Low-Power Cortex®-M4F 32-bit Microcontroller with FPU, 32 MHz, 256 KB Code Flash, 32 KB Data Flash, and 24 KB SRAM

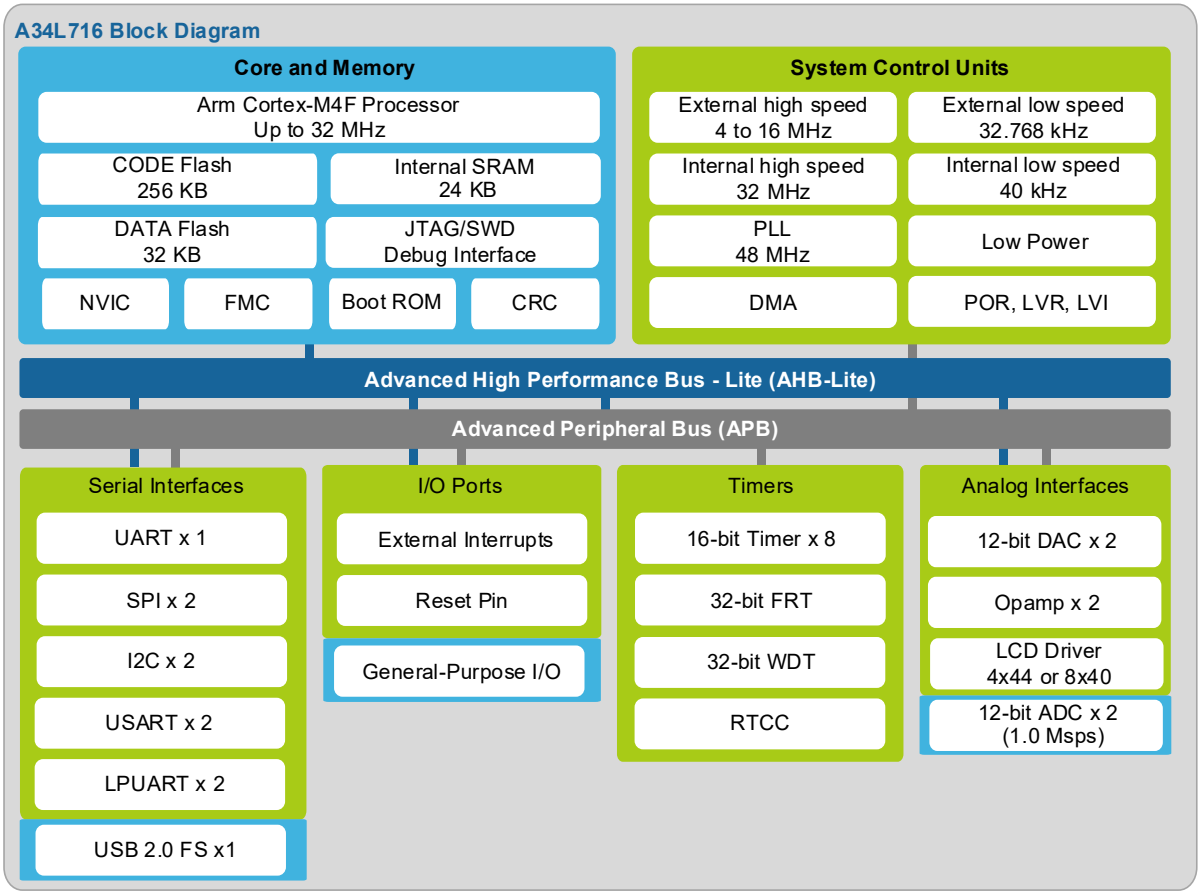
UM Rev. 1.04

Introduction

This product is a 32-bit microcontroller suitable for applications that require high-performance processing. The microcontroller includes the Arm Cortex-M4F, which is a high-performance 32-bit core, and a variety of peripheral devices.

The microcontroller has two high-speed 12-bit Analog-to-Digital Converters (ADCs) with 24 analog channels, two Digital-to-Analog Converters (DAC) units, and two OPAMP units. The microcontroller can also communicate with external devices through various interfaces, including UART, SPI, I2C, USART, LPUART, and USB modules.

Figure 1. A34L716 Conceptual Block Diagram



Reference Document

- Document ‘ID061113’ is provided by Arm and contains information on Cortex-M4F.
- A34L716 Datasheet is provided by ABOV and available at www.abovsemi.com.

Contents

Introduction 1
Reference Document 2
1. Description 27
1.1 Product Category Definition 27
1.2 Device Overview 28
2. Pinouts and Pin Descriptions 36
2.1 Pinouts 36
2.1.1 A34L716VLN (100-LQFP) 36
2.1.2 A34L716RLN (64-LQFP) 37
2.1.3 A34L716JYN (56-QFN) 38
2.2 Pin Description 39
3. System and Memory Overview 49
3.1 System Architecture 49
3.1.1 Cortex-M4F Core 50
3.1.2 Floating Point Unit (FPU) 50
3.1.3 Interrupt Controller 51
3.2 Memory Organization 55
3.2.1 Introduction 55
3.2.2 Memory Map and Register Boundary Address 56
3.3 Memory Map 58
3.3.1 Embedded SRAM 58
3.3.2 Flash Memory Overview 58
3.4 Boot Configuration 59
4. SCU (System Control Unit) 60
4.1 SCU Introduction 60
4.2 SCU Main Features 60
4.3 SCU Functional Description 61
4.3.1 SCU Block Diagram 61
4.3.2 SCU Pin and Type 61
4.4 Clocks 62
4.4.1 Overview 62
4.4.2 Clock Tree Configuration 64
4.4.3 HSE Clock 65
4.4.4 HSI Clock 65
4.4.5 PLL Clock 66
4.4.5.1 PLL Output Frequency Settings 67
4.4.5.2 Example PLL Clock Change 67
4.4.6 LSE Clock 68
4.4.6.1 PCB Design Considerations 69
4.4.7 LSI Clock 70
4.4.8 Two Main Operating Clocks: HCLK and PCLK 70
4.4.8.1 HCLK Clock 70
4.4.8.2 PCLK Clock 70
4.4.9 Configuration of Miscellaneous Clocks 71
4.4.10 Peripheral Clocks 73
4.4.11 Clock Monitoring 74
4.4.11.1 Detection Range 74

4.4.11.2 Clock Monitoring Functions and Clock Sources 75
4.4.11.3 MCLK Monitoring 76
4.4.11.4 HSE Monitoring 78
4.4.11.5 LSE Monitoring 79
4.4.12 Clock Configuration Procedure 80
4.5 Reset 83
4.5.1 Overview 83
4.5.2 Reset Tree Configuration 84
4.5.3 Cold Reset 85
4.5.4 Warm Reset 86
4.5.5 LVD Reset (LVR) 87
4.5.6 nRESET Pin Reset 87
4.5.7 Watchdog Timer Reset 88
4.5.8 Software Reset 88
4.5.9 HSE Fail Reset 88
4.5.10 LSE Fail Reset 88
4.5.11 CPU Request Reset 89
4.6 Power Supply 90
4.6.1 LVI Block Diagram 91
4.6.2 LVR Block Diagram 92
4.6.3 POR, LVR 92
4.7 Operation Modes 93
4.7.1 Overview 93
4.7.2 Transition of Operation Mode 94
4.7.3 RUN Mode 95
4.7.4 SLEEP Mode 95
4.7.5 DEEP-SLEEP (STOP) Mode 96
4.7.6 Operation Mode Summary 98
4.8 SCU Registers 99
4.8.1 CHIPCONFIG_VENDORID: Vendor ID Register 101
4.8.2 CHIPCONFIG_CHIPID: Chip ID Register 101
4.8.3 CHIPCONFIG_REVNR: Revision Number Register 102
4.8.4 SCU_SMR: System Mode Register 103
4.8.5 SCU_SRCR: System Reset Control Register 104
4.8.6 SCU_WUER: Wake-Up Source Enable Register 105
4.8.7 SCU_WUSR: Wake-Up Source Status Register 107
4.8.8 SCU_RSER: Reset Source Enable Register 109
4.8.9 SCU_RSSR: Reset Source Status Register 110
4.8.10 SCU_PRER1: Peripheral Reset Enable Register 1 112
4.8.11 SCU_PRER2: Peripheral Reset Enable Register 2 114
4.8.12 SCU_PER1: Peripheral Enable Register 1 116
4.8.13 SCU_PER2: Peripheral Enable Register 2 118
4.8.14 SCU_PCER1: Peripheral Clock Enable Register 1 120
4.8.15 SCU_PCER2: Peripheral Clock Enable Register 2 122
4.8.16 SCU_CSCR: Clock Source Control Register 124
4.8.17 SCU_SCCR: System Clock Control Register 125
4.8.18 SCU_CMR: Clock Monitoring Register 127
4.8.19 SCU_COR: Clock Output Register 129
4.8.20 SCU_NMICR: NMI Control Register 130
4.8.21 SCU_NMISR: NMI Status Register 131

4.8.22 SCU_PLLCON: PLL Control Register 132
4.8.23 SCU_VDCCON: VDC Control Register 133
4.8.24 SCU_LVICR: Low-Voltage Indicator (LVI) Control Register 134
4.8.25 SCU_LVISR: Low-Voltage Indicator (LVI) Status Register 135
4.8.26 SCU_LVRCR: Low-Voltage Detect Reset (LVR) Control Register 136
4.8.27 SCU_EOSCR: External Oscillator Control Register 137
4.8.28 SCU_MCCR1: Miscellaneous Clock Control Register 1 139
4.8.29 SCU_MCCR2: Miscellaneous Clock Control Register 2 140
4.8.30 SCU_MCCR3: Miscellaneous Clock Control Register 3 141
4.8.31 SCU_MCCR4: Miscellaneous Clock Control Register 4 142
4.8.32 SCU_MCCR5: Miscellaneous Clock Control Register 5 143
4.8.33 SCU_MCCR6: Miscellaneous Clock Control Register 6 144
4.8.34 SCU_MCCR7: Miscellaneous Clock Control Register 7 145
4.8.35 SCU_SYSTEN: System Access Enable Register 146
4.8.36 SCU Register Map Summary 147
5. PCU (Port Control Unit) and GPIO (General-Purpose I/O) 151
5.1 Introduction 151
5.1.1 PCU Introduction 151
5.1.2 GPIO Introduction 151
5.2 Main Features 152
5.2.1 PCU Features 152
5.2.2 GPIO Features 152
5.3 GPIO Functional Description 153
5.3.1 GPIO Pins and Internal Signals 157
5.3.2 I/O Port Control Registers 157
5.3.3 I/O Port Data Registers 158
5.3.4 I/O Data Bitwise Handling 158
5.3.5 I/O Alternate Function 159
5.3.6 Interrupt Functionality 163
5.3.7 Output Configuration 164
5.3.8 Input Configuration 165
5.3.9 Alternate Function Configuration 166
5.3.10 Debouncing Functionality 167
5.4 GPIO Registers 168
5.4.1 Pn_MR1: Port n MUX Control Register 1 169
5.4.2 Pn_MR2: Port n MUX Control Register 2 170
5.4.3 Pn_CR: Port n Control Register 171
5.4.4 Pn_PRCR: Port n Pull-Up/Pull-Down Control Register 171
5.4.5 Pn_DER: Port n Debounce Enable Register 172
5.4.6 Pn_STR: Port n Strength Configuration Register 172
5.4.7 Pn_IER: Port n Interrupt Enable Register 173
5.4.8 Pn_ISR: Port n Interrupt Status Register 174
5.4.9 Pn_ICR: Port n Interrupt Control Register 175
5.4.10 Pn_ODR: Port n Output Data Register 175
5.4.11 Pn_IDR: Port n Input Data Register 176
5.4.12 Pn_BSR: Port n Set/Reset Register 177
5.4.13 Pn_BCR: Port n Reset Register 177
5.4.14 PCU_PORTEN: Port Access Enable Register 178
5.4.15 PCU and GPIO Register Map Summary 179
6. Embedded Flash Memory 181

6.1 Introduction 181
6.2 Flash Main Features 181
6.3 Flash Functional Description 182
6.3.1 Flash Memory Organization 182
6.3.1.1 Code Flash Memory 183
6.3.1.2 Data Flash Memory 184
6.3.2 Read Access Latency 185
6.3.2.1 Code Flash Memory Wait Time 185
6.3.2.2 Data Flash Access Time 186
6.3.3 Flash Program and Erase Operations 187
6.3.3.1 Access Control for Writing and Erasing Flash Memory 187
6.3.3.2 Access Control for Writing and Erasing User Info Memory 188
6.3.4 Flash Memory Erase Sequences 189
6.3.4.1 Page Erase 189
6.3.4.2 2 KB Sector Erase 190
6.3.4.3 Bulk Erase 191
6.3.5 Flash Memory Programming Sequences 192
6.3.5.1 Standard Programming 192
6.4 Flash Memory Protection 193
6.4.1 Read Protection 193
6.4.1.1 Read Protection: UNPROT (Unprotection) 194
6.4.1.2 Read Protection: LVL1 (Level-1) 194
6.4.1.3 Read Protection: LVL2 (Level-2) 195
6.4.1.4 Read Protection Setting 195
6.4.1.5 Read Protection Removal 195
6.4.1.6 Password Settings 196
6.4.2 Write Protection 197
6.4.2.1 Write Protection for Code Flash Memory 197
6.4.2.2 Write Protection for Data Flash 198
6.4.3 Securable Memory Area (LVL1, LVL2) 199
6.4.4 Disabling Core Debug Access (LVL2) 199
6.5 Flash Interrupts 200
6.6 Code Flash Registers 201
6.6.1 CFMC_CONF: Code Flash Control Register 202
6.6.2 CFMC_FLSKEY: Code Flash Access Key Register 203
6.6.3 CFMC_INFOKEY: Code Flash User Info Access Key Register 204
6.6.4 CFMC_FLSPROT: Code Flash Protection Register 205
6.6.5 CFMC_INFOPROT: Code Flash Info Protection Register 207
6.6.6 CFMC_CTRL: Code Flash Access Control Register 208
6.6.7 CFMC_STAT: Code Flash Access Status Register 210
6.6.8 CFMC_READPROT: Code Flash Read Protection Register 211
6.6.9 CFMC_PWIN: Code Flash Password Input Register 212
6.6.10 CFMC_WTIMEOUT: Code Flash Write Time-Out Register 213
6.6.11 CFMC_ECCEADDR: Code Flash ECC Error Address Register 213
6.6.12 CFMC_ECCEDATA0: Code Flash ECC Error Data0 Address Register 214
6.6.13 CFMC_ECCEDATA1: Code Flash ECC Error Data1 Address Register 214
6.6.14 CFMC_ECCEDATA2: Code Flash ECC Error Data2 Address Register 215
6.6.15 CFMC_ECCEDATA3: Code Flash ECC Error Data3 Address Register 215
6.6.16 CFMC_ECCEPARITY: Code Flash ECC Error Parity Register 216
6.6.17 CFMC_PWPRST: Code Flash Password Preset Register 216

6.6.18 CFMC Register Map Summary 217
6.7 Data Flash Register 219
6.7.1 DFMC_CONF: Data Flash Control Register 220
6.7.2 DFMC_FLSKEY: Data Flash Access Key Register 220
6.7.3 DFMC_FLSPROT: Data Flash Protection Register 221
6.7.4 DFMC_CTRL: Data Flash Access Control Register 222
6.7.5 DFMC_STAT: Data Flash Access Status Register 223
6.7.6 DFMC_WTIMEOUT: Data Flash Write Time-Out Register 224
6.7.7 DFMC_ECCEADDR: Data Flash ECC Error Address Register 224
6.7.8 DFMC_ECCEDATA: Data Flash ECC Error Data Register 225
6.7.9 DFMC_ECCEPARITY: Data Flash ECC Error Parity Register 225
6.7.10 DFMC Register Map Summary 226
7. HSICMU (High-Speed Internal Oscillator Clock Management Unit) 227
7.1 Introduction 227
7.2 HSICMU Main Features 227
7.3 HSICMU Functional Description 228
7.3.1 HSICMU Block Diagram 228
7.3.2 Synchronization Input 228
7.3.3 Frequency Error Measurement 229
7.3.4 Frequency Error Evaluation and Automatic Trimming 230
7.3.5 HSICMU Initialization and Configuration 231
7.4 HSICMU Registers 235
7.4.1 HSICMU_CR: HSICMU Control Register 236
7.4.2 HSICMU_CFG: HSICMU Configure Register 237
7.4.3 HSICMU_ISR: HSICMU Interrupt and Status Register 238
7.4.4 HSICMU_ICR: HSICMU Control Interrupt and Clear Status Register 239
7.4.5 HSICMU_TRIM: HSICMU Control TRIM Value Register 240
7.4.6 HSICMU_ATRIM: HSICMU Control Auto-Trim Value Register 240
7.4.7 HSICMU_CAP: HSICMU Capture Value Register 241
7.4.8 HSICMU_THR: HSICMU Frequency Error Rate Threshold Register 241
7.4.9 HSICMU Register Map Summary 242
8. DMA (Direct Memory Access Controller) 243
8.1 Introduction 243
8.1.1 Modules Supported by DMA 243
8.2 DMA Functional Description 244
8.2.1 DMA Transfer Timing from Peripheral to Memory 245
8.2.2 DMA Transfer Timing from Memory to Peripheral 246
8.2.3 DMA Transfer 246
8.3 DMA Registers 247
8.3.1 DMAn_CR: DMA Channel n Control Register 248
8.3.2 DMAn_SR: DMA Channel n Status Register 249
8.3.3 DMAn_PAR: DMA Channel n Peripheral Address Register 250
8.3.4 DMAn_MAR: DMA Channel n Memory Address Register 250
8.3.5 DMA Register Map Summary 251
9. WDT (Watchdog Timer) 252
9.1 WDT Introduction 252
9.2 WDT Main Features 252
9.3 WDT Functional Description 253
9.3.1 WDT Block Diagram 253
9.3.2 Enabling WDT 253

9.3.3 Controlling Down-Counter 254
9.3.4 Interrupt Features 255
9.3.4.1 Interrupt Mode Operation Timing Diagram 255
9.3.5 How to Program Watchdog Timeout 256
9.3.5.1 Basic Reset Operation 256
9.3.6 Debug Mode 256
9.4 WDT Registers 257
9.4.1 WDT_LR: WDT Load Register 258
9.4.2 WDT_CNT: WDT Current Count Register 259
9.4.3 WDT_CON: WDT Control Register 260
9.4.4 WDT_AEN: WDT Access Enable Register 261
9.4.5 WDT Register Map Summary 262
10. 16-bit Timer 263
10.1 16-bit Timer Introduction 263
10.2 16-bit Timer Main Features 264
10.3 16-bit Timer Functional Description 265
10.3.1 Block Diagram 265
10.3.2 Pins and Internal Signals 266
10.3.3 Clock Selection 268
10.3.3.1 Peripheral System Clock (PCLK) 268
10.3.3.2 Timer External Clock (SCU_MCCR3 Register) 268
10.3.3.3 Input to Pin TnIO for Timer Clock Source 269
10.3.4 Time-Base Unit 270
10.3.4.1 Prescaler Description 271
10.3.5 Periodic Mode 272
10.3.5.1 Operation According to TIMERn_GRA and TIMERn_GRB Settings 273
10.3.5.2 Operation According to UAO in TIMERn_CR1, OUTPOL in TIMERn_CR1, and STARTLVL in TIMERn_CR1 Settings 274
10.3.5.3 Additional Precautions Over Counter Mode Operation 276
10.3.5.4 Counter Mode Register Setting Sequence 276
10.3.6 Capture Mode 277
10.3.6.1 Capture Mode Register Setting Sequence 278
10.3.7 PWM Mode 279
10.3.7.1 PWM Mode Operation Characteristics and Precautions 280
10.3.7.2 PWM Mode Register Setting Sequence 280
10.3.8 One-Shot Mode 281
10.3.8.1 One-Shot Mode Operation Characteristics and Precautions 282
10.3.8.2 One-Shot Mode Register Setting Sequence 283
10.3.9 ADC Trigger 284
10.3.9.1 ADC Trigger Operation Characteristics and Precautions 285
10.3.9.2 Register Setting Sequence for ADC Trigger Operation 286
10.3.10 Timer Synchronization 287
10.3.10.1 Delayed Timer Synchronization 289
10.3.10.2 Timer Synchronization Operation Characteristics and Precautions 290
10.3.10.3 Register Setting Sequence for Synchronous Function Operation 291
10.3.11 Direction Bit Output 292
10.3.12 Debug Mode 292
10.3.13 Interrupts 292
10.4 Timer Registers 293

10.4.1 TIMERn_CR1: Timer/Counter n Control Register 1 294
10.4.2 TIMERn_CR2: Timer/Counter n Control Register 2 296
10.4.3 TIMERn_PRS: Timer/Counter n Prescaler Register 296
10.4.4 TIMERn_GRA: Timer/Counter n General Data Register A 297
10.4.5 TIMERn_GRB: Timer/Counter n General Data Register B 298
10.4.6 TIMERn_CNT: Timer/Counter n Count Register 299
10.4.7 TIMERn_SR: Timer/Counter n Status Register 299
10.4.8 TIMERn_IER: Timer/Counter n Interrupt Enable Register 300
10.4.9 TIMERn_TRGPNT: Timer/Counter n Trigger Point Register 300
10.4.10 TIMERn_SYNC: Timer/Counter n Sync Configuration Register 301
10.4.11 Timer Register Map Summary 302
11. Free-Run Timer (FRT) 303
11.1 FRT Introduction 303
11.2 FRT Main Features 303
11.3 FRT Functional Description 304
11.3.1 FRT Block Diagram 304
11.3.2 FRT Clock Selection 304
11.3.3 FRT Operation Mode 305
11.3.3.1 Match Interrupt Operation 305
11.3.3.2 Overflow Interrupt Operation 306
11.3.3.3 Register Setting Procedure by FRT Operation Mode 307
11.3.4 FRT Low-Power Modes 308
11.3.4.1 Description of Steps of Low-Power Mode 308
11.4 FRT Registers 309
11.4.1 FRT_CTRL: FRT Control Register 310
11.4.2 FRT_MCNT: FRT Match Counter Register 310
11.4.3 FRT_CNT: FRT Counter Register 311
11.4.4 FRT_STAT: FRT Status Register 311
11.4.5 FRT Register Map Summary 312
12. UART (Universal Asynchronous Receiver/Transmitter) 313
12.1 UART Introduction 313
12.2 Main Features 314
12.3 UART Functional Description 315
12.3.1 UART Block Diagram 316
12.3.2 UART Pins and Signals 317
12.3.2.1 UART Bidirectional Communications 317
12.3.3 UART Character Description 318
12.3.4 UART Transmitter 321
12.3.4.1 Data Format 321
12.3.4.2 Transmit Interrupt 322
12.3.4.3 Character Transmission 322
12.3.4.4 Configurable Stop Bits 323
12.3.4.5 Character Transmission Procedure 324
12.3.4.6 Single Byte Communication 325
12.3.4.7 Break Condition 325
12.3.4.8 Inter-Frame Delay for Data Transmission 325
12.3.5 UART Receiver 326
12.3.5.1 Receive Data Sampling Timing 326
12.3.5.2 Start Bit Detection 327
12.3.5.3 Data Bit Sampling 329

12.3.5.4 Data Sampling Strategy 330
12.3.5.5 Character Reception 331
12.3.5.6 Break Condition 331
12.3.5.7 Overrun Error 331
12.3.5.8 Framing Error 332
12.3.5.9 Parity Error 332
12.3.5.10 Configurable Stop Bits during Reception 332
12.3.6 UART Baud Rate Generation 333
12.3.7 Tolerance of UART Receiver to Clock Deviation 335
12.3.8 UART Parity Control 336
12.3.8.1 Even Parity 336
12.3.8.2 Odd Parity 336
12.3.8.3 Stick Parity 337
12.3.8.4 Various Configurations of Parity Bit 337
12.3.8.5 Parity Checking in Reception 337
12.3.8.6 Parity Generation in Transmission 337
12.3.9 Continuous Communication Using DMA and UART 338
12.3.9.1 Transmission Using DMA 339
12.3.9.2 Reception Using DMA 340
12.3.9.3 Error Flagging and Interrupt Generation in Communication 341
12.4 UART Interrupts 342
12.5 UART Registers 343
12.5.1 UARTn_RBR: UART n Receive Data Buffer Register 344
12.5.2 UARTn_THR: UART n Transmit Data Hold Register 344
12.5.3 UARTn_IER: UART n Interrupt Enable Register 345
12.5.4 UARTn_IIR: UART n Interrupt ID Register 346
12.5.5 UARTn_LCR: UART n Line Control Register 347
12.5.6 UARTn_DCR: UART n Data Control Register 349
12.5.7 UARTn_LSR: UART n Line Status Register 350
12.5.8 UARTn_BDR: UART n Baud Rate Divisor Latch Register 352
12.5.9 UARTn_BFR: UART n Baud Rate Fraction Counter Register 352
12.5.10 UARTn_IDTR: UART n Inter-Frame Delay Time Register 353
12.5.11 UART Register Map Summary 354
13. SPI (Serial Peripheral Interface) 355
13.1 SPI Introduction 355
13.2 SPI Main Features 355
13.3 SPI Functional Description 356
13.3.1 SPI Block Diagram 357
13.3.2 SPI Pins and Internal Signals 358
13.3.3 Communication between Master and Single Slave 359
13.3.3.1 Full-Duplex Communication 359
13.3.3.2 Simplex Communication 359
13.3.4 Slave Selection Pin 360
13.3.5 Communication Formats 361
13.3.5.1 Clock Phase and Polarity Controls 361
13.3.5.2 Data Frame Format 364
13.3.6 Configuration of SPI 365
13.3.7 Procedure for Enabling SPI 365
13.3.8 Data Transmission and Reception Procedures 366
13.3.8.1 Sequence Handling 366

13.3.8.2 Procedure for Disabling SPI 369
13.3.8.3 Data Packing 369
13.3.8.4 DMA Handshake 370
13.3.8.5 Communication Diagrams with DMA 371
13.3.9 SPI Status Flags 373
13.3.9.1 Transmit Buffer Empty Flag (TRDY) 373
13.3.9.2 Transmit/Receive Operation Flag (SBUSY) 373
13.3.9.3 Receive Buffer Ready Flag (RRDY) 373
13.3.9.4 Slave Select Signal Status Flag (SSON) 373
13.3.9.5 Rising or Falling Edge of Slave Select Signal Detect Flag (SSDET) 373
13.3.9.6 DMA Transmit Operation Complete Flag. (DMA to SPI, TXDMAF) 373
13.3.9.7 DMA Receive Operation Complete Flag. (SPI to DMA, RXDMAF) 373
13.3.10 SPI Error Flags 374
13.3.10.1 Overrun Flag (OVRF) 374
13.3.10.2 Transmit Underrun Error Flag (UDRF) 374
13.4 SPI Registers 375
13.4.1 SPIn_TDR: SPI Transmit Data Register 376
13.4.2 SPIn_RDR: SPI Receive Data Register 376
13.4.3 SPIn_CR: SPI Control Register 377
13.4.4 SPIn_SR: SPI Status Register 379
13.4.5 SPIn_BR: SPI Baud Rate Register 380
13.4.6 SPIn_EN: SPI Enable Register 380
13.4.7 SPIn_LR: SPI Delay Length Register 381
13.4.8 SPI Registers Map Summary 382
14. I2C (Inter-Integrated Circuit) 383
14.1 I2C Introduction 383
14.2 I2C Main Features 383
14.3 I2C Implementation 384
14.4 I2C Functional Description 385
14.4.1 I2C Block Diagram 385
14.4.2 I2C Pins and Signals 386
14.4.3 I2C Mode Selection 386
14.4.4 I2C Initialization 387
14.4.4.1 Enabling or Disabling I2C 387
14.4.4.2 Noise Canceller 387
14.4.4.3 Using Noise Canceller in I2C Module 388
14.4.5 I2C Software Reset 389
14.4.6 I2C Module Protocol 390
14.4.6.1 I2C Bit Transfer 390
14.4.6.2 START/Repeated START/STOP 391
14.4.6.3 Acknowledge 392
14.4.6.4 Synchronization 393
14.4.6.5 Arbitration 394
14.4.6.6 Data Transfer 395
14.4.6.7 Master Transmit Timing 396
14.4.6.8 Master Receive Timing 397
14.4.6.9 Slave Transmit Timing 398
14.4.6.10 Slave Receive Timing 399
14.4.7 I2C Slave Mode 400
14.4.7.1 Slave Transmitter 400

14.4.7.2 Slave Receiver 402
14.4.8 I2C Master Mode 404
14.4.8.1 Master Transmitter 404
14.4.8.2 Master Receiver 407
14.4.9 I2C Low-Power Mode 410
14.4.9.1 Low-Power Mode Sequence 410
14.5 I2C Interrupts 411
14.6 I2C Registers 412
14.6.1 I2Cn_DR: I2C Data Register 413
14.6.2 I2Cn_SR: I2C Status Register 414
14.6.3 I2Cn_SAR0: I2C Slave Address Register 415
14.6.4 I2Cn_SAR1: I2C Slave Address Register 415
14.6.5 I2Cn_CR: I2C Control Register 416
14.6.6 I2Cn_SCLL: I2C SCL Low-Duration Register 418
14.6.7 I2Cn_SCLH: I2C SCL High-Duration Register 419
14.6.8 I2Cn_SDH: I2C SDA Hold Register 420
14.6.9 I2C Register Map Summary 421
15. USART (Universal Synchronous/Asynchronous Receiver/Transmitter) 422
15.1 USART Introduction 422
15.2 USART Main Features 422
15.3 USART Functional Description 423
15.3.1 USART Block Diagram 424
15.3.2 USART Pins and Signals 426
15.3.3 USART Clock Generation 427
15.3.4 External Clock (SCKn) 428
15.3.5 Synchronous Mode Operation 428
15.3.6 UART Data Format 429
15.3.7 UART Parity Bit 430
15.3.8 UART Transmitter 431
15.3.8.1 UART Sending TX Data 431
15.3.8.2 UART Transmitter Flag and Interrupt 431
15.3.8.3 UART Parity Generator 432
15.3.8.4 UART Disabling Transmitter 432
15.3.9 UART Receiver 433
15.3.9.1 UART Receiving RX Data 433
15.3.9.2 UART Receiver Flag and Interrupt 434
15.3.9.3 UART Parity Checker 434
15.3.9.4 UART Disabling Receiver 434
15.3.9.5 Asynchronous Data Reception 435
15.3.10 UART Low-Power Mode 437
15.3.10.1 Description of Steps of Low-Power Mode 437
15.3.11 SPI Mode 438
15.3.11.1 SPI Clock Formats and Timing 438
15.3.12 Baud Rate Settings (Example) 441
15.3.13 0% Error Baud Rate 442
15.3.14 Receive Time-Out (RTO) 442
15.4 USART Registers 443
15.4.1 USARTn_CR1: USART n Control Register 1 444
15.4.2 USARTn_CR2: USART n Control Register 2 446
15.4.3 USARTn_ST: USART n Status Register 448

15.4.4 USARTn_BDR: USART n Baud Rate Generation Register 450
15.4.5 USARTn_DR: USART n Data Register 450
15.4.6 USARTn_BFR: USART n Baud Rate Fraction Count Register 451
15.4.7 USARTn_RTO: USART n RTO Register 451
15.4.8 USART Register Map Summary 452
16. LPUART (Low-Power Universal Asynchronous Receiver/Transmitter) 453
16.1 LPUART Introduction 453
16.2 LPUART Main Features 453
16.3 LPUART Functional Description 454
16.3.1 LPUART Block Diagram 454
16.3.2 LPUART Pins and Signals 455
16.3.3 LPUARTn Clock Generation 455
16.3.4 LPUARTn Baud Rate Compensation 456
16.3.5 LPUARTn Interface Data Format 457
16.3.6 LPUARTn Transmitter 457
16.3.7 LPUARTn Sending TX Data 457
16.3.8 LPUARTn Parity Generator 457
16.3.9 LPUARTn Receiver 458
16.3.10 LPUARTn Receiving RX Data 458
16.3.11 LPUARTn Parity Checker 458
16.3.12 LPUARTn Data Reception 459
16.3.13 LPUARTn Receive Time-Out Function 461
16.3.14 1-Wire Half-Duplex Communication 461
16.3.15 Continuous Communication Using DMA and LPUART 461
16.3.16 LPUART Low-Power Mode 462
16.3.16.1 Wake-Up Source 462
16.3.16.2 Description of Steps of Low-Power Mode 462
16.4 LPUART Registers 463
16.4.1 LPUARTn_CR1: LPUART n Control Register1 464
16.4.2 LPUARTn_CR2: LPUART n Control Register2 466
16.4.3 LPUARTn_IER: LPUART n Interrupt Enable Register 468
16.4.4 LPUARTn_IFSR: LPUART n Interrupt Flag and Status Register 469
16.4.5 LPUARTn_RDR: LPUART n Receive Data Register 471
16.4.6 LPUARTn_TDR: LPUART n Transmit Data Register 471
16.4.7 LPUARTn_BDR: LPUART n Baud Rate Generation Register 472
16.4.8 LPUARTn_BCMP: LPUART n Baud Rate Compensation Register 472
16.4.9 LPUARTn_RTODR: LPUART n Receive Time-Out Data Register 473
16.4.10 LPUARTn_RCDR: LPUART n Receive Character Detection Register 473
16.4.11 LPUARTn_DLYDR: LPUART n Delay Time Data Register 474
16.4.12 LPUART Register Map Summary 475
17. USB Full Speed Device Interface 476
17.1 USB Introduction 476
17.2 USB Main Features 476
17.3 USB Implementation 477
17.4 USB Functional Description 478
17.4.1 USB Block Diagram 478
17.4.2 USB Pins and Signals 479
17.4.3 USB Description of USB Block 480
17.5 Programming Considerations 481
17.5.1 Select Clock Source 481

17.5.2 Endpoint Initialization 481
17.5.3 Endpoint Control 481
17.5.4 USB Interrupt Handling 482
17.5.5 Power Saving 483
17.5.6 Buffer Control 484
17.5.7 Handling Transactions with USB Device Peripheral 485
17.6 USB Registers 487
17.6.1 USB_FADDR: Function Address Register 489
17.6.2 USB_POWER: Power Management Register 490
17.6.3 USB_INTRIN1: Interrupt for EP0 + IN EP1 to EP5 Register 491
17.6.4 USB_INTROUT1: Interrupt for OUT EP1 to EP5 Register 492
17.6.5 USB_INTRUSB: Interrupt for Common USB 493
17.6.6 USB_INTRIN1E: Interrupt Enable for INTIN 494
17.6.7 USB_INTROUT1E: Interrupt Enable for INTOUT 495
17.6.8 USB_INTRUSBE: Interrupt Enable for INTUSB 496
17.6.9 USB_FRAME1: Frame Number Bit 0 to 7 497
17.6.10 USB_FRAME2: Frame Number Bit 8 to 10 497
17.6.11 USB_INDEX: Index for Selecting Endpoint Register 498
17.6.12 USB_PUPDCR: DP/DN Pins Pull-Up/Pull-Down Resistor Control Register 498
17.6.13 USB_PHYCR: USB Power Control Register 499
17.6.14 USB_INMAXP: Maximum Package Size for IN Register 499
17.6.15 USB_CSR0: Control Status Register 0 for EP0 Register 500
17.6.16 USB_INCSR1: Control Status Register 1 for IN Register 501
17.6.17 USB_INCSR2: Control Status Register 2 for IN Register 502
17.6.18 USB_OUTMAXP: Maximum Package Size for OUT Register 503
17.6.19 USB_OUTCSR1: Control Status Register 1 for OUT Register 504
17.6.20 USB_OUTCSR2: Control Status Register 2 for OUT Register 505
17.6.21 USB_COUNT0: Number of Bytes for EP0 Register 506
17.6.22 USB_OUTCOUNT1: Number of Bytes for Lower Byte Register 506
17.6.23 USB_OUTCOUNT2: Number of Bytes for Upper Byte Register 507
17.6.24 USB_FIFOn: FIFO for EP0 to EP5 Register 507
17.6.25 USB_DMAINTR: USB DMA Interrupt Pending Register 508
17.6.26 USB_DMA1CNTL: USB DMA Channel 1 Control Register 509
17.6.27 USB_DMA1ADDR: USB DMA Channel 1 Memory Address Register 510
17.6.28 USB_DMA1COUNT: USB DMA Channel 1 Transfer Byte Count Register 510
17.6.29 USB_DMA2CNTL: USB DMA Channel 2 Control Register 511
17.6.30 USB_DMA2ADDR: USB DMA Channel 2 Memory Address Register 512
17.6.31 USB_DMA2COUNT: USB DMA Channel 2 Transfer Byte Count Register 512
17.6.32 USB Register Map Summary 513
18. LCD Driver/Controller 517
18.1 LCD Introduction 517
18.2 LCD Main Features 517
18.3 LCD Functional Description 518
18.3.1 LCD Block Diagram 518
18.3.2 LCD Pins and Signals 519
18.3.3 LCD Display RAM Organization 520
18.3.4 LCD Signal Waveform 521
18.3.5 Internal Resistor Bias Connection 525
18.3.6 External Resistor Bias Connection 526
18.3.7 Capacitor Bias Connection 527

18.4 LCD Registers 528
18.4.1 LCD_CR: LCD Driver Control Register 529
18.4.2 LCD_CCR: LCD Contrast Control Register 530
18.4.3 LCD_DRx: LCD Display Data Register 531
18.4.4 LCD Register Map Summary 532
19. ADC (Analog-to-Digital Converter) 533
19.1 ADC Introduction 533
19.2 ADC Main Features 534
19.3 ADC Functional Description 535
19.3.1 ADC Block Diagram 535
19.3.2 ADC Pins and Signals 536
19.3.3 ADC Connectivity 538
19.3.3.1 ADC Internal Channel Wiring 538
19.3.4 ADC Clocks (ACLK) 539
19.3.4.1 ADC Module Control 539
19.3.4.2 ADC Clock Sources 539
19.3.5 ADC On and Off Control (ADEN) 540
19.3.5.1 Software Procedure to Enable ADC 540
19.3.5.2 Software Procedure to Disable ADC 540
19.3.6 Constraints when Writing ADC Control Bits 540
19.3.6.1 When Writing ADC Control Bit 540
19.3.7 Channel Selection 541
19.3.7.1 ADC Analog Input Channel 541
19.3.7.2 ADC Conversion Channel Selection 541
19.3.7.3 Single Ended and Differential Input Channels 541
19.3.7.4 Register Access when Changing A/D Conversion Channel 542
19.3.8 Programmable Sampling Time 543
19.3.9 Single Mode Conversion (ADMOD[1:0] = '00', SEQCNT[4:0] = 0) 544
19.3.10 Sequential Mode Conversion (ADMOD[1:0] = '00', SEQCNT[4:0] > 0) 546
19.3.11 Burst Mode Conversion (ADMOD[1:0] = '01', SEQCNT > 0) 549
19.3.11.1 Burst Continuous Mode 551
19.3.12 Multiple Mode Conversion (ADMOD[1:0] = '10', SEQCNT[4:0] > 0) 552
19.3.13 Starting Conversions (ASTART, AESTART, Start Trigger, ARST) 554
19.3.13.1 Start A/D Conversion (ASTART) 554
19.3.13.2 All Start A/D Conversion (AESTART) 554
19.3.13.3 Reset A/D Conversion (ARST) 554
19.3.13.4 Start A/D Conversion by Trigger Sources 555
19.3.14 ADC Timing 556
19.3.15 Stopping an Ongoing Conversion (ASTOP) 559
19.3.16 Conversion on External Trigger (TRGSEL[1:0], SEQTRGx[3:0] and TRGSRC[3:0], TRGIE, TRGIF) 560
19.3.16.1 ADC Trigger Source Selection (TRGSEL[1:0]) 560
19.3.16.2 ADC Trigger Channel Settings (SEQTRGx[3:0] and TRGSRC[3:0]) 560
19.3.16.3 ADC Trigger Start Control 561
19.3.16.4 ADC Trigger Interrupt and Flag (TRGIE, TRGIF) 561
19.3.17 End-of-Conversion, End-of-Sampling Phase (EOC) 562
19.3.18 End-of-Sequence Conversion (EOSIE, EOSIF) 563
19.3.19 Data Management and Information 565
19.3.19.1 Data Register and Data Alignment by Software 565

19.3.19.2 Conversion Data Offset Alignment 566
19.3.19.3 Channel and Trigger Information 567
19.3.20 Trigger Pending Function in Sequential Mode 568
19.3.21 ADC Low-Power Modes 568
19.3.22 A/D Conversion Using the DMA 569
19.3.23 ADC Overrun (OVR) in DMA Mode 571
19.3.24 Comparison of ADC Sensing Value 572
19.4 ADC Interrupts 574
19.5 ADC Registers 575
19.5.1 ADCn_MR: ADC Mode Register 577
19.5.2 ADCn_CSCR: ADC Current Sequence/Channel Register 579
19.5.3 ADCn_CCR: ADC Clock Control Register 580
19.5.4 ADCn_TRG1: ADC Trigger Select 1 Register 580
19.5.5 ADCn_TRG2: ADC Trigger Select 2 Register 581
19.5.6 ADCn_TRG3: ADC Trigger Select 3 Register 582
19.5.7 ADCn_TRG4: ADC Trigger Select 4 Register 583
19.5.8 ADCn_CSTR1: ADC Channel Sampling Time 1 Register 583
19.5.9 ADCn_CSTR2: ADC Channel Sampling Time 2 Register 584
19.5.10 ADCn_CSTR3: ADC Channel Sampling Time 3 Register 584
19.5.11 ADCn_CSTR4: ADC Channel Sampling Time 4 Register 585
19.5.12 ADCn_CSTR5: ADC Channel Sampling Time 5 Register 585
19.5.13 ADCn_CSTR6: ADC Channel Sampling Time 6 Register 586
19.5.14 ADCn_CSTR7: ADC Channel Sampling Time 7 Register 586
19.5.15 ADCn_SCSR1: ADC Channel Select 1 Register 587
19.5.16 ADCn_SCSR2: ADC Channel Select 2 Register 588
19.5.17 ADCn_SCSR3: ADC Channel Select 3 Register 589
19.5.18 ADCn_SCSR4: ADC Channel Select 4 Register 590
19.5.19 ADCn_SCSR5: ADC Channel Select 5 Register 591
19.5.20 ADCn_SCSR6: ADC Channel Select 6 Register 592
19.5.21 ADCn_SCSR7: ADC Channel Select 7 Register 593
19.5.22 ADCn_CR: ADC Control Register 594
19.5.23 ADCn_SR: ADC Status Register 595
19.5.24 ADCn_IER: ADC Interrupt Enable Register 596
19.5.25 ADCn_DDR: ADC DMA Data Register 597
19.5.26 ADCn_DRx: ADC Data 0 to 25 Register 598
19.5.27 ADCn_CMP0R: ADC Channel Comparison 0 Register 599
19.5.28 ADCn_CMP1R: ADC Channel Comparison 1 Register 600
19.5.29 ADC Register Map Summary 601
20. VREFBUF (Voltage Reference Buffer) 604
20.1 VREFBUF Introduction 604
20.2 VREFBUF Functional Description 605
20.2.1 VREFBUF Reset and Clock 605
20.2.2 VREFBUF Mode 606
20.2.3 Voltage Reference Output Initialization 607
20.3 VREFBUF Interrupt 608
20.4 VREFBUF Registers 609
20.4.1 VREFBUF_SR: VREFBUF Status Register 610
20.4.2 VREFBUF_CR: VREFBUF Control Register 611
20.4.3 VREFBUF_TCR: VREFBUF Voltage Control Register 612
20.4.4 VREFBUF Register Map Summary 613

21. OPAMP (Operational Amplifier) 614
21.1 OPAMP Introduction 614
21.2 OPAMP Main Features 614
21.3 OPAMP Functional Description 615
21.3.1 OPAMP Block Diagram 615
21.3.2 OPAMP Pins and Signals 616
21.3.3 OPAMP Reset and Clock 617
21.3.4 Initial Configuration 617
21.3.5 OPAMP Modes 617
21.4 OPAMP Low-Power Mode 618
21.5 OPAMP Registers 619
21.5.1 OPAMP_MR: OPAMP Mode Control Register 620
21.5.2 OPAMPn_CR: OPAMP n Control Register 621
21.5.3 OPAMP Register Map Summary 622
22. DAC (Digital-to-Analog Converter) 623
22.1 DAC Introduction 623
22.2 DAC Main Features 623
22.3 DAC Functional Description 624
22.3.1 DAC Block Diagram 624
22.3.2 DAC Pins and Signals 625
22.3.3 DAC Reset and Clock 625
22.3.4 DAC Channel Enable 626
22.3.5 DAC Output Buffer Enable 626
22.3.6 DAC Conversion 626
22.4 DAC Registers 627
22.4.1 DACn_MR: DACn Mode Register 628
22.4.2 DACn_CR: DACn Control Register 629
22.4.3 DAC Register Map Summary 630
23. CRC (Cyclic Redundancy Check Calculation Module) 631
23.1 CRC Introduction 631
23.2 CRC Main Features 631
23.3 CRC Functional Description 632
23.3.1 CRC Block Diagram 632
23.3.2 CRC Internal Signals 632
23.3.3 Timing Diagram of CRC Internal Signals for Data Size 633
23.3.4 CRC Operation 635
23.3.4.1 How to Obtain CRC Calculation Result 635
23.3.5 CRC Using DMA 637
23.4 CRC Registers 638
23.4.1 CRC_CTRL: CRC Control Register 639
23.4.2 CRC_INIT: CRC Initial Data Register 640
23.4.3 CRC_IDR: CRC Input Data Register 640
23.4.4 CRC_ODR: CRC Output Data Register 641
23.4.5 CRC_STAT: CRC Status Register 641
23.4.6 CRC Register Map Summary 642
24. RTCC (Real Time Clock and Calendar) 643
24.1 RTCC Introduction 643
24.2 RTC Main Features 643
24.3 RTCC Functional Description 644
24.3.1 RTCC Block Diagram 644

24.3.2 RTCC Pins and Signals 645
24.3.3 Starting the RTCC Operation 646
24.3.4 RTCC RW Mode (Read and Write Mode) 647
24.3.5 RTCC Alarm Setting 649
24.3.6 RTCC Constant-Period Matching 651
24.3.7 RTCC Timestamp 651
24.3.8 RTCC 1 Hz Output 651
24.3.9 Error Correction of RTCC 652
24.3.9.1 Example of Calculating Correction Value 652
24.3.9.2 Correction Example 653
24.4 RTCC Registers 654
24.4.1 RTCC_CR: RTCC Control Register 655
24.4.2 RTCC_ERRCOR: RTCC Error Correction Register 657
24.4.3 RTCC_SEC: RTCC Second Counter Register 658
24.4.4 RTCC_MIN: RTCC Minute Counter Register 658
24.4.5 RTCC_HOUR: RTCC Hour Counter Register 659
24.4.6 RTCC_DAY: RTCC Day Counter Register 660
24.4.7 RTCC_WEEK: RTCC Week Counter Register 661
24.4.8 RTCC_MONTH: RTCC Month Counter Register 662
24.4.9 RTCC_YEAR: RTCC Year Counter 663
24.4.10 RTCC_ALARMW: RTCC Alarm Register 663
24.4.11 RTCC_SUBDR: RTCC Sub-Second Counter Compare Register 664
24.4.12 RTCC_TIME_TS: RTCC TimeStamp Time Register 664
24.4.13 RTCC_DATE_TS: RTCC TimeStamp Date Register 665
24.4.14 RTCC Register Map Summary 666
Abbreviation for Registers 668
Glossary 669
Revision History 671

List of Tables

Table 1. A34L716 Series Memory Capacity 27
Table 2. A34L716 Series Features 28
Table 3. Pin Description(1) 39
Table 4. Interrupt Vector Map 51
Table 5. Peripheral Memory Map 57
Table 6. Boot Mode Pin List 59
Table 7. SCU Pins 61
Table 8. Clock Sources 62
Table 9. Peripheral Clock Selection 63
Table 10. Clock Generation Circuit Input/Output Pins 64
Table 11. HSE Clock Pins 65
Table 12. LSE Clock Pins 68
Table 13. A34L716 Clock Monitoring Function 75
Table 14. Clock Source for Clock Monitoring 75
Table 15. Flash Wait Control Recommendation 81
Table 16. Reset Sources for Cold Reset and Warm Reset 83
Table 17. LVR Level 87
Table 18. nRESET Pin 87
Table 19. Power Supply 90
Table 20. Operation Mode 94
Table 21. DEEP-SLEEP Mode Configuration 96
Table 22. Configurable Clock in Each RUN/SLEEP Mode 98
Table 23. Configurable Clocks in Each DEEP-SLEEP (STOP1, STOP2) Mode 98
Table 24. Base Address of CHIPCONFIG 99
Table 25. Register Map of CHIPCONFIG 99
Table 26. Base Address of SCU 100
Table 27. Register Map of SCU 100
Table 28. External Oscillator Control Settings 138
Table 29. SCU CHIPCONFIG Register Map Summary 147
Table 30. SCU Register Map Summary 147
Table 31. PCU and GPIO Internal Signal 157
Table 32. PCU and GPIO Pins 157
Table 33. GPIO Alternate Function (1) 160
Table 34. Interrupt Sources and Corresponding Pins in GPIO Module 163
Table 35. Base Address of PCU 168
Table 36. PCU and GPIO Register Map 168
Table 37. PCU and GPIO Register Map Summary 179
Table 38. Code Flash Memory Controller Features 183
Table 39. Data Flash Memory Controller Features 184
Table 40. Internal Code Flash Memory Wait Time by Operating Clock 185
Table 41. Internal Data Flash Wait Time by Operating Clock 186
Table 42. Available Operating Modes by Protection Level 193
Table 43. Read Protection Operation in UNPROT Level 194
Table 44. Read Protection Operation in Level-1 (LVL 1) 194
Table 45. Read Protection Operation in Level-2 (LVL 2) 195
Table 46. Code Flash Memory Write Protection Area in 16 KB Units 197
Table 47. Code Flash 0 User Boot Loader Protection Area (Bottom 32 KB) in 4 KB Units 197

Table 48. Code Flash Memory Unprotection Area (Top 4 KB) in 512 Bytes Units 197
Table 49. Data Flash Protection Area in 2 KB Units 198
Table 50. Data Flash Unprotection Area (Top 4 KB) in 512 Bytes Units 198
Table 51. Base Address of CFMC 201
Table 52. CFMC Register Map 201
Table 53. CFMC Register Map Summary 217
Table 54. Base Address of Data Flash Memory Controller 219
Table 55. DFMC Register Map 219
Table 56. DFMC Register Map Summary 226
Table 57. Examples HSICMU (SYNCSRC[1:0] = 00, SOF) 234
Table 58. Examples HSICMU (SYNCSRC[1:0] = 10, LSE) 234
Table 59. Base Address of HSICMU 235
Table 60. HSICMU Register Map 235
Table 61. HSICMU Register Map Summary 242
Table 62. Base Addresses of DMA 247
Table 63. DMA Register Map 247
Table 64. DMA PERISEL[4:0] Selection 248
Table 65. DMA Register Map Summary 251
Table 66. Prescaled WDT Counter Clock Frequency 254
Table 67. Base Address of WDT 257
Table 68. WDT Register Map 257
Table 69. WDT Register Map Summary 262
Table 70. Input and Output Pins for TIMERn (n = 0 to 7) 266
Table 71. Internal Input and Output Signals for TIMERn (n = 0 to 7) 267
Table 72. PCLK Clock Selection by CKSEL[2:0] in TIMERn_CR1 268
Table 73. SCU_MCCR3 Register and CKSEL[2:0] in TIMERn_CR1 268
Table 74. TnIO and IOSEL in TIMERn_CR1 269
Table 75. TnIO and CKSEL[2:0] in TIMERn_CR1 269
Table 76. Operations Comparison According to TIMERn_GRA/GRB Settings 273
Table 77. Operation Status by Capture Mode 278
Table 78. One-Shot Mode Operation View Table 282
Table 79. Interrupt Requests 292
Table 80. Base Address of 16-bit Timer 293
Table 81. 16-bit Timer Register Map 293
Table 82. TIMER Register Map Summary 302
Table 83. Base Address of FRT 309
Table 84. FRT Register Map 309
Table 85. FRT Register Map Summary 312
Table 86. Summarized Features of UART 315
Table 87. Pin Assignment of UART: External Pins 317
Table 88. Received Bit Value from Sampled Data 330
Table 89. Examples of Baud Rate Calculation 333
Table 90. Examples Calculation of Baud Rate Using UART Fraction Counter 334
Table 91. Tolerance of UART Receiver 335
Table 92. UART Frame Formats 336
Table 93. Various Configurations to Create a Parity Bit 337
Table 94. Base Address of UART 343
Table 95. UART Register Map 343
Table 96. Interrupt ID and Control 346
Table 97. UART Register Map Summary 354

