Cortex®-M4F 32-bit Microcontroller with FPU, 200 MHz, Up to 2048 KB Dual Bank Code Flash, 32 KB Data Flash, 256 KB SRAM, 3-Phase PWM, 12-bit ADC for Motor Control Application

UM Rev. 1.00

Introduction

The A34G43x series is a 32-bit microcontroller suitable for applications requiring high-performance processing, such as electric motors.

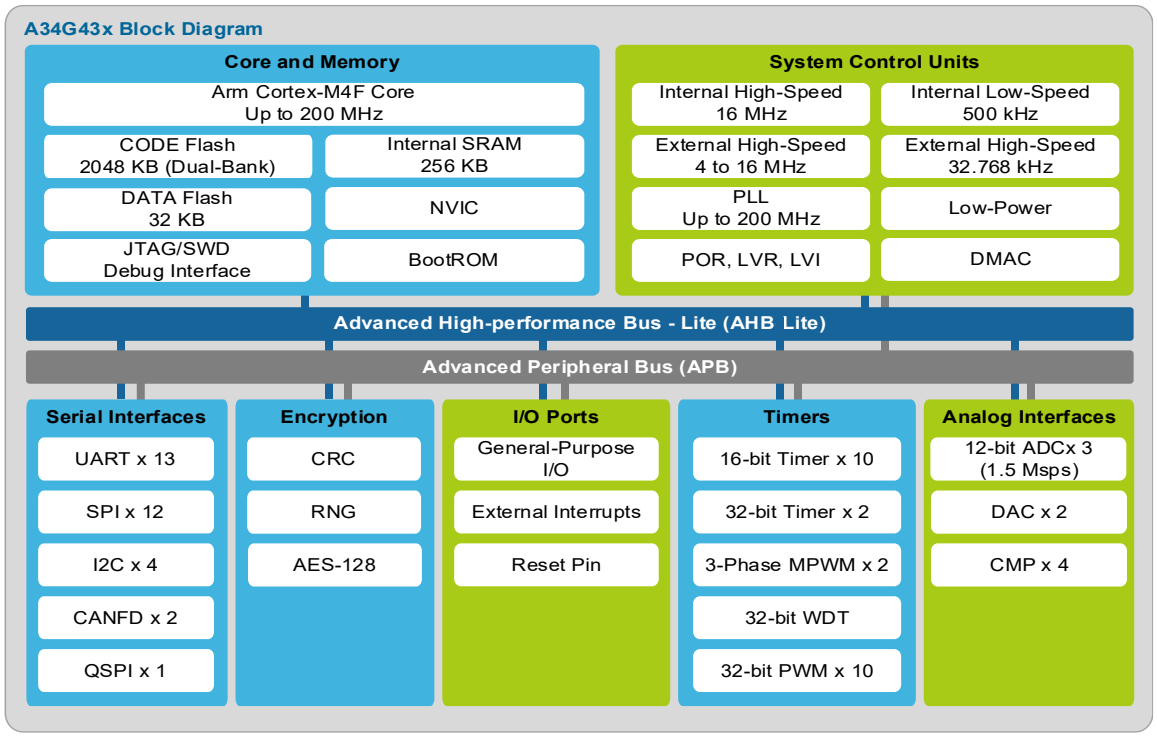
The A34G43x features an Arm® Cortex®-M4 with FPU, a high-performance 32-bit core, along with a variety of peripheral devices for motor control. It is equipped with two three-phase PWM generators capable of controlling two inverter motors simultaneously. Additionally, an individual mode optimized for Induction Heating (IH) has been added.

The A34G43x includes three high-speed 12-bit Analog-to-Digital Converters (ADCs), capable of processing motor driving information from 24 analog channels. With this functionality, the A34G43x can control either two inverter motors or one inverter motor along with a Power Factor Correction (PFC) process simultaneously.

The A34G43x offers various communication interfaces, including UART, SPI, I2C, QSPI, and CANFD modules. enabling seamless communication with external devices.

The A34G43x is equipped with Dual-Bank flash memory for versatile applications. These two banks can function either as a unified flash memory or operate independently. For example, one flash memory bank can be used as an upgrade buffer while the other executes a code. This feature enables a swift transition between applications in each bank, making the A34G43x ideal for upgrade purposes such as Over-The-Air (OTA) programming.

Figure 1. A34G43x Block Diagram



Reference Document

- Document ‘ID061113’ is provided by Arm and contains information on Cortex-M4F.
- A34G43x datasheet is provided by ABOV and available at www.abovsemi.com.

Contents

Introduction 1
Reference Document 2
1. Description 31
1.1 Product Category Definition 31
1.2 Availability of Peripherals 32
2. Pinouts and Pin Descriptions 40
2.1 Pinouts 40
2.1.1 A34G43xYL (120-LQFP) 40
2.1.2 A34G43xVL (100-LQFP) 41
2.1.3 A34G43xRL (64-LQFP) 42
2.2 Pin Description 43
3. System and Memory Overview 54
3.1 System Architecture 54
3.1.1 Cortex-M4F Core 55
3.1.2 Floating Point Unit (FPU) 55
3.1.3 Interrupt Controller 56
3.2 Memory Organization 61
3.2.1 Introduction 61
3.2.2 Memory Map and Register Boundary Address 61
3.3 Memory Map 63
3.3.1 Bit-Banding 63
3.3.2 Embedded SRAM 64
3.3.3 Flash Memory Overview 64
3.4 Boot Configuration 65
4. System Control Unit (SCU) 66
4.1 SCU Introduction 66
4.2 SCU Main Features 66
4.3 SCU Functional Description 67
4.3.1 SCU Block Diagram 67
4.3.2 SCU Pin and Type 67
4.4 Clocks 68
4.4.1 Overview 68
4.4.2 Clock Tree Configuration 70
4.4.3 HSE Clock 71
4.4.4 HSI Clock 71
4.4.5 PLL Clock 72
4.4.5.1 PLL Output Frequency Settings 73
4.4.5.2 Example PLL Clock Change 74
4.4.6 LSE Clock 75
4.4.7 LSI Clock 75
4.4.8 Two Main Operating Clocks: HCLK and PCLK 76
4.4.8.1 HCLK Clock 76
4.4.8.2 PCLK Clock 76
4.4.9 Configuration of Miscellaneous Clocks 77
4.4.10 Peripheral Clocks 78
4.4.11 Clock Monitoring 79
4.4.11.1 Detection Range 80

4.4.11.2 Clock Monitoring Functions and Clock Sources 81
4.4.11.3 MCLK Monitoring 82
4.4.11.4 HSE Monitoring 85
4.4.11.5 LSE Monitoring 86
4.4.12 Clock Configuration Procedure 87
4.5 Reset 90
4.5.1 Overview 90
4.5.2 Reset Tree Configuration 91
4.5.3 Cold Reset 92
4.5.4 Warm Reset 93
4.5.5 LVD Reset (LVR) 94
4.5.6 nRESET Pin Reset 94
4.5.7 Watchdog Timer Reset 95
4.5.8 Software Reset 95
4.5.9 HSE Fail Reset 95
4.5.10 LSE Fail Reset 95
4.5.11 CPU Request Reset 96
4.5.12 CFMC Bank Reset 96
4.6 Power Supply 97
4.6.1 LVI Block Diagram 98
4.6.2 LVR Block Diagram 99
4.6.3 POR, LVR 99
4.7 Operation Modes 100
4.7.1 Overview 100
4.7.2 Transition of Operation Mode 101
4.7.3 RUN Mode 102
4.7.4 SLEEP Mode 102
4.7.5 DEEP-SLEEP (STOP) Mode 103
4.7.6 Operation Mode Summary 105
4.8 SCU Registers 106
4.8.1 CHIPCONFIG_VENDORID: Vendor ID Register 109
4.8.2 CHIPCONFIG_CHIPID: Chip ID Register 109
4.8.3 CHIPCONFIG_REVNR: Revision Number Register 110
4.8.4 CHIPCONFIG2_CHIPID2: CHIP ID 2 Register 110
4.8.5 SCU_SMR: System Mode Register 111
4.8.6 SCU_SRCR: System Reset Control Register 112
4.8.7 SCU_WUER: Wake-up Source Enable Register 113
4.8.8 SCU_WUSR: Wake-up Source Status Register 114
4.8.9 SCU_RSER: Reset Source Enable Register 115
4.8.10 SCU_RSSR: Reset Source Status Register 116
4.8.11 SCU_PRER1: Peripheral Reset Enable Register 1 118
4.8.12 SCU_PRER2: Peripheral Reset Enable Register 2 120
4.8.13 SCU_PER1: Peripheral Enable Register 1 122
4.8.14 SCU_PER2: Peripheral Enable Register 2 124
4.8.15 SCU_PCER1: Peripheral Clock Enable Register 1 126
4.8.16 SCU_PCER2: Peripheral Clock Enable Register 2 128
4.8.17 SCU_CSCR: Clock Source Control register 130
4.8.18 SCU_SCCR: System Clock Control Register 131
4.8.19 SCU_CMR: Clock Monitoring Register 132
4.8.20 SCU_COR: Clock Output Register 133

4.8.21 SCU_NMICR: NMI Control Register 134
4.8.22 SCU_NMISR: NMI Status Register 135
4.8.23 SCU_PLLCON: PLL Control Register 136
4.8.24 SCU_VDCCON: VDC Control Register 137
4.8.25 SCU_LVICR: LVI (Low-Voltage Indicator) Control Register 138
4.8.26 SCU_LVISR: LVI (Low-Voltage Indicator) Status Register 139
4.8.27 SCU_LVRCR: LVR (Low-Voltage Detect Reset) Control Register 140
4.8.28 SCU_EOSCR: External Oscillator Control Register 141
4.8.29 SCU_MCCR1: Miscellaneous Clock Control Register 1 142
4.8.30 SCU_MCCR3: Miscellaneous Clock Control Register 3 143
4.8.31 SCU_MCCR4: Miscellaneous Clock Control Register 4 144
4.8.32 SCU_MCCR5: Miscellaneous Clock Control Register 5 145
4.8.33 SCU_MCCR6: Miscellaneous Clock Control Register 6 146
4.8.34 SCU_MCCR7: Miscellaneous Clock Control Register 7 147
4.8.35 SCU_MCCR8: Miscellaneous Clock Control Register 8 148
4.8.36 SCU_SYSTEN: System Access Enable Register 149
4.8.37 SCU_SPSCR: SRAM Power Sequence Control Register 150
4.8.38 SCU_SPSSR: SRAM Power Sequence Status Register 151
4.8.39 SCU_SPSDR: SRAM Power Sequence Delay Register 151
4.8.40 SCU_PRER3: Peripheral Reset Enable Register 3 152
4.8.41 SCU_PER3: Peripheral Enable Register 3 154
4.8.42 SCU_PCER3: Peripheral Clock Enable Register 3 156
4.8.43 SCU Register Map Summary 158
5. Port Control Unit (PCU) and General-Purpose I/O (GPIO) 164
5.1 Introduction 164
5.1.1 PCU Introduction 164
5.1.2 GPIO Introduction 164
5.2 Main Features 165
5.2.1 PCU Features 165
5.2.2 GPIO Features 165
5.3 GPIO Functional Description 166
5.3.1 GPIO Pins and Internal Signals 171
5.3.2 I/O Port Control Registers 171
5.3.3 I/O Port Data Registers 172
5.3.4 I/O Data Bitwise Handling 172
5.3.5 I/O Alternate Function 173
5.3.6 Interrupt Functionality 177
5.3.7 Output Configuration 178
5.3.8 Input Configuration 179
5.3.9 Alternate Function Configuration 180
5.3.10 Debouncing Functionality 181
5.4 GPIO Registers 182
5.4.1 Pn_MR1: Port n MUX Control Register 1 183
5.4.2 Pn_MR2: Port n MUX Control Register 2 184
5.4.3 PC_MR1: Port C MUX Control Register 1 185
5.4.4 PC_MR2: Port C MUX Control Register 2 186
5.4.5 Pn_CR: Port n Control Register 187
5.4.6 Pn_PRCR: Port n Pull-Up/Pull-Down Resistor Control Register 187
5.4.7 PC_PRCR: Port C Pull-Up/Pull-Down Resistor Control Register 188
5.4.8 Pn_DER: Port n Debounce Enable Register 188

5.4.9 Pn_STR: Port n Strength Configuration Register 189
5.4.10 Pn_IER: Port n Interrupt Enable Register 190
5.4.11 Pn_ISR: Port n Interrupt Status Register 191
5.4.12 Pn_ICR: Port n Interrupt Control Register 192
5.4.13 Pn_ODR: Port n Output Data Register 192
5.4.14 Pn_IDR: Port n Input Data Register 193
5.4.15 Pn_BSR: Port n Set/Reset Register 194
5.4.16 Pn_BCR: Port n Reset Register 195
5.4.17 PCU_RSTDBCR: Pin Reset Debounce Control Register 196
5.4.18 PCU_PORTEN: Port Access Enable Register 197
5.4.19 PCU and GPIO Register Map Summary 198
6. Embedded Flash Memory 200
6.1 Introduction 200
6.2 Embedded Flash Memory Main Features 201
6.3 Flash Functional Description 203
6.3.1 Flash Memory Organization 203
6.3.1.1 Code Flash Memory 203
6.3.1.2 Data Flash Memory 204
6.3.2 Read Access Latency 205
6.3.2.1 Code Flash Memory Wait Time 205
6.3.2.2 Data Flash Memory Wait Time 206
6.3.3 Usage of Cache 206
6.3.4 Flash Memory Program and Erase Operations 207
6.3.4.1 Access Control for Writing and Erasing Flash Memory 207
6.3.5 Flash Memory Erase Sequences 208
6.3.5.1 Page Erase 208
6.3.5.2 Sector Erase 209
6.3.5.3 Bulk Erase 210
6.3.6 Flash Memory Programming Sequences 211
6.3.6.1 Standard Programming 211
6.4 Flash Memory Protection 213
6.4.1 Read Protection 213
6.4.1.1 Read Protection: UNPROT (Unprotection) 214
6.4.1.2 Read Protection: LVL1 (Level-1) 214
6.4.1.3 Read Protection: LVL2 (Level-2) 215
6.4.1.4 Read Protection Setting 215
6.4.1.5 Read Protection Removal 215
6.4.1.6 Password Settings 216
6.4.2 Write Protection 217
6.4.2.1 Write Protection for Code Flash Memory 0 218
6.4.2.2 Write Protection for Code Flash Memory 1 220
6.4.2.3 Write Protection for Data Flash Memory 222
6.4.2.4 Write Protection with Code Flash Memory Bank Swap 223
6.4.3 Securable Memory Area (LVL1, LVL2) 224
6.4.4 Disabling Core Debug Access (LVL2) 224
6.5 Flash Memory CRC Check 225
6.6 Bank Swap of Code Flash Memory 226
6.6.1 Bank Swap of Code Flash Memory 226
6.6.2 Bank Swap Operation 227
6.6.3 Limitations and Recommendations 228

6.6.4 Sequence of Bank Mode Enable or Swap 229
6.7 Read-While-Write (RWW) 232
6.8 Error Correction Code (ECC) 234
6.8.1 Block Diagram 234
6.8.2 ECC Interrupt Service Routine 235
6.8.3 Sequence of ECC 236
6.9 Flash Memory Interrupts 237
6.10 Code Flash Memory Controller Registers 238
6.10.1 CFMC_CONF: Code Flash Control Register 240
6.10.2 CFMC_FLSKEY: Code Flash Access Key Register 241
6.10.3 CFMC_INFOKEY: Code Flash User Info Access Key Register 242
6.10.4 CFMC_FLS0PROT: Code Flash 0 Protection Register 243
6.10.5 CFMC_FLS1PROT: Code Flash 1 Protection Register 244
6.10.6 CFMC_INFOPROT: Code Flash User Info Protection Register 245
6.10.7 CFMC_CTRL: Code Flash Access Control Register 246
6.10.8 CFMC_STAT: Code Flash Access Status Register 248
6.10.9 CFMC_READPROT: Code Flash Read Protection Register 250
6.10.10 CFMC_PWIN: Code Flash Password Input Register 251
6.10.11 CFMC_CHKCTRL: Code Flash CRC Check Control Register 252
6.10.12 CFMC_CHKDOUT: Code Flash CRC Check Data Output Register 253
6.10.13 CFMC_CHKSADDR: Code Flash CRC Check Start Address Register 253
6.10.14 CFMC_CHKEADDR: Code Flash CRC Check End Address Register 254
6.10.15 CFMC_WTOUT: Code Flash Write Timeout Register 254
6.10.16 CFMC_BANK: Code Flash Bank Control Register 255
6.10.17 CFMC_BANKRST: Code Flash Bank Reset Register 256
6.10.18 CFMC_ECCEADDR: Code Flash ECC Error Address Register 256
6.10.19 CFMC_ECCEDATA0: Code Flash ECC Error Data0 Register 257
6.10.20 CFMC_ECCEDATA1: Code Flash ECC Error Data1 Register 257
6.10.21 CFMC_ECCEDATA2: Code Flash ECC Error Data2 Register 258
6.10.22 CFMC_ECCEDATA3: Code Flash ECC Error Data3 Register 258
6.10.23 CFMC_ECCEPARITY: Code Flash ECC Error Parity Register 259
6.10.24 CFMC_PWPRST: Code Flash Password Preset Register 259
6.10.25 CFMC Register Map Summary 260
6.11 Data Flash Registers 262
6.11.1 DFMC_CONF: Data Flash Control Register 263
6.11.2 DFMC_FLSKEY: Data Flash Access Key Register 264
6.11.3 DFMC_FLSPROT: Data Flash Protection Register 265
6.11.4 DFMC_CTRL: Data Flash Access Control Register 266
6.11.5 DFMC_STAT: Data Flash Access Status Register 267
6.11.6 DFMC_CHKCTRL: Code Flash CRC Check Control Register 268
6.11.7 DFMC_CHKDOUT: Data Flash CRC Check Data Output Register 268
6.11.8 DFMC_CHKSADDR: Data Flash CRC Check Start Address Register 269
6.11.9 DFMC_CHKEADDR: Data Flash CRC Check End Address Register 269
6.11.10 DFMC_WTOUT: Data Flash Write Timeout Register 270
6.11.11 DFMC_ECCEADDR: Data Flash ECC Error Address Register 270
6.11.12 DFMC_ECCEDATA: Data Flash ECC Error Data Register 271
6.11.13 DFMC_ECCEPARITY: Data Flash ECC Error Parity Register 271
6.11.14 DFMC Register Map Summary 272
7. Direct Memory Access Controller (DMA) 274
7.1 DMA Introduction 274

7.1.1 Modules Supported by DMA 275
7.2 DMA Functional Description 277
7.2.1 DMA Transfer Timing from Peripheral to Memory 278
7.2.2 DMA Transfer Timing from Memory to Peripheral 279
7.2.3 DMA Transfer 279
7.3 DMA Registers 280
7.3.1 DMAn_CR: DMA Channel n Control Register 281
7.3.2 DMAn_SR: DMA Channel n Status Register 283
7.3.3 DMAn_PAR: DMA Channel n Peripheral Address Register 284
7.3.4 DMAn_MAR: DMA Channel n Memory Address Register 285
7.3.5 DMA Register Map Summary 286
8. Watchdog Timer (WDT) 287
8.1 WDT Introduction 287
8.2 WDT Main Features 287
8.3 WDT Functional Description 288
8.3.1 WDT Block Diagram 288
8.3.2 Enabling WDT 288
8.3.3 Controlling Down Counter 289
8.3.4 Interrupt Feature 290
8.3.4.1 Interrupt Mode Operation Timing Diagram 290
8.3.5 How to Program Watchdog Timeout 291
8.3.5.1 Basic Reset Operation 291
8.3.6 Debug Mode 291
8.4 WDT Registers 292
8.4.1 WDT_LR: WDT Load Register 293
8.4.2 WDT_CNT: WDT Current Count Register 294
8.4.3 WDT_CON: WDT Control Register 295
8.4.4 WDT_AEN: WDT Access Enable Register 296
8.4.5 WDT Register Map Summary 297
9. Pulse-Width Modulation (PWM) Generator 298
9.1 PWM Generator Introduction 298
9.2 PWM Generator Main Features 298
9.3 PWM Generator Functional Description 299
9.3.1 Block Diagram 299
9.3.2 Pins and Internal Signals 300
9.3.3 Setting Input Clock of PWM (Pulse-Width Modulation) Generator 301
9.3.4 Setting Period and PWM Output Signal 302
9.3.5 Synchronization of PWM Channels 304
9.4 PWM Interrupts 305
9.4.1 Interrupt Generation 305
9.5 PWM Registers 306
9.5.1.1 PWMn_CTRL: PWM Timer n Control Register 307
9.5.1.2 PWMn_CNT: PWM Timer n Counter Register 308
9.5.1.3 PWMn_PER: PWM Timer n Counter Period Register 308
9.5.1.4 PWMn_CMP: PWM Timer n Compare Register 309
9.5.1.5 PWM_PRS0: PWM 0 to 4 Prescaler Register 309
9.5.1.6 PWM_PRS1: PWM 5 to 9 Prescaler Register 310
9.5.1.7 PWM Register Map Summary 311
10. 16-bit Timer 312
10.1 16-bit Timer Introduction 312

10.2 16-bit Timer Main Features 313
10.3 16-bit Timer Functional Description 314
10.3.1 Block Diagram 314
10.3.2 Pins and Internal Signals 315
10.3.3 Clock Selection 317
10.3.3.1 Peripheral System Clock (PCLK) 317
10.3.3.2 Timer External Clock (SCU_MCCR3 Register) 317
10.3.3.3 Input to Pin TnIO for Timer Clock Source 318
10.3.4 Time-Base Unit 319
10.3.4.1 Prescaler Description 320
10.3.5 Periodic Mode 321
10.3.5.1 Operation According to TIMERn_GRA and TIMERn_GRB Settings 322
10.3.5.2 Operation According to UAO, OUTPOL, and STARTLVL 323
10.3.5.3 Additional Precautions Over Counter Mode Operation 325
10.3.5.4 Counter Mode Register Setting Sequence 325
10.3.6 Capture Mode 326
10.3.6.1 Capture Mode Register Setting Sequence 327
10.3.7 PWM Mode 328
10.3.7.1 PWM Mode Operation Characteristics and Precautions 329
10.3.7.2 PWM Mode Register Setting Sequence 329
10.3.8 One-Shot Mode 330
10.3.8.1 One-Shot Mode Operation Characteristics and Precautions 331
10.3.8.2 One-Shot Mode Register Setting Sequence 332
10.3.9 ADC Trigger 333
10.3.9.1 ADC Trigger Operation Characteristics and Precautions 335
10.3.9.2 Register Setting Sequence for ADC Trigger Operation 335
10.3.10 Timer Synchronization 336
10.3.10.1 Timer Delayed Synchronization 338
10.3.10.2 Timer Synchronization Operation Characteristics and Precautions 339
10.3.10.3 Register Setting Sequence for Synchronous Function Operation 341
10.3.11 Direction Bit Output 342
10.3.12 Debug Mode 342
10.3.13 Interrupts 342
10.4 Timer Registers 343
10.4.1 TIMERn_CR1: Timer/Counter n Control Register 1 344
10.4.2 TIMERn_CR2: Timer/Counter n Control Register 2 346
10.4.3 TIMERn_PRS: Timer/Counter n Prescaler Register 347
10.4.4 TIMERn_GRA: Timer/Counter n General Data Register A 348
10.4.5 TIMERn_GRB: Timer/Counter n General Data Register B 349
10.4.6 TIMERn_CNT: Timer/Counter n Count Register 350
10.4.7 TIMERn_SR: Timer/Counter n Status Register 350
10.4.8 TIMERn_IER: Timer/Counter n Interrupt Enable Register 351
10.4.9 TIMERn_TRGPNT: Timer/Counter n Trigger Point Register 352
10.4.10 TIMERn_SYNC: Timer/Counter n Sync Configuration Register 353
10.4.11 Timer Register Map Summary 354
11. Free-Run Timer (FRT) 355
11.1 FRT Introduction 355
11.2 FRT Main Features 355
11.3 FRT Functional Description 356

11.3.1 FRT Block Diagram 356
11.3.2 FRT Clock Selection 356
11.3.3 FRT Operation Mode 357
11.3.3.1 Match Interrupt Operation 357
11.3.3.2 Overflow Interrupt Operation 358
11.3.3.3 Register Setting Procedure by FRT Operation Mode 359
11.3.4 FRT Low-Power Modes 360
11.3.4.1 Description of the Steps of Low-Power Mode 360
11.4 FRT Registers 361
11.4.1 FRTn_CTRL: FRT n Control Register 362
11.4.2 FRTn_MCNT: FRT n Match Counter Register 362
11.4.3 FRTn_CNT: FRT n Counter Register 363
11.4.4 FRTn_STAT: FRT n Status Register 363
11.4.5 FRT Register Map Summary 364
12. Universal Asynchronous Receiver/Transmitter (UART) 365
12.1 UART Introduction 365
12.2 UART Main Features 366
12.2.1 UART Implementation 366
12.3 UART Functional Description 367
12.3.1 UART Block Diagram 367
12.3.2 UART Pins and Signals 368
12.3.2.1 UART Bidirectional Communications 368
12.3.3 UART Character Description 369
12.3.4 UART Transmitter 372
12.3.4.1 Data Format 372
12.3.4.2 Transmit Interrupt 373
12.3.4.3 Character Transmission 373
12.3.4.4 Configurable Stop Bits 374
12.3.4.5 Character Transmission Procedure 375
12.3.4.6 Single Byte Communication 376
12.3.4.7 Break Condition 376
12.3.4.8 Inter-Frame Delay for Data Transmission 376
12.3.5 UART Receiver 377
12.3.5.1 Receive Data Sampling Timing 377
12.3.5.2 Start Bit Detection 378
12.3.5.3 Data Bit Sampling 380
12.3.5.4 Data Sampling Strategy 381
12.3.5.5 Character Reception 382
12.3.5.6 Break Condition 382
12.3.5.7 Overrun Error 382
12.3.5.8 Framing Error 383
12.3.5.9 Parity Error 383
12.3.5.10 Configurable Stop Bits During Reception 383
12.3.6 UART Baud-Rate Generation 384
12.3.7 Tolerance of UART Receiver to Clock Deviation 386
12.3.8 UART Parity Control 387
12.3.8.1 Even Parity 387
12.3.8.2 Odd Parity 387
12.3.8.3 Stick Parity 388
12.3.8.4 Various Configurations of Parity Bit 388

12.3.8.5 Parity Checking in Reception 388
12.3.8.6 Parity Generation in Transmission 388
12.3.9 Continuous Communication Using DMA and UART 389
12.3.9.1 Transmission Using DMA 390
12.3.9.2 Reception Using DMA 391
12.3.9.3 Error Flagging and Interrupt Generation in Communication 392
12.4 UART Interrupts 393
12.5 UART Registers 394
12.5.1 UARTn_RBR: UART n Receive Data Buffer Register 395
12.5.2 UARTn_THR: UART n Transmit Data Hold Register 395
12.5.3 UARTn_IER: UART n Interrupt Enable Register 396
12.5.4 UARTn_IIR: UART n Interrupt ID Register 397
12.5.5 UARTn_LCR: UART n Line Control Register 399
12.5.6 UARTn_DCR: UART n Data Control Register 401
12.5.7 UARTn_LSR: UART n Line Status Register 402
12.5.8 UARTn_ELSR: UART n Extended Line Status Register 405
12.5.9 UARTn_BDR: UART n Baud-Rate Divisor Latch Register 408
12.5.10 UARTn_BFR: UART n Baud-Rate Fraction Counter Register 408
12.5.11 UARTn_IDTR: UART n Inter-Frame Delay Time Register 409
12.5.12 UARTn_FCR: UART n FIFO Control Register 410
12.5.13 UARTn_FRBR: UART n FIFO Receive Buffer Register 411
12.5.14 UART Register Map Summary 412
13. Serial Peripheral Interface (SPI) 414
13.1 SPI Introduction 414
13.2 SPI Main Features 415
13.3 SPI Implementation 416
13.4 SPI Functional Description 417
13.4.1 SPI Block Diagram 417
13.4.2 SPI Pins and Internal Signals 418
13.4.3 Communications between Master and Slave 420
13.4.3.1 Full-Duplex Communication 420
13.4.3.2 Simplex Communication 420
13.4.4 Slave Selection Pin 421
13.4.5 Communication Formats 422
13.4.5.1 Clock Phase and Polarity Controls 422
13.4.5.2 Data Frame Format 425
13.4.6 Configuration of SPI 426
13.4.7 Procedure for Enabling SPI 426
13.4.8 Data Transmission and Reception Procedures 427
13.4.8.1 Sequence Handling 427
13.4.8.2 Procedure for Disabling SPI 429
13.4.8.3 Data Packing 429
13.4.8.4 DMA Handshake 430
13.4.8.5 Communication Diagrams with DMA 431
13.4.9 SPI Status Flags 433
13.4.9.1 Transmit Buffer Empty Flag (TRDY) 433
13.4.9.2 Transmit/Receive Operation Flag (SBUSY) 433
13.4.9.3 Receive Buffer Ready Flag (RRDY) 433
13.4.9.4 SS Signal Status Flag (SSON) 433
13.4.9.5 Rising or Falling Edge of SS Signal Detect Flag (SSDET) 433

13.4.9.6 DMA Transmit Operation Complete Flag (DMA to SPI, TXDMAF) 433
13.4.9.7 DMA Receive Operation Complete Flag (SPI to DMA, RXDMAF) 433
13.4.10 SPI FIFO Status Flags 434
13.4.10.1 Receive FIFO Trigger Level Interrupt Flag (RXFTLIF) 434
13.4.10.2 Transmit FIFO Trigger Level Interrupt Flag (TXFTLIF) 434
13.4.10.3 Receive FIFO Full Interrupt Flag (RXFFIF) 434
13.4.10.4 Transmit FIFO Full Interrupt Flag (TXFFIF) 435
13.4.10.5 Receive FIFO Not Empty Interrupt Flag (RXFNEIF) 435
13.4.10.6 Transmit FIFO Empty Interrupt Flag (TXFEIF) 435
13.4.10.7 Receive FIFO Empty Status (RXFES) 435
13.4.10.8 Transmit FIFO Empty Status (TXFES) 435
13.4.11 SPI Error Flags 436
13.4.11.1 Overrun Flag (OVRF) 436
13.5 SPI Registers 437
13.5.1 SPIn_TDR: SPI n Transmit Data Register 438
13.5.2 SPIn_RDR: SPI n Receive Data Register 438
13.5.3 SPIn_CR: SPI n Control Register 439
13.5.4 SPIn_SR: SPI n Status Register 441
13.5.5 SPIn_BR: SPI n Baud-Rate Register 443
13.5.6 SPIn_EN: SPI n Enable Register 443
13.5.7 SPIn_LR: SPI n Delay Length Register 444
13.5.8 SPIn_FCR: SPI n FIFO Control Register 445
13.5.9 SPIn_FTDR: SPI n FIFO Transmit Data Register 447
13.5.10 SPIn_FRDR: SPI n FIFO Receive Data Register 447
13.5.11 SPI Registers Map Summary 448
14. Inter-Integrated Circuit (I2C) 449
14.1 I2C Introduction 449
14.2 I2C Main Features 449
14.3 I2C Implementation 450
14.4 I2C Functional Description 451
14.4.1 I2C Block Diagram 451
14.4.2 I2C Pins and Signals 452
14.4.3 I2C Mode Selection 452
14.4.4 I2C Initialization 453
14.4.4.1 Enabling and Disabling Peripherals 453
14.4.4.2 Noise Canceller 453
14.4.4.3 Using Noise Canceller in I2C Module 454
14.4.5 I2C Software Reset 454
14.4.6 I2C Module Protocol 455
14.4.6.1 I2C Bit Transfer 455
14.4.6.2 START, Repeated START, and STOP 456
14.4.6.3 Acknowledge 457
14.4.6.4 Synchronization 458
14.4.6.5 Arbitration 459
14.4.6.6 Data Transfer 460
14.4.6.7 Master Transmit Timing 461
14.4.6.8 Master Receive Timing 462
14.4.6.9 Slave Transmit Timing 463
14.4.6.10 Slave Receive Timing 464
14.4.7 I2C Slave Mode 465

14.4.7.1 Slave Transmitter 465
14.4.7.2 Slave Receiver 467
14.4.8 I2C Master Mode 469
14.4.8.1 Master Transmitter 469
14.4.8.2 Master Receiver 472
14.5 I2C Low-Power Modes 475
14.6 I2C Interrupts 475
14.7 I2C Registers 476
14.7.1 I2Cn_DR: I2C n Data Register 477
14.7.2 I2Cn_SR: I2C n Status Register 478
14.7.3 I2Cn_SAR: I2C n Slave Address Register 479
14.7.4 I2C_CR: I2C n Control Register 480
14.7.5 I2Cn_SCLL: I2C n SCL Low Duration Register 481
14.7.6 I2Cn_SCLH: I2C n SCL High Duration Register 482
14.7.7 I2C_SDH: I2C n SDA Hold Register 483
14.7.8 I2C Register Map Summary 484
15. Quad Serial Peripheral Interface (QSPI) 485
15.1 QSPI Introduction 485
15.2 QSPI Main Features 486
15.3 QSPI Functional Description 487
15.3.1 QSPI Block Diagram 487
15.3.2 QSPI Pins and Signals 488
15.3.3 QSPI Transactions 489
15.3.3.1 QSPI Output Behavior 491
15.3.3.2 QSPI Input Behavior 493
15.3.3.3 Use of QSPI Data Lines 494
15.3.3.4 Command (Instruction) Phase 494
15.3.3.5 Address Phase 495
15.3.3.6 Mode Phase 495
15.3.3.7 Dummy Phase 496
15.3.3.8 Data Phase 497
15.3.3.9 Setting of Duration 498
15.3.4 QSPI Interface Protocol Modes 499
15.3.4.1 Single QSPI Mode 499
15.3.4.2 Dual QSPI Mode 499
15.3.4.3 Quad QSPI Mode 499
15.3.4.4 SDR Mode 500
15.3.4.5 DDR Mode 501
15.3.5 QSPI Transaction Execution 502
15.3.5.1 Executing Transactions Using Registers 502
15.3.5.2 Operation of the Interrupts and Status 503
15.3.5.3 Transactions with Data from Internal DMA Engine 505
15.3.6 QSPI DMA Mode of Operation 506
15.4 Memory Map 509
15.4.1 External Bus Space 509
15.4.2 Address Width of QSPI Space and QSPI Bus 509
15.4.3 Address Split 510
15.4.4 Memory Mapped Mode 511
15.4.5 Setting XIP Mode 512

15.4.6 Releasing XIP mode 514
15.5 QSPI Interrupts 515
15.6 QSPI Registers 516
15.6.1 QSPI_GCR: QSPI Global Control Register 518
15.6.2 QSPI_CLKCFGR: QSPI Clock Configuration Register 519
15.6.3 QSPI_MINCSHR: QSPI Minimum Chip Select High Register 520
15.6.4 QSPI_FSR: QSPI FIFO Status Register 520
15.6.5 QSPI_SR: QSPI Status Register 521
15.6.6 QSPI_ICR: QSPI Interrupt Control Register 522
15.6.7 QSPI_CMDR: QSPI Command Register 524
15.6.8 QSPI_CMDATR: QSPI Command Attributes Register 525
15.6.9 QSPI_AR: QSPI Address Register 526
15.6.10 QSPI_AATR: QSPI Address Attributes Register 527
15.6.11 QSPI_CSR: QSPI Chip Select Register 528
15.6.12 QSPI_DMYATR: QSPI Dummy Attributes Register 528
15.6.13 QSPI_XIPHOTR: QSPI XIP Hold Off Time Register 529
15.6.14 QSPI_XIPHOSR: QSPI XIP Hold Off Select Register 529
15.6.15 QSPI_XIPHOCR: QPSI XIP Hold Off Control Register 530
15.6.16 QSPI_DSZR: QSPI Data Size Register 530
15.6.17 QSPI_DATR: QSPI Data Attributes Register 531
15.6.18 QSPI_TDR: QSPI Transmit Data Register 532
15.6.19 QSPI_RDR: QSPI Receive Data Register 532
15.6.20 QSPI_MBR: QSPI Mode Bits Register 533
15.6.21 QSPI_MBATR: QSPI Mode Bits Attributes Register 534
15.6.22 QSPI_XIPCR: QSPI XIP Control Register 535
15.6.23 QSPI_XIPACCBNDR: QSPI XIP Access Boundary Register 536
15.6.24 QSPI_XIPSZR: QSPI XIP Size Register 536
15.6.25 QSPI_XIPWINAR: QSPI XIP Window Address Register 537
15.6.26 QSPI_XIPCMDR: QSPI XIP Command Register 537
15.6.27 QSPI_XIPCMDATR: QSPI XIP Command Attributes Register 538
15.6.28 QSPI_XIPBAR: QSPI XIP Base Address Register 539
15.6.29 QSPI_XIPAATR: QSPI XIP Address Attributes Register 540
15.6.30 QSPI_XIPCSR: QSPI XIP Chip Select Register 541
15.6.31 QSPI_XIPDMYATR: QSPI XIP Dummy Attributes Register 541
15.6.32 QSPI_XIPDATR: QSPI XIP Data Attributes Register 542
15.6.33 QSPI_XIPMBR: QSPI XIP Mode Bits Register 543
15.6.34 QSPI_XIPMBATR: QSPI XIP Mode Bits Attributes Register 544
15.6.35 QSPI_DMASTAR: QSPI DMA Start Address Register 545
15.6.36 QSPI_DMAFBLSR: QSPI DMA Master Fixed Burst Length Select Register 545
15.6.37 QSPI_DMASR: QSPI DMA Status Register 546
15.6.38 QSPI_DMABDCNTR: QSPI DMA Buffer Descriptor Count Register 546
15.6.39 QSPI_WPCR: QSPI Write Protect Control Register 547
15.6.40 QSPI_HTR: QSPI Hold Time Register 547
15.6.41 QSPI Register Map Summary 548
16. CAN with Flexible Data Rate (CANFD) 552
16.1 CANFD Introduction 552
16.2 CANFD Main Features 553
16.3 CANFD Functional Description 555
16.3.1 CANFD Block Diagram 555
16.3.2 CANFD Pins and Signals 556

16.3.3 CANFD Transceivers 557
16.3.4 Frame Type 558
16.3.5 Frame Buffer 560
16.3.5.1 Receive Buffer (RB) 561
16.3.5.2 Transmit Buffer (TB) 561
16.3.5.3 Frame Reception 562
16.3.5.4 Handling Frame Reception 563
16.3.5.5 Frame Transmission 564
16.3.6 Acceptance Filters (ACFM, ACFC) 566
16.3.7 Error Handling 568
16.3.8 Bus Off State 570
16.3.9 Extended Status and Error Report 571
16.3.9.1 Programmable Error Warning Limit 571
16.3.9.2 Arbitration Lost Capture (ALC) 572
16.3.9.3 Kind of Error (KOER) 573
16.3.9.4 Reception of All Data Frames (RBALL) 574
16.3.9.5 Transmit Status 575
16.3.10 Remote Mode 577
16.3.11 Bit Time 578
16.3.11.1 Definitions 578
16.3.11.2 Configuration 580
16.3.11.3 Flexible Data Bit Rate Switching and Sample Point 582
16.3.11.4 TDC and RDC 583
16.3.12 Extended Features 585
16.3.12.1 Re-Transmission and Re-Arbitration Limitation 585
16.3.12.2 Listen Only Mode (LOM) 586
16.3.12.3 Loop Back Mode (LBMI, LBME) 588
16.3.12.4 Transceiver Standby Mode 590
16.3.12.5 Error Counter Reset 590
16.3.13 Software Reset 591
16.4 CANFD Interrupts 594
16.5 CANFD Registers 595
16.5.1 CANFDn_VNCFGR: CANFD n Version and Node Configuration Register 597
16.5.2 CANFDn_ACBTR: CANFD n Arbitration and Classic Bit Timing Register 597
16.5.3 CANFDn_FDBTR: CANFD n Flexible Data Bit Timing Register 598
16.5.4 CANFDn_PTDCR: CANFD n Prescaler and Transmitter Delay Compensation Register 599
16.5.5 CANFDn_ISR: CANFD n Interrupt Status Register 601
16.5.6 CANFDn_IER: CANFD n Interrupt Enable Register 603
16.5.7 CANFDn_TSR: CANFD n Transmit Status Register 604
16.5.8 CANFDn_CSR: CANFD n Control and Status Register 605
16.5.9 CANFDn_ERR: CANFD n Error Register 609
16.5.10 CANFDn_AFCR: CANFD n Acceptance Filter Control Register 610
16.5.11 CANFDn_AFCIR: CANFD n Acceptance Filter Code Identifier Register 611
16.5.12 CANFDn_AFCFR: CANFD n Acceptance Filter Code Format Register 612
16.5.13 CANFDn_AFCTR: CANFD n Acceptance Filter Code Type Register 614
16.5.14 CANFDn_AFMIR: CANFD n Acceptance Filter Mask Identifier Register 614
16.5.15 CANFDn_AFMFR: CANFD n Acceptance Filter Mask Format Register 615
16.5.16 CANFDn_AFMTR: CANFD n Acceptance Filter Mask Type Register 617
16.5.17 CANFDn_RBIR: CANFD n Receive Buffer Identifier Register 617

16.5.18 CANFDn_RBFR: CANFD n Receive Buffer Format Register 618
16.5.19 CANFDn_RBTR: CANFD n Receive Buffer Type Register 620
16.5.20 CANFDn_RBPDR: CANFD n Receive Buffer Payload Data Register 620
16.5.21 CANFDn_TBIR: CANFD n Transmit Buffer Identifier Register 621
16.5.22 CANFDn_TBFR: CANFD n Transmit Buffer Format Register 622
16.5.23 CANFDn_TBTR: CANFD n Transmit Buffer Type Register 624
16.5.24 CANFDn_TBPDR: CANFD n Transmit Buffer Payload Data Register 624
16.5.25 CANFDn_TSCFGR: CANFD n Transmit Stream Configuration Register 625
16.5.26 CANFDn_LFCIR: CANFD n LLC Frame Calculator Input Register 626
16.5.27 CANFDn_LFCOR: CANFD n LLC Frame Calculator Output Register 626
16.5.28 CANFD Register Map Summary 627
17. Motor Pulse Width Modulation (MPWM) 630
17.1 MPWM Introduction 630
17.2 MPWM Main Features 630
17.3 MPWM Functional Description 631
17.3.1 MPWM Block Diagram 631
17.3.2 Pins and Internal Signals 632
17.3.3 Time-Base Unit 633
17.3.4 Counter Modes 634
17.3.5 Symmetric and Asymmetric Mode 635
17.3.6 Output Control 637
17.3.7 PWM Modes 640
17.3.7.1 Motor PWM Mode 0/Up-Down Counter/2-Ch Symmetric Mode 641
17.3.7.2 Motor PWM Mode 0/Up-Down Counter/1-Ch Asymmetric Mode 642
17.3.7.3 Motor PWM Mode 0/Up-Down Counter/1-Ch Symmetric Mode 643
17.3.7.4 Motor PWM Mode 1/Up Counter 644
17.3.7.5 Normal PWM Mode/Up Counter 645
17.3.8 Complimentary Outputs and Dead-Time Insertion 646
17.3.8.1 Complimentary Outputs 646
17.3.8.2 Dead-Time Insertion 647
17.3.8.3 Special Case About Dead-Time 649
17.3.9 Duty/Period Update Timing 653
17.3.10 IRQ Interval 653
17.3.11 Direction Bit 653
17.3.12 ADC Synchronization 654
17.3.13 Forced Output Mode 655
17.3.14 Protection and Overcurrent 656
17.3.15 Interrupt Generation 657
17.4 MPWM Low Power Modes 658
17.5 MPWM Interrupts 658
17.6 MPWM Registers: Motor PWM and Normal PWM Modes 659
17.6.1 MPWMn_MR: MPWM n Mode Register 660
17.6.2 MPWMn_OLR: MPWM n Output Level Register 662
17.6.3 MPWMn_FOLR: MPWM n Forced Output Register 664
17.6.4 MPWMn_PRD: MPWM n Period Register 665
17.6.5 MPWMn_DUH: MPWM n Duty UH Register 665
17.6.6 MPWMn_DVH: MPWM n Duty VH Register 666
17.6.7 MPWMn_DWH: MPWM n Duty WH Register 666
17.6.8 MPWMn_DUL: MPWM n Duty UL Register 667
17.6.9 MPWMn_DVL: MPWM n Duty VL Register 667

17.6.10 MPWMn_DWL: MPWM n Duty WL Register 668
17.6.11 MPWMn_CR1: MPWM n Control Register 1 668
17.6.12 MPWMn_CR2: MPWM n Control Register 2 669
17.6.13 MPWMn_SR: MPWM n Status Register 670
17.6.14 MPWMn_IER: MPWM n Interrupt Enable Register 672
17.6.15 MPWMn_CNT: MPWM n Counter Register 673
17.6.16 MPWMn_DTR: MPWM n Dead-Time Register 674
17.6.17 MPWMn_PCR: MPWM n Protection Control Register 675
17.6.18 MPWMn_PSR: MPWM n Protection Status Register 676
17.6.19 MPWMn_OCR: MPWM Overcurrent Detection Register 677
17.6.20 MPWMn_OSR: MPWM Overcurrent Detection Status Register 678
17.6.21 MPWMn_ATRx: MPWM n ADC Trigger Counter Register 679
17.6.22 MPWM Register Map Summary 680
18. Analog-to-Digital Converter (ADC) 683
18.1 ADC Introduction 683
18.2 ADC Main Features 684
18.3 ADC Functional Description 686
18.3.1 ADC Block Diagram 686
18.3.2 ADC Pins and Signals 687
18.3.3 ADC Connectivity 689
18.3.3.1 ADC Internal Channel Wiring 689
18.3.3.2 ADC Input Signals and External Input Channels 690
18.3.4 ADC Clocks (ACLK) 691
18.3.4.1 ADC Module Control 691
18.3.4.2 ADC Clock Sources 692
18.3.5 ADC On-Off Control (ADEN) 693
18.3.5.1 Software Procedure to Enable ADC 693
18.3.5.2 Software Procedure to Disable ADC 693
18.3.6 Constraints when Writing the ADC Control Bits 694
18.3.6.1 When Writing ADC Control Bit 694
18.3.6.2 When Initializing ADC 694
18.3.7 Channel Selection 695
18.3.7.1 ADC Analog Input Channel 695
18.3.7.2 ADC Conversion Channel Selection 696
18.3.7.3 Register Access when Changing A/D Conversion Channel 697
18.3.8 Programmable Sampling Time 698
18.3.9 Single Mode Conversion (ADMOD[1:0] = ‘00’, SEQCNT[2:0] = 0) 699
18.3.10 Sequential Mode Conversion (ADMOD[1:0] = ‘00’, SEQCNT[2:0] > 0) 701
18.3.11 Burst Mode Conversion (ADMOD[1:0] = ‘01’, SEQCNT > 0) 705
18.3.12 Multiple Mode Conversion (ADMOD[1:0] = ‘10’, SEQCNT[2:0] > 0) 708
18.3.13 Injection Mode Conversion (INJMODE = ‘1’) 711
18.3.14 Starting Conversions (ASTART, Start Trigger, ARST) 713
18.3.14.1 Start A/D Conversion (ASTART) 713
18.3.14.2 Start All AD Conversion (AASTART) 713
18.3.14.3 Reset A/D Conversion (ARST) 713
18.3.14.4 Start A/D Conversion by Trigger Source 714
18.3.15 ADC Timing 715
18.3.16 Stopping Ongoing Conversion (ASTOP) 717
18.3.17 Conversion on External Trigger (TRGSEL[2:0], SEQTRGx[3:0], TRGIE, TRGIF) 718

18.3.17.1 ADC Trigger Source Selection (TRGSEL[2:0]) 718
18.3.17.2 ADC Trigger Channel Settings (SEQTRGx[3:0] and TRGSRC[3:0]) 719
18.3.17.3 ADC Trigger Start Control 720
18.3.17.4 ADC Trigger Interrupt and Flag (TRGIE, TRGIF) 720
18.3.18 End-of-Conversion, End-of-Sampling Phase (EOC) 721
18.3.19 End-of-Sequence Conversion (EOSIE, EOSIF) 722
18.3.20 Data Management and Information 724
18.3.20.1 Data Register and Data Alignment 724
18.3.20.2 Channel and Trigger Information 725
18.3.21 Trigger Pending Function in Sequential Mode 726
18.3.22 ADC Low-Power Modes 726
18.3.23 Monitoring Internal Voltage Reference 727
18.3.24 A/D Conversion Using DMA 728
18.3.25 ADC Overrun (OVR) in DMA Mode 730
18.3.26 Comparison of ADC Sensing Value 731
18.4 ADC Interrupts 733
18.5 ADC Registers 734
18.5.1 ADCn_MR: ADC Mode Register 735
18.5.2 ADCn_CSCR: ADC Current Sequence/Channel Register 737
18.5.3 ADCn_CCR: ADC n Clock Control Register 738
18.5.4 ADCn_TRG: ADC n Trigger Select Register 739
18.5.5 ADCn_CSTR1: ADC n Channel Sampling Time 1 Register 740
18.5.6 ADCn_CSTR2: ADC n Channel Sampling Time 2 Register 740
18.5.7 ADCn_SCSR1: ADC n Channel Select 1 Register 741
18.5.8 ADCn_SCSR2: ADC n Channel Select 2 Register 742
18.5.9 ADCn_CR: ADC n Control Register 743
18.5.10 ADCn_SR: ADC n Status Register 744
18.5.11 ADCn_IER: ADC n Interrupt Enable Register 745
18.5.12 ADCn_DDR: ADC n DMA Data Register 746
18.5.13 ADCn_DRx: ADC n Data 0 to 7 Register 747
18.5.14 ADCn_IMR: ADC n Injection Mode Register 748
18.5.15 ADCn_ICR: ADC n Injection Control Register 749
18.5.16 ADCn_IDR: ADC n Injection Data Register 749
18.5.17 ADCn_CMPR1: ADC n Channel Comparison Register 1 750
18.5.18 ADCn_CMPR2: ADC n Channel Comparison Register 2 751
18.5.19 ADCn_CD01R: ADC0, 1 Data Register 752
18.5.20 ADCn_CD2R: ADC2 Data Register 752
18.5.21 ADC Register Map Summary 753
19. Comparator (CMP) 755
19.1 Comparator Introduction 755
19.2 Comparator Main Features 755
19.3 Comparator Functional Description 756
19.3.1 Comparator Block Diagram 756
19.3.2 Comparator Pins and Internal Signals 757
19.3.3 Comparator Reference and Input Voltage Configuration 759
19.3.4 Comparator Reset and Clocks 760
19.3.5 Comparator Hysteresis 761
19.3.6 Comparator Filter 762
19.4 Comparator Low-Power Modes 764

19.5 Comparator Interrupts 764
19.6 Comparator Registers 765
19.6.1 CMP0_CONF: Comparator 0 Configure Register 766
19.6.2 CMP1_CONF: Comparator 1 Configure Register 767
19.6.3 CMP2_CONF: Comparator 2 Configure Register 768
19.6.4 CMP3_CONF: Comparator 3 Configure Register 769
19.6.5 CMPn_CTRL: Comparator n Control Register 770
19.6.6 CMPn_STAT: Comparator n Status Register 770
19.6.7 Comparator Register Map Summary 771
20. Digital to Analog Converter (DAC) 772
20.1 DAC Introduction 772
20.2 DAC Main Features 772
20.3 DAC Functional Description 773
20.3.1 12-bit DAC Block Diagram 773
20.3.2 DAC Pins and Signals 773
20.3.3 DAC Reset and Clock 774
20.3.4 DAC Channel Enable 774
20.3.5 DAC Output Buffer Enable 775
20.3.6 DAC Output Voltage 775
20.3.7 DAC Conversion 776
20.4 DAC Registers 777
20.4.1 DACn_DR: DAC n Data Register 778
20.4.2 DACn_CR: DAC n Control Register 778
20.4.3 DAC Register Map Summary 779
21. Advanced Encryption Standard (AES-128) 780
21.1 Introduction 780
21.2 AES Main Features 780
21.2.1 AES Implementation 780
21.3 AES Functional Description 781
21.3.1 AES Block Diagram 781
21.3.2 AES Internal Signals 782
21.3.3 AES Cryptographic Core 783
21.3.3.1 Overview 783
21.3.3.2 Cipher Mode (Encryption) 784
21.3.3.3 Inverse Cipher Mode (Decryption) 784
21.3.4 AES Procedure to Perform Cipher Operation 785
21.3.5 AES Procedure to Perform Inverse Cipher Operation 786
21.3.6 AES Data Padding 787
21.3.7 AES Data Input and Output Registers 787
21.3.8 AES Data Swapping 788
21.3.8.1 Word Inverse Mode Operation 789
21.3.8.2 Byte Inverse Mode Operation in Word Mode 789
21.3.8.3 Byte Inverse Mode Operation 790
21.3.9 AES Key Registers 791
21.3.10 AES DMA Interface 792
21.3.10.1 DMA Operation in Different Operating Modes 792
21.3.11 AES Processing Latency 792
21.4 AES Interrupts 793
21.5 AES Registers 794
21.5.1 AES_CTRL: AES Control Register 795

21.5.2 AES_STAT: AES Status Register 796
21.5.3 AES_INFIFO: AES Input FIFO Register 797
21.5.4 AES_OUTFIFO: AES Input FIFO Register 797
21.5.5 AES_KEYINx: AES Key Input Register 798
21.5.6 AES_TEXTINx: AES Text Input Register 798
21.5.7 AES_TEXTOUTx: AES Text Output Register 799
21.5.8 AES Register Map Summary 800
22. Random Number Generator (RNG) 801
22.1 RNG Introduction 801
22.2 20.2 RNG Main Features 801
22.3 RNG Functional Description 802
22.3.1 RNG Block Diagram 802
22.3.2 RNG Internal Signals 802
22.3.3 Random Number Generation 803
22.3.4 RNG Initialization 804
22.3.5 RNG Operation 805
22.3.5.1 Normal Operations 805
22.3.5.2 Low-Power Operations 806
22.3.6 RNG Clocking 807
22.3.7 RNG Processing Time 807
22.3.8 Error Management 807
22.3.9 RNG Low-Power Consumption in RUN Mode 807
22.4 RNG Interrupts 808
22.5 RNG Registers 809
22.5.1 RNG_CTRL: RNG Control Register 810
22.5.2 RNG_SEED: RNG Seed Register 811
22.5.3 RNG_RNGD: RNG Random Number Data Register 811
22.5.4 RNG_STAT: RNG Status Register 812
22.5.5 RNG Register Map Summary 813
23. Cyclic Redundancy Check (CRC) Calculation Module 814
23.1 CRC Introduction 814
23.2 CRC Main Features 814
23.3 CRC Functional Description 815
23.3.1 CRC Block Diagram 815
23.3.2 CRC Internal Signals 815
23.3.3 Timing Diagram of CRC Internal Signals for Data Size 816
23.3.4 CRC Operation 818
23.3.4.1 How to Obtain CRC Calculation Result 818
23.3.5 CRC Using DMA 820
23.4 CRC Registers 821
23.4.1 CRC_CTRL: CRC Control Register 822
23.4.2 CRC_INIT: CRC Initial Data Register 823
23.4.3 CRC_IDR: CRC Input Data Register 823
23.4.4 CRC_ODR: CRC Output Data Register 824
23.4.5 CRC_STAT: CRC Status Register 824
23.4.6 CRC Register Map Summary 825
Abbreviation for Registers 826
Glossary 827
Revision History 829

List of Tables

Table 1. Product Memory Density 31
Table 2. A34G43x Features and Peripherals 32
Table 3. Pin Description 43
Table 4. Interrupt Vector Map 56
Table 5. Peripheral Address 62
Table 6. Boot Mode Pin List 65
Table 7. SCU Pins 67
Table 8. Clock Sources 68
Table 9. Peripheral Clock Selection 69
Table 10. Clock Generation Circuit Input/Output Pins 70
Table 11. HSE Clock Pins 71
Table 12. LSE Clock Pins 75
Table 13. Peripheral Clock Selection 78
Table 14. Clock Monitoring Function 81
Table 15. Clock Source for Clock Monitoring 81
Table 16. Flash Memory Wait Control Recommendation 88
Table 17. Reset Sources for Cold Reset and Warm Reset 90
Table 18. LVR Level 94
Table 19. nRESET Pin 94
Table 20. Power Supply of VDD 97
Table 21. Operation Mode 101
Table 22. DEEP-SLEEP Mode Configuration 103
Table 23. Configurable Clock in Each RUN/SLEEP Mode 105
Table 24. Configurable Clocks in Each DEEP-SLEEP (STOP) Mode 105
Table 25. Base Address of CHIPCONFIG 106
Table 26. Register Map of CHIPCONFIG 106
Table 27. Base Address of CHIPCONFIG2 106
Table 28. Register Map of CHIPCONFIG2 106
Table 29. Base Address of SCU 107
Table 30. Register Map of SCU 107
Table 31. External Oscillator Control Settings 141
Table 32. SRAM Block Description 150
Table 33. SCU CHIPCONFIG Register Map Summary 158
Table 34. SCU CHIPCONFIG2 Register Map Summary 158
Table 35. SCU Register Map Summary 159
Table 36. PCU and GPIO Internal Signal 171
Table 37. PCU and GPIO Pins 171
Table 38. GPIO Alternate Function 174
Table 39. Interrupt Sources and Corresponding Pins in GPIO Module 177
Table 40. Base Address of PCU 182
Table 41. PCU and GPIO Register Map(1) 182
Table 42. PCU and GPIO Register Map Summary 198
Table 43. Code Flash Memory Controller Features 203
Table 44. Data Flash Memory Controller Features 204
Table 45. Internal Code Flash Memory Wait Time by Operating Clock 205
Table 46. Internal Data Flash Memory Wait Time by Operating Clock 206
Table 47. PGM Results After 2 Words Programming 212

Table 48. Available Operating Modes by Protection Level 213
Table 49. Read Protection Operation in UNPROT Level 214
Table 50. Read Protection Operation in Level 1 (LVL1) 214
Table 51. Read Protection Operation in Level-2 (LVL2) 215
Table 52. Base Address of Protection in Code Flash Memory 0 218
Table 53. Code Flash Memory 0 Write Protection Area in 32 KB Units (A34G430) 218
Table 54. Code Flash Memory 0 Write Protection Area in 64 KB Units (A34G43A) 218
Table 55. Code Flash Memory 0 Unprotection Area (Bottom 64 KB) in 8 KB Units (A34G43x) 219
Table 56. Code Flash Memory 0 Unprotection Area (Top 8 KB) in 1 KB Units (A34G430) 219
Table 57. Code Flash Memory 0 Unprotection Area (Top 8 KB) in 1 KB Units (A34G43A) 219
Table 58. Code Flash Memory 1 Base Address of Protection 220
Table 59. Code Flash Memory 1 Write Protection Area in 32 KB Units (A34G430) 220
Table 60. Code Flash Memory 1 Write Protection Area in 64 KB Units (A34G43A) 220
Table 61. Code Flash Memory 1 Unprotection Area (Bottom 64 KB) in 8 KB Units (A34G43x) 221
Table 62. Code Flash Memory 1 Unprotection Area (Top 8 KB) in 1 KB Units (A34G430) 221
Table 63. Code Flash Memory 1 Unprotection Area (Top 8 KB) in 1 KB Units (A34G43A) 221
Table 64. Data Flash Memory Write Protection Area in 2 KB Units 222
Table 65. Data Flash Memory Unprotection Area (Top 4 KB) in 512 Bytes Units 222
Table 66. Memory Bank Usage Status of Code Flash Memory 226
Table 67. Code Flash Memory RWW Features - Bank Disable (BM = 0, BS = 0) 232
Table 68. Code Flash Memory RWW Features - Bank Enable (BM = 1), Swap Disable (BS = 0) 232
Table 69. Code Flash Memory RWW Features - Bank Enable (BM = 1), Swap Enable (BS = 1) 232
Table 70. Base Address of CFMC 238
Table 71. CFMC Register Map 238
Table 72. CFMC Register Map Summary 260
Table 73. Base Address of DFMC 262
Table 74. DFMC Register Map 262
Table 75. DFMC Register Map Summary 272
Table 76. Base Address of DMA 280
Table 77. DMA Register Map 280
Table 78. DMA PERISEL[5:0] Selection 282
Table 79. DMA Register Map Summary 286
Table 80. Prescaled WDT Counter Clock Frequency 289
Table 81. Base Address of WDT 292
Table 82. WDT Register Map 292
Table 83. WDT Register Map Summary 297
Table 84. Pin Assignment of PWM: External Pins 300
Table 85. Interrupt on PWMn 305
Table 86. Interrupt Vector of PWMn 305
Table 87. Base Addresses of PWM Interface 306
Table 88. PWM Register Map 306
Table 89. PWM Register Map Summary 311
Table 90. Input and Output Pins for TIMERn (n = 0 to 9) 315
Table 91. Internal Input and Output Signals for TIMERn (n = 0 to 9) 316
Table 92. PCLK Divide Levels and CKSEL[2:0] in TIMERn_CR1 317
Table 93. SCU_MCCR3 Register and CKSEL[2:0] in TIMERn_CR1 317
Table 94. TnIO and IOSEL in TIMERn_CR1 318
Table 95. TnIO and CKSEL[2:0] in TIMERn_CR1 318
Table 96. Operations Comparison according to TIMERn_GRA/GRB Settings 322
Table 97. Operation Status by Capture Mode 327

Table 98. One-Shot Mode Operation Setting Table 331
Table 99. Interrupt Requests 342
Table 100. Base Address of 16-bit Timer 343
Table 101. 16-bit Timer Register Map 343
Table 102. TIMER Register Map Summary 354
Table 103. Base Address of FRT 361
Table 104. FRT Register Map 361
Table 105. FRT Register Map Summary 364
Table 106. UART Features 366
Table 107. Pin Assignment of UART: External Pins 368
Table 108. Received Bit Value from Sampled Data 381
Table 109. Examples of Baud Rate Calculation 384
Table 110. Examples of Calculation of Baud-Rate Using UART Fraction Counter 385
Table 111. Tolerance of UART Receiver 386
Table 112. UART Frame Formats 387
Table 113. Various Configurations to Create Parity Bit 388
Table 114. Base Address of UART 394
Table 115. UART Register Map 394
Table 116. TXE Bit Control 398
Table 117. Interrupt ID and Control 398
Table 118. UART Register Map Summary 412
Table 119. SPI Implementation 416
Table 120. Pin Assignment of SPI: External Pins 418
Table 121. Base Address of SPI 437
Table 122. SPI Register Map 437
Table 123. SPI Register Map Summary 448
Table 124. Features of I2C 450
Table 125. Pin Assignment of I2C: External Pins 452
Table 126. Effect of Low-Power Modes on I2C 475
Table 127. Base Address of I2C 476
Table 128. I2C Register Map 476
Table 129. I2C Register Map Summary 484
Table 130. Pin Assignment of QSPI: External Pins 488
Table 131. I/O Behavior of Memory Controller in Different Modes 494
Table 132. Dummy Phase Parameter Determination 496
Table 133. Example of Duration 498
Table 134. DMA Buffer Descriptor Layout 507
Table 135. DMA Buffer Descriptor Contents 507
Table 136. External Bus Space in QSPI Memory Map 511
Table 137. Configuration of XIP Register 512
Table 138. Interrupt on QSPI 515
Table 139. Base Address of CAN FD 516
Table 140. QSPI Register Map 516
Table 141. Mode Based on TRNAS and CPOL 519
Table 142. QSPI Register Map Summary 548
Table 143. CANFD Protocol Feature 552
Table 144. Pin Assignment of CANFD: External Pins 556
Table 145. Internal Input and Output Signal for CANFDn (n = 0 to 1) 556
Table 146. CAN Transceiver Overview 557
Table 147. Frame Identifier 558

Table 148. Bit Abbreviations 559
Table 149. Definition of DLC 559
Table 150. Definition of Unified Frame for TB, RB 560
Table 151. Definition of Unified Frame for ACF 567
Table 152. Acceptance of Filter Reset Value 567
Table 153. States of CAN Node 569
Table 154. RBALL and KOER[2:0] 574
Table 155. TSTAT Status Encoding 575
Table 156. TSTAT Status Events 576
Table 157. CAN Bit Timing Configuration 580
Table 158. Correlation with CAN Software Reset 591
Table 159. Interrupt on CANFDn 594
Table 160. Base Address of CAN FD 595
Table 161. CANFD Register Map 595
Table 162. CANFD Register Map Summary 627
Table 163. Pin Assignment of MPWM: External Pins 632
Table 164. Time-Base Register of MPWMn 633
Table 165. Counter Modes of MPWM 634
Table 166. Output Level (MPWM_OLR = 0x00) 637
Table 167. PWM Modes of MPWM 640
Table 168. Dead-Time Support of MPWMn 647
Table 169. Trigger Source by MPWMn 654
Table 170. Interrupt Source and Event of Motor PWM and Normal PWM Modes 657
Table 171. Effect of Low Power Modes on MPWM 658
Table 172. Interrupt on MPWMn 658
Table 173. Base Address of MPWM 659
Table 174. MPWM Register Map 659
Table 175. MPWM Register Map Summary 680
Table 176. Pin Assignment of ADC External: Pins 687
Table 177. ADC Input/Output Signals 688
Table 178. ADC Input Signals and Input Pins(1) 690
Table 179. ADC Input Signal and Input Pin(1) 696
Table 180. ADC Frequency for Values of R_{AIN}. 716
Table 181. External Trigger Source Configuration 718
Table 182. Trigger Source Table 719
Table 183. Start Control of AD Trigger Conversion 720
Table 184. Effect of Low-Power Mode on ADC 726
Table 185. Comparison and Flag Generation Conditions by CMPM 731
Table 186. Control Bits and Event Flags for ADC Interrupts 733
Table 187. Base Address of ADC 734
Table 188. ADC Register Map 734
Table 189. ADC Register Map Summary 753
Table 190. Pin Assignment of Comparator: External Pins 757
Table 191. Comparator Input Sources 758
Table 192. Comparator Reference Sources 758
Table 193. Comparator Interrupt Polarity 758
Table 194. Comparator Low-Power Modes 764
Table 195. Base Address of CMP 765
Table 196. CMP Register Map 765
Table 197. Comparator Register Map Summary 771

Table 198. Pin Assignment of DAC: External Pins 773
Table 199. Base Address of DAC 777
Table 200. DAC Register Map 777
Table 201. DAC Register Map Summary 779
Table 202. AES Internal Input/Output Signals 782
Table 203. text_in Buffer Endianness in AES_TEXTINx Registers (128-bit Length) 787
Table 204. text_out Buffer Endianness in AES_TEXTOUTx Registers (128-bit Length) 787
Table 205. Key Endianness in AES_KEYINx Registers (128-bit Key Length) 791
Table 206. Processing Latency (in Clock Cycle) for Encryption or Decryption 792
Table 207. AES Interrupt Requests 793
Table 208. Base Address of AES 794
Table 209. AES Register Map 794
Table 210. AES Register Map Summary 800
Table 211. RNG Internal Input and Output Signals 802
Table 212. RNG Interrupt Requests 808
Table 213. Base Address of RNG 809
Table 214. RNG Register Map 809
Table 215. RNG Register Map Summary 813
Table 216. CRC Internal Input/Output Signals 815
Table 217. Base Address of CRC 821
Table 218. CRC Register Map 821
Table 219. CRC Register Map Summary 825
Table 220. Abbreviation for Registers 826

List of Figures

Figure 1. A34G43x Block Diagram 2
Figure 2. 120-LQFP Pinouts 40
Figure 3. 100-LQFP Pinouts 41
Figure 4. 64-LQFP Pinouts 42
Figure 5. System Block Diagram 54
Figure 6. Interrupt Block Diagram 60
Figure 7. Memory Map 61
Figure 8. Bit-Banding Memory Map 63
Figure 9. SCU Block Diagram 67
Figure 10. Clock Configuration 70
Figure 11. HSE Crystal/Ceramic Oscillator 71
Figure 12. PLL Block Diagram 72
Figure 13. LSE External Clock 75
Figure 14. Miscellaneous Clock Configuration 77
Figure 15. Clock Monitoring Detection Range 80
Figure 16. MCLK Monitoring Recovery Procedure 83
Figure 17. MCLK Monitoring Reset Procedure 84
Figure 18.HSE Monitoring Procedure 85
Figure 19.LSE Monitoring Procedure 86
Figure 20. Clock Change Procedure 87
Figure 21. Peripheral Clock Select 89
Figure 22. Reset Tree Configuration 91
Figure 23. Timing Diagram of Power-Up Procedure (Cold Reset) 92
Figure 24. Warm Reset Timing Diagram 93
Figure 25. LVR Timing Diagram 94
Figure 26. POR and LVR Timing Diagram 97
Figure 27. LVI Block Diagram 98
Figure 28. LVR Block Diagram 99
Figure 29. POR and LVR Block Diagram 99
Figure 30. Transition between Operation Modes 101
Figure 31. SLEEP Mode Operation Sequence 102
Figure 32. DEEP-SLEEP Mode Operation Sequence 104
Figure 33. PCU and GPIO Block Diagram 167
Figure 34. I/O Port Block Diagram (GPIO Pins) 168
Figure 35. I/O Port Block Diagram (ADC, CMP) 169
Figure 36. I/O Port Block Diagram (External Oscillator Pins) 170
Figure 37. Output Port Block Diagram 178
Figure 38. Input Port Block Diagram 179
Figure 39. Port Function Block Diagram 180
Figure 40. Debouncing Logic Block Diagram 181
Figure 41. Example Timing Diagram of Port Debouncing 181
Figure 42. Code Flash Memory Map 203
Figure 43. Data Flash Memory Map 204
Figure 44. Flash Memory Page Erase Timing Diagram 208
Figure 45. Flash Memory Sector Erase Timing Diagram 209
Figure 46. Flash Memory Bulk (Full-Chip) Erase Timing Diagram 210
Figure 47. Code Flash Memory Program Timing Diagram 211

Figure 48. Data Flash Memory Program Timing Diagram 211
Figure 49. Write-Protection Coverage of Code Flash Memory when Using Memory Swap 224
Figure 50. A34G43A Memory Structure with Bank Swap Available 226
Figure 51. Logical Address Change of Code Flash Memory by Bank Swap Settings 227
Figure 52. Sequence of Bank Mode Enable 229
Figure 53. Sequence of Bank Swap: Bank0 → Bank1 230
Figure 54. Sequence of Bank Swap: Bank1 → Bank0 231
Figure 55. RWW in Code Flash Bank Disable 233
Figure 56. RWW in Code Flash Bank Enable and Bank Swap (Same Bank) 233
Figure 57. RWW in Code Flash Bank Enable and Bank Swap (Different Bank) 233
Figure 58. ECC Block Diagram 234
Figure 59. Sequence of ECC 236
Figure 60. DMA Block Diagram 276
Figure 61. Diagram of DMA Transfer Timing from Peripheral to Memory 278
Figure 62. Diagram of DMA Transfer Timing from Memory to Peripheral 279
Figure 63. N Number of DMA Transfers 279
Figure 64. WDT Block Diagram 288
Figure 65. Interrupt Mode Operation Timing Diagram with External Clock 290
Figure 66. PWM Block Diagram 299
Figure 67. PWM Timing Diagram 303
Figure 68. 16-bit Timer Block Diagram (n = 0 to 9) 314
Figure 69. Counter Timing Diagram with Basic Start and Match Operations 320
Figure 70. Normal Periodic Mode Operation 321
Figure 71. Operation Waveform Depending on UAO Bit 323
Figure 72. Operation Waveform According to STARTLVL and OUTPOL Settings 324
Figure 73. Timing Diagram of Capture Mode Operation 326
Figure 74. Timing Diagram of PWM Output Operation 328
Figure 75. Timing Diagram of One-Shot Mode Operation 330
Figure 76. Timing Diagram of ADC Triggering (TRGMOD[1:0] = ‘00’) 333
Figure 77. Timing Diagram of ADC Triggering (TRGMOD[1:0] = ‘10’) 333
Figure 78. Timing Diagram of ADC Triggering (TRGMOD[1:0] = ‘11’) 334
Figure 79. Timer Synchronization Example (When SSYNC = ‘1’, CSYNC = ‘1’) 336
Figure 80. Timer Synchronization Example (When Csync = ‘1’) 337
Figure 81. Timer Delayed Synchronization Example (When SSYNC = ‘1’) 338
Figure 82. SSYNC Operation Example 340
Figure 83. CSYNC Operation Example 340
Figure 84. FRT Block Diagram 356
Figure 85. Match Interrupt Operation Timing Diagram 357
Figure 86. Overflow Interrupt Operation Timing Diagram 358
Figure 87. Timing Diagram in Low-Power Mode 360
Figure 88. UART Block Diagram 367
Figure 89. Data Inversion Control Diagram 369
Figure 90. Timing Diagram of 8-bit Data Length 370
Figure 91. Timing Diagram of 7-bit Data Length 370
Figure 92. Timing Diagram of 6-bit Data Length 371
Figure 93. Timing Diagram of 5-bit Data Length 371
Figure 94. Example of Transmitted Data Frame 372
Figure 95. Transmit Interrupt Timing Diagram 373
Figure 96. Example of Transfer Data Format 376
Figure 97. Sampling Timing of UART Receiver 377

Figure 98. Start Bit Detection of Single-Sampling Timing 378
Figure 99. Start Bit Detection of Multi-Sampling Timing 379
Figure 100. Single-Sampling Timing of Data Bit 380
Figure 101. Multi-Sampling Timing of Data Bit 380
Figure 102. Data Sampling Timing 381
Figure 103. Transmission using DMA 390
Figure 104. Reception using DMA 391
Figure 105. Error Fragging and Interrupt Generation Timing Diagram 392
Figure 106. SPI Block Diagram 417
Figure 107. Block Diagram of SPI Master to Slave Connection: Full-Duplex 420
Figure 108. Block Diagram of SPI Master to Slave Connection: Simplex 420
Figure 109. SPI Transfer Timing 1 of 4 (CPHA = '0', CPOL = '0', MSBF = '0') 423
Figure 110. SPI Transfer Timing 2 of 4 (CPHA = '0', CPOL = '1', MSBF = '1') 423
Figure 111. SPI Transfer Timing 3 of 4 (CPHA = '1', CPOL = '0', MSBF = '0') 424
Figure 112. SPI Transfer Timing 4 of 4 (CPHA = '1', CPOL = '1', MSBF = '1') 424
Figure 113. DMA Handshake State Diagram 430
Figure 114. Example of Timing of SPI Transfer Operation Using DMA 432
Figure 115. SPI Waveforms (STL[7:0], BTL[7:0], and SPL[7:0])(1) 444
Figure 116. I2C Block Diagram 451
Figure 117. Noise Canceller Timing 453
Figure 118. I2C Bus Bit Transfer Timing Diagram 455
Figure 119. START and STOP Conditions 456
Figure 120. I2C Bus Response 457
Figure 121. Clock Synchronization During Arbitration 458
Figure 122. Arbitration Process between Two Masters 459
Figure 123. I2C Bus Data Transfer 460
Figure 124. Master Transmit Timing 461
Figure 125. Master Receive Timing 462
Figure 126. Slave Transmit Timing 463
Figure 127. Slave Receive Timing 464
Figure 128. Slave Transmitter Flowchart 465
Figure 129. Slave Receiver Flowchart 467
Figure 130. Master Transmitter Flowchart 469
Figure 131. Master Receiver Flowchart 472
Figure 132. Internal Delay (INTDEL[1:0]) in Master Mode 480
Figure 133. SCL Low (SCLL[15:0]) Timing 481
Figure 134. SCL High (SCLH[15:0]) Timing 482
Figure 135. SDA Hold (SDH[14:0]) Timing 483
Figure 136. QSPI Block Diagram 487
Figure 137. Transaction Level Operation of QSPI Memory Controller 489
Figure 138. Individual Phases of QSPI Transaction 489
Figure 139. Data Flows in QSPI Memory Controller during Data Phase of QSPI Transaction 490
Figure 140. QSPI Clock Output Behavior (TRANS = '0', CPOL: '0') 491
Figure 141. QSPI Clock Output Behavior (TRANS = '0', CPOL = '1') 491
Figure 142. Output Operation of QSPI Memory Controller in Normal (STR) Modes 492
Figure 143. Clock Output Behavior at Beginning and End of Transaction 492
Figure 144. Input Sampling Behavior of Controller when Sampling Clock is 180 Degrees Out of Phase with Internal Clock Generating Output Clock 493
Figure 145. Example of SDR Command in QSPI Mode 500
Figure 146. Example of Read Command in QSPI Mode 501

Figure 147. Recommended Clock Output Division when Using DDR Mode 501
Figure 148. Busy Status/Interrupt Timing Diagram 503
Figure 149. Write FIFO Status/Interrupt Timing Diagram 503
Figure 150. Write FIFO Error Status/Interrupt Timing Diagram 504
Figure 151. Read FIFO Status/Interrupt Timing Diagram 504
Figure 152. Read FIFO Error Status/Interrupt Timing Diagram 505
Figure 153. DMA Buffer Descriptor List Example 506
Figure 154. DMA Idle and IRQ Status Timing Diagram 508
Figure 155. DMA Error Status Timing Diagram 508
Figure 156. XIP Memory Map 509
Figure 157. XIP Address Split 510
Figure 158. Flowchart of XIP Mode 513
Figure 159. Releasing XIP Mode 514
Figure 160. CANFD Block Diagram 555
Figure 161. CAN CC, CAN FD Frame Types 558
Figure 162. CANFD Frame Buffer Diagram 560
Figure 163. Frame Reception of Receive Buffer (RB) 562
Figure 164. Frame Transmission of Transmit Buffer (PTB, STB) 564
Figure 165. Access to Acceptance Filters 566
Figure 166. ALC Field in CAN Frame 572
Figure 167. CAN Bit Timing Specifications 578
Figure 168. CAN Bit Timing 581
Figure 169. CAN FD Bit Rate Switching at Bit BRS 582
Figure 170. Transmitter Delay Compensation (TDC) 583
Figure 171. Listen-Only Mode Block Diagram 586
Figure 172. Loop Back Mode Block Diagram 588
Figure 173. MPWM Block Diagram 631
Figure 174. 1-Ch Symmetric Mode Timing Diagram 635
Figure 175. 1-Ch Asymmetric Mode Timing Diagram 636
Figure 176. Output Control Block 637
Figure 177. MPnWH/MPnWL Output at Motor PWM Mode (WHL = ‘0’, WLL = ‘0’) 638
Figure 178. MPnWH/MPnWL Output at Motor PWM Mode (WHL = ‘1’, WLL = ‘1’) 638
Figure 179. PWM Waveform in Motor PWM/Up-Down Counter/2-ch Symmetric Modes 641
Figure 180. PWM Waveform in Motor PWM/Up-Down Counter/1-ch Asymmetric Modes 642
Figure 181. PWM Waveform in Motor PWM/Up-Down Counter/1-ch Symmetric Modes 643
Figure 182. PWM Waveforms in Motor PWM Mode 1/Up Counter 644
Figure 183. PWM Waveforms in Normal PWM Mode/Up-Counter 645
Figure 184. Dead-Time Operation Timing Diagram (1-ch Symmetric Mode, HDTOL=‘0’, LDTOL=‘0’) 648
Figure 185. Dead-Time Operation Timing Diagram (1-ch Asymmetric Mode, HDTOL=‘0’, LDTOL=‘0’) 648
Figure 186. Typical Dead-Time Operation (T_{DUTY} > T_{DT}) 649
Figure 187. High Side Minimum Pulse Timing (T_{DUTY} < T_{DT} < 2 * T_{DUTY}) 650
Figure 188. High Side Zero Pulse Timing (T_{DT} > 2 * T_{DUTY}) 650
Figure 189. Low Side Minimum Pulse Timing (T_{DT} < period - T_{DUTY}) 651
Figure 190. Low Side Zero Pulse Timing (T_{DT} > period - T_{DUTY}) 651
Figure 191. High Side Always On (T_{DUTY} = Period When Dead-Time is Disabled.) 652
Figure 192. Low Side Always On (T_{DUTY} = 0, When Dead-Time is Disabled.) 652
Figure 193. ADC Conversion Timing Diagram by MPWM 655
Figure 194. Protection and Overcurrent Block Diagram 656

Figure 195. 12-bit ADC Block Diagram 686
Figure 196. 12-bit ADC Internal Channel Wiring 689
Figure 197. Block Diagram of ADC Clock 692
Figure 198. Example of ADC Conversion Timing 698
Figure 199. ADC Single Mode Timing (When ADMOD[1:0] = '00') 700
Figure 200. ADC Sequential Mode Timing (When ADMOD[1:0] = '00' and SEQCNT[2:0] > 0) 703
Figure 201. ADC Trigger Timing in Sequential Mode (SEQCNT[2:0] = '111', 8 Sequential Conversions) 704
Figure 202. ADC Burst Mode Timing (When ADMOD[1:0] = '01') 707
Figure 203. ADC Trigger Timing in Burst Mode (SEQCNT[2:0] = '111', 8 Sequential Conversions) 707
Figure 204. ADC Multiple Mode Timing (When ADMOD[1:0] = '10' and MR.SEQCNT[2:0] > 0) 710
Figure 205. ADC Trigger Timing in Multiple Mode (SEQCNT[2:0] = '111', 8 Multiple Conversions) 710
Figure 206. Example of Injection Mode Conversion 712
Figure 207. Analog-to-Digital Conversion Time 715
Figure 208. Typical Connection Diagram Using ADC 716
Figure 209. Stopping Ongoing A/D Conversions 717
Figure 210. Example of A/D Conversion Timing in Sequence Mode 723
Figure 211. Left Alignment (Unsigned Value) 724
Figure 212. Channel and Trigger Source Information of A/D Conversion Sequence 725
Figure 213. Channel Block Diagram of V_{CORE} 727
Figure 214. Block Diagram of ADC Internal Comparator 731
Figure 215. Timing Diagram of Comparison between A/D Conversion Results and Reference Value 732
Figure 216. Comparator Block Diagram 756
Figure 217. Comparator Input Source Block Diagram 756
Figure 218. Schematic Diagram of Comparator Reset Signal 760
Figure 219. Comparator Hysteresis 761
Figure 220. Comparator Filter 762
Figure 221. Comparator Filter Timing 762
Figure 222. DAC Block Diagram 773
Figure 223. Schematic Diagram of DAC Reset Signal 774
Figure 224. DAC Conversion 776
Figure 225. AES Block Diagram 781
Figure 226. AES Encryption and Decryption Operation Flow 783
Figure 227. AES Timing Diagram 785
Figure 228. Inverse Cipher Mode Operation 786
Figure 229. No Data Swapping 788
Figure 230. Word Inverse Mode Operation 789
Figure 231. Byte Inverse Operation in Word Mode 789
Figure 232. Byte Inverse Mode Operation 790
Figure 233. RNG Block Diagram 802
Figure 234. Event Occurrences with EN in RNG_CTRL Register Set to '1' 804
Figure 235. CRC Block Diagram 815
Figure 236. Timing Diagram of CRC Internal Signals: 8-bit Size Input and CRC-8 Outputs 816
Figure 237. Timing Diagram of CRC Internal Signals: 32-bit Size Input and CRC-8 Outputs 816
Figure 238. Timing Diagram of CRC Internal Signals: 32-bit Size Input and CRC-16 Outputs 817
Figure 239. Timing Diagram of CRC Internal Signals: 32-bit Size Input and CRC-32 Outputs 817
Figure 240. Timing Diagram of CRC Operation Using DMA 820

1. Description

The A34G43x series is a 32-bit microcontroller based on the high-performance Arm Cortex-M4F+ core, with up to 2048 KB of Dual-Bank code flash memory (bank0: 1024 KB, bank1: 1024 KB), 32 KB of data flash memory, and 256 KB of SRAM. It is a powerful microcontroller with a Floating-Point Unit (FPU) that provides effective solutions for various electrical appliances requiring low power consumption and high performance, such as refrigerators, washing machines, dryers, dishwashers, water purifiers, and blenders. It is also suitable for controlling inverter motors where high-performance is critical.

The A34G43x series also offers bank swap with RWW (Read-While-Write), which allows for simultaneous programming while the system continues to operate. Used in conjunction with Dual-Bank flash memory, A34G43x series allows code to be executed on one bank while being erased or programmed on the other bank. These functions are effective for implementing features such as OTA (Over-The-Air programming).

1.1 Product Category Definition

This document describes the superset of features for each product category.

Table 1 gives an overview of memory density versus product line. (For the list of features per category, refer to Table 2.)

Table 1. Product Memory Density

Memory Density		Category
Flash	RAM	
2,048 KB	256 KB	A34G43A
1,024 KB	256 KB	A34G430

1.2 Availability of Peripherals

Table 2 summarizes and lists product specific features available in the A34G43x considering the largest package.

Table 2. A34G43x Features and Peripherals

Peripherals		Description
Core	CPU	• Maximum operating frequency: 200 MHz • 32-bit Arm Cortex-M4F core - 32-bit Thumb®-2 instruction set • Register settings in CPU: - General-Purpose registers specified - Main stack pointer (MSP) and process stack pointer (PSP): R13 - Link register (LR): R14 - Program counter (PC): R15 • Data ordering format: Little-Endian • Harvard Architecture • AHB/APB
	Interrupt	• NVIC (Nested-Vectored Interrupt Controller) • Up to 84 peripheral interrupts supported • Assignable with 16 different priority levels
	FPU	• Rendered by extending and transforming the ARMv7 floating-point arithmetic functionality • Compliant with the ANSI/IEEE 754 standard • Capable of binary floating-point arithmetic and computation

Table 2. A34G43x Features and Peripherals (continued)

Peripherals		Description
Memory	Code Flash	• Capacity: - A34G43A: 2048 KB code flash memory - A34G430: 1024 KB code flash memory • A high-capacity code flash memory built in • Max 33 MHz flash access speed • Two-word (8-byte) program • Bulk (full-chip), 1 KB, and 4 KB erases • Read protection • Self-programming (Supports updating data in some code flash memory region during the execution of user program in the code area.) • CRC code generation and verification for the flash memory • Dual Bank code flash (bank0: 1024 KB, bank1: 1024 KB) for OTA (Over-The-Air programming) • Support RWW (Read-While-Write) • Support ECC (Error Correct Code) • Endurance: 10,000 cycles • Lifetime: 10 years
	Data Flash	• Capacity: 32 KB • Max 33 MHz access speed • One-word (4-byte) program • 512 bytes and 2 KB erases • Bulk (full-chip) erase • CRC code generation and verification for the flash memory • Support ECC (Error Correct Code) • Endurance: 100,000 Cycles • Lifetime: 10 years
	BootROM	• Entering the boot mode by setting the nBOOT pin to low-level during a reset procedure. • UART boot mode • In-System Programming • The boot mode can program the internal flash memory via UART.
	SRAM	• Capacity: 256 KB • Usable as a program’s work area • High-speed execution enables the execution of time-critical codes • Part of the SRAM can be remapped into an interrupt vector area

Table 2. A34G43x Features and Peripherals (continued)

Peripherals		Description
System Control Unit (SCU)	Clock	• High-speed internal oscillator (HSI) • 16 MHz - ±3.0% @ −40°C to +85°C - ±5.0% @ −40°C to +105°C • Low-speed internal oscillator (LSI) - 500 kHz (±30% at −40°C to +105°C) • External main oscillator (HSE): 4 MHz to 16 MHz • External sub-oscillator (LSE): 32.768 kHz • Phase-locked loop (PLL) frequency generator generates a high-speed clock (up to 200 MHz)
	Clock Monitoring	• System Fail-Safe function by Clock Monitoring • External main oscillator (HSE) • External sub oscillator (LSE) • Main system clock (MCLK)
	Operating Mode	• Run mode • Sleep mode • Deep sleep (STOP1, STOP2) mode
	Reset	• nRESET pin reset • Core reset • Software reset • POR (Power-On Reset) • LVR (Low-Voltage Reset) • WDTR (Watchdog Timer Reset) • Reset due to clock oscillating error • CFMC bank reset
	VDC	• Low-Drop Out (LDO) regulator built in for Low-Voltage operation
	POR	• The POR generator detects an internal 1.8 V voltage and generates a reset signal.
	LVI	• 16 Low-Voltage detection levels • Supports LVD interrupt • Supports wake-up from SLEEP mode

Table 2. A34G43x Features and Peripherals (continued)

Peripherals		Description
System Control Unit (SCU)	Wake-up	• Wake-up by a General-Purpose input/output (GPIO) pin • Wake-up by a Free-Run Timer (FRT) • Wake-up by a Watchdog Timer (WDT) • Wake-up by a Low-Voltage Indicator (LVI)
General-Purpose I/O (GPIO)		• Input/output (I/O) port for general purposes • 120-LQFP - I/O pins: 105 • 100-LQFP - I/O pins: 85 • 64-LQFP - I/O pins: 49 • Each pin can be set for one of the following modes: - Push-pull output - Open drain output - Input • The use of each pin can be set by setting the mux • Each pin can be configured as an external interrupt source, either the high-level/low-level interrupt or the rising-edge/falling-edge interrupt • Pull-up or pull-down resistor, and debouncing can be set for each pin • Drive strength can be adjusted for each port pin • Each pin bit can be individually set/reset • Wake-up events triggered by external asynchronous inputs
Direct Memory Access Controller (DMA)		• 16-ch direct memory access (DMA) support peripherals • 8-bit/16-bit/32-bit data transfers • Compatible with 57 different types of peripherals - SPIn (SPI0, SPI1, SPI2, SPI3, SPI4, SPI5, SPI6, SPI7, SPI8, SPI9, SPI10, SPI11, SPI12) - UARTn (UART0, UART1, UART2, UART3, UART4, UART5, UART6, UART7, UART8, UART9, UART10, UART11, UART12, UART13) - CRC - AES-128 - ADC (ADC0, ADC1, ADC2)

Table 2. A34G43x Features and Peripherals (continued)

Peripherals		Description
TIMER	16-bit timer	• General-Purpose 16-bit up-count timer • 10 channels - 10 timer n capture port (TnIO) input channels - 10 timer n output port (TnIO) output channels • Timer operating modes - Periodic timer mode - One-shot mode - PWM mode - Capture mode • Interrupt events - Timer/counter match interrupt - Timer overflow interrupt • Input clock selection - Clocks are freely selectable through the miscellaneous clock control registers (SCU_MCCRx) - External clocks are selectable • Timer signals can be generated through TnIO pins • 10-bit prescaler
	WDT	• 32-bit down-count timer • Reset and periodic interrupts • Clocks are freely selectable through the miscellaneous clock control registers (SCU_MCCRx) • Eight different prescalers are selectable
	FRT	• Two 32-bit Free-Run Timers - Capable of calculating the internal system time - 32-bit up-count timer • Interrupt events - Period interrupt - Overflow interrupt
Pulse-Width Modulation	PWM	• PWM generator with 10 channels • PWM signal with 16-bit independent counter • Consists of 1 unit per 5 channels. • 8-bit prescaler per 1 unit • Configuration of duty and period of PWM output signal • 1/2, 1/4, 1/8, 1/16 clock divider • 16-bit period and count value set • Built-in channels that outputs inverted PWM signal

Table 2. A34G43x Features and Peripherals (continued)

Peripherals		Description
Serial Interface	UART	• A total of thirteen 16450 asynchronous serial communication ports • Configurable standard asynchronous communication bits (start, stop, and parity) • Flexible communication available through programming - 5- to 8-bit data transfers - Even/Odd/Non-parity generation and checking - 1-bit, 1.5-bit, or 2-bit stop bit generation and checking • 8-bit fraction controller and 16-bit baud rate generator
	SPI	• Twelve synchronous serial communication port channels • Master/slave operation • Loop-back mode • Programmable and flexible communication • 8-bit/9-bit/16-bit/17-bit data transmit/receive - SPI clock speed - Both least significant bit (LSB)-first and most significant bit (MSB)-first modes available
	I2C	• Standard I2C communication protocol • Four channels supported • Master and slave modes supported for each channel • 7-bit addressing supported for slave mode • SCL signal's high/low periods and SDA signal's hold time settable
	CANFD	• Supports Flexible Data Rate (CANFD) protocol for high-speed communication. • Backward compatible with Classical CAN. • Configurable data payload sizes (up to 64 bytes). • Multiple message buffers for efficient message handling. - Acceptance filter buffer: 16 - Transmit buffer: 16 - Receive buffer: 23 • Supports error detection and fault confinement.
	QSPI	• Supports Quad Serial Peripheral Interface for high-speed flash memory communication. • Dual/Quad SPI modes supported for faster data transfers. • DDR (Double Data Rate) and SDR (Single Data Rate) modes supported for flexible communication. • Configurable clock speed and data lines for optimized performance. • Ideal for executing code or reading data directly from external flash memory (XIP support).

Table 2. A34G43x Features and Peripherals (continued)

Peripherals		Description
Motor Pulse-Width Modulation	MPWM	• Two Motor PWM generators • Six channels (high and low signals of phases U, V, and W) generate different waveforms • 16-bit up/down counters • Eight ADC trigger sources - Six variable triggers - Two fixed triggers (bottom, period match) • Interrupt events - Bottom interrupts - Top (period) interrupts - Interval interrupt mode • Falling/rising dead-time applicable • A special operating mode: • Phases U, V, and W are independently controlled - Different carrier counters running for phases U, V, and W - Different duty and periods configurable for phases U, V, and W • Protection and over-current detection supported
12-bit A/D Converter	ADC	• Three independent ADC blocks • 24 analog input channels • A number of operating modes: - Single conversion - Sequence conversion - Burst conversion - Multiple conversion - Inject conversion • Up to eight sequential conversions supported • Software triggers supported • Three internal trigger sources (MPWM and timers) supported • Sample time and hold time are adjustable
12-bit D/A Converter	DAC	• Two independent DAC block • Comparator reference support • 12bit resolution
Comparator	CMP	• Equipped with Comparator 0 and 1, each of which has three input sources • Equipped with Comparator 2 and 3, each of which has one input source • Signals from the DAC can be fed to the comparator

Table 2. A34G43x Features and Peripherals (continued)

Peripherals		Description
Advanced Encryption Standard	AES-128	• One AES channel • Compatible with direct memory access (DMA) • Input/output FIFO configurable • Input/output inversion supported
Random Number Generator	RNG	• Random-number generator • Interrupt events • Generator ready interrupt • Error interrupt
Cyclic Redundancy Check	CRC	• CRC operating modes: - CRC-32 (0x04C1_1DB7) - CRC-16-USB (0x8005) - CRC-16-CCITT (0x1021) - CRC-8-CCITT (0x07) • Input/output data reversion supported • Compatible with DMA
Operating Voltage		• 2.7 V to 5.5 V
Operating Temperature		• Commercial grade (−40°C to +105°C)
Package		• Three types of package options - 120-pin LQFP - 100-pin LQFP - 64-pin LQFP

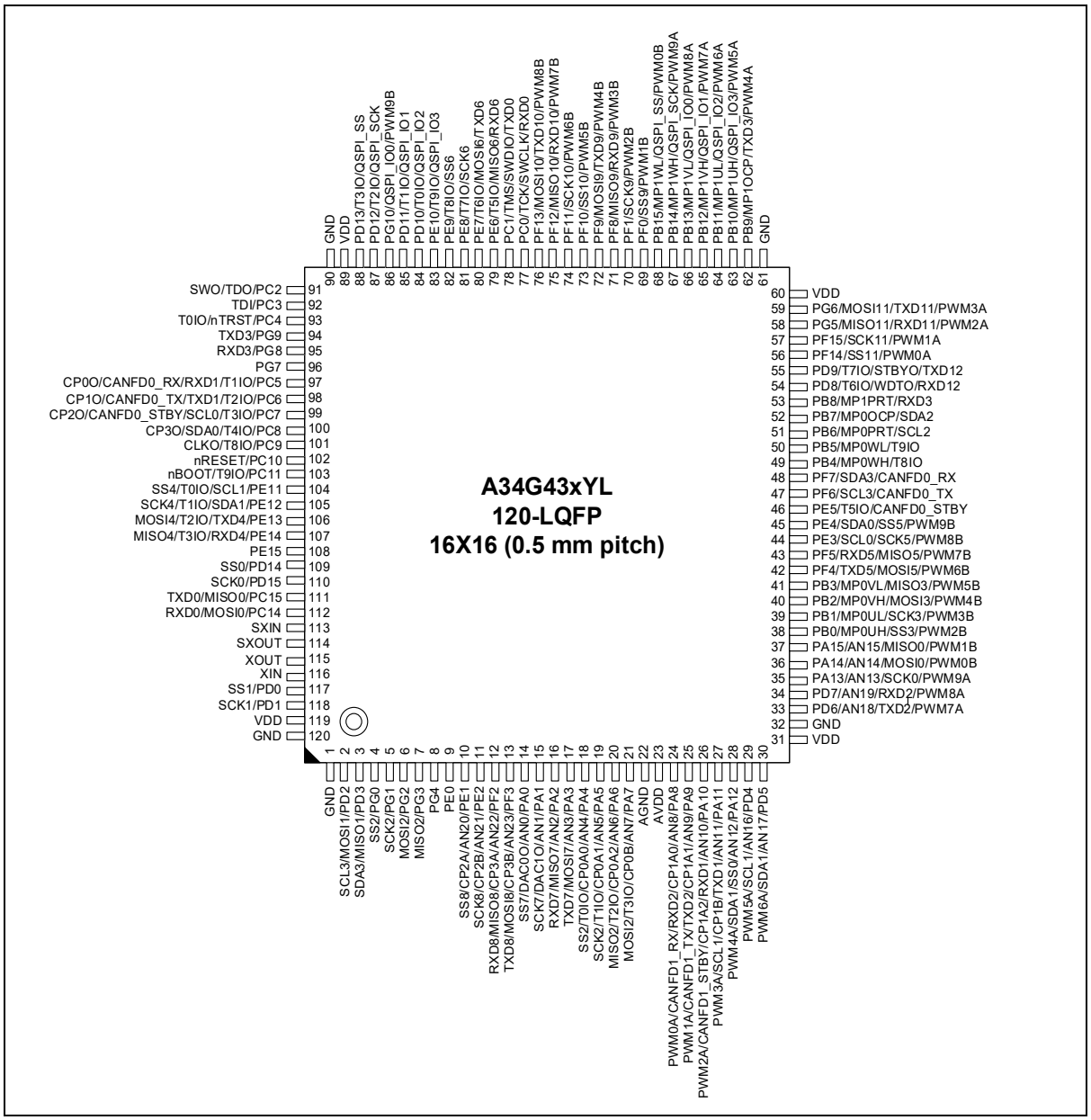
2. Pinouts and Pin Descriptions

In this chapter, pinouts and pin descriptions of the product are described.

2.1 Pinouts

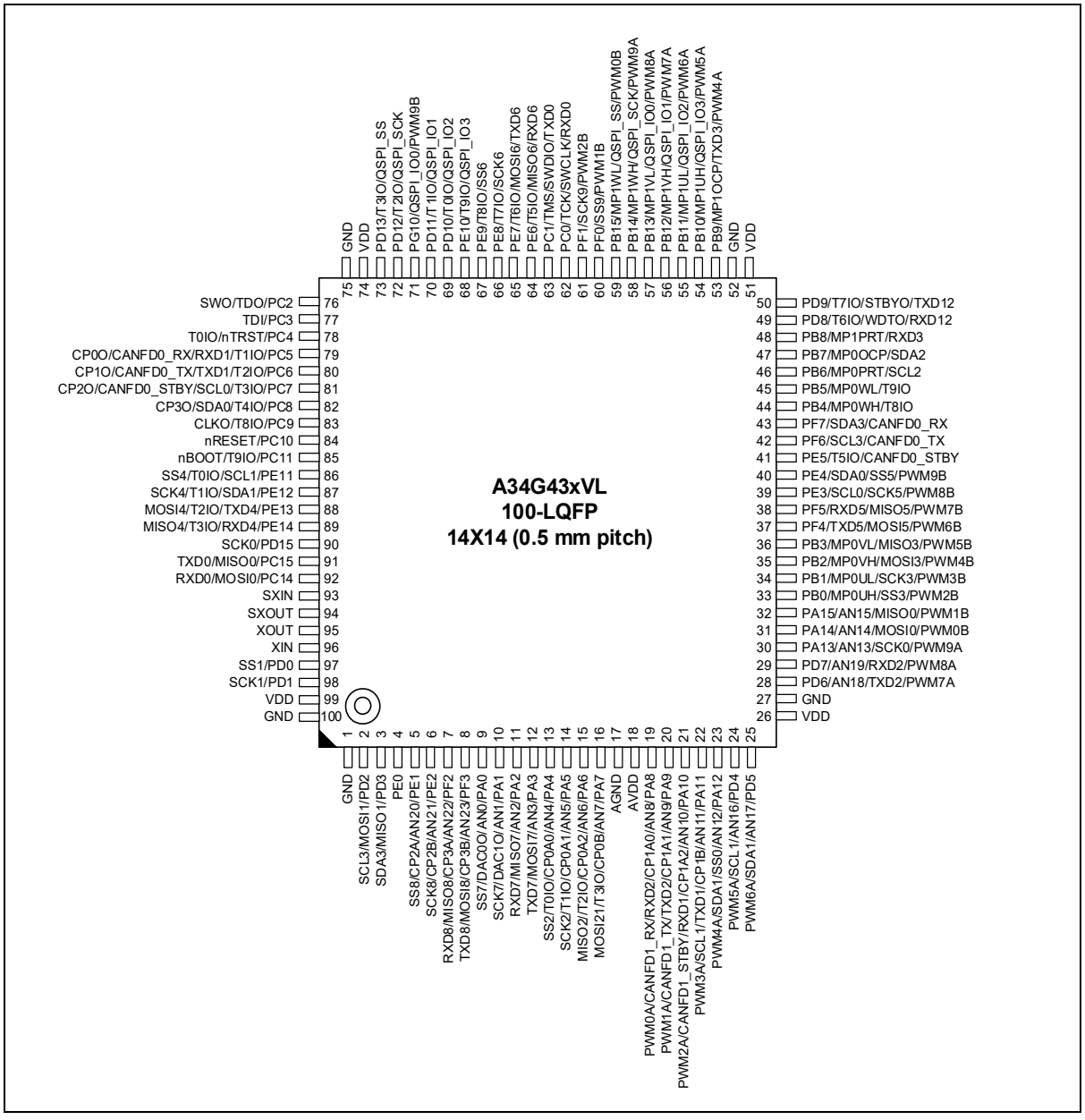
2.1.1 A34G43xYL (120-LQFP)

Figure 2. 120-LQFP Pinouts



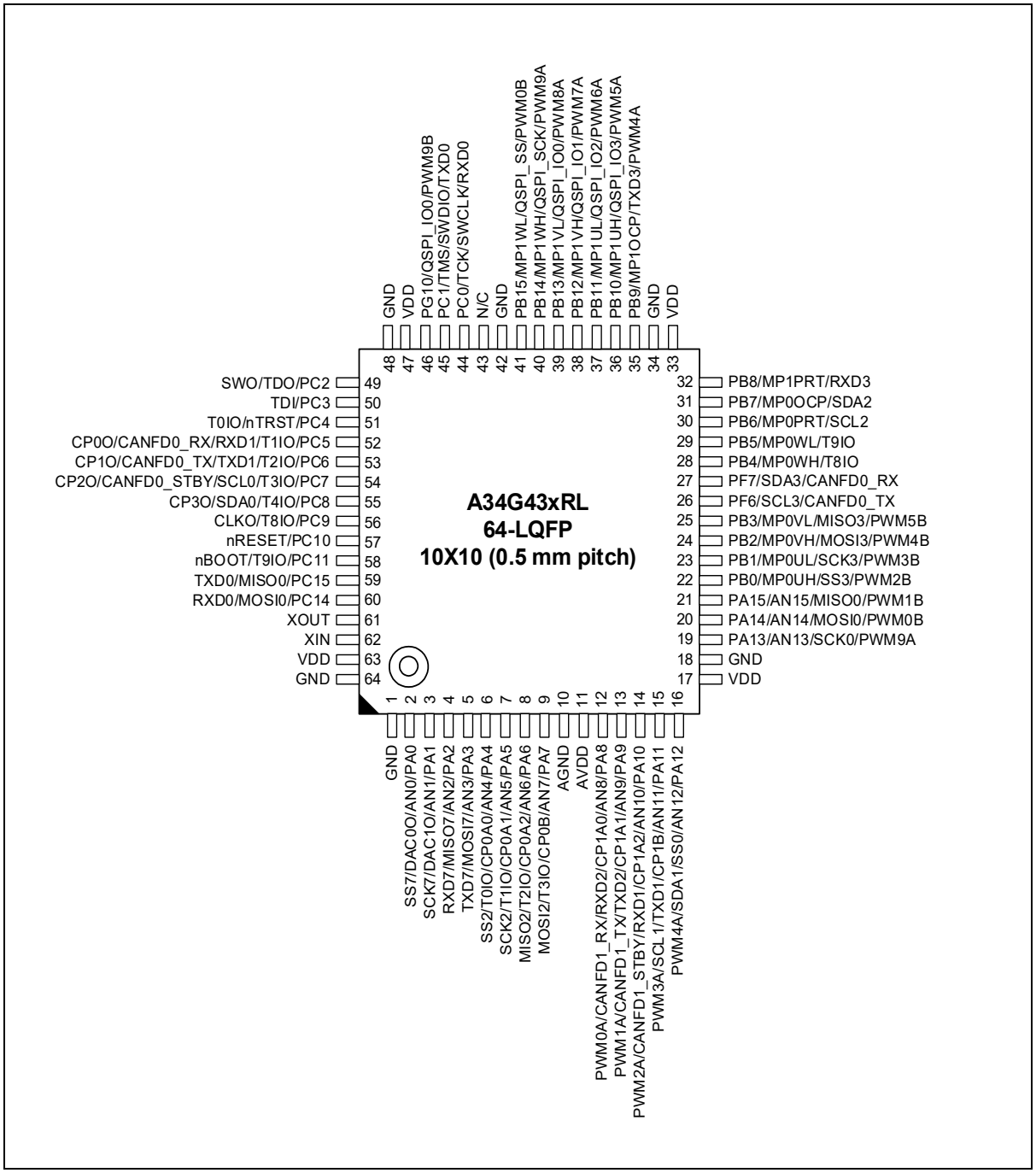
2.1.2 A34G43xVL (100-LQFP)

Figure 3. 100-LQFP Pinouts



2.1.3 A34G43xRL (64-LQFP)

Figure 4. 64-LQFP Pinouts



2.2 Pin Description

Pin configuration information in Table 3 contains two pairs of power, ground, and other dedicated pins. These multi-function pins have up to six selections of functions including GPIO. Configuration including pin ordering can be changed without notice.

Table 3. Pin Description

Pin No.			Pin Name	Type	Description	Remark
120-Pin	100-Pin	64-Pin				
1	1	1	GND	P	Ground	
2	2	-	PD2	IOUS	PORT D bit 2 input/output	
			MOSI1	IO	SPI channel 1 MOSI (Master out/Slave in) signal	
			SCL3[8]	IO	I2C channel 3 SCL input/output	Open-drain
3	3	-	PD3	IOUS	PORT D bit 3 input/output	
			MISO1	IO	SPI channel 1 MISO (Master in/Slave out) signal	
			SDA3[8]	IO	I2C channel 3 SDA input/output	Open-drain
4	-	-	PG0	IOUS	PORT G bit 0 input/output	
			SS2	IO	SPI channel 2 select signal input/output	
5	-	-	PG1	IOUS	PORT G bit 1 input/output	
			SCK2	IO	SPI channel 2 clock signal input/output	
6	-	-	PG2	IOUS	PORT G bit 2 input/output	
			MOSI2	IO	SPI channel 2 MOSI (Master out/Slave in) signal	
7	-	-	PG3	IOUS	PORT G bit 3 input/output	
			MISO2	IO	SPI channel 2 MISO (Master in/Slave out) signal	
8	-	-	PG4	IOUS	PORT G bit 4 input/output	
9	4	-	PE0	IOUS	PORT E bit 0 input/output	
10	5	-	PE1	IOUS	PORT E bit 1 input/output	
			SS8	IO	SPI channel 8 select signal input/output	
			AN20[2]	IA	Analog input 20	
			CP2A[2]	IA	Comparator 2 input A	
11	6	-	PE2	IOUS	PORT E bit 2 input/output	
			SCK8	IO	SPI channel 8 clock signal input/output	
			AN21[2]	IA	Analog input 21	
			CP2B[2]	IA	Comparator 2 input B	
12	7	-	PF2	IOUS	PORT F bit 2 input/output	
			RXD8	I	UART channel 8 RXD input	
			MISO8	IO	SPI channel 8 MISO (Master in/Slave out) signal	
			AN22[2]	IA	Analog input 22	
			CP3A[2]	IA	Comparator 3 input A	
13	8	-	PF3	IOUS	PORT F bit 3 input/output	
			TXD8	O	UART channel 8 TXD output	
			MOSI8	IO	SPI channel 8 MOSI (Master out/Slave in) signal	
			AN23[2]	IA	Analog input 23	
			CP3B[2]	IA	Comparator 3 input B	

Table 3. Pin Description (continued)

Pin No.			Pin Name	Type	Description	Remark
120-Pin	100-Pin	64-Pin				
14	9	2	PA0	IOUS	PORT A bit 0 input/output	
			SS7	IO	SPI channel 7 select signal input/output	
			DAC0O	OA	Digital to analog 0 output	
			AN0(2)	IA	Analog input 0	
15	10	3	PA1	IOUS	PORT A bit 1 input/output	
			SCK7	IO	SPI channel 7 clock signal input/output	
			DAC1O	OA	Digital to analog 1 output	
			AN1(2)	IA	Analog input 1	
16	11	4	PA2	IOUS	PORT A bit 2 input/output	
			RXD7	Input	UART channel 7 RXD input	
			MISO7	IO	SPI channel 7 MISO (Master in/Slave out) signal	
			AN2(2)	IA	Analog input 2	
17	12	5	PA3	IOUS	PORT A bit 3 input/output	
			TXD7	Output	UART channel 7 TXD output	
			MOSI7	IO	SPI channel 7 MOSI (Master out/Slave in) signal	
			AN3(2)	IA	Analog input 3	
18	13	6	PA4	IOUS	PORT A bit 4 input/output	
			SS2	IO	SPI channel 2 select signal input/output	
			T0IO	IO	Timer 0 input/output	
			AN4(2)	IA	Analog input 4	
			CP0A0(2)	IA	Comparator 0 input A0	
19	14	7	PA5	IOUS	PORT A bit 5 input/output	
			SCK2	IO	SPI channel 2 clock signal input/output	
			T1IO	IO	Timer 1 input/output	
			AN5(2)	IA	Analog input 5	
			CP0A1(2)	IA	Comparator 0 input A1	
20	15	8	PA6	IOUS	PORT A bit 6 input/output	
			MISO2	IO	SPI channel 2 MISO (Master in/Slave out) signal	
			T2IO	IO	Timer 2 input/output	
			AN6(2)	IA	Analog input 6	
			CP0A2(2)	IA	Comparator 0 input A2	
21	16	9	PA7	IOUS	PORT A bit 7 input/output	
			MOSI2	IO	SPI channel 2 MOSI (Master out/Slave in) signal	
			T3IO	IO	Timer 3 input/output	
			AN7(2)	IA	Analog input 7	
			CP0B(2)	IA	Comparator 0 input B	

Table 3. Pin Description (continued)

Pin No.			Pin Name	Type[1]	Description	Remark
120-Pin	100-Pin	64-Pin				
22	17	10	AGND	P	Analog Ground	
23	18	11	AVDD	P	Analog VDD	
24	19	12	PA8	IOUS	PORT A bit 8 input/output	
			RXD2	I	UART channel 2 RXD input	
			PWM0A	O	PWM channel 0 output	
			CANFD1_RX	I	CANFD channel 1 RXD input	
			AN8[2]	IA	Analog input 8	
			CP1A0[2]	IA	Comparator 1 input A0	
25	20	13	PA9	IOUS	PORT A bit 9 input/output	
			TXD2	O	UART channel 2 TXD output	
			PWM1A	O	PWM channel 1 output	
			CANFD1_TX	O	CANFD channel 1 TXD output	
			AN9[2]	IA	Analog input 9	
			CP1A1[2]	IA	Comparator 1 input A1	
26	21	14	PA10	IOUS	PORT A bit 10 input/output	
			RXD1	I	UART channel 1 RXD input	
			PWM2A	O	PWM channel 2 output	
			CANFD1_STBY	O	CANFD channel 1 standby signal output	
			AN10[2]	IA	Analog input 10	
			CP1A2[2]	IA	Comparator 1 input A2	
27	22	15	PA11	IOUS	PORT A bit 11 input/output	
			TXD1	O	UART channel 1 TXD output	
			PWM3A	O	PWM channel 3 output	
			AN11[2]	IA	Analog input 11	
			CP1B[2]	IA	Comparator 1 input B	
			SCL1[8]	IO	I2C channel 1 SCL input/output	Open-drain
28	23	16	PA12	IOUS	PORT A bit 12 input/output	
			SS0	IO	SPI channel 0 select signal input/output	
			PWM4A	O	PWM channel 4 output	
			AN12[2]	IA	Analog input 12	
			SDA1[8]	IO	I2C channel 1 SDA input/output	Open-drain
29	24	-	PD4	IOUS	PORT D bit 4 input/output	
			PWM5A	O	PWM channel 5 output	
			AN16[2]	IA	Analog input 16	
			SCL1[8]	IO	I2C channel 1 SCL input/output	Open-drain
30	25	-	PD5	IOUS	PORT D bit 5 input/output	
			PWM6A	O	PWM channel 6 output	
			AN17[2]	IA	Analog input 17	
			SDA1[8]	IO	I2C channel 1 SDA input/output	Open-drain
31	26	17	VDD	P	VDD	
32	27	18	GND	P	Ground	

Table 3. Pin Description (continued)

Pin No.			Pin Name	Type[1]	Description	Remark
120-Pin	100-Pin	64-Pin				
33	28	-	PD6	IOUS	PORT D bit 6 input/output	
			PWM7A	O	PWM channel 7 output	
			AN18[2]	IA	Analog input 18	
			TXD2	O	UART channel 2 TXD output	
34	29	-	PD7	IOUS	PORT D bit 7 input/output	
			PWM8A	O	PWM channel 8 output	
			AN19[2]	IA	Analog input 19	
			RXD2	I	UART channel 2 RXD input	
35	30	19	PA13	IOUS	PORT A bit 13 input/output	
			PWM9A	O	PWM channel 9 output	
			AN13[2]	IA	Analog input 13	
			SCK0	IO	SPI channel 0 clock input/output	
36	31	20	PA14	IOUS	PORT A bit 14 input/output	
			PWM0B	O	PWM channel 0 inversion output	
			AN14[2]	IA	Analog input 14	
			MOSI0	IO	SPI channel 0 MOSI (Master put/Slave in) signal	
37	32	21	PA15	IOUS	PORT A bit 15 input/output	
			PWM1B	O	PWM channel 1 inversion output	
			AN15[2]	IA	Analog input 15	
			MISO0	IO	SPI channel 0 MISO (Master in/Slave out) signal	
38	33	22	PB0	IOUS	PORT B bit 0 input/output	
			SS3	IO	SPI channel 3 select signal input/output	
			PWM2B	O	PWM channel 2 inversion output	
			MP0UH	O	MPWM channel 0 UH output	
39	34	23	PB1	IOUS	PORT B bit 1 input/output	
			SCK3	IO	SPI channel 3 clock input/output	
			PWM3B	O	PWM channel 3 inversion output	
			MP0UL	O	MPWM channel 0 UL output	
40	35	24	PB2	IOUS	PORT B bit 2 input/output	
			MOSI3	IO	SPI channel 3 MOSI (Master put/Slave in) signal	
			PWM4B	O	PWM channel 4 inversion output	
			MP0VH	O	MPWM channel 0 VH output	
41	36	25	PB3	IOUS	PORT B bit 3 input/output	
			MISO3	IO	SPI channel 3 MISO (Master in/Slave out) signal	
			PWM5B	O	PWM channel 5 inversion output	
			MP0VL	O	MPWM channel 0 VL output	
42	37	-	PF4	IOUS	PORT F bit 4 input/output	
			TXD5	O	UART channel 5 TXD output	
			PWM6B	O	PWM channel 6 inversion output	
			MOSI5	IO	SPI channel 5 MOSI (Master put/Slave in) signal	

Table 3. Pin Description (continued)

Pin No.			Pin Name	Type[1]	Description	Remark
120-Pin	100-Pin	64-Pin				
43	38	-	PF5	IOUS	PORT F bit 5 input/output	
			RXD5	I	UART channel 5 RXD input	
			PWM7B	O	PWM channel 7 inversion output	
			MISO5	IO	SPI channel 5 MISO (Master in/Slave out) signal	
44	39	-	PE3	IOUS	PORT E bit 3 input/output	
			PWM8B	O	PWM channel 8 inversion output	
			SCK5	IO	SPI channel 5 clock input/output	
			SCL0[8]	IO	I2C channel 0 SCL input/output	Open-drain
45	40	-	PE4	IOUS	PORT E bit 4 input/output	
			PWM9B	O	PWM channel 9 inversion output	
			SS5	IO	SPI channel 5 select signal input/output	
			SDA0[8]	IO	I2C channel 0 SDA input/output	Open-drain
46	41	-	PE5	IOUS	PORT E bit 5 input/output	
			T5IO	IO	Timer 5 input/output	
			CANFD0_STBY	O	CANFD channel 0 standby signal output	
47	42	26	PF6	IOUS	PORT F bit 6 input/output	
			CANFD0_TX	O	CANFD channel 0 TXD output	
			SCL3[8]	IO	I2C channel 3 SCL input/output	Open-drain
48	43	27	PF7	IOUS	PORT F bit 7 input/output	
			CANFD0_RX	I	CANFD channel 0 RXD input	
			SDA3[8]	IO	I2C channel 3 SDA input/output	Open-drain
49	44	28	PB4	IOUS	PORT B bit 4 input/output	
			MP0WH	O	MPWM channel 0 WH output	
			T8IO	IO	Timer 8 input/output	
50	45	29	PB5	IOUS	PORT B bit 5 input/output	
			MP0WL	O	MPWM channel 0 WL output	
			T9IO	IO	Timer 9 input/output	
51	46	30	PB6	IOUS	PORT B bit 6 input/output	
			MP0PRT	I	MPWM0 Protection input Signal	
			SCL2[8]	IO	I2C channel 2 SCL input/output	Open-drain
52	47	31	PB7	IOUS	PORT B bit 7 input/output	
			MP0OCP	I	MPWM0 Over-current protection input Signal	
			SDA2[8]	IO	I2C channel 2 SDA input/output	Open-drain
53	48	32	PB8	IOUS	PORT B bit 8 input/output	
			RXD3	I	UART channel 3 RXD input	
			MP1PRT	I	MPWM1 Protection input Signal	

Table 3. Pin Description (continued)

Pin No.			Pin Name	Type[1]	Description	Remark
120-Pin	100-Pin	64-Pin				
54	49	-	PD8	IOUS	PORT D bit 8 input/output	
			RXD12	I	UART channel 12 RXD input	
			T6IO	IO	Timer 6 input/output	
			WDTO	O	WDT output	
55	50	-	PD9	IOUS	PORT D bit 9 input/output	
			TXD12	I	UART channel 12 TXD output	
			T7IO	IO	Timer 7 input/output	
			STBYO	O	Power-down Mode Indication Signal	
56	-	-	PF14	IOUS	PORT F bit 14 input/output	
			PWM0A	O	PWM channel 0 output	
			SS11	IO	SPI channel 11 select signal input/output	
57	-	-	PF15	IOUS	PORT F bit 15 input/output	
			PWM1A	O	PWM channel 1 output	
			SCK11	IO	SPI channel 11 clock input/output	
58	-	-	PG5	IOUS	PORT G bit 5 input/output	
			RXD11	I	UART channel 11 RXD input	
			PWM2A	O	PWM channel 2 output	
			MISO11	IO	SPI channel 11 MISO (Master in/Slave out) signal	
59	-	-	PG6	IOUS	PORT G bit 6 input/output	
			TXD11	O	UART channel 11 TXD output	
			PWM3A	O	PWM channel 3 output	
			MOSI11	IO	SPI channel 11 MOSI (Master put/Slave in) signal	
60	51	33	VDD	P	VDD	
61	52	34	GND	P	Ground	
62	53	35	PB9	IOUS	PORT B bit 9 input/output	
			TXD3	O	UART channel 3 TXD output	
			PWM4A	O	PWM channel 4 output	
			MP1OCP	I	MPWM1 Over-current protection input Signal	
63	54	36	PB10	IOUS	PORT B bit 10 input/output	
			QSPI_IO3	IO	QSPI channel 0 IO3 input/output	
			PWM5A	O	PWM channel 5 output	
			MP1UH	O	MPWM channel 1 UH output	
64	55	37	PB11	IOUS	PORT B bit 11 input/output	
			QSPI_IO2	IO	QSPI channel 0 IO2 input/output	
			PWM6A	O	PWM channel 6 output	
			MP1UL	O	MPWM channel 1 UL output	

Table 3. Pin Description (continued)

Pin No.			Pin Name	Type[1]	Description	Remark
120-Pin	100-Pin	64-Pin				
65	56	38	PB12	IOUS	PORT B bit 12 input/output	
			QSPI_IO1	IO	QSPI channel 0 IO1 input/output	
			PWM7A	O	PWM channel 7 output	
			MP1VH	O	MPWM channel 1 VH output	
66	57	39	PB13	IOUS	PORT B bit 13 input/output	
			QSPI_IO0	IO	QSPI channel 0 IO0 input/output	
			PWM8A	O	PWM channel 8 output	
			MP1VL	O	MPWM channel 1 VL output	
67	58	40	PB14	IOUS	PORT B bit 14 input/output	
			QSPI_SCK	IO	QSPI channel 0 clock input/output	
			PWM9A	O	PWM channel 9 output	
			MP1WH	O	MPWM channel 1 WH output	
68	59	41	PB15	IOUS	PORT B bit 15 input/output	
			QSPI_SS	IO	QSPI channel 0 select signal input/output	
			PWM0B	O	PWM channel 0 inversion output	
			MP1WL	O	MPWM channel 1 WL output	
69	60	-	PF0	IOUS	PORT F bit 0 input/output	
			PWM1B	O	PWM channel 1 inversion output	
			SS9	IO	SPI channel 9 select signal input/output	
70	61	-	PF1	IOUS	PORT F bit 1 input/output	
			PWM2B	O	PWM channel 2 inversion output	
			SCK9	IO	SPI channel 9 clock input/output	
71	-	-	PF8	IOUS	PORT F bit 8 input/output	
			RXD9	I	UART channel 9 RXD input	
			PWM3B	O	PWM channel 3 inversion output	
			MISO9	IO	SPI channel 9 MISO (Master in/Slave out) signal	
72	-	-	PF9	IOUS	PORT F bit 9 input/output	
			TXD9	O	UART channel 9 TXD output	
			PWM4B	O	PWM channel 4 inversion output	
			MOSI9	IO	SPI channel 9 MOSI (Master put/Slave in) signal	
73	-	-	PF10	IOUS	PORT F bit 10 input/output	
			PWM5B	O	PWM channel 5 inversion output	
			SS10	IO	SPI channel 10 select signal input/output	
74	-	-	PF11	IOUS	PORT F bit 11 input/output	
			PWM6B	O	PWM channel 6 inversion output	
			SCK10	IO	SPI channel 10 clock input/output	
75	-	-	PF12	IOUS	PORT F bit 12 input/output	
			RXD10	I	UART channel 10 RXD input	
			PWM7B	O	PWM channel 7 inversion output	
			MISO10	IO	SPI channel 10 MISO (Master in/Slave out) signal	

Table 3. Pin Description (continued)

Pin No.			Pin Name	Type[1]	Description	Remark
120-Pin	100-Pin	64-Pin				
76	-	-	PF13	IOUS	PORT F bit 13 input/output	
			TXD10	O	UART channel 10 TXD output	
			PWM8B	O	PWM channel 8 inversion output	
			MOSI10	IO	SPI channel 10 MOSI (Master put/Slave in) signal	
-	-	42	GND	P	Ground	
-	-	43	N/C	-	N/C	
77	62	44	PC0	IOUS	PORT C bit 0 input/output	
			TCK/SWCLK[2] (3)(6)(7)	I	JTAG TCK, SWD clock input	Pull-up
			RXD0	I	UART channel 0 RXD input	
78	63	45	PC1	IOUS	PORT C bit 1 input/output	
			TMS/SWDIO[2] (3)(6)(7)	IO	JTAG TMS, SWD data input/output	Pull-up
			TXD0	O	UART channel 0 TXD output	
79	64	-	PE6	IOUS	PORT E bit 6 input/output	
			RXD6	I	UART channel 6 RXD input	
			T5IO	IO	Timer 5 input/output	
			MISO6	IO	SPI channel 6 MISO (Master in/Slave out) signal	
80	65	-	PE7	IOUS	PORT E bit 7 input/output	
			TXD6	O	UART channel 6 TXD output	
			T6IO	IO	Timer 6 input/output	
			MOSI6	IO	SPI channel 6 MOSI (Master put/Slave in) signal	
81	66	-	PE8	IOUS	PORT E bit 8 input/output	
			T7IO	IO	Timer 7 input/output	
			SCK6	IO	SPI channel 6 clock input/output	
82	67	-	PE9	IOUS	PORT E bit 9 input/output	
			T8IO	IO	Timer 8 input/output	
			SS6	IO	SPI channel 6 select signal input/output	
83	68	-	PE10	IOUS	PORT E bit 10 input/output	
			QSPI_IO3	IO	QSPI channel 0 IO3 input/output	
			T9IO	IO	Timer 9 input/output	
84	69	-	PD10	IOUS	PORT D bit 10 input/output	
			QSPI_IO2	IO	QSPI channel 0 IO2 input/output	
			T0IO	IO	Timer 0 input/output	
85	70	-	PD11	IOUS	PORT D bit 11 input/output	
			QSPI_IO1	IO	QSPI channel 0 IO1 input/output	
			T1IO	IO	Timer 1 input/output	
86	71	46	PG10	IOUS	PORT G bit 10 input/output	
			QSPI_IO0	IO	QSPI channel 0 IO0 input/output	
			PWM9B	O	PWM channel 9 inversion output	

Table 3. Pin Description (continued)

Pin No.			Pin Name	Type[1]	Description	Remark
120-Pin	100-Pin	64-Pin				
87	72	-	PD12	IOUS	PORT D bit 12 input/output	
			QSPI_SCK	IO	QSPI channel 0 clock input/output	
			T2IO	IO	Timer 2 input/output	
88	73	-	PD13	IOUS	PORT D bit 13 input/output	
			QSPI_SS	IO	QSPI channel 0 select signal input/output	
			T3IO	IO	Timer 3 input/output	
89	74	47	VDD	P	VDD	
90	75	48	GND	P	Ground	
91	76	49	PC2	IOUS	PORT C bit 2 input/output	
			TDO/SWO[2]	O	JTAG TDO output, Serial Wire Output	Output Low
92	77	50	PC3	IOUS	PORT C bit 3 input/output	
			TDI[2]	I	JTAG TDI input	Pull-up
93	78	51	PC4	IOUS	PORT C bit 4 input/output	
			nTRST[2]	O	JTAG nTRST input	Pull-up
			T0IO	IO	Timer 0 input/output	
94	-	-	PG9	IOUS	PORT G bit 9 input/output	
			TXD3	O	UART channel 3 TXD output	
95	-	-	PG8	IOUS	PORT G bit 8 input/output	
			RXD3	I	UART channel 3 RXD input	
96	-	-	PG7	IOUS	PORT G bit 7 input/output	
97	79	52	PC5	IOUS	PORT C bit 5 input/output	
			T1IO	IO	Timer 1 input/output	
			RXD1	I	UART channel 1 RXD input	
			CANFD0_RX	I	CANFD channel 0 RXD input	
			CP0O	OA	Comparator 0 output	
98	80	53	PC6	IOUS	PORT C bit 6 input/output	
			T2IO	IO	Timer 2 input/output	
			TXD1	O	UART channel 1 TXD output	
			CANFD0_TX	O	CANFD channel 0 TXD output	
			CP1O	OA	Comparator 1 output	
99	81	54	PC7	IOUS	PORT C bit 7 input/output	
			CP2O	OA	Comparator 2 output	
			T3IO	IO	Timer 3 input/output	
			SCL0[8]	IO	I2C channel 0 SCL input/output	Open-drain
			CANFD0_STBY	O	CANFD channel 0 standby signal output	
100	82	55	PC8	IOUS	PORT C bit 8 input/output	
			CP3O	OA	Comparator 3 output	
			T4IO	IO	Timer 4 input/output	
			SDA0[8]	IO	I2C channel 0 SDA input/output	Open-drain
101	83	56	PC9	IOUS	PORT C bit 9 input/output	
			T8IO	IO	Timer 8 input/output	
			CLKO	O	System Clock output	
102	84	57	PC10	IOUS	PORT C bit 9 input/output	
			T9IO	IO	Timer 9 input/output	
			nRESET[2][3]	I	External Reset input	Pull-up

Table 3. Pin Description (continued)

Pin No.			Pin Name	Type[1]	Description	Remark
120-Pin	100-Pin	64-Pin				
103	85	58	PC11	IOUS	PORT C bit 11 input/output	
			T9IO	IO	Timer 9 input/output	
			nBOOT[2][3][5]	I	Boot Mode Selection input	Pull-up
104	86	-	PE11	IOUS	PORT E bit 11 input/output	
			SS4	IO	SPI channel 4 select signal input/output	
			SCL1[8]	IO	I2C channel 1 SCL input/output	Open-drain
			T0IO	IO	Timer 0 input/output	
105	87	-	PE12	IOUS	PORT E bit 12 input/output	
			SCK4	IO	SPI channel 4 clock input/output	
			SDA1[8]	IO	I2C channel 1 SDA input/output	Open-drain
			T1IO	IO	Timer 1 input/output	
106	88	-	PE13	IOUS	PORT E bit 13 input/output	
			MOSI4	IO	SPI channel 4 MOSI (Master put/Slave in) signal	
			TXD4	O	UART channel 4 TXD output	
			T2IO	IO	Timer 2 input/output	
107	89	-	PE14	IOUS	PORT E bit 14 input/output	
			MISO4	IO	SPI channel 4 MISO (Master in/Slave out) signal	
			RXD4	I	UART channel 4 RXD input	
			T3IO	IO	Timer 3 input/output	
108	-	-	PE15	IOUS	PORT E bit 15 input/output	
109	-	-	PD14	IOUS	PORT D bit 14 input/output	
			SS0	IO	SPI channel 0 select signal input/output	
110	90	-	PD15	IOUS	PORT D bit 15 input/output	
			SCK0	IO	SPI channel 0 clock input/output	
111	91	59	PC15	IOUS	PORT C bit 15 input/output	
			MISO0	IO	SPI channel 0 MISO (Master in/Slave out) signal	
			TXD0	O	UART channel 0 TXD output	
112	92	60	PC14	IOUS	PORT C bit 14 input/output	
			MOSI0	IO	SPI channel 0 MOSI (Master out/Slave in) signal	
			RXD0	I	UART channel 0 RXD input	

Table 3. Pin Description (continued)

Pin No.			Pin Name	Type[1]	Description	Remark
120-Pin	100-Pin	64-Pin				
113	93	-	SXIN[2][9]	IA	Sub crystal oscillator input	
114	94	-	SXOUT[2][9]	OA	Sub crystal oscillator output	
115	95	61	XOUT[2][9]	OA	External crystal oscillator output	
116	96	62	XIN[2][9]	IA	External crystal oscillator input	
117	97	-	PD0	IOUS	PORT D bit 0 input/output	
			SS1	IO	SPI channel 1 select signal input/output	
118	98	-	PD1	IOUS	PORT D bit 1 input/output	
			SCK1	IO	SPI channel 1 clock input/output	
119	99	63	VDD	P	VDD	
120	100	64	GND	P	Ground	

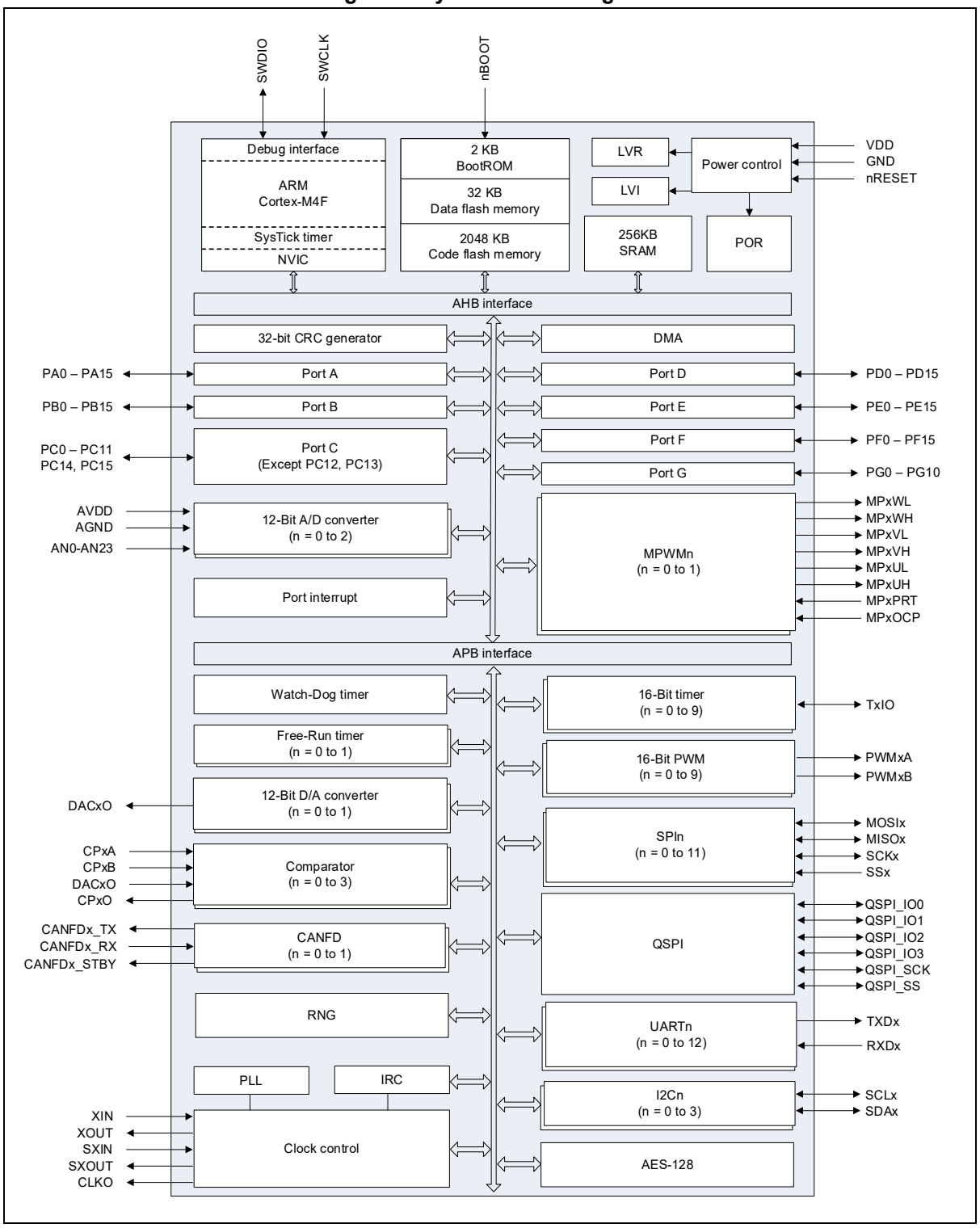
NOTES:

1. I = Input, O = Output, U = Pull-up, D = Pull-down, S = Schmitt-Trigger Input Type, C = CMOS Input Type, A = Analog, P = Power
2. After a reset, all the pins are configured to function defined by their initial values. The initial value of the pin depends on the package type. This configuration is in compliance with the 120-pin standard.
3. nBOOT, nRESET, PC1 (SWDIO), PC0 (SWCLK), PC3 and PC4 are the default pull-up pins.
4. Do not configure unused pins as floating inputs.
5. After a reset, the internal pull-up for the boot pin is enabled.
6. After a reset, the internal pull-up for the serial wire clock (SWCLK) and the serial wire data I/O (SWDIO) is enabled.
7. The SWCLK and SWDIO pins should not be switched to other functions while they are being used.
8. PA11, PA12, PB6, PB7, PC7, PC8, PD2, PD3, PD4, PD5, PE3, PE4, PE11, PE12, PF6 and PF7 pins are configured as open-drain ports when used as I2C ports.
9. XIN, XOUT, SXOUT, and SXIN cannot be used for GPIO or other alternate functions. When the HSE and LSE clocks are not in use, the clocks must be disabled by using software.

3. System and Memory Overview

3.1 System Architecture

Figure 5. System Block Diagram



3.1.1 Cortex-M4F Core

The Arm® Cortex-M4F has the same functions as the Cortex-M4 and includes optional floating point arithmetic functionality. The two processors are intended for deeply embedded applications that require fast interrupt response features.

3.1.2 Floating Point Unit (FPU)

Cortex-M4 FPU is an implementation of the single precision variant of the ARMv7-M Floating-Point Extension (FPv4-SP). It provides floating-point computation functionality that is compliant with the ANSI/IEEE Std 754-2008, IEEE Standard for Binary Floating-Point Arithmetic, referred to as the IEEE 754 standard. The FPU supports all single-precision data-processing instructions and data types described in the ARM®v7-M Architecture Reference Manual.

3.1.3 Interrupt Controller

Table 4. Interrupt Vector Map

Priority	Vector Address	Interrupt Source
-16	0x0000_0000	Stack Pointer
-15	0x0000_0004	Reset Handler
-14	0x0000_0008	NMI Handler
-13	0x0000_000C	Hard Fault Handler
-12	0x0000_0010	MemManage Handler
-11	0x0000_0014	BusFault Handler
-10	0x0000_0018	UsageFault Handler
-9	0x0000_001C	Reserved
-8	0x0000_0020	
-7	0x0000_0024	
-6	0x0000_0028	
-5	0x0000_002C	SVCall Handler
-4	0x0000_0030	Debug Monitor Handler
-3	0x0000_0034	Reserved
-2	0x0000_0038	PendSV Handler
-1	0x0000_003C	SysTick Handler
0	0x0000_0040	LVI
1	0x0000_0044	SYSCLKFAIL
2	0x0000_0048	HSEFAIL
3	0x0000_004C	LSEFAIL
4	0x0000_0050	Reserved
5	0x0000_0054	
6	0x0000_0058	WDT
7	0x0000_005C	Reserved
8	0x0000_0060	FRT0
9	0x0000_0064	FRT1
10	0x0000_0068	Reserved
11	0x0000_006C	CFMC
12	0x0000_0070	DFMC
13	0x0000_0074	Reserved
14	0x0000_0078	
15	0x0000_007C	TIMER0

Table 4. Interrupt Vector Map (continued)

Priority	Vector Address	Interrupt Source
16	0x0000_0080	TIMER1
17	0x0000_0084	TIMER2
18	0x0000_0088	TIMER3
19	0x0000_008C	TIMER4
20	0x0000_0090	TIMER5
21	0x0000_0094	TIMER6
22	0x0000_0098	TIMER7
23	0x0000_009C	TIMER8
24	0x0000_00A0	TIMER9
25	0x0000_00A4	Reserved
26	0x0000_00A8	
27	0x0000_00AC	RNG
28	0x0000_00B0	AES-128
29	0x0000_00B4	Reserved
30	0x0000_00B8	
31	0x0000_00BC	
32	0x0000_00C0	
33	0x0000_00C4	
34	0x0000_00C8	
35	0x0000_00CC	
36	0x0000_00D0	GPIOA
37	0x0000_00D4	GPIOB
38	0x0000_00D8	GPIOC
39	0x0000_00DC	GPIOD
40	0x0000_00E0	GPIOE
41	0x0000_00E4	GPIOF
42	0x0000_00E8	GPIOG
43	0x0000_00EC	Reserved
44	0x0000_00F0	
45	0x0000_00F4	MPWM0PROT
46	0x0000_00F8	MPWM0OC
47	0x0000_00FC	MPWM0
48	0x0000_0100	Reserved
49	0x0000_0104	
50	0x0000_0108	MPWM1PROT

Table 4. Interrupt Vector Map (continued)

Priority	Vector Address	Interrupt Source
51	0x0000_010C	MPWM1OC
52	0x0000_0110	MPWM1
53	0x0000_0114	Reserved
54	0x0000_0118	
55	0x0000_011C	SPI0
56	0x0000_0120	SPI1
57	0x0000_0124	SPI2
58	0x0000_0128	Reserved
59	0x0000_012C	
60	0x0000_0130	I2C0
61	0x0000_0134	I2C1
62	0x0000_0138	I2C2
63	0x0000_013C	UART0
64	0x0000_0140	UART1
65	0x0000_0144	UART2
66	0x0000_0148	UART3
67	0x0000_014C	UART4
68	0x0000_0150	UART5
69	0x0000_0154	Reserved
70	0x0000_0158	
71	0x0000_015C	CANFD0
72	0x0000_0160	CANFD1
73	0x0000_0164	Reserved
74	0x0000_0168	ADC0
75	0x0000_016C	ADC1
76	0x0000_0170	ADC2
77	0x0000_0174	Reserved
78	0x0000_0178	
79	0x0000_017C	CMP0
80	0x0000_0180	CMP1
81	0x0000_0184	CMP2
82	0x0000_0188	CMP3
83	0x0000_018C	Reserved
84	0x0000_0190	
85	0x0000_0194	CRC

Table 4. Interrupt Vector Map (continued)

Priority	Vector Address	Interrupt Source
86	0x0000_0198	Reserved
87	0x0000_019C	
88	0x0000_01A0	PWM0
89	0x0000_01A4	PWM1
90	0x0000_01A8	PWM2
91	0x0000_01AC	PWM3
92	0x0000_01B0	PWM4
93	0x0000_01B4	PWM5
94	0x0000_01B8	PWM6
95	0x0000_01BC	PWM7
96	0x0000_01C0	PWM8
97	0x0000_01C4	PWM9
98	0x0000_01C8	Reserved
99	0x0000_01CC	
100	0x0000_01D0	I2C3
101	0x0000_01D4	Reserved
102	0x0000_01D8	
103	0x0000_01DC	
104	0x0000_01E0	QSPI
105	0x0000_01E4	Reserved
106	0x0000_01E8	
107	0x0000_01EC	
108	0x0000_01F0	UART6
109	0x0000_01F4	UART7
110	0x0000_01F8	UART8
111	0x0000_01FC	UART9
112	0x0000_0200	UART10
113	0x0000_0204	UART11
114	0x0000_0208	UART12
115	0x0000_020C	Reserved
116	0x0000_0210	SPI3
117	0x0000_0214	SPI4
118	0x0000_0218	SPI5
119	0x0000_021C	SPI6
120	0x0000_0220	SPI7

Table 4. Interrupt Vector Map (continued)

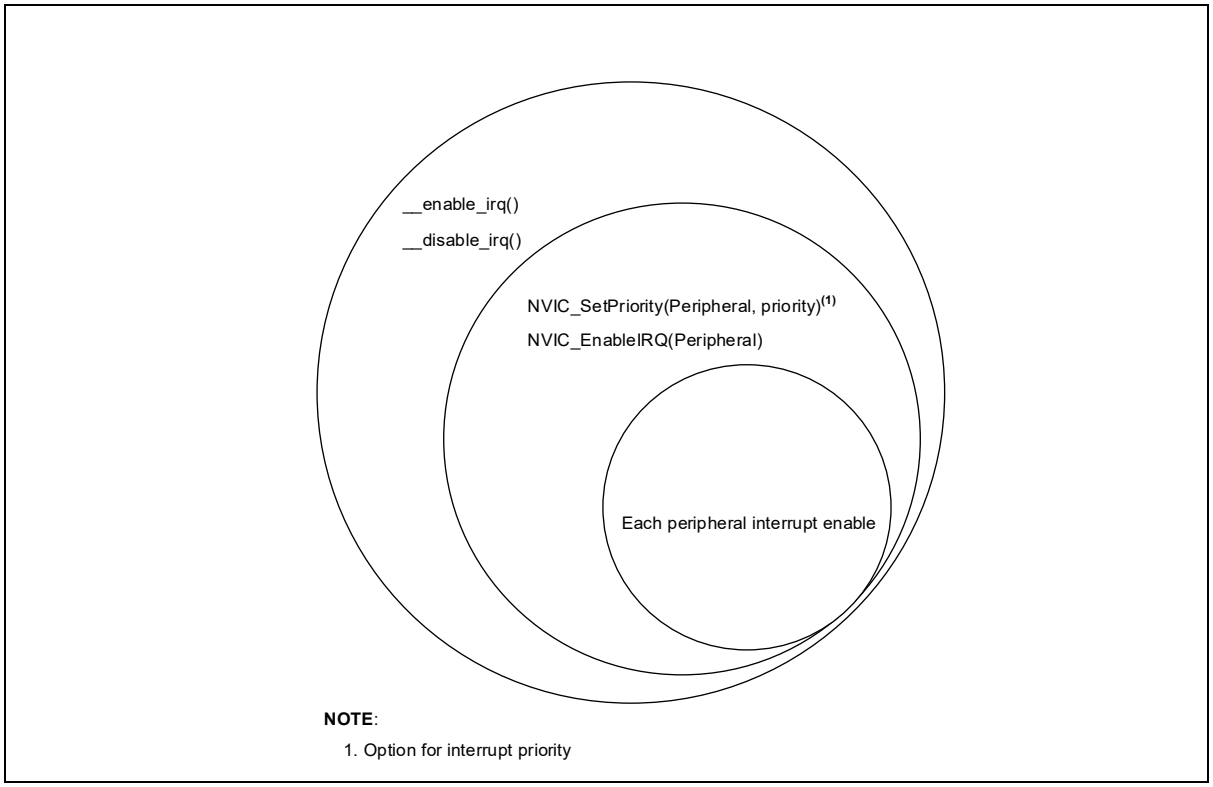
Priority	Vector Address	Interrupt Source
121	0x0000_0224	SPI8
122	0x0000_0228	SPI9
123	0x0000_022C	SPI10
124	0x0000_0230	SPI11

NOTE:

1. Each interrupt is linked to the priority level registers. Each interrupt priority register is 1-byte length and only the upper 4 bits of it are used for the priority level. NVIC registers of Cortex-M4F processor use four interrupt priority bytes at a time because it supports the transmission in word unit.

   ** __NVIC_PRIO_BITS = 4

Figure 6. Interrupt Block Diagram



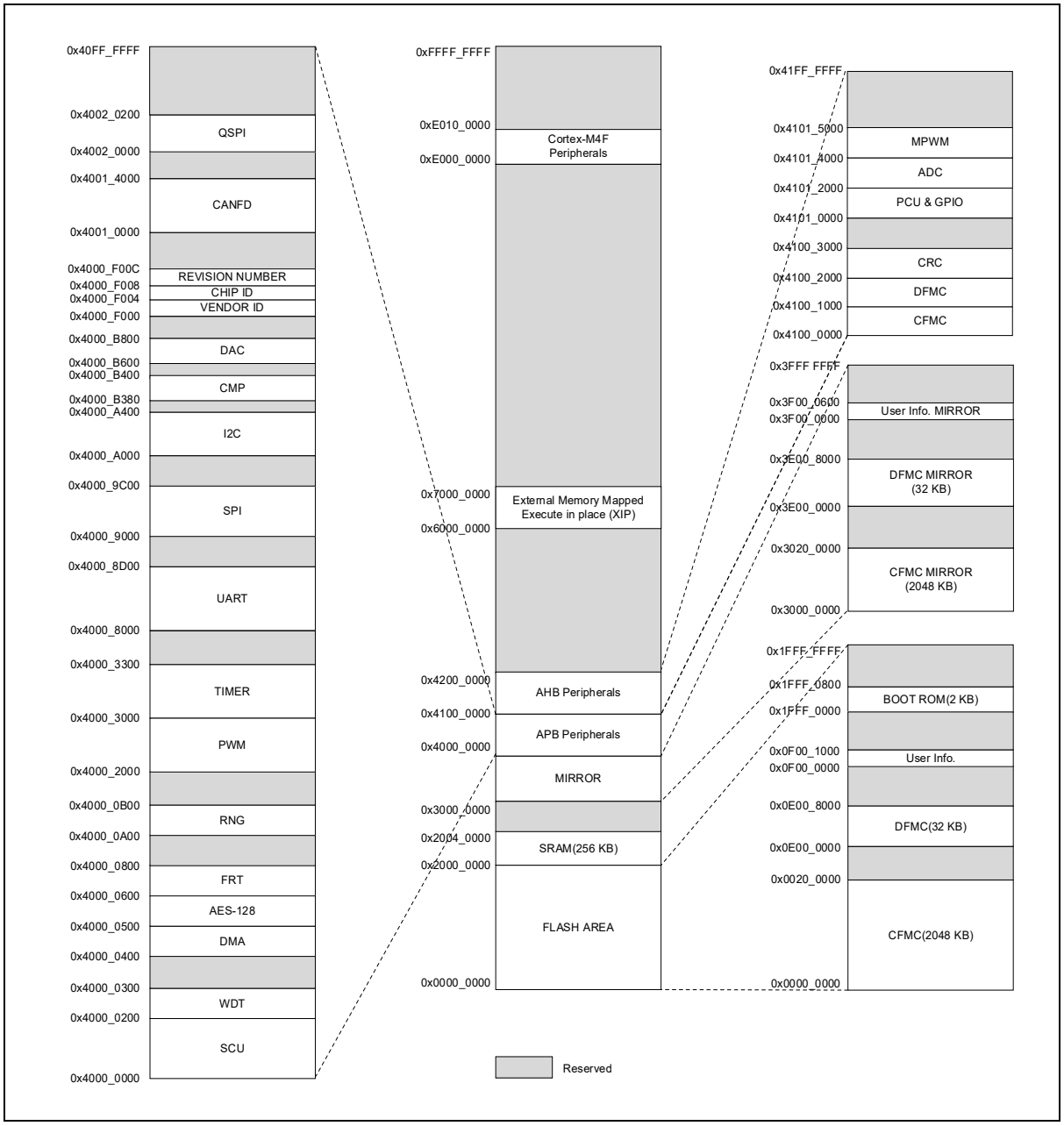
3.2 Memory Organization

3.2.1 Introduction

Program memory, data memory, registers and I/O ports are organized in the same address space. The bytes are coded in memory in Little Endian format. The lowest numbered byte in a word is considered the word's least significant byte and the highest numbered byte the most significant.

3.2.2 Memory Map and Register Boundary Address

Figure 7. Memory Map



All the memory map areas that are not allocated to on-chip memories and peripherals are considered “Reserved”. For the detailed mapping of available memory and register areas, refer to the following table.

The following Table 5 gives the boundary addresses of the peripherals available in the devices.

Table 5. Peripheral Address

Base Address	Peripheral Name
0x4000_0000	SCU
0x4000_0200	WDT
0x4000_0400	DMA 0/1/2/3/4/5/6/7/8/9/10/11/12/13/14/15
0x4000_0500	AES-128
0x4000_0600	FRT 0/1
0x4000_0A00	RNG
0x4000_2000	PWM 0/1/2/3/4/5/6/7/8/9/10
0x4000_3000	Timer 0/1/2/3/4/5/6/7/8/9
0x4000_8000	UART 0/1/2/3/4/5/6/7/8/9/10/11/12
0x4000_9000	SPI 0/1/2/3/4/5/6/7/8/9/10/11
0x4000_A000	I2C 0/1/2
0x4000_B380	CMP 0/1/2/3
0x4000_B600	DAC 0/1
0x4001_0000	CANFD0
0x4001_2000	CANFD1
0x4002_0000	QSPI
0x4100_0000	CFMC
0x4100_1000	DFMC
0x4100_2000	CRC
0x4101_0000	PCU A/B/C/D/E/F/G
0x4101_2000	ADC 0/1/2
0x4101_4000	MPWM0
0x4101_5000	MPWM1
0x6000_0000	QSPI(XIP, External Memory Mapped)
0x2000_0000	Internal SRAM

3.3 Memory Map

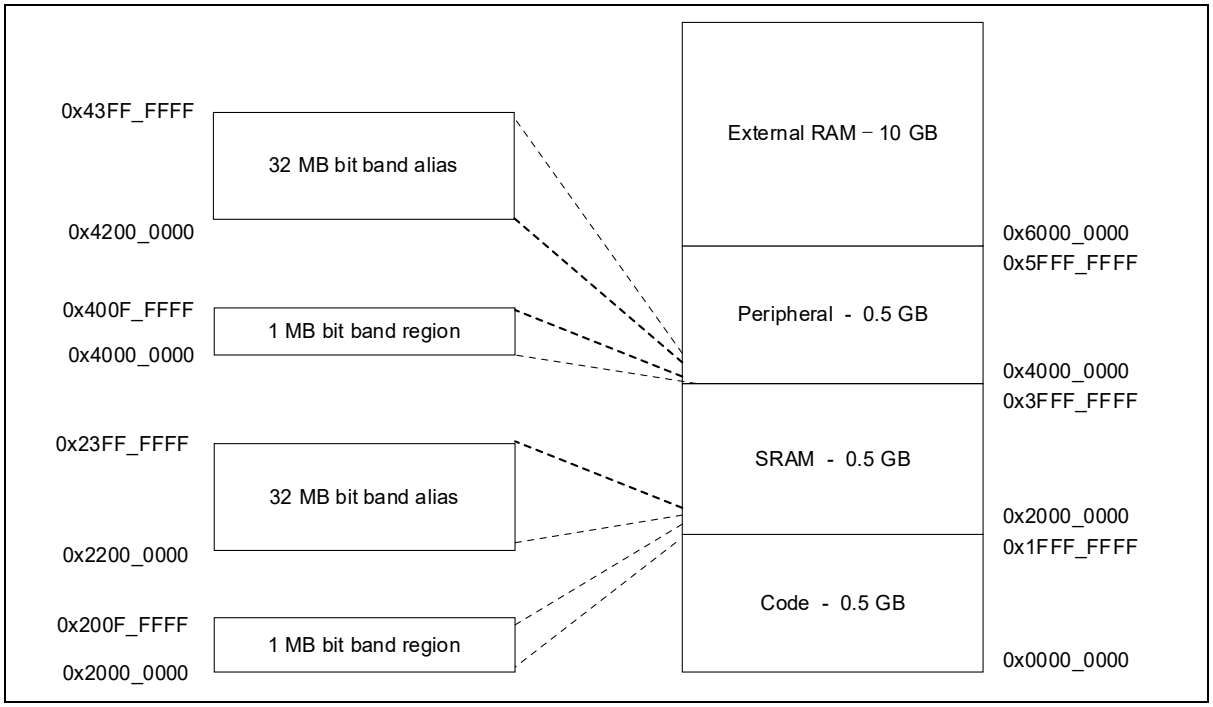
3.3.1 Bit-Banding

The Cortex-M4 with FPU memory map includes two bit-band regions. These regions map each word in an alias region of memory to a bit in a bit-band region of memory. Writing to a word in the alias region has the same effect as a read-modify-write operation on the targeted bit in the bit-band region.

In the A34G43x series devices both the peripheral registers and the SRAM are mapped to a bit-band region, so that single bit-band write and read operations are allowed.

The following is the bit-banding memory map and calculation formula of peripheral and SRAM.

Figure 8. Bit-Banding Memory Map



- Bit_Word_Addr = bit_band_base + (byte_offset × 32) + (bit_number × 4)
  - bit_band_base:
    - Peripheral start address = 0x4200_0000
    - SRAM start address = 0x2200_0000
  - byte_offset:
    - Peripheral = (Peripheral_base_addr - 0x4000_0000 + offset) × 32
    - SRAM = (SRAM_addr - 0x2000_0000 + offset) × 32

3.3.2 Embedded SRAM

The A34G43x series has a block of 0-wait on-chip SRAM. Size of the SRAM is 256 KB and its base address is 0x2000_0000.

This SRAM memory area is usually used for data memory and stack memory. Sometimes the code is dumped into the SRAM memory for fast operation or flash erase/programming operation.

3.3.3 Flash Memory Overview

The A34G43x series provides internal 2048 KB code flash memory and a controller. This is enough to control the general system. Self-programming is available and ISP and SWD programming is also supported in boot mode or in debugging mode.

Instruction and data cache buffer are present and overcome the low bandwidth flash memory. CPU can access flash memory with one wait state up to 33 MHz bus frequency.

3.4 Boot Configuration

The A34G43x series has a boot mode option to program internal flash memory. Boot mode can be entered by setting nBOOT pin to ‘L’ at reset timing (Normal state is ‘H’).

Boot mode supports UART boot. The pins for the boot mode are listed in Table 6.

Table 6. Boot Mode Pin List

Block	Pin Name	I/O	Pin Direction	Description
SYSTEM	nRESET	PC10	Input	Reset input signal
	nBOOT	PC11	Input	Boot mode setting pin
UART0	RXD0	PC14	Input	UART boot receive data
	TXD0	PC15	Output	UART boot transmit data

4. System Control Unit (SCU)

4.1 SCU Introduction

The A34G43x series has a built-in intelligent power control block which manages system analog blocks and operating modes. System Control Unit (SCU) module controls an internal reset and clock signals to maintain optimized system performance and power dissipation.

4.2 SCU Main Features

The clock features of A34G43x series are as follows:

- Operating frequency: Up to 200 MHz
- High-Speed Internal Oscillator (HSI): 16 MHz
- Low-Speed Internal Oscillator (LSI): 500 kHz
- High-Speed External Oscillator (HSE): 4 MHz to 16 MHz
- Low-Speed External Oscillator (LSE): 32.768 kHz
- Phase-locked loop (PLL) frequency generator generates a high-speed clock (Up to 200 MHz)

A34G43x series have clock monitoring as a system fail-safe function.

- The reset features of A34G43x series are as follows:
- nRESET pin reset
- CPU reset
- Software reset
- Power-On Reset (POR)
- Low-Voltage Detect Reset (LVR)
- Watchdog Timer Reset (WDT)
- Reset due to clock oscillating error

Operating mode features of A34G43x series are as follows:

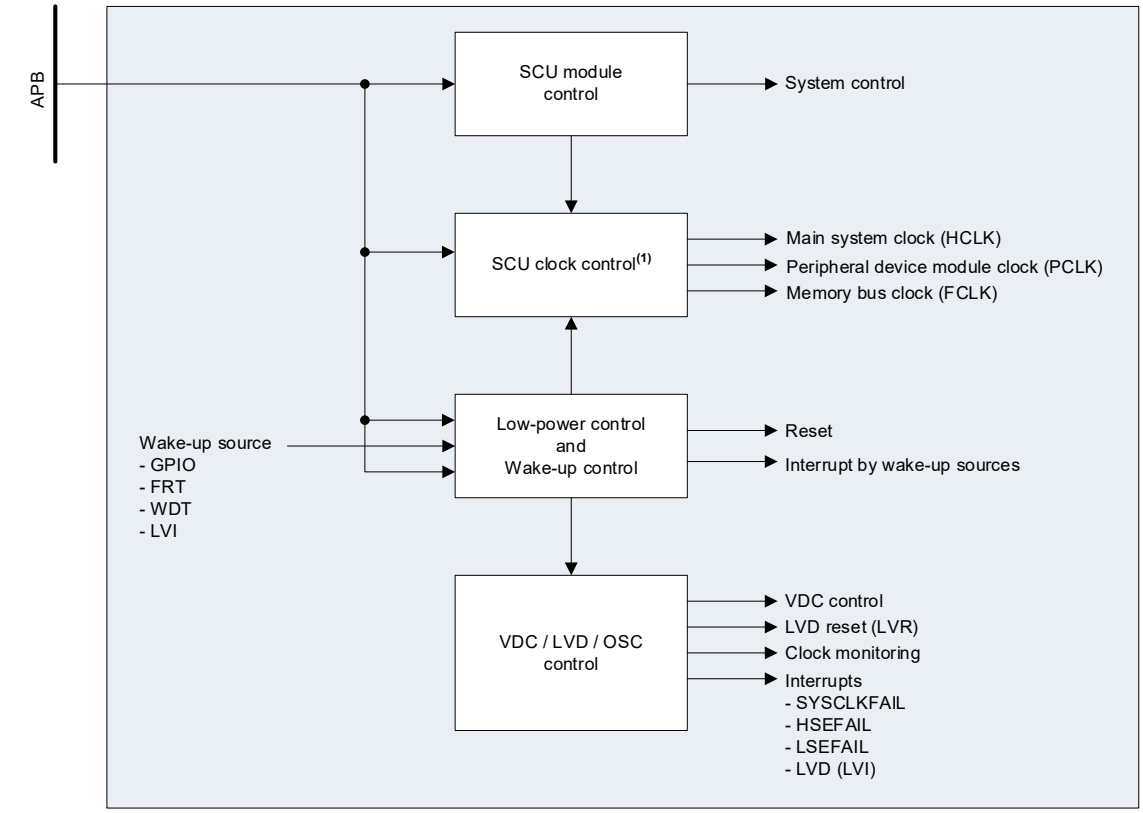
- RUN mode
- SLEEP mode
- DEEP-SLEEP (STOP1, STOP2) mode

4.3 SCU Functional Description

4.3.1 SCU Block Diagram

Figure 9 describes the SCU in a block diagram.

Figure 9. SCU Block Diagram



NOTE:

1. For details on the SCU clock control, see section 4.4.2.

4.3.2 SCU Pin and Type

Six pins in Table 7 are assigned for the SCU module.

Table 7. SCU Pins

Pin Name	Type	Description
nRESET	Input	External reset input
XIN/XOUT	OSC	External crystal oscillator (4 MHz to 16 MHz)
SXIN/SXOUT	OSC	External sub-crystal oscillator (32.768 kHz)
CLKO	Output	Clock output monitoring signal

4.4 Clocks

4.4.1 Overview

The A34G43x series has a clock generation circuit. The clock generation circuit consists of external oscillation circuit, PLL circuit, and internal oscillation circuit. The overview of each circuit is as follows:

- External oscillation circuit
  - High-Speed External Oscillator (HSE): 4 MHz to 16 MHz
    - Generates main clocks by connecting oscillator to XIN pin and XOUT pin.
    - It is recommended to use 8 MHz.
  - Low-Speed External Oscillator (LSE): 32.768 kHz
    - Generates sub clocks by connecting oscillator to SXIN pin and SXOUT pin.
- PLL circuit
  - Generates clocks by multiplying clocks which are generated in the external oscillation, internal oscillation circuit.
  - Phase-locked loop (PLL) frequency generator generates a high-speed clock (Up to 200 MHz)
- Internal oscillation circuit
  - High-speed internal oscillator (HSI): 16 MHz
  - Low-speed internal oscillator (LSI): 500 kHz

NOTE:

1. After the reset is released, the 4 MHz HSI signal, derived from the 16 MHz HSI signal divided by 4, and a 500 kHz LSI signal are automatically activated as the internal oscillator.

Table 8 describes the clock sources of the A34G43x.

Table 8. Clock Sources

Clock Name	Frequency	Description
HSE	4 to 16 MHz	High-speed external oscillator
PLL	8 to 200 MHz	Phase-locked loop
LSE	32.768 kHz	Low-speed external oscillator
HSI	16 MHz	High-speed internal oscillator
LSI	500 kHz	Low-speed internal oscillator

Five different clock sources can be used to drive the system clock:

- Low-Speed Internal oscillator (LSI)
- Low-Speed External oscillator (LSE)
- High-Speed Internal oscillator (HSI)
- High-Speed external oscillator (HSE)
- Phase-Locked Loop (PLL)

Peripheral clock features are as follows:

Table 9 describes the peripheral clock and the register selection:

Table 9. Peripheral Clock Selection

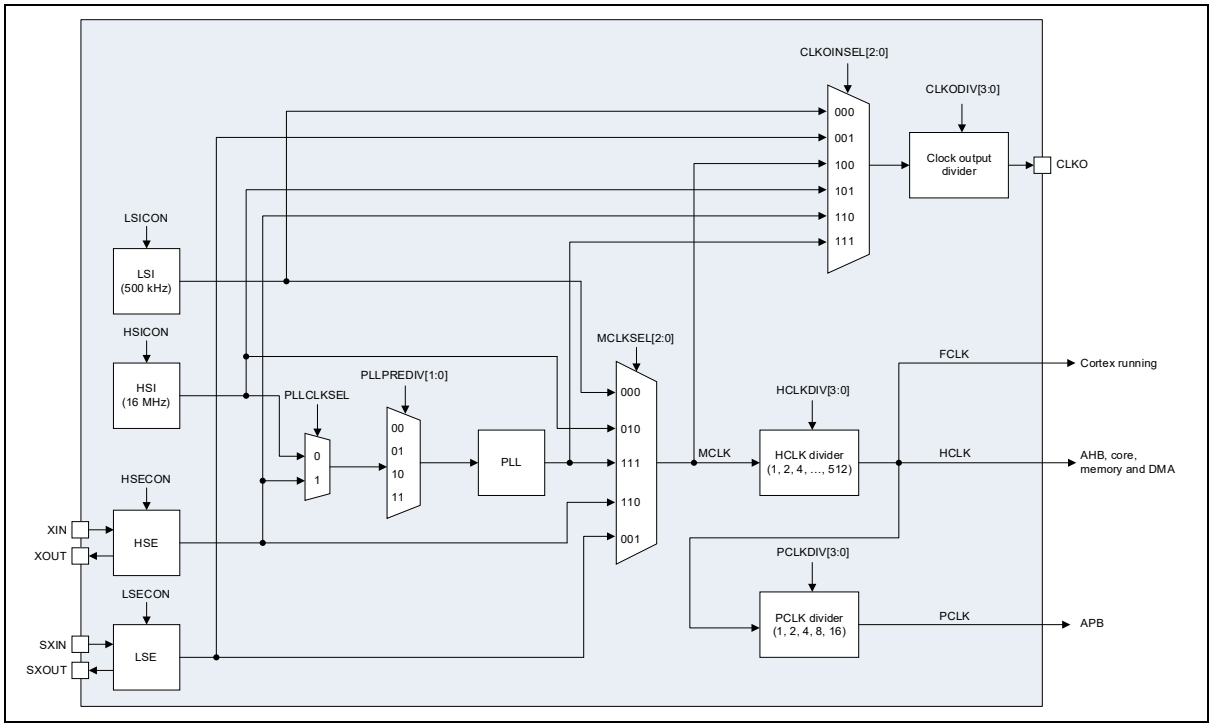
Peripheral	PCLK	Miscellaneous Clocks						
		SCU_MCCRx	MCLK	LSI	LSE	HSI	HSE	PLL
Systick	N/A	SCU_MCCR1	✓	✓	✓	✓	✓	✓
WDT	✓		✓	✓	✓	✓	✓	✓
TIMERn (n = 0 to 4)	✓	SCU_MCCR3	✓	✓	✓	✓	✓	✓
TIMERn (n = 5 to 9)	✓		✓	✓	✓	✓	✓	✓
PGAD	N/A	SCU_MCCR4	✓	✓	✓	✓	✓	✓
PGBD	N/A	SCU_MCCR5	✓	✓	✓	✓	✓	✓
PGCD	N/A		✓	✓	✓	✓	✓	✓
FRT0	N/A	SCU_MCCR6	✓	✓	✓	✓	✓	✓
FRT1	N/A		✓	✓	✓	✓	✓	✓
UART	N/A	SCU_MCCR7	✓	✓	✓	✓	✓	✓
CANFD0	N/A	SCU_MCCR8	✓	N/A	N/A	N/A	N/A	✓
CANFD1	N/A	SCU_MCCR8	✓	N/A	N/A	N/A	N/A	✓

4.4.2 Clock Tree Configuration

Users can maintain the clock system variation under software control. Figure 10 allows the users to learn about the clock system on the A34G43x.

A34G43x has two operation clocks. One is HCLK, which supplies the clock to the CPU and AHB bus systems. The other one is PCLK, which supplies the clock to the peripheral systems. Users can control this clock system variation by configuring internal registers.

Figure 10. Clock Configuration



Each clock multiplexer (selecting a clock source) has a glitch free circuit. So, a clock can be changed without glitch risks. When users change the clock mux control, be sure that both clocks (current clock and new clock) must be operating. If either is not alive, the clock change process will stop, and the system will shut down and will not recover.

Five pins in Table 10 are assigned for the clock pins.

Table 10. Clock Generation Circuit Input/Output Pins

Pin Name	I/O	Description
XIN	Input	External crystal oscillator (4 MHz to 16 MHz)
XOUT	Output	
SXIN	Input	External sub-crystal oscillator (32.768 kHz)
SXOUT	Output	
CLKO	Output	Clock output monitoring signal

4.4.3 HSE Clock

The HSE clock signal can be generated from two clock sources below:

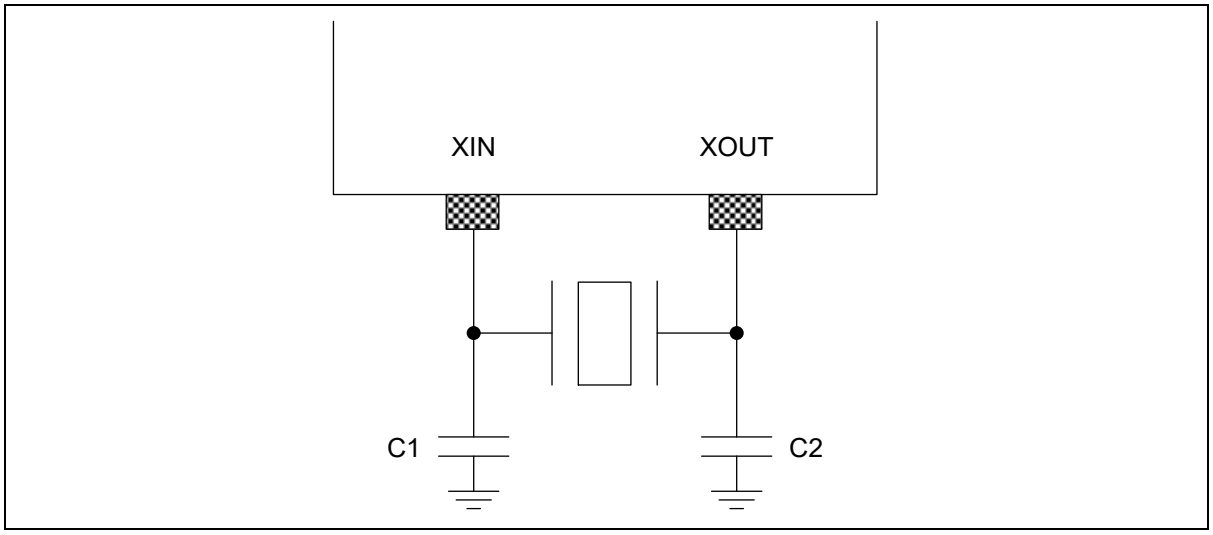
Two pins in Table 11 are assigned for the HSE clock pins block.

- HSE external crystal/ceramic oscillator

Table 11. HSE Clock Pins

Pin Name	I/O	Description
XIN	Input	• 4 MHz to 16 MHz • These pins are used to connect a crystal or ceramic oscillator. • For details, refer to Figure 11.
XOUT	Output	

Figure 11. HSE Crystal/Ceramic Oscillator



NOTE:

1. If the HSECON bit is enabled (SCU_CSCR) when the HSE (8 MHz) is not used, it can cause internal damage to the microcontroller. Disable the HSE clock source in the SCU_CSCR register when the external HSE (8 MHz) is not being used.

4.4.4 HSI Clock

The HSI is generated from the internal 16 MHz oscillator.

- HSI: 16 MHz
  - ±3.0% @ −40ºC to +85ºC
  - ±5.0% @ −40ºC to +105ºC

4.4.5 PLL Clock

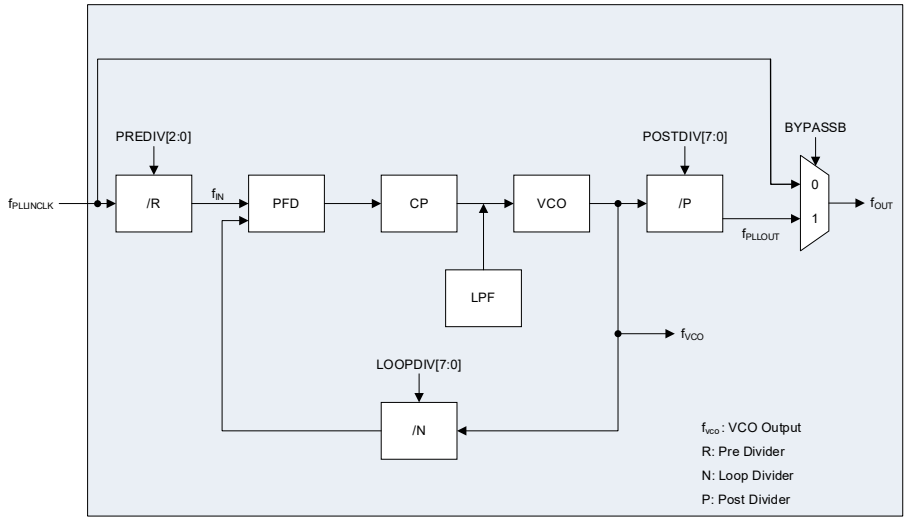
Phase-locked Loop (PLL) can be used to multiply the HSI or HSE output clock frequencies.

The PLL frequency generator generates a high-speed clock (up to 200 MHz). For more information on the PLL frequency generator, refer to Figure 10 and section 4.8.23.

Users must complete the PLL configuration (Selection of the input clock and multiplication factor) before enabling the PLL.

Once PLL is enabled, these parameters cannot be changed.

Figure 12. PLL Block Diagram



NOTES:

1. PLLEN ='1' to enable PLL block & set PLL frequency.
2. Wait 500 us for the stabilization of PLL output.
3. Wait until PLLLOCK (= LOCKSTS) flag occurs.
4. POSTDIV (P) must be at least '1'.
5. The output calculation formula is as follows:

$$f_{\mathrm{PLLOUT}} = \frac{f_{\mathrm{PLLINCLK}} \times (\mathrm{N}+1)}{(\mathrm{R}+1) \times (\mathrm{P}+1)}$$

Register	Bit	Symbol	Description
SCU_PLLCON	PREDIV[2:0]	R	PLLINCLK pre-divider value
	LOOPDIV[7:0]	N	Loop divider value
	POSTDIV[7:0]	P	Post divider value

4.4.5.1 PLL Output Frequency Settings

For the A34G43x, the PLL's output frequency f_{PLLOUT} can be precisely set in 1 MHz steps. The formula below calculates the input frequency (f_{IN}) fed to the PLL's VCO. The frequency range from 2 MHz to 3 MHz is supported, but it is recommended to set the f_{IN} to 2 MHz if possible.

$$f_{IN} = \frac{PLLINCLK}{(R+1)}, \qquad 2\ MHz \le f_{IN} \le 3\ MHz\ (Recommended\ f_{IN} = 2\ MHz)$$

At this time, the range of VCO frequency, f_{VCO} should be set to Max. VCO frequency (Max. f_{VCO}) or less, and the calculation formula is shown below. (Refer to PLL electrical characteristics in Datasheet.):

$$f_{\mathrm{VCO}}\ (\mathrm{MHz}) = f_{\mathrm{IN}} \times (\mathrm{N}_1 + 1), f_{\mathrm{VCO}} \le \mathrm{Max.}\, f_{\mathrm{VCO}}$$

As a result, the final frequency of the PLL block, f_{PLLOUT} can be obtained from the formula below:

$$f_{\mathrm{PLLOUT}}\ [\mathrm{MHz}] = \frac{f_{\mathrm{PLLINCLK}} \times (\mathrm{N}+1)}{(\mathrm{R}+1) \times (\mathrm{P}+1)}$$

$$= \frac{f_{\mathrm{IN}} \times (\mathrm{N}+1)}{(\mathrm{P}+1)}$$

$$= \frac{f_{\mathrm{VCO}}}{(\mathrm{P}+1)}$$

NOTE:

1. If the main clock is changed from HSI (4 MHz) to PLL, it is recommended to change the clock gradually in the order below section 4.4.5.2:

4.4.5.2 Example PLL Clock Change

1. Once the microcontroller is powered on, the system clock source is HSI (4 MHz).
2. Initialize data cache and instruction cache, and then set flash wait to ‘10’.
3. Set XIN and XOUT on pin MUX to use the HSE.
4. Enable the HSE.
5. Set an additional time of about 10 ms for clock stabilization.
6. Change the system clock source to the HSE.
7. Enable the PLL. Set the PLL options.
8. Set additional time of about 500 µs for clock stabilization.
9. Set the HCLK to the MCLK divided by 2, and the PCLK to the HCLK.
10. Change the system clock source to the PLL.
11. Set the HCLK to the MCLK divided by ‘1’, and the PCLK to the HCLK.
12. Set flash wait to ‘5’.

4.4.6 LSE Clock

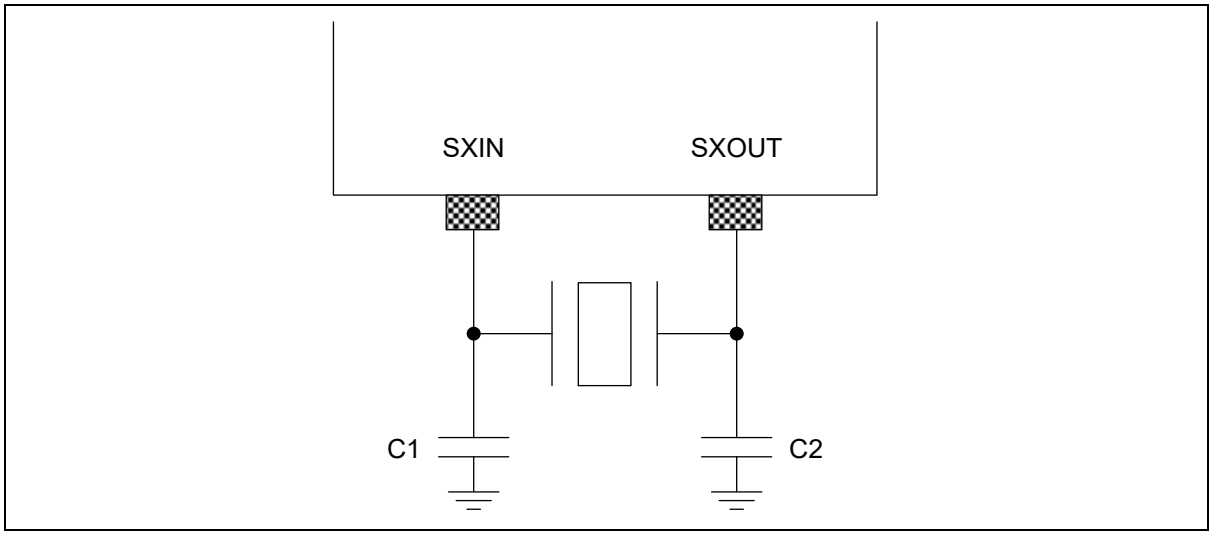
The Low-Speed External Oscillator (LSE) crystal is a 32.768 kHz crystal or ceramic resonator. It has the advantage of providing a low power but highly accurate clock source to the clock peripheral for clock/calendar or other timing functions.

- LSE: 32.768 kHz

Table 12. LSE Clock Pins

Pin Name	I/O	Description
SXIN	Input	These pins are used to connect a 32.768 kHz crystal resonator
SXOUT	Output	

Figure 13. LSE External Clock



NOTE:

1. If the LSECON bit is enabled (SCU_CSCR) when the LSE (32.768 kHz) is not used, it can cause internal damage to the microcontroller. Disable the LSE clock source in the SCU_CSCR register when the external LSE (32.768 kHz) is not being used.

4.4.7 LSI Clock

After the microcontroller is powered on, the LSI (500 kHz) is initially enabled by default in the system operation sequence.

For more details, refer to the electrical characteristics chapter in A34G43x series.

- LSI: 500 kHz
  - ±30% @ −40ºC to +105ºC

4.4.8 Two Main Operating Clocks: HCLK and PCLK

A34G43x has two main operating clocks. One is HCLK, which generates clock signals both for CPU and AHB system; the other is PCLK, which generates clock signals for peripheral systems.

4.4.8.1 HCLK Clock

The Cortex-M4F CPU requires two clocks, the HCLK and FCLK. The HCLK is fed to the core and AHB. The FCLK stays enabled except in DEEP-SLEEP mode, whereas the HCLK can be disabled in SLEEP mode.

The buses and memories are clocked by the HCLK. As the bus clock frequency is limited to a maximum of 200 MHz, the HCLK frequency must not exceed 200 MHz.

4.4.8.2 PCLK Clock

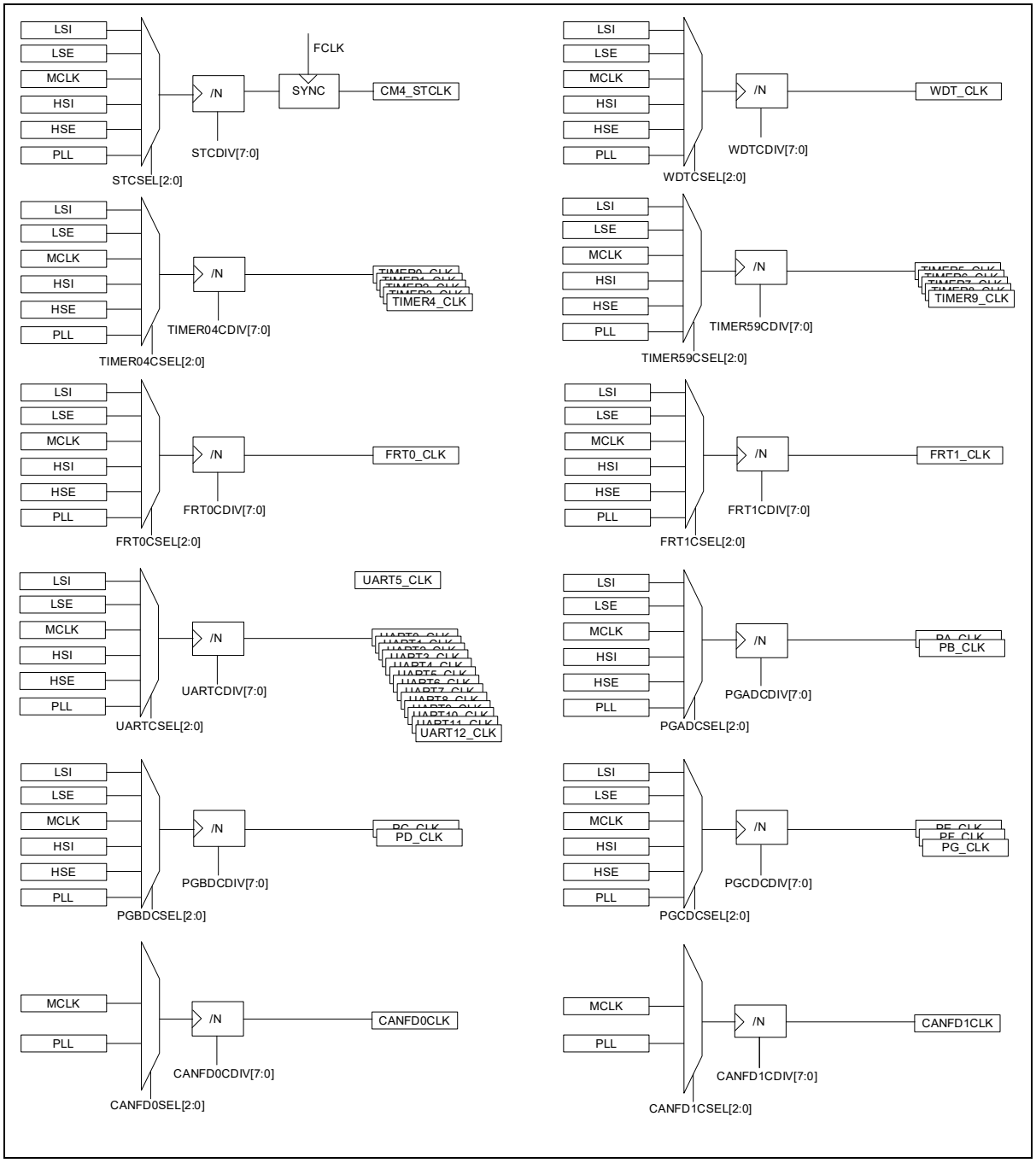
The PCLK can be used as a clock for any peripherals. Enable the PCLK for each peripheral is determined with the SCU_PCER registers; each peripheral block’s registers cannot be read unless its PCLK input is enabled. And the PCLK stops operating in DEEP-SLEEP mode.

4.4.9 Configuration of Miscellaneous Clocks

The A34G43x supports the “miscellaneous clocks” feature, which allows each peripheral to be assigned with a different clock source (MCLK, HSE, LSE, PLL, or HSI) at a different frequency division ratio.

Users can set each peripheral’s clock source and its frequency divider in the corresponding SCU_MCCRx register. The supported division ratio can be one ranging from 1 to 255.

Figure 14. Miscellaneous Clock Configuration



4.4.10 Peripheral Clocks

Table 13 describes the peripheral clock and the register selection:

Table 13. Peripheral Clock Selection

Peripheral	SCU_MCCRx (x = 1 to 8)	PCLK
Systick	SCU_MCCR1	N/A
WDT		✓
TIMERn (n = 0 to 4)	SCU_MCCR3	✓
TIMERn (n = 5 to 9)		✓
PGAD	SCU_MCCR4	N/A
PGBD	SCU_MCCR5	N/A
PGCD		N/A
FRT0	SCU_MCCR6	N/A
FRT1		N/A
UARTn (n = 0 to 12)	SCU_MCCR7	N/A
CANFD0	SCU_MCCR8	N/A
CANFD1		N/A

4.4.11 Clock Monitoring

The A34G43x provides clock monitoring functions used to detect whether the microcontroller's clock source is stopped. Using the clock monitoring function, if the microcontroller's clock source is detected to be stopped, it is possible to automatically switch to the alternative internal clock source, HSI 4 MHz (16 MHz / 4) or reset the system. (For reference, when the external crystal such as the HSE and LSE is damaged or an external clock signal is stopped, the A34G43x generates error flags or interrupt events to check the status and enter recovery mode to put the application in a secure state.)

The speed of the internal clock and external clock may be different when using the clock monitoring function, so it is required to adjust the clock speed to meet the requirements for normal operation in the interrupt functions.

NOTES:

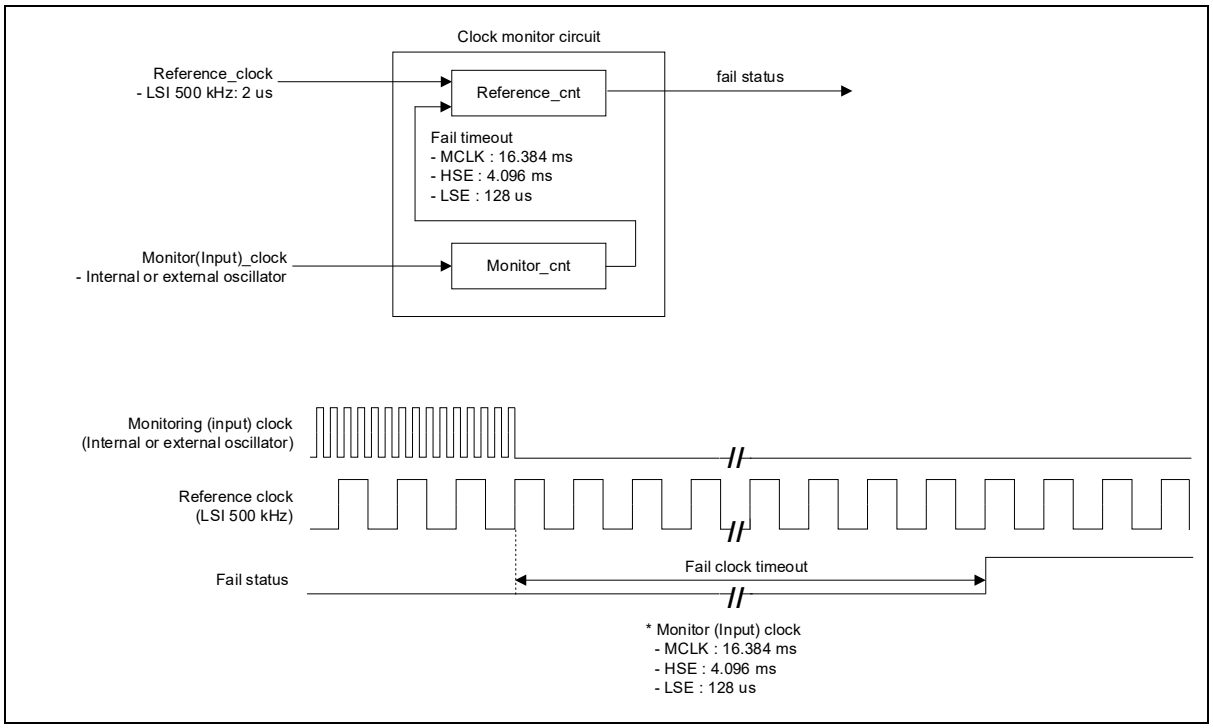
1. When using MCLK as an HSE PLL, the MCLK fail reset function cannot be used. In this case, the MCLK fail reset function can be supplemented by activating the HSE fail reset function. In other words, reset occurs when HSE fail occurs.
2. The interrupt function of HSE and LSE clock monitoring is available only when HSE and LSE are not used as MCLK.

4.4.11.1 Detection Range

The clock monitoring functions detect bad oscillators by comparing the external or internal oscillator with the main clock monitoring sample clock. The LSI is used as a reference clock, and error signals are generated due to the reference counter timeout. The timeout period is set differently for MCLK, HSE, and LSE.

Figure 15 shows that the LSI is set as a reference clock and the timeout period required to detect oscillation failure operation when performing the clock monitoring is different for each clock source.

Figure 15. Clock Monitoring Detection Range



4.4.11.2 Clock Monitoring Functions and Clock Sources

Table 14 shows clock monitoring functions provided by the A34G43x.

Table 14. Clock Monitoring Function

Clock Monitoring Function		Enable Bit Name
Automatic recovery of MCLK errors enable bit		MCLKREC
MCLK	Monitoring bit	MCLKMNT
	Interrupt enable bit	MCLKIE
HSE	Monitoring bit	HSEMNT
	Interrupt enable bit	HSEIE
LSE	Monitoring bit	LSEMNT
	Interrupt enable bit	LSEIE

Table 15 shows clock sources and their frequency characteristics that the A34G43x can monitor. After each clock source is enabled, stabilization delay is required for stable frequency oscillation.

Table 15. Clock Source for Clock Monitoring

Clock Source	Frequency	Stabilization Delay
HSE	4 to 16 MHz	10 ms
PLL	8 to 200 MHz	500 µs
LSE	32.768 kHz	1 s
LSI	500 kHz	100 µs
HSI	16 MHz	100 µs

4.4.11.3 MCLK Monitoring

MCLK (main system clock) refers to the clock on which the microcontroller operates. The MCLK can be set to a clock generated by an external oscillating element, or a clock generated by a microcontroller internal oscillation circuit.

To monitor the MCLK clock signal, follow the procedure below:

1. Set the HSECON and HSICON bits in the SCU_CSCR register to enable a clock source set to MCLK and wait for a stable time. (HSE: 10 ms, PLL: 500 μs, HSI: 100 μs)
2. Select the main system clock source by configuring the MCLKSEL[2:0] in the SCU_SCCR register. (1)(2)(3)
3. Set the MCLKMNT bit in the SCU_CMR register to '1' to activate the MCLK monitoring function. MCLK errors can be detected by the clock monitoring circuit operating internally at LSI 500 kHz.
4. To use the error detection interrupt function, set the MCLKIE in SCU_CMR register to '1'. If an error occurs with an interrupt enabled, the SYSCLKFAIL interrupt is generated. The error detection status can be identified by the MCLKFAIL flag in SCU_CMR register. The MCLKFAIL flag is cleared by writing '1'.
5. To disable the automatic switch of the main clock to HSI when an error is detected, set the MCLKREC in SCU_CMR register to '1'. If the main clock error occurs, the MCLKSTS is updated to '1'. The MCLKSTS is read-only and is updated according to the main clock status.
6. After switching the main clock to HSI, to use the clock source that was used before the error occurred, set the clock setting registers again. (The user can use the SCU_CSCR and SCU_SCCR registers to change clock frequencies after switching the clock and configuring a prescaler.)
7. Set the MCLKFRST bit in the SCU_RSER register to '1' to perform the microcontroller reset when the MCLK error flag occurs.
8. After the reset event occurs, the MCLKFRST flag in the SCU_RSSR register informs whether the reset event occurs in the microcontroller due to the MCLK error. The MCLKFRST flag is cleared by writing '1'.

NOTES:

1. When changing the MCLKSEL bits, both clock sources should be alive. For example, both HSI and HSE must be alive when MCLK is changed to HSE from HSI, otherwise the chip will malfunction.
2. When changing the PLLCLKSEL bits, both the HSI and HSE must be enabled. If the MCLKSEL bits are set to use the HSE, ports must be set for the functions of XIN and XOUT, respectively.
3. If the MCLKSEL bits are set to use the LSE, ports must be set for the functions of SXIN and SXOUT, respectively.

Figure 16. MCLK Monitoring Recovery Procedure

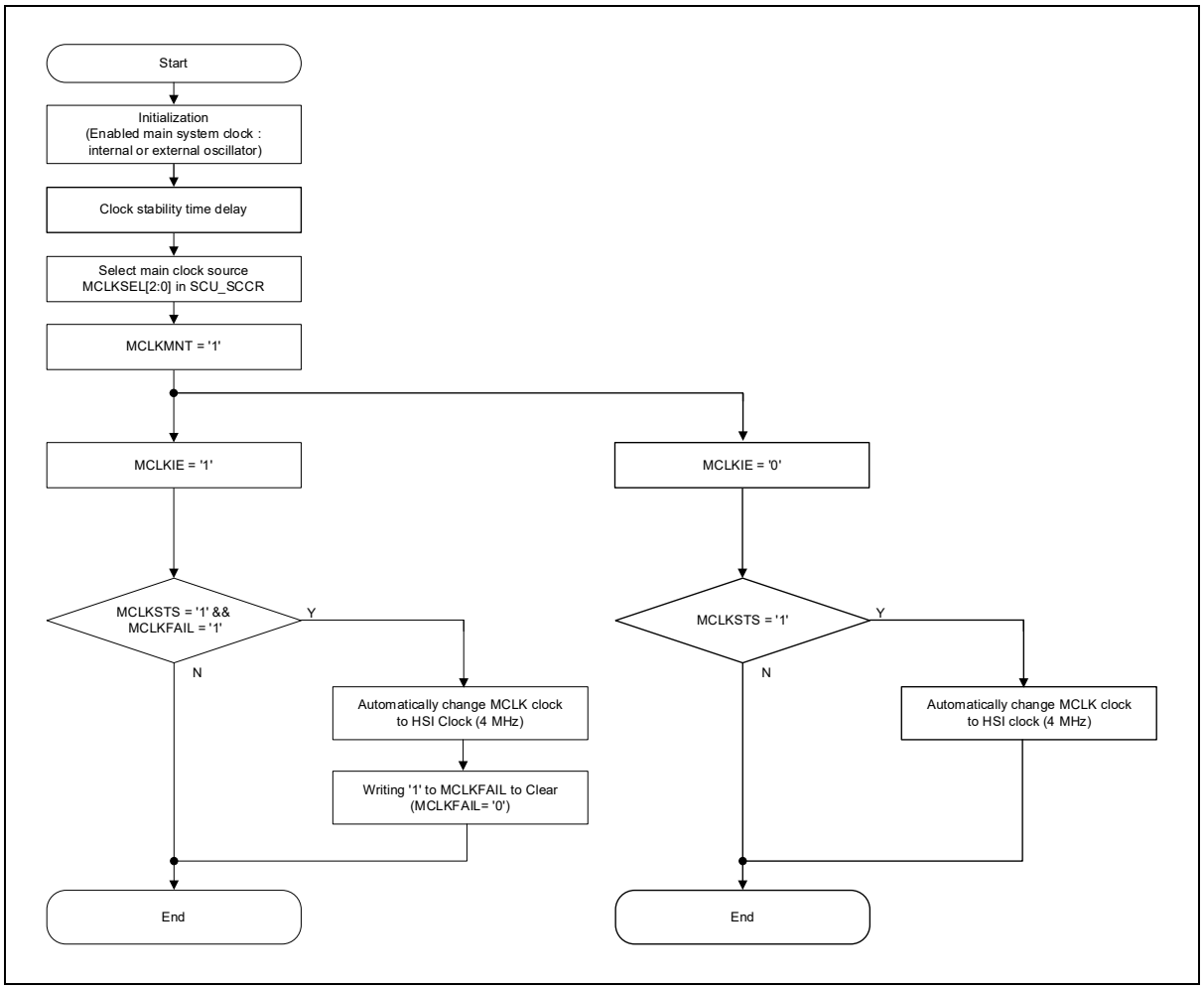
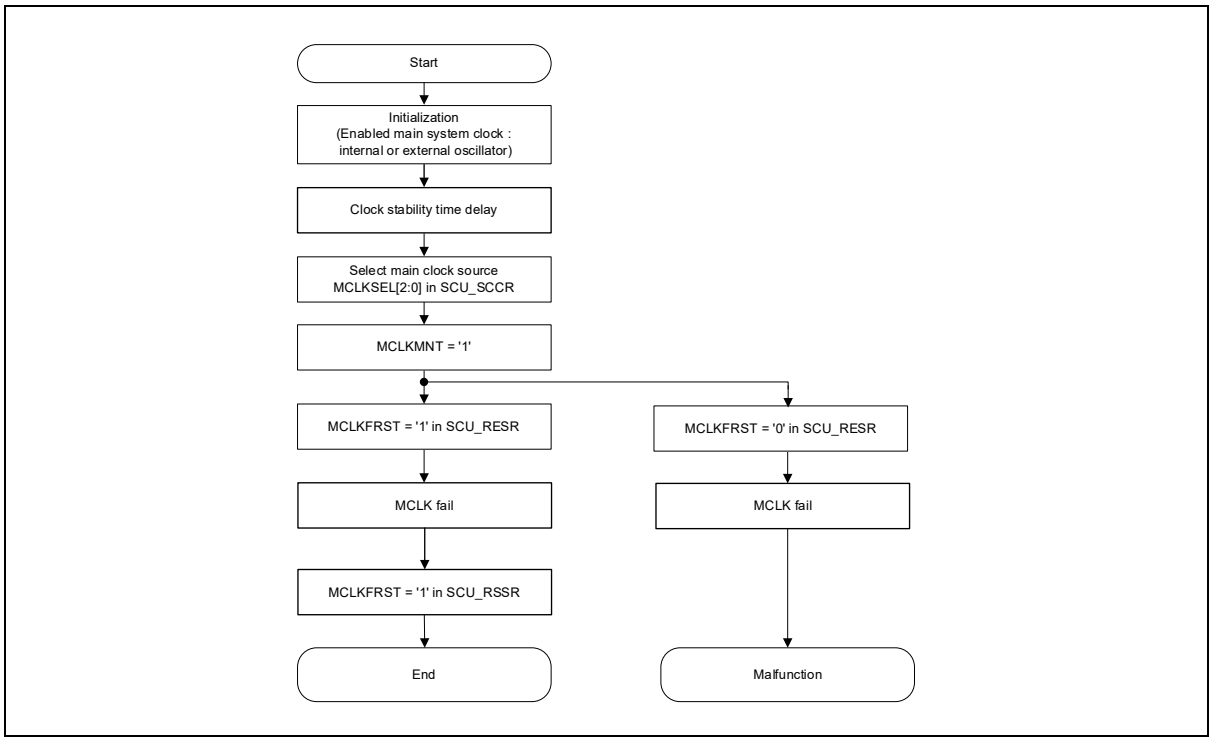


Figure 17. MCLK Monitoring Reset Procedure



4.4.11.4 HSE Monitoring

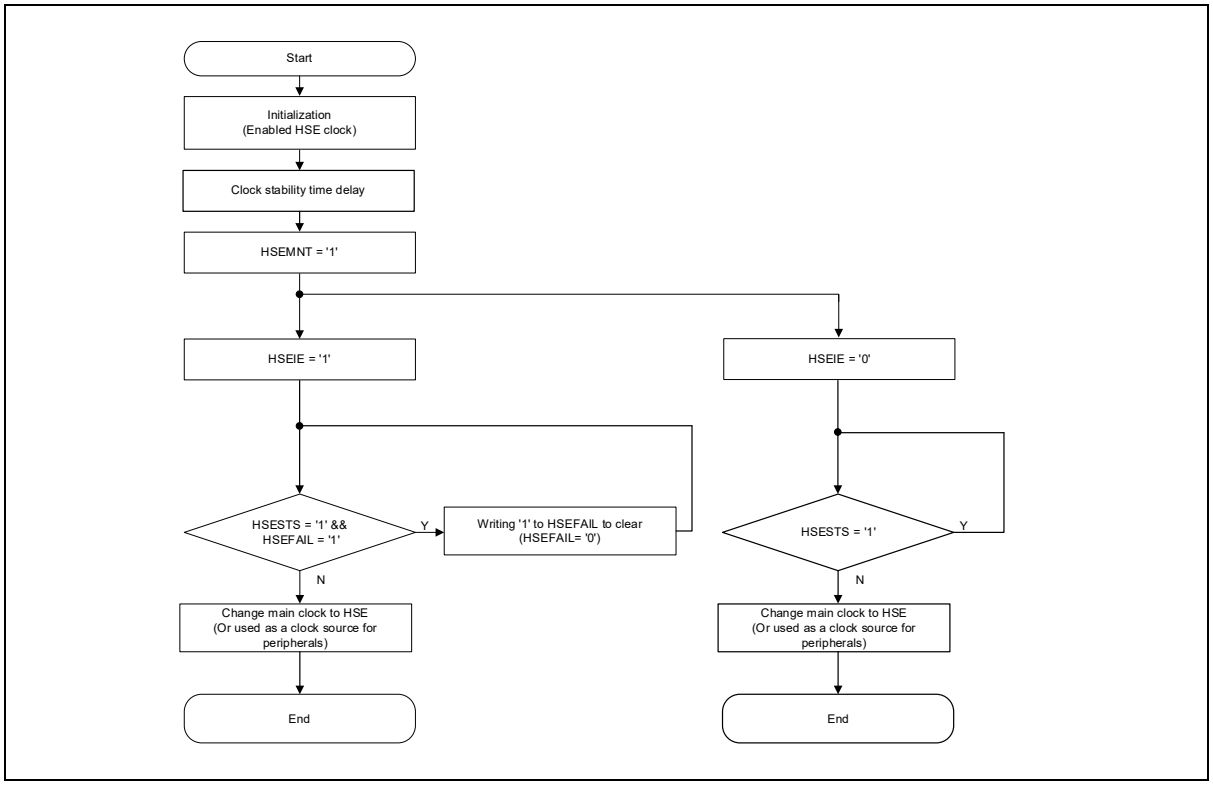
This section describes the HSE clock monitoring. For information on the HSE clock, see section 4.4.3.

To monitor the HSE clock signal, follow the procedure below:

1. Set the HSECON bit in the SCU_CSCR register to ‘1’ to enable the HSE clock source and wait approximately 10 ms for the clock to stabilize.
2. Set the HSEMNT bit in the SCU_CMR register to ‘1’ to activate the HSE monitoring function. HSE errors can be detected by the clock monitoring circuit operating internally at LSI 500 kHz.
3. To use the error detection interrupt function, set the HSEIE bit in the SCU_CMR register to ‘1’. If an error occurs with an interrupt enabled, the HSEFAIL interrupt event is generated. The interrupt error can be detected by the HSEFAIL flag in the SCU_CMR register. The HSEFAIL flag is cleared by writing ‘1’.
4. Set the HSEFRST bit in the SCU_RSER register to ‘1’ to perform the microcontroller reset event when the HSE clock error occurs.
5. After the reset event occurs, the HSEFRST flag in the SCU_RSSR register indicates whether the reset event occurs in the microcontroller due to the HSE error. The HSEFRST flag is cleared by writing ‘1’.

Figure 18 shows the procedure in which the clock monitoring circuit on the A34G43x monitors the HSE.

Figure 18.HSE Monitoring Procedure



4.4.11.5 LSE Monitoring

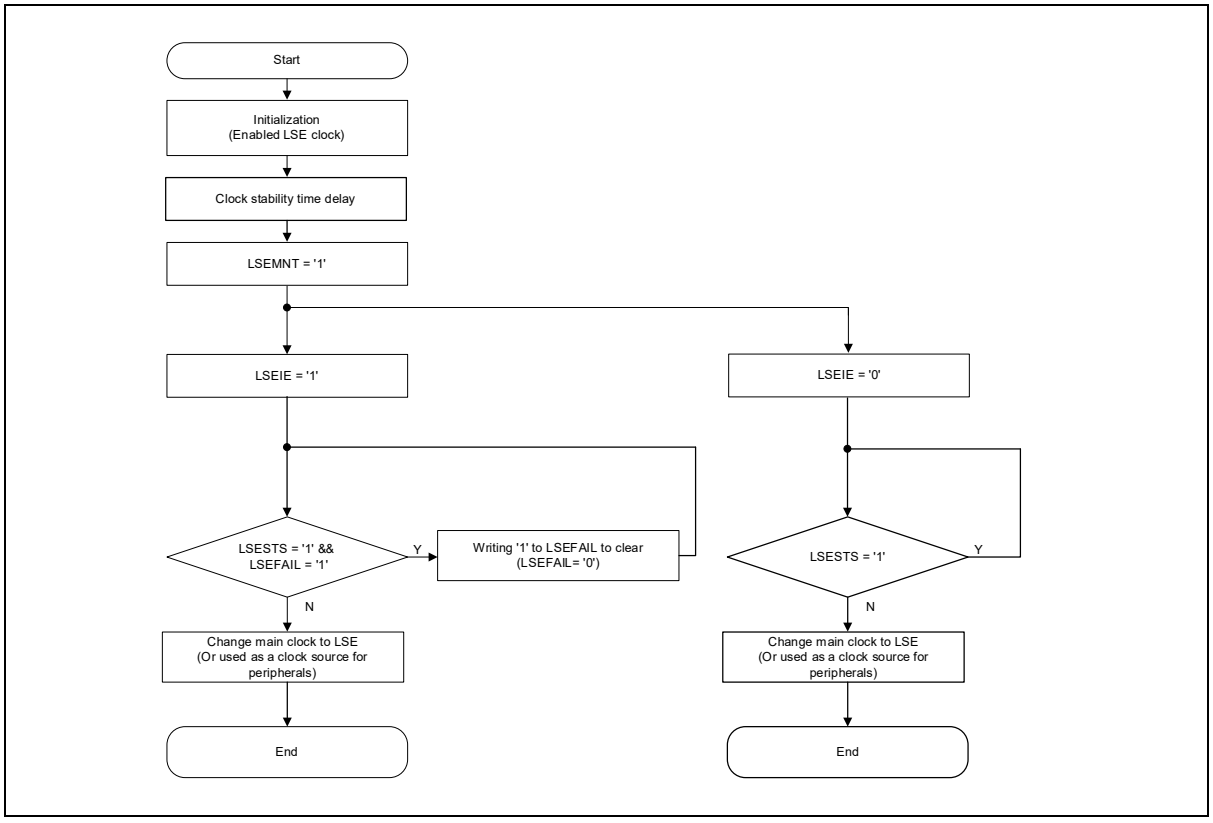
This section describes the LSE clock monitoring. For information on the LSE clock, see section 4.4.6.

To monitor the LSE clock signal, follow the procedure below:

1. Set the LSECON bit in the SCU_CSCR register to ‘1’ to enable the LSE clock source and wait approximately 1 s for the clock to stabilize.
2. Set the LSEMNT bit in the SCU_CMR register to ‘1’ to activate the LSE monitoring function. LSE errors can be detected by the clock monitoring circuit operating internally at LSI 500 kHz.
3. To use the error detection interrupt function, set the LSEIE bit in the SCU_CMR register to ‘1’. If an error occurs with an interrupt enabled, the LSEFAIL interrupt event is generated. The interrupt error can be detected by the LSEFAIL flag in the SCU_CMR register. The LSEFAIL flag is cleared by writing ‘1’.
4. Set the LSEFRST bit in the SCU_RSER register to ‘1’ to perform the microcontroller reset event when the LSE clock error occurs.
5. After the reset event occurs, the LSEFRST flag in the SCU_RSSR register informs whether the reset event occurs in the microcontroller due to the LSE error. The LSEFRST flag is cleared by writing ‘1’.

Figure 19 shows the procedure in which the clock monitoring circuit on the A34G43x monitors the LSE.

Figure 19.LSE Monitoring Procedure



4.4.12 Clock Configuration Procedure

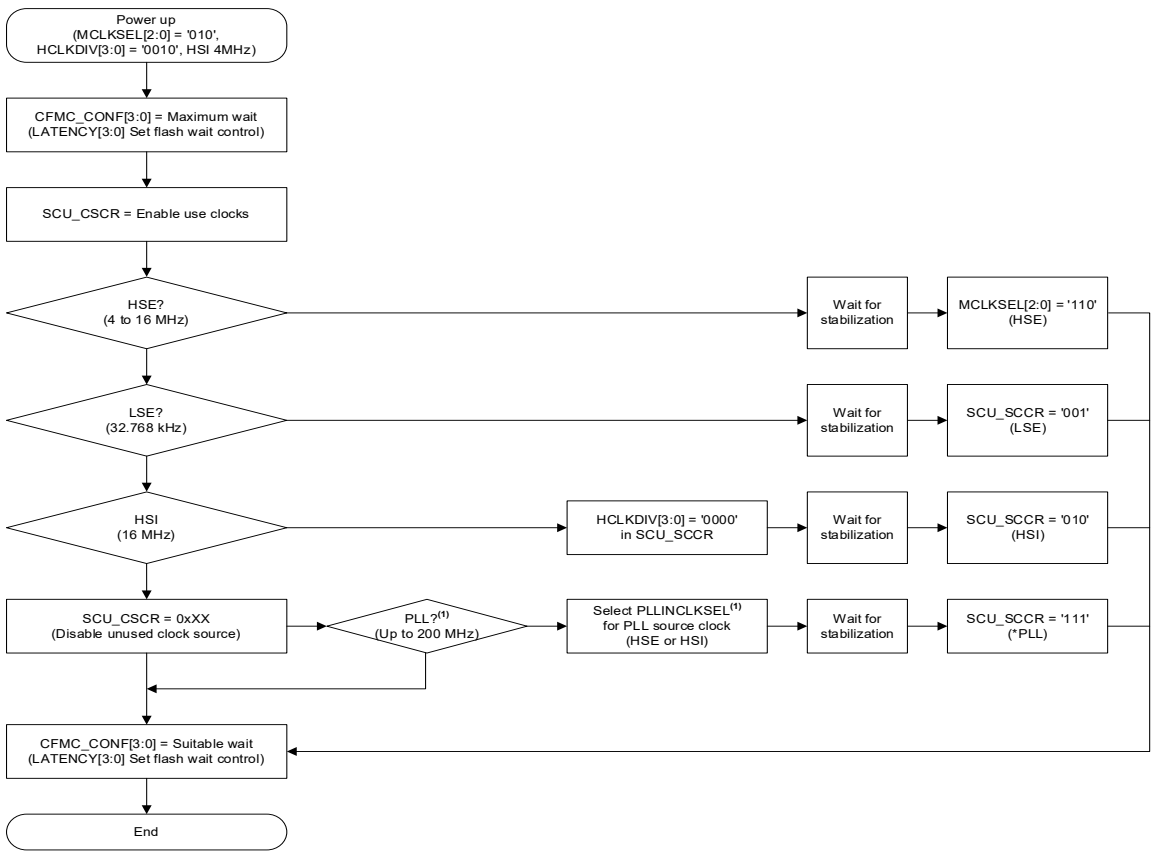
After powering on the device, the High-Speed Internal (HSI) clock of 16 MHz and the Low-Speed Internal (LSI) clock of 500 kHz are enabled by default. The HSI clock is divided by four to give a system clock of 4 MHz. The other clock sources are set by software while the system is clocked by the HSI. Users can enable the clocks using the SCU_CSCR register, which is the clock source control register.

Once the HSE block has been enabled, the users must wait for crystal oscillation to stabilize.

