF Register

Bits	Field Name	RW	Reset	Description
31	RSVD	R	0x0	Reserved bits
30	FORCE_VTTBL_PCLK	RW	0x1	VTTBL PCLK **Value:** • 0x0: Force the clock to be ON. • 0x1: Force the clock to be OFF.
29	FORCE_PWM_1_PCLK	RW	0x1	PWM_1 PCLK **Value:** • 0x0: Force the clock to be ON. • 0x1: Force the clock to be OFF.
28	FORCE_PWM_0_PCLK	RW	0x1	PWM_0 PCLK **Value:** • 0x0: Force the clock to be ON. • 0x1: Force the clock to be OFF.
27	FORCE_XQSPI_DIV4_PCLK	RW	0x1	XQSPI_DIV4 PCLK **Value:** • 0x0: Force the clock to be ON. • 0x1: Force the clock to be OFF.
26	WFI_XQSPI_DIV4_PCLK	RW	0x1	XQSPI_DIV4_PCLK during WFI **Value:** • 0x0: PCLK is always on. • 0x1: PCLK is gated during WFI/WFE.
25	FORCE_SECU_DIV4_PCLK	RW	0x1	SECU_DIV4 PCLK **Value:** • 0x0: Force the clock to be ON. • 0x1: Force the clock to be OFF.
24	WFI_SECU_DIV4_PCLK	RW	0x1	SECU_DIV4_PCLK during WFI **Value:** • 0x0: PCLK is always on. • 0x1: PCLK is gated during WFI/WFE.
23	RSVD	R	0x0	Reserved bits
22	FORCE_PDM_PCLK	RW	0x1	PDM PCLK **Value:** • 0x0: Force the clock to be ON.

Bits	Field Name	RW	Reset	Description
				• 0x1: Force the clock to be OFF.
21	FORCE_DSPI_PCLK	RW	0x1	DSPI PCLK **Value:** • 0x0: Force the clock to be ON. • 0x1: Force the clock to be OFF.
20	FORCE_I2S_S_PCLK	RW	0x1	I2S_S PCLK **Value:** • 0x0: Force the clock to be ON. • 0x1: Force the clock to be OFF.
19	FORCE_I2S_PCLK	RW	0x1	I2S PCLK **Value:** • 0x0: Force the clock to be ON. • 0x1: Force the clock to be OFF.
18	FORCE_SPI_S_PCLK	RW	0x1	SPI_S PCLK **Value:** • 0x0: Force the clock to be ON. • 0x1: Force the clock to be OFF.
17	FORCE_SPI_M_PCLK	RW	0x1	SPI_M PCLK **Value:** • 0x0: Force the clock to be ON. • 0x1: Force the clock to be OFF.
16	FORCE_QSPI2_PCLK	RW	0x1	QSPI_2 PCLK **Value:** • 0x0: Force the clock to be ON. • 0x1: Force the clock to be OFF.
15	FORCE_QSPI1_PCLK	RW	0x1	QSPI_1 PCLK **Value:** • 0x0: Force the clock to be ON. • 0x1: Force the clock to be OFF.
14	FORCE_QSPI0_PCLK	RW	0x1	QSPI_0 PCLK **Value:** • 0x0: Force the clock to be ON. • 0x1: Force the clock to be OFF.
13:11	RSVD	R	0x0	Reserved bits

Bits	Field Name	RW	Reset	Description
9	FORCE_I2C3_PCLK	RW	0x1	I2C_3 PCLK **Value:** • 0x0: Force the clock to be ON. • 0x1: Force the clock to be OFF.
8	FORCE_I2C2_PCLK	RW	0x1	I2C_2 PCLK **Value:** • 0x0: Force the clock to be ON. • 0x1: Force the clock to be OFF.
7	FORCE_I2C1_PCLK	RW	0x1	I2C_1 PCLK **Value:** • 0x0: Force the clock to be ON. • 0x1: Force the clock to be OFF.
6	FORCE_I2C0_PCLK	RW	0x1	I2C_0 PCLK **Value:** • 0x0: Force the clock to be ON. • 0x1: Force the clock to be OFF.
5:4	RSVD	R	0x0	Reserved bits
3	FORCE_UART3_PCLK	RW	0x1	uart_3 PCLK **Value:** • 0x0: Force the clock to be ON. • 0x1: Force the clock to be OFF.
2	FORCE_UART2_PCLK	RW	0x1	uart_2 PCLK **Value:** • 0x0: Force the clock to be ON. • 0x1: Force the clock to be OFF.
1	FORCE_UART1_PCLK	RW	0x1	uart_1 PCLK **Value:** • 0x0: Force the clock to be ON. • 0x1: Force the clock to be OFF.
0	FORCE_UART0_PCLK	RW	0x1	uart_0 PCLK **Value:** • 0x0: Force the clock to be ON. • 0x1: Force the clock to be OFF.

3.7.6.2.5 MCU_PERIPH_CG_LP_EN

- **Name:** MCU_PERIPH_CG_LP_EN Register
- **Description:** MCU_PERIPH_CG_LP_EN Register
- **Base Address:** 0x4000EE00
- **Offset:** 0xB0
- **Reset Value:** 0x00000F3D

Table 3-213 MCU_PERIPH_CG_LP_EN Register

Bits	Field Name	RW	Reset	Description
31:12	RSVD	R	0x0	Reserved bits
11	AHB2APB_ASYNC_EN	RW	0x1	Enable apb2apb async bridge low-power feature.
10	AHB2APB_SYNC_EN	RW	0x1	Enable ahb2apb bridge low-power feature.
9	QSPIM_EN	RW	0x1	Enable QSPIM low-power feature.
8	AHB_BUS_LP_EN	RW	0x1	Enable AHB bus low-power feature.
7:6	RSVD	R	0x0	Reserved bits
5	I2C_LP_SCLK_EN	RW	0x1	Enable I2C SCLK low-power feature.
4	SPIS_LP_SCLK_EN	RW	0x1	Enable SPIS SCLK low-power feature.
3	SPIM_LP_SCLK_EN	RW	0x1	Enable SPIM SCLK low-power feature.
2	I2S_LP_EN	RW	0x1	Enable I2S low-power feature.
1	UART_LP_PCLK_EN	RW	0x0	Enable uart PCLK low-power feature.
0	UART_LP_SCLK_EN	RW	0x1	Enable uart SCLK low-power feature.

3.7.6.2.6 MCU_PERIPH_CLK_SLP_OFF

- **Name:** MCU_PERIPH_CLK_SLP_OFF Register
- **Description:** MCU_PERIPH_CLK_SLP_OFF Register
- **Base Address:** 0x4000EE00
- **Offset:** 0xB4
- **Reset Value:** 0x007FFF0F

Table 3-214 MCU_PERIPH_CLK_SLP_OFF Register

Bits	Field Name	RW	Reset	Description
31:23	RSVD	R	0x0	Reserved bits
22	I2C3	RW	0x1	Enable I2C3 auto off when CPU is in sleep state.
21	I2C2	RW	0x1	Enable I2C2 auto off when CPU is in sleep state.

Bits	Field Name	RW	Reset	Description
20	I2C1	RW	0x1	Enable I2C1 auto off when CPU is in sleep state.
19	I2C0	RW	0x1	Enable I2C0 auto off when CPU is in sleep state.
18	PDM	RW	0x1	Enable PDM auto off when CPU is in sleep state.
17	DSPI	RW	0x1	Enable DSPI auto off when CPU is in sleep state.
16	QSPIM2	RW	0x1	Enable QSPIM2 auto off when CPU is in sleep state.
15	QSPIM1	RW	0x1	Enable QSPIM1 auto off when CPU is in sleep state.
14	QSPIM0	RW	0x1	Enable QSPIM0 auto off when CPU is in sleep state.
13	PWM1	RW	0x1	Enable PWM1 auto off when CPU is in sleep state.
12	PWM0	RW	0x1	Enable PWM0 auto off when CPU is in sleep state.
11	SPIS	RW	0x1	Enable SPIS auto off when CPU is in sleep state.
10	SPIM	RW	0x1	Enable SPIM auto off when CPU is in sleep state.
9	I2SS	RW	0x1	Enable I2SS auto off when CPU is in sleep state.
8	I2SM	RW	0x1	Enable I2SM auto off when CPU is in sleep state.
7:4	RSVD	R	0x0	Reserved bits
3	UART3	RW	0x1	Enable uart_3 auto off when CPU is in sleep state.
2	UART2	RW	0x1	Enable uart_2 auto off when CPU is in sleep state.
1	UART1	RW	0x1	Enable uart_1 auto off when CPU is in sleep state.
0	UART0	RW	0x1	Enable uart_0 auto off when CPU is in sleep state.

3.7.6.2.7 SECU_CLK_CTRL

- **Name:** SECU_CLK_CTRL Register
- **Description:** SECU_CLK_CTRL Register
- **Base Address:** 0x4000EE00
- **Offset:** 0xB8
- **Reset Value:** 0x00003FFF

Table 3-215 SECU_CLK_CTRL Register

Bits	Field Name	RW	Reset	Description
31:14	RSVD	R	0x0	Reserved bits
13	EFUSE_HCLK_SLP_OFF	RW	0x1	Enable eFuse HCLK auto off when CPU is in sleep state.
12	EFUSE_HCLK_FORCE_OFF	RW	0x1	eFuse HCLK **Value:** • 0x0: Force the clock to be ON. • 0x1: Force the clock to be OFF.

Bits	Field Name	RW	Reset	Description
11	RNG_HCLK_SLP_OFF	RW	0x1	Enable RNGHCLK auto off when CPU is in sleep state.
10	RNG_HCLK_FORCE_OFF	RW	0x1	RNG HCLK **Value:** • 0x0: Force the clock to be ON. • 0x1: Force the clock to be OFF.
9	RAMKEY_HCLK_SLP_OFF	RW	0x1	Enable KEYRAM HCLK auto off when CPU is in sleep state.
8	RAMKEY_HCLK_FORCE_OFF	RW	0x1	KEYRAM HCLK **Value:** • 0x0: Force the clock to be ON. • 0x1: Force the clock to be OFF.
7	PRESENT_HCLK_SLP_OFF	RW	0x1	Enable PRESENT HCLK auto off when CPU is in sleep state.
6	PRESENT_HCLK_FORCE_OFF	RW	0x1	PRESENT HCLK **Value:** • 0x0: Force the clock to be ON. • 0x1: Force the clock to be OFF.
5	PKC_HCLK_SLP_OFF	RW	0x1	Enable PKC HCLK auto off when CPU is in sleep state.
4	PKC_HCLK_FORCE_OFF	RW	0x1	PKC HCLK **Value:** • 0x0: Force the clock to be ON. • 0x1: Force the clock to be OFF.
3	HMAC_HCLK_SLP_OFF	RW	0x1	Enable HMAC HCLK auto off when CPU is in sleep state.
2	HMAC_HCLK_FORCE_OFF	RW	0x1	HMAC HCLK **Value:** • 0x0: Force the clock to be ON. • 0x1: Force the clock to be OFF.
1	AES_HCLK_SLP_OFF	RW	0x1	Enable AES HCLK auto off when CPU is in sleep state.
0	AES_HCLK_FORCE_OFF	RW	0x1	AES HCLK **Value:** • 0x0: Force the clock to be ON. • 0x1: Force the clock to be OFF.

3.7.6.2.8 MCU_MISC_CLK

- **Name:** MCU_MISC_CLK Register
- **Description:** MCU_MISC_CLK Register

- **Base Address:** 0x4000EE00
- **Offset:** 0xBC
- **Reset Value:** 0x0000003C

Table 3-216 MCU_MISC_CLK Register

Bits	Field Name	RW	Reset	Description
31:6	RSVD	R	0x0	Reserved bits
5	DMA2_HCLK_OFF	RW	0x1	**Value:** • 0x0: Turn on DMA2 CLK. • 0x1: Turn off DMA2 CLK.
4	DMA1_HCLK_OFF	RW	0x1	**Value:** • 0x0: Turn on DMA1 CLK. • 0x1: Turn off DMA1 CLK.
3	DMA0_HCLK_OFF	RW	0x1	**Value:** • 0x0: Turn on DMA0 CLK. • 0x1: Turn off DMA0 CLK.
2	XQSPI_SCK_OFF	RW	0x1	**Value:** • 0x0: Turn on XQSPI SPI CLK. • 0x1: Turn off XQSPI SPI CLK.
1:0	RSVD	R	0x0	Reserved bits

3.7.6.2.9 HFOSC_CLK_EN

- **Name:** HFOSC_CLK_EN Register
- **Description:** HFOSC_CLK_EN Register
- **Base Address:** 0x4000EE00
- **Offset:** 0xC0
- **Reset Value:** 0x00000001

Table 3-217 HFOSC_CLK_EN Register

Bits	Field Name	RW	Reset	Description
31:1	RSVD	R	0x0	Reserved bits
0	EN	RW	0x1	**Value:** • 0x0: Turn off HFOSC CLK. • 0x1: Turn on HFOSC CLK.

3.7.6.2.10 CLK_FORCE_ON

- **Name:** CLK_FORCE_ON Register
- **Description:** CLK_FORCE_ON Register
- **Base Address:** 0x4000EE00
- **Offset:** 0xC4
- **Reset Value:** 0x00000000

Table 3-218 CLK_FORCE_ON Register

Bits	Field Name	RW	Reset	Description
31:8	RSVD	R	0x0	Reserved bits
7	XO_16_FORCE_ON	RW	0x0	1: Force XO 16MHz clk on; 0: Automatially controlled.
6	XO_32_FORCE_ON	RW	0x0	1: Force XO 32MHz clk on; 0: Automatially controlled.
5	PLL_16_FORCE_ON	RW	0x0	1: Force cpll_hf_osc 16MHz clk on; 0: Automatially controlled.
4	PLL_24_FORCE_ON	RW	0x0	1: Force cpll_hf_osc 24MHz clk on; 0: Automatially controlled.
3	PLL_32_FORCE_ON	RW	0x0	1: Force cpll_hf_osc 32MHz clk on; 0: Automatially controlled.
2	PLL_48_FORCE_ON	RW	0x0	1: Force cpll_hf_osc 48MHz clk on; 0: Automatially controlled.
1	PLL_64_FORCE_ON	RW	0x0	1: Force cpll_hf_osc 64MHz clk on; 0: Automatially controlled.
0	PLL_96_FORCE_ON	RW	0x0	1: Force cpll_hf_osc 96MHz clk on; 0: Automatially controlled.

3.7.7 Clock Calibration

3.7.7.1 SL_CLK_CTRL

- **Name:** SL_CLK_CTRL Register
- **Description:** This register enables SL clock calibration.
- **Base Address:** 0x4000E400
- **Offset:** 0x0
- **Reset Value:** 0x00000000

Table 3-219 SL_CLK_CTRL Register

Bits	Field Name	RW	Reset	Description
31:1	RSVD	R	0x0	Reserved bits
0	EN	RW	0x0	Enable clock calibration block. **Value:** • 0x1: Enable the block. • 0x0: Disable the block.

3.7.7.2 SL_CLK_CNT

- **Name:** SL_CLK_CNT Register
- **Description:** This register sets the number of cycles of the slow clock.
- **Base Address:** 0x4000E400
- **Offset:** 0x4
- **Reset Value:** 0x00000000

Table 3-220 SL_CLK_CNT Register

Bits	Field Name	RW	Reset	Description
31:12	RSVD	R	0x0	Reserved bits
11:0	COUNT	RW	0x0	Number of cycles of the slow clock

3.7.7.3 SL_CLK_STAT

- **Name:** SL_CLK_STAT Register
- **Description:** This register indicates the calibration status.
- **Base Address:** 0x4000E400
- **Offset:** 0x8
- **Reset Value:** 0x00000000

Table 3-221 SL_CLK_STAT Register

Bits	Field Name	RW	Reset	Description
31:2	RSVD	R	0x0	Reserved bits
1	OVER	R	0x0	Fast clock counter overflow. Need to reduce slow clock cnt_val.
0	DONE	R	0x0	Calibration done status

3.7.7.4 SL_CLK_CNT0

- **Name:** SL_CLK_CNT0 Register
- **Description:** This register sets the fast clock counter value.
- **Base Address:** 0x4000E400
- **Offset:** 0xC
- **Reset Value:** 0x00000000

Table 3-222 SL_CLK_CNT0 Register

Bits	Field Name	RW	Reset	Description
31:24	RSVD	R	0x0	Reserved bits
23:0	VAL	R	0x0	Fast clock counter value

3.7.7.5 SL_CLK_CNT1

- **Name:** SL_CLK_CNT1 Register
- **Description:** This register reads the current counter value of the slow clock.
- **Base Address:** 0x4000E400
- **Offset:** 0x10
- **Reset Value:** 0x00000000

Table 3-223 SL_CLK_CNT1 Register

Bits	Field Name	RW	Reset	Description
31:12	RSVD	R	0x0	Reserved bits
11:0	VAL	R	0x0	Read the current counter value of the slow clock.

3.7.7.6 SL_CLK_INT_EN

- **Name:** SL_CLK_INT_EN Register
- **Description:** This register enables the clock interrupt.
- **Base Address:** 0x4000E400
- **Offset:** 0x14
- **Reset Value:** 0x00000003

Table 3-224 SL_CLK_INT_EN Register

Bits	Field Name	RW	Reset	Description
31:2	RSVD	R	0x0	Reserved bits

Bits	Field Name	RW	Reset	Description
1	OVER	RW	0x1	Interrupt Enable ONLY. The status in CLK_STAT can be read no matter whether the interrupt is enabled.
0	DONE	RW	0x1	Interrupt Enable ONLY. The status in CLK_STAT can be read no matter whether the interrupt is enabled.

3.7.7.7 SL_CLK_INT_CLR

- **Name:** SL_CLK_INT_CLR Register
- **Description:** This register clears the clock interrupt.
- **Base Address:** 0x4000E400
- **Offset:** 0x18
- **Reset Value:** 0x00000000

Table 3-225 SL_CLK_INT_CLR Register

Bits	Field Name	RW	Reset	Description
31:2	RSVD	R	0x0	Reserved bits
1	OVER	W	0x0	Write the corresponding bit to clear the interrupt.
0	DONE	W	0x0	Write the corresponding bit to clear the interrupt.

3.7.7.8 SL_CLK_SEL

- **Name:** SL_CLK_SEL Register
- **Description:** This register selects the slow clock source.
- **Base Address:** 0x4000E400
- **Offset:** 0x20
- **Reset Value:** 0x00000000

Table 3-226 SL_CLK_SEL Register

Bits	Field Name	RW	Reset	Description
31:2	RSVD	R	0x0	Reserved bits
1:0	SLOW_CLK_SEL	RW	0x0	Select slow clock source. Only valid for the defined source value **Value:** • 0x0: LFXO_32K • 0x1: LFRC_32K

Bits	Field Name	RW	Reset	Description
				• 0x2: RNG_32K from RNG_OSC

3.7.7.9 HS_CLK_CTRL

- **Name:** HS_CLK_CTRL Register
- **Description:** This register enables HS clock calibration.
- **Base Address:** 0x4000E400
- **Offset:** 0x100
- **Reset Value:** 0x00000000

Table 3-227 HS_CLK_CTRL Register

Bits	Field Name	RW	Reset	Description
31:1	RSVD	R	0x0	Reserved bits
0	EN	RW	0x0	Enable clock calibration block. **Value:** • 0x1: Enable the block. • 0x0: Disable the block.

3.7.7.10 HS_CLK_CNT

- **Name:** HS_CLK_CNT Register
- **Description:** This register sets the number of cycles of the slow clock.
- **Base Address:** 0x4000E400
- **Offset:** 0x104
- **Reset Value:** 0x00000000

Table 3-228 HS_CLK_CNT Register

Bits	Field Name	RW	Reset	Description
31:12	RSVD	R	0x0	Reserved bits
11:0	COUNT	RW	0x0	Number of cycles of the slow clock

3.7.7.11 HS_CLK_STAT

- **Name:** HS_CLK_STAT Register
- **Description:** This register indicates the calibration status.
- **Base Address:** 0x4000E400

- **Offset:** 0x108
- **Reset Value:** 0x00000000

Table 3-229 HS_CLK_STAT Register

Bits	Field Name	RW	Reset	Description
31:2	RSVD	R	0x0	Reserved bits
1	OVER	R	0x0	Fast clock counter overflow. Need to reduce slow clock cnt_val.
0	DONE	R	0x0	Calibration done status

3.7.7.12 HS_CLK_CNT0

- **Name:** HS_CLK_CNT0 Register
- **Description:** This register sets the fast clock counter value.
- **Base Address:** 0x4000E400
- **Offset:** 0x10C
- **Reset Value:** 0x00000000

Table 3-230 HS_CLK_CNT0 Register

Bits	Field Name	RW	Reset	Description
31:24	RSVD	R	0x0	Reserved bits
23:0	VAL	R	0x0	Fast clock counter value

3.7.7.13 HS_CLK_CNT1

- **Name:** HS_CLK_CNT1 Register
- **Description:** This register reads the current counter value of the slow clock.
- **Base Address:** 0x4000E400
- **Offset:** 0x110
- **Reset Value:** 0x00000000

Table 3-231 HS_CLK_CNT1 Register

Bits	Field Name	RW	Reset	Description
31:12	RSVD	R	0x0	Reserved bits
11:0	VAL	R	0x0	Read the current counter value of the slow clock.

3.7.7.14 HS_CLK_INT_EN

- **Name:** HS_CLK_INT_EN Register
- **Description:** This register enables the clock interrupt.
- **Base Address:** 0x4000E400
- **Offset:** 0x114
- **Reset Value:** 0x00000003

Table 3-232 HS_CLK_INT_EN Register

Bits	Field Name	RW	Reset	Description
31:2	RSVD	R	0x0	Reserved bits
1	OVER	RW	0x1	Interrupt Enable ONLY. The status in CLK_STAT can be read no matter whether the interrupt is enabled.
0	DONE	RW	0x1	Interrupt Enable ONLY. The status in CLK_STAT can be read no matter whether the interrupt is enabled.

3.7.7.15 HS_CLK_INT_CLR

- **Name:** HS_CLK_INT_CLR Register
- **Description:** This register clears the clock interrupt.
- **Base Address:** 0x4000E400
- **Offset:** 0x118
- **Reset Value:** 0x00000000

Table 3-233 HS_CLK_INT_CLR Register

Bits	Field Name	RW	Reset	Description
31:2	RSVD	R	0x0	Reserved bits
1	OVER	W	0x0	Write the corresponding bit to clear the interrupt.
0	DONE	W	0x0	Write the corresponding bit to clear the interrupt.

3.7.7.16 HS_CLK_SEL

- **Name:** HS_CLK_SEL Register
- **Description:** This register selects the slow clock source.
- **Base Address:** 0x4000E400
- **Offset:** 0x120
- **Reset Value:** 0x00000000

Table 3-234 HS_CLK_SEL Register

Bits	Field Name	RW	Reset	Description
31:2	RSVD	R	0x0	Reserved bits
1:0	SLOW_CLK_SEL	RW	0x0	Select slow clock source. Only valid for the defined source value **Value:** • 0x0: LFXO_32K • 0x1: LFRC_32K • 0x2: XO_32K from XO_32M

3.7.8 DDVS

3.7.8.1 DDVS_EN

- **Name:** DDVS_EN Register
- **Description:** DDVS_EN Register
- **Base Address:** 0x4000E800
- **Offset:** 0x0
- **Reset Value:** 0x00000000

Table 3-235 DDVS_EN Register

Bits	Field Name	RW	Reset	Description
31:1	RSVD	R	0x0	Reserved bits
0	DDVS_EN	RW	0x0	Enable DDVS. **Value:** • 0x0: Disable DDVS (default). • 0x1: Enable DDVS.

3.7.8.2 DDVS_CFG_1

- **Name:** DDVS_CFG_1 Register
- **Description:** DDVS_CFG_1 Register
- **Base Address:** 0x4000E800
- **Offset:** 0x4
- **Reset Value:** 0x00000000

Table 3-236 DDVS_CFG_1 Register

Bits	Field Name	RW	Reset	Description
31	CONF_DDVS_MODE	RW	0x0	DDVS mode **Value:** • 0x0: DDVS automatic mode (default) • 0x1: DDVS manual mode
30:29	CONF_DIV_FACTOR	RW	0x0	Ringo frequency division factor **Value:** • 0x0: 8K (default) • 0x1: 4K • 0x2: 16K • 0x3: reserved
28:25	CONF_RINGO_EN	RW	0x0	Enable each ringo. **Value:** • 0x0: Disable all 4 ringos (default). • Others: Enable corresponding ringos.
24	CONF_INT_EN	RW	0x0	Enable error interrupt. **Value:** • 0x0: Disable error interrupt. • 0x1: Enable error interrupt.
23	ERR_INT	WC	0x0	Error interrupt **Value:** • 0x0: no error • 0x1: error interrupt **Note:** CPU writes 1 to clear, and writing 0 is invalid.
22:17	CONF_VREF_MANUAL	RW	0x0	Manual vref In manual mode, MCU writes this register to set vref.
16	RSVD	R	0x0	Reserved bit
15:4	CONF_THRESHOLD_SLOW	RW	0x0	Drift slow threshold that changes lock state to re-adjust vref
3:0	RSVD	R	0x0	Reserved bits

3.7.8.3 DDVS_CFG_2

- **Name:** DDVS_CFG_2 Register

- **Description:** DDVS_CFG_2 Register
- **Base Address:** 0x4000E800
- **Offset:** 0x8
- **Reset Value:** 0x00000000

Table 3-237 DDVS_CFG_2 Register

Bits	Field Name	RW	Reset	Description
31:20	CONF_TARGET_CNT	RW	0x0	Target counter number
19:16	RSVD	R	0x0	Reserved bits
15:4	CONF_THRESHOLD_FAST	RW	0x0	Drift threshold that changes lock state to re-adjust vref
3:0	RSVD	R	0x0	Reserved bits

3.7.8.4 DDVS_RINGO_CNT_01

- **Name:** DDVS_RINGO_CNT_01 Register
- **Description:** DDVS_RINGO_CNT_01 Register
- **Base Address:** 0x4000E800
- **Offset:** 0xC
- **Reset Value:** 0x00000000

Table 3-238 DDVS_RINGO_CNT_01 Register

Bits	Field Name	RW	Reset	Description
31:20	RINGO_0_CNT	R	0x0	Ringo 0 counter
19:16	RSVD	R	0x0	Reserved bits
15:4	RINGO_1_CNT	R	0x0	Ringo 1 counter
3:0	RSVD	R	0x0	Reserved bits

3.7.8.5 DDVS_RINGO_CNT_23

- **Name:** DDVS_RINGO_CNT_23 Register
- **Description:** DDVS_RINGO_CNT_23 Register
- **Base Address:** 0x4000E800
- **Offset:** 0x10
- **Reset Value:** 0x00000000

Table 3-239 DDVS_RINGO_CNT_23 Register

Bits	Field Name	RW	Reset	Description
31:20	RINGO_2_CNT	R	0x0	Ringo 2 counter
19:16	RSVD	R	0x0	Reserved bits
15:4	RINGO_3_CNT	R	0x0	Ringo 3 counter
3:0	RSVD	R	0x0	Reserved bits

3.7.8.6 DDVS_FSM

- **Name:** DDVS_FSM Register
- **Description:** DDVS_FSM Register
- **Base Address:** 0x4000E800
- **Offset:** 0x14
- **Reset Value:** 0x00000000

Table 3-240 DDVS_FSM Register

Bits	Field Name	RW	Reset	Description
31:29	CURR	R	0x0	FSM current state **Value:** • 0x0: initial state • 0x1: vref plus 1 • 0x2: vref minus 1 • 0x3: lock • 0x4: An error occurs when vref reaches the maximum but still does not meet the system requirement. • Others: not used
28:0	RSVD	R	0x0	Reserved bits

3.7.8.7 DDVS_CLK_CTRL

- **Name:** DDVS_CLK_CTRL Register
- **Description:** DDVS_CLK_CTRL Register
- **Base Address:** 0x4000E800
- **Offset:** 0x40
- **Reset Value:** 0x00000000

Table 3-241 DDVS_CLK_CTRL Register

Bits	Field Name	RW	Reset	Description
31:10	RSVD	R	0x0	Reserved bits
9:8	CLK_SEL	RW	0x0	**Value:** • 0x0: xo_32MHz • 0x1: xo_16MHz • 0x2: sys_32MHz • 0x3: sys_16MHz
7:1	RSVD	R	0x0	Reserved bits
0	CLK_EN	RW	0x0	**Value:** • 0x0: Disable CLK. • 0x1: Enable CLK.

3.7.9 eFuse

3.7.9.1 TPGM

- **Name:** TPGM Register
- **Description:** TPGM Register
- **Base Address:** 0x40016000
- **Offset:** 0x0
- **Reset Value:** 0x2C0000F1

Table 3-242 TPGM Register

Bits	Field Name	RW	Reset	Description
31:24	WRITE_INTERVAL	RW	0x2C	Interval between two bits in eFuse, number of APB CLK cycles.
21:16	CRC_CHECK_LEN	RW	0x0	eFuse CRC check length , unit : word, this value must between 1 and 60
15:13	RSVD	R	0x0	Reserved bits
12	MAIN_OR_BACKUP	RW	0x0	If this bit set to "1", the eFuse read data through APB bus(such as MCU and other hardware module) will be the result of main info OR backup info#the read address on APB bus must a main info address.

Bits	Field Name	RW	Reset	Description
				If this bit set to "0", the eFuse read data through APB bus will be the result of the actual data in eFuse, the address on APB bus can be a main info address or backup address.
11:0	TIME	RW	0xF1	eFuse programming time, tpro = TIME*T, where T is clock period; typically tpro = 10000 ns, and 9000 ns < tpro < 11000 ns

3.7.9.2 PGENB

- **Name:** PGENB Register
- **Description:** PGENB Register
- **Base Address:** 0x40016000
- **Offset:** 0x4
- **Reset Value:** 0x00000018

Table 3-243 PGENB Register

Bits	Field Name	RW	Reset	Description
31:6	RSVD	R	0x0	Reserved bits
5:0	PGMEN_CTRL	RW	0x18	eFuse AVDD to PGMEN setup time or hold time

3.7.9.3 TEST_MODE

- **Name:** TEST_MODE Register
- **Description:** TEST_MODE Register
- **Base Address:** 0x40016000
- **Offset:** 0x8
- **Reset Value:** 0x00000000

Table 3-244 TEST_MODE Register

Bits	Field Name	RW	Reset	Description
31:16	RSVD	R	0x0	Reserved bits
15:0	TEST_MODE	R	0x0	test_mode read from eFuse

3.7.9.4 OPERATION

- **Name:** OPERATION Register

- **Description:** OPERATION Register
- **Base Address:** 0x40016000
- **Offset:** 0xC
- **Reset Value:** 0x00000000

Table 3-245 OPERATION Register

Bits	Field Name	RW	Reset	Description
31:5	RSVD	R	0x0	Reserved bits
4	RD_TEST_MODE	RW	0x0	This bit write "1" then efuse ctrl start to read eFuse test mode from efuse.
3	READ_TRIM	RW	0x0	This bit write "1", then eFuse ctrl start to read analog trim from efuse.
2	CRC_CHECK	RW	0x0	This bit write "1", then eFuse ctrl start to read the special eFuse addrress , and calculate this value ‘s CRC vaule.
1	INIT_CHECK	RW	0x0	This bit write "1", then eFuse ctrl start to read the whole eFuse vaule, at the same time check this vaule with "0".
0	WRITE_KEYRAM	RW	0x0	Write 1 to start read eFuse key.

3.7.9.5 STAT

- **Name:** STAT Register
- **Description:** STAT Register
- **Base Address:** 0x40016000
- **Offset:** 0x10
- **Reset Value:** 0x00000000

Table 3-246 STAT Register

Bits	Field Name	RW	Reset	Description
31:8	RSVD	R	0x0	Reserved bits
7	TEST_MODE_DONE	R	0x0	1: indicate test_mode, read from eFuse has done.
6	WRITE_DONE	R	0x0	1: indicate eFuse one word wirte done, note : read clear
5	CRC_CHECK_DONE	R	0x0	1: indicate efuse crc check done, note : read clear
4	INIT_SUCCESS	R	0x0	1: indicate eFuse initial value check success, that mean the efuse values are all "0",

Bits	Field Name	RW	Reset	Description
				note : read clear
3	INIT_DONE	R	0x0	1: indicate eFuse initial value check done, note : read clear
2	TRIM_CRC_SUCCESS	R	0x0	1: indicate crc check success. 0: indicate crc check fail. this bit is valid only when read_trim_done bit is "1"
1	READ_TRIM_DONE	R	0x0	1: indicate trim read from eFuse has done, and crc check also has done note : read clear
0	WRITE_KEYRAM_BUSY	R	0x0	1: indicate write keyram operation is in processing. 0: indicate write keyram operation is finished or in idle.

3.7.9.6 KEY_MASK

- **Name:** KEY_MASK Register
- **Description:** KEY_MASK Register
- **Base Address:** 0x40016000
- **Offset:** 0x14
- **Reset Value:** 0x00000000

Table 3-247 KEY_MASK Register

Bits	Field Name	RW	Reset	Description
31:0	KEY_MASK	RW	0x0	Keyram mask

3.7.9.7 CRC_ADDR

- **Name:** CRC_ADDR Register
- **Description:** CRC_ADDR Register
- **Base Address:** 0x40016000
- **Offset:** 0x18
- **Reset Value:** 0x00000000

Table 3-248 CRC_ADDR Register

Bits	Field Name	RW	Reset	Description
31:12	RSVD	R	0x0	Reserved bits

Bits	Field Name	RW	Reset	Description
11:0	CHECK_ADDR	RW	0x0	crc check start address, must be a main info address.

3.7.9.8 CRC_OUTPUT

- **Name:** CRC_OUTPUT Register
- **Description:** CRC_OUTPUT Register
- **Base Address:** 0x40016000
- **Offset:** 0x1C
- **Reset Value:** 0x00000000

Table 3-249 CRC_OUTPUT Register

Bits	Field Name	RW	Reset	Description
31:0	VALUE	R	0x0	eFuse crc check crc value output

3.7.9.9 TRIM_ADDR

- **Name:** TRIM_ADDR Register
- **Description:** TRIM_ADDR Register
- **Base Address:** 0x40016000
- **Offset:** 0x20
- **Reset Value:** 0x00000000

Table 3-250 TRIM_ADDR Register

Bits	Field Name	RW	Reset	Description
31:12	RSVD	R	0x0	Reserved bits
11:0	START_ADDR	RW	0x0	eFuse read trim start address, must be a main info address.

3.7.9.10 TRIM_LEN

- **Name:** TRIM_LEN Register
- **Description:** TRIM_LEN Register
- **Base Address:** 0x40016000
- **Offset:** 0x24
- **Reset Value:** 0x00000000

Table 3-251 TRIM_LEN Register

Bits	Field Name	RW	Reset	Description
31:5	RSVD	R	0x0	Reserved bits
4:0	LENGTH	RW	0x0	eFuse read trim length , unit : word, this vaule must between 1 and 20

3.7.9.11 TRIM[n]

- **Name:** TRIM[n] Register
- **Description:** TRIM[n] Register
- **Base Address:** 0x40016000
- **Offset:** 0x28+n*0x4 (n=0,1,2,…,13)
- **Reset Value:** 0x00000000

Table 3-252 TRIM[n] Register

Bits	Field Name	RW	Reset	Description
31:0	TRIM	R	0x0	eFuse the nth read trim (n=0,1,2,…,13)

4 Peripherals

4.1 Pin Mux

4.1.1 Introduction

The device has a configurable pin multiplexing module (Pin Mux) which can bring different peripherals on different GPIOs.

4.1.2 Functional Description

The pin multiplexing choices for all pads are shown in:

- Table 4-1 and Table 4-2 (GPIO_0–GPIO_15)
- Table 4-3 and Table 4-4 (GPIO_16–GPIO_33)
- Table 4-5 (MSIO_0–MSIO_7)
- Table 4-6 (AON_GPIO_0–AON_GPIO_7)

There are eight mux choices (from MUX_0 to MUX_7) in the pin mux tables as follows.

Table 4-1 Pin multiplexing for GPIO_0–GPIO_7

IO Name	MUX_0	MUX_1	MUX_2	MUX_3	MUX_4	MUX_5	MUX_6	MUX_7	MUX_8
GPIO_0	SWD_CLK	I2C0_SCL	I2C1_SCL	UART1_RTS	UART0_TX	UART1_TX	UART0_RTS	PDM_CLKO	GPIO_A0
GPIO_1	SWD_IO	I2C0_SDA	I2C1_SDA	UART1_CTS	UART0_RX	UART1_RX	UART0_CTS	PDM_DI	GPIO_A1
GPIO_2	UART0_CTS	PDM_CLKO	SWO	SPI_S_CS_N	I2C0_SDA	PWM0_A	FERP_GPIO_Trig	QSPI_M2_IO_2	GPIO_A2
GPIO_3	UART0_TX	PDM_DI	SPI_M_CLK	SPI_S_CLK	SPI_M_CS1_N	PWM0_B	DSPI_SCK	QSPI_M2_CLK	GPIO_A3
GPIO_4	UART0_RX	I2C2_SDA	SPI_M_MOSI	SPI_S_MISO	SPI_M_CS0_N	PWM0_C	DSPI_MOSI	QSPI_M2_IO_0	GPIO_A4
GPIO_5	UART0_RTS	I2C2_SCL	SPI_M_MISO	SPI_S_MOSI	SPI_M_MISO	I2C0_SCL	DSPI_MISO	QSPI_M2_IO_1	GPIO_A5
GPIO_6	I2S_WS	I2S_S_WS	SPI_M_CS0_N	UART1_RX	SPI_M_MOSI	I2C0_SDA	DSPI_NSS	QSPI_M2_CS_N	GPIO_A6
GPIO_7	I2S_TX_SDO	I2S_S_TX_SDO	SPI_M_CS1_N	UART1_TX	SPI_M_CLK	PWM1_A	DSPI_DCX	QSPI_M2_IO_3	GPIO_A7

Table 4-2 Pin multiplexing for GPIO_8–GPIO_15

IO Name	MUX_0	MUX_1	MUX_2	MUX_3	MUX_4	MUX_5	MUX_6	MUX_7	MUX_8
GPIO_8	QSPI_M0_IO_0	I2C1_SDA	QSPI_M1_IO_0	UART1_RX	PWM0_A	PWM1_B	PWM0_C		GPIO_A8
GPIO_9	QSPI_M0_CLK	I2C1_SCL	QSPI_M1_CLK	UART1_TX	PWM0_B	PWM1_C	PWM1_A		GPIO_A9

IO Name	MUX_0	MUX_1	MUX_2	MUX_3	MUX_4	MUX_5	MUX_6	MUX_7	MUX_8
GPIO_10	I2S_RX_SDI	I2S_S_RX_SDI	UART0_TX	I2C3_SCL	I2C0_SCL	PWM1_B		QSPI_M1_IO_2	GPIO_A10
GPIO_11	I2S_SCLK	I2S_S_SCLK	UART0_RX	I2C3_SDA	I2C0_SDA	PWM1_C	SWO	QSPI_M1_IO_3	GPIO_A11
GPIO_12	QSPI_M0_IO_3	SPI_M_CLK	QSPI_M1_IO_3	UART2_TX	I2S_WS	I2S_S_WS	SPI_S_CS_N		GPIO_A12
GPIO_13	QSPI_M0_IO_2	SPI_M_MOSI	QSPI_M1_IO_2	UART2_RX	I2S_TX_SDO	I2S_S_TX_SDO	SPI_S_CLK		GPIO_A13
GPIO_14	QSPI_M0_IO_1	SPI_M_MISO	QSPI_M1_IO_1	UART2_RTS	I2S_RX_SDI	I2S_S_RX_SDI	SPI_S_MISO		GPIO_A14
GPIO_15	QSPI_M0_CS_N	SPI_M_CS0_N	QSPI_M1_CS_N	UART2_CTS	I2S_SCLK	I2S_S_SCLK	SPI_S_MOSI	FERP_GPIO_Trig	GPIO_A15

Table 4-3 Pin multiplexing for GPIO_16–GPIO_23

IO Name	MUX_0	MUX_1	MUX_2	MUX_3	MUX_4	MUX_5	MUX_6	MUX_7	MUX_8
GPIO_16	SPI_M_MISO	SPI_S_MOSI	DSPI_MISO	I2S_RX_SDI	I2S_S_RX_SDI	QSPI_M1_IO_1	FERP_GPIO_Trig	UART3_RTS	GPIO_B0
GPIO_17	SPI_M_CS0_N	SPI_S_CS_N	DSPI_NSS	I2S_SCLK	I2S_S_SCLK	QSPI_M1_IO_2		UART3_CTS	GPIO_B1
GPIO_18	QSPI_M0_CLK	SPI_M_CS1_N		I2C3_SCL	UART3_TX		I2C1_SCL	QSPI_M1_CLK	GPIO_B2
GPIO_19	QSPI_M0_IO_0			I2C3_SDA	UART3_RX		I2C1_SDA	QSPI_M1_IO_1	GPIO_B3
GPIO_20	I2C2_SCL			QSPI_M2_IO_0	UART3_RTS	PDM_CLKO	df_ant_switch_sel_5	QSPI_M1_IO_0	GPIO_B4
GPIO_21	I2C2_SDA		DSPI_DCX	QSPI_M2_CS_N	UART3_CTS	PDM_DI	df_ant_switch_sel_6	QSPI_M1_CS_N	GPIO_B5
GPIO_22	SPI_S_CLK	I2S_WS	DSPI_SCK	QSPI_M2_CLK	I2S_S_WS	SPI_M_CLK	df_ant_switch_sel_0	QSPI_M1_CLK	GPIO_B6
GPIO_23	SPI_S_MISO	I2S_TX_SDO	DSPI_MOSI	QSPI_M2_IO_1	I2S_S_TX_SDO	SPI_M_MOSI	df_ant_switch_sel_1	QSPI_M1_IO_1	GPIO_B7

Table 4-4 Pin multiplexing for GPIO_24–GPIO_33

IO Name	MUX_0	MUX_1	MUX_2	MUX_3	MUX_4	MUX_5	MUX_6	MUX_7	MUX_8
GPIO_24	SPI_M_CLK	SPI_S_CLK	DSPI_SCK	I2S_WS	I2S_S_WS	QSPI_M1_CLK		UART3_TX	GPIO_B8
GPIO_25	SPI_M_MOSI	SPI_S_MISO	DSPI_MOSI	I2S_TX_SDO	I2S_S_TX_SDO	QSPI_M1_IO_0		UART3_RX	GPIO_B9
GPIO_26	I2C1_SDA	UART1_RX	I2C0_SDA	PWM0_C	PWM1_C	UART0_RX	df_ant_switch_sel_1	QSPI_M2_IO_0	GPIO_B10
GPIO_27	PDM_CLKO	UART1_RTS			I2C2_SCL	UART0_RTS	df_ant_switch_sel_2	QSPI_M2_IO_1	GPIO_B11
GPIO_28	PDM_DI	UART1_CTS			I2C2_SDA	UART0_CTS	df_ant_switch_sel_3	QSPI_M2_IO_2	GPIO_B12
GPIO_29	FERP_GPIO_Trig			PWM1_A	PWM0_C		df_ant_switch_sel_4	QSPI_M2_IO_3	GPIO_B13
GPIO_30	I2C1_SCL	UART1_TX	I2C0_SCL	PWM0_B	PWM1_B	UART0_TX		QSPI_M2_CLK	GPIO_B14
GPIO_31	SPI_M_CS1_N	FERP_GPIO_Trig	DSPI_DCX	PWM0_A	PWM1_A	QSPI_M1_IO_3		df_ant_switch_sel_0	GPIO_B15
GPIO_32	SPI_S_MOSI	I2S_RX_SDI	DSPI_MISO	QSPI_M2_IO_2	I2S_S_RX_SDI	SPI_M_MISO	df_ant_switch_sel_2	QSPI_M1_IO_2	GPIO_C0
GPIO_33	SPI_S_CS_N	I2S_SCLK	DSPI_NSS	QSPI_M2_IO_3	I2S_S_SCLK	SPI_M_CS0_N	df_ant_switch_sel_3	QSPI_M1_IO_3	GPIO_C1

Table 4-5 Pin multiplexing for MSIO_0–MSIO_7

IO Name	MUX_0	MUX_1	MUX_2	MUX_3	MUX_4	MUX_5	MUX_6	MUX_7	MUX_8
MSIO_0	PWM0_A	UART0_TX	UART1_TX	I2C0_SCL	I2C2_SCL	I2C1_SCL	I2C3_SCL	df_ant_switch_sel_0	MSIO_0
MSIO_1	PWM0_B	UART0_RX	UART1_RX	I2C0_SDA	I2C2_SDA	I2C1_SDA	I2C3_SDA	df_ant_switch_sel_1	MSIO_1
MSIO_2	PWM0_C							df_ant_switch_sel_2	MSIO_2
MSIO_3	PWM1_A	UART0_RTS	UART1_RTS	I2C1_SCL	I2C3_SCL	I2C0_SCL	I2C2_SCL	df_ant_switch_sel_3	MSIO_3
MSIO_4	PWM1_B	UART0_CTS	UART1_CTS	I2C1_SDA	I2C3_SDA	I2C0_SDA	I2C2_SDA	df_ant_switch_sel_4	MSIO_4
MSIO_5	PWM1_C			UART2_RTS	UART3_TX	UART2_TX	UART3_RTS	df_ant_switch_sel_5	MSIO_5
MSIO_6				UART2_CTS	UART3_RX	UART2_RX	UART3_CTS	df_ant_switch_sel_6	MSIO_6
MSIO_7									MSIO_7

Table 4-6 Pin multiplexing for AON_GPIO_0–AON_GPIO_7

IO Name	MUX_0	MUX_1	MUX_2	MUX_3	MUX_4	MUX_5	MUX_6	MUX_7	MUX_8
AON_GPIO_0		QSPI_M0_IO_2		I2C2_SCL	I2C1_SCL			df_ant_switch_sel_5	AON_GPIO_0
AON_GPIO_1		QSPI_M0_IO_3	DSPI_NSS	I2C2_SDA	I2C1_SDA	QSPI_M1_CS_N		DSPI_DCX	AON_GPIO_1
AON_GPIO_2	UART2_CTS	QSPI_M0_CS_N	I2S_WS	I2S_S_WS	UART3_CTS	PWM0_C		DSPI_NSS	AON_GPIO_2
AON_GPIO_3	UART2_TX	QSPI_M0_IO_0	I2S_TX_SDO	I2S_S_TX_SDO	UART3_TX	PWM1_A		DSPI_MOSI	AON_GPIO_3
AON_GPIO_4	UART2_RX	QSPI_M0_IO_1	I2S_RX_SDI	I2S_S_RX_SDI	UART3_RX	PWM1_B		DSPI_MISO	AON_GPIO_4
AON_GPIO_5	UART2_RTS	QSPI_M0_CLK	I2S_SCLK	I2S_S_SCLK	UART3_RTS	PWM1_C		DSPI_SCK	AON_GPIO_5
AON_GPIO_6							df_ant_switch_sel_5	df_ant_switch_sel_6	AON_GPIO_6
AON_GPIO_7							df_ant_switch_sel_6	QSPI_M2_CS_N	AON_GPIO_7

4.1.3 Registers

4.1.3.1 DPAD_MUX_CTRL_00_07

- **Name:** GPIO 00_07 Mux Control Register
- **Description:** This register controls GPIO_0–GPIO_7 mux.
- **Base Address:** 0x4000EE00
- **Offset:** 0x060
- **Reset Value:** 0x88888888

Table 4-7 DPAD_MUX_CTRL_00_07 Register

Bits	Field Name	RW	Reset	Description
31:28	DPAD_MUX_SEL_07	RW	0x7	Value for GPIO_7 mux
27:24	DPAD_MUX_SEL_06	RW	0x7	Value for GPIO_6 mux
23:20	DPAD_MUX_SEL_05	RW	0x7	Value for GPIO_5 mux
19:16	DPAD_MUX_SEL_04	RW	0x7	Value for GPIO_4 mux
15:12	DPAD_MUX_SEL_03	RW	0x7	Value for GPIO_3 mux

Bits	Field Name	RW	Reset	Description
11:8	DPAD_MUX_SEL_02	RW	0x7	Value for GPIO_2 mux
7:4	DPAD_MUX_SEL_01	RW	0x0	Value for GPIO_1 mux
3:0	DPAD_MUX_SEL_00	RW	0x0	Value for GPIO_0 mux

4.1.3.2 DPAD_MUX_CTRL_08_15

- **Name:** GPIO 08_15 Mux Control Register
- **Description:** This register controls GPIO_8–GPIO_15 mux.
- **Base Address:** 0x4000EE00
- **Offset:** 0x064
- **Reset Value:** 0x88888888

Table 4-8 DPAD_MUX_CTRL_08_15 Register

Bits	Field Name	RW	Reset	Description
31:28	DPAD_MUX_SEL_15	RW	0x7	Value for GPIO_15 mux
27:24	DPAD_MUX_SEL_14	RW	0x7	Value for GPIO_14 mux
23:20	DPAD_MUX_SEL_13	RW	0x7	Value for GPIO_13 mux
19:16	DPAD_MUX_SEL_12	RW	0x7	Value for GPIO_12 mux
15:12	DPAD_MUX_SEL_11	RW	0x7	Value for GPIO_11 mux
11:8	DPAD_MUX_SEL_10	RW	0x7	Value for GPIO_10 mux
7:4	DPAD_MUX_SEL_09	RW	0x7	Value for GPIO_9 mux
3:0	DPAD_MUX_SEL_08	RW	0x7	Value for GPIO_8 mux

4.1.3.3 DPAD_MUX_CTRL_16_23

- **Name:** GPIO 16_23 Mux Control Register
- **Description:** This register controls GPIO_16–GPIO_23 mux.
- **Base Address:** 0x4000EE00
- **Offset:** 0x068
- **Reset Value:** 0x88888888

Table 4-9 DPAD_MUX_CTRL_16_23 Register

Bits	Field Name	RW	Reset	Description
31:28	DPAD_MUX_SEL_23	RW	0x7	Value for GPIO_23 mux
27:24	DPAD_MUX_SEL_22	RW	0x7	Value for GPIO_22 mux

Bits	Field Name	RW	Reset	Description
23:20	DPAD_MUX_SEL_21	RW	0x7	Value for GPIO_21 mux
19:16	DPAD_MUX_SEL_20	RW	0x7	Value for GPIO_20 mux
15:12	DPAD_MUX_SEL_19	RW	0x7	Value for GPIO_19 mux
11:8	DPAD_MUX_SEL_18	RW	0x7	Value for GPIO_18 mux
7:4	DPAD_MUX_SEL_17	RW	0x7	Value for GPIO_17 mux
3:0	DPAD_MUX_SEL_16	RW	0x7	Value for GPIO_16 mux

4.1.3.4 DPAD_MUX_CTRL_24_31

- **Name:** GPIO 24_31 mux control register
- **Description:** This register controls GPIO_24–GPIO_31 mux.
- **Base Address:** 0x4000EE00
- **Offset:** 0x06C
- **Reset Value:** 0x88888888

Table 4-10 DPAD_MUX_CTRL_24_31 Register

Bits	Field Name	RW	Reset	Description
31:28	DPAD_MUX_SEL_31	RW	0x7	Value for GPIO_31 mux
27:24	DPAD_MUX_SEL_30	RW	0x7	Value for GPIO_30 mux
23:20	DPAD_MUX_SEL_29	RW	0x7	Value for GPIO_29 mux
19:16	DPAD_MUX_SEL_28	RW	0x7	Value for GPIO_28 mux
15:12	DPAD_MUX_SEL_27	RW	0x7	Value for GPIO_27 mux
11:8	DPAD_MUX_SEL_26	RW	0x7	Value for GPIO_26 mux
7:4	DPAD_MUX_SEL_25	RW	0x7	Value for GPIO_25 mux
3:0	DPAD_MUX_SEL_24	RW	0x7	Value for GPIO_24 mux

4.1.3.5 AON_PAD_MUX_CTRL

- **Name:** AON GPIO Pin Mux Control Register
- **Description:** This register controls the AON GPIO pin mux.
- **Base Address:** 0x4000EE00
- **Offset:** 0x074
- **Reset Value:** 0x88888888

Table 4-11 AON_PAD_MUX_CTRL Register

Bits	Field Name	RW	Reset	Description
31:28	AON_PAD_MUX_SEL_07	RW	0x8	Value for AON_GPIO_7 mux
27:24	AON_PAD_MUX_SEL_06	RW	0x8	Value for AON_GPIO_6 mux
23:20	AON_PAD_MUX_SEL_05	RW	0x8	Value for AON_GPIO_5 mux
19:16	AON_PAD_MUX_SEL_04	RW	0x8	Value for AON_GPIO_4 mux
15:12	AON_PAD_MUX_SEL_03	RW	0x8	Value for AON_GPIO_3 mux
11:8	AON_PAD_MUX_SEL_02	RW	0x8	Value for AON_GPIO_2 mux
7:4	AON_PAD_MUX_SEL_01	RW	0x8	Value for AON_GPIO_1 mux
3:0	AON_PAD_MUX_SEL_00	RW	0x8	Value for AON_GPIO_0 mux

4.1.3.6 MSIO_APAD_MUX_CTRL

- **Name:** MSIO Pin Mux Control Register
- **Description:** This register controls the MSIO pin mux.
- **Base Address:** 0x4000EE00
- **Offset:** 0x078
- **Reset Value:** 0x88888888

Table 4-12 MSIO_APAD_MUX_CTRL Register

Bits	Field Name	RW	Reset	Description
31:28	MSIO_MUX_SEL_07	RW	0x8	Value for MSIO_7 mux
27:24	MSIO _MUX_SEL_06	RW	0x8	Value for MSIO_6 mux
23:20	MSIO _MUX_SEL_05	RW	0x8	Value for MSIO_5 mux
19:16	MSIO _MUX_SEL_04	RW	0x8	Value for MSIO_4 mux
15:12	MSIO _MUX_SEL_03	RW	0x8	Value for MSIO_3 mux
11:8	MSIO _MUX_SEL_02	RW	0x8	Value for MSIO_2 mux
7:4	MSIO _MUX_SEL_01	RW	0x8	Value for MSIO_1 mux
3:0	MSIO _MUX_SEL_00	RW	0x8	Value for MSIO_0 mux

4.1.3.7 DPAD_MUX_CTRL_32_33

- **Name:** GPIO 32_33 Mux Control Register
- **Description:** This register controls the GPIO_32– GPIO_33 mux.
- **Base Address:** 0x4000EE00
- **Offset:** 0x070

- **Reset Value:** 0x00000088

Table 4-13 DPAD_MUX_CTRL_32_33 Register

Bits	Field Name	RW	Reset	Description
31:8	RSVD	R		Reserved bits
7:4	DPAD_MUX_SEL_33	RW	0x8	Value for GPIO_33 mux
3:0	DPAD_MUX_SEL_32	RW	0x8	Value for GPIO_32 mux

4.2 I/O

The GPIOs can be connected to various digital and mixed-signal interfaces, and the features are as follows:

- Up to 34 General Purpose I/Os (GPIOs): normal digital GPIOs in MCU domain
- 8 AON_GPIOs: digital GPIOs in the AON domain
- 8 MSIOs: digital and analog mixed-signal GPIOs, no interrupt
- All I/Os have internal pull up/down registers
- All I/Os can retain their last state when the system enters the sleep mode.
- GPIO_0–GPIO_15, GPIO_18–GPIO_23 and GPIO_32–GPIO_33 are supplied by VDDIO_1.
- MSIO_0–MSIO_7 are supplied by VBATL.
- AON_GPIO_0–AON_GPIO_7, GPIO_16–GPIO_27 and GPIO_24–GPIO_31 are supplied by VDDIO_0.

4.2.1 GPIO

4.2.1.1 Introduction

The device has up to 34 individually configurable GPIOs that can serve as peripherals or digital input/output.

4.2.1.2 Main Features

- Up to 34 digital GPIOs: GPIO_0–GPIO_15, GPIO_16–GPIO_23 and GPIO_32–GPIO_33 are supplied by VDDIO_1. GPIO_24–GPIO_31 are supplied by VDDIO_0.
- GPIO default state is input mode with pull-down.
- All GPIOs can be configured as being triggered by high level, low level, rising edge, falling edge, or both edges.

4.2.1.3 Functional Description

The 34 digital GPIOs can be used:

- For general-purpose input/output.
- To multiplex different peripherals on different output pins of the package.
- To multiplex debug signals from different blocks of the SoC.

A functional drawing of the GPIO pads is provided in Figure 4-1. The digital GPIO pads can be configured to set direction, enable pull-up/pull-down resistors, and enable output retention. GPIOs can also be read or written by firmware for applications that need direct access through the GPIOx registers.

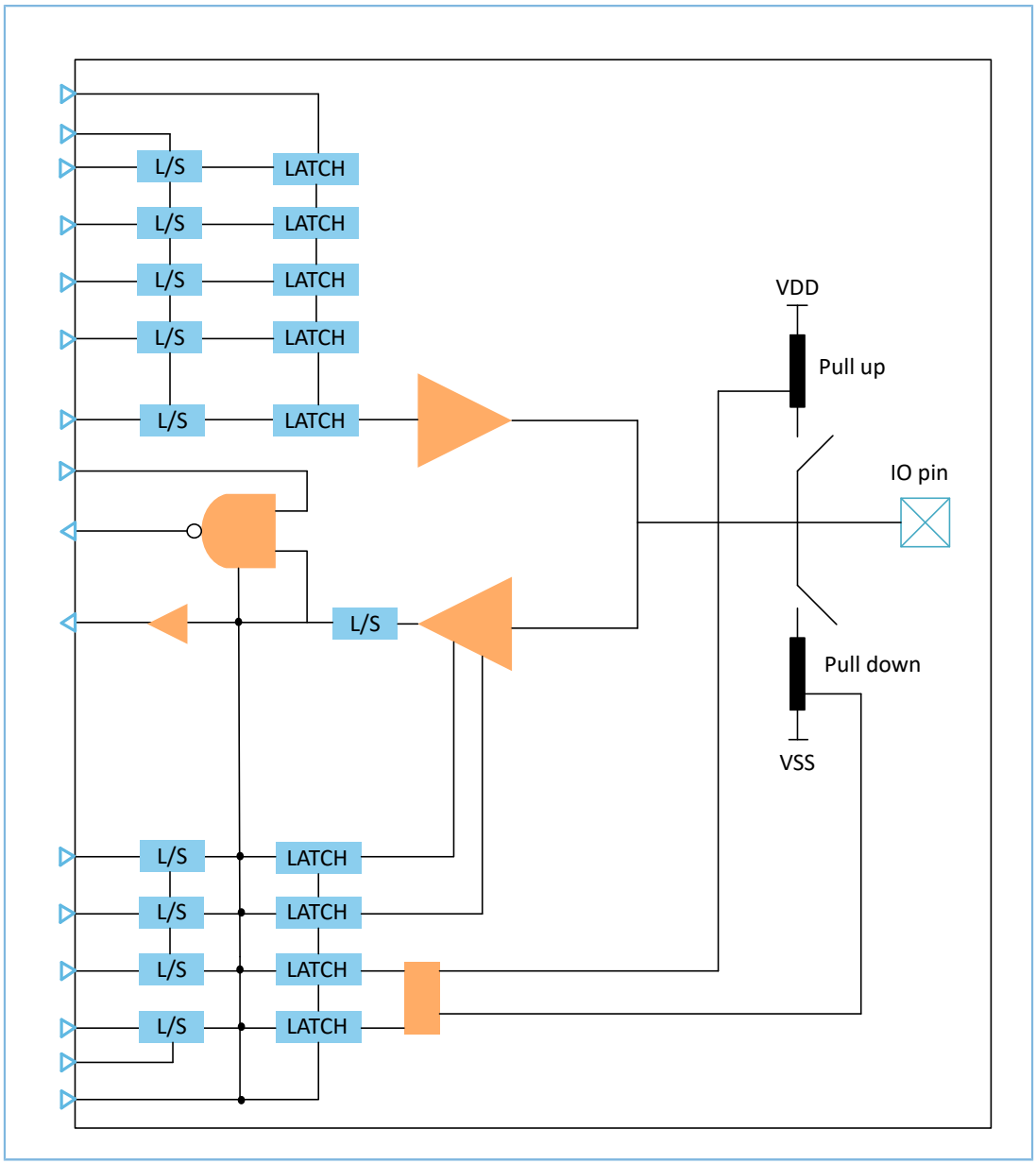

Figure 4-1 GPIO pad diagram

The general rule for enabling the on-chip pull-up/pull-down resistors is as follows:

- If a GPIO is unused, enable a pull-down resistor.
- If a GPIO is configured as an output, disable the pull-up/pull-down resistor.
- If a GPIO is used as an input and is being actively driven high or low by an external device, disable the pull-up/pull-down resistor (the same thing goes when there is an external pull-up/pull-down resistor on the board. In this case, the on-chip resistor should always be disabled).

4.2.1.4 Registers

4.2.1.4.1 DPAD_R_EN_BUS_0

- **Name:** GPIO Pull-Up/Down Resistor Enable Register
- **Description:** This register decides whether to use pull-up/down resistors.
- **Base Address:** 0x4000EE00
- **Offset:** 0x000
- **Reset Value:** 0xFFFFFFFF

Table 4-14 DPAD_R_EN_BUS_0 Register

Bits	Field Name	RW	Reset	Description
31:0	DPAD_R_EN	RW	0x0	Resistor enable control of GPIO_31–GPIO_0 **Value**: • 0x0: Disable pull-up/down resistor. • 0x1: Enable pull-up/down resistor.

4.2.1.4.2 DPAD_RTYP_BUS_0

- **Name:** GPIO Resistor Type Control Register
- **Description:** This register sets the resistor type to either pull-up or pull-down.
- **Base Address:** 0x4000EE00
- **Offset:** 0x004
- **Reset Value:** 0x00000000

Table 4-15 DPAD_RTYP_BUS_0 Register

Bits	Field Name	RW	Reset	Description
31:0	DPAD_RTYP	RW	0x0	Resistor type of GPIO_31–GPIO_0 **Value**: • 0x0: Pull down. • 0x1: Pull up.

4.2.1.4.3 DPAD_IN_EN_BUS_0

- **Name:** GPIO Input Enable Register
- **Description:** This register enables a GPIO as input.
- **Base Address:** 0x4000EE00

- **Offset:** 0x008
- **Reset Value:** 0xFFFFFFFF

Table 4-16 DPAD_IN_EN_BUS_0 Register

Bits	Field Name	RW	Reset	Description
31:0	DPAD_IN_EN	RW	0xFFFFFFFF	Active I/O inputs to GPIO_31–GPIO_0 **Value**: • 0x0: Disable inputs to pad. • 0x1: Enable inputs to pad.

4.2.1.4.4 DPAD_R_EN_BUS_1

- **Name:** GPIO Pull-Up/Down Resistor Enable Register
- **Description:** This register decides whether to use pull-up/down resistors.
- **Base Address:** 0x4000EE00
- **Offset:** 0x020
- **Reset Value:** 0x00000003

Table 4-17 DPAD_R_EN_BUS_1 Register

Bits	Field Name	RW	Reset	Description
31:2	RSVD	R		Reserved bits
1:0	DPAD_R_EN	RW	0x3	Resistor enable control of GPIO_33–GPIO_32. **Value**: • 0x0: Disable pull-up/down resistor. • 0x1: Enable pull-up/down resistor.

4.2.1.4.5 DPAD_RTYP_BUS_1

- **Name:** GPIO Pull-Up/Down Resistor Type Control Register
- **Description:** This register sets the resistor type to either pull-up or pull-down.
- **Base Address:** 0x4000EE00
- **Offset:** 0x024
- **Reset Value:** 0x00000000

Table 4-18 DPAD_RTYP_BUS_1 Register

Bits	Field Name	RW	Reset	Description
31:2	RSVD	R		Reserved bits

Bits	Field Name	RW	Reset	Description
1:0	DPAD_RTYP	RW	0x0	Resistor type of GPIO_33–GPIO_32 **Value**: • 0x0: Pull down. • 0x1: Pull up.

4.2.1.4.6 DPAD_IN_EN_BUS_1

- **Name:** GPIO Input Enable Register
- **Description:** This register enables a GPIO as input.
- **Base Address:** 0x4000E000
- **Offset:** 0x028
- **Reset Value:** 0x00000003

Table 4-19 DPAD_IN_EN_BUS_1 Register

Bits	Field Name	RW	Reset	Description
31:2	RSVD	R		Reserved bits
1:0	DPAD_IN_EN	RW	0x3	Active I/O inputs to the GPIO_33 – GPIO_32 **Value**: • 0x0: Disable inputs to pad. • 0x1: Enable inputs to pad.

4.2.1.4.7 DPAD_R_EN_MISC

- **Name:** XQSPI I/O Pull-Up/Down Resistor Enable Register
- **Description:** This register decides whether to use pull-up/down resistors.
- **Base Address:** 0x4000EE00
- **Offset:** 0x040
- **Reset Value:** 0x003F0000

Table 4-20 DPAD_R_EN_MISC Register

Bits	Field Name	RW	Reset	Description
31:22	RSVD	R		Reserved bits
21:18	XQSPI _IO	RW	0xF	**Value**: • 0x0: Disable pull-up/down resistor for XQSPI I/O. • 0x1: Enable pull-up/down resistor for XQSPI I/O.

Bits	Field Name	RW	Reset	Description
17	XQSPI_CS	RW	0x1	**Value**: • 0x0: Disable pull-up/down resistor for XQSPI CS. • 0x1: Enable pull-up/down resistor for XQSPI CS.
16	XQSPI_CLK	RW	0x0	**Value**: • 0x0: Disable pull-up/down resistor for XQSPI clk. • 0x1: Enable pull-up/down resistor for XQSPI clk.
15:0	RSVD	R		Reserved bits

4.2.1.4.8 DPAD_RTYP_MISC

- **Name:** XQSPI Pull-Up/Down Resistor Type Control Register
- **Description:** This register sets the resistor type to either pull-up or pull-down.
- **Base Address:** 0x4000EE00
- **Offset:** 0x044
- **Reset Value:** 0x003F0000

Table 4-21 DPAD_RTYP_MISC Register

Bits	Field Name	RW	Reset	Description
31:22	RSVD	R		Reserved bits
21:18	XQSPI_IO	RW	0xF	**Value**: • 0x0: Pull down. • 0x1: Pull up.
17	XQSPI_IO	RW	0x1	**Value**: • 0x0: Pull down. • 0x1: Pull up.
16	XQSPI_CLK	RW	0x1	**Value**: • 0x0: Pull down. • 0x1: Pull up.
15:0	RSVD	R		Reserved bits

4.2.1.4.9 DPAD_IN_EN_MISC

- **Name:** XQSPI Pin Input Enable Register
- **Description:** This register enables the input.
- **Base Address:** 0x4000E000

- **Offset:** 0x048
- **Reset Value:** 0x003F0000

Table 4-22 DPAD_IN_EN_MISC Register

Bits	Field Name	RW	Reset	Description
31:22	RSVD	R		Reserved bits
21:18	XQSPI_IO	RW	0xF	**Value**: • 0x0: Disable inputs to pad for XQSPI I/O. • 0x1: Enable inputs to pad for XQSPI I/O.
17	XQSPI_CS	RW	0x1	**Value**: • 0x0: Disable inputs to pad for XQSPI CS. • 0x1: Enable inputs to pad for XQSPI CS.
16	XQSPI_CLK	RW	0x1	**Value**: • 0x0: Disable inputs to pad for XQSPI clk. • 0x1: Enable inputs to pad for XQSPI clk.
15:0	RSVD	R		Reserved bits

4.2.1.4.10 GPIOx_DATA

- **Name:** GPIO Data Value Register
- **Description:** This register contains the input data and can write it to data output register.
- **Base Address:** 0x40010000 + x*0x1000
- **Offset:** 0x0000
- **Reset Value:** 0x00000000

Table 4-23 GPIOx_DATA Register

Bits	Field Name	RW	Reset	Description
31:16	RSVD	R		Reserved bits
15:0	DATA	RW	0x0	Data value [15:0]: • Read: Read the value that is sampled at pin • Write: Write the field to the GPIOx_DATA_OUT register.

Note:

GPIOx includes GPIOA (GPIO_0–GPIO_15), GPIOB (GPIO_16–GPIO_31), and GPIOC (GPIO_32–GPIO_33).

4.2.1.4.11 GPIOx_DATA_OUT

- **Name:** GPIO Data Output Register
- **Description:** This register contains the output data.
- **Base Address:** 0x40010000 + x*0x1000
- **Offset:** 0x0004
- **Reset Value:** 0x00000000

Table 4-24 GPIOx_DATA_OUT Register

Bits	Field Name	RW	Reset	Description
31:16	RSVD	R		Reserved bits
15:0	DATA_OUT	RW	0x0	Data output register value [15:0]: • Read: current value of data output register • Write: to data output register

4.2.1.4.12 GPIOx_OUT_EN

- **Name:** GPIO Output Enable Register
- **Description:** This register is used to enable the GPIO as output.
- **Base Address:** 0x40010000 + x*0x1000
- **Offset:** 0x0010
- **Reset Value:** 0x00000000

Table 4-25 GPIOx_OUT_EN Register

Bits	Field Name	RW	Reset	Description
31:16	RSVD	R		Reserved bits
15:0	EN	RW	0x0	Output enable set [15:0]: **Read**: • 0x0: As input • 0x1: As output **Write**: • 0x0: No effect • 0x1: Set the output enable bit.

4.2.1.4.13 GPIOx_OUT_CLR

- **Name:** GPIO Output Enable Clear Register
- **Description:** This register is used to disable output bit.

- **Base Address:** 0x40010000 + x*0x1000
- **Offset:** 0x0014
- **Reset Value:** 0x00000000

Table 4-26 GPIOx_OUT_CLR Register

Bits	Field Name	RW	Reset	Description
31:16	RSVD	R		Reserved bits
15:0	CLR	RW	0x0	Output enable clear [15:0]: **Read**: • 0x0: as input • 0x1: as output **Write**: • 0x0: No effect • 0x1: Clear the output enable bit.

4.2.1.4.14 GPIOx_INT_EN

- Name: GPIO Interrupt Enable Register
- Description: This register is used to enable GPIO interrupt.
- Base Address: 0x40010000 + x*0x1000
- Offset: 0x0020
- Reset Value: 0x00000000

Table 4-27 GPIOx_INT_EN Register

Bits	Field Name	RW	Reset	Description
31:16	RSVD	R		Reserved bits
15:0	EN	RW	0x0	Interrupt enable set [15:0]: **Read**: • 0x0: Interrupt disabled • 0x1: Interrupt enabled **Write**: • 0x0: No effect • 0x1: Set the interrupt enable bit.

4.2.1.4.15 GPIOx_INT_CLR

- **Name:** GPIO Interrupt Enable Clear Register
- **Description:** This register is used to disable GPIO interrupt.
- **Base Address:** 0x40010000 + x*0x1000
- **Offset:** 0x0024
- **Reset Value:** 0x00000000

Table 4-28 GPIOx_INT_CLR Register

Bits	Field Name	RW	Reset	Description
31:16	RSVD	R		Reserved bits
15:0	CLR	RW	0x0	Interrupt enable clear [15:0]: **Read**: • 0x0: Interrupt disabled • 0x1: Interrupt enabled **Write**: • 0x0: No effect • 0x1: Clear the interrupt enable bit.

4.2.1.4.16 GPIOx_INT_TYPE_EN

- **Name:** GPIO Interrupt Type Enable Register
- **Description:** This register is used to set GPIO interrupt type.
  - Interrupt type is 0 and interrupt polarity is 0: low-level trigger interrupt
  - Interrupt type is 0 and interrupt polarity is 1: high-level trigger interrupt
  - Interrupt type is 1 and interrupt polarity is 0: falling edge trigger interrupt
  - Interrupt type is 1 and interrupt polarity is 1: rising edge trigger interrupt
- **Base Address:** 0x40010000 + x*0x1000
- **Offset:** 0x0028
- **Reset Value:** 0x00000000

Table 4-29 GPIOx_INT_TYPE_EN Register

Bits	Field Name	RW	Reset	Description
31:16	RSVD	R		Reserved bits
15:0	EN	RW	0x0	Interrupt type set [15:0]: **Read**: • 0x0: Low or high level

Bits	Field Name	RW	Reset	Description
				• 0x1: Falling edge or rising edge **Write**: • 0x0: No effect • 0x1: Set the interrupt type bit.

4.2.1.4.17 GPIOx_INT_TYPE_CLR

- **Name:** GPIO Interrupt Type Clear Register
- **Description:** This register is used to disable GPIO interrupt type.
- **Base Address:** 0x40010000 + x*0x1000
- **Offset:** 0x002C
- **Reset Value:** 0x00000000

Table 4-30 GPIOx_INT_TYPE_CLR Register

Bits	Field Name	RW	Reset	Description
31:16	RSVD	R		Reserved bits
15:0	CLR	RW	0h 0	Interrupt type clear [15:0]: **Read**: • 0x0: Low or high level • 0x1: Falling edge or rising edge **Write**: • 0x0: No effect • 0x1: Clear the interrupt type bit.

4.2.1.4.18 GPIOx_INT_POL_EN

- **Name:** GPIO Interrupt Polarity Enable Register
- **Description:** This register is used to enable GPIO interrupt polarity.
- **Base Address:** 0x40010000 + x*0x1000
- **Offset:** 0x0030
- **Reset Value:** 0x00000000

Table 4-31 GPIOx_INT_POL_EN Register

Bits	Field Name	RW	Reset	Description
31:16	RSVD	R		Reserved bits

Bits	Field Name	RW	Reset	Description
15:0	EN	RW	0h 0	Polarity configuration [15:0]: **Read**: • 0x0: Low level or falling edge • 0x1: High level or rising edge **Write**: • 0x0: No effect • 0x1: Set the interrupt polarity bit.

4.2.1.4.19 GPIOx_INT_POL_CLR

- **Name:** GPIO Interrupt Polarity Disable Register
- **Description:** This register is used to disable GPIO interrupt polarity.
- **Base Address:** 0x40010000 + x*0x1000
- **Offset:** 0x0034
- **Reset Value:** 0x00000000

Table 4-32 GPIOx_INT_POL_CLR Register

Bits	Field Name	RW	Reset	Description
31:16	RSVD	R		Reserved bits
15:0	CLR	RW	0x0	Polarity configuration [15:0]: **Read**: • 0x0: Low level or falling edge • 0x1: High level or rising edge **Write**: • 0x0: No effect • 0x1: Clear the interrupt polarity bit.

4.2.1.4.20 GPIOx_INT_STAT

- **Name:** GPIO IRQ Status Register
- **Description:** This register contains GPIO interrupt status and is used to clear the interrupt request.
- **Base Address:** 0x40010000 + x*0x1000
- **Offset:** 0x0038
- **Reset Value:** 0x00000000

Table 4-33 GPIOx_INT_STAT Register

Bits	Field Name	RW	Reset	Description
31:16	RSVD	R		Reserved bits
15:0	STAT_CLR	RW	0x0	Write one to clear interrupt request: **Read**: • [15:0] IRQ status register **Write**: • 0x0: No effect • 0x1: To clear the interrupt request

4.2.1.5 Electrical Specifications

The electrical parameters for the GPIO_0–GPIO_33 are as follows:

Table 4-34 GPIO electrical specifications

Parameter	Description	Min.	Typ.	Max.	Unit
V_{IH}	Input high voltage	VDDIO x 0.7		VDDIO	V
V_{IL}	Input low voltage	VSSIO		VDDIO x 0.3	V
$V_{OH,L}$	Output high voltage, 4 mA, VDD ≥ 1.7 V	VDD – 0.4		VDD	V
$V_{OH,M}$	Output high voltage, 4 mA, VDD ≥ 2.5 V	VDD – 0.4		VDD	V
VOH,H	Output high voltage, 4 mA, VDD ≥ 3 V	VDD – 0.4		VDD	V
$V_{OL,L}$	Output low voltage, 4 mA, VDD ≥1.7 V	VSS		VSS + 0.4	V
$V_{OL,M}$	Output low voltage, 4 mA, VDD ≥ 2.5 V	VSS		VSS + 0.4	V
$V_{OL,H}$	Output low voltage, 4 mA, VDD ≥ 3 V	VSS		VSS + 0.4	V
I_{OL}	Current at VSS+0.4 V, output set low, VDD ≥1.7 V		4		mA
I_{OH}	Current at VDD-0.4 V, output set high, VDD ≥ 1.7 V		4		mA
R_{PU}	Pull-up resistance		18		kΩ
R_{PD}	Pull-down resistance		20		kΩ
C_{PAD}	Pad capacitance		5		pF

4.2.2 Always-On I/O

4.2.2.1 Introduction

The Always-on Watchdog(AON_WDT) is a 32-bit countdown hardware timer. When the AON_WDT counter counts to 0, a system reset signal is sent, and when the counter value decreases to the set warning value, an interrupt signal is generated. Because the AON_WDT timer is in the AON domain, it can keep running when the MCU is off.

4.2.2.2 Functional Description

- The LOCK register enables or disables write accesses to all other registers. This is to prevent malicious software from disabling the watchdog functionality. Writing a value of 0x15CC5A51 (LOCK.LOCK_SET[31:1]) enables write accesses to all other registers. Writing any other value to the LOCK register disables the write accesses.
- When the AON_WDT is disabled, all register values must be cleared to restore the default values.
- When the CFG0.EN bit is set to 1, the value of the TIMER_W register is automatically loaded into the TIMER_R register, and the value starts to decrease.
- The AON_WDT alarm interrupt is triggered when the value of the TIMER_R register decrements to the value of the ALARM_R register.
- When the TIMER_R count value decrements to 0, the system restarts.

4.2.2.3 Registers

4.2.2.3.1 AON_PAD_CTRL0

- **Name:** Always-on Pad Control Register
- **Description:** This register contains the AON_GPIO configurations.
- **Base Address:** 0x4000A000
- **Offset:** 0x1E4
- **Reset Value:** 0x00FF0000

Table 4-35 AON_PAD_CTRL_0 Register

Bits	Field Name	RW	Reset	Description
31:24	AON_INPUT_VAL	R	0x0	Always-on PAD signal from outside to cores
16:23	AON_IN_N	RW	0xFF	Always-on PAD input enable **Value**: • 0x0: Disable AONx input. • 0x1: Enable AONx input
15:8	AON_O_N	RW	0x0	Always-on PAD output enable **Value**: • 0x0: Disable AONx output. • 0x1: Enable AONx output.
7:0	AON_OUT_VAL	RW	0x0	Always-on PAD signal from core to outside

4.2.2.3.2 AON_PAD_CTRL1

- **Name**: Always-on Pad Control Register

- **Description**: This register contains the input and output data.
- **Base Address**: 0x4000A000
- **Offset**: 0x1E8
- **Reset Value**: 0x000000FF

Table 4-36 AON_PAD_CTRL_1 Register

Bits	Field Name	RW	Reset	Description
31:16	RSVD	R		Reserved bits
15:8	AON_R_TYPE	RW	0x00	Always-on PAD resistor type **Value**: • 0x0: pull-down • 0x1: pull-up
7:0	AON_RE_N	RW	0xFF	Always-on PAD resistor enable **Value**: • 0x0: Disable AONx resistor. • 0x1: Enable AONx resistor.

4.2.2.3.3 AON_PAD_CLK

- **Name:** Always-on PAD output clock controls register
- **Description:** This register controls inner clock output from AON_GPIO_4.
- **Base Address:** 0x4000A000
- **Offset:** 0x1F0
- **Reset Value:** 0x00000000

Table 4-37 AON_PAD_CLK Register

Bits	Field Name	RW	Reset	Description
31:5	RSVD	R		Reserved bits
4:2	CLK_OUT_SEL	RW	0x0	Clock out selection **Value:** • 0x0: RNG_OSC (RNG Osc 32 kHz Clock) • 0x1: HFXO_32K (HFXO 32 kHz derived from HFXO_32 MHz Clock) • 0x2: LFRC_32K (LFRC 32 kHz Clock) • 0x3: LFXO_32K (LF_XO 32 kHz Clock)
1	RSVD	R		Reserved bits

Bits	Field Name	RW	Reset	Description
0	CLK_OUT_EN	RW	0x0	Enable clock out via AON GPIO 4. **Value:** • 0x0: Disable selected clock out via AON GPIO 4. • 0x1: Enable selected clock out via AON GPIO 4.

4.2.2.3.4 AON_PAD_MCU_OVR

- **Name**: Always-on Pad Control Register
- **Description**: This register controls the MCU domain setting of always-on pads.
- **Base Address**: 0x4000A000
- **Offset**: 0x1F4
- **Reset Value**: 0x00000000

Table 4-38 AON_PAD_MCU_OVR Register

Bits	Field Name	RW	Reset	Description
31:17	RSVD	R		Reserved bits
23:16	AON_PAD_MCU_OVR	RW	0x0	Use the settings from MCU domain; only valid when MCU domain is ON • 0x0: Disable AONx digital setting. • 0x1: Enable AONx digital setting.
15:0	RSVD	R		Reserved bits

4.2.2.4 Electrical Specifications

The electrical parameters for the AON_GPIO_0–AON_GPIO_7 are as follows:

Table 4-39 GPIO electrical specifications

Parameter	Description	Min.	Typ.	Max.	Unit
V_{IH}	Input high voltage	VDDIO x 0.7		VDDIO	V
V_{IL}	Input low voltage	VSSIO		VDDIO x 0.3	V
$V_{OH,L}$	Output high voltage, 4 mA, VDD ≥ 1.7 V	VDD – 0.4		VDD	V
$V_{OH,M}$	Output high voltage, 4 mA, VDD ≥ 2.5 V	VDD – 0.4		VDD	V
$V_{OH,H}$	Output high voltage, 4 mA, VDD ≥ 3 V	VDD – 0.4		VDD	V
$V_{OL,L}$	Output low voltage, 4 mA, VDD ≥ 1.7 V	VSS		VSS + 0.4	V
$V_{OL,M}$	Output low voltage, 4 mA, VDD ≥ 2.5 V	VSS		VSS + 0.4	V
$V_{OL,H}$	Output low voltage, 4 mA, VDD ≥ 3 V	VSS		VSS + 0.4	V

Parameter	Description	Min.	Typ.	Max.	Unit
I_{OL}	Current at VSS+0.4 V, output set low, VDD ≥1.7 V		4		mA
I_{OH}	Current at VDD-0.4 V, output set high, VDD ≥ 1.7 V		4		mA
R_{PU}	Pull-up resistance		20		kΩ
R_{PD}	Pull-down resistance		20		kΩ
C_{PAD}	Pad capacitance		5		pF

4.2.3 Mixed Signal I/O

4.2.3.1 Introduction

The device has eight individually configurable MSIOs that can be applied by peripherals, digital input\output or analog input.

4.2.3.2 Functional Description

- MSIO default state is analog input mode.
- As analog input for sense ADC or Comparator.
- As digital MSIOs for input/output, no interrupt.
- To multiplex some different peripherals.
- For digital modules with high timing requirements, it is recommended to increase the MSIO drive current.

4.2.3.3 Registers

4.2.3.3.1 MSIO_APAD_CFG0

- **Name:** MSIO Pad Control Register
- **Description:** This register contains the input and output data.
- **Base Address:** 0x4000A1D4
- **Offset:** 0x00
- **Reset Value:** 0x00FF0000

Table 4-40 MSIO_APAD_CFG0 Register

Bits	Field Name	RW	Reset	Description
31:24	MSIO_VAL	R		MSIO read value
23:16	MSIO_IE_N	RW	0xFF	MSIO input enable **Value**: • 0x0: Disable MSIO_x input.

Bits	Field Name	RW	Reset	Description
				• 0x1: Enable MSIO_x input.
15:8	MSIO_OE_N	RW	0x0	MSIOA output enable (active low) **Value**: • 0x0: Disable MSIO_x output. • 0x1: Enable MSIO_x output.
7:0	MSIO_OUT	RW	0x0	MSIO output value (valid in output mode)

4.2.3.3.2 MSIO_APAD_CFG1

- **Name:** MSIO Pad Control Register
- **Description:** This register contains the MSIOA configurations.
- **Base Address:** 4000A1D8
- **Offset:** 0x00
- **Reset Value:** 0x00000000

Table 4-41 MSIO_APAD_CFG1 Register

Bits	Field Name	RW	Reset	Description
31:16	RSVD	R		Reserved bits
15:8	MSIO_RTYPE	RW	0x0	MSIO resistor type **Value**: • 0x0: Pull down. • 0x1: Pull up.
7:0	MSIO_AE_N	RW	0x0	MSIO input enable (active low) **Value**: • 0x0: Disable input. • 0x1: Enable input.

4.2.3.3.3 MSIO_APAD_MCU_OVR

- **Name:** MSIO Using MCU Domain Setting Control Register
- **Description:** This register controls MSIO using the settings from the MCU domain.
- **Base Address:** 0x4000A1E0
- **Offset:** 0x0
- **Reset Value:** 0x00000000

Table 4-42 MSIO_APAD_MCU_OVR Register

Bits	Field Name	RW	Reset	Description
31:8	RSVD	R	N/A	Reserved bits
7:0	MSIO_MCU_OVR	RW	0x0	Use the settings from MCU domain, only valid when MCU domain is ON **Value**: • 0x0: Use settings in MSIO_APAD_CFG as GPIO. • 0x1: Use PAD MUX settings in MCU.

4.2.3.4 Electrical Specifications

The electrical parameters for the MSIO_0–MSIO_7 are as follows:

Table 4-43 GPIO electrical specifications

Parameter	Description	Min.	Typ.	Max.	Unit
V_{IH}	Input high voltage	VDDIO x 0.7		VDDIO	V
V_{IL}	Input low voltage	VSSIO		VDDIO x 0.3	V
$V_{OH,L}$	Output high voltage, 2.5 mA, VDD ≥ 1.7 V	VDD – 0.4		VDD	V
$V_{OH,M}$	Output high voltage, 2.5 mA, VDD ≥ 2.5 V	VDD – 0.4		VDD	V
$V_{OH,H}$	Output high voltage, 2.5 mA, VDD ≥ 3 V	VDD – 0.4		VDD	V
$V_{OL,L}$	Output low voltage, 2.5 mA, VDD ≥ 1.7 V	VSS		VSS + 0.4	V
$V_{OL,M}$	Output low voltage, 2.5 mA, VDD ≥ 2.5 V	VSS		VSS + 0.4	V
$V_{OL,H}$	Output low voltage, 2.5 mA, VDD ≥ 3 V	VSS		VSS + 0.4	V
I_{OL}	Current at VSS+0.4 V, output set low, VDD ≥1.7 V		7		mA
I_{OH}	Current at VDD-0.4 V, output set high, VDD ≥ 1.7 V		7		mA
R_{PU}	Pull-up resistance	70	120	150	kΩ
RPD	Pull-down resistance		10		kΩ
C_{PAD}	Pad capacitance		10		pF

4.3 Timer Interfaces

4.3.1 Timer

4.3.1.1 Introduction

The device has two instances of Timer module that is a 32-bit general-purpose timer. Since the Timer is in the MCU Subsystem domain, Timer interrupt is not available during sleep, and the running Timer will not prevent MCU from

sleeping. In active mode, when Timer is started again after stopping, it will continue to count down from the stop time. The value of RELOAD register will not take effect until the end of the current cycle.

4.3.1.2 Main Features

- Generate an interrupt when the 32-bit down counter reaches 0.
- The clock source of the Timer follows the system clock.
- Each timer module has an independent interrupt entry.
- Timer pauses when MCU is suspended.

4.3.1.3 Functional Description

- The Timer can generate an interrupt request signal when the 32-bit down counter reaches 0. The interrupt request is held until it is cleared by writing **1** to the INTSTAT register.
- If the Timer count reaches 0 and, at the same time, the software clears a previous interrupt status, the interrupt status is set to **1**.
- When the value of the RELOAD register is configured to **0**, no count and no interrupt are generated.
- The RELOAD value only represents the count-down number of timers. If you want to configure a specific duration (for example, µs, ms, or s), you need to use the following formula:

$$\text{Expect_time}(s) = \text{RELOAD} / (\text{SystemCoreClock} - 1)$$

- If you need to start counting from the new RELOAD value immediately, you need to deinitialize the Timer first, and then initialize the Timer.

4.3.1.4 Registers

4.3.1.4.1 CTRL

- **Name**: Timer Control Register
- **Description**: The CTRL register enables the software to control the Timer.
- **Base Address**: 0x40000000 + x*0x1000
- **Offset**: 0x00
- **Reset Value**: 0x00

Note:

*x is used to identify the Timer Module. The “x” can be 0 or 1.

Table 4-44 CTRL Register

Bits	Field Name	RW	Reset	Description
31:4	RSVD	R		Reserved bits

Bits	Field Name	RW	Reset	Description
3	INT_EN	RW	0x0	Timer interrupt enable **Value**: • 0x0: Disable the Timer interrupt. • 0x1: Enable the Timer interrupt.
2:1	RSVD	R		Reserved bits
0	EN	RW	0x0	Timer enable. **Value**: • 0x0: Disable the Timer. • 0x1: Enable the Timer.

4.3.1.4.2 VALUE

- **Name**: Timer Value Register
- **Description**: The VALUE register indicates the current value of the decrementing counter.
- **Base Address**: 0x40000000 + x*0x1000
- **Offset**: 0x04
- **Reset Value**: 0x00

Table 4-45 VALUE Register

Bits	Field Name	RW	Reset	Description
31:0	CUR_VALUE	RW	0x1	Current value

4.3.1.4.3 RELOAD

- **Name**: Timer Reload Register
- **Description**: When this register is written to, the count value takes effect at the end of the current cycle.
- **Base Address**: 0x40000000 + x*0x1000
- **Offset**: 0x08
- **Reset Value**: 0x00

Table 4-46 RELOAD Register

Bits	Field Name	RW	Reset	Description
31:0	RELOAD_VALUE	RW	0x0	Sets reload value.

4.3.1.4.4 INTSTAT

- **Name**: Timer Interrupt Status Register

- **Description**: The INTSTAT register indicates the interrupt status from the counter. Writing one to the INTSTAT register clears the timer interrupt.
- **Base Address**: 0x40000000 + x*0x1000
- **Offset**: 0x0C
- **Reset Value**: 0x00000000

Table 4-47 INTSTAT Register

Bits	Field Name	RW	Reset	Description
31:1	RSVD	R		Reserved bits
0	STAT	RW	0x0	Timer interrupt. Write 1 to clear.

4.3.2 Dual Timer

4.3.2.1 Introduction

The Dual Timer module consists of two programmable down counters (32-bit or 16-bit) that can generate interrupts when they reach 0. Since the Dual-Timer is in the MCU Subsystem domain, Dual-Timer interrupt is not available during sleep, and the running Dual-Timer will not prevent the MCU from sleeping. If the RELOAD register is configured with a value, the Dual-Timer immediately counts down from the RELOAD value; if the BG_LOAD register is configured with a value, the Dual-Timer starts counting down again with the BG_LOAD value after the current counting cycle is completed. If BG_LOAD value is set before Dual-Timer is initialized, the BG_LOAD value will be invalid after the Dual-Timer is initialized with RELOAD value.

4.3.2.2 Main Features

- A 32-bit or 16-bit counter
- One of the following timer modes:
  - Periodic
  - One-shot
  - Free-running
- The clock source of the Dual Timer follows the system clock
- Dual Timer has a prescaler that can divide down the enabled timer clock rate by 1, 16, or 256.
- Dual Timer pauses when the MCU is suspended.

4.3.2.3 Functional Description

In Dual Timer, both of the counters are defined by default. For each counter, the following operational modes are available:

- Free-running mode

  The counter wraps after reaching zero, and continues to count down from the maximum value (0xFFFFFFFF). This is the default mode.

- Periodic mode

  The counter generates an interrupt at a fixed interval, reloading the original value after wrapping past zeros.

- One-shot mode

  The counter generates an interrupt once. When the counter reaches 0, it halts until you reprogram it.

  When the value of the RELOAD register is configured to **0**, no count and no interrupt are generated.

The RELOAD value only represents the count-down number of timers. If you want to configure a specific duration (for example, µs, ms, or s), you need to use the following formula:

$$Expect_time = RELOAD / (SystemCoreClock / PRE - 1)$$

where PRE stands for prescaled coefficient.

4.3.2.4 Registers

4.3.2.4.1 RELOAD

- **Name**: Dual Timer Reload Register
- **Description**: This register contains the value from which the counter is to decrease. The count is immediately restarted from the new RELOAD value.
- **Base Address**: 0x40002000
- **Offset**: 0x00 + x*0x20
- **Reset Value**: 0x00

Note:

*x is used to identify the counter of the Dual Timer Module. The “x” can be 0 or 1.

Table 4-48 RELOAD Register

Bits	Field Name	RW	Reset	Description
31:0	RELOAD_VALUE	RW	0x0	Reload value. Write to this register sets the current value.

4.3.2.4.2 VALUE

- **Name**: Dual Timer Current Value Register
- **Description**: This register provides the current count value of the decrementing counter.
- **Base Address**: 0x40002000
- **Offset**: 0x04 + x*0x20

- **Reset Value**: 0xFFFFFFFF

Table 4-49 VALUE Register

Bits	Field Name	RW	Reset	Description
31:0	CUR_VALUE	RO	0xFFFFFFFF	Current count value

4.3.2.4.3 CTRL

- **Name**: Dual Timer Control Register
- **Description**: This register enables the software to control the Dual Timer.
- **Base Address**: 0x40002000
- **Offset**: 0x08 + x*0x20
- **Reset Value**: 0x20

Table 4-50 CTRL Register

Bits	Field Name	RW	Reset	Description
31:8	RSVD	R		Reserved bits
7	EN	RW	0x0	Timer enable **Value**: • 0x0: Timer disabled; default • 0x1: Timer enabled
6	MODE	RW	0x0	Timer mode **Value**: • 0x0: Timer is in free-running mode; default • 0x1: Timer is in periodic mode
5	INT_EN	RW	0x1	Interrupt enable **Value**: • 0x0: Timer interrupt disabled • 0x1: Timer interrupt enabled; default
4	RSVD	R		Reserved bits
3:2	PRE	RW	0x0	Prescale bits **Value**: • 0x00: 0 stage of prescale, clock is divided by 1; default • 0x01: 4 stages of prescale, clock is divided by 16 • 0x10: 8 stages of prescale, clock is divided by 256 • 0x11: Undefined. Do not use this value.

Bits	Field Name	RW	Reset	Description
1	SIZE	RW	0x0	Select 16-bit or 32-bit counter operation: **Value**: • 0x0: 16-bit counter; default • 0x1: 32-bit counter
0	COUNT_MODE	RW	0x0	Select wrap mode or one-shot mode: **Value**: • 0x0: Wrap mode; default • 0x1: One-shot mode

4.3.2.4.4 INTCLR

- **Name**: Dual Timer Interrupt Clear Register
- **Description**: Any write to the INTCLR register clears the interrupt output from the counter.
- **Base Address**: 0x40002000
- **Offset**: 0x0C + x*0x20
- **Reset Value**: 0x00

Table 4-51 INTCLR Register

Bits	Field Name	RW	Reset	Description
31:0	STAT	WO	0x0	Clear the Dual-Timer interrupt event.

4.3.2.4.5 RAW_INTSTAT

- **Name**: Dual Timer Raw Interrupt Status Register
- **Description**: The RAW_INTSTAT register indicates the raw interrupt status from the counter.
- **Base Address**: 0x40002000
- **Offset**: 0x10 + x*0x20
- **Reset Value**: 0x00

Table 4-52 RAW_INTSTAT Register

Bits	Field Name	RW	Reset	Description
31:1	RSVD	R		Reserved bits
0	STAT	RO	0x0	Raw interrupt status from the counter

4.3.2.4.6 INTSTAT

- **Name**: Dual-Timer interrupt enable status register

- **Description**: The INTSTAT register indicates the masked interrupt status from the counter.
- **Base Address**: 0x40002000
- **Offset**: 0x14 + x*0x20
- **Reset Value**: 0x00

Table 4-53 INTSTAT Register

Bits	Field Name	RW	Reset	Description
31:1	RSVD	R		Reserved bits
0	STAT	RO	0x0	Enable interrupt status from the counter.

4.3.2.4.7 BG_LOAD

- **Name**: Dual-Timer background reload register
- **Description**: When the BG_LOAD value is set, the counter will count down again from the BG_LOAD value after completing the current counting cycle.
- **Base Address**: 0x40002000
- **Offset**: 0x18 + x*0x20
- **Reset Value**: 0x00

Table 4-54 BG_LOAD Register

Bits	Field Name	RW	Reset	Description
31:0	BG_LOAD_VALUE	RW	0x0	The BG_LOAD does not cause the counter to immediately restart from the new value.

4.3.3 PWM

4.3.3.1 Introduction

The device has two instances of Pulse Width Modulation (PWM) module. The PWM generates successive pulses with variable duty cycles that can be converted to analog levels by external discrete components, such as controlling LED brightness via a PWM signal.

4.3.3.2 Main Features

- Configurable output frequency and dynamic frequency configuration
- Three independent PWM channel outputs: pwma, pwmb, and pwmc
- Two operation modes: flicker mode and breath mode
- Two alignment modes: edge alignment mode and center alignment mode

- Configurable duty cycles in flicker mode
- Three configuration modes: synchronize-all, synchronize-separate, and asynchronous
- Polarity switching to support positive-drive and negative-drive mode
- Optional stop I/O voltage level in active/idle mode: low level or high level

4.3.3.3 Block Diagram

The PWM block diagram is illustrated in Figure 4-2.

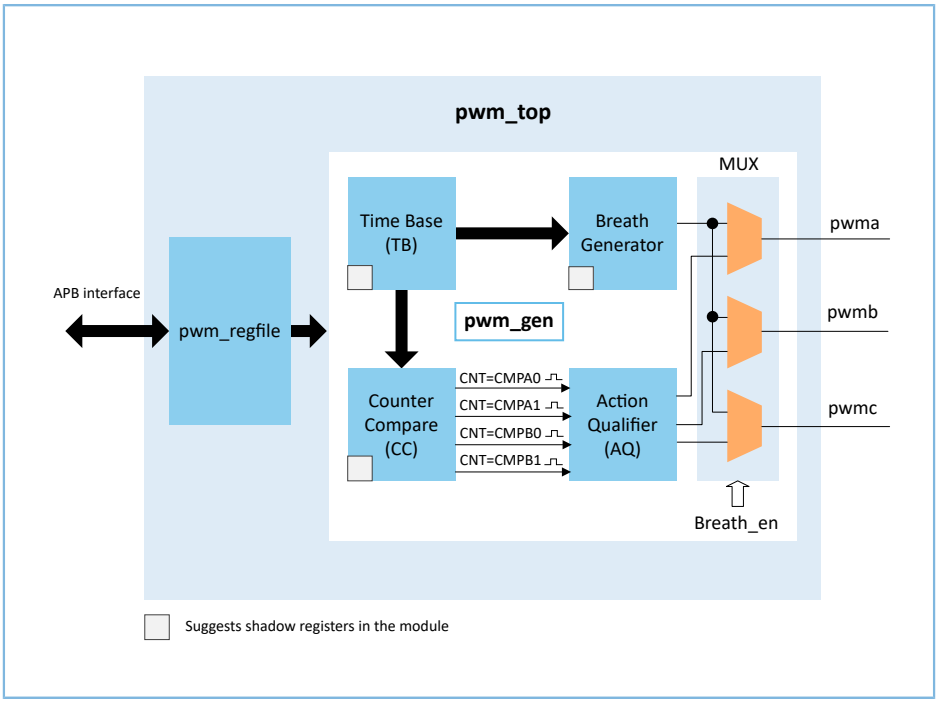

Figure 4-2 PWM block diagram

4.3.3.4 Functional Description

The PWM module has two instances which are PWM0 and PWM1. There are two operation modes: flicker mode and breath mode. Each PWM instance can provide three independent channel outputs: pwma, pwmb and pwmc. Each channel shares the same frequency but can be configured with different duty cycles.

Bothpositive-drive mode and negative-drive mode are supported and can be configured by the DPENx field in the MODE register.

- In positive-drive mode, a valid output is configured by setting the PWM output logic level to 1, whereas an invalid output is set when the PWM output logic level is 0.
- In negative-drive mode, a valid output is configured by setting the PWM output logic level to 0, whereas an invalid output is set when PWM output logic level is 1.

4.3.3.4.1 Flicker Mode

In flicker mode, the PWM module outputs successive pulses with a certain frequency and duty as configured. PRD register is used to configure the frequency in the flicker mode. The duty of PWM can be configured in the range of 0% to 100% through CMPx0, CMPx1 and AQCTRL register. The three outputs can be configured independently, and they share one output frequency.

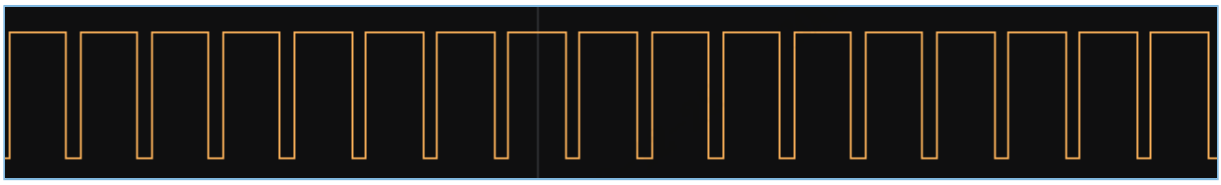

Figure 4-3 Flicker mode waveform

In flicker mode, if the stop level is configured (the default value of FLICKER_PAUSE_LEVEL_x is 0 when not set), the I/O level of the corresponding channel will be pulled to the set value when PWM stops.

4.3.3.4.2 Breath Mode

In breath mode, the duty of PWM output periodically changes from 0% to 100%, and then 100% to 0% uniformly. The breathing cycle can be set by the BRPRD register, which represents the number of PWM cycles required for the duty cycle 0%→100% and then 100%→0%.

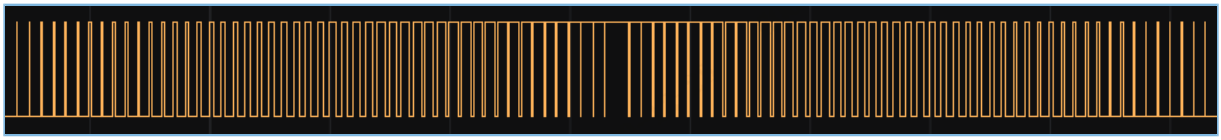

Figure 4-4 Breath mode waveform

A configurable duration of breath hold state (can be set by the HOLD rgister) is supported. The breath hold state is set between two adjacent breath processes, that is duty change (0%→100%→0%), breath hold, and duty change (0%→100%→0%). In the hold state, the PWM output is driven by inactive state.

In breath mode, the I/O levels of all channels are the same and set by the BREATH_PAUSE_LEVEL bit in the MODE register when PWM stops. When MCU enters sleep state, the stop level remains high in both flicker mode and breath mode.

4.3.3.4.3 Alignment

In flicker mode, both edge alignment and center alignment are supported through software approach.

- Choose edge alignment mode, as shown in Figure 4-5. All open channels are aligned at the beginning of each duty cycle.

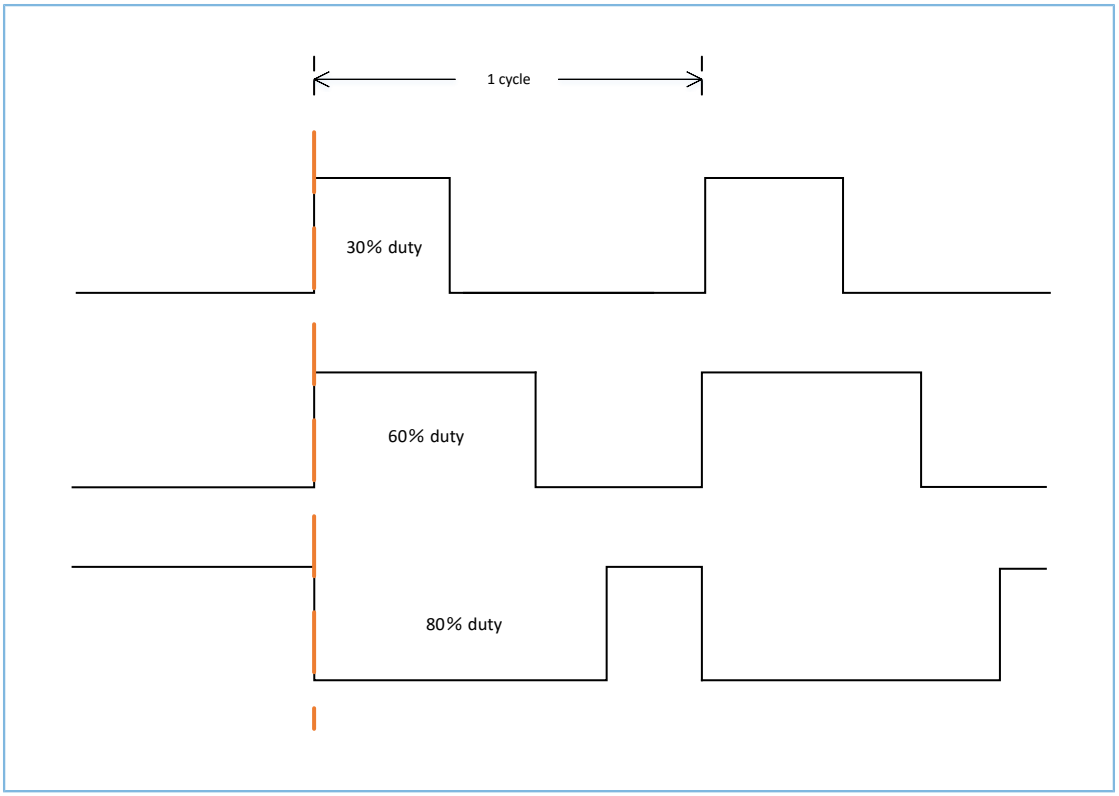

Figure 4-5 PWM edge alignment

- Choose center alignment mode, as shown in Figure 4-6. All open channels are aligned in the middle of each duty cycle.

Copyright © 2024 Shenzhen Goodix Technology Co., Ltd.