Table 98. Summarized Features of SPI 356
Table 99. Pin Assignment of SPI: External Pins (n = 0 and 1) 358
Table 100. Base Address of SPI 375
Table 101. SPI Register Map 375
Table 102. SPI Register Map Summary 382
Table 103. Features of I2C 384
Table 104. Pin Assignment of I2C: External Pins 386
Table 105. Effect of Low-Power Modes on I2C 410
Table 106. Base Address of I2C 412
Table 107. I2C Register Map 412
Table 108. I2C Register Map Summary 421
Table 109. Pin Assignment of USART: External Pins 426
Table 110. Equations for Calculating USART Baud Rate Register Settings (n= 10 and 11) 427
Table 111. Effect of Low-Power Mode on UART 437
Table 112. CPOL Functionality (n = 10 and 11) 438
Table 113. Baud Rate Settings Example 441
Table 114. Base Addresses of USART 443
Table 115. UART Register Map 443
Table 116. USART Register Map Summary 452
Table 117. Pin Assignment of LPUART: External Pins (n = 0 and 1) 455
Table 118. Equations for Calculating Baud Rate Register Settings 455
Table 119. Baud Rate Compensation Example 1 456
Table 120. Effect of Low-Power Mode on LPUART 462
Table 121. Base Addresses of LPUART 463
Table 122. LPUART Register Map 463
Table 123. LPUART Register Map Summary 475
Table 124. A34L716 USB Implementation 477
Table 125. Pin Assignment of USB: External Pins 479
Table 126. Base Address of USB 487
Table 127. USB Register Map 487
Table 128. USB Register Map Summary 513
Table 129. Pin Assignment of LCD: External Pins 519
Table 130. Base Address of LCD 528
Table 131. LCD Register Map 528
Table 132. LCD Register Map Summary 532
Table 133. Pin Assignment of ADC External Pins 536
Table 134. ADC Input/Output Signals 537
Table 135. ADC Frequency for R_{AIN} (1) 557
Table 136. ADC Frequency for R_{AIN} (2) 557
Table 137. External Trigger Source Configuration 560
Table 138. Trigger Source Table 560
Table 139. Start Control of AD Trigger Conversion 561
Table 140. Effect of Low-Power Modes on ADC 568
Table 141. Comparison and Flag Generation Conditions by CMPM 572
Table 142. Control Bits and Event Flags for ADC Interrupts (n = 0 and 1) 574
Table 143. Base Address of ADC 575
Table 144. ADC Register Map 575
Table 145. ADC Register Map Summary 601
Table 146. VREFBUF Modes 606
Table 147. Base Address of VREFBUF 609

Table 148. VREFBUF Register Map 609
Table 149. VREFBUF Register Map Summary 613
Table 150. Pin Assignment of OPAMP External Pins 616
Table 151. OPAMP Input/Output Signals 616
Table 152. Effect of Low-Power Mode on OPAMP 618
Table 153. Base Address of OPAMP 619
Table 154. OPAMP Register Map 619
Table 155. OPAMP Register Map Summary 622
Table 156. Pin Assignment of DAC External Pins 625
Table 157. Base Addresses of DAC 627
Table 158. DAC Register Map 627
Table 159. DAC Register Map Summary 630
Table 160. CRC Internal Input/Output Signals 632
Table 161. Base Address of CRC 638
Table 162. CRC Register Map 638
Table 163. CRC Register Map Summary 642
Table 164. Base Address of RTCC 644
Table 165. Pin Assignment of RTCC: External Pins 645
Table 166. WEEK and ALARMWW Registers 650
Table 167. Example of Setting Alarm 650
Table 168. Correction Resolution by Frequency 652
Table 169. Base Address of RTCC 654
Table 170. CRC Register Map 654
Table 171. Specified Time System by AMPM 659
Table 172. RTCC Register Map Summary 666
Table 173. Abbreviation 668

List of Figures

Figure 1. A34L716 Conceptual Block Diagram 1
Figure 2. 100-LQFP Pinouts 36
Figure 3. 64-LQFP Pinouts 37
Figure 4. 56-QFN Pinouts 38
Figure 5. System Block Diagram 49
Figure 6. Interrupt Block Diagram 54
Figure 7. Memory Map 56
Figure 8. SCU Block Diagram 61
Figure 9. Clock Configuration 64
Figure 10. HSE Crystal/Ceramic Oscillator 65
Figure 11. PLL Block Diagram 66
Figure 12. LSE External Clock 68
Figure 13. LSE PCB Design Considerations 69
Figure 14. Miscellaneous Clock Configuration 72
Figure 15. Clock Monitoring Detection Range 74
Figure 16. MCLK Monitoring Workflow 77
Figure 17. HSE Monitoring Workflow 78
Figure 18. LSE Monitoring Workflow 79
Figure 19. Clock Change Procedure 80
Figure 20. Peripheral Clock Selection 82
Figure 21. Reset Tree Configuration 84
Figure 22. Timing Diagram of Power-Up Procedure (Cold Reset) 85
Figure 23. Warm Reset Timing Diagram 86
Figure 24. LVR Diagram 87
Figure 25. POR and LVR Timing Diagram 91
Figure 26. LVI Block Diagram 91
Figure 27. LVR Block Diagram 92
Figure 28. POR and LVR Block Diagram 92
Figure 29. Transition between Operation Modes 94
Figure 30. SLEEP Mode Operation Sequence 95
Figure 31. DEEP-SLEEP Mode Operation Sequence 97
Figure 32. PCU and GPIO Block Diagram 154
Figure 33. I/O Port Block Diagram (GPIOs) 155
Figure 34. I/O Port Block Diagram (ADC, DAC, OPAMP and External Oscillator Pins) 156
Figure 35. Output Port Block Diagram 164
Figure 36. Input Port Block Diagram 165
Figure 37. Port Function Block Diagram 166
Figure 38. Debouncing Logic Block Diagram 167
Figure 39. Example Timing Diagram of Port Debouncing 167
Figure 40. Code Flash Memory Map 183
Figure 41. Data Flash Memory Map 184
Figure 42. Flash Memory Page Erase Timing Diagram 189
Figure 43. Flash Memory 2 KB Sector Erase Timing Diagram 190
Figure 44. Flash Memory Bulk (Full-Chip) Erase Timing Diagram 191
Figure 45. Flash Memory Program Timing Diagram 192
Figure 46. HSICMU Block Diagram 228
Figure 47. HSICMU Counter Behavior 229

Figure 48. HSICMU Initialization Flowchart 232
Figure 49. DMA Block Diagram 243
Figure 50. Diagram of DMA Transfer Timing from Peripheral to Memory 245
Figure 51. Diagram of DMA Transfer Timing from Memory to Peripheral 246
Figure 52. N Number of DMA Transfers 246
Figure 53. WDT Block Diagram 253
Figure 54. Interrupt Mode Operation Timing Diagram with External Clock Set as WDT Clock 255
Figure 55. 16-bit Timer Block Diagram (n = 0 to 7) 265
Figure 56. Counter Timing Diagram with Basic Start and Match Operations 271
Figure 57. Normal Periodic Mode Operation 272
Figure 58. Operation Waveform Depending on UAO 274
Figure 59. Operation Waveform According to STARTLVL and OUTPOL Settings 275
Figure 60. Timing Diagram of Capture Mode Operation 277
Figure 61. Timing Diagram of PWM Output Operation 279
Figure 62. Timing Diagram of One-Shot Mode Operation 281
Figure 63. Timing Diagram of ADC Triggering (TRGMOD[1:0] = ‘00’) 284
Figure 64. Timing Diagram of ADC Triggering (TRGMOD[1:0] = ‘10’) 284
Figure 65. Timing Diagram of ADC Triggering (TRGMOD[1:0] = ‘11’) 285
Figure 66. Timer Synchronization Example (when SSYNC = ‘1’, CSYNC = ‘1’) 287
Figure 67. Timer Synchronization Example (when CSYNC = ‘1’) 288
Figure 68. Delayed Timer Synchronization Example (when SSYNC = ‘1’) 289
Figure 69. SSYNC Operation Example 290
Figure 70. CSYNC Operation Example 290
Figure 71. FRT Block Diagram 304
Figure 72. Match Interrupt Operation Timing Diagram 305
Figure 73. Overflow Interrupt Operation Timing Diagram 306
Figure 74. Timing Diagram in Low-Power Mode 308
Figure 75. UART Block Diagram 316
Figure 76. Data Inversion Control Diagram 318
Figure 77. Timing Diagram of 8-bit Data Length Programming 319
Figure 78. Timing Diagram of 7-bit Data Length Programming 319
Figure 79. Timing Diagram of 6-bit Data Length Programming 320
Figure 80. Timing Diagram of 5-bit Data Length Programming 320
Figure 81. Example Format of Transmit Data 321
Figure 82. Transmit Interrupt Timing Diagram 322
Figure 83. Example Format of Transmit Data 325
Figure 84. Sampling Timing of UART Receiver 326
Figure 85. Start Bit Detection of Single-Sampling Timing 327
Figure 86. Start Bit Detection of Multi-Sampling Timing 328
Figure 87. Single-Sampling Timing of Data Bit 329
Figure 88. Multi-Sampling Timing of Data Bit 329
Figure 89. Data Sampling Timing 330
Figure 90. Transmission Using DMA 339
Figure 91. Reception Using DMA 340
Figure 92. Error Fragging and Interrupt Generation Timing Diagram 341
Figure 93. SPI Block Diagram 357
Figure 94. Block Diagram of SPI-Master to Slave Connections: Full-Duplex 359
Figure 95. Block Diagram of SPI-Master to Slave Connections: Simplex 359
Figure 96. SPI Transfer Timing 1 of 4 (CPHA = ‘0’, CPOL = ‘0’, MSBF = ‘0’) 362
Figure 97. SPI Transfer Timing 2 of 4 (CPHA = ‘0’, CPOL = ‘1’, MSBF = ‘1’) 362

Figure 98. SPI Transfer Timing 3 of 4 (CPHA = '1', CPOL = '0', MSBF = '0') 363
Figure 99. SPI Transfer Timing 4 of 4 (CPHA = '1', CPOL = '1', MSBF = '1') 363
Figure 100. DMA Handshake State Diagram 370
Figure 101. Example of Timing of SPI Transfer Operation Using DMA 372
Figure 102. SPI Waveforms (STL[7:0], BTL[7:0], and SPL[7:0])(1) 381
Figure 103. I2C Block Diagram 385
Figure 104. Noise Canceller Timing 387
Figure 105. I2C Bus Bit Transfer Timing Diagram 390
Figure 106. START and STOP Conditions 391
Figure 107. I2C Bus Response 392
Figure 108. Clock Synchronization during Arbitration 393
Figure 109. Arbitration Process between Two Masters 394
Figure 110. I2C Bus Data Transfer 395
Figure 111. Master Transmit Timing 396
Figure 112. Master Receive Timing 397
Figure 113. Slave Transmit Timing 398
Figure 114. Slave Receive Timing 399
Figure 115. Slave Transmitter Flowchart 400
Figure 116. Slave Receiver Flowchart 402
Figure 117. Master Transmitter Flowchart 404
Figure 118. Master Receiver Flowchart 407
Figure 119. Internal Delay (INTDEL[1:0]) in Master Mode 417
Figure 120. SCL Low (SCLL[15:0]) Timing 418
Figure 121. SCL High (SCLH[15:0]) Timing 419
Figure 122. SDA Hold (SDH[14:0]) Timing 420
Figure 123. UART Block Diagram (n = 10 and 11) 424
Figure 124. SPI Block Diagram (n = 10 and 11) 425
Figure 125. Clock Generation Block Diagram (n = 10 and 11) 427
Figure 126. Synchronous Mode SCKn Timing (USARTn, n = 10 and 11) 428
Figure 127. Frame Format (USART) 429
Figure 128. Asynchronous Start Bit Sampling (n = 10 and 11) 435
Figure 129. Asynchronous Data and Parity Bit Sampling (n = 10 and 11) 436
Figure 130. Stop Bit Sampling and Next Start Bit Sampling (n = 10 and 11) 436
Figure 131. USART SPIn Clock Formats when CPHAn = 0 (n = 10 and 11) 439
Figure 132. USART SPIn Clock Formats when CPHAn = 1 (n = 10 and 11) 440
Figure 133. 0% Error Baud Rate Diagram 442
Figure 134. LPUART Block Diagram (n = 0 and 1) 454
Figure 135. Start Bit Sampling 459
Figure 136. Sampling of Data and Parity Bit 460
Figure 137. Stop Bit Sampling and Next Start Bit Sampling 460
Figure 138. Start Bit Sampling 461
Figure 139. USART DEALST, DEALFT, DEPOL 467
Figure 140. USB Block Diagram 478
Figure 141. USB Interrupt Service Routine 482
Figure 142. USB Buffer Control 484
Figure 143. Setup Transaction Followed by Data IN Transaction 485
Figure 144. Setup Transaction Followed by Data Out Transfer 486
Figure 145. LCD Block Diagram 518
Figure 146. LCD Display Ram Organization 520
Figure 147. LCD Signal Waveforms (1/2-Duty, 1/2-Bias) 521

Figure 148. LCD Signal Waveforms (1/3-Duty, 1/3-Bias) 522
Figure 149. LCD Signal Waveforms (1/4-Duty, 1/3-Bias) 523
Figure 150. LCD Signal Waveforms (1/8-Duty, 1/4-Bias) 524
Figure 151. Internal Resistor Bias Connection 525
Figure 152. External Resistor Bias Connection 526
Figure 153. Capacitor Bias Connection 527
Figure 154. 12-bit ADC Block Diagram 535
Figure 155. 12-bit ADC Internal Channel Wiring 538
Figure 156. Block Diagram of ADC Clock 539
Figure 157. Example of ADC Conversion Timing 543
Figure 158. ADC Single Mode Timing (when ADMOD[1:0] = '00') 545
Figure 159. ADC Sequential Mode Timing (when ADMOD[1:0] = '00' and SEQCNT[4:0] = 7) 547
Figure 160. ADC Trigger Timing in Sequential Mode (SEQCNT[4:0] = '00111', 8-Sequential Conversion) 548
Figure 161. ADC Burst Mode Timing (when ADMOD[1:0] = '01') 550
Figure 162. ADC Burst Continuons Mode 551
Figure 163. Analog-to-Digital Conversion Time 556
Figure 164. Typical Connection Diagram with ADC 558
Figure 165. Stopping Ongoing A/D Conversions 559
Figure 166. Example of A/D Conversion Timing in Sequence Mode 564
Figure 167. Left Alignment (Unsigned Value) 565
Figure 168. Right Alignment (Unsigned Value) 566
Figure 169. Channel and Trigger Source Information of A/D Conversion Sequence 567
Figure 170. Block Diagram of ADC Internal Comparator 572
Figure 171. Timing Diagram of Comparison Between A/D Conversion Results and Reference Value 573
Figure 172. OPAMP Block Diagram 615
Figure 173. DAC Block Diagram 624
Figure 174. DAC Conversion 626
Figure 175. CRC Block Diagram 632
Figure 176. Timing Diagram of CRC Internal Signals: 8-bit Size Input and CRC-8 Outputs 633
Figure 177. Timing Diagram of CRC Internal Signals: 32-bit Size Input and CRC-8 Outputs 633
Figure 178. Timing Diagram of CRC Internal Signals: 32-bit Size Input and CRC-16 Outputs 634
Figure 179. Timing Diagram of CRC Internal Signals: 32-bit Size Input and CRC-32 Outputs 634
Figure 180. Timing Diagram of CRC Operation using DMA 637
Figure 181. RTCC Block Diagram 644
Figure 182. Start Operation Sequence of RTCC 646
Figure 183. Read Operation Sequence of RTCC 647
Figure 184. Sub-second Counter Match in RW Mode 648
Figure 185. Sub-second Counter Match in RW Mode when Writing to SEC 648
Figure 186. Alarm Operation Sequence of RTCC 649

1. Description

The A34L716 series is a 32-bit high-performance microcontroller with up to 256 KB of flash memory and 24 KB SRAM. It is a powerful microcontroller which provides effective solutions to various electrical appliances which require both low power consumption and high performance (Arm Cortex-M4 core with FPU.)

1.1 Product Category Definition

This document provides an overview of the features supported by each device, including high-level information and brief explanations for each feature. Refer to Table 2 for the list of features supported by each product category.

Table 1 provides an overview of the memory capacity for available devices in the A34L716 series.

Table 1. A34L716 Series Memory Capacity

Memory Capacity		Category
Flash	RAM	
256 KB	24 KB	A34L716

1.2 Device Overview

Table 2 summarizes and lists product specific features specific to the device in the A34L716 series while considering the largest package.

Table 2. A34L716 Series Features

Peripherals		Description
Core	CPU	• Maximum operating frequency: 32 MHz • 32-bit Arm Cortex-M4F core - 32-bit Thumb®-2 instruction set • CPU register set: - Uses 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 86 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. A34L716 Series Features (continued)

Peripherals		Description
Memory	Code flash	• Capacity: 256 KB • A high-capacity code flash memory built in • Max 25 MHz flash access speed • 512 bytes, 2 KB block erases • Bulk erase • Read protection • Self-programming • CRC code generation and verification for the flash memory • Endurance: 10,000 cycles • Lifetime: 10 years
	Data flash	• Capacity: 32 KB • Max 25 MHz access speed • 512 bytes, 2 KB block erases • CRC code generation and verification for the flash memory • Endurance: 100,000 Cycles • Lifetime: 10 years
	BOOT ROM	• Executes the processor's boot mode when receiving an input at the nBOOT pin from an external circuit • UART boot mode • In-System Programming - A user can program data into the internal flash memory by setting an application board.
	SRAM	• Capacity: 24 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
System Control Unit (SCU)	Operating frequency	Up to 32 MHz
	Clock	• High-speed internal oscillator (HSI): 32 MHz - HSICMU ON: ±0.5% @-40°C to +85°C - HSICMU OFF: ±3.0% @-10°C to +70°C, ±5.0% @-20°C to +85°C • Low-speed internal oscillator (LSI): 40 kHz (±30% @ -40°C to +85°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 48 MHz)

Table 2. A34L716 Series Features (continued)

Peripherals		Description
System Control Unit (SCU)	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 (PC12) • Core reset • Software reset • POR (Power-On Reset) • LVR (Low-Voltage Reset) • WDTR (Watchdog Timer Reset) • Reset due to clock oscillating error
	VDC	Low-dropout (LDO) regulator built in for low-voltage operation
	POR	The POR generator detects an internal 1.4 V voltage and generates a reset signal
	LVI	• Seven Low-Voltage Detection levels • Supports interrupts • Supports wake-up from DEEP-SLEEP mode
	Wake-up	• Wake-up by a General-Purpose Input/Output (GPIO) pins • Wake-up by a Free-Run Timer (FRT) • Wake-up by a Watchdog Timer (WDT) • Wake-up by a Low-Voltage Indicator (LVI) • Wake-up by a Real-Time Clock and Calendar (RTCC) • Wake-up by Inter-Integrated Circuit (I2C) • Wake-up by a Universal Synchronous and Asynchronous Receiver/Transmitter (USART) • Wake-up by a Low Power Universal Asynchronous Receiver/Transmitter (LPUART) • Wake-up by a USB

Table 2. A34L716 Series Features (continued)

Peripherals	Description
General Purpose I/O (GPIO)	• Input / output (I/O) port for general purposes • 100-LQFP - I/O pins: 86 • 64-LQFP - I/O pins: 53 • 56-QFN - I/O pins: 48 • Selectable input and output configuration for each pin: - Push-pull output - Open drain output - Input • Pin configurations through multiplexer settings • Configurable as an external interrupt source with the following options: - Rising-edge (high-level) - Falling-edge (low-level) - Edge-triggered (both rising-edge and falling-edge) • Configurable pull-up/down resistors and debouncing • Adjustable drive strength • Configurable independently to be set or reset through its corresponding bit • Wake-up events triggered by external asynchronous inputs
Direct Memory Access Controller (DMA)	• 7-ch direct memory access (DMA) supports peripherals • 8-/ 16-/ 32-bit data transfers • Compatible with 18 different types of peripherals • UART, USART10, USART11, LPUART0, LPUART1, SPI0, SPI1, ADC0, ADC1, CRC

Table 2. A34L716 Series Features (continued)

Peripherals		Description
TIMER	16-bit Timer	• General-purpose 16-bit up-count timer • 8 channels - 8 input capture channels (TnIO) - 8 output channels (TnIO) • Timer operating modes - Periodic timer mode - One-shot mode - PWM mode - Capture mode • Interrupt events - Timer / counter match interrupt - Timer overflow interrupt • Free selectable clock sources: - PCLK configurable through the peripheral clock enable register1 (SCU_PCER1) - LSI, LSE, MCLK, HSI, HSE or PLL configurable through the miscellaneous clock control register 3 (SCU_MCCR3) • Timer signals can be generated through TnIO pins • 10-bit prescaler
	WDT	• 32-bit down-count timer • Reset and periodic interrupts • Free selectable clock sources: - LSI, LSE, MCLK, HSI, HSE or PLL configurable through the miscellaneous clock control register 1 (SCU_MCCR1) • Selectable eight different prescalers
	FRT	• 32-bit Free-run Timer - Timer for system time measurement - 32-bit up-count timer • Interrupt event - Period interrupt - Overflow interrupt

Table 2. A34L716 Series Features (continued)

Peripherals		Description
TIMER	RTCC	• Real-time clock and Calendar • LSE clock as RTCC clock (f_{RTCC}) • Counters of year, month, week, day, hour, minute, and second (up to 99 years) • Interrupt event - Constant period matching interrupt function (period: 0.5 seconds, 1 second, 1 minute, 1 hour, 1 day, 1 month) - Alarm interrupt function (alarm: week, hour, minute) - Timestamp interrupt and function • Pin output of 1Hz clock • Error correction by 1.53ppm resolution when using 32768Hz LSE
Serial Interface	UART	• One 16450 asynchronous serial communication port • 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	• Two synchronous serial communication port channels • Master / slave operation • Loop-back mode • Flexible communication available through programming - 8-/ 9-/ 16-/ 17-bit data transmission / reception - SPI clock speed - Both Least Significant Bit (LSB)-first and Most Significant Bit (MSB)-first modes available
	I2C	• Standard I2C communication protocol • Two channel support • Master and slave modes • 7-bit addressing for slave mode • Configurable SCL signal’s high/low periods and SDA signal’s hold time

Table 2. A34L716 Series Features (continued)

Peripherals		Description
Serial Interface	LPUART	• Full-duplex and Half-duplex • 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, 2-bit stop bit generation and checking • Receive Character Detection and Receive Time-out function • Baud rate compensation • Up to 9600 bps with 32.768 kHz LSE • Data overrun and frame error detection
	USART	• Full Duplex Operation. (Independent Serial Receive and Transmit Registers) • Two Channel USART module • Asynchronous or Synchronous Operation. • 8-bit fraction controller and 16-bit baud rate generator • Flexible communication available through programming - 5- to 9-bit data transfers - Even / Odd / Non-parity generation and checking - 1-bit, 2-bit stop bit generation and checking • Receive character detection and receive time-out function • Data overrun and frame error detection • Three Separate Interrupts on TX Completion, TX Data Register Empty and RX Completion. • Double speed asynchronous communication mode.
	USB	• Crystal-less with clock recovery system • Compliant with USB 2.0 Full-Speed • 1 KB SRAM dedicated to USB • Endpoint 1 to 5 support. • IN and OUT Endpoint. • Two DMA channels, 64 bytes buffer size

Table 2. A34L716 Series Features (continued)

Peripherals	Description
12-bit AD Converter (ADC)	• Two independent ADC blocks • 24 analog input channels • Multiple operating modes: - Single conversion - Sequence conversion - Burst conversion - Multiple conversion • Support for up to 26 sequential conversions • Software triggers • One internal trigger source (timers) • Sample time and hold time adjustable • Support for the simultaneous start of two ADC blocks • In Burst mode, support for Continuous mode
12-bit D/A Converter (DAC)	• Two units of 12-bit DAC • Selectable output to the AIPn pin of OPAMP block.
Operational Amplifier (OPAMP)	• Two OPAMPs, each with two input sources • Compatible with ADC for joint operation
Cyclic Redundancy Check (CRC)	• CRC operating modes: - CRC32 (0x04C1_1DB7) - CRC16_USB (0x8005) - CRC16_CCITT (0x1021) - CRC8_CCITT (0x07) • Support for byte reversal of input/output data • Compatible with DMA
LCD Driver	• 8-COM x 40-SEG or 4-COM x 44-SEG (100-LQFP) • 8-COM x 18-SEG or 4-COM x 22-SEG (64-LQFP) • 8-COM x 13-SEG or 4-COM x 17-SEG (56-QFN) • Resistor/Capacitor bias • 8-step contrast control
Operating Voltage	1.8 V to 3.6 V
Operating temperature	Commercial grade (-40°C to +85°C)
Package	• Three types of package options - 100-pin LQFP - 64-pin LQFP - 56-pin QFN

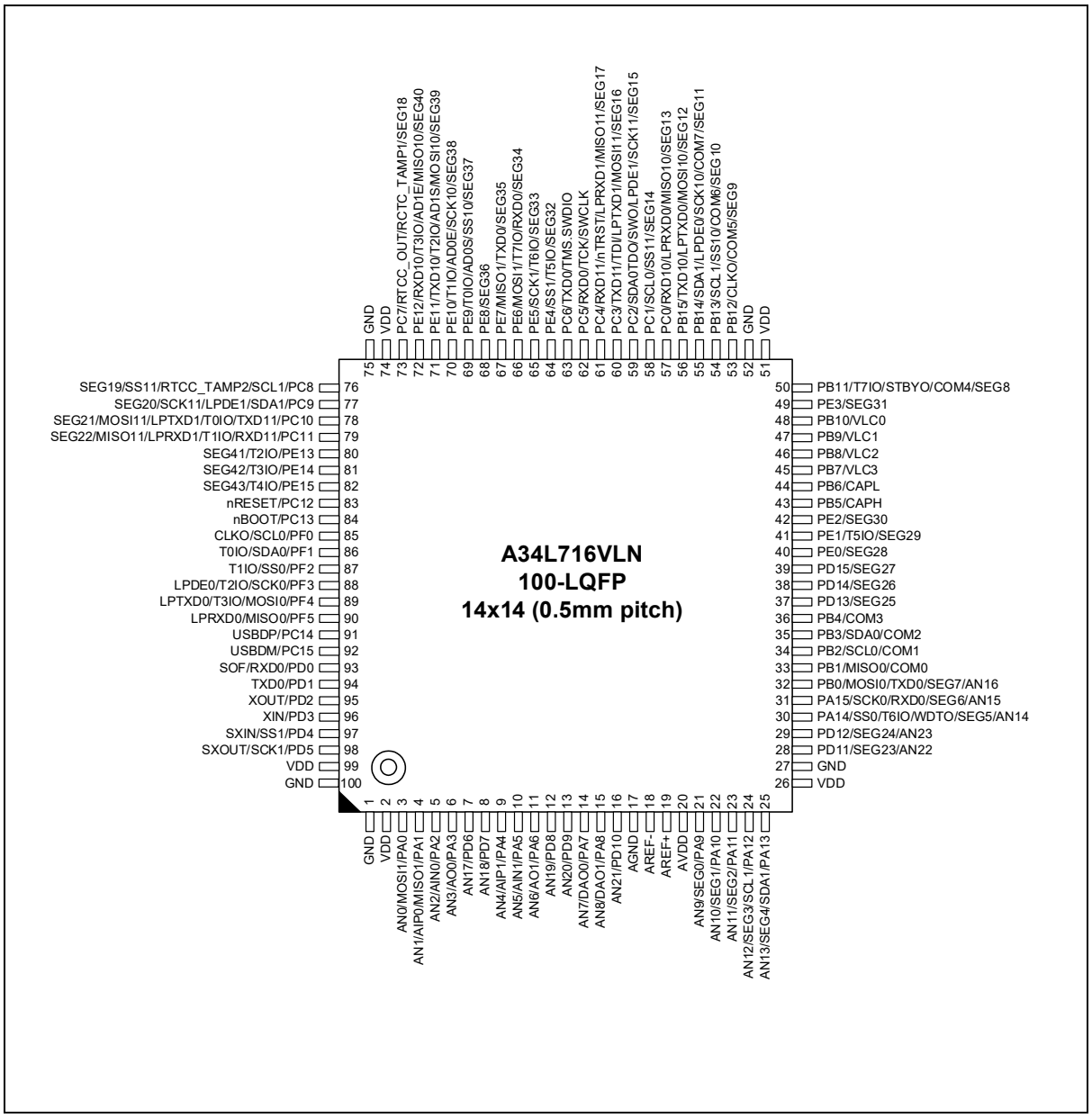
2. Pinouts and Pin Descriptions

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

2.1 Pinouts

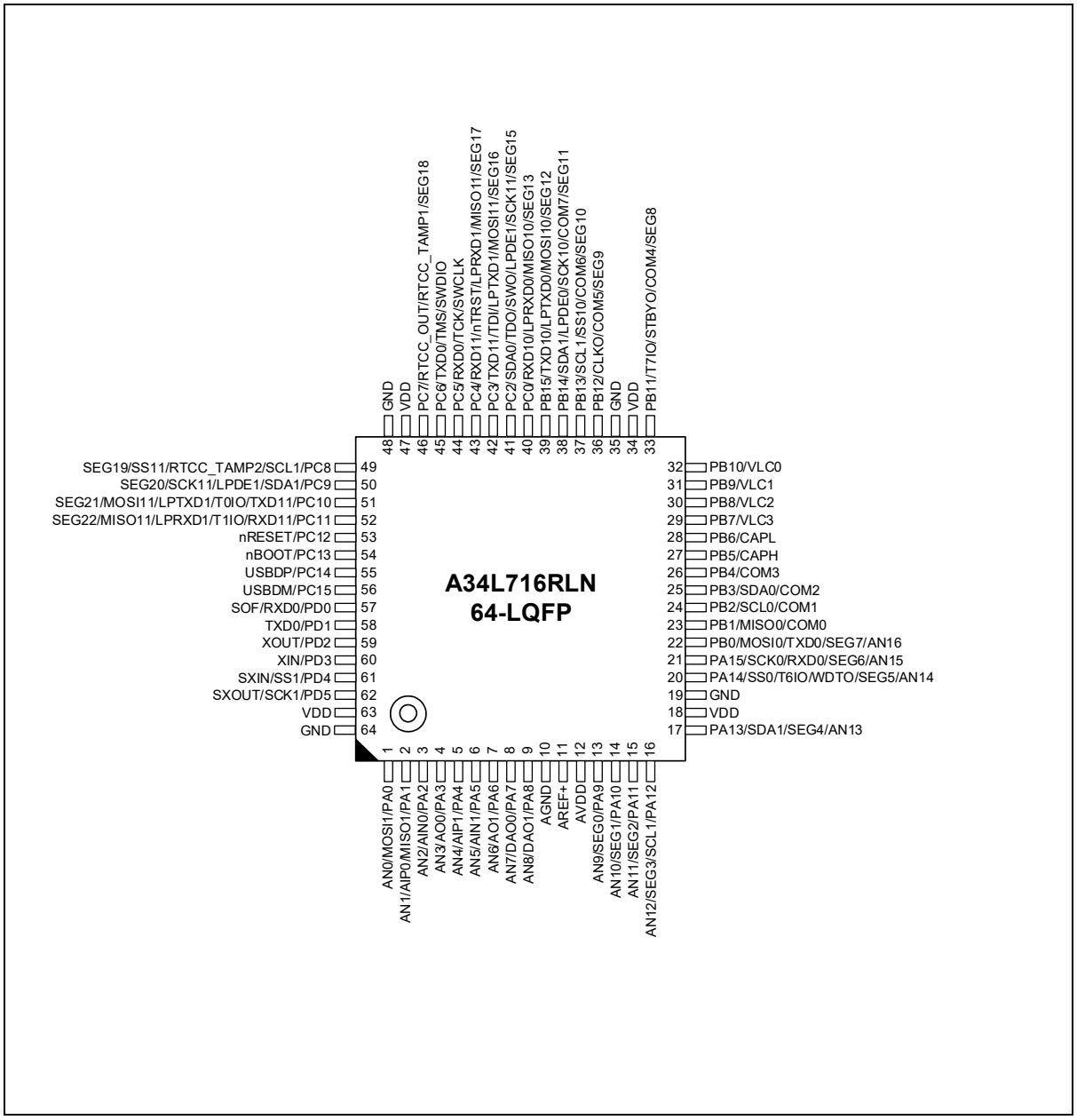
2.1.1 A34L716VLN (100-LQFP)

Figure 2. 100-LQFP Pinouts



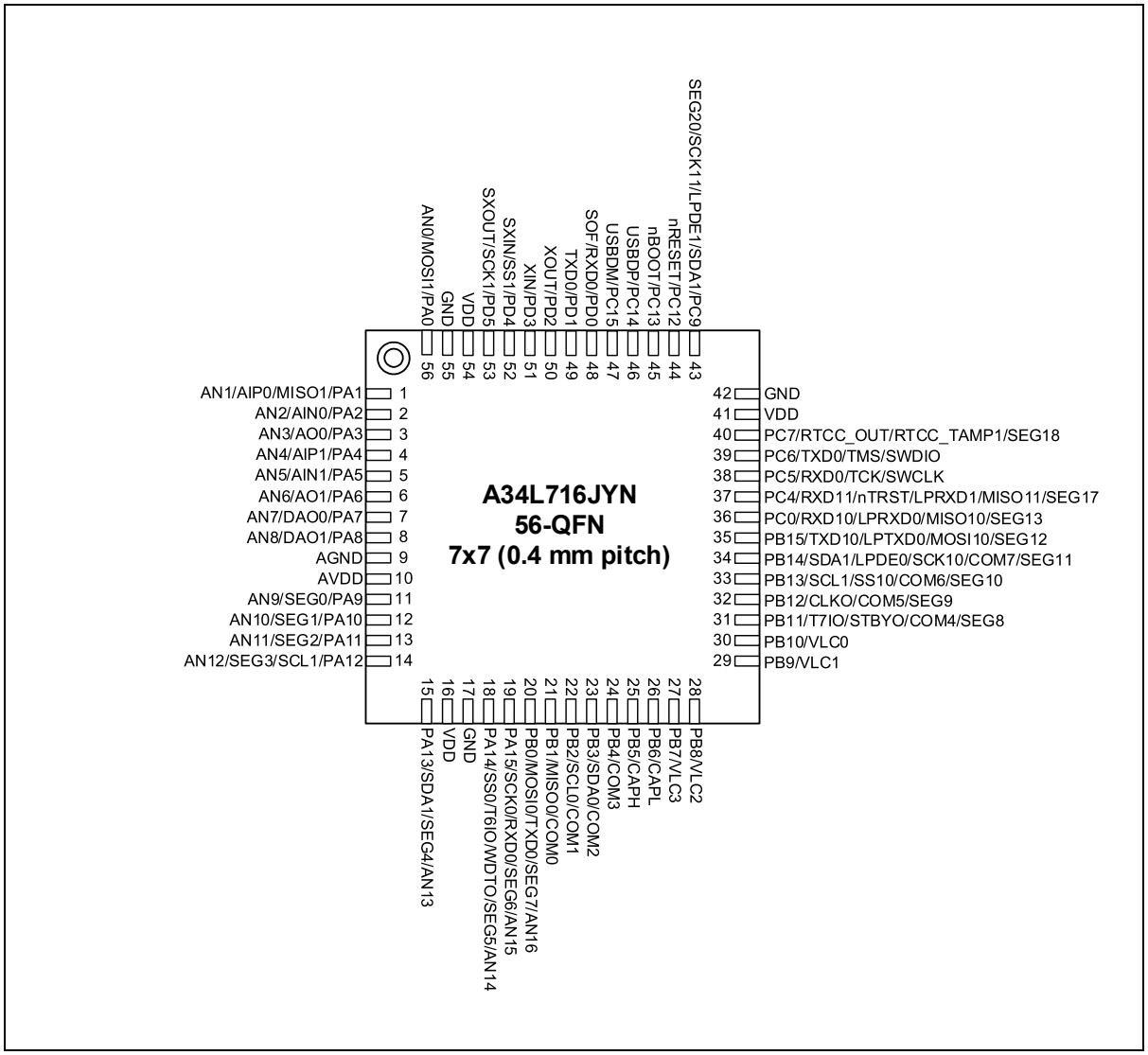
2.1.2 A34L716RLN (64-LQFP)

Figure 3. 64-LQFP Pinouts



2.1.3 A34L716JYN (56-QFN)

Figure 4. 56-QFN 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 seven selections of functions including GPIO. Configuration including pin ordering can be changed without notice.

Table 3. Pin Description(1)

Pin Number			Pin Name	Type	Description	Remark
100-LQFP	64-LQFP	56-QFN				
1	-	-	GND	P	GND	
2	-	-	VDD	P	VDD	
3	1	56	PA0	IOUS	PORT A Bit 0 Input/Output	
			MOSI1	IO	SPI CH1 Master Out/Slave In Signal	
			AN0(2)	IA	Analog Input 0	
4	2	1	PA1	IOUS	PORT A Bit 1 Input/Output	
			MISO1	IO	SPI CH1 Master In/Slave Out Signal	
			AN1/AIP0(2)	IA	Analog Input 1/OPAMP Positive Input 0	
5	3	2	PA2	IOUS	PORT A Bit 2 Input/Output	
			AN2/AIN0(2)	IA	Analog Input 2/OPAMP Negative Input 0	
6	4	3	PA3	IOUS	PORT A Bit 3 Input/Output	
			AN3/AO0(2)	IA	Analog Input 3/OPAMP Output 0	
7	-	-	PD6	IOUS	PORT D Bit 6 Input/Output	
			AN17(2)	IA	Analog Input 17	
8	-	-	PD7	IOUS	PORT D Bit 7 Input/Output	
			AN18(2)	IA	Analog Input 18	
9	5	4	PA4	IOUS	PORT A Bit 4 Input/Output	
			AN4/AIP1(2)	IA	Analog Input 4/OPAMP Positive Input 1	
10	6	5	PA5	IOUS	PORT A Bit 5 Input/Output	
			AN5/AIN1 (2)	IA	Analog Input 5/OPAMP Negative Input 1	
11	7	6	PA6	IOUS	PORT A Bit 6 Input/Output	
			AN6/AO1(2)	IA	Analog Input 6/OPAMP Output 1	
12	-	-	PD8	IOUS	PORT D Bit 8 Input/Output	
			AN19(2)	IA	Analog Input 19	
13	-	-	PD9	IOUS	PORT D Bit 9 Input/Output	
			AN20(2)	IA	Analog Input 20	
14	8	7	PA7	IOUS	PORT A Bit 7 Input/Output	
			AN7/DAO0(2)	IOA	Analog Input 7/Digital to analog output 0	
15	9	8	PA8	IOUS	PORT A Bit 8 Input/Output	
			AN8/DAO1(2)	IOA	Analog Input 8/Digital to analog output 1	
16	-	-	PD10	IOUS	PORT D Bit 10 Input/Output	
			AN21(2)	IA	Analog Input 21	
17	10	9	AGND	P	Analog Ground	
18	-	-	AREF-	P	Analog Reference Ground	
19	11	-	AREF+	P	Analog Reference VDD	
20	12	10	AVDD	P	Analog VDD	

Table 3. Pin Description(1) (continued)

Pin Number			Pin Name	Type	Description	Remark
100-LQFP	64-LQFP	56-QFN				
21	13	11	PA9	IOUS	PORT A Bit 9 Input/Output	
			SEG0	OA	LCD Segment Signal Output	
			AN9(2)	IA	Analog Input 9	
22	14	12	PA10	IOUS	PORT A Bit 10 Input/Output	
			SEG1	OA	LCD Segment Signal Output	
			AN10(2)	IA	Analog Input 10	
23	15	13	PA11	IOUS	PORT A Bit 11 Input/Output	
			SEG2	OA	LCD Segment Signal Output	
			AN11(2)	IA	Analog Input 11	
24	16	14	PA12	IOUS	PORT A Bit 12 Input/Output	
			SCL1	IO	I2C CH1 SCL Input/Output	
			SEG3	OA	LCD Segment Signal Output	
			AN12(2)	IA	Analog Input 12	
25	17	15	PA13	IOUS	PORT A Bit 13 Input/Output	
			SDA1	IO	I2C CH1 SDA Input/Output	
			SEG4	OA	LCD Segment Signal Output	
			AN13(2)	IA	Analog Input 13	
26	18	16	VDD	P	VDD	
27	19	17	GND	P	GND	
28	-	-	PD11	IOUS	PORT D Bit 11 Input/Output	
			SEG23	OA	LCD Segment Signal Output	
			AN22(2)	IA	Analog Input 22	
29	-	-	PD12	IOUS	PORT D Bit 12 Input/Output	
			SEG24	OA	LCD Segment Signal Output	
			AN23(2)	IA	Analog Input 23	
30	20	18	PA14	IOUS	PORT A Bit 14 Input/Output	
			SS0	IO	SPI CH0 Select Signal Input/Output	
			T6IO	IO	Timer 6 Input/Output	
			WDTO	O	WDT Output	
			SEG5	OA	LCD Segment Signal Output	
			AN14(2)	IA	Analog Input 14	

Table 3. Pin Description[1] (continued)

Pin Number			Pin Name	Type	Description	Remark
100-LQFP	64-LQFP	56-QFN				
31	21	19	PA15	IOUS	PORT A Bit 15 Input/Output	
			SCK0	IO	SPI CH0 Clock Input/Output	
			RXD0	I	UART CH0 RXD Input	
			SEG6	OA	LCD Segment Signal Output	
			AN15[2]	IA	Analog Input 15	
32	22	20	PB0	IOUS	PORT B Bit 0 Input/Output	
			MOSI0	IO	SPI CH0 Master Out/Slave In Signal	
			TXD0	O	UART CH0 TXD Output	
			SEG7	OA	LCD Segment Signal Output	
			AN16[2]	IA	Analog Input 16	
33	23	21	PB1	IOUS	PORT B Bit 1 Input/Output	
			MISO0	IO	SPI CH0 Master In/Slave Out Signal	
			COM0	OA	LCD Common Signal Output	
34	24	22	PB2	IOUS	PORT B Bit 2 Input/Output	
			SCL0	IO	I2C CH0 SCL Input/Output	
			COM1	OA	LCD Common Signal Output	
35	25	23	PB3	IOUS	PORT B Bit 3 Input/Output	
			SDA0	IO	I2C CH0 SDA Input/Output	
			COM2	OA	LCD Common Signal Output	
36	26	24	PB4	IOUS	PORT B Bit 4 Input/Output	
			COM3	OA	LCD Common Signal Output	
37	-	-	PD13	IOUS	PORT D Bit 13 Input/Output	
			SEG25	OA	LCD Segment Signal Output	
38	-	-	PD14	IOUS	PORT D Bit 14 Input/Output	
			SEG26	OA	LCD Segment Signal Output	
39	-	-	PD15	IOUS	PORT D Bit 15 Input/Output	
			SEG27	OA	LCD Segment Signal Output	
40	-	-	PE0	IOUS	PORT E Bit 0 Input/Output	
			SEG28	OA	LCD Segment Signal Output	
41	-	-	PE1	IOUS	PORT E Bit 1 Input/Output	
			T5IO	IO	Timer 5 Input/Output	
			SEG29	OA	LCD Segment Signal Output	
42	-	-	PE2	IOUS	PORT E Bit 2 Input/Output	
			SEG30	OA	LCD Segment Signal Output	

Table 3. Pin Description[1] (continued)

Pin Number			Pin Name	Type	Description	Remark
100-LQFP	64-LQFP	56-QFN				
43	27	25	PB5[9]	IOUS	PORT B Bit 5 Input/Output	
			CAPH	I	Capacitor terminal for voltage booster	
44	28	26	PB6[9]	IOUS	PORT B Bit 6 Input/Output	
			CAPL	I	Capacitor terminal for voltage booster	
45	29	27	PB7[9]	IOUS	PORT B Bit 7 Input/Output	
			VLC3	IOA	LCD bias voltage input/output	
46	30	28	PB8[9]	IOUS	PORT B Bit 8 Input/Output	
			VLC2	IOA	LCD bias voltage input/output	
47	31	29	PB9[9]	IOUS	PORT B Bit 9 Input/Output	
			VLC1	IOA	LCD bias voltage input/output	
48	32	30	PB10[9]	IOUS	PORT B Bit 10 Input/Output	
			VLC0	IOA	LCD bias voltage input/output	
49	-	-	PE3	IOUS	PORT E Bit 3 Input/Output	
			SEG31	OA	LCD Segment Signal Output	
50	33	31	PB11	IOUS	PORT B Bit 11 Input/Output	
			T7IO	IO	Timer 7 Input/Output	
			STBYO	O	Power-down Mode Indication Signal	
			COM4/SEG8	OA	LCD Common Signal Output/ LCD Segment Signal Output	
51	34	-	VDD	P	VDD	
52	35	-	GND	P	Ground	
53	36	32	PB12	IOUS	PORT B Bit 12 Input/Output	
			CLKO	O	System Clock Output	
			COM5/SEG9	OA	LCD Common Signal Output/ LCD Segment Signal Output	
54	37	33	PB13	IOUS	PORT B Bit 13 Input/Output	
			SCL1	IO	I2C CH1 SCL Input/Output	
			SS10	I	SPI(USART) CH0 slave select input	
			COM6/SEG10	OA	LCD Common Signal Output/ LCD Segment Signal Output	
55	38	34	PB14	IOUS	PORT B Bit 14 Input/Output	
			SDA1	IO	I2C CH1 SDA Input/Output	
			LPDE0	O	LPUART CH0 DE Signal Output	
			SCK10	I/O	SPI(USART) CH0 clock input/output	
			COM7/SEG11	OA	LCD Common Signal Output/ LCD Segment Signal Output	

Table 3. Pin Description(1) (continued)

Pin Number			Pin Name	Type	Description	Remark
100-LQFP	64-LQFP	56-QFN				
56	39	35	PB15	IOUS	PORT B Bit 15 Input/Output	
			TXD10	O	UART(USART) CH0 data output	
			LPTXD0	O	LPUART CH0 Data Output	
			MOSI10	IO	SPI(USART) CH0 master output, slave input	
			SEG12	OA	LCD Segment Signal Output	
57	40	36	PC0	IOUS	PORT C Bit 0 Input/Output	
			RXD10	I	UART(USART) CH0 data input	
			LPRXD0	I	LPUART CH0 Data Input	
			MISO10	I/O	SPI(USART) CH0 master input, slave output	
			SEG13	OA	LCD Segment Signal Output	
58	-	-	PC1	IOUS	PORT C Bit 1 Input/Output	
			SCL0	IO	I2C CH0 SCL Input/Output	
			SS11	I	SPI(USART) CH1 slave select input	
			SEG14	OA	LCD Segment Signal Output	
59	41	-	PC2	IOUS	PORT C Bit 2 Input/Output	
			SDA0	IO	I2C CH0 SDA Input/Output	
			LPDE1	O	LPUART CH1 DE Signal Output	
			SCK11	I/O	SPI(USART) CH1 clock input/output	
			SEG15	OA	LCD Segment Signal Output	
			TDO/SWO*	O	JTAG TDO Output, Serial Wire Output	
60	42	-	PC3	IOUS	PORT C Bit 3 Input/Output	
			TXD11	O	UART(USART) CH1 data output	
			LPTXD1	O	LPUART CH1 Data Output	
			MOSI11	IO	SPI(USART) CH1 master output, slave input	
			SEG16	OA	LCD Segment Signal Output	
			TDI*	I	JTAG TDI Input	Internal pull-up enable

Table 3. Pin Description(1) (continued)

Pin Number			Pin Name	Type	Description	Remark
100-LQFP	64-LQFP	56-QFN				
61	43	37	PC4	IOUS	PORT C Bit 4 Input/Output	
			RXD11	I	UART(USART) CH1 data input	
			LPRXD1	I	LPUART CH1 Data Input	
			MISO11	I/O	SPI(USART) CH1 master input, slave output	
			SEG17	OA	LCD Segment Signal Output	
			nTRST*	I	JTAG nTRST Input	Internal pull-up enable
62	44	38	PC5	IOUS	PORT C Bit 5 Input/Output	
			RXD0	I	UART CH0 RXD Input	
			TCK / SWCLK* (2),(3),(6),(7)	I	JTAG TCK, SWD Clock Input	Internal pull-up enable
63	45	39	PC6	IOUS	PORT C Bit 6 Input/Output	
			TXD0	O	UART CH0 TXD Output	
			TMS / SWDIO* (2),(3),(6),(7)	I	JTAG TMS, SWD Data Input/Output	Internal pull-up enable
64	-	-	PE4	IOUS	PORT E Bit 4 Input/Output	
			SS1	IO	SPI CH1 Select Signal Input/Output	
			T5IO	IO	Timer 5 Input/Output	
			SEG32	OA	LCD Segment Signal Output	
65	-	-	PE5	IOUS	PORT E Bit 5 Input/Output	
			SCK1	IO	SPI CH1 Clock Input/Output	
			T6IO	IO	Timer 6 Input/Output	
			SEG33	OA	LCD Segment Signal Output	
66	-	-	PE6	IOUS	PORT E Bit 6 Input/Output	
			MOSI1	IO	SPI CH1 Master Out/Slave In Signal	
			T7IO	IO	Timer 7 Input/Output	
			RXD0	I	UART CH0 RXD Input	
			SEG34	OA	LCD Segment Signal Output	
67	-	-	PE7	IOUS	PORT E Bit 7 Input/Output	
			MISO1	I/O	SPI CH1 master input, slave output	
			TXD0	O	UART CH0 TXD Output	
			SEG35	OA	LCD Segment Signal Output	

Table 3. Pin Description(1) (continued)