The secondary LSE (32.768 kHz) clock can be enabled with the SCU_CSCR register. Likewise with pins XIN and XOUT for the HSE, the LSE must be enabled after the pin mux configuration is completed for SXIN and SXOUT and then the stabilizing time is elapsed.

The MCLK can be changed with the SCU_SCCR register. Figure 20 shows an example sequence of how the system clock changes.

Figure 20. Clock Change Procedure



NOTE:

1. For details on the SCU Clock Control, see section 4.4.5 ‘PLL ’.

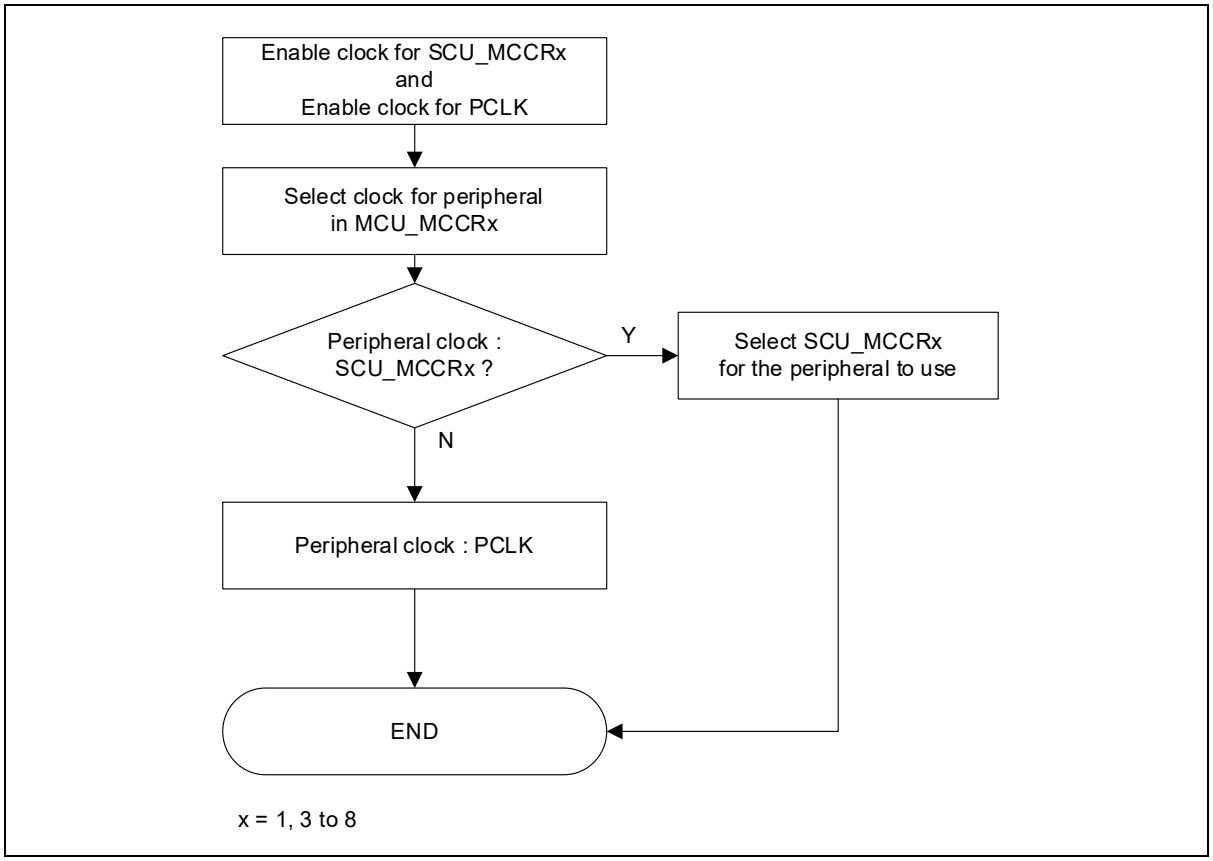
To speed up the system clock to the maximum operating frequency, users must check the flash memory wait control configuration. The flash memory read access time is a limitation factor for performance. The recommended wait control value is listed in Table 16.

Table 16. Flash Memory Wait Control Recommendation

LATENCY[3:0]	Flash Memory Access Wait	Available Max. System Clock Frequency
0000	0-clock wait	Up to 33 MHz
0001	1-clock wait	Up to 66 MHz
0010	2-clock wait	Up to 100 MHz
0011	3-clock wait	Up to 133 MHz
0100	4-clock wait	Up to 166 MHz
0101	5-clock wait	Up to 200 MHz
0110	6-clock wait	Up to 200 MHz
0111	7-clock wait	Up to 200 MHz
1000	8-clock wait	Up to 200 MHz
1001	9-clock wait	Up to 200 MHz
1010	10-clock wait	Up to 200 MHz
1011	11-clock wait	Up to 200 MHz
1100	12-clock wait	Up to 200 MHz
1101	13-clock wait	Up to 200 MHz
1110	14-clock wait	Up to 200 MHz
1111	15-clock wait	Up to 200 MHz

Figure 21 shows how to set the peripheral clock. The peripheral clocks are either set PCLK or selected by the SCU_MCCRx (x = 1, 3 to 8) registers.

Figure 21. Peripheral Clock Select



4.5 Reset

4.5.1 Overview

The reset features are as follows:

- Power-On reset (POR)
- Low-Voltage Reset (LVR)
- nRESET pin reset
- Watchdog Timer Reset (WDTR)
- Software reset
- Clock oscillating error reset
- CPU request reset
- CFMC bank reset

The A34G43x provides two reset source types. One is a cold reset, which is effective during power up or down sequence; the other is a warm reset, which is generated by several reset sources.

The I/Os state under and after reset is “analog state” (for details, see section 2.2).

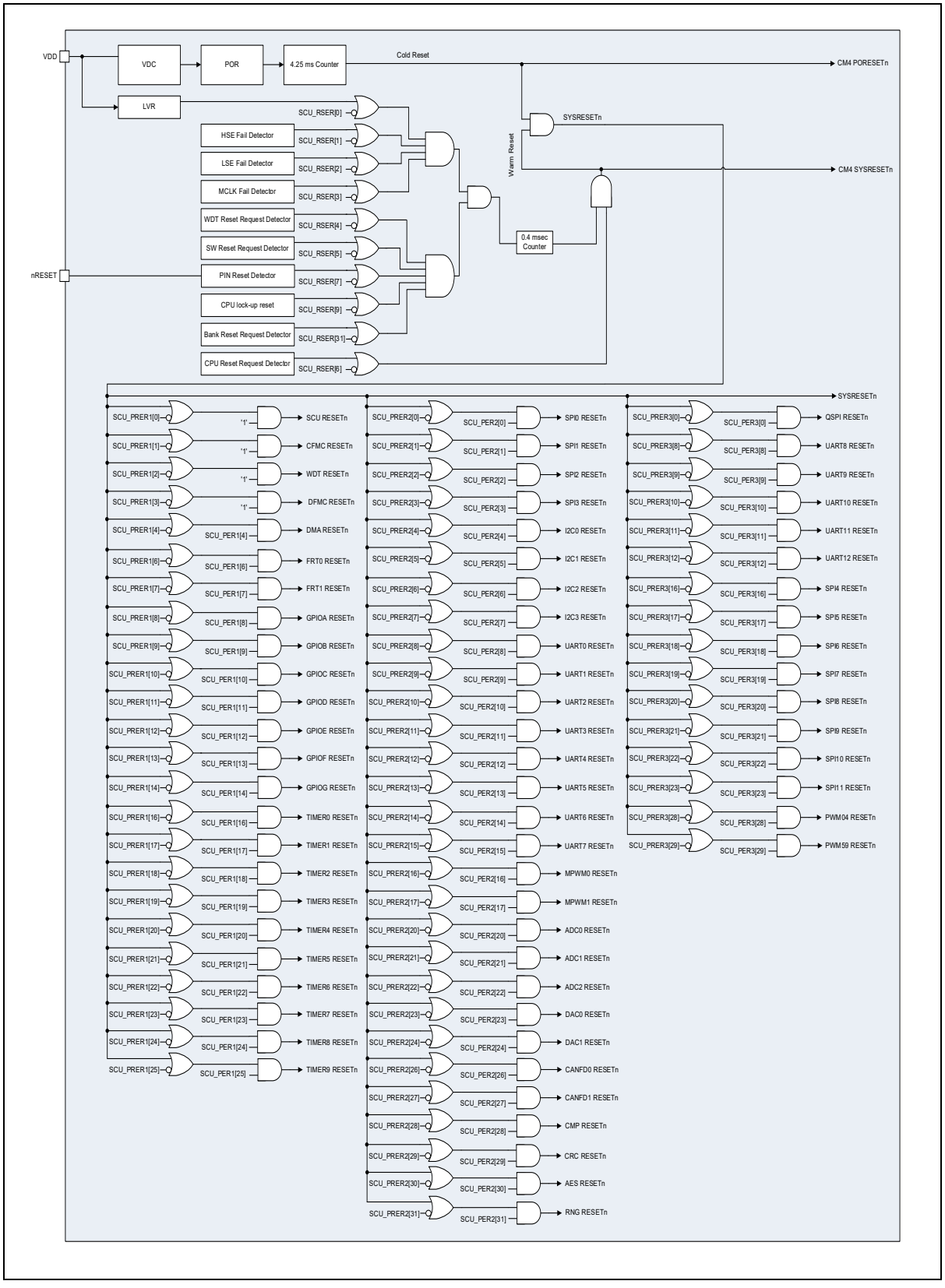
A reset event makes a chip to turn to an initial state. Reset sources of the cold reset and the warm reset are listed in Table 17.

Table 17. Reset Sources for Cold Reset and Warm Reset

Reset Type	Cold Reset	Warm Reset
Reset Source	POR (Power-On Reset)	• nRESET Pin Reset • LVD Reset • Watchdog Timer Reset • Software Reset • Clock Oscillating Error Reset • CPU Request Reset • CFMC Bank Reset

4.5.2 Reset Tree Configuration

Figure 22. Reset Tree Configuration



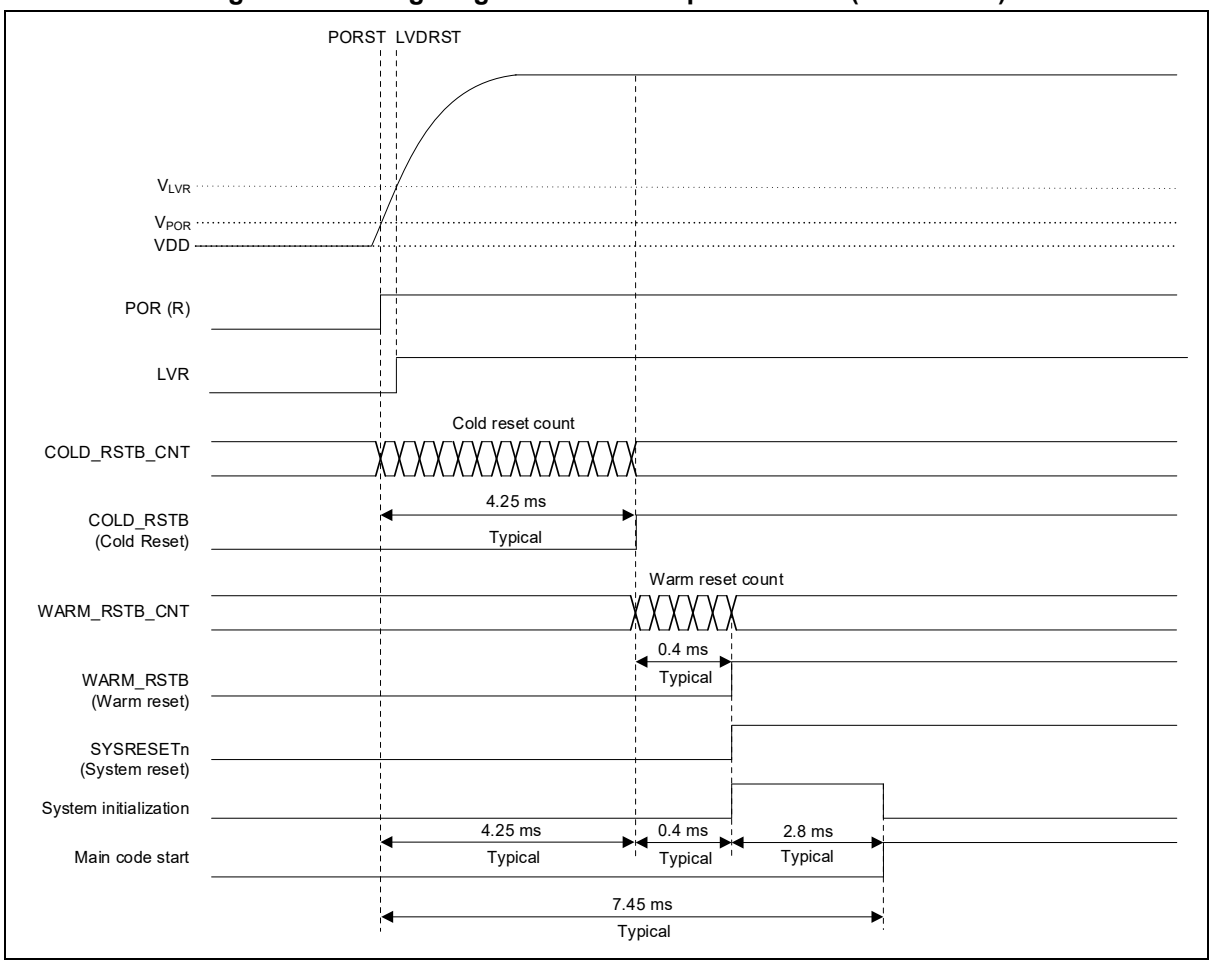
4.5.3 Cold Reset

A cold reset plays an important role during a power-up process and affects the entire process of system booting. The internal VDC becomes active as soon as the VDD is applied. The internal POR is triggered when the VDD is determined to reach 1.8 V based on the amount of the internal VDC output.

During the cold reset process, when the applied voltage exceeds 1.8 V, the LSI clock and the HSI clock are enabled. After the stabilization of the internal VDC level for 4.25 ms, the internal logic is initialized. And then, when the external VDD voltage rises above the LVD Reset voltage level (2.11 V), the cold reset is released and, after a waiting time of 0.4 ms for warm reset synchronization, booting begins.

Figure 23 shows the waveform of power up sequence and initial reset.

Figure 23. Timing Diagram of Power-Up Procedure (Cold Reset)



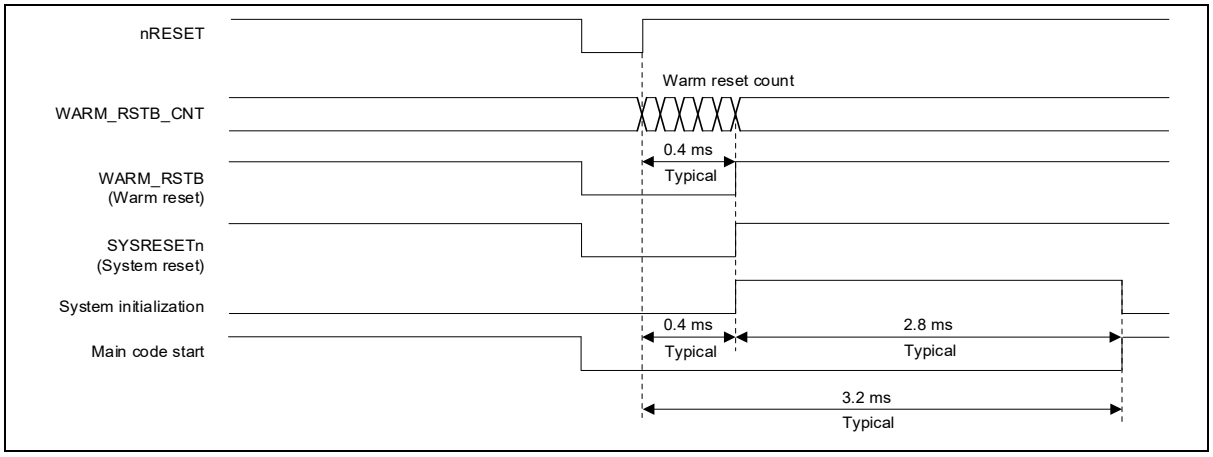
4.5.4 Warm Reset

A warm reset event is triggered for system initialization when the conditions of a reset source internally set are satisfied.

Warm reset sources are enabled or disabled by configuring the SCU_RSER register, and their occurrence is written to the SCU_RSSR register.

Which devices are to be initialized by the warm reset is determined by settings of the SCU_PRER register. Using this register, users can allow or disallow the initialization of individual peripheral.

Figure 24. Warm Reset Timing Diagram



4.5.5 LVD Reset (LVR)

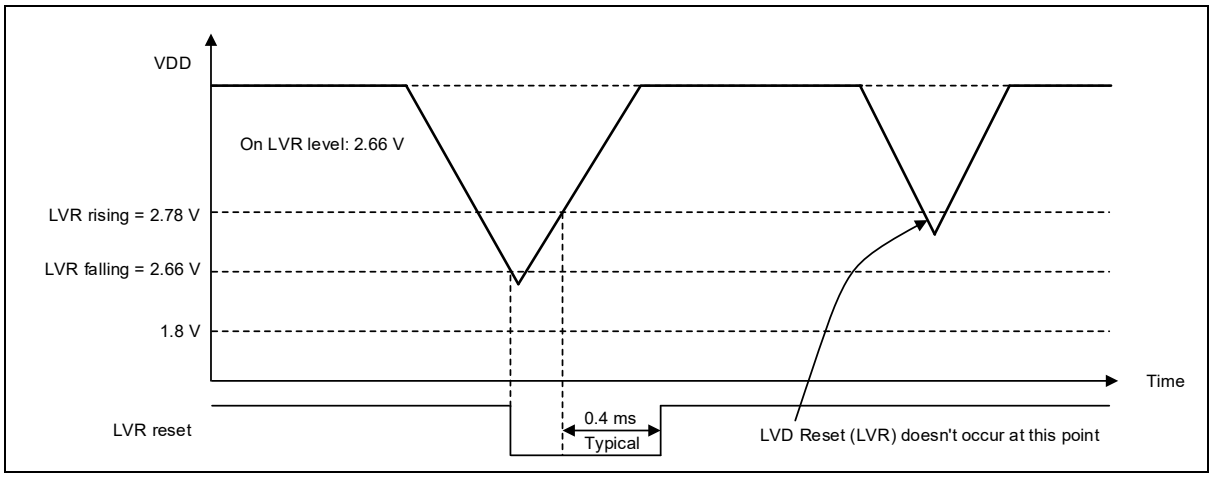
A Low Voltage Detect Reset (LVR) event is triggered when the operating voltage drops below a certain level during the microcontroller's operation. Users can set the microcontroller whether to perform a reset or to perform an interrupt, when an LVR event is triggered.

Table 18 describes the range of LVR levels and the register for LVR level control.

Table 18. LVR Level

LVR Level Range	Description
1.57 V to 4.47 V	• LVR level (16 steps) • This LVR level is set by the LVRVS[3:0] bits in the SCU_LVRCR register.

Figure 25. LVR Timing Diagram



4.5.6 nRESET Pin Reset

This reset signal is generated on the nRESET pin. When the nRESET pin is driven low, all the processing in progress is aborted and the microcontroller enters reset state.

The nRESET pin in Table 19 is assigned for RESET.

Table 19. nRESET Pin

Pin Name	Type	Description
nRESET	Input	External reset input

4.5.7 Watchdog Timer Reset

Watchdog Timer (WDT) monitors the operation of the microcontroller and is typically used to detect software errors. When the microcontroller becomes uncontrollable due to a malfunction, the WDT resets the microcontroller to recover it.

The A34G43x has one WDT module built in, which functions as a 32 bit down-counter. Once the WDT counts down to zero while operating as a reset source, the microcontroller will restart.

For details on the Watchdog Timer Reset, refer to chapter 8.

4.5.8 Software Reset

A system reset sets all registers to their reset values except the reset flags in the clock system reset control the SWRST bit in the SCU_SRCR register and the registers in the SCU block(1).

NOTE:

1. Users must use the SWRST bit in the SCU_SRCR register instead of NVIC_SystemReset() or the SCB_AIRCR register setting for microcontroller reset

4.5.9 HSE Fail Reset

The HSEFRST in SCU_RSER register allows users to configure input signals that trigger a reset event.

Setting the HSEFRST bit to ‘1’ enables its corresponding reset signal to trigger a reset; setting it to ‘0’ disables the reset signal, thereby masking the reset event.

The HSEFRST bit in the SCU_RSSR register records occurrences of reset events. A bit read as ‘1’ means that its corresponding reset source has triggered a reset.

4.5.10 LSE Fail Reset

The LSEFRST bit in the SCU_RSER register allows users to configure input signals that trigger a reset event.

Setting a bit to ‘1’ enables its corresponding reset signal to trigger a reset; setting it to ‘0’ disables the reset signal, thereby masking the reset event.

The LSEFRST bit in the SCU_RSSR register records occurrences of reset events. A bit read as ‘1’ means that its corresponding reset source has triggered a reset.

4.5.11 CPU Request Reset

The CPURST bit in the SCU_RSER register allows users to configure input signals that trigger a reset event.

Setting a bit to ‘1’ enables its corresponding reset signal to trigger a reset; setting it to ‘0’ disables the reset signal, thereby masking the reset event.

The CPURST bit in the SCU_RSSR register records occurrences of reset events. A bit read as ‘1’ means that its corresponding reset source has triggered a reset.

4.5.12 CFMC Bank Reset

The BANKRST in SCU_RSER register allows users to configure input signals that trigger a reset event.

Setting a bit to ‘1’ enables its corresponding reset signal to trigger a reset; setting it to ‘0’ disables the reset signal, thereby masking the reset event.

This reset does not support recording occurrences of reset events.

4.6 Power Supply

The device requires a 2.7 V to 5.5 V operating voltage supply (VDD).

Table 20. Power Supply of VDD

Parameter	Symbol	Min.	Max.	Unit
Supply voltage	VDD	2.7	5.5	V
	AVDD	2.7	5.5	V

During a power-up process, a reset plays an important role and affects the entire process of system booting. A34G43x has two power-related reset options as described below:

- POR_RST (Power-On Reset) that controls the voltage less than 1.8 V.
- LVD_RST (Low-Voltage Detect Reset) that controls the voltage less than 2.11 V (default).

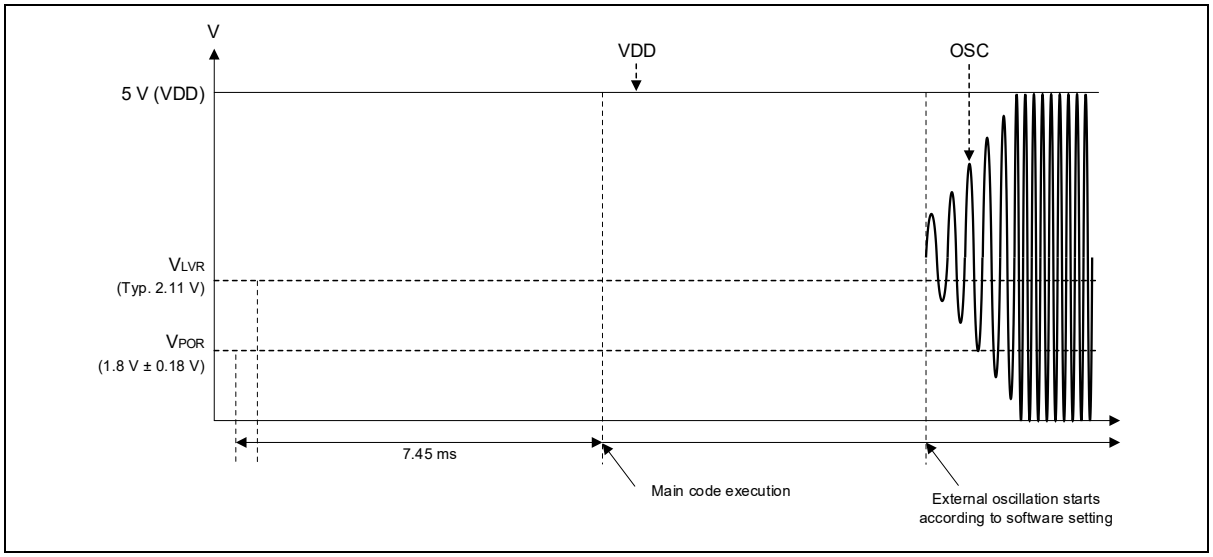
To prevent this abnormal code read operation, the LVD_RST operates to generate the SYSRESETn internal signal and enter the microcontroller into reset mode to prevent abnormal operation.

An LVR reset event is triggered when the operating voltage drops below the selected LVR level during the microcontroller's operation. Users can set the microcontroller to perform an LVD reset for LVR level setting. LVR is a warm reset. Refer to the description on section 4.5.4 for details.

Figure 26 shows a timing diagram for the operation sequence after the POR and LVR. As power is applied, it shows the waveform of main code execution and oscillator operation.

The main code is executed after the reset stabilization time has elapsed. In the case of A34G43x, note that external oscillation is set to operate by software setting.

Figure 26. POR and LVR Timing Diagram

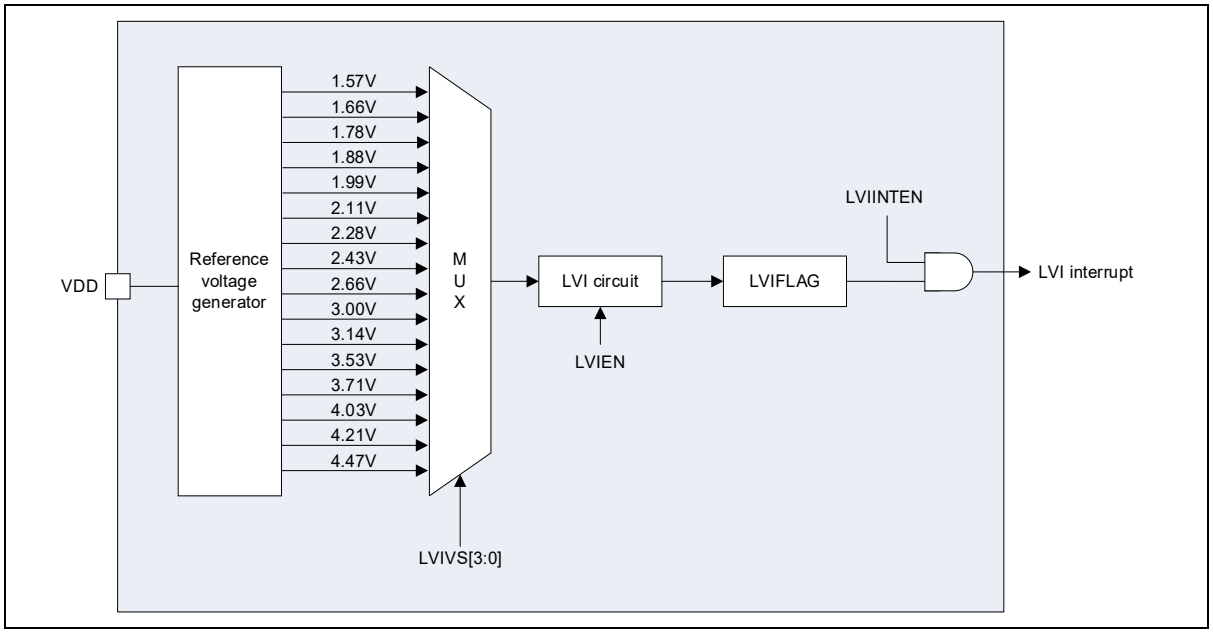


4.6.1 LVI Block Diagram

Using the LVI interrupt function, the LVI interrupt can be generated when the external VDD voltage reaches the voltage preset by users.

Figure 27 shows a block diagram of the LVI. For information on the LVI signal operation, refer to SCU_LVICR.

Figure 27. LVI Block Diagram

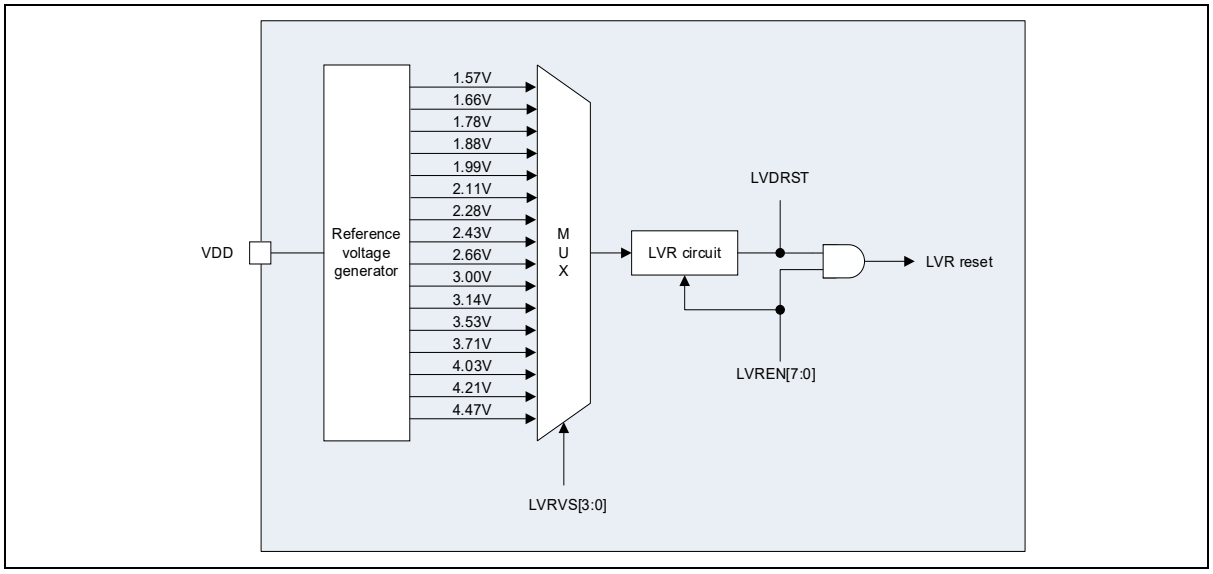


4.6.2 LVR Block Diagram

Using the LVR function, the microcontroller can be reset when the external VDD voltage reaches the voltage preset by users.

Figure 28 shows a block diagram of the LVR. For information on the LVR signal operation refer to SCU_LVRCR.

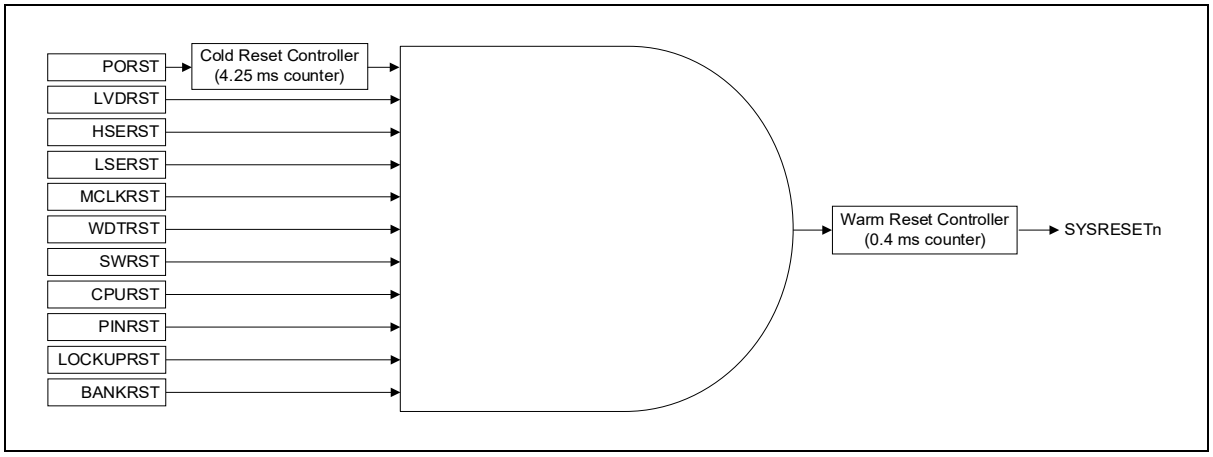
Figure 28. LVR Block Diagram



4.6.3 POR, LVR

For the case where the VDD input level is higher than the POR and lower than the minimum voltage (1.35 V) required for the flash operation, the LVDRST (Min. 1.57 V) is set to improve stability and prevent operation errors such as code read. For more information, refer to section 4.5.

Figure 29. POR and LVR Block Diagram



4.7 Operation Modes

4.7.1 Overview

The microcontroller has several functions for reducing power consumption, such as setting clock dividers, stopping modules, selecting power control mode in RUN mode, and transitioning to low power modes (SLEEP mode, DEEP-SLEEP mode)

The A34G43x operates in three modes below:

- RUN mode
- SLEEP mode
- DEEP-SLEEP (STOP 1, 2) mode

4.7.2 Transition of Operation Mode

INIT mode is the initial state of the chip when a reset is asserted.

In RUN mode, the CPU shows the maximum performance with high-speed clock system.

SLEEP mode and DEEP-SLEEP mode can be used as low power consumption mode. In the low power consumption mode, power is effectively managed to reduce power consumption by halting processor core and unused peripherals.

Figure 30 describes transition between the operation modes.

Figure 30. Transition between Operation Modes

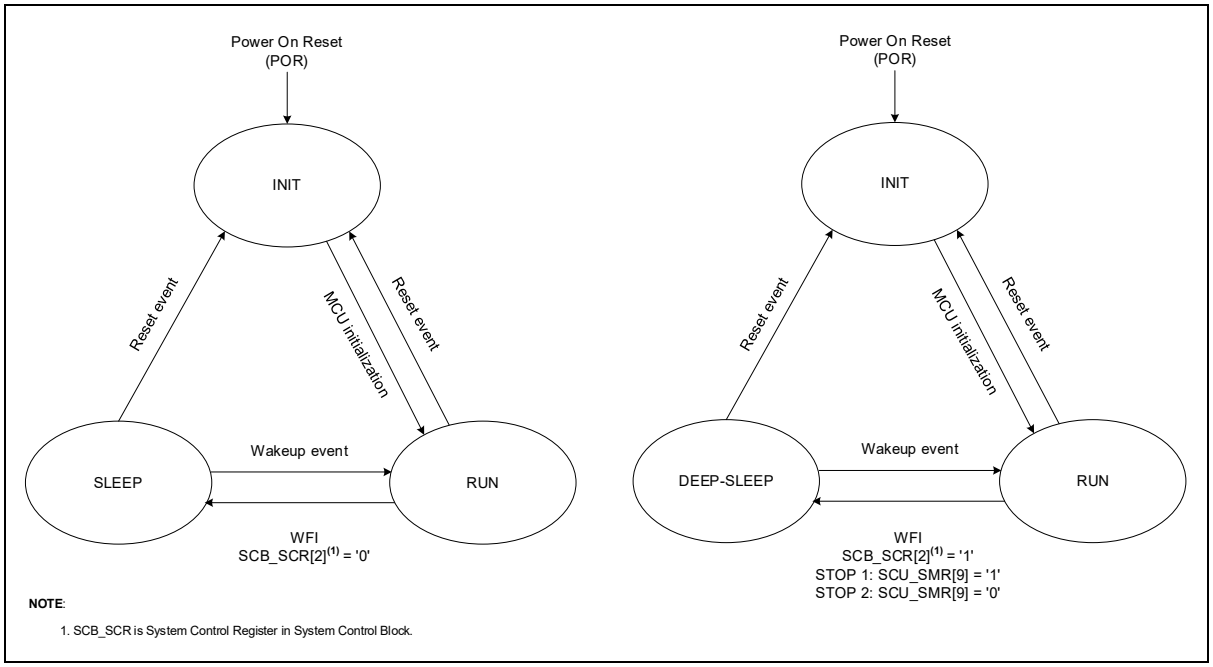


Table 21. Operation Mode

Mode	Conditions	After Wake-Up Event	After Reset Event
RUN	POWER ON	N/A	INIT
SLEEP	WFI (wait for interrupt): SCB_SCR[2](1) = 0	RUN	INIT
DEEP-SLEEP (STOP 1)	WFI (wait for interrupt): SCB_SCR[2](1) = 1 SCU_SMR[9] = 1	RUN	INIT
DEEP-SLEEP (STOP 2)	WFI (wait for interrupt): SCB_SCR[2](1) = 1 SCU_SMR[9] = 0	RUN	INIT

NOTE:

1. SCB_SCR is the System Control Register in the System Control Block.

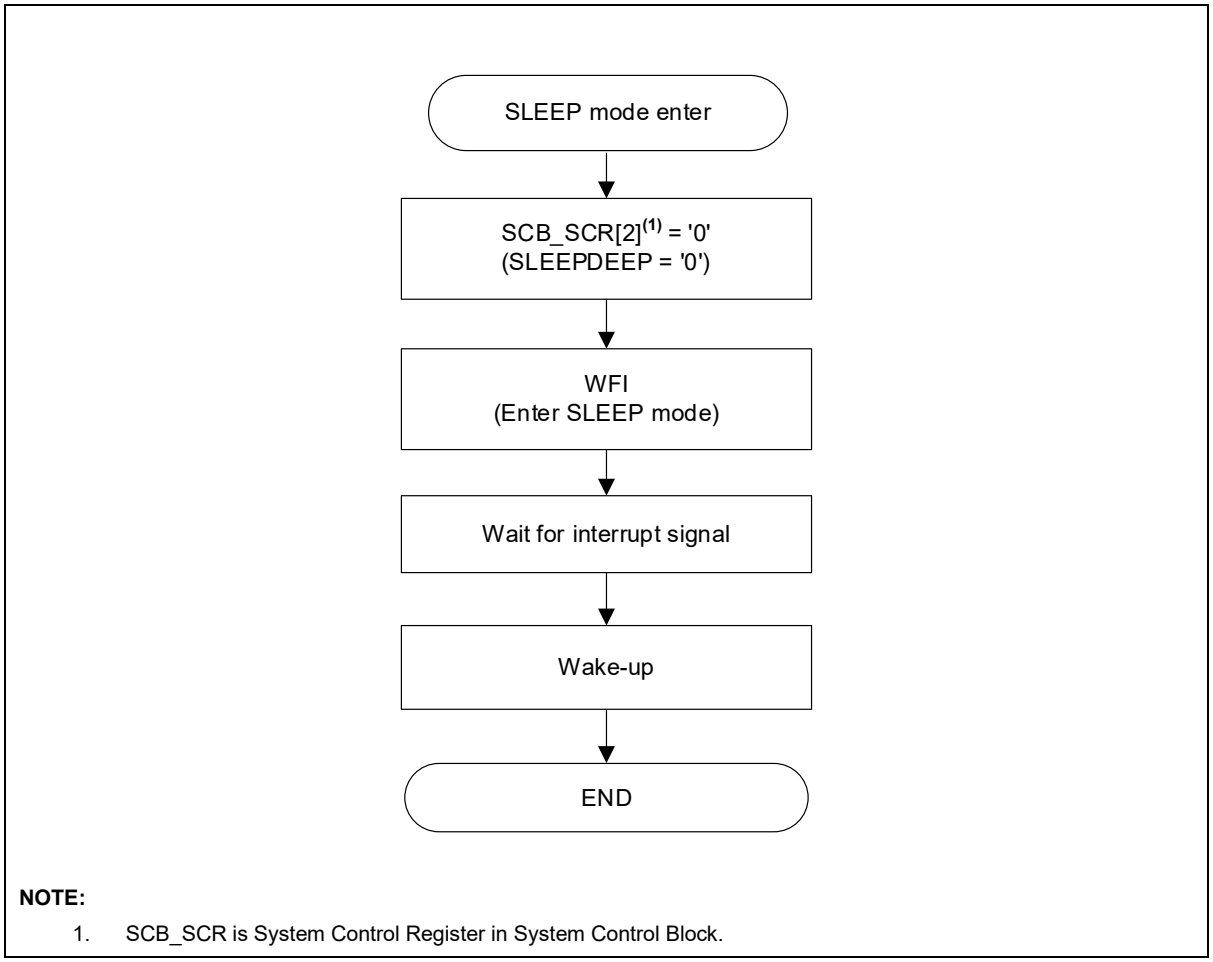
4.7.3 RUN Mode

In RUN mode, the CPU and peripheral hardware operate with a high-speed clock. If the INIT state occurs after a reset, the system enters RUN mode.

4.7.4 SLEEP Mode

Once the microcontroller enters SLEEP mode, the CPU becomes inactive. By setting the SCU_PER and SCU_PCER registers, users can determine which peripherals are to be inactive in SLEEP mode.

Figure 31. SLEEP Mode Operation Sequence



NOTE:

1. SCB_SCR is System Control Register in System Control Block.

4.7.5 DEEP-SLEEP (STOP) Mode

When the core goes under stop state using WFI instruction, the chip enters in DEEP-SLEEP mode. DEEP-SLEEP mode is divided into DEEP-SLEEP (STOP 1) and DEEP-SLEEP (STOP 2) modes.

To enter STOP 1 mode, first set SCU_SMR[9] = 1 and use the WFI command. In STOP 1 mode, all peripherals except WDT and FRT of internal VDC domain are in power off state.

To enter STOP 2 mode, first set SCU_SMR[9] = 0 and use the WFI command. In STOP 2 mode, all peripherals of internal VDC domain are stopped. To wake up in STOP 2 mode, an interrupt request must be generated and can be selected in the SCU_WUER register. Stabilization time is 4 ms after wakeup event occurs.

When entering DEEP-SLEEP mode, LSI (500 kHz) clock is automatically selected as the MCLK, HSE is selected as the PLL input clock, and all clock sources are automatically disabled. These are maintained after wake-up. If the SCU_SMR register is used, a function that automatically disables system clocks such as HSE, PLL, HSI, and LSI may not be used[1].

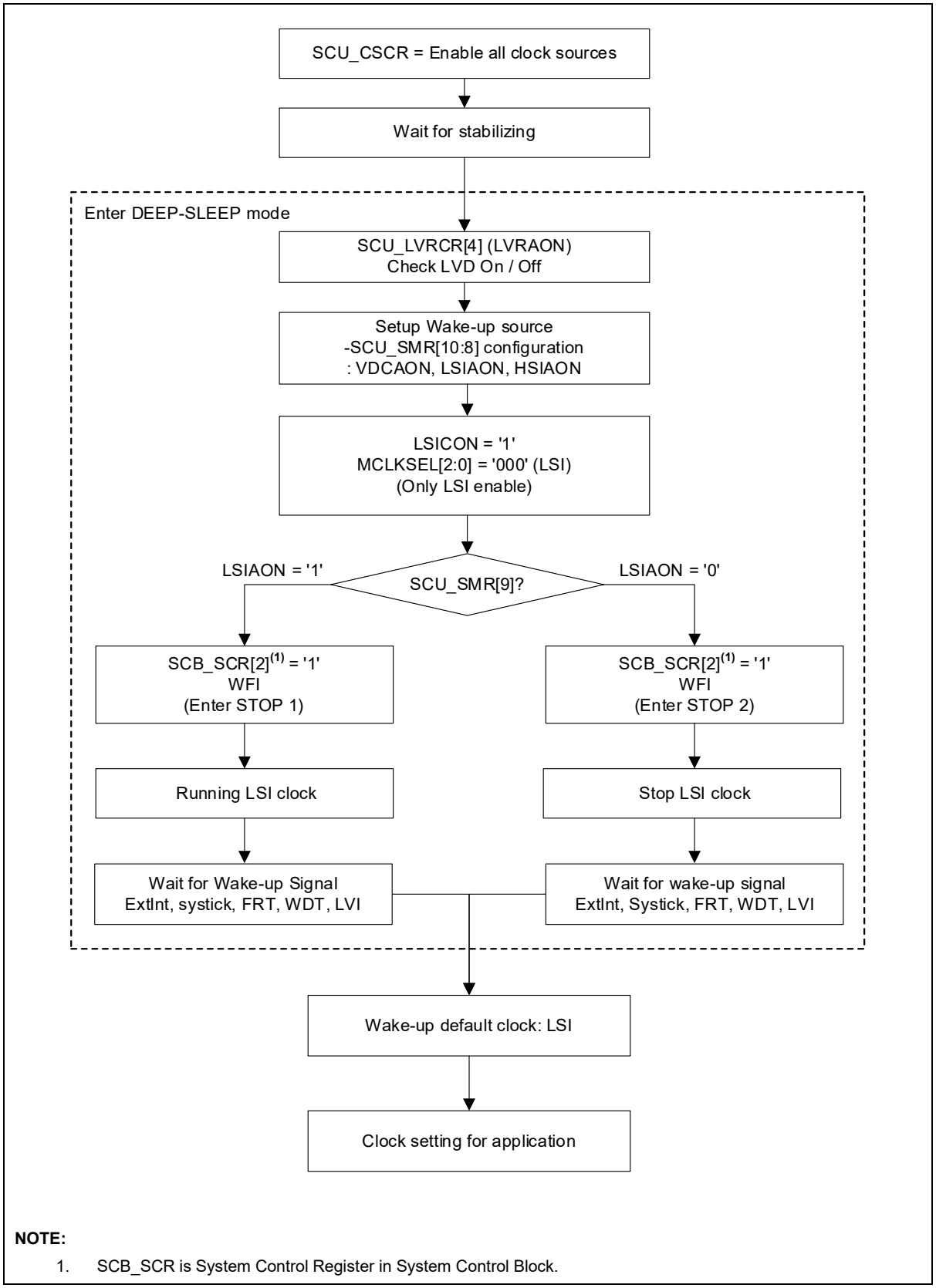
Table 22. DEEP-SLEEP Mode Configuration

Mode	Type	Conditions	Wakeup Source
DEEP-SLEEP	STOP 1	WFI SCB_SCR[2][2] = '1' (SLEEPDEEP = '1') SCU_SMR[9] = '1' (LSIAON = '1')	Source included in WUER
	STOP 2	WFI SCB_SCR[2][2] = '1' (SLEEPDEEP = '1') SCU_SMR[9] = '0' (LSIAON = '0')	Source included in WUER

NOTES:

1. When the WDT or FRT clock source is set to the PLL and the PLLAON bit in the SCU_SMR register enabled, the A34G43x cannot wake-up if it enters DEEP-SLEEP mode. In this case, the A34G43x must use SLEEP mode instead of DEEP-SLEEP mode.
2. SCB_SCR is the System Control Register in the System Control Block.

Figure 32. DEEP-SLEEP Mode Operation Sequence



NOTE:

1. SCB_SCR is System Control Register in System Control Block.

4.7.6 Operation Mode Summary

Table 23 and Table 24 are configurable clocks according to each operating mode.

Table 23. Configurable Clock in Each RUN/SLEEP Mode

Mode	CPU	Peripheral	VDC	Peripheral Clock Source				
				PLL	LSI	HSI	HSE	LSE
RUN	Active	User-defined	ON	User-defined	User-defined	User-defined	User-defined	User-defined
SLEEP	Inactive	User-defined	ON	User-defined	User-defined	User-defined	User-defined	User-defined

Table 24. Configurable Clocks in Each DEEP-SLEEP (STOP) Mode

Mode	CPU	Peripheral	SCU_SMR					
			VDCAON	PLLAON	LSIAON	HSIAON	HSEAON	LSEAON
RUN	Active	User-defined	Don't care	Don't care	Don't care	Don't care	Don't care	Don't care
SLEEP	Inactive	User-defined	Don't care	Don't care	Don't care	Don't care	Don't care	Don't care
DEEP-SLEEP (STOP 1)	Inactive	Only WDT, FRT	OFF	OFF	ON	OFF	OFF	OFF
DEEP-SLEEP (STOP 2)	Inactive	OFF	OFF	OFF	OFF	OFF	OFF	OFF

4.8 SCU Registers

The base address and register map of CHIPCONFIG are described in the following:

Table 25. Base Address of CHIPCONFIG

Name	Base Address
CHIPCONFIG	0x4000_F000

Table 26. Register Map of CHIPCONFIG

Name	Offset	Type	Description	Reset Value	Reference
CHIPCONFIG_VENDORID	0x0000	RO(1)	Vendor ID register	0x4142_4F56	4.8.1
CHIPCONFIG_CHIPID	0x0004	RO(1)	Chip ID register	0x4734_44Ax	4.8.2
CHIPCONFIG_REVNR	0x0008	RO(1)	Revision Number register	0x0000_00XX	4.8.3

NOTE:

1. 'RO' means 'Read-Only'.

Base address and register map of CHIPCONFIG2 are described in the following:

Table 27. Base Address of CHIPCONFIG2

Name	Base Address
CHIPCONFIG2	0x4000_0000

Table 28. Register Map of CHIPCONFIG2

Name	Offset	Type	Description	Reset Value	Reference
CHIPCONFIG2_CHIPID2	0x00FC	RO(1)	Chip ID 2 register	0x4734_43xx	4.8.4

NOTE:

1. 'RO' means 'Read-Only'.

The base address and register map of the SCU are described in the following:

Table 29. Base Address of SCU

Name	Base Address
SCU	0x4000_0000

Table 30. Register Map of SCU

Name	Offset	Type	Description	Reset value	Reference
SCU_SMR	0x0004	RW	System Mode Register	0x0000_0030	4.8.5
SCU_SRCR	0x0008	RW	System Reset Control Register	0x0000_0000	4.8.6
SCU_WUER	0x0010	RW	Wake-up Source Enable Register	0x0000_0000	4.8.7
SCU_WUSR	0x0014	RO	Wake-up Source Status Register	0x0000_0000	4.8.8
SCU_RSER	0x0018	RW	Reset Source Enable Register	0x0000_00D1	4.8.9
SCU_RSSR	0x001C	RWC1	Reset Source Status Register	0x0000_0101	4.8.10
SCU_PRER1	0x0020	RW	Peripheral Reset Enable Register 1	0x03FF_7FDF	4.8.11
SCU_PRER2	0x0024	RW	Peripheral Reset Enable Register 2	0xFDF3_FFFF	4.8.12
SCU_PER1	0x0028	RW	Peripheral Enable Register 1	0x0000_000F	4.8.13
SCU_PER2	0x002C	RW	Peripheral Enable Register 2	0x0000_0101	4.8.14
SCU_PCER1	0x0030	RW	Peripheral Clock Enable Register 1	0x0000_000F	4.8.15
SCU_PCER2	0x0034	RW	Peripheral Clock Enable Register 2	0x0000_0101	4.8.16
SCU_CSCR	0x0040	RW	Clock Source Control Register	0x0000_0028	4.8.17
SCU_SCCR	0x0044	RW	System Clock Control Register	0x0200_0002	4.8.18
SCU_CMR	0x0048	RW	Clock Monitor Register	0x0000_0000	4.8.19
SCU_COR	0x0050	RW	Clock Output Register	0x0000_000F	4.8.20
SCU_NMICR	0x0054	RW	Non-Maskable Interrupt Control Register	0x0000_0000	4.8.21
SCU_NMISR	0x0058	RWC1	Non-Maskable Interrupt Status Register	0x0000_0000	4.8.22
SCU_PLLCON	0x0060	RW	PLL Control Register	0x0000_0000	4.8.23
SCU_VDCCON	0x0064	RW	VDC Control Register	0x0000_007F	4.8.24
SCU_LVICR	0x0068	RW	LVI Control Register	0x0000_0000	4.8.25
SCU_LVISR	0x006C	RC	LVI Status Register	0x0000_0000	4.8.26
SCU_LVRCR	0x0070	RW	LVR Control Register	0x0000_0005	4.8.27
SCU_EOSCR	0x0080	RW	External Oscillator Control Register	0x0201_0201	4.8.28

Table 30. Register Map of SCU (continued)

Name	Offset	Type	Description	Reset value	Reference
SCU_MCCR1	0x0090	RW	MISC Clock Control Register 1	0x0001_0000	4.8.29
SCU_MCCR3	0x0098	RW	MISC Clock Control Register 3	0x0000_0000	4.8.30
SCU_MCCR4	0x009C	RW	MISC Clock Control Register 4	0x0000_0000	4.8.31
SCU_MCCR5	0x00A0	RW	MISC Clock Control Register 5	0x0000_0000	4.8.32
SCU_MCCR6	0x00A4	RW	MISC Clock Control Register 6	0x0000_0000	4.8.33
SCU_MCCR7	0x00A8	RW	MISC Clock Control Register 7	0x0000_0000	4.8.34
SCU_MCCR8	0x00AC	RW	MISC Clock Control Register 8	0x0000_0000	4.8.35
SCU_SYSTEN	0x00F0	WO	System Access Enable	0x0000_0000	4.8.36
SCU_SPSCR	0x0100	RW	SRAM Power Sequence Control Register	0x0000_0000	4.8.37
SCU_SPSSR	0x0104	RW	SRAM Power Sequence Status Register	0x0000_0000	4.8.38
SCU_SPSDR	0x0108	RW	SRAM Power Sequence Delay Register	0x0000_0FFF	4.8.39
SCU_PRER3	0x0120	RW	Peripheral Reset Enable Register 3	0x30FF_1F01	4.8.40
SCU_PER3	0x0128	RW	Peripheral Enable Register 3	0x0000_0000	4.8.41
SCU_PCER3	0x0130	RW	Peripheral Clock Enable Register 3	0x0000_0000	4.8.42

4.8.1 CHIPCONFIG_VENDORID: Vendor ID Register

The CHIPCONFIG_VENDORID register shows the vendor's identification information. This is a 32-bit read-only register.

CHIPCONFIG_VENDORID=0x4000_F000

31 30 29 28 27 26 25 24	23 22 21 20 19 18 17 16	15 14 13 12 11 10 9 8	7 6 5 4 3 2 1 0
VENDID[31:0]			
0x4142_4F56			
RO			

31	VENDID[31:0]	Vendor identification bits.	
		0x4142_4F56	

4.8.2 CHIPCONFIG_CHIPID: Chip ID Register

The CHIPCONFIG_CHIPID register shows the chip's identification information. This is a 32-bit read-only register.

CHIPCONFIG_CHIPID=0x4000_F004

31 30 29 28 27 26 25 24	23 22 21 20 19 18 17 16	15 14 13 12 11 10 9 8	7 6 5 4 3 2 1 0
CHIPID[31:0]			
0x4734_44Ax			
RO			

31 0	CHIPID[31:0]	Chip ID bit.	
		0x4734_44A0	2 MB flash memory / 256 KB RAM
		0x4734_44A1	1 MB flash memory / 256 KB RAM

4.8.3 CHIPCONFIG_REVNR: Revision Number Register

The CHIPCONFIG_REVNR shows the revision number information of a chip. This register is a 32-bit read-only register.

CHIPCONFIG_REVNR=0x4000_F008

31 30 29 28 27 26 25 24	23 22 21 20 19 18 17 16	15 14 13 12 11 10 9 8	7 6 5 4 3 2 1 0
Reserved			REVNO[7:0]
0x000000			0xXX
.			RO

7 0	REVNO[7:0]	Chip revision number These bits are modified by a manufacturer.

4.8.4 CHIPCONFIG2_CHIPID2: CHIP ID 2 Register

The CHIPCONFIG2_CHIPID2 register shows the chip's identification information. This is a 32-bit read-only register.

CHIPCONFIG2_CHIPID2=0x4000_00FC

31 30 29 28 27 26 25 24	23 22 21 20 19 18 17 16	15 14 13 12 11 10 9 8	7 6 5 4 3 2 1 0
CHIPID2[31:0]			
0x4734_43xx			
RO			

31 0	CHIPID2[31:0]	Chip ID 2 bit.	
		0x4734_43A0	2 MB flash memory / 256 KB RAM
		0x4734_4300	1 MB flash memory / 256 KB RAM

4.8.5 SCU_SMR: System Mode Register

The SCU_SMR contains bits controllable in DEEP-SLEEP[1] mode and informs of which operating mode the microcontroller had been in before it returned to initial mode. Once a reset event occurs, the previous operating mode is recorded in this register. The register is 32 bits wide.

SCU_SMR=0x4000_0004

31 30 29 28 27 26 25 24	23 22 21 20 19 18 17 16	15 14	13	12	11	10	9	8	7 6	5 4	3 2 1 0
Reserved			LSEAON	HSEAON	PLLAON	HSIAON	LSIAON	VDCAON	Reserved	PREVMODE[1:0]	Reserved
-			0	0	0	0	0	0	-	11	-
-			RW	RW	RW	RW	RW	RW	-	RO	-

13	LSEAON[1]	LSE enablement in mode	
		0	Automatically disables the LSE in DEEP-SLEEP mode.
		1	Leaves the LSE enabled in DEEP-SLEEP mode.
12	HSEAON[1]	HSE enablement in DEEP-SLEEP mode	
		0	Automatically disables the HSE in DEEP-SLEEP mode.
		1	Leaves the HSE enabled in DEEP-SLEEP mode.
11	PLLAON[2][3]	PLL enablement in DEEP-SLEEP mode	
		0	Automatically disables the PLL in DEEP-SLEEP mode.
		1	Leaves the PLL enabled in DEEP-SLEEP mode.
10	HSIAON[1]	HSI enablement in DEEP-SLEEP mode	
		0	Automatically disables the HSI in DEEP-SLEEP mode.
		1	Leaves the HSI enabled in DEEP-SLEEP mode.
9	LSIAON[1]	LSI enablement in DEEP-SLEEP mode	
		0	Automatically disables the LSI in DEEP-SLEEP mode.
		1	Leaves the LSI enabled in DEEP-SLEEP mode.
8	VDCAON	VDC enablement in DEEP-SLEEP mode	
		0	Automatically disables the VDC in DEEP-SLEEP mode.
		1	Leaves the VDC enabled in DEEP-SLEEP mode.
5 4	PREVMODE[1:0]	The operating mode when the last reset event occurred.	
		00	The microcontroller was in RUN mode.
		01	The microcontroller was in SLEEP mode.
		10	The microcontroller was in DEEP-SLEEP mode.
		11	The microcontroller was in INITIAL mode.

NOTES:

1. Even if you set the SCU_SMR register for a clock to stay enabled during DEEP-SLEEP mode, the clock will not run if it has been set disabled in the SCU_CSCR register.
2. If the PLLAON bit is set to '1', the VDC stays active regardless of the setting of the VDCAON bit.
3. When the WDT or FRT clock source is set to the PLL and the PLLAON bit in the SCU_SMR register enabled, the A34G43x cannot wakeup if it enters DEEP-SLEEP mode. In this case, the A34G43x must use SLEEP mode instead of DEEP-SLEEP mode.

4.8.6 SCU_SRCR: System Reset Control Register

It is possible to reset the microcontroller by setting the SWRST bit. This system reset control register is a 32-bit register.

SCU_SRCR=0x4000_0008

31 30 29 28 27 26 25 24	23 22 21 20 19 18 17 16	15 14 13 12 11 10 9 8	7 6 5	4	3 2 1	0
Reserved				STBYO	Reserved	SWRST
-				0	-	0
-				RW	-	WO

4	STBYO	Inversion selection for the STBYO pin's output	
		0	Low active when the chip is in power down.
		1	High active when the chip is in power down.
0	SWRST[1]	Internal soft reset activation bit (Check SWRST in SCU_RSER for reset)	
		0	Normal operation
		1	Internal soft resets can occur (The bit becomes automatically cleared).

NOTE:

1. Users must use SWRST bit in the SCU_SRCR register instead of NVIC_SystemReset() or the SCB_AIRCR register setting for microcontroller reset.

4.8.7 SCU_WUER: Wake-up Source Enable Register

The SCU_WUER register is used when the microcontroller is in DEEP-SLEEP mode. To enable a signal used as a wake-up source, users must set the corresponding bit to 1 and the remaining bits not used as a wake-up source must be set to 0. This register is 32 bits wide.

SCU_WUER=-0x4000_0010

31 30 29 28 27 26 25 24	23 22 21 20 19 18 17 16	15	14	13	12	11	10	9	8	7 6 5 4	3	2	1	0
Reserved			GPIOGWUE	GPIOFWUE	GPIOEWUE	GPIODWUE	GPIOCWUE	GPIOBWUE	GPIOAWUE	Reserved	FRT1WUE	FRT0WUE	WDTWUE	LVIWUE
-			0	0	0	0	0	0	0	-	0	0	0	0
-			RW	RW	RW	RW	RW	RW	RW	-	RW	RW	RW	RW

14	GPIOGWUE	Use the GPIOG port event as a wake-up source
	0	Does not use the event as a wake-up source.
	1	Uses the event as a wake-up source.
13	GPIOFWUE	Use the GPIOF port event as a wake-up source
	0	Does not use the event as a wake-up source.
	1	Uses the event as a wake-up source.
12	GPIOEWUE	Use the GPIOE port event as a wake-up source
	0	Does not use the event as a wake-up source.
	1	Uses the event as a wake-up source.
11	GPIODWUE	Use the GPIOD port event as a wake-up source
	0	Does not use the event as a wake-up source.
	1	Uses the event as a wake-up source.
10	GPIOCWUE	Use the GPIOC port event as a wake-up source
	0	Does not use the event as a wake-up source.
	1	Uses the event as a wake-up source.
9	GPIOBWUE	Use the GPIOB port event as a wake-up source
	0	Does not use the event as a wake-up source.
	1	Uses the event as a wake-up source.
8	GPIOAWUE	Use the GPIOA port event as a wake-up source
	0	Does not use the event as a wake-up source.
	1	Uses the event as a wake-up source.
3	FRT1WUE	Use the FRT1 event as a wake-up source
	0	Does not use the event as a wake-up source.
	1	Uses the event as a wake-up source.
2	FRT0WUE	Use the FRT0 event as a wake-up source
	0	Does not use the event as a wake-up source.
	1	Uses the event as a wake-up source.
1	WDTWUE	Use the WDT event as a wake-up source
	0	Does not use the event as a wake-up source.
	1	Uses the event as a wake-up source.
0	LVIWUE	Use the LVI event as a wake-up source
	0	Does not use the event as a wake-up source.
	1	Uses the event as a wake-up source.

4.8.8 SCU_WUSR: Wake-up Source Status Register

When the microcontroller is woken up by a wake-up source, the corresponding bit in the SCU_WUSR register is flagged. A bit set to '1' means that its corresponding wake-up event has occurred. It is a read-only register. When an event source is cleared, the corresponding bit in this register is also cleared.

SCU_WUSR=-0x4000_0014

31 30 29 28 27 26 25 24	23 22 21 20 19 18 17 16	15	14	13	12	11	10	9	8	7 6 5 4	3	2	1	0
Reserved			GPIOGWU	GPIOFWU	GPIOEWU	GPIODWU	GPIOCWU	GPIOBWU	GPIOAWU	Reserved	FRT1WU	FRT0WU	WDTWU	LVIWU
-			0	0	0	0	0	0	0	-	0	0	0	0
-			R0	R0	R0	R0	R0	R0	R0	-	R0	R0	R0	R0

14	GPIOGWU	GPIOG port event Wake-up status
	0	Wake-up event does not occur.
	1	Wake-up event occurs.
13	GPIOFWU	GPIOF port event Wake-up status
	0	Wake-up event does not occur.
	1	Wake-up event occurs.
12	GPIOEWU	GPIOE port event Wake-up status
	0	Wake-up event does not occur.
	1	Wake-up event occurs.
11	GPIODWU	GPIOD port event Wake-up status
	0	Wake-up event does not occur.
	1	Wake-up event occurs.
10	GPIOCWU	GPIOC port event Wake-up status
	0	Wake-up event does not occur.
	1	Wake-up event occurs.
9	GPIOBWU	GPIOB port event Wake-up status
	0	Wake-up event does not occur.
	1	Wake-up event occurs.
8	GPIOAWU	GPIOA port event Wake-up status
	0	Wake-up event does not occur.
	1	Wake-up event occurs.
3	FRT1WU	FRT1 event Wake-up status
	0	Wake-up event does not occur.
	1	Wake-up event occurs.
2	FRT0WU	FRT0 event Wake-up status
	0	Wake-up event does not occur.
	1	Wake-up event occurs.
1	WDTWU	Watchdog Timer (WDT) event Wake-up status
	0	Wake-up event does not occur.
	1	Wake-up event occurs.
0	LVIWU	LVI event Wake-up status
	0	Wake-up event does not occur.
	1	Wake-up event occurs.

4.8.9 SCU_RSER: Reset Source Enable Register

The SCU_RSER allows users to configure the reset input signals that trigger a reset event. Setting a bit to '1' enables the corresponding reset input signal to trigger a reset; setting it to '0' disables the reset input signal, thereby masking the reset event.

SCU_RSER=0x4000_0018

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
BANKRST	Reserved																					LOCKUPRST	Reserved	PINRST	CPURST	SWRST	WDTRST	MCKFRST	LSEFRST	HSEFRST	LVDRST
0	-																					0	-	1	1	0	1	0	0	0	1
RW	-																					RW	-	RW	RW	RW	RW	RW	RW	RW	RW

31	BANKRST	Enable the Bank reset signal
	0	Disables the signal to trigger a reset event.
	1	Enables the signal to trigger a reset event.
9	LOCKUPRST	Enable the CPU lock-up reset signal
	0	Disables the signal to trigger a reset event.
	1	Enables the signal to trigger a reset event.
7	PINRST	Enable the external-pin reset signal
	0	Disables the signal to trigger a reset event.
	1	Enables the signal to trigger a reset event.
6	CPURST	Enable the CPU request reset signal
	0	Disables the signal to trigger a reset event.
	1	Enables the signal to trigger a reset event.
5	SWRST	Enable the software reset signal
	0	Disables the signal to trigger a reset event.
	1	Enables the signal to trigger a reset event.
4	WDTRST	Enable the WDT reset signal
	0	Disables the signal to trigger a reset event.
	1	Enables the signal to trigger a reset event.
3	MCKFRST	Enable the MCLK error reset signal
	0	Disables the signal to trigger a reset event.
	1	Enables the signal to trigger a reset event.
2	LSEFRST	Enable the LSE error reset signal
	0	Disables the signal to trigger a reset event.
	1	Enables the signal to trigger a reset event.
1	HSEFRST	Enable the HSE error reset signal
	0	Disables the signal to trigger a reset event.
	1	Enables the signal to trigger a reset event.
0	LVDRST	Enable the LVD reset signal
	0	Disables the signal to trigger a reset event.
	1	Enables the signal to trigger a reset event.

4.8.10 SCU_RSSR: Reset Source Status Register

The SCU_RSSR register records the occurrences of the reset events. A bit read as '1' means that its corresponding reset source has triggered a reset event. To clear the flag and the reset status of a reset source, write a '1' to the corresponding flag bit in this register. This register is 32 bits wide.

SCU_RSSR=0x4000_001C

31 30 29 28 27 26 25 24	23 22 21 20 19 18 17 16	15 14 13 12 11 10	9	8	7	6	5	4	3	2	1	0
Reserved			LOCKUPRST	PORST	PINRST	CPURST	SWRST	WDTRST	MCLKFRST	LSERST	HSERST	LVDRST
-			0	1	0	0	0	0	0	0	0	1
-			RWC1	RWC1	RWC1	RWC1	RWC1	RWC1	RWC1	RWC1	RWC1	RWC1

9	LOCKUPRST	Indicates the CPU lock-up reset has occurred	
		0	Read: The reset has not occurred. Write: N/A
		1	Read: The reset has occurred. Write: Clears the flag.
8	PORST	Indicates the POR reset has occurred	
		0	Read: The reset has not occurred. Write: N/A
		1	Read: The reset has occurred. Write: Clears the flag.
7	PINRST	Indicates the external pin reset has occurred	
		0	Read: The reset has not occurred. Write: N/A
		1	Read: The reset has occurred. Write: Clears the flag.
6	CPURST	Indicates the CPU core reset has occurred	
		0	Read: The reset has not occurred. Write: N/A
		1	Read: The reset has occurred. Write: Clears the flag.
5	SWRST	Indicates the software reset has occurred	
		0	Read: The reset has not occurred. Write: N/A
		1	Read: The reset has occurred. Write: Clears the flag.
4	WDTRST	Indicates the WDT reset has occurred	
		0	Read: The reset has not occurred. Write: N/A
		1	Read: The reset has occurred. Write: Clears the flag.
3	MCLKFRST	Indicates the MCLK error reset has occurred	
		0	Read: The reset has not occurred. Write: N/A
		1	Read: The reset has occurred. Write: Clears the flag.
2	LSEFRST	Indicates the LSE error reset has occurred	
		0	Read: The reset has not occurred. Write: N/A
		1	Read: The reset has occurred. Write: Clears the flag.

1	HSEFRST	Indicates the HSE error reset has occurred	
		0	Read: The reset has not occurred. Write: N/A
		1	Read: The reset has occurred. Write: Clears the flag.
0	LVDRST	Indicates the LVD reset has occurred	
		0	Read: The reset has not occurred. Write: N/A
		1	Read: The reset has occurred. Write: Clears the flag.

4.8.11 SCU_PRER1: Peripheral Reset Enable Register 1

The SCU_PRERx registers determine whether to allow each peripheral to be initialized when a warm reset event occurs. They consist of SCU_PRER1, SCU_PRER2 and SCU_PRER3. Once a reset event occurs, the peripherals whose corresponding bits are set to '1' are initialized and those set to '0' remain uninitialized. However, the POR reset initializes all peripherals.

SCU_PRER1=0x4000_0020

31 30 29 28 27 26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved	TIMER9	TIMER8	TIMER7	TIMER6	TIMER5	TIMER4	TIMER3	TIMER2	TIMER1	TIMER0	Reserved	GPIOG	GPIOF	GPIOE	GPIOD	GPIOC	GPIOB	GPIOA	FRT1	FRT0	Reserved	DMA	DFMC	WDT	CFMC	SCU
-	1	1	1	1	1	1	1	1	1	1	-	1	1	1	1	1	1	1	1	1	-	1	1	1	1	1
-	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	-	RW	RW	RW	RW	RW	RW	RW	RW	RW	-	RW	RW	RW	RW	RW

25	TIMER9	TIMER9 reset mask
		0 Masked
		1 Not masked
24	TIMER8	TIMER8 reset mask
		0 Masked
		1 Not masked
23	TIMER7	TIMER7 reset mask
		0 Masked
		1 Not masked
22	TIMER6	TIMER6 reset mask
		0 Masked
		1 Not masked
21	TIMER5	TIMER5 reset mask
		0 Masked
		1 Not masked
20	TIMER4	TIMER4 reset mask
		0 Masked
		1 Not masked
19	TIMER3	TIMER3 reset mask
		0 Masked
		1 Not masked
18	TIMER2	TIMER2 reset mask
		0 Masked
		1 Not masked
17	TIMER1	TIMER1 reset mask
		0 Masked
		1 Not masked
16	TIMER0	TIMER0 reset mask
		0 Masked
		1 Not masked
14	GPIOG	GPIOG reset mask
		0 Masked
		1 Not masked
13	GPIOF	GPIOF reset mask
		0 Masked
		1 Not masked

12	GPIOE	GPIOE reset mask
		0 Masked
		1 Not masked
11	GPIOD	GPIOD reset mask
		0 Masked
		1 Not masked
10	GPIOC	GPIOC reset mask
		0 Masked
		1 Not masked
9	GPIOB	GPIOB reset mask
		0 Masked
		1 Not masked
8	GPIOA	GPIOA reset mask
		0 Masked
		1 Not masked
7	FRT1	FRT1 reset mask
		0 Masked
		1 Not masked
6	FRT0	FRT0 reset mask
		0 Masked
		1 Not masked
4	DMA	DMA reset mask
		0 Masked
		1 Not masked
3	DFMC	DFMC reset mask
		0 Masked
		1 Not masked
2	WDT	WDT reset mask
		0 Masked
		1 Not masked
1	CFMC	CFMC reset mask
		0 Masked
		1 Not masked
0	SCU	SCU reset mask
		0 Masked
		1 Not masked

4.8.12 SCU_PRER2: Peripheral Reset Enable Register 2

The SCU_PRER2 register is 32 bits wide.

SCU_PRER2=0x4000_0024

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RNG	AES	CRC	CMP	CANFD1	CANFD0	Reserved	DAC1	DAC0	ADC2	ADC1	ADC0	Reserved		MPWM1	MPWM0	UART7	UART6	UART5	UART4	UART3	UART2	UART1	UART0	I2C3	I2C2	I2C1	I2C0	SPI3	SPI2	SPI1	SPI0
1	1	1	1	1	1	-	1	1	1	1	1	-		1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1
RW	RW	RW	RW	RW	RW	-	RW	RW	RW	RW	RW	-		RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW

31	RNG	RNG reset mask
		0 Masked
		1 Not masked
30	AES	AES reset mask
		0 Masked
		1 Not masked
29	CRC	CRC reset mask
		0 Masked
		1 Not masked
28	CMP	CMP reset mask
		0 Masked
		1 Not masked
27	CANFD1	CANFD1 reset mask
		0 Masked
		1 Not masked
26	CANFD0	CANFD0 reset mask
		0 Masked
		1 Not masked
24	DAC1	DAC1 reset mask
		0 Masked
		1 Not masked
23	DAC0	DAC0 reset mask
		0 Masked
		1 Not masked
22	ADC2	ADC2 reset mask
		0 Masked
		1 Not masked
21	ADC1	ADC1 reset mask
		0 Masked
		1 Not masked
20	ADC0	ADC0 reset mask
		0 Masked
		1 Not masked
17	MPWM1	MPWM1 reset mask
		0 Masked
		1 Not masked
16	MPWM0	MPWM0 reset mask
		0 Masked
		1 Not masked
15	UART7	UART7 reset mask

		0	Masked
		1	Not masked
14	UART6	UART6 reset mask	
		0	Masked
		1	Not masked
13	UART5	UART5 reset mask	
		0	Masked
		1	Not masked
12	UART4	UART4 reset mask	
		0	Masked
		1	Not masked
11	UART3	UART3 reset mask	
		0	Masked
		1	Not masked
10	UART2	UART2 reset mask	
		0	Masked
		1	Not masked
9	UART1	UART1 reset mask	
		0	Masked
		1	Not masked
8	UART0	UART0 reset mask	
		0	Masked
		1	Not masked
7	I2C3	I2C3 reset mask	
		0	Masked
		1	Not masked
6	I2C2	I2C2 reset mask	
		0	Masked
		1	Not masked
5	I2C1	I2C1 reset mask	
		0	Masked
		1	Not masked
4	I2C0	I2C0 reset mask	
		0	Masked
		1	Not masked
3	SPI3	SPI3 reset mask	
		0	Masked
		1	Not masked
2	SPI2	SPI2 reset mask	
		0	Masked
		1	Not masked
1	SPI1	SPI1 reset mask	
		0	Masked
		1	Not masked
0	SPI0	SPI0 reset mask	
		0	Masked
		1	Not masked

4.8.13 SCU_PER1: Peripheral Enable Register 1

To enable a peripheral, users must write a '1' to its corresponding bit in SCU_PER1, SCU_PER2 or SCU_PER3. Before it is enabled, the peripheral is reset.

Because only a certain set of peripherals are enabled at the initial reset, additional action is required to have other peripherals enabled if you want to use them. By writing a '0' to the bit representing the peripheral you'll not use, you can reset it.

SCU_PER1=0x4000_0028

31 30 29 28 27 26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3 2 1 0
Reserved	TIMER9	TIMER8	TIMER7	TIMER6	TIMER5	TIMER4	TIMER3	TIMER2	TIMER1	TIMER0	Reserved	GPIOG	GPIOF	GPIOE	GPIOD	GPIOC	GPIOB	GPIOA	FRT1	FRT0	Reserved	DMA	Reserved
-	0	0	0	0	0	0	0	0	0	0	-	0	0	0	0	0	0	0	0	0	-	0	-
-	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	-	RW	RW	RW	RW	RW	RW	RW	RW	RW	-	RW	-

25	TIMER9	Enable TIMER9
		0 Disabled
		1 Enabled
24	TIMER8	Enable TIMER8
		0 Disabled
		1 Enabled
23	TIMER7	Enable TIMER7
		0 Disabled
		1 Enabled
22	TIMER6	Enable TIMER6
		0 Disabled
		1 Enabled
21	TIMER5	Enable TIMER5
		0 Disabled
		1 Enabled
20	TIMER4	Enable TIMER4
		0 Disabled
		1 Enabled
19	TIMER3	Enable TIMER3
		0 Disabled
		1 Enabled
18	TIMER2	Enable TIMER2
		0 Disabled
		1 Enabled
17	TIMER1	Enable TIMER1
		0 Disabled
		1 Enabled
16	TIMER0	Enable TIMER0
		0 Disabled
		1 Enabled
14	GPIOG	Enable GPIOG
		0 Disabled
		1 Enabled
13	GPIOF	Enable GPIOF

		0 Disabled
		1 Enabled
12	GPIOE	Enable GPIOE
		0 Disabled
		1 Enabled
11	GPIOD	Enable GPIOD
		0 Disabled
		1 Enabled
10	GPIOC	Enable GPIOC
		0 Disabled
		1 Enabled
9	GPIOB	Enable GPIOB
		0 Disabled
		1 Enabled
8	GPIOA	Enable GPIOA
		0 Disabled
		1 Enabled
7	FRT1	Enable FRT1
		0 Disabled
		1 Enabled
6	FRT0	Enable FRT0
		0 Disabled
		1 Enabled
4	DMA	Enable DMA
		0 Disabled
		1 Enabled

4.8.14 SCU_PER2: Peripheral Enable Register 2

The SCU_PER2 is 32bits wide.

SCU_PER2=0x4000_002C

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RNG	AES	CRC	CMP	CANFD1	CANFD0	Reserved	DAC1	DAC0	ADC2	ADC1	ADC0	Reserved		MPWM1	MPWM0	UART7	UART6	UART5	UART4	UART3	UART2	UART1	UART0	I2C3	I2C2	I2C1	I2C0	SPI3	SPI2	SPI1	SPI0
0	0	0	0	0	0	-	0	0	0	0	0	-		0	0	0	0	0	0	0	0	0	1	0	0	0	0	0	0	0	1
RW	RW	RW	RW	RW	RW	-	RW	RW	RW	RW	RW	-		RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW

31	RNG	Enable RNG
		0 Disabled
		1 Enabled
30	AES	Enable AES
		0 Disabled
		1 Enabled
29	CRC	Enable CRC
		0 Disabled
		1 Enabled
28	CMP	Enable CMP
		0 Disabled
		1 Enabled
27	CANFD1	Enable CANFD1
		0 Disabled
		1 Enabled
26	CANFD0	Enable CANFD0
		0 Disabled
		1 Enabled
24	DAC1	Enable DAC1
		0 Disabled
		1 Enabled
23	DAC0	Enable DAC0
		0 Disabled
		1 Enabled
22	ADC2	Enable ADC2
		0 Disabled
		1 Enabled
21	ADC1	Enable ADC1
		0 Disabled
		1 Enabled
20	ADC0	Enable ADC0
		0 Disabled
		1 Enabled
17	MPWM1	Enable MPWM1
		0 Disabled
		1 Enabled
16	MPWM0	Enable MPWM0
		0 Disabled
		1 Enabled
15	UART7	Enable UART7

		0 Disabled
		1 Enabled
14	UART6	Enable UART7
		0 Disabled
		1 Enabled
13	UART5	Enable UART5
		0 Disabled
		1 Enabled
12	UART4	Enable UART4
		0 Disabled
		1 Enabled
11	UART3	Enable UART3
		0 Disabled
		1 Enabled
10	UART2	Enable UART2
		0 Disabled
		1 Enabled
9	UART1	Enable UART1
		0 Disabled
		1 Enabled
8	UART0	Enable UART0
		0 Disabled
		1 Enabled
7	I2C3	Enable I2C3
		0 Disabled
		1 Enabled
6	I2C2	Enable I2C2
		0 Disabled
		1 Enabled
5	I2C1	Enable I2C1
		0 Disabled
		1 Enabled
4	I2C0	Enable I2C0
		0 Disabled
		1 Enabled
3	SPI3	Enable SPI3
		0 Disabled
		1 Enabled
2	SPI2	Enable SPI2
		0 Disabled
		1 Enabled
1	SPI1	Enable SPI1
		0 Disabled
		1 Enabled
0	SPI0	Enable SPI0
		0 Disabled
		1 Enabled

4.8.15 SCU_PCER1: Peripheral Clock Enable Register 1

To enable a specific clock for a peripheral, users must write '1' to its corresponding bit in the SCU_PCER1, SCU_PCER2 or SCU_PCER3 registers. A peripheral cannot reset or operate normally until it is clocked.

If a peripheral will not be used, the users must write '0' to its corresponding bit in the SCU_PCER1, SCU_PCER2 or SCU_PECR3 registers to disable its clock. When the microcontroller is in SLEEP or DEEP-SLEEP mode, the clock is fed only to active peripherals.

SCU_PCER1=0x4000_0030

31 30 29 28 27 26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3 2 1 0
Reserved	TIMER9	TIMER8	TIMER7	TIMER6	TIMER5	TIMER4	TIMER3	TIMER2	TIMER1	TIMER0	Reserved	GPIOG	GPIOF	GPIOE	GPIOD	GPIOC	GPIOB	GPIOA	FRT1	FRT0	Reserved	DMA	Reserved
-	0	0	0	0	0	0	0	0	0	0	-	0	0	0	0	0	0	0	0	0	-	0	1111
-	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	-	RW	RW	RW	RW	RW	RW	RW	RW	RW	-	RW	RO

25	TIMER9	Enable the clock to TIMER9
		0 Disabled
		1 Enabled
24	TIMER8	Enable the clock to TIMER8
		0 Disabled
		1 Enabled
23	TIMER7	Enable the clock to TIMER7
		0 Disabled
		1 Enabled
22	TIMER6	Enable the clock to TIMER6
		0 Disabled
		1 Enabled
21	TIMER5	Enable the clock to TIMER5
		0 Disabled
		1 Enabled
20	TIMER4	Enable the clock to TIMER4
		0 Disabled
		1 Enabled
19	TIMER3	Enable the clock to TIMER3
		0 Disabled
		1 Enabled
18	TIMER2	Enable the clock to TIMER2
		0 Disabled
		1 Enabled
17	TIMER1	Enable the clock to TIMER1
		0 Disabled
		1 Enabled
16	TIMER0	Enable the clock to TIMER0
		0 Disabled
		1 Enabled
14	GPIOG	Enable the clock to GPIOG
		0 Disabled

		1	Enabled
13	GPIOF	Enable the clock to GPIOF	
		0	Disabled
		1	Enabled
12	GPIOE	Enable the clock to GPIOE	
		0	Disabled
		1	Enabled
11	GPIOD	Enable the clock to GPIOD	
		0	Disabled
		1	Enabled
10	GPIOC	Enable the clock to GPIOC	
		0	Disabled
		1	Enabled
9	GPIOB	Enable the clock to GPIOB	
		0	Disabled
		1	Enabled
8	GPIOA	Enable the clock to GPIOA	
		0	Disabled
		1	Enabled
7	FRT1	Enable the clock to FRT1	
		0	Disabled
		1	Enabled
6	FRT0	Enable the clock to FRT0	
		0	Disabled
		1	Enabled
4	DMA	Enable the clock to DMA	
		0	Disabled
		1	Enabled

4.8.16 SCU_PCER2: Peripheral Clock Enable Register 2

To use peripheral unit, its clock should be activated by writing '1' to the corresponding bit in the SCU_PCER2 register.

SCU_PCER2=0x4000_0034

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RNG	AES	CRC	CMP	CANFD1	CANFD0	Reserved	DAC1	DAC0	ADC2	ADC1	ADC0	Reserved		MPWM1	MPWM0	UART7	UART6	UART5	UART4	UART3	UART2	UART1	UART0	I2C3	I2C2	I2C1	I2C0	SPI3	SPI2	SPI1	SPI0
0	0	0	0	0	0	-	0	0	0	0	0	-		0	0	0	0	0	0	0	0	0	1	0	0	0	0	0	0	0	1
RW	RW	RW	RW	RW	RW	-	RW	RW	RW	RW	RW	-		RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW

31	RNG	Enable the clock to RNG
		0 Disabled
		1 Enabled
30	AES	Enable the clock to AES
		0 Disabled
		1 Enabled
29	CRC	Enable the clock to CRC
		0 Disabled
		1 Enabled
28	CMP	Enable the clock to CMP
		0 Disabled
		1 Enabled
27	CANFD1	Enable the clock to CANFD1
		0 Disabled
		1 Enabled
26	CANFD0	Enable the clock to CANFD0
		0 Disabled
		1 Enabled
24	DAC1	Enable the clock to DAC1
		0 Disabled
		1 Enabled
23	DAC0	Enable the clock to DAC0
		0 Disabled
		1 Enabled
22	ADC2	Enable the clock to ADC2
		0 Disabled
		1 Enabled
21	ADC1	Enable the clock to ADC1
		0 Disabled
		1 Enabled
20	ADC0	Enable the clock to ADC0
		0 Disabled
		1 Enabled
17	MPWM1	Enable the clock to MPWM1
		0 Disabled
		1 Enabled
16	MPWM0	Enable the clock to MPWM0
		0 Disabled

		1	Enabled
15	UART7	Enable the clock to UART7	
		0	Disabled
		1	Enabled
14	UART6	Enable the clock to UART6	
		0	Disabled
		1	Enabled
13	UART5	Enable the clock to UART5	
		0	Disabled
		1	Enabled
12	UART4	Enable the clock to UART4	
		0	Disabled
		1	Enabled
11	UART3	Enable the clock to UART3	
		0	Disabled
		1	Enabled
10	UART2	Enable the clock to UART2	
		0	Disabled
		1	Enabled
9	UART1	Enable the clock to UART1	
		0	Disabled
		1	Enabled
8	UART0	Enable the clock to UART0	
		0	Disabled
		1	Enabled
7	I2C3	Enable the clock to I2C3	
		0	Disabled
		1	Enabled
6	I2C2	Enable the clock to I2C2	
		0	Disabled
		1	Enabled
5	I2C1	Enable the clock to I2C1	
		0	Disabled
		1	Enabled
4	I2C0	Enable the clock to I2C0	
		0	Disabled
		1	Enabled
3	SPI3	Enable the clock to SPI3	
		0	Disabled
		1	Enabled
2	SPI2	Enable the clock to SPI2	
		0	Disabled
		1	Enabled
1	SPI1	Enable the clock to SPI1	
		0	Disabled
		1	Enabled
0	SPI0	Enable the clock to SPI0	
		0	Disabled
		1	Enabled

4.8.17 SCU_CSCR: Clock Source Control register

A34G43x includes a number of different clock sources for generating internal operation clocks. By configuring the SCU_CSCR, each of the clock sources can be enabled. This register is 32 bits wide.

SCU_CSCR=0x4000_0040

31 30 29 28 27 26 25 24	23 22 21 20 19 18 17 16	15 14 13 12 11 10 9 8	7	6	5	4	3	2	1	0
Reserved			LSECON	Reserved	LSICON	Reserved	HSICON	Reserved	HSECON	Reserved
-			0	-	1	-	1	-	0	-
-			RW	-	RW	-	RW	-	RW	-

7	LSECON[1]	Low-speed external oscillator control	
		0	Disables the low-speed external oscillator.
		1	Enables the low-speed external oscillator.
5	LSICON	Low-speed internal oscillator control	
		0	Disables the low-speed internal oscillator.
		1	Enables the low-speed internal oscillator.
3	HSICON	High-speed internal oscillator control	
		0	Disables the high-speed internal oscillator.
		1	Enables the high-speed internal oscillator.
1	HSECON[2]	High-speed external oscillator control	
		0	Disables the high-speed external oscillator.
		1	Enables the high-speed external oscillator.

NOTE:

1. XIN, XOUT, SXOUT, and SXIN cannot be used for GPIO or other alternate functions. When the HSE and LSE clocks are not in use, the clocks must be disabled by using software.

4.8.18 SCU_SCCR: System Clock Control Register

By configuring the SCU_SCCR, users can select a clock source as the system clock.

SCU_SCCR=0x4000_0044

31 30 29 28	27 26 25 24	23 22 21 20 19 18 17 16	15 14 13 12	11 10	9 8	7 6 5 4 3 2 1 0	
Reserved	HCLKDIV[3:0]	Reserved	PCLKDIV[2:0]	Reserved	PLLCLKSEL	Reserved	PLLPREDIV[1:0]	Reserved	MCLKSEL[2:0]
-	0010	-	000	-	0	-	00	-	010
-	RW	-	RW	-	RW	-	RW	-	RW

27 24	HCLKDIV[3:0]	Clock divider for HCLK input	
		0000	HCLK = MCLK
		0001	HCLK = MCLK / 2
		0010	HCLK = MCLK / 4
		0011	HCLK = MCLK / 8
		0100	HCLK = MCLK / 16
		0101	HCLK = MCLK / 32
		0110	HCLK = MCLK / 64
		0111	HCLK = MCLK / 128
		1000	HCLK = MCLK / 256
		1001	HCLK = MCLK / 512
		Other	Reserved
18 16	PCLKDIV[2:0]	Clock divider for PCLK input	
		000	PCLK = HCLK
		001	PCLK = HCLK / 2
		010	PCLK = HCLK / 4
		011	PCLK = HCLK / 8
		100	PCLK = HCLK / 16
		Other	Reserved
12	PLLCLKSEL(2)	PLL clock selection	
		0	Selects the HSI as the clock source for the PLL.
		1	Selects the HSE as the clock source for the PLL.
9 8	PLLPREDIV[1:0]	Clock divider for PLL input	
		00	FXIN = PLLCLKSEL / 1
		01	FXIN = PLLCLKSEL / 2
		10	FXIN = PLLCLKSEL / 4
		11	FXIN = PLLCLKSEL / 8
2 0	MCLKSEL[2:0](1)	Main system clock selection	
		000	LSI
		001	LSE(3)
		010	HSI
		110	HSE
		111	PLL
		Others	Reserved

NOTES:

1. When changing the PLLCLKSEL bits, both the HSI and HSE must be enabled.
2. When changing the MCLKSEL[2:0] bits, both clock sources should be alive. Ex) Both HSI and HSE should be alive, otherwise the chip will malfunction.

4.8.19 SCU_CMR: Clock Monitoring Register

The SCU_CMR register is used to monitor internal clocks for protection purposes by using the LSI oscillator. The SCU_CMR register is 32 bits wide.

SCU_CMR=0x4000_0048

31 30 29 28 27 26 25 24	23 22 21 20 19 18 17 16	15	14 13 12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved		MCLKREC	Reserved	LSEMNT	LSEIE	LSEFAIL	LSESTS	MCLKMNT	MCLKIE	MCLKFAIL	MCLKSTS	HSEMNT	HSEIE	HSEFAIL	HSESTS
-		0	-	0	0	0	0	0	0	0	0	0	0	0	0
-		RW	-	RW	RW	RWC1	RO	RW	RW	RWC1	RO	RW	RW	RWC1	RO

15	MCLKREC	Enable Automatic recovery when error of the MCLK occur	
		0	Switches the MCLK to the HSI (16/4 = 4 MHz) when error occur
		1	Disables the automatic recovery of the MCLK
11	LSEMNT	Enable external LSE monitoring	
		0	Disables external LSE monitoring.
		1	Enables external LSE monitoring.
10	LSEIE	Enable the external LSE error interrupt	
		0	Disables the external LSE error interrupt.
		1	Enables the external LSE error interrupt.
9	LSEFAIL	External LSE error interrupt flag	
		0	The external LSE error interrupt has not occurred.
		1	Read: The external LSE error interrupt has occurred. Write: Clears the interrupt flag.
8	LSESTS	External LSE status flag	
		0	The external LSE is oscillating normally.
		1	The external LSE is not oscillating.
7	MCLKMNT	Enable MCLK monitoring	
		0	Disables MCLK monitoring.
		1	Enables MCLK monitoring.
6	MCLKIE	Enable the MCLK error interrupt	
		0	Disables the MCLK error interrupt.
		1	Enables the MCLK error interrupt.
5	MCLKFAIL	MCLK error interrupt flag	
		0	The MCLK error interrupt has not occurred.
		1	Read: The MCLK error interrupt has occurred. Write: Clears the interrupt flag.
4	MCLKSTS	MCLK status flag	
		0	The MCLK is oscillating normally.
		1	The MCLK is not oscillating.
3	HSEMNT	Enable external HSE monitoring	
		0	Disables external HSE monitoring.
		1	Enables external HSE monitoring.
2	HSEIE	Enable the external HSE error interrupt	
		0	Disables the external HSE error interrupt.
		1	Enables the external HSE error interrupt.
1	HSEFAIL	External HSE error interrupt flag	
		0	The external HSE error interrupt has not occurred.
		1	Read: The external HSE error interrupt has occurred.

		Write: Clears the interrupt flag.	
0	HSESTS	External HSE status flag	
		0	The external HSE is oscillating normally.
		1	The external HSE is not oscillating.

NOTES:

1. Clock monitoring can be active only when the corresponding clock is enabled.
2. As clock monitoring is performed on the basis of the LSI, only those clocks that are faster than the LSI can be monitored.

4.8.20 SCU_COR: Clock Output Register

The SCU_COR defines the frequency division ratio at which the internal MCLK is fed to external devices. To use the CLKO pin in output mode, users must configure the CLKO to output mode in the pin mux. This register is 32 bits wide.

SCU_COR=0x4000_0050

31 30 29 28 27 26 25 24	23 22 21 20 19 18 17 16	15 14 13 12 11 10 9 8	7 6 5	4	3 2 1 0
Reserved			CLKOINSEL[2:0]	CLKOEN	CLKODIV[3:0]
-			000	0	1111
-			RW	RW	RW

7 5	CLKOINSEL[2:0]	Selection of the clock to send a signal from CLKO	
		000	LSI
		001	LSE
		100	MCLK
		101	HSI
		110	HSE
		111	PLL
		Other	Reserved
4	CLKOEN	Clock output use	
		0	Not use for the CLKO (maintains the ‘Low’ signal output.)
		1	Use for the CLKO.
3 0	CLKODIV[3:0]	Clock output divider value.	
		CLKO = MCLK, when CLKODIV[3:0] = ‘0’ CLKO = MCLK / (CLKODIV[3:0] × 2), when CLKODIV[3:0] > 0	

4.8.21 SCU_NMICR: NMI Control Register

The SCU_NMICR register controls NMIs (Non-Maskable Interrupts), which are software configurable. Any interrupt can be an NMI source. Once an NMI event occurs, the microcontroller jumps to the NMI handler.

SCU_NMICR=0x4000_0054

31 30 29 28 27 26 25 24	23 22 21 20 19 18 17 16	15	14	13	12	11 10 9 8	7	6	5	4	3	2	1	0
Reserved	NMISRC[7:0]	NMIINEN	Reserved	DFEDEN	CFEDEN	Reserved	MPWM1PROTEN	MPWM1OCEN	MPWM0PROTEN	MPWM0OCEN	WDTINTEN	MCLKFAILEN	LVIEN	
-	0x00	0	-	0	0	-	0	0	0	0	0	0	0	
-	RW	RW	-	RW	RW	-	RW	RW	RW	RW	RW	RW	RW	

23 16	NMISRC[7:0]	NMI source selection bit One of the interrupt priority levels can be used as the NMI source.
15	NMIINEN	Enable the NMIs (Interrupts that have been specified as NMI sources) 0 Disables. 1 Enables.
13	DFEDEN	Enable the data flash ECC double error interrupt as an NMI 0 Disables 1 Enables.
12	CFEDEN	Enable the code flash ECC double error interrupt as an NMI 0 Disables 1 Enables.
6	MPWM1PROTEN	Enable the MPWM1 protection interrupt as an NMI 0 Disables 1 Enables.
5	MPWM1OCEN	Enable the MPWM1 overcurrent protection interrupt as an NMI 0 Disables
4	MPWM0PROTEN	Enable the MPWM0 protection interrupt as an NMI 0 Disables 1 Enables.
3	MPWM0OCEN	Enable the MPWM0 overcurrent protection interrupt as an NMI 0 Disables 1 Enables.
2	WDTINTEN	Enable the WDT interrupt as an NMI 0 Disables 1 Enables.
1	MCLKFAILEN	Enable the MCLK error interrupt as an NMI 0 Disables 1 Enables.
0	LVIEN	Enable the LVI error interrupt as an NMI 0 Disables 1 Enables.

4.8.22 SCU_NMISR: NMI Status Register

The SCU_NMISR register is used to clear the NMI occurrence flags. Individual interrupt flags can be cleared after writing a specific key value to the identification key bit field.

SCU_NMISR=0x4000_0058

31 30 29 28 27 26 25 24	23 22 21 20 19 18 17 16	15	14	13	12	11 10 9 8	7	6	5	4	3	2	1	0
WTIDKY[7:0]	Reserved	NMIINTSTS	Reserved	DFEDSTS	CFEDSTS	Reserved		MPWM1PROTSTS	MPWM1OCSTS	MPWM0PROTSTS	MPWM0OCSTS	WDTINTSTS	MCLKFAILSTS	LVISTS
0x00	-	0	-	0	0	-		0	0	0	0	0	0	0
WO	-	RO	-	RWC1	RWC1	-		RWC1	RWC1	RWC1	RWC1	RWC1	RWC1	RWC1

31 24	WTIDKY[7:0]	Write identification key Writing 0x8C to the bit field enables clearing flags in this register.
15	NMIINTSTS	Whether an NMI has occurred (the flag can be cleared by writing to the triggered NMI's bit.) 0 No NMI has occurred. 1 An NMI has occurred.
13	DFEDSTS	Indicates the data flash ECC double error interrupt has occurred. 0 The interrupt has not occurred. 1 The interrupt has occurred (writing a 1 to the bit clears the flag).
12	CFEDSTS	Indicates the code flash ECC double error interrupt has occurred. 0 The interrupt has not occurred. 1 The interrupt has occurred (writing a 1 to the bit clears the flag).
6	MPWM1PROTSTS	Indicates the MPWM1 protection interrupt has occurred. 0 The interrupt has not occurred. 1 The interrupt has occurred (writing a '1' to the bit clears the flag).
5	MPWM1OCSTS	Indicates the MPWM1 overcurrent protection interrupt has occurred. 0 The interrupt has not occurred. 1 The interrupt has occurred (writing a '1' to the bit clears the flag).
4	MPWM0PROTSTS	Indicates the MPWM0 protection interrupt has occurred. 0 The interrupt has not occurred. 1 The interrupt has occurred (writing a '1' to the bit clears the flag).
3	MPWM0OCSTS	Indicates the MPWM0 overcurrent protection interrupt has occurred. 0 The interrupt has not occurred. 1 The interrupt has occurred (writing a '1' to the bit clears the flag).
2	WDTINTSTS	Indicates the WDT interrupt has occurred (to ensure that the bit is flagged only once, the timer must be reloaded before the flag is cleared). 0 The interrupt has not occurred. 1 The interrupt has occurred (Writing a '1' to the bit clears the flag).

1	MCLKFAILSTS	Indicates the MCLK error interrupt has occurred.
		0 The interrupt has not occurred.
		1 The interrupt has occurred (writing a '1' to the bit clears the flag).
0	LVISTS	Indicates the LVI interrupt has occurred.
		0 The interrupt has not occurred.
		1 The interrupt has occurred (writing a '1' to the bit clears the flag).

4.8.23 SCU_PLLCON: PLL Control Register

Integrated PLL will synthesize high speed clock for extremely high performance of CPU. The PLL is controlled by setting the register. The PLL control register is 32-bit register.

SCU_PLLCON=0x4000_0060

31	30 29 28 27 26 25 24	23	22	21	20	19 18 17 16	15 14 13 12 11 10 9 8	7 6 5 4 3 2 1 0
PLLLOCK	Reserved		PLLEN	BYPASSB	Reserved	PREDIV[3:0]	LOOPDIV[7:0]	POSTDIV[7:0]
0	-		0	0	-	0x0	0x00	0x00
RO	-		RW	RW	-	RW	RW	RW

31	PLLLOCK	PLL Lock status bit
		0 The PLL is in unlock state.
		1 The PLL is in lock state.
22	PLLEN	PLL Enable bit
		0 PLL is disabled
		1 PLL is enabled
21	BYPASSB	Decide whether to bypass the PLL input clock.
		0 Bypasses the PLL input clock.
		1 Uses the PLL multiplier clock (Does not bypass).
19 16	PREDIV[3:0]	FIN predivider bit
		0 to 15 PLLINCLK divided by (PREDIV + 1), (PLLINCLK / 1 to PLLINCLK / 8)
15 8	LOOPDIV[7:0]	Loop divider control (N1)
		N value in the formula (N1 = 0x00 to 0xFF)
7 0	POSTDIV[7:0]	Post divider control (P)
		P value in the formula (P = 0x0 to 0xFF)

NOTES:

1. Refer to section 4.4.5 for how to set the PLL clock using the SCU_PLLCON register.

4.8.24 SCU_VDCCON: VDC Control Register

The SCU_VDCCON register controls the VDC, which represents the microcontroller's internal voltage. The VDCWDLY[7:0] bit field defines the length of the SCU's warm-up delay. This register is 32 bits wide.

SCU_VDCCON=0x4000_0064

31 30 29 28 27 26 25 24	23 22 21 20 19 18 17 16	15 14 13 12 11 10 9 8	7 6 5 4 3 2 1 0
Reserved			VDCWDLY[7:0]
-			0x7F
-			RW

7 0	VDCWDLY[7:0]	VDC Warm-up delay count value When the SCU wakes up from DEEP-SLEEP mode, the Warm-up delay is inserted after waiting until the VDC output is stabilized. Default delay value is 0x7F (2 ms). The amount of delay can be set with this register up to 0xFF (4 ms).

NOTE:

1. Reserved bits should never be modified.

4.8.25 SCU_LVICR: LVI (Low-Voltage Indicator) Control Register

The LVI block detects low voltage by 16 levels. When it detects a specified level of low voltage which was predefined in SCU_LVICR register, an LVI interrupt and a flag are generated. This function prevents system malfunction caused by unstable voltage supply and steadily supports to drive user applications.

SCU_LVICR=0x4000_0068

31 30 29 28 27 26 25 24	23 22 21 20 19 18 17 16	15 14 13 12 11 10 9 8	7	6	5	4	3 2 1 0
Reserved			LVIEN	Reserved	LVIINTEN	LVIAON	LVIVS[3:0]
-			0	-	0	0	0000
-			RW	-	RW	RW	RW

7	LVIEN	Enable the LVI	
		0	Disables.
		1	Enables.
5	LVIINTEN	Enable the LVI interrupt.	
		0	Disables.
		1	Enables.
4	LVIAON	Whether to deactivate the LVI in DEEP-SLEEP mode	
		0	Disables LVI auto-off.
		1	Enables the LVI to automatically turn off when the microcontroller enters DEEP-SLEEP mode.
3 0	LVIVS[3:0]	LVI interrupt voltage selection bits	
		0000	1.57 V
		0001	1.66 V
		0010	1.78 V
		0011	1.88 V
		0100	1.99 V
		0101	2.11 V
		0110	2.28 V
		0111	2.43 V
		1000	2.66 V
		1001	3.00 V
		1010	3.14 V
		1011	3.53 V
		1100	3.71 V
		1101	4.03 V
		1110	4.21 V
		1111	4.47 V

NOTE:

1. Refer to section 4.6.1 and Figure 27

4.8.26 SCU_LVISR: LVI (Low-Voltage Indicator) Status Register

The SCU_LVISR register is 32 bits wide and indicates the LVI's operating status.

SCU_LVISR=0x4000_006C

31 30 29 28 27 26 25 24	23 22 21 20 19 18 17 16	15 14 13 12 11 10 9 8	7 6	5	4 3 2 1	0
WTIDKY[7:0]	Reserved			LVIFLAG	Reserved	LVIINTSTS
0x00	-			0	-	0
WO	-			RWC1	-	RO

31 24	WTIDKY[7:0]	Write identification key	
		Writing '0x7A' to the bit field enables clearing flags in this register.	
5	LVIFLAG	LVI interrupt flag	
		0	Not flagged
		1	Flagged (Writing a '1' to clear the flagged bit).
0	LVIINTSTS	Indicates the LVI interrupt is active (Raw data)	
		0	The LVI interrupt has ended.
		1	The LVI interrupt is active.

4.8.27 SCU_LVRCR: LVR (Low-Voltage Detect Reset) Control Register

The SCU_LVRCR register is 32 bits wide and defines the default mode for LVR reset operation.

SCU_LVRCR=0x4000_0070

31 30 29 28 27 26 25 24	23 22 21 20 19 18 17 16	15 14 13 12 11 10 9 8	7 6 5	4	3 2 1 0
Reserved		LVREN[7:0]	Reserved	LVRAON	LVRVS[3:0]
-		0x00	-	0	0101
-		RW	-	RW	RW

15 8	LVREN[7:0](1)	LVR reset operation control master configuration.	
		0xAA	Disable LVR reset operation
		Others	Enables LVR reset operation
4	LVRAON	Whether to deactivate the LVR reset in DEEP-SLEEP mode	
		0	Disables LVR reset auto-off.
		1	Enables the LVR reset to automatically turn off when the microcontroller enters DEEP-SLEEP mode.
0	LVRVS [3:0]	LVR voltage selection bits	
		0000	1.57 V
		0001	1.66 V
		0010	1.78 V
		0011	1.88 V
		0100	1.99 V
		0101	2.11 V
		0110	2.28 V
		0111	2.43 V
		1000	2.66 V
		1001	3.00 V
		1010	3.14 V
		1011	3.53 V
		1100	3.71 V
		1101	4.03 V
		1110	4.21 V
		1111	4.47 V

NOTES:

1. The LVDRST bit in SCU_RSER register must be enabled before the LVREN[7:0] bits are set to trigger a reset operation at low voltage.
2. See section 4.6.2.

4.8.28 SCU_EOSCR: External Oscillator Control Register

The SCU_EOSCR register defines the drive current of the external main crystal oscillator and the delay time to reduce noises. This register is 32-bit wide.

SCU_EOSCR=0x4000_0080

31 30 29 28 27 26 25 24	23 22 21 20 19 18 17 16	15 14 13 12 11 10 9 8	7 6 5	4	3 2	1 0
Reserved	LSEISEL[1:0]	Reserved	LSENFEN	Reserved	HSEISEL[1:0]	Reserved	HSENFEN	Reserved	HSENFSEL[1:0]
-	10	-	1	-	10	-	0	-	01
-	RW	-	RW	-	RW	-	RW	-	RW

25 24	LSEISEL[1:0]	LSE current capability selection	
		00	500 nA
		01	1 µA
		10	2 µA
		11	2.5 µA
16	LSEFEN	Must be '1'.	
9 8	HSEISEL[1:0]	HSE current capability selection	
		00	12 MHz < f_{OUT} ≤ 16 MHz
		01	8 MHz < f_{OUT} ≤ 12 MHz
		10	4 MHz < f_{OUT} ≤ 8 MHz
		11	f_{OUT} = 4 MHz
4	HSENFEN	HSE noise filter enable bit	
		0	Disable
		1	Enable
1 0	HSENFSEL[1:0]	HSE noise filter selection	
		00	23 ns
		01	18 ns
		10	13 ns
		11	8 ns

Table 31. External Oscillator Control Settings

Frequency (MHz)	HSENFSEL[1:0]	HSEISEL[1:0]	Noise Filter Delay (ns)
4	00	11	25
8	01	10	20
12	10	01	15
16	11	00	10

4.8.29 SCU_MCCR1: Miscellaneous Clock Control Register 1

SCU_MCCR1 provides a different MCLK division ratio for each peripheral. This register is 32-bit wide.

SCU_MCCR1=0x4000_0090

31 30 29 28 27 26 25 24	23 22 21 20 19 18 17 16	15 14 13 12 11 10 9 8	7 6 5 4 3 2 1 0
Reserved	WDTCSEL[2:0]	WDTCDIV[7:0]	Reserved	STCSEL[2:0]	STCDIV[7:0]
-	000	0x01	-	000	0x00
-	RW	RW	-	RW	RW

25 24	WDTCSEL[2:0]	WDT clock source selection bit	
		000	LSI (500 kHz)
		001	LSE (32.768 kHz)
		100	MCLK
		101	HSI (16 MHz)
		110	HSE
		111	PLL
		Other	Reserved
23 16	WDTCDIV[7:0]	WDT clock divider (N)	
		0x00	Disables the divider
		0xN	Clock source / N, N must be larger than zero.
10 8	STCSEL[2:0]	SYSTICK clock source selection bit	
		000	LSI (500 kHz)
		001	LSE (32.768 kHz)
		100	MCLK
		101	HSI (16 MHz)
		110	HSE
		111	PLL
		Other	Reserved
7 0	STCDIV[7:0](1)	SYSTICK clock divider (N)	
		0x00	Disables the divider
		0xN	Clock source / (2 × N), N must be larger than zero.

CAUTION:

1. If the SYSTICK clock source register is set to '0' (external clock), STCDIV[7:0] cannot use a clock divider with ‘1’, and the clock divider must use two or more. Otherwise, the SYSTICK clock will not work.

4.8.30 SCU_MCCR3: Miscellaneous Clock Control Register 3

SCU_MCCR3 provides a different MCLK division ratio for each peripheral. If timer clock is set using the MCCR, the timer clock must be slower than the main clock divided by two. This register is 32-bit wide.

SCU_MCCR3=0x4000_0098

31 30 29 28 27	26 25 24	23 22 21 20 19 18 17 16	15 14 13 12 11	10 9 8	7 6 5 4 3 2 1 0
Reserved	TIMER59CSEL[2:0]	TIMER59CDIV[7:0]	Reserved	TIMER04CSEL[2:0]	TIMER04CDIV[7:0]
-	000	0x00	-	000	0x00
-	RW	RW	-	RW	RW

26 24	TIMER59CSEL[2:0]	Clock source selection for TIMER5 through TIMER9	
		000	LSI (500 kHz)
		001	LSE (32.768 kHz)
		100	MCLK
		101	HSI (16 MHz)
		110	HSE
		111	PLL
		Other	Reserved
23 16	TIMER59CDIV[7:0](1)	Clock divider (N) for TIMER5 through TIMER9	
		0x00	Disables the divider
		0xN	Clock source / N, N must be larger than 0. If the timer input clock (TMCLK) is set in this register, the clock speed must satisfy PCLK ≥ TMCLK × 3.
10 8	TIMER04CSEL[2:0]	Clock source selection for TIMER0 through TIMER4	
		000	LSI (500 kHz)
		001	LSE (32.768 kHz)
		100	MCLK
		101	HSI (16 MHz)
		110	HSE
		111	PLL
		Other	Reserved
7 0	TIMER04CDIV[7:0](1)	Clock divider (N) for TIMER0 through TIMER4	
		0x00	Disables the divider
		0xN	Clock source / N, N must be larger than 0. If the timer input clock (TMCLK) is set in this register, the clock speed must satisfy PCLK ≥ TMCLK × 3.

NOTE:

1. Because the PCLK is the default operation clock, the clock source set by SCU_MCCRx register should be slower than the PCLK. Therefore, if the clock source set by SCU_MCCRx register uses the same frequency as the PCLK, it should be divided more than three times. If the PCLK is 1/2 slower than the clock source set by SCU_MCCRx register, frequency should be divided more than six times.

4.8.31 SCU_MCCR4: Miscellaneous Clock Control Register 4

SCU_MCCR4 register provides a different MCLK division ratio for each peripheral. This register is 32-bit wide.

SCU_MCCR4=0x4000_009C

31 30 29 28 27	26 25 24	23 22 21 20 19 18 17 16	15 14 13 12 11 10 9 8	7 6 5 4 3 2 1 0
Reserved	PGADCSEL[2:0]	PGADCDIV[7:0]	Reserved	
-	000	0x00	-	
-	RW	RW	-	

26 24	PGADCSEL[2:0] (PA, PB)	Clock source for port group A (PA, PB) debouncing	
		000	LSI (500 kHz)
		001	LSE (32.768 kHz)
		100	MCLK
		101	HSI (16 MHz)
		110	HSE
		111	PLL
		Other	Reserved
23 16	PGADCDIV[7:0] (PA, PB)	Clock divider N for port group A (PA, PB) debouncing	
		0x00	Disables the divider
		0xN	Clock source / N, N must be larger than zero.

4.8.32 SCU_MCCR5: Miscellaneous Clock Control Register 5

SCU_MCCR5 register provides a different MCLK division ratio for each peripheral. This register is 32 bits wide.

SCU_MCCR5=0x4000_00A0

31 30 29 28 27	26 25 24	23 22 21 20 19 18 17 16	15 14 13 12 11	10 9 8	7 6 5 4 3 2 1 0
Reserved	PGCDCSEL[2:0]	PGCDCDIV[7:0]	Reserved	PGBDCSEL[2:0]	PGBDCDIV[7:0]
-	000	0x00	-	000	0x00
-	RW	RW	-	RW	RW

26 24	PGCDCSEL[2:0] (PE, PF, PG)	Clock source for port group C (PE, PF, PG) debouncing	
		000	LSI (500 kHz)
		001	LSE (32.768 kHz)
		100	MCLK
		101	HSI (16 MHz)
		110	HSE
		111	PLL
		Other	Reserved
23 16	PGCDCDIV[7:0] (PE, PF, PG)	Clock divider (N) for port group C (PE, PF, PG) debouncing	
		0x00	Disables the divider
		0xN	Clock source / N, N must be larger than zero.
10 8	PGBDCSEL[2:0] (PC, PD)	Clock source for port group B (PC, PD) debouncing	
		000	LSI (500 kHz)
		001	LSE (32.768 kHz)
		100	MCLK
		101	HSI (16 MHz)
		110	HSE
		111	PLL
		Other	Reserved
7 0	PGBDCDIV[7:0] (PC, PD)	Clock divider (N) for port group B (PC, PD) debouncing	
		0x00	Disables the divider
		0xN	Clock source / N, N must be larger than zero.

4.8.33 SCU_MCCR6: Miscellaneous Clock Control Register 6

SCU_MCCR6 register provides a different MCLK division ratio for each peripheral. This register is 32 bits wide.

SCU_MCCR6=0x4000_00A4

31 30 29 28 27 26 25 24	23 22 21 20 19 18 17 16	15 14 13 12 11 10 9 8	7 6 5 4 3 2 1 0
Reserved	FRT1CSEL[2:0]	FRT1CDIV[7:0]	Reserved	FRT0CSEL[2:0]	FRT0CDIV[7:0]
-	000	0x00	-	000	0x00
-	RW	RW	-	RW	RW

26 24	FRT1CSEL[2:0]	FRT1 clock source selection bit	
		000	LSI (500 kHz)
		001	LSE (32.768 kHz)
		100	MCLK
		101	HSI (16 MHz)
		110	HSE
		111	PLL
		Other	Reserved
23 16	FRT1CDIV[7:0]	FRT1 clock divider (N)	
		0x00	Disables the divider
		0xN	Clock source / N, N must be larger than zero.
10 8	FRT0CSEL[2:0]	FRT clock source selection bit	
		000	LSI (500 kHz)
		001	LSE (32.768 kHz)
		100	MCLK
		101	HSI (16 MHz)
		110	HSE
		111	PLL
		Other	Reserved
7 0	FRT0CDIV[7:0]	FRT clock divider (N)	
		0x00	Disables the divider
		other	Clock source / N, N must be larger than zero.

4.8.34 SCU_MCCR7: Miscellaneous Clock Control Register 7

SCU_MCCR7 provides a different MCLK division ratio for each peripheral. This register is 32-bit wide.

SCU_MCCR7=0x4000_00A8

31 30 29 28 27 26 25 24	23 22 21 20 19 18 17 16	15 14 13 12 11 10 9 8	7 6 5 4 3 2 1 0
Reserved	UARTCSEL[2:0]	UARTCDIV[7:0]	Reserved
-	000	0x01	-
-	RW	RW	-

26 24	UARTCSEL[2:0]	UART clock source selection bit (UART0 through UART12)	
		000	LSI (500 kHz)
		001	LSE (32.768 kHz)
		100	MCLK
		101	HSI (16 MHz)
		110	HSE
		111	PLL
		Other	Reserved
23 16	UARTCDIV[7:0]	UART clock divider (N)	
		0x00	Disables the divider
		other	Clock source / N, N must be larger than zero. If the UART clock is set in this register, the clock speed must satisfy PCLK ≥ UART clock set by SCU_MCCR7 × 2.

4.8.35 SCU_MCCR8: Miscellaneous Clock Control Register 8

SCU_MCCR8 register provides a different MCLK division ratio for each peripheral. This register is 32 bits wide.

SCU_MCCR8=0x4000_00AC

31 30 29 28 27 26 25 24	23 22 21 20 19 18 17 16	15 14 13 12 11 10 9 8	7 6 5 4 3 2 1 0
Reserved	CANFD1CSEL[2:0]	CANFD1CDIV[7:0]	Reserved	CANFD0CSEL[2:0]	CANFD0CDIV[7:0]
-	000	0x00	-	000	0x00
-	RW	RW	-	RW	RW

26 24	CANFD1CSEL[2:0]	CANFD1 clock source selection bit	
		100	MCLK (No other clock sources are supported, so only PLLs should be used.)
		111	PLL
		Other	Reserved
23 16	CANFD1CDIV[7:0](1)	CANFD1 clock divider (N)	
		0x00	Disables the divider
		0xN	Clock source / N, N must be larger than 0. Recommended CANFD frequencies: 20, 40, 80 MHz Example) For the CANFD of 80 MHz, 160 MHz must be divided in two, so MCLK must be set to PLL 160 MHz.
10 8	CANFD0CSEL[2:0]	CANFD0 clock source selection bit	
		100	MCLK (Only use PLL, another clock source does not support)
		111	PLL
		Other	Reserved
7 0	CANFD0CDIV[7:0](1)	CANFD0 clock divider (N)	
		0x00	Disables the divider
		0xN	Clock source / N, N must be larger than 0. Recommended CANFD frequencies: 20, 40, 80 MHz Example) For the CANFD of 80 MHz, 160 MHz must be divided in two, so MCLK must be set to PLL 160 MHz.

4.8.36 SCU_SYSTEN: System Access Enable Register

The SCU_SYSTEN determines whether to allow the changes of the settings in all SCU registers.

SCU_SYSTEN=0x4000_00F0

31 30 29 28 27 26 25 24	23 22 21 20 19 18 17 16	15 14 13 12 11 10 9	8	7 6 5 4 3 2 1 0
Reserved			ENS	SYSTEN[7:0]
-			0	0x00
-			RO	RW

8	ENS	Enable SCU registers enable status bit
		0 Disables the write access to SCU registers.
		1 Enables the write access to SCU registers.
7 0	SYSTEN[7:0]	SCU register access enable key. Writing ‘0x57’ and then ‘0x75’ to the bit field enables writing new values to SCU registers. After this, write a different value to this bit field to protect the SCU registers against being updated with new values. However, access to the NMISR and LVISR registers is not defined by the SYSTEM register. Their bits can be cleared regardless of SYSTEN enablement.

4.8.37 SCU_SPSCR: SRAM Power Sequence Control Register

The SCU_SPSCR register can control the power of SRAM. This register is 32-bit wide.

SCU_SPSCR=0x4000_0100

31 30 29 28 27 26 25 24	23 22 21 20 19 18 17 16	15 14	13 12	11 10	9 8	7 6	5 4	3 2	1 0
SPS_KEY[7:0]	Reserved	SPSEN	SRAM7_PMODE[1:0]	SRAM6_PMODE[1:0]	SRAM5_PMODE[1:0]	SRAM4_PMODE[1:0]	SRAM3_PMODE[1:0]	SRAM2_PMODE[1:0]	SRAM1_PMODE[1:0]	SRAM0_PMODE[1:0]
0x00	-	0	00	00	00	00	00	00	00	00
WO	-	WO	RW	RW	RW	RW	RW	RW	RW	RO

31 24	SPS_KEY[7:0]	SRAM power sequence access key (auto-clear) Writing '0x4E' to the bit field enables clearing flags in this register.	
16	SPSEN	SRAM power sequence start control bit (auto-clear)	
		0	No operation
		1	Operation
2x+1 2x	SRAMx_PMODE [1:0]	SRAMx power mode, x: 0 to 7	
		00	SRAM Run
		10	SRAM Sleep
		x1	SRAM Power Down

NOTES:

1. An SRAM block that has entered power down mode cannot be erased, written, or read.
2. When setting the power down mode of SRAM, if the SPSEN bit of the **SCU_SPSCR** register is not set together, the value will be reflected in the **SCU_SPSCR** register, but it will not actually affect the SRAM mode. (The busy flag of the **SCU_SPSSR** register will not be triggered.)
3. When powering down the SRAM, please ensure that the stack pointer is not located in the SRAM region being powered down. Otherwise, the system may malfunction. This precaution must be taken, for example, if the microcontroller powers down the SRAM to reduce current consumption before entering Sleep or Deep-Sleep (Stop) mode.

Table 32. SRAM Block Description

Parameter	Address	Description
SRAM0	0x2000_0000 ~ 0x2000_7FFF	Always Run
SRAM1	0x2000_8000 ~ 0x2000_FFFF	Run/Sleep/Power down
SRAM2	0x2001_0000 ~ 0x2001_7FFF	Run/Sleep/Power down
SRAM3	0x2001_8000 ~ 0x2001_FFFF	Run/Sleep/Power down
SRAM4	0x2002_0000 ~ 0x2002_7FFF	Run/Sleep/Power down
SRAM5	0x2002_8000 ~ 0x2002_FFFF	Run/Sleep/Power down
SRAM6	0x2003_0000 ~ 0x2003_7FFF	Run/Sleep/Power down
SRAM7	0x2003_8000 ~ 0x2003_FFFF	Run/Sleep/Power down

4.8.38 SCU_SPSSR: SRAM Power Sequence Status Register

The SCU_SPSSR register provides the status of each SRAM. This register is 32-bit wide.

SCU_SPSSR=0x4000_0104

31 30	29 28	27 26	25 24	23 22	21 20	19 18	17 16	15 14 13 12 11 10 9 8	7 6 5 4 3 2	1	0
SRAM7_STS [1:0]	SRAM6_STS [1:0]	SRAM5_STS [1:0]	SRAM4_STS [1:0]	SRAM3_STS [1:0]	SRAM2_STS [1:0]	SRAM1_STS [1:0]	SRAM0_STS [1:0]	Reserved		SPSBUSY	SPSDONE
00	00	00	00	00	00	00	00	-		0	0
RO	RO	RO	RO	RO	RO	RO	RO	-		RO	RW

2x+17 2x+16	SRAMx_STS [1:0]	SRAMx power mode status, x: 0 to 7	
		00	SRAM Run
		10	SRAM Sleep
		x1	SRAM Power Down
1	SPSBUSY	SRAM power sequence busy check	
		0	Not busy
		1	Busy
0	SPSDONE	SRAM power sequence done status	
		0	Busy
		1	Write done (Writing a '1' to the bit clears this flag)

4.8.39 SCU_SPSDR: SRAM Power Sequence Delay Register

The SCU_SPSDR register can be configured to set the SRAM power sequence delay. This register is 32-bit wide.

SCU_SPSDR=0x4000_0108

31 30 29 28 27 26 25 24	23 22 21 20 19 18 17 16	15 14 13 12 11 10 9 8	7 6 5 4 3 2 1 0
SPS_KEY[7:0]	Reserved	SPSDLY[7:0]	
0x00	-	0xFFF	
WO	-	RW	

31 24	SPS_KEY[7:0]	SRAM power sequence access key (auto-clear)	
		Writing '0x4E' to the bit field enables clearing flags in this register.	
11 0	SPSDLY[7:0]	SRAM power sequence delay	

NOTE:

1. When the SPSEN bit of the SCU_SPSCR register is set to '1', the power of the SRAM is determined sequentially. A total of six delays operate, with the delay clock functioning as MCLK.

4.8.40 SCU_PRER3: Peripheral Reset Enable Register 3

The SCU_PRER3 register is 32 bits wide.

SCU_PRER3=0x4000_0120

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved		PWM59	PWM04	Reserved				SPI11	SPI10	SPI9	SPI8	SPI7	SPI6	SPI5	SPI4	Reserved			UART12	UART11	UART10	UART9	UART8	Reserved							QSPI
-		1	1	-				1	1	1	1	1	1	1	1	-			1	1	1	1	1	-							1
-		RW	RW	-				RW	RW	RW	RW	RW	RW	RW	RW	-			RW	RW	RW	RW	RW	-							RW

29	PWM59	PWM59 reset mask
		0 Masked
		1 Not masked
28	PWM04	PWM04 reset mask
		0 Masked
		1 Not masked
23	SPI11	SPI11 reset mask
		0 Masked
		1 Not masked
22	SPI10	SPI10 reset mask
		0 Masked
		1 Not masked
21	SPI9	SPI9 reset mask
		0 Masked
		1 Not masked
20	SPI8	SPI8 reset mask
		0 Masked
		1 Not masked
19	SPI7	SPI7 reset mask
		0 Masked
		1 Not masked
18	SPI6	SPI6 reset mask
		0 Masked
		1 Not masked
17	SPI5	SPI5 reset mask
		0 Masked
		1 Not masked
16	SPI4	SPI4 reset mask
		0 Masked
		1 Not masked
12	UART12	UART12 reset mask
		0 Masked
		1 Not masked
11	UART11	UART11 reset mask
		0 Masked
		1 Not masked
10	UART10	UART10 reset mask
		0 Masked
		1 Not masked
9	UART9	UART9 reset mask

		0	Masked
		1	Not masked
8	UART8	UART8 reset mask	
		0	Masked
		1	Not masked
0	QSPI	QSPI reset mask	
		0	Masked
		1	Not masked

4.8.41 SCU_PER3: Peripheral Enable Register 3

The SCU_PER3 is 32bits wide.

SCU_PER3=0x4000_0128

31 30 29 28 27 26 25 24	23 22 21 20 19 18 17 16	15 14 13 12 11 10 9 8	7 6 5 4 3 2 1 0
Reserved | PWM59 | PWM04 | Reserved	SPI11 | SPI10 | SPI9 | SPI8 | SPI7 | SPI6 | SPI5 | SPI4	Reserved | UART12 | UART11 | UART10 | UART9 | UART8	Reserved | QSPI
- | 0 | 0 | -	0 | 0 | 0 | 0 | 0 | 0 | 0 | 0	- | 0 | 0 | 0 | 0 | 0	- | 0
- | RW | RW | -	RW | RW | RW | RW | RW | RW | RW | RW	- | RW | RW | RW | RW | RW	- | RW

29	PWM59	Enable PWM59
		0 Disabled
		1 Enabled
28	PWM04	Enable PWM04
		0 Disabled
		1 Enabled
23	SPI11	Enable SPI11
		0 Disabled
		1 Enabled
22	SPI10	Enable SPI10
		0 Disabled
		1 Enabled
21	SPI9	Enable SPI9
		0 Disabled
		1 Enabled
20	SPI8	Enable SPI8
		0 Disabled
		1 Enabled
19	SPI7	Enable SPI7
		0 Disabled
		1 Enabled
18	SPI6	Enable SPI6
		0 Disabled
		1 Enabled
17	SPI5	Enable SPI5
		0 Disabled
		1 Enabled
16	SPI4	Enable SPI4
		0 Disabled
		1 Enabled
12	UART12	Enable UART12
		0 Disabled
		1 Enabled
11	UART11	Enable UART11
		0 Disabled
		1 Enabled
10	UART10	Enable UART10
		0 Disabled
		1 Enabled
9	UART9	Enable UART9

		0	Disabled
		1	Enabled
8	UART8	Enable UART8	
		0	Disabled
		1	Enabled
0	QSPI	Enable QSPI	
		0	Disabled
		1	Enabled

4.8.42 SCU_PCER3: Peripheral Clock Enable Register 3

To use peripheral unit, its clock should be activated by writing '1' to the corresponding bit in the SCU_PCER3 register.

SCU_PCER3=0x4000_0130

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved		PWM59	PWM04	Reserved				SPI11	SPI10	SPI9	SPI8	SPI7	SPI6	SPI5	SPI4	Reserved			UART12	UART11	UART10	UART9	UART8	Reserved							QSPI
-		0	0	-				0	0	0	0	0	0	0	0	-			0	0	0	0	0	-							0
-		RW	RW	-				RW	RW	RW	RW	RW	RW	RW	RW	-			RW	RW	RW	RW	RW	-							RW

29	PWM59	Enable PWM59
		0 Disabled
		1 Enabled
28	PWM04	Enable PWM04
		0 Disabled
		1 Enabled
23	SPI11	Enable SPI11
		0 Disabled
		1 Enabled
22	SPI10	Enable SPI10
		0 Disabled
		1 Enabled
21	SPI9	Enable SPI9
		0 Disabled
		1 Enabled
20	SPI8	Enable SPI8
		0 Disabled
		1 Enabled
19	SPI7	Enable SPI7
		0 Disabled
		1 Enabled
18	SPI6	Enable SPI6
		0 Disabled
		1 Enabled
17	SPI5	Enable SPI5
		0 Disabled
		1 Enabled
16	SPI4	Enable SPI4
		0 Disabled
		1 Enabled
12	UART12	Enable UART12
		0 Disabled
		1 Enabled
11	UART11	Enable UART11
		0 Disabled
		1 Enabled
10	UART10	Enable UART10
		0 Disabled

		1	Enabled
9	UART9	Enable UART9	
		0	Disabled
		1	Enabled
8	UART8	Enable UART8	
		0	Disabled
		1	Enabled
0	QSPI	Enable QSPI	
		0	Disabled
		1	Enabled

4.8.43 SCU Register Map Summary

Table 33. SCU CHIPCONFIG Register Map Summary

Offset	Register name	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0x00	CHIPCONFIG_ VENDORID	VENDID[31:0]																															
	Reset value	0	1	0	0	0	0	0	1	0	1	0	0	0	0	1	0	0	1	0	0	1	1	1	1	0	1	0	1	0	1	1	0
0x04	CHIPCONFIG_ CHIPID	CHIPID[31:0]																															
	Reset value	0	1	0	0	0	1	1	1	0	0	1	1	0	1	0	0	0	1	0	0	0	1	0	0	1	0	1	0	x	x	x	x
0x08	CHIPCONFIG_ REVNR	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	REVNO[7:0]							
	Reset value																									x	x	x	x	x	x	x	x

Table 34. SCU CHIPCONFIG2 Register Map Summary

Offset	Register name	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0xFC	CHIPCONFIG2_ CHIPID2	CHIPID2[31:0]																															
	Reset value	0	1	0	0	0	1	1	1	0	0	1	1	0	1	0	0	0	1	0	0	0	0	1	1	x	x	x	x	x	x	x	x

Table 35. SCU Register Map Summary

Offset	Register name	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0x04	SCU_SMR	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	LSEAON	HSEAON	PLLAON	HSIAON	LSIAON	VDCAON	Res.	Res.	PREVMODE[1:0]		Res.	Res.	Res.	Res.
	Reset value																			0	0	0	0	0	0			1	1				
0x08	SCU_SRCR	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	STBYO	Res.	Res.	Res.	SWRST
	Reset value																												0				0
0x10	SCU_WUER	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	GPIOGWUE	GPIOFWUE	GPIOEWUE	GPIODWUE	GPIOCWUE	GPIOBWUE	GPIOAWUE	Res.	Res.	Res.	Res.	FRT1WUE	FRT0WUE	WDTWUE	LVIWUE
	Reset value																		0	0	0	0	0	0	0					0	0	0	0
0x14	SCU_WUSR	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	GPIOGWU	GPIOFWU	GPIOEWU	GPIODWU	GPIOCWU	GPIOBWU	GPIOAWU	Res.	Res.	Res.	Res.	FRT1WU	FRT0WU	WDTWU	LVIWU
	Reset value																		0	0	0	0	0	0	0					0	0	0	0
0x18	SCU_RSER	BANKRST	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	LOCKUPRST	Res.	PINRST	CPURST	SWRST	WDTRST	MCLKRST	LSERST	HSERST	LVDRST
	Reset value	0																						0		1	1	0	1	0	0	0	1
0x1C	SCU_RSSR	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	LOCKUPRST	PORST	PINRST	CPURST	SWRST	WDTRST	MCLKFRST	LSERST	HSERST	LVDRST
	Reset value																							0	1	0	0	0	0	0	0	0	1

Table 35. SCU Register Map Summary (continued)

Offset	Register name	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0x20	SCU_PRER1	Res.	Res.	Res.	Res.	Res.	Res.	TIMER9	TIMER8	TIMER7	TIMER6	TIMER5	TIMER4	TIMER3	TIMER2	TIMER1	TIMER0	Res.	GPIOG	GPIOF	GPIOE	GPIOD	GPIOC	GPIOB	GPIOA	FRT1	FRT0	Res.	DMA	DFMC	WDT	CFMC	SCU
	Reset value							1	1	1	1	1	1	1	1	1	1		1	1	1	1	1	1	1	1	1		1	1	1	1	1
0x24	SCU_PRER2	RNG	AES	CRC	CMP	CANFD1	CANFD0	Res.	DAC1	DAC0	ADC2	ADC1	ADC0	Res.	Res.	MPWM1	MPWM0	UART7	UART6	UART5	UART4	UART3	UART2	UART1	UART0	I2C3	I2C2	I2C1	I2C0	SPI3	SPI2	SPI1	SPI0
	Reset value	1	1	1	1	1	1		1	1	1	1	1			1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1
0x28	SCU_PER1	Res.	Res.	Res.	Res.	Res.	Res.	TIMER9	TIMER8	TIMER7	TIMER6	TIMER5	TIMER4	TIMER3	TIMER2	TIMER1	TIMER0	Res.	GPIOG	GPIOF	GPIOE	GPIOD	GPIOC	GPIOB	GPIOA	FRT1	FRT0	Res.	DMA	Res.	Res.	Res.	Res.
	Reset value							0	0	0	0	0	0	0	0	0	0		0	0	0	0	0	0	0	0	0		0	1	1	1	1
0x2C	SCU_PER2	RNG	AES	CRC	CMP	CANFD1	CANFD0	Res.	DAC1	DAC0	ADC2	ADC1	ADC0	Res.	Res.	MPWM1	MPWM0	UART7	UART6	UART5	UART4	UART3	UART2	UART1	UART0	I2C3	I2C2	I2C1	I2C0	SPI3	SPI2	SPI1	SPI0
	Reset value	0	0	0	0	0	0		0	0	0	0	0			0	0	0	0	0	0	0	0	0	1	0	0	0	0	0	0	0	1
0x30	SCU_PCER1	Res.	Res.	Res.	Res.	Res.	Res.	TIMER9	TIMER8	TIMER7	TIMER6	TIMER5	TIMER4	TIMER3	TIMER2	TIMER1	TIMER0	Res.	GPIOG	GPIOF	GPIOE	GPIOD	GPIOC	GPIOB	GPIOA	FRT1	FRT0	Res.	DMA	Res.	Res.	Res.	Res.
	Reset value							0	0	0	0	0	0	0	0	0	0		0	0	0	0	0	0	0	0	0		0	1	1	1	1
0x34	SCU_PCER2	RNG	AES	CRC	CMP	CANFD1	CANFD0	Res.	DAC1	DAC0	ADC2	ADC1	ADC0	Res.	Res.	MPWM1	MPWM0	UART7	UART6	UART5	UART4	UART3	UART2	UART1	UART0	I2C3	I2C2	I2C1	I2C0	SPI3	SPI2	SPI1	SPI0
	Reset value	0	0	0	0	0	0		0	0	0	0	0			0	0	0	0	0	0	0	0	0	1	0	0	0	0		0	0	1
0x40	SCU_CSCR	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	LSECON	Res.	LSICON	Res.	HSICON	Res.	HSECON	Res.
	Reset value																									0		1		1		0	
0x44	SCU_SCCR	Res.	Res.	Res.	Res.	HCLKDIV[3:0]				Res.	Res.	Res.	Res.	Res.	PCLKDIV[2:0]			Res.	Res.	Res.	PLLCLKSEL	Res.	Res.	PLLPREDIV[1:0]		Res.	Res.	Res.	Res.	Res.	MCLKSEL[2:0]		
	Reset value					0	0	1	0						0	0	0				0			0	0							0	1	0
0x48	SCU_CMR	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	MCLKREC	Res.	Res.	Res.	LSEMNT	LSEIE	LSEFAIL	LSESTS	MCLKMNT	MCLKIE	MCLKFAIL	MCLKSTS	HSEMNT	HSEIE	HSEFAIL	HSESTS
	Reset value																	0				0	0	0	0	0	0	0	0	0	0	0	0
0x50	SCU_COR	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	CLKOINSEL[2:0]		CLKOEN	CLKODIV[3:0]				
	Reset value																									0	0	0	0	1	1	1	1

Table 35. SCU Register Map Summary (continued)

Offset	Register name	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0x54	SCU_NMICR	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	NMISRC[7:0]								NMIINEN	Res.	DFEDEN	CFEDEN	Res.	Res.	Res.	Res.	MPWM1PROTEN	MPWM1OCEN	MPWM0PROTEN	MPWM0OCEN	WDTINTEN	MCLKFAILEN	LVIEN
	Reset value									0	0	0	0	0	0	0	0	0		0	0						0	0	0	0	0	0	0
0x58	SCU_NMISR	WTIDKY[7:0]								Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	NMIINTSTS	Res.	DFEDSTS	CFEDSTS	Res.	Res.	Res.	Res.	MPWM1PROTSTS	MPWM1OCSTS	MPWM0PROTSTS	MPWM0OCSTS	WDTINTSTS	MCLKFAILSTS	LVISTS
	Reset value	0	0	0	0	0	0	0	0									0		0	0						0	0	0	0	0	0	0
0x60	SCU_PLLCON	PLLLOCK	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	PLLEN	BYPASSB	Res.	PREDIV[3:0]				LOOPDIV[7:0]								POSTDIV[7:0]							
	Reset value	0									0	0		0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
0x64	SCU_VDCCON	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	VDCWDLY[7:0]							
	Reset value																									0	1	1	1	1	1	1	1
0x68	SCU_LVICR	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	LVIEN	Res.	LVIINTEN	LVIAON	LVIVS[3:0]			
	Reset value																									0		0	0	0	0	0	0
0x6C	SCU_LVISR	WTIDKY[7:0]								Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	LVIIFLAG	Res.	Res.	Res.	Res.	LVIINTSTS
	Reset value	0	0	0	0	0	0	0	0																			0					0
0x70	SCU_LVRCR	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	LVREN[7:0]								Res.	Res.	Res.	LVRAON	LVRVS[3:0]			
	Reset value																	0	0	0	0	0	0	0	0				0	0	1	0	1
0x80	SCU_EOSCR	Res.	Res.	Res.	Res.	Res.	Res.	LSEISEL[1:0]		Res.	Res.	Res.	Res.	Res.	Res.	Res.	LSENFEN	Res.	Res.	Res.	Res.	Res.	Res.	HSEISEL[1:0]		Res.	Res.	HSENFEN	Res.	Res.	HSENFSEL[1:0]	
	Reset value							0	0								0							0	0				0			0	0
0x90	SCU_MCCR1	Res.	Res.	Res.	Res.	Res.	WDTCSEL[2:0]			WDTCDIV[7:0]								Res.	Res.	Res.	Res.	Res.	STCSEL[2:0]			STCDIV[7:0]							
	Reset value						0	0	0	0	0	0	0	0	0	0	1						0	0	0	0	0	0	0	0	0	0	0

Table 35. SCU Register Map Summary (continued)

Offset	Register name	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0x98	SCU_MCCR3	Res.	Res.	Res.	Res.	Res.	TIMER59CSEL[2:0]			TIMER59CDIV[7:0]								Res.	Res.	Res.	Res.	Res.	IMER04CSEL[2:0]			TIMER04CDIV[7:0]							
	Reset value						0	0	0	0	0	0	0	0	0	0	0						0	0	0	0	0	0	0	0	0	0	0	0
0x9C	SCU_MCCR4	Res.	Res.	Res.	Res.	Res.	PGADCSEL[2:0]			PGADCDIV[7:0]								Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.
	Reset value						0	0	0	0	0	0	0	0	0	0	0																
0xA0	SCU_MCCR5	Res.	Res.	Res.	Res.	Res.	PGCDCSEL[2:0]			PGCDCDIV[7:0]								Res.	Res.	Res.	Res.	Res.	PGBDCSEL[2:0]			PGBDCDIV[7:0]							
	Reset value						0	0	0	0	0	0	0	0	0	0	0						0	0	0	0	0	0	0	0	0	0	0	0
0xA4	SCU_MCCR6	Res.	Res.	Res.	Res.	Res.	FRT1CSEL[2:0]			FRT1CDIV[7:0]								Res.	Res.	Res.	Res.	Res.	FRT0CSEL[2:0]			FRT0CDIV[7:0]							
	Reset value						0	0	0	0	0	0	0	0	0	0	0						0	0	0	0	0	0	0	0	0	0	0	0
0xA8	SCU_MCCR7	Res.	Res.	Res.	Res.	Res.	UARTCSEL[2:0]			UARTCDIV[7:0]								Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.
	Reset value						0	0	0	0	0	0	0	0	0	0	1																
0xAC	SCU_MCCR8	Res.	Res.	Res.	Res.	Res.	CANFD1CSE[2:0]			CANFD1CDIV[7:0]								Res.	Res.	Res.	Res.	Res.	CANFD0CSE[2:0]			CANFD0CDIV[7:0]							
	Reset value						0	0	0	0	0	0	0	0	0	0	0						0	0	0	0	0	0	0	0	0	0	0	0
0xF0	SCU_SYSTEN	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	ENS	SYSTEN[7:0]							
	Reset value																								0	0	0	0	0	0	0	0	0

Table 35. SCU Register Map Summary (continued)

Offset	Register name	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0x100	SCU_SPSCR	SPS_KEY[7:0]								Res.	Res.	Res.	Res.	Res.	Res.	Res.	SPSEN	SRAM7_PMODE[1:0]		SRAM6_PMODE[1:0]		SRAM5_PMODE[1:0]		SRAM4_PMODE[1:0]		SRAM3_PMODE[1:0]		SRAM2_PMODE[1:0]		SRAM1_PMODE[1:0]		SRAM0_PMODE[1:0]	
	Reset value	0	0	0	0	0	0	0	0								0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
0x104	SCU_SPSSR	SRAM7_STS[1:0]		SRAM6_STS[1:0]		SRAM5_STS[1:0]		SRAM4_STS[1:0]		SRAM3_STS[1:0]		SRAM2_STS[1:0]		SRAM1_STS[1:0]		SRAM0_STS[1:0]		Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	SPSBUSY	SPSDONE
	Reset value	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0															0	0
0x108	SCU_SPSCR	SPS_KEY[7:0]								Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	SPSDLY[7:0]											
	Reset value	0	0	0	0	0	0	0	0													1	1	1	1	1	1	1	1	1	1	1	1
0x120	SCU_PRER3	Res.	Res.	PWM59	PWM04	Res.	Res.	Res.	Res.	SPI11	SPI10	SPI9	SPI8	SPI7	SPI6	SPI5	SPI4	Res.	Res.	Res.	UART12	UART11	UART10	UART9	UART8	Res.	Res.	Res.	Res.	Res.	Res.	Res.	QSPI
	Reset value			1	1					1	1	1	1	1	1	1	1				1	1	1	1	1								1
0x128	SCU_PER3	Res.	Res.	PWM59	PWM04	Res.	Res.	Res.	Res.	SPI11	SPI10	SPI9	SPI8	SPI7	SPI6	SPI5	SPI4	Res.	Res.	Res.	UART12	UART11	UART10	UART9	UART8	Res.	Res.	Res.	Res.	Res.	Res.	Res.	QSPI
	Reset value			0	0					0	0	0	0	0	0	0	0				0	0	0	0	0								0
0x130	SCU_PCER3	Res.	Res.	PWM59	PWM04	Res.	Res.	Res.	Res.	SPI11	SPI10	SPI9	SPI8	SPI7	SPI6	SPI5	SPI4	Res.	Res.	Res.	UART12	UART11	UART10	UART9	UART8	Res.	Res.	Res.	Res.	Res.	Res.	Res.	QSPI
	Reset value			0	0					0	0	0	0	0	0	0	0				0	0	0	0	0								0

5. Port Control Unit (PCU) and General-Purpose I/O (GPIO)

5.1 Introduction

5.1.1 PCU Introduction

The A34G43x series has a Port Control Unit (PCU) module that controls the external input and output (I/O) ports. By setting up the PCU registers, users can configure the pins' uses, input/output direction, Pull-up/Pull-down resistor, and debouncing for their applications as needed.

5.1.2 GPIO Introduction

Pins except the VDD, GND, and certain specific-purpose pins can be used as General-Purpose Input/Output (GPIO) pins.

The GPIO module controls the general I/O ports. Output pins can be set to generate high- or low-level signals by configuring the corresponding bits of the GPIO control register, while logic input pins can be monitored for their input status in the control registers.

5.2 Main Features

5.2.1 PCU Features

The Port Control Unit (PCU) configures and controls external I/Os as listed below:

- MUX registers define the use of each pin.
  - Input/Output
  - Push-pull output
  - Open-drain output
  - Logic input
  - Analog input
- Internal Pull-up/down resistor and open-drain mode can be configured for each pin.
- Interrupts listed below can be set for each pin:
  - Input level interrupt
  - Input rising-edge interrupt
  - Input falling-edge interrupt
  - Input both-edge interrupt
- Up to seven GPIO interrupts are supported, including GPIOn (n = A to G).
- Each pin can be set for debouncing.

5.2.2 GPIO Features

The GPIO module controls the GPIO ports as listed below:

- Selects the output signal level.
- Enables or disables the pull-up and pull-down resistors for pins.
- External interrupt interfaces

5.3 GPIO Functional Description

Subject to the specific hardware characteristics of each I/O port listed in DC characteristics of Datasheet, each port bit of the General-Purpose I/O (GPIO) ports can be individually configured by software in multiple modes:

- Input floating
- Input pull-up resistor
- Input pull-down resistor
- Analog input
- Output open-drain with pull-up or pull-down capability
- Output push-pull with pull-up or pull-down capability
- Alternate function

During and just after reset, the alternate functions are not active and most of the I/O ports are configured in analog mode. The debug pins are in alternate function pull-up after reset:

- PC0: TCK/SWCLK with pull-up resistor
- PC1: TMS/SWDIO with pull-up resistor
- PC2: TDO/SWO with output low
- PC3: TDI with pull-up resistor
- PC4: nTRST with pull-up resistor
- PC10: nRESET with pull-up resistor
- PC11: nBOOT with pull-up resistor

Following figures (Figure 33, Figure 34 and Figure 35) show block diagrams of GPIO and external interrupt I/O pins, respectively.

Figure 33. PCU and GPIO Block Diagram

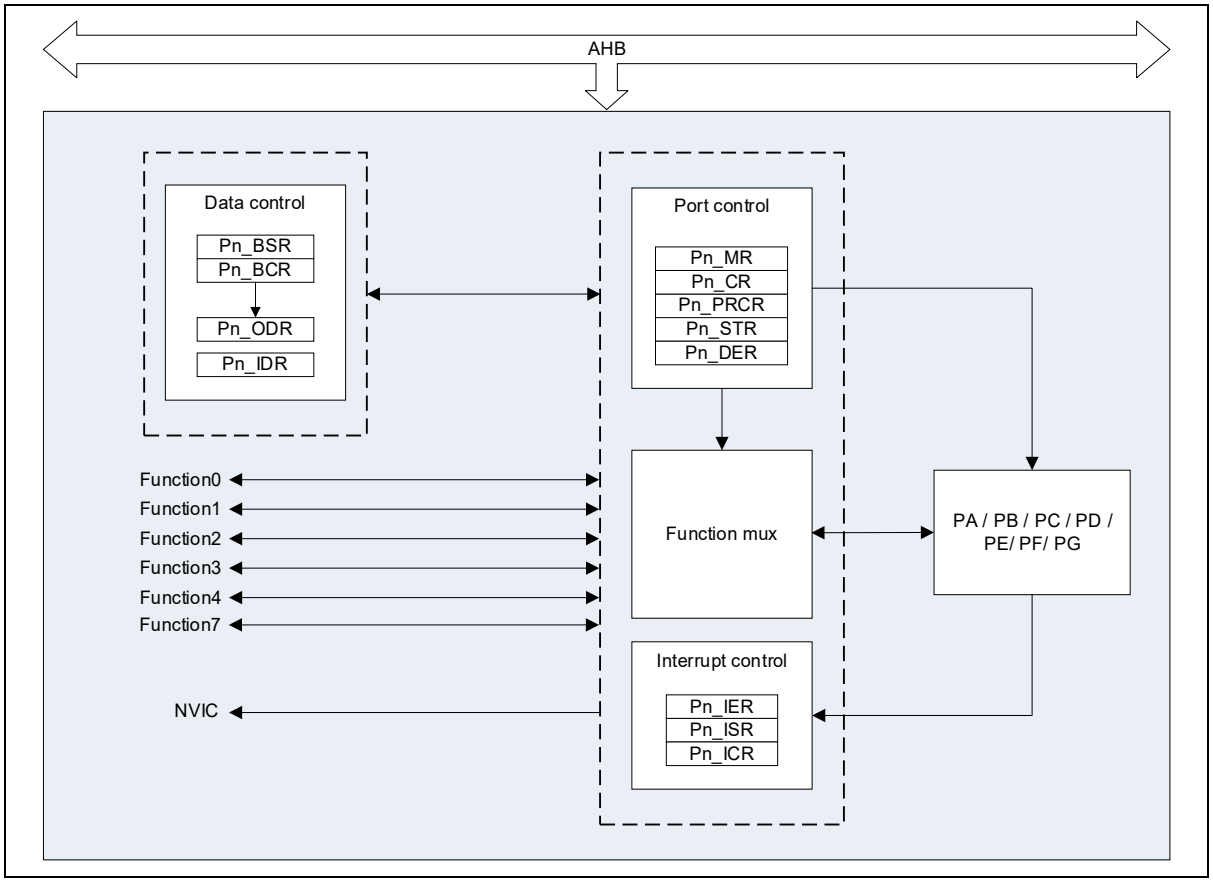


Figure 34. I/O Port Block Diagram (GPIO Pins)

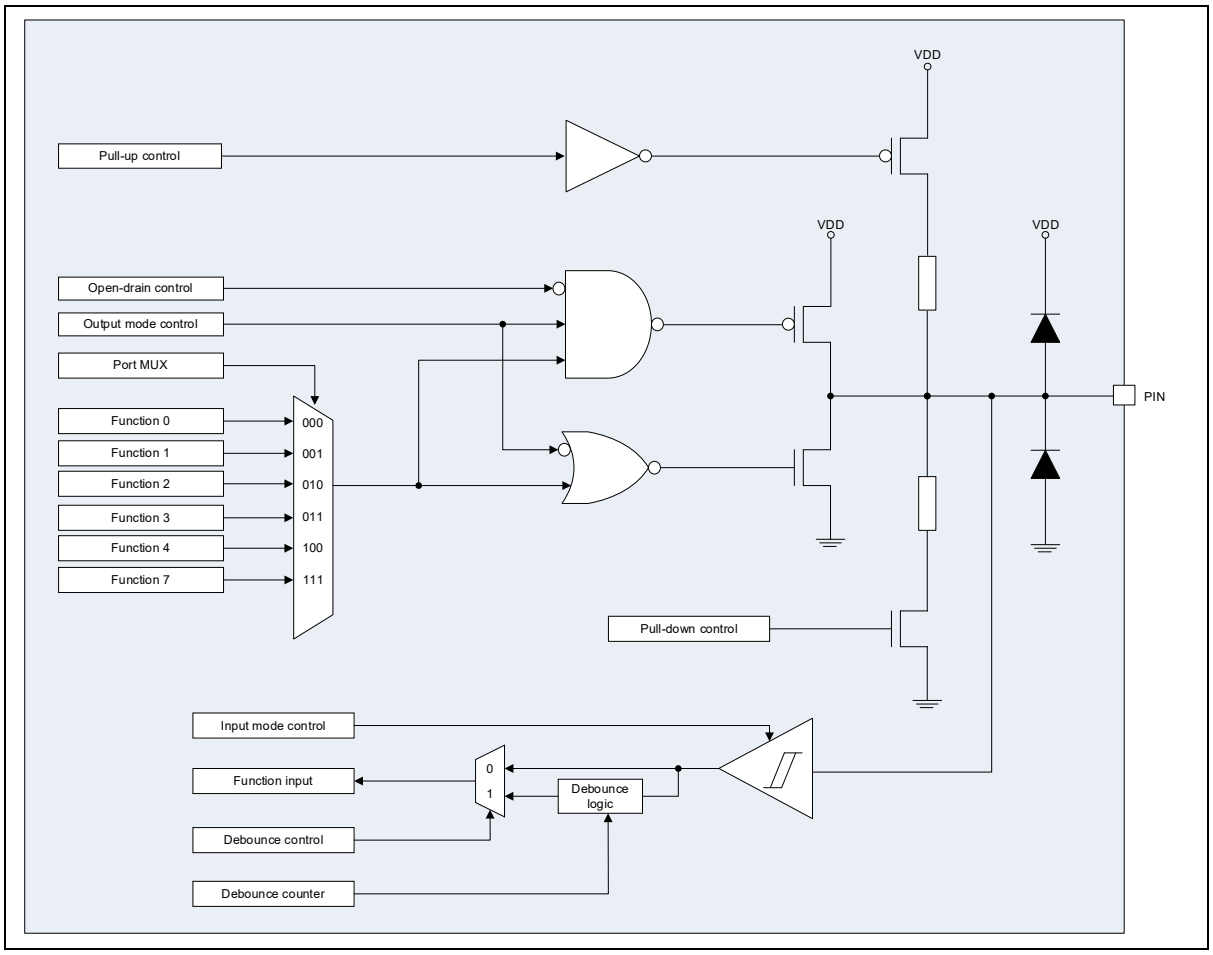


Figure 35. I/O Port Block Diagram (ADC, CMP)

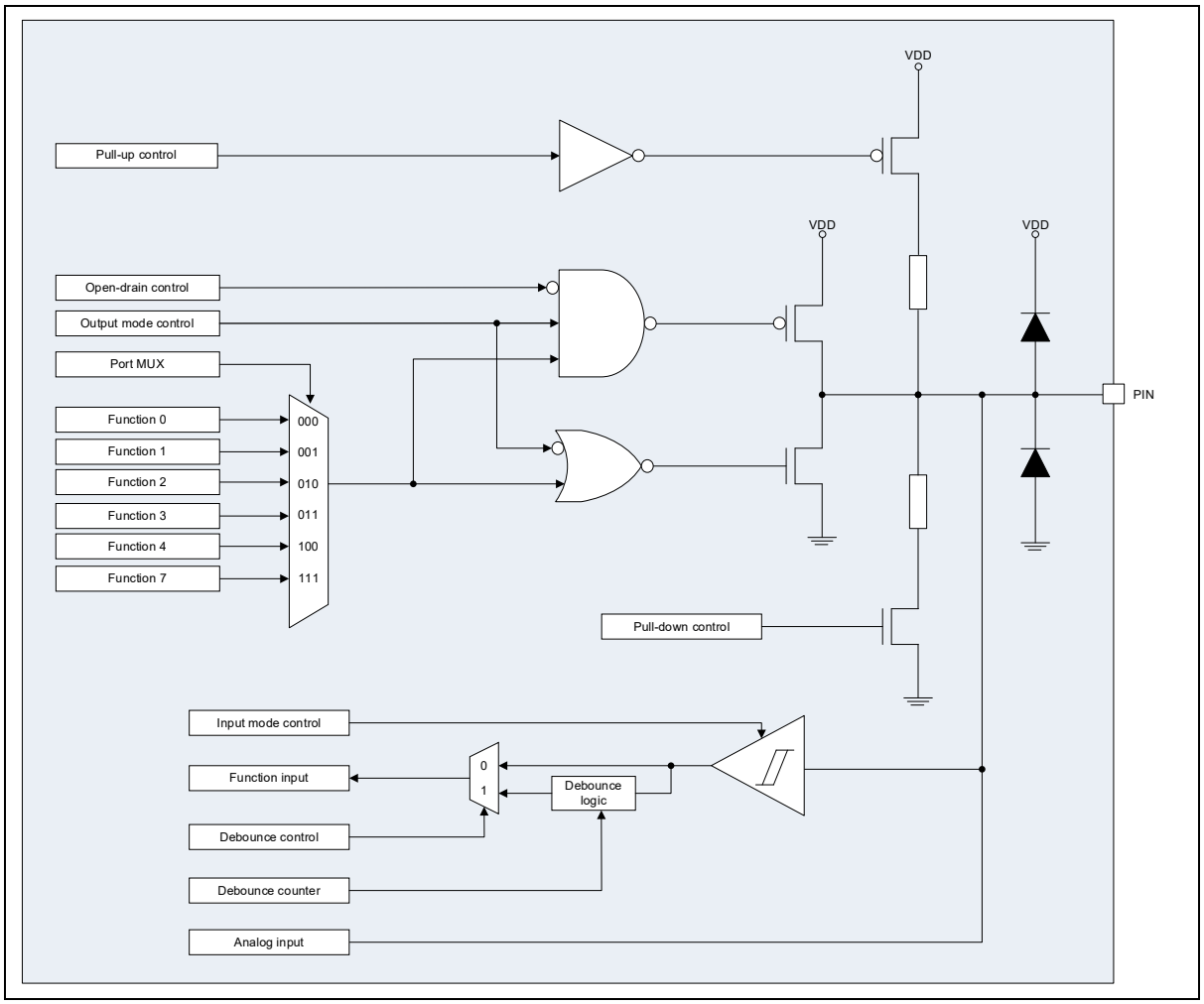
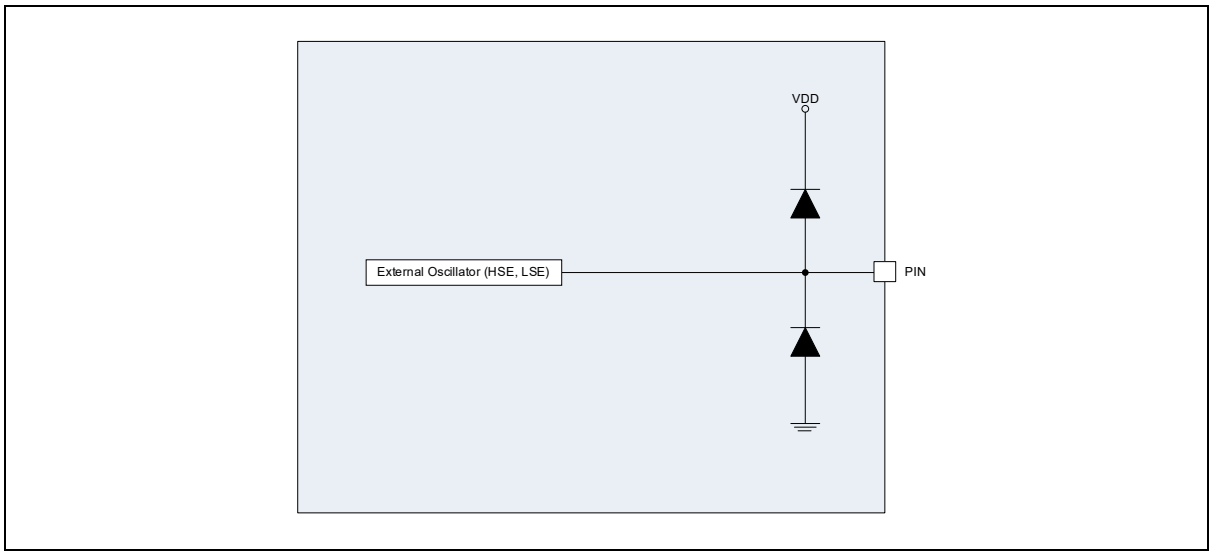


Figure 36. I/O Port Block Diagram (External Oscillator Pins)



NOTE:

1. XIN, XOUT, SXOUT, and SXIN cannot be used for GPIO or other alternate functions. When the HSE and LSE clocks are not in use, the clocks must be disabled by using software.

5.3.1 GPIO Pins and Internal Signals

Table 36 describes the PCU and GPIO internal signal.

Table 36. PCU and GPIO Internal Signal

Signal Name	Signal Type	Description
HCLK	clock	AHB clock

Table 37 describes seven pins assigned for PCU and GPIO module.

Table 37. PCU and GPIO Pins

Pin Name	Type	Description
PAx	IO	PA0 to PA15
PBx	IO	PB0 to PB15
PCx	IO	PC0 to PC11, PC14, PC15
PDx	IO	PD0 to PD15
PEx	IO	PE0 to PE15
PFx	IO	PF0 to PF15
PGx	IO	PG0 to PG10

NOTE:

1. PC12 and PC13 ports are not available. When configuring the PC_MR2, PC_CR, PC_PRCR, PC_DER, PC_STR, PC_IER, PC_ISR, PC_ICR, PC_ODR, PC_IDR, PC_BSR, PC_BCR registers, the 'default values' must be maintained.

5.3.2 I/O Port Control Registers

Each GPIO controls up to sixteen I/Os by using three 32-bit memory mapping control registers such as the Pn_CR and Pn_PRCR registers as shown below:

- The Pn_CR register selects the I/O mode from the Input, Output, and Output open-drain modes.
- The Pn_PRCR register controls the pull-up and pull-down resistors of all ports I/Os.

Some GPIO control registers can be updated with new data only when ‘0x15’ is written to the PORTEN[7:0](1) register and then ‘0x51’ is written again to the same register. After writing ‘0x00’ to the PORTEN[7:0] register, users cannot change the Pn_MR1, Pn_MR2, Pn_CR, and Pn_PRCR register values.

NOTE:

1. Users must be careful when using PORTEN[7:0] control in other contexts such as interrupts. When an interrupt occurs and the PORTEN[7:0] value is changed in interrupt service routine, the PORTEN[7:0] in original process could be unintentionally disabled.

5.3.3 I/O Port Data Registers

Each GPIO has two 16-bit memory mapping data registers that operate as shown below:

- The Pn_ODR register can be read and written when it has output data.
- The Pn_IDR register stores input data entered via the I/O pin. It can only be read.

5.3.4 I/O Data Bitwise Handling

The bit set/clear register (Pn_BSR) and the bit clear register (Pn_BCR) can be set or reset by the application that controls corresponding bit of the Pn_ODR register. These registers support bit banding.

5.3.5 I/O Alternate Function

Each GPIO has four 32-bit memory mapped alternate function(1) registers that operate as shown below:

- The Pn_MR1(2) register selects an alternate function for each port I/O pin (pin 0 to 7).
- The Pn_MR2(2) register selects an alternate function for each port I/O pin (pin 8 to 15).

The direction of the alternate function is automatically determined according to the purpose, whether it is input or output.

NOTES:

1. For the port that supports analog input/output, if a PORT mux on the port is set to GPIO with the corresponding analog module enabled, it doesn't operate correctly because of port conflicts during the operation. Therefore, if users want to use the PORT mux for GPIO, the corresponding analog module must be disabled.
2. The Pn_MR1, Pn_MR2, Pn_CR, and Pn_PRCR registers can be updated with new data only when '0x15' is written to the PORTEN[7:0] register and then '0x51' is written again to the same register. After writing '0x00' to the PORTEN[7:0] register, users cannot change values of these registers.

Table 38 describes the GPIO alternate functions.

Table 38. GPIO Alternate Function

Pin Name	Alternate Function					
	AF0	AF1	AF2	AF3	AF4	AF7
PA0	PA0			SS7		AN0/DAC0O[1]
PA1	PA1			SCK7		AN1/DAC1O[1]
PA2	PA2	RXD7		MISO7		AN2[1]
PA3	PA3	TXD7		MOSI7		AN3[1]
PA4	PA4	SS2	T0IO			AN4/CP0A0[1]
PA5	PA5	SCK2	T1IO			AN5/CP0A1[1]
PA6	PA6	MISO2	T2IO			AN6/CP0A2[1]
PA7	PA7	MOSI2	T3IO			AN7/CP0B[1]
PA8	PA8	RXD2	PWM0A	CANFD1_RX		AN8/CP1A0[1]
PA9	PA9	TXD2	PWM1A	CANFD1_TX		AN9/CP1A1[1]
PA10	PA10	RXD1	PWM2A	CANFD1_STBY		AN10/CP1A2[1]
PA11	PA11	TXD1	PWM3A	SCL1		AN11/CP1B[1]
PA12	PA12	SS0	PWM4A	SDA1		AN12[1]
PA13	PA13	SCK0	PWM9A			AN13[1]
PA14	PA14	MOSI0	PWM0B			AN14[1]
PA15	PA15	MISO0	PWM1B			AN15[1]
PB0	PB0	SS3	PWM2B	MP0UH		[1]
PB1	PB1	SCK3	PWM3B	MP0UL		[1]
PB2	PB2	MOSI3	PWM4B	MP0VH		[1]
PB3	PB3	MISO3	PWM5B	MP0VL		[1]
PB4	PB4		T8IO	MP0WH		[1]
PB5	PB5		T9IO	MP0WL		[1]
PB6	PB6	SCL2		MP0PRT		[1]
PB7	PB7	SDA2		MP0OCP		[1]
PB8	PB8	RXD3		MP1PRT		[1]
PB9	PB9	TXD3	PWM4A	MP1OCP		[1]
PB10	PB10	QSPI_IO3	PWM5A	MP1UH		[1]
PB11	PB11	QSPI_IO2	PWM6A	MP1UL		[1]
PB12	PB12	QSPI_IO1	PWM7A	MP1VH		[1]
PB13	PB13	QSPI_IO0	PWM8A	MP1VL		[1]
PB14	PB14	QSPI_SCK	PWM9A	MP1WH		[1]
PB15	PB15	QSPI_SS	PWM0B	MP1WL		[1]
PC0	PC0	RXD0		TCK/SWCLK[1]		
PC1	PC1	TXD0		TMS/SWDIO[1]		
PC2	PC2			TDO/SWO[1]		
PC3	PC3			TDI[1]		
PC4	PC4		T0IO	nTRST[1]		
PC5	PC5	RXD1	T1IO	CANFD0_RX	CP0O	[1]
PC6	PC6	TXD1	T2IO	CANFD0_TX	CP1O	[1]
PC7	PC7	SCL0	T3IO	CANFD0_STBY	CP2O	[1]
PC8	PC8	SDA0	T4IO		CP3O	[1]

Table 38. GPIO Alternate Function (continued)

Pin Name	Alternate Function					
	AF0	AF1	AF2	AF3	AF4	AF7
PC9	PC9		T8IO	CLKO		(1)
PC10	PC10			nRESET(1)		
PC11	PC11		T9IO	nBOOT(1)		
PC14	PC14	MOSI0		RXD0		
PC15	PC15	MISO0		TXD0		
PD0	PD0	SS1				(1)
PD1	PD1	SCK1				(1)
PD2	PD2	MOSI1		SCL3		(1)
PD3	PD3	MISO1		SDA3		(1)
PD4	PD4	SCL1	PWM5A			AN16(1)
PD5	PD5	SDA1	PWM6A			AN17(1)
PD6	PD6	TXD2	PWM7A			AN18(1)
PD7	PD7	RXD2	PWM8A			AN19(1)
PD8	PD8	RXD12	T6IO	WDTO		(1)
PD9	PD9	TXD12	T7IO	STBYO		(1)
PD10	PD10	QSPI_IO2	T0IO			(1)
PD11	PD11	QSPI_IO1	T1IO			(1)
PD12	PD12	QSPI_SCK	T2IO			(1)
PD13	PD13	QSPI_SS	T3IO			(1)
PD14	PD14	SS0				(1)
PD15	PD15	SCK0				(1)
PE0	PE0					(1)
PE1	PE1			SS8		AN20/CP2A(1)
PE2	PE2			SCK8		AN21/CP2B(1)
PE3	PE3	SCL0	PWM8B	SCK5		(1)
PE4	PE4	SDA0	PWM9B	SS5		(1)
PE5	PE5		T5IO	CANFD0_STBY		(1)
PE6	PE6	RXD6	T5IO	MISO6		(1)
PE7	PE7	TXD6	T6IO	MOSI6		(1)
PE8	PE8		T7IO	SCK6		(1)
PE9	PE9		T8IO	SS6		(1)
PE10	PE10	QSPI_IO3	T9IO			(1)
PE11	PE11	SCL1	T0IO	SS4		(1)
PE12	PE12	SDA1	T1IO	SCK4		(1)
PE13	PE13	TXD4	T2IO	MOSI4		(1)
PE14	PE14	RXD4	T3IO	MISO4		(1)
PE15	PE15					(1)
PF0	PF0		PWM1B	SS9		(1)
PF1	PF1		PWM2B	SCK9		(1)
PF2	PF2	RXD8		MISO8		AN22/CP3A(1)
PF3	PF3	TXD8		MOSI8		AN23/CP3B(1)
PF4	PF4	TXD5	PWM6B	MOSI5		(1)

Table 38. GPIO Alternate Function (continued)

Pin Name	Alternate Function					
	AF0	AF1	AF2	AF3	AF4	AF7
PF5	PF5	RXD5	PWM7B	MISO5		(1)
PF6	PF6	SCL3		CANFD0_TX		(1)
PF7	PF7	SDA3		CANFD0_RX		(1)
PF8	PF8	RXD9	PWM3B	MISO9		(1)
PF9	PF9	TXD9	PWM4B	MOSI9		(1)
PF10	PF10		PWM5B	SS10		(1)
PF11	PF11		PWM6B	SCK10		(1)
PF12	PF12	RXD10	PWM7B	MISO10		(1)
PF13	PF13	TXD10	PWM8B	MOSI10		(1)
PF14	PF14		PWM0A	SS11		(1)
PF15	PF15		PWM1A	SCK11		(1)
PG0	PG0	SS2				(1)
PG1	PG1	SCK2				(1)
PG2	PG2	MOSI2				(1)
PG3	PG3	MISO2				(1)
PG4	PG4					(1)
PG5	PG5	RXD11	PWM2A	MISO11		(1)
PG6	PG6	TXD11	PWM3A	MOSI11		(1)
PG7	PG7					(1)
PG8	PG8	RXD3				(1)
PG9	PG9	TXD3				(1)
PG10	PG10	QSPI_IO0	PWM9B			(1)
XIN						
XOUT						
SXIN						
SXOUT						

NOTES:

1. It means ‘selected pin function after reset condition’.
2. Unused pins must be set to output by software (low output is recommended).
3. PC12 and PC13 ports are not available. When configuring the PC_MR2, PC_CR, PC_PRCR, PC_DER, PC_STR, PC_IER, PC_ISR, PC_ICR, PC_ODR, PC_IDR, PC_BSR, PC_BCR registers, the 'default values' must be maintained.

5.3.6 Interrupt Functionality

The GPIO module has four interrupt sources, and each port can be an interrupt source. To set each GPIO pin as interrupt sources, users must configure the interrupt control register (Pn_ICR) and the interrupt enable register (Pn_IER).

A level-triggered or edge-triggered interrupt can be set for each pin. Once an interrupt occurs, the pin's corresponding bit of the Pn_ISR register is flagged. This flag can be cleared by writing a '1' to the bit.

Table 39. Interrupt Sources and Corresponding Pins in GPIO Module

Interrupt Name	Configurable Pins
GPIOA	Port A[15:0]
GPIOB	Port B[15:0]
GPIOC	Port C[11:0], Port C[15:14]
GPIOD	Port D[15:0]
GPIOE	Port E[15:0]
GPIOF	Port F[15:0]
GPIOG	Port G[10:0]

NOTE:

1. PC12 and PC13 ports are not available. When configuring the PC_MR2, PC_CR, PC_PRCR, PC_DER, PC_STR, PC_IER, PC_ISR, PC_ICR, PC_ODR, PC_IDR, PC_BSR, PC_BCR registers, the 'default values' must be maintained.

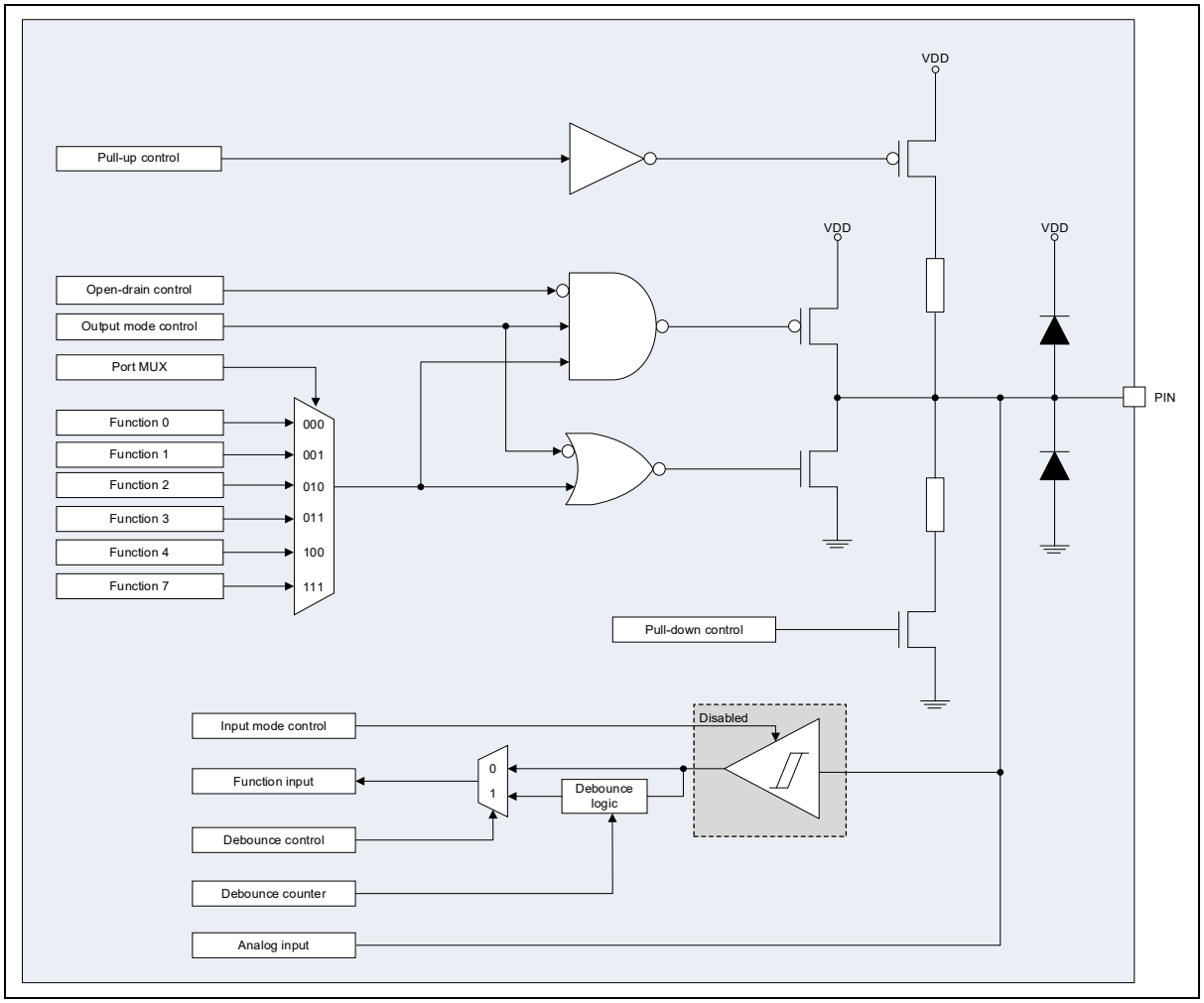
5.3.7 Output Configuration

When the I/O port is programed as output, it features the following:

- The push-pull driver is turned on or off depending on the Pn_ODR register or function output value.
- Each pin's drive strength can be controlled by configuring the Pn_STR register.
- Open-drain mode is available.
- The use of the pull-up and pull-down resistors is determined by the Pn_PRCR register.
- The path of analog signal must be opened for the individual module control.

Figure 37 shows a block diagram of an output port.

Figure 37. Output Port Block Diagram



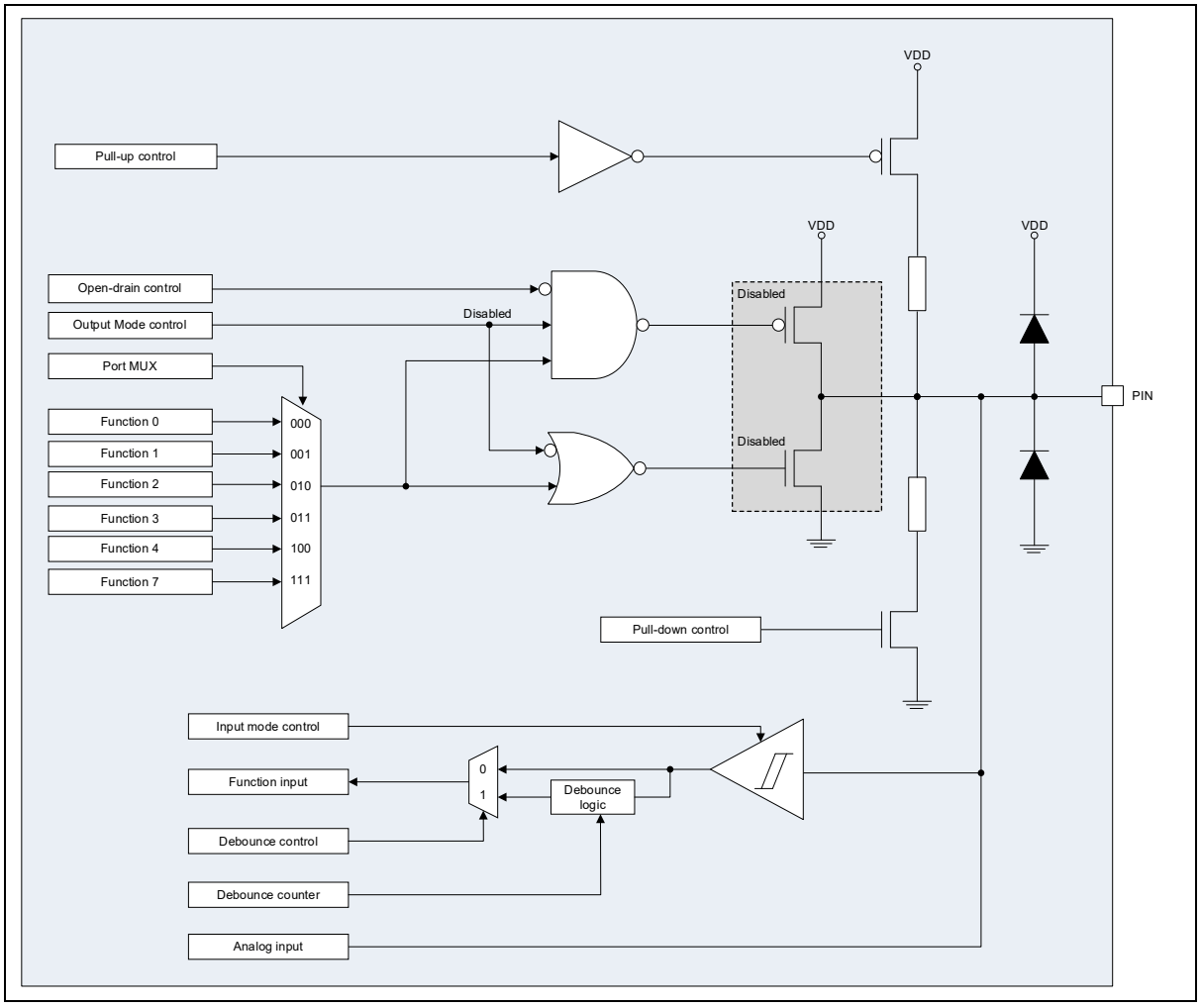
5.3.8 Input Configuration

When the I/O port is programed as input, it features the following:

- All push-pull drivers are turned off, and Input Schmitt Trigger opens its input.
- Input data is stored in the Pn_IDR register or input signal enters toward functions.
- The path of analog signal must be opened for the individual module control.

Figure 38 shows a block diagram of an input port.

Figure 38. Input Port Block Diagram



5.3.9 Alternate Function Configuration

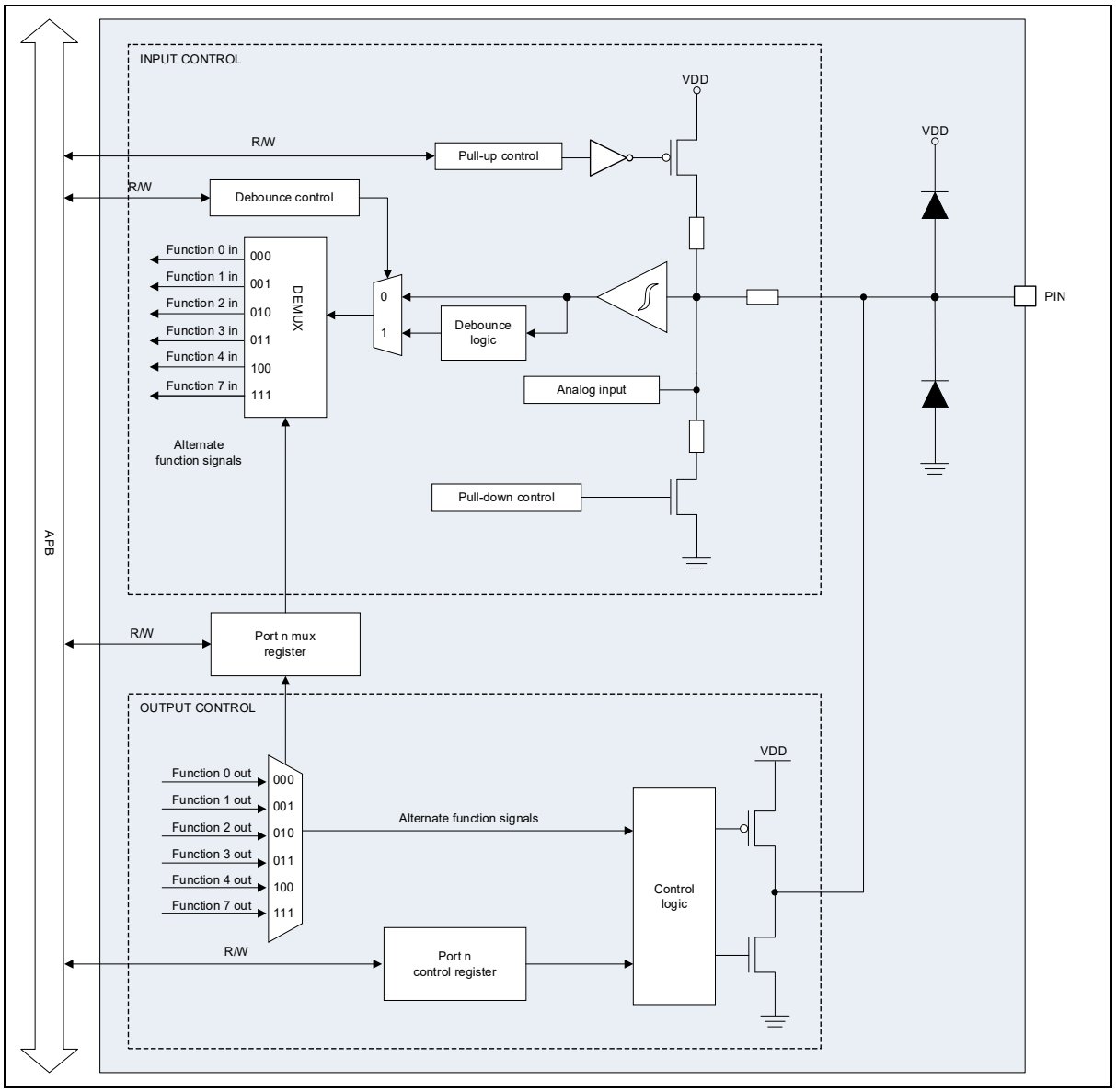
For the A34G43x, all pins except those with specific purposes can be used as GPIO pins, and the GPIO function can be set by the pin mux control register on each port.

If an I/O port's pin is set as an input pin by the pin control register, its output function becomes disabled.

Users can select different functions for each port pin based on the settings of the alternate function selection register. The input data registers capture data entering each I/O pin, as either un-debounced or debounced, at every GPIO clock cycle.

Figure 39 shows a block diagram that describes alternate function selection on a port.

Figure 39. Port Function Block Diagram



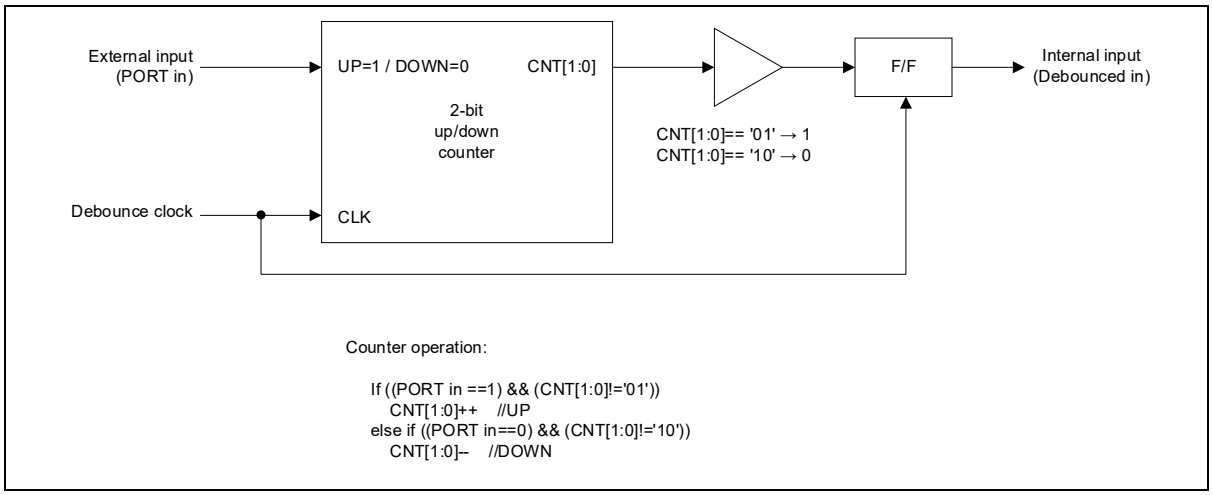
5.3.10 Debouncing Functionality

For the A34G43x, each port is capable of debouncing input signals. Debouncing is to filter out noise that can interfere with the port's data input.

Filtering levels can be adjusted for each 16-pin port, and each individual pin is configured based on whether filtering is enabled or disabled. The debouncing clock used for each port is set in the SCU_MCCR4 and SCU_MCCR5 registers.

Figure 40 shows a block diagram of a simplified debouncing logic.

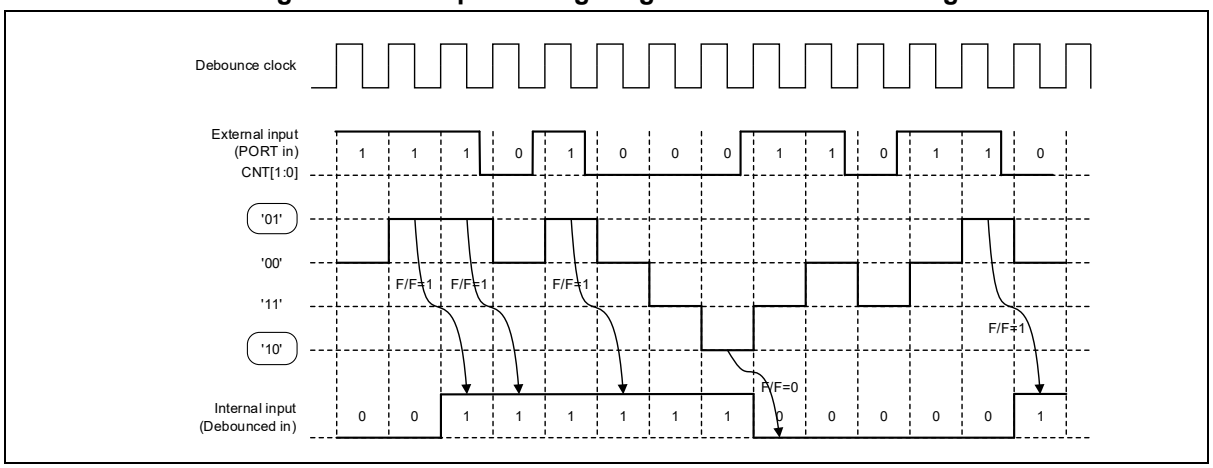
Figure 40. Debouncing Logic Block Diagram



When debouncing is enabled, the debouncing logic is clocked at the frequency division ratio set in the SCU_MCCR4 or SCU_MCCR5 register, depending on which register the port belongs to. A pin recognizes a change in the input if the changed value remains constant for three clock pulses. If the internal count (CNT[1:0]) is set to '01', the internal input value is set to '1'; while if the former is set to '10', the latter is set to '0'. Otherwise, the previous input value is maintained.

Figure 41 shows a timing diagram of a port debouncing.

Figure 41. Example Timing Diagram of Port Debouncing



5.4 GPIO Registers

The base address and register map of PCU are described in the following:

Table 40. Base Address of PCU

Name	Base Address	Description
PA	0x4101_0000	GPIO Port A
PB	0x4101_0100	GPIO Port B
PC	0x4101_0200	GPIO Port C
PD	0x4101_0300	GPIO Port D
PE	0x4101_0400	GPIO Port E
PF	0x4101_0500	GPIO Port F
PG	0x4101_0600	GPIO Port G

Table 41. PCU and GPIO Register Map(1)

Name	Offset	Type	Description	Reset Value	Reference
Pn_MR1	0x0000	RW	Port n MUX1 Select Register	PA: 0x7777_7777 PB: 0x7777_7777 PC: 0x7773_3333 PD: 0x7777_7777 PF: 0x7777_7777 PG: 0x7777_7777	5.4.1 5.4.3
Pn_MR2	0x0004	RW	Port n MUX2 Select Register	PA: 0x7777_7777 PB: 0x7777_7777 PC: 0x7777_3377 PD: 0x7777_7777 PF: 0x7777_7777 PG: 0x7777_7777	5.4.2 5.4.4
Pn_CR	0x0008	RW	Port n Control Register	0xFFFF_FFFF	5.4.5
Pn_PRCR	0x000C	RW	Port n Pull-up/pull-down Control Register	PA: 0x0000_0000 PB: 0x0000_0000 PC: 0x00A0_028A PD: 0x0000_0000 PE: 0x0000_0000 PF: 0x0000_0000 PG: 0x0000_0000	5.4.6
Pn_DER	0x0010	RW	Port n Debouncing Enable Register	0x0000_0000	5.4.8
Pn_STR	0x0014	RW	Port n Strength Configuration Register	0x0000_0000	5.4.9
Pn_IER	0x0020	RW	Port n Interrupt Enable Register	0x0000_0000	5.4.10
Pn_ISR	0x0024	RWC1	Port n Interrupt Status Register	0x0000_0000	5.4.11
Pn_ICR	0x0028	RW	Port n Interrupt Control Register	0x0000_0000	5.4.12
Pn_ODR	0x0030	RW	Port n Output Data Register	0x0000_0000	5.4.13
Pn_IDR	0x0034	RO	Port n Input Data Register	0x0000_0000	5.4.14
Pn_BSR	0x0038	WO	Port n Set/Reset register	0x0000_0000	5.4.15
Pn_BCR	0x003C	WO	Port n Reset Register	0x0000_0000	5.4.16
PCU_RSTDBCR	0x0F0C	RW	Pin Reset Debounce Control Register	0x0000_7801	5.4.17
PCU_PORTEN	0x0FF0	RW	Port Access Enable Register	0x0000_0000	5.4.18

NOTE:

1. n = A, B, C, D, E, F and G.

5.4.1 Pn_MR1: Port n MUX Control Register 1

The Pn_MR1 register selects the mode of each pin (pin 0 to 7) for port n. Before the ports are used, the register must be set correctly; otherwise, their proper operations are not ensured.

PA_MR1=0x4101_0000, PB_MR1=0x4101_0100, PD_MR1=0x4101_0300
PE_MR1=0x4101_0400, PF_MR1=0x4101_0500, PG_MR1=0x4101_0600

31	30 29 28	27	26 25 24	23	22 21 20	19	18 17 16	15	14 13 12	11	10 9 8	7	6 5 4	3	2 1 0
P7_KEY	P7MUX[2:0]	P6_KEY	P6MUX[2:0]	P5_KEY	P5MUX[2:0]	P4_KEY	P4MUX[2:0]	P3_KEY	P3MUX[2:0]	P2_KEY	P2MUX[2:0]	P1_KEY	P1MUX[2:0]	P0_KEY	P0MUX[2:0]
0	111	0	111	0	111	0	111	0	111	0	111	0	111	0	111
WO	RW	WO	RW	WO	RW	WO	RW	WO	RW	WO	RW	WO	RW	WO	RW

4x+3	Px_KEY	Port multiplexer selection bit access key, x=0 to 7 (auto-clear)	
		0	No effect
		1	Access each port multiplexer selection bit
4x+2 4x	PxMUX[2:0]	Port multiplexer selection bit, x=0 to 7	
		000	Alternate function 0 (AF0)
		001	Alternate function 1 (AF1)
		010	Alternate function 2 (AF2)
		011	Alternate function 3 (AF3)
		100	Alternate function 4 (AF4)
		111	Alternate function 7 (AF7)
		Others	Reserved

NOTES:

1. The PxMUX[2:0] (x = 1 to 7) bits of PD0 and PD1 cannot be modified while the LSE is set as the system clock (MCLK).
2. When selecting a port multiplexer for a pin, the corresponding I/O settings are automatically configured for that function. (If the alternate function is selected by the Pn_MR register settings, the I/O is automatically configured as an input or output of that function.)
3. If a port multiplexer selection is set to AF1, AF2, AF3, AF4, or AF7, the port is configured for the alternate function regardless of the Pn_CR register.
4. If a port multiplexer selection is set to AF0, the Pn_CR register configures the port type.
5. Pn_MR1, Pn_MR2, Pn_CR, and Pn_PRCR are registers that can change the value when the PORTEN[7:0] register is activated.
6. When configuring the Pn_MR1 and Pn_MR2 registers, the corresponding Px_KEY for each port must also be set.

5.4.2 Pn_MR2: Port n MUX Control Register 2

The Pn_MR2 register selects the mode of each pin (pin 8 to 15) for port n. Before the ports are used, the register must be set correctly; otherwise, their proper operations are not ensured.

PA_MR2=0x4101_0004, PB_MR2=0x4101_0104, PD_MR2=0x4101_0304
PE_MR2=0x4101_0404, PF_MR2=0x4101_0504, PG_MR2=0x4101_0604

31	30 29 28	27	26 25 24	23	22 21 20	19	18 17 16	15	14 13 12	11	10 9 8	7	6 5 4	3	2 1 0
P15_KEY	P15MUX[2:0]	P14_KEY	P14MUX[2:0]	P13_KEY	P13MUX[2:0]	P12_KEY	P12MUX[2:0]	P11_KEY	P11MUX[2:0]	P10_KEY	P10MUX[2:0]	P9_KEY	P9MUX[2:0]	P8_KEY	P8MUX[2:0]
0	111	0	111	0	111	0	111	0	111	0	111	0	111	0	111
WO	RW	WO	RW	WO	RW	WO	RW	WO	RW	WO	RW	WO	RW	WO	RW

4(x-8)+3	Px_KEY	Port multiplexer selection bit access key, x=8 to 15 (auto-clear)	
		0	No effect
		1	Access each port multiplexer selection bit
4(x-8)+2 4(x-8)	PxMUX[2:0]	Port multiplexer selection bit, x=8 to 15	
		000	Alternate function 0 (AF0)
		001	Alternate function 1 (AF1)
		010	Alternate function 2 (AF2)
		011	Alternate function 3 (AF3)
		100	Alternate function 4 (AF4)
		111	Alternate function 7 (AF7)
		Others	Reserved

NOTES:

1. When selecting a port multiplexer for a pin, the corresponding I/O settings are automatically configured for that function. (If the alternate function is selected by the Pn_MR register settings, the I/O is automatically configured as the input or output of that function.)
2. If a port multiplexer selection is set to AF1, AF2, AF3, AF4, or AF7, the port is configured for the alternate function regardless of the Pn_CR register.
3. If a port multiplexer selection is set to AF0, the Pn_CR register configures the port type.
4. Pn_MR1, Pn_MR2, Pn_CR, and Pn_PRCR are registers that can change the value when PORTEN[7:0] register is activated.
5. When configuring the Pn_MR1 and Pn_MR2 registers, the corresponding Px_KEY for each port must also be set.
6. PC12 and PC13 ports are not available. When configuring PC_MR2, PC_CR, PC_PRCR, PC_DER, PC_STR, PC_IER, PC_ISR, PC_ICR, PC_ODR, PC_IDR, PC_BSR, PC_BCR registers, the 'default values' must be maintained.

5.4.3 PC_MR1: Port C MUX Control Register 1

The PC_MR1 register selects the mode of each pin (pin 0 to 7) for port n. Before the ports are used, the register must be set correctly; otherwise, their proper operations are not ensured.

PC_MR1=0x4101_0200

31	30 29 28	27	26 25 24	23	22 21 20	19	18 17 16	15	14 13 12	11	10 9 8	7	6 5 4	3	2 1 0
P7_KEY	P7MUX[2:0]	P6_KEY	P6MUX[2:0]	P5_KEY	P5MUX[2:0]	P4_KEY	P4MUX[2:0]	P3_KEY	P3MUX[2:0]	P2_KEY	P2MUX[2:0]	P1_KEY	P1MUX[2:0]	P0_KEY	P0MUX[2:0]
0	111	0	111	0	111	0	011	0	011	0	011	0	011	0	011
WO	RW	WO	RW	WO	RW	WO	RW	WO	RW	WO	RW	WO	RW	WO	RW

4x+3	Px_KEY	Port multiplexer selection bit access key, x=0 to 7 (auto-clear)	
		0	No effect
		1	Access each port multiplexer selection bit
4x+2 4x	PxMUX[2:0]	Port multiplexer selection bit, x=0 to 7	
		000	Alternate function 0 (AF0)
		001	Alternate function 1 (AF1)
		010	Alternate function 2 (AF2)
		011	Alternate function 3 (AF3)
		100	Alternate function 4 (AF4)
		111	Alternate function 7 (AF7)
		Others	Reserved

NOTES:

1. When selecting a port multiplexer for a pin, the corresponding I/O settings are automatically configured for that function. (If the alternate function is selected by the PC_MR register settings, the I/O is automatically configured as the input or output of that function.)
2. If a port multiplexer selection is set to AF1, AF2, AF3, AF4, or AF7, the port is configured for the alternate function regardless of the PC_CR register.
3. If a port multiplexer selection is set to AF0, the PC_CR register configures the port type.
4. PC_MR1, PC_MR2, PC_CR, and PC_PRCR are registers that can change the value when the PORTEN[7:0] register is activated.
5. When configuring the PC_MR1 and PC_MR2 registers, the corresponding Px_KEY for each port must also be set.

5.4.4 PC_MR2: Port C MUX Control Register 2

The PC_MR2 register selects the mode of each pin (pin 8 to 15) for port n. Before the ports are used, the register must be set correctly; otherwise, their proper operations are not ensured.

PC_MR2=0x4101_0204

31	30 29 28	27	26 25 24	23 22 21 20 19 18 17 16	15	14 13 12	11	10 9 8	7	6 5 4	3	2 1 0
P15_KEY	P15MUX[2:0]	P14_KEY	P14MUX[2:0]	Reserved	P11_KEY	P11MUX[2:0]	P10_KEY	P10MUX[2:0]	P9_KEY	P9MUX[2:0]	P8_KEY	P8MUX[2:0]
0	111	0	111	0x77	0	011	0	011	0	111	0	111
WO	RW	WO	RW	-	WO	RW	WO	RW	WO	RW	WO	RW

4(x-8)+3	Px_KEY	Port multiplexer selection bit access key, x=8 to 15 (auto-clear)	
		0	No effect
		1	Access each port multiplexer selection bit
4(x-8)+2 4(x-8)	PxMUX[2:0]	Port multiplexer selection bit, x=8 to 15	
		000	Alternate function 0 (AF0)
		001	Alternate function 1 (AF1)
		010	Alternate function 2 (AF2)
		011	Alternate function 3 (AF3)
		100	Alternate function 4 (AF4)
		111	Alternate function 7 (AF7)
		Others	Reserved

NOTES:

1. The PxMUX[2:0] bits of PC12, PC13 cannot be modified while the HSE is set as the system clock (MCLK).
2. When selecting a port multiplexer for a pin, the corresponding I/O settings are automatically configured for that function. (If the alternate function is selected by the PC_MR register settings, the I/O is automatically configured as the input or output of that function.)
3. If a port multiplexer selection is set to AF1, AF2, AF3, AF4, or AF7, the port is configured for the alternate function regardless of the PC_CR register.
4. If a port multiplexer selection is set to AF0, the PC_CR register configures the port type.
5. The PC_MR1, PC_MR2, PC_CR, and PC_PRCR are registers that can change the value when the PORTEN[7:0] register is activated.
6. When configuring the Pn_MR1 and Pn_MR2 registers, the corresponding Px_KEY for each port must also be set.
7. PC12 and PC13 ports are not available. When configuring PC_MR2, PC_CR, PC_PRCR, PC_DER, PC_STR, PC_IER, PC_ISR, PC_ICR, PC_ODR, PC_IDR, PC_BSR, PC_BCR registers, the 'default values' must be maintained.

5.4.5 Pn_CR: Port n Control Register

The Pn_CR register sets the pins in each port as input, open-drain output, or push-pull port pins.

PA_CR=0x4101_0008, PB_CR=0x4101_0108, PC_CR=0x4101_0208, PD_CR=0x4101_0308
PE_CR=0x4101_0408, PF_CR=0x4101_0508, PG_CR=0x4101_0608

31 30	29 28	27 26	25 24	23 22	21 20	19 18	17 16	15 14	13 12	11 10	9 8	7 6	5 4	3 2	1 0
P15[1:0]	P14[1:0]	P13[1:0]	P12[1:0]	P11[1:0]	P10[1:0]	P9[1:0]	P8[1:0]	P7[1:0]	P6[1:0]	P5[1:0]	P4[1:0]	P3[1:0]	P2[1:0]	P1[1:0]	P0[1:0]
11	11	11	11	11	11	11	11	11	11	11	11	11	11	11	11
RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW

2x+1 2x	Px[1:0]	Port control, x = 0 to 15.	
		00	Push-pull output
		01	Open-drain output
		1x	Input

NOTES:

1. If a port multiplexer selection is set to AF0, the Pn_CR register configures the port type.
2. Pn_MR1, Pn_MR2, Pn_CR, and Pn_PRCR are registers that can change the value when PORTEN register is activated.
3. PC12 and PC13 ports are not available. When configuring PC_MR2, PC_CR, PC_PRCR, PC_DER, PC_STR, PC_IER, PC_ISR, PC_ICR, PC_ODR, PC_IDR, PC_BSR, PC_BCR registers, the 'default values' must be maintained.

5.4.6 Pn_PRCR: Port n Pull-Up/Pull-Down Resistor Control Register

All pins of each port include internal pull-up and pull-down resistors that can be determined by the Pn_PRCR register. This resistor can enable or disable the pull-up/ pull-down resistors of each port pin.

PA_PRCR=0x4101_000C, PB_PRCR=0x4101_010C, PD_PRCR=0x4101_030C
PE_PRCR=0x4101_040C, PF_PRCR=0x4101_050C, PG_PRCR=0x4101_060C

31 30	29 28	27 26	25 24	23 22	21 20	19 18	17 16	15 14	13 12	11 10	9 8	7 6	5 4	3 2	1 0
PUE15[1:0]	PUE14[1:0]	PUE13[1:0]	PUE12[1:0]	PUE11[1:0]	PUE10[1:0]	PUE9[1:0]	PUE8[1:0]	PUE7[1:0]	PUE6[1:0]	PUE5[1:0]	PUE4[1:0]	PUE3[1:0]	PUE2[1:0]	PUE1[1:0]	PUE0[1:0]
00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00
RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW

2x+1 2x	PUEx[1:0]	Port pull-up and pull-down control, x = 0 to 15.	
		0x	Disables the pull-up and pull-down resistors.
		10	Enables the pull-up resistor and disables the pull-down resistor.
		11	Disables the pull-up resistor and/ enables the pull-down resistor.

NOTES:

1. Pn_MR1, Pn_MR2, Pn_CR, and Pn_PRCR are registers that can change the value when PORTEN register is activated.
2. PC12 and PC13 ports are not available. When configuring PC_MR2, PC_CR, PC_PRCR, PC_DER, PC_STR, PC_IER, PC_ISR, PC_ICR, PC_ODR, PC_IDR, PC_BSR, PC_BCR registers, the 'default values' must be maintained.

5.4.7 PC_PRCR: Port C Pull-Up/Pull-Down Resistor Control Register

All pins of each port include internal pull-up and pull-down resistors that can be determined by the PC_PRCR register. This resistor can enable or disable the pull-up/ pull-down resistors of each port pin.

PC_PRCR=0x4101_020C

31 30 29 28	27 26 25 24	23 22 21 20	19 18 17 16	15 14 13 12	11 10 9 8	7 6 5 4	3 2 1 0
PUE15[1:0]	PUE14[1:0]	Reserved	PUE11[1:0]	PUE10[1:0]	PUE9[1:0]	PUE8[1:0]	PUE7[1:0]	PUE6[1:0]	PUE5[1:0]	PUE4[1:0]	PUE3[1:0]	PUE2[1:0]	PUE1[1:0]	PUE0[1:0]
00	00	0x0	10	10	00	00	00	00	00	10	10	00	10	10
RW	RW	-	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW

2x+1 2x	PUEx[1:0]	Port pull-up and pull-down control, x = 0 to 15.	
		0x	Disables the pull-up and pull-down resistors.
		10	Enables the pull-up resistor and disables the pull-down resistor.
		11	Disables the pull-up resistor and enables the pull-down resistor.

NOTES:

1. PC_MR1, PC_MR2, PC_CR, and PC_PRCR are registers that can change the value when PORTEN register is activated.
2. PC12 and PC13 ports are not available. When configuring PC_MR2, PC_CR, PC_PRCR, PC_DER, PC_STR, PC_IER, PC_ISR, PC_ICR, PC_ODR, PC_IDR, PC_BSR, PC_BCR registers, the 'default values' must be maintained.

5.4.8 Pn_DER: Port n Debounce Enable Register

All pins in each port have a digital debouncing filter, which can be set in the Pn_DER register.

PA_DER=0x4101_0010, PB_DER=0x4101_0110, PC_DER=0x4101_0210, PD_DER=0x4101_0310
PE_DER=0x4101_0410, PF_DER=0x4101_0510, PG_DER=0x4101_0610

31 30 29 28 27 26 25 24	23 22 21 20 19 18 17 16	15 14 13 12	11 10 9 8	7 6 5 4	3 2 1 0
Reserved	PDE15	PDE14	PDE13	PDE12	PDE11	PDE10	PDE9	PDE8	PDE7	PDE6	PDE5	PDE4	PDE3	PDE2	PDE1	PDE0
-	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
-	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW

x	PDEx	Enable pin debouncing, x = 0 to 15	
		0	Disables the debouncing filter.
		1	Enables the debouncing filter.

NOTES:

1. To use the port debounce function, the debounce clock of SCU_MCCR4 and SCU_MCCR5 must be enabled first. If the debounce function is activated without setting the debounce clock, the port may malfunction.
2. Pn_DER, Pn_STR, Pn_IER, Pn_ISR, Pn_ICR, Pn_ODR, Pn_IDR, Pn_BSR, Pn_BCR and PCU_RSTDBCR registers can change the value of the register without activating PORTEN[7:0].
3. PC12 and PC13 ports are not available. When configuring PC_MR2, PC_CR, PC_PRCR, PC_DER, PC_STR, PC_IER, PC_ISR, PC_ICR, PC_ODR, PC_IDR, PC_BSR, PC_BCR registers, the 'default values' must be maintained.

5.4.9 Pn_STR: Port n Strength Configuration Register

The Pn_STR register configures each pin’s drive strength. These settings affect the port’s speed and current consumption.

PA_STR=0x4101_0014, PB_STR=0x4101_0114, PC_STR=0x4101_0214, PD_STR=0x4101_0314
PE_STR=0x4101_0414, PF_STR=0x4101_0514, PG_STR=0x4101_0614

31 30 29 28 27 26 25 24	23 22 21 20 19 18 17 16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved		PST15	PST14	PST13	PST12	PST11	PST10	PST9	PST8	PST7	PST6	PST5	PST4	PST3	PST2	PST1	PST0
-		0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
-		RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW

x	PSTx	Pin drive strength setting, x = 0 to 15.	
		0	Disables the strength.
		1	Enables the strength.

NOTES:

1. Pn_DER, Pn_STR, Pn_IER, Pn_ISR, Pn_ICR, Pn_ODR, Pn_IDR, Pn_BSR, Pn_BCR and PCU_RSTDBCR registers can change the value of the register without activating PORTEN[7:0]
2. PC12 and PC13 ports are not available. When configuring PC_MR2, PC_CR, PC_PRCR, PC_DER, PC_STR, PC_IER, PC_ISR, PC_ICR, PC_ODR, PC_IDR, PC_BSR, PC_BCR registers, the 'default values' must be maintained.

5.4.10 Pn_IER: Port n Interrupt Enable Register

All pins can be set as an external interrupt source which is either the edge-triggered interrupt or the level-triggered interrupt. The Pn_IER register enables or disables these interrupts for each pin.

PA_IER=0x4101_0020, PB_IER=0x4101_0120, PC_IER=0x4101_0220, PD_IER=0x4101_0320
PE_IER=0x4101_0420, PF_IER=0x4101_0520, PG_IER=0x4101_0620

31 30	29 28	27 26	25 24	23 22	21 20	19 18	17 16	15 14	13 12	11 10	9 8	7 6	5 4	3 2	1 0
PIE15[1:0]	PIE14[1:0]	PIE13[1:0]	PIE12[1:0]	PIE11[1:0]	PIE10[1:0]	PIE9[1:0]	PIE8[1:0]	PIE7[1:0]	PIE6[1:0]	PIE5[1:0]	PIE4[1:0]	PIE3[1:0]	PIE2[1:0]	PIE1[1:0]	PIE0[1:0]
00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00
RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW

2x+1 2x	PIEx[1:0]	The interrupt source to be enabled for the pin, x = 0 to15	
		00	Disable the external interrupt source.
		01	Enables the level-triggered interrupt (Non-pending).
		10	Enables the level-triggered interrupt (Pending).
		11	Enables the edge-triggered interrupt.

NOTES:

1. In non-pending mode, users can check if the current state of the pin is an interrupt state. In high level non-pending interrupt mode, an interrupt signal is generated when the pin state is high.

   The interrupt status flag can be checked with the ISR register. In high level non-pending mode, the interrupt status is automatically cleared. Therefore, when the pin state changes from high to low, the interrupt state is automatically cleared.

2. In the pending mode, you can check whether an interrupt has occurred, not the current state of the pin. In high level pending interrupt mode, an interrupt signal is generated when the pin state is high.

   The interrupt status flag can be checked with the ISR register. In high level pending mode, the interrupt status is not automatically cleared. Therefore, when the state of the pin is changed from high to low, the interrupt will continue to occur unless the user clears the interrupt flag. In the pending mode, once an interrupt occurs, the user must clear the interrupt flag directly because the state is maintained until the user clears the interrupt flag directly.

3. Pn_DER, Pn_STR, Pn_IER, Pn_ISR, Pn_ICR, Pn_ODR, Pn_IDR, Pn_BSR, Pn_BCR and PCU_RSTDBCR registers can change the value of the register without activating PORTEN[7:0].

4. PC12 and PC13 ports are not available. When configuring PC_MR2, PC_CR, PC_PRCR, PC_DER, PC_STR, PC_IER, PC_ISR, PC_ICR, PC_ODR, PC_IDR, PC_BSR, PC_BCR registers, the 'default values' must be maintained.

5.4.11 Pn_ISR: Port n Interrupt Status Register

When an interrupt is received by the microcontroller, the status of the interrupt can be detected in the Pn_ISR register. This register informs what kind of interrupt occurred at each interrupt source pin. To clear the interrupt status register, write a '1' to the corresponding bit.

PA_ISR=0x4101_0024, PB_ISR=0x4101_0124, PC_ISR=0x4101_0224, PD_ISR=0x4101_0324
PE_ISR=0x4101_0424, PF_ISR=0x4101_0524, PG_ISR=0x4101_0624

31 30	29 28	27 26	25 24	23 22	21 20	19 18	17 16	15 14	13 12	11 10	9 8	7 6	5 4	3 2	1 0
PIS15[1:0]	PIS14[1:0]	PIS13[1:0]	PIS12[1:0]	PIS11[1:0]	PIS10[1:0]	PIS9[1:0]	PIS8[1:0]	PIS7[1:0]	PIS6[1:0]	PIS5[1:0]	PIS4[1:0]	PIS3[1:0]	PIS2[1:0]	PIS1[1:0]	PIS0[1:0]
00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00
RWC1	RWC1	RWC1	RWC1	RWC1	RWC1	RWC1	RWC1	RWC1	RWC1	RWC1	RWC1	RWC1	RWC1	RWC1	RWC1

2x+1 2x	PISx[1:0]	Pin interrupt status, x = 0 to 15
	00	No interrupt event has occurred.
	01	The low-level interrupt or falling-edge interrupt event has occurred.
	10	The high-level interrupt or rising-edge interrupt event has occurred.
	11	Both the rising- and falling-edge interrupt events have been detected in edge-triggered interrupt mode. Level-triggered interrupt mode is not supported.