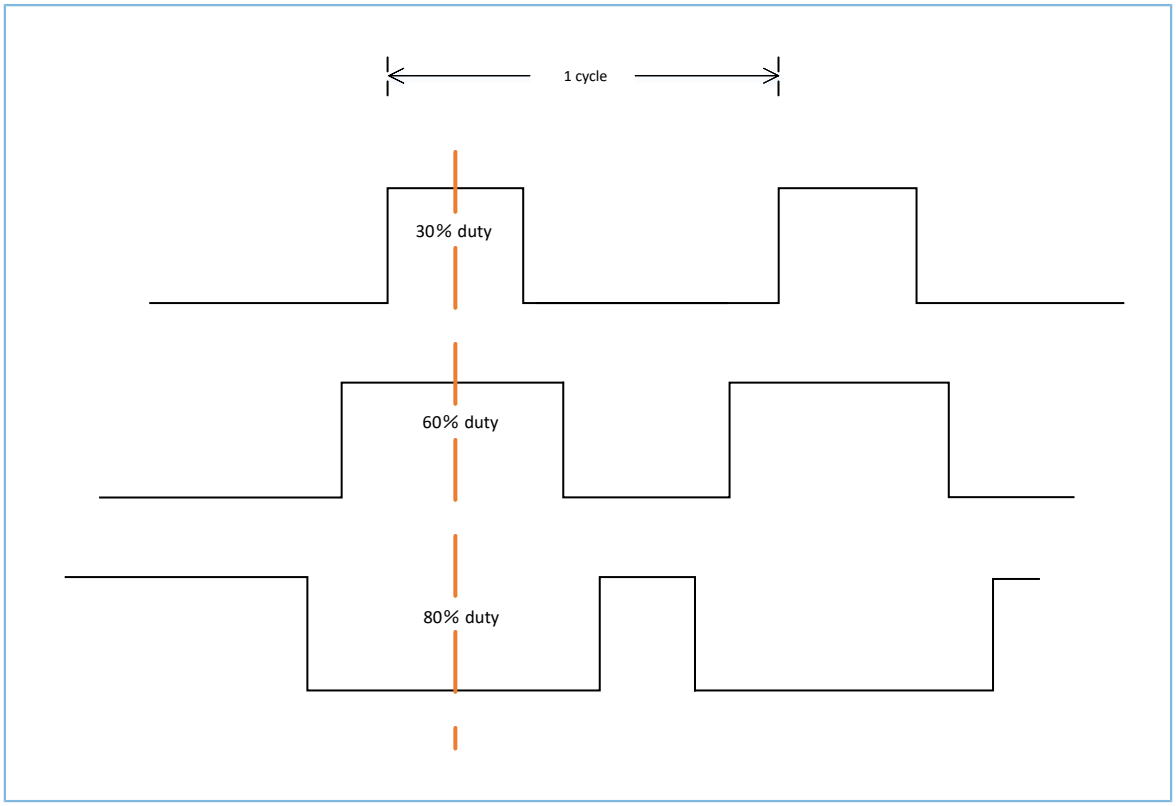

Figure 4-6 PWM center alignment

4.3.3.4.4 Active Update

PWM module is capable of configuring output frequency dynamically. There are several shadow registers in the PWM module. The write operation will first change the shadow registers, and then will update the value of active registers from the shadow registers under certain conditions.

The configured data can be updated into active registers in three modes: synchronize-all mode (enabled by UPDATE.SA), synchronize-separate mode (enabled by UPDATE.SSCMPxx) and asynchronous update mode (enabled by UPDATE.SA and UPDATE.SSCMPxx).

Active registers update conditions:

1. In synchronize-all update mode (UPDATE.SA = 0x1), all effective registers of flicker mode or breath mode will be updated simultaneously. In flicker mode (MODE.BREATHEN = 0), after writing to PRD, the CMPxx and AQCTRL registers will update from shadow registers simultaneously when the Time Base counter counts to a value of PRD minus 1.

   In breath mode (MODE.BREATHEN = 1), after writing to PRD, the BRPRD and HOLD registers will update from shadow registers simultaneously when the current breath process ends (the moment PWM duty reduces to 0%). When using the app_pwm_update_freq interface, the PWM frequency will be updated immediately.

2. In synchronize-separate update mode (UPDATE.SA = 0x0), the PRD, CMPxx, and AQCTRL registers will update when the Time Base counter counts to value of the PRD register minus 1, and if the corresponding UPDATE.SSCMPxx bit is set. The BRPRD and HOLD registers will update when the current breath process ends and the corresponding UPDATE.SSCMPxx bit is set.

3. In asynchronous update mode (UPDATE.SA = 0x0 and UPDATE.SSCMPxx = 0x0), all registers will update instantly.

   If the configurations need to be changed under synchronize-all update mode, CMPXX and AQCTRL in flicker mode, or HOLD in breath mode should be written before modifying the values of PRD and BRPRD.

4.3.3.4.5 Duty Cycle Calculation

The PWM duty cycle can be configured as an integer from 0 to 100, depending on the following calculation formulas of the duty cycle accuracy "n".

- In edge-aligned mode:

$$n = f_{PWM} \div f_{SYSCLK} \times 100$$

- In center-aligned mode:

$$n = f_{PWM} \div f_{SYSCLK} \times 200$$

For example, when the clock frequency is 64 MHz, and the duty cycle is in the range of 0 to 100, the maximum PWM frequency can only be set to 640 kHz in the edge-aligned mode and 320 kHz in the center-aligned mode.

4.3.3.4.6 Recommended Setting in Breath Mode

In breath mode, in order to satisfy that the duty cycle of the waveform changes from 0 to 100 and then from 100 to 0, it is necessary to set the values of PRD and BRPRD to an integer that is multiple(s) of 128. The freq, bperiod and hperiod in the structure pwm_init_t need to meet the following formula during initialization:

$$freq \times bperiod = 1000 \times k(k \geq 2 \&\& k \in \text{int}\,eger)$$

$$\frac{SYS_{CLK}}{1000} \times hperiod \leq 16777215$$

where, k represents the number of PWM cycles required for the duty cycle from 0% to 100%; SYS_{CLK} represents the system clock frequency; bperiod represents the breath period; hperiod represents the hold period.

4.3.3.5 Registers

4.3.3.5.1 MODE

- **Name**: Mode Register
- **Description**: This register sets the PWM operation mode.
- **Base Address**: 0x4000CB00 + x*0x100
- **Offset**: 0x0
- **Reset Value**: 0x00000000

Note:

*x is used to identify the PWM Module instance. The “x” can be 0 or 1.

Table 4-55 Mode Register

Bits	Field Name	RW	Reset	Description
31:10	RSVD	R		Reserved bits
9	BREATH_PAUSE_LEVEL	RW	0x0	PWMA/B/C output IO level when suspending in Breath Mode
8	FLIKCER_PAUSE_LEVEL_C	RW	0x0	PWMC output IO level when suspending in Flicker Mode
7	FLIKCER_PAUSE_LEVEL_B	RW	0x0	PWMB output IO level when suspending in Flicker Mode
6	FLIKCER_PAUSE_LEVEL_A	RW	0x0	PWMA output IO level when suspending in Flicker Mode
5	DPENC	RW	0x0	Enable PWMC positive-drive mode. **Value**: • 0x0: negative-drive mode • 0x1: positive-drive mode
4	DPENB	RW	0x0	Enable PWMB positive-drive mode. **Value**: • 0x0: negative-drive mode • 0x1: positive-drive mode
3	DPENA	RW	0x0	Enable PWMA positive-drive mode. **Value**: • 0x0: negative-drive mode • 0x1: positive-drive mode
2	BREATHEN	RW	0x0	Enable Breath mode. **Value**: • 0x0: flicker mode • 0x1: breath mode
1	PAUSE	RW	0x0	PWM pause signal. **Value**: • 0x0: Ongoing • 0x1: Pause
0	EN	RW	0x0	Enable PWM. **Value**: • 0x0: Disable • 0x1: Enable

4.3.3.5.2 UPDATE

- **Name**: Update Register

- **Description**: This register is used to set the synchronous update mode.
- **Base Address**: 0x4000CB00 + x*0x100
- **Offset**: 0x4
- **Reset Value**: 0x00000000

Table 4-56 Update Register

Bits	Field Name	RW	Reset	Description
31:18	RSVD	R		Reserved bits
17	SSAQCTRL	RW	0x0	Enable separate update of AQCTRL. **Value**: • 0x0: Disable • 0x1: Enable
16	SSHOLD	RW	0x0	Enable separate update of HOLD. **Value**: • 0x0: Disable • 0x1: Enable
15	SSBRPRD	RW	0x0	Enable separate update of BRPRD. **Value**: • 0x0: Disable • 0x1: Enable
14	SSCMPC1	RW	0x0	Enable separate update of CMPC1. **Value**: • 0x0: Disable • 0x1: Enable
13	SSCMPC0	RW	0x0	Enable separate update of CMPC0. **Value**: • 0x0: Disable • 0x1: Enable
12	SSCMPB1	RW	0x0	Enable separate update of CMPB1. **Value**: • 0x0: Disable • 0x1: Enable
11	SSCMPB0	RW	0x0	Enable separate update of CMPB0. **Value**: • 0x0: Disable

Bits	Field Name	RW	Reset	Description
				• 0x1: Enable
10	SSCMPA1	RW	0x0	Enable separate update of CMPA1. **Value**: • 0x0: Disable • 0x1: Enable
9	SSCMPA0	RW	0x0	Enable separate update of CMPA0. **Value**: • 0x0: Disable • 0x1: Enable
8	SSPRD	RW	0x0	Enable separate update of PRD. **Value**: • 0x0: Disable • 0x1: Enable
7:2	RSVD	R		Reserved bits
1	SA	RW	0x0	Enable synchronous update. **Value**: • 0x0: Disable • 0x1: Enable
0	SAG	R	0x0	Synchronous update ongoing **Value**: • 0x0: Not ongoing • 0x1: Ongoing

4.3.3.5.3 PRD

- **Name**: Period Register
- **Description**: This register is used to set the PWM cycle.
- **Base Address**: 0x4000CB00 + x*0x100
- **Offset**: 0x8
- **Reset Value**: 0x00000064

Table 4-57 Period Register

Bits	Field Name	RW	Reset	Description
31:0	PRD	RW	0x64	The period of PWM output, PRD=f_{CLK}/f_{PWM}

4.3.3.5.4 CMPA0

- **Name**: Compare A0 Register
- **Description**: This register controls the PWM duty of Compare register 0 of channel A.
- **Base Address**: 0x4000CB00 + x*0x100
- **Offset**: 0xC
- **Reset Value**: 0x00000000

Table 4-58 Compare A0 Register

Bits	Field Name	RW	Reset	Description
31:0	CMPA0	RW	0x0	PWMA duty control register0

4.3.3.5.5 CMPA1

- **Name**: Compare A1 Register
- **Description**: This register controls the PWM duty of Compare register 1 of channel A.
- **Base Address**: 0x4000CB00 + x*0x100
- **Offset**: 0x10
- **Reset Value**: 0x00000000

Table 4-59 Compare A1 Register

Bits	Field Name	RW	Reset	Description
31:0	CMPA1	RW	0x0	PWMA duty control register1

4.3.3.5.6 CMPB0

- **Name**: Compare B0 Register
- **Description**: This register controls the PWM duty of Compare register 0 of channel B.
- **Base Address**: 0x4000CB00 + x*0x100
- **Offset**: 0x14
- **Reset Value**: 0x00000000

Table 4-60 Compare B0 Register

Bits	Field Name	RW	Reset	Description
31:0	CMPB0	RW	0x0	PWMB duty control register0

4.3.3.5.7 CMPB1

- **Name**: Compare B1 Register
- **Description**: This register controls the PWM duty of Compare register 1 of channel B.
- **Base Address**: 0x4000CB00 + x*0x100
- **Offset**: 0x18
- **Reset Value**: 0x00000000

Table 4-61 Compare B1 Register

Bits	Field Name	RW	Reset	Description
31:0	CMPB1	RW	0x0	PWMB duty control register1

4.3.3.5.8 CMPC0

- **Name**: Compare C0 Register
- **Description**: This register controls the PWM duty of Compare register 0 of channel C.
- **Base Address**: 0x4000CB00 + x*0x100
- **Offset**: 0x1C
- **Reset Value**: 0x00000000

Table 4-62 Compare C0 Register

Bits	Field Name	RW	Reset	Description
31:0	CMPC0	RW	0x0	PWMC duty control register0

4.3.3.5.9 CMPC1

- **Name**: Compare C1 Register
- **Description**: This register controls the PWM duty of Compare register 1 of channel C.
- **Base Address**: 0x4000CB00 + x*0x100
- **Offset**: 0x20
- **Reset Value**: 0x00000000

Table 4-63 Compare C1 Register

Bits	Field Name	RW	Reset	Description
31:0	CMPC1	RW	0x0	PWMC duty control register1

4.3.3.5.10 AQCTRL

- **Name**: Action Qualifier Control Register

- **Description**: This register is used to set actions when counter reaches the compare value.
- **Base Address**: 0x4000CB00 + x*0x100
- **Offset**: 0x24
- **Reset Value**: 0x00000000

Table 4-64 Action Qualifier Control Register

Bits	Field Name	RW	Reset	Description
31:12	RSVD	R		Reserved bits
11:10	C1	RW	0x0	Action of event CNT_CMPC1 control register **Value**: • 0x0: Do nothing • 0x1: Clear • 0x2: Set • 0x3: Toggle
9:8	C0	RW	0x0	Action of event CNT_CMPC0 control register **Value**: • 0x0: Do nothing • 0x1: Clear • 0x2: Set • 0x3: Toggle
7:6	B1	RW	0x0	Action of event CNT_CMPB1 control register **Value**: • 0x0: Do nothing • 0x1: Clear • 0x2: Set • 0x3: Toggle
5:4	B0	RW	0x0	Action of event CNT_CMPB0 control register **Value**: • 0x0: Do nothing • 0x1: Clear • 0x2: Set • 0x3: Toggle
3:2	A1	RW	0x0	Action of event CNT_CMPA1 control register **Value**: • 0x0: Do nothing

Bits	Field Name	RW	Reset	Description
				• 0x1: Clear • 0x2: Set • 0x3: Toggle
1:0	A0	RW	0x0	Action of event CNT_CMPA0 control register **Value**: • 0x0: Do nothing • 0x1: Clear • 0x2: Set • 0x3: Toggle

4.3.3.5.11 BRPRD

- **Name**: Breath Period Register
- **Description**: This register configures the period of breath mode.
- **Base Address**: 0x4000CB00 + x*0x100
- **Offset**: 0x28
- **Reset Value**: 0x00000000

Table 4-65 Breath Period Register

Bits	Field Name	RW	Reset	Description
31:0	BRPRD	RW	0x0	Register for breath period that is the required time (number of clock) that the duty changes from 0% to 100% in breath mode.

4.3.3.5.12 HOLD

- **Name**: Hold Register
- **Description**: This register configures the hold period in breath mode.
- **Base Address**: 0x4000CB00 + x*0x100
- **Offset**: 0x2C
- **Reset Value**: 0x00000000

Table 4-66 Hold Register

Bits	Field Name	RW	Reset	Description
31:24	RSVD	R		Reserved bits

Bits	Field Name	RW	Reset	Description
23:0	HOLD	RW	0x0	Breath hold control register. The value should be the required number of clock in breath hold state.

4.3.4 Always-on Watchdog (AON_WDT)

4.3.4.1 Introduction

The Always-on Watchdog(AON_WDT) is a 32-bit countdown hardware timer. When the AON_WDT counter counts to 0, a system reset signal is sent, and when the counter value decreases to the set warning value, an interrupt signal is generated. Because the AON_WDT timer is in the AON domain, it can keep running when the MCU is off.

4.3.4.2 Main Features

- The CLK register is a read/write register when the AON_WDT is disabled and is read only when the AON_WDT is enabled. Writing to the CLK register when the AON_WDT is enabled has no effect.
- 32-bit down counter running with a low-power clock
- Generates an interrupt to warn the MCU when the 32-bit down counter reaches alarm values
- 16-bit alarm register, supporting reading and writing
- Flexible clock source selections from LFRC_32K or LFXO_32K
- An independent interrupt entry
- Supports write access protection

4.3.4.3 Functional Description

- The LOCK register enables or disables write accesses to all other registers. This is to prevent malicious software from disabling the watchdog functionality. Writing a value of 0x15CC5A51 (LOCK.LOCK_SET[31:1]) enables write accesses to all other registers. Writing any other value to the LOCK register disables the write accesses.
- When the AON_WDT is disabled, all register values must be cleared to restore the default values.
- When the CFG0.EN bit is set to 1, the value of the TIMER_W register is automatically loaded into the TIMER_R register, and the value starts to decrease.
- The AON_WDT alarm interrupt is triggered when the value of the TIMER_R register decrements to the value of the ALARM_R register.
- When the TIMER_R count value decrements to 0, the system restarts.

4.3.4.3.1 Block Diagram

The following figure shows the functional blocks of the Always-on watchdog module.

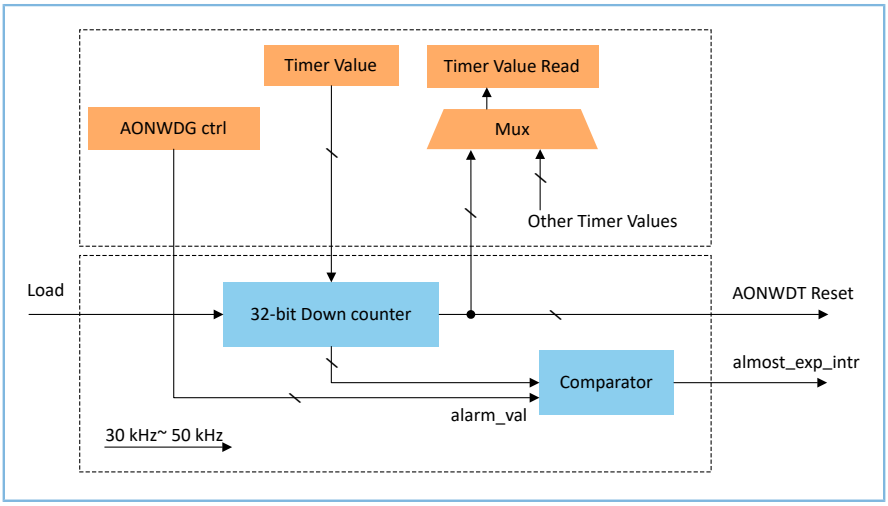

Figure 4-7 Always-on Watchdog Block Diagram

The watchdog starts running when AON_WDT_EN = 1 is synchronized to AON_WDT clock domain, which takes 1 – 3 RNG (Random Number Generators) clock cycles (20 – 100 µs).

If no reload operation is performed, the down counter default value is **0xFFFFFFFF**. Otherwise, the counter will be reloaded with the value of timer_value.

4.3.4.3.2 Clock

The AON_WDT works in RNG clock (30 kHz – 50 kHz), which is much slower than MCU system clock (16 MHz – 64 MHz). Therefore, synchronization time is needed to propagate the MCU operations. For example, in reload operation, MCU needs to check whether the **VAL_RD** value has changed as expected.

4.3.4.3.3 Enable/Disable and Timer Reload

- To enable the block, set AON_WDT_EN = 1 and wait until **AON_WDT_RUNNING** being high.
- To disable the block, set AON_WDT_EN = 0 and wait until **AON_WDT_RUNNING** being low.
- To reload down counter, program the desired value to timer_value, then set write-only register AON_WDT_RELOAD = 1. MCU can check read-only register **VAL_RD** to confirm.

4.3.4.3.4 Alarm

The AON_WDT provides an almost-expired alarm to warn MCU before system reset. By setting **ALARM_VAL= N(non-zero)**, the AON_WDT generates an interrupt N RNG clocks before the system reset.

4.3.4.4 Registers

4.3.4.4.1 CFG_0

- **Name:** CFG_0 Register
- **Description:** This register sets AON_WDT configurations.

- **Base Address:** 0x4000A700
- **Offset:** 0x0
- **Reset Value:** 0x00000000

Table 4-67 CFG_0 Register

Bits	Field Name	RW	Reset	Description
31:25	RSVD	R	0x0	Reserved bits
24	CFG	W	0x0	Writing 1 to this register triggers the watchdog timer configuration.
23:4	RSVD	R	0x0	Reserved bits
3	ALARM_SET	RW	0x0	Sets the timer alarm value.
2	ALARM_EN	RW	0x0	Enables AON WDT alarm.
1	RELOAD	W	0x0	Enables AON WDT timer reload (feed).
0	EN	RW	0x0	Enables AON WDT.

4.3.4.4.2 LOCK

- **Name:** LOCK Register
- **Description:** This register can prevent other registers of AON_WDT from being overwritten
- **Base Address:** 0x4000A700
- **Offset:** 0x4
- **Reset Value:** 0x00000000

Table 4-68 LOCK Register

Bits	Field Name	RW	Reset	Description
31:1	LOCK_VALUE	W	–	Enable write access to all other registers by writing 0x15CC5A51. Disable write access by writing any other value to the LOCK register.
0	STATUS	R	0x0	• 0: Unlocked. Write to other watchdog registers is enabled. • 1: Locked. Write to other watchdog registers is disabled.

4.3.4.4.3 STAT

- **Name:** STAT Register
- **Description:** This register can be used to get status of AON_WDT.
- **Base Address:** 0x4000A700
- **Offset:** 0x38

- **Reset Value:** 0x00000000

Table 4-69 STAT Register

Bits	Field Name	RW	Reset	Description
31:2	RSVD	R	0x0	Reserved bits
1	BUSY	R	0x0	Watchdog timer is busy. The application layer can perform other operations until this bit is set is 0.
0	STAT	R	0x0	AON WDT is running.

4.3.4.4.4 CLK

- **Name:** CLK Register
- **Description:** This register is used to select an AON_WDT clock source.
- **Base Address:** 0x4000A700
- **Offset:** 0x3C
- **Reset Value:** 0x00000000

Table 4-70 CLK Register

Bits	Field Name	RW	Reset	Description
31:2	RSVD	R	0x0	Reserved bits
1:0	CLK_SEL	RW	0x0	• 0x02: LFRC_32K (LFRC 32 kHz Clock) • 0x03: LFXO_32K (LF_XO 32 kHz Clock)

4.3.4.4.5 TIMER_WR

- **Name:** TIMER_WR Register
- **Description:** This register is used to configure the initial AON_WDT value.
- **Base Address:** 0x4000A700
- **Offset:** 0x40
- **Reset Value:** 0x00000000

Table 4-71 TIMER_WR Register

Bits	Field Name	RW	Reset	Description
31:0	TIMER_SET	RW	0x0	**Write**: Write the initial AON WDT value to the register. **Read**: • Read the initial AON WDT value that users have configured.

Bits	Field Name	RW	Reset	Description
				• Read TIMER_RD to get the current internal value.

4.3.4.4.6 ALARM_WR

- **Name:** ALARM_WR Register
- **Description:** This register is used to configure the AON_WDT alarm value.
- **Base Address:** 0x4000A700
- **Offset:** 0x44
- **Reset Value:** 0x64

Table 4-72 ALARM_WR Register

Bits	Field Name	RW	Reset	Description
31:16	RSVD	R	0x0	Reserved bits
15:0	ALARM_SET	RW	0x64	**Write**: Write an AON WDT alarm value to the register. **Read**: • Read the AON WDT alarm value that users have configured. • Read ALARM_RD to get the current internal alarm value of AON WDT.

4.3.4.4.7 TIMER_RD

- **Name:** TIMER_RD Register
- **Description:** This register is used to get the current value of AON_WDT.
- **Base Address:** 0x4000A700
- **Offset:** 0x4C
- **Reset Value:** 0x00000000

Table 4-73 TIMER_RD Register

Bits	Field Name	RW	Reset	Description
31:0	TIMER_RD	R	0x0	Current internal value of AON WDT

4.3.4.4.8 ALARM_RD

- **Name:** ALARM_RD Register
- **Description:** This register reads the current alarm value of AON_WDT.

- **Base Address:** 0x4000A700
- **Offset:** 0x50
- **Reset Value:** 0x00000000

Table 4-74 ALARM_RD Register

Bits	Field Name	RW	Reset	Description
31:16	RSVD	R	0x0	Reserved bits
15:0	ALARM_RD	R	0x0	Current internal alarm value of the AON WDT

4.3.5 Real-Time Counter (RTC/Calendar)

The Real-time Counter (RTC) modules provide a generic, low-power timer running on a low-frequency clock source. Since the RTC modules are in AON domain, the timer can keep running when the MCU is off. The modules can also wake up the system from Deep Sleep.

There are two identical RTC modules: one for calendar and the other for RTOS.

The RTC tick event interrupt module enables low power "tick-less" RTOS implementation as it optionally provides a regular interrupt source for an RTOS without the need to use the ARM® SysTick feature. Using the RTC tick event rather than the SysTick allows the CPU to be powered down while still keeping RTOS scheduling active.

4.3.5.1 Main Features

- One ordinary compare interrupt
- One tick event, supporting one-time counting and auto loading mode
- 32-bit up-counter register, supporting reading and writing
- 32-bit compare register, supporting reading and writing
- 32-bit down-tick register, supporting reading and writing
- One clock divider, supporting NO_DIV, DIV_2, DIV_4, DIV_8, DIV_16, DIV_32, DIV_64, and DIV_128
- Flexible clock source selections from LFRC_32K or LFXO_32K
- An independent interrupt entry

4.3.5.2 Functional Description

- The software has to enable RTC PMU LPD (RTC_EN) before using the RTC.
- The CLK_DIV/CLK_SEL register is a read/write register when the RTC is disabled and is read only when the RTC is enabled. Writing to the CLK_DIV/CLK_SEL register when the RTC is started has no effect.
- The OVERFLOW, COMPARE, and tick event interrupts are enabled by default.

- When RTC is enabled, the TIMER_RD register starts to count. When users set the values of the TIMER_SET register and TIME_SET bit to 1, the calendar module synchronizes the count value to the TIMER_RD register and starts timing.
- When RTC is disabled, all register values must be cleared to the default values.
- When the value of the TIMER_RD register is equal to that of the CMP_RD register, the ordinary compare interrupt is triggered.
- RTC tick event interrupt register is a value-decreasing register. When the auto reloading mode is set (TICK_TYPE), the value of the counter (TICK_SET) is automatically reloaded (TICK_RD) when the interrupt is triggered.
- When the count value of the TIMER_RD register overflows, the overflow interrupt is triggered, and the TIMER_RD register continues counting from 0.

4.3.5.3 Registers

4.3.5.3.1 CTRL

- **Name:** CTRL Register
- **Description:** This register sets RTC configurations.
- **Base Address:** 0x4000A600 + x*0x80
- **Offset:** 0x00
- **Reset Value:** 0x00000000

Table 4-75 CTRL Register

Bits	Field Name	RW	Reset	Description
31:25	RSVD	R	0x0	Reserved bits
24	CFG	RW	0x0	Writing 1 to this register triggers the calendar timer configuration.
23:11	RSVD	R	0x	Reserved bits
10	TICK_TYPE	RW	0x0	Sets the RTC tick event mode. • 0x0: One-time • 0x1: Auto reloading
9	TICK_SET	RW	0x0	Sets the RTC tick event timer value.
8	TICK_EN	RW	0x0	Enables RTC tick event.
7:5	RSVD	R	0x0	Reserved bits
4	WRAP_CLR	RW	0x0	Clears wrap counter.
3	CMP_SET	RW	0x0	Sets the RTC COMPARE timer value.
2	COMPARE_EN	RW	0x0	Enables RTC COMPARE timer.
1	TIME_SET	RW	0x0	Sets the RTC initial timer value.

Bits	Field Name	RW	Reset	Description
0	EN	RW	0x0	Enables RTC timer.

4.3.5.3.2 CLK_DIV

- **Name:** CLK_DIV Register
- **Description:** This register is used to set the frequency division coefficient
- **Base Address:** 0x4000A600 + x*0x80
- **Offset:** 0x4
- **Reset Value:** 0x00000000

Table 4-76 CLK_DIV Register

Bits	Field Name	RW	Reset	Description
31:3	RSVD	R	0x0	Reserved bits
2:0	CLK_DIV	RW	0x0	• 0x0: NO_DIV. Not divided. Counting at SLP_CLK • 0x1: DIV_2. Counting at 1/2 of SLP_CLK • 0x2: DIV_4. Counting at 1/4 of SLP_CLK • 0x3: DIV_8. Counting at 1/8 of SLP_CLK • 0x4: DIV_16. Counting at 1/16 of SLP_CLK • 0x5: DIV_32. Counting at 1/32 of SLP_CLK • 0x6: DIV_64. Counting at 1/64 of SLP_CLK • 0x7: DIV_128. Counting at 1/128 of SLP_CLK

4.3.5.3.3 IRQ_EN

- **Name:** IRQ_EN Register
- **Description:** This register is used to enable RTC interrupts.
- **Base Address:** 0x4000A600 + x*0x80
- **Offset:** 0x14
- **Reset Value:** 0x00000007

Table 4-77 IRQ_EN Register

Bits	Field Name	RW	Reset	Description
31:3	RSVD	R	0x0	Reserved bits
2	TICK_EVT	RW	0x1	RTC tick event interrupt. Write 1 to enable the interrupt.
1	OVERFLOW	RW	0x1	RTC overflow interrupt. Write 1 to enable the interrupt.

Bits	Field Name	RW	Reset	Description
0	COMPARE	RW	0x1	RTC COMPARE interrupt. Write 1 to enable the interrupt.

4.3.5.3.4 IRQ_STAT

- **Name:** IRQ_STAT Register
- **Description:** This register is used to get interrupt status of the RTC.
- **Base Address:** 0x4000A600 + x*0x80
- **Offset:** 0x18
- **Reset Value:** 0x00000000

Table 4-78 IRQ_STAT Register

Bits	Field Name	RW	Reset	Description
31:3	RSVD	R	0x0	Reserved bits
2	TICK_EVT	RW	0x0	RTC tick event interrupt. Write 1 to clear the interrupt.
1	OVERFLOW	RW	0x0	RTC overflow interrupt. Write 1 to clear the interrupt.
0	COMPARE	RW	0x0	RTC COMPARE interrupt. Write 1 to clear the interrupt.

4.3.5.3.5 STAT

- **Name:** STAT Register
- **Description:** This register is used to get the RTC status.
- **Base Address:** 0x4000A600 + x*0x80
- **Offset:** 0x38
- **Reset Value:** 0x00000000

Table 4-79 STAT Register

Bits	Field Name	RW	Reset	Description
31:12	RSVD	R	0x0	Reserved bits
11:8	WRAP_CNT	R	0x0	Calendar timer wrap-around counter
7:5	RSVD	R		Reserved bits
4	BUSY	R	0x0	Calendar timer is busy. The application layer can perform other operations until this bit is set is 0.
3:1	RSVD	R		Reserved bits
0	STAT	R	0x0	RTC timer is running.

4.3.5.3.6 CLK_SEL

- **Name:** CLK_SEL Register
- **Description:** This register is used to select a clock source.
- **Base Address:** 0x4000A600 + x*0x80
- **Offset:** 0x3C
- **Reset Value:** 0x00000000

Table 4-80 CLK_SEL Register

Bits	Field Name	RW	Reset	Description
31:2	RSVD	R	0x0	Reserved bits
1:0	CLK_SEL	RW	0x0	• 0x02: LFRC_32K (LFRC 32 kHz Clock) • 0x03: LFXO_32K (LF_XO 32 kHz Clock)

4.3.5.3.7 CMP_SET

- **Name:** CMP_SET Register
- **Description:** This register is used to set the COMPARE timer value.
- **Base Address:** 0x4000A600 + x*0x80
- **Offset:** 0x40
- **Reset Value:** 0x00000000

Table 4-81 CMP_SET Register

Bits	Field Name	RW	Reset	Description
31:0	CMP_SET	RW	0x0	• Write: RTC COMPARE timer value to configure • Read: The value you just set. Read CMP_RD to get the current internal value.

4.3.5.3.8 TIMER_SET

- **Name:** TIMER_SET Register
- **Description:** This register is used to set the RTC timer value.
- **Base Address:** 0x4000A600 + x*0x80
- **Offset:** 0x44
- **Reset Value:** 0x00000000

Table 4-82 TIMER_SET Register

Bits	Field Name	RW	Reset	Description
31:0	TIMER_SET	RW	0x0	• Write: RTC timer value to configure

Bits	Field Name	RW	Reset	Description
				• Read: The value you just set. Read TIMER_RD to get the current internal value.

4.3.5.3.9 TICK_SET

- **Name**: TICK_SET Register
- **Description**: This register is used to set the tick event timer value
- **Base Address**: 0x4000A600 + x*0x80
- **Offset**: 0x48
- **Reset Value**: 0x00000000

Table 4-83 TICK_SET Register

Bits	Field Name	RW	Reset	Description
31:0	TICK_SET	RW	0x0	• Write: RTC tick event timer value to configure • Read: The value you just set. Read TICK_RD to get the current internal value.

4.3.5.3.10 TIMER_RD

- **Name:** TIMER_RD Register
- **Description:** This register is used to get the internal value of the RTC timer.
- **Base Address:** 0x4000A600 + x*0x80
- **Offset:** 0x50
- **Reset Value:** 0x00000000

Table 4-84 TIMER_RD Register

Bits	Field Name	RW	Reset	Description
31:0	TIMER_RD	R	0x0	Current internal value of the RTC timer

4.3.5.3.11 CMP_RD

- **Name:** CMP_RD Register
- **Description:** This register is used to get the value of the RTC COMPARE timer.
- **Base Address:** 0x4000A600 + x*0x80
- **Offset:** 0x54
- **Reset Value:** 0x00000000

Table 4-85 CMP_RD Register

Bits	Field Name	RW	Reset	Description
31:0	CMP_RD	R	0x0	Current internal value of the RTC COMPARE timer

4.3.5.3.12 TICK_RD

- **Name:** TICK_RD Register
- **Description:** This register used to get the value of the RTC tick event timer.
- **Base Address:** 0x4000A600 + x*0x80
- **Offset:** 0x58
- **Reset Value:** 0x00000000

Table 4-86 TICK_RD Register

Bits	Field Name	RW	Reset	Description
31:0	TICK_RD	RW	0x0	Current internal value of the RTC tick event timer.

4.3.6 Sleep Timer (SLP TIMER)

4.3.6.1 Introduction

The Sleep Timer is a 32-bit hardware timer that can wake the SoC up from the Deep Sleep state. It can also be used while the SoC is awake.

4.3.6.2 Main Features

- 32-bit down counter running with two different clock sources: LFXO_32K clock (32.768 kHz), and LFRC_32K clock (about 32 kHz).
- Three operation modes: Basic Sleep Timer, Single-load Sleep Timer, Auto-reload Sleep Timer.

4.3.6.3 Functional Description

The Sleep Timer supports three operation modes.

- Mode 0 – Basic Sleep Timer: This is the original mode of the Sleep Timer. It only counts down when the SoC is in Deep Sleep state and is not activated when the SoC is awake.
- Mode 1 – Single-load Sleep Timer: In this mode, the timer starts counting as soon as it is put in this mode and it continues to count down even when the SoC is awake. The timer stops when the counter hits zero. The time-out signal is saved as a timeout event.
- Mode 2 – Auto-reload Sleep Timer: Similar to Single-load Sleep Timer mode, the timer does not stop counting down even when the SoC is awake. However, the timer can automatically reload to a value that is the same as that before automatic reload until it is disabled by the MCU.

4.3.6.4 Registers

4.3.6.4.1 CFG_0

- **Base Address:** 0x4000A500
- **Offset:** 0x00
- **Reset Value:** 0x00000000

Table 4-87 CFG_0 Register

Bits	Field Name	RW	Reset	Description
24	SLP_TIME_CFG	W	N/A	**Write**: • Writing 1 to this register triggers the Sleep Timer configuration. • Writing to this bit during sleep timer configuration will generate a configuration failure interrupt. **Read**: • The read value is always 0.
3	SLP_TIMER_CNT_SLP_ONLY	RW	0x0	**Value**: • 0x1: Count during sleep only. • 0x0: Count in any condition.
2	SLP_TIMER_SINGLE_EN	RW	0x0	**Value**: • 0x1: Single-load mode • 0x0: Auto reload mode
1	SLP_TIMER_SET	RW	0x0	Sets the timer value; write 1 to make timer value effective.
0	SLP_TIMER_EN	RW	0x0	Enables Sleep Timer. **Value**: • 0x1: Enable Sleep Timer. • 0x0: Disable Sleep Timer.

4.3.6.4.2 STS

- **Base Address:** 0x4000A500
- **Offset:** 0x38
- **Reset Value:** 0x00000000

Table 4-88 STS Register

Bits	Field Name	RW	Reset	Description
1	SLP_TIMER_BUSY	R	N/A	Sleep Timer is running.

Bits	Field Name	RW	Reset	Description
0	SLP_TIMER_STATUS	R	N/A	Sleep Timer is busy. Do not configure this bit if the signal is 1.

4.3.6.4.3 CLK

- **Base Address:** 0x4000A500
- **Offset:** 0x3C
- **Reset Value:** 0x00000000

Table 4-89 CLK Register

Bits	Field Name	RW	Reset	Description
1:0	SLP_TIMER_CLK_SEL	RW	0x0	Selects Sleep Timer clock. **Value**: • 0x02: LFRC_32K (LFRC 32 kHz Clock) • 0x03: LFXO_32K (LF_XO 32 kHz Clock)

4.3.6.4.4 VAL_W

- **Base Address:** 0x4000A500
- **Offset:** 0x40
- **Reset Value:** 0x00000000

Table 4-90 VAL_W Register

Bits	Field Name	RW	Reset	Description
31:0	SLP_TIMER_VAL_SET	RW	0x0	**Write**: • Write the Sleep Timer value to the VAL_W register. **Read**: • The value you just set. Read SLP_TIMER_VAL_RD in the VAL_R register for the current interval value.

4.3.6.4.5 VAL_R

- **Base Address:** 0x4000A500
- **Offset:** 0x48
- **Reset Value:** 0x00000000

Table 4-91 VAL_R

Bits	Field Name	RW	Reset	Description
31:0	SLP_TIMER_VAL_RD	R	N/A	Read internal value of the Sleep Timer.

4.4 Communication Interfaces

4.4.1 UART

4.4.1.1 Introduction

The device has four instances of Universal Asynchronous Receiver/Transmitter (UART) module. The UART complies with the industry-standard 16550. All the four UART instances support full duplex asynchronous serial communication with hardware flow control. The register address space is relocated to 32-bit data boundaries for Advance Peripheral Bus (APB) implementation.

4.4.1.2 Main Features

- Full-duplex asynchronous communications
- Programmable character properties
- Transmit and receive FIFO depth at 128 bytes
- Hardware flow control support
- DMA support
- IrDA 1.0 SIR support
- Four UART instances

4.4.1.3 Block Diagram

The functional block diagram of UART is shown in Figure 4-8.

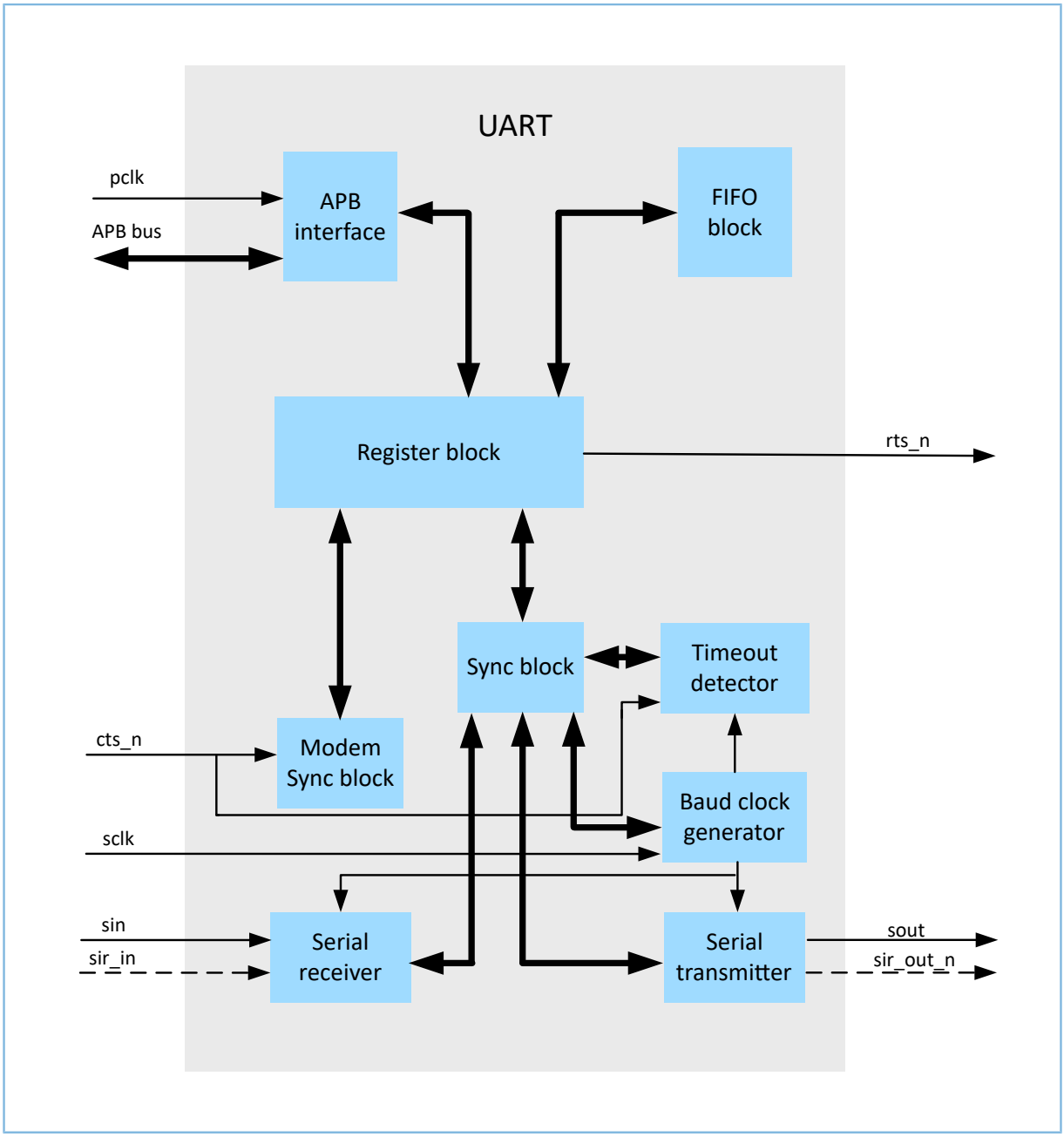

Figure 4-8 UART block diagram

The following list describes each of the major blocks shown in Figure 4-8.

- Reset Block resets the clock domain.
- APB Interface connects to APB bus.
- Register Block is responsible for the main UART functionality including control, status, and interrupt generation.
- Modem Sync Block synchronizes the modem input signal.
- FIFO Block is responsible for FIFO control and storage — when using internal RAM — or optionally signaling to control external RAM.

- Sync Block is implemented when the peripheral is configured to have a separate serial data clock.
- Timeout Detector Block indicates the absence of character data movement in the receiver FIFO within a given time period; this is used to generate character timeout interrupts when enabled.
- Baud Clock Generator produces the transmitter and receiver baud clock along with the output reference clock signal.
- Serial Transmitter converts the parallel data — written to the UART — into serial form and adds all additional bits, as specified by the control register, for transmission.
    - Serial UART
    - IrDA 1.0 SIR
- Serial Receiver converts the serial data character — specified by the control register — received in UART form. This block controls:
    - Parity error detection
    - Framing error detection
    - Line break detection

4.4.1.4 Functional Description

4.4.1.4.1 UART (RS232) Serial Protocol

The serial communication between the UART and a selected device is asynchronous, so additional start and stop bits are added to the serial data to indicate the beginning and end. Utilizing these bits allows two devices to be synchronized. This structure of serial data, accompanied by start and stop bits, is referred to as a character, as shown in Figure 4-9.

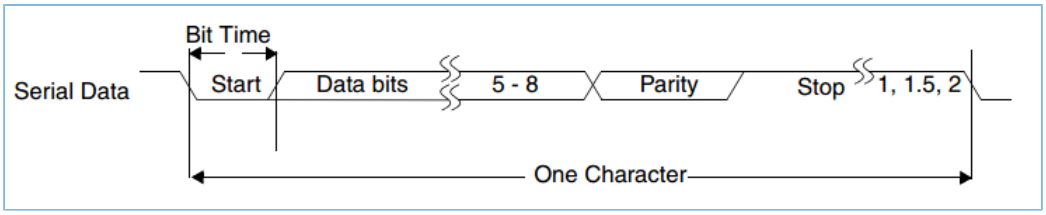

Figure 4-9 Timing diagram

An additional parity bit can be added to the serial character. This bit appears after the last data bit and before the stop bit(s) in the character structure in order to provide the UART with the ability to perform simple error checking on the received data.

The UART Line Control Register (LCR) is used to control the serial character characteristics. The individual bits of the data word are sent after the start bit, starting with the least-significant bit (LSB). These are followed by the optional parity bit, followed by the stop bit(s), which can be 1, 1.5, or 2.

All the bits in the transmission are transmitted for exactly the same time duration; the exception to this is the half-stop bit when 1.5 stop bits are used. This duration is referred to as a Bit Period or Bit Time; one Bit Time equals sixteen baud clocks.

To ensure stability on the line, the receiver samples the serial input data at approximately the midpoint of the Bit Time once the start bit has been detected. The exact number of baud clocks that are transmitted by each bit is known, so calculating the midpoint for sampling is not difficult; the midpoint for sampling occurs every sixteen baud clocks after sampling is performed at the midpoint of the start bit.

Together with serial input debouncing, this sampling helps to avoid the detection of false start bits. Short glitches are filtered out by debouncing, and no transition is detected on the line. If a glitch is wide enough to avoid filtering by debouncing, a falling edge is detected. However, a start bit is detected only if the line is again sampled low after half a Bit Time has elapsed. Figure 4-10 shows the sampling points of the first two bits in a serial character.

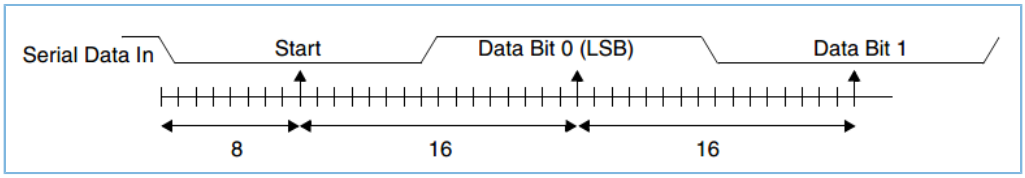

Figure 4-10 Receiver serial data sample points

As part of the 16550 standard, an optional baud clock reference output signal (baudout_n) provides timing information for receiving devices that require it. The baud rate of the UART is controlled by the serial clock – sclk (in asynchronous serial clock implementation, see Figure 4-8) and the Divisor Latch Register (DLH and DLL).

Serial data baud rate can be calculated as follows: Baud rate = (System clock frequency)/(16*Divisor). The Divisor is composed of an integer part (DIV_LATCH_HIGH/DIV_LATCH_LOW register) and a fractional part (4-bit DIV_LATCH_FRACTION register).

4.4.1.4.2 Fractional Baud Rate Support

UART supports fractional baud rate that enables users to program the fractional part of the divisor value to generate a fractional baud rate that results in reduced frequency error. The baud rate of UART is controlled by sclk and the Divisor Latch Register (DLH and DLL).

The baud rate is determined by the following factors:

- Serial clock operating frequency (sclk in asynchronous serial clock implementation)
- The desired baud rate, or Baud rate (selected)
- The baud rate generator divisor value, DIVISOR (composed of DLH & DLL registers).
- The acceptable baud-rate error: Percentage ERROR.

The equation to calculate the baud rate is as follows:

$$Baud\ Rate = \frac{Serial\ clock\ operating\ frequency}{(16 \times DIVISOR)}$$

where

DIVISOR – Number (in hexadecimal) to program the DLL and DLH. Serial clock operating frequency – Frequency at sclk pin of UART. The Error between the Baud rate and Baud rate (selected) is given as:

$$Percentage\,ERROR = \frac{|Baud\ Rate - Baud\ Rate(selected)|}{(Baud\ Rate)} \times 100$$

When configuring the UART for Fractional Baud Rate, pay attention to the following:

- The configurable parameter DLF_SIZE is used to choose the width of the register that stores fractional part of the divisor.
- The fractional value of the divisor is programmed in the Divisor Latch Fraction (DLF) register. The fractional value is computed by using the $(DivisorFractionValue)/(2^{\wedge}DLF_SIZE)$ formula. Table 4-92 shows fractional values when the **DLF_SIZE = 4**.

Table 4-92 Divisor latch fractional values

DLF Value	Fraction	Fractional Value
0000	0/16	0.0000
0001	1/16	0.0625
0010	2/16	0.125
0011	3/16	0.1875
0100	4/16	0.25
0101	5/16	0.3125
0110	6/16	0.375
0111	7/16	0.4375
1000	8/16	0.5
1001	9/16	0.5625
1010	10/16	0.625
1011	11/16	0.6875
1100	12/16	0.75
1101	13/16	0.8125
1110	14/16	0.875
1111	15/16	0.9375

- The programmable fractional baud rate divisor enables a finer resolution of baud clock than the conventional integer divider. The programmable fractional baud clock divider allows for the programmability of both an integer divisor as well as fractional component. The average frequency of the baud clock from the fractional baud rate divisor is dependent upon both the integer divisor and the fractional component, thereby providing a finer resolution to the average frequency of the baud clock.

$$DIVISOR = \frac{Serial\ clock\ frequency}{(16 \times Required\ baud\ rate)} = BRD_I + BRD_F$$

where

BRD_I: Integer part of the divisor

BRD_F: Fractional part of the divisor

4.4.1.4.3 FIFO Support

The UART module provides two 128-byte FIFO submodules: one (UART TX FIFO) for transmission and one (UART RX FIFO) for reception. By configuring the Shadow RCVR Trigger (SRT) register, you can program the trigger level in the RX FIFO at which the Received Data Available Interrupt is generated. Available configurations include 1 character, 1/4 full, 1/2 full, and 2 less than full. For example, if the 1/4 full option is selected for the receiver FIFO, the UART generates a receive interrupt after 32 data bytes are received. By configuring the Shadow TX Empty Trigger (STET) register, you can program the trigger level in the TX FIFO at which the Empty Threshold Reached Interrupt is generated. Available configurations include empty, 2 characters, 1/4 full, and 1/2 full. By configuring the Shadow FIFO Enable (SFE) register, you can enable/disable transmit and receive FIFOs.

4.4.1.4.4 Clock Support

UART is configured to have two system clocks (pclk and sclk). Pclk is the APB clock and sclk is the serial clock (See Figure 4-8). This mode accommodates accurate serial baud rate settings, as well as APB bus interface requirements. In dual clock mode, a synchronization module is implemented for synchronization of all control and data across the two-system clock boundaries. There are two types of signal synchronization:

- Data-synchronized signals – full synchronization handshake takes place on signals.
- Level-synchronized signals – signals are passed through two destination clock registers.

Both synchronization types incur additional data path latencies. However, this additional latency has no negative effect on received or transmitted data, other than to limit how faster sclk can be in relation to pclk for serial communications with no idle assertion.

A serial clock that exceeds this limit does not leave enough time for a complete incoming character to be received and pushed into the receiver FIFO. To ensure that you do not exceed the limit, the following equation must hold true:

$$((2 * pclk_cycles) + 4) < (39 * (BaudDivisor))$$

Where: pclk_cycles is expressed in sclk cycles, and Baud Divisor is determined by DLH and DLL. In most cases, the pclk signal is faster than sclk, so this should not be an issue.

4.4.1.4.5 Interrupts

Assertion of the UART interrupt output signal occurs whenever one of the several prioritized interrupt types are enabled and active. When an interrupt occurs, the master accesses the IIR register. The following interrupt types can be enabled with the IER register:

- Receiver Error
- Receiver Data Available

- Character Timeout (in FIFO mode only)
- Transmitter Holding Register Empty at/below threshold (in Programmable THRE interrupt mode)
- Modem Status

These interrupt types are explained in detail in Table 4-93.

Table 4-93 Interrupt control functions

Interrupt ID	Interrupt Set and Reset Functions			
Bit3 Bit2 Bit1 Bit0	**Priority Level**	**Interrupt Type**	**Interrupt Source**	**Interrupt Reset Control**
0 0 0 1	-	None	None	-
0 1 1 0	Highest	Receiver line status	Overrun/Parity/Framing errors, break interrupt, or address received interrupt	For overrun/parity/framing/break interrupt reset control, the behavior is as follows: • If LSR_STATUS_CLEAR=0 (RBR Read or LSR Read), the status is cleared on: Reading the line status register. In addition to an LSR read, the Receiver line status is also cleared when RX_FIFO is read. • If LSR_STATUS_CLEAR=1 (LSR Read), the status is cleared only on: Reading the line status register. For address received interrupt, the status is cleared on: Reading the LINE_STAT register.
0 1 0 0	Second	Received data available	Receiver data available (non-FIFO mode or FIFOs disabled) or RCVR FIFO trigger level reached (FIFO mode and FIFOs enabled)	Reading the receiver buffer register (non-FIFO mode or FIFOs disabled) or the FIFO increases above the trigger level (FIFO mode and FIFOs enabled).
1 1 0 0	Second	Character time indication	No characters in or out of the RCVR FIFO during the last 4 character times and there is at least 1 character in it during this time	Reading the receiver buffer register
0 0 1 0	Third	Transmit holding register empty	Transmitter holding register empty (Prog. THRE Mode disabled) or XMIT FIFO at or below threshold (Prog. THRE Mode enabled)	Reading the IIR register (if source of interrupt); or, writing into THR (FIFOs or THRE Mode not selected or disabled) or XMIT FIFO above threshold (FIFOs and THRE Mode selected and enabled)

Interrupt ID	Interrupt Set and Reset Functions			
Bit3 Bit2 Bit1 Bit0	**Priority Level**	**Interrupt Type**	**Interrupt Source**	**Interrupt Reset Control**
0 0 0 0	Fourth	Modem status	Clear to send or data set ready or ring indicator or data carrier detect. Note: If auto flow control mode is enabled, a change in Clear to Send (CTS) (that is, DCTS set) does not cause an interrupt.	Reading the Modem status register

4.4.1.4.6 Auto Flow Control

The UART can be configured to have a 16750-compatible Auto Request to Send (RTS) and Auto Clear To Send (CTS) serial data flow control mode available; if FIFOs are not implemented, this mode cannot be selected. When Auto Flow Control is not selected, none of the corresponding logics is implemented, and the mode cannot be enabled, reducing overall gate counts. When Auto Flow Control mode is selected, it can be enabled with the Modem Control Register (MCR[5]). Figure 4-11 shows a block diagram of the Auto Flow Control functionality.

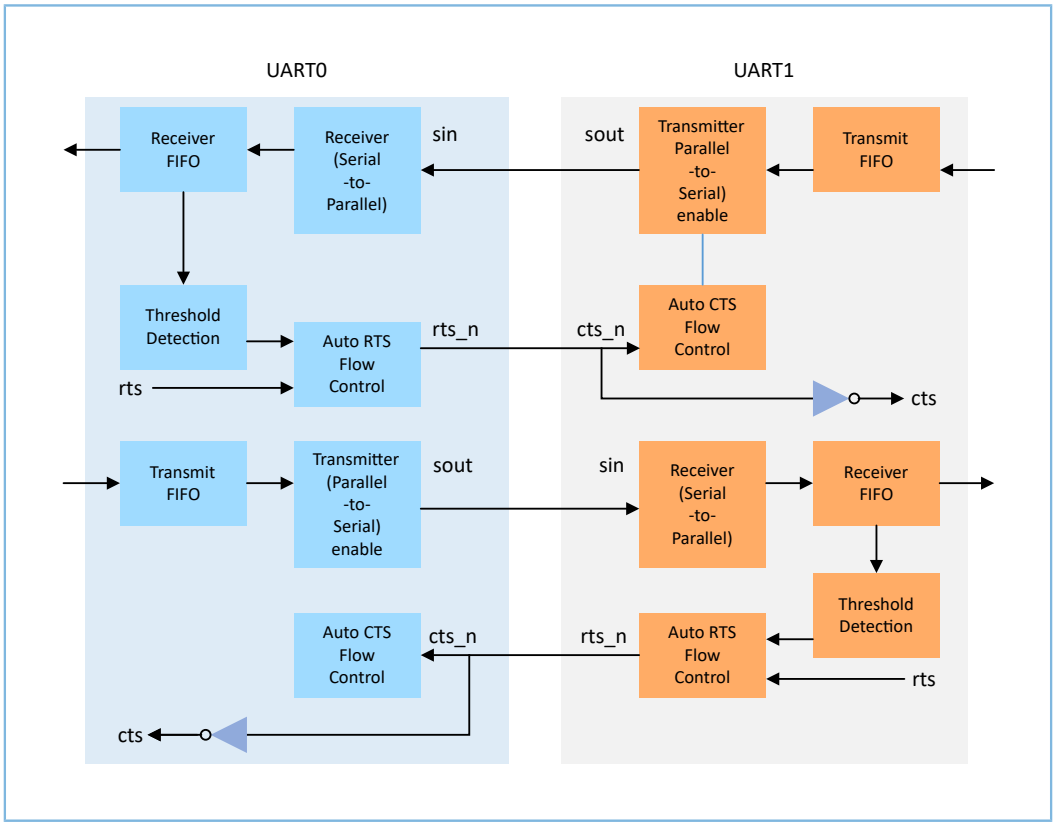

Figure 4-11 Auto Flow Control Block Diagram

- Auto RTS becomes active when all the following conditions occur:
  - Auto Flow Control is selected during configuration
  - FIFOs are implemented
  - RTS (MCR[1] bit and MCR[5] bit are both set)
  - FIFOs are enabled (FCR[0]) bit is set)
  - SIR mode is disabled (MCR[6] bit is not set)

    When Auto RTS is enabled, the rts_n output is forced inactive (high) when the FIFO is almost full, where “almost full” refers to two available slots in the FIFO. When rts_n is connected to the cts_n input of another UART device, the other UART stops sending serial data until the receiver FIFO has available space; that is, until it is completely empty.

    The selectable receiver FIFO threshold values are:

    - 1
    - 1/4
    - 1/2
    - 2 less than full

  Since one additional character can be transmitted to the UART after rts_n has become inactive—due to data already having entered the transmitter block in the other UART— setting the threshold to “2 less than full” allows maximum use of the FIFO with a safety zone of one character.

  When the receiver FIFO becomes completely empty by reading the Receiver Buffer Register (RBR), the rts_n again becomes active (low), signaling the other UART to continue sending data. Figure 4-12 shows a timing diagram of the Auto RTS operation.

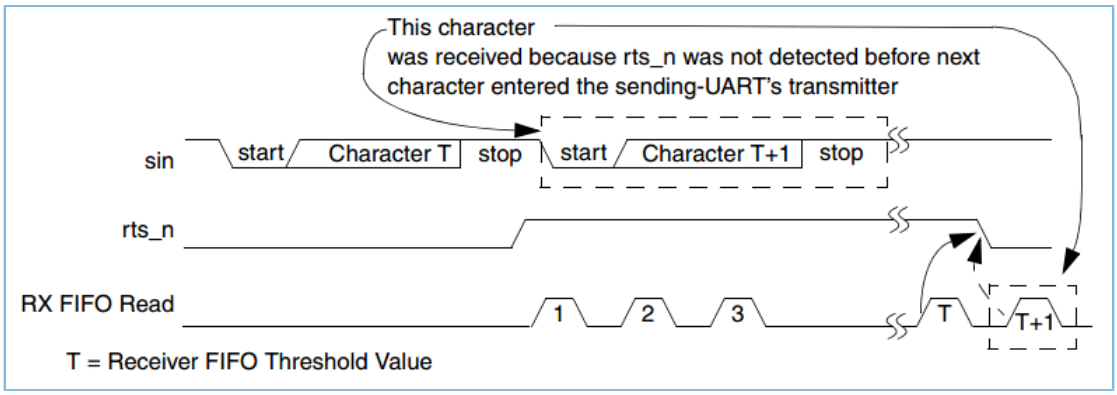

Figure 4-12 Auto RTS timing

- Auto CTS becomes active when all the following conditions occur:
  - Auto Flow Control is selected during configuration
  - FIFOs are implemented
  - AFCE (MCR[5] bit = **1**)

- FIFOs are enabled through FIFO Control Register FCR[0] bit
- SIR mode is disabled (MCR[6] bit = 0)

  When Auto CTS is enabled (active), the UART transmitter is disabled whenever the cts_n input becomes inactive (high); this prevents overflowing the FIFO of the receiving UART. If the cts_n input is not inactivated before the middle of the last stop bit, another character is transmitted before the transmitter is disabled. While the transmitter is disabled, the transmitter FIFO can still be written to, and even overflowed. Therefore, when using this Auto CTS mode, the following happens:

- UART status register can be read to check if transmit FIFO is full (USR[1] set to **0**).
- Current FIFO level can be read using TFL register.
- Programmable THRE Interrupt mode must be enabled to access “FIFO full” status using Line Status Register (LSR).

When using the “FIFO full” status, software can poll this before each write to the Transmitter FIFO. When the cts_n input becomes active (low) again, transmission resumes. Figure 4-13 illustrates a timing diagram that shows the Auto CTS operation.

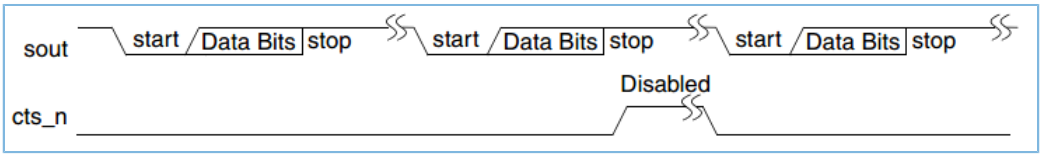

Figure 4-13 Auto CTS timing

4.4.1.4.7 Programmable THRE Interrupt

The UART can be configured to Programmable THRE Interrupt mode to increase system performance; if FIFOs are not implemented, this mode cannot be selected.

- When Programmable THRE Interrupt mode is not selected, none of the logics is implemented, and the mode cannot be enabled, reducing the overall gate counts.
- When Programmable THRE Interrupt mode is selected, it can be enabled using the Interrupt Enable Register (IER[7]).
- When FIFOs and THRE mode are implemented and enabled, the THRE Interrupts and dma_tx_req_n are active at and below a programmed transmitter FIFO empty threshold level, as opposed to empty, as shown in the flowchart in Figure 4-14.

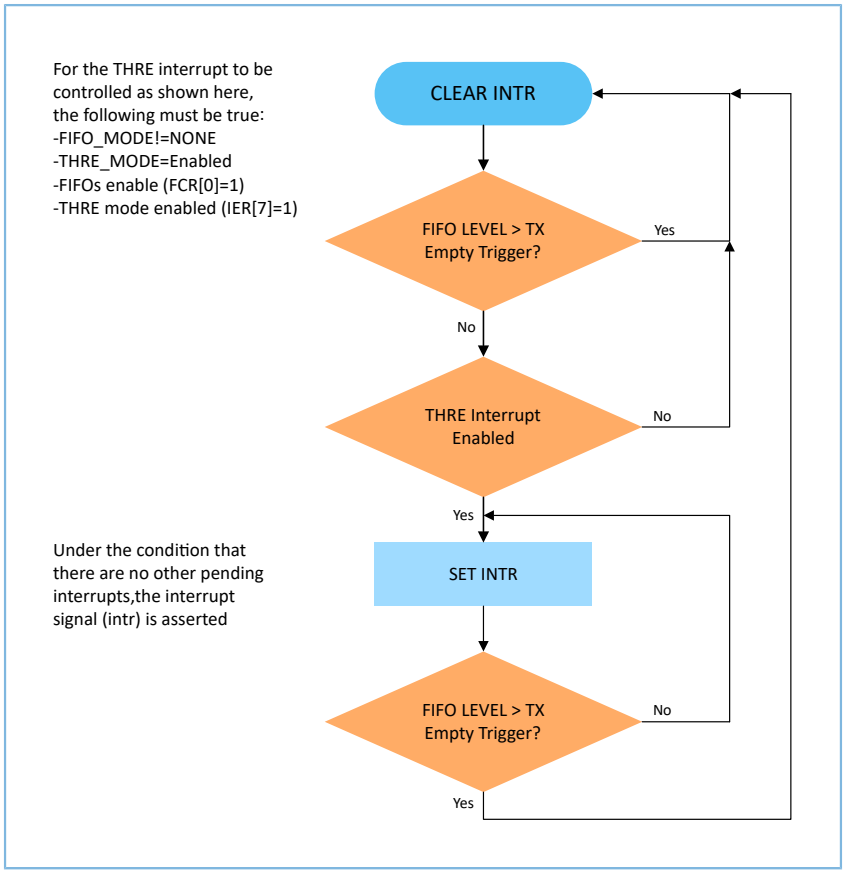

Figure 4-14 Flowchart of Interrupt Generation for Programmable THRE Interrupt Mode

The threshold level is programmed into FCR[5:4]. Available empty thresholds are:

- Empty
- 2
- 1/4
- 1/2

Selection of the best threshold value depends on the ability of the system to begin a new transmission sequence in a timely manner. However, one of these thresholds should be optimal for increasing system performance by preventing the transmitter FIFO from running empty.

In addition to the interrupt change, the Line Status Register (LSR[5]) also switches from indicating the transmitter FIFO being empty to being full. This allows software to fill the FIFO for each transmit sequence by polling LSR[5] before writing another character. The flow then allows the transmitter FIFO to be filled whenever an interrupt occurs and there is data to transmit, rather than waiting until the FIFO is completely empty. Waiting until the FIFO is empty causes a reduction in performance whenever the system is too busy to respond immediately. Further system efficiency is achieved when this mode is enabled in combination with Auto Flow Control.

Even if everything (the conditions are true for the THRE interrupt, as shown in the upper-left corner of Figure 4-14) is selected and enabled, if the FIFOs are disabled using the FCR[0] bit, the Programmable THRE Interrupt mode is also disabled. When the programmable THRE interrupt is not selected or disabled, THRE interrupts and the LSR[5] bit

function normally, signifying an empty THR or FIFO. Figure 4-15 illustrates the flowchart of THRE interrupt generation when not in programmable THRE interrupt mode.

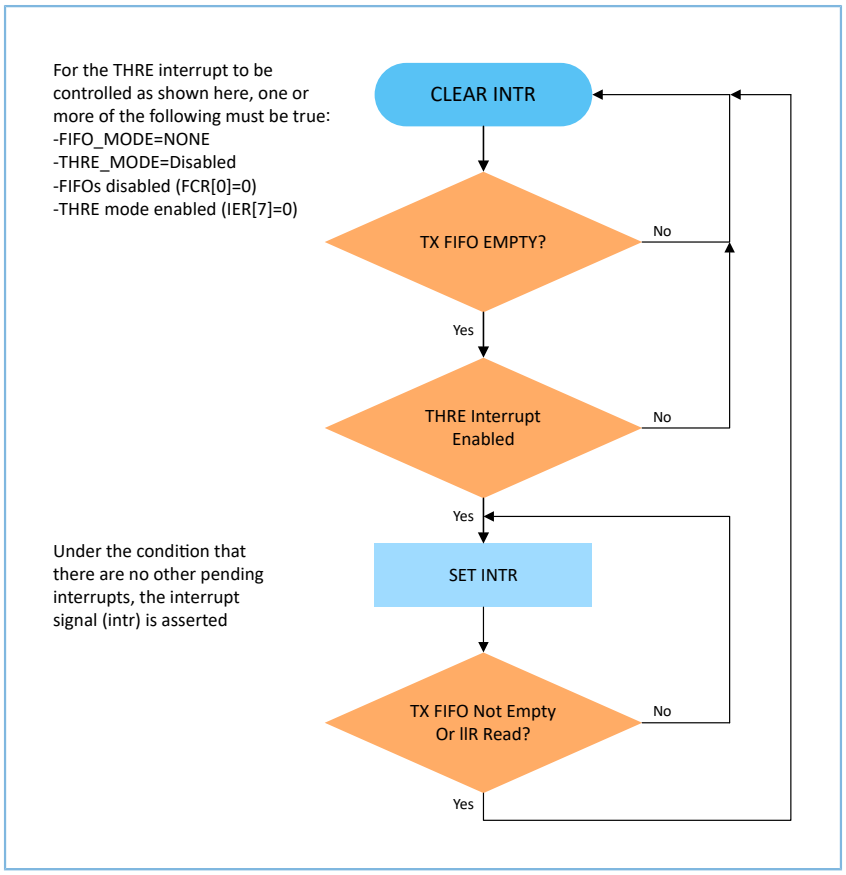

Figure 4-15 Interrupt generation when not in Programmable THRE Interrupt Mode

4.4.1.4.8 DMA Support

The UART uses two DMA channels: one for transmitting data and one for receiving data. There are two DMA modes: mode 0 and mode 1, controllable via bit 3 of the FCR register (only DMA mode 0 is available when the FIFOs are not implemented or disabled). DMA mode 0 supports single DMA data transfers at a time. DMA mode 1 supports multi-DMA data transfers, where multiple transfers are made continuously until the receiver FIFO has been emptied or the transmit FIFO has been filled. For the same DMA module, only one DMA channel is working at one time. The DMA request list of UART instance is as follows.

Table 4-94 UART-DMA handshake signal

Item	UART0	UART1	UART2	UART3	UART4	UART5
DMA0	√	√	√	√	√	
DMA1	√			√	√	√

Where, the “√” represents the UARTx can send DMAx request signal.

4.4.1.4.9 IrDA 1.0 SIR Protocol

The Infrared Data Association (IrDA) 1.0 Serial Infrared (SIR) mode supports bi-directional data communications with remote devices using infrared radiation as the transmission medium. IrDA 1.0 SIR mode specifies a maximum baud rate of 115.2 Kbaud.

The data format is similar to that of standard serial data. Each data character is sent serially in the following order:

- Begins with a start bit
- Followed by 8 data bits
- Ends with at least one stop bit

Thus, the number of data bits that can be sent is fixed. No parity information can be supplied, and only one stop bit is used in this mode. Trying to adjust the number of data bits sent or enable parity with the Line Control Register (LCR) has no effect.

You can configure the UART for IrDA 1.0 SIR to perform the following:

- Bit 6 of the Mode Control Register (MCR) enables or disables the IrDA 1.0 SIR mode.
- Disabling IrDA SIR mode causes the logic not to be implemented; the mode cannot be activated, which reduces total gate counts.
- When IrDA SIR mode is enabled and active, serial data is transmitted and received respectively on the sir_out_n and sir_in ports.

Transmission or non-transmission of a single infrared pulse indicates the following:

- Transmitting a single infrared pulse indicates logic 0.
- Non-transmission of a pulse indicates logic 1.

The width of each pulse is three sixteenths of a normal serial bit time. Thus, each new character begins with an infrared pulse for the start bit. However, received data is inverted from transmitted data due to infrared pulses energizing the photo transistor base of the IrDA receiver, which pulls its output low. This inverted transistor output is then fed to the UART sir_in port, which gives it the correct UART polarity.

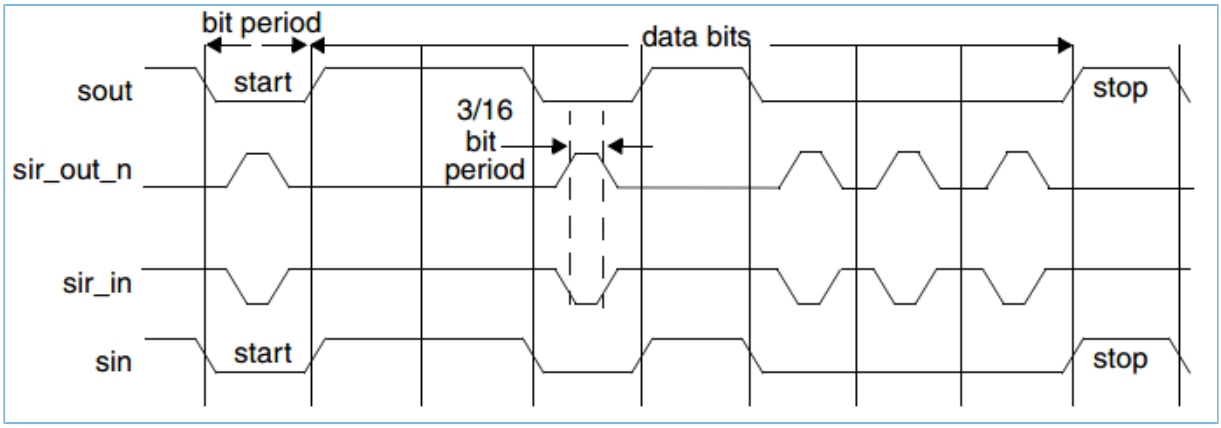

Figure 4-16 IrDA SIR data format

When IrDA SIR mode is enabled, the UART operates in a manner similar to when the mode is disabled, with one exception: data transfers can only occur in half-duplex fashion when IrDA SIR mode is enabled. This is because the IrDA

SIR physical layer specifies a minimum of 10 ms delay between transmission and reception; this 10 ms delay must be generated by software.

4.4.1.5 Registers

4.4.1.5.1 RBR

- **Name**: Receive Buffer Register
- **Description**: This register can be accessed only when the DIV_LATCH_AB bit (LCR[7]) is cleared.
- **Base Address**: 0x4000C500 + x*0x100
- **Offset**: 0x0
- **Reset Value**: 0x00000000

Note:

*x is used to identify the UART Module instance. The “x” can be 0, 1, 2, 3, 4, or 5.

Table 4-95 Receive Buffer Register

Bits	Field Name	RW	Reset	Description
31:8	RSVD	R		Reserved bits
7:0	RX_BUF	R	0x0	Receive Buffer Register. This register contains the data byte received on the serial input port (sin) in UART mode. The data in this register is valid only if the Data Ready (DATA_READY) bit in the Line Status Register (LSR)) is set.

4.4.1.5.2 DLL

- **Name**: Divisor Latch Low Register
- **Description**: This register can be accessed only when the DIV_LATCH_AB bit (LCR[7]) is set.
- **Base Address**: 0x4000C500 + x*0x100
- **Offset**: 0x00
- **Reset Value**: 0x00000000

Table 4-96 Divisor Latch Low Register

Bits	Field Name	RW	Reset	Description
31:8	RSVD	R		Reserved bits
7:0	DIV_LATCH_LOW	RW	0x0	Divisor Latch (Low).

Bits	Field Name	RW	Reset	Description
				This register makes up the lower 8 bits of a 16-bit, read/write, Divisor Latch register that contains the baud rate divisor for the UART. Note that with the Divisor Latch Registers (DIV_LATCH_LOW and DIV_LATCH_HIGH) set to zero, the baud clock is disabled and no serial communications will occur. In addition, once the DIV_LATCH_LOW is set, at least 8 clock cycles of the slowest UART clock should be allowed to pass before transmitting or receiving data.

4.4.1.5.3 THR

- **Name**: Transmit Holding Register
- **Description**: This register can be accessed only when the DIV_LATCH_AB bit (LCR[7]) is cleared.
- **Base Address**: 0x4000C500 + x*0x100
- **Offset**: 0x00
- **Reset Value**: 0x00000000

Table 4-97 Transmit Holding Register

Bits	Field Name	RW	Reset	Description
31:8	RSVD	R	0x0	Reserved bits
7:0	TX_HDG	W	0x0	Transmit Holding Register. This register contains data to be transmitted on the serial output port (sout) in UART mode. Data should only be written to the TX_HDG when the TX_HDG Empty (TX_HDG_EMPTY) bit (LSR [5]) is set.

4.4.1.5.4 DLH

- **Name**: Divisor Latch High Register
- **Description**: This register can be accessed only when the DIV_LATCH_AB bit (LCR [7]) is set.
- **Base Address**: 0x4000C500 + x*0x100
- **Offset**: 0x04
- **Reset Value**: 0x00000000

Table 4-98 Divisor Latch High Register

Bits	Field Name	RW	Reset	Description
31:8	RSVD	R		Reserved bits
7:0	DIV_LATCH_HIGH	RW	0x0	Divisor Latch (HIGH). Upper 8-bit of a 16-bit, read/write, Divisor Latch register that contains the baud rate divisor for the UART. Note that with the Divisor Latch Registers (DIV_LATCH_LOW and DIV_LATCH_HIGH) set to zero, the baud clock is disabled and no serial communications will occur. Also, once the DIV_LATCH_HIGH is set, at least 8 clock cycles of the slowest UART clock should be allowed to pass before transmitting or receiving data.

4.4.1.5.5 IER

- **Name**: Interrupt Enable Register
- **Description**: This register can be accessed only when the DIV_LATCH_AB bit (LCR [7]) is cleared.
- **Base Address**: 0x4000C500 + x*0x100
- **Offset**: 0x04
- **Reset Value**: 0x00000000

Table 4-99 Interrupt Enable Register

Bits	Field Name	RW	Reset	Description
31:8	RSVD	R		Reserved bits
7	INT_GTHEE	RW	0x0	This is used to enable/disable the generation of TX_HDG_EMPTY Interrupt. **Value**: • 0x0 (DISABLED): Disable Programmable TX_HDG_EMPTY Interrupt Mode • 0x1 (ENABLED): Enable Programmable TX_HDG_EMPTY Interrupt Mode
6:4	RSVD	R		Reserved bits
3	INT_MSE	RW	0x0	Enable Modem Status Interrupt. This is used to enable/disable the generation of Modem Status Interrupt. This is the fourth highest priority interrupt. **Value**:

Bits	Field Name	RW	Reset	Description
				• 0x0 (DISABLED): Disable Modem Status Interrupt • 0x1 (ENABLED): Enable Modem Status Interrupt
2	INT_RX_LSE	RW	0x0	Enable Receiver Line Status Interrupt. This is used to enable/disable the generation of Receiver Line Status Interrupt. This is the highest priority interrupt. **Value**: • 0x0 (DISABLED): Disable Receiver Line Status Interrupt • 0x1 (ENABLED): Enable Receiver Line Status Interrupt
1	INT_TX_HEE	RW	0x0	Enable Transmit Holding Register Empty Interrupt. This is used to enable/disable the generation of Transmitter Holding Register Empty Interrupt. This is the third highest priority interrupt. **Value**: • 0x0 (DISABLED): Disable Transmit empty interrupt • 0x1 (ENABLED): Enable Transmit empty interrupt
0	INT_RX_EN	RW	0x0	Enable Received Data Available Interrupt. This is used to enable/disable the generation of Received Data Available Interrupt and the Character Timeout Interrupt (if in FIFO mode and FIFO's enabled). These are the second highest priority interrupts. **Value**: • 0x0 (DISABLED): Disable Receive data Interrupt • 0x1 (ENABLED): Enable Receive data Interrupt

4.4.1.5.6 FCR

- **Name**: FIFO Control Register
- **Description**: FIFO Control Register
- **Base Address**: 0x4000C500 + x*0x100
- **Offset**: 0x08
- **Reset Value**: 0x00000400

Table 4-100 FIFO Control Register

Bits	Field Name	RW	Reset	Description
31:8	RSVD	R		Reserved bits

Bits	Field Name	RW	Reset	Description
7:6	RX_FIFO_TRG	W	0x0	RX Trigger (or RX_FIFO_TRG). This is used to select the trigger level in the receiver FIFO at which the Received Data Available Interrupt will be generated. In auto flow control mode, it is used to determine when the rts_n signal will be de-asserted only when RTC_FCT is disabled. It also determines when the dma_rx_req_n signal will be asserted in certain modes of operation. **Value**: • 0x0 (FIFO_CHAR_1): 1 character in FIFO • 0x1 (FIFO_QUARTER_FULL): FIFO 1/4 full • 0x2 (FIFO_HALF_FULL): FIFO 1/2 full • 0x3 (FIFO_FULL_2): FIFO 2 less than full
5:4	TX_EMPTY_TRG	W	0x0	TX Empty Trigger (or TX_EMPTY_TRG). This is used to select the empty threshold level at which the TX_HDG_EMPTY Interrupts will be generated when the mode is active. It also determines when the dma_tx_req_n signal will be asserted when in certain modes of operation. **Value**: • 0x0 (FIFO_EMPTY): FIFO Empty • 0x1 (FIFO_CHAR_2): 2 characters in FIFO • 0x2 (FIFO_QUARTER_FULL): FIFO 1/4 full • 0x3 (FIFO_HALF_FULL): FIFO 1/2 full
3	DMA_MODE	W	0x0	DMA Mode (or DMA_MODE). This determines the DMA signaling mode used for the dma_tx_req_n and dma_rx_req_n output signals when additional DMA handshaking signals are not selected. **Value**: • 0x0 (MODE0): Mode 0 • 0x1 (MODE1): Mode 1
2	TX_FIFO_RST	W	0x0	TX FIFO Reset (or TX_FIFO_RST). This resets the control portion of the transmit FIFO and treats the FIFO as empty. This will also de-assert the DMA TX request and single signals when additional DMA handshaking signals are selected.

Bits	Field Name	RW	Reset	Description
				Value: 0x1 (RESET): Transmit FIFO reset
1	RX_FIFO_RST	W	0x0	RX FIFO Reset (or RX_FIFO_RST). This resets the control portion of the receive FIFO and treats the FIFO as empty. This will also de-assert the DMA RX request and single signals when additional DMA handshaking signals are selected. **Value**: 0x1 (RESET): Receive FIFO reset
0	FIFO_EN	W	0x0	FIFO Enable (or FIFO_EN). This enables/disables the transmit (TX) and receive (RX) FIFOs. Whenever the value of this bit is changed both the TX and RX controller portion of FIFOs is reset. **Value**: • 0x0 (DISABLED): FIFO disabled • 0x1 (ENABLED): FIFO enabled

4.4.1.5.7 IIR

- **Name**: Interrupt Identification Register
- **Description**: Interrupt Identification Register
- **Base Address**: 0x4000C500 + x*0x100
- **Offset**: 0x08
- **Reset Value**: 0x00000001

Table 4-101 Interrupt Identification Register

Bits	Field Name	RW	Reset	Description
31:8	RSVD	R		Reserved bits
7:6	FIFO_EN	R	0x0	FIFOs Enabled (or FIFO_EN). This is used to indicate whether the FIFOs are enabled or disabled. **Value**: • 0x0 (DISABLED): FIFOs are disabled • 0x3 (ENABLED): FIFOs are enabled
5:4	RSVD	R		Reserved bits

Bits	Field Name	RW	Reset	Description
3:0	INT_PRIOR	R	0x1	Interrupt ID (or priority index). This indicates the highest priority pending interrupt which can be one of the following types specified in Values. **Value**: • 0x0 (MODEM_STATUS): modem status • 0x1 (NO_INTERRUPT_PENDING): no interrupt pending • 0x2 (TX_HDG _EMPTY): TX_HDG empty • 0x4 (RECEIVED_DATA_AVAILABLE): received data available • 0x6 (RECEIVER_LINE_STATUS): receiver line status • 0xC (CHARACTER_TIMEOUT): character timeout

4.4.1.5.8 LCR

- **Name**: Line Control Register
- **Description**: Line Control Register
- **Base Address**: 0x4000C500 + x*0x100
- **Offset**: 0x0C
- **Reset Value**: 0x00000000

Table 4-102 Line Control Register

Bits	Field Name	RW	Reset	Description
31:8	RSVD	R		Reserved bits
7	DIV_LATCH_AB	RW	0x0	Divisor Latch Access Bit. This bit is used to enable reading and writing of the Divisor Latch Register (DIV_LATCH_LOW and DIV_LATCH_HIGH) to set the baud rate of the UART. This bit must be cleared after initial baud rate setup in order to access other registers. **Value**: • 0x0 (DISABLED): Divisor Latch register is writable only when UART is not busy. • 0x1 (ENABLED): Divisor Latch register is always readable and writable
6	BREAK_CTRL_BIT	RW	0x0	Break Control Bit. This is used to cause a break condition to be transmitted to the receiving device. If set to one the serial output is forced to the spacing (logic 0) state. When not in Loopback Mode,

Bits	Field Name	RW	Reset	Description
				as determined by MODEM_CTRL[4], the sout line is forced low until the Break bit is cleared. When in Loopback Mode, the break condition is internally looped back to the receiver and the sir_out_n line is forced low. **Value**: • 0x0 (DISABLED): Serial output is released for data transmission • 0x1 (ENABLED): Serial output is forced to spacing state
5	STICK_PARITY	RW	0x0	Stick Parity. This bit is used to force parity value. When PARITY_EN, EVEN_PARITY_SEL and STICK_PARITY are set to 1, the parity bit is transmitted and checked as logic 0. If PARITY_EN and Stick Parity are set to 1 and EVEN_PARITY_SEL is a logic 0, then parity bit is transmitted and checked as a logic 1. If this bit is set to 0, Stick Parity is disabled. **Value**: • 0x0 (DISABLED): STICK_PARITY disabled • 0x1 (ENABLED): STICK_PARITY enabled
4	EVEN_PARITY_SEL	RW	0x0	Even Parity Select. This is used to select between even and odd parity, when parity is enabled (PEN set to one). If this bit is set to one, an even number of logic '1's is transmitted or checked. If set to zero, an odd number of logic '1's is transmitted or checked. **Value**: • 0x0 (ODD_PARITY): an odd parity is transmitted or checked • 0x1 (EVEN_PARITY): an even parity is transmitted or checked
3	PARITY_EN	RW	0x0	Parity Enable. This bit is used to enable and disable parity generation and detection in transmitted and received serial character respectively. **Value**: • 0x0 (DISABLED): disable parity • 0x1 (ENABLED): enable parity

Bits	Field Name	RW	Reset	Description
2	STOP_BITS	RW	0x0	Number of Stop bits. This is used to select the number of Stop bits per character that the peripheral will transmit and receive. If set to zero, one stop bit is transmitted in the serial data. If this bit is set to one and the Data bits are set to 5 (LCR[1:0] set to zero), one and a half Stop bits are transmitted. Otherwise, two Stop bits are transmitted. Note that regardless of the number of stop bits selected, the receiver will only check the first Stop bit. **Note**: The Stop bit duration implemented by UART may appear longer due to idle time inserted between characters for some configurations and baud clock divisor values in the transmit direction. **Value**: • 0x0 (STOP_1BIT): 1 stop bit • 0x1 (STOP_1_5BIT_OR_2BIT): 1.5 stop bits when DATA_LEN_SEL (LINE_CTRL[1:0]) is zero, else 2 stop bit
1:0	DATA_LEN_SEL	RW	0x0	Data Length Select (or CLS as used in legacy). When DATA_LEN_SEL_E in LCR_EXT is set to 0, this register is used to select the number of data bits per character that the peripheral will transmit and receive. **Value**: • 0x0 (CHAR_5BITS): 5 data bits per character • 0x1 (CHAR_6BITS): 6 data bits per character • 0x2 (CHAR_7BITS): 7 data bits per character • 0x3 (CHAR_8BITS): 8 data bits per character

4.4.1.5.9 MCR

- **Name**: Modem Control Register
- **Description**: Modem Control Register
- **Base Address**: 0x4000C500 + x*0x100
- **Offset**: 0x10
- **Reset Value**: 0x00000000

Table 4-103 Modem Control Register

Bits	Field Name	RW	Reset	Description
31:7	RSVD	R		Reserved bits
6	SIRE	RW	0x0	SIR Mode Enable Writeable only when SIR_MODE == Enabled, always readable. This is used to enable/disable the IrDA SIR Mode features. **Value**: • 0x0 (DISABLED): IrDA SIR Mode disabled. • 0x1 (ENABLED): IrDA SIR Mode enabled.
5	AFCE	RW	0x0	Auto Flow Control Enable Writeable only when AFCE_MODE == Enabled, always readable. When FIFOs are enabled and the Auto Flow Control Enable (AFCE) bit is set, Auto Flow Control features are enabled. **Value**: • 0x0 (DISABLED): Auto Flow Control Mode disabled. • 0x1 (ENABLED): Auto Flow Control Mode enabled.
4	LOOP_BACK	RW	0x0	Loopback Bit. This is used to put the UART into a diagnostic mode for test purposes. **Value**: • 0x0 (DISABLED): Loopback mode disabled • 0x1 (ENABLED): Loopback mode enabled
3:2	RSVD	R		Reserved bits
1	REQ_TO_SEND	RW	0x0	Request to Send. This is used to directly control the Request to Send (rts_n) output. The Request To Send (rts_n) output is used to inform the modem or data set that the UART is ready to exchange data. **Value**: • 0x0 (INACTIVE): Request to Send rts_n de-asserted (logic 1) • 0x1 (ACTIVE): Request to Send rts_n asserted (logic 0)
0	RSVD	R		Reserved bits

4.4.1.5.10 LSR

- **Name**: Line Status Register
- **Description**: Line Status Register
- **Base Address**: 0x4000C500 + x*0x100
- **Offset**: 0x14
- **Reset Value**: 0x00000000

Table 4-104 Line Status Register

Bits	Field Name	RW	Reset	Description
31:8	RSVD	R		Reserved bits
7	RX_FIFO_ERR	R	0x0	Receiver FIFO Error bit. This is used to indicate if there is at least one parity error, framing error, or break indication in the FIFO. This bit is cleared when the LSR is read and the character with the error is at the top of the receiver FIFO and there are no subsequent errors in the FIFO. **Value**: • 0x0 (NO_RX_FIFO_ERROR): No error in RX FIFO • 0x1 (RX_FIFO_ERROR): Error in RX FIFO
6	TX_EMPTY	R	0x1	Transmitter Empty bit. If FIFO is enabled (FCR[0] set to one), this bit is set whenever the Transmitter Shift Register and the FIFO are both empty. **Value**: • 0x0 (DISABLED): Transmitter not empty
5	TX_HDG_EMPTY	R	0x1	Transmit Holding Register Empty bit. If TX_HDG_EMPTY mode is disabled (IER[7] set to zero) and regardless of FIFO's being implemented/ enabled or not, this bit indicates that the TX_HDG or TX FIFO is empty. **Value**: • 0x0 (DISABLED): TX_HDG_EMPTY interrupt control is disabled • 0x1 (ENABLED): TX_HDG_EMPTY interrupt control is enabled
4	BREAK_INT	R	0x0	Break Interrupt bit. This is used to indicate the detection of a break sequence on the serial input data.

Bits	Field Name	RW	Reset	Description
				Value: • 0x0 (NO_BREAK): No break sequence detected • 0x1 (BREAK): Break sequence detected
3	FRAMING_ERR	R	0x0	Framing Error bit. This is used to indicate the occurrence of a framing error in the receiver. A framing error occurs when the receiver does not detect a valid STOP_BITS bit in the received data. **Value**: • 0x0 (NO_FRAMING_ERROR): no framing error • 0x1 (FRAMING_ERROR): framing error
2	PARITY_ERR	R	0x0	Parity Error bit. This is used to indicate the occurrence of a parity error in the receiver if the Parity Enable (PARITY_EN) bit (LCR[3]) is set. **Value**: • 0x0 (NO_PARITY_ERROR): no parity error • 0x1 (PARITY_ERROR): parity error
1	OVER_ERR	R	0x0	Overrun error bit. This is used to indicate the occurrence of an overrun error. This occurs if a new data character was received before the previous data was read. **Value**: • 0x0 (NO_OVER_RUN_ERROR): no overrun error • 0x1 (OVER_RUN_ERROR): overrun error
0	DATA_READY	R	0x0	Data Ready bit. This is used to indicate that the receiver contains at least one character in the RX_BUF or the receiver FIFO. This bit is cleared when the receiver FIFO is empty, in the FIFO mode. **Value**: • 0x0 (NOT_READY): data not ready • 0x1 (READY): data ready

4.4.1.5.11 MSR

- **Name**: Modem Status Register

- **Description**: Whenever bit 0 is set to logic 1 to indicate a change on the modem control inputs, a modem status interrupt will be generated if enabled via the INT_EN regardless of when the change occurred. The bits (bit 0) can be set after a reset, even though their respective modem signals are inactive, because the synchronized version of the modem signals have a reset value of 0 and change to value 1 after reset. To prevent unwanted interrupts due to this change, a read of the MODEM_STAT register can be performed after reset.
- **Base Address**: 0x4000C500 + x*0x100
- **Offset**: 0x18
- **Reset Value**: 0x00000000

Table 4-105 Modem Status Register

Bits	Field Name	RW	Reset	Description
31:5	RSVD	R		Reserved bits
4	CLR_SEND	R	0x0	Clear to Send. This is used to indicate the current state of the modem control line cts_n. That is, this bit is the complement cts_n. When the Clear to Send input (cts_n) is asserted, it is an indication that the modem or data set is ready to exchange data with the UART. In Loopback Mode (MODEM_CTRL[4] set to one), CLR_SEND is the same as MODEM_CTRL[1] (REQ_TO_SEND). **Value**: • 0x0 (DEASSERTED): cts_n input is de-asserted (logic 1) • 0x1 (ASSERTED): cts_n input is asserted (logic 0)
3:1	RSVD	R		Reserved bits
0	DELTA_CLR_SEND	R	0x0	Delta Clear to Send. This is used to indicate that the modem control line cts_n has changed since the last time the MODEM_STAT was read. Reading the MODEM_STAT clears the DELTA_CLR_SEND bit. In Loopback Mode (MODEM_CTRL[4] set to one), DELTA_CLR_SEND reflects changes on MODEM_CTRL[1] (REQ_TO_SEND). **Note**: If the DELTA_CLR_SEND bit is not set and the cts_n signal is asserted (low) and a reset occurs (software or otherwise), then the DELTA_CLR_SEND bit will get set when the reset is removed if the cts_n signal remains asserted. **Value**:

Bits	Field Name	RW	Reset	Description
				• 0x0 (NO_CHANGE): no change on cts_n since last read of MODEM_STAT • 0x1 (CHANGE): change on cts_n since last read of MODEM_STAT

4.4.1.5.12 SRBR

- **Name**: Shadow Receive Buffer Register
- **Description**: This register can be accessed only when the DIV_LATCH_AB bit (LINE_CTRL[7]) is cleared.
- **Base Address**: 0x4000C500 + x*0x100
- **Offset**: 0x30
- **Reset Value**: 0x00000000

Table 4-106 Shadow Receive Buffer Register

Bits	Field Name	RW	Reset	Description
31:8	RSVD	R		Reserved bits
7:0	SHADOW_RX_BUFn	R		Shadow Receive Buffer Register n. This is a shadow register for the RX_BUF and has been allocated sixteen 32-bit locations so as to accommodate burst accesses from the master. This register contains the data byte received on the serial input port (sin) in UART mode. The data in this register is valid only if the Data Ready (DATA_READY) bit in the Line Status Register (LINE_STAT) is set. • If FIFOs are disabled (FCR[0] set to zero), the data in the RX_BUF must be read before the next data arrives, otherwise it will be overwritten, resulting in an overrun error. • If FIFOs are enabled (FCR[0] set to one), this register accesses the head of the receive FIFO. If the receive FIFO is full and this register is not read before the next data character arrives, then the data already in the FIFO will be preserved but any incoming data will be lost. An overrun error will also occur.

4.4.1.5.13 STHR

- **Name**: Shadow Transmit Holding Register
- **Description**: This register can be accessed only when the DIV_LATCH_AB bit (LINE_CTRL[7]) is cleared.

- **Base Address**: 0x4000C500 + x*0x100
- **Offset**: 0x30
- **Reset Value**: 0x00000000

Table 4-107 Shadow Transmit Holding Register

Bits	Field Name	RW	Reset	Description
31:8	RSVD_STHRn	R		SHADOW_TX_HDG0 31 to SHADOW_TX_HDGN_REG_SIZE Reserved bits read as 0.
7:0	SHADOW_TX_HDGn	W	0x0	Shadow Transmit Holding Register n. This is a shadow register for the TX_HDG and has been allocated sixteen 32-bit locations so as to accommodate burst accesses from the master. This register contains data to be transmitted on the serial output port (sout) in UART mode. Data should only be written to the TX_HDG when the TX_HDG Empty (TX_HDG_EMPTY) bit (LINE_STAT[5]) is set. • If FIFOs are disabled (FCR[0] set to zero) and TX_HDG_EMPTY is set, writing a single character to the TX_HDG clears the TX_HDG_EMPTY. Any additional writes to the TX_HDG before the TX_HDG_EMPTY is set again causes the TX_HDG data to be overwritten. • If FIFOs are enabled (FCR[0] set to one) and TX_HDG_EMPTY is set, x number of characters of data may be written to the TX_HDG before the FIFO is full. The number x (default=16) is determined by the value of FIFO Depth that you set during configuration. Any attempt to write data when the FIFO is full results in the write data being lost.

4.4.1.5.14 FAR

- **Name**: FIFO Access Register
- **Description**: FIFO Access Register
- **Base Address**: 0x4000C500 + x*0x100
- **Offset**: 0x70
- **Reset Value**: 0x00000000

Table 4-108 FIFO Access Register

Bits	Field Name	RW	Reset	Description
31:1	RSVD	R		Reserved bits

Bits	Field Name	RW	Reset	Description
0	FIFO_ACCESS	R	0x0	This register is use to enable a FIFO access mode for testing, so that the receive FIFO can be written by the master and the transmit FIFO can be read by the master when FIFO's are implemented and enabled. When FIFOs are not implemented or not enabled, it allows the RX_BUF to be written by the master and the TX_HDG to be read by the master. **Note**: When the FIFO access mode is enabled/disabled, the control portion of the receive FIFO and transmit FIFO is reset and the FIFO's are treated as empty. **Value**: • 0x0 (DISABLED): FIFO access mode disabled • 0x1 (ENABLED): FIFO access mode enabled

4.4.1.5.15 USR

- **Name**: UART Status Register
- **Description**: UART Status register
- **Base Address**: 0x4000C500 + x*0x100
- **Offset**: 0x7C
- **Reset Value**: 0x00000000

Table 4-109 UART Status Register

Bits	Field Name	RW	Reset	Description
31:5	RSVD	R		Reserved bits
4	RX_FIFO_FULL	R	0x0	Receive FIFO Full. This is used to indicate that the receive FIFO is completely full. This bit is cleared when the RX FIFO is no longer full. **Value**: • 0x0 (NOT_FULL): Receive FIFO not full • 0x1 (FULL): Receive FIFO full
3	RX_FIFO_EMPTY	R	0x0	Receive FIFO Not Empty. This is used to indicate that the receive FIFO contains one or more entries. This bit is cleared when the RX FIFO is empty.

Bits	Field Name	RW	Reset	Description
				Value: • 0x0 (EMPTY): Receive FIFO is empty • 0x1 (NOT_EMPTY): Receive FIFO is not empty
2	TX_FIFO_EMPTY	R	0x1	Transmit FIFO Empty. This is used to indicate that the transmit FIFO is completely empty. This bit is cleared when the TX FIFO is no longer empty. **Value**: • 0x0 (NOT_EMPTY): Transmit FIFO is not empty • 0x1 (EMPTY): Transmit FIFO is empty
1	TX_FIFO_FULL	R	0x1	Transmit FIFO Not Full. This is used to indicate that the transmit FIFO in not full. This bit is cleared when the TX FIFO is full. **Value**: • 0x0 (FULL): Transmit FIFO is full • 0x1 (NOT_FULL): Transmit FIFO is not full
0	RSVD	R		Reserved bit

4.4.1.5.16 TFL

- **Name**: Transmit FIFO Level
- **Description**: Transmit FIFO Level
- **Base Address**: 0x4000C500 + x*0x100
- **Offset**: 0x80
- **Reset Value**: 0x00000000

Table 4-110 Transmit FIFO Level

Bits	Field Name	RW	Reset	Description
31:8	RSVD	R		Reserved bits
7:0	TX_FIFO_LEVEL	R	0x0	Transmit FIFO Level. This indicates the number of data entries in the transmit FIFO.

4.4.1.5.17 RFL

- **Name**: Receive FIFO Level

- **Description**: Receive FIFO Level
- **Base Address**: 0x4000C500 + x*0x100
- **Offset**: 0x84
- **Reset Value**: 0x00000000

Table 4-111 Receive FIFO Level

Bits	Field Name	RW	Reset	Description
31:8	RSVD	R		Reserved bits
7:0	RX_FIFO_LEVEL	R	0x0	Receive FIFO Level. This is indicates the number of data entries in the receive FIFO.

4.4.1.5.18 SRR

- **Name**: Software Reset Register
- **Description**: Software Reset Register
- **Base Address**: 0x4000C500 + x*0x100
- **Offset**: 0x88
- **Reset Value**: 0x00000000

Table 4-112 Software Reset Register

Bits	Field Name	RW	Reset	Description
31:3	RSVD	R		Reserved bits
2	TX_FIFO_RST	W	0x0	TX FIFO Reset. This is a shadow register for the TX FIFO Reset bit (FCR[2]). This can be used to remove the burden on software having to store previously written FIFO_CTRL values (which are pretty static) just to reset the transmit FIFO
1	RX_FIFO_RST	W	0x0	RX FIFO Reset. This is a shadow register for the RX FIFO Reset bit (FCR[1]). This can be used to remove the burden on software having to store previously written FIFO_CTRL values (which are pretty static) just to reset the receive FIFO.
0	UART_RST	W	0x0	UART Reset. This asynchronously resets the UART and synchronously removes the reset assertion. For a two-clock implementation both PCLK and SCLK domains will be reset.

Bits	Field Name	RW	Reset	Description
				Value: • 0x0 (NO_RESET): No UART Reset • 0x1 (RESET): UART reset

4.4.1.5.19 SRTS

- **Name**: Shadow Request to Send Register
- **Description**: Shadow Request to Send
- **Base Address**: 0x4000C500 + x*0x100
- **Offset**: 0x8C
- **Reset Value**: 0x00000000

Table 4-113 Shadow Request to Send

Bits	Field Name	RW	Reset	Description
31:1	RSVD	R		Reserved bits
0	SHADOW_REQ_SEND	RW	0x0	Shadow Request to Send. This is a shadow register for the REQ_TO_SEND bit (MODEM_CTRL [1]), this can be used to remove the burden of having to performing a read modify write on the MODEM_CTRL. This is used to directly control the Request to Send (rts_n) output. The Request To Send (rts_n) output is used to inform the modem or data set that the UART is ready to exchange data. **Value**: • 0x0 (DEASSERTED): Shadow Request to Send uart_rts_n logic 1 • 0x1 (ASSERTED): Shadow Request to Send uart_rts_n logic 0

4.4.1.5.20 SBCR

- **Name**: Shadow Break Control Register
- **Description**: Shadow Break Control Register
- **Base Address**: 0x4000C500 + x*0x100
- **Offset**: 0x90
- **Reset Value**: 0x00000000

Table 4-114 Shadow Break Control Register

Bits	Field Name	RW	Reset	Description
31:1	RSVD	R		Reserved bits
0	SHADOW_BREAK_CB	RW	0x0	Shadow Break Control Bit. This is a shadow register for the Break bit (LCR[6]). This can be used to remove the burden of having to performing a read modify write on the LINE_CTRL. This is used to cause a break condition to be transmitted to the receiving device. If set to one the serial output is forced to the spacing (logic 0) state. When not in Loopback Mode, as determined by MODEM_CTRL [4], the sout line is forced to be low until the Break bit is cleared. **Value**: • 0x0 (NO_BREAK): No spacing on serial output • 0x1 (BREAK): Serial output forced to the spacing

4.4.1.5.21 SFE

- **Name**: Shadow FIFO Enable Register
- **Description**: Shadow FIFO Enable Register
- **Base Address**: 0x4000C500 + x*0x100
- **Offset**: 0x98
- **Reset Value**: 0x00000000

Table 4-115 Shadow FIFO Enable Register

Bits	Field Name	RW	Reset	Description
31:1	RSVD	R		Reserved bits
0	SHADOW_FIFO_EN	RW	0x0	Shadow FIFO Enable. This is a shadow register for the FIFO enable bit (FCR[0]). This can be used to remove the burden of having to store the previously written value to the FIFO_CTRL in memory and having to mask this value so that only the FIFO enable bit gets updated. This enables/disables the transmit (TX) and receive (RX) FIFO's. If this bit is set to zero (disabled) after being enabled then both the TX and RX controller portions of FIFO's will be reset.

Bits	Field Name	RW	Reset	Description
				Value: • 0x0 (DISABLED): FIFOs are disabled • 0x1 (ENABLED): FIFOs are enabled

4.4.1.5.22 SRT

- **Name**: Shadow RX Trigger Register
- **Description**: Shadow RX Trigger Register
- **Base Address**: 0x4000C500 + x*0x100
- **Offset**: 0x9C
- **Reset Value**: 0x00000000

Table 4-116 Shadow RCVR Trigger Register

Bits	Field Name	RW	Reset	Description
31:2	RSVD	R		Reserved bits
1:0	SHADOW_RX_TRG	RW	0x0	Shadow RX Trigger. This is a shadow register for the RX trigger bits (FCR[7:6]). This can be used to remove the burden of having to store the previously written value to the FIFO_CTRL in memory and having to mask this value so that only the RX trigger bit gets updated. **Value**: • 0x0 (FIFO_CHAR_1): 1 character in FIFO • 0x1 (FIFO_QUARTER_FULL): FIFO 1/4 full • 0x2 (FIFO_HALF_FULL): FIFO 1/2 full • 0x3 (FIFO_FULL_2): FIFO 2 less than full

4.4.1.5.23 STET

- **Name**: Shadow TX Empty Trigger Register
- **Description**: Shadow TX Empty Trigger Register
- **Base Address**: 0x4000C500 + x*0x100
- **Offset**: 0xA0
- **Reset Value**: 0x00000000

Table 4-117 Shadow TX Empty Trigger Register

Bits	Field Name	RW	Reset	Description
31:2	RSVD	R		Reserved bits
1:0	SHADOW_TX_TRG	RW	0x0	Shadow TX Empty Trigger. This is a shadow register for the TX empty trigger bits (FCR[5:4]). This can be used to remove the burden of having to store the previously written value to the FIFO_CTRL in memory and having to mask this value so that only the TX empty trigger bit gets updated. **Value**: • 0x0 (FIFO_EMPTY): FIFO empty • 0x1 (FIFO_CHAR_2): 2 characters in FIFO • 0x2 (FIFO_QUARTER_FULL): FIFO 1/4 full • 0x3 (FIFO_HALF_FULL): FIFO 1/2 full

4.4.1.5.24 DLF

- **Name**: Divisor Latch Fraction Register
- **Description**: Divisor Latch Fraction Register
- **Base Address**: 0x4000C500 + x*0x100
- **Offset**: 0xC0
- **Reset Value**: 0x00000000

Table 4-118 Divisor Latch Fraction Register

Bits	Field Name	RW	Reset	Description
31:4	RSVD	R		Reserved bits
3:0	DIV_LATCH_FRACTION	RW	0x0	Fractional part of divisor. The fractional value is added to integer value set by DIV_LATCH_HIGH and DIV_LATCH_LOW. Fractional value is determined by (Divisor Fraction value)/(2^DLF_SIZE).