Pin Number			Pin Name	Type	Description	Remark
100-LQFP	64-LQFP	56-QFN				
68	-	-	PE8	IOUS	PORT E Bit 8 Input/Output	
			SEG36	OA	LCD Segment Signal Output	
69	-	-	PE9	IOUS	PORT E Bit 9 Input/Output	
			T0IO	IO	Timer 0 Input/Output	
			AD0S	O	ADC0 Start of Conversion	
			SS10	I	SPI(USART) CH0 slave select input	
			SEG37	OA	LCD Segment Signal Output	
70	-	-	PE10	IOUS	PORT E Bit 10 Input/Output	
			T1IO	IO	Timer 1 Input/Output	
			AD0E	O	ADC0 End of Conversion	
			SCK10	I/O	SPI(USART) CH0 clock input/output	
			SEG38	OA	LCD Segment Signal Output	
71	-	-	PE11	IOUS	PORT E Bit 11 Input/Output	
			TXD10	O	UART(USART) CH0 data output	
			T2IO	IO	Timer 2 Input/Output	
			AD1S	O	ADC1 Start of Conversion	
			MOSI10	IO	SPI(USART) CH0 master output, slave input	
			SEG39	OA	LCD Segment Signal Output	
72	-	-	PE12	IOUS	PORT E Bit 12 Input/Output	
			RXD10	I	UART(USART) CH0 data input	
			T3IO	IO	Timer 3 Input/Output	
			AD1E	O	ADC1 End of Conversion	
			MISO10	I/O	SPI(USART) CH0 master input, slave output	
			SEG40	OA	LCD Segment Signal Output	
73	46	40	PC7	IOUS	PORT C Bit 7 Input/Output	
			RTCC_OUT	O	RTCC Output	
			RTCC_TAMP1	I	RTCC TimeStamp Input 1	
			SEG18	OA	LCD Segment Signal Output	
74	47	41	VDD	P	VDD	
75	48	42	GND	P	Ground	

Table 3. Pin Description(1) (continued)

Pin Number			Pin Name	Type	Description	Remark
100-LQFP	64-LQFP	56-QFN				
76	49	-	PC8	IOUS	PORT C Bit 8 Input/Output	
			SCL1	IO	I2C CH1 SCL Input/Output	
			RTCC_TAMP2	I	RTCC TimeStamp Input 2	
			SS11	I	SPI(USART) CH1 slave select input	
			SEG19	OA	LCD Segment Signal Output	
77	50	43	PC9	IOUS	PORT C Bit 9 Input/Output	
			SDA1	IO	I2C CH1 SDA Input/Output	
			LPDE1	O	LPUART CH1 DE Signal Output	
			SCK11	I/O	SPI(USART) CH1 clock input/output	
			SEG20	OA	LCD Segment Signal Output	
78	51	-	PC10	IOUS	PORT C Bit 10 Input/Output	
			TXD11	O	UART(USART) CH1 data output	
			T0IO	IO	Timer 0 Input/Output	
			LPTXD1	O	LPUART CH1 Data Output	
			MOSI11	IO	SPI(USART) CH1 master output, slave input	
			SEG21	OA	LCD Segment Signal Output	
79	52	-	PC11	IOUS	PORT C Bit 11 Input/Output	
			RXD11	I	UART(USART) CH1 data input	
			T1IO	IO	Timer 1 Input/Output	
			LPRXD1	I	LPUART CH1 Data Input	
			MISO11	I/O	SPI(USART) CH1 master input, slave output	
			SEG22	OA	LCD Segment Signal Output	
80	-	-	PE13	IOUS	PORT E Bit 13 Input/Output	
			T2IO	IO	Timer 2 Input/Output	
			SEG41	OA	LCD Segment Signal Output	
81	-	-	PE14	IOUS	PORT E Bit 14 Input/Output	
			T3IO	IO	Timer 3 Input/Output	
			SEG42	OA	LCD Segment Signal Output	
82	-	-	PE15	IOUS	PORT E Bit 15 Input/Output	
			T4IO	IO	Timer 4 Input/Output	
			SEG43	OA	LCD Segment Signal Output	

Table 3. Pin Description(1) (continued)

Pin Number			Pin Name	Type	Description	Remark
100-LQFP	64-LQFP	56-QFN				
83	53	44	PC12	IOUS	PORT C Bit 12 Input/Output	
			nRESET(2),(3)	I	External Reset Input	Internal pull-up enable
84	54	45	PC13	IOUS	PORT C Bit 13 Input/Output	
			nBOOT(2),(3),(5)	I	Boot Mode Selection Input	Internal pull-up enable
85	-	-	PF0	IOUS	PORT F Bit 0 Input/Output	
			SCL0	IO	I2C CH0 SCL Input/Output	
			CLKO	O	System Clock Output	
86	-	-	PF1	IOUS	PORT F Bit 1 Input/Output	
			SDA0	IO	I2C CH0 SDA Input/Output	
			T0IO	IO	Timer 0 Input/Output	
87	-	-	PF2	IOUS	PORT F Bit 2 Input/Output	
			SS0	I	SPI CH0 slave select input	
			T1IO	IO	Timer 1 Input/Output	
88	-	-	PF3	IOUS	PORT F Bit 3 Input/Output	
			SCK0	I/O	SPI CH0 clock input/output	
			T2IO	IO	Timer 2 Input/Output	
			LPDE0	O	LPUART CH0 DE Signal Output	
89	-	-	PF4	IOUS	PORT F Bit 4 Input/Output	
			MOSI0	IO	SPI CH0 master output, slave input	
			T3IO	IO	Timer 3 Input/Output	
			LPTXD0	O	LPUART CH0 Data Output	
90	-	-	PF5	IOUS	PORT F Bit 5 Input/Output	
			MISO0	I/O	SPI CH0 master input, slave output	
			LPRXD0	I	LPUART CH0 Data Input	
91	55	46	PC14	IOUS	PORT C Bit 14 Input/Output	
			USBDP	I/O	USB Data Line Plus	
92	56	47	PC15	IOUS	PORT C Bit 15 Input/Output	
			USBDM	I/O	USB Data Line Minus	
93	57	48	PD0	IOUS	PORT D Bit 0 Input/Output	
			SOF	I	HSICMU External SYNC pin	
			RXD0	I	UART CH0 RXD Input	

Table 3. Pin Description[1] (continued)

Pin Number			Pin Name	Type	Description	Remark
100-LQFP	64-LQFP	56-QFN				
94	58	49	PD1	IOUS	PORT D Bit 1 Input/Output	
			TXD0	O	UART CH0 TXD Output	
95	59	50	PD2	IOUS	PORT D Bit 2 Input/Output	
			XOUT[8]	OA	External Crystal Oscillator Output	
96	60	51	PD3	IOUS	PORT D Bit 3 Input/Output	
			XIN[8]	IA	External Crystal Oscillator Input	
97	61	52	PD4	IOUS	PORT D Bit 4 Input/Output	
			SS1	IO	SPI CH1 Select Signal Input/Output	
			SXIN[8]	IA	Sub Crystal Oscillator Input	
98	62	53	PD5	IOUS	PORT D Bit 5 Input/Output	
			SCK1	IO	SPI CH1 Clock Input/Output	
			SXOUT[8]	OA	Sub Crystal Oscillator Output	
99	63	54	VDD	P	VDD	
100	64	55	GND	P	Ground	

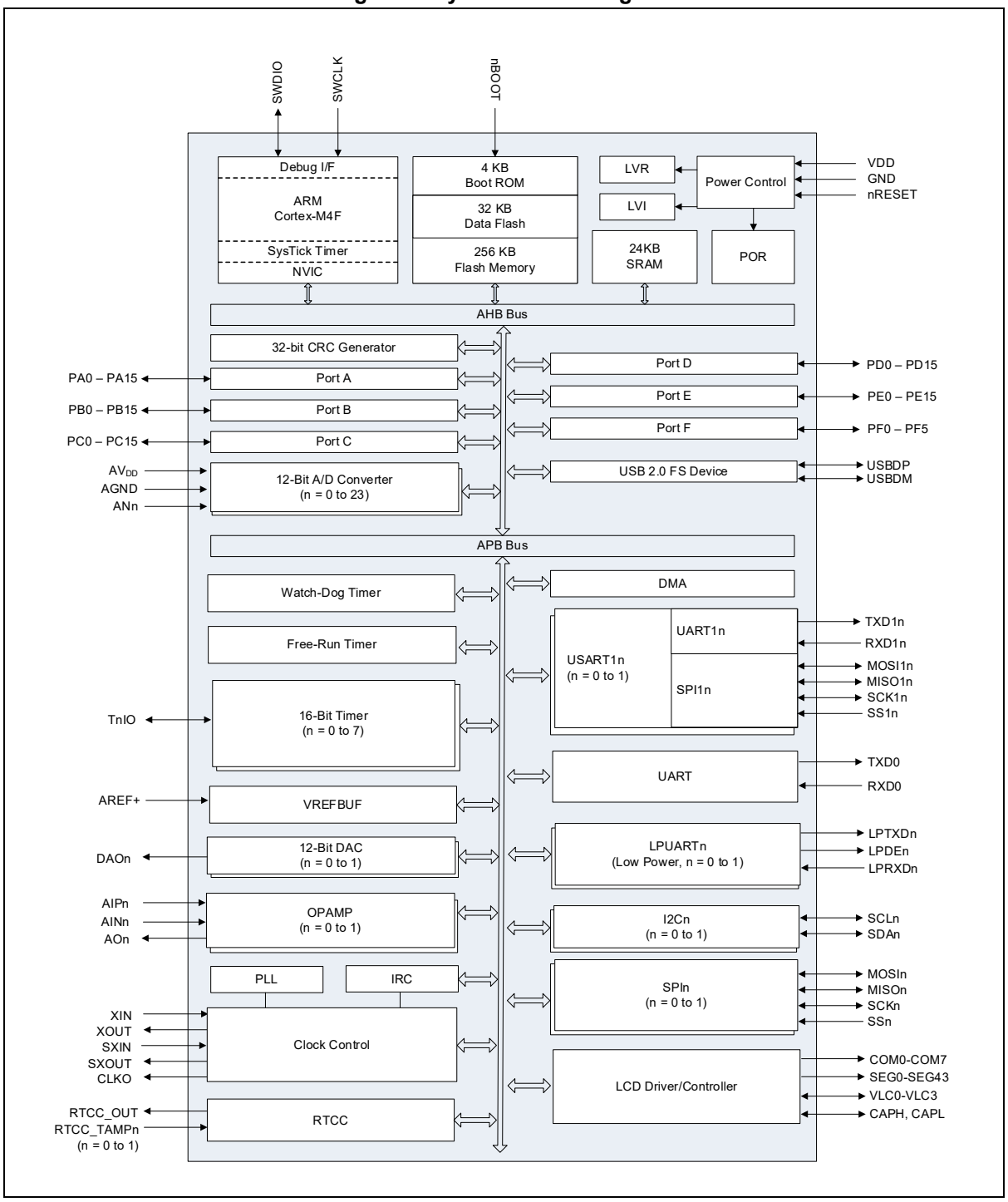
NOTES:

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 100-pin standard.
3. PC13 (nBOOT), PC12 (nRESET), PC6 (SWDIO), PC5 (SWCLK), PC3 (TDI) and PC4 (nTRST) 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. When a function other than a clock is set through the PD3 (XIN), PD2 (XOUT), PD5 (SXOUT) and PD4 (SXIN) pins, other functions than the set clock will not operate normally if the clock is enabled by software.
9. Ports PB5, PB6, PB7, PB8, PB9, and PB10 support the internal pull-down resistor function, but the internal pull-up resistor and port strength functions are not supported. Please take caution when using these ports.

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.

For detailed information on the Cortex-M4F, refer to document ID061113 provided by the ARM.

3.1.2 Floating Point Unit (FPU)

The 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	Reserved
-11	0x0000_0014	
-10	0x0000_0018	
-9	0x0000_001C	
-8	0x0000_0020	
-7	0x0000_0024	
-6	0x0000_0028	
-5	0x0000_002C	SVCall handler
-4	0x0000_0030	Reserved
-3	0x0000_0034	
-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	FRT
9	0x0000_0064	Reserved
10	0x0000_0068	
11	0x0000_006C	CFMC
12	0x0000_0070	DFMC
13	0x0000_0074	Reserved
14	0x0000_0078	
15	0x0000_007C	TIMER0
16	0x0000_0080	TIMER1
17	0x0000_0084	TIMER2
18	0x0000_0088	TIMER3

Table 4. Interrupt Vector Map (continued)

Priority	Vector Address	Interrupt Source
19	0x0000_008C	TIMER4
20	0x0000_0090	TIMER5
21	0x0000_0094	TIMER6
22	0x0000_0098	TIMER7
23	0x0000_009C	Reserved
24	0x0000_00A0	
25	0x0000_00A4	
26	0x0000_00A8	
27	0x0000_00AC	
28	0x0000_00B0	
29	0x0000_00B4	RTCC
30	0x0000_00B8	VREFBUF
31	0x0000_00BC	Reserved
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	Reserved
43	0x0000_00EC	
44	0x0000_00F0	USB
45	0x0000_00F4	Reserved
46	0x0000_00F8	
47	0x0000_00FC	
48	0x0000_0100	
49	0x0000_0104	
50	0x0000_0108	LPUART0
51	0x0000_010C	LPUART1
52	0x0000_0110	Reserved
53	0x0000_0114	
54	0x0000_0118	
55	0x0000_011C	SPI0
56	0x0000_0120	SPI1

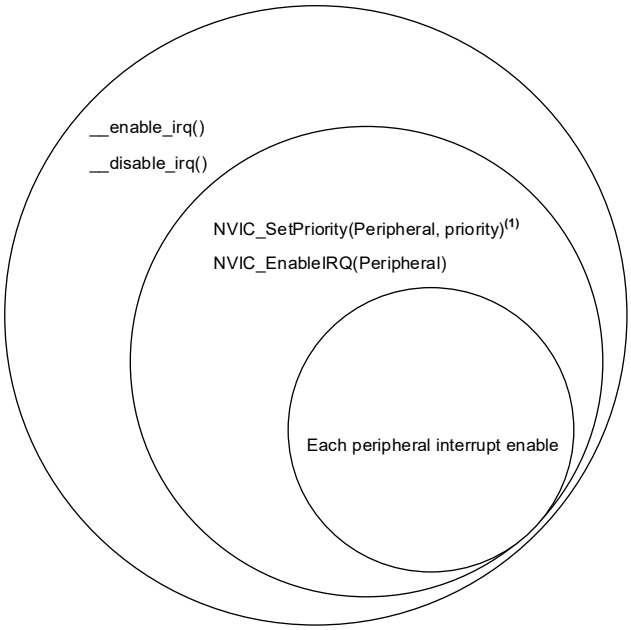
Table 4. Interrupt Vector Map (continued)

Priority	Vector Address	Interrupt Source
57	0x0000_0124	Reserved
58	0x0000_0128	
59	0x0000_012C	
60	0x0000_0130	I2C0
61	0x0000_0134	I2C1
62	0x0000_0138	Reserved
63	0x0000_013C	UART0
64	0x0000_0140	Reserved
65	0x0000_0144	
66	0x0000_0148	
67	0x0000_014C	
68	0x0000_0150	
69	0x0000_0154	USART10
70	0x0000_0158	USART11
71	0x0000_015C	Reserved
72	0x0000_0160	
73	0x0000_0164	Reserved
74	0x0000_0168	ADC0
75	0x0000_016C	ADC1
76	0x0000_0170	Reserved
77	0x0000_0174	
78	0x0000_0178	
79	0x0000_017C	
80	0x0000_0180	
81	0x0000_0184	
82	0x0000_0188	
83	0x0000_018C	
84	0x0000_0190	
85	0x0000_0194	CRC

NOTE:

1. Each interrupt is linked to the priority level registers. Each interrupt priority register is one byte length and only 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



NOTE:
1. Option for interrupt priority

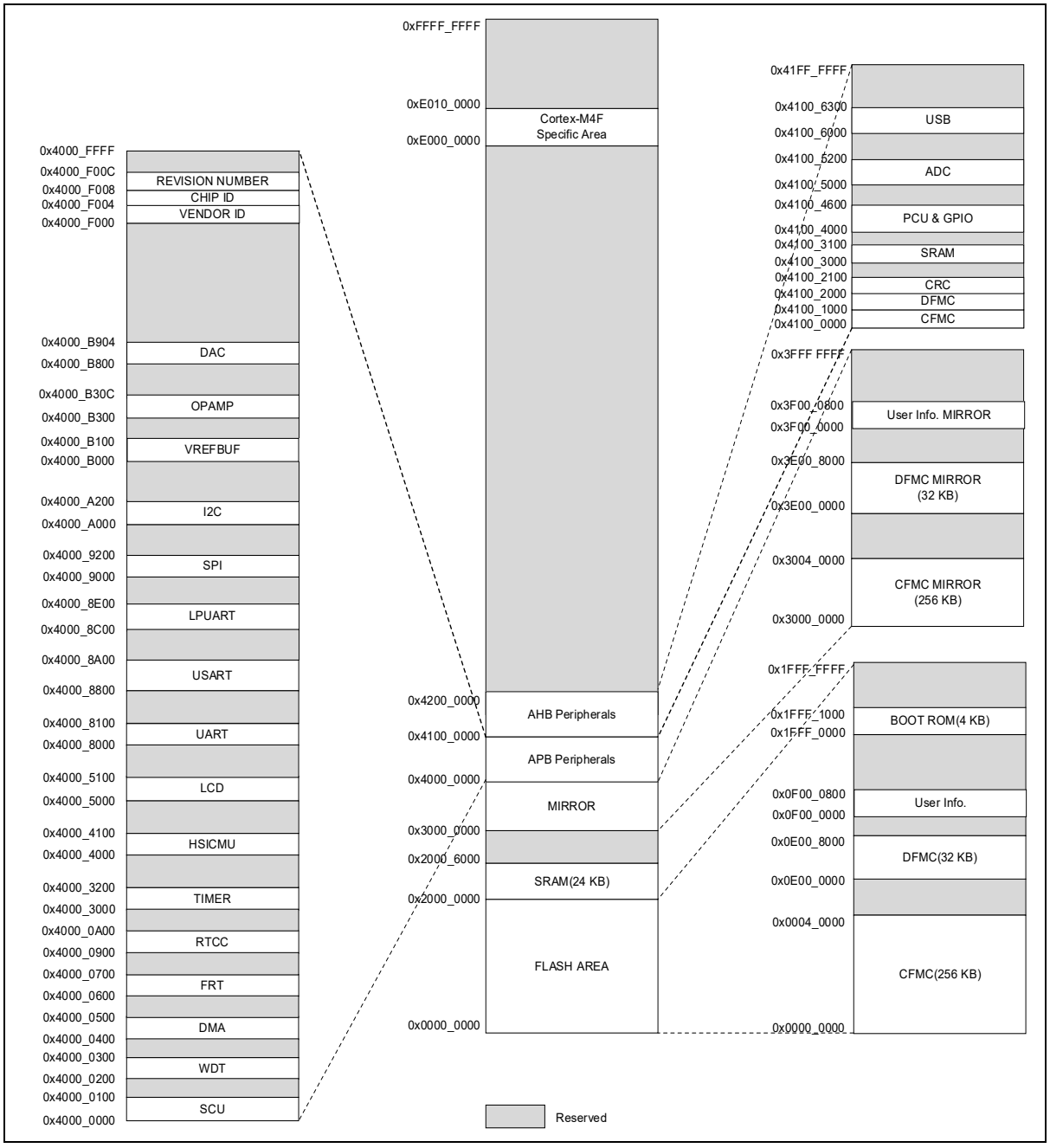
3.2 Memory Organization

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 gives the boundary addresses of the peripherals available in the devices.

Table 5. Peripheral Memory Map

Base Address	Peripheral Name
0x4000_0000	SCU
0x4000_0200	WDT
0x4000_0400	DMA 0/1/2/3/4/5/6
0x4000_0600	FRT
0x4000_0900	RTCC
0x4000_3000	TIMER 0/1/2/3/4/5/6/7
0x4000_4000	HSICMU
0x4000_5000	LCD
0x4000_8000	UART
0x4000_8800	USART 0/1
0x4000_8C00	LPUART 0/1
0x4000_9000	SPI 0/1
0x4000_A000	I2C 0/1
0x4000_B000	VREFBUF
0x4000_B300	OPAMP 0/1
0x4000_B800	DAC 0/1
0x4000_F000	VENDOR ID
0x4000_F004	CHIP ID
0x4000_F00B	REVISION NUMBER
0x4100_0000	CFMC
0x4100_1000	DFMC
0x4100_2000	CRC
0x4100_3000	Internal SRAM
0x4100_4000	PCU and GPIO
0x4100_5000	ADC 0/1
0x4100_6000	USB

3.3 Memory Map

3.3.1 Embedded SRAM

The A34L716 series has a block of 0-wait on-chip SRAM. The size of the SRAM is 24 KB and its base address is 0x2000_0000.

The SRAM memory area is commonly utilized for storing data and as a stack memory. Additionally, it can also be utilized for storing program code for faster execution or during flash erase/programming operations.

3.3.2 Flash Memory Overview

The A34L716 provides internal 256 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.

CPU can access the flash memory with one wait state up to 25 MHz bus frequency.

3.4 Boot Configuration

The A34L716 series offers one boot mode for programming internal flash memory. The boot mode can be entered by setting the nBOOT pin to ‘L’ during the system reset process (the normal state is ‘H’).

The boot mode supports the UART boot which uses the UART interface to transmit and receive boot data (TXD0/RXD0).

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	PC12	Input	Receive the reset signal.
	nBOOT	PC13	Input	Set the boot mode.
UART0	RXD0	PD0	Input	Receive the UART boot data.
	TXD0	PD1	Output	Transmit the UART boot data.

4. SCU (System Control Unit)

4.1 SCU Introduction

A34L716 has a built-in intelligent power control block, which maintains optimized system performance and power dissipation. This block controls reset, clock, operating modes, and analog part. It is called System Control Unit (SCU).

4.2 SCU Main Features

The clock features of A34L716 series are as follows:

- Operating frequency: Up to 32 MHz
- High speed internal oscillator (HSI): 32 MHz
- Low speed internal oscillator (LSI): 40 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 48 MHz)
- A34L716 has clock monitoring as a system fail-safe function.

The reset features of the A34L716 series are as follows:

- nRESET Pin Reset
- CPU Reset
- Software Reset
- Power-On Reset (POR)
- Low-Voltage Detect Reset (LVR)
- Watchdog Timer Reset (WDTR)
- Reset due to clock oscillating error.

Operating mode features of A34L716 series are as follows:

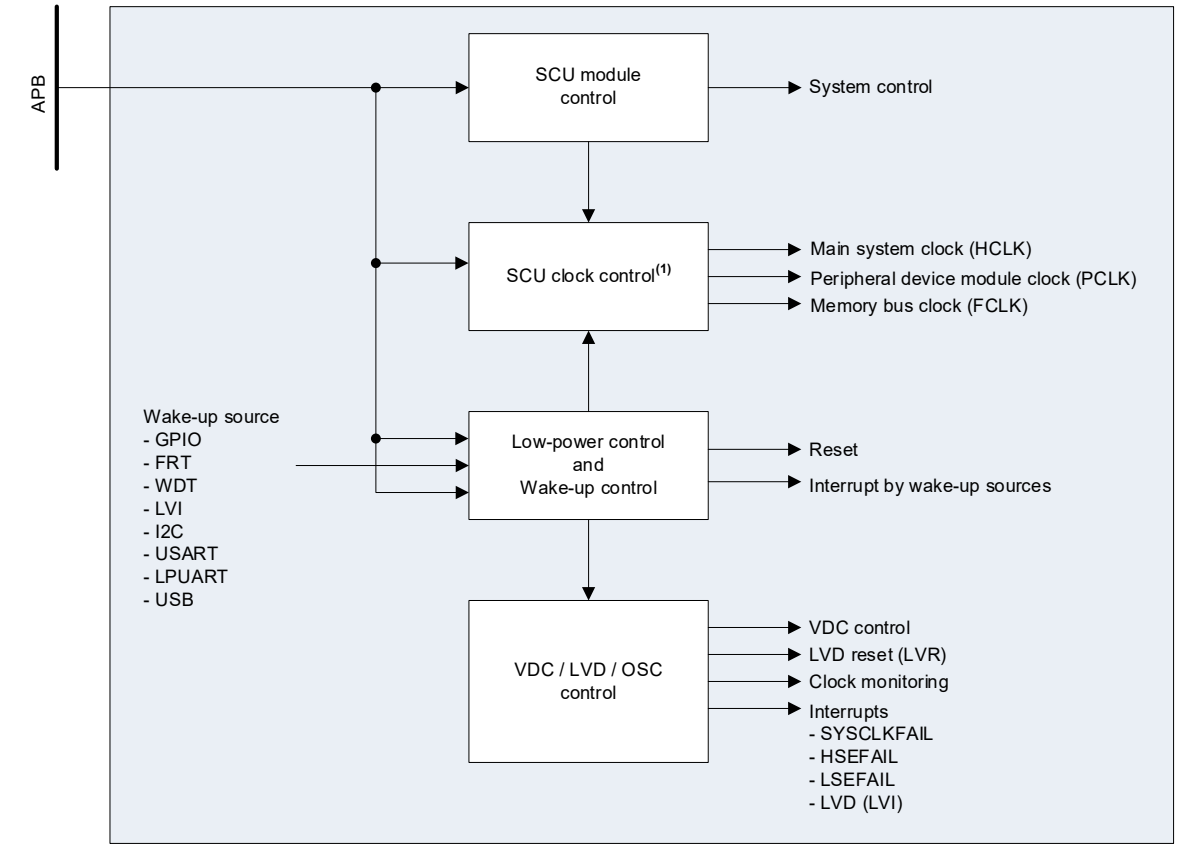
- RUN mode
- SLEEP mode
- DEEP-SLEEP (STOP1, STOP2) mode

4.3 SCU Functional Description

4.3.1 SCU Block Diagram

Figure 8 describes the SCU in a block diagram.

Figure 8. SCU Block Diagram



NOTE:

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

4.3.2 SCU Pin and Type

Table 7 describes the six pins 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 A34L716 has a clock generation circuit. The clock generation circuit consists of external oscillation circuits, PLL circuit, internal oscillation circuits. The overview of each circuit is as follows:

- External oscillation circuit
  - High-Speed External Oscillator (HSE): 4 MHz to 16 MHz (It is recommended to use 8 MHz)
    - Generates main clocks by connecting oscillator to XIN pin and XOUT pin.
    - It is recommended to us 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 48 MHz)
- Internal oscillation circuit
  - High speed internal oscillator (HSI): 32 MHz(1)
  - Low speed internal oscillator (LSI): 40 kHz(1)

NOTE:

1. After the reset is released, the 4 MHz HSI signal, which is derived from the 32 MHz HSI signal divided by 8, and a 40 kHz LSI signal are automatically activated as the internal oscillator.

Table 8 describes the clock sources of the A34L716.

Table 8. Clock Sources

Clock Name	Frequency	Description
HSE	4 to 16 MHz	High-speed external oscillator
PLL	Up to 48 MHz	Phase-locked loop
LSE	32.768 kHz	Low-speed external oscillator
HSI	32 MHz	High-speed internal oscillator
LSI	40 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's 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	O	O	O	O	O	O
WDT	O		O	O	O	O	O	O
RTCC	N/A	SCU_MCCR2	O	O	O	O	O	O
LCD(4)	N/A		O	O	O	O	O	O
TIMERn (n = 0 to 7)	O	SCU_MCCR3	O	O	O	O	O	O
LPUART	N/A	SCU_MCCR4	O	O	O	O	O	O
PGAD(1)	N/A		O	O	O	O	O	O
PGBD(2)	N/A	SCU_MCCR5	O	O	O	O	O	O
PGCD(3)	N/A		O	O	O	O	O	O
FRT	N/A	SCU_MCCR6	O	O	O	O	O	O
USB	N/A		O	O	O	O	O	O
OPAMP(5)	N/A	SCU_MCCR7	O	O	O	O	O	O
UART	N/A		O	O	O	O	O	O

NOTES:

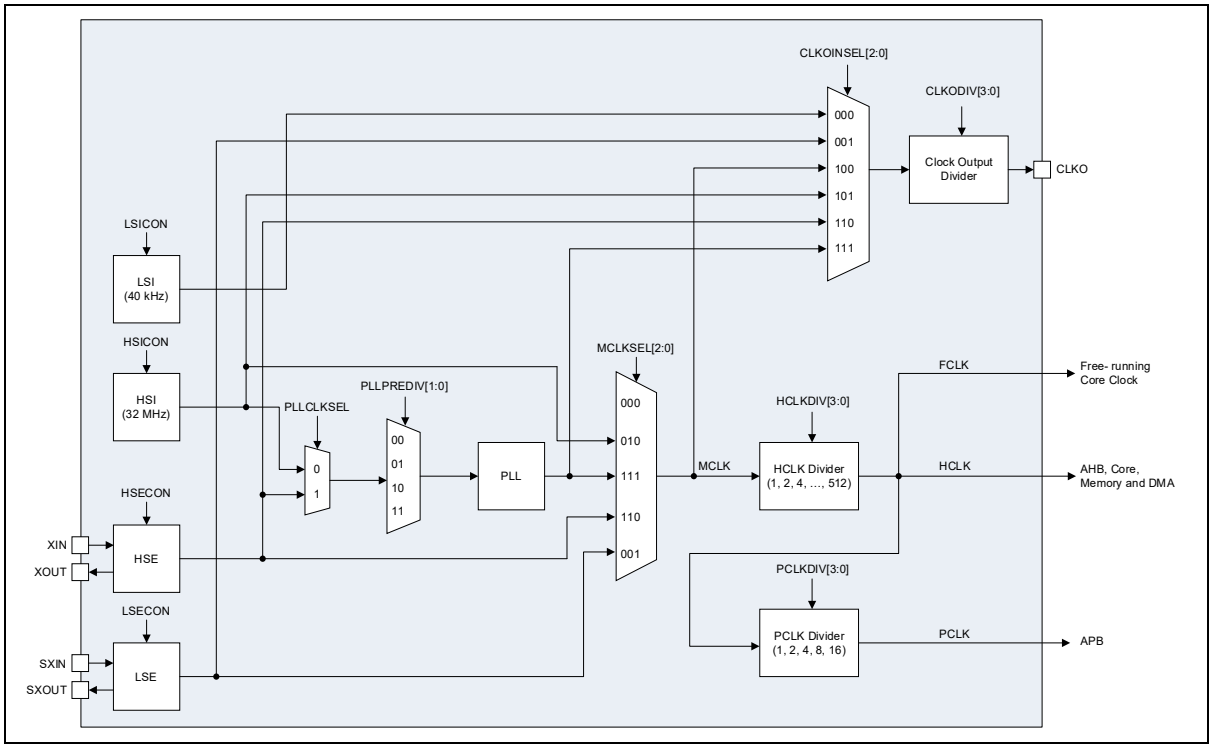
1. PGAD Port Group A Debouncing: PA, PB
2. PGBD Port Group B Debouncing: PC, PD
3. PGCD Port Group C Debouncing: PE, PF
4. LCD must use the LSI and LSE as clock sources.
5. For the OPAMP clock, the LSI and LSE cannot be used as OPAMP clock sources because the input clock requires minimum 50 kHz. When the HSI is selected as a clock source, it is automatically divided by 4 internally, resulting in 8 MHz input. (32 MHz / 4 = 8 MHz)

4.4.2 Clock Tree Configuration

Users can maintain the clock system variation under software control. Figure 9 allows the users to learn about the clock system on the A34L716.

The A34L716 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 9. Clock Configuration



Each mux (selecting a clock source) has a glitch free circuit. So, a clock can be switched on 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 Clocks.

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:

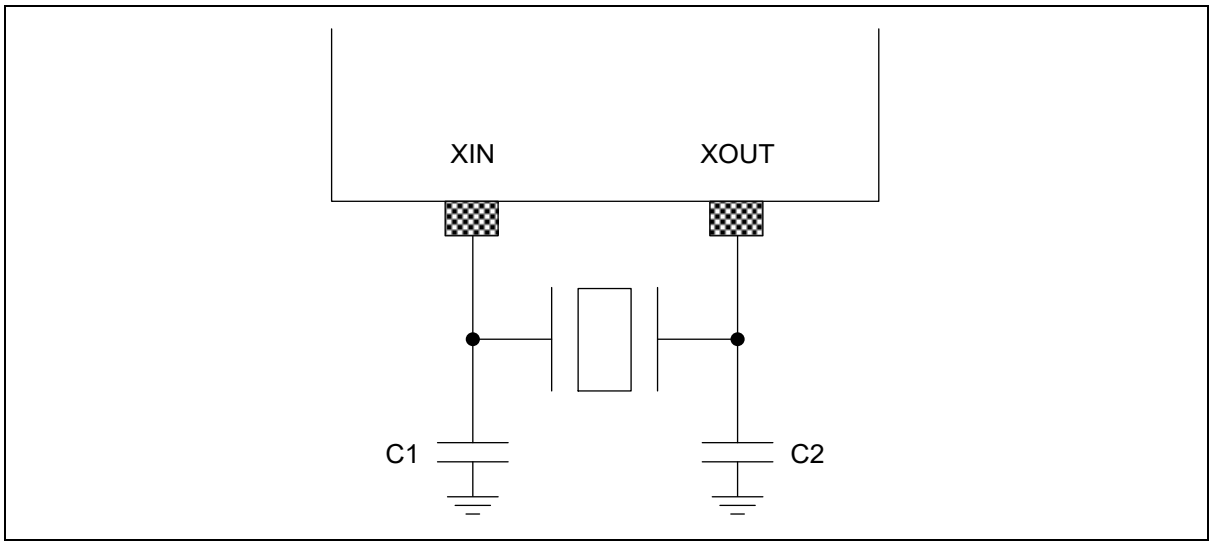
Two pins in Table 11 are assigned for the HSE clock pins block.

- 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 / ceramic oscillator. • For details, refer to Figure 10.
XOUT	Output	

Figure 10. HSE Crystal/Ceramic Oscillator



NOTE:

1. When not using HSE (8 MHz), If the HSECON bit is enabled (SCU_CSCR), it may collide with XOUT (output) and receive internal damage to the microcontroller when used as a GPIO output. When not using an external HSE (8 MHz), enable only the clock source to be used in the SCU_CSCR register.

4.4.4 HSI Clock

The HSI is generated from the internal 32 MHz oscillator.

- HSI: 32 MHz
  - ±0.5% @ −40°C to +85°C (HSICMU ON)
  - ±3.0% @ −10°C to +70°C (HSICMU OFF)
  - ±5.0% @ −20°C to +85°C (HSICMU OFF)

4.4.5 PLL Clock

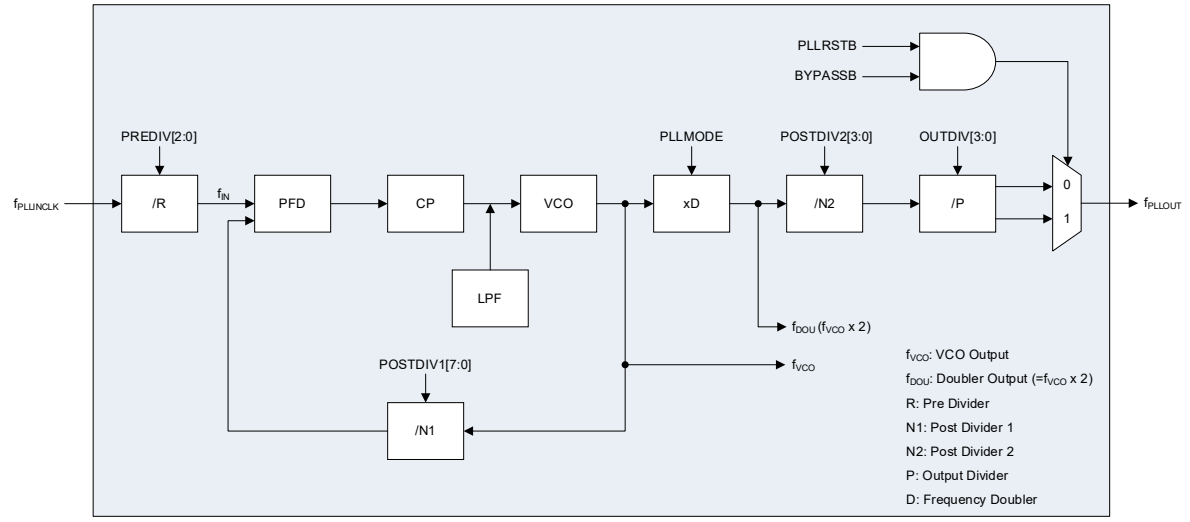
Phase-locked Loop (PLL) can be used to multiply the HSI or HSE output clock frequencies.

The PLL frequency generator generates a high-speed clock (up to 48 MHz). For more information on the PLL frequency generator, refer to Figure 9 and section 4.8.22.

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 11. PLL Block Diagram



NOTES:

1. PLLEN ='1' → min. 1 us Delay → PLLRSTB = '1' to enable PLL block & set PLL Frequency.
2. Wait 500 μs for the stabilization of PLL output.
3. Wait until PLLLOCK (= LOCKSTS) flag occurs.
4. PLLMODE (D) must be '0'.
5. (POSTDIV2 (N_2) and OUTDIV (P)) ≥1 or (POSTDIV2 (N_2) or OUTDIV (P) ≥ 3)
6. The output calculation formula is as follows:

$$f_{\mathrm{PLLOUT}} = \frac{f_{\mathrm{PLLINCLK}} \times (\mathrm{N}_1 + 1)}{(\mathrm{R} + 1) \times (\mathrm{N}_2 + 1) \times (\mathrm{P} + 1)} * (\mathrm{D} + 1)$$

Register	Bit	Symbol	Description
SCU_PLLCON	PREDIV[2:0]	R	PLLINCLK pre-divider value
	POSTDIV1[7:0]	N_1	Post multiplier value
	POSTDIV2[3:0]	N_2	Post divider value
	OUTDIV[3:0]	P	Output divider value
	PLLMODE	D	Frequency doubler value

4.4.5.1 PLL Output Frequency Settings

For the A34L716, 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 1 MHz to 3 MHz is supported, but it is recommended to set the f_{IN} to 2 MHz if possible.

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

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:

$$f_{\mathrm{PLLOUT}}\ [\mathrm{MHz}] = \frac{f_{\mathrm{PLLINCLK}} \times (\mathrm{N}_1 + 1)}{(\mathrm{R}+1) \times (\mathrm{N}_2 + 1) \times (\mathrm{P}+1)} \times (\mathrm{D}+1)$$

$$= \frac{f_{\mathrm{IN}} \times (\mathrm{N}_1 + 1)}{(\mathrm{N}_2 + 1) \times (\mathrm{P}+1)} \quad \mathrm{when}, \mathrm{D} = 0\ (D\ \ must\ be\ \ '0'.)$$

$$(N_2\ and\ P \ge 1)\ or\ (N_2\ or\ P \ge 3)$$

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. Set flash wait to '1'.
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 '1', and the PCLK to the HCLK.
10. Change the system clock source to the PLL.
11. Set flash wait to '1'.

4.4.6 LSE Clock

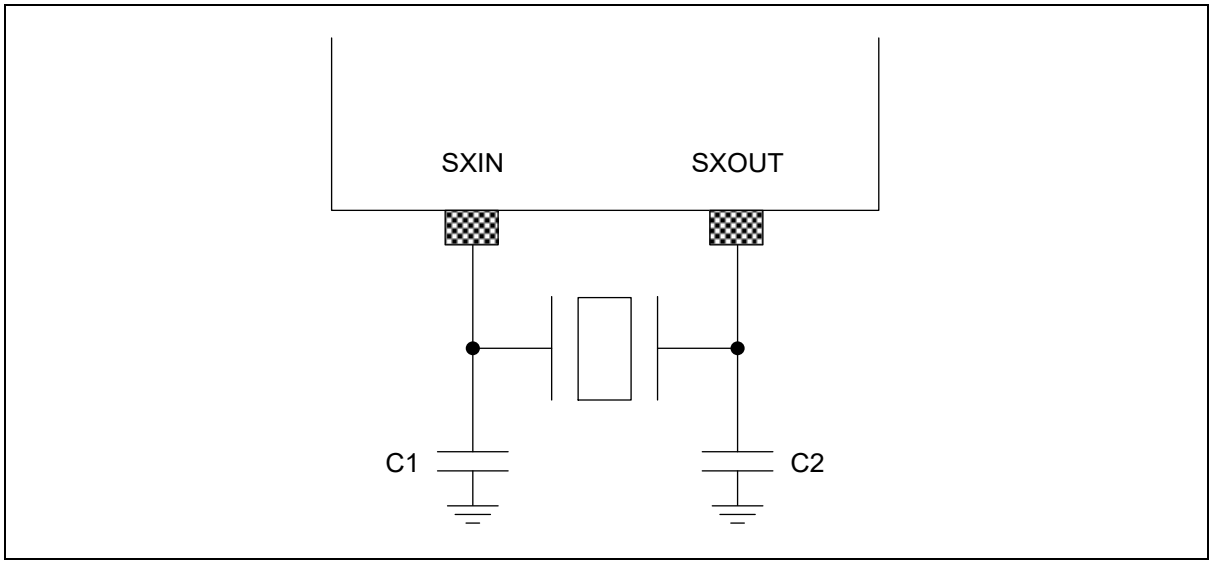
The Low-Speed External Oscillator (LSE) crystal is a 32.768 kHz crystal or ceramic resonator. It has the advantage of providing a low power but highly accurate clock source to the clock peripheral for clock / calendar or other timing functions.

- 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 12. LSE External Clock



NOTES:

1. When not using LSE (32.768 kHz), If the LSECON bit is enabled (SCU_CSCR), if it is used as GPIO output or other functions, it may collide with SXOUT (output) and receive internal damage to the microcontroller. If external LSE (32.768 kHz) is not used, enable only the clock source to be used in the SCU_CSCR register.
2. Before using the LSE (32.768 kHz) clock, users must set the EOSCR register. (See section 4.8.27.)

4.4.6.1 PCB Design Considerations

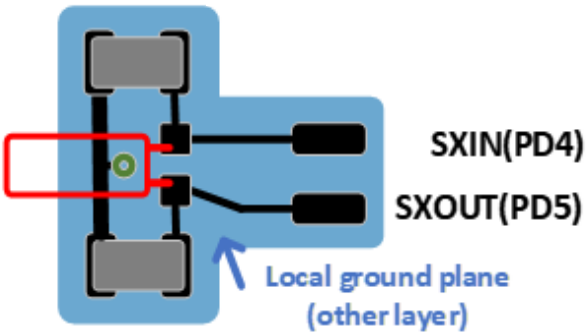
The A34L716 LSE oscillator is designed for low-power consumption.

Hence, the current flowing between the A34L716 microcontroller pins, the crystal and, if used, the external capacitors are extremely low. Long signal lines make the oscillator very sensitive to crosstalk. Even the best components cannot solve problems caused by a poor layout.

- For optimal performance of the crystal oscillator, an analog circuit, it's crucial to adhere to these straightforward guidelines when configuring the circuit on the board:
- Traces between the A34L716 microcontroller and the crystal, the trace between the crystal and the capacitor, and the trace between the two capacitors should be as short as possible. This minimizes excess electrical capacitance and reduces the likelihood of signal interference. Consider the capacitance of these connections when determining capacitor sizes.
- A ground area must be placed below the crystal oscillator area.
- Maintain ample distance between the crystal connections and any other digital signals, especially those involved in timing or frequent switching. Signals from these sources can disrupt the delicate oscillator signal.
- Ensure a significant separation between the XIN and XOUT signals to minimize excess electrical capacitance.

Examples for recommended layouts are shown in Figure 13.

Figure 13. LSE PCB Design Considerations



4.4.7 LSI Clock

After the microcontroller is powered on, the LSI (40 kHz) is initially enabled by default in the system operation sequence.

For more details, refer to the electrical characteristics chapter in A34L716.

- LSI: 40 kHz
  - ±30% @ -40 ºC to +85 ºC

4.4.8 Two Main Operating Clocks: HCLK and PCLK

A34L716 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 CPU 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 32MHz, the HCLK frequency must not exceed 32 MHz.

4.4.8.2 PCLK Clock

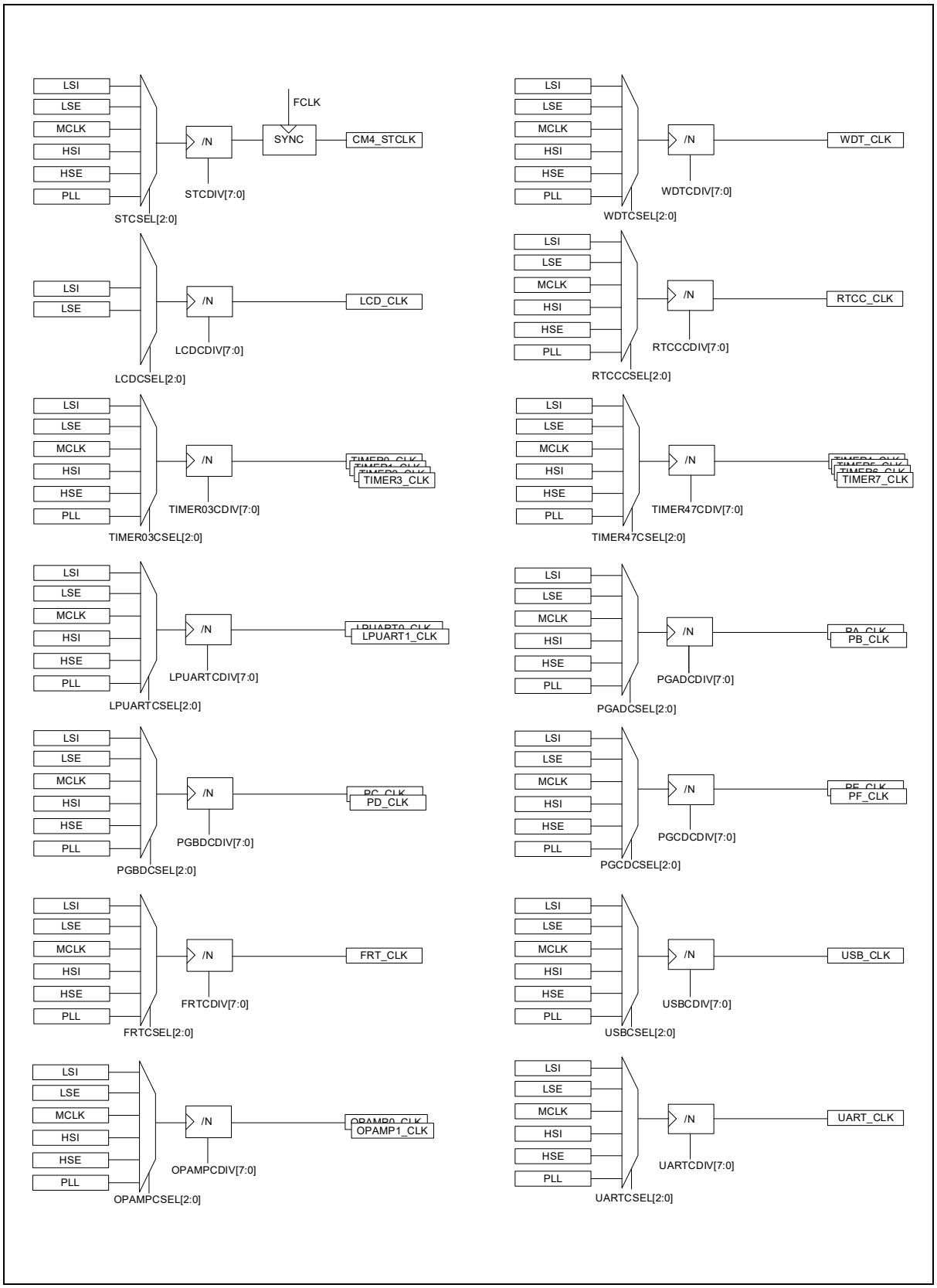
The PCLK can be used as a clock for any peripherals. Whether to enable or disable the PCLK for each peripheral is determined with the SCU_PCER registers; each peripheral block’s registers cannot be read unless its PCLK input is enabled. And the PCLK stops operating in DEEP-SLEEP mode.

4.4.9 Configuration of Miscellaneous Clocks

The A34L716 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 9 describes the peripheral’s clock and the register selection. (Refer to Table 9 for the information of peripheral’s clock and the register selection.)

4.4.11 Clock Monitoring

The A34L716 provides clock monitoring functions used to detect whether the microcontroller's main clock source is stopped. Using the clock monitoring function, if the microcontroller's main clock source is detected to be stopped, it is possible to automatically switch to the alternate internal clock source, HSI 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 A34L716 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.

NOTE:

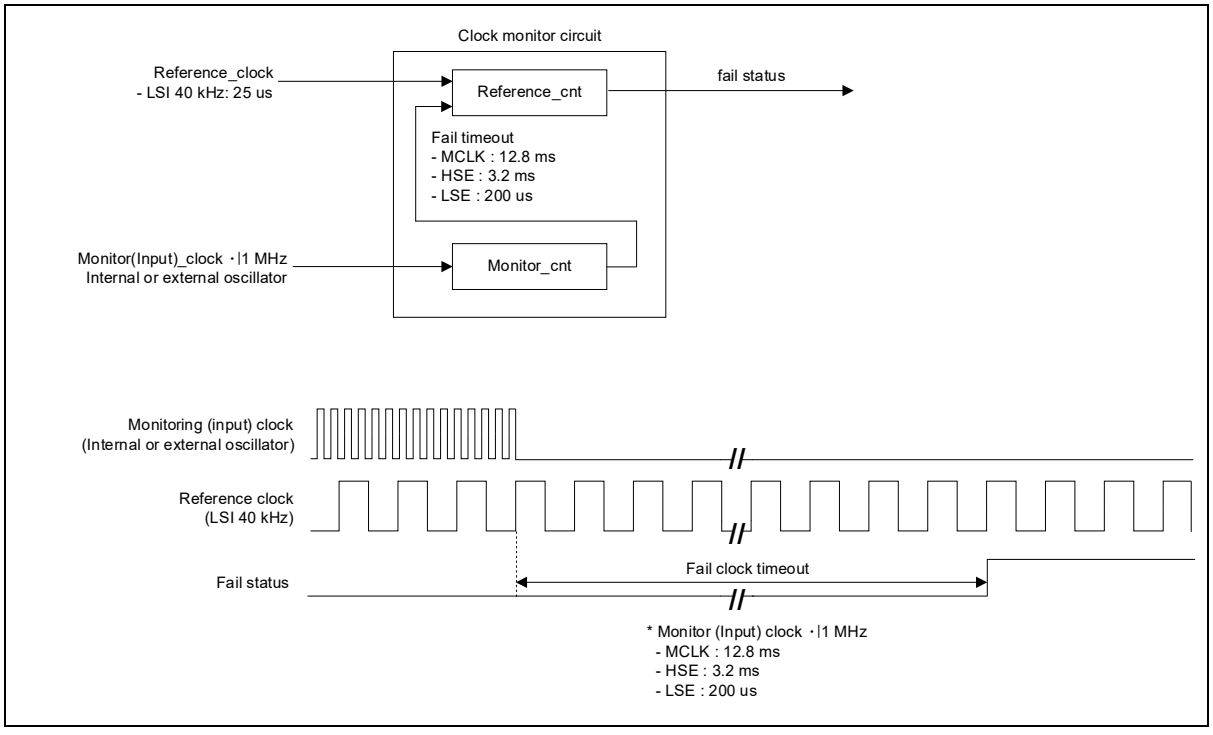
1. The interrupt function of HSE and LSE clock monitoring is available only when HSE and LSE are not used as MCLK.

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 13 shows clock monitoring functions provided by the A34L716.

Table 13. A34L716 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 14 shows clock sources and their frequency characteristics that the A34L716 can monitor. After each clock source is enabled, stabilization delay is required for stable frequency oscillation.

Table 14. Clock Source for Clock Monitoring

Clock Source	Frequency	Stabilization Delay
HSE	4 to 16 MHz	10 ms
PLL	up to 48 MHz	500 µs
LSE	32.768 kHz	1 s
LSI	40 kHz (±30%)	250 µs
HSI	32 MHz (±1.5%)	30 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 the microcontroller's internal oscillation circuit.