NOTES:

1. In rising and falling edge interrupt modes, the pin state becomes '10' when the rising edge occurs, and the pin state becomes '01' when the falling edge occurs.
2. Pn_DER, Pn_STR, Pn_IER, Pn_ISR, Pn_ICR, Pn_ODR, Pn_IDR, Pn_BSR, Pn_BCR and PCU_RSTDBCR registers can change the value of the register without activating PORTEN[7:0].
3. PC12 and PC13 ports are not available. When configuring PC_MR2, PC_CR, PC_PRCR, PC_DER, PC_STR, PC_IER, PC_ISR, PC_ICR, PC_ODR, PC_IDR, PC_BSR, PC_BCR registers, the 'default values' must be maintained.

5.4.12 Pn_ICR: Port n Interrupt Control Register

The Pn_ICR register controls each pin's interrupt modes that define interrupt-triggering signals.

PA_ICR=0x4101_0028, PB_ICR=0x4101_0128, PC_ICR=0x4101_0228, PD_ICR=0x4101_0328
PE_ICR=0x4101_0428, PF_ICR=0x4101_0528, PG_ICR=0x4101_0628

31 30	29 28	27 26	25 24	23 22	21 20	19 18	17 16	15 14	13 12	11 10	9 8	7 6	5 4	3 2	1 0
PIC15[1:0]	PIC14[1:0]	PIC13[1:0]	PIC12[1:0]	PIC11[1:0]	PIC10[1:0]	PIC9[1:0]	PIC8[1:0]	PIC7[1:0]	PIC6[1:0]	PIC5[1:0]	PIC4[1:0]	PIC3[1:0]	PIC2[1:0]	PIC1[1:0]	PIC0[1:0]
00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00
RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW

2x+1 2x	PICx[1:0]	Pin interrupt mode, x = 0 to 15	
		00	Disable the external interrupt source.
		01	Enables low-level or falling-edge interrupt mode.
		10	Enables high-level or rising-edge interrupt mode.
		11	Enables both rising- and falling-edge interrupt modes. Level-triggered interrupt mode is not supported.

NOTE:

1. Pn_DER, Pn_STR, Pn_IER, Pn_ISR, Pn_ICR, Pn_ODR, Pn_IDR, Pn_BSR, Pn_BCR and PCU_RSTDBCR registers can change the value of the register without activating PORTEN[7:0].
2. PC12 and PC13 ports are not available. When configuring PC_MR2, PC_CR, PC_PRCR, PC_DER, PC_STR, PC_IER, PC_ISR, PC_ICR, PC_ODR, PC_IDR, PC_BSR, PC_BCR registers, the 'default values' must be maintained.

5.4.13 Pn_ODR: Port n Output Data Register

The Pn_ODR register defines the output data at each pin when the port is used as GPIO mode. The output level at each pin is decided by this register's settings.

PA_ODR=0x4101_0030, PB_ODR=0x4101_0130, PC_ODR=0x4101_0230, PD_ODR=0x4101_0330
PE_ODR=0x4101_0430, PF_ODR=0x4101_0530, PG_ODR=0x4101_0630

31 30 29 28 27 26 25 24	23 22 21 20 19 18 17 16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved		POD15	POD14	POD13	POD12	POD11	POD10	POD9	POD8	POD7	POD6	POD5	POD4	POD3	POD2	POD1	POD0
-		0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
-		RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW

15 0	PODx	The pin's output data configuration, x = 0 to 15	
		0	Outputs a '0' if the port pin is in output mode.
		1	Outputs a '1' if the port pin is in output mode.

NOTES:

1. Pn_DER, Pn_STR, Pn_IER, Pn_ISR, Pn_ICR, Pn_ODR, Pn_IDR, Pn_BSR, Pn_BCR and PCU_RSTDBCR registers can change the value of the register without activating PORTEN[7:0].
2. PC12 and PC13 ports are not available. When configuring PC_MR2, PC_CR, PC_PRCR, PC_DER, PC_STR, PC_IER, PC_ISR, PC_ICR, PC_ODR, PC_IDR, PC_BSR, PC_BCR registers, the 'default values' must be maintained.

5.4.14 Pn_IDR: Port n Input Data Register

The Pn_IDR register allows each pin level of the port to be read. This register's value represents the input at each pin in logic input mode even though the pin is currently in a different mode, except for in analog mode. If a pin is set to serve a special function, such as an ADC or clock Input/Output, the register bit always reads '1'. If a level interrupt has been enabled for a port's pin by the Pn_ICR register, the pin's input level can be checked by reading the Pn_IDR register.

PA_IDR=0x4101_0034, PB_IDR=0x4101_0134, PC_IDR=0x4101_0234, PD_IDR=0x4101_0334
PE_IDR=0x4101_0434, PF_IDR=0x4101_0534, PG_IDR=0x4101_0634

31 30 29 28 27 26 25 24	23 22 21 20 19 18 17 16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved		PID15	PID14	PID13	PID12	PID11	PID10	PID9	PID8	PID7	PID6	PID5	PID4	PID3	PID2	PID1	PID0
-		0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
-		RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO

15 0	PIDx	The pin's input data (represents the port's status), x = 0 to 15	
		0	The current pin input is '0'.
		1	The current pin input is '1'.

NOTES:

1. If an external pin of the microcontroller is used as an input pin while it is floating, it will fall in an 'unknown' state, where the input signal level is identified as either '0' or '1', regardless of the input value.
2. If the microcontroller is programmed to perform a certain function based on the level identified by the Pn_IDR register in this floating state, this is highly likely to cause malfunctions. Therefore, it is recommended to pull up or down the pins that receive external input signals, depending on their uses.
3. Pn_DER, Pn_STR, Pn_IER, Pn_ISR, Pn_ICR, Pn_ODR, Pn_IDR, Pn_BSR, Pn_BCR and PCU_RSTDBCR registers can change the value of the register without activating PORTEN[7:0].
4. PC12 and PC13 ports are not available. When configuring PC_MR2, PC_CR, PC_PRCR, PC_DER, PC_STR, PC_IER, PC_ISR, PC_ICR, PC_ODR, PC_IDR, PC_BSR, PC_BCR registers, the 'default values' must be maintained.

5.4.15 Pn_BSR: Port n Set/Reset Register

The Pn_BSR register sets or clears the level status flag of each pin of a Port n. The values written in this register are output to the Pn_ODR register. If values of the BCDx and BSDx bits become 1 at the same time, this register sets the corresponding pin level status to '1'.

PA_BSR=0x4101_0038, PB_BSR=0x4101_0138, PC_BSR=0x4101_0238, PD_BSR=0x4101_0338
PE_BSR=0x4101_0438, PF_BSR=0x4101_0538, PG_BSR=0x4101_0638

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
BCD15	BCD14	BCD13	BCD12	BCD11	BCD10	BCD9	BCD8	BCD7	BCD6	BCD5	BCD4	BCD3	BCD2	BCD1	BCD0	BSD15	BSD14	BSD13	BSD12	BSD11	BSD10	BSD9	BSD8	BSD7	BSD6	BSD5	BSD4	BSD3	BSD2	BSD1	BSD0
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
WO	WO	WO	WO	WO	WO	WO	WO	WO	WO	WO	WO	WO	WO	WO	WO	WO	WO	WO	WO	WO	WO	WO	WO	WO	WO	WO	WO	WO	WO	WO	WO

X+16	BCDx	Pin clear bit, x = 16 to 31	
		0	No change
		1	Sets the bit if the corresponding bit is '1'.
X	BSDx	Pin set bit, x = 0 to 15	
		0	No change
		1	Sets the bit if the corresponding bit is '1'.

NOTES:

1. Pn_DER, Pn_STR, Pn_IER, Pn_ISR, Pn_ICR, Pn_ODR, Pn_IDR, Pn_BSR, Pn_BCR and PCU_RSTDBCR registers can change the value of the register without activating PORTEN[7:0].
2. PC12 and PC13 ports are not available. When configuring PC_MR2, PC_CR, PC_PRCR, PC_DER, PC_STR, PC_IER, PC_ISR, PC_ICR, PC_ODR, PC_IDR, PC_BSR, PC_BCR registers, the 'default values' must be maintained.

5.4.16 Pn_BCR: Port n Reset Register

The Pn_BCR register clears the level status flag of each pin of a Port n. Setting a bit to '1' clears the corresponding bit in the Pn_ODR register.

PA_BCR = 0x4101_003C, PB_BCR=0x4101_013C, PC_BCR=0x4101_023C, PD_BCR=0x4101_033C
PE_BCR=0x4101_043C, PF_BCR=0x4101_053C, PG_BCR=0x4101_063C

31 30 29 28 27 26 25 24	23 22 21 20 19 18 17 16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved		BCD15	BCD14	BCD13	BCD12	BCD11	BCD10	BCD9	BCD8	BCD7	BCD6	BCD5	BCD4	BCD3	BCD2	BCD1	BCD0
-		0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
-		WO	WO	WO	WO	WO	WO	WO	WO	WO	WO	WO	WO	WO	WO	WO	WO

X	BCDx	Pin Clear bit, x = 0 to 15	
		0	Causes no changes.
		1	Resets the corresponding pin bit.

NOTES:

1. Pn_DER, Pn_STR, Pn_IER, Pn_ISR, Pn_ICR, Pn_ODR, Pn_IDR, Pn_BSR, Pn_BCR and PCU_RSTDBCR registers can change the value of the register without activating PORTEN[7:0].
2. PC12 and PC13 ports are not available. When configuring PC_MR2, PC_CR, PC_PRCR, PC_DER, PC_STR, PC_IER, PC_ISR, PC_ICR, PC_ODR, PC_IDR, PC_BSR, PC_BCR registers, the 'default values' must be maintained.

5.4.17 PCU_RSTDBCR: Pin Reset Debounce Control Register

The PCU_RSTDBCR register can be configured to set the debounce of the reset pin.

PCU_RSTDBCR=0x4101_0F0C

31 30 29 28 27 26 25 24	23 22 21 20 19 18 17 16	15 14 13 12 11 10 9 8	7 6 5	4	3 2 1	0
WTIDKEY[15:0]		CLK_CNT[7:0]	Reserved	CLK_SEL	Reserved	EN
0x0000		0x78	-	0	-	1
WO		RW	-	RW	-	RW

31 16	WTIDKEY	Write Identification Key On Writes, write 0x0514 to these bits. Otherwise, the writing is ignored.
15 8	CLK_CNT[7:0]	Noise-canceller delay option for selected debounce clock N N clock checking for debounce by selected debounce clock Default filter is set '7.5 us' with HSI (16 MHz) debounce clock
4	CLK_SEL	Debounce clock selection 0 HSI (16 MHz) is debounce clock 1 LSI (500 kHz) is debounce clock
0	EN	Pin reset debounce enable 0 Disables 1 Enables

NOTES:

1. If user want to operate pin reset debounce, user must enable LSI (500kHz) or HSI (16MHz). Because pin reset debounce use LSI or HSI as debounce clock source.
2. Pn_DER, Pn_STR, Pn_IER, Pn_ISR, Pn_ICR, Pn_ODR, Pn_IDR, Pn_BSR, Pn_BCR and PCU_RSTDBCR registers can change the value of the register without activating PORTEN[7:0].

5.4.18 PCU_PORTEN: Port Access Enable Register

The PORTEN register allows to change the values in all PCU registers.

PORTEN=0x4101_0FF0

31 30 29 28 27 26 25 24	23 22 21 20 19 18 17 16	15 14 13 12 11 10 9 8	7 6 5 4 3 2 1 0
Reserved		ENS	PORTEN[7:0]
-		0	0x00
-		RO	RW

8	ENS	Pin Clear bit, x = 0 to 15
		0 Causes no changes.
		1 Resets the corresponding pin bit.
7 0	PORTEN[7:0]	PCU and GPIO register access enable
		Writing ‘0x15’ and then ‘0x51’ to the bit field enables writing new values to GPIO registers. After this, write a different value to this bit field to protect the Pn_MR1, Pn_MR2, Pn_CR, and Pn_PRCR registers against being updated with new values.

NOTE:

1. Some GPIO control registers can be updated with new data only when ‘0x15’ is written to the PORTEN[7:0] and then ‘0x51’ is written again to the same register. After writing ‘0x00’ to the PORTEN[7:0], users cannot change values. The corresponding registers are Pn_MR1, Pn_MR2, Pn_CR, and Pn_PRCR.

5.4.19 PCU and GPIO Register Map Summary

Table 42. PCU and GPIO Register Map Summary

Offset	Register name	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0x00	Pn_MR1	P7_KEY	P7MUX[2:0]			P6_KEY	P6MUX[2:0]			P5_KEY	P5MUX[2:0]			P4_KEY	P4MUX[2:0]			P3_KEY	P3MUX[2:0]			P2_KEY	P2MUX[2:0]			P1_KEY	P1MUX[2:0]			P0_KEY	P0MUX[2:0]		
	Reset value	0	1	1	1	0	1	1	1	0	1	1	1	0	1	1	1	0	1	1	1	0	1	1	1	0	1	1	1	0	1	1	1
0x04	Pn_MR2	P15_KEY	P15MUX[2:0]			P14_KEY	P14MUX[2:0]			P713_KEY	P13MUX[2:0]			P12_KEY	P12MUX[2:0]			P11_KEY	P11MUX[2:0]			P10_KEY	P10MUX[2:0]			P9_KEY	P9MUX[2:0]			P8_KEY	P8MUX[2:0]		
	Reset value	0	1	1	1	0	1	1	1	0	1	1	1	0	1	1	1	0	1	1	1	0	1	1	1	0	1	1	1	0	1	1	1
0x200	PC_MR1	P7_KEY	P7MUX[2:0]			P6_KEY	P6MUX[2:0]			P5_KEY	P5MUX[2:0]			P4_KEY	P4MUX[2:0]			P3_KEY	P3MUX[2:0]			P2_KEY	P2MUX[2:0]			P1_KEY	P1MUX[2:0]			P0_KEY	P0MUX[2:0]		
	Reset value	0	1	1	1	0	1	1	1	0	1	1	1	0	0	1	1	0	0	1	1	0	0	1	1	0	0	1	1	0	0	1	1
0x204	PC_MR2	P15_KEY	P15MUX[2:0]			P14_KEY	P14MUX[2:0]			Res	Res	Res	Res	Res	Res	Res	Res	P11_KEY	P11MUX[2:0]			P10_KEY	P10MUX[2:0]			P9_KEY	P9MUX[2:0]			P8_KEY	P8MUX[2:0]		
	Reset value	0	1	1	1	0	1	1	1									0	0	1	1	0	0	1	1	0	1	1	1	0	1	1	1
0x08	Pn_CR	P15[1:0]		P14[1:0]		P13[1:0]		P12[1:0]		P11[1:0]		P10[1:0]		P9[1:0]		P8[1:0]		P7[1:0]		P6[1:0]		P5[1:0]		P4[1:0]		P3[1:0]		P2[1:0]		P1[1:0]		P0[1:0]	
	Reset value	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1
0x0C	Pn_PRCR	PUE15[1:0]		PUE14[1:0]		PUE13[1:0]		PUE12[1:0]		PUE11[1:0]		PUE10[1:0]		PUE9[1:0]		PUE8[1:0]		PUE7[1:0]		PUE6[1:0]		PUE5[1:0]		PUE4[1:0]		PUE3[1:0]		PUE2[1:0]		PUE1[1:0]		PUE0[1:0]	
	Reset value	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
0x20C	PC_PRCR	PUE15[1:0]		PUE14[1:0]		Res		Res		PUE11[1:0]		PUE10[1:0]		PUE9[1:0]		PUE8[1:0]		PUE7[1:0]		PUE6[1:0]		PUE5[1:0]		PUE4[1:0]		PUE3[1:0]		PUE2[1:0]		PUE1[1:0]		PUE0[1:0]	
	Reset value	0	0	0	0					1	0	1	0	0	0	0	0	0	0	0	0	0	0	1	0	1	0	0	0	1	0	1	0
0x10	Pn_DER	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	PDE15	PDE14	PDE13	PDE12	PDE11	PDE10	PDE9	PDE8	PDE7	PDE6	PDE5	PDE4	PDE3	PDE2	PDE1	PDE0
	Reset value																	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
0x14	Pn_STR	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	PST15	PST14	PST13	PST12	PST11	PST10	PST9	PST8	PST7	PST6	PST5	PST4	PST3	PST2	PST1	PST0
	Reset value																	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
0x20	Pn_IER	PIE15[1:0]		PIE14[1:0]		PIE13[1:0]		PIE12[1:0]		PIE11[1:0]		PIE10[1:0]		PIE9[1:0]		PIE8[1:0]		PIE7[1:0]		PIE6[1:0]		PIE5[1:0]		PIE4[1:0]		PIE3[1:0]		PIE2[1:0]		PIE1[1:0]		PIE0[1:0]	
	Reset value	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
0x24	Pn_ISR	PIS15[1:0]		PIS14[1:0]		PIS13[1:0]		PIS12[1:0]		PIS11[1:0]		PIS10[1:0]		PIS9[1:0]		PIS8[1:0]		PIS7[1:0]		PIS6[1:0]		PIS5[1:0]		PIS4[1:0]		PIS3[1:0]		PIS2[1:0]		PIS1[1:0]		PIS0[1:0]	
	Reset value	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Table 42. PCU and GPIO Register Map Summary (continued)

Offset	Register name	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0x28	Pn_ICR	PIC15[1:0]		PIC14[1:0]		PIC13[1:0]		PIC12[1:0]		PIC11[1:0]		PIC10[1:0]		PIC9[1:0]		PIC8[1:0]		PIC7[1:0]		PIC6[1:0]		PIC5[1:0]		PIC4[1:0]		PIC3[1:0]		PIC2[1:0]		PIC1[1:0]		PIC0[1:0]	
	Reset value	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
0x30	Pn_ODR	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	POD[15:0]															
	Reset value																	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
0x34	Pn_IDR	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	PID[15:0]															
	Reset value																	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
0x38	Pn_BSR	BCD15	BCD14	BCD13	BCD12	BCD11	BCD10	BCD9	BCD8	BCD7	BCD6	BCD5	BCD4	BCD3	BCD2	BCD1	BCD0	BSD15	BSD14	BSD13	BSD12	BSD11	BSD10	BSD9	BSD8	BSD7	BSD6	BSD5	BSD4	BSD3	BSD2	BSD1	BSD0
	Reset value	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
0x3C	Pn_BCR	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	BCD15	BCD14	BCD13	BCD12	BCD11	BCD10	BCD9	BCD8	BCD7	BCD6	BCD5	BCD4	BCD3	BCD2	BCD1	BCD0
	Reset value																	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
0x0F0C	PCU_RSTDBCR	WTIDKEY[15:0]																CLK_CNT[7:0]								Res.	Res.	Res.	CLK_SEL	Res.	Res.	Res.	EN
	Reset value	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	1	1	1	0	0	0				0				1
0x0FF0	PORTEN	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	ENS	PORTEN[7:0]							
	Reset value																								0	0	0	0	0	0	0	0	0

6. Embedded Flash Memory

6.1 Introduction

The flash memory controller is an interface controller for embedded flash memory. It manages data stored on flash memory.

The flash memory controller of A34G43x supports the following:

- Code data and instruction cache in code flash area
- Erase, read, write functions in code flash area
- Read protection function
- Write and erase protection function
- Data flash memory for programming user data

6.2 Embedded Flash Memory Main Features

The flash memory of A34G43x has the features listed below:

- Code flash memory:
  - Flash memory size
    - A34G43A (2,048 KB)
    - A34G430 (1,024 KB)
    - Dual bank code flash memory (bank0, bank1) for OTA (Over-The-Air programming)
    - Bank swap with RWW (Read-While-Write)
  - Program unit: 2 words (8 bytes)
  - Erase unit:
    - Page (1 KB)
    - Sector (4 KB)
    - Bulk (full-chip)
  - Zero-wait (when the clock is lower than 33 MHz), 1- to 15-wait, and cache (flash acceleration) access
  - Write protection
  - Read protection
  - Code data and instruction cache
  - Built-in 16-bit specific polynomial for code flash memory
    - ECC (Error Correct Code)
  - Self-program function for the code flash memory
    - Supports updating data in a code flash memory region while executing user program in the code flash memory area.
- Data flash memory:
  - Flash memory size: 32 KB
  - Program unit: 1 word (4 bytes)
  - Erase unit:
    - Page (512 bytes)
    - Sector (2 KB)
    - Bulk (full-chip)

- Zero wait (when the clock is lower than 33 MHz), 1- to 15-wait
- Write protection
- Built-in 16-bit specific polynomial for data flash memory
- ECC (Error Correct Code)
- Self-program function for data flash memory
  - Supports updating data in the data flash memory region while executing user program in the code flash memory area.

6.3 Flash Functional Description

6.3.1 Flash Memory Organization

The flash memory consists of code and data flash memory. Code flash memory stores application codes or constant data, and data flash memory stores parameters, log data, etc.

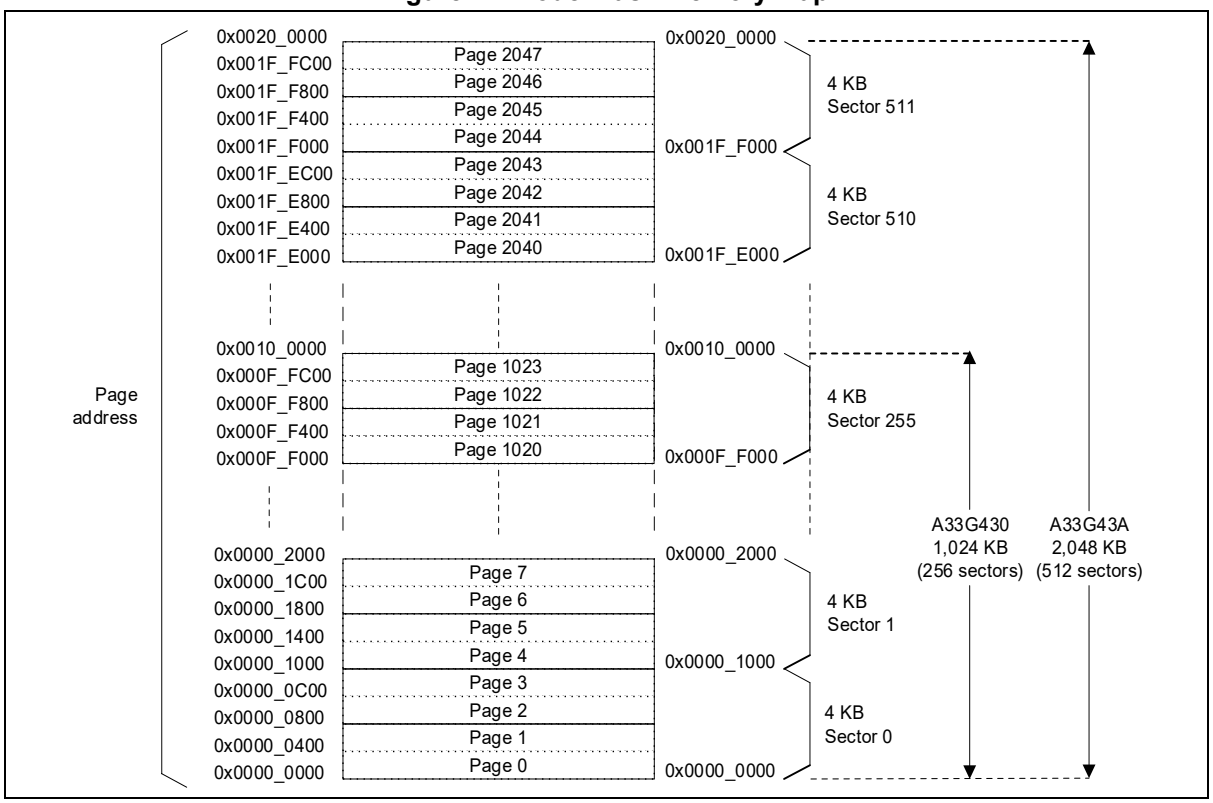
6.3.1.1 Code Flash Memory

Code flash memory has the features as shown below:

Table 43. Code Flash Memory Controller Features

Item	A34G430	A34G43A
Size	1,024 KB	2,048 KB
Start address	0x0000_0000	0x0000_0000
End address	0x000F_FFFF	0x001F_FFFF
Page size	1 KB	
Total page count	1024 pages	2048 pages
PGM unit	8 bytes (2 word)	
Erase unit	1 KB 4 KB Bulk (full-chip)	

Figure 42. Code Flash Memory Map



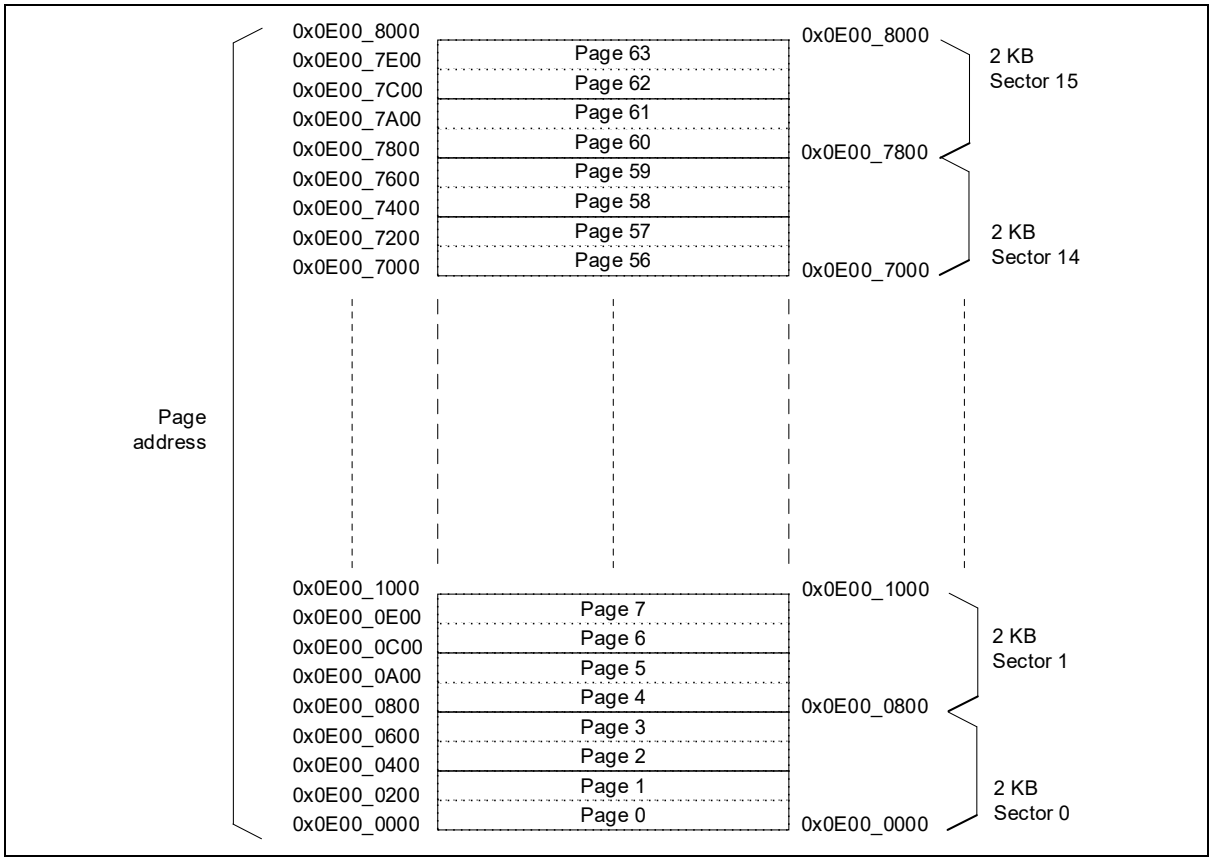
6.3.1.2 Data Flash Memory

Data flash memory has the features as shown below:

Table 44. Data Flash Memory Controller Features

Item	A34G43x
Size	32 KB
Start address	0x0E00_0000
End address	0x0E00_7FFF
Page size	512 bytes
Total page count	64 pages
PGM unit	4 bytes (1 word)
Erase unit	512 bytes 2 KB Bulk (full-chip)

Figure 43. Data Flash Memory Map



6.3.2 Read Access Latency

Flash memory has a read latency that requires a waiting time to access the flash memory, and it needs to be adjusted according to the microcontroller main clock.

How to adjust each latency of code flash memory is described in the following sections.

6.3.2.1 Code Flash Memory Wait Time

1. Determine the correct wait time corresponding to the microcontroller main clock by referring to Table 45.
2. Program the LATENCY[3:0] in CFMC_CONF register to have a new wait time.

Table 45. Internal Code Flash Memory Wait Time by Operating Clock

LATENCY[3:0]	Flash Memory Wait Time	Max. Available Clock Frequency
0000	0-clock wait	Up to 33 MHz
0001	1-clock wait	Up to 66 MHz
0010	2-clock wait	Up to 100 MHz
0011	3-clock wait	Up to 133 MHz
0100	4-clock wait	Up to 166 MHz
0101	5-clock wait	Up to 200 MHz
0110	6-clock wait	Up to 200 MHz
0111	7-clock wait	Up to 200 MHz
1000	8-clock wait	Up to 200 MHz
1001	9-clock wait	Up to 200 MHz
1010	10-clock wait	Up to 200 MHz
1011	11-clock wait	Up to 200 MHz
1100	12-clock wait	Up to 200 MHz
1101	13-clock wait	Up to 200 MHz
1110	14-clock wait	Up to 200 MHz
1111	15-clock wait	Up to 200 MHz

NOTE:

1. Incoming clock signal to the flash must be less than the ‘Max. available clock frequency’ corresponding to the ‘flash wait time’ set in the LATENCY[3:0] in CFMC_CONF. For more information about the incoming clocks, please refer to section 4.4.5.2.

6.3.2.2 Data Flash Memory Wait Time

1. Determine the correct wait time corresponding to the microcontroller main clock by referring to Table 46.
2. Program the LATENCY[3:0] bits in the DFMC_CONF register to have a new wait time.

Table 46. Internal Data Flash Memory Wait Time by Operating Clock

LATENCY[3:0]	Flash Memory Wait Time	Max. Available Clock Frequency
000	0-clock wait	Up to 33 MHz
001	1-clock wait	Up to 66 MHz
010	2-clock wait	Up to 100 MHz
011	3-clock wait	Up to 133 MHz
100	4-clock wait	Up to 166 MHz
101	5-clock wait	Up to 200 MHz
110	6-clock wait	Up to 200 MHz
111	7-clock wait	Up to 200 MHz

NOTE:

1. Incoming clock signal to the flash memory must be less than the 'Max. Available Clock Frequency' corresponding to the 'flash memory wait time' set in the LATENCY[3:0] bits in the DFMC_CONF register. For more information of the incoming clocks, refer to section 4.4.5.2.

6.3.3 Usage of Cache

It is recommended that the code and instruction cache at code flash memory operate in the order described below:

For initial code operation:

1. Cache disable (CFMC_CONF[9:8] = '00')
2. Cache reset enable (CFMC_CTRL[25:24] = '11', Auto clear bit)
3. Cache enable (CFMC_CONF[9:8] = '11')

To disable cache:

1. Cache disable (CFMC_CONF[9:8] = '00')
2. Cache reset enable (CFMC_CTRL[25:24] = '11', Auto clear bit)

To enable cache:

1. Cache enable (CFMC_CONF[9:8] = '11')

If the active flash bank was erased or programmed, the data and instruction cache must be reset. See section 6.6 for a description of active and non-active banks.

6.3.4 Flash Memory Program and Erase Operations

Any interrupt must not be issued while erasing or programming the flash memory. The following routine must be executed before starting the flash memory control.

- All Interrupt disable
- Cache disable
- Interrupts are only available when programming and erasing a different bank area, and the Read-While-Write (RWW) function is also supported. However, to use the RWW function, bank mode must be enabled, and the flash memory must be programmed or erased in another bank.

For access to the flash memory, users must unlock the flash memory before starting the flash memory control, as described in the following section.

6.3.4.1 Access Control for Writing and Erasing Flash Memory

Before writing and erasing data on flash memory, an access enable process must be executed by setting the write protection registers. The access control of the flash memory follows the steps below:

1. Enter key values in the CFMC_FLSKEY or DFMC_FLSKEY register in the order described below:
   A. In normal input case:
      i. KEY1 → KEY2 → KEY3
   B. In non-consecutive input case:
      i. KEY1 → APB (No KEY) write → KEY2 → APB (No KEY) write → KEY3
2. To disable write and erase access to flash memory:
   A. Set the FLOCK in CFMC_CTRL or DFMC_CTRL register to ‘1’.

Read access of flash memory is always possible, regardless of whether read protection or write protection is enabled.

The flash erase operation must be proceeded before starting the flash memory programming.

6.3.5 Flash Memory Erase Sequences

6.3.5.1 Page Erase

Figure 44 shows a timing diagram for page erase operations in the flash memory.

Figure 44. Flash Memory Page Erase Timing Diagram

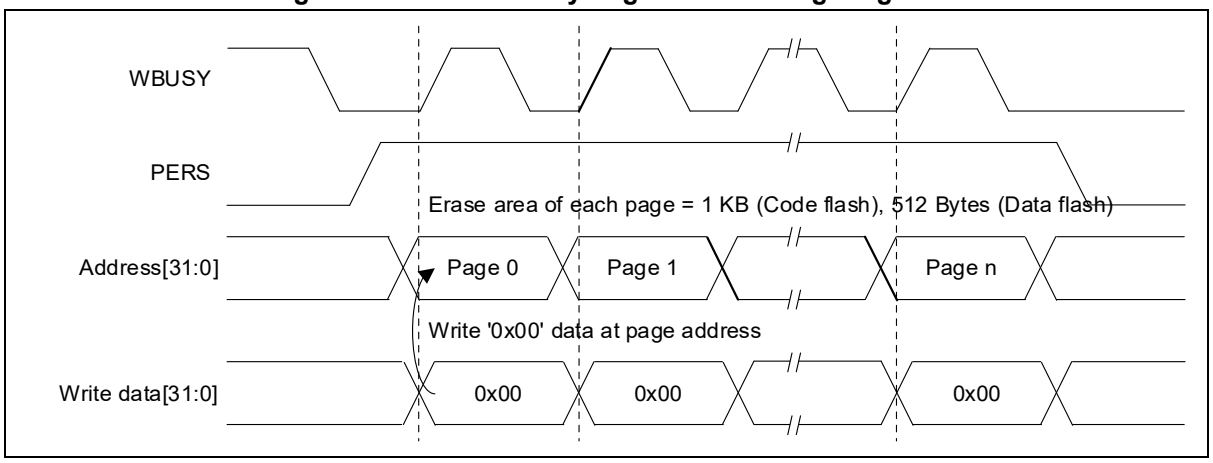


Figure 44 describes the following procedure for the page erase:

1. Check the WBUSY flag in the CFMC_STAT (or DFMC_STAT) register.
2. Set the PERS bit in the CFMC_CTRL (or DFMC_CTRL) register.
3. Write ‘0x00’ data at page address.
4. Check the WBUSY flag in the CFMC_STAT (or DFMC_STAT) register at polling mode or the WDONE flag in the CFMC_STAT (or DFMC_STAT) register at interrupt mode.
5. Clear the PERS bit in the CFMC_CTRL (or DFMC_CTRL) register.

6.3.5.2 Sector Erase

Figure 45 shows a timing diagram for sector erase operations in the flash memory.

Figure 45. Flash Memory Sector Erase Timing Diagram

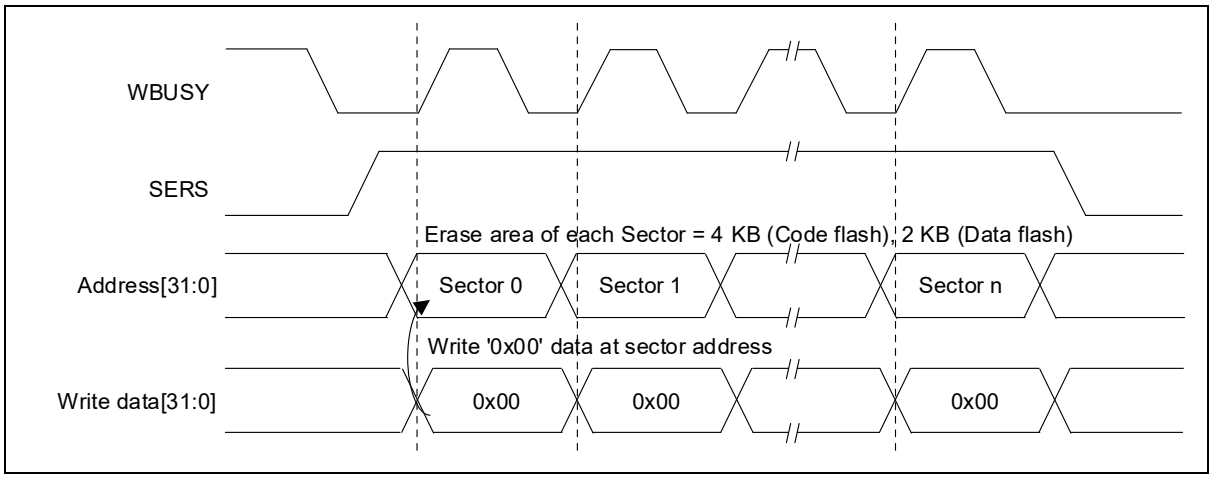


Figure 45 describes the following procedure for the sector erase:

1. Check the WBUSY flag in the CFMC_STAT (or DFMC_STAT) register.
2. Set the SERS bit in the CFMC_CTRL (or DFMC_CTRL) register.
3. Write ‘0x00’ data at sector address.
4. Check the WBUSY flag in the CFMC_STAT (or DFMC_STAT) register at polling mode or the WDONE flag in the CFMC_STAT (or DFMC_STAT) register at interrupt mode.
5. Clear the SERS bit in the CFMC_CTRL (or DFMC_CTRL) register.

6.3.5.3 Bulk Erase

Figure 46 shows a timing diagram for bulk (full-chip) erase operations in the flash memory.

Figure 46. Flash Memory Bulk (Full-Chip) Erase Timing Diagram

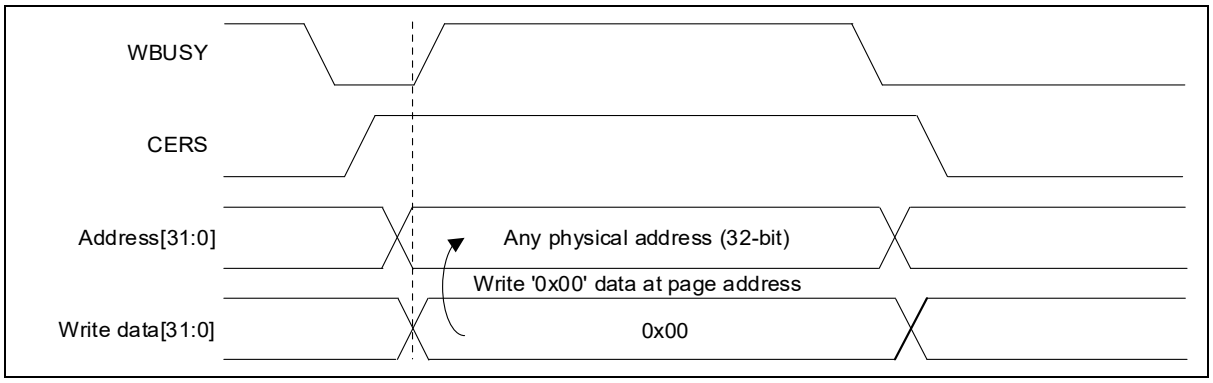


Figure 46 describes the following procedure for the bulk erase.

1. Check the WBUSY flag in the CFMC_STAT (or DFMC_STAT) register.
2. Set the CERS bit in the CFMC_CTRL (or DFMC_CTRL) register.
3. Write ‘0x00’ data at any address.
4. Check the WBUSY flag in the CFMC_STAT (or DFMC_STAT) register at polling mode or the WDONE flag in the CFMC_STAT (or DFMC_STAT) register at interrupt mode.
5. Clear the CERS bit in the CFMC_CTRL (or DFMC_CTRL) register.

6.3.6 Flash Memory Programming Sequences

Before writing to the flash memory, it is important to unlock and unprotect the access area. Note that lock, protection, and all caches (instruction and data) must be disabled during the writing is complete.

6.3.6.1 Standard Programming

Figure 47. Code Flash Memory Program Timing Diagram

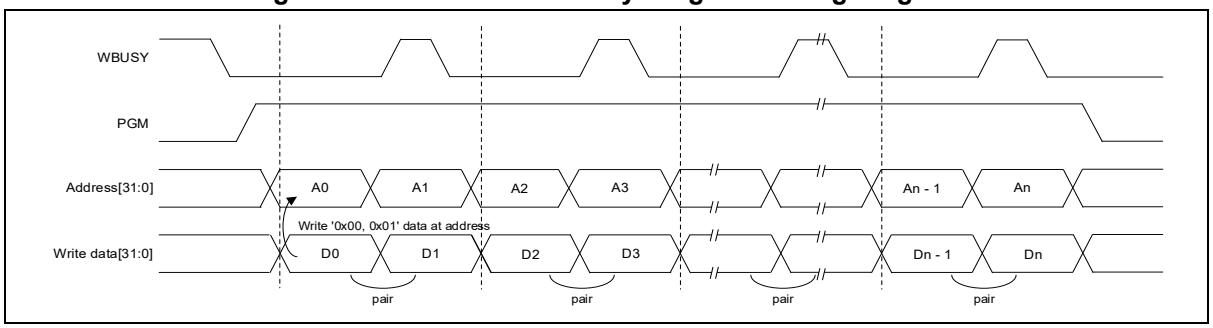


Figure 48. Data Flash Memory Program Timing Diagram

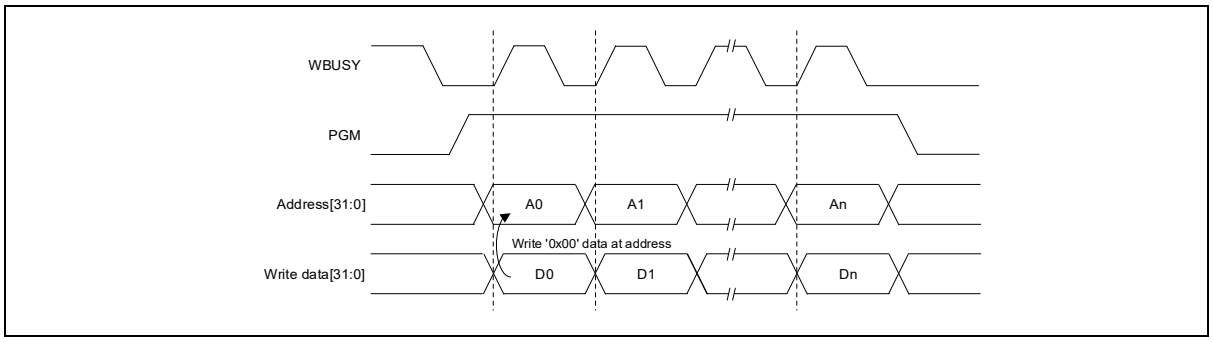


Figure 47 and Figure 48 describe the standard programming procedure below:

1. Check the WBUSY flag in the CFMC_STAT (or DFMC_STAT) register.
2. For the first data write, set the Write Buffer Clear (WBCLR) bit of the CFMC_CTRL register before writing data. (DFMC is not supported.)
   A. When the PGM unit of the code flash writes the first data of two words, it is stored in the write buffer. When the second data is written, it is programmed (PGM) together with the data stored in the write buffer.
3. Set the PGM bit of the CFMC_CTRL (or DFMC_CTRL) register.
4. Write target data at the address.
   A. The PGM unit of the code flash must be written in two words (8 bytes). If the address ends in 0x0 or 0x8 when writing the last data, the write operation will not be performed correctly. In such cases, to ensure proper PGM operation, a dummy write with the erase value (0xFFFF_FFFF) is required at addresses 0x4 or 0xC.

5. Check the WBUSY flag in the CFMC_STAT (or DFMC_STAT) register at polling mode or the WDONE flag in the CFMC_STAT (or DFMC_STAT) register at interrupt mode.
6. Clear the PGM bit in the CFMC_CTRL (or DFMC_CTRL) register.

Table 47. PGM Results After 2 Words Programming

1st Word PGM Address	2nd Word PGM Address	3rd Word PGM Address	Result	Current Write Buffer	WAERR in CFMC_STAT
0x0	0x0	-	No effect	Address 0x0	Not error
0x0	0x4	-	PGM	Empty	Not error
0x0	0x8	-	No effect	Address 0x8	Not error
0x0	0xC	-	Error	Address 0xC	Error Detection
0x4	0x0	-	PGM	Empty	Not error
0x4	0x4	-	No effect	Address 0x4	Not error
0x4	0x8	-	Error	Address 0x8	Error Detection
0x4	0xC	-	No effect	Address 0xC	Not error
0x8	0x0	-	No effect	Address 0x0	Not error
0x8	0x4	-	Error	Address 0x4	Error Detection
0x8	0x8	-	No effect	Address 0x8	Not error
0x8	0xC	-	PGM	Empty	Not error
0xC	0x0	-	No effect	Address 0x0	Not error
0xC	0x4	-	No effect	Address 0x4	Not error
0xC	0x8	-	PGM	Empty	Not error
0xC	0xC	-	No effect	Address 0xC	Not error
0x0	0x8	0xC	PGM	Empty	Not error
0x8	0x0	0xC	Error	Address 0xC	Error Detection

NOTES:

1. The table above explains cases where writing in word form at random addresses, rather than incrementing the address in word units, may not function correctly. If the PGM operation is performed in the correct sequential order of word units for the address, it will function properly.
2. The PGM unit must be written in 2-word (8-byte) units. If writing sequentially to the address and the last data is written at an address ending in 0x0 or 0x8, the PGM operation will not be executed, and the data will only be stored in the write buffer. In such cases, to ensure proper PGM execution, an erase value (0xFFFF_FFFF) must be written to addresses 0x4 or 0xC.

6.4 Flash Memory Protection

Read protection is a feature provided in the internal code flash memory and protects user code from external hacking. The read protection operates in three different protection mode levels such as unprotection (UNPROT), Level 1 (LVL1), and Level 2 (LVL2).

6.4.1 Read Protection

Table 48 describes three different protection mode levels of the read protection, such as UNPROT, LVL1, and LVL2.

Table 48. Available Operating Modes by Protection Level

Protection Level	Operation Mode	Main (Code Flash)		User Info0		User Info1/2/3		Main (Data Flash)	
		Read	Write	Read	Write	Read	Write	Read	Write
UNPROT	Normal mode	O	O	O	O	O	O	O	O
	Debug mode	O	O	O	O	O	O	O	O
	Boot mode	O	O	O	O	O	O	O	O
LVL1	Normal mode	O	O	O	X	O	O	O	O
	Debug mode	X	X	X	X	X	X	X	X
	Boot mode	X	X	X	X	X	X	X	X
LVL2	Normal mode	O	O	O	X	O	O	O	O
	Debug mode	This mode cannot be entered.							
	Boot mode	X	X	X	X	X	X	X	X

NOTES:

1. Priority levels of the read protection mode are as follows:
   Normal protection to password protection
   A. Normal protection: LVL2 > LVL1 > UNPROT
   B. Password protection: LVL2 > LVL1 > UNPROT
2. A higher protection level cannot transition to a lower level without the bulk (chip) erase and User Info0 erase executed.
   e.g., procedure for transitioning from read protection LVL1 to UNPROT.
   A. Full-chip erase
   B. Check the CERSD flag in the CFMC_READPROT register.
   C. User Info0 erase
   D. Check the INFO0ERSD flag in the CFMC_READPROT register.
   E. Write '0xFF' to the RPROT[7:0] bits in the CFMC_READPROT register. After that, read protection LVL1 transition to UNPROT mode.

6.4.1.1 Read Protection: UNPROT (Unprotection)

Debug access is available, and there is no restriction on accessing the entire internal flash memory.

Table 49. Read Protection Operation in UNPROT Level

Normal Mode		Target Access Area			
		BootROM	Code Flash	SRAM	User Info
Execution Area	Boot/Code/SRAM	R	All	All	All
	User Info	X	X	X	X
	DMA	X	X	R/W	X
	Debug	R	All	All	All

NOTE:

1. R: read access, W: write access, All = read, write, erase access

6.4.1.2 Read Protection: LVL1 (Level-1)

Debug access is available, but the internal flash memory is read as '0x55AA_55AA' by external access. Therefore, the microcontroller cannot run while accessing the internal flash memory for debugging, and the flash memory read operation by DMA is not supported because the DMA settings can cause code leakage.

Table 50. Read Protection Operation in Level 1 (LVL1)

Read Protection Level-1		Target Access Area			
		BootROM	Code Flash	SRAM	User Info
Execution Area	Boot/Code/SRAM	R	All	All	R/W/(E)
	User Info	X	X	X	X
	DMA	X	X	R/W	X
	Debug	X	X	R/W	-/W/(E)

NOTE:

1. R: read access, W: write access, E: erase access, All = read, write and erase access

6.4.1.3 Read Protection: LVL2 (Level-2)

Debug access is not available, and the flash memory read operation by DMA is not supported because the DMA settings can cause code leakage.

Table 51. Read Protection Operation in Level-2 (LVL2)

Read Protection Level-2		Target Access Area			
		BootROM	Code Flash	SRAM	User Info
Execution Area	Boot/Code/SRAM	R	All	All	R/W/(E)
	User Info	X	X	X	X
	DMA	X	X	R/W	X
	Debug	X	X	X	X

NOTE:

1. R: read access, W: write access, E: erase access, All = read, write and erase access

6.4.1.4 Read Protection Setting

Users write data to the read protection area of the User Info0, and then the BootROM reads the data to enable the read protection when it runs on reset. After enabling the read protection, users must reset microcontroller to apply the read protection settings.

For the direct setup, users can write data in the read protection register to apply the settings immediately.

The read protection register is available for the change of the read protection mode level from low to high, and the change from high to low level is ignored. In addition, if a value other than the pre-set value is written to the read protection register, the highest level (LVL2) of the read protection mode level is applied.

6.4.1.5 Read Protection Removal

To remove the read protection, users must reset after deleting the related data in the User Info0. To delete the data of the User Info0 in the read protection mode, bulk erase must be executed for the code flash. The state when the bulk erase is complete for the code flash is stored and maintained until reset occurs.

6.4.1.6 Password Settings

Users can set the password only once by writing the password preset value to the CFMC_PWPRST register. If the input value of CFMC_PWIN register matches the preset value of the CFMC_PWPRST register, the read protection mode becomes normal mode.

For example, if users want to set the password to 0x1234_5678, the preset value 0x1234_5678 is written to the CFMC_PWPRST register as shown below:

```
CFMC_PWPRST = 0x12345678;
CFMC_PWPRST = 0x12345678;
```

6.4.2 Write Protection

Users can set up the write or erase protection for a certain area of the code and data flash. Before setting the protections, remember the following:

- If any of the FPBY16_n or the FUBLP8K_y bit in the CFMC_FLS0PROT register and the CFMC_FLS1PROT register is ‘1’ (Protection), the Bulk Erase does not work because the protected area is included in the erase area.
- Similarly, if some **Unprotection** area is set to **Protection** (the FUP1K_x bit in the CFMC_FLSPROT register is ‘0’) when the FPBY16_15 bit is '1’, the erase commands such as ‘Chip erase’ do not work for the area that includes protection area.
- Write Protection follows the physical address.

6.4.2.1 Write Protection for Code Flash Memory 0

The CFMC_FLS0PROT register is used to set the **Protection** and **Unprotection** as described in Table 52, Table 53, Table 55, Table 56 and Table 57.

Table 52. Base Address of Protection in Code Flash Memory 0

Mode	Conditions	Base Address (A34G430)	Base Address (A34G43A)
Linear Mode	-	0x0000_0000	0x0000_0000
Bank Mode	No swap	0x0000_0000	0x0000_0000
	Swap	0x0008_0000	0x0010_0000

NOTES:

1. Code write protection is based on the physical address, so be careful when swapping banks. In the case of the A34G43A, write protection is set based on physical address. For example, under the above conditions, if 64 KB from 0x0000_0000 is write protected, 64 KB from 0x0010_0000 becomes write protected after a bank swap.
2. Unprotection Selection bit in this table has a higher priority level than the Protection Selection bit.

Table 53. Code Flash Memory 0 Write Protection Area in 32 KB Units (A34G430)

n	Offset	n	Offset
0	0x0000_0000 ~ 0x0000_7FFF	1	0x0000_8000 ~ 0x0000_FFFF
2	0x0001_0000 ~ 0x0001_7FFF	3	0x0001_8000 ~ 0x0001_FFFF
4	0x0002_0000 ~ 0x0002_7FFF	5	0x0002_8000 ~ 0x0002_FFFF
6	0x0003_0000 ~ 0x0003_7FFF	7	0x0003_8000 ~ 0x0003_FFFF
8	0x0004_0000 ~ 0x0004_7FFF	9	0x0004_8000 ~ 0x0004_FFFF
10	0x0005_0000 ~ 0x0005_7FFF	11	0x0005_8000 ~ 0x0005_FFFF
12	0x0006_0000 ~ 0x0006_7FFF	13	0x0006_8000 ~ 0x0006_FFFF
14	0x0007_0000 ~ 0x0007_7FFF	15	0x0007_8000 ~ 0x0007_FFFF

Table 54. Code Flash Memory 0 Write Protection Area in 64 KB Units (A34G43A)

n	Offset	n	Offset
0	0x0000_0000 ~ 0x0000_FFFF	1	0x0001_0000 ~ 0x0001_FFFF
2	0x0002_0000 ~ 0x0002_FFFF	3	0x0003_0000 ~ 0x0003_FFFF
4	0x0004_0000 ~ 0x0004_FFFF	5	0x0005_0000 ~ 0x0005_FFFF
6	0x0006_0000 ~ 0x0006_FFFF	7	0x0007_0000 ~ 0x0007_FFFF
8	0x0008_0000 ~ 0x0008_FFFF	9	0x0009_0000 ~ 0x0009_FFFF
10	0x000A_0000 ~ 0x000A_FFFF	11	0x000B_0000 ~ 0x000B_FFFF
12	0x000C_0000 ~ 0x000C_FFFF	13	0x000D_0000 ~ 0x000D_FFFF
14	0x000E_0000 ~ 0x000E_FFFF	15	0x000F_0000 ~ 0x000F_FFFF

Table 55. Code Flash Memory 0 Unprotection Area (Bottom 64 KB) in 8 KB Units (A34G43x)

y	Offset	y	Offset
0	0x0000_0000 ~ 0x0000_1FFF	1	0x0000_2000 ~ 0x0000_3FFF
2	0x0000_4000 ~ 0x0000_5FFF	3	0x0000_6000 ~ 0x0000_7FFF
4	0x0000_8000 ~ 0x0000_9FFF	5	0x0000_A000 ~ 0x0000_BFFF
6	0x0000_C000 ~ 0x0000_DFFF	7	0x0000_E000 ~ 0x0000_FFFF

Table 56. Code Flash Memory 0 Unprotection Area (Top 8 KB) in 1 KB Units (A34G430)

x	Offset	x	Offset
0	0x0007_E000 ~ 0x0007_E3FF	1	0x0007_E400 ~ 0x0007_E7FF
2	0x0007_E800 ~ 0x0007_EBFF	3	0x0007_EC00 ~ 0x0007_EFFF
4	0x0007_F000 ~ 0x0007_F3FF	5	0x0007_F400 ~ 0x0007_F7FF
6	0x0007_F800 ~ 0x0007_FBFF	7	0x0007_FC00 ~ 0x0007_FFFF

Table 57. Code Flash Memory 0 Unprotection Area (Top 8 KB) in 1 KB Units (A34G43A)

x	Offset	x	Offset
0	0x000F_E000 ~ 0x000F_E3FF	1	0x000F_E400 ~ 0x000F_E7FF
2	0x000F_E800 ~ 0x000F_EBFF	3	0x000F_EC00 ~ 0x000F_EFFF
4	0x000F_F000 ~ 0x000F_F3FF	5	0x000F_F400 ~ 0x000F_F7FF
6	0x000F_F800 ~ 0x000F_FBFF	7	0x000F_FC00 ~ 0x000F_FFFF

6.4.2.2 Write Protection for Code Flash Memory 1

The CFMC_FLS1PROT register is used to set the Protection and Unprotection as described in Table 58, Table 59, Table 61, and Table 63.

Table 58. Code Flash Memory 1 Base Address of Protection

Mode	Conditions	Base Address (A34G430)	Base Address (A34G43A)
Linear Mode	-	0x0008_0000	0x0010_0000
Bank Mode	No swap	0x0008_0000	0x0010_0000
	Swap	0x0000_0000	0x0000_0000

NOTES:

1. Code write protection is based on the physical address, so be careful when swapping banks. In the case of the A34G43A, write protection is set based on physical address. For example, under the above conditions, if 64 KB from 0x0000_0000 is write protected, 64 KB from 0x0010_0000 becomes write protected after a bank swap.
2. Unprotection Selection bit in this table has a higher priority level than the Protection Selection bit.

Table 59. Code Flash Memory 1 Write Protection Area in 32 KB Units (A34G430)

n	Offset	n	Offset
0	0x0000_0000 ~ 0x0000_7FFF	1	0x0000_8000 ~ 0x0000_FFFF
2	0x0001_0000 ~ 0x0001_7FFF	3	0x0001_8000 ~ 0x0001_FFFF
4	0x0002_0000 ~ 0x0002_7FFF	5	0x0002_8000 ~ 0x0002_FFFF
6	0x0003_0000 ~ 0x0003_7FFF	7	0x0003_8000 ~ 0x0003_FFFF
8	0x0004_0000 ~ 0x0004_7FFF	9	0x0004_8000 ~ 0x0004_FFFF
10	0x0005_0000 ~ 0x0005_7FFF	11	0x0005_8000 ~ 0x0005_FFFF
12	0x0006_0000 ~ 0x0006_7FFF	13	0x0006_8000 ~ 0x0006_FFFF
14	0x0007_0000 ~ 0x0007_7FFF	15	0x0007_8000 ~ 0x0007_FFFF

Table 60. Code Flash Memory 1 Write Protection Area in 64 KB Units (A34G43A)

n	Offset	n	Offset
0	0x0000_0000 ~ 0x0000_FFFF	1	0x0001_0000 ~ 0x0001_FFFF
2	0x0002_0000 ~ 0x0002_FFFF	3	0x0003_0000 ~ 0x0003_FFFF
4	0x0004_0000 ~ 0x0004_FFFF	5	0x0005_0000 ~ 0x0005_FFFF
6	0x0006_0000 ~ 0x0006_FFFF	7	0x0007_0000 ~ 0x0007_FFFF
8	0x0008_0000 ~ 0x0008_FFFF	9	0x0009_0000 ~ 0x0009_FFFF
10	0x000A_0000 ~ 0x000A_FFFF	11	0x000B_0000 ~ 0x000B_FFFF
12	0x000C_0000 ~ 0x000C_FFFF	13	0x000D_0000 ~ 0x000D_FFFF
14	0x000E_0000 ~ 0x000E_FFFF	15	0x000F_0000 ~ 0x000F_FFFF

Table 61. Code Flash Memory 1 Unprotection Area (Bottom 64 KB) in 8 KB Units (A34G43x)

y	Offset	y	Offset
0	0x0000_0000 ~ 0x0000_1FFF	1	0x0000_2000 ~ 0x0000_3FFF
2	0x0000_4000 ~ 0x0000_5FFF	3	0x0000_6000 ~ 0x0000_7FFF
4	0x0000_8000 ~ 0x0000_9FFF	5	0x0000_A000 ~ 0x0000_BFFF
6	0x0000_C000 ~ 0x0000_DFFF	7	0x0000_E000 ~ 0x0000_FFFF

Table 62. Code Flash Memory 1 Unprotection Area (Top 8 KB) in 1 KB Units (A34G430)

x	Offset	x	Offset
0	0x0007_E000 ~ 0x0007_E3FF	1	0x0007_E400 ~ 0x0007_E7FF
2	0x0007_E800 ~ 0x0007_EBFF	3	0x0007_EC00 ~ 0x0007_EFFF
4	0x0007_F000 ~ 0x0007_F3FF	5	0x0007_F400 ~ 0x0007_F7FF
6	0x0007_F800 ~ 0x0007_FBFF	7	0x0007_FC00 ~ 0x0007_FFFF

Table 63. Code Flash Memory 1 Unprotection Area (Top 8 KB) in 1 KB Units (A34G43A)

x	Offset	x	Offset
0	0x000F_E000 ~ 0x000F_E3FF	1	0x000F_E400 ~ 0x000F_E7FF
2	0x000F_E800 ~ 0x000F_EBFF	3	0x000F_EC00 ~ 0x000F_EFFF
4	0x000F_F000 ~ 0x000F_F3FF	5	0x000F_F400 ~ 0x000F_F7FF
6	0x000F_F800 ~ 0x000F_FBFF	7	0x000F_FC00 ~ 0x000F_FFFF

6.4.2.3 Write Protection for Data Flash Memory

The DFMC_FLSPROT register is used to set the Protection and Unprotection as described in Table 64 and Table 65.

Table 64. Data Flash Memory Write Protection Area in 2 KB Units

N	Address	N	Address
0	0x0E00_0000 ~ 0x0E00_07FF	1	0x0E00_0800 ~ 0x0E00_0FFF
2	0x0E00_1000 ~ 0x0E00_17FF	3	0x0E00_1800 ~ 0x0E00_1FFF
4	0x0E00_2000 ~ 0x0E00_27FF	5	0x0E00_2800 ~ 0x0E00_2FFF
6	0x0E00_3000 ~ 0x0E00_37FF	7	0x0E00_3800 ~ 0x0E00_3FFF
8	0x0E00_4000 ~ 0x0E00_47FF	9	0x0E00_4800 ~ 0x0E00_4FFF
10	0x0E00_5000 ~ 0x0E00_57FF	11	0x0E00_5800 ~ 0x0E00_5FFF
12	0x0E00_6000 ~ 0x0E00_67FF	13	0x0E00_6800 ~ 0x0E00_6FFF
14	0x0E00_7000 ~ 0x0E00_77FF	15	0x0E00_7800 ~ 0x0E00_7FFF

Table 65. Data Flash Memory Unprotection Area (Top 4 KB) in 512 Bytes Units

X	Address	X	Address
0	0x0E00_7000 ~ 0x0E00_71FF	1	0x0E00_7200 ~ 0x0E00_73FF
2	0x0E00_7400 ~ 0x0E00_75FF	3	0x0E00_7600 ~ 0x0E00_77FF
4	0x0E00_7800 ~ 0x0E00_79FF	5	0x0E00_7A00 ~ 0x0E00_7BFF
6	0x0E00_7C00 ~ 0x0E00_7DFF	7	0x0E00_7E00 ~ 0x0E00_7FFF

6.4.2.4 Write Protection with Code Flash Memory Bank Swap

Write protection by the CFMC_FLS0PROT and CFMC_FLS1PROT registers varies in area depending on the swap status of the memory bank. Applications that require memory area data updates while user code is running should disable write protection to allow data updates only in inactive bank areas depending on memory swap status. Write protection follows the physical address.

Write protected coverage of Non-active block before Bank Swap in A34G43A

CFMC_FLS0PROT = 0x0000_0000

CFMC_FLS1PROT = 0x0000_FFFF

- Active block (0x0000_0000 to 0x000F_FFFF) is write/erase accessible.
- Non-active block (0x0010_0000 to 0x001F_FFFF) is write/erase inaccessible.

Write protected coverage of Non-active block after Bank Swap in A34G43A

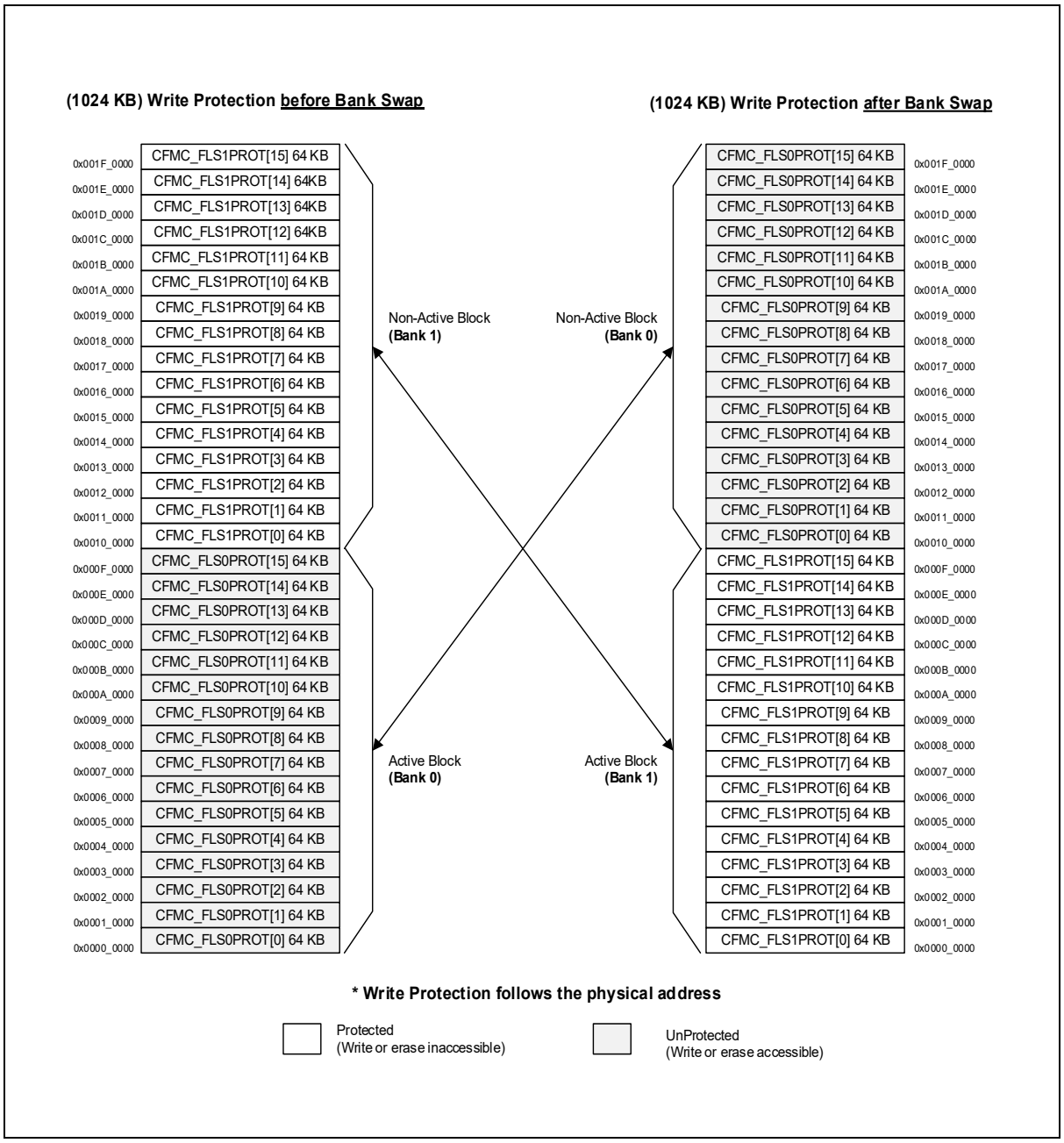
CFMC_FLS0PROT = 0x0000_0000

CFMC_FLS1PROT = 0x0000_FFFF

- Active block (0x0000_0000 to 0x000F_FFFF) is write/erase inaccessible.
- Non-active block (0x0010_0000 to 0x001F_FFFF) is write/erase accessible.

As above, the write protection area changes after bank swap.

Figure 49. Write-Protection Coverage of Code Flash Memory when Using Memory Swap



6.4.3 Securable Memory Area (LVL1, LVL2)

Using the read protection, users can secure the flash memory area against being read by external devices (refer to 6.4.1).

6.4.4 Disabling Core Debug Access (LVL2)

Using the read protection, users can block the debugger connections (refer to 6.4.1).

6.5 Flash Memory CRC Check

The A34G43x uses the 16-bit specific polynomial to calculate the CRC check for the code and data flash memories.

Its internal data input bus width is 128 bits, and the CRC check control sequence is as follows:

Background mode (CRC check is running at only flash memory idle state):

1. Set the CRC check start address.
2. Set the CRC check end address.
3. Reset the CRC check data.
4. Enable the BGEN (Background mode Enable) bit in the CFMC_CHKCTRL (or DFMC_CHKCTRL) register.
5. Check the CBUSY flag in the CFMC_STAT (or DFMC_STAT) register at polling mode or the CDONE flag in the CFMC_STAT (or DFMC_STAT) register at interrupt mode.

Burst mode (CRC Check is running continuously with microcontroller core halt state):

1. Set the CRC check start address.
2. Set the CRC check end address.
3. Reset the CRC check data.
4. Enable the BSTEN (Burst mode enable) bit in the CFMC_CHKCTRL (or DFMC_CHKCTRL) register.
5. Check the CBUSY flag in the CFMC_STAT register at polling mode or the CDONE flag in the CFMC_STAT (or DFMC_STAT) register at interrupt mode.

6.6 Bank Swap of Code Flash Memory

6.6.1 Bank Swap of Code Flash Memory

The A34G43x series provides a flash memory bank selection function for user applications and stable update and execution of User Bootloader.

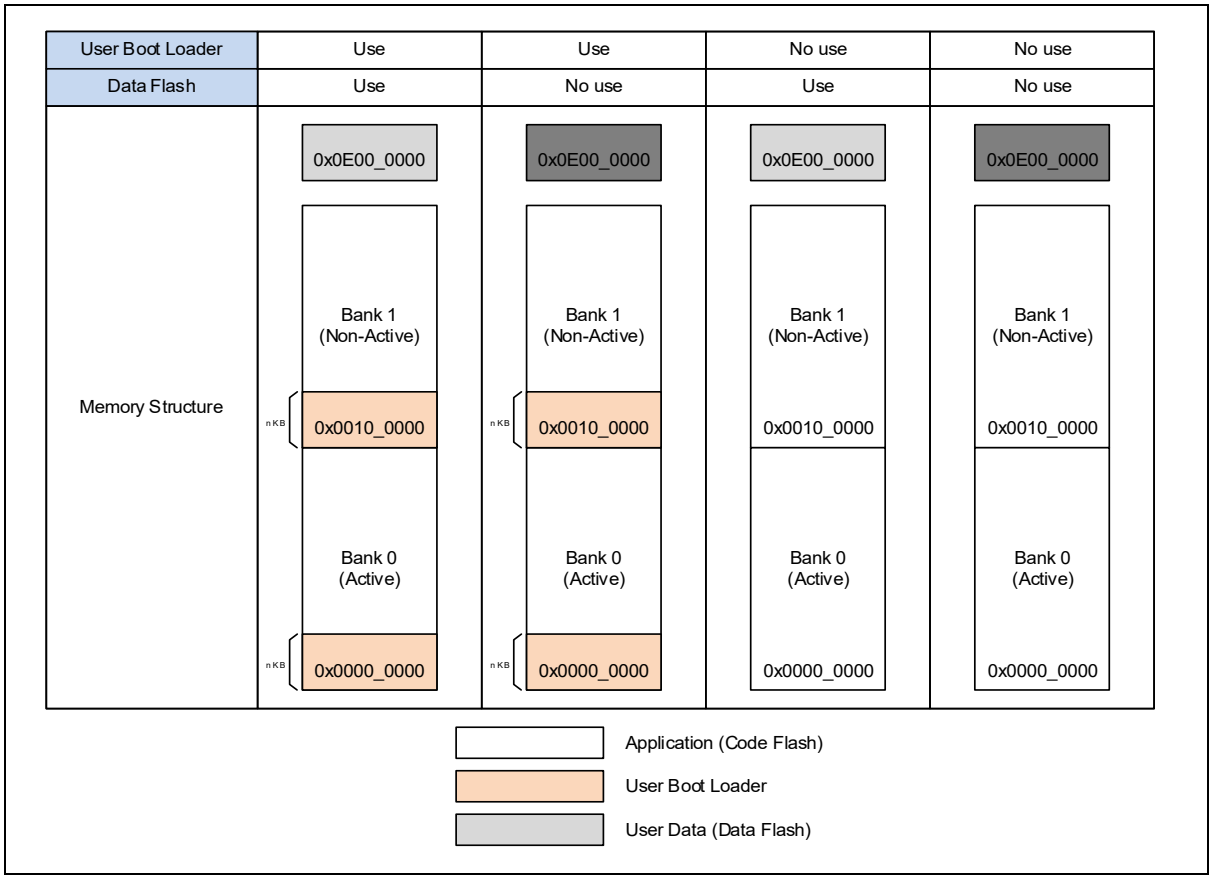
To enable Memory Swap function in the A34G43A, users need to divide 2048 KB flash memory into two of 1024 KB and develop them for programming.

To use the Memory Swap function in user applications, a user is required to acquire information of Bank Selection registers and memory structures.

Table 66. Memory Bank Usage Status of Code Flash Memory

CFMC_BANK <BM>	CFMC_BANK <BS>	Bank Status	Active Block	Non-Active Block
1	1	Disable	Single	
0	1	Enable	Bank0	Bank1
0	0	Swap	Bank1	Bank0

Figure 50. A34G43A Memory Structure with Bank Swap Available



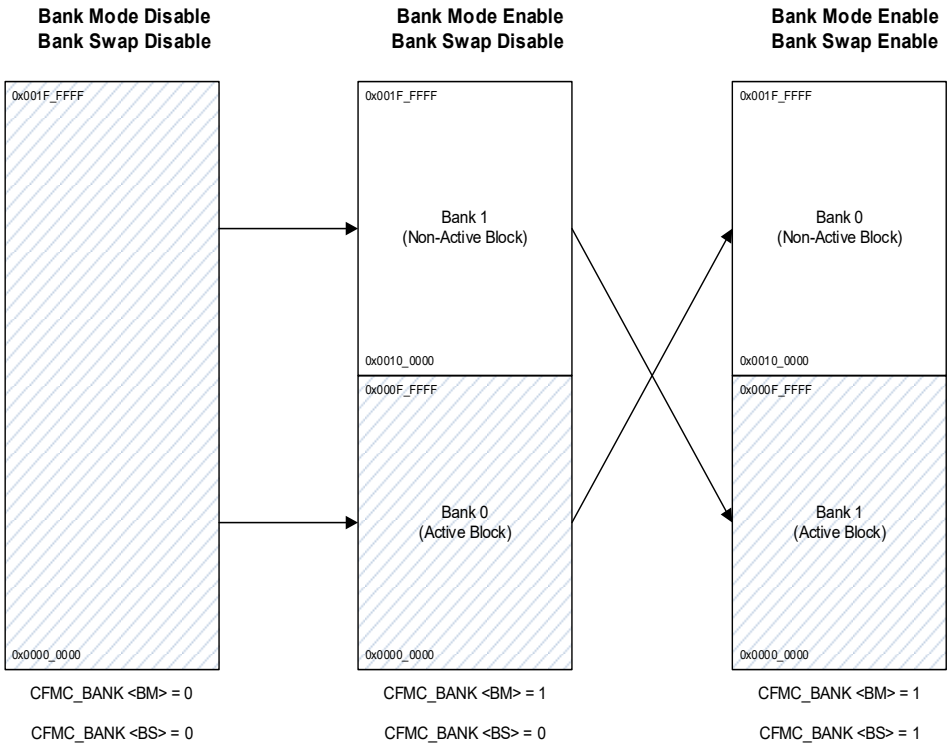
RWW function works when writing non-active bank in bank mode.

6.6.2 Bank Swap Operation

When the Memory Bank Swap function is enabled, memory block starting with address 0x0000_0000 is considered as Active block, where code will be executed after the BootROM operation. Next to the active block, non-active block is located where code will be updated or is used for back up operation.

When the Memory Bank Swap function is enabled, certain code is updated in the non-active area, Swap control bit is set, and the system restarts successively. Then the value of Swap Flag (SFLAG) is monitored in BootROM and switching between the Active block and the non-active block occurs. By the Swap operation, the non-active block where the code was updated switches to the Active block, while the Active block where the code was executed before the update operation switches to the non-active block.

Figure 51. Logical Address Change of Code Flash Memory by Bank Swap Settings



NOTE:

1. Note that write protection must be set based on physical address during bank swap mode. (See Figure 49.)

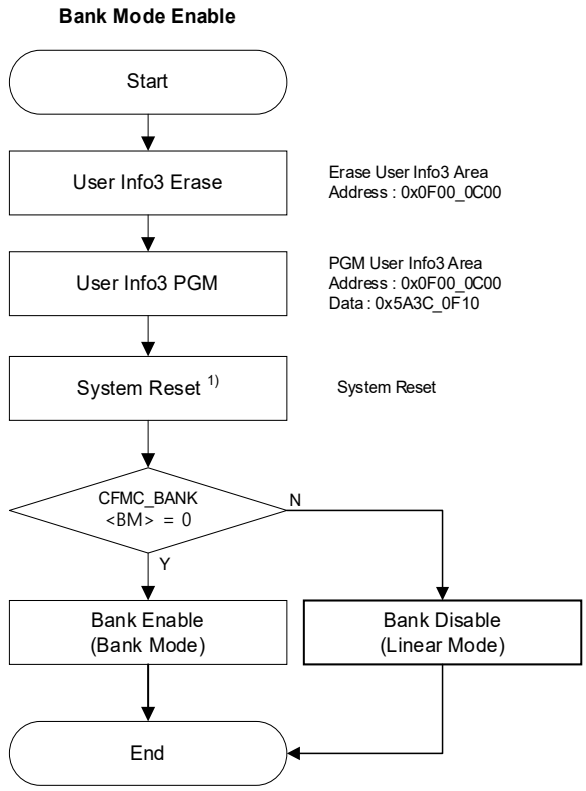
6.6.3 Limitations and Recommendations

When a main program is running in Bank1 area, where the read protection is applied, if full-chip erase is executed to disable only the read protection permanently, user data in whole area of Bank0 and Bank1 is cleared. After system reset, the read protection is disabled. However, since Bank Swap function is still active, program in Bank1 is supposed to be executed but Bank1 does not have any command at all.

6.6.4 Sequence of Bank Mode Enable or Swap

Figure 52, Figure 53, and Figure 54 shows how to enable Bank mode and use Bank swap in the code flash memory. Please note that to disable Bank mode or Bank swap, users must erase the User Info3 area. Refer to Table 66 for the memory bank usage status of the code flash memory.

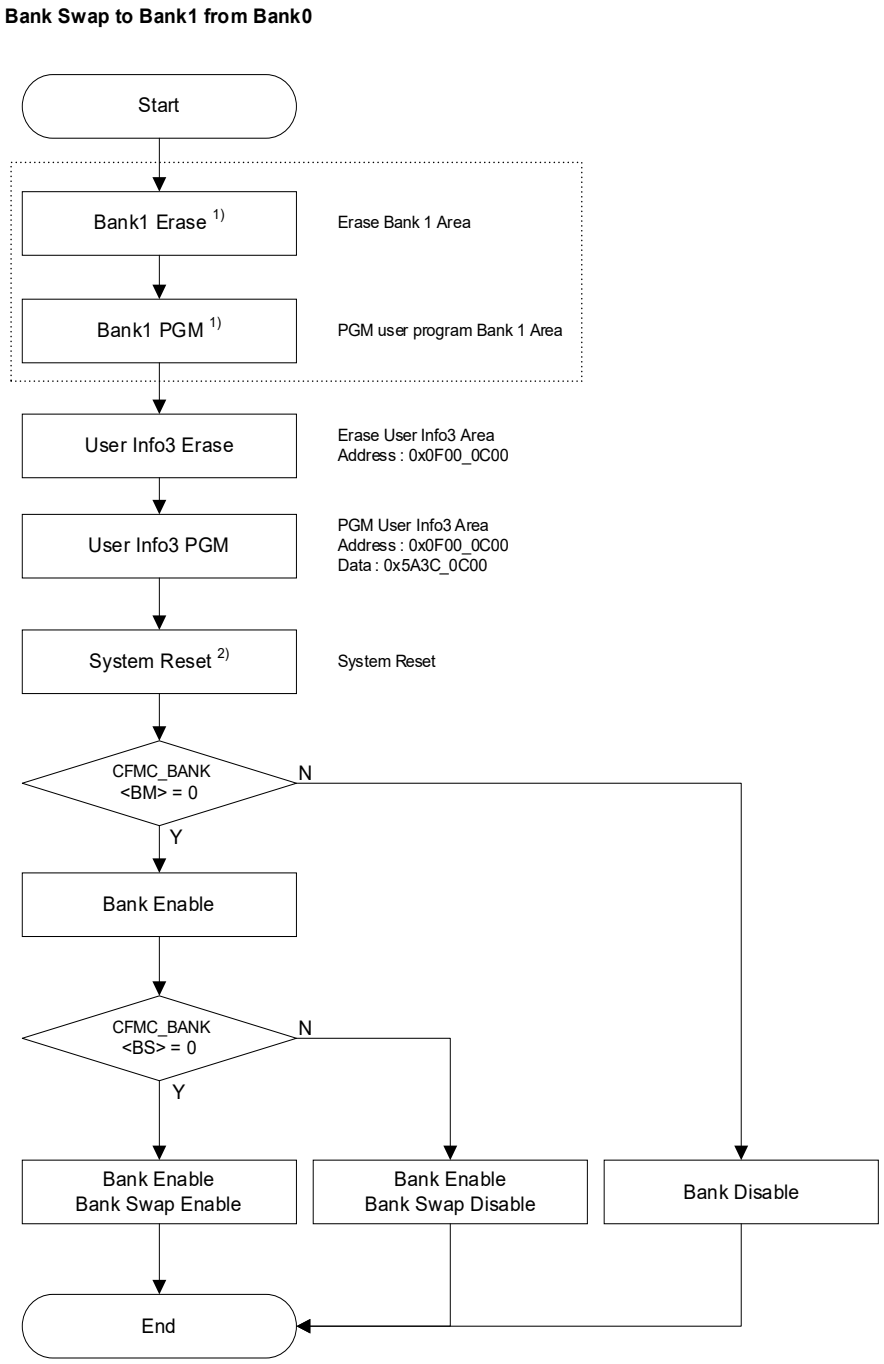
Figure 52. Sequence of Bank Mode Enable



NOTE:

1. Users must use the SWRST bit in the SCU_SRCR register instead of NVIC_SystemReset() or the SCB_AIRCR register setting for microcontroller reset

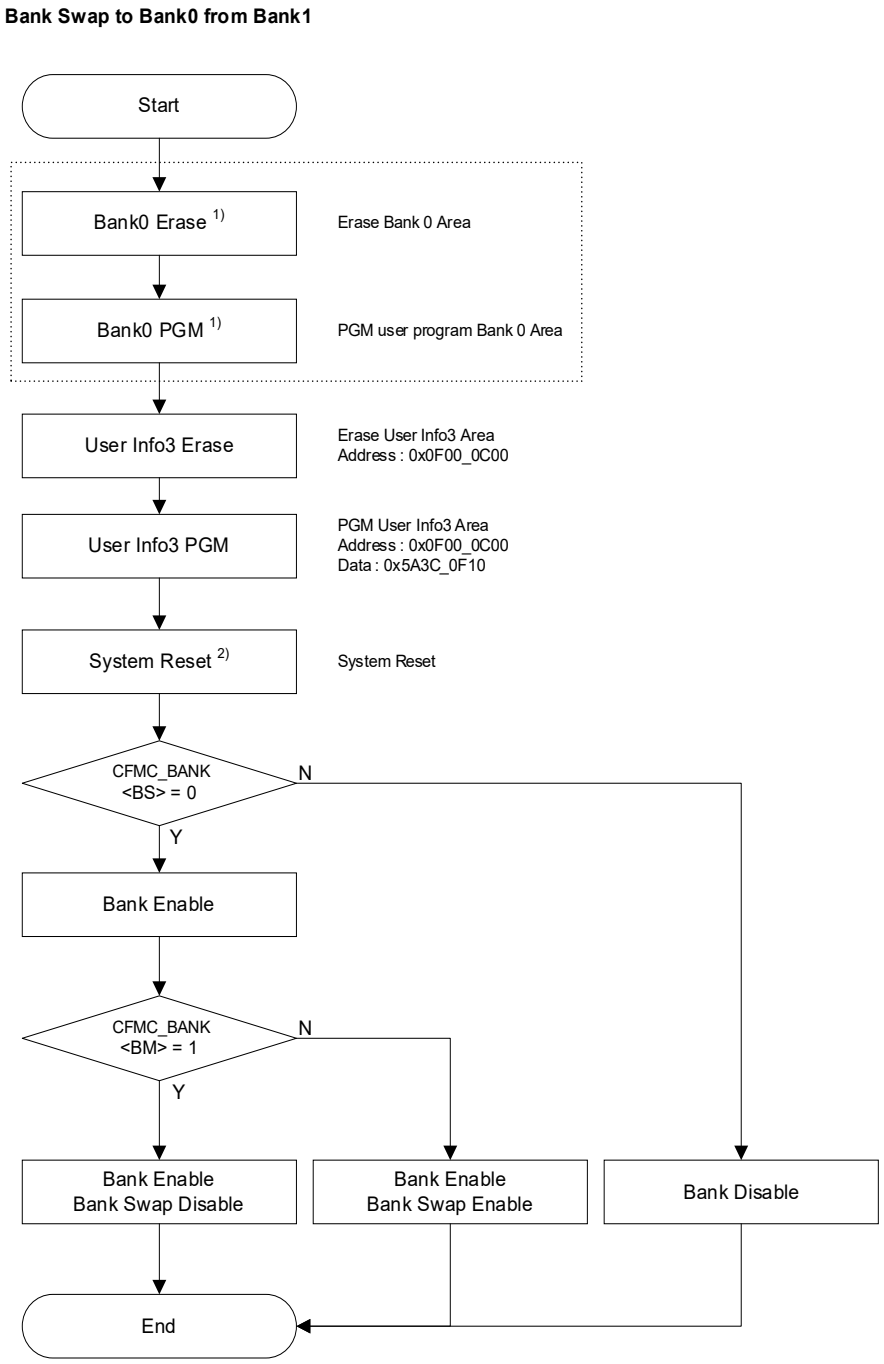
Figure 53. Sequence of Bank Swap: Bank0 → Bank1



NOTES:

1. When performing Bank Swap from Bank1 to Bank0, program can be executed in Bank0 after system reset only if there is a program in Bank0.
2. Users must use the SWRST bit in the SCU_SRCR register instead of NVIC_SystemReset() or the SCB_AIRCR register setting for microcontroller reset

Figure 54. Sequence of Bank Swap: Bank1 → Bank0



NOTES:

1. When performing Bank Swap from Bank1 to Bank0, program can be executed in Bank0 after system reset only if there is a program in Bank0.
2. Users must use the SWRST bit in the SCU_SRCR register instead of NVIC_SystemReset() or the SCB_AIRCR register setting for microcontroller reset.

6.7 Read-While-Write (RWW)

The A34G43x supports Read-While-Write (RWW) function, which allows the microcontroller to be programmed while running from code flash memory. However, RWW is only available under the following conditions.

Requires Bank enable or Bank swap. However, the same bank does not support RWW even if it is in Bank Enable or Bank Swap state. Refer to Figure 48.

Bank Disable does not support RWW. Refer to the 6.10.16

Table 67. Code Flash Memory RWW Features - Bank Disable (BM = 0, BS = 0)

Bank Status: Disable	Bank0		Bank1	
	Erase	Write	Erase	Write
Bank0	X [1)]	X [1)]	X [1)]	X [1)]
Bank1	X [1)]	X [1)]	X [1)]	X [1)]

Table 68. Code Flash Memory RWW Features - Bank Enable (BM = 1), Swap Disable (BS = 0)

Bank Status: Enable	Bank0 (Active block)		Bank1 (Non-Active block)	
	Erase	Write	Erase	Write
Bank0 (Active block)	X [1)]	X [1)]	O	O
Bank1 (Non-Active block)	O	O	X [1)]	X [1)]

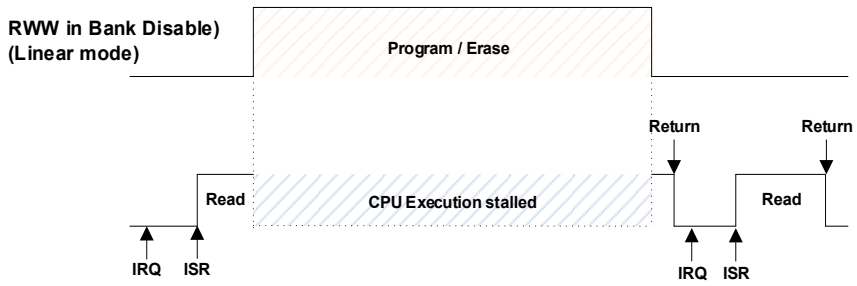
Table 69. Code Flash Memory RWW Features - Bank Enable (BM = 1), Swap Enable (BS = 1)

Bank Status: Enable & Swap	Bank1 (Active block)		Bank0 (Non-Active block)	
	Erase	Write	Erase	Write
Bank1 (Active block)	X [1)]	X [1)]	O	O
Bank0 (Non-Active block)	O	O	X [1)]	X [1)]

NOTE:

1. CPU execution is stalled. E.g.) Interrupt doesn't operate during erase and write.

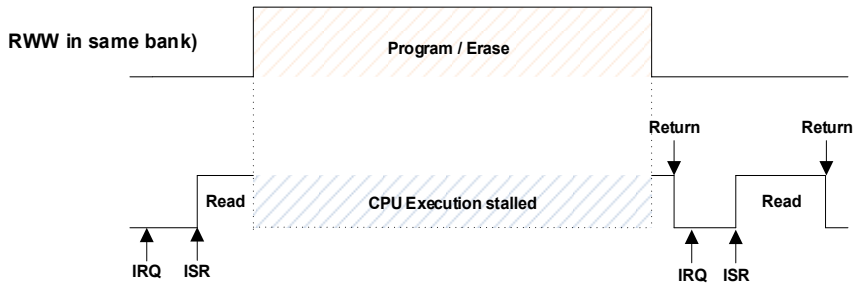
Figure 55. RWW in Code Flash Bank Disable



NOTES:

1. IRQ: Interrupt Request
2. ISR: Interrupt Service Routine

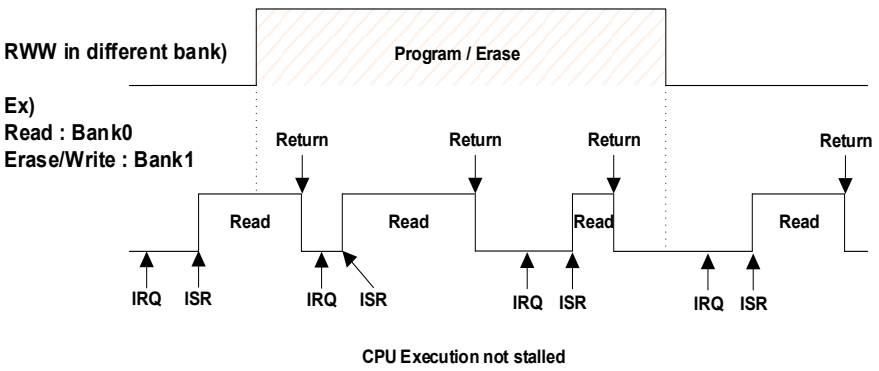
Figure 56. RWW in Code Flash Bank Enable and Bank Swap (Same Bank)



NOTES:

1. IRQ: Interrupt Request
2. ISR: Interrupt Service Routine

Figure 57. RWW in Code Flash Bank Enable and Bank Swap (Different Bank)



NOTES:

1. IRQ: Interrupt Request
2. ISR: Interrupt Service Routine

6.8 Error Correction Code (ECC)

The ECC is based on an optimal minimum odd weight error parity code class that provides better performance than SECDED based on the Hamming code principle, and its use provides improved protection against bit flipping in flash memory.

When writing to the code flash area in 2-word (64-bit) units or to the data flash area in 1-word (32-bit) units, ECC parity bits are generated as follows: for the Code area, ECC parity bits (8-bit) are generated in 2-word units, and for the Data area, ECC parity bits (7-bit) are generated in 1-word units.

Single-bit or double-bit errors can be detected by comparing the generated parity bit with the flash area word when reading code or data flash memory, and errors can be prevented using ECC. However, in the case of a single bit error, the ECC controller corrects it, but it is difficult to correct a double bit error, but it is possible to detect the error flag and handle it through software.

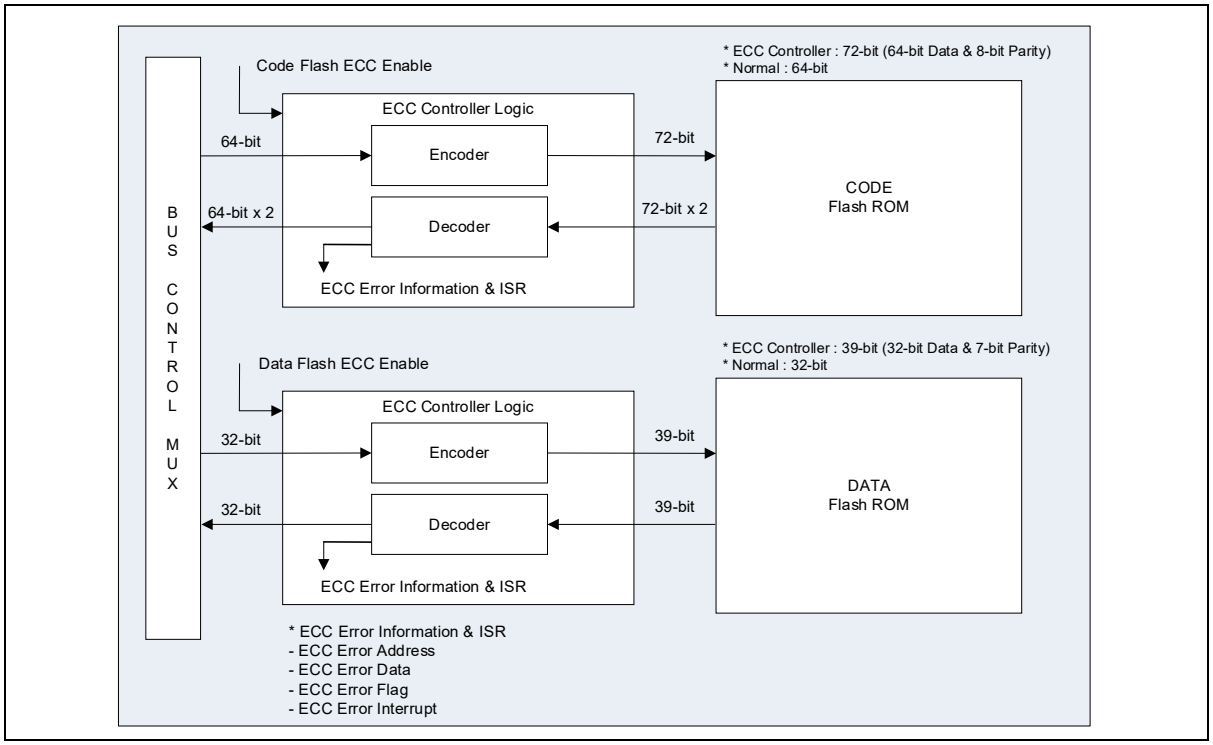
Please note that if you program in units of one byte, two bytes, etc. instead of words (four bytes), multiple ECC errors may occur. To use ECC, you must program in word units.

6.8.1 Block Diagram

When programming in code or data flash memory, 64 bits (including 8-bit parity) are stored in the code flash, and 39 bits (including 7-bit parity) are stored in the data flash via the ECC block's encoder. When reading the flash memory, the value stored in the flash memory is checked through the decoder to ensure that it does not change.

Users can check the error status in the CFMC_STAT (or DFMC_STAT) register if an ECC error occurs.

Figure 58. ECC Block Diagram



6.8.2 ECC Interrupt Service Routine

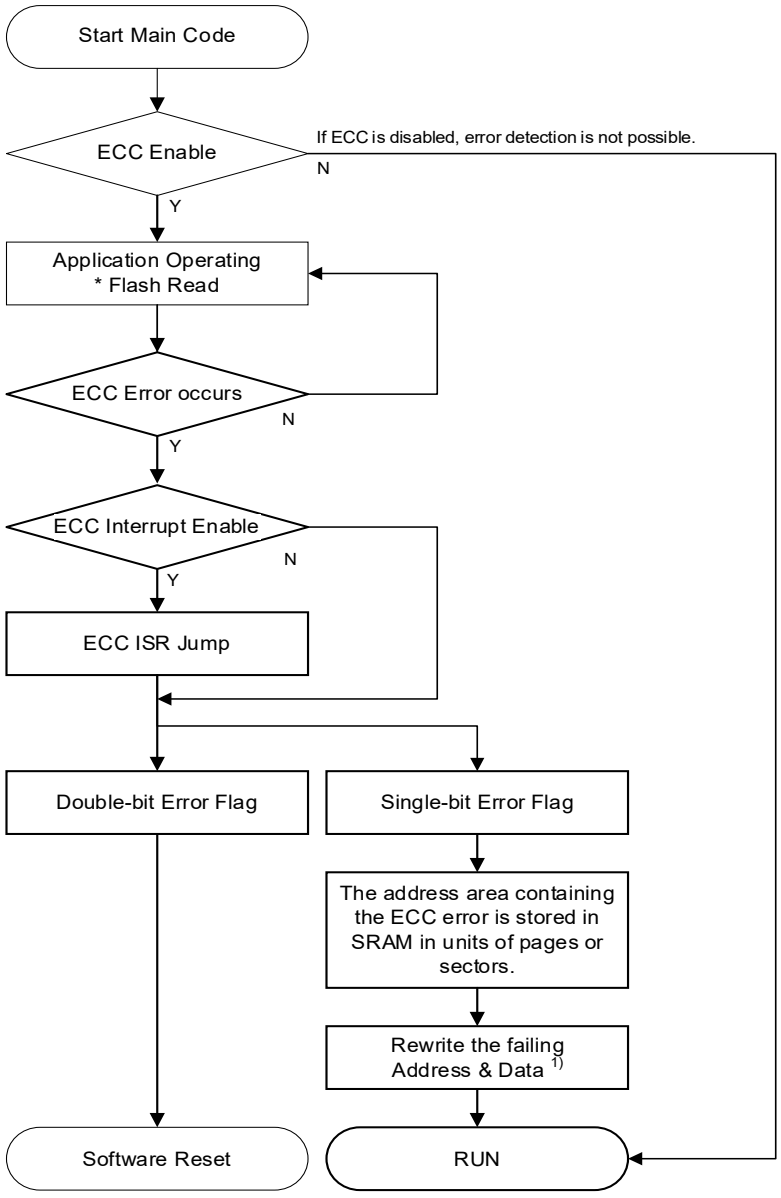
The ECC interrupt service routine provides an opportunity to take immediate action when an ECC error occurs. If an ECC error occurs when ECC interrupts are enabled, it will branch to the NMI interrupt or CFMC and DFMC interrupt routine.

In the case of single-bit error when reading the flash memory, a bit error occurs in the flash memory value, but it is automatically corrected by the ECC controller. If a single-bit error occurs, other bits in the same word may be additionally damaged later, resulting in a double-bit error. Therefore, it is recommended that the corrected data value be rewritten to the address where the ECC error occurred. When rewriting, it must be in a page or sector unit that contains the ECC error address. (See Figure 59.)

If it changes to a double-bit error, it is difficult to correct. The ECC controller can correct single-bit errors but not double-bit errors and reports the error only as an error flag.

6.8.3 Sequence of ECC

Figure 59. Sequence of ECC



NOTES:

1. If single-bit error occurs, check the ECC error address and store the page or sector in the address area in SRAM. Then, users must erase and write the address where the error occurred.
   For example, if an ECC error occurs at 0x7300, when erasing and writing by page (e.g., 1024 bytes), 0x7000 to 0x73FF must be stored in SRAM, and then SRAM stored from 0x7000 to 0x73FF must be erased and written.
2. If double-bit error occurs, it is recommended to force a shutdown because it is impossible to know what application operation is being performed at the address where the error occurred without implementing it in software.

6.9 Flash Memory Interrupts

For the code and data flash control, A34G43x supports interrupt function. This interrupt can be activated by setting the CFMC_CTRL (or DFMC_CTRL) registers. To clear the error flag bit, Write “1” to this bit.

WDIEN (Write done interrupt)

- This interrupt occurs when word write is completed. The corresponding interrupt flag is the WDONE bit of CFMC_STAT (or DFMC_STAT) register.

WAEIEN (Write address error interrupt)

- This interrupt occurs when an incorrect access is made to the write address. The corresponding interrupt flag is the WAERR bit of the CFMC_STAT register. Refer to Table 47.

ECCSI (ECC single-bit error interrupt)

- This interrupt occurs when one bit changed value is read instead of the original written value of code (or data) flash memory. The corresponding interrupt flag is the ECCSE bit of the CFMC_STAT (or DFMC_STAT) register.

ECCDN (ECC double error interrupt)

- This interrupt occurs when a value that has been changed by more than two bits, other than the original written value of code (or data) flash memory, is read. The corresponding interrupt flag is the ECCDE bit of the CFMC_STAT (or DFMC_STAT) register.

6.10 Code Flash Memory Controller Registers

The base address and register map of the Code Flash Memory Controller (CFMC) are described in the following tables.

Table 70. Base Address of CFMC

Name	Base Address
CFMC	0x4100_0000

Table 71. CFMC Register Map

Name	Offset	Type	Description	Reset value	Ref.
CFMC_CONF	0x0000	RW	Code Flash Control Register	0x0000_0000	6.10.1
CFMC_FLSKEY	0x0004	WO	Code Flash Access Key Register	0x0000_0000	6.10.2
CFMC_INFOKEY	0x0008	WO	Code Flash User Info Access Key Register	0x0000_0000	6.10.3
CFMC_FLS0PROT	0x000C	RW	Code Flash 0 Protection Register	0x0000_0000	6.10.4
CFMC_FLS1PROT	0x0010	RW	Code Flash 1 Protection Register	0x0000_0000	6.10.5
CFMC_INFOPROT	0x0014	RW	Code Flash User Info. Protection Register	0x0000_0000	6.10.6
CFMC_CTRL	0x0018	RW	Code Flash Access Control Register	0xC000_0000	6.10.7
CFMC_STAT	0x001C	RW	Code Flash Access Status Register	0x0000_0000	6.10.8
CFMC_READPROT	0x0020	RW	Code Flash Read Protection Register	0x0000_00FF	6.10.9
CFMC_PWIN	0x0024	WO	Code Flash Password Input Register	0x0000_0000	6.10.10
CFMC_CHKCTRL	0x0030	RW	Code Flash CRC Check Control Register	0x0000_0000	6.10.11
CFMC_CHKDOUT	0x0034	RO	Code Flash CRC Check Data Output Register	0x0000_FFFF	6.10.12
CFMC_CHKSADDR	0x0038	RW	Code Flash CRC Check Start Address Register	0x0000_0000	6.10.13
CFMC_CHKEADDR	0x003C	RW	Code Flash CRC Check End Address Register	0x001F_FFFF	6.10.14
CFMC_WTOUT	0x0040	RW	Code Flash Write Timeout Register	0x0000_0000	6.10.15
CFMC_BANK	0x0050	RW	Code Flash Bank Control Register	0x0000_0011	6.10.16
CFMC_BANKRST	0x0054	WO	Code Flash Bank Reset Register	0x0000_0000	6.10.17

Table 71. CFMC Register Map Table 71. CFMC Register (continued)

Name	Offset	Type	Description	Reset value	Ref.
CFMC_ECCEADDR	0x0060	RO	Code Flash ECC Error Address Register	0x0000_0000	6.10.18
CFMC_ECCEDATA0	0x0064	RO	Code Flash ECC Error Data0 Register	0x0000_0000	6.10.19
CFMC_ECCEDATA1	0x0068	RO	Code Flash ECC Error Data1 Register	0x0000_0000	6.10.20
CFMC_ECCEDATA2	0x006C	RO	Code Flash ECC Error Data2 Register	0x0000_0000	6.10.21
CFMC_ECCEDATA3	0x0070	RO	Code Flash ECC Error Data3 Register	0x0000_0000	6.10.22
CFMC_ECCEPARITY	0x0074	RO	Code Flash ECC Error Parity Register	0x0000_0000	6.10.23
CFMC_PWPRST	0x0F30	WO	Code Flash Password Preset Register	0x0000_0000	6.10.24

6.10.1 CFMC_CONF: Code Flash Control Register

The CFMC_CONF register is a control register for internal flash memory. This register is 32 bits wide.

CFMC_CONF=0x4100_0000

31 30 29 28 27 26 25 24	23 22 21 20 19 18 17 16	15 14 13 12 11 10 9	8	7 6 5 4	3 2 1 0
Reserved	WTEN	Reserved	DCEN	ICEN	Reserved	LATENCY[3:0]
-	0	-	0	0	-	0000
-	RW	-	RW	RW	-	RW

25	WTEN	Enable the code flash memory write operation.
		0 Disable
		1 Enable
9	DCEN	Code data cache enable bit
		0 Disable
		1 Enable
8	ICEN	Instruction cache enable bit
		0 Disable
		1 Enable
3 0	LATENCY[3:0](2)	Flash memory wait value bits The flash wait value between 0 and 15.

NOTES:

1. The following is how to set up a cache.
   A. Disable instruction and code data cache.
   B. Reset instruction and code data cache (auto-clear bit).
   C. Enable instruction and code data cache.
2. For detailed information on the LATENCY[3:0], refer to section 6.3.2.

6.10.2 CFMC_FLSKEY: Code Flash Access Key Register

The CFMC_FLSKEY register is the access key register for internal flash memory. This register is 32bits wide.

CFMC_FLSKEY=0x4100_0004

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
FKEY[31:0]																															
0x00000000																															
WO																															

31 0	FKEY[31:0]	These bits are the key values used to access the flash memory. FKEY[31:0] values must be written to this register in the following order to access the flash memory. (KEY1 → KEY2 → KEY3)
	KEY1	0x01234567
	KEY2	0x12345678
	KEY3	0x23456789

NOTE:

1. Flash access control bit must be disabled before controlling flash memory. To this end, the KEY values must be written in the CFMC_FLSKEY register in the order of KEY1 -> KEY2 -> KEY3. Once the flash access control bit is disabled, the access enable status is maintained until the FLOCK bit of the CFMC_CTRL register is written as '1'.

6.10.3 CFMC_INFOKEY: Code Flash User Info Access Key Register

The CFMC_INFOKEY register is the access key register for the internal flash option area. This register is 32 bits wide.

CFMC_INFOKEY=0x4100_0008

31 30 29 28 27 26 25 24	23 22 21 20 19 18 17 16	15 14 13 12 11 10 9 8	7 6 5 4 3 2 1 0
INFOKEY[31:0]			
0x00000000			
WO			

31 0	INFOKEY[31:0]	These bits are the key values used to access the User Info area. INFOKEY[31:0] values must be written to this register in the following order to access the flash option area. (KEY1 → KEY2 → KEY3)
		KEY1 0x3456789A
		KEY2 0x456789AB
		KEY3 0x56789ABC

NOTE:

1. User Info access control bit must be disabled before controlling User Info memory. To this end, the KEY values must be written in the CFMC_INFOKEY register in the order of KEY1 → KEY2 → KEY3. Once the User Info access control bit is disabled, the access enable status is maintained until the ILOCK bit of the CFMC_CTRL register is written as '1'.

6.10.4 CFMC_FLS0PROT: Code Flash 0 Protection Register

The CFMC_FLS0PROT register is an internal flash memory protection register. This register is 32 bits wide.

CFMC_FLS0PROT=0x4100_000C

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
FUP1K_7	FUP1K_6	FUP1K_5	FUP1K_4	FUP1K_3	FUP1K_2	FUP1K_1	FUP1K_0	FUBLP8K_7	FUBLP8K_6	FUBLP8K_5	FUBLP8K_4	FUBLP8K_3	FUBLP8K_2	FUBLP8K_1	FUBLP8K_0	FPBY16_15	FPBY16_14	FPBY16_13	FPBY16_12	FPBY16_11	FPBY16_10	FPBY16_9	FPBY16_8	FPBY16_7	FPBY16_6	FPBY16_5	FPBY16_4	FPBY16_3	FPBY16_2	FPBY16_1	FPBY16_0
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW

x+24	FUP1K_x (x = 0 to 7)	Unprotection of the 8 KB of Flash memory. (The settings of these bits have a higher priority level than the protection selection bits) NOTE) This bit is only meaningful if the FPBY16_15 bit is '1'.	
		Unprotection area for 8 KB flash memory Refer to Table 56 and Table 57.	
		0	Disable (protection)
		1	Enable (unprotection)
y+16	FUBLP8K_y	Protection for user boot loader area	
		0	Unprotection
		1	Protection
n	FPBY16_n (n = 0 to 15)	Flash area protection selection bits	
		0	Unprotection
		1	Protection

NOTES:

1. When any of the FPBY16_n bit or FUBLP8K_y is enabled, chip erase does not work because the protected area is included in the erase.
2. Likewise, as each unprotection sector is 1 KB, other erase commands (1 KB, 4 KB and chip erases) do not work if any of the FUP1K_x bit is '0' when the FPBY16_15 bit is '1'.
3. For detailed address and description of each area, refer to section 6.4.2.
4. During bank swap mode, write protection follows the physical address. See section 6.4.2.4.
5. FPBY16_n bit has 1/16 size of the Bank0 code flash size.

6.10.5 CFMC_FLS1PROT: Code Flash 1 Protection Register

The CFMC_FLS1PROT register is an internal flash memory protection register. This register is 32 bits wide.

CFMC_FLS1PROT=0x4100_0010

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
FUP1K_7	FUP1K_6	FUP1K_5	FUP1K_4	FUP1K_3	FUP1K_2	FUP1K_1	FUP1K_0	FUBLP8K_7	FUBLP8K_6	FUBLP8K_5	FUBLP8K_4	FUBLP8K_3	FUBLP8K_2	FUBLP8K_1	FUBLP8K_0	FPBY16_15	FPBY16_14	FPBY16_13	FPBY16_12	FPBY16_11	FPBY16_10	FPBY16_9	FPBY16_8	FPBY16_7	FPBY16_6	FPBY16_5	FPBY16_4	FPBY16_3	FPBY16_2	FPBY16_1	FPBY16_0
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW

x+24	FUP1K_x (x = 0 to 7)	Unprotection of the 8 KB of flash memory. (The settings of these bits have a higher priority level than the protection selection bits) NOTE) This bit is only meaningful if the FPBY16_15 bit is '1'.	
		Unprotection area for 8 KB flash memory Refer to Table 62 and Table 63.	
		0	Disable (protection)
		1	Enable (unprotection)
y+16	FUBLP8K_y	Protection for user boot loader area	
		0	Unprotection
		1	Protection
n	FPBY16_n (n = 0 to 15)	Flash area protection selection bits	
		0	Unprotection
		1	Protection

NOTES:

1. When any of the FPBY16_n bit or FUBLP8K_y is enabled, chip erase does not work because the protected area is included in the erase.
2. Likewise, as each unprotection sector is 1 KB, other erase commands (1 KB, 4 KB and chip erases) do not work if any of the FUP1K_x bit is '0' when the FPBY16_15 bit is '1'.
3. For detailed address and description of each area, refer to section 6.4.2.
4. During bank swap mode, write protection follows the physical address. See section 6.4.2.4.
5. FPBY16_n bit has 1/16 size of the Bank1 code flash size.

6.10.6 CFMC_INFOPROT: Code Flash User Info Protection Register

The CFMC_ INFOPROT register is an internal flash memory User Info register. This register is 32 bits wide.

CFMC_FLS1PROT=0x4100_0014

31 30 29 28 27 26 25 24	23 22 21 20 19 18 17 16	15 14 13 12 11 10 9 8	7 6 5 4	3	2	1	0
Reserved				INFO3	INFO2	INFO1	INFO0
-				0	0	0	0
-				RW	RW	RW	RW

3	INFO3	User Info3 area protection (0x0F00_0C00 ~ 0x0F00_0FFF)	
		0	Disable
		1	Enable
2	INFO2	User Info2 area protection (0x0F00_0800 ~ 0x0F00_0BFF)	
		0	Disable
		1	Enable
1	INFO1	User Info1 area protection (0x0F00_0400 ~ 0x0F00_07FF)	
		0	Disable
		1	Enable
0	INFO0	User Info0 area protection (0x0F00_0000 ~ 0x0F00_03FF)	
		0	Disable
		1	Enable

6.10.7 CFMC_CTRL: Code Flash Access Control Register

The CFMC_CTRL register is an internal flash memory access control register. This register is 32bits wide.

CFMC_CTRL=0x4100_0018

31	30	29 28 27 26	25	24	23 22 21	20	19 18 17 16	15 14 13 12	11	10	9	8	7	6 5 4	3	2	1	0
FLOCK	ILOCK	Reserved	DCRST	ICRST	Reserved	WBCLR	Reserved		ECCSI	ECCDN	WAEIEN	WDIEN	WABORT	Reserved	CERS	SERS	PERS	PGM
1	1	-	0	0	-	0	-		0	0	0	0	0	-	0	0	0	0
RW	RW	-	RW	RW	-	WO	-		RW	RW	RW	RW	RW	-	RW	RW	RW	RW

31	FLOCK[1]	Code flash access control bit Flash access control bit must be disabled before controlling flash memory. To this end, the KEY values must be written in the CFMC_FLSKEY register in the order of KEY1→KEY2→KEY3. Once the flash access control bit is disabled, the access enable status is maintained until the FLOCK bit of the CFMC_CTRL register is written as '1'.
	0	Disable (unlock status)
	1	Enable (lock status and flash memory access lock)
30	ILOCK	User Info access control bit Flash access control bit must be disabled before controlling flash memory. To this end, the KEY values must be written in the CFMC_INFOKEY register in the order of KEY1→KEY2→KEY3. Once the flash access control bit is disabled, the access enable status is maintained until the ILOCK bit of the CFMC_CTRL register is written as '1'.
	0	Disable (unlock status)
	1	Enable (lock status and flash memory access lock)
25	DCRST	Code data cache reset control bit (auto clear)
	0	No effect
	1	Reset
24	ICRST	Instruction cache reset control bit (auto clear)
	0	No effect
	1	Reset
20	WBCLR	Write buffer clear enable (auto clear)
	0	No effect
	1	Enable
11	ECCSI	ECC single error interrupt enable
	0	Disable
	1	Enable
10	ECCDN	ECC double error non-maskable interrupt enable
	0	Disable
	1	Enable
9	WAEIEN	Write address error interrupt enable
	0	Disable
	1	Enable
8	WDIEN[2]	Write done interrupt enable
	0	Disable
	1	Enable
7	WABORT	Abort write operation

		0	No effect
		1	Abort write operation.
3	CERS	Whole flash memory erase (bulk, chip)	
		0	Disable
		1	Enable
2	SERS	Flash memory erase in 4 KB sector units	
		0	Disable
		1	Enable
1	PERS	Flash memory erase in 1 KB page units	
		0	Disable
		1	Enable
0	PGM	Flash memory write in 2-word (8 bytes) units	
		0	Disable
		1	Enable

NOTES:

1. To disable the code flash or the User Info access control bit, the correct key values must be entered in the correct order by using the CFMC_FLSKEY or CFMC_INFOKEY register, respectively.
2. For detailed information of the write done interrupt, refer to section 6.9.

6.10.8 CFMC_STAT: Code Flash Access Status Register

The CFMC_STAT is an internal flash memory access status register. This register is 32 bits wide.

CFMC_STAT=0x4100_001C

31 30 29 28 27 26 25 24	23	22	21	20	19	18	17	16	15 14 13 12	11	10	9	8	7 6 5 4 3 2	1	0
Reserved	Reserved	WTERR	RPERR	WSERR	IPERR	FPERR	ILERR	FLERR	Reserved	ECCSE	ECCDE	CDONE	WDONE	Reserved	CBUSY	WBUSY
-	-	0	0	0	0	0	0	0	-	0	0	0	0	-	0	0
-	-	RWC1	RWC1	RWC1	RWC1	RWC1	RWC1	RWC1	-	RWC1	RWC1	RWC1	RWC1	-	RO	RO

24	WAERR	Write address error	
		0	Not error
		1	Error detection (cleared by writing a ‘1’)
22	WTERR	Write time out error	
		0	Not error
		1	Error detection (cleared by writing a ‘1’)
21	RPERR	Read-protect error	
		0	Not error
		1	Error detection (cleared by writing a ‘1’)
20	WSERR	Write sequence error	
		0	Not error
		1	Error detection (cleared by writing a ‘1’)
19	IPERR	User Info protect error	
		0	Not error
		1	Error detection (cleared by writing a ‘1’)
18	FPERR	Flash protect error	
		0	Not error
		1	Error detection (cleared by writing a ‘1’)
17	ILERR	User Info lock error	
		0	Not error
		1	Error detection (cleared by writing a ‘1’)
16	FLERR	Flash lock error	
		0	Not error
		1	Error detection (cleared by writing a ‘1’)
11	ECCSE	Code flash ECC single error	
		0	Not error
		1	ECC single error detection (cleared by writing a ‘1’)
10	ECCDE	Code flash ECC double error	
		0	Not error
		1	ECC double error detection (cleared by writing a ‘1’)
9	CDONE	CRC check done check	
		0	Busy
		1	CRC check done (cleared by writing a ‘1’)
8	WDONE	Write done interrupt status	
		0	Busy
		1	Write done (cleared by writing a ‘1’)
1	CBUSY	CRC check busy check	
		0	Not busy
		1	Busy
0	WBUSY	Write busy check	

	0	Not busy
	1	Busy

6.10.9 CFMC_READPROT: Code Flash Read Protection Register

The CFMC_READPROT register is the internal flash memory's read protection register. It is a 32-bit register.

CFMC_READPROT=0x4100_0020

31	30	29 28 27	26	25	24	23 22 21 20 19 18	17	16	15 14 13 12 11 10	9	8	7 6 5 4 3 2 1 0
DBGMOD	SRBOOT	Reserved	PWMATCH	INFO0ERSD	CERSD	Reserved	LVL2_STS	LVL1_STS	Reserved	LVL2_EN	LVL1_EN	RPROT[7:0]
0	0	-	0	0	0	-	0	0	-	0	0	0xFF
RO	RO	-	RO	RO	RO	-	RO	RO	-	RO	RO	RW

31	DBGMOD	Debug operating status bit
		0 Not operating
		1 Operating
30	SRBOOT	SRAM Boot mode status bit
		0 Normal mode
		1 SRAM Boot mode
26	PWMATCH	Password match flag bit
		0 Not matched
		1 Password preset value and password in value are matched.
25	INFO0ERSD	Chip erase and Info0 erase done flag bit
		0 Not occurred
		1 User Info0 erase done (RPROT erase/Write permit)
24	CERSD	Chip erase done flag bit
		0 Not occurred
		1 Chip erase done (User Info0 erase/Write permit)
17	LVL2_STS	Protection Level-2 status bit in debug mode This bit can be checked in **debug mode only**.
		0 Normal status
		1 Protection Level-2 status
16	LVL1_STS	Protection Level-1 status bit in debug mode This bit can be checked in **debug mode only**.
		0 Normal status
		1 Protection Level-1 status
9	LVL2_EN	Protection Level-2 status bit
		0 Normal status
		1 Protection Level-2 status
8	LVL1_EN	Protection Level-1 status bit
		0 Normal status
		1 Protection Level-1 status
7 0	RPROT[7:0]	Read protection control bit When the memory is in either protection mode, setting the 8th bit to '1' enables the password function for the protection mode. 0xB9: Protection 1 password mode 0x80: Protection 2 password mode
		0xFF Unprotection (UNPROT)
		0x39 Protection Level-1 (LVL1)
		others Protection Level-2 (LVL2)

NOTE: For detailed information of the Read Protection, refer to 6.4.1.

6.10.10 CFMC_PWIN: Code Flash Password Input Register

The CFMC_PWIN register is used to enter the password value, while the microcontroller is in read protection password mode. If the password value matches the value fed to the CFMC_PWPRST register, the protected data can be read.

CFMC_PWIN=0x4100_0024

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
PWIN[31:0]																															
0x00000000																															
WO																															

31 0	PWIN[31:0]	Password input data bit Writing is done twice[1] to PWIN[31:0] in the order of 'LSB → MSB'. Rewriting is restricted once the register is written.

NOTES:

1. Example 1) Password value is set to 0x1234_5678, 0x8765_4321
   CFMC_PWIN = 0x12345678;
   CFMC_PWIN = 0x87654321;

2. Example 2) Read protection password mode is set again when a user consecutively enters the same password value two times after the protection mode was released due to the password match in read protection password mode.
   CFMC_PWIN = 0x12345678;
   CFMC_PWIN = 0x87654321; // Protection mode is released due to password match.
   CFMC_PWIN = 0x12345678;
   CFMC_PWIN = 0x87654321; // Read protection password mode is set again.

3. Example 3) If a password that was written first is not correct, users must reset before writing a password again.
   CFMC_PWIN = 0x11111111; // Wrong password value is input.
   CFMC_PWIN = 0x12345678;
   CFMC_PWIN = 0x87654321; // Password cannot disable the protection.

6.10.11 CFMC_CHKCTRL: Code Flash CRC Check Control Register

The CFMC_CHKCTRL register is an internal flash memory CRC control register. This register is 32 bits wide.

The internal CRC check function can operate while the read protection is applied.

CFMC_CHKCTRL=0x4100_0030

31 30 29 28 27 26 25 24	23 22 21 20 19 18 17 16	15 14 13 12 11 10 9 8	7 6 5 4 3 2 1 0
Reserved	CDRST	Reserved	CDIEN	Reserved	BSTEN	BGEN
-	0	-	0	-	0	0
-	WO	-	RW	-	WO	WO

16	CDRST	CRC data reset (auto-clear)	
		0	No effect
		1	Reset
8	CDIEN	CRC done interrupt	
		0	Disable
		1	Enable
1	BSTEN	Burst mode (auto-clear)	
		0	Disable
		1	Enable
0	BGEN	Background mode (auto-clear)	
		0	Disable
		1	Enable

NOTES:

1. 16-bit specific Polynomial (internal use only)

6.10.12 CFMC_CHKDOUT: Code Flash CRC Check Data Output Register

The CFMC_CHKDOUT register is an internal flash memory CRC data output register. This register is 32 bits wide.

CFMC_CHKDOUT=0x4100_0034

31 30 29 28 27 26 25 24	23 22 21 20 19 18 17 16	15 14 13 12 11 10 9 8	7 6 5 4 3 2 1 0
Reserved		CDOUT[15:0]	
-		0xFFFF	
-		RO	

15 0	CDOUT[15:0]	CRC data output value

6.10.13 CFMC_CHKSADDR: Code Flash CRC Check Start Address Register

The CFMC_CHKSADDR register is an internal flash memory CRC start address register. This register is 32 bits wide.

CFMC_CHKSADDR=0x4100_0038

31 30 29 28 27 26 25 24	23 22 21 20 19 18 17 16	15 14 13 12 11 10 9 8	7 6 5 4 3 2 1 0
SADDR[23:0]			FIXED_VALUE[7:0]
0x0000000			0x00
RW			RO

31 8	SADDR [23:0]	CRC start address value
7 0	FIXED_VALUE[7:0]	Fixed value

6.10.14 CFMC_CHKEADDR: Code Flash CRC Check End Address Register

The CFMC_CHKEADDR register is an internal flash memory CRC end address register. This register is 32 bits wide.

CFMC_CHKEADDR=0x4100_003C

31 30 29 28 27 26 25 24	23 22 21 20 19 18 17 16	15 14 13 12 11 10 9 8	7 6 5 4 3 2 1 0
EADDR[23:0]			FIXED VALUE[7:0]
0x001FFF			0xFF
RW			RO

31 8	EADDR[23:0]	CRC end address value
7 0	FIXED_VALUE[7:0]	Fixed value

6.10.15 CFMC_WTOUT: Code Flash Write Timeout Register

The CFMC_WTOUT register is an internal flash memory Write Timeout register. This register is 32 bits wide. After the Write operation (Program, Erase), if the Write operation does not end during $t_{Timeout}$, the Write operation is terminated.

CFMC_WTOUT=0x4100_0040

31 30 29 28 27 26 25 24	23 22 21 20 19 18 17 16	15 14 13 12 11 10 9 8	7 6 5 4 3 2 1 0
WTOUT[31:0]			
0x0000_0000			
RW			

31 0	WTOUT[31:0]	Write timeout count	
		0	Disables.
		1	Enables. $t_{Timeout}$ = WTOUT / HCLK

6.10.16 CFMC_BANK: Code Flash Bank Control Register

The CFMC_BANK register is an internal flash memory bank control register. This register is 32 bits wide. This register must be accessed from BootROM or SRAM.

CFMC_BANK=0x4100_0050

31 30 29 28 27 26 25 24	23 22 21 20 19 18 17 16	15 14 13 12 11 10 9 8	7 6 5	4	3 2 1	0
BANK_WTIDKY[23:0]			Reserved	BS	Reserved	BM
0x000000			-	1	-	1
RW			-	RW	-	RW

31 8	BANK_WTIDKY	Bank register Identification key To control a bank, you must write to BS bit, BM bit and this ID key to CFMC_BANK register. Writing to a BS bit and BM without writing to BANK_WTIDKY[23:0] is prohibited.	
		0x5A3C0F	ID Key
		Other	No effect
1	BS	Bank swap (valid in bank mode)	
		0	Swap mode
		1	No swap mode
0	BM	Bank mode	
		0	Bank mode
		1	Linear mode

6.10.17 CFMC_BANKRST: Code Flash Bank Reset Register

The CFMC_BANK register is an internal flash memory bank reset register. This register is 32 bits wide.

CFMC_BANKRST=0x4100_0054

31 30 29 28 27 26 25 24	23 22 21 20 19 18 17 16	15 14 13 12 11 10 9 8	7 6 5 4 3 2 1	0
BANKRST_WTIDKY[23:0]			Reserved	BANKRST
0x000000			-	0
WO			-	WO

31 8	BANKRST_WTIDKY	Bank reset register Identification key To control a bank reset, this ID key to CFMC_BANKRST register. Writing to a BANKRST without writing to BANKRST_WTIDKY[23:0] is prohibited.	
		0x5A3C0F	ID Key
		Other	No effect
0	BANKRST	Reset for bank swap (Check BANKRST in SCU_RSER for reset)	
		0	No effect
		1	Reset

6.10.18 CFMC_ECCEADDR: Code Flash ECC Error Address Register

The CFMC_ECCEADDR register is an internal flash memory ECC error address register. This register is 32 bits wide.

CFMC_ECCEADDR=0x4100_0060

31 30 29 28 27 26 25 24	23 22 21 20 19 18 17 16	15 14 13 12 11 10 9 8	7 6 5 4 3 2 1 0
ECCEA[31:0]			
0x0000_0000			
RO			

31 0	ECCEA[31:0]	ECC error address	

6.10.19 CFMC_ECCEDATA0: Code Flash ECC Error Data0 Register

The CFMC_ECCEDATA0 register is an internal flash memory ECC data0 register. This register is 32 bits wide.

CFMC_ECCEDATA0=0x4100_0064

31 30 29 28 27 26 25 24	23 22 21 20 19 18 17 16	15 14 13 12 11 10 9 8	7 6 5 4 3 2 1 0
ECCED0[31:0]			
0x0000_0000			
R0			

31 0	ECCED0[31:0]	ECC error data0 of 128 bits [31:0] ECC error data0 address: ECC error address + 0x0

6.10.20 CFMC_ECCEDATA1: Code Flash ECC Error Data1 Register

The CFMC_ECCEDATA1 register is an internal flash memory ECC data1 register. This register is 32 bits wide.

CFMC_ECCEDATA1=0x4100_0068

31 30 29 28 27 26 25 24	23 22 21 20 19 18 17 16	15 14 13 12 11 10 9 8	7 6 5 4 3 2 1 0
ECCED1[31:0]			
0x0000_0000			
R0			

31 0	ECCED1[31:0]	ECC error data1 of 128 bits [63:32] ECC error data1 address: ECC error address + 0x4

6.10.21 CFMC_ECCEDATA2: Code Flash ECC Error Data2 Register

The CFMC_ECCEDATA2 register is an internal flash memory ECC data2 register. This register is 32 bits wide.

CFMC_ECCEDATA2=0x4100_006C

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
ECCED2[31:0]																															
0x0000_0000																															
R0																															

31 0	ECCED2[31:0]	ECC error data2 of 128 bits [95:64] ECC error data2 address: ECC error address + 0x8

6.10.22 CFMC_ECCEDATA3: Code Flash ECC Error Data3 Register

The CFMC_ECCEDATA3 register is an internal flash memory ECC data3 register. This register is 32 bits wide.

CFMC_ECCEDATA3=0x4100_0070

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
ECCED3[31:0]																															
0x0000_0000																															
R0																															

31 0	ECCED3[31:0]	ECC error data3 of 128 bits [127:96] ECC error data3 address: ECC error address + 0xC

6.10.23 CFMC_ECCEPARITY: Code Flash ECC Error Parity Register

The CFMC_ECCEPARITY register is an internal flash memory ECC parity register. This register is 32 bits wide.

CFMC_ECCEPARTIY=0x4100_0074

31 30 29 28 27 26 25 24	23 22 21 20 19 18 17 16	15 14 13 12 11 10 9 8	7 6 5 4 3 2 1 0
Reserved		ECCEP1[7:0]	ECCEP0[7:0]
-		0x00	0x00
-		R0	R0

15 8	ECCEP1[7:0]	ECC error parity 1 of 128 bits
7 0	ECCEP0[7:0]	ECC error parity 0 of 128 bits

6.10.24 CFMC_PWPRST: Code Flash Password Preset Register

The CFMC_PWPRST register is used to preset the password value. This register is 32 bits wide.

CFMC_PWPRST=0x4100_0F30

31 30 29 28 27 26 25 24	23 22 21 20 19 18 17 16	15 14 13 12 11 10 9 8	7 6 5 4 3 2 1 0
PWPRST[31:0]			
0x00000000			
WO			

31 0	PWPRST[31:0]	Password preset data bit Writing is done twice to the register in the order of ‘LSB → MSB’. Rewriting is restricted once the register is written. For example, presetting 0x1234_5678 as a password. CFMC_PWPRST = 0x1234_5678 CFMC_PWPRST = 0x1234_5678

6.10.25 CFMC Register Map Summary

Table 72. CFMC Register Map Summary

Offset	Register name	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0x00	CFMC_CONF	Res.	Res.	Res.	Res.	Res.	Res.	WTEN	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	DCEN	ICEN	Res.	Res.	Res.	Res.	LATENCY[3:0]			
	Reset value							0																0	0					0	0	0	0
0x04	CFMC_FLSKEY	FKEY[31:0]																															
	Reset value	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
0x08	CFMC_INFOKEY	INFOKEY[31:0]																															
	Reset value	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
0x0C	CFMC_FLS0PROT	FUP1K_7	FUP1K_6	FUP1K_5	FUP1K_4	FUP1K_3	FUP1K_2	FUP1K_1	FUP1K_0	FUBLP8K_7	FUBLP8K_6	FUBLP8K_5	FUBLP8K_4	FUBLP8K_3	FUBLP8K_2	FUBLP8K_1	FUBLP8K_0	FPBY16_15	FPBY16_14	FPBY16_13	FPBY16_12	FPBY16_11	FPBY16_10	FPBY16_9	FPBY16_8	FPBY16_7	FPBY16_6	FPBY16_5	FPBY16_4	FPBY16_3	FPBY16_2	FPBY16_1	FPBY16_0
	Reset value	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
0x10	CFMC_FLS1PROT	FUP1K_7	FUP1K_6	FUP1K_5	FUP1K_4	FUP1K_3	FUP1K_2	FUP1K_1	FUP1K_0	FUBLP8K_7	FUBLP8K_6	FUBLP8K_5	FUBLP8K_4	FUBLP8K_3	FUBLP8K_2	FUBLP8K_1	FUBLP8K_0	FPBY16_15	FPBY16_14	FPBY16_13	FPBY16_12	FPBY16_11	FPBY16_10	FPBY16_9	FPBY16_8	FPBY16_7	FPBY16_6	FPBY16_5	FPBY16_4	FPBY16_3	FPBY16_2	FPBY16_1	FPBY16_0
	Reset value	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
0x14	CFMC_INFOPROT	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	INFO3	INFO2	INFO1	INFO0
	Reset value																													0	0	0	0
0x18	CFMC_CTRL	FLOCK	ILOCK	Res.	Res.	Res.	Res.	DCRST	ICRST	Res.	Res.	Res.	WBCLR	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	ECCSI	ECCDN	WAEIEN	WDIEN	WABORT	Res.	Res.	Res.	CERS	SERS	PERS	PGM
	Reset value	1	1					0	0				0									0	0	0	0	0				0	0	0	0
0x1C	CFMC_STAT	Res.	Res.	Res.	Res.	Res.	Res.	Res.	WAERR	Res.	WTERR	RPERR	WSERR	IPERR	FPERR	ILERR	FLERR	Res.	Res.	Res.	Res.	ECCSE	ECCDE	CDONE	WDONE	Res.	Res.	Res.	Res.	Res.	Res.	CBUSY	WBUSY
	Reset value								0		0	0	0	0	0	0	0					0	0	0	0							0	0
0x20	CFMC_READPROT	DBGMOD	SRBOOT	Res.	Res.	Res.	PWMATCH	INFO0ERSD	CERSD	Res.	Res.	Res.	Res.	Res.	Res.	LVL2_STS	LVL1_STS	Res.	Res.	Res.	Res.	Res.	Res.	LVL2_EN	LVL1_EN	RPROT[7:0]							
	Reset value	0	0				0	0	0							0	0							0	0	1	1	1	1	1	1	1	1
0x24	CFMC_PWIN	PWIN[31:0]																															
	Reset value	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
0x30	CFMC_CHKCTRL	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	CDRST	Res.	Res.	Res.	Res.	Res.	Res.	Res.	CDIEN	Res.	Res.	Res.	Res.	Res.	Res.	BSTEN	BGEN
	Reset value																0								0							0	0
0x34	CFMC_CHKDOUT	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	CDOUT[15:0]															
	Reset value																	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1

Table 72. CFMC Register Map Summary (continued)

Offset	Register name	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0x38	CFMC_CHKSADDR	SADDR[23:0]																								FIXED_VALUE[7:0]							
	Reset value	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
0x3C	CFMC_CHKEADDR	EADDR[23:0]																								FIXED_VALUE[7:0]							
	Reset value	0	0	0	0	0	0	0	0	0	0	0	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1
0x40	CFMC_WTOUT	WTOUT[31:0]																															
	Reset value	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
0x50	CFMC_BANK	BANK_WTIDKY[23:0]																								Res.	Res.	Res.	BS	Res.	Res.	Res.	BM
	Reset value	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0				0				0
0x54	CFMC_BANKRST	BANKRST_WTIDKY[23:0]																								Res.	Res.	Res.	Res.	Res.	Res.	Res.	BANKRST
	Reset value	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
0x60	CFMC_ECCEADDR	ECCEA[31:0]																															
	Reset value	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
0x64	CFMC_ECCEDATA0	ECCED0[31:0]																															
	Reset value	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
0x68	CFMC_ECCEDATA1	ECCED1[31:0]																															
	Reset value	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
0x6C	CFMC_ECCEDATA2	ECCED2[31:0]																															
	Reset value	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
0x70	CFMC_ECCEDATA3	ECCED3[31:0]																															
	Reset value	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
0x74	CFMC_ECCEPARITY	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	ECCEP1[7:0]								ECCEP0[7:0]							
	Reset value																	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
0xF30	CFMC_PWPRST	PWPRST[31:0]																															
	Reset value	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

6.11 Data Flash Registers

The base address and register map of the Data Flash Memory Controller (DFMC) are described in the following tables.

Table 73. Base Address of DFMC

Name	Base address
DFMC	0x4100_1000

Table 74. DFMC Register Map

Name	Offset	Type	Description	Reset value	Ref.
DFMC_CONF	0x0000	RW	Data Flash Control Register	0x0000_0000	6.11.1
DFMC_FLSKEY	0x0004	WO	Data Flash Access Key Register	0x0000_0000	6.11.2
DFMC_FLSPROT	0x000C	RW	Data Flash Protection Register	0x0000_0000	6.11.3
DFMC_CTRL	0x0014	RW	Data Flash Access Control Data	0x8000_0000	6.11.4
DFMC_STAT	0x0018	RW	Data Flash Access Status Register	0x0000_0000	6.11.5
DFMC_CHKCTRL	0x0030	RW	Data Flash CRC Check Control Register	0x0000_0000	6.11.6
DFMC_CHKDOUT	0x0034	RW	Data Flash CRC Check Data Output Register	0x0000_FFFF	6.11.7
DFMC_CHKSADDR	0x0038	RW	Data Flash CRC Check Start Address Register	0x0000_0000	6.11.8
DFMC_CHKEADDR	0x003C	RW	Data Flash CRC Check End Address Register	0x0000_7FFF	6.11.9
DFMC_WTOUT	0x0040	RW	Data Flash Write Timeout Register	0x0000_0000	6.11.10
DFMC_ECCEADDR	0x0050	RO	Data Flash ECC Error Address Register	0x0000_0000	6.11.11
DFMC_ECCEDATA	0x0054	RO	Data Flash ECC Error Data Register	0x0000_0000	6.11.12
DFMC_ECCEPARITY	0x0058	RO	Data Flash ECC Error Parity Register	0x0000_0000	6.11.13

6.11.1 DFMC_CONF: Data Flash Control Register

The DFMC_CONF register is a control register for the internal data flash memory. This register is 32 bits wide.

DFMC_CONF=0x4100_1000

31 30 29 28 27 26 25 24	23 22 21 20 19 18 17 16	15 14 13 12 11 10 9 8	7 6 5 4 3 2 1 0
Reserved	WTEN	Reserved	LATENCY[3:0]
-	0	-	0000
-	RW	-	RW

25	WTEN	Enable the data flash memory write operation
		0 Disable
		1 Enable
3 0	LATENCY[3:0](2)	Flash memory wait value bits
		The flash memory wait value between 0 and 15.

NOTE:

1. For detailed information on the LATENCY[3:0], refer to section 6.3.2.

6.11.2 DFMC_FLSKEY: Data Flash Access Key Register

The DFMC_FLSKEY register is the access key register for internal flash memory. This register is 32 bits wide.

DFMC_FLSKEY=0x4100_1004

31 30 29 28 27 26 25 24	23 22 21 20 19 18 17 16	15 14 13 12 11 10 9 8	7 6 5 4 3 2 1 0
FKEY[31:0]			
0x00000000			
WO			

31 0	FKEY[31:0](1)	These bits are the key values used to access the flash memory. FKEY[31:0] values must be written to this register in the following order to access the flash memory. (KEY1 → KEY2 → KEY3)
		KEY1 0x01234567
		KEY2 0x12345678
		KEY3 0x23456789

NOTE:

1. Flash lock must be disabled before controlling flash memory. To this end, the KEY values must be written in the DFMC_FLSKEY register in the order of KEY1 → KEY2 → KEY3. Once the flash lock is disabled, the unlock status is maintained until the FLOCK bit of the DFMC_CTRL register is written as '1'.

6.11.3 DFMC_FLSPROT: Data Flash Protection Register

The DFMC_FLSPROT register is an internal flash memory protection register. This register is 32 bits wide.

DFMC_FLSPROT=0x4100_100C

31	30	29	28	27	26	25	24	23 22 21 20 19 18 17 16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
FUP512B_7	FUP512B_6	FUP512B_5	FUP512B_4	FUP512B_3	FUP512B_2	FUP512B_1	FUP512B_0	Reserved	FPBY2K_15	FPBY2K_14	FPBY2K_13	FPBY2K_12	FPBY2K_11	FPBY2K_10	FPBY2K_9	FPBY2K_8	FPBY2K_7	FPBY2K_6	FPBY2K_5	FPBY2K_4	FPBY2K_3	FPBY2K_2	FPBY2K_1	FPBY2K_0
0	0	0	0	0	0	0	0	-	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
RW	RW	RW	RW	RW	RW	RW	RW	-	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW

x+24	FUP512B_x (x = 0 to 7)	Unprotection of the 4 KB of flash memory (0x0E00_7000 ~ 0x0E00_7FFF) (The settings of these bits have a higher priority level than the protection selection bits) NOTE: This bit is only meaningful if the FPBY2K_15 bit and FPBY2K_14 is '1'.
		Unprotection area for 512 KB flash memory x = 0: 0x0E00_7000 to 0x0E00_71FF x = 1: 0x0E00_7200 to 0x0E00_73FF x = 2: 0x0E00_7400 to 0x0E00_75FF x = 3: 0x0E00_7600 to 0x0E00_77FF x = 4: 0x0E00_7800 to 0x0E00_79FF x = 5: 0x0E00_7A00 to 0x0E00_7BFF x = 6: 0x0E00_7C00 to 0x0E00_7DFF x = 7: 0x0E00_7E00 to 0x0E00_7FFF
		0 Protection.
		1 Unprotection.
n	FPBY2K_n (n = 0 to 15)	Flash area protection selection bit n = 0 to 15 values for 32 KB flash memory
		n = 0: 0x0E00_0000 to 0x0E00_07FF n = 1: 0x0E00_0800 to 0x0E00_0FFF n = 2: 0x0E00_1000 to 0x0E00_17FF n = 3: 0x0E00_1800 to 0x0E00_1FFF n = 4: 0x0E00_2000 to 0x0E00_27FF n = 5: 0x0E00_2800 to 0x0E00_2FFF n = 6: 0x0E00_3000 to 0x0E00_37FF n = 7: 0x0E00_3800 to 0x0E00_3FFF n = 8: 0x0E00_4000 to 0x0E00_47FF n = 9: 0x0E00_4800 to 0x0E00_4FFF n = 10: 0x0E00_5000 to 0x0E00_57FF n = 11: 0x0E00_5800 to 0x0E00_5FFF n = 12: 0x0E00_6000 to 0x0E00_67FF n = 13: 0x0E00_6800 to 0x0E00_6FFF n = 14: 0x0E00_7000 to 0x0E00_77FF n = 15: 0x0E00_7800 to 0x0E00_7FFF
		0 Unprotection
		1 Protection

NOTES:

1. When any of the FPBY2K_n bit is enabled, chip erase does not work because the protected area is included in the erase.
2. Likewise, as each unprotection sector is 512 bytes, other erase commands (2 KB and full chip erases) do not work if any of the FUP512B_x bit is '0' when the FPBY2K_15 bit is '1'.
3. For detailed address and description of each area, refer to section 6.4.2.

6.11.4 DFMC_CTRL: Data Flash Access Control Register

The DFMC_CTRL register is an internal flash memory access control register. This register is 32 bits wide.

DFMC_CTRL=0x4100_1014

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
FLOCK	Reserved																			ECCSI	ECCDN	Reserved	WDIEN	WABORT	Reserved					CERS	SERS	PERS	PGM
1	-																			0	0	-	0	0	-					0	0	0	0
RW	-																			RW	RW	-	RW	RW	-					RW	RW	RW	RW

31	FLOCK[1]	Data flash memory access control bit Flash access control bit must be disabled before controlling flash memory. To this end, the KEY values must be written in the DFMC_FLSKEY register in the order of KEY1 -> KEY2 -> KEY3. Once the flash access control bit is disabled, the access enable status is maintained until the FLOCK bit of the DFMC_CTRL register is written as '1'.
		0 Disable Access
		1 Enable Access
11	ECCSI	ECC single error interrupt enable
		0 Disable
		1 Enable
10	ECCDN	ECC double error non-maskable interrupt enable
		0 Disable
		1 Enable
8	WDIEN[2]	Write done interrupt enable
		0 Disable
		1 Enable
7	WABORT	Abort write operation
		0 No effect
		1 Abort write operation
3	CERS	Whole flash memory erase
		0 Disable
		1 Enable
2	SERS	Flash memory erase in 2 KB sector units
		0 Disable
		1 Enable
1	PERS	Flash memory erase in 512 bytes page units
		0 Disable
		1 Enable
0	PGM	Flash memory write in word (4 bytes) units
		0 Disable
		1 Enable

NOTES:

1. To disable data flash memory access, the correct key values must be entered in the correct order by using the DFMC_FLSKEY register, respectively.
2. For detailed information of the write-done interrupt, refer to section 6.9.

6.11.5 DFMC_STAT: Data Flash Access Status Register

The DFMC_STAT is an internal flash memory access status register. This register is 32 bits wide.

DFMC_STAT=0x4100_1018

31 30 29 28 27 26 25 24	23	22	21	20	19	18	17	16	15 14 13 12	11	10	9	8	7 6 5 4 3 2	1	0
Reserved		WTERR	RPERR	WSERR	Reserved	FPERR	Reserved	FLERR	Reserved	ECCSE	ECCDE	CDONE	WDONE	Reserved	CBUSY	WBUSY
-		0	0	0	-	0	-	0	-	0	0	0	0	-	0	0
-		RWC1	RWC1	RWC1	-	RWC1	-	RWC1	-	RWC1	RWC1	RWC1	RWC1	-	RO	RO

22	WTERR	Write time out error	
		0	Not error
		1	Error detection (cleared by writing a ‘1’)
21	RPERR	Read-protect error	
		0	Not error
		1	Error detection (cleared by writing a ‘1’)
20	WSERR	Write sequence error	
		0	Not error
		1	Error detection (cleared by writing a ‘1’)
18	FPERR	Flash protect error	
		0	Not error
		1	Error detection (cleared by writing a ‘1’)
16	FLERR	Flash lock error	
		0	Not error
		1	Error detection (cleared by writing a ‘1’)
11	ECCSE	Code flash ECC single error	
		0	Not error
		1	ECC single error detection (cleared by writing a ‘1’)
10	ECCDE	Code flash ECC double error	
		0	Not error
		1	ECC double error detection (cleared by writing a ‘1’)
9	CDONE	CRC check done check	
		0	Busy
		1	CRC check done (cleared by writing a ‘1’)
8	WDONE	Write done interrupt status	
		0	Busy
		1	Write done (cleared by writing a ‘1’)
1	CBUSY	CRC check busy check	
		0	Not busy
		1	Busy
0	WBUSY	Write busy check	
		0	Not busy
		1	Busy

6.11.6 DFMC_CHKCTRL: Code Flash CRC Check Control Register

The DFMC_CHKCTRL register is an internal flash memory CRC control register. This register is 32 bits wide.

The internal CRC check function can operate while the read protection is applied.

DFMC_CHKCTRL=0x4100_1030

31 30 29 28 27 26 25 24	23 22 21 20 19 18 17 16	15 14 13 12 11 10 9 8	7 6 5 4 3 2 1 0		
Reserved	CDRST	Reserved	CDIEN	Reserved	BSTEN	BGEN
-	0	-	0	-	0	0
-	WO	-	RW	-	WO	WO

16	CDRST	CRC data reset (auto-clear)
	0	No effect
	1	Reset
8	CDIEN	CRC done interrupt
	0	Disable
	1	Enable
1	BSTEN	Burst mode (auto-clear)
	0	Disable
	1	Enable
0	BGEN	Background mode (auto-clear)
	0	Disable
	1	Enable

NOTES:

1. 16-bit specific Polynomial (internal use only)

6.11.7 DFMC_CHKDOUT: Data Flash CRC Check Data Output Register

The DFMC_CHKDOUT register is an internal flash memory CRC data output register. This register is 32 bits wide.

DFMC_CHKDOUT=0x4100_1034

31 30 29 28 27 26 25 24	23 22 21 20 19 18 17 16	15 14 13 12 11 10 9 8	7 6 5 4 3 2 1 0
Reserved		CDOUT[15:0]	
-		0xFFFF	
-		R0	

15 0	CDOUT[15:0]	CRC data output value

6.11.8 DFMC_CHKSADDR: Data Flash CRC Check Start Address Register

The DFMC_CHKSADDR register is an internal flash memory CRC start address register. This register is 32 bits wide.

DFMC_CHKSADDR=0x4100_1038

31 30 29 28 27 26 25 24	23 22 21 20 19 18 17 16	15 14 13 12 11 10 9 8	7 6 5 4 3 2 1 0
SADDR[25:0]			FIXED_VALUE[5:0]
0x0000000			0x00
RW			RO

31 6	SADDR[25:0]	CRC start address value (physical address) Address range: 0x000 ~ 0x1FF
5 0	FIXED_VALUE[5:0]	Fixed value

6.11.9 DFMC_CHKEADDR: Data Flash CRC Check End Address Register

The DFMC_CHKEADDR register is an internal flash memory CRC end address register. This register is 32 bits wide.

DFMC_CHKEADDR=0x4100_103C

31 30 29 28 27 26 25 24	23 22 21 20 19 18 17 16	15 14 13 12 11 10 9 8	7 6 5 4 3 2 1 0
EADDR[25:0]			FIXED_VALUE[5:0]
0x00001FF			0x3F
RW			RO

31 6	EADDR[25:0]	CRC end address value (physical address) Address range: 0x000 ~ 0x1FF
5 0	FIXED_VALUE[5:0]	Fixed value

6.11.10 DFMC_WTOUT: Data Flash Write Timeout Register

The DFMC_WTOUT register is an internal flash memory Write Timeout register. This register is 32 bits wide. After the Write operation (Program, Erase), if the Write operation does not end during $t_{Timeout}$, the write operation is terminated.

DFMC_WTOUT=0x4100_1040

31 30 29 28 27 26 25 24	23 22 21 20 19 18 17 16	15 14 13 12 11 10 9 8	7 6 5 4 3 2 1 0
WTOUT[31:0]			
0x0000_0000			
RW			

31 0	WTOUT[31:0]	Write timeout count
		0 Disable
		1 Enable $t_{Timeout}$ = WTOUT / HCLK

6.11.11 DFMC_ECCEADDR: Data Flash ECC Error Address Register

The DFMC_ECCEADDR register is an internal flash memory ECC error address register. This register is 32 bits wide.

DFMC_ECCEADDR=0x4100_1050

31 30 29 28 27 26 25 24	23 22 21 20 19 18 17 16	15 14 13 12 11 10 9 8	7 6 5 4 3 2 1 0
ECCEA[31:0]			
0x0000_0000			
RO			

31 0	ECCEA[31:0]	ECC error address (Offset: 0x0E00_0000)

6.11.12 DFMC_ECCEDATA: Data Flash ECC Error Data Register

The DFMC_ECCEDATA register is an internal flash memory ECC data register. This register is 32 bits wide.

DFMC_ECCEDATA=0x4100_1054

31 30 29 28 27 26 25 24	23 22 21 20 19 18 17 16	15 14 13 12 11 10 9 8	7 6 5 4 3 2 1 0
ECCED[31:0]			
0x00000000			
R0			

31 0	ECCED[31:0]	ECC error data

6.11.13 DFMC_ECCEPARITY: Data Flash ECC Error Parity Register

The DFMC_ECCEPARITY register is an internal flash memory ECC parity register. This register is 32 bits wide.

DFMC_ECCEPARTIY=0x4100_1058

31 30 29 28 27 26 25 24	23 22 21 20 19 18 17 16	15 14 13 12 11 10 9 8	7 6 5 4 3 2 1 0
Reserved			ECCEP[6:0]
-			0x00
-			R0

6 0	ECCEP[6:0]	ECC error parity

6.11.14 DFMC Register Map Summary

Table 75. DFMC Register Map Summary

Offset	Register name	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0x00	DFMC_CONF	Res	Res	Res	Res	Res	Res	WTEN	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	LATENCY[3:0]			
	Reset value							0																						0	0	0	0
0x04	DFMC_FLSKEY	FKEY[31:0]																															
	Reset value	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
0x0C	DFMC_FLSPROT	FUP512B_7	FUP512B_6	FUP512B_5	FUP512B_4	FUP512B_3	FUP512B_2	FUP512B_1	FUP512B_0	Res	Res	Res	Res	Res	Res	Res	Res	FPBY2K_15	FPBY2K_14	FPBY2K_13	FPBY2K_12	FPBY2K_11	FPBY2K_10	FPBY2K_9	FPBY2K_8	FPBY2K_7	FPBY2K_6	FPBY2K_5	FPBY2K_4	FPBY2K_3	FPBY2K_2	FPBY2K_1	FPBY2K_0
	Reset value	0	0	0	0	0	0	0	0									0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
0x14	DFMC_CTRL	FLOCK	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	ECCSI	ECCDN	Res	WDIEN	WABORT	Res	Res	Res	CERS	SERS	PERS	PGM
	Reset value	1																				0	0		0	0				0	0	0	0
0x18	DFMC_STAT	Res	Res	Res	Res	Res	Res	Res	Res	Res	WTERR	RPERR	WSERR	Res	FPERR	Res	FLERR	Res	Res	Res	Res	ECCSE	ECCDE	CDONE	WDONE	Res	Res	Res	Res	Res	Res	CBUSY	WBUSY
	Reset value										0	0	0		0	0	0					0	0	0	0							0	0
0x30	DFMC_CHKCTRL	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	CDRST	Res	Res	Res	Res	Res	Res	Res	CDIEN	Res	Res	Res	Res	Res	Res	BSTEN	BGEN
	Reset value																0								0							0	0
0x34	DFMC_CHKDOUT	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	CDOUT[15:0]															
	Reset value																	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1
0x38	DFMC_CHKSADDR	SADDR[25:0]																										FIXED_VALUE[5:0]					
	Reset value	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
0x3C	DFMC_CHKEADDR	EADDR[25:0]																										FIXED_VALUE[5:0]					
	Reset value	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1

Table 75. DFMC Register Map Summary (continued)

Offset	Register name	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0x40	DFMC_WTOUT	WTOUT[31:0]																															
	Reset value	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
0x50	DFMC_ECCEADDR	ECCEA[15:0]																															
	Reset value	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
0x54	DFMC_ECCEDATA	ECCED[31:0]																															
	Reset value	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
0x58	DFMC_ECCEPARITY	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	ECCEP[6:0]						
	Reset value																										0	0	0	0	0	0	0

7. Direct Memory Access Controller (DMA)

7.1 DMA Introduction

The direct memory access (DMA) controller is used for high-speed data transfers between peripherals and memories. DMA enables quick data transfers having memory to memory copying or moving of data within memory.

- 16 channels
- 8-/16-/32-bit data transfers supported
- Various buffers with the same size supported
- DMA transfers are triggered through peripheral interrupts

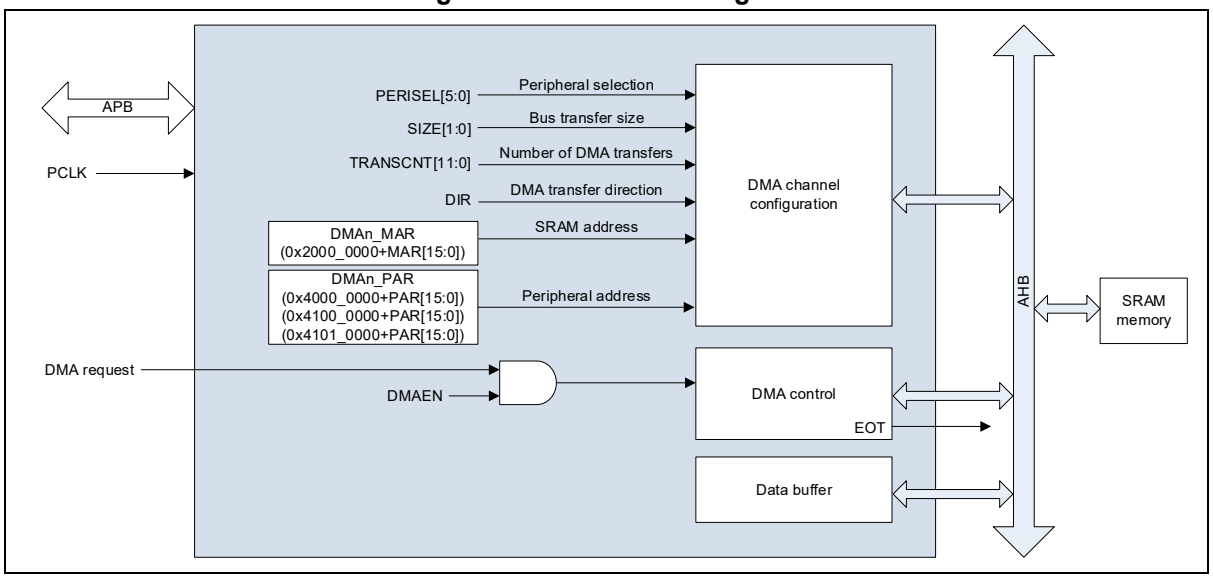
7.1.1 Modules Supported by DMA

A34G43x DMA supports the following 5 modules. Refer to this link for details.

- UART
  - UART functional description
    - Character transmission procedure/character reception
  - Continuous communication using DMA and UART
    - Transmission and reception using DMA
  - UART interrupts
- SPI
  - Configuration of SPI
  - Procedure for enabling SPI
  - Data transmission and reception procedures
    - DMA handshake
  - SPI status flags
    - DMA transmit operation complete flag (DMA to SPI, TXDMAF).
    - DMA receive operation complete flag (SPI to DMA, RXDMAF).
- ADC
  - A/D conversion using the DMA
  - ADC overrun (OVR) in DMA mode
  - ADC interrupts
- CRC
  - CRC using DMA
- AES-128
  - AES data register and data swapping
    - Data input and output
  - AES DMA interface
  - AES interrupts

Figure 60 shows a DMA block diagram.

Figure 60. DMA Block Diagram



7.2 DMA Functional Description

The DMA controller directly transfers data to memories by sharing the AHB with the microcontroller core. The AHB shares two masters operating in a round-robin fashion. Therefore, the DMA controller shares only half the system bandwidth with the microcontroller core.

The DMA controller is triggered only by requests made by peripherals. Once a peripheral requests a transfer to the DMA controller, the connected channel becomes enabled. Then, the bus is accessed to transfer the requested data from the memory buffer to the peripheral's data buffer (Or the other way around).

The DMA controller operates in the following sequence:

1. The programmer sets the address of the peripheral to use in the DMAn_PAR register and the memory address in the DMAn_MAR register.
2. In the DMAn_CR register, the programmer sets the DMA transfer count (0 to 4095), transferring direction, and bus transfer size (8-bit, 16-bit, and 32-bit).
3. In the DMAn_SR register, the programmer sets the DMAEN to '1' to enable the DMA channel.
4. A DMA request is made by the peripheral set in the PERISEL[5:0] bits of the DMAn_CR register and this triggers the requested peripheral channel to become active.
5. Data received from the source DMA address is read and stored in the internal buffer. The DMA operation writes this data to the target address.
6. Each time data is written to the target address, the DMA transfer count decreases by '1'. When the DMA transfer count reaches zero, the EOT bit of the DMAn_SR register is set to '1'. This signals an interrupt to the connected peripheral(1).

NOTE:

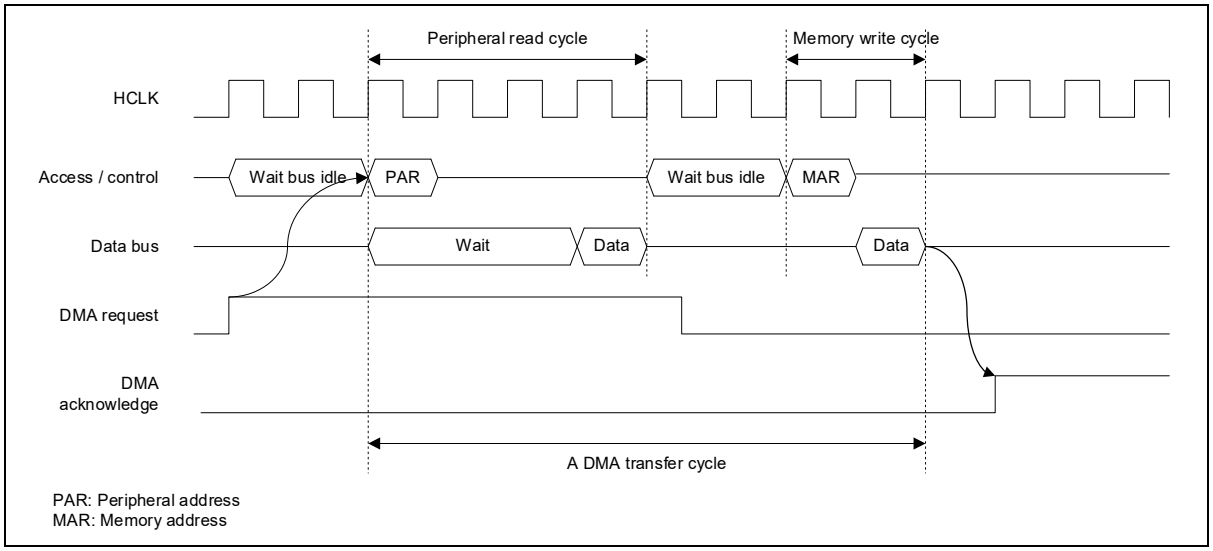
1. DMA itself has no interrupt sources; instead, each peripheral has flag bits that display the status of the connected DMA channel's transfer interrupts.

7.2.1 DMA Transfer Timing from Peripheral to Memory

The diagram below shows the functional timing of the DMA controller. A transfer request from a peripheral is internally held off while the AHB calls the address of the data read transfer source.

The data read from the source address is stored in the internal buffer. Then, once the AHB becomes available, the stored data is transferred to the target address. As shown in the diagram below, a waiting time of four clock cycles is required until the peripheral is accessed. If the bus is occupied by another master, additional wait-cycles will be taken.

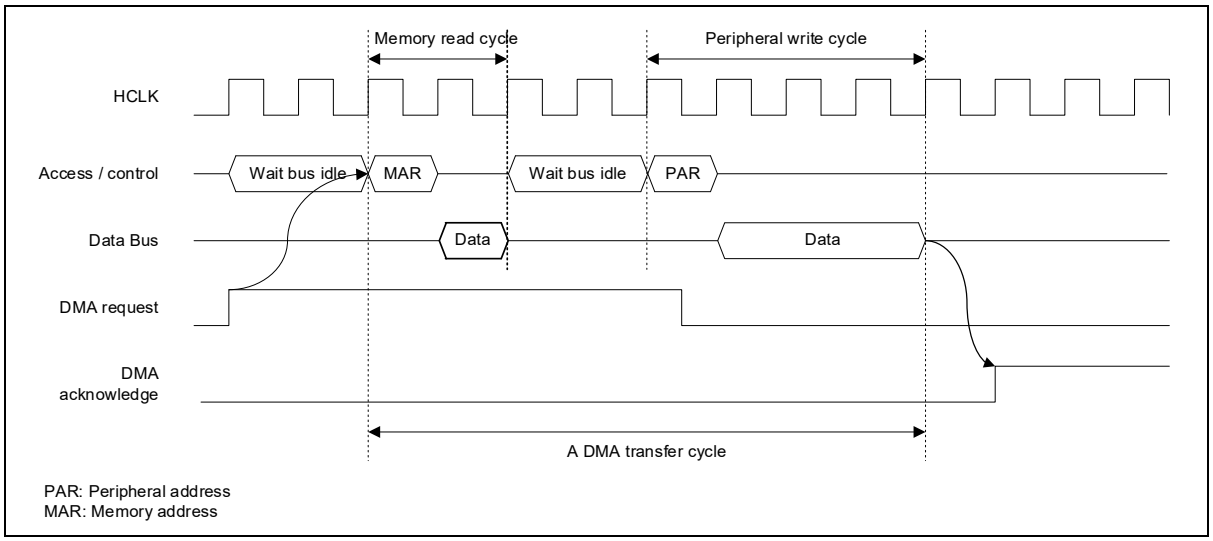
Figure 61. Diagram of DMA Transfer Timing from Peripheral to Memory



7.2.2 DMA Transfer Timing from Memory to Peripheral

The diagram below depicts the timing of a DMA transfer made from a memory to a peripheral. A waiting time of four clock cycles is required until the peripheral is accessed. If the bus is occupied by another master, additional wait-cycles will be taken.

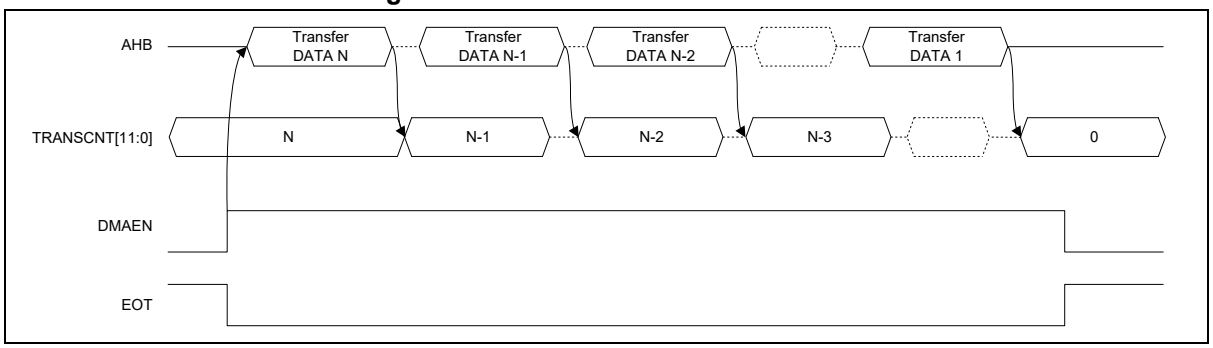
Figure 62. Diagram of DMA Transfer Timing from Memory to Peripheral



7.2.3 DMA Transfer

The diagram below depicts N number of DMA transfers in the queue as an example. DMA transferring starts when a DMAEN bit of DMAn_SR is set to ‘1’. And the bit is cleared to ‘0’ when all transfers have been completed.

Figure 63. N Number of DMA Transfers



7.3 DMA Registers

The base address of DMA is described in the following tables.

Table 76. Base Address of DMA

Name	Base Address
DMA0	0x4000_0400
DMA1	0x4000_0410
DMA2	0x4000_0420
DMA3	0x4000_0430
DMA4	0x4000_0440
DMA5	0x4000_0450
DMA6	0x4000_0460
DMA7	0x4000_0470
DMA8	0x4000_0480
DMA9	0x4000_0490
DMA10	0x4000_04A0
DMA11	0x4000_04B0
DMA12	0x4000_04C0
DMA13	0x4000_04D0
DMA14	0x4000_04E0
DMA15	0x4000_04F0

Table 77. DMA Register Map

Name	Offset	Type	Description	Reset Value	Reference
DMAn_CR	0x0000	RW	DMA Channel n Control Register	0x0000_0000	7.3.1
DMAn_SR	0x0004	RW	DMA Channel n Status Register	0x0000_0080	7.3.2
DMAn_PAR	0x0008	RW	DMA Channel n Peripheral Address Register	0x4000_0000	7.3.3
DMAn_MAR	0x000C	RW	DMA Channel n Memory Address Register	0x2000_0000	7.3.4

NOTE:

1. n = 0, 1, 2, …, and 15.

7.3.1 DMAn_CR: DMA Channel n Control Register

The DMAn_CR registers are DMA operation control registers. Each register is 32 bits wide.

DMA0_CR=0x4000_0400, DMA1_CR=0x4000_0410, DMA2_CR=0x4000_0420, DMA3_CR=0x4000_0430, DMA4_CR=0x4000_0440, DMA5_CR=0x4000_0450, DMA6_CR=0x4000_0460, DMA7_CR=0x4000_0470, DMA8_CR=0x4000_0480, DMA9_CR=0x4000_0490, DMA10_CR=0x4000_04A0, DMA11_CR=0x4000_04B0, DMA12_CR=0x4000_04C0, DMA13_CR=0x4000_04D0, DMA14_CR=0x4000_04E0, DMA15_CR=0x4000_04F0

1 30 29 28 27 26 25 24	23 22 21 20 19 18 17 16	15 14 13 12 11 10 9 8	7 6 5 4 3 2	1	0
Reserved	TRANSCNT[11:0]	Reserved	PERISEL[5:0]	Reserved	SIZE[1:0]	DIR	Reserved
-	0x000	-	000000	-	00	0	-
-	RW	-	RW	-	RW	RW	-

27 16	TRANSCNT[11:0]	Number of DMA transfers Before enabling DMA transfer, the number of transfers must be written to TRANSCNT[11:0].
	0	All DMA transfers have been completed.
	N	There are N transfers remaining.
13 8	PERISEL[5:0](1)	Peripheral selection The selected peripheral is connected to the DMA channel. PERISEL[5:0] must be written with the number representing the peripheral to be connected to the DMA interface.
	N	Selects the peripheral to use. Refer to Table 78.
3 2	SIZE[1:0]	Bus transfer size
	00	Sets the DMA transfer size to 1 byte.
	01	Sets the DMA transfer size to half-word (2 bytes) size.
	10	Sets the DMA transfer size to one-word (4 bytes) size.
	11	Holds off the size setting.
1	DIR	Transfer direction selection
	0	Memory-to-peripheral (TX)
	1	Peripheral-to-memory (RX)

NOTE:

1. A DMA channel will be connected with selected peripheral. Below table shows peripheral selection numbers. This PERISEL[5:0] field should be set with proper number of peripherals which will be connected with DMA interface.

Table 78. DMA PERISEL[5:0] Selection

PERISEL[5:0]	Associate Peripheral	PERISEL[5:0]	Associate Peripheral
0	CHANNEL IDLE	29	UART8 RX
1	UART0 RX	30	UART8 TX
2	UART0 TX	31	UART9 RX
3	UART1 RX	32	UART9 TX
4	UART1 TX	33	UART10 RX
5	UART2 RX	34	UART10 TX
6	UART2 TX	35	UART11 RX
7	UART3 RX	36	UART11 TX
8	UART3 TX	37	UART12 RX
9	UART4 RX	38	UART12 TX
10	UART4 TX	39	SPI3 RX
11	UART5 RX	40	SPI3 TX
12	UART5 TX	41	SPI4 RX
13	SPI0 RX	42	SPI4 TX
14	SPI0 TX	43	SPI5 RX
15	SPI1 RX	44	SPI5 TX
16	SPI1 TX	45	SPI6 RX
17	SPI2 RX	46	SPI6 TX
18	SPI2 TX	47	SPI7 RX
19	ADC0 RX	48	SPI7 TX
20	ADC1 RX	49	SPI8 RX
21	ADC2 RX	50	SPI8 TX
22	AES-128 RX	51	SPI9 RX
23	AES-128 TX	52	SPI9 TX
24	CRC Tx	53	SPI10 RX
25	UART6 RX	54	SPI10 TX
26	UART6 TX	55	SPI11 RX
27	UART7 RX	56	SPI11 TX
28	UART7 TX	-	-

The PERISEL[5:0] bits cannot have the same value on different channels. If the same PERISEL[5:0] value is written in more than one channel, proper operation cannot be guaranteed. Unused channels should have CHANNEL IDLE value in the PERISEL[5:0] bit positions.

7.3.2 DMAn_SR: DMA Channel n Status Register

DMAn_SR is a 32-bit register. It shows the status of the DMA controller and whether the DMA channel is enabled or disabled.

DMA0_SR=0x4000_0404, DMA1_SR=0x4000_0414, DMA2_SR=0x4000_0424, DMA3_SR=0x4000_0434, DMA4_SR=0x4000_0444, DMA5_SR=0x4000_0454, DMA6_SR=0x4000_0464, DMA7_SR=0x4000_0474, DMA8_SR=0x4000_0484, DMA9_SR=0x4000_0494, DMA10_SR=0x4000_04A4, DMA11_SR=0x4000_04B4, DMA12_SR=0x4000_04C4, DMA13_SR=0x4000_04D4, DMA14_SR=0x4000_04E4, DMA15_SR=0x4000_04F4

31 30 29 28 27 26 25 24	23 22 21 20 19 18 17 16	15 14 13 12 11 10 9 8	7	6 5 4 3 2 1	0
Reserved			EOT	Reserved	DMAEN
-			1	-	0
-			RO	-	RW

7	EOT	End-of-Transfer flag.	
		0	There are remaining transfers. TRANSCNT[11:0] has a value other than zero.
		1	All data has been transferred. The TRANSCNT[11:0] value is zero.
0	DMAEN(1)	DMA enable	
		0	Disables DMA or fix the status.
		1	Enables DMA or available to use.

7.3.3 DMAn_PAR: DMA Channel n Peripheral Address Register

DMAn_PAR shows the address value of the connected peripheral.

DMA0_PAR=0x4000_0408, DMA1_PAR=0x4000_0418, DMA2_PAR=0x4000_0428, DMA3_PAR=0x4000_0438, DMA4_PAR=0x4000_0448, DMA5_PAR=0x4000_0458, DMA6_PAR=0x4000_0468, DMA7_PAR=0x4000_0478, DMA8_PAR=0x4000_0488, DMA9_PAR=0x4000_0498, DMA10_PAR=0x4000_04A8, DMA11_PAR=0x4000_04B8, DMA12_PAR=0x4000_04C8, DMA13_PAR=0x4000_04D8, DMA14_PAR=0x4000_04E8, DMA15_PAR=0x4000_04F8

31 30 29 28 27 26 25 24	23 22 21 20 19 18 17 16	15 14 13 12 11 10 9 8	7 6 5 4 3 2 1 0
Peripheral base address offset(1)		PAR[15:0]	
0x4000		0x0000	
RW		RW	

15 0	PAR[15:0]	Address of the transfer/receive buffer's target peripheral. The address does not change until the ongoing transfer is completed.

NOTE:

1. Peripheral base address in DMAn_PAR register must be set to '0x4000' or '0x4100' or '0x4101'.

7.3.4 DMAn_MAR: DMA Channel n Memory Address Register

The DMAn_MAR registers show the DMA transfer's target memory address.

DMA0_MAR=0x4000_040C, DMA1_MAR=0x4000_041C,
DMA2_MAR=0x4000_042C, DMA3_MAR=0x4000_043C,
DMA4_MAR=0x4000_044C, DMA5_MAR=0x4000_045C,
DMA6_MAR=0x4000_046C, DMA7_MAR=0x4000_047C,
DMA8_MAR=0x4000_048C, DMA9_MAR=0x4000_049C,
DMA10_MAR=0x4000_04AC, DMA11_MAR=0x4000_04BC,
DMA12_MAR=0x4000_04CC, DMA13_MAR=0x4000_04DC,
DMA14_MAR=0x4000_04EC, DMA15_MAR=0x4000_04FC

31 30 29 28 27 26 25 24	23 22 21 20 19 18 17 16	15 14 13 12 11 10 9 8	7 6 5 4 3 2 1 0
Memory base address offset(1)		MAR[17:0]	
0x2000		0x00000	
RW		RW	

17 0	MAR[17:0]	Target memory address of the data transfer On the completion of each transfer, the address automatically increases, depending on the setting of the SIZE[1:0] bits.

NOTES:

1. Memory Base Address in DMAn_MAR register must be set to '0x2000'.
2. When powering on/off the SRAM, it is necessary to check the SRAM size. (Refer to 4.8.38.)

7.3.5 DMA Register Map Summary

Table 79. DMA Register Map Summary

Offset	Register name	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0x00	DMAn_CR	Res.	Res.	Res.	Res.	TRANSCNT[11:0]												Res.	Res.	PERISEL[5:0]						Res.	Res.	Res.	Res.	SIZE[1:0]		DIR	Res.
	Reset value					0	0	0	0	0	0	0	0	0	0	0	0			0	0	0	0	0	0					0	0	0	
0x04	DMAn_SR	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	EOT	Res.	Res.	Res.	Res.	Res.	Res.	DMAEN
	Reset value																									1							0
0x08	DMAn_PAR	Peripheral Base Address																PAR[15:0]															
	Reset value	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
0x0C	DMAn_MAR	Memory Base Address																MAR[17:0]															
	Reset value	0	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

8. Watchdog Timer (WDT)

8.1 WDT Introduction

The Watchdog Timer (WDT) is used to detect microcontroller errors due to external interference or unexpected logical condition. These errors cause the application program to give up the normal sequence. If the microcontroller is out of control, the WDT resets the microcontroller and returns to normal.

The WDT of the A34G43x series is a 32-bit down counter. If the WDT is set as a reset source, the microcontroller restarts when the down counter reaches zero.

The WDT can be used as a cycle timer along with an interrupt.

8.2 WDT Main Features

The WDT module has key features as listed below:

- 32-bit down counter
- WDT underflow reset functionality
- Cycle timer and underflow interrupt functionalities
- WDT input clock sources selectable
  - PCLK
  - The clock source can be selected by configuring the WDTCSEL[2:0] bits in the SCU_MCCR1 register. (LSI, LSE, MCLK, HSI, HSE, and PLL)
- Eight levels prescaler for the WDT input clock
- In debug mode, the WDT counter can be set not to operate.

8.3 WDT Functional Description

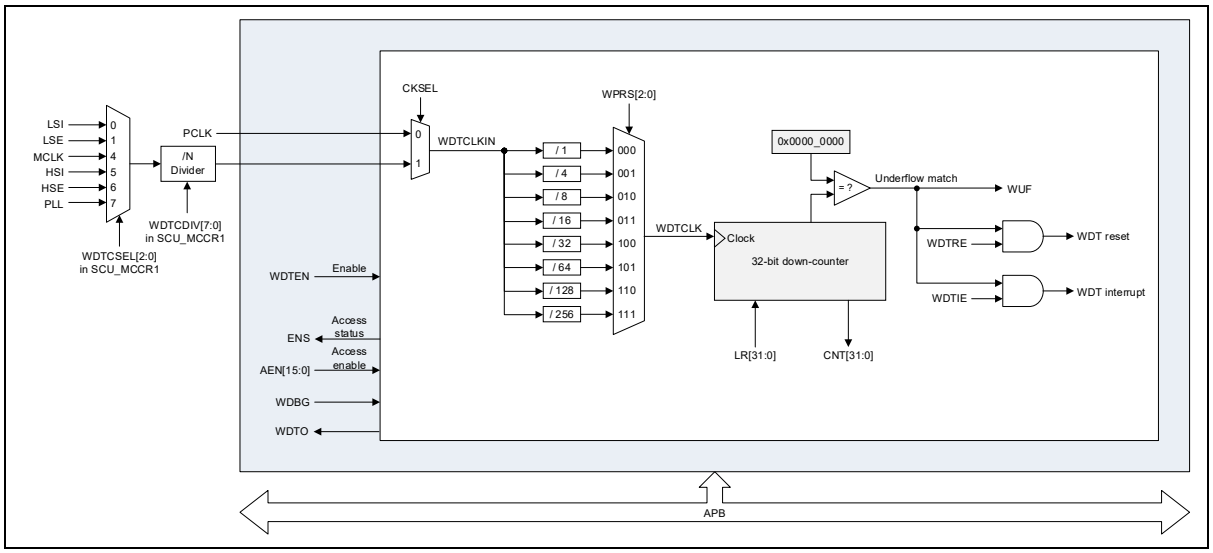
Once the WDT module is enabled and the WDTEN bit in the WDT_CON register is set to '1', the 32-bit down counter decrements. The application program writes a value in the WDT_LR register regularly to prevent an instant reset or an interrupt. Users must write certain values in the WDT_LR register to update WDT_CNT value.

Timeout cycle of the WDT is set to 2,000 ms by default.

8.3.1 WDT Block Diagram

Figure 64 shows a block diagram for the WDT.

Figure 64. WDT Block Diagram



8.3.2 Enabling WDT

The WDT is always enabled after power on, and it operates in WDT reset mode.

The PCLK (Peripheral Clock) or an external clock source can be set as a clock source for the WDT. The external clock source can be selected as the WDT's clock source when the CKSEL in WDT_CON register is set to '1' and the SCU_MCCR1 register is set to select the external clock source. When changing the clock source, set it after disabling WDTEN in WDT_CON register.

8.3.3 Controlling Down Counter

When the WDTEN in WDT_CON register is set to '1', the 32-bit down counter that is the WDT_CNT register starts counting down. To prevent an instant reset or an interrupt during the down counter operation, the WDT_CNT register must be set to a certain value.

To change the value of the WDT_CNT register, users need to follow the procedure below:

1. Write a number larger than '0' to the WDT_LR register with the WDTEN bit in the WDT_CON register set to '1'.
2. Reload the counter value using the WDT_LR register while the down counter operates. It is important that the operating down counter must have a value larger than '0'.

The WDT includes a programmable 32-bit down counter prescaler that enables users to set the timeout intervals using many different methods. Users can generate the WDT base clock by configuring the WPRS[2:0] bits in the WDT_CON register, and the minimum frequency value of the prescaler is 1 / 256 of the clock source frequency.

Table 80 describes the prescaled frequencies of various clock sources in the WDT.

Table 80. Prescaled WDT Counter Clock Frequency

Clock Source	WDTCLKIN / 1	WDTCLKIN / 4	WDTCLKIN / 16	WDTCLKIN / 64	WDTCLKIN / 256
LSI	500 kHz	125 kHz	31.25 kHz	7.8125 kHz	1.953125 kHz
LSE	32.768 kHz	8.192 kHz	2.048 kHz	512Hz	128Hz
MCLK	MCLK	MCLK / 4	MCLK / 16	MCLK / 64	MCLK / 256
HSI	32 MHz	8 MHz	2 MHz	500 kHz	125 kHz
HSE	XTAL	XTAL / 4	XTAL / 16	XTAL / 64	XTAL / 256
PLL	PLL	PLL / 4	PLL / 16	PLL / 64	PLL / 256

NOTE:

1. Time-out period = (Load Value + 1) × (1 / pre-scaled WDT counter clock frequency) at interrupt operation
   Time-out period = (Load Value) × (1 / pre-scaled WDT counter clock frequency) at reset operation
   // At the load value reaches '0', underflow flag is set to '1'

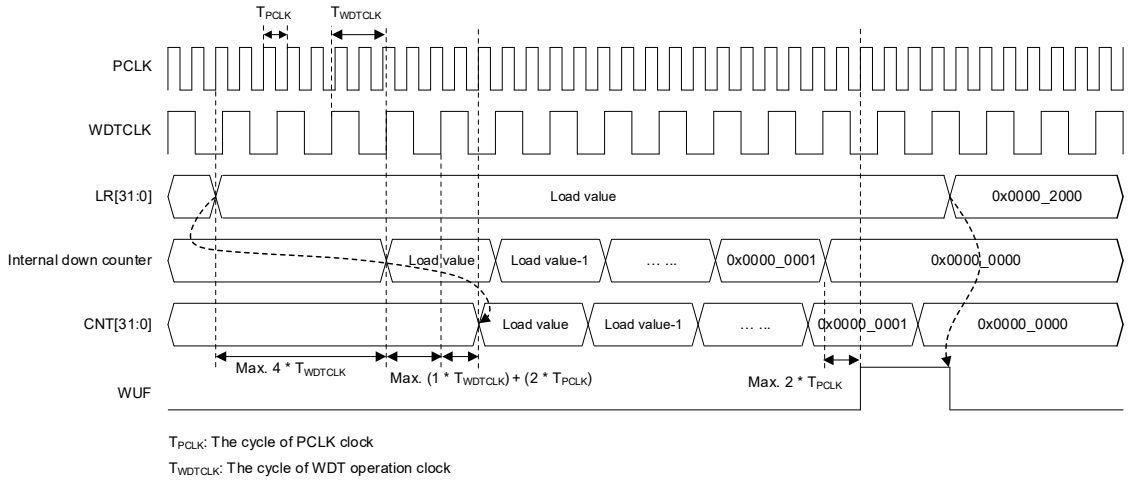
8.3.4 Interrupt Feature

The WDT interrupt can be used to perform safety tasks or data logging before triggering an actual reset.

The WDT interrupt is enabled by setting the WDTIE bit in the WDT_CON register to '1'. When the value of the WDT_CNT down counter changes from one to zero, the interrupt is triggered and the WUF flag in the WDT_CON register is set to '1'. To clear the WUF flag, users must write a value other than zero in the WDT_LR register.

8.3.4.1 Interrupt Mode Operation Timing Diagram

Figure 65. Interrupt Mode Operation Timing Diagram with External Clock



NOTE:

1. The internal WDT 32-bit down counter uses the WDTCLK as a reference clock, and the WDT registers operate based on the PCLK. To synchronize these two clocks, the delay is introduced as shown in the above timing diagram.

In WDT interrupt mode, to prevent the WDT's next interrupt in short time after the WDT underflows, specific counter value should be reloaded. This reload operation can be used only when the WDT timer counter is set for Interrupt Mode.

It takes five cycles from the loaded value to the count value. The WDT interrupt signal for the Internal down counter value can be delayed up to two system bus clocks (For the case of synchronized logic).

8.3.5 How to Program Watchdog Timeout

8.3.5.1 Basic Reset Operation

To reset the WDT, users must set the WDTRE bit in the WDT_CON register and the WDTRST bit in the SCU_RSER register to ‘1’. When the down counter reaches zero (the moment the value of the down counter changes from one to zero), the WDT generates the reset signal to reset microcontroller.

8.3.6 Debug Mode

If the WDBG bit in the WDT_CON register is set to ‘1’, users can stop the WDT in Debug Mode. Users can see that the down counter value stops without decreasing, when pressing the Stop in Debug Mode. If the users execute the run again, the down counter value starts to decrease by one.

Conversely, if the WDBG bit in the WDT_CON register is set to ‘0’, the WDT counter continues to operate without stopping in Debug Mode. If users press Stop in Debug Mode, the down counter value appears to be stationary. However, when executing Run again, the users can check that the down counter value is reduced because the down counter was not stationary.

8.4 WDT Registers

The base address of WDT and register map are described in the following. Initial setting of WDT's time-out period is 2,000 ms.

Table 81. Base Address of WDT

Name	Base Address
WDT	0x4000_0200

Table 82. WDT Register Map

Name	Offset	Type	Description	Reset Value	Reference
WDT_LR	0x0000	RW	WDT Load Register	0x0000_0000	8.4.1
WDT_CNT	0x0004	RO	WDT Current Count Register	0x0000_7A12	8.4.2
WDT_CON	0x0008	RW	WDT Control Register	0x0000_805C	8.4.3
WDT_AEN	0x00F0	RW	WDT Access Enable Register	0x0000_0000	8.4.4

8.4.1 WDT_LR: WDT Load Register

The WDT_LR is used to update the value of the WDT_CNT register. To change the WDT_CNT's value, two conditions must be satisfied: the WDTEN in WDT_CON register must be set to '1', and the WDT_LR must be written.

If the WDTEN bit in the WDT_CON register is set to '0', the value written to the WDT_LR register will remain unrepresented in the WDT_CNT register until the WDTEN bit is changed to '1'.

If the WDT is being used as a reset source, the WDT_LR register must be written before the WDT_CNT value becomes '0' to prevent a reset.

The WDT triggers an event at the moment when the WDT_CNT value is changed from '1' to '0'. Therefore, when the WDT is used in reset mode, the WDT's count value is written to the WDT_LR as it is; whereas when the WDT is used in interrupt mode, '1' must be subtracted from the count value before the WDT_LR is written to.

At least five WDT counter clock cycles are required to update the WDT_CNT with the WDT_LR value.

WDT_LR=0x4000_0200

31 30 29 28 27 26 25 24	23 22 21 20 19 18 17 16	15 14 13 12 11 10 9 8	7 6 5 4 3 2 1 0
LR[31:0]			
0x0000_0000			
RW			

31 0	LR[31:0]	WDT load value register If the WDTEN remains at '1', the WDT_CNT register will be updated with the WDT_LR value. Since the initial value of WDT_LR is zero, it must be set to the desired value before use.

8.4.2 WDT_CNT: WDT Current Count Register

The WDT_CNT register is a 32-bit down counter that shows the WDT's current value. It is a read-only register. Its value can be changed by writing to the WDT_LR register while the WDTEN bit in the WDT_CON register is '1'.

Thus, when the WDT's count value reaches zero, an interrupt or reset is triggered.

To use the WDT as a reset source, both the WDTRE bit in the WDT_CON register and the WDTRST bit in the SCU_RSER register must be set to '1'.

WDT_CNT=0x4000_0204

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
CNT[31:0]																															
0x0000_7A12																															
R0																															

31 0	CNT[31:0]	WDT current counter register The 32-bit down-counter counts down from the value written to the WDT_LR register.

8.4.3 WDT_CON: WDT Control Register

The microcontroller's WDT module must be set appropriately before it is enabled. The WDT module can be programmed to trigger a reset event or interrupt signal.

Instead of being used as a reset source or an interrupt source, the WDT can also function as a countdown timer starting from the set value of the down counter.

The WUF bit is a flag that is set when the WDT counts down to zero.

WDT_CON=0x4000_0208

31 30 29 28 27 26 25 24	23 22 21 20 19 18 17 16	15	14 13 12 11 10 9	8	7	6	5	4	3	2 1 0
Reserved		WDBG	Reserved	WUF	WDTIE	WDTRE	Reserved	WDTEN	CKSEL	WPRS[2:0]
-		1	-	0	0	1	-	1	1	100
-		RW	-	RO	RW	RW	-	RW	RW	RW

15	WDBG	Enable the WDT in debug mode	
		0	Enables the WDT counter stop in debugging mode
		1	Disables the WDT counter stop in debugging mode
8	WUF	WDT underflow flag (The bit is cleared when WDT_LR is written.)	
		0	There is no underflow.
		1	Underflow is pending.
7	WDTIE	Enable the WDT counter underflow interrupt	
		0	Disables the interrupt.
		1	Enables the interrupt.
6	WDTRE	Enable the WDT counter underflow reset	
		0	Disables the WDT counter underflow reset.
		1	Enables the WDT counter underflow reset.
4	WDTEN	Enable the WDT counter	
		0	Disables the WDT counter.
		1	Enables the WDT counter.
3	CKSEL	WDTCLKIN clock source selection	
		0	PCLK
		1	External clock (SCU_MCCR1)
2 0	WPRS[2:0]	Counter clock prescaler	
		000	WDTCLK = WDTCLKIN
		001	WDTCLK = WDTCLKIN / 4
		010	WDTCLK = WDTCLKIN / 8
		011	WDTCLK = WDTCLKIN / 16
		100	WDTCLK = WDTCLKIN / 32
		101	WDTCLK = WDTCLKIN / 64
		110	WDTCLK = WDTCLKIN / 128
		111	WDTCLK = WDTCLKIN / 256

NOTE:

1. WDTO output is output according to WUF (WDT underflow flag) in the WDT_CON Register.
   If WUF is underflow pending with '1', WDTO outputs high-level, and if WUF is '0', WDTO outputs low-level.

8.4.4 WDT_AEN: WDT Access Enable Register

The WDT_AEN register enables or disables writing to the settings of all WDT registers. In addition, when the WDT_AEN register is written '0x555A', WDT_CNT value is updated as current WDT_LR value instantly. After using the instant reload, WDT access is disabled automatically.

WDT_AEN=0x4000_02F0

31 30 29 28 27 26 25 24	23 22 21 20 19 18 17 16	15 14 13 12 11 10 9 8	7 6 5 4 3 2 1 0
Reserved	ENS	AEN[15:0]	
-	0	-	
-	R0	RW	

16	ENS	Enable status bit
	0	Disables AEN.
	1	Enables AEN (WDT register access possible state)
15 0	AEN[15:0](1)	Enable the write access of WDT registers
		Writing '0xA55A' to the bit field enables writing new values to WDT module's registers. Users can write a different value to this bit field to protect the WDT registers against being updated with new values. Additionally, writing 0x555A to the bit field triggers an instant reload. When using the instant reload, the WDT_LR register should be set appropriate value before.

NOTES:

1. Example code for using the WDT_AEN register

```
WDT_AEN = 0xA55A; // Enables WDT access.
                  // WDT_LR and WDT_CON, etc. become settable.
WDT_AEN = 0;      // Disables WDT access.
```

2. To use the reload feature, a value greater than 0 must be set in the WDT_LR register.

```
WDT_AEN = 0xA55A; // Enable WDT access.
WDT_LR = 0x7A12;  // Set timer load value.
WDT_AEN = 0;      // Disable WDT access.
```

3. The instant reload feature

```
WDT_AEN = 0x555A; // Triggers an instant reload.
```

4. When using the instant reload, the WDT_LR register should be set appropriate value before. After using the instant reload, WDT access is disabled automatically.

8.4.5 WDT Register Map Summary

Table 83. WDT Register Map Summary

Offset	Register name	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0x00	WDT_LR	LR[31:0]																															
	Reset value	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
0x04	WDT_CNT	CNT[31:0]																															
	Reset value	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	1	1	1	0	1	0	0	0	0	1	0	0	1	0
0x08	WDT_CON	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	WDBG	Res.	Res.	Res.	Res.	Res.	Res.	WUF	WDTIE	WDTRE	Res.	WDTEN	CKSEL	WPRS[2:0]		
	Reset value																	1							0	0	1		1	1	1	0	0
0xF0	WDT_AEN	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	ENS	AEN[15:0]															
	Reset value																0	-															

9. Pulse-Width Modulation (PWM) Generator

9.1 PWM Generator Introduction

The A34G43x series has ten built-in PWM (Pulse-Width Modulation) generators that can output PWM (Pulse-Width Modulation) waveforms. This PWM (Pulse-Width Modulation) generator has a 16-bit resolution, five channels form a 1-unit, and each unit has a built-in prescaler. PWM (Pulse-Width Modulation) It can be used as square waveform signal required for user applications such as LED, motor, inverter, etc. through PWM related register setting.

9.2 PWM Generator Main Features

The PWM Generator module of the A34G43x series has the following main features in PWM modes:

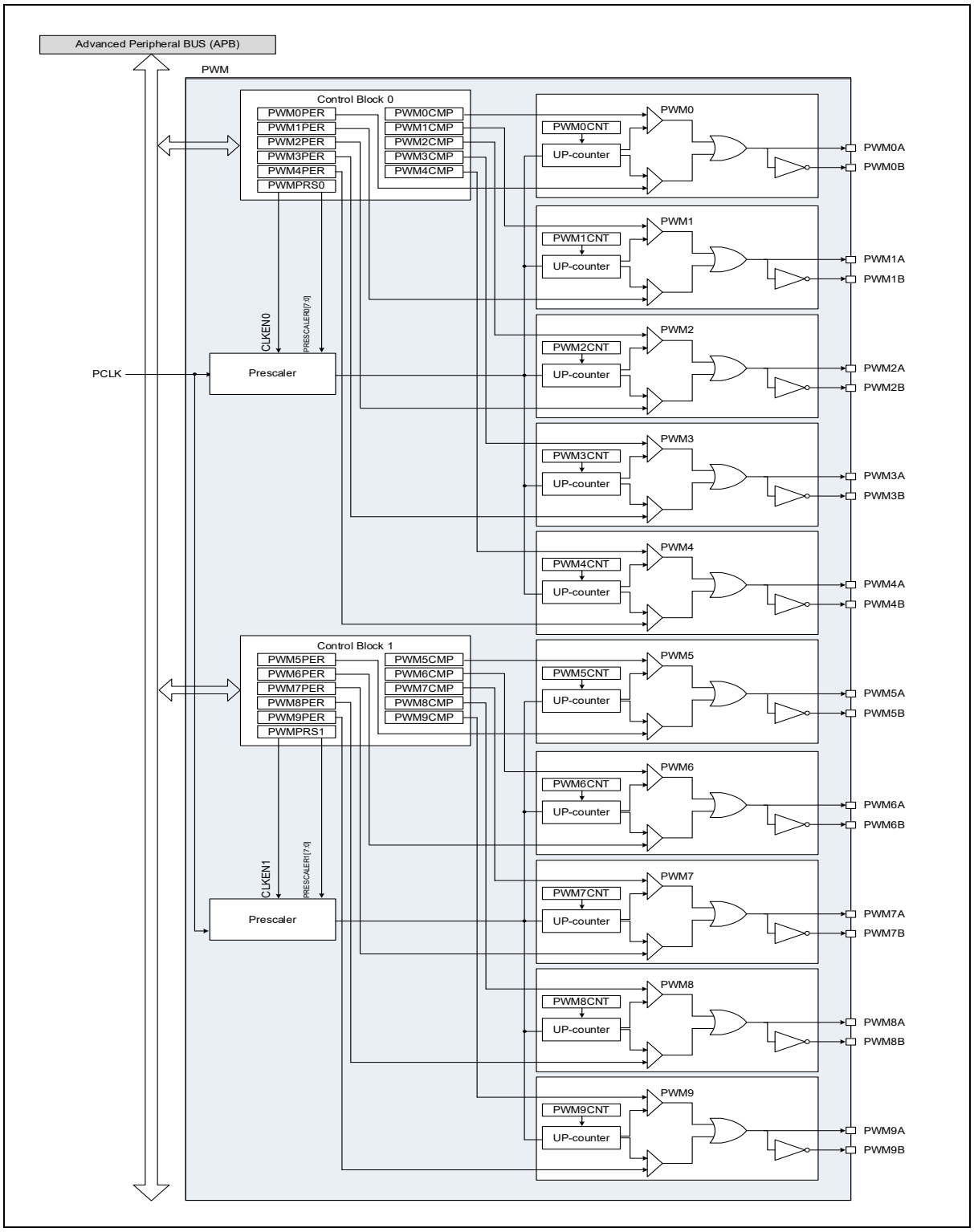
- Output with 16-bit resolution
- Outputs from up to ten channels
- PWM invert signal output channels
  - 120-pin (PWM0 to PWM9)
  - 100-pin (PWM0 to PWM9)
  - 64-pin (PWM0 to PWM9, Except PWM5B, PWM6B, PWM7B)
- PWM duty setting function by setting counter and period register.
- PWM input clocks
  - PCLK
  - Four steps (1/2, 1/4, 1/8, 1/16) prescaler
- Integrated 1-unit 8-bit prescaler for PWM input clock division
- In multichannel PWM operation, timing synchronization is based on the PWM signal with the shortest cycle.

9.3 PWM Generator Functional Description

9.3.1 Block Diagram

Figure 66 shows a block diagram of the PWM Generator.

Figure 66. PWM Block Diagram



9.3.2 Pins and Internal Signals

The external pins connected to the PWM module are described in Table 84.

Table 84. Pin Assignment of PWM: External Pins

Pin Name	Type	Description	Supported Packages		
			A34G43xYL (120-LQFP)	A34G43xVL (100-LQFP)	A34G43xRL (64-LQFP)
PWM0A	O	Output of PWM0	O	O	O
PWM0B	O	Reverse output of PWM0	O	O	O
PWM1A	O	Output of PWM1	O	O	O
PWM1B	O	Reverse output of PWM1	O	O	O
PWM2A	O	Output of PWM2	O	O	O
PWM2B	O	Reverse output of PWM2	O	O	O
PWM3A	O	Output of PWM3	O	O	O
PWM3B	O	Reverse output of PWM3	O	O	O
PWM4A	O	Output of PWM4	O	O	O
PWM4B	O	Reverse output of PWM4	O	O	O
PWM5A	O	Output of PWM5	O	O	O
PWM5B	O	Reverse output of PWM5	O	O	O
PWM6A	O	Output of PWM6	O	O	O
PWM6B	O	Reverse output of PWM6	O	O	N/A
PWM7A	O	Output of PWM7	O	O	O
PWM7B	O	Reverse output of PWM7	O	O	N/A
PWM8A	O	Output of PWM8	O	O	O
PWM8B	O	Reverse output of PWM8	O	O	N/A
PWM9A	O	Output of PWM9	O	O	O
PWM9B	O	Reverse output of PWM9	O	O	O

9.3.3 Setting Input Clock of PWM (Pulse-Width Modulation) Generator

The source clock for PWM (Pulse-Width Modulation) is determined by the system peripheral clock and the prescaler in the PWM_PRSn register or the prescaler in the PWMn_CTRL register. PWM (Pulse-Width Modulation) source clock can be obtained by the formula below:

$$\mathrm{PWMCLKn} = \frac{\mathrm{PCLK}}{\mathrm{PRESCALERn} + 1}$$

The first way to get the PWM source clock is to obtain the PWM input clock PWMCLKn by activating PWMCLKn with the PWMPRSn [15] bits in the PWM_PRSn register and setting the PRESCALERn [7: 0] value to the system peripheral input clock. PWMCLK0 is the clock applied to PWM0~4 group, and PWMCLK1 is the clock applied to PWM5~9 group.

PWMCLKn applied to each PWMn can be set to the input clock of each PWM (Pulse-Width Modulation) channel by dividing 2, 4, 8, or 16 by the value set in CKSEL [7: 5] through the clock divider.

9.3.4 Setting Period and PWM Output Signal

Each PWMn channel has a PWMn_PER register for setting the PWM frequency.

PWM period can be obtained by below formula with 16-bit width value.

The PWM (Pulse-Width Modulation) period can be obtained by applying the following formula with 16-bit PWMn_PER value and the PWM input clock.

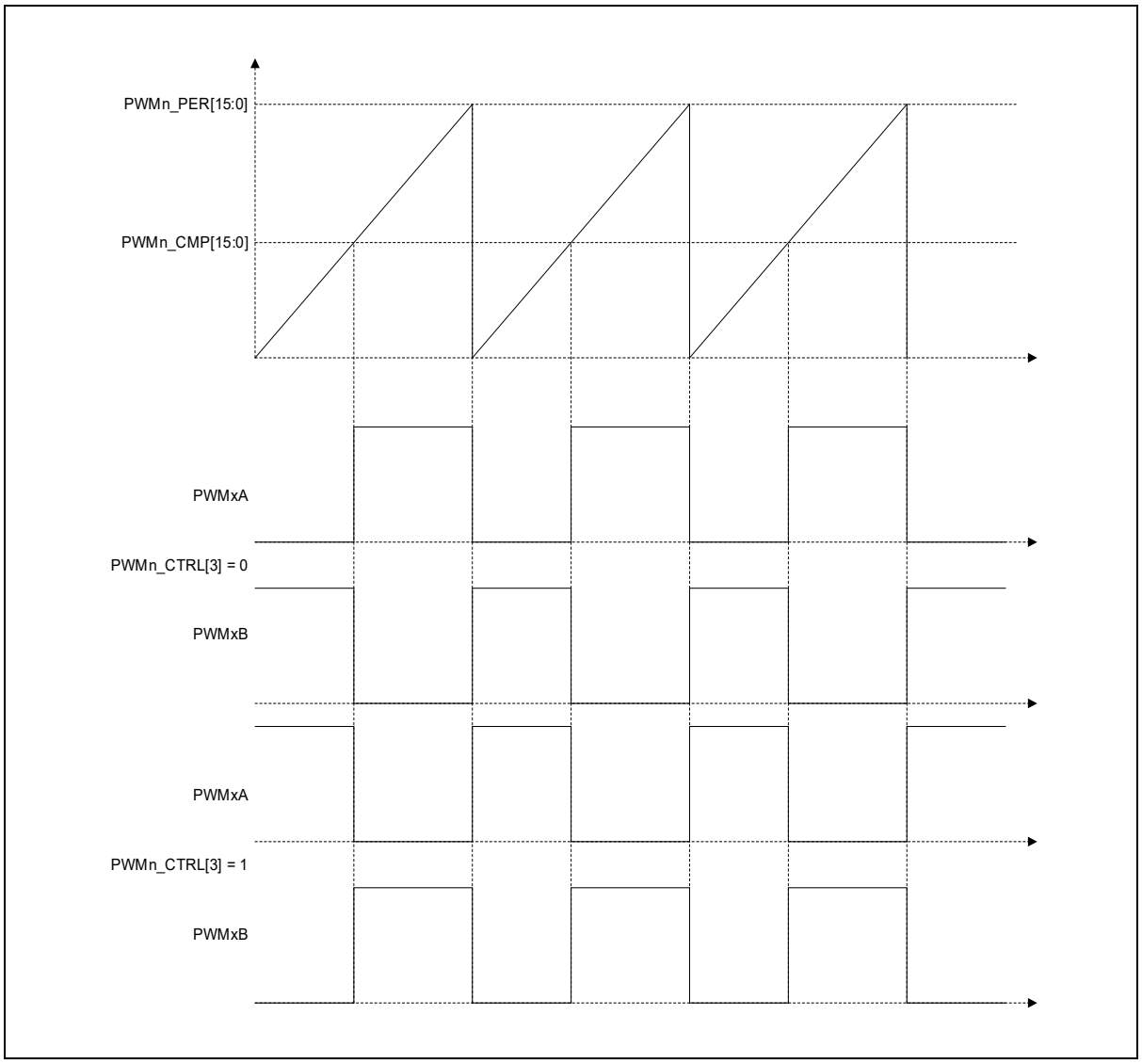
$$\text{PWM Period} = \frac{\text{PWMCLK}}{\text{PERIOD} + 1}$$

Each PWMn channel can set PWM duty as well as period. PWM duty can be set to a value in the range of 0 ~ PERIOD in PWMn_CMP[15:0].

The default output of PWMxA is 'L', and the PWMxA output changes to 'H' when the PWMn_CNT register value reaches the PWMn_CMP value.

You can also invert the starting output level of PWMxA by entering a value in the INVA bit in the PWMn_CTRL register. If you enter 0 as the default value, the start output value will be Low. If you input 1, High will be output. PWMxB is output when the PWMxA waveform is inverted.

Figure 67. PWM Timing Diagram



After each of the above functions are set, if 1 is input to the STRT bit in the PWMn_CTRL register, PWM is output. If ‘0’ is input, PWM output is stopped.

9.3.5 Synchronization of PWM Channels

The PWM (Pulse-Width Modulation) block has a function to set synchronization between channels. if the synchronization function is enabled through the control registers of each channel in the PWM0~4 group synchronizes the counting of the active PWM channels. The timing of the synchronized PWM channels is set based on the period of the shortest PWM channel. Similarly, PWM5~9 groups operate in the same way as PWM0~4 groups. However, the synchronization function of PWM0~4 group and PWM5~9 group is independent and does not affect the operation between PWM0~4 and PWM5~9 groups.

NOTE:

1. In a PWM group (PWM0~4 or PWM5~9), when the enabled synchronous (SYNC = 1) channel and asynchronous (SYNC = 0) channel are used at the same time, the period of synchronous PWM channels is affected by asynchronous PWM channels. Therefore, if the application requires independent PWM operation, it is recommended to use PWM channel of other PWM groups or to use TIMER with PWM mode.

9.4 PWM Interrupts

Each of PWMn channels can be configured periodic interrupt. If the PRIE bit in the PWMn_CTRL register is set to '1', the PWM periodic interrupt is enabled. If the PRIE bit in the PWMn_CTRL register is set to '0', the periodic interrupt is disabled.

Interrupt vectors are supported for each channel. If an interrupt occurs in each PWMn channel, it is possible to check whether an interrupt is generated by reading the PRF bit in the PWMn_CTRL register. If a periodic interrupt occurs, the PRF bit must be cleared by writing 1 to it.

Table 85. Interrupt on PWMn

Interrupt Event	Event Flag	Enable Control Bit	Interrupt Clear Method
PWM0 period	PRF in PWM0_CTRL	PRIE in PWM0_CTRL	Write 1 in PRF
PWM1 period	PRF in PWM1_CTRL	PRIE in PWM1_CTRL	Write 1 in PRF
PWM2 period	PRF in PWM2_CTRL	PRIE in PWM2_CTRL	Write 1 in PRF
PWM3 period	PRF in PWM3_CTRL	PRIE in PWM3_CTRL	Write 1 in PRF
PWM4 period	PRF in PWM4_CTRL	PRIE in PWM4_CTRL	Write 1 in PRF
PWM5 period	PRF in PWM5_CTRL	PRIE in PWM5_CTRL	Write 1 in PRF
PWM6 period	PRF in PWM6_CTRL	PRIE in PWM6_CTRL	Write 1 in PRF
PWM7 period	PRF in PWM7_CTRL	PRIE in PWM7_CTRL	Write 1 in PRF
PWM8 period	PRF in PWM8_CTRL	PRIE in PWM8_CTRL	Write 1 in PRF
PWM9 period	PRF in PWM9_CTRL	PRIE in PWM9_CTRL	Write 1 in PRF

9.4.1 Interrupt Generation

Table 86 describes the interrupts that the PWM module.

Table 86. Interrupt Vector of PWMn

Units	Interrupt Priority	Vector Address	Interrupt Source
PWM0	88	0x0000_01A0	PWM0 periodic interrupt
PWM1	89	0x0000_01A4	PWM1 periodic interrupt
PWM2	90	0x0000_01A8	PWM2 periodic interrupt
PWM3	91	0x0000_01AC	PWM3 periodic interrupt
PWM4	92	0x0000_01B0	PWM4 periodic interrupt
PWM5	93	0x0000_01B4	PWM5 periodic interrupt
PWM6	94	0x0000_01B8	PWM6 periodic interrupt
PWM7	95	0x0000_01BC	PWM7 periodic interrupt
PWM8	96	0x0000_01C0	PWM8 periodic interrupt
PWM9	97	0x0000_01C4	PWM9 periodic interrupt

9.5 PWM Registers

The base addresses and register map of the PWM are as follows:

Table 87. Base Addresses of PWM Interface

Name	Base Address
PWM0	0x4000_2000
PWM1	0x4000_2020
PWM2	0x4000_2040
PWM3	0x4000_2060
PWM4	0x4000_2080
PWM5	0x4000_2100
PWM6	0x4000_2120
PWM7	0x4000_2140
PWM8	0x4000_2160
PWM9	0x4000_2180
PWMPRS0	0x4000_20A0
PWMPRS1	0x4000_21A0

Table 88. PWM Register Map

Name	Offset	Type	Description	Reset Value	Reference
PWMn_CTRL	0x0000	RW	PWM Timer n Control Register	0x0000_0000	9.5.1.1
PWMn_CNT	0x0004	RW	PWM Timer n Counter Register	0x0000_0000	9.5.1.2
PWMn_PER	0x0008	RW	PWM Timer n Period Register	0x0000_0000	9.5.1.3
PWMn_CMP	0x000C	RW	PWM Timer n Compare Register	0x0000_0000	9.5.1.4
PWM_PRS0	0x00A0	RW	PWM0 to 4 Prescaler Register 0	0x0000_0000	9.5.1.5
PWM_PRS1	0x01A0	RW	PWM5 to 9 Prescaler Register 1	0x0000_0000	9.5.1.6

NOTE:

1. n = 0, 1, 2, 3, 4, 5, 6, 7, 8 and 9.

9.5.1.1 PWMn_CTRL: PWM Timer n Control Register

PWM (Pulse-Width Modulation) This register is used to control channel n. This register has a function to specify PWM clock and control settings for each PWM output channel.

PWM0_CTRL=0x4000_2000, PWM1_CTRL=0x4000_2020
PWM2_CTRL=0x4000_2040, PWM3_CTRL=0x4000_2060
PWM4_CTRL=0x4000_2080, PWM5_CTRL=0x4000_2100
PWM6_CTRL=0x4000_2120, PWM7_CTRL=0x4000_2140
PWM8_CTRL=0x4000_2160, PWM9_CTRL=0x4000_2180

31 30 29 28 27 26 25 24	23 22 21 20 19 18 17 16	15 14 13 12	11 10 9	8	7 6 5	4	3	2	1	0
Reserved		PRF	Reserved	PRIE	CKSEL[2:0]	Reserved	INVA	Reserved	SYNC	STRT
-		0	-	0	000	-	0	-	0	0
-		RWC1	-	RW	RW	-	RW	-	RW	RW

12	PRF	Display the PWM periodic interrupt flag	
		0	No periodic interrupt
		1	Periodic interrupt occurs (write '1' to clear)
8	PRIE	Enable/Disable PWM periodic interrupt	
		0	Disable PWM periodic interrupt
		1	Enable PWM periodic interrupt
7 5	CKSEL[2:0]	Select an input clock source of PWM	
		000	divided by 2
		001	divided by 4
		010	divided by 8
		011	divided by 16
		1xx	Disable input clock
3	INVA	Selection for PWM Inversion Output	
		0	Normal PWM output A (Not inverted)
		1	Inverted PWM output A
1	SYNC	Synchronize the current PWM channel to another channel's timing.	
		0	Do not synchronize this channel with the others
		1	Synchronize this channel to the other channels (Synchronized to the timing of the PWM channel with the smallest PRD value among the several PWM output channels.) (See chapter 9.3.5)
0	STRT	Control the operation of PWM	
		0	Stop counter
		1	Run counter

9.5.1.2 PWMn_CNT: PWM Timer n Counter Register

This register is the count value register of the PWM channel n. This register consists of 16 bits.