4.4.1.6 Electrical Specifications

Table 4-119 Baud rate with 96 MHz Serial Clock

Baud Rate	Divider	DLH Reg	DLL Reg	DLF Reg	Actual Baud Rate	Error (%)
1200	5000	19	136	0	1200	0

Baud Rate	Divider	DLH Reg	DLL Reg	DLF Reg	Actual Baud Rate	Error (%)
2400	2500	9	196	0	2400	0
4800	1250	4	226	0	4800	0
9600	626	2	113	0	9600	0
14400	416.67	1	160	11	14399.28	0
19200	312.50	1	56	8	19200	0
28800	208.33	0	208	5	28802.88	0.01
38400	156.25	0	156	4	38400	0
57600	104.17	0	104	3	57588.48	0.02
115200	52.08	0	52	1	115246.10	0.04
230400	26.04	0	26	4	230215.83	0.08
460800	13.021	0	13	0	461538.46	0.16
921600	6.51	0	6	8	923076.92	0.16
1000000	6	0	6	0	1000000	0
2000000	3	0	3	0	2000000	0

4.4.2 SPI

4.4.2.1 Introduction

The device has two Serial Peripheral Interface (SPI) instances.

- One SPI Master (SPIM) with two slave select lines that can interface to up to two slaves that share the same lines with separate selects.
- One SPI Slave (SPIS) that can connect to a single master.

4.4.2.2 Main Features

Main features of the SPIM/SPIS modules include:

- Motorola SPI support
- Two slave select lines for master mode to support up to two slaves.
- Built-in 16 word RX/TX FIFOs for continuous SPI bursts.
- Transfer data size up to 32 bits.
- Programmable output interface frequency up to 48 MHz for SPIM and 16 MHz for SPIS.
- Supports SPI four modes for different clock edge and phase configurations.
- DMA support
- Maskable interrupt generation

4.4.2.3 Block Diagram

Figure 4-17 shows the SPI Controller with the following functions and interfaces:

- APB interface and DMA Controller Interface
- Transmit and receive FIFO controllers and an FSM controller
- Register block
- Shift control and interrupt logic

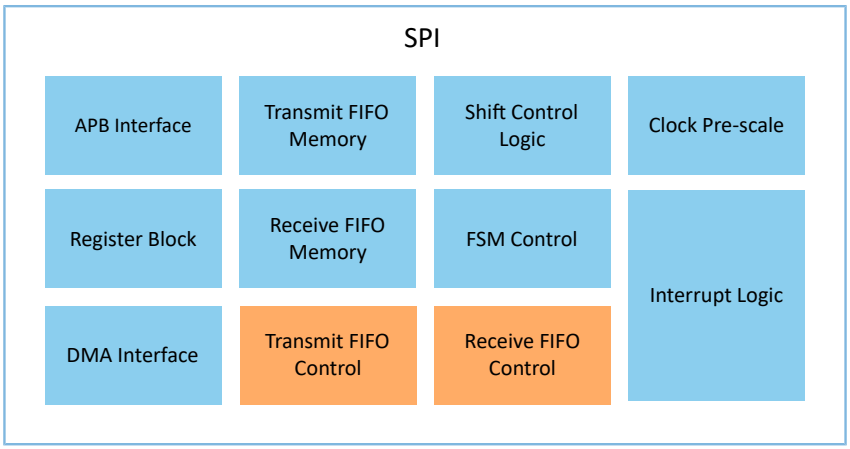

Figure 4-17 SPI block diagram

4.4.2.4 Functional Description

4.4.2.4.1 Motorola Serial Peripheral Interface

With the SPI, the clock polarity (SERIAL_CLK_POL) configuration parameter determines whether the inactive state of the serial clock is high or low. To transmit data, both SPI peripherals must have identical serial clock phase (SERIAL_CLK_PHASE, controlled by CTRL0[6]) and clock polarity (SERIAL_CLK_POL, controlled by CTRL0[7]) values. The data frame can be 4 bits to 32 bits.

When the configuration parameter SERIAL_CLK_PHASE = 0, data transmission begins on the falling edge of the slave select signal. The first data bit is captured by the master and slave peripherals on the first edge of the serial clock; therefore, valid data must be present on the txd and rxd lines prior to the first serial clock edge.

Figure 4-18 shows a timing diagram for a single SPI data transfer with SERIAL_CLK_PHASE = 0 (CTRL0[6] = 0). The serial clock is shown for configuration parameters SERIAL_CLK_POL = 0 (CTRL0[7] = 0) and SERIAL_CLK_POL = 1 (CTRL0[7] = 1).

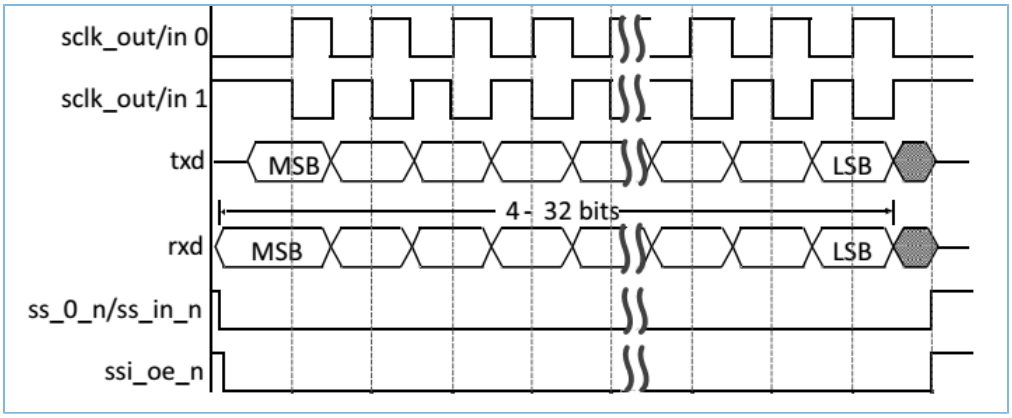

Figure 4-18 SPI serial format (SERIAL_CLK_PHASE = 0)

The following signals are illustrated in the timing diagrams in this section:

- sclk_out – serial clock from SPI block master (master configuration only)
- sclk_in – serial clock from SPI block slave (slave configuration only)
- txd – transmit data line for the SPI block master/slave
- rxd – receive data line for the SPI block master/slave
- ss_0_n – slave select signal from SPI block master (master configuration only)
- ss_in_n – slave select input to the SPI block slave
- ss_oe_n – output enable for the SPI block master/slave

Two different modes of continuous data transfers are supported when SERIAL_CLK_PHASE = 0; the selection of the desired operation mode is done by configuring bit24 (S_ST_EN) of the CTRL0 register:

- When CTRL0[24] is programmed to 1, SPI block toggles the slave select signal between frames, and the serial clock is held to its default value while the slave select signal is active; this operating mode is illustrated in the figure below.

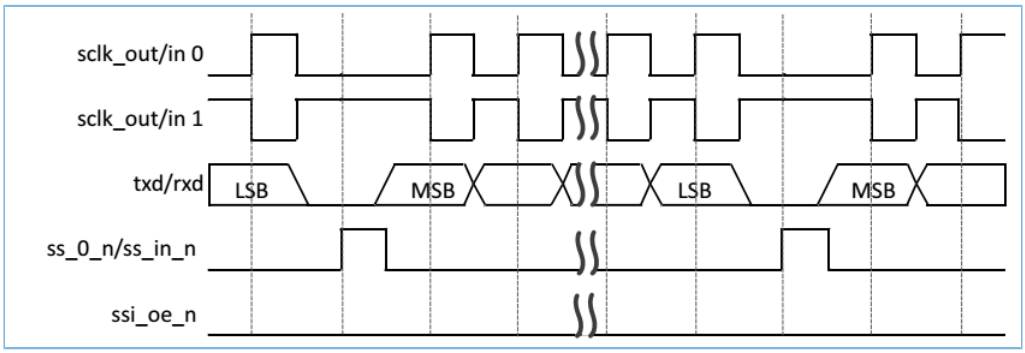

Figure 4-19 Serial format continuous transfers (SERIAL_CLK_PHASE = 0) when CTRL0[24] = 1

- When CTRL0[24] is programmed to 0, the slave select signal stays low and the serial clock runs continuously for the duration of the transfer; this operating mode is illustrated below.

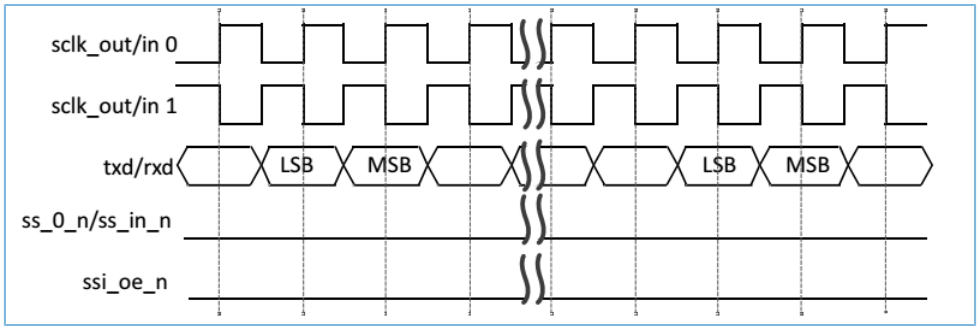

Figure 4-20 Serial format continuous transfers (SERIAL_CLK_PHASE = 0) when CTRL0[24] = 0

- When the configuration parameter SERIAL_CLK_PHASE = 1 (CTRL0[6] = 1), both master and slave peripherals begin transmitting data on the first serial clock edge after the slave select line is activated. The first data bit is captured on the second (trailing) serial clock edge. Data is propagated by the master and slave peripherals on the leading edge of the serial clock. During continuous data frame transfers, the slave select line may be held active-low until the last bit of the last frame has been captured.

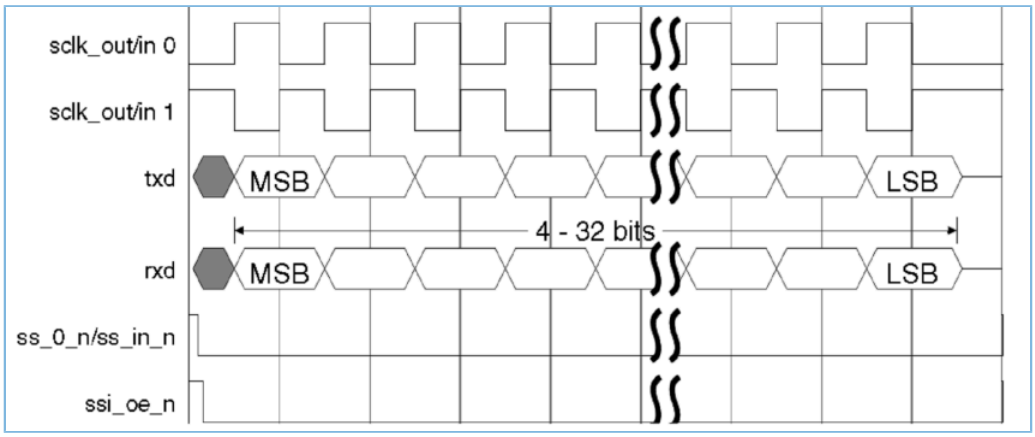

Figure 4-21 SPI serial format (SERIAL_CLK_PHASE = 1)

4.4.2.4.2 Transmit and Receive FIFO Buffers

The width of both transmit and receive FIFO buffers is fixed at 32 bits. Data frames that are less than 32 bits must be right-justified when written into the transmit FIFO buffer. The shift control logic automatically right-justifies received data in the receive FIFO buffer.

Each data entry in the FIFO buffers contains a single data frame. It is not possible to store multiple data frames in a single FIFO location; for example, users may not store two 8-bit data frames in a single FIFO location. If an 8-bit data frame is required, the upper 8-bits of the FIFO entry are ignored or unused when the serial shifter transmits the data.

Transmit FIFO

The transmit FIFO is loaded by writing to the SPI Controller data register (DATA). Data is popped from the transmit FIFO by the shift control logic into the transmit shift register. The transmit FIFO generates a FIFO empty interrupt request (ssi_txe_intr) when the number of entries in the FIFO is less than or equal to the FIFO threshold value. The threshold value, set through the programmable transmit FIFO threshold level register (TX_FIFO_TL[2:0]), determines the level of FIFO entries at which an interrupt is generated. The threshold value allows users to provide early indication to the

processor that the transmit FIFO is nearly empty. A transmit FIFO overflow interrupt (ssi_txo_intr) is generated if users attempt to write data into an already full transmit FIFO.

Receive FIFO

Data is popped from the receive FIFO by reading the SPI Controller data register (DATA). The receive FIFO is loaded from the receive shift register by the shift control logic. The receive FIFO generates a FIFO-full interrupt request (ssi_rxf_intr) when the number of entries in the FIFO is greater than or equal to the FIFO threshold value plus 1. The threshold value, set through programmable receive FIFO threshold level register (RX_FIFO_TL[2:0]), determines the level of FIFO entries at which an interrupt is generated.

The threshold value allows users to provide early indication to the processor that the receive FIFO is nearly full. A receive FIFO overrun interrupt (ssi_rxo_intr) is generated when the receive shift logic attempts to load data into a completely full receive FIFO. However, this newly received data is lost. A receive FIFO underflow interrupt (ssi_rxu_intr) is generated if users attempt to read from an empty receive FIFO. This alerts the processor that the read data is invalid.

4.4.2.4.3 SPI Interrupts

The SPI Controller supports combined and individual interrupt requests, each of which can be masked. The combined interrupt request is the OR'ed result of all other SPI Controller interrupts after masking. The system designer has the choice of routing individual interrupt requests or only the combined interrupt request to the Interrupt Controller. All SPI Controller interrupts are level interrupts that share the same active polarity level; users can configure this polarity level as active-high or active-low.

The SPI Controller interrupts are described as follows:

- Transmit FIFO Empty Interrupt (ssi_txe_intr) – Set when the transmit FIFO is equal to or below its threshold value and requires service to prevent an under-run. The threshold value, set through a software-programmable register, determines the level of transmit FIFO entries at which an interrupt is generated. This interrupt is cleared by hardware when data is written into the transmit FIFO buffer, bringing it over the threshold level.
- Transmit FIFO Overflow Interrupt (ssi_txo_intr) – Set when an APB access attempts to write into the transmit FIFO after it has been completely filled. When set, data written from the APB is discarded. This interrupt remains set until users read the transmit FIFO overflow interrupt clear register (TX_FIFO_OIC[0]).
- Receive FIFO Full Interrupt (ssi_rxf_intr) – Set when the receive FIFO is equal to or above its threshold value plus 1 and requires service to prevent an overflow. The threshold value, set through a software-programmable register, determines the level of receive FIFO entries at which an interrupt is generated. This interrupt is cleared by hardware when data is read from the receive FIFO buffer, bringing it below the threshold level.
- Receive FIFO Overflow Interrupt (ssi_rxo_intr) – Set when the receive logic attempts to place data into the receive FIFO after it has been completely filled. When set, newly received data is discarded. This interrupt remains set until users read the receive FIFO overflow interrupt clear register (RX_FIFO_OIC[0]).
- Receive FIFO Underflow Interrupt (ssi_rxu_intr) – Set when an APB access attempts to read from the receive FIFO when it is empty. When set, zeros are read back from the receive FIFO. This interrupt remains set until users read the receive FIFO underflow interrupt clear register (RX_FIFO_UIC[0]).

4.4.2.4.4 Transfer Modes

When transferring data on the serial bus, the SPI Controller operates in the modes discussed in this section. The transfer mode (XFE_MODE) is set by writing to bit8 and bit9 of the SPI control register 0 (CTRL0).

Note:

The transfer mode setting does not affect the duplex of the serial transfer.

4.4.2.4.4.1 Transmit and Receive

When CTRL0[9:8] = 0x0 (XFE_MODE = 2‘b00), both transmit and receive logics are valid. The data transfer occurs as normal according to the selected frame format (serial protocol). Transmit data is popped from the transmit FIFO and sent through the txd (tx data) line to the target device, which replies with data on the rxd (rx data) line. The receive data from the target device is moved from the receive shift register into the receive FIFO at the end of each data frame.

4.4.2.4.4.2 Transmit Only

When CTRL0[9:8] = 0x1 (XFE_MODE = 2‘b01), the receive data is invalid and should not be stored in the receive FIFO. The data transfer occurs as normal, according to the selected frame format (serial protocol). Transmit data is popped from the transmit FIFO and sent through the txd line to the target device, which replies with data on the rxd line. At the end of the data frame, the receive shift register does not load its newly received data into the receive FIFO. The data in the receive shift register is overwritten by the next transfer. Users should mask interrupts originating from the receive logic when this mode is entered.

4.4.2.4.4.3 Receive Only

When CTRL0[9:8] = 0x2 (XFE_MODE = 2‘b10), the transmit data is invalid. When configured as a slave, the transmit FIFO is never popped in Receive Only mode. The txd output remains at a constant logic level during the transmission. The data transfer occurs as normal according to the selected frame format (serial protocol). The receive data from the target device is moved from the receive shift register into the receive FIFO at the end of each data frame. Users should mask interrupts originating from the transmit logic when this mode is entered.

4.4.2.4.4.4 EEPROM Read

Note:

This transfer mode is only valid for master configurations.

When CTRL0[9:8] = 0x3 (XFE_MODE = 2‘b11), the transmit data is used to transmit an opcode and/or an address to the EEPROM device. Typically, this takes three data frames (8-bit opcode followed by 8-bit upper address and 8-bit lower address). During the transmission of the opcode and address, no data is captured by the receive logic (as long as the SPI Controller master is transmitting data on its txd line, data on the rxd line is ignored). The SPI Controller master continues to transmit data until the transmit FIFO is empty. Therefore, users should only have enough data frames in the transmit FIFO to supply the opcode and address to the EEPROM. If more data frames are in the transmit FIFO than needed, then read data is lost.

When the transmit FIFO becomes empty (all control information has been sent), data on the receive line (rxd) is valid and is stored in the receive FIFO; the txd output is held at a constant logic level. The serial transfer continues until the number of data frames received by the SPI Controller master matches the value of the NUM_DATA_FRAME (Number of Data Frames) field in the SPI Control register 1 (CTRL1[15:0]) plus 1.

4.4.2.4.5 SPI Master Operation

Data transfers are started by the serial-master device. When the SPI Controller is enabled (SPI Enable Register, SSI_EN[0]=1), at least one valid data entry is present in the transmit FIFO and a serial-slave device is selected. When actively transferring data, the busy flag (BUSY) in the status register (STAT[0]) is set. Users must wait until the busy flag is cleared before attempting a new serial transfer.

When the transfer mode is “transmit and receive” or “transmit only” (CTRL0[9:8] = 0x0 or CTRL0[9:8] = 0x1, respectively), transfers are terminated by the shift control logic when the transmit FIFO is empty. For continuous data transfers, users must ensure that the transmit FIFO buffer does not become empty before all the data has been transmitted. The transmit FIFO threshold level (TX_FIFO_TL[2:0]) can be used to early interrupt (ssi_txe_intr) the processor, indicating that the transmit FIFO buffer is nearly empty. When a DMA is used for APB accesses, the DMA transmit data level (DMA_TX_DL[2:0]) can be used to early request (dma_tx_req) the DMA Controller, indicating that the transmit FIFO is nearly empty. The FIFO can then be refilled with data to continue the serial transfer. Users may also write a block of data (at least two FIFO entries) into the transmit FIFO before enabling a serial slave. This ensures that serial transmission does not begin until the number of data frames that make up the continuous transfer are present in the transmit FIFO.

When the transfer mode is “receive only” (CTRL0[9:8] = 0x2), a serial transfer is started by writing one “dummy” data word into the transmit FIFO when a serial slave is selected. The TXD output from the SPI Controller is held at a constant logic level for the duration of the serial transfer. The transmit FIFO is popped only once at the beginning and may remain empty for the duration of the serial transfer. The end of the serial transfer is controlled by the “number of data frames” (NUM_DATA_FRAME) field in control register 1 (CTRL1[15:0]).

If, for example, users want to receive 24 data frames from a serial-slave peripheral, you should program the NUM_DATA_FRAME field with the value 23; the receive logic terminates the serial transfer when the number of frames received is equal to the NUM_DATA_FRAME value + 1. This transfer mode increases the bandwidth of the APB bus as the transmit FIFO never needs to be serviced during the transfer. The receive FIFO buffer should be read each time the receive FIFO generates a FIFO full interrupt request to prevent an overflow.

When the transfer mode is “eeprom_read” (CTRL0[9:8] = 0x3), a serial transfer is started by writing the opcode and/or address into the transmit FIFO when a serial slave (EEPROM) is selected. The opcode and address are transmitted to the EEPROM device, after which read data is received from the EEPROM device and stored in the receive FIFO. The end of the serial transfer is controlled by the NUM_DATA_FRAME field in the control register 1 (CTRL1[15:0]).

The receive FIFO threshold level (RX_FIFO_TL[2:0]) can be used to give early indication that the receive FIFO is nearly full. When a DMA is used for APB accesses, the DMA receive data level (DMA_RX_DL[2:0]) can be used to early request (dma_rx_req) the DMA Controller, indicating that the receive FIFO is nearly full.

4.4.2.4.6 SPI Slave Operation

If the SPI Controller slave is receive only (CTRL0[9:8] = 0x2), the transmit FIFO does not need to contain valid data because the data currently in the transmit shift register is resent each time the slave device is selected. The TX_ERR

error flag in the status register (STAT[5]) is not set when CTRL0[9:8] = 0x1. Users should mask the transmit FIFO empty interrupt when this mode is used.

If the SPI Controller slave transmits data to the master, users must ensure that data exists in the transmit FIFO before a transfer is initiated by the serial-master device. If the master initiates a transfer to the SPI Controller slave when no data exists in the transmit FIFO, an error flag (TX_ERR) is set in the status register (STAT[5]), and the previously transmitted data frame is resent on txd. For continuous data transfers, users must ensure that the transmit FIFO buffer does not become empty before all the data has been transmitted. The transmit FIFO threshold level register (TX_FIFO_TL[2:0]) can be used to early interrupt (ssi_txe_intr) the processor, indicating that the transmit FIFO buffer is nearly empty. When a DMA Controller is used for APB accesses, the DMA transmit data level register (DMA_TX_DL[2:0]) can be used to early request (dma_tx_req) the DMA Controller, indicating that the transmit FIFO is nearly empty. The FIFO can then be refilled with data to continue the serial transfer. The receive FIFO buffer should be read each time the receive FIFO generates a FIFO full interrupt request to prevent an overflow. The receive FIFO threshold level register (RX_FIFO_TL[2:0]) can be used to give early indication that the receive FIFO is nearly full. When a DMA Controller is used for APB accesses, the DMA receive data level register (DMA_RX_DL[2:0]) can be used to early request (dma_rx_req) the DMA controller, indicating that the receive FIFO is nearly full.

4.4.2.5 Registers

4.4.2.5.1 CTRL0

- **Name:** SPI Control Register 0
- **Description:** This register controls the serial data transfer. It is impossible to write to this register when the SPI is enabled.
- **Base Address:** 0x4000C000 + x*0x100 (x: 0 for master, 1 for slave)
- **Offset:** 0x00
- **Reset Value:** 0x01070000

Table 4-120 Control Register 0

Bits	Field Name	RW	Reset	Description
31:25	RSVD	R		Reserved bits.
24	S_ST_EN	RW	0x1	Slave Select Toggle Enable. When operating in SPI mode with clock phase (SCPH) set to 0, this register controls the behavior of the slave select line (CS) between data frames. If this register field is set to 1 the CS line will toggle between consecutive data frames, with the serial clock (sclk) being held to its default value while CS is high; if this register field is set to 0 the CS will stay low and sclk will run continuously for the duration of the transfer. **Note:**

Bits	Field Name	RW	Reset	Description
				When the SPI is configured as a slave, this register serves no purpose.
23:21	RSVD	R		Reserved bits.
20:16	DATA_FRAME_SIZE	RW	0x7	Data Frame Size in 32-bit transfer size mode. Used to select the data frame size in 32-bit transfer mode. When the data frame size is programmed to be less than 32 bits, the receive data is automatically right-justified by the receive logic, with the upper bits of the receive FIFO zero-padded. You are responsible for making sure that transmit data is right-justified before writing into the transmit FIFO. The transmit logic ignores the upper unused bits when transmitting the data. **Value:** • 0x3 (FRAME_04BITS): 4-bit serial data transfer • 0x4 (FRAME_05BITS): 5-bit serial data transfer • 0x5 (FRAME_06BITS): 6-bit serial data transfer • 0x6 (FRAME_07BITS): 7-bit serial data transfer • 0x7 (FRAME_08BITS): 8-bit serial data transfer • 0x8 (FRAME_09BITS): 9-bit serial data transfer • 0x9 (FRAME_10BITS): 10-bit serial data transfer • 0xa (FRAME_11BITS): 11-bit serial data transfer • 0xb (FRAME_12BITS): 12-bit serial data transfer • 0xc (FRAME_13BITS): 13-bit serial data transfer • 0xd (FRAME_14BITS): 14-bit serial data transfer • 0xe (FRAME_15BITS): 15-bit serial data transfer • 0xf (FRAME_16BITS): 16-bit serial data transfer • 0x10 (FRAME_17BITS): 17-bit serial data transfer • 0x11 (FRAME_18BITS): 18-bit serial data transfer • 0x12 (FRAME_19BITS): 19-bit serial data transfer • 0x13 (FRAME_20BITS): 20-bit serial data transfer • 0x14 (FRAME_21BITS): 21-bit serial data transfer • 0x15 (FRAME_22BITS): 22-bit serial data transfer • 0x16 (FRAME_23BITS): 23-bit serial data transfer • 0x17 (FRAME_24BITS): 24-bit serial data transfer

Bits	Field Name	RW	Reset	Description
				• 0x18 (FRAME_25BITS): 25-bit serial data transfer • 0x19 (FRAME_26BITS): 26-bit serial data transfer • 0x1a (FRAME_27BITS): 27-bit serial data transfer • 0x1b (FRAME_28BITS): 28-bit serial data transfer • 0x1c (FRAME_29BITS): 29-bit serial data transfer • 0x1d (FRAME_30BITS): 30-bit serial data transfer • 0x1e (FRAME_31BITS): 31-bit serial data transfer • 0x1f (FRAME_32BITS): 32-bit serial data transfer
15:12	CTRL_FRAME_SIZE	RW	0x0	Control Frame Size. Selects the length of the control word for the Microwire frame format. **Value:** • 0x0 (SIZE_01_BIT): 1-bit Control Word • 0x1 (SIZE_02_BIT): 2-bit Control Word • 0x2 (SIZE_03_BIT): 3-bit Control Word • 0x3 (SIZE_04_BIT): 4-bit Control Word • 0x4 (SIZE_05_BIT): 5-bit Control Word • 0x5 (SIZE_06_BIT): 6-bit Control Word • 0x6 (SIZE_07_BIT): 7-bit Control Word • 0x7 (SIZE_08_BIT): 8-bit Control Word • 0x8 (SIZE_09_BIT): 9-bit Control Word • 0x9 (SIZE_10_BIT): 10-bit Control Word • 0xa (SIZE_11_BIT): 11-bit Control Word • 0xb (SIZE_12_BIT): 12-bit Control Word • 0xc (SIZE_13_BIT): 13-bit Control Word • 0xd (SIZE_14_BIT): 14-bit Control Word • 0xe (SIZE_15_BIT): 15-bit Control Word • 0xf (SIZE_16_BIT): 16-bit Control Word
11	SHIFT_REG_LOOP	RW	0x0	Shift Register Loop. Used for testing purposes only. When internally active, connects the transmit shift register output to the receive shift register input. **Value:** • 0x0 (NORMAL_MODE): Normal mode operation

Bits	Field Name	RW	Reset	Description
				• 0x1 (TESTING_MODE): Test mode: TX & RX shift register connected
10	S_OUT_EN	RW	0x0	Slave Output Enable. Relevant only when the SPI is configured as a serial-slave device. When configured as a serial master, this bit field has no functionality. **Value:** • 0x0 (ENABLED): Slave Output is enabled • 0x1 (DISABLED): Slave Output is disabled **Note:** When the SPI is configured as a master, this register serves no purpose.
9:8	XFE_MODE	RW	0x0	Transfer Mode. This transfer mode is only valid when the SPI is configured as master device. 0x0 - Transmit & Receive 0x1 - Transmit Only 0x2 - Receive Only 0x3 - EEPROM Read **Value:** • 0x0 (TX_AND_RX): Transmit & receive • 0x1 (TX_ONLY): Transmit only mode • 0x2 (RX_ONLY): Receive only mode • 0x3 (EEPROM_READ): EEPROM Read mode
7	SERIAL_CLK_POL	RW	0x0	Serial Clock Polarity. Valid when the frame format (FRF) is set to Motorola SPI. Used to select the polarity of the inactive serial clock, which is held inactive when the SPI master is not actively transferring data on the serial bus. **Value:** • 0x0 (SCLK_LOW): Inactive state of serial clock is low • 0x1 (SCLK_HIGH): Inactive state of serial clock is high
6	SERIAL_CLK_PHASE	RW	0x0	Serial Clock Phase.

Bits	Field Name	RW	Reset	Description
				Valid when the frame format (FRF) is set to Motorola SPI. The serial clock phase selects the relationship of the serial clock with the slave select signal. When SCPH = 0, data is captured on the first edge of the serial clock. When SCPH = 1, the serial clock starts toggling one cycle after the slave select line is activated, and data is captured on the second edge of the serial clock. **Value:** • 0x0 (SCPH_MIDDLE): Serial clock toggles in middle of first data bit • 0x1 (SCPH_START): Serial clock toggles at start of first data bit
5:4	FRAME_FORMAT	RW	0x0	Frame Format. Selects which serial protocol transfers the data. **Value:** • 0x0 (MOTOROLA_SPI): Motorola SPI Frame Format • 0x1 (TEXAS_SSP): Texas Instruments SSP Frame Format • 0x2 (NS_MICROWIRE): National Microwire Frame Format • 0x3 (RESERVED): Reserved value
3:0	RSVD	R		Reserved bits.

4.4.2.5.2 CTRL1

- **Name:** SPI Control Register 1
- **Description:** This register exists only when the SPI is configured as a master device. Control register 1 controls the end of serial transfers in receive-only mode. It is impossible to write to this register when the SPI is enabled.
- **Base Address:** 0x4000C000 + x*0x100 (x: 0 for master, 1 for slave)
- **Offset:** 0x04
- **Reset Value:** 0x00000000

Table 4-121 Control Register 1

Bits	Field Name	RW	Reset	Description
31:16	RSVD	R		Reserved bits.
15:0	NUM_DATA_FRAME	RW	0x0	Number of Data Frames. When CTRL0[9:8] = 0x2 or CTRL0[9:8] = 0x3, this register field sets the number of data frames to be continuously

Bits	Field Name	RW	Reset	Description
				received by the SPI. The SPI continues to receive serial data until the number of data frames received is equal to this register value plus 1, which enables you to receive up to 64 KB of data in a continuous transfer. **Note:** When the SPI is configured as a slave, this register serves no purpose.

4.4.2.5.3 SSI_EN

- **Name:** SPI Enable Register
- **Description:** This register enables and disables the SPI.
- **Base Address:** 0x4000C000 + x*0x100 (x: 0 for master, 1 for slave)
- **Offset:** 0x08
- **Reset Value:** 0x00000000

Table 4-122 SSI Enable Register

Bits	Field Name	RW	Reset	Description
31:1	RSVD	R		Reserved bits.
0	SSI_EN	RW	0x0	SSI Enable. Enables and disables all SPI operations. When disabled, all serial transfers are halted immediately. Transmit and receive FIFO buffers are cleared when the device is disabled. **Value:** • 0x0 (DISABLE): Disables Serial Transfer • 0x1 (ENABLED): Enables Serial Transfer

4.4.2.5.4 MW_CTRL

- **Name:** Microwire Control Register
- **Description:** This register controls the direction of the data word for the half-duplex Microwire serial protocol. It is impossible to write to this register when the SPI is enabled.
- **Base Address:** 0x4000C000 + x*0x100 (x: 0 for master, 1 for slave)
- **Offset:** 0x0C
- **Reset Value:** 0x00000000

Table 4-123 Microwire Control Register

Bits	Field Name	RW	Reset	Description
31:3	RSVD	R		Reserved bits.
2	MW_HSG	RW	0x0	Microwire Handshaking. Used to enable and disable the busy/ready handshaking interface for the Microwire protocol. When enabled, the SPI checks for a ready status from the target slave, after the transfer of the last data/control bit, before clearing the BUSY status in the STAT register. **Value:** • 0x0 (DISABLE): Handshaking interface is disabled • 0x1 (ENABLED): Handshaking interface is enabled **Note:** When the SPI is configured as a slave, this register serves no purpose.
1	MW_DIR_DW	RW	0x0	Microwire Control. Defines the direction of the data word when the Microwire serial protocol is used. **Value:** • 0x0 (RECEIVE): SPI receives data • 0x1 (TRANSMIT): SPI transmits data
0	MW_XFE_MODE	RW	0x0	Microwire Transfer Mode. Defines whether the Microwire transfer is sequential or non-sequential. When sequential mode is used, only one control word is needed to transmit or receive a block of data words. When non-sequential mode is used, there must be a control word for each data word that is transmitted or received. **Value:** • 0x0 (NON_SEQUENTIAL): Non-Sequential Microwire Transfer • 0x1 (SEQUENTIAL): Sequential Microwire Transfer

4.4.2.5.5 S_EN

- **Name:** Slave Enable Register

- **Description:** This register exists only when the SPI is configured as a master device. The register enables the individual slave select output lines from the SPI master.
- **Base Address:** 0x4000C000 + x*0x100 (x: 0 for master, 1 for slave)
- **Offset:** 0x10
- **Reset Value:** 0x00000000

Table 4-124 Slave Enable Register

Bits	Field Name	RW	Reset	Description
31:2	RSVD	R		Reserved bits.
1:0	SEL_EN	RW	0x0	Slave Select Enable. Each bit in this register corresponds to a slave select line (CSn) from the SPI master. **Value:** • 0x0 (NOT_SELECTED): No slave selected • 0x1 (SELECTED): Slave is selected **Note:** When the SPI is configured as a slave, this register serves no purpose.

4.4.2.5.6 BAUD

- **Name:** Baud Rate Register
- **Description:** This register exists only when the SPI is configured as a master device. The register derives the frequency of the serial clock that regulates the data transfer. It is impossible to write to this register when the SPI is enabled.
- **Base Address:** 0x4000C000 + x*0x100 (x: 0 for master, 1 for slave)
- **Offset:** 0x14
- **Reset Value:** 0x00000000

Table 4-125 Baud Rate Register

Bits	Field Name	RW	Reset	Description
31:16	RSVD	R		Reserved bits.
15:0	CLK_DIV	RW	0x0	SPI Clock Divider. The LSB for this field is always set to 0 and is unaffected by a write operation, which ensures an even value is held in this register. If the value is 0, the serial output clock (sclk_out) is

Bits	Field Name	RW	Reset	Description
				disabled. The frequency of the sclk_out is derived from the following equation: Fsclk_out = Fssi_clk/SCKDV **Note:** When the SPI is configured as a slave, this register serves no purpose.

4.4.2.5.7 TX_FIFO_TL

- **Name:** Transmit FIFO Threshold Level Register
- **Description:** This register controls the threshold value for the transmit FIFO memory.
- **Base Address:** 0x4000C000 + x*0x100 (x: 0 for master, 1 for slave)
- **Offset:** 0x18
- **Reset Value:** 0x00000000

Table 4-126 Transmit FIFO Threshold Level

Bits	Field Name	RW	Reset	Description
31:3	RSVD	R		Reserved bits.
2:0	TX_FIFO_THD	RW	0x0	Transmit FIFO Threshold. Controls the level of entries at (or below) which the transmit FIFO controller triggers an interrupt. The FIFO depth is configurable in 16 words; this register is sized to the number of address bits needed to access the FIFO. If users attempt to set this value greater than or equal to the depth of the FIFO, this field is not written and retains its current value. When the number of transmit FIFO entries is less than or equal to this value, the transmit FIFO empty interrupt is triggered.

4.4.2.5.8 RX_FIFO_TL

- **Name:** Receive FIFO Threshold Level
- **Description:** This register controls the threshold value for the receive FIFO memory.
- **Base Address:** 0x4000C000 + x*0x100 (x: 0 for master, 1 for slave)
- **Offset:** 0x1C
- **Reset Value:** 0x00000000

Table 4-127 Receive FIFO Threshold Level

Bits	Field Name	RW	Reset	Description
31:3	RSVD	R		Reserved bits.
2:0	RX_FIFO_THD	RW	0x0	Receive FIFO Threshold. Controls the level of entries at (or above) which the receive FIFO controller triggers an interrupt. The FIFO depth is configurable in 16 words. This register is sized to the number of address bits needed to access the FIFO. If users attempt to set this value greater than the depth of the FIFO, this field is not written and retains its current value. When the number of receive FIFO entries is greater than or equal to this value + 1, the receive FIFO full interrupt is triggered.

4.4.2.5.9 TX_FIFO_LV

- **Name:** Transmit FIFO Level Register
- **Description:** This register contains the number of valid data entries in the transmit FIFO memory.
- **Base Address:** 0x4000C000 + x*0x100 (x: 0 for master, 1 for slave)
- **Offset:** 0x20
- **Reset Value:** 0x00000000

Table 4-128 Transmit FIFO Level Register

Bits	Field Name	RW	Reset	Description
31:4	RSVD	R		Reserved bits.
3:0	TX_FIFO_LV	R	0x0	Transmit FIFO Level. Contains the number of valid data entries in the transmit FIFO.

4.4.2.5.10 RX_FIFO_LV

- **Name:** Receive FIFO Level Register
- **Description:** This register contains the number of valid data entries in the receive FIFO memory. This register can be ready at any time.
- **Base Address:** 0x4000C000 + x*0x100 (x: 0 for master, 1 for slave)
- **Offset:** 0x24
- **Reset Value:** 0x00000000

Table 4-129 Receive FIFO Level Register

Bits	Field Name	RW	Reset	Description
31:4	RSVD	R		Reserved bits.
3:0	RX_FIFO_LV	R	0x0	Receive FIFO Level. Contains the number of valid data entries in the receive FIFO.

4.4.2.5.11 STAT

- **Name:** Status Register
- **Description:** This is a read-only register used to indicate the current transfer status, FIFO status, and any transmission/reception errors that may have occurred. The status register may be read at any time.
- **Base Address:** 0x4000C000 + x*0x100 (x: 0 for master, 1 for slave)
- **Offset:** 0x28
- **Reset Value:** 0x00000006

Table 4-130 Status Register

Bits	Field Name	RW	Reset	Description
31:7	RSVD	R		Reserved bits.
6	DATA_COLN_ERR	R	0x0	Data Collision Error. This bit will be set if MISO input is asserted by other master, when the SPI master is in the middle of the transfer. This informs the processor that the last data transfer was halted before completion. This bit is cleared when read. **Note:** When the SPI is configured as a slave, this register serves no purpose.
5	TX_ERR	R	0x0	Transmission Error. Set if the transmit FIFO is empty when a transfer is started. Data from the previous transmission is resent on the txd line. This bit is cleared when read. **Value:** • 0x0 (NO_ERROR): No Error • 0x1 (TX_ERROR): Transmission Error **Note:**

Bits	Field Name	RW	Reset	Description
				When the SPI is configured as a master, this register serves no purpose.
4	RX_FIFO_FULL	R	0x0	Receive FIFO Full. When the receive FIFO is completely full, this bit is set. When the receive FIFO contains one or more empty location, this bit is cleared. **Value:** • 0x0 (NOT_FULL): Receive FIFO is not full • 0x1 (FULL): Receive FIFO is full
3	RX_FIFO_NE	R	0x0	Receive FIFO Not Empty. Set when the receive FIFO contains one or more entries and is cleared when the receive FIFO is empty. This bit can be polled by software to completely empty the receive FIFO. **Value:** • 0x0 (EMPTY): Receive FIFO is empty • 0x1 (NOT_EMPTY): Receive FIFO is not empty
2	TX_FIFO_EMPTY	R	0x1	Transmit FIFO Empty. When the transmit FIFO is completely empty, this bit is set. When the transmit FIFO contains one or more valid entries, this bit is cleared. This bit field does not request an interrupt. **Value:** • 0x0 (NOT_EMPTY): Transmit FIFO is not empty • 0x1 (EMPTY): Transmit FIFO is empty
1	TX_FIFO_NF	R	0x1	Transmit FIFO Not Full. Set when the transmit FIFO contains one or more empty locations, and is cleared when the FIFO is full. **Value:** • 0x0 (FULL): Transmit FIFO is full • 0x1 (NOT_FULL): Transmit FIFO is not Full
0	BUSY	R	0x0	SSI Busy Flag. When set, indicates that a serial transfer is in progress; when cleared indicates that the SPI is idle or disabled. **Value:** • 0x0 (INACTIVE): SPI is idle or disabled

Bits	Field Name	RW	Reset	Description
				• 0x1 (ACTIVE): SPI is actively transferring data

4.4.2.5.12 INT_MASK

- **Name:** Interrupt Mask Register
- **Description:** This read/write register masks or enables all interrupts generated by the SPI. When the SPI is configured as a slave device, the MSTIM bit field is not present. This changes the reset value from 0x3F for serial-master configurations to 0x1F for serial-slave configurations.
- **Base Address:** 0x4000C000 + x*0x100 (x: 0 for master, 1 for slave)
- **Offset:** 0x2C
- **Reset Value:** master ? 0x0000003F : 0x0000001F

Table 4-131 Interrupt Mask Register

Bits	Field Name	RW	Reset	Description
31:6	RSVD	R		Reserved bits.
5	MULTI_M_CIM	RW	0x1	Multi-Master Contention Interrupt Mask. **Value:** • 0x0 (MASKED): Disable mst_intr interrupt • 0x1 (UNMASKED): Enable mst_intr interrupt **Note:** When the SPI is configured as a slave, this register serves no purpose.
4	RX_FIFO_FIM	RW	0x1	Receive FIFO Full Interrupt Mask **Value:** • 0x0 (MASKED): Disable rxf_intr interrupt • 0x1 (UNMASKED): Enable rxf_intr interrupt
3	RX_FIFO_OIM	RW	0x1	Receive FIFO Overflow Interrupt Mask **Value:** • 0x0 (MASKED): Disable rxo_intr interrupt • 0x1 (UNMASKED): Enable rxo_intr interrupt
2	RX_FIFO_UIM	RW	0x1	Receive FIFO Underflow Interrupt Mask **Value:** • 0x0 (MASKED): Disable rxu_intr interrupt • 0x1 (UNMASKED): Enable rxu_intr interrupt
1	TX_FIFO_OIM	RW	0x1	Transmit FIFO Overflow Interrupt Mask

Bits	Field Name	RW	Reset	Description
				Value: • 0x0 (MASKED): Disable txo_intr interrupt • 0x1 (UNMASKED): Enable txo_intr interrupt
0	TX_FIFO_EIM	RW	0x1	Transmit FIFO Empty Interrupt Mask **Value:** • 0x0 (MASKED): Disable txe_intr interrupt • 0x1 (UNMASKED): Enable txe_intr interrupt

4.4.2.5.13 INT_STAT

- **Name:** Interrupt Status Register
- **Description:** This register reports the status of the SPI interrupts after they have been enabled.
- **Base Address:** 0x4000C000 + x*0x100 (x: 0 for master, 1 for slave)
- **Offset:** 0x30
- **Reset Value:** 0x00000000

Table 4-132 Interrupt Status Register

Bits	Field Name	RW	Reset	Description
31:6	RSVD	R		Reserved bits.
5	MULTI_M_CIS	R	0x0	Multi-Master Contention Interrupt Status **Value:** • 0x0 (INACTIVE): mst_intr interrupt is not active after being enabled • 0x1 (ACTIVE): mst_intr interrupt is full after being enabled
4	RX_FIFO_FIS	R	0x0	Receive FIFO Full Interrupt Status **Value:** • 0x0 (INACTIVE): rxf_intr interrupt is not active after being enabled • 0x1 (ACTIVE): rxf_intr interrupt is full after being enabled
3	RX_FIFO_OIS	R	0x0	Receive FIFO Overflow Interrupt Status **Value:** • 0x0 (INACTIVE): rxo_intr interrupt is not active after being enabled • 0x1 (ACTIVE): rxo_intr interrupt is active after being enabled
2	RX_FIFO_UIS	R	0x0	Receive FIFO Underflow Interrupt Status

Bits	Field Name	RW	Reset	Description
				Value: • 0x0 (INACTIVE): rxu_intr interrupt is not active after being enabled • 0x1 (ACTIVE): rxu_intr interrupt is active after being enabled
1	TX_FIFO_OIS	R	0x0	Transmit FIFO Overflow Interrupt Status **Value:** • 0x0 (INACTIVE): txo_intr interrupt is not active after being enabled • 0x1 (ACTIVE): txo_intr interrupt is active after being enabled
0	TX_FIFO_EIS	R	0x0	Transmit FIFO Empty Interrupt Status **Value:** • 0x0 (INACTIVE): txe_intr interrupt is not active after being enabled • 0x1 (ACTIVE): txe_intr interrupt is active after being enabled

4.4.2.5.14 RAW_INT_STAT

- **Name:** Raw Interrupt Status Register
- **Description:** This read-only register reports the status of the SPI interrupts prior to being enabled.
- **Base Address:** 0x4000C000 + x*0x100 (x: 0 for master, 1 for slave)
- **Offset:** 0x34
- **Reset Value:** 0x00000000

Table 4-133 Raw Interrupt Status Register

Bits	Field Name	RW	Reset	Description
31:6	RSVD	R		Reserved bits.
5	MULTI_M_CRIS	R	0x0	Multi-Master Contention Raw Interrupt Status **Value:** • 0x0 (INACTIVE): mst_intr interrupt is not active prior to being enabled • 0x1 (ACTIVE): mst_intr interrupt is full prior to being enabled
4	RX_FIFO_FRIS	R	0x0	Receive FIFO Full Raw Interrupt Status **Value:**

Bits	Field Name	RW	Reset	Description
				• 0x0 (INACTIVE): rxf_intr interrupt is not active prior to being enabled • 0x1 (ACTIVE): rxf_intr interrupt is active prior to being enabled
3	RX_FIFO_ORIS	R	0x0	Receive FIFO Overflow Raw Interrupt Status **Value:** • 0x1 (ACTIVE): rxo_intr interrupt is not active prior to being enabled • 0x0 (INACTIVE): rxo_intr interrupt is active prior to being enabled
2	RX_FIFO_URIS	R	0x0	Receive FIFO Underflow Raw Interrupt Status **Value:** • 0x0 (INACTIVE): rxu_intr interrupt is not active prior to being enabled • 0x1 (ACTIVE): rxu_intr interrupt is active prior to being enabled
1	TX_FIFO_ORIS	R	0x0	Transmit FIFO Overflow Raw Interrupt Status **Value:** • 0x0 (INACTIVE): txo_intr interrupt is not active prior to being enabled • 0x1 (ACTIVE): txo_intr interrupt is active prior to being enabled
0	TX_FIFO_ERIS	R	0x0	Transmit FIFO Empty Raw Interrupt Status **Value:** • 0x0 (INACTIVE): txe_intr interrupt is not active prior to being enabled • 0x1 (ACTIVE): txe_intr interrupt is active prior to being enabled

4.4.2.5.15 TX_FIFO_OIC

- **Name:** Transmit FIFO Overflow Interrupt Clear Register
- **Description:** Transmit FIFO Overflow Interrupt Clear Register.
- **Base Address:** 0x4000C000 + x*0x100 (x: 0 for master, 1 for slave)
- **Offset:** 0x38
- **Reset Value:** 0x00000000

Table 4-134 Transmit FIFO Overflow Interrupt Clear Register

Bits	Field Name	RW	Reset	Description
31:1	RSVD	R		Reserved bits.
0	TX_FIFO_OIC	R	0x0	Clear Transmit FIFO Overflow Interrupt. This register reflects the status of the interrupt. A read from this register clears the txo_intr interrupt; writing has no effect.

4.4.2.5.16 RX_FIFO_OIC

- **Name:** Receive FIFO Overflow Interrupt Clear Register
- **Description:** Receive FIFO Overflow Interrupt Clear Register.
- **Base Address:** 0x4000C000 + x*0x100 (x: 0 for master, 1 for slave)
- **Offset:** 0x3C
- **Reset Value:** 0x00000000

Table 4-135 Receive FIFO Overflow Interrupt Clear Register

Bits	Field Name	RW	Reset	Description
31:1	RSVD	R		Reserved bits.
0	RX_FIFO_OIC	R	0x0	Clear Receive FIFO Overflow Interrupt. This register reflects the status of the interrupt. A read from this register clears the rxo_intr interrupt; writing has no effect.

4.4.2.5.17 RX_FIFO_UIC

- **Name:** Receive FIFO Underflow Interrupt Clear Register
- **Description:** Receive FIFO Underflow Interrupt Clear Register.
- **Base Address:** 0x4000C000 + x*0x100 (x: 0 for master, 1 for slave)
- **Offset:** 0x40
- **Reset Value:** 0x00000000

Table 4-136 Receive FIFO Underflow Interrupt Clear Register

Bits	Field Name	RW	Reset	Description
31:1	RSVD	R		Reserved bits.
0	RX_FIFO_UIC	R	0x0	Clear Receive FIFO Underflow Interrupt.

Bits	Field Name	RW	Reset	Description
				This register reflects the status of the interrupt. A read from this register clears the rxu_intr interrupt; writing has no effect.

4.4.2.5.18 MULTI_M_IC

- **Name:** Multi-Master Interrupt Clear Register
- **Description:** Multi-Master Interrupt Clear Register.
- **Base Address:** 0x4000C000 + x*0x100 (x: 0 for master, 1 for slave)
- **Offset:** 0x44
- **Reset Value:** 0x00000000

Table 4-137 Multi-Master Interrupt Clear Register

Bits	Field Name	RW	Reset	Description
31:1	RSVD	R		Reserved bits.
0	MULTI_M_IC	R	0x0	Clear Multi-Master Contention Interrupt. This register reflects the status of the interrupt. A read from this register clears the mst_intr interrupt; writing has no effect.

4.4.2.5.19 INT_CLR

- **Name:** Interrupt Clear Register
- **Description:** Interrupt Clear Register.
- **Base Address:** 0x4000C000 + x*0x100 (x: 0 for master, 1 for slave)
- **Offset:** 0x48
- **Reset Value:** 0x00000000

Table 4-138 Interrupt Clear Register

Bits	Field Name	RW	Reset	Description
31:1	RSVD	R		Reserved bits.
0	INT_CLR	R	0x0	Clear Interrupts. This register is set if any of the interrupts below are active. A read clears the txo_intr, rxu_intr, rxo_intr, and the mst_intr interrupts. Writing to this register has no effect.

4.4.2.5.20 DMA_CTRL

- **Name:** DMA Control Register
- **Description:** The register is used to enable the DMA Controller interface operation.
- **Base Address:** 0x4000C000 + x*0x100 (x: 0 for master, 1 for slave)
- **Offset:** 0x4C
- **Reset Value:** 0x00000000

Table 4-139 DMA Control Register

Bits	Field Name	RW	Reset	Description
31:2	RSVD	R		Reserved bits.
1	TX_DMA_EN	RW	0x0	Transmit DMA Enable. This bit enables/disables the transmit FIFO DMA channel. **Value:** • 0x0 (DISABLE): Transmit DMA disabled • 0x1 (ENABLED): Transmit DMA enabled
0	RX_DMA_EN	RW	0x0	Receive DMA Enable. This bit enables/disables the receive FIFO DMA channel **Value:** • 0x0 (DISABLE): Receive DMA disabled • 0x1 (ENABLED): Receive DMA enabled

4.4.2.5.21 DMA_TX_DL

- **Name:** DMA Transmit Data Level
- **Description:** This register controls the threshold value for the transmit FIFO memory.
- **Base Address:** 0x4000C000 + x*0x100 (x: 0 for master, 1 for slave)
- **Offset:** 0x50
- **Reset Value:** 0x00000000

Table 4-140 DMA Transmit Data Level

Bits	Field Name	RW	Reset	Description
31:3	RSVD	R		Reserved bits.
2:0	DMA_TX_DL	RW	0x0	Transmit Data Level. This bit field controls the level at which a DMA request is made by the transmit logic. It is equal to the watermark level; that is, the dma_tx_req signal is generated when the

Bits	Field Name	RW	Reset	Description
				number of valid data entries in the transmit FIFO is equal to or below this field value, and TX_DMA_EN = 1.

4.4.2.5.22 DMA_RX_DL

- **Name:** DMA Receive Data Level
- **Description:** This register controls the threshold value for the receive FIFO memory.
- **Base Address:** 0x4000C000 + x*0x100 (x: 0 for master, 1 for slave)
- **Offset:** 0x54
- **Reset Value:** 0x00000000

Table 4-141 DMA Receive Data Level

Bits	Field Name	RW	Reset	Description
31:3	RSVD	R		Reserved bits
2:0	DMA_RX_DL	RW	0x0	Receive Data Level. This bit field controls the level at which a DMA request is made by the receive logic. The watermark level = DMA_RX_DL+1; that is, dma_rx_req is generated when the number of valid data entries in the receive FIFO is equal to or above this field value + 1, and RX_DMA_EN=1.

4.4.2.5.23 DATA

- **Name:** Data Register
- **Description:** The SPI data register is a 32-bit read/write buffer for the transmit/receive FIFOs. When the register is read, data in the receive FIFO buffer is accessed. When it is written to, data is moved into the transmit FIFO buffer; a write can occur only when SSI_EN[0] = 1. FIFOs are reset when SSI_EN[0] = 0
- **Base Address:** 0x4000C000 + x*0x100 (x: 0 for master, 1 for slave)
- **Offset:** 0x60
- **Reset Value:** 0x00000000

Table 4-142 Data Register

Bits	Field Name	RW	Reset	Description
31:0	DATA	RW	0x0	Data Register. When writing to this register, you must right-justify the data. Read data is automatically right-justified.

4.4.2.5.24 RX_SMP_DLY

Name: Receive sample delay Register

Description: This register control the number of SCLK cycles that are delayed (from the default sample time) before the actual sample of the rxd input occurs. It is impossible to write to this register when the SPI is enabled.

- **Base Address:** 0x4000C000 + x*0x100 (x: 0 for master, 1 for slave)
- **Offset:** 0xF0
- **Reset Value:** 0x00000000

Table 4-143 Data Register

Bits	Field Name	RW	Reset	Description
3:0	RX_SMP_DLY	RW	0x0	Receive sample delay

4.4.2.6 Electrical Specifications

4.4.2.6.1 SPIM Electrical Specifications

Table 4-144 SPIM electrical specifications

Parameter	Description	Min.	Typ.	Max.	Unit
f_{SCK}	Clock frequency			48	MHz
t_{CSCK}	SCK clock period	20.83			ns
t_{RSCK}	SCK rise time, 15 pF loading			$t_{RF,15pF}$[1]	ns
t_{FSCK}	SCK fall time, 15 pF loading			$t_{RF,15pF}$[1]	ns
t_{WSCKH}	SCK clock high time	7.625[2]			ns
t_{WSCKL}	SCK clock low time	7.625[2]			ns
t_{SUMI}	Data input setup time (MISO to SCK edge)	5			ns
t_{HMI}	Data input hold time (SCK edge to MISO change)	5			ns
t_{VMO}	Data output valid time (SCK edge to MOSI valid)	0			ns
t_{HMO}	MOSI hold time after CLK edge			20	ns

[1] The SCK rise time and the SCK fall time are obtained with SPIM timing shown in Figure 4-22.
[2] The SCK clock high time and SCK clock low time are calculated by $(t_{CSCK}/2)–t_{RSCK}$ and $(t_{CSCK}/2)–t_{FSCK}$, respectively.

The SPIM timing diagram is as below:

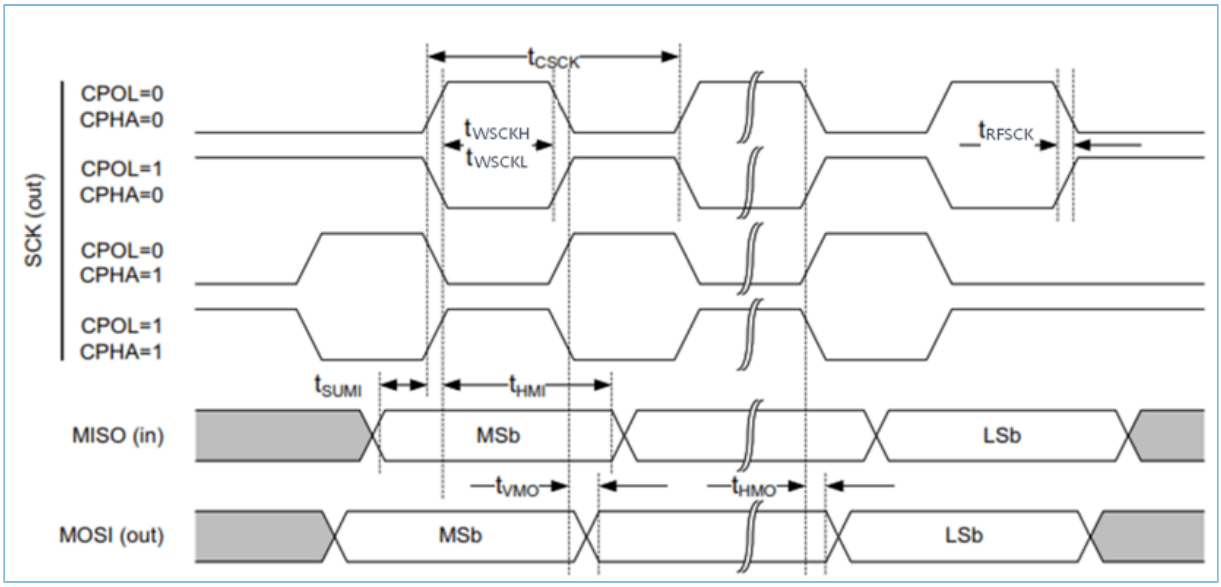

Figure 4-22 SPIM timing diagram

4.4.2.6.2 SPIS Electrical Specifications

Table 4-145 SPIS electrical specifications

Parameter	Description	Min.	Typ.	Max.	Unit
f_{SCK}	Clock frequency			16	MHz
t_{CSCK}	SCK input clock period	62.5			ns
t_{RFSCK}	SCK input clock rise/fall time			22	ns
t_{WSCKH}	SCK input clock high time	22			ns
t_{WSCKL}	SCK input clock low time	22			ns
t_{SUCSN}	CSN input setup time	62.5			ns
t_{HCSN}	CSN input hold time	31.25			ns
t_{ASO}	Data output access time (from CSN to MISO valid)			62.5	ns
t_{DISSO}	Data output disable time (from CSN to MISO disabled)			62.5	ns
t_{VSO}	Data output valid time (from SCK to MISO valid)			20	ns
t_{HSO}	Data output hold time (from SCK to MISO change)	8			ns
t_{SUSI}	Data input setup time (from MOSI to SCK)	18			ns
t_{HSI}	Data input hold time (from SCK to MOSI change)	0			ns

Note:

If SPIS performs RX and TX concurrently, the maximum bit rate decreases to 8 Mbps.

The SPIS timing diagram is shown below:

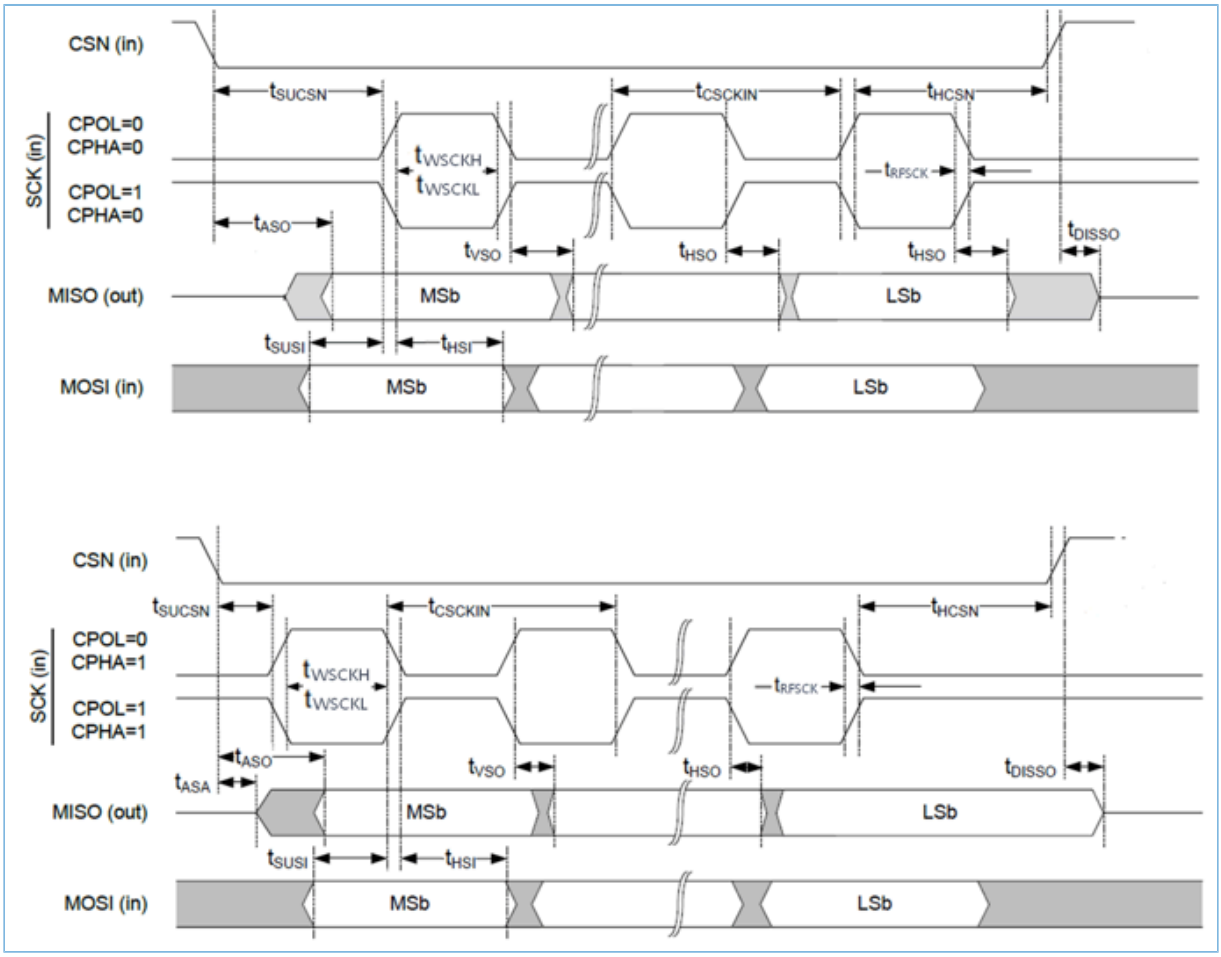

Figure 4-23 SPIS timing diagram

4.4.3 I2C

4.4.3.1 Introduction

The Inter-Integrated Circuit (I2C) bus is a two-wire serial interface that is widely used for low-speed communication between chipsets in a system. Using I2C, a microcontroller can communicate with peripherals such as sensors, data converters, and I/O interfaces.

The device has four I2C module instances: I2C0, I2C1, I2C2, and I2C3 that can be configured as either Master or Slave.

4.4.3.2 Main Features

The two-wire I2C serial interface consists of a serial data line (SDA) and a serial clock line (SCL) that support:

- Standard Mode (up to 100 kbit/s)
- Fast Mode (up to 400 kbit/s)
- Fast Plus Mode (up to 1000 kbit/s)
- High-speed Mode (up to 3.4 Mbit/s)

- Clock synchronization
- Master or Slave I2C operation
- 7-bit or 10-bit addressing
- Mixed 7-bit and mixed 10-bit format transfer
- Transmit and receive FIFO depth at 32 bytes
- Interrupt or polled-mode operation
- Simple software interface
- DMA support

4.4.3.3 Functional Description

The I2C Controller is made up of an APB slave interface, an I2C interface, and FIFO logic to maintain consistency between the two interfaces, as well as DMA and interrupt logics. A simplified block diagram is illustrated in Figure 4-24.

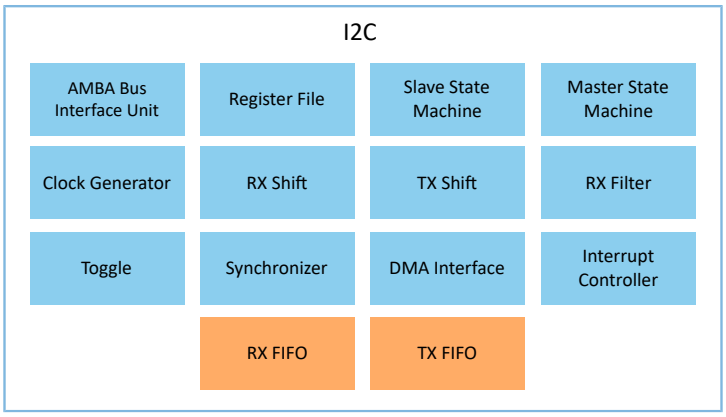

Figure 4-24 I2C block diagram

The following list describes each of the major blocks shown above.

- AMBA Bus Interface Unit connects to APB bus.
- Register File contains configuration registers and is the interface with software.
- Slave State Machine follows the protocol for a slave and monitors bus for address match.
- Master State Machine generates the I2C protocol for the master transfers.
- Clock Generator produces the required timing to do the following:
  - Generate the SCL clock when configured as a master.
  - Check for bus idle.
  - Generate a START and a STOP.
  - Set up the data and hold the data.

- RX Shift takes data into the design and extracts it in byte format.
- TX Shift presents data supplied by GPU for transfer on the I2C bus.
- RX Filter detects the events in the bus; for example, start, stop, and arbitration lost.
- Toggle generates pulses on both sides and toggles to transfer signals across clock domains.
- Synchronizer transfers signals from one clock domain to another.
- DMA Interface generates handshaking signals to the central DMA controller in order to automate the data transfer without CPU intervention.
- Interrupt Controller generates raw interrupt and interrupt flags, allowing them to be set and cleared.
- RX FIFO/TX FIFO holds the RX FIFO and TX FIFO register banks and controllers, along with their status levels.

4.4.3.3.1 I2C Operation

The I2C is a synchronous serial interface. The SDA line is a bidirectional signal and changes only when the SCL line is low, except for STOP, START, and RESTART conditions. The output drivers are open-drain or open-collector to perform wired-AND functions on the bus. The maximum number of devices on the bus is limited by only the maximum capacitance specification of 400 pF. Data is transmitted in byte packages.

The master is responsible for generating the clock and controlling the transfer of data. The slave is responsible for either transmitting or receiving data to/from the master. The acknowledgement of data is sent by the device that is receiving data, which can be either a master or a slave. The I2C protocol also allows multiple masters to reside on the I2C bus and uses an arbitration procedure to determine the bus ownership.

When a master wants to communicate with a slave, the master transmits a START/RESTART condition that is then followed by the slave address and a control bit ($R/\overline{W}$) to determine if the master wants to transmit data to or receive data from the slave. The slave then sends an acknowledge (ACK) pulse after the address.

If the master (master-transmitter) is writing to the slave (slave-receiver), the receiver gets one byte of data. This transaction continues until the master terminates the transmission with a STOP condition. If the master is reading from a slave (master-receiver), the slave transmits (slave-transmitter) a byte of data to the master, and the master then acknowledges the transaction with the ACK pulse. This transaction continues until the master terminates the transmission by not acknowledging (NACK) the transaction after the last byte is received, and then the master issues a STOP condition or addresses another slave after issuing a RESTART condition. This behavior is illustrated in Figure 4-25.

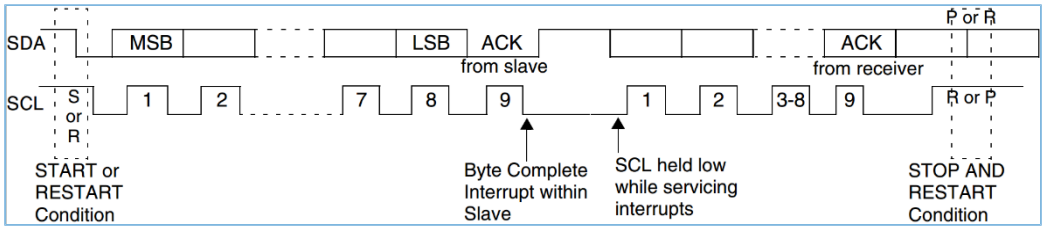

Figure 4-25 Data transfer on the I2C Bus

Note that, I2C block should only be set to operate as an I2C master or an I2C slave, but not both simultaneously. This is achieved by ensuring that bit 0 and bit 6 of the I2C Control Register (CTRL) are never set to 0 and 1, respectively.

4.4.3.3.2 START and STOP Conditions

When I2C operates as a master (writing 1 to CTRL[0], and writing 1 to CTRL[6]), putting data into the transmit FIFO causes the I2C block to generate a START condition on the I2C bus. Writing 1 to DATA_CMD[9] causes the I2C block to generate a STOP condition on the I2C bus; a STOP condition is not issued if this bit is not set, even if the transmit FIFO is empty. This behavior is illustrated as follows.

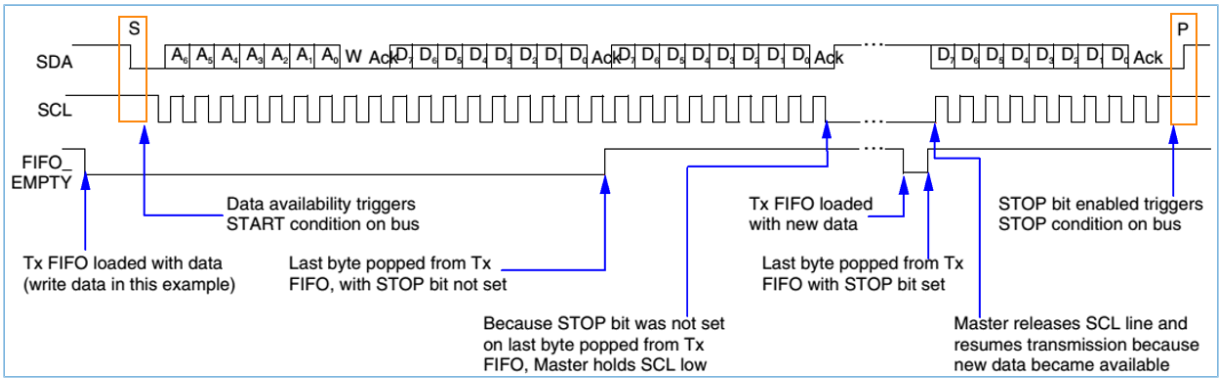

Figure 4-26 STOP generation

When operating as a slave (writing 0 to CTRL[0], and writing 0 to CTRL[6]), I2C block does not generate START and STOP conditions, as per the protocol. However, if a read request is made to the I2C block, it holds the SCL line low until read data has been supplied to it. This stalls the I2C bus until read data is provided to the slave I2C block, or the I2C block slave is disabled by writing 0 to I2C ENABLE Register (EN[0]). When the bus is idle, both the SCL and SDA signals are pulled high through external pull-up resistors on the bus. When a master wants to start a transmission on the bus, the master issues a START condition. This is defined as a high-to-low transition of the SDA signal while SCL is high. When a master wants to terminate the transmission, the master issues a STOP condition. This is defined as a low-to-high transition of the SDA line while SCL is high. Figure 4-27 shows the timing of the START and STOP conditions. When data is being transmitted on the bus, the SDA line must be stable when the SCL is high.

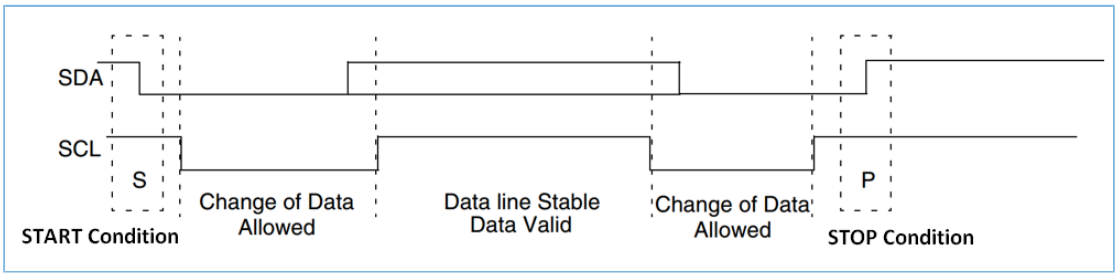

Figure 4-27 Timing of START and STOP conditions

4.4.3.3.3 Addressing Slave

There are two address formats: the 7-bit address format and the 10-bit address format. The I2C block supports mixed formats of read and write transactions in 7-bit or 10-bit addressing modes. The I2C block does not support mixed address or mixed address format, that is, a 7-bit address transaction followed by a 10-bit address transaction or vice versa (combined format transactions).

To initiate combined format transfers, CTRL[5] should be enabled. With this value set and operating as a master, when the I2C block completes an I2C transfer, it checks the transmit FIFO and executes the next transfer. If the direction of this transfer differs from the previous transfer, the combined format is used to issue the transfer. If the transmit FIFO is empty when the current I2C transfer completes, a STOP is issued and the next transfer is issued following a START condition.

7-bit Address Format

Using the 7-bit address format, users should write 0 to CTRL[3] and the slave address is stored at I2C Slave Address Register (S_ADDR[6:0]). The first seven bits (bits 7:1) of the first byte set the slave address and the least significant bit (bit 0) (LSB) is the R/W bit as shown in Figure 4-28. When bit 0 ($R/\overline{W}$) is set to 0, the master writes to the slave. When bit 0 ($R/\overline{W}$) is set to 1, the master reads from the slave.

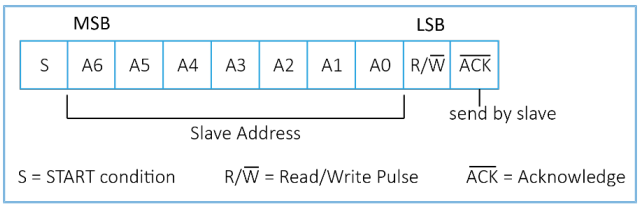

Figure 4-28 7-bit address format

10-bit Address Format

Using the 10-bit address format, users should write 1 to CTRL[3] and the slave address is stored at I2C Slave Address Register (S_ADDR[9:0]). Two bytes are transferred to set the 10-bit address. The transfer of the first byte contains the following bit definition. The first five bits (bits 7:3) notify the slave that this is a 10-bit transfer followed by the next two bits (bits 2:1), which set the slave address (bits 9:8), and the LSB bit (bit 0) is the $R/\overline{W}$ bit. The second byte transferred sets bits 7:0 of the slave address.

Figure 4-29 shows the 10-bit address format.

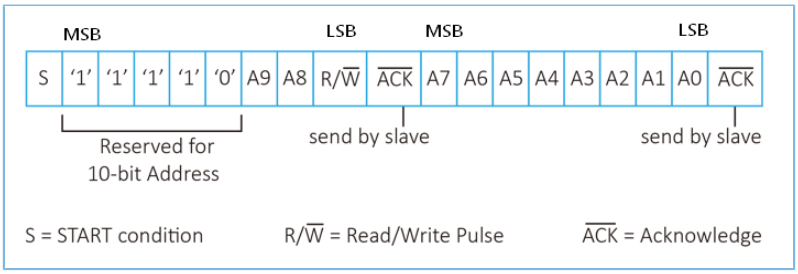

Figure 4-29 10-bit address format

Table 4-146 defines the special purpose and reserved first byte addresses.

Table 4-146 I2C definition of bits in first byte

Slave Address	R/W Bit	Description
0000 000	0	General Call Address. DW_apb_i2c places the data in the receive buffer and issues a General Call interrupt.
0000 000	1	START byte

Slave Address	R/W Bit	Description
0000 001	X	CBUS address. I2C ignores these accesses.
0000 010	X	Reserved
0000 010	X	Reserved
0000 011	X	High-speed master code
1111 1XX	X	Reserved
1111 0XX	X	10-bit slave addressing
0001 000	X	SMBus Host
0001 100	X	SMBus Alert Response Address
1100 001	X	SMBus Device Default Address

4.4.3.3.4 Transmitting and Receiving

A master can initiate data transmission and reception to/from the bus, acting as either a master-transmitter or master-receiver. A slave responds to requests from the master to either transmit data or receive data to/from the bus, acting as either a slave-transmitter or slave-receiver, respectively.

Master-Transmitter and Slave-Receiver

All data is transmitted in byte format, with no limit on the number of bytes transferred per data transfer. After the master sends the address and $R/\overline{W}$ bit or the master transmits a byte of data to the slave, the slave-receiver must respond with the ACK signal. When a slave-receiver does not respond with an ACK pulse, the master aborts the transfer by issuing a STOP condition. The slave must leave the SDA line high so that the master can abort the transfer.

If the master-transmitter is transmitting data as shown in Figure 4-30, the slave-receiver responds to the master-transmitter with an ACK pulse after every data byte is received.

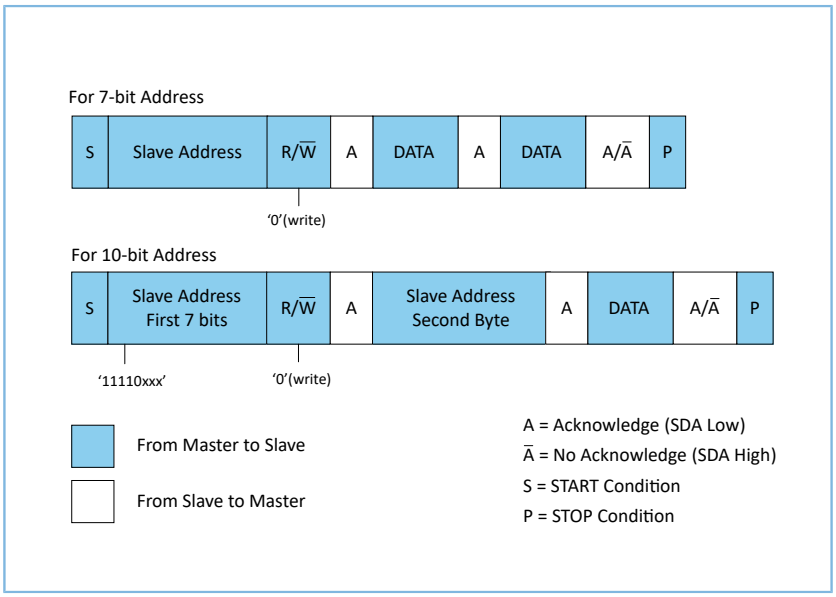

Figure 4-30 Master-transmitter protocol

Master-Receiver and Slave-Transmitter

If the master is receiving data as shown in Figure 4-31, the master responds to the slave-transmitter with an ACK pulse after a byte of data has been received, except for the last byte. This is the way in which the master-receiver notifies the slave-transmitter that this is the last byte. The slave-transmitter relinquishes the SDA line after detecting the No Acknowledge (NACK) so that the master can issue a STOP condition.

When a master does not want to relinquish the bus with a STOP condition, the master can issue a RESTART condition. This is identical to a START condition except it occurs after the ACK pulse. Operating in master mode, the I2C block can then communicate with the same slave using a transfer of a different direction.

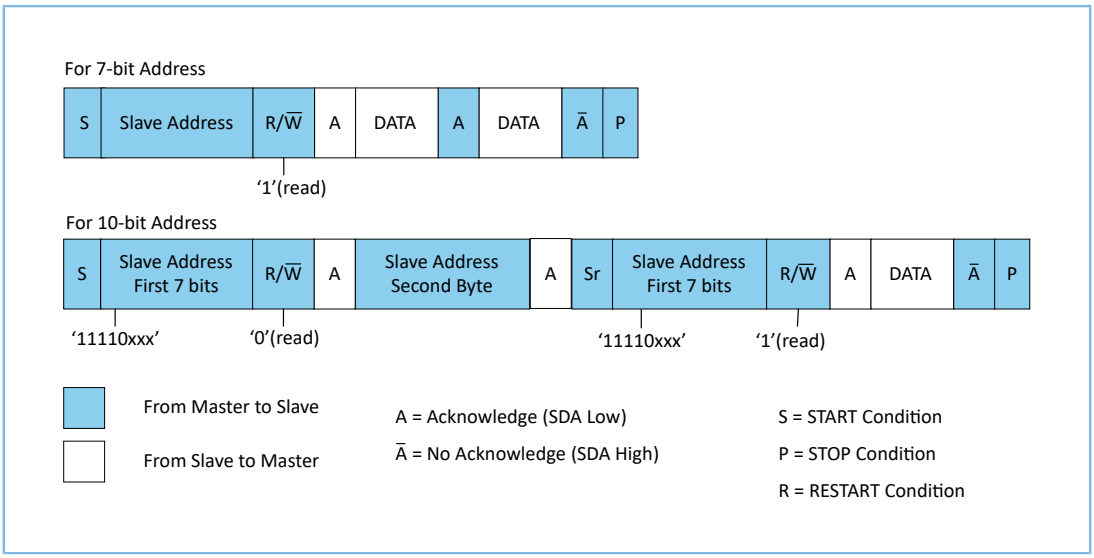

Figure 4-31 Master-receiver protocol

Bus Clear Feature

I2C block supports the bus clear feature that provides graceful recovery of data (SDA) and clock (SCL) lines during unlikely events in which either the clock or data line is stuck at low.

SDA Line Stuck at Low Recovery

In cast of SDA line stuck at low, master performs the following actions to recover as shown in Figure 4-32.

- Master sends a maximum of 9 clock pulses to recover the bus low within those 9 clocks.
  - The number of clock pulses will vary with the number of bits that remain to be sent by the slave. As the maximum number of bits is 9, master sends up to 9 clock pluses and allows the slave to recover it.
  - The master attempts to assert a Logic 1 on the SDA line and check whether SDA is recovered. If the SDA is not recovered, it will continue to send a maximum of 9 SCL clocks.
- If SDA line is recovered within 9 clock pulses, then the master will send the STOP to release the bus.
- If SDA line is not recovered even after 9th clock pulse, then the system needs a hardware reset.

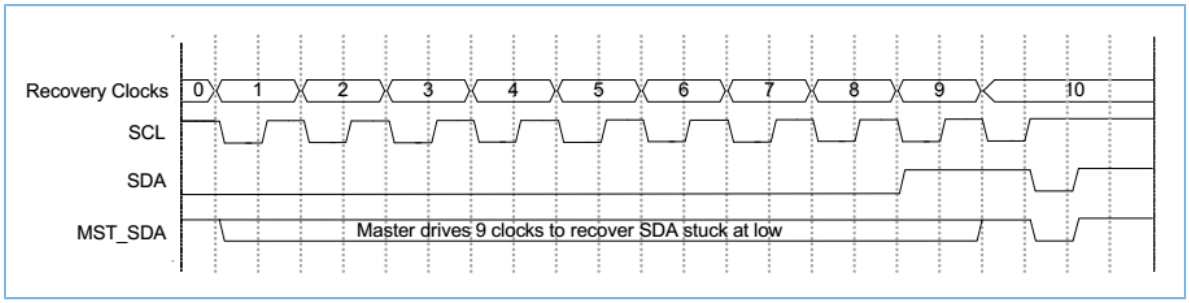

Figure 4-32 SDA recovery with 9 SCL clocks

The detailed flow to recover the SDA stuck at LOW is shown in Figure 4-33.

SCL Line Stuck at Low Recovery

In the unlikely event (due to an electric failure of a circuit) where the clock (SCL) is stuck at LOW, there is no effective method to overcome this problem but to reset the bus using the hardware reset signal. The detailed flow to recover the SCL stuck at LOW is shown in Figure 4-33.

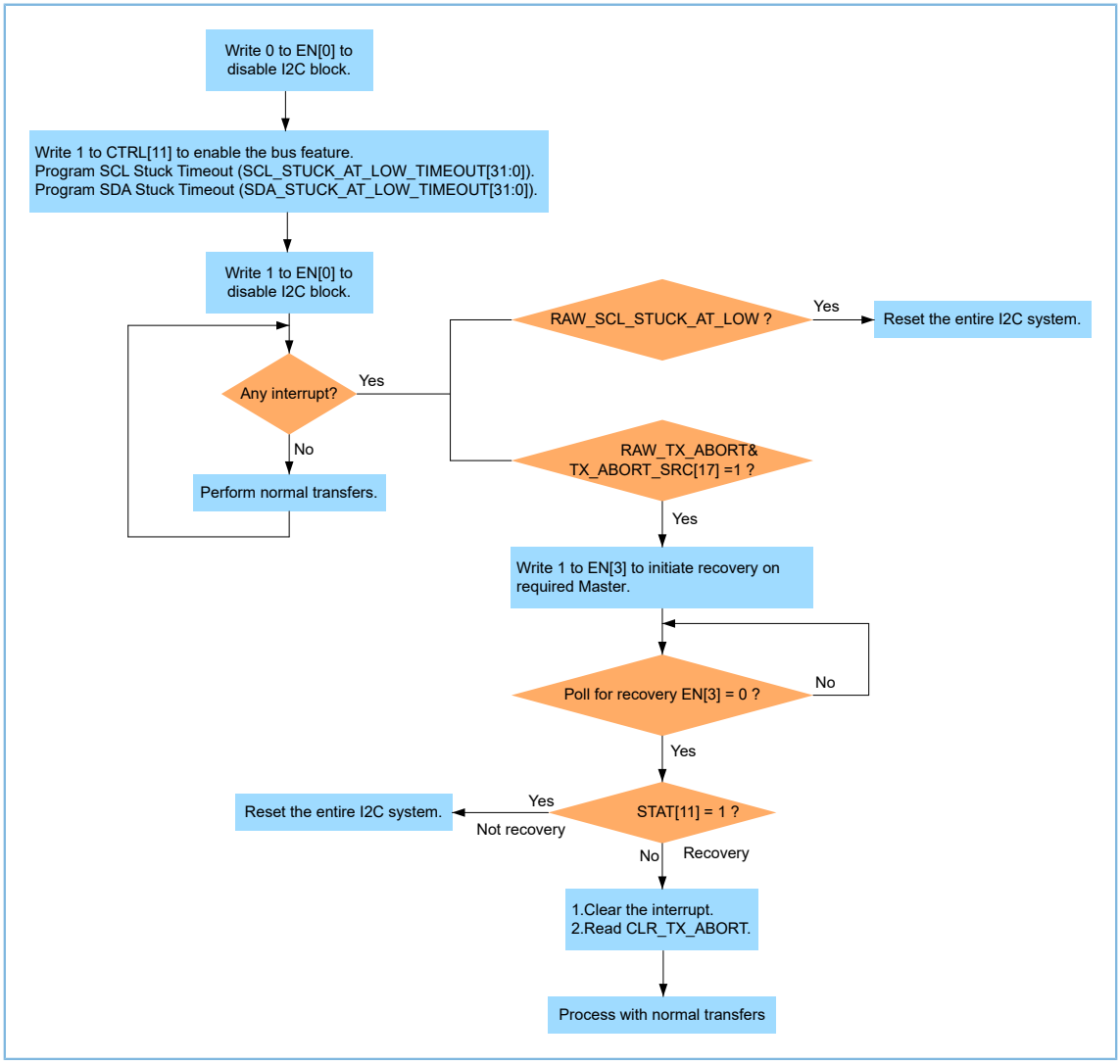

Figure 4-33 Flowchart for SCL and SDA bus recovery

4.4.3.3.5 SDA Hold Time

The I2C protocol specification requires 300 ns of hold time on the SDA signal (t_{HD_DAT}, SDA hold time, see Figure 4-36 in standard mode and fast mode, and a hold time long enough to bridge the undefined part between logic 1 and logic 0 of the falling edge of SCL in high speed mode and fast mode plus.

Board delays on the SCL and SDA signals can mean that the hold-time requirement is met at the I2C master, but not at the I2C slave (or vice versa). As each application encounters differing board delays, the I2C block contains a software programmable register (SDA_HOLD) to enable dynamic adjustment of the SDA hold time.

SDA_HOLD[15:0] are used to control the hold time of SDA during transmit in both slave and master mode (After SCL goes from HIGH to LOW).

SDA_HOLD[23:16] are used to extend the SDA transition (if any) whenever SCL is HIGH in the receiver (in either master or slave mode).

If different SDA hold times are required for different speed modes, the SDA_HOLD register must be reprogrammed when the speed mode is being changed. The SDA_HOLD register can be programmed only when the DW_apb_i2c is disabled (EN[0] = 0).

SDA Hold Timings in Receiver

When I2C block acts as a receiver, according to the I2C protocol, the device should internally hold the SDA line to bridge undefined gap between logic 1 and logic 0 of SCL.

SDA_HOLD[23:16] can be used to alter the internal hold time which I2C block applies to the incoming SDA line. Each value in the SDA_HOLD[23:16] represents a unit of one sclk (in serial clock implementation) period. The minimum value of SDA_HOLD[23:16] is 0. This hold time is applicable only when SCL is HIGH. The receiver does not extend the SDA after SCL goes LOW internally. Figure 4-34 shows the I2C block as receiver with SDA_HOLD[23:16] being programmed to greater than or equal to 3. If SDA_HOLD[23:16] are greater than 3, I2C block does not hold SDA beyond 3 sclk cycles, because SCL goes low internally.

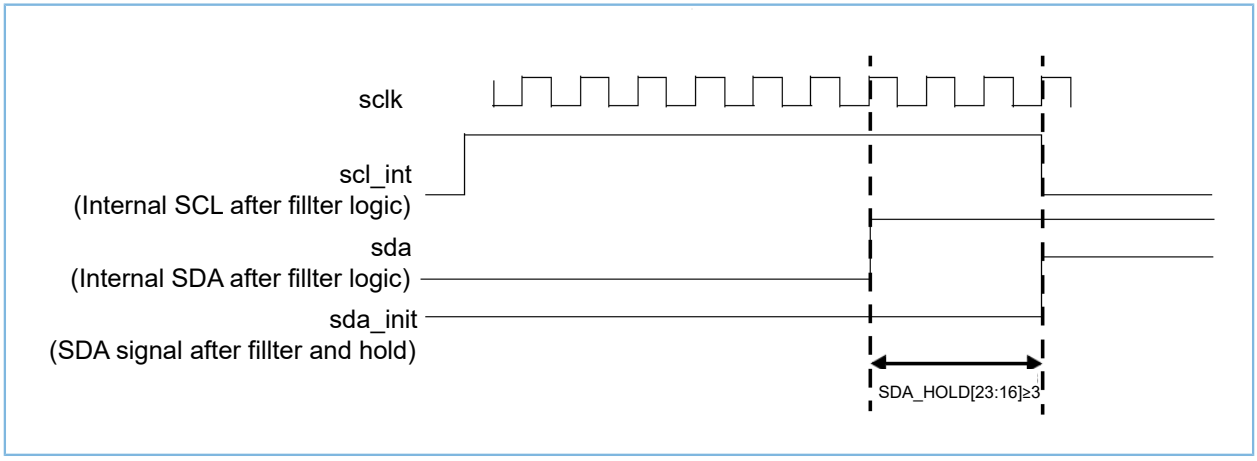

Figure 4-34 I2C block as receiver with SDA hold time

The maximum values of SDA_HOLD[23:16] that can be programmed in the register for the respective speed modes are derived from the equations shown in the table below.

Table 4-147 Maximum values for SDA_HOLD[23:16]

Speed Mode	Maximum SDA_HOLD[23:16] Value
Standard Mode	SS_CLK_HCNT[15:0] – FS_SPKLEN[7:0] -3
Fast Mode or Fast Mode Plus	FS_CLK_HCNT[15:0] – FS_SPKLEN[7:0] -3
High Speed	Min{FS_CLK_HCNT[15:0] – FS_SPKLEN[7:0] -3, HS_CLK_HCNT[15:0] – HS_SPKLEN[7:0] -3}

SDA Hold Timings in Transmitter

The SDA_HOLD [15:0] can be used to alter the timing of the generated SDA (data_oe, see Figure 4-35) signal by the I2C block. Each value in the SDA_HOLD [15:0] represents a unit of one sclk period.

When the I2C block is operating in master mode, the minimum t_{HD_DAT} timing is one ic_clk period. Therefore even when SDA_TX_HOLD has a value of zero, the I2C block drives SDA (data_oe) one sclk cycle after driving SCL (scl_oe) to logic 0. For all other values of SDA_HOLD[15:0], drive on SDA (data_oe) occurs SDA_HOLD[15:0] sclk cycles after driving SCL (scl_oe) to logic 0.

When the I2C block is operating in slave mode, the minimum t_{HD_DAT} timing is SPKLEN + 7 sclk periods, where SPKLEN is FS_SPKLEN[7:0] (operating in standard mode, fast mode, or fast mode plus) or HS_SPKLEN[7:0] (operating in high speed mode).

This delay allows for synchronization and spike suppression on the SCL sample. Therefore, even when SDA_HOLD[15:0] has a value less than SPKLEN + 7, the I2C block drives SDA (data_oe) SPKLEN + 7 ic_clk cycles after SCL (scl_oe) has transitioned to logic 0. For all other values of SDA_HOLD[15:0], drive on SDA occurs SDA_HOLD[15:0] sclk cycles after SCL has transitioned to logic 0.

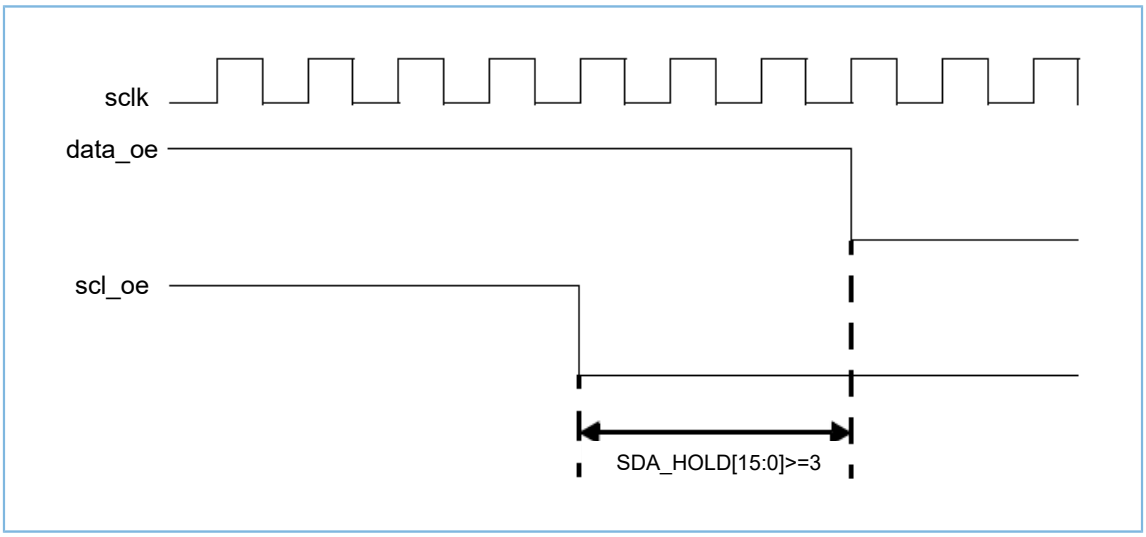

Figure 4-35 I2C Master implementing t_{HD_DAT} with SDA_HOLD[15:0] = 3

When the I2C block is operating in Slave Mode, the minimum t_{HD_DAT} timing is SPKLEN + 7 sclk periods, where SPKLEN is FS_SPKLEN[7:0] (operating in standard mode, fast mode, or fast mode plus) or HS_SPKLEN[7:0] (operating in high speed mode).

4.4.3.4 Registers

4.4.3.4.1 CTRL

- **Name**: I2C Control Register
- **Description**: This register can be written only when the I2C is disabled, which corresponds to the EN[0] register being set to 0. Writes at other times have no effect.
- **Base Address**: 0x4000C300 + x*0x100
- **Offset**: 0x00
- **Reset Value**: 0x0000007F

Table 4-148 I2C Control Register

Bits	Field Name	RW	Reset	Description
31:12	RSVD	R		Reserved bits
11	BUS_CLEAR_FEATURE_CTRL	RW	0x0	In Master mode **Value**: • 0x1: Bus Clear Feature is enabled. • 0x0: Bus Clear Feature is disabled.
10	STOP_DET_M_ACTIVE	RW	0x0	In Master mode **Value**: • 0x1: Issues the STOP_DET interrupt only when master is active. • 0x0: Issues the STOP_DET irrespective of whether master is active or not.
9	RX_FIFO_FULL_HLD_CTRL	RW	0x0	This bit controls whether I2C should hold the bus when the RX FIFO is physically full. **Value**: • 0x1 (ENABLED): Hold bus when RX_FIFO is full. • 0x0 (DISABLED): Overflow when RX_FIFO is full.
8	TX_EMPTY_CTRL	RW	0x0	This bit controls the generation of the TX_EMPTY interrupt, as described in the RAW_INT_STAT register. **Value**: • 0x0 (DISABLED): Default behavior of TX_EMPTY interrupt. • 0x1 (ENABLED): Controlled generation of TX_EMPTY interrupt.
7	STOP_DET_INT	RW	0x0	In slave mode **Value**: • 0x0: Issues the STOP_DET irrespective of whether it is addressed or not.

Bits	Field Name	RW	Reset	Description
				• 0x1: Issues the STOP_DET interrupt only when it is addressed. **Note**: During a general call address, the slave does not issue the STOP_DET interrupt if STOP_DET_INT = 1, even if the slave responds to the general call address by generating ACK. The STOP_DET interrupt is generated only when the transmitted address matches the slave address (S_ADDR).
6	S_DIS	RW	0x1	This bit controls whether I2C has its slave disabled. By default, the slave is always disabled (in reset state as well). If you need to enable it after reset, set this bit to 0. If this bit is set (slave is disabled), I2C functions only as a master and does not perform any action that requires a slave. **Note**: Software should ensure that if this bit is written with 0, then bit 0 should also be written with 0. **Value**: • 0x0 (S_EN): Slave mode is enabled. • 0x1 (S_DIS): Slave mode is disabled.
5	RESTART_EN	RW	0x1	Determine whether RESTART conditions may be sent when I2C acting as a master. Some older slaves do not support handling RESTART conditions; however, RESTART conditions are used in several I2C operations. When RESTART is disabled, the master is prohibited from performing the following functions: • Performing any high-speed mode operation. • Performing direction changes in combined format mode. • Performing a read operation with a 10-bit address. By replacing RESTART condition followed by a STOP and a subsequent START condition, split operations are broken down into multiple I2C transfers. If the above operations are performed, it will result in setting bit 6 (TX_ABORT) of the RAW_INT_STAT register. **Value**:

Bits	Field Name	RW	Reset	Description
				• 0x0 (DISABLED): Master restart disabled. • 0x1 (ENABLED): Master restart enabled.
4	ADDR_BIT_M	RW	0x1	This bit controls whether the I2C starts its transfers in 7-bit or 10-bit addressing mode when I2C acts as a master. **Value**: • 0x0 (ADDR_7BITS): Master 7-bit addressing mode • 0x1 (ADDR_10BITS): Master 10-bit addressing mode
3	ADDR_BIT_S	RW	0x1	When I2C acts as a slave, this bit controls whether the I2C responds to 7- or 10-bit addresses. **Value**: • 0x0 (ADDR_7BITS): Slave 7-Bit addressing • 0x1 (ADDR_10BITS): Slave 10-Bit addressing
2:1	SPEED	RW	0x3	These bits control at which speed the I2C operates; its setting is relevant only if one is operating the I2C in master mode. Hardware protects against illegal values being programmed by software. These bits must be programmed appropriately for slave mode, as it is used to capture correct value of spike filter as per the speed mode. This register should be programmed only with a value in the range of 1 to 3. • 0x1: Standard Mode (100 kbit/s) • 0x2: Fast Mode (≤ 400 kbit/s) or Fast Plus Mode (≤ 1000 kbit/s) • 0x3: High-speed Mode (2 Mbit/s) **Value**: • 0x1 (STANDARD): Standard Speed Mode of operation • 0x2 (FAST): Fast or Fast Plus Mode of operation • 0x3 (HIGH): High-speed Mode of operation
0	M_MODE	RW	0x1	This bit controls whether the I2C master is enabled. **Note**: Software should ensure that if this bit is written with '1', then bit 6 should also be written with '1'. **Value**: • 0x0 (DISABLED): Master mode is disabled.

Bits	Field Name	RW	Reset	Description
				• 0x1 (ENABLED): Master mode is enabled.

4.4.3.4.2 TARGET_ADDR

- **Name**: I2C Target Address Register
- **Description**: This register can be written to only when EN[0] is set to 0. You cannot change the TARGET_ADDR address dynamically.

Note:

If the software or application is aware that the I2C is not using the TARGET_ADDR address for the pending commands in the TX FIFO, it is possible to update the TARGET_ADDR address even when the TX FIFO has entries (STAT[2]= 0).

It is not necessary to perform any write to this register if the I2C is enabled as an I2C slave only.

- **Base address**: 0x4000C300 + x*0x100
- **Offset**: 0x04
- **Reset Value**: 0x00000055

Table 4-149 I2C Target Address Register

Bits	Field Name	RW	Reset	Description
31:12	RSVD	R		Reserved bits
11	SPECIAL	RW	0x0	This bit indicates whether software performs a Device-ID or General Call. **Value**: • 0x0 (DISABLED): Disable programming of GENERAL_CALL or START_BYTE transmission • 0x1 (ENABLED): Enable programming of GENERAL_CALL or START_BYTE transmission
10	TX_CTRL	RW	0x0	If bit 11 (SPECIAL) is set to 1, this bit indicates whether a General Call is to be performed by the I2C.
9:0	TARGET	RW	0x55	This is the target address for any master transaction. When transmitting a General Call, these bits are ignored. If the TARGET_ADDR and S_ADDR are the same, loopback exists but the FIFOs are shared between master and slave, so full loopback is not feasible. Only one direction loopback mode is supported (simplex), not duplex. A master cannot transmit to itself; it can transmit to a slave only.

4.4.3.4.3 S_ADDR

- **Name**: I2C Slave Address Register
- **Description**: This register sets the address for I2C when it acts as a slave.
- **Base Address**: 0x4000C300 + x*0x100
- **Offset**: 0x08
- **Reset Value**: 0x00000055

Table 4-150 I2C Slave Address Register

Bits	Field Name	RW	Reset	Description
31:10	RSVD	R		Reserved bits
9:0	S_ADDR	RW	0x55	The S_ADDR holds the slave address when the I2C is operating as a slave. For 7-bit addressing, only S_ADDR[6:0] is used. This register can be written only when the I2C interface is disabled, which corresponds to the EN[0] register being set to 0. Writes at other times have no effect. **Note**: The default values cannot be any of the reserved address locations: 0x00 to 0x07, or 0x78 to 0x7F. The correct operation of the device is not guaranteed if you program the S_ADDR or TARGET_ADDR to a reserved value. Refer to Table 4-146 for a complete list of these reserved values.

4.4.3.4.4 M_HS_ADDR

- **Name**: I2C High-Speed Master Mode Code Address Register
- **Description**: I2C High-Speed Master Mode Code Address Register
- **Base Address**: 0x4000C300 + x*0x100
- **Offset**: 0x0C
- **Reset Value**: 0x00000001

Table 4-151 I2C High-Speed Master Mode Code Address Register

Bits	Field Name	RW	Reset	Description
31:3	RSVD	R		Reserved bits
2:0	M_HS_ADDR	RW	0x1	This bit field holds the value of the I2C High-speed mode master code. High-speed-mode master codes are reserved 8-bit codes (00001xxx) that are not used for slave

Bits	Field Name	RW	Reset	Description
				addressing or other purposes. Each master has its unique master code; up to eight high-speed mode masters can be present on the same I2C bus system. Valid values are from 0 to 7. This register can be written only when the I2C interface is disabled, which corresponds to the EN[0] register being set to 0. Writes at other times have no effect.

4.4.3.4.5 DATA_CMD

- **Name**: I2C RX/TX Data Buffer and Command Register
- **Description**: This is the register the CPU writes to when filling the TX FIFO and the CPU reads from when retrieving bytes from RX FIFO.

Note:

In order to continue acknowledging reads, a read command should be written for every byte that is to be received; otherwise the I2C will stop acknowledging.

- **Base Address**: 0x4000C300 + x*0x100
- **Offset**: 0x10
- **Reset Value**: 0x00000000

Table 4-152 I2C RX/TX Data Buffer and Command Register

Bits	Field Name	RW	Reset	Description
31:11	RSVD	R		Reserved bits
10	RESTART	W	0x0	This bit controls whether a RESTART is issued before the byte is sent or received. **Value**: • 0x0 (DISABLE): Do not issue RESTART before this command • 0x1 (ENABLE): Issue RESTART before this command
9	STOP	W	0x0	This bit controls whether a STOP is issued after the byte is sent or received. **Value**: • 0x0 (DISABLE): Do not issue STOP after this command • 0x1 (ENABLE): Issue STOP after this command

Bits	Field Name	RW	Reset	Description
8	CMD	W	0x0	This bit controls whether a read or a write is performed. This bit does not control the direction when the I2C acts as a slave. It controls only the direction when it acts as a master. **Value**: • 0x0 (WRITE): Master Write command • 0x1 (READ): Master Read command
7:0	DATA	RW	0x0	This register contains the data to be transmitted or received on the I2C bus. If you are writing to this register and want to perform a read, bits 7:0 (DATA) are ignored by the I2C. However, when you read this register, these bits return the value of data received on the I2C interface.

4.4.3.4.6 SS_CLK_HCNT

- **Name**: Standard Speed I2C Clock SCL High Count Register
- **Description**: Standard Speed I2C Clock SCL High Count Register
- **Base Address**: 0x4000C300 + x*0x100
- **Offset**: 0x14
- **Reset Value**: 0x00000190

Table 4-153 Standard Speed I2C Clock SCL High Count Register

Bits	Field Name	RW	Reset	Description
31:16	RSVD	R		Reserved bits
15:0	COUNT	RW	0x0190	This register must be set before any I2C bus transaction can take place to ensure proper I/O timing. This register can be written only when the I2C interface is disabled which corresponds to the EN[0] register being set to 0. Writes at other times have no effect. The minimum valid value is 6; hardware prevents values less than this being written, and if attempted, the value will be set to 6. **Note**: This register must not be programmed to a value higher than 65525, because I2C uses a 16-bit counter to flag an

Bits	Field Name	RW	Reset	Description
				I2C bus idle condition when this counter reaches a value of COUNT + 10.