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

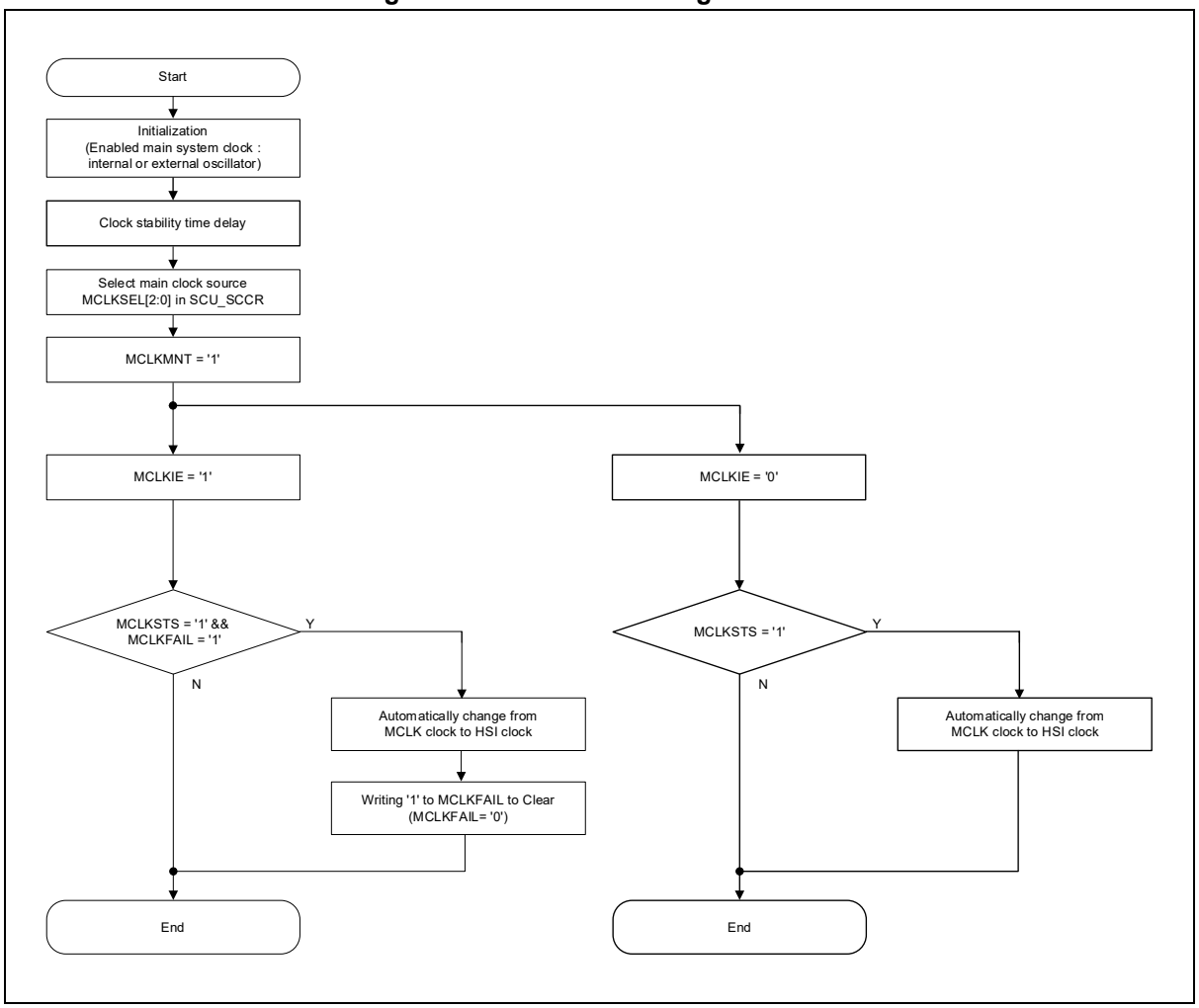
1. Set the HSECON and HSICON in SCU_CSCR register to enable a clock source set to MCLK and wait for a stable time. (HSE: 10 ms, PLL: 500 μs, HSI: 30 μs)
2. Select the main system clock source by configuring the MCLKSEL[2:0] in SCU_SCCR register. (1)(2)(3)
3. Set the MCLKMNT in SCU_CMR register to '1' to activate the MCLK monitoring function. MCLK errors can be detected by the clock monitoring circuit operating internally at LSI 40 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 in SCU_RSER register to '1' to perform the microcontroller's reset when the MCLK error flag occurs.
8. After the reset event occurs, the MCLKFRST flag in 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 MCLKSEL, 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 PLLCLKSEL, 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 shows a workflow in which the clock monitoring circuit on the A34L716 monitors the MCLK.

Figure 16. MCLK Monitoring Workflow



4.4.11.4 HSE Monitoring

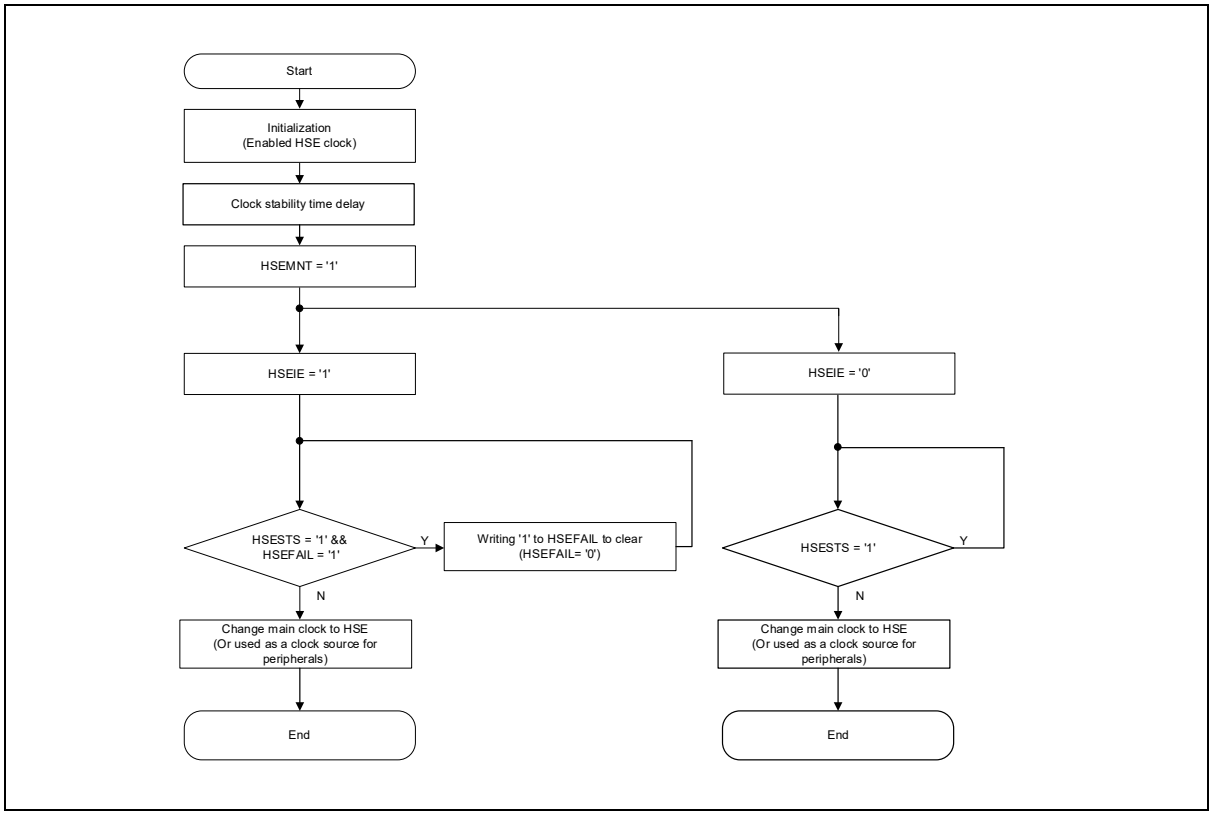
This section describes the HSE clock monitoring. For information on the HSE clock, see section 4.4.3.

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

1. Set the HSECON in 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 in 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 40 kHz.
3. To use the error detection interrupt function, set the HSEIE in 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 SCU_CMR register. The HSEFAIL flag is cleared by writing ‘1’.
4. Set the HSEFRST in SCU_RSER register to ‘1’ to perform the microcontroller’s reset event when the HSE clock error occurs.
5. After the reset event occurs, the HSEFRST flag in SCU_RSSR register informs whether the reset event occurs in the microcontroller due to the HSE error. The HSEFRST flag is cleared by writing ‘1’.

Figure 17 shows a workflow in which the clock monitoring circuit on the A34L716 monitors the HSE.

Figure 17. HSE Monitoring Workflow



4.4.11.5 LSE Monitoring

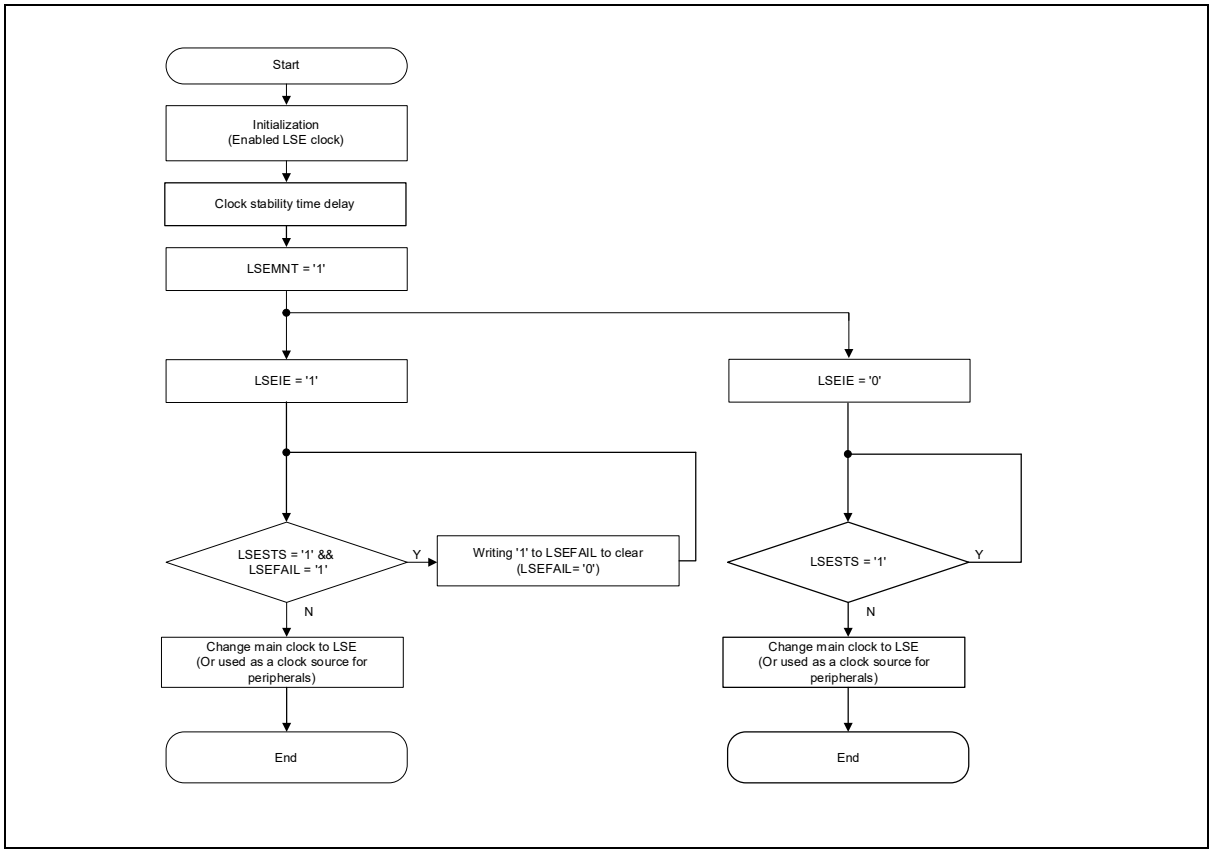
This section describes the LSE clock monitoring. For information on the LSE clock, see section 4.4.6.

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

1. Set the LSECON in 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 in 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 40 kHz.
3. To use the error detection interrupt function, set the LSEIE in 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 SCU_CMR register. The LSEFAIL flag is cleared by writing ‘1’.
4. Set the LSEFRST in SCU_RSER register to ‘1’ to perform the microcontroller’s reset event when the LSE clock error occurs.
5. After the reset event occurs, the LSEFRST flag in 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 18 shows a workflow in which the clock monitoring circuit on the A34L716 monitors the LSE.

Figure 18. LSE Monitoring Workflow



4.4.12 Clock Configuration Procedure

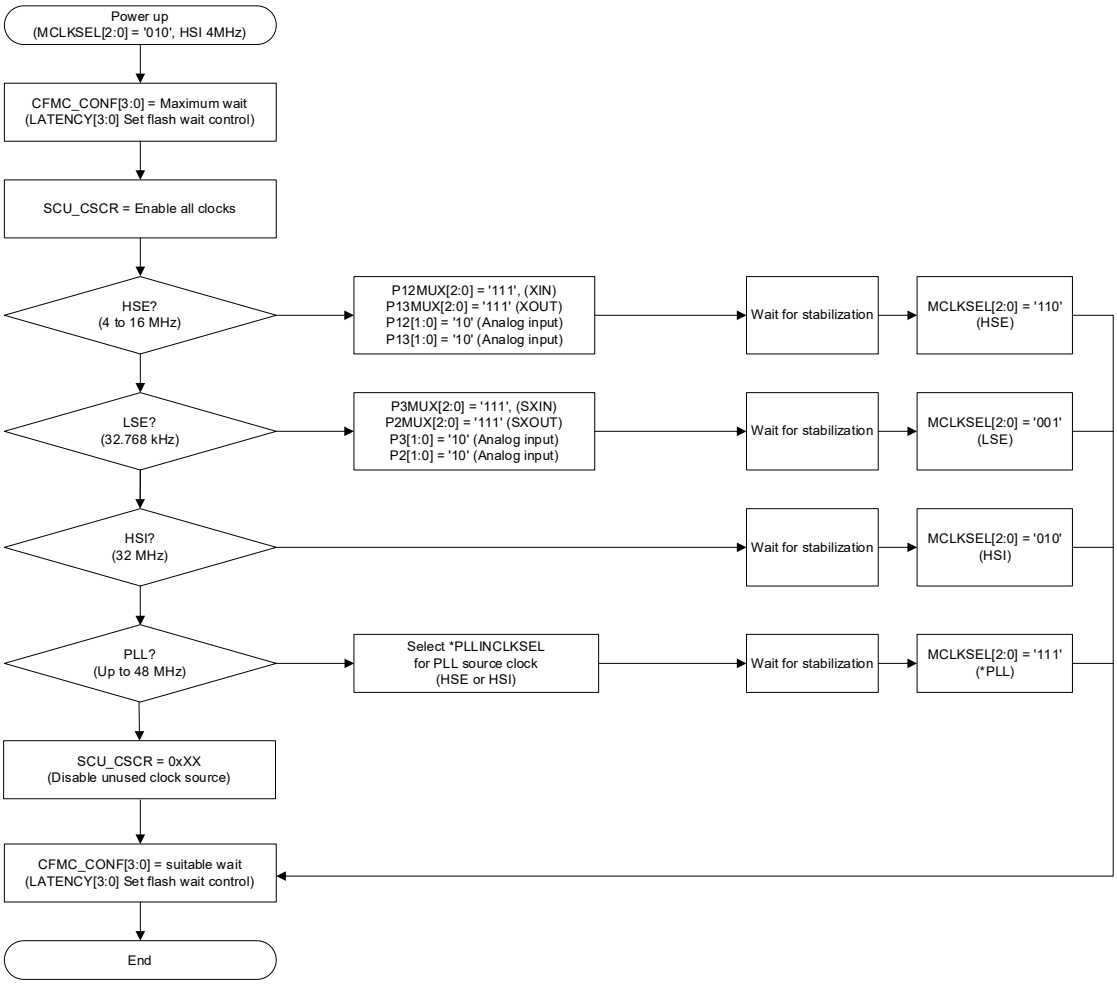
After power on the device, 4 MHz HSI that 32 MHz HSI divided by 8 operates as a default system clock. The HSI is enabled during the power up sequence, and the HSE is configured by user controls with the HSI system clock.

Before enabling the HSE block, the pin mux configuration must be completed for XIN and XOUT. Users must be careful not to affect other bits of the PC_MR and PC_CR registers during this process. 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 19 shows an example sequence of how the system clock changes.

Figure 19. Clock Change Procedure



NOTE:

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

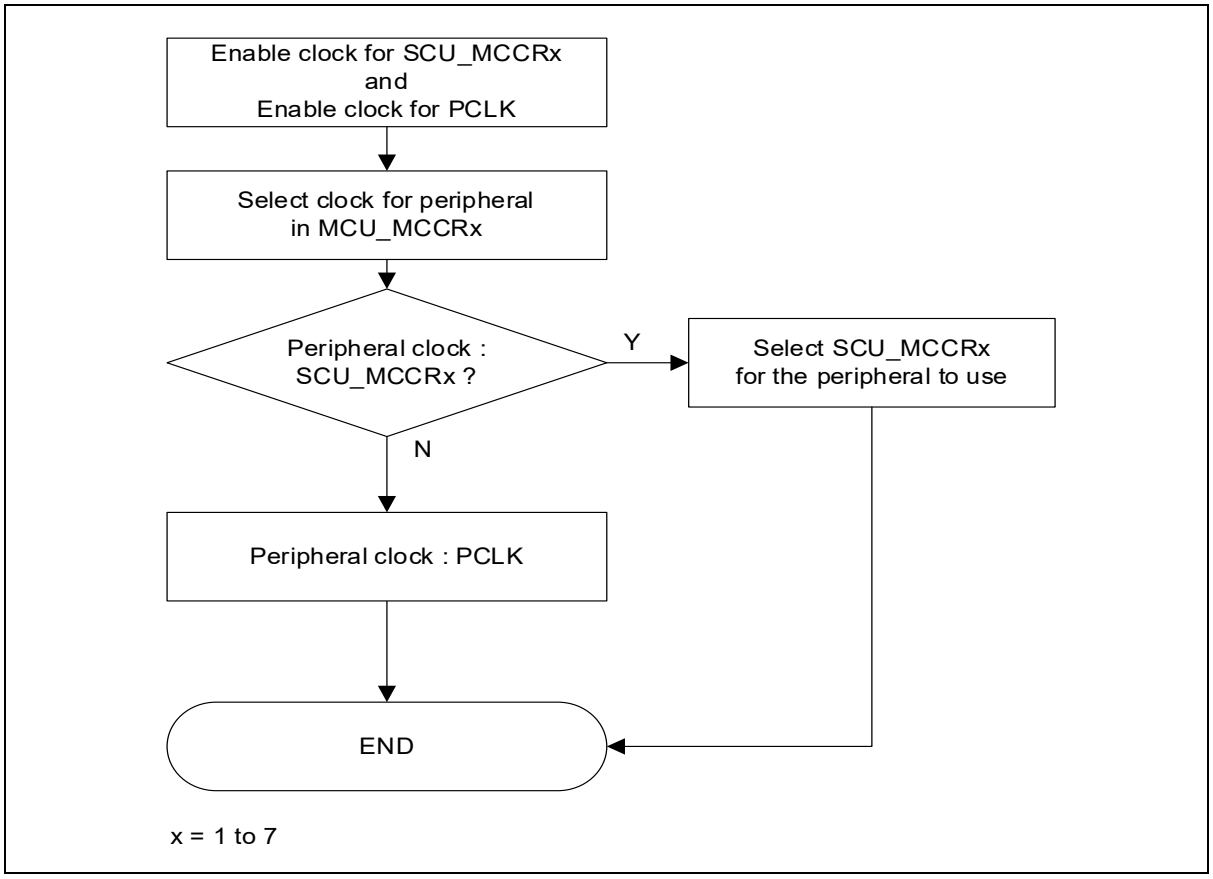
To speed up the system clock to the maximum operating frequency, users must check the flash wait control configuration. flash read access time is one of limitation factors for performance. The wait control recommendation is suggested in Table 15.

Table 15. Flash Wait Control Recommendation

LATENCY[3:0]	Flash Access Wait	Available Max. System Clock Frequency
0000	0-clock wait	Up to 25 MHz
0001	1-clock wait	Up to 32 MHz
0010	2-clock wait	Up to 32 MHz
0011	3-clock wait	Up to 32 MHz
0100	4-clock wait	Up to 32 MHz
0101	5-clock wait	Up to 32 MHz
0110	6-clock wait	Up to 32 MHz
0111	7-clock wait	Up to 32 MHz
1000	8-clock wait	Up to 32 MHz
1001	9-clock wait	Up to 32 MHz
1010	10-clock wait	Up to 32 MHz
1011	11-clock wait	Up to 32 MHz
1100	12-clock wait	Up to 32 MHz
1101	13-clock wait	Up to 32 MHz
1110	14-clock wait	Up to 32 MHz
1111	15-clock wait	Up to 32 MHz

Figure 20 shows how to set the peripheral clock. Typical selection of peripheral clocks is to use SCU_MCCRx (x = 1 to 7) and PCLK.

Figure 20. Peripheral Clock Selection



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

The A34L716 provides two reset options. 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.)

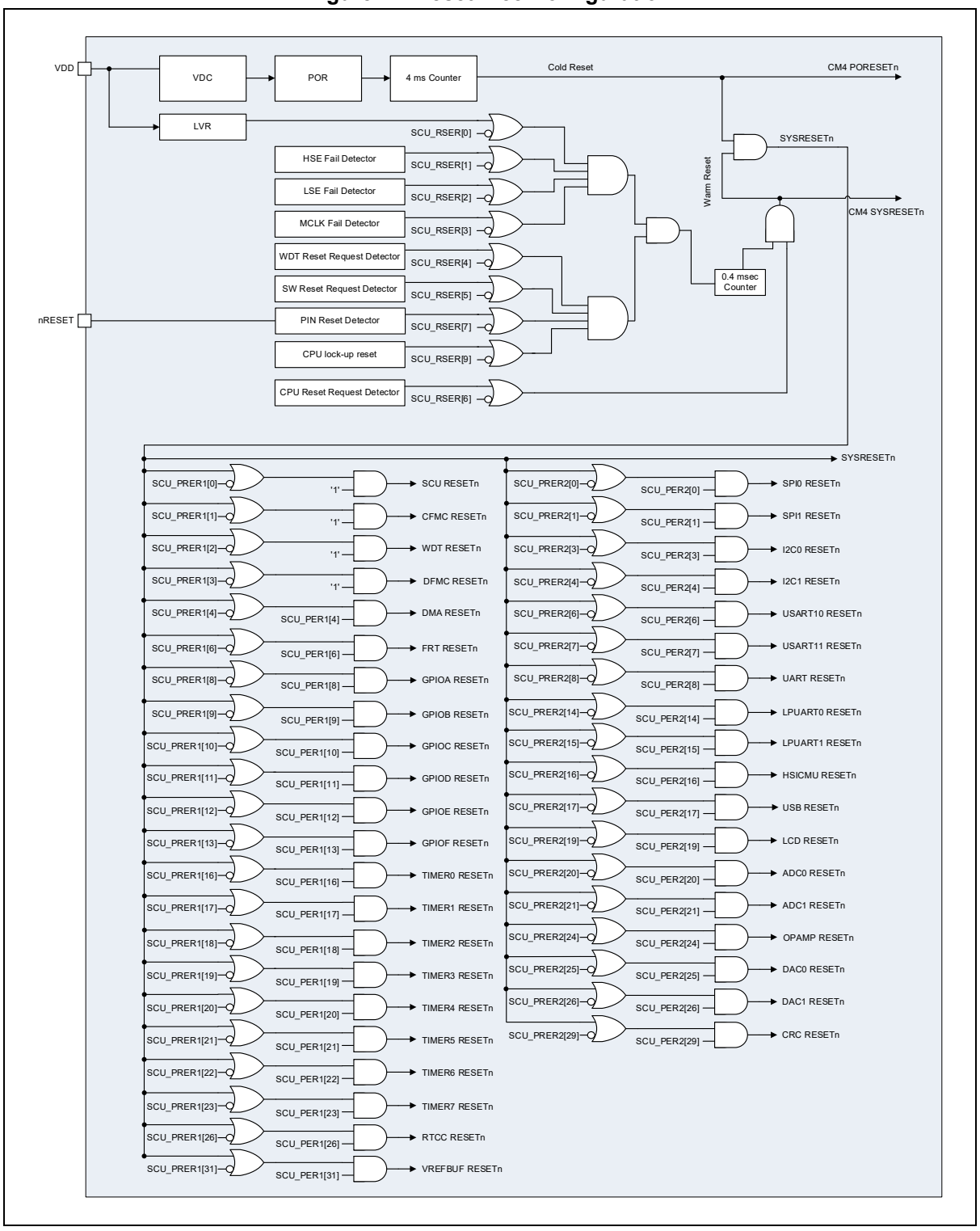
A reset event makes a chip to turn to an initial state. Reset sources of the cold reset and the warm reset are listed in Table 16.

Table 16. 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

4.5.2 Reset Tree Configuration

Figure 21. Reset Tree Configuration



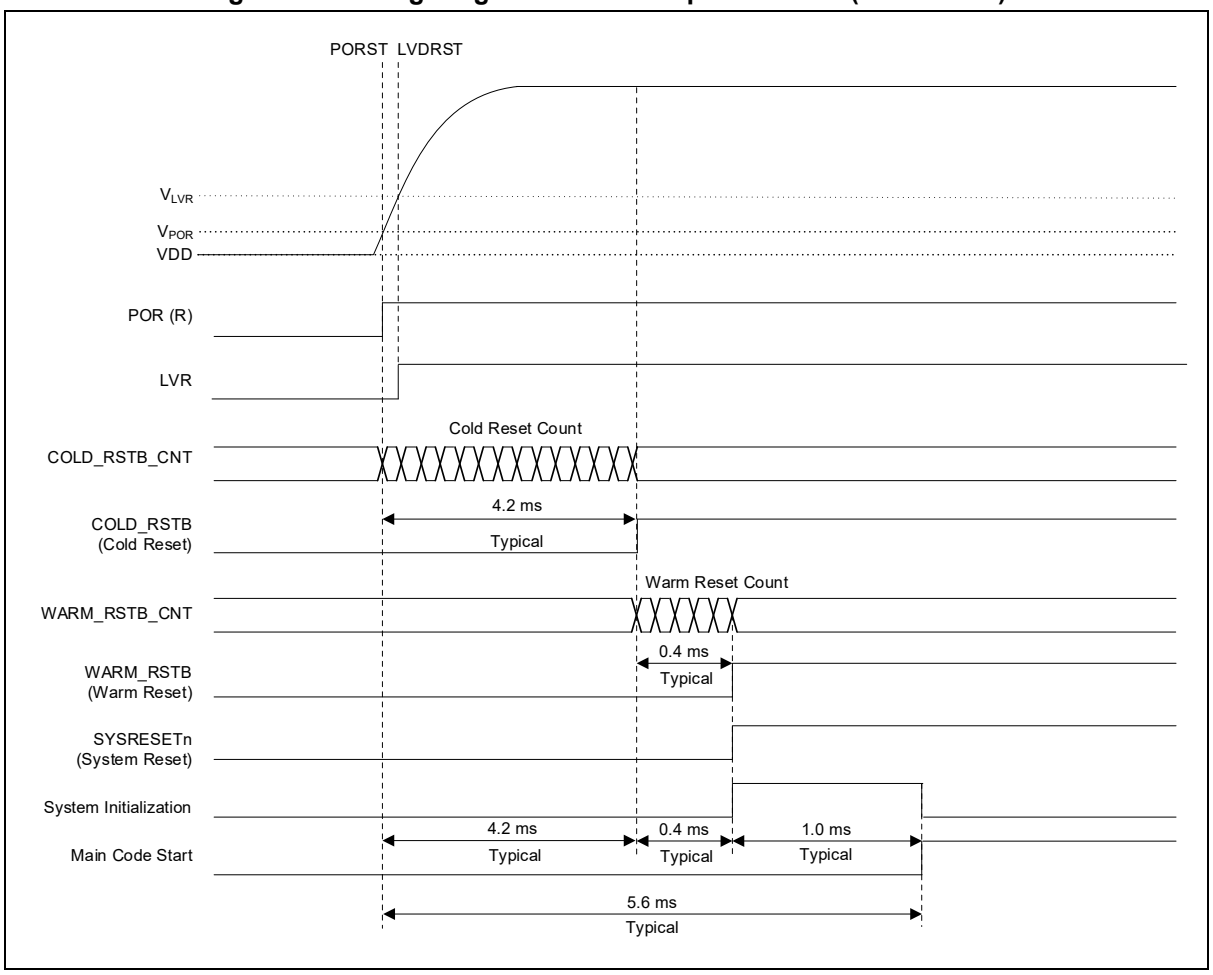
4.5.3 Cold Reset

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 1.4 V based on the amount of the internal VDC output.

During the cold reset process, when the applied voltage exceeds 1.4 V, the HSI clock is enabled. After the stabilization of the internal VDC level for 4.2 msec, the internal logic is initialized. And then, when the external VDD voltage rises above the LVD Reset voltage level (1.69 V), the cold reset is released and, after a waiting time of 0.4 ms for warm reset synchronization, booting begins.

Figure 22 shows the waveform of power up sequence and initial reset.

Figure 22. Timing Diagram of Power-Up Procedure (Cold Reset)



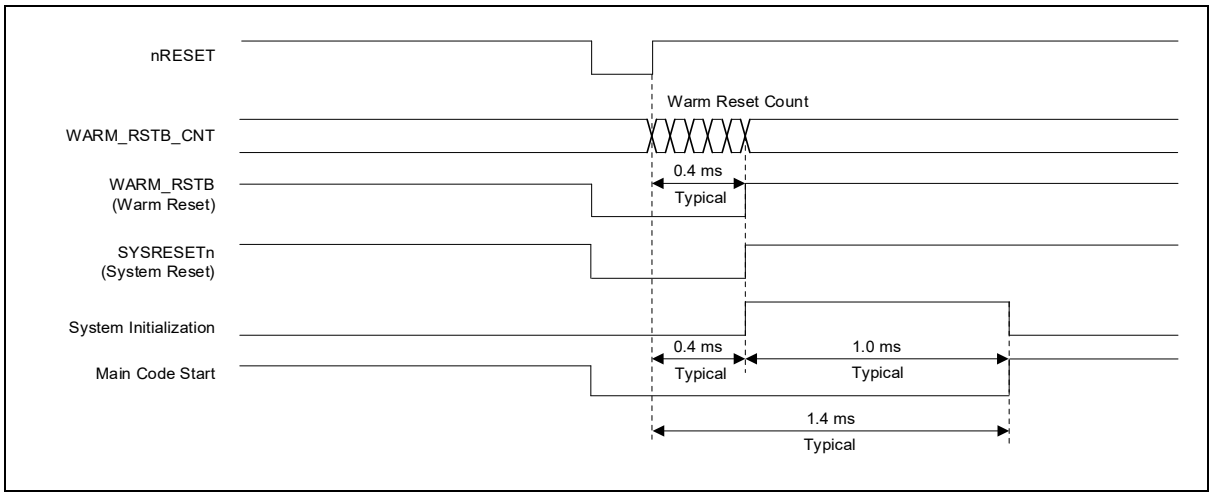
4.5.4 Warm Reset

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, and their occurrence is written to the SCU_RSSR.

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

Figure 23. Warm Reset Timing Diagram



4.5.5 LVD Reset (LVR)

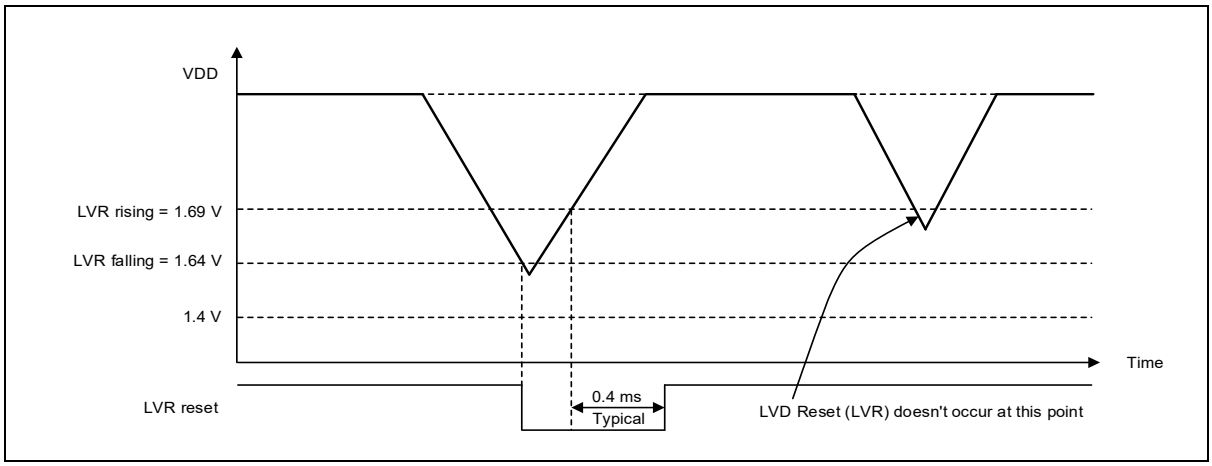
A Low Voltage Detect Reset (LVR) event is triggered when the operating voltage drops below a certain level during the microcontroller's operation. Users can set the microcontroller whether to perform a reset or to perform an interrupt, when an LVR event is triggered.

Table 17 describes the range of LVR level and the register for LVR level control.

Table 17. LVR Level

LVR Level Range	Description
1.69 V to 2.88 V	• LVR level with 5 steps • This LVR level is set by LVRVS[2:0] in SCU_LVRCR.

Figure 24. LVR 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 18 is assigned for RESET.

Table 18. 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 A34L716 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 gets reset.

For details on the watchdog timer reset, refer to chapter 9.

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 SWRST in SCU_SRCR and the registers in the SCU block(1).

NOTE:

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

4.5.9 HSE Fail Reset

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

4.6 Power Supply

The device requires a 1.8 V to 3.6 V operating voltage supply (V_{DD}) and analog peripherals are supplied through the independent power domain AV_{DD}.

Table 19. Power Supply

Parameter	Symbol	Min.	Max.	Unit
Supply voltage	V_{DD}	1.8	3.6	V
	AV_{DD}	1.8	3.6	V

NOTE:

1. AV_{DD} must be supplied at the exact same voltage level as V_{DD} under all conditions. ($AV_{DD} = V_{DD}$)

During a power-up process, a reset plays an important role and affects the entire process of system booting. A34L716 has two power-related reset options as described below:

- POR_RST (Power-On Reset) that controls the voltage less than 1.4 V.
- LVD_RST (Low Voltage Detect Reset) that controls the voltage less than 1.69 V (Default)

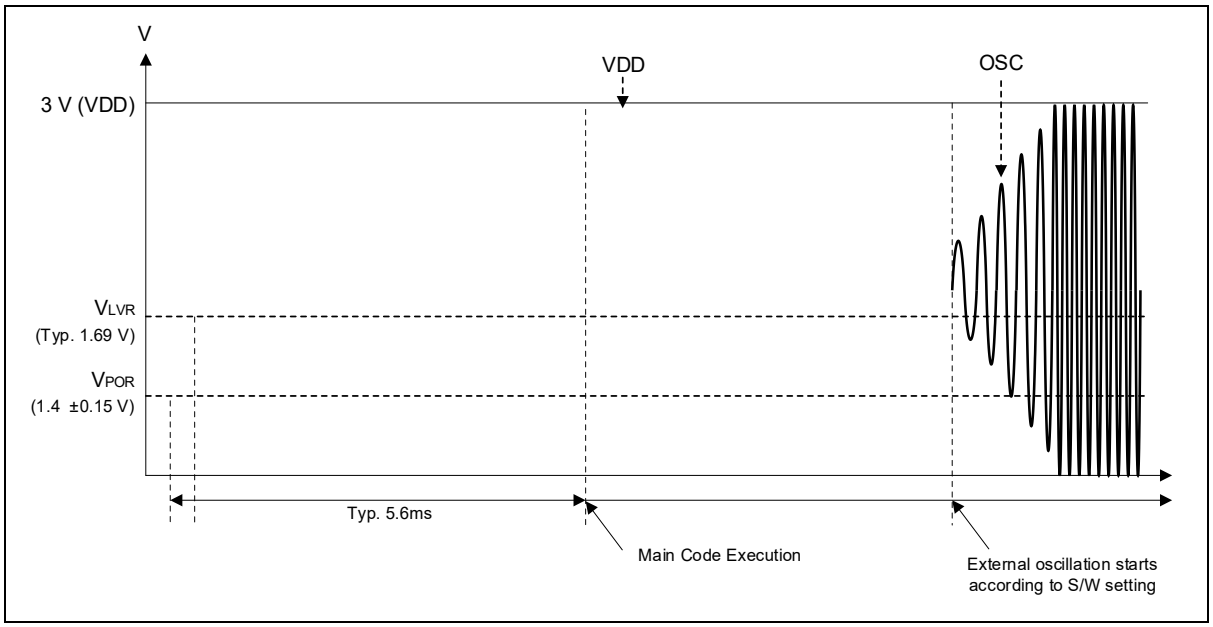
If the power level is higher than the POR and below the flash operating voltage, code read operation may malfunction. To prevent this abnormal code read operation, the LVD_RST operates to generate the SYSRESETn internal signal and enter the microcontroller into reset mode to prevent the abnormal operation.

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 25 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 A34L716, note that external oscillation is set to operate by software setting.

Figure 25. POR and LVR Timing Diagram

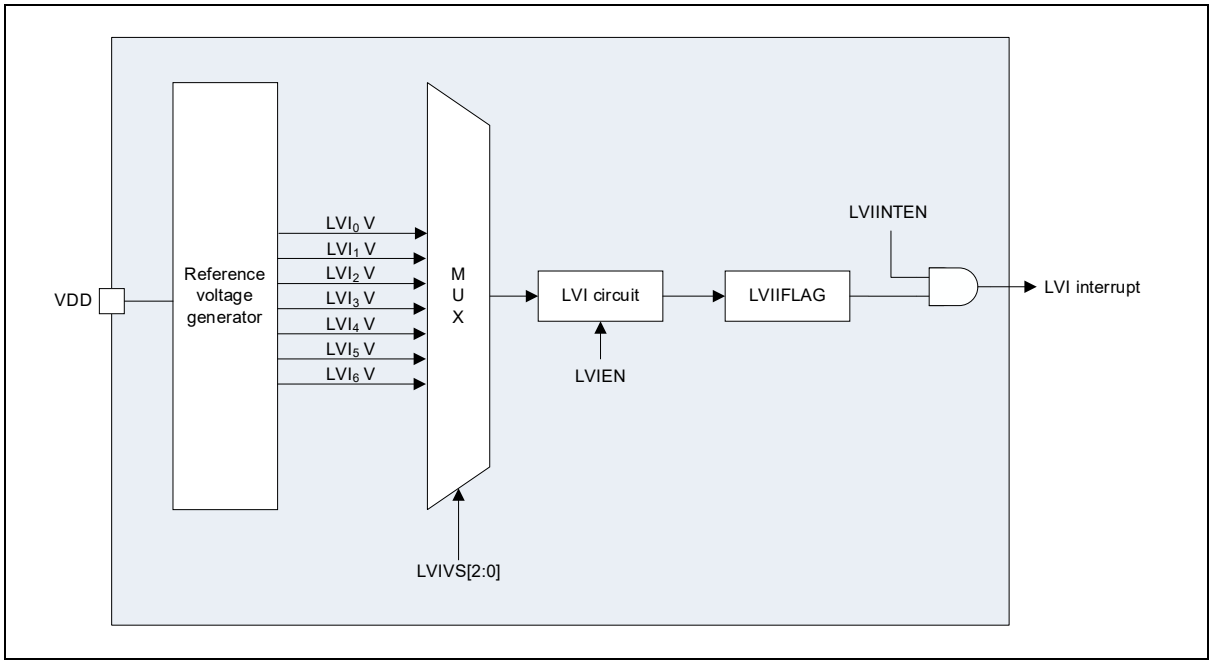


4.6.1 LVI Block Diagram

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

Figure 26 shows a block diagram of the LVI. For information on the LVI signal operation, refer to SCU_LVICR.

Figure 26. LVI Block Diagram

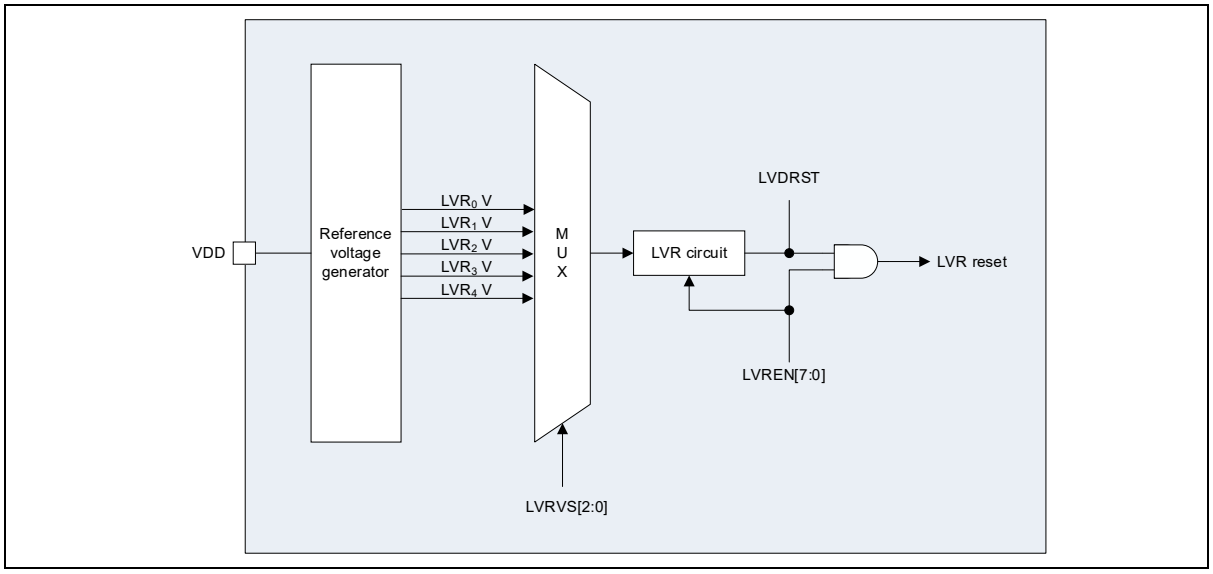


4.6.2 LVR Block Diagram

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

Figure 27 shows a block diagram of the LVR. For information on the LVR signal operation refer to SCU_LVRCR.

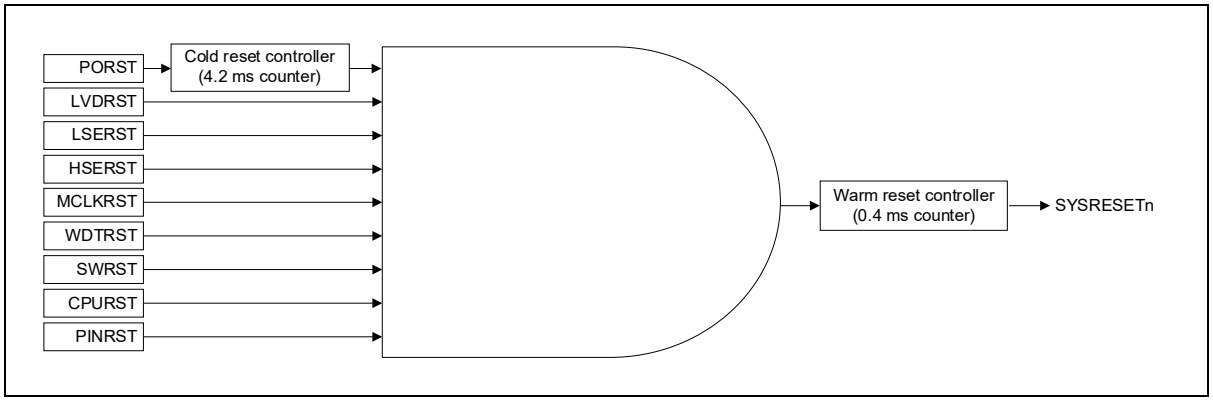
Figure 27. LVR Block Diagram



4.6.3 POR, LVR

For the case where the VDD input level is higher than the POR and lower than the minimum voltage required for the flash operation, the LVDRST (Typ. 1.69 V) is set to improve stability and prevent operation errors such as code read. For more information, refer to section 4.5.

Figure 28. 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 A34L716 operates in three modes below:

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

4.7.2 Transition of Operation Mode

INIT mode is an 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 29 describes transition between the operation modes.

Figure 29. Transition between Operation Modes

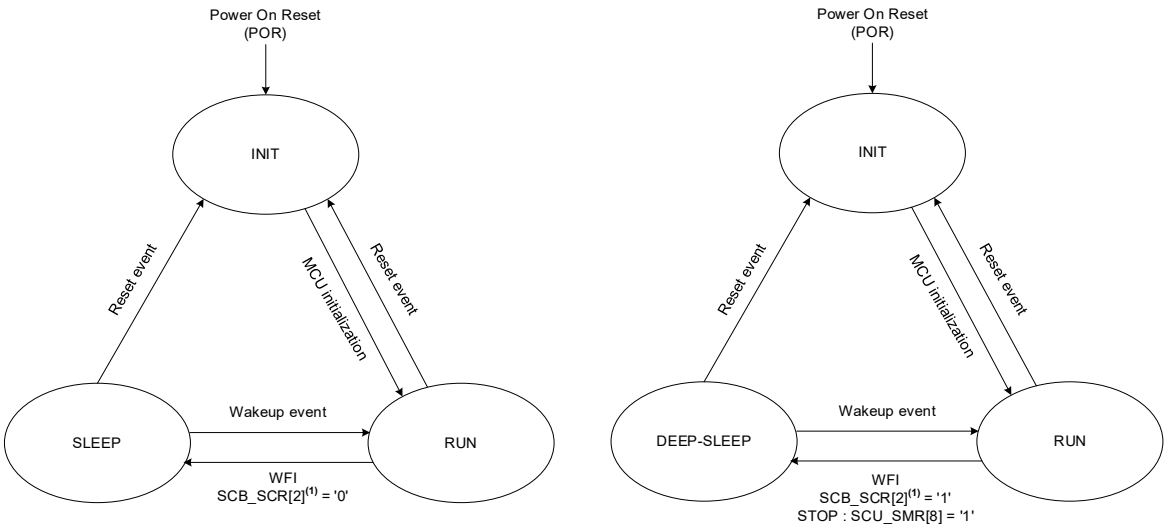


Table 20. Operation Mode

Mode	Condition	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 (STOP1)	WFI (Wait for Interrupt): SCB_SCR[2](1) = 1, SCU_SMR[9] = 1 or SCU_SMR[13] = 1	RUN	INIT
DEEP-SLEEP (STOP2)	WFI (Wait for Interrupt): SCB_SCR[2](1) = 1 SCU_SMR[9] = 0, SCU_SMR[13] = 0	RUN	INIT

NOTE:

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

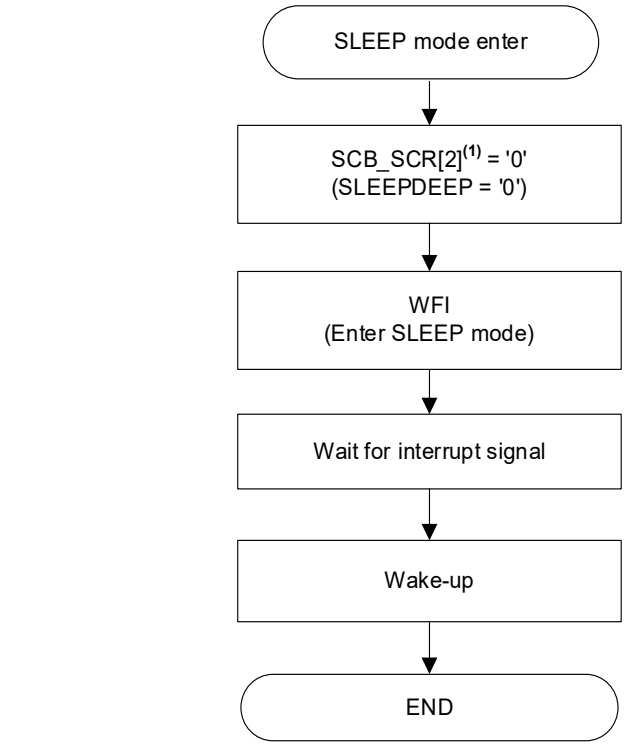
4.7.3 RUN Mode

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, users can determine which peripherals are to be inactive in SLEEP mode.

Figure 30. 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 (STOP1) and DEEP-SLEEP (STOP2) modes.

To enter STOP 1 mode, first set SCU_SMR[9] = 1 or SCU_SMR[13] = 1 and use the WFI command. In STOP 1 mode, all peripherals except RTC of internal VDC domain are in power off state.

To enter STOP 2 mode, first set SCU_SMR[9] = 0 and SCU_SMR[13] = 0 and use the WFI command. In STOP 2 mode, all peripherals of internal VDC domain are stopped. To wake up in STOP1, STOP2 mode, an interrupt request must be generated and can be selected in the SCU_WUER register. Stabilization time is 4ms after wakeup event occurs.

When entering DEEP-SLEEP mode, HSI 4 MHz 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 is used, a function that automatically disables system clocks such as HSE, PLL, HSI, and LSI may be used.