PWM0_CNT=0x4000_2004, PWM1_CNT=0x4000_2024
PWM2_CNT=0x4000_2044, PWM3_CNT=0x4000_2064
PWM4_CNT=0x4000_2084, PWM5_CNT=0x4000_2104
PWM6_CNT=0x4000_2124, PWM7_CNT=0x4000_2144
PWM8_CNT=0x4000_2164, PWM9_CNT=0x4000_2184

31 30 29 28 27 26 25 24	23 22 21 20 19 18 17 16	15 14 13 12 11 10 9 8	7 6 5 4 3 2 1 0
Reserved		CNT[15:0]	
-		0x0000	
-		RW	

15 0	CNT[15:0]	Current PWM counter value

9.5.1.3 PWMn_PER: PWM Timer n Counter Period Register

This register specifies the period of PWM channel n counter and consists of 16 bits. The value of the counter reaches the value specified in this register, the counter value is reset.

PWM0_PER=0x4000_2008, PWM1_PER=0x4000_2028
PWM2_PER=0x4000_2048, PWM3_PER=0x4000_2068
PWM4_PER=0x4000_2088, PWM5_PER=0x4000_2108
PWM6_PER=0x4000_2128, PWM7_PER=0x4000_2148
PWM8_PER=0x4000_2168, PWM9_PER=0x4000_2188

31 30 29 28 27 26 25 24	23 22 21 20 19 18 17 16	15 14 13 12 11 10 9 8	7 6 5 4 3 2 1 0
Reserved		PERIOD[15:0]	
-		0x0000	
-		RW	

15 0	PERIOD[15:0]	The period value of PWM channel n

$$\text{PWM Period} = \frac{\text{PWMCLK}}{\text{PERIOD} + 1}$$

9.5.1.4 PWMn_CMP: PWM Timer n Compare Register

This register specifies the value of PWM (Pulse-Width Modulation) comparator A for output A of channel n and consists of 16 bits. When the counter is reset, the output A is output as 'L'. If the value of this register matches the value of the PWM (Pulse-Width Modulation) counter, the output A is set to 'H'.

PWM0_CMP=0x4000_200C, PWM1_CMP=0x4000_202C
PWM2_CMP=0x4000_204C, PWM3_CMP=0x4000_206C
PWM4_CMP=0x4000_208C, PWM5_CMP=0x4000_210C
PWM6_CMP=0x4000_212C, PWM7_CMP=0x4000_214C
PWM8_CMP=0x4000_216C, PWM9_CMP=0x4000_218C

31 30 29 28 27 26 25 24	23 22 21 20 19 18 17 16	15 14 13 12 11 10 9 8	7 6 5 4 3 2 1 0
Reserved		CMP[15:0]	
-		0x0000	
-		RW	

15 0	CMP[15:0]	The comparator value of PWM channel n

9.5.1.5 PWM_PRS0: PWM 0 to 4 Prescaler Register

This register is used to set unit 0 of the PWM (Pulse-Width Modulation) generator. This register provides a prescaler for dividing the PWM input clock of unit 0 and the input clock set in this register is applied to PWM0 to PWM4 channels.

PWM_PRS0=0x4000_20A0

31 30 29 28 27 26 25 24	23 22 21 20 19 18 17 16	15	14 13 12 11 10 9 8	7 6 5 4 3 2 1 0
Reserved		CLKEN0	Reserved	PRESCALER0[7:0]
-		0	-	0x00
-		RW	-	RW

15	CLKEN0	Operation control of prescaler 0 clock	
		0	Stop prescaler 0 clock
		1	Run prescaler 0 clock
7 0	PRESCALER0 [7:0]	The prescaler value of unit 0	

$$\text{PWMCLK0} = \frac{\text{PCLK}}{\text{PRESCALER0} + 1}$$

9.5.1.6 PWM_PRS1: PWM 5 to 9 Prescaler Register

This register is used to set unit 1 of the PWM (Pulse-Width Modulation) generator. This register provides a prescaler for dividing the PWM input clock of unit 1 and the input clock set in this register is applied to the PWM5 to PWM9 channels.

PWM_PRS1=0x4000_21A0

31 30 29 28 27 26 25 24	23 22 21 20 19 18 17 16	15	14 13 12 11 10 9 8	7 6 5 4 3 2 1 0
Reserved		CLKEN1	Reserved	PRESCALER1[7:0]
-		0	-	0x00
.		RW	.	RW

15	CLKEN1	Operation control of prescaler 1 clock	
		0	Stop prescaler 1 clock
		1	Run prescaler 1 clock
7 0	PRESCALER1 [7:0]	The prescaler value of unit 1 $\text{PWMCLK1} = \frac{\text{PCLK}}{\text{PRESCALER1} + 1}$	

9.5.1.7 PWM Register Map Summary

Table 89. PWM Register Map Summary

Offset	Register name	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0x00	PWMn_CTRL	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	PRF	Res.	Res.	Res.	PRIE	CKSEL[2:0]			Res.	INVA	Res.	SYNC	STRT
	Reset value																				0				0	0	0	0	0	0	0	0	0
0x04	PWMn_CNT	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	CNT[15:0]															
	Reset value																	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
0x08	TIMERn_PER	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	PERIOD[15:0]															
	Reset value																	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
0x0C	TIMERn_CMP	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	CMP[15:0]															
	Reset value																	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
0x20A0	PWM_PRS0	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	CLKEN0	Res.	Res.	Res.	Res.	Res.	Res.	Res.	PRESCALER0[7:0]							
	Reset value																	0								0	0	0	0	0	0	0	0
0x21A0	PWM_PRS1	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	CLKEN1	Res.	Res.	Res.	Res.	Res.	Res.	Res.	PRESCALER1[7:0]							
	Reset value																	0								0	0	0	0	0	0	0	0

10. 16-bit Timer

10.1 16-bit Timer Introduction

The A34G43x series has a TIMER module each configured with 10 units. This 16-bit Timer supports four operating modes such as Periodic mode, PWM mode, One-shot mode, and Capture mode.

Users can use a divided PCLK or an external clock as an input clock source for the 16-bit Timer. An internal 10-bit prescaler allows to generate a variety of timer base clocks.

Interrupts can be triggered at regular intervals when the timer is used in Periodic Mode. Users can set the period and duty to form a PWM signal that is used in PWM Mode.

In One-shot Modes, the timer can generate one PWM waveform. In Capture Mode, the external input signal’s pulse intervals can be measured based on the preset condition. Moreover, the timer can export signals to other devices to control them. This timer is primarily used as periodic tick timer or wake-up source.

10.2 16-bit Timer Main Features

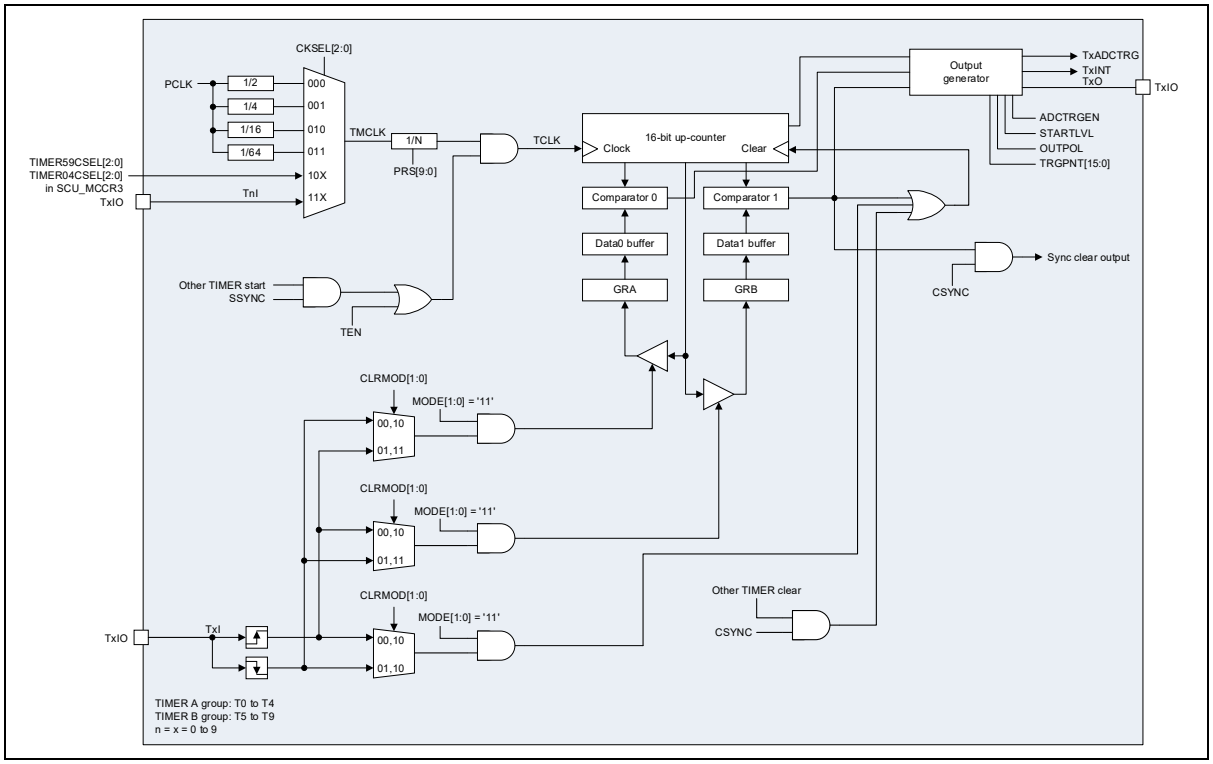
TIMERn (n = 0 to 9) includes the features below:

- 16-bit up-counter timers
- Four operating modes:
  - Periodic mode
  - One-shot mode
  - PWM mode
  - Capture mode
- Various interrupts:
  - Match interrupt
  - Overflow interrupt
- Various clock sources selectable as a timer input:
  - Four PCLK prescaler levels (1/2, 1/4, 1/16, and 1/64)
  - External clock sources selectable using the SCU_MCCR3 register: LSI, LSE, MCLK, HSI, HSE, and PLL
  - Timer clock source on port TnIO used as input. (n = 0 to 9)
- A built-in 10-bit prescaler supporting the timer input clock
- PWM synchronization
  - Start delay and clear synchronization

10.3 16-bit Timer Functional Description

10.3.1 Block Diagram

Figure 68. 16-bit Timer Block Diagram (n = 0 to 9)



10.3.2 Pins and Internal Signals

Table 90 summarizes the I/O information of the TIMERn (n = 0 to 9).

Table 90. Input and Output Pins for TIMERn (n = 0 to 9)

Pin Name	Type	Description	Supported Packages		
			A34G43xYL (120-LQFP)	A34G43xVL (100-LQFP)	A34G43xRL (64-LQFP)
T0IO	IO	TIMER0 capture input signal/external clock input TIMER0 periodic/pwm/one-shot output	O	O	O
T1IO	IO	TIMER1 capture input signal/external clock input TIMER1 periodic/pwm/one-shot output	O	O	O
T2IO	IO	TIMER2 capture input signal/external clock input TIMER2 periodic/pwm/one-shot output	O	O	O
T3IO	IO	TIMER3 capture input signal/external clock input TIMER3 periodic/pwm/one-shot output	O	O	O
T4IO	IO	TIMER4 capture input signal/external clock input TIMER4 periodic/pwm/one-shot output	O	O	O
T5IO	IO	TIMER5 capture input signal/external clock input TIMER5 periodic/pwm/one-shot output	O	O	X
T6IO	IO	TIMER6 capture input signal/external clock input TIMER6 periodic/pwm/one-shot output	O	O	X
T7IO	IO	TIMER7 capture input signal/external clock input TIMER7 periodic/pwm/one-shot output	O	O	X
T8IO	IO	TIMER8 capture input signal/external clock input TIMER8 periodic/pwm/one-shot output	O	O	O
T9IO	IO	TIMER9 capture input signal/external clock input TIMER9 periodic/pwm/one-shot output	O	O	O

Table 91. Internal Input and Output Signals for TIMERn (n = 0 to 9)

Internal Signal Name	Signal Type	Description
PCLK	Input	Peripheral system clock
EXT_CLK	Input	Timer external clock (Selectable with the setting of SCU_MCCR3)
TnI	Input	Timer n input to TnIO for external clock or capture input
CKSEL[2:0] in TIMERn_CR1	Input	Counter clock source selection bit in the Timer/counter Control Register 1
TIMERn_PRS	Input	Prescaler value for the counter clock
TEN in TIMERn_CR2	Input	Enable timer signal in the Timer/counter Control Register 2
SSYNC in TIMERn_SYNC	Input	Synchronized counter start mode signal in the Timer/counter Sync Configuration Register
CSYNC in TIMERn_SYNC	Input	Synchronized counter clear mode signal in the Timer/counter Sync Configuration Register
ADCTRGEN in TIMERn_CR1	Input	Uses the timer as an ADC trigger source signal in the Timer/counter Control Register 1
STARTLVL in TIMERn_CR1	Input	Starting output value signal in the Timer/counter Control Register 1
OUTPOL in TIMERn_CR1	Input	Output polarity signal in the Timer/counter Control Register 1
TnADTRG	Output	Timer n ADC trigger source signal
TnINT	Output	Timer n interrupt signal
TnO	Output	Timer n output signal to TnIO
SYNC CLEAR OUT	Output	Synchronized counter clear signal

NOTE:

1. n = 0 to 9.

10.3.3 Clock Selection

The counter clock can be provided by the following clock sources:

- Peripheral system clock (PCLK)
- Timer external clock (SCU_MCCR3 register)
- Input to pin TnIO (n = 0 to 9) for timer clock source

Users can select a clock source by configuring the CKSEL[2:0] in TIMERn_CR (n = 0 to 9).

10.3.3.1 Peripheral System Clock (PCLK)

Users can select a timer input clock source using four PCLK divider levels (PCLK / 2, PCLK / 4, PCLK / 16, PCLK / 64). If the CKSEL[2:0] bit in the TIMERn_CR1 register is set to a number between 0 and 3, the timer clock is provided by the PCLK divided level.

Table 92. PCLK Divide Levels and CKSEL[2:0] in TIMERn_CR1

CKSEL[2:0] in TIMERn_CR1	Counter Clock Source Selection
000	PCLK / 2
001	PCLK / 4
010	PCLK / 16
011	PCLK / 64

10.3.3.2 Timer External Clock (SCU_MCCR3 Register)

Users can select a timer input clock source by configuring the SCU_MCCR3 register. The selectable clock sources include LSI, LSE, MCLK, HSI, HSE and PLL. If the CKSEL[2:0] bits in the TIMERn_CR1 is set to ‘10X’, the timer clock is provided by the SCU_MCCR3 register.

Table 93. SCU_MCCR3 Register and CKSEL[2:0] in TIMERn_CR1

CKSEL[2:0] in TIMERn_CR1	Counter Clock Source Selection
10X	EXT_CLK (SCU_MCCR3 register)

EXT_CLK can be set by the SCU_MCCR3 register, Miscellaneous Clock Control Register 3. Because the PCLK is the standard of operation, the SCU_MCCR3 value must be set slower than the PCLK, and therefore, it must be divided at least three times if the EXT_CLK clock source uses the same frequency as the PCLK. If the PCLK is 1/2 slower than the EXT_CLK clock source, the EXT_CLK clock source must be divided at least six times.

10.3.3.3 Input to Pin TnIO for Timer Clock Source

Users can select a timer input clock source using input on the port TnIO (n = 0 to 9). If the IOSEL bit in the TIMERn_CR1 is set to ‘0’, the TnIO pin is set as input.

Table 94. TnIO and IOSEL in TIMERn_CR1

IOSEL in TIMERn_CR1	Description
0	Sets the pin as an input port.
1	Sets the pin as an output port.

If the CKSEL[2:0] bits in the TIMERn_CR1 are set to ‘11X’, the timer clock is provided by input on the port TnIO.

Table 95. TnIO and CKSEL[2:0] in TIMERn_CR1

CKSEL[2:0] in TIMERn_CR1	Counter Clock Source Selection
11X	Input to pin TnIO (n = 0 to 9)

10.3.4 Time-Base Unit

The main block of the programmable 16-bit timer is a timer/counter with its related timer/counter count register. The counter increments based on the specified input clock. The counter clock can be divided by a prescaler.

The time-base unit includes the following registers:

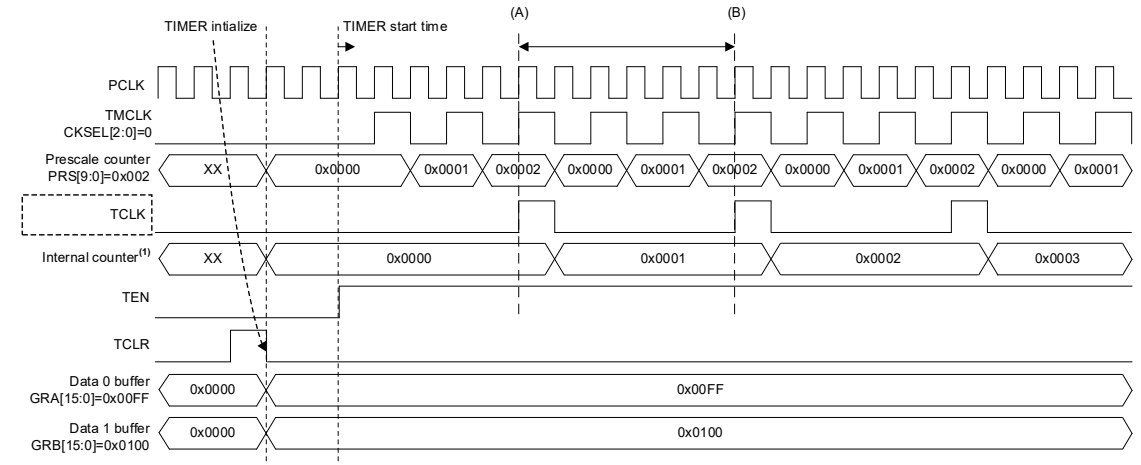
- Timer Counter Register: TIMERn_CNT (n = 0 to 9)
- Timer Prescaler Register: TIMERn_PRS (n = 0 to 9)
- Timer General Data Register A: TIMERn_GRA (n = 0 to 9)
- Timer General Data Register B: TIMERn_GRB (n = 0 to 9)
- Miscellaneous Clock Control Register 3: SCU_MCCR3

10.3.4.1 Prescaler Description

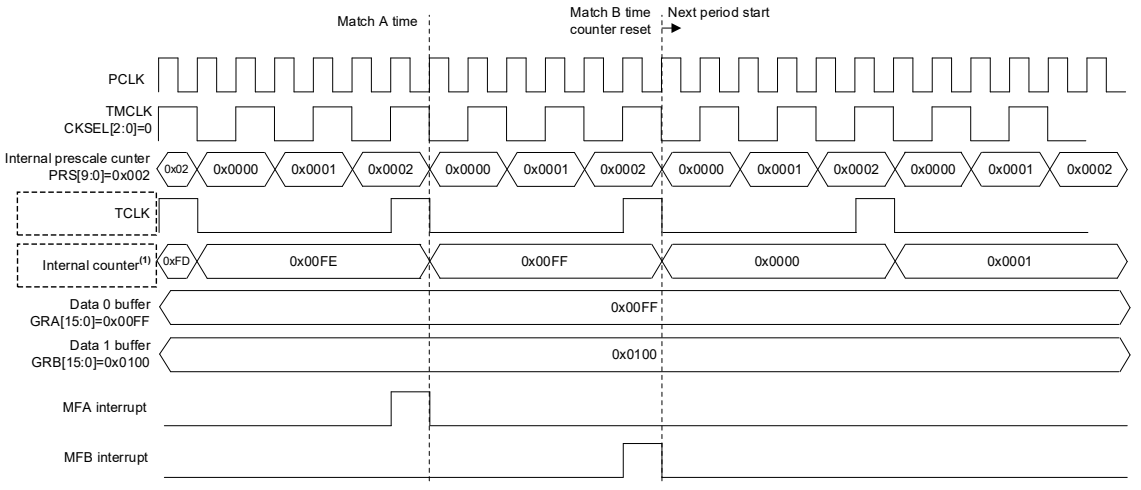
The prescaler is used to set the timer input frequency divider. It is 10-bit wide. Users can generate precise and varied timer base clocks by applying the prescaler to the timer clock source that has been selected in TIMERn_CR1 register.

The TMCLK shown in Figure 69 is a reference clock for operating the timer. The frequency of this clock can be divided by setting the prescaler to operate a counting clock. This figure shows the start and end points of a counter in normal periodic mode. When changing the timer settings or restarting the timer with a new value, it is recommended that users set the TCLR in TIMERn_CR2 before setting the TEN in TIMERn_CR2.

Figure 69. Counter Timing Diagram with Basic Start and Match Operations



(a) Timer initialization is done by TCLR command and timer will be started by TEN command.



(b) Timer will be reset by match GRB timing and count again from '00'.

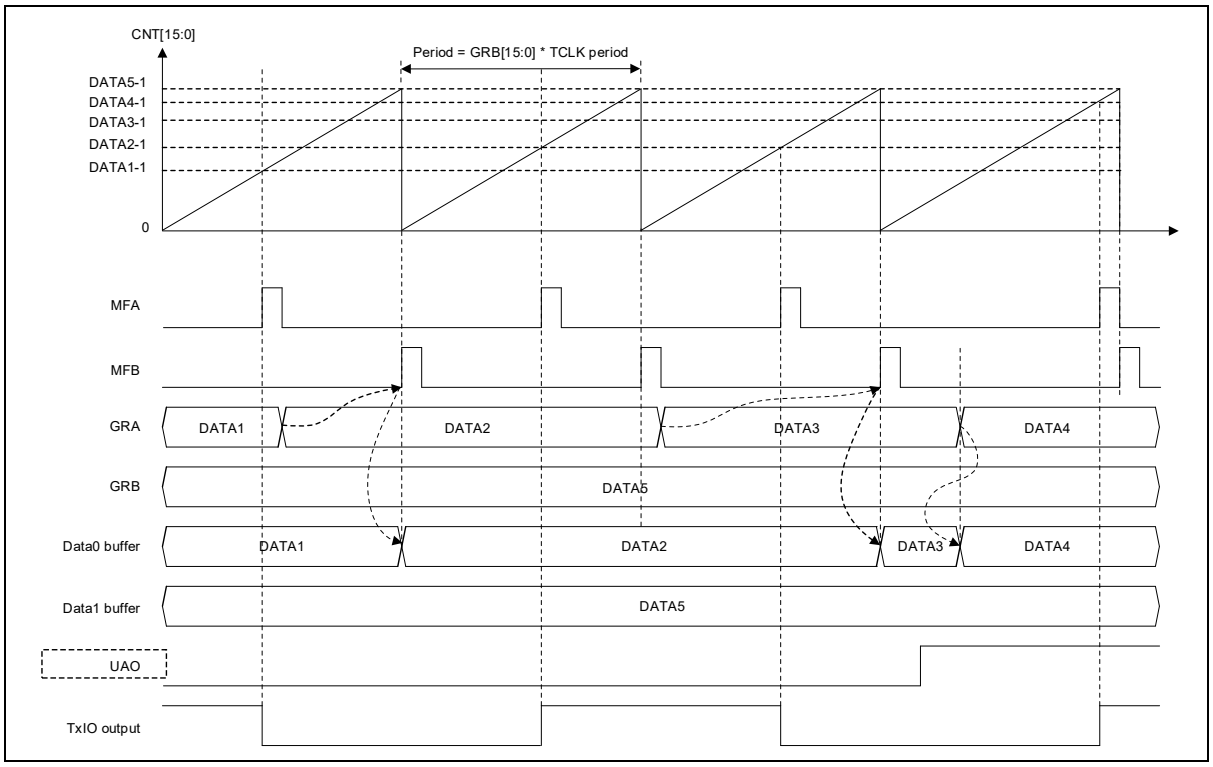
NOTE:

1. Internal counter clock period (A~B) is PCLK x (2 x (2 + 1)), that is, 6 x PCLK.

10.3.5 Periodic Mode

Figure 70 shows a timing diagram for normal periodic mode operation. The TIMERn_GRA value determines the timer period. At this time, the TIMERn_GRB value must be set to be equal to the TIMERn_GRA value.

Figure 70. Normal Periodic Mode Operation



The timer's count period can be calculated by using the formula below:

- Period = TCLK period × TIMERn_GRA value
- Match A interrupt time = TCLK period × TIMERn_GRA value

If the TIMERn_GRA value is '0, the timer cannot be started even if the TEN in TIMERn_CR2 is set to '1' because the period is zero.

Values of the TIMERn_GRA and TIMERn_GRB registers are loaded into the internal compare data buffers 0 and 1 when the loading condition is satisfied. In this periodic mode with the UAO bit in the TIMERn_CR1 set value of '0', it will load the TIMERn_GRA and TIMERn_GRB values to the compare data buffers when user write '1' to the TCLR bit in the TIMERn_CR2 register or the GRB match event occurred.

When the UAO bit in the TIMERn_CR1 register is set to ‘1’, the internal compare data buffer is updated whenever the TIMERn_GRA or TIMERn_GRB data is updated. The TnIO output signal will be toggled at every match A condition time.

If the TIMERn_GRA value is zero and the TIMERn_GRB value is not zero, the TnIO output does not change its previous level. If the TIMERn_GRA value is the same as the TIMERn_GRB value, the TnIO output will toggle simultaneously depending on the counter clear time. The initial level of the TnIO signal is decided by the STARTLVL bit in TIMERn_CR1 register.

10.3.5.1 Operation According to TIMERn_GRA and TIMERn_GRB Settings

- For normal counter mode operation, users must enter the same value in the TIMERn_GRA and TIMERn_GRB registers.
  - Please refer to Table 96 according to the TIMERn_GRA and TIMERn_GRB register settings.
- To use the Timer Overflow function, users must write ‘0’ to both the TIMERn_GRA and TIMERn_GRB registers.
  - If the TIMERn_GRA register or the TIMERn_GRB register has a value other than ‘0’, the Match flag occurs.
  - To generate the normal OverFlow flag, write ‘0’ to both the TIMERn_GRA and TIMERn_GRB registers.

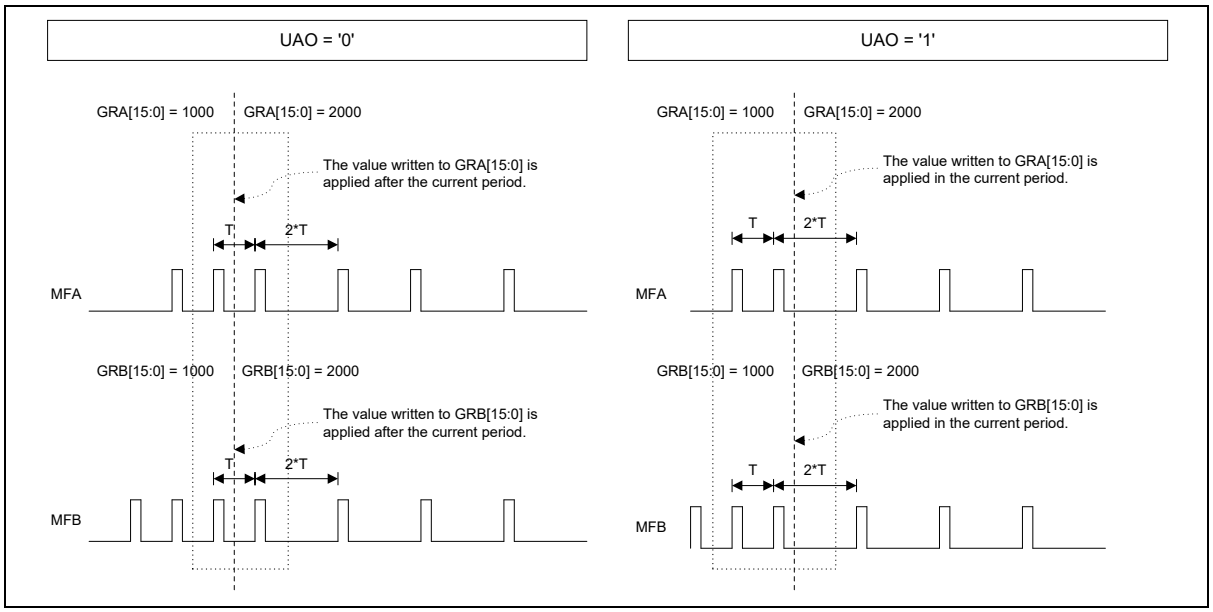
Table 96. Operations Comparison according to TIMERn_GRA/GRB Settings

Case	GRA[15:0]	GRB[15:0]	UAO	OUTPOL	IOSEL	STARTLVL	TnIO Pin Output
1	1,000	1,000	0/1	0	1	0	Normal operation
2	1,000	2,000	0/1	0	1	0	Outputs the GRB[15:0] setting value.
3	1,000	0	0/1	0	1	0	Outputs the GRB[15:0] setting value.
4	1,000	1 to 999	0/1	0	1	0	No output
5	1,000	65,535	0/1	0	1	0	Outputs the GRB[15:0] setting value.

10.3.5.2 Operation According to UAO, OUTPOL, and STARTLVL

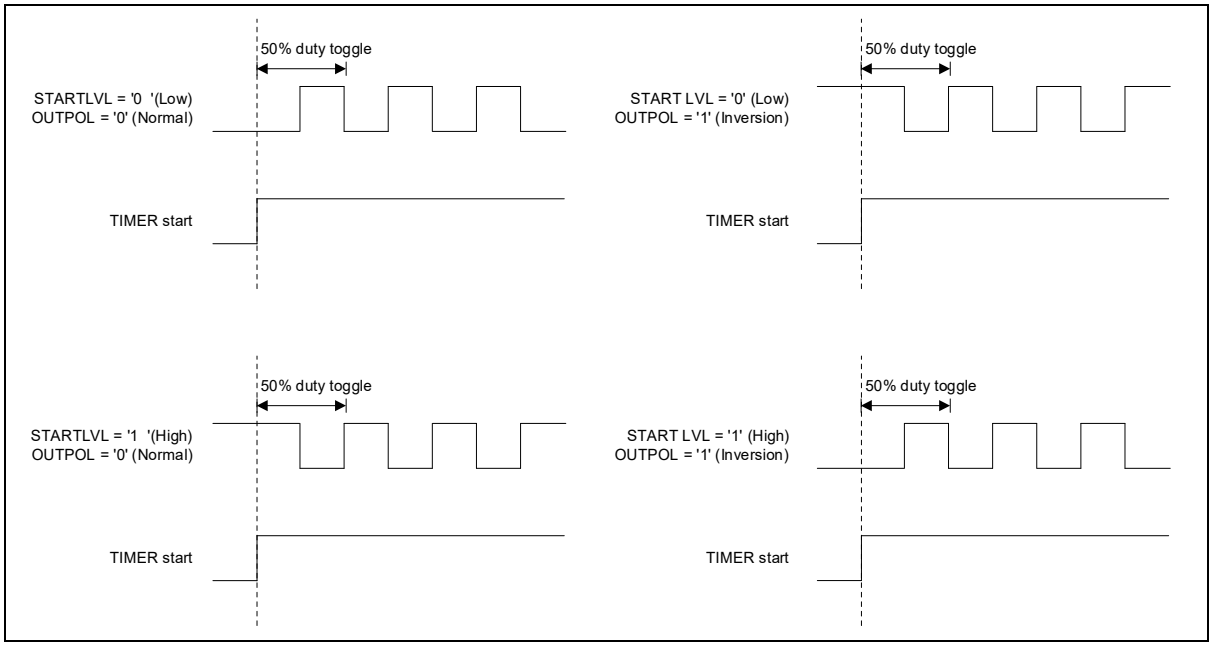
- Refer to the Figure 71 for the operation waveform according to the UAO in TIMERn_CR1 settings.
  - If the UAO bit in TIMERn_CR1 value is ‘0’, the TIMERn_GRA/GRB value which has been updated after the current cycle operation is applied.
  - If the UAO bit in TIMERn_CR1 value is ‘1’, the TIMERn_GRA/GRB value is applied within the current cycle operation.

Figure 71. Operation Waveform Depending on UAO Bit



- Refer to the Figure 72 below for the operation waveform according to the STARTLVL bit in TIMERn_CR1 register, OUTPOL bit in TIMERn_CR1 register settings (n = 0 to 9).
  - The STARTLVL bit in the TIMERn_CR1 register is reflected in TnIO output by the TCLR bit in the TIMERn_CR2 register.
  - The OUTPOL bit in the TIMERn_CR1 register is applied when the timer counter is cleared after setting. (Applied after the current cycle operation.)

Figure 72. Operation Waveform According to STARTLVL and OUTPOL Settings



10.3.5.3 Additional Precautions Over Counter Mode Operation

1. If the timer counter is cleared during timer operation, the output of the timer pin (TnIO, n = 0 to 9) is changed.
2. The timer counter is cleared when the TIMERn_CNT register is forcibly cleared or when the TIMERn_GRB register is matched.
3. When the timer stops, the Timer pin (TnIO, n = 0 to 9) output maintains the status at the time of stopping, and when the timer counter is cleared, it is changed to the setting value of the STARTLVL bit in TIMERn_CR1 register.

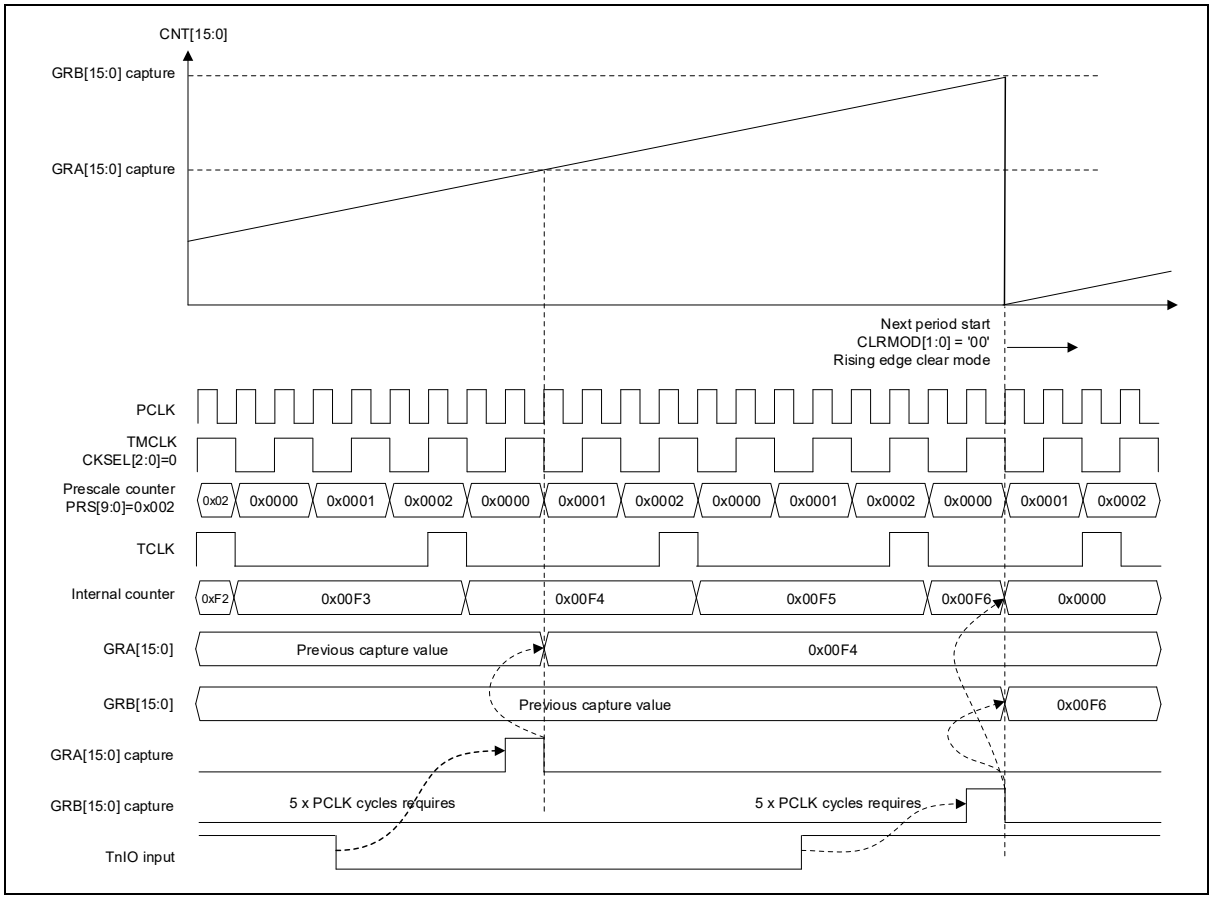
10.3.5.4 Counter Mode Register Setting Sequence

1. To enable the system access, write a reserved value to the corresponding bit of the System access enable register (SCU_SYSTEN).
2. To enable the timer module, write '1' to the corresponding bit of the Peripheral Enable Register 1 (SCU_PER1) and the Peripheral Clock Enable Register 1 (SCU_PCER1).
3. Write zero to the TIMERn_CNT register to initialize the timer counter.
4. To set the timer mode, enter the corresponding mode set value in the MODE[1:0] bits in the TIMERn_CR1 register.
5. To set the timer clock source, enter the corresponding set value in the CKSEL[2:0] bits in the TIMERn_CR1 register.
6. To generate a more precision base clock, apply a prescaler to the timer clock source and enter the set value in the TIMERn_PRS register.
7. Write '1' to the IOSEL bit in the TIMERn_CR1 register to set the timer input/output pin (TnIO, n = 0 to 9).
8. Enter the same set value in the TIMERn_GRA and TIMERn_GRB registers for period timer operation.
9. Write '1' to the corresponding bit of the TIMERn_SR register to clear the pending interrupt.
10. Write '1' to the MAIE bit in the TIMERn_IER register to set the timer match interrupt.
11. Write '1' to the TLCR bit in the TIMERn_CR2 register to apply current timer setting.
12. Write '1' to the TEN bit in the TIMERn_CR2 register to start the timer operation.

10.3.6 Capture Mode

Figure 73 shows a timing diagram for the capture mode operation. The TnIO input signal is used for capturing the pulse. Rising and falling edges can capture the counter value in each capture condition.

Figure 73. Timing Diagram of Capture Mode Operation



The actual capture point is five PCLK clock cycles after the rising or falling edge of the TnIO input signal. The internal counter can be cleared in multiple modes. The CLRMOD[1:0] bits in the TIMERn_CR1 controls counter clearing in capture mode. The supported modes include rising-edge clear mode, falling-edge clear mode, both-edge clear mode, and non-clear mode.

Example timing diagram in Figure 73 describes rising-edge clear mode. On the falling edge of the input signal on the TnIO, the TIMERn_GRA register captures the CNT value; on the rising edge, the TIMERn_GRB register captures the CNT value.

NOTE:

1. Since the timer input operates as a functional pin of a GPIO port, users must first set the GPIO port to function pin mode. When the GPIO port is set to function pin mode, it operates as an input port or an output port according to the timer operation mode.

To use the capture mode, set the GPIO port to timer function pin mode (TnIO, n = 0 to 9) and then set the IOSEL bit in the TIMERn_CR1 register to ‘0’.

Table 97. Operation Status by Capture Mode

CLRMOD[1:0] in TIMERn_CR1	Clear Mode	TIMERn_GRA data	TIMERn_GRB data
00	Rising edge clear	Falling edge capture	Rising edge capture
01	Falling edge clear	Rising edge capture	Falling edge capture
10	Both edge clear	Falling edge capture	Rising edge capture
11	Disable clear	Rising edge capture	Falling edge capture

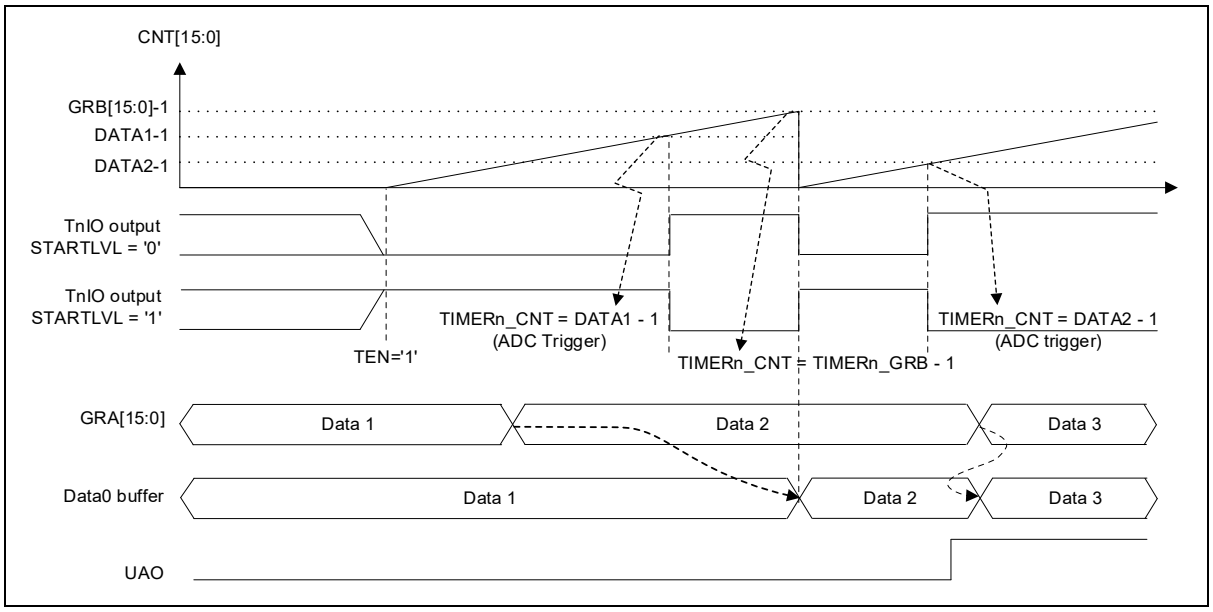
10.3.6.1 Capture Mode Register Setting Sequence

1. To enable the system access, write a reserved value to the corresponding bit of the System access enable register (SCU_SYSTEN).
2. To enable the timer module, write ‘1’ to the corresponding bit of the peripheral enable register1 (SCU_PER1) and the peripheral clock enable register1 (SCU_PCER1).
3. Initialize the TIMERn_GRA and TIMERn_GRB registers to zero.
4. Write ‘0’ to the TIMERn_CNT register to initialize the timer counter.
5. To set the timer mode, enter the corresponding mode set value in the MODE[1:0] bits in the TIMERn_CR1 register.
6. To set the timer clock source, enter the corresponding set value in the CKSEL[2:0] bits in the TIMERn_CR1 register.
7. To generate a more precision base clock, apply a prescaler to the timer clock source and enter the set value in the TIMERn_PRS register.
8. Write ‘0’ to the IOSEL bit in the TIMERn_CR1 register to set the timer input/output pin (TnIO, n = 0 to 9).
9. For the clear selection in capture mode, enter the corresponding set value in the CLRMOD[1:0] bits in the TIMERn_CR1 register.
10. Write ‘1’ to the TLCR bit in the TIMERn_CR2 register to apply current timer setting.
11. Write ‘1’ to the TEN bit in the TIMERn_CR2 register to start the timer operation.

10.3.7 PWM Mode

Figure 74 shows a timing diagram for the PWM output mode operation. The TIMERn_GRB value decides the PWM pulse period. An additional comparison point is provided by the TIMERn_GRA register value which defines the pulse width of PWM output. (n = 0 to 9)

Figure 74. Timing Diagram of PWM Output Operation



The PWM pulse period can be calculated by using the formula below:

- Period = TCLK period × TIMERn_GRB value
- Duty = TCLK period × TIMERn_GRA value

If the TIMERn_GRB value is zero, the timer cannot be started even if the TEN bit in the TIMERn_CR2 register is set to ‘1’ because the period is zero.

Values of the TIMERn_GRA and TIMERn_GRB registers are loaded into the internal compare data buffers 0 and 1 when the loading condition is satisfied. In this periodic mode with the UAO bit in the TIMERn_CR1 register set value of ‘0’, it will load TIMERn_GRA and TIMERn_GRB values to the compare data buffers when user write '1' to the TCLR bit in the TIMERn_CR2 register or the GRB match event occurred.

When the UAO bit in the TIMERn_CR1 register is set to ‘1’, the internal compare data buffer is updated whenever the TIMERn_GRA or TIMERn_GRB data is updated.

The TnIO output signal generates a PWM pulse.

The TIMERn_GRB value defines the output pulse period and the TIMERn_GRA value defines the pulse width of PWM pulse. The active level of the PWM pulse can be controlled by the STARTLVL bit in the TIMERn_CR1 register.

ADC trigger generation is available at Match A interrupt time.

10.3.7.1 PWM Mode Operation Characteristics and Precautions

Precautions for setting the TIMERn_GRA register are shown below:

- The TIMERn_GRA value must be greater than '0' and less than or equal to the TIMERn_GRB value. (0 < TIMERn_GRA ≤ TIMERn_GRB)
- If the TIMERn_GRA value is zero, the GRA matching event does not occur, and the output does not change.

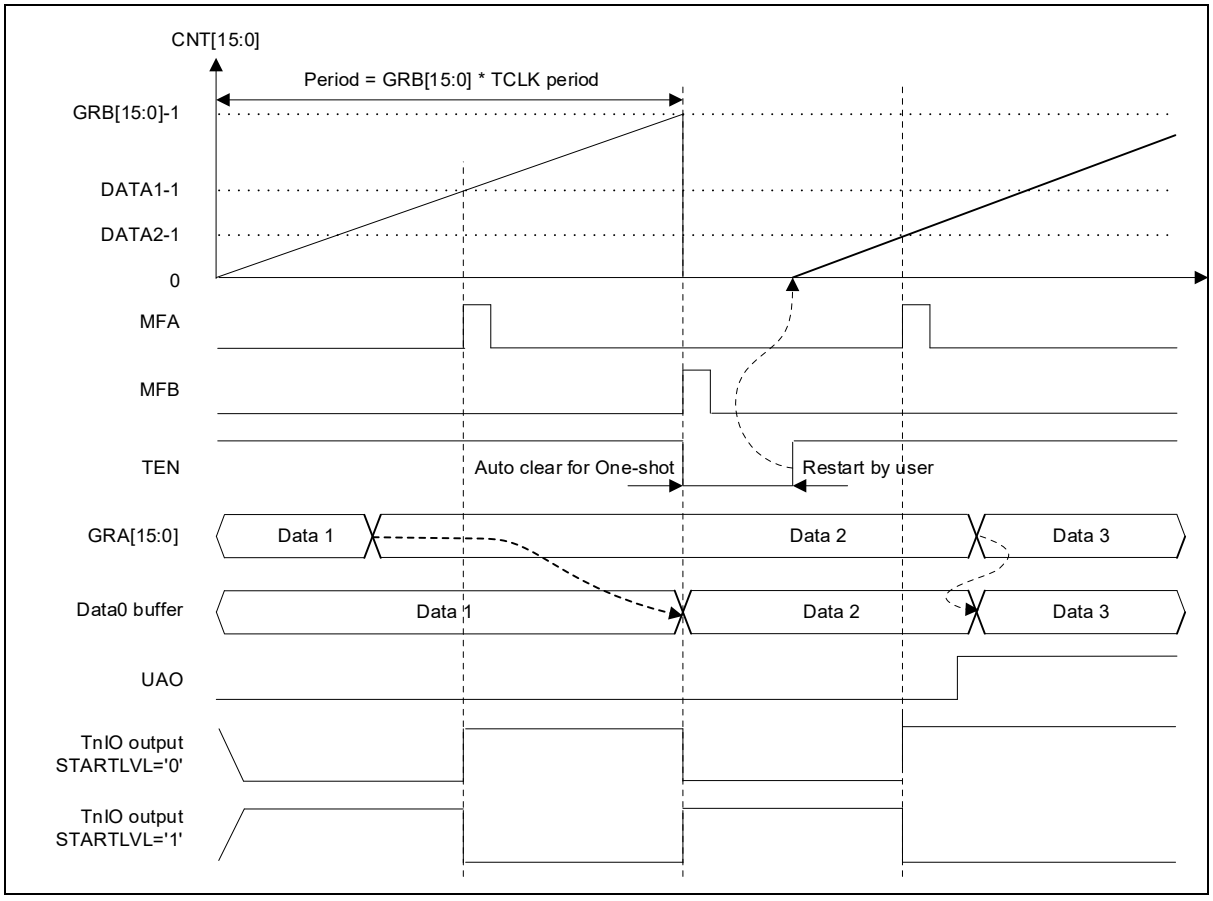
10.3.7.2 PWM Mode Register Setting Sequence

1. To enable the system access, write a reserved value to the corresponding bit of the System access enable register (SCU_SYSTEN).
2. To enable the timer module, write '1' to the corresponding bit of the peripheral enable register1 (SCU_PER1) and the Peripheral Clock Enable Register1 (SCU_PCER1).
3. Write zero to the TIMERn_CNT register to initialize the timer counter.
4. To set the timer mode, enter the corresponding mode set value in the MODE[1:0] bits in the TIMERn_CR1 register.
5. To set the timer clock source, enter the corresponding set value in the CKSEL[2:0] bit in the TIMERn_CR1 register.
6. To generate a more precision base clock, apply a prescaler to the timer clock source and enter the set value in the TIMERn_PRS register.
7. Write '1' to the IOSEL bit in the TIMERn_CR1 register to set the timer input/output pin (TnIO, n = 0 to 9).
8. Enter the set values in the TIMERn_GRA and TIMERn_GRB registers for PWM operation.
9. Write '1' to the TLCR bit in the TIMERn_CR2 register to apply current timer setting.
10. Write '1' to the TEN bit in the TIMERn_CR2 register to start the timer operation.

10.3.8 One-Shot Mode

Figure 75 shows a timing diagram for the one-shot mode operation. The TIMERn_GRB value determines the one-shot period, and the TIMERn_GRA value provides another comparative point.

Figure 75. Timing Diagram of One-Shot Mode Operation



The one-shot count period can be calculated by using the formula below:

- Period = TCLK period × TIMERn_GRB value
- Duty = TCLK period × TIMERn_GRA value

If the TIMERn_GRB value is zero, the timer cannot be started even if the TEN bit in the TIMERn_CR2 register is set to ‘1’ because the period is zero.

Values of the TIMERn_GRA and TIMERn_GRB registers are loaded into the internal compare data buffers 0 and 1 when the loading condition is satisfied. In this one-shot mode with the UAO bit in the TIMERn_CR1 register set value of ‘0’, it will load TIMERn_GRA and TIMERn_GRB values to the compare data buffers when user write '1' to the TCLR bit in the TIMERn_CR2 register or the GRB match event occurred.

When the UAO bit in the TIMERn_CR1 register is set to '1', the internal compare data buffer is updated whenever the TIMERn_GRA or TIMERn_GRB data is updated.

The TnIO output signal format is the same as in PWM mode. The TIMERn_GRB value defines the output pulse period and the TIMERn_GRA value defines the one-shot pulse width.

10.3.8.1 One-Shot Mode Operation Characteristics and Precautions

- Precautions when setting the TIMERn_GRA register:
  - The TIMERn_GRA value must be greater than zero and less than or equal to the TIMERn_GRB value. (0 < TIMERn_GRA ≤ TIMERn_GRB)
  - If the TIMERn_GRA value is zero, the GRA Matching event does not occur, and the output does not change.
- Operation sequence according to the STARTLVL in TIMERn_CR1 settings:
  - If the STARTLVL is '1', GRA (Duty) High output → GRB-GRA Low output → High output
  - If the STARTLVL is '0', GRA (Duty) Low output → GRB-GRA High output → Low output
  - Refer to the table below that compares the operations according to the STARTLVL and OUTPOL settings.

Table 98. One-Shot Mode Operation Setting Table

Case	GRA[15:0]	GRB[15:0]	UAO	POL	IOSEL	STARTLVL	Remarks
#1	500	2,000	0	0 (N)	1 (Out)	0	Initial output: Low High pulse output between GRA[15:0] and GRB[15:0]
#2	500	2,000	0	0 (N)	1 (Out)	1	Initial output: High Low pulse output between GRA[15:0] and GRB[15:0]
#3	500	2,000	0	1 (I)	1 (Out)	0	Initial output: High Low pulse output between GRA[15:0] and GRB[15:0]
#4	500	2,000	0	1 (I)	1 (Out)	1	Initial output: Low High pulse output between GRA[15:0] and GRB[15:0]

10.3.8.2 One-Shot Mode Register Setting Sequence

1. To enable the system access, write a reserved value to the corresponding bit of the System Access Enable Register (SCU_SYSTEN).
2. To enable the timer module, write ‘1’ to the corresponding bit of the Peripheral Enable Register1 (SCU_PER1) and the peripheral clock enable register1 (SCU_PCER1).
3. Write zero to the TIMERn_CNT register to initialize the timer counter.
4. To set the timer mode, enter the corresponding mode set value in the MODE[1:0] bits in the TIMERn_CR1 register.
5. To set the timer clock source, enter the corresponding set value in the CKSEL[2:0] bit in the TIMERn_CR1 register.
6. To generate a more precision base clock, apply a prescaler to the timer clock source and enter the set value in the TIMERn_PRS register.
7. Write ‘1’ to the IOSEL bit in the TIMERn_CR1 to set the timer input/output pin (TnIO, n = 0 to 9).
8. Enter the set values in the TIMERn_GRA and TIMERn_GRB registers for the one-shot operation.
9. Write ‘1’ to the TLCR bit in the TIMERn_CR2 register to apply current timer setting.
10. Write ‘1’ to the TEN bit in the TIMERn_CR2 register to start the timer operation.

10.3.9 ADC Trigger

The timer module can generate ADC start trigger signals. One timer can be one trigger source of the ADC module. The trigger source is controlled by the ADC control registers.

Figure 76, Figure 77 and Figure 78 show how the ADC triggering works in each mode. The conversion rate must be shorter than the timer period; otherwise, an overrun can occur. ADC acknowledge is not required because the trigger signal is automatically cleared three PCLK clock pulses later.

Figure 76. Timing Diagram of ADC Triggering (TRGMOD[1:0] = '00')

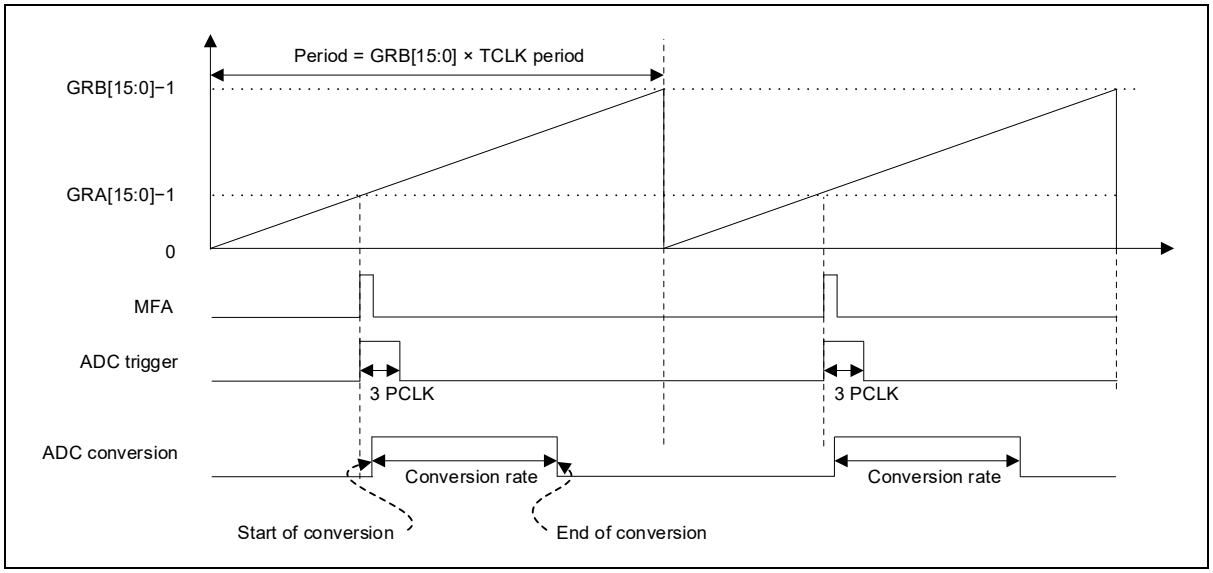


Figure 77. Timing Diagram of ADC Triggering (TRGMOD[1:0] = '10')

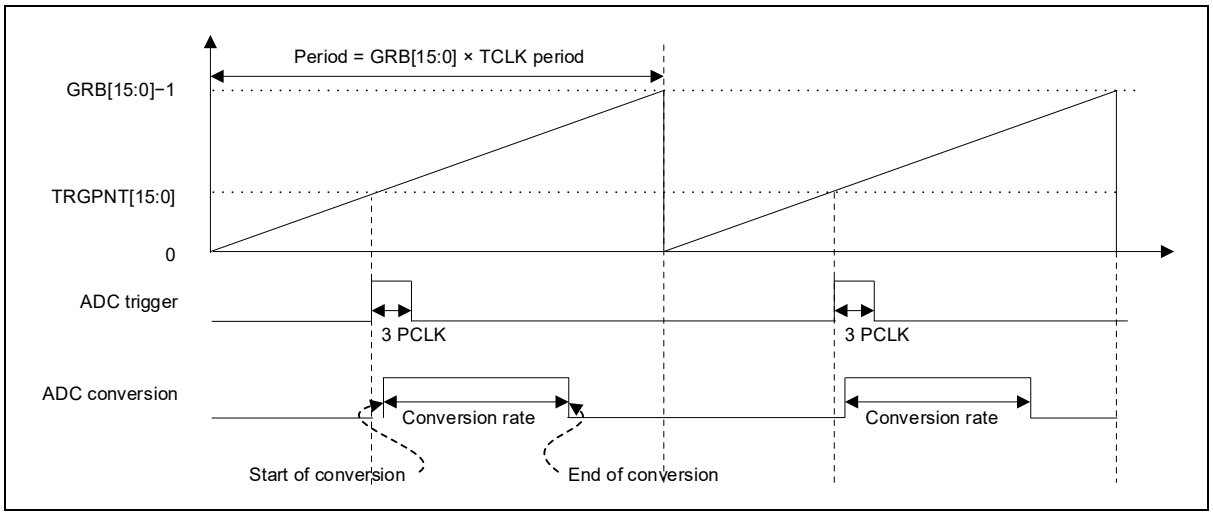
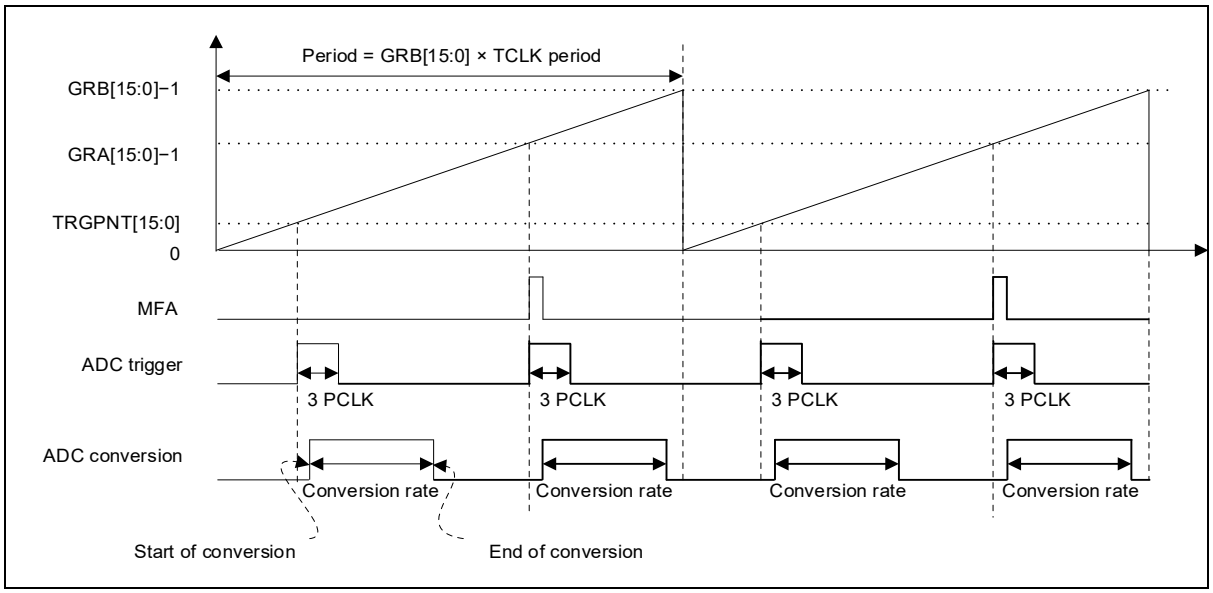


Figure 78. Timing Diagram of ADC Triggering (TRGMOD[1:0] = '11')



10.3.9.1 ADC Trigger Operation Characteristics and Precautions

If the TIMERn_GRB value is less than the TIMERn_GRA value or the TIMERn_TRGPNT value, the timer counter is cleared whenever timer counter matches TIMERn_GRB value. So, the matching event does not occur.

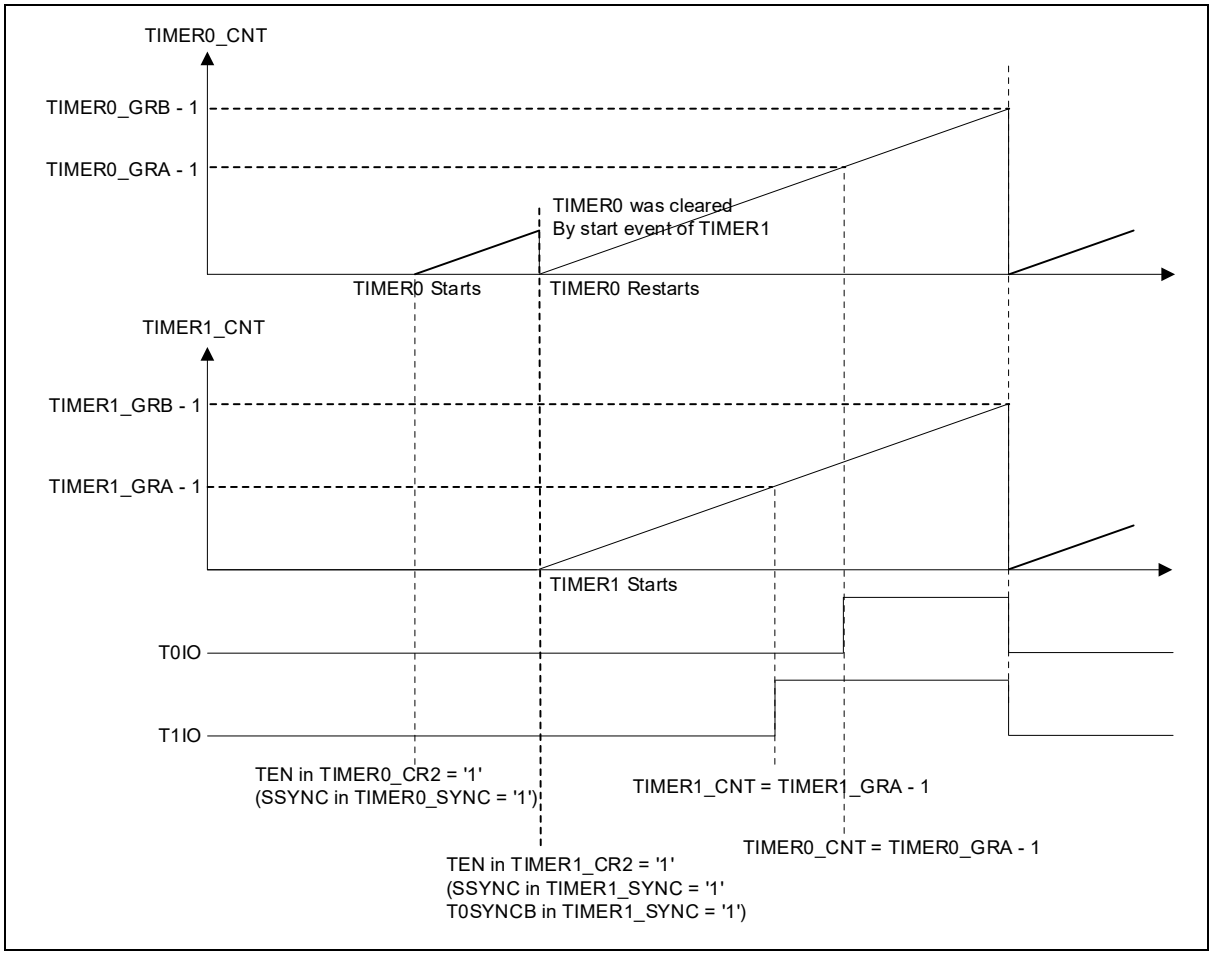
10.3.9.2 Register Setting Sequence for ADC Trigger Operation

1. To enable the system access, write a reserved value to the corresponding bit of the System Access Enable Register (SCU_SYSTEN).
2. To enable the timer module, write '1' to the corresponding bit of the Peripheral Enable Register1 (SCU_PER1) and the peripheral clock enable register1 (SCU_PCER1).
3. Write zero to the TIMERn_CNT register to initialize the timer counter.
4. To set the timer mode, enter the corresponding mode set value in the MODE[1:0] bits in the TIMERn_CR1 register.
5. To set the timer clock source, enter the corresponding set value in the CKSEL[2:0] bits in the TIMERn_CR1 register.
6. To generate a more precision base clock, apply a prescaler to the timer clock source and enter the set value in the TIMERn_PRS register.
7. Enter the corresponding set values in the TIMERn_GRB and TIMERn_GRA registers to set the timer period and trigger point.
8. Write '1' to the ADCTRGEN bit in the TIMERn_CR1 register to use the trigger source.
9. Enter the corresponding mode set value in the TRGMOD[1:0] bits in the TIMERn_CR1 register to select the trigger mode.
10. Enter the set value in the TIMERn_TRGPNT register to set the additional trigger point.
    A. If the TIMERn_GRA register or the TIMERn_TRGPNT register is selected by the TRGMOD[1:0] bits in the TIMERn_CR1 register, when the value set in the selected register matches the TIMERn_CNT value, the ADC is triggered, and its value can be read.
11. Initialize the ADC registers. (Set the TRGSEL[1:0] bits in the ADCn_MR register to '01'.)
12. Write '1' to the TLCR bit in the TIMERn_CR2 register to apply current timer setting.
13. Write '1' to the TEN bit in the TIMERn_CR2 register to start the timer operation.

10.3.10 Timer Synchronization

PWM outputs are usually used as synchronous PWM signal control. This function is provided with synchronous start. Figure 79 shows synchronous PWM generation.

Figure 79. Timer Synchronization Example (When SSYNC = '1', CSYNC = '1')



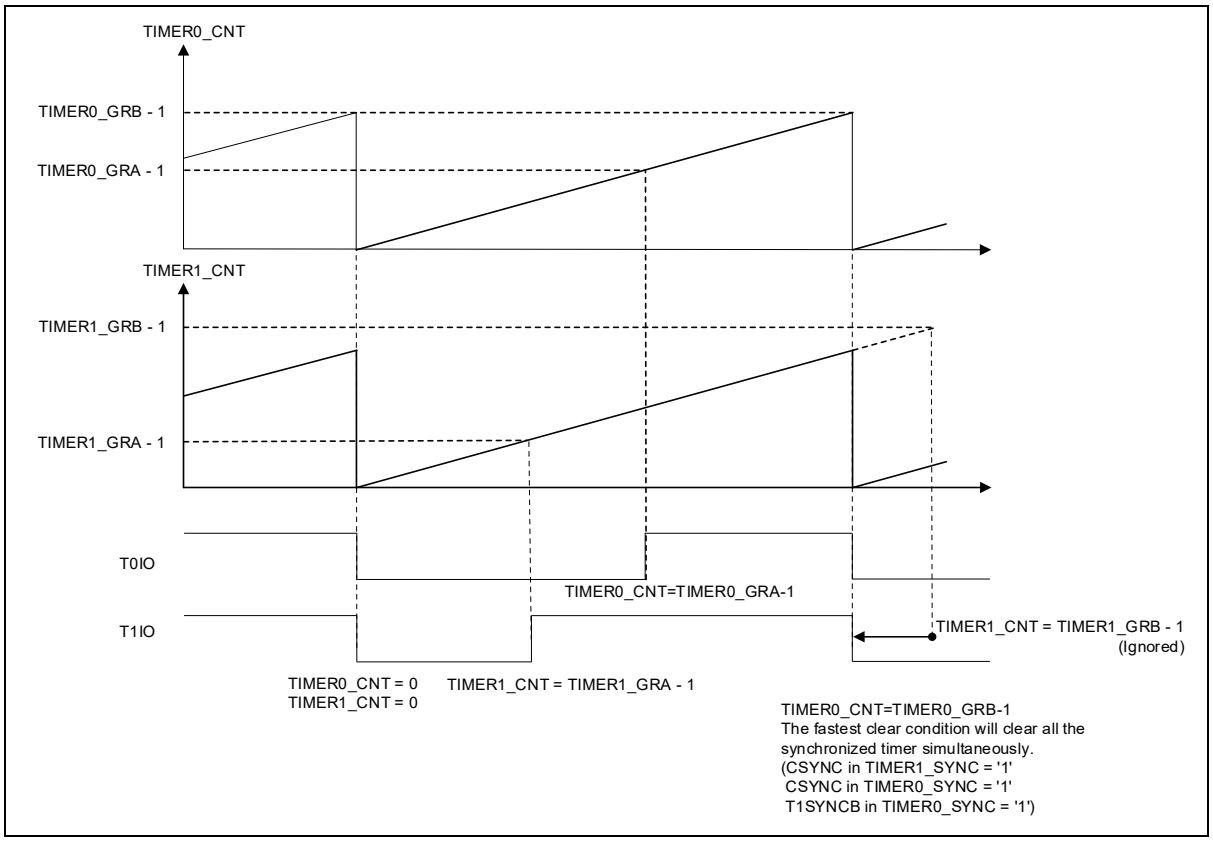
The SSYNC bit in the TIMERn_SYNC register controls start synchronization with other timer modules. In the above figure, the TIMER0 (slave timer) waveform shows that the timer's start is synchronized to the start of the signal from TIMER1 (master timer).

For this SYNC function in the figure above, both the master and slave timers' synchronous start bits must be set enabled: In the master timer's TIMER1_SYNC register, the T0SYNCB and SSYNC bits must be set to '1'. In the slave timer's TIMER0_SYNC register, the SSYNC bit must be set to '1'.

If only the master timer's synchronous start bit is set enabled for the slave timer while the slave's synchronous start bit is disabled, the slave timer can run independently without being affected by the master timer's start synchronization signal.

Additionally, because TIMER0 (slave) operates as synchronized to TIMER1's (master's) start synchronization signal, it runs even if the TEN bit in the TIMER0_CR2 register is set to '0'.

Figure 80. Timer Synchronization Example (When Csync = '1')



The CSYNC bit in the TIMERn_SYNC register controls clear synchronization with other timer modules. In the above figure, the TIMER1 (slave timer) waveform shows that the timer's clear is synchronized to the clear of the signal from the TIMER0 (master timer).

For this CSYNC function in the figure above, both the Master and Slave Timers' synchronous clear bits must be set enabled: In the master timer's TIMER0_SYNC register, the T1SYNCB and CSYNC bits must be set to '1'. In the slave timer's TIMER1_SYNC register, the CSYNC bit must be set to '1'.

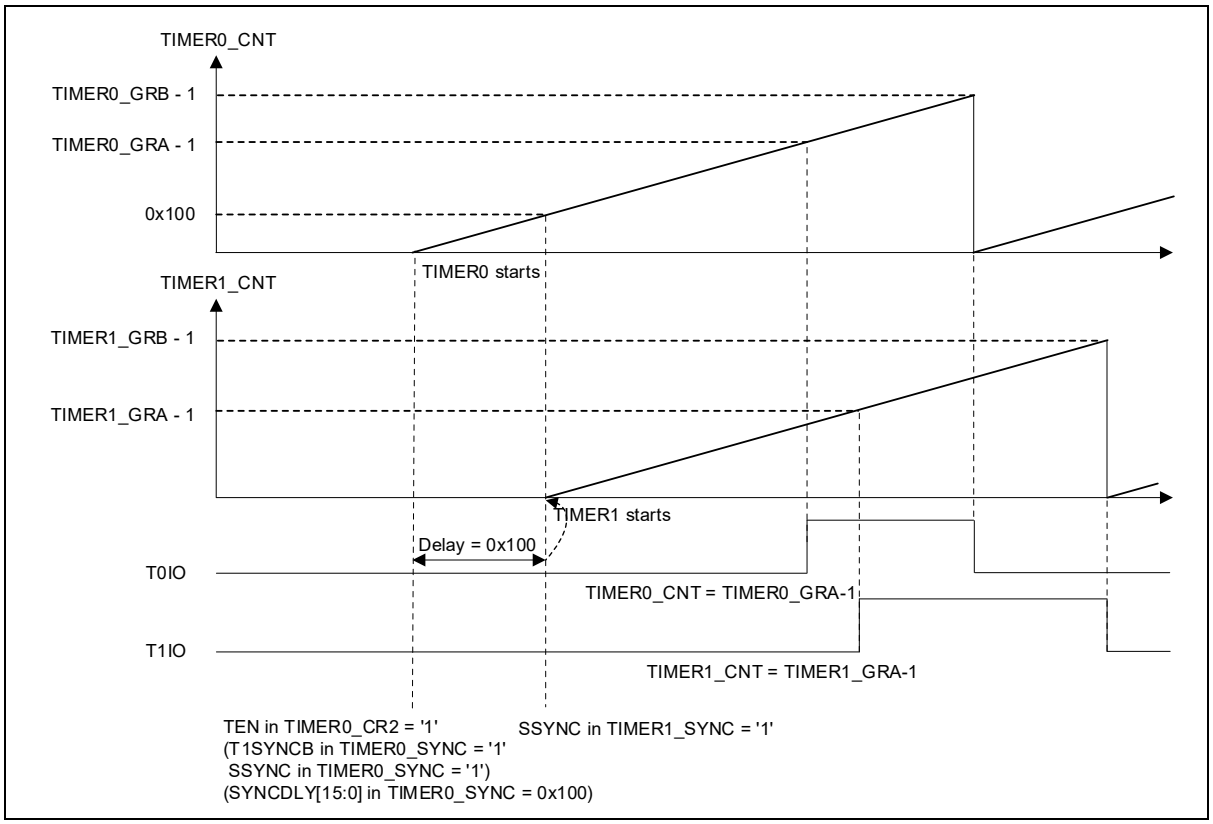
If only the master timer's synchronous clear bit is set enabled for the slave timer while the slave's synchronous clear bit is disabled, the slave timer can run independently without being affected by the master timer's clear synchronization signal.

10.3.10.1 Timer Delayed Synchronization

The PWM delayed synchronization function is used between timers within the same group (group 1: T0, T1, ..., T3 and group 2: T5, T6, …, T9); synchronization is delayed by the amount of time written to the SYNCDLY[15:0] bits in the TIMERn_SYNC.

Using this function, users can have slave timers start sequentially after a certain amount of delay time since the master timer started.

Figure 81. Timer Delayed Synchronization Example (When SSYNC = '1')



For example as shown in Figure 59, to start the TIMER1 (slave) after 0x100 counts from the TIMER0's (master's) synchronization signal output, users must set the master timer's TIMER0_SYNC register to T1SYNCB=1, SSYNC=1, and SYNCDLY[15:0]=0x100. And then, the users must set the slave timer's SSYNC bit in the TIMER1_SYNC register to '1'.

Additionally, because the TIMER1 (slave) operates as synchronized to the TIMER0's (master's) start signal, it runs even if the TEN bit in the TIMERn_CR2 register is set to '0'.

The Start SYNC can be used in cascaded structure, but the Clear SYNC cannot be used in cascaded structure.

10.3.10.2 Timer Synchronization Operation Characteristics and Precautions

1. To use SSYNC, the TEN bit in the TIMERn_CR2 register must be set to ‘0’ for the slave timer. (This synchronizes to the initial operation and can create a sync-delay by cascading multiple timers.)
2. To use CSYNC, the TEN bit in the TIMERn_CR2 register must be set to ‘1’ for the slave timer. (This clears the slave timer counter using the clear signal of the master timer, and timers cannot be cascaded.)
3. The SSYNC is a signal that is generated once at the initial stage, and the CSYNC is a signal that is generated at every cycle.
4. If the SSYNC and CSYNC are set at the same time, the timer operates based on the SSYNC at the initial operation, and afterwards, the timer counter is cleared by the CSYNC.

Figure 82 and Figure 83 show the operation waveform of the SSYNC and CSYNC signals in cascaded structure. The timers in the figures below are cascaded in the order of TIMER0 → TIMER1 → TIMER3.

Figure 82. SSYNC Operation Example

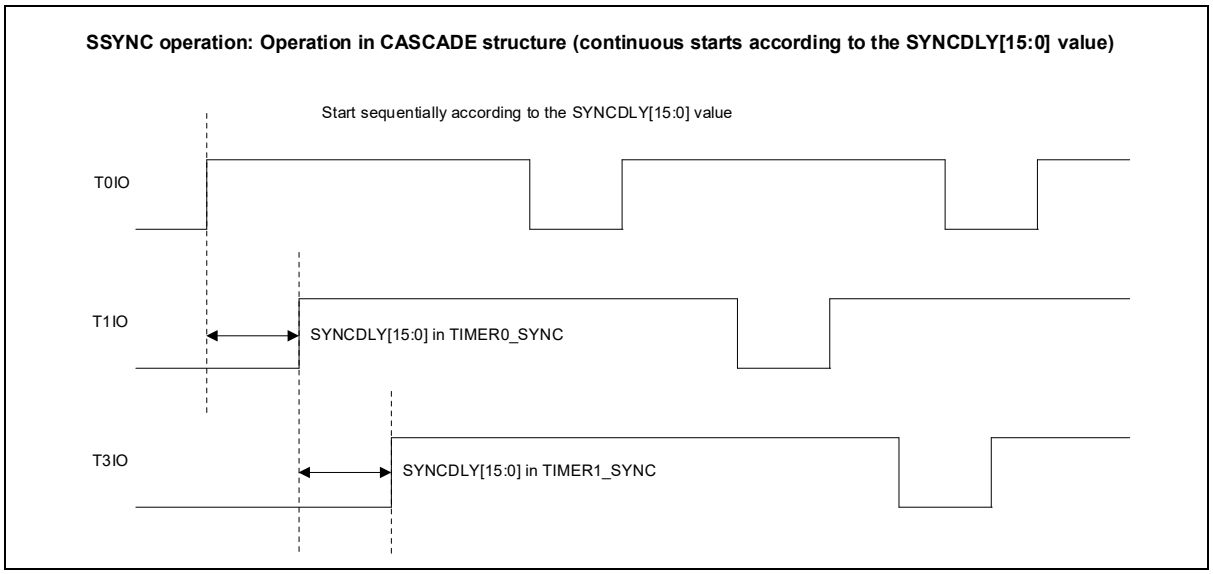
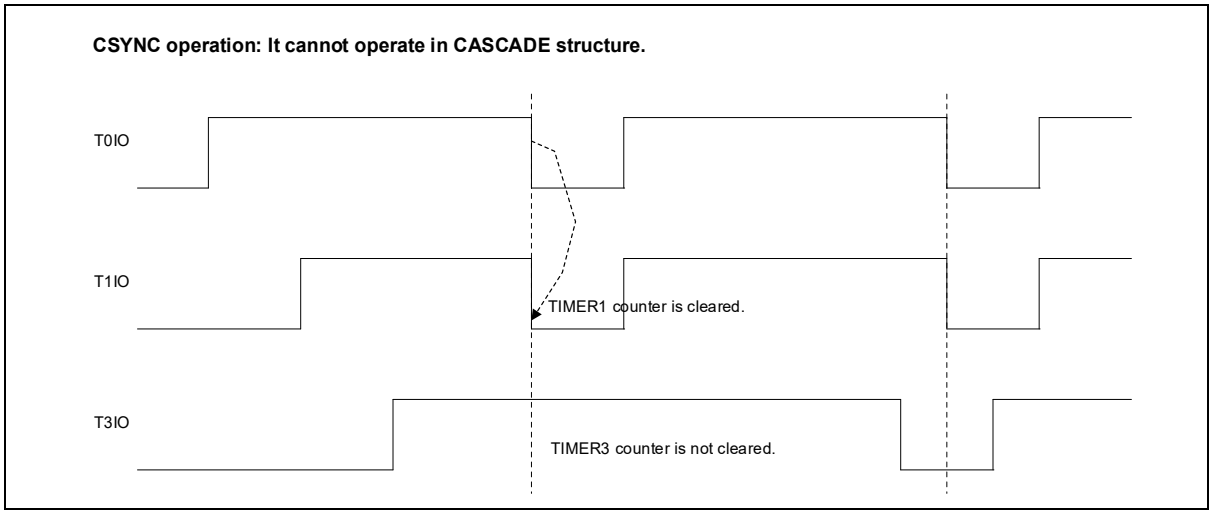


Figure 83. CSYNC Operation Example



10.3.10.3 Register Setting Sequence for Synchronous Function Operation

1. To enable the system access, write a reserved value to the corresponding bit of the System Access Enable Register (SCU_SYSTEN).
2. To enable the timer module, write '1' to the corresponding bit of the Peripheral Enable Register1 (SCU_PER1) and the Peripheral Clock Enable Register1 (SCU_PCER1).
3. Write zero to the TIMERn_CNT register to initialize the timer counter.
4. To set the timer mode, enter the corresponding mode set value in the MODE[1:0] bits in the TIMERn_CR1 register.
5. To set the timer clock source, enter the corresponding set value in the CKSEL[2:0] bits in the TIMERn_CR1 register.
6. To generate a more precision base clock, apply a prescaler to the timer clock source and enter the set value in the TIMERn_PRS register.
7. Write '1' to the IOSEL bit in the TIMERn_CR1 register to set the timer input/output pin (TnIO, n = 0 to 9).
8. Enter the set values in the TIMERn_GRA and TIMERn_GRB registers for PWM operation.
9. To use the synchronization function, write '1' to the SSYNC bit in the TIMERn_SYNC register or the CSYNC bit in the TIMERn_SYNC register (n = 0 to 9).
    A. When using a synchronized counter start mode, set the SSYNC bit in the TIMERn_SYNC register.
    B. When using a synchronized counter clear mode, set the CSYNC bit in the TIMERn_SYNC register.
10. To use a sync delay, enter the corresponding set value in the SYNCDLY[15:0] bits in the TIMERn_SYNC register.
11. Repeat steps 1 through 10 above to set all timers that use the synchronization function. And write '1' to the TLCR bit in the TIMERn_CR2 register to apply current timer setting.
12. For the timer operating as the master timer, enter the corresponding set value in the TnSYNCB bit in the TIMERn_SYNC register to select the slave timer (n = 0 to 9).
13. Write '1' to the TLCR bit in the TIMERn_CR2 register to apply current timer setting.
14. Write '1' to the TEN bit in the TIMERn_CR2 register to start the timer that operates as the Master.

10.3.11 Direction Bit Output

Since the timer outputs through a function pin of a GPIO port, users must set the GPIO port as a function pin. When the GPIO port is set to function pin mode, it operates as an input port or an output port depending on the timer operation mode.

Users can set the timer output value by configuring the IOSEL, OUTPOL, and STARTLVL bits in the TIMERn_CR1 register. Setting the IOSEL bit determines the timer input/output pin and setting the OUTPOL bit selects the polarity between the normal output and inversion output. Setting the STARTLVL bit allows users to select the initial output value and select whether to set the start output value to Low or High.

10.3.12 Debug Mode

When the microcontroller enters debug mode, the timer counter continues to work. Therefore, the outputs (TnIO, n = 0 to 9) are enabled, although the core halted, when the TEN bit in the TIMERn_CR2 register is set. When the TEN bit in the TIMERn_CR2 register is cleared, the outputs (TnIO, n = 0 to 9) are disabled.

10.3.13 Interrupts

The TIMERn can generate interrupts, as shown in Table 99.

Table 99. Interrupt Requests

Interrupt Event	Event Flag	Enable Control Bit	Interrupt Clear Method
Overflow interrupt	OVF in TIMERn_SR	OVIE in TIMERn_IER	Writing a ‘1’ to OVF in TIMERn_SR
GRB match interrupt	MFB in TIMERn_SR	MBIE in TIMERn_IER	Writing a ‘1’ to MFB in TIMERn_SR
GRA match interrupt	MFA in TIMERn_SR	MAIE in TIMERn_IER	Writing a ‘1’ to MFA in TIMERn_SR

NOTE:

1. When using overflow interrupt, set the TIMERn_GRA value and TIMERn_GRB value to zero. (Especially if the TIMERn_GRB value is not set to zero, OVF in TIMERn_SR will not be generated.)

10.4 Timer Registers

The base address of 16-bit timer is described in the following:

Table 100. Base Address of 16-bit Timer

Name	Base Address
TIMER0	0x4000_3000
TIMER1	0x4000_3040
TIMER2	0x4000_3080
TIMER3	0x4000_30C0
TIMER4	0x4000_3100
TIMER5	0x4000_3140
TIMER6	0x4000_3180
TIMER7	0x4000_31C0
TIMER8	0x4000_3200
TIMER9	0x4000_3240

Table 101. 16-bit Timer Register Map

Name	Offset	Type	Description	Reset Value	Ref.
TIMERn_CR1	0x0000	RW	Timer n Control Register 1	0x0000_0000	10.4.1
TIMERn_CR2	0x0004	RW	Timer n Control Register 2	0x0000_0000	10.4.2
TIMERn_PRS	0x0008	RW	Timer n Prescaler Register	0x0000_0000	10.4.3
TIMERn_GRA	0x000C	RW	Timer n General Data Register A	0x0000_0000	10.4.4
TIMERn_GRB	0x0010	RW	Timer n General Data Register B	0x0000_0000	10.4.5
TIMERn_CNT	0x0014	RW	Timer n Counter Register	0x0000_0000	10.4.6
TIMERn_SR	0x0018	RWC1	Timer n Status Register	0x0000_0000	10.4.7
TIMERn_IER	0x001C	RW	Timer n Interrupt Enable Register	0x0000_0000	10.4.8
TIMERn_TRGPNT	0x0020	RW	Timer n Trigger Point Register	0x0000_0000	10.4.9
TIMERn_SYNC	0x0024	RW	Timer n Sync Configuration Register	0x0000_0000	10.4.10

NOTE:

1. n = 0, 1, 2, 3, 4, 5, 6, 7, 8, and 9.

10.4.1 TIMERn_CR1: Timer/Counter n Control Register 1

TIMERn_CR1 is a 32-bit register. The timer module must be set appropriately before running the timer. After the use of the timer is specified, the timer can be set in TIMERn_CR1. After this register is set, users can enable or disable the timer by setting the TIMERn_CR2.

TIMER0_CR1=0x4000_3000, TIMER1_CR1=0x4000_3040, TIMER2_CR1=0x4000_3080, TIMER3_CR1=0x4000_30C0, TIMER4_CR1=0x4000_3100, TIMER5_CR1=0x4000_3140, TIMER6_CR1=0x4000_3180, TIMER7_CR1=0x4000_31C0, TIMER8_CR1=0x4000_3200, TIMER9_CR1=0x4000_3240

31 30 29 28 27 26 25 24	23 22 21 20 19 18 17 16	15 14	13	12	11	10 9	8	7 6 5 4	3 2	1 0
Reserved		TRGMOD[1:0]	UAO	OUTPOL	IOSEL	Reserved	ADCTRGEN	STARTLVL	CKSEL[2:0]	CLRMOD[1:0]	MODE[1:0]
-		00	0	0	0	-	0	0	000	00	00
-		RW	RW	RW	RW	-	RW	RW	RW	RW	RW

15 14	TRGMOD[1:0]	ADC trigger mode selection	
		0X	Selects TIMERn_GRA value trigger mode (normal mode).
		10	Selects TIMERn_TRGPNT value trigger mode.
		11	Triggers both TIMERn_GRA and TIMERn_TRGPNT values.
13	UAO	GRA/GRB update mode selection	
		0	The value written to TIMERn_GRA or TIMERn_GRB is applied after the current period.
		1	The value written to TIMERn_GRA or TIMERn_GRB is applied in the current period.
12	OUTPOL	Timer output polarity	
		0	General output
		1	General output inversion
11	IOSEL	TnIO pin configuration (n = 0 to 9)	
		0	Sets the pin as an input port
		1	Sets the pin as an output port
8	ADCTRGEN	Use the timer as an ADC trigger source	
		0	Does not use the timer as an ADC trigger source.
		1	Uses the timer as an ADC trigger source.
7	STARTLVL	Starting output value in periodic/PWM/one-shot modes	
		0	Sets the starting output value to “L”.
		1	Sets the starting output value to “H”.
6 4	CKSEL[2:0]	Counter clock source selection	
		000	PCLK / 2
		001	PCLK / 4
		010	PCLK / 16
		011	PCLK / 64
		10X	EXT_CLK (SCU_MCCR3) (1)
		11X	Input to pin TnIO (n = 0 to 9)
3 2	CLRMOD[1:0]	Clear mode selection in capture mode	
		00	Rising-edge clear mode
		01	Falling-edge clear mode
		10	Both-edge clear mode
		11	Disables clearing.

1 0	MODE[1:0]	Timer operation mode control	
		00	Normal periodic mode
		01	PWM mode
		10	One-shot mode
		11	Capture mode

NOTE:

1. If EXT_CLK is set in CKSEL[2:0] of TIMERn_CR1 register, the clock speed must satisfy PCLK ≥ TMCLK × 3.

10.4.2 TIMERn_CR2: Timer/Counter n Control Register 2

TIMERn_CR2 is a 32-bit register and is used to control the timer's operation. Before starting the timer, users must set the TCLR in TIMERn_CR2 to '1' to clear the timer count register.

TIMER0_CR2=0x4000_3004, TIMER1_CR2=0x4000_3044, TIMER2_CR2=0x4000_3084, TIMER3_CR2=0x4000_30C4, TIMER4_CR2=0x4000_3104, TIMER5_CR2=0x4000_3144, TIMER6_CR2=0x4000_3184, TIMER7_CR2=0x4000_31C4, TIMER8_CR2=0x4000_3204, TIMER9_CR2=0x4000_3244

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved																														TCLR	TEN
-																														0	0
-																														WO	RW

1	TCLR	Clearing of the timer count register(1)	
		0	No action
		1	Initializes the timer. Writing a '1' to TCLR clears the counter register (automatically cleared to '0' after operation).
0	TEN	Enable the timer	
		0	Stop the timer.
		1	Start the timer.

NOTE:

1. Users must set the TCLR bit to '1' before starting the timer to prevent a time error.

10.4.3 TIMERn_PRS: Timer/Counter n Prescaler Register

TIMERn_PRS is used to set the timer input frequency divider. It is 10 bits wide. users can generate precise and varied timer base clocks by applying the prescaler to the timer clock source that has been selected in the TIMERn_CR1 register.

TIMER0_PRS=0x4000_3008, TIMER1_PRS=0x4000_3048, TIMER2_PRS=0x4000_3088, TIMER3_PRS=0x4000_30C8, TIMER4_PRS=0x4000_3108, TIMER5_PRS=0x4000_3148, TIMER6_PRS=0x4000_3188, TIMER7_PRS=0x4000_31C8, TIMER8_PRS=0x4000_3208, TIMER9_PRS=0x4000_3248

31 30 29 28 27 26 25 24	23 22 21 20 19 18 17 16	15 14 13 12 11 10 9 8	7 6 5 4 3 2 1 0
Reserved			PRS[9:0]
-			000
-			RW

9 0	PRS[9:0]	Prescaler value for the counter clock TCLK = TMCLK / (PRS[9:0] + 1) (TMCLK = Timer input clock selected by the setting of CKSEL[2:0] in TIMERn_CR1)

10.4.4 TIMERn_GRA: Timer/Counter n General Data Register A

The TIMERn_GRA register is 16 bits wide.

TIMER0_GRA=0x4000_300C, TIMER1_GRA=0x4000_304C, TIMER2_GRA=0x4000_308C, TIMER3_GRA=0x4000_30CC, TIMER4_GRA=0x4000_310C, TIMER5_GRA=0x4000_314C, TIMER6_GRA=0x4000_318C, TIMER7_GRA=0x4000_31CC, TIMER8_GRA=0x4000_320C, TIMER9_GRA=0x4000_324C

31 30 29 28 27 26 25 24	23 22 21 20 19 18 17 16	15 14 13 12 11 10 9 8	7 6 5 4 3 2 1 0
Reserved		GRA[15:0]	
-		0x0000	
-		RW	

15 0	GRA[15:0]	Timer general register A This register is used for different purposes depending on the operating mode. **Period/PWM/One-shot Modes:** The target count value is stored in the register. If the counter value matches the TIMERn_GRA value, the counter is cleared to restart or stop running. When the timer is cleared, TnIO outputs the STARTLVL in TIMERn_CR1 value. If the counter value matches the register's value, the MFA in TIMERn_SR is triggered. When the GRA interrupt is triggered or clearing is demanded, the counter value is copied to internal data buffer 0. In PWM mode, the GRA[15:0] value represents the duty value. **Capture mode:** - In rising-edge clear mode, the register stores the counter value captured on the falling edge of the signal at port TnIO. - In falling-edge clear mode, the register stores the counter value captured on the rising edge of the signal at port TnIO.

10.4.5 TIMERn_GRB: Timer/Counter n General Data Register B

The TIMERn_GRB register is 16 bits wide.

TIMER0_GRB=0x4000_3010, TIMER1_GRB=0x4000_3050, TIMER2_GRB=0x4000_3090, TIMER3_GRB=0x4000_30D0, TIMER4_GRB=0x4000_3110, TIMER5_GRB=0x4000_3150, TIMER6_GRB=0x4000_3190, TIMER7_GRB=0x4000_31D0, TIMER8_GRB=0x4000_3210, TIMER9_GRB=0x4000_3250

31 30 29 28 27 26 25 24	23 22 21 20 19 18 17 16	15 14 13 12 11 10 9 8	7 6 5 4 3 2 1 0
Reserved		GRB[15:0]	
-		0x0000	
-		RW	

15 0	GRB[15:0]	Timer general register B This register is used for different purposes depending on the operating mode. **Periodic mode:** It should be set the same as the TIMERn_GRA **PWM/One-shot mode:** The target count value is stored in the register. If the counter value matches the register's value, the MFB in TIMERn_SR is triggered. In PWM mode, the TIMERn_GRB value represents the period value. **Capture mode:** In rising-edge clear mode, the register stores the counter value captured on the rising edge of the signal at port TnIO. (The opposite edge to that of TIMERn_GRA) In falling-edge clear mode, the register stores the counter value captured on the falling edge of the signal at port TnIO. (The opposite edge to that of TIMERn_GRA)

10.4.6 TIMERn_CNT: Timer/Counter n Count Register

TIMERn_CNT register is 16 bits wide. The count is incremented based on the specified input clock. This register can be both read and written to.

TIMER0_CNT=0x4000_3014, TIMER1_CNT=0x4000_3054, TIMER2_CNT=0x4000_3094, TIMER3_CNT=0x4000_30D4, TIMER4_CNT=0x4000_3114, TIMER5_CNT=0x4000_3154, TIMER6_CNT=0x4000_3194, TIMER7_CNT=0x4000_31D4, TIMER8_CNT=0x4000_3214, TIMER9_CNT=0x4000_3254

31 30 29 28 27 26 25 24	23 22 21 20 19 18 17 16	15 14 13 12 11 10 9 8	7 6 5 4 3 2 1 0
Reserved		CNT[15:0]	
-		0x0000	
-		RW	

15 0	CNT[15:0]	Timer count value	
		R	Reads the current timer count.
		W	Sets the count value.

10.4.7 TIMERn_SR: Timer/Counter n Status Register

The TIMERn_SR register shows a timer module's status. This register is 8 bits wide.

TIMER0_SR=0x4000_3018, TIMER1_SR=0x4000_3058, TIMER2_SR=0x4000_3098, TIMER3_SR=0x4000_30D8, TIMER4_SR=0x4000_3118, TIMER5_SR=0x4000_3158, TIMER6_SR=0x4000_3198, TIMER7_SR=0x4000_31D8, TIMER8_SR=0x4000_3218, TIMER9_SR=0x4000_3258

31 30 29 28 27 26 25 24	23 22 21 20 19 18 17 16	15 14 13 12 11 10 9 8	7 6 5 4 3	2	1	0
Reserved				MFA	MFB	OVF
-				0	0	0
-				RWC1	RWC1	RWC1

2	MFA	TIMERn_GRA match flag	
		0	No TIMERn_GRA match has been detected.
		1	A TIMERn_GRA match has been flagged. (Writing a '1' to the bit clears the flag)
1	MFB	TIMERn_GRB match flag	
		0	No TIMERn_GRB match has been detected.
		1	A TIMERn_GRB match has been flagged. (Writing a '1' to the bit clears the flag)
0	OVF	Counter overflow flag	
		0	No overflow event has occurred.
		1	A counter overflow event has been flagged. (Writing a '1' to the bit clears the flag)

10.4.8 TIMERn_IER: Timer/Counter n Interrupt Enable Register

TIMERn_IER register is 8 bits wide. Each status flag in the timer module can generate an interrupt. To trigger an interrupt, users must write '1' to the corresponding bit in this register.

TIMER0_IER=0x4000_301C, TIMER1_IER=0x4000_305C, TIMER2_IER=0x4000_309C, TIMER3_IER=0x4000_30DC, TIMER4_IER=0x4000_311C, TIMER5_IER=0x4000_315C, TIMER6_IER=0x4000_319C, TIMER7_IER=0x4000_31DC, TIMER8_IER=0x4000_321C, TIMER9_IER=0x4000_325C

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved																													MAIE	MBIE	OVIE
-																													0	0	0
-																													RW	RW	RW

2	MAIE	Enable the TIMERn_GRA match interrupt	
		0	Disables the TIMERn_GRA match interrupt.
		1	Enables the TIMERn_GRA match interrupt.
1	MBIE	Enable the TIMERn_GRB match interrupt	
		0	Disables the TIMERn_GRB match interrupt.
		1	Enables the TIMERn_GRB match interrupt.
0	OVIE	Enable the counter overflow interrupt	
		0	Disables the counter overflow interrupt.
		1	Enables the counter overflow interrupt.

10.4.9 TIMERn_TRGPNT: Timer/Counter n Trigger Point Register

The TIMERn_TRGPNT register is used to trigger the ADC at the desired timer counter point. This register is 16 bits wide.

TIMER0_TRGPNT=0x4000_3020, TIMER1_TRGPNT=0x4000_3060, TIMER2_TRGPNT=0x4000_30A0, TIMER3_TRGPNT=0x4000_30E0, TIMER4_TRGPNT=0x4000_3120, TIMER5_TRGPNT=0x4000_3160, TIMER6_TRGPNT=0x4000_31A0, TIMER7_TRGPNT=0x4000_31E0, TIMER8_TRGPNT=0x4000_3220, TIMER9_TRGPNT=0x4000_3260

31 30 29 28 27 26 25 24	23 22 21 20 19 18 17 16	15 14 13 12 11 10 9 8	7 6 5 4 3 2 1 0
Reserved		TRGPNT[15:0]	
-		0x0000	
-		RW	

15 0	TRGPNT[15:0]	Timer trigger point value
		Read: Reads the current trigger point value.
		Write: Sets the trigger point value.

NOTE:

1. The ADC value can be read when the timer count and TPGPNT[15:0] value are matched.

10.4.10 TIMERn_SYNC: Timer/Counter n Sync Configuration Register

TIMERn_SYNC is used to synchronize Group 1 (TIMER0 to TIMER4) and Group 2 (TIMER5 to TIMER9) to start or clear the respective timers. For example, it needs to set '15' to SYNCDLY[15:0] in TIMER0_SYNC register and set '1' to T1SYNCB in TIMER0_SYNC register if user wants to start TIMER1 after 15 counts after starting TIMER0.

It has no effect to set '1' to T0SYNCB in TIMER0_SYNC register. And Syncing TIMER0 from TIMER1 after TIMER0 to TIMER1 has started syncing has no effect.

TIMER0_SYNC=0x4000_3024, TIMER1_SYNC=0x4000_3064, TIMER2_SYNC=0x4000_30A4 TIMER3_SYNC=0x4000_30E4, TIMER4_SYNC=0x4000_3124, TIMER5_SYNC=0x4000_3164, TIMER6_SYNC=0x4000_31A4, TIMER7_SYNC=0x4000_31E4, TIMER8_SYNC=0x4000_3224 TIMER9_SYNC=0x4000_3264

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved		T9SYNCB	T8SYNCB	T7SYNCB	T6SYNCB	T5SYNCB	T4SYNCB	T3SYNCB	T2SYNCB	T1SYNCB	T0SYNCB	Reserved		SSYNC	CSYNC	SYNCDLY[15:0]															
-		0	0	0	0	0	0	0	0	0	0	-		0	0	0x0000															
-		RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	-		RW	RW	RW															

20+n	TnSYNCB (n=0 to 9)	Synchronization settings to the Timer n: As a master, the timer must select a slave timer that is enabled to synchronize. The slave timer must not select a master timer. (GROUP1: TIMER0 to TIMER4, GROUP2: TIMER5 to TIMER9)	
		0	Disables synchronization.
		1	Enables synchronization.
17	SSYNC[(1)]	Synchronizes other synchronization timer to start counter (The slave timer needs this configuration.)	
		0	Disables synchronization.
		1	Counter start mode that is synchronized.
16	CSYNC	Synchronizes other synchronization timer to clear counter (The slave timer needs this configuration.)	
		0	Disables synchronization.
		1	Counter clear mode that is synchronized.
15 0	SYNCDLY[15:0][(2)]	SYNC delay start count value setting bit(based on the master counter)	
		The SYNCDLY[15:0] is used to set the interval between the starting points of two synchronized timers. If this field is set to zero, the timers start simultaneously. (In this case, the two timers must not be running. If a value other than zero is written while the timers are running, the interval is applied from the next cycle onward.)	

NOTES:

1. If synchronized slave timer is not enabled and SSYNC = 1, the synchronized slave timer starts counting when the master counter's count value reaches the value written to SYNCDLY[15:0].
2. If you enable synchronization with TIMER1 in TIMER0_SYNC and synchronization with TIMER2 in TIMER1_SYNC, TIMER0, TIMER1 and TIMER2 will start successively with the delay time set at SYNCDLY[15:0] between them.

10.4.11 Timer Register Map Summary

Table 102. TIMER Register Map Summary

Offset	Register name	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0x00	TIMERn_CR1	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	TRGMOD[1:0]		UAO	OUTPOL	IOSEL			ADCTRGEN	STARTLVL	CKSEL[2:0]			CLRMOD[1:0]		MODE[1:0]	
	Reset value																	0	0	0	0	0			0	0	0	0	0	0	0	0	0
0x04	TIMERn_CR2	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	TCLR	TEN
	Reset value																															0	0
0x08	TIMERn_PRS	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	PRS[9:0]									
	Reset value																							0	0	0	0	0	0	0	0	0	0
0x0C	TIMERn_GRA	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	GRA[15:0]															
	Reset value																	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
0x10	TIMERn_GRB	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	GRB[15:0]															
	Reset value																	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
0x14	TIMERn_CNT	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	CNT[15:0]															
	Reset value																	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
0x18	TIMERn_SR	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	MFA	MFB	OVF
	Reset value																														0	0	0
0x1C	TIMERn_IER	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	MAIE	MBIE	OVIE
	Reset value																														0	0	0
0x20	TIMERn_TRGPNT	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	TRGPNT[15:0]															
	Reset value																	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
0x24	TIMERn_SYNC	Res.	Res.	T9SYNCB	T8SYNCB	T7SYNCB	T6SYNCB	T5SYNCB	T4SYNCB	T3SYNCB	T2SYNCB	T1SYNCB	T0SYNCB	Res.	Res.	SSYNC	CSYNC	SYNCDLY[15:0]															
	Reset value			0	0	0	0	0	0	0	0	0	0			0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

11. Free-Run Timer (FRT)

11.1 FRT Introduction

The A34G43x series has two built-in Free-Run Timer (FRT) that is 32-bit up count timer. The FRT can run with an overflow interrupt or a match interrupt according to their register setting and can remain active in DEEP-SLEEP STOP 1 mode.

11.2 FRT Main Features

The FRT of the A34G43x has the features shown below:

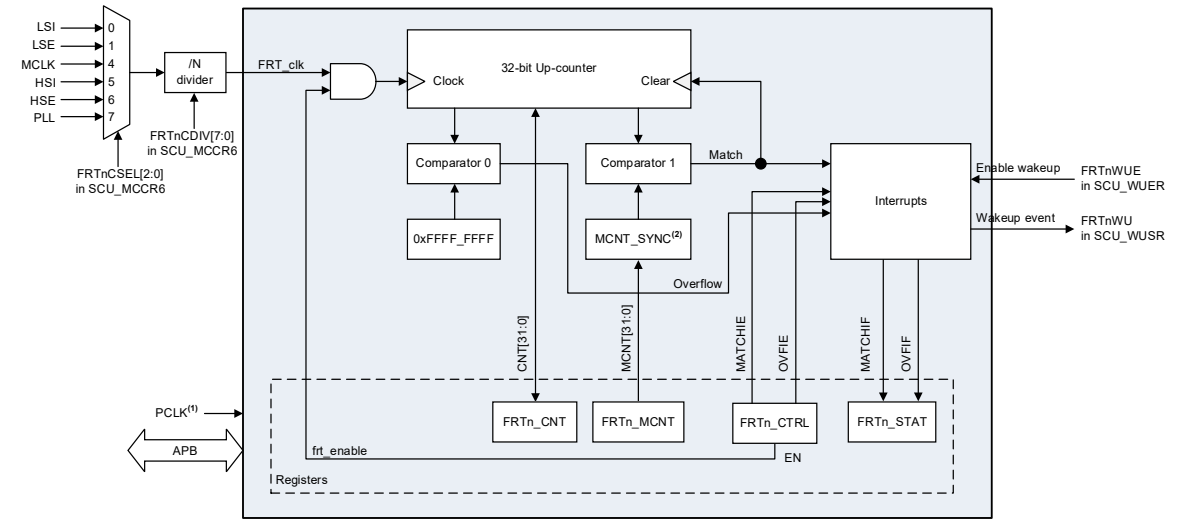
- Two FRT units
- 32-bit up count timer‘s two operations
  - Free-Run Timer mode operation
  - Periodic timer mode operation based on the setup period
- FRT overflow interrupt and match interrupt functions
- Selectable FRT input clocks
  - Input clock source can be selected using the SCU_MCCR6 register: LSI, LSE, MCLK, HSI, HSE, or PLL.

11.3 FRT Functional Description

11.3.1 FRT Block Diagram

Figure 84 shows a block diagram of the FRT.

Figure 84. FRT Block Diagram



NOTES:

1. The FRT register provides the PCLK clock on the APB and uses an external clock set in the SCU_MCCR6 register.
2. MCNT_SYNC block is built-in to synchronize these two clocks.

11.3.2 FRT Clock Selection

The FRT can be enabled by configuring the SCU_PER1 and SCU_PCER1 registers that are peripheral enable register and peripheral clock enable register, respectively. Users must set the FRTn (n = 0 to 1) in the SCU_PER1 and SCU_PCER1 register to '1'. Until the clock is provided to the FRT module, it is not reset nor operates normally.

An external clock source can be set as a clock source for the FRT. The external clock source can be selected as the FRT's clock source when the FRTnCDIV[7:0] bits in the SCU_MCCR6 register is set to a certain value larger than zero and the FRTnCSEL[2:0] bits in the SCU_MCCR6 register is set to select the external clock source.

There is an important thing to remember. When the FRT main clock is set to the MCLK, the A34G43x cannot wake up from DEEP-SLEEP (STOP 1) mode. To enable the system to wake from Deep Sleep mode, the FRT main clock must be set to a clock other than the MCLK.

11.3.3 FRT Operation Mode

The MODE bit in the FRTn_CTRL register determines whether the FRT operates in Periodic Timer Mode or Free-Run Timer Mode (n = 0 to 1).

When the FRT runs in Periodic Timer Mode, the MATCHIE bit in the FRTn_CTRL register determines whether to enable or disable the match interrupt. This bit does not affect the overflow interrupt.

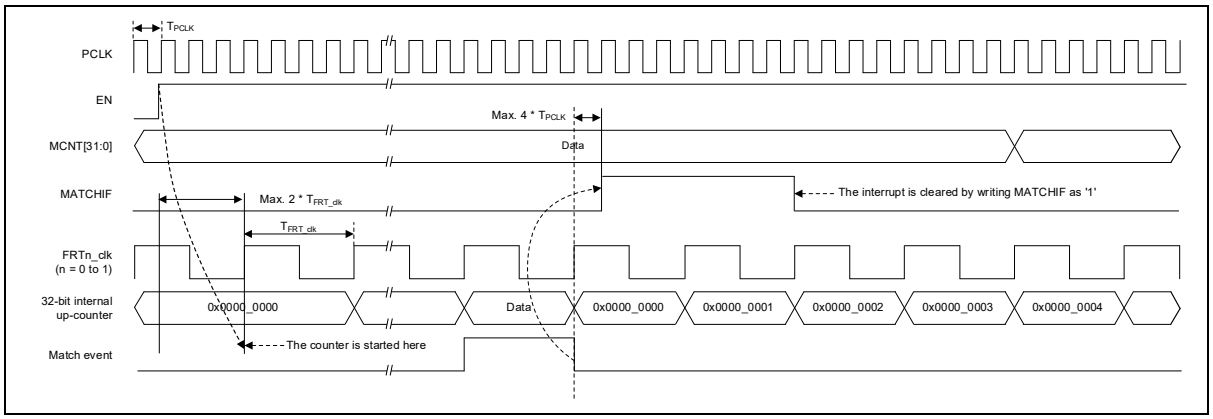
When the FRT runs in Free-Run Timer Mode, the OVFIE bit in the FRTn_CTRL register determines whether to enable or disable the overflow interrupt. This bit does not affect the match interrupt.

11.3.3.1 Match Interrupt Operation

Figure 85 shows a timing diagram of the match interrupt operation. To enable the match interrupt, the MATCHIE bit in the FRTn_CTRL register must be set to '1'.

When the EN bit in the FRTn_CTRL register is set to '1', the FRT counter starts counting. Once the counter value reaches the FRTn_MCNT value, the FRT counter is clear to zero and the interrupt and wake-up signal are triggered. An interrupt signal can be delayed by up to approximately four system clock (PCLK) pulses.

Figure 85. Match Interrupt Operation Timing Diagram



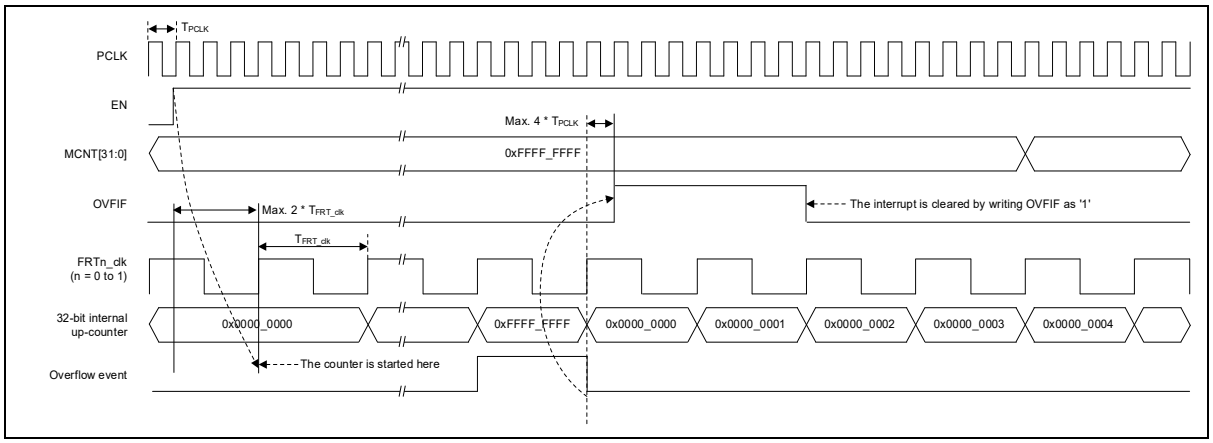
The following formula calculates the FRT's count period in Periodic Timer Mode:

- Period = FRTn_clk period × (MCNT + 1)
- Match interrupt time = FRTn_clk period × (MCNT + 1)

11.3.3.2 Overflow Interrupt Operation

Figure 86 shows a timing diagram of the overflow interrupt operation. The overflow interrupt operates almost in the same way as the match interrupt. It is triggered when the FRT counter value matches 0xFFFF_FFFF.

Figure 86. Overflow Interrupt Operation Timing Diagram



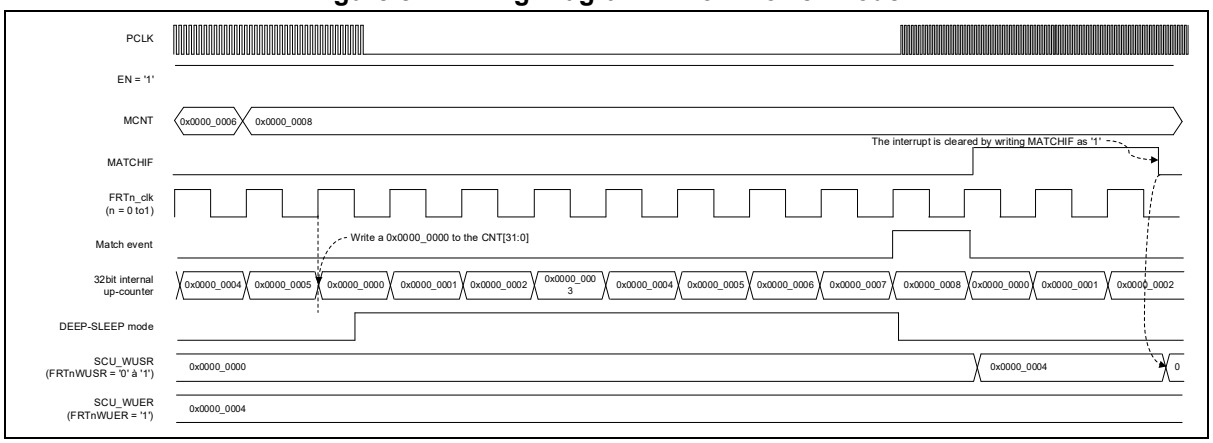
11.3.3.3 Register Setting Procedure by FRT Operation Mode

1. First, write the predefined value to the SCU_SYSTEN register. This is the system access enable register and writing predefined value to this register can enable users to access the system.
2. To enable the FRT module, set each corresponding bit of the SCU_PER1 and SCU_PCER1 registers to '1'. The SCU_PER1 and SCU_PCER1 registers are the peripheral enable register and peripheral clock enable register, respectively.
3. To select an external clock for the FRT module, set the FRTnCDIV[7:0] bits in the SCU_MCCR6 register to a value larger than '0'. Then set the FRTnCSEL[2:0] bits in the SCU_MCCR6 register to the value of the desired clock.
4. Set the MODE in FRTn_CTRL register to select FRT operation mode:
   A. '0': Free-Run Timer Mode
   B. '1': Match Interrupt Mode
5. To enable interrupts for each mode, set the corresponding bit of the FRTn_CTRL register to '1' depending on the mode that was selected in the previous step:
   A. OVFIE bit: FRT counter overflow interrupts are enabled in Free-Run Timer Mode.
   B. MATCHIE bit: FRT counter match interrupts are enabled in Match Interrupt Mode.
6. In Match Interrupt Mode, use the FRTn_MCNT register to specify the match counter value.

   Match interrupt time = FRTn_clk period × (MCNT + 1)
7. Set the NVIC registers that are assigned for the FRT interrupts.
8. To initialize the FRT counter, write '0' to the FRTn_CNT register.
9. To enable the FRT counter, set the EN in FRTn_CTRL register to '1'.
10. Read the FRTn_STAT register to check if the FRT interrupt event occurs. If the corresponding bit in the register is set, clear it by writing '1' to the bit field.

11.3.4 FRT Low-Power Modes

Figure 87. Timing Diagram in Low-Power Mode



11.3.4.1 Description of the Steps of Low-Power Mode

1. To enable the signal to be used as a wake-up source, set FRTnWUE bit in the SCU_WUER register to ‘1’.
2. Set MCNT[31:0] bits in the FRTn_MCNT register, which triggers the match interrupt. (In the diagram above, it is set to 8 as an example.)
3. Clear CNT[31:0] bits in the FRTn_CNT register to zero, which initialize 32-bit internal up-counter.
4. Enter low-power mode (DEEP-SLEEP mode)
5. When 32-bit internal up-counter reaches the MCNT[31:0] bits in the FRTn_MCNT register, the Match Event is generated. Then MACHIF flag in FRTn_STAT register and FRTnWU flag in SCU_WUSR register are triggered after delay of up to four system clocks.
6. And writing as ‘1’ to MATHIF bit in the FRTn_STAT register to clear triggered flags. At this time the FRTnWU flag in the SCU_WUSR register is also cleared.

11.4 FRT Registers

The base address of the FRT and register map are described in the following:

Table 103. Base Address of FRT

Name	Base Address
FRT0	0x4000_0600
FRT1	0x4000_0700

Table 104. FRT Register Map

Name	Offset	Type	Description	Reset Value	Reference
FRTn_CTRL	0x0000	RW	FRT n Control Register	0x0000_0000	11.4.1
FRTn_MCNT	0x0004	RW	FRT n Match Counter Register	0xFFFF_FFFF	11.4.2
FRTn_CNT	0x0008	RW	FRT n Counter Register	0x0000_0000	11.4.3
FRTn_STAT	0x000C	RWC1	FRT n Status Register	0x0000_0000	11.4.4

NOTE:

1. n = 0 and 1.

11.4.1 FRTn_CTRL: FRT n Control Register

The FRT_CTRL register has control bits to the FRT operation. This register is 32 bits wide.

FRT0_CTRL=0x4000_0600, FRT1_CTRL=0x4000_0700

31 30 29 28 27 26 25 24	23 22 21 20 19 18 17 16	15 14 13 12 11 10	9	8	7 6 5 4 3 2	1	0
Reserved			OVFIE	MATCHIE	Reserved	MODE	EN
-			0	0	-	0	0
-			RW	RW	-	RW	RW

9	OVFIE	Enable the FRT counter overflow interrupt in Free-run timer mode (MODE = '0')	
		0	Disable the overflow interrupt
		1	Enable the overflow interrupt
8	MATCHIE	Enable the FRT counter match interrupt in Periodic timer mode (MODE = '1')	
		0	Disable the match interrupt
		1	Enable the match interrupt
1	MODE	FRT mode selection	
		0	Free-Run Timer Mode
		1	Periodic Timer Mode
0	EN	Enable the FRT	
		0	Disable
		1	Enable

11.4.2 FRTn_MCNT: FRT n Match Counter Register

The FRTn_MCNT is a 32-bit register. It is used to specify the period value when the FRT operates in Periodic timer mode. In Periodic Timer Mode, the value of the FRTn_CNT register counts up until it reaches the FRTn_MCNT value, which triggers the match interrupt if it has been set enabled.

FRT0_MCNT=0x4000_0604, FRT1_MCNT=0x4000_0704

31 30 29 28 27 26 25 24	23 22 21 20 19 18 17 16	15 14 13 12 11 10 9 8	7 6 5 4 3 2 1 0
MCNT[31:0]			
0xFFFF_FFFF			
RW			

31 0	MCNT[31:0]	FRT's match counter value	
		The match interrupt is triggered when the current counter value reaches the set match counter value. Match interrupt time = FRTn_clk period × (MCNT + 1)	

11.4.3 FRTn_CNT: FRT n Counter Register

The FRTn_CNT is a 32-bit register that shows the FRT's current counting value. The register can be both read and written as an up-count timer whose count value is incremented.

If the EN in FRTn_CTRL register is set to '1', this register can be read and written.

FRT0_CNT=0x4000_0608, FRT1_CNT=0x4000_0708

31 30 29 28 27 26 25 24	23 22 21 20 19 18 17 16	15 14 13 12 11 10 9 8	7 6 5 4 3 2 1 0
CNT[31:0]			
0x0000_0000			
RW			

31 0	CNT[31:0]	FRT count data Represents the current count value. (Only the initializing value 0x0 can be written to the bit field.)

11.4.4 FRTn_STAT: FRT n Status Register

The FRTn_STAT is a 32-bit register that shows the FRTn status. It can be read the status of each of FRTn events and can be clear the status bit by writing '1' to the corresponding bit when the event occurs.

FRT0_STAT=0x4000_060C, FRT1_STAT=0x4000_070C

31 30 29 28 27 26 25 24	23 22 21 20 19 18 17 16	15 14 13 12 11 10	9	8	7 6 5 4 3 2 1 0
Reserved			OVFIE	MATCHIF	Reserved
-			0	0	-
-			RWC1	RWC1	-

9	OVFIF	FRT counter overflow interrupt flag	
		0	The overflow interrupt event has not occurred.
		1	The overflow interrupt event has occurred. (Writing a '1' to the bit clears this flag)
8	MATCHIF	FRT counter match interrupt flag	
		0	The match interrupt event has not occurred.
		1	The match interrupt event occurred. (Writing a '1' to the bit clears this flag)

11.4.5 FRT Register Map Summary

Table 105. FRT Register Map Summary

Offset	Register name	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0x00	FRTn_CTRL	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	OVFIE	MATCHIE	Res	Res	Res	Res	Res	Res	MODE	EN
	Reset value																							0	0							0	0
0x04	FRTn_MCNT	MCNT[31:0]																															
	Reset value	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1
0x08	FRTn_CNT	CNT[31:0]																															
	Reset value	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
0x0C	FRTn_STAT	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	OVFIF	MATCHIF	Res	Res	Res	Res	Res	Res	Res	Res
	Reset value																							0	0								

12. Universal Asynchronous Receiver/Transmitter (UART)

12.1 UART Introduction

The A34G43x series has thirteen channels of UART module. This built-in UART module transmits and receives data according to user-specified settings and reads the current UART status.

The UART status information includes type and conditions of the current UART transmission/reception process and can be used to check for errors (parity, overrun, framing, or break interrupts) that may occur during data reception.

Each UART channel has a programmable baud-rate generator that generates an internal clock for the corresponding UART unit by dividing the prescaled clock using a baud-rate divisor (ranging from 1 to 65,535) and then dividing the result by 16.

Users can program interrupts that can control UART communication.

- Using Direct Memory Access (DMA), users can perform high-speed data communication.
- Using receive FIFO, users can perform high-speed receive data.

12.2 UART Main Features

The UART of the A34G43x features the following:

- Six 16450 full-duplex asynchronous serial communication ports
- Configurable standard asynchronous communication bits (start, stop, and parity) are supported.
- User-programmable serial communication is available.
  - 5, 6, 7, or 8 data bits
  - Even, odd, or no parity generation and checking
  - 1-, 1.5-, or 2-stop bit generation and checking
- A 16-bit baud-rate generator and an 8-bit fractional compensator are included.
- Single or multi-sampling for start and data bits
- Delay between data frames is possible.
- Separate signal polarity control for transmission and reception
- Transfer status indicated by the interrupt ID and line status registers
  - Stop bit error detection
  - Display of information about the current status
  - Line break generation and checking
  - Receive error diagnosis
- A priority-based interrupt system
- Continuous communication is possible using DMA.
- Received/transmitted bytes are buffered in reserved SRAM using centralized DMA.
- Received bytes are buffered in FIFO. Received FIFO depth is 8.

12.2.1 UART Implementation

Table 106 describes the UART implementation on A34G43x.

Table 106. UART Features

UART Modes/Features	UART
Burst transfer using DMA	O
UART data length	5, 6, 7 and 8 bits
Received FIFO buffer	Depth 8

12.3 UART Functional Description

The UART module is compatible with 16450 UART. It provides DMA channels and fractional compensation logic to obtain baud-rates. Because it does not include a transmit FIFO block, data transfers are performed either interactively or with DMA support.

The DMA operates as follows:

A UART can be linked with two DMA channels, of which one is responsible for Tx transfers and the other is responsible for Rx transfers. Each channel has a 32-bit memory address register and a 32-bit peripheral address register. Users must set these two registers before enabling the DMA.

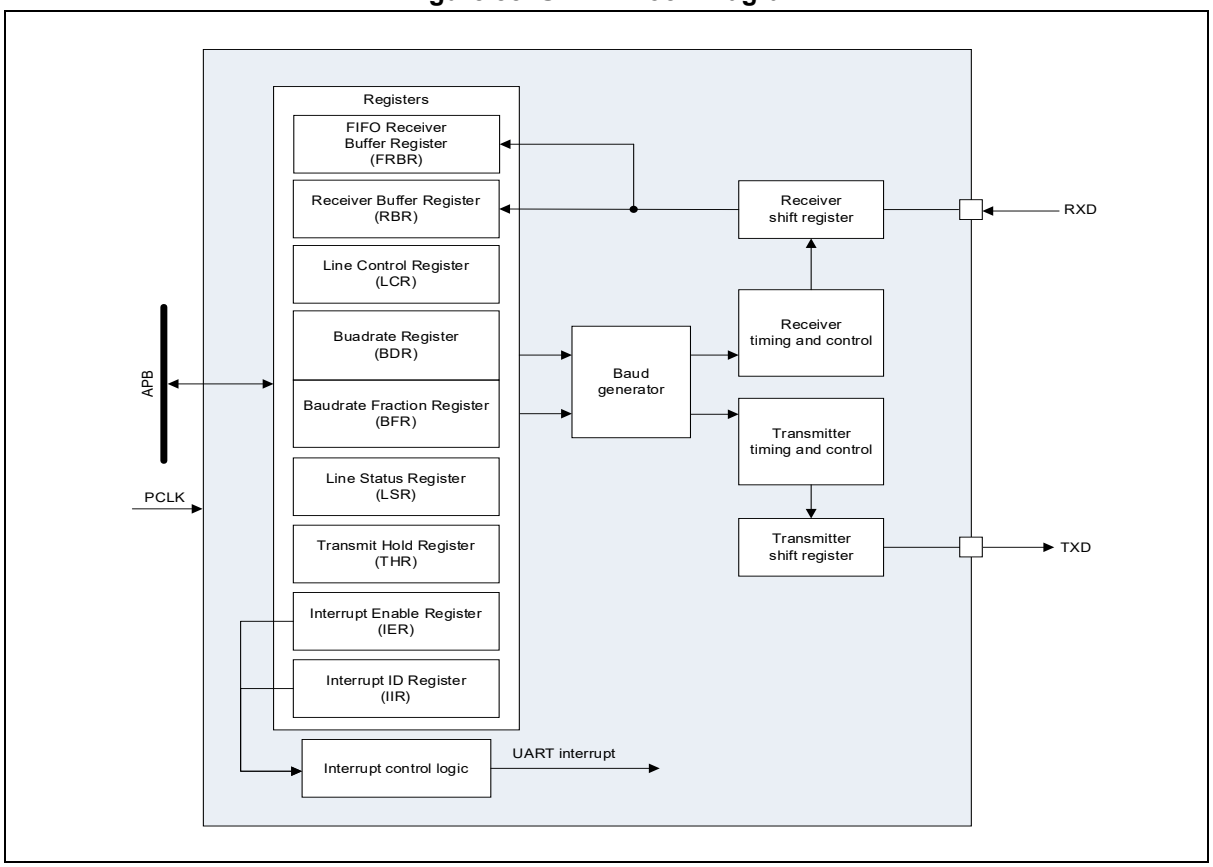
The memory address for the Rx transfers is the target address, whereas the address for Tx transfers is the source address.

The TRANSCNT[11:0] bit in the DMAn_CR register records the number of transfers. The counter decreases by one every time a single-ended transfer is completed. When the counter reaches zero, a DMA complete flag is sent to the UART control block. Then, the corresponding interrupt is flagged if the interrupt has been set enabled.

12.3.1 UART Block Diagram

Figure 88 shows a block diagram of the UART.

Figure 88. UART Block Diagram



12.3.2 UART Pins and Signals

Table 107 describes the external pins assigned for UART.

Table 107. Pin Assignment of UART: External Pins

Pin Name	Type	Description	Supported Packages		
			A34G43xYL (120-LQFP)	A34G43xVL (100-LQFP)	A34G43xRL (64-LQFP)
TXD0	O	UART channel 0 transmit output	O	O	O
RXD0	I	UART channel 0 receive input	O	O	O
TXD1	O	UART channel 1 transmit output	O	O	O
RXD1	I	UART channel 1 receive input	O	O	O
TXD2	O	UART channel 2 transmit output	O	O	O
RXD2	I	UART channel 2 receive input	O	O	O
TXD3	O	UART channel 3 transmit output	O	O	O
RXD3	I	UART channel 3 receive input	O	O	O
TXD4	O	UART channel 4 transmit output	O	O	N/A
RXD4	I	UART channel 4 receive input	O	O	N/A
TXD5	O	UART channel 5 transmit output	O	O	N/A
RXD5	I	UART channel 5 receive input	O	O	N/A
TXD6	O	UART channel 6 transmit output	O	O	N/A
RXD6	I	UART channel 6 receive input	O	O	N/A
TXD7	O	UART channel 7 transmit output	O	O	O
RXD7	I	UART channel 7 receive input	O	O	O
TXD8	O	UART channel 8 transmit output	O	O	N/A
RXD8	I	UART channel 8 receive input	O	O	N/A
TXD9	O	UART channel 9 transmit output	O	N/A	N/A
RXD9	I	UART channel 9 receive input	O	N/A	N/A
TXD10	O	UART channel 10 transmit output	O	N/A	N/A
RXD10	I	UART channel 10 receive input	O	N/A	N/A
TXD11	O	UART channel 11 transmit output	O	N/A	N/A
RXD11	I	UART channel 11 receive input	O	N/A	N/A
TXD12	O	UART channel 12 transmit output	O	O	N/A
RXD12	I	UART channel 12 receive input	O	O	N/A

12.3.2.1 UART Bidirectional Communications

UART bidirectional communication requires at least two pins such as Receive Data In (RXD) and Transmit Data Out (TXD).

The RXD is a serial data input. Oversampling techniques are used for data recovery. They discriminate between valid incoming data and noise.

The TXD is a serial data output. When the UART module is enabled and no data needs to be transmitted, the TXD pin is high state.

12.3.3 UART Character Description

The data length can be set to 5, 6, 7 or 8 bits, by configuring the DLEN[1:0] in UARTn_LCR register as described below:

- For 5-bit character length: DLEN[1:0] = '00' (refer to Figure 93)
- For 6-bit character length: DLEN[1:0] = '01' (refer to Figure 92)
- For 7-bit character length: DLEN[1:0] = '10' (refer to Figure 91)
- For 8-bit character length: DLEN[1:0] = '11' (refer to Figure 90)

By default, the signal (TXD or RXD) is in low state during the start bit. It is in high-state during the stop bit.

Figure 89 shows that the signal value (TXD and RXD) can be inverted, separately for each signal, through UARTn_DCR's data inversion control.

Figure 89. Data Inversion Control Diagram

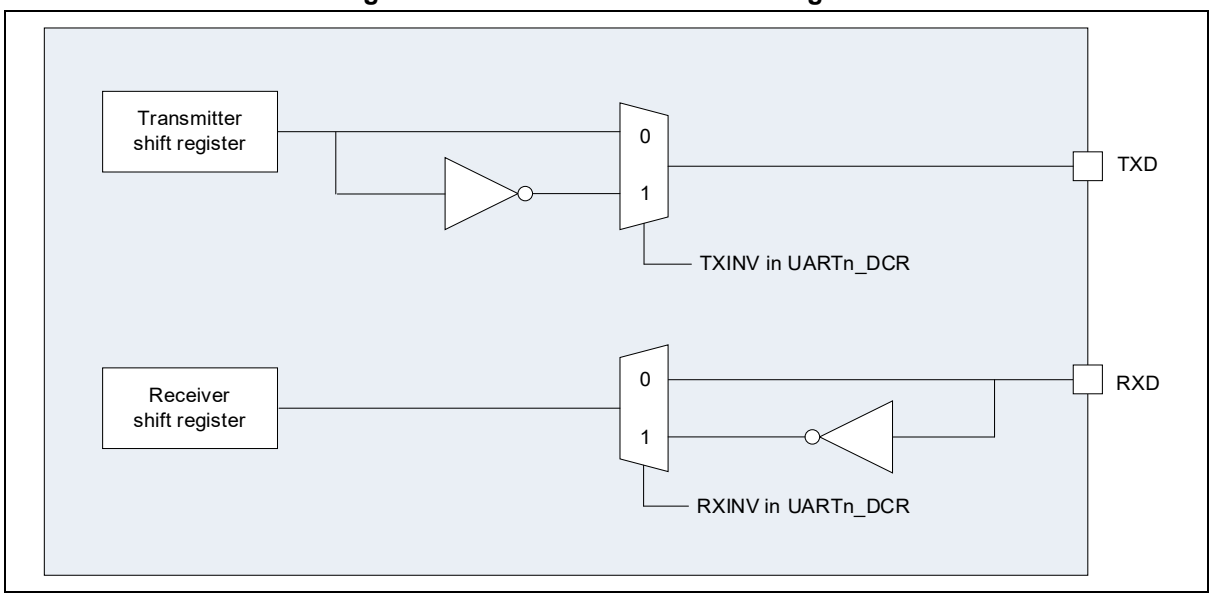


Figure 90 shows a timing diagram of data length programming. A Break condition is interpreted on receiving ‘0’s for longer than the time taken to receive the entire data word (i.e., the sum of the start, data, parity, and stop bits). The transmission and reception clocks are generated when the enable bit is set for the UART peripheral and clock, respectively.

Figure 90. Timing Diagram of 8-bit Data Length

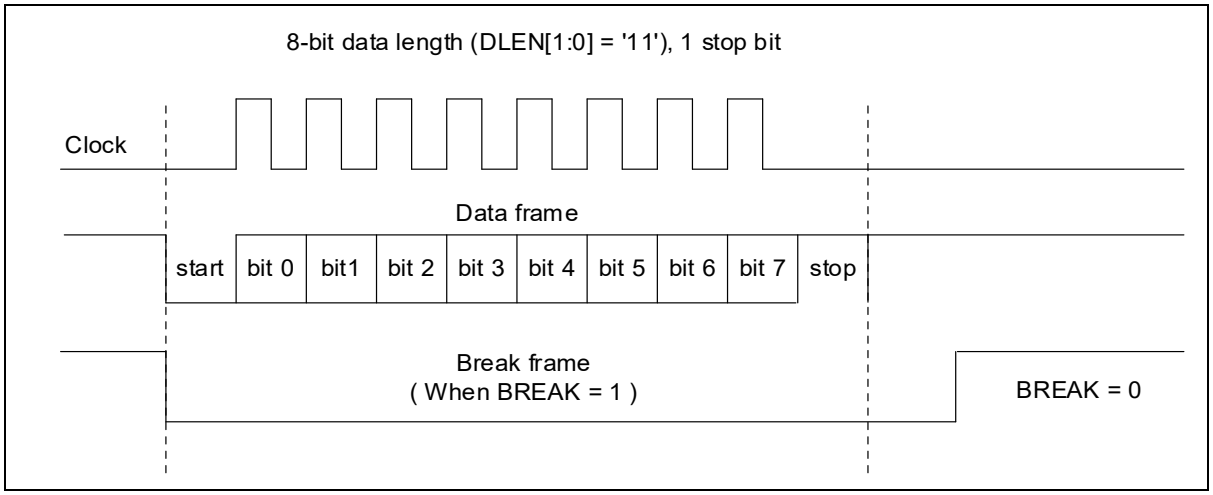


Figure 91. Timing Diagram of 7-bit Data Length

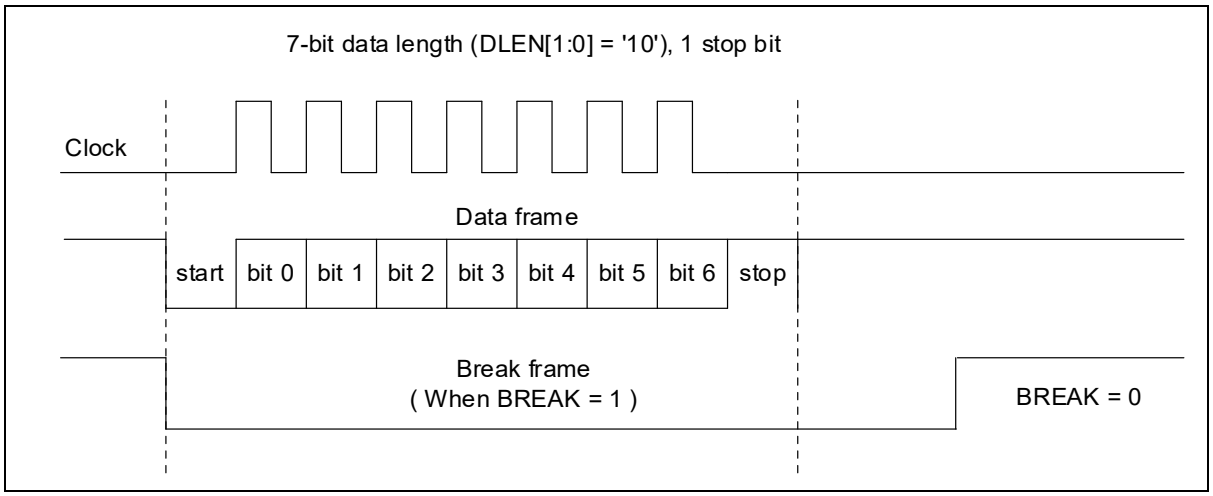


Figure 92. Timing Diagram of 6-bit Data Length

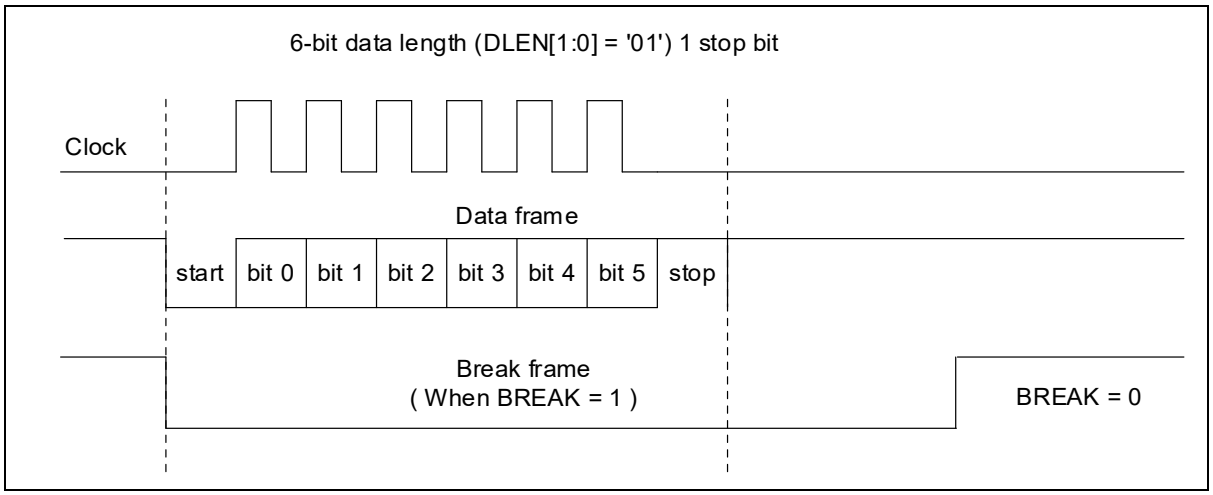
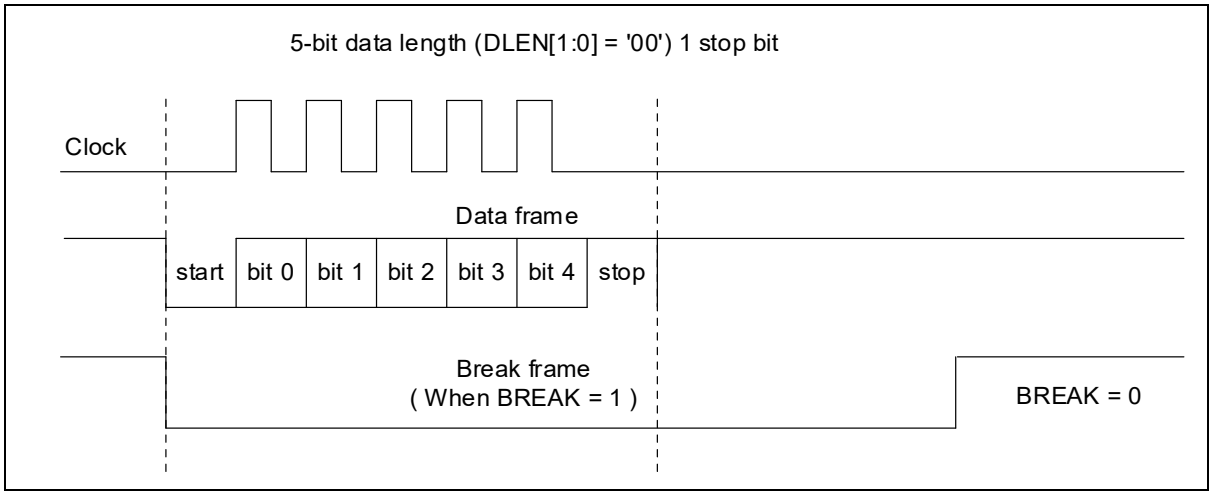


Figure 93. Timing Diagram of 5-bit Data Length



12.3.4 UART Transmitter

12.3.4.1 Data Format

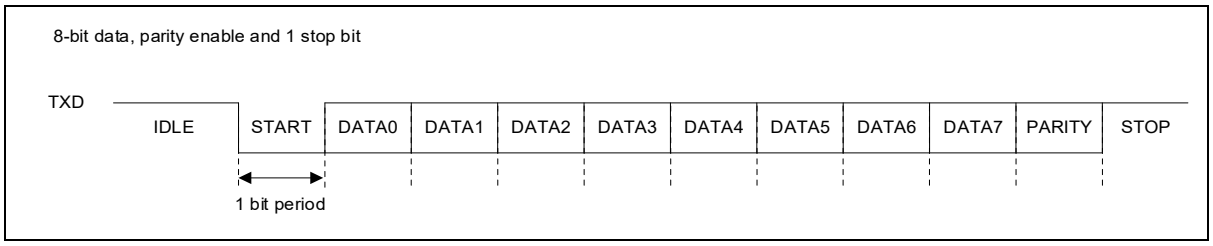
The UART transmitter is in charge of transmitting data. For each data word, the start bit, data bits, optional bit, and stop bit are shifted serially in and out of the corresponding register, one bit at a time at the Least Significant Bit (LSB).

The number of data bits can be determined by setting the DLEN[1:0] bits in the UARTn_LCR register. The UART transmitter sends data words of either 5, 6, 7, or 8 bits, based on set value of the DLEN[1:0] bits in the UARTn_LCR register.

The parity bit type can be determined by setting the PARITY and PEN bits in the UARTn_LCR register. If even parity is selected, the parity bit is determined by the bit sum of all the data bits. For odd parity, the parity bit takes the opposite value to that of even parity. The number of stop bits is defined by configuring the STOPBIT bit in the UARTn_LCR register.

Figure 94 shows an example of a transmitted data frame.

Figure 94. Example of Transmitted Data Frame



NOTES:

1. The UART peripheral and clock must be enabled to activate the UART transmitter function.
2. The data in the Transmit Shift Register is output on the TXD pin.

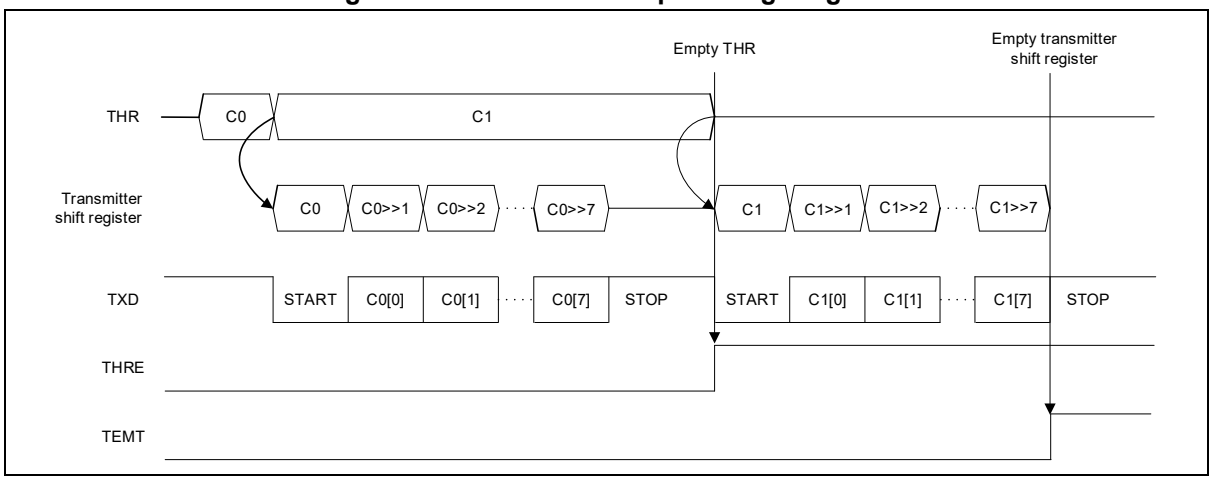
12.3.4.2 Transmit Interrupt

During UART transmission, several types of interrupt flags are used.

When the Transmit data Hold Register (UARTn_THR) is empty, the THRE interrupt flag is set to '1'; when the Transmit Shift Register is empty, the TEMT interrupt flag is set to '1'.

Users can select the best interrupt timing for their applications, by referring Figure 95.

Figure 95. Transmit Interrupt Timing Diagram



12.3.4.3 Character Transmission

During UART transmission, data is shifted out at the Least Significant Bit first on the TXD pin. In this mode, the UART transmission register consists of a buffer (UARTn_THR) between the internal bus and the Transmit Shift Register.(1)

Every character is preceded by a start bit which corresponds to a low logic level for one bit period. The character is terminated by a configurable number of stop bits. The number of stop bits can be configured to 1, 1.5 or 2, depending on the STOPBIT bit status.

NOTE:

1. The UART peripheral and clock enable bits must be set in the SCU_PER2 and SCU_PCER2 registers before writing the data to be transmitted.

The UART peripheral and clock enable bits must not be reset during data transmission. Resetting the UART peripheral and clock enable bits during the transmission corrupts the data on the TXD pin as the baud-rate counter gets frozen. The current data being transmitted are then lost.

12.3.4.4 Configurable Stop Bits

The number of stop bits to be transmitted with every character can be programmed in the STOPBIT bit in the UARTn_LCR register.

- 1 stop bit: Default number of stop bits.
- 1.5/2 stop bits: Number of stop bits that users can configure.

A break transmission is possible to transmit long breaks (break of length greater than the time of entire data word – i.e., The sum of the start bit, data bits, parity bit, and stop bits).

12.3.4.5 Character Transmission Procedure

Users can use DMA, Interrupts, and Polling to transmit the UART communication packets following the procedure below:

1. Entering data in the UARTn_THR register starts UART communication.
2. Since data in the UARTn_THR register is transferred to the Transmit Shift Register for the communication, updating the UARTn_THR register does not affect the data in communication.
   A. If the UARTn_THR register is updated, the Transmit Shift Register is automatically updated with the data in the UARTn_THR register after entire data in the Transmit Shift Register is output.
   B. The Transmit Shift Register continues to output its updated data for communication.
3. DMA, Interrupts, and Polling can be used to update the UARTn_THR register after confirming the communication is completed.
   A. With the DMA Tx enabled,
      i. The UART block generates the DMA Request signal immediately after data in the UARTn_THR register is transferred to the Transmit Shift Register.
      ii. The DMA updates the memory data with the UARTn_THR register. When this operation is repeated as many times as set in the DMA counter, the DMA Done signal is generated, the IID[2:0] bits is set to ‘101’, and the DMA Tx interrupt is generated.
   B. Checking if the TXE interrupt is generated or verifying that the IID[2:0] value is ‘001’,
      i. The TXE interrupt is generated when the communication is completed after the UARTn_THR register transfers data and the Transmit Shift Register outputs data. By checking the generation of the TXE interrupt, users can update the UARTn_THR register to restart communication.
      ii. If the IID[2:0] bits in the UARTn_IIR register is set to ‘001’ without the generation of the TXE interrupt, it means that data in the UARTn_THR register has been transferred to the Transmit Shift Register.
      iii. If the UARTn_THR register is not updated with new data, the TXE interrupt is generated when the Transmit Shift Register completes to output its current data.
4. Polling the TEMT and THRE bits in the UARTn_LSR register,
   A. The TEMT bit implies that the Transmit Shift Register is empty.
   B. The THRE bit implies that the UARTn_THR register is empty.

   (The UARTn_THR register can be updated only when TEMT and THRE bits are enabled.)

12.3.4.6 Single Byte Communication

When data is stored in the UARTn_THR (Transmitter Hold Register) register for transmission, it is automatically transferred to the Transmit Shift Register.

Before the Transmit Shift Register transfers data, the Transmitter outputs a start bit. After the start bit, the Transmit Shift Register shifts one bit at a time to the output of the Transmitter.

After data in the Transmit Shift Register is output by the bit size set in the DLEN[1:0] bits in the UARTn_LCR register, the Transmitter outputs a stop bit to complete the communication.

During the data transmission or after the communication is completed, if data in the UARTn_THR register is updated, the Transmit Shift Register that output data will be updated with new data to start next communication.

The Transmitter, which outputs a stop bit, restarts communication outputting a start bit.

12.3.4.7 Break Condition

Setting the BREAK(1) bit in the UARTn_LCR register transmits a long break (Break of length greater than the time of entire data word - i.e., The sum of the start, data, parity, and stop bits). Whereas writing a '0' to the bit disables the break condition.

NOTE:

1. If a '1' is written to the BREAK bit, a break is sent on the Tx line immediately. Therefore, users must confirm the completion of Tx transmission and set the BREAK bit.

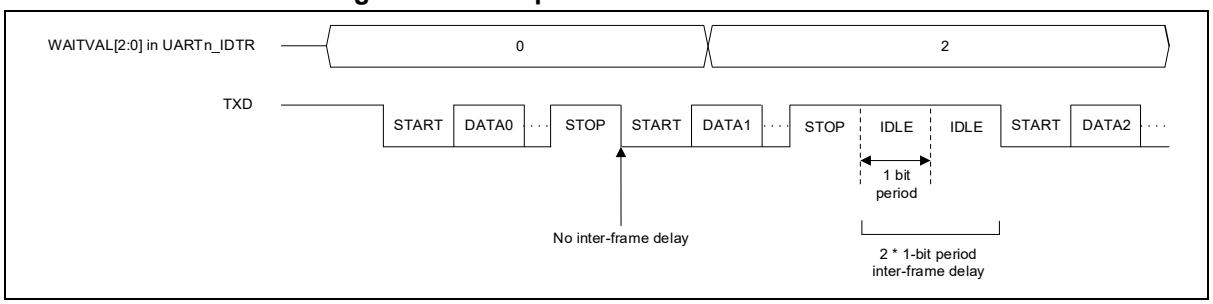
12.3.4.8 Inter-Frame Delay for Data Transmission

The inter-frame delay functionality creates an idle state between two characters on the TXD line. The length of the idle state can be determined by configuring the WAITVAL[2:0] bits in the UARTn_IDTR.

If the WAITVAL[2:0] bits are set to '000', a delay does not occur; otherwise, the transmitter waits for the number of bit periods defined in the WAITVAL[2:0] after transmitting the STOP part of TXD.

Figure 96 shows an example format of transmitting data.

Figure 96. Example of Transfer Data Format



12.3.5 UART Receiver

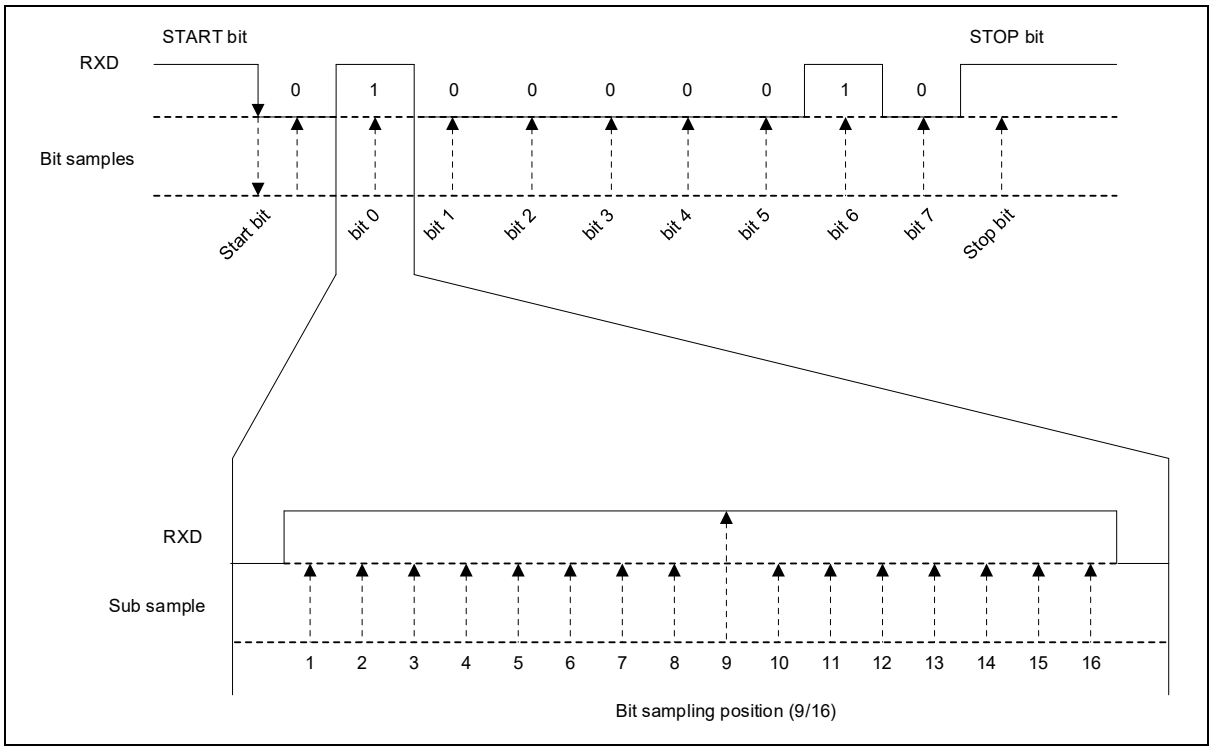
12.3.5.1 Receive Data Sampling Timing

Timing of the UART operation is described below:

Once a falling edge is detected in the receive line, the UART determines that a start bit is being received. UART oversamples the received data 16 times per bit beginning from the start bit. Among the 16 samples, the value at the nineth clock pulse is determined to represent the value of the bit.

The UART receiver can receive data words of either 5, 6, 7, or 8 bits, depending on the DLEN[1:0] bits in the UARTn_LCR register.

Figure 97. Sampling Timing of UART Receiver



To enhance protection against external glitch noise, it is recommended to enable port debouncing[1] in the PCU module.

NOTE:

1. To use the port debounce function, the debounce clock in the SCU_MCCR4 and SCU_MCCR5 registers must be enabled first. If the debounce function is activated without setting the debounce clock, the port may malfunction.

12.3.5.2 Start Bit Detection

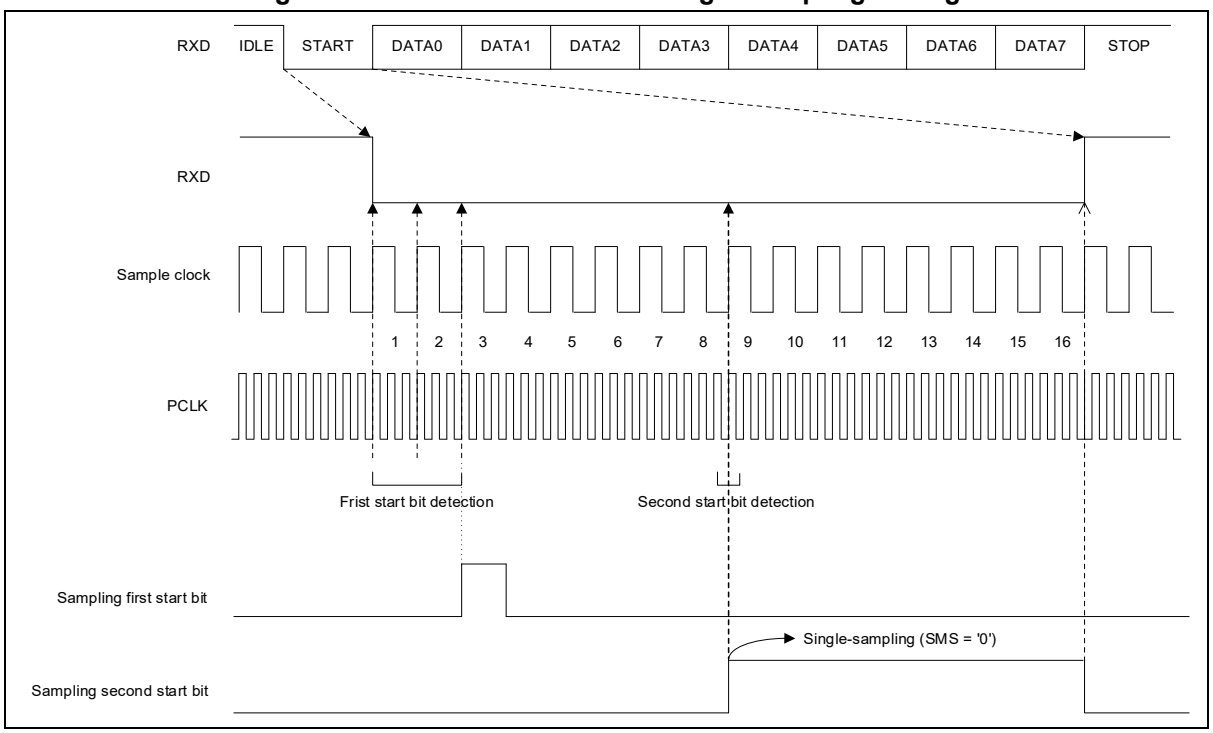
On the falling edge of the start bit, a digital filter rejects noise signal. A pulse shorter than ((3 × (1 / baud_rate)) / 16) s is rejected by this filter. Even when a pulse is longer than ((3 × (1 / baud_rate)) / 16) s, it can be rejected under the multi-sampling strategy during start-bit sampling.

In the UART, the start bit is detected when a specific sequence of samples is recognized as shown below:

- Single sampling sequence: 1 1 1 0 0 0 X X X X X 0 X X X X X X X
- Multi sampling sequence: 1 1 1 0 0 0 X X X X 0 0 0 X X X X X X

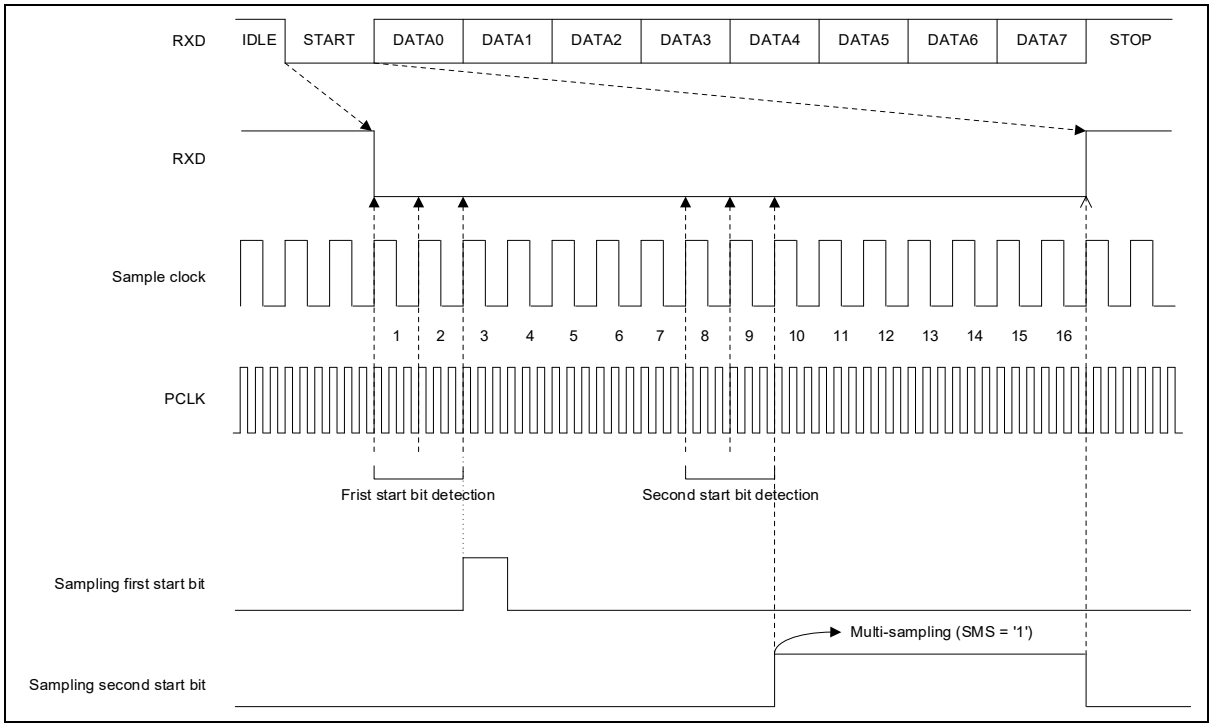
In single-sampling mode, the start bit is confirmed in the following sequence. The first sampling on the 1st, 2nd, and 3rd bits finds the three bits are all '0's, and the second sampling on the 9th bit also finds that the bit is '0'.

Figure 98. Start Bit Detection of Single-Sampling Timing



Similarly, the start bit is confirmed in the following sequence in multi-sampling mode. The first sampling on the 1st, 2nd, and 3rd bits finds that these three bits are all ‘0’s, and the second sampling on the 8th, 9th, and 10th bits finds that more than two bits are ‘0’s.

Figure 99. Start Bit Detection of Multi-Sampling Timing



NOTE:

1. If the sequence is not complete, the start bit detection stops and the receiver returns to the idle state, where it waits for a falling edge on RXD line.

12.3.5.3 Data Bit Sampling

If the DMS in UARTn_IDTR register is set to ‘1’, the values sampled at the 8th, 9th, and 10th clock pulses are determined to represent the value of the receive data bit. The multi-sampling technique is used for data recovery by discriminating between valid incoming data and noise.

Figure 100 shows the single sampling of data bits when the DMS is set to ‘0’.

Figure 100. Single-Sampling Timing of Data Bit

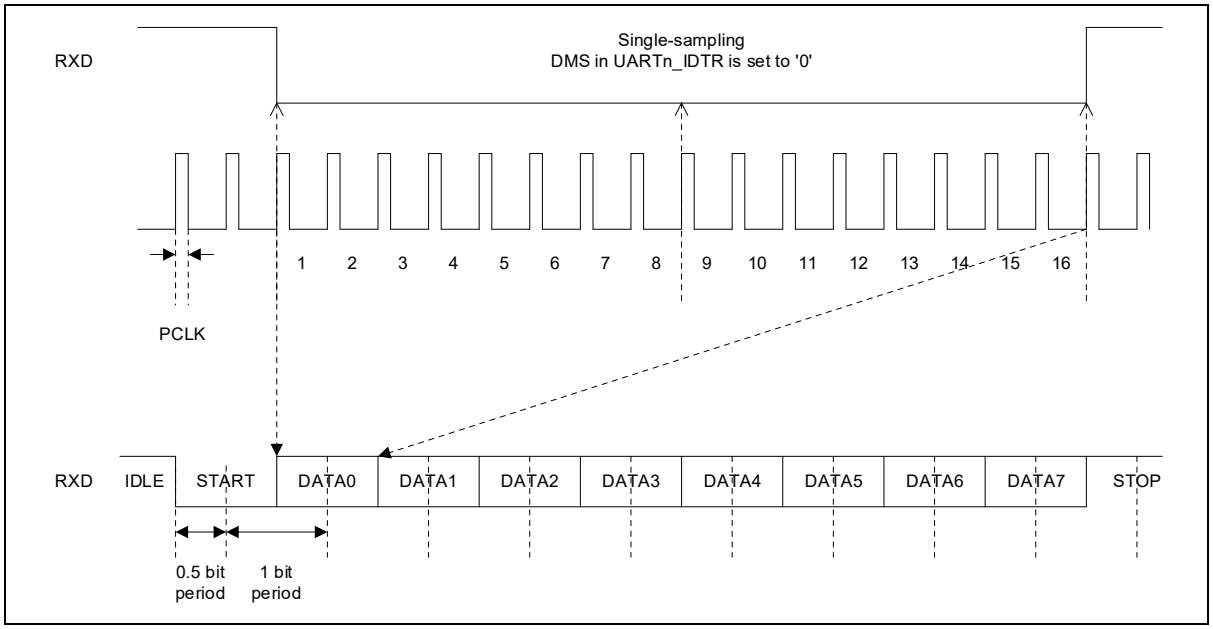
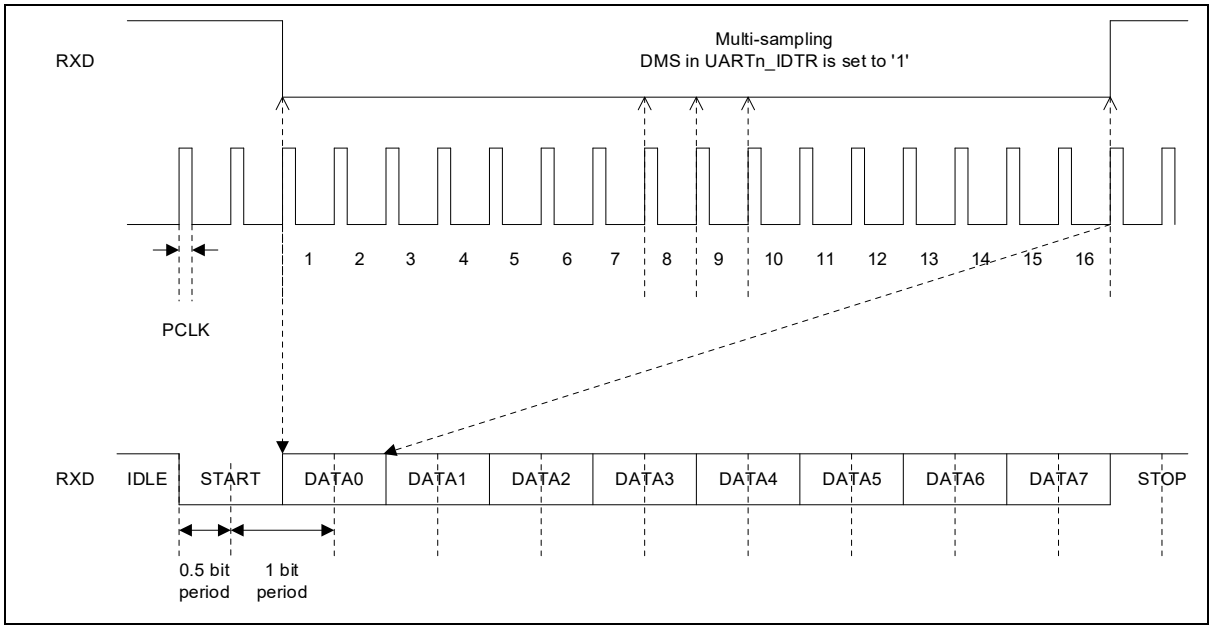


Figure 101 shows the multi-sampling of data bits when the DMS is set to ‘1’.

Figure 101. Multi-Sampling Timing of Data Bit



12.3.5.4 Data Sampling Strategy

An internal synchronous circuit is used in the receiver block to prevent abnormal noise in the RXD signal line. The start bit and data bits can be either single-sampled or multi-sampled. The SMS bit in the UARTn_IDTR register defines the sampling strategy for the start bit, and the DMS bit in the UARTn_IDTR register defines the sampling strategy for the data bits.

Figure 102. Data Sampling Timing

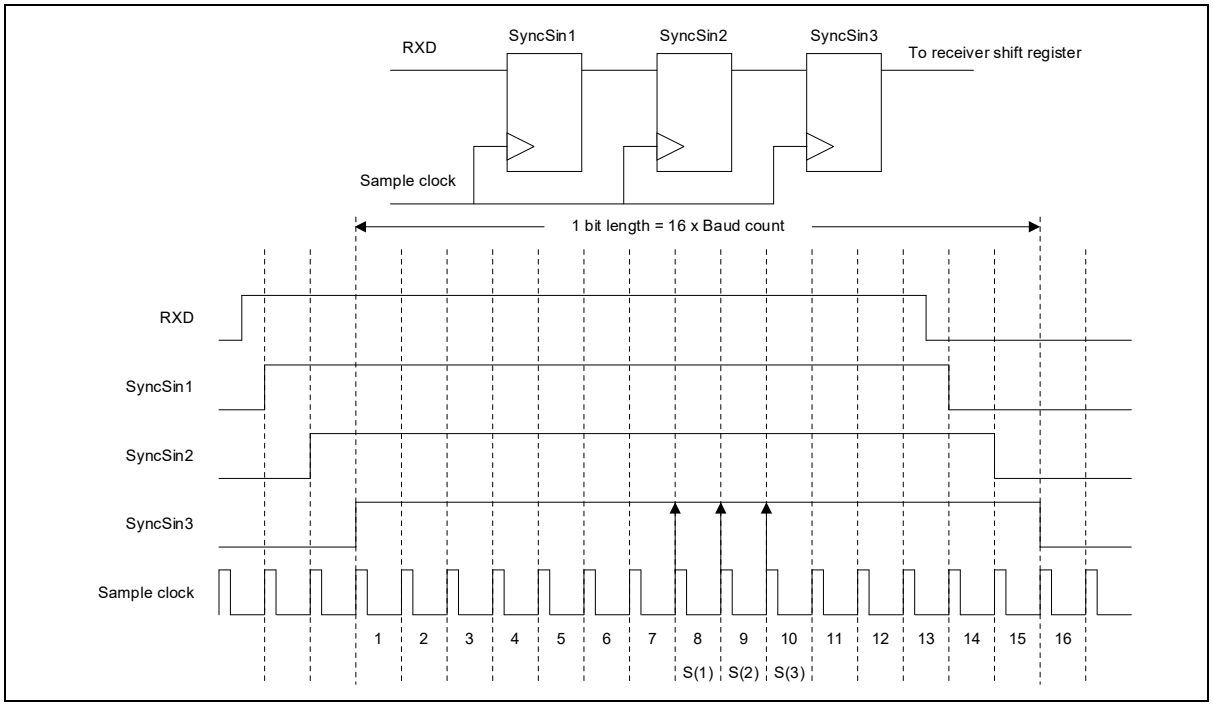


Figure 102 shows a block diagram and a sampling timing diagram of the anti-noise synchronous circuit.

If the DMS bit is set enabled, the value of each bit is sampled at multiple clock pulses. For the length of each data bit, values at S(1), S(2), and S(3) are taken out of 16 samples to determine the bit's value by taking the value indicated by a majority of the samples.

If the DMS bit is set disabled, on the other hand, the single-sampling process takes the value at S(2) only. The DMS bit does not affect the start bit, but the SMS bit works in a similar way for the Start bit.

Table 108. Received Bit Value from Sampled Data

Sampled Value	Received Bit Value	Sampled Value	Received Bit Value
000	0	100	0
001	0	101	1
010	0	110	1
011	1	111	1

12.3.5.5 Character Reception

When the falling edge is detected on the RXD Pin, the Receiver determines it as a start bit and starts the reception operation. The Data frame following the start bit is transferred one bit at a time at the LSB and stored in the Receiver Shift Register.

Receiving the stop bit means that a set of communication is completed. If the UARTn_RBR (Receive data Buffer Register) register is empty, data in the Receiver Shift Register is transferred into this register and kept until it is read out over the Bus.

[Character Reception Procedure]

DMA, Interrupt, and Polling can be used to read data from the UARTn_RBR register after confirming that the communication is completed, as described below:

- With DMA enabled, the DMA Rx Request signal is generated at the moment when data in the UARTn_RBR is updated. The DMA stores data of the UARTn_RBR into Memory as many times as set in the DMA counter and then sends the DMA Done signal to the UART. The UART outputs the DMA Rx Complete interrupt upon receiving the DMA Done signal.
- If the interrupt is generated, the IID[2:0] bits in the UARTn_IIR register must be checked. If the IID[2:0] bits is set to ‘010’, this means that the reception operation is completed and the UARTn_RBR register can be read.
- To confirm the reception is completed using polling, the DR in UARTn_LSR register must be checked. If data is received successfully, the DR bit is set to ‘1’ and data in the UARTn_RBR can be read. Reading the UARTn_RBR register clears the DR bit.

12.3.5.6 Break Condition

When a break condition is received, the UART generates break interrupt.

12.3.5.7 Overrun Error

The OE bit in the UARTn_LSR register indicates whether or not the Overrun Error has been detected. The Overrun Error occurs if the Receiver completes the new Reception operation when the UARTn_RBR data was not read (The DR bit in the UARTn_LSR register is '1'). At this moment, although the Overrun Error occurs, data in the UARTn_RBR register is updated.

Reading the UARTn_LSR register clears the OE bit.

12.3.5.8 Framing Error

When the Frame data transmission is finished, if the sampled value is read as ‘0’ at the point of time expected to receive the stop bit, the FE bit in the UARTn_LSR register is set to ‘1’ (Even when multi-sampling, only the first sampled value is read). This means that the Frame data transmission is not completed on time.

The Frame data size can be determined by configuring the DLEN[1:0] bits in the UARTn_LCR register. If the data size is set in the DLEN[1:0] bits and the received data’s bit size are different, problems may occur. Reading the UARTn_LSR register clears the FE bit.

12.3.5.9 Parity Error

The parity bit is appended to the data frame and provides a simple way to check whether the Receiver received correct data frame. If the parity bit is incorrect, the PE bit in the UARTn_LSR register is set to ‘1’. Reading the UARTn_LSR register clears the PE bit.

12.3.5.10 Configurable Stop Bits During Reception

During Reception operations, the first stop bit is sampled regardless of the STOPBIT bit in the UARTn_LCR register value. Users can change the STOPBIT value while the Receiver is operating.

12.3.6 UART Baud-Rate Generation

To establish communication via a UART channel, users must set an appropriate baud-rate. To this end, a programmable baud-rate generator is built in, providing a baud-rate divisor ranging from 1 to 65,535.

Users must write an appropriate divisor value to the 16-bit UARTn_BDR register to obtain the desired baud-rate.

Below is the formula for calculating the baud-rate:

$$\text{UARTn_BDR} = \frac{(\text{UART clock set by SCU}_{\text{MCCR7}})}{16 \times \ BaudRate}$$

If the UART clock has been set in the SCU_MCCR7(1) register, the PCLK clock speed must satisfy the formula below:

$$\text{PCLK} \geq (\text{UART clock set by SCU}_{\text{MCCR7}}) \times 2$$

NOTE:

1. The UART only uses a clock source set by MCCR7. It does not use PCLK.

Table 109 lists the divisor and error rate for each baud-rate, at the UART clock speed of 100 MHz.

Table 109. Examples of Baud Rate Calculation

UART Clock = 100 MHz		
Baud-rate	**Divisor (UARTn_BDR)**	**Error (%)**
1,200	5,208	0.01 %
2,400	2,604	0.01 %
4,800	1,302	0.01 %
9,600	651	0.01 %
19,200	325	0.16 %
38,400	162	0.47 %
57,600	108	0.47 %
115,200	54	0.47 %

The 8-bit fraction counter compensates for the fractional part of the actual baud-rate divisor. This is needed because the baud-rate divisor latch register indicates the integer part of the baud-rate divisor only.

By adding the fractional part indicated in the UARTn_BFR register to the integer part indicated in the UARTn_BDR register, the correct divisor is obtained to calculate the accurate baud-rate.

NOTE:

1. If the UARTn_BDR register value is equal to or less than two, the UARTn_BFR register value is always 0 regardless of the UARTn_BFR register's set value.

For example, when the baud-rate is 9,600 bps:

$$\frac{UART\ clock}{16 \times\ BaudRate} = \frac{100000000}{16 \times\ 9600} = 651.0416, \qquad Divisor = 651, Float = 0.0416$$

$$CNT = Float \times 256\ = 0.0416 \times 256 = 10$$

$$UARTn_BDR\ =\ 651, UARTn_BFR\ =\ 10$$

Table 110. Examples of Calculation of Baud-Rate Using UART Fraction Counter

UART Clock = 100 MHz			
Baud-rate	**Divisor (UARTn_BDR)**	**FCNT (UARTn_BFR)**	**Error (%)**
1200	5,208	85	0.0 %
2400	2,604	42	0.0 %
4800	1,302	21	0.0 %
9600	651	10	0.0 %
19200	325	133	0.0 %
38400	162	194	0.0 %
57600	108	129	0.0 %
115200	54	64	0.0 %

12.3.7 Tolerance of UART Receiver to Clock Deviation

During Rx operation, each bit of data is sampled 16 times. To determine the value of the read data, the 8th sampling signal is used for Single-sampling, and the 8th, 9th, and 10th sampling signals are used for multi-sampling.

Therefore, a slight difference is allowed unless the UART Rx signal is outside the sampling point within a frame (Regardless of whether it is faster or slower). This can be calculated by the formular shown below:

- Single-sampling: 7 / (16 × Total_bit_size)
- Multi-sampling: 6 / (16 × Total_bit_size)
- Total_bit_size: Receive bit size. (Data bit (5 to 8 bits) + Start bit + Stop bit)

Based on the calculation results from the formular above, Table 111 shows an acceptable error that depends on the bit size of transmitted data.

Table 111. Tolerance of UART Receiver

Sampling	DLEN[1:0] Bits	UART Frame(1)	Tolerance of UART Receive
Single-Sampling	00	| St | 5-bit data | Sp |	7 / (16 × 7) = 6.25%
	01	| St | 6-bit data | Sp |	7 / (16 × 8) = 5.46%
	10	| St | 7-bit data | Sp |	7 / (16 × 9) = 4.86%
	11	| St | 8-bit data | Sp |	7 / (16 × 10) = 4.37%
Multi-Sampling	00	| St | 5-bit data | Sp |	6 / (16 × 7) = 5.35%
	01	| St | 6-bit data | Sp |	6 / (16 × 8) = 4.68%
	10	| St | 7-bit data | Sp |	6 / (16 × 9) = 4.16%
	11	| St | 8-bit data | Sp |	6 / (16 × 10) = 3.75%

NOTE:

1. St: Start bit, Sp: Stop bit.

12.3.8 UART Parity Control

Parity control (Generation of parity bit in transmission and parity checking in reception) can be enabled by setting the PEN in UARTn_LCR register. Depending on the frame length defined in the DLEN[1:0] in UARTn_LCR register, the possible UART frame formats are as listed in Table 112.

Table 112. UART Frame Formats

DLEN[1:0] Bits	PEN Bit	UART Frame(1)(2)
00	0	| St | 5-bit data | Sp |
00	1	| St | 5-bit data | P | Sp |
01	0	| St | 6-bit data | Sp |
01	1	| St | 6-bit data | P | Sp |
10	0	| St | 7-bit data | Sp |
10	1	| St | 7-bit data | P | Sp |
11	0	| St | 8-bit data | Sp |
11	1	| St | 8-bit data | P | Sp |

NOTES:

1. St: Start bit, Sp: Stop bit, P: Parity bit
2. In the data register, the parity is always taking the MSB position (5th, 6th, 7th or 8th depending on the DLEN[1:0] bits value).

12.3.8.1 Even Parity

The parity bit is calculated to obtain an even number of ‘1’s inside the frame made of the 5, 6, 7 or 8 LSB bits (Depending on the DLEN[1:0] bit values) and the parity bit when PEN bit is ‘1’.

As an example, if data = ’01011001’, and 4 bits are set, then the parity bit will be ‘0’ if even-parity is selected (PARITY=1 and PEN= 1).

12.3.8.2 Odd Parity

The parity bit is calculated to obtain an odd number of ‘1’s inside the frame made of the 5, 6, 7 or 8 LSB bits (Depending on the DLEN[1:0] bit values) and the parity bit when PEN bit is ‘1’.

As an example, if data = ’01011001' and four bits are set, then the parity bit will be ‘1’ if odd-parity is selected (PARITY=0 and PEN=1).

12.3.8.3 Stick Parity

The STICKP bit in the UARTn_LCR register determines whether to use stick parity.

- If PEN = '1', PARITY = '1', and STICKP = '1', then the parity bit is always '0'.
- If PEN = '1', PARITY = '0', and STICKP = '1', then the parity bit is always '1'.
- If STICKP = '0', stick parity is disabled.

12.3.8.4 Various Configurations of Parity Bit

A parity bit is generated based on the settings of the PEN, PARITY and STICKP bit in the UARTn_LCR register.

Table 113 describes various configurations to create a parity bit.

Table 113. Various Configurations to Create Parity Bit

STICKP	PARITY	PEN	Parity
X	X	0	No parity
0	0	1	Odd parity
0	1	1	Even parity
1	0	1	Forced '1' to stick parity
1	1	1	Forced '0' to stick parity

12.3.8.5 Parity Checking in Reception

If the parity check fails, the PE flag is set by configuring the PE bit in the UARTn_LSR register, and the corresponding interrupt is generated if the RLSIE bit in the UARTn_IER register is set to '1'.

Reading the UARTn_LSR register clears the PE bit to '0'.

12.3.8.6 Parity Generation in Transmission

If the PEN bit in the UARTn_LCR register is set to '1', the Transmitter generates a parity bit between the last data bit and the stop bit. The parity bit is set to '0' or '1' to ensure that the total number of '1's in the data and parity bits is either even or odd depending on how the PARITY and STICKP bits are set.

12.3.9 Continuous Communication Using DMA and UART

The UART module supports DMA transfers using the DMA device built in the microcontroller. The start of the memory address for data transfer and the length of the data to be transferred are determined by the programming registers in the DMA module.

Completion of the data transfer is related to a notification by the transfer complete flag. Once the entire transmit data is written to the Transmit data Hold Register (THR), the UARTn_IIR register's DMA Tx complete flag (TXE bit) is set and the interrupt source ID of the complete interrupt is written to the IID[2:0] in UARTn_IIR register.

Similarly, once the entire receive data is written to the target DMA memory, the UARTn_IIR register's DMA Rx complete flag is set and the interrupt source ID of the complete interrupt is written to the IID[2:0] bits in the UARTn_IIR register. Therefore, the UART RXD signal already becomes idle when the DMA Rx complete interrupt occurs.

12.3.9.1 Transmission Using DMA

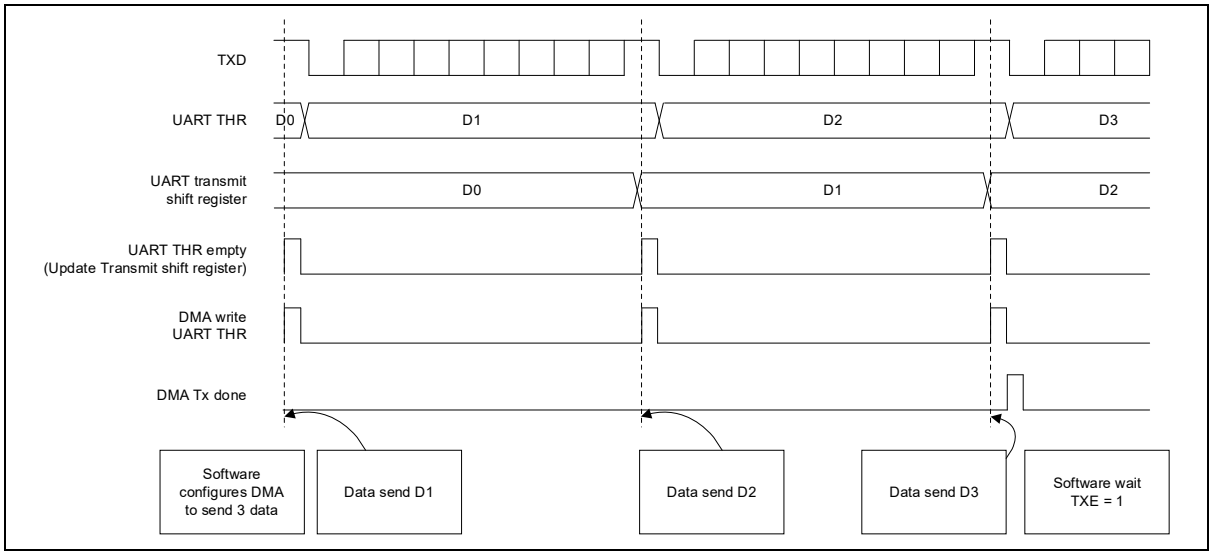
With DMA enabled, when the Transmit Shift Register is updated, the DMA Tx request signal is generated to allow the UARTn_THR register to store new data.

When the UARTn_THR register stores the new data using DMA transfer and the Transmit Shift Register completes to output data, the UARTn_THR register transmits the new data to the Transmit Shift Register again. The data in the Transmit Shift Register is output to the TXD Line according to the Transmitter operation.

After the DMA transmits data as many times as set in the DMA Tx count (the set number in Figure 103 is 3), the DMA generates the DMA Tx done signal. Upon receiving this signal, UART outputs the DMA Tx complete interrupt to inform that the DMA Tx operation is completed.

When the DMA operation is completed, unless new data is written to the UARTn_THR register, the last data in the Transmit Shift Register is output and the Transmitter operation is complete. (TXE interrupt is generated.)

Figure 103. Transmission using DMA



12.3.9.2 Reception Using DMA

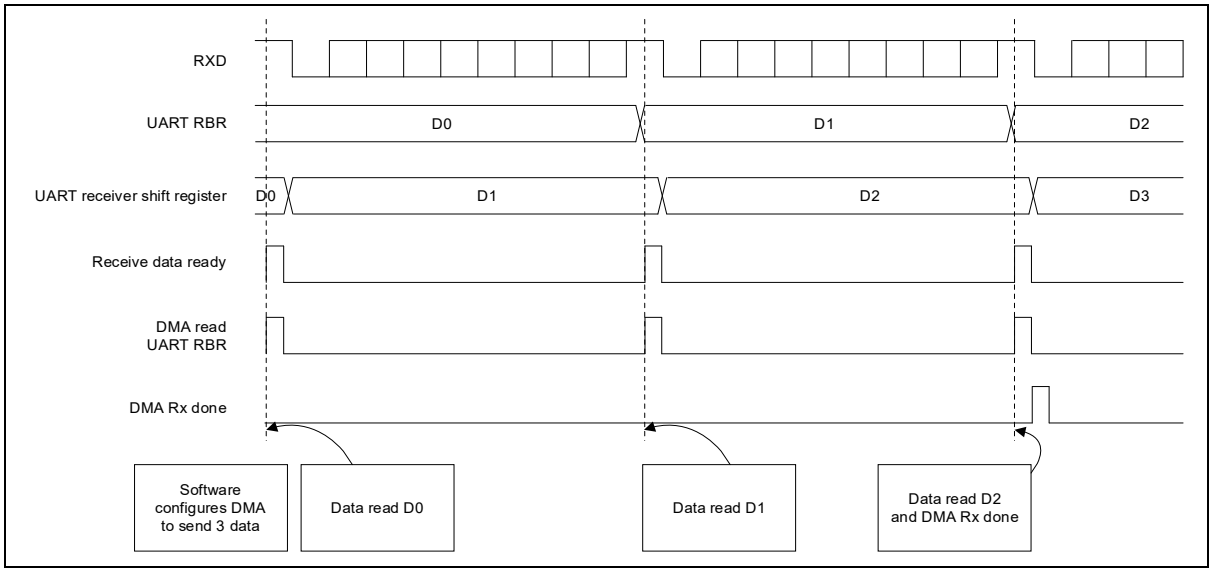
When the Receiver Shift Register is updated according to the reception operation, new data is stored in the UARTn_RBR register. With DMA enabled, when the UARTn_RBR register is updated with new data, the DMA Rx Request signal is generated.

DMA reads the UARTn_RBR register to store the data into memory.

After the DMA stores the data of the UARTn_RBR register into memory as many times as set in the DMA Rx count (The set number in Figure 104 is 3), the DMA generates the DMA Rx done signal.

Upon receiving this signal, the UART outputs the DMA Rx Complete interrupt to inform that the DMA Rx operation is completed.

Figure 104. Reception using DMA



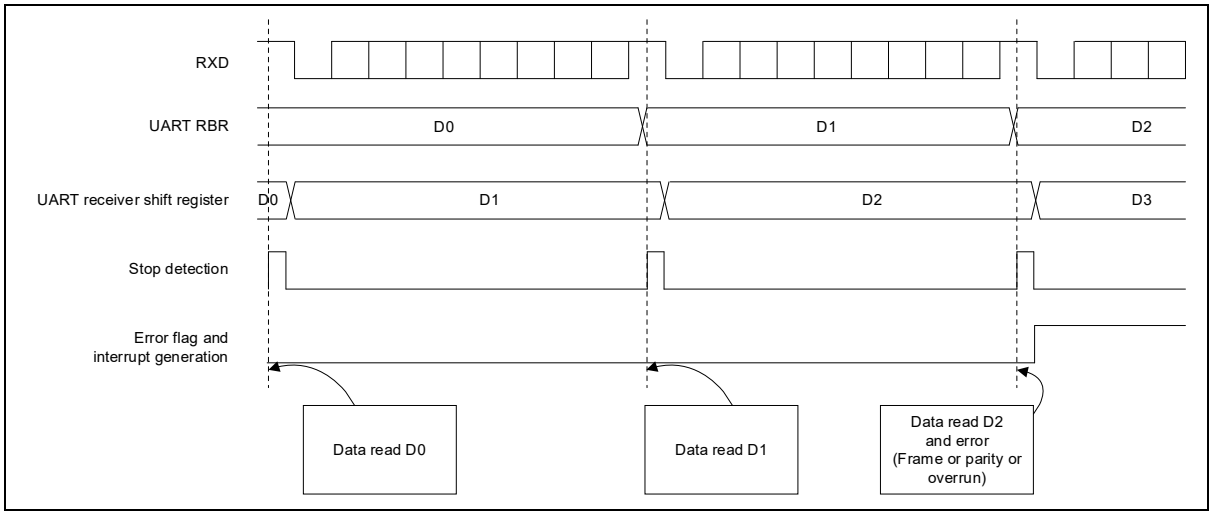
12.3.9.3 Error Flagging and Interrupt Generation in Communication

Frame errors, Parity errors, and Overrun errors may occur after the data receiving is completed, during Rx operation. If one of these errors occurs as a result of the Rx operation, the Receive Line Status Error Interrupt is generated depending on the settings of the RLSIE bit in the UARTn_IER register.

At this point of time, the IID bits in the UARTn_IIR register represents the interrupt source ID, and the UARTn_LSR register shows the type of error. The UARTn_LSR register is updated whenever an error occurs, even if the RLSIE bit in the UARTn_IER register is disabled.

Although an error occurs, data in the Receiver Shift Register is transmitted to UARTn_RBR register and then UART Rx operation is maintained normally.

Figure 105. Error Fragging and Interrupt Generation Timing Diagram



12.4 UART Interrupts

During UART communications, various interrupts can be generated by different events. They can be checked by reading the UARTn_IIR register.

When the microcontroller core accesses the UARTn_IIR register, the UART locks all interrupts and the highest-priority interrupt is read.

Interrupts can occur even while the register is being accessed by the microcontroller core; however, the interrupt status remains unchanged until the current access is completed.

Among the pending interrupts, the highest-priority interrupt's source ID is represented to the IID[2:0] bits in the UARTn_IIR register.

The UART module uses a total of seven interrupts with different priorities. These interrupts include the followings:

- Receive line status interrupt
- Receive data ready interrupt
- Transmit data hold register empty interrupt
- DMA receive complete interrupts
- DMA transmit complete interrupts

Additionally, the UART module supports the TXE interrupt, which is triggered when both the Transmit shift register and the Transmit data hold register are empty.

12.5 UART Registers

The base address and register map of the UART are described in the following tables:

Table 114. Base Address of UART

Name	Base Address
UART0	0x4000_8000
UART1	0x4000_8100
UART2	0x4000_8200
UART3	0x4000_8300
UART4	0x4000_8400
UART5	0x4000_8500
UART6	0x4000_8600
UART7	0x4000_8700
UART8	0x4000_8800
UART9	0x4000_8900
UART10	0x4000_8A00
UART11	0x4000_8B00
UART12	0x4000_8C00

Table 115. UART Register Map

Name	Offset	Type	Description	Reset Value	Reference
UARTn_RBR	0x0000	RO	UART n Receive Data Buffer Register	0x0000_0000	12.5.1
UARTn_THR	0x0000	WO	UART n Transmit Data Hold Register	-	12.5.2
UARTn_IER	0x0004	RW	UART n Interrupt Enable Register	0x0000_0000	12.5.3
UARTn_IIR	0x0008	RO	UART n Interrupt ID Register	0x0000_0001	12.5.4
UARTn_LCR	0x000C	RW	UART n Line Control Register	0x0000_0000	12.5.5
UARTn_DCR	0x0010	RW	UART n Data Control Register	0x0000_0000	12.5.6
UARTn_LSR	0x0014	RO	UART n Line Status Register	0x0000_0060	12.5.7
UARTn_ELSR	0x0014	RO	UART n Line Status Register	0x0000_0060	12.5.8
UARTn_BDR	0x0020	RW	UART n Baud-rate Divisor Latch Register	0x0000_0000	12.5.9
UARTn_BFR	0x0024	RW	UART n Baud-rate Fraction Counter Register	0x0000_0000	12.5.10
UARTn_IDTR	0x0030	RW	UART n Inter-Frame Delay Time Register	0x0000_0000	12.5.11
UARTn_FCR	0x0040	RW	UART n FIFO Control Register	0x0000_0400	12.5.12
UARTn_FRBR	0x0044	RO	UART n Receive FIFO Register	0x0000_0000	12.5.13

NOTE:

1. n = 0, 1, 2, 3, 4, 5, 6, 7, 8, 9, 10, 11 and 12.

12.5.1 UARTn_RBR: UART n Receive Data Buffer Register

UARTn_RBR is an 8-bit read-only register. Received data is read from this register, and the maximum length of data is 8 bits. The last data received is retained until a new byte is received.

UART0_RBR=0x4000_8000, UART1_RBR=0x4000_8100, UART2_RBR=0x4000_8200
UART3_RBR=0x4000_8300, UART4_RBR=0x4000_8400, UART5_RBR=0x4000_8500
UART6_RBR=0x4000_8600, UART7_RBR=0x4000_8700, UART8_RBR=0x4000_8800
UART9_RBR=0x4000_8900, UART10_RBR=0x4000_8A00
UART11_RBR=0x4000_8B00, UART12_RBR=0x4000_8C00

31 30 29 28 27 26 25 24	23 22 21 20 19 18 17 16	15 14 13 12 11 10 9 8	7 6 5 4 3 2 1 0
Reserved			RBR[7:0]
-			0x00
-			RO

7 0	RBR[7:0]	Receive buffer register

12.5.2 UARTn_THR: UART n Transmit Data Hold Register

UARTn_THR is an 8-bit write-only register. Data is stored in this register to be transmitted. However, data written to UARTn_THR cannot be read but sent to the Transmit Shift Register when this register is empty.

UART0_THR=0x4000_8000, UART1_THR=0x4000_8100, UART2_THR=0x4000_8200
UART3_THR=0x4000_8300, UART4_THR=0x4000_8400, UART5_THR=0x4000_8500
UART6_THR=0x4000_8600, UART7_THR=0x4000_8700, UART8_THR=0x4000_8800
UART9_THR=0x4000_8900, UART10_THR=0x4000_8A00
UART11_THR=0x4000_8B00, UART12_THR=0x4000_8C00

31 30 29 28 27 26 25 24	23 22 21 20 19 18 17 16	15 14 13 12 11 10 9 8	7 6 5 4 3 2 1 0
Reserved			THR[7:0]
-			-
-			WO

7 0	THR[7:0]	Transmit data hold register

12.5.3 UARTn_IER: UART n Interrupt Enable Register

UARTn_IER enables six types of UART interrupts. Each interrupt generates an interrupt output signal. If users set all bits in the UARTn_IER register to '0', all UART interrupts become disabled. A particular interrupt can be set by setting the corresponding bit in this register to '1'.

UART0_IER=0x4000_8004, UART1_IER=0x4000_8104, UART2_IER=0x4000_8204
UART3_IER=0x4000_8304, UART4_IER=0x4000_8404, UART5_IER=0x4000_8504
UART6_IER=0x4000_8604, UART7_IER=0x4000_8704, UART8_IER=0x4000_8804
UART9_IER=0x4000_8904, UART10_IER=0x4000_8A04
UART11_IER=0x4000_8B04, UART12_IER=0x4000_8C04

31 30 29 28 27 26 25 24	23 22 21 20 19 18 17 16	15	14 13 12	11 10 9 8	7 6	5	4	3	2	1	0
Reserved		MODE	UMKEY	Reserved		DTXIEN	DRXIEN	TXEIE	RLSIE	THREIE	DRIE
-		0	000	-		0	0	0	0	0	0
-		RW	WO	-		RW	RW	RW	RW	RW	RW

15	MODE	UARTn mode selection bit.
		0 IIR legacy mode
		1 IIR extended mode
14 12	UMKEY	UARTn mode access key.
		Writing '0x5' to the bit field enables writing new values to MODE bit in UARTn_IER registers.
5	DTXIEN	Enable or disable the DMA transmit complete interrupt
		0 Disables the DMA transmit complete interrupt.
		1 Enables the DMA transmit complete interrupt.
4	DRXIEN	Enable or disable the DMA receive complete interrupt
		0 Disables the DMA receive complete interrupt.
		1 Enables the DMA receive complete interrupt.
3	TXEIE	Enable or disable the transmit complete interrupt
		0 Disables the transmit complete interrupt.
		1 Enables the transmit complete interrupt.
2	RLSIE	Enable or disable the receive line status interrupt
		0 Disables the receive line status interrupt.
		1 Enables the receive line status interrupt.
1	THREIE	Enable or disable the transmit data hold register empty (THRE) interrupt
		0 Disables the THRE interrupt.
		1 Enables the THRE interrupt.
0	DRIE	Enable or disable the data receive interrupt
		0 Disables the data receive interrupt.
		1 Enables the data receive interrupt.

12.5.4 UARTn_IIR: UART n Interrupt ID Register

[Legacy mode]

The UARTn_IIR register is a read-only register informing the operation of the interrupt that is set to enabled in the UARTn_IER register. The UARTn_IIR register is supported when the MODE bit in the UARTn_IER register is set to '0'. In Extended mode, it is always read as '0' and cannot be used.

UART0_IIR=0x4000_8008, UART1_IIR=0x4000_8108, UART2_IIR=0x4000_8208
UART3_IIR=0x4000_8308, UART4_IIR=0x4000_8408, UART5_IIR=0x4000_8508
UART6_IIR=0x4000_8608, UART7_IIR=0x4000_8708, UART8_IIR=0x4000_8808
UART9_IIR=0x4000_8908, UART10_IIR=0x4000_8A08
UART11_IIR=0x4000_8B08, UART12_IIR=0x4000_8C08

31 30 29 28 27 26 25 24	23 22 21 20 19 18 17 16	15	14	13	12 11 10 9 8	7 6 5	4	3 2 1	0
Reserved		RXFTLSTS	RXFNESTS	RXFFSTS	Reserved		TXE	IID[2:0]	IPEN
-		0	0	0	-		0	000	1
-		RO	RO	RO	-		RO	RO	RO

15	RXFTLSTS	Receive FIFO trigger level status.
	0	Receive FIFO is not triggering level.
	1	Receive FIFO is triggering level.
14	RXFNESTS	Receive FIFO not empty status.
	0	Receive FIFO is empty
	1	Receive FIFO is not empty
13	RXFFSTS	Receive FIFO full status.
	0	Receive FIFO is not full.
	1	Receive FIFO is full.
4	TXE	Transmit complete (refer to Table 116)
		TXE Indicates whether or not the transmit shift register is empty. TXE = 1 indicates that UARTn_THR and transmit shift register is currently empty and thus new data can be written to the UARTn_THR register.
3 1	IID[2:0]	Interrupt source ID (Refer to Table 116, Table 117)
		IID[2:0] Displays the interrupt with the highest priority among those currently active.
0	IPEN	Presence of pending interrupts IPEN Indicates there are currently unprocessed interrupts. IPEN=0 indicates that a certain interrupt condition has occurred, and IPEN=1 means that there are no currently unprocessed interrupts.
	0	An interrupt is pending
	1	There are no interrupts pending

NOTE:

1. When reading UARTn_IIR, if a TXE (Transmit complete) occurs, the TXE flag may be unintentionally cleared. Additionally, if any of RLSIE, THREIE, or DRIE are set along with DRXIEN in UARTn_IER, the DMA RX Complete Interrupt flag in UARTn_IIR may also be unintentionally cleared. For more details, please refer to the Errata.

Table 116. TXE Bit Control

Priority Level	TXE	Interrupt Source		
	Bit 4	Interrupt Name	Interrupt Condition	Interrupt Clear
-	1	Transmit register empty	Transmit shift register and transmit data hold register are empty	Write to the transmit data hold register or read UARTn_IIR

Table 117. Interrupt ID and Control

Priority Level	TXE	IID[2:0]			IPEN	Interrupt Source		
	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0	Interrupt Name	Interrupt Condition	Interrupt Clear
-	0	0	0	0	1	N/A	-	-
1	0	0	1	1	0	Receive line status	Overrun, parity, framing, break error, etc.	Read UARTn_LSR
2	0	0	1	0	0	Receive data present	There is data received	Read the receive register
3	0	0	0	1	0	Transmit hold register empty	Transmit data hold register empty	Write to the transmit data hold register or UARTn_IIR
4	0	1	1	0	0	DMA Rx complete	DMA Rx complete	Read UARTn_IIR
5	0	1	0	1	0	DMA Tx complete	DMA Tx complete	Read UARTn_IIR

12.5.5 UARTn_LCR: UART n Line Control Register

The UARTn_LCR register is an 8-bit register.

UART0_LCR=0x4000_800C, UART1_LCR=0x4000_810C, UART2_LCR=0x4000_820C
UART3_LCR=0x4000_830C, UART4_LCR=0x4000_840C, UART5_LCR=0x4000_850C
UART6_LCR=0x4000_860C, UART7_LCR=0x4000_870C, UART8_LCR=0x4000_880C
UART9_LCR=0x4000_890C, UART10_LCR=0x4000_8A0C
UART11_LCR=0x4000_8B0C, UART12_LCR=0x4000_8C0C

31 30 29 28 27 26 25 24	23 22 21 20 19 18 17 16	15 14 13 12 11 10 9 8	7	6	5	4	3	2	1 0
Reserved				BREAK	STICKP	PARITY	PEN	STOPBIT	DLEN[1:0]
-				0	0	0	0	0	00
-				RW	RW	RW	RW	RW	RW

6	BREAK	When this bit is set to '1', the TXD pin is driven low to alert the receiver. BREAK is a break control bit. If BREAK = '1', the serial output pin (TXD pin) is driven low (Spacing state), whereas writing a 0 to the bit disables the break condition. 0 General communication mode 1 Transmit break mode
5	STICKP	Use stick parity. The setting of this bit is valid only when the PEN bit is set to '1'. STICKP determines whether or not to use stick parity. If PEN = '1', PARITY = '1', and STICKP = '1', the parity bit is always set low. If PEN = '1', STICKP = '1', and PARITY = '0', the transmit parity bit is always set high. If STICKP = 0, stick parity is disabled. 0 Does not use stick parity. 1 Uses stick parity.
4	PARITY	Parity mode selection PARITY determines which parity mode is used between odd and even parity. If PEN = '1' and PARITY = '0', odd parity is used depending upon the number of data bits; if PEN = '1' and PARITY = '1', even parity is used. 0 Odd-parity mode 1 Even-parity mode
3	PEN	Enable parity PEN determines whether or not to enable the use of parity. If PEN = '1', the transmitter creates a parity bit between the last data bit and the stop bit and the receiver inspects this parity bit. (The parity bit is set to '0' or '1' to ensure that the total number of 1s in the data and parity bits is either even or odd depending on how the PARITY and STICKP bits are set. 0 Disables parity. 1 Enables parity.
2	STOPBIT	The number of stop bits demanded by the preceding data bits STOPBIT defines the number of stop bits attached to the end of each transmitted/received data word. STOPBIT = '0' includes 1 stop bit at the end of each transmitted data word. STOPBIT = '1' includes 1.5 or 2 stop bits at the end of each transmitted data word depending upon the number of transmit bits defined at DLEN[1:0]; a 5-bit data word includes 1.5 stop bits and a 6-, 7-, or 8-bit data word includes 2 stop bits. The receiver checks only the first stop bit regardless of the number of stop bits defined at DLEN[1:0] and STOPBIT.

		0	The number of stop bits is 1.
		1	The number of stop bits is 1.5 or 2. If the data word is 5 bits long, 1.5 stop bits are added. If the data word is 6, 7, or 8 bits long, 2 stop bits are added.
1 0	DLEN[1:0]	The length of data bits included by each data transfer DLEN[1:0] defines the length of each transmitted/received data bit length.	
		00	5-bit data
		01	6-bit data
		10	7-bit data
		11	8-bit data

12.5.6 UARTn_DCR: UART n Data Control Register

The UARTn_DCR register is a 32-bit register that controls Tx or Rx data inversion.

UART0_DCR=0x4000_8010, UART1_DCR=0x4000_8110, UART2_DCR=0x4000_8210
UART3_DCR=0x4000_8310, UART4_DCR=0x4000_8410, UART5_DCR=0x4000_8510
UART6_DCR=0x4000_8610, UART7_DCR=0x4000_8710, UART8_DCR=0x4000_8810
UART9_DCR=0x4000_8910, UART10_DCR=0x4000_8A10
UART11_DCR=0x4000_8B10, UART12_DCR=0x4000_8C10

31 30 29 28 27 26 25 24	23 22 21 20 19 18 17 16	15 14 13 12 11 10 9 8	7 6 5 4	3	2	1 0
Reserved				RXINV	TXINV	Reserved
-				0	0	-
-				RW	RW	-

3	RXINV	Receive data inversion selection	
		0	Inputs normal receive data.
		1	Inputs inverted receive data.
2	TXINV	Transmit data inversion selection	
		0	Outputs normal transmit data.
		1	Outputs inverted transmit data.

12.5.7 UARTn_LSR: UART n Line Status Register

[Legacy mode]

The UARTn_LSR register is supported when MODE bit in UARTn_IER register is set to '0'.

The UARTn_LSR register is a read-only register that shows the status of data transmission and reception. This register reports the status of data transfers between the transmitter and receiver. Through this register, users can check the status of the UART lines and set interrupts as follows: Status interrupts for bits 1, 2, 3, and 4 can be called if they are set enabled in the RLSIE bit in the UARTn_IER register. Other bits enable their corresponding interrupts to be called if they are set enabled in the UARTn_IER register.

UART0_LSR=0x4000_8014, UART1_LSR=0x4000_8114, UART2_LSR=0x4000_8214
UART3_LSR=0x4000_8314, UART4_LSR=0x4000_8414, UART5_LSR=0x4000_8514
UART6_LSR=0x4000_8614, UART7_LSR=0x4000_8714, UART8_LSR=0x4000_8814
UART9_LSR=0x4000_8914, UART10_LSR=0x4000_8A14
UART11_LSR=0x4000_8B14, UART12_LSR=0x4000_8C14

31 30 29 28 27 26 25 24	23 22 21 20 19 18 17 16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved	FRLVL[3:0]	RXFTLIF	RXFNEIF	RXFFIF	Reserved		DTXBUSY	DRXBUSY	Reserved		TEMT	THRE	BI	FE	PE	OE	DR
-	0x0	0	0	0	-		0	0	-		1	1	0	0	0	0	0
-	RO	RWC1	RWC1	RWC1	-		RO	RO	-		RO	RO	RO	RO	RO	RO	RO

19 16	FRLVL[3:0]	FIFO Reception level
	0	FIFO Rx buffer empty
	1	Receive 1 data
	2	Receive 2 data
	3	Receive 3 data
	4	Receive 4 data
	5	Receive 5 data
	6	Receive 6 data
	7	Receive 7 data
	8	FIFO buffer full (Receive 8 data)
15	RXFTLIF	Receive FIFO trigger level interrupt flag
	0	The receive FIFO trigger level event has not occurred
	1	The receive FIFO trigger level event has occurred. It occurs when the receive trigger level matches RXFTLSEL in the FCR register. (Writing a '1' to the bit clears this flag.)
14	RXFNEIF	Receive FIFO not empty interrupt flag
	0	The receive FIFO note empty event has not occurred.
	1	The receive FIFO note empty event has occurred (Writing a '1' to the bit clears this flag)
13	RXFFIF	Receive FIFO full interrupt flag
	0	The receive FIFO full event has not occurred
	1	The receive FIFO full event has occurred (Writing a '1' to the bit clears this flag)
10	DTXBUSY	DMA transmit busy check. (DTXBUSY Indicates whether or not the DMA transmit data is busy)
	0	Not busy
	1	Busy
9	DRXBUSY	DMA receive busy check.

		(DRXBUSY Indicates whether or not the DMA receive data is busy)
		0 Not busy
		1 Busy
6	TEMT	Indicates the transmit registers are empty. TEMT indicates whether or not the transmit registers are empty. When both UARTn_THR and transmit shift register are empty, the bit is set to 1. Once either UARTn_THR or transmit shift register holds any data, the bit is cleared to 0.
		0 Data is being transmitted through the transmit registers.
		1 The transmit registers are empty.
5	THRE	Indicates the transmit data hold register is empty. THRE indicates whether or not UARTn_THR is empty, which means the UART is ready to receive new data from the microcontroller core for transmission. UARTn_THR empty interrupts can be generated if THREIE is set enabled in UARTn_IER. The THRE bit is set to 1 when new data can be written to UARTn_THR as the previous transmit data has been transferred from UARTn_THR to the transmit shift register. The bit is cleared to 0 when new data is written to UARTn_THR by the microcontroller core (When UARTn_THR is no longer empty).
		0 The transmit data hold register is not empty.
		1 The transmit data hold register is empty.
4	BI	Indicates a break condition has been detected. BI shows the occurrence of a break interrupt. If the incoming receive data remains in low state for longer than the time taken to receive the entire data word (i.e., The sum of the start, data, parity, and stop bits), the bit is set to 1. Once UARTn_LSR is read by the microcontroller core, the bit is cleared to 0.
		0 Normal.
		1 A break condition has been detected.
3	FE	Indicates presence of a frame error. FE Indicates whether or not a communication framing error has been detected. A framing error signifies that a received data word does not have an appropriate stop bit. The FE bit is set to 1, when the stop bit attached to the last data bit or the parity bit is detected to be 0. Once UARTn_LSR is read by the microcontroller core, the bit is cleared to 0.
		0 No frame error present.
		1 A frame error has been detected. The received data does not have appropriate stop bits.
2	PE	Indicates presence of a parity error. PE indicates whether or not a parity error has been detected. If a received data word does not meet the odd/even parity condition defined in UARTn_LCR, the bit is set to 1. Once UARTn_LSR is read by the microcontroller core, the bit is cleared to 0.
		0 No parity error present
		1 A parity error has been detected. The received data does not satisfy parity conditions.
1	OE	Indicates presence of an overrun error. OE indicates whether or not an overrun error has been detected. When a data word is transferred to UARTn_RBR before the register reads the previous data word, an error occurs and the OE bit is set to 1. Once UARTn_LSR is read by the microcontroller core, the overrun error is not detected and the bit is cleared to 0. In FIFO mode, OE occurs when receiving the 10th data after exceeding the receive FIFO depth of 8. If DR is the first FIFO overrun, OE is used as the second FIFO overrun.
		0 No overrun error present.
		1 An overrun error has been detected. Additional data has

		arrived when the UARTn_RBR is already full.	
0	DR	Indicates data reception status. DR indicates that data reception has been completed. The bit is set to 1 when a data word has completely been received and transferred to UARTn_RBR. Once the data is read by UARTn_RBR, the bit is cleared to 0. In FIFO mode, data reception is not confirmed through the DR. It is used as the first FIFO overrun status when receiving the ninth data after exceeding the Receive FIFO depth of 8. To clear the DR, read the FRBR. When reading while the DR is '1', the FIFO level decreases from 8 to 7, and the value at FIFO level 0 is read, making it a valid value.	
		0	There is no data in the receive data buffer register.
		1	Data is received and stored in the receive data buffer register.