4.4.3.4.7 SS_CLK_LCNT

- **Name**: Standard Speed I2C Clock SCL Low Count Register
- **Description**: Standard Speed I2C Clock SCL Low Count Register
- **Base Address**: 0x4000C300 + x*0x100
- **Offset**: 0x18
- **Reset Value**: 0x000001D6

Table 4-154 Standard Speed I2C Clock SCL Low Count Register

Bits	Field Name	RW	Reset	Description
31:16	RSVD	R		Reserved bits
15:0	COUNT	RW	0x01D6	This register must be set before any I2C bus transaction can take place to ensure proper I/O timing. This register can be written only when the I2C interface is disabled which corresponds to the EN[0] register being set to 0. Writes at other times have no effect. The minimum valid value is 8; hardware prevents values less than this being written, and if attempted, the value will be set to 8.

4.4.3.4.8 FS_CLK_HCNT

- **Name**: Fast Mode or Fast Mode Plus I2C Clock SCL High Count Register
- **Description**: Fast Mode or Fast Mode Plus I2C Clock SCL High Count Register
- **Base Address**: 0x4000C300 + x*0x100
- **Offset**: 0x1C
- **Reset Value**: 0x0000003C

Table 4-155 Fast Mode or Fast Mode Plus I2C Clock SCL High Count Register

Bits	Field Name	RW	Reset	Description
31:16	RSVD	R		Reserved bits
15:0	COUNT	RW	0x003C	This register must be set before any I2C bus transaction can take place to ensure proper I/O timing.

Bits	Field Name	RW	Reset	Description
				This register can be written only when the I2C interface is disabled, which corresponds to the EN[0] register being set to 0. Writes at other times have no effect. The minimum valid value is 6; hardware prevents values less than this being written, and if attempted, the value will be set to 6.

4.4.3.4.9 FS_CLK_LCNT

- **Name**: Fast Mode or Fast Mode Plus I2C Clock SCL Low Count Register
- **Description**: Fast Mode or Fast Mode Plus I2C Clock SCL Low Count Register
- **Base Address**: 0x4000C300 + x*0x100
- **Offset**: 0x20
- **Reset Value**: 0x00000082

Table 4-156 Fast Mode or Fast Mode Plus I2C Clock SCL Low Count Register

Bits	Field Name	RW	Reset	Description
31:16	RSVD	R		Reserved bits
15:0	COUNT	RW	0x0082	This register must be set before any I2C bus transaction can take place to ensure proper I/O timing. This register can be written only when the I2C interface is disabled, which corresponds to the EN[0] register being set to 0. Writes at other times have no effect. The minimum valid value is 8; hardware prevents values less than this being written, and if attempted, the value will be set to 8.

4.4.3.4.10 HS_CLK_HCNT

- **Name**: High Speed I2C Clock SCL High Count Register
- **Description**: High Speed I2C Clock SCL High Count Register
- **Base Address**: 0x4000C300 + x*0x100
- **Offset**: 0x24
- **Reset Value**: 0x00000006

Table 4-157 High Speed I2C Clock SCL High Count Register

Bits	Field Name	RW	Reset	Description
31:16	RSVD	R		Reserved bits
15:0	COUNT	RW	0x0006	This register must be set before any I2C bus transaction can take place to ensure proper I/O timing. The SCL High time depends on the loading of the bus. For 100 pF loading, the SCL High time is 60 ns; for 400 pF loading, the SCL High time is 120 ns. This register can be written only when the I2C interface is disabled, which corresponds to the EN[0] register being set to 0. Writes at other times have no effect. The minimum valid value is 6; hardware prevents values less than this being written, and if attempted, the value will be set to 6.

4.4.3.4.11 HS_CLK_LCNT

- **Name**: High Speed I2C Clock SCL Low Count Register
- **Description**: High Speed I2C Clock SCL Low Count Register
- **Base Address**: 0x4000C300 + x*0x100
- **Offset**: 0x28
- **Reset Value**: 0x00000010

Table 4-158 High Speed I2C Clock SCL Low Count Register

Bits	Field Name	RW	Reset	Description
31:16	RSVD	R		Reserved bits
15:0	COUNT	RW	0x0010	This register must be set before any I2C bus transaction can take place to ensure proper I/O timing. The SCL low time depends on the loading of the bus. For 100pF loading, the SCL low time is 160 ns; for 400 pF loading, the SCL low time is 320 ns. This register can be written only when the I2C interface is disabled, which corresponds to the EN[0] register being set to 0. Writes at other times have no effect.

Bits	Field Name	RW	Reset	Description
				The minimum valid value is 8; hardware prevents values less than this being written, and if attempted, the value will be set to 8.

4.4.3.4.12 INT_STAT

- **Name**: I2C Interrupt Status Register
- **Description**: Each bit in this register has a corresponding mask bit in the INT_MASK register. These bits are cleared by reading the matching interrupt clear register. The unmasked raw versions of these bits are available in the RAW_INT_STAT register.
- **Base Address**: 0x4000C300 + x*0x100
- **Offset**: 0x2C
- **Reset Value**: 0x00000000

Table 4-159 I2C Interrupt Status Register

Bits	Field Name	RW	Reset	Description
31:15	RSVD	R		Reserved bits
14	SCL_STUCK_AT_LOW	R	0x0	See RAW_INT_STAT for a detailed description of SCL_STUCK_AT_LOW bit. **Value**: • 0x1 (ACTIVE): SCL_STUCK_AT_LOW interrupt is active. • 0x0 (INACTIVE): SCL_STUCK_AT_LOW interrupt is inactive.
13	M_HOLD	R	0x0	See RAW_INT_STAT for a detailed description of M_HOLD bit. **Value**: • 0x0 (INACTIVE): M_HOLD interrupt is inactive. • 0x1 (ACTIVE): M_HOLD interrupt is active.
12	RESTART_DET	R	0x0	See RAW_INT_STAT for a detailed description of RESTART_DET bit. **Value**: • 0x0 (INACTIVE): RESTART_DET interrupt is inactive. • 0x1 (ACTIVE): RESTART_DET interrupt is active.
11	GEN_CALL	R	0x0	See RAW_INT_STAT for a detailed description of GEN_CALL bit. **Value**:

Bits	Field Name	RW	Reset	Description
				• 0x0 (INACTIVE): GEN_CALL interrupt is inactive. • 0x1 (ACTIVE): GEN_CALL interrupt is active.
10	START_DET	R	0x0	See RAW_INT_STAT for a detailed description of START_DET bit. **Value**: • 0x0 (INACTIVE): START_DET interrupt is inactive. • 0x1 (ACTIVE): START_DET interrupt is active.
9	STOP_DET	R	0x0	See RAW_INT_STAT for a detailed description of STOP_DET bit. **Value**: • 0x0 (INACTIVE): STOP_DET interrupt is inactive. • 0x1 (ACTIVE): STOP_DET interrupt is active.
8	ACTIVITY	R	0x0	See RAW_INT_STAT for a detailed description of ACTIVITY bit. **Value**: • 0x0 (INACTIVE): ACTIVITY interrupt is inactive. • 0x1 (ACTIVE): ACTIVITY interrupt is active.
7	RX_DONE	R	0x0	See RAW_INT_STAT for a detailed description of RX_DONE bit. **Value**: • 0x0 (INACTIVE): RX_DONE interrupt is inactive. • 0x1 (ACTIVE): RX_DONE interrupt is active.
6	TX_ABORT	R	0x0	See RAW_INT_STAT for a detailed description of TX_ABORT bit. **Value**: • 0x0 (INACTIVE): TX_ABORT interrupt is inactive. • 0x1 (ACTIVE): TX_ABORT interrupt is active.
5	RD_REQ	R	0x0	See RAW_INT_STAT for a detailed description of RD_REQ bit. **Value**: • 0x0 (INACTIVE): RD_REQ interrupt is inactive. • 0x1 (ACTIVE): RD_REQ interrupt is active.
4	TX_EMPTY	R	0x0	See RAW_INT_STAT for a detailed description of TX_EMPTY bit. **Value**:

Bits	Field Name	RW	Reset	Description
				• 0x0 (INACTIVE): TX_EMPTY interrupt is inactive. • 0x1 (ACTIVE): TX_EMPTY interrupt is active.
3	TX_OVER	R	0x0	See RAW_INT_STAT for a detailed description of TX_OVER bit. **Value**: • 0x0 (INACTIVE): TX_OVER interrupt is inactive. • 0x1 (ACTIVE): TX_OVER interrupt is active.
2	RX_FULL	R	0x0	See RAW_INT_STAT for a detailed description of RX_FULL bit. **Value**: • 0x0 (INACTIVE): RX_FULL interrupt is inactive. • 0x1 (ACTIVE): RX_FULL interrupt is active.
1	RX_OVER	R	0x0	See RAW_INT_STAT for a detailed description of RX_OVER bit. **Value**: • 0x0 (INACTIVE): RX_OVER interrupt is inactive. • 0x1 (ACTIVE): RX_OVER interrupt is active.
0	RX_UNDER	R	0x0	See RAW_INT_STAT for a detailed description of RX_UNDER bit. **Value**: • 0x0 (INACTIVE): RX_UNDER interrupt is inactive. • 0x1 (ACTIVE): RX_UNDER interrupt is active.

4.4.3.4.13 INT_MASK

- **Name**: I2C Interrupt Mask Register
- **Description**: These bits mask their corresponding interrupt status bits. This register is active low; a value of 0 masks the interrupt, whereas a value of 1 unmasks the interrupt.
- **Base Address**: 0x4000C300 + x*0x100
- **Offset**: 0x30
- **Reset Value**: 0x000008FF

Table 4-160 I2C Interrupt Mask Register

Bits	Field Name	RW	Reset	Description
31:12	RSVD	R		Reserved bits

Bits	Field Name	RW	Reset	Description
11	MASK_GEN_CALL	RW	0x1	This bit masks the GEN_CALL interrupt in INT_STAT register. **Value**: • 0x0 (ENABLED): GEN_CALL interrupt is masked. • 0x1 (DISABLED): GEN_CALL interrupt is unmasked.
10	MASK_START_DET	RW	0x0	This bit masks the START_DET interrupt in INT_STAT register. **Value**: • 0x0 (ENABLED): START_DET interrupt is masked. • 0x1 (DISABLED): START_DET interrupt is unmasked.
9	MASK_STOP_DET	RW	0x0	This bit masks the STOP_DET interrupt in INT_STAT register. **Value**: • 0x0 (ENABLED): STOP_DET interrupt is masked. • 0x1 (DISABLED): STOP_DET interrupt is unmasked.
8	MASK_ACTIVITY	RW	0x0	This bit masks the ACTIVITY interrupt in INT_STAT register. **Value**: • 0x0 (ENABLED): ACTIVITY interrupt is masked. • 0x1 (DISABLED): ACTIVITY interrupt is unmasked.
7	MASK_RX_DONE	RW	0x1	This bit masks the RX_DONE interrupt in INT_STAT register. **Value**: • 0x0 (ENABLED): RX_DONE interrupt is masked. • 0x1 (DISABLED): RX_DONE interrupt is unmasked.
6	MASK_TX_ABORT	RW	0x1	This bit masks the TX_ABORT interrupt in INT_STAT register. **Value**: • 0x0 (ENABLED): TX_ABORT interrupt is masked. • 0x1 (DISABLED): TX_ABORT interrupt is unmasked.
5	MASK_RD_REQ	RW	0x1	This bit masks the RD_REQ interrupt in INT_STAT register. **Value**: • 0x0 (ENABLED): RD_REQ interrupt is masked. • 0x1 (DISABLED): RD_REQ interrupt is unmasked.
4	MASK_TX_EMPTY	RW	0x1	This bit masks the TX_EMPTY interrupt in INT_STAT register. **Value**: • 0x0 (ENABLED): TX_EMPTY interrupt is masked. • 0x1 (DISABLED): TX_EMPTY interrupt is unmasked.
3	MASK_TX_OVER	RW	0x1	This bit masks the TX_OVER interrupt in INT_STAT register. **Value**:

Bits	Field Name	RW	Reset	Description
				• 0x0 (ENABLED): TX_OVER interrupt is masked. • 0x1 (DISABLED): TX_OVER interrupt is unmasked.
2	MASK_RX_FULL	RW	0x1	This bit masks the RX_FULL interrupt in INT_STAT register. **Value**: • 0x0 (ENABLED): RX_FULL interrupt is masked. • 0x1 (DISABLED): RX_FULL interrupt is unmasked.
1	MASK_RX_OVER	RW	0x1	This bit masks the RX_OVER interrupt in INT_STAT register. **Value**: • 0x0 (ENABLED): RX_OVER interrupt is masked. • 0x1 (DISABLED): RX_OVER interrupt is unmasked.
0	MASK_RX_UNDER	RW	0x1	This bit masks the RX_UNDER interrupt in INT_STAT register. **Value**: • 0x0 (ENABLED): RX_UNDER interrupt is masked. • 0x1 (DISABLED): RX_UNDER interrupt is unmasked.

4.4.3.4.14 RAW_INT_STAT

- **Name**: I2C Raw Interrupt Status Register
- **Description**: Unlike the INT_STAT register, these bits are not masked so they always show the true status of the I2C.
- **Base Address**: 0x4000C300 + x*0x100
- **Offset**: 0x34
- **Reset Value**: 0x00000000

Table 4-161 I2C Raw Interrupt Status Register

Bits	Field Name	RW	Reset	Description
31:15	RSVD	R		Reserved bits
14	RAW_SCL_STUCK_AT_LOW	R	0x0	Indicates whether the SCL Line is stuck at low for the SCL_STUCK_LOW_TIMEOUT number of ic_clk periods **Value**: • 0x0 (ACTIVE): RAW_SCL_STUCK_AT_LOW interrupt is active. • 0x1 (ACTIVE): RAW_SCL_STUCK_AT_LOW interrupt is inactive.
13	RAW_M_HOLD	R	0x0	Indicate whether the master is holding the bus and TX FIFO is empty.

Bits	Field Name	RW	Reset	Description
				Value: • 0x1 (ACTIVE): RAW_M_HOLD interrupt is active • 0x0 (INACTIVE): RAW_M_HOLD interrupt is inactive
12	RAW_RESTART_DET	R	0x0	Indicate whether a RESTART condition has occurred on the I2C interface when I2C is operating in Slave mode and the slave is being addressed. **Note:** However, in High-speed Mode, the RESTART comes before the address field as per the I2C protocol. In this case, the slave is not the addressed slave when the RESTART is issued, therefore I2C does not generate the RESTART_DET interrupt. **Value**: • 0x0 (INACTIVE): RAW_RESTART_DET interrupt is inactive • 0x1 (ACTIVE): RAW_RESTART_DET interrupt is active
11	RAW_GEN_CALL	R	0x0	Set only when a General Call address is received and it is acknowledged. It stays set until it is cleared either by disabling I2C or when the CPU reads bit 0 of the CLR_GEN_CALL register. I2C stores the received data in the RX buffer. **Value**: • 0x0 (INACTIVE): RAW_GEN_CALL interrupt is inactive • 0x1 (ACTIVE): RAW_GEN_CALL interrupt is active
10	RAW_START_DET	R	0x0	Indicate whether a START or RESTART condition has occurred on the I2C interface regardless of whether I2C is operating in slave or master mode. **Value**: • 0x0 (INACTIVE): RAW_START_DET interrupt is inactive • 0x1 (ACTIVE): RAW_START_DET interrupt is active
9	RAW_STOP_DET	R	0x0	Indicate whether a STOP condition has occurred on the I2C interface regardless of whether I2C is operating in slave or master mode. In Slave Mode: If CTRL[7] = 0x1 (STOP_DET_INT), the STOP_DET interrupt will be issued only if slave is addressed. **Note**:

Bits	Field Name	RW	Reset	Description
				During a general call address, this slave does not issue a STOP_DET interrupt if STOP_DET_INT = 0x1, even if the slave responds to the general call address by generating ACK. The STOP_DET interrupt is generated only when the transmitted address matches the slave address (S_ADDR). If CTRL[7] = 0x0 (STOP_DET_INT), the STOP_DET interrupt is issued irrespective of whether it is being addressed. **In Master Mode**: • If CTRL[10] = 0x1, the STOP_DET interrupt will be issued only if Master is active. • If CTRL[10] = 0x0, the STOP_DET interrupt will be issued irrespective of whether master is active or not. **Value**: • 0x0 (INACTIVE): RAW_STOP_DET interrupt is inactive. • 0x1 (ACTIVE): RAW_STOP_DET interrupt is active.
8	RAW_ACTIVITY	R	0x0	This bit captures I2C activity and stays set until it is cleared. There are four ways to clear it: • Disabling the I2C • Reading the CLR_ACTIVITY register • Reading the CLR_INT register • System reset Once this bit is set, it stays set unless one of the four methods is used to clear it. Even if the I2C module is idle, this bit remains set until cleared, indicating that there was activity on the bus. **Value**: • 0x0 (INACTIVE): RAW_ACTIVITY interrupt is inactive. • 0x1 (ACTIVE): RAW_ACTIVITY interrupt is active.
7	RAW_RX_DONE	R	0x0	When the I2C is acting as a slave-transmitter, this bit is set to 1 if the master does not acknowledge a transmitted byte. This occurs on the last byte of the transmission, indicating that the transmission is done. **Value**: • 0x0 (INACTIVE): RAW_RX_DONE interrupt is inactive.

Bits	Field Name	RW	Reset	Description
				• 0x1 (ACTIVE): RAW_RX_DONE interrupt is active.
6	RAW_TX_ABORT	R	0x0	This bit indicates if I2C, as an I2C transmitter, is unable to complete the intended actions on the contents of the transmit FIFO. This situation can occur both as an I2C master or an I2C slave, and is referred to as a 'transmit abort'. When this bit is set to 1, the TX_ABORT_SRC register indicates the reason why the transmit abort takes place. **Note**: The I2C flushes/resets/empties only the TX_FIFO whenever there is a transmit abort caused by any of the events tracked by the TX_ABORT_SRC register. The TX FIFO remains in this flushed state until the register CLR_TX_ABORT is read. Once this read is performed, the TX FIFO is then ready to accept more data bytes from the APB interface. **Value**: • 0x0 (INACTIVE): RAW_TX_ABORT interrupt is inactive. • 0x1 (ACTIVE): RAW_TX_ABORT interrupt is active.
5	RAW_RD_REQ	R	0x0	This bit is set to 1 when I2C is acting as a slave and a I2C master is attempting to read data from I2C. The I2C holds the I2C bus in a wait state (SCL=0) until this interrupt is serviced, which means that the slave has been addressed by a remote master that is asking for data to be transferred. The processor must respond to this interrupt and then write the requested data to the DATA_CMD register. This bit is set to 0 just after the processor reads the CLR_RD_REQ register. **Value**: • 0x0 (INACTIVE): RAW_RD_REQ interrupt is inactive. • 0x1 (ACTIVE): RAW_RD_REQ interrupt is active.
4	RAW_TX_EMPTY	R	0x0	The behavior of the TX_EMPTY interrupt status differs based on the TX_EMPTY_CTRL selection in the CTRL register. • When TX_EMPTY_CTRL = 0: This bit is set to 1 when the transmit buffer is at or below the threshold value set in the TX_FIFO_THD register. • When TX_EMPTY_CTRL = 1: This bit is set to 1 when the transmit buffer is at or below the threshold value set in the TX_FIFO_THD register and

Bits	Field Name	RW	Reset	Description
				the transmission of the address/data from the internal shift register for the most recently popped command is completed. It is automatically cleared by hardware when the buffer level goes above the threshold. When EN[0] is set to 0, the TX FIFO is flushed and held in reset. Therefore, the TX FIFO looks like it has no data within it, so this bit is set to 1, provided there is any activity in the master or slave state machines. When there is no longer any activity, then with EN[0] = 0, this bit is set to 0. **Value**: • 0x0 (INACTIVE): RAW_TX_EMPTY interrupt is inactive. • 0x1 (ACTIVE): RAW_TX_EMPTY interrupt is active.
3	RAW_TX_OVER	R	0x0	Set this bit during transmit if the transmit buffer is filled to 8 and the processor attempts to issue another I2C command by writing to the DATA_CMD register. When I2C module is disabled, this bit keeps its level until the master or slave state machines go into idle; when EN[0] goes to 0, this interrupt is cleared. **Value**: • 0x0 (INACTIVE): RAW_TX_OVER interrupt is inactive • 0x1 (ACTIVE): RAW_TX_OVER interrupt is active
2	RAW_RX_FULL	R	0x0	Set this bit when the receive buffer reaches or goes above the RX_FIFO_THD threshold in the RX_FIFO_THD register. It is automatically cleared by hardware when buffer level goes below the threshold. If I2C module is disabled (EN[0] = 0), the RX FIFO is flushed and held in reset; therefore the RX FIFO is not full. So this bit is cleared once the EN bit 0 is programmed with a 0, regardless of the activity that continues. **Value**: • 0x0 (INACTIVE): RAW_RX_FULL interrupt is inactive. • 0x1 (ACTIVE): RAW_RX_FULL interrupt is active.
1	RAW_RX_OVER	R	0x0	Set this bit if the receive buffer is completely filled to 8 and an additional byte is received from an external I2C

Bits	Field Name	RW	Reset	Description
				device. The I2C acknowledges this, but any data bytes received after the FIFO is full is lost. If I2C module is disabled (EN[0]=0), this bit keeps its level until the master or slave state machines go into idle, and when EN[0] goes to 0, this interrupt is cleared. **Value**: • 0x0 (INACTIVE): RAW_RX_OVER interrupt is inactive. • 0x1 (ACTIVE): RAW_RX_OVER interrupt is active.
0	RAW_RX_UNDER	R	0x0	Set this bit if the processor attempts to read the receive buffer when it is empty by reading from the DATA_CMD register. If the module is disabled (EN[0]=0), this bit keeps its level until the master or slave state machines go into idle, and when EN[0] goes to 0, this interrupt is cleared. **Value**: • 0x0 (INACTIVE): RAW_RX_UNDER interrupt is inactive. • 0x1 (ACTIVE): RAW_RX_UNDER interrupt is active.

4.4.3.4.15 RX_FIFO_THD

- **Name**: I2C Receive FIFO Threshold Register
- **Description**: I2C Receive FIFO Threshold Register
- **Base Address**: 0x4000C300 + x*0x100
- **Offset**: 0x38
- **Reset Value**: 0x00000000

Table 4-162 I2C Receive FIFO Threshold Register

Bits	Field Name	RW	Reset	Description
31:8	RSVD	R		Reserved bits
7:0	THD	RW	0x00	Receive FIFO Threshold Level. Control the level of entries (or above) that triggers the RAW_RX_FULL interrupt (bit 2 in RAW_INT_STAT register). The valid range is 0–255, with the additional restriction that hardware does not allow this value to be set to a value larger than the depth of the buffer. If an attempt is made to

Bits	Field Name	RW	Reset	Description
				do that, the actual value set will be the maximum depth of the buffer. A value of 0 sets the threshold for 1 entry, and a value of 255 sets the threshold for 256 entries.

4.4.3.4.16 TX_FIFO_THD

- **Name**: I2C Transmit FIFO Threshold Register
- **Description**: I2C Transmit FIFO Threshold Register
- **Base Address**: 0x4000C300 + x*0x100
- **Offset**: 0x3C
- **Reset Value**: 0x00000000

Table 4-163 I2C Transmit FIFO Threshold Register

Bits	Field Name	RW	Reset	Description
31:8	RSVD	R		Reserved bits
7:0	THD	RW	0x00	Transmit FIFO Threshold Level. Control the level of entries (or below) that trigger the RAW_TX_EMPTY interrupt (bit 4 in RAW_INT_STAT register). The valid range is 0–255, with the additional restriction that it may not be set to value larger than the depth of the buffer. If an attempt is made to do that, the actual value set will be the maximum depth of the buffer. A value of 0 sets the threshold for 0 entries, and a value of 255 sets the threshold for 255 entries.

4.4.3.4.17 CLR_INT

- **Name**: Clear Combined and Individual Interrupt Register
- **Description**: Clear Combined and Individual Interrupt Register
- **Base Address**: 0x4000C300 + x*0x100
- **Offset**: 0x40
- **Reset Value**: 0x00000000

Table 4-164 Clear Combined and Individual Interrupt Register

Bits	Field Name	RW	Reset	Description
31:1	RSVD	R		Reserved bits
0	CLR_INT	R	0x0	Read this register to clear the combined interrupt, all individual interrupts, and the TX_ABORT_SRC register. This bit does not clear hardware clearable interrupts but software clearable interrupts.

4.4.3.4.18 CLR_RX_UNDER

- **Name**: Clear RX_UNDER Interrupt Register
- **Description**: Clear RX_UNDER Interrupt Register
- **Base Address**: 0x4000C300 + x*0x100
- **Offset**: 0x44
- **Reset Value**: 0x00000000

Table 4-165 Clear RX_UNDER Interrupt Register

Bits	Field Name	RW	Reset	Description
31:1	RSVD	R		Reserved bits
0	CLR_RX_UNDER	R	0x0	Read this register to clear the RAW_RX_UNDER interrupt (bit 0) of the RAW_INT_STAT register.

4.4.3.4.19 CLR_RX_OVER

- **Name**: Clear RX_OVER Interrupt Register
- **Description**: Clear RX_OVER Interrupt Register
- **Base Address**: 0x4000C300 + x*0x100
- **Offset**: 0x48
- **Reset Value**: 0x00000000

Table 4-166 Clear RX_OVER Interrupt Register

Bits	Field Name	RW	Reset	Description
31:1	RSVD	R		Reserved bits
0	CLR_RX_OVER	R	0x0	Read this register to clear the RAW_RX_OVER interrupt (bit 1) of the RAW_INT_STAT register.

4.4.3.4.20 CLR_TX_OVER

- **Name**: Clear TX_OVER Interrupt Register
- **Description**: Clear TX_OVER Interrupt Register
- **Base Address**: 0x4000C300 + x*0x100
- **Offset**: 0x4C
- **Reset Value**: 0x00000000

Table 4-167 Clear TX_OVER Interrupt Register

Bits	Field Name	RW	Reset	Description
31:1	RSVD	R		Reserved bits
0	CLR_TX_OVER	R	0x0	Read this register to clear the RAW_TX_OVER interrupt (bit 3) of the RAW_INT_STAT register.

4.4.3.4.21 CLR_RD_REQ

- **Name**: Clear RD_REQ Interrupt Register
- **Description**: Clear RD_REQ Interrupt Register
- **Base Address**: 0x4000C300 + x*0x100
- **Offset**: 0x50
- **Reset Value**: 0x00000000

Table 4-168 Clear RD_REQ Interrupt Register

Bits	Field Name	RW	Reset	Description
31:1	RSVD	R		Reserved bits
0	CLR_RD_REQ	R	0x0	Read this register to clear the RAW_RD_REQ interrupt (bit 5) of the RAW_INT_STAT register.

4.4.3.4.22 CLR_TX_ABORT

- **Name**: Clear TX_ABORT Interrupt Register
- **Description**: Clear TX_ABORT Interrupt Register
- **Base Address**: 0x4000C300 + x*0x100
- **Offset**: 0x54
- **Reset Value**: 0x00000000

Table 4-169 Clear TX_ABORT Interrupt Register

Bits	Field Name	RW	Reset	Description
31:1	RSVD	R		Reserved bits

Bits	Field Name	RW	Reset	Description
0	CLR_TX_ABORT	R	0x0	Read this register to clear the RAW_TX_ABORT interrupt (bit 6) of the RAW_INT_STAT register, and the TX_ABORT_SRC register.

4.4.3.4.23 CLR_RX_DONE

- **Name**: Clear RX_DONE Interrupt Register
- **Description**: Clear RX_DONE Interrupt Register
- **Base Address**: 0x4000C300 + x*0x100
- **Offset**: 0x58
- **Reset Value**: 0x00000000

Table 4-170 Clear RX_DONE Interrupt Register

Bits	Field Name	RW	Reset	Description
31:1	RSVD	R		Reserved bits
0	CLR_RX_DONE	R	0x0	Read this register to clear the RAW_RX_DONE interrupt (bit 7) of the RAW_INT_STAT register.

4.4.3.4.24 CLR_ACTIVITY

- **Name**: Clear ACTIVITY Interrupt Register
- **Description**: Clear ACTIVITY Interrupt Register
- **Base Address**: 0x4000C300 + x*0x100
- **Offset**: 0x5C
- **Reset Value**: 0x00000000

Table 4-171 Clear ACTIVITY Interrupt Register

Bits	Field Name	RW	Reset	Description
31:1	RSVD	R		Reserved bits
0	CLR_ACTIVITY	R	0x0	Reading this register clears the ACTIVITY interrupt if the I2C is not active anymore. If the I2C module is still active on the bus, the ACTIVITY interrupt bit continues to be set. It is automatically cleared by hardware if the module is disabled and if there is no further activity on the bus. The value read from this register to get status of the RAW_ACTIVITY interrupt (bit 8) of the RAW_INT_STAT register.

4.4.3.4.25 CLR_STOP_DET

- **Name**: Clear STOP_DET Interrupt Register
- **Description**: Clear STOP_DET Interrupt Register
- **Base Address**: 0x4000C300 + x*0x100
- **Offset**: 0x60
- **Reset Value**: 0x00000000

Table 4-172 Clear STOP_DET Interrupt Register

Bits	Field Name	RW	Reset	Description
31:1	RSVD	R		Reserved bits
0	CLR_STOP_DET	R	0x0	Read this register to clear the RAW_STOP_DET interrupt (bit 9) of the RAW_INT_STAT register.

4.4.3.4.26 CLR_START_DET

- **Name**: Clear START_DET Interrupt Register
- **Description**: Clear START_DET Interrupt Register
- **Base Address**: 0x4000C300 + x*0x100
- **Offset**: 0x64
- **Reset Value**: 0x00000000

Table 4-173 Clear START_DET Interrupt Register

Bits	Field Name	RW	Reset	Description
31:1	RSVD	R		Reserved bits
0	CLR_START_DET	R	0x0	Read this register to clear the RAW_START_DET interrupt (bit 10) of the RAW_INT_STAT register.

4.4.3.4.27 CLR_GEN_CALL

- **Name**: Clear GEN_CALL Interrupt Register
- **Description**: Clear GEN_CALL Interrupt Register
- **Base Address**: 0x4000C300 + x*0x100
- **Offset**: 0x68
- **Reset Value**: 0x00000000

Table 4-174 Clear GEN_CALL Interrupt Register

Bits	Field Name	RW	Reset	Description
31:1	RSVD	R		Reserved bits
0	CLR_GEN_CALL	R	0x0	Read this register to clear the RAW_GEN_CALL interrupt (bit 11) of RAW_INT_STAT register.

4.4.3.4.28 EN

- **Name**: I2C ENABLE Register
- **Description**: I2C enable register
- **Base Address**: 0x4000C300 + x*0x100
- **Offset**: 0x6C
- **Reset Value**: 0x00000000

Table 4-175 I2C ENABLE Register

Bits	Field Name	RW	Reset	Description
31:4	RSVD	R		Reserved bits
3	SDA_STUCK_RECOVERY_ENABLE	RW	0x0	If SDA is stuck at low, which is indicated by the TX_ABORT interrupt (TX_ABORT_SRC[17]), this bit is used as a control knob to initiate the SDA Recovery Mechanism (that is, send at most 9 SCL clocks and STOP to release the SDA line) and then this bit gets auto clear. **Value**: • 0x1 (SDA_STUCK_RECOVERY_ENABLED): Master initiates the SDA stuck at low recovery mechanism. • 0x0 (SDA_STUCK_RECOVERY_DISABLED): Master disables the SDA stuck at low recovery mechanism.
2	TX_CMD_BLOCK	RW	0x0	In Master mode: **Value**: • 0x0 (NOT_BLOCKED): TX Command execution not blocked • 0x1 (BLOCKED): TX Command execution blocked
1	ABORT	RW	0x0	When set, the controller initiates the transfer abort. The software can abort the I2C transfer in master mode by setting this bit. The software can set this bit only when EN is already set; otherwise, the controller ignores any write to ABORT bit. The software cannot clear the ABORT bit

Bits	Field Name	RW	Reset	Description
				once set. In response to an ABORT, the controller issues a STOP and flushes the TX FIFO after completing the current transfer, then sets the TX_ABORT interrupt after the abort operation. The ABORT bit is cleared automatically after the abort operation. **Value**: • 0x0 (DISABLED): ABORT operation not in progress • 0x1 (ENABLED): ABORT operation in progress
0	ACTIVITY	RW	0x0	Control whether the I2C is enabled. Software can disable I2C while it is active. However, it is important that care should be taken to ensure that I2C is disabled properly. When I2C is disabled, the following occurs: • The TX FIFO and RX FIFO get flushed. • Status bits in the INT_STAT register are still active until I2C goes into IDLE state. If the I2C module is transmitting, it stops as well as deletes the contents of the transmit buffer after the current transfer is completed. If the module is receiving, the I2C stops the current transfer at the end of the current byte and does not acknowledge the transfer. **Value**: • 0x0 (DISABLED): I2C is disabled. • 0x1 (ENABLED): I2C is enabled.

4.4.3.4.29 STAT

- **Name**: I2C STATUS Register
- **Description**: This is a read-only register used to indicate the current transfer status and FIFO status. The status register may be read at any time. None of the bits in this register requests an interrupt.

  When the I2C is disabled by writing 0 in Bit 0 of the EN register:

  - Bits 1 and 2 are set to 1
  - Bits 3 and 6 are set to 0
  - When the master or slave state machine goes idle and EN[0]=0:
  - Bits 5 and 6 are set to 0

- **Base Address**: 0x4000C300 + x*0x100
- **Offset**: 0x70
- **Reset Value**: 0x00000006

Table 4-176 I2C STATUS Register

Bits	Field Name	RW	Reset	Description
31:12	RSVD	R		Reserved bits
11	SDA_STUCK_NOT_RECOVERED	R	0x0	This bit indicates that SDA stuck at low is not recovered after the recovery mechanism. In Slave mode, this register bit is not applicable. **Value**: • 0x1 (ACTIVE): SDA Stuck at low is recovered after recovery mechanism. • 0x0 (INACTIVE): SDA Stuck at low is not recovered after recovery mechanism.
10:7	RSVD	R		Reserved bits
6	S_ACTIVITY	R	0x0	Slave Activity Status. When the Slave is not in the IDLE state, this bit is set. **Value**: • 0x0 (INACTIVE): Slave is idle. • 0x1 (ACTIVE): Slave is not idle.
5	M_ACTIVITY	R	0x0	Master Activity Status. When the Master is not in the IDLE state, this bit is set. **Note**: STAT[0], ACTIVITY bit, is the OR of S_ACTIVITY and M_ACTIVITY Y bits. **Value**: • 0x0 (INACTIVE): Master is idle. • 0x1 (ACTIVE): Master not idle.
4	RX_FIFO_CF	R	0x0	Receive FIFO Completely Full. When the receive FIFO is completely full, this bit is set. When the receive FIFO contains one or more empty location, this bit is cleared. **Value**: • 0x0 (NOT_FULL): RX FIFO is not full. • 0x1 (FULL): RX FIFO is full.

Bits	Field Name	RW	Reset	Description
3	RX_FIFO_NE	R	0x0	Receive FIFO Not Empty. This bit is set when the receive FIFO contains one or more entries; it is cleared when the receive FIFO is empty. **Value**: • 0x0 (EMPTY): RX FIFO is empty. • 0x1 (NOT_EMPTY): RX FIFO not empty.
2	TX_FIFO_CE	R	0x1	Transmit FIFO Completely Empty. When the transmit FIFO is completely empty, this bit is set. When it contains one or more valid entries, this bit is cleared. This bit field does not request an interrupt. **Value**: • 0x0 (NOT_EMPTY): TX FIFO not empty. • 0x1 (EMPTY): TX FIFO is empty.
1	TX_FIFO_NF	R	0x1	Transmit FIFO Not Full. Set when the transmit FIFO contains one or more empty locations, and is cleared when the FIFO is full. **Value**: • 0x0 (FULL): TX FIFO is full. • 0x1 (NOT_FULL): TX FIFO not full.
0	ACTIVITY	R	0x0	I2C Activity Status **Value**: • 0x0 (INACTIVE): I2C is idle. • 0x1 (ACTIVE): I2C is active.

4.4.3.4.30 TX_FIFO_LV

- **Name**: I2C Transmit FIFO Level Register
- **Description**: This register contains the number of valid data entries in the transmit FIFO buffer. It is cleared whenever:
  - The I2C is disabled.
  - If RAW_TX_ABORT bit is set in the RAW_INT_STAT register, the slave bulk transmit mode is aborted.
  - The register increments whenever data is placed into the transmit FIFO and decrements when data is taken from the transmit FIFO.
- **Base Address**: 0x4000C300 + x*0x100
- **Offset**: 0x74

- **Reset Value**: 0x00000000

Table 4-177 I2C Transmit FIFO Level Register

Bits	Field Name	RW	Reset	Description
31:4	RSVD	R		Reserved bits
3:0	TLEVEL	R	0x0	Transmit FIFO Level. It contains the number of valid data entries in the transmit FIFO.

4.4.3.4.31 RX_FIFO_LV

- **Name**: I2C Receive FIFO Level Register
- **Description**: This register contains the number of valid data entries in the receive FIFO buffer. It is cleared whenever:
  - The I2C is disabled.
  - There is a transmit abort caused by any of the events tracked in TX_ABORT_SRC.
  - The register increments whenever data is placed into the receive FIFO and decrements when data is taken from the receive FIFO.
- **Base Address**: 0x4000C300 + x*0x100
- **Offset**: 0x78
- **Reset Value**: 0x00000000

Table 4-178 I2C Receive FIFO Level Register

Bits	Field Name	RW	Reset	Description
31:4	RSVD	R		Reserved bits
3:0	LEVEL	R	0x0	Receive FIFO Level. It contains the number of valid data entries in the receive FIFO.

4.4.3.4.32 SDA_HOLD

- **Name**: I2C SDA Hold Time Length Register
- **Description**: I2C SDA Hold Time Length Register
  - The bits [15:0] of this register are used to control the hold time of SDA during transmit in both slave and master mode (after SCL goes from HIGH to LOW).
  - The bits [23:16] of this register are used to extend the SDA transition (if any) whenever SCL is HIGH in the receiver in either master or slave mode.
  - Writes to this register succeed only when EN[0]=0.

- The values in this register are in units of clock period. The value programmed in TX_HOLD must be greater than the minimum hold time in each mode one cycle in master mode, seven cycles in slave mode for the value to be implemented.
- The programmed SDA hold time during transmit (TX_HOLD) cannot exceed at any time the duration of the low part of SCL. Therefore the programmed value cannot be larger than N_SCL_LOW minus 2, where N_SCL_LOW is the duration of the low part of the SCL period measured in clock cycles.

- **Base Address**: 0x4000C300 + x*0x100
- **Offset**: 0x7C
- **Reset Value**: 0x00000001

Table 4-179 I2C SDA Hold Time Length Register

Bits	Field Name	RW	Reset	Description
31:24	RSVD	R		Reserved bits
23:16	RX_HOLD	RW	0x0	Set the required SDA hold time in units of clock period, when I2C acts as a receiver.
15:0	TX_HOLD	RW	0x1	Set the required SDA hold time in units of clock period, when I2C acts as a transmitter.

4.4.3.4.33 TX_ABORT_SRC

- **Name**: I2C Transmit Abort Source Register
- **Description**: I2C Transmit Abort Source Register
  - This register has 32 bits that indicate the source of the TX_ABORT bit. Except for Bit 9, this register is cleared whenever the CLR_TX_ABORT register bit or the CLR_INT register is read. To clear Bit 9, the source of ABORT_SBYTE_NORSTRT must be fixed first; RESTART must be enabled (CTRL[5] = 1), the SPECIAL bit must be cleared (TARGET_ADDR[11]), or the TX_CTRL bit must be cleared (TARGET_ADDR[10]).
  - Once the source of the ABORT_SBYTE_NORSTRT is fixed, this bit can be cleared in the same manner as other bits in this register. If the source of the ABORT_SBYTE_NORSTRT is not fixed before attempting to clear this bit, Bit 9 clears for one cycle and is then re-asserted.

- **Base Address**: 0x4000C300 + x*0x100
- **Offset**: 0x80
- **Reset Value**: 0x00000000

Table 4-180 I2C Transmit Abort Source Register

Bits	Field Name	RW	Reset	Description
31:23	TX_FLUSH_CNT	R	0x0	This field indicates the number of TX FIFO Data Commands which are flushed due to TX_ABORT interrupt. It is cleared whenever I2C is disabled. Role of I2C: Master-Transmitter or Slave-Transmitter
22:18	RSVD	R		Reserved bits
17	ABRT_SDA_STUCK_AT_LOW	R	0x0	This is a master-mode-only bit. Master detects the SDA Stuck at low for the SDA_STUCK_AT_LOW_TIMEOUT value of ic_clk cycles. **Value**: • 0x1 (ACTIVE): This abort is generated because of SDA stuck at low for SDA_STUCK_AT_LOW_TIMEOUT value of ic_clk cycles. • 0x0 (INACTIVE): This abort is not generated.
16	ABORT_USER_ABORT	R	0x0	This is a master-mode-only bit. Master has detected the transfer abort (EN[1]) event. Role of I2C: Master-Transmitter **Value**: • 0x0 (ABORT_USER_ABORT_VOID): Transfer abort not detected by master • 0x1 (ABORT_USER_ABORT_GENERATED): Transfer abort detected by master
15	ABORT_SLVRD_INTX	R	0x0	When the processor side responds to a slave mode request for data to be transmitted to a remote master, users write a 1 in CMD (bit 8) of DATA_CMD register. Role of I2C: Slave-Transmitter **Value**: • 0x0 (ABORT_SLVRD_INTX_VOID): Slave trying to transmit to remote master in read mode – scenario not present • 0x1 (ABORT_SLVRD_INTX_GENERATED): Slave trying to transmit to remote master in read mode
14	ABORT_S_ARBLOST	R	0x0	This field indicates that a Slave has lost the bus while transmitting data to a remote master. TX_ABORT_SRC[12] is set at the same time. **Note:**

Bits	Field Name	RW	Reset	Description
				Even though the slave never 'owns' the bus, something could go wrong on the bus. This is a fail – safe check. For instance, during a data transmission at the low-to-high transition of SCL, if what is on the data bus is not what is supposed to be transmitted, then I2C no longer owns the bus. Role of I2C: Slave-Transmitter **Value**: • 0x0 (ABORT_S_ARBLOST_VOID): Slave lost arbitration to remote master- scenario not present • 0x1 (ABORT_S_ARBLOST_GENERATED): Slave lost arbitration to remote master
13	ABORT_SLVFLUSH_TXFIFO	R	0x0	This field specifies that the Slave has received a read command and some data exists in the TX FIFO, so the slave issues a TX_ABORT interrupt to flush old data in TX FIFO. Role of I2C: Slave-Transmitter **Value**: • 0x0 (ABORT_SLVFLUSH_TXFIFO_VOID): Slave flushes existing data in TX-FIFO upon getting read command- scenario not present • 0x1 (ABORT_SLVFLUSH_TXFIFO_GENERATED): Slave flushes existing data in TX-FIFO upon getting read command
12	ABORT_LOST	R	0x0	This field specifies that the Master has lost arbitration, or if TX_ABORT_SRC[14] is also set, then the slave transmitter has lost arbitration. Role of I2C: Master-Transmitter or Slave-Transmitter **Value**: • 0x0 (ABORT_LOST_VOID): Master or Slave-Transmitter lost arbitration- scenario not present • 0x1 (ABORT_LOST_GENERATED): Master or Slave-Transmitter lost arbitration
11	ABORT_MASTER_DIS	R	0x0	This field indicates that users try to initiate a Master operation with the Master mode disabled. Role of I2C: Master-Transmitter or Master-Receiver **Value**:

Bits	Field Name	RW	Reset	Description
				• 0x0 (ABORT_MASTER_DIS_VOID): Users initiates master operation when Master is disabled- scenario not present • 0x1 (ABORT_MASTER_DIS_GENERATED): Users initiates master operation when Master is disabled
10	ABORT_10B_RD_NORSTR	R	0x0	This field indicates that the restart is disabled (RESTART_EN bit (CTRL[5]) =0) and the master sends a read command in 10-bit addressing mode. Role of I2C: Master-Receiver **Value**: • 0x0 (ABRT_10B_RD_VOID): Master not trying to read in 10-bit addressing mode when RESTART disabled • 0x1 (ABRT_10B_RD_GENERATED): Master trying to read in 10-bit addressing mode when RESTART disabled
9	ABORT_SBYTE_NORSTRT	R	0x0	To clear Bit 9, the source of the ABORT_SBYTE_NORSTRT must be fixed first; RESTART must be enabled (RESTART_EN bit CTRL[5]=1), the SPECIAL bit must be cleared (TARGET_ADDR[11]), or the TX_CTRL bit must be cleared (TARGET_ADDR[10]). Once the source of the ABORT_SBYTE_NORSTRT is fixed, then this bit can be cleared in the same manner as other bits in this register. If the source of the ABORT_SBYTE_NORSTRT is not fixed before attempting to clear this bit, bit 9 clears for one cycle and then gets reasserted. When this field is set to 1, RESTART is disabled (RESTART_EN bit (CTRL[5]) =0) and users are trying to send a START BYTE. Role of I2C: Master **Value**: • 0x0 (ABORT_SBYTE_NORSTRT_VOID): Users trying to send START BYTE when RESTART is disabled- scenario not present • 0x1 (ABORT_SBYTE_NORSTRT_GENERATED): Users trying to send START BYTE when RESTART is disabled
8	ABORT_HS_NORSTRT	R	0x0	This field indicates that the RESTART is disabled (RESTART_EN bit (CTRL[5]) =0) and users are trying to use the master to transfer data in High-speed mode. Role of I2C: Master-Transmitter or Master-Receiver **Value**:

Bits	Field Name	RW	Reset	Description
				• 0x0 (ABORT_HS_NORSTRT_VOID): Users trying to switch Master to High-speed mode when RESTART is disabled- scenario not present • 0x1 (ABORT_HS_NORSTRT_GENERATED): Users trying to switch Master to High-speed mode when RESTART is disabled
7	ABORT_SBYTE_ACKDET	R	0x0	This field indicates that the Master has sent a START Byte and the START Byte was acknowledged (wrong behavior). Role of I2C: Master **Value**: • 0x0 (ABORT_SBYTE_ACKDET_VOID): ACK detected for START byte- scenario not present • 0x1 (ABORT_SBYTE_ACKDET_GENERATED): ACK detected for START byte
6	ABORT_HS_ACKDET	R	0x0	This field indicates that the Master is in High-speed mode and the High-speed Master code was acknowledged (wrong behavior). Role of I2C: Master **Value**: • 0x0 (ABRT_HS_ACK_VOID): HS Master code ACKed in High-speed Mode- scenario not present • 0x1 (ABRT_HS_ACK_GENERATED): High-speed Master code ACKed in High-speed Mode
5	ABORT_GCALL_RD	R	0x0	This field indicates that I2C in the master mode has sent a General Call but users program the byte that follows the General Call to be a read from the bus (DATA_CMD[9] is set to 1). Role of I2C: Master-Transmitter **Value**: • 0x0 (ABORT_GCALL_RD_VOID): GCALL is followed by read from bus-scenario not present • 0x1 (ABORT_GCALL_RD_GENERATED): GCALL is followed by read from bus
4	ABORT_GCALL_NOACK	R	0x0	This field indicates that I2C in master mode has sent a General Call and no slave on the bus acknowledged the General Call. Role of I2C: Master-Transmitter **Value**:

Bits	Field Name	RW	Reset	Description
				• 0x0 (ABORT_GCALL_NOACK_VOID): GCALL not ACKed by any slave-scenario not present • 0x1 (ABORT_GCALL_NOACK_GENERATED): GCALL not ACKed by any slave
3	ABORT_TX_NOACK	R	0x0	This field indicates the master-mode only bit. When the master receives an acknowledgement for the address, but when it sends data byte(s) following the address, it did not receive an acknowledge from the remote slave(s). Role of I2C: Master-Transmitter **Value**: • 0x0 (ABORT_TX_NOACK_VOID): Transmitted data non-ACKed by addressed slave-scenario not present • 0x1 (ABORT_TX_NOACK_GENERATED): Transmitted data not ACKed by addressed slave
2	ABORT_10B2_NOACK	R	0x0	This field indicates that the Master is in 10-bit address mode and that the second address byte of the 10-bit address was not acknowledged by any slave. Role of I2C: Master-Transmitter or Master-Receiver **Value**: • 0x0 (INACTIVE): This abort is not generated. • 0x1 (ACTIVE): Byte 2 of 10-bit Address is not ACKed by any slave.
1	ABORT_10B1_NOACK	R	0x0	This field indicates that the Master is in 10-bit address mode and the first 10-bit address byte was not acknowledged by any slave. Role of I2C: Master-Transmitter or Master-Receiver **Value**: • 0x0 (INACTIVE): This abort is not generated. • 0x1 (ACTIVE): Byte 1 of 10-bit Address is not ACKed by any slave.
0	ABORT_7B_NOACK	R	0x0	This field indicates that the Master is in 7-bit address mode and the address sent was not acknowledged by any slave. Role of I2C: Master-Transmitter or Master-Receiver **Value**: • 0x0 (INACTIVE): This abort is not generated.

Bits	Field Name	RW	Reset	Description
				• 0x1 (ACTIVE): This abort is generated because of NACK for 7-bit address.

4.4.3.4.34 DMA_CTRL

- **Name**: DMA Control Register
- **Description**: This register is only valid when I2C is configured with a set of DMA Controller interface signals. When I2C is not configured for DMA operation, this register does not exist and writing to the register address has no effect and reading from this register address will return zero. The register is used to enable the DMA Controller interface operation. There is a separate bit for transmitting and receiving. This can be programmed regardless of the state of EN.
- **Base Address**: 0x4000C300 + x*0x100
- **Offset**: 0x88
- **Reset Value**: 0x00000000

Table 4-181 DMA Control Register

Bits	Field Name	RW	Reset	Description
31:2	RSVD	R		Reserved bits
1	TX_EN	RW	0x0	Transmit DMA Enable. This bit enables/disables the transmit FIFO DMA channel. **Value**: • 0x0 (DISABLED): Transmit FIFO DMA channel disabled • 0x1 (ENABLED): Transmit FIFO DMA channel enabled
0	RX_EN	RW	0x0	Receive DMA Enable. This bit enables/disables the receive FIFO DMA channel. **Value**: • 0x0 (DISABLED): Receive FIFO DMA channel disabled • 0x1 (ENABLED): Receive FIFO DMA channel enabled

4.4.3.4.35 DMA_TX_LV

- **Name**: DMA Transmit Data Level Register
- **Description**: This register is only valid when the I2C is configured with a set of DMA interface signals. When I2C is not configured for DMA operation, this register does not exist; writing to its address has no effect; reading from its address returns zero.
- **Base Address**: 0x4000C300 + x*0x100
- **Offset**: 0x8C

- **Reset Value**: 0x00000000

Table 4-182 DMA Transmit Data Level Register

Bits	Field Name	RW	Reset	Description
31:3	RSVD	R		Reserved bits
2:0	LEVEL	RW	0x0	Transmit Data Level. This bit field controls the level at which a DMA request is made by the transmit logic.

4.4.3.4.36 DMA_RX_LV

- **Name**: DMA Receive Data Level Register
- **Description**: This register is only valid when I2C is configured with a set of DMA interface signals. When I2C is not configured for DMA operation, this register does not exist; writing to its address has no effect; reading from its address returns zero.
- **Base Address**: 0x4000C300 + x*0x100
- **Offset**: 0x90
- **Reset Value**: 0x00000000

Table 4-183 DMA Receive Data Level Register

Bits	Field Name	RW	Reset	Description
31:3	RSVD	R		Reserved bits
2:0	LEVEL	RW	0x0	Receive Data Level. This bit field controls the level at which a DMA request is made by the receive logic.

4.4.3.4.37 SDA_SETUP

- **Name**: I2C SDA Setup Register
- **Description**: This register controls the amount of time delay (in terms of number of clock periods) introduced in the rising edge of SCL – relative to SDA changing – when I2C services a read request in a slave-transmitter operation. The relevant I2C requirement is t_{SU_STO} as detailed in the Figure 4-36. This register must be programmed with a value that is equal to or greater than 2.

  Writes to this register succeed only when EN[0] = 0.

Note:

The length of setup time is calculated using [(SDA_SETUP – 1)*(clk_period)], so if users require 10 clock periods of setup time, they should program a value of 11. The SDA_SETUP register is only used by the I2C when operating as a slave transmitter.

- **Base Address**: 0x4000C300 + x*0x100

- **Offset**: 0x94
- **Reset Value**: 0x00000064

Table 4-184 I2C SDA Setup Register

Bits	Field Name	RW	Reset	Description
31:8	RSVD	R		Reserved bits
7:0	SETUP	RW	0x64	SDA Setup. It is recommended that if the required delay is 1000 ns, then for a clock frequency of 10 MHz, SETUP should be programmed to a value of 11. SETUP must be programmed with a minimum value of 2.

4.4.3.4.38 ACK_GEN_CALL

- **Name**: I2C ACK General Call Register
- **Description**: The register controls whether I2C responds with an ACK or NACK when it receives an I2C General Call address. This register is applicable only when the I2C is in slave mode.
- **Base Address**: 0x4000C300 + x*0x100
- **Offset**: 0x98
- **Reset Value**: 0x00000001

Table 4-185 I2C ACK General Call Register

Bits	Field Name	RW	Reset	Description
31:1	RSVD	R		Reserved bits
0	ACK_GEN_CALL	RW	0x1	ACK General Call. When this bit is set to 1, I2C responds with an ACK (by asserting data_oe) when it receives a General Call. Otherwise, I2C responds with a NACK (by negating data_oe). **Value**: • 0x0 (DISABLED): Generate NACK for General Call. • 0x1 (ENABLED): Generate ACK for a General Call.

4.4.3.4.39 EN_STAT

- **Name**: I2C Enable Status Register
- **Description**: The register is used to report the I2C hardware status when the EN[0] register bit is set from 1 to 0 (when I2C is disabled).
  - If EN[0] has been set to 1, Bits [2:1] of the EN_STAT register are forced to 0, and Bit 0 is forced to 1.

- If EN[0] has been set to 0, Bits [2:1] of the EN_STAT register are only valid as soon as Bit 0 is read as '0'.

Note:

When EN[0] has been set to 0, a delay occurs for Bit 0 to be read as 0 because disabling the I2C depends on I2C bus activities.

- **Base Address**: 0x4000C300 + x*0x100
- **Offset**: 0x9C
- **Reset Value**: 0x00000000

Table 4-186 I2C Enable Status Register

Bits	Field Name	RW	Reset	Description
31:3	RSVD	R		Reserved bits
2	S_RX_DATA_LOST	R	0x0	Slave Received Data Lost. This bit indicates if a Slave-Receiver operation has been aborted with at least one data byte received from an I2C transfer due to the setting bit 0 of EN from 1 to 0. When this bit is read as 1, I2C is deemed to have been actively engaged in an aborted I2C transfer (with matching address) and the data phase of the I2C transfer has been entered, even though a data byte has been responded with a NACK. **Note**: If the remote I2C master terminates the transfer with a STOP condition before the I2C has a chance to NACK transfer, and EN[0] has been set to 0, then this bit is also set to 1. When this bit is read as 0, I2C is deemed to have been disabled without being actively involved in the data phase of a Slave-Receiver transfer. The CPU can safely read this bit when EN (bit 0) is read as 0. **Value**: • 0x0 (INACTIVE): Slave RX Data is not lost • 0x1 (ACTIVE): Slave RX Data is lost
1	S_DIS_BUSY	R	0x0	Slave Disabled While Busy (Transmit, Receive). This bit indicates if a potential or active Slave operation has been aborted due to the setting bit 0 of the EN register from 1 to

Bits	Field Name	RW	Reset	Description
				0. This bit is set when the CPU writes a 0 to the EN register while: I2C is receiving the address byte of the Slave-Transmitter operation from a remote master, or address and data bytes of the Slave-Receiver operation from a remote master. When this bit is read as 1, I2C is deemed to have forced a NACK during any part of an I2C transfer, irrespective of whether the I2C address matches the slave address set in I2C (S_ADDR register) or if the transfer is completed before EN is set to 0 but has not taken effect. **Note**: If the remote I2C master terminates the transfer with a STOP condition before the I2C has a chance to NACK a transfer, and EN[0] has been set to 0, then this bit will also be set to 1. When this bit is read as 0, I2C is deemed to have been disabled when there is master activity, or when the I2C bus is idle. The CPU can safely read this bit when EN (bit 0) is read as 0. **Value**: • 0x0 (INACTIVE): Slave is disabled when it is idle • 0x1 (ACTIVE): Slave is disabled when it is active
0	EN	R	0x0	EN Status. This bit always reflects the value driven on the output port EN. When read as 1, I2C is deemed to be in an enabled state. When read as 0, I2C is deemed completely inactive. **Note**: The CPU can safely read this bit anytime. When this bit is read as 0, the CPU can safely read S_RX_DATA_LOST (bit 2) and S_DIS_BUSY (bit 1). **Value**: • 0x0 (DISABLED): I2C disabled • 0x1 (ENABLED): I2C enabled

4.4.3.4.40 FS_SPKLEN

- **Name**: I2C SS, FS or FM+ Spike Suppression Limit
- **Description**: This register is used to store the duration, measured in clock cycles, of the longest spike that is filtered out by the spike suppression logic when the component is operating in Standard Mode (SS), Fast Mode (FS), or Fast Plus (FM+) Mode. The relevant I2C requirement is tSP as detailed in the I2C Bus Specification. This register must be programmed with a minimum value of 1.
- **Base Address**: 0x4000C300 + x*0x100
- **Offset**: 0xA0
- **Reset Value**: 0x00000005

Table 4-187 I2C SS, FS or FM+ spike suppression limit

Bits	Field Name	RW	Reset	Description
31:8	RSVD	R		Reserved bits
7:0	FS_SPKLEN	RW	0x5	This register must be set before any I2C bus transaction can take place to ensure stable operation. This register sets the duration, measured in clock cycles, of the longest spike in the SCL or SDA lines that will be filtered out by the spike suppression logic. This register can be written only when the I2C interface is disabled which corresponds to the EN[0] register being set to 0. Writes at other times have no effect. The minimum valid value is 1; hardware prevents values less than 1 being written, and any attempt to write a value less than 1 results in 1 being set.

4.4.3.4.41 HS_SPKLEN

- **Name**: I2C HS Spike Suppression Limit Register
- **Description**: This register is used to store the duration, measured in clock cycles, of the longest spike that is filtered out by the spike suppression logic when the component is operating in High-speed mode. The relevant I2C requirement is tSP as detailed in the I2C Bus Specification. This register must be programmed with a minimum value of 1 and is implemented only if the component is configured to support High-speed mode.
- **Base Address**: 0x4000C300 + x*0x100
- **Offset**: 0xA4
- **Reset Value**: 0x00000001

Table 4-188 I2C HS Spike Suppression Limit Register

Bits	Field Name	RW	Reset	Description
31:8	RSVD	R		Reserved bits

Bits	Field Name	RW	Reset	Description
7:0	HS_SPKLEN	RW	0x1	This register must be set before any I2C bus transaction can take place to ensure stable operation. This register sets the duration, measured in clock cycles, of the longest spike in the SCL or SDA lines that will be filtered out by the spike suppression logic. This register can be written only when the I2C interface is disabled which corresponds to the EN[0] register being set to 0. Writes at other times have no effect. The minimum valid value is 1; hardware prevents values less than 1 being written, and any attempt to write a value less than 1 results in 1 being set.

4.4.3.4.42 SCL_STUCK_AT_LOW_TIMEOUT

- **Name**: I2C SCL Stuck at Low Timeout Register
- **Description**: This register is used to store the duration, measured in ic_clk cycles, used to generate an interrupt (SCL_STUCK_AT_LOW) if SCL is held low for the SCL_STUCK_LOW_TIMEOUT duration
- **Base Address**: 0x4000C300 + x*0x100
- **Offset**: 0xAC
- **Reset Value**: 0xFFFFFFFF

Table 4-189 I2C SCL Stuck at Low Timeout Register

Bits	Field Name	RW	Reset	Description
31:0	SCL_STUCK_LOW_TIMEOUT	RW	0xFFFFFFFF	I2C generates the interrupt to indicate SCL stuck at low (SCL_STUCK_AT_LOW) if it detects that the SCL is stuck at low for the SCL_STUCK_LOW_TIMEOUT in units of ic_clk period. This register can be written only when the I2C interface is disabled which corresponds to the EN[0] register being set to 0. Writes at other times have no effect.

4.4.3.4.43 SDA_STUCK_AT_LOW_TIMEOUT

- **Name**: I2C SDA Stuck at Low Timeout Register
- **Description**: This register is used to store the duration, measured in ic_clk cycles, used to recover the data (SDA) line through sending SCL pulses if SDA is held low for the SDA_STUCK_LOW_TIMEOUT duration
- **Base Address**: 0x4000C300 + x*0x100

- **Offset**: 0xB0
- **Reset Value**: 0xFFFFFFFF

Table 4-190 I2C SDA Stuck at Low Timeout Register

Bits	Field Name	RW	Reset	Description
31:0	SDA_STUCK_LOW_TIMEOUT	RW	0xFFFFFFFF	I2C initiates the recovery of SDA line through enabling the SDA_STUCK_RECOVERY_ENABLE (EN[3]) register bit, if it detects that the SDA is stuck at low for the SDA_STUCK_LOW_TIMEOUT in units of ic_clk period.

4.4.3.4.44 CLR_SCL_STUCK_DET

- **Name**: Clear SCL Stuck at Low Detect Interrupt Register
- **Description**: Clear SCL Stuck at Low Detect Interrupt Register
- **Base Address**: 0x4000C300 + x*0x100
- **Offset**: 0xB4
- **Reset Value**: 0x00000001

Table 4-191 I2C Clear SCL Stuck at Low Detect Interrupt Register

Bits	Field Name	RW	Reset	Description
31:1	RSVD	R		Reserved bits
0	CLR_SCL_STUCK_DET	R	0x0	Read this register to clear the SCL_STUCK_AT_LOW interrupt (Bit 14) of the RAW_INT_STAT register.

4.4.3.5 Electrical Specifications

Table 4-192 Electrical Specifications

Symbol	Parameter	Standard Mode (SS)		Fast Mode (FS)		Fast-plus Mode (FM+)		High-speed Mode (HS)		Unit
		Min.	Max.	Min.	Max.	Min.	Max.	Min.	Max.	
f_{SCL}	SCL clock frequency	0	100	0	400	0	1000	0	2000	kHz
t_{LOW}	SCL clock LOW time	4.7	-	1.3	-	0.5	-	0.16	-	µs
t_{HIGH}	SCL clock HIGH time	4.0	-	0.6	-	0.26	-	0.06	-	µs
t_{RSDA}	SDArise time	-	1000	20	300	-	120	10	80	ns
t_{FSDA}	SDA fall time	-	300		300		120	10	80	ns

Symbol	Parameter	Standard Mode (SS)		Fast Mode (FS)		Fast-plus Mode (FM+)		High-speed Mode (HS)		Unit
		Min.	Max.	Min.	Max.	Min.	Max.	Min.	Max.	
t_{RSCL}	SCLrise time	-	1000	20	300	-	120	10	40	ns
t_{FSCL}	SCL fall time	-	300		300		120	10	40	ns
t_{HD_STA}	START hold time	4.0	-	0.6	-	0.26	-	0.16	-	µs
t_{SU_DAT}	SDA setup time	250	-	100	-	50	-	10	-	ns
t_{HD_DAT}	SDA hold time	0	-	-	-	-	-	0	70	µs
t_{SU_STO}	STOP setup time	4.0	-	0.6	-	0.26	-	0.16	-	µs
t_{BUF}	BUS FREE time from STOP to START	4.7	-	1.3	-	0.5	-	-	-	µs

Note:

The bus load capacitance Cb = 100 pF.

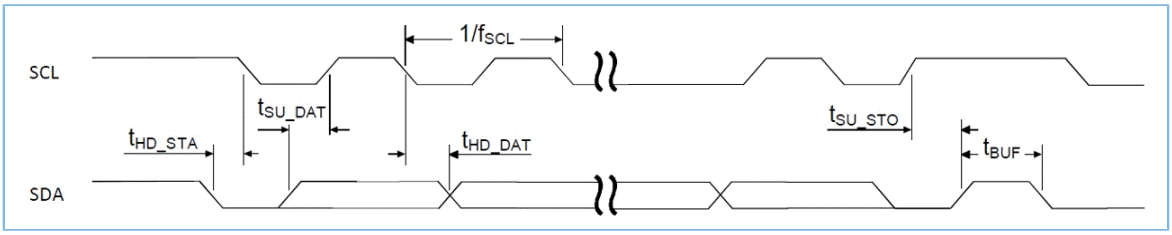

Figure 4-36 I2C timing diagram

4.4.4 DSPI

4.4.4.1 Introduction

DSPI (Display SPI) block implements a display interface that complies with MIPI DBI Type-C standard. It is based on SPI protocol and incorporates protocol extensions that are common in the industry. It is important to note that DSPI module is not intended to be used as a general-purpose SPI Master.

4.4.4.2 Main Features

Features of the DSPI controller include:

- Master operation (in Display applications, the host always serves as the master and only supports write operations.)
- Multiple transmission options: 3W1L, 4W1L, and 4W2L
- Data size options enhanced to ensure [8, 9, 16, 18, 24] bits to support transmission of common pixel formats
- Baud rate prescalers up to fPCLK/2

- Chip selection signal CS management by hardware or software
- Dedicated transmission and reception flags with interrupt capability
- DMA supported
- Bus busy status flag
- 16-byte TX FIFOs with DMA capability

4.4.4.3 Functional Description

DSPI allows synchronous, serial communication between the host and external display module. Application software can manage the communication by polling the status flag or using a DSPI interrupt. The main elements of DSPI and their interactions are shown in the following block diagram.

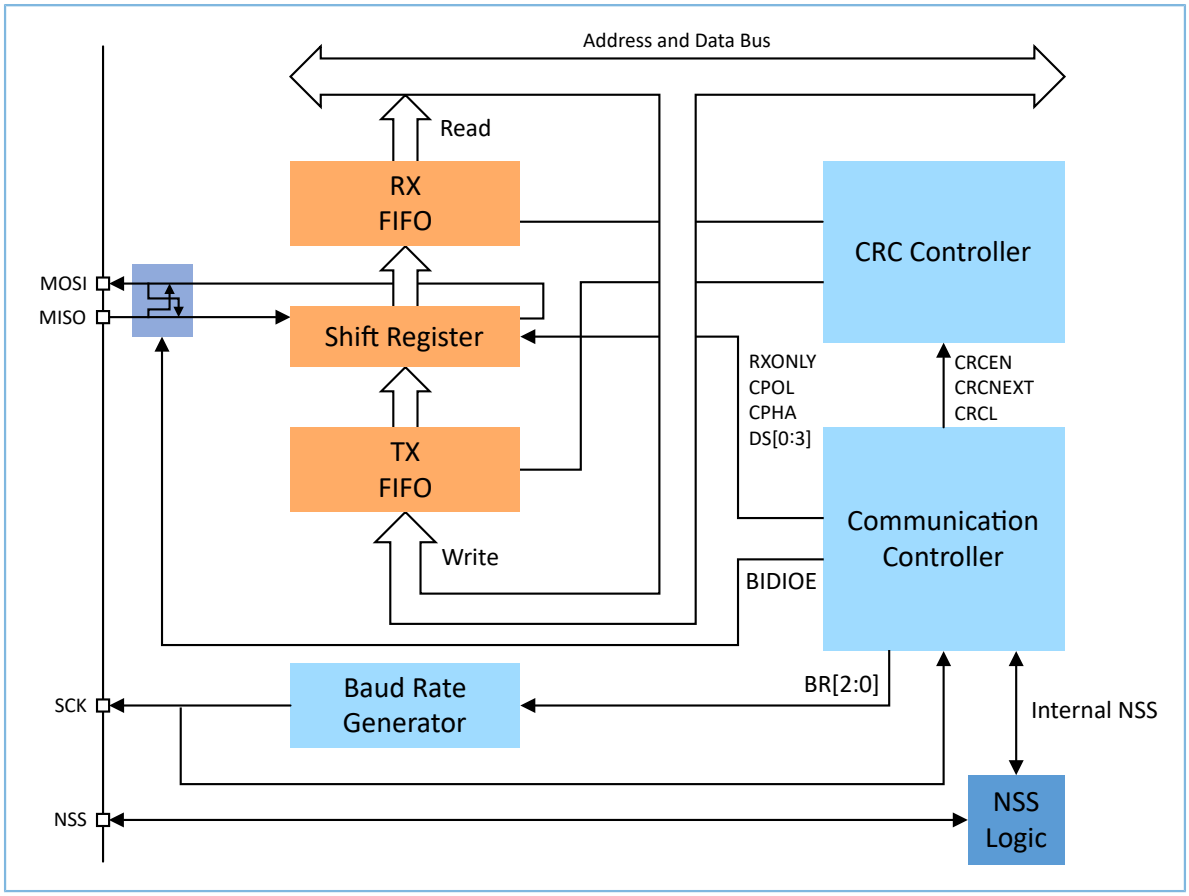

Figure 4-37 DSPI block diagram

4.4.4.3.1 Write Timing

DSPI implements the following three interfaces for data transmission to Display Modules, which is a serial data write.

- 3W1L (3-Wire 1-Lane) display interface features 3 interface wires & 1 data lane.
- 4W1L (4-Wire 1-Lane) display interface features 4 interface wires & 1 data lane.
- 4W2L (4-Wire 2-Lane) display interface features 4 interface wires & 2 data lanes.

4.4.4.3.1.1 3W1L Display Interface

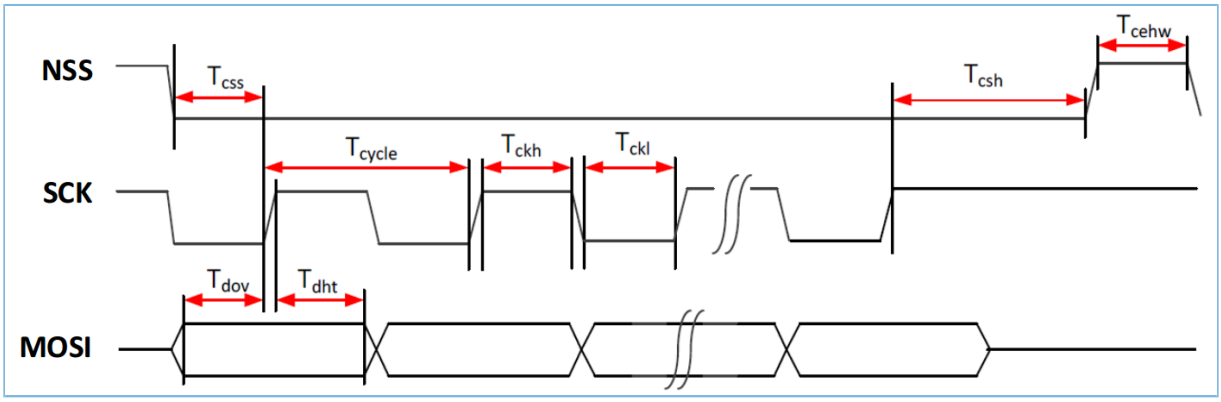

Figure 4-38 Serial data write timing on 3W1L display interface

4.4.4.3.1.2 4W1L Display Interface

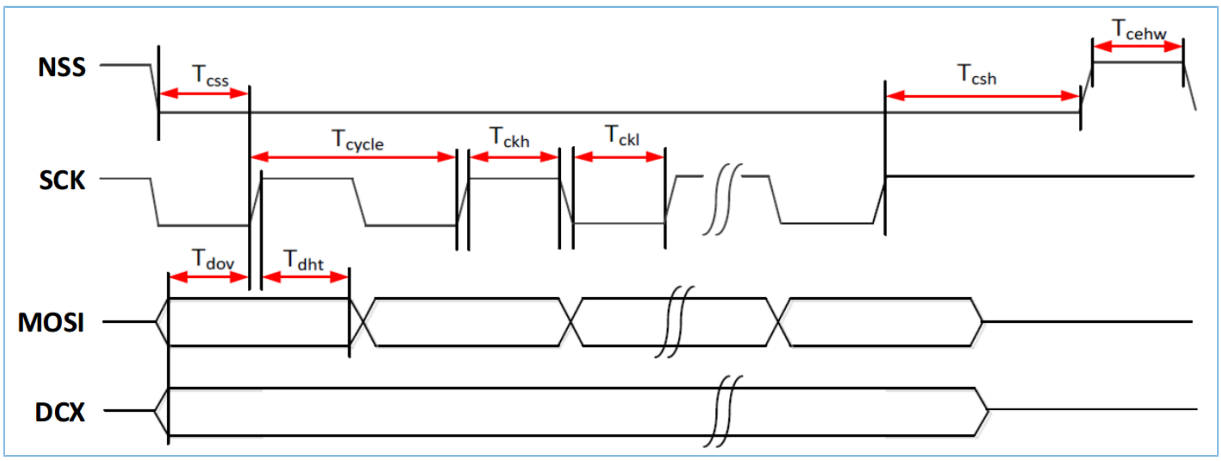

Figure 4-39 Serial data write timing on 4W1L display interface

4.4.4.3.1.3 4W2L Display Interface

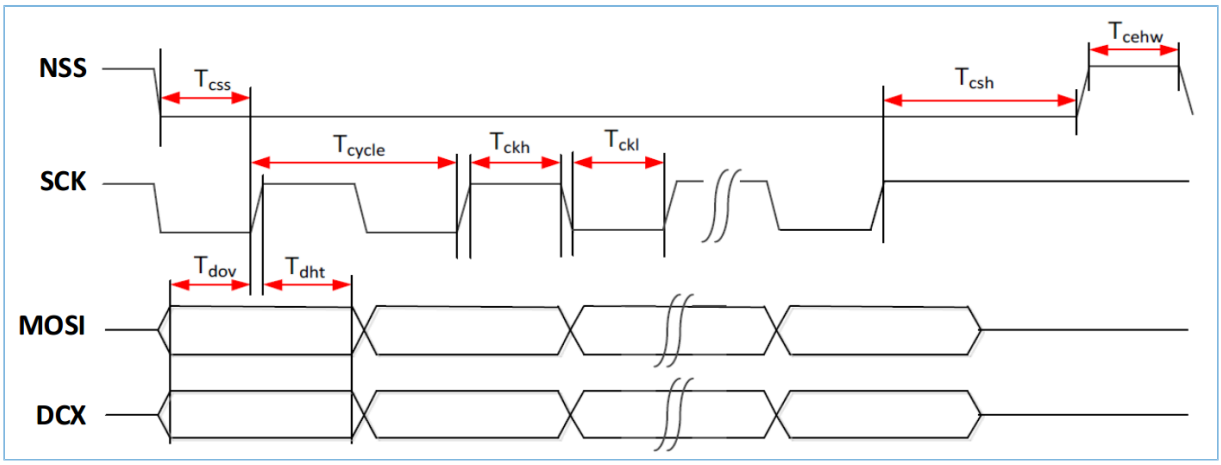

Figure 4-40 Serial data write timing on 4W2L display interface

Copyright © 2024 Shenzhen Goodix Technology Co., Ltd.

4.4.4.3.2 Color Coding

DSPI design facilitates several modes of operation to stream pixels to Display Modules, using the 3 Display Interface options. Mapping and timing of pixel bits to the interface wires are illustrated in the following.

4.4.4.3.2.1 3W1L Display Interface [A0+8-bit]

3W1L interface in [A0+8-bit] mode, as illustrated in the figure below, allows putting a total of 9 bits per transaction on the MOSI wire, in which

- First bit ("A0") is the data-versus-command indicator bit.
- Next 8 bits (D7 through D0) are data bits.

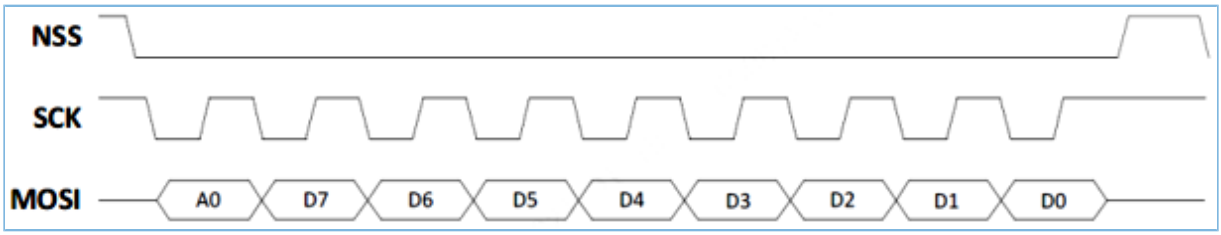

Figure 4-41 3W1L interface in [A0+8-bit] mode

The following 3 pixel formats may be used in this 8-data-bit mode:

- RGB565 pixel requires 16 bits hence it can be sent over 2 transactions.
- RGB666 pixel requires 18 bits hence it can be sent over 3 transactions. Some bit periods are unused.
- RGB888 pixel requires 24 bits hence it can be sent over 3 transactions.

Examples of pixel data mappings to D7:D0 bit periods are shown below..

Table 4-193 Example pixel data mappings to Dx bit periods for 3W1L [A0+8-bit] write

Format	Pixel over N transactions		D7	D6	D5	D4	D3	D2	D1	D0
RGB565	**2** transactions/pixel	*TR #1*	G2	G1	G0	B4	B3	B2	B1	B0
		TR #2	R4	R3	R2	R1	R0	G5	G4	G3
RGB666	**3** transactions/pixel	*TR #1*	B5	B4	B3	B2	B1	B0	x	x
		TR #2	G5	G4	G3	G2	G1	G0	x	x
		TR #3	R5	R4	R3	R2	R1	R0	x	x
RGB888	**3** transactions/pixel	*TR #1*	B7	B6	B5	B4	B3	B2	B1	B0
		TR #2	G7	G6	G5	G4	G3	G2	G1	G0
		TR #3	R7	R6	R5	R4	R3	R2	R1	R0

4.4.4.3.2.2 3W1L Display Interface [A0+9-bit]

3W1L interface in [A0+9-bit] mode, as illustrated in the figure below, allows putting a total of 10 bits per transaction on the MOSI wire, in which

- First bit ("A0") is the data-versus-command indicator bit.

- Next 9 bits (D8 through D0) are data bits.

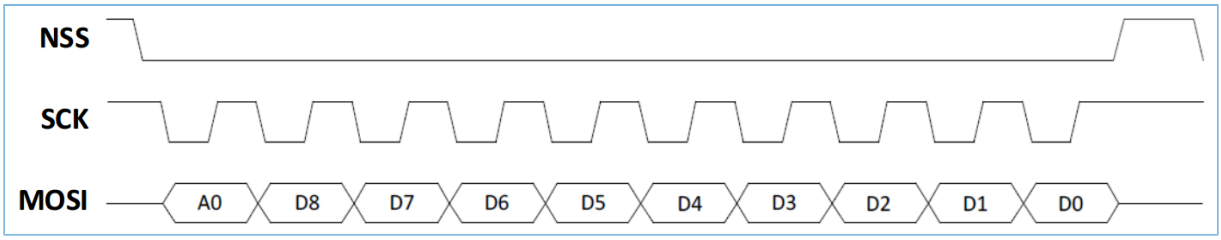

Figure 4-42 3W1L display interface [A0+9-bit] write

The following pixel format may be used in this 9-data-bit mode:

RGB666 pixel requires 18 bits hence it can be sent over 2 transactions.

Example pixel data mapping to D8:D0 bit periods is shown below.

Table 4-194 Example pixel data mapping to Dx bit periods for 3W1L [A0+9-bit] write

Format	Pixel over N transactions		D8	D7	D6	D5	D4	D3	D2	D1	D0
RGB666	**2** transactions/pixel	TR #1	G2	G1	G0	B5	B4	B3	B2	B1	B0
		TR #2	R5	R4	R3	R2	R1	R0	G5	G4	G3

4.4.4.3.2.3 3W1L Display Interface [A0+16-bit]

3W1L interface in [A0+16-bit] mode allows putting a total of 17 bits per transaction on the MOSI wire, in which

- First bit (“A0”) is the data-versus-command indicator bit.
- Next 16 bits (D15 through D0) are data bits.

Bit periods for this mode are not illustrated but it is very similar to the diagrams in Figure 4-41 and Figure 4-42. The difference is that, in this mode, data bits run from D15 to D0.

The following pixel format may be used in this 16-data-bit mode:

RGB565 pixel requires 16 bits hence it can be sent over 1 transaction.

4.4.4.3.2.4 3W1L Display Interface [A0+18-bit]

3W1L interface in [A0+18-bit] mode allows putting a total of 19 bits per transaction on the MOSI wire, in which

- First bit (“A0”) is the data-versus-command indicator bit.
- Next 18 bits (D17 through D0) are data bits.

Bit periods for this mode are not illustrated but it is very similar to the diagrams in Figure 4-41 and Figure 4-42. The difference is that, in this mode, data bits run from D17 to D0.

The following pixel format may be used in this 18-data-bit mode:

RGB666 pixel requires 18 bits hence it can be sent over 1 transaction.

4.4.4.3.2.5 3W1L Display Interface [A0+24-bit]

3W1L interface in [A0+24-bit] mode allows putting a total of 25 bits per transaction on the MOSI wire, in which

- First bit ("A0") is the data-versus-command indicator bit.
- Next 24 bits (D23 through D0) are data bits.

4.4.4.3.2.6 4W1L Display Interface [8-bit]

4W1L interface in [8-bit] mode, as illustrated in the figure below, allows putting

- 8 bits per transaction on the MOSI wire, in which all bits (D7 through D0) are data bits.
- Data-versus-command information on the dedicated DCX wire, for the duration of the transfer.

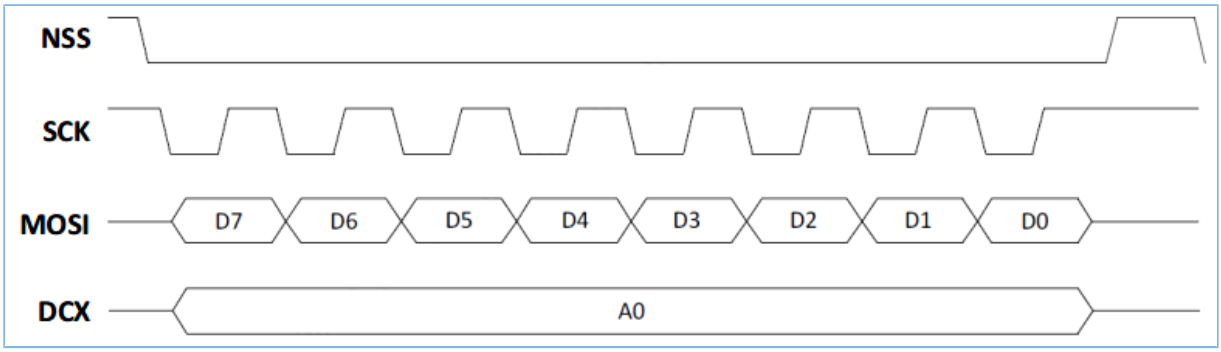

Figure 4-43 4W1L interface in [8-bit] mode

The following 3 pixel formats may be used in this 8-data-bit mode:

- RGB565 pixel requires 16 bits hence it can be sent over 2 transactions.
- RGB666 pixel requires 18 bits hence it can be sent over 3 transactions. Some bit periods are unused.
- RGB888 pixel requires 24 bits hence it can be sent over 3 transactions.

Example pixel data mappings to D7:D0 bit periods are shown below.

Table 4-195 Example pixel data mappings to Dx bit periods for 4W1L [8-bit] write

Format	Pixel over N transactions		D7	D6	D5	D4	D3	D2	D1	D0
RGB565	**2** transactions/pixel	*TR #1*	G2	G1	G0	B4	B3	B2	B1	B0
		TR #2	R4	R3	R2	R1	R0	G5	G4	G3
RGB666	**3** transactions/pixel	*TR #1*	B5	B4	B3	B2	B1	B0	x	x
		TR #2	G5	G4	G3	G2	G1	G0	x	x
		TR #3	R5	R4	R3	R2	R1	R0	x	x
RGB888	**3** transactions/pixel	*TR #1*	B7	B6	B5	B4	B3	B2	B1	B0
		TR #2	G7	G6	G5	G4	G3	G2	G1	G0
		TR #3	R7	R6	R5	R4	R3	R2	R1	R0

4.4.4.3.2.7 4W2L Display Interface [A0+8-bit]

4W2L interface in [A0+8-bit] mode, as illustrated in the figure below, allows putting a total of 18 bits per transaction on the MOSI and DCX wires, in which

- First bit ("A0") on both wires is the data-versus-command indicator bit. This bit is duplicated.
- Next 8 bits (D7 through D0) on MOSI wire are data bits.
- Next 8 bits (D15 through D8) on DCX wire are data bits.

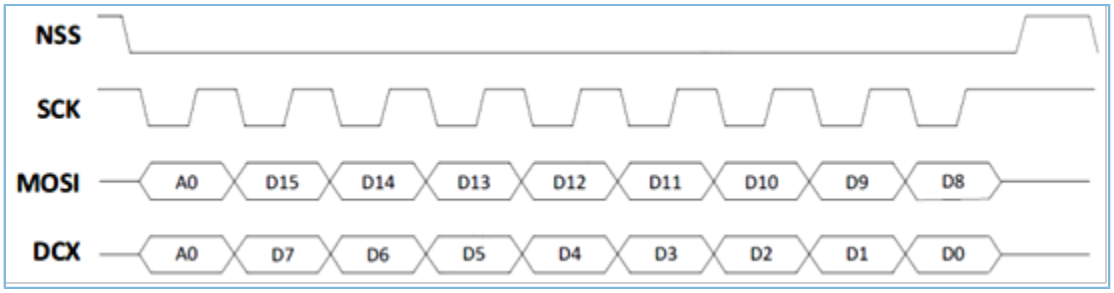

Figure 4-44 4W2L interface in [A0+8-bit] mode

The following pixel format may be used in this 16-data-bit mode:

RGB565 pixel requires 16 bits hence it can be sent over 1 transaction.

Example pixel data mappings to D15:D0 bit periods are shown below..

Table 4-196 Example pixel data mappings to Dx bit periods for 4W2L [A0+8-bit] write

Format	Pixel over N transactions		D15-D8 on MOSI				D7-D0 on DCX			
			D15	D14	D13	D12	D11	D10	D9	D8
RGB565	1 transaction/pixel	TR #1	R4	R3	R2	R1	R0	G5	G4	G3
			D7	D6	D5	D4	D3	D2	D1	D0
			G2	G1	G0	B4	B3	B2	B1	B0

4.4.4.3.2.8 4W2L Display Interface [A0+9-bit]

4W2L interface in [A0+9-bit] mode, as illustrated in the figure below, allows putting a total of 20 bits per transaction on the MOSI and DCX wires, in which

- First bit ("A0") on both wires is the data-versus-command indicator bit. This bit is duplicated.
- Next 9 bits (D8 through D0) on MOSI wire are data bits.
- Next 9 bits (D17 through D9) on DCX wire are data bits.

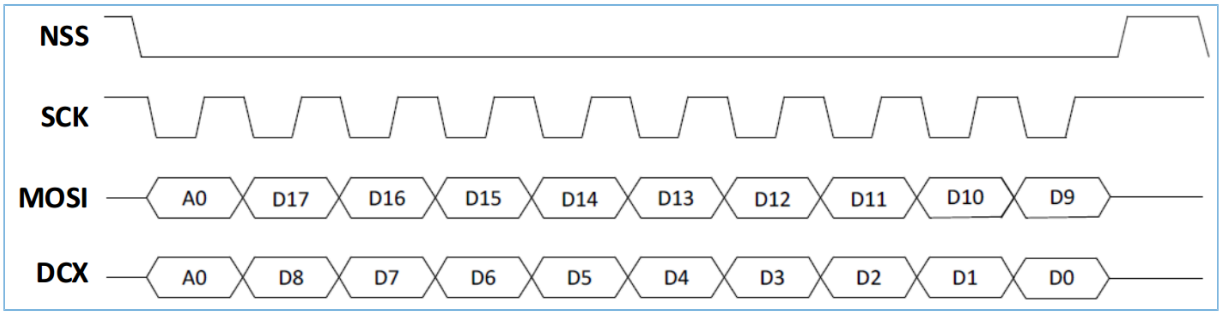

Figure 4-45 4W2L display interface [A0+9-bit]

The following pixel format may be used in this 18-data-bits mode:

RGB666 pixel requires 18 bits hence it can be sent over 1 transaction.

Example pixel data mappings to D17:D0 bit periods are shown below.

Table 4-197 Example pixel data mappings to Dx bit periods for 4W2L [A0+9-bit] write

Format	Pixel over N transactions		D17-D9 on MOSI					D8-D0 on DCX			
			D17	D16	D15	D14	D13	D12	D11	D10	D9
RGB666	1 transaction/pixel	TR #1	R5	R4	R3	R2	R1	R0	G5	G4	G3
			D8	D7	D6	D5	D4	D3	D2	D1	D0
			G2	G1	G0	B5	B4	B3	B2	B1	B0

4.4.4.3.2.9 4W2L Display Interface [A0+12-bit]

4W2L interface in [A0+12-bit] mode, as illustrated in Figure 4-46, allows putting a total of 26 bits per transaction on the MOSI and DCX wires, in which

- First bit (“A0”) on both wires is the data-versus-command indicator bit. This bit is duplicated.
- Next 12 bits (D23 through D12) on MOSI wire are data bits.
- Next 12 bits (D11 through D0) on DCX wire are data bits.

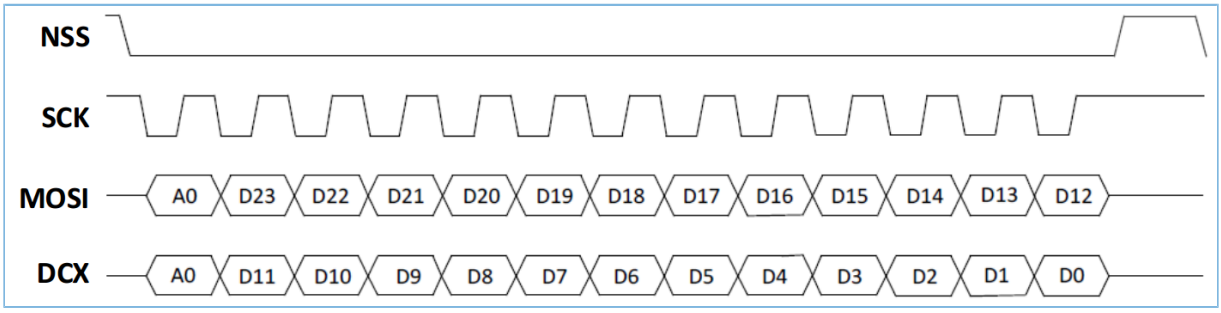

Figure 4-46 4W2L display interface [A0+12-bit]

The following pixel format may be used in this 24-data-bit mode:

RGB888 pixel requires 24 bits hence it can be sent over 1 transaction.

Example pixel data mappings to D23:D0 bit periods are shown below.

Table 4-198 Example pixel data mappings to Dx bit periods for 4W2L [A0+12-bit] write

Format	Pixel over N transactions		D23-D12 on MOSI								D11-D0 on DCX			
RGB888	1 transaction/pixel	TR #1	D23	D22	D21	D20	D19	D18	D17	D16	D15	D14	D13	D12
			R7	R6	R5	R4	R3	R2	R1	R0	G7	G6	G5	G4
			D11	D10	D9	D8	D7	D6	D5	D4	D3	D2	D1	D0
			G3	G2	G1	G0	B7	B6	B5	B4	B3	B2	B1	B0

4.4.4.3.3 Choice of DS and SPIMODE for Target Pixel Sizes

There are various choices of DS[4:0] and SPIMODE[1:0] when users desire to send pixels of a particular format. The choice mostly depends on the Display Module capabilities (number of wires and data lanes and pixel formats supported) and the desired level of performance (throughput).

The DMA transfer size, which is a setting outside of the DSPI IP, is also an important parameter, because it determines the granularity of data that is written into DSPI TX FIFO.

Typical choices of DS and SPIMODE for target pixel sizes are shown in Table 4-199. This table applies to 3 common pixel formats: RGB565, RGB666, and RGB888.

Table 4-199 Typical choices of DS and SPIMODE for target pixel sizes

Pixel Format	Pixel Size	Transfers per Pixel	DSPI Transfer Description	DMA Data Size DST_TR_WIDTH[2:0]	Transfer Mode SPIMODE[1:0]	DSPI Data size DS[4:0]
RGB565	16b	2	A0 + 8b on 1 lane	0x0: 8 bits	0x1: 3W1L	0x07: 8 bits
		1	A0 + 16b on 1 lane	0x1: 16 bits	0x1: 3W1L	0x0F: 16 bits
		2	8b on 1 lane	0x0: 8 bits	0x2: 4W1L	0x07: 8 bits
		1	A0 + 8b on 2 lanes	0x1: 16 bits	0x3: 4W2L	0x0F: 16 bits
RGB666	18b	3	A0 + 8b on 1 lane	0x0: 8 bits	0x1: 3W1L	0x07: 8 bits
		2	A0 + 9b on 1 lane	0x1: 16 bits	0x1: 3W1L	0x08: 9 bits
		1	A0 + 18b on 1 lane	0x2: 32 bits	0x1: 3W1L	0x11: 18 bits
		3	8b on 1 lane	0x0: 8 bits	0x2: 4W1L	0x07: 8 bits
		1	A0 + 9b on 2 lanes	0x2: 32 bits	0x3: 4W2L	0x11: 18 bits
RGB888	24b	3	A0 + 8b on 1 lane	0x0: 8 bits	0x1: 3W1L	0x07: 8 bits
		1	A0 + 24b on 1 lane	0x2: 32 bits	0x1: 3W1L	0x17: 24 bits
		1	A0 + 12b on 2 lanes	0x2: 32 bits	0x3: 4W2L	0x17: 24 bits

4.4.4.4 Registers

4.4.4.4.1 CTRL1

- **Name:** DSPI Control Register 1
- **Description:** This register specifies the configuration parameters of DSPI.

- **Base Address:** 0x4002E000
- **Offset:** 0x0
- **Reset Value:** 0x00000000

Table 4-200 DSPI Control Register 1

Bits	Field Name	RW	Reset	Description
31:16	RSVD	R		Reserved bits
15	BIDIMODE	RW	0x0	Bidirectional data mode enable. This bit enables half-duplex communication using common single bidirectional data line. Keep RXONLY bit clear when bidirectional mode is active. • 0x0: 2-line unidirectional data mode selected • 0x1: 1-line bidirectional data mode selected
14	BIDIOE	RW	0x0	Output enable in bidirectional mode. This bit combined with the BIDIMODE bit selects the direction of transfer in bidirectional mode. • 0x0: Output disabled (receive-only mode) • 0x1: Output enabled (transmit-only mode) **Note:** This bit applies to MOSI pin.
13	FPRST	RW	0x0	FIFO pointer reset This bit resets the pointers for TX FIFO and RX FIFO, works as a flush signal for both FIFOs. Any status flags related to RX FIFO and TX FIFO are also reset, as a result. • 0x0: No action. • 0x1: Flush TX FIFO and RX FIFO. **Note:** After this bit is written with 1 to flush the FIFOs, it should be written with 0 to ensure normal operation.
12:10	RSVD	R		Reserved bits
9	SSM	RW	0x0	Software slave management When the SSM bit is set, the CS pin input is replaced with the value from the SSI bit. • 0x0: Software slave management disabled • 0x1: Software slave management enabled

Bits	Field Name	RW	Reset	Description
8	SSI	RW	0x0	Internal slave select This bit takes effect only when the SSM bit is set. The value of this bit is forced onto the CS pin and the I/O. Value of the CS pin is ignored.
7	RSVD	R		Reserved bit
6	EN	RW	0x0	SPI enable • 0x0: Peripheral disabled • 0x1: Peripheral enabled
5:3	BR	RW	0x0	Baud rate control • 0x0: fPCLK/2 • 0x1: fPCLK/4 • 0x1: fPCLK/8 • 0x3: fPCLK/16 • 0x4: fPCLK/32 • 0x5: fPCLK/64 • 0x6: fPCLK/128 • 0x7: fPCLK/256 **Note:** These bits should not be changed when communication is ongoing.
2	MSTR	RW	0x0	Master selection • 0x0: Slave configuration • 0x1: Master configuration **Note:** This bit should not be changed when communication is ongoing.
1	CPOL	RW	0x0	Clock polarity • 0x0: CK to 0 in idle state. • 0x1: CK to 1 in idle state. **Note:** This bit should not be changed when communication is ongoing.

Bits	Field Name	RW	Reset	Description
				For DSPI operation adopting MIPI DBI Type-C, only CPOL=1, CPHA=1 combination is relevant; other combinations should not be used.
0	CPHA	RW	0x0	Clock phase • 0x0: The first clock transition is the first data capture edge • 0x1: The second clock transition is the first data capture edge **Note:** This bit should not be changed when communication is ongoing. For DSPI operation adopting MIPI DBI Type-C, only CPOL=1, CPHA=1 combination is relevant; other combinations should not be used.

4.4.4.4.2 CTRL2

- **Name:** DSPI Control Register 2
- **Description:** This register specifies the configuration parameters of DSPI.
- **Base Address:** 0x4002E000
- **Offset:** 0x04
- **Reset Value:** 0x0700

Table 4-201 DSPI Control Register 2

Bits	Field Name	RW	Reset	Description
31:13	RSVD	R		Reserved bits
12:8	DS	RW	0x07	Data size These 5 bits configure the data length for SPI transfers: • 0x0-0x2: Not used • 0x3-0x1F: 4 bits – 32 bits For regular SPI transfers, typical DS values are 8-bit, 16-bit, and 32-bit. For display SPI transfers, typical DS values are 8-bit, 9-bit, 16-bit, 18-bit, and 24-bit. **Note:**

Bits	Field Name	RW	Reset	Description
				If software attempts to write one of the “Not used” values, they are forced to be the value “0111” (8-bit).
7	TXEIE	RW	0x0	TX buffer empty interrupt enable • 0x0: TXE interrupt masked • 0x1: TXE interrupt not masked. Used to generate an interrupt request when the TXE flag is set.
6	RSVD	R		Reserved bit
5	ERRIE	RW	0x0	Error interrupt enable This bit controls the generation of an interrupt when an error condition occurs (OVR, MODF in SPI mode). • 0x0: Error interrupt is masked • 0x1: Error interrupt is enabled **Note:** This error will not occur in the current DSPI application.
4	FRF	RW	0x0	Frame format • 0x0: SPI Motorola mode • 0x1: SPI TI mode **Note:** This bit must be 0 at all times.
3	CSP	RW	0x0	CS pulse management This bit is used in master mode only. It allows the SPI to generate a CS pulse between two pieces of consecutive data in continuous transfers. In the case of a single data transfer, it forces the CS pin high level after the transfer. It has no meaning if CPHA = ’1’, or FRF = ’1’. • 0x0: No CS pulse • 0x1: CS pulse generated **Note:** This bit must be written only when the SPI is disabled (SPE=0).
2	SSOE	RW	0x0	SS output enable

Bits	Field Name	RW	Reset	Description
				• 0x0: SS output is disabled in master mode and the SPI interface can work in multimaster configuration. • 0x1: SS output is enabled in master mode and when the SPI interface is enabled. The SPI interface cannot work in a multimaster environment. **Note:** This bit must be 1 at all times.
1	TXDMAEN	RW	0x0	Tx buffer DMA enable When this bit is set, a DMA request is generated whenever the TXE flag is set. • 0x0: Tx buffer DMA disabled • 0x1: Tx buffer DMA enabled
0	RSVD	R		Reserved bit

4.4.4.4.3 STAT

- **Name:** DSPI Status Register
- **Description:** This register indicates the current working status of DSPI.
- **Base Address:** 0x4002E000
- **Offset:** 0x08
- **Reset Value:** 0x0002

Table 4-202 DSPI Status Register

Bits	Field Name	RW	Reset	Description
31:17	RSVD	R	0x0	Reserved bits
16:13	FTLCNT	RW	0x0	TX FIFO count in units of bytes These bits are set and cleared by hardware. • 0x0: FIFO empty • 0x1: 1/4 FIFO • 0x2: 1/2 FIFO • 0x3: FIFO full (considered as FULL when the FIFO threshold is greater than 1/2)
12:9	RSVD	R		Reserved bits
8	FRE	RW	0x0	Frame format error This flag is used for SPI in TI slave mode and I2S slave mode. [Not relevant]

Bits	Field Name	RW	Reset	Description
				This flag is set by hardware and reset when DSPI_SR is read by software. • 0x0: No frame format error • 0x1: A frame format error occurred
7	BUSY	RW	0x0	Busy flag • 0x0: SPI not busy • 0x1: SPI is busy in communication or Tx buffer is not empty This flag is set and cleared by hardware.
6	OVR	RW	0x0	Overrun flag • 0x0: No overrun occurred • 0x1: Overrun occurred This flag is set by hardware and reset by a software sequence. **Note:** This error will not occur in the current DSPI application.
5	MODF	RW	0x0	Mode fault • 0x0: No mode fault occurred • 0x1: Mode fault occurred This flag is set by hardware and reset by a software sequence. **Note:** This error will not occur in the current DSPI application.
4:2	RSVD	R		Reserved bits
1	TXE	RW	0x0	Transmit buffer empty • 0x0: Tx buffer not empty • 0x1: Tx buffer empty
0	RSVD	R		Reserved bit

4.4.4.4.4 DATA

- **Name:** DSPI Data Register
- **Description:** This register is a write data register.
- **Base Address:** 0x4002E000

- **Offset:** 0x0C
- **Reset Value:** 0x0000

Table 4-203 DSPI Data Register

Bits	Field Name	RW	Reset	Description
31:0	DATA	RW	0x0	Data register Data to be transmitted **Note:** Data is always right-aligned. Unused bits are ignored when writing to the register.

4.4.4.4.5 MODE

- **Name:** DSPI Display Mode Register
- **Description:** This register specifies the working mode parameters of DSPI.
- **Base Address:** 0x4002E000
- **Offset:** 0x10
- **Reset Value:** 0x0000

Table 4-204 DSPI Display Mode Register

Bits	Field Name	RW	Reset	Description
31:3	RSVD	R		Reserved bits
2	DCX	RW	0x0	Value of data-versus-command information used in Display SPI Protocol Modes. This bit specifies the value to put on the DCX wire of Display SPI interface or the “A0” value to use on the DOUT bit stream when this information is provided as part of the outgoing data bits. • 0x0: Sets the data-versus-command value to 0. • 0x1: Sets the data-versus-command value to 1.
1:0	SPIMODE	RW	0x0	SPI protocol mode This bit selects the SPI protocol mode. • 0x0: Normal SPI (Motorola protocol only, not used) • 0x1: Display SPI 3-wire 1-lane interface • 0x2: Display SPI 4-wire 1-lane interface • 0x3: Display SPI 4-wire 2-lane interface

4.4.5 QSPI

4.4.5.1 Introduction

The device supports three instances of Quad Serial Peripheral Interface (QSPI) master, namely QSPI M0, QSPI M1, and QSPI M2, which are used to interface to external slave devices. All the QSPI instances support memory mapped mode that can be used in accessing a storage device such as NOR Flash or PSRAM.

Each QSPI instance can be configured to work with standard SPI/Dual SPI/Quad SPI mode.

4.4.5.2 Main Features

- Standard/Dual/Quad Motorola SPI support
- Single slave select line to support 1 slave
- Built-in 16-/32-word RX/TX FIFOs for different instances
  - QSPI0: 16-word RX/TX FIFOs
  - QSPI1: 32-word RX/TX FIFOs
  - QSPI2: 16-word RX, 32-word TX FIFOs
- Transfer data size up to 32 bits
- Programmable output interface frequency up to 48 MHz
- Supporting four SPI modes for different clock edge and phase configurations
- DMA support
- Access mode support:
  - Read and write transfer support in register mode
  - Read and write transfer support in memory mapped (XIP) mode
  - Supporting free data endian mode in XIP operation (for reads)
  - Continuous transfer mode in XIP operation (for reads)
  - Data prefetch support in XIP operation (for reads)
- Enhanced (Dual/Quad) SPI features:
  - Programmable address, instruction, wait cycles, and data frame size
  - Clock stretch feature support
- Programmable receive sample delay support
- Maskable interrupt generation
- Supporting adjusting the CS setup delay time
- Perfect access throughput

4.4.5.3 Functional Description

4.4.5.3.1 Block Diagram

- AHB interface and DMA controller interface
- Transmit and receive FIFO controllers and an FSM controller
- Register block
- Shift control and interrupt logic

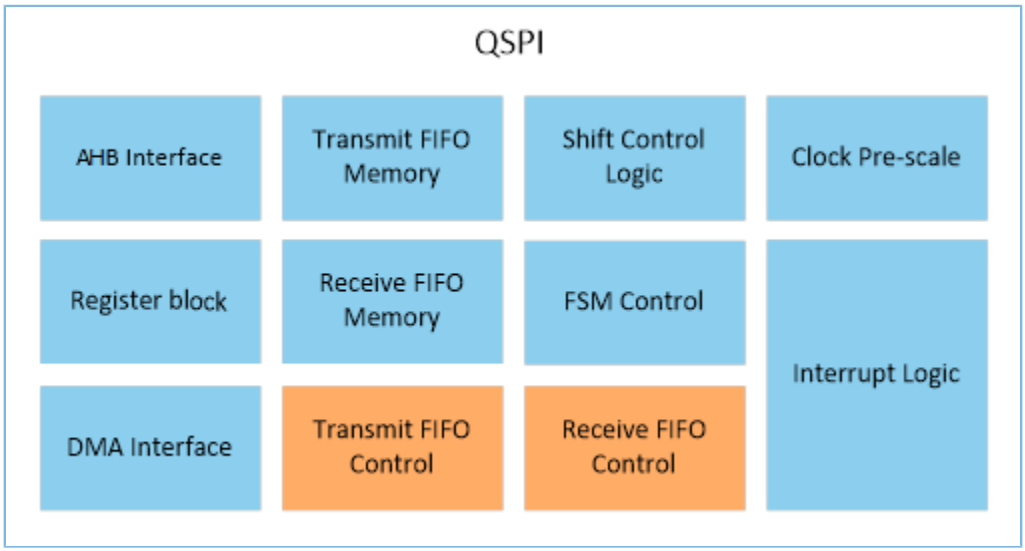

Figure 4-47 QSPI block diagram

4.4.5.3.2 Clock Ratios

The maximum frequency of the bit-rate clock (sclk_out) is one-half the frequency of ssi_clk signal. This is to allow the shift control logic to capture data on one clock edge of sclk_out signal and propagate data on the opposite edge; this is illustrated in the following figure.