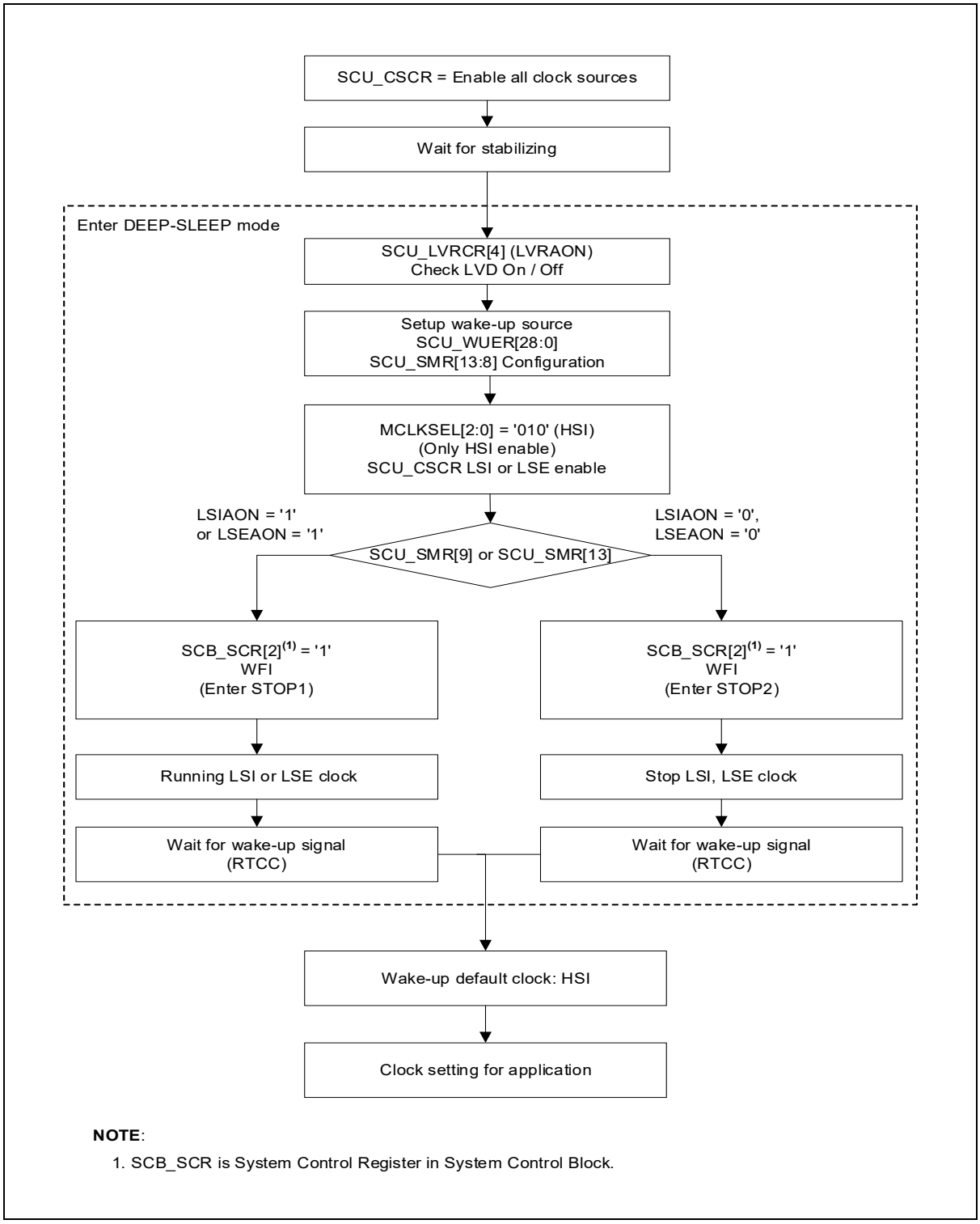
Table 21. DEEP-SLEEP Mode Configuration

Mode	Type	Condition	Wake-up Source
DEEP-SLEEP	STOP 1	WFI SCB_SCR[2](1) = '1' (SLEEPDEEP = '1') SCU_SMR[9] = '1' (LSIAON = '1') or SCU_SMR[13] = '1' (LSEAON = '1')	Source included in WUER
	STOP 2	WFI SCB_SCR[2](1) = '1' (SLEEPDEEP = '1') SCU_SMR[9] = '0' (LSIAON = '0') SCU_SMR[13] = '0' (LSEAON = '0')	Source included in WUER

NOTES:

1. SCB_SCR is System Control Register in System Control Block.
2. When the FRT clock source is set to the MCLK, the microcontroller cannot wake up once it enters DEEP-SLEEP mode. To avoid this, the microcontroller must use SLEEP mode instead of DEEP-SLEEP mode.

Figure 31. DEEP-SLEEP Mode Operation Sequence



NOTE:

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

4.7.6 Operation Mode Summary

Table 22 and Table 23 are configurable clocks according to each operating mode.

Table 22. 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 23. Configurable Clocks in Each DEEP-SLEEP (STOP1, STOP2) Mode

Mode	CPU	Peripheral	SCU_SMR					
			VDCAON	PLLAON	LSIAON	HSIAON	VDCAON	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 RTC	OFF	OFF	User-defined	OFF	OFF	User-defined
DEEP-SLEEP (STOP 2)	Inactive	OFF	OFF	OFF	OFF	OFF	OFF	OFF

4.8 SCU Registers

The base addresses and register map for the CHIPCONFIG are described in the tables below:

Table 24. Base Address of CHIPCONFIG

Name	Base Address
CHIPCONFIG	0x4000_F000

Table 25. 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	0x4C34_7160	4.8.2
CHIPCONFIG_REVNR	0x0008	RO(1)	Revision Number register	0x0000_00xx	4.8.3

NOTE:

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

The base addresses and register map for the SCU are described in the tables below:

Table 26. Base Address of SCU

Name	Base Address
SCU	0x4000_0000

Table 27. Register Map of SCU

Name	Offset	Type	Description	Reset Value	Reference
SCU_SMR	0x0004	RW	System mode register	0x0000_0000	4.8.4
SCU_SRCR	0x0008	RW	System reset control register	0x0000_0000	4.8.5
SCU_WUER	0x0010	RW	Wake-up source enable register	0x0000_0000	4.8.6
SCU_WUSR	0x0014	RO	Wake-up source status register	0x0000_0000	4.8.7
SCU_RSER	0x0018	RW	Reset source enable register	0x0000_00D1	4.8.8
SCU_RSSR	0x001C	RWC1	Reset source status register	0x0000_0100	4.8.9
SCU_PRER1	0x0020	RW	Peripheral reset enable register 1	0x84FF_3F5F	4.8.10
SCU_PRER2	0x0024	RW	Peripheral reset enable register 2	0x273B_C1DB	4.8.11
SCU_PER1	0x0028	RW	Peripheral enable register 1	0x0000_000F	4.8.12
SCU_PER2	0x002C	RW	Peripheral enable register 2	0x0000_0101	4.8.13
SCU_PCER1	0x0030	RW	Peripheral clock enable register 1	0x0000_000F	4.8.14
SCU_PCER2	0x0034	RW	Peripheral clock enable register 2	0x0000_0101	4.8.15
SCU_CSCR	0x0040	RW	Clock source control register	0x0000_0028	4.8.16
SCU_SCCR	0x0044	RW	System clock control register	0x0300_0002	4.8.17
SCU_CMR	0x0048	RW	Clock monitor register	0x0000_0000	4.8.18
SCU_COR	0x0050	RW	Clock output register	0x0000_000F	4.8.19
SCU_NMICR	0x0054	RW	Non-maskable interrupt control register	0x0000_0000	4.8.20
SCU_NMISR	0x0058	RWC1	Non-maskable interrupt status register	0x0000_0000	4.8.21
SCU_PLLCON	0x0060	RW	PLL control register	0x0600_0000	4.8.22
SCU_VDCCON	0x0064	RW	VDC control register	0x0000_007F	4.8.23
SCU_LVICR	0x0068	RW	LVI control register	0x0000_0000	4.8.24
SCU_LVISR	0x006C	RC	LVI status register	0x0000_0000	4.8.25
SCU_LVRCR	0x0070	RW	LVR control register	0x0000_0000	4.8.26
SCU_EOSCR	0x0080	RW	External oscillator control register	0x8711_0000	4.8.27
SCU_MCCR1	0x0090	RW	MISC clock control register 1	0x0001_0000	4.8.28
SCU_MCCR2	0x0094	RW	MISC clock control register 2	0x0000_0000	4.8.29
SCU_MCCR3	0x0098	RW	MISC clock control register 3	0x0000_0000	4.8.29
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	0x0001_0000	4.8.34
SCU_SYSTEN	0x00F0	WO	System access enable	0x0000_0000	4.8.35

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]																															
0x4C34_7160																															
RO																															

15	CHIPID[31:0]	Chip ID bit.	
		0x4C34_7160	256 KB flash memory / 24 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]
-			0xXX
-			RO

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

4.8.4 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	LSE enablement in DEEP-SLEEP mode	
		0	Automatically disables the LSE in DEEP-SLEEP mode.
		1	Leaves the LSE enabled in DEEP-SLEEP mode.
12	HSEAON	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	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	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	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.

NOTE:

1. Even if users 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.

4.8.5 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 in SCU_SRCR register instead of NVIC_SystemReset() or SCB_AIRCR register setting for the microcontroller reset.

4.8.6 SCU_WUER: Wake-Up Source Enable Register

The SCU_WUER 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, USBWUE, Reserved	LPUART1WUE, LPUART0WUE, Reserved, I2C1WUE, I2C0WUE	Reserved, GPIOFWUE, GPIOEWUE, GPIODWUE, GPIOCWUE, GPIOBWUE, GPIOAWUE	Reserved, USART11WUE, USART10WUE, RTCCWUE, Reserved, FRTWUE, WDTWUE, LVIWUE
- 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

28	USBWUE	Whether or not to use the USB 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.
21	LPUART1WUE	Whether or not to use the LPUART1 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.
20	LPUART0WUE	Whether or not to use the LPUART0 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.
17	I2C1WUE	Whether or not to use the I2C1 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.
16	I2C0WUE	Whether or not to use the I2C0 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	Whether or not to 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	Whether or not to 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	Whether or not to 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	Whether or not to 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	Whether or not to 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	Whether or not to 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.
6	USART11WUE	Whether or not to use the USART11 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.

5	USART10WUE	Whether or not to use the USART10 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	RCTCWUE	Whether or not to use the RTCC 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	FRTWUE	Whether or not to use the FRT 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	Whether or not to 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	Whether or not to 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.

4.8.7 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	USBWU	Reserved	LPUART1WU	LPUART0WU	Reserved	I2C1WU	I2C0WU	Reserved	GPIOFWU	GPIOEWU	GPIODWU	GPIOCWU	GPIOBWU	GPIOAWU	Reserved	USART11WU	USART10WU	RTCCWU	Reserved	FRTWU	WDTWU	LVIWU
-	0	-	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	RO

28	USBWU	USB event Wake-up status
		0 Wake-up event does not occur.
		1 Wake-up event occurs.
21	LPUART1WU	LPUART1 event Wake-up status
		0 Wake-up event does not occur.
		1 Wake-up event occurs.
20	LPUART0WU	LPUART0 event Wake-up status
		0 Wake-up event does not occur.
		1 Wake-up event occurs.
17	I2C1WU	I2C1 event Wake-up status
		0 Wake-up event does not occur.
		1 Wake-up event occurs.
16	I2C0WU	I2C0 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.
6	USART11WU	USART11 event Wake-up status
		0 Wake-up event does not occur.
		1 Wake-up event occurs.

5	USART10WU	USART10 event Wake-up status
		0 Wake-up event does not occur.
		1 Wake-up event occurs.
4	RTCCWU	RTCC event Wake-up status
		0 Wake-up event does not occur.
		1 Wake-up event occurs.
2	FRTWU	FRT event Wake-up status
		0 Wake-up event does not occur.
		1 Wake-up event occurs.
1	WDTWU	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.

4.8.8 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
Reserved			LOCKUPRST	Reserved	PINRST	CPURST	SWRST	WDTRST	MCLKFRST	LSEFRST	HSEFRST	LVDRST
-			0	-	1	1	0	1	0	0	0	1
-			RW	-	RW	RW	RW	RW	RW	RW	RW	RW

9	LOCKUPRST	Whether to enable or disable 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	Whether to enable or disable 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	Whether to enable or disable 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	Whether to enable or disable the software reset signal
	0	Disables the signal to trigger a reset event.
	1	Enables the signal to trigger a reset event.
4	WDTRST	Whether to enable or disable the WDT reset signal
	0	Disables the signal to trigger a reset event.
	1	Enables the signal to trigger a reset event.
3	MCLKFRST	Whether to enable or disable 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	Whether to enable or disable 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	Whether to enable or disable 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	Whether to enable or disable the LVD reset signal
	0	Disables the signal to trigger a reset event.
	1	Enables the signal to trigger a reset event.

4.8.9 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	LSEFRST	HSEFRST	LVDRST
-			0	1	0	0	0	0	0	0	0	0
-			RWC1	RWC1	RWC1	RWC1	RWC1	RWC1	RWC1	RWC1	RWC1	RWC1

9	LOCKUPRST	Whether or not 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	Whether or not 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	Whether or not 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	Whether or not 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	Whether or not 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	Whether or not 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	Whether or not 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	Whether or not 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	Whether or not 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	Whether or not 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.

4.8.10 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 and SCU_PRER2. 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
VREFBUF	Reserved	RTCC	Reserved	TIMER7	TIMER6	TIMER5	TIMER4	TIMER3	TIMER2	TIMER1	TIMER0	Reserved	GPIOF	GPIOG	GPIOD	GPIOC	GPIOB	GPIOA	Reserved	FRT	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
RW	-	RW	--	RW	RW	RW	RW	RW	RW	RW	RW	-	RW	RW	RW	RW	RW	RW	-	RW	-	RW	RW	RW	RW	RW

31	VREFBUF	VREFBUF reset mask
		0 Masked
		1 Not masked
26	RTCC	RTCC 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
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
6	FRT	FRT 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 Disabled
		1 Not masked

4.8.11 SCU_PRER2: Peripheral Reset Enable Register 2

The SCU_PRER2 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
Reserved		CRC	Reserved		DAC1	DAC0	OPAMP	Reserved		ADC1	ADC0	LCD	Reserved	USB	HSICMU	LPUART1	LPUART0	Reserved					UART	USART11	USART10	Reserved		I2C1	I2C0	Reserved		SPI1	SPI0
-		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

29	CRC	CRC reset mask
		0 Masked
		1 Not masked
26	DAC1	DAC1 reset mask
		0 Masked
		1 Not masked
25	DAC0	DAC0 reset mask
		0 Masked
		1 Not masked
24	OPAMP	OPAMP 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
19	LCD	LCD reset mask
		0 Masked
		1 Not masked
17	USB	USB reset mask
		0 Masked
		1 Not masked
16	HSICMU	HSICMU reset mask
		0 Masked
		1 Not masked
15	LPUART1	LPUART1 reset mask
		0 Masked
		1 Not masked
14	LPUART0	LPUART0 reset mask
		0 Masked
		1 Not masked
8	UART	UART reset mask
		0 Masked
		1 Not masked
8	USART11	USART11 reset mask
		0 Masked
		1 Not masked
8	USART10	USART10 reset mask

		0	Masked
		1	Not masked
4	I2C1	I2C1 reset mask	
		0	Masked
		1	Not masked
3	I2C0	I2C0 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

4.8.12 SCU_PER1: Peripheral Enable Register 1

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

Because only a certain set of peripherals is enabled at initial reset, additional actions are required to enable other peripherals. By writing '0' to the bit corresponding to an unused peripheral, the peripheral can be reset.

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
VREFBUF	Reserved				RTCC	Reserved		TIMER7	TIMER6	TIMER5	TIMER4	TIMER3	TIMER2	TIMER1	TIMER0	Reserved		GPIOF	GPIOE	GPIOD	GPIOC	GPIOB	GPIOA	Reserved	FRT	Reserved	DMA	Reserved			
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	RO			

31	VREFBUF	Whether to enable or disable VREFBUF
		0 Disabled
		1 Enabled
26	RTCC	Whether to enable or disable RTCC
		0 Disabled
		1 Enabled
23	TIMER7	Whether to enable or disable TIMER7
		0 Disabled
		1 Enabled
22	TIMER6	Whether to enable or disable TIMER6
		0 Disabled
		1 Enabled
21	TIMER5	Whether to enable or disable TIMER5
		0 Disabled
		1 Enabled
20	TIMER4	Whether to enable or disable TIMER4
		0 Disabled
		1 Enabled
19	TIMER3	Whether to enable or disable TIMER3
		0 Disabled
		1 Enabled
18	TIMER2	Whether to enable or disable TIMER2
		0 Disabled
		1 Enabled
17	TIMER1	Whether to enable or disable TIMER1
		0 Disabled
		1 Enabled
16	TIMER0	Whether to enable or disable TIMER0
		0 Disabled
		1 Enabled
13	GPIOF	Whether to enable or disable GPIOF
		0 Disabled
		1 Enabled
12	GPIOE	Whether to enable or disable GPIOE

		0	Disabled
		1	Enabled
11	GPIOD	Whether to enable or disable GPIOD	
		0	Disabled
		1	Enabled
10	GPIOC	Whether to enable or disable GPIOC	
		0	Disabled
		1	Enabled
9	GPIOB	Whether to enable or disable GPIOB	
		0	Disabled
		1	Enabled
8	GPIOA	Whether to enable or disable GPIOA	
		0	Disabled
		1	Enabled
6	FRT	Whether to enable or disable FRT	
		0	Disabled
		1	Enabled
4	DMA	Whether to enable or disable DMA	
		0	Disabled
		1	Enabled

4.8.13 SCU_PER2: Peripheral Enable Register 2

The SCU_PER2 is 32 bits 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
Reserved		CRC	Reserved		DAC1	DAC0	OPAMP	Reserved		ADC1	ADC0	LCD	Reserved	USB	HSICMU	LPUART1	LPUART0	Reserved					UART	USART11	USART10	Reserved		I2C1	I2C0	Reserved		SPI1	SPI0
-		0	-		0	0	0	-		0	0	0	-	0	0	0	0	-					1	0	0	-		0	0	-		0	1
-		RW	-		RW	RW	RW	-		RW	RW	RW	-	RW	RW	RW	RW	-					RW	RW	RW	-		RW	RW	-		RW	RW

29	CRC	Whether to enable or disable CRC
		0 Disabled
		1 Enabled
26	DAC1	Whether to enable or disable DAC1
		0 Disabled
		1 Enabled
25	DAC0	Whether to enable or disable DAC0
		0 Disabled
		1 Enabled
24	OPAMP	Whether to enable or disable OPAMP
		0 Disabled
		1 Enabled
21	ADC1	Whether to enable or disable ADC1
		0 Disabled
		1 Enabled
20	ADC0	Whether to enable or disable ADC0
		0 Disabled
		1 Enabled
19	LCD	Whether to enable or disable LCD
		0 Disabled
		1 Enabled
17	USB	Whether to enable or disable USB
		0 Disabled
		1 Enabled
16	HSICMU	Whether to enable or disable HSICMU
		0 Disabled
		1 Enabled
15	LPUART1	Whether to enable or disable LPUART1
		0 Disabled
		1 Enabled
14	LPUART0	Whether to enable or disable LPUART0
		0 Disabled
		1 Enabled
8	UART	Whether to enable or disable UART
		0 Disabled
		1 Enabled
7	USART11	Whether to enable or disable USART11
		0 Disabled
		1 Enabled
6	USART10	Whether to enable or disable USART10

		0	Disabled
		1	Enabled
4	I2C1	Whether to enable or disable I2C1	
		0	Disabled
		1	Enabled
3	I2C0	Whether to enable or disable I2C0	
		0	Disabled
		1	Enabled
1	SPI1	Whether to enable or disable SPI1	
		0	Disabled
		1	Enabled
0	SPI0	Whether to enable or disable SPI0	
		0	Disabled
		1	Enabled

4.8.14 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 or SCU_PCER2. A peripheral cannot reset or operate normally until its clock is enabled. If a peripheral is not used, the users must write '0' to its corresponding bit in the SCU_PCER1 or SCU_PECR2 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
VREFBUF	Reserved				RTCC	Reserved		TIMER7	TIMER6	TIMER5	TIMER4	TIMER3	TIMER2	TIMER1	TIMER0	Reserved		GPIOF	GPIOE	GPIOD	GPIOC	GPIOB	GPIOA	Reserved		FRT	Reserved		DMA	Reserved			
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	RO			

31	VREFBUF	Whether to enable or disable the clock to VREFBUF
		0 Disabled
		1 Enabled
26	RTCC	Whether to enable or disable the clock to RTCC
		0 Disabled
		1 Enabled
23	TIMER7	Whether to enable or disable the clock to TIMER7
		0 Disabled
		1 Enabled
22	TIMER6	Whether to enable or disable the clock to TIMER6
		0 Disabled
		1 Enabled
21	TIMER5	Whether to enable or disable the clock to TIMER5
		0 Disabled
		1 Enabled
20	TIMER4	Whether to enable or disable the clock to TIMER4
		0 Disabled
		1 Enabled
19	TIMER3	Whether to enable or disable the clock to TIMER3
		0 Disabled
		1 Enabled
18	TIMER2	Whether to enable or disable the clock to TIMER2
		0 Disabled
		1 Enabled
17	TIMER1	Whether to enable or disable the clock to TIMER1
		0 Disabled
		1 Enabled
16	TIMER0	Whether to enable or disable the clock to TIMER0
		0 Disabled
		1 Enabled
13	GPIOF	Whether to enable or disable the clock to GPIOF
		0 Disabled
		1 Enabled
12	GPIOE	Whether to enable or disable the clock to GPIOE

		0	Disabled
		1	Enabled
11	GPIOD	Whether to enable or disable the clock to GPIOD	
		0	Disabled
		1	Enabled
10	GPIOC	Whether to enable or disable the clock to GPIOC	
		0	Disabled
		1	Enabled
9	GPIOB	Whether to enable or disable the clock to GPIOB	
		0	Disabled
		1	Enabled
8	GPIOA	Whether to enable or disable the clock to GPIOA	
		0	Disabled
		1	Enabled
6	FRT	Whether to enable or disable the clock to FRT	
		0	Disabled
		1	Enabled
4	DMA	Whether to enable or disable the clock to DMA	
		0	Disabled
		1	Enabled

4.8.15 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.

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
Reserved		CRC	Reserved		DAC1	DAC0	OPAMP	Reserved		ADC1	ADC0	LCD	Reserved	USB	HSICMU	LPUART1	LPUART0	Reserved					UART	USART11	USART10	Reserved		I2C1	I2C0	Reserved		SPI1	SPI0
-		0	-		0	0	0	-		0	0	0	-	0	0	0	0	-					1	0	0	-		0	0	-		0	1
-		RW	-		RW	RW	RW	-		RW	RW	RW	-	RW	RW	RW	RW	-					RW	RW	RW	-		RW	RW	-		RW	RW

29	CRC	Whether to enable or disable the clock to CRC
		0 Disabled
		1 Enabled
26	DAC1	Whether to enable or disable the clock to DAC1
		0 Disabled
		1 Enabled
25	DAC0	Whether to enable or disable the clock to DAC0
		0 Disabled
		1 Enabled
24	OPAMP	Whether to enable or disable the clock to OPAMP
		0 Disabled
		1 Enabled
21	ADC1	Whether to enable or disable the clock to ADC1
		0 Disabled
		1 Enabled
20	ADC0	Whether to enable or disable the clock to ADC0
		0 Disabled
		1 Enabled
19	LCD	Whether to enable or disable the clock to LCD
		0 Disabled
		1 Enabled
17	USB	Whether to enable or disable the clock to USB
		0 Disabled
		1 Enabled
16	HSICMU	Whether to enable or disable the clock to HSICMU
		0 Disabled
		1 Enabled
15	LPUART1	Whether to enable or disable the clock to LPUART1
		0 Disabled
		1 Enabled
14	LPUART0	Whether to enable or disable the clock to LPUART0
		0 Disabled
		1 Enabled
8	UART	Whether to enable or disable the clock to UART
		0 Disabled
		1 Enabled
7	USART11	Whether to enable or disable the clock to USART11
		0 Disabled

		1	Enabled
6	USART10	Whether to enable or disable the clock to USART10	
		0	Disabled
		1	Enabled
4	I2C1	Whether to enable or disable the clock to I2C1	
		0	Disabled
		1	Enabled
3	I2C0	Whether to enable or disable the clock to I2C0	
		0	Disabled
		1	Enabled
1	SPI1	Whether to enable or disable the clock to SPI1	
		0	Disabled
		1	Enabled
0	SPI0	Whether to enable or disable the clock to SPI0	
		0	Disabled
		1	Enabled

4.8.16 SCU_CSCR: Clock Source Control Register

A34L716 includes several 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.

NOTES:

1. For the LSE operation, the corresponding Pn_MR registers of the SXIN, SXOUT ports must be set to '111'. LSE (PD4, PD5).
2. For the HSE operation, the corresponding Pn_MR registers of the XIN, XOUT ports must be set to '111'. HSE (PD2, PD3).

4.8.17 SCU_SCCR: System Clock Control Register

By configuring the SCU_SCCR, users can select one from many different clock sources 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]
-	0011	-	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	PLLCLKSE(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	
		001	LSE(3)
		010	HSI
		110	HSE
		111	PLL
		Others	Reserved

NOTES:

1. When changing PLLCLKSEL, both the HSI and HSE must be enabled. If the MCLKSEL[2:0] bits are set to use the HSE, PD2 and PD3 must be set for the functions of XIN and XOUT respectively.

2. When changing MCLKSEL[2:0], both clock sources should be alive. Ex) Both HSI and HSE should be alive, otherwise the chip will malfunction.
3. If the MCLKSEL[2:0] bits are set to use the LSE, PD4 and PD5 must be set for the functions of SXIN and SXOUT respectively.

4.8.18 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	MCLK error automatic recovery	
		0	Turns the MCLK to the HSI when clock error occurs.
		1	Disables the MCLK automatic recovery.
11	LSEMNT	Whether to enable or disable external LSE monitoring	
		0	Disables external LSE monitoring.
		1	Enables external LSE monitoring.
10	LSEIE	Whether to enable or disable 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	Whether to enable or disable external MCLK monitoring	
		0	Disables external MCLK monitoring.
		1	Enables external MCLK monitoring.
6	MCLKIE	Whether to enable or disable the external MCLK error interrupt	
		0	Disables the external MCLK error interrupt.
		1	Enables the external MCLK error interrupt.
5	MCLKFAIL	External MCLK error interrupt flag	
		0	The external MCLK error interrupt has not occurred.
		1	Read: The external MCLK error interrupt has occurred. Write: Clears the interrupt flag.
4	MCLKSTS	External MCLK status flag	
		0	The external MCLK is oscillating normally.
		1	The external MCLK is not oscillating.
3	HSEMNT	Whether to enable or disable external HSE monitoring	
		0	Disables external HSE monitoring.
		1	Enables external HSE monitoring.
2	HSEIE	Whether to enable or disable 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.

NOTE:

1. Clock monitoring can be active only when the corresponding clock is enabled.

4.8.19 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	

4.8.20 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, it 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		WDTINTEN	MCLKFAILIEN	LVIEN
-	0x00	0	-	0	0	-		0	0	0
-	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	Whether to enable or disable the NMIs (Interrupts that have been specified as NMI sources) 0 Disables. 1 Enables.
13	DFEDEN	Whether to enable or disable the data flash ECC Double error interrupt as an NMI 0 Disables 1 Enables.
12	CFEDEN	Whether to enable or disable the code flash ECC Double error interrupt as an NMI 0 Disables 1 Enables.
2	WDTINTEN	Whether to enable or disable the WDT interrupt as an NMI 0 Disables 1 Enables.
1	MCLKFAILEN	Whether to enable or disable the MCLK error interrupt as an NMI 0 Disables 1 Enables.
0	LVIEN	Whether to enable or disable the LVI error interrupt as an NMI 0 Disables 1 Enables.

4.8.21 SCU_NMISR: NMI Status Register

The SCU_NMISR 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		WDTINTSTS	MCLKFAILSTS	LVISTS
0x00	-	0	-	0	0	-		0	0	0
WO	-	RO	-	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 or not 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	Whether or not 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	Whether or not 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).
2	WDTINTSTS	Whether or not 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	Whether or not 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	Whether or not 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).

4.8.22 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 a 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	PLLRSTB	PLLEN	BYPASSB	PLLMODE	Reserved	PREDIV[2:0]	POSTDIV1[7:0]	POSTDIV2[3:0]	OUTDIV[3:0]
0	-	0	0	0	0	-	000	0x00	0000	0000
RO	-	RW	RW	RW	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.		
27 24	CTRLOTP[3:0]	PLL current option			
		[27:26]	Current option	[25:24]	VCO bias
		00	5 μA	00	×1/4
		01	10 μA	01	×1/2
		10	15 μA	10	×1
		11	20 μA	11	×2
23	PLLRSTB	PLL reset			
		0	PLL reset signal applied.		
		1	PLL reset signal released (PLL operation).		
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).		
20	PLLMODE(2)	PLL voltage-controlled oscillator (VCO) mode selection			
		0	The VCO frequency is the same as FOUT.		
		1	The VCO frequency is twice FOUT.		
18 16	PREDIV[2:0]	FIN predivider bit			
		0 to 7	FIN divided by (PREDIV + 1), (FIN / 1 to FIN / 8)		
15 8	POSTDIV1[7:0]	Feedback control 1 (N1)			
		N1 value in the formula (N1 = 0x00 to 0xFF)			
7 4	POSTDIV2[3:0](3)	Feedback control 1 (N2)			
		N2 value in the formula (N2 = 0x0 to 0xF)			
3 0	OUTDIV[3:0](3)	Output divider control (P)			
		P value in the formula (P = 0x0 to 0xF)			

NOTES:

1. Refer to section 4.4.5 for how to set the PLL clock using the SCU_PLLCON register.
2. PLLMODE must be '0'.
3. (POSTDIV2 or OUTDIV ≥ 3) or (POSTDIV2 and OUTDIV ≥ 1)

4.8.23 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. The delay value can be set to the max. value of 0.8 ms in the VDCWDLY bits (Max: 0xFF).

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

The LVI block detects low voltage by 7 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	Reserved	LVIVS[2:0]
-			0	-	0	0	-	000
-			RW	-	RW	RW	-	RW

7	LVIEN	Whether to enable or disable the LVI	
		0	Disables.
		1	Enables.
5	LVIINTEN	Whether to enable or disable the LVI interrupt.	
		0	Disables.
		1	Enables.
4	LVIAON	Whether or not 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.
2 0	LVIVS[2:0]	LVI interrupt voltage selection bits	
		000	2.07 V
		001	2.24 V
		010	2.40 V
		011	2.54 V
		100	2.67 V
		101	2.85 V
		110	2.90 V

4.8.25 SCU_LVISR: Low-Voltage Indicator (LVI) 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			LVIIFLAG	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	LVIIFLAG	LVI interrupt flag	
		0	Not flagged
		1	Flagged (writing a '1' to clear to flag bit).
0	LVIINTSTS	Whether or not the LVI interrupt is active (Raw data)	
		0	The LVI interrupt has ended.
		1	The LVI interrupt is active.

4.8.26 SCU_LVRCR: Low-Voltage Detect Reset (LVR) 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	Reserved	LVRVS[2:0]
-		0x00	-	0	-	000
-		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 or not 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.
2 0	LVRVS [2:0]	LVR voltage selection bits	
		000	1.58 V
		001	1.96 V
		010	2.23 V
		011	2.54 V
		100	2.75 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. Refer to section 4.6.2.

4.8.27 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 bits 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
LSEISELF	Reserved	LSEFSMODE	LSEFSTIME	LSEISEL[3:0]				Reserved								Reserved						HSEISEL[1:0]		Reserved			HSENFEN	Reserved		HSENFSEL[1:0]	
1	-	0	0	0	1	1	1	00010001								-						0	0	-			0	-		00	
RO	-	RW	RW	RW				-								-						RW		-			RW	-		RW	

31	LSEISELF	Flag indicating that the ISEL value of the OSOC IP is being changed to the LSEISEL value.	
		0	If LSE_FS=1, ISEL of SOSC IP will be applied as ='b111.
		1	The ISEL bit of SOSC IP will be equal to LSEISEL bit value. (If LSE_FS is '0', this bit will be always '1')
29	LSEFSMODE	SOSC Fast Start-up mode	
		0	The ISEL bit in SOSC IP set to the value of LSEISEL bit in normal mode. (keep the register value)
		1	Fast start-up mode
28	LSEFSTIME	If LSEFSMODE=1, set the time when ISEL=LSEISEL of SOSC IP is applied.	
		0	Typical 1.1 s
		1	Typical 2 s
27 24	LSEISEL[3:0](1)	LSE current capability selection	
		0000	×1.0
		0001	×1.3
		0010	×1.6
		0011	×2.0
		0100	×2.3
		0101	×2.6
		0110	×3.0
		0111	×3.3
		1000	×3.6
		1001	×4.0
		1010	×4.3
		1011	×4.6
		1100	×5.0
		1101	×5.3
		1110	×5.6
		1111	×6.0
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	1 MHz < 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

NOTE:

1. The corresponding value is a simulation (typ.) value and may differ from the actual measurement value.

Table 28. External Oscillator Control Settings

Freq. (MHz)	HSENFSEL[1:0]	HSEISEL[1:0]	NC Delay (ns)
4	00, 01, 10, 11	11, 10, 01, 00	8, 13, 18, 23
8	01, 10, 11	10, 01, 00	8, 13, 18
12	10, 11	01, 00	8, 13
16	11	00	8

4.8.28 SCU_MCCR1: Miscellaneous Clock Control Register 1

SCU_MCCR1 provides a different MCLK division ratio for each peripheral. This register is 32 bits 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

26 24	WDTCSEL[2:0]	WDT clock source selection bit	
		000	LSI (40 kHz)
		001	LSE (32.768 kHz)
		100	MCLK
		101	HSI (32 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 (40 kHz)
		001	LSE (32.768 kHz)
		100	MCLK
		101	HSI (32 MHz)
		110	HSE
		111	PLL
		Other	Reserved
7 0	STCDIV[7:0]	SYSTICK clock divider (N)	
		0x00	Disables the divider
		0xN	Clock source / (2 × N), N must be larger than zero.

4.8.29 SCU_MCCR2: Miscellaneous Clock Control Register 2

SCU_MCCR2 provides a different MCLK division ratio for each peripheral. This register is 32 bits wide.

SCU_MCCR2=0x4000_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	LCDCSEL[2:0]	LCDCDIV[7:0]	Reserved	RTCCCSEL[2:0]	RTCCCDIV[7:0]
-	000	0x00	-	000	0x00
-	RW	RW	-	RW	RW

26 24	LCDCSEL[2:0](1)	LCD clock source selection bit	
		000	LSI (40 kHz)
		001	LSE (32.768 kHz)
		100	MCLK
		101	HSI (32 MHz)
		110	HSE
		111	PLL
		Other	Reserved
23 16	LCDCDIV[7:0]	LCD clock divider (N)	
		0x00	Disables the divider
		0xN	Clock source / N, N must be larger than zero.
10 8	RTCCCSEL[2:0]	RTCC clock source selection bit	
		000	LSI (40 kHz)
		001	LSE (32.768 kHz)
		100	MCLK
		101	HSI (32 MHz)
		110	HSE
		111	PLL
		Other	Reserved
7 0	RTCCCDIV[7:0]	RTCC clock divider (N)	
		0x00	Disables the divider
		0xN	Clock source / N, N must be larger than zero.

NOTE:

1. The LCD Clock can be selected as MCLK, HSI, HSE, and PLL, but its use is not recommended, so only LSI and LSE can be used.

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 bits 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	TIMER47CSEL[2:0]	TIMER47CDIV[7:0]	Reserved	TIMER03CSEL[2:0]	TIMER03CDIV[7:0]
-	000	0x00	-	000	0x00
-	RW	RW	-	RW	RW

26 24	TIMER47CSEL[2:0]	Clock source selection for TIMER4 through TIMER7	
		000	LSI (40 kHz)
		001	LSE (32.768 kHz)
		100	MCLK
		101	HSI (32 MHz)
		110	HSE
		111	PLL
		Other	Reserved
23 16	TIMER47CDIV[7:0][1]	Clock divider (N) for TIMER4 through TIMER7	
		0x00	Disables the divider
		0xN	Clock source / N, N must be larger than 0. If the timer clock (TMCLK) is set in this register, the clock speed must satisfy PCLK ≥ TMCLK × 3.
10 8	TIMER03CSEL[2:0]	Clock source selection for TIMER0 through TIMER3	
		000	LSI (40 kHz)
		001	LSE (32.768 kHz)
		100	MCLK
		101	HSI (32 MHz)
		110	HSE
		111	PLL
		Other	Reserved
7 0	TIMER03CDIV[7:0][1]	Clock divider (N) for TIMER0 through TIMER3	
		0x00	Disables the divider
		0xN	Clock source / N, N must be larger than 0. If the timer clock (TMCLK) is set in this register, the clock speed must satisfy PCLK ≥ TMCLK × 3.

NOTE:

1. Because the PCLK is the standard of operation, 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 frequency is slower than half of 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 bits 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	LPUARTCSEL[2:0]	LPUARTCDIV[7:0]
-	000	0x00	-	000	0x00
-	RW	RW	-	RW	RW

26 24	PGADCSEL[2:0] (PA, PB)	Clock source for port group A debouncing	
		000	LSI (40 kHz)
		001	LSE (32.768 kHz)
		100	MCLK
		101	HSI (32 MHz)
		110	HSE
		111	PLL
		Other	Reserved
23 16	PGADCDIV[7:0] (PA, PB)	Clock divider N for port group A debouncing	
		0x00	Disables the divider
		0xN	Clock source / N, N must be larger than zero.
10 8	LPUARTCSEL[2:0]	LPUART clock source selection bit	
		000	LSI (40 kHz)
		001	LSE (32.768 kHz)
		100	MCLK
		101	HSI (32 MHz)
		110	HSE
		111	PLL
		Other	Reserved
7 0	LPUARTCDIV[7:0]	LPUART clock divider (N)	
		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)	Clock source for port group C debouncing	
		000	LSI (40 kHz)
		001	LSE (32.768 kHz)
		100	MCLK
		101	HSI (32 MHz)
		110	HSE
		111	PLL
		Other	Reserved
23 16	PGCDCDIV[7:0] (PE, PF)	Clock divider (N) for port group C 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 debouncing	
		000	LSI (40 kHz)
		001	LSE (32.768 kHz)
		100	MCLK
		101	HSI (32 MHz)
		110	HSE
		111	PLL
		Other	Reserved
7 0	PGBDCDIV[7:0] (PC, PD)	Clock divider (N) for port group B 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 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	USBCSEL[2:0]	USBCDIV[7:0]	Reserved	FRTCSEL[2:0]	FRTCDIV[7:0]
-	000	0x00	-	000	0x00
-	RW	RW	-	RW	RW

26 24	USBCSEL[2:0]	USB clock source selection bit	
		000	LSI (40 kHz)
		001	LSE (32.768 kHz)
		100	MCLK
		101	HSI (32 MHz)
		110	HSE
		111	PLL
		Other	Reserved
23 16	USBCDIV[7:0]	USB clock divider (N)	
		0x00	Disables the divider
		0xN	Clock source / N, N must be larger than zero.
10 8	FRTCSEL[2:0]	FRT clock source selection bit	
		000	LSI (40 kHz)
		001	LSE (32.768 kHz)
		100	MCLK
		101	HSI (32 MHz)
		110	HSE
		111	PLL
		Other	Reserved
7 0	FRTCDIV[7:0]	FRT clock divider (N)	
		0x00	Disables the divider
		0xN	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 bits 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	UARTCSEL[2:0]	UARTCDIV[7:0]	Reserved	OPAMPCSEL[2:0]	OPAMPCDIV[7:0]
-	000	0x01	-	000	0x00
-	RW	RW	-	RW	RW

26 24	UARTCSEL[2:0]	UART clock source selection bit	
		000	LSI (40 kHz)
		001	LSE (32.768 kHz)
		100	MCLK
		101	HSI (32 MHz)
		110	HSE
		111	PLL
		Other	Reserved
23 16	UARTCDIV[7:0]	UART clock divider (N)	
		0x00	Disables the divider
		0xN	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.
10 8	OPAMPCSEL[2:0](1)	OPAMP clock source selection bit	
		000	LSI (40 kHz)
		001	LSE (32.768 kHz)
		100	MCLK
		101	HIS/4 (32 MHz/4 = 8 MHz)
		110	HSE
		111	PLL
		Other	Reserved
7 0	OPAMPCDIV[7:0]	OPAMP clock divider (N)	
		0x00	Disables the divider
		0xN	Clock source / N, N must be larger than zero.

NOTE:

1. The OPAMP Clock must be manually inputted at 50 kHz (LSI, LSE cannot be used). If the user selects HSI, it is automatically divided into four internally (32 MHz/4 = 8 MHz).

4.8.35 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	--
-			RO	WO

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.	

4.8.36 SCU Register Map Summary

Table 29. 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	1	1	0	0	0	0	1	1	0	1	0	0	0	1	1	1	0	0	0	1	0	1	1	0	0	0	0	0
0x08	CHIPCONFIG_ REVNR																										REVNO[7:0]						
	Reset value																										x	x	x	x	x	x	x

Table 30. 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																			LSEAON	HSEAON	PLLAON	HSIAON	LSIAON	VDCAON			PREVMODE[1:0]					
	Reset value																			0	0	0	0	0	0			1	1				
0x08	SCU_SRCR																												STBYO				SWRST
	Reset value																												0				0
0x10	SCU_WUER				USBWUE							LPUART1WUE	LPUART0WUE			I2C1WUE	I2C0WUE			GPIOFWUE	GPIOEWUE	GPIODWUE	GPIOCWUE	GPIOBWUE	GPIOAWUE			USART11WUE	USART10WUE	RTCWUE		FRTWUE	WDTWUE	LVIWUE
	Reset value				0							0	0			0	0			0	0	0	0	0	0			0	0	0		0	0	0
0x14	SCU_WUSR				USBWU							LPUART1WU	LPUART0WU			I2C1WU	I2C0WU			GPIOFWU	GPIOEWU	GPIODWU	GPIOCWU	GPIOBWU	GPIOAWU			USART11WU	USART10WU	RTCWU		FRTWU	WDTWU	LVIWU
	Reset value				0							0	0			0	0			0	0	0	0	0	0			0	0	0		0	0	0
0x18	SCU_RSER																							LOCKUPRST		PINRST	CPURST	SWRST	WDTRST	MCLKFRST	LSEFRST	HSEFRST	LVDRST
	Reset value																							0		1	1	0	1	0	0	0	1
0x1C	SCU_RSSR																							LOCKUPRST	PORST	PINRST	CPURST	SWRST	WDTRST	MCLKRST	LSERST	HSERST	LVDRST
	Reset value																							0	1	0	0	0	0	0	0	0	0

Table30. 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	VREFBUF	Res.	Res.	Res.	Res.	RTC	Res.	Res.	TIMER7	TIMER6	TIMER5	TIMER4	TIMER3	TIMER2	TIMER1	TIMER0			GPIOF	GPIOE	GPIOD	GPIOC	GPIOB	GPIOA	Res.	FRT		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
0x24	SCU_PRER2			CRC			DAC1	DAC0	OPAMP	Res.	Res.	ADC1	ADC0	LCD		USB	HSICMU	LPUART1	LPUART0	Res.	Res.	Res.	Res.	Res.	UART0	USART11	USART10		I2C1	I2C0	Res.	SPI1	SPI0
	Reset value			1			1	1	1			1	1	1		1	1	1	1						1	1	1		1	1		1	1
0x28	SCU_PER1	VREFBUF	Res.				RTC			TIMER7	TIMER6	TIMER5	TIMER4	TIMER3	TIMER2	TIMER1	TIMER0			GPIOF	GPIOE	GPIOD	GPIOC	GPIOB	GPIOA		FRT	Res.	DMA				
	Reset value	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			CRC			DAC1	DAC0	OPAMP			ADC1	ADC0	LCD		USB	HSICMU	LPUART1	LPUART0						UART0	USART11	USART10		I2C1	I2C0		SPI1	SPI0
	Reset value			0			0	0	0			0	0	0		0	0	0	0						1	0	0		0	0		0	1
0x30	SCU_PCER1	VREFBUF	Res.				RTC			TIMER7	TIMER6	TIMER5	TIMER4	TIMER3	TIMER2	TIMER1	TIMER0			GPIOF	GPIOE	GPIOD	GPIOC	GPIOB	GPIOA		FRT	Res.	DMA				
	Reset value	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			CRC			DAC1	DAC0	OPAMP			ADC1	ADC0	LCD		USB	HSICMU	LPUART1	LPUART0						UART0	USART11	USART10		I2C1	I2C0		SPI1	SPI0
	Reset value			0			0	0	0			0	0	0		0	0	0	0						1	0	0		0	0		0	1
0x40	SCU_CSCR																									LSECON	Res.	LSICON	Res.	HSICON	Res.	HSECON	Res.
	Reset value																									0		1		1		0	
0x44	SCU_SCCR					HCLKDIV[3:0]									PCLKDIV[2:0]						PLLCLKSEL			PLLPREDIV[1:0]							MCLKSEL[2:0]		
	Reset value					0	0	1	1						0	0	0				0			0	0						0	1	0
0x48	SCU_CMR																	MCLKREC				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																									CLKOINSEL[2:0]			CLKOEN	CLKODIV[3:0]			
	Reset value																									0	0	0	0	1	1	1	1

Table30. 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									NMISRC[7:0]								NMIINEN		DFEDEN	CFEDEN										WDTINTEN	MCLKFAILEN	LVIEN
	Reset value									0	0	0	0	0	0	0	0	0		0	0										0	0	0
0x58	SCU_NMISR	WTIDKY[7:0]																NMIINTSTS		DFEDSTS	CFEDSTS										WDTINTSTS	MCLKFAILSTS	LVISTS
	Reset value	0	0	0	0	0	0	0	0									0		0	0										0	0	0
0x60	SCU_PLLCON	PLLLOCK				CTRLOTP[3:0]				PLLRSTB	PLLEN	BYPASSB	PLLMODE		PREDIV[2:0]			POSTDIV1[7:0]								POSTDIV2[3:0]				OUTDIV[3:0]			
	Reset value	0				0	1	1	0	0	0	0	0		0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
0x64	SCU_VDCCON																									VDCWDLY[7:0]							
	Reset value																									0	1	1	1	1	1	1	1
0x68	SCU_LVICR																										LVIEN		LVIINTEN	LVIAON		LVIVS[2:0]	
	Reset value																										0		0	0		0	0
0x6C	SCU_LVISR	WTIDKY[7:0]																										LVIFLAG					LVIINTSTS
	Reset value	0	0	0	0	0	0	0																				0					0
0x70	SCU_LVRCR																	LVREN[7:0]								Reserved			LVRAON	Reserved	LVRVS[2:0]		
	Reset value																	0	0	0	0	0	0	0	0				0		0	0	0
0x80	SCU_EOSCR	LSEISELF		LSEFSMODE	LSEFSTIME	LSEISEL[3:0]				LSEDCBLOCK		LSEFAMPSEL [1:0]				LSEFBSEL	LSENFEN							HSEISEL[1:0]				HSENFEN			HSENFSEL[1:0]	
	Reset value	1		0	0	0	1	1	1	0		0	1			0	1							0	0				0			0	0
0x90	SCU_MCCR1						WDTCSEL[2:0]			WDTCDIV[7:0]													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	0

Table30. 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
0x94	SCU_MCCR2						LCDCSEL[2:0]			LCDCDIV[7:0]													RTCCSEL[2:0]			RTCCDIV[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
0x98	SCU_MCCR3						TIMER47CSEL[2:0]			TIMER47CDIV[7:0]													TIMER03CSEL[2:0]			TIMER03CDIV[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
0x9C	SCU_MCCR4						PGADCSEL[2:0]			PGADCDIV[7:0]													LPUARTCSEL[2:0]			LPUARTCDIV[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
0xA0	SCU_MCCR5						PGCDCSEL[2:0]			PGCDCDIV[7:0]													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
0xA4	SCU_MCCR6						USBCSEL[2:0]			USBCDIV[7:0]													FRTCSEL[2:0]			FRTCDIV[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
0xA8	SCU_MCCR7						UARTCSE[2:0]			UARTCDIV[7:0]													OPAMPCSEL[2:0]			OPAMPCDIV[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
0xF0	SCU_SYSTEN																								ENS	SYSTEN[7:0]							
	Reset value																								0	0	0	0	0	0	0	0	0

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

5.1 Introduction

5.1.1 PCU Introduction

The A34L716 has a Port Control Unit (PCU) module that controls the external input and output (I/O) ports. By setting 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 four GPIO interrupts are supported, including GPIOn (n = A to F).
- 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/Pull-down resistor 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
- Input pull-down
- Analog
- 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:

- PC6: TMS / SWDIO with pull-up
- PC5: TCK / SWCLK with pull-up
- PC4: nTRST with pull-up
- PC3: TDI with pull-up
- PC2: TDO / SWO with Output Low
- PC12: nRESET with pull-up
- PC13: nBOOT with pull-up

Following figures (Figure 32, Figure 33, and Figure 34) show block diagrams of GPIO and external interrupt I/O pins, respectively.