12.5.8 UARTn_ELSR: UART n Extended Line Status Register

[Extended mode]

The UARTn_ELSR register is supported when MODE bit in UARTn_IER register is set to '1'.

In Extended mode, the UARTn_IIR Register cannot be used.

The UARTn_ELSR register is status register that shows the status of data transmission and reception. This register reports the status of data transfers between the transmitter and receiver. Through this register, users can check the status of the UART lines and set interrupts as follows: if they are set enabled in the RLSIE bit in the UARTn_IER register. Other bits enable their corresponding interrupts to be called if they are set enabled in the UARTn_IER register.

UART0_ELSR=0x4000_8014, UART1_ELSR=0x4000_8114, UART2_ELSR=0x4000_8214
UART3_ELSR=0x4000_8314, UART4_ELSR=0x4000_8414, UART5_ELSR=0x4000_8514
UART6_ELSR=0x4000_8614, UART7_ELSR=0x4000_8714, UART8_ELSR=0x4000_8814
UART9_ELSR=0x4000_8914, UART10_ELSR=0x4000_8A14
UART11_ELSR=0x4000_8B14, UART12_ELSR=0x4000_8C14

31 30 29 28 27 26 25 24	23 22 21 20 19 18 17 16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved	FRLVL[3:0]	RXFTLIF	RXFNEIF	RXFFIF	DTXIF	DRXIF	DTXBUSY	DRXBUSY	TEMTIF	THREIF	TEMT	THRE	BIIF	FEIF	PEIF	OEIF	DR
-	0x0	0	0	0	0	0	0	0	0	0	1	1	0	0	0	0	0
-	RO	RWC1	RWC1	RWC1	RWC1	RWC1	RO	RO	RWC1	RWC1	RO	RO	RWC1	RWC1	RWC1	RWC1	RO

19 16	FRLVL[3:0]	FIFO Reception level	
		0	FIFO Rx buffer empty
		1	Receive 1 data
		2	Receive 2 data
		3	Receive 3 data
		4	Receive 4 data
		5	Receive 5 data
		6	Receive 6 data
		7	Receive 7 data
		8	FIFO buffer full (Receive 8 data)
15	RXFTLIF	Receive FIFO trigger level interrupt flag It occurs when the receive trigger level matches RXFTLSEL in the FCR register.	
		0	The receive FIFO trigger level event has not occurred
		1	The receive FIFO trigger level event has occurred. (Writing a '1' to the bit clears this flag)
14	RXFNEIF	Receive FIFO not empty interrupt flag	
		0	The receive FIFO note empty event has not occurred.
		1	The receive FIFO note empty event has occurred (Writing a '1' to the bit clears this flag)
13	RXFFIF	Receive FIFO full interrupt flag	
		0	The receive FIFO full event has not occurred
		1	The receive FIFO full event has occurred (Writing a '1' to the bit clears this flag)
12	DTXIF	DMA Transmit complete interrupt flag	
		0	The DMA transmit data has not completed.
		1	The DMA transmit data has completed. (Writing a '1' to the bit clears this flag)

11	DRXIF	DMA Receive complete interrupt flag
	0	The DMA receive data has not completed.
	1	The DMA receive data has completed. (Writing a '1' to the bit clears this flag)
10	DTXBUSY	DMA transmit busy check. (DTXBUSY Indicates whether or not the DMA transmit data is busy)
	0	Not busy
	1	Busy
9	DRXBUSY	DMA receive busy check. (DRXBUSY Indicates whether or not the DMA receive data is busy)
	0	Not busy
	1	Busy
8	TEMTIF [1)]	The transmit registers empty interrupt flag. TEMTIF Indicates whether or not the transmit shift register is empty. TEMT = 1 indicates that UARTn_THR and transmit shift register is currently empty and thus new data can be written to the UARTn_THR register.
	0	Data is being transmitted through the transmit registers.
	1	The transmit registers are empty. (Writing a '1' to the bit clears this flag)
7	THREIF [1)]	The transmit data hold register empty interrupt flag.
	0	The transmit data hold register is not empty.
	1	The transmit data hold register is empty. (Writing a '1' to the bit clears this flag)
6	TEMT	Indicates the transmit registers are empty status TEMT indicates whether or not the transmit registers are empty. When both UARTn_THR and transmit shift register are empty, the bit is set to 1. Once either UARTn_THR or transmit shift register holds any data, the bit is cleared to 0.
	0	Data is being transmitted through the transmit registers.
	1	The transmit registers are empty.
5	THRE	Indicates the transmit data hold register is empty status THRE indicates whether or not UARTn_THR is empty, which means the UART is ready to receive new data from the microcontroller core for transmission. UARTn_THR empty interrupts can be generated if THREIE is set enabled in UARTn_IER. The THRE bit is set to 1 when new data can be written to UARTn_THR as the previous transmit data has been transferred from UARTn_THR to the transmit shift register. The bit is cleared to 0 when new data is written to UARTn_THR by the microcontroller core (When UARTn_THR is no longer empty).
	0	The transmit data hold register is not empty.
	1	The transmit data hold register is empty.
4	BIIF	Indicates a break condition has been detected. BI shows the occurrence of a break interrupt. If the incoming receive data remains in low state for longer than the time taken to receive the entire data word (i.e., The sum of the start, data, parity, and stop bits), the bit is set to 1.
	0	Normal.
	1	A break condition has been detected. (Writing a '1' to the bit clears this flag)
3	FEIF	Indicates presence of a frame error. FE Indicates whether or not a communication framing error has been detected. A framing error signifies that a received data word does not have an appropriate stop bit. The FE bit is set to 1, when the stop bit attached to the last data bit or the parity bit is detected to be 0.
	0	No frame error present.
	1	A frame error has been detected. The received data does not have appropriate stop bits.

		(Writing a '1' to the bit clears this flag)
2	PEIF	Indicates presence of a parity error. PE indicates whether or not a parity error has been detected. If a received data word does not meet the odd/even parity condition defined in UARTn_LCR, the bit is set to 1.
	0	No parity error present
	1	A parity error has been detected. The received data does not satisfy parity conditions. (Writing a '1' to the bit clears this flag)
1	OEIF	Indicates presence of an overrun error. OE indicates whether or not an overrun error has been detected. When a data word is transferred to UARTn_RBR before the register reads the previous data word, an error occurs and the OE bit is set to 1. In FIFO mode, OEIF occurs when receiving the 10th data after exceeding the receive FIFO depth of 8. If the DR is the first FIFO overrun, the OEIF is used as the second FIFO overrun.
	0	No overrun error present.
	1	An overrun error has been detected. Additional data has arrived when the UARTn_RBR is already full. (Writing a '1' to the bit clears this flag)
0	DR	Indicates data reception status. DR indicates that data reception has been completed. The bit is set to 1 when a data word has completely been received and transferred to UARTn_RBR. Once the data is read by UARTn_RBR, the bit is cleared to 0. In FIFO mode, data reception is not confirmed through the DR. It is used as the first FIFO overrun status when receiving the ninth data after exceeding the Receive FIFO depth of 8. To clear the DR, read the FRBR. When reading while the DR is '1,' the FIFO level decreases from 8 to 7, and the value at FIFO level 0 is read, making it a valid value.
	0	There is no data in the receive data buffer register.
	1	Data is received and stored in the receive data buffer register.

NOTE:

1. TEMTIF and THREIF recommend writing to and clearing the UARTn_THR register. These two bits serve as flags for events, and clearing the flags without writing data can cause TEMTIF and THREIF to be set to '0' even when the THR is empty.

12.5.9 UARTn_BDR: UART n Baud-Rate Divisor Latch Register

The UARTn_BDR register is a 16-bit register.

UART0_BDR=0x4000_8020, UART1_BDR=0x4000_8120, UART2_BDR=0x4000_8220
UART3_BDR=0x4000_8320, UART4_BDR=0x4000_8420, UART5_BDR=0x4000_8520
UART6_BDR=0x4000_8620, UART7_BDR=0x4000_8720, UART8_BDR=0x4000_8820
UART9_BDR=0x4000_8920, UART10_BDR=0x4000_8A20
UART11_BDR=0x4000_8B20, UART12_BDR=0x4000_8C20

31 30 29 28 27 26 25 24	23 22 21 20 19 18 17 16	15 14 13 12 11 10 9 8	7 6 5 4 3 2 1 0
Reserved		BDR[15:0]	
-		0x0000	
-		RW	

15 0	BDR[15:0](1)	Baud-rate divisor latch (1 to 65535)

NOTE:

1. If the UARTn_BDR value is equal to or less than two, the UARTn_BFR is always zero regardless of the UARTn_BFR set value.

12.5.10 UARTn_BFR: UART n Baud-Rate Fraction Counter Register

The UARTn_BFR register is an 8-bit register.

UART0_BFR=0x4000_8024, UART1_BFR=0x4000_8124, UART2_BFR=0x4000_8224
UART3_BFR=0x4000_8324, UART4_BFR=0x4000_8424, UART5_BFR=0x4000_8524
UART6_BFR=0x4000_8624, UART7_BFR=0x4000_8724, UART8_BFR=0x4000_8824
UART9_BFR=0x4000_8924, UART10_BFR=0x4000_8A24
UART11_BFR=0x4000_8B24, UART12_BFR=0x4000_8C24

31 30 29 28 27 26 25 24	23 22 21 20 19 18 17 16	15 14 13 12 11 10 9 8	7 6 5 4 3 2 1 0
Reserved			BFR[7:0]
-			0x00
-			RW

7 0	BFR[7:0](1)	Enable the fraction counter to compensate error from the integer baud-rate divisor	
		0	Disables the fraction counter.
		other	Enables the fraction counter. In fractional compensation mode, the value of the fraction counter is added to the integer baud-rate divisor.

NOTE:

1. If the UARTn_BDR value is equal to or less than two, the UARTn_BFR is always zero regardless of the UARTn_BFR set value.

12.5.11 UARTn_IDTR: UART n Inter-Frame Delay Time Register

The UARTn_IDTR register is an 8-bit register. A dummy delay is inserted between two consecutive transmit data frames.

UART0_IDTR=0x4000_8030, UART1_IDTR=0x4000_8130, UART2_IDTR=0x4000_8230
UART3_IDTR=0x4000_8330, UART4_IDTR=0x4000_8430, UART5_IDTR=0x4000_8530
UART6_IDTR=0x4000_8630, UART7_IDTR=0x4000_8730, UART8_IDTR=0x4000_8830
UART9_IDTR=0x4000_8930, UART10_IDTR=0x4000_8A30
UART11_IDTR=0x4000_8B30, UART12_IDTR=0x4000_8C30

31 30 29 28 27 26 25 24	23 22 21 20 19 18 17 16	15 14 13 12 11 10 9 8	7	6	5 4 3	2 1 0
Reserved			SMS	DMS	Reserved	WAITVAL[2:0]
-			0	0	-	000
-			RW	RW	-	RW

7	SMS[1]	Enable multi-sampling the start bit If SMS bit is set to 1, the start bit is sampled three times. The value indicated by the majority of the samples is taken. 0 Disables multi-sampling the start bit. At this setting, sampling is conducted only once at clock pulse 8 (Among the 16 pulses in total). 1 Enables multi-sampling the start bit. At this setting, sampling is conducted three times at clock pulses 7, 8, and 9 (Among the 16 pulses in total). The value indicated by a majority of the three samples is taken.
6	DMS	Enable multi-sampling each data bit If DMS bit is set to 1, each data bit is sampled three times. The value indicated by the majority of the samples is taken. 0 Disables multi-sampling each data bit. At this setting, sampling is conducted only once at clock pulse 9 (Among the 16 pulses in total). 1 Enables multi-sampling each data bit. At this setting, sampling is conducted three times at clock pulses 8, 9, and 10 (among the 16 pulses in total). The value indicated by a majority of the three samples is taken.
3 0	WAITVAL[2:0]	Defines the wait time for the next data frame. [unit: 1-bit period] **Wait Time = WAITVAL[2:0] / Baud-rate** WAITVAL[2:0] Based on the value (0 through 7) indicated by the WAITVAL[2:0] bits, a wait time is set between successive data transfers. The formula above is used to compute the wait time.

NOTE:

1. After the microcontroller's power-on, if the UART bit in the SCU_PER2 register is disabled and enabled again, the SMS bit is set to '1'.

12.5.12 UARTn_FCR: UART n FIFO Control Register

The UARTn_FCR register is a 32-bit read/write accessible register. It sets the operating modes and enablement of the UART FIFO receive module.

UART0_FCR=0x4000_8040, UART1_FCR=0x4000_8140, UART2_FCR=0x4000_8240
UART3_FCR=0x4000_8340, UART4_FCR=0x4000_8440, UART5_FCR=0x4000_8540
UART6_FCR=0x4000_8640, UART7_FCR=0x4000_8740, UART8_FCR=0x4000_8840
UART9_FCR=0x4000_8940, UART10_FCR=0x4000_8A40
UART11_FCR=0x4000_8B40, UART12_FCR=0x4000_8C40

31 30 29 28 27 26 25 24	23 22 21 20 19 18 17 16	15 14 13 12 11 10 9 8	7	6	5	4	3 2	1	0
Reserved		RXFTLVL [3:0]	Reserved	RXFTLIE	RXFNEIE	RXFFIE	Reserved	RXFBCLR	RXFEN
-		0x4	-	0	0	0	-	0	0
-		RW	-	RW	RW	RW	-	WO	RW

11 8	RXFTLVL[3:0]	Receive FIFO trigger level selection bit	
		0000	Receive 0 data Trigger (Rx empty)
		0001	Receive 1 data Trigger
		0010	Receive 2 data Trigger
		0011	Receive 3 data Trigger
		0100	Receive 4 data Trigger
		0101	Receive 5 data Trigger
		0110	Receive 6 data Trigger
		0111	Receive 7 data Trigger
		1000	Receive 7 data Trigger (Similar RXFFIE)
6	RXFTLIE	Enable or disable the Receive FIFO trigger level interrupt	
		0	Disables the Rx FIFO trigger level interrupt
		1	Enables the Rx FIFO trigger level interrupt
5	RXFNEIE	Enable or disable the Receive FIFO not empty interrupt.	
		0	Disables the Rx FIFO not empty interrupt
		1	Enables the Rx FIFO not empty interrupt
4	RXFFIE	Enable or disable the Receive FIFO full interrupt	
		0	Disables the Rx FIFO full interrupt
		1	Enables the Rx FIFO full interrupt
1	RXFBCLR	Receive FIFO buffer clear bit. (auto-clear)	
		0	No action
		1	Clear receive FIFO buffer
0	RXFEN	Enable the Receive FIFO	
		0	Disable the receive FIFO
		1	Enable the receive FIFO

12.5.13 UARTn_FRBR: UART n FIFO Receive Buffer Register

The UARTn_FRBR register is supported when RXFEN bit in UARTn_FCR register is set to '1'.

UARTn_FRBR is an 8-bit read-only register. Received data is read from this register, and the maximum length of data is 8 bits. The last data received is retained until a new byte is received.

FIFO receive buffer depth is 8.

UART0_FRBR =0x4000_8044, UART1_FRBR=0x4000_8144, UART2_FRBR=0x4000_8244
UART3_FRBR =0x4000_8344, UART4_FRBR=0x4000_8444, UART5_FRBR=0x4000_8544
UART6_FRBR=0x4000_8644, UART7_FRBR=0x4000_8744, UART8_FRBR=0x4000_8844
UART9_FRBR=0x4000_8944, UART10_FRBR=0x4000_8A44
UART11_FRBR=0x4000_8B44, UART12_FRBR=0x4000_8C44

31 30 29 28 27 26 25 24	23 22 21 20 19 18 17 16	15 14 13 12 11 10 9 8	7 6 5 4 3 2 1 0
Reserved			FRBR[7:0]
-			0x00
-			R0

7 0	FRBR[7:0]	FIFO receive data register

12.5.14 UART Register Map Summary

Table 118. UART Register Map Summary

Offset	Register name	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0x00	UARTn_RBR	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	RBR[7:0]							
	Reset value																									0	0	0	0	0	0	0	0
0x00	UARTn_THR	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	THR[7:0]							
	Reset value																									-	-	-	-	-	-	-	-
0x04	UARTn_IER	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	MODE	UMKEY				Res.	Res.	Res.	Res.	Res.	DTXIEN	DRXIEN	TXEIE	RLSIE	THREIE	DRIE
	Reset value																	0	0	0	0							0	0	0	0	0	0
0x08	UARTn_IIR	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	RXFTLSTS	RXFNESTS	RXFFSTS									TXE	IID[2:0]			IPEN
	Reset value																	0	0	0									0	0	0	0	1
0x0C	UARTn_LCR	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.										BREAK	STICKP	PARITY	PEN	STOPBIT	DLEN[1:0]	
	Reset value																										0	0	0	0	0	0	0
0x10	UARTn_DCR	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.													RXINV	TXINV	Res.	Res.
	Reset value																													0	0		
0x14	UARTn_LSR [Legacy mode]	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	FRLVL[3:0]				RXFTLIF	RXFNEIF	RXFFIF			DTXBUSY	DRXBUSY			TEMT	THRE	BI	FE	PE	OE	DR
	Reset value													0	0	0	0	0	0	0			0	0			1	1	0	0	0	0	0
0x14	UARTn_ELSR [Extended mode]	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	FRLVL[3:0]				RXFTLIF	RXFNEIF	RXFFIF	DTXIF	DRXIF	DTXBUSY	DRXBUSY	TEMTIF	THREIF	TEMT	THRE	BIIF	FEIF	PEIF	OEIF	DR
	Reset value													0	0	0	0	0	0	0	0	0	0	0	0	0	1	1	0	0	0	0	0

Table 118. UART Register Map Summary (continued)

Offset	Register name	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0x20	UARTn_BDR	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	BDR[15:0]															
	Reset value																	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
0x24	UARTn_BFR	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	BFR[7:0]							
	Reset value																									0	0	0	0	0	0	0	0
0x30	UARTn_IDTR	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	SMS	DMS	Res.	Res.	Res.	WAITVAL[2:0]		
	Reset value																									0	0				0	0	0
0x40	UARTn_FCR	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	RXFTLVL[3:0]				Res.	RXFTLIE	RXFNEIE	RXFFIE	Res.	Res.	RXFBCLR	RXFEN
	Reset value																					0	1	0	0		0	0	0			0	0
0x44	UARTn_FRBR	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	FRBR[16:0]																
	Reset value																0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

13. Serial Peripheral Interface (SPI)

13.1 SPI Introduction

The A34G43x series has twelve built-in Serial Peripheral Interface (SPI) modules. The SPI modules are synchronized by clocks, and the specifications of the transfer clocks are adjustable.

The SPI module supports communication between one master and slaves. Slaves can be selected by the Slave Select (SS) signal.

The SPI module performs four-wire synchronous transfers via four signal terminals (SS, SCK, MOSI, and MISO). Its separate Transmit and Receive Buffers enable full-duplex communication, which is capable of reading and writing data simultaneously.

Specifically, the SPI module of the A34G43x series can enhance the port drive strength so that the SPI rate can be supported up to 12 MHz with enhanced port drive strength.

- Using receive FIFO, users can perform high-speed receive data.

13.2 SPI Main Features

The SPI of the A34G43x features the following:

- Selectable between the master and slave operations
- Full-duplex and four-wire synchronous transfers
  - SS: Slave Select
  - SCK: Serial Clock
  - MOSI: Master Output, Slave Input
  - MISO: Master Input, Slave Output
- SPI clock speed and polarity adjustable
- Separate transmit and receive data registers with different data transfer sizes
  - Available transmit/receive data sizes: 8-, 9-, 16-, and 17-bit
- Configurable interrupts triggered by the transmit status and SS signal
- Loop-back mode for internal checkups
- User-programmable start, burst, and stop delay settings
- Up to 12 MHz of SPI clock supported
- Port drive strength for high-speed communication
- Selectable between MSB first transfer or LSB first transfer
- DMA Tx, Rx transfers
- FIFO Tx, Rx transfers

13.3 SPI Implementation

The transmitter and receiver of the Serial Peripheral Interface (SPI) share the same clock but are independent of each other. This enables full-duplex transfers. The transmitter and receiver are equipped with double buffers, thereby supporting back-to-back data transfers either by reading previous receive data from the SPI_RDR register while subsequent data is being received, or by writing subsequent data to the SPI_TDR register while previous data is being transmitted.

To enable the SPI transmission and reception, the MOSI and MISO lines of the master and slave must be connected directly, enabling back-to-back transfers between them. The most significant bits are given priority during these data transfers. Communication is always commenced by the master. Once the master transfers data to the slave through the MOSI pin, the slave responds through the MISO pin.

Table 119 describes the built-in SPI features.

Table 119. SPI Implementation

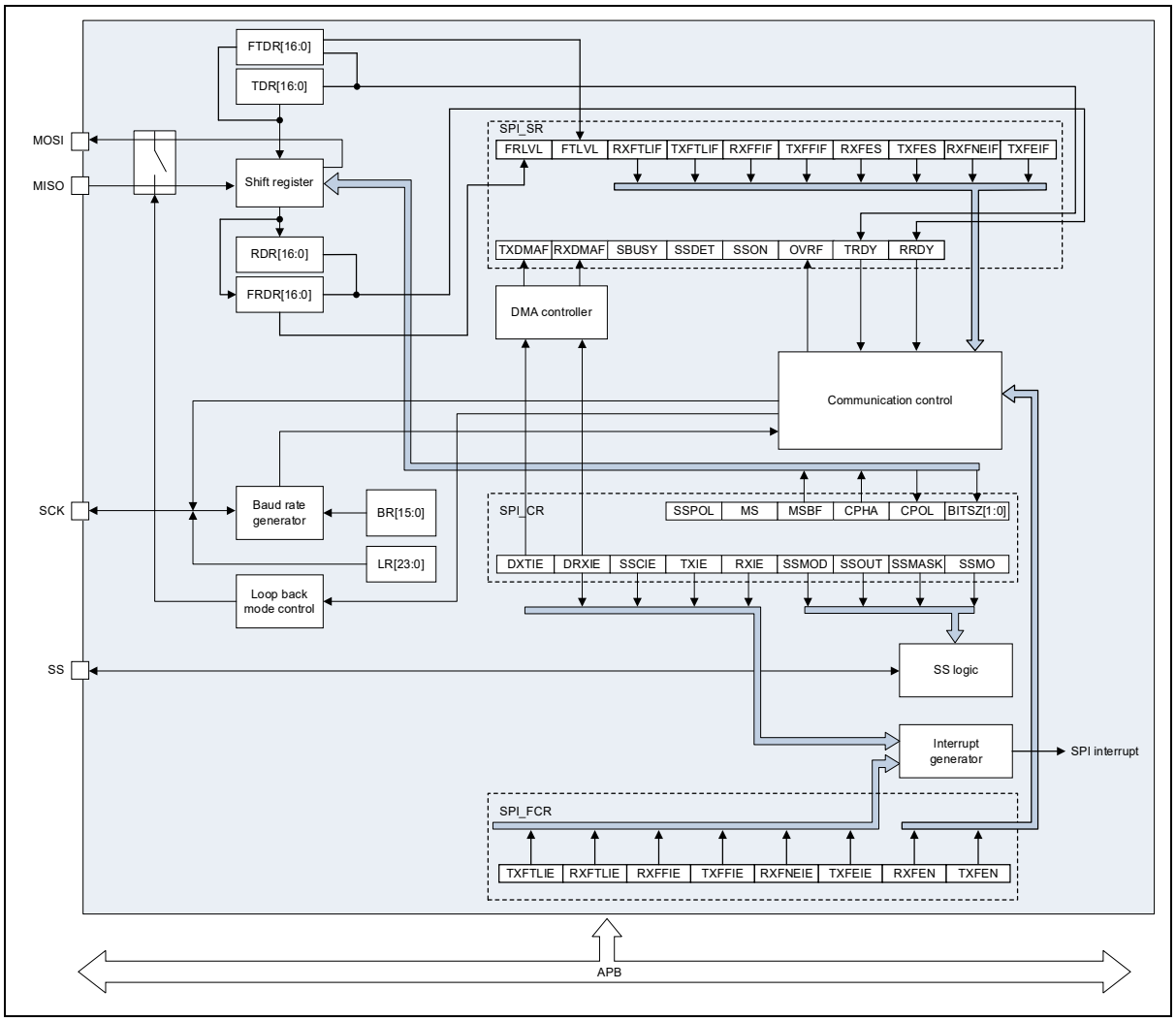
SPI Feature	Description
DMA	DMA transfer for receive and transmit
Data size	Data size selection for transmit and receive (8-, 9-, 16-, 17-bit)
MSB/LSB selection	Transfer format selection (MSB or LSB)
Error flag	Error flags (Overrun)
Latency	Programmable latency (start, burst, and stop delay)
FIFO	FIFO transfer and receive depth is 8

13.4 SPI Functional Description

13.4.1 SPI Block Diagram

Figure 106 shows the block diagram for the SPI of the A34G43x.

Figure 106. SPI Block Diagram



13.4.2 SPI Pins and Internal Signals

Table 120 describes the pins assigned for the SPI.

Table 120. Pin Assignment of SPI: External Pins

Pin Name	Type	Description	Supported Packages		
			A34G43xYL (120-LQFP)	A34G43xVL (100-LQFP)	A34G43xRL (64-LQFP)
SS0	I/O	SPI0 serial port I/O signal for slave selection	O	O	O
SCK0	I/O	SPI0 clock I/O (Master: output/ slave: input)	O	O	O
MOSI0	I/O	SPI0 transmit and receive data (Master: output/ slave: input)	O	O	O
MISO0	I/O	SPI0 transmit and receive data (Master: input/ slave: output)	O	O	O
SS1	I/O	SPI1 serial port I/O signal for slave selection	O	O	N/A
SCK1	I/O	SPI1 clock I/O (Master: output/ slave: input)	O	O	N/A
MOSI1	I/O	SPI1 transmit and receive data (Master: output/ slave: input)	O	O	N/A
MISO1	I/O	SPI1 transmit and receive data (Master: input/ slave: output)	O	O	N/A
SS2	I/O	SPI2 serial port I/O signal for slave selection	O	O	O
SCK2	I/O	SPI2 clock I/O (Master: output/ slave: input)	O	O	O
MOSI2	I/O	SPI2 transmit and receive data (Master: output/ slave: input)	O	O	O
MISO2	I/O	SPI2 transmit and receive data (Master: input/ slave: output)	O	O	O
SS3	I/O	SPI3 serial port I/O signal for slave selection	O	O	O
SCK3	I/O	SPI3 clock I/O (Master: output/ slave: input)	O	O	O
MOSI3	I/O	SPI3 transmit and receive data (Master: output/ slave: input)	O	O	O
MISO3	I/O	SPI3 transmit and receive data (Master: input/ slave: output)	O	O	O
SS4	I/O	SPI4 serial port I/O signal for slave selection	O	O	N/A
SCK4	I/O	SPI4 clock I/O (Master: output/ slave: input)	O	O	N/A
MOSI4	I/O	SPI4 transmit and receive data (Master: output/ slave: input)	O	O	N/A
MISO4	I/O	SPI4 transmit and receive data (Master: input/ slave: output)	O	O	N/A
SS5	I/O	SPI5 serial port I/O signal for slave selection	O	O	N/A
SCK5	I/O	SPI5 clock I/O (Master: output/ slave: input)	O	O	N/A
MOSI5	I/O	SPI5 transmit and receive data (Master: output/ slave: input)	O	O	N/A
MISO5	I/O	SPI5 transmit and receive data (Master: input/ slave: output)	O	O	N/A
SS6	I/O	SPI6 serial port I/O signal for slave selection	O	O	N/A

Table 120. Pin Assignment of SPI: External Pins (continued)

Pin Name	Type	Description	Supported Packages		
			A34G43xYL (120-LQFP)	A34G43xVL (100-LQFP)	A34G43xRL (64-LQFP)
SCK6	I/O	SPI6 clock I/O (Master: output/ slave: input)	O	O	N/A
MOSI6	I/O	SPI6 transmit and receive data (Master: output/ slave: input)	O	O	N/A
MISO6	I/O	SPI6 transmit and receive data (Master: input/ slave: output)	O	O	N/A
SS7	I/O	SPI7 serial port I/O signal for slave selection	O	O	O
SCK7	I/O	SPI7 clock I/O (Master: output/ slave: input)	O	O	O
MOSI7	I/O	SPI7 transmit and receive data (Master: output/ slave: input)	O	O	O
MISO7	I/O	SPI7 transmit and receive data (Master: input/ slave: output)	O	O	O
SS8	I/O	SPI8 serial port I/O signal for slave selection	O	O	N/A
SCK8	I/O	SPI8 clock I/O (Master: output/ slave: input)	O	O	N/A
MOSI8	I/O	SPI8 transmit and receive data (Master: output/ slave: input)	O	O	N/A
MISO8	I/O	SPI8 transmit and receive data (Master: input/ slave: output)	O	O	N/A
SS9	I/O	SPI9 serial port I/O signal for slave selection	O	O	N/A
SCK9	I/O	SPI9 clock I/O (Master: output/ slave: input)	O	O	N/A
MOSI9	I/O	SPI9 transmit and receive data (Master: output/ slave: input)	O	N/A	N/A
MISO9	I/O	SPI9 transmit and receive data (Master: input/ slave: output)	O	N/A	N/A
SS10	I/O	SPI10 serial port I/O signal for slave selection	O	N/A	N/A
SCK10	I/O	SPI10 clock I/O (Master: output/ slave: input)	O	N/A	N/A
MOSI10	I/O	SPI10 transmit and receive data (Master: output/ slave: input)	O	N/A	N/A
MISO10	I/O	SPI10 transmit and receive data (Master: input/ slave: output)	O	N/A	N/A
SS11	I/O	SPI11 serial port I/O signal for slave selection	O	N/A	N/A
SCK11	I/O	SPI11 clock I/O (Master: output/ slave: input)	O	N/A	N/A
MOSI11	I/O	SPI11 transmit and receive data (Master: output/ slave: input)	O	N/A	N/A
MISO11	I/O	SPI11 transmit and receive data (Master: input/ slave: output)	O	N/A	N/A
SS12	I/O	SPI12 serial port I/O signal for slave selection	O	O	N/A
SCK12	I/O	SPI12 clock I/O (Master: output/ slave: input)	O	O	N/A
MOSI12	I/O	SPI12 transmit and receive data (Master: output/ slave: input)	O	O	N/A
MISO12	I/O	SPI12 transmit and receive data (Master: input/ slave: output)	O	O	N/A

13.4.3 Communications between Master and Slave

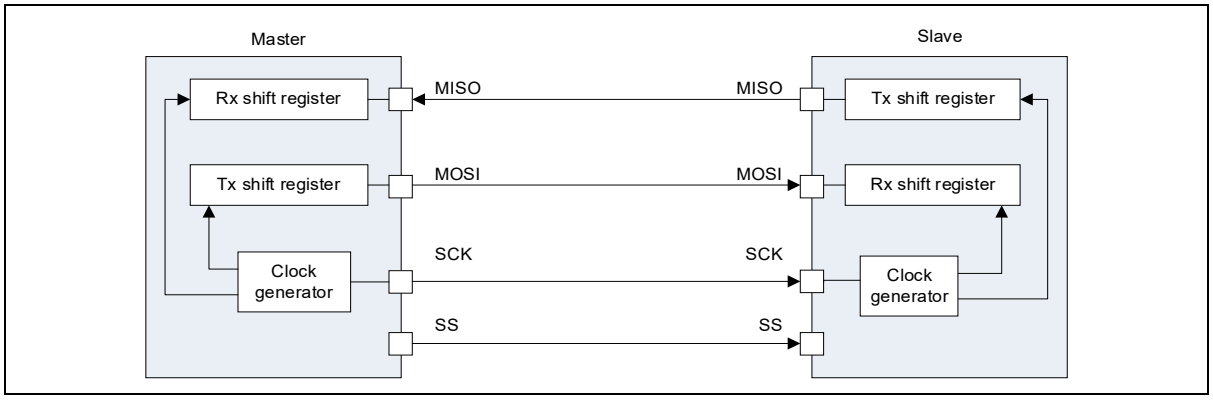
13.4.3.1 Full-Duplex Communication

Figure 107 shows a normal connection for the communication between the SPI-Master and a Slave. For full-duplex communication, the SPI- Master and slave are connected as shown in Figure 107.

The SPI- Master generates the SCK clock for transmission, and outputs data on the MOSI pin. At the same time, the Slave transmits data through the MISO pin according to the Master's SCK.

The SS pin outputs Low for the Slave Chip Select when the Master starts transmitting.

Figure 107. Block Diagram of SPI Master to Slave Connection: Full-Duplex



13.4.3.2 Simplex Communication

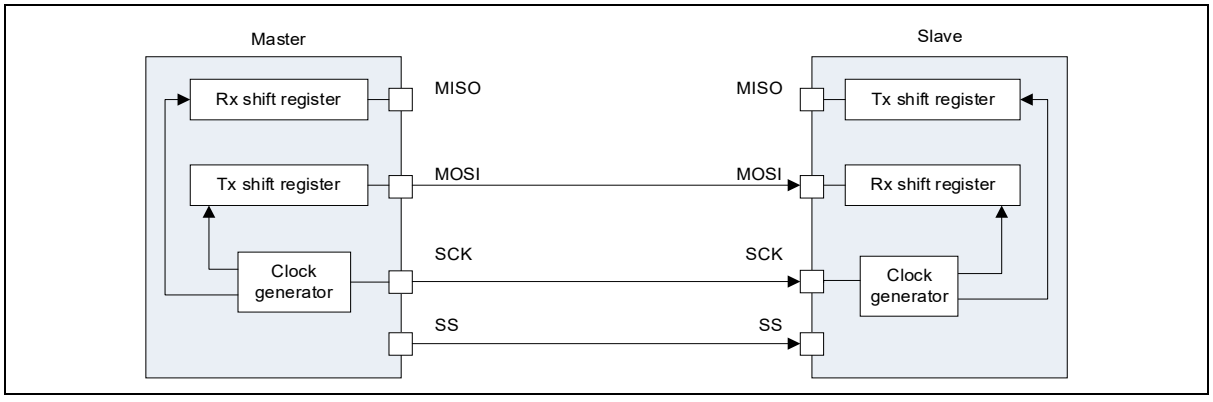
If only the Master's transmission and the Slave's reception are required, the Master and Slave need to be connected as shown in Figure 108.

The SPI- Master generates the SCK clock for transmission and outputs data on the MOSI pin. At the same time, the Slave receives data on the MOSI pin according to the Master's SCK.

The SS pin outputs Low for the Slave Chip Select when the Master starts transmitting.

The MISO pin can be used as an alternate function pin such as a GPIO pin.

Figure 108. Block Diagram of SPI Master to Slave Connection: Simplex



13.4.4 Slave Selection Pin

The SS line is used for slave select input that enables the slave to communicate with the master. The SS pin can be driven as a standard IO port of the master device and is configured by setting the SSMOD in SPI_CR register.

If the SSMOD is set to ‘1’, the pin is driven internally according to the settings of the SSOUT in SPI_CR register. If the SSMOD bit is set to ‘0’, the pin performs one of the two functions depending on the settings of the SSMO in SPI_CR register.

When SSMO = ‘1’ and SSMOD = ‘0’, the SS pin is used only in Master mode. In this case, the SS signal is driven to Low when the Master starts communication. This low state is maintained until the SPI becomes disabled.

When SSMO = ‘0’ and SSMOD = ‘0’, this setting is used as Master mode. In Slave mode, the SS pin functions as an input pin. When the SS signal is Low, the device is selected as a Slave; when it is driven High, the Slave selection is released.

For the device in Master mode, therefore, users can either set the SS pin as an output pin to deactivate the pin (SSMOD = ‘0’ , SSMO = ‘0’) or set the pin as an input pin, feed it with a high-level signal (SSMOD = ‘1’, SSOUT = ‘1’), and, after a period of waiting time, activate the pin (SSMOD = ‘0’, SSMO = ‘1’) or feed the pin with a low-level signal (SSMOD = ‘1’, SSOUT = ‘0’) to generate a falling edge.

13.4.5 Communication Formats

13.4.5.1 Clock Phase and Polarity Controls

Each SPI has four operating modes that synchronize data transferred through the MOSI and MISO lines with the SCK clock. There are four mode SPI transfer formats which the CPHA and CPOL bits in the SPIn_CR register set the phase and polarity of the SPI clock.

The CPHA (Clock phase control) bit is used to select a transfer format among two different types. When the bit is set to ‘0’, data is read at the first clock edge, which is an odd-numbered edge; when it is set to ‘1’, data is read at the second clock edge, which is an even-numbered edge.

The CPOL (Clock polarity control) bit is used to set the default level of the clock signal as active-high or active-low. This does not significantly affect the transfer format. Switching the bit causes the clock signal to be inverted (e.g., Active-high is inverted to active-low; idle-low is inverted to idle-high).

To ensure appropriate communication between the master and slave devices, the two devices must be set in the same operating mode. Therefore, it might be needed to reconfigure the master device to fit the requirements of peripheral slaves.

Figure 109 and Figure 110 illustrate SPI transfer timings when the CPHA is set to ‘0’.

When the CPHA ‘0’, the Master and Slave devices can read data at an odd-numbered (First) clock edge and change data at an even-numbered (Second) clock edge.

The CPOL is used to set the default level of the SCK clock signal. If the CPOL = ‘0’, the default level is set Low; if the CPOL = ‘1’, the default level is set High.

The MSBF bit is used to select among the MSB- or LSB-first transfer modes for output from the MISO line. If the MSBF = ‘1’, data is shifted out bit by bit from the Most Significant Bit down to the Least Significant Bit; if the MSBF = ‘0’, data is shifted out bit by bit from the Least Significant Bit down to the Most Significant Bit.

Once all data has been transferred and the SS pin is set to ‘1’, the SCK clock signal is no longer generated and the MISO and MOSI lines become High or Low, depending on the last data of SPI.

Figure 109. SPI Transfer Timing 1 of 4 (CPHA = ‘0’, CPOL = ‘0’, MSBF = ‘0’)

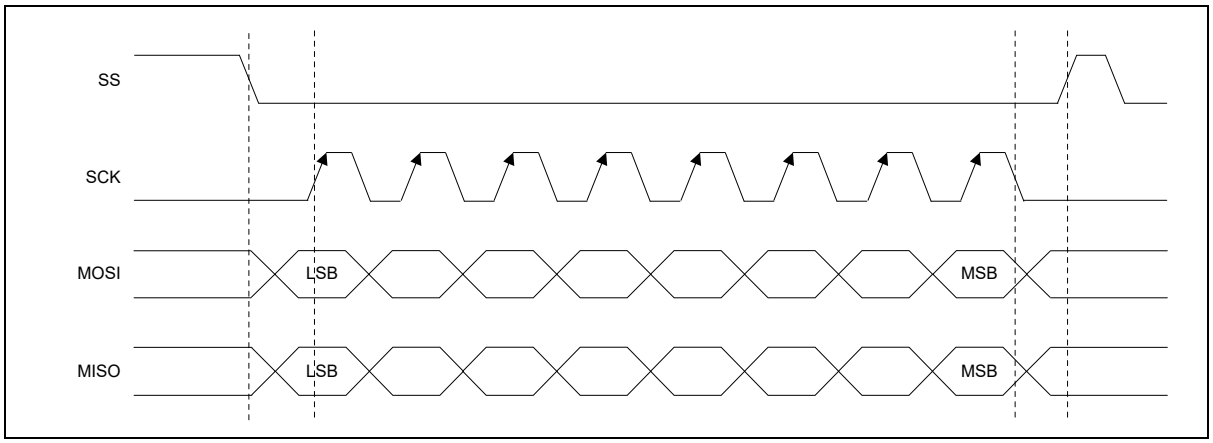


Figure 110. SPI Transfer Timing 2 of 4 (CPHA = ‘0’, CPOL = ‘1’, MSBF = ‘1’)

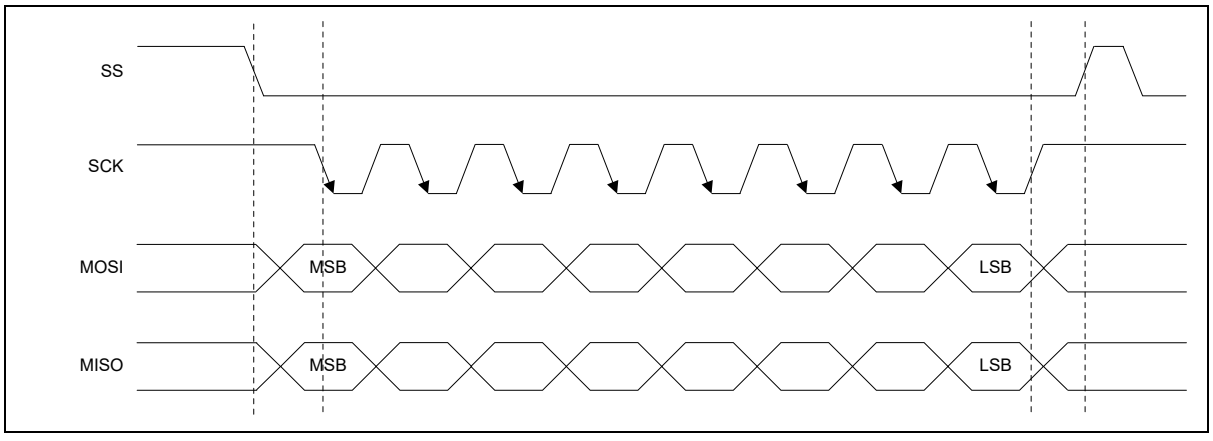


Figure 111 and Figure 112 illustrate SPI transfer timings when the CPHA is set to ‘1’.

When the CPHA = ‘1’, the master and slave devices can read data at an even-numbered (second) clock edge and change data at an odd-numbered (first) clock edge.

The CPOL is used to set the default level of the SCK clock signal. If the CPOL = ‘0’, the default level is set low; if the CPOL = ‘1’, the default level is set high.

The MSBF bit is used to select among the MSB- or LSB-first transfer modes for output from the MISO line. If the MSBF = ‘1’, data is shifted out bit by bit from the Most Significant Bit down to the Least Significant Bit; if the MSBF = ‘0’, data is shifted out bit by bit from the Least Significant Bit down to the Most Significant Bit.

Once all data has been transferred and the SS pin is set to ‘1’, the SCK clock signal is no longer generated and the MISO and MOSI lines become High or Low, depending on the last data of SPI.

Figure 111. SPI Transfer Timing 3 of 4 (CPHA = ‘1’, CPOL = ‘0’, MSBF = ‘0’)

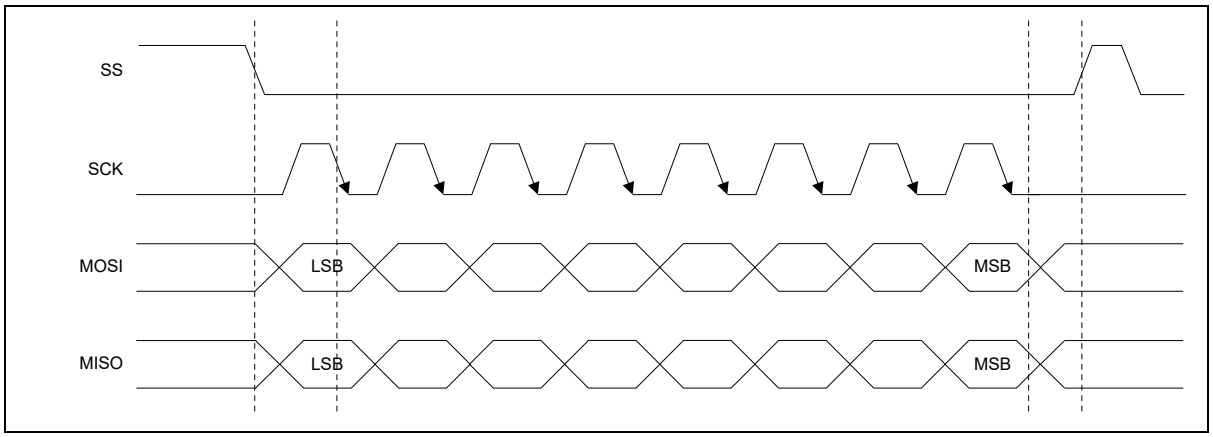
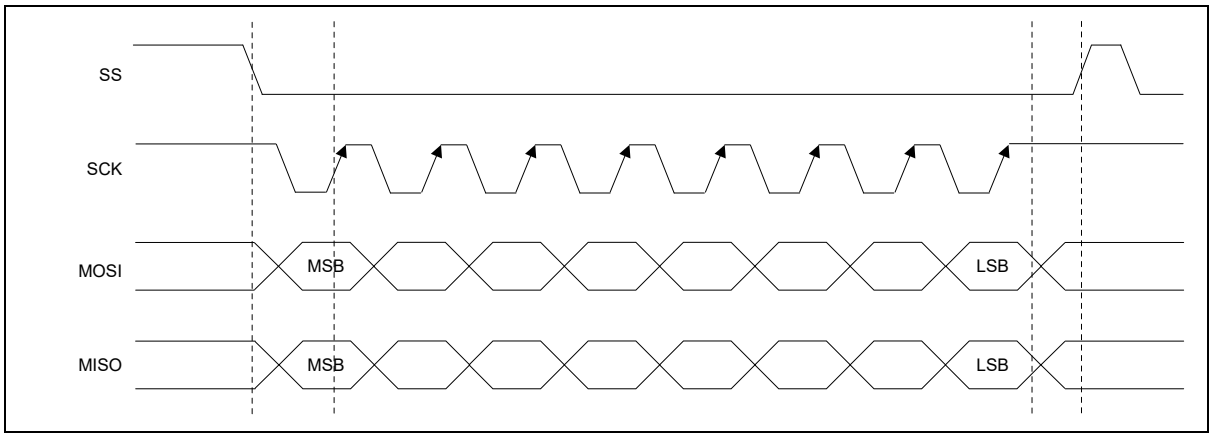


Figure 112. SPI Transfer Timing 4 of 4 (CPHA = ‘1’, CPOL = ‘1’, MSBF = ‘1’)



13.4.5.2 Data Frame Format

A data frame follows the rules below to form a format:

- When the LSB is transferred first, data is transferred starting at bit position 0.
- When the MSB is transferred first, data is transferred starting at bit position 7.

13.4.6 Configuration of SPI

Typically, users can set the initial setting of the SPI to Master mode or Slave mode by following the procedure below:

1. Set the GPIO pin to fit the alternate function. (MOSI, MISO, SCK, and SS pins)
2. Enable the SPI related bits of the SCU_PER and SCU_PRER registers.
3. Steps 4, 5, …, and 9 (Listed below) describe how to set the SPIn_CR register and can be set in any order.
4. Configure a combination of the CPOL and CPHA bits.
5. Enable the TXBC and RXBC bits of the SPIn_CR register to clear the Transmit Buffer and Receive Buffer, respectively.
6. Set the MS in SPIn_CR register depending on whether the SPI is in Master mode or Slave mode.
7. Set the MSBF in SPIn_CR register to determine whether to transmit the MSB first or the LSB first.
8. Set the BITSZ[1:0] in SPI_CR register to determine the data transfer size.
9. In Master mode, set the SSMO in SPI_CR register to ‘1’ to enable the SS output signal.
10. Write the baud-rate to the SPIn_BR register to determine the SPI communication rate.

When all of the above steps are complete, the basic initial setup is concluded. Additional setups such as SPI latency settings and information on how to use DMA functionality are described in the later sections in this chapter.

For more information on data communication, refer to section 13.4.8

13.4.7 Procedure for Enabling SPI

The SPI_EN register is used to enable or disable the SPI. When the ENABLE bit in the SPIn_EN register is set to ‘1’, the SPI is ready to transmit or receive, based on the SPI settings. The communication and clock start to operate immediately after the SPI is enabled. To process the DMA signal, the DMA-dedicated registers need to be controlled.

When the ENABLE bit is written as ‘1’, the dummy data of the Transmit Buffer will be shifted. To prevent this, data must be written to the SPIn_TDR register before the ENABLE bit is set to be enabled.

13.4.8 Data Transmission and Reception Procedures

13.4.8.1 Sequence Handling

This section describes how to transmit and receive data in master mode or slave mode, in consideration of the interrupt and polling methods.

Transmit methods in master mode are as follows:

Interrupt Method

1. When the SPI initial setup is complete, set the ENABLE bit in the SPIn_EN register to '1'.
2. Set the microcontroller core NVIC interrupts.
3. Write the first data to start the transfer to the SPIn_TDR register.
4. Set the TXIE bit in the SPIn_CR register to '1' to enable the transmit interrupt.
5. Process the following in the SPI interrupt service routine.
    A. Check that the transmit buffer is ready by checking the TRDY flag in the SPIn_SR register.
    B. If the transmit buffer is ready, write the data to the SPIn_TDR register.

Polling Method

1. When the SPI initial setup is complete, set the ENABLE bit in the SPIn_EN register to '1'.
2. Check that the transmit buffer is ready by checking the TRDY flag in the SPIn_SR register.
    A. If the transmit buffer is ready, write the data to be sent to the SPIn_TDR register.

Receive methods in master mode are as follows:

Interrupt Method

1. When the SPI initial setup is complete, set the ENABLE bit in SPIn_EN register to '1'.
2. Set the microcontroller core NVIC interrupts.
3. Write a dummy datum to start the transfer to the SPIn_TDR register.
4. Set the RXIE bit in the SPIn_CR register to '1' to enable the receive interrupt.
5. Process the following in the SPI interrupt service routine.
    A. Check that the transmit buffer is ready by checking the TRDY flag in the SPIn_SR register.
    B. If the transmit buffer is ready, write a dummy datum to the SPIn_TDR register.
    C. Check that the receive buffer is not empty by checking the RRDY flag in the SPIn_SR register.
    D. Read the SPIn_RDR register if the RRDY flag in the SPIn_SR register is '1'.

Polling Method

1. When the SPI initial setup is complete, set the ENABLE bit in SPIn_EN register to ‘1’.
2. Check that the transmit buffer is ready by checking the TRDY flag in the SPIn_SR register.
   A. If the transmit buffer is ready, write dummy data to the SPIn_TDR register.
3. Check that the receive buffer is not empty by checking the RRDY flag in the SPIn_SR register.
   A. Read the SPIn_RDR register if the RRDY flag in the SPIn_SR register is ‘1’.

Transmit and receive methods in slave mode are as follows:

Interrupt Method

1. When the SPI initial setup is complete, set the ENABLE in SPIn_EN register to ‘1’.
2. Set the microcontroller core NVIC interrupts.
3. Write the first data to be sent to the SPIn_TDR register before receiving data.
   (Because the slave device sends data simultaneously when the master device sends data, the slave device must store the data to be sent in advance.)
4. Set the RXIE bit in the SPIn_CR register to ‘1’ to enable the receive interrupt.
5. Process the following in the SPI interrupt service routine.
   A. Check that data is received from the master by checking the RRDY flag in the SPIn_SR register.
   B. If the RRDY flag in the SPIn_SR register is ‘1’, read the received data from the SPIn_RDR register and write the next data to be sent to the SPIn_TDR register.

Polling Method

1. When the SPI initial setup is complete, set the ENABLE bit in SPIn_EN register to ‘1’.
2. Write the data to be sent to the SPIn_TDR register before receiving data. (Because the slave device sends data simultaneously when the master device sends data, the slave device must store the data to be sent in advance.)
3. Check that data is received from the master by checking the RRDY flag in the SPIn_SR register.
   A. If the RRDY flag in the SPIn_SR register is ‘1’, read the received data from the SPIn_RDR register and write the data to be sent to the SPIn_TDR register.

13.4.8.2 Procedure for Disabling SPI

If the SPI is disabled, users need to follow the deactivation procedure described below. It is important that this deactivation procedure is performed before the system enters Low-power mode where the clock stops. If the deactivation procedure cannot be performed before the system enters Low-power mode, the on-going transaction may be corrupted.

The SPI deactivation procedure is as follows:

1. When the SPI communication is completed, disable the SPI Transmit Interrupt or Receive Interrupt.
2. Set the ENABLE bit in the SPIn_EN register to '0' to disable the SPI.
3. Before restoring the SPI communication, write the next transfer data to the SPIn_TDR register.

13.4.8.3 Data Packing

The SPI transmit and receive data size can be set by setting the BITSZ[1:0] bits in the SPIn_CR register, which can be set by 17-, 16-, 8-,or 7-bit.

13.4.8.4 DMA Handshake

The SPI supports the DMA handshaking operation. To operate a DMA handshake, the DMA registers should first be set. (Refer to chapter 7 for more information.)

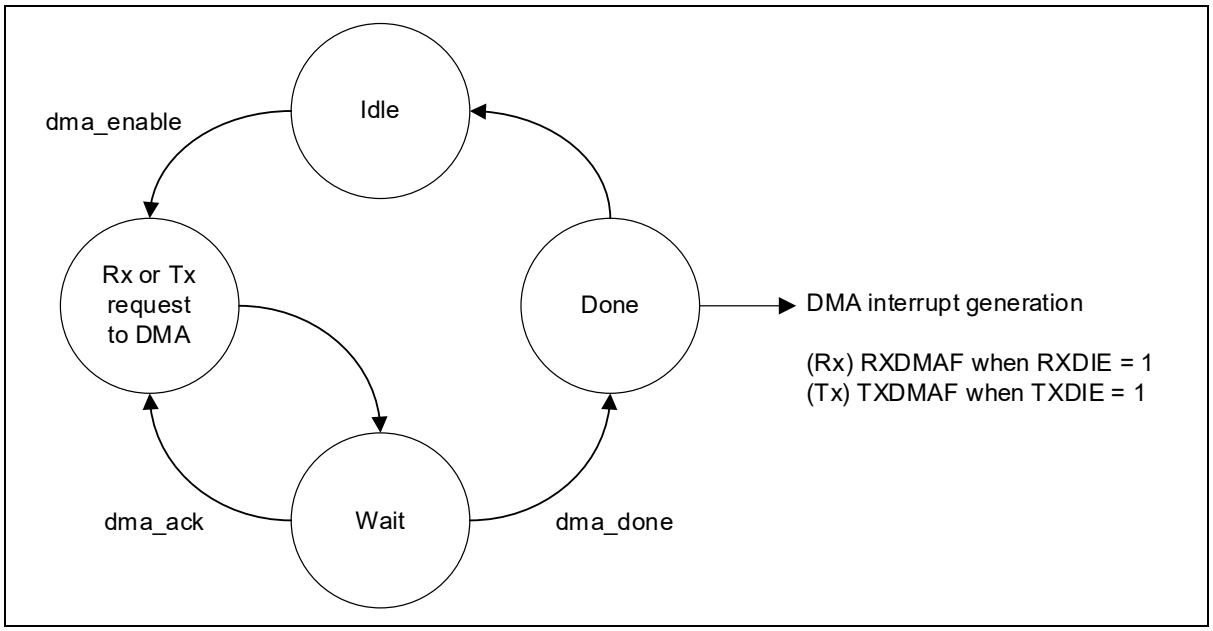
The SPI uses two DMA channels. As the transmitter and receiver are independent of each other, the SPI can operate the two channels at the same time. Once the DMA channel for the receiver is enabled and the Receive Buffer (SPIn_RDR register) is filled, the SPI sends an Rx request to the DMA to empty the buffer and waits for an acknowledge signal from the DMA.

If the Receive Buffer (SPIn_RDR register) is filled again after the acknowledge signal, the SPI sends an Rx request. If the DMA Rx DONE becomes High, the RXDMAF flag in the SPIn_SR register becomes '1' and an interrupt is serviced when the RXDIE bit in the SPIn_CR register is set.

Similarly, if the Transmit Buffer (SPIn_TDR register) is empty after the DMA channel for the transmitter is enabled, the SPI sends a Tx request to the DMA to fill the buffer and waits for an acknowledge signal from the DMA. If the Transmit Buffer is empty again after the acknowledge signal, the SPI sends a Tx request.

If the DMA Tx DONE becomes High, the TXDMAF flag in the SPIn_SR register becomes '1' and an interrupt is serviced when the TXDIE bit in the SPIn_CR register is set. The Slave Transmitter sends dummy data at the first transfer (For 8–17 SCK pulses) in DMA handshake mode.

Figure 113. DMA Handshake State Diagram



13.4.8.5 Communication Diagrams with DMA

The procedure of how to set up the SPI using DMA is shown below:

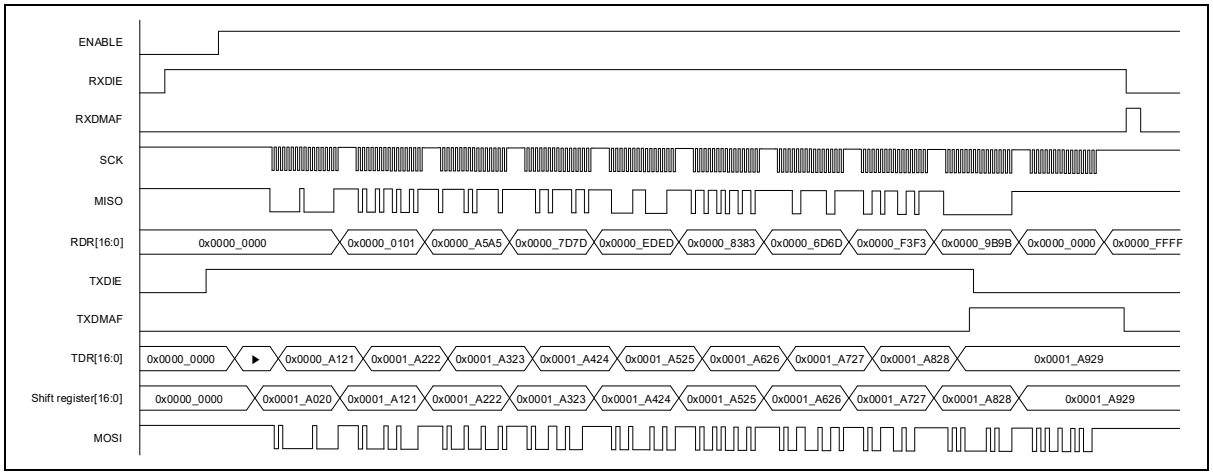
DMA TX

1. Set up the initial settings of DMA and SPI.
   A. Enable the DMAEN bit in the DMAn_SR register.
   B. Write SPI data into memory.
   C. Configure each bit of the DMAn_CR register by referring to the following.
      i. The TRANSCNT[11:0] determines the number of DMA transfers.
      ii. The PERISEL[5:0] select the SPIn_Tx.
      iii. The SIZE[1:0] select the size of a data transfer among byte, half-word, and word.
      iv. The DIR sets the direction from memory to peripheral as TX.
   D. Set the DMAn_PAR register to 0x4000_9000, the address of the SPI_TDR register.
   E. To the DMAn_MAR register, write the initial address of the memory to be written.
2. Enable TXDIE (DMA Tx Done Interrupt Enable) bit in the SPIn_CR register and SPI NVIC settings.
   A. Once the values are set in the previous steps, enable the DMAEN bit in the DMAn_SR register to send the SPI data using DMA.
   B. Once the DMA transfer is started and when value of the TRANSCNT[11:0] bits in the DMAn_CR register reaches ‘0’, the TXDMAF flag in the DMAn_SR register that indicates that the DMA transfer ends is set to ‘1’.

DMA RX

1. Set up the initial settings of DMA and SPI.
   A. Enable the DMAEN bit in the DMAn_SR register.
   B. Configure each bit of the DMAn_CR register by referring to the following.
      i. The TRANSCNT[15:0] determines the number of DMA transfers.
      ii. The PERISEL[5:0] select the SPIn_Rx.
      iii. The SIZE[1:0] select the size of a data transfer among byte, half-word, and word.
      iv. The DIR sets the direction from peripheral to memory as RX.
   C. Set the DMAn_PAR register to 0x4000_9000, the address of the SPIn_RDR register.
   D. To the DMAn_MAR register, write the initial address of the memory to be read.
2. Enable RXDIE (DMA Rx Done Interrupt Enable) bit in the SPIn_CR register and SPI NVIC settings.
   A. Once the values are set in the previous steps, enable the DMAEN bit in the DMAn_SR register to receive the SPI data using DMA.
   B. Once the DMA transfer starts and when the value of the TRANSCNT[11:0] bits in the DMAn_CR register reaches '0', the RXDMAF flag in the DMAn_SR register indicates that the DMA transfer ends is set to '1'.
   C. Read SPI data into memory.

Figure 114. Example of Timing of SPI Transfer Operation Using DMA



13.4.9 SPI Status Flags

13.4.9.1 Transmit Buffer Empty Flag (TRDY)

The TRDY flag is set to ‘1’ when the Transmit Register is ready for data transmission. If this flag is ‘0’, it means that the SPI is not ready for Tx transfer. The initial status of this flag is ‘1’, and data transfer is possible.

13.4.9.2 Transmit/Receive Operation Flag (SBUSY)

The SBUSY flag is set to the initial state of ‘0’ when the transmission and reception operations are IDLE. If the transmission or reception is in progress for communication, this flag is set to ‘1’, which indicates busy.

13.4.9.3 Receive Buffer Ready Flag (RRDY)

The RRDY flag is set to 1’ when the Receive Register is ready to receive data. If this flag is ‘0’, it means that the Receive Register has no received data. The initial status of this flag is ‘0’, which requires receiving data.

13.4.9.4 SS Signal Status Flag (SSON)

The SSON flag implies the status of the SS signal. The SSON flag is set to ‘1’ when the SS signal is active, and the SSON flag is set to ‘0’ when the SS signal is deactivated.

13.4.9.5 Rising or Falling Edge of SS Signal Detect Flag (SSDET)

The SSDET flag is set to ‘1’ when falling edge or rising edge of the SS signal is detected. Writing ‘0’ to this bit field changes the flag value to ‘0’.

13.4.9.6 DMA Transmit Operation Complete Flag (DMA to SPI, TXDMAF)

The TXDMAF flag is set to ‘1’ when DMA transfer operation (Transmission) is completed. During DMA transmitting operation or if DMA is disabled, this flag is set to ’0’.

13.4.9.7 DMA Receive Operation Complete Flag (SPI to DMA, RXDMAF)

The RXDMAF flag is set to ‘1’ when DMA transfer operation (Reception) is completed. During DMA receiving operation or if DMA is disabled, this flag is set to ‘0’.

13.4.10 SPI FIFO Status Flags

This section describes the flags and status required when using the SPI FIFO. Before enabling the SPI FIFO function, the RXFEN, TXFEN bits in the SPIn_FCR Register must be set to '1'.

13.4.10.1 Receive FIFO Trigger Level Interrupt Flag (RXFTLIF)

Users can set the receive FIFO trigger level using RXFTLVL[3:0] in the SPIn_FCR Register. The current FIFO receive data level can be determined by checking FRLVL[3:0] in the SPIn_SR Register.

When the current FIFO reception level (FRLVL[3:0]) is greater than or equal to the set receive FIFO trigger level (RXFTLVL[3:0]), the RXFTLIF flag in the SPIn_SR Register is set to '1'.

To enable the Receive FIFO trigger Interrupt, set RXFTLIE bit in the SPIn_FCR Register to '1'. Writing '0' to this bit field changes the RXFTLIF flag value to '0'.

13.4.10.2 Transmit FIFO Trigger Level Interrupt Flag (TXFTLIF)

Users can set the transmit trigger level using TXFTLVL[3:0] in the SPIn_FCR Register. The current FIFO transmit data level can be determined by checking FTLVL[3:0] in the SPIn_SR Register.

When the current FIFO transmission level (FTLVL[3:0]) is less than or equal to the set transmit FIFO trigger level (TXFTLVL[3:0]), the TXFTLIF flag in the SPIn_SR Register is set to '1'.

To enable the Transmit FIFO trigger Interrupt, set TXFTLIE bit in the SPIn_FCR Register to '1'. Writing '0' to this bit field changes the TXFTLIF flag value to '0'.

13.4.10.3 Receive FIFO Full Interrupt Flag (RXFFIF)

The current FIFO receive data level can be determined by checking FRLVL[3:0] in the SPIn_SR Register.

The highest FIFO reception level is 8, and when the level becomes 8, the RXFFIF flag in the SPIn_SR Register is set to '1'.

To enable the Receive FIFO full Interrupt, set RXFFIE bit in the SPIn_CR Register to '1'. Writing '0' to this bit field changes the RXFFIF flag value to '0'.

13.4.10.4 Transmit FIFO Full Interrupt Flag (TXFFIF)

The current FIFO transmission data level is determined by checking FTLVL[3:0] in the SPIn_SR Register. The highest FIFO transmission level is 8, and when the level becomes 8, the TXFFIF flag in the SPIn_SR Register is set to '1'.

To enable the Transmit FIFO full Interrupt, set TXFFIE bit in the SPIn_FCR Register to '1'. Writing '0' to this bit field changes the RXFF flag value to '0'.

13.4.10.5 Receive FIFO Not Empty Interrupt Flag (RXFNEIF)

The current FIFO receive data level is determined by checking FRLVL[3:0] in the SPIn_SR Register. If the FIFO reception level is not empty, the RXFNEIF flag in the SPIn_SR Register is set to '1'; if the FIFO reception level is empty, the RXFNEIF flag in the SPIn_SR Register is set to '0'.

To enable the Receive FIFO not empty Interrupt, set RXFNEIE bit in the SPIn_FCR Register to '1'. Writing '0' to this bit field changes the RXFNEIF flag value to '0'.

13.4.10.6 Transmit FIFO Empty Interrupt Flag (TXFEIF)

The current FIFO transmission data level is determined by checking FTLVL[3:0] in the SPIn_SR Register.

If the FIFO transmission level is empty, the TXFEIF flag in the SPIn_SR Register is set to '1'; if the FIFO transmission level is not empty, the TXFEIF flag in the SPIn_SR Register is set to '0'.

To enable the Transmit FIFO empty Interrupt, set TXFEIE bit in the SPIn_FCR Register to '1'. Writing '0' to this bit field changes the TXFEIF flag value to '0'.

13.4.10.7 Receive FIFO Empty Status (RXFES)

The RXFES is a status bit, not a flag, which indicates the current state. This bit is set to '1' when FRLVL[3:0] in the SPIn_SR Register is empty; this bit is set to ‘0’ when FRLVL[3:0] in the SPIn_SR Register is not empty.

13.4.10.8 Transmit FIFO Empty Status (TXFES)

The TXFES is a status bit, not a flag, which indicates the current state. This bit is set to '1' when FTLVL[3:0] in the SPIn_SR Register is empty; this bit is set to ‘0’ when FTLVL[3:0] in the SPIn_SR Register is not empty.

13.4.11 SPI Error Flags

13.4.11.1 Overrun Flag (OVRF)

The OVRF flag is set to ‘1’ if additional SPI data is received while the SPI_RDR register is not yet read. Reading the SPI_RDR register clears the OVRF flag.

13.5 SPI Registers

The base address and register map of the SPI are described in the following tables:

Table 121. Base Address of SPI

Name	Base Address
SPI0	0x4000_9000
SPI1	0x4000_9100
SPI2	0x4000_9200
SPI3	0x4000_9300
SPI4	0x4000_9400
SPI5	0x4000_9500
SPI6	0x4000_9600
SPI7	0x4000_9700
SPI8	0x4000_9800
SPI9	0x4000_9900
SPI10	0x4000_9A00
SPI11	0x4000_9B00

Table 122. SPI Register Map

Name	Offset	Type	Description	Reset Value	Reference
SPIn_TDR	0x0000	WO	SPI Transmit Data Register	-	13.5.1
SPIn_RDR	0x0000	RO	SPI Receive Data Register	0x0000_0000	13.5.2
SPIn_CR	0x0004	RW	SPI Control Register	0x0000_1020	13.5.3
SPIn_SR	0x0008	RC	SPI Status Register	0x0000_0006	13.5.4
SPIn_BR	0x000C	RW	SPI Baud-rate Register	0x0000_FFFF	13.5.5
SPIn_EN	0x0010	RW	SPI Enable Register	0x0000_0000	13.5.6
SPIn_LR	0x0014	RW	SPI Delay Length Register	0x0001_0101	13.5.7
SPIn_FCR	0x0018	RW	SPI FIFO Control Register	0x0000_4000	13.5.8
SPIn_FTDR	0x001C	WO	SPI FIFO Transmit Data Register	0x0000_0000	13.5.9
SPIn_FRDR	0x0020	RO	SPI FIFO Receive Data Register	0x0000_0000	13.5.10

NOTE:

1. n = 0, 1, 2, 3, 4, 5, 6, 7, 8, 9, 10 and 11.

13.5.1 SPIn_TDR: SPI n Transmit Data Register

The SPI_TDR register is a 17-bit sized write register. It contains serial transmit data.

SPI0_TDR=0x4000_9000, SPI1_TDR=0x4000_9100, SPI2_TDR=0x4000_9200
SPI3_TDR=0x4000_9300, SPI4_TDR=0x4000_9400, SPI5_TDR=0x4000_9500
SPI6_TDR=0x4000_9600, SPI7_TDR=0x4000_9700, SPI8_TDR=0x4000_9800
SPI9_TDR=0x4000_9900, SPI10_TDR=0x4000_9A00, SPI11_TDR=0x4000_9B00

31 30 29 28 27 26 25 24	23 22 21 20 19 18 17 16	15 14 13 12 11 10 9 8	7 6 5 4 3 2 1 0
Reserved		TDR[16:0]	
-		-	
-		WO	

16 0	TDR[16:0]	Transmit data register

13.5.2 SPIn_RDR: SPI n Receive Data Register

The SPI_RDR register is a 17-bit sized read register. It contains serial receive data.

SPI0_RDR=0x4000_9000, SPI1_RDR=0x4000_9100, SPI2_RDR=0x4000_9200
SPI3_RDR=0x4000_9300, SPI4_RDR=0x4000_9400, SPI5_RDR=0x4000_9500
SPI6_RDR=0x4000_9600, SPI7_RDR=0x4000_9700, SPI8_RDR=0x4000_9800
SPI9_RDR=0x4000_9900, SPI10_RDR=0x4000_9A00, SPI11_RDR=0x4000_9B00

31 30 29 28 27 26 25 24	23 22 21 20 19 18 17 16	15 14 13 12 11 10 9 8	7 6 5 4 3 2 1 0
Reserved		RDR[16:0]	
-		0x0000	
-		RO	

16 0	RDR[16:0]	Receive data register

13.5.3 SPIn_CR: SPI n Control Register

The SPI_CR register is a 20-bit sized read/write register and can configure SPI operation mode.

SPI0_CR=0x4000_9004, SPI1_CR=0x4000_9104, SPI2_CR=0x4000_9204
SPI3_CR=0x4000_9304, SPI4_CR=0x4000_9404, SPI5_CR=0x4000_9504
SPI6_CR=0x4000_9604, SPI7_CR=0x4000_9704, SPI8_CR=0x4000_9804
SPI9_CR=0x4000_9904, SPI10_CR=0x4000_9A04, SPI11_CR=0x4000_9B04

31 30 29 28 27 26 25 24	23 22 21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved		TXBC	RXBC	DTXIE	DRXIE	SSCIE	TXIE	RXIE	SSMOD	SSOUT	LBE	SSMARK	SSMO	SSPOL	Reserved		MS	MSBF	CPHA	CPOL	BITSZ[1:0]	
-		0	0	0	0	0	0	0	0	1	0	0	0	0	-		1	0	0	0	00	
-		WO	WO	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	-		RW	RW	RW	RW	RW	

20	TXBC	Tx buffer clear bit. (auto-clear)
		0 No action
		1 Clear Tx buffer
19	RXBC	Rx buffer clear bit. (auto-clear)
		0 No action
		1 Clear Rx buffer
18	DTXIE	DMA Tx done interrupt enable bit.
		0 DMA Tx done interrupt is disabled.
		1 DMA Tx done interrupt is enabled.
17	DRXIE	DMA Rx done interrupt enable bit.
		0 DMA Rx done interrupt is disabled.
		1 DMA Rx done interrupt is enabled.
16	SSCIE	SS edge change interrupt enable bit.
		0 SS interrupt is disabled.
		1 SS interrupt is enabled for both edges (L → H, H → L)
15	TXIE	Transmit interrupt enable bit.
		0 Transmit interrupt is disabled.
		1 Transmit interrupt is enabled.
14	RXIE	Receive interrupt enable bit.
		0 Receive interrupt is disabled.
		1 Receive interrupt is enabled.
13	SSMOD	SS auto/manual output select bit.
		0 SS output is not set by SSOUT (SPI_CR[12]) SS signal is in Auto output operation mode.
		1 SS output signal is set by SSOUT. (SS Manual output)
12	SSOUT	SS output signal select bit.
		0 SS output is 'L.'
		1 SS output is 'H'.
11	LBE	Loop-back mode select bit in master mode.
		0 Loop-back mode is disabled.
		1 Loop-back mode is enabled.
10	SSMASK	SS signal masking bit in slave mode.
		0 SS signal masking is disabled. Receive data when SS signal is active.
		1 SS signal masking is enabled. Receive data at SCK edges. SS signal is ignored.
9	SSMO	SS output signal enable bit.
		0 SS output signal is disabled.

		1	SS output signal is enabled.
8	SSPOL	SS signal polarity select bit.	
		0	SS signal is active-low.
		1	SS signal is active-high.
5	MS	Master/Slave select bit.	
		0	SPI is in Slave mode.
		1	SPI is in Master mode.
4	MSBF	MSB/LSB transmit select bit.	
		0	LSB is transferred first.
		1	MSB is transferred first.
3	CPHA(1)	SPI clock phase bit.	
		0	Sampling of data occurs at odd edges (1, 3, 5,…, 15).
		1	Sampling of data occurs at even edges (2, 4, 6,…, 16).
2	CPOL(1)	SPI clock polarity bit.	
		0	Idle-low clocks selected.
		1	Idle-high clocks selected.
1 0	BITSZ[1:0]	Transmit/Receive data bits select bit.	
		00	8-bit
		01	9-bit
		10	16-bit
		11	17-bit

NOTE:

1. CPOL = ‘0’, CPHA = ‘0’ : data sampling at rising edge, data changing at falling edge
   CPOL = ‘0’, CPHA = ‘1’ : data sampling at falling edge, data changing at rising edge
   CPOL = ‘1’, CPHA = ‘0’ : data sampling at falling edge, data changing at rising edge
   CPOL = ‘1’, CPHA = ‘1’ : data sampling at rising edge, data changing at falling edge

13.5.4 SPIn_SR: SPI n Status Register

The SPI_SR register is a 10-bit sized read/write register. It contains the status of SPI interface.

Initially, the FIFO is disabled and the RXFES, TXFES, and TXFEIF bits have a reset value of '0'.

- Setting RXFEN in the SPIn_FCR Register to '1' sets RXFES bit to '1'.
- Setting TXFEN in the SPIn_FCR Register to '1' sets TXFES bit and TXFEIF bit to '1'.

SPI0_SR=0x4000_9008, SPI1_SR=0x4000_9108, SPI2_SR=0x4000_9208
SPI3_SR=0x4000_9308, SPI4_SR=0x4000_9408, SPI5_SR=0x4000_9508
SPI6_SR=0x4000_9608, SPI7_SR=0x4000_9708, SPI8_SR=0x4000_9808
SPI9_SR=0x4000_9908, SPI10_SR=0x4000_9A08, SPI11_SR=0x4000_9B08

31 30 29 28	27 26 25 24	23	22	21	20	19	18	17	16	15 14 13 12 11 10	9	8	7	6	5	4	3	2	1	0
FRLVL[3:0]	FTLVL[3:0]	RXFTLIF	TXFTLIF	RXFFIF	TXFFIF	RXFES	TXFES	RXFNEIF	TXFEIF	Reserved	TXDMAF	RXDMAF	SBUSY	SSDET	SSON	OVRF	Reserved	TXIDLE	TRDY	RRDY
0	0	0	0	0	0	0	0	0	0	-	0	0	0	0	0	0	-	1	1	0
RO	RO	RWC0	RWC0	RWC0	RWC0	RO	RO	RWC0	RWC0	-	RWC0	RWC0	RO	RWC0	RO	RWC0	-	RO	RO	RO

31 28	FRLVL[3:0]	FIFO Reception level
		0 FIFO receive buffer empty
		1 Receive 1 data
		2 Receive 2 data
		3 Receive 3 data
		4 Receive 4 data
		5 Receive 5 data
		6 Receive 6 data
		7 Receive 7 data
		8 FIFO receive buffer full (Receive 8 data)
27 24	FTLVL[3:0]	FIFO transmission level
		0 FIFO transmit buffer empty
		1 Transmit 1 data
		2 Transmit 2 data
		3 Transmit 3 data
		4 Transmit 4 data
		5 Transmit 5 data
		6 Transmit 6 data
		7 Transmit 7 data
		8 FIFO transmit buffer full (Transmit 8 data)
23	RXFTLIF	Receive FIFO trigger level interrupt flag If the current FIFO reception level (FRLVL[3:0]) is greater than or equal to the set receive FIFO trigger level (RXFTLVL[3:0]), this bit is set to '1'.
		0 Not flagged. (FRLVL < RXFTLVL)
		1 Flagged (Writing a '0' to the bit clears the flag).
22	TXFTLIF	Transmit FIFO trigger level interrupt flag If the current FIFO transmission level (FTLVL[3:0]) is less than or equal to the set transmit trigger level (TXFTLVL[3:0]), this bit is set to '1'.
		0 Not flagged. (FTLVL > TXFTLVL)
		1 Flagged (Writing a '0' to the bit clears the flag).
21	RXFFIF	Receive FIFO full interrupt flag

		0	Not flagged. (Receive FIFO not full)
		1	Flagged (Writing a ‘0’ to the bit clears the flag).
20	TXFFIF	Transmit FIFO full interrupt flag	
		0	Not flagged. (Transmit FIFO not full)
		1	Flagged (Writing a ‘0’ to the bit clears the flag).
19	RXFES	Receive FIFO empty status	
		0	Receive FIFO not empty
		1	Receive FIFO empty
18	TXFES	Transmit FIFO empty status	
		0	Transmit FIFO not empty
		1	Transmit FIFO empty
17	RXFNEIF	Receive FIFO not empty interrupt flag	
		0	Not flagged. (Receive FIFO empty)
		1	Flagged (Writing a ‘0’ to the bit clears the flag).
16	TXFEIF	Transmit FIFO empty interrupt flag	
		0	Not flagged. (Transmit FIFO not empty)
		1	Flagged (Writing a ‘0’ to the bit clears the flag).
9	TXDMAF	DMA Transmit Operation Complete flag. (DMA to SPI)	
		0	DMA transmit operation is working or is disabled.
		1	DMA transmit operation is done. This value is cleared by writing ‘0’ to this field.
8	RXDMAF	DMA receive operation complete flag. (SPI to DMA)	
		0	DMA receive operation is working or is disabled.
		1	DMA receive operation is done. This value is cleared by writing ‘0’ to this field.
7	SBUSY	Transmit or receive flag	
		0	Idle state
		1	Transmit or receive in progress
6	SSDET	The rising or falling edge of SS signal detect flag.	
		0	SS edge is not detected.
		1	SS edge is detected. The bit is cleared when it is written as ‘0’.
5	SSON	SS signal Status flag.	
		0	SS signal is inactive.
		1	SS signal is active.
4	OVRF	Receive overrun error flag.	
		0	Receive overrun error is not detected.
		1	Receive overrun error is detected. This bit is cleared by reading SPI_RDR and SPI_FRDR, or writing ‘0’ to this field.
2	TXIDLE	Transmit/receive operation flag.	
		0	SPI is transmitting data
		1	SPI is in IDLE state.
1	TRDY	Transmit buffer empty flag.	
		0	Transmit buffer is busy.
		1	Transmit buffer is ready. This bit is cleared by writing data to SPI_TDR.
0	RRDY	Receive buffer ready flag.	
		0	Receive buffer has no data.
		1	Receive buffer has data. This bit is cleared by reading data from SPI_RDR.

13.5.5 SPIn_BR: SPI n Baud-Rate Register

The SPI_BR register is a 16-bit sized read/write register. Baud-rate can be set by writing the register.

SPI0_BR=0x4000_900C, SPI1_BR=0x4000_910C, SPI2_BR=0x4000_920C
SPI3_BR=0x4000_930C, SPI4_BR=0x4000_940C, SPI5_BR=0x4000_950C
SPI6_BR=0x4000_960C, SPI7_BR=0x4000_970C, SPI8_BR=0x4000_980C
SPI9_BR=0x4000_990C, SPI10_BR=0x4000_9A0C, SPI11_BR=0x4000_9B0C

31 30 29 28 27 26 25 24	23 22 21 20 19 18 17 16	15 14 13 12 11 10 9 8	7 6 5 4 3 2 1 0
Reserved		BR[15:0]	
-		0xFFFF	
-		RW	

1	BR[15:0]	Baud-rate setting bits Baud-rate = PCLK / (BR[15:0] + 1) (BR[15:0] must be set 2 or greater. (BR[15:0] ≥ 2)

NOTE:

1. For SPI speed, it is recommended to set the BR[15:0] value to two or greater so that the SPI input clock is divided by at least 3.
   e.g., PCLK = 24 MHz, BR = 2, SPI freq. = 24 MHz / (2 + 1) = 8 MHz

13.5.6 SPIn_EN: SPI n Enable Register

The SPI_EN register is a bit sized read/write register. It contains SPI enable bit.

SPI0_EN=0x4000_9010, SPI1_EN=0x4000_9110, SPI2_EN=0x4000_9210
SPI3_EN=0x4000_9310, SPI4_EN=0x4000_9410, SPI5_EN=0x4000_9510
SPI6_ EN=0x4000_9610, SPI7_EN=0x4000_9710, SPI8_EN=0x4000_9810
SPI9_ EN=0x4000_9910, SPI10_EN=0x4000_9A10, SPI11_EN=0x4000_9B10

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

0	ENABLE	SPI enable bit
	0	SPI is disabled. The SPI_SR register is initialized by writing '0' to this bit but other registers aren't initialized.
	1	SPI is enabled. When this bit is written as '1', the dummy data of transmit buffer will be shifted. To prevent this, write data to the SPI_TDR register before this bit is active.

13.5.7 SPIn_LR: SPI n Delay Length Register

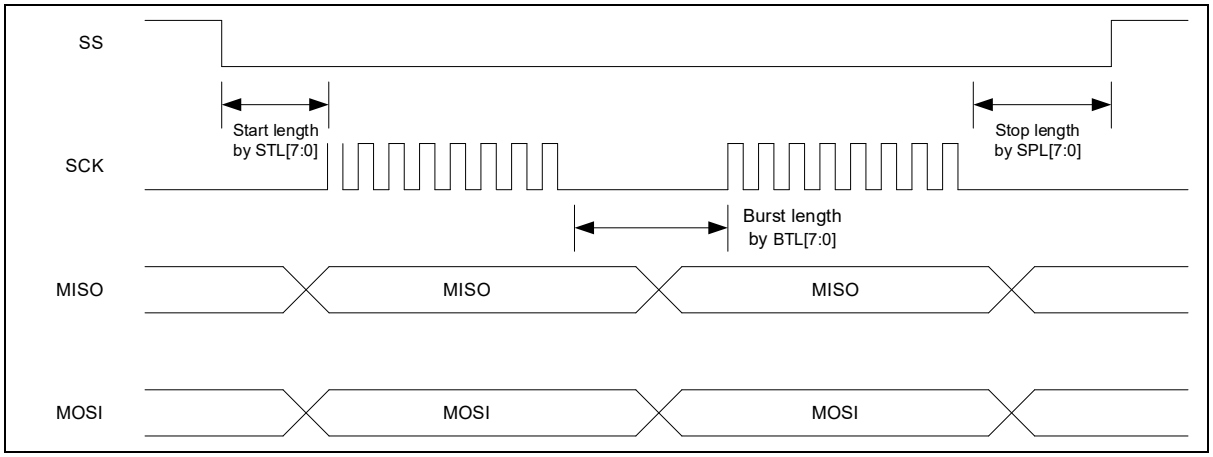
The SPI_LR register contains the length values of Start, Burst, and Stop. The SPI must be disabled in the SPI_EN register before configuring this register.

SPI0_LR=0x4000_9014, SPI1_LR=0x4000_9114, SPI2_LR=0x4000_9214
SPI3_LR=0x4000_9314, SPI4_LR=0x4000_9414, SPI5_LR=0x4000_9514
SPI6_LR=0x4000_9614, SPI7_LR=0x4000_9714, SPI8_LR=0x4000_9814
SPI9_LR=0x4000_9914, SPI10_LR=0x4000_9A14, SPI11_LR=0x4000_9B14

31 30 29 28 27 26 25 24	23 22 21 20 19 18 17 16	15 14 13 12 11 10 9 8	7 6 5 4 3 2 1 0
Reserved	SPL[7:0]	BTL[7:0]	STL[7:0]
-	0x01	0x01	0x01
-	RW	RW	RW

23 16	SPL[7:0]	Stop length value 0x01 to 0xFF: 1 to 255 SCKs. (SPL[7:0] ≥ 1)
15 8	BTL[7:0]	Burst length value 0x01 to 0xFF: 1 to 255 SCKs. (BTL[7:0] ≥ 1)
7 0	STL[7:0]	Start length value 0x01 to 0xFF: 1 to 255 SCKs. (STL[7:0] ≥ 1)

Figure 115. SPI Waveforms (STL[7:0], BTL[7:0], and SPL[7:0])(1)



NOTE:

1. Due to the nature of code flash memory, the access time is slower than RAM, causing flash waits. Since the reception of data during SPI high-speed communication performed in code flash memory may be faster than the data access time, so we recommend using a delay between the sending SPI data. By running code in SRAM and using DMA, you can use it without delay in high-speed communication.

$$BTL(burst\ delay) \geq \frac{(Access\ wait + 1) \times SCK \times (xbit + 2)}{HCLK} + \frac{4}{(BR + 1)}$$

13.5.8 SPIn_FCR: SPI n FIFO Control Register

The SPIn_FCR register is a 32-bit register. It sets the operating modes and enablement of the SPI FIFO module.

SPI0_FCR=0x4000_9018, SPI1_FCR=0x4000_9118, SPI2_FCR=0x4000_9218
SPI3_FCR=0x4000_9318, SPI4_FCR=0x4000_9418, SPI5_FCR=0x4000_9518
SPI6_FCR=0x4000_9618, SPI7_FCR=0x4000_9718, SPI8_FCR=0x4000_9818
SPI9_FCR=0x4000_9918, SPI10_FCR=0x4000_9A18, SPI11_FCR=0x4000_9B18

31 30 29 28 27 26 25 24	23 22 21 20	19 18 17	16	15 14 13 12	11 10 9	8	7	6	5	4	3	2	1	0
Reserved	TXFTLVL[3:0]	Reserved	TXFTLIE	RXFTLVL[3:0]	Reserved	RXFTLIE	RXFFIE	TXFFIE	RXFNEIE	TXFEIE	RXFBCLR	TXFBCLR	RXFEN	TXFEN
-	0x0	-	0	0x4	-	0	0	0	0	0	0	0	0	0
-	RW	-	RW	RW	-	RW	RW	RW	RW	RW	WO	WO	RW	RW

23 20	TXFTLVL[3:0]	Transmit FIFO trigger level select bit.	
		0000	Transmit 0 data Trigger (Tx FIFO empty)
		0001	Transmit 1 data Trigger
		0010	Transmit 2 data Trigger
		0011	Transmit 3 data Trigger
		0100	Transmit 4 data Trigger
		0101	Transmit 5 data Trigger
		0110	Transmit 6 data Trigger
		0111	Transmit 7 data Trigger
		Other	Transmit 7 data Trigger (Similar TXFFIE)
16	TXFTLIE	Transmit FIFO trigger level interrupt enable bit.	
		0	Transmit FIFO trigger interrupt is disabled.
		1	Transmit FIFO trigger interrupt is enabled.
15 12	RXFTLVL[3:0]	Receive FIFO trigger level select bit.	
		0000	Receive 0 data Trigger (Rx FIFO empty)
		0001	Receive 1 data Trigger
		0010	Receive 2 data Trigger
		0011	Receive 3 data Trigger
		0100	Receive 4 data Trigger
		0101	Receive 5 data Trigger
		0110	Receive 6 data Trigger
		0111	Receive 7 data Trigger
		Other	Receive 7 data Trigger (Similar RXFFIE)
8	RXFTLIE	Receive FIFO trigger level interrupt enable bit.	
		0	Receive FIFO trigger interrupt is disabled.
		1	Receive FIFO trigger interrupt is enabled.
7	RXFFIE	Receive FIFO full interrupt enable bit.	
		0	Receive FIFO full Interrupt is disabled.
		1	Receive FIFO full Interrupt is enabled.
6	TXFFIE	Transmit FIFO full interrupt enable bit.	
		0	Transmit FIFO full interrupt is disabled.
		1	Transmit FIFO full interrupt is enabled.
5	RXFNEIE	Receive FIFO not empty interrupt enable bit.	
		0	Receive FIFO not empty interrupt is disabled.
		1	Receive FIFO not empty interrupt is enabled.

4	TXFEIE	Transmit FIFO empty interrupt enable bit.	
		0	Transmit empty interrupt is disabled.
		1	Transmit empty interrupt is enabled.
3	RXFBCLR	Receive FIFO buffer clear bit. (auto-clear)	
		0	Not active
		1	Clear receive FIFO buffer (FRLVL[3:0] set ‘0’)
2	TXFBCLR	Transmit FIFO buffer clear bit. (auto-clear)	
		0	Not Active
		1	Clear transmit FIFO buffer (FTLVL[3:0] set ‘0’)
1	RXFEN	Receive FIFO enable bit.	
		0	Receive FIFO is disabled.
		1	Receive FIFO is enabled.
0	TXFEN	Transmit FIFO enable bit.	
		0	Transmit FIFO is disabled.
		1	Transmit FIFO is enabled.

NOTE:

1. To ensure stable operation of the receive FIFO trigger level function, set the RXFTLVL[3:0] bits to ‘1’ or higher.

13.5.9 SPIn_FTDR: SPI n FIFO Transmit Data Register

The SPI_FTDR register is a 17-bit sized write register. It contains FIFO transmit data. Only operation in FIFO mode.

SPI0_FTDR=0x4000_901C, SPI1_FTDR=0x4000_911C, SPI2_FTDR=0x4000_921C
SPI3_FTDR=0x4000_931C, SPI4_FTDR=0x4000_941C, SPI5_FTDR=0x4000_951C
SPI6_FTDR=0x4000_961C, SPI7_FTDR=0x4000_971C, SPI8_FTDR=0x4000_981C
SPI9_FTDR=0x4000_991C, SPI10_FTDR=0x4000_9A1C, SPI11_FTDR=0x4000_9B1C

31 30 29 28 27 26 25 24	23 22 21 20 19 18 17 16	15 14 13 12 11 10 9 8	7 6 5 4 3 2 1 0
Reserved		FTDR[16:0]	
-		0x0000	
-		WO	

16	FTDR[16:0]	FIFO transmit data register
0		

13.5.10 SPIn_FRDR: SPI n FIFO Receive Data Register

The SPI_FRDR register is a 17-bit sized read register. It contains FIFO receive data. Only operation in FIFO mode.

SPI0_FRDR=0x4000_9020, SPI1_FRDR=0x4000_9120, SPI2_FRDR=0x4000_9220
SPI3_FRDR=0x4000_9320, SPI4_FRDR=0x4000_9420, SPI5_FRDR=0x4000_9520
SPI6_FRDR=0x4000_9620, SPI7_FRDR=0x4000_9720, SPI8_FRDR=0x4000_9820
SPI9_FRDR=0x4000_9920, SPI10_FRDR=0x4000_9A20, SPI11_FRDR=0x4000_9B20

31 30 29 28 27 26 25 24	23 22 21 20 19 18 17 16	15 14 13 12 11 10 9 8	7 6 5 4 3 2 1 0
Reserved		FRDR[16:0]	
-		0x0000	
-		RO	

16	FRDR[16:0]	FIFO receive data register
0		

13.5.11 SPI Registers Map Summary

Table 123. SPI Register Map Summary

Offset	Register name	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0x00	SPIn_TDR	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	TDR[16:0]																
	Reset value																-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-
0x00	SPIn_RDR	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	RDR[16:0]																
	Reset value																0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
0x04	SPIn_CR	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	TXBC	RXBC	TXDIE	RXDIE	SSCIE	TXIE	RXIE	SSMOD	SSOUT	LBE	SSMARK	SSMO	SSPOL	Res.	Res.	MS	MSBF	CPHA	CPOL	BITSZ[1:0]	
	Reset value												0	0	0	0	0	0	0	0	1	0	0	0	0			1	0	0	0	0	0
0x08	SPIn_SR	FRLVL[3:0]				FTLVL[3:0]				RXFTLIF	TXFTLIF	RXFFIF	TXFFIF	RXFES	TXFES	RXFNEIF	TXFEIF	Res.	Res.	Res.	Res.	Res.	Res.	TXDMAF	RXDMAF	SBUSY	SSDET	SSON	OVRF	Res.	TXIDLE	TRDY	RRDY
	Reset value	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0							0	0	0	0	0	0		1	1	0
0x0C	SPIn_BR	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	BR[15:0]															
	Reset value																	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1
0x10	SPIn_EN	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	ENABLE
	Reset value																																0
0x14	SPIn_LR	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	SPL[7:0]								BTL[7:0]								STL[7:0]							
	Reset value									0	0	0	0	0	0	0	1	0	0	0	0	0	0	0	1	0	0	0	0	0	0	0	1
0x18	SPIn_FCR	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	TXFTLVL[3:0]				Res.	Res.	Res.	TXFTLIE	RXFTLVL[3:0]				Res.	Res.	Res.	RXFTLIE	RXFFIE	TXFFIE	RXFNEIE	TXFEIE	RXFBCLR	TXFBCLR	RXFEN	TXFEN
	Reset value									0	0	0	0				0	0	1	0	0				0	0	0	0	0	0	0	0	0
0x1C	SPIn_FTDR	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	FTDR[16:0]																
	Reset value																0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
0x20	SPIn_FRDR	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	FRDR[16:0]																
	Reset value																0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

14. Inter-Integrated Circuit (I2C)

14.1 I2C Introduction

The I2C interface built into the A34G43x series satisfies the standard I2C communication protocol and is used for serial communication with internal and external devices via the I2C protocol. Equipped with four units, it supports both master and slave modes and is capable of transmitting and receiving data in bytes by using interrupts or polling.

The I2C of the A34G43x operates in Standard mode (100 kHz) or Fast mode (400 kHz) and supports General call.

It helps communicate with various peripherals that have the same bus type. To use the I2C, it is recommended to set the SCL and SDA pins to open-drain and then connect external pull-up resistors to render their output signals ‘HIGH’.

14.2 I2C Main Features

The I2C of the A34G43x features the following:

- Compliant with I2C protocol
- Four units supported
- Master and slave modes
- Multi-slave mode
- General call
- Transfer rates configurable
  - Maximum transfer rate: 400 kHz
- I2C interrupts
- 7-bit addressing
- Delay time can be set for pin SCL’s high or low waveform
- Hold time can be set for previous data
- Generates and detects STOP, START, and ACK signals
- Noise canceller

14.3 I2C Implementation

The I2C module has all features of an I2C peripheral as shown in Table 124.

Table 124. Features of I2C

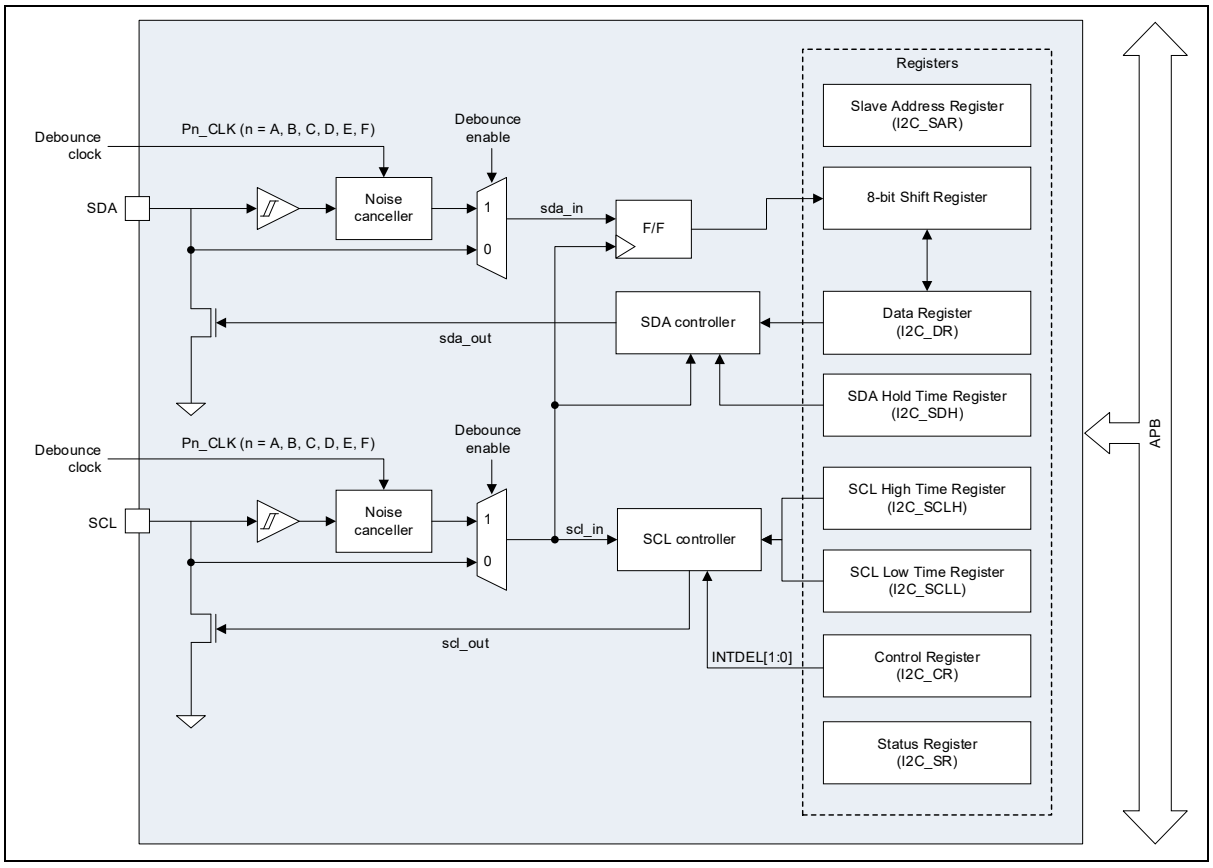
I2C Features	I2C
7-bit addressing mode	O
Standard-mode (up to 100 kbit/s)	O
Fast-mode (up to 400 kbit/s)	O
General call	O

14.4 I2C Functional Description

14.4.1 I2C Block Diagram

Figure 116 shows a block diagram of the I2C module.

Figure 116. I2C Block Diagram



NOTES:

1. When the corresponding port of the I2C is a sub-function for SCL/SDA pin, the SCL/SDA pins are automatically configured in open-drain outputs, and the input latch is read in the case of reading the pins. The corresponding internal pull-up resistor could be determined by the port n control register.
2. For more information on the clock source used by the noise canceler, refer to section 4.8.31 or section 4.8.32.

14.4.2 I2C Pins and Signals

Table 125 describes pins assigned for the I2C interface.

Table 125. Pin Assignment of I2C: External Pins

Pin Name	Type	Description	Supported Packages		
			A34G43xYL (120-LQFP)	A34G43xVL (100-LQFP)	A34G43xRL (64-LQFP)
SCL0	I/O	I2C channel 0 serial clock bus line (Open-drain)	O	O	O
SDA0	I/O	I2C channel 0 serial data bus line (Open-drain)	O	O	O
SCL1	I/O	I2C channel 1 serial clock bus line (Open-drain)	O	O	O
SDA1	I/O	I2C channel 1 serial data bus line (Open-drain)	O	O	O
SCL2	I/O	I2C channel 2 serial clock bus line (Open-drain)	O	O	O
SDA2	I/O	I2C channel 2 serial data bus line (Open-drain)	O	O	O
SCL3	I/O	I2C channel 3 serial clock bus line (Open-drain)	O	O	O
SDA3	I/O	I2C channel 3 serial data bus line (Open-drain)	O	O	O

14.4.3 I2C Mode Selection

The I2C module operates in one of the following modes:

- Slave transmitter
- Slave receiver
- Master transmitter
- Master receiver

The I2C module supports the four operation modes above and operates as a slave by default. It can operate as a master by configuring the START bit in the I2C_CR register.

START and STOP conditions can be controlled by software, and ACK functionality and General call recognition capability can be controlled by software too.

14.4.4 I2C Initialization

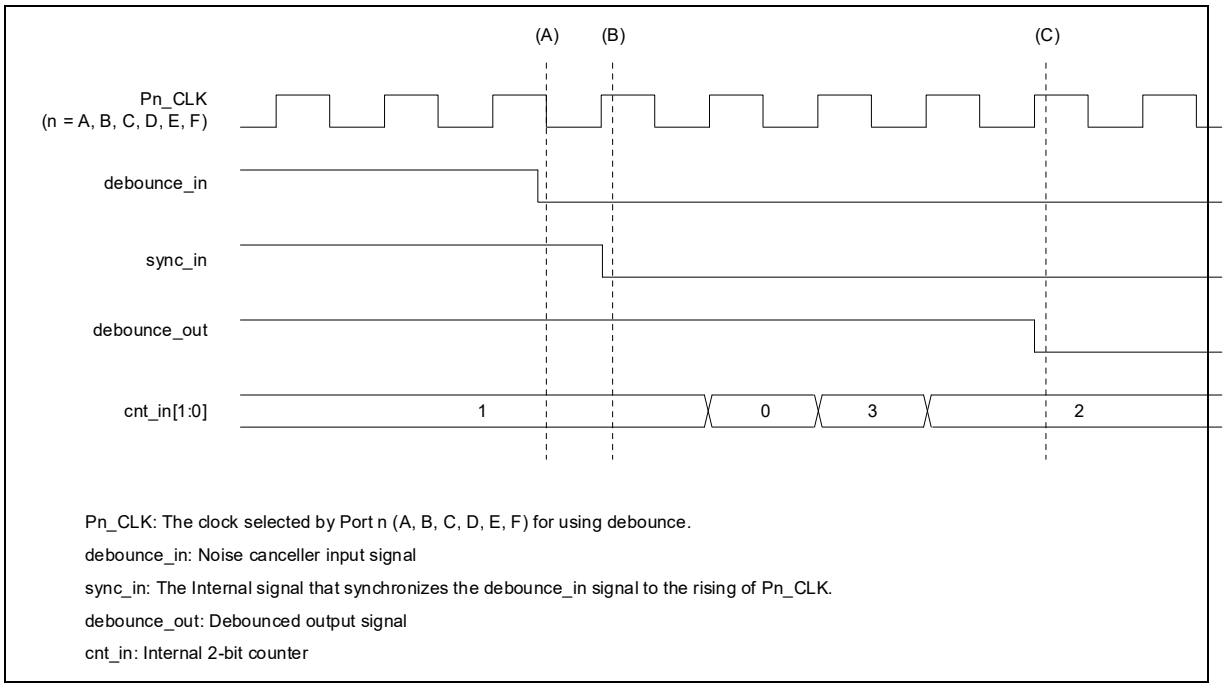
14.4.4.1 Enabling and Disabling Peripherals

To use the I2C, follow the procedure below:

1. Set the I2C in SCU_PER2 register to ‘1’. This enables the I2C module.
2. Set the I2C in SCU_PCER2 register to ‘1’. This enables the I2C module clock.
3. Configure the I2C_CR register to control the I2C communication.

14.4.4.2 Noise Canceller

Figure 117. Noise Canceller Timing



To apply the noise canceller to I2C, follow the procedure below:

1. Synchronize the debounce_in, input signal to the Noise canceller, with the rising edge of the Pn_CLK to generate the sync_in signal.
2. The sync_in signal checks the internal counter value from starting point of the (B) section and outputs the filtered value at the point of (C).
3. The time elapsed between the input signal and output signal is calculated by the formular below:

   A. Pn_CLK period × 4 to Pn_CLK × 5 (n = A, B, C, D, E, F)

14.4.4.3 Using Noise Canceller in I2C Module

To enable noise canceller, follow the procedure below.

1. I2C module uses PA, PB, PC, PD, PE, PF. So, select the PGBDCSEL (PC, PD) bits in the SCU_MCCR5 register or the PGCDCSEL (PE, PF, PG) bits in the SCU_MCCR5 register or the PGADCSEL (PA, PB) bits in the SCU_MCCR4. According to the port to be used and set the clock source for debouncing.
2. Set the PGBDCDIV (PC, PD) bits in the SCU_MCCR5 register or the PGCDCDIV (PE, PF, PG) bits in the SCU_MCCR5 register or the PGADCDIV (PA, PB) bits in the SCU_MCCR4 register to divide the clock source for debouncing.
3. Enable debouncing by setting the PDEx (x = 0 to 15) bits in the Px_DER (x = A to G) registers according to the port pin using I2C module.

NOTE:

1. If debounce is used, the I2C may experience a slowdown due to the Clock Stretching that can occur as much as the Debounce Delay. Users can increase the I2C speed by setting the SCLH[15:0] bits in the I2C_SCLH register to the value less than the current setting.

14.4.5 I2C Software Reset

The I2C can be reset by software, that is, users can reset the I2C by configuring the SCU_PER2 register. Writing ‘0’ to the I2C in SCU_PER2 register resets all related registers to the initialization state.

To use the I2C module again, set the I2Cn in SCU_PER2 register to ‘1’ again.

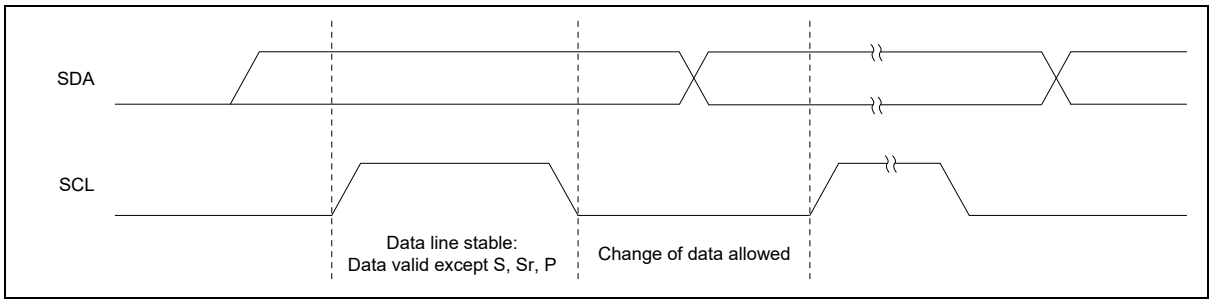
14.4.6 I2C Module Protocol

14.4.6.1 I2C Bit Transfer

Data on the SDA line must be stable during the “H” period of the clock. The “H” or “L” state of the data line can only change when the clock signal on the SCL line is “L”. However, START (S), repeated START (Sr), and STOP (P) occur when the SDA data changes while the SCL signal is high.

Figure 118 shows a timing diagram of the I2C bus bit transfer.

Figure 118. I2C Bus Bit Transfer Timing Diagram



14.4.6.2 START, Repeated START, and STOP

During the operation of the I2C bus, unique situations arise, which are defined as the START (S) and STOP (P) conditions (See Figure 119):

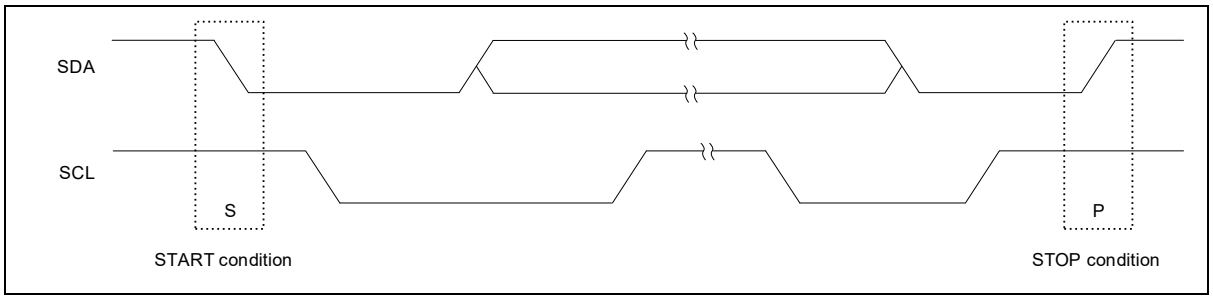
- START (S) condition: An “H” to “L” transition on the SDA line while SCL is “H.”
- STOP (P) condition: An “L” to “H” transition on the SDA line while SCL is “H.”

These START and STOP conditions are always generated by the master. The bus is considered busy once the START condition occurs and is considered as free again after the STOP condition occurs.

Thus, the bus stays busy between a START and a STOP. If a repeated START (Sr) is generated instead of a STOP, the bus remains busy. In this respect, the START (S) and repeated START (Sr) conditions are functionally identical. For the remainder of this document, therefore, an S will be used to represent both the START and repeated START conditions, unless Sr is particularly relevant.

The detection of the START and STOP conditions by devices connected to the bus is easy if the necessary interfacing hardware is incorporated. However, microcontrollers with no such interface have to sample the SDA line at least twice per clock period to sense the transition.

Figure 119. START and STOP Conditions



14.4.6.3 Acknowledge

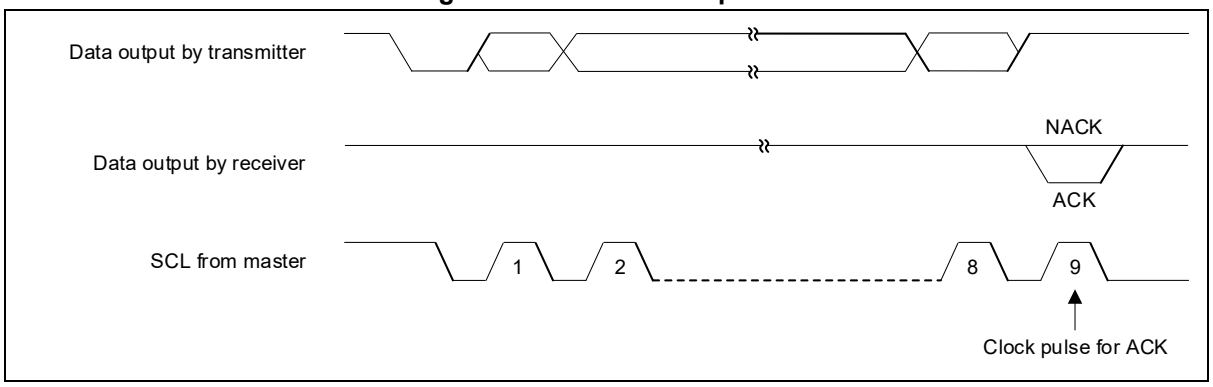
Data transfer with acknowledgement is necessary. The acknowledge-related clock pulse is generated by the master. The transmitter releases the SDA line (“H”) during the acknowledge clock pulse. The receiver must pull down the SDA line during the acknowledge clock pulse so that it remains at stable “L” during the “H” period of this clock pulse. (See Figure 120)

Set-up and hold times must also be considered. When a slave does not acknowledge the slave address (For example, it’s unable to receive or transmit because it’s performing some real-time function), the data line must be left “H” by the slave. Then the master can generate either the STOP condition to abort the transfer, or the repeated START condition to start a new transfer.

If a slave-receiver acknowledges the slave address but can receive no more data bytes later during the transfer, the slave leaves the data line at “H” and the master generates either the STOP or the repeated START condition.

If a master-receiver is involved in a transfer, the slave-transmitter must release the SDA line (“H”) during the acknowledge clock pulse of the end of data. So, the master could generate the STOP or repeated START condition.

Figure 120. I2C Bus Response

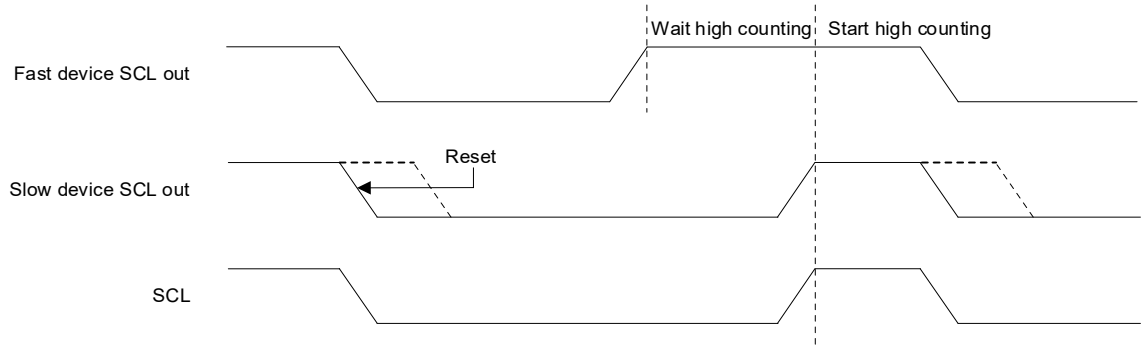


14.4.6.4 Synchronization

All masters generate their own clock on the SCL line to transfer messages through the I2C-bus. Data is only valid during the “H” period of the clock. A defined clock is therefore needed for the bit-by-bit arbitration procedure to take place.

Clock synchronization is performed using the wired-AND connection of I2C interfaces to the SCL line. This means that an “H” to “L” transition on the SCL line will cause the devices concerned to start counting off their “L” period and, once a device clock has gone “L”, it will hold the SCL line in that state until the clock “H” state is reached.

Figure 121. Clock Synchronization During Arbitration



NOTE:

1. The “L” to “H” transition of this clock may not change the state of the SCL line if another clock is still within its “L” by the device with the longest “L” period. Devices with shorter “L” periods enter an “H” wait-state during this time.

When all devices concerned have counted off their “L” period, the clock line will be released and go “H.” Then, there will be no difference between the device clocks and the state of the SCL line, and the devices will start counting their “H” periods. The first device to complete its “H” period will again pull the SCL line “L.”

14.4.6.5 Arbitration

A master may start a transfer only if the bus is free. Two or more masters may generate the START condition to the bus within the minimum hold time defined for this condition.

Arbitration takes place on the SDA line while the SCL line is in the “H” level in the condition that one master transmits the “H” level but another master is transmitting the “L” level; as a result, the master transmitting the “H” level switches off its DATA output stage because the level on the bus doesn’t correspond to its own level.

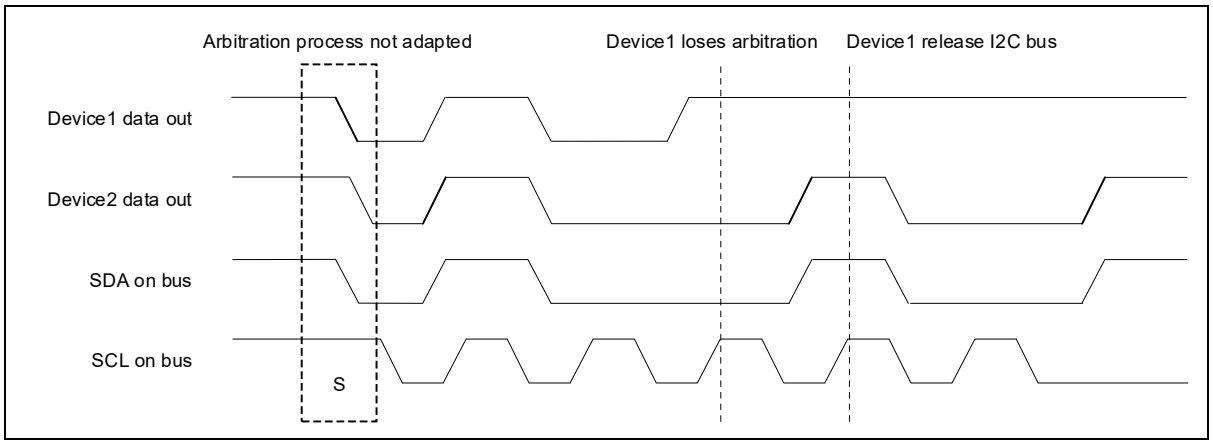
Arbitration can continue for many bits. Its first stage is the comparison of the address bits. If the masters are trying to address the same device, arbitration proceeds with comparing either the data-bits (If they are master-transmitters) or the acknowledge-bits (If they are master-receivers).

Because address and data information on the I2C bus is determined by the winning master, no information is lost during the arbitration process. A master that loses arbitration can generate clock pulses until the end of the byte in which it loses arbitration.

If a master also incorporates a slave function and it loses arbitration during the addressing stage, it is possible that the winning master is trying to address it. The losing master must therefore switch over immediately to its slave mode.

Figure 122 shows the arbitration process for two masters. Of course, more may be involved depending on how many masters are connected to the bus. As soon as a difference is made between the internal data level of the master generating Device1 Data out and the actual level on the SDA line, its data output is switched off, which means that a Device1 releases I2C bus. This will not affect the data transfer initiated by the winning master.

Figure 122. Arbitration Process between Two Masters



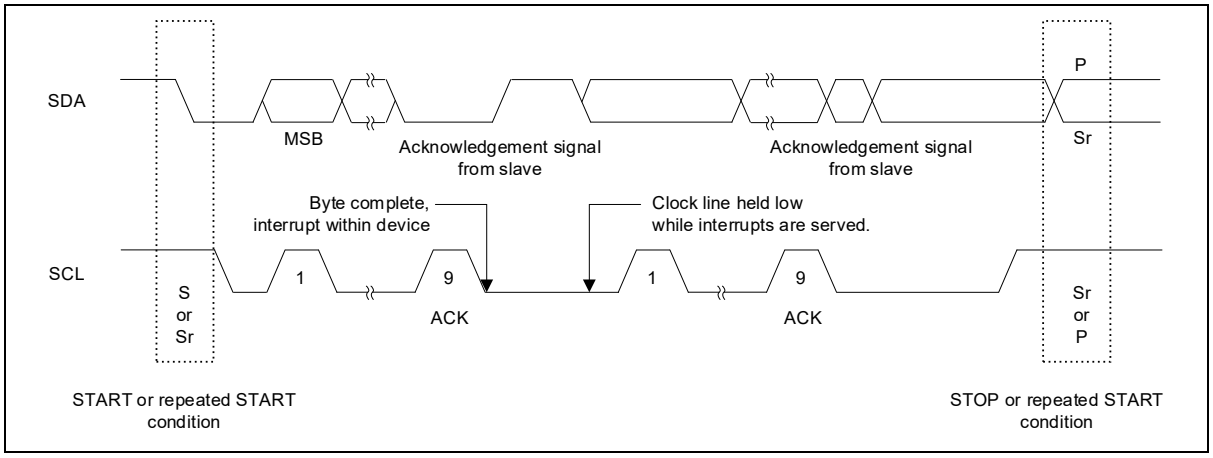
14.4.6.6 Data Transfer

Every byte put on the SDA line must be 8-bits long. The number of bytes that can be transmitted per transfer is unrestricted. Each byte must be followed by an acknowledge bit and is transferred MSB first. (Refer to Figure 123.)

If a slave cannot receive or transmit another complete byte of data until it has performed some other function, for example servicing an internal interrupt, it can hold the clock line SCL to “L” to force the master into a wait state. If the slave is ready to transfer another byte of data, releases clock line SCL and master continues data transfer.

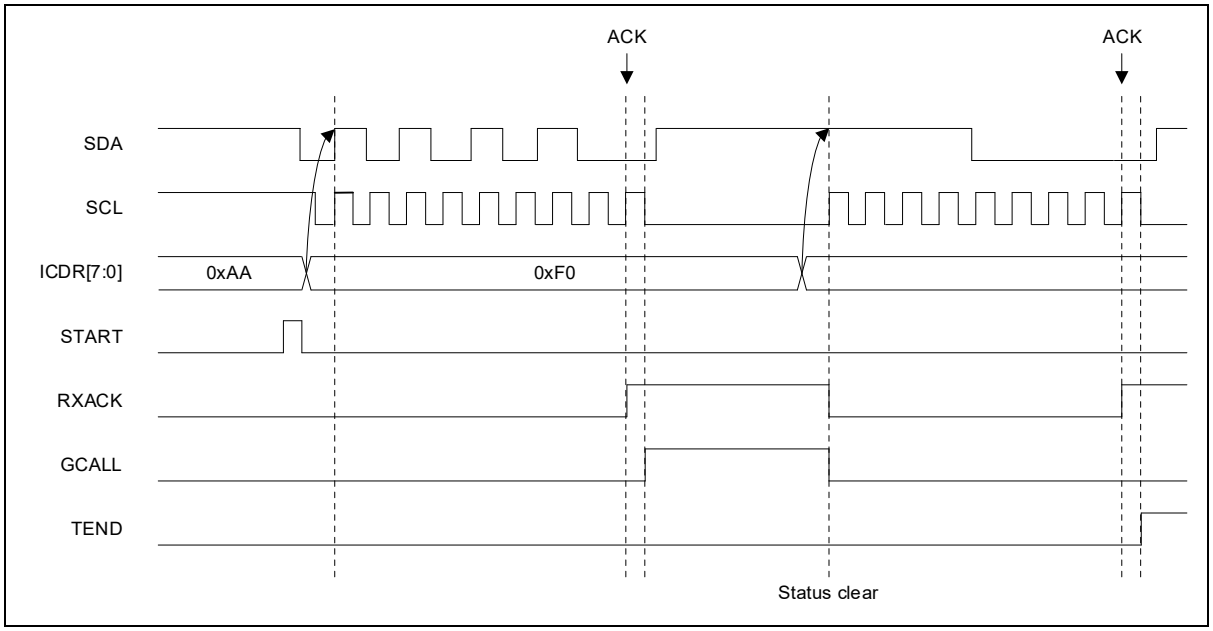
A message starting with such an address can be terminated by an occurrence of the STOP condition, even during the transmission of a byte. In this case, no acknowledgement is generated.

Figure 123. I2C Bus Data Transfer



14.4.6.7 Master Transmit Timing

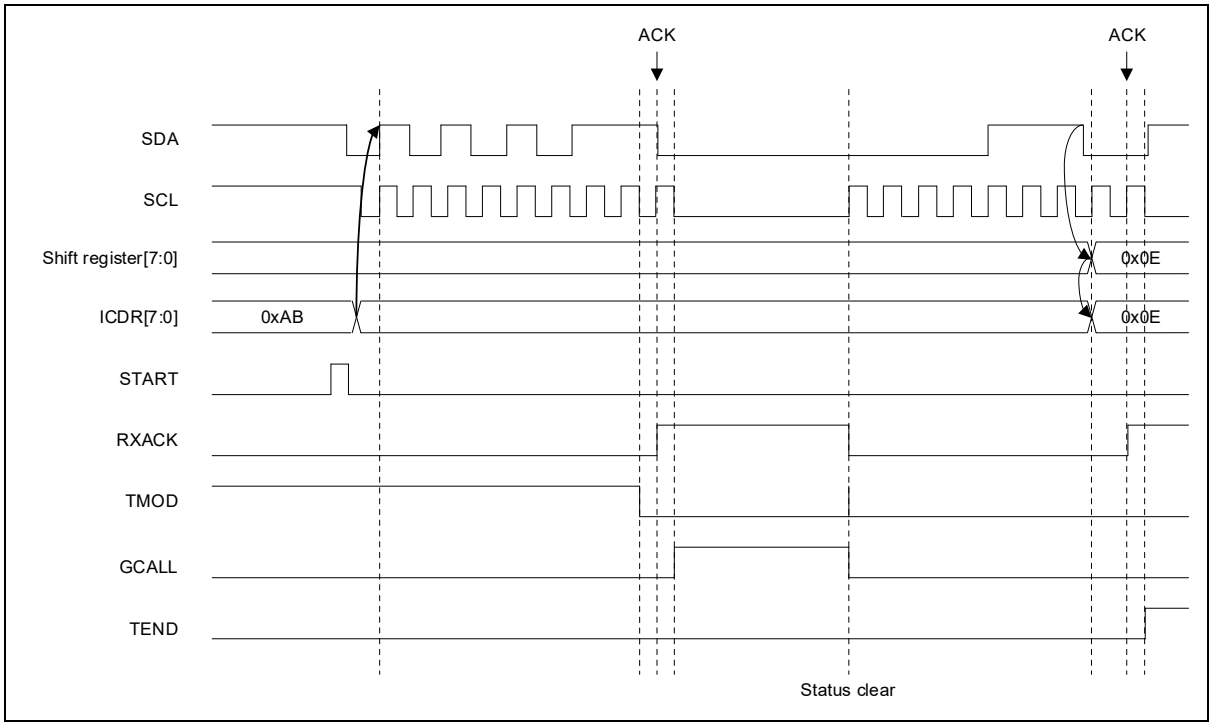
Figure 124. Master Transmit Timing



1. When the value of bit-0 stored in the I2Cn_DR register is ‘0’, if the START bit in the I2Cn_CR register is written as ‘1’, Master operates in Transmit mode.
2. 8-bit data stored in the I2Cn_DR register is transmitted during eight SCL pulses and ACK is received at the rising edge of the 9th pulse.
3. If ACK is normally received, the RXACK flag in I2Cn_SR register becomes ‘1’ and the GCALL flag in the I2Cn_SR register becomes ‘1’ at the falling edge of the nineth pulse. In Master mode, GCALL flag in the I2Cn_SR register is used as AACK (Address ACK judgment) bit.
4. If AACK is confirmed after address transmission, clear the I2Cn_SR register and transmit the data stored in the I2Cn_DR register again according to the SCL pulse. When data transfer is completed, the TEND flag in I2Cn_SR register becomes 1.

14.4.6.8 Master Receive Timing

Figure 125. Master Receive Timing

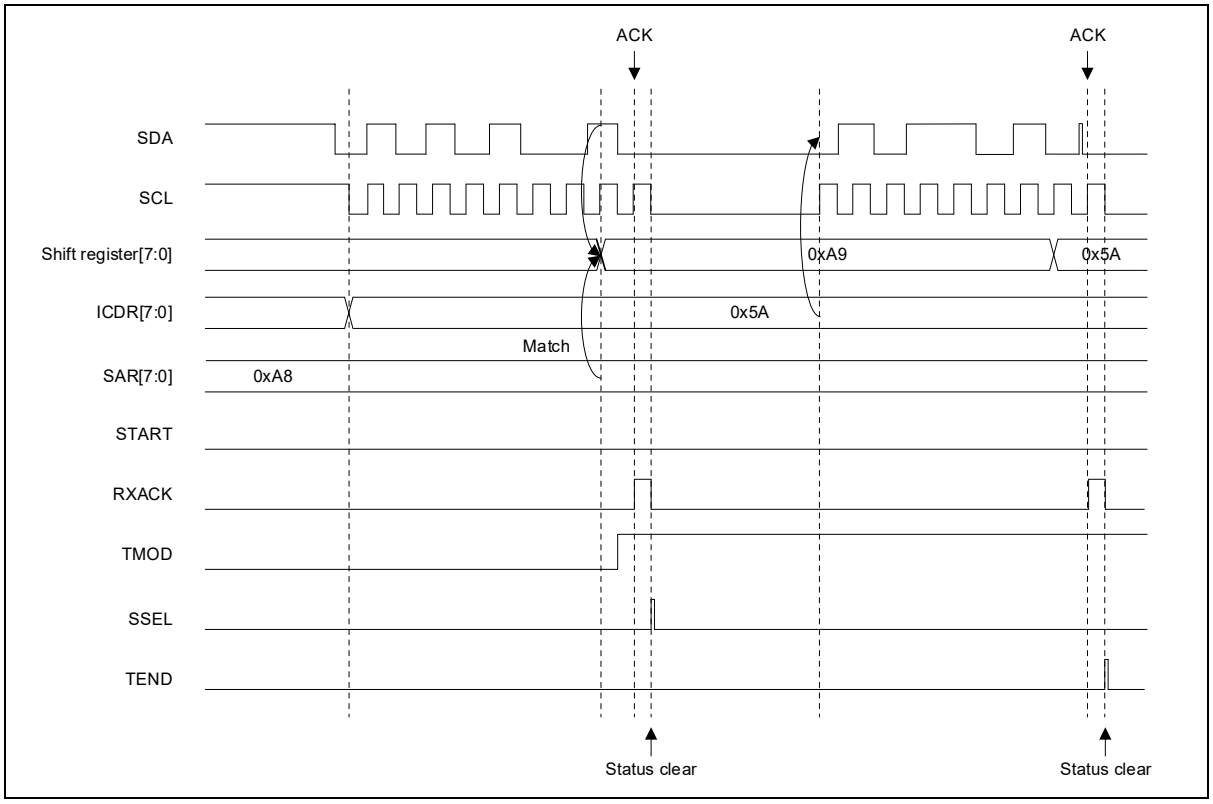


1. When the value of bit-0 stored in the I2Cn_DR register is ‘1’, if START bit in the I2Cn_CR register is written as ‘1’, Master operates in Receive mode.
2. 8-bit data stored in the I2Cn_DR register is transmitted during eight SCL pulses, and TMODE flag in the I2Cn_SR register becomes ‘0’ (Receive mode).
3. If ACK is normally received at the rising edge of the 9th pulse, the RXACK flag in the I2Cn_SR register becomes ‘1’ and the GCALL flag in the I2Cn_SR register becomes ‘1’ at the falling edge of the 9th pulse. In master mode, GCALL flag in the I2Cn_SR register is used as AACK (address ACK judgment) bit.
4. If ACK is confirmed after sending address, clear the I2Cn_SR register and prepare to receive data from slave.
5. The data received through the SDA line is saved in the shift register (SHFTR) at the 8th pulse of SCL and then immediately saved to the I2Cn_DR register without delay.

At the 9th pulse of SCL, the RXACK flag in the I2Cn_SR register becomes ‘1’. When data reception is completed, the TEND flag in the I2Cn_SR register becomes ‘1’.

14.4.6.9 Slave Transmit Timing

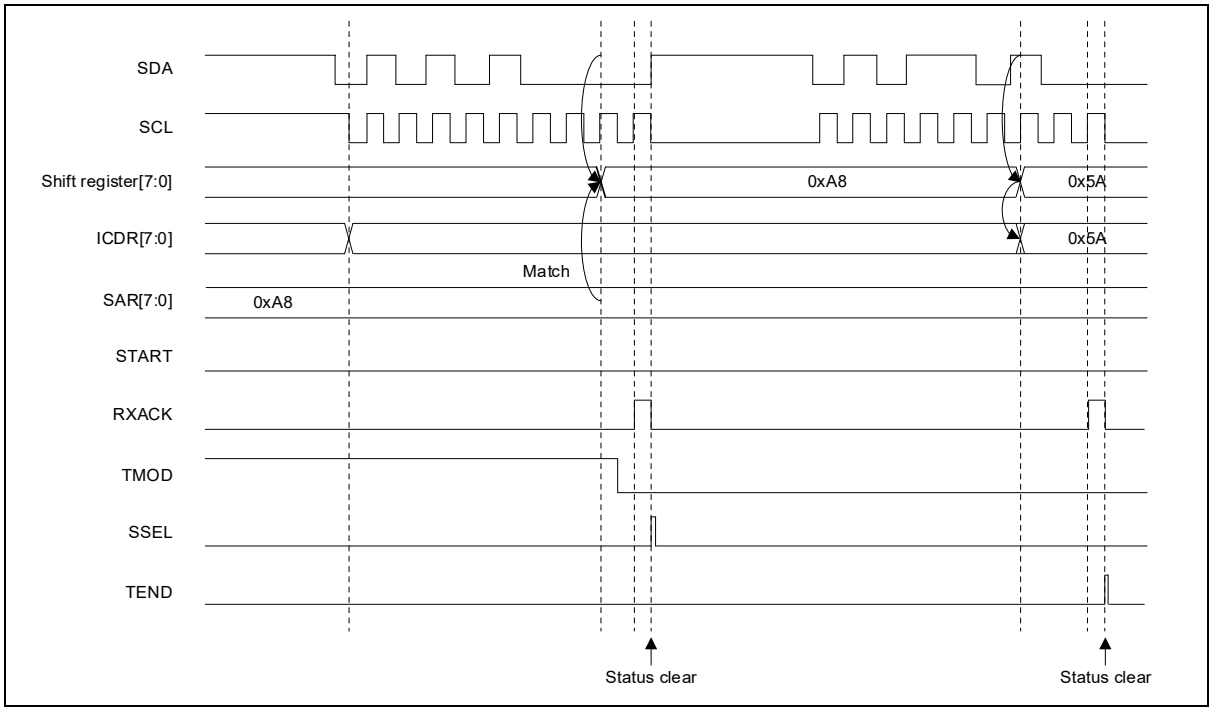
Figure 126. Slave Transmit Timing



1. If START bit in the I2Cn_CR register is kept as ‘0’, it operates in slave mode.
2. When SDA drops to Low while both SCL and SDA are high, the start condition starts, and the data received through the SDA line is saved in the shift register (SHFTR).
3. If the SHFTR[0] is '1' after the I2Cn_SAR register value and the address (SHFTR[7:1]) value stored in the shift register (SHFTR) match, it is transmit mode, so the TMODE flag in the I2Cn_SR register is '1'.
4. When the address matches, the SSEL flag in the I2Cn_SR register changes to ‘1’ and it is selected as slave, and the data stored in the I2Cn_DR register is transmitted according to the SCL pulse.
5. When all 8-bit data is transmitted, the TEND flag in the I2Cn_SR register becomes ‘1’ at the falling edge of the 9th pulse of SCL.

14.4.6.10 Slave Receive Timing

Figure 127. Slave Receive Timing



1. If START bit in the I2Cn_CR register is kept as ‘0’, it operates in slave mode.
2. When SDA drops to Low while both SCL and SDA are high, the start condition starts, and the data received through the SDA line is saved in the shift register (SHFTR).
3. If the I2Cn_SAR register value and the Address (SHFTR[7:1]) value stored in the shift register (SHFTR) match and SHFTR[0] is '0' in the GCALL state, it is Receive mode, so the TMODE flag in the I2C_SR register is '0'.
4. When the address matches, the SSEL flag in the I2Cn_SR register is changed to ‘1’ and it is selected as slave and receives 8-bit data from master.
5. When all 8-bit data is received, it is saved to the shift register (SHFTR) and then immediately saved to the I2Cn_DR register without delay.

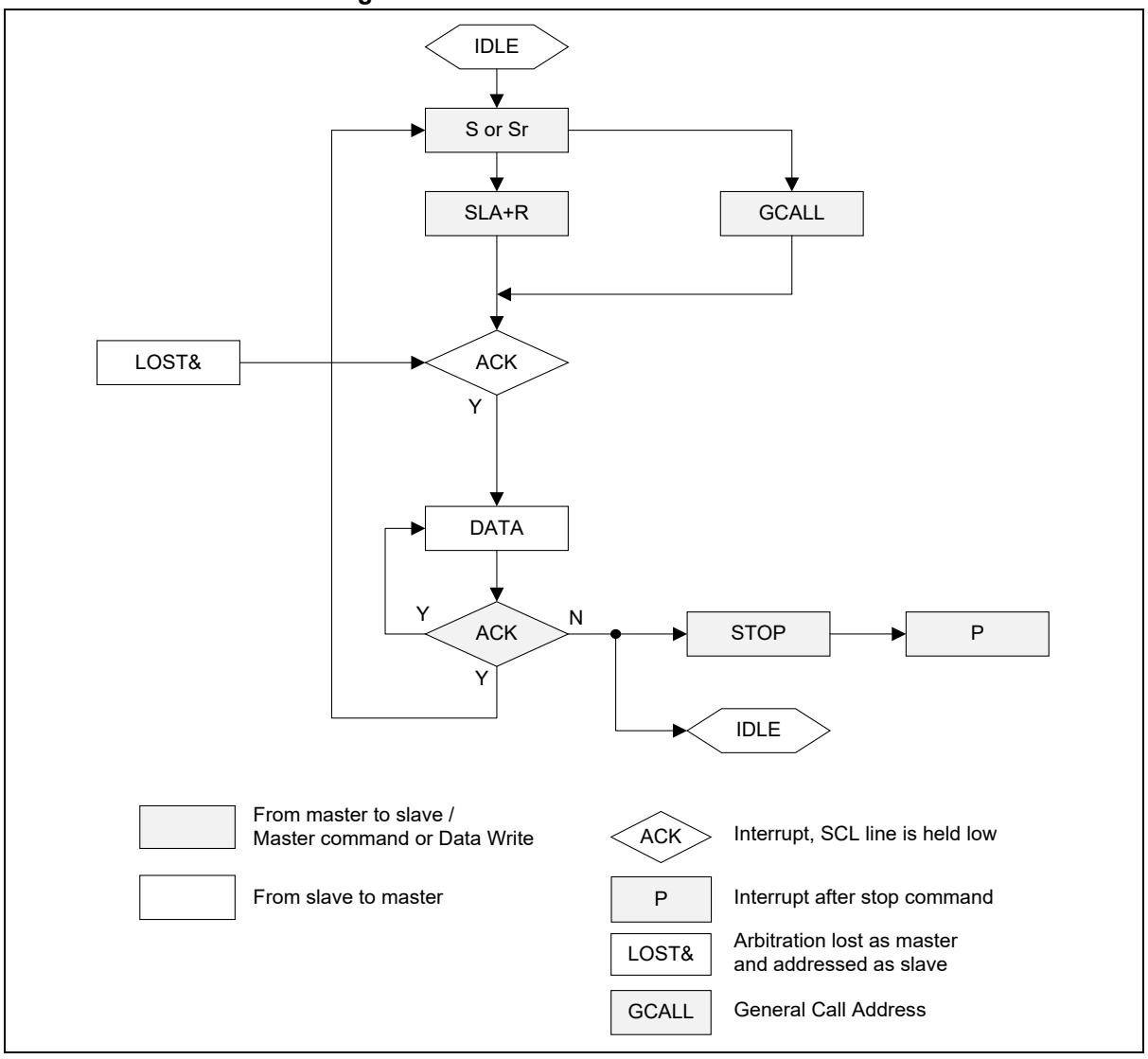
At the 9th pulse of SCL, the RXACK flag in the I2C_SR register becomes ‘1’. When data reception is completed, the TEND flag in the I2Cn_SR register becomes ‘1’.

14.4.7 I2C Slave Mode

14.4.7.1 Slave Transmitter

Figure 128 shows a flowchart of the transmitter in slave mode.

Figure 128. Slave Transmitter Flowchart



To operate the I2C in slave transmitter mode, users must follow the steps below:

1. If the main operating clock (PCLK) is slower than the SCL, users must set the I2Cn_SDH register to 0x00, so that the SDA is changed at the falling edge of the SCL. The SDA hold time is obtained by multiplying the I2Cn_SDH register value by the PCLK period. If the SDA hold time is longer than the SCL period, data cannot be properly transmitted.
2. Set the I2C bit in the SCU_PER2 and SCU_PCER2 registers to enable and clock the I2C.
3. Set the INTEN bit in the I2Cn_CR register to enable the I2C interrupts.

4. When the START condition is detected, the I2C receives one byte of data and compares it with value of the SVAD[6:0] in the I2Cn_SAR register. In the case that the GCEN bit in the I2Cn_SAR register is set enabled, the I2C raises the “general call” flag if the received data is 0x00.

5. If the received address does not match the SVAD[6:0] in the I2Cn_SAR register, the I2C becomes idle and waits until another START bit is detected. If the ACKEN bit is set enabled and the received address matches the address in the SVAD[6:0] bit field, the I2C triggers the SSEL interrupt and maintains the SCL line in the “L” level. If the ACKEN bit is set disabled, the I2C becomes idle even though the received address matches the SVAD[6:0] bit field. When the SSEL interrupt occurs, users must load the data in the I2Cn_DR register for transmission and release the SCL line by writing a value 0xFF to the I2Cn_SR register.

6. One byte of data is transmitted.

7. Once the transmission is completed, the I2C triggers the TEND interrupt regardless of an ACK signal from the master, and it maintains the SCL line in the “L” level. The slave performs one of the following operations:

   Case 1. If a NACK signal is detected, the I2C waits for the STOP condition or consecutive START condition.

   Case 2. If an ACK signal from the master is detected, the I2C loads the next data in the I2Cn_DR register.

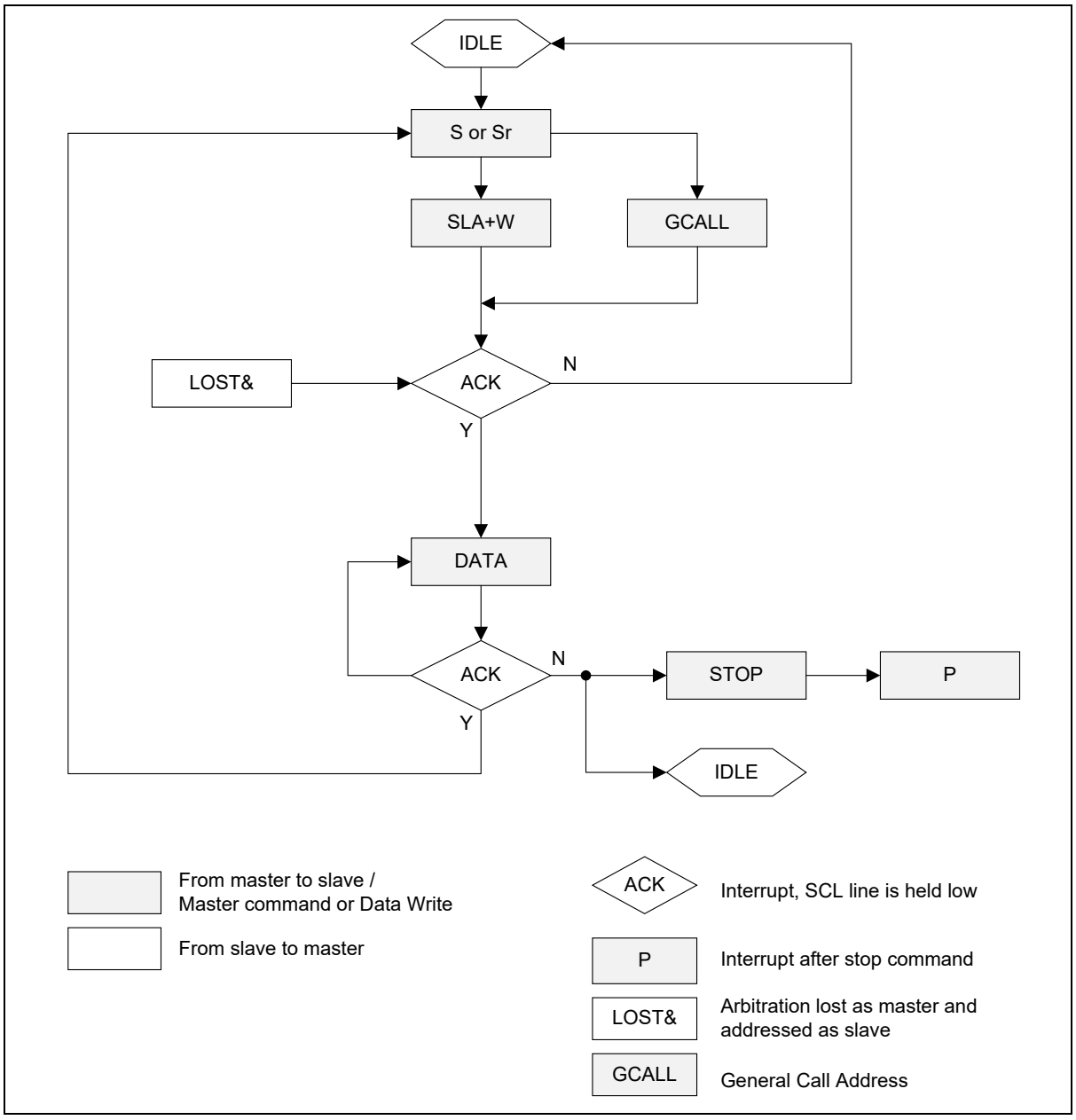
   Once one of the above operations is performed, release the SCL line by writing a value 0xFF to the I2Cn_SR register. For Case 1, go to step 8 and end the communication. For Case 2, go to step 6. In either case, the consecutive START condition can be detected; then, go to step 4.

8. The last step is to process the STOP interrupt. This interrupt indicates that data transfer between a master and slave has been completed. To clear the I2C_SR register, write the value 0xFF to the register. This causes the I2C to become idle.

14.4.7.2 Slave Receiver

Figure 129 shows a flowchart of the receiver in slave mode.

Figure 129. Slave Receiver Flowchart



To operate the I2C in slave receiver mode, users must follow the steps below:

1. If the main operating clock (PCLK) is slower than the SCL, users must set the I2Cn_SDH register to 0x00, so that the SDA is changed at the falling edge of the SCL. The SDA hold time is obtained by multiplying the I2Cn_SDH register value by the PCLK period. If the SDA hold time is longer than the SCL period, data cannot be properly transmitted.

2. Set the I2C bit in the SCU_PER2 and SCU_PCER2 registers to enable and clock the I2C.
3. Set the INTEN bit in the I2Cn_CR register to enable the I2C interrupts.
4. When the START condition is detected, the I2C receives one byte of data and compares it with the SVAD[6:0] in the I2Cn_SAR register. In the case that the GCEN bit in the I2Cn_SAR register is set enabled, the I2C raises the “general call” flag if the received data is 0x00.
5. If the received address does not match the SVAD[6:0] in the I2Cn_SAR register, the I2C becomes idle and waits until another START bit is detected. If the ACKEN bit is set enabled and the received address matches the address in the SVAD[6:0] bit field, the I2C triggers the SSEL interrupt and maintains the SCL line in the “L” level. If the ACKEN bit is set disabled, the I2C becomes idle even though the received address matches the SVAD[6:0] bit field. If the SSEL interrupt occurs, the I2C is ready to receive data. Release the SCL line by writing a value 0xFF to the I2Cn_SR register.
6. One byte of data is received.
7. Once the reception is completed, the I2C triggers the TEND interrupt regardless of an ACK signal from the slave, and it maintains the SCL line in the “L” level. The slave performs one of the following operations:

   Case 1. If an ACKEN = ‘0’, a NACK signal is generated and the I2C waits for the STOP condition or consecutive START condition.

   Case 2. If an ACKEN = ‘1’, an ACK signal is generated and the I2C can continue receiving data from the master.

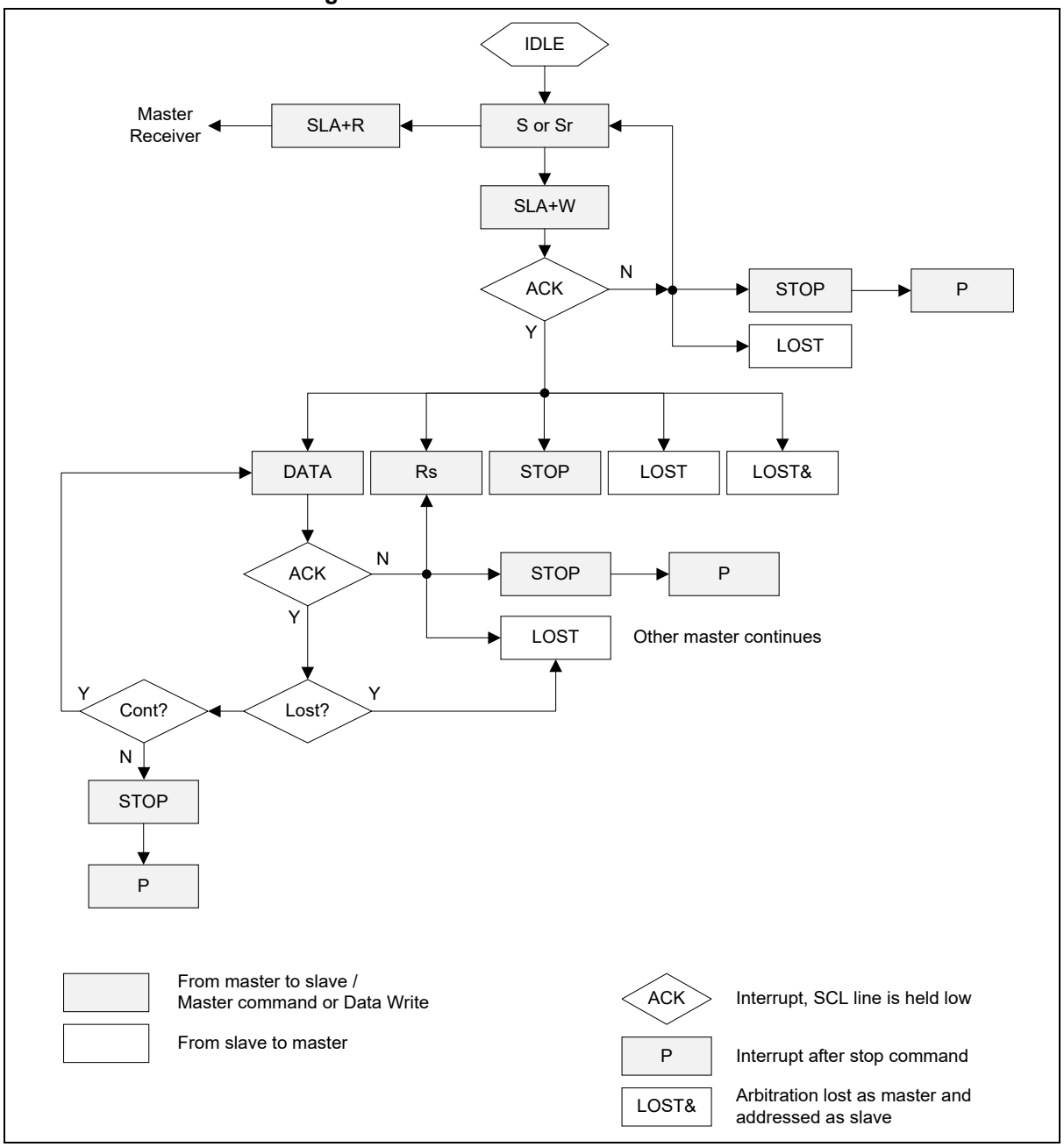
   Once one of the above operations is performed, release the SCL line by writing a value 0xFF to the I2Cn_SR register. For Case 1, go to step 8 and end the communication. For Case 2, go to step 6. In either case, the consecutive START condition can be detected; then, go to step 4.
8. The last step is to process the STOP interrupt. This interrupt indicates that data transfer between a master and slave has been completed. To clear the I2Cn_SR register, write a value 0xFF to the register. This causes the I2C to become idle.

14.4.8 I2C Master Mode

14.4.8.1 Master Transmitter

Figure 130 shows a flowchart of the transmitter in master mode.

Figure 130. Master Transmitter Flowchart



To operate the I2C in master transmitter mode, users must follow the steps below:

1. Set the I2C bit in the SCU_PER2 and SCU_PCER2 registers to enable and clock the I2C.
2. Set the INTEN bit in the I2Cn_CR register. to enable the I2C interrupts.

3. Enter “SLA+W” in the I2Cn_DR register (“SLA” = slave address, “W” = write). “W” must be set to ‘0’ in master transmitter mode. Note that the I2Cn_DR register is used for both address and data.

4. Set the I2Cn_SCLL and I2Cn_SCLH registers to configure the SCL transfer speed.

5. Set the I2Cn_SDH register, which determines the time SCL maintains the SDA value during the “L” period.

6. Write a ‘1’ to the START bit in the I2Cn_CR register to transmit the START condition. And then, configure interrupts and ACK processing conditions. Once the START bit is set enabled, the 8-bit data in the I2Cn_DR register is transmitted at the specified transfer speed.

7. First is the ACK processing sector for the address packet sent from the master. The slave receives a 7-bit address and a 1-bit transmit-receive direction. The master checks for an ACK response in the “H” sector of the ninth clock pulse. If the master has bus priority, the GCALL interrupts occur, regardless of receiving an ACK. If the I2C loses its bus mastership, the MLOST flag in the I2Cn_SR register is set to ‘1’. And the I2C either waits in idle mode or is operated in slave mode. To operate the I2C in slave mode when it has lost its bus mastership, the ACKEN bit in the I2Cn_CR register must be set to ‘1’ and the SVAD[6:0] in I2Cn_SAR register must be set to the specified slave address. When the I2C operates in slave transmit or receive mode, its SCL must be maintained in the “L” level, so that the I2C can decide whether to continue or stop transmission.

   The following is an example of how to operate the I2C when it does not lose its mastership during the first byte transmission. The I2C (master) performs one of the following operations regardless of receiving an ACK signal from the slave:

   Case 1. The master receives an ACK signal from the slave. Thereby, the master sends data to the slave. In this case, the data is written to the I2Cn_DR register for transmission.

   Case 2. Even if an ACK signal has been received, the master can stop transmission. In this case, the STOP in I2Cn_CR register must be set enabled.

   Case 3. The master sends the consecutive START condition without checking for an ACK signal. In this case, “SLA+R/W” must be written to the I2Cn_DR register; and the START bit in the I2Cn_CR register must be set to ‘1’.

   The SCL line is released when one of the above is performed and a value 0xFF is written to the I2Cn_SR register. For Case 1, go to step 8; for Case 2, go to step 10; and for Case 3, write the data in the I2Cn_DR register and go to step 6. If the transfer direction bit in “SLA+R/W” that is written to the I2Cn_DR register is set to ‘1’, go to the section on master receiver.

8. One byte of data is transmitted. Bus arbitration is valid during data transmission.

9. Next is the ACK processing sector for the data transmitted by the master. The I2C maintains the SCL in the “L” level. If the I2C’s bus mastership is taken over by another master during data transmission arbitration, the MLOST flag in the I2Cn_SR register is set to ‘1’. In this case, the I2C waits until it becomes idle. Once the data in the I2Cn_DR register is completely transmitted, the I2C triggers the TEND interrupt.

   The I2C (master) performs one of the following operations regardless of receiving an ACK signal from the slave:

   Case A. The master receives an ACK signal from the slave. Thereby, the master sends data to the slave. In this case, the data is written to the I2Cn_DR register for transmission.

   Case B. Even if an ACK signal has been received, the master can stop transmission. In this case, the STOP bit in the I2Cn_CR register must be set enabled.

   Case C. The master sends the consecutive START condition without checking for an ACK signal. In this case, “SLA+R/W” must be written to the I2Cn_DR register; and the START bit in the I2Cn_CR register must be set to ‘1’.

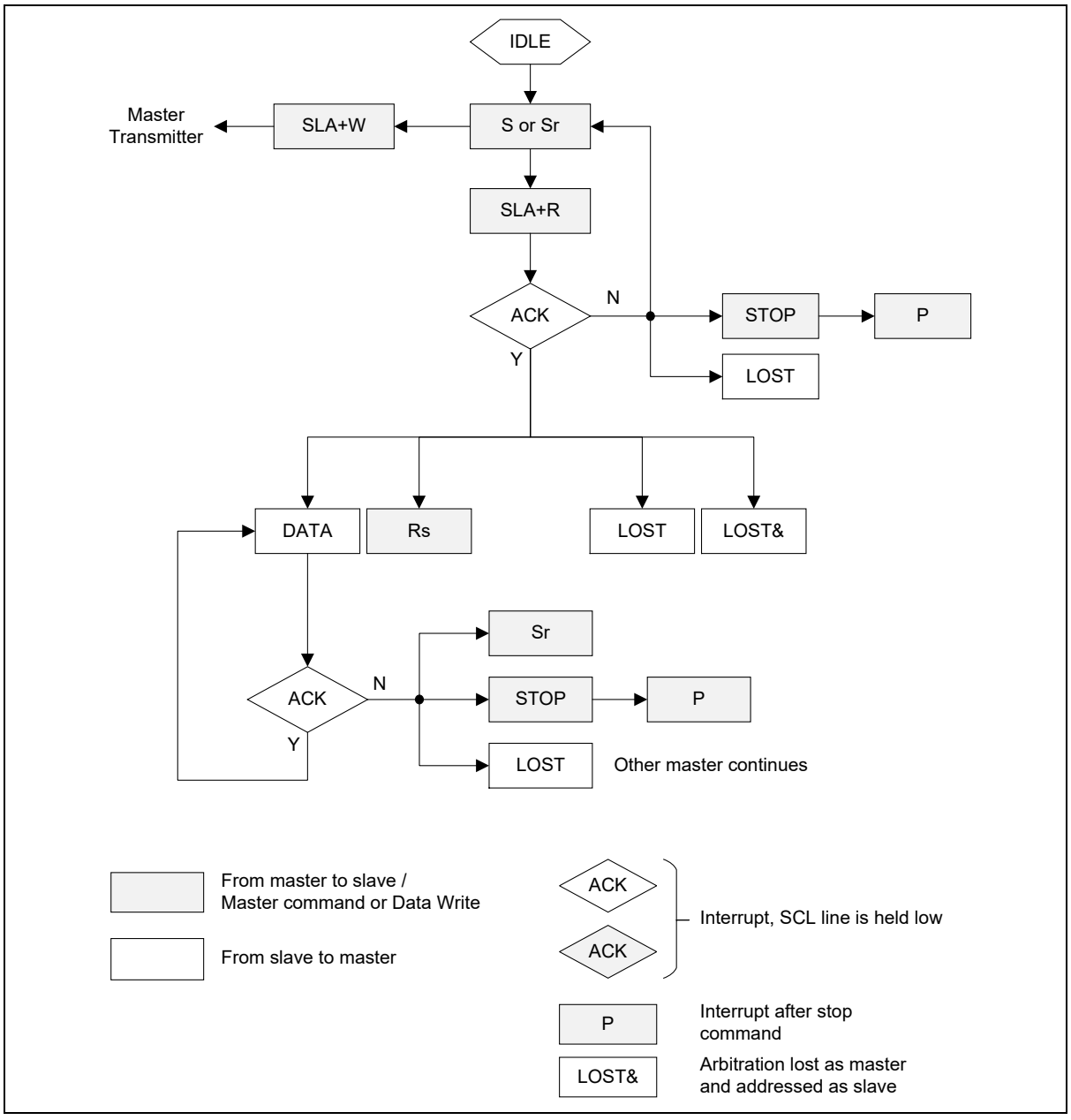
   The SCL line is released when one of the above is performed and a value 0xFF is written to the I2Cn_SR register. For Case A, go to step 8; for Case B, go to step 10; and for Case C, write the data in the I2Cn_DR register and go to step 6. If the transfer direction bit in “SLA+R/W” that is written to the I2Cn_DR register is set to ‘1’, go to the section on master receiver.

10. The last step is to process the STOP interrupt. This interrupt indicates that data transfer between a master and slave has been stopped by the STOP bit. To clear the I2Cn_SR register, write a value 0xFF to the register. This causes the I2C to become idle.

14.4.8.2 Master Receiver

Figure 131 shows a flowchart of the receiver in master mode.

Figure 131. Master Receiver Flowchart



To operate the I2C in master receiver mode, users must follow the steps below:

1. Set the I2C bit in the SCU_PER2 and SCU_PCER2 registers to enable and clock the I2C.
2. Set the INTEN bit in the I2Cn_CR register to enable the I2C interrupts.
3. Enter “SLA+R” in the I2C_DR register (“SLA” = slave address, “R” = read). “R” must be set to ‘1’ in master receiver mode. Note that the I2Cn_DR register is used for both address and data.

4. Set the I2Cn_SCLL and I2Cn_SCLH registers to configure the SCL transfer speed.
5. Set the I2Cn_SDH register, which determines the time SCL maintains the SDA value during the “L” period.
6. Write a ‘1’ to the START bit in the I2Cn_CR register to transmit the START condition. And then, configure interrupts and ACK processing conditions. Once the START bit is set enabled, the 8-bit data in the I2Cn_DR register is transmitted at the specified transfer speed.
7. The ACK processing sector for the address packet sent from the master. The slave receives a 7-bit address and a 1-bit transmit-receive direction. The master checks for an ACK response in the “H” sector of the ninth clock pulse. If the master has bus priority, the GCALL interrupts occur, regardless of receiving an ACK. If the I2C loses its bus mastership, the MLOST flag in the I2Cn_SR register is set to ‘1’. And the I2C either waits in idle mode or is operated in slave mode. To operate the I2C in slave mode when it has lost its bus mastership, the ACKEN bit in the I2Cn_CR register must be set to ‘1’ and the SVAD[6:0] in the I2Cn_SAR register must be set to the specified slave address. When the I2C operates in slave transmit or receive mode, its SCL must be maintained in the “L” level, so that the I2C can decide whether to continue or stop transmission. The following is an example for how to operate the I2C when it does not lose its mastership during the first byte transmission.

   The I2C (master) performs one of the following operations regardless of receiving an ACK signal from the slave:

   Case 1. The master receives an ACK signal from the slave. Thereby, the I2C is ready to receive data. In this case, the receive data is read from the I2Cn_DR register.

   Case 2. Even if an ACK signal has been received, the master can stop reception. In this case, the STOP bit in the I2Cn_CR register must be set enabled.

   Case 3. The master sends the consecutive START condition without checking for an ACK signal. In this case, “SLA+R/W” must be written to the I2Cn_DR register; and the START bit in the I2Cn_CR register must be set to ‘1’.

   The SCL line is released when one of the above is performed and a value 0xFF is written to the I2Cn_SR register. For case 1, go to step 8; for case 2, go to step 10; and for case 3, write the data in the I2Cn_DR register and go to step 6. If the transfer direction bit in “SLA+R/W” that is written to the I2Cn_DR register is set to ‘0’, go to the section on master transmitter.
8. One byte of data is received.

9. The ACK processing sector for the data received from the slave. The I2C maintains the SCL in the “L” level. Once a byte of data is received, the TEND interrupt occurs in the I2C.

   The I2C (master) performs one of the following operations regardless of the RXACK in I2Cn_SR register:

   Case A. The master continues to receive data from the slave. The ACKEN bit in the I2Cn_CR register is set to ‘1’ to transmit an ACK signal so that the slave can transmit the next data.

   Case B. The master receiver stops receiving because an ACK is not generated in the next receive data. To do so, the ACKEN bit in the I2Cn_CR register must be cleared.

   Case C. The master can stop data reception through generating NACK signal. When an ACK signal is not generated, the master stops data transfer. In this case, the STOP bit in the I2Cn_CR register must be set enabled.

   Case D. ACK signals are not generated, and the master sends the consecutive START condition. In this case, “SLA+R/W” must be loaded to the I2C_DR register; and the START bit in the I2Cn_CR register must be set to ‘1’.

   The SCL line is released when one of the above is performed and a value 0xFF is written to the I2Cn_SR register. For case A, go to step 8; for case B and C, go to step 10; and for case D, write the data in the I2Cn_DR register and go to step 6. If the transfer direction bit in “SLA+R/W” that is written to the I2Cn_DR register is set to ‘0’, go to the section on master transmitter.

10. The last step is to process the STOP interrupt. This interrupt indicates that data transfer between a master and slave has been stopped by the STOP bit. To clear the I2Cn_SR register, write a value 0xFF to the register. This causes the I2C to become idle.

14.5 I2C Low-Power Modes

Table 126. Effect of Low-Power Modes on I2C

Mode	Description
SLEEP	No effect
DEEP-SLEEP	The I2C module and clock are disabled. I2C module must be re-initialized after waking from DEEP-SLEEP mode.

14.6 I2C Interrupts

Setting the INTEN bit in the I2Cn_CR register to '1' can enable the I2C interrupt. With the INTEN bit enabled, the interrupt is executed as the SCL pulse is output when the master sets the START bit in I2Cn_CR register to '1'.

Operation status of the I2C can be monitored by checking the corresponding bits of the I2Cn_SR register, and writing '1' to the bit field clears the corresponding flag. For detailed information of the I2Cn_SR register, refer to section 14.4.7 and section14.4.8.

14.7 I2C Registers

The base address of the I2C and register map are described in the following:

Table 127. Base Address of I2C

Name	Base Address
I2C0	0x4000_A000
I2C1	0x4000_A100
I2C2	0x4000_A200
I2C3	0x4000_A300

Table 128. I2C Register Map

Name	Offset	Type	Description	Reset Value	Reference
I2Cn_DR	0x0000	RW	I2C n Data Register	0x0000_00FF	14.7.1
I2Cn_SR	0x0008	RW	I2C n Status Register	0x0000_0000	14.7.2
I2Cn_SAR	0x000C	RW	I2C n Slave Address Register	0x0000_0000	14.7.3
I2Cn_CR	0x0014	RW	I2C n Control Register	0x0000_0000	14.7.4
I2Cn_SCLL	0x0018	RW	I2C n SCL Low Duration Register	0x0000_FFFF	14.7.5
I2Cn_SCLH	0x001C	RW	I2C n SCL High Duration Register	0x0000_FFFF	14.7.6
I2Cn_SDH	0x0020	RW	I2C n SDA Hold Register	0x0000_7FFF	14.7.7

NOTE:

1. n = 0, 1, 2 and 3.

14.7.1 I2Cn_DR: I2C n Data Register

The I2Cn_DR register is a 32-bit register. It stores received byte-sized data or serial data for transmission.

I2C0_DR=0x4000_A000, I2C1_DR=0x4000_A100
I2C2_DR=0x4000_A200, I2C3_DR=0x4000_A300

31 30 29 28 27 26 25 24	23 22 21 20 19 18 17 16	15 14 13 12 11 10 9 8	7 6 5 4 3 2 1 0
Reserved			ICDR[7:0]
-			0xFF
-			RW

7 0	ICDR[7:0]	In transmit mode, the data stored in this register is output from pin SDA in the serial data line. In receive mode, data received at pin SDA is stored in this register; hence, the stored data is loaded by reading this register.

14.7.2 I2Cn_SR: I2C n Status Register

The I2Cn_SR register is a 32-bit read/write accessible register. It displays the status of the I2C bus interface. Writing to the register clears the status bits. Once an I2C interrupt other than the stop interrupt occurs, the SCL line is set low. To release this SCL setting, a value 0xFF must be written to the SR register. This will clear the status of the GCALL, TEND, STOP, SSEL, MLOST, and RXACK bits.

I2C0_SR=0x4000_A008, I2C1_SR=0x4000_A108
I2C2_SR=0x4000_A208, I2C3_SR=0x4000_A308

31 30 29 28 27 26 25 24	23 22 21 20 19 18 17 16	15 14 13 12 11 10 9 8	7	6	5	4	3	2	1	0
Reserved			GCALL	TEND	STOP	SSEL	MLOST	BUSY	TMOD	RXACK
-			0	0	0	0	0	0	0	0
-			RWC1	RWC1	RWC1	RWC1	RWC1	RO	RO	RWC1

7	GCALL	This bit signifies different things, depending on whether the I2C is in master or slave mode. (If the I2C module functions as a master, the bit represents whether or not the ACK address (AACK) has been received from a slave. If the I2C module is a slave, it indicates whether or not a general call has been detected.)	
		0	Master mode: AACK has not been received.
		1	Master mode: AACK has been received.
		0	Slave mode: A general call has not been detected.
		1	Slave mode: A general call has been detected.
6	TEND	One-byte transmit end flag	
		0	Transmit is in progress.
		1	Transmit has been completed.
5	STOP	Stop flag	
		0	No stop has been detected.
		1	A stop has been detected.
4	SSEL	Slave flag	
		0	The module has not been selected as a slave.
		1	The module has been selected as a slave.
3	MLOST	Mastership loss flag	
		0	Mastership has not been lost.
		1	Mastership has been lost.
2	BUSY	Bus busy flag	
		0	The I2C bus is idle.
		1	The I2C bus is busy.
1	TMOD	Transmit/receive mode flag	
		0	Receive mode
		1	Transmit mode
0	RXACK	Rx ACK flag	
		0	An Rx ACK has not occurred.
		1	An Rx ACK has occurred.

14.7.3 I2Cn_SAR: I2C n Slave Address Register

The I2Cn_SAR register is an 8-bit readable and writable register. The first seven bits store the address selected when the I2C module operates as a slave.

I2C0_SAR=0x4000_A00C, I2C1_SAR=0x4000_A10C
I2C2_SAR=0x4000_A20C, I2C3_SAR=0x4000_A30C

31 30 29 28 27 26 25 24	23 22 21 20 19 18 17 16	15 14 13 12 11 10 9 8	7 6 5 4 3 2 1	0
Reserved			SVAD[6:0]	GCEN
-			0x00	0
-			RW	RW

7 1	SVAD[6:0]	7-bit slave address	
0	GCEN	Enable or disable general call	
		0	Dismisses the general call address.
		1	Receives the general call address.

14.7.4 I2C_CR: I2C n Control Register

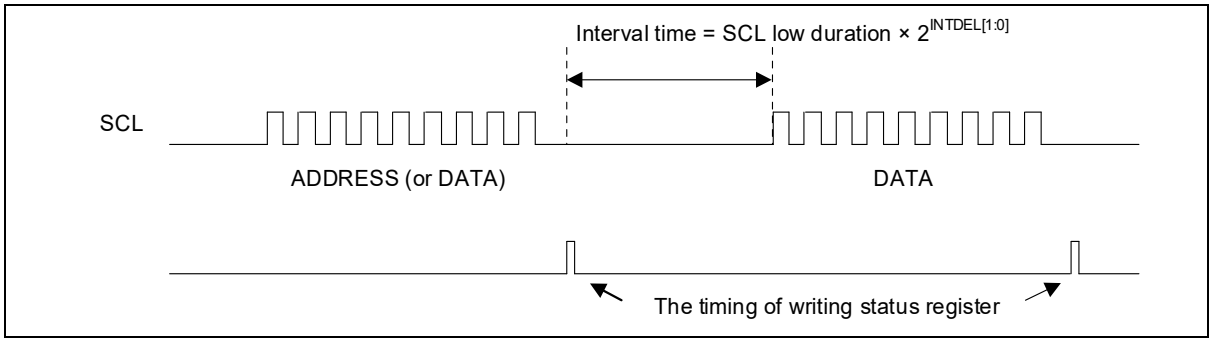
The I2Cn_CR register is a 32-bit read/write accessible register. It sets the operating modes and enablement of the I2C module.

I2C0_CR=0x4000_A014, I2C1_CR=0x4000_A114
I2C2_CR=0x4000_A214, I2C3_CR=0x4000_A314

31 30 29 28 27 26 25 24	23 22 21 20 19 18 17 16	15 14 13 12 11 10 9 8	7	6 5	4	3	2	1	0
Reserved			INTDEL[1:0]	IIF	Reserved	INTEN	ACKEN	Reserved	STOP	START
-			00	0	-	0	0	-	0	0
-			RW	RO	-	RW	RW	-	RW	RW

9 8	INTDEL[1:0]	Internal delay between address and data transfers (Or between two data transfers)
	0	1 × SCL low duration
	1	2 × SCL low duration
	2	4 × SCL low duration
	3	8 × SCL low duration
7	IIF	Interrupt flag
	0	No interrupt has occurred or the flagged interrupt has been cleared.
	1	An interrupt has occurred.
4	INTEN	Enable or disable interrupts
	0	Disables interrupts.
	1	Enables interrupts.
3	ACKEN	Enable ACK in receive mode
	0	No ACK signal is generated.
	1	ACK signal is generated.
1	STOP	Enable or disable stop If this bit is set to '1' in transmit mode, the next transmission is stopped even if an ACK signal has been received.
	0	Disables stop.
	1	Enables stop. Transmission is stopped when this bit is set to '1'.
0	START	Transmission starts in master mode
	0	The I2C waits in slave mode.
	1	The I2C starts transmitting in master mode.

Figure 132. Internal Delay (INTDEL[1:0]) in Master Mode



14.7.5 I2Cn_SCLL: I2C n SCL Low Duration Register

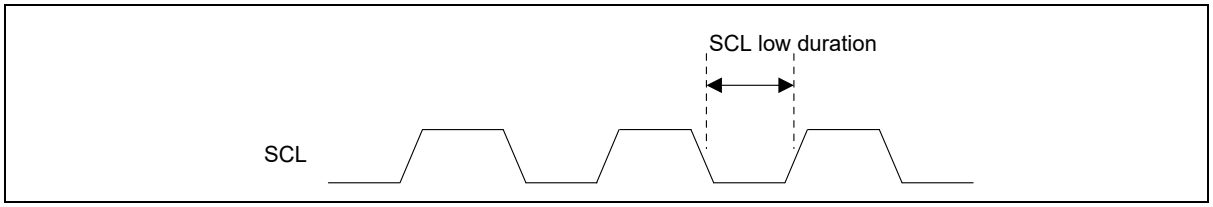
The I2Cn_SCLL register is a 32-bit register. The SCL low duration time can be set in master mode.

I2C0_SCLL=0x4000_A018, I2C1_SCLL=0x4000_A118
I2C2_SCLL=0x4000_A218, I2C3_SCLL=0x4000_A318

31 30 29 28 27 26 25 24	23 22 21 20 19 18 17 16	15 14 13 12 11 10 9 8	7 6 5 4 3 2 1 0
Reserved		SCLL[15:0]	
-		0xFFFF	
-		RW	

15 0	SCLL[15:0]	SCL low duration value SCL low duration = (PCLK period × SCLL[15:0]) + 2 × PCLK period Default value: 0xFFFF

Figure 133. SCL Low (SCLL[15:0]) Timing



14.7.6 I2Cn_SCLH: I2C n SCL High Duration Register

The I2Cn_SCLH register is a 32-bit register. The SCL high duration time can be set in master mode.

I2C0_SCLH=0x4000_A01C, I2C1_SCLH=0x4000_A11C
I2C2_SCLH=0x4000_A21C, I2C3_SCLH=0x4000_A31C

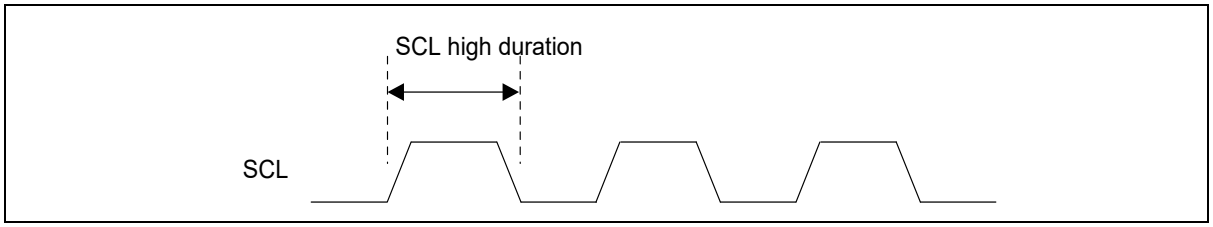
31 30 29 28 27 26 25 24	23 22 21 20 19 18 17 16	15 14 13 12 11 10 9 8	7 6 5 4 3 2 1 0
Reserved		SCLH[15:0]	
-		0xFFFF	
-		RW	

15 0	SCLH[15:0]	SCL high duration value SCL high duration = (PCLK period × SCLH[15:0]) + 3 × PCLK period Default value: 0xFFFF

NOTES:

1. In case of SCLH, a delay that counts the SCLH section may occur depending on external input conditions (Pull up resistance, VDDEXT).

Figure 134. SCL High (SCLH[15:0]) Timing



14.7.7 I2C_SDH: I2C n SDA Hold Register

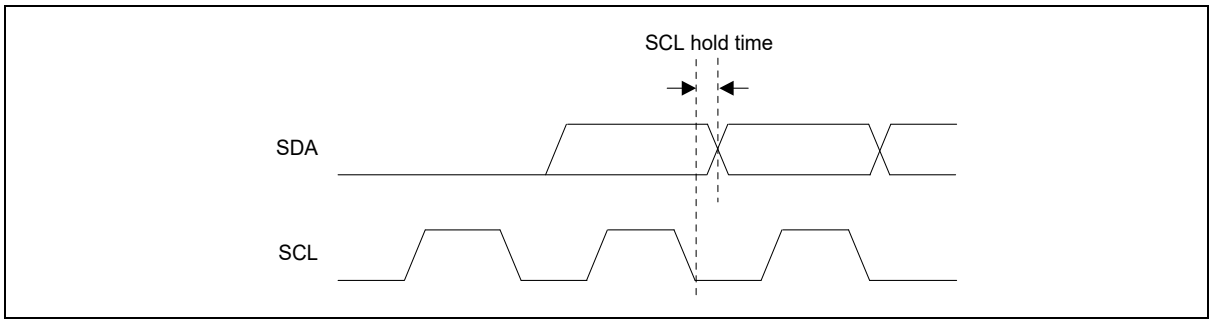
The I2C_SDH register is a 15-bit R/W register. The SCL hold time can be set in master mode.

I2C0_SDH=0x4000_A020, I2C1_SDH=0x4000_A120
I2C2_SDH=0x4000_A220, I2C3_SDH=0x4000_A320

31 30 29 28 27 26 25 24	23 22 21 20 19 18 17 16	15 14 13 12 11 10 9 8	7 6 5 4 3 2 1 0
Reserved		SDH[14:0]	
-		0x7FFF	
-		RW	

14 0	SDH[14:0]	SDA hold time value SDA hold time = (PCLK period × SDH[14:0]) + 4 × PCLK period Default value: 0x7FFF

Figure 135. SDA Hold (SDH[14:0]) Timing



14.7.8 I2C Register Map Summary

Table 129. I2C Register Map Summary

Offset	Register name	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0x00	I2Cn_DR	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	ICDR[7:0]							
	Reset value																									1	1	1	1	1	1	1	1
0x08	I2Cn_SR	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	GCALL	TEND	STOP	SSEL	MLOST	BUSY	TMOD	RXACK
	Reset value																									0	0	0	0	0	0	0	0
0x0C	I2Cn_SAR	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	SVAD[6:0]							GCEN
	Reset value																									0	0	0	0	0	0	0	0
0x14	I2Cn_CR	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	INTDEL[1:0]		IIF	Res.	Res.	INTEN	ACKEN	Res.	STOP	START
	Reset value																							0	0	0			0	0		0	0
0x18	I2Cn_SCLL	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	SCLL[15:0]															
	Reset value																	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1
0x1C	I2Cn_SCLH	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	SCLH[15:0]															
	Reset value																	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1
0x20	I2Cn_SDH	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	SDH[14:0]														
	Reset value																		1	1	1	1	1	1	1	1	1	1	1	1	1	1	1

15. Quad Serial Peripheral Interface (QSPI)

15.1 QSPI Introduction

The QSPI (Quad Serial Peripheral Interface) built in the A34G43x series is a specialized communication interface targeting single, dual or quad QSPI NOR flash memories. It can operate in any of the two following modes:

- Indirect mode: all the operations are performed using the QUADSPI registers.
- memory-mapped mode: the external flash memory is mapped to the device address space and is seen by the system as if it was an internal memory.

15.2 QSPI Main Features

The QSPI of A34G43x features the following:

QSPI Interface

- All x1, x2, x4 modes supported in a single instance
- Double data rate (DDR) supported without custom PHY blocks
- Flexible clocking schemes supporting high frequencies
- Only NOR flash chip supported

User Interface

- SDR(Normal/STR) and DDR(DTR) support
- Execute-in-place (XIP) supporting read-only flash chips
- Arbitrary memory reads and writes by user software supported with DMA transfer lists
- Arbitrary user command execution on QSPI chip
- Programming of flash chips

NOR flash chips are the simplest devices to operate though the controller.

The following general setup may be used:

- Set clock divider and related settings
- Set the appropriate read command and attributes (address, mode, etc.)
- Programming of the flash is performed using custom commands executed through the QSPI_CMDR register

15.3 QSPI Functional Description

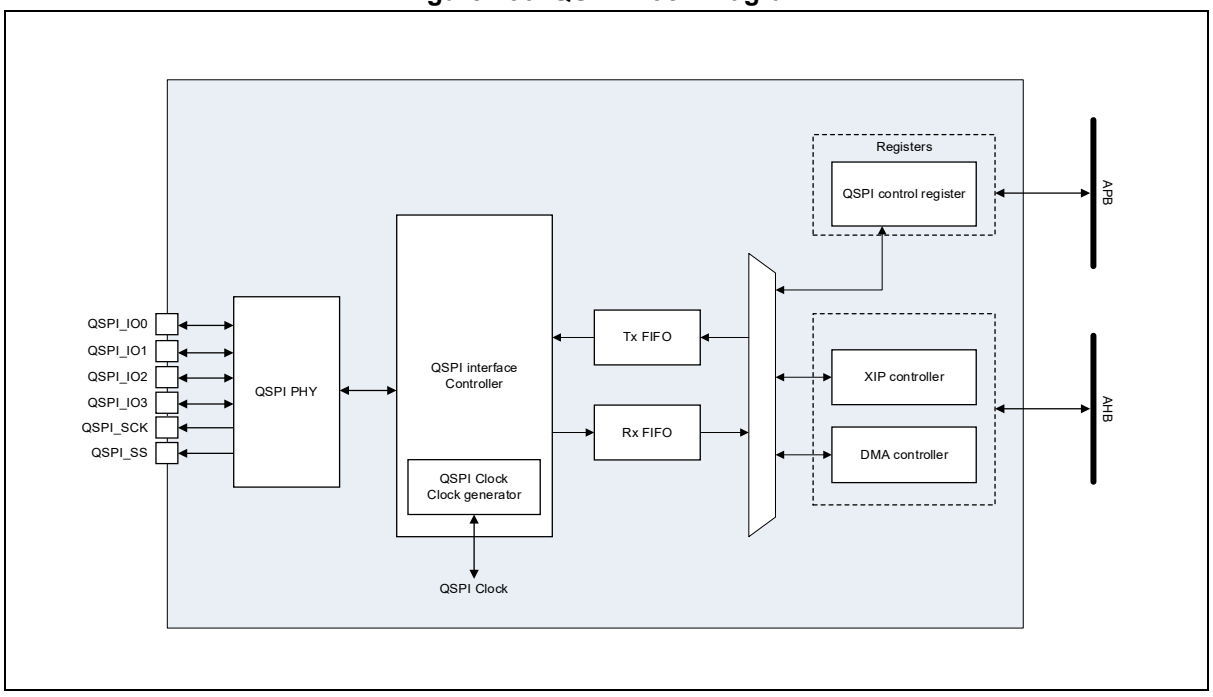
Figure 136 shows the high-level functional block diagram of the QSPI memory controller. The QSPI memory controller provides both a memory mapped view of the attached memory (execute-in place – XIP) and access to it via DMA features.

The QSPI DMA function is controlled through the QSPI register.

15.3.1 QSPI Block Diagram

Figure 136 shows a block diagram of the QSPI.

Figure 136. QSPI Block Diagram



15.3.2 QSPI Pins and Signals

Table 130 describes external pins assigned for QSPI.

Table 130. Pin Assignment of QSPI: External Pins

Pin Name	Type	Description	Supported Packages		
			A34G43xYL (120-LQFP)	A34G43xVL (100-LQFP)	A34G43xRL (64-LQFP)
QSPI_IO0	IO	QSPI channel 0 input/output 0	O	O	O
QSPI_IO1	IO	QSPI channel 0 input/output 1	O	O	O
QSPI_IO2	IO	QSPI channel 0 input/output 2	O	O	O
QSPI_IO3	IO	QSPI channel 0 input/output 3	O	O	O
QSPI_SCK	IO	QSPI channel 0 clock input/output	O	O	O
QSPI_SS	IO	QSPI channel 0 select signal input/output	O	O	O

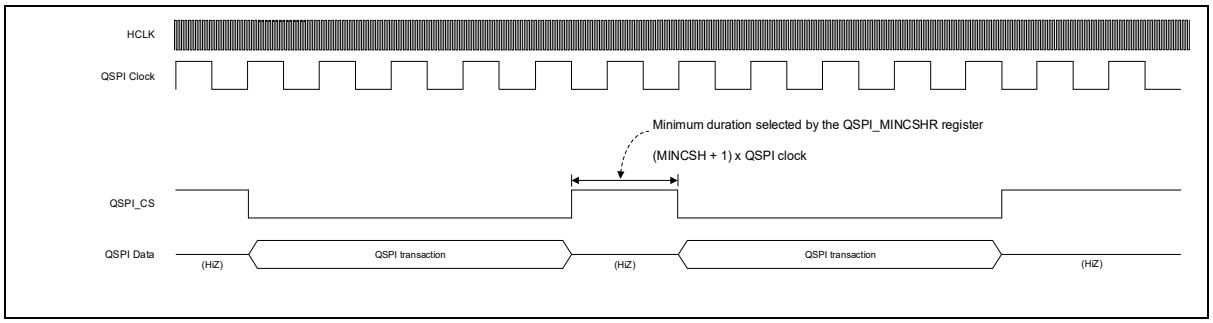
NOTES:

1. When the MODE[3:0] bits in the attribute register of each phase are set to single mode, it is compatible with SPI, allowing QSPI_IO0 to function as MOSI and QSPI_IO1 as MISO.
2. Regarding PCB design, the flash chip should be placed as close as possible to the main chip, as the signal propagation distance is effectively doubled (once for the clock to reach the flash and once for the data to return). A reasonable placement distance is 5 to 10 cm, which results in an additional input delay of approximately 0.6 to 1.2 ns.

15.3.3 QSPI Transactions

The QSPI controller performs all operations (memory read, register write, etc.) as individual QSPI transactions each of which is always a predetermined number of clocks. Between QSPI transactions the QSPI memory controller will de-assert all chip select lines as given in Figure 137 below.

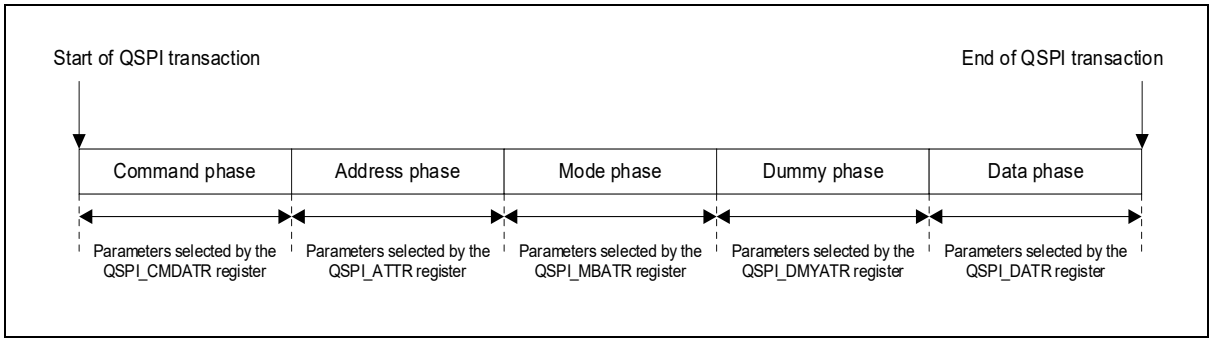
Figure 137. Transaction Level Operation of QSPI Memory Controller



Each QSPI transaction is made up of five individual phases (Command, Address, Mode, Dummy, Data) as shown in Figure 138 below, each of which may or may not be present. The parameters used for each phase are configured for each phase in individual registers. The parameters are as follows:

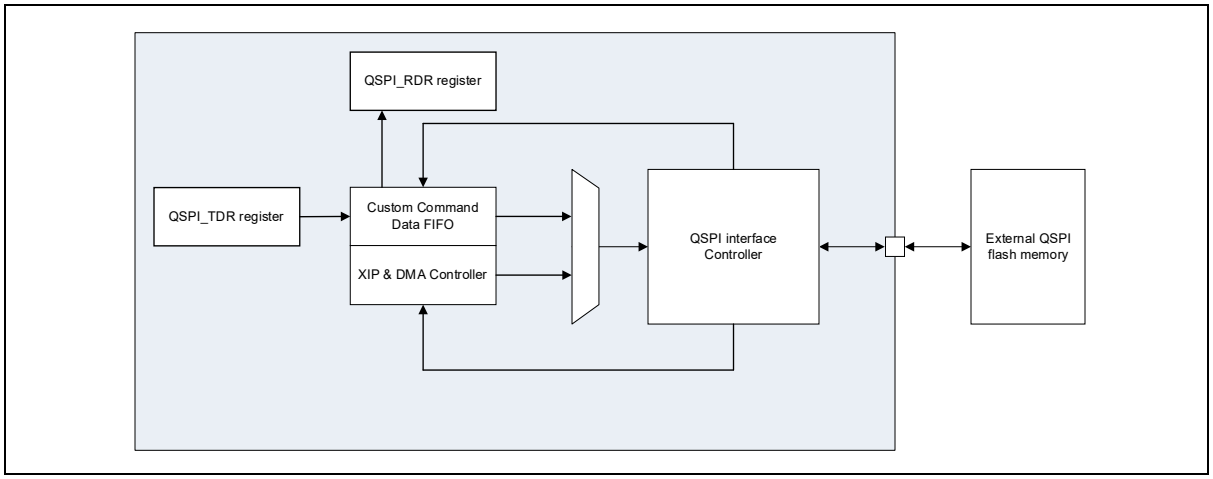
- The phase is present or not. (Use the SKIP bit in each attribute register.)
- Parallel bits to use (x1, x2 or x4)
- Clock edges to use (DDR/DTR or Normal/STR)
- Number of clock cycles used for the phase
- A QSPI transaction is transmitted only after writing a value to the Command register.

Figure 138. Individual Phases of QSPI Transaction



The data in the data phase may come from or go to a number of sources or destinations, as given in Figure 139 below. Custom command accesses and XIP accesses are seamlessly interleaved by the controller. However, care must be taken if custom commands would change the meaning of the XIP commands (e.g., writes to status register in many flash chips).

Figure 139. Data Flows in QSPI Memory Controller during Data Phase of QSPI Transaction



15.3.3.1 QSPI Output Behavior

The QSPI memory controller has various settings which control the operation of the QSPI clock output. The first setting to decide is the output clock's polarity which may be selected using CPOL bit in the QSPI_CLKCFGR register. The TRANS bit in the QSPI_CLKCFGR register must be '0'. The timing of data changes occurs at the negative (falling) edge of the output clock, and the positive (rising) edge is not supported. The traditional method to drive QSPI is to change the output data on the opposite edge on which the chip samples; however more setup time is afforded if the same edge is selected, but whether that is feasible depends on the implementation. Figure 140 and Figure 141 below explain how these two settings affect each other.

Figure 140. QSPI Clock Output Behavior (TRANS = '0', CPOL: '0')

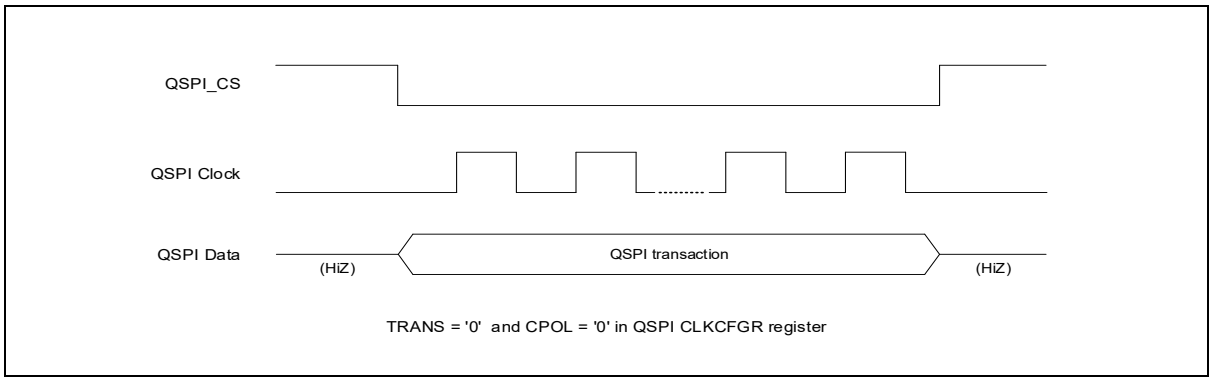
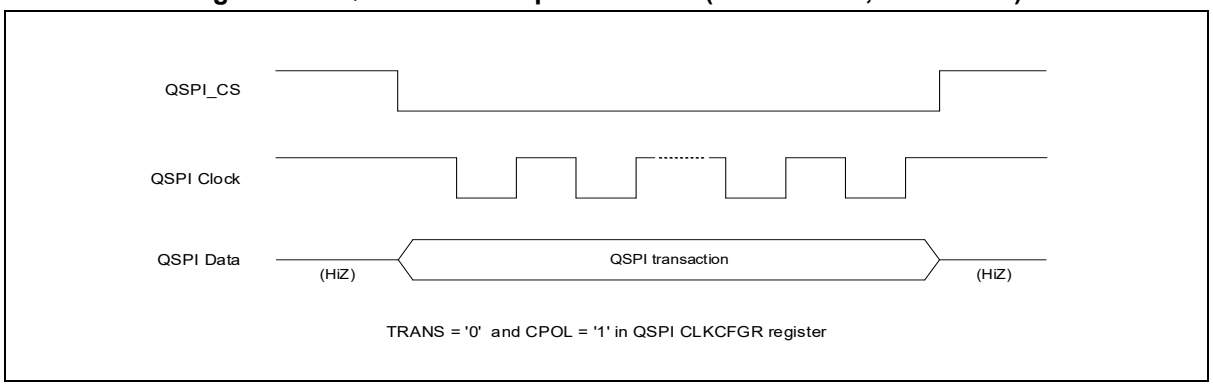
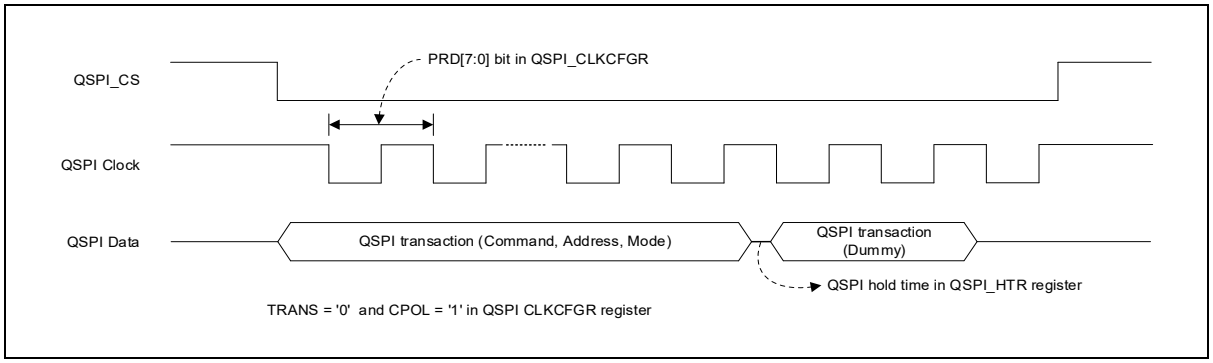


Figure 141. QSPI Clock Output Behavior (TRANS = '0', CPOL = '1')



At the end of each transaction or the beginning of the dummy or data phase for reads, the QSPI memory controller will tristate its outputs at the time programmed in the QSPI_HTR register (see Figure 142). It is usually safe if the hold time is programmed to be one less than half of the clock period setting for divide-by-4 and higher modes. For the divide-by-2 mode safe is to use at least a single dummy bit.

Figure 142. Output Operation of QSPI Memory Controller in Normal (STR) Modes

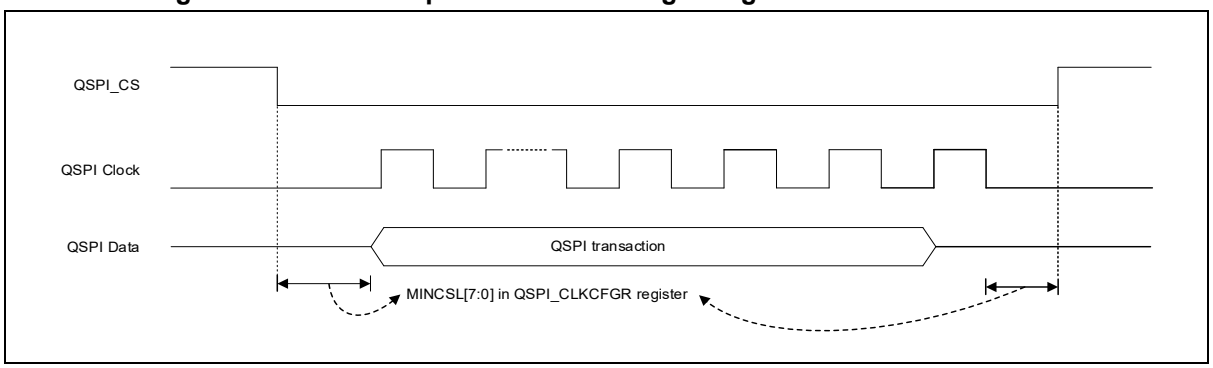


NOTE:

1. The optimum values here depend on the implementation.

The time between CS activation and the first clock edge is given using MINCSL[7:0] bits in the QSPI_CLKCFGR register. The same value is used for the time between the last clock edge and CS de-assertion, as given in Figure 143 below.

Figure 143. Clock Output Behavior at Beginning and End of Transaction



NOTE:

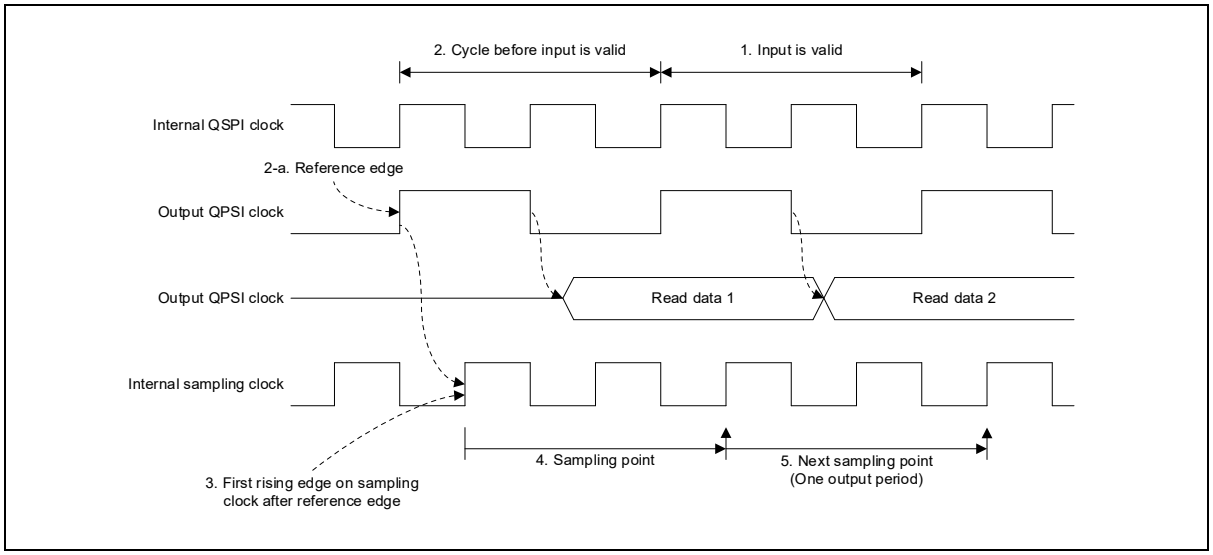
1. For Single SPI, it is MIN_CS_LOW (MINCSL) + SAMPLING_POINT (SP).

15.3.3.2 QSPI Input Behavior

The controller will sample the input data using the algorithm given below. Refer to Figure 144 below for examples.

1. Identify the output QSPI clock cycle on which the input data is valid
2. Identify the output QSPI clock cycle before the input data is valid
   A. The first output clock edge in this cycle is the reference edge
3. From the reference edge identify the first rising edge on the sampling clock
4. Count “sampling point” number of clock cycles (on the internal sampling clock) forward from the edge identified in point 3. At this edge, the controller will make a sample. The sampling point is given in SP[7:0] bits of the QSPI_CLKCFGR register. The sampling point must be configured such that the sampling point is at most four output QSPI clock cycles from the reference edge.
5. The following sample will be made either 1 QSPI output cycle or ½ QSPI output cycle from the edge identified in point 4 above depending on whether the controller is configured for STR or DTR input mode.

Figure 144. Input Sampling Behavior of Controller when Sampling Clock is 180 Degrees Out of Phase with Internal Clock Generating Output Clock



15.3.3.3 Use of QSPI Data Lines

In supporting multiple modes with a single controller, the QSPI memory controller drives and uses the possible four I/O lines in the manner given in Table 131 below:

Table 131. I/O Behavior of Memory Controller in Different Modes

Mode	Transaction phase	QSPI_IO0	QSPI_IO1	QSPI_IO2	QSPI_IO3
IDLE	N/A	HiZ	HiZ	Driven WP[1)]	HiZ
Single (x1)	Read data and dummy phases	Driven	HiZ / sampled	Driven WP[1)]	Driven 1[2)]
	All others	Driven	HiZ	Driven WP[1)]	Driven 1[2)]
Dual (x2)	Read data and dummy phases	HiZ / sampled	HiZ / sampled	Driven WP[1)]	Driven 1[2)]
	All others	Driven	Driven	Driven WP[1)]	Driven 1[2)]
Quad (x4)	Read data and dummy phases	HiZ / sampled	HiZ / sampled	HiZ / sampled	HiZ / sampled
	All others	Driven	Driven	Driven	Driven

NOTES:

1. Driven with the value in the QSPI_WPCR register.
2. This is usually the HOLD pin on x1 and x2 modes. Driven 1 means output high.

15.3.3.4 Command (Instruction) Phase

In the command phase, a command to be transmitted to the external flash memory can be set. The command can be configured as a 16-bit instruction in total; however, most flash memory devices use 8-bit commands and receive them in single mode using QSPI_IO0 (MOSI), accepting one bit per clock cycle.

Optionally, the command phase can use Dual QSPI (QSPI_IO0, QSPI_IO1) to send 2 bits per clock cycle or Quad QSPI (QSPI_IO0, QSPI_IO1, QSPI_IO2, QSPI_IO3) to send 4 bits per clock cycle. The command phase can be skipped by using the SKIP bit in the QSPI_CMDATR register, proceeding directly to the next transaction.

The QSPI_CMDATR register allows the configuration of the mode (Single, Dual, Quad), DDR mode (Normal, DDR), and duration (QSPI output clock length).

15.3.3.5 Address Phase

In the address phase, 1 to 4 bytes are sent to the flash memory to specify the operation address. The address phase can optionally use Single QSPI (QSPI_IO0/MOSI) for transmitting 1 bit per clock cycle, Dual QSPI (QSPI_IO0, QSPI_IO1) for transmitting 2 bits per clock cycle, or Quad QSPI (QSPI_IO0, QSPI_IO1, QSPI_IO2, QSPI_IO3) for transmitting 4 bits per clock cycle. The address phase can be skipped by using the SKIP bit in the QSPI_AATR register, after which the next transaction is executed.

In Indirect mode, the address bytes to be transmitted are written to QSPI_AR, while in memory-mapped mode (XIP), the address is directly provided through the AHB. For memory-mapped mode address configuration, refer to section 15.4.4.

The QSPI_AATR register allows configuration of the mode (Single, Dual, Quad), DDR mode (Normal, DDR), and duration (QSPI output clock length).

15.3.3.6 Mode Phase

In the mode phase, 1 to 4 bytes are sent to the flash memory to specify the operation address. The mode phase can optionally use Single QSPI (QSPI_IO0/MOSI) to send 1 bit per clock, Dual QSPI (QSPI_IO0, QSPI_IO1) to send 2 bits per clock, or Quad QSPI (QSPI_IO0, QSPI_IO1, QSPI_IO2, QSPI_IO3) to send 4 bits per clock. The mode phase can be skipped by using the SKIP bit in the QSPI_MBATR register, and the next transaction will be executed.

The QSPI_MBATR register allows configuration of the mode (Single, Dual, Quad), DDR mode (Normal, DDR), and duration (QSPI output clock length).

15.3.3.7 Dummy Phase

In the dummy phase, 1 to 32 clock cycles are provided without any data being transmitted or received, allowing the flash memory sufficient time to prepare for the data phase when higher clock frequencies are used. Unlike other phases, the dummy phase allows only the configuration of the number of clock cycles, without setting any data values. The specified number of clock cycles is output through the QSPI output clock.

The dummy phase can be skipped by using the SKIP bit in the QSPI_DMYATR register, after which the next transaction is executed. Additionally, the behavior of the dummy phase varies depending on the DIR bit in the QSPI_DATR register. The DIR bit determines whether the QSPI interface operates in data output mode (writing to QSPI memory) or data input mode (reading from QSPI memory).

The table below summarizes the dummy phase settings based on data output and input modes.

NOTE:

1. If DIR in the QSPI_DATR register is '1', it is used for the read latency.

Table 132. Dummy Phase Parameter Determination

Parameter	Used for read latency
Dummy phase present	If SKIP bit in QSPI_DMYATR register is '0', dummy phase skipped.
Dummy phase duration	DUR[4:0] in QSPI_DMYATR register
Value of data lines	Hi-Z
First cycle of dummy	Cycle following command, address or mode phase whichever comes last.
Dummy length double	Not support

15.3.3.8 Data Phase

During the data phase, any number of bytes can be sent to, or received from the flash memory. In indirect, the number of bytes to be sent/received is specified in the QSPI_DSZR register.

In indirect-write mode the data to be sent to the flash memory must be written to the QSPI_TDR register. In indirect-read mode the data received from the flash memory is obtained by reading the QSPI_RDR register. In memory-mapped mode (XIP), the data which is read is sent back directly over the AHB to the Cortex or to a DMA.

The data phase can optionally use Single QSPI (QSPI_IO0/MOSI) to transmit 1 bit per clock cycle, Dual QSPI (QSPI_IO0, QSPI_IO1) to transmit 2 bits, or Quad QSPI (QSPI_IO0, QSPI_IO1, QSPI_IO2, QSPI_IO3) to transmit 4 bits. The data phase can be skipped by using the SKIP bit in the QSPI_DATR register. Since there is no subsequent transaction, communication is completed by maintaining the state for the duration specified by the MINCSL[7:0] bits in the QSPI_CLKCFGR register and then setting QSPI_CSR to “high”. The MINCSL[7:0] bits are configured based on the QSPI clock.

The QSPI_DATR register allows configuration of the mode (Single, Dual, Quad), DDR mode (Normal, DDR), and duration (QSPI output clock length).

15.3.3.9 Setting of Duration

The duration of the command, address, mode, and data phases in a QSPI transaction, excluding the dummy phase, is calculated based on the bit count and mode as shown in Table 133. For the dummy phase, the duration should be set according to the required number of QSPI clock cycles.

Example 1

If the address phase is 24-bit, DDR mode, and Quad mode, the calculation is as follows:

$$\{24\ (bit) / 2\ (DDR) / 4\ (QUAD)\} - 1 = 2$$

In this case, the address phase is output with 3 QSPI clock cycles.

Example 2

If the dummy phase requires 3 QSPI clock cycles, the read duration value should be set to 2, while the write duration should be set to 3.

Table 133. Example of Duration

BIT	DDR Mode	MODE [3:0]	Duration [3:0]
8	Normal (STR)	Single (x1, 1-bits)	7
8	Normal (STR)	Dual (x2, 2-bits)	3
8	Normal (STR)	Quad (x4, 4-bits)	1
8	DDR (DTR)	Single (x1, 1-bits)	3
8	DDR (DTR)	Dual (x2, 2-bits)	1
8	DDR (DTR)	Quad (x4, 4-bits)	0
24	Normal (STR)	Single (x1, 1-bits)	23
24	Normal (STR)	Dual (x2, 2-bits)	11
24	Normal (STR)	Quad (x4, 4-bits)	5
24	DDR (DTR)	Single (x1, 1-bits)	11
24	DDR (DTR)	Dual (x2, 2-bits)	5
24	DDR (DTR)	Quad (x4, 4-bits)	2

NOTE:

1. The duration setting value must be written as the actual duration value minus one.

15.3.4 QSPI Interface Protocol Modes

This section describes the modes supported by QSPI control. Each phase (command, address, mode, data) must be configured according to the selected mode. The dummy phase does not require a mode setting.

15.3.4.1 Single QSPI Mode

The legacy QSPI mode allows only a single bit to be transmitted and received serially. In this mode, data is transmitted to the flash memory through QSPI_IO0 (MOSI). Data sent from the flash memory is received through QSPI_IO1 (MISO).

To configure the phase (command, address, mode, data) for Single QSPI mode, the MODE[3:0] bits in each attributes register must be set to '1' (Single mode).

15.3.4.2 Dual QSPI Mode

In Dual-QSPI mode, two bits are transmitted and received simultaneously over QSPI_IO0 and QSPI_IO1.

To configure the phases (command, address, mode, and data) for Dual-QSPI mode, the MODE[3:0] bits in each attributes register must be set to ‘2’ (Dual mode).

15.3.4.3 Quad QSPI Mode

In Dual-QSPI mode, two bits are transmitted and received simultaneously over QSPI_IO0, QSPI_IO1, QSPI_IO2, and QSPI_IO3.

To configure the phases (command, address, mode, data) for Dual-QSPI mode, the MODE[3:0] bits in each attributes register set to ‘2’ (Dual mode).

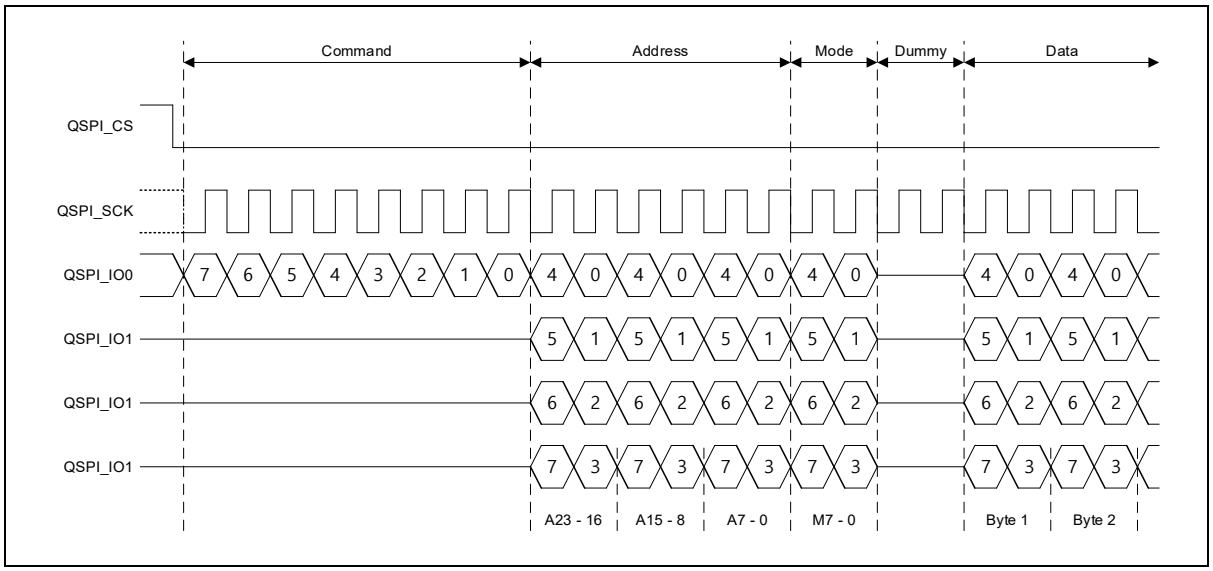
QSPI_IO2 and QSPI_IO3 are used only in Quad QSPI mode. If no phase is configured to use Quad QSPI mode, QSPI_IO2 and QSPI_IO3 can be used for other functions. Refer to Table 131 for more information.

15.3.4.4 SDR Mode

QSPI operates in Single Data Rate (SDR) mode by default, and its configuration can be modified in the attributes register of each phase (command, address, mode, data) except for the dummy phase. The dummy phase always operates in SDR mode.

In SDR mode, QSPI_IO0, QSPI_IO1, QSPI_IO2, and QSPI_IO3 transition data only on the negative (falling) edge of the QSPI clock, while data is sampled on the positive (rising) edge.