The frequency of sclk_out signal can be derived from the following equation.

$$F_{sclk_{out}} = \frac{F_{ssi_{clk}}}{SCKDV}$$

Where, SCKDV (Baud Rate Register, BAUD[15:0]) is a programmable register holding any even value in the range of 0–65,534. If SCKDV = 0, sclk_out is disabled.

The sclk_out line only toggles when an active transfer is in progress. At all other time, it is held in inactive state, as define by the serial protocol under which it operates.

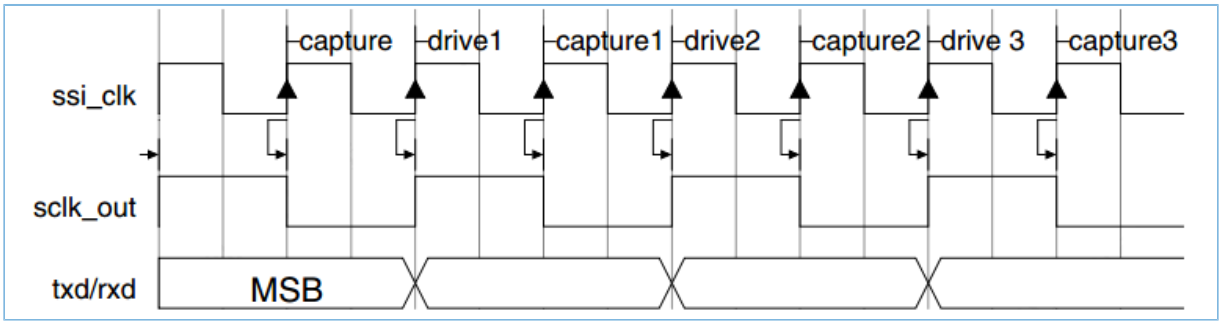

Figure 4-48 Maximum sclk_out/ssi_clk ratio

4.4.5.3.3 Transmit and Receive FIFO Buffers

The width of both TX and RX FIFO buffers is fixed at 32 bits. Data frames that are less than 32 bits must be right-justified when written into the TX FIFO buffer. The shift control logic automatically right-justifies receive data in the RX FIFO buffer.

Each data entry in the FIFO buffers contains a single data frame. Storing multiple data frames in a single FIFO location is not supported; for example, users may not store two 8-bit data frames in a single FIFO location. If an 8-bit data frame is required, the upper 8-bits of the FIFO entry are ignored or unused when the serial shifter transmits the data.

Transmit FIFO

The TX FIFO is loaded by AHB write commands to the QSPI data register (DATA). Data is popped (removed) from the TX FIFO by the shift control logic into the transmit shift register. The TX FIFO generates a FIFO empty interrupt request (qspi_txe_intrl) when the number of entries in the FIFO is less than or equal to the FIFO threshold value. The threshold value, set through the programmable transmit FIFO threshold level register (TX_FTL[4:0]), determines the level of FIFO entries at which an interrupt is generated. The threshold value allows users to provide early indication to the processor that the TX FIFO is nearly empty. A TX FIFO overflow interrupt (qspi_txo_intr) is generated if users attempt to write data into an already full TX FIFO.

Receive FIFO

Data is popped from the RX FIFO by AHB read commands to the QSPI data register (DATA). The RX FIFO is loaded from the receive shift register by the shift control logic. The RX FIFO generates a FIFO-full interrupt request (qspi_rxf_intr) when the number of entries in the FIFO is greater than or equal to the FIFO threshold value plus 1. The threshold value, set through programmable receive FIFO threshold level register (RX_FTL[4:0]), determines the level of FIFO entries at which an interrupt is generated.

The threshold value allows users to provide early indication to the processor that the RX FIFO is nearly full. A RX FIFO overrun interrupt (qspi_rxo_intr) is generated when the receive shift logic attempts to load data into a completely full RX FIFO. However, this newly received data is lost. A RX FIFO underflow interrupt (qspi_rxu_intr) is generated if users attempt to read from an empty RX FIFO. This alerts the processor that the read data is invalid.

4.4.5.3.4 QSPI Interrupts

The QSPI controller supports individual interrupt requests, each of which can be masked.

The QSPI controller interrupts are described as follows:

- TX FIFO empty interrupt (qspi_txe_intr) – Set when the TX FIFO is equal to or below its threshold value and requires service to prevent an under-run. The threshold value, set through a software-programmable register, determines the level of TX FIFO entries at which an interrupt is generated. This interrupt is cleared by hardware when data is written into the TX FIFO buffer, bringing it over the threshold level.
- TX FIFO overflow interrupt (qspi_txo_intr) – Set when an AHB access attempts to write into the TX FIFO after it has been completely filled. When set, data written from the AHB is discarded. This interrupt remains set until users read the TX FIFO error interrupt clear register (TXEICR[0]).
- TX FIFO underflow interrupt (qspi_txu_intr) - Present only when internal DMA mode or SPI bridge mode or XIP write transfer is enabled. The interrupt is set when TX FIFO pop happens from serial FSM and FIFO is empty. This interrupt remains set until users read the TX FIFO error interrupt clear register (TXEICR[0]).
- RX FIFO full interrupt (qspi_rxf_intr) – Set when the RX FIFO is equal to or above its threshold value plus 1 and requires service to prevent an overflow. The threshold value, set through a software-programmable register, determines the level of RX FIFO entries at which an interrupt is generated. This interrupt is cleared by hardware when data is read from the RX FIFO buffer, bringing it below the threshold level.
- RX FIFO overflow interrupt (qspi_rxo_intr) – Set when the receive logic attempts to place data into the RX FIFO after it has been completely filled. When set, newly received data is discarded. This interrupt remains set until users read the RX FIFO overflow interrupt clear register (RXOICR[0]).
- RX FIFO underflow interrupt (qspi_rxu_intr) – Set when an AHB access attempts to read from the RX FIFO when it is empty. When set, zeros are read back from the RX FIFO. This interrupt remains set until users read the RX FIFO underflow interrupt clear register (RXUICR[0]).

4.4.5.3.5 Transfer Modes

When transferring data on the serial bus, the SPI controller operates in the modes discussed in this section. The transfer mode (TMOD) is set by writing to bit8 and bit9 of the SPI control register 0 (CTRL0).

Note:

The transfer mode setting does not affect the duplex of the serial transfer.

4.4.5.3.5.1 Transmit and Receive

When CTRL0[11:10] = 0x0, both transmit and receive logics are valid. The data transfer occurs as normal according to the selected frame format (serial protocol). Transmit data is popped from the TX FIFO and sent through the TXD line to the target device, which replies with data on the RXD line. The receive data from the target device is moved from the receive shift register into the RX FIFO at the end of each data frame.

4.4.5.3.5.2 Transmit Only

When CTRL0[11:10] = 0x1, the received data is invalid and should not be stored in the RX FIFO. The data transfer occurs as normal, according to the selected frame format (serial protocol). Transmit data is popped from the TX FIFO and sent through the TXD line to the target device, which replies with data on the RXD line. At the end of the data frame, the receive shift register does not load its newly received data into the RX FIFO. The data in the receive shift

register is overwritten by the next transfer. Users should mask interrupts originating from the receive logic when this mode is entered.

4.4.5.3.5.3 Receive Only

When CTRL0[11:10] = 0x2, the transmit data is invalid. When QSPI is configured as a slave, the TX FIFO is never popped in Receive Only mode. The TXD output remains at a constant logic level during the transmission. The data transfer occurs as normal according to the selected frame format (serial protocol). The receive data from the target device is moved from the receive shift register into the RX FIFO at the end of each data frame. Users should mask interrupts originating from the transmit logic when this mode is entered.

4.4.5.3.5.4 EEPROM Read

When CTRL0[11:10] = 0x3, the transmit data is used to transmit an opcode and/or an address to the EEPROM device. Typically, this takes three data frames (8-bit opcode followed by 8-bit upper address and 8-bit lower address). During the transmission of the opcode and address, no data is captured by the receive logic (as long as the SPI controller master is transmitting data on its TXD line, data on the RXD line is ignored). The QSPI controller master continues to transmit data until the TX FIFO is empty. Therefore, users should only have enough data frames in the TX FIFO to supply the opcode and address to the EEPROM. If more data frames are in the TX FIFO than needed, read data is lost.

When the TX FIFO becomes empty (all control information has been sent), data on the receive line (RXD) is valid and is stored in the RX FIFO; the TXD output is held at a constant logic level. The serial transfer continues until the number of data frames received by the QSPI controller master matches the value of the NDF (Number of Data Frames) field in the SPI Control register 1 (CTRL1[15:0]) + 1.

4.4.5.3.6 DMA Controller Interface

The QSPI has optional built-in DMA capability which can be selected during configuration; it has a handshaking interface to a DMA controller to request and control transfers. The AHB bus is used to perform the data transfer to or from the DMA. While the QSPI DMA operation is designed in a generic way to fit any DMA controller as easily as possible, it is designed to work seamlessly, and is best to be used with the DMA controller.

The QSPI has the following DMA registers:

- DMACR – control register to enable DMA operation
- DMATDLR –register to set the TX FIFO level at which a DMA request is made
- DMARDLR– register to set the RX FIFO level at which a DMA request is made

To enable the DMA controller interface on the QSPI, users must write the DMA control register (DMACR). Writing 1 into the TDMAE bit field of DMACR register enables the QSPI transmit handshaking interface. Writing 1 into the RDMAE bit field of the DMACR register enables the QSPI receive handshaking interface.

4.4.5.3.6.1 DMA Operation

As a block flow control device, the DMA controller is programmed by the processor with the number of data items (block size) that are to be transmitted or received by the QSPI; this is programmed into the BLOCK_TS field of the

CTL_HIx register. The block size includes the instructions, address, and data in case of the enhanced SPI mode of operation.

The block is broken into a number of transactions, with each initiated by a request from the QSPI. The DMA controller must also be programmed with the number of data items (in this case, QSPI FIFO entries) to be transferred for each DMA request. This is also known as the burst transaction length, and is programmed into the SRC_MSIZE and DST_MSIZE fields of the DMA CTL_LOx register for source and destination, respectively.

4.4.5.3.6.2 Transmit Watermark Level

During QSPI serial transfers, TX FIFO requests are made to the DMA whenever the number of entries in the TX FIFO is less than or equal to the DMA transmit data level register (DMATDLR) value; this is known as the watermark level. The DMA responds by writing a burst of data to the TX FIFO buffer of length CTL_LOx.DST_MSIZE.

Data should be fetched from the DMA often enough for the TX FIFO to perform serial transfers continuously; that is, when the FIFO begins to empty, another DMA request should be triggered. Otherwise, the FIFO runs out of data (underflow). To prevent this condition, users must set the watermark level correctly.

4.4.5.3.6.3 Receive Watermark Level

During QSPI serial transfers, RX FIFO requests are made to the DMA controller whenever the number of entries in the RX FIFO is at or above the DMA receive data level register; that is, DMARDLR+1. This is known as the watermark level. The DMA controller responds by fetching a burst of data from the RX FIFO buffer of length CTL_LOx.SRC_MSIZE.

Data should be fetched by the DMA often enough for the RX FIFO to accept serial transfers continuously; that is, when the FIFO begins to fill, another DMA transfer is requested. Otherwise, the FIFO fills with data (overflow). To prevent this condition, users must set the watermark level correctly.

4.4.5.3.6.4 DMA Usage Allocation

In the register mode (non-XIP mode), QSPI uses a handshake request signal to start the DMA transfer. DMA0 instance provides handshake request signals to QSPI0 and QSPI1 instances, and DMA1 instance provides handshake request signals to QSPI1 and QSPI2 instances, so

- QSPI M0 can use DMA0 to receive and transmit data.
- QSPI M1 can use DMA0/DMA1 to receive and transmit data.
- QSPI M2 can use DMA1 to receive and transmit data.

Meanwhile, Channel0 and Channel1 of DMA have the biggest FIFO depth. It is suggested to assign the Channel0 & Channel1 of DMA to QSPI instances to get the maximum throughput.

4.4.5.3.7 Clock Stretching in Enhanced SPI Transfers

While QSPI is transmitting or receiving data from a QSPI slave device, the software may not be able to keep up with the transfer rate due to the bandwidth issues on the slave interface. In such cases, TX FIFO becomes empty or RX FIFO overflows while transmitting and receiving the data.

To avoid data corruption, CPU must discard the current operation and start a new transfer. This process is time consuming and inefficient.

To handle such scenarios, clock stretching support has been added in QSPI.

- For write transactions, whenever TX FIFO becomes empty, QSPI does not de-select the slave. Instead, it masks the clock until the new data is pushed into the TX FIFO. Hence, the transfer is not broken, and CPU intervention is not required. The transfer is resumed after the TX FIFO receives at least one FIFO entry.
- For read transactions, similar to the write transaction, whenever QSPI detects that RX FIFO is full, the clock is masked until the data is read from FIFO. The transfer is resumed when FIFO level goes below the set threshold level in the RX_FTL[4:0] register.

  QSPI block provides a programming bit (CLK_STRETCH_EN) in QSPI enhanced control register 0 (SPI_CTRL0[30]) to enable clock stretching feature.

4.4.5.3.8 Memory Mapped Mode (XIP Mode)

QSPI module allows directly performing memory read operations from AHB transaction. This is called memory mapped access mode (also called XIP mode), in which QSPI acts as memory mapped interface to an SPI memory.

4.4.5.3.8.1 Memory Mapped Read Model

XIP read operation works under memory mapped mode. Typical devices are NOR Flash and PSRAM with QSPI/DSPI interface.

XIP operations are supported only in Dual or Quad enhanced SPI modes of operation, so bit (SPI_FRF) of the CTRL0[23:22] must not be programmed to 0. When CTRL0[23:22] = 0x1, select Dual SPI frame format. When CTRL0[23:22] = 0x2, select Quad SPI frame format. Typically, an XIP operation consists of an address phase and a data phase. The programming flow to set up an XIP transfer is as follows:

- Set the SPI protocol format value to Motorola format in CTRL0.FRF register.
- Set the SPI frame format value in XIP_CTRL.FRF register.
- Set the address length, wait cycles, and transaction type in XIP_CTRL.ADDR_L register. It should be noted that the maximum address length is 32 bits.

The instruction phase can also be included in XIP transfer by using INST_EN field of the XIP_CTRL register. In this case, the following registers must be set:

- Set length of instruction in the INST_L field of XIP_CTRL[10:9] register.
- Write the instruction opcodes in the XIP_INCR_INST[15:0] and XIP_WRAP_INST[15:0] registers.

After the programming is complete, users can initiate a read transaction through the AHB interface that is transferred to the SPI peripheral using programmed values.

After finishing above setting, users can read the device's data by accessing the corresponding QSPI's memory mapped space.

4.4.5.3.8.2 Memory Mapped Write Model

XIP write feature can be used to access PSRAM device supporting QSPI/DSPI interface (it cannot be used to write the NOR Flash device). Each typical write time meets with the PSRAM's tCEM time.

XIP operations are supported only in enhanced SPI modes of operation (Dual and Quad modes).

QSPI can write into the DSPI/QSPI interface memory without software overhead of writing the instruction and address separately into the transmit buffer. XIP write feature executes the write transactions directly without software intervention.

The programming flow to set up an XIP transfer is as follows:

- Set the SPI frame format value in XIP_WR_CTRL. WR_CTRL_FRF register.
- Set the address length, wait cycles, and transaction type in XIP_WR_CTRL.WR_CTRL_ADDRL. It should be noted that the maximum address length is 32 bits.
- Set length of instruction in the XIP_WR_CTRL.WR_INST_L register.
- Write the instruction opcodes in the XIP_WR_INCR_INST and XIP_WR_WRAP_INST registers.

4.4.5.3.8.3 Memory Mapped Space

After activating the memory mapped mode, users can perform access operations from AHB transaction. The AHB memory space for QSPI instances are allocated as Figure 4-49.

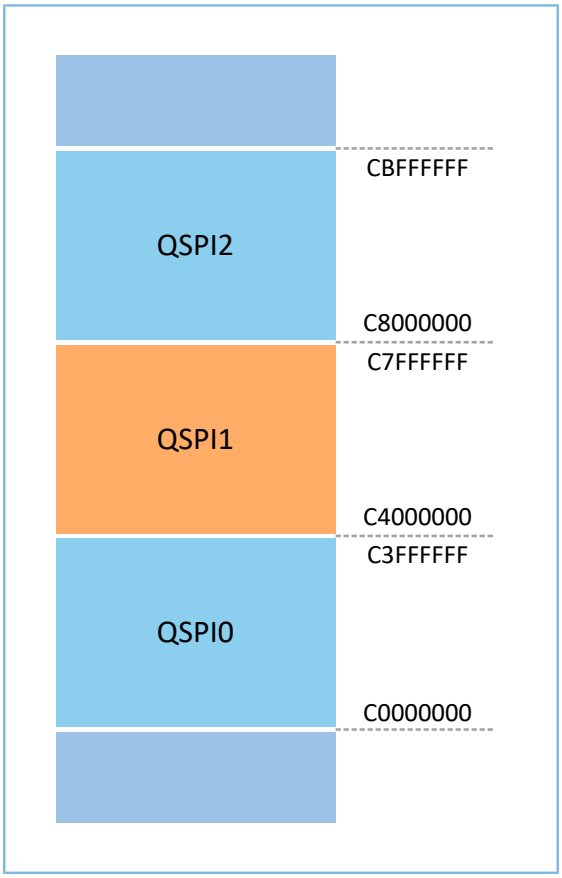

Figure 4-49 Memory map address for QSPI

- QSPI0 memory mapped space is from C0000000 to C3FFFFFF; 65,536 KB in total.
- QSPI1 memory mapped space is from C4000000 to C7FFFFFF; 65,536 KB in total.

- QSPI2 memory mapped space is from C8000000 to CBFFFFFF; 65,536 KB in total.

4.4.5.3.8.4 Data Endian Mode for Memory Mapped Read Model

QSPI supports different endian data in XIP Read mode. Users can refer to the following chart to configure the endian mode as needed. The related register is QSPI_MODE_XIP register.

Table 4-205 Different endian data in XIP read mode

AHB Address	**0x20040000**	**0x20040001**	**0x20040002**	**0x20040**	**0x20040**	**0x20040**	**0x20040**	**0xAC000007**	**0x20040**	**0x20040**	**0x20040**	**0x20040**
PSRAM Address	0x000000	0x000001	0x000002	0x000003	0x000004	0x000005	0x000006	0x000007	0x000008	0x000009	0x00000A	0x00000B
Saved Data	0x01	0x02	0x03	0x04	0x05	0x06	0x07	0x08	0x09	0x0A	0x0B	0x0C

Table 4-206 XIP read by different mode & address type

Data Type	**By Byte**				**By Short**		**By Integer**	
Access Addr/ Endian Mode	0x20040000	20040001	20040002	20040003	20040000	20040002	20040000	20040004
Mode 0/3	0x01	0x02	0x03	0x04	0x0102	0x0304	0x01020304	0x05060708
Mode 1	0x01	0x02	0x03	0x04	0x0201	0x0403	0x02010403	0x06050807
Mode 2	0x01	0x02	0x03	0x04	0x0201	0x0403	0x04030201	0x08070605

4.4.5.3.9 Enhanced Feature

4.4.5.3.9.1 Related Delay Between CS and SCK

T_{csu} (CS Setup Delay Time) is an important parameter to access QSPI device. If T_{csu} is not enough, the access may fail. T_{ch} (CS Release Delay Time) is symmetric to T_{csu}.

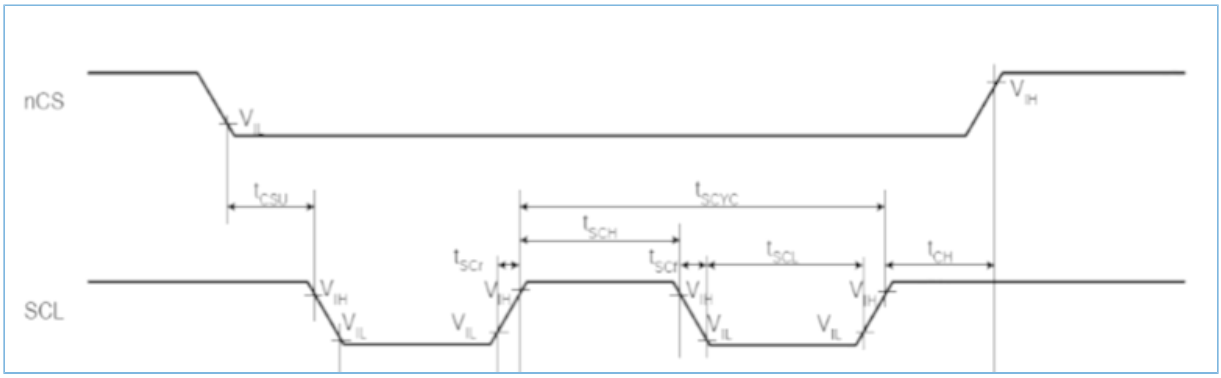

Figure 4-50 Relationship between T_{ch} and T_{csu}

If necessary, users can set the two parameters wider to fit with devices' requirement. The related registers are QSPI_CS_SETUP_DLY and QSPI_CS_RELEASE_DLY.

4.4.5.4 Registers

4.4.5.4.1 CTRL0

- **Name:** QSPI Control Register 0
- **Description:** This register controls the serial data transfer. It is impossible to write to this register when the QSPI is enabled.
- **Base Address:** QSPI0: 0x40021000; QSPI1: 0x40022000; QSPI2: 0x40023000
- **Offset:** 0x00
- **Reset Value:** 0x01070000

Table 4-207 QSPI Control Register 0

Bits	Field Name	RW	Reset	Description
31:24	RSVD	R		Reserved bits
23:22	SPI_FRF	RW	0x0	QSPI frame format: Selects data frame format for transmitting/receiving the data bits. **Value:** • 0x0 (STD_SPI_FRF): Standard SPI frame format • 0x1 (DUAL_SPI_FRF): Dual SPI frame format • 0x2 (QUAD_SPI_FRF): Quad SPI frame format • 0x3: Reserved
21:15	RSVD	R		Reserved bits
14	SSTEN	RW	0x0	Slave select toggle enable. When operating in SPI mode with clock phase (SCPH) set to 0, this register controls the behavior of the slave select line (CS) between data frames. If this register field is set to 1, the CS line will toggle between consecutive data frames, with the serial clock (SCLK) being held to its default value while CS is high; if this register field is set to 0, the CS will stay low and SCLK will run continuously for the duration of the transfer. **Note:**

Bits	Field Name	RW	Reset	Description
				When the SPI is configured as a slave, this register serves no purpose.
13	SRL	RW	0x0	Shift register loop. Used for testing purposes only. When internally active, connects the transmit shift register output to the receive shift register input. **Value:** • 0x0 (NORMAL_MODE): Normal mode operation • 0x1 (TESTING_MODE): Test mode: TX & RX shift reg connected
12	SLVOE	RW	0x0	Slave output enable. **Value:** • 0x0 (ENABLED): Slave output is enabled. • 0x1 (DISABLED): Slave output is disabled.
11:10	TMOD	RW	0x0	Transfer mode. This transfer mode is only valid when the SPI is configured as master device. • 0x0: Transmit & receive • 0x1: Transmit only • 0x2: Receive only • 0x3: EEPROM read **Value:** • 0x0 (TX_AND_RX): Transmit & receive • 0x1 (TX_ONLY): Transmit only mode • 0x2 (RX_ONLY): Receive only mode • 0x3 (EEPROM_READ): EEPROM read mode
9	SCPOL	RW	0x0	Serial clock polarity. Valid when the frame format (FRF) is set to Motorola SPI. Used to select the polarity of the inactive serial clock, which is held inactive when the SPI master is not actively transferring data on the serial bus. **Value:** • 0x0 (SCLK_LOW): Inactive state of serial clock is low • 0x1 (SCLK_HIGH): Inactive state of serial clock is high

Bits	Field Name	RW	Reset	Description
8	SCPHA	RW	0x0	Serial clock phase. Valid when the frame format (FRF) is set to Motorola SPI. The serial clock phase selects the relationship of the serial clock with the slave select signal. When SCPH = 0, data is captured on the first edge of the serial clock. When SCPH = 1, the serial clock starts toggling one cycle after the slave select line is activated, and data is captured on the second edge of the serial clock. **Value:** • 0x0 (SCPH_MIDDLE): Serial clock toggles in middle of first data bit • 0x1 (SCPH_START): Serial clock toggles at start of first data bit
7:6	FRF	RW	0x0	Frame format. Selects which serial protocol transfers the data. **Value:** • 0x0 (MOTOROLA_SPI): Motorolla SPI frame format • 0x1 (TEXAS_SSP): Texas Instruments SSP frame format • 0x2 (NS_MICROWIRE): National Microwire frame format • 0x3 (RESERVED): Reserved value
5	RSVD	R		Reserved bits
4:0	DFS	RW	0x7	Data frame size in 32-bit transfer size mode. Used to select the data frame size in 32-bit transfer mode. When the data frame size is programmed to be less than 32 bits, the receive data is automatically right-justified by the receive logic, with the upper bits of the receive FIFO zero-padded. Users are responsible for making sure that transmit data is right-justified before writing into the transmit FIFO. The transmit logic ignores the upper unused bits when transmitting the data. **Value:** • 0x3 (FRAME_04BITS): 4-bit serial data transfer • 0x4 (FRAME_05BITS): 5-bit serial data transfer • 0x5 (FRAME_06BITS): 6-bit serial data transfer • 0x6 (FRAME_07BITS): 7-bit serial data transfer

Bits	Field Name	RW	Reset	Description
				• 0x7 (FRAME_08BITS): 8-bit serial data transfer • 0x8 (FRAME_09BITS): 9-bit serial data transfer • 0x9 (FRAME_10BITS): 10-bit serial data transfer • 0xa (FRAME_11BITS): 11-bit serial data transfer • 0xb (FRAME_12BITS): 12-bit serial data transfer • 0xc (FRAME_13BITS): 13-bit serial data transfer • 0xd (FRAME_14BITS): 14-bit serial data transfer • 0xe (FRAME_15BITS): 15-bit serial data transfer • 0xf (FRAME_16BITS): 16-bit serial data transfer • 0x10 (FRAME_17BITS): 17-bit serial data transfer • 0x11 (FRAME_18BITS): 18-bit serial data transfer • 0x12 (FRAME_19BITS): 19-bit serial data transfer • 0x13 (FRAME_20BITS): 20-bit serial data transfer • 0x14 (FRAME_21BITS): 21-bit serial data transfer • 0x15 (FRAME_22BITS): 22-bit serial data transfer • 0x16 (FRAME_23BITS): 23-bit serial data transfer • 0x17 (FRAME_24BITS): 24-bit serial data transfer • 0x18 (FRAME_25BITS): 25-bit serial data transfer • 0x19 (FRAME_26BITS): 26-bit serial data transfer • 0x1a (FRAME_27BITS): 27-bit serial data transfer • 0x1b (FRAME_28BITS): 28-bit serial data transfer • 0x1c (FRAME_29BITS): 29-bit serial data transfer • 0x1d (FRAME_30BITS): 30-bit serial data transfer • 0x1e (FRAME_31BITS): 31-bit serial data transfer • 0x1f (FRAME_32BITS): 32-bit serial data transfer

4.4.5.4.2 CTRL1

- **Name:** QSPI Control Register 1
- **Description:** Control register 1 controls the end of serial transfers in receive only mode. It is impossible to write to this register when the QSPI is enabled.
- **Base Address:** QSPI0: 0x40021000; QSPI1: 0x40022000; QSPI2: 0x40023000
- **Offset:** 0x04
- **Reset Value:** 0x00000000

Table 4-208 QSPI Control Register 1

Bits	Field Name	RW	Reset	Description
31:16	RSVD	R		Reserved bits
15:0	NDF	RW	0x0	Number of data frames. When CTRL0[11:10] = 0x2 or CTRL0[11:10] = 0x3, this register field sets the number of data frames to be continuously received by the QSPI. The QSPI continues to receive serial data until the number of data frames received is equal to this register value plus 1, which enables users to receive up to 64 KB of data in a continuous transfer.

4.4.5.4.3 QSPI_EN

- **Name:** QSPI Enable Register
- **Description:** This register enables and disables the QSPI.
- **Base Address:** QSPI0: 0x40021000; QSPI1: 0x40022000; QSPI2: 0x40023000
- **Offset:** 0x08
- **Reset Value:** 0x00000000

Table 4-209 QSPI Enable Register

Bits	Field Name	RW	Reset	Description
31:1	RSVD	R		Reserved bits
0	SSI_EN	RW	0x0	QSPI enable. Enables and disables all QSPI operations. When disabled, all serial transfers are halted immediately. Transmit and receive FIFO buffers are cleared when the device is disabled. **Value:** • 0x0 (DISABLE): Disable serial transfer • 0x1 (ENABLED): Enable serial transfer

4.4.5.4.4 MWC

- **Name:** Microwire Control Register
- **Description:** This register controls the direction of the data word for the half-duplex Microwire serial protocol. It is impossible to write to this register when the QSPI is enabled.
- **Base Address:** QSPI0: 0x40021000; QSPI1: 0x40022000; QSPI2: 0x40023000
- **Offset:** 0x0C

- **Reset Value:** 0x00000000

Table 4-210 Microwire Control Register

Bits	Field Name	RW	Reset	Description
31:3	RSVD	R		Reserved bits
2	MHS	RW	0x0	Microwire handshaking. Used to enable and disable the busy/ready handshaking interface for the Microwire protocol. When this bit is enabled, the QSPI checks for a ready status from the target slave, after the transfer of the last data/control bit, before clearing the BUSY status in the STAT register. **Value:** • 0x0 (DISABLE): Handshaking interface is disabled • 0x1 (ENABLED): Handshaking interface is enabled
1	MDD	RW	0x0	Microwire control. Defines the direction of the data word when the Microwire serial protocol is used. **Value:** • 0x0 (RECEIVE): QSPI receives data • 0x1 (TRANSMIT): QSPI transmits data
0	MWMOD	RW	0x0	Microwire transfer mode. Defines whether the Microwire transfer is sequential or non-sequential. When sequential mode is used, only one control word is needed to transmit or receive a block of data words. When non-sequential mode is used, there must be a control word for each data word that is transmitted or received. **Value:** • 0x0 (NON_SEQUENTIAL): Non-sequential Microwire transfer • 0x1 (SEQUENTIAL): Sequential Microwire transfer

4.4.5.4.5 SE

- **Name:** Slave Enable Register
- **Description:** The register enables the individual slave select output lines from the QSPI master.
- **Base Address:** QSPI0: 0x40021000; QSPI1: 0x40022000; QSPI2: 0x40023000
- **Offset:** 0x10

- **Reset Value:** 0x00000000

Table 4-211 Slave Enable Register

Bits	Field Name	RW	Reset	Description
31:1	RSVD	R		Reserved bits
0	SLAVE0	RW	0x0	Slave select enable. Each bit in this register corresponds to a slave select line (CSn) from the QSPI master. **Value:** • 0x0 (NOT_SELECTED): No slave selected • 0x1 (SELECTED): Slave is selected **Note:** When the QSPI is configured as a slave, this register serves no purpose.

4.4.5.4.6 BAUD

- **Name:** Baud Rate Register
- **Description:** The register derives the frequency of the serial clock that regulates the data transfer. It is impossible to write to this register when the QSPI is enabled.
- **Base Address:** QSPI0: 0x40021000; QSPI1: 0x40022000; QSPI2: 0x40023000
- **Offset:** 0x14
- **Reset Value:** 0x00000000

Table 4-212 Baud Rate Register

Bits	Field Name	RW	Reset	Description
31:16	RSVD	R		Reserved bits
15:0	SCLKDIV	RW	0x0	QSPI clock divider. The LSB for this field is always set to 0 and is unaffected by a write operation, which ensures an even value is held in this register. If the value is 0, the serial output clock (sclk_out) is disabled. The frequency of the sclk_out is derived from the following equation: Fsclk_out = Fssi_clk/SCKDV

4.4.5.4.7 TX_FTL

- **Name:** Transmit FIFO Threshold Level Register

- **Description:** This register controls the threshold value for the transmit FIFO memory.
- **Base Address:** QSPI0: 0x40021000; QSPI1: 0x40022000; QSPI2: 0x40023000
- **Offset:** 0x18
- **Reset Value:** 0x00000000

Table 4-213 Transmit FIFO Threshold Level Register

Bits	Field Name	RW	Reset	Description
31:5	RSVD	R		Reserved bits
20:16	TXFTHR	RW	0x0	Transfer start FIFO level. Used to control the level of entries in transmit FIFO above which transfer will start on serial line. This register can be used to ensure that sufficient data is present in transmit FIFO before starting a write operation on serial line.
15:5	RSVD	R		Reserved bits
4:0	TFT	RW	0x0	Transmit FIFO threshold. Controls the level of entries at (or below) which the transmit FIFO controller triggers an interrupt. The FIFO depth is configurable in 16/32 words. This register is sized to the number of address bits needed to access the FIFO. If users attempt to set this value greater than or equal to the depth of the FIFO, this field is not written and retains its current value. When the number of transmit FIFO entries is less than or equal to this value, the transmit FIFO empty interrupt is triggered.

4.4.5.4.8 RX_FTL

- **Name:** Receive FIFO Threshold Level Register
- **Description:** This register controls the threshold value for the receive FIFO memory.
- **Base Address:** QSPI0: 0x40021000; QSPI1: 0x40022000; QSPI2: 0x40023000
- **Offset:** 0x1C
- **Reset Value:** 0x00000000

Table 4-214 Receive FIFO Threshold Level Register

Bits	Field Name	RW	Reset	Description
31:5	RSVD	R		Reserved bits
4:0	RFT	RW	0x0	Receive FIFO threshold.

Bits	Field Name	RW	Reset	Description
				Controls the level of entries at (or above) which the receive FIFO controller triggers an interrupt. The FIFO depth is configurable in 16/32 words. This register is sized to the number of address bits needed to access the FIFO. If users attempt to set this value greater than the depth of the FIFO, this field is not written and retains its current value. When the number of receive FIFO entries is greater than or equal to this value + 1, the receive FIFO full interrupt is triggered.

4.4.5.4.9 TX_FL

- **Name:** Transmit FIFO Level Register
- **Description:** This register contains the number of valid data entries in the transmit FIFO memory.
- **Base Address:** QSPI0: 0x40021000; QSPI1: 0x40022000; QSPI2: 0x40023000
- **Offset:** 0x20
- **Reset Value:** 0x00000000

Table 4-215 Transmit FIFO Level Register

Bits	Field Name	RW	Reset	Description
31:6	RSVD	R		Reserved bits
5:0	TX_FL	R	0x0	Transmit FIFO level. Contains the number of valid data entries in the transmit FIFO.

4.4.5.4.10 RX_FL

- **Name:** Receive FIFO Level Register
- **Description:** This register contains the number of valid data entries in the receive FIFO memory. This register can be ready at any time.
- **Base Address:** QSPI0: 0x40021000; QSPI1: 0x40022000; QSPI2: 0x40023000
- **Offset:** 0x24
- **Reset Value:** 0x00000000

Table 4-216 Receive FIFO Level Register

Bits	Field Name	RW	Reset	Description
31:6	RSVD	R		Reserved bits

Bits	Field Name	RW	Reset	Description
5:0	RX_FL	R	0x0	Receive FIFO level. Contains the number of valid data entries in the receive FIFO.

4.4.5.4.11 STAT

- **Name:** Status Register
- **Description:** This is a read-only register used to indicate the current transfer status, FIFO status, and any transmission/reception errors that may have occurred. The status register may be read at any time.
- **Base Address:** QSPI0: 0x40021000; QSPI1: 0x40022000; QSPI2: 0x40023000
- **Offset:** 0x28
- **Reset Value:** 0x00000006

Table 4-217 Status Register

Bits	Field Name	RW	Reset	Description
31:7	RSVD	R		Reserved bits
6	DCOL	R	0x0	Data collision error. This bit will be set if MISO input is asserted by other master, when the QSPI master is in the middle of the transfer. This informs the processor that the last data transfer was halted before completion. This bit is cleared when read.
5	TXE	R	0x0	Transmission Error. Set if the transmit FIFO is empty when a transfer is started. Data from the previous transmission is resent on the TXD line. This bit is cleared when read. **Value:** • 0x0 (NO_ERROR): No error • 0x1 (TX_ERROR): Transmission error
4	RFF	R	0x0	Receive FIFO full. When the receive FIFO is completely full, this bit is set. When the receive FIFO contains one or more empty locations, this bit is cleared. **Value:** • 0x0 (NOT_FULL): Receive FIFO is not full • 0x1 (FULL): Receive FIFO is full

Bits	Field Name	RW	Reset	Description
3	RFNE	R	0x0	Receive FIFO not empty. Set when the receive FIFO contains one or more entries and is cleared when the receive FIFO is empty. This bit can be polled by software to completely empty the receive FIFO. **Value:** • 0x0 (EMPTY): Receive FIFO is empty • 0x1 (NOT_EMPTY): Receive FIFO is not empty
2	TFE	R	0x1	Transmit FIFO empty. When the transmit FIFO is completely empty, this bit is set. When the transmit FIFO contains one or more valid entries, this bit is cleared. This bit field does not request an interrupt. **Value:** • 0x0 (NOT_EMPTY): Transmit FIFO is not empty • 0x1 (EMPTY): Transmit FIFO is empty
1	TFNF	R	0x1	Transmit FIFO not full. Set when the transmit FIFO contains one or more empty locations, and is cleared when the FIFO is full. **Value:** • 0x0 (FULL): Transmit FIFO is full • 0x1 (NOT_FULL): Transmit FIFO is not Full
0	BUSY	R	0x0	QSPI busy flag. This bit being set indicates that a serial transfer is in progress; this bit being cleared indicates that the QSPI is idle or disabled. **Value:** • 0x0 (INACTIVE): QSPI is idle or disabled • 0x1 (ACTIVE): QSPI is actively transferring data

4.4.5.4.12 INTMASK

- **Name:** Interrupt Mask Register
- **Description:** This read/write register masks or enables all interrupts generated by the QSPI. When the QSPI is configured as a slave device, the MSTIM bit field is not present. This changes the reset value from 0x3F for serial-master configurations to 0x1F for serial-slave configurations.

- **Base Address:** QSPI0: 0x40021000; QSPI1: 0x40022000; QSPI2: 0x40023000
- **Offset:** 0x2C
- **Reset Value:** master ? 0x0000003F: 0x0000001F

Table 4-218 Interrupt Mask Register

Bits	Field Name	RW	Reset	Description
31:6	RSVD	R		Reserved bits
5	MSTIM	RW	0x1	Multi-master contention interrupt mask. **Value:** • 0x0 (MASKED): Disable mst_intr interrupt • 0x1 (UNMASKED): Enable mst_intr interrupt
4	RXFIM	RW	0x1	Receive FIFO full interrupt mask **Value:** • 0x0 (MASKED): Disable rxf_intr interrupt • 0x1 (UNMASKED): Enable rxf_intr interrupt
3	RXOIM	RW	0x1	Receive FIFO overflow interrupt mask **Value:** • 0x0 (MASKED): Disable rxo_intr interrupt • 0x1 (UNMASKED): Enable rxo_intr interrupt
2	RXUIM	RW	0x1	Receive FIFO underflow interrupt mask **Value:** • 0x0 (MASKED): Disable rxu_intr interrupt • 0x1 (UNMASKED): Enable rxu_intr interrupt
1	TXOIM	RW	0x1	Transmit FIFO overflow interrupt mask **Value:** • 0x0 (MASKED): Disable txo_intr interrupt • 0x1 (UNMASKED): Enable txo_intr interrupt
0	TXEIM	RW	0x1	Transmit FIFO empty interrupt mask **Value:** • 0x0 (MASKED): Disable txe_intr interrupt • 0x1 (UNMASKED): Enable txe_intr interrupt

4.4.5.4.13 INTSTAT

- **Name:** Interrupt Status Register
- **Description:** This register reports the status of the QSPI interrupts after they have been enabled.

- **Base Address:** QSPI0: 0x40021000; QSPI1: 0x40022000; QSPI2: 0x40023000
- **Offset:** 0x30
- **Reset Value:** 0x00000000

Table 4-219 Interrupt Status Register

Bits	Field Name	RW	Reset	Description
31:6	RSVD	R		Reserved bits
5	MSTIS	R	0x0	Multi-master contention interrupt status **Value:** • 0x0 (INACTIVE): mst_intr interrupt is not active after being enabled • 0x1 (ACTIVE): mst_intr interrupt is full after being enabled
4	RXFIS	R	0x0	Receive FIFO full interrupt status **Value:** • 0x0 (INACTIVE): rxf_intr interrupt is not active after being enabled • 0x1 (ACTIVE): rxf_intr interrupt is full after being enabled
3	RXOIS	R	0x0	Receive FIFO overflow interrupt status **Value:** • 0x0 (INACTIVE): rxo_intr interrupt is not active after being enabled • 0x1 (ACTIVE): rxo_intr interrupt is active after being enabled
2	RXUIS	R	0x0	Receive FIFO underflow interrupt status **Value:** • 0x0 (INACTIVE): rxu_intr interrupt is not active after being enabled • 0x1 (ACTIVE): rxu_intr interrupt is active after being enabled
1	TXOIS	R	0x0	Transmit FIFO overflow interrupt status **Value:** • 0x0 (INACTIVE): txo_intr interrupt is not active after being enabled • 0x1 (ACTIVE): txo_intr interrupt is active after being enabled
0	TXEIS	R	0x0	Transmit FIFO empty interrupt status **Value:**

Bits	Field Name	RW	Reset	Description
				• 0x0 (INACTIVE): txe_intr interrupt is not active after being enabled • 0x1 (ACTIVE): txe_intr interrupt is active after being enabled

4.4.5.4.14 RAW_INTSTAT

- **Name:** Raw Interrupt Status Register
- **Description:** This read-only register reports the status of the QSPI interrupts prior to being enabled.
- **Base Address:** QSPI0: 0x40021000; QSPI1: 0x40022000; QSPI2: 0x40023000
- **Offset:** 0x34
- **Reset Value:** 0x00000000

Table 4-220 Raw Interrupt Status Register

Bits	Field Name	RW	Reset	Description
31:6	RSVD	R		Reserved bits
5	MSTIR	R	0x0	Multi-master contention raw interrupt status **Value:** • 0x0 (INACTIVE): mst_intr interrupt is not active prior to being enabled • 0x1 (ACTIVE): mst_intr interrupt is full prior to being enabled
4	RXFIR	R	0x0	Receive FIFO full raw interrupt status **Value:** • 0x0 (INACTIVE): rxf_intr interrupt is not active prior to being enabled • 0x1 (ACTIVE): rxf_intr interrupt is active prior to being enabled
3	RXOIR	R	0x0	Receive FIFO overflow raw interrupt status **Value:** • 0x1 (ACTIVE): rxo_intr interrupt is not active prior to being enabled • 0x0 (INACTIVE): rxo_intr interrupt is active prior to being enabled
2	RXUIR	R	0x0	Receive FIFO underflow raw interrupt status **Value:**

Bits	Field Name	RW	Reset	Description
				• 0x0 (INACTIVE): rxu_intr interrupt is not active prior to being enabled • 0x1 (ACTIVE): rxu_intr interrupt is active prior to being enabled
1	TXOIR	R	0x0	Transmit FIFO overflow raw interrupt status **Value:** • 0x0 (INACTIVE): txo_intr interrupt is not active prior to being enabled • 0x1 (ACTIVE): txo_intr interrupt is active prior to being enabled
0	TXEIR	R	0x0	Transmit FIFO empty raw interrupt status **Value:** • 0x0 (INACTIVE): txe_intr interrupt is not active prior to being enabled • 0x1 (ACTIVE): txe_intr interrupt is active prior to being enabled

4.4.5.4.15 TXEIC

- **Name:** Transmit FIFO Error Interrupt Clear Register
- **Description:** This register is used to clear the transmit FIFO error interrupt.
- **Base Address:** QSPI0: 0x40021000; QSPI1: 0x40022000; QSPI2: 0x40023000
- **Offset:** 0x38
- **Reset Value:** 0x00000000

Table 4-221 Transmit FIFO Error Interrupt Clear Register

Bits	Field Name	RW	Reset	Description
31:1	RSVD	R		Reserved bits
0	TXEIC	R	0x0	Clear transmit FIFO overflow/underflow interrupt. This register reflects the status of the interrupt. A read from this register clears the txo_intr interrupt; writing has no effect.

4.4.5.4.16 RXOIC

- **Name:** Receive FIFO Overflow Interrupt Clear Register
- **Description:** This register is used to clear the receive FIFO overflow interrupt.

- **Base Address:** QSPI0: 0x40021000; QSPI1: 0x40022000; QSPI2: 0x40023000
- **Offset:** 0x3C
- **Reset Value:** 0x00000000

Table 4-222 Receive FIFO Overflow Interrupt Clear Register

Bits	Field Name	RW	Reset	Description
31:1	RSVD	R		Reserved bits
0	RXOIC	R	0x0	Clear receive FIFO overflow interrupt. This register reflects the status of the interrupt. A read from this register clears the rxo_intr interrupt; writing has no effect.

4.4.5.4.17 RXUIC

- **Name:** Receive FIFO Underflow Interrupt Clear Register
- **Description:** This register is used to clear the receive FIFO underflow interrupt.
- **Base Address:** QSPI0: 0x40021000; QSPI1: 0x40022000; QSPI2: 0x40023000
- **Offset:** 0x40
- **Reset Value:** 0x00000000

Table 4-223 Receive FIFO Underflow Interrupt Clear Register

Bits	Field Name	RW	Reset	Description
31:1	RSVD	R		Reserved bits
0	RXUIC	R		Clear receive FIFO underflow interrupt. This register reflects the status of the interrupt. A read from this register clears the rxu_intr interrupt; writing has no effect.

4.4.5.4.18 MSTIC

- **Name:** Multi-Master Interrupt Clear Register
- **Description:** This register is used to clear the multi-master interrupt.
- **Base Address:** QSPI0: 0x40021000; QSPI1: 0x40022000; QSPI2: 0x40023000
- **Offset:** 0x44
- **Reset Value:** 0x00000000

Table 4-224 Multi-Master Interrupt Clear Register

Bits	Field Name	RW	Reset	Description
31:1	RSVD	R		Reserved bits
0	MSTIC	R		Clear multi-master contention interrupt. This register reflects the status of the interrupt. A read from this register clears the mst_intr interrupt; writing has no effect.

4.4.5.4.19 INTCLR

- **Name:** Interrupt Clear Register
- **Description:** This register is used to clear interrupts.
- **Base Address:** QSPI0: 0x40021000; QSPI1: 0x40022000; QSPI2: 0x40023000
- **Offset:** 0x48
- **Reset Value:** 0x00000000

Table 4-225 Interrupt Clear Register

Bits	Field Name	RW	Reset	Description
31:1	RSVD	R		Reserved bits
0	INTCLR	R		Clear interrupts. This register is set if any of the interrupts below are active. A read clears the txo_intr, rxu_intr, rxo_intr, and the mst_intr interrupts. Writing to this register has no effect.

4.4.5.4.20 DMAC

- **Name:** DMA Control Register
- **Description:** The register is used to enable the DMA Controller interface operation.
- **Base Address:** QSPI0: 0x40021000; QSPI1: 0x40022000; QSPI2: 0x40023000
- **Offset:** 0x4C
- **Reset Value:** 0x00000000

Table 4-226 DMA Control Register

Bits	Field Name	RW	Reset	Description
31:2	RSVD	R		Reserved bits
1	TDMAE	RW		Transmit DMA enable. This bit enables/disables the TX FIFO DMA channel.

Bits	Field Name	RW	Reset	Description
				Value: • 0x0 (DISABLE): Transmit DMA disabled • 0x1 (ENABLED): Transmit DMA enabled
0	RDMAE	RW		Receive DMA enable. This bit enables/disables the receive FIFO DMA channel. **Value:** • 0x0 (DISABLE): Receive DMA disabled • 0x1 (ENABLED): Receive DMA enabled

4.4.5.4.21 DMA_TDL

- **Name:** DMA Transmit Data Level Register
- **Description:** This register controls the threshold value for the TX FIFO memory.
- **Base Address:** QSPI0: 0x40021000; QSPI1: 0x40022000; QSPI2: 0x40023000
- **Offset:** 0x50
- **Reset Value:** 0x00000000

Table 4-227 DMA Transmit Data Level Register

Bits	Field Name	RW	Reset	Description
31:5	RSVD	R		Reserved bits
4:0	DMA_TDL	RW		Transmit data level. This bit field controls the level at which a DMA request is made by the transmit logic. It is equal to the watermark level; that is, the dma_tx_req signal is generated when the number of valid data entries in the TX FIFO is equal to or below this field value, and TDMAE = 1.

4.4.5.4.22 DMA_RDL

- **Name:** DMA Receive Data Level Register
- **Description:** This register controls the threshold value for the receive FIFO memory.
- **Base Address:** QSPI0: 0x40021000; QSPI1: 0x40022000; QSPI2: 0x40023000
- **Offset:** 0x54
- **Reset Value:** 0x00000000

Table 4-228 DMA Receive Data Level Register

Bits	Field Name	RW	Reset	Description
31:5	RSVD	R		Reserved bits
4:0	DMA_RDL	RW		Receive data level. This bit field controls the level at which a DMA request is made by the receive logic. The watermark level = DMA_RDL +1; that is, dma_rx_req is generated when the number of valid data entries in the receive FIFO is equal to or above this field value + 1, and RDMAE=1.

4.4.5.4.23 DATA

- **Name:** Data Register
- **Description:** The QSPI data register is a 32-bit read/write buffer for the TX/RX FIFOs. When the register is read, data in the RX FIFO buffer is accessed. When it is written to, data is moved into the TX FIFO buffer; a write can occur only when SSI_EN[0] = 1. FIFOs are reset when SSI_EN[0] = 0.
- **Base Address:** QSPI0: 0x40021000; QSPI1: 0x40022000; QSPI2: 0x40023000
- **Offset:** 0x60
- **Reset Value:** 0x00000000

Table 4-229 Data Register

Bits	Field Name	RW	Reset	Description
31:0	DATA	RW		Data register. When writing to this register, users must right-justify the data. Read data is automatically right-justified.

4.4.5.4.24 RX_SMP_DLY

- **Name:** Receive Sample Delay Register
- **Description:** This register controls the number of SCLK cycles that are delayed (from the default sample time) before the actual sample of the RXD input occurs. It is impossible to write to this register when the QSPI is enabled.
- **Base Address:** QSPI0: 0x40021000; QSPI1: 0x40022000; QSPI2: 0x40023000
- **Offset:** 0xF0
- **Reset Value:** 0x00000000

Table 4-230 Receive Sample Delay Register

Bits	Field Name	RW	Reset	Description
3:0	RX_SAMPLEDLY	RW		Receive sample delay

4.4.5.4.25 SPI_CTRL0

- **Name:** QSPI Enhanced Control Register 0
- **Description:** This register is used to control the serial data transfer in enhanced SPI mode of operation. The register is relevant only when CTRL0.SPI_FRF is set to either 01 or 10 or 11. It is impossible to write to this register when the QSPI is enabled.
- **Base Address:** QSPI0: 0x40021000; QSPI1: 0x40022000; QSPI2: 0x40023000
- **Offset:** 0xF4
- **Reset Value:** 0x00000000

Table 4-231 QSPI Enhanced Control Register 0

Bits	Field Name	RW	Reset	Description
31	RSVD	R		Reserved bits
30	CLK_STRETCH_EN	RW	0x0	Enables clock stretching capability in QSPI transfers. • In case of write operations, if the transmit FIFO becomes empty, DWC_ssi will stretch the clock until the FIFO has enough data to continue the transfer. • In case of read operations, if the receive FIFO becomes full, DWC_ssi will stop the clock until data has been read from the FIFO.
29:16	RSVD	R		Reserved bits
15:11	WAITCYCLES	RW	0x0	Wait cycles in Dual/Quad mode between control frames transmit and data reception. Specified as number of QSPI clock cycles.
10	RSVD	R		Reserved bit
9:8	INSTL	RW	0x0	Dual/Quad mode instruction length in bits. **Value:** • 0x0 (INST_L0): No instruction • 0x1 (INST_L4): 4-bit instruction length • 0x2 (INST_L8): 8-bit instruction length • 0x3 (INST_L16): 16-bit instruction length
7:6	RSVD	R		Reserved bits

Bits	Field Name	RW	Reset	Description
5:2	ADDRL	RW	0x0	These bits define the length of address to be transmitted. Only after these bits are programmed into the FIFO, the transfer can begin. **Value:** • 0x0 (ADDR_L0): No address • 0x1 (ADDR_L4): 4-bit address length • 0x2 (ADDR_L8): 8-bit address length • 0x3 (ADDR_L12): 12-bit address length • 0x4 (ADDR_L16): 16-bit address length • 0x5 (ADDR_L20): 20-bit address length • 0x6 (ADDR_L24): 24-bit address length • 0x7 (ADDR_L28): 28-bit address length • 0x8 (ADDR_L32): 32-bit address length
1:0	TRANSTYPE	RW	0x0	Address and instruction transfer format. Selects whether QSPI will transmit instruction/address either in standard SPI mode or the SPI mode selected in CTRL0.SPI_FRF field. **Value:** • 0x0: Instruction and address will be sent in standard SPI Mode. • 0x1: Instruction will be sent in standard SPI mode and address will be sent in the mode specified by CTRL0.SPI_FRF. • 0x2: Both instruction and address will be sent in the mode specified by SPI_FRF. • 0x3: Reserved.

4.4.5.4.26 XIP_MODE_BITS

- **Name:** XIP Mode Bits Register
- **Description:** This register carries the mode bits which are sent in the XIP mode of operation after address phase. This is an 8-bit register and can only be written when SE register is set to 0.
- **Base Address:** QSPI0: 0x40021000; QSPI1: 0x40022000; QSPI2: 0x40023000
- **Offset:** 0xFC
- **Reset Value:** 0x00000000

Table 4-232 XIP Mode Bits Register

Bits	Field Name	RW	Reset	Description
15:0	MODE_BITS	RW		XIP mode bits to be sent after address phase of XIP transfer

4.4.5.4.27 XIP_INCR_INST

- **Name:** XIP INCR Transfer Opcode Register
- **Description:** This register is used to store the instruction op-code to be used in INCR transactions when the same is requested on AHB interface. It is impossible to write to this register when the QSPI is enabled.
- **Base Address:** QSPI0: 0x40021000; QSPI1: 0x40022000; QSPI2: 0x40023000
- **Offset:** 0x100
- **Reset Value:** 0x00000000

Table 4-233 XIP INCR Transfer Opcode Register

Bits	Field Name	RW	Reset	Description
15:0	XIP_INCR_INST	RW	0x00	XIP INCR transfer opcode. When XIP_CTRL.INST_EN bit is set to 1, QSPI sends instruction for all XIP transfers, this register field stores the instruction op-code to be sent when an INCR type transfer is requested on AHB bus. The number of bits to be sent in instruction phase is determined by XIP_CTRL.INST_L field.

4.4.5.4.28 XIP_WRAP_INST

- **Name:** XIP WRAP Transfer Opcode Register
- **Description:** This register is used to store the instruction op-code to be used in WRAP transactions when the same is requested on AHB interface. It is impossible to write to this register when the QSPI is enabled.
- **Base Address:** QSPI0: 0x40021000; QSPI1: 0x40022000; QSPI2: 0x40023000
- **Offset:** 0x104
- **Reset Value:** 0x00000000

Table 4-234 XIP WRAP Transfer Opcode Register

Bits	Field Name	RW	Reset	Description
15:0	XIP_WRAP_INST	RW	0x00	XIP WRAP transfer opcode. When XIP_CTRL.INST_EN bit is set to 1, QSPI sends instruction for all XIP transfers, this register field stores the instruction op-code to be sent when a WRAP type transfer

Bits	Field Name	RW	Reset	Description
				is requested on AHB bus. The number of bits to be sent in instruction phase is determined by XIP_CTRL.INST_L field.

4.4.5.4.29 XIP_CTRL

- **Name:** XIP Control Register
- **Description:** This register is used to store the control information for the XIP transfer. It is impossible to write to this register when the QSPI is enabled.
- **Base Address:** QSPI0: 0x40021000; QSPI1: 0x40022000; QSPI2: 0x40023000
- **Offset:** 0x108
- **Reset Value:** 0x00000000

Table 4-235 XIP Control Register

Bits	Field Name	RW	Reset	Description
31:30	RSVD	R		Reserved bits
29	XIP_PREFETCH_EN	RW	0x0	Enables XIP pre-fetch functionality in QSPI. Once the functionality is enabled, QSPI will pre-fetch data frames from next contiguous location, to reduce the latency for the upcoming contiguous transfer. If the next XIP request is not contiguous, then pre-fetched bits will be discarded.
28	RSVD	R		Reserved bit
27:26	XIP_MBL	RW	0x0	XIP Mode bits length. Sets the length of mode bits in XIP mode of operation. These bits are valid only when XIP_CTRL.MD_BITS_EN is set to 1. **Value:** • 0x0 : Mode bits length equal to 2 • 0x1 : Mode bits length equal to 4 • 0x2 : Mode bits length equal to 8 • 0x3 : Mode bits length equal to 16
25:24	RSVD	R		Reserved bits
23	CONT_XFER_EN	RW	0x0	Enable continuous transfer in XIP mode. If this bit is set to 1, then continuous transfer mode in XIP will be enabled, in this mode QSPI will keep slave selected until a non-XIP transfer is detected on the AHB interface.

Bits	Field Name	RW	Reset	Description
22	INST_EN	RW	0x0	XIP instruction enable bit. If this bit is set to 1, then XIP transfers will also have instruction phase. The instruction op-codes will be chosen from XIP_INCR_INST or XIP_WRAP_INST register based on AHB transfer type.
21:19	RSVD	R		Reserved bits
18	DFS_HC	RW	0x0	Fix DFS for XIP transfers. If this bit is set to 1, then data frame size for XIP transfers will be fixed to the programmed value in CTRL0.DFS. The number of data frames to fetch will be determined by HSIZE and HBURST signals. If this bit is set to 0 then data frame size and number of data frames to fetch will be determined by HSIZE and HBURST signals.
17:13	WAIT_CYCLES	RW	0x0	Wait cycles in Dual/Quad mode between control frames transmit and data reception. Specified as number of QSPI clock cycles.
12	MD_BITS_EN	RW	0x0	Mode bits enable in XIP mode. If this bit is set to 1, then in XIP mode of operation, QSPI will insert mode bits after the address phase. These bits are set in XIP_MODE_BITS register. The length of mode bits is always set to 8 bits.
11	RSVD	R		Reserved bit
10:9	INST_L	RW	0x0	Dual/Quad mode instruction length in bits. **Value:** • 0x0 (INST_L0): No instruction • 0x1 (INST_L4): 4-bit instruction length • 0x2 (INST_L8): 8-bit instruction length • 0x3 (INST_L16): 16-bit instruction length
8	RSVD	R		Reserved bit
7:4	ADDR_L	RW	0x0	These bits define the length of address to be transmitted. Only after these bits are programmed into the FIFO, the transfer can begin. **Value:** • 0x0 (ADDR_L0): No address • 0x1 (ADDR_L4): 4-bit address length • 0x2 (ADDR_L8): 8-bit address length • 0x3 (ADDR_L12): 12-bit address length

Bits	Field Name	RW	Reset	Description
				• 0x4 (ADDR_L16): 16-bit address length • 0x5 (ADDR_L20): 20-bit address length • 0x6 (ADDR_L24): 24-bit address length • 0x7 (ADDR_L28): 28-bit address length • 0x8 (ADDR_L32): 32-bit address length
3:2	TRANS_TYPE	RW	0x0	Address and instruction transfer format. Selects whether QSPI will transmit instruction/address either in standard SPI mode or the SPI mode selected in CTRL0.SPI_FRF field. **Value:** • 0x0: Instruction and address will be sent in standard SPI Mode. • 0x1: Instruction will be sent in standard SPI mode and address will be sent in the mode specified by CTRL0.SPI_FRF. • 0x2: Both instruction and address will be sent in the mode specified by SPI_FRF. • 0x3: Reserved.
1:0	FRF	RW	0x0	QSPI frame format. Selects data frame format for transmitting/receiving the data. **Value:** • 0x0 (RSVD): Reserved • 0x1 (SPI_DUAL): Dual SPI format • 0x2 (SPI_QUAD): Quad SPI format • 0x3 (RSVD): Reserved

4.4.5.4.30 XIP_SER

- **Name:** XIP Slave Enable Register
- **Description:** The register enables the individual slave select output lines from the QSPI master for XIP mode of operation. Users cannot write to this register when QSPI is busy or when QSPI is enabled.
- **Base Address:** QSPI0: 0x40021000; QSPI1: 0x40022000; QSPI2: 0x40023000
- **Offset:** 0x10C
- **Reset Value:** 0x00000000

Table 4-236 XIP Slave Enable Register

Bits	Field Name	RW	Reset	Description
31:1	RSVD	R		Reserved bits
0	SLAVE0	RW	0x0	Slave select enable flag

4.4.5.4.31 XIP_CNT_TIME_OUT

- **Name:** XIP Time Out Register for Continuous Transfers
- **Description:** XIP count down register for continuous mode. The counter is used to de-select the slave during continuous transfer mode. It is impossible to write to this register when the QSPI is enabled.
- **Base Address:** QSPI0: 0x40021000; QSPI1: 0x40022000; QSPI2: 0x40023000
- **Offset:** 0x114
- **Reset Value:** 0x00000000

Table 4-237 XIP Time Out Counter Register for Continuous Transfers

Bits	Field Name	RW	Reset	Description
31:8	RSVD	R		Reserved bits
7:0	XTOC	RW	0	XIP time out value in terms of HCLK. Once slave is selected in continuous XIP mode, this counter will be used to de-select the slave if there is no request for the time specified in the counter.

4.4.5.4.32 XIP_WR_INCR_INST

- **Name:** XIP Write INCR Transfer Opcode Register
- **Description:** This register is used to store the instruction op-code to be used in INCR transactions for XIP Write when the same is requested on AHB interface. It is impossible to write to this register when the QSPI is enabled.
- **Base Address:** QSPI0: 0x40021000; QSPI1: 0x40022000; QSPI2: 0x40023000
- **Offset:** 0x140
- **Reset Value:** 0x00000000

Table 4-238 XIP Write INCR Transfer Opcode Register

Bits	Field Name	RW	Reset	Description
15:0	XIP_INCR_INST	RW	0x00	XIP Write INCR transfer opcode. When XIP_WR_CTRL.INST_L is not equal to 0, QSPI sends instruction for all XIP write transfers, this register field stores the instruction op-code to be sent when an INCR type

Bits	Field Name	RW	Reset	Description
				XIP Write transfer is requested on AHB bus. The number of bits to be sent in instruction phase is determined by XIP_WR_CTRL.INST_L field.

4.4.5.4.33 XIP_WR_WRAP_INST

- **Name:** XIP Write WRAP Transfer Opcode Register
- **Description:** This register is used to store the instruction op-code to be used in WRAP transactions for XIP Write when the same is requested on AHB interface. It is impossible to write to this register when the QSPI is enabled.
- **Base Address:** QSPI0: 0x40021000; QSPI1: 0x40022000; QSPI2: 0x40023000
- **Offset:** 0x144
- **Reset Value:** 0x00000000

Table 4-239 XIP Write WRAP Transfer Opcode Register

Bits	Field Name	RW	Reset	Description
15:0	XIP_WRAP_INST	RW	0x00	XIP Write WRAP transfer opcode. When XIP_WR_CTRL.INST_L is not equal to 0, QSPI sends instruction for all XIP write transfers, this register field stores the instruction op-code to be sent when a WRAP type XIP Write transfer is requested on AHB bus. The number of bits to be sent in instruction phase is determined by XIP_WR_CTRL.INST_L field.

4.4.5.4.34 XIP_WR_CTRL

- **Name:** XIP Write Control Register
- **Description:** This register is used to store the control information that will be used for XIP write transfer in the xip mode. It is impossible to write to this register when the QSPI is enabled.
- **Base Address:** QSPI0: 0x40021000; QSPI1: 0x40022000; QSPI2: 0x40023000
- **Offset:** 0x148
- **Reset Value:** 0x00000000

Table 4-240 XIP Write Control Register

Bits	Field Name	RW	Reset	Description
31:21	RSVD	R		Reserved bits

Bits	Field Name	RW	Reset	Description
20:16	WR_CTRL_WAITCYCLES	RW	0x0	Wait cycles in Dual/Quad mode between control frames transmit and data reception. Specified as number of QSPI clock cycles.
15:10	RSVD	R		Reserved bits
9:8	WR_CTRL_INSTL	RW	0x0	Dual/Quad mode instruction length in bits. **Value:** • 0x0 (INST_L0): No Instruction • 0x1 (INST_L4): 4-bit Instruction length • 0x2 (INST_L8): 8-bit Instruction length • 0x3 (INST_L16): 16-bit Instruction length
7:4	WR_CTRL_ADDRL	RW	0x0	This bit defines Length of Address to be transmitted. Only after these bits are programmed into the FIFO, the transfer can begin. **Value:** • 0x0 (ADDR_L0): No Address • 0x1 (ADDR_L4): 4-bit Address length • 0x2 (ADDR_L8): 8-bit Address length • 0x3 (ADDR_L12): 12-bit Address length • 0x4 (ADDR_L16): 16-bit Address length • 0x5 (ADDR_L20): 20-bit Address length • 0x6 (ADDR_L24): 24-bit Address length • 0x7 (ADDR_L28): 28-bit Address length • 0x8 (ADDR_L32): 32-bit Address length • 0x9 (ADDR_L36): 36-bit Address length • 0xa (ADDR_L40): 40-bit Address length • 0xb (ADDR_L44): 44-bit Address length • 0xc (ADDR_L48): 48-bit Address length • 0xd (ADDR_L52): 52-bit Address length • 0xe (ADDR_L56): 56-bit Address length • 0xf (ADDR_L60): 60-bit Address length
3:2	WR_CTRL_TRANSTYPE	RW	0x0	Address and instruction transfer format. Selects whether QSPI will transmit instruction/address either in Standard SPI mode or the SPI mode selected in CTRL0.SPI_FRF field.

Bits	Field Name	RW	Reset	Description
				Value: • 0x0: Instruction and Address will be sent in Standard SPI Mode. • 0x1: Instruction will be sent in Standard SPI Mode and Address will be sent in the mode specified by CTRL0.SPI_FRF. • 0x2: Both Instruction and Address will be sent in the mode specified by SPI_FRF. • 0x3: Reserved.
1:0	WR_CTRL_FRF	RW	0x0	QSPI frame format. Selects data frame format for Transmitting/Receiving the data. **Value:** • 0x0 (RSVD): Reserved • 0x1 (SPI_DUAL): Dual SPI Format • 0x2 (SPI_QUAD): Quad SPI Format • 0x3 (RSVD): Reserved

4.4.5.4.35 QSPI_MODE_XIP

- **Name:** XIP Endian Mode Configure Register
- **Description:** This register is used to configure endian mode under XIP read access.
- **Base Address:** 0x4000E000
- **Offset:** 0x180
- **Reset Value:** 0x00000000

Table 4-241 XIP Endian Mode Configure Register

Bits	Field Name	RW	Reset	Description
31	M2_DYNAMIC_LE_EN	RW	0x0	Configure QSPI2 Dynamic Endian Mode. **Value:** • 0: Read and write 32-bit data without re-ordering in XIP mode • 1: Re-order the read and write data as [7:0], [15:8], [23:16], [31:24] when HSIZE is word. Re-order the read and write data as [23:16], [31:24], [7:0], [15:8] when HSIZE is half-word.
30	M1_DYNAMIC_LE_EN	RW	0x0	Configure QSPI1 Dynamic Endian Mode. **Value:**

Bits	Field Name	RW	Reset	Description
				• 0: Read and write 32-bit data without re-ordering in XIP mode • 1: Re-order the read and write data as [7:0], [15:8], [23:16], [31:24] when HSIZE is word. Re-order the read and write data as [23:16], [31:24], [7:0], [15:8] when HSIZE is half-word.
29	M0_DYNAMIC_LE_EN	RW	0x0	Configure QSPI0 Dynamic Endian Mode. **Value:** • 0: Read and write 32-bit data without re-ordering in XIP mode • 1: Re-order the read and write data as [7:0], [15:8], [23:16], [31:24] when HSIZE is word. Re-order the read and write data as [23:16], [31:24], [7:0], [15:8] when HSIZE is half-word.
28:12	RSVD	RW	0x0	Reserved bits
11:10	M2_STATIC_ENDIAN_MODE	RW	0x0	Configure QSPI2's static endian mode for XIP read. **Value:** • 0x0/0x3: Endian mode 0 • 0x1: Endian mode 1 • 0x2: Endian mode 2
9:8	M2_XIP_EN	RW	0x0	Control the QSPI2's XIP mode to turn to active or inactive. **Value:** • 0x0/0x2/0x3: active • 0x1: inactive
7:6	M1_STATIC_ENDIAN_MODE	RW	0x0	Configure QSPI1's static endian mode for XIP read. **Value:** • 0x0/0x3: Endian mode 0 • 0x1: Endian mode 1 • 0x2: Endian mode 2
5:4	M1_XIP_EN	RW	0x0	Control the QSPI1's XIP mode to turn to active or inactive. **Value:** • 0x0/0x2/0x3: active • 0x1: inactive
3:2	M0_STATIC_ENDIAN_MODE	RW	0x0	Configure QSPI0's static endian mode for XIP read. **Value:** • 0x0/0x3: Endian mode 0

Bits	Field Name	RW	Reset	Description
				• 0x1: Endian mode 1 • 0x2: Endian mode 2
1:0	M0_XIP_EN	RW	0x0	Control the QSPI0's XIP mode to be active or inactive. **Value:** • 0x0/0x2/0x3: active • 0x1: inactive

4.4.5.4.36 QSPI_CS_SETUP_DLY

- **Name:** CS Setup Delay Clocks Setting Register for QSPI
- **Description:** It works for both register and XIP access mode.
- **Base Address:** 0x4000E000
- **Offset:** 0x188
- **Reset Value:** 0x00000000

Table 4-242 CS Setup Delay Clocks Setting Register for QSPI

Bits	Field Name	RW	Reset	Description
31:24	RSVD	R	0x0	Reserved bits
23:16	M2_CS2SCK_DELAY	RW	0x0	Configure QSPI2's CS setup delay clocks. **Value:** • 0x0* SCLKDIV: Delay 0 clock. • 0x1* SCLKDIV: Delay 1 clock. … **Note:** The maximum cannot exceed 0xff.
15:8	M1_CS2SCK_DELAY	RW	0x0	Configure QSPI1's CS setup delay clocks. **Value:** • 0x0* SCLKDIV: Delay 0 clock. • 0x1* SCLKDIV: Delay 1 clock. … **Note:** The maximum cannot exceed 0xff.
7:0	M0_CS2SCK_DELAY	RW	0x0	Configure QSPI0's CS setup delay clocks. **Value:** • 0x0* SCLKDIV: Delay 0 clock.

Bits	Field Name	RW	Reset	Description
				• 0x1* SCLKDIV: Delay 1 clock. … **Note:** The maximum cannot exceed 0xff.

4.4.5.4.37 QSPI_CS_RELEASE_DLY

- **Name:** CS Release Delay Clocks Setting Register for QSPI
- **Description:** It works for both register and XIP access mode.
- **Base Address:** 0x4000E000
- **Offset:** 0x18C
- **Reset Value:** 0x00000000

Table 4-243 CS Release Delay Clocks Setting Register for QSPI

Bits	Field Name	RW	Reset	Description
31:24	RSVD	R	0x0	Reserved bits
23:16	M2_SCK2CS_DELAY	RW	0x0	Configure QSPI2's CS release delay clocks. **Value:** • 0x0* SCLKDIV: Delay 0 clock. • 0x1* SCLKDIV: Delay 1 clock. … **Note:** The maximum cannot exceed 0xff.
15:8	M1_SCK2CS_DELAY	RW	0x0	Configure QSPI1's CS release delay clocks. **Value:** • 0x0* SCLKDIV: Delay 0 clock. • 0x1* SCLKDIV: Delay 1 clock. … **Note:** The maximum cannot exceed 0xff.
7:0	M0_SCK2CS_DELAY	RW	0x0	Configure QSPI0's CS release delay clocks. **Value:** • 0x0* SCLKDIV: Delay 0 clock. • 0x1* SCLKDIV: Delay 1 clock.

Bits	Field Name	RW	Reset	Description
				... **Note:** The maximum cannot exceed 0xff.

4.4.5.5 Electrical Specifications

Table 4-244 QSPI electrical specifications

Parameter[1]	Description	Min.	Typ.	Max.	Unit
f_{sck}	Bit (baud) rate	47.96	48		MHz
t_{clk}	SCK clock period		20.83	20.85	ns
t_{rclk}	SCK clock rise time (10% -90%), 9pF Loading		1.45		ns
t_{fclk}	SCK clock fall time (90% - 10%), 9pF Loading		1.40		ns
t_{ch}	SCK clock high time		8.97		ns
t_{cl}	SCK clock low time		9.01		ns
t_{csp}	CS Setup delay time to active clock edge		10.4[2]		ns
			20.8[3]		ns
t_{sp}	Data Setup time to active clock edge			10.4	ns
				20.8	ns
t_{hd}	Data Hold time from active clock edge		8.97		ns

[1] All values are measured under 48 MHz baud rate in Clock Mode 3.
[2] Default t_{csp} in XIP Mode and can be adjusted by either software or hardware.
[3] Default t_{csp} in Register Mode and can be adjusted by either software or hardware.

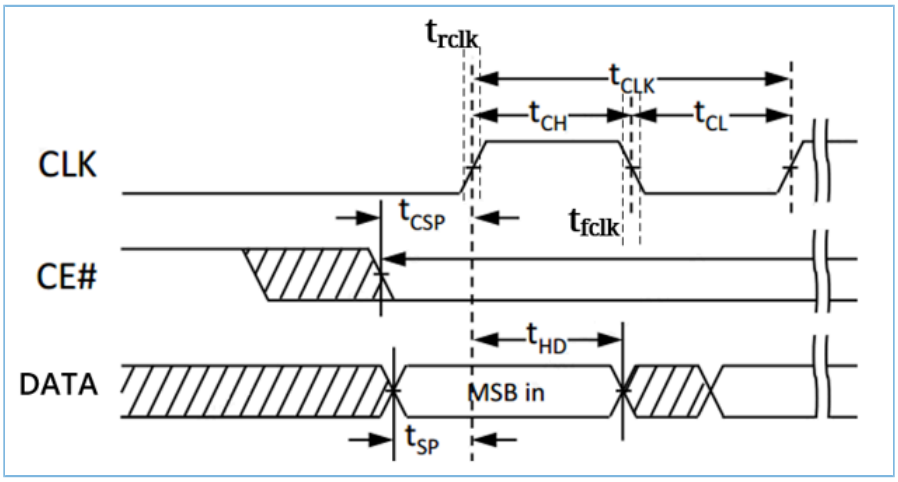

Figure 4-51 QSPI timing diagram

4.5 Audio Interface

Copyright © 2024 Shenzhen Goodix Technology Co., Ltd.

4.5.1 PDM

4.5.1.1 Introduction

The Pulse Density Modulation (PDM) module provides one instance and two channels for external audio front ends, for example, digital microphones. It also supports DMA for data transfer.

4.5.1.2 Main Features

- 1 instance
- 2 channels
- Left channel: Collect data on the rising edge.
- Right channel: Collect data on the failing edge.
- Up to two PDM microphones configured as a left/right pair using the same data input
- Supporting 8 kHz, 15.625 kHz, and 16 kHz sampling rates
- 16-bit samples
- Hardware decimation filters
- DMA support
- Programmable gain control

4.5.1.3 Block Diagram

The functional block diagram of PDM is shown in Figure 4-52. The PDM module comprises DMIC Chain Left, DMIC Chain Right, and PDM Register Block.

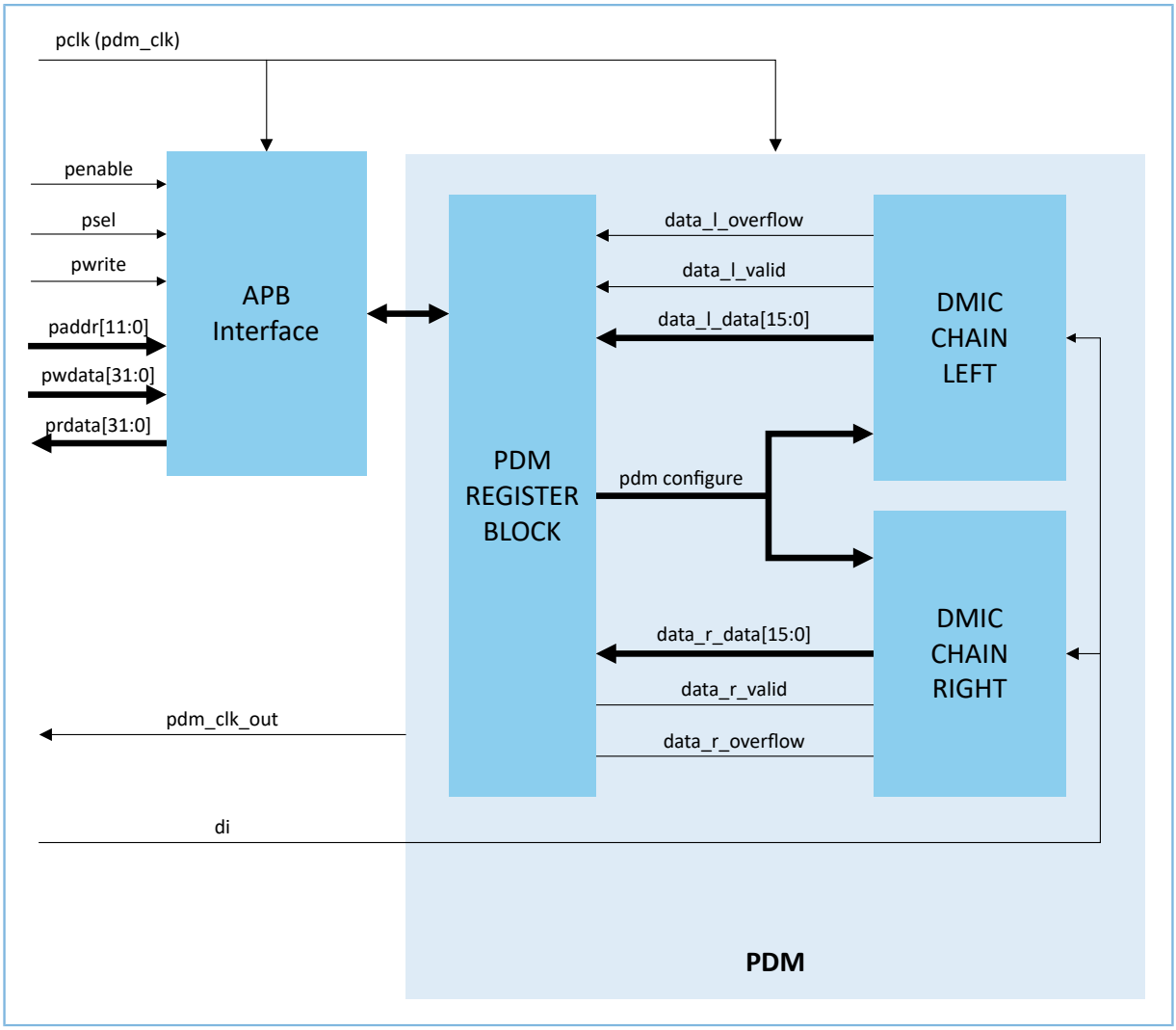

Figure 4-52 PDM block diagram

4.5.1.4 Protocols Introduction

PDM module is an interface for PDM signal input to PCM audio format output, so the input is 1-bit data in PDM format, and the output data is in PCM format.

4.5.1.4.1 Protocols Introduction

PDM is a modulation method that uses digital signals to represent analog signals. There is only 1-bit output, and non-zero is 1, so digital audio represented by PDM is also called oversampled 1-bit Audio. It is a high-frequency data stream of 1-bit digital sampling. In a PDM signal, the relative density of the pulse corresponds to the amplitude of the analog signal. As shown below, a large number of “1s” corresponds to high (positive) amplitude values, and a large number of “0s” corresponds to low (negative) amplitude values; alternating “1s” and “0s” corresponds to amplitude values.

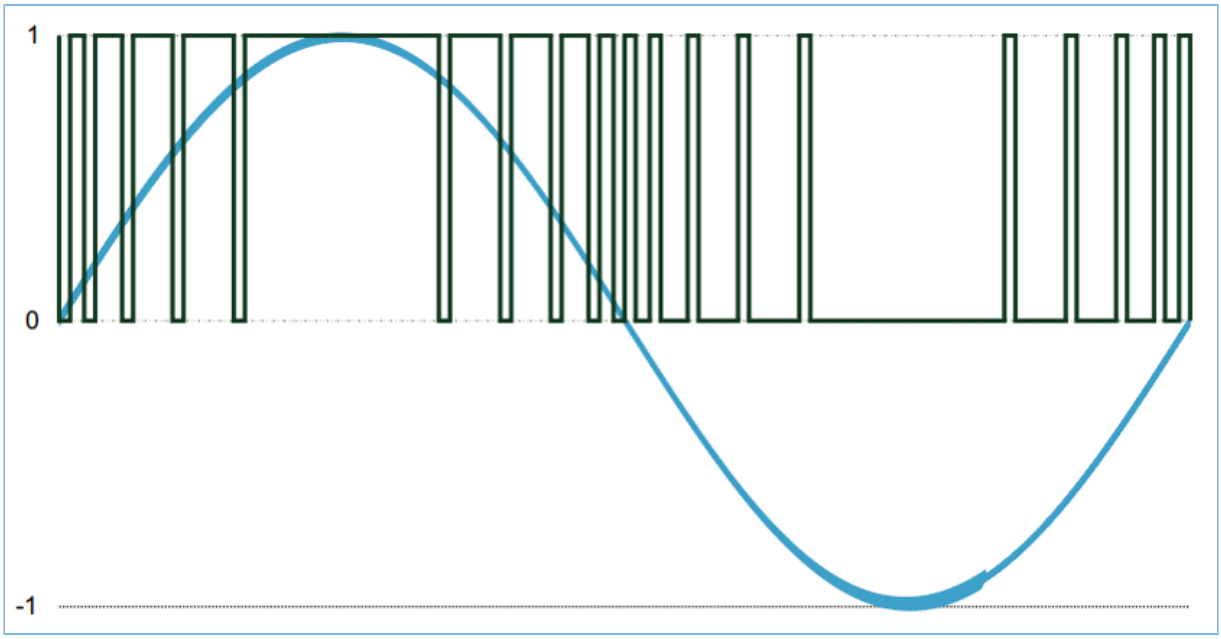

Figure 4-53 Pulse density modulation

4.5.1.4.2 PCM Introduction

In the PCM (Pulse Code Modulation) signal, the specific amplitude value is encoded as a pulse. The PCM stream has two basic properties which determine the fidelity of the stream relative to the original analog signal.

- Sampling rate
- Bit depth

Sampling rate is used to represent the signal digitally, the number of signal samples collected per second. The bit depth determines the number of bits of information in each sample.

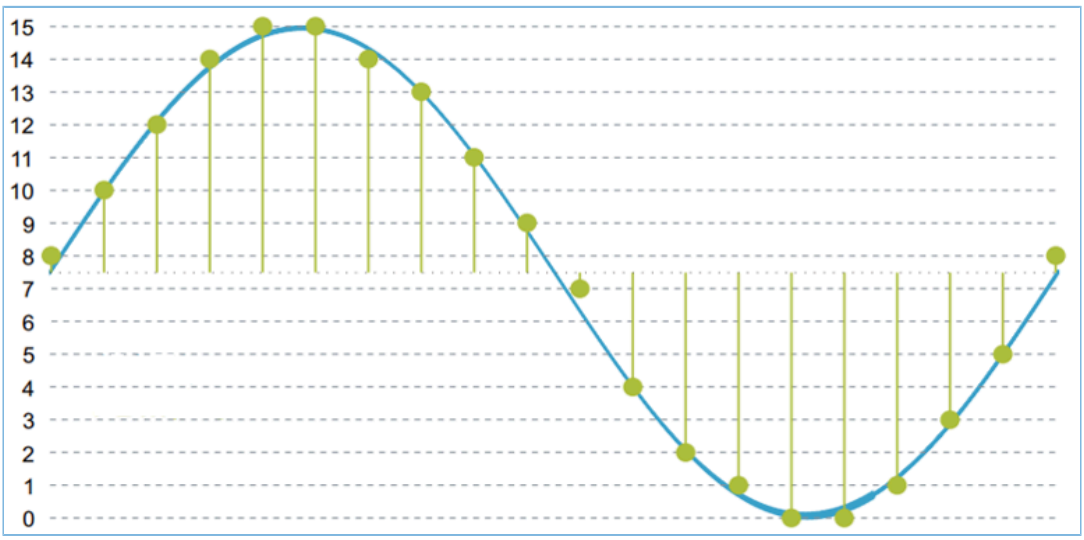

Figure 4-54 Pulse code modulation

4.5.1.5 Functional Description

The PDM module supports single-channel and dual-channel operating modes. It has two wires – serial data in (DI) and serial clock (CLOCK).

4.5.1.5.1 PDM Pin Mux

Before configuring the PDM module functions, make sure the pin mux of PDM interface GPIO is configured as PDM_CLKO and PDM_DI.

4.5.1.5.2 PDM Operation

By default, the left channel collects data on the rising edge of the clock, and the right channel collects data on the falling edge of the clock, resulting in two bit streams. In order to convert 1-bit data of PDM entrance into PCM audio samples, a decimation filter is included in the PDM module. The input of the filter is the two-channel PDM serial stream (with the left channel on clock high and the right channel on clock low). The output is 2 × 16-bit PCM samples at a sampling rate 64 times lower than the PDM clock rate.

4.5.1.5.3 PDM Gain

PDM module supports programmable gain control. The gain is controlled by the PGA_CFG_LEFT and PGA_CFG_RIGHT registers. PGA_CFG_LEFT controls the gain of the left channel, and PGA_CFG_RIGHT controls the gain of the right channel.

The control bit is a u4.10. This means it is a 14-bit unsigned number with 4 integer bits and 10 fractional bits. 0x400 gain is 0 dB, shift left by one bit approximately equal to +6dB; otherwise, shift right by one bit approximately equal to -6dB. For example, 0x800 = +6 dB and 0x200 = -6 dB. The conversion formula is as follows:

$$Gain = 20 * \log_{10}\left(\frac{X}{1024}\right)(dB)$$

Where, X is the register value.

Some calculated values are given below:

Table 4-245 Calculated PDM gain values (examples)

Value (HEX)	Gain (dB)	Value (HEX)	Gain (dB)
00000000000001(0x01)	-60.2	00000100011111(0x11F)	-11.0
00000000000010(0x02)	-54.2	00000101000010(0x142)	-10.0
00000000000011(0x03)	-50.7	00000101101010(0x16A)	-9.0
00000000000100(0x04)	-48.2	00000110010110(0x196)	-8.0
00000000000101(0x05)	-46.2	00000111000111(0x1C7)	-7.0
00000000000110(0x06)	-44.6	00000111111111(0x1FF)	-6.0
00000000000111(0x07)	-43.3	00001000111101(0x23D)	-5.0
00000000001000(0x08)	-42.1	00001010000011(0x283)	-4.0
00000000001001(0x09)	-41.1	00001011010001(0x2D1)	-3.0

Value (HEX)	Gain (dB)	Value (HEX)	Gain (dB)
00000000001010(0x0A)	-40.2	00001100101001(0x329)	-2.0
00000000001011(0x0B)	-39.4	00001110001100(0x38C)	-1.0
00000000001100(0x0C)	-38.6	00010000000000(0x400)	0.0
00000000001110(0x0E)	-37.3	00010001110111(0x477)	1.0
00000000010000(0x10)	-36.1	00010100000010(0x502)	2.0
00000000010010(0x12)	-35.1	00010110011111(0x59F)	3.0
00000000010100(0x14)	-34.2	00011001001110(0x64E)	4.0
00000000010111(0x17)	-33.0	00011100010011(0x713)	5.0
00000000011001(0x19)	-32.2	00011111110000(0x7F0)	6.0
00000000011101(0x1D)	-31.0	00100011101000(0x8E8)	7.0
00000000100000(0x20)	-30.1	00100111111110(0x9FE)	8.0
00000000100100(0x24)	-29.1	00101010010101(0xA95)	8.5
00000000101001(0x29)	-28.0	00101100110110(0xB36)	9.0
00000000101110(0x2E)	-27.0	00110010010100(0xC94)	10.0
00000000110011(0x33)	-26.1	00111000011101(0xE1D)	11.0
00000000111001(0x39)	-25.1	00111111010110(0xFD6)	12.0
00000001000000(0x40)	-24.1	01000111000100(0x11C4)	13.0
00000001001000(0x48)	-23.1	01001111101111(0x13EF)	14.0
00000001010001(0x51)	-22.0	01011001011110(0x165E)	15.0
00000001011011(0x5B)	-21.0	01100100011000(0x1918)	16.0
00000001100110(0x66)	-20.0	01110000101000(0x1C28)	17.0
00000001110011(0x73)	-19.0	01111110011000(0x1F98)	18.0
00000010000001(0x81)	-18.0	10001101110011(0x2373)	19.0
00000010010000(0x90)	-17.0	10011111000110(0x27C6)	20.0
00000010100010(0xA2)	-16.0	10110010100000(0x2CA0)	21.0
00000010110110(0xB6)	-15.0	11001000010010(0x3212)	22.0
00000011001100(0xCC)	-14.0	11100000101110(0x382E)	23.0
00000011100100(0Xe4)	-13.0	11111100001001(0x3F09)	24.0
00000100000000(0x100)	-12.0		

4.5.1.5.4 PDM Enable

The PDM starts data sampling after the EN_LEFT/EN_RIGHT register and CLK registers have been written. When starting the module, it will take some time for the filters to start outputting valid data. Transients from the PDM

microphone itself may also occur. The first few samples might hence contain invalid values or transients. It is therefore advised to discard the first few samples after a PDM start.

4.5.1.5.5 PDM Interrupt

The PDM module has two interrupts for the left and right channels respectively. One is the valid data interrupt, which will be generated when the PDM outputs data. The other is the data overflow interrupt, which will be generated when the DMA does not remove the data in time. The INT_LEFT/INT_RIGHT register can mask interrupts. The DATA_LEFT/DATA_RIGHT register can check interrupt flag.

4.5.1.5.6 PDM DMA

The PDM module supports one DMA channel for PDM transmit. It can carry data from three places: DATA_LEFT register, DATA_RIGHT register, and DATA register. The lower 16 bits of the DATA_LEFT register store the data of the left channel, and the lower 16 bits of the DATA_RIGHT register store the data of the right channel. The lower 16 bits of the DATA register store the left channel data, and the upper 16 bits store the right channel data. They are equivalent to the data of DATA_LEFT and DATA_RIGHT.

Note:

Since the PDM module does not have an internal FIFO, it is necessary to ensure the efficiency of the DMA as much as possible in the PDM DMA mode; otherwise the PDM is prone to overflow.

4.5.1.5.7 Hardware Connection Example

The PDM module is applicable to microphone voice acquisition. Connect the microphone clock to CLK, and data to DIN. The module supports three connection methods: left channel sample, right channel sample, and stereo sample, as shown below:

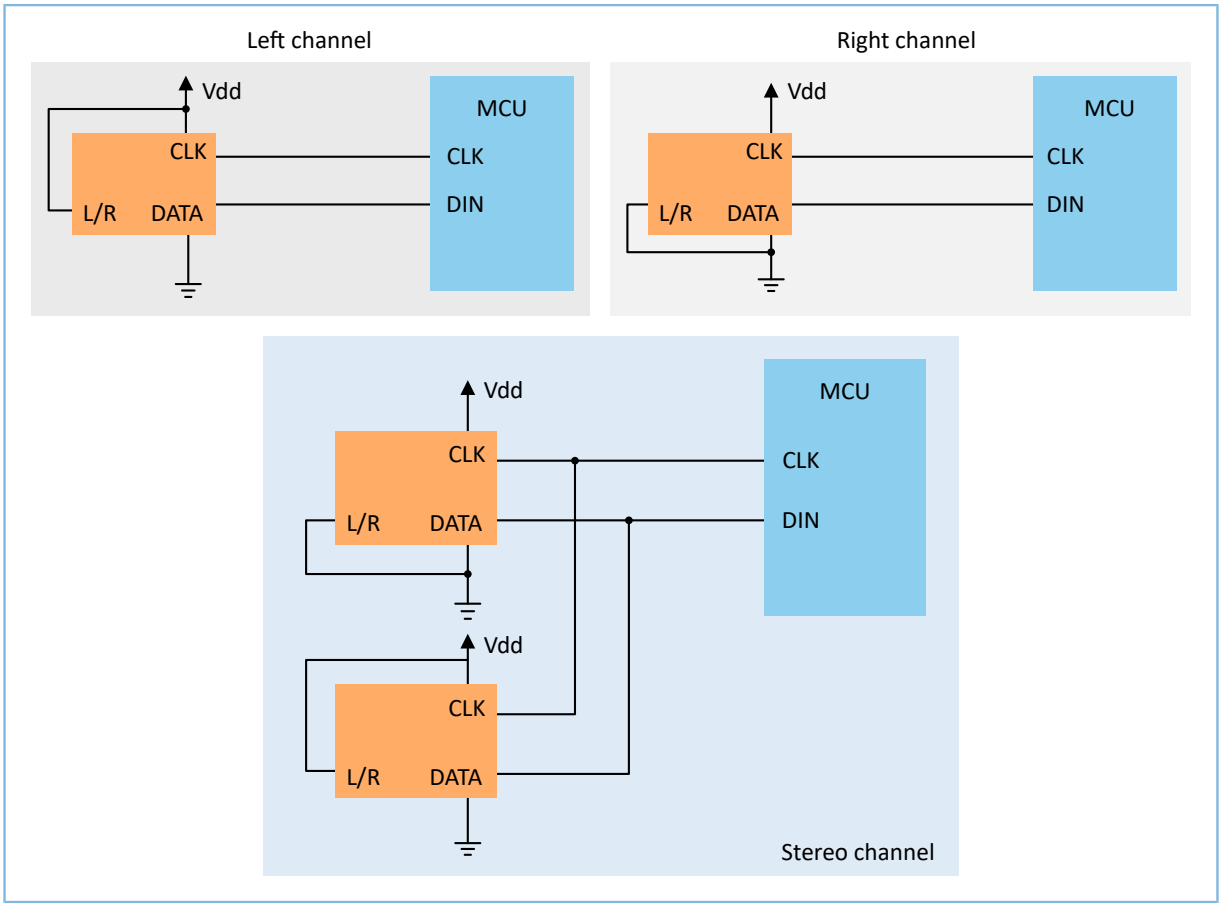

Figure 4-55 Hardware connection example

4.5.1.6 Registers

4.5.1.6.1 EN_LEFT

- **Name:** Left PDM Enable Register
- **Description:** This register specifies the sampling enable bit of the left channel.
- **Base Address:** 0x4000F700
- **Offset:** 0x0
- **Reset Value:** 0x00000000

Table 4-246 Left PDM Enable Register

Bits	Field Name	RW	Reset	Description
31:2	RSVD	R		Reserved bits
1	SAMPLE_DMIC	RW	0x0	Left channel sampling data • 0x0: Stop sampling data in. • 0x1: Start sampling data in.

Bits	Field Name	RW	Reset	Description
0	EN_RX	RW	0x0	Left channel reset • 0x0: Reset • 0x1: Not reset

4.5.1.6.2 PGA_CFG_LEFT

- **Name:** Left Channel Gain Register
- **Description:** This register specifies the gain of the left channel.
- **Base Address:** 0x4000F700
- **Offset:** 0x10
- **Reset Value:** 0x00000400

Table 4-247 Left Channel Gain Register

Bits	Field Name	RW	Reset	Description
31:14	RSVD	R		Reserved bits
0:13	VAL	RW	0x400	Left channel gain: It is a u4.10. This means it is a 14-bit unsigned number with 4 integer bits and 10 fractional bits. 0x400 gain is 0 dB, shift left by one bit (plus 3 dB); otherwise, shift right by one bit (minus 3 dB).

4.5.1.6.3 DATA_LEFT

- **Name:** Left Channel Data Register
- **Description:** This register specifies the data of the left channel.
- **Base Address:** 0x4000F700
- **Offset:** 0x14
- **Reset Value:** 0x00000000

Table 4-248 Left Channel Data Register

Bits	Field Name	RW	Reset	Description
31:19	RSVD	R	0x0	Reserved bits
18	MASK	RW	0x0	Write 1 to mask dma_req.
17	OVER	RW	0x0	Left channel data overflow flag. If DMA does not read this register or combined data register before another sample is written, this bit gets asserted. Moreover, the bit can be used as an interrupt to MCU. Write a 1 to clear.

Bits	Field Name	RW	Reset	Description
16	VALID	R	0x0	Left channel data valid flag, also can be used as an interrupt to MCU. Read to clear.
15:0	DATA	R	0x0	Left channel data

4.5.1.6.4 INT_LEFT

- **Name:** Left Channel Interrupt Register
- **Description:** This register specifies the interrupt of the left channel.
- **Base Address:** 0x4000F700
- **Offset:** 0x18
- **Reset Value:** 0x00000001

Table 4-249 Left Channel Interrupt Register

Bits	Field Name	RW	Reset	Description
31:2	RSVD	R	0x0	Reserved bits
1	OVER_MASK	RW	0x0	Left channel data overflow interrupt. Write 1 to mask the interrupt.
0	VALID_MASK	RW	0x1	Left channel data valid interrupt. Write 1 to mask the interrupt.

4.5.1.6.5 EN_RIGHT

- **Name:** Right PDM Enable Register
- **Description:** This register specifies the sampling enable bit of the right channel.
- **Base Address:** 0x4000F700
- **Offset:** 0x1C
- **Reset Value:** 0x00000000

Table 4-250 Right PDM Enable Register

Bits	Field Name	RW	Reset	Description
Bits	Field Name	RW	Reset	Description
31:2	RSVD	R		Reserved bits
1	SAMPLE_DMIC	RW	0x0	Right channel sampling data • 0x0: Stop sampling data in. • 0x1: Start sampling data in.
0	EN_RX	RW	0x0	Right channel reset

Bits	Field Name	RW	Reset	Description
				• 0x0: Reset • 0x1: Not reset

4.5.1.6.6 PGA_CFG_RIGHT

- **Name:** Right Channel Gain Register
- **Description:** This register specifies the gain of the right channel.
- **Base Address:** 0x4000F700
- **Offset:** 0x44
- **Reset Value:** 0x00000400

Table 4-251 Right Channel Gain Register

Bits	Field Name	RW	Reset	Description
31:14	RSVD	R		Reserved bits
0:13	VAL	RW	0x400	Right channel gain: It is a u4.10. This means it is a 14-bit unsigned number with 4 integer bits and 10 fractional bits. 0x400 gain is 0dB, shift left by one bit (plus 3 dB), otherwise, shift right by one bit (minus 3dB).

4.5.1.6.7 DATA_RIGHT

- **Name:** Right Channel Data Register
- **Description:** This register specifies the data of the right channel.
- **Base Address:** 0x4000F700
- **Offset:** 0x48
- **Reset Value:** 0x00000000

Table 4-252 Right Channel Data Register

Bits	Field Name	RW	Reset	Description
31:19	RSVD	R	0x0	Reserved bits
18	MASK	RW	0x0	Write 1 to mask dma_req.
17	OVER	RW	0x0	Right channel data overflow flag. If DMA does not read this register or combined data register before another sample is written, this bit gets asserted. Moreover, this bit can be used as an interrupt to MCU. Write a 1 to clear.

Bits	Field Name	RW	Reset	Description
16	VALID	R	0x0	Right channel data valid flag, also can be used as an interrupt to MCU. Read to clear.
15:0	DATA	R	0x0	Right channel data

4.5.1.6.8 INT_RIGHT

- **Name:** Right Channel Interrupt Register
- **Description:** This register specifies the interrupt of the right channel.
- **Base Address:** 0x4000F700
- **Offset:** 0x4C
- **Reset Value:** 0x00000001

Table 4-253 Right Channel Interrupt Register

Bits	Field Name	RW	Reset	Description
31:2	RSVD	R	0x0	Reserved bits
1	OVER_MASK	RW	0x0	Right channel data overflow interrupt. Write 1 to mask the interrupt.
0	VALID_MASK	RW	0x1	Right channel data valid interrupt. Write 1 to mask the interrupt.

4.5.1.6.9 DATA

- **Name:** Data Register
- **Description:** This specifies the Data Register.
- **Base Address:** 0x4000F700
- **Offset:** 0x50
- **Reset Value:** 0x00000000

Table 4-254 Data Register

Bits	Field Name	RW	Reset	Description
31:16	DATA_R	R	0x0	Right channel data, consistent with DATA_RIGHT register
15:0	DATA_L	R	0x0	Left channel data, consistent with DATA_LEFT register

4.5.1.6.10 CLK

- **Name:** CLK Enable Register
- **Description:** This specifies the CLK Enable Register.

- **Base Address:** 0x4000F700
- **Offset:** 0x58
- **Reset Value:** 0x00000000

Table 4-255 CLK Enable Register

Bits	Field Name	RW	Reset	Description
31:1	RSVD	R	0x0	Reserved bits
9:8	SAMPLE_RATE	RW	0x1	PDM sampling rate. • 0x0: 15.625K • 0x1: 16K • 0x3: 8K
0	EN	R	0x0	PDM CLK enable bit

4.5.1.7 Electrical Specifications

Table 4-256 Electrical specifications

Symbol	Description	Min.	Typ.	Max.	Unit
$f_{Pdm,clk,64}$	PDM clock speed		1		M
$t_{PDM,JITTER}$	Jitter in PDM clock output			20	ns
$t_{PDM,DATA}$	Decimation filter delay			6.125	ms
$t_{PDM,cv}$	Allowed clock edge to data valid		56		ns
$t_{PDM,ci}$	Allowed (other) clock edge to data invalid		49		ns
$t_{PDM,s}$	Data setup time at $f_{PDM,CLK}$ = 1 MHz	65			ns
$t_{PDM,h}$	Data hold time at $f_{PDM,CLK}$ = 1 MHz	0			ns
$G_{PDM,default}$	Default (reset) absolute gain of the PDM module		0		dB

The PDM timing diagram is as below:

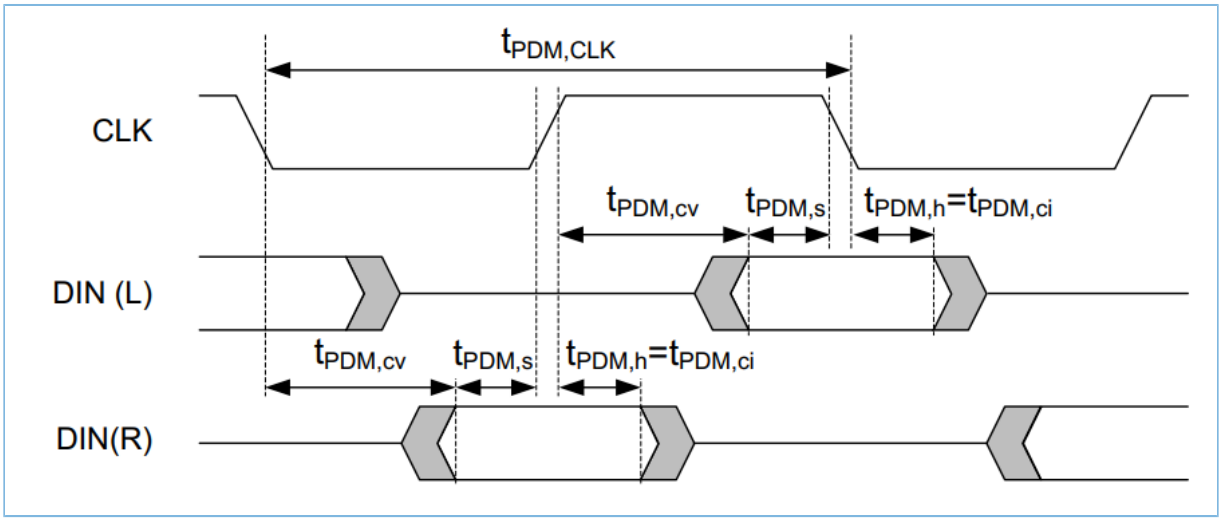

Figure 4-56 PDM timing diagram

4.5.2 I2S

4.5.2.1 Introduction

The I2S controller supports Inter-IC Sound Bus protocol to interface with external audio codec. Two 16-word deep FIFOs for read path and write path respectively are capable of handling 12-bit, 16-bit, 24-bit, and 32-bit word size. DMA controller handles the data movement between FIFO and memory.

The device has two I2S instances:

- I2S Master (I2S_M)
- I2S Slave (I2S_S)

I2S bus is used in systems that process digital audio signals, such as:

- A/D and D/A converters
- Digital signal processors
- Error correction for compact disc and digital recording
- Digital filters
- Digital I/O interfaces

I2S is a simple three-wire serial bus protocol developed by Philips to transfer stereo audio data. The bus only handles the transfer of audio data; hence, signals control and subcode need to be transferred separately using a different bus protocol (such as I2C).