Figure 32. PCU and GPIO Block Diagram

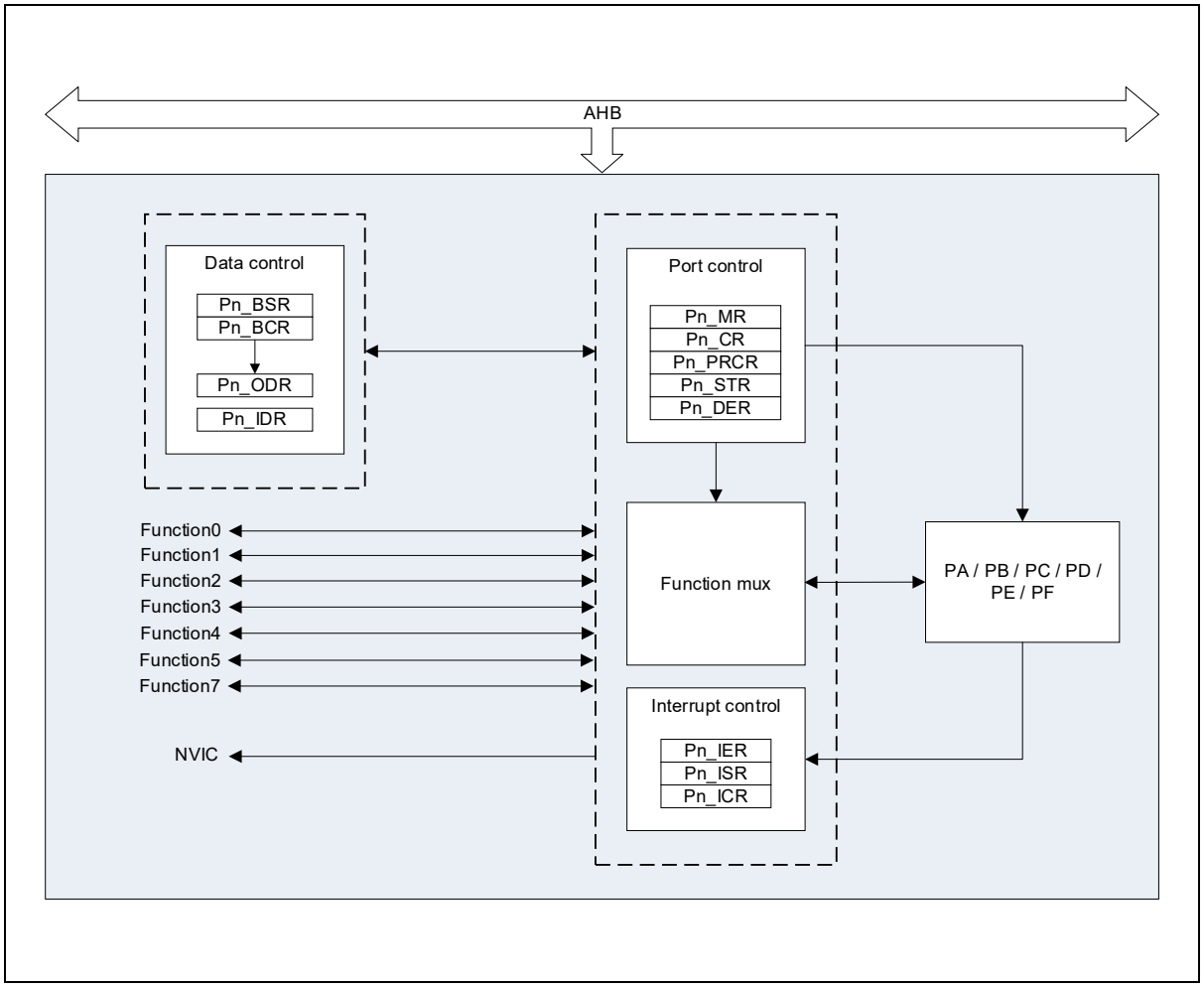


Figure 33. I/O Port Block Diagram (GPIOs)

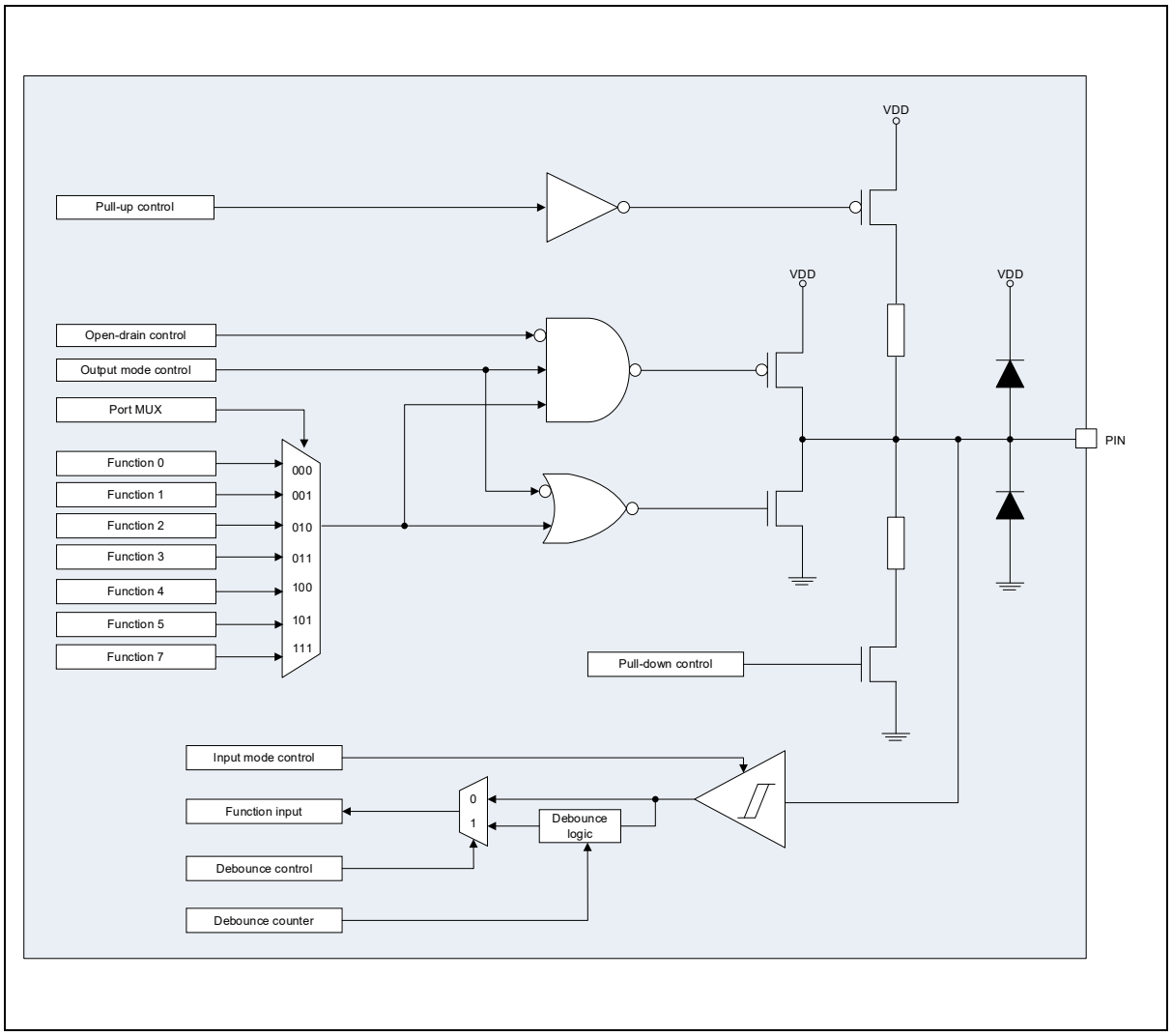
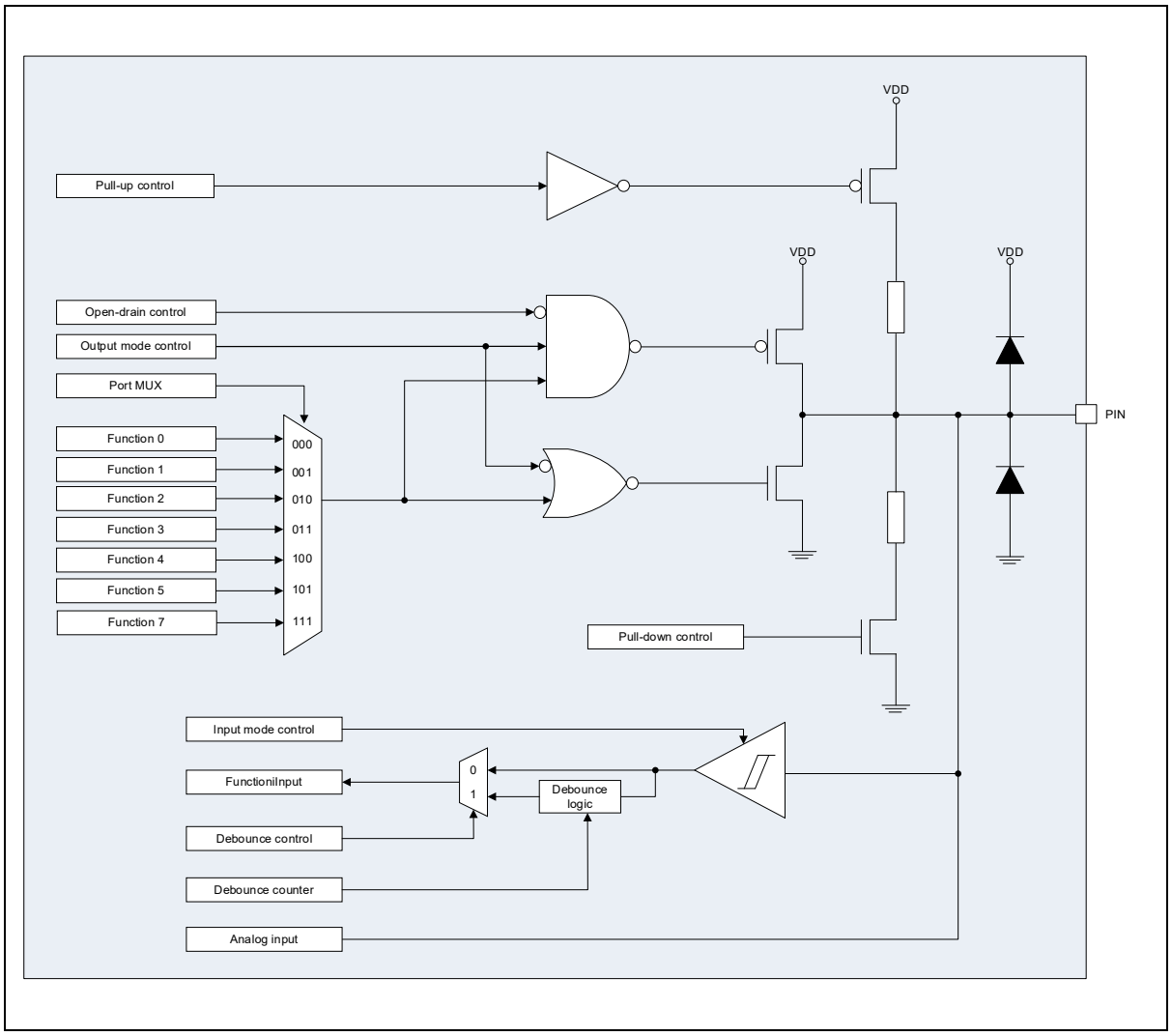


Figure 34. I/O Port Block Diagram (ADC, DAC, OPAMP and External Oscillator Pins)



5.3.1 GPIO Pins and Internal Signals

The internal signal of PCU and GPIO is described in Table 31.

Table 31. PCU and GPIO Internal Signal

Signal Name	Signal Type	Description
AHB	Clock	AHB Clock

Table 32 describes seven pins assigned for PCU and GPIO modules.

Table 32. PCU and GPIO Pins

Pin Name	Type	Description
PA	IO	PA0 to PA15
PB	IO	PB0 to PB15
PC	IO	PC0 to PC15
PD	IO	PD0 to PD15
PE	IO	PE0 to PE15
PF	IO	PF0 to PF5

5.3.2 I/O Port Control Registers

Each GPIO controls up to sixteen I/Os by using two 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 / 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. The user 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 be only 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 mapping 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, users must disable the corresponding analog module.
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 33 describes the GPIO alternate functions.

Table 33. GPIO Alternate Function [1]

Pin Name	Alternate Function						
	AF0	AF1	AF2	AF3	AF4	AF5	AF7
PA0	PA0	MOSI1					AN0*
PA1	PA1	MISO1					AN1/AIP0*
PA2	PA2						AN2/AIN0*
PA3	PA3						AN3/AO0*
PA4	PA4						AN4/AIP1*
PA5	PA5						AN5/AIN1*
PA6	PA6						AN6/AO1*
PA7	PA7						AN7/DAO0*
PA8	PA8						AN8/DAO1*
PA9	PA9					SEG0	AN9*
PA10	PA10					SEG1	AN10*
PA11	PA11					SEG2	AN11*
PA12	PA12	SCL1				SEG3	AN12*
PA13	PA13	SDA1				SEG4	AN13*
PA14	PA14	SS0	T6IO	WDTO		SEG5	AN14*
PA15	PA15	SCK0		RXD0		SEG6	AN15*
PB0	PB0	MOSI0		TXD0		SEG7	AN16*
PB1	PB1	MISO0				COM0	*
PB2	PB2	SCL0				COM1	*
PB3	PB3	SDA0				COM2	*
PB4	PB4					COM3	*
PB5	PB5					CAPH	*
PB6	PB6					CAPL	*
PB7	PB7					VLC3	*
PB8	PB8					VLC2	*
PB9	PB9					VLC1	*
PB10	PB10					VLC0	*
PB11	PB11		T7IO	STBYO		COM4/SEG8	*
PB12	PB12			CLKO		COM5/SEG9	*
PB13	PB13	SCL1			SS10	COM6/SEG10	*
PB14	PB14	SDA1		LPDE0	SCK10	COM7/SEG11	*
PB15	PB15	TXD10		LPTXD0	MOSI10	SEG12	*
PC0	PC0	RXD10		LPRXD0	MISO10	SEG13	*
PC1	PC1	SCL0			SS11	SEG14	*
PC2	PC2	SDA0	TDO/SWO*	LPDE1	SCK11	SEG15	
PC3	PC3	TXD11	TDI*	LPTXD1	MOSI11	SEG16	
PC4	PC4	RXD11	nTRST*	LPRXD1	MISO11	SEG17	
PC5	PC5	RXD0	TCK/SWCLK*				
PC6	PC6	TXD0	TMS/ SWDIO*				
PC7	PC7		RTCC_OUT	RTCC_TAMP1		SEG18	*

Table 33. GPIO Alternate Function (1) (continued)

Pin Name	Alternate Function						
	AF0	AF1	AF2	AF3	AF4	AF5	AF7
PC8	PC8	SCL1		RTCC_TAMP2	SS11	SEG19	*
PC9	PC9	SDA1		LPDE1	SCK11	SEG20	*
PC10	PC10	TXD11	T0IO	LPTXD1	MOSI11	SEG21	*
PC11	PC11	RXD11	T1IO	LPRXD1	MISO11	SEG22	*
PC12	PC12		nRESET*				
PC1	PC13		nBOOT*				
PC14	PC14				USBDP		*
PC15	PC15				USBDM		*
PD0	PD0	SOF		RXD0			*
PD1	PD1			TXD0			*
PD2	PD2						XOUT*
PD3	PD3						XIN*
PD4	PD4	SS1					SXIN*
PD5	PD5	SCK1					SXOUT*
PD6	PD6						AN17*
PD7	PD7						AN18*
PD8	PD8						AN19*
PD9	PD9						AN20*
PD10	PD10						AN21*
PD11	PD11					SEG23	AN22*
PD12	PD12					SEG24	AN23*
PD13	PD13					SEG25	*
PD14	PD14					SEG26	*
PD15	PD15					SEG27	*
PE0	PE0					SEG28	*
PE1	PE1		T5IO			SEG29	*
PE2	PE2					SEG30	*
PE3	PE3					SEG31	*
PE4	PE4	SS1	T5IO			SEG32	*
PE5	PE5	SCK1	T6IO			SEG33	*
PE6	PE6	MOSI1	T7IO	RXD0		SEG34	*
PE7	PE7	MISO1		TXD0		SEG35	*
PE8	PE8					SEG36	*
PE9	PE9		T0IO	AD0S	SS10	SEG37	*
PE10	PE10		T1IO	AD0E	SCK10	SEG38	*
PE11	PE11	TXD10	T2IO	AD1S	MOSI10	SEG39	*
PE12	PE12	RXD10	T3IO	AD1E	MISO10	SEG40	*
PE13	PE13		T2IO			SEG41	*
PE14	PE14		T3IO			SEG42	*
PE15	PE15		T4IO			SEG43	*

Table 33. GPIO Alternate Function (1) (continued)

Pin Name	Alternate Function						
	AF0	AF1	AF2	AF3	AF4	AF5	AF7
PF0	PF0	SCL0		CLKO			*
PF1	PF1	SDA0	T0IO				*
PF2	PF2	SS0	T1IO				*
PF3	PF3	SCK0	T2IO	LPDE0			*
PF4	PF4	MOSI0	T3IO	LPTXD0			*
PF5	PF5	MISO0		LPRXD0			*

NOTES:

1. Unused pins are set to output from firmware (low output is recommended).
2. An * indicates that all the pins are configured to function defined by their initial values after reset. The initial value of the pin depends on the package type. This configuration is in compliance with the 100-pin standard.
3. Ports PB5, PB6, PB7, PB8, PB9, and PB10 support the internal pull-down resistor function, but the internal pull-up resistor and port strength functions are not supported. Please take care when using these ports.

5.3.6 Interrupt Functionality

The GPIO module has six 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 34. 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[15:0]
GPIOD	Port D[15:0]
GPIOE	Port E[15:0]
GPIOF	Port F[5:0]

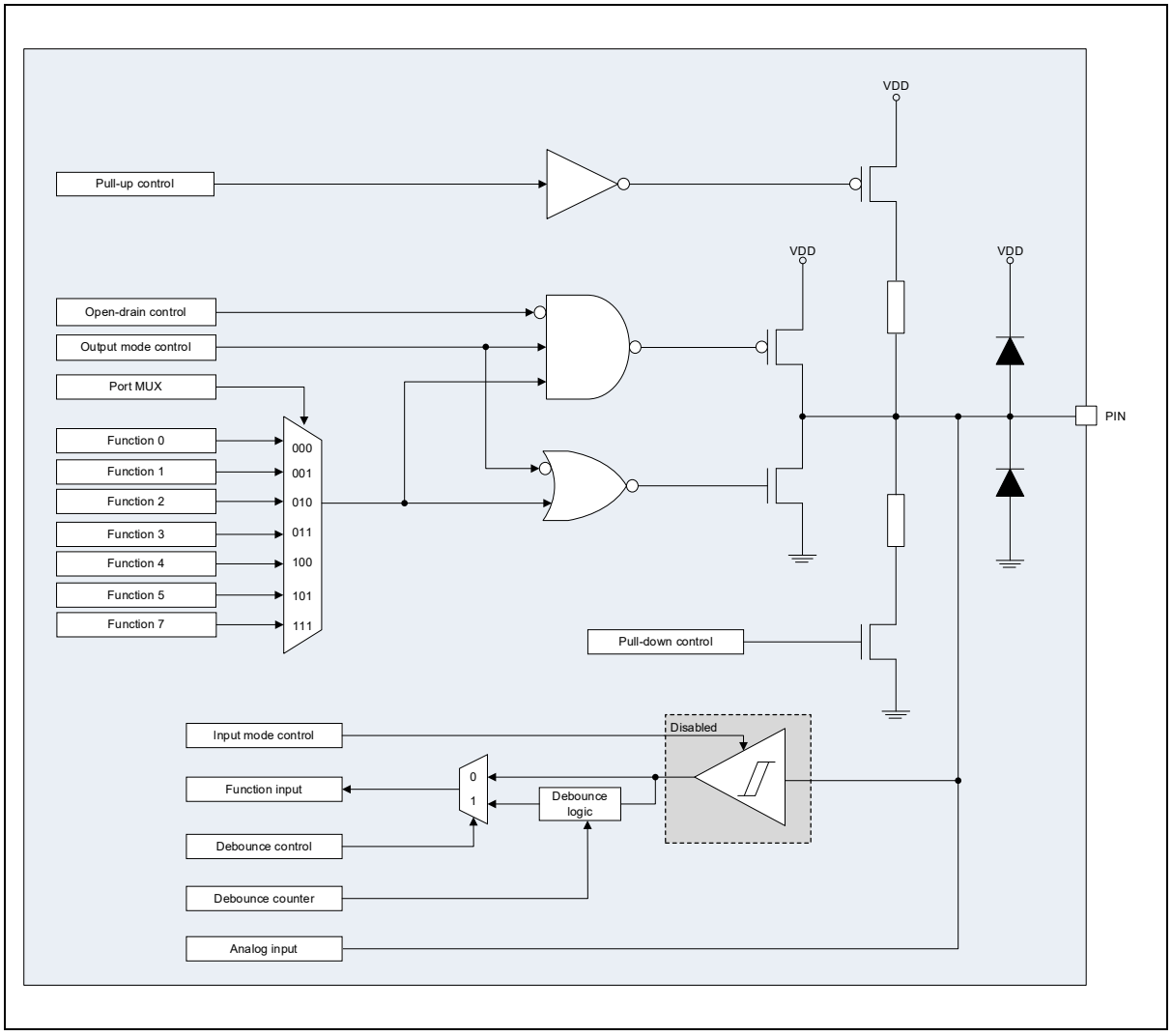
5.3.7 Output Configuration

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 35 shows a block diagram of an output port.

Figure 35. Output Port Block Diagram



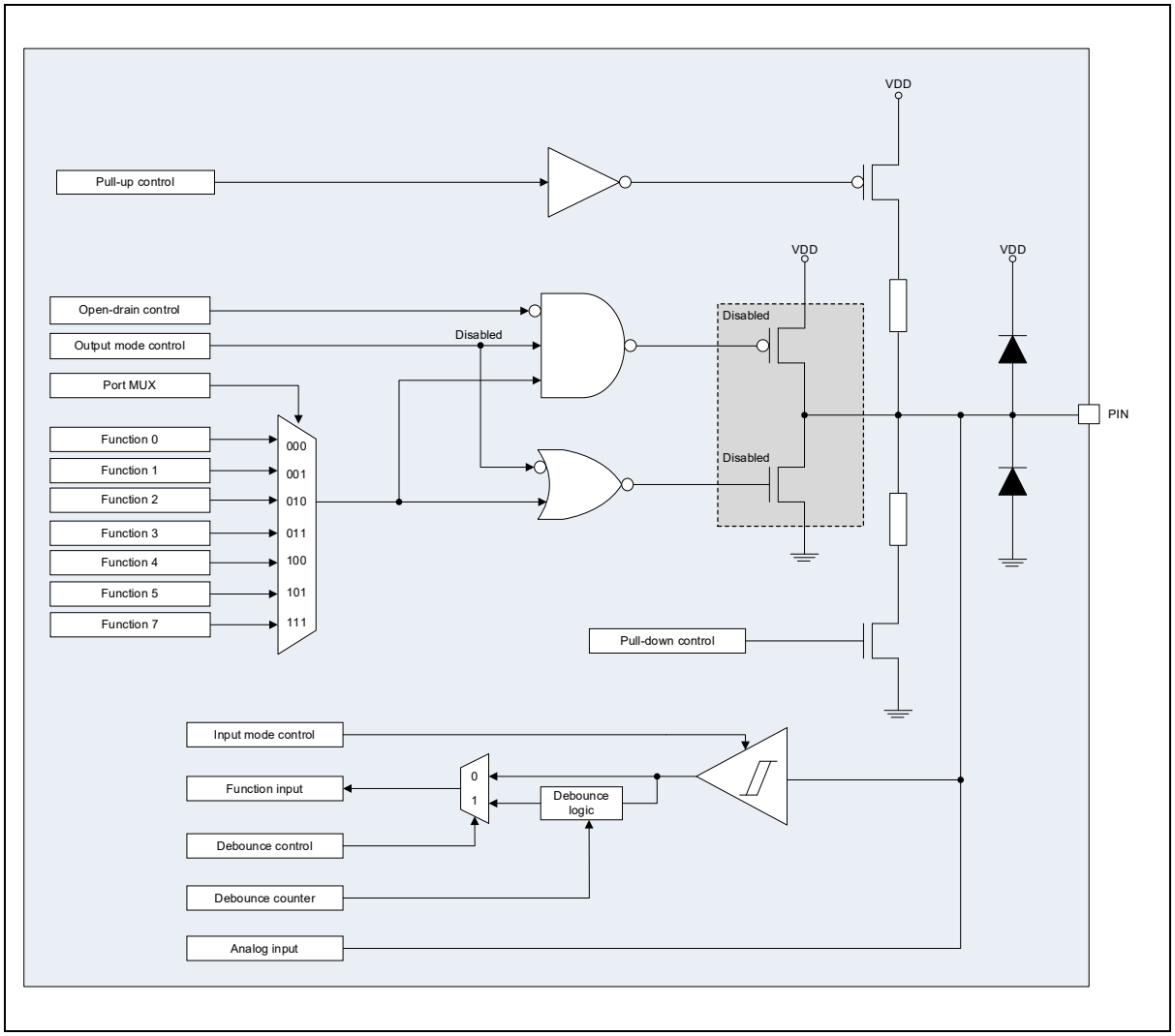
5.3.8 Input Configuration

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 36 shows a block diagram of an input port.

Figure 36. Input Port Block Diagram



5.3.9 Alternate Function Configuration

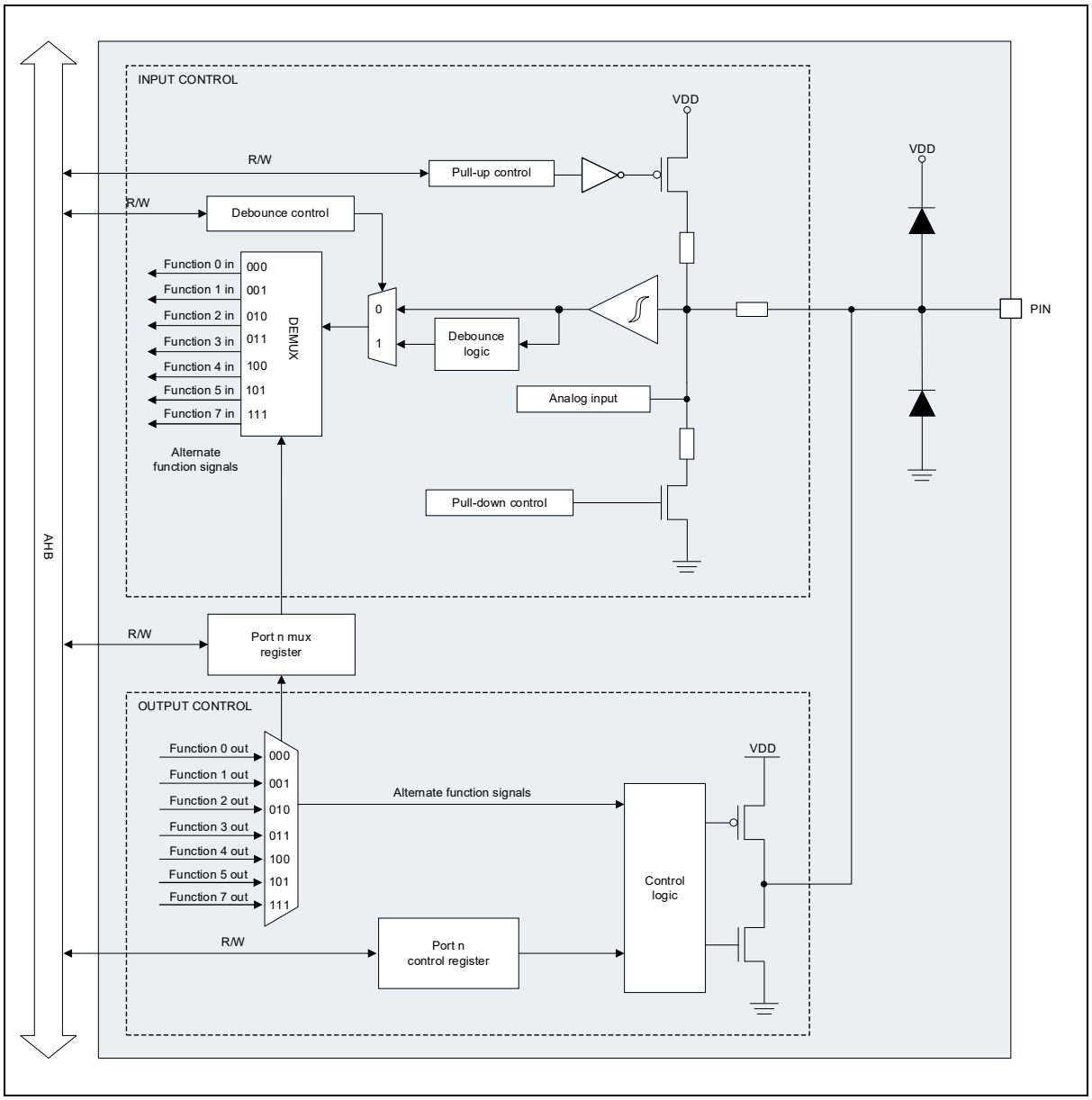
For the A34L716, all pins except those with specific purposes can be used as GPIO pins, and the GPIO function can be set by the pin mux control register on each port.

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's pins 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 37 shows a block diagram that describes alternate function selection on a port.

Figure 37. Port Function Block Diagram

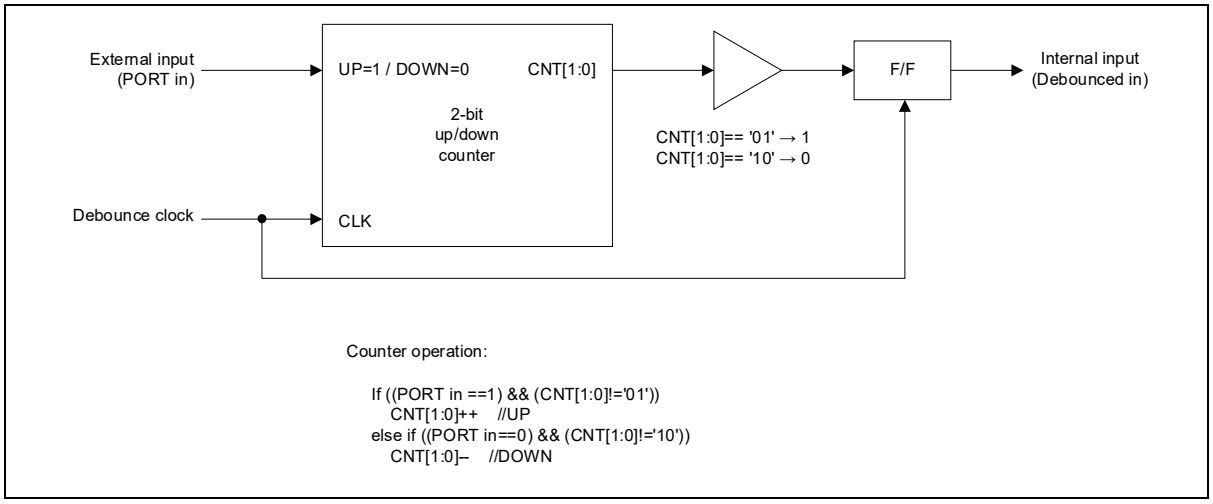


5.3.10 Debouncing Functionality

For the A34L716, 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 38 shows a block diagram of simplified debouncing logic.

Figure 38. Debouncing Logic Block Diagram

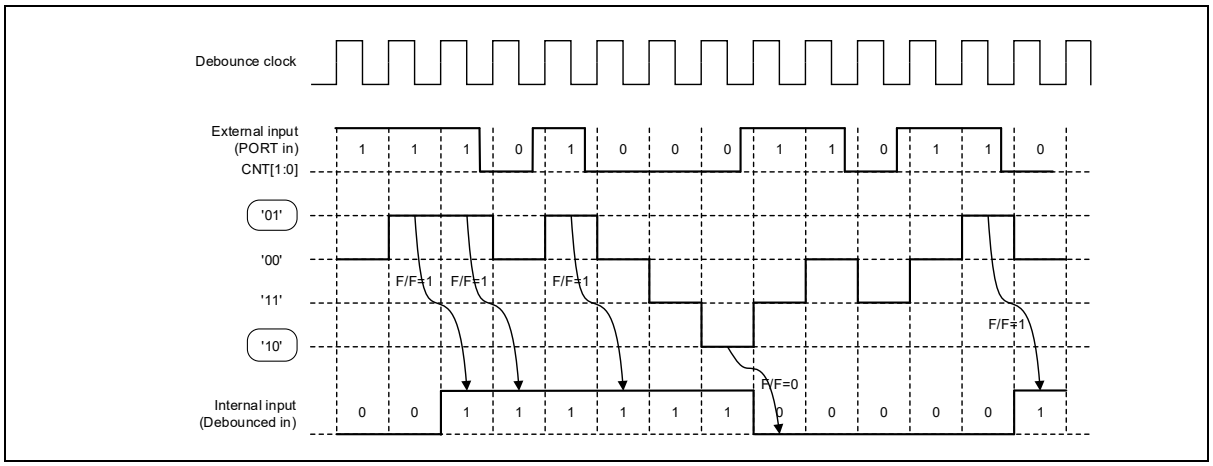


When enabled, the debouncing logic is clocked at the frequency division ratio set in the SCU_MCCR4 or SCU_MCCR5 registers, depending on which register the port belongs to.

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 39 shows a timing diagram of a port debouncing.

Figure 39. Example Timing Diagram of Port Debouncing



5.4 GPIO Registers

The base addresses and register map for the PCU are described in the tables below:

Table 35. Base Address of PCU

Name	Base Address	Description
PA	0x4100_4000	General port A
PB	0x4100_4100	General port B
PC	0x4100_4200	General port C
PD	0x4100_4300	General port D
PE	0x4100_4400	General port E
PF	0x4100_4500	General port F

Table 36. PCU and GPIO Register Map

Name	Offset	Type	Description	Reset value	Reference
Pn_MR1	0x0000	RW	Port n MUX1 select register	0x7777_7777 PC: 0x7222_2277	5.4.1
Pn_MR2	0x0004	RW	Port n MUX2 select register	0x7777_7777 PC: 0x7722_7777	5.4.2
Pn_CR	0x0008	RW	Port n control register	0xFFFF_FFFF	5.4.3
Pn_PRCR	0x000C	RW	Port n pull-up/pull-down resistor control register	0x0000_0000 PC: 0x0A00_2A80	5.4.4
Pn_DER	0x0010	RW	Port n debouncing enable register	0x0000_0000	5.4.5
Pn_STR	0x0014	RW	Port n strength configuration register	0x0000_0000	5.4.6
Pn_IER	0x0020	RW	Port n interrupt enable register	0x0000_0000	5.4.7
Pn_ISR	0x0024	RWC1	Port n interrupt status register	0x0000_0000	5.4.8
Pn_ICR	0x0028	RW	Port n interrupt control register	0x0000_0000	5.4.9
Pn_ODR	0x0030	RW	Port n output data register	0x0000_0000	5.4.10
Pn_IDR	0x0034	RO	Port n input data register	0xFFFF_FFFF PC: 0x0000_FFFB PF: 0x0000_003F	5.4.11
Pn_BSR	0x0038	WO	Port n set / reset register	0x0000_0000	5.4.12
Pn_BCR	0x003C	WO	Port n reset register	0x0000_0000	5.4.13
PCU_PORTEN	0x0FF0	RW	Port access enable register	0x0000_0000	5.4.14

NOTE:

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

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=0x4100_4000, PB_MR1=0x4100_4100
PC_MR1=0x4100_4200, PD_MR1=0x4100_4300
PE_MR1=0x4100_4400, PF_MR1=0x4100_4500

31	30 29 28	27	26 25 24	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	P7MUX[2:0]	Reserved	P6MUX[2:0]	Reserved	P5MUX[2:0]	Reserved	P4MUX[2:0]	Reserved	P3MUX[2:0]	Reserved	P2MUX[2:0]	Reserved	P1MUX[2:0]	Reserved	P0MUX[2:0]
-	111	-	111	-	111	-	111	-	111	-	111	-	111	-	111
-	RW	-	RW	-	RW	-	RW	-	RW	-	RW	-	RW	-	RW

4x+2 4x	PxMUX[2:0]	Port MUX 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)
		101	Alternate function 5 (AF5)
		111	Alternate function 7 (AF7)
		Others	Reserved

NOTES:

1. The PxMUX[2:0] (x = 0 to 15) bits of PD2, PD3, PD4 and PD5 cannot be modified while the HSE or 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 setting, the I/O is automatically configured as an input or output of that function.)
3. When pin multiplexing is set to AF1, AF2, AF3, AF4, AF5 and AF7 it cannot be set to open-drain output or input even if users use 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 PORTEN register is activated.

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=0x4100_4004, PB_MR2=0x4100_4104
PC_MR2=0x4100_4204, PD_MR2=0x4100_4304
PE_MR2=0x4100_4404, PD_MR2=0x4100_4504

31	30 29 28	27	26 25 24	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	P15MUX[2:0]	Reserved	P14MUX[2:0]	Reserved	P13MUX[2:0]	Reserved	P12MUX[2:0]	Reserved	P11MUX[2:0]	Reserved	P10MUX[2:0]	Reserved	P9MUX[2:0]	Reserved	P8MUX[2:0]
-	111	-	111	-	111	-	111	-	111	-	111	-	111	-	111
-	RW	-	RW	-	RW	-	RW	-	RW	-	RW	-	RW	-	RW

4×(x-8)+2 4×(x-8)	PxMUX[2:0]	Port muxing selection bit, x = 8 to15	
		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)
		101	Alternate function 5 (AF5)
		111	Alternate function 7 (AF7)
		Others	Reserved

NOTES:

1. The PxMUX[2:0] (x = 0 to 15) bits of PD2, PD3, PD4 and PD5 cannot be modified while the HSE or 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 setting, the I/O is automatically configured as an input or output of that function.)
3. When pin multiplexing is set to AF1, AF2, AF3, AF4, AF5 and AF7 it cannot be set to open-drain output or input even if users use 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 PORTEN register is activated.

5.4.3 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=0x4100_4008, PB_CR=0x4100_4108
PC_CR=0x4100_4208, PD_CR=0x4100_4308
PE_CR=0x4100_4408, PF_CR=0x4100_4508

31 30	29 28	27 26	25 24	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
		11	Input

5.4.4 Pn_PRCR: Port n Pull-Up/Pull-Down Control Register

All pins of each port include internal pull-up/ 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=0x4100_400C, PB_PRCR=0x4100_410C
PC_PRCR=0x4100_420C, PD_PRCR=0x4100_430C
PE_PRCR=0x4100_440C, PF_PRCR=0x4100_450C

31 30	29 28	27 26	25 24	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/ pull-down control, x = 0 to 15	
		0x	Disables the pull-up / pull-down resistors.
		10	Enables the pull-up resistor / disables the pull-down resistor.
		11	Disables the pull-up resistor / enables the pull-down resistor.

NOTES:

1. Pn_DER, Pn_STR, Pn_IER, Pn_ISR, Pn_ICR, Pn_ODR, Pn_IDR, Pn_BSR and Pn_BCR registers can change the value of the register without activating PORTEN[7:0].
2. Exceptionally, the reset value of PC_PRCR is 0x0A00_2A80.
3. Ports PB5, PB6, PB7, PB8, PB9, and PB10 support the internal pull-down resistor function, but the internal pull-up resistor and port strength functions are not supported. Please take care when using these ports.

5.4.5 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=0x4100_4010, PB_DER=0x4100_4110
PC_DER=0x4100_4210, PD_DER=0x4100_4310
PE_DER=0x4100_4410, PF_DER=0x4100_4510

31 30 29 28 27 26 25 24	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	Whether to enable or disable 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 and Pn_BCR registers can change the value of the register without activating PORTEN.

5.4.6 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=0x4100_4014, PB_STR=0x4100_4114
PC_STR=0x4100_4214, PD_STR=0x4100_4314
PE_STR=0x4400_4414, PF_STR=0x4100_4514

31 30 29 28 27 26 25 24	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 = 0–15
		0 Disables the strength.
		1 Enables the strength.

NOTE:

1. Pn_DER, Pn_STR, Pn_IER, Pn_ISR, Pn_ICR, Pn_ODR, Pn_IDR, Pn_BSR and Pn_BCR registers can change the value of the register without activating PORTEN.
2. Ports PB5, PB6, PB7, PB8, PB9, and PB10 support the internal pull-down resistor function, but the internal pull-up resistor and port strength functions are not supported. Please take care when using these ports.

5.4.7 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=0x4100_4020, PB_IER=0x4100_4120
PC_IER=0x4100_4220, PD_IER=0x4100_4320
PE_IER=0x4100_4420, PF_IER=0x4100_4520

31 30	29 28	27 26	25 24	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.
2. 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.
3. In pending mode, it is possible to check whether an interrupt has occurred, but not the current state of the pin. In high level pending interrupt mode, an interrupt signal is generated when the pin state is high.
4. 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.
5. Pn_DER, Pn_STR, Pn_IER, Pn_ISR, Pn_ICR, Pn_ODR, Pn_IDR, Pn_BSR and Pn_BCR registers can change the value of the register without activating PORTEN.

5.4.8 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=0x4100_4024, PB_ISR=0x4100_4124
PC_ISR=0x4100_4224, PD_ISR=0x4100_4324
PE_ISR=0x4100_4424, PF_ISR=0x4100_4524

31 30	29 28	27 26	25 24	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.

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 and Pn_BCR registers can change the value of the register without activating PORTEN[7:0].

5.4.9 Pn_ICR: Port n Interrupt Control Register

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

PA_ICR=0x4100_4028, PB_ICR=0x4100_4128
PC_ICR=0x4100_4228, PD_ICR=0x4100_4328
PE_ICR=0x4100_4428, PF_ICR=0x4100_4528

31 30	29 28	27 26	25 24	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.

NOTE:

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

5.4.10 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=0x4100_4030, PB_ODR=0x4100_4130
PC_ODR=0x4100_4230, PD_ODR=0x4100_4330
PE_ODR=0x4100_4430, PF_ODR=0x1200_4530

31 30 29 28 27 26 25 24	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 = 0–15	
		0	Outputs a '0' if the port pin is in output mode.
		1	Outputs a '1' if the port pin is in output mode.

NOTE:

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

5.4.11 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 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=0x4100_4034, PB_IDR=0x4100_4134
PC_IDR=0x4100_4234, PD_IDR=0x4100_4334
PE_IDR=0x4100_4434, PD_IDR=0x4100_4534

31 30 29 28 27 26 25 24	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
-		1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1
-		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 current 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 and Pn_BCR registers can change the value of the register without activating PORTEN[7:0].
4. Exceptionally, the reset value of PC_IDR register is 0x0000_FFFB.
   Exceptionally, the reset value of the PF_IDR register is 0x0000_003F.

5.4.12 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=0x4100_4038, PB_BSR=0x4100_4138
PC_BSR=0x4100_4238, PD_BSR=0x4100_4338
PE_BSR=0x4100_4438, PF_BSR=0x4100_4538

31	30	29	28	27	26	25	24	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'.

NOTE:

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

5.4.13 Pn_BCR: Port n Reset Register

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

PA_BSR=0x4100_4038, PB_BSR=0x4100_4138
PC_BSR=0x4100_4238, PD_BSR=0x4100_4338
PE_BSR=0x4100_4438, PF_BSR=0x4100_4538

31	30	29	28	27	26	25	24	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.

NOTE:

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

5.4.14 PCU_PORTEN: Port Access Enable Register

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

PCU_PORTEN=0x4100_4FF0

31 30 29 28 27 26 25 24	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
-			RC	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.

5.4.15 PCU and GPIO Register Map Summary

Table 37. PCU and GPIO Register Map Summary

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
Pn_MR1	Res.	P7MUX[2:0]			Res.	P6MUX[2:0]			Res.	P5MUX[2:0]			Res.	P4MUX[2:0]			Res.	P3MUX[2:0]			Res.	P2MUX[2:0]			Res.	P1MUX[2:0]			Res.	P0MUX[2:0]		
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
Pn_MR2	Res.	P15MUX[2:0]			Res.	P14MUX[2:0]			Res.	P13MUX[2:0]			Res.	P12MUX[2:0]			Res.	P11MUX[2:0]			Res.	P10MUX[2:0]			Res.	P9MUX[2:0]			Res.	P8MUX[2:0]		
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
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
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
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
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
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
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
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
Pn_ODR	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	POD15	POD14	POD13	POD12	POD11	POD10	POD9	POD8	POD7	POD6	POD5	POD4	POD3	POD2	POD1	POD0
Reset value																	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
Pn_IDR	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	PID15	PID14	PID13	PID12	PID11	PID10	PID9	PID8	PID7	PID6	PID5	PID4	PID3	PID2	PID1	PID0
Reset value																	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1

Table 37. PCU and GPIO Register Map Summary (continued)

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
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
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
PCU_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 A34L716 supports the following:

- Erase, read, write functions in code flash areas.
- Read protection function.
- Write / Erase protection function.

6.2 Flash Main Features

The flash memory of A34L716 has several key features as listed below:

- Code flash memory:
  - Flash size: 256 KB
  - Program unit: 1 word (4 bytes)
  - Erase unit:
    - Page (512 bytes)
    - Sector (2 KB)
    - Bulk (chip)
  - Zero-wait (less than 25 MHz), 1 to 15-wait
  - Code flash read protection
  - Code flash write protection
- Data flash memory:
  - Flash size: 32 KB
  - Program unit: 1 word (4 bytes)
  - Erase unit:
    - Page (512 bytes)
    - Sector (2 KB)
    - Bulk (chip)
  - Zero-wait (less than 25 MHz), 1 to 15-wait access
  - Data flash write protection

6.3 Flash Functional Description

6.3.1 Flash Memory Organization

The flash memory is composed of code flash memory and data flash memory. The code flash memory is used to store the application code or constant data, and the data flash memory is used to store parameters and log data.

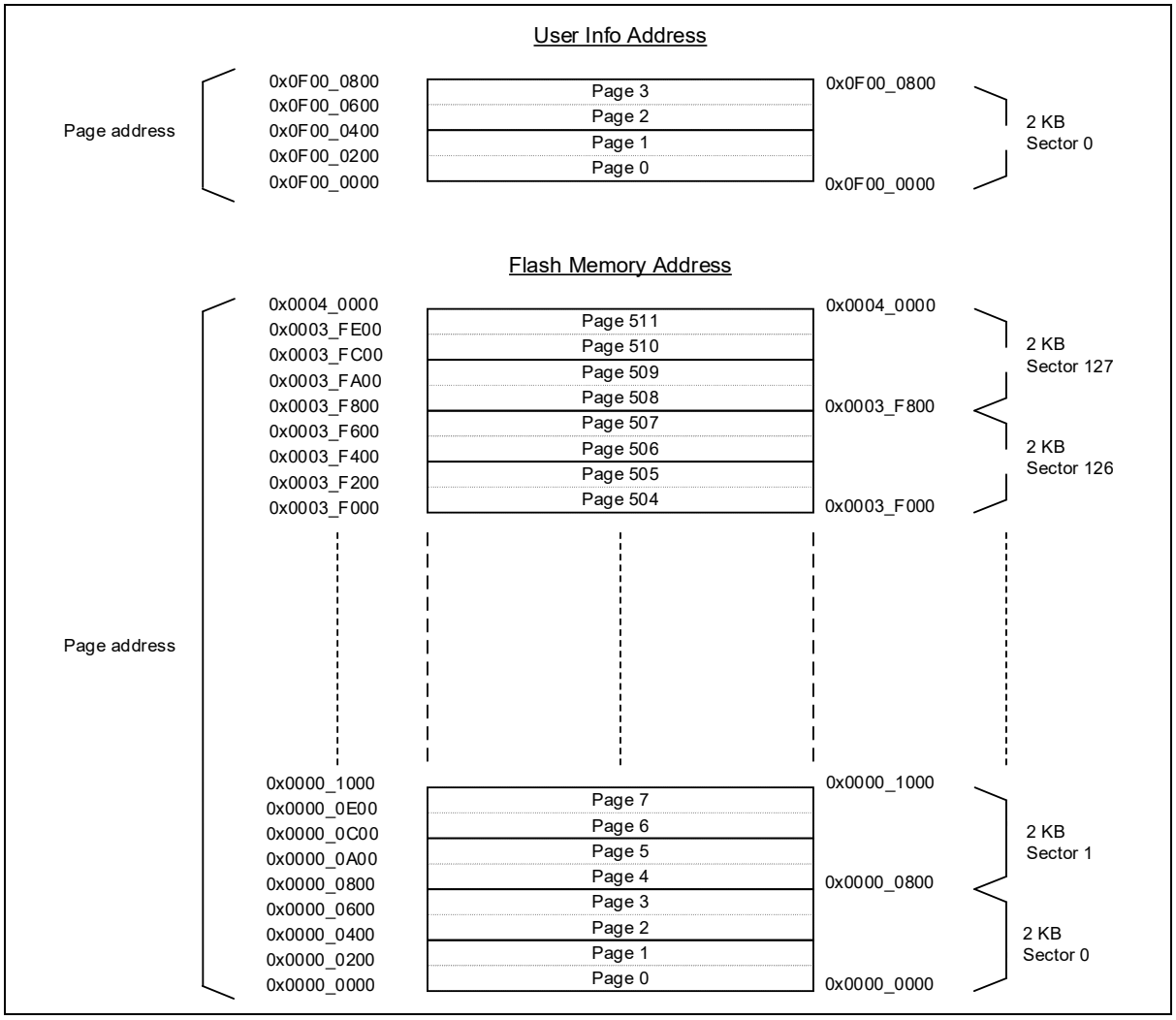
6.3.1.1 Code Flash Memory

Code flash memory has key features as shown below:

Table 38. Code Flash Memory Controller Features

Item	Description
Size	256 KB
Start address	0x0000_0000
End address	0x0003_FFFF
Page size	512 bytes
Total page count	512 pages
Program unit	4 bytes (1 word)
Erase unit	512 bytes, 2 KB, Bulk

Figure 40. Code Flash Memory Map



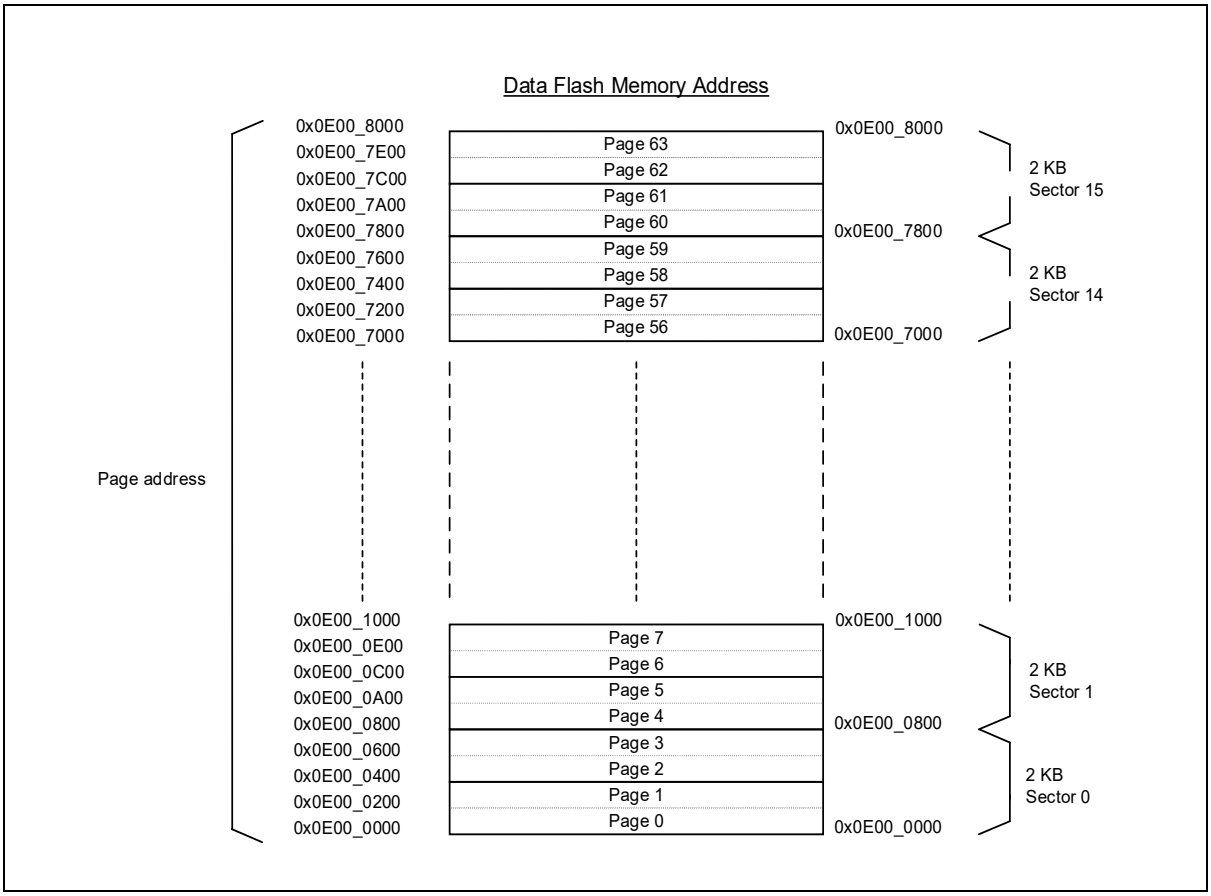
6.3.1.2 Data Flash Memory

Data flash memory has key features as shown below:

Table 39. Data Flash Memory Controller Features

Item	Description
Size	32 KB
Start address	0x0E00_0000
End address	0x0E00_7FFF
Page size	512 bytes
Total page count	64 pages
Program unit	4 bytes (1 word)
Erase unit	512 bytes, 2 KB, Bulk

Figure 41. Data Flash Memory Map



6.3.2 Read Access Latency

Flash memory has a read latency that is a required wait 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 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 40.
2. Program the LATENCY[3:0] in CFMC_CONF register to have a new wait time.

Table 40. Internal Code Flash Memory Wait Time by Operating Clock

LATENCY[3:0]	Flash Wait Time	Max. Available Clock Frequency
0000	0-clock wait	Up to 25 MHz
0001	1-clock wait	Up to 32 MHz
0010	2-clock wait	Up to 32 MHz
0011	3-clock wait	Up to 32 MHz
0100	4-clock wait	Up to 32 MHz
0101	5-clock wait	Up to 32 MHz
0110	6-clock wait	Up to 32 MHz
0111	7-clock wait	Up to 32 MHz
1000	8-clock wait	Up to 32 MHz
1001	9-clock wait	Up to 32 MHz
1010	10-clock wait	Up to 32 MHz
1011	11-clock wait	Up to 32 MHz
1100	12-clock wait	Up to 32 MHz
1101	13-clock wait	Up to 32 MHz
1110	14-clock wait	Up to 32 MHz
1111	15-clock wait	Up to 32 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 Access Time

1. Determine the correct wait time corresponding to the microcontroller’s main clock by referring to Table 41.
2. Program the LATENCY[3:0] in DFMC_CONF register to have a new wait time.

Table 41. Internal Data Flash Wait Time by Operating Clock

LATENCY[3:0]	Flash Wait Time	Max. Available Clock Frequency
0000	0-clock wait	Up to 25 MHz
0001	1-clock wait	Up to 32 MHz
0010	2-clock wait	Up to 32 MHz
0011	3-clock wait	Up to 32 MHz
0100	4-clock wait	Up to 32 MHz
0101	5-clock wait	Up to 32 MHz
0110	6-clock wait	Up to 32 MHz
0111	7-clock wait	Up to 32 MHz
1000	8-clock wait	Up to 32 MHz
1001	9-clock wait	Up to 32 MHz
1010	10-clock wait	Up to 32 MHz
1011	11-clock wait	Up to 32 MHz
1100	12-clock wait	Up to 32 MHz
1101	13-clock wait	Up to 32 MHz
1110	14-clock wait	Up to 32 MHz
1111	15-clock wait	Up to 32 MHz

6.3.3 Flash Program and Erase Operations

It is important to remember that any interrupt must not be triggered for other operations while erasing or programming the flash. The operations below must be set up before starting the flash control.