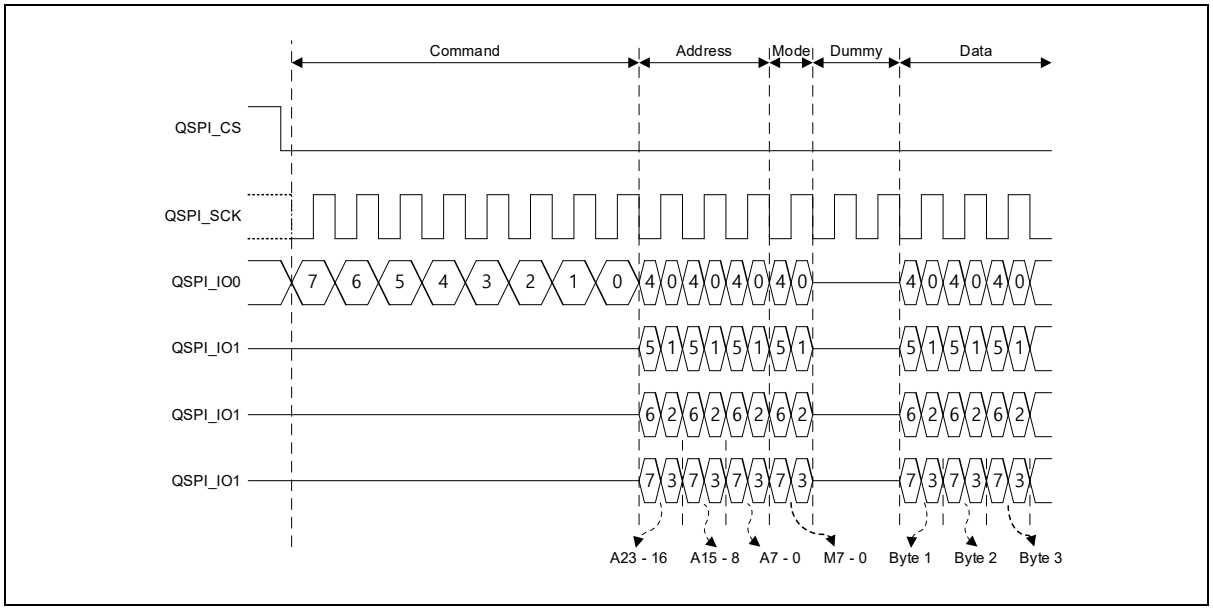
Figure 145. Example of SDR Command in QSPI Mode



15.3.4.5 DDR Mode

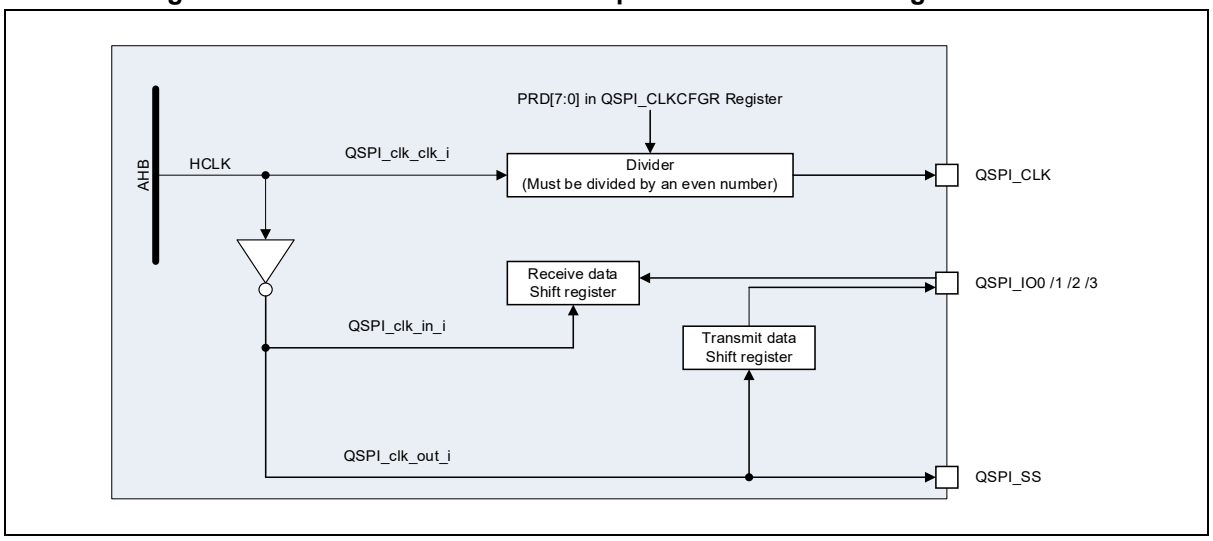
The Double Data Rate (DDR) mode can be configured in the attributes register of each phase (command, address, mode, data), except for the dummy phase. In DDR mode, QSPI_IO0, QSPI_IO1, QSPI_IO2, and QSPI_IO3 transition and transmit data on both the rising and falling edges of the QSPI clock. The data received in DDR mode is sampled at the half-cycle of the QSPI clock.

Figure 146. Example of Read Command in QSPI Mode



The following figure shows the recommended division ratio for using DDR mode. When using DDR mode, it is recommended to set the QSPI clock divide value to an even number, at least ¼ or higher (1/4, 1/6, …). Similarly, when using SDR mode, it is also recommended to divide the QSPI clock by an even number (1/4, 1/6, …). Refer to Figure 147.

Figure 147. Recommended Clock Output Division when Using DDR Mode



15.3.5 QSPI Transaction Execution

Before executing a custom QSPI command, the controller needs to be set up with the appropriate parameters detailing how the command is to be executed. This setup is required for both DMA commands and commands executed by writing the QSPI_CMDR register.

15.3.5.1 Executing Transactions Using Registers

For this mode of command execution, data for the data phase comes from the custom command data FIFO. Once the transaction parameters have been set up, the QSPI transaction may be executed as follows:

1. Ensure that DIR bit is set to ‘0’ for writing transactions and to ‘1’ for reading transactions in the QSPI_DATR register.
2. Write the command to execute to the QSPI_CMDR register, starting the command.
3. In case of writing transactions with data phase fill the FIFO with data
   A. The FIFO is written by writing to the QSPI_TDR register. The FIFO may be filled:
      i. Using a copy routine running on a CPU.
      ii. The FIFO may be filled using an external DMA engine.
4. In case of read transactions the availability of read data is signaled by FILL bit in the QSPI_FSR register and by Read FIFO Status FRER bit in the QSPI_SR register.
5. Command execution complete is signaled by the BUSY bit being set to ‘0’ in the QSPI_SR register.
   A. The busy bit may be monitored either by polling the register.
   B. Enabling the “not-busy” interrupt using NBUSYIE bit of the QSPI_ICR register.

The amount of data transferred can be larger than the FIFO size. If the FIFO is empty (for writes) or full (for reads) the controller will throttle the requests until the FIFO is written or read. The FIFO may be read or written any time there is data or space available.

15.3.5.2 Operation of the Interrupts and Status

The operation of interrupts and status is shown in the following figures:

Figure 148 describes the Busy status, where the BUSY bit in the QSPI_SR register indicates the current state. The Busy status is read-only, and the interrupt setting for the not busy state can be configured using the NBUSYIE bit in the QSPI_ICR register.

Figure 148. Busy Status/Interrupt Timing Diagram

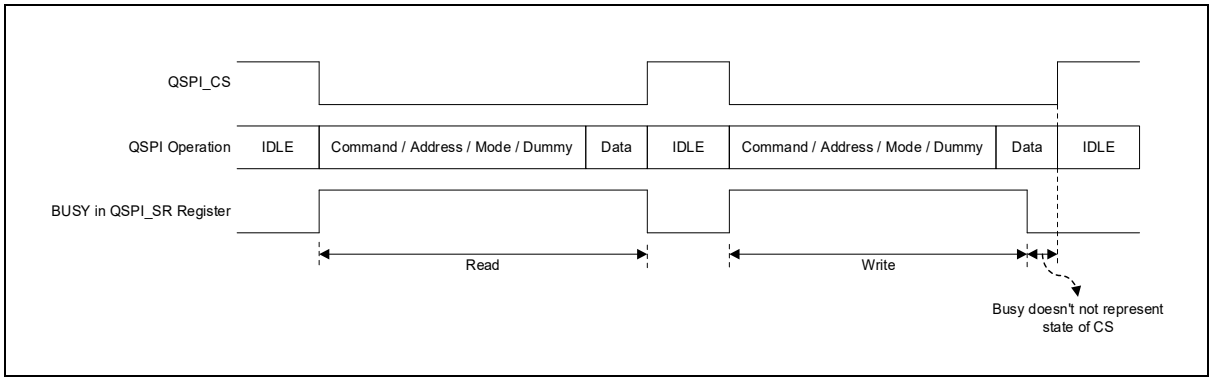


Figure 149 shows the Write FIFO status, where the FWER bit in the QSPI_SR register indicates the current state. The Write FIFO status is read-only and is determined by the value of the FBND[2:0] bits in the QSPI_ICR register.

Figure 149. Write FIFO Status/Interrupt Timing Diagram

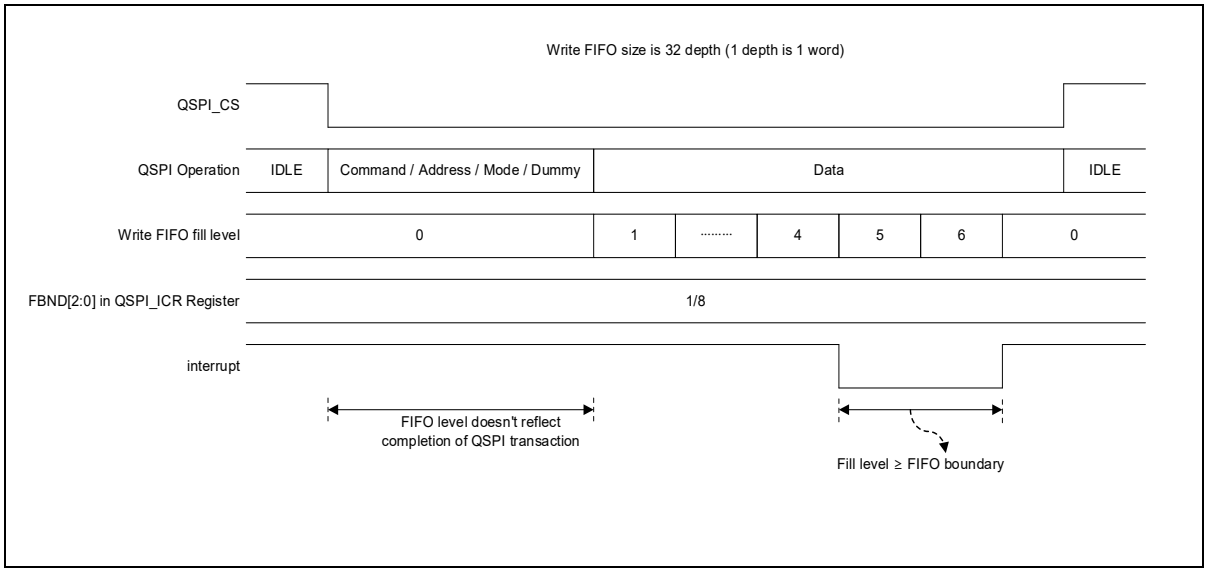


Figure 150 describes the Write FIFO error, which can be checked through the FWERR bit in the QSPI_SR register. The Write FIFO error flag is cleared by writing '1', and the interrupt configuration can be set using the FWERRIE bit in the QSPI_ICR register.

Figure 150. Write FIFO Error Status/Interrupt Timing Diagram

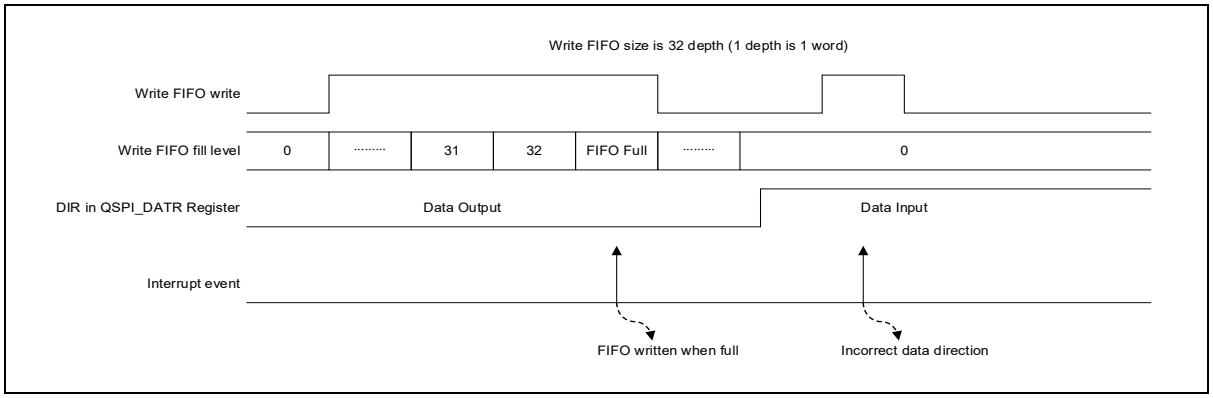


Figure 151 describes the Read FIFO status, which can be checked through the FRER bit in the QSPI_SR register. The Read FIFO status is read-only, and its status is determined based on the FBND[2:0] bits in the QSPI_ICR register.

Figure 151. Read FIFO Status/Interrupt Timing Diagram

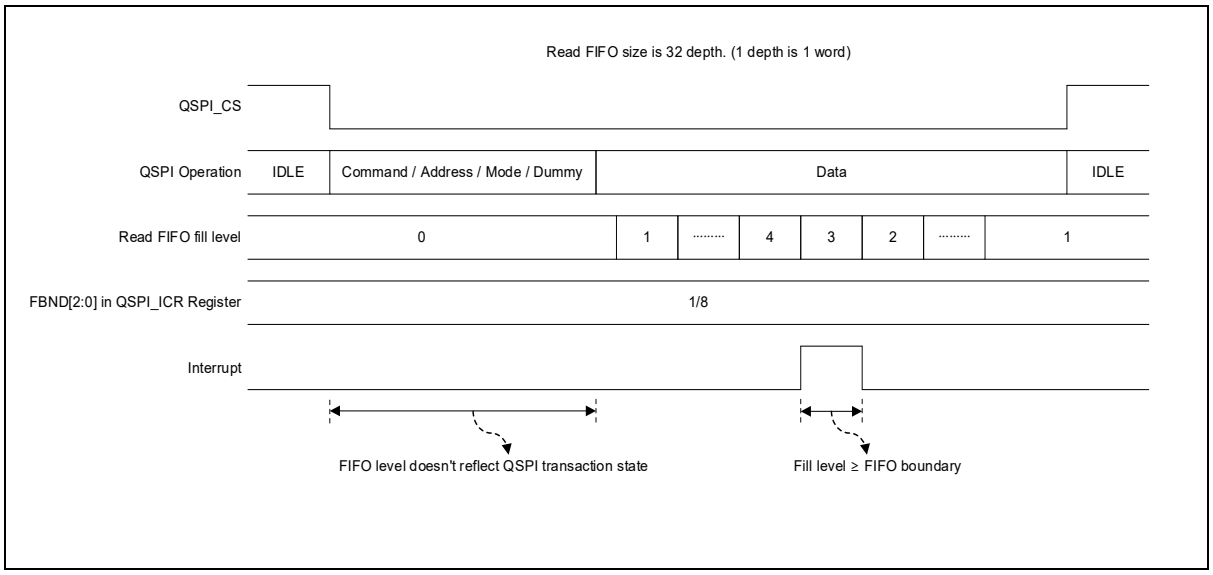
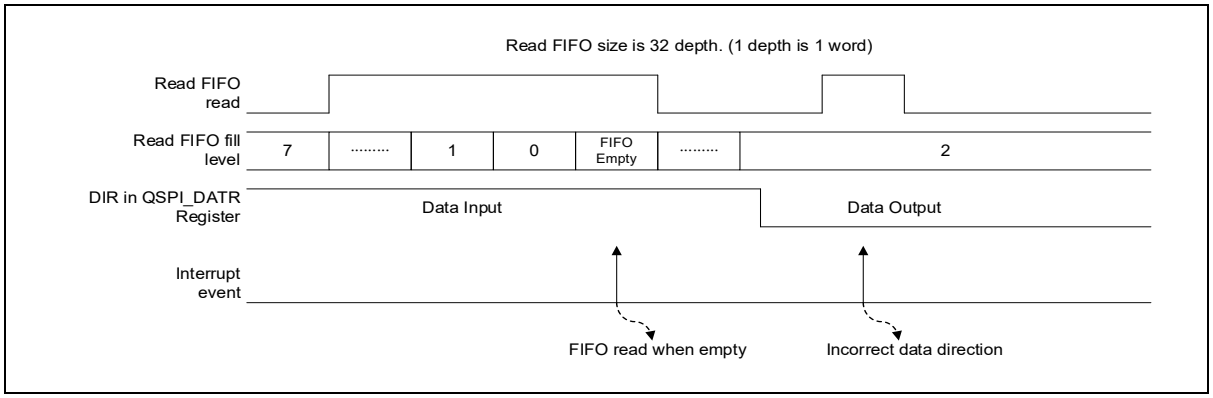


Figure 152 describes the Read FIFO error, which can be checked through the FRERR bit in the QSPI_SR register. The Read FIFO error flag is cleared by writing '1', and the interrupt settings can be configured using the FRERRIE bit in the QSPI_ICR register.

Figure 152. Read FIFO Error Status/Interrupt Timing Diagram



15.3.5.3 Transactions with Data from Internal DMA Engine

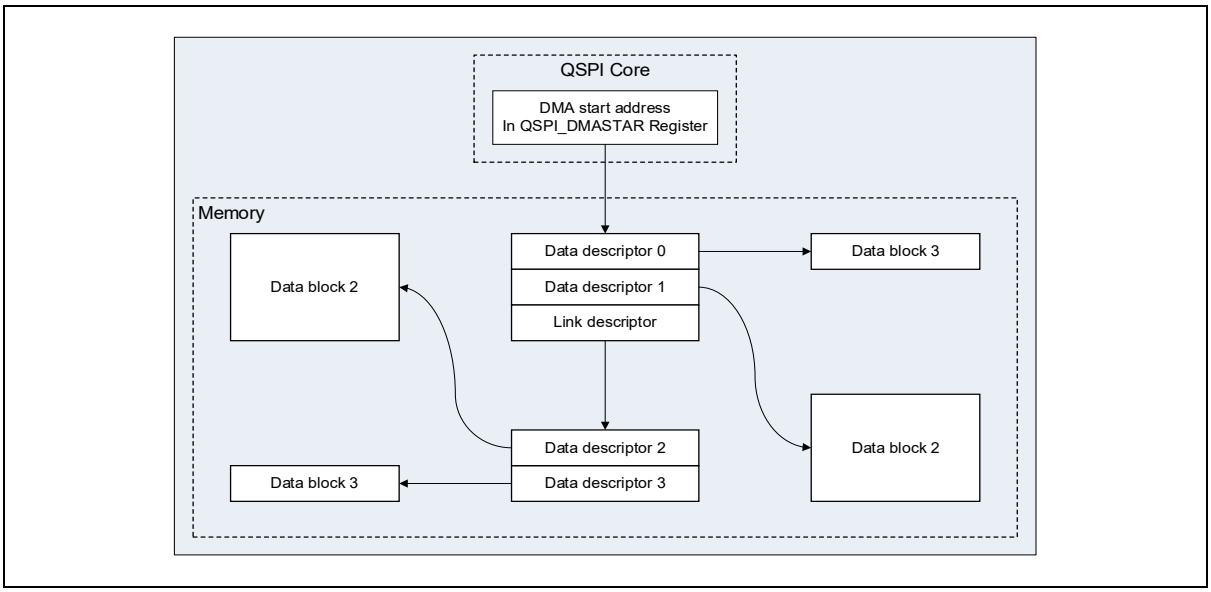
Once the transaction parameters have been set up, the transaction may be executed as follows:

1. Ensure that SKIP bit is set to '0' in the QSPI_DATR register.
2. The QSPI_DSZR register must be programmed with the total size of the DMA's data.
   A. If the programmed data size is not the sum of the sizes in the DMA transaction list (see section 15.3.6), the DMA will finish with an error signaled in DERR bit of the QSPI_DMASR register.
3. Setup the DMA descriptor lists in memory as per section 15.3.6.
4. Write the address of the first descriptor to the QSPI_DMASTAR register.
5. Enable DMA using DMAEN bit in the QSPI_GCR register.
   A. At this point, requests are issued to both the system memory and the QSPI memory.
6. Write the command to execute to the QSPI_CMDR register.
7. Command execution completed is signaled by the BUSY bit set to '0' in the QSPI_SR register.
   A. The busy bit may be monitored either by polling the register.
   B. Enabling the "not-busy" interrupt using NBUSYIE bit of the QSPI_ICR register.

15.3.6 QSPI DMA Mode of Operation

In the DMA mode of operation, the QSPI memory controller reads a linked list and data from memory and executes them as specified. Both read and write commands are supported to be executed through DMA. Figure 153 below shows the high-level structure of the DMA memory layout. The starting point of the DMA commands is provided in the QSPI_DMASTAR register.

Figure 153. DMA Buffer Descriptor List Example



Before starting DMA the host software needs to set up the appropriate structures in memory structure and write the starting address to the QSPI_DMASTAR register before starting DMA enabled QSPI command.

The memory structure is primarily a list of buffer descriptors, placed in memory one after the other. The only exception is the link descriptor which can be used to create non-linear lists of buffer descriptors (e.g., as in Figure 153). Each buffer descriptor is made up of two 32-bit words the format of which is given in Table 134 and the meaning of the fields is given in Table 135. Processing of buffer descriptors stops when a descriptor has the end bit set to ‘1’.

Table 134. DMA Buffer Descriptor Layout

First word

31 30 29 28 27 26 25 24	23 22 21 20 19 18 17 16	15 14 13 12 11 10 9 8	7 6	5 4	3	2	1	0
SIZE[15:0]		Reserved		TYPE[1:0]	Reserved	IRQ	END	VALID
0x0000		-		00	0	0	0	0
RW		-		RW	-	RW	RW	RW

Second word

31 30 29 28 27 26 25 24	23 22 21 20 19 18 17 16	15 14 13 12 11 10 9 8	7 6	5 4	3	2	1	0
ADDR[29:0]								Reserved
00000000000000000000000000000000								-
RW								-

Table 135. DMA Buffer Descriptor Contents

Field	Description
ADDR[29:0]	Pointer to start of data block (Data Descriptor) or next buffer descriptor (Link Descriptor).
SIZE[15:0]	Only used for Data Type Buffer Descriptors. It indicates the size of allocated memory block in 4-byte words. Size in bytes is therefore Size *4. Size of all Data Buffers used for transfers is summed and checked against QSPI_DSZR register. If word counts differ, DMA reports an error.
TYPE[1:0]	01: Link As suggested by its name, it does not point to data buffer. Instead, it points to next buffer descriptor location. This is used for flexible memory allocation, where large contiguous memory blocks are hard to obtain and buffer descriptors are scattered. Size is not used. 10: Data Points to start of allocated block of memory used for data storage. 00, 11: Invalid and cause DMA stop with error.
IRQ	'1' indicates interrupt request should be triggered after successful completion of this buffer.
END	'1' indicates final buffer in a list. DMA completes after it successfully transfers buffer marked as final.
VALID	Must be '1' and is used for runaway protection. If '0' is encountered, DMA stops and signals an error.

Operation of the DMA interrupts are given in the figures below as follows:

Figure 154 describes the DMA idle status and the DMA successfully transmitted flag, which can be checked in the DIDLE bit and DIRQ bit of the QSPI_DMASR register. The DIDLE bit is read-only, while the DIRQ bit is cleared by writing '1'. Interrupt settings can be configured using the DIDLEIE bit and DIE bit in the QSPI_ICR register.

Figure 154. DMA Idle and IRQ Status Timing Diagram

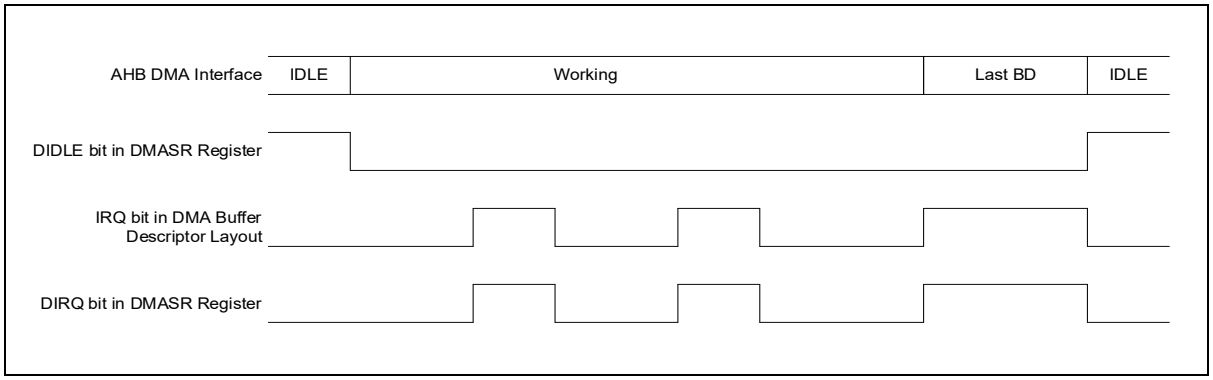
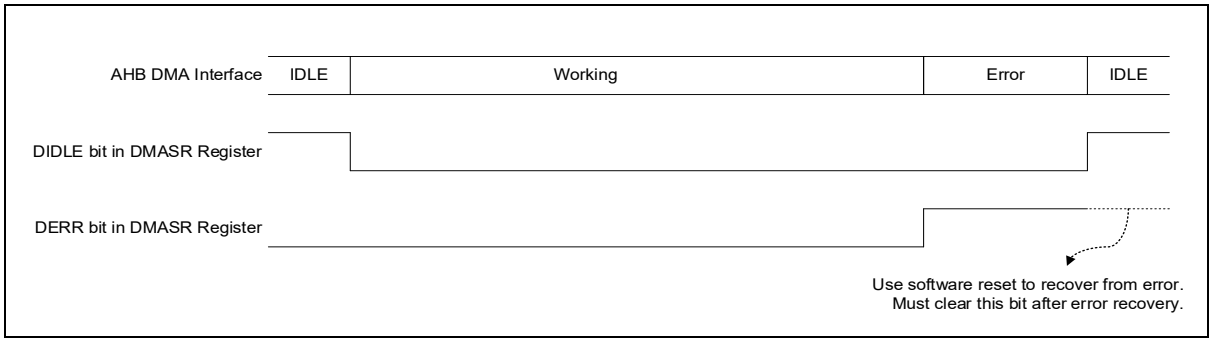


Figure 155 describes the DMA error status, which can be checked through the DERR bit in the QSPI_DMASR register. To recover from a DMA error, a reset is required, and the DERR bit must be cleared after error recovery. The reset can be performed using the SWRST bit in the QSPI_GCR register. The DERR bit is cleared by writing '1', and interrupt settings can be configured using the DERRIE bit in the QSPI_ICR register.

Figure 155. DMA Error Status Timing Diagram

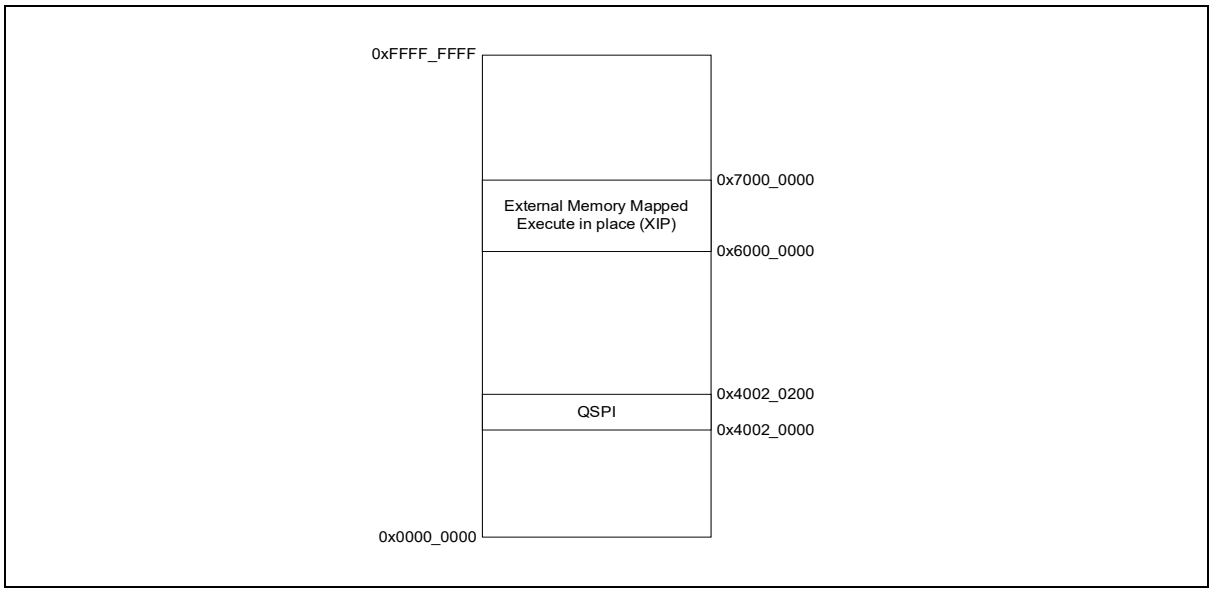


15.4 Memory Map

15.4.1 External Bus Space

The locations of a serial flash memory and control register on the address space are determined by the address range of the area set in the configuration.

Figure 156. XIP Memory Map



15.4.2 Address Width of QSPI Space and QSPI Bus

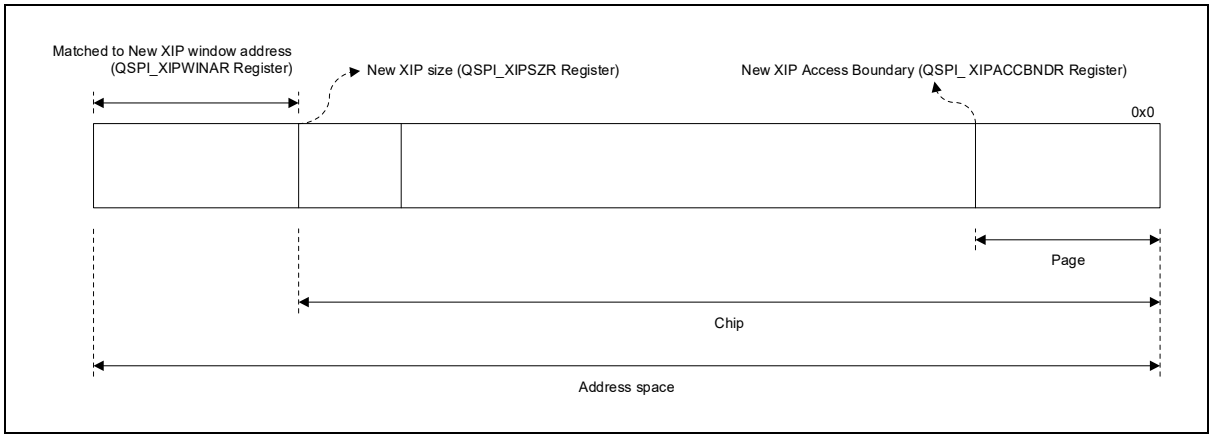
QSPI has a 32-bit address width for referencing serial flash memory. When accessing the QSPI space for reading, the QSPI bus cycle is automatically initiated, and the data read from the serial flash memory is returned. The address width of the QSPI space is fixed to 32 bits, and the data read from the serial flash memory is mapped to the memory-mapped region.

15.4.3 Address Split

The XIP address can be split into windows. Each window accesses the external device with a completely separate set of parameters (read command, modes, chip select value, etc.). This mechanism allows to externally attach multiple different devices with the controller selecting the used parameters based on the address. Each window has a size and base address to determine when to use a window if the following evaluates to true:

New XIP window address ≤ XIP address < New XIP window address + New XIP size

Figure 157. XIP Address Split



15.4.4 Memory Mapped Mode

In memory-mapped mode, the external flash memory is seen as an internal memory but with some latency during access. Only read operations are allowed to the external flash memory in this mode. When configured in memory-mapped mode, the external QSPI device is seen as an internal memory.

It is forbidden to access the quad-QSPI flash bank area before having properly configured and enabled the QSPI peripheral. No more than 256 Mbytes can address even if the flash memory capacity is larger.

If access is made to an address outside of the range defined by QSPI_XIPSZR but still within the 256-Mbyte range, then a bus error is given. The effect of this error depends on the bus master that attempted the access:

- If it is the Cortex CPU, bus fault exception is generated when enabled (or an HardFault exception when bus fault is disabled).
- If it is a DMA, a DMA transfer error is generated, and the corresponding DMA channel is automatically disabled.

The QSPI_XIPSZR register determines the data size for reading external flash memory data. The data is read in 32-bit units, with the address increasing in 4-byte increments. The size is specified in bytes. The SIZE[27:0] bits in the QSPI_XIPSZR register must be set to 2^n - 1. For example, to read 64 bytes (2^6) from flash memory, the SIZE[27:0] bits in the QSPI_XIPSZR register must be set to 0x3F.

The QSPI_XIPWINAR register defines the address in internal memory where the external flash memory data will be stored. The storage address in internal memory must be set as a multiple of (SIZE[27:0] bits in the QSPI_XIPSZR register + 1). For example, if the SIZE[27:0] bits in the QSPI_XIPSZR register are set to 0x3F, the ADDR[27:0] bits in the QSPI_XIPWINAR register must be set to 0x0, 0x40, or 0x80.

The QSPI_XIPBAR register specifies the address for reading external flash memory data.

The QSPI_XIPACCBNDR register sets an access boundary within a 256 Mbytes range. If an address outside the defined boundary is accessed, a bus error will occur, even if the address falls within the 256 Mbytes range.

Table 136. External Bus Space in QSPI Memory Map

QSPI Memory Map	Address
External Bus Space	0x6000_0000 ~ 0x6FFF_FFFF

15.4.5 Setting XIP Mode

To start XIP mode in serial flash memory, perform the following register settings:

First, set the clock configuration of the QSPI.

- Sample point, minimum chip select, clock period and clock polarity, and transmit data on edge in QSP_CLKCFGR register.
- Clock polarity and transmit data on edge sets are important. Refer to Table 141.

Set the configuration of the XIP-dedicated registers. The registers that must be configured by default are listed below. Control should be applied based on the information regarding the performance enhance mode (XIP, execute in place) of the NOR flash in use.

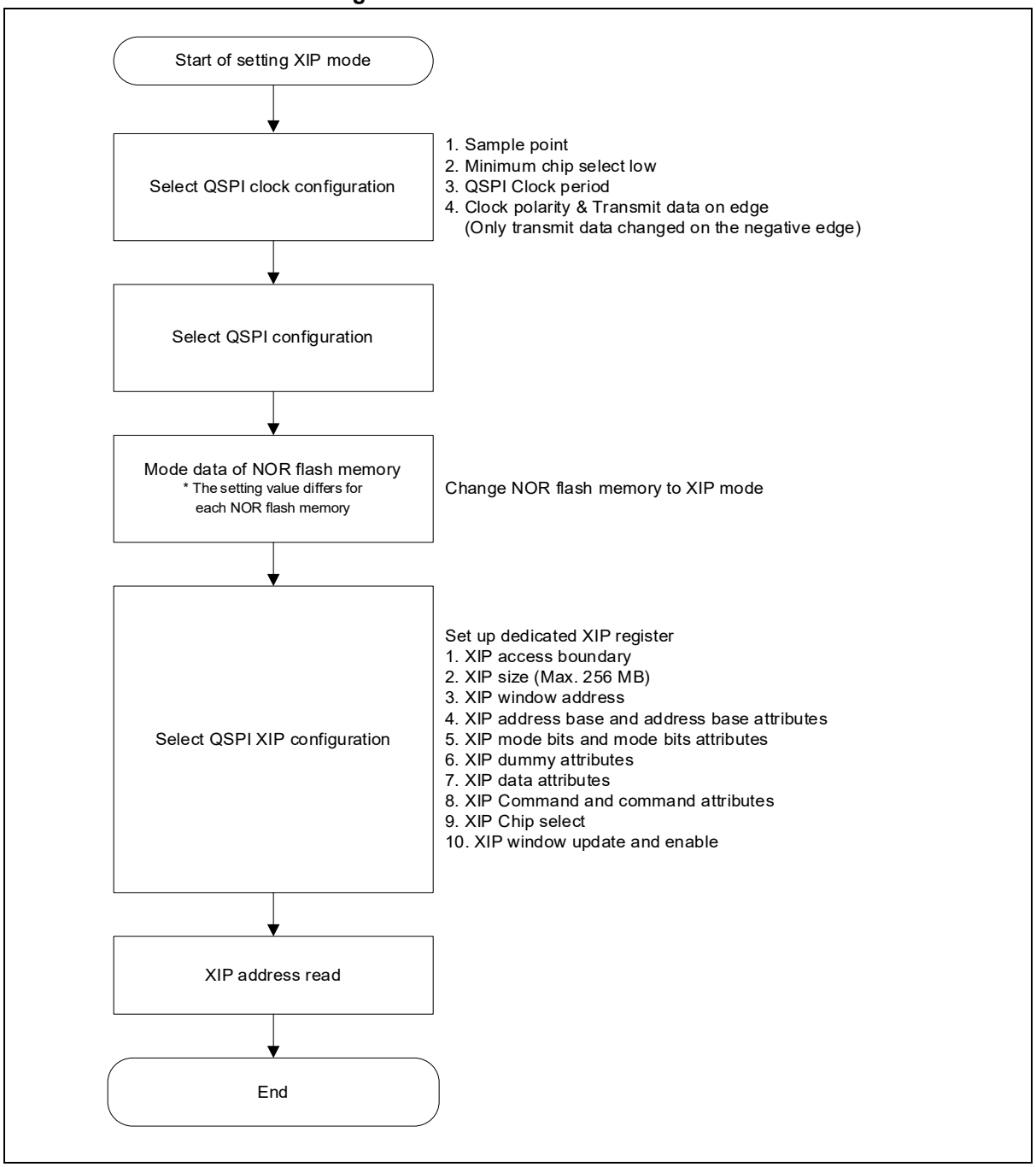
Table 137. Configuration of XIP Register

Configuration	XIP Register
Access boundary	QSPI_XIPACCBNDR
Size	QSPI_XIPSZR
Window address (default 0x6000_0000)	QSPI_XIPWINAR
Address base	QSPI_XIPBAR
Address attributes	QSPI_XIPAATR
Mode bits	QSPI_XIPMBR
Mode bits attributes	QSPI_XIPMBATR
Dummy attributes	QSPI_XIPDMYATR
Data attributes	QSPI_XIPDATR
Command	QSPI_XIPCMDR
Command attributes	QSPI_XIPCMDATR
Chip select	QSPI_XIPCSR
XIP Control	QSPI_XIPCR

When reading the XIP area, the data from the addressed region is fetched from external memory via QSPI.

The following figure shows an example of the XIP mode setting procedure.

Figure 158. Flowchart of XIP Mode



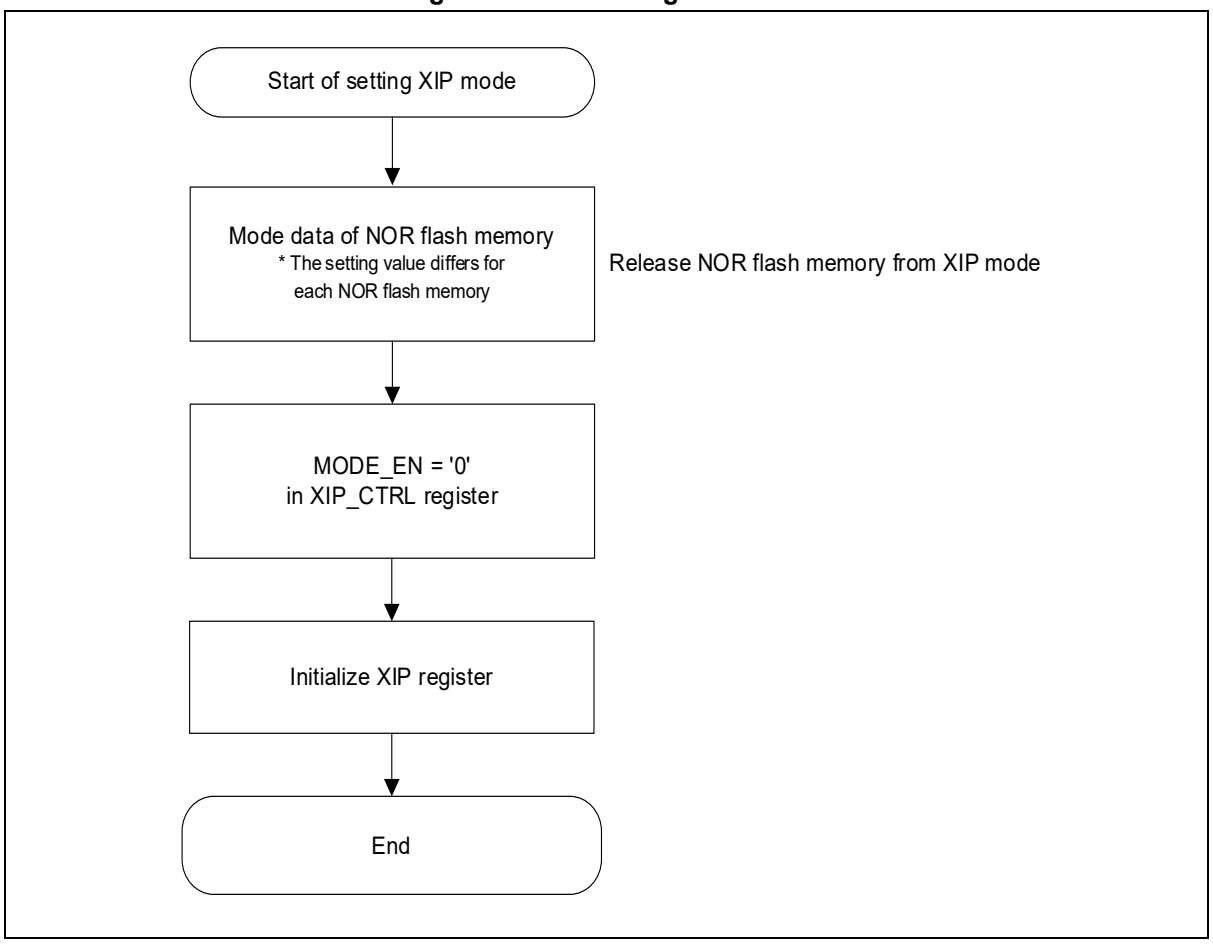
15.4.6 Releasing XIP mode

To release XIP mode in serial flash memory, perform the following register setting:

To disable the XIP mode of the NOR flash, control must be applied according to the specifications of the NOR flash being used. Then, setting the EN bit in the QSPI_XIPCR register to '0' will disable the XIP mode.

Figure 159 shows an example of the procedure for releasing XIP mode.

Figure 159. Releasing XIP Mode



15.5 QSPI Interrupts

During QSPI communications, various interrupts can be generated by different events. They can be checked by reading the QSPI_SR and QSPI_FSR register.

Table 138. Interrupt on QSPI

Interrupt Event	Event Flag	Enable Control Bit	Interrupt Clear Method
(status) Not Busy	BUSY	NBUSYIE	Read-Only
Write FIFO error flag	FWERR	FWERRIE	Write '1' in FWERR
(status) Write FIFO indicate	FWER	FWFILLIE	Read-Only
Read FIFO error flag	FRERR	FRERRIE	Write '1' in FRERR
(status) Read FIFO indicate	FRER	FRFILLIE	Read-Only
DMA successfully transfer flag	DIRQ	DIE	Write '1' in DIRQ
DMA error flag	DERR	DERRIE	Write '1' in DERR
(status) DMA idle	DIDLE	DIDLEIE	Read-Only
(status) FIFO full	FULL	-	Read-Only
(status) FIFO empty	EMPTY	-	Read-Only
(status) FIFO fill level	FILL	-	Read-Only

15.6 QSPI Registers

The base address and register map of the UART are described in the following tables:

Table 139. Base Address of CAN FD

Name	Base Address
QSPI	0x4002_0000

Table 140. QSPI Register Map

Name	Offset	Type	Description	Reset value	Reference
QSPI_GCR	0x0000	RW	QSPI Global Control Register	0x0000_0000	15.6.1
QSPI_CLKCFGR	0x0004	RW	QSPI Clock Configuration Register	0x0F00_070F	15.6.2
QSPI_MINCSHR	0x0008	RW	QSPI Min CS High Register	0x0000_0000	15.6.3
QSPI_FSR	0x0010	RW	QSPI FIFO Status Register	0x0001_0000	15.6.4
QSPI_SR	0x0014	RO	QSPI Status Register	0x0004_0001	15.6.5
QSPI_ICR	0x0018	RW	QSPI Interrupt Control Register	0x0000_0000	15.6.6
QSPI_CMDR	0x0020	RW	QSPI Command Register	0x0000_0000	15.6.7
QSPI_CMDATR	0x0024	RW	QSPI Command Attributes Register	0x0000_0107	15.6.8
QSPI_AR	0x0030	RW	QSPI Address Register	0x0000_0000	15.6.9
QSPI_AATR	0x0034	RW	QSPI Address Attributes Register	0x0006_0117	15.6.10
QSPI_CSR	0x0038	RW	QSPI Chip Select Register	0x0000_0001	15.6.11
QSPI_DMYATR	0x0040	RW	QSPI Dummy Attributes Register	0x0100_0000	15.6.12
QSPI_XIPHOTR	0x0044	RW	QSPI XIP Hold Off Time Register	0x0000_0000	15.6.13
QSPI_XIPHOSR	0x0048	RW	QSPI XIP Hold Off Select Register	0x0000_0000	15.6.14
QSPI_XIPHOCR	0x004C	RW	QSPI XIP Hold Off Control Register	0x0000_0000	15.6.15
QSPI_DSZR	0x0050	RW	QSPI Data Size Register	0x0000_0000	15.6.16
QSPI_DATR	0x0054	RW	QSPI Data Attributes Register	0x0000_0000	15.6.17
QSPI_TDR	0x0058	WO	QSPI Transmit Data Register	0x0000_0000	15.6.18
QSPI_RDR	0x005C	RO	QSPI Receive Data Register	0x0000_0000	15.6.19
QSPI_MBR	0x0068	RW	QSPI Mode Bits Register	0x0000_0000	15.6.20
QSPI_MBATR	0x006C	RW	QSPI Mode Bits Attributes Register	0x0001_0000	15.6.21

Table 140. QSPI Register Map (continued)

Name	Offset	Type	Description	Reset value	Reference
QSPI_XIPCR	0x0080	RW	QSPI XIP Control Register	0x0000_0001	15.6.22
QSPI_XIPACCBNDR	0x0084	RW	QSPI XIP Access Boundary Register	0x0000_001F	15.6.23
QSPI_XIPSZR	0x0088	RW	QSPI XIP Size Register	0x001F_FFFF	15.6.24
QSPI_XIPWINAR	0x008C	RW	QSPI XIP Window Address Register	0x0000_0000	15.6.25
QSPI_XIPCMDR	0x0090	RW	QSPI XIP Command Register	0x0000_0B00	15.6.26
QSPI_XIPCMDATR	0x0094	RW	QSPI XIP Command Attributes Register	0x0000_0107	15.6.27
QSPI_XIPBAR	0x00A0	RW	QSPI XIP Base Address Register	0x0000_0000	15.6.28
QSPI_XIPAATR	0x00A4	RW	QSPI XIP Address Attributes Register	0x000C_0100	15.6.29
QSPI_XIPCSR	0x00A8	RW	QSPI XIP Chip Select Register	0x0000_0001	15.6.30
QSPI_XIPDMYATR	0x00B0	RW	QSPI XIP Dummy Attributes Register	0x0100_0007	15.6.31
QSPI_XIPDATR	0x00C0	RW	QSPI XIP Data Attributes Register	0x0000_0001	15.6.32
QSPI_XIPMBR	0x00D0	RW	QSPI XIP Mode Bits Register	0x0000_0000	15.6.33
QSPI_XIPMBATR	0x00D4	RW	QSPI XIP Mode Bits Attributes Register	0x0001_0000	15.6.34
QSPI_DMASTAR	0x00E0	RW	QSPI DMA Start Address Register	0x0000_0000	15.6.35
QSPI_DMAFBLSR	0x00E4	RW	QSPI DMA Master Fixed Burst Length Select Register	0x0000_0F00	15.6.36
QSPI_DMASR	0x00E8	RW	QSPI DMA Status Register	0x0000_0001	15.6.37
QSPI_DMABDCNTR	0x00EC	RW	QSPI DMA Buffer Descriptor Count Register	0x0000_0000	15.6.38
QSPI_WPCR	0x00F8	RW	QSPI Write Protect Control Register	0x0000_0001	15.6.39
QSPI_HTR	0x00FC	RW	QSPI Hold Time Register	0x0000_0007	15.6.40

15.6.1 QSPI_GCR: QSPI Global Control Register

The QSPI_GCR register is a 32-bit read/write accessible register. It sets the operating QSPI DMA modes and software reset.

QSPI_GCR=0x4002_0000

31 30 29 28 27 26 25 24	23 22 21 20 19 18 17 16	15 14 13 12 11 10 9 8	7 6 5 4 3 2 1 0
Reserved	DMAEN	Reserved	SWRST
-	0	-	0
-	RW	-	RW

16	DMAEN	To complete next command using DMA, set to ‘1’.	
		0	Disables DMA
		1	Enables DMA
0	SWRST	When set to ‘1’, holds most of the internal logic in reset. Registers keep their old values.	
		0	No effect
		1	Software reset enable

15.6.2 QSPI_CLKCFGR: QSPI Clock Configuration Register

The QSPI_CLKCFGR register is a 32-bit read/write accessible register. It sets the sample point, chip select low duration, clock period in QSPI clock configuration.

QSPI_CLKCFGR=0x4002_0004

31 30 29 28 27 26 25 24	23 22 21 20 19 18	17	16	15 14 13 12 11 10 9 8	7 6 5 4 3 2 1 0
SP[7:0]	Reserved	CPOL	TRANS	MINCSL[7:0]	PRD[7:0]
0x0F	-	0	0	0x07	0x0F
RW	-	RW	RW	RW	RW

31 24	SP[7:0]	Sets the sampling point of the input. The sampling point is given as a number of clock cycles, less one, since the transmitting edge. Refer to section 15.3.3.2.
17	CPOL	Selects the clock polarity 0 QSPI clock output starts low and ends low 1 QSPI clock output starts high and ends high
16	TRANS	Select the clock edge on which the output data is changed 0 Change output on the negative edge 1 Reserved
15 8	MINCSL[7:0]	Time between CS going active and first clock edge, and time from last clock edge to CS going inactive. The time is given in a number of input clock cycles as follows: Min, CS Low + 1 Refer to section 15.3.3.1.
7 0	PRD[7:0]	The period of the QSPI output clock. The internal QSPI clock is divided by the following value to provide the QSPI output clock: (Clock period + 1) Note that the PHY used may have some restriction on the clock divider value. Only support even clock division factors. Refer to Figure 147.

Table 141. Mode Based on TRNAS and CPOL

TRANS[16]	CPOL[17]	Description
0	0	Support (Mode 0)
0	1	Support (Mode 3)

NOTE:

1. Data change is only negative(falling) edge and data sampling is only positive(rising) edge.

15.6.3 QSPI_MINCSHR: QSPI Minimum Chip Select High Register

The QSPI_MINCSHR register is a 32-bit read/write accessible register that determines the minimum high period of the QSPI chip select.

QSPI_MINCSHR=0x4002_0008

31 30 29 28 27 26 25 24	23 22 21 20 19 18 17 16	15 14 13 12 11 10 9 8	7 6 5 4 3 2 1 0
Reserved			MINCSH[7:0]
-			0x00
-			WO

7 0	MINCSH[7:0]	Whenever the QSPI memory controller is required to drive the CS line inactive, the QSPI controller will wait for the following number of internal QSPI clock cycles before driving it active again: MINCSH + 1

15.6.4 QSPI_FSR: QSPI FIFO Status Register

The QSPI_FSR register is a 32-bit read accessible register. It displays the FIFO status of the QSPI interface.

QSPI_FSR=0x4002_0010

31 30 29 28 27 26 25 24	23 22 21 20 19 18 17 16			15 14 13 12 11 10 9 8	7 6 5 4 3 2 1 0
Reserved		FULL	EMPTY	FILL[15:0]	
-		0	1	0x0000	
-		RO	RO	RO	

17	FULL	The QSPI FIFO full status bit	
		0	FIFO is not full
		1	FIFO is full
16	EMPTY	The QSPI FIFO empty status bit	
		0	FIFO is not empty
		1	FIFO is empty
15 0	FILL[15:0]	Current FIFO fill level. The Read and Write FIFOs each have a depth of 32. Each depth unit consists of 1 word (4 bytes), allowing a total of 32 words (128 bytes) per FIFO. The FIFO is divided into 8 levels, with each level consisting of 4 words (16 bytes).	

15.6.5 QSPI_SR: QSPI Status Register

The QSPI_SR register is a 32-bit read/write accessible register. It displays the status of the QSPI interface.

QSPI_SR=0x4002_0014

31 30 29 28 27 26 25 24	23 22 21	20	19	18	17	16	15 14 13 12 11 10 9 8	7 6 5 4 3 2 1	0
Reserved		FWERR	FRERR	FWER	FRER	BUSY	Reserved		RDY
-		0	0	1	0	0	-		1
-		RWC1	RWC1	RO	RO	RO	-		RO

20	FWERR	Erroneous (unexpected) FIFO write detected flag. Unexpected write events include: Writing the FIFO when a read operation has been requested. Writing the FIFO while full.
	0	Not flagged.
	1	Flagged (Writing a '1' to the bit clears the flag)
19	FRERR	Erroneous (unexpected) FIFO read detected. Unexpected read events include: Reading the FIFO when a write operation has been requested Reading the FIFO while empty.
	0	Not flagged.
	1	Flagged (Writing a '1' to the bit clears the flag)
18	FWER	Write FIFO indicates status bit. 0b1 indicates that Write FIFO fill level <= FIFO boundary. The FIFO boundary is defined using FBND[2:0] in the QSPI_ICR register.
	0	Not indicating status
	1	Indicating status
17	FRER	Read FIFO indicate status bit. 0b1 indicates that Read FIFO fill level > FIFO boundary. The FIFO boundary is defined using FBND[2:0] in the QSPI_ICR register.
	0	Not indicating status
	1	Indicating status
16	BUSY	QSPI busy status bit. '1' indicates QSPI engine is busy performing QSPI memory access. While this is set to '1' the configuration may not be changed nor new commands issued.
	0	Not Busy
	1	Busy
0	RDY	QSPI ready status bit. When the SCU_PCER3 register is enabled, it is always read as '1'.
	0	Not ready (If QSPI bit in SCU_PCER3 register is disable)
	1	Ready

15.6.6 QSPI_ICR: QSPI Interrupt Control Register

The QSPI_ICR register is a 32-bit read/write accessible register. Determines whether to enable the QSPI interrupts.

QSPI_ICR=0x4001_0018

31 30 29 28 27 26 25 24	23 22 21 20 19	18	17	16	15 14 13 12 11	10 9 8	7 6 5	4	3	2	1	0
Reserved		DERRIE	DIE	DIDLEIE	Reserved	FBND[2:0]	Reserved	FWERRIE	FRERRIE	FWFILLIE	FRFILLIE	NBUSYIE
-		0	0	0	-	000	-	0	0	0	0	0
-		RW	RW	RW	-	RW	-	RW	RW	RW	RW	RW

18	DERRIE	DMA error interrupt bit. Interrupt active when this bit is '1' and DERR bit is '1' in the QSPI_DAMSR register.
	0	Disables DMA error Interrupt.
	1	Enables DMA error Interrupt.
17	DIE	DMA interrupt bit. Interrupt active when this bit is '1' and DIRQ bit is '1' in the QSPI_DAMSR register.
	0	Disables DMA Interrupt.
	1	Enables DMA Interrupt.
16	DIDLEIE	DMA idle interrupt bit. Interrupt active when this bit is '1' and DIDLE bit is '1' in the QSPI_DAMSR register.
	0	Disables DMA idle Interrupt.
	1	Enables DMA idle Interrupt.
10 8	FBND[2:0]	FIFO boundary FIFO fill limits setting in 1/8 of FIFO depth steps. The Read and Write FIFOs each have a depth of 32. Each depth unit consists of 1 word (4 bytes), allowing a total of 32 words (128 bytes) per FIFO. The FIFO is divided into 8 levels, with each level consisting of 4 words (16 bytes). Activates Read FIFO Status when FIFO is at least n/8 full during or after read command. Activates Write FIFO Status when FIFO is less than n/8 full during write command.
	0	Read FIFO is detected when FIFO is at least 1/8 full. Write FIFO is detected when FIFO is less than 1/8 full.
	1	Read FIFO is detected when FIFO is at least 2/8 full. Write FIFO is detected when FIFO is less than 2/8 full.
	2	Read FIFO is detected when FIFO is at least 3/8 full. Write FIFO is detected when FIFO is less than 3/8 full.
	3	Read FIFO is detected when FIFO is at least 4/8 full. Write FIFO is detected when FIFO is less than 4/8 full.
	4	Read FIFO is detected when FIFO is at least 5/8 full. Write FIFO is detected when FIFO is less than 5/8 full.
	5	Read FIFO is detected when FIFO is at least 6/8 full. Write FIFO is detected when FIFO is less than 6/8 full.
	6	Read FIFO is detected when FIFO is at least 7/8 full. Write FIFO is detected when FIFO is less than 7/8 full.
	7	Read FIFO is detected when FIFO is at least 8/8 full. Write FIFO is detected when FIFO is less than 8/8 full.

4	FWERRIE	Write FIFO error interrupt bit. Interrupt active when this bit is '1' and FWERR bit is '1' in the QSPI_SR register.
		0 Disables write FIFO error Interrupt.
		1 Enables write FIFO error Interrupt.
3	FRERRIE	Read FIFO error interrupt bit. Interrupt active when this bit is '1' and FRERR bit is '1' in the QSPI_SR register.
		0 Disables read FIFO error Interrupt.
		1 Enables read FIFO error Interrupt.
2	FWFILLIE	Write FIFO fill interrupt bit. Enable interrupt when the write FIFO fill level is less-than or equal to the level set by FBND[2:0] bits in this register. (FWER bit in QSPI_SR register). In XIP mode the write FIFO fill level interrupt is unconditionally disabled, irrespective of this bit. This bit should be '0' during DMA, read or commands without data.
		0 Disables write FIFO fill Interrupt.
		1 Enables write FIFO fill Interrupt.
1	FRFILLIE	Read FIFO fill interrupt bit. Enable interrupt when the read FIFO fill level is more than the level set by FBND[2:0] bits in this register (FRER bit in QSPI_SR register). In XIP mode the read FIFO fill level interrupt is unconditionally disabled, irrespective of this bit. Should be '0' during DMA, write or commands without data.
		0 Disables read FIFO fill Interrupt.
		1 Enables read FIFO fill Interrupt.
0	NBUSYIE	Not busy interrupt bit. If this bit is set to '1', the QSPI memory controller will interrupt if it is not busy. (i.e., when BUSY bit in the QSPI_SR register is set to '0').
		0 Disables not busy interrupt.
		1 Enables not busy interrupt.

15.6.7 QSPI_CMDR: QSPI Command Register

The QSPI_CMDR register is a 32-bit read/write accessible register. Writing to this register starts executing the written command on the QSPI bus. Note that all other command parameters should be configured before writing to this register. This is necessary even if the command is skipped in such cases the written value is not used.

QSPI_CMDR=0x4002_0020

31 30 29 28 27 26 25 24	23 22 21 20 19 18 17 16	15 14 13 12 11 10 9 8	7 6 5 4 3 2 1 0
Reserved		CMD[15:0]	
-		0x0000	
-		RW	

15 0	CMD[15:0]	For commands less than 16 bits wide, align command's most significant bit to bit [15].

NOTE:

1. The Most Significant Bit (MSB) is output. For example, to use the command value 0xF5, 0xF500 must be written to CMD[15:0].

15.6.8 QSPI_CMDATR: QSPI Command Attributes Register

The QSPI_CMDATR register is a 32-bit read/write accessible register. This register configures how the command portion of the QSPI transaction shall be output on the QSPI bus.

QSPI_CMDATR=0x4002_0024

31 30 29 28 27 26 25 24	23 22 21 20 19 18 17 16	15 14 13 12 11 10 9 8	7 6 5 4 3 2 1 0
Reserved	SKIP	Reserved DDR MODE[3:0]	Reserved DUR[3:0]
-	0	- 0 0x1	- 0x7
-	RW	- RW RW	- RW

16	SKIP	This bit selects whether the command phase is to be skipped or not on the next QSPI transaction
	0	Command phase is present
	1	Command phase is skipped
12	DDR	This bit selects whether the command phase is executed in DDR mode (using both edges of the output clock)
	0	Normal (STR) output mode is used
	1	DDR (DTR) output mode is used
11 8	MODE[3:0]	This field selects the number of lines to use for the command output
	001	Single (x1, 1-bits)
	010	Dual (x2, 2-bits)
	100	Quad (x4, 4-bits)
	Others	Reserved
3 0	DUR[3:0]	This field must reflect the length of the command phase of the QSPI transaction in QSPI clock cycles minus 1. A few examples: - 8-bit command in STR x1 mode: '7' - 8-bit command in DDR x1 mode: '3'

NOTE:

1. The bit[13] and bit[15] values in QSPI_CMDATR register must be '0'.

15.6.9 QSPI_AR: QSPI Address Register

The QSPI_AR register is a 32-bit read/write accessible register. This register is the address phase of the QSPI transaction.

QSPI_AR=0x4002_0030

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
ADDR[31:0]																															
0x00000000																															
RW																															

31 0	ADDR[31:0]	This field holds the value of the address output on the address phase of the QSPI transaction. For addresses less than 32 bits wide, align the most significant bit of the address to bit[31].

15.6.10 QSPI_AATR: QSPI Address Attributes Register

The QSPI_AATR register is a 32-bit read/write accessible register. This register configures how the address portion of the QSPI transaction shall be output on the QSPI bus.

QSPI_AATR=0x4002_0034

31 30 29 28 27 26 25 24	23 22 21 20 19 18 17 16	15 14 13 12 11 10 9 8	7 6 5 4 3 2 1 0
Reserved	MSK[1:0] SKIP	Reserved DDR MODE[3:0]	Reserved DUR[4:0]
-	11 0	- 0 0x1	- 10111
-	RW RW	- RW RW	- RW

18 17	MSK[1:0]	Must be '3'.
16	SKIP	This bit selects whether the address phase is to be skipped or not on the next QSPI transaction
	0	Address phase is present
	1	Address phase is skipped
12	DDR	This bit selects whether the command phase is executed in DDR mode (using both edges of the output clock)
	0	Normal (STR) output mode is used
	1	DDR (DTR) output mode is used
11 8	MODE[3:0]	This field selects the number of lines to use for the command output
	001	Single (x1, 1-bits)
	010	Dual (x2, 2-bits)
	100	Quad (x4, 4-bits)
	Others	Reserved
3 0	DUR[4:0]	This field must reflect the length of the command phase of the QSPI transaction in QSPI clock cycles minus 1. A few examples: - 24-bit address in STR x4 mode: 5 - 24-bit address in STR x1 mode: 23 (Default) - 24-bit address in DDR x1 mode: 11

NOTE:

1. The bit[13] value in QSPI_AATR register must be '0'.

15.6.11 QSPI_CSR: QSPI Chip Select Register

The QSPI_CSR register is a 32-bit read/write accessible register.

QSPI_CSR=0x4002_0038

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

0	CS	Chip Select line. For bit set in this register, the corresponding chip-select line shall be active for the next command.
	0	Disable the QSPI chip select line.
	1	Enable the QSPI chip select line.

15.6.12 QSPI_DMYATR: QSPI Dummy Attributes Register

The QSPI_DMYATR register is a 32-bit read/write accessible register. This register configures how the dummy portion of the QSPI transaction shall be output on the QSPI bus.

QSPI_DMYATR=0x4002_0040

31 30 29 28 27 26 25 24	23 22 21 20 19 18 17 16	15 14 13 12 11 10 9 8	7 6 5 4 3 2 1 0		
Reserved	NO_WRLAT	Reserved	SKIP	Reserved	DUR[4:0]
-	1	-	0	-	00000
-	RW	-	RW	-	RW

24	NO_WRLAT	Must be '1'.
8	SKIP	This bit selects whether the dummy phase is to be skipped or not on the next QSPI transaction
	0	Dummy phase is present
	1	Dummy phase is skipped
4 0	DUR[4:0]	This field reflects the length of the dummy phase of the QSPI transaction in QSPI clock cycles minus 1.

NOTE:

1. The bit[25], bit[20:16] values in QSPI_DMYATR register must be '0'.

15.6.13 QSPI_XIPHOTR: QSPI XIP Hold Off Time Register

The QSPI_XIPHOTR register is a 32-bit read/write accessible register.

QSPI_XIPHOTR=0x4002_0044

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
TIME[31:0]																															
0x00000000																															
RW																															

31 0	TIME[31:0]	This register gives the time during which XIP accesses are denied, when the command is executed with WERS bit in the QSPI_CMDATR register is '1'. The time is given in a number of bus clock cycles.

15.6.14 QSPI_XIPHOSR: QSPI XIP Hold Off Select Register

The QSPI_XIPHOSR register is a 32-bit read/write accessible register.

QSPI_XIPHOSR=0x4002_0048

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

0	EN	This register gives the XIP windows for which the XIP hold off should be applied	
		0	Disable the hold off time.
		1	Enable the hold off time.

15.6.15 QSPI_XIPHOCR: QPSI XIP Hold Off Control Register

The QSPI_XIPHOCR register is a 32-bit read/write accessible register.

QSPI_XIPHOCR=0x4002_004C

31 30 29 28 27 26 25 24	23 22 21 20 19 18 17 16	15 14 13 12 11 10 9 8	7 6 5 4 3 2 1 0
Reserved			EN
-			0x0
			RW

0	EN	Hold off in XIP enable bit. This register is two separate registers: Reading this register returns read-only data about the hold off statuses: a bit being set to '1' means that the given XIP window has a hold off in progress. Writing this register sets a write-only register which each set bit cancels the hold off state for the given XIP window. 0 Disable the hold off in XIP. 1 Enable the hold off in XIP.

15.6.16 QSPI_DSZR: QSPI Data Size Register

The QSPI_DSZR register is a 32-bit read/write accessible register.

QSPI_DSZR=0x4002_0050

31 30 29 28 27 26 25 24	23 22 21 20 19 18 17 16	15 14 13 12 11 10 9 8	7 6 5 4 3 2 1 0
SIZE[31:0]			
0x0000000			
RW			

31 0	SIZE[31:0]	Number of data bits to transmit or receive, minus 1. The number of bits provided here must be evenly divisible by the number of bits handled per clock cycles. For example, for normal (STR) x4 mode (4-bits/cycle) this register may be set to 4, 8, 12, etc. bits, or for DDR (DTR) x4 mode (8-bits/clock edge) this register may be set to 16, 32, 24, etc. bits.

15.6.17 QSPI_DATR: QSPI Data Attributes Register

The QSPI_DATR register is a 32-bit read/write accessible register. This register configures how the data portion of the QSPI transaction shall be output on the QSPI bus.

QSPI_DATR =0x4002_0054

31 30 29 28 27 26 25 24	23 22 21 20 19 18 17	16	15 14 13 12 11 10 9	8	7	6 5	4	3 2 1 0
Reserved		DIR	Reserved	SKIP	FLIP	Reserved	DDR	MODE[3:0]
-		0	-	0	0	-	0	0x0
-		RW	-	RW	RW	-	RW	RW

16	DIR	Data direction select	
		0	Data is output (QSPI memory is written)
		1	Data in input (QSPI memory is read)
8	SKIP	This bit selects whether the data phase is to be skipped or not on the next QSPI transaction	
		0	Data phase is present
		1	Data phase is skipped
7	FLIP	Byte order select	
		0	Return data using the same order as the memory chip provided the data in
		1	Flip the order of 4-bits in each 8-bit bytes This bit only has effect in DDR x4 mode and is ignored in all other cases.
4	DDR	This bit selects whether the data phase is executed in DDR mode (using both edges of the clock)	
		0	Normal (STR) mode is used
		1	DDR (DTR) mode is used
3 0	MODE[3:0]	This field selects the number of lines to use for the data output	
		001	Single (x1, 1-bits)
		010	Dual (x2, 2-bits)
		100	Quad (x4, 4-bits)
		Others	Reserved

NOTE:

1. The bit[6], bit[5] values in QSPI_DATR register must be '0'.

15.6.18 QSPI_TDR: QSPI Transmit Data Register

The QSPI_TDR register is a 32-bit read/write accessible register.

QSPI_TDR =0x4002_0058

31 30 29 28 27 26 25 24	23 22 21 20 19 18 17 16	15 14 13 12 11 10 9 8	7 6 5 4 3 2 1 0
TDR[31:0]			
0x00000000			
WO			

31 0	TDR[31:0]	Window into the QSPI custom command data FIFO. Writing this register pushes one 32-bit word into the QSPI custom command data FIFO. This data is used in the data phase of custom transactions. See section 15.3.5. For data widths less than 32 bits or the last word of a transaction not a multiple of 32-bits, the most significant data bit must be aligned to bit[31]. Reads return 0 and don't pop the FIFO.

15.6.19 QSPI_RDR: QSPI Receive Data Register

The QSPI_RDR register is a 32-bit read/write accessible register.

QSPI_RDR =0x4002_005C

31 30 29 28 27 26 25 24	23 22 21 20 19 18 17 16	15 14 13 12 11 10 9 8	7 6 5 4 3 2 1 0
RDR[31:0]			
0x00000000			
RO			

31 0	RDR[31:0]	Window into the QSPI custom command data FIFO. Reading this register pops one 32-bit word from the QSPI custom command data FIFO. Read data from the data phase of custom transactions can be read from this register. The received data is such that the least-significant bit of the data is aligned to bit[0] after reception.

15.6.20 QSPI_MBR: QSPI Mode Bits Register

The QSPI_MBR register is a 32-bit read/write accessible register. This register is the mode bits phase of the QSPI transaction.

QSPI_MBR =0x4002_0068

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
MODE[31:0]																															
0x00000000																															
RW																															

31 0	MODE[31:0]	QSPI Mode bits to use for the mode phase of the QSPI transaction. For data widths less than 32 bits, the most significant data bit must be aligned to bit[31].

15.6.21 QSPI_MBATR: QSPI Mode Bits Attributes Register

The QSPI_MBATR register is a 32-bit read/write accessible register. This register configures how the mode bits portion of the QSPI transaction shall be output on the QSPI bus.

QSPI_MBATR =0x4002_006C

31 30 29 28 27 26 25 24	23 22 21 20 19 18 17	16	15 14 13	12	11 10 9 8	7 6 5	4 3 2 1 0
Reserved		SKIP	Reserved	DDR	MODE[3:0]	Reserved	DUR[4:0]
-		1	-	0	0000	-	00000
-		RW	-	RW	RW	-	RW

16	SKIP	This bit selects whether the mode phase is to be skipped or not on the next QSPI transaction	
		0	Mode phase is present
		1	Mode phase is skipped
12	DDR	This bit selects whether the mode phase is executed in DDR mode (using both edges of the clock)	
		0	Normal (STR) mode is used
		1	DDR (DTR) mode is used
11 8	MODE[3:0]	This field selects the number of lines to use for the mode output	
		001	Single (x1, 1-bits)
		010	Dual (x2, 2-bits)
		100	Quad (x4, 4-bits)
		Others	Reserved
4 0	DUR[4:0]	This field must reflect the length of the mode phase of the QSPI transaction in QSPI clock cycles minus 1. 8-bit mode in STR x4 mode: 1 8-bit mode in STR x1 mode: 7 8-bit mode in DDR x1 mode: 3 8-bit mode in DDR x4 mode: 0	

NOTE:

1. The bit[13] values in QSPI_MBATR register must be '0'.

15.6.22 QSPI_XIPCR: QSPI XIP Control Register

The QSPI_XIPCR register is a 32-bit read/write accessible register.

QSPI_XIPCR =0x4002_0080

31 30 29 28 27 26 25 24	23 22 21 20 19 18 17 16	15 14 13 12 11 10	9	8	7 6 5 4 3 2	1	0
Reserved			UPDATE	SUSRES	Reserved	BUSY	EN
-			0	0	-	0	1
-			RW	RW	-	RO	RW

9	UPDATE	If set to '1', the attributes specified in the New *XIP** registers are applied.
	0	Disable (No update)
	1	Enable (Update the *XIP** register)
8	SUSRES	Set to '1' to enable suspend/resume support. Setting this bit is only applied when UPDATE in QSPI_XIPCR register is also set to '1'.
	0	Disable the suspend and resume function
	1	Enable the suspend and resume function
1	BUSY	XIP operation state
	0	Idle (XIP transfer is not active)
	1	Busy (XIP transfer in progress)
0	EN	Enable the XIP window. Any XIP window may be disabled at any time. If an XIP access is in progress when the XIP window is disabled it will be finished cleanly.
	0	Disable the XIP window.
	1	Enable the XIP window.

15.6.23 QSPI_XIPACCBNDR: QSPI XIP Access Boundary Register

The QSPI_XIPACCBNDR register is a 32-bit read/write accessible register.

QSPI_XIPACCBNDR =0x4002_0084

31 30 29 28 27 26 25 24	23 22 21 20 19 18 17 16	15 14 13 12 11 10 9 8	7 6 5 4 3 2 1 0
Reserved			BND [4:0]
-			0x1F
-			RW

4 0	BND [4:0]	Access boundary for XIP accesses. For any burst that would pass this boundary, the controller will end the transaction and start a new one. boundary is on. For example to set a boundary on 512 bytes, this register should be set to 9. Some chips have a restriction that reads and writes can't cross a certain boundary. For these chips, the XIP access boundary register must be set to the value in its datasheet. Another use for this setting is to limit the CS low time which, again, some chips have. If your chip doesn't have such restrictions, this register is to be set to max value.

15.6.24 QSPI_XIPSZR: QSPI XIP Size Register

The QSPI_XIPSZR register is a 32-bit read/write accessible register.

QSPI_XIPSZR=0x4002_0088

31 30 29 28 27 26 25 24	23 22 21 20 19 18 17 16	15 14 13 12 11 10 9 8	7 6 5 4 3 2 1 0
Reserved	SIZE[27:0]		
-	0x001FFFFF		
-	RW		

28 0	SIZE [27:0]	This register gives the size of this XIP window. The size must a power of two. If the size is 2^N, the size is given by setting the low N bits to 1 in this register and the rest to '0'.

NOTE:

1. If SIZE[27:0] set to 0x3F, XIP size is 64 bytes.

15.6.25 QSPI_XIPWINAR: QSPI XIP Window Address Register

The QSPI_XIPWINAR register is a 32-bit read/write accessible register that allows setting the start address of the XIP memory region.

QSPI_XIPWINAR=0x4002_008C

31 30 29 28 27 26 25 24	23 22 21 20 19 18 17 16	15 14 13 12 11 10 9 8	7 6 5 4 3 2 1 0
Reserved	ADDR[27:0]		
-	0x0000000		
-	RW		

31 0	ADDR[27:0]	If ADDR[27:0] set to '0', this register is hard wired to '0x6000_0000'. For all other windows this is the window address as given on the AHB interface.

NOTE:

1. This register specifies the address offset when storing read data in XIP memory. Remember that it must be set as a multiple of the configured value of SIZE[27:0] bits in the XIPSZR register plus 1. (For example, if SIZE[27:0] bits in the QSPI_XIPSZR register is set to 0x7F, set ADDR[27:0] bits in the QSPI_XIPWINAR register to 0x0 or a multiple of 0x80, such as 0x80 or 0x100.)

15.6.26 QSPI_XIPCMDR: QSPI XIP Command Register

The QSPI_XIPCMDR register is a 32-bit read/write accessible register. Writing to this register starts executing the written command on the QSPI bus. Note that all other command parameters should be configured before writing to this register. This is necessary even if the command is skipped in such cases the written value is not used.

QSPI_XIPCMDR=0x4002_0090

31 30 29 28 27 26 25 24	23 22 21 20 19 18 17 16	15 14 13 12 11 10 9 8	7 6 5 4 3 2 1 0
Reserved		RCMD[15:0]	
-		0x0B00	
-		RW	

15 0	RCMD[15:0]	QSPI command to use for XIP reads. For commands less than 16 bits wide, align command's most significant bit to bit[15].

15.6.27 QSPI_XIPCMDATR: QSPI XIP Command Attributes Register

The QSPI_XIPCMDATR register is a 32-bit read/write accessible register. This register configures how the command portion of the QSPI transaction in XIP mode, shall be output on the QSPI bus.

QSPI_XIPCMDATR=0x4002_0094

31 30 29 28 27 26 25 24	23 22 21 20 19 18 17	16	15 14 13	12	11 10 9 8	7 6 5 4	3 2 1 0
Reserved		SKIP	Reserved	DDR	MODE[3:0]	Reserved	DUR[3:0]
-		0	-	0	0x1	-	0x7
-		RW	-	RW	RW	-	RW

16	SKIP	This bit selects whether the command phase is to be skipped or not for XIP-based QSPI transactions	
		0	Command phase is present
		1	Command phase is skipped
12	DDR	This bit selects whether the command phase is executed in DDR mode (using both edges of the output clock)	
		0	Normal (STR) output mode is used
		1	DDR (DTR) output mode is used
11 8	MODE	This field selects the number of lines to use for XIP command output	
		001	Single (x1, 1-bits)
		010	Dual (x2, 2-bits)
		100	Quad (x4, 4-bits)
		Others	Reserved
3 0	DUR	This field must reflect the length of the command phase of the XIP read in QSPI clock cycles minus 1. 8-bit mode in STR x8 mode: 0 8-bit mode in STR x1 mode: 7 8-bit mode in DDR x1 mode: 3 16-bit mode in DDR x8 mode: 0	

NOTE:

1. The bit[15:13] values in QSPI_XIPCMDATR register must be '0'.

15.6.28 QSPI_XIPBAR: QSPI XIP Base Address Register

The QSPI_XIPBAR register is a 32-bit read/write accessible register that allows the configuration of the starting address of the XIP slave interface.

QSPI_XIPBAR=0x4002_00A0

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
ADDR[31:0]																															
0x00000000																															
RW																															

31 0	ADDR[31:0]	This register selects the memory address to which the controller maps address on the XIP slave interface. This base address must be aligned to the value given SIZE[27:0] bits in the QSPI_XIPSZR register.

15.6.29 QSPI_XIPAATR: QSPI XIP Address Attributes Register

The QSPI_XIPAATR register is a 32-bit read/write accessible register. This register configures how the address portion of the QSPI transaction in XIP mode, shall be output on the QSPI bus.

QSPI_XIPAATR=0x4002_00A4

31 30 29 28 27 26 25 24	23 22 21 20 19 18 17 16	15 14 13 12 11 10 9 8	7 6 5 4 3 2 1 0
Reserved	MSK[1:0] WIDTH[1:0]	Reserved DDR MODE[3:0]	Reserved
-	11 00	- 0 0x1	-
-	RW RW	- RW RW	-

19 18	MSK[1:0]	Must be '3'.	
17 16	WIDTH[1:0]	Select address length used for XIP reads	
		00	Use 24-bit address field
		01	Use 32-bit address field
		10	Use 16-bit address field
		11	Use 8-bit address field
12	DDR	Die select, include die mode in DDR mode after update.	
11 8	MODE[4:0]	This field selects the number of lines to use for XIP command output	
		001	Single (x1, 1-bits)
		010	Dual (x2, 2-bits)
		100	Quad (x4, 4-bits)
		Others	Reserved

NOTE:

1. The bit[13] value in QSPI_XIPAATR register must be '0'.

15.6.30 QSPI_XIPCSR: QSPI XIP Chip Select Register

The QSPI_XIPCSR register is a 32-bit read/write accessible register.

QSPI_XIPCSR=0x4002_00A8

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

0	CS	The chip select line to use for XIP mode reads. Setting multiple bits to 1 in this register is guaranteed to cause bus conflicts and may damage the external memory chips.
		0 Disable the XIP chip select line.
		1 Enable the XIP chip select line.

15.6.31 QSPI_XIPDMYATR: QSPI XIP Dummy Attributes Register

The QSPI_XIPDMYATR register is a 32-bit read/write accessible register. This register configures how the dummy portion of the QSPI transaction in XIP mode, shall be output on the QSPI bus.

QSPI_XIPDMYATR=0x4002_00B0

31 30 29 28 27 26 25 24	23 22 21 20 19 18 17 16	15 14 13 12 11 10 9 8	7 6 5	4 3 2 1 0
Reserved	NO_WRLAT	Reserved	SKIP	Reserved	DUR[4:0]
-	1	-	0	-	00111
-	RW	-	RW	-	RW

24	NO_WRLAT	Must be '1'.
8	SKIP	This bit selects whether the dummy phase is to be skipped or not on XIP reads
		0 Dummy phase is present
		1 Dummy phase is skipped
4 0	DUR	This field reflects the length of the dummy phase of the XIP read in QSPI clock cycles minus 1.

NOTE:

1. The bit[25], bit[20:16] values in QSPI_XIPDMYATR register must be '0'.

15.6.32 QSPI_XIPDATR: QSPI XIP Data Attributes Register

The QSPI_XIPDATR register is a 32-bit read/write accessible register. This register configures how the data portion of the QSPI transaction in XIP mode, shall be output on the QSPI bus.

QSPI_XIPDATR=0x4002_00C0

31 30 29 28 27 26 25 24	23 22 21 20 19 18 17 16	15 14 13 12 11 10 9 8	7	6 5	4	3 2 1 0
Reserved			FLIP	Reserved	DDR	MODE[3:0]
-			0	-	0	0x01
-			RW	-	RW	RW

7	FLIP	Byte order select	
		0	Return data using the same order as the memory chip provided the data in
		1	Flip the order of 4-bits in each 8-bit bytes This bit only has effect in DDR x4 mode and is ignored in all other cases.
4	DDR	This bit selects whether the data phase is executed in DDR mode (using both edges of the clock)	
		0	Normal (STR) mode is used
		1	DDR (DTR) mode is used
3 0	MODE[3:0]	This field selects the number of lines to use for the data output	
		001	Single (x1, 1-bits)
		010	Dual (x2, 2-bits)
		100	Quad (x4, 4-bits)
		Others	Reserved

NOTE:

1. The bit[6:5] values in QSPI_XIPDATR register must be ‘0’.

15.6.33 QSPI_XIPMBR: QSPI XIP Mode Bits Register

The QSPI_XIPMBR register is a 32-bit read/write accessible register. This register is the mode bits phase of the QSPI transaction in XIP mode.

QSPI_XIPMBR=0x4002_00D0

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
MODE[31:0]																															
0x00000000																															
RW																															

31 0	MODE[31:0]	QSPI Mode bits to use for the mode phase of XIP reads. For data widths less than 32 bits, the most significant data bit must be aligned to bit[31].

15.6.34 QSPI_XIPMBATR: QSPI XIP Mode Bits Attributes Register

The QSPI_XIPMBATR register is a 32-bit read/write accessible register. This register configures how the mode bits portion of the QSPI transaction in XIP mode, shall be output on the QSPI bus.

QSPI_XIPMBATR =0x4002_00D4

31 30 29 28 27 26 25 24	23 22 21 20 19 18 17	16	15 14 13	12	11 10 9 8	7 6 5	4 3 2 1 0
Reserved		SKIP	Reserved	DDR	MODE[3:0]	Reserved	DUR[4:0]
-		1	-	0	0000	-	00000
-		RW	-	RW	RW	-	RW

16	SKIP	This bit selects whether the mode phase is to be skipped or not on the next QSPI transaction	
		0	Mode phase is present
		1	Mode phase is skipped
12	DDR	This bit selects whether the mode phase is executed in DDR mode (using both edges of the clock)	
		0	Normal (STR) mode is used
		1	DDR (DTR) mode is used
11 8	MODE[3:0]	This field selects the number of lines to use for the mode output	
		001	Single (x1, 1-bits)
		010	Dual (x2, 2-bits)
		100	Quad (x4, 4-bits)
		Others	Reserved
4 0	DUR[4:0]	This field must reflect the length of the mode phase of the QSPI transaction in QSPI clock cycles minus 1. 8-bit mode in STR x8 mode: 0 8-bit mode in STR x1 mode: 7 8-bit mode in DDR x1 mode: 3 16-bit mode in DDR x8 mode: 0	

NOTE:

1. The bit[13] value in QSPI_XIPMBATR register must be '0'.

15.6.35 QSPI_DMASTAR: QSPI DMA Start Address Register

The QSPI_DMASTAR register is a 32-bit read/write accessible register. To use QSPI DMA, the DMA descriptor must be configured. Refer to section 15.3.6 for configuration details.

QSPI_DMASTAR =0x4002_00E0

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
ADDR[31:0]																															
0x00000000																															
RW																															

31 0	ADDR[31:0]	Set address for the first DMA descriptor. This register can only be changed when DMA is idle. The DMA engine updates this register to the address of the current buffer descriptor being processed.

15.6.36 QSPI_DMAFBLSR: QSPI DMA Master Fixed Burst Length Select Register

The QSPI_DMAFBLSR register is a 32-bit read/write accessible register.

QSPI_DMAFBLSR=0x4002_00E4

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved																														LEN[1:0]	
-																														00	
-																														RW	

1 0	LEN[1:0]	Select AHB Master interface's preferred incremental burst behavior. When LEN[1:0] is not 0x0, undetermined size burst can still be observed due to 1KB boundary, buffer descriptor boundary, data size boundary and ½ FIFO size limit, it will however always be shorter than the preferred burst length.	
		00	undetermined size
		01	Burst 4
		10	Burst 8
		11	Burst 16

15.6.37 QSPI_DMASR: QSPI DMA Status Register

The QSPI_DMASR register is a 32-bit read/write accessible register that provides the current status of the DMA. Bits in this register can trigger interrupts if enabled in the IRQ Control (0x0018) register.

QSPI_DMASR =0x4002_00E8

31 30 29 28 27 26 25 24	23 22 21 20 19 18 17 16	15 14 13 12 11 10 9 8	7 6 5 4 3	2	1	0
Reserved				DERR	DIRQ	DIDLE
-				0	0	1
-				RWC1	RWC1	RO

2	DERR	When set, it indicates that DMA completed with an error. Use software reset to recover from error. Software must clear this bit after error recovery. It is cleared by writing '1' to it.	
		0	Not error
		1	Error (Writing a '1' to the bit clears the flag).
1	DIRQ	When set, it indicates DMA successfully transferred one or more buffer descriptors with IRQ bit set. Software must clear this bit. It is cleared by writing '1' to it.	
		0	No transmission.
		1	Successfully transmission. (Writing a '1' to the bit clears the flag).
0	DIDLE	Indicates DMA is available to process new requests.	
		0	Busy
		1	Idle

15.6.38 QSPI_DMABDCNTR: QSPI DMA Buffer Descriptor Count Register

The QSPI_DMABDCNTR register is a 32-bit read/write accessible register.

QSPI_DMABDCNTR =0x4002_00EC

31 30 29 28 27 26 25 24	23 22 21 20 19 18 17 16	15 14 13 12 11 10 9 8	7 6 5 4 3 2 1 0
CNT[31:0]			
0x000000000			
RO			

31 0	CNT[31:0]	Indicates how many DMA buffer descriptors were successfully transferred during last DMA transfer.

15.6.39 QSPI_WPCR: QSPI Write Protect Control Register

The QSPI_WPCR register is a 32-bit read/write accessible register. Refer to 15.3.3.3.

QSPI_WPCR =0x4002_00F8

31 30 29 28 27 26 25 24	23 22 21 20 19 18 17 16	15 14 13 12 11 10 9 8	7 6 5 4 3 2	1	0
Reserved				WPZ	WP
-				0	1
-				RW	RW

1	WPZ	State of bit 2 during idle (CS inactive):
	0	QSPI output bit 2 is driven during transactions
	1	QSPI output bit 2 is Hi-Z during transactions
0	WP	Value to drive on the data bit 2 during idle (CS inactive). Many memory chips use data bit 2 as a write-protect signal and its value may be set using this register.
	0	Output low in data bit 2.
	1	Output high in data bit 2.

NOTE:

1. QSPI output bit 2 is QSPI_IO2 pin.

15.6.40 QSPI_HTR: QSPI Hold Time Register

The QSPI_HTR register is a 32-bit read/write accessible register. The hold time set before the dummy phase of the QSPI transaction is added.

QSPI_HTR =0x4002_00FC

31 30 29 28 27 26 25 24	23 22 21 20 19 18 17 16	15 14 13 12 11 10 9 8	7 6 5 4 3 2 1 0
Reserved			TIME[7:0]
-			0x07
-			RW

7 0	TIME[7:0]	Specifies the hold time of the output value for the phase preceding the dummy phase during QSPI transactions. The time is given in a number of bus clock cycles. Refer to section 15.3.3.1 and specifically Figure 142.

15.6.41 QSPI Register Map Summary

Table 142. QSPI Register Map Summary

Offset	Register name	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0x00	QSPI_GCR	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	DMAEN	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	SWRST
	Reset value																0																0
0x04	QSPI_CLKCFGR	SP[7:0]								Res	Res	Res	Res	Res	Res	CPOL	TRANS	MINCSL[7:0]								PRD[7:0]							
	Reset value	0	0	0	0	0	1	1	1							0	0	0	0	0	0	0	1	1	1	0	0	0	0	1	1	1	1
0x08	QSPI_MINCSHR	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	MINCSH[7:0]							
	Reset value																									0	0	0	0	0	0	0	0
0x10	QSPI_FSR	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	FULL	EMPTY	FILL[15:0]															
	Reset value															0	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
0x14	QSPI_SR	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	FWERR	FRERR	FWER	FRER	BUSY	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	RDY
	Reset value												0	0	1	0	0																1
0x18	QSPI_ICR	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	DERRIE	DIE	DIDLEIE	Res	Res	Res	Res	Res	FBND	Res	Res	Res	Res	Res	FWERRIE	FRERRIE	FWFILLIE	FRFILLIE	NBUSYIE
	Reset value														0	0	0						0						0	0	0	0	0
0x20	QSPI_CMDR	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	CMD[15:0]															
	Reset value																	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
0x24	QSPI_CMDATR	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	SKIP	Res	Res	Res	DDR	MODE[3:0]				Res	Res	Res	Res	DUR[3:0]			
	Reset value																0				0	0	0	0	1					0	1	1	1
0x30	QSPI_AR	ADDR[31:0]																															
	Reset value	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
0x34	QSPI_AATR	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	MSK	Res	Res	SKIP	Res	Res	Res	DDR	MODE[3:0]				Res	Res	Res	DUR[4:0]				
	Reset value													1	1		0				0	0	0	0	1				1	0	1	1	1

Table 142. QSPI Register Map Summary (continued)

Offset	Register name	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0x38	QSPI_CSR	CS[31:0]																															
	Reset value	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1
0x40	QSPI_DMYATR								NO_WRLAT																SKIP				DUR[4:0]				
	Reset value								1																0				0	0	0	0	0
0x44	QSPI_XIPHOTR	TIME[31:0]																															
	Reset value	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
0x48	QSPI_XIPHOSR																																EN
	Reset value																																0
0x4C	QSPI_XIPHOCR																																EN
	Reset value																																0
0x50	QSPI_DSZR	SIZE[31:0]																															
	Reset value	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
0x54	QSPI_DATR																DIR								SKIP	FLIP			DDR	MODE [3:0]			
	Reset value																0								0	0			0	0	0	0	0
0x58	QSPI_TDR	TDR[31:0]																															
	Reset value	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
0x5C	QSPI_RDR	RDR[31:0]																															
	Reset value	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Table 142. QSPI Register Map Summary (continued)

Offset	Register name	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0x68	QSPI_MBR	MODE[31:0]																															
	Reset value	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
0x6C	QSPI_MBATR	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	SKIP	Res	Res	Res	DDR	MODE[3:0]				Res	Res	Res	DUR[4:0]				
	Reset value																1				0	0	0	0	0				0	0	0	0	0
0x80	QSPI_XIPCR	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	UPDATE	SUSRES	Res	Res	Res	Res	Res	Res	BUSY	EN
	Reset value																							0	0							0	1
0x84	QSPI_XIPACCBNDR	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	BND[4:0]				
	Reset value																												1	1	1	1	1
0x88	QSPI_XIPSZR	Res	Res	Res	Res	SIZE[27:0]																											
	Reset value					0	0	0	0	0	0	0	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1
0x8C	QSPI_XIPWINAR	Res	Res	Res	Res	ADDR[27:0]																											
	Reset value					0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
0x90	QSPI_XIPCMDR	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	RCMD[15:0]															
	Reset value																	0	0	0	0	1	0	1	1	0	0	0	0	0	0	0	0
0x94	QSPI_XIPCMDATR	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	SKIP	Res	Res	Res	DDR	MODE[3:0]				Res	Res	Res	Res	DUR[3:0]			
	Reset value																0				0	0	0	0	1					0	1	1	1
0xA0	QSPI_XIPBAR	ADDR[31:0]																															
	Reset value	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
0xA4	QSPI_XIPAATR	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	MSK[1:0]		WIDTH[1:0]		Res	Res	Res	DDR	MODE[3:0]				Res	Res	Res	Res	Res	Res	Res	Res
	Reset value													1	1	0	0				0	0	0	0	1								

Table 142. QSPI Register Map Summary (continued)

Offset	Register name	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0xA8	QSPI_XIPCSR	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	CS
	Reset value																																1
0xB0	QSPI_XIPDMYATR	Res.	Res.	Res.	Res.	Res.	Res.	Res.	NO_WRLAT	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	SKIP	Res.	Res.	Res.	DUR[4:0]				
	Reset value								1																0				0	0	1	1	1
0xC0	QSPI_XIPDATR	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.		FLIP	Res.	Res.	DDR	MODE[3:0]			
	Reset value																									0			0	0	0	0	1
0xD0	QSPI_XIPMBR	MODE[31:0]																															
	Reset value	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
0xD4	QSPI_XIPMBATR	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	SKIP	Res.	Res.	Res.	DDR	MODE[3:0]				Res.	Res.	Res.	DUR[4:0]				
	Reset value																1				0	0	0	0	0				0	0	0	0	0
0xE0	QSPI_DMASTAR	ADDR[31:0]																															
	Reset value	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
0xE4	QSPI_DMAFBLSR	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	LEN[1:0]	
	Reset value																															0	0
0xE8	QSPI_DMASR	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	DERR	DIRQ	DIDLE
	Reset value																														0	0	1
0xEC	QSPI_DMABDCNTR	CNT[31:0]																															
	Reset value	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
0xF8	QSPI_WPCR	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	WPZ	WP
	Reset value																															0	1
0xFC	QSPI_HTR	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	TIME[7:0]							
	Reset value																									0	0	0	0	0	1	1	1

16. CAN with Flexible Data Rate (CANFD)

16.1 CANFD Introduction

The A34G43x series includes two built-in Controller Area Network with Flexible Data Rate (CANFD) modules.

The CANFD module transmits and receives data according to the specified configuration and reads the current CANFD status. The CANFD protocol can be used for frame transmission (communication objects) and error handling management between network nodes.

The CANFD module supports CAN protocols, including CAN CC (Classic CAN 2.0B) and CAN FD (ISO 11898-1:2015). Interrupts for controlling CANFD communication can be programmed as needed.

Table 143. CANFD Protocol Feature

Specification	Description
CAN 2.0B	CAN CC, FD intolerant
ISO 11898-1:2015	CAN CC, FD tolerant CAN FD

16.2 CANFD Main Features

The CANFD of A34G43x features the following:

- Support CAN specification
  - CAN CC (CAN 2.0B)
    - up to 8 bytes payload,
  - CAN FD (ISO 11898-1:2015)
    - up to 64 bytes payload
- Free programmable data rate:
    - CAN CC defines data rates up to 1 Mbit/s
    - CAN FD is limited by the transceiver and the clock frequency of CAN-CTRL
- Programmable baud rate prescaler
- Configurable receive buffer (RB) size
  - The number of buffer slots up to 23
  - FIFO-like behavior
  - Received frames which are “not accepted” or “incorrect” don’t overwrite already stored frames.
- Two Transmit Buffers
  - Primary Transmit Buffer (PTB) (one frame slot)
  - optional configurable Secondary Transmit Buffer (STB)
    - The STB is optional. The number of buffer slots up to 16.
    - Operation in FIFO or priority decision mode
- Independent and programmable internal acceptance filters
  - Number of acceptance filters selectable by generic parameter in the range of 1 to 16
- Extended features
  - Limitation of re-arbitration and re-transmission
  - Listen Only Mode
  - Loop Back Mode (internal and external)
  - Transceiver Standby Mode

- Extended status and error report
  - Status report for frames, which are commanded to be transmitted
  - Capturing of last occurred kind of error and of arbitration lost position
  - Programmable Error Warning Limit
- APB interface.
- Two-wire interface to external transceiver.
- Error Handling.
- CANFD only SRAM separate from the provided SRAM: 4 KB
- CANFD recommended frequency: 20, 40, 80 MHz
  - When using CANFD 80 MHz, PLL is divided by two at 160 MHz, so the entire system clock is lowered to 160 MHz and used.

16.3 CANFD Functional Description

The CANFD module is a serial communication controller that performs serial communication according to the protocol and meets all constraints of each specification.:

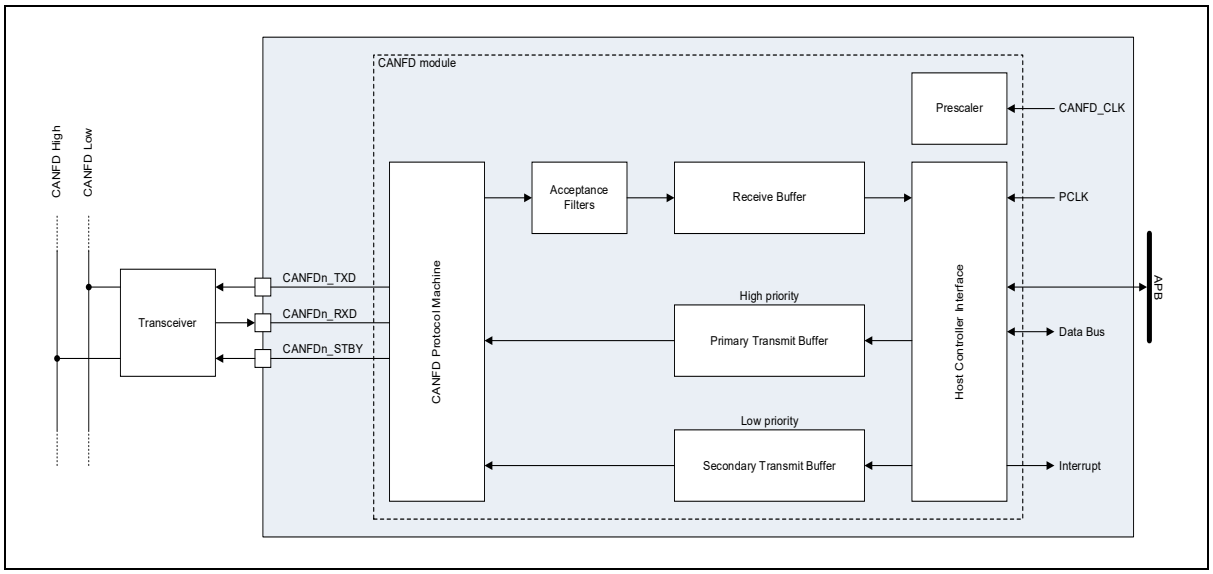
- CAN CC(Classic CAN 2.0B)
- CAN FD(ISO 11898-1:2015)

The CANFD protocol defines the transmission of frames (communication objects) and error handling management between network nodes. Data transmitted and received through the CANFD transceiver can be processed quickly and reliably by the CANFD module.

16.3.1 CANFD Block Diagram

Figure 160 shows a block diagram of the CANFD.

Figure 160. CANFD Block Diagram



16.3.2 CANFD Pins and Signals

Table 144 describes external pins assigned for CANFD.

Table 144. Pin Assignment of CANFD: External Pins

Pin Name	Type	Description	Supported Packages		
			A34G43xYL (120-LQFP)	A34G43xVL (100-LQFP)	A34G43xRL (64-LQFP)
CANFD0_TXD	O	CANFD channel 0 transmit output	O	O	O
CANFD0_RXD	I	CANFD channel 0 receive input	O	O	O
CANFD0_STBY	O	CANFD channel 0 standby output	O	O	O
CANFD1_TXD	O	CANFD channel 1 transmit output	O	O	O
CANFD1_RXD	I	CANFD channel 1 receive input	O	O	O
CANFD0_STBY	O	CANFD channel 1 standby output	O	O	O

Table 145. Internal Input and Output Signal for CANFDn (n = 0 to 1)

Internal Signal Name	Signal Type	Description
PCLK	Input	Peripheral system clock
CANFD_CLK	Input	CANFDn external clock (Selectable with the setting of SCU_MCCR8)

16.3.3 CANFD Transceivers

It is required to use a transceiver as shown in Figure 160 for both real life tests as well as in simulation. CAN communication without a transceiver is impossible. The CANFD module includes a transceiver model for simulation and will work with every CAN transceiver available on the market.

The reasons for the requirement of a transceiver are the following:

A transmitting node needs to read back its own transmitted frame bit-by-bit. Any difference between transmission and reception is a bit error.

- A receiver will generate a dominant ACK bit inside the ACK slot of a frame. If the ACK slot stays recessive, then this is an ACK error.
- After each error, a CAN node will generate an error flag (6 dominant bits) which violates the rule for bit stuffing. Therefore, every node attached to the bus will detect the error flag and will join by generating its own error flag resulting in a superposition of error flags. An error flag invalidates data frames (and remote frames).

These basic rules apply to CAN CC and CAN FD.

Table 146. CAN Transceiver Overview

Transceiver	Short Name	Description
CAN HS	CAN high-speed transceiver	CAN CC up to 0.5 - 1 Mbit/s
CAN FD	CAN FD transceiver	CAN FD with bit rate switching with up to 2-3 Mbit/s
CAN FD SIC	CAN FD transceiver with signal improvement capability	CAN FD with bit rate switching with up to 5-8 Mbit/s

All CAN CC and CAN FD frames can coexist on the same network, allowing a step-by-step upgrade or migration of the existing CAN network. Most transceivers are pin-compatible, enabling replacement without PCB redesign. However, the slowest and least capable transceiver in the network may limit overall performance.

For example, in a CAN CC network, a CAN HS transceiver should be used. If faster CAN FD frames are transmitted, the CAN HS transceiver should be replaced with a CAN FD or CAN FD SIC transceiver.

16.3.4 Frame Type

The main differences between CAN CC and CAN FD are as follows:

- CAN CC supports a maximum data payload of 8 bytes, whereas CAN FD supports up to 64 bytes.
- CAN CC allows only a single configurable bit rate, while CAN FD supports two bit rates: slow for the arbitration phase and fast for the data phase.

All frame types for CAN CC and CAN FD are shown in Figure 161. The CAN specifications and abbreviations are briefly described in Table 147.

Figure 161. CAN CC, CAN FD Frame Types

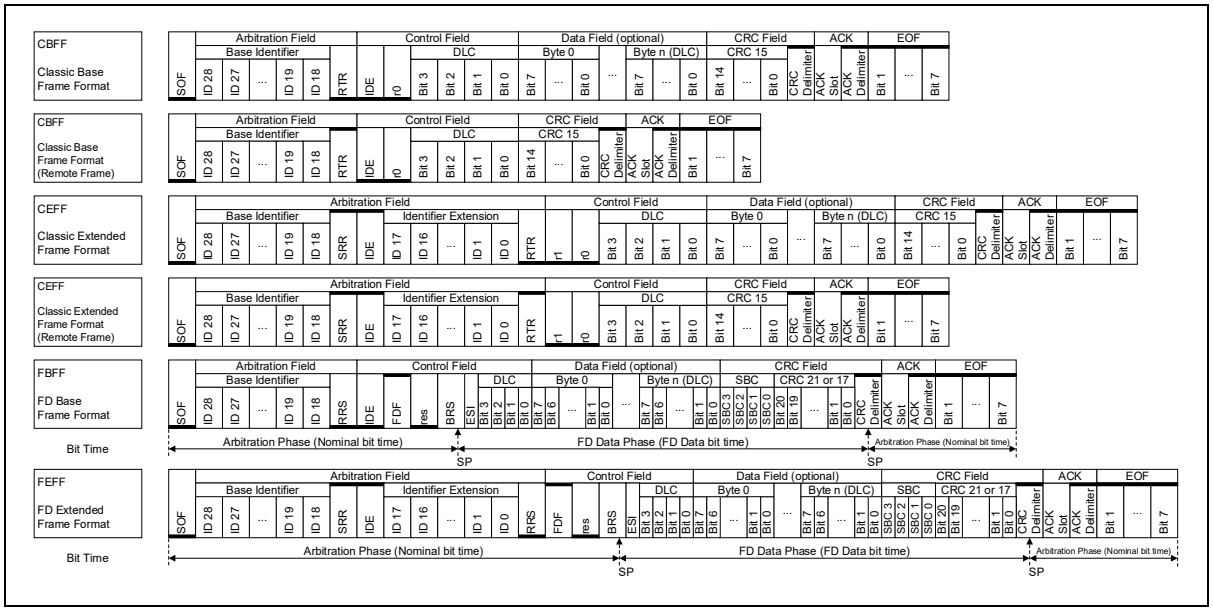


Table 147. Frame Identifier

Frame Type	Identifier used by the Frame	Identifier
CBFF	Base Identifier (Base ID)	ID(28:18)
CEFF	Base Identifier (Base ID) and Identifier Extension (Extended ID)	ID(28:0)
FBFF	Base Identifier (Base ID)	ID(28:18)
FEFF	Base Identifier (Base ID) and Identifier Extension (Extended ID)	ID(28:0)

Table 148. Bit Abbreviations

Abbreviation	Description	Comment
ID	Identifier	
RTR	Remote Transmission Request	Remote or Data frame
SRR	Substitute Remote Request	
RRS	Remote Request Substitution	
IDE	Identifier Extension	Standard or Extended frame
DLC	Data Length Code	Number of payload bytes
EDL	Extended Data Length	
FDF	FD Format indicator (=EDL)	
BRS	Bit Rate Switch	
ESI	Error State Indicator	
r1, r0, res	Reserved bits	

Table 149. Definition of DLC

DLC[3:0] (binary)	Frame Type	Payload in Bytes
0000 to 1000	CAN CC and CAN FD	0 to 8
1001 to 1111	CAN CC	8
1001	CAN FD	12
1010	CAN FD	16
1011	CAN FD	20
1100	CAN FD	24
1101	CAN FD	32
1110	CAN FD	48
1111	CAN FD	64

16.3.5 Frame Buffer

Figure 162 shows the Frame Buffer. The Receive Buffer (RB) supports up to 23 buffers, while the Transmit Buffer (TB) can transmit frames using two different methods. The Secondary Transmit Buffer (STB) supports up to 16 buffers, whereas the Primary Transmit Buffer (PTB) operates as a single buffer.

Table 150 provides a description of the defined frames for TB and RB.

Figure 162. CANFD Frame Buffer Diagram

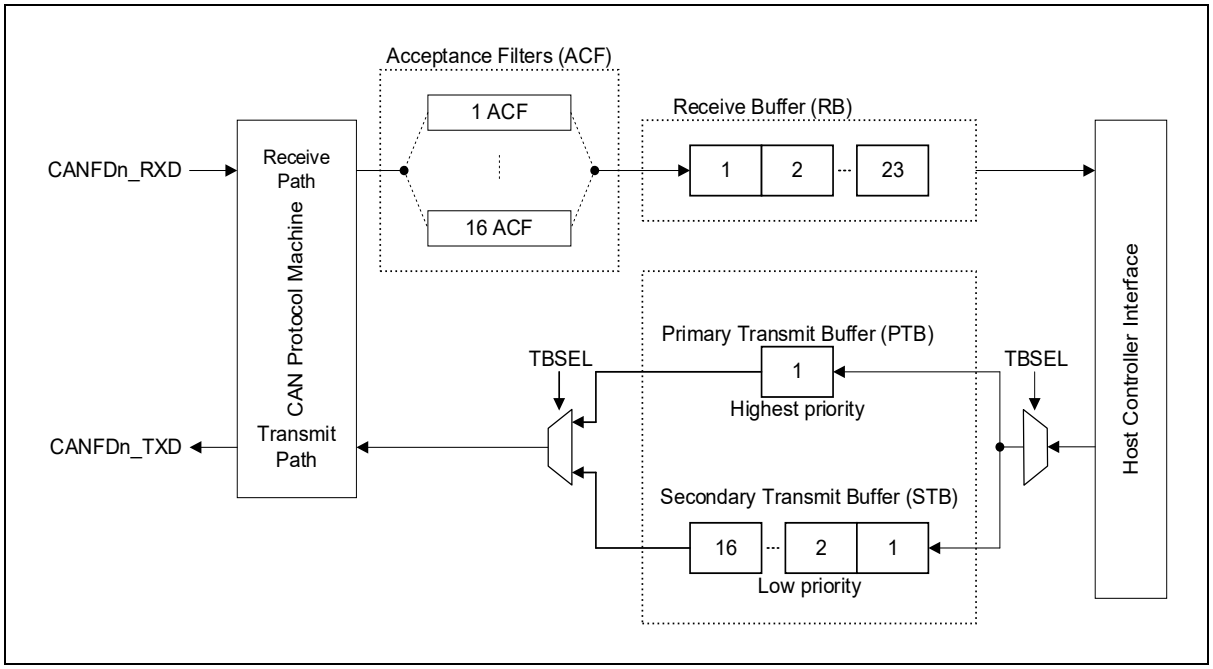


Table 150. Definition of Unified Frame for TB, RB

Element	Name	Function	Register
Receive Buffer	RB	Identifier	CANFDn_RBIR
		Format	CANFDn_RBFR
		Type	CANFDn_RBTR
		Payload	CANFDn_RBPDR
Transmit Buffer	TB	Identifier	CANFDn_TBIR
		Format	CANFDn_TBFR
		Type	CANFDn_TBTR
		Payload	CANFDn_TBPDR

16.3.5.1 Receive Buffer (RB)

To enhance the processing capability of received frames, frames that match the acceptance filter conditions are stored in the Receive Buffer (RB). When the next frame is received, the internal pointer of the RB-related address shifts its position in a FIFO-like manner. The total number of available RBs is 23, and frames are always read in the order they were received, similar to FIFO operation.

The reception status of frames can be checked using the RSTAT[1:0] bits in the CANFDn_CSR register. Interrupts are available depending on the number of frames stored in the RB.

For details regarding interrupts, refer to Table 158.

16.3.5.2 Transmit Buffer (TB)

Two Transmit Buffers (TB) are provided for frame transmission. The Primary Transmit Buffer (PTB) has a higher priority than the Secondary Transmit Buffer (STB) but can only use a single frame buffer. On the other hand, the Secondary Transmit Buffer (STB) utilizes a total of 16 frame buffers, and its position changes based on an internal pointer for the TB-related address, similar to a FIFO.

The PTB always holds a higher priority than the STB and operates independently of CAN bus arbitration. CAN bus arbitration determines priority based on transmit frame identifiers.

The selection of whether the Transmit Buffer (TB) operates as PTB or STB can be made using the TBSEL bit in the CANFDn_CSR register.

The STB supports two transmission modes. Depending on the TSMODE bit in the CANFDn_CSR register, either FIFO mode or Priority mode can be selected.

In FIFO mode, frames are transmitted in the order they were stored in the Transmit Buffer. In Priority mode, transmission begins with the frame that has the highest priority based on its identifier. Additionally, the STB allows for the transmission of either a single frame (TSONE) or all stored frames (TSALL) using the TSONE bit and TSALL bit in the CANFDn_CSR register.

PTB transmission can halt and delay STB transmission. Once the higher-priority PTB frame than STB has been successfully transmitted, STB transmission automatically resumes.

The STB transmission can be stopped using PTB transmission in the following cases:

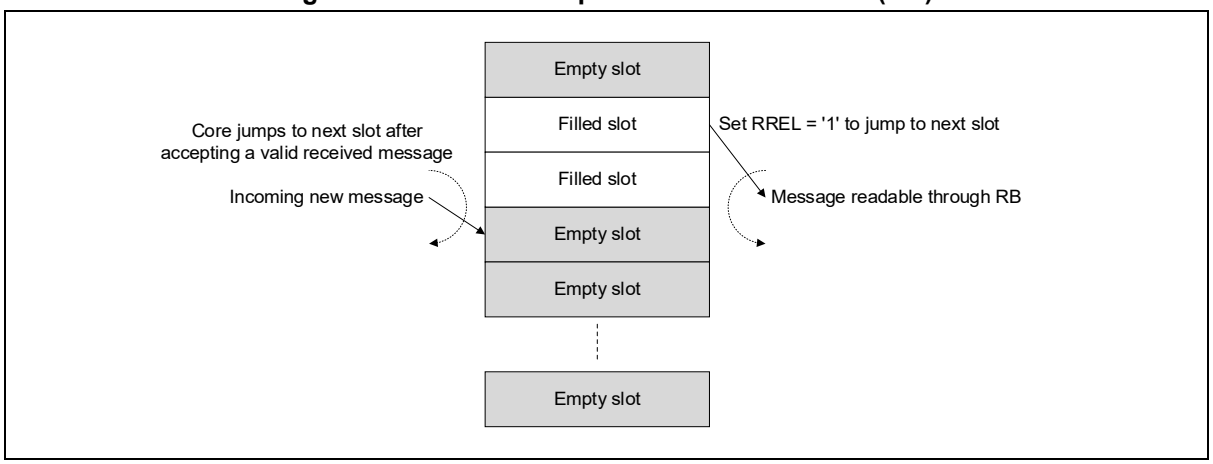
- When PTB transmission is set after configuring the transmission of all frames stored in the STB (TSALL) but before the completion of all STB transmissions.
- When PTB transmission is set after the STB is configured for single-frame transmission (TSONE) but before the STB transmission starts or before re-transmission or re-arbitration occurs.

16.3.5.3 Frame Reception

The received frame is stored in the Receive Buffer (RB) if it passes the acceptance filter configured as shown in Figure 163.

The size of the RB determines the number of payload data bytes in the frame based on the received DLC[3:0] in the CANFDn_RBFR register. When the RIE bit in the CANFDn_IER register is set to ‘1’, the RIF bit in the CANFDn_ISR register is set to ‘1’ upon frame reception. The RSTAT field in the CANFDn_CSR register indicates the status based on the number of received buffers. If all receive buffers are full and the RFIE bit in the CANFDn_IER register is enabled (‘1’), the RFIF bit in the CANFDn_ISR register is set to ‘1’.

Figure 163. Frame Reception of Receive Buffer (RB)



RB operates similarly to FIFO, mapping the earliest received frame slot to the RB register. To read the next slot in RB, the RREL bit in the CANFDn_CSR register must be set to ‘1’. When all RB slots have been received, if the ROM bit in the CANFDn_CSR register is set to ‘0’, the old frame is overwritten.

On the other hand, if the ROM bit in the CANFDn_CSR register is set to ‘1’, the received frame is not stored in RB. In this case, setting the ROIE bit in the CANFDn_ISR register to ‘1’ causes the ROIF bit in the CANFDn_IER register to be set to ‘1’, indicating that RB has overflowed.

16.3.5.4 Handling Frame Reception

To receive a frame, the RIE bit in the CANFDn_IER register is typically set to '1'. If an acceptance filter is not used, additional resources are required to verify whether the received frame is valid. The acceptance filter allows selective reception of frames. (For details on the acceptance filter, refer to section 16.3.6.)

To reduce the frequency of receive interrupts, the RAFIE (Receive buffer Almost Full) bit or the RFIE (Receive buffer Full) bit in the CANFDn_IER register can be used. The RAFIE bit generates an interrupt flag by comparing 23 RB (Receive Buffer) entries with the AFWL[3:0] (Receive buffer Almost Full Warning Limit) bits in the CANFDn_ERR register.

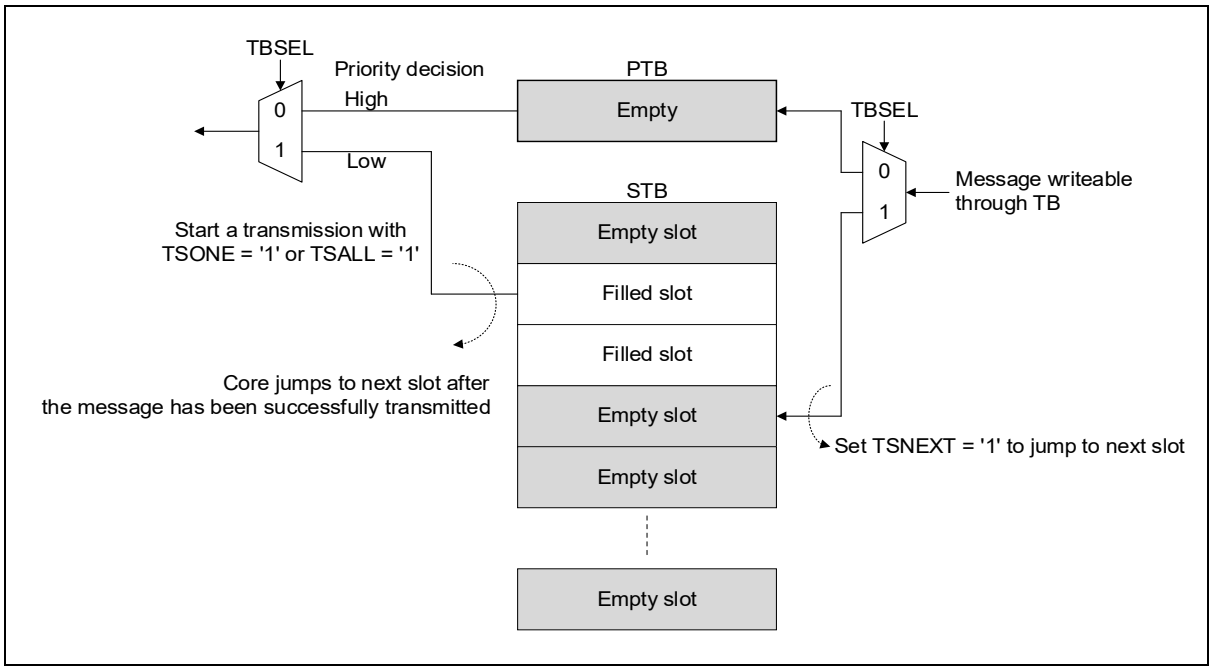
The RB read operation is performed as follows:

- The oldest stored frame in the RB FIFO is read using the RB register.
- When the RREL bit in the CANFDn_CSR register is set to '1,' the RB slot advances to the next slot, and the RB register is automatically updated.
- The above operation should be repeated until the RSTAT[1:0] bits in the CANFDn_CSR register indicate an empty status.

16.3.5.5 Frame Transmission

Before transmitting a frame, the transmit buffer must select either PTB or STB. The TB register is accessible by both PTB and STB. Refer to Figure 164.

Figure 164. Frame Transmission of Transmit Buffer (PTB, STB)



- Select PTB or STB using the TBSEL bit in the CANFDn_CSR register.
- Write the frame to be transmitted into the TB register.
- When the TSNEXT bit in the CANFDn_CSR register is set to '1', the operation proceeds to the next slot in STB mode, allowing a new frame to be written to the TB register for transmission. In contrast, in PTB mode, the operation is ignored, and the TXNEXT bit in the CANFDn_CSR register is automatically cleared.
- PTB transmits a frame when the TPE bit in the CANFDn_CSR register is set to '1'.
- STB can transmit either one stored frame (TSONE) or all stored frames (TSALL) using the TSONE or TSALL bit in the CANFDn_CSR register.

To check the transmission status, set the TPIE or TSIE bit in the CANFDn_IER register. The status can be verified through the TPIF or TSIF bit in the CANFDn_ISR register.

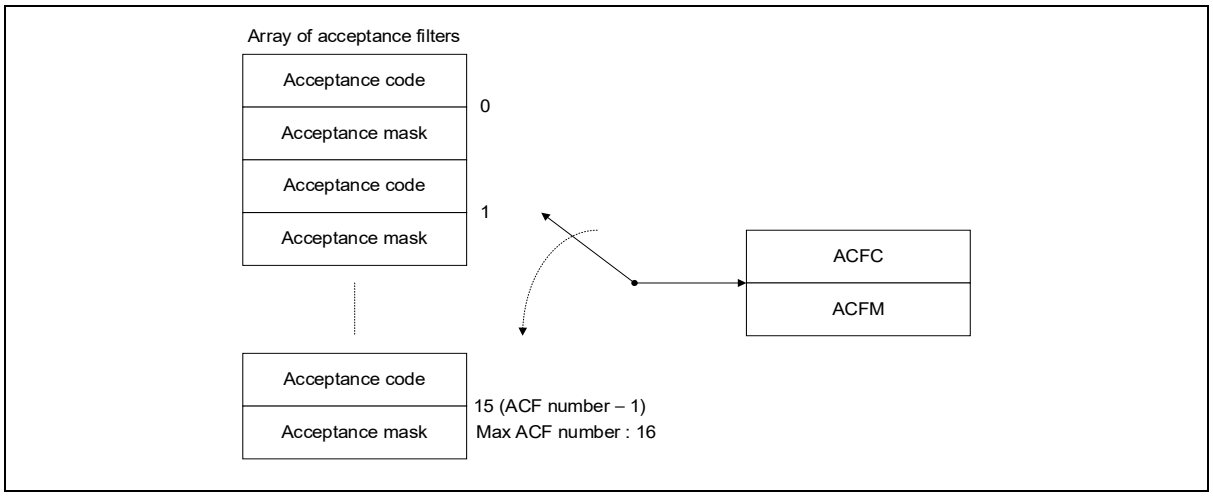
- For PTB, when the TPIE bit in the CANFDn_IER register is set to '1', the TPIF bit in the CANFDn_ISR register is set to '1' after transmitting one frame.
- For STB using the TSONE bit in the CANFDn_CSR register, when the TSIE bit in the CANFDn_IER register is set to '1', one frame is transmitted, and the TSIE bit in the CANFDn_ISR register is set to '1'.
- For STB using the TSALL bit in the CANFDn_CSR register, when the TSIE bit in the CANFDn_IER register is set to '1', all stored frames are transmitted, and the TSIE bit in the CANFDn_ISR register is set to '1'.

16.3.6 Acceptance Filters (ACFM, ACFC)

The Acceptance Filters (ACF) can be used up to a maximum of 16 and allows to selectively receive Receive frames. A frame is received and stored in the Receive Buffer (RB) if conditions match the configured Acceptance Filter. Only the necessary frames are received through the Acceptance Filter, eliminating the need to verify whether additionally received frames are unnecessary.

If the RIE bit in the CANFDn_IER register is set to '1,' the RIF bit in the CANFDn_ISR register will be set to '1'. If a received frame does not match any Acceptance Filter, the RIF bit in the CANFDn_ISR register will not be issued, even if the RIE bit in the CANFDn_IER register is set to '1,' and the received frame will not be stored in the RB. Only frames that match an Acceptance Filter will be stored. In other words, frames matching the Acceptance Filter are stored in the receive buffer, while non-matching frames are not stored in the receive buffer.

Figure 165. Access to Acceptance Filters



The acceptance filter consists of the Acceptance Filter Mask (ACFM) and Acceptance Filter Code (ACFC).

The ACFM is used when comparing a received frame with the ACFC. The necessity of an acceptance check is determined based on the bit configuration of the ACFM. If a mask bit in the ACFM is set to '1,' the acceptance check is bypassed. Conversely, if a mask bit is set to '0,' the acceptance check is performed.

The ACFC selectively receives or discards the receive frames based on the mask bits set in the ACFM.

The acceptance filter number for ACFM and ACFC can be selected using the ACFADR[3:0] bits in the CANFDn_AFCR register. Since ACFM and ACFC are modified using an internal pointer, changing the acceptance filter number via the ACFADR[3:0] bits in the CANFDn_AFCR register does not affect the fixed address.

The structure of ACFM and ACFC is as shown in the table below, and the initial reset values vary depending on the filter number. The initial filter number is set to 0, and to receive all frames, ACFM is initialized to 0xFFFF_FFFF. Other filter numbers can be modified as needed.

Table 151. Definition of Unified Frame for ACF

Element	Name	Function	Register
Acceptance Filter Code	ACFC	Identifier	CANFDn_AFCIR
		Format	CANFDn_AFCFR
		Type	CANFDn_AFCTR
Acceptance Filter Mask	ACFM	Identifier	CANFDn_AFMIR
		Format	CANFDn_AFMFR
		Type	CANFDn_AFMTR

Table 152. Acceptance of Filter Reset Value

Number	Acceptance filter	Reset Value
0	ACFC	0x0000_0000
	ACFM	0xFFFF_FFFF
1 ~ 15	ACFC	0x0000_0000
	ACFM	0x0000_0000

16.3.7 Error Handling

The CANFD module automatically processes errors in the CAN network, ensuring that the microcontroller core is not burdened with error management.

The key features of the CANFD module include re-transmission and automatic deletion of received frames with errors. If error information is required, an interrupt can be triggered by enabling the CANFDn_IER register.

- Re-transmission: When an error occurs, the CANFD module automatically re-transmits the frame.
- Automatic deletion of received frames: The CANFD module automatically deletes the received frames with errors.

However, if the host requires, the CANFD module can provide detailed information about errors and notify the host via interrupt whenever an error occurs. This allows the host to run applications such as a CAN bus monitor.

Every CAN node has three states of error handling:

- Error Active
  - The node automatically transmits active error frames upon detection of an error.
- Error Passive
  - The node transmits a passive error frame upon detection of an error. This means that it does not transmit a dominant value to the bus but expects an error frame transmitted by other CAN nodes.
  - Error-passive nodes are allowed to transmit frames, but if they have been the transmitter of the previous frame, they are required to suspend a further frame transmission for 8 bits following the intermission.
- Bus Off
  - After too many errors, a node goes “bus off” where it stops touching the bus.

The CAN node has error counters for transmit (TECNT[7:0] bits in the CANFDn_ERR register) and receive (RECNT[7:0] bits in the CANFDn_ERR register) to process three types of error states. Both counters increase and decrease as defined in the CAN specification. When the node reaches the counter levels specified in the CAN specification, it enters the appropriate error state. Refer to Table 153.

The error counters are incremented if there are errors. Dangerous errors that are probably caused by this node will result in incrementing the counter by 8, errors that are probably caused by other nodes will result in incrementing the counter by 1. Valid frame transmission or valid receptions result in decrementing the counters. All this is defined by the CAN specification and handled automatically by CANFD module.

An error frame (being the result of error flags) is different to a data frame. It is a dominant pulse for at least 6 consecutive dominant bits, which is a stuff bit violation for other nodes. If a CAN node detects this violation, then it also transmits an error flag resulting in an error frame which is the superposition of error flags from all nodes. (As a result, the duration of error frames is 6-to-12-bit times.) A host controller cannot command transmit an error frame. It is done fully automatically by CANFD module.

If CANFD module is commanded to transmit a frame then it automatically tries re-transmissions as fast as possible until the frame gets transmitted without error or the node goes bus off. If a frame is received and CANFD module detects an error, then the received data is discarded. Because of the automatically transmitted error frame the sender of the frame will re-transmit the frame. Frames in RB are never overwritten by frames with errors. Only valid received frames may result in RB overflow.

Re-transmission is limited by TECNT[7:0] bits in CANFDn_ERR register (if the node goes bus-off) but can be further limited using RETLIM[7:0] bits in CANFDn_PTDCR register.

Table 153. States of CAN Node

State	TECNT[7:0]	RECNT[7:0]	EPASS	BUSOFF
Error active	both smaller than 128		0	0
Error passive	one of them not smaller than 128		1	0
Bus off	bigger than 255	-	1	1

NOTES:

1. In the case of an ACK error, TECNT[7:0] is incremented in the active state but not in the passive state.
2. In the case of a Bit error, TECNT[7:0] may be incremented in the passive state.

16.3.8 Bus Off State

The bus off state is signaled using the BUSOFF bit in CANFDn_CSR register. A CAN node enters the “bus off” state automatically if its TECNT[7:0](transmit error counter) bits in CANFDn_ERR register becomes > 255. Then it will not take part in further communications until it returns to the error-active state again. Setting BUSOFF to ’1’ also sets the EIF bit in CANFDn_ISR register interrupt if EIE bit in CANFDn_IER register is enabled. A CAN node returns to error-active state if it is reset by a power-on reset or if it receives 128 sequences of 11 recessive bits (recovery sequences). Please note that between each recovery sequence the bus may have dominant state. That the minimum time between two valid frames is sufficient to be recognized as recovery sequence.

In the bus off state, RECNT[7:0] and TECNT[7:0] bits in CANFDn_ERR register stay unchanged. Please note that while entering bus off state TECNT[7:0] rolls over and therefore may hold a small value. Therefore, it is recommended to use TECNT[7:0] before the node enters bus off state and status bit BUSOFF afterwards.

If the node recovers from “bus off” state then RECNT[7:0] and TECNT[7:0] are automatically reset to ‘0’. If a frame is pending for transmission but the CAN node has entered bus off state, then this frame is kept pending. If a node returns to error-active state after bus off, then the transmission of the pending frame will be retried. If this is not desired, then the frame should be aborted by the host controller.

16.3.9 Extended Status and Error Report

During CAN bus communication errors may occur. The following features support detection and analysis of them. This can be used for extended bus monitoring.

16.3.9.1 Programmable Error Warning Limit

Errors during reception/transmission are counted by RECNT[7:0] bits and TECNT[7:0] bits in CANFDn_ERR register. A programmable error warning limit EWL[3:0] bits in CANFDn_ERR register, can be used by the host controller for flexible reaction on those events. The limit values can be chosen in steps of 8 errors from 8 to 128:

Limit of count of errors = (EWL+1) * 8

The interrupt EIF bit in CANFDn_ISR register will be set if enabled by EIE bit in CANFDn_IER register under the following conditions:

- The border of the error warning limit has been crossed in either direction by RECNT[7:0] or TECNT[7:0] or the BUSOFF bit has been changed in either direction.
- The BUSOFF bit has been changed in either direction.

16.3.9.2 Arbitration Lost Capture (ALC)

The loss of arbitration is not an error and therefore not relevant for TECNT[7:0] and RECNT[7:0] bits in CANFDn_ERR register but also unwanted.

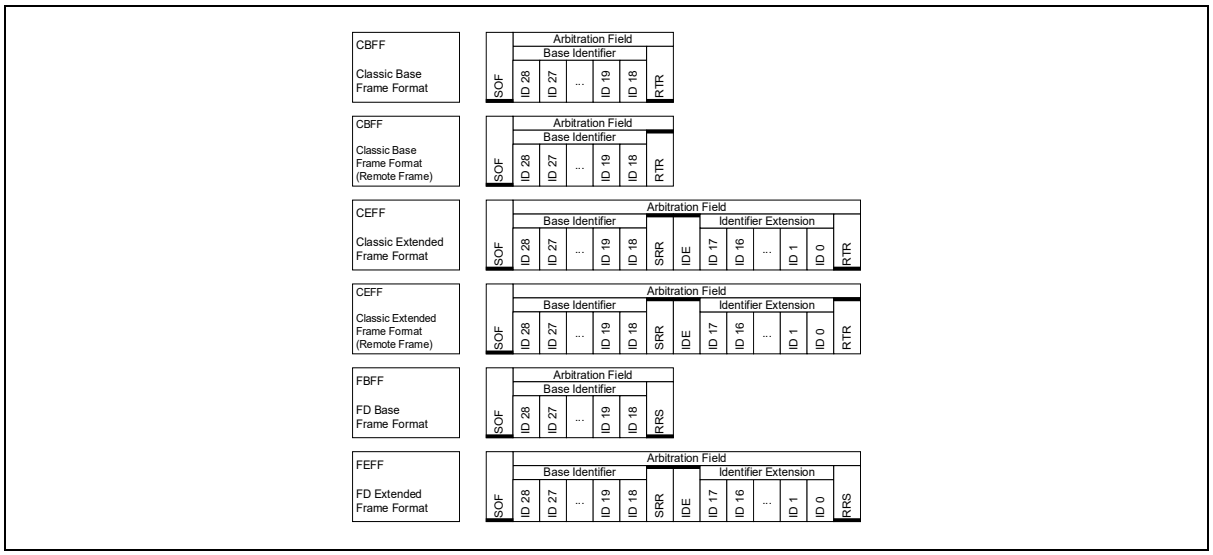
CANFD module can detect the exact bit position in the arbitration field where the arbitration has been lost. This event can be signaled by the ALIF bit in CANFDn_ISR register if it is enabled by ALIE bit in CANFDn_IER register set to '1'. The value of ALC[4:0] bits in CANFDn_ERR register stays unchanged until another event of arbitration loss occurs.

The value of ALC[4:0] is defined as follows:

- ALC[4:0] provides the bit position in the frame where the arbitration has been lost.
- A frame starts with the SOF bit and then the first bit of the ID is transmitted.
  - If the first ID bit (ID28) is arbitration, ALC[4:0] is set to '0'.
  - If the second ID bit (ID27) is arbitration, ALC[3:0] is set to '1'.

Arbitration is only allowed in the arbitration field. See Figure 166.

Figure 166. ALC Field in CAN Frame



Therefore, the maximum value of ALC[4:0] in CANFDn_ERR register is 31 which is the RTR/RRS bit in extended frames.

Example

If a standard remote frame arbitrates with an extended frame, then the extended frame loses arbitration at the IDE bit and ALC[4:0] will be 12. The node transmitting the standard remote frame will not notice that an arbitration has been taken place, because this node has won. It is impossible to lose arbitration outside of the arbitration field. Such an event would be a bit of an error.

16.3.9.3 Kind of Error (KOER)

CANFD module recognizes errors on the CAN bus and stores the last error event in the KOER[2:0] bits. A CAN bus error can be signaled by the BEIF bit in CANFDn_ISR register if it is enabled with BEIE bit in CANFDn_IER register set to '1'. Every new error event overwrites the previous stored value of KOER[2:0]. Therefore, the host controller must react quickly to the error event.

KOER[2:0] is updated with each new error. Therefore, it stays untouched when frames are successfully transmitted or received. This opens the opportunity for a delayed error investigation.

16.3.9.4 Reception of All Data Frames (RBALL)

If RBALL in CANFDn_CSR register set to '1' then frames with error will also be stored inside RB. The LLC(Logical Link Control) frame also includes the KOER[2:0] information. This holds also for loop back mode. Also, acceptance filtering is disabled. Error frames or overload frames are not stored.

Most of the possible errors can only occur if the node is the transmitter of the frame. In this case a frame would only be stored in RB if a loop back mode is activated. Depending on the type of error parts of the frame stored in an RB slot may be valid while other parts are unknown. Table 154 lists the possibilities.

Table 154. RBALL and KOER[2:0]

KOER	Condition	Description
NO ERROR	All	Successful reception
BIT ERROR	Receiver	Can only occur in the ACK slot. All stored data are valid.
	Transmitter	The payload is always invalid. The header bits including the ID may be valid. In the arbitration phase detection of a wrong bit is part of the arbitration and therefore not a BIT ERROR. But if a stuff bit in the arbitration phase is detected wrong by the transmitter of the frame, then this is a BIT ERROR and in this case the header bits are invalid. Therefore, the header bits cannot be trusted, but if the header matches an expected header of a frame, it can be used for further investigation.
FORM ERROR	All	Only FORM ERRORs in the data frame are covered. This includes errors in the CRC delimiter, the ACK delimiter or the EOF bits. All stored data are valid.
STUFF ERROR	Receiver	The position of a STUFF ERROR is unknown. All stored data are invalid.
	Transmitter	Can only happen during arbitration. All stored data are invalid.
ACK ERROR	Receiver	Can only occur if the node is in LOM. All stored data are valid.
	Transmitter	Can only occur in loop back mode without self-ACK. All stored data are valid.
CRC ERROR	Receiver	All stored data are invalid.
OTHER ERROR	All	Dominant bits after own error flag, received active Error Flag too long, dominant bit during Passive-Error-Flag after ACK error

Please note that even if Table 154 states that (parts of) the data are valid under certain conditions, then care must be taken. There is no guarantee for this, and it depends on the reason for the error if this statement is correct.

Example: if the node has a bad bit timing configuration, then there is no correct synchronization and therefore no valid data. Therefore, the statement that (parts of) the data is valid signals only the possibility that it is so.

If RBALL in CANFDn_CSR register set to '1', then storing a frame in RB can only be completed after a final decision about if the frame is valid or there was an error. This adds a few clocks for data storage before an interrupt for reception will be set.

16.3.9.5 Transmit Status

There are various status and interrupt bits, which can be used to keep track of the status of frame transmissions. In addition to that comprehensive information is also provided by TSTAT1[2:0] and TSTAT2[2:0] bits in CANFDn_TSR register. CANFDn_TSR register used to track frame transmission, not part of the CAN protocol.

TSTAT1[2:0] holds information about the frame which is currently in transmission. As a consequence, the content of TSTAT1[2:0] is highly volatile. Once a frame transmission is successfully completed or stopped because of reaching the re-arbitration limit or the re-transmission limit (defined in CANFDn_PTDCR register), then this information is stored in TSTAT2[2:0]. In other words, TSTAT2[2:0] is updated when a frame reaches a final state.

A transmit status provides the following information:

- A handle to identify the frame
- The status of the transmission of the frame referenced by the handle.
- The related timestamp of the frame referenced by the handle.

TSTAT1[2:0] and TSTAT2[2:0] bits in CANFDn_TSR register encode the status of a frame as defined in Table 155.

Table 155. TSTAT Status Encoding

Value	Label	Description
000	IDLE	No transmission in progress
001	ONGOING	Transmission active without any issues
010	LOST_ARBITRATION	Arbitration lost, re-arbitration may take place with respect to REALIM[2:0] bits in CANFDn_PTDCR register.
011	TRANSMITTED	Transmission successfully completed
100	ABORTED	Transmission aborted (TPA or TSA bit in CANFDn_CSR register).
101	DISTURBED	Transmission error. Re-transmission may take place with respect to RETLIM[2:0] bits in CANFDn_PTDCR register.
110	REJECTED	Misconfiguration of the frame format in the LLC frame.
111	-	reserved

There are several possible events which may cause an update of TSTAT1[2:0] or TSTAT2[2:0] or both. Table 156 gives a comprehensive overview of these events.

Table 156. TSTAT Status Events

TSTAT1	TSTAT2	Description
IDLE	IDLE	Power-On reset or RESET bit in CANFDn_CSR register
ONGOING	Unchanged	Start of a transmission
IDLE	TRANSMITTED	Successful completion of a transmission
IDLE	ABORTED	Transmission aborted (TPA or TSA bit in CANFDn_CSR register).
LOST_ARBITRATION	Unchanged	Arbitration lost, re-arbitration limit REALIM[2:0] bits in CANFDn_PTDCR register not reached.
IDLE	LOST_ARBITRATION	Arbitration lost, re-arbitration limit REALIM[2:0] bits in CANFDn_PTDCR register reached
DISTURBED	Unchanged	Error, re-transmission limit RETLIM[2:0] bits in CANFDn_PTDCR register not reached
IDLE	DISTURBED	Error, re-transmission limit RETLIM[2:0] bits in CANFDn_PTDCR register reached
IDLE	REJECTED	Misconfiguration in the LLC frame detected at start of transmission

16.3.10 Remote Mode

Remote mode is used to request data from the transmitter side to the receiver. When specific data is required at the transmitter side, a remote frame can be sent to request the data.

The structure of a remote frame is almost identical to that of a standard CAN frame format, with the main difference being the absence of a Data field. For a comparison of the structures, refer to Figure 161. The reception process for a remote frame is the same as the receive method.

When the RMF bit in the CANFDn_RBFR register is set to ‘1,’ it indicates that the remote mode has been received. Similarly, when the RMF bit in the CANFDn_TBFR register is set to ‘1,’ it transmits in remote mode.

Selective reception is possible using the Acceptance Filter. For more details, refer to Figure 165.

The following describes the operation process of the remote frame:

- Remote Frame Transmission at the transmitter side: A node requests data to another node using a Remote Frame. This frame includes the identifier (ID) of the requested data and the DLC[3:0] value, which specifies the data length.
- Response at the receiver side: Upon receiving the Remote Frame, the node retrieves the corresponding data from its database and responds with a Data Frame.
- Data Reception at the transmitter side: The transmitter processes the received Data Frame from the receiver to obtain the required data.

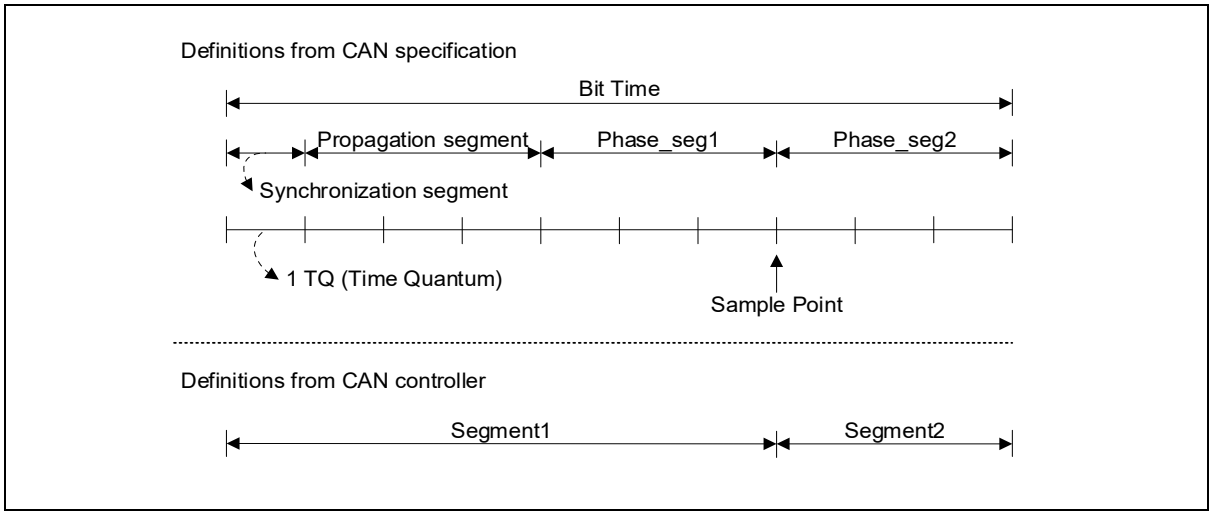
16.3.11 Bit Time

16.3.11.1 Definitions

The CAN CC supports a maximum bit rate of 1 Mbit/s, while CAN FD can be configured to support up to 8 Mbit/s. The CAN Bit Time can be configured using the CANFDn_ACBTR register and the CANFDn_FDBTR register.

The CANFDn_ACBTR register determines the arbitration and bit timing of the CAN CC, whereas the CANFDn_FDBTR register defines the flexible data bit timing.

Figure 167. CAN Bit Timing Specifications



The CAN Bit Time (BT) consists of multiple segments. Each segment is composed of multiple time quantum (n_{TQ}). The duration of a Time Quantum (t_{TQ}) is as follows:

The duration of a time quantum, t_{TQ}, is calculated as follows:

$$t_{TQ} = \frac{n_{prescaler}}{\text{CANFD_CLK}}$$

The values of nTQ and $n_{prescaler}$ should be selected according to CANFD_CLK to match the desired Bit Time (BT).

The Baud Rate (BR) is given by BR = 1/Bit Time.

$$\text{Bit Time} = n_{TQ} \times t_{TQ} = n_{TQ} \times \frac{n_{prescaler}}{\text{CANFD_CLK}} = t_{seg1} + t_{seg2}$$

The Synchronization segment, Propagation segment, and Phase_Seg1 are grouped into Segment 1. The duration of Segment 1 is referred to as t_{seg1}, while the duration of Segment 2 is referred to as t_{seg2}.

Synchronization Jump Width (SJW) defines the synchronization jump width. Synchronization operates in the steps of 1 Time Quantum (TQ). The purpose of synchronization is to ensure that the signal level transition of the received data stream falls within the Synchronization Segment, thereby adjusting the sample point to a stable position in the received signal.

If the transition occurs earlier or later than expected, the CAN node decreases or increases the number of CAN bits by removing or adding a few TQs. SJW limits the number of TQs that can be used for a single synchronization event.

16.3.11.2 Configuration

Table 157 describes the configuration of Bit Time. In general, bit timing can set the prescaler, segment1, segment2, and SJW. Flexible data bit timing may require additional TDC configuration.

Table 157. CAN Bit Timing Configuration

Setting	Bit Name	Register	Purpose	Description
t_{SEG_1}	AC_SEG1[8:0]	CANFDn_ACBTR	Segment 1	Nominal bit rate (Arbitration phase and CAN CC)
	FD_SEG1[7:0]	CANFDn_FDBTR		FD data bit rate (CAN FD)
t_{SEG_2}	AC_SEG2[6:0]	CANFDn_ACBTR	Segment 2	Nominal bit rate (Arbitration phase and CAN CC)
	FD_SEG2[6:0]	CANFDn_FDBTR		FD data bit rate (CAN FD)
t_{SJW}	AC_SJW[6:0]	CANFDn_ACBTR	Synchronization Jump Width	Nominal bit rate (Arbitration phase and CAN CC)
	FD_SJW[6:0]	CANFDn_FDBTR		FD data bit rate (CAN FD)
$n_{prescaler}$	PRS[4:0]	CANFDn_PTDCR	Prescaler	CANFD_CLK / (PRS[4:0] + 1)

The following shows how the Bit Timing Register is related to segment1, segment2, Synchronization Jump Width, and the Prescaler during configuration:

$$t_{seg1} = (n_{SEG1} + 2) \times TQ$$

$$t_{seg2} = (n_{SEG2} + 1) \times TQ$$

$$t_{SJW} = (n_{SJW} + 1) \times TQ$$

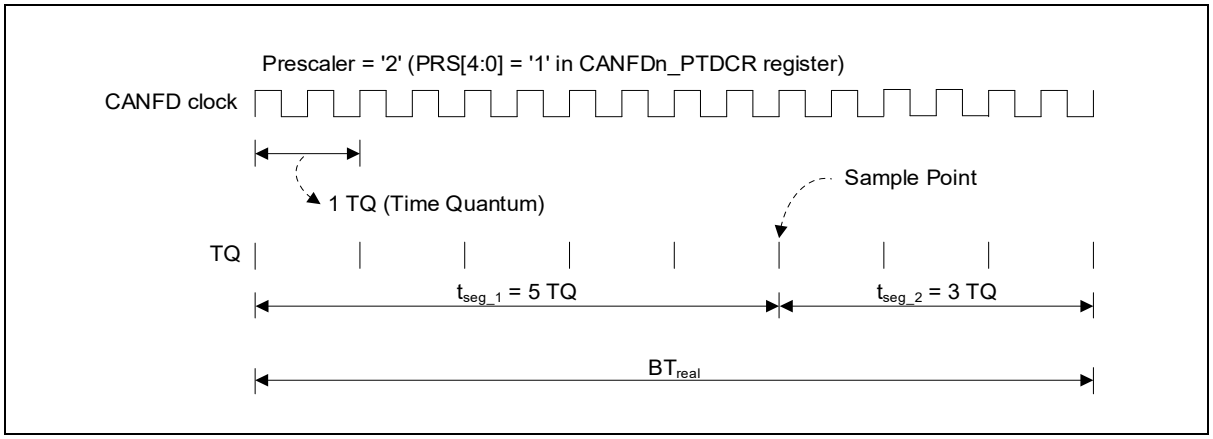
$$n_{prescaler} = (PRS + 1) \times TQ$$

The CANFD module does not verify whether the bit timing configuration is valid or complies with the CAN specification rules. The most important rules are as follows:

- Minimum value of the two SJW parameters must be less than or equal to phase_seg1 and phase_seg2, and it is recommended to set it equal to phase_seg2.
- The bit rate for CAN CC and CAN FD must be the same across all nodes in the CAN network. For CAN FD, the sample points for both the nominal bit rate and the data bit rate must also be identical across all nodes in the CAN network.
- The bit-sampling point should be within the range of 60% to 80%, with 75% being the recommended setting. If possible, it is advisable to set the prescaler to a lower value to maximize the number of TQ per bit.

The CAN bit is recommended to be composed of at least 8 TQs. For high bit rates, it is preferable to configure the CAN bit with at least 10 TQs. This recommendation is based on the resynchronization concept defined in the CAN specification, where resynchronization adjusts the bit length in 1 TQ increments to align the receiver with the received data stream. If a bit is composed of only 8 TQs, an adjustment of 1 TQ would result in a significant change of 1/8 (12.5%).

Figure 168. CAN Bit Timing



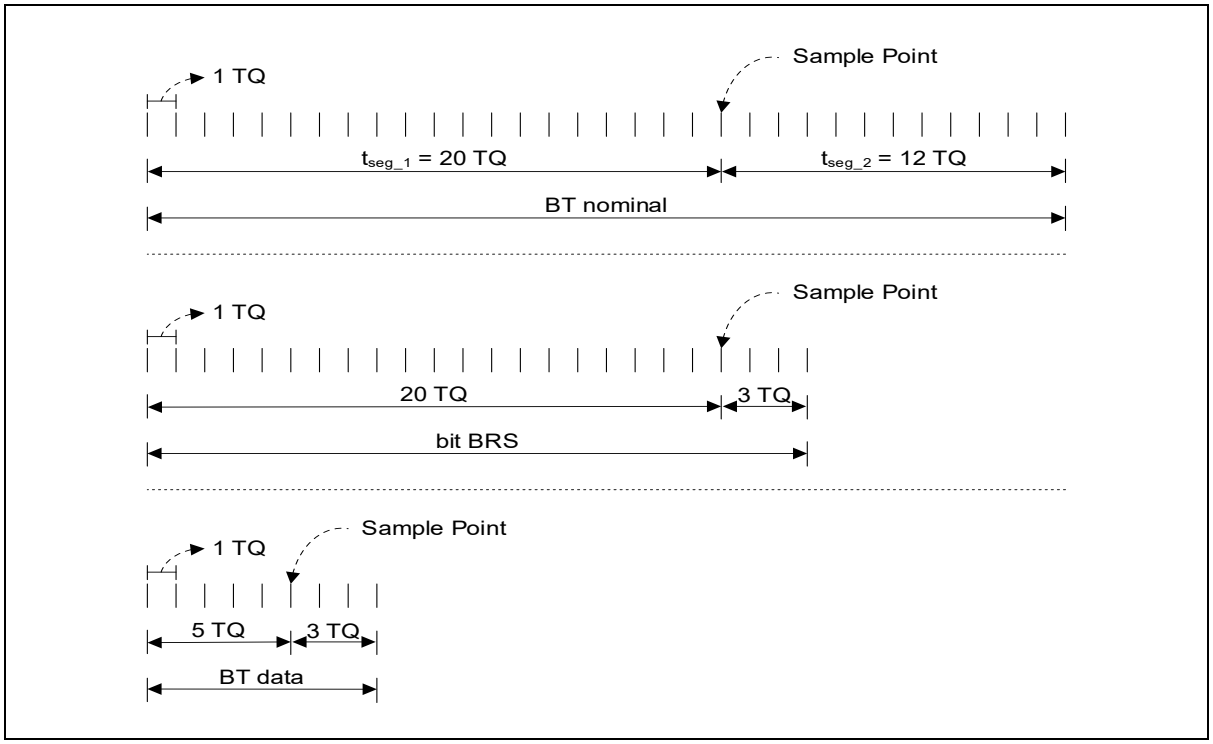
NOTE:

1. The clock for CANFDn is recommended to be 20 MHz, 40 MHz, or 80 MHz.

16.3.11.3 Flexible Data Bit Rate Switching and Sample Point

When using a CAN FD frame with bit rate switching (BRS bit set to '1'), the position of the sample point is critical and must be identical across all nodes in the CAN network. Bit rate switching occurs after the BRS bit and after the CRC delimiter. As shown in the figure below, the position of the sample point significantly affects the absolute length of the BRS bit. If the data bit rate is much higher than the nominal bit rate, the sample point occurs earlier, whereas if it is lower, the sample point is delayed, potentially leading to incorrect sampling.

Figure 169. CAN FD Bit Rate Switching at Bit BRS



When performing bit rate switching, the position of the sample point is very important. (This holds for both the sample points of the nominal bit rate and the FD data bit rate.) It is recommended that Segment1 and Segment2 be configured with the same length across all nodes, and that the length of a single Time Quantum (TQ) be uniform.

16.3.11.4 TDC and RDC

For CAN FD node transmitter delay compensation (TDC) can be optionally enabled while receiver delay compensation (RDC) is automatically active.

TDC and RDC have the following background:

For communication with CAN FD data bit rate, it may be that the delay of the transmitting transceiver or the delay of the bus is bigger than a bit time. This needs to be compensated. Without TDC, the bit rate in the data-phase of FD frames is limited by the fact that the transmitter detects a bit error if it cannot receive its own transmitted bit latest at the sample point of that bit.

Figure 170. Transmitter Delay Compensation (TDC)

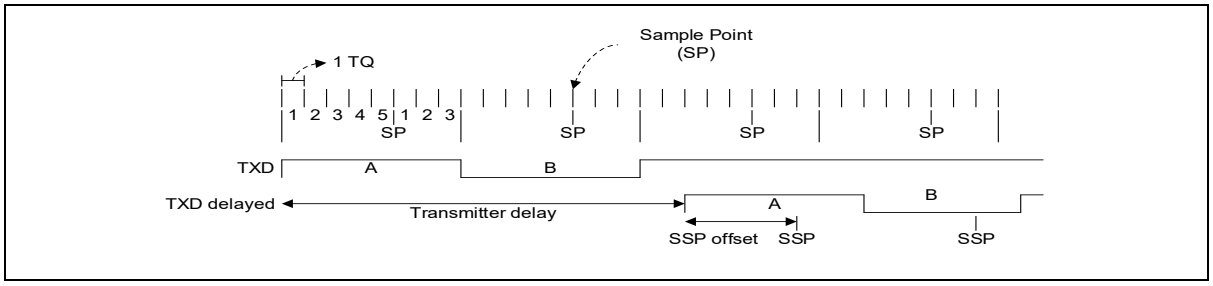


Figure 170 gives an example of the effect of a big transmitter delay. In this figure a TXD data stream is shown starting with bits A and B. One bit time consists of t_{seg1} = 5 TQ before the Sample Point (SP) and t_{seg2} = 3 TQ behind. In this example the transmitter delay is greater than 2-bit times. Therefore, the original SP(Sample Point) cannot be used to sample the correct bit value, and the CAN FD specification defines an additional Secondary Sample Point (SSP). If TDC is enabled (when FD_SSPOFF[7:0] bits in CANFDn_PTDCR register isn't '0') then CANFD module automatically determines the transmitter delay. The position of the SSP is the measured transmitter delay plus the SSP offset which is defined by the configuration FD_SSPOFF[7:0] bits in CANFDn_PTDCR register. FD_SSPOFF[7:0] must be given as a number of TQ and it is suggested to set the SSP offset equal to t_{seg1} + 1.

NOTE:

1. It is recommended to set it to t_{seg1} + 1 and not to t_{seg1}, because of the quantization error when sampling some data.

Automatic measurement of the transmitter delay is done for each frame at the following events:

- CAN FD: at the falling edge between bits FDF and res.
- CAN FD use configurations for SSPOFF: FD_SSPOFF.

ISO 11898-1:2015 requires using only PRS = ‘0’ or ‘1’ if TDC is used. With this requirement CANFD module is capable of automatically determining a transmitter delay of up to 3-bit times of the fast data bit rate. A delay bigger than a bit time needs to be considered also for receiving nodes. CAN FD node performs additional hard synchronizations at the following events:

- CAN FD: at the falling edge between bit FDF = ‘1’ and the following res bit

Synchronization for CAN and CAN FD is in general defined to operate in time steps of one TQ. But steps of one TQ at nominal bit rate may be too coarse. Therefore, the additional hard synchronizations need to violate this rule and need to synchronize as exact as possible and only limited by the frequency of CANFD_CLK. CANFD module uses this special synchronization and this is called RDC. RDC is automatically done during reception, no matter if TDC is enabled or not. RDC is only in effect if $n_{prescaler} > 1$.

16.3.12 Extended Features

16.3.12.1 Re-Transmission and Re-Arbitration Limitation

When transmission fails, **re-transmission** or **re-arbitration** can be used if the CAN node's priority is overridden. Since repeated attempts may cause delays and increase the load on the CAN node, proper usage is required. Re-transmission and re-arbitration can be independently controlled and limited.

Re-transmission occurs after an error, while re-arbitration takes place when arbitration is lost. Refer to section 16.3.8, where re-transmission stops once the bus-off state is reached. The number of re-transmission and re-arbitration attempts can be set up to a maximum of 7 times using the RETLIM[2:0] and REALIM[2:0] bits in the CANFDn_PTDCR register. These settings apply to both PTB and STB.

If the limitation is set and transmission fails, the following phenomena may be observed:

- Even when the retry limitation is set and the transmission fails, if the TSIE bit or TPIE bit in the CANFDn_IER register is set to ‘1’, the TSIF bit or TPIF bit in the CANFDn_ISR register will also be set to ‘1’. In STB transmission, the process does not proceed past the failed data.
- In STB mode, which can contain two or more frames, if the TSALL bit in the CANFDn_CSR register is set to ‘1’, the retry limitation is applied to each frame. If transmission fails during the retry limitation for a frame, the process moves to the next STB slot. If the last frame of the STB reaches its retry limitation and becomes empty, the transmission is terminated.
- When a transmission error occurs, the error type is stored in the KOER[2:0] bits of the CANFDn_ERR register. Additionally, if the BEIE bit in the CANFDn_IER register is set to ‘1’, the BEIF bit in the CANFDn_ISR register will be set to ‘1’.
- In the case of an arbitration error, if the ALIE bit in the CANFDn_IER register is set to ‘1’, the ALIF bit in the CANFDn_ISR register will also be set to ‘1’.

It is not possible to ensure that a frame has been successfully transmitted solely based on the TPIF bit or TSIF bit in the CANFDn_ISR register in situations where re-transmission or re-arbitration occurs with a configured retry limitation.

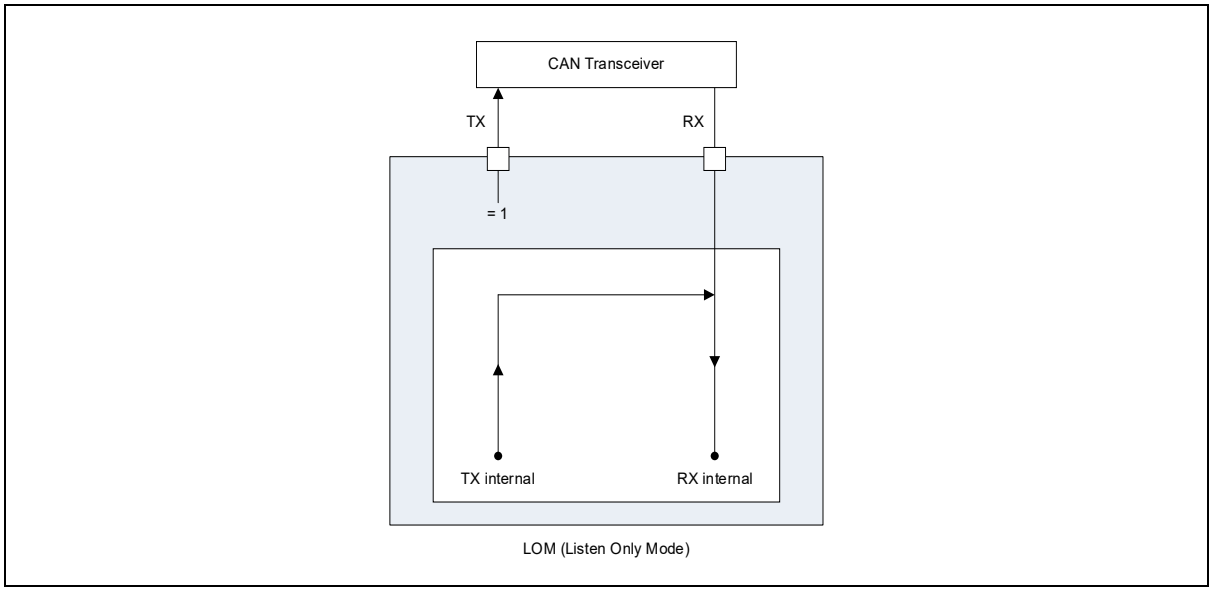
Therefore, when the RETLIM[2:0] bits in the CANFDn_PTDCR register are set, the BEIF bit in the CANFDn_ISR register should also be checked. Similarly, when the REALIM[2:0] bits in the CANFDn_PTDCR register are set, the ALIF bit in the CANFDn_ISR register should also be checked. Additionally, using the CANFDn_TSR register can be beneficial for tracking events that are not included in the CAN frame.

16.3.12.2 Listen Only Mode (LOM)

The Listen-Only Mode (LOM) is a feature that allows monitoring of the CAN bus without affecting its operation. LOM is similar to the bus monitoring function defined in ISO 11898-1:2015 but is not fully compatible. The key difference lies in the acknowledge (ACK) mechanism; in LOM, ACK is not generated and relies on the dominant ACK from other CAN nodes. LOM is also similar to the Restricted Operation mode defined in ISO 11898-1:2015 but is not identical.

When the LOM bit in the CANFDn_CSR register is set to '1', it can be used for automatic bit rate detection. The host controller attempts various bit rate settings until no errors occur. However, automatic bit rate detection is not applicable in a CAN FD network with bit rate switching. It can only be used in a CAN network (CAN CC) operating at a fixed nominal bit rate. Additionally, LOM can monitor errors in the CAN network (KOER[2:0] bits in the CANFDn_RBFR register, BEIF bit in the CANFDn_ISR register), but it cannot utilize dominant bits (no active error flag, overload flag, or acknowledgment).

Figure 171. Listen-Only Mode Block Diagram



Important facts regarding ACKs for LOM:

- If a frame is transmitted by a node, then an ACK visible at the bus will only be generated if at least one additional node is attached to the bus and permitted to generate the ACK. Then there will be no error, and all nodes (also those in LOM) will receive the frame. Otherwise, an ACK error invalidates the frame.
- If there is an ACK error, then the node in LOM can detect this.

Activation of LOM bit in CANFDn_CSR register cannot be done while a transmission is active. If LOM bit in CANFDn_CSR register is enabled, then no transmission
can be started.

The Loop Back Mode External (LBME) plays an important role in the operation of Listen Only Mode (LOM). (Refer to section 16.3.12.3.)

When the LBME bit in the CANFDn_CSR register is disabled ('0'), LOM operates as described above and cannot generate a dominant bit.

When the LBME bit in the CANFDn_CSR register is enabled ('1'), it allows the transmission of frames, including Self ACK (SACK bit in the CANFDn_CSR register). The LBME bit in the CANFDn_CSR register only permits the transmission of frames containing the SACK bit in the CANFDn_CSR register, does not respond with an ACK for frames from other CAN nodes, and does not generate error or overload frames.

16.3.12.3 Loop Back Mode (LBMI, LBME)

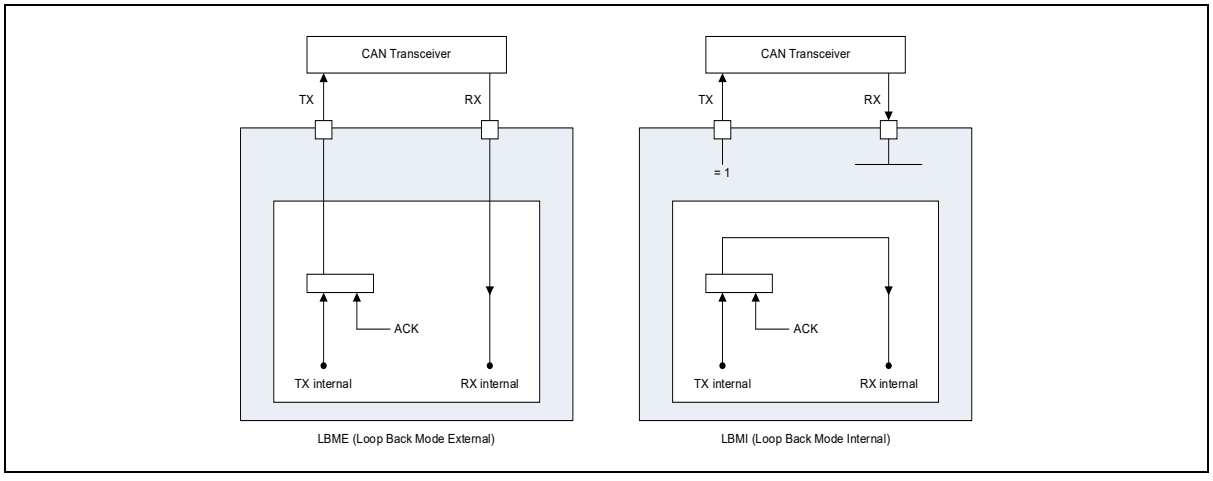
The CANFD module provides two types of loopback modes: Loop Back Mode Internal (LBMI) and Loop Back Mode External (LBME). Both modes receive self-transmitted frames, which are stored in the Receive Buffer (RB) if they pass the acceptance filter. The loopback mode is provided for self-testing of the CAN module. Refer to Figure 172.

When the LBMI bit in the CANFDn_CSR register is set to '1', the connection to the CAN bus is disconnected, and TX and RX are internally connected. The TX state becomes recessive, and the transmitted data is internally received as feedback. In LBMI mode, self-ACK is generated even if the SACK bit in the CANFDn_CSR register is not set to '1' to prevent ACK errors.

When the LBME bit in the CANFDn_CSR register is set to '1', frames transmitted while the CAN transceiver remains connected can be received. The LBME bit in the CANFDn_CSR register does not generate self-ACK. To generate self-ACK, the SACK bit in the CANFDn_CSR register must be set to '1'. If the SACK bit in the CANFDn_CSR register is set to '1', normal communication is possible. However, if the SACK bit is set to '0', two possible outcomes can occur.

- Another node receives the frame too and generates an ACK. This will result in successful transmission and reception.
- No other node is connected to the bus, and this results in an ACK error. To avoid re-transmissions and incrementing the error counters it is recommended to set RETLIM[2:0] bits in CANFDn_PTDCR register '0' if it is unknown if other nodes are connected. The value is reflected in the RB register but is not actually stored, and the RSTAT[1:0] bits in the CANFDn_CSR register do not increase. The subsequently received frame with an ACK will overwrite it. To store invalid data that has not received an ACK, set the RBALL bit in the CANFDn_CSR register to '1', which will store the data in the RB register. The error status of the received frame can be checked using the KOER[2:0] bits in the CANFDn_RBFR register.

Figure 172. Loop Back Mode Block Diagram



If the node is connected to a CAN bus, then switching back from LBMI to normal operation must not be done by simply setting LBMI bit in CANFDn_CSR register to ‘0’, because then it may occur just at the moment while another CAN node is transmitting. In this case switching back to normal operation shall be done by setting bit RESET bit in CANFDn_CSR register to ‘1’. This automatically clears LBMI bit in CANFDn_CSR register to ‘0’. Finally RESET can be disabled and CANFD module returns back to normal operation. In contrast to this LBME bit in CANFDn_CSR register can be disabled every time.

NOTE:

1. The LBMI bit and LBME bit in the CANFDn_CSR register must not be changed together with the TPE bit, TSONE bit, or TSALL bit in the CANFDn_CSR register.

16.3.12.4 Transceiver Standby Mode

The STBY bit in the CANFDn_CSR register is used to activate the standby mode of the CAN transceiver. This function determines the standby mode of the transceiver, not the CAN controller, so the transceiver specifications should be checked beforehand.

When the STBY bit in the CANFDn_CSR register is set, the transceiver enters standby mode, making transmission impossible. Therefore, the TPE bit, TSONE bit, and TSALL bit in the CANFDn_CSR register cannot be set. Conversely, if transmission is already in progress, the STBY bit in the CANFDn_CSR register cannot be set.

If a dominant signal is received on RX while the STBY bit in the CANFDn_CSR register is set to '1', the CANFD module automatically sets the STBY bit in the CANFDn_CSR register to '0' and deactivates the transceiver's standby mode. During the transition from standby mode to active mode, the initial wake-up frame cannot be successfully received, and an ACK error may occur.

NOTE:

1. The specifications of each transceiver vary, so refer to the corresponding transceiver's datasheet.

16.3.12.5 Error Counter Reset

According to the CAN standard RECNT[7:0] bits in CANFDn_ERR register counts receive errors and TECNT[7:0] bits in CANFDn_ERR register counts transmit errors. After too many transmit errors a CAN node has to go to bus off state (refer to 16.3.8). RESET bit in CANFDn_CSR register does not modify the errors counters or the bus off state. The CAN specification defines rules on how to exit from bus off state and to decrease the error counters. A good node will recover from all of this automatically if only a temporary error has caused the problems. The classic CAN 2.0B specification requires this automatic behavior without host controller interaction to avoid the babbling idiot problem.

The CAN FD specification relaxes this and allows the host controller to override the automatic error counter handling. But this should be used with extra care, and it is suggested to be used only for debugging purposes.

Writing a ‘1’ to the BUSOFF bit in CANFDn_CSR register resets the error counters and therefore forces the node to leave the bus off state. This is done without setting EIF bit in CANFDn_ISR register.

16.3.13 Software Reset

When the RESET bit in the CANFDn_CSR register is set to ‘1,’ the CANFD module enters the reset state. The RESET bit is one of the key factors in configuring the entire CANFDn registers. Some registers can only be modified when the CAN module is in the reset state.

When the RESET bit is changed to ‘0,’ the transition to the non-reset state does not take effect immediately. Instead, CAN communication becomes available after 11 CAN bit times. This delay is bus-idle time and required by the CAN standard.

Table 158 describes the behavior of CANFDn registers according to the RESET bit state.

Table 158. Correlation with CAN Software Reset

Register Name	Bit Name	Reset [1)]	Remark
CANFDn_AFCR	ACFADR	No	-
CANFDn_AFCIR CANFDn_AFCFR CANFDn_AFCTR	ACFC	No	RESET bit '1' is writeable, '0' is not writeable
CANFDn_AFMIR CANFDn_AFMFR CANFDn_AFMTR	ACFM	No	RESET bit '1' is writeable, '0' is not writeable
CANFDn_ACBTR	AC_SEG1	No	RESET bit '1' is writeable, '0' is not writeable
CANFDn_ACBTR	AC_SEG2	No	RESET bit '1' is writeable, '0' is not writeable
CANFDn_ACBTR	AC_SJW	No	RESET bit '1' is writeable, '0' is not writeable
CANFDn_AFCR	AE_x (x:0 to 15)	No	-
CANFDn_ERR	AFWL	No	-
CANFDn_ISR	AIF	Yes	-
CANFDn_ERR	ALC	Yes	-
CANFDn_IER	ALIE	No	-
CANFDn_ISR	ALIF	Yes	-
CANFDn_IER	BEIE	No	-
CANFDn_ISR	BEIF	Yes	-
CANFDn_CSR	BUSOFF	(No)	An error counter reset by setting BUSOFF bit ‘1’ also resets BUSOFF bit.
CANFDn_VNCFGR	CES	No	-
CANFDn_IER	EIE	No	-
CANFDn_ISR	EIF	No	-
CANFDn_ISR	EPASS	No	-
CANFDn_IER	EPIE	No	-
CANFDn_ISR	EPIF	Yes	-
CANFDn_ISR	EWARN	No	-

Table 158. Correlation with CAN Software Reset (continued)

Register Name	Bit Name	Reset [1)]	Remark
CANFDn_ERR	EWL [2)]	Yes	RESET bit '0' is writeable, '1' is not writeable
CANFDn_CSR	FD_ISO	No	RESET bit '1' is writeable, '0' is not writeable
CANFDn_FDBTR	FD_SEG1	No	RESET bit '1' is writeable, '0' is not writeable
CANFDn_FDBTR	FD_SEG2	No	RESET bit '1' is writeable, '0' is not writeable
CANFDn_FDBTR	FD_SJW	No	RESET bit '1' is writeable, '0' is not writeable
CANFDn_PTDCR	FD_SSPOFF	No	RESET bit '1' is writeable, '0' is not writeable
CANFDn_ERR	KOER	Yes	-
CANFDn_CSR	LBME	Yes	-
CANFDn_CSR	LBMI	Yes	-
CANFDn_LFCIR	LLCFORMAT	No	-
CANFDn_LFCIR	LLCPBYTES	No	-
CANFDn_CSR	LOM	No	-
CANFDn_PTDCR	PRS	No	RESET bit '1' is writeable, '0' is not writeable
CANFDn_IER	RAFIE	No	-
CANFDn_ISR	RAFIF	Yes	-
CANFDn_CSR	RBALL	Yes	-
CANFDn_RBIR CANFDn_RBFR CANFDn_RBTR	RB (RBUF)	(Yes)	All RB slots are marked as empty. RB contains unknown data.
CANFDn_TSCFGR	RBEN	No	RESET bit '1' is writeable, '0' is not writeable
CANFDn_PTDCR	REALIM [2)]	Yes	RESET bit '0' is writeable, '1' is not writeable
CANFDn_ERR	RECNT	No	An error counter reset is possible by setting BUSOFF bit '1'.
CANFDn_PTDCR	RETLIM [2)]	Yes	RESET bit '0' is writeable, '1' is not writeable
CANFDn_IER	RFIE	No	-
CANFDn_ISR	RFIF	Yes	-
CANFDn_IER	RIE	No	-
CANFDn_ISR	RIF	Yes	-
CANFDn_IER	ROIE	No	-
CANFDn_ISR	ROIF	Yes	-
CANFDn_CSR	ROM	No	-
CANFDn_VNCFGR	ROP	No	-
CANFDn_CSR	ROV	Yes	All RB slots are marked as empty.
CANFDn_CSR	RREL	Yes	-
CANFDn_CSR	RSTAT	Yes	All RB slots are marked as empty.
CANFDn_CSR	SACK	Yes	-
CANFDn_CSR	STBY	No	-
CANFDn_CSR	TBSEL	Yes	TB fixed to point to PTB.

Table 158. Correlation with CAN Software Reset (continued)

Register Name	Bit Name	Reset [1)]	Remark
CANFDn_TBIR CANFDn_TBFR CANFDn_TBTR	TB (TBUF)	(Yes)	All STB slots are marked as empty. Because TBSEL TB points to the PTB.
CANFDn_ERR	TECNT	No	An error counter reset is possible by setting BUSOFF=1.
CANFDn_CSR	TPA	Yes	-
CANFDn_CSR	TPE	Yes	-
CANFDn_CSR	TSA	Yes	-
CANFDn_CSR	TSALL	Yes	-
CANFDn_CSR	TSMODE	No	RESET bit '1' is writeable, '0' is not writeable
CANFDn_CSR	TSNEXT	Yes	-
CANFDn_CSR	TSONE	Yes	-
CANFDn_IER	TPIE	No	-
CANFDn_ISR	TPIF	Yes	-
CANFDn_CSR	TSFF	Yes	All STB slots are marked as empty.
CANFDn_IER	TSIE	No	-
CANFDn_ISR	TSIF	Yes	-
CANFDn_CSR	TSSTAT	Yes	All STB slots are marked as empty.
CANFDn_TSR	HANDLE1	No	-
CANFDn_TSR	TSTAT1	Yes	-
CANFDn_TSR	HANDLE2	Yes	-
CANFDn_TSR	TSTAT2	Yes	-

NOTES:

1. When the CANFD module undergoes a software reset, changes in the register can be observed. (Yes: Reset, No: No change)
2. The EWL[3:0] bits in the CANFDn_ERR register and the RETLIM[2:0] and REALIM[2:0] bits in the CANFDn_PTDCR register cannot be written when the RESET bit in the CANFDn_CSR register is set to ‘1’. Once the RESET bit is changed to ‘0’, writing becomes possible; however, the written values are applied to the register only after a certain period of time. This period is approximately {(4 + Max 2) CANFD_CLK} + {(5 + Max 2) PCLK} clocks after the RESET bit is changed to ‘0’. Due to this delay, even if the EWL[3:0], RETLIM[2:0], and REALIM[2:0] bits are written immediately after the RESET bit changes to ‘0’, the values may not be immediately reflected in the register. Therefore, caution is required, and it is recommended to configure other registers first before setting these bits after the RESET bit is changed to ‘0’.

16.4 CANFD Interrupts

During CAN FD communications, various interrupts may be generated by different events. These interrupts can be checked by reading the CANFDn_ISR register. To verify them by reading the CANFDn_ISR register, the corresponding bit in the CANFDn_IER register must be enabled.

Table 159. Interrupt on CANFDn

Interrupt event	Event flag	Enable control bit	Interrupt clear method
Error warning limit reached [2)]	EWARN	-	-
Error Passive status [3)]	EPASS	-	-
Error passive interrupt flag	EPIF	EPIE	Write 1 in EPIF
Arbitration lost	ALIF	ALIE	Write 1 in ALIF
Bus error	BEIF	BEIE	Write 1 in BEIF
Receive Interrupt	RIF	RIE	Write 1 in RIF
Receive Buffer Overflow	ROIF	ROIE	Write 1 in ROIF
Receive Buffer Full	RFIF	RFIF	Write 1 in RFIF
Receive Buffer Almost Full	RAFIF	RAFIE	Write 1 in RAFIF
Transmission PTB completed	TPIF	TPIE	Write 1 in TPIF
Transmission STB completed	TSIF	TSIE	Write 1 in TSIF
Error interrupt flag	EIF	EIE	Write 1 in EIF
Abort Error	AIF	-	Write 1 in AIF

NOTES:

1. n = 0 and 1.
2. The EWARN bit in the CANFDn_ISR register indicates the current status by comparing it with the Error Warning Limit (EWL). Refer to section 16.3.9.1.
3. The EPASS bit in the CANFDn_ISR register represents the Error Passive state. Refer to Table 153.

16.5 CANFD Registers

The base address and register map of the UART are described in the following tables:

Table 160. Base Address of CAN FD

Name	Base Address
CANFD0	0x4001_0000
CANFD1	0x4001_2000

Table 161. CANFD Register Map

Name	Offset	Type	Description	Reset Value	Reference
CANFDn_VNCFGR	0x0000	RO	CANFD n Version and Node Configuration Register	0x0001_0809	16.5.1
CANFDn_ACBTR	0x0004	RW	CANFD n Arbitration and Classic Bit Timing Register	0x0505_0008	16.5.2
CANFDn_FDBTR	0x0008	RW	CANFD n Flexible Data Bit Timing Register	0x0202_0003	16.5.3
CANFDn_PTDCR	0x0010	RW	CANFD n Prescaler and Transmitter Delay Compensation Register	0x7700_0000	16.5.4
CANFDn_ISR	0x0014	RW	CANFD n Interrupt Status Register	0x0000_0000	16.5.5
CANFDn_IER	0x0018	RO	CANFD n Interrupt Enable Register	0x0000_00FE	16.5.6
CANFDn_TSR	0x001C	RW	CANFD n Transmit Status Register	0x0000_0000	16.5.7
CANFDn_CSR	0x0028	RW	CANFD n Control and Status Register	0x0080_0080	16.5.8
CANFDn_ERR	0x002C	RW	CANFD n Error Register	0x0000_001B	16.5.9
CANFDn_AFCR	0x0044	RW	CANFD n Acceptance Filter Control Register	0x0001_0000	16.5.10
CANFDn_AFCIR	0x0048	RW	CANFD n Acceptance Filter Code Identifier Register	0x0000_0000	16.5.11
CANFDn_AFCFR	0x004C	RW	CANFD n Acceptance Filter Code Format Register	0x0000_0000	16.5.12
CANFDn_AFCTR	0x0050	RW	CANFD n Acceptance Filter Code Type Register	0x0000_0000	16.5.13
CANFDn_AFMIR	0x0058	RW	CANFD n Acceptance Filter Mask Identifier Register	0x1FFF_FFFF	16.5.14
CANFDn_AFMFR	0x005C	RW	CANFD n Acceptance Filter Mask Format Register	0x1F17_000F	16.5.15
CANFDn_AFMTR	0x0060	RW	CANFD n Acceptance Filter Mask Type Register	0xFF00_0000	16.5.16

Table 161. CANFD Register Map (continued)

Name	Offset	Type	Description	Reset Value	Reference
CANFDn_RBIR	0x0070	RW	CANFD n Receive Buffer Identifier Register	0xXXXX_XXXX	16.5.17
CANFDn_RBFR	0x0074	RW	CANFD n Receive Buffer Format Register	0xXXXX_XXXX	16.5.18
CANFDn_RBTR	0x0078	RW	CANFD n Receive Buffer Type Register	0xXXXX_XXXX	16.5.19
CANFDn_RBPDR	0x0080	RW	CANFD n Receive Buffer Payload Data Register	0xXXXX_XXXX	16.5.20
CANFDn_TBIR	0x0890	RW	CANFD n Transmit Buffer Identifier Register	0xXXXX_XXXX	16.5.21
CANFDn_TBFR	0x0894	RW	CANFD n Transmit Buffer Format Register	0xXXXX_XXXX	16.5.22
CANFDn_TBTR	0x0898	RW	CANFD n Transmit Buffer Type Register	0xXXXX_XXXX	16.5.23
CANFDn_TBPDR	0x08A0	RW	CANFD n Transmit Buffer Payload Data Register	0xXXXX_XXXX	16.5.24
CANFDn_TSCFGR	0x10B0	RW	CANFD n Transmit Stream Configuration Register	0x0000_0001	16.5.25
CANFDn_LFCIR	0x10B4	RW	CANFD n LLC Frame Calculator Input Register	0x0000_0000	16.5.26
CANFDn_LFCOR	0x10B8	RO	CANFD n LLC Frame Calculator Output Register	0x0000_0000	16.5.27

NOTE:

1. n = 0 and 1.

16.5.1 CANFDn_VNCFGR: CANFD n Version and Node Configuration Register

The CANFDn_VNCFGR register is a control register for the node configuration. This register is 32 bits wide.

CANFD0_VNCFGR=0x4001_0000, CANFD1_VNCFGR=0x4001_2000

31 30 29 28 27 26 25 24	23 22 21 20 19 18	17	16	15 14 13 12 11 10 9 8	7 6 5 4 3 2 1 0
Reserved		ROP	CES	VERSION[7:0]	
-		0	1	0x0809	
-		RW	RO	RO	

17	ROP	Restricted Operation
		0 Disables restricted operation
		1 Enables restricted operation
16	CES	Always ‘1’
15 0	VERSION[7:0]	Version of CANFD

16.5.2 CANFDn_ACBTR: CANFD n Arbitration and Classic Bit Timing Register

The CANFDn_ACBTR register is a control register for nominal bit time of the arbitration phase. This register is 32 bits wide.

CANFD0_ ACBTR=0x4001_0004, CANFD1_ ACBTR=0x4001_2004

31 30 29 28 27 26 25 24	23 22 21 20 19 18 17 16	15 14 13 12 11 10 9 8	7 6 5 4 3 2 1 0
Reserved	AC_SJW[6:0]	Reserved	AC_SEG2[6:0]	Reserved	AC_SEG1[8:0]	
-	0x05	-	0x05	-	0x08	
-	RW	-	RW	-	RW	

30 24	AC_SJW[6:0](1)	Synchronization Jump Width
22 16	AC_SEG2[6:0](1)	Bit Timing Segment 2
8 0	AC_SEG1[8:0](1)	Bit Timing Segment 1

NOTE:

1. The RESET bit in the CANFDn_CSR register must be set to ‘1’ to allow writing. Refer to 16.3.13.

16.5.3 CANFDn_FDBTR: CANFD n Flexible Data Bit Timing Register

The CANFDn_FDBTR register is a control register for data bit time of the data phase. This register is 32 bits wide.

CANFD0_ FDBTR=0x4001_0008, CANFD1_ FDBTR=0x4001_2008

31 30 29 28 27 26 25 24		23 22 21 20 19 18 17 16		15 14 13 12 11 10 9 8	7 6 5 4 3 2 1 0
Reserved	FD_SJW[6:0]	Reserved	FD_SEG2[6:0]	Reserved	FD_SEG1[7:0]
-	0x02	-	0x02	-	0x03
-	RW	-	RW	-	RW

30 24	FD_SJW[6:0](1)	Synchronization Jump Width
22 16	FD_SEG2[6:0](1)	Bit Timing Segment 2
7 0	FD_SEG1[7:0](1)	Bit Timing Segment 1

NOTES:

1. The RESET bit in the CANFDn_CSR register must be '1' to allow writing. Refer to section 16.3.13.
2. The settings are applied when transmitting in CAN FD mode.

16.5.4 CANFDn_PTDCR: CANFD n Prescaler and Transmitter Delay Compensation Register

The CANFDn_PTDCR register is a control register for the CANFD prescaler and transmitter delay compensation. This register is 32 bits wide.

CANFD0_PTDCR=0x4001_0010, CANFD1_PTDCR=0x4001_2010

31	30 29 28	27	26 25 24	23 22 21 20 19 18 17 16	15 14 13 12 11 10 9 8	7 6 5	4 3 2 1 0
Reserved	RETLIM[2:0]	Reserved	REALIM[2:0]	Reserved	FD_SSPOFF[7:0]	Reserved	PRS[4:0]
-	0x7	-	0x7	-	0x00	-	0x00
-	RW	-	RW	-	RW	-	RW

30 28	RETLIM[2:0](2)	Re-transmission Limit If there is any error during transmission, a CAN node is able to automatically retry it as soon as the bus is idle. The number of re-transmission attempts can be limited using RETLIM[2:0] bits in CANFDn_PTDCR register. RETLIM[2:0] may be updated at any time, but after an update the content of the register needs to be transferred to the CAN protocol machine using clock domain crossing. During this short amount of time (a few clocks) RETLIM[2:0] is write-locked.
		000 1 attempts (no re-transmission)
		001 2 attempts
		010 3 attempts
		011 4 attempts
		100 5 attempts
		101 6 attempts
		110 7 attempts
		111 Unlimited (Only limited by the transmit error counter TECNT[7:0] bits in CANFDn_ERR register
26 24	REALIM[2:0](2)	Re-arbitration Limit If two or more CAN nodes try to transmit frames simultaneously, then the lower-priority frames lose arbitration and silently step back without interrupting the higher-priority frame. A CAN node is able to automatically retry it as soon as the bus is idle. The number of re-arbitration attempts can be limited using REALIM[2:0] bits in CANFDn_PTDCR register. REALIM[2:0] may be updated at any time, but after an update the content of the register needs to be transferred to the CAN protocol machine using clock domain crossing. During this short amount of time (a few clocks) REALIM[2:0] is write-locked.
		000 1 attempt (no re-arbitration)
		001 2 attempts
		010 3 attempts
		011 4 attempts
		100 5 attempts
		101 6 attempts
		110 7 attempts
		111 Unlimited
15 8	FD_SSPOFF [7:0](1)	Secondary Sample Point Offset For CAN FD frames FD_SSPOFF defines SSPOFF. Transmitter Delay Compensation (TDC) is enabled if FD_SSPOFF ≠ '0' and CES = '1'. TDC will be activated during the data phase of a CAN FD frame.

		The transmitter delay plus FD_SSPOFF defines the time of the secondary sample point for TDC. FD_SSPOFF is given as a number of TQ.
4 0	PRS[4:0][1]	Prescaler The prescaler divides the system clock to synthesize the Time Quanta (TQ). The prescaler is $n_{prescaler}$ = PRS[4:0] + 1. The duration of a time-quanta (TQ) is $t_{TQ} = n_{prescaler}$ / CANFD_CLK

NOTES:

1. The RESET bit in the CANFDn_CSR register must be '1' to allow writing. Refer to section 16.3.13.
2. The RESET bit in the CANFDn_CSR register must be '0' to allow writing. Refer to section 16.3.13.

16.5.5 CANFDn_ISR: CANFD n Interrupt Status Register

The CANFDn_ISR register is a status register for the CANFD interrupt flag. This register is 32 bits wide.

CANFD0_ISR=0x4001_0014, CANFD1_ISR=0x4001_2014

31	30	29 28 27 26 25 24	23 22 21 20 19 18 17 16	15 14 13 12 11	10	9	8	7	6	5	4	3	2	1	0
EWARN	EPASS	Reserved			EPIF	ALIF	BEIF	RIF	ROIF	RFIF	RAFIF	TPIF	TSIF	EIF	AIF
0	0	-			0	0	0	0	0	0	0	0	0	0	0
RO	RO	-			RWC1	RWC1	RWC1	RWC1	RWC1	RWC1	RWC1	RWC1	RWC1	RWC1	RWC1

31	EWARN	Error warning limit reached
	0	The values in both counters are less than EWL[3:0] bits in CANFDn_ERR register.
	1	One of the error counters RECNT[7:0] bits or TECNT[7:0] bits in CANFDn_ERR register is equal or bigger than EWL[3:0] bits in CANFDn_ERR register.
30	EPASS	Error Passive status
	0	Node is error active
	1	Node is error passive
10	EPIF	Error passive interrupt flag EPIF will be activated if EPASS changes and if EPIE bit in CANFDn_IER register '1'.
	0	The error passive interrupt event has not occurred
	1	The error passive interrupt event has occurred
9	ALIF	Arbitration lost interrupt flag
	0	The arbitration lost interrupt event has not occurred
	1	The arbitration lost interrupt event has occurred
8	BEIF	Bus error interrupt flag
	0	The bus error interrupt event has not occurred
	1	The bus error interrupt event has occurred
7	RIF	Receive interrupt flag
	0	No frame has been received
	1	Data or a remote frame has been received and is available in the RB
6	ROIF	RB overflow interrupt flag In case of an overrun both ROIF and RFIF will be set
	0	No overflow
	1	At least one received frame has been overwritten in the RB or discarded.
5	RFIF	RB full interrupt flag
	0	The RB FIFO is not full
	1	All RBs are full. If no RB will be released until the next valid frame is received, the oldest frame will be lost.
4	RAFIF	RB almost full interrupt flag
	0	Number of filled RB slots < AFWL[3:0] bits in CANFDn_ERR register.
	1	Number of filled RB slots >= AFWL[3:0] bits in CANFDn_ERR register.
3	TPIF	Transmission primary interrupt flag
	0	No transmission of the PTB has been completed.
	1	The requested transmission of the PTB has been successfully completed.
2	TSIF	Transmission secondary interrupt flag

		0	No transmission of the STB has been completed.
		1	The requested transmission of the STB has been successfully completed.
1	EIF	Error interrupt flag	
		0	There has been no change.
		1	The border of the error warning limit has been crossed in either direction, or the BUSOFF bit in CANFDn_CSR register has been changed in either direction.
0	AIF	Abort interrupt flag It is not recommended to set both TPA bit and TSA bit in CANFDn_CSR register simultaneously because both source AIF. The AIF does not have an associated enable register.	
		0	No abort has been executed.
		1	After setting TPA bit or TSA bit in CANFDn_CSR register the appropriated frame has been aborted.

NOTE:

1. Refer to section 16.4 for details on setting the interrupt flag.

16.5.6 CANFDn_IER: CANFD n Interrupt Enable Register

The CANFDn_IER register is a control register for the CANFD interrupt enable. This register is 32 bits wide.

CANFD0_IER=0x4001_0018, CANFD1_IER=0x4001_2018

31 30 29 28 27 26 25 24	23 22 21 20 19 18 17 16	15 14 13 12 11	10	9	8	7	6	5	4	3	2	1	0
Reserved			EPIE	ALIE	BEIE	RIE	ROIE	RFIE	RAFIE	TPIE	TSIE	EIE	Reserved
-			0	0	0	1	1	1	1	1	1	1	-
-			RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	-

10	EPIE	Enable the error passive interrupt	
		0	Disables the error passive interrupt
		1	Enables the error passive interrupt
9	ALIE	Enable the arbitration lost interrupt	
		0	Disables the arbitration lost interrupt
		1	Enables the arbitration lost interrupt
8	BEIE	Enable the bus error interrupt	
		0	Disables the bus error interrupt
		1	Enables the bus error interrupt
7	RIE	Enable the receive interrupt	
		0	Disables the receive interrupt
		1	Enables the receive interrupt
6	ROIE	Enable the RB overflow interrupt	
		0	Disables the RB overflow interrupt
		1	Enables the RB overflow interrupt
5	RFIE	Enable the RB full interrupt	
		0	Disables the RB full interrupt
		1	Enables the RB full interrupt
4	RAFIE	Enable the RB almost full interrupt flag	
		0	Disables the RB almost full interrupt
		1	Enables the RB almost full interrupt
3	TPIE	Enable the transmission primary interrupt	
		0	Disables the transmission primary interrupt
		1	Enables the transmission primary interrupt
2	TSIE	Enable the transmission secondary interrupt	
		0	Disables the transmission secondary interrupt
		1	Enables the transmission secondary interrupt
1	EIE	Enable the error interrupt	
		0	Disables the error interrupt
		1	Enables the error interrupt

16.5.7 CANFDn_TSR: CANFD n Transmit Status Register

The CANFDn_TSR register is a control register for the CANFD transmit status. This register is 32 bits wide.

CANFD0_TSR=0x4001_001C, CANFD1_TSR=0x4001_201C

31 30 29 28 27	26 25 24	23 22 21 20 19 18 17 16	15 14 13 12 11	10 9 8	7 6 5 4 3 2 1 0
Reserved	TSTAT2[2:0]	HANDLE2[7:0]	Reserved	TSTAT1[2:0]	HANDLE1[7:0]
-	000	0x00	-	000	0x00
-	R0	R0	-	R0	R0

26 24	TSTAT2[2:0]	TSTAT2 Transmission status code
23 16	HANDLE2[7:0]	TSTAT2 Handle for frame identification
10 8	TSTAT1[2:0]	TSTAT1 Transmission status code
7 0	HANDLE1[7:0]	TSTAT1 Handle for frame identification

NOTE:

1. Refer to Table 155 and Table 156 in section 16.3.9.5.

16.5.8 CANFDn_CSR: CANFD n Control and Status Register

The CANFDn_CSR register is a control register for the configuration, status, transmission and receive. This register is 32 bits wide.

CANFD0_CSR=0x4001_0028, CANFD1_CSR=0x4001_2028

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
SACK	ROM	ROV	RREL	RBALL	Reserved	RSTAT[1:0]		FD_ISO	TSNEXT	TSMODE	Reserved		TSFF	TSSTAT[1:0]		TBSEL	LOM	STBY	TPE	TPA	TSONE	TSALL	TSA	RESET	LBME	LBMI	Reserved				BUSOFF
0	0	0	0	0	-	00		1	0	0	-		0	00		0	0	0	0	0	0	0	0	1	0	0	-				0
RW	RW	RO	RW	RW	-	RO		RW	RW	RW	-		RO	RO		RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	-				RW

31	SACK	Self Acknowledge
		0 Disables the self ACK.
		1 Enables the self ACK. When using Loop Back Mode External (LBME), it must be set to '1'. If the LBME bit is set to '1', the SACK bit is also set to '1'.
30	ROM	Receive buffer overflow mode In case of a full RB when a new frame is received, then ROM selects the following:
		0 The oldest frame will be overwritten.
		1 The new frame will not be stored.
29	ROV	Receive buffer overflow status
		0 No Overflow.
		1 Overflow. At least one frame is lost. ROV is cleared by setting RREL = '1'.
28	RREL	Receive buffer release. The host controller has read the actual RB slot and releases it. Afterwards CANFD module points to the next RB slot. RSTAT[1:0] gets updated.
		0 No release
		1 Release (The host has read the RB)
27	RBALL(2)	Receive Buffer stores ALL data frames
		0 Normal operation (only valid frames)
		1 RB stores correct data frames as well as data frames with error.
25 24	RSTAT[1:0]	Receive buffer status
		00 Empty
		01 > empty and < almost full (AFWL[3:0] bits in CANFDn_ERR register)
		10 >= almost full (programmable threshold by AFWL[3:0] bits in CANFDn_ERR register) but not full and no overflow
		11 Full (Stays set in case of overflow – for overflow signaling see ROV)
23	FD_ISO(1)	CAN FD ISO mode. ISO CAN FD mode has a different CRC initialization value and an additional stuff bit count. Both modes are incompatible and must not be mixed in one CAN network. This bit has no impact on CAN CC.
		0 Bosch CAN FD (non-ISO) mode
		1 ISO CAN FD mode (ISO 11898-1:2015)
22	TSNEXT	Transmit buffer secondary next After all frame bytes are written to the TB registers, the host controller has to set TSNEXT to signal that this slot has been filled. Then CANFD module connects the TB registers to the next slot.

		Once a slot is marked as filled a transmission can be started using TSONE or TSALL. It is possible to set TSNEXT and TSONE or TSALL together in one write access. TSNEXT has to be set by the host controller and is automatically reset by CANFD module immediately after it is set. Setting TSNEXT is meaningless if TBSEL = '0'. In this case TSNEXT is ignored and automatically cleared. It does not do any harm. If all slots of the STB are filled, TSNEXT stays set until a slot becomes free.
		0 No action
		1 STB slot filled, select next slot.
21	TSMODE(1)	Transmit buffer secondary operation mode. In FIFO mode frames are transmitted in the order in that they are written into the STB. In priority reordering mode the frame with the highest priority in the STB is automatically transmitted first. The ID of a frame is used for priority decision. A lower ID means a higher priority of a frame. A frame in the PTB always has the highest priority regardless of the ID. TSMODE bit shall be switched only if the STB if empty.
		0 FIFO mode
		1 Priority reordering mode
18	TSFF	Transmit secondary buffer full flag The update of TSFF bit after setting TSNEXT = '1' takes one additional cycle of PCLK. Therefore, reading TSFF must not be done immediately after setting TSNEXT = '1'.
		0 The STB is not filled with the maximal number of frames.
		1 The STB is filled with the maximal number of frames.
17 16	TSSTAT[1:0]	Transmit secondary status The update of TSSTAT bit after setting TSNEXT = '1' takes a one additional cycle of PCLK. Therefore, reading TSSTAT bit must not be done immediately after setting TSNEXT = '1'.
		00 STB is empty
		01 STB is less than or equal to half full
		10 STB is more than half full
		11 STB is full
15	TBSEL	Transmit buffer select Selects the transmit buffer to be loaded with a frame. Use the TB registers for access. TBSEL bit needs to be stable all the time the TB registers are written and when TSNEXT is set.
		0 PTB (high-priority buffer)
		1 STB
14	LOM	Listen only mode (Refer to 16.3.12.2) LOM cannot be changed while transmission is active. No transmission can be started if LOM is enabled and LBME is disabled. LOM = '1' and LBME = '0' disables all transmissions. LOM = '1' and LBME = '1' disables the ACK for received frames and error frames but enables the transmission of own frames.
		0 Disables the listen only mode
		1 Enables the listen only mode
13	STBY	Transceiver standby mode (Refer to 16.3.12.4) This register bit is connected to the output signal CANFDn_STBY which can be used to control a standby mode of a transceiver. STBY bit cannot be set to '1' if TPE = '1', TSONE = '1' or TSALL = '1'. If the host sets STBY to '0' then the host needs to wait for the time required by the transceiver to start up before the host requests a new transmission.
		0 Disables the transceiver standby mode.
		1 Enables the transceiver standby mode.
12	TPE	Transmit primary enable

		0	No transmission for the PTB
		1	Transmission enable for the frame in the high-priority PTB
11	TPA	Transmit primary abort The bit has to be set by the host controller and will be reset by CANFD module. Setting TPA automatically de-asserts TPE. The host controller can set TPA to '1' but cannot reset it to '0'. During the short time while CANFD module resets the bit, it cannot be set by the host. The bit will be reset to the hardware reset value if RESET = '1'. TPA should not be set simultaneously with TPE.	
		0	No abort
		1	Aborts a transmission from PTB which has been requested by TPE = '1' but has not started yet. (The data bytes of the frame remain in the PTB.)
10	TSONE	Transmit secondary one frame The host controller can set TSONE to '1' but cannot reset it to '0'. This would only be possible using TSA and aborting the frame. The bit will be reset to the hardware reset value if RESET = '1' or STBY = '1' or (LOM = '1' and LBME = '0').	
		0	No transmission for the STB.
		1	Transmission enable of one in the STB. In FIFO mode this is the oldest frame and in priority mode this is the one with the highest priority. TSONE in priority mode is difficult to handle because it is not always clear which frame will be transmitted if new frames are written to the STB meanwhile. The controller starts the transmission as soon as the bus becomes vacant and no request of the PTB (bit TPE) is pending. TSONE stays set until the frame has been transmitted successfully or it is aborted using TSA.
9	TSALL	Transmit secondary all frames. The host controller can set TSALL to '1' but cannot reset it to '0'. This would only be possible using TSA and aborting the frames. The bit will be reset to the hardware reset value if RESET = '1' or STBY = '1' or (LOM = '1' and LBME = '0'). If during a transmission the STB is loaded with a new frame, then the new frame will be transmitted too. In other words: a transmission initiated by TSALL is finished when the STB becomes empty.	
		0	No transmission for the STB.
		1	Transmission enable of all frames in the STB. The controller starts the transmission as soon as the bus becomes vacant and no request for the PTB (TPE = '1') is pending. TSALL stays set until all frames have been transmitted successfully or they are aborted using TSA.
8	TSA	Transmit secondary abort The bit has to be set by the host controller and will be reset by CANFD module. Setting TSA, automatically de-asserts TSONE or TSALL, respectively. The host controller can set TSA to '1' but cannot reset it to '0'. The bit will be reset to the hardware reset value if RESET = '1'. TSA should not be set simultaneously with TSONE or TSALL.	
		0	No abort
		1	Aborts a transmission from STB which has been requested but not started yet. For a TSONE transmission, only one frame is aborted while for TSALL Transmission, all frames are aborted. One or all frame slots will be released which updates TSSTAT. All aborted frames are lost because they are not accessible anymore. If in priority mode a TSONE transmission is aborted, then it is not clear which frame will be aborted if new frames are written to the STB meanwhile.
7	RESET	Software reset Some register (e.g., for node configuration) can only be modified if RESET = '1'. RESET forces several components to a reset state. Note that a CAN node will participate in CAN communication after RESET is switched to '0' after 11 CAN bit times. This delay is required by the CAN standard (bus idle time).	

		If RESET is set to ‘1’ and immediately set to ‘0’, then it takes some time until RESET can be read as ‘0’ and becomes inactive. The reason is clock domain crossing from host to CAN clock domain. RESET is held active as long as needed depending on the relation between MCLK and CANFD_CLK. Refer to 16.3.13.	
		0	No reset
		1	Reset of CANFD module
6	LBME	Loop Back Mode, External LBME cannot be changed while transmission is active. LBME shall not be changed together with TPE, TSONE or TSALL.	
		0	Disable
		1	Enable
5	LBMI	Loop Back Mode, Internal LBMI cannot be changed while transmission is active. LBMI shall not be changed together with TPE, TSONE or TSALL.	
		0	Disable
		1	Enable
0	BUSOFF	Bus off Writing a ‘1’ to BUSOFF will reset TECNT[7:0] and RECNT[7:0]. This shall be done only for debugging. Refer to 16.3.8	
		0	The CAN node status is "error active" or "error passive".
		1	The CAN node status is “bus off”.

NOTES:

1. In the CANFD module, the RESET bit in the CANFDn_CSR register must be '1' to allow writing. Refer to Section 16.3.13.
2. Section 16.3.9.4 and 16.3.12.2 provide further details about the RBALL is ‘1’.

16.5.9 CANFDn_ERR: CANFD n Error Register

The CANFDn_ERR register is a control register for the CANFD warning limit, error and arbitration lost capture, transmit and receive error count. This register is 32 bits wide.

CANFD0_ERR=0x4001_002C, CANFD1_ERR=0x4001_202C

31 30 29 28 27 26 25 24	23 22 21 20 19 18 17 16	15 14 13	12 11 10 9 8	7 6 5 4	3 2 1 0
TECNT[7:0]	RECNT[7:0]	KOER[2:0]	ALC[4:0]	AFWL[3:0]	EWL[3:0]
0x00	0x00	000	00000	0x1	0xB
RO	RO	RO	RO	RW	RW

31 24	TECNT[7:0]	Transmit error count TECNT[7:0] is incremented and decremented as defined in the CAN specification. In case of the “bus off state” TECNT[7:0] may overflow.
23 16	RECNT[7:0]	Receive error count RECNT[7:0] is incremented and decremented as defined in the CAN specification. RECNT[7:0] does not overflow.
15 13	KOER[2:0]	Kind of error (error code)
		000 No error
		001 Bit error
		010 Form error
		011 Stuff error
		100 Acknowledgement error
		101 CRC error
		110 Other error (Dominant bits after own error flag, received active error flag too long, dominant bit during passive-error-flag after ACK error)
		111 Reserved
12 8	ALC[4:0]	Arbitration lost capture ALC[4:0] provides the bit position in the frame where the arbitration has been lost.
7 4	AFWL[3:0](2)	Receive buffer almost full warning limit AFWL[3:0] defines the internal warning limit AFWL_i with n_RB being the number of available RB slots.
3 0	EWL[3:0](1)	Error warning limit = (EWL + 1) * 8. (Possible Limit values: 8, 16, … 128.) The value of EWL[3:0] controls EIF bit in CANFDn_ISR register. EWL[3:0] needs to be transferred using CDC from host to CAN clock domain. During transfer EWL register bits are write-locked for the host for a few clocks until CDC is complete.

NOTE:

1. The RESET bit in the CANFDn_CSR register must be ‘0’ to allow writing. Refer to 16.3.13.

16.5.10 CANFDn_AFCR: CANFD n Acceptance Filter Control Register

The CANFDn_AFCR register is a control register for the acceptance filter control. This register is 32 bits wide.

CANFD0_AFCR=0x4001_0044, CANFD1_AFCR=0x4001_2044

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15 14 13 12 11 10 9 8	7 6 5 4 3 2 1 0
AE15	AE14	AE13	AE12	AE11	AE10	AE9	AE8	AE7	AE6	AE5	AE4	AE3	AE2	AE1	AE0	Reserved	ACFADR[3:0]
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	-	0x0
RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	-	RW

x+16	AEx (x = 0 to 15)	Enable the acceptance filter
		0 Disables the acceptance filter
		1 Enables the acceptance filter
3 0	ACFADR[3:0]	Acceptance filter address ACFADR points to a specific acceptance filter. The selected filter is accessible using the registers ACFC and ACFM. A value of ACFADR > (ACF_NUMBER – 1) is meaningless. ACF_NUMBER is ‘16’.

NOTES:

1. The CANFDn_AFCR register can be written only when the RESET bit in the CANFDn_CSR register is set to '1'. Refer to 16.3.13.
2. The filter number of ACFC and ACFM can be selected using ACFADR[3:0]. Since ACFC and ACFM are modified using an internal pointer, they remain fixed at the same address even if the filter number is changed via ACFADR[3:0]. Refer to 16.3.6.

16.5.11 CANFDn_AFCIR: CANFD n Acceptance Filter Code Identifier Register

The CANFDn_AFCIR register is a control register for the acceptance filter code identifier. This register is 32 bits wide.

CANFD0_AFCIR=0x4001_0048, CAN1_AFCIR=0x4001_2048

31 30 29 28 27 26 25 24	23 22 21 20 19 18 17 16	15 14 13 12 11 10 9 8	7 6 5 4 3 2 1 0
Reserved	ID[28:0]		
-	0x0000000		
-	RW		

28 0	ID[28:0]	Frame Identifier

NOTES:

1. The CANFDn_AFCIR register can be written only when the RESET bit in the CANFDn_CSR register is set to '1'. Refer to 16.3.13.
2. Refer to Table 147 for more information about Frame Identifiers.

16.5.12 CANFDn_AFCFR: CANFD n Acceptance Filter Code Format Register

The CANFDn_AFCFR register is a control register for the acceptance filter code format. This register is 32 bits wide.

CANFD0_AFCFR=0x4001_004C, CANFD1_AFCFR=0x4001_204C

31 30 29	28	27	26 25 24	23 22 21	20	19	18	17	16	15 14 13 12 11 10 9 8	7 6 5 4 3 2 1 0
Reserved	LBF	ESI	KOER[2:0]	Reserved	RMF	Reserved	BRS	FDF	IDE	Reserved	DLC[3:0]
-	0	0	000	-	0	-	0	0	0	-	0x0
-	RW	RW	RW	-	RW	-	RW	RW	RW	-	RW

28	LBF	Loop back frame LBF for received frames is set to '1' if the loop back mode (Section 16.3.12.3) is activated and CANFD module has received its own transmitted frame. This can be useful if LBME = '1' while other nodes in the network also do transmissions.
		0 No effect
		1 Receive own frame
27	ESI	Error state indicator The protocol machine automatically embeds the correct value of ESI into transmitted frames. ESI is only included in CAN FD frames and therefore always 0 for classic CAN 2.0B frames for reception. The error state for transmission is shown with EPASS bit in CANFDn_ISR register.
		0 CAN node is error active
		1 CAN node is error passive
26 24	KOER[2:0]	Kind of error KOER[2:0] for received frames has the same meaning as the KOER[2:0] bits in CANFDn_ERR register. KOER[2:0] for received frames becomes meaningful if RBALL bit in CANFDn_CSR register is '1' (Section 16.3.9.4). If RBALL = '1', then acceptance filtering is disabled in general. KOER[2:0] is always zero inside a reception stream.
		000 No error
		001 Bit error
		010 Form error
		011 Stuff error
		100 Acknowledgement error
		101 CRC error
		110 Other error (Dominant bits after own error flag, received active error flag too long, dominant bit during passive-error-flag after ACK error)
		111 Reserved
20	RMF	Remote frame This maps to RTR or RRS depending on the frame format. See Figure 161 and Table 148. Remote frames carry zero bytes of payload. The value of the DLC[3:0] is transmitted as it is, but it has no impact to frame size. As a consequence, the value of the DLC[3:0] for a remote frame may carry some encoded information. Only classic CAN 2.0B frames can be remote frames. There is no remote frame for CAN FD. For CAN FD frames the following is valid: • Reception of CAN FD frames with RRS = '1' is possible.

		• Transmission of CAN FD frames with RMF = '1' will be executed as requested resulting in RRS = RMF, but it is forbidden by ISO 11898-1:2015. The host application needs to take care of this. • Always the DLC[3:0] is relevant for the length of a frame for transmission and reception. In the past remote frames were intended to be used to request a data frame with the same frame ID in classic CAN networks. This concept is prone to unresolvable collisions if more than one node may transmit a remote frame with the same ID but with different DLC[3:0]. Therefore, remote frames are recommended not to be used in general.
		0 Data frame
		1 Remote frame
18	BRS	CAN FD bit rate switch enable BRS has no meaning for classic CAN.
		0 Nominal/slow bit rate for the complete frame
		1 Switch to data/fast bit rate for the data payload and the CRC of CAN FD frames
17	FDF	CAN FD frame format
		0 Use CAN 2.0 frame (up to 8 bytes payload)
		1 Use CAN FD frame (up to 64 bytes payload)
16	IDE	Identifier extension (Refer to Table 147.)
		0 Standard Format: ID(28:18)
		1 Extended Format: ID(28:0)
3 0	DLC[3:0]	Data length code The DLC[3:0] defines the number of payload bytes inside the frame. The length of the DLC[3:0] for classic CAN 2.0B and CAN FD frames is 4 bits. For classic CAN 2.0B there is the option to transmit remote frames for which the DLC[3:0] has no meaning. The explanation for bit RMF provides further details.

NOTES:

1. The CANFDn_AFCFR register can be written only when the RESET bit in the CANFDn_CSR register is set to '1'. Refer to 16.3.13.
2. Refer to Table 149 for the number of payload bytes inside the frame according to the DLC[3:0] settings.

16.5.13 CANFDn_AFCTR: CANFD n Acceptance Filter Code Type Register

The CANFDn_AFCTR register is a control register for the acceptance filter code type. This register is 32 bits wide.

CANFD0_AFCTR=0x4001_0050, CANFD1_AFCTR=0x4001_2050

31 30 29 28 27 26 25 24	23 22 21 20 19 18 17 16	15 14 13 12 11 10 9 8	7 6 5 4 3 2 1 0
HANDLE[7:0]	Reserved		
0x00	-		
RW	-		

31 24	HANDLE[7:0]	Handle for frame identification. The purpose of the handle is frame identification using TSTAT (Section 16.3.5.4). HANDLE[7:0] is not used in the frame. It is suggested that the host application writes the value of a software counter to HANDLE[7:0]. Such a software counter could be incremented with every new frame to be transmitted and should roll over.

NOTES:

1. The CANFDn_AFCTR register can be written only when the RESET bit in the CANFDn_CSR register is set to '1'. Refer to 16.3.13.
2. To avoid an uninitialized memory issue during simulation, the HANDLE[7:0] shall always been defined by the host application for each frame to be transmitted.

16.5.14 CANFDn_AFMIR: CANFD n Acceptance Filter Mask Identifier Register

The CANFDn_AFMIR register serves as the mask register for the acceptance filter mask identifier. This register is 32 bits wide. Each bit can be masked independently. When a bit is set to '1', it remains unmasked, whereas setting it to '0' results in masking.