4.5.2.2 Main Features

- 4-channel:
  - 1 RX channel and 1 TX channel for I2S Slave Controller

- 1 RX channel and 1 TX channel for I2S Master Controller

- Full duplex communication due to the independence of transmitter and receiver
- Asynchronous clocking of APB and I2S SCLK
- Audio data resolutions of 12 bits, 16 bits, 20 bits, 24 bits, and 32 bits
- SCLK gating for power saving
- Two 16-word FIFOs: one for transmission and the other for reception
- DMA support
- Programmable FIFO thresholds
- Various sampling rates: up to 96 kHz (16-bit WS)

4.5.2.3 Protocols Introduction

4.5.2.3.1 Signals and Formats

- The Inter-IC Sound (I2S) bus is a simple three-wire serial bus protocol developed by Philips to transfer stereo audio data. It is used extensively in audio interfaces. It adopts the design of data and clock signal separation, avoiding the distortion induced by time difference.
- The I2S bus has three signals:
  - SCLK: serial clock (also called BCLK. Each SCLK corresponds to each bit of audio data.)
  - WS: word select (also called LRCK. WS is used to switch between the left and right channel data.)
  - SD(SDATA): serial data (MSB → LSB: Data is transmitted from high-bit to low-bit)
- The I2S module of this device adopts the I2S Philips standard (neither MSB justified standard nor LSB justified standard). For instance, when WS is low, the word being transferred is left stereo data; when WS is high, the word being transferred is right stereo data. For standard I2S formats, the MSB of a word is sent one SCLK cycle after a WS change. Serial data sent by the transmitter can be synchronized with either the negative edge or positive edge of the SCLK signal. However, the receiver must latch the serial data on the rising edge of SCLK. The formats (as receiver) are shown in Figure 4-57.

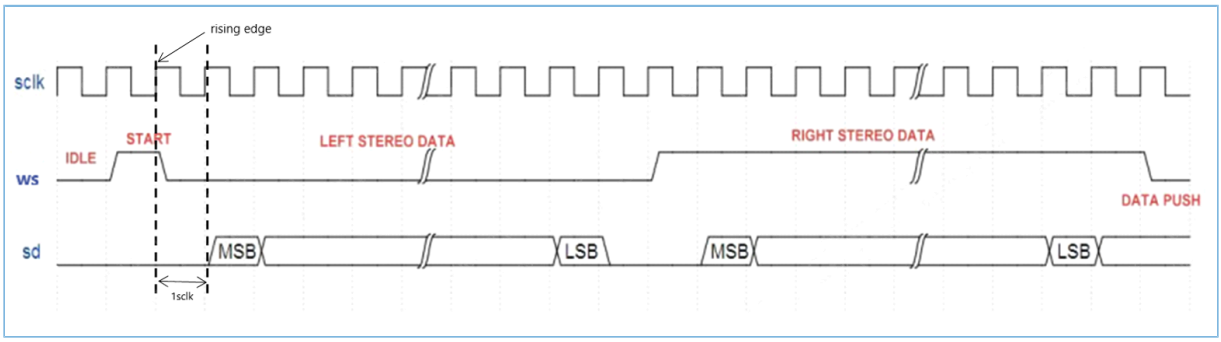

Figure 4-57 I2S formats (receiver)

4.5.2.3.2 Sampling and CLK frequency

As the frequency of SCLK is up to 3072 kHz, the number of SCLK cycles for which the (WS) line of I2S can be configured as 16 SCLK cycles, 24 SCLK cycles, or 32 SCLK cycles. If the 16 SCLK cycles are selected, the sampling rate is up to 96 kHz. The calculation formula is as follows:

- Sampling rate = Frequency of SCLK/[2 x Sampling depth (16,24 or 32)]
- Sampling rate = Frequency of WS

4.5.2.4 Functional Description

The I2S module f this device supports full duplex communication thanks to the independence of transmitter and receiver. The module has four wires: serial data in (SDI), serial data out (SDO), word select (WS), and serial clock (SCLK).

I2S controller block diagram is illustrated in Figure 4-58.

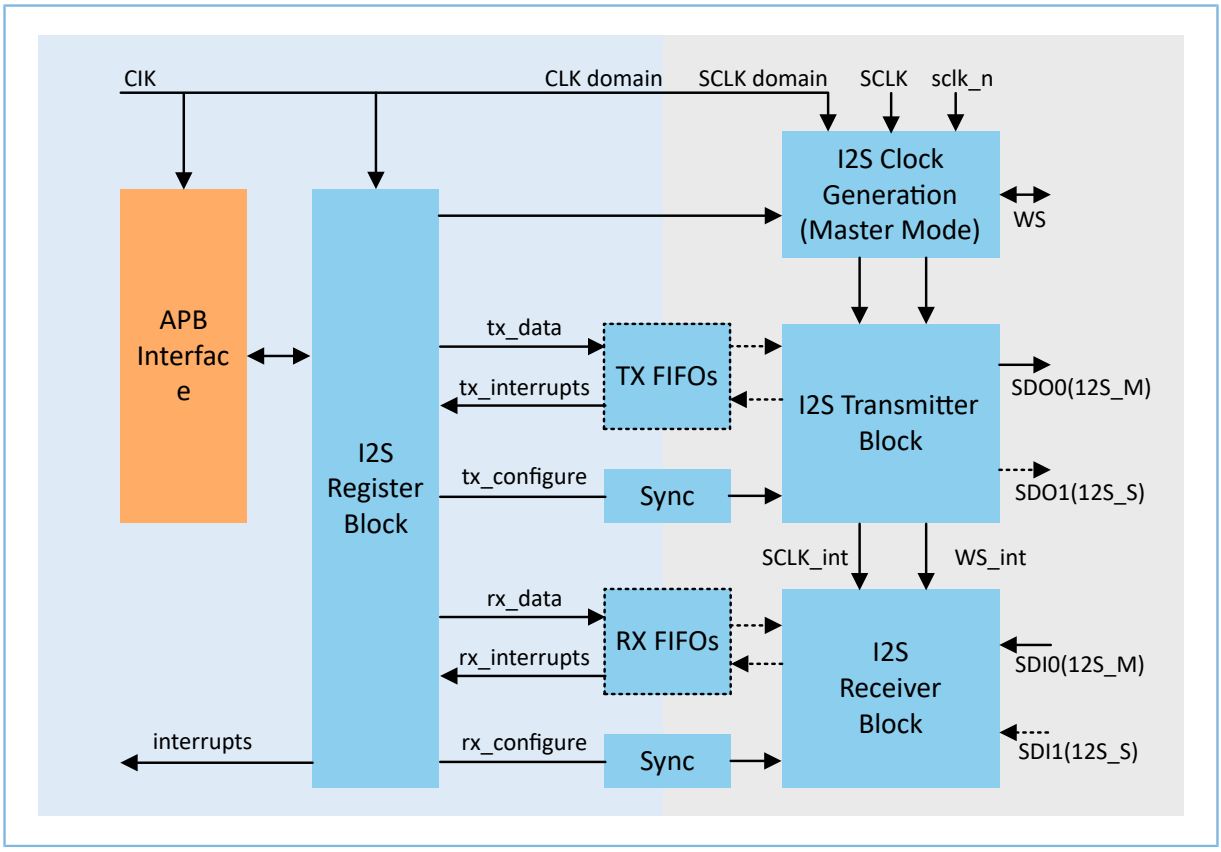

Figure 4-58 I2S block diagram

4.5.2.4.1 I2S Pin Mux

Before configuring the I2S module functions, the pin mux of four I2S interface GPIOs must be configured as I2S_M_SCLK (or I2S_S_SCLK), I2S_M_WS (or I2S_S_WS), I2S_M_SDI (or I2S_S_SDI), and I2S_M_SDO (or I2S_S_SDO).

4.5.2.4.2 Master and Slave

The I2S module can be used as a master or slave. The instance of master mode is I2S_M, and the instance of slave mode is I2S_S. When configured as a master, I2S controller initializes the WS signal and supplies the clock gating and clock enabling signals. Otherwise, when configured as a slave, I2S controller responds to externally generated SCLK and WS signals. When the I2S mode works in master mode, I2S clock generation must be enabled by setting the Clock Enable Register(CER) bit 0 to '1'.

4.5.2.4.3 Clock Divider

If the I2S module is used as master, the I2S Clock Configure Register (CLK_CFG) should be properly configured. The value of bit0–bit11 (div) is decided by the sampling rate, sampling depth (16, 24, or 32) and clock divider source (bit18). For example, if the CLK_CFG's bit18 is '1' (32 MHz) and the sampling rate is 48 kHz, the calculation formula is as follows:

$$Actual\ sampling\ rate = CLK_SRC / Actual\ divider / Bits$$

```
div_TEMP0 = 32MHz *10/ (2 X Sampling Depth X Sample Rate)
if ((div_TEMP0 % 10) >=5)
{
div_TEMP1 = div_TEMP0 /10 +1;
}
else
{div_TEMP1 = div_TEMP0 /10;
}
if (div_TEMP1>= 2)
{
div = div_TEMP1: 2;
}
else
{
div = 0;
}
```

Table 4-257 Sampling rate configuration

			CLK_SRC		CLK_SRC		CLK_SRC	
			32000K		64000K		96000K	
Sampling Rate (k)	**Bits**	**Desired Bit Clock (kHz)**	**Desired Divider (Value)**	**Actual Divider (Value)**	**Desired Divider (Value)**	**Actual Divider (Value)**	**Desired Divider (Value)**	**Actual Divider (Value)**
8	2*16 (left + right)	256	125	125	250	250	375	375
8	2*24 (left + right)	384	83.333333333	83	166.666666666	167	250	250
8	2*32 (left + right)	512	62.5	63	125	125	187.5	188
16	2*16 (left + right)	512	62.5	63	125	125	187.5	188

			CLK_SRC		CLK_SRC		CLK_SRC	
			32000K		64000K		96000K	
Sampling Rate (k)	Bits	Desired Bit Clock (kHz)	Desired Divider (Value)	Actual Divider (Value)	Desired Divider (Value)	Actual Divider (Value)	Desired Divider (Value)	Actual Divider (Value)
16	2*24 (left + right)	768	41.666666666	42	83.33333333	83	125	125
16	2*32 (left + right)	1024	31.25	31	62.5	63	93.75	94
48	2*16 (left + right)	1,536	20.83333333	21	41.666666667	42	62.5	63
48	2*24 (left + right)	2,304	13.88888889	14	27.777777778	28	41.666666667	42
48	2*32 (left + right)	3072	10.41666667	10	20.833333333	21	31.21951	31
44.1	2*16 (left + right)	1,411.2	22.67573696	23	45.351473923	45	68.02721088435	68
44.1	2*24 (left + right)	2,116.8	15.1171579743	15	30.234315969	30	45.3514739229	45
44.1	2*32 (left + right)	2,822.4	11.33786848	11	22.675736961	23	34.013605442	34

4.5.2.4.4 I2S Enable

You must enable the I2S controller before any data can be received or transmitted from or into the FIFOs. To enable the component, set the I2S enable (I2S_EN) bit of the I2S Enable Register (EN) to 1. When you disable the module, it acts as a global disable. To disable I2S controller, set EN[0] to 0.

After disable, the following events occur:

- TX and RX FIFOs are cleared, and read/write pointers are reset.
- Any data being transmitted or received is lost.
- All other programmable enables (such as transmitter/receiver block enables and individual TX/RX channel enables) in the component are overridden.

- Generation of master mode clock signals sclk_en, ws_out, and sclk_gate are disabled (for instance, they are held low).

When I2S controller is enabled and configured as a master, the device always starts in the left stereo data cycle (WS = 0), and one SCLK cycle after transitions to the right stereo data cycle (WS = 1). This allows for half a frame of SCLKS to write data to the TX FIFOs and to ensure that any connected slave receivers do not miss the start of the data frame (for instance, the WS 1: to-0 transition) once the SCLK restarts. (When I2S controller is configured as a slave, WS is externally supplied.) On reset, the EN[0] is set to 0 (disable).

4.5.2.4.5 I2S as Transmitter

I2S Master controller and I2S Slave controller support 1 I2S transmit (TX) channel. Stereo data pairs (such as, left and right audio data) written to a TX channel by CPU are shifted out serially on the appropriate serial data out line. The shifting is timed with respect to SCLK and WS.

Figure 4-59 illustrates the basic usage flow for an I2S controller when it acts as a transmitter.

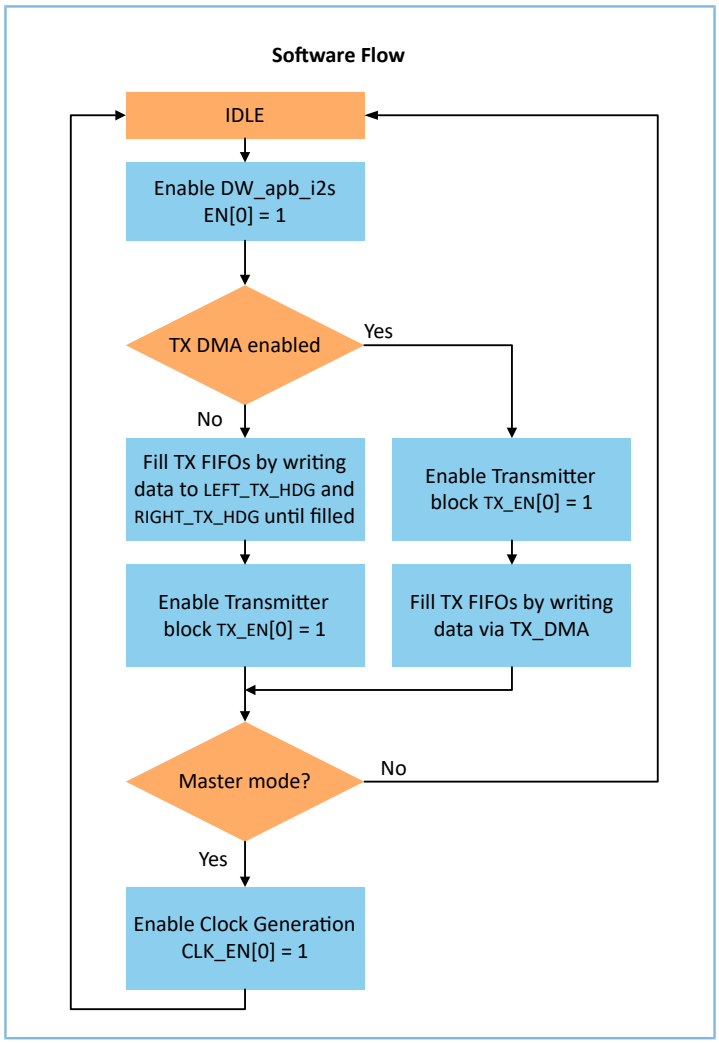

Figure 4-59 Basic usage flow when I2S as transmitter

4.5.2.4.6 I2S as Receiver

I2S Master controller and I2S Slave controller support 1 I2S receive (RX) channel. Stereo data pairs (such as, left and right audio data) are received serially from a data input line. These data words are stored in RX FIFOs until they are read by CPU. The receiving is timed with respect to SCLK and WS.

Figure 4-60 illustrates the basic usage flow for an I2S controller when it acts as a receiver.

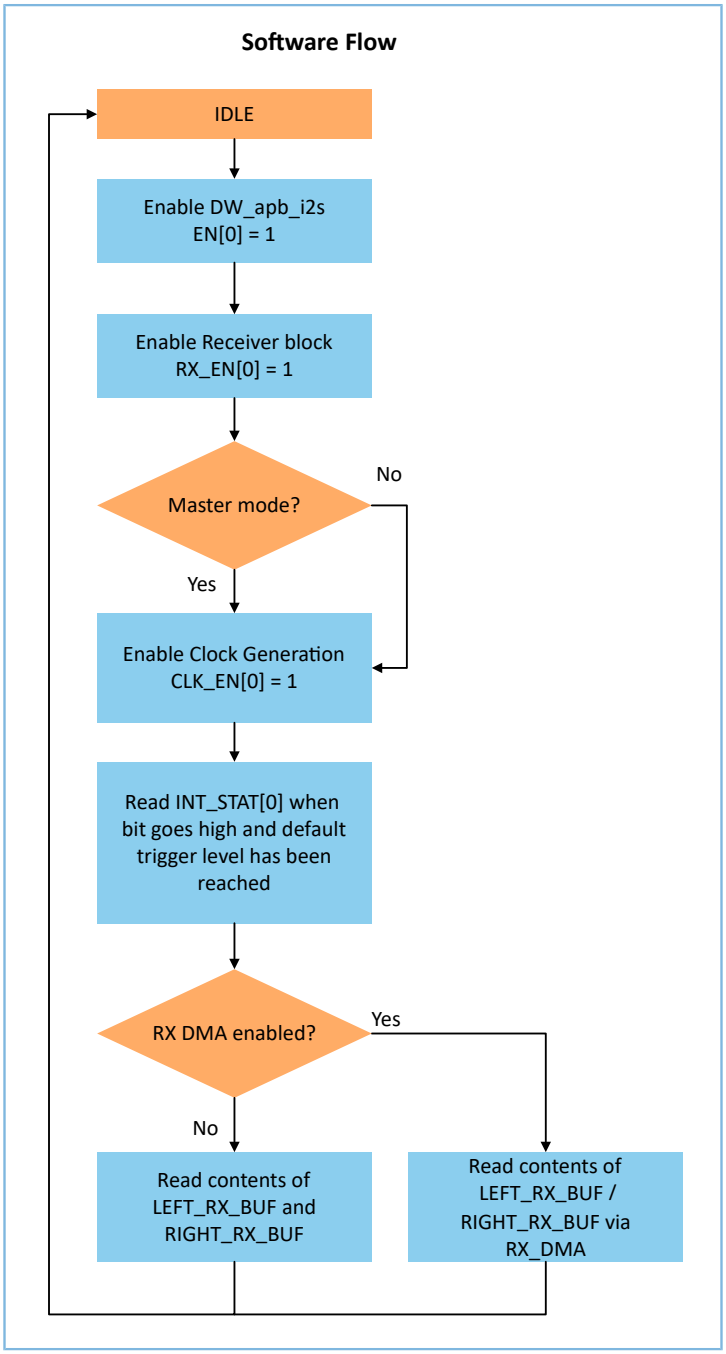

Figure 4-60 Basic usage flow when I2S as receiver

4.5.2.4.7 I2S FIFO

The I2S module provides two 16-word FIFOs, one for transmission and the other for reception. By configuring the Receive FIFO Configuration Register (RX_FIFO_CFG), you can program the trigger level in the RX FIFO at which the

Received Data Available Interrupt is generated. By configuring Transmit FIFO Configuration Register (TX_FIFO_CFG), you can program the trigger level in the TX FIFO at which the Empty Threshold Reached Interrupt is generated. In addition, the Interrupt Mask Register (INT_MASK) is supported to mask TX/RX FIFO overrun and empty interrupts.

4.5.2.4.8 I2S DMA

The I2S module supports four DMA channels for I2S slave transmit, I2S slave receive, I2S master transmit, and I2S master receive respectively. The Receiver Block DMA Register (RX_DMA), Reset Receiver Block DMA Register (RST_RX_DMA), Transmitter Block DMA Register (TX_DMA), and Reset Transmitter Block DMA Register (RST_TX_DMA) are provided to configure these channels.

4.5.2.5 Registers

4.5.2.5.1 EN

- **Name:** I2S Enable Register
- **Description:** This register acts as a global enable/disable for I2S.
- **Base Address:** 0x4000F200 (master) and 0x4000F000 (slave)
- **Offset:** 0x0
- **Reset Value:** 0x00000000

Table 4-258 I2S Enable Register

Bits	Field Name	RW	Reset	Description
31:1	RSVD	R		Reserved bits
0	I2S_EN	RW	0x0	I2S enable bit. This bit enables or disables I2S. A disable on this bit overrides any other block or channel as well as enables and flushes all FIFOs. **Value:** • 0x0: I2S disabled • 0x1: I2S enabled

4.5.2.5.2 RX_EN

- **Name:** I2S Receiver Block Enable Register
- **Description:** This register acts as an enable/disable for the I2S receiver block.
- **Base Address:** 0x4000F200 (master) and 0x4000F000 (slave)
- **Offset:** 0x4
- **Reset Value:** 0x00000000

Table 4-259 I2S Receiver Block Enable Register

Bits	Field Name	RW	Reset	Description
31:1	RSVD	R		Reserved bits
0	RX_EN	RW	0x0	Receiver block enable. This bit enables or disables the receiver. A disable on this bit overrides any individual receive channel. **Value:** • 0x0: Receiver disabled • 0x1: Receiver enabled

4.5.2.5.3 TX_EN

- **Name:** I2S Transmitter Block Enable Register
- **Description:** This register acts as an enable/disable for the I2S transmitter block.
- **Base Address:** 0x4000F200 (master) and 0x4000F000 (slave)
- **Offset:** 0x8
- **Reset Value:** 0x00000000

Table 4-260 I2S Transmitter Block Enable Register

Bits	Field Name	RW	Reset	Description
31:1	RSVD	R		Reserved bits
0	TX_EN	RW	0x0	Transmitter block enable. This bit enables or disables the transmitter. A disable on this bit overrides any individual transmit channel. **Value:** • 0x0: Transmitter disabled • 0x1: Transmitter enabled

4.5.2.5.4 CLK_EN

- **Name:** Clock Enable Register
- **Description:** This register acts as an enable/disable for the I2S clock generation block.
- **Base Address:** 0x4000F200 (master) and 0x4000F000 (slave)
- **Offset:** 0xC
- **Reset Value:** 0x00000000

Table 4-261 Clock Enable Register

Bits	Field Name	RW	Reset	Description
31:1	RSVD	R		Reserved bits
0	CLK_EN	RW	0x0	Clock generation enable/disable. This bit enables or disables the clock generation signals when I2S is a master. **Value:** • 0x0: Clock generation disabled • 0x1: Clock generation enabled **Note:** When the I2S is configured as a slave, this register serves no purpose.

4.5.2.5.5 SCLK_CFG

- **Name:** Clock Configuration Register
- **Description:** This register configures the WS when I2S is a master.
- **Base Address:** 0x4000F200 (master) and 0x4000F000 (slave)
- **Offset:** 0x10
- **Reset Value:** 0x00000000

Table 4-262 Clock Configuration Register

Bits	Field Name	RW	Reset	Description
31:5	RSVD	R		Reserved bits
4:3	WS_SCLK	RW	0x0	These bits are used to program the number of SCLK cycles for which the word select line (ws_out) stays in the left or right sample mode. The I2S clock generation block must be disabled (CLK_EN [0] = 0) prior to any changes in this value. **Value:** • 0x0: 16 SCLK cycles • 0x1: 24 SCLK cycles • 0x2: 32 SCLK cycles
2:0	SCLK_GAT	RW	0x0	These bits are used to program the gating of SCLK. The programmed gating value must be less than or equal to the largest configured/programmed audio resolution to prevent the truncating of RX/TX data. The I2S clock generation

Bits	Field Name	RW	Reset	Description
				block must be disabled (CLK_EN[0] = 0) before making any changes in this value. **Values:** • 0x0 (NO_CLOCK_GATING): Clock gating is disabled • 0x1 (CLOCK_CYCLES_12): Gating after 12 SCLK cycles • 0x2 (CLOCK_CYCLES_16): Gating after 16 SCLK cycles • 0x3 (CLOCK_CYCLES_20): Gating after 20 SCLK cycles • 0x4 (CLOCK_CYCLES_24): Gating after 24 SCLK cycles Exists: This register is only relevant when component is configured to be a master (I2S_MODE_EN = 1).

4.5.2.5.6 RX_FIFO_RST

- **Name:** Receiver Block FIFO Reset Register
- **Description:** This register specifies the Receiver Block FIFO Reset Register.
- **Base Address:** 0x4000F200 (master) and 0x4000F000 (slave)
- **Offset:** 0x14
- **Reset Value:** 0x00000000

Table 4-263 Receiver Block FIFO Reset Register

Bits	Field Name	RW	Reset	Description
31:1	RSVD	W		Reserved bits
0	RX_FIFO_RST	W	0x0	Receiver FIFO reset bit. Writing a 1 to this register flushes all the RX FIFOs (this is a self clearing bit). The receiver block must be disabled before writing to this bit. **Value:** • 0x0: Does not flush the RX FIFO. • 0x1: Flushes the RX FIFO.

4.5.2.5.7 TX_FIFO_RST

- **Name:** Transmitter Block FIFO Reset Register
- **Description:** This register specifies the Transmitter Block FIFO Reset Register.
- **Base Address:** 0x4000F200 (master) and 0x4000F000 (slave)

- **Offset:** 0x18
- **Reset Value:** 0x00000000

Table 4-264 Transmitter Block FIFO Reset Register

Bits	Field Name	RW	Reset	Description
31:1	RSVD	W		Reserved bits
0	TX_FIFO_RST	W	0x0	Transmitter FIFO Reset. Writing a 1 to this register flushes all the TX FIFOs (this is a self clearing bit). The transmitter block must be disabled before writing to this bit. **Value:** • 0x0: Does not flush the TX FIFO. • 0x1: Flushes the TX FIFO.

4.5.2.5.8 LEFT_RX_BUF

- **Name:** Left Receive Buffer Register
- **Description:** This specifies the Left Receive Buffer Register.
- **Base Address:** 0x4000F200 (master) and 0x4000F000 (slave)
- **Offset:** 0x20
- **Reset Value:** 0x00000000

Table 4-265 Left Receive Buffer Register

Bits	Field Name	RW	Reset	Description
31:0	LEFT_RX_BUF	R	0x0	The left stereo data received serially from the receive channel input (SDI) is read through this register. If the RX FIFO is full, and the two-stage read operation (for instance, a read from LEFT_RX_BUF followed by a read from RIGHT_RX_BUF) is not performed before the start of the next stereo pair, the new data is lost and an overrun interrupt occurs (data already in the RX FIFO is preserved). **Note:** Before reading this register again, the right stereo data must be read from RIGHT_RX_BUF; otherwise the status/interrupts will not be valid.

4.5.2.5.9 LEFT_TX_HDG

- **Name:** Left Transmit Holding Register
- **Description:** This specifies the Left Transmit Holding Register.
- **Base Address:** 0x4000F200 (master) and 0x4000F000 (slave)
- **Offset:** 0x20
- **Reset Value:** 0x00000000

Table 4-266 Left Transmit Holding Register

Bits	Field Name	RW	Reset	Description
31:0	LEFT_TX_HDG	W	0x0	The left stereo data to be transmitted serially through the transmit channel output (SDO) is written through this register. Writing is a two-stage process: • A write to this register passes the left stereo sample to the transmitter. • A right stereo sample MUST be written to the RIGHT_TX_HDG register. Data must only be written to the FIFO when it is not full. Any attempt to write to a full FIFO results in data being lost and an overrun interrupt being generated.

4.5.2.5.10 RIGHT_RX_BUF

- **Name:** Right Receive Buffer Register
- **Description:** This specifies the Right Receive Buffer Register.
- **Base Address:** 0x4000F200 (master) and 0x4000F000 (slave)
- **Offset:** 0x24
- **Reset Value:** 0x00000000

Table 4-267 Right Receive Buffer Register

Bits	Field Name	RW	Reset	Description
31:0	RIGHT_RX_BUF	R	0x0	The right stereo data received serially from the receive channel input (SDI) is read through this register. If the RX FIFO is full, and the two-stage read operation (for instance, read from LEFT_RX_BUF followed by a read from RIGHT_RX_BUF) is not performed before the start of the next stereo pair, the new data is lost and an overrun interrupt occurs (data already in the RX FIFO is preserved).

Bits	Field Name	RW	Reset	Description
				Note: Before reading this register, the left stereo data MUST be read from LEFT_RX_BUF; otherwise the status/interrupts will not be valid.

4.5.2.5.11 RIGHT_TX_HDG

- **Name:** Right Transmit Holding Register
- **Description:** This specifies the Right Transmit Holding Register.
- **Base Address:** 0x4000F200 (master) and 0x4000F000 (slave)
- **Offset:** 0x24
- **Reset Value:** 0x00000000

Table 4-268 Right Transmit Holding Register

Bits	Field Name	RW	Reset	Description
31:0	RIGHT_TX_HDG	W	0x0	The right stereo data to be transmitted serially through the transmit channel output (SDO) is written through this register. Writing is a two-stage process: • A write to this register passes the right stereo sample to the transmitter. • A left stereo sample MUST be written to the LEFT_TX_HDG register. Data must only be written to the FIFO when it is not full. Any attempt to write to a full FIFO results in data being lost and an overrun interrupt being generated.

4.5.2.5.12 RX_CH_EN

- **Name:** Receive Enable Register
- **Description:** This specifies the Receive Enable Register.
- **Base Address:** 0x4000F200 (master) and 0x4000F000 (slave)
- **Offset:** 0x28
- **Reset Value:** 0x00000001

Table 4-269 Receive Enable Register

Bits	Field Name	RW	Reset	Description
31:1	RSVD	R		Reserved bits
0	RX_CH_EN	RW	0x1	Receive channel enable bit. This bit enables/disables a receive channel. On enable, the channel begins receiving on the next left stereo cycle. A global disable of I2S (EN [0] = 0) or the receiver block (RX_EN [0] = 0) overrides this value. **Value:** • 0x0: Receive channel disabled • 0x1: Receive channel enabled

4.5.2.5.13 TX_CH_EN

- **Name:** Transmit Enable Register
- **Description:** This specifies the Transmit Enable Register.
- **Base Address:** 0x4000F200 (master) and 0x4000F000 (slave)
- **Offset:** 0x2c
- **Reset Value:** 0x00000001

Table 4-270 Transmit Enable Register

Bits	Field Name	RW	Reset	Description
31:1	RSVD	R		Reserved bits
0	TX_CH_EN	RW	0x1	Transmit channel enable bit. This bit enables/disables a transmit channel. On enable, the channel begins transmitting on the next left stereo cycle. A global disable of I2S (EN [0] = 0) or transmitter block (TX_EN [0] = 0) overrides this value. **Value:** • 0x0: Transmit channel disabled • 0x1: Transmit channel enabled

4.5.2.5.14 RX_CFG

- **Name:** Receive Configuration Register

- **Description:** This specifies the Receive Configuration Register.
- **Base Address:** 0x4000F200 (master) and 0x4000F000 (slave)
- **Offset:** 0x30
- **Reset Value:** 0x00000005

Table 4-271 Receive Configuration Register

Bits	Field Name	RW	Reset	Description
31:3	RSVD	R		Reserved bits
2:0	WORD_LEN	RW	0x5	These bits are used to program the desired data resolution of the receiver and enable the LSB of the incoming left (or right) word to be placed in the LSB of the LEFT_RX_BUF (or RIGHT_RX_BUF) register. Programmed data resolutions must be less than or equal to 0x5. If the selected resolution is greater than 0x5, the receive channel defaults go back to 0x5. The channel must be disabled prior to any changes in this value (RX_EN0[0] = 0). **Value:** • 0x0: Ignore the word length. • 0x1: 12-bit data resolution of the receiver • 0x2: 16-bit data resolution of the receiver • 0x3: 20-bit data resolution of the receiver • 0x4: 24-bit data resolution of the receiver • 0x5: 32-bit data resolution of the receiver

4.5.2.5.15 TX_CFG

- **Name:** Transmit Configuration Register
- **Description:** This specifies the Transmit Configuration Register.
- **Base Address:** 0x4000F200 (master) and 0x4000F000 (slave)
- **Offset:** 0x34
- **Reset Value:** 0x00000005

Table 4-272 Transmit Configuration Register

Bits	Field Name	RW	Reset	Description
31:3	RSVD	R		Reserved bits

Bits	Field Name	RW	Reset	Description
2:0	WORD_LEN	RW	0x5	These bits are used to program the data resolution of the transmitter and ensure the MSB of the data is transmitted first. Programmed resolutions must be less than or equal to 0x5. If the selected resolution is greater than 0x5, the transmit channel defaults go back to 0x5. The channel must be disabled prior to any changes in this value (TX_EN [0] = 0). **Value:** • 0x0: Ignore the word length. • 0x1: 12-bit data resolution of the transmitter • 0x2: 16-bit data resolution of the transmitter • 0x3: 20-bit data resolution of the transmitter • 0x4: 24-bit data resolution of the transmitter • 0x5: 32-bit data resolution of the transmitter

4.5.2.5.16 INT_STAT

- **Name:** Interrupt status Register
- **Description:** This specifies the Interrupt Status Register.
- **Base Address:** 0x4000F200 (master) and 0x4000F000 (slave)
- **Offset:** 0x38
- **Reset Value:** 0x00000010

Table 4-273 Interrupt status Register

Bits	Field Name	RW	Reset	Description
31:6	RSVD	R		Reserved bits
5	TX_FIFO_OVER	R	0x0	Status of data overrun interrupt for the TX channel. This bit specifies whether the TX FIFO write is valid or an overrun. Attempt to write to full TX FIFO. **Value:** • 0x0: TX FIFO write valid • 0x1: TX FIFO write overrun
4	TX_FIFO_EMPTY	R	0x1	Status of transmit empty trigger interrupt.

Bits	Field Name	RW	Reset	Description
				This bit specifies whether the TX FIFO trigger level has been reached or not. TX FIFO is empty. **Value:** • 0x0: TX FIFO trigger level is reached. • 0x1: TX FIFO trigger level is not reached.
3:2	RSVD	R		Reserved bits
1	RX_FIFO_OVER	R	0x0	Status of data overrun interrupt for the RX channel. Incoming data will be lost due to a full RX FIFO. **Value:** • 0x0: RX FIFO write valid • 0x1: RX FIFO write overrun
0	RX_DATA_AVL	R	0x0	Status of receive data available interrupt. This bit denotes the status of the RX FIFO trigger level. **Value:** • 0x1: RX FIFO trigger level is reached. • 0x0: RX FIFO trigger level is not reached.

4.5.2.5.17 INT_MASK

- **Name:** Interrupt Mask Register
- **Description:** This specifies the Interrupt Mask Register.
- **Base Address:** 0x4000F200 (master) and 0x4000F000 (slave)
- **Offset:** 0x3c
- **Reset Value:** 0x00000033

Table 4-274 Interrupt Mask Register

Bits	Field Name	RW	Reset	Description
31:6	RSVD	R		Reserved bits
5	TX_FOM	RW	0x1	Masks TX FIFO overrun interrupts. This bit masks or unmasks a TX FIFO overrun interrupt. **Value:** • 0x1: Disable TX FIFO overrun interrupt. • 0x0: Enable TX FIFO overrun interrupt.
4	TX_FEM	RW	0x1	Masks TX FIFO empty interrupt. This bit masks or unmasks a TX FIFO empty interrupt.

Bits	Field Name	RW	Reset	Description
				Value: • 0x1: Disable TX FIFO Empty interrupt. • 0x0: Enable TX FIFO Empty interrupt.
3:2	RSVD	R		Reserved bits
1	RX_FOM	RW	0x1	Masks RX FIFO overrun interrupt. This bit masks or unmasks an RX FIFO overrun interrupt. **Value:** • 0x1: Disable RX FIFO overrun interrupt. • 0x0: Enable RX FIFO overrun interrupt.
0	RX_DAM	RW	0x1	Masks RX FIFO data available interrupt. This bit masks or unmasks an RX FIFO data available interrupt. **Value:** • 0x1: Disable RX FIFO data available interrupt. • 0x0: Enable RX FIFO data available interrupt.

4.5.2.5.18 RX_OVER

- **Name:** Receive Overrun Register
- **Description:** This specifies the Receive Overrun Register.
- **Base Address:** 0x4000F200 (master) and 0x4000F000 (slave)
- **Offset:** 0x40
- **Reset Value:** 0x00000000

Table 4-275 Receive Overrun Register

Bits	Field Name	RW	Reset	Description
31:1	RSVD	R		Reserved bits
0	RX_CLR_FDO	R	0x0	Read this bit to clear the RX FIFO data overrun interrupt. **Value:** • 0x0: RX FIFO write valid • 0x1: RX FIFO write overrun

4.5.2.5.19 TX_OVER

- **Name:** Transmit Overrun Register
- **Description:** This specifies the Transmit Overrun Register.

- **Base Address:** 0x4000F200 (master) and 0x4000F000 (slave)
- **Offset:** 0x44
- **Reset Value:** 0x00000000

Table 4-276 Transmit Overrun Register

Bits	Field Name	RW	Reset	Description
31:1	RSVD	R		Reserved bits
0	TX_CLR_FDO	R	0X0	Read this bit to clear the TX FIFO data overrun interrupt. **Value:** • 0x0: TX FIFO write valid • 0x1: TX FIFO write overrun

4.5.2.5.20 RX_FIFO_CFG

- **Name:** Receive FIFO Configuration Register
- **Description:** This specifies the Receive FIFO Configuration Register.
- **Base Address:** 0x4000F200 (master) and 0x4000F000 (slave)
- **Offset:** 0x48
- **Reset Value:** 0x00000008

Table 4-277 Receive FIFO Configuration Register

Bits	Field Name	RW	Reset	Description
31:4	RSVD	R		Reserved bits
3:0	RX_FIFO_TL	RW	0x8	These bits program the trigger level in the RX FIFO at which the received data available interrupt is generated. Trigger Level = Programmed Value + 1 **Value:** • 0x0: Interrupt trigger when FIFO level is 1. • 0x1: Interrupt trigger when FIFO level is 2. • 0x2: Interrupt trigger when FIFO level is 3. • 0x3: Interrupt trigger when FIFO level is 4. • 0x4: Interrupt trigger when FIFO level is 5. • 0x5: Interrupt trigger when FIFO level is 6. • 0x6: Interrupt trigger when FIFO level is 7. • 0x7: Interrupt trigger when FIFO level is 8. • 0x8: Interrupt trigger when FIFO level is 9.

Bits	Field Name	RW	Reset	Description
				• 0x9: Interrupt trigger when FIFO level is 10. • 0xA: Interrupt trigger when FIFO level is 11. • 0xB: Interrupt trigger when FIFO level is 12. • 0xC: Interrupt trigger when FIFO level is 13. • 0xD: Interrupt trigger when FIFO level is 14. • 0xE: Interrupt trigger when FIFO level is 15. • 0xF: Interrupt trigger when FIFO level is 16.

4.5.2.5.21 TX_FIFO_CFG

- **Name:** Transmit FIFO Configuration Register
- **Description:** This specifies the Transmit FIFO Configuration Register.
- **Base Address:** 0x4000F200 (master) and 0x4000F000 (slave)
- **Offset:** 0x4c
- **Reset Value:** 0x00000008

Table 4-278 Transmit FIFO Configuration Register

Bits	Field Name	RW	Reset	Description
31:4	RSVD	R		Reserved bits
3:0	TX_FIFO_TL	RW	0x8	These bits program the trigger level in the TX FIFO at which the empty threshold reached interrupt is generated. Trigger Level = TXCHET **Value:** • 0x0: Interrupt trigger when FIFO level is 1. • 0x1: Interrupt trigger when FIFO level is 2. • 0x2: Interrupt trigger when FIFO level is 3. • 0x3: Interrupt trigger when FIFO level is 4. • 0x4: Interrupt trigger when FIFO level is 5. • 0x5: Interrupt trigger when FIFO level is 6. • 0x6: Interrupt trigger when FIFO level is 7. • 0x7: Interrupt trigger when FIFO level is 8. • 0x8: Interrupt trigger when FIFO level is 9. • 0x9: Interrupt trigger when FIFO level is 10. • 0xA: Interrupt trigger when FIFO level is 11. • 0xB: Interrupt trigger when FIFO level is 12.

Bits	Field Name	RW	Reset	Description
				• 0xC: Interrupt trigger when FIFO level is 13. • 0xD: Interrupt trigger when FIFO level is 14. • 0xE: Interrupt trigger when FIFO level is 15. • 0xF: Interrupt trigger when FIFO level is 16.

4.5.2.5.22 RX_FIFO_FLUSH

- **Name:** Receive FIFO Flush Register
- **Description:** This specifies the Receive FIFO Flush Register.
- **Base Address:** 0x4000F200 (master) and 0x4000F000 (slave)
- **Offset:** 0x50
- **Reset Value:** 0x00000000

Table 4-279 Receive FIFO Flush Register

Bits	Field Name	RW	Reset	Description
31:1	RSVD	W		Reserved bits
0	RX_FIFO_RST	W	0x0	Receive channel FIFO reset. Writing a 1 to this register flushes an individual RX FIFO (this is a self clearing bit). An RX channel or block must be disabled prior to writing to this bit. **Value:** • 0x0: Does not flush an individual RX FIFO. • 0x1: Flushes an individual RX FIFO.

4.5.2.5.23 TX_FIFO_FLUSH

- **Name:** Transmit FIFO Flush Register
- **Description:** This specifies the Transmit FIFO Flush Register.
- **Base Address:** 0x4000F200 (master) and 0x4000F000 (slave)
- **Offset:** 0x54
- **Reset Value:** 0x00000000

Table 4-280 Transmit FIFO Flush Register

Bits	Field Name	RW	Reset	Description
31:1	RSVD	W		Reserved bits
0	TX_FIFO_RST	W	0x0	Transmit channel FIFO reset.

Bits	Field Name	RW	Reset	Description
				Writing a 1 to this register flushes an individual TX FIFO (this is a self clearing bit). A TX channel or block must be disabled prior to writing to this bit. **Value:** • 0x0: Does not flush an individual TX FIFO. • 0x1: Flushes an individual TX FIFO.

4.5.2.5.24 RX_DMA

- **Name:** Receiver Block DMA Register
- **Description:** The RX_DMA register allows access to receive channel via a single point rather than through the LEFT_RX_BUF and RIGHT_RX_BUF registers. The receive channel is targeted in a cyclical fashion (starting at the lowest numbered enabled channel) and takes two reads (left and right stereo data) before the component points to the next channel.
- **Base Address:** 0x4000F200 (master) and 0x4000F000 (slave)
- **Offset:** 0x1c0
- **Reset Value:** 0x00000000

Table 4-281 Receiver Block DMA Register

Bits	Field Name	RW	Reset	Description
31:1	RSVD	R		Reserved bits
0	RX_DMA	RW	0x0	Receiver Block DMA Register. This bit is used to cycle repeatedly through the enabled receive channel, reading stereo data pairs.

4.5.2.5.25 RST_RX_DMA

- **Name:** Reset Receiver Block DMA Register
- **Description:** The RX_DMA can be reset to the enabled channel via the RST_RX_DMA register. The RST_RX_DMA register can be written to at any stage of the RX_DMA's read cycle; however, it has no effect when the component is in the middle of a stereo pair read.
- **Base Address:** 0x4000F200 (master) and 0x4000F000 (slave)
- **Offset:** 0x1c4
- **Reset Value:** 0x00000000

Table 4-282 Reset Receiver Block DMA Register

Bits	Field Name	RW	Reset	Description
31:1	RSVD	W		Reserved bits
0	RST_RX_DMA	W	0x0	Reset Receiver Block DMA Register. Writing a 1 to this self-clearing register resets the RX_DMA register mid-cycle to point to the enabled receive channel. **Value:** • 0x0: No effect. • 0x1: Reset receiver block DMA register.

4.5.2.5.26 TX_DMA

- **Name:** Transmitter Block DMA Register
- **Description**: The TX_DMA register functions in a similar way to the RX_DMA register and allows write accesses to all of the enabled transmit channels via a single point rather than through the LEFT_TX_HDG and RIGHT_TX_HDG registers.
- **Base Address:** 0x4000F200 (master) and 0x4000F000 (slave)
- **Offset:** 0x1c8
- **Reset Value:** 0x00000000

Table 4-283 Transmitter Block DMA Register

Bits	Field Name	RW	Reset	Description
31:1	RSVD	R		Reserved bits
0	TX_DMA	RW	0x0	Transmitter Block DMA Register. This bit is used to cycle repeatedly through the enabled transmit channel to allow writing of stereo data pairs.

4.5.2.5.27 RST_TX_DMA

- **Name:** Reset Transmitter Block DMA Register
- **Description:** This register provides the same functionality as the **RST_RX_DMA** register but targets **TX_DMA** instead.
- **Base Address:** 0x4000F200 (master) and 0x4000F000 (slave)
- **Offset:** 0x1cc
- **Reset Value:** 0x00000000

Table 4-284 Reset Transmitter Block DMA Register

Bits	Field Name	RW	Reset	Description
31:1	RSVD	W		Reserved bits
0	RST_TX_DMA	W	0x0	Reset Transmitter Block DMA Register. Writing a 1 to this self-clearing register resets the TX_DMA register mid-cycle to point to the enabled transmit channel. **Value:** • 0x0: No effect. • 0x1: Reset transmitter block DMA register.

4.5.2.5.28 CLK_CFG

- **Name:** I2S Clock Configure Register
- **Description:** This register is used to configure I2S master clock.
- **Base Address:** 0x4000F200 (master) and 0x4000F000 (slave)
- **Offset:** 0x268
- **Reset Value:** 0x00040008

Table 4-285 I2S Clock Configure Register

Bits	Field Name	RW	Reset	Description
31:19	RSVD	R		Reserved bits
18	CLK_SRC_SEL	RW	0x1	Selects a clock divider source. • 0: 96 M • 1: 64 M • 2: 32 M
17	RSVD	R		Reserved bit
16	DIV_EN	RW	0x0	Enables I2S clock divider.
15:12	RSVD	R		Reserved bits
11:0	DIV	RW	0x8	Ratio = 1/(div + 2). Duty cycle is not 50 when the number is odd. For example: div = 0, ratio = 1/2

4.5.2.6 Electrical Specifications

Table 4-286 Electrical specifications

Symbol	Description	Min.	Typ.	Max.	Unit
f_{SCK}	I2S clock frequency		6.021	6.144	MHz
t_{SUSDI}	Data input setup time (from SDI to SCK)	20			ns
t_{HDSDI}	Data input hold time (from SCK to SDI change)	15			ns
t_{VSDO}	Data output valid time (from SCK to SDO valid)			20	ns
DC_{SCK}	SCK clock duty cycle	45		55	%

The I2S timing diagram is as below:

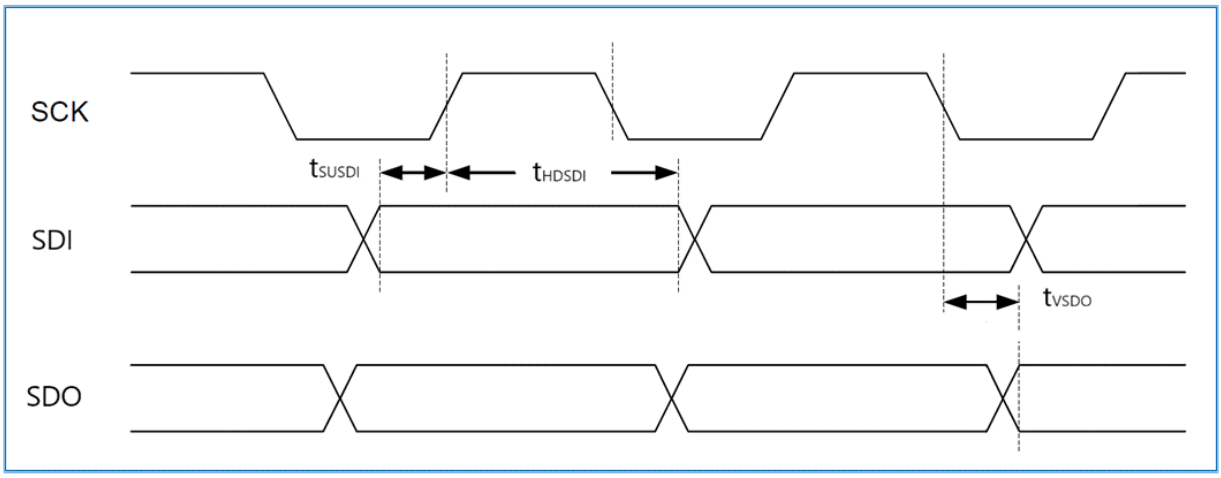

Figure 4-61 I2S timing diagram

4.6 Security Interfaces

4.6.1 KEYRAM

KEYRAM is mainly used for key derivation after power up. It has a memory area for storing key information. There is an embedded encryption and decryption hardware sub-module in KEYRAM to protect the key information.

In security mode, random numbers are generated as masks during each boot stage in order to prevent the external exhaustive method from cracking the key information. The random number masks are stored in KEYRAM.

When the security module needs to obtain the key from the KEYRAM through the key port bus, the KEYRAM embedded sub-module will automatically decrypt and import the key with the confusion of the random number mask into the security module. In addition, the security module also needs to load the random number mask so as to recovery the plain key.

4.6.2 Advanced Encryption Standard (AES)

4.6.2.1 Introduction

Advanced Encryption Standard (AES), also known as Rijndael encryption method in cryptography, is a block encryption standard adopted by the U.S. federal government. This standard is used to replace the Data Encryption Standard (DES). The block length of AES is fixed at 128 bits, and the key length can be 128, 192, or 256 bits. The basic processes of AES include SubBytes, ShiftRows, MixColumns, and AddRoundKey.

4.6.2.2 Main Features

- Compliance with NIST “Advanced Encryption Standard (AES), FIPS Publication 197” from November 2001
- Supports 128-bit, 192-bit, and 256-bit key.
- Supports encryption mode and decryption mode.
- Supports interrupt and query register to report status.
- Supports Electronic Codebook (ECB) and Cipher Block chaining (CBC) modes.
- Supports MCU and DMA operation.

4.6.2.3 Functional Description

4.6.2.3.1 Standard

Table 4-287 AES module standard

Document Code	Name of Standard
FIPS PUB 197	Advanced Encryption Standard (AES)

4.6.2.3.2 Module Overview

The AES co-processor encrypts or decrypts data, using an algorithm and implementation fully compliant with the AES defined in Federal Information Processing Standards (FIPS) publication 197. The device embeds the AES module in order to realize hardware acceleration of calculation.

Copyright © 2024 Shenzhen Goodix Technology Co., Ltd.

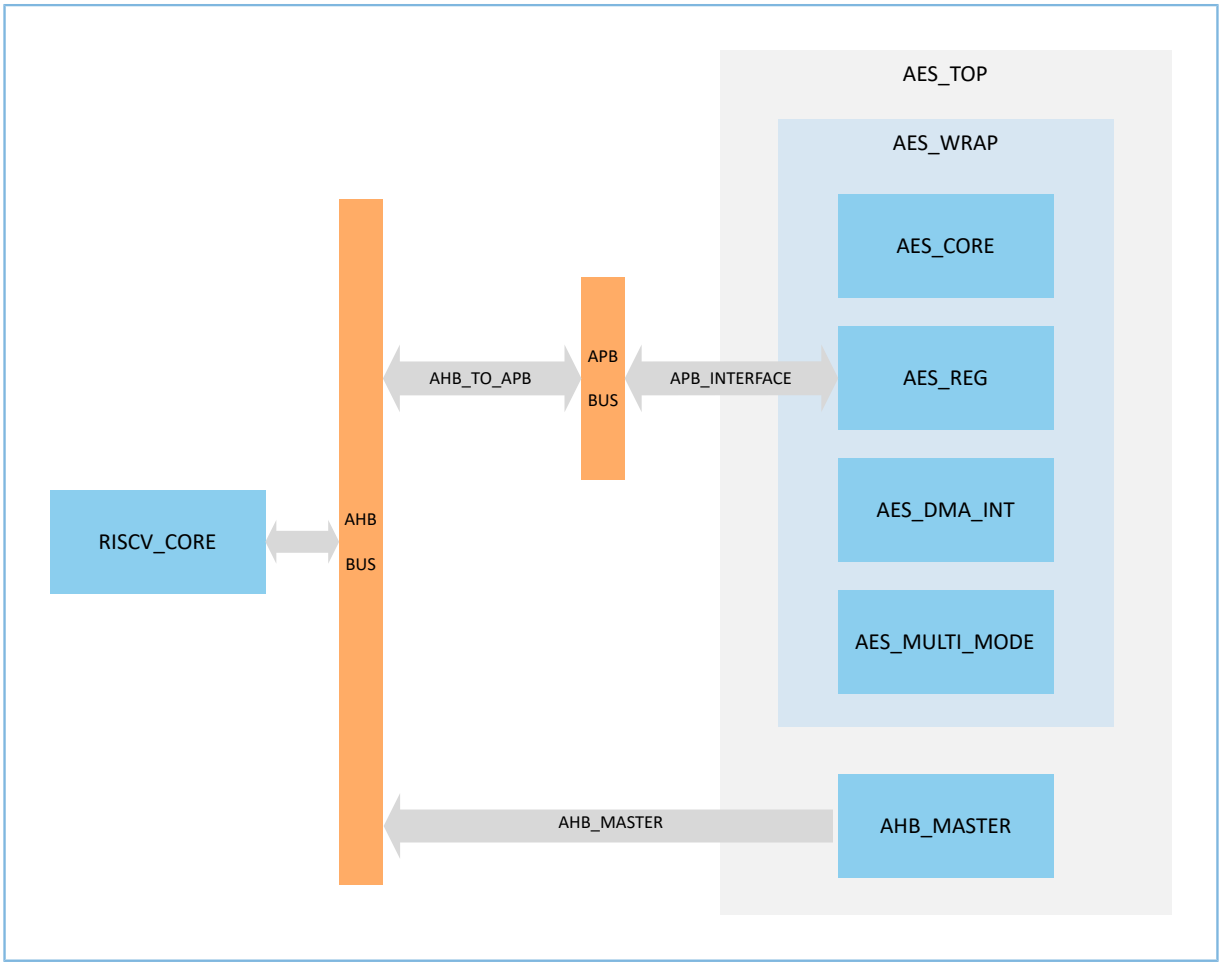

Figure 4-62 AES module in the system

Electronic Codebook (ECB) and Cipher Block Chaining (CBC) are supported for key size of 128 bits, 192 bits, and 256 bits.

Due to the feature of block cipher of AES, the data to be encrypted requires an integer multiple of 16 bytes.

The AES co-processor adopts both 32-bit AHB and APB interfaces. It supports DMA transfers for incoming and outgoing data (through a dedicated integrated DMA).

4.6.2.3.3 Usage

The CFG register can be used to configure relevant parameters for the AES module. CFG.KEY_MODE can be used to configure the key size to 128 bits, 192 bits, or 256 bits. CFG.DEC_ENC_SEL can be used to configure the encryption or decryption mode. CFG.OPT_MODE can be used to configure the working mode of ECB or CBC. CFG.KEY_TYPE can be used to configure the mode of fetching the key.

If the key is configured by MCU, the KEY0 to KEY7 registers need to be configured (128-bit for KEY0 to KEY3 only and 192-bit for KEY0 to KEY5 only). If the key is fetched through AHB, the KEY_ADDR register should be configured.

CTRL.FKEY_EN is configured in order to enable the AES module to fetch the key through AHB master interface. The STAT register is used to indicate the current status of the AES module. The random seeds are set to the INIT_SSI, INIT_SSO, MASK_SSI, and MASK_SSO registers.

The initialization vector (IV) should be set to the INIT_V0 to INIT_V3 registers for AES CBC mode.

If the AES module is in interrupt or DMA operation mode, the INT register needs to be configured.

In MCU operation mode, a block data (128-bit) is written into the DATA_IN0 to DATA_IN3 registers. In DMA operation mode, the XFER_SIZE, RD_START_ADDR, and WR_START_ADDR registers should be configured. The AES module will read the data from the address which is stored in the RD_START_ADDR register.

The AES module can be started through the CTRL register. The STAT register or the interrupt callback function can judge whether the calculation completes.

The plain data or the encrypted data can be read from the DATA_OUT0 to DATA_OUT3 registers in MCU operation mode. In DMA operation mode, the result has been sent to the address which is stored in the WR_START_ADDR register.

The endian of the data needs to be noticed and configured; otherwise it will affect the result of encryption and decryption. If the data is encrypted and decrypted in batches in CBC mode, the encryption result should be set as the IV to the INIT_V0 to INIT_V3 registers.

4.6.2.4 Registers

4.6.2.4.1 CTRL

- **Name:** AES Controller Register
- **Description:** This register acts as a global enable/disable for AES.
- **Base Address:** 0x40015000
- **Offset:** 0x0
- **Reset Value:** 0x00000000

Table 4-288 AES Controller Register

Bits	Field Name	RW	Reset	Description
31:4	RSVD	R		Reserved bits
3	FKEY_EN	W	0x0	Enable AES fetch key by itself through AHB master interface or key port. This register can be cleared by itself when key_valid is set to 1.
2	DMA_MODE_EN	RW	0x0	DMA mode start enable bit. Start N block data encryption or decryption in DMA mode.
1	MCU_MODE_EN	RW	0x0	MCU mode start enable bit. Start a block data encryption or decryption in DMA mode. This signal should be cleared to zero before a new block input data is ready to start.
0	MODULE_EN	RW	0x0	Enables AES module. **Value:**

Bits	Field Name	RW	Reset	Description
				• 0x0: Disable • 0x1: Enable

4.6.2.4.2 CFG

- **Name:** AES Configuration Register
- **Description:** This register acts as a configuration for AES.
- **Base Address:** 0x40015000
- **Offset:** 0x4
- **Reset Value:** 0x00000000

Table 4-289 AES Configuration Register

Bits	Field Name	RW	Reset	Description
31:13	RSVD	R		Reserved bits
12:11	KEY_TYPE	RW	0x0	Key type selection for encryption/decryption **Value:** • 0x0: Configured by MCU (default) • 0x1: Fetched through AHB interface • 0x2: Fetched through key port • 0x3: Reserved
10:8	OPT_MODE	RW	0x0	Selection for operation mode **Value:** • 0x0: Electronic Codebook (ECB) mode • 0x1: Cipher Block Chaining (CBC) mode • 0x2–0x7: Reserved for future application
7	ENDIAN	RW	0x0	Selection for data endian control **Value:** • 0x0: Reverse to big-endian (default) • 0x1: No reverse
6	FIRST_BLK	W	0x0	This register should be set to 1 before starting the first block in normal CBC and DMA CBC modes, and this register will be cleared by itself.
5	LOAD_SEED	W	0x0	Loads seed for LFSR. Seed for LFSR will be reloaded when this register is written with 1. This register can be cleared by itself.

Bits	Field Name	RW	Reset	Description
4	DEC_ENC_SEL	RW	0x0	Selection for encryption/decryption **Value:** • 0x0: Decryption (default) • 0x1: Encryption
3	FULL_MASK_EN	RW	0x0	Full mask enable signal **Value:** • 0x0: Disable (default) • 0x1: Enable (DPA resistance)
2	RSVD	R		Reserved bit
1:0	KEY_MODE	RW	0x0	Key mode selection for encryption/decryption **Value:** • 0x0: 128 bits (default) • 0x1: 192 bits • 0x2: 256 bits • 0x3: Reserved

4.6.2.4.3 STAT

- **Name:** AES Status Register
- **Description:** This is a read-only register used to indicate the current status of AES.
- **Base Address:** 0x40015000
- **Offset:** 0x8
- **Reset Value:** 0x00000000

Table 4-290 AES Status Register

Bits	Field Name	RW	Reset	Description
31:4	RSVD	R		Reserved bits
3	KEY_STAT	R	0x0	Key status, whether the key is ready to read or not **Value:** • 0x0: Not ready • 0x1: Ready
2	DMA_XFE_ERR	R WC	0x0	DMA transfer error. Write 1 to clear this register. **Value:** • 0x0: No error • 0x1: Error

Bits	Field Name	RW	Reset	Description
1	DMA_XFE_CPLT	R	0x0	DMA transfer complete **Value:** • 0x0: Not complete • 0x1: Complete
0	READY	R	0x0	AES result data out ready or not. This signal may be cleared when MCUEN (CTRL[1]) is disabled. **Value:** • 0x0: Not ready • 0x1: Ready

4.6.2.4.4 INT

- **Name:** AES Interrupt Register
- **Description:** This register enables or disables all interrupts generated by the AES. Additionally, it reports the status of the AES interrupts after they have been enabled.
- **Base Address:** 0x40015000
- **Offset:** 0x0C
- **Reset Value:** 0x00000000

Table 4-291 AES Interrupt Register

Bits	Field Name	RW	Reset	Description
31:2	RSVD	R		Reserved bits
1	CPLT_INT_EN	RW	0x0	AES result data complete interrupt **Value:** • 0x0: Disable • 0x1: Enable
0	CPLT_INT_FLAG	R WC	0x0	AES result data complete interrupt flag. Write 1 to clear. **Read:** • 0x0: Not interrupt • 0x1: Interrupt **Write:** • 0x0: Not effect • 0x1: Clear

4.6.2.4.5 XFER_SIZE

- **Name:** AES Transfer Size Register
- **Description:** This register sets a size for AES DMA mode transfer.
- **Base Address:** 0x40015000
- **Offset:** 0x10
- **Reset Value:** 0x00000000

Table 4-292 AES Transfer Size Register

Bits	Field Name	RW	Reset	Description
31:15	RSVD	R		Reserved bits
14:0	SIZE	RW	0x0	Total transfer size. Up to 32 KB • 0x000f: 1 block • 0x001f: 2 blocks • 0x002f: 3 blocks … • 0x7fff: 2048 blocks

4.6.2.4.6 RD_START_ADDR

- **Name:** AES Read Start Address Register
- **Description:** This register acts as a start address for AES DMA reading.
- **Base Address:** 0x40015000
- **Offset:** 0x14
- **Reset Value:** 0x00000000

Table 4-293 AES Read Start Address Register

Bits	Field Name	RW	Reset	Description
31:0	ADDR	RW	0x0	In DMA mode, sets the read start address of transfer.

4.6.2.4.7 WR_START_ADDR

- **Name:** AES Write Start Address Register
- **Description:** This register acts as a start address for AES DMA writing.
- **Base Address:** 0x40015000
- **Offset:** 0x18
- **Reset Value:** 0x00000000

Table 4-294 AES Write Start Address Register

Bits	Field Name	RW	Reset	Description
31:0	ADDR	RW	0x0	In DMA mode, sets the write start address of transfer.

4.6.2.4.8 KEY_ADDR

- **Name:** AES Key Address Register
- **Description:** This register acts as a key address for AES encryption or decryption.
- **Base Address:** 0x40015000
- **Offset:** 0x1C
- **Reset Value:** 0x00000000

Table 4-295 AES Key Address Register

Bits	Field Name	RW	Reset	Description
31:0	ADDR	RW	0x0	Indicates the AES key address in memory.

4.6.2.4.9 DATA_OUT0

- **Name:** AES Data Output 0 Register
- **Description:** This register is a 32-bit read-only buffer for the result data[127:96] from AES.
- **Base Address:** 0x40015000
- **Offset:** 0x20
- **Reset Value:** 0x00000000

Table 4-296 AES Data Output 0 Register

Bits	Field Name	RW	Reset	Description
31:0	DATA_OUT0	R	0x0	AES result data[127:96]

4.6.2.4.10 DATA_OUT1

- **Name:** AES Data Output 1 Register
- **Description:** This register is a 32-bit read-only buffer for the result data[95:64] from AES.
- **Base Address:** 0x40015000
- **Offset:** 0x24
- **Reset Value:** 0x00000000

Table 4-297 AES Data Output 1 Register

Bits	Field Name	RW	Reset	Description
31:0	DATA_OUT1	R	0x0	AES result data[95:64]

4.6.2.4.11 DATA_OUT2

- **Name:** AES Data Output 2 Register
- **Description:** This register is a 32-bit read-only buffer for the result data[63:32] from AES.
- **Base Address:** 0x40015000
- **Offset:** 0x28
- **Reset Value:** 0x00000000

Table 4-298 AES Data Output 2 Register

Bits	Field Name	RW	Reset	Description
31:0	DATA_OUT2	R	0x0	AES result data[63:32]

4.6.2.4.12 DATA_OUT3

- **Name:** AES Data Output 3 Register
- **Description:** This register is a 32-bit read-only buffer for the result data[31:0] from AES.
- **Base Address:** 0x40015000
- **Offset:** 0x2C
- **Reset Value:** 0x00000000

Table 4-299 AES Data Output 3 Register

Bits	Field Name	RW	Reset	Description
31:0	DATA_OUT3	R	0x0	AES result data[31:0]

4.6.2.4.13 KEY0

- **Name:** AES Key 0 Register
- **Description:** This register is a 32-bit write-only buffer for the key data[255:224] for AES.
- **Base Address:** 0x40015000
- **Offset:** 0x30
- **Reset Value:** 0x00000000

Table 4-300 AES Key 0 Register

Bits	Field Name	RW	Reset	Description
31:0	KEY0	W	0x0	AES key[255:224]

4.6.2.4.14 KEY1

- **Name:** AES Key 1 Register
- **Description:** This register is a 32-bit write-only buffer for the key data[223:192] for AES.
- **Base Address:** 0x40015000
- **Offset:** 0x34
- **Reset Value:** 0x00000000

Table 4-301 AES Key 1 Register

Bits	Field Name	RW	Reset	Description
31:0	KEY1	W	0x0	AES key[223:192]

4.6.2.4.15 KEY2

- **Name:** AES Key 2 Register
- **Description:** This register is a 32-bit write-only buffer for the key data[191:160] for AES.
- **Base Address:** 0x40015000
- **Offset:** 0x38
- **Reset Value:** 0x00000000

Table 4-302 AES Key 2 Register

Bits	Field Name	RW	Reset	Description
31:0	KEY2	W	0x0	AES key[191:160]

4.6.2.4.16 KEY3

- **Name:** AES Key 3 Register
- **Description:** This register is a 32-bit write-only buffer for the key data[159:128] for AES.
- **Base Address:** 0x40015000
- **Offset:** 0x3C
- **Reset Value:** 0x00000000

Table 4-303 AES Key 3 Register

Bits	Field Name	RW	Reset	Description
31:0	KEY3	W	0x0	AES key[159:128]

4.6.2.4.17 KEY4

- **Name:** AES Key 4 Register
- **Description:** This register is a 32-bit write-only buffer for the key data[127:96] for AES.
- **Base Address:** 0x40015000
- **Offset:** 0x40
- **Reset Value:** 0x00000000

Table 4-304 AES Key 4 Register

Bits	Field Name	RW	Reset	Description
31:0	KEY4	W	0x0	AES key[127:96]

4.6.2.4.18 KEY5

- **Name:** AES Key 5 Register
- **Description:** This register is a 32-bit write-only buffer for the key data[95:64] for AES.
- **Base Address:** 0x40015000
- **Offset:** 0x44
- **Reset Value:** 0x00000000

Table 4-305 AES Key 5 Register

Bits	Field Name	RW	Reset	Description
31:0	KEY5	W	0x0	AES key[95:64]

4.6.2.4.19 KEY6

- **Name:** AES Key 6 Register
- **Description:** This register is a 32-bit write-only buffer for the key data[63:32] for AES.
- **Base Address:** 0x40015000
- **Offset:** 0x48
- **Reset Value:** 0x00000000

Table 4-306 AES Key 6 Register

Bits	Field Name	RW	Reset	Description
31:0	KEY6	W	0x0	AES key[63:32]

4.6.2.4.20 KEY7

- **Name:** AES Key 7 Register
- **Description:** This register is a 32-bit write-only buffer for the key data[31:0] for AES.
- **Base Address:** 0x40015000
- **Offset:** 0x4C
- **Reset Value:** 0x00000000

Table 4-307 AES Key 7 Register

Bits	Field Name	RW	Reset	Description
31:0	KEY7	W	0x0	AES key[31:0]

4.6.2.4.21 INIT_SSI

- **Name:** AES Sbox Initial Seed Input Register
- **Description:** This register acts as an initial seed input for Sbox.
- **Base Address:** 0x40015000
- **Offset:** 0x50
- **Reset Value:** 0x00000000

Table 4-308 AES Sbox Initial Seed Input Register

Bits	Field Name	RW	Reset	Description
31:0	SEED	RW	0x0	Sbox initial seed input

4.6.2.4.22 INIT_SSO

- **Name:** AES Sbox Initial Seed Output Register
- **Description:** This register acts as an initial seed output for Sbox.
- **Base Address:** 0x40015000
- **Offset:** 0x54
- **Reset Value:** 0x00000000

Table 4-309 AES Sbox Initial Seed Output Register

Bits	Field Name	RW	Reset	Description
31:0	SEED	RW	0x0	Sbox initial seed output

4.6.2.4.23 MASK_SSI

- **Name:** AES Sbox Seed Input Mask Register
- **Description:** This register acts as a mask for Sbox initial seed input mask.
- **Base Address:** 0x40015000
- **Offset:** 0x58
- **Reset Value:** 0x00000000

Table 4-310 AES Sbox Seed Input Mask Register

Bits	Field Name	RW	Reset	Description
31:0	MASK	RW	0x0	Sbox initial seed input mask

4.6.2.4.24 MASK_SSO

- **Name:** AES Sbox Seed Output Mask Register
- **Description:** This register acts as a mask for Sbox initial seed output mask.
- **Base Address:** 0x40015000
- **Offset:** 0x5C
- **Reset Value:** 0x00000000

Table 4-311 AES Sbox Seed Output Mask Register

Bits	Field Name	RW	Reset	Description
31:0	MASK	RW	0x0	Sbox initial seed output mask

4.6.2.4.25 INIT_V0

- **Name:** AES Initialization Vector 0 Register
- **Description:** This register acts as an initialization vector[127:96] for CBC mode.
- **Base Address:** 0x40015000
- **Offset:** 0x60
- **Reset Value:** 0x00000000

Table 4-312 AES Initialization Vector 0 Register

Bits	Field Name	RW	Reset	Description
31:0	VECTOR	W	0x0	Initialization vector[127:96] for CBC mode

4.6.2.4.26 INIT_V1

- **Name:** AES Initialization Vector 1 Register
- **Description:** This register acts as an initialization vector[95:64] for CBC mode.
- **Base Address:** 0x40015000
- **Offset:** 0x64
- **Reset Value:** 0x00000000

Table 4-313 AES Initialization Vector 1 Register

Bits	Field Name	RW	Reset	Description
31:0	VECTOR	W	0x0	Initialization vector[95:64] for CBC mode

4.6.2.4.27 INIT_V2

- **Name:** AES Initialization Vector 2 Register
- **Description:** This register acts as an initialization vector[63:32] for CBC mode.
- **Base Address:** 0x40015000
- **Offset:** 0x68
- **Reset Value:** 0x00000000

Table 4-314 AES Initialization Vector 2 Register

Bits	Field Name	RW	Reset	Description
31:0	VECTOR	W	0x0	Initialization vector[63:32] for CBC mode

4.6.2.4.28 INIT_V3

- **Name:** AES Initialization Vector 3 Register
- **Description:** This register acts as an initialization vector[31:0] for CBC mode.
- **Base Address:** 0x40015000
- **Offset:** 0x6C
- **Reset Value:** 0x00000000

Table 4-315 AES Initialization Vector 3 Register

Bits	Field Name	RW	Reset	Description
31:0	VECTOR	W	0x0	Initialization vector[31:0] for CBC mode

4.6.2.4.29 DATA_IN0

- **Name:** AES Data Input 0 Register
- **Description:** This register is a 32-bit only write buffer for the input data[127:96] for encryption or decryption.
- **Base Address:** 0x40015000
- **Offset**: 0x70
- **Reset Value:** 0x00000000

Table 4-316 AES Data Input 0 Register

Bits	Field Name	RW	Reset	Description
31:0	DATA_IN0	W	0x0	Input data[127:96] for encryption or decryption. This register is valid in MCU mode.

4.6.2.4.30 DATA_IN1

- **Name:** AES Data Input 1 Register
- **Description:** This register is a 32-bit only write buffer for the input data[95:64] for encryption or decryption.
- **Base Address:** 0x40015000
- **Offset:** 0x74
- **Reset Value:** 0x00000000

Table 4-317 AES Data Input 1 Register

Bits	Field Name	RW	Reset	Description
31:0	DATA_IN1	W	0x0	Input data[95:64] for encryption or decryption. This register is valid in MCU mode.

4.6.2.4.31 DATA_IN2

- **Name:** AES Data Input 2 Register
- **Description:** This register is a 32-bit only write buffer for the input data[63:32] for encryption or decryption.
- **Base Address:** 0x40015000
- **Offset:** 0x78
- **Reset Value:** 0x00000000

Table 4-318 AES Data Input 2 Register

Bits	Field Name	RW	Reset	Description
31:0	DATA_IN2	W	0x0	Input data[63:32] for encryption or decryption. This register is valid in MCU mode.

4.6.2.4.32 DATA_IN3

- **Name:** AES Data Input 3 Register
- **Description:** This register is a 32-bit only write buffer for the input data[31:0] for encryption or decryption.
- **Base Address:** 0x40015000
- **Offset:** 0x7C
- **Reset Value:** 0x00000000

Table 4-319 AES Data Input 3 Register

Bits	Field Name	RW	Reset	Description
31:0	DATA_IN3	W	0x0	Input data[31:0] for encryption or decryption. This register is valid in MCU mode.

4.6.2.4.33 KEYPORT_MASK

- **Name:** AES Keyport Mask Register
- **Description:** This register acts as a mask for key from key port.
- **Base Address:** 0x40015000
- **Offset:** 0x80
- **Reset Value:** 0x00000000

Table 4-320 AES Keyport Mask Register

Bits	Field Name	RW	Reset	Description
31:0	MASK	W	0x0	Mask for key from key port

4.6.3 Hash-based Message Authentication Code (HMAC)

4.6.3.1 Introduction

Hash-based Message Authentication Code (HMAC) is a mechanism for message authentication by using hash functions in cryptography. The message authentication provided by HMAC includes two aspects:

- Message integrity authentication: This can prove that the message has not been modified during transmission.

- Source identity authentication: Because both sides of the communication share the authentication key, receiver can verify that the source sending the data is consistent with the claim so that the receiver can reliably confirm that the received message is consistent with the sent message.

4.6.3.2 Main Features

- Compatible with SHA-256
- Supports custom initial hash values.
- Supports querying registers and interrupts to report status.
- Supports MCU mode and DMA mode.

4.6.3.3 Functional Description

4.6.3.3.1 Standard

Table 4-321 HMAC module standard

Document Code	Name of Standard
FIPS PUB 180-4	Secure Hash Standard (SHS) (Compliant with SHA-256)
FIPS PUB 198-1	Keyed-Hash Message Authentication Code (HMAC)

4.6.3.3.2 Module Overview

The HMAC co-processor authenticates data, using an algorithm and implementation fully compliant with the Keyed-Hash Message Authentication Code defined in Federal Information Processing Standards (FIPS) Publication 198. GR5525 embeds the HMAC module in order to realize hardware acceleration of calculation.

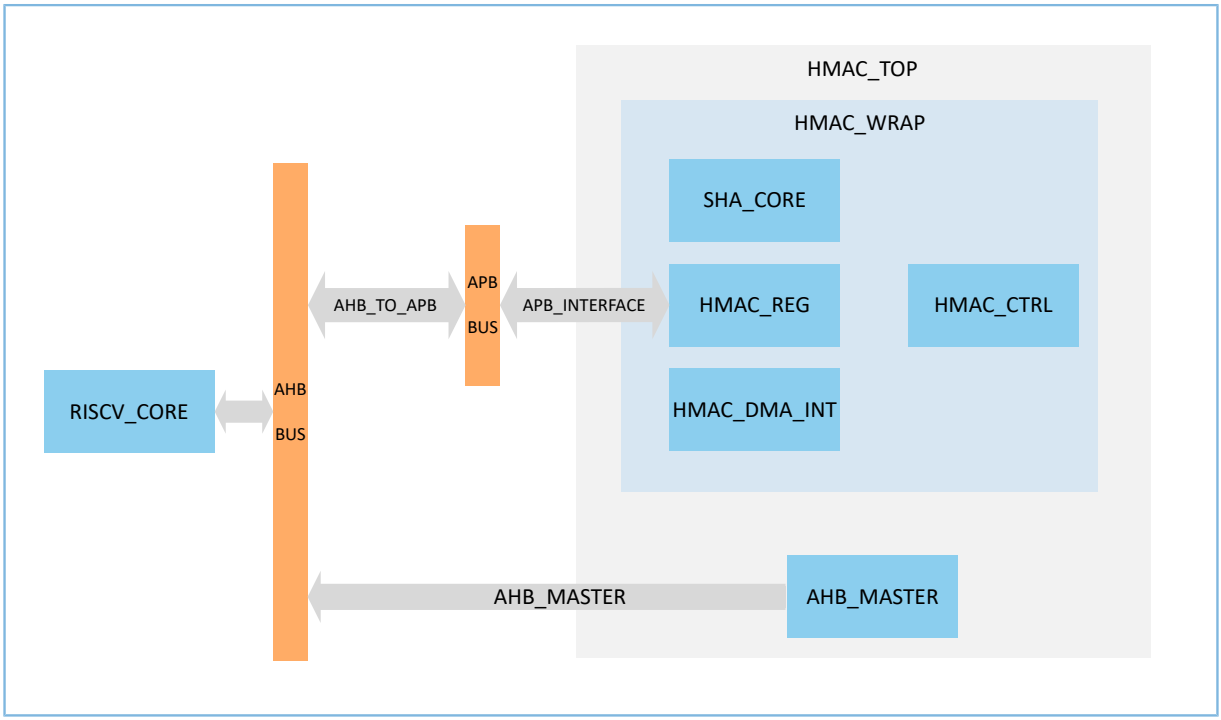

Figure 4-63 HMAC module in the system

Multiple modes are supported (SHA256, HMAC-SHA256) for a key size of 256 bits.

Information preprocessing is required for input data, including additional padding bits and additional length value.

The HMAC co-processor adopts both 32-bit AHB and APB interfaces. It supports DMA transfers for incoming and outgoing data (through a dedicated integrated DMA).

4.6.3.3.3 Usage

The CFG register can be used to configure relevant parameters for the HMAC module. CFG.CALC_TYPE can be used to configure the working mode including SHA-256 and HMAC. If the working mode is set to HMAC, the mode of fetching the key should be configured through CFG.KEY_TYPE.

If the key is configured by MCU, the KEY0 to KEY7 registers need to be configured. If the key is fetched through AHB, the KEY_ADDR register should be configured.

The SER_HASH0 to USER_HASH7 registers are used to customize the initial HASH values if the CFG register enables users to define the initial HASH.

If the HMAC module is in interrupt or DMA operation mode, the INT register needs to be configured.

In DMA operation mode, the XFER_SIZE, RD_START_ADDR, and WR_START_ADDR registers should be configured. If misalignment is required when MCU reads or writes data, the last two bits of the RD_START_ADDR and the WR_START_ADDR registers can be configured.

The HMAC module can be started through the CTRL register. CTRL.KEY_EN is configured in order to enable the HMAC module to fetch the key through AHB master interface.

In MCU operation mode, a block data (512-bit) is continuously written into the DATA_IN register. In DMA operation mode, the HMAC module will read the data from the address which is stored in the RD_START_ADDR register.

The STAT register or interrupt callback function can judge whether the calculation completes. Before writing next block data into the DATA_IN register, the INT register needs to be cleared.

In MCU operation mode, the result of the SHA-256 or HMAC can be obtained by repeatedly reading the DATA_OUT register until getting 256-bit data. In DMA operation mode, the result has been sent to the address which is stored in the WR_START_ADDR register.

4.6.3.4 Registers

4.6.3.4.1 CTRL

- **Name:** HMAC Controller Register
- **Description:** This register acts as a global enable/disable for HMAC.
- **Base Address:** 0x40015800
- **Offset:** 0x0
- **Reset Value:** 0x00000000

Table 4-322 HMAC Controller Register

Bits	Field Name	RW	Reset	Description
31:4	RSVD	R		Reserved bits
3	LST_TX	W	0x0	Last block in MCU mode or last DMA transfer. This signal may be cleared by itself when hmac_ready was set.
2	KEY_EN	W	0x0	Enable HMAC fetch key by itself through AHB master interface or key port. This signal may be cleared by itself when hmac_key_valid was set.
1	DMA_START	RW	0x0	DMA mode start enable. Starting DMA transfer, this signal should be cleared after DMA has finished all transfers.
0	EN	RW	0x0	HMAC enable, high valid for whole HMAC processing. Users must disable it once all HMAC blocks complete. This signal should be asserted until all blocks complete.

4.6.3.4.2 CFG

- **Name:** HMAC Configuration Register
- **Description:** This register acts as a configuration for HMAC.
- **Base Address:** 0x40015800
- **Offset:** 0x4
- **Reset Value:** 0x00000000

Table 4-323 HMAC Configuration Register

Bits	Field Name	RW	Reset	Description
31:6	RSVD	R		Reserved bits
5	PRIVT_MOD	RW	0x0	To resist DPA, select private mode. **Value:** • 0x0: Standard mode • 0x1: Private mode
4	CALC_TYPE	RW	0x0	Selects a calculation type. **Value:** • 0x0: HMAC • 0x1: SHA
3:2	KEY_TYPE	RW	0x0	Selects a key type. **Value:** • 0x0: Configured by MCU • 0x1: Fetched through AHB interface • 0x2: Fetched through key port • 0x3: Reserved
1	ENDIAN	RW	0x0	Selection for endian control **Value:** • 0x0: Reverse to big-endian (default) • 0x1: No reverse
0	HASH	RW	0x0	Enables user-defined hash. **Value:** • 0x0: Initial Hash from standard (default) • 0x1: User-defined initial Hash enabled

4.6.3.4.3 STAT

- **Name:** HMAC Status Register
- **Description:** This is a read-only register used to indicate the current status of HMAC.
- **Base Address:** 0x40015800
- **Offset:** 0x8
- **Reset Value:** 0x00000000

Table 4-324 HMAC Status Register

Bits	Field Name	RW	Reset	Description
31:6	RSVD	R		Reserved bits
5	DMA_TX_DONE	R	0x0	HMAC DMA transfer done or not
4	HMAC_READY	R	0x0	HMAC process status **Value:** • 0x0: Result data is not ready. • 0x1: Result data is valid.
3	KEY_VALID	R	0x0	HMAC has fetched key or not
2	DMA_TX_ERR	R WC	0x0	HMAC DMA transfer error. Write 1 to clear.
1	DMA_MSG_DONE	R	0x0	If the number of all block messages is bigger than the transfer size, it indicates all block messages are sent when this signal is set.
0	HASH_READY	R	0x0	Hash process status **Value:** • 0x0: Result data is not ready. • 0x1: Result data is valid.

4.6.3.4.4 XFER_SIZE

- **Name:** HMAC Transfer Size Register
- **Description:** This register sets a size for HMAC DMA mode transfer.
- **Base Address:** 0x40015800
- **Offset:** 0xC
- **Reset Value:** 0x00000000

Table 4-325 HMAC Transfer Size Register

Bits	Field Name	RW	Reset	Description
31:15	RSVD	R		Reserved bits
14:0	SIZE	RW	0x0	Total transfer size. Up to 32 KB • 0x003F: 1 block • 0x007F: 2 blocks • 0x00BF: 3 blocks …

Bits	Field Name	RW	Reset	Description
				• 0x7FFF: 512 blocks

4.6.3.4.5 INT

- **Name:** HMAC Interrupt Register
- **Description:** This register enables or disables all interrupts generated by the HMAC. Additionally, it reports the status of the HMAC interrupts after they have been enabled.
- **Base Address:** 0x40015800
- **Offset:** 0x10
- **Reset Value:** 0x00000000

Table 4-326 HMAC Interrupt Register

Bits	Field Name	RW	Reset	Description
31:2	RSVD	R		Reserved bits
1	EN	RW	0x0	HMAC result data complete interrupt **Value:** • 0x0: Disable • 0x1: Enable
0	DONE	R WC	0x0	HMAC result data complete interrupt flag. Write 1 to clear. **Read:** • 0x0: Not interrupt • 0x1: Interrupt **Write:** • 0x0: Not effect • 0x1: Clear

4.6.3.4.6 RD_START_ADDR

- **Name:** HMAC Read Start Address Register
- **Description:** This register acts as a start address for HMAC DMA reading.
- **Base Address:** 0x40015800
- **Offset:** 0x14
- **Reset Value:** 0x00000000

Table 4-327 HMAC Read Start Address Register

Bits	Field Name	RW	Reset	Description
31:0	ADDR	RW	0x0	In DMA mode, set the read start address of transfer.

4.6.3.4.7 WR_START_ADDR

- **Name:** HMAC Write Start Address Register
- **Description:** This register acts as a start address for HMAC DMA writing.
- **Base Address:** 0x40015800
- **Offset:** 0x18
- **Reset Value:** 0x00000000

Table 4-328 HMAC Write Start Address Register

Bits	Field Name	RW	Reset	Description
31:0	ADDR	RW	0x0	In DMA mode, set the write start address of transfer.

4.6.3.4.8 USER_HASH_0

- **Name:** HMAC User Hash 0 Register
- **Description:** This register acts as a user Hash value for HASH calculation.
- **Base Address:** 0x40015800
- **Offset:** 0x20
- **Reset Value:** 0x00000000

Table 4-329 HMAC User Hash 0 Register

Bits	Field Name	RW	Reset	Description
Bits	Field Name	RW	Reset	Description
31:0	HASH_0	RW	0x0	If UHASH is selected, you can configure a user-defined Hash value by using USER_HASH_0/1/2/3/4/5/6/7. User defined Hash value[255:224]

4.6.3.4.9 USER_HASH_1

- **Name:** HMAC User Hash 1 Register
- **Description:** This register acts as a user Hash value for HASH calculation.
- **Base Address:** 0x40015800
- **Offset:** 0x24

- **Reset Value:** 0x00000000

Table 4-330 HMAC User Hash 1 Register

Bits	Field Name	RW	Reset	Description
31:0	HASH_1	RW	0x0	If UHASH is selected, you can configure a user-defined Hash value by using USER_HASH_0/1/2/3/4/5/6/7. User defined Hash value[223:192]

4.6.3.4.10 USER_HASH_2

- **Name:** HMAC User Hash 2 Register
- **Description:** This register acts as a user Hash value for HASH calculation.
- **Base Address:** 0x40015800
- **Offset:** 0x28
- **Reset Value:** 0x00000000

Table 4-331 HMAC User Hash 2 Register

Bits	Field Name	RW	Reset	Description
31:0	HASH_2	RW	0x0	If UHASH is selected, you can configure a user-defined Hash value by using USER_HASH_0/1/2/3/4/5/6/7. User defined Hash value[191:160]

4.6.3.4.11 USER_HASH_3

- **Name:** HMAC User Hash 3 Register
- **Description:** This register acts as a user Hash value for HASH calculation.
- **Base Address:** 0x40015800
- **Offset:** 0x2C
- **Reset Value:** 0x00000000

Table 4-332 HMAC User Hash 3 Register

Bits	Field Name	RW	Reset	Description
31:0	HASH_3	RW	0x0	If UHASH is selected, you can configure a user-defined Hash value by using USER_HASH_0/1/2/3/4/5/6/7. User defined Hash value[159:128]

4.6.3.4.12 USER_HASH_4

- **Name:** HMAC User Hash 4 Register
- **Description:** This register acts as a user Hash value for HASH calculation
- **Base Address:** 0x40015800
- **Offset:** 0x30
- **Reset Value:** 0x00000000

Table 4-333 HMAC User Hash 4 Register

Bits	Field Name	RW	Reset	Description
31:0	HASH_4	RW	0x0	If UHASH is selected, you can configure a user-defined Hash value with USER_HASH_0/1/2/3/4/5/6/7. User defined Hash value[127:96]

4.6.3.4.13 USER_HASH_5

- **Name:** HMAC User Hash 5 Register
- **Description:** This register acts as a user Hash value for HASH calculation.
- **Base Address:** 0x40015800
- **Offset:** 0x34
- **Reset Value:** 0x00000000

Table 4-334 HMAC User Hash 5 Register

Bits	Field Name	RW	Reset	Description
31:0	HASH_5	RW	0x0	If UHASH is selected, you can configure a user-defined Hash value by using USER_HASH_0/1/2/3/4/5/6/7. User defined Hash value[95:64]

4.6.3.4.14 USER_HASH_6

- **Name:** HMAC User Hash 6 Register
- **Description:** This register acts as a user Hash value for HASH calculation.
- **Base Address:** 0x40015800
- **Offset:** 0x38
- **Reset Value:** 0x00000000

Table 4-335 HMAC User Hash 6 Register

Bits	Field Name	RW	Reset	Description
31:0	HASH_6	RW	0x0	If UHASH is selected, you can configure a user-defined Hash value by using USER_HASH_0/1/2/3/4/5/6/7. User defined Hash value[63:32]

4.6.3.4.15 USER_HASH_7

- **Name:** HMAC User Hash 7 Register
- **Description:** This register acts as a user Hash value for HASH calculation.
- **Base Address:** 0x40015800
- **Offset:** 0x3C
- **Reset Value:** 0x00000000

Table 4-336 HMAC User Hash 7 Register

Bits	Field Name	RW	Reset	Description
31:0	HASH_7	RW	0x0	If UHASH is selected, you can configure a user-defined Hash value by using USER_HASH_0/1/2/3/4/5/6/7. User defined Hash value[31:0]

4.6.3.4.16 DATA_OUT

- **Name:** HMAC Data Output Register
- **Description:** This register is a 32-bit only read buffer for the result data from HMAC.
- **Base Address:** 0x40015800
- **Offset:** 0x40
- **Reset Value:** 0x00000000

Table 4-337 HMAC Data Output Register

Bits	Field Name	RW	Reset	Description
31:0	DATA	R	0x0	In MCU mode, users can read out results with this register: When all blocks of messages are processed, 256-bit HMAC data should be read out 8 or 9 times. HMAC data can be read when hmac_ready is valid.

4.6.3.4.17 DATA_IN

- **Name:** HMAC Data Input Register

- **Description:** This register is a 32-bit only write buffer for the input data for HMAC.
- **Base Address:** 0x40015800
- **Offset:** 0x44
- **Reset Value:** 0x00000000

Table 4-338 HMAC Data Input Register

Bits	Field Name	RW	Reset	Description
Bits	Field Name	RW	Reset	Description
31:0	DATA	W	0x0	In MCU mode, users can configure input data with this register: 32-bit data should be sent 16 or 17 times.

4.6.3.4.18 KEY0

- **Name:** HMAC Key 0 Register
- **Description:** This register is a 32-bit only write buffer for the key data[255:224] for HMAC.
- **Base Address:** 0x40015800
- **Offset:** 0x48
- **Reset Value:** 0x00000000

Table 4-339 HMAC Key 0 Register

Bits	Field Name	RW	Reset	Description
31:0	KEY0	W	0x0	Key is set as 256 bits (8 words). WORD0

4.6.3.4.19 KEY1

- **Name:** HMAC Key 1 Register
- **Description:** This register is a 32-bit only write buffer for the key data[223:192] for HMAC.
- **Base Address:** 0x40015800
- **Offset:** 0x4C
- **Reset Value:** 0x00000000

Table 4-340 HMAC Key 1 Register

Bits	Field Name	RW	Reset	Description
31:0	KEY1	W	0x0	Key is set as 256 bits (8 words). WORD1

4.6.3.4.20 KEY2

- **Name:** HMAC Key 2 Register
- **Description:** This register is a 32-bit only write buffer for the key data[191:160] for HMAC.
- **Base Address:** 0x40015800
- **Offset:** 0x50
- **Reset Value:** 0x00000000

Table 4-341 HMAC Key 2 Register

Bits	Field Name	RW	Reset	Description
31:0	KEY2	W	0x0	Key is set as 256 bits (8 words). WORD2

4.6.3.4.21 KEY3

- **Name:** HMAC Key 3 Register
- **Description:** This register is a 32-bit only write buffer for the key data[159:128] for HMAC.
- **Base Address:** 0x40015800
- **Offset:** 0x54
- **Reset Value:** 0x00000000

Table 4-342 HMAC Key 3 Register

Bits	Field Name	RW	Reset	Description
31:0	KEY3	W	0x0	Key is set as 256 bits (8 words). WORD3

4.6.3.4.22 KEY4

- **Name:** HMAC Key 4 Register
- **Description:** This register is a 32-bit only write buffer for the key data[127:96] for HMAC.
- **Base Address:** 0x40015800
- **Offset:** 0x58
- **Reset Value:** 0x00000000

Table 4-343 HMAC Key 4 Register

Bits	Field Name	RW	Reset	Description
31:0	KEY4	W	0x0	Key is set as 256 bits (8 words).

Bits	Field Name	RW	Reset	Description
				WORD4

4.6.3.4.23 KEY5

- **Name:** HMAC Key 5 Register
- **Description:** This register is a 32-bit only write buffer for the key data[95:64] for HMAC.
- **Base Address:** 0x40015800
- **Offset:** 0x5C
- **Reset Value:** 0x00000000

Table 4-344 HMAC Key 5 Register

Bits	Field Name	RW	Reset	Description
31:0	KEY5	W	0x0	Key is set as 256 bits (8 words). WORD5

4.6.3.4.24 KEY6

- **Name:** HMAC Key 6 Register
- **Description:** This register is a 32-bit only write buffer for the key data[63:32] for HMAC.
- **Base Address:** 0x40015800
- **Offset:** 0x60
- **Reset Value:** 0x00000000

Table 4-345 HMAC Key 6 Register

Bits	Field Name	RW	Reset	Description
Bits	Field Name	RW	Reset	Description
31:0	KEY6	W	0x0	Key is set as 256 bits (8 words). WORD6

4.6.3.4.25 KEY7

- **Name:** HMAC Key 7 Register
- **Description:** This register is a 32-bit only write buffer for the key data[31:0] for HMAC.
- **Base Address:** 0x40015800
- **Offset:** 0x64
- **Reset Value:** 0x00000000

Table 4-346 HMAC Key 7 Register

Bits	Field Name	RW	Reset	Description
31:0	KEY7	W	0x0	Key is set as 256 bits (8 words). WORD7

4.6.3.4.26 KEY_ADDR

- **Name:** HMAC Key Address Register
- **Description:** This register acts as a key address for HMAC.
- **Base Address:** 0x40015800
- **Offset:** 0x68
- **Reset Value:** 0x00000000

Table 4-347 HMAC Key Address Register

Bits	Field Name	RW	Reset	Description
31:0	KEY_ADDR	RW	0x0	Indicates the HMAC key address in memory.

4.6.3.4.27 KEYPORT_MASK

- **Name:** HMAC Keyport Mask Register
- **Description:** This register acts as a mask for key from key port.
- **Base Address:** 0x40015800
- **Offset:** 0x6C
- **Reset Value:** 0x00000000

Table 4-348 HMAC Keyport Mask Register

Bits	Field Name	RW	Reset	Description
31:0	MASK	W	0x0	Mask for key from key port

4.6.4 Public Key Cryptography (PKC)

4.6.4.1 Introduction

Public Key Cryptography (PKC) algorithm, also known as asymmetric key algorithm, uses a pair of keys: a public key and a private key. The security of the private key should be ensured. The public key can be published. Data encrypted with the public key can only be decrypted with the private key, and vice versa. The public key algorithm does not need an online key server, and the key distribution protocol is simple, which greatly simplifies the key management. In addition to the encryption function, the public key algorithm provides key exchange function and digital signature function.

4.6.4.2 Main Features

- Compliance with FIPS 186-4 standard
- Supports Scalar multiplication for P-256 elliptic curve
- Supports diverse modular operations:
  - Montgomery modular multiplication for a configurable 256–2048 bits' length
  - Partial Montgomery Inversion for a configurable 256–2048 bits' length
  - Modular addition for a configurable 256–2048 bits' length
  - Modular subtraction for a configurable 256–2048 bits' length
  - Modular comparison for a configurable 256–2048 bits' length
  - Modular left-shift operation for a configurable 256–2048 bits' length
  - Big integer multiplication for a configurable 256–1024 bits' length
  - Big integer addition for a configurable 256–2048 bits' length
  - Hardware dummy multiplication
- Supports random clocking gating.
- Protects register read/write operations (registers cannot be read or written in the DISABLE module state).
- Supports interrupt and query.

4.6.4.3 Functional Description

4.6.4.3.1 Standard

Table 4-349 PKC module standard

Document Code	Name of Standard
FIPS PUB 186-4	Digital Signature Standard (DSS) (Compliant with P-256)
SEC 2	Recommended Elliptic Curve Domain Parameters

4.6.4.3.2 Module Overview

The PKC controller module completes the basic underlying modular arithmetic and the FIPS standard 256-point ECC point multiplication in the public key algorithm. The device embeds the PKC module in order to realize hardware acceleration of calculation.

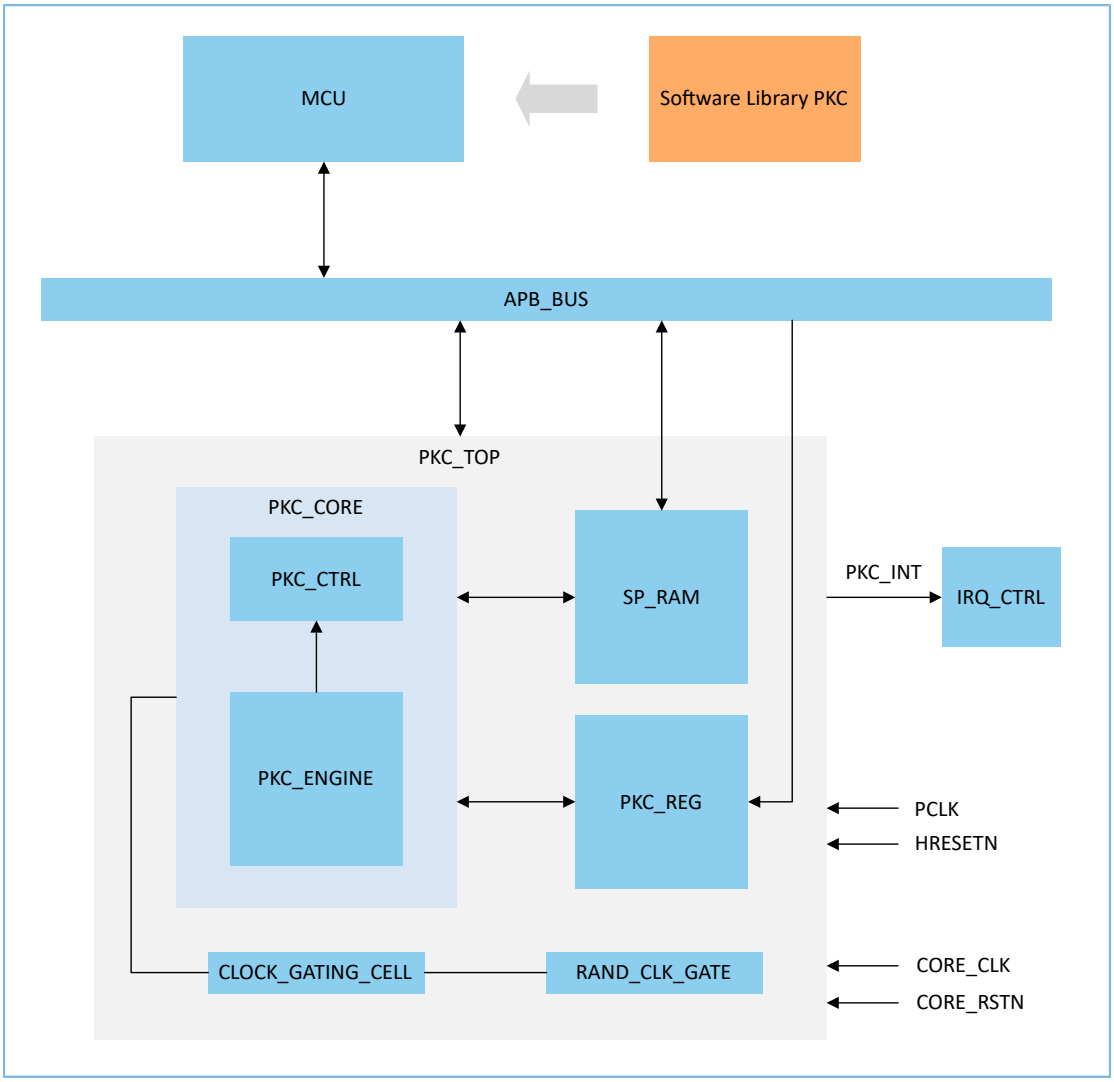

Figure 4-64 PKC module in the system

PKC_TOP includes PKC_CORE, PKC_REG, RAND_CLK_GATE, and SP_RAM. PKC_CORE includes PKC_ENGINE and PKC_CTRL sub modules, in which PKC_ENGINE completes the basic underlying modular operations (modular multiplication, modular inversion, modular addition, modular subtraction, modular comparison, modular shift, big integer multiplication, big integer addition, and dummy multiplication) in the public key algorithm.

The ECC point multiplication only supports P-256 curve, such as secp256r1 and secp256k1. The public key algorithm needs a software library which can be open source or user-defined for implementation.

4.6.4.3.3 Usage

The CTRL register is used to start and reset the PKC module. The STAT register is used to indicate the status of the PKC module. The INT_EN register is used to configure the interrupt mode, and the INT_STAT register is used to indicate the interrupt status.

- **Point Multiplication:** MCU writes the following parameters to SP_RAM that is a memory area of the PKC module. The parameters include the 256-bit random number k, order n, prime P, base point G, R2mod P, and the initial

infinity point Q0. CFG0 to CFG12 registers are used to configure the location pointers. The calculation can be started by using the CTRL register. MCU will read the result from the SP_RAM when the calculation completes. In the interrupt mode, the INT_STAT register can indicate whether the calculation completes.

- **Modular Multiplication:** MCU writes operand a, operand b, and prime p to SP_RAM. The SW_CFG7 register is used to configure the operand word length. SW_CFG0 and SW_CFG1 registers are used to configure the location pointers. The CFG13 register is used to configure the modular multiplication constant if the lowest 32-bit of prime p is not 0xFFFFFFFF. The calculation can be started by using the CTRL register. MCU will read the result from the SP_RAM when the calculation completes. In the interrupt mode, the INT_STAT register can indicate whether the calculation completes.
- **Modular Inversion:** MCU writes operand a, modulo p, x1 initial value 1, and x2 initial value 0 to SP_RAM. The SW_CFG7 register is used to configure the operand word length. SW_CFG4 and SW_CFG5 registers are used to configure the location pointers. The SW_CFG6 register is configured to save the intermediate result of modular inversion operation. The calculation can be started by using the CTRL register. MCU will read the modular inversion x1 from the SP_RAM and read the SW_CFG8 register to get the modular inversion k. In the interrupt mode, the INT_STAT register can indicate whether the calculation completes.
- **Modular Subtraction and Modular Addition:** MCU writes operand a, operand b, and prime p to SP_RAM. The SW_CFG7 register is used to configure the operand word length. SW_CFG2 and SW_CFG3 registers are used to configure the location pointers. The calculation can be started with the CTRL register. MCU will read the result from the SP_RAM when the calculation completes. In the interrupt mode, the INT_STAT register can indicate whether the calculation completes.
- **Modular Comparison and Modular Shift:** MCU writes operand a and prime p to SP_RAM. The SW_CFG7 register is used to configure the operand word length. SW_CFG0 and SW_CFG1 registers are used to configure the location pointers. The calculation can be started by using the CTRL register. MCU will read the result from the SP_RAM when the calculation completes. In the interrupt mode, the INT_STAT register can indicate whether the calculation completes.
- **Big Integer Multiplication:** MCU writes operand a and operand b to SP_RAM. The SW_CFG7 register is used to configure the operand word length. The maximum configurable length of the big number multiplication is 1024 bits. SW_CFG10 and SW_CFG11 registers are used to configure the location pointers. The calculation can be started by using the CTRL register. MCU will read the result from the SP_RAM when the calculation completes. In the interrupt mode, the INT_STAT register can indicate whether the calculation completes.
- **Big Integer Addition:** MCU writes operand a and operand b to SP_RAM. The SW_CFG7 register is used to configure the operand word length. SW_CFG11 and SW_CFG12 registers are used to configure the location pointers. The calculation can be started by using the CTRL register. MCU will read the result from the SP_RAM when the calculation completes. In the interrupt mode, the INT_STAT register can indicate whether the calculation completes.
- **Dummy Multiplication:** The SW_CFG9 register is used to configure the random seed of dummy multiplication. The dummy multiplication can be started by using the CTRL register. Subsequently, modular addition or big integer addition is performed. The result of the modular addition or the big integer addition can be obtained.

- **Random Clock Gating:** The SW_CFG13 register is used to configure the random seed of random clock gating. The random clock gating can be enabled by using the CTRL register. Subsequently, PKC operation can be performed.

4.6.4.4 Registers

4.6.4.4.1 CTRL

- **Name:** PKC Controller Register
- **Description:** This register acts as a global enable/disable for PKC.
- **Base Address:** 0x40014000
- **Offset:** 0x0
- **Reset Value:** 0x00000000

Table 4-350 PKC Controller Register

Bits	Field Name	RW	Reset	Description
31:9	RSVD	R		Reserved bits
8	RST	RW	0x0	Write 0 and then write 1 to this bit to force the PKC core to reset.
7:5	RSVD	R		Reserved bits
4	SW_CTRL	RW	0x0	Only used in software mode. **Value:** • 0x0: Disable software controller. • 0x1: Enable software controller.
3:2	RSVD	R		Reserved bits
1	START	RW	0x0	Only used in hardware mode. After MCU configures all parameters; write 0 and then write 1 to this bit, and the PKC starts to work.
0	EN	RW	0x0	PKC core enable bit **Value:** • 0x0: Disable PKC. • 0x1: Enable PKC.