- All Interrupt disable

For access to the flash memory, users must unlock the flash before starting the flash control, as described in the following section 6.3.3.1.

6.3.3.1 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 code flash key and access control registers. The access control of the flash memory follows the steps below:

1. To enable access of the flash memory, enter key values in the CFMC_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 enable access to the code flash memory, set the FLOCK in CFMC_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 programming or erasing.

6.3.3.2 Access Control for Writing and Erasing User Info Memory

Before writing and erasing data on use info memory, an access enable process must be executed by setting the User info key and access control registers. The access control of the User info follows the steps below:

1. Enter key values in the CFMC_INFOKEY 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. Set the INFOLOCK in CFMC_CTRL register to ‘1’.

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

The info memory erase operation must be proceeded before starting the info memory programming or erasing.

6.3.4 Flash Memory Erase Sequences

6.3.4.1 Page Erase

Figure 42 shows a timing diagram for page erase operations in the flash memory.

Figure 42. Flash Memory Page Erase Timing Diagram

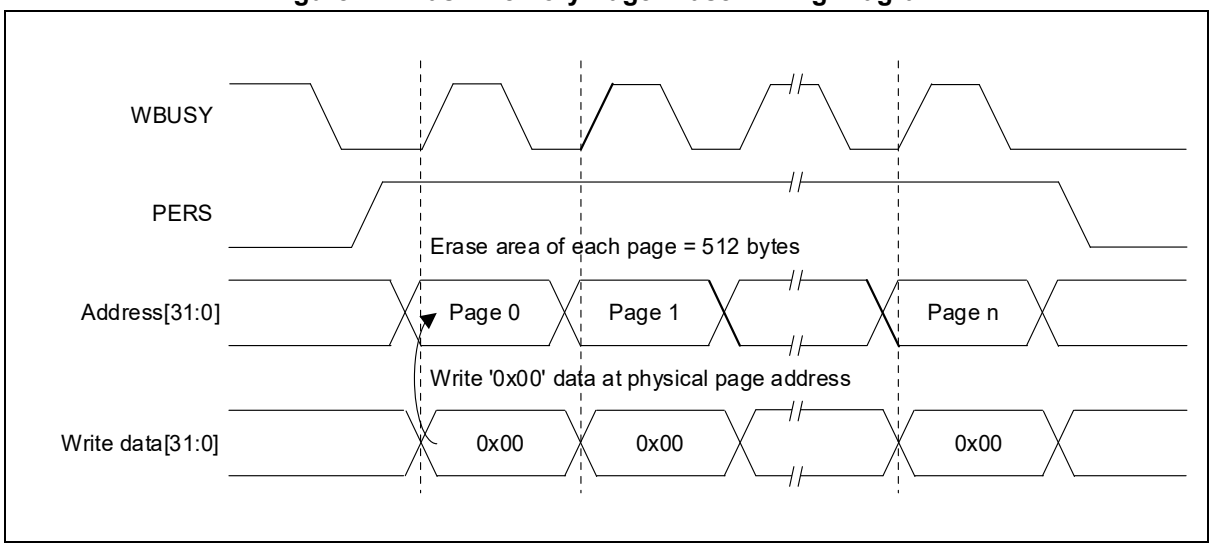


Figure 42 describes the following procedure for the page erase:

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

6.3.4.2 2 KB Sector Erase

Figure 43 shows a timing diagram for 2 KB sector erase operations in the flash memory.

Figure 43. Flash Memory 2 KB Sector Erase Timing Diagram

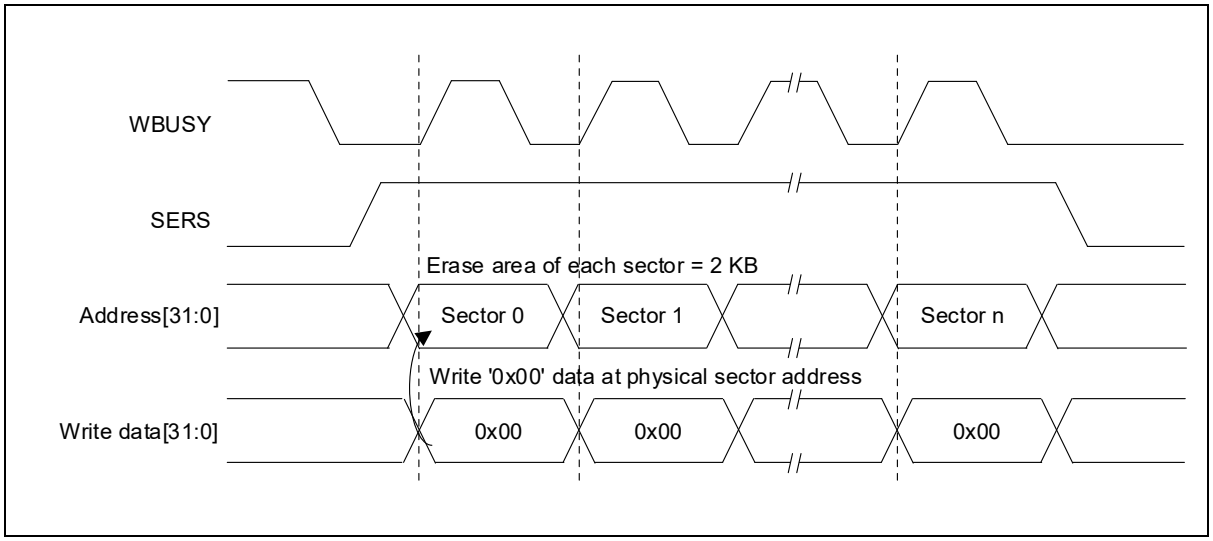


Figure 43 describes the following procedure for the 2 KB sector erase:

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

6.3.4.3 Bulk Erase

Figure 44 shows a timing diagram for bulk (full-chip) erase operations in the flash memory.

Figure 44. Flash Memory Bulk (Full-Chip) Erase Timing Diagram

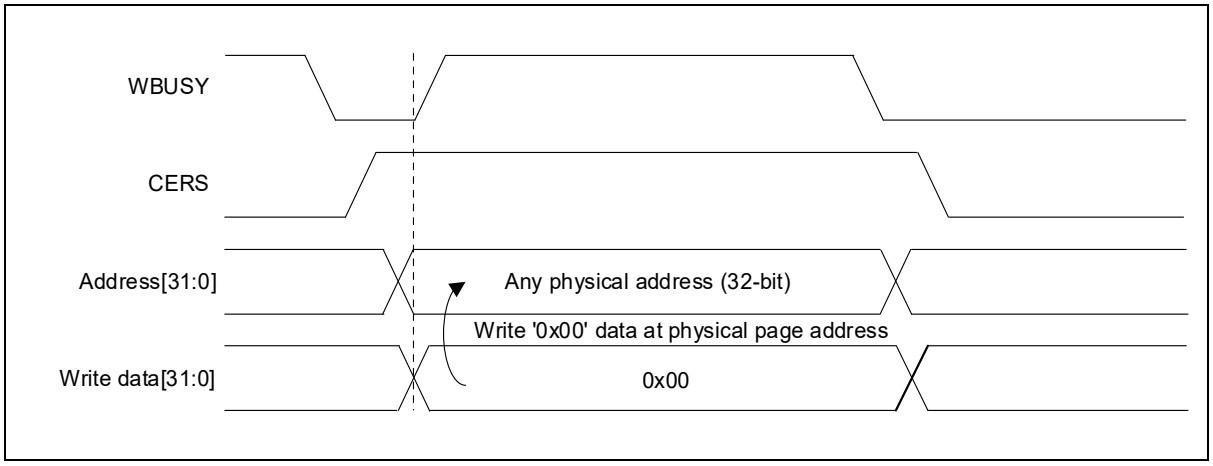


Figure 44 describes the following procedure for the bulk erase.

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

6.3.5 Flash Memory Programming Sequences

Before writing to the flash memory, it is important to flash unlock and unprotect the area. Remember that Lock, Protection, must be disabled during the writing is complete.

6.3.5.1 Standard Programming

Figure 45. Flash Memory Program Timing Diagram

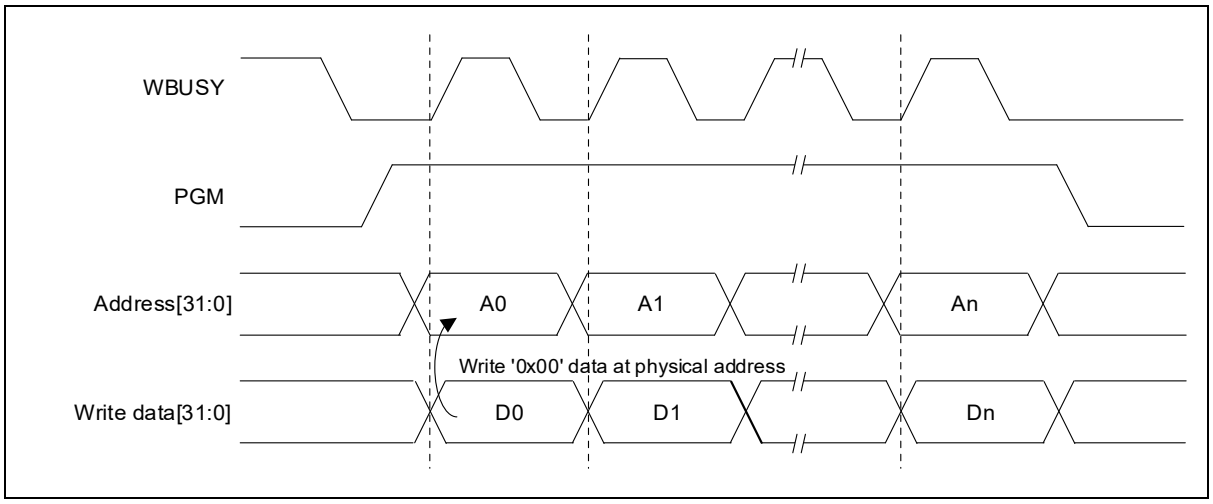


Figure 45 describes the following procedure for standard programming:

1. Check the WBUSY in CFMC_STAT (or DFMC_STAT) register.
2. Set the PGM in CFMC_CTRL (or DFMC_CTRL) register.
3. Write target data at physical address.
4. Check the WBUSY in CFMC_STAT (or DFMC_STAT) register at polling mode or the WDONE in CFMC_STAT (or DFMC_STAT) register at interrupt mode.
5. Clear the PGM in CFMC_CTRL (or DFMC_CTRL) register.

6.4 Flash Memory Protection

Read protection is a feature provided in the internal code flash 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 42 describes three different protection mode levels of the read protection, such as UNPROT, LVL1, and LVL2.

Table 42. Available Operating Modes by Protection Level

Protection Level	Operation Mode	Main (Code Flash)		User Info 0		User Info 1 / 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	N/A	O	O	O	O
	Debug mode	N/A	N/A	N/A	N/A	N/A	N/A	N/A	N/A
	Boot mode	N/A	N/A	N/A	N/A	N/A	N/A	N/A	N/A
LVL2	Normal mode	O	O	O	N/A	O	O	O	O
	Debug mode	This mode cannot be entered.							
	Boot mode	N/A	N/A	N/A	N/A	N/A	N/A	N/A	N/A

NOTES:

1. Priority levels of the read protection mode are as follows:
   Normal protection > 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 chip erase and INFO0 erase executed.
   E.g.) Procedures for transitioning from read protection LVL1 to UNPROT.
   A. Chip erase
   B. Check the CERSD in CFMC_READPROT register.
   C. INFO0 erase
   D. Check the INFO0ERSD in CFMC_READPROT register.
   E. Write '0xFF' to the RPROT[7:0] in 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 43. 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	N/A	N/A	N/A	N/A
	DMA	N/A	N/A	R / W	N/A
	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 be run while accessing the internal flash for debugging, and the flash read with DMA cannot be supported because the DMA settings can cause code leakage.

Table 44. Read Protection Operation in Level-1 (LVL 1)

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	N/A	N/A	N/A	N/A
	DMA	N/A	N/A	R / W	N/A
	Debug	N/A	N/A	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 read with DMA cannot be supported because the DMA settings can cause code leakage.

Table 45. Read Protection Operation in Level-2 (LVL 2)

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	N/A	N/A	N/A	N/A
	DMA	N/A	N/A	R / W	N/A
	Debug	N/A	N/A	N/A	N/A

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 can write data to the read protection area of the User info, and then the Boot ROM reads the data to enable the read protection when it runs on reset. Once read protection is enabled, users must reset the 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 allows changing the read protection mode level from low to high, but changes from high to low levels are 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 info. To delete the data of the User info in the read protection mode, the 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 CFMC_PWPRST register. If the input value of CFMC_PWIN register matches the preset value of 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 = 0x1234_5678;
- CFMC_PWPRST = 0x1234_5678;

6.4.2 Write Protection

Users can set up the Write and Erase Protection for a certain area of the code flash and data flash. Before setting up the protections, remember the following:

- Protection of bottom 32KB is set by the FPBY16K_0 bit or the FUBLP4K_y bit.
- If any of the FPBY16_z or the FUBLP4K_y in CFMC_FLSPROT 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 protection (FUP512B_x in CFMC_FLSPROT register is '0') when the FPBY16_15 bit is '1', the erase commands such as '2 KB Sector Erase' and 'Bulk Erase' do not work for the area that includes protection area.

6.4.2.1 Write Protection for Code Flash Memory

The CFMC_FLSPROT register is used to set the Protection and Unprotection as described in Table 46 and Table 47, Table 48.

Table 46. Code Flash Memory Write Protection Area in 16 KB Units

z	Address	z	Address
0	0x0000_0000 ~ 0x0000_3FFF	1	0x0000_4000 ~ 0x0000_7FFF
2	0x0000_8000 ~ 0x0000_BFFF	3	0x0000_C000 ~ 0x0000_FFFF
4	0x0001_0000 ~ 0x0001_3FFF	5	0x0001_4000 ~ 0x0001_7FFF
6	0x0001_8000 ~ 0x0001_BFFF	7	0x0001_C000 ~ 0x0001_FFFF
8	0x0002_0000 ~ 0x0002_3FFF	9	0x0002_4000 ~ 0x0002_7FFF
10	0x0002_8000 ~ 0x0002_BFFF	11	0x0002_C000 ~ 0x0002_FFFF
12	0x0003_0000 ~ 0x0003_3FFF	13	0x0003_4000 ~ 0x0003_7FFF
14	0x0003_8000 ~ 0x0003_BFFF	15	0x0003_C000 ~ 0x0003_FFFF

Table 47. Code Flash 0 User Boot Loader Protection Area (Bottom 32 KB) in 4 KB Units

y	Address	y	Address
0	0x0000_0000 ~ 0x0000_0FFF	1	0x0000_1000 ~ 0x0000_1FFF
2	0x0000_2000 ~ 0x0000_2FFF	3	0x0000_3000 ~ 0x0000_3FFF
4	0x0000_4000 ~ 0x0000_4FFF	5	0x0000_5000 ~ 0x0000_5FFF
6	0x0000_6000 ~ 0x0000_6FFF	7	0x0000_7000 ~ 0x0000_7FFF

Table 48. Code Flash Memory Unprotection Area (Top 4 KB) in 512 Bytes Units

x	Address	x	Address
0	0x0003_F000 ~ 0x0003_F1FF	1	0x0003_F200 ~ 0x0003_F3FF
2	0x0003_F400 ~ 0x0003_F5FF	3	0x0003_F600 ~ 0x0003_F7FF
4	0x0003_F800 ~ 0x0003_F9FF	5	0x0003_FA00 ~ 0x0003_FBFF
6	0x0003_FC00 ~ 0x0003_FDFF	7	0x0003_FE00 ~ 0x0003_FFFF

6.4.2.2 Write Protection for Data Flash

The DFMC_FLSPROT register is used to set the Protection and Unprotection as described in Table 49 and Table 50.

Table 49. Data Flash Protection Area in 2 KB Units

y	Address	y	Address
0	0x0E00_0000 ~ 0x0E00_07FF	1	0x0E00_0800 ~ 0x0E00_0FFF
2	0x0E00_1000 ~ 0x0E01_17FF	3	0x0E00_1800 ~ 0x0E01_1FFF
4	0x0E00_2000 ~ 0x0E02_27FF	5	0x0E00_2800 ~ 0x0E02_2FFF
6	0x0E00_3000 ~ 0x0E03_37FF	7	0x0E00_3800 ~ 0x0E03_3FFF
8	0x0E00_4000 ~ 0x0E04_47FF	9	0x0E00_4800 ~ 0x0E04_4FFF
10	0x0E00_5000 ~ 0x0E05_57FF	11	0x0E00_5800 ~ 0x0E05_5FFF
12	0x0E00_6000 ~ 0x0E06_67FF	13	0x0E00_6800 ~ 0x0E06_6FFF
14	0x0E00_7000 ~ 0x0E07_77FF	15	0x0E00_7800 ~ 0x0E07_7FFF

Table 50. Data Flash 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

6.4.3 Securable Memory Area (LVL1, LVL2)

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

6.4.4 Disabling Core Debug Access (LVL2)

Using the read protection, users can block the debugger connections (see section 6.4.1).

6.5 Flash Interrupts

For the code flash control, A34L716 supports the write done interrupt. This interrupt can be activated by setting the WDIEN in CFMC_CTRL and DFMC_CTRL register to ‘1’.

6.6 Code Flash Registers

The base addresses and register map for the Code Flash Memory Controller (CFMC) are described in the tables below:

Table 51. Base Address of CFMC

Name	Base Address
CFMC	0x4100_0000

Table 52. CFMC Register Map

Name	Offset	Type	Description	Reset Value	Reference
CFMC_CONF	0x0000	RW	Code flash control register	0x0000_0000	6.6.1
CFMC_FLSKEY	0x0004	WO	Code flash access key register	0x0000_0000	6.6.2
CFMC_INFOKEY	0x0008	WO	Code flash User info access key register	0x0000_0000	6.6.3
CFMC_FLSPROT	0x000C	RW	Code flash protection register	0x0000_0000	6.6.4
CFMC_INFOPROT	0x0014	RW	Code flash User info protection register	0x0000_0000	6.6.5
CFMC_CTRL	0x0018	RW	Code flash access control register	0xC000_0000	6.6.6
CFMC_STAT	0x001C	RW	Code flash access status register	0x0000_0000	6.6.7
CFMC_READPROT	0x0020	RW	Code flash read protection register	0x0000_00FF	6.6.8
CFMC_PWIN	0x0024	WO	Code flash password input register	0x0000_0000	6.6.9
CFMC_WTIMEOUT	0x0040	RW	Code flash write timeout register	0x0000_0000	6.6.10
CFMC_ECCEADDR	0x0060	RO	Code flash ECC Error Address Register	0x0000_0000	6.6.11
CFMC_ECCEDATA0	0x0064	RO	Code flash ECC Error Data0 Register	0x0000_0000	6.6.12
CFMC_ECCEDATA1	0x0068	RO	Code flash ECC Error Data1 Register	0x0000_0000	6.6.13
CFMC_ECCEDATA2	0x006C	RO	Code flash ECC Error Data2 Register	0x0000_0000	6.6.14
CFMC_ECCEDATA3	0x0070	RO	Code flash ECC Error Data3 Register	0x0000_0000	6.6.15
CFMC_ECCEPARITY	0x0074	RO	Code flash ECC Error Parity Register	0x0000_0000	6.6.16
CFMC_PWPRST	0x0F30	WO	Code flash password preset register	0x0000_0000	6.6.17

6.6.1 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
ECCDIS	Reserved							WRITE	Reserved																			LATENCY[3:0]			
0	-							0	-																			0000			
RW	-							RW	-																			RW			

31	ECCDIS	Disable code flash ECC	
		0	ECC enable
		1	ECC disable
25	WRITE	Code flash write enable bit	
		0	Disables.
		1	Enables.
3 0	LATENCY[3:0][1]	Flash wait value bits	
		Flash wait values between 0 and 15.	

NOTE:

1. For detailed information of the LATENCY[3:0], Refer to section 6.3.2.

6.6.2 CFMC_FLSKEY: Code Flash Access Key Register

The CFMC_FLSKEY register is the access key register for internal flash memory. This register is 32 bits 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 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. The FLOCK bit in the CFMC_CTRL register must be unlocked before controlling the flash memory. To do this, the KEY values must be written to the CFMC_FLSKEY register in the order 'KEY1 -> KEY2 -> KEY3'. Once the FLOCK bit is in the unlock state, the access enable status is maintained until the FLOCK bit is locked.

6.6.3 CFMC_INFOKEY: Code Flash User Info Access Key Register

The CFMC_INFOKEY register is the access key register for the internal flash User info 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 flash User info area. INFOKEY[31:0] values must be written to this register in the following order to access the flash User info area. (KEY1 → KEY2 → KEY3)	
		KEY1	0x3456789A
		KEY2	0x456789AB
		KEY3	0x56789ABC

NOTE:

1. The INFOLOCK bit in the CFMC_CTRL register must be unlocked before controlling the user info memory. To do this, the KEY values must be written to the CFMC_ INFOKEY register in order KEY1 -> KEY2 -> KEY3'. Once the INFOLOCK bit is in the unlock state, the access enable status is maintained until the INFOLOCK bit is locked.

6.6.4 CFMC_FLSPROT: Code Flash Protection Register

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

CFMC_FLSPROT=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
FUP512B_7	FUP512B_6	FUP512B_5	FUP512B_4	FUP512B_3	FUP512B_2	FUP512B_1	FUP512B_0	FUBLP4K_7	FUBLP4K_6	FUBLP4K_5	FUBLP4K_4	FUBLP4K_3	FUBLP4K_2	FUBLP4K_1	FUBLP4K_0	FPBY16K_15	FPBY16K_14	FPBY16K_13	FPBY16K_12	FPBY16K_11	FPBY16K_10	FPBY16K_9	FPBY16K_8	FPBY16K_7	FPBY16K_6	FPBY16K_5	FPBY16K_4	FPBY16K_3	FPBY16K_2	FPBY16K_1	FPBY16K_0
0	0	0	0	0	0	0	0	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	FUP512B_x (x = 0 to 7)	Unprotection of the 4 KB of flash memory (0x0003_F000 to 0x0003_FFFF). - The settings of these bits have a higher priority level than the protection selection bits. - This bit is only meaningful if the FPBY16K_15 bit is '1'.
		Unprotection area for A34L716 512 B x = 0: 0x0003_F000 to 0x0003_F1FF x = 1: 0x0003_F200 to 0x0003_F3FF x = 2: 0x0003_F400 to 0x0003_F5FF x = 3: 0x0003_F600 to 0x0003_F7FF x = 4: 0x0003_F800 to 0x0003_F9FF x = 5: 0x0003_FA00 to 0x0003_FBFF x = 6: 0x0003_FC00 to 0x0003_FDFF x = 7: 0x0003_FE00 to 0x0003_FFFF
		0 Disable (protection).
		1 Enable (unprotection).
y	FUBLP4K_y (y = 0 to 7)	Protection for user boot loader area
		y = 0: 0x0000_0000 to 0x0000_0FFF y = 1: 0x0000_1000 to 0x0000_1FFF y = 2: 0x0000_2000 to 0x0000_2FFF y = 3: 0x0000_3000 to 0x0000_3FFF y = 4: 0x0000_4000 to 0x0000_4FFF y = 5: 0x0000_5000 to 0x0000_5FFF y = 6: 0x0000_6000 to 0x0000_6FFF y = 7: 0x0000_7000 to 0x0000_7FFF
		0 Unprotection
		1 Protection
z	FPBY16K_z (z = 0 to 15)	Flash area protection selection bit z is used for 16 KB flash memory.
		z = 0: 0x0000_0000 to 0x0000_3FFF z = 1: 0x0000_4000 to 0x0000_7FFF z = 2: 0x0000_8000 to 0x0000_BFFF z = 3: 0x0000_C000 to 0x0000_FFFF z = 4: 0x0001_0000 to 0x0001_3FFF z = 5: 0x0001_4000 to 0x0001_7FFF z = 6: 0x0001_8000 to 0x0001_BFFF z = 7: 0x0001_C000 to 0x0001_FFFF z = 8: 0x0002_0000 to 0x0002_3FFF z = 9: 0x0002_4000 to 0x0002_7FFF z = 10: 0x0002_8000 to 0x0002_BFFF z = 11: 0x0002_C000 to 0x0002_FFFF z = 12: 0x0003_0000 to 0x0003_3FFF z = 13: 0x0003_4000 to 0x0003_7FFF z = 14: 0x0003_8000 to 0x0003_BFFF z = 15: 0x0003_C000 to 0x0003_FFFF
		0 Unprotection
		1 Protection

NOTES:

1. When any of the FPBY16K_z bit or FUBLP4K_y 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 chip erases) do not work if any of the FUP512B_x bit is '0' when the FPBY16K_15 bit is '1'.
3. For detailed address and description of each area, refer to section 6.4.2.

6.6.5 CFMC_INFOPROT: Code Flash Info Protection Register

The CFMC_INFOPROT register is an internal flash User info area protection register. This register is 32 bits wide.

CFMC_INFOPROT=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 info 3 area protection Range: 0x0F00_0600 to 0x0F00_07FF
	0	Unprotection
	1	Protection
2	INFO2	User info 2 area protection Range: 0x0F00_0400 to 0x0F00_05FF
	0	Unprotection
	1	Protection
1	INFO1	User info 1 area protection Range: 0x0F00_0200 to 0x0F00_03FF
	0	Unprotection
	1	Protection
0	INFO0	User info 0 area protection Range: 0x0F00_0000 to 0x0F00_01FF
	0	Unprotection
	1	Protection

6.6.6 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	INFOLOCK	Reserved			ECCSI	ECCDN	Reserved	WDIEN	WABORT	Reserved	CERS	SERS	PERS	PGM
1	1	-			0	0	-	0	0	-	0	0	0	0
RW	RW	-			RW	RW	-	RW	RW	-	RW	RW	RW	RW

31	FLOCK[1]	Code flash access control bit The FLOCK bit in the CFMC_CTRL register must be unlocked before controlling the flash memory. To do this, the KEY values must be written to the CFMC_FLSKEY register in the order KEY1 -> KEY2 -> KEY3. Once the FLOCK bit is in unlock state, the access enable status is maintained until the FLOCK bit is locked.
		0 Unlock status
		1 Lock status
30	INFOLOCK	User info access control bit INFOLOCK bit of the CFMC_CTRL register must be unlocked 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 INFOLOCK bit is unlock, the access enable status is maintained until the INFOLOCK bit is lock.
		0 Unlock status
		1 Lock status
11	ECCSI	Whether to enable or disable ECC single error interrupt
		0 Disables.
		1 Enables.
10	ECCDN	Whether to enable or disable ECC double error non-Maskable interrupt
		0 Disables.
		1 Enables.
8	WDIEN[2]	Write done interrupt enable
		0 Disables.
		1 Enables.
7	WABORT	Write abort settings
		0 Disables.
		1 Enables (possible to stop during PGM).
3	CERS	Whole flash memory erase
		0 Disables.
		1 Enables.
2	SERS	Flash memory erase in sector units
		0 Disables.
		1 Enables.
1	PERS	Flash memory erase in page units
		0 Disables.
		1 Enables.
0	PGM	Flash memory write in word (4 bytes) units

	0	Disables.
	1	Enables.

NOTES:

1. To disable the code flash or the User Info access, 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 6.5.

6.6.7 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		WTERR	RPERR	WSERR	INFOPERR	FPERR	INFOLERR	FLERR	Reserved	ECCSE	ECCDE	Reserved	WDONE	Reserved	WBUSY
-		0	0	0	0	0	0	0	-	0	0	-	0	-	0
-		RWC1	RWC1	RWC1	RWC1	RWC1	RWC1	RWC1	-	RWC1	RWC1	-	RWC1	-	RO

22	WTERR	Write timeout 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	INFOPERR	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	INFOLERR	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	ECC single error	
		0	Not error
		1	ECC single error detection (cleared by writing a ‘1’)
10	ECCDE	ECC double error	
		0	Not error
		1	ECC double error detection (cleared by writing a ‘1’)
8	WDONE	Write done interrupt status	
		0	Busy
		1	Write done (cleared by writing a ‘1’)
0	WBUSY	Write busy check	
		0	Not busy
		1	Busy

6.6.8 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 User info 0 erase done flag bit
	0	Not occurred
	1	User info 0 area erase done (RPROT erase / Write permit)
24	CERSD	Chip erase done flag bit
	0	Not occurred
	1	Chip erase done (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)

6.6.9 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 = 0x1234_5678;
   CFMC_PWIN = 0x8765_4321;
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 = 0x1234_5678;
   CFMC_PWIN = 0x8765_4321; // Protection mode is released due to password match.
   CFMC_PWIN = 0x1234_5678;
   CFMC_PWIN = 0x8765_4321; // 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 = 0x1111_1111; // Wrong password value is input.
   CFMC_PWIN = 0x1234_5678;
   CFMC_PWIN = 0x8765_4321; // Password cannot disable the protection.

6.6.10 CFMC_WTIMEOUT: Code Flash Write Time-Out Register

The CFMC_WTIMEOUT register is used to set the flash write time. After the write operation (program, erase), if the write operation does not end during 'HCLK x WTIMEOUT', the write operation is terminated. This register is 32 bits wide.

CFMC_WTIMEOUT=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
WTIMEOUT[31:0]																															
0x0000_0000																															
RW																															

31 0	WTIMEOUT[31:0]	Flash write time-out setting bit	
		0	Disable
		Others	Write timeout time setting (HCLK x WTIMEOUT)

6.6.11 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	

6.6.12 CFMC_ECCEDATA0: Code Flash ECC Error Data0 Address Register

The CFMC_ECCEDATA0 register is an internal flash memory ECC data0 address 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

6.6.13 CFMC_ECCEDATA1: Code Flash ECC Error Data1 Address Register

The CFMC_ECCEDATA1 register is an internal flash memory ECC data1 address 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

6.6.14 CFMC_ECCEDATA2: Code Flash ECC Error Data2 Address Register

The CFMC_ECCEDATA2 register is an internal flash memory ECC data2 address 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

6.6.15 CFMC_ECCEDATA3: Code Flash ECC Error Data3 Address Register

The CFMC_ECCEDATA3 register is an internal flash memory ECC data3 address 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

6.6.16 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	ECCEP3[6:0]	Reserved	ECCEP2[6:0]	Reserved	ECCEP1[6:0]	Reserved	ECCEP0[6:0]
-	0x00	-	0x00	-	0x00	-	0x00
-	RO	-	RO	-	RO	-	RO

31 24	ECCEP3[6:0]	ECC error parity 3 of 128 bits
22 16	ECCEP2[6:0]	ECC error parity 2 of 128 bits
14 8	ECCEP1[6:0]	ECC error parity 1 of 128 bits
6 0	ECCEP0[6:0]	ECC error parity 0 of 128 bits

6.6.17 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

6.6.18 CFMC Register Map Summary

Table 53. 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.	WRITE	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	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_FLSPROT	FUP512B_7	FUP512B_6	FUP512B_5	FUP512B_4	FUP512B_3	FUP512B_2	FUP512B_1	FUP512B_0	FUBLP4K_7	FUBLP4K_6	FUBLP4K_5	FUBLP4K_4	FUBLP4K_3	FUBLP4K_2	FUBLP4K_1	FUBLP4K_0	FPBY16K_15	FPBY16K_14	FPBY16K_13	FPBY16K_12	FPBY16K_11	FPBY16K_10	FPBY16K_9	FPBY16K_8	FPBY16K_7	FPBY16K_6	FPBY16K_5	FPBY16K_4	FPBY16K_3	FPBY16K_2	FPBY16K_1	FPBY16K_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_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
0x14	CFMC_CTRL	FLOCK	INFOLOCK	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	ECCSI	ECCDN	Res.	WDIEN	WABOART	Res.	Res.	Res.	CERS	SERS	PERS	PGM
	Reset value	1	1																			0	0		0	0				0	0	0	0
0x18	CFMC_STAT	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	WTERR	RPERR	WSERR	INFOPERR	FPERR	INFOLERR	FLERR	Res.	Res.	Res.	Res.	ECCSE	ECCDE	Res.	WDONE	Res.	Res.	Res.	Res.	Res.	Res.	Res.	WBUSY
	Reset value										0	0	0	0	0	0	0					0	0		0								0
0x1C	CFMC_READPROT	DBGMOD	SRBOOT	Res.	Res.	Res.	PWMATCH	INFOOERSD	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
0x20	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

Table 53. 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
0x40	CFMC_WTIMEOUT	WTIMEOUT[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
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	ECCEP3[6:0]							Res	ECCEP2[6:0]							Res	ECCEP1[6:0]							Res	ECCEP0[6:0]						
	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
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

6.7 Data Flash Register

The base addresses and register map for the Data Flash Memory Controller (DFMC) are described in the tables below:

Table 54. Base Address of Data Flash Memory Controller

Name	Base Address
DFMC	0x4100_1000

Table 55. DFMC Register Map

Name	Offset	Type	Description	Reset Value	Reference
DFMC_CONF	0x0000	RW	Data flash control register	0x0000_0000	6.7.1
DFMC_FLSKEY	0x0004	WO	Data flash access key register	0x0000_0000	6.7.2
DFMC_FLSPROT	0x000C	RW	Data flash protection register	0x0000_0000	6.7.3
DFMC_CTRL	0x0014	RW	Data flash access control register	0xE000_0000	6.7.4
DFMC_STAT	0x0018	RW	Data flash access status register	0x0000_0000	6.7.5
DFMC_WTIMEOUT	0x0040	RW	Code flash write timeout register	0x0000_0000	6.7.6
DFMC_ECCEADDR	0x0050	RO	Data flash ECC error address register	0x0000_0000	6.7.7
DFMC_ECCEDATA	0x0054	RO	Data flash ECC error data register	0x0000_0000	6.7.8
DFMC_ECCEPARITY	0x0058	RO	Data flash ECC error parity register	0x0000_0000	6.7.9

6.7.1 DFMC_CONF: Data Flash Control Register

The DFMC_CONF register is the internal data flash memory's config register.

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
ECCDIS	Reserved							WRITE	Reserved																			LATENCY[3:0]			
0	-							0	-																			0000			
RW	-							RW	-																			RW			

31	ECCDIS	Disable data flash ECC
		0 ECC enable.
		1 ECC disable.
25	WRITE	Whether to enable or disable the data flash write
		0 Disables.
		1 Enables.
3 0	LATENCY[3:0](1)	Flash wait value bits Flash wait values between 0 and 15.

NOTE:

1. For detailed information of the LATENCY[3:0], Refer to section 0

6.7.2 DFMC_FLSKEY: Data Flash Access Key Register

The DFMC_FLSKEY register is the internal flash memory's access key register.

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]																															
0x0000_0000																															
WO																															

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

6.7.3 DFMC_FLSPROT: Data Flash Protection Register

The DFMC_FLSPROT register is an internal memory protection register.

CFMC_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 2 KB of flash (0x0E00_7000 ~ 0x0E00_7FFF) - This bit is only meaningful if the FPBY2K_15 bit and the FPBY2K_14 bit is '1'.	
		0	Disable (protection).
		1	Enable (unprotection).
n	FPBY2K_y (y = 0 to 15)	Flash area protection selection bit.	
		0	Unprotection
		1	Protection

NOTES:

1. When any of the FPBY2K_y bit is enabled, bulk 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 chip erases) do not work if any of the FUP512B_x bit is ‘0’ when the FPBY2K_15 bit and the FPBY2K_14 bit is ‘1’.
3. For detailed address and description of each area, refer to 6.4.2.2

6.7.4 DFMC_CTRL: Data Flash Access Control Register

The DFMC_CTRL register is the internal flash memory's access control register.

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]	Flash memory lock
		0 Unlock status
		1 Lock status and flash access Lock
11	ECCSI	Whether to enable or disable the ECC single error interrupt.
		0 Disables.
		1 Enables.
10	ECCDN	Whether to enable or disable the ECC double error non-maskable interrupt
		0 Disables.
		1 Enables.
8	WDIEN[2]	Write done interrupt enable
		0 Disables.
		1 Enables.
7	WABORT	Write abort settings
		0 Disables.
		1 Enables (possible to stop during PGM).
3	CERS	Whole flash memory erase
		0 Disables.
		1 Enables.
2	SERS	Flash memory erase in sector units
		0 Disables.
		1 Enables.
1	PERS	Flash memory erase in page units
		0 Disables.
		1 Enables.
0	PGM	Flash memory write in word (4 bytes) units
		0 Disables.
		1 Enables.

NOTES:

1. To disable the data flash 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 6.5.

6.7.5 DFMC_STAT: Data Flash Access Status Register

The DFMC_STAT register is the internal flash memory's access status register.

DFMC_STAT=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
Reserved		WTERR	RPERR	WSERR	Reserved	FPERR	Reserved	FLERR	Reserved	ECCSE	ECCDE	Reserved	WDONE	Reserved	WBUSY
-		0	0	0	-	0	-	0	-	0	0	-	0	-	0
-		RWC1	RWC1	RWC1	-	RWC1	-	RWC1	-	RWC1	RWC1	-	RWC1	-	RO

22	WTERR	Write timeout 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 (auto cleared)	
		0	Not error
		1	Error detection (cleared by writing a '1')
16	FLERR	Flash lock error (auto cleared)	
		0	Not error
		1	Error detection (cleared by writing a '1')
11	ECCSE	Data flash ECC single error	
		0	Not error
		1	ECC single error detection (cleared by writing a '1')
10	ECCDE	Data flash ECC double error	
		0	Not error
		1	ECC double error detection (cleared by writing a '1')
8	WDONE	Write done interrupt status	
		0	Busy
		1	Write done (cleared by writing a '1')
0	WBUSY	Write busy check	
		0	Not busy
		1	Busy

6.7.6 DFMC_WTIMEOUT: Data Flash Write Time-Out Register

The DFMC_WTIMEOUT register is used to set the Data flash write time. After the write operation (program, erase), if the write operation does not end during 'HCLK x WTIMEOUT', the write operation is terminated. This register is 32 bits wide.

DFMC_WTIMEOUT=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
WTIMEOUT[31:0]			
0x0000_0000			
RW			

31 0	WTIMEOUT[31:0]	Write timeout count
		0 Disable write timeout
		Others Enable write timeout

$$t_{Timeout} = \frac{WTOUT}{HCLK}$$

6.7.7 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
Reserved		ECCEA[15:0]	
-		0x0000	
-		RO	

15 0	ECCEA[15:0]	ECC error address (Offset: 0x0E00_0000)

6.7.8 DFMC_ECCEDATA: Data Flash ECC Error Data Register

The DFMC_ECCEDATA register is an internal flash memory ECC Error 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
Reserved		ECCED[15:0]	
-		0x0000	
-		RO	

15 0	ECCED[15:0]	ECC error data (Offset: 0x0E00_0000)

6.7.9 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
-			RO

6 0	ECCEP[6:0]	ECC error parity

6.7.10 DFMC Register Map Summary

Table 56. 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	ECCDIS						WRITE																						LATENCY[3:0]			
	Reset value	0						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									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																				ECCSI	ECCDN		WDIEN	WABOART				CERS	SERS	PERS	PGM
	Reset value	1																				0	0		0	0				0	0	0	0
0x18	DFMC_STAT										WTERR	RPERR	WSERR		FPERR		FLERR					ECCSE	ECCDE		WDONE								WBUSY
	Reset value										0	0	0		0		0					0	0		0								0
0x40	DFMC_WTIMEOUT	WTIMEOUT[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
0x54	DFMC_ECCEDATA																	ECCED[15:0]															
	Reset value																	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
0x58	DFMC_ECCEPARITY																										ECCEP[6:0]						
	Reset value																										0	0	0	0	0	0	0

7. HSICMU (High-Speed Internal Oscillator Clock Management Unit)

7.1 Introduction

The High-Speed Internal OSC Clock Management Unit (HSICMU) integrated into the A34L716 is designed for the auto-trimming of the Internal OSC clock used for X'tal-less USB.

The following are characteristics of the High Speed Internal OSC Clock Management Unit (HSICMU) built in the A34L716. It is used to auto-trimming of Internal OSC clock for X'tal-less USB.

The HSICMU is ideally suited to provide a precise clock for the USB peripheral. In such a case, the synchronization signal can be derived from the start-of-frame (SOF) packet signalization on the USB bus, which is sent by a USB host at 1 ms intervals.

The synchronization signal can also be derived from the LSE oscillator output, or it can be generated by user software.

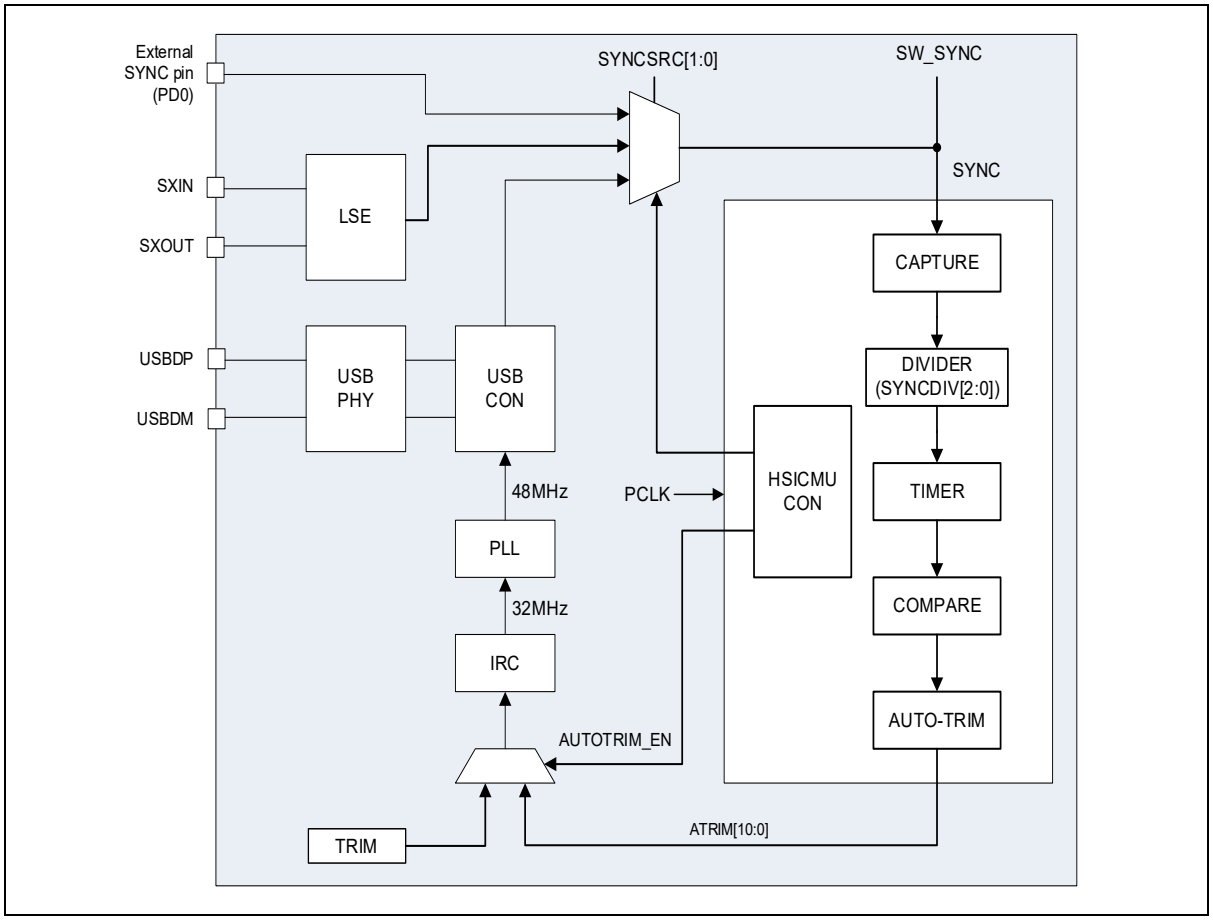
7.2 HSICMU Main Features

- Selectable synchronization source with programmable prescaler and polarity:
  - USB SOF (Start of Frame) sent received toggle
  - LSE
  - External SYNC pin (PD0)
- Possibility to generate synchronization pulses by software
- Automatic oscillator trimming capability with no need for CPU action
- 16-bit frequency error counter with automatic error value capture and reload
- Programmable limit for automatic frequency error value evaluation and status reporting

7.3 HSICMU Functional Description

7.3.1 HSICMU Block Diagram

Figure 46. HSICMU Block Diagram



7.3.2 Synchronization Input

The HSICMU synchronization (SYNC) source, selectable through the HSICMU_CFG register, can be the signal from the LSE clock or the USB SOF signal.

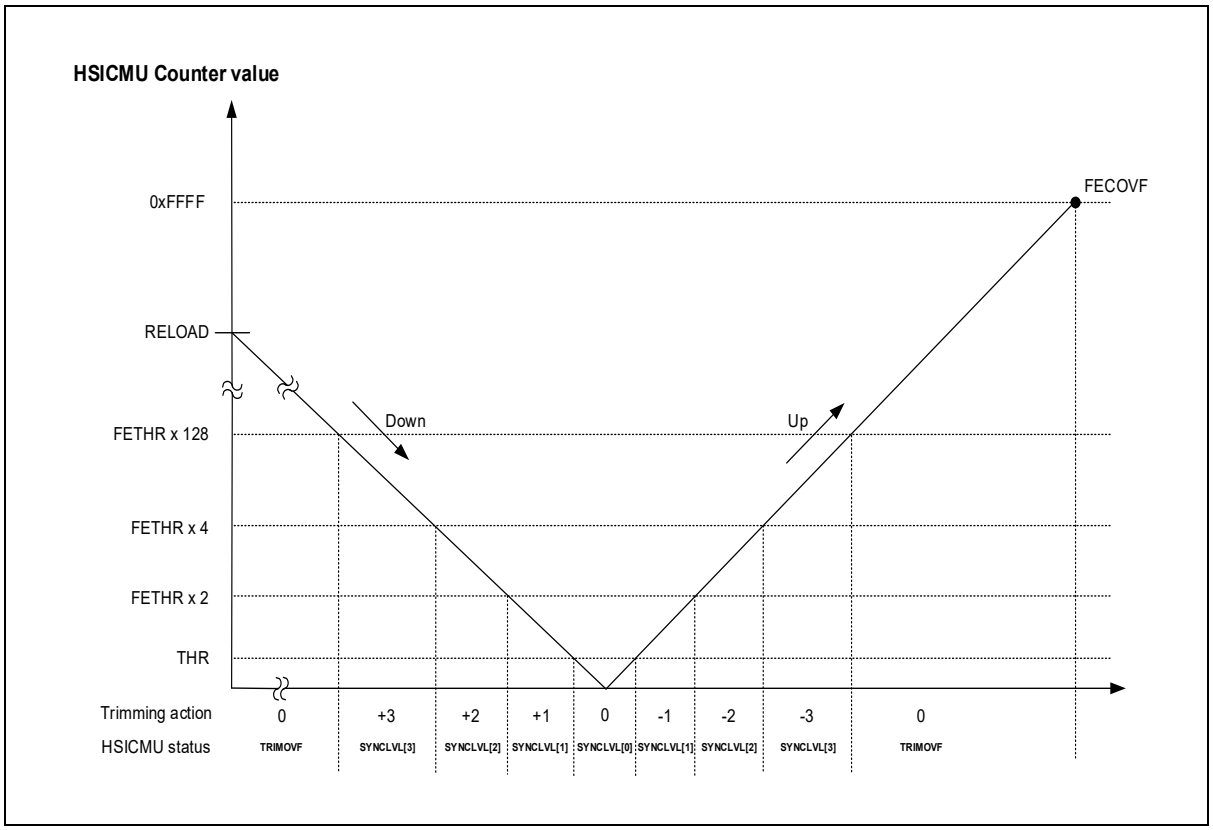
For more information on the HSICMU synchronization source configuration, refer to 7.4.2

7.3.3 Frequency Error Measurement

The frequency error counter is a 16-bit down/up counter which is reloaded with the RELOAD value on each SYNC event. It starts counting down till it reaches zero value. Then it starts counting to the 0xFFFF where it eventually stops (if no SYNC event is received) and generates a FECOVF event.

When the SYNC event is detected, the actual value of the frequency error counter and its counting direction are stored in the CAPTURE (frequency error capture) field and in the FECDIR (frequency error direction) bit of the HSICMU_ISR register. When the SYNC event is detected during the down counting phase (before reaching the zero value), it means that the actual frequency is lower than the target (and so, that the TRIM value must be incremented), while when it is detected during the up counting phase it means that the actual frequency is higher (and that the TRIM value must be decremented).

Figure 47. HSICMU Counter Behavior



7.3.4 Frequency Error Evaluation and Automatic Trimming

The measured frequency error is evaluated by comparing its value with a set of limits:

- SYNCLVL[0] limit, given directly in the FETHR field of the HSICMU_THR register.
- SYNCLVL[1] limit, defined as 2 × FETHR value.
- SYNCLVL[2] limit, defined as 4 × FETHR value.
- SYNCLVL[3] limit, defined as 128 × FETHR value.

The result of this comparison is used to generate the status indication and also to control the automatic trimming which is enabled by setting the AUTOTRIM_EN bit in the HSICMU_CR register:

- When the frequency error is below the SYNCLVL[0] limit , it means that the actual trimming value in the TRIM field is the optimal one, hence no trimming action is needed.
  - SYNCLVL[0] status indicated (TRIM value not changed in AUTOTRIM mode)
- When the frequency error is above or equal to the SYNCLVL[0] limit but below the SYNCLVL[1] limit, it means that some trimming action is necessary but that adjustment by one trimming step is enough to reach the optimal TRIM value.
  - SYNCLVL[1] status indicated (TRIM value adjusted by one trimming step in AUTOTRIM mode)
- When the frequency error is above or equal to the SYNCLVL[1] limit but below the SYNCLVL[2] limit, it means that a stronger trimming action is necessary, and there is a risk that the optimal TRIM value is not reached for the next period.
  - SYNCLVL[2] status indicated (TRIM value adjusted by two trimming steps in AUTOTRIM mode)
- When the frequency error is above or equal to the SYNCLVL[2] limit but below the SYNCLVL[3] limit, it means that a stronger trimming action is necessary, and there is a risk that the optimal TRIM value is not reached for the next period.
  - SYNCLVL[3] status indicated (TRIM value adjusted by three trimming steps in AUTOTRIM mode)
- When the frequency error is above or equal to the SYNCLVL[3] limit, it means that the frequency is out of the trimming range. This can also happen when the SYNC input is not clean or when some SYNC pulse is missing (for example when one USB SOF is corrupted).
  - SYNCLVL[4] status indicated (TRIM value not changed in AUTOTRIM mode)

NOTE:

1. If the actual value of the ATRIM field is so close to its limits that the automatic trimming would force it to overflow or underflow, then the TRIM value is set just to the limit and the TRIMOVF status is indicated
   In AUTOTRIM mode (AUTOTRIM_EN bit set in the HSICMU_CR register), the ATRIM field of HSICMU_ATRIM is adjusted by hardware and is read-only.

7.3.5 HSICMU Initialization and Configuration

The RELOAD value must be selected according to the ratio between the target frequency and the frequency of the synchronization source after prescaling. It is then decreased by one to reach the expected synchronization on the zero value. The formula is the following:

$$\mathrm{RELOAD} = \left(\frac{\mathrm{f_{TARGET}}}{\mathrm{f_{SYNC}}}\right) - 1$$

The reset value of the RELOAD field corresponds to a target frequency of 32 MHz and a synchronization signal frequency of 1 kHz (SOF signal from USB).