CANFD0_AFMIR=0x4001_0058, CANFD1_AFMIR=0x4001_2058

31 30 29 28 27 26 25 24	23 22 21 20 19 18 17 16	15 14 13 12 11 10 9 8	7 6 5 4 3 2 1 0
Reserved	ID[28:0]		
-	0x1FFFFFF		
-	RW		

28 0	ID[28:0]	Frame Identifier

NOTES:

1. The CANFDn_AFMIR register can be written only when the RESET bit in the CANFDn_CSR register is set to '1'. Refer to 16.3.13.
2. Refer to Table 147 for the information about Frame Identifiers.

16.5.15 CANFDn_AFMFR: CANFD n Acceptance Filter Mask Format Register

The CANFDn_AFMFR register is the mask register for the acceptance filter mask format. This register is 32 bits wide.

CAN0_ AFMFR=0x4001_005C, CAN1_ AFMFR=0x4001_205C

31 30 29	28	27	26 25 24	23 22 21	20	19	18	17	16	15 14 13 12 11 10 9 8	7 6 5 4 3 2 1 0
Reserved	LBF	ESI	KOER[2:0]	Reserved	RMF	Reserved	BRS	FDF	IDE	Reserved	DLC[3:0]
-	1	1	111	-	1	-	1	1	1	-	0xF
-	RW	RW	RW	-	RW	-	RW	RW	RW	-	RW

28	LBF	Loop back frame LBF for received frames is set to '1' if the loop back mode (Section 16.3.12.3) is activated and CANFD module has received its own transmitted frame. This can be useful if LBME = '1' while other nodes in the network also do transmissions.
	0	No effect
	1	Receive own frame
27	ESI	Error state indicator The protocol machine automatically embeds the correct value of ESI into transmitted frames. ESI is only included in CAN FD frames and therefore always 0 for classic CAN 2.0B frames for reception. The error state for transmission is shown with EPASS bit in CANFDn_ISR register.
	0	CAN node is error active
	1	CAN node is error passive
26 24	KOER[2:0]	Kind of error KOER[2:0] for received frames has the same meaning as the KOER[2:0] bits in CANFDn_ERR register. KOER[2:0] for received frames becomes meaningful if RBALL bit in CANFDn_CSR register is '1' (Section 16.3.9.4). If RBALL = '1', then acceptance filtering is disabled in general. KOER[2:0] is always zero inside a reception stream.
	000	No error
	001	Bit error
	010	Form error
	011	Stuff error
	100	Acknowledgement error
	101	CRC error
	110	Other error (Dominant bits after own error flag, received active error flag too long, dominant bit during passive-error-flag after ACK error)
	111	Reserved
20	RMF	Remote frame This maps to RTR or RRS depending on the frame format. See Figure 161 and Table 148. Remote frames carry zero byte payload. The value of the DLC[3:0] is transmitted as it is, but it has no impact to frame size. As a consequence, the value of the DLC[3:0] for a remote frame may carry some encoded information. Only classic CAN 2.0B frames can be remote frames. There is no remote frame for CAN FD. For CAN FD frames the following is valid: • Reception of CAN FD frames with RRS = '1' is possible.

		• Transmission of CAN FD frames with RMF = '1' will be executed as requested resulting in RRS = RMF, but it is forbidden by ISO 11898-1:2015. The host application needs to take care of this. • Always the DLC[3:0] is relevant for the length of a frame for transmission and reception. In the past remote frames were intended to be used to request a data frame with the same frame ID in classic CAN networks. This concept is prone to unresolvable collisions if more than one node may transmit a remote frame with the same ID but with different DLC[3:0]. Therefore, remote frames are recommended to not to be used in general.
		0 Data frame
		1 Remote frame
18	BRS	CAN FD bit rate switch enable BRS has no meaning for classic CAN.
		0 Nominal/slow bit rate for the complete frame
		1 Switch to data/fast bit rate for the data payload and the CRC of CAN FD frames
17	FDF	CAN FD frame format
		0 Use CAN 2.0 frame (up to 8 bytes payload)
		1 Use CAN FD frame (up to 64 bytes payload)
16	IDE	Identifier extension (Refer to Table 147.)
		0 Standard Format: ID(28:18)
		1 Extended Format: ID(28:0)
3 0	DLC[3:0]	Data length code The DLC[3:0] defines the number of payload bytes inside the frame. The length of the DLC[3:0] for classic CAN 2.0B and CAN FD frames is 4 bits. For classic CAN 2.0B there is the option to transmit remote frames for which the DLC[3:0] has no meaning. The explanation for bit RMF provides further details.

NOTES:

1. The CANFDn_AFMFR register can be written only when the RESET bit in the CANFDn_CSR register is set to '1'. Refer to 16.3.13.
2. Refer to Table 149 for the number of payload bytes inside the frame according to the DLC[3:0] settings.

16.5.16 CANFDn_AFMTR: CANFD n Acceptance Filter Mask Type Register

The CANFDn_AFMTR register is the mask register for the acceptance filter mask type. This register is 32 bits wide.

CAN0_AFMTR=0x4001_0060, CAN1_AFMTR=0x4001_2060

31 30 29 28 27 26 25 24	23 22 21 20 19 18 17 16	15 14 13 12 11 10 9 8	7 6 5 4 3 2 1 0
HANDLE[7:0]	Reserved		
0xFF	-		
RW	-		

31 24	HANDLE[7:0]	Handle for frame identification. The purpose of the handle is frame identification using TSTAT (Section 16.3.5.4). HANDLE[7:0] is not used in the frame. It is suggested that the host application writes the value of a software counter to HANDLE[7:0]. Such a software counter could be incremented with every new frame to be transmitted and should roll over.

NOTES:

1. The CANFDn_AFMTR register can be written only when the RESET bit in the CANFDn_CSR register is set to '1'. Refer to 16.3.13.
2. To avoid an uninitialized memory issue during simulation, the HANDLE[7:0] shall always been defined by the host application for each frame to be transmitted.

16.5.17 CANFDn_RBIR: CANFD n Receive Buffer Identifier Register

The CANFDn_RBIR register is identifier of receive buffer. This register is 32 bits wide.

CAN0_RBIR=0x4001_0070, CAN1_RBIR=0x4001_2070

31 30 29 28 27 26 25 24	23 22 21 20 19 18 17 16	15 14 13 12 11 10 9 8	7 6 5 4 3 2 1 0
Reserved	ID[28:0]		
-	0xXXXXXXX		
-	RW		

28 0	ID[28:0]	Frame Identifier

NOTE:

1. Refer to Table 147 for more information about Frame Identifiers.

16.5.18 CANFDn_RBFR: CANFD n Receive Buffer Format Register

The CANFDn_RBFR register is format of receive buffer. This register is 32 bits wide.

CANFD0_ RBFR=0x4001_0074, CANFD1_ RBFR=0x4001_2074

31 30 29	28	27	26 25 24	23 22 21	20	19	18	17	16	15 14 13 12 11 10 9 8	7 6 5 4 3 2 1 0
Reserved	LBF	ESI	KOER[2:0]	Reserved	RMF	Reserved	BRS	FDF	IDE	Reserved	DLC[3:0]
-	X	X	XXX	-	X	-	X	X	X	-	0xX
-	RW	RW	RW	-	RW	-	RW	RW	RW	-	RW

28	LBF	Loop back frame LBF for received frames is set to '1' if the loop back mode (Section 16.3.12.3) is activated and CANFD module has received its own transmitted frame. This can be useful if LBME = '1' while other nodes in the network also do transmissions.
		0 No effect
		1 Receive own frame
27	ESI	Error state indicator The protocol machine automatically embeds the correct value of ESI into transmitted frames. ESI is only included in CAN FD frames and therefore always 0 for classic CAN 2.0B frames for reception. The error state for transmission is shown with EPASS bit in CANFDn_ISR register.
		0 CAN node is error active
		1 CAN node is error passive
26 24	KOER[2:0]	Kind of error KOER[2:0] for received frames has the same meaning as the KOER[2:0] bits in CANFDn_ERR register. KOER[2:0] for received frames becomes meaningful if RBALL bit in CANFDn_CSR register is '1' (Section 16.3.9.4). If RBALL = '1', then acceptance filtering is disabled in general. KOER[2:0] is always zero inside a reception stream.
		000 No error
		001 Bit error
		010 Form error
		011 Stuff error
		100 Acknowledgement error
		101 CRC error
		110 Other error (Dominant bits after own error flag, received active error flag too long, dominant bit during passive-error-flag after ACK error)
		111 Reserved
20	RMF	Remote frame This maps to RTR or RRS depending on the frame format. See Figure 161 and Table 148. Remote frames carry zero bytes of payload. The value of the DLC[3:0] is transmitted as it is, but it has no impact to frame size. As a consequence, the value of the DLC[3:0] for a remote frame may carry some encoded information. Only classic CAN 2.0B frames can be remote frames. There is no remote frame for CAN FD. For CAN FD frames the following is valid: • Reception of CAN FD frames with RRS = '1' is possible. • Transmission of CAN FD frames with RMF = '1' will be executed as requested resulting in RRS = RMF, but it is forbidden by ISO

		11898-1:2015. The host application needs to take care of this. • Always the DLC[3:0] is relevant for the length of a frame for transmission and reception. In the past remote frames were intended to be used to request a data frame with the same frame ID in classic CAN networks. This concept is prone to unresolvable collisions if more than one node may transmit a remote frame with the same ID but with different DLC[3:0]. Therefore, remote frames are recommended not to be used in general.
		0 Data frame
		1 Remote frame
18	BRS	CAN FD bit rate switch enable BRS has no meaning for classic CAN.
		0 Nominal/slow bit rate for the complete frame
		1 Switch to data/fast bit rate for the data payload and the CRC of CAN FD frames
17	FDF	CAN FD frame format
		0 Use CAN 2.0 frame (up to 8 bytes payload)
		1 Use CAN FD frame (up to 64 bytes payload)
16	IDE	Identifier extension (Refer to Table 147.)
		0 Standard Format: ID(28:18)
		1 Extended Format: ID(28:0)
3 0	DLC[3:0]	Data length code The DLC[3:0] defines the number of payload bytes inside the frame. The length of the DLC[3:0] for classic CAN 2.0B and CAN FD frames is 4 bits. For classic CAN 2.0B there is the option to transmit remote frames for which the DLC[3:0] has no meaning. The explanation for bit RMF provides further details.

NOTE:

1. Refer to Table 149 for the number of payload bytes inside the frame according to the DLC[3:0] settings.

16.5.19 CANFDn_RBTR: CANFD n Receive Buffer Type Register

The CANFDn_RBTR register is type of receive buffer. This register is 32 bits wide.

CANFD0_ RBTR=0x4001_0078, CANFD1_ RBTR=0x4001_2078

31 30 29 28 27 26 25 24	23 22 21 20 19 18 17 16	15 14 13 12 11 10 9 8	7 6 5 4 3 2 1 0
HANDLE[7:0]	Reserved		
0xXX	-		
RW	-		

31 24	HANDLE[7:0]	Handle for frame identification. The purpose of the handle is frame identification using TSTAT (Section 16.3.5.4). HANDLE[7:0] is not used in the frame. It is suggested that the host application writes the value of a software counter to HANDLE[7:0]. Such a software counter could be incremented with every new frame to be transmitted and should roll over.

NOTE:

1. To avoid an uninitialized memory issue during simulation, the HANDLE[7:0] shall always been defined by the host application for each frame to be transmitted.

16.5.20 CANFDn_RBPDR: CANFD n Receive Buffer Payload Data Register

The CANFDn_RBPDR register is payload data of receive buffer. This register is 32 bits wide. Depth of receive buffer is 23.

CANFD0_RBPDR=0x4001_0080, CANFD1_RBPDR=0x4001_2080

31 30 29 28 27 26 25 24	23 22 21 20 19 18 17 16	15 14 13 12 11 10 9 8	7 6 5 4 3 2 1 0
RBPDR[31:0]			
0xXXXXXXXX			
RW			

31 0	RBPDR[31:0]	Payload data of the frame. Up to 8 bytes for CAN CC, up to 64 bytes for CAN FD. See also Table 149 for further details. Unused payload data words in RB do contain garbage data which needs to be ignored.

16.5.21 CANFDn_TBIR: CANFD n Transmit Buffer Identifier Register

The CANFDn_TBIR register is identifier of transmit buffer. This register is 32 bits wide.

CANFD0_TBIR=0x4001_0890, CANFD1_TBIR=0x4001_2890

31 30 29 28 27 26 25 24	23 22 21 20 19 18 17 16	15 14 13 12 11 10 9 8	7 6 5 4 3 2 1 0
Reserved	ID[28:0]		
-	0xXXXXXXX		
-	RW		

28 0	ID[28:0]	Frame Identifier

NOTE:

1. Refer to Table 147 for more information about Frame Identifiers.

16.5.22 CANFDn_TBFR: CANFD n Transmit Buffer Format Register

The CANFDn_TBFR register is format of transmit buffer. This register is 32 bits wide.

CANFD0_TBFR=0x4001_0894, CANFD1_TBFR=0x4001_2894

31 30 29	28	27	26 25 24	23 22 21	20	19	18	17	16	15 14 13 12 11 10 9 8	7 6 5 4 3 2 1 0
Reserved	LBF	ESI	KOER[2:0]	Reserved	RMF	Reserved	BRS	FDF	IDE	Reserved	DLC[3:0]
-	X	X	XXX	-	X	-	X	X	X	-	0xX
-	RW	RW	RW	-	RW	-	RW	RW	RW	-	RW

28	LBF	Loop back frame LBF for received frames is set to '1' if the loop back mode (Section 16.3.12.3) is activated and CANFD module has received its own transmitted frame. This can be useful if LBME = '1' while other nodes in the network also do transmissions.
	0	No effect
	1	Receive own frame
27	ESI	Error state indicator The protocol machine automatically embeds the correct value of ESI into transmitted frames. ESI is only included in CAN FD frames and therefore always 0 for classic CAN 2.0B frames for reception. The error state for transmission is shown with EPASS bit in CANFDn_ISR register.
	0	CAN node is error active
	1	CAN node is error passive
26 24	KOER[2:0]	Kind of error KOER[2:0] for received frames has the same meaning as the KOER[2:0] bits in CANFDn_ERR register. KOER[2:0] for received frames becomes meaningful if RBALL bit in CANFDn_CSR register is '1' (Section 16.3.9.4). If RBALL = '1', then acceptance filtering is disabled in general. KOER[2:0] is always zero inside a reception stream.
	000	No error
	001	Bit error
	010	Form error
	011	Stuff error
	100	Acknowledgement error
	101	CRC error
	110	Other error (Dominant bits after own error flag, received active error flag too long, dominant bit during passive-error-flag after ACK error)
	111	Reserved
20	RMF	Remote frame This maps to RTR or RRS depending on the frame format. See Figure 161 and Table 148. Remote frames carry zero bytes of payload. The value of the DLC[3:0] is transmitted as it is, but it has no impact to frame size. As a consequence, the value of the DLC[3:0] for a remote frame may carry some encoded information. Only classic CAN 2.0B frames can be remote frames. There is no remote frame for CAN FD. For CAN FD frames the following is valid: • Reception of CAN FD frames with RRS = '1' is possible. • Transmission of CAN FD frames with RMF = '1' will be executed as requested resulting in RRS = RMF, but it is forbidden by ISO

		11898-1:2015. The host application needs to take care of this. • Always the DLC[3:0] is relevant for the length of a frame for transmission and reception. In the past remote frames were intended to be used to request a data frame with the same frame ID in classic CAN networks. This concept is prone to unresolvable collisions if more than one node may transmit a remote frame with the same ID but with different DLC[3:0]. Therefore, remote frames are recommended to not to be used in general.
		0 Data frame
		1 Remote frame
18	BRS	CAN FD bit rate switch enable BRS has no meaning for classic CAN.
		0 Nominal/slow bit rate for the complete frame
		1 Switch to data/fast bit rate for the data payload and the CRC of CAN FD frames
17	FDF	CAN FD frame format
		0 Use CAN 2.0 frame (up to 8 bytes payload)
		1 Use CAN FD frame (up to 64 bytes payload)
16	IDE	Identifier extension (Refer to Table 147.)
		0 Standard Format: ID(28:18)
		1 Extended Format: ID(28:0)
3 0	DLC[3:0]	Data length code The DLC[3:0] defines the number of payload bytes inside the frame. The length of the DLC[3:0] for classic CAN 2.0B and CAN FD frames is 4 bits. For classic CAN 2.0B there is the option to transmit remote frames for which the DLC[3:0] has no meaning. The explanation for bit RMF provides further details.

NOTE:

1. Refer to Table 149 for the number of payload bytes inside the frame according to the DLC[3:0] settings.

16.5.23 CANFDn_TBTR: CANFD n Transmit Buffer Type Register

The CANFDn_TBTR register is the type of transmit buffer. This register is 32 bits wide.

CANFD0_TBTR=0x4001_0898, CANFD1_TBTR=0x4001_2898

31 30 29 28 27 26 25 24	23 22 21 20 19 18 17 16	15 14 13 12 11 10 9 8	7 6 5 4 3 2 1 0
HANDLE[7:0]	Reserved		
0xXX	-		
RW	-		

31 24	HANDLE[7:0]	Handle for frame identification. The purpose of the handle is frame identification using TSTAT (Section 16.3.5.4). HANDLE[7:0] is not used in the frame. It is suggested that the host application writes the value of a software counter to HANDLE[7:0]. Such a software counter could be incremented with every new frame to be transmitted and should roll over.

NOTE:

1. To avoid an uninitialized memory issue during simulation, the HANDLE[7:0] shall always been defined by the host application for each frame to be transmitted.

16.5.24 CANFDn_TBPDR: CANFD n Transmit Buffer Payload Data Register

The CANFDn_TBPDR register is payload data of transmit buffer. This register is 32 bits wide. Depth of transmit buffer is 16.

CANFD0_TBPDR=0x4001_08A0, CANFD1_TBPDR=0x4001_28A0

31 30 29 28 27 26 25 24	23 22 21 20 19 18 17 16	15 14 13 12 11 10 9 8	7 6 5 4 3 2 1 0
TBPDR[31:0]			
0xXXXXXXXX			
RW			

31 0	TBPDR[31:0]	Payload data of the frame. Up to 8 bytes for CAN CC, up to 64 bytes for CAN FD. See also Table 149 for further details.

16.5.25 CANFDn_TSCFGR: CANFD n Transmit Stream Configuration Register

The CANFDn_TSCFGR register configures the transmit stream. This register is 32 bits wide.

CANFD0_TSCFGR=0x4001_10B0, CANFD1_TSCFGR=0x4001_30B0

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

0	RBEN	Enable the reception buffer. If enabled, then received frames are forwarded to RB. If RBEN is changed into any direction while there is a frame in reception, then this may result in the loss of the frame. Therefore, it is recommended to set RBEN to the desired value before communication starts. It is suggested to do it while RESET = '1'. If RBEN = '0', then RIF bit, ROIF bit, RFIF bit, RAFIF bit in CANFDn_ISR register and RSTAT[1:0] bits in CANFDn_CSR register will be reset to '0'. If any frame was stored in RB before RBEN was set to '0', then all frames inside RB will be lost when RBEN = '0'. 0 Disables the reception buffer 1 Enables the reception buffer

NOTE:

1. The CANFDn_TSCFGR register can be written only when the RESET bit in the CANFDn_CSR register is set to '1'. Refer to 16.3.13.

16.5.26 CANFDn_LFCIR: CANFD n LLC Frame Calculator Input Register

This register is 32 bits wide.

CANFD0_LFCIR=0x4001_10B4, CANFD1_LFCIR=0x4001_30B4

31 30 29 28 27 26 25 24	23 22 21 20	19 18	17 16	15 14 13 12 11 10 9 8	7 6 5 4 3 2 1 0
Reserved	RMF	Reserved	FDF	Reserved	DLC[3:0]
-	0	-	0	-	0x0
-	RW	-	RW	-	RW

20	RMF	Remote frame
17	FDF	CAN FD frame format
3 0	DLC[3:0]	Data length code

NOTES:

1. Write access to CANFDn_LFCIR register results in the automatic calculation of the size information of an LLC frame inside LPB[15:0] bits in CANFDn_LFCOR register.
2. The layout of CANFDn_LFCIR register is identical to the layout of the format word of the *CANFDn_*TR* register. (AFC, AFM, RB, TB)
3. Only bits relevant for the calculation of LPB[15:0] do exist inside CANFDn_LFCIR register. Other bits must be '0'. Relevant bits are: DLC, FDF and RMF

16.5.27 CANFDn_LFCOR: CANFD n LLC Frame Calculator Output Register

This register is 32 bits wide.

CANFD0_LFCOR=0x4001_10B8, CANFD1_LFCOR=0x4001_30B8

31 30 29 28 27 26 25 24	23 22 21 20 19 18 17 16	15 14 13 12 11 10 9 8	7 6 5 4 3 2 1 0
Reserved		LPB[15:0]	
-		0x0000	
-		R0	

15 0	LPB[15:0]	LLC frame payload bytes LPB[15:0] provides the number of payload data bytes inside LLC frame depending on the value inside CANFDn_LFCIR register.

16.5.28 CANFD Register Map Summary

Table 162. CANFD Register Map Summary

Offset	Register name	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0x00	CANFDn_VNCFGR	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	ROP	CES	VERSION[7:0]															
	Reset value															0	1	0	0	0	0	1	0	0	0	0	0	0	1	0	0	1
0x04	CANFDn_ACBTR	Res	AC_SJW[6:0]							Res	AC_SEG2[6:0]							Res	Res	Res	Res	Res	Res	Res	AC_SEG1[8:0]								
	Reset value		0	0	0	0	1	0	1		0	0	0	0	1	0	1								0	0	0	0	0	1	0	0	0
0x08	CANFDn_FDBTR	Res	FD_SJW[6:0]							Res	FD_SEG2[6:0]							Res	Res	Res	Res	Res	Res	Res	Res	FD_SEG1[7:0]							
	Reset value		0	0	0	0	0	1	0		0	0	0	0	0	1	0									0	0	0	0	0	0	1	1
0x10	CANFDn_PTDCR	Res	RETLIM[2:0]			Res	REALIM[2:0]			Res	Res	Res	Res	Res	Res	Res	Res	FD_SSPOFF[7:0]								Res	Res	Res	PRS[4:0]					
	Reset value		1	1	1		1	1	1									0	0	0	0	0	0	0	0				0	0	0	0	0
0x14	CANFDn_ISR	EWARN	EPASS	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	EPIF	ALIF	BEIF	RIF	ROIF	RFIF	RAFIF	TPIF	TSIF	EIF	AIF
	Reset value	0	0																				0	0	0	0	0	0	0	0	0	0	0
0x18	CANFDn_IER	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	EPIE	ALIE	BEIE	RIE	ROIE	RFIE	RAFIE	TPIE	TSIE	EIE	Res
	Reset value																						0	0	0	1	1	1	1	1	1	1	
0x1C	CANFDn_TSR	Res	Res	Res	Res	Res	TSTAT2[2:0]			HANDLE2[7:0]								Res	Res	Res	Res	Res	TSTAT1[2:0]			HANDLE1[7:0]							
	Reset value						0	0	0	0	0	0	0	0	0	0	0						0	0	0	0	0	0	0	0	0	0	0
0x28	CANFDn_CSR	SACK	ROM	ROV	RREL	RBALL	Res	RSTAT [1:0]		FD_ISO	TSNEXT	TSMODE	Res	Res	TSFF	TSSTAT [1:0]		TBSEL	LOM	STBY	TPE	TPA	TSONE	TSALL	TSA	RESET	LBME	LBMI	Res	Res	Res	Res	BUSOFF
	Reset value	0	0	0	0	0		0	0	1	0	0			0	0	0	0	0	0	0	0	0	0	0	1	0	0					0
0x2C	CANFDn_ERR	TECNT[7:0]								RECNT[7:0]								KOER[2:0]			ALC[4:0]					AFWL[3:0]				EWL[3:0]			
	Reset value	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	1	0	1	1

Table 162. CANFD Register Map Summary (continued)

Offset	Register name	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0x44	CANFDn_AFCR	AE15	AE14	AE13	AE12	AE11	AE10	AE9	AE8	AE7	AE6	AE5	AE4	AE3	AE2	AE1	AE0	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	ACFADR [3:0]			
	Reset value	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1													0	0	0	0
0x48	CANFDn_AFCIR	Res	Res	Res	ID[28:0]																												
	Reset value				0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
0x4C	CANFDn_AFCFR	Res	Res	Res	LBF	ESI	KOER[2:0]			Res	Res	Res	RMF	Res	BRS	FDF	IDE	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	DLC[3:0]			
	Reset value				0	0	0	0	0				0		0	0	0													0	0	0	0
0x50	CANFDn_AFCTR	HANDLE[7:0]								Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res
	Reset value	0	0	0	0	0	0	0	0																								
0x58	CANFDn_AFMIR	Res	Res	Res	ID[28:0]																												
	Reset value				1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1
0x5C	CANFDn_AFMFR	Res	Res	Res	LBF	ESI	KOER[2:0]			Res	Res	Res	RMF	Res	BRS	FDF	IDE	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	DLC[3:0]			
	Reset value				1	1	1	1	1				1		1	1	1													1	1	1	1
0x60	CANFDn_AFMTR	HANDLE[7:0]								Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res
	Reset value	1	1	1	1	1	1	1	1																								
0x70	CANFDn_RBIR	Res	Res	Res	ID[28:0]																												
	Reset value				x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x
0x74	CANFDn_RBFR	Res	Res	Res	LBF	ESI	KOER[2:0]			Res	Res	Res	RMF	Res	BRS	FDF	IDE	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	DLC[3:0]			
	Reset value				x	x	x	x	x				x		x	x	x													x	x	x	x
0x78	CANFDn_RBTR	HANDLE[7:0]								Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res
	Reset value	x	x	x	x	x	x	x	x																								
0x80	CANFDn_RBPDR	RBPDR[31:0]																															
	Reset value	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x

Table 162. CANFD Register Map Summary (continued)

Offset	Register name	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0x890	CANFDn_TBIR	Res.	Res.	Res.	ID[28:0]																												
	Reset value				x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x
0x894	CANFDn_TBFR	Res.	Res.	Res.	LBF	ESI	KOER[2:0]			Res.	Res.	Res.	RMF	Res.	BRS	FDF	IDE	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	DLC[3:0]			
	Reset value				x	x	x	x	x				x		x	x	x												x	x	x	x
0x898	CANFDn_TBTR	HANDLE[7:0]								Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.
	Reset value	x	x	x	x	x	x	x	x																								
0x8A0	CANFDn_TBPDR	TBPDR[31:0]																															
	Reset value	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x
0x10B0	CANFDn_TSCFGR	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	RBEN
	Reset value																																1
0x10B4	CANFDn_LFCIR	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	RMF	Res.	Res.	FDF	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	DLC[3:0]			
	Reset value												0			0														0	0	0	0
0x10B8	CANFDn_LFCOR	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	LPB[15:0]															
	Reset value																	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

17. Motor Pulse Width Modulation (MPWM)

17.1 MPWM Introduction

The A34G43x series supports two Motor Pulse Width Modulation (MPWM) units. The MPWM in the A34G43x operates total of three PWM modes as shown below to support various applications:

- Motor PWM mode for motor control
  - Supports up-down count and up-count PWM mode.
- Normal PWM mode for a variety of PWM applications.

In Motor or Normal PWM mode, the MPWM module of the A34G43x many different PWM control functionalities such as up to six channels' outputs, dead-time, up to six ADC triggers, a variety of interrupt event generations in many different conditions, protection events, and so on.

17.2 MPWM Main Features

The MPWM module of the A34G43x has the following main features in Motor and Normal PWM modes:

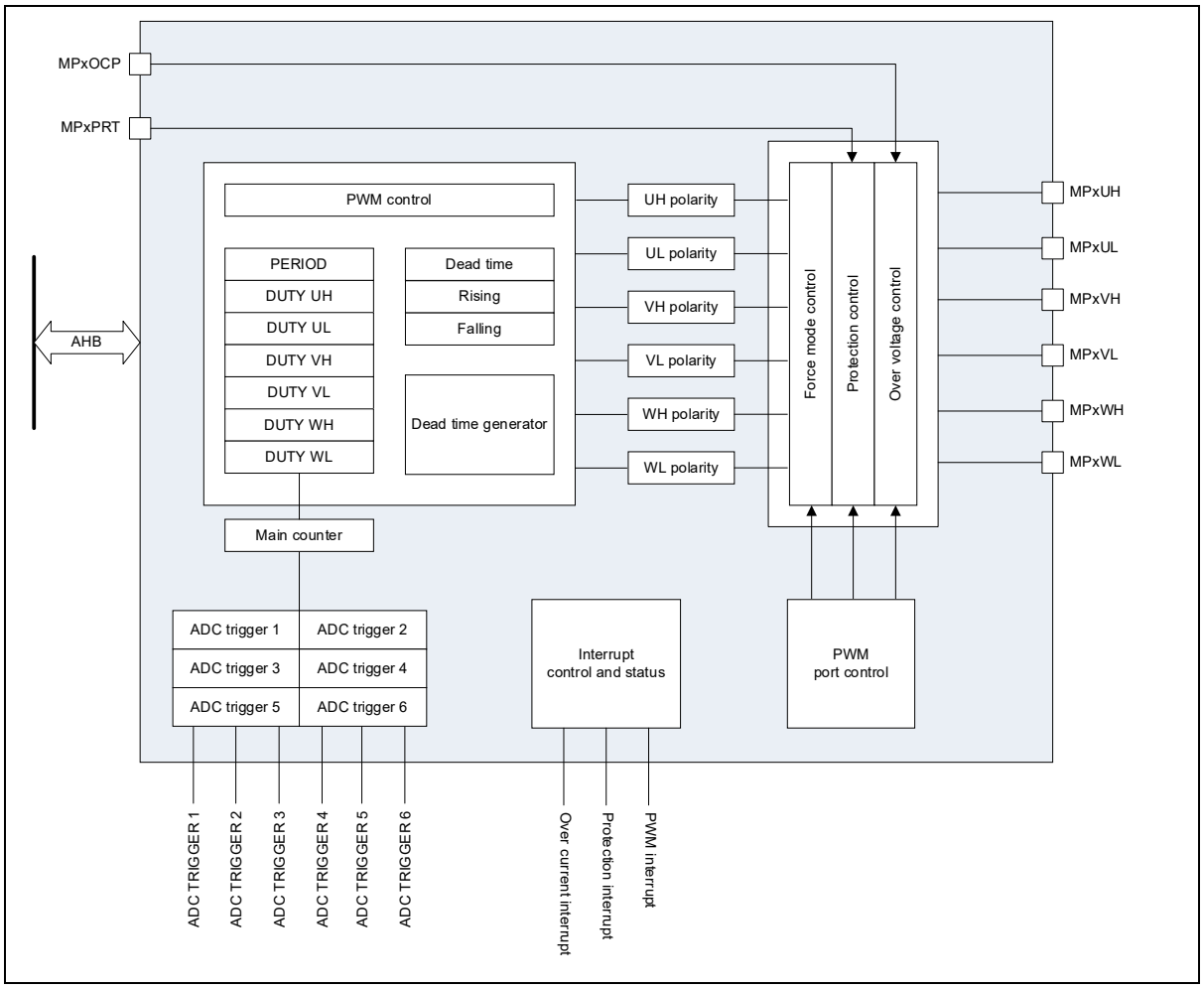
- Outputs from up to six channels
- 16-bit counter that repeats operation
- Up count/up-down count mode
- Symmetrical/asymmetrical mode is supported.
- Dead-time for half-bridge circuit is supported.
- Eight ADC trigger outputs
  - Six variable triggers
  - Two fixed trigger (Bottom, Period Match)
- Interval interrupt mode
- Processing of the protection and overcurrent events

17.3 MPWM Functional Description

17.3.1 MPWM Block Diagram

Figure 173 shows a block diagram of MPWM.

Figure 173. MPWM Block Diagram



NOTE:

1. 'n' is MPWM unit 0 or 1.

17.3.2 Pins and Internal Signals

Table 163 describes external pins connected to the MPWM module.

Table 163. Pin Assignment of MPWM: External Pins

Pin Name	Type	Description	Supported Packages		
			A34G43xYL (120-LQFP)	A34G43xVL (100-LQFP)	A34G43xRL (64-LQFP)
MP0UH	O	MPWM0 high side output port of phase U	O	O	O
MP0UL	O	MPWM0 low side output port of phase U	O	O	O
MP0VH	O	MPWM0 high side output port of phase V	O	O	O
MP0VL	O	MPWM0 low side output port of phase V	O	O	O
MP0WH	O	MPWM0 high side output port of phase W	O	O	O
MP0WL	O	MPWM0 low side output port of phase W	O	O	O
MP0PRT	I	Protection input pin dedicated to MPWM0	O	O	O
MP0OCP	I	Overcurrent input pin dedicated to MPWM0	O	O	O
MP1UH	O	MPWM1 high side output port of phase U	O	O	O
MP1UL	O	MPWM1 low side output port of phase U	O	O	O
MP1VH	O	MPWM1 high side output port of phase V	O	O	O
MP1VL	O	MPWM1 low side output port of phase V	O	O	O
MP1WH	O	MPWM1 high side output port of phase W	O	O	O
MP1WL	O	MPWM1 low side output port of phase W	O	O	O
MP1PRT	I	Protection input pin dedicated to MPWM1	O	O	O
MP1OCP	I	Overcurrent input pin dedicated to MPWM1	O	O	O

NOTE:

1. ‘n’ is MPWM unit 0 or 1.

MPWM0 and MPWM1 have pins assigned for the protection operation as described in Table 163. Pin Assignment of MPWM: External Pins.

For more information about the protection operation, refer to 17.3.14.

17.3.3 Time-Base Unit

The MPWM module of the A34G43x series has a 16-bit counter that is a main component of the MPWM module. This 16-bit counter supports up counter and up-down counter modes, and the input clock for the counter can be controlled by the HCLK clock.

Table 164 describes registers related to the MPWM operation time.

Table 164. Time-Base Register of MPWMn

Name	PWM Mode	Description	Access Type
MPWMn_PRD	Motor/normal	PWM period setting register	Read-write
MPWMn_CR1	Motor/normal	Period interrupt interval control register	Read-write
MPWMn_CNT	Motor/normal	Counter register	Read-Only

17.3.4 Counter Modes

In Motor PWM mode, the up counter and up-down counter are supported, depending on the set value of the MOTORB in MPWMn_MR register. However, in Normal PWM mode, only up counter is supported.

Counter modes available in each PWM mode are listed in Table 165.

Table 165. Counter Modes of MPWM

PWM Mode	Counter Mode
Motor PWM mode 0	Up-down counter
Motor PWM mode 1	Up counter
Normal PWM mode	Up counter

If the counter operates in up counter mode, the counter is incremented by one starting from one to PERIOD[15:0]. When the counter increases and reaches PERIOD[15:0], then the counter is reset to one at the next clock and is incremented by one starting from one again. This operation repeats until the counter stops.

Period T in up counter mode can be calculated by the formular below:

$$T_{Period} = PERIOD[15:0] / MPWM\ clock$$

If the counter operates in up-down counter mode, the counter is incremented by one starting from one to PERIOD[15:0], the same as in up-counter mode. However, when the counter increases and reaches PERIOD[15:0], the counter operates as a down counter from the next clock, decrementing by one. On reaching zero, the counter operates as an up counter from the next clock, incrementing by one.

Period T in up-down counter mode can be calculated by the formular below:

$$T_{Period} = 2 \times PERIOD[15:0] / MPWM\ clock$$

17.3.5 Symmetric and Asymmetric Mode

In Up-down counter mode, the PWM can output symmetrically around the period matching event or asymmetrically, by configuring the MCHMOD[1:0] in MPWMn_MR register.

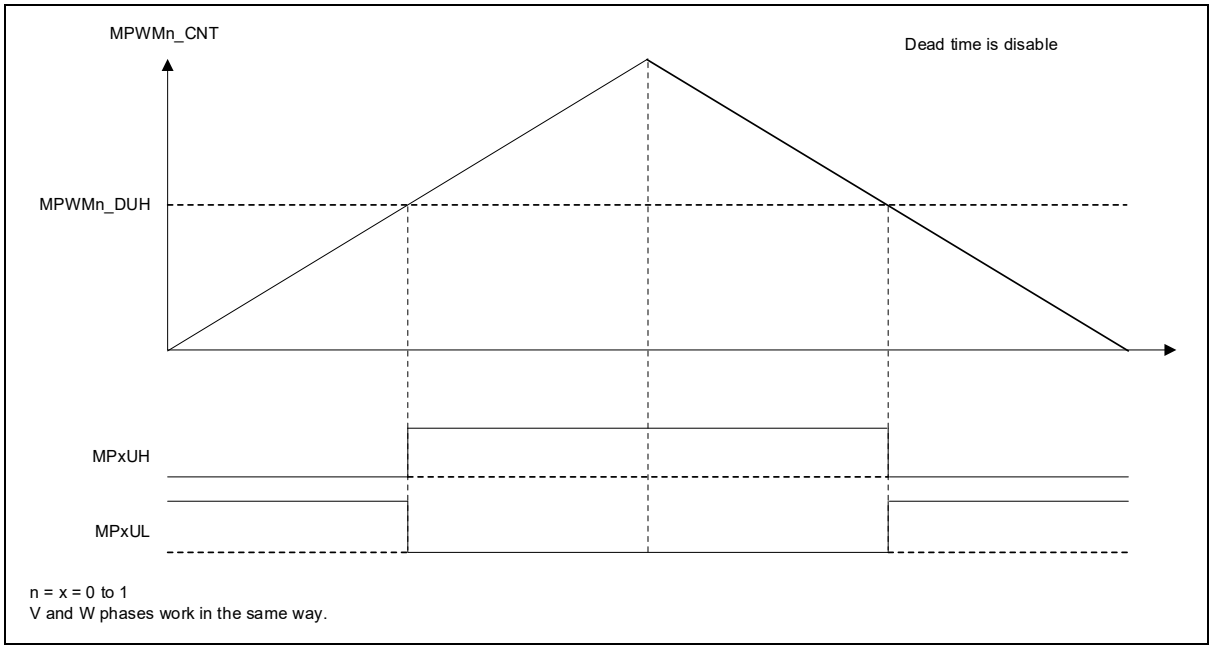
However, in Normal PWM mode, the PWM is fixed to output symmetrically, regardless of the value set in the MCHMOD[1:0] in MPWMn_MR register.

In symmetric mode, there are two types of modes as described below:

- 2-ch symmetric mode: The high and low sides of the output can be controlled independently.
- 1-ch symmetric mode: The high and low sides of the output can be controlled in complementary symmetry.

Figure 174 shows a timing diagram of the PWM outputs in 1-ch symmetric mode.

Figure 174. 1-Ch Symmetric Mode Timing Diagram

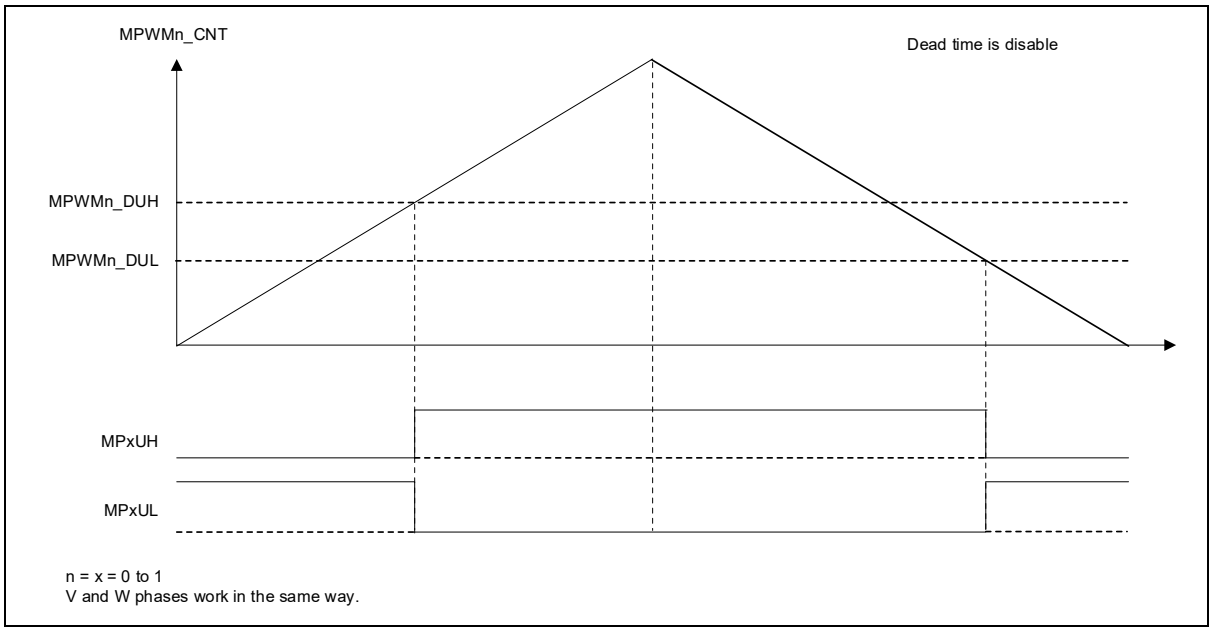


In asymmetric mode, there is only one type of mode as described below:

- 1-ch asymmetric mode: The high and low sides of the output can be controlled in complementary symmetry.

Figure 175 shows a timing diagram of the PWM outputs in 1-ch asymmetric mode.

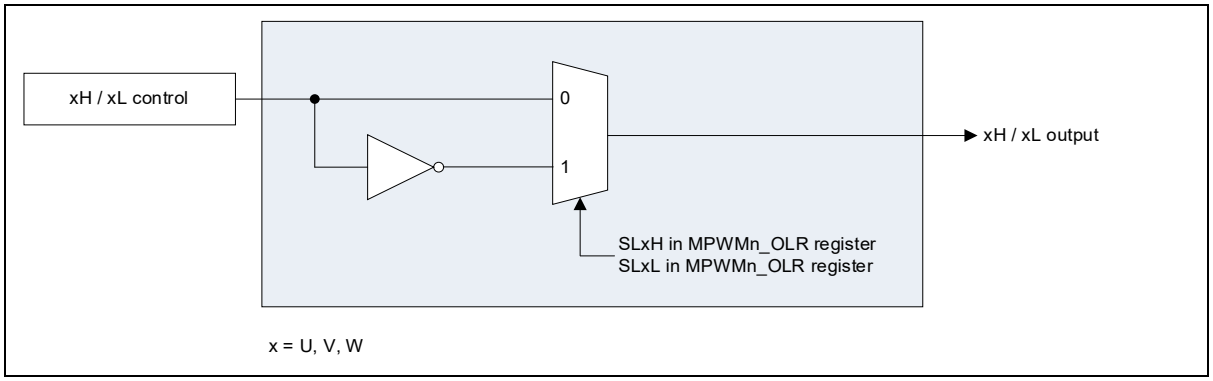
Figure 175. 1-Ch Asymmetric Mode Timing Diagram



17.3.6 Output Control

The MPWM module includes an output control block that controls the outputs as shown in Figure 176.

Figure 176. Output Control Block



With the configuration of the output control block, users can set the output levels both of the operating counter and the non-operating counter by using the MPWMn_OLR register. The users simply set the MPWMn_OLR register considering the external circuit's switch polarity and desired PWM type.

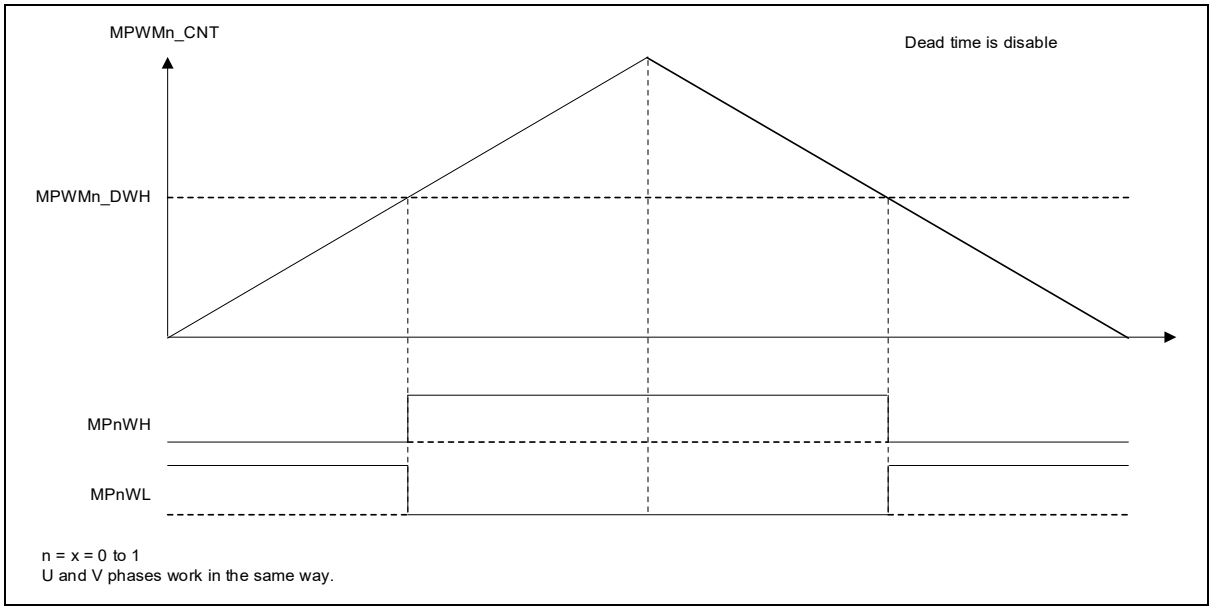
Each bit of the MPWMn_OLR register, such as WHL, VHL, UHL, WLL, VLL, and ULL, determines the output level of the PWM when the counter starts, as described in Table 166.

Table 166. Output Level (MPWM_OLR = 0x00)

PWM Output	Level	Motor PWM Mode		Normal PWM Mode (MOTORB[1:0] = 2)
		Up Count Mode (MOTORB[1:0] = 1)	Up-Down Count Mode (MOTORB[1:0] = 0)	
MPxWH	Basic output level	LOW	LOW	LOW
	Inverted output level	HIGH	HIGH	HIGH
MPxWL	Basic output level	HIGH	HIGH	LOW
	Inverted output level	LOW	LOW	HIGH
MPxVH	Basic output level	LOW	LOW	LOW
	Inverted output level	HIGH	HIGH	HIGH
MPxVL	Basic output level	HIGH	HIGH	LOW
	Inverted output level	LOW	LOW	HIGH
MPxUH	Basic output level	LOW	LOW	LOW
	Inverted output level	HIGH	HIGH	HIGH
MPxUL	Basic output level	HIGH	HIGH	LOW
	Inverted output level	LOW	LOW	HIGH

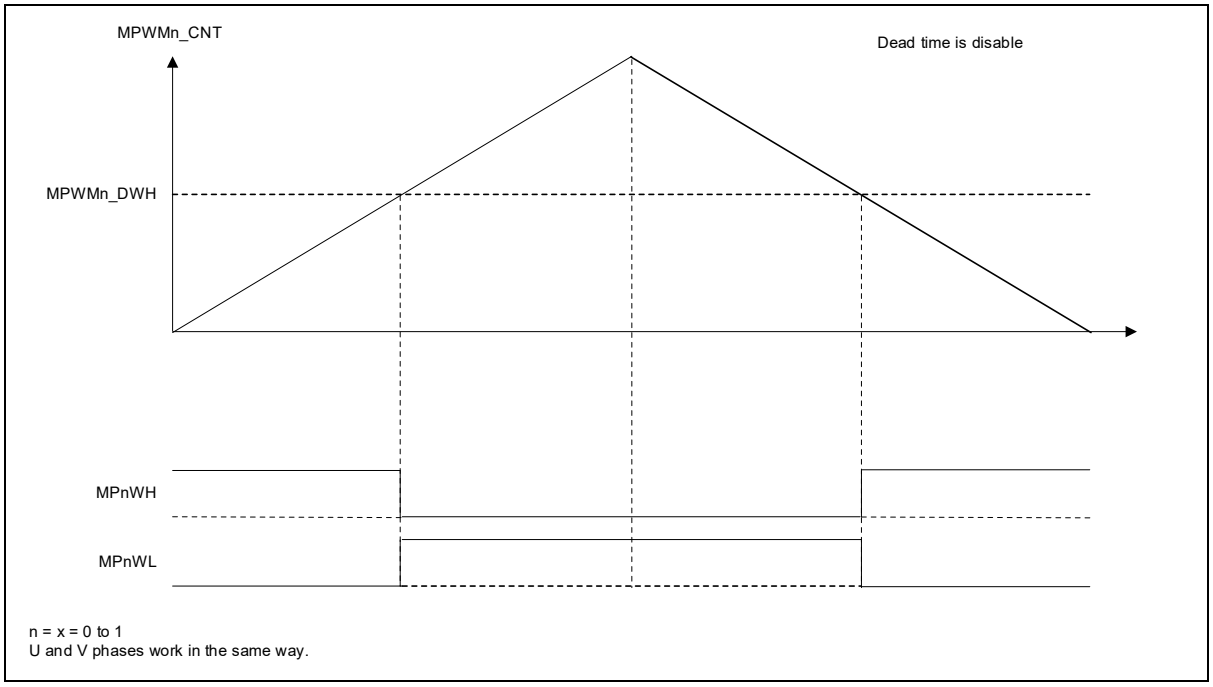
For example, if the WHL and WLL bits of the MPWMn_OLR register are set to ‘0’ in Motor PWM mode (1-ch symmetric mode), the 1 period output of the MPnWH and MPnWL appears as shown in Figure 177.

Figure 177. MPnWH/MPnWL Output at Motor PWM Mode (WHL = ‘0’, WLL = ‘0’)



If the WHL and WLL bits of the MPWMn_OLR register are set to ‘1’ in Motor PWM mode (1-ch symmetric mode), the 1 period output of the MPnWH and MPnWL appears as shown in Figure 178.

Figure 178. MPnWH/MPnWL Output at Motor PWM Mode (WHL = ‘1’, WLL = ‘1’)



Each bit of the MPWMn_OLR register, such as DOLWH, DOLVH, DOLUH, DOLWL, DOLVL, and DOLUL, determines the output level of the PWM when the counter is disabled. Please remember that the output level can be a polar reversal according to the values of the bits WHL, VHL, UHL, WLL, VLL, and ULL.

17.3.7 PWM Modes

PWM modes can be controlled by configuring the MOTORB[1:0] in MPWMn_MR register. Two PWM modes such as Motor, Normal PWM modes are available for the PWM operations as described in Table 167.

Table 167. PWM Modes of MPWM

PWM Mode	Counter Mode	Symmetric/Asymmetric Mode
Motor PWM mode 0	Up-down counter	2-ch symmetric mode
		1-ch asymmetric mode
		1-ch symmetric mode
Motor PWM mode 1	Up counter	-
Normal PWM mode	Up counter	-

17.3.7.1 Motor PWM Mode 0/Up-Down Counter/2-Ch Symmetric Mode

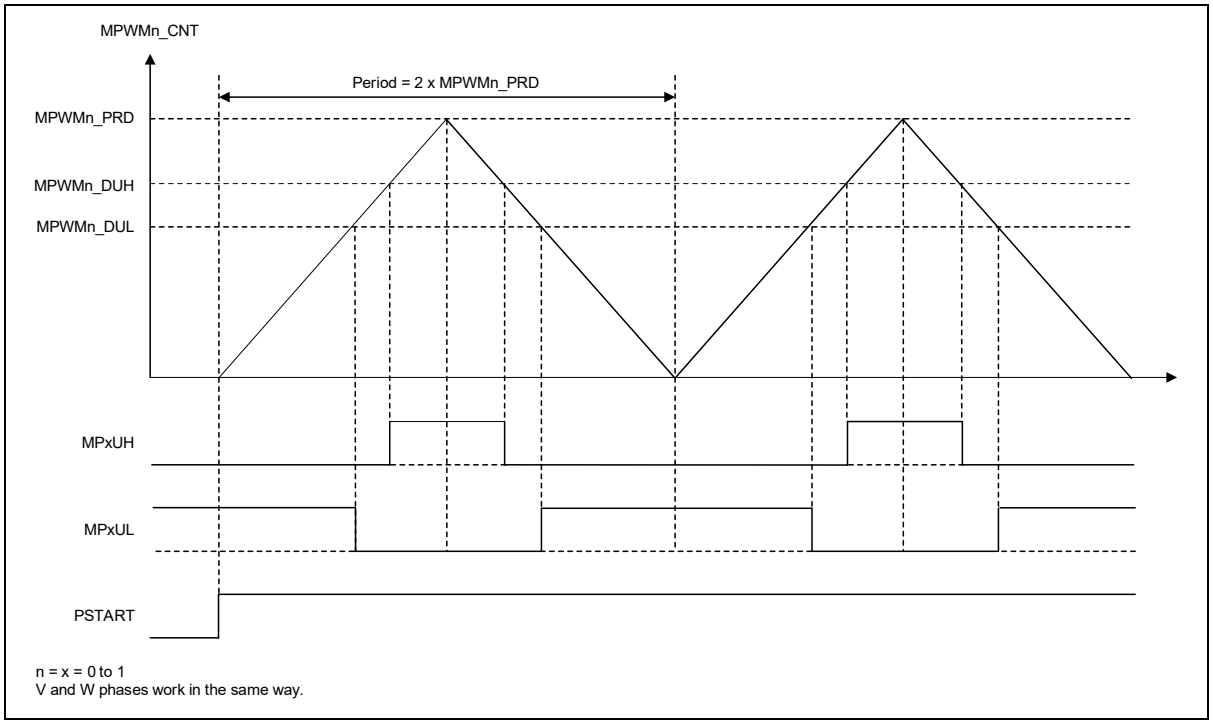
In this mode, the PWM module can generate up to six outputs for the same period.

Once the counter starts operating, it repeats incrementing and decrementing in accordance with the p eriod. The PWM outputs with the high side reversed when the DUTY_xH[15:0] in MPWMn_DxH regist er matches the counter value; while the PWM outputs with the low side reversed when the DUTY_xL[15:0] in MPWMn_DxL register matches the counter value.

Therefore, the high and low sides operate independently and symmetrically around the period matching event.

Figure 179 shows PWM waveform in Motor PWM/up-down counter/2-ch symmetric modes.

Figure 179. PWM Waveform in Motor PWM/Up-Down Counter/2-ch Symmetric Modes



17.3.7.2 Motor PWM Mode 0/Up-Down Counter/1-Ch Asymmetric Mode

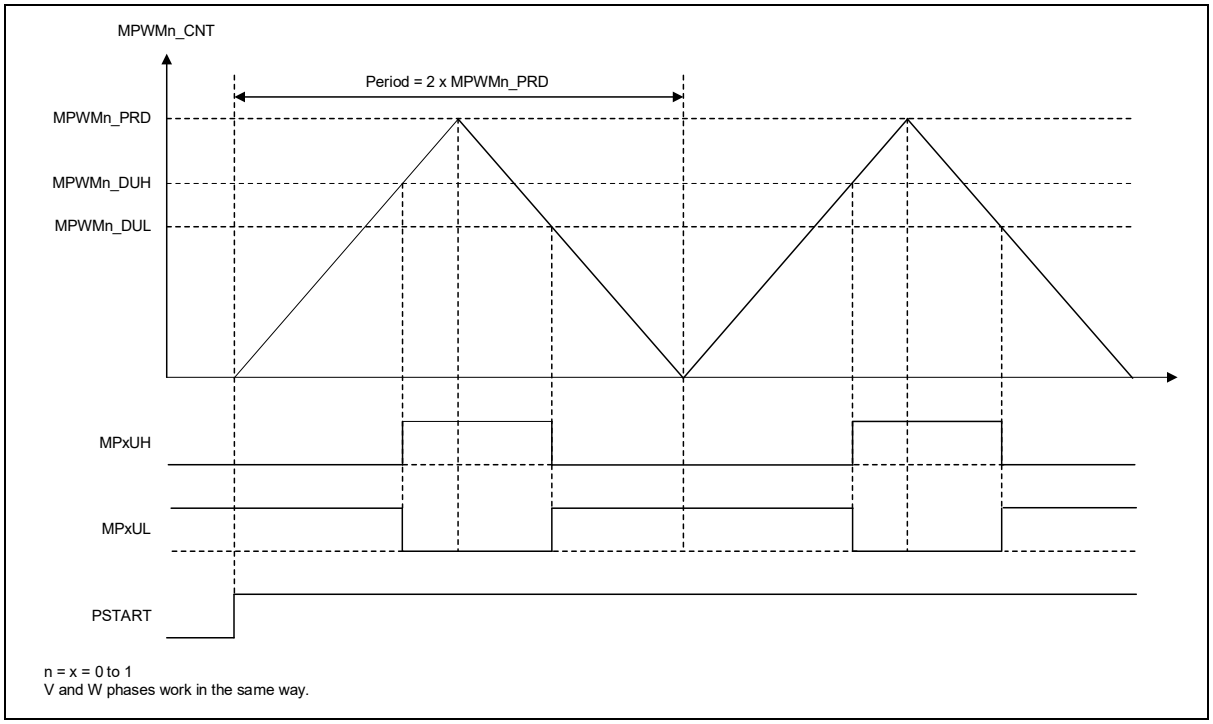
In this mode, the PWM module can generate up to three pairs of PWM outputs that are complementary symmetries for the same period. A pair consists of two channels.

If the counter operates as an up counter, the PWM outputs with the high side reversed when the DUTY_xH[15:0] in MPWMn_DxH register matches the counter value; while if the counter operates as a down counter, the PWM outputs inversion of the output when the DUTY_xL[15:0] in MPWMn_DxL register matches the counter value.

The low side output appears in complementary symmetry of the high side output. Using the characteristics of operating asymmetrically, the PWM phases can be controlled.

Figure 180 shows PWM waveform in Motor PWM/up-down counter/1-ch asymmetric modes.

Figure 180. PWM Waveform in Motor PWM/Up-Down Counter/1-ch Asymmetric Modes



17.3.7.3 Motor PWM Mode 0/Up-Down Counter/1-Ch Symmetric Mode

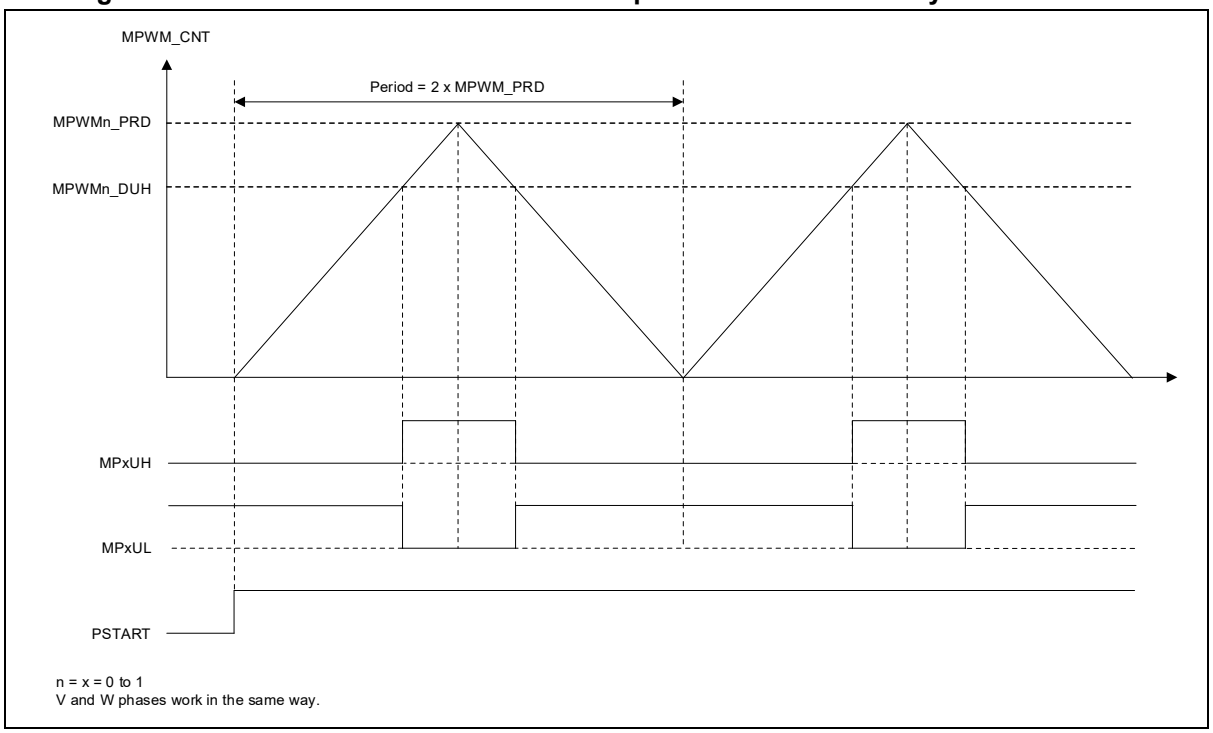
In this mode, the PWM module can generate up to three pairs of PWM outputs that are complementary symmetries for the same period. A pair consists of two channels.

The PWM outputs with the high side reversed when the DUTY_xH[15:0] in MPWM_DxH register matches the counter value; while the low side output appears in complementary symmetry of the high side output.

In this mode, the DUTY_xL[15:0] in MPWM_DxL register do not affect the PWM outputs.

Figure 181 shows PWM waveform in Motor PWM/up-down counter/1-ch symmetric modes.

Figure 181. PWM Waveform in Motor PWM/Up-Down Counter/1-ch Symmetric Modes



17.3.7.4 Motor PWM Mode 1/Up Counter

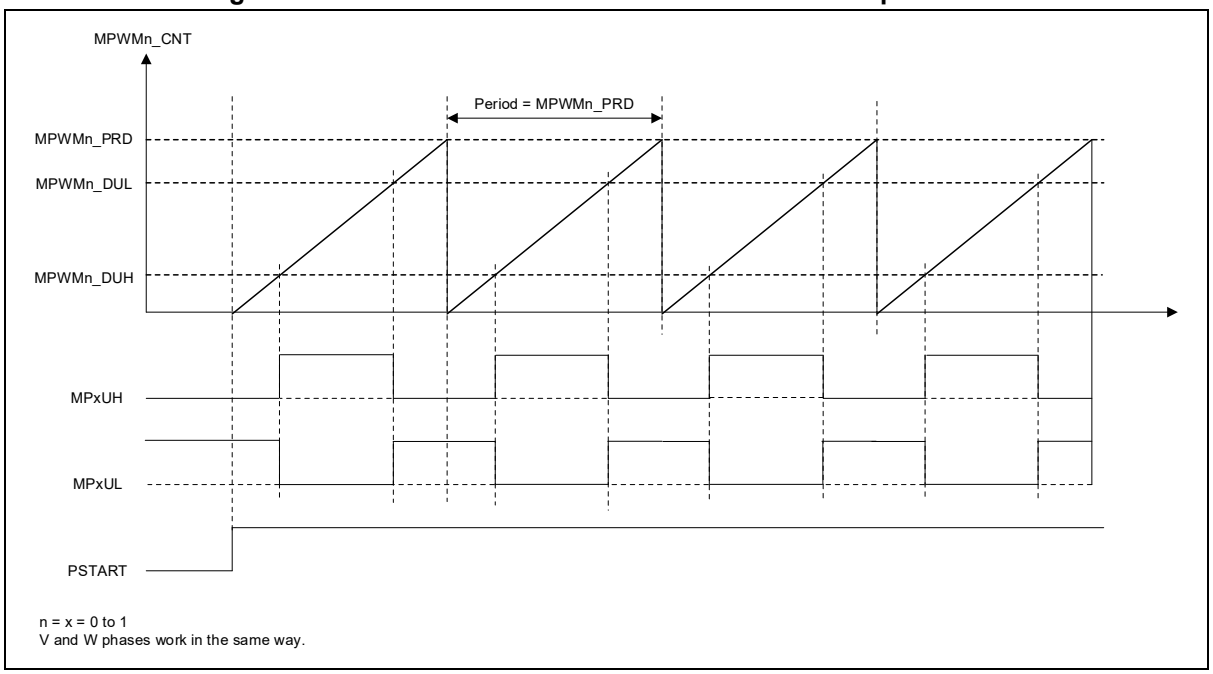
In this mode, the PWM module can generate up to three pairs of PWM outputs that are complementary symmetries for the same period. A pair consists of two channels.

During the up-count period, the registers of DxH and DxL are matched to determine the output of MPnxH. MPnxL is output as the inversion of the MPnxH signal.

In this mode, during the up-count period, MPnxH outputs Low when DxH matches the counter value, and MPnxH outputs High when DxH matches the counter value.

Figure 182 shows PWM waveform in Motor PWM mode 1/up counter.

Figure 182. PWM Waveforms in Motor PWM Mode 1/Up Counter



17.3.7.5 Normal PWM Mode/Up Counter

In this mode, the PWM module can generate up to 6 outputs for the same period.

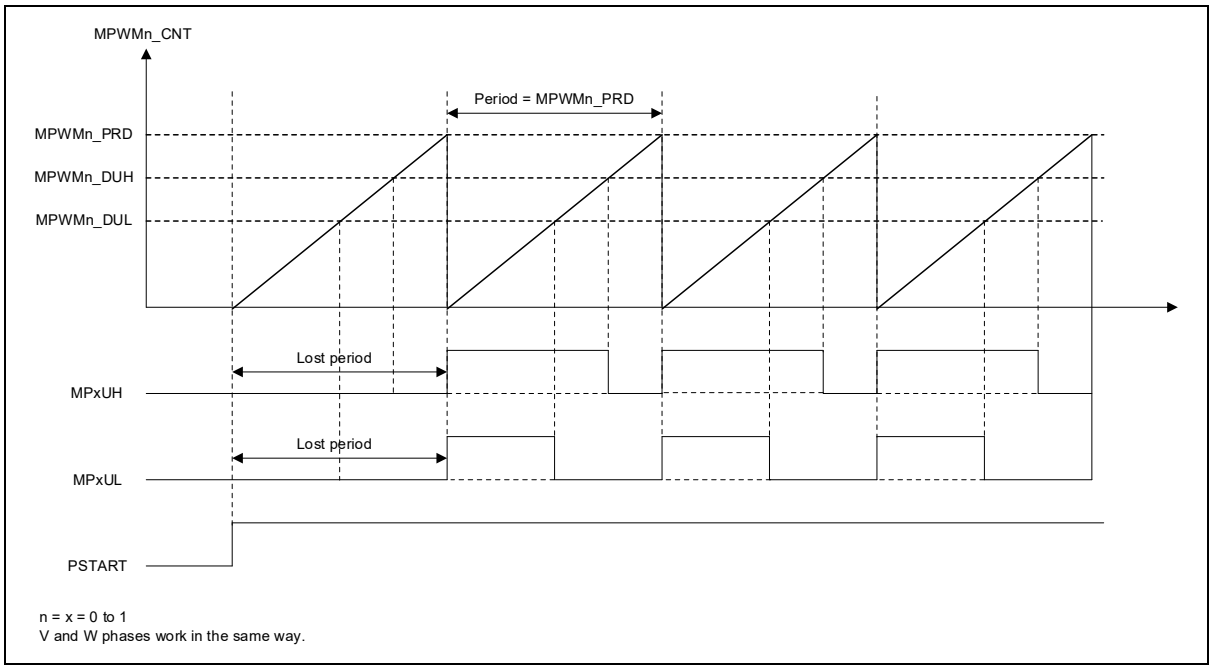
Once the counter starts operating, it repeats incrementing from '0' to 'PERIOD[15:0]' in accordance with the period. The PWM outputs with the high side reversed when the DUTY_xH[15:0] in MPWMn_DxH register matches the counter value; while the PWM outputs with the low side reversed when the DUTY_xL[15:0] in MPWMn_DxL register matches the counter value.

When the counter reaches PERIOD[15:0], both of the high and low sides are reversed at the next clock, and this will be repeated. Like this, the high and low sides of the output operate independently.

Please remember that the high and low side pulses are not output during the first period since the PWM started.

Figure 183 shows PWM waveform in Normal PWM mode/up counter.

Figure 183. PWM Waveforms in Normal PWM Mode/Up-Counter



17.3.8 Complimentary Outputs and Dead-Time Insertion

17.3.8.1 Complimentary Outputs

Circuits such as half-bridge can control devices by activating the high side output and low side output alternately, which is called complementary symmetric output.

When dead-time is off (dead-time is described in the following sections), sum of the duty cycle of the high side output and the duty cycle of the low side output must be 100% (Period time) in the complementary symmetric output method.

$$\text{Duty cycle of high side} + \text{Duty cycle of low side} = 100\ \%$$

17.3.8.2 Dead-Time Insertion

Usually, circuits operating in the complementary symmetric output method require latency which is called dead-time. Dead-time prevents short circuits that can be caused by the switch on/off delay characteristics associated with the output.

Table 168 lists the mode where dead-time can be supported.

Table 168. Dead-Time Support of MPWMn

PWM Mode	Counter Mode	Symmetric/Asymmetric Mode	Dead-Time Support
Motor PWM mode 0	up-down counter	2-ch symmetric mode	N/A
		1-ch asymmetric mode	O
		1-ch symmetric mode	O
Motor PWM mode 1	Up counter	-	O
Normal PWM	Up counter	-	N/A

In Motor PWM mode, dead-time can be enabled by setting the DTEN in MPWMn_DTR register to '1'.

With dead-time enabled, when the high side output or low side output becomes active, dead-time is inserted. The dead-time is determined by configuring the DTCLK[1:0], HDT[7:0] and LDT[7:0] bits of the MPWMn_DTR register, and can be obtained by calculating the formular shown below:

$$\text{Dead} - \text{time} = \frac{\text{HDT}[7:0] \text{ or } \text{LDT}[7:0]}{\text{MPWM clock} \div 2^{(\text{DTCLK}[1:0]+1)}}$$

Figure 184 shows a timing diagram of the dead-time operation in 1-ch symmetric mode set to high active.

Figure 184. Dead-Time Operation Timing Diagram (1-ch Symmetric Mode, HDTOL='0', LDTOL='0')

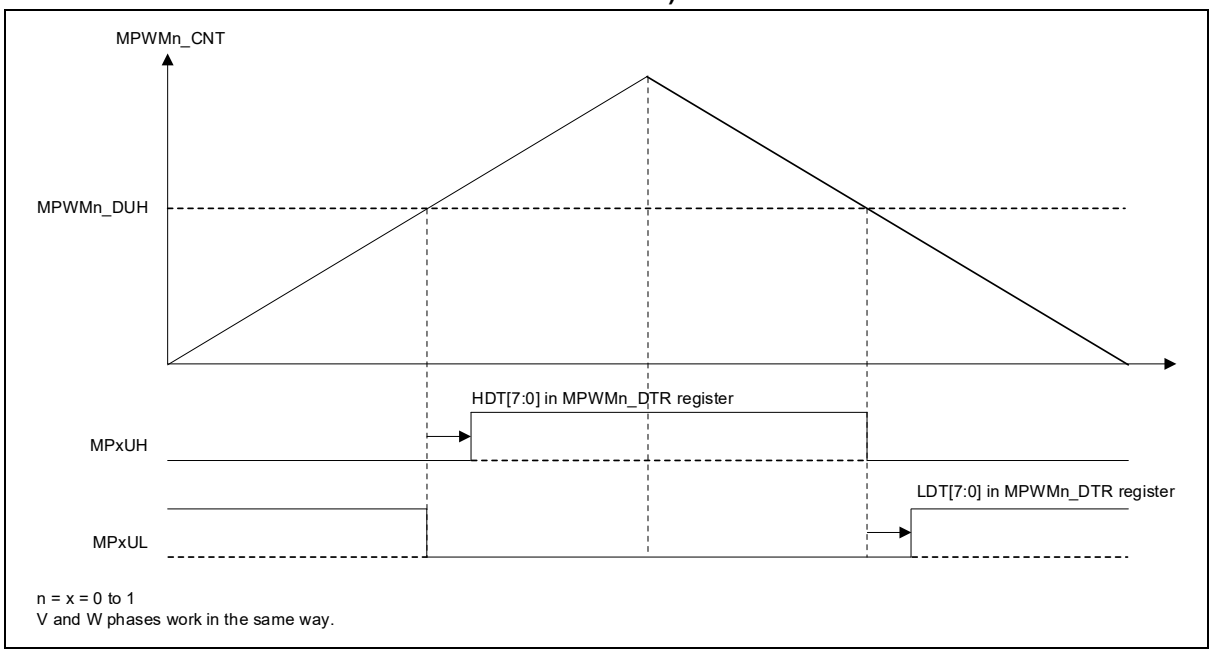
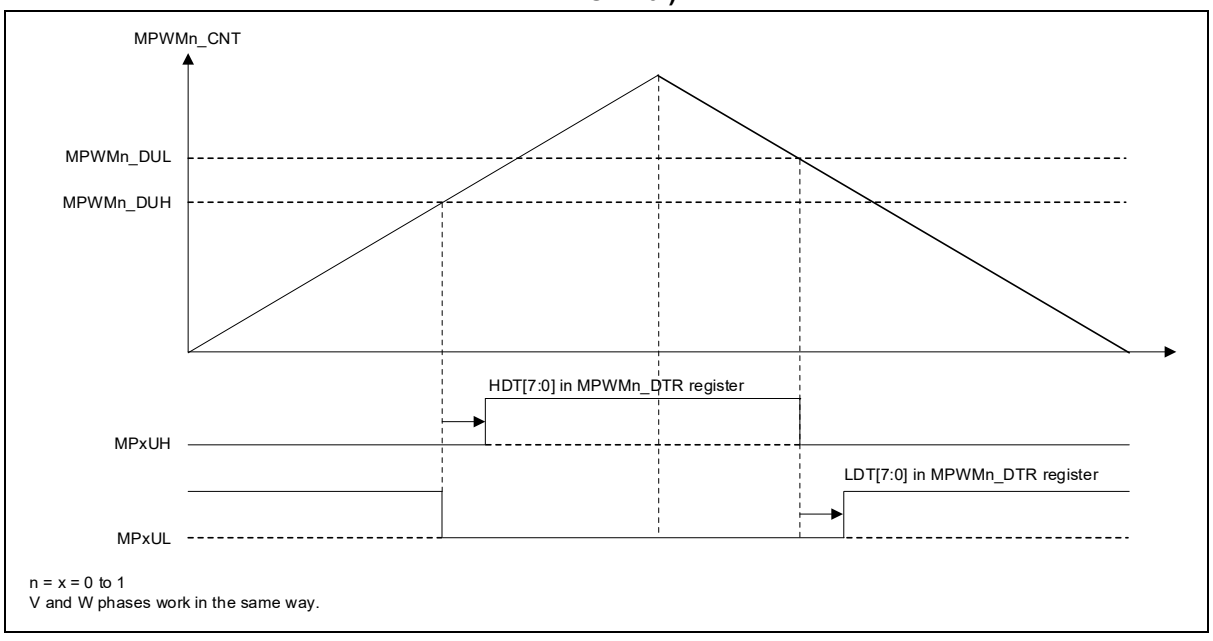


Figure 185 shows a timing diagram of the dead-time operation in 1-ch asymmetric mode set to high active.

Figure 185. Dead-Time Operation Timing Diagram (1-ch Asymmetric Mode, HDTOL='0', LDTOL='0')

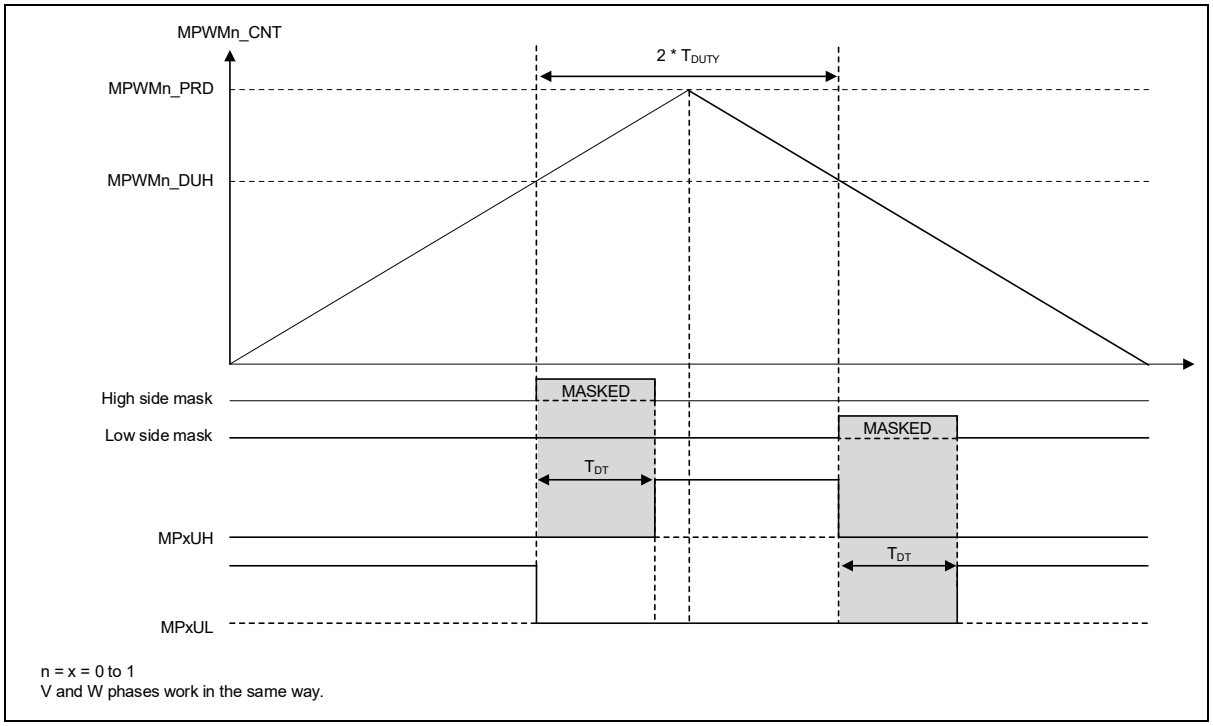


17.3.8.3 Special Case About Dead-Time

Figure 186 shows a timing diagram of the dead-time operations in the typical case where the T_{DUTY} is larger than the T_{DT}.

Dead-time mask is enabled at the Duty Matches and disabled when the dead-time counter reaches the dead-time set by the DTCLK[1:0], HDT[7:0] and LDT[7:0] bits of the MPWM_DTR register.

Figure 186. Typical Dead-Time Operation (T_{DUTY} > T_{DT})



Figures below (Figure 187 to Figure 188) show timing diagrams of dead-time operations in special cases.

Figure 187. High Side Minimum Pulse Timing (T_{DUTY} < T_{DT} < 2 * T_{DUTY})

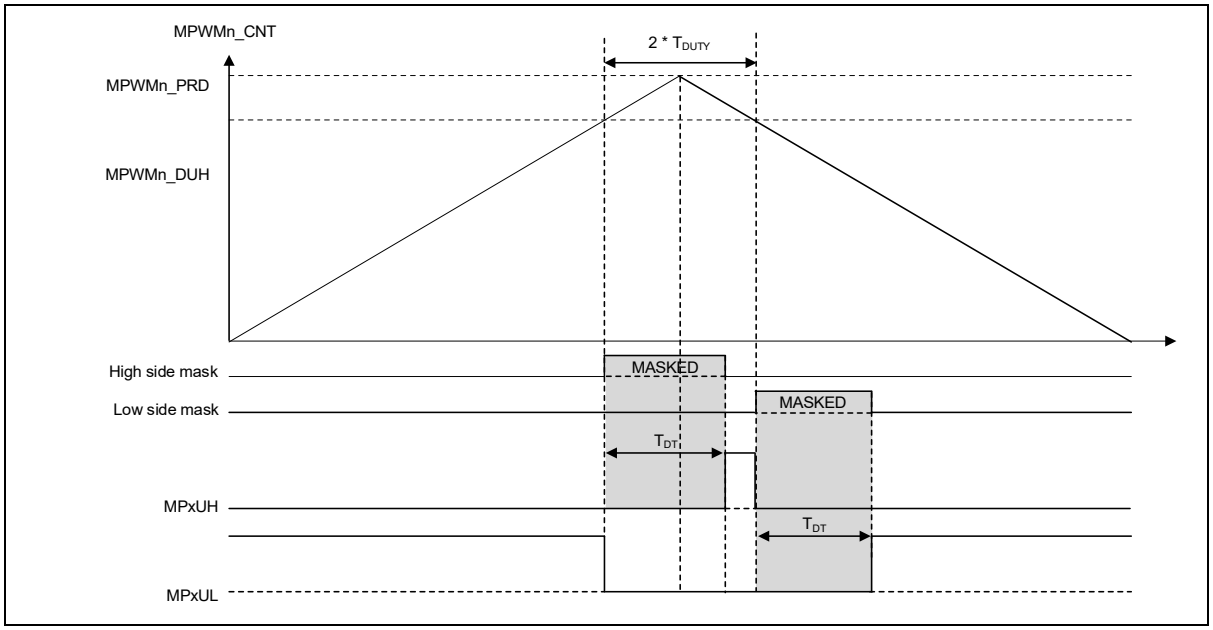


Figure 188. High Side Zero Pulse Timing (T_{DT} > 2 * T_{DUTY})

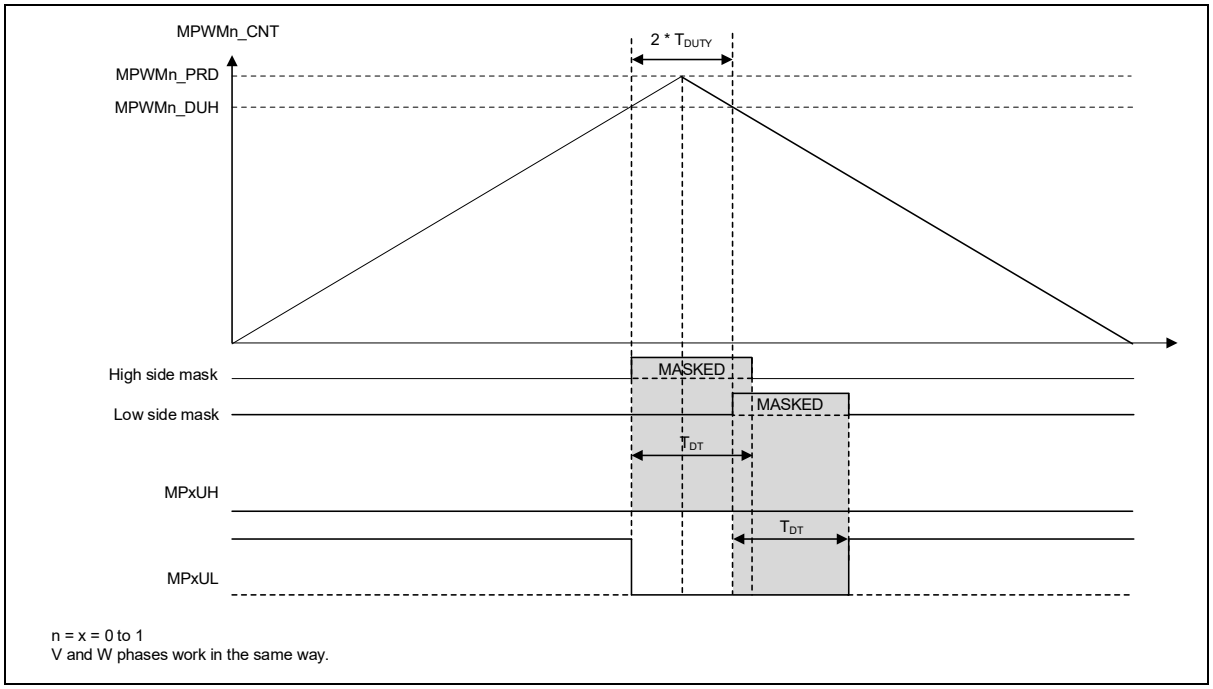


Figure 189. Low Side Minimum Pulse Timing (T_{DT} < period - T_{DUTY})

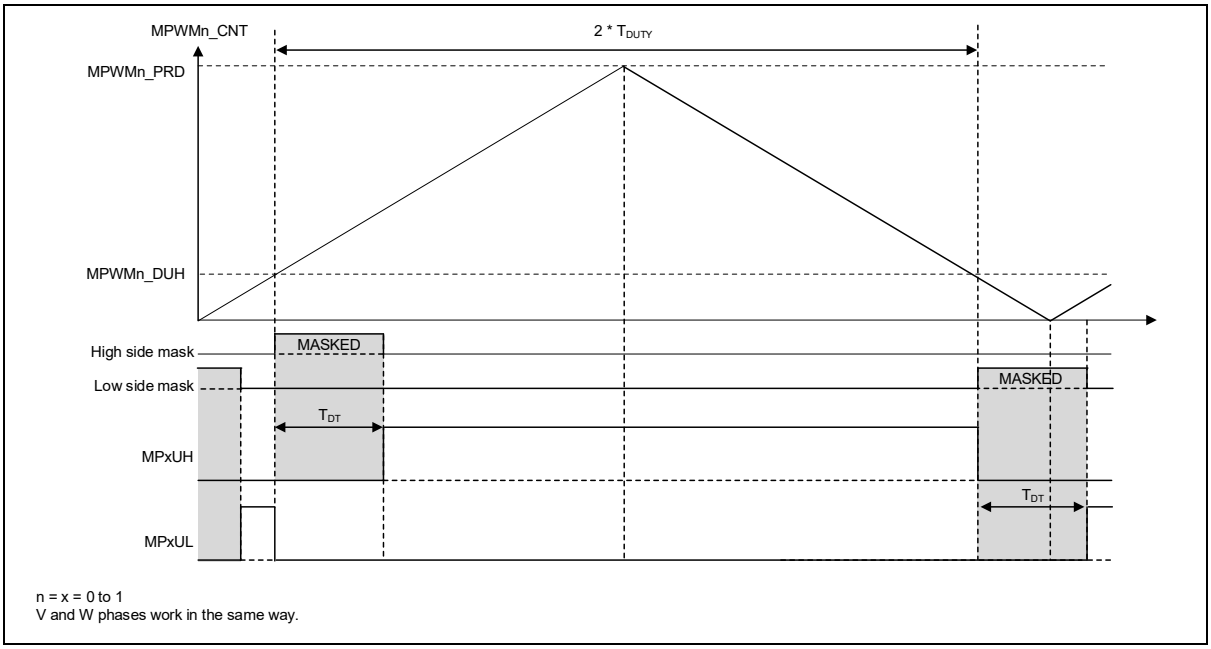


Figure 190. Low Side Zero Pulse Timing (T_{DT} > period - T_{DUTY})

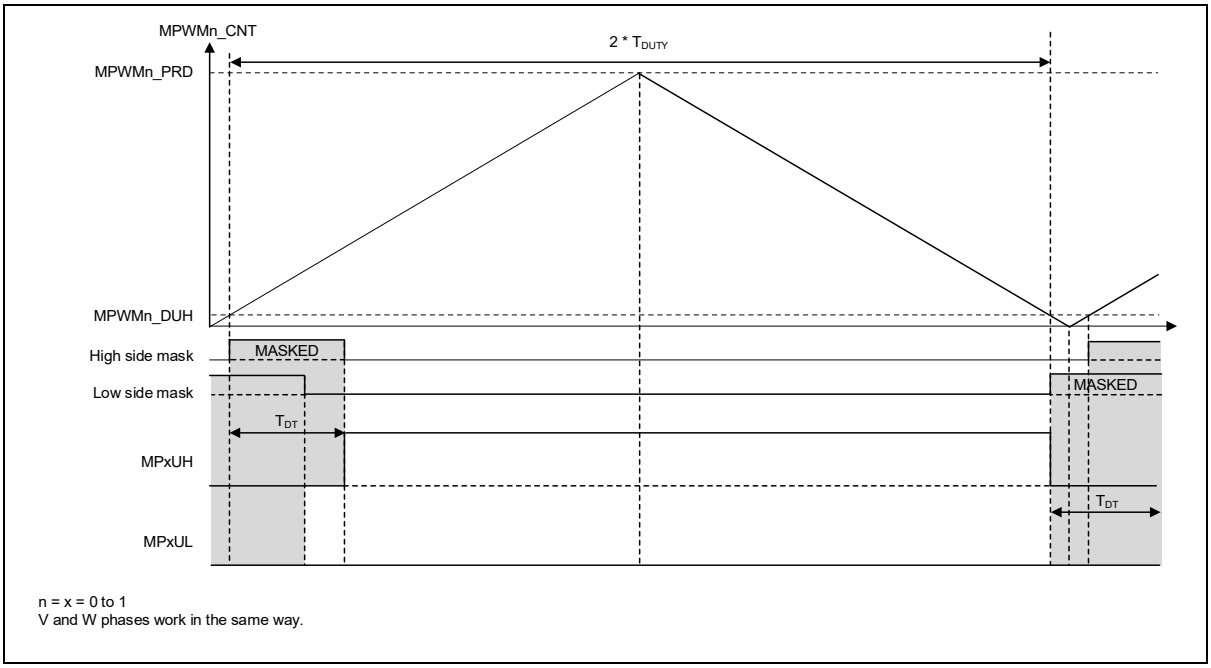


Figure 191. High Side Always On (T_{DUTY} = Period When Dead-Time is Disabled.)

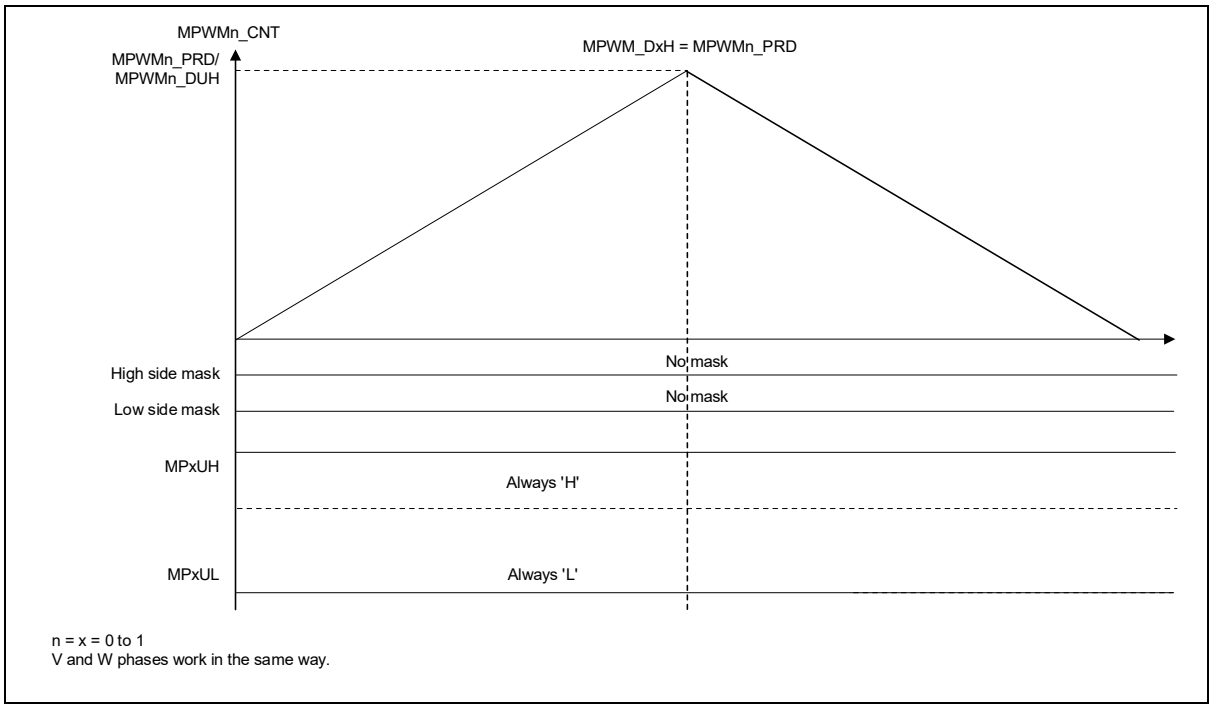
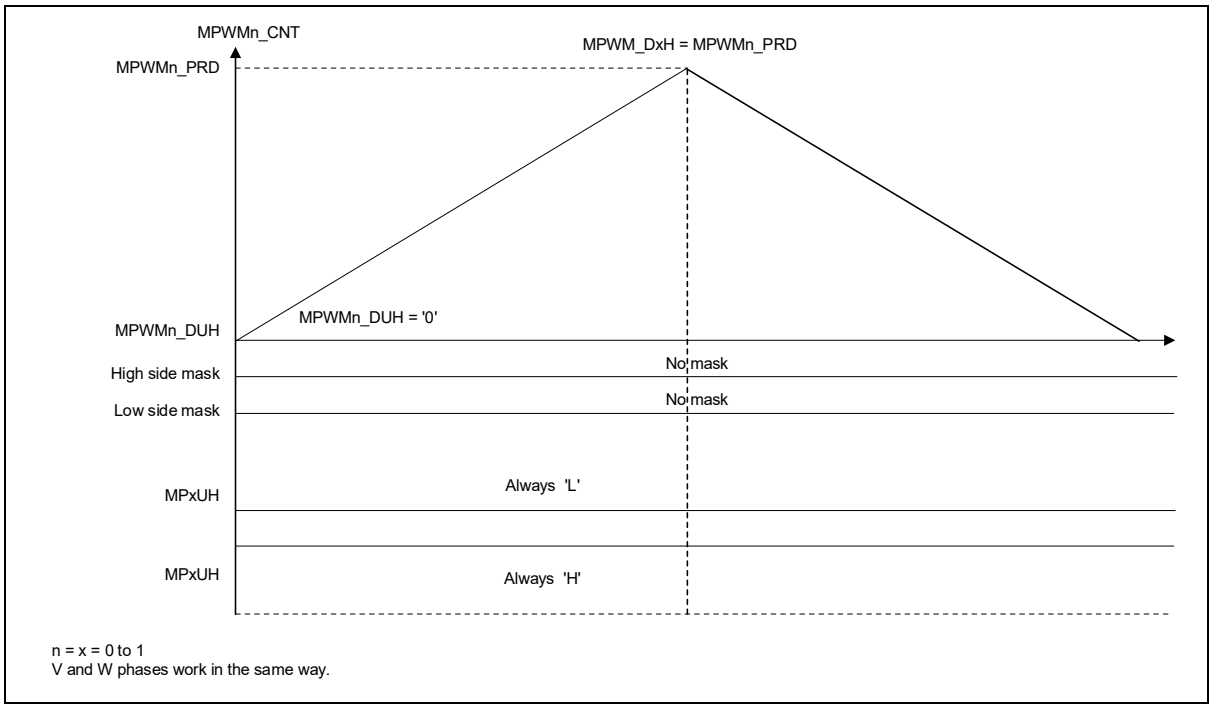


Figure 192. Low Side Always On (T_{DUTY} = 0, When Dead-Time is Disabled.)



17.3.9 Duty/Period Update Timing

Users can determine the time when the duty and period are applied to internal logic by configuring the UAO, TUP, and BUP bits of the MPWM_MR register.

The UAO bit allows the users to select an update method between two shown below:

- If UAO = ‘0’, the updates are performed at preset time.
- If UAO = ‘1’, the updates for the duty and period are performed immediately when requested. With this setting, the duty and period registers are updated after two PWM clock cycles.

The TUP and BUP bits are valid when the UAO is set to ‘0’. The duty and period values are applied to the internal logic according to the TUP and BUP bits when the period matching event or bottom event occurs.

17.3.10 IRQ Interval

In Motor and Normal PWM modes, users can adjust intervals between interrupts by configuring the IRQN[2:0] in MPWMn_CR1 register. Number set in the IRQN[2:0] bits can be applied to the bottom, period matching, and duty matching (or ATRm) interrupts. For detailed information of these interrupts, refer to section 17.3.15. After adjusting the intervals between interrupts, the current interrupt count can be confirmed in the IRQCNT[2:0] in MPWMn_SR register.

Please remember that the IRQCNT[2:0] and UIRQCNT[2:0] are represented in the same bit position.

17.3.11 Direction Bit

In Motor and Normal PWM modes, users can see if the current counter is up counter or down counter by checking the DOWN in MPWMn_SR register.

17.3.12 ADC Synchronization

A certain application may require that the ADC module synchronizes to the PWM before measuring analog signals, and after the measurement, is controlled based on the measured data. To this end, the MPWM module of A34G43x causes a conversion trigger toward the ADC module.

Table 169 describes the trigger sources that can be generated in the MPWM module.

Table 169. Trigger Source by MPWMn

Tigger Source (ADC Module)	Motor/Normal PWM Mode
0	ATR1
1	ATR2
2	ATR3
3	ATR4
4	ATR5
5	ATR6
6	BOTTOM
7	PERIOD

In Motor and Normal PWM modes, the period matching and bottom trigger occur repeatedly in a fixed period, according to the set in the MPWMn_PRD register. However, the triggers due to the values of the MPWMn_ATRx registers (x = 1 to 6) occur in variable periods depending on the ATCNT[15:0] bits.

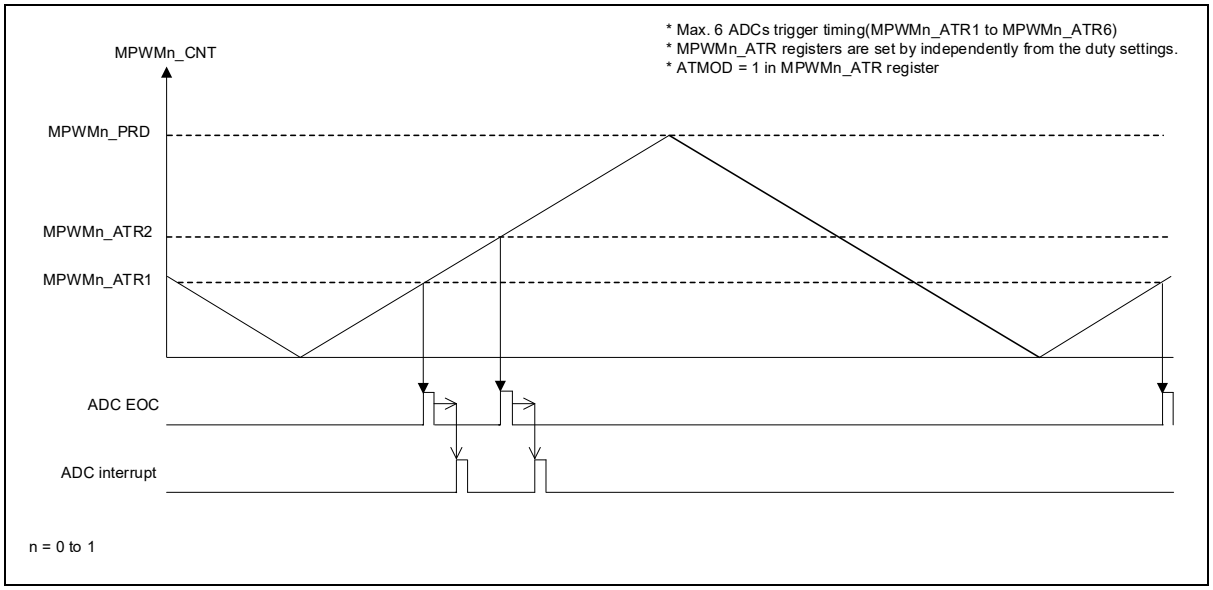
The ATR (ADC trigger counter register) register supports the following functionalities:

- Selection of ADC trigger source counter
- Selection of count match section.
- Selection of internal application and update timing

Users can select a count match section by configuring each ATR register. Configuring the ATMOD[1:0] in MPWMn_ATRx register causes an ADC trigger, if the corresponding counter value matches the ATCNT[15:0] value while the counter is operating as an up counter, a down counter, or an up-down counter. Please remember that the counter causes the ADC trigger a single time per period when it operates as an up counter or a down counter; while the counter causes the ADC trigger two times per period when it operates as an up-down counter.

The MPWMn_ATRx register (x = 1 to 6) is applied to the internal logic at a specific point of time, similar to the period and duty control registers. A certain application may require changing the time when the ADC trigger occurs, in conjunction with PWM duty. To this end, the UAO, TUP, BUP bits of the MPWMn_MR register configure the update timing so that the values set in the MPWMn_ATRx register are applied to the internal logic.

Figure 193. ADC Conversion Timing Diagram by MPWM



17.3.13 Forced Output Mode

A certain application requires setting the PWM outputs to high or low and immediately stopping the external switch operation. The MPWM of the A34G43x provides a function to force each output under external condition or for user purposes.

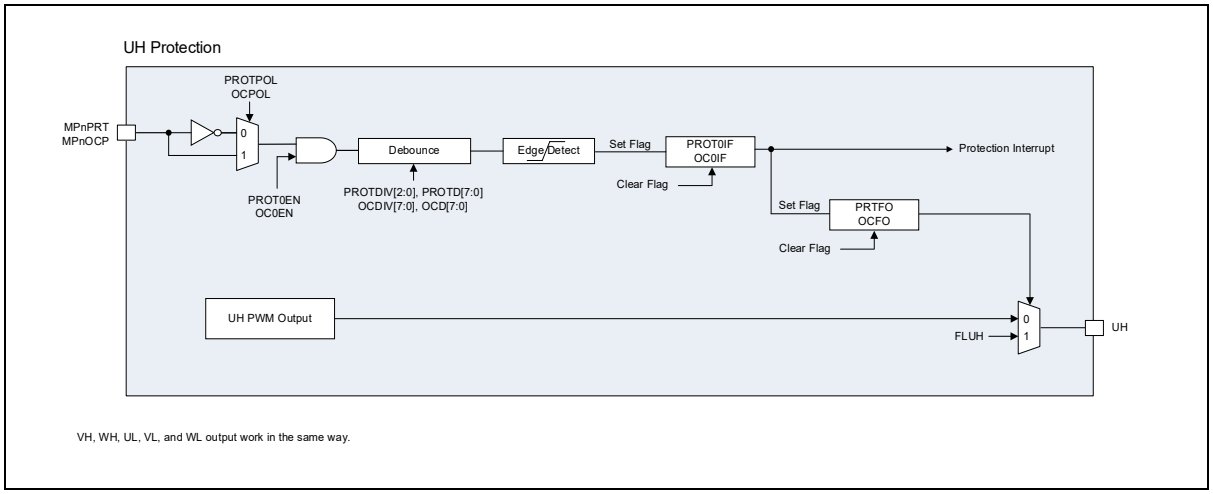
The MPWMn_FOLR register allows to set the polarity of the six forced outputs. Usually, it is set to the off pole of the externally connected switch.

The forced outputs can be caused during the protection operation, and for more information, refer to section 17.3.14.

17.3.14 Protection and Overcurrent

Figure 194 shows a block diagram of protection and overcurrent. When protection or overcurrent event occurs, the outputs can be forcibly set to the preset high or low instead of the PWM outputs.

Figure 194. Protection and Overcurrent Block Diagram



NOTE:

1. When a protection event occurs, all phase outputs of the corresponding MPWM unit are stopped.

The MPWM_PCR and MPWMn_OCR registers support the following protection operations:

- Protection input activation
- Protection input polarity selection
- Protection input debounce setup
- Protection interrupt
- During the protection operation, set up of whether or not to force output for each output

Regarding the protection operation, the MPWMn_FOLR register allows users to enable or disable forced outputs using software. When writing to these registers using software, the FOLKEY [7:0] in MPWMn_FOLR register must be written with ‘0xCA’.

Refer to section 17.3.2 or protection pin information, and section 17.5 for protection related interrupts.

17.3.15 Interrupt Generation

Table 170 describes the interrupts that the MPWM module can cause in Motor PWM mode or Normal PWM mode.

Table 170. Interrupt Source and Event of Motor PWM and Normal PWM Modes

Interrupt Source	Interrupt Event Description
MPWMnPROT	Protection event by protection input pin (MPnPRT)
MPWMnOC	Overcurrent event by overcurrent input pin (MPnOCP)
MPWMn	Bottom
	Period matching
	Duty or ATRx matching

NOTE:

1. 'n' is MPWM unit 0 or 1.

17.4 MPWM Low Power Modes

Table 171. Effect of Low Power Modes on MPWM

Mode	Description
SLEEP	Support
DEEP-SLEEP	Not support

17.5 MPWM Interrupts

Table 172. Interrupt on MPWMn

Interrupt Event	Event Flag	Enable Control Bit	Interrupt Clear Method
Period matching	PRDIF	PRDIE	Write 1 in PRDIF
Bottom	BOTIF	BOTIE	Write 1 in BOTIF
WH duty or ATR6 Matching	DWHIF	WHIE	Write 1 in DWHIF
VH duty or ATR5 Matching	DVHIF	VHIE	Write 1 in DVHIF
UH duty or ATR4 Matching	DUHIF	UHIE	Write 1 in DUHIF
WL duty or ATR3 Matching	DWLIF	WLIE	Write 1 in DWLIF
VL duty or ATR2 Matching	DVLIF	VLIE	Write 1 in DVLIF
UL duty or ATR1 Matching	DULIF	ULIE	Write 1 in DULIF
MPnPRT Input	PROT0IF	PROT0EN	Write 1 in PROT0IF
MPnOCP Input	OC0IF	OC0EN	Write 1 in OC0IF
CMP0 protection	PROT1IF	PROT1EN	Write 1 in PROT1IF
CMP0 overcurrent protection	OC1IF	OC1EN	Write 1 in OC1IF
CMP1 protection	PROT2IF	PROT2EN	Write 1 in PROT2IF
CMP1 overcurrent protection	OC2IF	OC2EN	Write 1 in OC2IF
CMP2 protection	PROT3IF	PROT3EN	Write 1 in PROT3IF
CMP2 overcurrent protection	OC3IF	OC3EN	Write 1 in OC3IF
CMP3 protection	PROT4IF	PROT4EN	Write 1 in PROT4IF
CMP3 overcurrent protection	OC4IF	OC4EN	Write 1 in OC4IF
ADC0 CMPR1 protection	PROT5IF	PROT5EN	Write 1 in PROT5IF
ADC0 CMPR1 overcurrent protection	OC5IF	OC5EN	Write 1 in OC5IF
ADC1 CMPR1 protection	PROT6IF	PROT6EN	Write 1 in PROT6IF
ADC1 CMPR1 overcurrent protection	OC6IF	OC6EN	Write 1 in OC6IF
ADC2 CMPR1 protection	PROT7IF	PROT7EN	Write 1 in PROT7IF
ADC2 CMPR1 overcurrent protection	OC7IF	OC7EN	Write 1 in OC7IF

NOTE:

1. 'n' is MPWM unit 0 or 1.

17.6 MPWM Registers: Motor PWM and Normal PWM Modes

The base address of the MPWM and register map are described in the following:

Table 173. Base Address of MPWM

Name	Base Address
MPWM0	0x4101_4000
MPWM1	0x4101_4100

Table 174. MPWM Register Map

Name	Offset	Type	Description	Reset Value	Reference
MPWMn_MR	0x0000	RW	MPWM n mode register	0x0000_0000	17.6.1
MPWMn_OLR	0x0004	RW	MPWM n output level register	0x0000_0000	17.6.2
MPWMn_FOLR	0x0008	RW	MPWM n forced output register	0x0000_0000	17.6.3
MPWMn_PRD	0x000C	RW	MPWM n PWM period register	0x0000_0000	17.6.4
MPWMn_DUH	0x0010	RW	MPWM n duty UH register	0x0000_0000	17.6.5
MPWMn_DVH	0x0014	RW	MPWM n duty VH register	0x0000_0000	17.6.6
MPWMn_DWH	0x0018	RW	MPWM n duty WH register	0x0000_0000	17.6.7
MPWMn_DUL	0x001C	RW	MPWM n duty UL register	0x0000_0000	17.6.8
MPWMn_DVL	0x0020	RW	MPWM n duty VL register	0x0000_0000	17.6.9
MPWMn_DWL	0x0024	RW	MPWM n duty WL register	0x0000_0000	17.6.10
MPWMn_CR1	0x0028	RW	MPWM n control register 1	0x0000_0000	17.6.11
MPWMn_CR2	0x002C	RW	MPWM n control register 2	0x0000_0000	17.6.12
MPWMn_SR	0x0030	RW	MPWM n status register	0x0000_0000	17.6.13
MPWMn_IER	0x0034	RW	MPWM n interrupt enable register	0x0000_0000	17.6.14
MPWMn_CNT	0x0038	RO	MPWM n counter register	0x0000_0000	17.6.15
MPWMn_DTR	0x003C	RW	MPWM n dead-time register	0x0000_0000	17.6.16
MPWMn_PCR	0x0040	RW	MPWM n protection register	0x0000_0000	17.6.17
MPWMn_PSR	0x0044	RW	MPWM n protection status register	0x0000_0000	17.6.18
MPWMn_OCR	0x0048	RW	MPWM n overcurrent detection register	0x0000_0000	17.6.19
MPWMn_OSR	0x004C	RW	MPWM n overcurrent detection status register	0x0000_0000	17.6.20
MPWMn_ATR1	0x0058	RW	MPWM n ADC trigger 1 register	0x0000_0000	17.6.21
MPWMn_ATR2	0x005C	RW	MPWM n ADC trigger 2 register	0x0000_0000	17.6.21
MPWMn_ATR3	0x0060	RW	MPWM n ADC trigger 3 register	0x0000_0000	17.6.21
MPWMn_ATR4	0x0064	RW	MPWM n ADC trigger 4 register	0x0000_0000	17.6.21
MPWMn_ATR5	0x0068	RW	MPWM n ADC trigger 5 register	0x0000_0000	17.6.21
MPWMn_ATR6	0x006C	RW	MPWM n ADC trigger 6 register	0x0000_0000	17.6.21

NOTE:

1. n = 0 and 1

17.6.1 MPWMn_MR: MPWM n Mode Register

The MPWMn_MR register is used to set the MPWM driving method.

After the initial PWM period and duty setting is completed, the UAO bit must be set once to update internal operating registers with these set values. This helps transfer the setup data from the user interface register to internal operating registers. The settings of the UAO bit must remain unchanged for at least two PWM clock cycles; otherwise, the update command may get lost, resulting in retaining old data in the internal registers.

The MCHMOD[1:0] value of this register is valid only when the MOTORB[1:0] value is '0'; otherwise, the MCHMOD[1:0] value is internally ignored and these bits field remains '00'

MPWM0_MR=0x4101_4000, MPWM1_MR=0x4101_5000

31 30 29 28 27 26 25 24	23 22 21 20	19 18 17 16	15 14	13 12	11	10	9	8	7	6	5	4	3	2 1	0
Reserved		CLKDIV[3:0]	MOTORB[1:0]	Reserved	UOL	Reserved	TUOL	BUOL	UAO	Reserved	TUP	BUP	Reserved	MCHMOD[1:0]	Reserved
-		0000	00	-	0	-	0	0	0	-	0	0	-	00	-
-		RW	RW	-	RW	-	RW	RW	RW	-	RW	RW	-	RW	-

19 16	CLKDIV[3:0]	MPWM Clock Divider setting bit MPWM Clock = HCLK / (CLKDIV + 1)
15 14	MOTORB[1:0]	MPWM mode selection bit
		00 Motor PWM mode0 (up-down count mode)
		01 Motor PWM mode1 (up-count mode)
		10 Normal PWM mode (up-count mode)
11	UOL	Output level update timing setting bit
		0 Updates at specified timings.
		1 Updates all output levels at once.
9	TUOL	Output level update setting bit (when same period)
		0 MPWM_OLR register is not updated at every period-match.
		1 MPWM_OLR register is updated at every period-match.
8	BUOL	Output level update setting bit (when bottom)
		0 MPWM_OLR register is not updated at every bottom-match.
		1 MPWM_OLR register is updated at every bottom-match.
7	UAO	Update timing setting bit
		0 Updates at specified timings.
		1 Updates all duties and periods at once.
5	TUP	DUTY, Period Update Enable bit (when same period)
		0 Period and Duty value are not updated at every period-match.
		1 Period and Duty value are updated at every period-match.
4	BUP	DUTY, Period Update Enable bit (when same bottom)
		0 Period and Duty value are not updated at every bottom-match.
		1 Period and Duty value are updated at every bottom-match.
2 1	MCHMOD[1:0]	Channel symmetry/asymmetry mode selection bit (In motor PWM mode 1 and general PWM mode, it operates regardless of the settings.)
		00 Two-channel symmetric mode Duty H determines the H channel signal's timing of switching between the high and low levels.

		Duty L determines the L channel signal's timing of switching between the high and low levels.
	01	One-channel asymmetric mode (L channel and H channel are inverted.) Duty H determines the H channel signal's timing of switching to the high level. Duty L determines the L channel signal's timing of switching to the low level.
	10	One-channel symmetric mode (L channel and H channel are inverted.) Duty H determines the H channel signal's timing of switching between the high and low levels.
	11	Reserved

17.6.2 MPWMn_OLR: MPWM n Output Level Register

The MPWMn_OLR register controls the level of each PWM output port.

MPWM0_OLR=0x4101_4004, MPWM1_OLR=0x4101_500

31 30 29 28 27 26 25 24	23 22 21 20 19 18 17 16								15 14 13 12 11 10 9 8							7 6 5 4 3 2 1 0						
Reserved	FORCEU	Reserved	POCWH	POCVH	POCUH	POCWL	POCVL	POCUL	Reserved	DOLWH	DOLVH	DOLUH	DOLWL	DOLVL	DOLUL	Reserved	SLWH	SLVH	SLUH	SLWL	SLVL	SLUL
-	0	-	0	0	0	0	0	0	-	0	0	0	0	0	0	-	0	0	0	0	0	0
-	RW	-	RW	RW	RW	RW	RW	RW	-	RW	RW	RW	RW	RW	RW	-	RW	RW	RW	RW	RW	RW

24	FORCEU	Upper output settings
	0	6-channel mode
	1	Force U-Channel Mode (All outputs of MPWMxH/MPWMxL are controlled according to the MPWM_DUH/DUL registers.)
21	POCWH	WH PWM output control
	0	Disables.
	1	Enables (PWM Output).
20	POCVH	VH PWM output control
	0	Disables.
	1	Enables (PWM Output).
19	POCUH	UH PWM output control
	0	Disables.
	1	Enables (PWM Output).
18	POCWL	WL PWM output control
	0	Disables.
	1	Enables (PWM Output).
17	POCVL	VL PWM output control
	0	Disables.
	1	Enables (PWM Output).
16	POCUL	UL PWM output control
	0	Disables.
	1	Enables (PWM Output).
13	DOLWH	Output level selection bit when disabling WH PWM output
	0	Low level
	1	High level
12	DOLVH	Output level selection bit when disabling VH PWM output
	0	Low level
	1	High level
11	DOLUH	Output level selection bit when disabling UH PWM output
	0	Low level
	1	High level
10	DOLWL	Output level selection bit when disabling WL PWM output
	0	Low level
	1	High level
9	DOLVL	Output level selection bit when disabling VL PWM output
	0	Low level
	1	High level
8	DOLUL	Output level selection bit when disabling UL PWM output
	0	Low level

		1	High level
5	SLWH	Output selection bit when starting WH PWM	
		0	Basic output level (Low Level)
		1	Inverted output level (High Level)
4	SLVH	Output selection bit when starting VH PWM	
		0	Basic output level (Low Level)
		1	Inverted output level (High Level)
3	SLUH	Output selection bit when starting UH PWM	
		0	Basic output level (Low Level)
		1	Inverted output level (High Level)
2	SLWL	Output selection bit when starting WL PWM	
		0	Basic output level (Low Level)
		1	Inverted output level (High Level)
1	SLVL	Output selection bit when starting VL PWM	
		0	Basic output level (Low Level)
		1	Inverted output level (High Level)
0	SLUL	Output selection bit when starting UL PWM	
		0	Basic output level (Low Level)
		1	Inverted output level (High Level)

NOTE:

1. Refer to Table 166 for the basic output levels in each operating mode. DOLxH, DOLxL refers to the output level when the PWM output has been stopped.

17.6.3 MPWMn_FOLR: MPWM n Forced Output Register

A specific PWM output level can be forcibly generated by an abnormal event triggered by an external condition or the user's intentional manipulation. Once a forcing condition is met, each PWM channel generates an output signal in the level set in the MPWMn_FOLR register.

MPWM0_FOLR=0x4101_4008, MPWM1_FOLR=0x4101_5008

31 30 29 28 27 26 25 24	23 22 21 20 19 18 17 16	15 14 13 12 11	10	9	8	7 6	5	4	3	2	1	0
Reserved	FOLKEY[7:0]	Reserved	SWFO	PRTFO	OCFO	Reserved	FLWH	FLVH	FLUH	FLWL	FLVL	FLUL
-	-	-	0	0	0	-	0	0	0	0	0	0
-	WO	-	RW	RWC1	RWC1	-	RW	RW	RW	RW	RW	RW

23 16	FOLKEY[7:0]	Forced output level register access key (FOLR key: 0xCA) To write register value, the register value and the key value must be written together. If the access key value does not match, the writings are ignored.
10	SWFO	Software forced output settings
	0	Outputs PWM.
	1	Outputs forced level instead of PWM.
9	PRTFO	Protection forced output settings (This bit is automatically set to "1" when Protection Event occurs. Writing "1" clears the flag)
	0	Outputs PWM.
	1	Outputs forced level instead of PWM.
8	OCFO	Over-current forced output settings (This bit is automatically set to "1" when Over-current Event occurs. Writing "1" clears the flag)
	0	Outputs PWM.
	1	Outputs forced level instead of PWM.
5	FLWH	WH output forced level selection
	0	Output forced level: "L"
	1	Output forced level: "H"
4	FLVH	VH output forced level selection
	0	Output forced level: "L"
	1	Output forced level: "H"
3	FLUH	UH output forced level selection
	0	Output forced level: "L"
	1	Output forced level: "H"
2	FLWL	WL output forced level selection
	0	Output forced level: "L"
	1	Output forced level: "H"
1	FLVL	VL output forced level selection
	0	Output forced level: "L"
	1	Output forced level: "H"
0	FLUL	UL output forced level selection
	0	Output forced level: "L"
	1	Output forced level: "H"

17.6.4 MPWMn_PRD: MPWM n Period Register

The MPWMn_PRD register sets the PWM period.

MPWM0_PRD=0x4101_400C, MPWM1_PRD=0x4101_500C

31 30 29 28 27 26 25 24	23 22 21 20 19 18 17 16	15 14 13 12 11 10 9 8	7 6 5 4 3 2 1 0
Reserved		PERIOD[15:0]	
-		0x0000	
-		RW	

15 0	PERIOD[15:0]	16-bit PWM period of phase U, V, and W The bits value must be larger than 0x0010.

17.6.5 MPWMn_DUH: MPWM n Duty UH Register

The MPWMn_DUH register sets the PWM duty of U-phase. The PWM duty of U-phase is determined by the MPWMn_DUH and MPWM_DUL registers according to the MCHMOD[1:0] bit settings.

MPWM0_DUH=0x4101_4010, MPWM1_DUH=0x4101_5010

31 30 29 28 27 26 25 24	23 22 21 20 19 18 17 16	15 14 13 12 11 10 9 8	7 6 5 4 3 2 1 0
Reserved		DUTY_UH[15:0]	
-		0x0000	
-		RW	

15 0	DUTY_UH[15:0]	16-bit PWM duty for UH output (Duty can be set to 0.)

NOTE:

1. Setting a full duty generates a “low” output, while setting zero-duty generates a “high” output.

17.6.6 MPWMn_DVH: MPWM n Duty VH Register

The MPWMn_DVH register sets the PWM duty of V-phase. The PWM duty of V-phase is determined by the MPWMn_DVH and MPWMn_DVL registers according to the MCHMOD[1:0] bit settings.

MPWM0_DVH=0x4101_4014, MPWM1_DVH=0x4101_5014

31 30 29 28 27 26 25 24	23 22 21 20 19 18 17 16	15 14 13 12 11 10 9 8	7 6 5 4 3 2 1 0
Reserved		DUTY_VH[15:0]	
-		0x0000	
-		RW	

15 0	DUTY_VH[15:0]	16-bit PWM duty for VH output (Duty can be set to 0.)

NOTE:

1. Setting a full duty generates a “low” output, while setting zero-duty generates a “high” output.

17.6.7 MPWMn_DWH: MPWM n Duty WH Register

The MPWMn_DWH register sets the PWM duty of W-phase. The PWM duty of W-phase is determined by the MPWMn_DWH and MPWMn_DWL registers according to the MCHMOD[1:0] bit settings.

MPWM0_DWH=0x4101_4018, MPWM1_DWH=0x4101_5018

31 30 29 28 27 26 25 24	23 22 21 20 19 18 17 16	15 14 13 12 11 10 9 8	7 6 5 4 3 2 1 0
Reserved		DUTY_WH[15:0]	
-		0x0000	
-		RW	

15 0	DUTY_WH[15:0]	16-bit PWM duty for WH output (Duty can be set to 0.)

NOTE:

1. Setting a full duty generates a “low” output, while setting zero-duty generates a “high” output.

17.6.8 MPWMn_DUL: MPWM n Duty UL Register

The MPWMn_DUL register sets the PWM duty of U-phase.

MPWM0_DUL=0x4101_401C, MPWM1_DUL=0x4101_501C

31 30 29 28 27 26 25 24	23 22 21 20 19 18 17 16	15 14 13 12 11 10 9 8	7 6 5 4 3 2 1 0
Reserved		DUTY_UL[15:0]	
-		0x0000	
-		RW	

15 0	DUTY_UL[15:0]	16-bit PWM duty for UL output (Duty can be set to 0.)

NOTE:

1. Setting a full duty generates a “low” output, while setting zero-duty generates a “high” output.

17.6.9 MPWMn_DVL: MPWM n Duty VL Register

The MPWMn_DVL register sets the PWM duty of V-phase.

MPWM0_DVL=0x4101_4020, MPWM1_DVL=0x4101_5020

31 30 29 28 27 26 25 24	23 22 21 20 19 18 17 16	15 14 13 12 11 10 9 8	7 6 5 4 3 2 1 0
Reserved		DUTY_VL[15:0]	
-		0x0000	
-		RW	

15 0	DUTY_VL[15:0]	16-bit PWM duty for VL output (Duty can be set to 0.)

NOTE:

1. Setting a full duty generates a “low” output, while setting zero-duty generates a “high” output.

17.6.10 MPWMn_DWL: MPWM n Duty WL Register

The MPWMn_DWL register sets the PWM duty of W-phase.

MPWM0_DWL=0x4101_4024, MPWM1_DWL=0x4101_5024

31 30 29 28 27 26 25 24	23 22 21 20 19 18 17 16	15 14 13 12 11 10 9 8	7 6 5 4 3 2 1 0
Reserved		DUTY_WL[15:0]	
-		0x0000	
-		RW	

15 0	DUTY_VL[15:0]	16-bit PWM duty for VL output (Duty can be set to 0.)

NOTE:

1. Setting a full duty generates a “low” output, while setting zero-duty generates a “high” output.

17.6.11 MPWMn_CR1: MPWM n Control Register 1

The MPWMn_CR1 register sets the MPWM interrupt period and enables the MPWM.

MPWM0_CR1=0x4101_4028, MPWM1_CR1=0x4101_5028

31 30 29 28 27 26 25 24	23 22 21 20 19 18 17 16	15 14 13 12 11	10 9 8	7 6 5 4 3 2 1	0
Reserved			IRQN[2:0]	Reserved	PWMEN
-			000	-	0
-			RW	-	RW

10 8	IRQN[2:0](1)	Interrupt request (IRQ) interval number PRDIRQ, BOTIRQ, and ATRx are made at the specified intervals.(1)
0	PWMEN	PWM Enable If this bit is set to ‘0’, the PWM block remains in the reset status, but the user interface is accessible. This bit must be set to ‘1’ to operate the PWM block.
		0 Disables
		1 Enables

NOTE:

1. By default, PRDIRQ and BOTIRQ are made at every period. However, interrupt intervals can be set from 1 to 8 periods. If IRQN[2:0] = 0, IRQ is made at every period. Otherwise, they are made at every “IRQN[2:0] + 1” periods.

17.6.12 MPWMn_CR2: MPWM n Control Register 2

The MPWMn_CR2 register controls MPWM operations such as start, stop, and halt.

MPWM0_CR2=0x4101_402C, MPWM1_CR2=0x4101_502C

31 30 29 28 27 26 25 24	23 22 21 20 19 18 17 16	15 14 13 12 11 10 9 8	7	6 5 4 3 2	1	0
Reserved			HALT	Reserved		PSTART
-			0	-		0
-			RW	-		RW

7	HALT	Setting PWM output halting which stops the PWM counter but does not reset it. When HALT = '1', the PWM output retains its state when a halt is made.	
		0	Disable PWM output halt
		1	Enable PWM output halt
0	PSTART(1)	PWM counter start or stop/clear	
		1	Starts the PWM counter (Re-initialized at the second PWM clock period).
		0	Stops and clears the PWM counter.

NOTE:

1. Before setting PSTART, PWMEN must be set to '1' to start the PWM counter.

17.6.13 MPWMn_SR: MPWM n Status Register

The MPWMn_SR register shows MPWM status.

MPWM0_SR=0x4101_4030, MPWM1_SR=0x4101_5030

31 30 29 28 27 26 25 24	23 22 21 20 19 18 17 16	15	14 13 12	11 10 9 8	7	6	5	4	3	2	1	0
Reserved		DOWN	IRQCNT[2:0]	Reserved	PRDIF	BOTIF	DWHIF/ATR6	DVHIF/ATR5	DUHIF/ATR4	DWLIF/ATR3	DVLIF/ATR2F	DULIF/ATR1F
-		0	000	-	0	0	0	0	0	0	0	0
-		RO	RO	-	RWC1	RWC1	RWC1	RWC1	RWC1	RWC1	RWC1	RWC1

15	DOWN	Current PWM count mode
		0 Current count mode = Up
		1 Current count mode = Down
14 12	IRQCNT[2:0]	Interrupt count number
7	PRDIF	PWM period interrupt flag (Writing “1” clears the flag.)
		0 Interrupt is not generated.
		1 Interrupt is generated.
6	BOTIF	PWM Bottom interrupt flag (Writing “1” clears the flag.)
		0 Interrupt is not generated.
		1 Interrupt is generated.
5	WHIF ATR6IF	PWM Duty WH interrupt flag Disables ATR6: Duty interrupt flag Enables ATR6: ATR6 interrupt flag (Writing “1” clears the flag.)
		0 Interrupt is not generated.
		1 Interrupt is generated.
4	VHIF ATR5IF	PWM Duty VH interrupt flag Disables ATR5: Duty interrupt flag Enables ATR5: ATR5 interrupt flag (Writing “1” clears the flag.)
		0 Interrupt is not generated.
		1 Interrupt is generated.
3	UHIF ATR4IF	PWM Duty UH interrupt flag Disables ATR4: Duty interrupt flag Enables ATR4: ATR4 interrupt flag (Writing “1” clears the flag.)
		0 Interrupt is not generated.
		1 Interrupt is generated.
2	WLIF ATR3IF	PWM Duty WL interrupt flag Disables ATR3: Duty interrupt flag Enables ATR3: ATR3 interrupt flag (Writing “1” clears the flag.)
		0 Interrupt is not generated.
		1 Interrupt is generated.
1	VLIF ATR2IF	PWM Duty VL interrupt flag Disables ATR2: Duty interrupt flag Enables ATR2: ATR2 interrupt flag (Writing “1” clears the flag.)
		0 Interrupt is not generated.
		1 Interrupt is generated.
0	ULIF ATR1IF	PWM Duty UL interrupt flag Disables ATR1: Duty interrupt flag

	Enables ATR1: ATR6 interrupt flag (Writing “1” clears the flag.)	
	0	Interrupt is not generated.
	1	Interrupt is generated.

NOTE:

1. Each of the MPWMn_SR [5:0] status bits is shared by a duty match interrupt event or an ADC trigger match interrupt event. If the ADC trigger mode is set disabled, these bits are flagged by duty match interrupts; otherwise, they are flagged by ADC trigger counter match interrupts. The ADC trigger mode is selected in the ATMOD[1:0] bit field of the MPWMn_ATRn register.

17.6.14 MPWMn_IER: MPWM n Interrupt Enable Register

The MPWMn_IER register controls the interrupts that are used for PWM control.

MPWM0_IER=0x4101_4034, MPWM1_IER=0x4101_5034

31 30 29 28 27 26 25 24	23 22 21 20 19 18 17 16	15 14 13 12 11 10 9 8	7	6	5	4	3	2	1	0
Reserved			PRDIE	BOTIE	WHIE	VHIE	UHIE	WLIE	VLIE	ULIE
-			0	0	0	0	0	0	0	0
-			RW	RW	RW	RW	RW	RW	RW	RW

7	PRDIE PRDUIE	Enable the PWM's phase U counter bottom interrupt Phase U, V, and W channels' period interrupt
		0 Disables.
		1 Enables.
6	BOTIE BOTUIE	Enable the PWM's phase U counter bottom interrupt Phase U, V, and W channels' bottom interrupt
		0 Disables.
		1 Enables.
5	WHIE(1)(2) ATR6IE(3)	WH Duty or ATR6 match interrupt
		0 Disables.
		1 Enables.
4	VHIE(1)(2) ATR5IE(3)	VH Duty or ATR5 match interrupt
		0 Disables.
		1 Enables.
3	UHIE(1)(2) ATR4IE(3)	UH Duty or ATR4 match interrupt
		0 Disables.
		1 Enables.
2	WLIE(1)(2) ATR3IE(3)	WL Duty or ATR3 match interrupt
		0 Disables.
		1 Enables.
1	VLIE(1)(2) ATR2IE(3)	VL Duty or ATR2 match interrupt
		0 Disables.
		1 Enables.
0	ULIE(1)(2) ATR1IE(3)	UL Duty or ATR1 match interrupt
		0 Disables.
		1 Enables.

NOTES:

1. Each control bit of the MPWMn_IER [5:0] is shared by a duty match interrupt event or an ADC trigger match interrupt event.
1. If the ADC trigger mode is set disabled, these bits are flagged by interrupt duty match conditions; otherwise, they are flagged by ADC interrupt trigger counter match conditions.
2. The ADC trigger mode is selected in the ATMOD[1:0] bits of the MPWMn_ATRn register

17.6.15 MPWMn_CNT: MPWM n Counter Register

The MPWM_CNT register is used to read the count value that generates a PWM waveform.

MPWM0_CNT=0x4101_4038, MPWM1_CNT=0x4101_5038

31 30 29 28 27 26 25 24	23 22 21 20 19 18 17 16	15 14 13 12 11 10 9 8	7 6 5 4 3 2 1 0
Reserved		CNT[15:0]	
-		0x0000	
-		R0	

15 0	CNT[15:0]	PWM counter value Main counter value in PWM

17.6.16 MPWMn_DTR: MPWM n Dead-Time Register

The MPWMn_DTR register is used to prevent short circuits that may be caused on a circuit or in the program.

MPWM0_DTR=0x4101_403C, MPWM1_DTR=0x4101_503C

31 30 29 28 27 26	25	24	23 22	21	20	19 18	17 16	15 14 13 12 11 10 9 8	7 6 5 4 3 2 1 0
Reserved	DTEN	PSHRT	Reserved	HDTOL	LDTOL	Reserved	DTCLK[1:0]	HDT[7:0]	LDT[7:0]
-	0	0	-	0	0	-	00	0x00	0x00
-	RW	RW	-	RW	RW	-	RW	RW	RW

25	DTEN	Dead-time function enable
	0	Disables dead-time function.
	1	Enables dead-time function.
24	PSHRT	Short condition protection
	0	Disables short condition protection.
	1	Enables short condition protection (Automatically ensures that both the H and L sides are not simultaneously active.)
21	HDTOL	Output level at H side dead-time **NOTE**: This bit and H side dead-time are used to determine short condition. If “HDTOL = 0”, condition of H side external circuit Off: Output Low If “HDTOL = 1”, condition of H side external circuit Off: Output High
	0	Low Level
	1	High Level
20	LDTOL	Output level at L side dead-time **NOTE**: This bit and L side dead-time are used to determine short condition. If “LDTOL = 0”, condition of L side external circuit Off: Output Low If “LDTOL = 1”, condition of L side external circuit Off: Output High
	0	Low Level
	1	High Level
17 16	DTCLK[1:0]	Dead-time prescaler
	00	Dead-time counter uses PWM CLK/2.
	01	Dead-time counter uses PWM CLK/4.
	10	Dead-time counter uses PWM CLK/8.
	11	Dead-time counter uses PWM CLK/16.
15 8	HDT[7:0]	H side dead-time value 0x01 to 0xFF: dead-time **NOTE**: Writing ‘0’ does the same thing as disable.
7 0	LDT[7:0]	L side dead-time value 0x01 to 0xFF: dead-time **NOTE**: Writing ‘0’ does the same thing as disable.

NOTE:

1. The values written to this register are applied to the rising or falling edge dead-times of all three phases U, V, and W. To set a particular phase’s dead-time. This short-circuit protection is applied to the PWM’s internal signals and not to its external signals. When the internal signals from the H and L sides are all in the high level, short-circuit protection induces both sides to generate low-level outputs.

17.6.17 MPWMn_PCR: MPWM n Protection Control Register

The MPWMn_PCR register is used to control the protection operation.

MPWM0_PCR=0x4101_4040, MPWM1_PCR=0x4101_5040

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
PROTPOL	Reserved				PROTDIV[2:0]			PROTD[7:0]								Reserved					PROT7EN	PROT6EN	PROT5EN	Reserved		PROT4EN	PROT3EN	PROT2EN	PROT1EN	Reserved	PROT0EN
0	-				000			0x00								-					0	0	0	-		0	0	0	0	-	0
RW	-				RW			RW								-					RW	RW	RW	-		RW	RW	RW	RW	-	RW

31	PROTPOL	MPxPRT Pin Input Polarity settings (x = 0, 1)	
		0	L-active
		1	H-active
26 24	PROTDIV[2:0]	MPxPRT Pin debounce divider settings (x = 0, 1)	
			2^(PROTDIV[2:0])
23 16	PROTD[7:0]	MPxPRT Pin debounce counter settings (x = 0, 1)	
		0x00: Not debounce 0x01 to 0xff: Debounce counter settings Debounce Time = PROTD[7:0] / (MPWMCLK/(2^(PROTDIV[2:0]))	
10	PROT7EN	ADC2_CMPR1 protection operation enable	
		0	Disables.
		1	Enables.
9	PROT6EN	ADC1_CMPR1 protection operation enable	
		0	Disables.
		1	Enables.
8	PROT5EN	ADC0_CMPR1 protection operation enable	
		0	Disables.
		1	Enables.
5	PROT4EN	CMP3 protection operation enable	
		0	Disables.
		1	Enables.
4	PROT3EN	CMP2 protection operation enable	
		0	Disables.
		1	Enables.
3	PROT2EN	CMP1 protection operation enable	
		0	Disables.
		1	Enables.
2	PROT1EN	CMP0 protection operation enable	
		0	Disables.
		1	Enables.
0	PROT0EN	MPxPRT Pin protection operation enable (x = 0, 1)	
		0	Disables.
		1	Enables.

17.6.18 MPWMn_PSR: MPWM n Protection Status Register

Using the MPWMn_PSR register, users can check the status of interrupts that are generated during the protection operation.

MPWM0_PSR=0x4101_4044, MPWM1_PSR=0x4101_5044

31 30 29 28 27 26 25 24	23 22 21 20 19 18 17 16	15 14 13 12 11	10	9	8	7 6	5	4	3	2	1	0
Reserved			PROT7IF	PROT6IF	PROT5IF	Reserved	PROT4IF	PROT3IF	PROT2IF	PROT1IF	Reserved	PROT0IF
-			0	0	0	-	0	0	0	0	-	0
-			RWC1	RWC1	RWC1	-	RWC1	RWC1	RWC1	RWC1	-	RWC1

10	PROT7IF	ADC2_CMPR1 protection interrupt flag (Writing “1” clears the flag.)
	0	Protection interrupt does not occur.
	1	Protection interrupt occurs.
9	PROT6IF	ADC1_CMPR1 protection interrupt flag (Writing “1” clears the flag.)
	0	Protection interrupt does not occur.
	1	Protection interrupt occurs.
8	PROT5IF	ADC0_CMPR1 protection interrupt flag (Writing “1” clears the flag.)
	0	Protection interrupt does not occur.
	1	Protection interrupt occurs.
5	PROT4IF	CMP3 protection interrupt flag (Writing “1” clears the flag.)
	0	Protection interrupt does not occur.
	1	Protection interrupt occurs.
4	PROT3IF	CMP2 protection interrupt flag (Writing “1” clears the flag.)
	0	Protection interrupt does not occur.
	1	Protection interrupt occurs.
3	PROT2IF	CMP1 protection interrupt flag (Writing “1” clears the flag.)
	0	Protection interrupt does not occur.
	1	Protection interrupt occurs.
2	PROT1IF	CMP0 protection interrupt flag (Writing “1” clears the flag.)
	1	Protection interrupt does not occur.
	0	Protection interrupt occurs.
0	PROT0IF	MPxPRT Pin protection interrupt flag (x = 0, 1) (Writing “1” clears the flag.)
	1	Protection interrupt does not occur.
	0	Protection interrupt occurs.

17.6.19 MPWMn_OCR: MPWM Overcurrent Detection Register

The MPWMn_OCR register controls the protection operation when over-current is caused.

MPWM0_OCR=0x4101_4048, MPWM1_OCR=0x4101_5048

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
OCPOL	Reserved				OCDIV[2:0]			OCD[7:0]								Reserved					OC7EN	OC6EN	OC5EN	Reserved		OC4EN	OC3EN	OC2EN	OC1EN	Reserved	OC0EN
0	-				0x00			0x00								-					0	0	0	-		0	0	0	0	-	0
RW	-				RW			RW								-					RW	RW	RW	-		RW	RW	RW	RW	-	RW

31	OCPOL	OCIN Pin Input Polarity settings
	0	L-active
	1	H-active
26 24	OCDIV[2:0]	OCIN Pin debounce divider settings
		2^(OCINDIV[2:0])
23 16	OCD[7:0]	OCIN Pin debounce counter settings
		I0x00: Not debounce 0x01 to 0xFF: Debounce counter settings Debounce Time = OCD [7:0] /(MPWMCLK/(2^(OCDIV[2:0]))
10	OC7EN	ADC2 CMPR1 protection operation enable
	0	Disables.
	1	Enables.
9	OC6EN	ADC1 CMPR1 protection operation enable
	0	Disables.
	1	Enables.
8	OC5EN	ADC0 CMPR1 protection operation enable
	0	Disables.
	1	Enables.
5	OC4EN	CMP3 protection operation enable
	0	Disables.
	1	Enables.
4	OC3EN	CMP2 protection operation enable
	0	Disables.
	1	Enables.
3	OC2EN	CMP1 protection operation enable
	0	Disables.
	1	Enables.
2	OC1EN	CMP0 protection operation enable
	0	Disables.
	1	Enables.
0	OC0EN	MPxOCP Pin protection operation enable
	0	Disables.
	1	Enables.

17.6.20 MPWMn_OSR: MPWM Overcurrent Detection Status Register

Using the MPWMn_OSR register, users can check the status of interrupts that are generated during the over-current protection operation.

MPWM0_OSR=0x4101_404C, MPWM1_OSR=0x4101_504C

31 30 29 28 27 26 25 24	23 22 21 20 19 18 17 16	15 14 13 12 11	10	9	8	7 6	5	4	3	2	1	0
Reserved			OC7IF	OC6IF	OC5IF	Reserved	OC4IF	OC3IF	OC2IF	OC1IF	Reserved	OC0IF
-			0	0	0	-	0	0	0	0	-	0
-			RWC1	RWC1	RWC1	-	RWC1	RWC1	RWC1	RWC1	-	RWC1

10	OC7IF	ADC2_CMPR1 protection interrupt flag (Writing “1” clears the flag.)
	0	Protection interrupt does not occur.
	1	Protection interrupt occurs.
9	OC6IF	ADC1_CMPR1 protection interrupt flag (Writing “1” clears the flag.)
	0	Protection interrupt does not occur.
	1	Protection interrupt occurs.
8	OC5IF	ADC0_CMPR1 protection interrupt flag (Writing “1” clears the flag.)
	0	Protection interrupt does not occur.
	1	Protection interrupt occurs.
5	OC4IF	CMP3 protection interrupt flag (Writing “1” clears the flag.)
	0	Protection interrupt does not occur.
	1	Protection interrupt occurs.
4	OC3IF	CMP2 protection interrupt flag (Writing “1” clears the flag.)
	0	Protection interrupt does not occur.
	1	Protection interrupt occurred.
3	OC2IF	CMP1 protection interrupt flag (Writing “1” clears the flag.)
	0	Protection interrupt does not occur.
	1	Protection interrupt occurs.
2	OC1IF	CMP0 protection interrupt flag (Writing “1” clears the flag.)
	0	Protection interrupt does not occur.
	1	Protection interrupt occurs.
0	OC0IF	MPxOCP Pin protection interrupt flag (Writing “1” clears the flag.)
	0	Protection interrupt does not occur.
	1	Protection interrupt occurs.

17.6.21 MPWMn_ATRx: MPWM n ADC Trigger Counter Register

The MPWMn_ATRx register is 32 bits wide.

MPWM0_ATR1=0x4101_4058, MPWM0_ATR2=0x4101_405C, MPWM0_ATR3=0x4101_4060
MPWM0_ATR4=0x4101_4064, MPWM0_ATR5=0x4101_4068, MPWM0_ATR6=0x4101_406C
MPWM1_ATR1=0x4101_5058, MPWM1_ATR2=0x4101_505C, MPWM1_ATR3=0x4101_5060
MPWM1_ATR4=0x4101_5064, MPWM1_ATR5=0x4101_5068, MPWM1_ATR6=0x4101_506C

31 30 29 28 27 26 25 24	23 22 21 20	19	18	17 16	15 14 13 12 11 10 9 8	7 6 5 4 3 2 1 0
Reserved		ATUDT	Reserved	ATMOD[1:0]	ATCNT[15:0]	
-		0	-	0	0x0000	
-		RW	-	RW	RW	

19	ATUDT	Trigger register update mode
	0	ADC trigger value applied at period match event (at the same time with period and duty registers update)
	1	Trigger register update mode When this bit is set enabled, written trigger register values are sent to the trigger compare block after two PWM clock cycles (through synchronization logic)
17 16	ATMOD[1:0]	ADC trigger mode register
	00	ADC trigger Disable
	01	Trigger out when up count match
	10	Trigger out when down count match (Motor PWM mode 0 only)
	11	Trigger out when up-down count match (Motor PWM mode 0 only)
15 0	ATCNT[15:0]	ADC trigger counter (ADC trigger counter value must be smaller than the PWM period.)) Setting the counter value to 0 prevents interrupts from being triggered.

17.6.22 MPWM Register Map Summary

Table 175. MPWM Register Map Summary

Offset	Register name	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0x00	MPWMn_MR	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	CLKDIV[3:0]				MOTORB[1:0]		Res.	Res.	UOL	Res.	TUOL	BUOL	UAO	Res.	TUP	BUP	Res.	MCHMOD[1:0]		Res.
	Reset value													0	0	0	0	0	0			0		0	0	0		0	0		0	0	
0x04	MPWMn_OLR	Res.	Res.	Res.	Res.	Res.	Res.	Res.	FORCEU	Res.	Res.	POCWH	POCVH	POCUH	POCWL	POCVL	POCUL	Res.	Res.	DOLWH	DOLVH	DOLUH	DOLWL	DOLVL	DOLUL	Res.	Res.	SLWH	SLVH	SLUH	SLWL	SLVL	SLUL
	Reset value								0			0	0	0	0	0	0			0	0	0	0	0	0			0	0	0	0	0	0
0x08	MPWMn_FOLR	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	FOLKEY[7:0]								Res.	Res.	Res.	Res.	Res.	SWFO	PRTFO	OCFO	Res.	Res.	FLWH	FLVH	FLUH	FLWL	FLVL	FLUL
	Reset value									-	-	-	-	-	-	-	-						0	0	0			0	0	0	0	0	0
0x0C	MPWMn_PRD	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	PERIOD[15:0]															
	Reset value																	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0
0x10	MPWMn_DUH	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	DUTY_UH[15:0]															
	Reset value																	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1
0x14	MPWMn_DVH	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	DUTY_VH[15:0]															
	Reset value																	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1
0x18	MPWMn_DWH	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	DUTY_WH[15:0]															
	Reset value																	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1
0x1C	MPWMn_DUL	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	DUTY_UL[15:0]															
	Reset value																	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1
0x20	MPWMn_DVL	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	DUTY_VL[15:0]															
	Reset value																	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1
0x24	MPWMn_DWL	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	DUTY_WL[15:0]															
	Reset value																	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1
0x28	MPWMn_CR1	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	IRQN[2:0]			Res.	Res.	Res.	Res.	Res.	Res.	Res.	PWMEN
	Reset value																						0	0	0								0

Table 175. MPWM Register Map Summary: Normal PWM Mode (continued)

Offset	Register name	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0x2C	MPWMn_CR2	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	HALT	Res.	Res.	Res.	Res.	Res.	Res.	PSTART
	Reset value																									0							0
0x30	MPWMn_SR	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	DOWN	IRQCNT[2:0]			Res.	Res.	Res.	Res.	PRDIF	BOTIF	DWHIF/ATR6F	DVHIF/ATR5F	DUHIF/ATR4F	DWLIF/ATR3F	DVLIF/ATR2F	DULIF/ATR1F
	Reset value																	0	0	0	0					0	0	0	0	0	0	0	0
0x34	MPWMn_IER	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.									PRDIE	BOTIE	WHIE	VHIE	UHIE	WLIE	VLIE	ULIE
	Reset value																									0	0	0	0	0	0	0	0
0x38	MPWMn_CNT	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	CNT[15:0]															
	Reset value																	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
0x3C	MPWMn_DTR	Res.	Res.	Res.	Res.	Res.	Res.	DTEN	PSHRT	Res.	Res.	HDTOL	LDTOL	Res.	Res.	DTCLK[1:0]		HDT[7:0]								LDT[7:0]							
	Reset value							0	0			0	0			0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
0x40	MPWMn_PCR	PROTPOL	Res.	Res.	Res.	Res.	PROTDIV[2:0]			PROTD[7:0]								Res.	Res.	Res.	Res.	Res.	PROT7EN	PROT6EN	PROT5EN	Res.	Res.	PROT4EN	PROT3EN	PROT2EN	PROT1EN	Res.	PROT0EN
	Reset value	0					0	0	0	0	0	0	0	0	0	0	0						0	0	0			0	0	0	0		0
0x44	MPWMn_PSR	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	PROT7IF	PROT6IF	PROT5IF	Res.	Res.	PROT4IF	PROT3IF	PROT2IF	PROT1IF	Res.	PROT0IF
	Reset value																						0	0	0			0	0	0	0		0
0x48	MPWMn_OCR	OCPOL	Res.	Res.	Res.	Res.	OCDIV[2:0]			OCD[7:0]								Res.	Res.	Res.	Res.	Res.	OC7EN	OC6EN	OC5EN	Res.	Res.	OC4EN	OC3EN	OC2EN	OC1EN	Res.	OC0EN
	Reset value	0	0				0	0	0	0	0	0	0	0	0	0	0						0	0	0			0	0	0	0		0
0x4C	MPWMn_OSR	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	OC7IF	OC6IF	OC5IF	Res.	Res.	OC4IF	OC3IF	OC2IF	OC1IF	Res.	OC0IF
	Reset value																						0	0	0			0	0	0	0		0
0x58	MPWMn_ATR1	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	ATUDT	Res.	ATMOD[1:0]		ATCNT[15:0]															
	Reset value													0		0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Table 175. MPWM Register Map Summary: Normal PWM Mode (continued)

Offset	Register name	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0x5C	MPWMn_ATR2	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	ATUDT	Res.	ATMOD[1:0]		ATCNT[15:0]															
	Reset value													0		0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
0x60	MPWMn_ATR3	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	ATUDT	Res.	ATMOD[1:0]		ATCNT[15:0]															
	Reset value													0		0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
0x64	MPWMn_ATR4	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	ATUDT	Res.	ATMOD[1:0]		ATCNT[15:0]															
	Reset value													0		0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
0x68	MPWMn_ATR5	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	ATUDT	Res.	ATMOD[1:0]		ATCNT[15:0]															
	Reset value													0		0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
0x6C	MPWMn_ATR6	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	ATUDT	Res.	ATMOD[1:0]		ATCNT[15:0]															
	Reset value													0		0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

18. Analog-to-Digital Converter (ADC)

18.1 ADC Introduction

The A34G43x series ADC module is a 12-bit resolution Analog-to-Digital Converter (ADC) that consists of three independently controllable units and has up to 24 analog input channels. Input channels consist of 24 external pin inputs and two internal input channels.

This ADC module can process A/D conversion of many different channels in Single mode, Sequential mode, Multiple mode and Burst mode. It can sample the voltage inputs of multiple channels at critical time, which is required for various applications and can be achieved using the triggering functionality in association with TIMER and MPWM.

Once the A/D conversion is complete, the result is left aligned in 16-bit data registers.

The ADC is mapped to the AHB and capable of high-speed operation based on the internal clock source. In addition, it uses built-in DMA to reduce microcontroller core load while enabling fast data sampling.

The analog input monitoring and comparison capabilities of the ADC allow the applications to determine whether the input voltage is above or below the predefined level.

NOTES:

1. Peripheral names ending in 'n' represent units, and registers or bit field ending in 'x' represent channels.
2. For example, ‘n’ in ADCn means unit.(0 to 2) 'x' in ANx means the number of analog input channels.(0 to 23) ‘x’ in CSTRx means register ADC_CSTR1, ADC_CSTR2.
3. The definition of Unit means an independent peripheral capable of multitasking.
4. The definition of Channel means a register or bit field inside a peripheral, and multitasking is not possible.

18.2 ADC Main Features

The key features of the ADC module are listed below:

- Composed of three independent ADC units
- 12-bit analog to digital resolution
- 1.5 Msps maximum conversion rate with full resolution
  - @42 MHz ADC clock division
- Three independently implemented ADCs.
- 2.7 V minimum to 5.5 V operating voltage range
- Data can be managed by DMA for high-performance conversion.
- Analog Input channels
  - ADC0 is connected to 16 external channels.
  - ADC1 is connected to 16 external channels.
  - ADC2 is connected to 16 external channels.
- Internally dedicated channels
  - V_{CORE}, DAC0O, DAC1O internal reference voltage outputs connected to ADC0.
  - V_{CORE}, DAC0O, DAC1O internal reference voltage outputs connected to ADC1.
  - V_{CORE}, DAC0O, DAC1O internal reference voltage outputs connected to ADC2.
- Flexible sampling time control
- Four Conversion modes
  - Single mode
  - Sequential mode
  - Multiple mode
  - Burst mode
- Up to eight sequential conversions can be supported.
- Internal reference outputs (V_{CORE}) for voltage monitoring
- Software trigger sources per channel
  - Internal software trigger source
  - TIMERn (n = 0, 1, 2, …, 9) trigger sources
  - MPWMn (n = 0, 1) trigger sources

- Interrupt generation at the end of conversion in the last sequence, the trigger conversion, the DMA done and the comparison interrupt.
- ADC input range
  - $\mathrm{AGND} \leq \mathrm{ANx\,(x = 0\ to\ 23)} \leq \mathrm{AVDD}\ \{(\mathrm{VDD} - 0.3) \leq \mathrm{AVDD} \leq (\mathrm{VDD} + 0.3)\}$

NOTE:

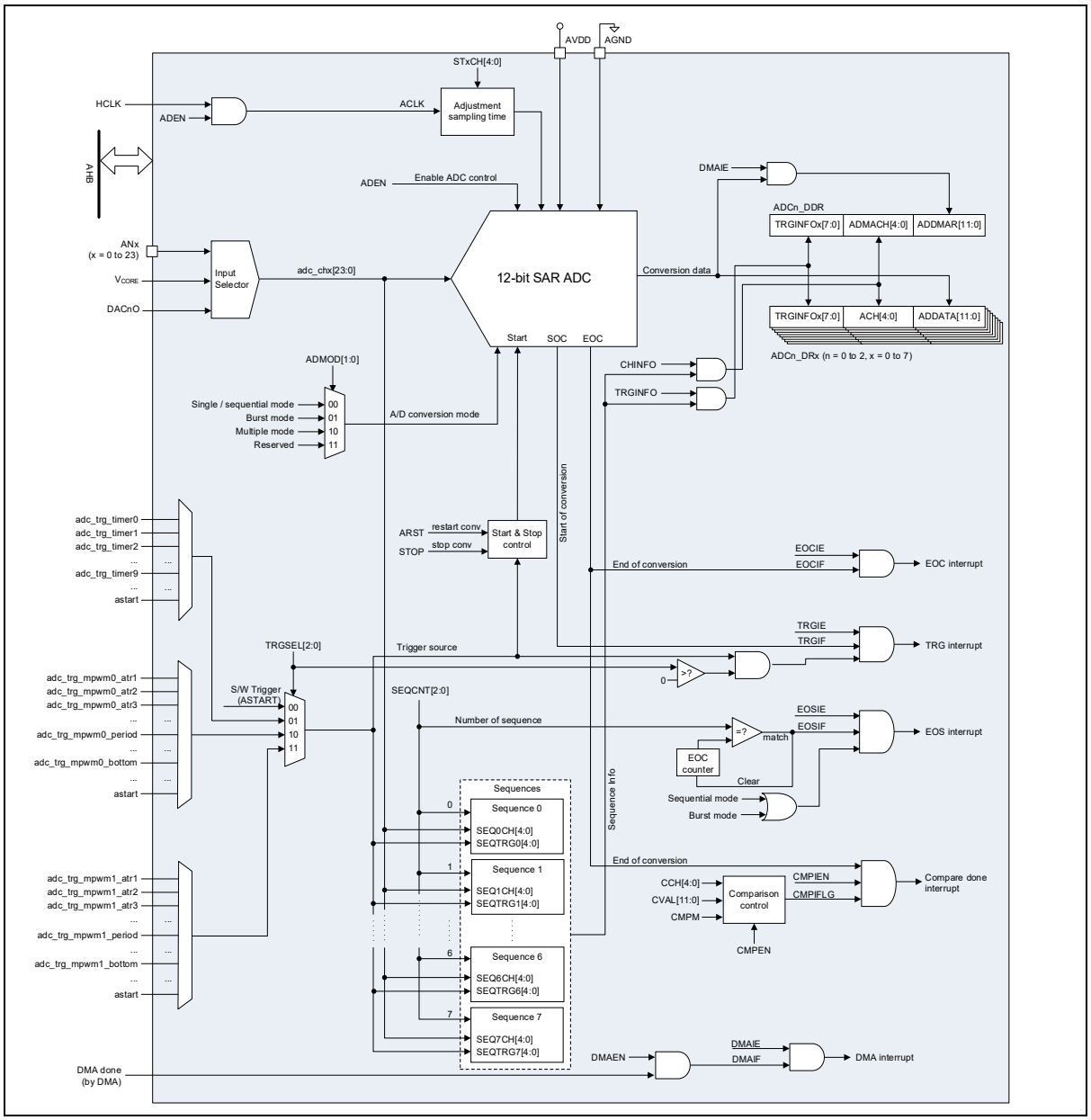
1. ADC performance such as INL or DNL may be degraded if other analog modules use the AVDD reference voltage. Therefore, in applications where the precision of the ADC is important, it is recommended that the software be configured to use only the ADC analog module.

18.3 ADC Functional Description

18.3.1 ADC Block Diagram

Figure 195 shows a block diagram of the 12-bit ADC.

Figure 195. 12-bit ADC Block Diagram



18.3.2 ADC Pins and Signals

Table 176 describes pins assigned for the ADC.

Table 176. Pin Assignment of ADC External: Pins

Pin Name	Signal Type(1)	Description	Supported Packages		
			A34G43xYL (120-LQFP)	A34G43xVL (100-LQFP)	A34G43xRL (64-LQFP)
AVDD	P	Analog power (reference voltage)	O	O	O
AGND	P	Analog GND	O	O	O
AN0	A	ADC external analog input 0	O	O	O
AN1	A	ADC external analog input 1	O	O	O
AN2	A	ADC external analog input 2	O	O	O
AN3	A	ADC external analog input 3	O	O	O
AN4	A	ADC external analog input 4	O	O	O
AN5	A	ADC external analog input 5	O	O	O
AN6	A	ADC external analog input 6	O	O	O
AN7	A	ADC external analog input 7	O	O	O
AN8	A	ADC external analog input 8	O	O	O
AN9	A	ADC external analog input 9	O	O	O
AN10	A	ADC external analog input 10	O	O	O
AN11	A	ADC external analog input 11	O	O	O
AN12	A	ADC external analog input 12	O	O	O
AN13	A	ADC external analog input 13	O	O	O
AN14	A	ADC external analog input 14	O	O	O
AN15	A	ADC external analog input 15	O	O	O
AN16	A	ADC external analog input 16	O	O	X
AN17	A	ADC external analog input 17	O	O	X
AN18	A	ADC external analog input 18	O	O	X
AN19	A	ADC external analog input 19	O	O	X
AN20	A	ADC external analog input 20	O	O	X
AN21	A	ADC external analog input 21	O	O	X
AN22	A	ADC external analog input 22	O	O	X
AN23	A	ADC external analog input 23	O	O	X

NOTE:

1. P: Power, O: Output, A: Analog Input.

Table 177 describes the input and output signals of the ADC.

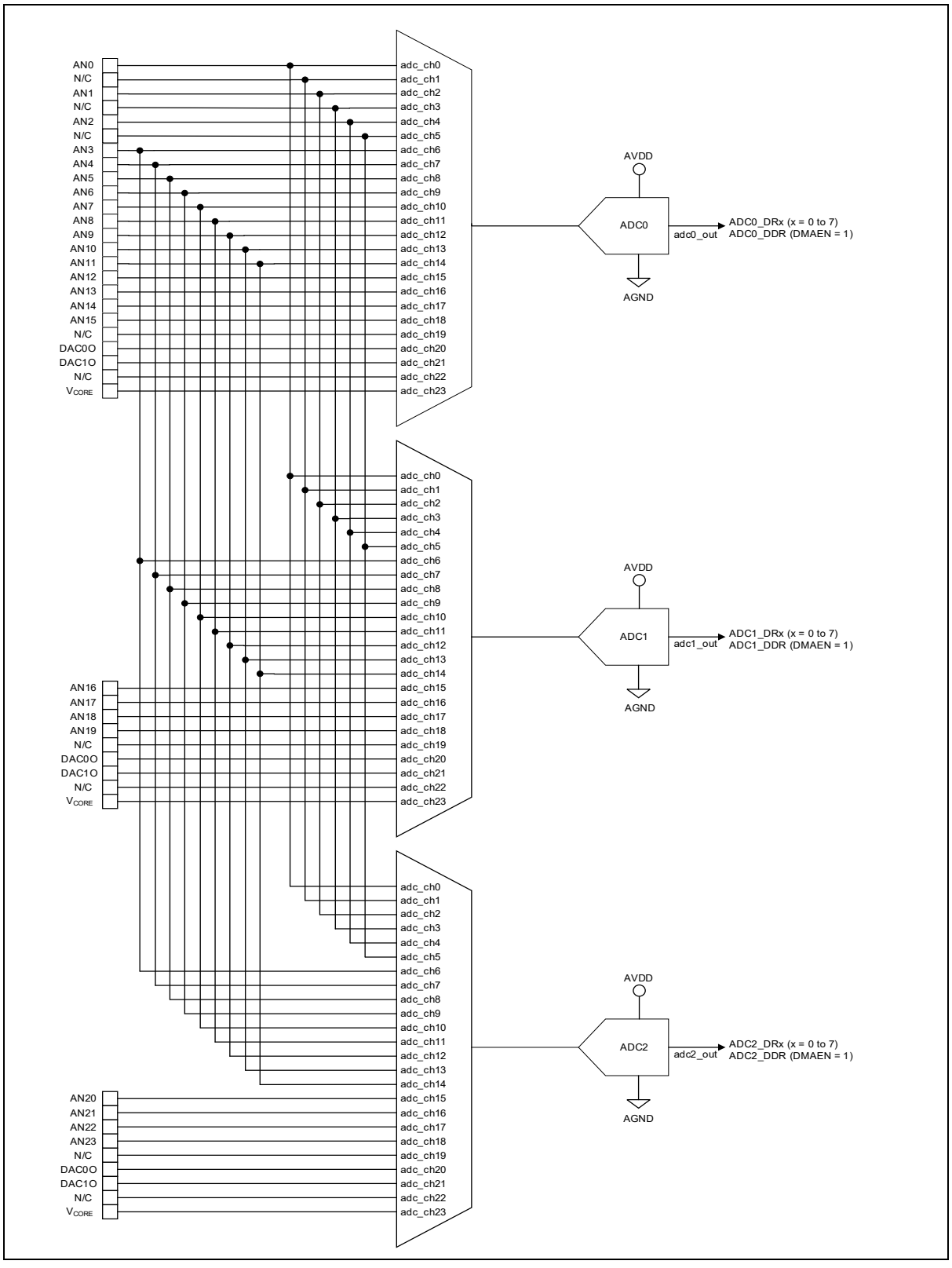
Table 177. ADC Input/Output Signals

Signal Name	Signal Type	Description
ACLK	clock	ADC operating clock
adc_trg_timerx (x = 0 to 9)	Inputs	The internal trigger source of TIMERn (n = 0 to 9)
adc_trg_mpwmn_atrx (n = 0 to 1, x = 1 to 6)	Inputs	The internal trigger source of MPWMn (n = 0 to 1)
adc_trg_astart	Inputs	The internal clock of ADC
DACxO (x = 0 to 1)	Analog Input	Output voltage from DACn (n = 0 to 1)
V_{CORE}	Input	Output voltage from internal reference voltage (1.0 V)
adc_ch[23:0]	Analog Input	ADC external input signal
adcx_out (x = 0 to 2)	Digital Output	AD Conversion data by ADCn (n = 0 to 2)

18.3.3 ADC Connectivity

18.3.3.1 ADC Internal Channel Wiring

Figure 196. 12-bit ADC Internal Channel Wiring



18.3.3.2 ADC Input Signals and External Input Channels

Table 178. ADC Input Signals and Input Pins(1)

ADC Input Signal	SEQxCH (x = 0 to 7)	ADC0	ADC1	ADC2
adc_ch0	0	AN0	AN0	AN0
adc_ch1	1	N/C	N/C	N/C
adc_ch2	2	AN1	AN1	AN1
adc_ch3	3	N/C	N/C	N/C
adc_ch4	4	AN2	AN2	AN2
adc_ch5	5	N/C	N/C	N/C
adc_ch6	6	AN3	AN3	AN3
adc_ch7	7	AN4	AN4	AN4
adc_ch8	8	AN5	AN5	AN5
adc_ch9	9	AN6	AN6	AN6
adc_ch10	10	AN7	AN7	AN7
adc_ch11	11	AN8	AN8	AN8
adc_ch12	12	AN9	AN9	AN9
adc_ch13	13	AN10	AN10	AN10
adc_ch14	14	AN11	AN11	AN11
adc_ch15	15	AN12	AN16	AN20
adc_ch16	16	AN13	AN17	AN21
adc_ch17	17	AN14	AN18	AN22
adc_ch18	18	AN15	AN19	AN23
adc_ch19	19	N/C	N/C	N/C
adc_ch20	20	DAC0O	DAC0O	DAC0O
adc_ch21	21	DAC1O	DAC1O	DAC1O
adc_ch22	22	N/C	N/C	N/C
adc_ch23	23	V_{CORE}	V_{CORE}	V_{CORE}

NOTES:

1. The SEQxCH (x = 0 to 7) are configured in the ADC_SCSR1 and ADC_SCSR2 registers (n = 0 to 2).
2. Single mode conversion uses the SEQ0CH in ADC_SCSR1 register and the ADC_DR0 register only (n = 0 to 2).

18.3.4 ADC Clocks (ACLK)

18.3.4.1 ADC Module Control

Before enabling the ADC clock, users must select an ADC unit to be enabled by setting a bit of the ADC[2:0] in the SCU_PER2 register to ‘1’ which is corresponding to the ADC unit. Then, enable the clock of the selected ADC unit by setting a bit of the ADC[2:0] in the SCU_PCER2 register to ‘1’ which corresponds to the clock of the selected ADC unit.

To initialize the registers of each ADC unit, set each corresponding bit of the ADC[2:0] in SCU_PER2 register and the ADC[2:0] bits in the SCU_PCER2 register to ‘0’ and then set back to ‘1’.

Each bit of the ADC[2:0] bits in the SCU_PER2 register controls the corresponding ADC unit. To enable or disable all ADC units at a time, set the ADC[2:0] in SCU_PER2 register to the number shown below:

- If the ADC[2:0] bits in the SCU_PER2 register = ‘000’, all ADC units are disabled.
- If the ADC[2:0] bits in the SCU_PER2 register = ‘111’, all ADC units are enabled.

To provide the ADC unit with the ADC clock, configure the ADC[2:0] bits in the SCU_PCER2 register referring to the settings shown below:

- If the ADC[2:0] bits in the SCU_PCER2 register = ‘000’, all ADC clocks are disabled.
- If the ADC[2:0] bits in the SCU_PCER2 register = ‘111’, all ADC clocks are enabled.

18.3.4.2 ADC Clock Sources

The HCLK operates as the ADC operation clock (ACLK) based on the AHB domain. Internal clock source can be selected to operate as the ACLK by setting the EXTCLK bit in the ADCn_CCR register. Each clock source for the three ADC units can be configured independently.

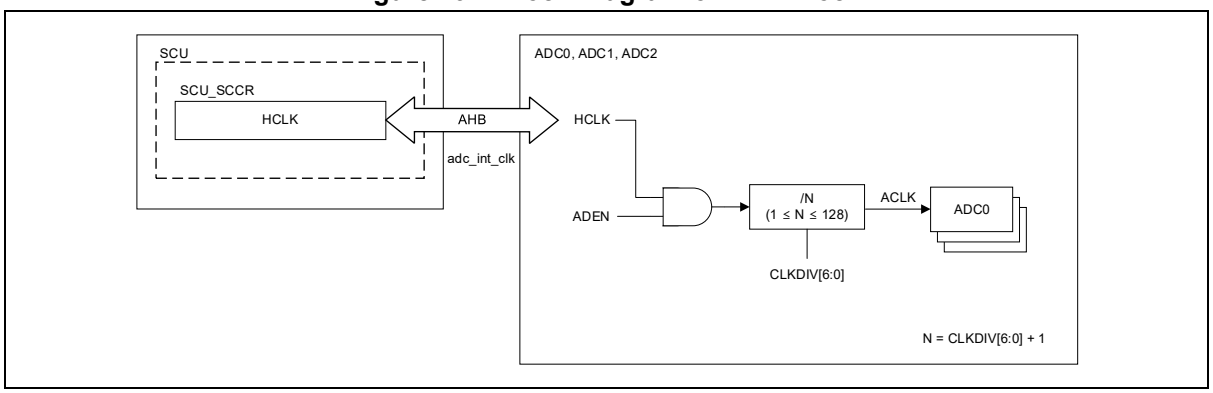
Internal Clock Source

The programmable divider coefficient can be selected to range from 0 to 127 by configuring the CLKDIV[6:0] bits in the ADCn_CCR register, and the PCLK is divided by the coefficient to be used as the ADC operation clock.

NOTES:

1. Main system clock must be divided at least 3 times to be used as an ADC operation clock.
2. The CLKDIV[6:0] bits in the ADCn_CCR register is used only when the HCLK, the ADC's internal system clock is used.
3. If the CLKDIV[6:0] value is set to two or larger number for the ADC clock division, the ADC clock's frequency must not exceed 42 MHz.

Figure 197. Block Diagram of ADC Clock



18.3.5 ADC On-Off Control (ADEN)

Before starting the A/D conversion, users must enable the ADC module using the ADEN in ADCn_MR register.

- Setting the ADEN = '1' enables the ADC unit. When the ADC unit is enabled, the ADC functionality and the ASTART bit control are possible.
- Setting the ADEN = '0' disables the ADC unit. When the ADC unit is disabled, the A/D conversion cannot be performed.

18.3.5.1 Software Procedure to Enable ADC

1. Clear the EOCIF bit in the ADCn_SR register by writing '1'.
2. Setting the ADEN bit in the ADCn_MR register to '1' enables the ADC module.

18.3.5.2 Software Procedure to Disable ADC

1. Confirm that the ASTART bit and the ASTOP bit in the ADC_CR register are set to '0', respec tively. This ensures that no conversion is ongoing. If required, set the ASTOP bit to '1' to stop the conversion and then wait until the ASTOP bit becomes '0'[(1)].
2. Disable the ADC module by setting the ADEN bit in the ADCn_MR register to '0'

NOTE:

1. This ASTOP bit is auto cleared at the next ADC clock cycle.

18.3.6 Constraints when Writing the ADC Control Bits

18.3.6.1 When Writing ADC Control Bit

While accessing the SCU_PER2, SCU_PCER2, and ADC_CCR registers to enable the ADC operation clock source and ADC module, the ADC module must be disabled. For this, set the ADEN bit in the ADCn_MR register to ‘0’.

During the A/D conversion operations, if users write to each ADC control bit of the SCU_PER2, SCU_PCER2, and ADC_CCR registers, the ADC clock bus and ADC registers are not controllable and become Unknown state.

Software can write to the control bits configuring the A/D conversion, only when the ADC module is enabled (ADEN = ‘1’) and the A/D conversion is not performed (e.g., ASTART = ‘0’).

- Set the ADEN bit in the ADCn_MR register to ‘0’.
- Clear all bits in the ADCn_CR register.

18.3.6.2 When Initializing ADC

The initial state of the End-of-Conversion (EOC) of the ADC is unknown when the microcontroller is first powered on. At this point of time, leakage current may flow if the microcontroller enters SLEEP mode or DEEP-SLEEP mode. To avoid this situation, set the ASTART to ‘1’ and perform two dummy transfers of the ADCn_CR register to clear the EOC.

Users must consider this situation when initializing the ADC, since it cannot be processed by hardware.

18.3.7 Channel Selection

18.3.7.1 ADC Analog Input Channel

Each ADC has up to 24 multiplexed channels.

When the analog inputs come from GPIO pads:

1. For the analog inputs on the ADC channel, set mode of the channel you want to use to ADC using the Pn_MR1 or Pn_MR2 registers (n = A to F).
2. Set the pin direction to ‘input’ using the Pn_CR register (n = A to F).
3. Disable the internal pull-up or pull-down resistance functionality using the Pn_PRCR register (n = A to F).

The ADC is connected to the analog inputs as described below:

- Internal reference voltage DAC0O is connected to each adc_ch20.
- Internal reference voltage DAC1O is connected to each adc_ch21.
- Internal reference voltage 1.0 V (V_{CORE}) is connected to each adc_ch23.

NOTE:

1. Since the internal analog inputs ($V_{CORE,}$ DACxO) use the analog source enabled when the system internal power is applied, additional bit programming for the analog source settings is not necessary.

18.3.7.2 ADC Conversion Channel Selection

To convert the input signal received from the analog channel, users must select a channel to sample by setting the SEQxCH bits in the ADCn_SCSR1 or ADCn_SCSR2 registers. (n = 0 to 2, x = 0 to 7)

The 'SEQxCH' bit fields are used to perform multiple ADC. 'x' in the bit field indicates the ADC conversion order. For example, if you set SEQ0CH = 1, SEQ1CH = 5, and SEQ2CH = 20, the first conversion performs adc_ch1, the second performs adc_ch5, and the third performs adc_ch20.

For the A/D conversion operation in single mode, use the SEQ0CH bits in the ADCn_SCSR1 register to set the channel.

Table 179. ADC Input Signal and Input Pin(1)

ADC Input Signal	ADC_SCSR1				ADC_SCSR2			
	SEQ0CH[4:0](2)	SEQ1CH[4:0]	SEQ2CH[4:0]	SEQ3CH[4:0]	SEQ4CH[4:0]	SEQ5CH[4:0]	SEQ6CH[4:0]	SEQ7CH[4:0]
adc_ch0	0	0	0	0	0	0	0	0
adc_ch1	1	1	1	1	1	1	1	1
adc_ch2	2	2	2	2	2	2	2	2
adc_ch3	3	3	3	3	3	3	3	3
adc_ch4	4	4	4	4	4	4	4	4
adc_ch5	5	5	5	5	5	5	5	5
adc_ch6	6	6	6	6	6	6	6	6
adc_ch7	7	7	7	7	7	7	7	7
adc_ch8	8	8	8	8	8	8	8	8
adc_ch9	9	9	9	9	9	9	9	9
adc_ch10	10	10	10	10	10	10	10	10
adc_ch11	11	11	11	11	11	11	11	11
adc_ch12	12	12	12	12	12	12	12	12
adc_ch13	13	13	13	13	13	13	13	13
adc_ch14	14	14	14	14	14	14	14	14
adc_ch15	15	15	15	15	15	15	15	15
adc_ch16	16	16	16	16	16	16	16	16
adc_ch17	17	17	17	17	17	17	17	17
adc_ch18	18	18	18	18	18	18	18	18
adc_ch19	19	19	19	19	19	19	19	19
adc_ch20	20	20	20	20	20	20	20	20
adc_ch21	21	21	21	21	21	21	21	21
adc_ch22	22	22	22	22	22	22	22	22
adc_ch23	23	23	23	23	23	23	23	23

NOTES:

1. For the multi-channel sampling, up to 8 channels can be input and the number of conversions is set in the SEQCNT[2:0] in ADCn_MR register.
2. For the single-channel sampling, the channel must be selected by configuring the SEQ0CH[4:0] in ADCn_SCSR1 register. Once the A/D conversion starts with the ASTART in ADCn_CR register set to '1', information of the channel currently converting is stored in the CACH[4:0] in ADCn_CSCR register and the most recently sampled channel value is retained.

18.3.7.3 Register Access when Changing A/D Conversion Channel

The ADCn_SCSR1 or ADCn_SCSR2 registers cannot be modified or changed during the A/D conversion. To change the channel you want to sample, write ‘1’ to the ASTOP bits in the ADCn_CR register to stop the A/D conversion.

Users can select a number of conversions in a sequence by configuring the SEQCNT[2:0] in ADCn_MR register. Updates are available before the Sequence A/D conversion, or immediately after the End of Sequence (EOS) before a new sequence is executed.

Assuming that the SEQCNT[2:0] bits is set to ‘3’ and the sequence is configured in the order of adc_ch0, adc_ch1, adc_ch2, and adc_ch3 by using the SEQ0CH, SEQ1CH, SEQ2CH, and SEQ3CH, and the CSEQN[2:0] bits in the ADCn_CSCR register is set to ‘2’, then the A/D conversion is performed in the order of adc_ch2 and adc_ch3, and the Sequence End Interrupt Flag, EOSIF in ADCn_SR register is set to ‘1’.

18.3.8 Programmable Sampling Time

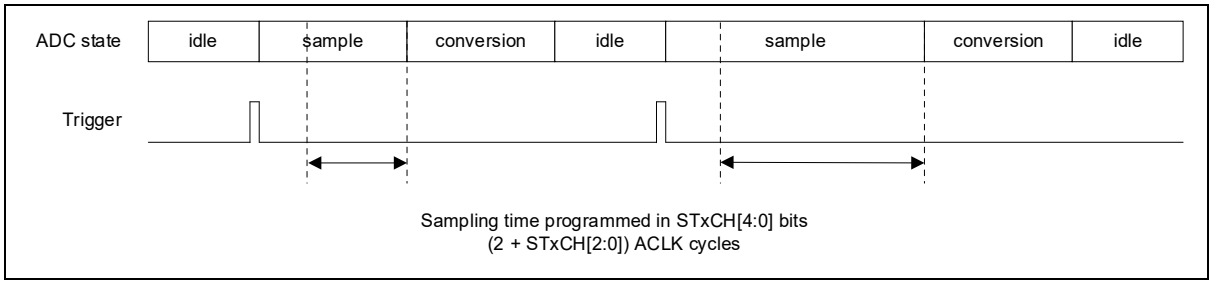
The STxCH[4:0] in ADCn_CSTRx register determines the time window where the next trigger is recognized. This set value is applied right after the occurrence of the current trigger. It is applied equally to all ADC channels and must be set before starting the ADC sampling.

The ADC sampling time is calculated by (2 + STxCH[4:0]) ACLK cycles. The minimum sampling time is two ACLK cycles, and the sampling channel is always active when the STxCH[4:0] in ADCn_CSTRx register are set to ‘11111’.

- If STxCH[4:0] = ‘00000’: 2 ACLK cycles
- If STxCH[4:0] = ‘00001’: 3 ACLK cycles
- If STxCH[4:0] = ‘00010’: 4 ACLK cycles

  …
- If STxCH[4:0] = ‘11101’: 31 ACLK cycles
- If STxCH[4:0] = ‘11110’: 32 ACLK cycles
- If STxCH[4:0] = ‘11111’: sampling channel is always active

Figure 198. Example of ADC Conversion Timing



18.3.9 Single Mode Conversion (ADMOD[1:0] = '00', SEQCNT[2:0] = 0)

Single mode conversion means that the A/D conversion is performed in a single channel. The single mode performs the sampling and the A/D conversion of analog inputs on the channel selected in the SEQ0CH[4:0] bits in the ADCn_SCSR1 register.

- If ADMOD[1:0] = '00' and SEQCNT[2:0] = '000', A/D conversion in single mode.

There are Four Types of Trigger Sources in Single Mode:

- If TRGSEL[2:0] = '000', Software trigger event (ACLK)
- If TRGSEL[2:0] = '001', TIMERn (n = 0 to 9) trigger event
- If TRGSEL[2:0] = '010', MPWM0 trigger event
- If TRGSEL[2:0] = '011', MPWM1 trigger event

The SEQCNT[2:0] in ADCn_MR Register in Single Mode:

- If SEQCNT[2:0] = '000', Single channel sampling

Analog Input Channel in Single Mode:

- If SEQ0CH[4:0] = x (x = 0 to 23), adc_chx channel for A/D conversion

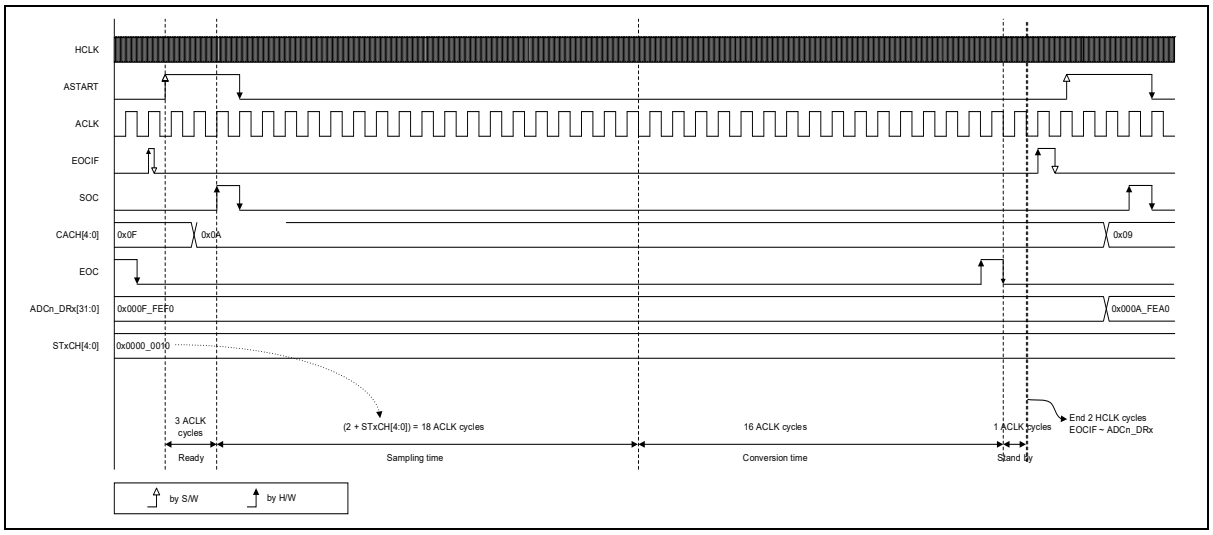
Trigger Event Source Settings:

- If SEQTRG0[3:0] = '0000' to '1111', ADC trigger source

After the options for the ADC operation mode and trigger event sources, A/D conversion channel, and trigger event source are set, once the trigger event starts, sampling for the analog input starts.

Digitized data of the analog inputs is stored in the ADCn_DRx register (n = 0 to 2, x = 0 to 7), and the EOCIF flag in the ADCn_SR register is set to '1'.

Figure 199. ADC Single Mode Timing (When ADMOD[1:0] = '00')



Users can obtain the A/D conversion result of the analog input by polling the EOCIF flag in the ADCn_SR register using software or read-accessing the ADCn_DRx register (n = 0 to 2, x = 0 to 7) at the point of time when the EOCIF flag in the ADCn_SR register interrupt occurs.

18.3.10 Sequential Mode Conversion (ADMOD[1:0] = '00', SEQCNT[2:0] > 0)

Sequential mode is used when the A/D conversion of the analog inputs are sequentially performed by each trigger source. The A/D conversion of the analog inputs is executed as many times as set in the SEQCNT[2:0] bits in the ADCn_MR register.

- If ADMOD[1:0] = '00' and SEQCNT[2:0] > '000', A/D conversion in sequential mode

There are four types of trigger sources in sequential mode:

- If TRGSEL[2:0] = '000', Software trigger event (ACLK)
- If TRGSEL[2:0] = '001', TIMERn (n = 0 to 9) trigger event
- If TRGSEL[2:0] = '010', MPWM0 trigger event
- If TRGSEL[2:0] = '011', MPWM1 trigger event

The SEQCNT[2:0] in ADCn_MR register in sequential mode:

Since the A/D conversion is executed for multiple channels, the SEQCNT[2:0] in ADCn_MR register must be set to a number other than '0'.

- If SEQCNT[2:0] = '001' to '111', Multi-channel sampling in sequential mode

Analog input channel in sequential mode:

The number of sequence channels can be configured is set in the SEQCNT[2:0] in ADCn_MR register.

- If SEQ0CH[4:0] = x (x = 0 to 23), 1st sequence's channel is adc_chx.
- If SEQ1CH[4:0] = x (x = 0 to 23), 2nd sequence's channel is adc_chx.
- If SEQ2CH[4:0] = x (x = 0 to 23), 3rd sequence's channel is adc_chx.
- If SEQ3CH[4:0] = x (x = 0 to 23), 4th sequence's channel is adc_chx.
- If SEQ4CH[4:0] = x (x = 0 to 23), 5th sequence's channel is adc_chx.
- If SEQ5CH[4:0] = x (x = 0 to 23), 6th sequence's channel is adc_chx.
- If SEQ6CH[4:0] = x (x = 0 to 23), 7th sequence's channel is adc_chx.
- If SEQ7CH[4:0] = x (x = 0 to 23), 8th sequence's channel is adc_chx.

Trigger event source settings:

The trigger sources in each sequence can be configured is set in the SEQTRGx[3:0] bits in the ADCn_TRG register. In sequence mode, the 1st sequence has the highest priority, and the 8th sequence has the lowest priority.

- If SEQTRG0[3:0] = x (x = 0 to 15), 1st channel's trigger source is x.
- If SEQTRG1[3:0] = x (x = 0 to 15), 2nd channel's trigger source is x.
- If SEQTRG2[3:0] = x (x = 0 to 15), 3rd channel's trigger source is x.
- If SEQTRG3[3:0] = x (x = 0 to 15), 4th channel's trigger source is x.
- If SEQTRG4[3:0] = x (x = 0 to 15), 5th channel's trigger source is x.
- If SEQTRG5[3:0] = x (x = 0 to 15), 6th channel's trigger source is x.
- If SEQTRG6[3:0] = x (x = 0 to 15), 7th channel's trigger source is x.
- If SEQTRG7[3:0] = x (x = 0 to 15), 8th channel's trigger source is x.

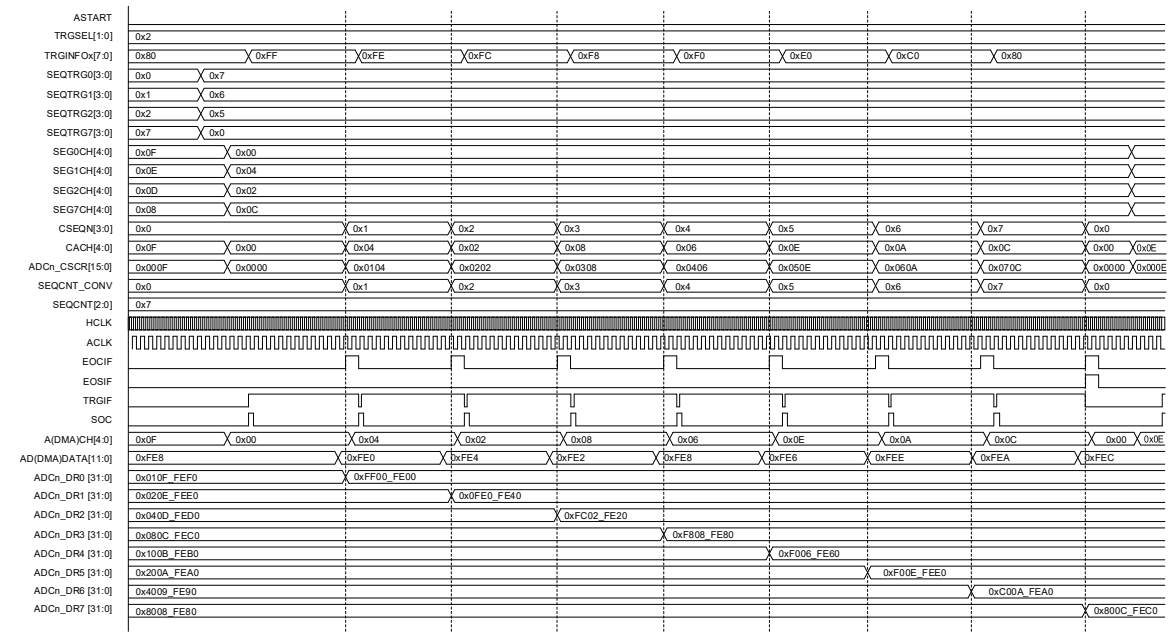
After the options for the ADC operation mode and trigger event sources, A/D conversion channel, and trigger event source are set, once the trigger event starts, sampling for the analog input starts.

Digitized data of the analog inputs is stored in the ADCn_DRx register (n = 0 to 2, x = 0 to 7), and the EOCIF flag in the ADCn_SR register is set to '1'.

The EOCIF flag in the ADCn_SR register is generated as many as the number of sequences set by the SEQCNT[2:0] bits in the ADCn_MR register, and the End-of-Sequence (EOS) flag that indicates the end of the sequence is generated at the time EOC occurs during the last sequence. Users can obtain the digitized data of the analog inputs by polling the EOSIF flag in the ADCn_SR register using software or read accessing the ADCn_DRx register (n = 0 to 2, x = 0 to 7) of the corresponding sequence at the point when the EOSIF interrupt occurs.

Operation in sequential mode is almost identical to those in burst mode. The difference is SOC (Start of Conversion) trigger sources. In sequential mode, each sequence's SOC is triggered by the corresponding SEQTRGx[3:0] (x = 0 to 7) trigger, followed by as many conversions as defined by the SEQCNT[2:0] bits in the ADCn_MR register.

Figure 200. ADC Sequential Mode Timing (When ADMOD[1:0] = '00' and SEQCNT[2:0] > 0)



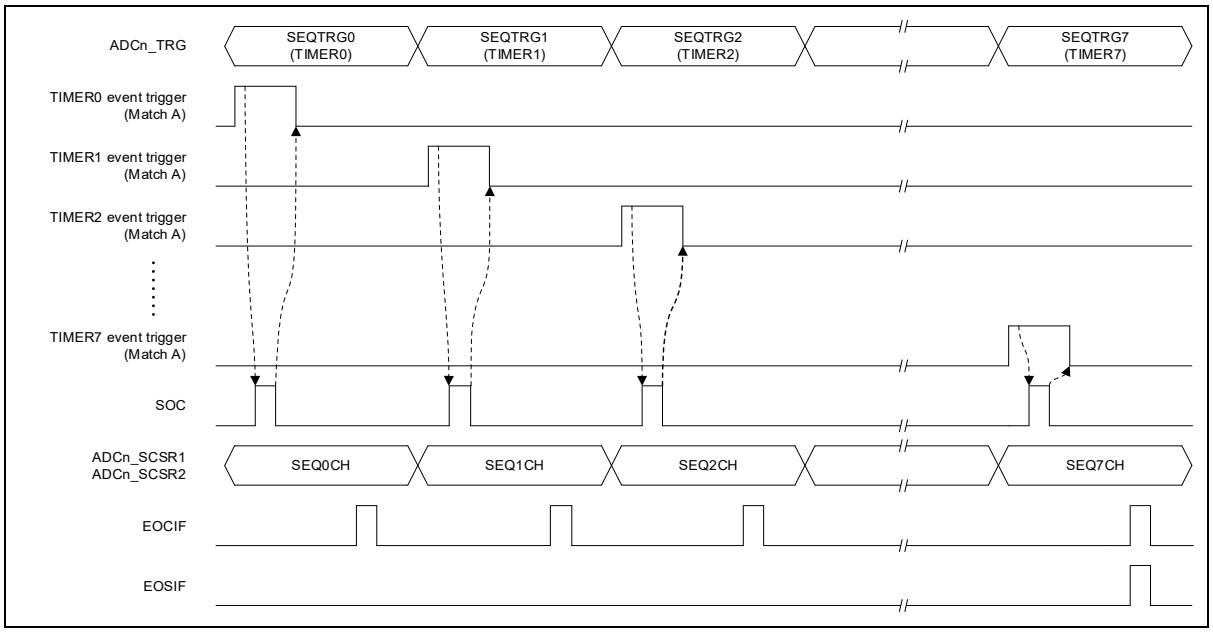
NOTES:

1. HCLK = 200 MHz, ACLK = 10 MHz (CLKDIV = 20)
2. n = 0 to 2, x = 0 to 7

Figure 201 shows a timing diagram that uses 8 sequences in Sequential mode, where each sequence is set to have TIMER trigger source. Based on the sequence' priority, the A/D conversion is sequentially executed for each trigger event.

When the trigger event of each channel occurs, the internal SOC signal is set to 'H' and becomes 'L' when the trigger event ends. Then the EOC flag is generated when the A/D conversion of the corresponding analog input signal is complete. The SOC and EOC occur as many times as set in the SEQCNT[2:0] bits, and the sequence end flag EOS is generated when the EOC occurs during the last sequence.

Figure 201. ADC Trigger Timing in Sequential Mode (SEQCNT[2:0] = '111', 8 Sequential Conversions)



18.3.11 Burst Mode Conversion (ADMOD[1:0] = '01', SEQCNT > 0)

Burst mode is used when the A/D conversion of the analog inputs is performed by one trigger source in the order of channel sequence configuration.

- If ADMOD[1:0] = '01', A/D conversion in burst mode

There are four types of trigger sources in burst mode:

- If TRGSEL[2:0] = '00', Software trigger event (ACLK)
- If TRGSEL[2:0] = '01', TIMERn trigger event (n = 0 to 9)
- If TRGSEL[2:0] = '10', MPWM0 trigger event
- If TRGSEL[2:0] = '11', MPWM1 trigger event

The SEQCNT[2:0] in ADCn_MR Register in Burst Mode:

Since the A/D conversion is executed for the multi-channels, the SEQCNT[2:0] bits in the ADCn_MR register must be set to a number other than '0' in burst mode.

- If SEQCNT[2:0] = '001' to '111', Multi-channel sampling in burst mode

Analog Input Channel in Burst Mode:

The number of sequence channels can be configured is set in the SEQCNT[2:0] in ADCn_MR register. In burst mode, when the trigger event occurs, the A/D conversions are performed in the order starting from SEQ0CH as shown below:

- If SEQ0CH[4:0] = x (x = 0 to 23), 1st sequence's channel is adc_chx
- If SEQ1CH[4:0] = x (x = 0 to 23), 2nd sequence's channel is adc_chx
- If SEQ2CH[4:0] = x (x = 0 to 23), 3rd sequence's channel is adc_chx
- If SEQ3CH[4:0] = x (x = 0 to 23), 4th sequence's channel is adc_chx
- If SEQ4CH[4:0] = x (x = 0 to 23), 5th sequence's channel is adc_chx
- If SEQ5CH[4:0] = x (x = 0 to 23), 6th sequence's channel is adc_chx
- If SEQ6CH[4:0] = x (x = 0 to 23), 7th sequence's channel is adc_chx
- If SEQ7CH[4:0] = x (x = 0 to 23), 8th sequence's channel is adc_chx

Trigger Event Source Settings:

In burst mode, the A/D conversion on the SEQxCH[4:0] multi-channel (x = 0 to 7) is triggered by the trigger source event assigned in the BSTTRG[3:0] bits in the ADCn_TRG register.

If the TRGSEL[2:0] bits in the ADCn_MR register is set for the TIMER or MPWM event triggers, SOC (Start of Conversion) signals are triggered by the BSTTRG[3:0] bits in the ADCn_TRG set trigger.

If the BSTTRG[3:0] bits in the ADCn_TRG register is set for the TIMER 3, for example, the ADC conversion gets started by the TIMER 3 trigger. Once an event is triggered by the BSTTRG[3:0] bits set trigger source, the ADC shifts the ADC channels as many times as set in the SEQCNT[2:0] bits in the ADCn_MR.

- If BSTTRG[3:0] = x (x = 0 to 15), Trigger source of burst mode sequence is x

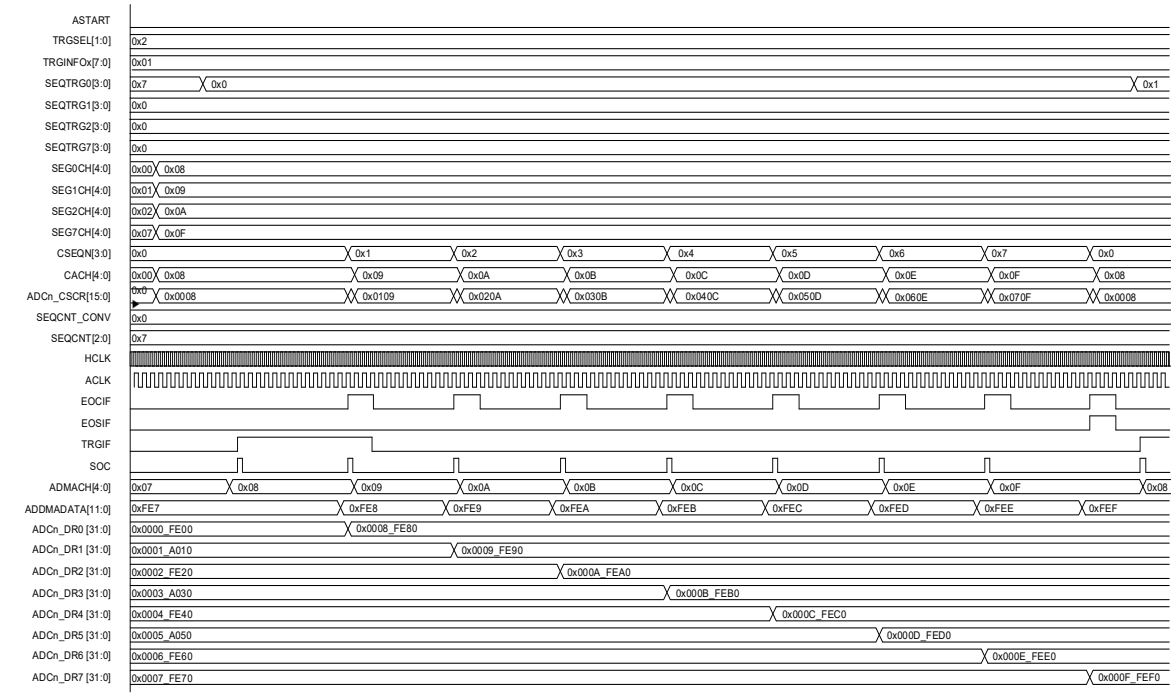
After the options for the ADC operation mode and trigger event sources, A/D conversion channel, and trigger event source are set, once the trigger event starts, sampling for the analog input starts.

Digitized data of the analog inputs is stored in the ADCn_DRx register (n = 0 to 2, x = 0 to 7), and the EOCIF flag in the ADCn_SR register is set to ‘1’.

The EOCIF flag in the ADCn_SR register is generated as many times as set in the SEQCNT[2:0] bits in the ADCn_MR register, and the End-of-Sequence (EOS) flag that indicates the end of the sequence is generated at the time EOC occurs during the last sequence.

Users can obtain the digitized data of the analog inputs by polling the EOSIF flag in the ADCn_SR register using software or read accessing the ADCn_DRx register (n = 0 to 2, x = 0 to 7) of the corresponding sequence at the point when the EOSIF flag in the ADCn_SR register interrupt occurs.

Figure 202. ADC Burst Mode Timing (When ADMOD[1:0] = '01')



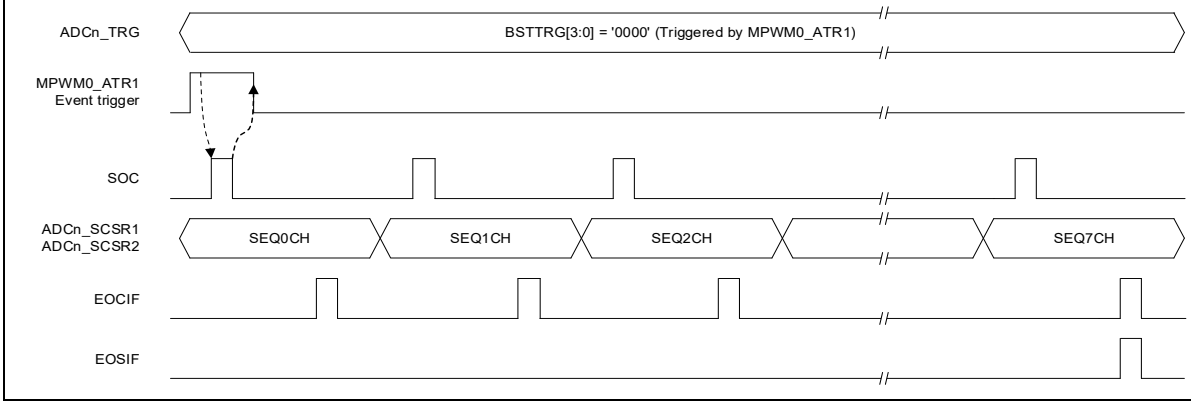
NOTE:

1. HCLK = 200 MHz, ACLK = 10 MHz (CLKDIV = 20)

Figure 203 shows a timing diagram that burst mode continues the A/D conversion for the analog inputs on the 8 channels since the MPWM0_ATR1 trigger event occurs when the BSTTRG[3:0] bits in the ADCn_TRG register is set to MPWM0_ATR1.

When the MPWM0_ATR1 trigger event occurs, the internal SOC signal is set to 'H' and becomes 'L' if the trigger event ends. Then the EOCIF flag in the ADCn_SR register is generated when the A/D conversion is complete. The SOC and EOCIF flags in the ADCn_SR register occur as many times as set in the SEQCNT[2:0] bits in the ADCn_MR register, and the sequence end flag EOSIF flag in the ADCn_SR register is generated when the EOCIF occurs during the last sequence.

Figure 203. ADC Trigger Timing in Burst Mode (SEQCNT[2:0] = '111', 8 Sequential Conversions)



18.3.12 Multiple Mode Conversion (ADMOD[1:0] = ‘10’, SEQCNT[2:0] > 0)

In multiple mode, the A/D conversions are performed in the order in which trigger events occur for each analog input. Since the A/D conversion starts for the analog inputs in the corresponding sequence at the point when the trigger event occurs, the EOS signal indicating end of the sequence is ignored.

- If ADMOD[1:0] = ‘10’, A/D conversion in multiple mode.

There are four types of SOC trigger sources in multiple mode:

- If TRGSEL[2:0] = ‘000’, software trigger event (ACLK).
- If TRGSEL[2:0] = ’001’, TIMERn (n = 0 to 9) trigger event.
- If TRGSEL[2:0] = ‘010’, MPWM0 trigger event.
- If TRGSEL[2:0] = ‘011’, MPWM1 trigger event.

The SEQCNT[2:0] in ADCn_MR register in multiple mode:

Since the A/D conversion is executed for multiple channels, the SEQCNT[2:0] bits in the ADCn_MR register must be set to a number other than ‘0’.

- If SEQCNT[2:0] = ‘001’ to ‘111’, Multi-channel sampling in multiple mode

Trigger event source settings:

The number of trigger sources in each sequence can be configured is set in the SEQCNT[2:0] bits in the ADCn_MR register.

Users can set the desired trigger sources in the SEQTRGx[3:0] bits in the ADCn_TRG register (n= 0 to 2, x = 0 to 7). Each of these sources triggers an SOC when the triggering conditions are met, regardless of the sequence set in the SEQTRGx[3:0] bits in the ADCn_TRG register (n= 0 to 2, x = 0 to 7).

For example, if the SEQCNT[2:0] bits in the ADCn_MR register is set to '11' and four trigger sources are selected from among the TIMER and MPWM trigger sources, setting start bit of trigger source event lets the conversions be triggered by these trigger sources in the order of reception, unlike sequential or burst modes.

- If SEQTRG0[3:0] = x (x = 0 to 15), 1st channel's trigger source is x.
- If SEQTRG1[3:0] = x (x = 0 to 15), 2nd channel's trigger source is x.
- If SEQTRG2[3:0] = x (x = 0 to 15), 3rd channel's trigger source is x.
- If SEQTRG3[3:0] = x (x = 0 to 15), 4th channel's trigger source is x.
- If SEQTRG4[3:0] = x (x = 0 to 15), 5th channel's trigger source is x.
- If SEQTRG5[3:0] = x (x = 0 to 15), 6th channel's trigger source is x.
- If SEQTRG6[3:0] = x (x = 0 to 15), 7th channel's trigger source is x.
- If SEQTRG7[3:0] = x (x = 0 to 15), 8th channel's trigger source is x.

NOTE:

1. Unused trigger source of the SEQTRGx[3:0] bits must be set to 14 for avoid unwanted trigger operation.

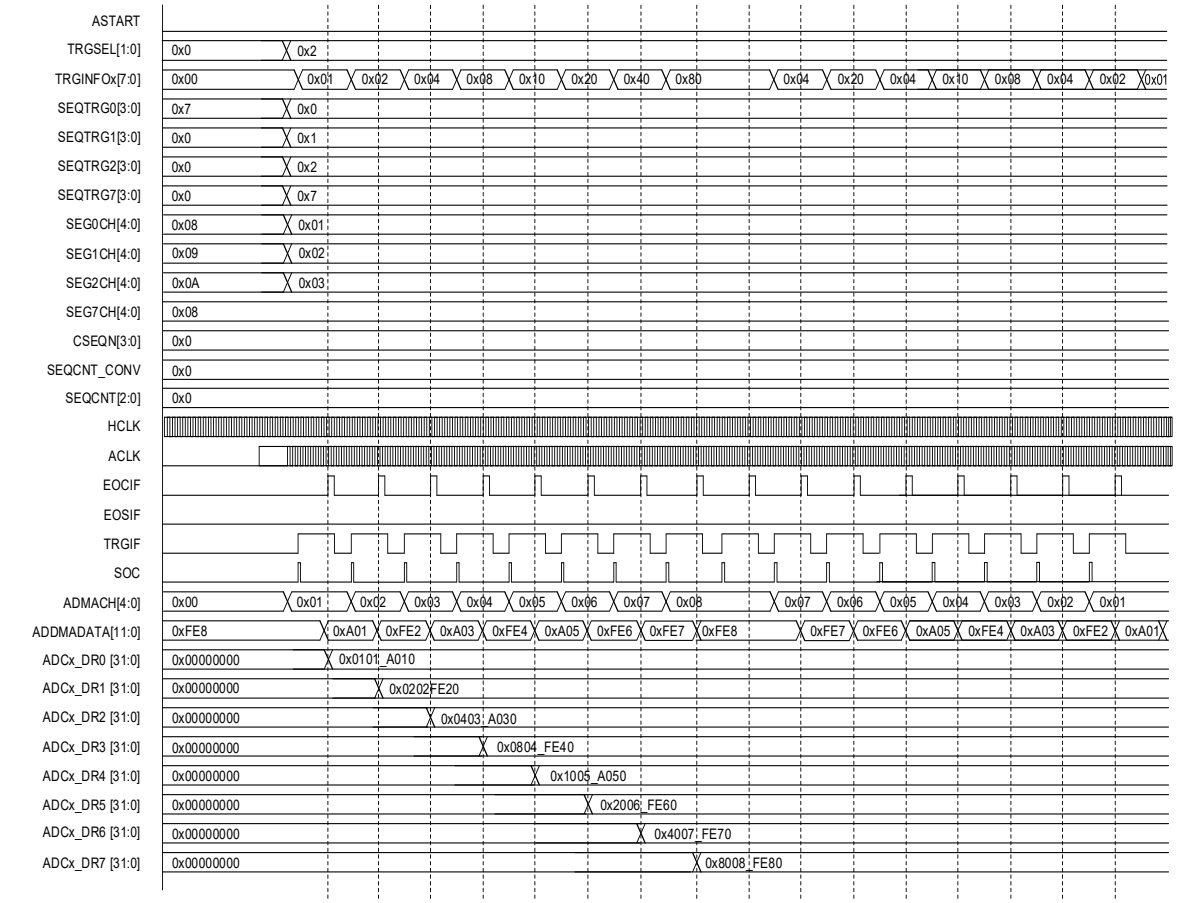
After the options for the ADC operation mode and trigger event sources, A/D conversion channel, and trigger event source are set, once the trigger event starts, sampling for the analog input starts.

Digitized data of the analog inputs is stored in the ADCn_DRx register (n = 0 to 2, x = 0 to 7), and the EOCIF flag in the ADCn_SR register is set to '1'.

The EOCIF flag in the ADCn_SR register is generated as many as the number of sequences set in the SEQCNT[2:0] bits , and the End of Sequence Interrupt Flag (EOSIF) flag in the ADCn_SR register does not occur because the A/D conversions are performed at every trigger source event, regardless of the sequence order.

Therefore, users can obtain the digitized data of the analog inputs by polling the EOCIF flag in the ADCn_SR register using software or read accessing the ADCn_DRx register (n = 0 to 2, x = 0 to 7) of the corresponding sequence at the point when the EOCIF flag in the ADCn_SR register interrupt occurs.

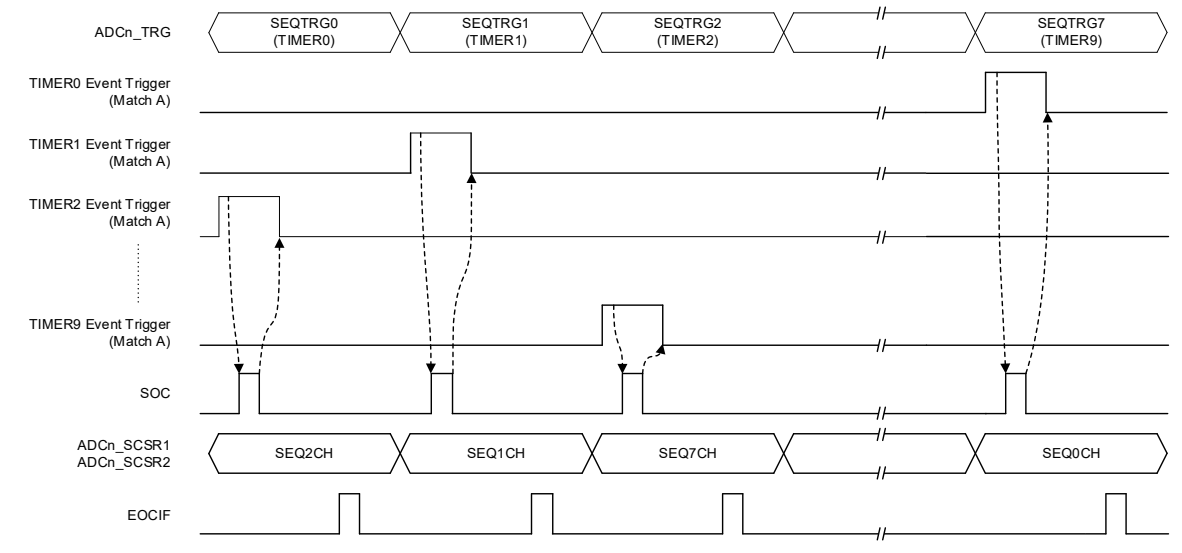
Figure 204. ADC Multiple Mode Timing (When ADMOD[1:0] = '10' and MR.SEQCNT[2:0] > 0)



NOTES:

1. HCLK = 200 MHz, ACLK = 10 MHz (CLKDIV = 20)
2. n = 0 to 2, x = 0 to 7.
3. When ADC operates in Multiple Mode, the channel information of current A/D conversion should be referenced with ACH[4:0] in ADCn_DRx register or ADMACH[4:0] in ADCn_DDR register after enabling DMAEN = '1'

Figure 205. ADC Trigger Timing in Multiple Mode (SEQCNT[2:0] = '111', 8 Multiple Conversions)



18.3.13 Injection Mode Conversion (INJMODE = '1')

In Injection mode, if the ADC conversion is triggered by external events or software triggers, each channel is prioritized and the conversion value is read from high-priority channel first.

- If INJECTMODE = '1', AD conversion in injection mode

There are four types of trigger sources in injection mode:

- If TRGSEL[2:0] = '000', Software trigger event (ACLK)
- If TRGSEL[2:0] = '001', TIMERn (n = 0 to 9) trigger event
- If TRGSEL[2:0] = '010', MPWM0 trigger event
- If TRGSEL[2:0] = '011', MPWM1 trigger event

The CST[4:0] in ADC_IMR register in injection mode:

- If CST[4:0] = '00000' to '11111', Injection channel sampling time setting

Analog input channel in injection mode:

- If CHSEL = x (x = 0 to 23), adc_chx channel for A/D conversion

Trigger event source settings:

- If TRGSRC[3:0] = '0000' to '1111', ADC injection trigger source

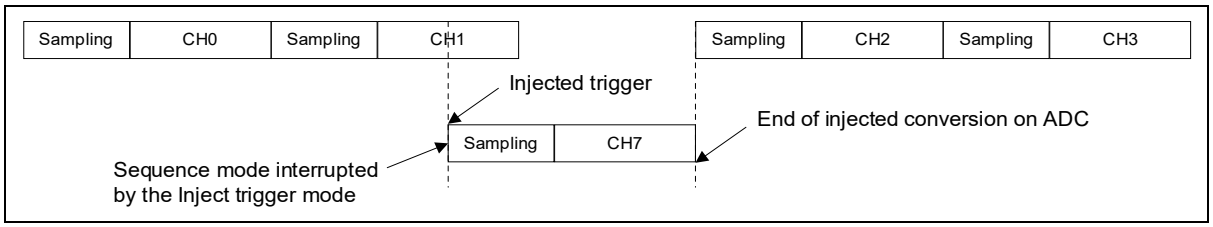
After the options for the ADC injection mode operation and trigger event sources, A/D conversion channel, and trigger event source are set, once the trigger event starts, sampling for the analog input starts.

Digitized data of the Injection mode analog inputs is stored in the ADC_IDR register, and the IOCIF in ADC_SR register is set to '1'.

Users can obtain the AD conversion result of the analog input by polling the IOCIF in ADC_SR register using software or read-accessing the ADC_IDR register at the point of time when the IOCIF in ADC_SR register interrupt occurs.

If the ADC trigger occurs on the channel set to Injection mode, current ADC sampling is interrupted and new ADC sampling on the channel in Injection mode is forced to start. The channel set to Injection mode has the highest priority for the ADC conversion, and the conversion value is read in order of Injection mode to Sequence channel. Injection mode is mainly used in motor control applications. It prevents noise due to the transistor switching operation from affecting ADC measurement and causing generation of incorrect conversion values.

Figure 206. Example of Injection Mode Conversion



18.3.14 Starting Conversions (ASTART, Start Trigger, ARST)

18.3.14.1 Start A/D Conversion (ASTART)

The A/D conversion starts when the ASTART bit in the ADCn_CR register is set to ‘1’ by software as described below:

- If ASTART = 0, ADC conversion does not start. (software trigger)
- If ASTART = 1, ADC conversion starts. This bit is cleared at the next ADC clock cycle. (software trigger)

18.3.14.2 Start All AD Conversion (AASTART)

AASTART is a function that can start ADC0, ADC1, and ADC2 simultaneously. Before performing AASTART, the EXTADCRQEN bit (ADC_MR) of ADC0, ADC1, and ADC2 must be '1' to proceed with start. If it is '0', the corresponding ADC does not operate.