4.6.4.4.2 CFG0

- **Name:** PKC Configuration 0 Register
- **Description:** This register acts as a configuration for PKC.
- **Base Address:** 0x40014000

- **Offset:** 0x4
- **Reset Value:** 0x00100000

Table 4-351 PKC Configuration 0 Register

Bits	Field Name	RW	Reset	Description
31:25	RSVD	R		Reserved bits
24:16	R_POINT	RW	0x10	R point in sp_ram
15:9	RSVD	R		Reserved bits
8:0	K_POINT	RW	0x0	K point in sp_ram

4.6.4.4.3 CFG1

- **Name:** PKC Configuration 1 Register
- **Description:** This register acts as a configuration for PKC.
- **Base Address:** 0x40014000
- **Offset:** 0x8
- **Reset Value:** 0x00280020

Table 4-352 PKC Configuration 1 Register

Bits	Field Name	RW	Reset	Description
31:25	RSVD	R		Reserved bits
24:16	R2_POINT	RW	0x28	R^2 point in sp_ram
15:9	RSVD	R		Reserved bits
8:0	P_POINT	RW	0x20	P point in sp_ram

4.6.4.4.4 CFG2

- **Name:** PKC Configuration 2 Register
- **Description:** This register acts as a configuration for PKC.
- **Base Address:** 0x40014000
- **Offset:** 0xC
- **Reset Value:** 0x00380030

Table 4-353 PKC Configuration 2 Register

Bits	Field Name	RW	Reset	Description
31:25	RSVD	R		Reserved bits
24:16	GY_POINT	RW	0x38	Gy point in sp_ram

Bits	Field Name	RW	Reset	Description
15:9	RSVD	R		Reserved bits
8:0	GX_POINT	RW	0x30	Gx point in sp_ram

4.6.4.4.5 CFG3

- **Name:** PKC Configuration 3 Register
- **Description:** This register acts as a configuration for PKC.
- **Base Address:** 0x40014000
- **Offset:** 0x10
- **Reset Value:** 0x00480040

Table 4-354 PKC Configuration 3 Register

Bits	Field Name	RW	Reset	Description
31:25	RSVD	R		Reserved bits
24:16	R0X_POINT	RW	0x48	R0x point in sp_ram
15:9	RSVD	R		Reserved bits
8:0	GZ_POINT	RW	0x40	Gz point in sp_ram

4.6.4.4.6 CFG4

- **Name:** PKC Configuration 4 Register
- **Description:** This register acts as a configuration for PKC.
- **Base Address:** 0x40014000
- **Offset:** 0x14
- **Reset Value:** 0x00580050

Table 4-355 PKC Configuration 4 Register

Bits	Field Name	RW	Reset	Description
31:25	RSVD	R		Reserved bits
24:16	R0Z_POINT	RW	0x58	R0z point in sp_ram
15:9	RSVD	R		Reserved bits
8:0	R0Y_POINT	RW	0x50	R0y point in sp_ram

4.6.4.4.7 CFG5

- **Name:** PKC Configuration 5 Register

- **Description:** This register acts as a configuration for PKC.
- **Base Address:** 0x40014000
- **Offset:** 0x18
- **Reset Value:** 0x00680060

Table 4-356 PKC Configuration 5 Register

Bits	Field Name	RW	Reset	Description
31:25	RSVD	R		Reserved bits
24:16	R1Y_POINT	RW	0x68	R1y point in sp_ram
15:9	RSVD	R		Reserved bits
8:0	R1X_POINT	RW	0x60	R1x point in sp_ram

4.6.4.4.8 CFG6

- **Name:** PKC Configuration 6 Register
- **Description:** This register acts as a configuration for PKC.
- **Base Address:** 0x40014000
- **Offset:** 0x1C
- **Reset Value:** 0x00780070

Table 4-357 PKC Configuration 6 Register

Bits	Field Name	RW	Reset	Description
31:25	RSVD	R		Reserved bits
24:16	TEMP1_POINT	RW	0x78	Tmp1 point in sp_ram
15:9	RSVD	R		Reserved bits
8:0	R1Z_POINT	RW	0x70	R1Z point in sp_ram

4.6.4.4.9 CFG7

- **Name:** PKC Configuration 7 Register
- **Description:** This register acts as a configuration for PKC.
- **Base Address:** 0x40014000
- **Offset:** 0x20
- **Reset Value:** 0x00880080

Table 4-358 PKC Configuration 7 Register

Bits	Field Name	RW	Reset	Description
31:25	RSVD	R		Reserved bits
24:16	TEMP3_POINT	RW	0x88	TMP3 point in sp_ram
15:9	RSVD	R		Reserved bits
8:0	TEMP2_POINT	RW	0x80	TMP2 point in sp_ram

4.6.4.4.10 CFG8

- **Name:** PKC Configuration 8 Register
- **Description:** This register acts as a configuration for PKC.
- **Base Address:** 0x40014000
- **Offset:** 0x24
- **Reset Value:** 0x00980090

Table 4-359 PKC Configuration 8 Register

Bits	Field Name	RW	Reset	Description
31:25	RSVD	R		Reserved bits
24:16	TEMP5_POINT	RW	0x98	TMP5 point in sp_ram
15:9	RSVD	R		Reserved bits
8:0	TEMP4_POINT	RW	0x90	TMP4 point in sp_ram

4.6.4.4.11 CFG9

- **Name:** PKC Configuration 9 Register
- **Description:** This register acts as a configuration for PKC.
- **Base Address:** 0x40014000
- **Offset:** 0x28
- **Reset Value:** 0x00A800A0

Table 4-360 PKC Configuration 9 Register

Bits	Field Name	RW	Reset	Description
31:25	RSVD	R		Reserved bits
24:16	CONT1_POINT	RW	0xA8	Constant 1 point in sp_ram
15:9	RSVD	R		Reserved bits
8:0	TEMP6_POINT	RW	0xA0	TMP6 point in sp_ram

4.6.4.4.12 CFG10

- **Name:** PKC Configuration 10 Register
- **Description:** This register acts as a configuration for PKC.
- **Base Address:** 0x40014000
- **Offset:** 0x2C
- **Reset Value:** 0x00B800B0

Table 4-361 PKC Configuration 10 Register

Bits	Field Name	RW	Reset	Description
31:25	RSVD	R		Reserved bits
24:16	X2_POINT	RW	0xB8	X2 point in sp_ram
15:9	RSVD	R		Reserved bits
8:0	X1_POINT	RW	0xB0	X1 point in sp_ram

4.6.4.4.13 CFG11

- **Name:** PKC Configuration 11 Register
- **Description:** This register acts as a configuration for PKC.
- **Base Address:** 0x40014000
- **Offset:** 0x30
- **Reset Value:** 0x00C800C0

Table 4-362 PKC Configuration 11 Register

Bits	Field Name	RW	Reset	Description
31:25	RSVD	R		Reserved bits
24:16	KT_POINT	RW	0xC8	Software computes 2^256 mod p and writes to sp_ram.
15:9	RSVD	R		Reserved bits
8:0	MIT_POINT	RW	0xC0	Only used in hardware mode. MI point in sp_ram

4.6.4.4.14 CFG12

- **Name:** PKC Configuration 12 Register
- **Description:** This register acts as a configuration for PKC.
- **Base Address:** 0x40014000
- **Offset:** 0x34

- **Reset Value:** 0x00D800D0

Table 4-363 PKC Configuration 12 Register

Bits	Field Name	RW	Reset	Description
31:25	RSVD	R		Reserved bits
24:16	B_POINT	RW	0xD8	Curve parameter b in Montgomery field
15:9	RSVD	R		Reserved bits
8:0	A_POINT	RW	0xD0	Curve parameter a in Montgomery field

4.6.4.4.15 CFG13

- **Name:** PKC Configuration 13 Register
- **Description:** This register acts as a configuration for PKC.
- **Base Address:** 0x40014000
- **Offset:** 0x38
- **Reset Value:** 0x00000001

Table 4-364 PKC Configuration 13 Register

Bits	Field Name	RW	Reset	Description
31:0	CONSTQ	RW	0x1	Constant used in Montgomery multiplication

4.6.4.4.16 SW_CTRL

- **Name:** PKC Software Controller Register
- **Description:** This register acts as a controller for software operation.
- **Base Address:** 0x40014000
- **Offset:** 0x40
- **Reset Value:** 0x00000000

Table 4-365 PKC Software Controller Register

Bits	Field Name	RW	Reset	Description
31:10	RSVD	R		Reserved bits
9	RCG_EN	RW	0x0	Enable random clock gating. **Value:** • 0x0: Disable • 0x1: Enable
8	DM_EN	RW	0x0	Enable dummy multiplication.

Bits	Field Name	RW	Reset	Description
				Value: • 0x0: Disable • 0x1: Enable
7	RSVD	R		Reserved bit
6:4	MODE	RW	0x0	Only used in software mode. **Value:** • 0x0: Montgomery multiplication • 0x1: Mod inversion • 0x2: Mod addition • 0x3: Mod subtraction • 0x4: Mod comparison • 0x5: Mod shift • 0x6: Big integer multiplication • 0x7: Bit integer addition
3:1	RSVD	R		Reserved bits
0	START	RW	0x0	Only used in software mode. MCU writes 0 to this bit and then writes 1 to this bit to start the PKC.

4.6.4.4.17 SW_CFG0

- **Name:** PKC Software Configuration 0 Register
- **Description:** This register acts as a configuration for PKC in software mode.
- **Base Address:** 0x40014000
- **Offset:** 0x44
- **Reset Value:** 0x00080000

Table 4-366 PKC Software Configuration 0 Register

Bits	Field Name	RW	Reset	Description
31:25	RSVD	R		Reserved bits
24:16	MMB_POINT	RW	0x8	Modular multiplication B point in sp_ram
15:9	RSVD	R		Reserved bits
8:0	MMA_POINT	RW	0x0	Modular multiplication A point in sp_ram

4.6.4.4.18 SW_CFG1

- **Name:** PKC Software Configuration 1 Register
- **Description:** This register acts as a configuration for PKC in software mode.
- **Base Address:** 0x40014000
- **Offset:** 0x48
- **Reset Value:** 0x00180010

Table 4-367 PKC Software Configuration 1 Register

Bits	Field Name	RW	Reset	Description
31:25	RSVD	R		Reserved bits
24:16	MMC_POINT	RW	0x18	Modular multiplication C point in sp_ram
15:9	RSVD	R		Reserved bits
8:0	MMP_POINT	RW	0x10	Modular multiplication P point in sp_ram

4.6.4.4.19 SW_CFG2

- **Name:** PKC Software Configuration 2 Register
- **Description:** This register acts as a configuration for PKC in software mode.
- **Base Address:** 0x40014000
- **Offset:** 0x4C
- **Reset Value:** 0x00280020

Table 4-368 PKC Software Configuration 2 Register

Bits	Field Name	RW	Reset	Description
31:25	RSVD	R		Reserved bits
24:16	MASB_POINT	RW	0x28	Modular addition/subtraction B point in sp_ram
15:9	RSVD	R		Reserved bits
8:0	MASA_POINT	RW	0x20	Modular addition/subtraction A point in sp_ram

4.6.4.4.20 SW_CFG3

- **Name:** PKC Software Configuration 3 Register
- **Description:** This register acts as a configuration for PKC in software mode.
- **Base Address:** 0x40014000
- **Offset:** 0x50
- **Reset Value:** 0x00380030

Table 4-369 PKC Software Configuration 3 Register

Bits	Field Name	RW	Reset	Description
31:25	RSVD	R		Reserved bits
24:16	MASC_POINT	RW	0x38	Modular addition/subtraction C point in sp_ram
15:9	RSVD	R		Reserved bits
8:0	MASP_POINT	RW	0x30	Modular addition/subtraction P point in sp_ram

4.6.4.4.21 SW_CFG4

- **Name:** PKC Software Configuration 4 Register
- **Description:** This register acts as a configuration for PKC in software mode.
- **Base Address:** 0x40014000
- **Offset:** 0x54
- **Reset Value:** 0x00480040

Table 4-370 PKC Software Configuration 4 Register

Bits	Field Name	RW	Reset	Description
31:25	RSVD	R		Reserved bits
24:16	MIV_POINT	RW	0x48	Modular inversion V point in sp_ram
15:9	RSVD	R		Reserved bits
8:0	MIU_POINT	RW	0x40	Modular inversion U point in sp_ram

4.6.4.4.22 SW_CFG5

- **Name:** PKC Software Configuration 5 Register
- **Description:** This register acts as a configuration for PKC in software mode.
- **Base Address:** 0x40014000
- **Offset:** 0x58
- **Reset Value:** 0x00580050

Table 4-371 PKC Software Configuration 5 Register

Bits	Field Name	RW	Reset	Description
31:25	RSVD	R		Reserved bits
24:16	MIX2_POINT	RW	0x58	Modular inversion X2 point in sp_ram
15:9	RSVD	R		Reserved bits
8:0	MIT_POINT	RW	0x50	Modular inversion X1 point in sp_ram

4.6.4.4.23 SW_CFG6

- **Name:** PKC Software Configuration 6 Register
- **Description:** This register acts as a configuration for PKC in software mode.
- **Base Address:** 0x40014000
- **Offset:** 0x5C
- **Reset Value:** 0x00000060

Table 4-372 PKC Software Configuration 6 Register

Bits	Field Name	RW	Reset	Description
31:9	RSVD	R		Reserved bits
8:0	MITP	RW	0x60	Modular inversion TEMP point in sp_ram

4.6.4.4.24 SW_CFG7

- **Name:** PKC Software Configuration 7 Register
- **Description:** This register acts as a configuration for PKC in software mode.
- **Base Address:** 0x40014000
- **Offset:** 0x60
- **Reset Value:** 0x00000007

Table 4-373 PKC Software Configuration 7 Register

Bits	Field Name	RW	Reset	Description
31:6	RSVD	R		Reserved bits
5:0	LEN	RW	0x7	Operator word length configuration in software mode: • 0x07: 256 bits • 0x08: 288 bits … • 0x3F: 2048 bits In hardware mode, this register is fixed to 0x07.

4.6.4.4.25 SW_CFG8

- **Name:** PKC Software Configuration 8 Register
- **Description:** This register acts as a configuration for PKC in software mode.
- **Base Address:** 0x40014000

- **Offset:** 0x64
- **Reset Value:** 0x00000000

Table 4-374 PKC Software Configuration 8 Register

Bits	Field Name	RW	Reset	Description
31:13	RSVD	R		Reserved bits
12:0	MIK_OUT	R	0x0	K out in modular inversion operation

4.6.4.4.26 SW_CFG9

- **Name:** PKC Software Configuration 9 Register
- **Description:** This register acts as a configuration for PKC in software mode.
- **Base Address:** 0x40014000
- **Offset:** 0x68
- **Reset Value:** 0xAABBCCDD

Table 4-375 PKC Software Configuration 9 Register

Bits	Field Name	RW	Reset	Description
31:0	RDMS	RW	0xAABBCCDD	Random dummy multiplication seed

4.6.4.4.27 SW_CFG10

- **Name:** PKC Software Configuration 10 Register
- **Description:** This register acts as a configuration for PKC in software mode.
- **Base Address:** 0x40014000
- **Offset:** 0x6C
- **Reset Value:** 0x00780070

Table 4-376 PKC Software Configuration 10 Register

Bits	Field Name	RW	Reset	Description
31:25	RSVD	R		Reserved bits
24:16	BMB_POINT	RW	0x78	Big multiplication B point in sp_ram
15:9	RSVD	R		Reserved bits
8:0	BMA_POINT	RW	0x70	Big multiplication A point in sp_ram

4.6.4.4.28 SW_CFG11

- **Name:** PKC Software Configuration 11 Register

- **Description:** This register acts as a configuration for PKC in software mode.
- **Base Address:** 0x40014000
- **Offset:** 0x70
- **Reset Value:** 0x00880080

Table 4-377 PKC Software Configuration 11 Register

Bits	Field Name	RW	Reset	Description
31:25	RSVD	R		Reserved bits
24:16	BAA_POINT	RW	0x88	Big addition A point in sp_ram
15:9	RSVD	R		Reserved bits
8:0	BMC_POINT	RW	0x80	Big multiplication C point in sp_ram

4.6.4.4.29 SW_CFG12

- **Name:** PKC Software Configuration 12 Register
- **Description:** This register acts as a configuration for PKC in software mode.
- **Base Address:** 0x40014000
- **Offset:** 0x74
- **Reset Value:** 0x00980090

Table 4-378 PKC Software Configuration 12 Register

Bits	Field Name	RW	Reset	Description
31:25	RSVD	R		Reserved bits
24:16	BAC_POINT	RW	0x98	Big addition C point in sp_ram
15:9	RSVD	R		Reserved bits
8:0	BAB_POINT	RW	0x90	Big addition B point in sp_ram

4.6.4.4.30 SW_CFG13

- **Name:** PKC Software Configuration 13 Register
- **Description:** This register acts as a configuration for PKC in software mode.
- **Base Address:** 0x40014000
- **Offset:** 0x78
- **Reset Value:** 0x11223344

Table 4-379 PKC Software Configuration 13 Register

Bits	Field Name	RW	Reset	Description
31:0	RCG_SEED	RW	0x11223344	Random clock gating seed

4.6.4.4.31 INT_STAT

- **Name:** PKC Interrupt Status Register
- **Description:** This is a read-only register used to indicate the interrupt status of PKC.
- **Base Address:** 0x40014000
- **Offset:** 0x80
- **Reset Value:** 0x00000000

Table 4-380 PKC Interrupt Status Register

Bits	Field Name	RW	Reset	Description
31:3	RSVD	R		Reserved bits
2	BIAO_INT_FLAG	R WC	0x0	PKC big integer add overflow flag. Write 1 to clear.
1	ERR_INT_EN	R WC	0x0	PKC error interrupt flag. Write 1 to clear.
0	CPLT_INT_FLAG	R WC	0x0	PKC complete interrupt flag. Write 1 to clear.

4.6.4.4.32 INT_EN

- **Name:** PKC Interrupt Enable Register
- **Description:** This register enables or disables all interrupts generated by the PKC.
- **Base Address:** 0x40014000
- **Offset:** 0x84
- **Reset Value:** 0x00000000

Table 4-381 PKC Interrupt Enable Register

Bits	Field Name	RW	Reset	Description
31:3	RSVD	R		Reserved bits
2	BIAO_INT_EN	RW	0x0	PKC big integer add overflow interrupt enable
1	ERR_INT_EN	RW	0x0	PKC error interrupt enable
0	CPLT_INT_EN	RW	0x0	PKC complete interrupt enable

4.6.4.4.33 STAT

- **Name:** PKC Status Register
- **Description:** This is a read-only register used to indicate the status of PKC.
- **Base Address:** 0x40014000
- **Offset:** 0x88
- **Reset Value:** 0x00000000

Table 4-382 PKC Status Register

Bits	Field Name	RW	Reset	Description
31:1	RSVD	R		Reserved bits
0	BUSY	R	0x0	PKC busy status

4.6.5 True Random Number Generator (TRNG)

4.6.5.1 Introduction

The True Random Number Generator (TRNG) generates random numbers used in encryption and decryption. Post-processing is added to the TRNG module to ensure the quality of random number and amend the bias of TRNG.

The goal of true random data generated by TRNG is to pass the NIST SP 800-22 standard test suite.

4.6.5.2 Main Features

- Supports true and pseudo random number generation.
- Random number quality can pass the NIST SP 800-22 standard test suite.
- Supports multiple post-processing mechanisms, including bit-skipping, bit-counting, and von-Neumann.
- Supports multiple combinations of random numbers.
- Supports both interrupt and query modes.

4.6.5.3 Functional Description

4.6.5.3.1 Standard

Table 4-383 TRNG module standard

Document Code	Name of Standard
NIST SP 800-22	A Statistical Test Suite for Random and Pseudorandom Number Generators for Cryptographic Applications

4.6.5.3.2 Module Overview

The TRNG module supports pseudo random number generation and true random number generation. Pseudo random numbers are generated by a linear feedback shift register (LFSR). True random numbers are generated by using a feedforward ring oscillator (FRO) as the entropy source.

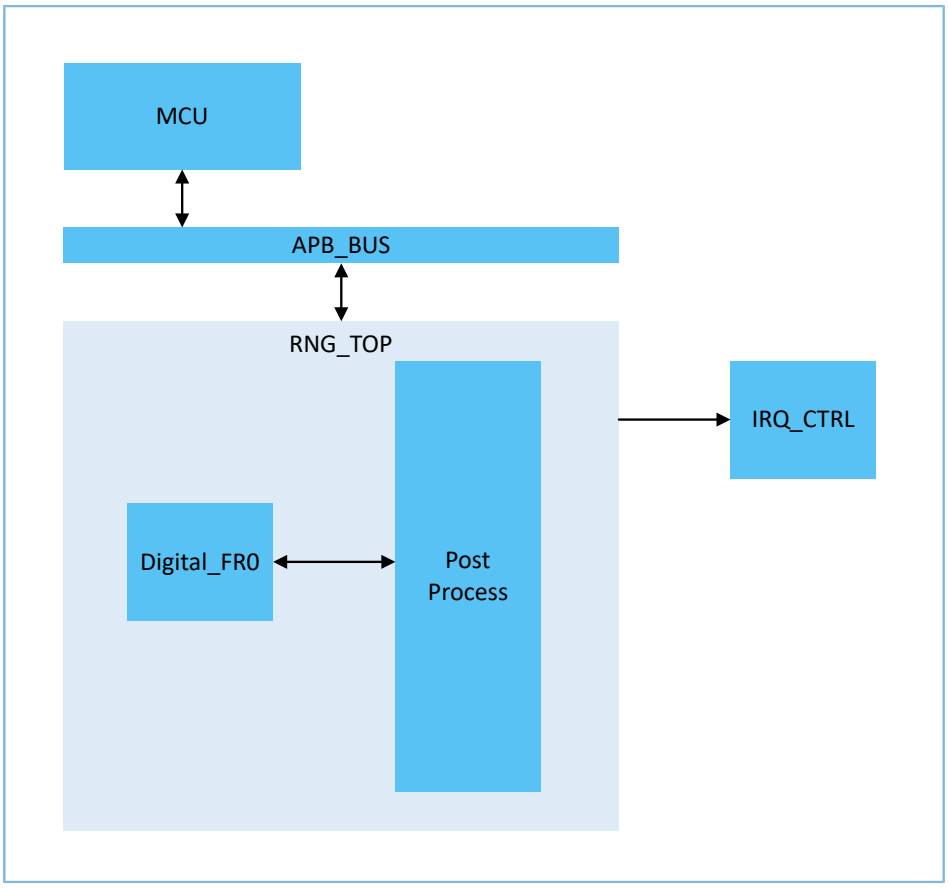

Figure 4-65 RNG module in the system

Digital_FRO is the digital true random number generator, based on the FRO scheme.

Copyright © 2024 Shenzhen Goodix Technology Co., Ltd.

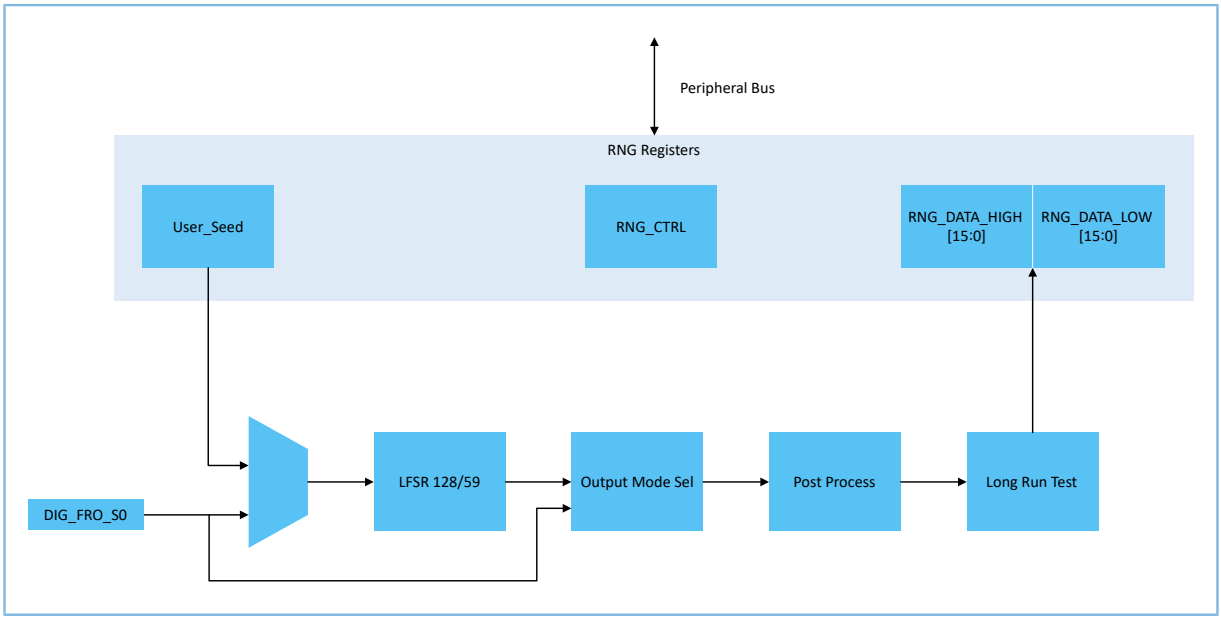

Figure 4-66 RNG post-process module

The work mode of the LFSR can be configured as 128 or 59 polynomial which is implemented by Galois structure. The LFSR seed can be configured from the FRO or in parallel be written in user-configured seed by using the user_seed register.

The TRNG module supports multiple output modes, including LFSR direct output, LFSR XOR TRNG, LFSR and TRNG cyclic sampling, LFSR and TRNG cyclic sampling and parity generation, and the FRO direct output. Since only the FRO is used as the true random number source, the results of LFSR direct output, LFSR XOR TRNG, as well as LFSR and TRNG cyclic sampling will be the same when the LFSR seed is configured by users.

The TRNG module supports multiple post-processing mechanisms, including bit-skipping, bit-counting, and von-Neumann.

4.6.5.3.3 Usage

The CFG register can be used to configure relevant parameters for the TRNG module. CFG.OUT_MODE can be used to configure the output mode. CFG.P_MODE_CFG can be used to configure the post-process mode. CFG.LFSR_MODE_CFG can be used to configure the 59-bit or 128-bit LFSR operation mode. CFG.LFSR_SEED_CFG can be used to configure the LFSR seed source. CFG.HW_INT_EN can set the TRNG module into interrupt mode.

The FRO acts as the entropy source, so that CFG.FRO_EN should be enabled in order to generate true random numbers. It can directly output or be set as the LFSR seed.

The TRNG module also supports user-defined LFSR seeds. In the 59-bit LFSR operation mode, the USER_SEED register needs to be written 4 consecutive times to configure the seed. In the 128-bit LFSR operation mode, the USER_SEED register needs to be written 8 consecutive times to configure the seed. If the output mode is set to LFSR directly output, the random number will be pseudo random number.

The TRNG module can be started by using the CTRL register. The STAT register is used to indicate the current status of the TRNG module. The generated random number can be read out from the DATA register.

4.6.5.4 Registers

4.6.5.4.1 CTRL

- **Name:** TRNG Controller Register
- **Description:** This register acts as a global enable/disable for TRNG.
- **Base Address:** 0x40017800
- **Offset:** 0x0
- **Reset Value:** 0x00000000

Table 4-384 TRNG Controller Register

Bits	Name	Access	Reset Value	Description
31:1	RSVD	R		Reserved bits
0	RUN	RW	0x0	TRNG working enable signal, valid at HIGH • 0x0: Disable TRNG module. • 0x1: TRNG starts working to execute an operation of reading random number.

4.6.5.4.2 STAT

- **Name:** TRNG Status Register
- **Description:** This is a read-only register used to indicate the current status of TRNG.
- **Base Address:** 0x40017800
- **Offset:** 0x4
- **Reset Value:** 0x00000000

Table 4-385 TRNG Status Register

Bits	Name	Access	Reset Value	Description
31:1	RSVD	R		Reserved bits
0	READY	R WC	0x0	TRNG status flag bit, queried by CPU. **Value:** • 0x0: TRNG output data is not valid. • 0x1: TRNG output data is valid. This register is automatically cleared by hardware after READY is set HIGH. This register is automatically cleared by hardware after data reads.

4.6.5.4.3 DATA

- **Name:** TRNG Data Register
- **Description:** This register is a 32-bit read buffer for the random numbers.
- **Base Address:** 0x40017800
- **Offset:** 0x8
- **Reset Value:** 0x00000000

Table 4-386 TRNG Data Register

Bits	Name	Access	Reset Value	Description
31:0	DATA	R	0x0	TRNG data. This register is automatically cleared by hardware after data reads.

4.6.5.4.4 LONG_RUN_STAT

- **Name:** TRNG Long Run Status Register
- **Description:** This is a read-only register used to indicate the current status of long run mode.
- **Base Address:** 0x40017800
- **Offset:** 0x10
- **Reset Value:** 0x00000000

Table 4-387 TRNG Long Run Status Register

Bits	Name	Access	Reset Value	Description
31:9	RSVD	R		Reserved bits
8:1	LONG_RUN_COUNT	R	0x0	Counts of TRNG long run test. This value is incremented by 1 for every long run detected by TRNG. After TRNG is disabled or reset, the register is cleared.
0	LONG_RUN_FLAG	R	0x0	Flag of TRNG long run test **Value:** • 0x0: Long run never occurs. • 0x1: Long run was occurred in random number sequence. After TRNG is disabled or reset, this register is cleared.

4.6.5.4.5 CFG

- **Name:** TRNG Configuration Register
- **Description:** This register acts as a configuration for TRNG.
- **Base Address:** 0x40017800
- **Offset:** 0x14
- **Reset Value:** 0x0000B246

Table 4-388 TRNG Configuration Register

Bits	Name	Access	Reset Value	Description
31:16	RSVD	R		Reserved bits
15	FRO_EN	RW	0x1	Ring oscillator TRNG enable signal **Value:** • 0x0: Disable • 0x1: Enable
14	RSVD	R		Reserved bits
13	HW_INT_EN	RW	0x1	TRNG hardware interrupt enable bit. **Value:** • 0x0: Disable • 0x1: Enable
12:10	LFSR_SEED_CFG	RW	0x4	LFSR seed configuration mode. Select a source for LFSR seed. **Value:** • 0x0: LFSR seed is from the switching current s0. • 0x1: LFSR seed is from the switching current s1. • 0x2: LFSR seed is from the switching current s2. • 0x3: LFSR seed is from the switching current s3. • 0x4: LFSR seed is from the oscillator s0. • 0x5: LFSR seed is from the low frequency sampling s0. • 0x6: LFSR seed is configured by users. • 0x7: Random numbers do not use LFSR and completely use true random mode. Value is limited to 0x4 or 0x6 in the device.
9	LFSR_MODE_CFG	RW	0x1	LFSR configuration mode: • 0x0: 59-bit LFSR • 0x1: 128-bit LFSR
8:7	P_MODE_CFG	RW	0x0	TRNG post-process configuration

Bits	Name	Access	Reset Value	Description
				Value: • 0x0: No post-process • 0x1: Bit skipping • 0x2: Bit counting • 0x3: Von-Neumann
6:4	LFSR_XOR_SEL	RW	0x4	In output mode 8, select TRNG with LFSR XOR. 0x4: LFSR ⊕ Ring Oscillator s0 **Note:** ⊕ means XOR. Value is limited to 0x4 in the device.
3:0	OUT_MODE	RW	0x6	Select a TRNG output mode. **Value:** • 0x4: Digital TRNG direct output, ring oscillator s0 • 0x6: LFSR and TRNG cyclic sampling and parity generation • 0x7: LFSR and TRNG cyclic sampling • 0x8: LFSR ⊕ TRNG • 0x9: LFSR direct output **Note:** ⊕ means XOR. Value is limited to 0x4, 0x6, 0x7, 0x8, or 0x9 in the device.

4.6.5.4.6 SRC_CFG

- **Name:** TRNG Source Configuration Register
- **Description:** This register acts as a configuration for source of TRNG.
- **Base Address:** 0x40017800
- **Offset:** 0x18
- **Reset Value:** 0x00007864

Table 4-389 TRNG Source Configuration Register

Bits	Name	Access	Reset Value	Description
31:15	RSVD	R		Reserved bits
14:11	FRO_CHAIN_SEL	RW	0xF	Selection of Chain4–Chain7 for FRO **Value:**

Bits	Name	Access	Reset Value	Description
				• 0x0: Select long ring. • 0x1: Select short ring.
10:8	RSVD	R		Reserved bits
7:0	WAIT_TIME	RW	0x64	The waiting time during which the TRNG input reaches stable. Default value is 100 time cycles.

4.6.5.4.7 FRO_CFG

- **Name:** TRNG FRO Configuration Register
- **Description:** This register acts as a configuration for FRO.
- **Base Address:** 0x40017800
- **Offset:** 0x1C
- **Reset Value:** 0x0000FFFF

Table 4-390 TRNG FRO Configuration Register

Bits	Name	Access	Reset Value	Description
31:16	RSVD	R		Reserved bits
15:8	TEST_IN	RW	0xFF	Test input signal of each chain for FRO. The default value is 1.
7:0	CHAIN_EN	RW	0xFF	Enable signal of each chain in ring oscillator module. It is valid when FROEN is high.

4.6.5.4.8 USER_SEED

- **Name:** TRNG User Seed Register
- **Description:** This register acts as a user seed value for TRNG.
- **Base Address:** 0x40017800
- **Offset:** 0x20
- **Reset Value:** 0x00000000

Table 4-391 TRNG User Seed Register

Bits	Name	Access	Reset Value	Description
31:16	RSVD	R		Reserved bits
15:0	USER_SEED	W	0x0	RNG seed configured by users.

Bits	Name	Access	Reset Value	Description
				Write four times to user_seed[58:0] to configure a 59-bit random number. Write eight times to user_seed[127:0] to configure a 128-bit random number.

4.6.5.4.9 LONG_RUN_CFG

- **Name:** TRNG Long Run Configuration Register
- **Description:** This register acts as a configuration for long run mode.
- **Base Address:** 0x40017800
- **Offset:** 0x24
- **Reset Value:** 0x00000035

Table 4-392 TRNG Long Run Configuration Register

Bits	Name	Access	Reset Value	Description
31:6	RSVD	R		Reserved bits
5:1	LONG_RUN_THD	RW	0x1A	Threshold configuration of RNG long run test. The run which lasts in a longer or an equal period than this value will be detected. Default value is 26 and the maximum value is 31.
0	THD_LR_TEST	RW	0x1	RNG long run test enable bit **Value:** • 0x0: Disable • 0x1: Enable

4.7 Analog Interfaces

4.7.1 13-Bit SNSADC

4.7.1.1 Introduction

The Sense Analog-to-Digital Converter (SNSADC) is an ADC based on successive approximation registers (SARs). It is an 8-sensing-channel 13-bit 1 Msps ADC that can be used for single ended or for differential measurements. The SNSADC supports DMA to allow for large sample collection without overloading the MCU.

4.7.1.2 Main Features

- 13-bit analog-to-digital conversion

- 8-channel (MSIO0–MSIO7) analog-to-digital conversion
- Up to 16 MHz ADC clock frequency
- Up to 1 Msps conversion rate
- Single-ended or differential measurements
- Internal temperature and Vbattery sensors
- Ability to capture ADC samples using DMA, unburdening the MCU.
- Programmable internal reference

4.7.1.3 Block Diagram

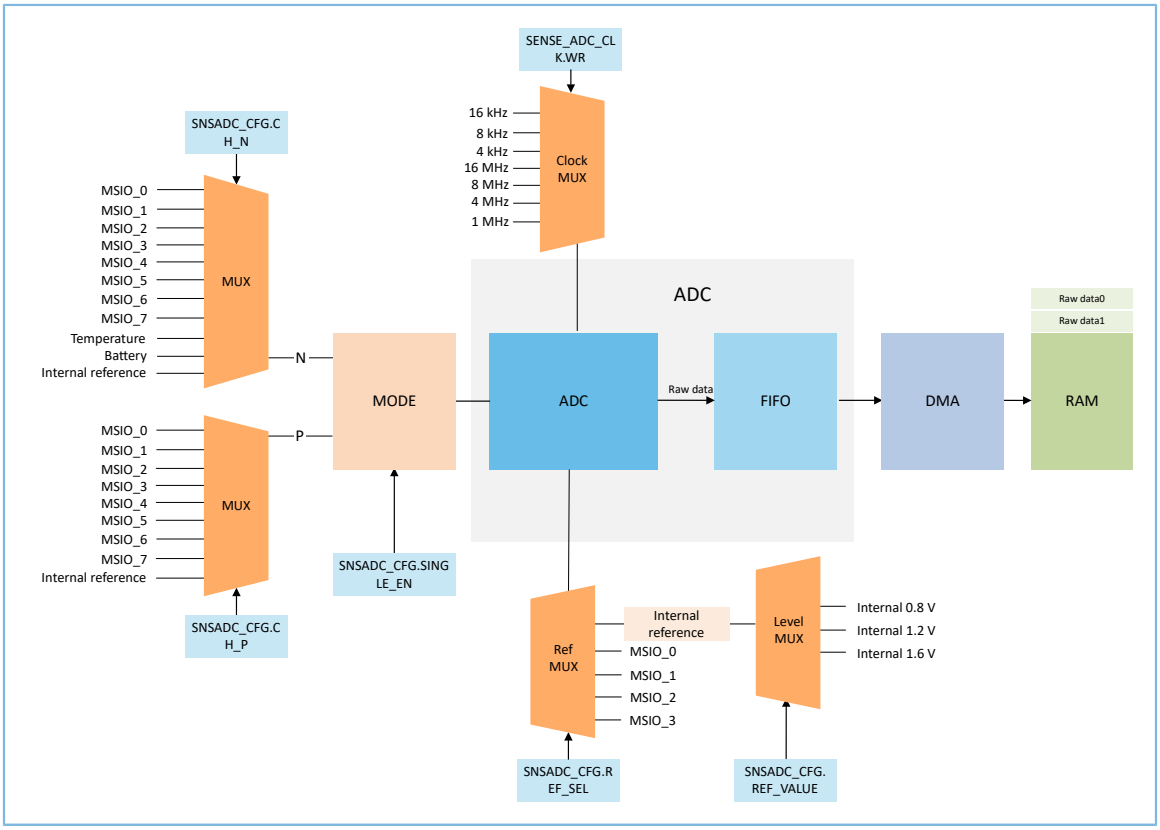

Figure 4-67 SNSADC block diagram

4.7.1.4 Resource Sharing

ADC and Comp can share the input channel from MSIO, but MSIO selected for Comp cannot be used at the same time for ADC sampling. The selection and configuration of I/Os can be found in the Pin Mux table.

4.7.1.5 Functional Description

The SNSADC can sample one of eight analog sources at a given time, using an input mux. Another two of the channel_N are used internally for VBAT and temperature monitoring for adjusting the analog and RF circuitry

configurations for optimum performance. The SNSADC can be reconfigured to perform single-ended or differential measurements through the SNSADC_CFG.SINGLE_EN bit.

The reference voltage can be internally generated or supplied as one of the external inputs and can be selected through the SNSADC_CFG.REF_SEL field. The external reference source can be input through MSIO0–MSIO3, and the input voltage must not exceed half of the supply voltage.

The internal reference voltage can be programmed as 0.8 V, 1.2 V and 1.6 V through SNSADC_CFG.REF_VALUE field. For any specific reference value (Vref), the maximum ADC input is supposed to be approximately 2 x Vref, such as 1.85 x Vref. For example, in the case of Vref = 1.2 V, the maximum ADC input can be 2.25 V, which closely approximates 2.4 V. The minimum ADC input can be 0.

As shown in the following figure, the Iref comes from the relatively constant current generated by the chip bandgap module and will not be affected by other modules; it will generate different Vref levels by resistive voltage division.

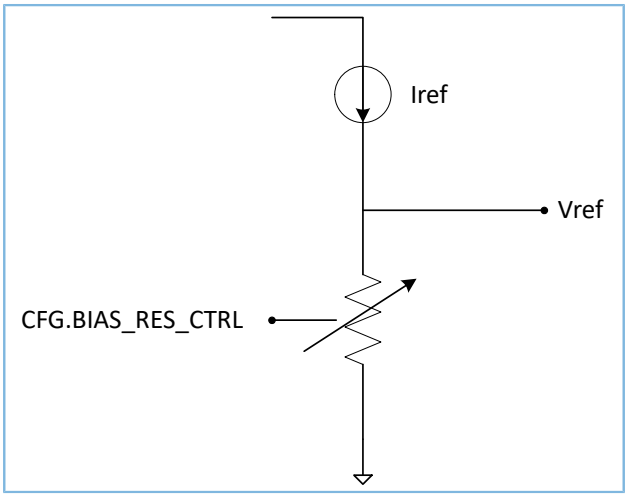

Figure 4-68 Set of ADC internal Vref control level

The ADC sampling clock can be configured as 16 kHz, 8 kHz, 4 kHz, 16 MHz, 8 MHz, 4 MHz, or 1 MHz through the SENSE_ADC_CLK.WR field. The output value of the ADC will be stored into the FIFO, and users can transfer the data to RAM through DMA or MCU.

4.7.1.5.1 Raw Data Stream

- Pairs of successive 13-bit samples from SNSADC are combined into 32-bit words by aligning the 16-bit samples. 32-bit words are stored into an SNSADC FIFO of 64 entries, where each entry is 32-bit long and would hold two samples.
- When the SNSADC FIFO entry count reaches the 6-bit threshold value specified in the SENSE_FF_THRESH register, the samples stored in the FIFO are automatically streamed to the SRAM by using DMA controller. (DMA0 assigned to SNSADC must be programmed accordingly).
- The SNSADC FIFO status (not-empty signal and the entry count) can be polled in the SENSE_ADC_STAT register.
- The SNSADC FIFO can also be “popped” by one entry, by reading the SENSE_ADC_FIFO register.

4.7.1.5.2 Sample Operation Mode

The SNSADC supports two sample operation modes: single-ended measurement mode and differential measurement mode. Users can choose either of the modes by programming SINGLE_EN bit of SNSADC_CFG register.

- In single-ended measurement mode, the input signal should be assigned to channel N and the SNSADC will only measure the input signal of channel N. In this mode, larger voltage input indicates smaller raw data output. The input signal of channel N can be selected as MSIO0–MSIO7, temperature sensor, battery voltage sensor, or internal reference voltage by configuring the SNSADC_CFG.CH_N field. The raw data of the output is the measurement from channel N to GND and the allowed range of channel N is 0 to 2*Vref.
- In differential measurement mode, the input signals should be assigned to channel P and channel N and the SNSADC will measure the input signal of channel P minus channel N. The input signal of channel N is the same with that in single-ended measurement mode, and the input signal of channel P can be selected as MSIO0–MSIO7 or internal reference voltage by configuring the SNSADC_CFG.CH_P field. The raw data of the output is the measurement of difference voltage between channel P to channel N and the allowed range of channel P minus channel N is -2*Vref to 2*Vref.

4.7.1.5.3 Clock and Sampling Rate

- The SNSADC supports both fast sampling and slow sampling. If you need fast sampling, you can select 1M, 4M, 8M, or 16M ADC clock frequency. If you need slow sampling, you can select 16K, 8K, or 4K ADC clock frequency. The SNSADC clock can be selected by configuring the SENSE_ADC_CLK.WR field. The source of the SNSADC clock is shown in the figure below.

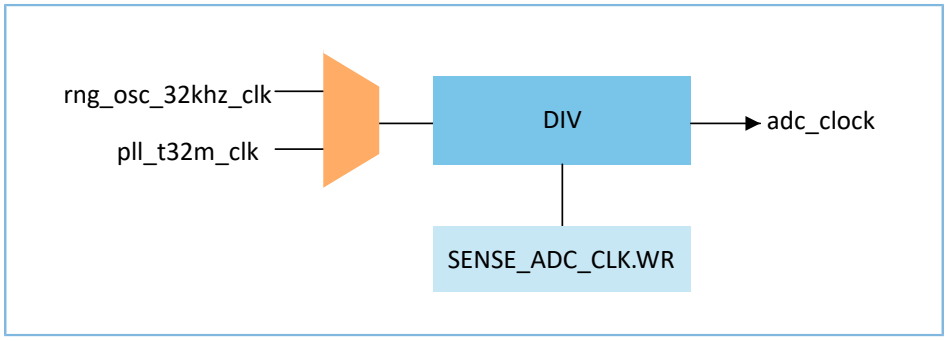

Figure 4-69 Source of SNSADC clock

It takes two clocks for the SNSADC to sample and hold data, and then 13 clocks to convert analog semaphore to 13-bit binary digits, totaling 15 clocks to generate one code. Therefore, the sampling rate of the SNSADC is 1/15 of the ADC clock frequency. For example, a 1 MHz ADC clock has a sampling rate of 66.7 ksps.

The ADC sampling capacitor is 2 pF, which is affected by the RC circuit. If the external impedance is larger, the charging time of the capacitor will be longer. When a higher SNSADC sampling rate is set, it is likely to get abnormal raw data during sampling, because the capacitor is not fully charged. This can be solved by reducing the external impedance or reducing the SNSADC sampling rate.

4.7.1.5.4 Considerations for Using SNSADC

When using an internal reference source, it is necessary for users to ensure the stability of the reference source. The internal reference voltage is divided by the chip power supply voltage, so the reference source voltage of different

gears has certain requirements for the chip power supply voltage. Insufficient chip supply voltage will lead to great ADC measurement error. The chip supply voltage range corresponding to the reference sources of different gears is as follows:

Table 4-393 Chip supply voltage reference range corresponding to different reference source gears

Internal Reference	Chip Supply Voltage
0P8V	2.4 V to 3.8 V
1P2V	2.6 V to 3.8 V
1P6V	3.2 V to 3.8 V

For the first data acquisition after SNSADC initialization, the SNSADC module needs to spend a cycle to clear the previous sampling parameters and another cycle to make the new sampling parameters effective. Therefore, when starting the SNSADC for the first time, the results of the first 2 to 3 pieces of 13-bit data may be unstable. It is recommended to filter out the first 2 to 3 pieces of unstable 13-bit data after SNSADC starts working.

4.7.1.5.5 Measuring MSIO Input

In single-ended mode:

Usually, the original value of 0–8191 corresponds to the voltage of 0–2*Vref, so the current voltage value is calculated by raw_data *lsb, where lsb is 2*Vref/8192. Due to the offset of the ADC itself, this correspondence is not accurate. the conversion relationship between output voltage and raw_data is as follows.

$$\text{raw_data} = cur_voltage * slope + offset$$

The slope and offset of the straight line are obtained by the method of linear fitting, and the current voltage can be converted by substituting the raw data. You need to obtain the slope and offset values during calibration based on different reference voltages. For more detailed information, refer to Section 4.7.1.5.8 Calibration. The calibration values are stored in eFuse.

As the offset and slope have been calibrated based on different reference voltages and stored in eFuse, the corresponding voltage value can be obtained from the raw data using the following formulas:

- In the eFuse, the **slope_int_0p8** and **offset_int_0p8** of the 0.8 V internal reference voltage are obtained by measuring the two input voltages and their corresponding raw data. You can use the following formula to convert the raw data into the current voltage value.

  $$\text{cur_voltage} = (\text{raw_data} - \text{offset_int_0p8})/\text{slope_int_0p8}$$

  Similarly, the eFuse also stores **slope_int_1p2**, **offset_int_1p2** of the 1.2 V internal reference voltage, and **slope_int_1p6**, **offset_int_1p6** of the 1.6 V internal reference voltage.

  When the internal 1.2 V reference voltage is selected, the raw data can be converted into the current voltage value by the following formula:

  $$\text{cur_voltage} = (\text{raw_data} - \text{offset_int_1p2})/\text{slope_int_1p2}$$

  Whenthe internal 1.6 V reference voltage is selected, the raw data can be converted into the current voltage value by the following formula:

$$cur_voltage = (raw_data - offset_int_1p6)/slope_int_1p6$$

- The **slope_ext_1p0** and **offset_ext_1p0** of the external 1.0 V reference source are also stored in eFuse. When the external reference voltage is selected, you only need to pay attention to the scaling of the **slope_ext_1p0**. The conversion formula is as follows:

$$cur_voltage = (raw_data - offset_ext_1p0)/(slope_ext_1p0 * 1.0/external_reference_voltage)$$

The above conversion formulas are used when the single-ended mode is selected.

In differential mode:

Usually, the original value of 8191-0 correspond to the voltage of -2*Vref – 2*Vref, so the slope in differential mode is -2 times of the slope in single-ended mode. In the differential mode, when both input channels are configured with internal reference voltage, i.e. the voltage being sampled is 0, the obtained raw_data is the offset value. Therefore, according to the obtained raw_data value, the calibration value obtained in single-ended mode is used to obtain the sampled voltage of the input.

- When 0.8 V internal reference voltage is selected, the calculation of input voltage values in dual-channel mode is as follows:

$$cur_voltage = (offset - raw_data * 2)/slope_int_0p8$$

  where "offset" refers to "raw_data", which is obtained by setting both input channels to internal reference voltage, and selecting internal reference voltage of 0.8 V.

- When 1.2 V internal reference voltage is selected, the calculation of input voltage values in dual-channel mode is as follows:

$$cur_voltage = (offset - raw_data * 2)/slope_int_1p2$$

  where "offset" refers to "raw_data", which is obtained by setting both input channels to internal reference voltage, and selecting internal reference voltage of 1.2 V.

- When 1.6 V internal reference voltage is selected, the calculation of input voltage values in dual-channel mode is as follows:

$$cur_voltage = (offset - raw_data * 2)/slope_int_1p6$$

  where "offset" refers to "raw_data", which is obtained by setting both input channels to internal reference voltage, and selecting internal reference voltage of 1.6 V.

- When 1.0 V external reference voltage is selected, the calculation of input voltage values in dual-channel mode is as follows:

$$cur_voltage = (offset - raw_data * 2)/(slope_ext_1p0 * 1.0/external_reference_voltage)$$

  where "offset" refers to "raw_data", which is obtained by setting both input channels to internal reference voltage, and selecting external reference voltage of 1.0 V.

4.7.1.5.6 Measuring Battery Voltage

There is a voltage divider resistor inside the SoC to divide the voltage of the battery as VBAT/n. Users need to configure the SNSADC_CFG.CH_N field as 0xE (battery sensor as input), and select the internal reference voltage of 0.8 V.

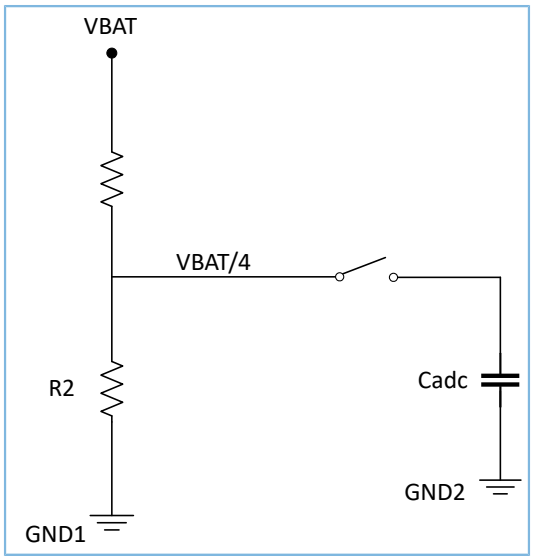

Figure 4-70 Principle of VBAT measurement

Therefore, users can obtain the raw data of the current voltage, and then convert the raw data into the actual battery voltage value through the following formula:

$$\text{bat_voltage} = [(\text{raw_data} - \text{offset_int_0p8})/\text{slope_int_0p8}] * \text{n}$$

Where, slope_int_0p8 and offset_int_0p8 are the slope and calibration offset when 0.8 V internal reference voltage is selected; n represents the divider ratio of the resistor to the battery voltage. The typical value of n is 3.857.

4.7.1.5.7 Measuring Internal Temperature

The device embeds a temperature sensor module, and the measurement range is from -40°C to 85°C. The temperature sensor converts the internal temperature of the chip into a voltage value as Vtemp and outputs it to the ADC for sample and conversion. After temperature calibration, the ADC converts the raw data into an actual temperature value.

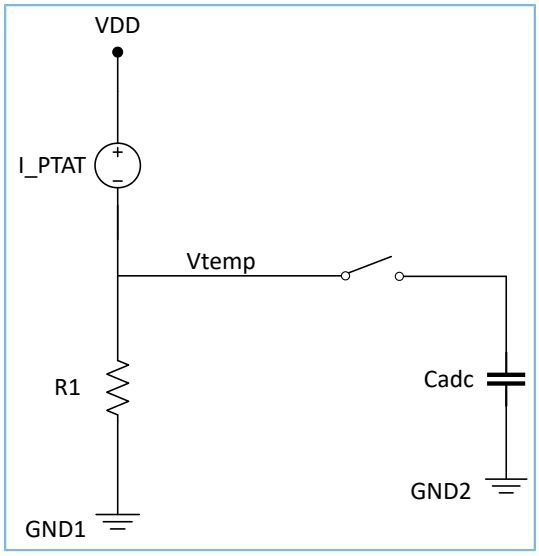

Figure 4-71 Principle of temperature measurement

If users want to measure the internal temperature of the chip, it is necessary to configure the SNSADC_CFG.CH_N field as 0xD (temperature sensor as input), and select the internal reference voltage of 0.8 V. Given **temperature2–temperature1 = [(raw_data2–rawdata1)/slope]**, the calibration formula is as follows:

$$\text{temperature} = \left[(\text{raw_data} - \text{temp_offset})/(\text{slope_int_0p8} * 2)/(-0.00175)\right] + \text{temp_ref}$$

Where, temp_ref represents an ambient temperature during calibration; temp_offset represents ADC raw data at temp_ref ambient temperature during calibration; slope_int_0p8 represents the calibrated slope value of the internal 0.8 V reference voltage; -0.00175 represents the amplification factor for the slope when used to measure temperature. In general, the output voltage decreases by a typical value of 1.863 mV for every 1 degree increase in temperature. The typical value of the voltage is 807.5 mV at -40°C and 583.9 mV at 85°C.

4.7.1.5.8 Calibration

Although some calibration values are provided to be stored in the eFuse at 3.3 V supply and normal temperature, these calibration values may have errors at other ambient temperatures. Users can use the following methods to recalibrate the conversion relationship between raw data and actual temperature or voltage.

- Single-ended mode calibration with internal reference source
  1. Select internal reference source by configuring the SNSADC_CFG.REF_SEL field, select internal reference level by configuring the SNSADC_CFG.REF_VALUE field, and select single-ended mode by configuring the SNSADC_CFG. SINGLE_EN bit to initialize ADC.
  2. Usea noise-free power supply to connect to the N terminal by configuring the SNSADC_CFG.CH_N field to start ADC sampling.
  3. Sample 1024 pieces of data and remove the maximum and minimum values to obtain the average of 1022 numbers.
  4. At this time, the average value is recorded as x0, and the input voltage is recorded as y0.

5. Collect another voltage value using the above method, record the average code value as x1, and record the input voltage value as y1.
6. Use the formula **y = (x-offset)/slope**, and bring in x0, y0, x1, y1 to obtain the offset and slope.
7. The raw data can be converted into the actual voltage using the following formula:

$$\text{actual_voltage} = (\text{raw_data} - \text{offset})/\text{slope}$$

- Single-ended mode calibration with external reference source

1. Select external reference source by configuring the SNSADC_CFG.REF_SEL field, and select single-ended mode by configuring the SNSADC_CFG. SINGLE_EN bit to initialize ADC.
2. Use a noise-free power supply to connect to the N terminal by configuring the SNSADC_CFG.CH_N field to start ADC sampling.
3. Use a noise-free power supply to connect to the ADC external input pin, and record the current external input as z (note that the external input is only connected from MSIO_0–MSIO_3).
4. Sample 1024 pieces of data and remove the maximum and minimum values to obtain the average of 1022 numbers.
5. At this time, the average value is recorded as x0, and the input voltage is recorded as y0.
6. Collect another voltage value using the above method, record the average code value as x1, and record the input voltage value as y1.
7. Use the formula **y = (x-offset)/slope**, and bring in x0, y0, x1, y1 to obtain the offset and slope.
8. The raw_data can be converted into the actual voltage using the following formula:

$$\text{actual_voltage} = (\text{raw_data} - \text{offset})/(\text{slope} * 1.0\text{f})/\text{external_reference_voltage}$$

where external_refernce_voltage represents the voltage value of the external input as this time.

- Temperature sensor calibration

1. Select internal reference source by configuring the SNSADC_CFG.REF_SEL field, and select internal 0.8 V reference source by configuring the SNSADC_CFG.REF_VALUE field.
2. Select single-ended mode by configuring the SNSADC_CFG. SINGLE_EN bit, and select temperature as N-terminal input by configuring the SNSADC_CFG.CH_N field to initialize ADC.
3. Put the chip into a constant temperature environment.
4. Sample 1024 pieces of data and remove the maximum and minimum values to obtain the average of 1022 numbers.
5. At this time, the average value is recorded as , and the current temperature is recorded as temp_ref.
6. Given **temperature2–temperature1 = [(raw_data2–rawdata1)/slope]**, and the **slope = (slope_int_0p8 * 2)/(–0.00175)** , the raw data can be converted into the actual temperature using the following formula:

$$\text{actual_temperature} = \left[(\text{raw_data} - \text{temp_offset})/(\text{slope_int_0p8} * 2)/(-0.00175)\right] + \text{temp_ref}$$

4.7.1.6 Registers

4.7.1.6.1 SNSADC_CFG

- **Name:** Sense ADC Configuration Register
- **Description:** This register is used to configure reference, channels, and input mode of ADC.
- **Base Address:** 0x4000A800
- **Offset:** 0x00
- **Reset Value:** 0x0710070A

Bits	Field Name	RW	Reset	Description
31	MAS_RST	RW	0x0	Reset SNSADC master. **Value:** • 0x0: None • 0x1: Reset
30	EN	RW	0x0	Enable SNSADC. **Value:** • 0x0: Disable SNSADC • 0x1: Enable SNSADC
29:27	REF_SEL	RW	0x0	Reference voltage select **Value:** • 0x0: Buffered internal reference • 0x1: Reserved • 0x2: Reserved • 0x3: Using MSIO0 as a reference • 0x4: Using MSIO1 as a reference • 0x5: Using MSIO2 as a reference • 0x6: Using MSIO3 as a reference • 0x7: Reserved **Note**: Because the maximum input for any MSIO is supposed to be VBAT, there is no benefit from using an external input higher than VBAT/2.
26:24	REF_HP	RW	0x7	Used to define the current in the reference circuit: 0x0: 100 ksps

Bits	Field Name	RW	Reset	Description
				0x7: 1 Msps
23:20	CH_P	RW	0x1	Used to define input for channel P: **Value:** • 0x0–0x7: MSIO0–MSIO7 as input • 0xF: Internal reference as input
19:16	CH_N	RW	0x0	Used to define input for channel N: **Value:** • 0x0–0x7: MSIO0–MSIO7 as input • 0xD: Temperature sensor as input • 0xE: Battery sensor as input • 0xF: Internal reference as input
15	TEMP_EN	RW	0x0	Enable temperature sensor. **Value:** • 0x0: Disable • 0x1: Enable
14	VBAT_EN	RW	0x0	Enable battery sensor. **Value:** • 0x0: Disable • 0x1: Enable
13	SINGLE_EN	RW	0x0	Used to define operation to either single ended or differential mode. **Value:** • 0x0: Differential • 0x1: Single ended
12	OFS_CAL_EN	RW	0x0	Used to swap inputs of comparator for offset calibration. **Value:** • 0x0: Disable • 0x1: Enable
11	RSVD	R		Reserved bit
10:8	DYMAMIC	RW	0x7	Used to define dynamic range of ADC. Default: 0x7; for higher input signal frequencies close to Nyquist rate, use 0x1.
7:4	RSVD	R		Reserved bits
3:0	REF_VALUE	RW	0xA	Buffered internal reference **Value:** • 0x3: 0.8 V

Bits	Field Name	RW	Reset	Description
				• 0x7: 1.2 V • 0xB: 1.6 V **Note**: For any specific reference value (Vref), the maximum ADC input is supposed to be 2 x Vref.

4.7.1.6.2 SENSE_ADC_FIFO

- **Name:** Sense ADC FIFO Output Register
- **Description:** Reading this register will pop an entry from the sense ADC FIFO.
- **Base Address:** 0x4000E000
- **Offset:** 0x00
- **Reset Value:** 0x00000000

Table 4-394 Sense ADC FIFO Output Register

Bits	Field Name	RW	Reset	Description
31:0	DATA	R	0x0	When the register is read, data in the FIFO buffer is accessed.

4.7.1.6.3 SENSE_FF_THRESH

- **Name:** Sense ADC FIFO Control Register
- **Description:** This register is used to enable DMA request and set the threshold at which the DMA request is made to the hardware handshake mechanism.
- **Base Address:** 0x4000E000
- **Offset:** 0x004
- **Reset Value:** 0x00000000

Table 4-395 Sense ADC FIFO Control Register

Bits	Field Name	RW	Reset	Description
31:17	RSVD	R		Reserved bits
16	DMA_EN	RW	0x0	• 1: DMA HW handshake from SNSADC FIFO is enabled • 0: No DMA HW handshake (suitable for polling mode)
15:6	RSVD	R		Reserved bits
5:0	THD	RW	0x0	This is used to set the threshold at which the DMA request is made to the hardware handshake mechanism, should be a power of 2.

4.7.1.6.4 SENSE_ADC_STAT

- **Name:** Sense ADC Status Register
- **Description:** This register contains the number of valid data entries in the ADC FIFO memory.
- **Base Address:** 0x4000E000
- **Offset:** 0x008
- **Reset Value:** 0x00000000

Table 4-396 Sense ADC Status Register

Bits	Field Name	RW	Reset	Description
31:17	RSVD	R		Reserved bits
16	FLUSH	W		Write 1 to flush FIFO DATA.
15:9	RSVD	R		Reserved bits
8	VAL	R	0x0	Identify whether FIFO is empty. **Value:** • 0x0: FIFO is empty. • 0x1: FIFO is not empty.
7	RSVD	R		Reserved bit
6:0	COUNT	R	0x0	Sense ADC FIFO Count, number of 32-bit words in the FIFO. Maximum value is 0x40 or decimal 64.

4.7.1.6.5 SENSE_ADC_CLK

- **Name:** Sense ADC Clock Register
- **Description:** This register selects ADC clock.
- **Base Address:** 0x4000E000
- **Offset:** 0x00C
- **Reset Value:** 0x00000000

Table 4-397 Sense ADC Clock Register

Bits	Field Name	RW	Reset	Description
31:19	RSVD	R		Reserved bits
18:16	RD	R		Actual clock select value
15:3	RSVD	R		Reserved bits
2:0	WR	RW	0x0	SNSADC clock select **Value (Hz):** • 0: No clock

Bits	Field Name	RW	Reset	Description
				• 1: 16 kHz • 2: 8 kHz • 3: 4 kHz • 4: 16 MHz • 5: 8 MHz • 6: 4 MHz • 7: 1 MHz

4.7.1.6.6 SENSE_ADC_GET_TKN_HW

- **Name:** Sense ADC Get Hardware Token Register
- **Description:** Read this register to try to get token for Hardware and return the result of the trial.
- **Base Address:** 0x4000E000
- **Offset:** 0x014
- **Reset Value:** 0x00000000

Table 4-398 Sense ADC Get Hardware Token Register

Bits	Field Name	RW	Reset	Description
31:9	RSVD	R		Reserved bits
8	HW_OWNER	R	0x0	• 0: Hardware has the token. • 1: Software has the token.
7:1	RSVD	R		Reserved bits
0	HW_LOCKED	R	0x0	• 0: Token available • 1: Token locked

4.7.1.6.7 SENSE_ADC_GET_TKN_SW

- **Name:** Sense ADC Get Software Token Register
- **Description:** Read this register to try to get token for software and return the result of the trial.
- **Base Address:** 0x4000E000
- **Offset:** 0x018
- **Reset Value:** 0x00000000

Table 4-399 Sense ADC Get Software Token Register

Bits	Field Name	RW	Reset	Description
31:9	RSVD	R		Reserved bits

Bits	Field Name	RW	Reset	Description
8	SW_OWNER	R	0x0	0: Hardware has the token. 1: Software has the token.
7:1	RSVD	R		Reserved bits
0	SW_LOCKED	R	0x0	0: Token available 1: Token locked

4.7.1.6.8 SENSE_ADC_RET_TKN_HW

- **Name:** Sense ADC Release Hardware Token Register
- **Description:** Write any value to this register to release the Hardware token.
- **Base Address:** 0x4000E000
- **Offset:** 0x020
- **Reset Value:** 0x00000000

Table 4-400 Sense ADC Release Hardware Token Register

Bits	Field Name	RW	Reset	Description
31:1	RSVD	R		Reserved bits
0	HW_RELEASE	W	0x0	Write any value to this bit to release the HW token.

4.7.1.6.9 SENSE_ADC_RET_TKN_SW

- **Name:** Sense ADC Release Software Token Register
- **Description:** Write any value to this register to release the software token.
- **Base Address:** 0x4000E000
- **Offset:** 0x020
- **Reset Value:** 0x00000000

Table 4-401 Sense ADC Release Software Token Register

Bits	Field Name	RW	Reset	Description
31:1	RSVD	R		Reserved bits
0	SW_RELEASE	W	0x0	Write any value to this bit to release the SW token.

4.7.1.6.10 SENSE_ADC_GET_TKN_STS

- **Name:** Sense ADC Token Status Register
- **Description:** This register indicates the token status and the token owner.

- **Base Address:** 0x4000E000
- **Offset:** 0x024
- **Reset Value:** 0x00000000

Table 4-402 Sense ADC Token Status Register

Bits	Field Name	RW	Reset	Description
31:9	RSVD	R		Reserved bits
8	OWNER	R	0x0	0: Hardware has the token. 1: Software has the token.
7:1	RSVD	R		Reserved bits
0	LOCKED	R	0x0	0: Token available 1: Token locked

Note:

Reading this register would not issue any action of getting a token.

4.7.1.7 Electrical Specifications

All the data in the following table are obtained under the conditons: VBAT=3.3 V and VREF = 1.6 V.

Table 4-403 Electrical specifications

Parameter	Symbol	Min.	Typ.	Max.	Unit
Physical number of bits	PNOB		13		bit
Sampling clock frequency	Freq	4K		16M	Hz
Integrated nonlinearity	INL	-8		8	LSB
Differential nonlinearity	DNL	-8		8	LSB
Signal-to-noise and distortion ratio	SNDR		60		dB
Total harmonic distortion	THD		65		dB
Effective number of bits	ENOB		9.5		bit
Differential offset voltage, 13-bit resolution	Vos	-4		4	LSB
Gain error	Eg	-1		1	%
Sampling capacitor	Cs		2		pF
Sampling rate	F_S	0.27		1067	ksps
Input resistance	Rin		Hi-Z		ohm
Current consumption during ADC conversion @100 Ksps	I_ADC		< 20		µA
Spurious free dynamic range	SFDR		68		dB

4.7.2 Temperature Sensor

4.7.2.1 Introduction

A temperature sensor is integrated in the device to measure the die temperature inside the chip. As one of the analog input signals of SNSADC, temperature signals can be converted into digital signals by using the SNS ADC. Users can obtain the temperature value according to the conversion formula. It also uses temperature sensors to adjust RF parameters inside the chip.

4.7.2.2 Main Features

- Integrated into the SNSADC module and serves as one of the SNSADC input signals.
- Measures the die temperature to check whether it is greater than or equal to the operating temperature of the device.
- Temperature measurement range: –40°C to +85°C.
- The measurement accuracy at normal temperature is 1°C to 2°C, and the maximum error at high or low temperature is 5°C.

4.7.2.3 Functional Description

The temperature sensor converts the temperature signals inside the chip into voltage. The SNSADC module inside the chip can be used to sample the converted voltage signals and then calculate the temperature value through the conversion formula.

4.7.2.3.1 Acquisition Parameter Configuration

In order to sample the voltage signals from the temperature sensor, SNSADC needs to be configured to single-ended mode: Temperature sensor should be set as input source of channel N, and enable temperature sensor by setting the bit of SNSADC_CFG.TEMP_EN. Set SNSADC as internal reference source, and select 0p8 for reference value. A 1 MHz clock is recommended for sampling clock.

4.7.2.3.2 Conversion Formula

After getting some code from the SNSADC, you can average the code and then convert the average code value into an actual temperature value according to the following conversion formula:

$$\text{Temperature value} = [(\text{average_code} - \text{adc_temp})/(\text{adc_slope} * 2)]/(0.00175) + \text{Temp_ref}$$

In the formula above, adc_slope is the slope value corresponding to the reference voltage gear of SNSADC. The slope value corresponds to 0p8 gear. The adc_temp is the offset value after calibration based on normal temperature and the Temp_ref is the environment temperature value when calibrated.

These parameters are the calibration parameters of the chip and are saved in eFuse.

4.7.2.3.3 Measuring Internal Temperature

The device has a built-in temperature sensor module inside, and the measurement range is from -40°C to 85°C. The temperature sensor converts the internal temperature of the chip into a voltage value and outputs it to the ADC for sample and conversion. After temperature calibration, the ADC converts the digital value into an actual temperature value.

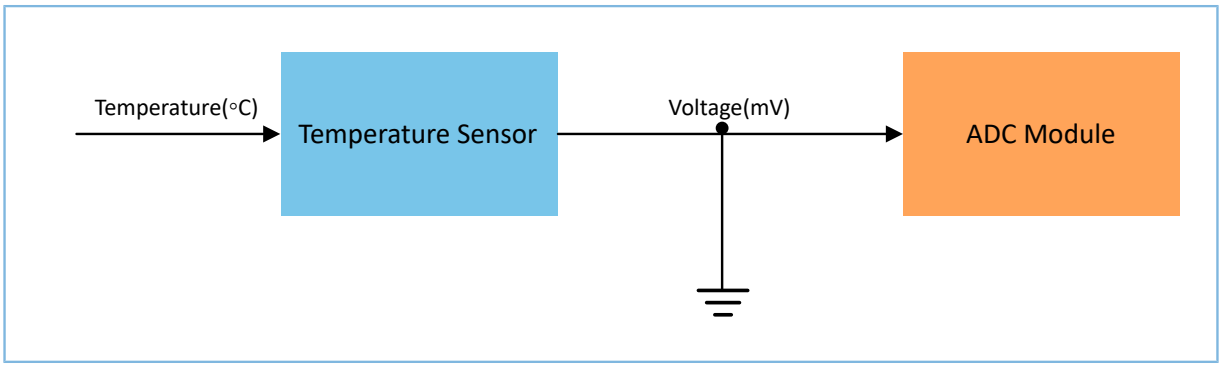

Figure 4-72 Schematic diagram of temperature measurement

If users want to measure the internal temperature of the chip, it is necessary to configure the SNSADC_CFG.CH_N field as as 0xD (temperature sensor as input of channel N), and select the internal reference voltage of 0.8 V.

4.7.2.4 Registers

See the description of register Section 4.7.1.6.1 SNSADC_CFG ((SNSADC Configuration Register) of SNSADC.

4.7.3 Comparator

4.7.3.1 Introduction

The comparator compares two analog signals: V1 (input voltage) and V2 (reference voltage), and generates a rising edge interrupt when V1 exceeds V2 or falling edge interrupt when V1 is lower than V2. Here it is assumed that V1 is connected to the positive side of the comparator, and V2 is connected to the negative side of the comparator. It is worth noting that interrupts will also be triggered when V1 is already lower than V2 before the comparator is enabled, or V2 is reset to a value higher than V1. The interrupt can be used to wake the system up from sleep state. These two signals can be derived from any of the eight MSIOs (MSIO_0–MSIO_7). Additionally, the comparator can be used to compare an external signal with one of the following:

- An internal programmable reference voltage.
- A programmable divided version of VBATL.

4.7.3.2 Main Features

- Input voltage (V1)
  - Internal reference voltage (Vref)
  - VBAT-based reference voltage (Vbat_ref), internal reference based on VBAT
  - Eight MSIOs (MSIO_0–MSIO_7)

- Reference voltage (V2)
  - Internal reference voltage (Vref)
  - VBAT-based reference voltage (Vbat_ref), internal reference based on VBAT
  - Eight MSIOs (MSIO_0–MSIO_7)
- Configurable 26 mV hysteresis
  - No hysteresis
  - –26 mV negative side of hysteresis
  - +26 mV positive side of hysteresis
  - Fully hysteresis 52mV (±26mV)
- Res degeneration is set for calibration.
  - Negative side calibration of comparator
  - Positive side calibration of comparator
- Bias current set (COMP_REG_0.ICOM_CTRL_3_0).
- Comparator channel selection

4.7.3.3 Functional Description

4.7.3.3.1 Overview

The following figure shows the overview of the wakeup comparator module.

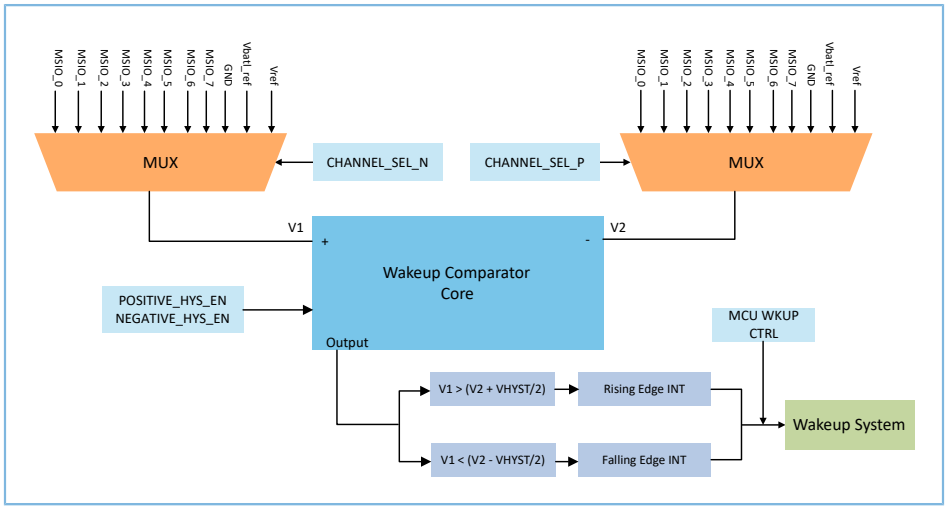

Figure 4-73 COMP overview

Once enabled by configuring theCOMP_REG_0.COMP_EN bit, the comparator starts to work. When the wakeup comparator module is started, V1 starts to be compared with V2 continuously. When V1 exceeds V2, the wakeup interrupt will be triggered. The COMP_REG_0.CHANNEL_SEL_P_3_0 and COMP_REG_0.CHANNEL_SEL_N_3_0 fields

are used to select the input of V1 and V2. When Vbat_ref is selected in the COMP_REG_0.CHANNEL_SEL_N_3_0 or COMP_REG_0.CHANNEL_SEL_P_3_0 field, the bit COMP_REG_1.EN_VBAT_REF should be enabled first. The COMP_REG_0.COMP_BATT_LVL_CTRL_2_0 field is used to select a level value of Vbat_ref.

The following figure shows the settings of VBATT control level. The COMP_BATT_LVL_CTRL_2_0 level value ranges from 0 to 7, and the calculation formula of Vbatt_ref is shown below:

$$\text{Vbatt_ref} = [(\text{Level}+1)/10] * \text{VBAT}$$

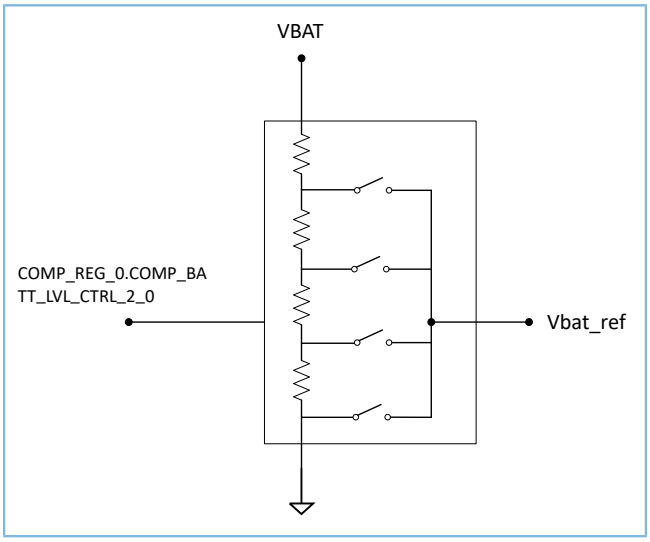

Figure 4-74 Set of Vbat_ref control level

When Vref is selected in the COMP_REG_0.CHANNEL_SEL_P_3_0 or COMP_REG_0.CHANNEL_SEL_N_3_0 field, the bit COMP_REG_1.EN_REF should be enabled first. The COMP_REG_0.COMP_REF_CTRL_7_0 field is used to select a level value of Vref. The settings of Vref control level is showed in the following figure. The COMP_REF_CTRL_7_0 level value ranges from 0 to 255 and can be configured by the COMP_REG_0.COMP_REF_CTRL_7_0 field. The calculation formula of Vref is shown below. Iref comes from the relatively constant current generated by the chip's bandgap module and will not be affected by other modules, as shown in the following figure.

$$\text{Vref} = 9.2\text{mV} * \text{Level} + 4.5\text{mV}$$

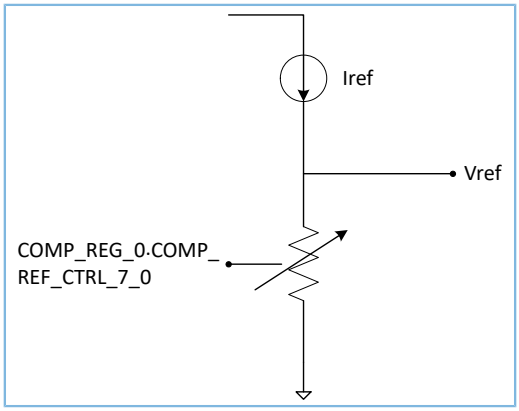

Figure 4-75 Set of Vref control level

Note:

When Vref is selected as reference, to keep the Iref stable, the chip supply voltage (VBAT) must be equal to or higher than 2.5 V. A better suggestion is that the VBAT is equal to or higher than 3 V.

The selected analog pins will be acquired by the comparator once it is enabled.

4.7.3.3.2 Hysteresis

The configurable 52 mV (±26 mV) hysteresis could be added into the wakeup comparator, which helps ensure the slowly changing voltage or a slightly noisy voltage that will cause the comparator not to output to oscillate frequently when the two input voltages are almost close. This could increase the anti-interference ability of the comparator.

The hysteresis mode can be configured in COMP_REG_1.HYS_EN. The hysteresis on both sides mode is illustrated in Figure 4-76. The comparator will trigger a falling edge interrupt whenever V1 becomes lower than V2- (VHYST/2). It will also trigger a rising edge interrupt whenever V1 becomes higher than V2+ (VHYST/2). Here it is assumed that V1 is connected to the positive side of the comparator, and V2 is connected to the negative side of the comparator.

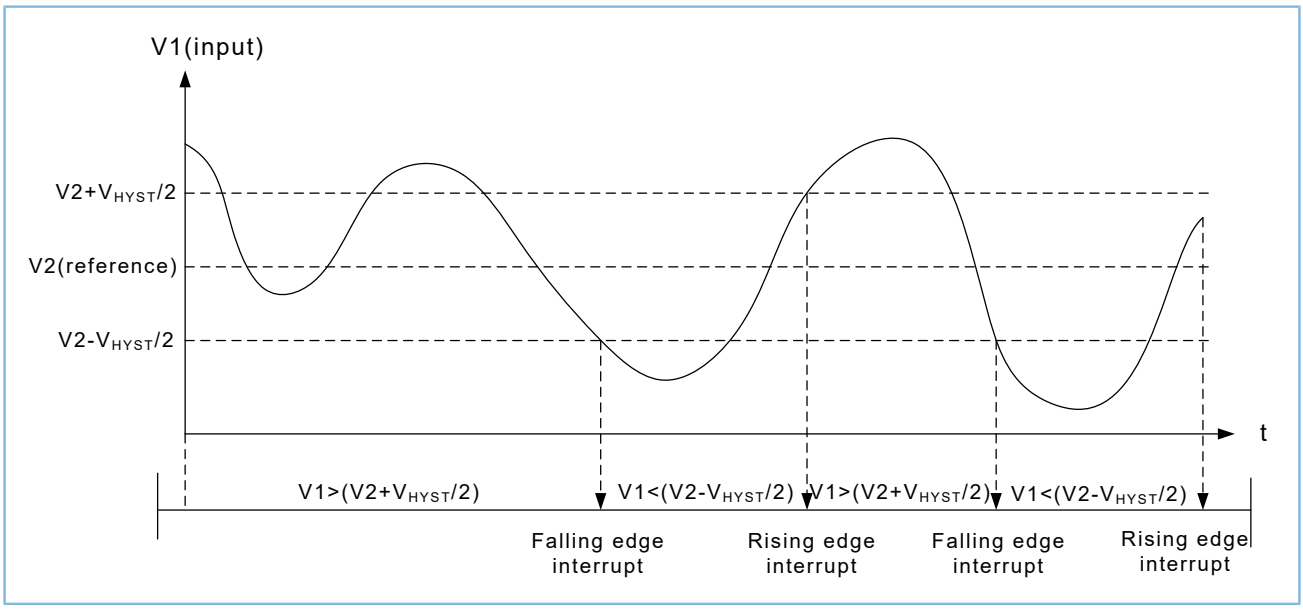

Figure 4-76 Both sides hysteresis enabled in wakeup comparator

4.7.3.3.3 Res Degeneration

Res degeneration is set for calibration. The default values of the COMP_REG_1.RDEG_N_3_0 field and the COMP_REG_1.RDEG_P_3_0 field are 0x8 and are always kept unchanged. The COMP module obtains positive and negative res degeneration which means users can calibrate the comparator by adjusting positive and negative sides. For example, when the comparator outputs a valid signal, the voltage values at N and P sides of the comparator should be approximately equal; if it is found that the actual voltage at P side is too much larger than that at N side, you can set the value of COMP_REG_1.RDEG_P_3_0 smaller, or set the value of COMP_REG_1.RDEG_N_3_0 larger.

4.7.3.3.4 CascRes Half

The COMP_REG_1.CASCRES_HALF bit is used to ensure that the comparator is in proper operation at different bias currents.

- The COMP_REG_1.CASCRES_HALF is set to 1 for half resistor when COMP_REG_0.ICOMP_CTRL_3_0 is close to 0000.
- The COMP_REG_1.CASCRES_HALF is set to 0 for full resistor when COMP_REG_0.ICOMP_CTRL_3_0 is close to 1111.

4.7.3.3.5 Bias Current Set

The COMP_REG_0. ICOMP_CTRL_3_0 field is used to configure the bias current. Higher bias currents will reduce the response time of the comparator, but with a corresponding increase in power consumption. Lower bias currents will increase the response time of the comparator, but with a corresponding reduce in power consumption. It is worth noting that this bias current is also from bandgap module, but this current and Iref are independent of each other and will not interfere with each other.

4.7.3.3.6 Glitch Remove

When the voltage reaches the threshold, the signal from the comp output will jitter because of noise interference. In order to eliminate jitter, users can specify a time by configuring the PMU_COMP_GLITCH_REMOVE.GLITCH_REMOVE_CYCLE field and the signal is considered valid if it does not change within this period of time. As the following figure shows, if the duration of the signal is less than the user set time Glitch_t, then the probability of triggering the interrupt will be greatly reduced. If the duration of the signal is larger than the user set time Glitch_t, then the interrupt will definitely be triggered.

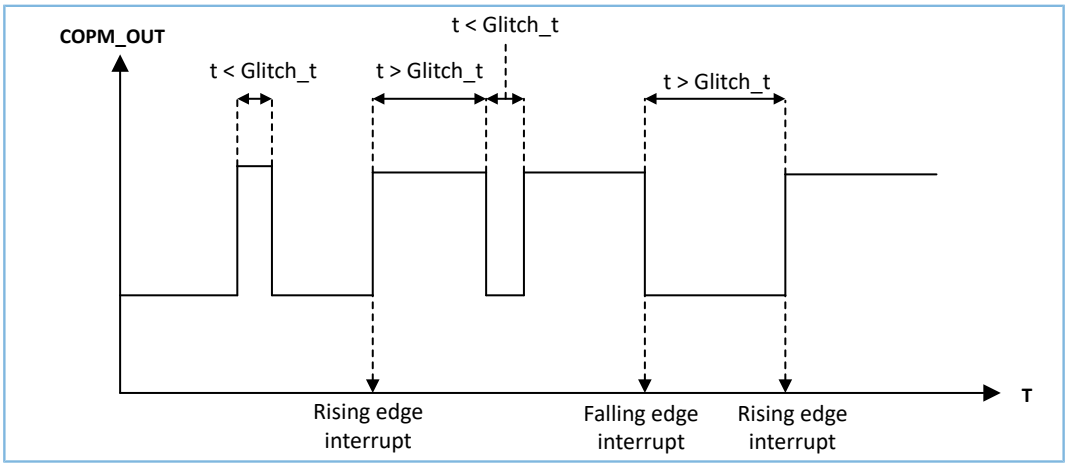

Figure 4-77 Glitch removing of comp

4.7.3.4 Registers

4.7.3.4.1 COMP_REG_0

- **Name:** Comparator Register 0
- **Description:** This register includes REF_VALUE setting, COMP enablement, and channel selection.

- **Base Address:** 0x4000A800
- **Offset:** 0x40
- **Reset Value:** 0x00000000

Table 4-404 Comparator Register 0

Bits	Field Name	RW	Reset	Description
31: 24	RSVD	R		Reserved bits
23:20	CHANNEL_SEL_N_3_0	WR	0x0	Wakeup comparator channel N selection: • 0–7: MSIO_0–MSIO_7 • 8: GND • 9: Vbat_ref • 10: Vref
19:16	CHANNEL_SEL_P_3_0	WR	0x0	Wakeup comparator channel P selection: • 0–7: MSIO_0–MSIO_7 • 8: GND • 9: Vbat_ref • 10: Vref
15:8	COMP_REF_CTRL_7_0	WR	0x0	COMP REF CTRL (Vref) Changes the BG-based reference: Min = 0 V, Max =1.9 V (step = 9 mV). • 11111111: 1.9 V • 00000000: 0 V
7	COMP_EN	WR	0x0	Enables or disables wakeup comparator. • 0: Disable • 1: Enable
6:4	COMP_BATT_LVL_CTRL_2_0	WR	0x0	Changes the Vbat based ref (Vbat_vref): • 000: Vbat/10 • 001: 2 Vbat/10 … • 111 :8 Vbat/10
3:0	ICOMP_CTRL_3_0	WR	0x0	• 0000: Max current (1 µA) • 1111: Min current (50 nA)

4.7.3.4.2 COMP_REG_1

- **Name:** Comparator Register 1

- **Description:** This register includes res degeneration, reference source enable, CascRes Half, and Hysteresis.
- **Base Address:** 0x4000A800
- **Offset:** 0x44
- **Reset Value:** 0x00000088

Table 4-405 Comparator Register 1

Bits	Field Name	RW	Reset	Description
15	EN_REF	WR	0x0	• 1: Enable internal Vref. • 0: Disable internal Vref.
14	EN_VBAT_REF	WR	0x0	• 1: Enable Vbat_ref. • 0: Disable Vbat_ref.
13	CASCRES_HALF	WR	0x0	CascRes Half: • 1: Half resistor (high power when ICOMP_CTRL_3_0 is close to 0000) • 0: Full resistor (low power when ICOMP_CTRL_3_0 is close to 1111)
12:10	RSVD	R	0x0	Reserved bits
9:8	HYS_EN	WR	0x0	• 00: No hysteresis • 01: Positive Hysteresis only • 10: Negative Hysteresis only • 11: Hysteresis on both sides Hysteresis = 26 mV
7:4	RDEG_N_3_0	WR	0x8	Res degeneration N, for calibration only
3:0	RDEG_P_3_0	WR	0x8	Res degeneration P, for calibration only

4.7.3.4.3 MCU_WKUP_CTRL

- **Name:** MCU Wakeup Control Register of COMP
- **Description:** For each bit, 0 indicates IRQ is disabled, and 1 indicates IRQ is enabled.
- **Base Address:** 0x4000A000
- **Offset:** 0x80
- **Reset Value:** 0x0000FFFF

Table 4-406 MCU Wakeup Control Register of COMP

Bits	Field Name	RW	Reset	Description
5	EVT_PMU_MSIO_COMP	WR	0x1	• 1: Enable the wakeup function of the comp rising edge. • 0: Disable the wakeup function of the comp rising edge.

4.7.3.4.4 AON_SLP_EVENT

- **Name:** AON Sleep Event Register of COMP
- **Description:** Events that happen in sleep or active mode, controlled by MCU_WKUP_CTRL.

Note:

When clearing the individual event, do not use read-mask-write but simply write **~ (1<< EVENT_BIT)** to AON_SLP_EVENT.

- **Base Address:** 0x4000A000
- **Offset:** 0x84
- **Reset Value:** 0x00000000

Table 4-407 AON Sleep Event Register of COMP

Bits	Field Name	RW	Reset	Description
5	EVT_PMU_MSIO_COMP	WR	0x0	Write 0 to this bit to clear comp event.

4.7.3.4.5 PMU_COMP_GLITCH_REMOVE

- **Name:** COMP Glitch Remove Register
- **Description:** To remove voltage jitter, add deglitch circuit on comparator output (glitch: from 4 µs to 28 µs at a 4 µs step).
- **Base Address:** 0x4000A000
- **Offset:** 0x2C0
- **Reset Value:** 0x00000000

Table 4-408 COMP Glitch Remove Register

Bits	Field Name	RW	Reset	Description
31:3	RSVD	R		Reserved bits
2:0	GLITCH_REMOVE_CYCLE	WR	0	• 0: No glitch removal • 1–7: 4 µs*N glitch removal

4.7.3.5 Electrical Specifications

In the table below, the typical values are measured at 25°C, and the minimum and maximum values are measured at the worst process and temperature conditions.

Table 4-409 Electrical specifications

Symbol	Description	Min.	Typ.	Max.	Unit
VREF	Internal reference voltage in wakeup comparator (when VBAT is equal to or higher than 3 V)	0		1900	mV
VBAT_REF	Battery based reference voltage in wakeup comparator	0.1 * VBATL		0.8 * VBATL	V
ERR_VREF	The error rate of the Internal reference voltage (when VBAT is equal to or higher than 3 V)		1.3	5	%
ERR_VBAT_REF	The error rate of the battery based reference voltage		1	5	%
P_HYS	Positive hysteresis voltage in wakeup comparator	16	26	36	mV
N_HYS	Negative hysteresis voltage in wakeup comparator	16	26	36	mV
T_RES	Comparator response time	8		12	µs

4.8 DMA

4.8.1 Introduction

The DMA module has two instances of Direct Memory Access (DMA) controller that performs fast data transfers from/to peripherals to/from on-chip SRAM over the on-chip bus. The DMA controller is highly programmable and allows the CPU to offload the burden of serving the slow-rate peripherals.

4.8.2 Main Features

- Two DMA controller instances
- Six DMA channels per DMA instance
- 16 handshaking interfaces per DMA instance
- Each DMA has one AHB master interface for moving data.
- Each DMA has one AHB slave interface for configuration.
- Configurable maximum block size (Maximum size = 4095)
- Scatter/Gather
- Block chaining
- Low power mode with global clock gating and channel clock gating

4.8.3 Block Diagram

Figure 4-78 shows the following functional groupings of the main interfaces to the DMA controller block:

- DMA hardware request interface
- Six channels

- 128 bytes FIFO for Channel 0 and 32 bytes FIFO per other channel for source and destination
- Arbiter
- AHB master interface
- AHB slave interface

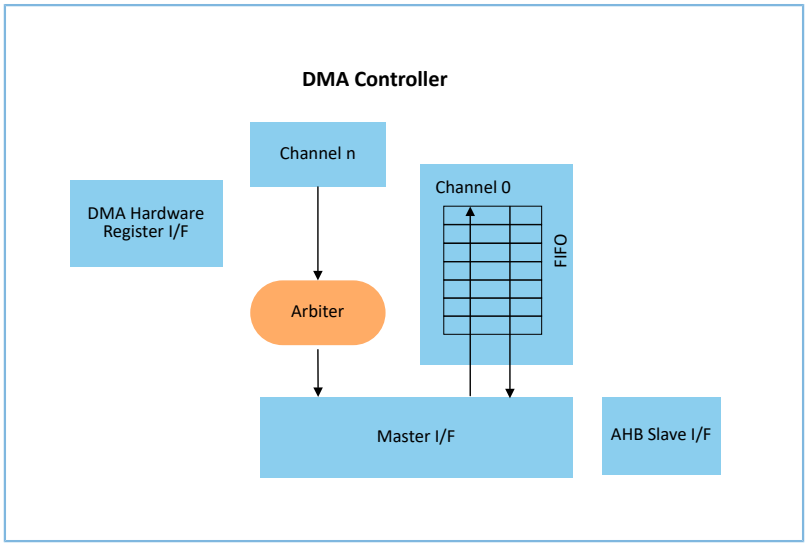

Figure 4-78 DMA controller block diagram

Figure 4-79 illustrates a peripheral-to-peripheral DMA transfer, where Peripheral A (source) uses a hardware handshaking interface, and Peripheral B (destination) uses a software handshaking interface. For example, the request to send data to Peripheral B is originated by the CPU, while writing to Peripheral B is handled by the DMA controller. The channel source and destination arbitrate independently for the AHB master interface, along with other channels.

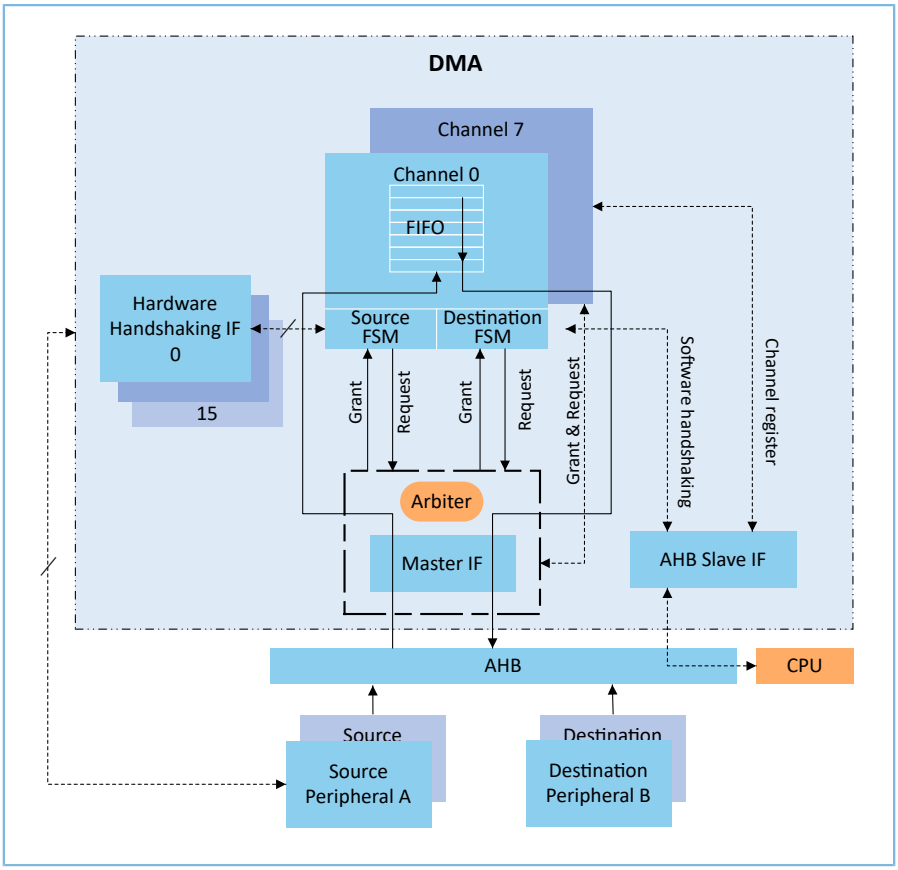

Figure 4-79 Peripheral-to-peripheral DMA transfer on the same AHB layer

4.8.4 Functional Description

4.8.4.1 Transfer Type Flow Control

The device supports DMA as flow controller only. There are four transfer types:

- Memory to memory
- Memory to peripheral
- Peripheral to memory
- Peripheral to peripheral

4.8.4.2 Transfer Width

For memory, the source transfer width must be less than or equal to the data bus width of AHB master interface, normally 8, 16, or 32 bits. For a non-memory, the source transfer width is equal to the peripheral FIFO width.

For memory, the destination transfer width must be less than or equal to the data bus width of AHB master interface, normally 8, 16, or 32 bits. For a non-memory, the destination transfer width is equal to the peripheral FIFO width.

4.8.4.3 Source and Destination Address Increment

- Source address increment: It indicates whether to increase, decrease, or remain the source address on every source transfer. If the device is fetching data from a source peripheral FIFO with a fixed address, then set this field to **No change**.
- Destination address increment: This decides on whether to increment, decrement, or remain the destination address on every source transfer. If the device is fetching data from a destination peripheral FIFO with a fixed address, then set this field to **No change**.

4.8.4.4 Channel Priority

Each channel has a programmable priority in the range of 0–5, from lowest to highest. Channel priority applies to most scenarios except those on master bus. It is worth noting that a request for the master bus interface can be made at any time, but the request is granted only after the current AHB transfer (burst or single) completes. Therefore, when the master interface is transferring data for a low-priority channel while a high priority channel requests service, it completes the current AHB transfer (burst or single) for the low-priority channel before switching to data transfer for the high one.

4.8.4.5 DMA Transfers

DMA has six channels in total; each channel has a FIFO and is independent from each other. The FIFO of each channel is 32-bit wide. After the DMA channel is enabled, the DMA obtains data from the source according to the bit width configured for the source and stores the data into the DMA FIFO. The DMA takes out the data from the DMA FIFO according to the bit width configured for the target and sends the data to the destination. Regardless of the configured transmission bit width, the data obtained from the source is compactly stored (this helps maximize the use of DAM FIFO storage space). For example, the transmission bit width is 16 bits, and the FIFO bit width is 32 bits, so one depth of DMA FIFO will store two 16-bit data entries.

It is worth mentioning that when DMA is used as flow control, the total number of bytes (blk_size_bytes_dma) transmitted by DMA in a single transmission depends on the product of the source transmission bit width (src_single_size_bytes) and the block size (CTL_HIx.BLOCK_TS). The detailed calculation methods are as follows:

$$src_single_size_bytes = CTL_LOx.SRC_TR_WIDTH / 8$$

$$blk_size_bytes_dma = CTL_HIx.BLOCK_TS * src_single_size_bytes$$

Table 4-410 Transfer operations by source/destination bit width in DMA transfer

Source Interface Width	Destination Interface Width	Block Size (DMA as Flow Control)	Source Content: Address/ Data	Transfer Operations	Destination Content: Address/ Data
8	8	4	@0x0/B0 @0x1/B1 @0x2/B2 @0x3/B3	1: READ B0[7:0] @0x0 To DMA then WRITE B0[7:0] @0x0 2 :READ B1[7:0] @0x1 To DMA then WRITE B1[7:0] @0x1 3: READ B2[7:0] @0x2 To DMA then WRITE B2[7:0] @0x2 4: READ B3[7:0] @0x3 To DMA then WRITE B3[7:0] @0x3	@0x0/B0 @0x1/B1 @0x2/B2 @0x3/B3

Source Interface Width	Destination Interface Width	Block Size (DMA as Flow Control)	Source Content: Address/ Data	Transfer Operations	Destination Content: Address/ Data
8	16	4	@0x0/B0 @0x1/B1 @0x2/B2 @0x3/B3	1 :READ B0[7:0] @0x0 To DMA 2: READ B1[7:0] @0x1 To DMA then WRITE B1B0[15:0] @0x0 3: READ B2[7:0] @0x2 To DMA 4: READ B3[7:0] @0x3 To DMA then WRITE B3B2[15:0] @0x2	@0x0/B1B0 @0x2/B3B2
8	32	4	@0x0/B0 @0x1/B1 @0x2/B2 @0x3/B3	1: READ B0[7:0] @0x0 To DMA 2: READ B1[7:0] @0x1 To DMA 3: READ B2[7:0] @0x2 To DMA 4: READ B3[7:0] @0x3 To DMA then WRITE B3B2B1B0[32:0] @0x0	@0x0/ B3B2B1B0
16	8	2	@0x0/B1B0 @0x2/B3B2	1: READ B1B0[15:0] @0x0 To DMA then WRITE B0[7:0] @0x0 2: WRITE B1[7:0] @0x1 3: READ B3B2[15:0] @0x2 To DMA then WRITE B2[7:0] @0x2 4: WRITE B3[7:0] @0x3	@0x0/B0 @0x1/B1 @0x2/B2 @0x3/B3
16	16	2	@0x0/B1B0 @0x2/B3B2	1: READ B1B0[15:0] @0x0 To DMA then WRITE B1B0[15:0] @0x0 2: READ B3B2[15:0] @0x2 To DMA then WRITE B3B2[15:0] @0x2	@0x0/B1B0 @0x2/B3B2
16	32	2	@0x0/B1B0 @0x2/B3B2	1: READ B1B0[15:0] @0x0 To DMA 2: READ B3B2[15:0] @0x2 To DMA then WRITE B3B2B1B0[32:0] @0x0	@0x0/ B3B2B1B0
32	8	1	@0x0/ B3B2B1B0	1: READ B3B2B1B0[32:0] @0x0 To DMA then WRITE B0[7:0] @0x0 2: WRITE B1[7:0] @0x1 3: WRITE B2[7:0] @0x2 4: WRITE B3[7:0] @0x3	@0x0/B0 @0x1/B1 @0x2/B2 @0x3/B3
32	16	1	@0x0/ B3B2B1B0	1: READ B3B2B1B0[32:0] @0x0 To DMA then WRITE B1B0[15:0] @0x0 2: WRITE B3B2[15:0] @0x2	@0x0/B1B0 @0x2/B3B2
32	32	1	@0x0/ B3B2B1B0	1: READ B3B2B1B0[32:0] @0x0 To DMA then WRITE B3B2B1B0[32:0] @0x0	@0x0/ B3B2B1B0

4.8.4.6 Gather and Scatter

Gather or scatter is a function that supports input and output data into/from multiple buffers. As shown in the figure below, gather refers to writing data fetched from multiple buffers to a channel (collecting data); scatter refers to reading data fetched from one channel to multiple buffers (scattering data).

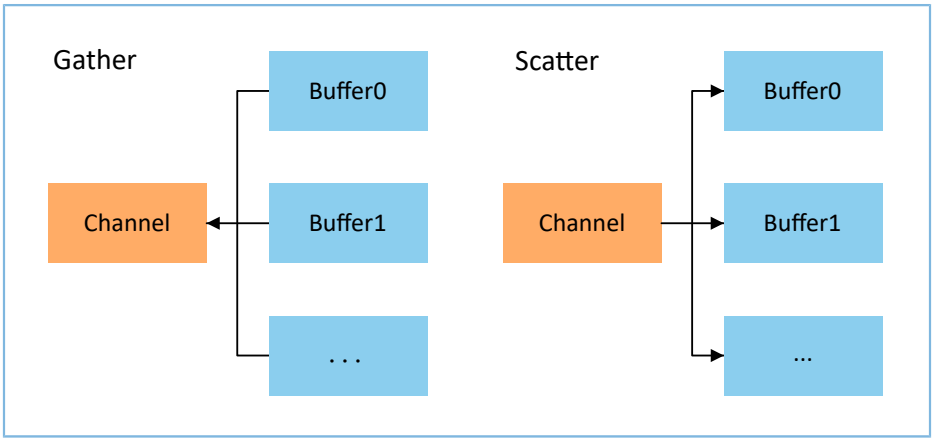

Figure 4-80 Gather/Scatter function

Gather is relevant to a source transfer. The source address is incremented or decremented by a programmed amount when a gather boundary is reached. The number of source transfers between successive gather boundaries is programmed into the Source Gather Count (SGRx.SGC) field. The source address is incremented or decremented by the value stored in the Source Gather Increment (SGRx.SGI) field multiplied by the number of bytes in a single AHB transfer from the source – (decoded value of CTL_LOx.SRC_TR_WIDTH)/8 – when a gather boundary is reached. Gather is enabled by writing a 1 to the CTL_LOx.SRC_GATHER_EN bit.

Note:

If the CTL_LOx.SINC field indicates a fixed-address control throughout a DMA transfer, then the CTL_LOx.SRC_GATHER_EN bit is ignored, and the gather feature is automatically disabled.

Figure 4-81 shows an example of a source gather transfer:

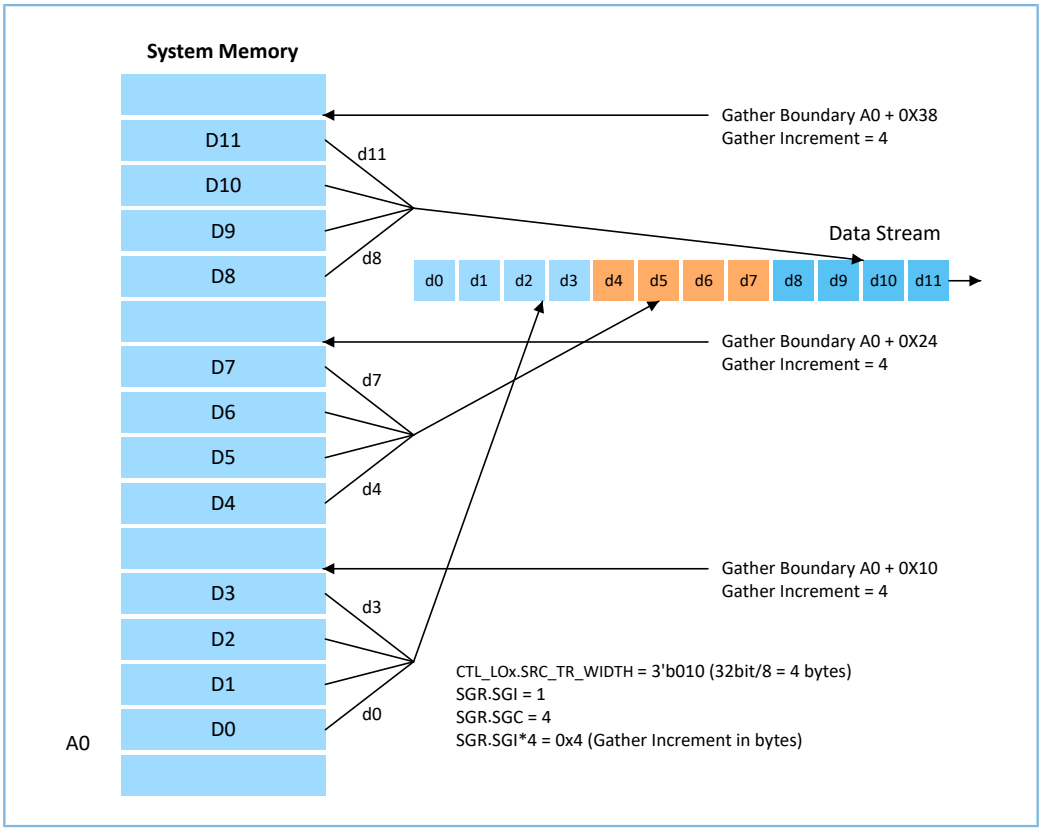

Figure 4-81 Source gather when SGR.SGI = 0x1

Scatter is relevant to a destination transfer. The destination address is incremented or decremented by a programmed amount – the scatter increment – when a scatter boundary is reached. The destination address is incremented or decremented by the value stored in the destination scatter increment (DSRx.DSI) field, multiplied by the number of bytes in a single AHB transfer to the destination (decoded value of CTL_LOx.DST_TR_WIDTH)/8 – when a scatter boundary is reached. The number of destination transfers between successive scatter boundaries is programmed into the Destination Scatter Count (DSC) field of the DSRx register. Scatter is enabled by writing a 1 to the CTL_LOx.DST_SCATTER_EN field.

Note:

If the CTL_LOx.DINC field indicates a fixed-address control throughout a DMA transfer, then the CTL_LOx.DST_SCATTER_EN field is ignored, and the scatter feature is automatically disabled.

Figure 4-82 shows an example of a destination scatter transfer:

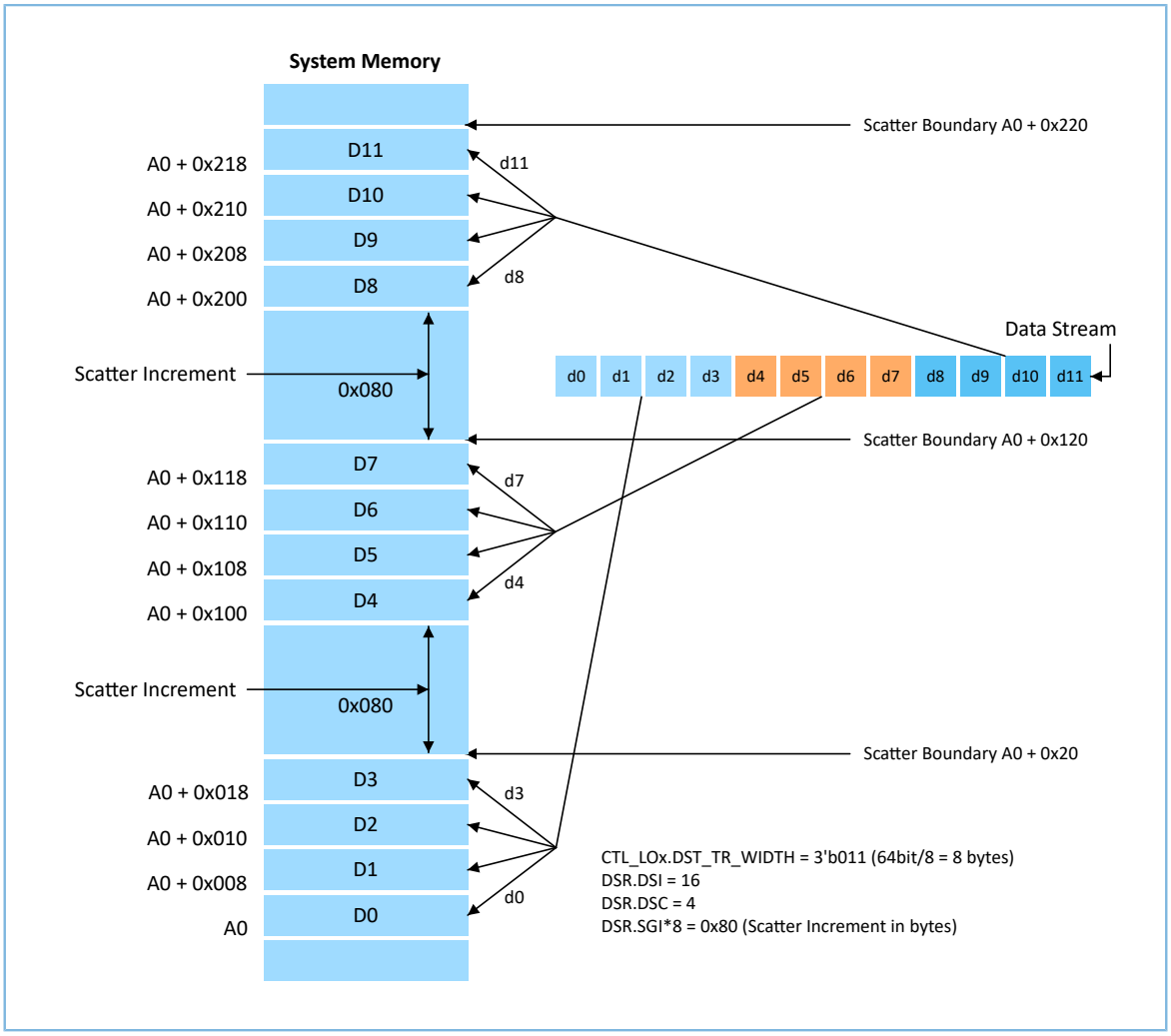

Figure 4-82 Example of destination scatter transfer

4.8.4.7 Multi-block DMA Transfer

DMA transfer may consist of multiple DW_AHB_DMAC blocks. Multi-block DMA transfers are supported through block chaining (linked list pointers) and auto-reloading channel registers. The DMA transfer modes for source and destination can be independently selected.

4.8.4.7.1 Auto-reloading

During auto-reloading, the channel registers are reloaded with their initial values at the completion of each block and the new values used for the new block. Some or all of the SARx, DARx, and CTRL_LOx registers are reloaded from their initial values at the start of a block transfer.

By configuring the Configuration Register (CFG_LOx.RELOAD_SRC), you can program the register if the SARx register can be automatically reloaded from its initial value at the end of every block for multi-block transfers.

By configuring the Configuration Register (CFG_LOx.RELOAD_DST), you can program the register if the DARx register can be automatically reloaded from its initial value at the end of every block for multi-block transfers.

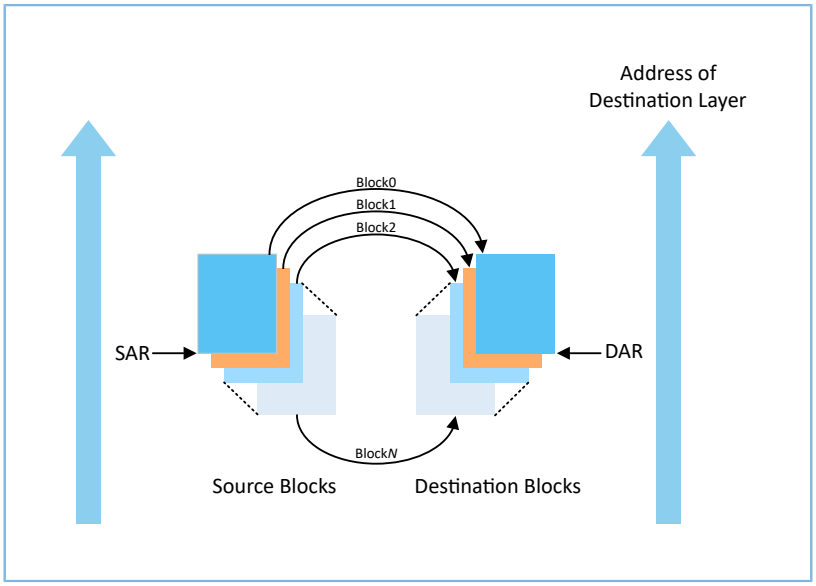

Figure 4-83 Multi-block DMA transfer with source and destination address auto-reloaded

4.8.4.7.2 Linked Lists (Block Chaining)

Linked list pointer (LLP) points to the location in system memory where the next linked list item (LLI) exists. The LLI is a set of registers that describe the next block (block descriptor) and an LLP register. The DMA fetches the LLI at the beginning of every block when block chaining is enabled. Each LLI is composed of the following parts: SAR, DAR, LLP, CTL, SSTAT, and DSTAT. Therefore, to use block chaining, a series of linked lists need to be implemented in memory. The LLP bit field of each LLI block stores the first address of the next LLI, thus forming a linked list of blocks. The following figure shows how to use chained linked lists in memory to define multi-block transfers using block chaining.

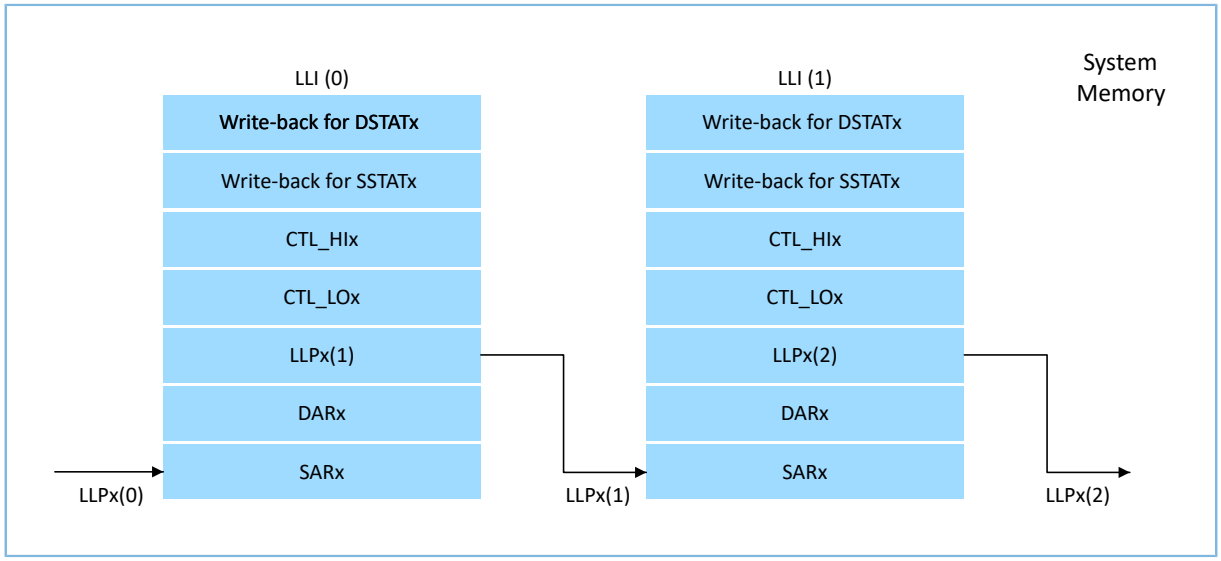

Figure 4-84 Multi-block transfer using linked lists when DMAH_CHx_STAT_SRC set to true

The DMA reprograms the channel registers prior to the start of each block by fetching the block descriptor for that block from system memory. This is known as an LLI update. The SARx, DARx, LLPx, CTL_LOx and CTL_HIx registers are

fetched from system memory on an LLI update. After that, the updated contents of the CTL_LOx, CTL_HIx, SSTATx, and DSTATx registers are written back to memory on block completion.

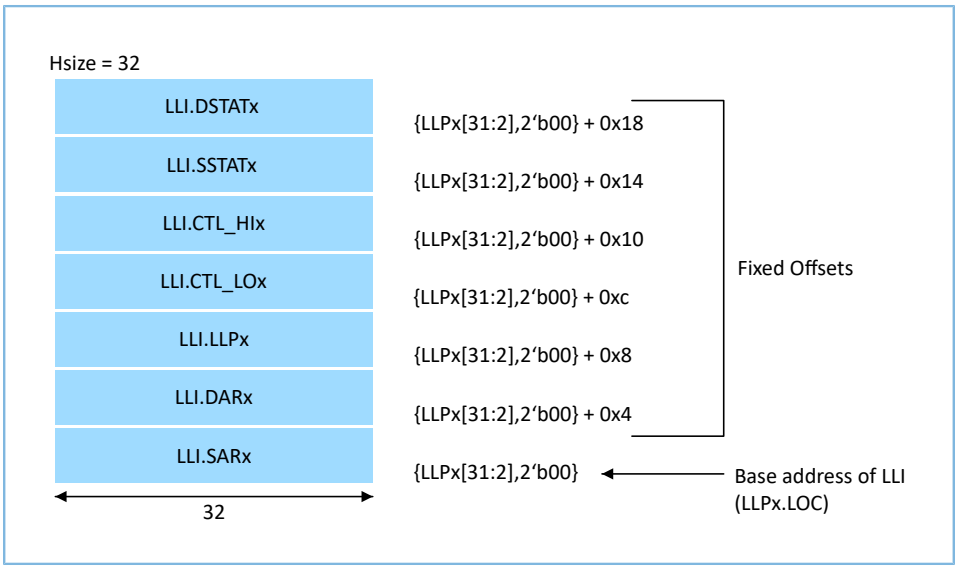

Figure 4-85 Mapping of block descriptor (LLI) in memory to channel registers when DMAH_CHx_STAT_SRC set to true

Note:

LLI accesses are always 32-bit accesses (**Hsize** = **2**) aligned to 32-bit boundaries and cannot be changed or programmed to anything other than 32-bit, even if the AHB master interface of the LLI supports more than a 32-bit data width.

4.8.4.8 DMA Interrupt

For each channel, there are three types of interrupt sources:

- IntTfr: DMA Transfer Complete Interrupt
- IntBlock: Block Transfer Complete Interrupt. This interrupt is generated on DMA block transfer completion to the destination peripheral.
- IntErr: Error Interrupt. This interrupt is generated when an ERROR response is received from an AHB slave on the HRESP bus during a DMA transfer. In addition, the DMA transfer is cancelled, and the channel is disabled.

The DMA_RAW_x, DMA_STAT_x, DMA_MASK_x, and DMA_CLR_x registers are provided to manage interrupts. The DMA_MASK_x registers is used to enable or disable the block transfer complete interrupt, DMA transfer complete interrupt, and error interrupt. By reading the DMA_STAT_x registers or DMA_RAW_x registers, you can check whether the interruption have occurred. To clear interrupts, write 1 to the DMA_CLR_x registers.

- DMA_MASK_x registers: Used to enable or disable the block transfer complete interrupt, DMA transfer complete interrupt, and error interrupt.
- DMA_STAT_x or DMA_RAW_x registers: Used to reply whether the interruption have occurred.
- DMA_CLR_x registers: Write 1 to DMA_CLR_x to clear interrupts.

4.8.4.9 DMA Hardware Interface Assignment

Each channel of DMA can communicate with designated peripheral devices through the assigned hardware handshaking interface. The hardware handshaking interfaces are assigned as shown in the following table. The developer can program the CFG_CHx.DEST_PER or CFG_CHx.SRC_PER to assign a handshaking interface for each channel source and destination.

Table 4-411 DMA0 hardware handshaking interfaces

DMA0 Request	Hardware Handshake Choice0	Handshake Peripheral Description	Hardware Handshake Choice1	Handshake Peripheral Description
0x0	QSPI0 TX	QSPI0 Master Transmit	NC	/
0x1	QSPI0 RX	QSPI0 Master Receive	NC	/
0x2	SPIM TX	SPI Master Transmit	I2C2 TX	I2C2 Transmit
0x3	SPIM RX	SPI Master Receive	I2C2 RX	I2C2 Receive
0x4	SPIS TX	SPI Slave Transmit	UART3 TX	UART3 Transmit
0x5	SPIS RX	SPI Slave Receive	UART3 RX	UART3 Receive
0x6	UART0 TX	UART0 Transmit	NC	/
0x7	UART0 RX	UART0 Receive	NC	/
0x8	UART1 TX	UART1 Transmit	NC	/
0x9	UART1 RX	UART1 Receive	NC	/
0xA	SNSADC	Sense ADC	NC	/
0xB	NC	/	NC	/
0xC	NC	/	QSPI1 TX	QSPI1 Master Transmit
0xD	NC	/	QSPI1 RX	QSPI1 Master Receive
0xE	UART2 TX	UART2 Transmit	I2C3 TX	I2C3 Transmit
0xF	UART2 RX	UART2 Receive	I2C3 RX	I2C3 Receive

Table 4-412 DMA1 hardware handshaking interfaces

DMA1 Request	Hardware Handshake Choice0	Handshake Peripheral Description	Hardware Handshake Choice1	Handshake Peripheral Description
0x0	NC	/	SPIM TX	SPI Master Transmit
0x1	NC	/	SPIM RX	SPI Master Receive
0x2	QSPI2 TX	QSPI2 Master Transmit	DSPIM TX	DSPI Master Transmit
0x3	QSPI2 RX	QSPI2 Master Receive	DSPIM RX	DSPI Master Receive
0x4	I2SM TX	I2S Master Transmit	QSPI1 TX	QSPI1 Master Transmit
0x5	I2SM RX	I2S Master Receive	QSPI1 RX	QSPI1 Master Receive
0x6	I2SS TX	I2S Slave Transmit	UART3 TX	UART3 Transmit

DMA1 Request	Hardware Handshake Choice0	Handshake Peripheral Description	Hardware Handshake Choice1	Handshake Peripheral Description
0x7	I2SS RX	I2S Slave Receive	UART3 RX	UART3 Receive
0x8	PDM	PDM Receive	NC	/
0x9	NC	/	NC	/
0xA	QSPI1 TX	QSPI1 Master Transmit	NC	/
0xB	QSPI1 RX	QSPI1 Master Receive	NC	/
0xC	I2C0 TX	I2C0 Transmit	NC	/
0xD	I2C0 RX	I2C0 Receive	NC	/
0xE	I2C1 TX	I2C1 Transmit	UART0 TX	UART0 Transmit
0xF	I2C1 RX	I2C1 Receive	UART0 RX	UART0 Receive

4.8.5 Registers

4.8.5.1 Channel_x_Registers

4.8.5.1.1 SARx

- **Name**: Source Address for Channel x
- **Description**: The starting source address is programmed by software before the DMA channel is enabled, or by an LLI update before the start of the DMA transfer. While the DMA transfer is in progress, this register is updated to reflect the source address of the current AHB transfer.

Note:

You must program the SAR address to be aligned to CT_LOx.SRC_TR_WIDTH.

- **Base Address**: 0x40019000 (DMA0)/0x4001A000 (DMA1)
- **Offset**: 0x0 + x*0x58
- **Reset Value**: 0x0

Table 4-413 Source Address Register for Channel x

Bits	Field Name	RW	Reset	Description
63:32	RSVD	R		Reserved bits
31:0	SRC_ADDR	RW	0x0	Current Source Address of DMA transfer. Updated after each source transfer. The SRC_ADDR_INC field in the CTL_LOx register determines whether the address increments, decrements, or is left unchanged on every source transfer through the block transfer.

Bits	Field Name	RW	Reset	Description
				Volatile: true

4.8.5.1.2 DARx

- **Name**: Destination Address Register for Channel x
- **Description**: The starting destination address is programmed by software before the DMA channel is enabled, or by an LLI update before the start of the DMA transfer. While the DMA transfer is in progress, this register is updated to reflect the destination address of the current AHB transfer.

Note:

You must program the DAR to be aligned to CTL_LOx.DST_TR_WIDTH.

- **Base Address**: 0x40019000 (DMA0)/0x4001A000 (DMA1)
- **Offset**: 0x8 + x*0x58
- **Reset Value**: 0x0

Table 4-414 Destination Address Register for Channel x

Bits	Field Name	RW	Reset	Description
63:32	RSVD	R		Reserved bits
31:0	DEST_ADDR	RW	0x0	Current Destination address of DMA transfer. Updated after each destination transfer. The DEST_ADDR_INC field in the CTRL_CHx register determines whether the address increments, decrements, or is left unchanged on every destination transfer throughout the block transfer. Volatile: true

4.8.5.1.3 LLPx

- **Name**: Linked List Pointer Register for Channel x
- **Description**: The LLP register has two functions:
  - The logical result of the equation LLP.LOC !=0 is used to set up the type of DMA transfer: single or multi-block. If LLP.LOC is set to 0x0, transfers using linked lists are not enabled. This register must be programmed prior to enabling the channel in order to set up the transfer type.
  - LLP.LOC != 0 contains the pointer to the next LLI for block chaining using linked lists. The LLPx register can also point to the address where write back of the control and source/destination status information occurs after block completion.
- **Base Address**: 0x40019000 (DMA0)/0x4001A000 (DMA1)

- **Offset**: 0x10 + x*0x58
- **Reset Value**: 0x0

Table 4-415 Linked List Pointer Register for Channel x

Bits	Field Name	RW	Reset	Description
63:32	RSVD	R		Reserved bits
31:2	LOC	RW	0x0	Starting Address In Memory of next LLI if block chaining is enabled. Note that the two LSBs of the starting address are not stored because the address is assumed to be aligned to a 32-bit boundary. LLI accesses are always 32-bit accesses (Hsize = 2) aligned to 32-bit boundaries and cannot be changed or programmed to anything other than 32-bit. Volatile: true
1:0	RSVD	R		Reserved bits

4.8.5.1.4 CTL_LOx

- **Name**: Control Register Low for Channel x
- **Description**: This register contains fields that control the DMA transfer.

Note:

You need to program this register prior to enabling the channel.

- **Base Address**: 0x40019000 (DMA0)/0x4001A000 (DMA1)
- **Offset**: 0x18 + x*0x58
- **Reset Value**: 0x00304801

Table 4-416 Control Register Low for Channel x

Bits	Field Name	RW	Reset	Description
31:29	RSVD	R		Reserved bits **Volatile:** true
28	LLP_SRC_EN	RW	0x0	Block chaining is enabled on the source side only if LLP_SRC_EN field is high and LLPx.LOC is non-zero . **Value:** • 0x0 (LLP_SRC_DISABLE): Block chaining using linked list is disabled on the source size.

Bits	Field Name	RW	Reset	Description
				• 0x1 (LLP_DST_ENABLE): Block chaining using linked list is enabled on the source size. **Volatile:** true
27	LLP_DST_EN	RW	0x0	Block chaining is enabled on the destination side only if LLP_DST_EN field is high and LLPx.LOC is non-zero . **Value:** • 0x0 (LLP_DST_DISABLE): Block chaining using linked list is disabled on the destination size. • 0x1 (LLP_DST_ENABLE): Block chaining using linked list is enabled on the destination size. **Volatile:** true
26:23	RSVD	R		Reserved bits **Volatile:** true
22:20	TT_FC	RW	0x3	**Transfer Type and Flow Control.** Flow control can be assigned to the DMA, the source peripheral, or the destination peripheral. **Value:** • 0x0: Transfer type is memory to memory, and flow controller is DMA. • 0x1: Transfer type is memory to peripheral, and flow controller is DMA. • 0x2: Transfer type is Peripheral to memory, and flow controller is DMA. • 0x3: Transfer type is peripheral to peripheral, and flow controller is DMA. **Volatile:** true
19	RSVD	R		Reserved bits **Volatile:** true
18	DST_GATHER_EN	RW	0x0	**Destination scatter enable.** Scatter on the destination side is applicable only when the CTL_LOx.DINC bit indicates an incrementing or decrementing address control. **Value:** • 0x0 (DST_SCATTER_DISABLE): Destination scatter is disabled.

Bits	Field Name	RW	Reset	Description
				• 0x1 (DST_SCATTER_ENABLE): Destination scatter is enabled. **Volatile:** true
17	SRC_GATHER_EN	RW	0x0	**Source gather enable**. Gather on the source side is applicable only when the CTL_LOx.SINC bit indicates an incrementing or decrementing address control. **Value:** • 0x0 (SRC_GATHER_DISABLE): Source gather is disabled. • 0x1 (SRC_GATHER_ENABLE): Source gather is enabled. **Volatile:** true
16:14	SRC_MSIZE	RW	0x1	**Source burst transaction length.** Number of data items, each of width CTL_LOx. SRC_TR_WIDTH, to be read from the source every time a burst transferred request is made from either the corresponding hardware or software handshaking interface. **Note:** This value is not related to the AHB bus master HBURST bus. **Value:** • 0x0: Number of data items to be transferred is 1. • 0x1: Number of data items to be transferred is 4. • 0x2: Number of data items to be transferred is 8. **Volatile:** true
13:11	DST_MSIZE	RW	0x1	**Destination burst transaction length.** Number of data items, each of width CTL_LOx. DST_TR_WIDTH, to be written to the destination every time a destination burst transaction request is made from either the corresponding hardware or software handshaking interface. **Note:** This value is not related to the AHB bus master HBURST bus. **Value:** • 0x0: Number of data items to be transferred is 1. • 0x1: Number of data items to be transferred is 4.

Bits	Field Name	RW	Reset	Description
				• 0x2: Number of data items to be transferred is 8. **Volatile:** true
10:9	SINC	RW	0x0	**Source address increment.** Indicates whether to increment or decrement the source address on every source transfer. If the device is fetching data from a source peripheral FIFO with a fixed address, set this field to "No change". **Value:** • 0x0: Increments the source address. • 0x1: Decrements the source address. • 0x2/0x3: No change in the source address **Volatile:** true
8:7	DINC	RW	0x0	**Destination address increment.** Indicates whether to increment or decrement the destination address on every destination transfer. If your device is writing data to a destination peripheral FIFO with a fixed address, set this field to "No Change". **Value:** • 0x0: Increments the destination address. • 0x1: Decrements the destination address. • 0x2/0x3: No change in the destination address **Volatile:** true
6:4	SRC_TR_WIDTH	RW	0x0	**Source Transfer Width.** Mapped to AHB bus Hsize. For a non-memory peripheral, typically the peripheral (source) FIFO width. **Value:** • 0x0: Source transfer width is 8 bits. • 0x1: Source transfer width is 16 bits. • 0x2: Source transfer width is 32 bits. **Volatile:** true
3:1	DST_TR_WIDTH	RW	0x0	**Destination Transfer Width.** Mapped to AHB bus Hsize. For a non-memory peripheral, typically the peripheral (destination) FIFO width. **Value:**

Bits	Field Name	RW	Reset	Description
				• 0x0: Destination transfer width is 8 bits. • 0x1: Destination transfer width is 16 bits. • 0x2: Destination transfer width is 32 bits. **Volatile:** true
0	INT_EN	RW	0x1	**Interrupt Enable Bit.** If set, all interrupt-generating sources are enabled. Functions as a global mask bit for all interrupts for the channel; raw interrupt registers still assert if CTL_LOx.INT_EN = 0. **Value:** • 0x0: Interrupt is disabled. • 0x1: Interrupt is enabled. **Volatile:** true

4.8.5.1.5 CTL_HIx

- **Name:** Control Register High for Channel x
- **Description:** This register contains fields that control the DMA transfer.

Note:

You need to program this register prior to enabling the channel.

- **Base Address**: 0x40019000 (DMA0)/0x4001A000 (DMA1)
- **Offset:** 0x1C + x*0x58
- **Reset Value:** 0x00000002

Table 4-417 Control Register High for Channel x

Bits	Field Name	RW	Reset	Description
31:13	RSVD	R		Reserved bits **Volatile:** true
12	DONE	RW	0x0	**Done bit.** If status write-back is enabled, the upper word of the control register, CTL_HIx, is written to the control register location of the LLI in system memory at the end of the block transfer with the done bit set.

Bits	Field Name	RW	Reset	Description
				Software can poll the LLI CTL_HIx.DONE bit to see when a block transfer completes. The LLI CTL_HIx.DONE bit should be cleared when the linked lists are set up in memory prior to enabling the channel. LLI accesses are always 32-bit accesses (Hsize = 2) aligned to 32-bit boundaries and cannot be changed or programmed to anything other than 32-bit.
11:0	BLOCK_TS	RW	0x2	**Block Transfer Size.** When the DMA is the flow controller, the user writes this field before the channel is enabled in order to indicate the block size. The number programmed into BLOCK_TS indicates the total number of single transactions to perform for every block transfer; a single transaction is mapped to a single AMBA beat. **Width:** The width of single transaction is determined by CTL_LOx. SRC_ TR_WIDTH. Once the transfer starts, the read-back value is the total number of data items already read from the source peripheral, regardless of what the flow controller is. When the source or destination peripheral is assigned as the flow controller, the maximum block size that can be read back saturates at 0xFFF, but the actual block size can be greater. **Volatile:** true

4.8.5.1.6 SSTATx

- **Name**: Source Status Register for Channel x
- **Description**: After each block transfer completes, hardware can retrieve the source status information from the address pointed to by the contents of the SSTATARx register. This status information is then stored in the SSTATx register and written out to the SSTATx register location of the LLI before the start of the next block. For conditions under which the source status information is fetched, it can be varied on a block-by-block basis within a DMA transfer when block chaining is enabled. If status write-back is enabled, the upper word of the control register, CTL_HIx, is written to the control register location of the LLI in system memory at the end of the block transfer.

Note:

This register is a temporary placeholder for the source status information on its way to the SSTATx register location of the LLI. The source status information should be retrieved by software from the SSTATx register location of the LLI, and not by a read of this register over the DMA controller slave interface.

- **Base Address**: 0x40019000 (DMA0)/0x4001A000 (DMA1)
- **Offset**: 0x20 + x*0x58
- **Reset Value**: 0x0

Table 4-418 Source Status Register for Channel x

Bits	Field Name	RW	Reset	Description
63:32	RSVD	R		Reserved bits
31:0	SSTAT	RW	0x0	Source status information retrieved by hardware from the address pointed to by the contents of the STATARx register Volatile: true

4.8.5.1.7 DSTATx

- **Name**: Destination Status Register for Channel x
- **Description**: After each block transfer completes, hardware can retrieve the destination status information from the address pointed to by the contents of the DSTATARx register. This status information is then stored in the DSTATx register and written out to the DSTATx register location of the LLI before the start of the next block.

Note:

This register is a temporary placeholder for the destination status information on its way to the DSTATx register location of the LLI. The destination status information should be retrieved by software from the DSTATx register location of the LLI, and not by a read of this register over the DMAC controller slave interface.

- **Base Address**: 0x40019000 (DMA0)/0x4001A000 (DMA1)
- **Offset**: 0x28 + x*0x58
- **Reset Value**: 0x0

Table 4-419 Destination Status Register for Channel x

Bits	Field Name	RW	Reset	Description
63:32	RSVD	R		Reserved bits
31:0	DSTAT	RW	0x0	Destination status information retrieved by hardware from the address pointed to by the contents of the DSTATARx register

Bits	Field Name	RW	Reset	Description
				Volatile: true

4.8.5.1.8 SSTATARx

- **Name**: Source Status Address Register for Channel x
- **Description**: After completion of each block transfer, hardware can retrieve the source status information from the user-defined address to which the contents of the SSTATARx register point. Users can select any location in system memory that would provide a 32-bit value to indicate the status of the source transfer.
- **Base Address**: 0x40019000 (DMA0)/0x4001A000 (DMA1)
- **Offset**: 0x30 + x*0x58
- **Reset Value**: 0x0

Table 4-420 Source Status Address Register for Channel x

Bits	Field Name	RW	Reset	Description
63:32	RSVD	R		Reserved bits
31:0	SSTATAR	RW	0x0	Pointer from where hardware can fetch the source status information which is registered in the SSTATx register and written out to the SSTATx register location of the LLI before the start of the next block.

4.8.5.1.9 DSTATARx

- **Name**: Destination Status Address Register for Channel x
- **Description**: After completion of each block transfer, hardware can retrieve the destination status information from the user-defined address to which the contents of the DSTATARx register point. Users can select any location in system memory that would provide a 32-bit value to indicate the status of the destination transfer.
- **Base Address**: 0x40019000 (DMA0)/0x4001A000 (DMA1)
- **Offset**: 0x38 + x*0x58
- **Reset Value**: 0x0

Table 4-421 Destination Status Address Register for Channel x

Bits	Field Name	RW	Reset	Description
63:32	RSVD	R		Reserved bits
31:0	DSTATAR	RW	0x0	Pointer from where hardware can fetch the destination status information which is registered in the DSTATx register

Bits	Field Name	RW	Reset	Description
				and written out to the DSTATx register location of the LLI before the start of the next block.

4.8.5.1.10 CFG_LOx

- **Name**: Configuration Register Low for Channel x
- **Description**: This register contains fields that configure the DMA transfer. The channel configuration register remains fixed for all blocks of a multi-block transfer.

Note:

You need to program this register prior to enabling the channel.

- **Base Address**: 0x40019000 (DMA0)/0x4001A000 (DMA1)
- **Offset**: 0x40 + x*0x58
- **Reset Value**: 0x00000e00 + (x*0x20)

Table 4-422 Configuration Register Low for Channel x

Bits	Field Name	RW	Reset	Description
31	RELOAD_DST	RW	0x0	**Automatic Destination Reload.** The DARx register can be automatically reloaded from its initial value at the end of every block for multi-block transfers. A new block transfer is then initiated. **Value:** • 0x0: Destination reload disabled. • 0x1: Destination reload disabled.
30	RELOAD_SRC	RW	0x0	**Automatic Source Reload.** The SARx register can be automatically reloaded from its initial value at the end of every block for multi-block transfers. A new block transfer is then initiated. **Value:** • 0x0: Source reload disabled. • 0x1: Source reload enabled.
29:20	RSVD	R		Reserved bits
19	SRC_HS_POL	RW	0x0	**Source handshaking interface polarity.** **Value:** • 0x0: Source handshaking interface polarity is active high.

Bits	Field Name	RW	Reset	Description
				• 0x1: Source handshaking interface polarity is active low.
18	DST_HS_POL	RW	0x0	**Destination handshaking interface polarity.** **Value:** • 0x0: Destination handshaking interface polarity is active high. • 0x1: Destination handshaking interface polarity is active low.
17:12	RSVD	R		Reserved bits
11	HSG_SEL_SRC	RW	0x1	**Selects software or hardware handshaking interface for source.** This register selects which of the handshaking interfaces - hardware or software - is active for source requests on this channel. If the source peripheral is memory, this bit is ignored. **Value:** • 0x0: Hardware handshaking interface. Software initiated transaction requests are ignored. • 0x1: Software handshaking interface. Hardware initiated transaction requests are ignored.
10	HSG_SEL_DEST	RW	0x1	**Selects a software or hardware handshaking interface for destination.** This register selects which of the handshaking interfaces - hardware or software - is active for destination requests on this channel. If the destination peripheral is memory, this bit is ignored. **Value:** • 0x0: Hardware handshaking interface. Software initiated transaction requests are ignored. • 0x1: Software handshaking interface. Hardware initiated transaction requests are ignored.
9	FIFO_EMPTY	R	0x1	**Channel FIFO status.** Indicates if there is data left in the channel FIFO. It can be used in conjunction with CFGx to cleanly disable a channel. **Value:** • 0x0: Channel FIFO is not empty.

Bits	Field Name	RW	Reset	Description
				• 0x1: Channel FIFO is empty.
8	CH_SUSP	RW	0x0	**Channel suspend.** Suspends all DMA data transfers from the source until this bit is cleared. There is no guarantee that the current transaction will complete. It can also be used in conjunction with CFGx to cleanly disable a channel without losing any data. **Value:** • 0x0: DMA transfer from the source is not suspended. • 0x1: Suspend DMA transfer from the source.
7:5	CH_PRIOR	RW	0x0	**Channel priority.** A priority of 7 is the highest priority and 0 is the lowest. This field must be programmed within the range of 0 to 3. A programmed value beyond this range will cause erroneous behavior. **Value:** • 0x0: Channel priority is 0. • 0x1: Channel priority is 1. • 0x2: Channel priority is 2. • 0x3: Channel priority is 3.
4:0	RSVD	R		Reserved bits

4.8.5.1.11 CFG_HIx

- **Name:** Configuration Register High for Channel x
- **Description:** This register contains fields that configure the DMA transfer. The channel configuration register remains fixed for all blocks of a multi-block transfer.

Note:

You need to program this register prior to enabling the channel.

- **Base Address**: 0x40019000 (DMA0)/0x4001A000 (DMA1)
- **Offset:** 0x44 + x*0x58
- **Reset Value:** 0x00000004 + (x*0x20)

Table 4-423 Configuration Register High for Channel x

Bits	Field Name	RW	Reset	Description
31:15	RSVD	R		Reserved bits
14:11	DEST_PER	RW	0x0	**Destination hardware interface.** Assigns a hardware handshaking interface to the destination of channel x if the CFG_LOx.HSG_SEL_DEST field is 0; otherwise, this field is ignored. The channel can communicate with the destination peripheral connected to that interface through the assigned hardware handshaking interface. **Note:** For correct DMA operation, only one peripheral (source or destination) should be assigned to the same handshaking interface.
10:7	SRC_PER	RW	0x0	**Source hardware interface.** Assigns a hardware handshaking interface to the source of channel x if the CFG_LOx.HSG_SEL_SRC field is 0; otherwise, this field is ignored. The channel can then communicate with the source peripheral connected to that interface through the assigned hardware handshaking interface. **Note:** For correct DMA operation, only one peripheral (source or destination) should be assigned to the same handshaking interface.
6	SS_UPD_EN	RW	0x0	**Source status update enable.** Source status information is fetched only from the location pointed to by the SSTATARx register, stored in the SSTATx register and written out to the SSTATx location of the LLI, if SS_UPD_EN is high. **Value:** • 0x0 (DISABLED):Source status update is disabled. • 0x1 (ENABLED):Source status update is enabled.
5	DS_UPD_EN	RW	0x0	**Destination status update enable.** Destination status information is fetched only from the location pointed to by the DSTATARx register, stored in the

Bits	Field Name	RW	Reset	Description
				DSTATx register and written out to the DSTATx location of the LLI, if DS_UPD_EN is high. **Value:** • 0x0 (DISABLED):Destination status update is disabled. • 0x1 (ENABLED):Destination status update is enabled.
4:2	PROTCTL	RW	0x1	**Protection control bits used to drive the AHB HPROT[3:1] bus.** The AMBA specification recommends that the default of HPROT indicate a non-cached, non-buffered, and privileged data access. The reset value is used to indicate such an access. HPROT[0] is tied high because all transfers are data accesses, as there are no opcode fetches. There is a one-to-one mapping of these register bits to the HPROT[3:1] master interface signals. Mapping of HPROT bus is as follows: • 0x1 to HPROT[0] • CFG_HIx. PROTCT [1] to HPROT[1] • CFG_HI x. PROTCT[2] to HPROT[2] • CFG_HI x. PROTCT [3] to HPROT[3]
1	FIFO_MODE	RW	0x0	**FIFO mode select.** Determines how much space or data needs to be available in the FIFO before a burst transaction request is serviced. **Value:** • 0x0: Space/Data available for a single AHB transfer of the specified transfer width • 0x1: Data available is greater than or equal to half the FIFO depth for destination transfers, and space available is greater than half the FIFO depth for source transfers. The exceptions are at the end of a burst transaction request or at the end of a block transfer.
0	FCMODE	RW	0x0	**Flow control mode.** Determines when source transaction requests are serviced when the destination peripheral is the flow controller. **Value:**

Bits	Field Name	RW	Reset	Description
				• 0x0: Source transaction requests are serviced when they occur. Data pre-fetching is enabled. • 0x1: Source transaction requests are not serviced until a destination transaction request occurs. In this mode, the amount of data transferred from the source is limited so that it is guaranteed to be transferred to the destination prior to block termination by the destination. Data pre-fetching is disabled.

4.8.5.1.12 SGRx

- **Name**: Source Gather Register for Channel x
- **Description**: The Source Gather Register contains two fields.
  - Source gather count field (SGRx.SGC): Specifies the number of contiguous source transfers of CTL_LOx.SRC_TR_WIDTH between successive gather intervals. This is defined as a gather boundary.
  - Source gather interval field (SGRx.SGI): Specifies the source address increment/decrement in multiples of CTL_LOx.SRC_TR_WIDTH on a gather boundary when gather mode is enabled for the source transfer.
- **Base Address**: 0x40019000 (DMA0)/0x4001A000 (DMA1)
- **Offset**: 0x48 + x*0x58
- **Reset Value**: 0x0

Table 4-424 Source Gather Register for Channel x

Bits	Field Name	RW	Reset	Description
63:32	RSVD	R		Reserved bits
31:20	SGC	RW	0x0	Source gather count.
19:0	SGI	RW	0x0	Source gather interval.

4.8.5.1.13 DSRx

- **Name**: Destination Scatter Register for Channel x
- **Description**: The Destination Status Register contains two fields.
  - Destination scatter count field (DSRx.DSC): Specifies the number of contiguous destination transfers of CTL_LOx.DST_TR_WIDTH between successive scatter boundaries.
  - Destination scatter interval field (DSRx.DSI): Specifies the destination address increment/decrement in multiples of CTL_LOx.DST_TR_WIDTH on a scatter boundary when scatter mode is enabled for the destination transfer.
- **Base Address**: 0x40019000 (DMA0)/0x4001A000 (DMA1)

- **Offset**: 0x50 + x*0x58
- **Reset Value**: 0x0

Table 4-425 Destination Scatter Register for Channel x

Bits	Field Name	RW	Reset	Description
63:32	RSVD	R		Reserved bits
31:20	DSC	RW	0x0	Destination scatter count.
19:0	DSI	RW	0x0	Destination scatter interval.

4.8.5.2 Interrupt_Registers

4.8.5.2.1 INT_RSTAT_TC

- **Name**: Raw Status for Transfer Complete Interrupt
- **Description**: Interrupt events are stored in this Raw Interrupt Status Register before masking. This register has a bit allocated to each channel; for example, INT_RSTAT_TC[2] is the Channel 2 raw transfer complete interrupt. Each bit in this register is cleared by writing a 1 to the corresponding location in the INT_CLR_TC register.

Note:

Write access is available to this register or software for testing purposes only. Under normal operation, writes to this register are not recommended.

- **Base Address**: 0x40019000 (DMA0)/0x4001A000 (DMA1)
- **Offset**: 0x2c0
- **Reset Value**: 0x0

Table 4-426 Raw Status for Transfer Complete Interrupt

Bits	Field Name	RW	Reset	Description
63:8	Rsvd_INT_RSTAT_TC	R	0x0	Reserved bits
7:0	RAW	RW	0x0	Raw status for transfer complete interrupt **Value**: • 0x0: Inactive raw interrupt status • 0x1: Active raw interrupt status

4.8.5.2.2 INT_RSTAT_BTC

- **Name**: Raw Status for Block Transfer Complete Interrupt

- **Description**: Interrupt events are stored in this Raw Interrupt Status Register before masking. This register has a bit allocated to each channel; for example, INT_RSTAT_BTC[2] is the Channel 2 raw block complete interrupt. Each bit in this register is cleared by writing a 1 to the corresponding location in the CLR_BLK register.

Note:

Write access is available to this register or software for testing purposes only. Under normal operation, writes to this register are not recommended.

- **Base Address**: 0x40019000 (DMA0)/0x4001A000 (DMA1)
- **Offset**: 0x2c8
- **Reset Value**: 0x0

Table 4-427 Raw Status for Block Transfer Complete Interrupt

Bits	Field Name	RW	Reset	Description
63:8	RSVD_INT_RSTAT_BTC	R	0x0	Reserved bits
7:0	RAW	RW	0x0	Raw Status for Block Transfer Complete Interrupt **Value**: • 0x0: Inactive raw interrupt status • 0x1: Active raw interrupt status

4.8.5.2.3 INT_RSTAT_STC

- **Name**: Raw Status for Source Transaction Complete Interrupt
- **Description**: Interrupt events are stored in this Raw Interrupt Status Register before masking. This register has a bit allocated to each channel; for example, INT_RSTAT_STC[2] is the Channel 2 raw source transaction complete interrupt. Each bit in this register is cleared by writing a 1 to the corresponding location in the INT_CLR_STC register.

Note:

Write access is available to this register or software testing purposes only. Under normal operation, writes to this register are not recommended.

- **Base Address**: 0x40019000 (DMA0)/0x4001A000 (DMA1)
- **Offset**: 0x2D0
- **Reset Value**: 0x0

Table 4-428 Raw Status for Source Transaction Complete Interrupt

Bits	Field Name	RW	Reset	Description
63:8	RSVD_INT_RSTAT_STC	R	0x0	Reserved bits
7:0	RAW	RW	0x0	Raw status for source transaction complete interrupt

Bits	Field Name	RW	Reset	Description
				Value: • 0x0: Inactive raw interrupt status • 0x1: Active raw interrupt status

4.8.5.2.4 INT_RSTAT_DTC

- **Name**: Raw Status for Destination Transaction Complete Interrupt
- **Description**: Interrupt events are stored in this Raw Interrupt Status register before masking. This register has a bit allocated to each channel; for example, INT_RSTAT_DTC[2] is the Channel 2 raw destination transaction complete interrupt. Each bit in this register is cleared by writing a 1 to the corresponding location in the INT_CLR_DTC register.

Note:

Write access is available to this register or software testing purposes only. Under normal operation, writes to this register are not recommended.

- **Base Address**: 0x40019000 (DMA0)/0x4001A000 (DMA1)
- **Offset**: 0x2d8
- **Reset Value**: 0x0

Table 4-429 Raw Status for Destination Transaction Complete Interrupt

Bits	Field Name	RW	Reset	Description
63:8	RSVD_INT_RSTAT_DTC	R	0x0	Reserved bits
7:0	RAW	RW	0x0	Raw status for destination transaction complete interrupt **Value**: • 0x0: Inactive raw interrupt status • 0x1: Active raw interrupt status

4.8.5.2.5 INT_RSTAT_ERR

- **Name**: Raw Status for Error Interrupt
- **Description**: Interrupt events are stored in this Raw Interrupt Status Register before masking. This register has a bit allocated to each channel; for example, INT_RSTAT_ERR[2] is the Channel 2 raw error interrupt. Each bit in this register is cleared by writing a 1 to the corresponding location in the INT_CLR_ERR register.

Note:

Write access is available to this register or software testing purposes only. Under normal operation, writes to this register are not recommended.

- **Base Address**: 0x40019000 (DMA0)/0x4001A000 (DMA1)
- **Offset**: 0x2E0
- **Reset Value**: 0x0

Table 4-430 Raw Status for Error Interrupt

Bits	Field Name	RW	Reset	Description
63:8	RSVD_INT_RSTAT_ERR	R	0x0	Reserved bits
7:0	RAW	RW	0x0	Raw Status for Error Interrupt **Value**: • 0x0: Inactive raw interrupt status • 0x1: Active raw interrupt status

4.8.5.2.6 INT_STAT_TC

- **Name**: Status for Transfer Complete Interrupt
- **Description**: Channel DMA transfer complete interrupt events from all channels are stored in this interrupt status register after masking. This register has a bit allocated to each channel; for example, INT_STAT_TC [2] is the Channel 2 source DMA transfer complete interrupt. The contents of this register are used to generate the interrupt signals (int or int_n bus, depending on interrupt polarity) leaving the DMA.
- **Base Address**: 0x40019000 (DMA0)/0x4001A000 (DMA1)
- **Offset**: 0x2E8
- **Reset Value**: 0x0

Table 4-431 Status for Transfer Complete Interrupt

Bits	Field Name	RW	Reset	Description
63:8	RSVD_INT_STAT_TC	R	0x0	Reserved bits
7:0	STAT	R	0x0	Status for transfer complete interrupt **Value**: • 0x0: Inactive interrupt status • 0x1: Active interrupt status

4.8.5.2.7 INT_STAT_BTC

- **Name**: Status for Block Transfer Complete Interrupt
- **Description**: Channel block complete interrupt events from all channels are stored in this interrupt status register after masking. This register has a bit allocated to each channel; for example, INT_STAT_BTC[2] is the Channel 2 block complete interrupt. The contents of this register are used to generate the interrupt signals (int or int_n bus, depending on interrupt polarity) leaving the DMA.

- **Base Address**: 0x40019000 (DMA0)/0x4001A000 (DMA1)
- **Offset**: 0x2F0
- **Reset Value**: 0x0

Table 4-432 Status for Block Transfer Complete Interrupt

Bits	Field Name	RW	Reset	Description
63:8	RSVD_INT_STAT_BTC	R	0x0	Reserved bits
7:0	STAT	R	0x0	Status for block transfer complete interrupt **Value**: • 0x0: Inactive interrupt status • 0x1: Active interrupt status

4.8.5.2.8 INT_STAT_STC

- **Name**: Status for Source Transaction Complete Interrupt
- **Description**: Channel source transaction complete interrupt events from all channels are stored in this interrupt status register after masking. This register has a bit allocated to each channel; for example, INT_STAT_STC[2] is the Channel 2 source transaction complete interrupt. The contents of this register are used to generate the interrupt signals (int or int_n bus, depending on interrupt polarity) leaving the DMA.
- **Base Address**: 0x40019000 (DMA0)/0x4001A000 (DMA1)
- **Offset**: 0x2F8
- **Reset Value**: 0x0

Table 4-433 Status for Source Transaction Complete Interrupt

Bits	Field Name	RW	Reset	Description
63:8	RSVD_INT_STAT_STC	R	0x0	Reserved bits
7:0	STAT	R	0x0	Status for source transaction complete interrupt **Value**: • 0x0: Inactive interrupt status • 0x1: Active interrupt status

4.8.5.2.9 INT_STAT_DTC

- **Name**: Status for Destination Transaction Complete Interrupt
- **Description**: Channel destination transaction complete interrupt events from all channels are stored in this interrupt status register after masking. This register has a bit allocated to each channel; for example, INT_STAT_DTC[2] is the Channel 2 status destination transaction complete interrupt. The contents of this register are used to generate the interrupt signals (int or int_n bus, depending on interrupt polarity) leaving the DMA.

- **Base Address**: 0x40019000 (DMA0)/0x4001A000 (DMA1)
- **Offset**: 0x300
- **Reset Value**: 0x0

Table 4-434 Status for Destination Transaction Complete Interrupt

Bits	Field Name	RW	Reset	Description
63:8	RSVD_INT_STAT_DTC	R	0x0	Reserved bits
7:0	STAT	R	0x0	Status for destination transaction complete interrupt **Value**: • 0x0: Inactive interrupt status • 0x1: Active interrupt status

4.8.5.2.10 INT_STAT_ERR

- **Name**: Status for Error Interrupt
- **Description**: Channel error interrupt events from all channels are stored in this interrupt status register after masking. This register has a bit allocated to each channel; for example, INT_STAT_ERR[2] is the Channel 2 status error interrupt. The contents of this register are used to generate the interrupt signals (int or int_n bus, depending on interrupt polarity) leaving the DMA.
- **Base Address**: 0x40019000 (DMA0)/0x4001A000 (DMA1)
- **Offset**: 0x308
- **Reset Value**: 0x0

Table 4-435 Status for Error Interrupt

Bits	Field Name	RW	Reset	Description
63:8	RSVD_INT_STAT_ERR	R	0x0	Reserved bits
7:0	STAT	R	0x0	Status for error interrupt **Value**: • 0x0: Inactive interrupt status • 0x1: Active interrupt status

4.8.5.2.11 INT_MASK_TC

- **Name**: MASK for Transfer Complete Interrupt
- **Description**: The contents of the raw status register INT_RSTAT_TC are masked with the contents of the mask register INT_MASK_TC. This register has a bit allocated to each channel; for example, INT_MASK_TC[2] is the mask bit for the Channel 2 transfer complete interrupt. A channel INT_MASK bit will be written only if the corresponding mask write enable bit in the INT_MASK_WE field is asserted on the same AHB write transfer. This

allows software to set a mask bit without performing a read-modified write operation. For example, writing hex 01x1 to the INT_MASK_TC register writes a 1 into INT_MASK_TC[0], while INT_MASK_TC[7:1] remains unchanged. Writing hex 00xx leaves INT_MASK_TC[7:0] unchanged. Writing a 1 to any bit in this register unmasks the corresponding interrupt, thus allowing the DMA to set the appropriate bit in the status registers and int_port signals.

- **Base Address**: 0x40019000 (DMA0)/0x4001A000 (DMA1)
- **Offset**: 0x310
- **Reset Value**: 0x0

Table 4-436 MASK for Transfer Complete Interrupt

Bits	Field Name	RW	Reset	Description
63:16	RSVD_INT_MASK_TC	R	0x0	Reserved bits
15:8	INT_MASK_WE	W	0x0	Interrupt Mask Write Enable **Value**: • 0x0: Interrupt mask write disabled • 0x1: Interrupt mask write enabled
7:0	INT_MASK	RW	0x0	Mask for Transfer Complete Interrupt **Value**: • 0x0: Mask the interrupts. • 0x1: Unmask the interrupts.

4.8.5.2.12 INT_MASK_BTC

- **Name**: Mask for Block Transfer Complete Interrupt
- **Description**: The contents of the raw status register INT_RSTAT_BTC are masked with the contents of the mask register INT_MASK_BTC. This register has a bit allocated to each channel; for example, INT_MASK_BTC[2] is the mask bit for the Channel 2 block complete interrupt. A channel INT_MASK bit will be written only if the corresponding mask write enable bit in the INT_MASK_WE field is asserted on the same AHB write transfer. This allows software to set a mask bit without performing a read-modified write operation. For example, writing hex 01x1 to the INT_MASK_BTC register writes a 1 into INT_MASK_BTC[0], while INT_MASK_BTC[7:1] remains unchanged. Writing hex 00xx leaves INT_MASK_BTC[7:0] unchanged. Writing a 1 to any bit in this register unmasks the corresponding interrupt, thus allowing the DMA to set the appropriate bit in the status registers and int_port signals.
- **Base Address**: 0x40019000 (DMA0)/0x4001A000 (DMA1)
- **Offset**: 0x318
- **Reset Value**: 0x0

Table 4-437 Mask for Block Transfer Complete Interrupt

Bits	Field Name	RW	Reset	Description
63:16	RSVD_INT_MASK_BTC	R	0x0	Reserved bits
15:8	INT_MASK_WE	W	0x0	Interrupt Mask Write Enable **Value**: • 0x0: Interrupt mask write disabled • 0x1: Interrupt mask write enabled
7:0	INT_MASK	RW	0x0	Mask for Block Transfer Complete Interrupt **Value**: • 0x0: Mask the interrupts. • 0x1: Unmask the interrupts.

4.8.5.2.13 INT_MASK_STC

- **Name**: MASK for Source Transaction Complete Interrupt
- **Description**: The contents of the raw status register INT_RSTAT_STC are masked with the contents of the mask register INT_MASK_STC. This register has a bit allocated to each channel; for example, INT_MASK_STC[2] is the mask bit for the Channel 2 source transaction complete interrupt. When the source peripheral of DMA channel i is memory, then the source transaction complete interrupt, INT_MASK_STC[i], must be masked to prevent an erroneous triggering of an interrupt on the int_combined signal. A channel INT_MASK bit will be written only if the corresponding mask write enable bit in the INT_MASK_WE field is asserted on the same AHB write transfer. This allows software to set a mask bit without performing a read-modified write operation. For example, writing hex 01x1 to the INT_MASK_STC register writes a 1 into INT_MASK_STC[0], while INT_MASK_STC[7:1] remains unchanged. Writing hex 00xx leaves INT_MASK_STC[7:0] unchanged. Writing a 1 to any bit in this register unmasks the corresponding interrupt, thus allowing the DMA to set the appropriate bit in the status registers and int_port signals.
- **Base Address**: 0x40019000 (DMA0)/0x4001A000 (DMA1)
- **Offset**: 0x320
- **Reset Value**: 0x0

Table 4-438 MASK for Source Transaction Complete Interrupt

Bits	Field Name	RW	Reset	Description
63:16	RSVD_INT_MASK_STC	R	0x0	Reserved bits
15:8	INT_MASK_WE	W	0x0	Interrupt Mask Write Enable **Value**: • 0x0: Interrupt mask write disabled • 0x1: Interrupt mask write enabled

Bits	Field Name	RW	Reset	Description
7:0	INT_MASK	RW	0x0	Mask for Source Transaction Complete Interrupt **Value**: • 0x0: Mask the interrupts. • 0x1: Unmask the interrupts.

4.8.5.2.14 INT_MASK_DTC

- **Name**: Mask for Destination Transaction Complete Interrupt
- **Description**: The contents of the raw status register INT_RSTAT_DTC are masked with the contents of the mask register INT_MASK_DTC. This register has a bit allocated to each channel; for example, INT_MASK_DTC[2] is the mask bit for the Channel 2 destination transaction complete interrupt. When the destination peripheral of DMA channel i is memory, then the destination transaction complete interrupt, INT_MASK_DTC[i], must be masked to prevent an erroneous triggering of an interrupt on the int_combined(_n) signal. A channel INT_MASK bit will be written only if the corresponding mask write enable bit in the INT_MASK_WE field is asserted on the same AHB write transfer. This allows software to set a mask bit without performing a read-modified write operation. For example, writing hex 01x1 to the INT_MASK_DTC register writes a 1 into INT_MASK_DTC[0], while INT_MASK_DTC[7:1] remains unchanged. Writing hex 00xx leaves INT_MASK_DTC[7:0] unchanged. Writing a 1 to any bit in this register unmasks the corresponding interrupt, thus allowing the DMA to set the appropriate bit in the status registers and int_port signals.
- **Base Address**: 0x40019000 (DMA0)/0x4001A000 (DMA1)
- **Offset**: 0x328
- **Reset Value**: 0x0

Table 4-439 Mask for Destination Transaction Complete Interrupt

Bits	Field Name	RW	Reset	Description
63:16	RSVD_INT_MASK_DTC	R	0x0	Reserved bits
15:8	INT_MASK_WE	W	0x0	Interrupt Mask Write Enable **Value**: • 0x0: Interrupt mask write disabled • 0x1: Interrupt mask write enabled
7:0	INT_MASK	RW	0x0	Mask for Destination Transaction Complete Interrupt **Value**: • 0x0: Mask the interrupts. • 0x1: Unmask the interrupts.

4.8.5.2.15 INT_MASK_ERR

- **Name**: Mask for Error Interrupt
- **Description**: The contents of the raw status register INT_RSTAT_ERR are masked with the contents of the mask register INT_MASK_ERR. This register has a bit allocated to each channel; for example, INT_MASK_ERR[2] is the mask bit for the Channel 2 error interrupt. A channel INT_MASK bit will be written only if the corresponding mask write enable bit in the INT_MASK_WE field is asserted on the same AHB write transfer. This allows software to set a mask bit without performing a read-modified write operation. For example, writing hex 01x1 to the INT_MASK_ERR register writes a 1 into INT_MASK_ERR[0], while INT_MASK_ERR[7:1] remains unchanged. Writing hex 00xx leaves INT_MASK_ERR[7:0] unchanged. Writing a 1 to any bit in this register unmasks the corresponding interrupt, thus allowing the DMA to set the appropriate bit in the Status registers and int_port signals.
- **Base Address**: 0x40019000 (DMA0)/0x4001A000 (DMA1)
- **Offset**: 0x330
- **Reset Value**: 0x0

Table 4-440 Mask for Error Interrupt

Bits	Field Name	RW	Reset	Description
63:16	RSVD_INT_MASK_ERR	R	0x0	Reserved bits
15:8	INT_MASK_WE	W	0x0	Interrupt Mask Write Enable **Value**: • 0x0: Interrupt mask write disabled • 0x1: Interrupt mask write enabled
7:0	INT_MASK	RW	0x0	Mask for Error Interrupt **Value**: • 0x0: Mask the interrupts. • 0x1: Unmask the interrupts.

4.8.5.2.16 INT_CLR_TC

- **Name**: Clear for Transfer Complete Interrupt
- **Description**: Each bit in the INT_RSTAT_TC and INT_STAT_TC is cleared on the same cycle by writing a 1 to the corresponding location in the registers. This register has a bit allocated to each channel; for example, INT_CLR_TC[2] is the clear bit for the Channel 2 transfer done interrupt. Writing a 0 has no effect. This register is not readable.
- **Base Address**: 0x40019000 (DMA0)/0x4001A000 (DMA1)
- **Offset**: 0x338
- **Reset Value**: 0x0

Table 4-441 Clear for Transfer Complete Interrupt

Bits	Field Name	RW	Reset	Description
63:8	RSVD_INT_CLR_TC	W	0x0	Reserved bits
7:0	CLR	W	0x0	Clear for Transfer Complete Interrupt **Value**: • 0x0: No effect • 0x1: Clears interrupts.

4.8.5.2.17 INT_CLR_BTC

- **Name**: Clear for Block Transfer Complete Interrupt
- **Description**: Each bit in the INT_RSTAT_BTC and INT_STAT_BTC is cleared on the same cycle by writing a 1 to the corresponding location in the registers. This register has a bit allocated to each channel; for example, INT_CLR_BTC[2] is the clear bit for the Channel 2 block done interrupt. Writing a 0 has no effect. This register is not readable.
- **Base Address**: 0x40019000 (DMA0)/0x4001A000 (DMA1)
- **Offset**: 0x340
- **Reset Value**: 0x0

Table 4-442 Clear for Block Transfer Complete Interrupt

Bits	Field Name	RW	Reset	Description
63:8	RSVD_INT_CLR_BTC	W	0x0	Reserved bits
7:0	CLR	W	0x0	Clear for Block Transfer Complete Interrupt

4.8.5.2.18 INT_CLR_STC

- **Name**: Clear for Source Transaction Complete Interrupt
- **Description**: Each bit in the INT_RSTAT_STC and INT_STAT_STC is cleared on the same cycle by writing a 1 to the corresponding location in the registers. This register has a bit allocated to each channel; for example, INT_CLR_STC[2] is the clear bit for the Channel 2 source transaction done interrupt. Writing a 0 has no effect. This register is not readable.
- **Base Address**: 0x40019000 (DMA0)/0x4001A000 (DMA1)
- **Offset**: 0x348
- **Reset Value**: 0x0

Table 4-443 Clear for Source Transaction Complete Interrupt

Bits	Field Name	RW	Reset	Description
63:8	RSVD_INT_CLR_STC	W	0x0	Reserved bits
7:0	CLR	W	0x0	Clear for Source Transaction Complete Interrupt **Value**: • 0x0: No effect • 0x1: Clears interrupts.

4.8.5.2.19 INT_CLR_DTC

- **Name**: Clear for Destination Transaction Complete Interrupt
- **Description**: Each bit in the INT_RSTAT_DTC and INT_STAT_DTC is cleared on the same cycle by writing a 1 to the corresponding location in the registers. This register has a bit allocated to each channel; for example, INT_CLR_DTC[2] is the clear bit for the Channel 2 destination transaction done interrupt. Writing a 0 has no effect. This register is not readable.
- **Base Address**: 0x40019000 (DMA0)/0x4001A000 (DMA1)
- **Offset**: 0x350
- **Reset Value**: 0x0

Table 4-444 Clear for Destination Transaction Complete Interrupt

Bits	Field Name	RW	Reset	Description
63:8	RSVD_INT_CLR_DTC	W	0x0	Reserved bits
7:0	CLR	W	0x0	Clear for Destination Transaction Complete Interrupt **Value**: • 0x0: No effect • 0x1: Clears interrupts.

4.8.5.2.20 INT_CLR_ERR

- **Name**: Clear for Error Interrupt
- **Description**: Each bit in the INT_RSTAT_ERR and INT_STAT_ERR is cleared on the same cycle by writing a 1 to the corresponding location in the registers. This register has a bit allocated to each channel; for example, INT_CLR_ERR[2] is the clear bit for the Channel 2 error interrupt. Writing a 0 has no effect. This register is not readable.
- **Base Address**: 0x40019000 (DMA0)/0x4001A000 (DMA1)
- **Offset**: 0x358
- **Reset Value**: 0x0

Table 4-445 Clear for Error Interrupt

Bits	Field Name	RW	Reset	Description
63:8	RSVD_INT_CLR_ERR	W	0x0	Reserved bits
7:0	CLR	W	0x0	Clear for Error Interrupt **Value**: • 0x0: No effect • 0x1: Clears interrupts.

4.8.5.2.21 INT_STAT_ET

- **Name**: Status for Each Interrupt Type
- **Description**: The contents of each of the five status registers INT_STAT_TC, INT_STAT_BTC, INT_STAT_STC, INT_STAT_DTC, and INT_STAT_ERR are ORed to produce a single bit for each interrupt type in the combined status register (INT_STAT_ET). This register is read-only.
- **Base Address**: 0x40019000 (DMA0)/0x4001A000 (DMA1)
- **Offset**: 0x360
- **Reset Value**: 0x0

Table 4-446 Status for Each Interrupt Type

Bits	Field Name	RW	Reset	Description
63:5	RSVD_INT_STAT_ET	R	0x0	Reserved bits
4	ERR	R	0x0	OR of the contents of INT_STAT_ERR **Value**: • 0x0: OR of the contents of INT_STAT_ERR register is 0. • 0x1: OR of the contents of INT_STAT_ERR register is 1.
3	DEST_XFE_CPLT	R	0x0	OR of the contents of INT_STAT_DTC **Value**: • 0x0: OR of the contents of INT_STAT_DTC register is 0. • 0x1: OR of the contents of INT_STAT_DTC register is 1.
2	SRC_XFE_CPLT	R	0x0	OR of the contents of INT_STAT_STC **Value**: • 0x0: OR of the contents of INT_STAT_STC register is 0. • 0x1: OR of the contents of INT_STAT_STC register is 1.
1	BLK_XFE_CPLT	R	0x0	OR of the contents of INT_STAT_BTC register **Value**: • 0x0: OR of the contents of INT_STAT_BTC register is 0.

Bits	Field Name	RW	Reset	Description
				• 0x1: OR of the contents of INT_STAT_BTC register is 1.
0	XFE_CPLT	R	0x0	OR of the contents of INT_STAT_TC register **Value**: • 0x0: OR of the contents of INT_STAT_TC register is 0. • 0x1: OR of the contents of INT_STAT_TC register is 1.

4.8.5.3 Software_Handshake_Registers

4.8.5.3.1 REQ_SST

- **Name**: Source Software Transaction Request Register
- **Description**: A bit is assigned for each channel in this register. REQ_SST[n] is ignored when software handshaking is not enabled for the source of channel n. A channel SRC_REQ bit is written only if the corresponding channel write enable bit in the SRC_REQ_WE field is asserted on the same AHB write transfer, and if the channel is enabled in the CH_EN register. For example, writing hex 0101 writes a 1 into REQ_SST[0], while REQ_SST[7:1] remains unchanged. Writing hex 00xx leaves REQ_SST[7:0] unchanged. This allows software to set a bit in the REQ_SST register without performing a read-modified write operation.
- **Base Address**: 0x40019000 (DMA0)/0x4001A000 (DMA1)
- **Offset**: 0x368
- **Reset Value**: 0x0

Table 4-447 Source Software Transaction Request Register

Bits	Field Name	RW	Reset	Description
63:16	Rsvd_1_REQ_SST	R	0x0	Reserved bits
15:8	SRC_REQ_WE	RW	0x0	Source Software Transaction Request write enable **Value**: • 0x0: Source request write disabled • 0x1: Source request write enabled
7:0	SRC_REQ	RW	0x0	Source Software Transaction Request **Value**: • 0x0: Source request is not active. • 0x1: Source request is active.

4.8.5.3.2 REQ_DST

- **Name**: Destination Software Transaction Request Register

- **Description**: A bit is assigned for each channel in this register. REQ_DST[n] is ignored when software handshaking is not enabled for the source of channel n. A channel DEST bit is written only if the corresponding channel write enable bit in the DEST_REQ_WE field is asserted on the same AHB write transfer, and if the channel is enabled in the CH_EN register.
- **Base Address**: 0x40019000 (DMA0)/0x4001A000 (DMA1)
- **Offset**: 0x370
- **Reset Value**: 0x0

Table 4-448 Destination Software Transaction Request Register

Bits	Field Name	RW	Reset	Description
63:16	Rsvd_1_REQ_DST	R	0x0	Reserved bits
15:8	DEST_REQ_WE	RW	0x0	Destination Software Transaction Request write enable **Value**: • 0x0: Destination request write disabled • 0x1: Destination request write enabled
7:0	DEST_REQ	RW	0x0	Destination Software Transaction Request **Value**: • 0x0: Destination request is not active. • 0x1: Destination request is active.

4.8.5.3.3 REQ_SGL_ST

- **Name**: Source Single Transaction Request Register
- **Description**: A bit is assigned for each channel in this register. REQ_SGL_ST is ignored when software handshaking is not enabled for the source of channel n. A channel SRC_SGL_REQ bit is written only if the corresponding channel write enable bit in the SRC_SGL_REQ_WE field is asserted on the same AHB write transfer, and if the channel is enabled in the CH_EN register.
- **Base Address**: 0x40019000 (DMA0)/0x4001A000 (DMA1)
- **Offset**: 0x378
- **Reset Value**: 0x0

Table 4-449 Source Single Transaction Request Register

Bits	Field Name	RW	Reset	Description
63:16	Rsvd_1_REQ_SGL_ST	R	0x0	Reserved bits
15:8	SRC_SGL_REQ_WE	RW	0x0	Source Single Transaction Request write enable **Value**: • 0x0: Single write disabled

Bits	Field Name	RW	Reset	Description
				• 0x1: Single write enabled
7:0	SRC_SGL_REQ	RW	0x0	Source Single Transaction Request **Value**: • 0x0: Source request is not active. • 0x1: Source request is active.

4.8.5.3.4 REQ_SGL_DT

- **Name**: Destination Single Transaction Request Register
- **Description**: A bit is assigned for each channel in this register. REQ_SGL_DT is ignored when software handshaking is not enabled for the destination of channel n. A channel DEST_SGL_REQ bit is written only if the corresponding channel write enable bit in the DST_SGL_REQ_WE field is asserted on the same AHB write transfer, and if the channel is enabled in the CH_EN register.
- **Base Address**: 0x40019000 (DMA0)/0x4001A000 (DMA1)
- **Offset**: 0x380
- **Reset Value**: 0x0

Table 4-450 Destination Single Transaction Request Register

Bits	Field Name	RW	Reset	Description
63:16	Rsvd_1_REQ_SGL_DT	R	0x0	Reserved bits
11:8	DST_SGL_REQ_WE	RW	0x0	Destination Single Transaction Request write enable **Value**: • 0x0: Destination write disabled • 0x1: Destination write enabled
7:0	DEST_SGL_REQ	RW	0x0	Destination Single Transaction Request **Value**: • 0x0: Destination single or burst request is not active. • 0x1: Destination single or burst request is active.

4.8.5.3.5 REQ_LST_ST

- **Name**: Source Last Transaction Request Register
- **Description**: A bit is assigned for each channel in this register. REQ_LST_ST is ignored when software handshaking is not enabled for the source of channel n, or when the source of channel n is not a flow controller. A channel LST_SRC bit is written only if the corresponding channel write enable bit in the LST_SRC_WE field is asserted on the same AHB write transfer, and if the channel is enabled in the CH_EN register.
- **Base Address**: 0x40019000 (DMA0)/0x4001A000 (DMA1)

- **Offset**: 0x388
- **Reset Value**: 0x0

Table 4-451 Source Last Transaction Request Register

Bits	Field Name	RW	Reset	Description
63:16	Rsvd_1_REQ_LST_ST	R	0x0	Reserved bits
15:8	LST_SRC_WE	RW	0x0	Source Last Transaction Request write enable **Value**: • 0x0: Source last transaction request write disabled • 0x1: Source last transaction request write enabled
7:0	LST_SRC	RW	0x0	Source Last Transaction Request register **Value**: • 0x0: Not last transaction in current block • 0x1: Last transaction in current block

4.8.5.3.6 REQ_LST_DT

- **Name**: Destination Last Transaction Request Register
- **Description**: A bit is assigned for each channel in this register. REQ_LST_DT is ignored when software handshaking is not enabled for the destination of channel n or when the destination of channel n is not a flow controller. A channel LST_DEST bit is written only if the corresponding channel write enable bit in the LST_DEST_WE field is asserted on the same AHB write transfer, and if the channel is enabled in the CH_EN register.
- **Base Address**: 0x40019000 (DMA0)/0x4001A000 (DMA1)
- **Offset**: 0x390
- **Reset Value**: 0x0

Table 4-452 Destination Last Transaction Request Register

Bits	Field Name	RW	Reset	Description
63:16	Rsvd_1_REQ_LST_DT	R	0x0	Reserved bits
15:8	LST_DEST_WE	RW	0x0	Destination Last Transaction Request write enable **Value**: • 0x0: Destination last transaction request write disabled • 0x1: Destination last transaction request write enabled
7:0	LST_DEST	RW	0x0	Destination Last Transaction Request **Value**: • 0x0: Not last transaction in current block

Bits	Field Name	RW	Reset	Description
				• 0x1: Last transaction in current block

4.8.5.4 Miscellaneous_Registers

4.8.5.4.1 CFG

- **Name**: DMA Configuration Register
- **Description**: This register is used to enable the DMA, which must be done before any channel activity. If the global channel enable bit is cleared while any channel is still active, CFG.DMA_EN still returns 1 to indicate that there are channels still active until hardware has terminated all activities on all channels, at which point the CFG.DMA_EN bit returns 0.
- **Base Address**: 0x40019000 (DMA0)/0x4001A000 (DMA1)
- **Offset**: 0x398
- **Reset Value**: 0x0

Table 4-453 DMA Configuration Register

Bits	Field Name	RW	Reset	Description
63:1	RSVD_CFG	R	0x0	Reserved bits
0	DMA_EN	RW	0x0	DMA Enable bit. **Value**: • 0x0: DMA disabled • 0x1: DMA enabled

4.8.5.4.2 CH_EN

- **Name**: DMA Channel Enable Register
- Description: This is the DMA Channel Enable Register. If software needs to set up a new channel, it can read this register in order to find out which channels are currently inactive; it can then enable an inactive channel with the required priority. All bits of this register are cleared to 0 when the global DMA channel enable bit, CFG[0], is 0. When the global channel enable bit is 0, a write to the CH_EN register is ignored and a read will always read back 0. The channel enable bit, CH_EN.CH_EN, is written only if the corresponding channel write enable bit, CH_EN.CH_EN_WE, is asserted on the same AHB write transfer. For example, writing hex 01x1 writes a 1 into CH_EN[0], while CH_EN[7:1] remains unchanged. Writing hex 00xx leaves CH_EN[7:0] unchanged. Note that a read-modified write is not required.
- **Base Address**: 0x40019000 (DMA0)/0x4001A000 (DMA1)
- **Offset**: 0x3a0
- **Reset Value**: 0x0

Table 4-454 DMA Channel Enable Register

Bits	Field Name	RW	Reset	Description
63:16	Rsvd_1_CH_EN	R	0x0	Reserved bits
15:8	CH_EN_WE	W	0x0	Channel enable register
7:0	CH_EN	RW	0x0	Channel Enable. The CH_EN.CH_EN bit is automatically cleared by hardware to disable the channel after the last AMBA transfer of the DMA transfer to the destination has completed. Software can therefore poll this bit to determine when this channel is free for a new DMA transfer. **Value**: • 0x0: Disable the channel. • 0x1: Enable the channel.

5 Communication Subsystem

5.1 Supported Features

The device has a communication subsystem that supports Bluetooth Low Energy 5.3.

5.2 Transceiver

The transceiver consists of a receiver path with Low Noise Amplifier (LNA), Mixer, Baseband (BB) amplifier, and an Analog-to-Digital Converter (ADC).

The digitized signals are sent to the digital frontend (DFE) for demodulation. The DFE provides AGC feedback signals to adjust the gain of the LNA and BB amplifier to maximize the SNR at the demodulation.

On the transmit side, the digital signal from the DFE is used to modulate the CPLL_192M and delivers the modulated carrier to the PA which is configurable by the digital gain settings to deliver the required output power.

RF frequency and digital clocks are generated from the HFXO_32M.

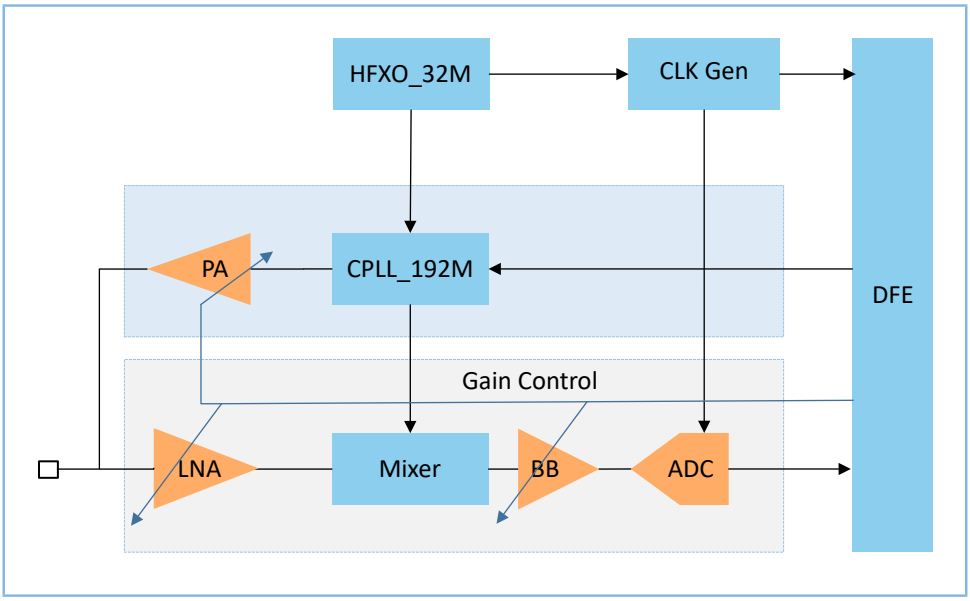

Figure 5-1 Transceiver architecture

5.3 Digital Bluetooth LE Subsystem

The digital side of Bluetooth LE subsystem is shown in Figure 5-2.

On the receive side, the digitized samples from the transceiver are filtered by a digital filter to remove the unwanted blockers, and the RSSI information is extracted. Based on the signal strength, a feedback signal is sent back to the transceiver to adjust the gain of the LNA and BB accordingly. The modem detects the frame start using the preamble, and the frequency demodulator extracts the bits inside the Bluetooth LE packet. Bit stream processing block checks the CRC of the packet content and writes it to the packet memory along with any control or status information (RSSI and RF Channel). The crypto engine decrypts the packet if it is encrypted using AES-128. The MAC firmware running on the SoC's MCU gets the packet information and processes it accordingly.

On the transmit side, the MAC firmware prepares the data to be transmitted and maps it to an event with a specific time stamp according to its connection. The Event Scheduler sets the event time and configures the packet controller to fill the data into packets and writes it to the packet memory along with its control information. The memory controller sends the packet from the memory to the modem and configures the RF channel frequency for the data packets using the packet control parameters. The packet is encrypted; CRC is added and modulated, and then sent to the frontend that passes it through a Gaussian filter and hands it to the RF transceiver.

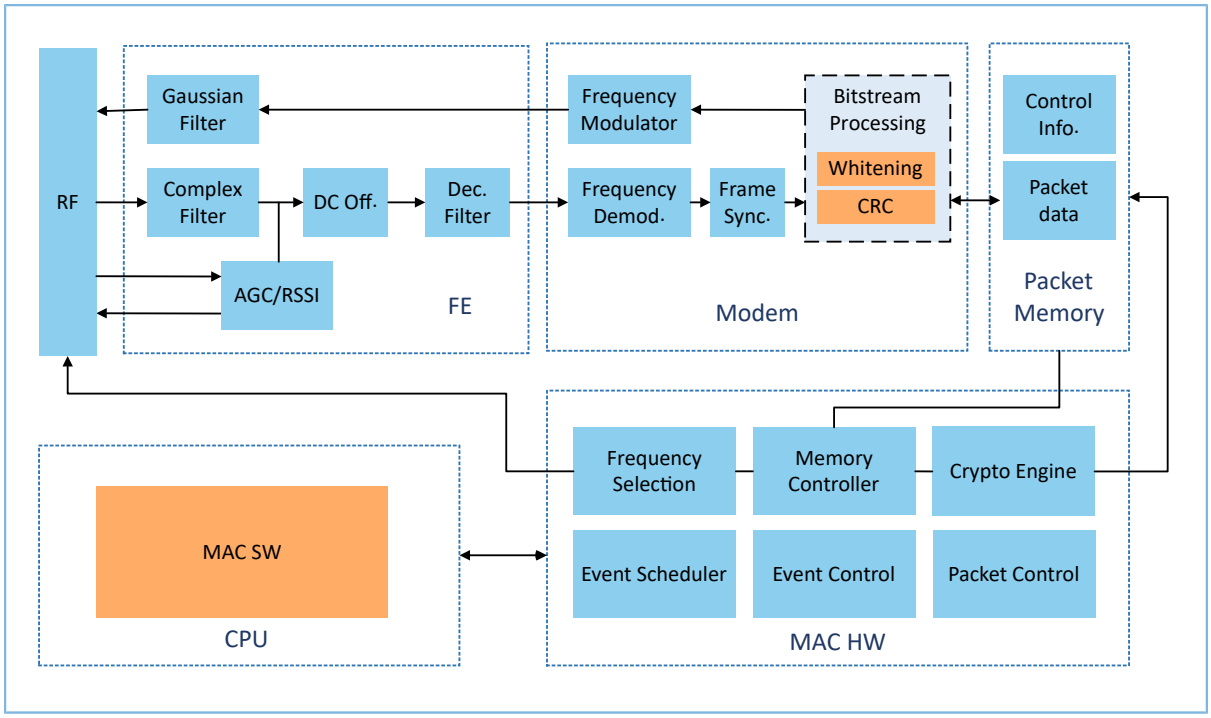

Figure 5-2 Digital Bluetooth LE subsystem architecture

5.4 Electrical Specifications

All electrical parameters in the tables below are specified under the following conditions, unless stated otherwise:

- Typical values are based on TA = 25°C and supplies at 3.3 V, by production test and/or technology characterization.
- Radio performance paramters are measured in conducted mode, based on Goodix Laboratories reference designs using output power-specific external RF impedance-matching networks for interfacing to a 50 Ω antenna.
- The minimum and maximum values represent the worst conditions across supply voltage, process variation, and operating temperature, unless stated otherwise.

5.4.1 Receiver Performance

Table 5-1 Receiver performance

Parameter	Min.	Typ.	Max.	Unit
Frequency	2402		2480	MHz

Parameter		Min.	Typ.	Max.	Unit
Sensitivity	1 Mbps, 37 B, dirty off[1]	-97.5	-97	-96.5	dBm
	1 Mbps, 255 B, dirty on	-95	-94.5	-94	
	2 Mbps, 37 B, dirty off	-95.5	-95	-94.5	
	LR2, 37 B, dirty off	-103	-102.5	-102	
	LR8, 37 B, dirty off	-105	-104.5	-104	
Maximum receive signal level		-5	-5	-5	
CCI		8	8	8	dBc
ACI (N±1)		-5	-8	-11	
N-2 Blocker (Image)		-26	-25	-24	
N+2 Blocker		-37	-40	-42	
N-3 Blocker (Adj. Image)		-37	-37	-37	
N+3 Blocker		-42	-42	-42	
N±4 or greater		-45	-45	-45	

[1] The typical sensitivity is the average value across all channels with RF PRIOR_RF_RX_PERFORMANCE settings. If with RF PRIOR_RF_RX_POWER settings, the sensitivity would degrade 0.6 dBm. Please refer to the API ble_rf_rx_performance_selection for this settings.

5.4.2 Transmitter Performance

Table 5-2 Transmitter performance

Parameter	Min.	Typ.	Max.	Unit
Frequency	2402		2480	MHz
Output power range	-20	0	7	dBm
In-band Spurious (N±2)	-40	-37	-35	
In-band Spurious (N±3)	-45	-42	-41	
2nd harmonic P_{out}[1]	-	-42	-40	
3rd harmonic P_{out}	-	-45	-40	
4th harmonic P_{out}	-	-50	-40	
5th harmonic P_{out}	-	-45	-40	
Frequency deviation (Df1avg)	238	248	258	kHz

[1] Notch circuits need to be added to board level applications for better 2nd harmonic performance.

6 Package Information

This chapter provides comprehensive details on mechanical packaging information.

6.1 QFN68

The following table illustrates the dimensions of a device in QFN68 package, which is qualified to MSL3.

Table 6-1 QFN68 package information

Parameter	Value	Unit	Tolerance
Package Size	7.0 x 7.0	mm	±0.05 mm
QFN Pad Count	68		
Total Thickness	0.85	mm	±0.05 mm
QFN Pad Pitch	0.35		
Pad Width	0.15		±0.05 mm
Exposed Pad Size	5.49 x 5.49		±0.1 mm

The figure below shows the QFN68 package outlines.

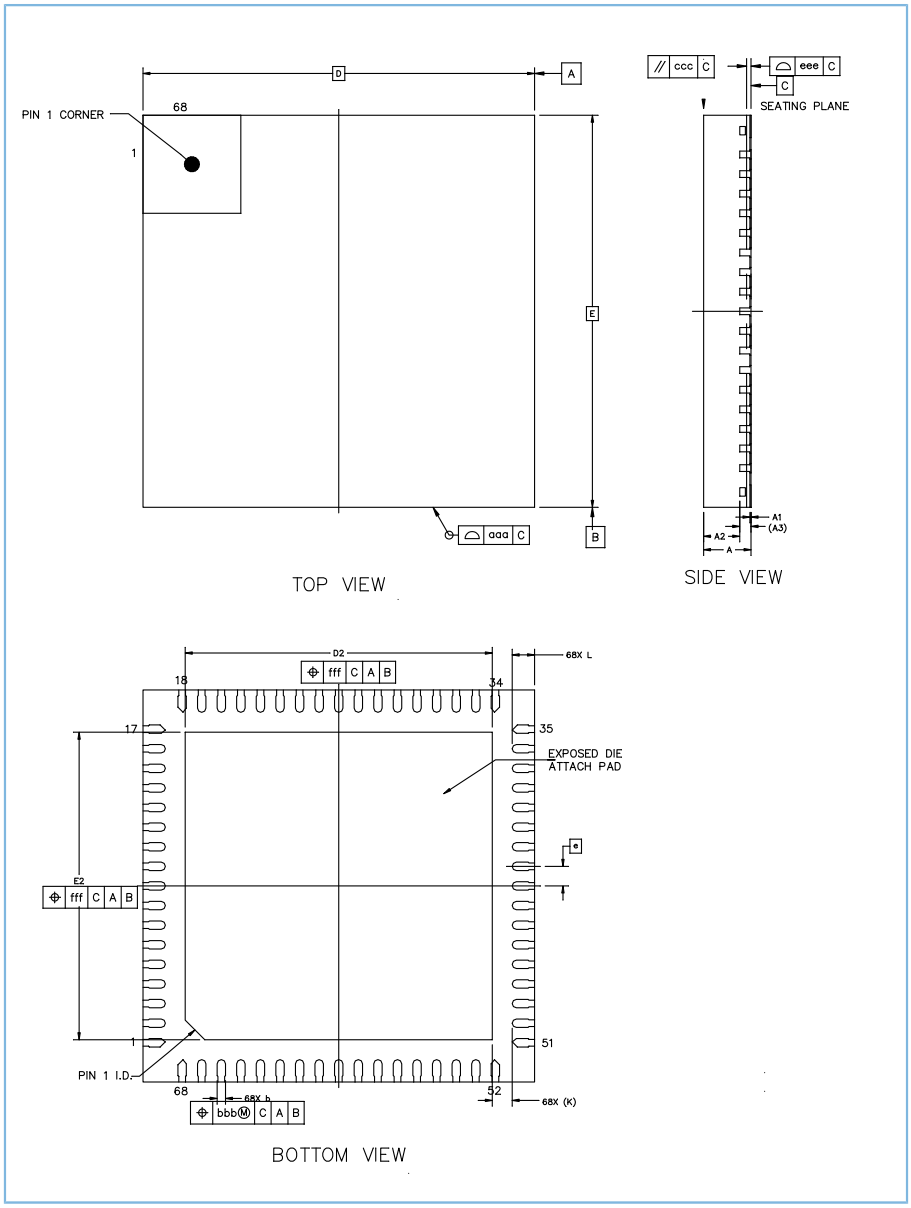

Figure 6-1 QFN68 package outlines

Note:

Drawing is not to scale.

Table 6-2 QFN68 package dimensions

Symbol	Dimensions in mm			Dimensions in inch		
	MIN.	NOM.	MAX.	MIN.	NOM.	MAX.
A	0.800	0.850	0.900	0.031	0.033	0.035
A1	0.000	0.020	0.050	0.000	0.001	0.002
A2	-	0.650	-	-	0.026	-

Symbol	Dimensions in mm			Dimensions in inch		
	MIN.	NOM.	MAX.	MIN.	NOM.	MAX.
A3	0.203 REF.			0.008 REF.		
b	0.100	0.150	0.200	0.004	0.006	0.008
D	7.000 BSC.			0.276 BSC.		
E	7.000 BSC.			0.276 BSC.		
e	0.350 BSC.			0.014 BSC.		
D2	5.390	5.490	5.590	0.212	0.216	0.220
E2	5.390	5.490	5.590	0.212	0.216	0.220
L	0.350	0.400	0.450	0.014	0.016	0.018
K	0.355 REF.			0.014 REF.		
aaa	0.100			0.004		
ccc	0.100			0.004		
eee	0.080			0.003		
bbb	0.070			0.003		
fff	0.100			0.004		

Note:

Values in inches are converted from values in millimeter and rounded to three decimal digits.

6.2 QFN56

The following table illustrates the dimensions of a device in QFN56 package, which is qualified to MSL3.

Table 6-3 QFN56 package information

Parameter	Value	Unit	Tolerance
Package Size	7.0 x 7.0	mm	±0.1 mm
QFN Pad Count	56		
Total Thickness	0.75	mm	±0.05 mm
QFN Pad Pitch	0.40		
Pad Width	0.20		±0.05 mm
Exposed Pad Size	5.2 x 5.2		±0.1 mm

The figure below shows the QFN56 package outlines.

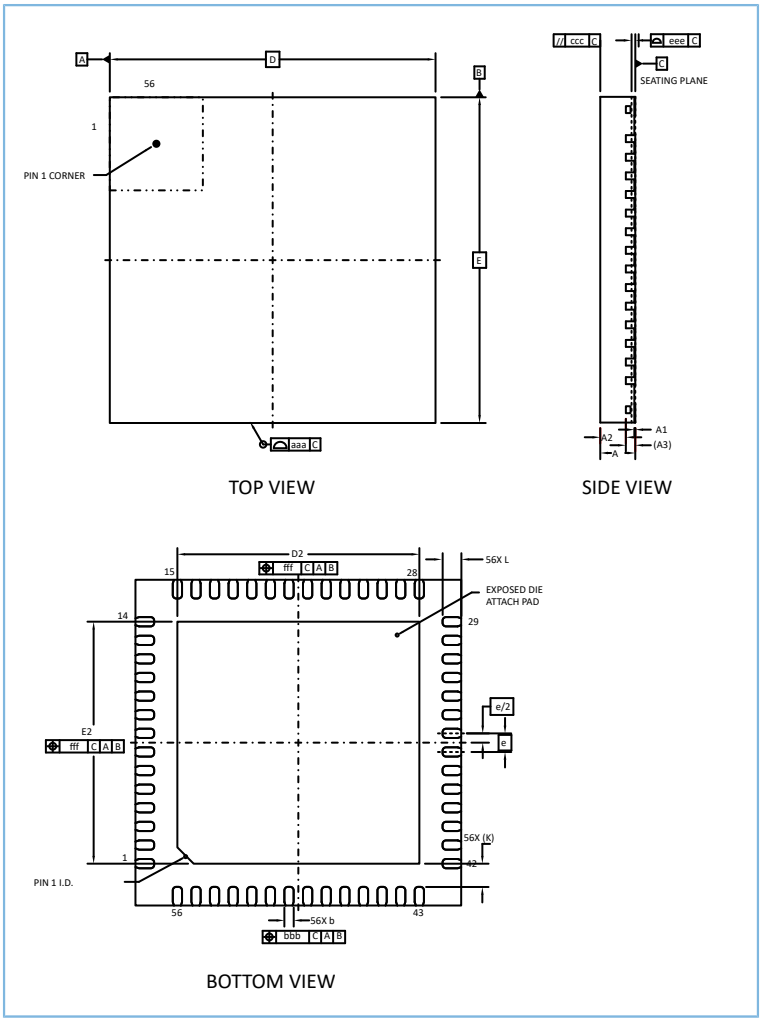

Figure 6-2 QFN56 package outlines

Note:

Drawing is not to scale.

Table 6-4 QFN56 package dimensions

Symbol	Dimensions in mm			Dimensions in inch		
	MIN.	NOM.	MAX.	MIN.	NOM.	MAX.
A	0.700	0.750	0.800	0.028	0.030	0.032
A1	0.000	0.020	0.050	0.000	0.001	0.002
A2	-	0.550	-	-	0.022	-
A3	0.203 REF.			0.008 REF.		
b	0.150	0.200	0.250	0.006	0.008	0.010
D	7.000 BSC.			0.276 BSC.		
E	7.000 BSC.			0.276 BSC.		

Symbol	Dimensions in mm			Dimensions in inch		
	MIN.	NOM.	MAX.	MIN.	NOM.	MAX.
e	0.400 BSC.			0.016 BSC.		
D2	5.100	5.200	5.300	0.201	0.205	0.209
E2	5.100	5.200	5.300	0.201	0.205	0.209
L	0.300	0.400	0.500	0.012	0.016	0.020
K	0.500 REF.			0.020 REF.		
aaa	0.100			0.004		
ccc	0.100			0.004		
eee	0.080			0.003		
bbb	0.070			0.003		
fff	0.100			0.004		

Note:

Values in inches are converted from values in millimeter and rounded to three decimal digits.

7 Absolute Maximum Ratings

Maximum ratings are the extreme limits to which the chip can be exposed for a limited amount of time without permanently damaging it. Exposure to absolute maximum ratings for prolonged periods of time may affect the reliability of the device.

Table 7-1 Absolute maximum ratings

Symbol	Description	Min.	Max.	Unit
VBATL	VBATL power supply	-0.3	3.8	V
VBATT_RF	VBATT_RF power supply	-0.3	3.8	V
VDDIO_1	Digital I/O supply input	-0.3	3.8	V
$V_{I/O}$	I/O pin voltage	-0.3	3.8	V
T_{STG}	Storage temperature	-50	150	°C
T_{JMAX}	Junction temperature		125	°C
R_{JA}	Junction-to-ambient thermal resistance			°C/W
R_{JC}	Junction-to-case thermal resistance			°C/W
MSL	Moisture sensitivity level		2	
ESD HBM	Human body model; all pins		2	kV
ESD CDM	Charged device model; all pins (excluding the antenna pin)		500[1]	V

[1] The ESD CDM of GR5525 rated at 400 V on the antenna pin. JEDEC document JEP157 states that 250-V CDM allows secure manufacturing with a standard ESD control process.

8 Recommended Operating Conditions

The operating conditions are the physical parameters that the chip can operate within.

Table 8-1 Recommended operating conditions

Symbol	Description	Min.	Typ.	Max.	Unit
VBATL	VBATL power supply	2.4	3.3	3.8	V
VBATT_RF	VBATT_RF power supply	2.4	3.3	3.8	V
VIO_3V3	3.3 V I/O pins		3.3		V
VIO_1V8	1.8 V I/O pins		1.8		V
TA	Operating temperature	-40	25	85	°C

9 Ordering Codes

Taking GR5525RGNI as an example, the figure bellow shows the ordering codes and definitions for GR5525 series.

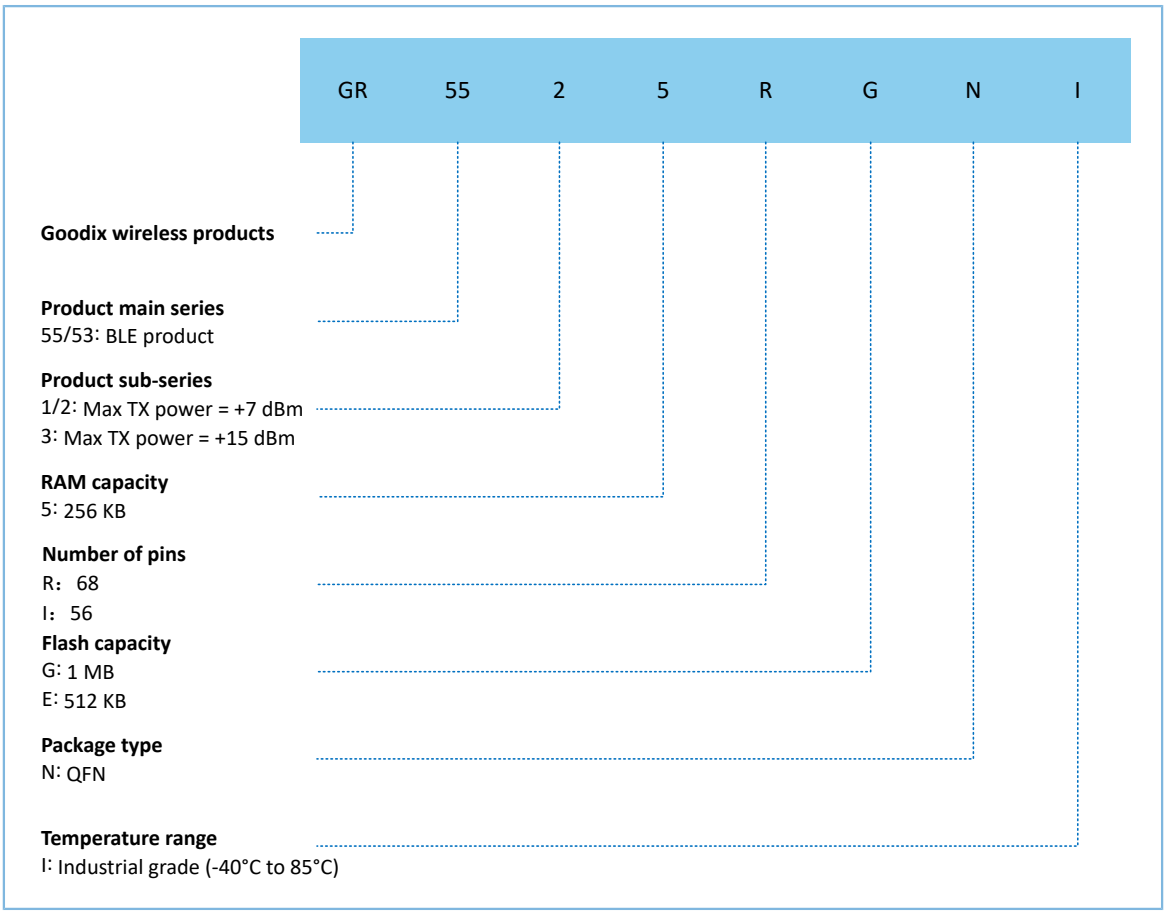

Figure 9-1 Ordering codes and definitions for GR5525 series

10 Tape and Reel Information

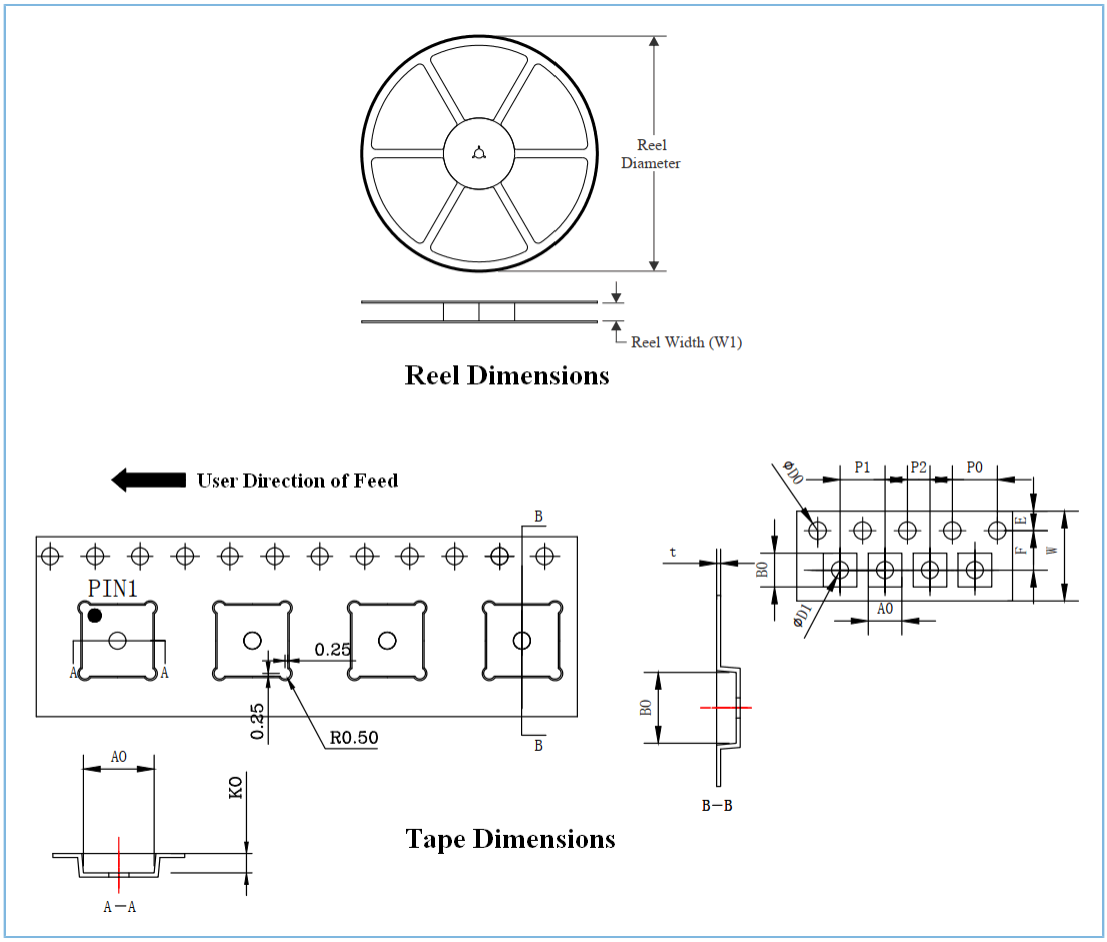

Figure 10-1 Tape and reel dimensions

Note:

- The design of carrier tape must meet EIA-481.
- Each 10 sprocket holes pitch cumulative tolerance is ± 0.2 mm.
- Carrier tape camber should not exceed 1 mm in 250 mm.
- Meet RoHs environment material management.

The detailed dimensions are outlined in the table below.

Table 10-1 Tape and reel dimensions (mm)

Package Type	Reel Diameter	Reel Width	W	E	F	D0	D1	P0	P2	10P0	P1	A0	B0	K0	t
QFN68 7 x 7	178.00	16.00	16.00 ±0.15	1.75 ±0.10	7.50 ±0.10	1.50 +0.10/-0	1.50 +0.10/-0	4.00 ±0.10	2.00 ±0.10	40.00 ±0.20	12.00 ±0.10	7.30 ±0.10	7.30 ±0.10	1.10 ±0.10	0.30 ±0.05

Package Type	Reel Diameter	Reel Width	W	E	F	D0	D1	P0	P2	10P0	P1	A0	B0	K0	t
QFN56 7 x 7	178.00	16.00	16.00 ±0.15	1.75 ±0.10	7.50 ±0.10	1.50 +0.10/-0	1.50 +0.10/-0	4.00 ±0.10	2.00 ±0.10	40.00 ±0.20	12.00 ±0.10	7.30 ±0.10	7.30 ±0.10	1.10 ±0.10	0.30 ±0.05

11 Glossary

Table 11-1 Glossary

Name	Description
ADC	Analog to Digital Converter
AES	Advanced Encryption Standard
AGC	Automatic Gain Control
AMBA	Advanced Microcontroller Bus Architecture
AMS	Analog Mix Signal
AON	Always-on
AONWDG	Always-on Watchdog Timer
APB	Advanced Peripheral Bus
ATCSR	Auto Temperature Compensated Self-Refresh
BB	Baseband
BOD	Brown-Out Detector
Bluetooth LE	Bluetooth Low Energy
BOD	Brown-out Detector
CBC	Cipher Block Chaining
DC	Display Control
DFE	Digital Frontend
DPA	Differential Power Analysis
DSPI	Dual-lane SPI
DWT	Data Watchpoint Trigger
ECB	Electronic Codebook
ESL	Electronic Shelf Label
FIPS	Federal Information Processing Standards
FPB	Flash Patch and Breakpoint
FPU	Floating Point Unit
GPIO	General-Purpose I/O
HMAC	Hash-based Message Authentication Code
IFS	Inter Frame Spacing
IoT	Internet of Things
IRQ	Interrupt Request
ISRs	Interrupt Service Routines
ITM	Instrumentation Trace and Macrocell

Name	Description
IV	Initialization Vector
LDO	Low Dropout
LFU	Least Frequently Used
LNA	Low Noise Amplifier
LP Comp	Low-Power Comparator
LPD	Low Power Domain
LSB	Least Significant Bit
MPU	Memory Protection Unit
MSB	Most Significant Bit
MSIO	Mixed Signal I/O
NMI	Non-maskable Interrupt
NVIC	Nested Vectored Interrupt Controller
NVM	Non-volatile Memory
PA	Power Amplifier
PKC	Public Key Cryptography
PLL	Phase Locked Loop
PMU	Power Management Unit
POR	Power-on Reset
PSC	Power State Controller
QSPI	Quad SPI
RBX	Row boundary crossing
RNG	RING Oscillator
RTC	Real-Time Counters
SCL	Serial Clock Line
SDA	Serial Data Line
SHS	Secure Hash Standard
SiP	System-in-Package
SNR	Signal to Noise Ratio
SoC	System-on-Chip
SWD	Serial Wire Debug
SWO	Serial Wire Output
TPIU	Trace Port Interface Unit
TPMS	Tire Pressure Monitor System
TRNG	True Random Number Generator

Name	Description
Typ	Typical
ULP	Ultra Low Power
VIO	I/O Voltage
W1C	Write 1 to Clear
XO	Crystal Oscillator

12 Reference Documents

The following documents are referred in this datasheet:

Table 12-1 Reference documents

Documents	Description
GR5525 Hardware Design Guidelines	Introduce GR5525 hardware design.
GR5525 Developer User Guide	Introduce GR5525 SDK and its related resources to developers.

13 Legal and Contact Information

Copyright © 2024 Shenzhen Goodix Technology Co., Ltd. All rights reserved.

Any excerption, backup, modification, translation, transmission or commercial use of this document or any portion of this document, in any form or by any means, without the prior written consent of Shenzhen Goodix Technology Co., Ltd. is prohibited.

Trademarks and Permissions

GOODIX and other Goodix trademarks are trademarks of Shenzhen Goodix Technology Co., Ltd. All other trademarks and trade names mentioned in this document are the property of their respective holders.

Disclaimer

Information contained in this document is intended for your convenience only and is subject to change without prior notice. It is your responsibility to ensure its application complies with technical specifications.

Shenzhen Goodix Technology Co., Ltd. (hereafter referred to as “Goodix”) makes no representation or guarantee for this information, express or implied, oral or written, statutory or otherwise, including but not limited to representation or guarantee for its application, quality, performance, merchantability or fitness for a particular purpose. Goodix shall assume no responsibility for this information and relevant consequences arising out of the use of such information.

Without written consent of Goodix, it is prohibited to use Goodix products as critical components in any life support system. Under the protection of Goodix intellectual property rights, no license may be transferred implicitly or by any other means.

Shenzhen Goodix Technology Co., Ltd.

Headquarters: Floor 13, Phase B, Tengfei Industrial Building, Futian Free Trade Zone, Shenzhen, China

TEL: +86-755-33338828 Zip Code: 518000

Website: www.goodix.com

14 Revision History

Table 14-1 Revision history

Version	Date	Description
1.0	2023-09-20	Initial release
1.1	2024-06-06	Added "Tape and Reel Information".
1.2	2024-09-02	Deleted the GR5525IGNI SoC.