Below shows ADC operation according to the combination of EXTADCRQEN and AASTART.

- If EXTADCRQEN = 0
  - AASTART = 0, ADCn conversion does not start. (software trigger)
  - AASTART = 1, ADCn conversion does not start. (software trigger)
- If EXTADCRQEN = 1, ADC0 and ADC1,2 simultaneous start is enabled.
  - AASTART = 0, ADCn conversion does not start. (software trigger)
  - AASTART = 1, ADC0 and ADC1,2 simultaneous conversion starts. This bit is cleared at the next ADC clock cycle. (Software Trigger)

18.3.14.3 Reset A/D Conversion (ARST)

If the ARST bit in the ADCn_MR register is set to ‘0’ and the software trigger is used as a trigger source, users must set the ASTART bit in the ADCn_CR register to ‘1’ using software to restart the A/D conversion at the end of the A/D conversion sequence.

If the A/D conversions are performed per sequence by the trigger source in Sequential mode, set the ARST bit in the ADCn_MR register to ‘1’ to restart the A/D conversion when the EOSIF flag in the ADCn_SR register is generated at the end of the A/D conversion.

- If ARST = 0, A/D conversion stops at EOS flag. (To restart the A/D conversion, the ASTART must be set to ‘1’).
- If ARST = 1, A/D conversion restarts when proper trigger source occurs after End-of-Sequence (EOS) flag.

18.3.14.4 Start A/D Conversion by Trigger Source

If the ADC timer trigger source is used, software allows the A/D conversion to start when the timer trigger source occurs after timer operating. (Refer to section 18.3.17)

- The TEN = 1 in TIMERn_CR2 (n = 0 to 9)
- The PWMEN = 1 in MPWMn_CR (n = 0 to 1)
- The ASTART = 1 in ADCn_CR (n = 0 to 2)

When the A/D conversion is complete for one channel, the EOCIF flag is generated and the ASTART bit in the ADC_CR register is cleared by hardware automatically.

In sequential, multiple, and burst mode, the EOCIF flags are generated as many channels as set in a sequence, and the EOSIF flag is generated when sampling of the last channel is complete except Multiple mode.

18.3.15 ADC Timing

The A/D conversion time consists of the following:

t_{ADC} = 3 HCLK cycles [delay for synchronizing until SOC event]

+ (2 + STxCH[4:0]) ACLK cycles [sampling time]

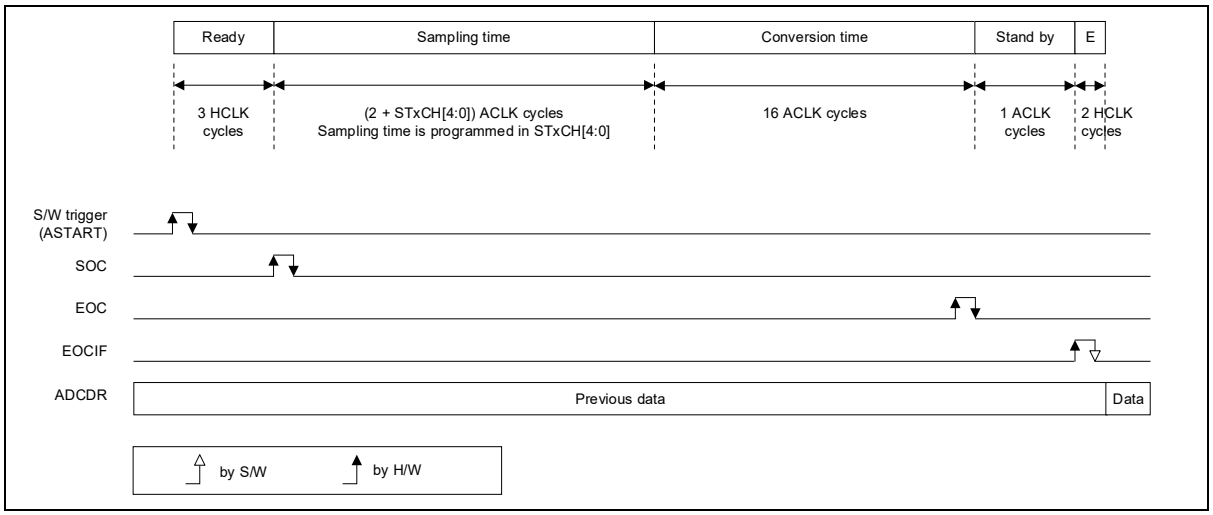
+ 16 ACLK cycles [conversion time]

+ 1 ACLK cycles [EOC flagged time]

+ 2 HCLK cycles [EOCIF flagged time]

= (2 + STxCH[4:0]) ACLK cycles [sampling time] + 17 ACLK cycles + 5 HCLK cycles

Figure 207. Analog-to-Digital Conversion Time



The total conversion time is calculated as follow example:

- With f_{ACLK} = 20 MHz and a STxCH[4:0] = 2, sampling time is 4 ACLK cycles
- Then, t_{ADC} = { (2 + 2) + 16 + 1 } ACLK cycles = 21 ACLK cycles = 1.05 µs

ADC sampling timing adjustment is required for high-speed A/D conversion of analog input.

For high-speed analog input A/D conversion, it is necessary to adjust the A/D conversion timing for one sample by setting the ADC operation clock and external resistors R_{AIN} connected to the analog input and STxCH[4:0] bits in the ADCn_CSTRx register value. Table 180 shows the recommended values for R_{AIN} and STxCH[4:0] bits in the ADCn_CSTRx register for A/D conversion when the ADC's reference voltage is 5V and ACLK 42 MHz.

Table 180. ADC Frequency for Values of R_{AIN}.

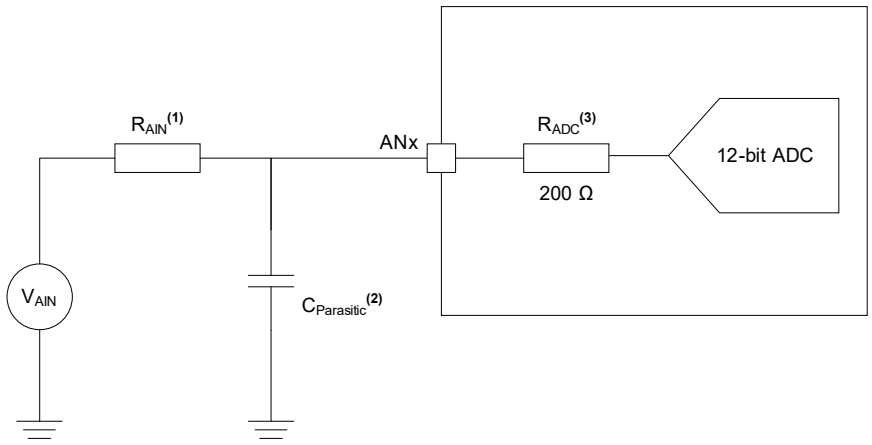
R_{AIN} Max. (kΩ)	t_{CONV} [cycle]	t_{CONV} [µs]	STxCH[4:0]	t_{SMPL} [cycle]	t_{SMPL} [µs]	f_{ADC} [Msps]
0.01	16	0.38	9	11	0.26	1.50
0.1	16	0.38	9	11	0.26	1.50
0.3	16	0.38	9	11	0.26	1.50
0.5	16	0.38	9	11	0.26	1.50
1.0	16	0.38	9	11	0.26	1.50
1.5	16	0.38	9	11	0.26	1.50
2.0	16	0.38	9	11	0.26	1.50
4.0	16	0.38	15	17	0.40	1.24
5.0	16	0.38	19	21	0.50	1.11

NOTES:

1. The parameters are guaranteed by design.
2. f_{ADC} is an A/D conversion rate from SOC to EOC.
3. See Figure 208 for how to design the R_{AIN} resistor circuit for the ANx analog input.

The circuit of the analog input ANx for ADC to which the source is supplied can be designed as follows. When using I/O set to analog input mode, the ratio of the internal resistance R_{ADC} to the external resistance R_{AIN} has to be considered when inputting an external analog source.

Figure 208. Typical Connection Diagram Using ADC



NOTES:

1. Refer to Table 180 for more information of the ADC frequency.
2. $C_{Parasitic}$ represents the capacitance of the PCB (Dependent on soldering and PCB layout quality) plus the pad capacitance. A high $C_{Parasitic}$ value downgrades conversion accuracy. To remedy this, f_{ADC} should be reduced.
3. R_{ADC} is the internal resistance value when the I/O pin is set to ANx, which is an analog input state.

18.3.16 Stopping Ongoing Conversion (ASTOP)

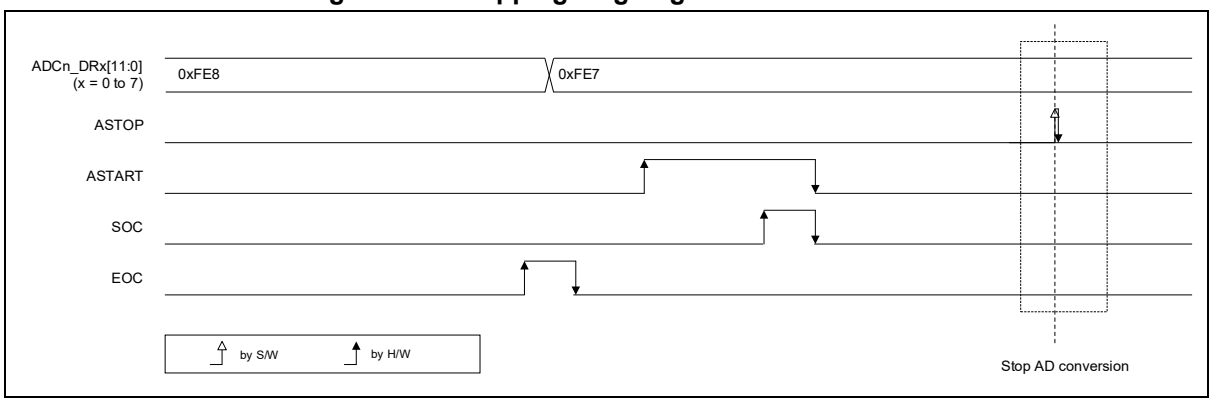
The A/D conversion in progress can be stopped by setting the ASTOP bit in the ADCn_CR register to '1' using software. When the A/D conversion stops, the ADC operation in progress is reset. The ADC can be reconfigured (e.g., channel selection, trigger change) for new input data.

Once the ASTOP bit in the ADCn_CR register is set to '1' by software, all A/D conversions in progress stop and partial results are deleted. (However, the ADCn_DRx (n = 0 to 2, x = 0 to 7) register is not updated due to this deletion.)

During the sequence conversion of the multi-channel, if the ASTOP bit in the ADCn_CR register is set to '1', then the A/D conversion of a sequence stops.

When this procedure is completed, hardware clears the ASTOP bit in the ADCn_CR register, and software needs to clear the EOCIF flag in the ADCn_SR register.

Figure 209. Stopping Ongoing A/D Conversions



18.3.17 Conversion on External Trigger (TRGSEL[2:0], SEQTRGx[3:0], TRGIE, TRGIF)

The A/D conversion can be triggered by the software or the events of the TIMER module and MPWM module (Timer capture with input pin or timer match A interrupt, etc.). When setting the trigger source, it is recommended to stop the A/D conversion by setting the ASTART bit to '0'.

18.3.17.1 ADC Trigger Source Selection (TRGSEL[2:0])

Users can select a trigger source by configuring the TRGSEL[2:0] in ADC_MR register as described in Table 181.

Table 181. External Trigger Source Configuration

TRGSEL[2:0]	Trigger Source	Description
000	ASTART	Disables event triggers and enables the soft trigger only.
001	TIMERn (n = 0 to 9)	Enables the timer event trigger and soft trigger.
010	MPWM0	Enables the MPWM0 event trigger and soft trigger.
011	MPWM1	Enables the MPWM1 event trigger and soft trigger.

To use TIMER as the ADC trigger source, select the event and polarity of the TIMER channel, then set the ADCTRGEN in TIMER_CR1 register bit to '1'. (n = 0 to 9).

To use the MPWM as the ADC trigger source, configure the MPWM_ATRx register to set a channel, an event, and the point of time for trigger updates. Then, set the corresponding ATRxIE bit in the MPWM_IER register of the event channel to '1' to enable the interrupt. Refer to section 17.6.21 and 17.6.14.

18.3.17.2 ADC Trigger Channel Settings (SEQTRGx[3:0] and TRGSRC[3:0])

Once a type of the trigger source is determined, up to eight trigger sources (As a single trigger source or a sequence of trigger sources per an ADC unit) can be selected by configuring the ADC_TRG register. If a trigger source for a single A/D channel is used, users can use the SEQTRG0[3:0] bits in the ADC_TRG register to select a trigger source.

For each sequence sampling in Sequential/Burst/Multiple mode, assign the corresponding trigger source for the A/D conversion channel using the SEQTRGx[3:0] bits in the ADC_TRG register (x = 0 to 7).

By configuring the TRGSRC[3:0] bits in the ADC_IMR register, users can select one from four types of trigger sources in Inject mode.

Table 182. Trigger Source Table

Trigger Source (SEQTRGx)	TIMER (TRGSEL[1:0] = '01')	MPWM (TRGSEL[1:0] = '10')	MPWM (TRGSEL[1:0] = '11')
0	TIMER 0	MPWM0_ATR1	MPWM1_ATR1
1	TIMER 1	MPWM0_ATR2	MPWM1_ATR2
2	TIMER 2	MPWM0_ATR3	MPWM1_ATR3
3	TIMER 3	MPWM0_ATR4	MPWM1_ATR4
4	TIMER 4	MPWM0_ATR5	MPWM1_ATR5
5	TIMER 5	MPWM0_ATR6	MPWM1_ATR6
6	TIMER 6	BOTTOM	BOTTOM
7	TIMER 7	PERIOD	PERIOD
8	TIMER 8	-	-
9	TIMER 9	-	-
10	-	-	-
11	-	-	-
12	-	-	-
13	-	-	-
14	-	-	-
15	ASTART(1)	ASTART(1)	ASTART(1)

NOTE:

1. The ASTART bit in the ADC_CR register is a software trigger.

18.3.17.3 ADC Trigger Start Control

If a trigger event occurs after the trigger source's timer operation starts, A/D conversion is performed by trigger source.

Table 183. Start Control of AD Trigger Conversion

TRGSEL[1:0]	Start Control Bit
ASTART	• ASTART = '1' in ADC_CR register • ASTART = '1' in ADC_ICR register
TIMERn (n = 0 to 9)	TEN = '1' in TIMER_CR2 register
MPWMn (n = 0 to 1)	PWMEN = '1' MPWM_CR1 register

18.3.17.4 ADC Trigger Interrupt and Flag (TRGIE, TRGIF)

Interrupts for the ADC trigger events can be enabled by configuring the TRGIE bit in the ADCn_IER register as shown below:

- If TRGIE = 0, Trigger source event interrupt is disabled.
- If TRGIE = 1, Trigger source event interrupt is enabled.

When the ADC trigger interrupt is enabled, the TRGIF flag is generated at the point of time when an ADC trigger source event of each A/D conversion channel is generated. The TRGIF flag is cleared by writing '1' to this bit.

Although the trigger interrupt of MPWM or TIMER is not enabled, TRGIF flag of ADC is generated when trigger source event has happened. At this time, If the TRGIE bit was enabled, a trigger event interrupt is generated. If the trigger event occurs while TRGIE bit disabled, the TRGIF bit can be checked by polling with software.

18.3.18 End-of-Conversion, End-of-Sampling Phase (EOC)

The ADC notifies user application at the end of A/D conversion (EOC) event. For the polling operation, set the ASTART bit to ‘1’ and poll until the EOCIF bit is set to ‘1’. Then read access the ADCn_DDR or ADCn_DRx register to obtain the A/D conversion data.

For the use of interrupts, setting the EOCIE bit in the ADCn_IER register to ‘1’ generates an interrupt when the EOCIF bit in the ADCn_SR register is set to ‘1’ after the A/D conversion is completed. Then read access the ADCn_DDR or ADCn_DRx (n = 0 to 2, x = 0 to 7) register to obtain the A/D conversion data. The EOCIF flag in the ADCn_SR register is cleared by writing ‘1’ to this bit using software.

NOTE:

1. The initial state of the End-of-Conversion (EOC) of the ADC is unknown status when the microcontroller is first powered on. In this case, leakage current may flow if the microcontroller enters SLEEP mode or DEEP-SLEEP mode. To avoid this situation, set the ASTART bit in the ADCn_CR register to ‘1’ and perform two dummy transfers of the ADCn_DRx (n = 0 to 2, x = 0 to 7) register to clear the EOC. And clear also the EOCIF flag in the ADCn_SR register. Users must consider this situation when initializing the ADC, since it cannot be processed by hardware.

18.3.19 End-of-Sequence Conversion (EOSIE, EOSIF)

The ADC module allows the sequence setup for the A/D conversion of analog inputs in multi-channels. A sequence can be configured with up to 8 channels using the SEQCNT[2:0] bits in the ADCn_MR register as shown below:

- If SEQCNT[2:0] = '000': 1 Sequence ADC (single mode conversion).
- If SEQCNT[2:0] = '001': 2 Sequences ADC.
- If SEQCNT[2:0] = '010': 3 Sequences ADC.
- If SEQCNT[2:0] = '011': 4 Sequences ADC.
- If SEQCNT[2:0] = '100': 5 Sequences ADC.
- If SEQCNT[2:0] = '101': 6 Sequences ADC.
- If SEQCNT[2:0] = '110': 7 Sequences ADC.
- If SEQCNT[2:0] = '111': 8 Sequences ADC.

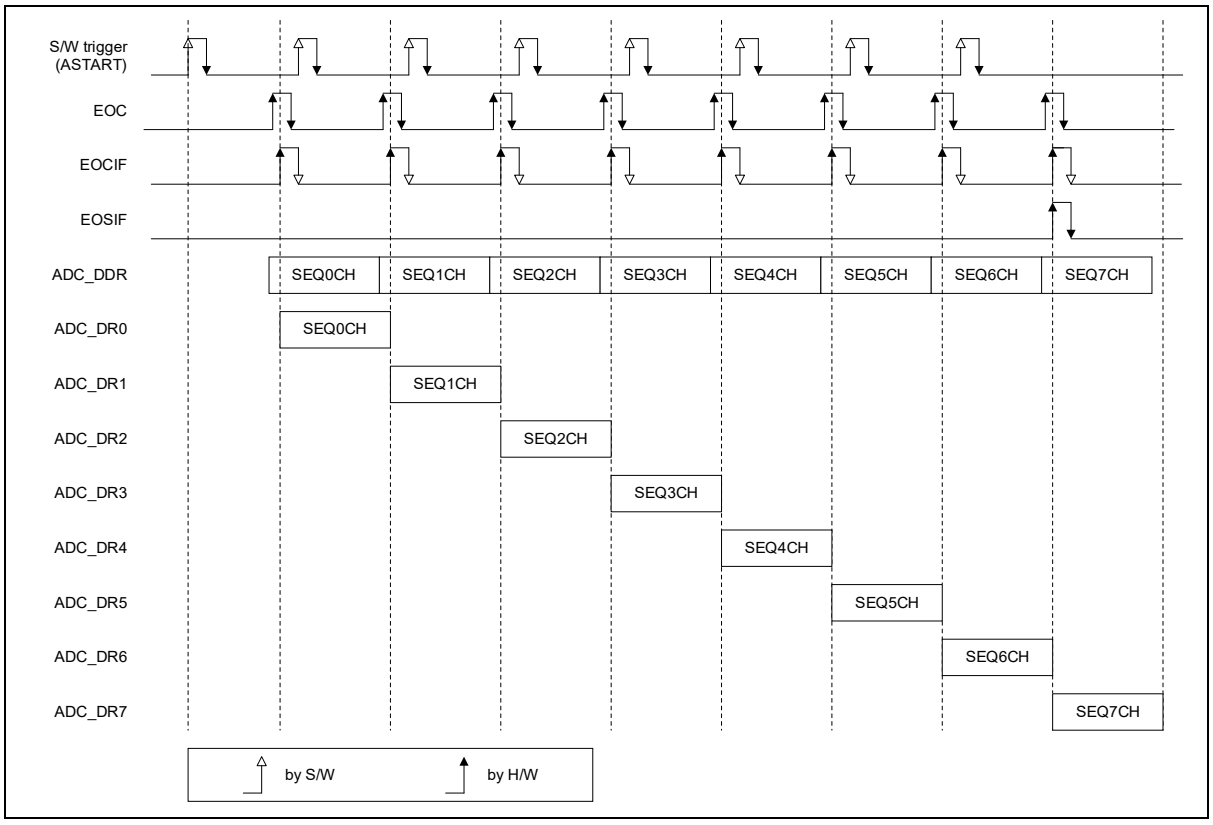
After setting the number of channels, users can assign the channels that are performed in each sequence using the ADCn_SCSR1 register (SEQ0CH to SEQ3CH) and the ADCn_SCSR2 (SEQ4CH to SEQ7CH) register. Refer to section 18.3.7.

Sequences in sequential mode are performed by following the ascending order of the sequence trigger source number set in the SEQTRGx[3:0] bits in the ADCn_TRG register. In the Multiple mode, the sampling order is determined by the timing of each trigger source's event.

Users can adjust the trigger timing to set the sampling period for the ADC sequence. Refer to section 18.3.17.

For the ADC sequence conversion interrupt operation, enable the EOSIE bit in the ADCn_IER register. If the EOSIF = '1', the A/D conversion is performed for the specified channel of a sequence. When the A/D conversion is completed for each channel, the EOCIF flag in the ADCn_SR register is set to '1'.

Figure 210. Example of A/D Conversion Timing in Sequence Mode



18.3.20 Data Management and Information

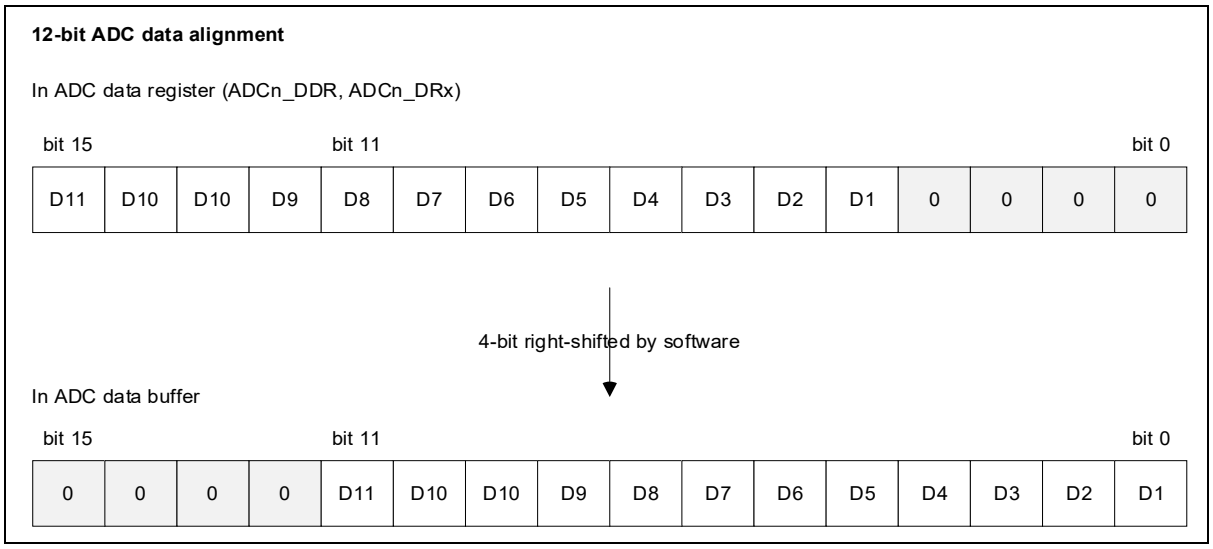
18.3.20.1 Data Register and Data Alignment

At the EOC event of each A/D conversion channel, the result is stored in ADC data registers such as ADCn_DDR, ADCn_DR0, ADCn_DR1, … and ADCn_DR7.

- ADCn_DDR: This register stores the A/D data value of a channel where the A/D conversion was completed most recently.
- ADCn_DRx: This register stores the A/D conversion result of a channel set in each sequence.

As shown in Figure 211, the A/D conversion results are left-aligned in the lower 16 bits of the data registers and then shifted by software to the right by 4 bits, generating a 12-bit value:

Figure 211. Left Alignment (Unsigned Value)



18.3.20.2 Channel and Trigger Information

Users can get information of the A/D conversion data channel and the trigger sources by checking the upper 16 bits of the ADC data registers.

- The A/D conversion channel information to the ACH[4:0] bit field in the ADC_DRx register.
- The trigger source information for the channel where the A/D conversion is performed to the TRGINFOx (x = 0 to 7) bit in the ADC_DRx register.

The TRGINFOx bit in the ADC_DDR register (x = 0 to 7) show that the lower the channel number set in a sequence, the higher the channel priority and is executed first.

Figure 212 shows the configuration of the channel and trigger information in the ADC_DDR and ADC_DRx registers. As shown in the figure, the SEQxCH (x = 0 to 7) sequence is configured with eight channels adc_ch0 to adc_ch7, allow users to check the channel and trigger information in the in the ADC_DDR and ADC_DRx registers.

Figure 212. Channel and Trigger Source Information of A/D Conversion Sequence

	SEQ0CH	SEQ1CH	SEQ2CH	SEQ3CH	SEQ4CH	SEQ5CH	SEQ6CH	SEQ7CH
	adc_ch0	adc_ch1	adc_ch2	adc_ch3	adc_ch4	adc_ch5	adc_ch6	adc_ch7
ACH[4:0] (Channel number of AD conversion)	0x0	0x1	0x2	0x3	0x4	0x5	0x6	0x7
TRGINFO[7:0] (Trigger source priority of AD conversion)	0x01	0x02	0x04	0x08	0x10	0x20	0x40	0x80

18.3.21 Trigger Pending Function in Sequential Mode

In sequential mode of the ADC module, the SEQTRGT bit in the ADC_MR determines how the ADC processes trigger events when multiple analog input channels are operated in sequential mode.

For example, when two analog input channels are operated in sequential mode, the method of processing trigger events is different depending on the value of the SEQTRGT bit.

- SEQTRGT is set to 1, if a trigger that does not match the current sequence occurs, the sequence that matches the trigger is ignored. For example, if the 6th trigger event occurs before the 5th trigger event occurs, the 6th trigger event will be ignored. If the 5th trigger event does not occur, it will wait forever.
- SEQTRGT = 0, if a trigger that does not match the current sequence occurs, the generated trigger is suspended without being ignored. For example, if the 6th trigger event occurs in the sequence in which the 5th trigger event occurs, the 6th trigger is recognized as having occurred and is pending. After that, the 5th trigger event occurs and processes the 5th sequence, followed by the 6th sequence.

18.3.22 ADC Low-Power Modes

Table 184. Effect of Low-Power Mode on ADC

Mode	Description
SLEEP	No effect, ADC used with DMA
DEEP-SLEEP	• The ADC module control is reset in DEEP-SLEEP mode. • ADC module must be reinitialized after waking up from DEEP-SLEEP mode.

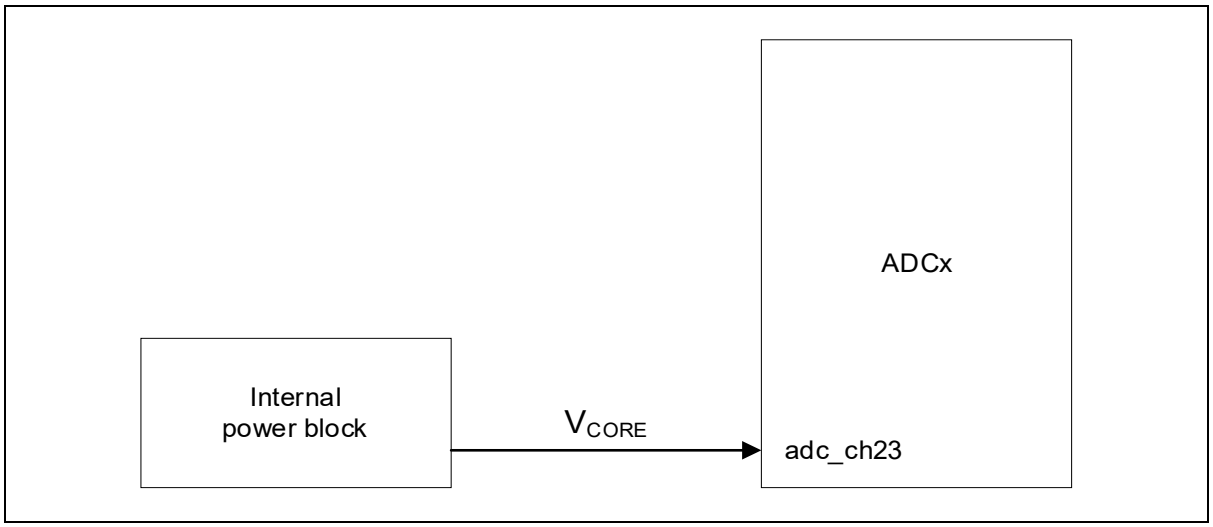
NOTE:

1. Applications using ADC modules must clear the EOC at initialization, when using DEEP-SLEEP mode. Please refer to 18.3.18.

18.3.23 Monitoring Internal Voltage Reference

It is possible to have reference points for the evaluation of external analog input voltage level with monitoring the internal voltage reference V_{CORE}. The analog signals of the internal voltage references V_{CORE} outputting Typ. 1.0 V is entered into the adc_ch23 of all ADC unit.

Figure 213. Channel Block Diagram of V_{CORE}



18.3.24 A/D Conversion Using DMA

The ADC module sends a request for data transfer to the DMA module when the A/D conversion of the analog inputs is completed and the EOCIF flag in the ADCn_SR register is set to ‘1’. The DMA module transfers data to the specified memory area upon receiving this request.

To use the A/D conversion functionality, enable the ADC and DMA using the SCU and then activate the DMAEN bit in the DMAn_SR register to ‘1’ with the ADEN bit in the ADCn_MR register to ‘1’ as listed below:

- If DMAEN = ‘0’, ADC DMA disable.
- If DMAEN = ‘1’, ADC DMA enable.

To connect the ADC to the DMA channel, set the PERISEL[5:0] bits in the DMA_CR register to select the ADC, referring to the number below:

- PERISEL[5:0] = ‘001101’ (0xD, Selected ADC module of DMA transferring) in DMA_CR register.

Set the SIZE[1:0] bits in the DMAn_CR register to half-word for transferring DMA data of the A/D conversion data, and write the number of DMA data to the TRANSCNT[11:0] bits in the DMAn_CR register (n = 0 to 15). For example, if users want to transfer 64 A/D conversion data values to the DMA, the DMA data width is set to 16-bit, and the number of the DMA data transfers is set to 64 as shown below:

- SIZE[1:0] = ‘01’ (half-word, 16-bit DMA data width)
- TRANSCNT[11:0] = 0x40 (64 half-word DMA data)

Set the register address of an ADC unit that to be transferred by DMA to the PAR[15:0] bits in the DMAn_PAR register and the DMA transfer’s destination memory (SRAM) address in the MAR[15:0] bits in the DMAn_MAR register.

- PAR[15:0] = 0x202C (0x4101_202C, ADCn_DDR register address)
- MAR[15:0] = 0x3000 (0x2000_3000, buffer data address)

The direction of the DMA transfer is determined based on the memory and can be controlled by setting the DIR bit in the DMAn_CR register. For example, to store the A/D conversion results in memory, users must set the DIR = ‘1’.

- If DIR = ‘0’, Memory to Peripheral (Tx)
- If DIR = ‘1’, Peripheral to Memory (Rx)

The ADC DMA interrupt can be controlled by setting the DMAIE bit in the ADCn_IER.

- If DMAIE = ‘0’, Disable ADC DMA-done interrupt
- If DMAIE = ‘1’, Enable ADC DMA-done interrupt

When the DMAEN bit in the DMAn_SR is set to ‘1’, the DMA is enabled, and the A/D conversion results are transferred to the address specified by the DMA controller using software. The value in the TRANSCNT[11:0] bits in the DMAn_CR register is decremented by one each time a DMA transfer of a specified size is completed.

When the DMA transfer is completed by the data size requested to the DMA, and the TRANSCNT[11:0] bits in the DMAn_CR register is set to ‘0’, the interrupt is generated with the EOT flag in the DMAn_SR register of the DMA module and the DMAF flag in the ADCn_SR register indicating the DMA data transfer is done of the ADC module.

- If DMAF = 0, There is DMA data to transfer.
- If DMAF = 1, There is no DMA data to transfer. The DMA transfer is complete.
- The DMAF is cleared by writing ‘1’ to this bit.

The EOT flag in the DMAn_SR register of the DMA module is enabled when the transmission is complete and the TRANSCNT[11:0] value becomes 0. The EOT flag is cleared by writing a number larger than zero to the TRANSCNT[11:0] bits in the DMAn_CR register.

The A/D conversion results before the generation of the EOCIF can be referenced only by accessing the ADCn_DRx and ADCn_DDR registers in the ADC module, since the latest results are always updated in the registers.

By using DMA functionality, users can store the A/D conversion results as much as the size set by software and prevent loss of already stored data.

18.3.25 ADC Overrun (OVR) in DMA Mode

The DMA data transfer requests and ACK signals between the ADC and the DMA modules occur in the following order:

1. When the ADC completes the A/D conversion and the EOCIF flag in the ADCn_SR register is set to ‘1’, the ADC sends a data transfer request to the DMA.
2. The DMA transfers the A/D conversion results to the specified SRAM memory address and sends the ACK signal to the ADC.
3. When the TRANSCNT[15:0] bits in the DMAn_CR register value becomes ‘0’, the EOT flag in the DMAn_SR register signal is sent to the ADC module to inform that all DMA data transmissions are complete.

When the DMA does not complete the transmission yet and cannot send the ACK signal to the ADC, if it receives another transfer request from the ADC, the DMA overrun flag (DOVRUN) in the ADCn_SR register) is set to ‘1’. The DOVRUN does not support interrupt function.

If the overrun flag is generated due to the fact that the DMA cannot process the DMA transfer requests on time, the ADC stops requesting transfer to the DMA and the DMA does not transfer new ADC conversion results. In other words, all data transferred to SRAM can be considered valid. The DMA transfer request is blocked until software clears the DOVRUN flag in the ADCn_SR register.

Even if the overrun flag is generated, the ADC maintains operating. Unless software sets the ASTOP bit in the ADCn_CR register to ‘1’ to stop and reset the ADC sequence, the ADC continues to perform the A/D conversions to store the latest results in the ADCn_DDR register until the EOCIF flag in the ADCn_SR register is generated.

18.3.26 Comparison of ADC Sensing Value

The ADC has a built-in comparator that compares the sampling results of the AD inputs and software set values.

Users can compare the value set by CVAL[11:0] with the value A/D conversion channel specified by CCH[4:0] based on the VDD reference voltage.

Set the CCH[4:0] bits in the ADCn_CMPRx register to select a target ADC channel for comparison and enter the comparison value in the CVAL[11:0] bits in the ADCn_CMPRx register.

- If CMPEN = 0, ADC comparator operation is disabled.
- If CMPEN = 1, ADC comparator operation is enabled.

Figure 214. Block Diagram of ADC Internal Comparator

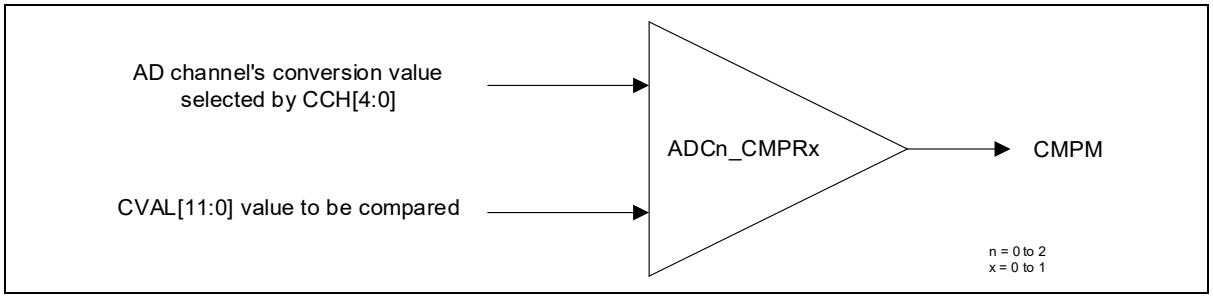


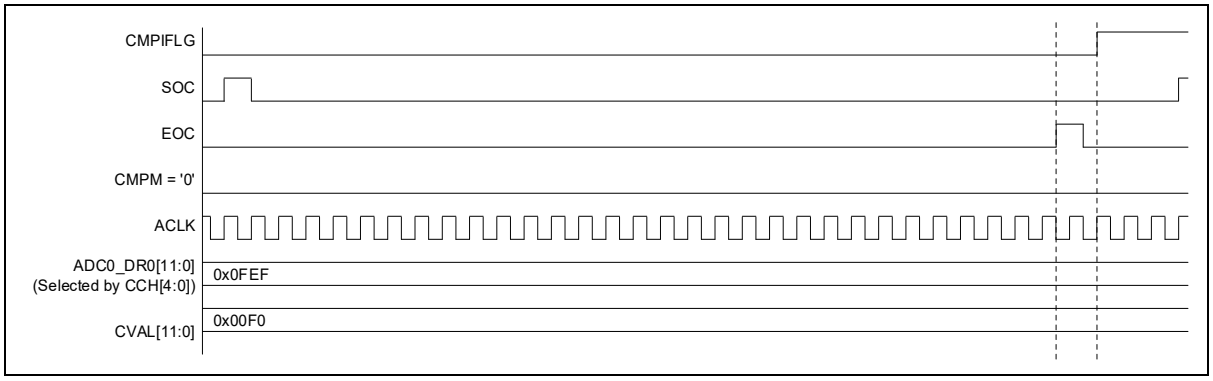
Table 185 shows the conditions under which the CMPIFLG flag occurs by comparing the CVAL[11:0] and the ADC channel values set in the CCH[4:0] bits in the ADCn_CMPRx register according to the CMPM bit in the ADCn_CMPRx register.

Table 185. Comparison and Flag Generation Conditions by CMPM

LTE Bit	Comparison Condition	CMPIFLG Occurrence Condition
0	ADC value of CCH[4:0] channel > CVAL[11:0]	1
0	ADC value of CCH[4:0] channel ≤ CVAL[11:0]	0
1	ADC value of CCH[4:0] channel > CVAL[11:0]	0
1	ADC value of CCH[4:0] channel ≤ CVAL[11:0]	1

Since the ASTART bit in the ADCn_CR register is set to ‘1’, when the EOCIF flag in the ADCn_SR register indicating the A/D conversion completion is set to ‘1’, two input signals are compared internally and the CMPIFLG flag in the ADCn_SR register indicating the comparison completion is generated (CMPIFLG = ‘1’).

Figure 215. Timing Diagram of Comparison between A/D Conversion Results and Reference Value



The comparison results based on the CMPM bit in the ADCn_CMPRx register value affect the flag setting condition of the CMPIFLG flag in the ADCn_SR register as shown below:

- If CMPM = 0, the interrupt is generated when the A/D conversion result is larger than the comparison value (CVAL[11:0]).
- If CMPM = 1, the interrupt is generated when the A/D conversion result is less than or equal to the comparison value (CVAL[11:0]).

The interrupts for the comparison results can be enabled by setting the CMPIEN bit in the ADCn_CMPRx register to ‘1’ as shown below:

- If CMPIEN = 0, ADC compare interrupt is disabled.
- If CMPIEN = 1, ADC compare interrupt is enabled.

18.4 ADC Interrupts

For each ADC unit, an interrupt can be generated by setting below:

- End-of-Single data conversion (EOCIE)
- End-of-Sequence of conversion (EOSIE)
- Interrupt by trigger source (TRGIE)
- End-of-DMA transfer of A/D conversion data (DMAIE)

Each interrupt enable bits are available for flexibility.

Table 186. Control Bits and Event Flags for ADC Interrupts

Interrupt Event	Event Flag	Enable Control Bit
End of conversion of single data	EOCIF	EOCIE
End of sequence conversion	EOSIF	EOSIE
End of conversion by timer trigger source	TRGIF	TRGIE
End of DMA transfer of ADC conversion data	DMADF	DMAIE
ADC conversion data comparison	CMPxIF	CMPxIE

NOTE:

1. x = 0 and 1

18.5 ADC Registers

The base address of the ADC and register map are described in the following:

Table 187. Base Address of ADC

Name	Base Address
ADC0	0x4101_2000
ADC1	0x4101_2100
ADC2	0x4101_2200

Table 188. ADC Register Map

Name	Offset	Type	Description	Reset value	Reference
ADCn_MR	0x0000	RW	ADC n mode register	0x0000_0000	18.5.1
ADCn_CSCR	0x0004	RO	ADC n current sequence and channel register	0x0000_0000	18.5.2
ADCn_CCR	0x0008	RW	ADC n clock control register	0x0000_0000	18.5.3
ADCn_TRG	0x000C	RW	ADC n trigger select register	0x0000_0000	18.5.4
ADCn_CSTR1	0x0010	RW	ADC n channel sampling time register 1	0x0000_0000	18.5.5
ADCn_CSTR2	0x0014	RW	ADC n channel sampling time register 2	0x0000_0000	18.5.6
ADCn_SCSR1	0x0018	RW	ADC n channel select 1 register	0x0000_0000	18.5.7
ADCn_SCSR2	0x001C	RW	ADC n channel select 2 register	0x0000_0000	18.5.8
ADCn_CR	0x0020	RW	ADC n control register	0x0000_0000	18.5.9
ADCn_SR	0x0024	RW	ADC n status register	0x0000_0000	18.5.10
ADCn_IER	0x0028	RW	ADC n interrupt enable register	0x0000_0000	18.5.11
ADCn_DDR	0x002C	RO	ADC n DMA data register	0x0000_0000	18.5.12
ADCn_DR0	0x0030	RO	ADC n sequence 0 data register	0x0000_0000	18.5.13
ADCn_DR1	0x0034	RO	ADC n sequence 1 data register	0x0000_0000	18.5.13
ADCn_DR2	0x0038	RO	ADC n sequence 2 data register	0x0000_0000	18.5.13
ADCn_DR3	0x003C	RO	ADC n sequence 3 data register	0x0000_0000	18.5.13
ADCn_DR4	0x0040	RO	ADC n sequence 4 data register	0x0000_0000	18.5.13
ADCn_DR5	0x0044	RO	ADC n sequence 5 data register	0x0000_0000	18.5.13
ADCn_DR6	0x0048	RO	ADC n sequence 6 data register	0x0000_0000	18.5.13
ADCn_DR7	0x004C	RO	ADC n sequence 7 data register	0x0000_0000	18.5.13
ADCn_IMR	0x0050	RW	ADC n injection mode register	0x0000_0000	18.5.14
ADCn_ICR	0x0054	RW	ADC n injection control register	0x0000_0000	18.5.15
ADCn_IDR	0x0058	RW	ADc n injection data register	0x0000_0000	18.5.16
ADCn_CMPR0	0x0060	RW	ADC n channel compare register 1	0x0000_0000	18.5.17
ADCn_CMPR1	0x0064	RW	ADC n channel compare register 2	0x0000_0000	18.5.18
ADCn_CD01R	0x0080	RO	ADC 0, 1 unit data register	0x0000_0000	18.5.19
ADCn_CD2R	0x0084	RO	ADC 2 unit data register	0x0000_0000	18.5.20

NOTE:

1. n = 0, 1 and 2

18.5.1 ADCn_MR: ADC Mode Register

ADCn_MR configures the modes of the ADC module. You should set this register prior to all other ADC registers according to the intended use of the ADC module.

ADC0_MR=0x4101_2000, ADC1_MR=0x4101_2100, ADC2_MR=0x4101_2200

31 30 29 28 27 26 25 24	23	22	21 20	19	18	17	16	15 14 13 12	11	10 9 8	7	6	5 4	3	2 1 0
Reserved		EXTADCRQEN	Reserved	CDATA	DMAIDE	DMAEN		Reserved	SEQTRGT	SEQCNT[2:0]	ADEN	ARST	ADMOD[1:0]	Reserved	TRGSEL[2:0]
-		0	-	0	0	0		-	0	000	0	0	00	-	000
-		RW	-	RW	RW	RW		-	RW	RW	RW	RW	RW	-	RW

22	EXTADCRQEN	Conversion start control request bit from another external ADC
	0	Disable
	1	Enable
19	CDATA	Conversion data offset alignment
	0	[15:4] offset alignment
	1	[11:0] offset alignment
18	DMAIDE	Whether the DMA includes the injection data
	0	Does not include.
	1	Includes.
17	DMAEN	DMA enable bit (This bit must be set if ADCEN = 1.)
		If DMA is enabled, each DMA interrupt request is made when the ADC is signaled by the DMA controller that the previous DMA transfer has been completed (the status bit of the ADC_SR register is '1'). For the multiple transfers, the DMA requests is automatically made at every interrupt and conversion completion (The bit setting is valid in Burst mode as well).
11	SEQTRGT	Trigger pending type control in sequential mode (Refer to section 18.3.21.)
	0	Trigger that is not related current count in sequential is pending.
	1	Trigger that is not related current count in sequential is not pending.
10 8	SEQCNT[2:0]	Number of conversions in a sequence If ADMOD[1:0] is 0 and SEQCNT[2:0] is not 0, the CSEQN value increments to the SEQCNT[2:0] value by a trigger event. (The setting of the bit field is valid only in single/Sequential modes.)
	000	1 sequence ADC
	001	2 sequence ADC
	010	3 sequence ADC
	011	4 sequence ADC
	100	5 sequence ADC
	101	6 sequence ADC
	110	7 sequence ADC
	111	8 sequence ADC
7	ADEN	ADC enable bit
	0	Disables
	1	Enables
6	ARST	Setting bit to restart at the end of the sequence
	0	Stops at the end of the sequence. (Set the ASTART to '1' to enable the restart.)

		1	Restart at the end of the Sequency.
5 4	ADMOD[1:0](1)	ADC mode selection bit	
		00	Single or Sequential conversion mode
		01	Burst conversion mode
		10	Multiple conversion mode
		11	Not used
2 0	TRGSEL[2:0]	Trigger selection bit	
		000	Event trigger disable/Software trigger only
		001	Timer event trigger/Software trigger
		010	MPWM0 event trigger/Software trigger
		011	MPWM1 event trigger/Software trigger
		Others	Reserved

NOTES:

1. If ADMOD[1:0] has been set for burst conversion mode, the ADC channels are assigned as BST0CH through BST7CH. Burst mode always begins with BST0CH. (In three-burst mode, for example, the analog inputs to the BST0CH, BST1CH, and BST2CH channels are converted in the order of channel number.)

   If ADMOD[1:0] has been set for Multiple mode, any trigger source set enabled in the ADC_TRG register triggers a start of conversion immediately when its triggering conditions are met, regardless of the sequence setting.

2. If EXTADCRQEN = 1, this bit allows START by another ADC unit.

18.5.2 ADCn_CSCR: ADC Current Sequence/Channel Register

ADC_CSCR displays the ADC's current sequence and channel. It is comprised of current sequence number bits and current active channel number bits. The CSEQN[2:0] in ADC_CSCR register enable the user to instantly set the current sequence number.

Before setting this register, you must set the ADEN in ADC_MR register.

ADC0_CSCR=0x4101_2004, ADC1_CSCR =0x4101_2104, ADC2_CSCR =0x4101_2204

31 30 29 28 27 26 25 24	23 22 21 20 19 18 17 16	15 14 13 12 11 10 9 8	7 6 5	4 3 2 1 0
Reserved		CSEQN[2:0]	Reserved	CACH[4:0]
-		000	-	00000
-		R0	-	R0

10 8	CSEQN[2:0]	Current sequence number	
		000	Current sequence is 0
		001	Current sequence is 1
		010	Current sequence is 2
		011	Current sequence is 3
		100	Current sequence is 4
		101	Current sequence is 5
		110	Current sequence is 6
		111	Current sequence is 7
4 0	CACH[4:0]	Current active channel	
		00000	ADC channel 0 is active
		00001	ADC channel 1 is active
		00010	ADC channel 2 is active
		00011	ADC channel 3 is active
		00100	ADC channel 4 is active
		00101	ADC channel 5 is active
		00110	ADC channel 6 is active
		00111	ADC channel 7 is active
		01000	ADC channel 8 is active
		01001	ADC channel 9 is active
		01010	ADC channel 10 is active
		01011	ADC channel 11 is active
		01100	ADC channel 12 is active
		01101	ADC channel 13 is active
		01110	ADC channel 14 is active
		01111	ADC channel 15 is active
		10000	ADC channel 16 is active
		10001	ADC channel 17 is active
		10010	ADC channel 18 is active
		10011	ADC channel 19 is active
		10100	ADC channel 20 is active
		10101	ADC channel 21 is active
		10110	ADC channel 22 is active
		10111	ADC channel 23 is active
		Others	Reserved

18.5.3 ADCn_CCR: ADC n Clock Control Register

The ADCn_CCR register controls the clock of the ADC module. This register is 16 bits wide.

ADC0_CCR=0x4101_2008, ADC1_CCR=0x4101_2108, ADC2_CCR=0x4101_2208

31 30 29 28 27 26 25 24	23 22 21 20 19 18 17 16	15 14 13 12 11 10 9 8	7 6 5 4 3 2 1 0
Reserved		CLKDIV[6:0]	Reserved
-		0000000	-
-		RW	-

14 8	CLKDIV[6:0]	ADC clock division ratio value bit CLKDIV = 0, ADC clock = ADC input clock (bypass HCLK) CLKDIV ≥ 1, ADC clock = ADC input clock / CLKDIV[6:0]+1 The CLKDIV[6:0] value must be set so that the frequency of the ADC clock's frequency must not exceed 42 MHz.

18.5.4 ADCn_TRG: ADC n Trigger Select Register

The ADCn_TRG register selects trigger sources for the ADC module.

ADC0_TRG=0x4101_200C, ADC1_TRG=0x4101_210C, ADC2_TRG=0x4101_220C

31 30 29 28	27 26 25 24	23 22 21 20	19 18 17 16	15 14 13 12	11 10 9 8	7 6 5 4	3 2 1 0
SEQTRG7[3:0]	SEQTRG6[3:0]	SEQTRG5[3:0]	SEQTRG4[3:0]	SEQTRG3[3:0]	SEQTRG2[3:0]	SEQTRG1[3:0]	SEQTRG0[3:0] BSTTRG[3:0]
0000	0000	0000	0000	0000	0000	0000	0000
RW	RW	RW	RW	RW	RW	RW	RW

30 28	SEQTRG7[3:0]	8th sequence trigger source
27 24	SEQTRG6[3:0]	7th sequence trigger source
23 20	SEQTRG5[3:0]	6th sequence trigger source
19 16	SEQTRG4[3:0]	5th sequence trigger source
15 12	SEQTRG3[3:0]	4th sequence trigger source
11 8	SEQTRG2[3:0]	3rd sequence trigger source
7 4	SEQTRG1[3:0]	2nd sequence trigger source
3 0	SEQTRG0[3:0] BSTTRG[3:0]	1st sequence trigger source Burst conversion trigger source

NOTE:

1. In Multiple mode, the 1st sequence trigger source is given the highest priority, and the 8th sequence trigger source is given the lowest priority. Table 182 shows trigger sources represented by each value. (Refer to section 18.3.17)

18.5.5 ADCn_CSTR1: ADC n Channel Sampling Time 1 Register

The ADCn_CSTR1 register is used to select channel sampling time 1 for the ADC module.

ADC0_CSTR1=0x4101_2010, ADC1_CSTR1=0x4101_2110, ADC2_CSTR1=0x4101_2210

31 30 29 28 27 26 25 24	23 22 21 20 19 18 17 16	15 14 13 12 11 10 9 8	7 6 5 4 3 2 1 0
Reserved / ST3CH[4:0]	Reserved / ST2CH[4:0]	Reserved / ST1CH[4:0]	Reserved / ST0CH[4:0]
- / 00000	- / 00000	- / 00000	- / 00000
- / RW	- / RW	- / RW	- / RW

28 24	ST3CH[4:0]	4th conversion sequence channel sampling time
20 16	ST2CH[4:0]	3rd conversion sequence channel sampling time
12 8	ST1CH[4:0]	2nd conversion sequence channel sampling time
4 0	ST0CH[4:0]	1st conversion sequence channel sampling time

18.5.6 ADCn_CSTR2: ADC n Channel Sampling Time 2 Register

The ADCn_CSTR2 register is used to select the channel sampling time 2 for the ADC module.

ADC0_CSTR2=0x4101_2014, ADC1_CSTR2=0x4101_2114, ADC2_CSTR2=0x4101_2214

31 30 29 28 27 26 25 24	23 22 21 20 19 18 17 16	15 14 13 12 11 10 9 8	7 6 5 4 3 2 1 0
Reserved / ST7CH[4:0]	Reserved / ST6CH[4:0]	Reserved / ST5CH[4:0]	Reserved / ST4CH[4:0]
- / 00000	- / 00000	- / 00000	- / 00000
- / RW	- / RW	- / RW	- / RW

28 24	ST7CH[4:0]	8th conversion sequence channel sampling time
20 16	ST6CH[4:0]	7th conversion sequence channel sampling time
12 8	ST5CH[4:0]	6th conversion sequence channel sampling time
4 0	ST4CH[4:0]	5th conversion sequence channel sampling time

18.5.7 ADCn_SCSR1: ADC n Channel Select 1 Register

ADCn_SCSR1 is the first register for the ADC module's selection of channels.

Each selected sequence's channel works based on the corresponding setting of the sequence's trigger source clock selected by ADCn_TRG register. To write to this register, you must first enable the ADEN in ADCn_MR register.

ADC0_SCSR1=0x4101_2018, ADC1_SCSR1=0x4101_2118, ADC2_SCSR1=0x4101_2218

31 30 29 28 27 26 25 24	23 22 21 20 19 18 17 16	15 14 13 12 11 10 9 8	7 6 5 4 3 2 1 0
Reserved	SEQ3CH[4:0]	Reserved	SEQ2CH[4:0]	Reserved	SEQ1CH[4:0]	Reserved	SEQ0CH[4:0]
-	00000	-	00000	-	00000	-	00000
-	RW	-	RW	-	RW	-	RW

28 24	SEQ3CH[4:0]	4th conversion sequence channel selection
20 16	SEQ2CH[4:0]	3rd conversion sequence channel selection
12 8	SEQ1CH[4:0]	2nd conversion sequence channel selection
4 0	SEQ0CH[4:0](1)	1st conversion sequence channel selection

NOTE:

1. When setting the ADC mode to Single mode, the channel must be set to SEQ0CH[4:0].

18.5.8 ADCn_SCSR2: ADC n Channel Select 2 Register

ADCn_SCSR2 is the first register for the ADC module's selection of channels.

Each selected sequence's channel works based on the corresponding setting of the sequence's trigger source clock selected by ADCn_TRG register. To write to this register, you must first enable the ADEN in ADCn_MR register.

ADC0_SCSR1=0x4101_201C, ADC1_SCSR1=0x4101_211C, ADC2_SCSR1=0x4101_221C

31 30 29 28 27 26 25 24		23 22 21 20 19 18 17 16		15 14 13 12 11 10 9 8		7 6 5 4 3 2 1 0	
Reserved	SEQ7CH[4:0]	Reserved	SEQ6CH[4:0]	Reserved	SEQ5CH[4:0]	Reserved	SEQ4CH[4:0]
-	00000	-	00000	-	00000	-	00000
-	RW	-	RW	-	RW	-	RW

28 24	SEQ7CH[4:0]	8th conversion sequence channel selection
20 16	SEQ6CH[4:0]	7th conversion sequence channel selection
12 8	SEQ5CH[4:0]	6th conversion sequence channel selection
4 0	SEQ4CH[4:0](1)	5th conversion sequence channel selection

18.5.9 ADCn_CR: ADC n Control Register

ADCn_CR is used to control the operation of ADC module.

ADC0_CR=0x4101_2020, ADC1_CR=0x4101_2120, ADC2_CR=0x4101_2220

31 30 29 28 27 26 25 24	23 22 21 20 19 18 17 16	15 14 13 12 11 10 9 8	7	6 5	4	3 2	1	0
Reserved			ASTOP	Reserved	AASTART	Reserved	TRGCLR	ASTART
-			0	-	0	-	0	0
-			WO	-	RW	-	RW	RW

7	ASTOP	ADC STOP bit
	0	No effect
	1	Stops ADC conversion (Cleared at the next ADC clock cycle) If ASTOP sets enabled after a conversion cycle starts, the current conversion is completed.
4	AASTART	All ADC START bit
	0	No effect
	1	All ADC Conversion start
1	TRGCLR	ADC all trigger flags clear option
	0	No clear
	1	Clears all trigger flags generated during the previous ADC operation.
0	ASTART	ADC START bit
	0	No ADC Conversion
	1	Starts ADC conversion (ASTART bit will be clear at the next ADC clock cycle). To start ADC conversion, the ADEN bit must be set to 1. If ARST is 0 in trigger event mode, setting ASTART to 1 will start ADC conversion, which then will be performed as many times as set with SEQCNT[2:0] in ADC_MR register.

18.5.10 ADCn_SR: ADC n Status Register

The ADCn_SR register displays the status of the ADC.

ADC0_SR=0x4101_2024, ADC1_SR=0x4101_2124, ADC2_SR=0x4101_2224

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
ABUSY	Reserved																					CMP1IF	CMP0IF	Reserved		DOVRUN	DMAF	TRGIF	EOSIF	IOCIF	EOCIF
0	-																					0	0	-		0	0	0	0	0	0
RO	-																					RWC1	RWC1	-		RO	RO	RWC1	RWC1	RWC1	RWC1

31	ABUSY	ADC busy status
		0 Not busy
		1 Busy
9	CMP1IF	Compare 1 interrupt flag bit
		0 Interrupt is not generated.
		1 Interrupt is generated (Writing 1 clears the flag).
8	CMP0IF	Compare 0 interrupt flag bit
		0 Interrupt is not generated.
		1 Interrupt is generated (Writing 1 clears the flag).
5	DOVRUN	DMA overrun flag
		0 Flag is not detected.
		1 Flag is detected.
4	DMAF	DMA done received flag (DMA transfer is completed.)
		0 Flag is not detected.
		1 Flag is detected.
3	TRGIF	ADC trigger interrupt flag
		0 Flag is not detected.
		1 Flag is detected (writing 1 clears the flag).
2	EOSIF	Sequence end interrupt flag
		0 Flag is not detected.
		1 Flag is detected (writing 1 clears the flag).
		0 Interrupt is not generated.
1	IOCIF	Injection conversion end interrupt flag
		0 Flag is not detected.
		1 Flag is detected (writing 1 clears the flag).
0	EOCIF(1)	Sequence conversion end interrupt flag
		0 Flag is not detected.
		1 Flag is detected (writing 1 clears the flag).

NOTE:

1. To poll the flag, you must use the EOCIF in ADC0_SR bit. Channels 1 and 2 can also be set in the same way.
   Example)
   ```
   while ( (ADC0_SR & 0x01) == 0 ); // EOCIF flag checking
   ADC0_SR = 0x1;                   // EOCIF flag clear
   …
   ```

18.5.11 ADCn_IER: ADC n Interrupt Enable Register

The ADCn_IER register determines whether to enable the ADC interrupts.

ADC0_IER=0x4101_2028, ADC1_IER=0x4101_2128, ADC2_IER=0x4101_2228

31 30 29 28 27 26 25 24	23 22 21 20 19 18 17 16	15 14 13 12 11 10	9	8	7 6	5	4	3	2	1	0
Reserved			CMP1IE	CMP0IE	Reserved	DOVRUNIE	DMAIE	TRGIE	EOSIE	IOCIE	EOCIE
-			0	0	-	0	0	0	0	0	0
-			RW	RW	-	RW	RW	RW	RW	RW	RW

9	CMP1IE	Compare 1 event interrupt enable
		0 Disables.
		1 Enables.
8	CMP0IE	Compare 0 event interrupt enable
		0 Disables.
		1 Enables.
5	DOVRUNIE	DMA overrun interrupt enable
		0 Disables
		1 Enables
4	DMAIE	DMA done interrupt enable
		0 Disables.
		1 Enables.
3	TRGIE	ADC trigger conversion interrupt enable
		0 Disables.
		1 Enables.
2	EOSIE(1)	ADC sequence conversion interrupt enable
		0 Disables.
		1 Enables.
1	IOCIE	Injection conversion end interrupt enable
		0 Disables.
		1 Enables.
0	EOCIE	ADC single conversion interrupt
		0 Disables.
		1 Enables.

NOTE:

1. Burst mode sets the EOSIE = '1' and check the EOSIF in ADCn_SR register.

18.5.12 ADCn_DDR: ADC n DMA Data Register

ADCn_DDR manages the ADC module's DMA data. It displays the results of the ADC module's DMA conversions.

ADC0_DDR=0x4101_202C, ADC1_DDR=0x4101_212C, ADC2_DDR=0x4101_222C

31	30	29	28	27	26	25	24	23	22 21 20 19 18 17 16	15 14 13 12 11 10 9 8	7 6 5 4 3 2 1 0
TRGINFO7	TRGINFO6	TRGINFO5	TRGINFO4	TRGINFO3	TRGINFO2	TRGINFO1	TRGINFO0	ITRG	Reserved	ADMACH[4:0]	ADDMAR[11:0] ADC DMA temporary data	Reserved
0	0	0	0	0	0	0	0	0	-	0x00	0x000	-
R0	R0	R0	R0	R0	R0	R0	R0	R0	-	R0	R0	-

31 24	TRGINFOx (x = 0 to 7)	ADC trigger information - Indicates whether or not each trigger source has been captured until the end of conversion (EOC). **For Multiple mode:** The lower the TRGINFOx number, the higher priority it has. If a trigger is pending due to another ongoing trigger, multiple TRGINFOx can be read as '1' (x = 0 to 7). **For Single/Sequential modes:** The bit field displays which trigger sources are pending due to another ongoing trigger. You can find out which trigger source is currently being processed by referring to the CSEQN[2:0] in ADCn_CSCR register. (n = 0 to 2)
23	ITRG	Injected mode trigger information 0 Disables. 1 Enables.
20 16	ADMACH[4:0]	DMA ADC channel indicator
15 4	ADDMAR[11:0]	DMA ADC conversion result data (12-bit)

NOTES:

1. Even when DMA is inactive, data is temporarily stored in this register before stored in the designated buffer. Additionally, the register also displays which channel the data belongs to.
2. TRGINFOx indicates the flag state in which a trigger occurred, and the corresponding trigger is cleared by matching the SOC after the sequence event is processed. However, in burst mode, it is masked as TRGINFO0 = 1 and other TFGIFOx bits are cleared

18.5.13 ADCn_DRx: ADC n Data 0 to 7 Register

The ADCn_DRx register displays the results of ADC conversions. There are eight of these registers.

ADC0_DR0=0x4101_2030, ADC0_DR1=0x4101_2034, ADC0_DR2=0x4101_2038, ADC0_DR3=0x4101_203C
ADC0_DR4=0x4101_2040, ADC0_DR5=0x4101_2044, ADC0_DR6=0x4101_2048, ADC0_DR7=0x4101_204C
ADC1_DR0=0x4101_2130, ADC1_DR1=0x4101_2134, ADC1_DR2=0x4101_2138, ADC1_DR3=0x4101_213C
ADC1_DR4=0x4101_2140, ADC1_DR5=0x4101_2144, ADC1_DR6=0x4101_2148, ADC1_DR7=0x4101_214C
ADC2_DR0=0x4101_2230, ADC2_DR1=0x4101_2234, ADC2_DR2=0x4101_2238, ADC2_DR3=0x4101_223C
ADC2_DR4=0x4101_2240, ADC2_DR5=0x4101_2244, ADC2_DR6=0x4101_2248, ADC2_DR7=0x4101_224C

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
TRGINFO7	TRGINFO6	TRGINFO5	TRGINFO4	TRGINFO3	TRGINFO2	TRGINFO1	TRGINFO0	Reserved			ACH[4:0]					ADDATA[11:0]												Reserved				
0	0	0	0	0	0	0	0	-			0x00					0x000												-			
R0	R0	R0	R0	R0	R0	R0	R0	-			R0					R0												-			

31 24	TRGINFOx (x = 0 to 7)	ADC trigger information - Indicates whether or not each trigger source has been captured until the end of conversion (EOC). **For Multiple mode:** The lower the TRGINFOx number, the higher priority it has. If a trigger is pending due to another ongoing trigger, multiple TRGINFOx can be read as '1' (x = 0 to 7). **For Single/Sequential modes:** The bit field displays which trigger sources are pending due to another ongoing trigger. You can find out which trigger source is currently being processed by referring to the CSEQN[2:0] in ADCn_CSCR register. (n = 0 to 2)
20 16	ACH[4:0]	ADC channel indicator
15 4	ADDATA[11:0]	ADC conversion result data (12-bit)

NOTE:

1. TRGINFOx indicates the flag state in which a trigger occurred, and the corresponding trigger is cleared by matching the SOC after the sequence event is processed. However, in burst mode, it is masked as TRGINFO0 = 1 and other TFGIFOx bits are cleared

18.5.14 ADCn_IMR: ADC n Injection Mode Register

The ADCn_IMR register is used for ADC injection mode.

ADC0_IMR=0x4101_2050, ADC1_IMR=0x4101_2150, ADC2_IMR=0x4101_2250

31 30 29 28 27 26 25 24	23 22 21 20 19 18 17 16	15 14 13 12 11 10 9 8	7 6 5 4 3 2 1 0
Reserved / TRGSRC[3:0]	Reserved / CST[4:0]	Reserved / CHSEL[4:0]	Reserved / TRGSEL[2:0] / Reserved / INJECTMODE
- / 0000	- / 00000	- / 00000	- / 000 - / 0
- / RW	- / RW	- / RW	- / RW - / RW

27 24	TRGSRC[3:0]	Trigger source selection bit	
20 16	CST[4:0]	Channel sampling time	
12 8	CHSEL[4:0]	Channel selection	
6 4	TRGSEL[2:0]	Trigger selection bit	
		000	Event trigger disable/Soft trigger only
		001	Timer event trigger/Soft trigger
		010	MPWM0 event trigger/Soft trigger
		011	MPWM1 event trigger/Soft trigger
		Others	Reserved
1	EXTADCRQEN	Conversion start control request bit from another external ADC	
		0	Disable
		1	Enable
0	INJECTMODE	Injection conversion mode enable bit	
		0	Disable
		1	Enable

18.5.15 ADCn_ICR: ADC n Injection Control Register

The ADCn_ICR register is used to control the ADC injection.

ADC0_ICR=0x4101_2054, ADC1_ICR=0x4101_2154, ADC2_ICR=0x4101_2254

31 30 29 28 27 26 25 24	23 22 21 20 19 18 17 16	15 14 13 12 11 10 9 8	7 6 5	4	3 2 1	0
Reserved				AASTART	Reserved	ASTART
-				0	-	0
-				RW	-	RW

4	AASTART	All ADC START bit
		0 No effect
		1 All ADC Conversion start
0	ASTART	ADC START bit
		0 No ADC conversion
		1 Starts ADC conversion (Cleared at the next ADC clock cycle). To start ADC conversion, the ADEN bit must be set to 1. If ARST is '0' in trigger event mode, setting ASTART to 1 will start ADC conversion.

18.5.16 ADCn_IDR: ADC n Injection Data Register

The ADCn_IDR register is used for ADC injection data.

ADC0_IDR=0x4101_2058, ADC1_IDR=0x4101_2158, ADC2_IDR=0x4101_2258

31 30 29 28 27 26 25 24	23 22 21 20 19 18 17 16	15 14 13 12 11 10 9 8	7 6 5 4 3 2 1 0
Reserved	ACH[4:0]	ADDATA[11:0]	Reserved
-	00000	0x000	-
-	R0	R0	-

20 16	ACH[4:0]	ADC channel information
15 4	ADDATA[11:0]	ADC input data

18.5.17 ADCn_CMPR1: ADC n Channel Comparison Register 1

The ADCn_CMPR1 register controls the comparison between the ADC channels.

ADC0_CMPR1=0x4101_2060, ADC1_CMPR1=0x4101_2160, ADC2_CMPR1=0x4101_2260

31 30 29 28 27 26 25 24	23	22	21	20 19 18 17 16	15 14 13 12 11 10 9 8	7 6 5 4 3 2 1 0
MATCH_CNT[5:0]	CMPEN	Reserved	CMPM	CCH[4:0]	CVAL[11:0]	Reserved
000000	0	-	0	00000	0x000	-
RW	RW	-	RW	RW	RW	-

31 26	MATCH_CNT[5:0]	Interrupt is generated only when ADC data compare data meets the condition (MATCHCNT+1) consecutively.	
23	CMPEN	Compare operation enable bit	
		0	Disables.
		1	Enables.
21	CMPM	ADC compare mode bit	
		0	ADC value is greater or equal to the CVAL[11:0] value.
		1	ADC value is less than CVAL[11:0] value.
20 16	CCH[4:0]	Compare channel	
		00000	Compare channel is ADC channel 0
		00001	Compare channel is ADC channel 1
		00010	Compare channel is ADC channel 2
		00011	Compare channel is ADC channel 3
		00100	Compare channel is ADC channel 4
		00101	Compare channel is ADC channel 5
		00110	Compare channel is ADC channel 6
		00111	Compare channel is ADC channel 7
		01000	Compare channel is ADC channel 8
		01001	Compare channel is ADC channel 9
		01010	Compare channel is ADC channel 10
		01011	Compare channel is ADC channel 11
		01100	Compare channel is ADC channel 12
		01101	Compare channel is ADC channel 13
		01110	Compare channel is ADC channel 14
		01111	Compare channel is ADC channel 15
		10000	Compare channel is ADC channel 16
		10001	Compare channel is ADC channel 17
		10010	Compare channel is ADC channel 18
		10011	Compare channel is ADC channel 19
		10100	Compare channel is ADC channel 20
		10101	Compare channel is ADC channel 21
		10110	Compare channel is ADC channel 22
		10111	Compare channel is ADC channel 23
		Others	Reserved
15 4	CVAL[11:0]	Compare value bit	

18.5.18 ADCn_CMPR2: ADC n Channel Comparison Register 2

The ADCn_CMPR2 register controls the comparison between the ADC channels.

ADC0_CMPR2=0x4101_2064, ADC1_CMPR2=0x4101_2164, ADC2_CMPR2=0x4101_2264

31 30 29 28 27 26 25 24	23	22	21	20 19 18 17 16	15 14 13 12 11 10 9 8	7 6 5 4 3 2 1 0
MATCH_CNT[5:0]	CMPEN	Reserved	CMPM	CCH[4:0]	CVAL[11:0]	Reserved
000000	0	-	0	00000	0x000	-
RW	RW	-	RW	RW	RW	-

31 26	MATCH_CNT[5:0]	Interrupt is generated only when ADC data compare data meets the condition (MATCHCNT+1) consecutively.	
23	CMPEN	Compare operation enable bit	
		0	Disables.
		1	Enables.
21	CMPM	ADC compare mode bit	
		0	ADC value is greater or equal to the CVAL[11:0] value.
		1	ADC value is less than CVAL[11:0] value.
20 16	CCH[4:0]	Compare channel	
		00000	Compare channel is ADC channel 0
		00001	Compare channel is ADC channel 1
		00010	Compare channel is ADC channel 2
		00011	Compare channel is ADC channel 3
		00100	Compare channel is ADC channel 4
		00101	Compare channel is ADC channel 5
		00110	Compare channel is ADC channel 6
		00111	Compare channel is ADC channel 7
		01000	Compare channel is ADC channel 8
		01001	Compare channel is ADC channel 9
		01010	Compare channel is ADC channel 10
		01011	Compare channel is ADC channel 11
		01100	Compare channel is ADC channel 12
		01101	Compare channel is ADC channel 13
		01110	Compare channel is ADC channel 14
		01111	Compare channel is ADC channel 15
		10000	Compare channel is ADC channel 16
		10001	Compare channel is ADC channel 17
		10010	Compare channel is ADC channel 18
		10011	Compare channel is ADC channel 19
		10100	Compare channel is ADC channel 20
		10101	Compare channel is ADC channel 21
		10110	Compare channel is ADC channel 22
		10111	Compare channel is ADC channel 23
		Others	Reserved
15 4	CVAL[11:0]	Compare value bit	

18.5.19 ADCn_CD01R: ADC0, 1 Data Register

The ADC0, 1 data register displays the results of ADC conversions.

ADC0_CD01R=0x4101_2080, ADC1_CD01R=0x4101_2180, ADC2_CD01R=0x4101_2280

31 30 29 28 27 26 25 24	23 22 21 20 19 18 17 16	15 14 13 12 11 10 9 8	7 6 5 4 3 2 1 0
ADC1_DATA[27:16]	Reserved	ADC0_DATA[11:0]	Reserved
0x000	-	0x000	-
R0	-	R0	-

31 20	ADC1_DATA [27:16]	ADC1 conversion result data (12-bit)
15 4	ADC0_DATA [11:0]	ADC0 conversion result data (12-bit)

18.5.20 ADCn_CD2R: ADC2 Data Register

The ADC2 data register displays the results of ADC conversions.

ADC0_CD2R=0x4101_2084, ADC1_CD2R=0x4101_2184, ADC2_CD2R=0x4101_2284

31 30 29 28 27 26 25 24	23 22 21 20 19 18 17 16	15 14 13 12 11 10 9 8	7 6 5 4 3 2 1 0
Reserved		ADC2_DATA[11:0]	Reserved
-		0x000	-
-		R0	-

15 4	ADC2_DATA [11:0]	ADC2 conversion result data (12-bit)

18.5.21 ADC Register Map Summary

Table 189. ADC Register Map Summary

Offset	Register name	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0x00	ADCn_MR	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	EXTADCRQEN	Res.	Res.	CDATA	DMAIDE	DMAEN	Res.	Res.	Res.	Res.	Res.	SEQTRGT	SEQCNT[2:0]			ADEN	ARST	ADMOD[1:0]		Res.	TRGSEL[2:0]		
	Reset value										0			0	0	0						0	0	0	0	0	0	0	0		0	0	0
0x04	ADCn_CSCR	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	CSEQN[2:0]			Res.	Res.	Res.	CACH[4:0]				
	Reset value																						0	0	0				0	0	0	0	0
0x08	ADCn_CCR	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	CLKDIV[6:0]							Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.
	Reset value																		0	0	0	0	0	0	0								
0x0C	ADCn_TRG	SEQTRG7[3:0]				SEQTRG6[3:0]				SEQTRG5[3:0]				SEQTRG4[3:0]				SEQTRG3[3:0]				SEQTRG2[3:0]				SEQTRG1[3:0]				SEQTRG0[3:0] BSTTRG[3:0]			
	Reset value	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
0x10	ADCn_CSTR1	Res.	Res.	Res.	ST3CH[4:0]					Res.	Res.	Res.	ST2CH[4:0]					Res.	Res.	Res.	ST1CH[4:0]					Res.	Res.	Res.	ST0CH[4:0]				
	Reset value				0	0	0	0	0				0	0	0	0	0				0	0	0	0	0				0	0	0	0	0
0x14	ADCn_CSTR2	Res.	Res.	Res.	ST7CH[4:0]					Res.	Res.	Res.	ST6CH[4:0]					Res.	Res.	Res.	ST5CH[4:0]					Res.	Res.	Res.	ST4CH[4:0]				
	Reset value				0	0	0	0	0				0	0	0	0	0				0	0	0	0	0				0	0	0	0	0
0x18	ADCn_SCSR1	Res.	Res.	Res.	SEQ3CH[4:0]					Res.	Res.	Res.	SEQ2CH[4:0]					Res.	Res.	Res.	SEQ1CH[4:0]					Res.	Res.	Res.	SEQ0CH[4:0]				
	Reset value				0	0	0	0	0				0	0	0	0	0				0	0	0	0	0				0	0	0	0	0
0x1C	ADCn_SCSR2	Res.	Res.	Res.	SEQ7CH[4:0]					Res.	Res.	Res.	SEQ6CH[4:0]					Res.	Res.	Res.	SEQ5CH[4:0]					Res.	Res.	Res.	SEQ4CH[4:0]				
	Reset value				0	0	0	0	0				0	0	0	0	0				0	0	0	0	0				0	0	0	0	0
0x20	ADCn_CR	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	ASTOP	Res.	Res.	AASTART	Res.	Res.	TRGCLR	ASTART
	Reset value																									0			0			0	0
0x24	ADCn_SR	ABUSY	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	CMP1IF	CMP0IF			DOVRUN	DMAF	TRGIF	EOSIF	IOCIF	EOCIF
	Reset value	0																						0	0			0	0	0	0	0	0
0x28	ADCn_IER	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	CMP1IE	CMP0IE			DOVRUNIE	DMAIE	TRGIE	EOSIE	IOCIE	EOCIE
	Reset value																							0	0			0	0	0	0	0	0
0x2C	ADCn_DDR	TRGINFO7	TRGINFO6	TRGINFO5	TRGINFO4	TRGINFO3	TRGINFO2	TRGINFO1	TRGINFO0	ITRG	Res.	Res.	ADMACH[4:0]					ADDMAR[11:0]													Res.	Res.	Res.	Res.
	Reset value	0	0	0	0	0	0	0	0	0			0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0				

Table 189. ADC Register Map Summary (continued)

Offset	Register name	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0x30-0x4C	ADCn_DRx	TRGINFO7	TRGINFO6	TRGINFO5	TRGINFO4	TRGINFO3	TRGINFO2	TRGINFO1	TRGINFO0	Res	Res	Res	ACH[4:0]					ADDDATA[11:0]												Res	Res	Res	Res
	Reset value	0	0	0	0	0	0	0	0				0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0				
0x50	ADCn_IMR	Res	Res	Res	Res	TRGSRC[3:0]				Res	Res	Res	CST[4:0]					Res	Res	Res	CHSEL[4:0]					Res	TRGSEL[2:0]			Res	Res	Res	INJECTMODE
	Reset value					0	0	0	0				0	0	0	0	0				0	0	0	0	0		0	0	0				0
0x54	ADCn_ICR	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	AASTART	Res	Res	Res	ASTART
	Reset value																												0				0
0x58	ADCn_IDR	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	ACH[4:0]					ADDATA[11:0]												Res	Res	Res	Res
	Reset value												0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0				
0x60	ADCn_CMPR1	MATCH_CNT[5:0]						Res	Res	CMPEN	Res	CMPM	CCH[4:0]					CVAL[11:0]												Res	Res	Res	Res
	Reset value	0	0	0	0	0	0			0		0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0				
0x64	ADCn_CMPR2	MATCH_CNT[5:0]						Res	Res	CMPEN	Res	CMPM	CCH[4:0]					CVAL[11:0]												Res	Res	Res	Res
	Reset value	0	0	0	0	0	0			0		0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0				
0x80	ADCn_CD01R	ADC1_DATA[11:0]												Res	Res	Res	Res	ADC0_DATA[11:0]												Res	Res	Res	Res
	Reset value	0	0	0	0	0	0	0	0	0	0	0	0					0	0	0	0	0	0	0	0	0	0	0	0				
0x84	ADCn_CD2R	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	ADC2_DATA[11:0]												Res	Res	Res	Res
	Reset value																	0	0	0	0	0	0	0	0	0	0	0	0				

19. Comparator (CMP)

19.1 Comparator Introduction

The A34G43x series has four comparators. Each comparator outputs the comparison results of two analog voltages via an I/O pin or generates interrupts.

Users can control the comparators by configuring the CMPn_CONF register. The HYSSEL in CMPn_CONF register is used to select the comparator hysteresis voltage, and the HYSEN enables the hysteresis.

19.2 Comparator Main Features

The comparators of A34G43x have key features as listed below:

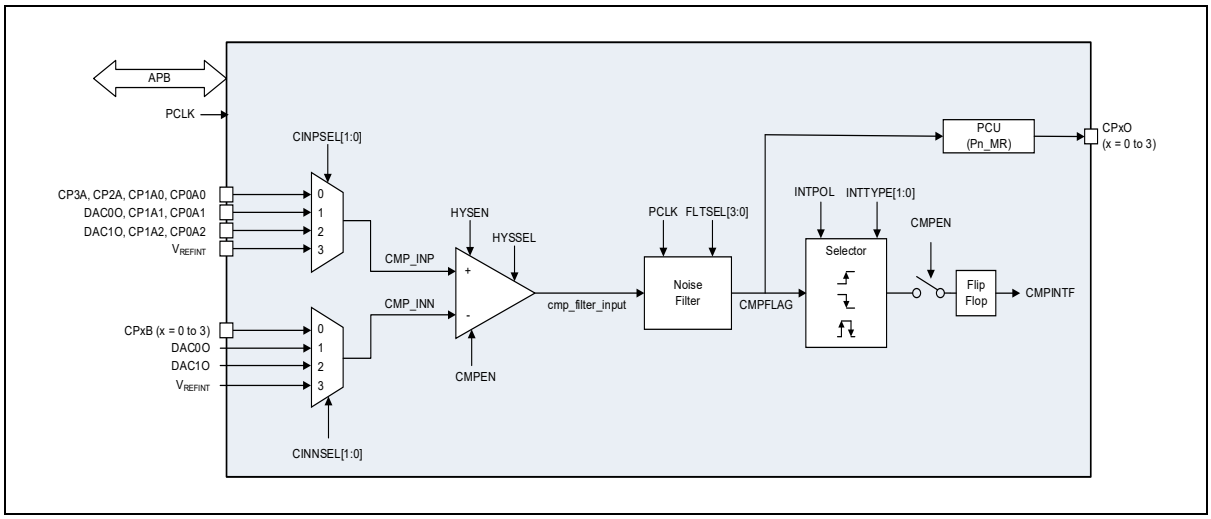
- Four built-in Comparators (Comparator0, 1, 2 and 3), each of which has two input sources (CINNSEL, CINPSEL).
- The selection between falling and rising of the interrupt signal outputs is available.
- Noise filtering capability
- Four hysteresis voltages (40 mV, 80 mV, 120 mV and 160 mV) available.

19.3 Comparator Functional Description

19.3.1 Comparator Block Diagram

The comparator is described in a block diagram as shown in Figure 216.

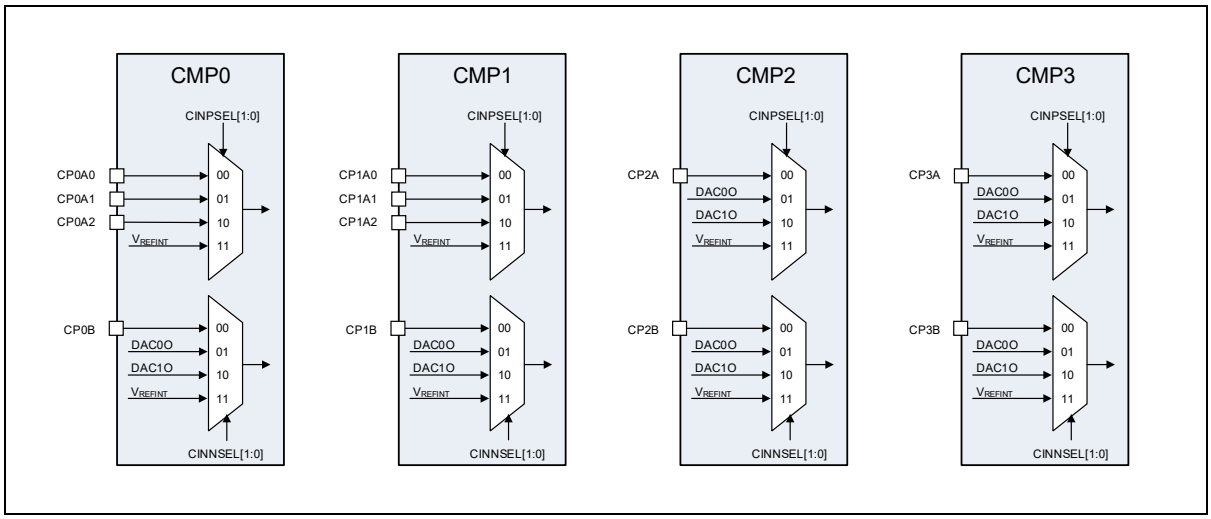
Figure 216. Comparator Block Diagram



NOTE:

1. V_{REFINT} is Typ. 0.6 V

Figure 217. Comparator Input Source Block Diagram



19.3.2 Comparator Pins and Internal Signals

To use I/O ports for the comparators, users must change the mode using the Pn_MR1 and Pn_MR2 (n = A) register in the Port Control Unit (PCU). More information about the PCU, refer to Chapter 5.

The Comparator input pins of the Pin Mux are set to have the same value as the ADC (refer to section 5.4.1 section 5.4.2 and Table 38). Functions can work together to allow users to use a combination of the Comparator and the ADC.

Table 190 describes pins assigned for the comparator.

Table 190. Pin Assignment of Comparator: External Pins

Pin Name	Type	Description	Supported Packages		
			A34G43xYL (120-LQFP)	A34G43xVL (100-LQFP)	A34G43xRL (64-LQFP)
AVDD	P	Analog VDD (2.7V to VDD)	O	O	O
AGND	P	Analog GND	O	O	O
CP0A0	A	Comparator 0 input A0	O	O	O
CP0A1	A	Comparator 0 input A1	O	O	O
CP0A2	A	Comparator 0 input A2	O	O	O
CP1A0	A	Comparator 1 input A0	O	O	O
CP1A1	A	Comparator 1 input A1	O	O	O
CP1A2	A	Comparator 1 input A2	O	O	O
CP2A	A	Comparator 2 input A	O	O	N/A
CP3A	A	Comparator 3 input A	O	O	N/A
CP0B	A	Comparator 0 input B	O	O	O
CP1B	A	Comparator 1 input B	O	O	O
CP2B	A	Comparator 2 input B	O	O	N/A
CP3B	A	Comparator 3 input B	O	O	N/A
CP0O	O	Comparator 0 output	O	O	O
CP1O	O	Comparator 1 output	O	O	O
CP2O	O	Comparator 2 output	O	O	O
CP3O	O	Comparator 3 output	O	O	O

NOTE:

1. P: Power, O: Output, A: Analog Input.

Users can refer to the values shown in Table 191 and Table 192, when selecting the input signal and reference signal.

NOTE:

1. During flash erase or program operation, it is not allowed to change the input to V_{REFINT} or from V_{REFINT} to another input by using the CINNSEL[1:0] or CINPSEL[1:0] bits.

Table 191. Comparator Input Sources

CINPSEL[1:0]	Input (+)			
	CMP0_INP	CMP1_INP	CMP2_INP	CMP3_INP
00	CP0A0 (PA4)	CP1A0 (PA8)	CP2A (PE1)	CP3A (PF2)
01	CP0A1 (PA5)	CP1A1 (PA9)	DAC0O	DAC0O
10	CP0A2 (PA6)	CP1A2 (PA10)	DAC1O	DAC1O
11	V_{REFINT}	V_{REFINT}	V_{REFINT}	V_{REFINT}

Table 192. Comparator Reference Sources

CINNSEL[1:0]	Reference (−)			
	CMP0_INN	CMP1_INN	CMP2_INN	CMP3_INN
00	CP0B (PA7)	CP1B (PA11)	CP2B (PE2)	CP3B (PF3)
01	DAC0O	DAC0O	DAC0O	DAC0O
10	DAC1O	DAC1O	DAC1O	DAC1O
11	V_{REFINT}	V_{REFINT}	V_{REFINT}	V_{REFINT}

For interrupt polarity settings, refer to Table 193.

If the INTTYPE[1:0] bits is ‘01’ (level type), the INTPOL bit is selected as low or high level. If the INTTYPE[1:0] bits is ‘10’ (edge type), the INTPOL bit is selected as falling or rising edge. The setting of the INTPOL bit is ignored when the INTTYPE[1:0] bits is selected as ’11’ (both edges).

Table 193. Comparator Interrupt Polarity

Interrupt Type	Interrupt Polarity		Remark
INTTYPE[1:0]	INTPOL = ‘0’	INTPOL = ‘1’	
00	Disable	Disable	
01	Low	High	Level type
10	Falling	Rising	Edge type
11	don’t care	don’t care	Both edge

19.3.3 Comparator Reference and Input Voltage Configuration

A comparator generates a digital output signal based on a voltage comparison between an source (input +) signal and a reference (input -) signal. If the source voltage is higher than the reference voltage, the digital output signal is '1'. Otherwise, the output signal is '0'.

The comparator module must be enabled before working, so the CMP bit in the SCU_PER2 and SCU_PCER2 register must be set to '1'. The comparator in A34G43x series is enabled when the CMPEN bit in the CMPn_CTRL register is set to '1'.

Users can confirm the comparator's output by checking the CMPFLAG flag in the CMP_STAT register. If an interrupt is enabled, users can use the CMPINTF bit in the CMP_STAT register to check whether the interrupt is generated.

The reference voltage of the comparator can be decided by configuring the CINNSEL[1:0] bits in the CMPn_CONF register. If the CINNSEL[1:0] bits is set to '00', the input voltage via an external I/O pin is set as the reference voltage (CPxB, where x = 0 to 3). If the CINNSEL[1:0] is set to '1' or '2', the DACxO(x = 0 to 1) voltage in the microcontroller is set as the reference voltage. If the CINNSEL[1:0] bits is set to '3', the V_{REFINT} voltage in the microcontroller is set as the reference voltage. Refer to Table 192.

The input(source) voltage of the comparator can be decided by configuring the CINPSEL[1:0] bits in the CMPn_CONF register. Users can select one from three input sources by writing a number ranging from '0' to '3' in the CINPSEL[1:0] bits.

The comparator module in the A34G43x series provides a function that adds hysteresis voltage to the input pin. Users can select a hysteresis voltage value by writing a proper number in the HYSSEL[1:0] bit in the CMP_CONF register after setting the HYSEN bit to '1' to enable the hysteresis function.

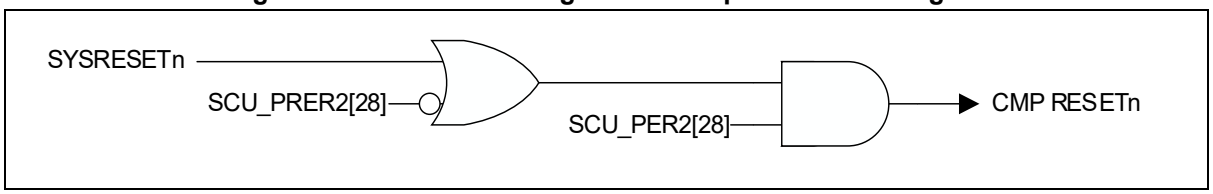
19.3.4 Comparator Reset and Clocks

The comparator module must be enabled before working, so the CMP bit in the SCU_PER2 and SCU_PCER2 register must be set to ‘1’.

The comparator module can be reset by disabling and enabling again the CMP bit in the SCU_PER2 and SCU_PCER2 register, respectively. If the corresponding reset mask bit in the SCU_PRER2 register is set to ‘0’, the settings for the comparator module can be maintained when the warm reset occurs.

Figure 218 shows a schematic diagram regarding the comparator reset signal. For more information about the comparator reset and clocks, refer to section 4.4 and section 4.5.

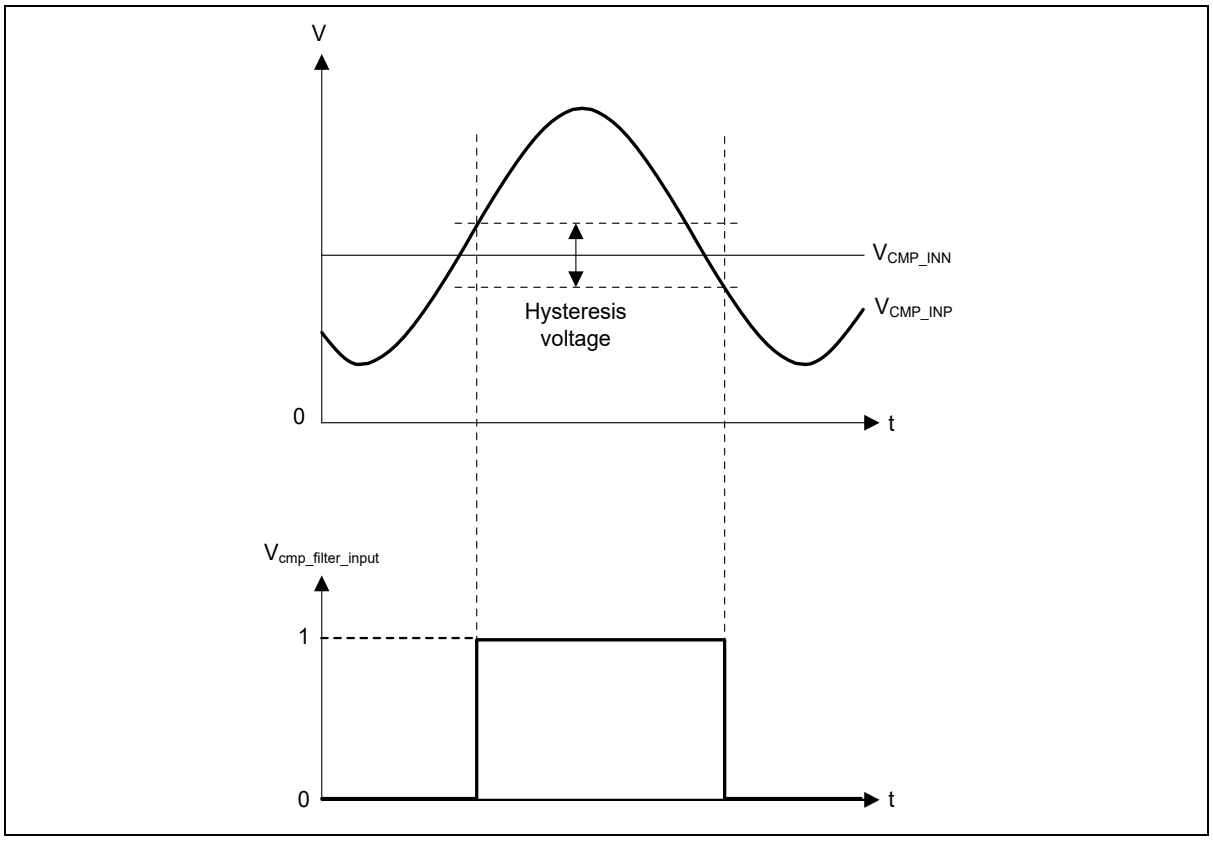
Figure 218. Schematic Diagram of Comparator Reset Signal



19.3.5 Comparator Hysteresis

If the comparator's input signal contains noise, the comparator's output may be changed at the unwanted point of time. To prevent this behavior, the comparator in A34G43x has hysteresis function and can select comparator hysteresis voltage between 40 mV and 160 mV using the HYSSEL[1:0] bit. It is enabled or disabled by configuring the HYSEN bit.

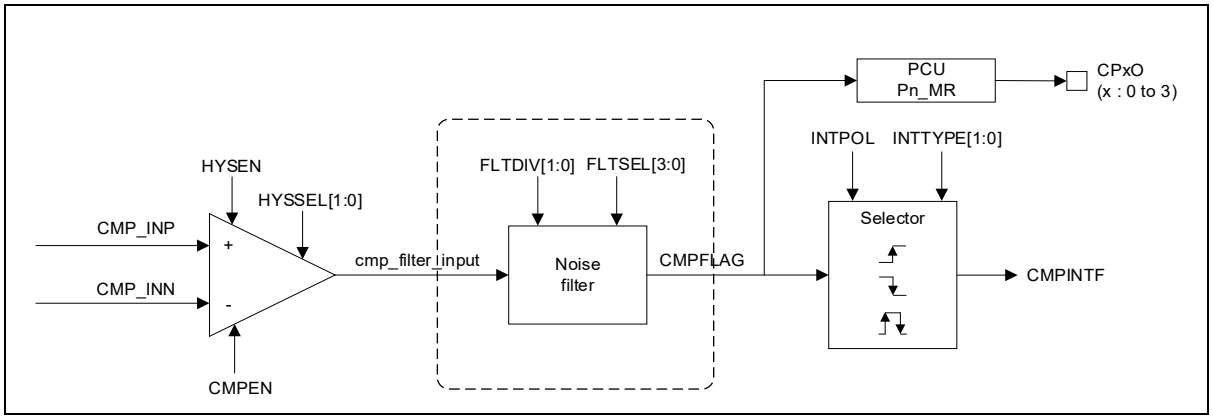
Figure 219. Comparator Hysteresis



19.3.6 Comparator Filter

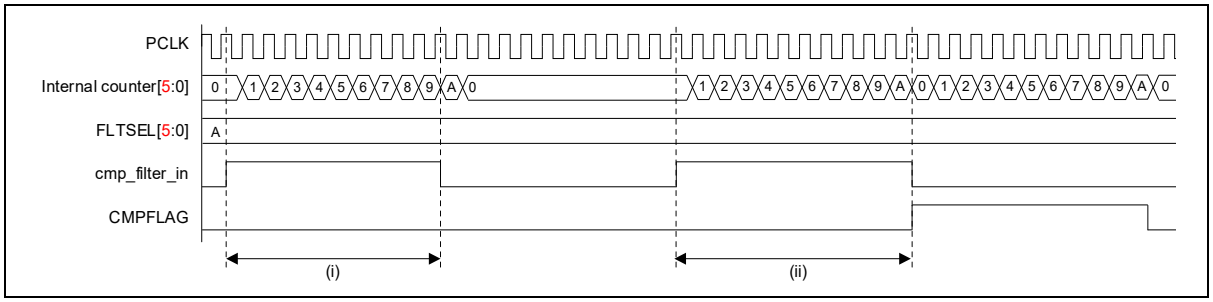
A34G43x comparators have a noise filter inside. Figure 220 shows a block diagram of the comparator filter that operates based on the PCLK.

Figure 220. Comparator Filter



The comparator filter uses the value of the FLTSEL[5:0] multiplied by the 1/PCLK time for filtering. If the FLTSEL[5:0] value and PCLK have values of two and 1 MHz respectively, the input signal must be maintained for 2 μs or longer time value corresponding to 500 kHz to generate the CMPFLAG.

Figure 221. Comparator Filter Timing



NOTE:

1. FLTDIV[1:0] is '0'.

According to the example simulation shown in Figure 221, the input signal requires delay time that is longer than one tenth of the PCLK (800 kHz or higher), if the PCLK period is 8 MHz and the FLTSEL[5:0] bits is '0x0A'.

In the Figure 221,

1. This section shows that the cmp_filter_input (input signal for noise filter) has 1,250 ns (FLTSEL[5:0] × 1 / PCLK) of input pulse width. In this section, the input value becomes the same as the noise filter time width, therefore the filter outputs (CMPFLAG) are canceled to generate ‘0’.
2. This section shows that the cmp_filter_input has 1,350 ns ((FLTSEL[5:0] + 1) × (1 / PCLK)) of input pulse width. In this section, the input value becomes greater than the values of the noise filter time width, therefore the filter outputs (CMPFLAG) are generated.

   This means that the cmp_filter_input must be maintained for at least the time of ((FLTSEL[5:0] + 1) × (1 / PCLK)) to generate outputs normally.

If the cmp_filter_input changes while the input signal is entering, an internal counter is reset and counts from the beginning to filter the noise that occurs while the input signal enters.

19.4 Comparator Low-Power Modes

Table 194 describes two low-power modes that A34G43x supports for the comparator.

Table 194. Comparator Low-Power Modes

Mode	Description
SLEEP	No effect
DEEP-SLEEP	In this mode, incoming clocks to the comparator is blocked, and the corresponding comparator device is disabled.

19.5 Comparator Interrupts

Comparator interrupt occurs at every rising/falling edge or high/low level of the comparator output. When the INTTYPE[1:0] bits in CMPn_CONF register is set to ‘10’ (Single Edge), the comparator of A34G43x series detects rising edges if the INTPOL in CMPn_CONF register is set to ‘1’; while it detects falling edges if the INTPOL bit is set to ‘0’. When the INTTYPE[1:0] bits in CMPn_CONF register is set to ‘01’, the comparator of A34G43x series detects high level if the INTPOL bit in CMPn_CONF register is set to ‘1’; while it detects low level if the INTPOL bit is set to ‘0’.

Users can confirm this interrupt’s status by checking the CMPINTF bit in CMPn_STAT register. Writing ‘1’ to the bit field clears the corresponding interrupt flag.

Even if CMPINTEN bit in CMP_CTRL register is set to ‘0’, the CMPINTF bit in CMPn_STAT is set according to the INTPOL bit and INTTYPE[1:0] bit in CMPn_CONF register. However, the comparator interrupt is not generated.

19.6 Comparator Registers

The base address of the Comparator (CMP) and register map are described in the following:

Table 195. Base Address of CMP

Name	Base Address
CMP0	0x4000_B380
CMP1	0x4000_B390
CMP2	0x4000_B3A0
CMP3	0x4000_B3B0

Table 196. CMP Register Map

Name	Offset	Type	Description	Reset Value	Reference
CMPn_CONF	0x0000	RW	CMP n Configuration Register	0x0000_0000	19.6.1
CMPn_CTRL	0x0004	RW	CMP n Control Register	0x0000_0000	19.6.5
CMPn_STAT	0x0008	RW	CMP n Status Register	0x0000_0000	19.6.6

NOTE:

1. n = 0, 1, 2, and 3

19.6.1 CMP0_CONF: Comparator 0 Configure Register

The CMP0_CONF register is used for overall settings of the Comparator.

CMP0_CONF=0x4000_B380

31 30	29 28 27 26 25 24	23 22 21	20	19 18	17 16	15 14 13 12 11	10	9 8	7 6	5 4	3 2	1 0
FLTDIV[1:0]	FLTSEL[5:0]	Reserved	HYSEN	Reserved	HYSSEL[1:0]	Reserved	INTPOL	INTTYPE[1:0]	Reserved	CINNSEL[1:0]	Reserved	CINPSEL[1:0]
00	000000	-	0	-	00	-	0	00	-	00	-	00
RW	RW	-	RW	-	RW	-	RW	RW	-	RW	-	RW

31 30	FLTDIV[1:0]	Digital filter clock divider Filter Clock = (1 / PCLK) * (2 ^ FLTDIV[1:0])
29 24	FLTSEL[5:0]	Digital filter count Filtered by the amount of FLTSEL[5:0] value multiplied by the comparator filter clock value. If the FLTSEL[5:0] value = '2' and the comparator filter clock value = 1 MHz, then the corresponding time value of 2 MHz is 0.5 us and the signal must be maintained for 0.5us or longer to generate the flag.
20	HYSEN	Hysteresis enable bit 0 Disables. 1 Enables.
17 16	HYSSEL[1:0]	Hysteresis selection bit 00 40 mV Hysteresis 01 80 mV Hysteresis 10 120 mV Hysteresis 11 160 mV Hysteresis
10	INTPOL	Interrupt polarity selection bit (The settings are irrespective to the selection of both edge.) 0 Low (Falling) 1 High (Rising)
9 8	INTTYPE[1:0]	Interrupt type selection bit 00 Disable 01 Level 10 Single edge 11 Both edge
5 4	CINNSEL[1:0]	Reference (input -) selection bit 00 CP0B 01 DAC0O 10 DAC1O 11 V_{REFINT} (0.6 V)
1 0	CINPSEL[1:0]	Source (Input +) selection bit 00 CP0A0 01 CP0A1 10 CP0A2 11 V_{REFINT} (0.6 V)

NOTES:

1. Refer to 19.3.2 for more information about CINPSEL[1:0], CINNSEL[1:0], INTTYPE[1:0], and INTPOL.
2. During flash erase or program operation, it is not allowed to change the input to V_{REFINT} or from V_{REFINT} to another input by using the CINNSEL[1:0] or CINPSEL[1:0] bits.

19.6.2 CMP1_CONF: Comparator 1 Configure Register

The CMP1_CONF register is used for overall settings of the Comparator.

CMP1_CONF=0x4000_B390

31 30	29 28 27 26 25 24	23 22 21	20	19 18	17 16	15 14 13 12 11	10	9 8	7 6	5 4	3 2	1 0
FLTDIV[1:0]	FLTSEL[5:0]	Reserved	HYSEN	Reserved	HYSSEL[1:0]	Reserved	INTPOL	INTTYPE[1:0]	Reserved	CINNSEL[1:0]	Reserved	CINPSEL[1:0]
00	000000	-	0	-	00	-	0	00	-	00	-	00
RW	RW	-	RW	-	RW	-	RW	RW	-	RW	-	RW

31 30	FLTDIV[1:0]	Digital filter clock divider Filter Clock = (1 / PCLK) * (2 ^ FLTDIV[1:0])
29 24	FLTSEL[5:0]	Digital filter count Filtered by the amount of FLTSEL[5:0] value multiplied by the comparator filter clock value. If the FLTSEL[5:0] value = '2' and the comparator filter clock value = 1 MHz, then the corresponding time value of 2 MHz is 0.5 us and the signal must be maintained for 0.5us or longer to generate the flag.
20	HYSEN	Hysteresis enable bit 0 Disables. 1 Enables.
17 16	HYSSEL[1:0]	Hysteresis selection bit 00 40 mV Hysteresis 01 80 mV Hysteresis 10 120 mV Hysteresis 11 160 mV Hysteresis
10	INTPOL	Interrupt polarity selection bit (The settings are irrespective to the selection of both edge.) 0 Low (Falling) 1 High (Rising)
9 8	INTTYPE[1:0]	Interrupt type selection bit 00 Disable 01 Level 10 Single edge 11 Both edge
5 4	CINNSEL[1:0]	Reference (input -) selection bit 00 CP1B 01 DAC0O 10 DAC1O 11 V_{REFINT} (0.6 V)
1 0	CINPSEL[1:0]	Source (Input +) selection bit 00 CP1A0 01 CP1A1 10 CP1A2 11 V_{REFINT} (0.6 V)

NOTES:

1. Refer to 19.3.2 for more information about CINPSEL[1:0], CINNSEL[1:0], INTTYPE[1:0], and INTPOL.
2. During flash erase or program operation, it is not allowed to change the input to V_{REFINT} or from V_{REFINT} to another input by using the CINNSEL[1:0] or CINPSEL[1:0] bits.

19.6.3 CMP2_CONF: Comparator 2 Configure Register

The CMP2_CONF register is used for overall settings of the Comparator.

CMP2_CONF=0x4000_B3A0

31 30	29 28 27 26 25 24	23 22 21	20	19 18	17 16	15 14 13 12 11	10	9 8	7 6	5 4	3 2	1 0
FLTDIV[1:0]	FLTSEL[5:0]	Reserved	HYSEN	Reserved	HYSSEL[1:0]	Reserved	INTPOL	INTTYPE[1:0]	Reserved	CINNSEL[1:0]	Reserved	CINPSEL[1:0]
00	000000	-	0	-	00	-	0	00	-	00	-	00
RW	RW	-	RW	-	RW	-	RW	RW	-	RW	-	RW

31 30	FLTDIV[1:0]	Digital filter clock divider Filter Clock = (1 / PCLK) * (2 ^ FLTDIV[1:0])
29 24	FLTSEL[5:0]	Digital filter count Filtered by the amount of FLTSEL[5:0] value multiplied by the comparator filter clock value. If the FLTSEL[5:0] value = '2' and the comparator filter clock value = 1 MHz, then the corresponding time value of 2 MHz is 0.5 us and the signal must be maintained for 0.5us or longer to generate the flag.
20	HYSEN	Hysteresis enable bit 0 Disables. 1 Enables.
17 16	HYSSEL[1:0]	Hysteresis selection bit 00 40 mV Hysteresis 01 80 mV Hysteresis 10 120 mV Hysteresis 11 160 mV Hysteresis
10	INTPOL	Interrupt polarity selection bit (The settings are irrespective to the selection of both edge.) 0 Low (Falling) 1 High (Rising)
9 8	INTTYPE[1:0]	Interrupt type selection bit 00 Disable 01 Level 10 Single edge 11 Both edge
5 4	CINNSEL[1:0]	Reference (input -) selection bit 00 CP2B 01 DAC0O 10 DAC1O 11 V_{REFINT} (0.6 V)
1 0	CINPSEL[1:0]	Source (Input +) selection bit 00 CP2A 01 DAC0O 10 DAC1O 11 V_{REFINT} (0.6 V)

NOTES:

1. Refer to 19.3.2 for more information about CINPSEL[1:0], CINNSEL[1:0], INTTYPE[1:0], and INTPOL.
2. During flash erase or program operation, it is not allowed to change the input to V_{REFINT} or from V_{REFINT} to another input by using the CINNSEL[1:0] or CINPSEL[1:0] bits.

19.6.4 CMP3_CONF: Comparator 3 Configure Register

The CMP3_CONF register is used for overall settings of the Comparator.

CMP3_CONF=0x4000_B3B0

31 30	29 28 27 26 25 24	23 22 21	20	19 18	17 16	15 14 13 12 11	10	9 8	7 6	5 4	3 2	1 0
FLTDIV[1:0]	FLTSEL[5:0]	Reserved	HYSEN	Reserved	HYSSEL[1:0]	Reserved	INTPOL	INTTYPE[1:0]	Reserved	CINNSEL[1:0]	Reserved	CINPSEL[1:0]
00	000000	-	0	-	00	-	0	00	-	00	-	00
RW	RW	-	RW	-	RW	-	RW	RW	-	RW	-	RW

31 30	FLTDIV[1:0]	Digital filter clock divider Filter Clock = (1 / PCLK) * (2 ^ FLTDIV[1:0])
29 24	FLTSEL[5:0]	Digital filter count Filtered by the amount of FLTSEL[5:0] value multiplied by the comparator filter clock value. If the FLTSEL[5:0] value = '2' and the comparator filter clock value = 1 MHz, then the corresponding time value of 2 MHz is 0.5 us and the signal must be maintained for 0.5us or longer to generate the flag.
20	HYSEN	Hysteresis enable bit 0 Disables. 1 Enables.
17 16	HYSSEL[1:0]	Hysteresis selection bit 00 40 mV Hysteresis 01 80 mV Hysteresis 10 120 mV Hysteresis 11 160 mV Hysteresis
10	INTPOL	Interrupt polarity selection bit (The settings are irrespective to the selection of both edge.) 0 Low (Falling) 1 High (Rising)
9 8	INTTYPE[1:0]	Interrupt type selection bit 00 Disable 01 Level 10 Single edge 11 Both edge
5 4	CINNSEL[1:0]	Reference (input -) selection bit 00 CP3B 01 DAC0O 10 DAC1O 11 V_{REFINT} (0.6 V)
1 0	CINPSEL[1:0]	Source (Input +) selection bit 00 CP3A 01 DAC0O 10 DAC1O 11 V_{REFINT} (0.6 V)

NOTES:

1. Refer to 19.3.2 for more information about CINPSEL[1:0], CINNSEL[1:0], INTTYPE[1:0], and INTPOL.
2. During flash erase or program operation, it is not allowed to change the input to V_{REFINT} or from V_{REFINT} to another input by using the CINNSEL[1:0] or CINPSEL[1:0] bits.

19.6.5 CMPn_CTRL: Comparator n Control Register

The CMPn_CTRL register controls the comparator operation.

CMP0_CTRL=0x4000_B384, CMP1_CTRL=0x4000_B394
CMP2_CTRL=0x4000_B3A4, CMP3_CTRL=0x4000_B3B4

31 30 29 28 27 26 25 24	23 22 21 20 19 18 17 16	15 14 13 12 11 10 9	8	7 6 5 4 3 2 1	0
Reserved			CMPINTEN	Reserved	CMPEN
-			0	-	0
-			RW	-	RW

8	CMPINTEN	Comparator interrupt enable bit	
		0	Disable
		1	Enable
0	CMPEN	Comparator enable bit	
		0	Disable
		1	Enable

19.6.6 CMPn_STAT: Comparator n Status Register

The CMPn_STAT register displays the operation status of the Comparator.

CMP0_STAT=0x4000_B388, CMP1_STAT=0x4000_B398
CMP2_STAT=0x4000_B3A8, CMP3_STAT=0x4000_B3B8

31 30 29 28 27 26 25 24	23 22 21 20 19 18 17 16	15 14 13 12 11 10 9	8	7 6 5 4 3 2 1	0
Reserved			CMPINTF	Reserved	CMPFLAG
-			0	-	0
-			RWC1	-	RO

8	CMPINTF	Comparator interrupt flag bit	
		0	Not occurred
		1	occurred (Writing a '1' to the bit clears it to '0')
0	CMPFLAG	Comparator output status	
		0	Low
		1	High

19.6.7 Comparator Register Map Summary

Table 197. Comparator Register Map Summary

Offset	Register name	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0x00	CMPn_CONF	FLTDIV[1:0]		FLTSEL[5:0]						Res.	Res.	Res.	HYSEN	Res.	Res.	HYSSEL[1:0]		Res.	Res.	Res.	Res.	Res.	INTPOL	INTTYPE[1:0]		Res.	Res.	CINNSEL[1:0]		Res.	Res.	CINPSEL[1:0]	
	Reset value	0	0	0	0	0	0	0	0				0			0	0						0	0	0			0	0			0	0
0x04	CMPn_CTRL	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	CMPINTEN	Res.	Res.	Res.	Res.	Res.	Res.	Res.	CMPEN
	Reset value																								0								0
0x08	CMPn_STAT	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	CMPINTF	Res.	Res.	Res.	Res.	Res.	Res.	Res.	CMPFLAG
	Reset value																								0								0

20. Digital to Analog Converter (DAC)

20.1 DAC Introduction

The Digital-to-Analog Converter (DAC) in A34G43x series provides a function to convert 12-bit digital value to analog output level. The DACxO pin can be used as General-Purpose Input/Output (GPIO) when the DAC output is disconnected from the output pad and connected to on-chip peripheral.

20.2 DAC Main Features

The key features of the DAC module are listed below:

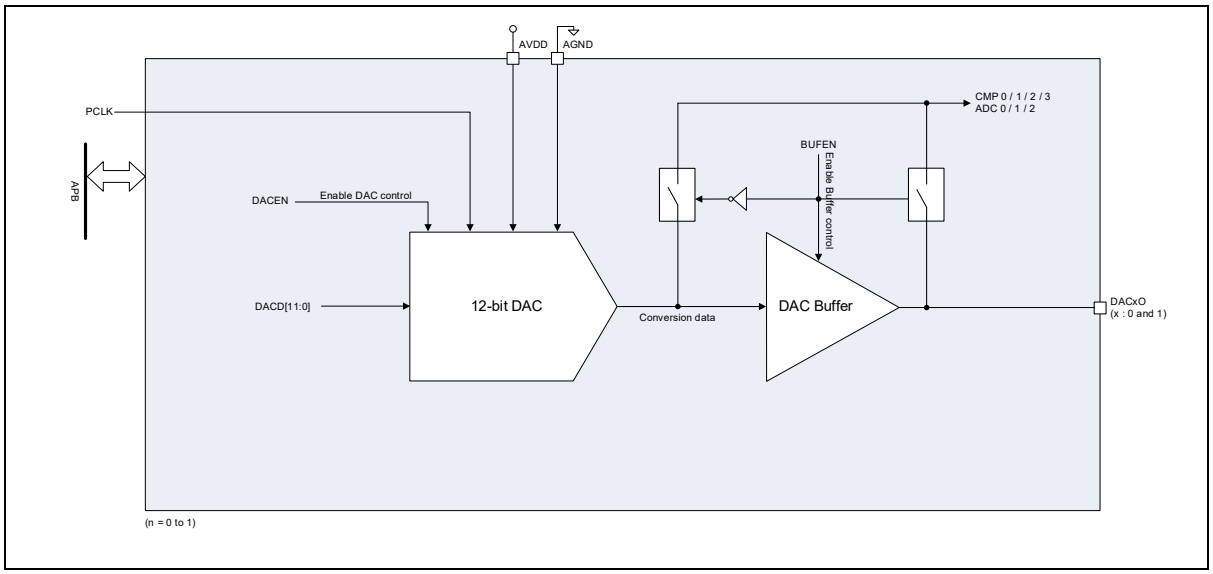
- 12-bit digital to analog resolution
- 1 μs conversion time
- The DAC output can be used as the source input or the reference input of comparator 0/1/2.
- The DAC output can be used as the input of ADC 0/1/2.

20.3 DAC Functional Description

20.3.1 12-bit DAC Block Diagram

Figure 222 shows a block diagram of the 12-bit DAC.

Figure 222. DAC Block Diagram



20.3.2 DAC Pins and Signals

Table 198 describes pins assigned for the DAC.

To use output ports for the DAC, users must change the mode using the PA_MR1 register in the Port Control Unit (PCU). More information about the PCU, please refer to chapter 5.

And the BUFEN bit in DACn_CR register must be set to ‘1’. (Refer to section 20.3.5.)

Table 198. Pin Assignment of DAC: External Pins

Pin Name	Type	Description	Supported Packages		
			A34G43xYL (120-LQFP)	A34G43xVL (100-LQFP)	A34G43xRL (64-LQFP)
AVDD	P	Analog VDD (2.7V to VDD)	O	O	O
AGND	P	Analog GND	O	O	O
DAC0O	O	Digital to analog output 0	O	O	O
DAC1O	O	Digital to analog output 1	O	O	O

NOTE:

1. P: Power, O: Output, A: Analog Input.

20.3.3 DAC Reset and Clock

The DAC module must be enabled before working, so the DAC0 and DAC1 bit in the SCU_PER2 and SCU_PCER2 register must be set to ‘1’.

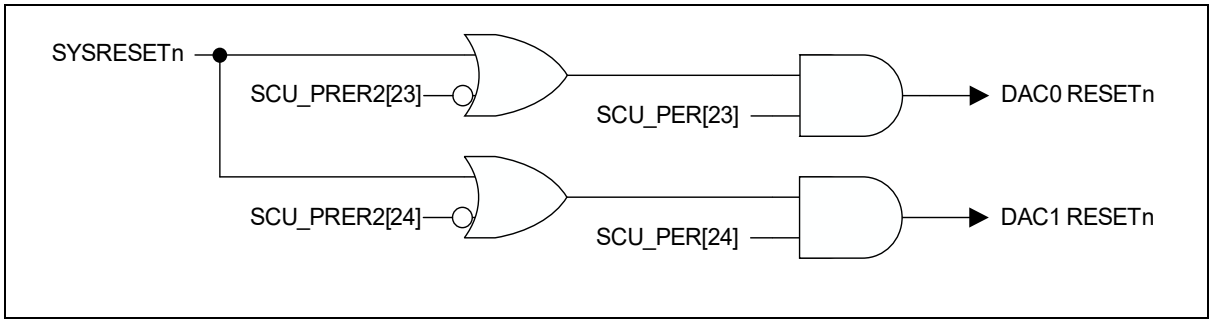
The DAC module can be reset by disabling and enabling again the DAC0 and DAC1 bit in the SCU_PER2 and SCU_PCER2 register, respectively.

If the corresponding reset mask bit in the SCU_PRER2 register is set to ‘0’, the settings for the DAC module can be maintained when the warm reset occurs.

Figure 223 shows a schematic diagram regarding the DAC reset signal.

For more information about the DAC reset and clocks, refer to section 4.4 and section 4.5.

Figure 223. Schematic Diagram of DAC Reset Signal



20.3.4 DAC Channel Enable

Each DAC channel can be powered on by setting its corresponding DAC_EN bit in the DACn_CR register. The DAC channel is then enabled after a startup time T_{WAKEUP}.

20.3.5 DAC Output Buffer Enable

The DAC integrates output buffers that can be used to reduce the output impedance, and to drive external loads directly without having to add an external operational amplifier. Each DAC channel output buffer can be enabled and disabled using the corresponding BUFEN bit in the DACn_CR register.

For DAC INL and DNL performance, the usage of the DAC output is determined based on the DAC buffer state. The DAC must be enabled before enabling the DAC buffer.

- Enable DAC Output Buffer
  - When the DAC buffer is enabled, the DAC value is always output to the port. Therefore, the DAC output port should only be used for DAC outputs. Accordingly, the ports corresponding to DAC0O and DAC1O in the PA_MR1 register must be set as DAC outputs.
  - The DAC output can be used as a reference voltage for the internal ADC and CMP. However, if the adjacent port of the DAC output port swings at high speed, it may affect the DAC output port. Therefore, using it as a reference for internal ADC and CMP is not recommended.
- Disable DAC Output Buffer
  - When the DAC buffer is disabled, the ports cannot be used as DAC output ports. However, they can be used as I/O or for other alternate functions.
  - The DAC output can only be used as a reference voltage for the internal ADC and CMP.

20.3.6 DAC Output Voltage

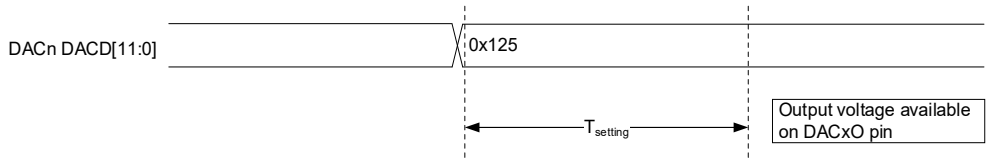
Digital inputs are converted to output voltages on a linear conversion between 0 V and AVDD. The analog output voltages on each DAC channel pin are determined by the following equation:

$$\mathrm{DAOn} = AVDD + \times \left(\frac{\mathrm{DIN}}{4096}\right)$$

20.3.7 DAC Conversion

When a value is written to the DACD in DACn_DR register, the analog output voltage becomes available after a settling time Tsetting, which varies depending on the power supply voltage and the analog output load.

Figure 224. DAC Conversion



NOTE:

1. Refer to DAC electrical characteristics in Datasheet

20.4 DAC Registers

The base address of the DAC and register map are described in the following:

Table 199. Base Address of DAC

Name	Base Address
DAC0	0x4000_B600
DAC1	0x4000_B700

Table 200. DAC Register Map

Name	Offset	Type	Description	Reset Value	Ref.
DAC_DR	0x0000	RW	DAC data register	0x0000_0000	20.4.1
DAC_CR	0x0004	RW	DAC control register	0x0000_0000	20.4.2

NOTE:

1. n = 0 and 1

20.4.1 DACn_DR: DAC n Data Register

The DACn_DR register is a 12-bit input data register of DACn.

DAC0_DR=0x4000_B600, DAC1_DR=0x4000_B700

31 30 29 28 27 26 25 24	23 22 21 20 19 18 17 16	15 14 13 12 11 10 9 8	7 6 5 4 3 2 1 0
Reserved			DACD[11:0]
-			0x000
-			RW

11 0	DACD[11:0]	D/A Converter Data

20.4.2 DACn_CR: DAC n Control Register

The DACn_CR register is used to control the DAC. Reserved bits must be '0'.

DAC0_CR=0x4000_B604, DAC1_DR=0x4000_B704

31 30 29 28 27 26 25 24	23 22 21 20 19 18 17 16	15 14 13 12 11 10	9	8	7 6 5 4 3 2 1 0
Reserved			BUFEN	DACEN	Reserved
-			0	0	-
-			RW	RW	-

9	BUFEN	DAC Buffer Enable Bit. Enable BUFEN after 20us from DACEN = '1'.	
		0	Disables.
		1	Enables.
8	DACEN	DAC Enable Bit.	
		0	Disables.
		1	Enables.

20.4.3 DAC Register Map Summary

Table 201. DAC Register Map Summary

Offset	Register name	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0x00	DACn_DR	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	DACD[11:0]											
	Reset value																					0	0	0	0	0	0	0	0	0	0	0	0
0x04	DACn_CR	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	BUFEN	DACEN	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.
	Reset value																							0	0								

21. Advanced Encryption Standard (AES-128)

21.1 Introduction

The A34G43x series has a built-in Advanced Encryption Standard-128 (AES-128) module that is used to encrypt or decrypt data.

The AES module supports DMA single transfer for the incoming and outgoing data through 32-bit AMBA APB (two DMA channels are required).

As a slave peripheral on the AMBA APB, the AES module allows only single access of 32-bit size. Writing other size than 32-bit may corrupt the register content.

21.2 AES Main Features

The AES module of the A34G43x series features the following:

- Advanced Encryption Standard-128 (AES-128) is supported.
- AES text input/output alignment mode configuration to support 8- or 32-bit data.
- Input and output FIFO flush modes.
- Encryption and decryption modes
- Interrupts available.
- Compatible with DMA transfers
- AES low latency processing (12 PCLK for encryption, 24 PCLK for decryption)
- 128-bit data block processing
- Support for cipher key lengths of 128-bit
- AMBA APB slave peripheral, accessible through 32-bit word single accesses only
- 128-bit register to store the cryptographic key (Four 32-bit registers)
- 32-bit buffer for data input and output
- Automatic data flow control with support of single-transfer Direct Memory Access (DMA) using two channels (one for incoming data, one for processed data)

21.2.1 AES Implementation

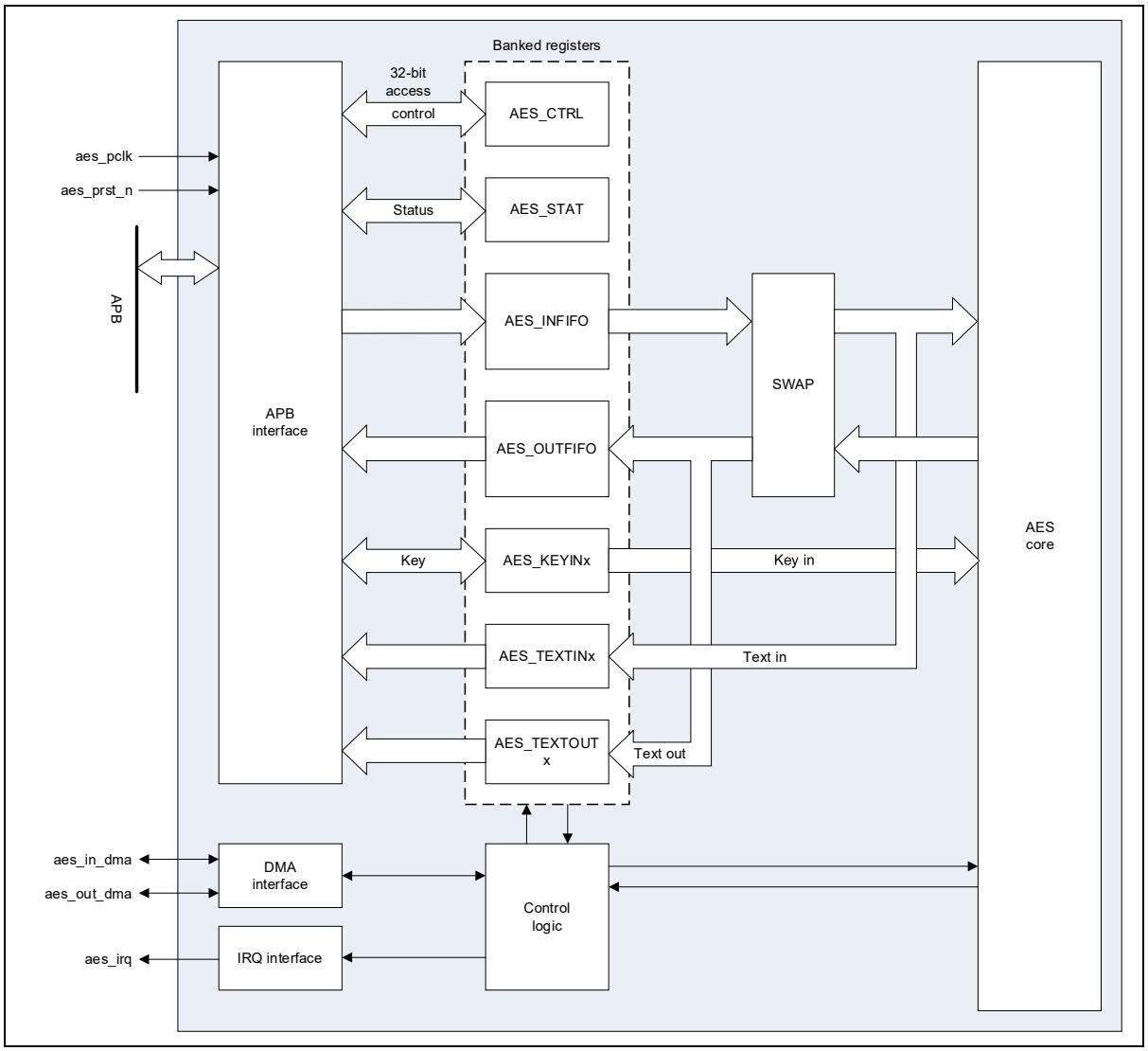
The A34G43x series includes a single instance of the AES module.

21.3 AES Functional Description

21.3.1 AES Block Diagram

Figure 225 shows a block diagram of the AES module.

Figure 225. AES Block Diagram



21.3.2 AES Internal Signals

Table 202 describes the internal signals that are connected to the AES module.

Table 202. AES Internal Input/Output Signals

Signal Name	Signal Type	Description
aes_prst_n	digital input	AES reset
aes_pclk	digital input	APB clock
aes_irq	digital output	AES interrupt request
aes_in_dma	digital input/output	Input DMA single request/acknowledge
aes_out_dma	digital input/output	Output DMA single request/acknowledge

21.3.3 AES Cryptographic Core

21.3.3.1 Overview

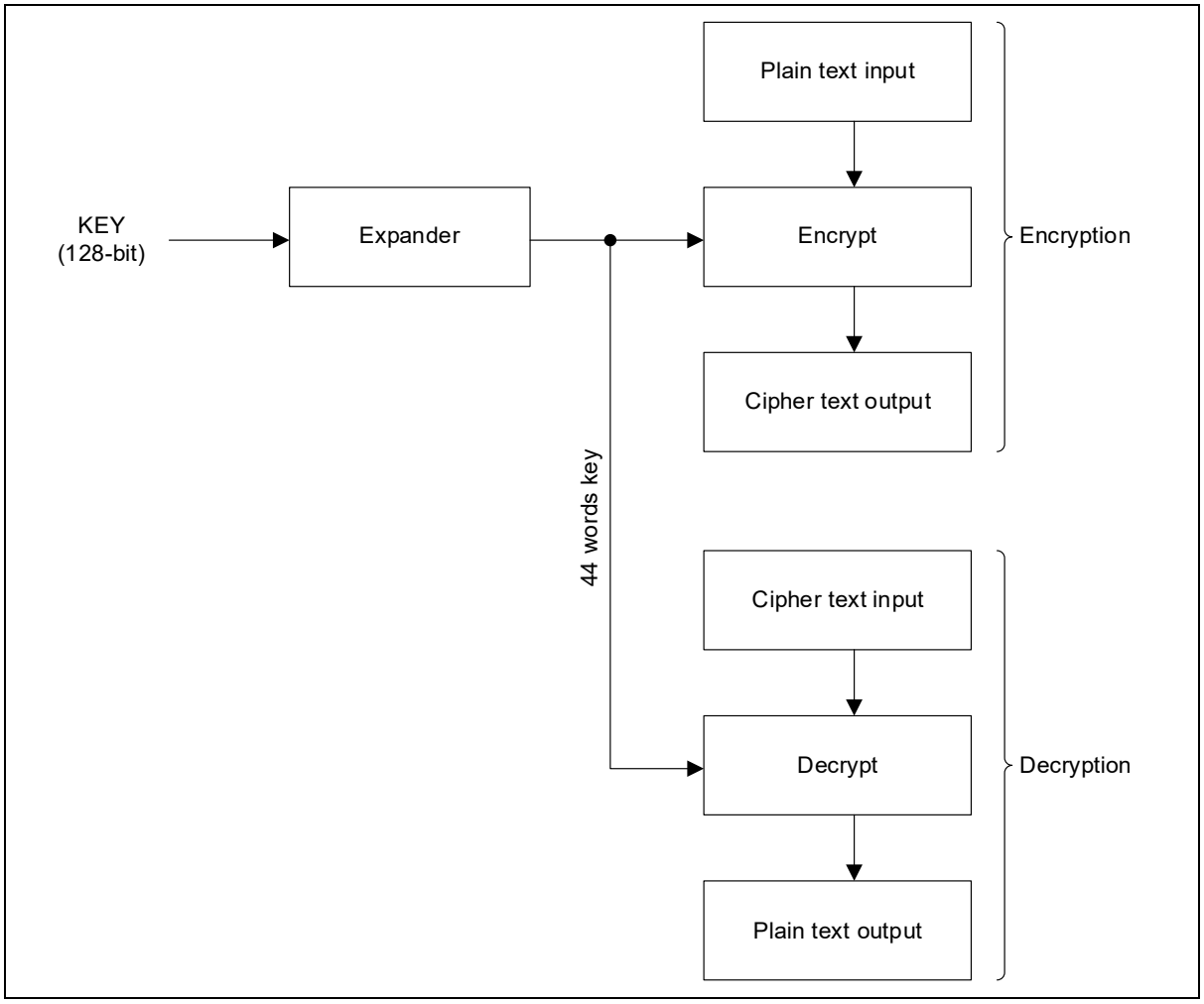
The AES cryptographic core consists of the following components:

- AES core algorithm
- Text input/output
- Key input

The AES core works on 128-bit data blocks (four words) with 128-bit key length.

Figure 226 shows an AES encryption and decryption operation flow.

Figure 226. AES Encryption and Decryption Operation Flow



21.3.3.2 Cipher Mode (Encryption)

In cipher mode, the AES module uses a 128-bit input key to encrypt a 128-bit plain text. It encrypts the plain text using the key, and outputs a 128-bit length cipher text as a result.

In cipher mode, the time required for encryption is 12 PCLKs.

21.3.3.3 Inverse Cipher Mode (Decryption)

In inverse cipher mode, the AES module uses a 128-bit key to decrypt a 128-bit cipher text. It decrypts the cipher text using the key, and outputs a 128-bit length plain text as a result.

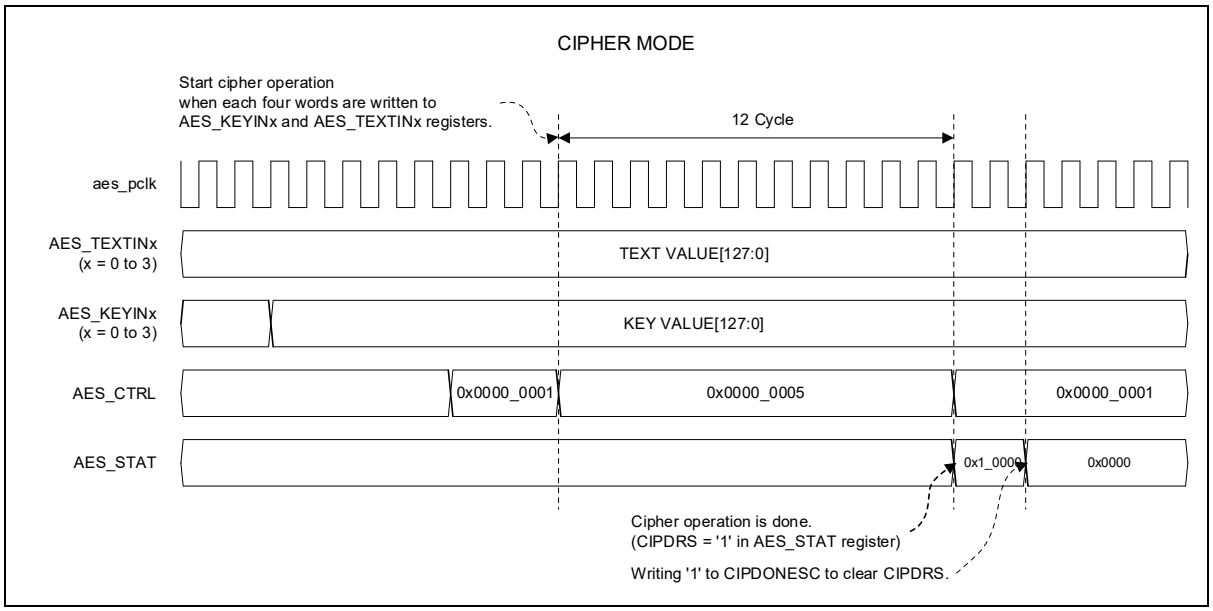
In inverse cipher mode, the time required for decryption is 24 PCLKs, which includes the time to calculate the key expansion of 44 words.

21.3.4 AES Procedure to Perform Cipher Operation

The operation in AES cipher mode can be enabled by following the procedure below:

1. Activate the AES module by setting the AES bit in the SCU_PER2 and SCU_PCER2 register bit to ‘1’.
2. Activate AES cipher mode by setting the MODE[1:0] bits in the AES_CTRL register to ‘01’.
3. Enter a 128-bit key in the AES_KEYINx register (x = 0 to 3).
4. Start the AES input/output FIFO flush operation by setting the INFIFOFLUSH and OUTFIFOFLUSH bits in the AES_CTRL register to ‘1’.
5. Enter a 128-bit plain text (four words) in the AES_INFIFO register, and then the AES encryption operation is started.
6. When the encryption is completed, the CIPDRS flag in the AES_STAT register set to ‘1’ and its corresponding flag is cleared when the CIPDONESC bit in the AES_STAT register is written to ‘1’.
7. As a result, the 128-bit cipher text is stored in the AES_TEXTOUTx register (x = 0 to 3).
8. According to length of plain text, step 4 to step 7 can be repeated.

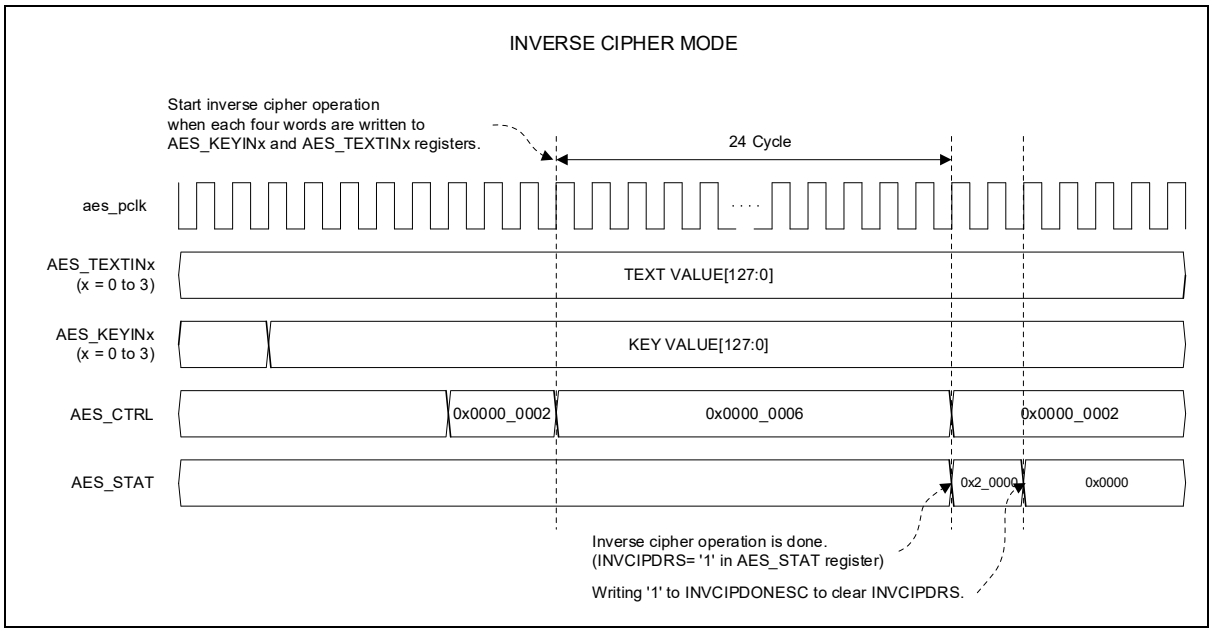
Figure 227. AES Timing Diagram



21.3.5 AES Procedure to Perform Inverse Cipher Operation

1. The operation in AES inverse cipher mode can be enabled by following the procedure below:
2. Activate the AES module by setting the AES bit in the SCU_PER2 and SCU_PCER2 register bit to ‘1’.
3. Activate AES inverse cipher mode by setting the MODE[1:0] bits in the AES_CTRL register to ‘10’.
4. Enter a 128-bit key in the AES_KEYINx register (x = 0 to 3).
5. Start the AES input/output FIFO flush operation by setting the INFIFOFLUSH and OUTFIFOFLUSH bits in the AES_CTRL register to ‘1’.
6. Enter a 128-bit cipher text (four words) in the AES_INFIFO register, and then the AES decryption operation starts.
7. When the decryption is completed, the INVCIPDRS flag in the AES_STAT register set to ‘1’ and its corresponding flag is cleared when the INVCIPDONESC bit in the AES_STAT register is written to ‘1’.
8. As a result, the 128-bit plain text is stored in the AES_TEXTOUTx register (x = 0 to 3).
9. According to length of cipher text, step 4 to step 7 can be repeated.

Figure 228. Inverse Cipher Mode Operation



21.3.6 AES Data Padding

When the AES module is operating, an incomplete input data block (that is, a block with input data shorter than 128-bits) must be padded with '0's prior to encryption or decryption. Extra bits must be appended to the trailing end of the input data string. After decryption, the extra bits must be discarded.

NOTE:

1. If you don't append padding data of '0's when using a block with input data shorter than 128 bits, encryption or decryption block don't operate.

21.3.7 AES Data Input and Output Registers

When a 128-bit data block is entered into the AES module with four successive 32-bit words using AES_INFIFO register. The data block is written into the text_in buffer[127:0] in the LSW first and MSW last order, where the LSW means the Least Significant Word (text_in buffer[31:0]) and the MSW means the Most Significant Word (text_in buffer[127:96]).

Table 203. text_in Buffer Endianness in AES_TEXTINx Registers (128-bit Length)

AES_TEXTIN3[31:0]	AES_TEXTIN2[31:0]	AES_TEXTIN1[31:0]	AES_TEXTIN0[31:0]
text_in[127:96]	text_in[95:64]	text_in[63:32]	text_in[31:0]

Similarly, a 128-bit data block is retrieved from the AES module with four successive 32-bit word by reading the AES_TEXTOUTx register (x = 0 to 3), where the LSW means the Least Significant Word (AES_ TEXTOUT0[31:0]) and the MSW means the Most Significant Word (AES_TEXTOUT3 [31:0]).

Table 204. text_out Buffer Endianness in AES_TEXTOUTx Registers (128-bit Length)

AES_TEXTOUT3[31:0]	AES_TEXTOUT2[31:0]	AES_TEXTOUT1[31:0]	AES_TEXTOUT0[31:0]
text_out[127:96]	text_out[95:64]	text_out[63:32]	text_out[31:0]

The 32-bit data word for the AES_INFIFO or AES_OUTFIFO register is organized in little endian order as described below:

- The least significant byte of a word to be written into the AES_INFIFO register must be located at the lowest address out of the four adjacent memory locations keeping the word to write.
- The least significant byte of a word to be read from the AES_OUTFIFO register must be located at the lowest address out of the four adjacent memory locations receiving the word.

When using DMA for the input data block, the four words of the input data block must be stored in the memory consecutively in little-endian order. That is, the least significant word of the input data block must be placed at the lowest address. For more information, refer to section 21.3.10.

NOTE:

1. Users can read AES processing output data through the AES_TEXTOUTx register (x = 0 to 3) in polling or interrupt mode. While User must read AES processing output data by reading the AES_OUTFIFO register in DMA mode.

21.3.8 AES Data Swapping

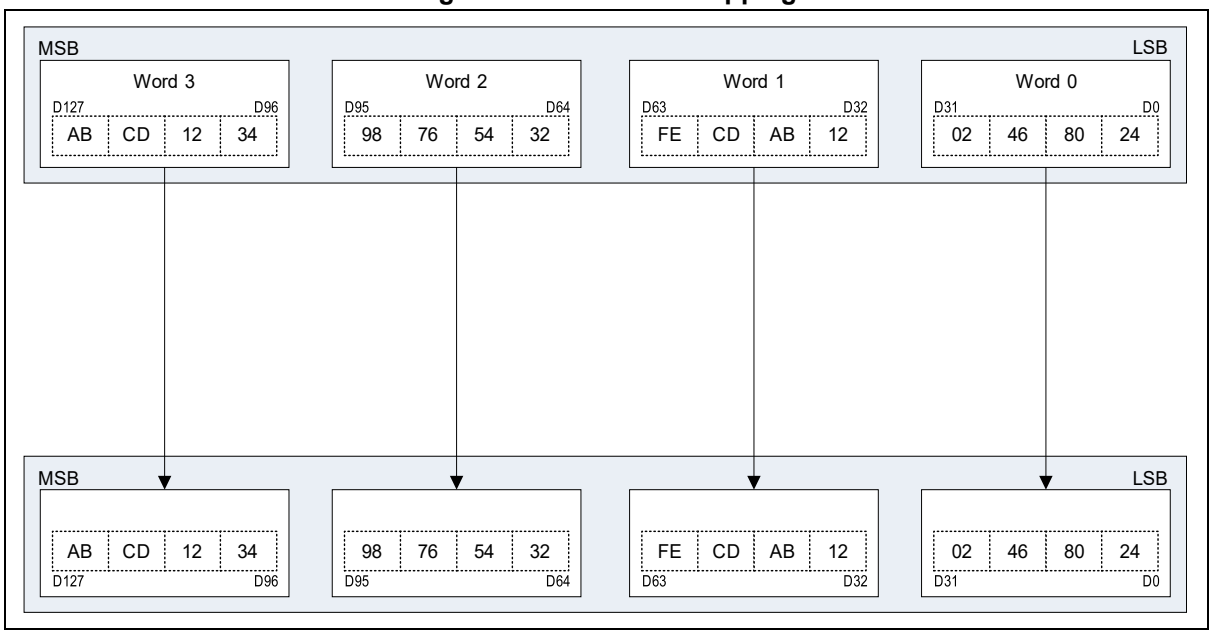
The AES module can be configured to perform data swapping with options of byte, word, or no swapping. The data swapping options can be applied for the input data before loading into the AES processing core as well as for the data output from the AES processing core before sending to the AES_OUTFIFO register. One of the options is selected according to the data type.

The data swapping type can be selected by configuring the TIINVMD[2:0] and TOINVMD[2:0] bits in the AES_CTRL register. This selection is applied to the input and output of the AES core, respectively.

Figure 230, Figure 231 and Figure 232 outline three types of data swapping that is processed in the SWAP block between the AES processing core and the input/output data buffer.

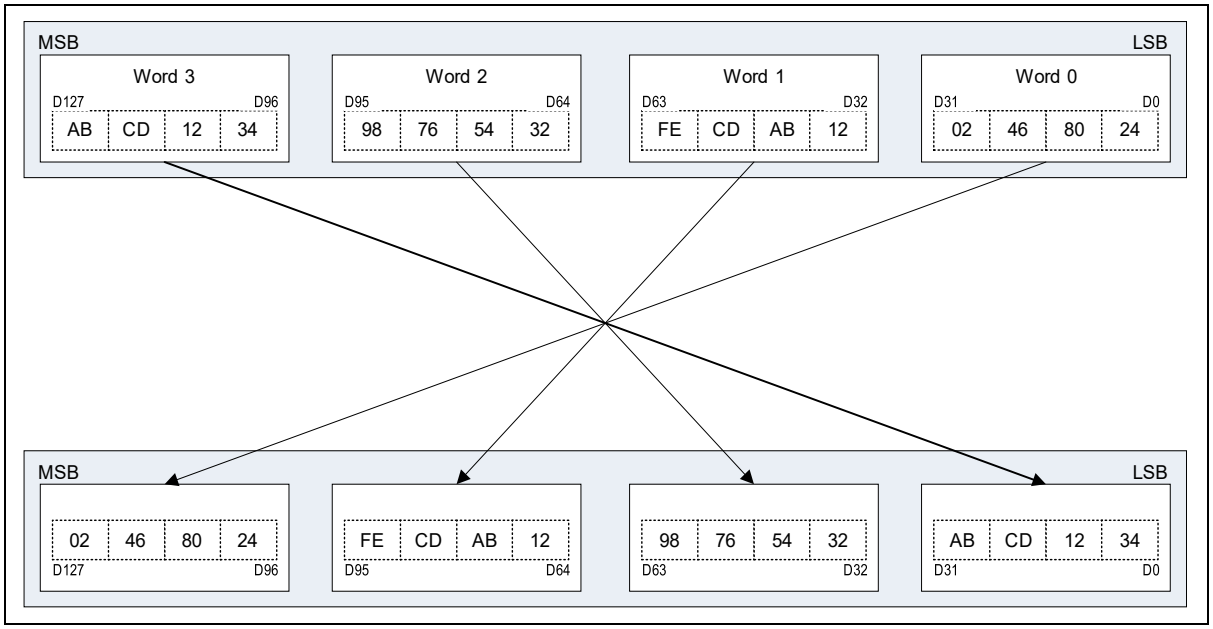
For encryption, the SWAP block receives input data from the AES_INFIFO register and outputs the result toward the AES processing core input buffer (text_in buffer[127:0]). Similarly for decryption, it receives data from the AES processing core output buffer (text_out buffer[127:0]) and outputs the result towards the AES_OUTFIFO register.

Figure 229. No Data Swapping



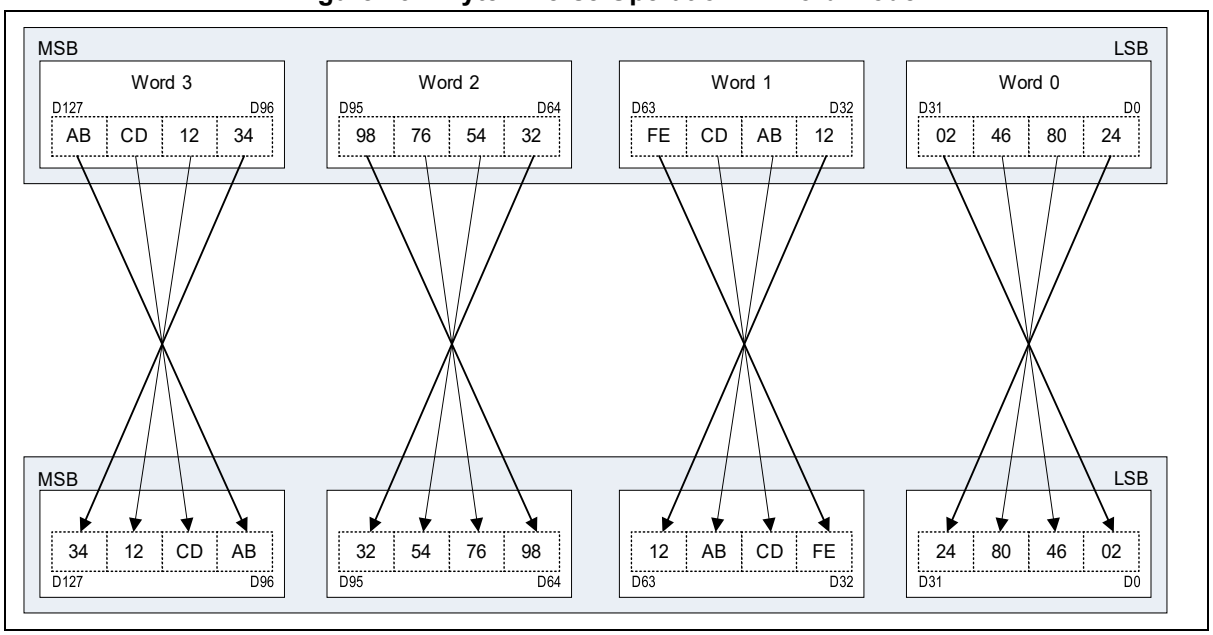
21.3.8.1 Word Inverse Mode Operation

Figure 230. Word Inverse Mode Operation



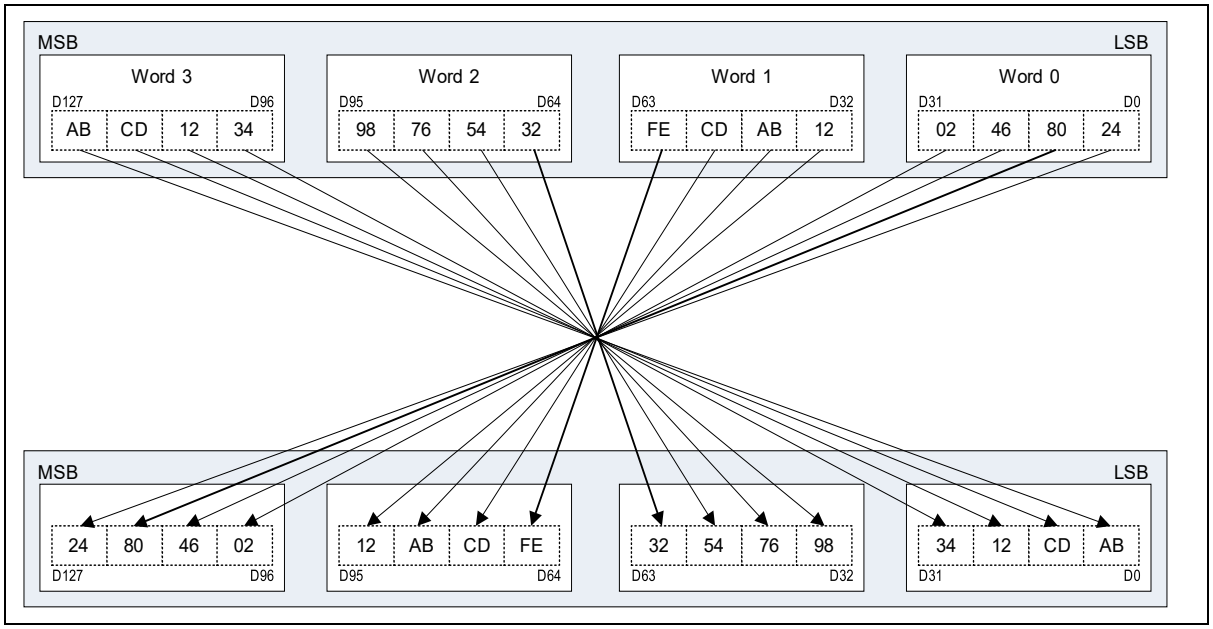
21.3.8.2 Byte Inverse Mode Operation in Word Mode

Figure 231. Byte Inverse Operation in Word Mode



21.3.8.3 Byte Inverse Mode Operation

Figure 232. Byte Inverse Mode Operation



21.3.9 AES Key Registers

128-bit size key used for encryption and decryption is stored in the AES_KEYINx registers (x = 0 to 3). The key is written to or read from the AES_KEYINx registers which organize the data in memory in little-endian order. That is, the 128-bit size key is spread over the four 32-bit registers which store the least significant byte at the lowest address.

Table 205 shows that the 128-bit key is segmented and stored in four AES_KEYINx registers (x = 0 to 3).

Table 205. Key Endianness in AES_KEYINx Registers (128-bit Key Length)

AES_KEYIN3[31:0]	AES_KEYIN2[31:0]	AES_KEYIN1[31:0]	AES_KEYIN0[31:0]
KEY[127:96]	KEY[95:64]	KEY[63:32]	KEY[31:0]

The key for encryption and decryption can be written into these registers when the AES module and the clock enable bit are set enabled in the SCU_PER2 and SCU_PCER2 registers.

The key registers are not affected by text alignment that is controlled by the TIINVMD[2:0] and TOINVMD[2:0] bits in the AES_CTRL register.

21.3.10 AES DMA Interface

The AES module provides an interface to connect to the Direct Memory Access (DMA) controller. The DMA operation can be controlled by configuring the AES_CTRL register.

21.3.10.1 DMA Operation in Different Operating Modes

DMA operation is usable when cipher or inverse cipher mode is enabled in the MODE[1:0] bits in the AES_CTRL register.

DMA single request signals are generated by the AES module until it is disabled. Therefore, after the data output phase when the encryption or decryption process for the 128-bit data block is completed, the AES module automatically switches to new data input phase for the next data block.

If data transfers between the AES module and memory are managed by DMA, the CIPDRS flag and INVCIPDRS flag are not related to the data transfers and can be ignored (left set) by software. They can be cleared only when data transfers are managed by software.

21.3.11 AES Processing Latency

Table 206 shows a processing latency required to encrypt or decrypt a 128-bit block.

Table 206. Processing Latency (in Clock Cycle) for Encryption or Decryption

Key Size	Operation Mode	Computation Clock (PCLK)
128-bit	Encryption	12
	Decryption	24

21.4 AES Interrupts

There are four individual maskable interrupt sources generated by the AES module, which signal the following events:

- Cipher mode completed
- Inverse cipher mode completed
- DMA input completed (AES to Memory)
- DMA output completed (Memory to AES)

Each of the four interrupt sources listed above is combined into the common interrupt signal aes_irq that connects to the Nested Vectored Interrupt Controller (NVIC).

Each AES interrupt source can be individually enabled/disabled, by setting/clearing the corresponding enable bit of the AES_CTRL register (see Table 207. AES Interrupt Requests).

The status of the individual maskable interrupt sources is read from the AES_STAT register.

Table 207 gives a summary of the interrupt sources, their event flags and enable bits.

Table 207. AES Interrupt Requests

AES Interrupt Event	Event Flag	Enable Bit
Cipher mode completed interrupt flag	CIPDRS	CIPHERIE
Inverse cipher mode completed interrupt flag	INVCIPDRS	INVCIPHERIE
DMA input completed interrupt flag	DMAINRS	DMAINIE
DMA output completed interrupt flag	DMAOUTRS	DMAOUTIE

21.5 AES Registers

The base address and register map of the AES module are described in the following tables:

Table 208. Base Address of AES

Name	Base Address
AES	0x4000_0500

Table 209. AES Register Map

Name	Offset	Type	Description	Reset Value	Reference
AES_CTRL	0x0000	RW	AES Control Register	0x0000_0001	21.5.1
AES_STAT	0x0004	RW	AES Status Register	0x0000_0000	21.5.2
AES_INFIFO	0x0008	WO	AES Input FIFO Register	0x0000_0000	21.5.3
AES_OUTFIFO	0x000C	RO	AES Output FIFO Register	0x0000_0000	21.5.4
AES_KEYIN0	0x0010	RW	AES Key Input 0 Register	0x0000_0000	21.5.5
AES_KEYIN1	0x0014	RW	AES Key Input 1 Register	0x0000_0000	21.5.5
AES_KEYIN2	0x0018	RW	AES Key Input 2 Register	0x0000_0000	21.5.5
AES_KEYIN3	0x001C	RW	AES Key Input 3 Register	0x0000_0000	21.5.5
AES_TEXTIN0	0x0020	RO	AES Text Input 0 Register	0x0000_0000	21.5.6
AES_TEXTIN1	0x0024	RO	AES Text Input 1 Register	0x0000_0000	21.5.6
AES_TEXTIN2	0x0028	RO	AES Text Input 2 Register	0x0000_0000	21.5.6
AES_TEXTIN3	0x002C	RO	AES Text Input 3 Register	0x0000_0000	21.5.6
AES_TEXTOUT0	0x0030	RO	AES Text Output 0 Register	0x0000_0000	21.5.7
AES_TEXTOUT1	0x0034	RO	AES Text Output 1 Register	0x0000_0000	21.5.7
AES_TEXTOUT2	0x0038	RO	AES Text Output 2 Register	0x0000_0000	21.5.7
AES_TEXTOUT3	0x003C	RO	AES Text Output 3 Register	0x0000_0000	21.5.7

21.5.1 AES_CTRL: AES Control Register

The AES_CTRL register controls the AES module.

AES_CTRL=0x4000_0500

31 30 29 28 27 26 25 24	23 22 21 20	19	18	17	16	15 14 13 12 11 10	9	8	7 6 5	4 3 2	1 0
Reserved		DMAINIE	DMAOUTIE	INVCIPHERIE	CIPHERIE	Reserved	OUTFIFOFLUS	INFIFOFLUSH	TOINVMD[2:0]	TIINVMD[2:0]	MODE[1:0]
-		0	0	0	0	-	0	0	000	000	01
-		RW	RW	RW	RW	-	WO	WO	RW	RW	RW

19	DMAINIE	Enable the AES DMA input complete interrupt	
		0	Disables.
		1	Enables.
18	DMAOUTIE	Enable the AES DMA output complete interrupt	
		0	Disables.
		1	Enables.
17	INVCIPHERIE	Enable the AES inverse cipher mode complete interrupt	
		0	Disables.
		1	Enables.
16	CIPHERIE	Enable the AES cipher mode complete interrupt	
		0	Disables.
		1	Enables.
9	OUTFIFOFLUSH	AES output FIFO flush	
		0	No operation
		1	AES output FIFO is flushed. This bit is auto-cleared.
8	INFIFOFLUSH	AES input FIFO flush	
		0	No operation
		1	AES input FIFO is flushed. This bit is auto-cleared.
7 5	TOINVMD[2:0]	AES text out alignment mode configuration	
		000	Default word row (LSB)
		001	Word Inverse (MSB)
		010	Byte inverse in word
		100	Byte inverse
		Others	Default word row (LSB)
4 2	TIINVMD[2:0]	AES text in alignment mode configuration	
		000	Default word row (LSB)
		001	Word Inverse (MSB)
		010	Byte inverse in word
		100	Byte inverse
		Others	Default word row (LSB)
1 0	MODE[1:0]	AES mode selection	
		00	No operation
		01	Cipher mode
		10	Inverse cipher mode
		11	No operation

21.5.2 AES_STAT: AES Status Register

The AES_STAT[19:16] bits display four separate statuses of the AES module. Writing '1' to the corresponding bit of the AES_STAT[3:0] clears the corresponding status of the AES_STAT[19:16].

AES_STAT=0x4000_0504

31 30 29 28 27 26 25 24	23 22 21 20	19	18	17	16	15 14 13 12 11 10 9 8	7 6 5 4	3	2	1	0
Reserved		DMAINRS	DMAOUTRS	INVCIPDRS	CIPDRS	Reserved		DMAINSC	DMAOUTSC	INVCIPDONES	CIPDONESC
-		0	0	0	0	-		0	0	0	0
-		RO	RO	RO	RO	-		WO	WO	WO	WO

19	DMAINRS	AES DMA input done status This bit is cleared by writing a '1' to DMAINSC bit.	
		0	Not done.
		1	Done.
18	DMAOUTRS	AES DMA output done status This bit is cleared by writing a '1' to DMAOUTSC bit.	
		0	Not done.
		1	Done.
17	INVCIPDRS	AES inverse cipher done status This bit is cleared by writing a '1' to INVCIPDONESC bit.	
		0	Not done.
		1	Done.
16	CIPDRS	AES cipher done status This bit is cleared by writing a '1' to CIPDONESC bit.	
		0	Not done.
		1	Done.
3	DMAINSC	AES DMA input done status clear	
		0	No operation
		1	Writing a '1' to this bit clears the DMAINRS bit. This bit is auto-cleared.
2	DMAOUTSC	AES DMA output done status clear	
		0	No operation
		1	Writing a '1' to this bit clears the DMAOUTRS bit. This bit is auto-cleared.
1	INVCIPDONESC	AES inverse cipher done status clear	
		0	No operation
		1	Writing a '1' to this bit clears the INVCIPDRS bit. This bit is auto-cleared.
0	CIPDONESC	AES cipher done status clear	
		0	No operation
		1	Writing a '1' to this bit clears the CIPDRS bit. This bit is auto-cleared.

21.5.3 AES_INFIFO: AES Input FIFO Register

The AES_INFIFO register is the AES text input FIFO register.

AES_INFIFO=0x4000_0508

31 30 29 28 27 26 25 24	23 22 21 20 19 18 17 16	15 14 13 12 11 10 9 8	7 6 5 4 3 2 1 0
INFIFO[31:0]			
0x0000_0000			
WO			

31 0	INFIFO[31:0]	Text in FIFO register Once 128-bit data is entered by feeding word-size (32-bit) data four times, the input data is automatically loaded. You can check this data in the AES_TEXTINx register. (x = 0 to 3)

21.5.4 AES_OUTFIFO: AES Input FIFO Register

The AES_OUTFIFO register is the AES text output FIFO register.

AES_OUTFIFO=0x4000_050C

31 30 29 28 27 26 25 24	23 22 21 20 19 18 17 16	15 14 13 12 11 10 9 8	7 6 5 4 3 2 1 0
OUTFIFO[31:0]			
0x0000_0000			
RO			

31 0	OUTFIFO[31:0]	Text out FIFO register The 128-bit data is read in word-size (32-bit) four times in DMA operation mode. You can check this data in the AES_TEXTOUTx register. (x = 0 to 3)

21.5.5 AES_KEYINx: AES Key Input Register

The AES_KEYINx register is the AES key in the register. (x = 0 to 3)

AES_KEYIN0=0x4000_0510, AES_KEYIN1=0x4000_0514,
AES_KEYIN2=0x4000_0518, AES_KEYIN3=0x4000_051C

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
KEYINx[31:0]																															
0x0000_0000																															
RW																															

31 0	KEYINx[31:0] (x = 0 to 3)	Key in value register

21.5.6 AES_TEXTINx: AES Text Input Register

The AES_TEXTINx is the AES text in the register. (x = 0 to 3)

AES_TEXTIN0=0x4000_0520, AES_TEXTIN1=0x4000_0524
AES_TEXTIN2=0x4000_0528, AES_TEXTIN3=0x4000_052C

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
TEXTINx[31:0]																															
0x0000_0000																															
RO																															

31 0	TEXTINx[31:0] (x = 0 to 3)	Text in value register (read-only)

21.5.7 AES_TEXTOUTx: AES Text Output Register

The AES_TEXTOUTx register is the AES text out register. (x = 0 to 3)

AES_TEXTOUT0=0x4000_0530, AES_TEXTOUT1=0x4000_0534
AES_TEXTOUT2=0x4000_0538, AES_TEXTOUT3=0x4000_053C

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
TEXTOUTx[31:0]																															
0x0000_0000																															
R0																															

31 0	TEXTOUTx[31:0] (x = 0 to 3)	Text out value register (read-only)

21.5.8 AES Register Map Summary

Table 210. AES Register Map Summary

Offset	Register name	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0x00	AES_CTRL	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	DMAINIE	DMAOUTIE	INVCIPHERIE	CIPHERIE	Res.	Res.	Res.	Res.	Res.	Res.	OUTFIFOFLUSH	INFIFOFLUSH	TOINVMD[2:0]			TIINVMD[2:0]			MODE[1:0]	
	Reset value													0	0	0	0							0	0	0	0	0	0	0	0	0	1
0x04	AES_STAT	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	DMAINRS	DMAOUTRS	INVCIPDRS	CIPDRS	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	DMAINSC	DMAOUTSC	INVCIPDONESC	CIPDONESC
	Reset value													0	0	0	0													0	0	0	0
0x08	AES_INFIFO	INFIFO[31:0]																															
	Reset value	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
0x0C	AES_OUTFIFO	OUTFIFO[31:0]																															
	Reset value	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
0x10	AES_KEYIN0	KEYIN0[31:0]																															
	Reset value	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
0x14	AES_KEYIN1	KEYIN1[31:0]																															
	Reset value	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
0x18	AES_KEYIN2	KEYIN2[31:0]																															
	Reset value	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
0x1C	AES_KEYIN3	KEYIN3[31:0]																															
	Reset value	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
0x20	AES_TEXTIN0	TEXTIN0[31:0]																															
	Reset value	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
0x24	AES_TEXTIN1	TEXTIN1[31:0]																															
	Reset value	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
0x28	AES_TEXTIN2	TEXTIN2[31:0]																															
	Reset value	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
0x2C	AES_TEXTIN3	TEXTIN3[31:0]																															
	Reset value	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
0x30	AES_TEXTOUT0	TEXTOUT0[31:0]																															
	Reset value	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
0x34	AES_TEXTOUT1	TEXTOUT1[31:0]																															
	Reset value	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
0x38	AES_TEXTOUT2	TEXTOUT2[31:0]																															
	Reset value	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
0x3C	AES_TEXTOUT3	TEXTOUT3[31:0]																															
	Reset value	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

22. Random Number Generator (RNG)

22.1 RNG Introduction

The A34G43x series has a built-in Random Number Generator (RNG) that operates with an RNG clock, which users can select the clock source.

The RNG generates a random number based on the set seed value. If the users read data before the random number generation is completed, an error interrupt is flagged to prevent the users from reading the false data.

22.2 20.2 RNG Main Features

The RNG of A34G43x series features the following:

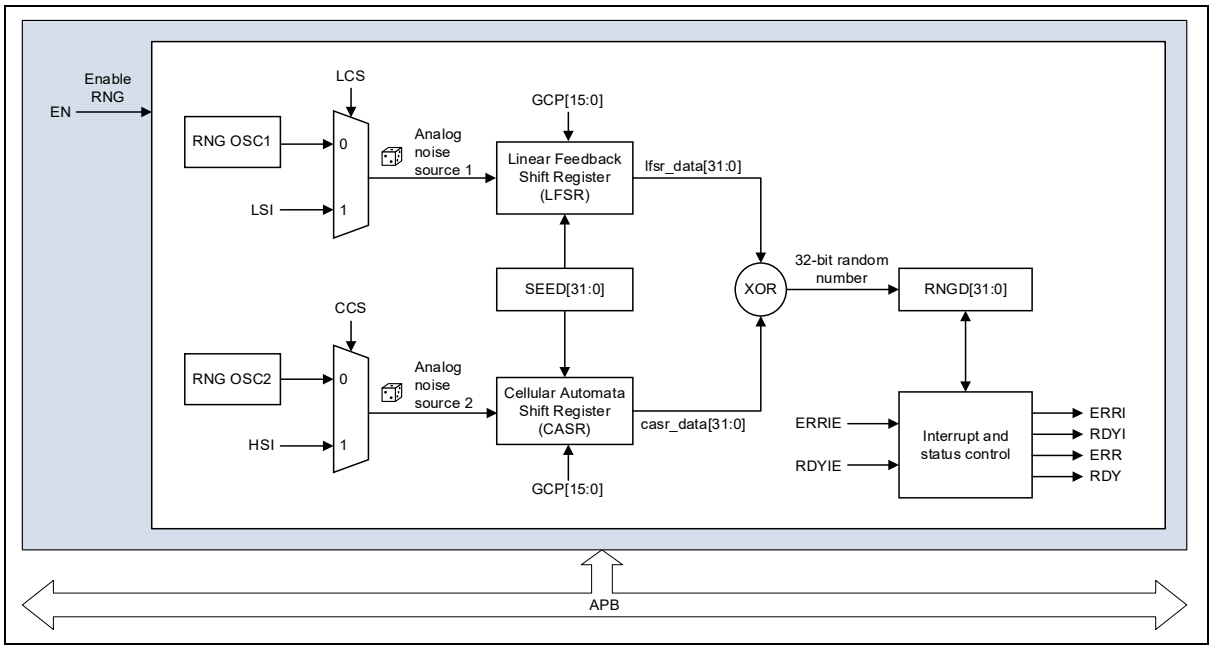
- Generation of 32-bit random number data
- Interrupt events:
  - Error interrupt
  - Ready interrupt
- Programmable random number-generating seed

22.3 RNG Functional Description

22.3.1 RNG Block Diagram

Figure 233 shows a block diagram of the RNG.

Figure 233. RNG Block Diagram



22.3.2 RNG Internal Signals

Table 211. RNG Internal Input and Output Signals

Signal Name	Signal Type	Description
rng_it	Digital output	RNG interrupt request
rng_pclk	Digital input	APB clock
internal_LFSR_ clock	Digital input	RNG dedicated clock, asynchronous to rng_pclk (Selectable between RNG OSC1 or LSI)
Internal_CASR_clock	Digital input	RNG dedicated clock, asynchronous to rng_pclk (Selectable between RNG OSC2 or HSI)

22.3.3 Random Number Generation

The 32-bit hardware RNG is based on a Linear Feedback Shift Register (LFSR) and a Cellular Automata Shift Register (CASR). Each shift register is clocked by an independent ring oscillator, and the output is sampled only when a new number is requested.

The LFSR has 43-bits, and a characteristic polynomial of $X^{43} + X^{41} + X^{20} + X + 1$. This is a primitive polynomial and gives a period length of 2^{43} - 1. The hardware RNG uses a 37-bit CASR, and its maximum length is $2^{37} – 1$.

To generate a random number, 32bits of the LFSR and CASR are selected and permuted and then XORed together.

The ERR in RNG_STAT register shows the error status indicating whether an error has occurred or not while generating a random number.

Users can find more information about the error status, by referring to section 22.3.8.

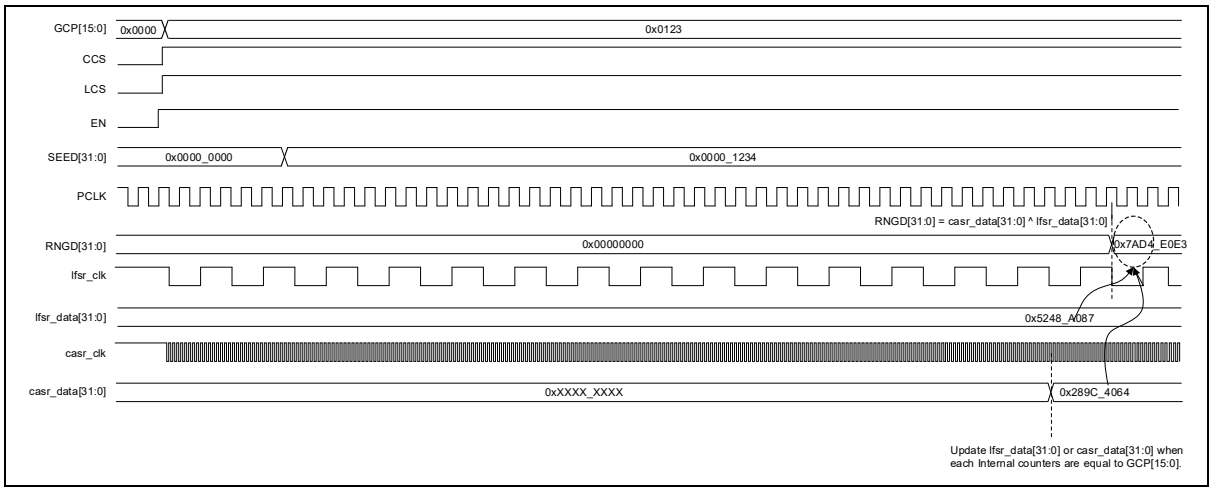
The interrupt can be generated when an error is detected.

22.3.4 RNG Initialization

Figure 234 shows that a series of events shown below occur after enabling the RNG (the EN in RNG_CTRL register is set to ‘1’).

- Register settings:
  - RNG_SEED register = 0x0000_1234
  - GCP[15:0] in RNG_CTRL register = 0x0123
  - LCS in RNG_CTRL register = ‘1’ (clock selection: LSI)
  - CCS in RNG_CTRL register = ‘1’ (clock selection: HSI)

Figure 234. Event Occurrences with EN in RNG_CTRL Register Set to ‘1’



1. By referencing the Generation Counter Parameter set in the GCP[15:0] in RNG_CTRL register using the seed value set in the RNG_SEED register, the internal LFSR_DATA and CASR_DATA are generated. At this time, the RDY in RNG_STAT register is set to ‘1’, and the LFSR_DATA and CASR_DATA, which are just generated, are XORed and output to the RNG_RNGD register.
2. Once the RNG value is generated, the RDY in RNG_STAT register becomes ‘0’ when reading the RNG_RNGD register, and the LFSR counter, CASR counter, and the RNG_RNGD register are cleared to ‘0’.

22.3.5 RNG Operation

22.3.5.1 Normal Operations

1. To use interrupts to run the RNG, we recommend that users follow the order below:
2. Set the seed value by configuring the RNG_SEED register.
3. Set the GCP[15:0] bits in the RNG_CTRL register to determine the LFSR and CASR generation time.
4. Set the ERRIE and RDYIE bits in the RNG_CTRL register to enable interrupts. At the same time, set the EN bit in the RNG_CTRL register to '1' to activate the RNG.
5. Once the steps 1, 2, and 3 are set, an interrupt is generated when a random number is ready or an error occurs. Therefore, check the following for each interrupt.
   A. No error occurs. The ERRI flag in the RNG_STAT register is set to '0'.
   B. A random number is ready. The RDYI flag in the RNG_STAT register is set to '1'.
      i. If the two conditions (step 1 and step 2) are true, the RNG_RNGD register can be read to obtain the random number generated by the RNG.
      ii. If one or both two conditions (step 1 and step 2) are false, the RNG_RNGD register must not be read.
      iii. Refer to section 22.3.8.

The RNG in Polling mode requires the sequence below for execution:

1. Set the seed value by configuring the RNG_SEED register.
2. Set the GCP[15:0] bits in the RNG_CTRL register to determine the LFSR and CASR generation time.
3. Set the EN bit in the RNG_CTRL register to '1' to activate the RNG.
4. Read the RNG_STAT register and check the following.
   A. No error occurs. The ERR flag in the RNG_STAT register is set to '0'.
   B. A random number is ready. The RDY flag in the RNG_STAT bit is set to '1'.
      i. If the two conditions (step 1 and 2) are true, the RNG_RNGD register can be read to obtain the random number generated by the RNG.
      ii. If one or two conditions (step 1 and 2) are false, the RNG_RNGD register must not be read.
      iii. Refer to 22.3.8 Error management to run the error recovery sequence in the event of an error.

NOTE:

1. If data is not ready (RDY = '0'), the RNG_RNGD register returns '0'.

22.3.5.2 Low-Power Operations

When entering DEEP-SLEEP mode, the RNG goes under Low-Power mode and the EN bit in the RNG_CTRL register is set to ‘0’, which stops the RNG_STAT and RNG_RGND registers, the RNG counter, and the internal RNG OSC1 and RNG OSC2 outputs.

When the DEEP-SLEEP mode is released by the Wake-up signal, the EN bit in the RNG_CTRL register is set back to ‘1’ and the RNG resumes operation.

22.3.6 RNG Clocking

The RNG operates on two different clocks: the APB clock and the dedicated RNG clock. The APB clock is used to configure each component of the APB Bank Register such as the RNG_CTRL, RNG_SEED, RNG_RNGD, and RNG_STAT registers. The dedicated RNG clock is used for noise source sampling.

The RNG module can be reset by disabling and enabling again the RNG bit in the SCU_PER2 and SCU_PCER2 register, respectively.

22.3.7 RNG Processing Time

RNG processing time varies based on the GCP[15:0] in RNG_CTRL register value. The RDY flag in the RNG_STAT register is set to ‘1’ only when the internal LFSR counter and CASR counter match to the value set in the GCP[15:0] bits in the RNG_CTRL register field to generate two match signals.

The LFSR clock and CASR clock can be set by configuring the LCS and CCS bits in the RNG_CTRL register.

22.3.8 Error Management

If the RNG_RNGD is read before the RDY flag in the RNG_STAT register set to ‘1’, the ERR flag in the RNG_STAT register becomes to set to ‘1’ to enable the RNG error detection. In this case, if the ERRIE bit in the RNG_CTRL register is enabled, the interrupt is generated to detect the RNG error as soon as the ERRI flag in the RNG_STAT register is set to ‘1’.

22.3.9 RNG Low-Power Consumption in RUN Mode

If power consumption is important in the overall operation, users can deactivate the RNG by setting the EN bit in the RNG_CTRL register to ‘0’ when the RDY flag in the RNG_STAT register is ‘1’.

When the RNG is deactivated, analog seed generators and all logics using the RNG clock are disabled.

22.4 RNG Interrupts

The RNG generates interrupts from the events shown below:

- Error interrupt
- Ready interrupt

Table 212 shows the RNG interrupt requests. The RNG interrupts can be used with dedicated interrupt enable control bits.

Table 212. RNG Interrupt Requests

Interrupt Source	Interrupt Event	Event Flag	Enable Control Bit	Interrupt Clear Method
RNG	Error flag	ERRI	ERRIE	Write '1' to ERRI
	RNGD ready flag	RDYI	RDYIE	Write '1' to RDYI

Users can independently enable or disable each interrupt source by changing the ERRIE and RDYIE bits in the RNG_CTRL register. The status of individual interrupt sources can be read from the RNG_STAT register.

NOTE:

1. The interrupts are generated only if the RNG is enabled.

22.5 RNG Registers

The RNG generates interrupts from the events shown below:

The base address and register map of the AES module are described in the following tables:

Table 213. Base Address of RNG

Name	Base Address
RNG	0x4000_0A00

Table 214. RNG Register Map

Name	Offset	Type	Description	Reset Value	Reference
RNG_CTRL	0x0000	RW	RNG Control Register	0xFFFF_0000	22.5.1
RNG_SEED	0x0004	RW	RNG Seed Register	0x0000_0000	22.5.2
RNG_RNGD	0x0008	RO	RNG Random Number Data Register	0x0000_0000	22.5.3
RNG_STAT	0x000C	RW	RNG Status Register	0x0000_0000	22.5.4

22.5.1 RNG_CTRL: RNG Control Register

The RNG_CTRL register is used to control the RNG. It is 32 bits wide.

RNG_CTRL = 0X4000_0A00

31 30 29 28 27 26 25 24	23 22 21 20 19 18 17 16	15	14	13 12 11 10	9	8	7 6 5 4 3 2 1	0
GCP[15:0]		CCS	LCS	Reserved	ERRIE	RDYIE	Reserved	EN
0xFFFF		0	0	-	0	0	-	0
RW		RW	RW	-	RW	RW	-	RW

31 16	GCP[15:0]	Generation Counter Parameter Determines linear feedback shift register (LFSR) and cellular automata shift register (CASR) generation time.
15	CCS	CASR clock selection
	0	RNG OSC2
	1	HSI
14	LCS	LFSR clock selection
	0	RNG OSC1
	1	LSI
9	ERRIE	Enable the error interrupt (Occurs when RNG_RNGD is read while its value is not ready)
	0	Disable
	1	Enable
8	RDYIE	Enable the RNGD ready interrupt
	0	Disable
	1	Enable
0	EN	Enable the RNG
	0	Disable
	1	Enable

22.5.2 RNG_SEED: RNG Seed Register

The RNG_SEED register is used to set the RNG seed. It is 32 bits wide.

RNG_SEED=0x4000_0A04

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
SEED[31:0]																															
0x0000_0000																															
RW																															

31 0	SEED[31:0]	RNG seed setting bit
		Sets the RNG seed.

22.5.3 RNG_RNGD: RNG Random Number Data Register

The RNG_RNGD register displays the number data that has been generated by the RNG. It is 32 bits wide.

RNG_RNGD=0x4000_0A08

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RNGD[31:0]																															
0x0000_0000																															
RO																															

31 0	RNGD[31:0]	Random number generation data bit
		Displays the random number data that has been generated by the RNG.

22.5.4 RNG_STAT: RNG Status Register

RNG_STAT register displays the operating status of the RNG. It is 32 bits wide.

RNG_STAT=0x4000_0A0C

31 30 29 28 27 26 25 24	23 22 21 20 19 18 17 16	15 14 13 12 11 10	9	8	7 6 5 4 3 2	1	0
Reserved			ERRI	RDYI	Reserved	ERR	RDY
-			0	0	-	0	0
-			RWC1	RWC1	-	RWC1	RO

9	ERRI	Error interrupt flag	
		0	Not flagged.
		1	Flagged (Writing a ‘1’ to the bit clears the flag).
8	RDYI	RNGD ready interrupt flag	
		0	Not flagged.
		1	Flagged (Writing a ‘1’ to the bit clears the flag).
1	ERR	Error status	
		0	The error has not occurred.
		1	The error has occurred (Writing a ‘1’ to the bit clears the flag).
0	RDY	RNGD ready status	
		0	Not ready
		1	Ready

22.5.5 RNG Register Map Summary

Table 215. RNG Register Map Summary

Offset	Register name	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0x00	RNG_CTRL	GCP[15:0]																CCS	LCS	Res	Res	Res	Res	ERRIE	RDYIE	Res	Res	Res	Res	Res	Res	Res	EN
	Reset value	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	0	0					0	0								0
0x04	RNG_SEED	SEED[31:0]																															
	Reset value	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
0x08	RNG_RNGD	RNGD[31:0]																															
	Reset value	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
0x0C	RNG_STAT	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	ERRI	RDYI	Res	Res	Res	Res	Res	Res	ERR	RDY
	Reset value																							0	0							0	0

23. Cyclic Redundancy Check (CRC) Calculation Module

23.1 CRC Introduction

A Cyclic Redundancy Check (CRC) module is used to get 32-bit, 16-bit USB, 16-bit CCITT and 8-bit CCITT codes from 8-bit or 32-bit data size. Application programs employ CRC-based technologies to examine the integrity of data transfers, storages, and flash memories in conformance with functional safety standards.

Using this module, it is possible to examine the integrity of transfer data and flash memory data.

Specifically, the CRC module of the A34G43x supports DMA.

23.2 CRC Main Features

The CRC of the A34G43x series features the following:

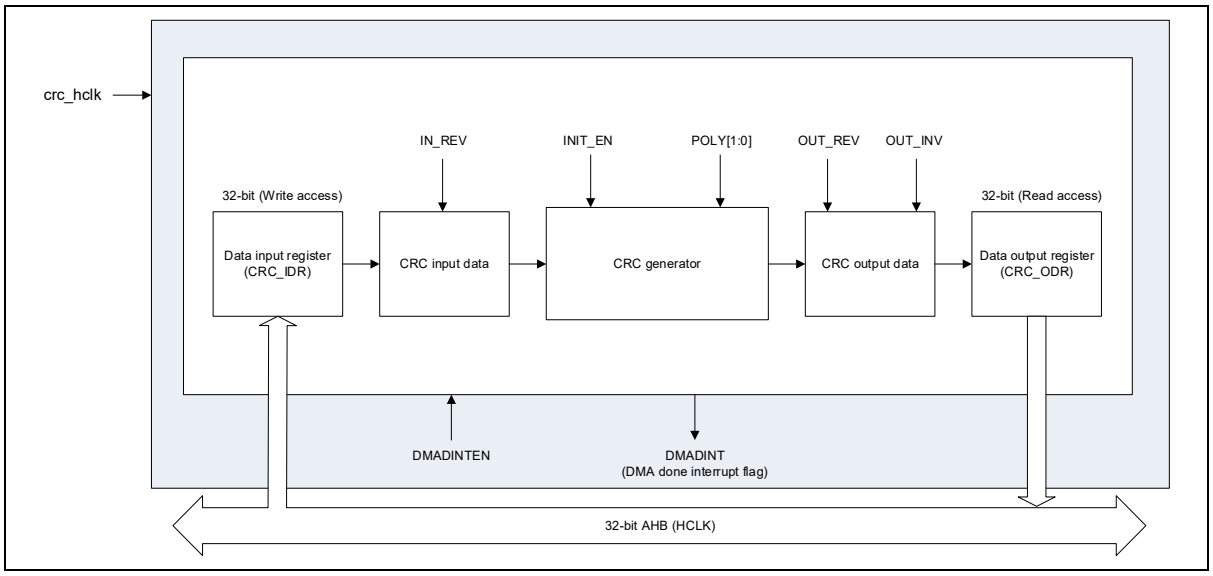
- Handles 8-, 32-bit data size
- Input buffer to avoid bus stall during calculation
- CRC calculation is automatically done for word/byte input data size.
  - Four AHB clock cycles (HCLK) for 32-bit
  - Two AHB clock cycles (HCLK) for 16-bit
  - One AHB clock cycle (HCLK) for 8-bit
- CRC code selectable by POLY[1:0]
  - CRC-32 (0x04C1_1DB7) when POLY[1:0] is set to ‘0’
  - CRC-16 USB (0x8005) when POLY[1:0] is set to ‘1’
  - CRC-16 CCITT (0x1021) when POLY[1:0] is set to ‘2’
  - CRC-8 CCITT (0x07) when POLY[1:0] is set to ‘3’
- Reversibility option on Input data by the IN_REV
- Reversibility option on Output data by the OUT_REV
- Invert option on Output data by the OUT_INV

23.3 CRC Functional Description

23.3.1 CRC Block Diagram

Figure 235 shows a block diagram of the A34G43x CRC module.

Figure 235. CRC Block Diagram



23.3.2 CRC Internal Signals

Table 216. CRC Internal Input/Output Signals

Signal Name	Signal Type	Description
crc_hclk	clock	AHB clock

23.3.3 Timing Diagram of CRC Internal Signals for Data Size

Figures in this section show timing diagrams of the CRC's internal operations. Each figure describes AHB clock cycles for different CRC data sizes.

Figure 236 shows a timing diagram of 8-bit size input data and CRC-8 calculation (outputs). Signals in Figure 236 such as the HCLK (crc_hclk), CRC enable bit (crc_cal_en), input data (IDR[7:0]), and output data (ODR[7:0] are internal signals, and show the cycles required for internal calculation.

The CRC-8 calculated values are output every 1 AHB clock cycle.

Figure 236. Timing Diagram of CRC Internal Signals: 8-bit Size Input and CRC-8 Outputs

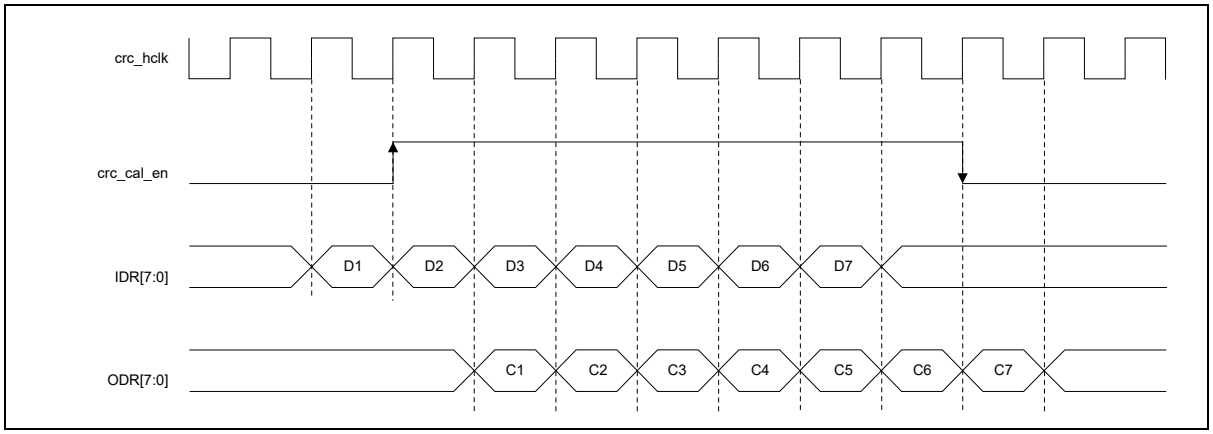


Figure 237 shows a timing diagram of 32-bit size input data and the CRC-8 calculations (outputs).

Signals in Figure 237 show that the 32-bit size input data enter at once and the CRC-8 calculated values are output from the point of single AHB clock cycle.

Figure 237. Timing Diagram of CRC Internal Signals: 32-bit Size Input and CRC-8 Outputs

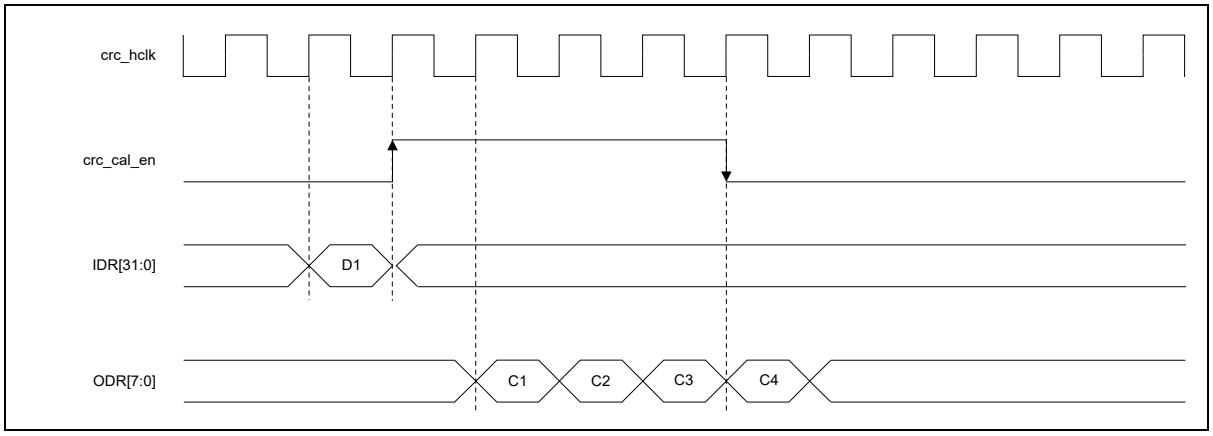


Figure 238 shows a timing diagram of 32-bit size input data and CRC-16 calculation (outputs).

Signals in Figure 238 show that the 32-bit size input data enter at once and two CRC-16 values are output. The first CRC-16 calculated value is output at the point of two AHB clock cycle and the second CRC-16 calculated in parallel is output at the final point of 2 + 2 AHB clock cycles.

Figure 238. Timing Diagram of CRC Internal Signals: 32-bit Size Input and CRC-16 Outputs

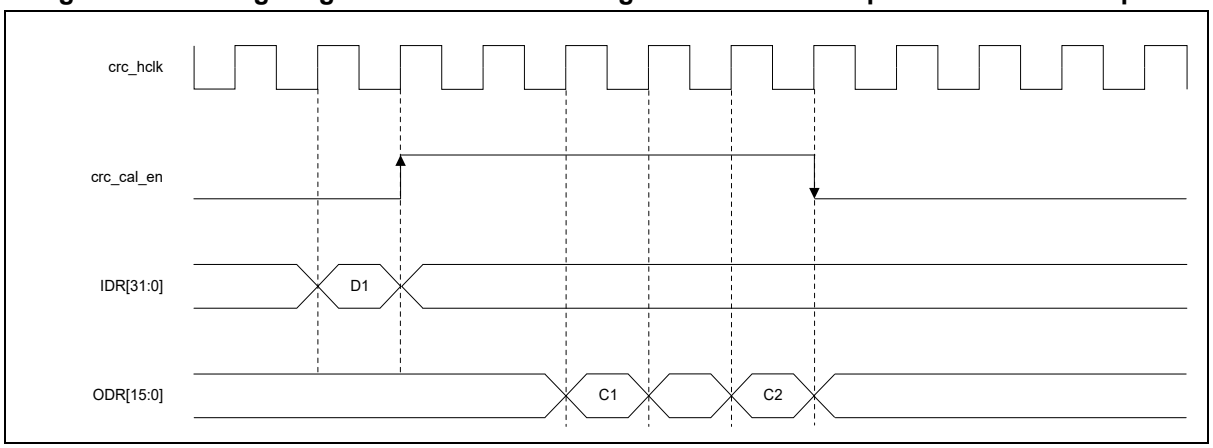
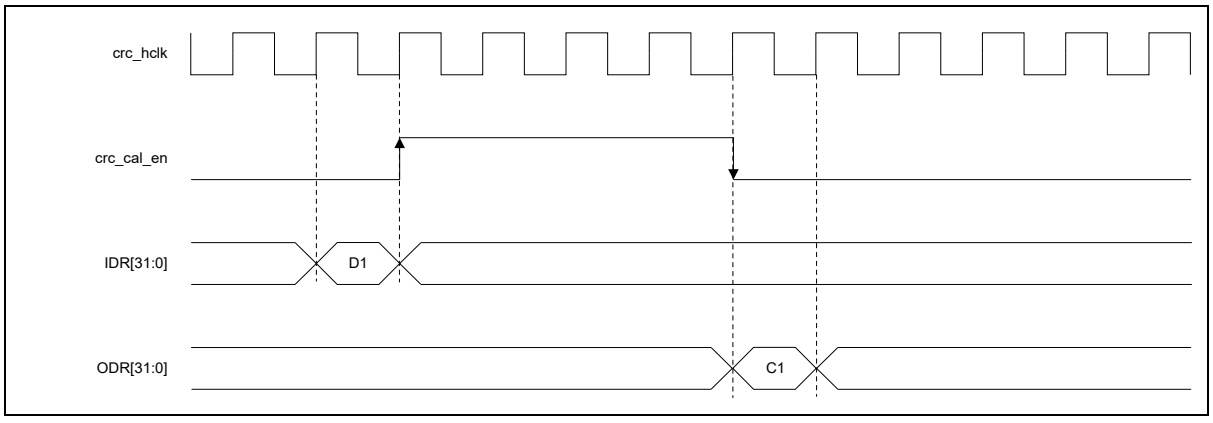


Figure 239 shows a timing diagram of 32-bit size input data and CRC-32 calculation (outputs).

Signals in Figure 239 show that the 32-bit size input data enter at once and the CRC-32 calculated values are output from the point of four AHB clock cycles.

Figure 239. Timing Diagram of CRC Internal Signals: 32-bit Size Input and CRC-32 Outputs



23.3.4 CRC Operation

The CRC calculated value can be obtained with 32-bit, 16-bit, 8-bit polynomial.

The CRC module must be enabled before working, so the CRC bit in the SCU_PER2 and SCU_PCER2 register must be set to ‘1’.

New input data is written to the CRC_IDR register (Input data register) and the previous CRC calculated value is read from the CRC_ODR register (Output data register).

The CRC_IDR register is available for 8-bit and 32-bit size input data. For the 16-bit CRC, the CRC module receives a 32-bit input data and internally processes the CRC calculation of 16-bit data two times.

23.3.4.1 How to Obtain CRC Calculation Result

This section describes how to obtain the CRC calculated value as shown below:

1. Set each bit of the CRC_CTRL register (CRC control register) as needed.
2. Update the CRC_IDR register (Input data register) to have a new CRC calculated value in combination with the previous CRC values.
3. Read the input data from the CRC_IDR register in the order below.
    A. For the 32-bit data size, for example, 0x1122_3344 is read from the least significant bit and calculated in the order of 0x44, 0x33, 0x22, and 0x11.
    B. For the 8-bit data size, users can input 0x44, 0x33, 0x22, and 0x11 data by using byte pointer for the CRC_IDR register. These written data is read and calculated in the order of 0x44, 0x33, 0x22, and 0x11.
4. The CRC calculation rate depends on the data width shown below.
    A. Four AHB clock cycles for 32-bit
    B. Two AHB clock cycles for 16-bit
    C. One AHB clock cycles for 8-bit

The CRC initial value can be programmed in the CRC_INIT register (CRC initial data register), and its reset initial value is defined as 0xFFFF_FFFF. Before setting this register, the INIT_EN bit in the CRC_CTRL register must be enabled.

The CRC module can reverse input data or output data and control the output data inversion. The OUT_REV or IN_REV bits in the CRC_CTRL register is converted using bit reversal. For example, 0x1122_3344’ is converted to ‘0x22CC_4488’.

The OUT_INV bit in the CRC_CTRL register can enable the inversion of CRC output data. For example, if the CRC output data is ‘0xAA55_FF00’, it is converted to ‘0x55AA_00FF’.

If the OUT_INV bit is set to ‘1’, each byte of CRC output data is XORed with ‘0xFF’ before it is output; while if the OUT_INV bit is set to ‘0’, each byte of CRC output data is XORed with ‘0x00’ before it is output.

The CRC module uses the following polynomial coefficients:

- CRC-32 (0x04C1_1DB7) when POLY[1:0] bits is set to ‘0’.
- CRC-16 USB (0x8005) when POLY[1:0] bits is set to ‘1’.
- CRC-16 CCITT (0x1021) when POLY[1:0] bits is set to ‘2’.
- CRC-8 CCITT (0x07) when POLY[1:0] bits is set to ‘3’.

23.3.5 CRC Using DMA

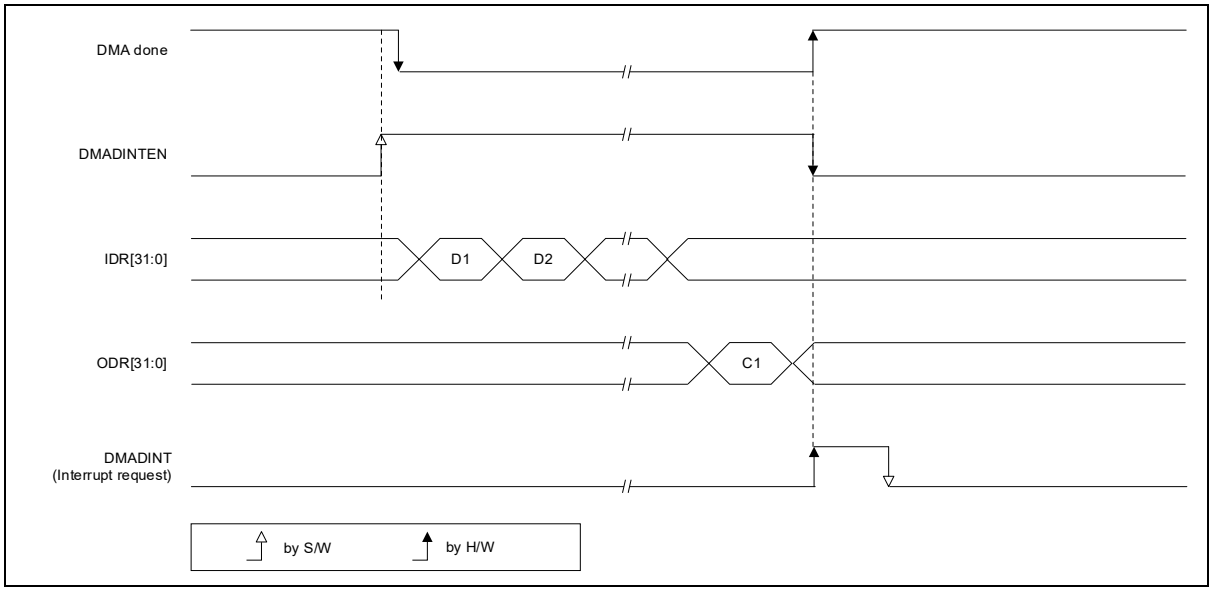
The CRC module supports DMA function.

Following procedure allows users to learn how to obtain the CRC calculated values using the DMA function. The DMA function is used with the CRC_CTRL register set to default.

1. Write CRC input data into memory.
2. Configure each bit of the DMAn_CR register by referring the followings:
   A. The TRANSCNT[11:0] bits determine the number of DMA transfers.
   B. The PERISEL[5:0] bits select the CRC_Tx (24).
   C. The SIZE[1:0] bits select the size of a data transfer among byte and word.
   D. The DIR bit sets the direction from memory to peripheral as TX.
3. Set the DMAn_PAR register to 0x4100_2008, the address of the CRC_IDR register.
4. To the DMAn_MAR register, write the initial address of the memory to be read.
5. Once the values are set in the previous steps, enable the DMAEN bit in the DMAn_SR register to transfer the CRC value using DMA.
6. Once the DMA transfer is started and when value of the TRANSCNT[11:0] bits in the DMAn_CR register reaches '0', the EOT flag in the DMAn_SR register is enabled indicating that the DMA transfer ends.
7. For detailed information on the DMA operation, refer to chapter 7.

Figure 240 shows a timing diagram of the CRC operation using DMA.

Figure 240. Timing Diagram of CRC Operation Using DMA



23.4 CRC Registers

The base address and register map of the CRC module are described in the following tables.

Table 217. Base Address of CRC

Name	Base Address
CRC	0x4100_2000

Table 218. CRC Register Map

Name	Offset	Type	Description	Reset Value	Reference.
CRC_CTRL	0x0000	RW	CRC Control Register	0x0000_0000	23.4.1
CRC_INIT	0x0004	RW	CRC Initial Data Register	0xFFFF_FFFF	23.4.2
CRC_IDR	0x0008	WO	CRC Input Data Register	-	23.4.3
CRC_ODR	0x0008	RO	CRC Output Data Register	0xFFFF_FFFF	23.4.4
CRC_STAT	0x000C	RWC1	CRC Status Register	0x0000_0000	23.4.5

23.4.1 CRC_CTRL: CRC Control Register

The CRC_CTRL register controls the CRC calculation module.

CRC_CTRL=0x4100_2000

31 30 29 28 27 26 25 24	23 22	21	20	19 18 17	16	15 14 13 12 11 10 9	8	7 6 5 4 3	2 1	0
Reserved		OUT_INV	OUT_REV	Reserved	IN_REV	Reserved	DMADINTEN	Reserved	POLY[1:0]	INIT_EN
-		0	0	-	0	-	0	-	00	0
-		RW	RW	-	RW	-	RW	-	RW	WO

21	OUT_INV	Enable CRC output data inversion	
		0	Normal
		1	Inverted
20	OUT_REV	Reverse CRC output data	
		0	Not affected (LSB first)
		1	Bit-reversed output format (MSB first)
16	IN_REV	Reverse CRC input data	
		0	Not affected (LSB first)
		1	Reverse the input data. (MSB first)
8	DMADINTEN	Enable the DMA done interrupt	
		0	Disable
		1	Enable
2 1	POLY[1:0]	Polynomial selection bit	
		00	CRC-32 (0x04C1_1DB7)
		01	CRC-16-USB (0x8005)
		10	CRC-16-CCITT (0x1021)
		11	CRC-8-CCITT (0x07)
0	INIT_EN	Whether to apply the CRC initial value register (auto clear)	
		0	No Effect
		1	Applies the CRC_INIT register value.

23.4.2 CRC_INIT: CRC Initial Data Register

The CRC initial value data is written to the CRC_INIT register.

CRC_INIT=0x4100_2004

31 30 29 28 27 26 25 24	23 22 21 20 19 18 17 16	15 14 13 12 11 10 9 8	7 6 5 4 3 2 1 0
INIT[31:0]			
0xFFFF_FFFF			
RW			

31 0	INIT[31:0](1)	CRC initial value

NOTE:

1. To write INIT[31:0] data to the internal CRC initial register, the INIT_EN bit in the CRC_CTRL register must be enabled. For example, writing '0x8005' to CRC_INIT register and writing a '1' to the INIT_EN bit in the CRC_CTRL register changes the internal CRC initial register to '0x8005'.

23.4.3 CRC_IDR: CRC Input Data Register

The CRC_IDR register can be accessed by 32-, 16- and 8-bit. This register is 32 bits wide.

CRC_IDR=0x4100_2008

31 30 29 28 27 26 25 24	23 22 21 20 19 18 17 16	15 14 13 12 11 10 9 8	7 6 5 4 3 2 1 0
IDR[31:0]			
-			
WO			

31 0	IDR[31:0]	CRC input data bits Once a datum is written in this bit field, its polynomial result is automatically output to the CRC_ODR register.

23.4.4 CRC_ODR: CRC Output Data Register

The CRC_ODR register can be accessed by 32-, 16- and 8-bit. This register is 32 bits wide.

CRC_ODR=0x4100_2008

31 30 29 28 27 26 25 24	23 22 21 20 19 18 17 16	15 14 13 12 11 10 9 8	7 6 5 4 3 2 1 0
ODR[31:0]			
0xFFFF_FFFF			
RO			

31 0	ODR[31:0]	CRC output data bits

23.4.5 CRC_STAT: CRC Status Register

The CRC_STAT register displays the operating status of the CRC.

CRC_STAT = 0x4100_200C

31 30 29 28 27 26 25 24	23 22 21 20 19 18 17 16	15 14 13 12 11 10 9	8	7 6 5 4 3 2 1 0
Reserved			DMADINT	Reserved
-			0	-
-			RWC1	-

8	DMADINT	DMA Done Interrupt Flag Bit	
		0	DMA transfer is not done.
		1	DMA transfer is done. (Writing to the bit clears the flag.)

23.4.6 CRC Register Map Summary

Table 219. CRC Register Map Summary

Offset	Register name	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0x00	CRC_CTRL	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	OUT_INV	OUT_REV	Res.	Res.	Res.	IN_REV	Res.	Res.	Res.	Res.	Res.	Res.	Res.	DMADINTEN	Res.	Res.	Res.	Res.	Res.	POLY[1:0]		INIT_EN
	Reset value											0	0				0								0						0	0	0
0x04	CRC_INIT	INIT[31:0]																															
	Reset value	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1
0x08	CRC_IDR	IDR[31:0]																															
	Reset value	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1
0x0C	CRC_ODR	ODR[31:0]																															
	Reset value	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1
0x10	CRC_STAT	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	DMADINT	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.
	Reset value																								0								

Abbreviation for Registers

The following abbreviations are used in register descriptions:

Table 220. Abbreviation for Registers

Abbreviation	Description
RW (Read/Write)	Software can read and write to this bit.
RO (Read-Only)	Software can only read this bit.
WO (Write-Only)	Software can only write to this bit. Reading this bit returns the reset value.
RC (Read/Clear)	Software can read to clear this bit.
RWC0 (Read/write 0 clear)	Software can read and clear this bit by writing ‘0’.
RWC1 (Read/write 1 clear)	Software can read and clear this bit by writing ‘1’.
Res. (Reserved)	Reserved bit, must be kept at reset value.

Glossary

This section gives a brief definition of acronyms, abbreviations, and terminology used in this document:

- Word: Data of 32-bit length
- Half-word: Data of 16-bit length
- Byte: Data of 8-bit length
- PGM: Flash Programming
- ERS: Flash Erasing
- AHB: Advanced High-performance Bus
- APB: Advanced Peripheral Bus
- CRC: Cyclic Redundancy Check
- CANFD: Controller Area Network Flexible Data-Rate
- DMA: Direct Memory Access
- FRT: Free-Run Timer
- HSE: High-Speed External
- HSI: High-Speed Internal
- I2C: Inter-Integrated Circuit
- LSB: Least Significant Bit
- LQFP: Low-profile Quad Flat Package
- LSE: Low-Speed External
- LSI: Low-Speed Internal
- LVI: Low-Voltage Indicator
- LVR: Low-Voltage Reset
- MPWM: Motor Pulse Width Modulation
- MSB: Most Significant Bit
- OPAMP: Operational Amplifier
- PGM: Programming
- PLL: Phase Locked Loop
- POR: Power On Reset
- QSPI: Quad Serial Peripheral Interface
- SCU: System Control Unit

- SPI: Serial Peripheral Interface
- UART: Universal Asynchronous Receiver Transmitter
- WDT: Watchdog Timer

Revision History

Revision	Date	Notes
1.00	Dec. 02, 2025	Initial release.

Korea

Regional Office, Seoul
R&D, Marketing & Sales
8th Fl., 330, Yeongdong-daero, Gangnam-gu, Seoul,
06177, Korea

Tel: +82-2-2193-2200
Fax: +82-2-508-6903
www.abovsemi.com

HQ, Ochang
R&D, QA, and Test Center
37, Gangni 1-gil, Ochang-eup, Cheongwon-gun,
Chungcheongbuk-do, 28126, Korea

Tel: +82-43-219-5200
Fax: +82-43-217-3534
www.abovsemi.com

Domestic Sales
Tel: +82-2-2193-2206
Fax: +82-2-508-6903
Email: sales_kr@abov.co.kr

Global Sales
Tel: +82-2-2193-2281
Fax: +82-2-508-6903
Email: sales_gl@abov.co.kr

China Sales
Tel: +86-755-8287-2205
Fax: +86-755-8287-2204
Email: sales_cn@abov.co.kr

Japan Sales
Email: sales_jp@abov.co.kr

ABOV Disclaimer

IMPORTANT NOTICE – PLEASE READ CAREFULLY

ABOV Semiconductor (“ABOV”) reserves the right to make changes, corrections, enhancements, modifications, and improvements to ABOV products and/or to this document at any time without notice. **ABOV DOES NOT GIVE WARRANTIES AS TO THE ACCURACY OR COMPLETENESS OF THE INFORMATION INCLUDED HEREIN.** Purchasers should obtain the latest relevant information on ABOV products before placing orders. Purchasers are entirely responsible for the choice, selection, and use of ABOV products and ABOV assumes no liability for application assistance or the design of purchasers’ products. **NO LICENSE, EXPRESS OR IMPLIED, TO ANY INTELLECTUAL PROPERTY RIGHTS IS GRANTED BY ABOV HEREIN. ABOV DISCLAIMS ALL EXPRESS AND IMPLIED WARRANTIES AND SHALL NOT BE RESPONSIBLE OR LIABLE FOR ANY INJURIES OR DAMAGES RELATED TO USE OF ABOV PRODUCTS IN SUCH UNAUTHORIZED APPLICATIONS.** ABOV and the ABOV logo are trademarks of ABOV. For additional information about ABOV trademarks, please refer to https://www.abov.co.kr/en/about/corporate_identity.php. All other products or service names are the property of their respective owners. Information in this document supersedes and replaces the information previously supplied in any former versions of this document.

© 2025 ABOV Semiconductor – All rights reserved