The selection of the FETHR value is closely coupled with the HSI oscillator characteristics and its typical trimming step size. The optimal value corresponds to half of the trimming step size, expressed as several HSI oscillator clock ticks. The following formula can be used:

$$\mathrm{FETHR} = \left(\frac{\mathrm{f_{TARGET}}}{\mathrm{f_{SYNC}}}\right) \times \mathrm{HSI\ TRIM\ STEP[\%]}/100\%$$

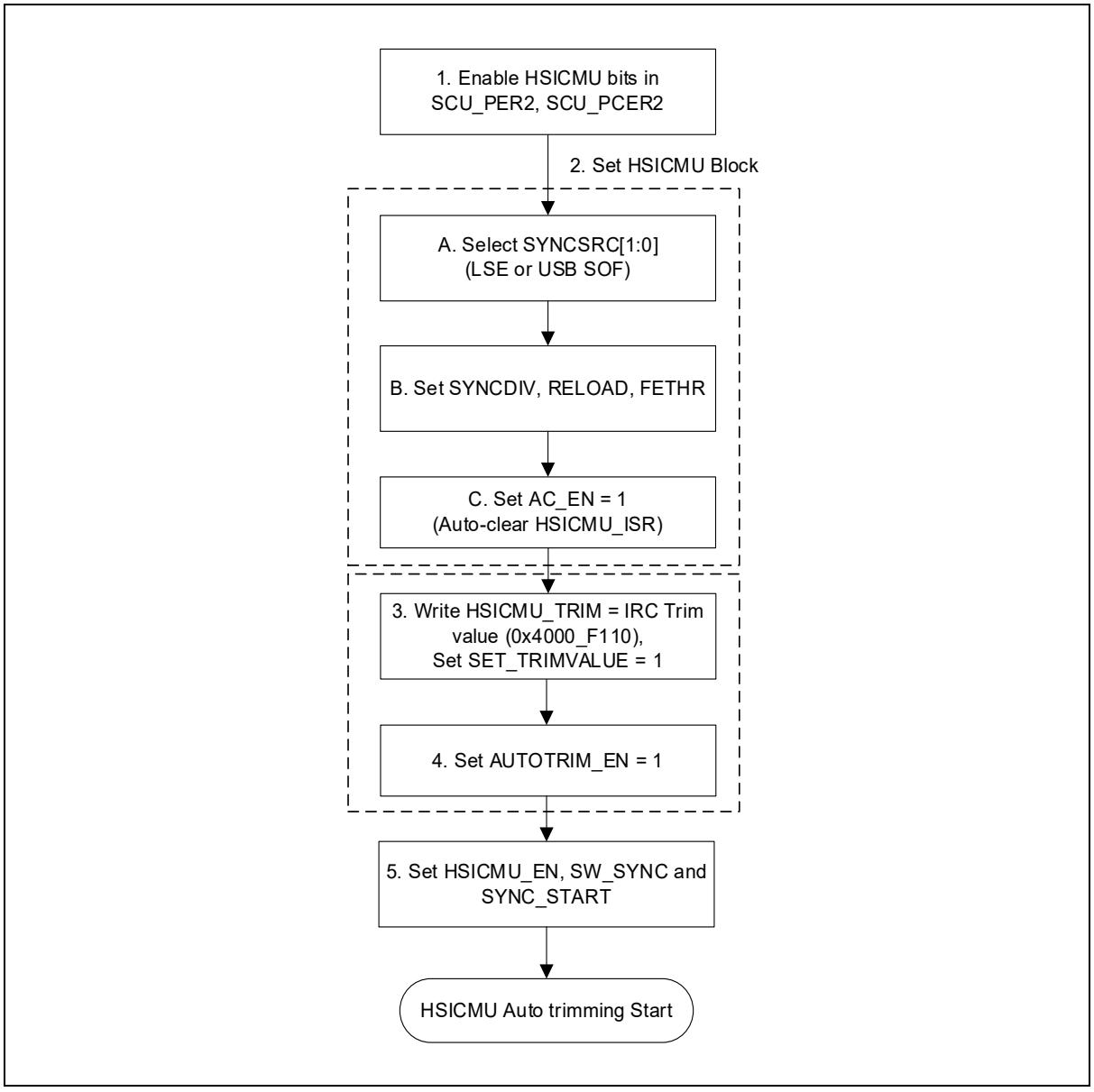
The result must always be rounded up to the nearest integer value to obtain the best trimming response. If frequent trimming actions are not needed in the application, the hysteresis can be increased by slightly increasing the FETHR value.

The reset value of the FELIM field corresponds to (f_{TARGET} / f_{SYNC}) = 32000 and to a typical trimming step size of 0.2%.

There is no hardware protection from a wrong configuration of the RELOAD and FETHR fields which can lead to an erratic trimming response. The expected operational mode requires proper setup of the RELOAD value (according to the synchronization source frequency), which is also greater than 128 × FETHR value (SYNCLVL[3] limit).

In general, users can configure the HSICMU based on the information below. Note that the HSICMU is only valid when PCLK is sourced from HSI, since it calibrates the HSI based on PCLK.

Figure 48. HSICMU Initialization Flowchart



1. Enable the HSICMU-related bits in the SCU_PER2 and SCU_PCER2 registers.
2. The following A, B, and C describe how to configure the HSICMU_CFG register.
   A. Select SYNCSRC[1:0] as a reference source for comparison with PCLK. The SYNCSRC[1:0] can be set to either LSE (32.768 kHz) or USB SOF (1 kHz).

B. Calculate and set SYNCDIV[2:0] so that RELOAD[15:0] is not set too small. Examples of appropriate SYNCDIV[2:0], RELOAD[15:0], and FETHR[15:0] settings for each SYNCSRC[1:0] can be found in Table 57 and Table 58. These examples assume that SESYNCPOL[1:0] is set to either Rising edge or Falling edge. If Both edge is selected, the RELOAD value should be halved from the previously calculated value.

C. The AC_EN bit determines whether the HSICMU_ISR register is automatically cleared. It is recommended to set this bit to '1'.

3. Before starting the HSICMU, the value stored in the IRC Trim register must be written to the HSICMU_TRIM register. The address of the IRC Trim register is 0x4000_F110. After writing the value to the HSICMU_TRIM register, set the SET_TRIMVALUE bit in the HSICMU_CR register to '1'.

4. To configure AUTO TRIM mode, set the AUTOTRIM_EN bit to '1'. The AUTOTRIM_EN bit must be enabled only after the SET_TRIMVALUE bit has been enabled. Do not enable the SET_TRIMVALUE bit and the AUTOTRIM_EN bit simultaneously, as the trim value may not be loaded correctly.

   A. Example code for setting the HSICMU SET_TRIMVALUE bit and AUTOTRIM_EN bit

   ```
   HSICMU_CR = 0x0000_2000; // Enable SET_TRIMVALUE bit

   HSICMU_CR = 0x0000_0200; // Enable AUTOTRIM_EN bit
   ```

5. To start the HSICMU, enable the HSICMU_EN, SW_SYNC, and SYNC_START bits in the HSICMU_CR register.

6. Then, HSI calibration starts automatically, and the following operations A, B, and C are repeated:

   A. A down-count is performed based on the value written to the RELOAD register. When the count reaches zero, it continues counting up to 0xFFFF. However, reaching 0xFFFF typically indicates a malfunction, in which case the FECOVF flag is triggered. For more details on this operation, refer to Figure 47.

   B. After counting ends, the HSICMU_CAP register value is compared with the configured FETHR value to determine whether to adjust the IRC Trim. For details on trimming based on FETHR, refer to section 7.3.4.

   C. The calibrated IRC Trim value can be checked in the HSICMU_ATRIM register.

Table 57 shows examples of RELOAD[15:0] and FETHR[15:0] values according to PCLK for each SYNCSRC[1:0]. When f_{TARGET} was SOF(1 kHz), the calculation was made assuming the threshold was 0.15%.

Table 57. Examples HSICMU (SYNCSRC[1:0] = 00, SOF)

SYNCSRC[1:0]	SYNCDIV[2:0]	PCLK	RELOAD[15:0]	FETHR[15:0]
00 (SOF, 1kHz)	000	32 MHz	32000	48
		16 MHz	16000	24
		8 MHz	8000	12
		4 MHz	4000	6
		2 MHz	2000	3
	001	16 MHz	32000	48
	010	8 MHz	32000	48
	011	4 MHz	32000	48
	100	2 MHz	32000	48
	101	1 MHz	32000	48

Table 58 provides an example when SYNCSRC[1:0] is set to LSE (32.7689 kHz).

When f_{TARGET} is 32.768 kHz, it is recommended to divide it using SYNCDIV[2:0] to increase the RELOAD[15:0] value, since a small RELOAD value may result in lower accuracy.

For this reason, a threshold of 0.15% is assumed for the calculation.

Table 58. Examples HSICMU (SYNCSRC[1:0] = 10, LSE)

SYNCSRC[1:0]	SYNCDIV[2:0]	PCLK	RELOAD[15:0]	FETHR[15:0]
10 (LSE, 32.768kHz)	000	32 MHz	976	3
		16 MHz	488	3
		8 MHz	244	3
		4 MHz	122	2
		2 MHz	61	2
	101	32 MHz	31250	46
	110	16 MHz	31250	46
	111	8 MHz	31250	46
	111	4 MHz	15625	23
	111	2 MHz	7812	11
	111	1 MHz	3906	5

7.4 HSICMU Registers

The base addresses and register map for HSICMU are described in the tables below:

Table 59. Base Address of HSICMU

Name	Base Address
HSICMU	0x4000_4000

Table 60. HSICMU Register Map

Name	Offset	Type	Description	Reset Value	Reference
HSICMU_CR	0x0000	RW	HSICMU Control register	0x0000_1000	7.4.1
HSICMU_CFG	0x0004	RW	HSICMU Configure register	0x0000_FFFF	7.4.2
HSICMU_ISR	0x0008	RW	HSICMU Interrupt & status register	0x0000_0000	7.4.3
HSICMU_ICR	0x000C	RW	HSICMU Interrupt & status clear register	0x0000_0000	7.4.4
HSICMU_TRIM	0x0010	RW	HSICMU Trim register	0x0000_0000	7.4.5
HSICMU_ATRIM	0x0014	RW	HSICMU Auto trim register	0x0000_0000	7.4.6
HSICMU_CAP	0x0018	RW	HSICMU Capture register	0x0000_0000	7.4.7
HSICMU_THR	0x001C	RW	HSICMU Freq. error threshold register	0x0000_0000	7.4.8

7.4.1 HSICMU_CR: HSICMU Control Register

The HSICMU_CR register has control bits to the HSICMU operation. This register is 32bits wide.

HSICMU_CR=0x4000_4000

31 30 29 28 27 26 25 24	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			SET_TRIMVALUE	SYNC_START	SET_TIMER	SW_SYNC	AUTOTRIM_EN	HSICMU_EN	Reserved
-			0	1	0	0	0	0	-
-			WO	RW	WO	WO	RW	RW	-

13	SET_TRIMVALUE	Trim value is set to HSICMU_TRIM in HSICMU_ATRIM block.
	0	Disables
	1	Enables
12	SYNC_START	Start synchronized timer by SOF signals
	0	Disables
	1	Enables
11	SET_TIMER	The value in the reload register is written to the HSICMU_TIMER counter
	0	Disables
	1	Enables
10	SW_SYNC	Generate Software SYNC event
	0	Disables
	1	Enables
9	AUTOTRIM_EN	Auto trim enable
	0	Disables
	1	Enables
8	HSICMU_EN	HSICMU Peripheral block Enable
	0	Disables.
	1	Enables.

7.4.2 HSICMU_CFG: HSICMU Configure Register

The HSICMU_CFG register is used to set configure HSICMU. This register is 32bits wide.

HSICMU_CFG=0x4000_4004

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

31 30	SESYNCPOL[1:0]	Sync Signal Polarity Selection.	
		00	Both edge
		01	Rising edge
		10	Falling edge
		11	Reserved
29 28	SYNCSRC[1:0]	Sync Signal Source Selection	
		00	Capture Data Update when SOF is successfully received.
		01	Reserved
		10	SOSC(LSE) CLK
		11	External SYNC pin (PD0)
27	AC_EN	Auto Clear HSICMU_ISR.	
		0	Disables
		1	Enables
26 24	SYNCDIV[2:0]	Sync divider	
		000	Sync not divided.
		001	Sync divided by 2.
		010	Sync divided by 4.
		011	Sync divided by 8.
		100	Sync divided by 16.
		101	Sync divided by 32.
		110	Sync divided by 64.
		111	Sync divided by 128.
15 0	RELOAD[15:0]	HSICMU_TIMER counter is set to RELOAD Value	

7.4.3 HSICMU_ISR: HSICMU Interrupt and Status Register

The HSICMU_ISR is used to check interrupt & status of HSICMU.

HSICMU_CR=0x4000_4008

31 30 29 28 27 26 25 24	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		FECDIR	SYNC_FLAG	TRIMOVF	FECOVF	Reserved		SYNCLVL[4]	SYNCLVL[3]	SYNCLVL[2]	SYNCLVL[1]	SYNCLVL[0]
-		0	0	0	0	-		0	0	0	0	0
-		RO	RO	RO	RO	-		RO	RO	RO	RO	RO

15	FECDIR	Frequency Error Counter Direction.
		0 Counter in decreasing direction
		1 Counter in increasing direction
14	SYNC_FLAG	Sync Signal Flag
		0 None
		1 Updated HSICMU_CAP register with input sync signal
13	TRIMOVF	The value in the reload register is written to the HSICMU_TIMER counter
		0 None
		1 The trim value is out of limits when AUTOTRIMEN = '1'
12	FECOVF	Frequency Error Overflow
		0 None
		1 The Counter value changed in increasing direction and the counter value increased to the end (0xFFFF)
4 0	SYNCLVL[4:0]	Sync Error Rate level
		SYNCLVL[0]: HSICMU_CAP ≤ FETHR
		SYNCLVL[1]: FETHR < HSICMU_CAP < FETHR × 2
		SYNCLVL[2]: FETHR × 2 < HSICMU_CAP ≤ FETHR × 4
		SYNCLVL[3]: FETHR × 4 < HSICMU_CAP ≤ FETHR × 128
		SYNCLVL[4]: FETHR × 128 < HSICMU_CAP

7.4.4 HSICMU_ICR: HSICMU Control Interrupt and Clear Status Register

The HSICMU_ICR is used to control interrupt and clear status of HSICMU.

HSICMU_ICR=0x4000_400C

31 30 29 28 27 26 25 24	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			SYNC_FLAG	TRIMOVF	FECOVF	Reserved		SYNCLVL[4]	SYNCLVL[3]	SYNCLVL[2]	SYNCLVL[1]	SYNCLVL[0]
-			0	0	0	-		0	0	0	0	0
-			WO	WO	WO	-		WO	WO	WO	WO	WO

14	SYNC_FLAG	Sync Signal Flag clear	
		0	None
		1	Clear when set to ‘1’
13	TRIMOVF	Trim value over flow flag clear	
		0	None
		1	Clear when set to ‘1’
12	FECOVF	Frequency Error Over Flow flag clear after HSICMU_CR.HSICMU_EN	
		0	None
		1	Clear when set to ‘1’
4 0	SYNCLVL[4:0]	Error Rate level flag clear after HSICMU_CR.SYNC_START set to ‘0’.	
		SYNCLVL[0]: Clear when set to ‘1’.	
		SYNCLVL[1]: Clear when set to ‘1’.	
		SYNCLVL[2]: Clear when set to ‘1’.	
		SYNCLVL[3]: Clear when set to ‘1’.	
		SYNCLVL[4]: Clear when set to ‘1’.	

7.4.5 HSICMU_TRIM: HSICMU Control TRIM Value Register

The HSICMU_TRIM is used to control trim value of HSICMU. This register is 32bits wide.

HSICMU_TRIM=0x4000_4010

31 30 29 28 27 26 25 24	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			TRIM[10:0]
-			0 0 0 0 0 0 0 0 0 0 0
-			RW

10 0	TRIM[10:0]	Trim value register for trimming

7.4.6 HSICMU_ATRIM: HSICMU Control Auto-Trim Value Register

The HSICMU_ATRIM is used to check auto-trim value of HSICMU. This register is 32bits wide.

HSICMU_ATRIM=0x4000_4014

31 30 29 28 27 26 25 24	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			ATRIM[10:0]
-			0 0 0 0 0 0 0 0 0 0 0
-			RO

10 0	ATRIM[10:0]	Current trim value during auto trimming

7.4.7 HSICMU_CAP: HSICMU Capture Value Register

The HSICMU_CAP is used to capture the value of frequency error rate counter. This register is 32bits wide.

HSICMU_ATRIM=0x4000_4018

31 30 29 28 27 26 25 24	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		CAPTURE[15:0]	
-		0x0000	
-		RO	

15 0	CAPTURE[15:0]	Capture value of the frequency error rate counter

7.4.8 HSICMU_THR: HSICMU Frequency Error Rate Threshold Register

The HSICMU_THR is used to set frequency error rate threshold. This register is 32bits wide.

HSICMU_THR=0x4000_401C

31 30 29 28 27 26 25 24	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		FETHR[15:0]	
-		0x0000	
-		RO	

15 0	FETHR[15:0]	Frequency error rate threshold

7.4.9 HSICMU Register Map Summary

Table 61. HSICMU 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	HSICMU_CR	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	SET_TRIMVALUE	SYNC_START	SET_TIMER	SW_SYNC	AUTOTRIM_EN	HSICMU_EN	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.
	Reset value																			0	1	0	0	0	0								
0x04	HSICMU_CFGR	SESYNCPOL[1:0]		SYNCSRC[1:0]		AC_EN	SYNCDIV[2:0]			Res	Res	Res	Res	Res	Res	Res	Res	RELOAD[15:0]															
	Reset value	0	0	0	0	0	0	0	0									1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1
0x08	HSICMU_ISR	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	FECDIR	SYNC_FLAG	TRIMOVF	FECOVF	Res.	Res.	Res.	Res.	Res.	Res.	Res.	SYNCLVL[4]	SYNCLVL[3]	SYNCLVL[2]	SYNCLVL[1]	SYNCLVL[0]
	Reset value																	0	0	0	0								0	0	0	0	0
0x0C	HSICMU_ICR	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.		SYNC_FLAG	TRIMOVF	FECOVF	Res.	Res.	Res.	Res.	Res.	Res.	Res.	SYNCLVL[4]	SYNCLVL[3]	SYNCLVL[2]	SYNCLVL[1]	SYNCLVL[0]
	Reset value																		0	0	0								0	0	0	0	0
0x10	HSICMU_TRIM						Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.						TRIM[10:0]										
	Reset value																						0	0	0	0	0	0	0	0	0	0	0
0x14	HSICMU_ATRIM						Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.						ATRIM[10:0]										
	Reset value																						0	0	0	0	0	0	0	0	0	0	0
0x18	HSICMU_CAP	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	CAPTURE[15:0]															
	Reset value																	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
0x1C	HSICMU_THR	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	FETHER[15:0]															
	Reset value																	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

8. DMA (Direct Memory Access Controller)

8.1 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.

- Seven channels
- 8-/ 16- / 32-bit data transfers supported.
- Various buffers with the same size supported.
- DMA transfers are triggered through peripheral interrupts.

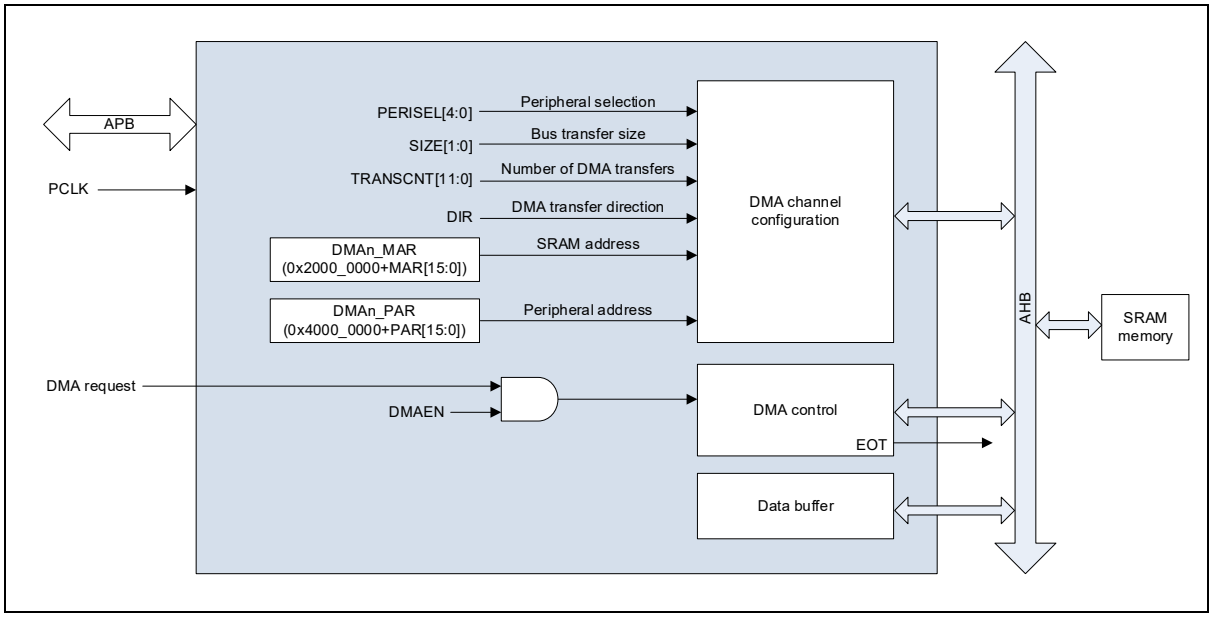
8.1.1 Modules Supported by DMA

A34L716 DMA supports the following six peripherals.

- UART
- SPI
- USART
- LPUART
- ADC
- CRC

Figure 49 shows a DMA block diagram.

Figure 49. DMA Block Diagram



8.2 DMA Functional Description

The DMA controller directly transfers data to memories by sharing the AHB with the microcontroller's 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's 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 / 32-bit).
3. In the DMAn_SR register, the programmer sets the DMARC to '1' to clear the DMA channel.
4. In the DMAn_SR register, the programmer sets the DMAEN to '1' to enable the DMA channel.
5. A DMA request is made by the peripheral set in PERISEL[4:0] of DMAn_CR and this triggers the requested peripheral channel to become active.
6. 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.
7. Each time data is written to the target address, the DMA transfer count decreases by '1'. When the DMA transfer count reaches 0, EOT bit of DMAn_SR is set to '1'. This signals an interrupt to the connected peripheral(1).

NOTE:

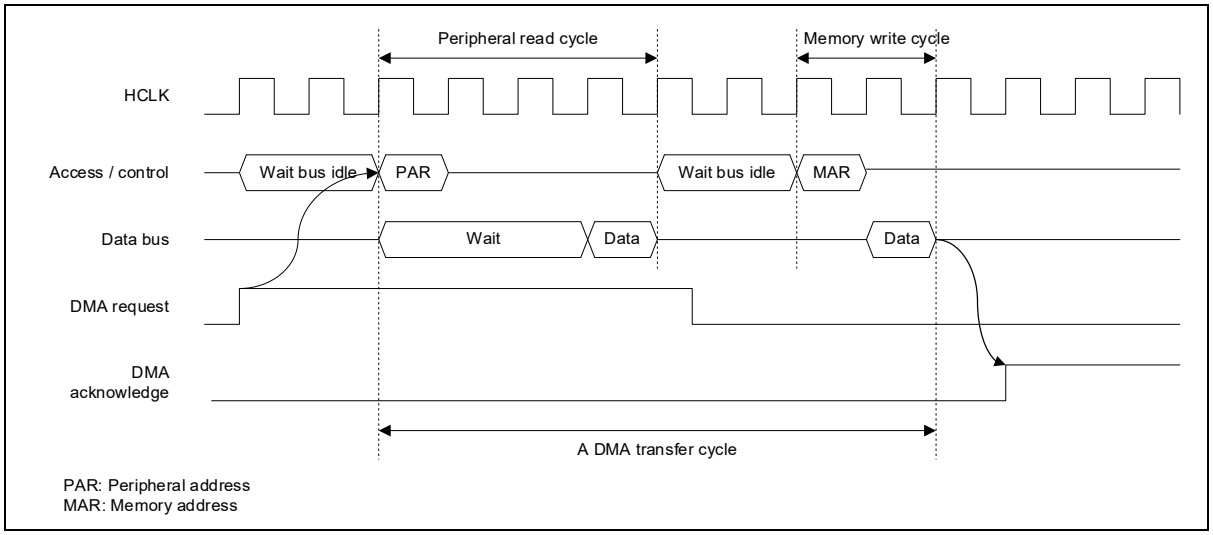
1. DMA itself has no interrupt sources; instead, each peripheral has flag bits that display the status of the connected DMA channel's transfer interrupts.

8.2.1 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.

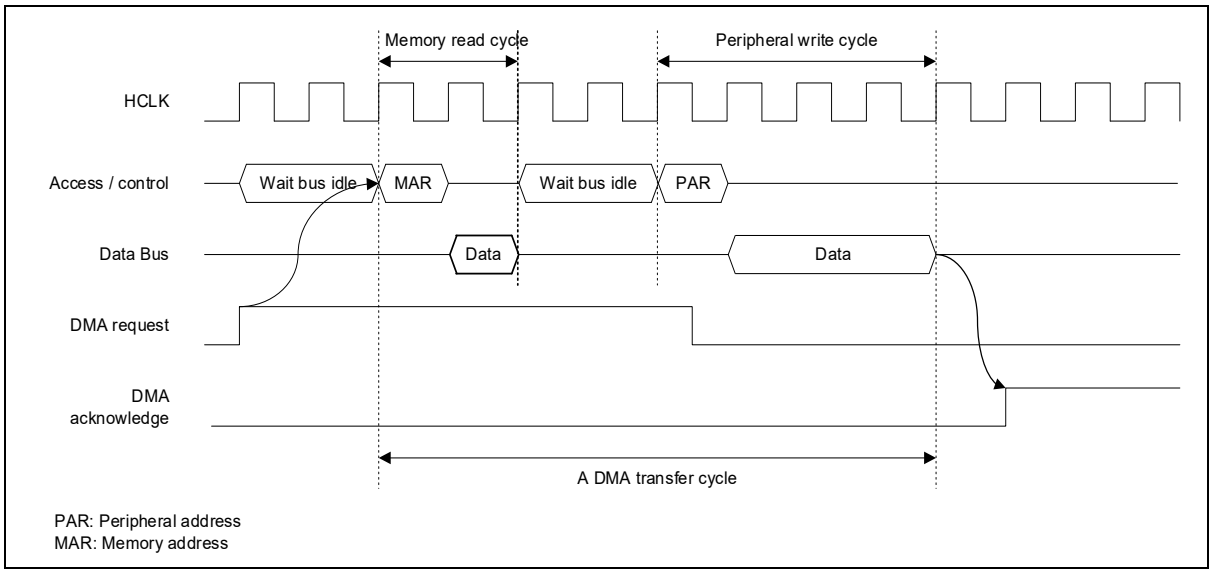
Figure 50. Diagram of DMA Transfer Timing from Peripheral to Memory



8.2.2 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.

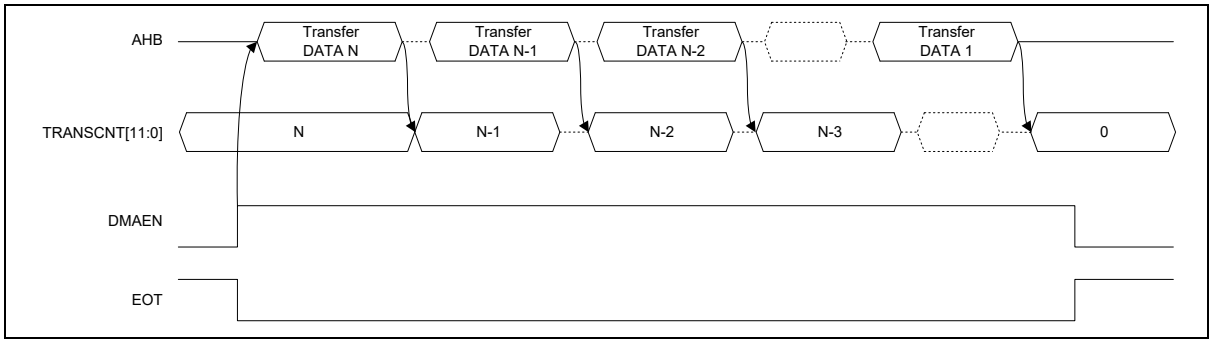
Figure 51. Diagram of DMA Transfer Timing from Memory to Peripheral



8.2.3 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 52. N Number of DMA Transfers



8.3 DMA Registers

The base addresses and register map for the DMA are described in the tables below:

Table 62. Base Addresses 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

Table 63. DMA Register Map

Name	Offset	Type	Description	Reset value	Reference
DMAn_CR	0x0000	RW	DMA channel n control register	0x0000_0000	8.3.1
DMAn_SR	0x0004	RW	DMA channel n status register	0x0000_0080	8.3.2
DMAn_PAR	0x0008	RW	DMA channel n peripheral address register	0x4000_0000	8.3.3
DMAn_MAR	0x000C	RW	DMA channel n memory address register	0x2000_0000	8.3.4

NOTE:

1. n = 0, 1, 2, 3, 4, 5, 6.

8.3.1 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

31 30 29 28 27 26 25 24	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[4:0]	Reserved	SIZE[1:0]	DIR	Reserved
-	0x000	-	0 0 0 0 0	-	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.
12 8	PERISEL[4:0](1)	Peripheral selection The selected peripheral is connected to the DMA channel. PERISEL[4:0] must be written with the number representing the peripheral to be connected to the DMA interface.
	N	Select the peripheral to use. Refer to Table 64.
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 -> peripheral (TX)
	1	Peripheral -> memory (RX)

NOTE:

1. A DMA channel will be connected with selected peripheral. Below table shows peripheral selection numbers. This PERISEL[4:0] field should be set with proper number of peripherals which will be connected with DMA interface.

Table 64. DMA PERISEL[4:0] Selection

PERISEL[4:0]	Peripheral	PERISEL[4:0]	Peripheral
0	CHANNEL IDLE	9	LPUART1 RX
1	UART RX	10	LPUART1 TX
2	UART TX	11	SPI0 RX
3	USART10 RX	12	SPI0 TX
4	USART10 TX	13	SPI1 RX
5	USART11 RX	14	SPI1 TX
6	USART11 TX	15	ADC0 RX
7	LPUART0 RX	16	ADC1 TX
8	LPUART0 TX	17	CRC TX

NOTE:

1. PERISEL[4:0] cannot have the same value in different channels. If the same PERISEL[4:0] value is written in more than one channel, proper operation cannot be guaranteed. Unused channels should have CHANNEL IDLE value in PERISEL[4:0] bit positions.

8.3.2 DMAn_SR: DMA Channel n Status Register

DMAn_SR is a 32-bit register. It shows the current 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

31 30 29 28 27 26 25 24	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		DMARC	Reserved			DMAEN
-			1	-		0	-			0
-			RO	-		RWC1	-			RW

7	EOT	Transfer ends.	
		0	There are remaining transfers. TRANSCNT[11:0] has a value other than zero.
		1	All data has been transferred. TRANSCNT[11:0] value is zero.
4	DMARC(1)	DMA request clear (Auto clear)	
		0	Does not use.
		1	Initializes DMA.
0	DMAEN(1)	Enables DMA use.	
		0	Disables DMA or fix the status.
		1	Enables DMA or available to use.

NOTE:

1. When enabling the DMA, The DMA request clear bit must be set to '1' to initialize the DMA before using.
   DMARC bit in DMAn_SR register = '1' // Initializes DMA.
   DMAEN bit in DMAn_SR register = '1' // Enables DMA.

8.3.3 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

31 30 29 28 27 26 25 24	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

NOTE:

1. Peripheral base address in DMAn_PAR register must be set to '0x4000'.

8.3.4 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

31 30 29 28 27 26 25 24	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[15:0]	
0x2000		0x0000	
RW		RW	

15 0	MAR[15:0]	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.

NOTE:

1. Memory Base Address in DMAn_MAR register must be set to '0x2000'.

8.3.5 DMA Register Map Summary

Table 65. 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.	Res.	PERISEL[4:0]					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	
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.	DMARC	Res.	Res.	Res.	DMAEN
	Reset value																									1			0				0
0x08	DMAn_PAR	Peripheral Base Address offset																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 offset																MAR[15:0]															
	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

9. WDT (Watchdog Timer)

9.1 WDT Introduction

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 it to normal.

The WDT of the A34L716 is a 32-bit down counter. If the WDT is set as a reset source, the microcontroller restarts when the down counter reaches 0.

When it is not used to monitor the microcontroller, the WDT can be used as a cycle timer along with an interrupt.

9.2 WDT Main Features

The WDT module has main 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] in 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.

9.3 WDT Functional Description

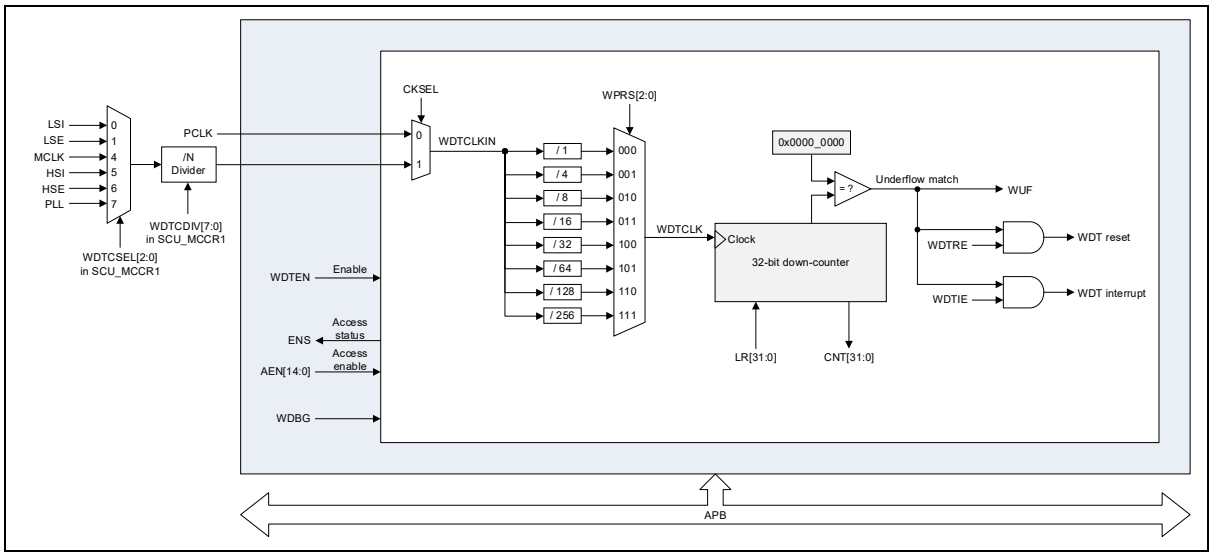
Once the WDT module is enabled and the WDTEN in WDT_CON register is set to ‘1’, the 32-bit down counter is decremented. The application program writes a value in the WDT_LR register regularly to prevent an instant reset or an interrupt. Users must write certain values in the WDT_LR register to update WDT_CNT value.

The timeout cycle of the WDT is set to 2,000 ms by default.

9.3.1 WDT Block Diagram

Figure 53 shows a block diagram for the WDT.

Figure 53. WDT Block Diagram



9.3.2 Enabling WDT

The WDT is always enabled after the microcontroller is powered 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.

9.3.3 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 in 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] in WDT_CON register, and the minimum frequency value of the prescaler is 1 / 256 of the clock source frequency.

Table 66 describes the prescaled frequencies of various clock sources in the WDT.

Table 66. Prescaled WDT Counter Clock Frequency

Clock Source	WDTCLKIN/ 1	WDTCLKIN/ 4	WDTCLKIN/ 16	WDTCLKIN/ 64	WDTCLKIN/ 256
LSI	40 kHz	10 kHz	2.5 kHz	625Hz	156.25Hz
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’

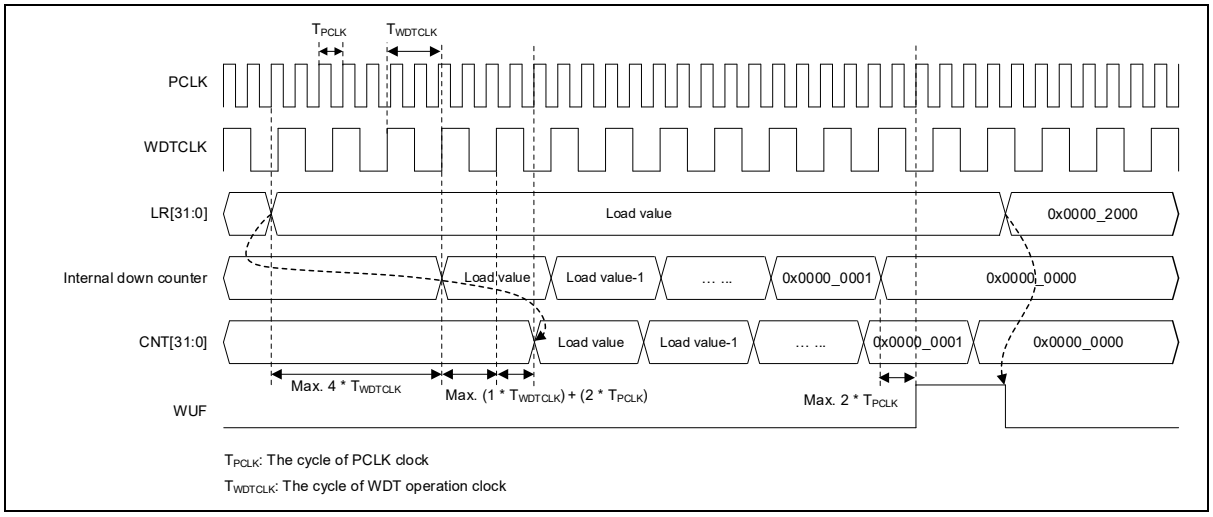
9.3.4 Interrupt Features

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 in WDT_CON register to ‘1’. At the moment when the value of the WDT_CNT down counter changes from ‘1’ to ‘0’, the interrupt is triggered and the WUF in WDT_CON register is set to ‘1’. To clear the WUF bit, users must write a value other than 0 in the WDT_LR register.

9.3.4.1 Interrupt Mode Operation Timing Diagram

Figure 54. Interrupt Mode Operation Timing Diagram with External Clock Set as WDT Clock



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).

9.3.5 How to Program Watchdog Timeout

9.3.5.1 Basic Reset Operation

To reset the WDT, users must set the WDTRE in WDT_CON register and WDTRST in SCU_RSER register to ‘1’. At the moment when the down counter reaches ‘0’ (The moment the value of the down counter changes from ‘1’ to ‘0’), the WDT generates the reset signal to reset the microcontroller.

9.3.6 Debug Mode

If the WDBG in 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 ‘1’.

Conversely, if the WDBG in 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.

9.4 WDT Registers

The base addresses and register map for the WDT are described in the tables below. The initial setting of the WDT's timeout period is 2,000 ms.

Table 67. Base Address of WDT

Name	Base Address
WDT	0x4000_0200

Table 68. WDT Register Map

Name	Offset	Type	Description	Reset Value	Reference
WDT_LR	0x0000	RW	WDT load register	0x0000_0000	9.4.1
WDT_CNT	0x0004	RO	WDT current count register	0x0000_09C4	9.4.2
WDT_CON	0x0008	RW	WDT control register	0x0000_805C	9.4.3
WDT_AEN	0x00F0	RW	WDT access enable register	0x0000_0000	9.4.4

9.4.1 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 in WDT_CON register is set to '0', the value written to the WDT_LR will remain unrepresented in the WDT_CNT until the WDTEN bit is changed to '1'.

If the WDT is being used as a reset source, the WDT_LR 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 0, it must be set to the desired value before use.

9.4.2 WDT_CNT: WDT Current Count Register

The WDT_CNT 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 while the WDTEN in 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 WDTRE in WDT_CON register and WDTRST in 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_09C4																															
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.

9.4.3 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
LP_UPDAT	Reserved															WDBG	Reserved							WUF	WDTIE	WDTRE	Reserved	WDTEN	CKSEL	WPRS[2:0]	
0	-															1	-							0	0	1	-	1	1	100	
RO	-															RW	-							RW	RW	RW	-	RW	RW	RW	

31	LP_UPDATE	A flag indicating that the WDT_LR register is being updated. If WDT_CLK is significantly slower than PCLK, an update error may occur in the WDT_LR register.	
		0	WDT_LR register can be updated (written).
		1	WDT_LR register update is not allowed. Writing to the WDT_LR register sets this bit to '1'. If an additional write operation is performed while this bit is '1', the previous written value is retained, and the current write is ignored.
15	WDBG	Whether to enable or disable 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	Whether to enable or disable the WDT counter underflow interrupt	
		0	Disables the interrupt.
		1	Enables the interrupt.
6	WDTRE	Whether to enable or disable the WDT counter underflow reset	
		0	Disables the WDT counter underflow reset.
		1	Enables the WDT counter underflow reset.
4	WDTEN	Whether to enable or disable 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

9.4.4 WDT_AEN: WDT Access Enable Register

The WDT_AEN register determines whether to allow changes 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	WO	

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 different values 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.

NOTE:

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.

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.

The instant reload feature
WDT_AEN = 0x555A; // Triggers an instant reload.
```

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.

9.4.5 WDT Register Map Summary

Table 69. 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	0	0	0	1	0	0	1	1	1	0	0	0	1	0	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	-															

10. 16-bit Timer

10.1 16-bit Timer Introduction

The A34L716 has a TIMER module, each configured with eight 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 a periodic tick timer or wake-up sources.

10.2 16-bit Timer Main Features

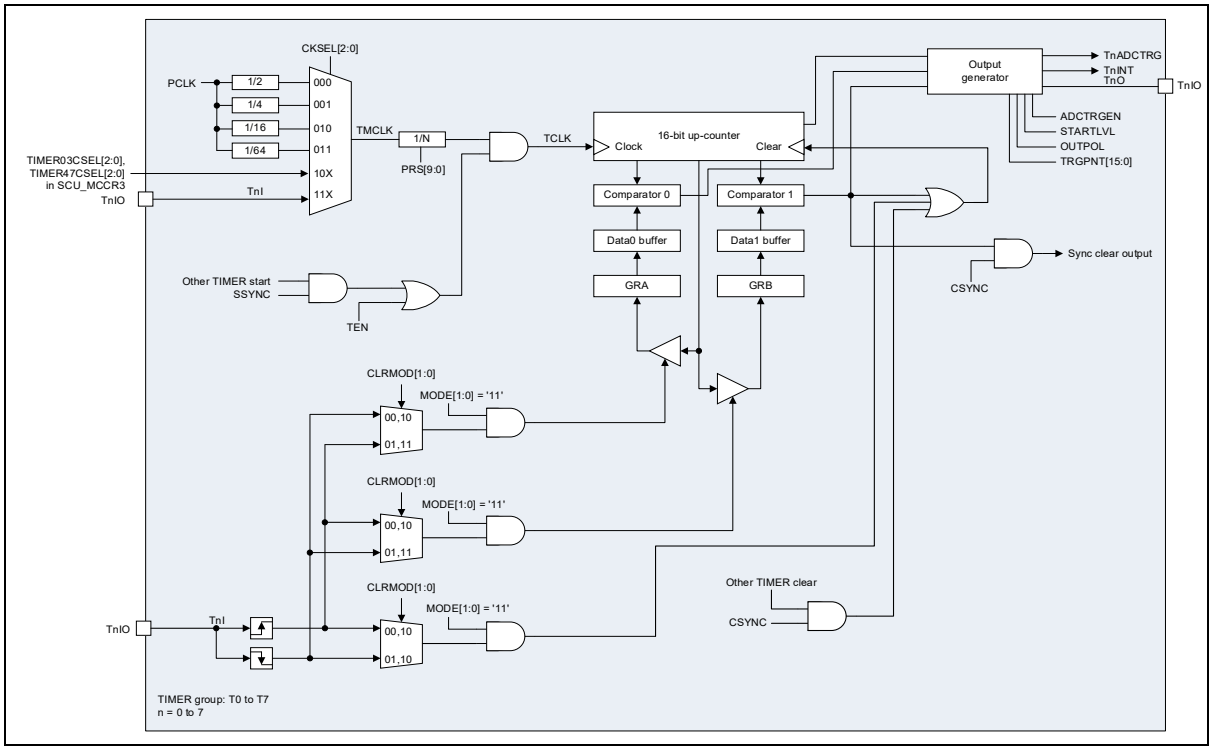
TIMERn (n = 0 to 7) includes the features below:

- 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 7)
- 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 55. 16-bit Timer Block Diagram (n = 0 to 7)



10.3.2 Pins and Internal Signals

Table 70 summarizes the I/O information of the TIMERn (n = 0 to 7).

Table 70. Input and Output Pins for TIMERn (n = 0 to 7)

Pin Name	Type	Description	Package Type		
			A34L716 (100-LQFP)	A34L716 (64-LQFP)	A34L716 (56-QFN)
T0IO	IO	TIMER0 capture input signal external clock input TIMER0 periodic / pwm / one-shot output	O	O	X
T1IO	IO	TIMER1 capture input signal external clock input TIMER1 periodic / pwm / one-shot output	O	O	X
T2IO	IO	TIMER2 capture input signal external clock input TIMER2 periodic / pwm / one-shot output	O	X	X
T3IO	IO	TIMER3 capture input signal external clock input TIMER3 periodic / pwm / one-shot output	O	X	X
T4IO	IO	TIMER4 capture input signal external clock input TIMER4 periodic / pwm / one-shot output	O	X	X
T5IO	IO	TIMER5 capture input signal external clock input TIMER5 periodic / pwm / one-shot output	O	X	X
T6IO	IO	TIMER6 capture input signal external clock input TIMER6 periodic / pwm / one-shot output	O	O	O
T7IO	IO	TIMER7 capture input signal external clock input TIMER7 periodic / pwm / one-shot output	O	O	O

Table 71. Internal Input and Output Signals for TIMERn (n = 0 to 7)

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
TnADCTRG	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 7.

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 7) for timer clock source

Users can select a clock source by configuring the CKSEL[2:0] in TIMERn_CR (n = 0 to 7).

10.3.3.1 Peripheral System Clock (PCLK)

Users can select a timer input clock source using 4 PCLK divide levels (PCLK / 2, PCLK / 4, PCLK / 16, PCLK / 64). If the CKSEL[2:0] in TIMERn_CR1 is set to a number between 0 and 3, the timer clock is provided by the PCLK divided level.

Table 72. PCLK Clock Selection by 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] in TIMERn_CR1 is set to ‘10X’, the timer clock is provided by the SCU_MCCR3 register.

Table 73. SCU_MCCR3 Register and CKSEL[2:0] in TIMERn_CR1

CKSEL[2:0] in TIMERn_CR1	TIMERn Clock Source Selection
10X	EXT_CLK (SCU_MCCR3 register)

The 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 7). If the IOSEL in TIMERn_CR1 is set to 0, the TnIO pin is set as input.

Table 74. 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] in TIMERn_CR1 is set to '11X', the timer clock is provided by input on port TnIO.

Table 75. 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 7)

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 is incremented based on the specified input clock. The counter clock can be divided by a prescaler.

The time-base unit includes the following registers:

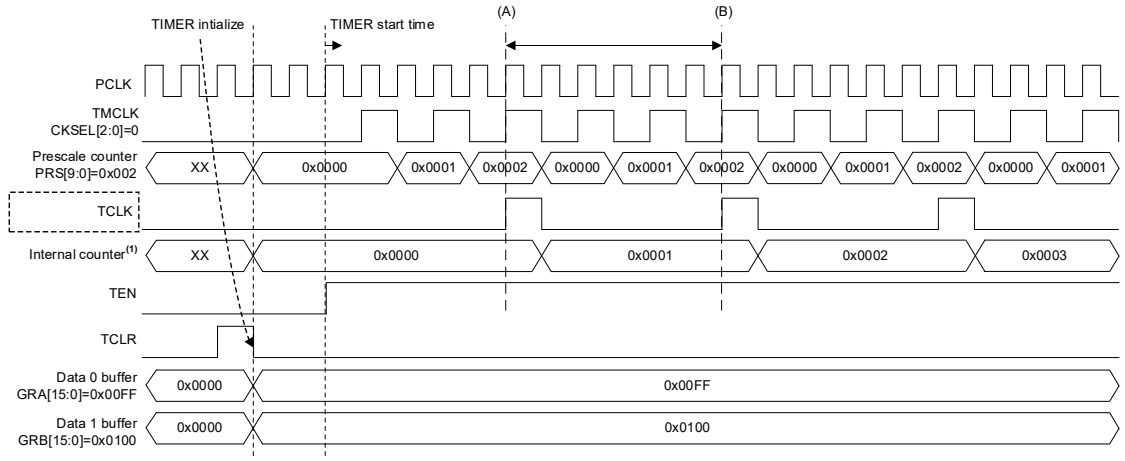
- Timer counter register: TIMERn_CNT (n = 0 to 7)
- Timer prescaler register: TIMERn_PRS (n = 0 to 7)
- Timer general data register A: TIMERn_GRA (n = 0 to 7)
- Timer general data register B: TIMERn_GRB (n = 0 to 7)
- Miscellaneous clock control register 3: SCU_MCCR3

10.3.4.1 Prescaler Description

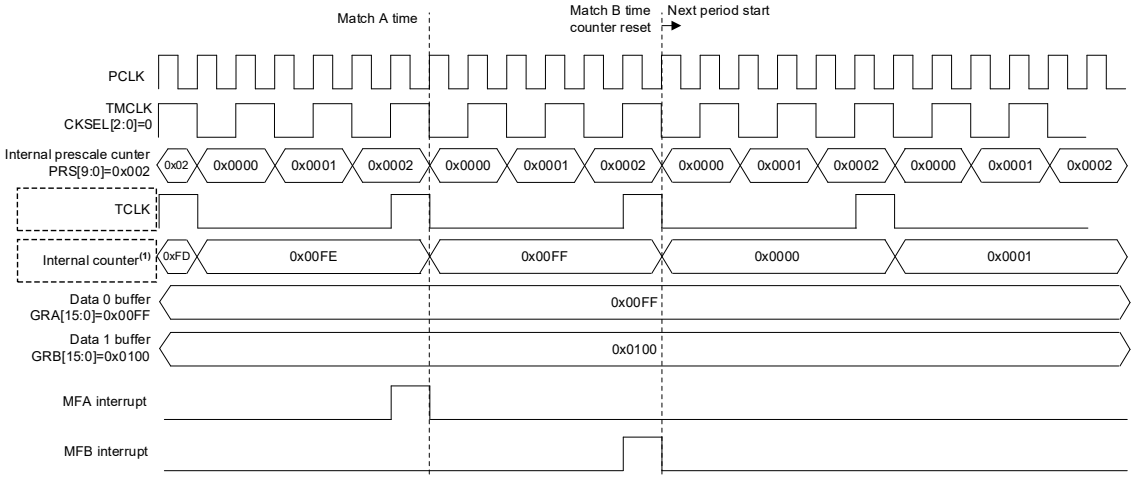
The prescaler is used to set the timer input frequency divider. It is 10-bit wide. Users can generate precise and varied timer base clocks by applying the prescaler to the timer clock source that has been selected in TIMERn_CR1 register.

The TMCLK shown in Figure 56 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 56. 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'.

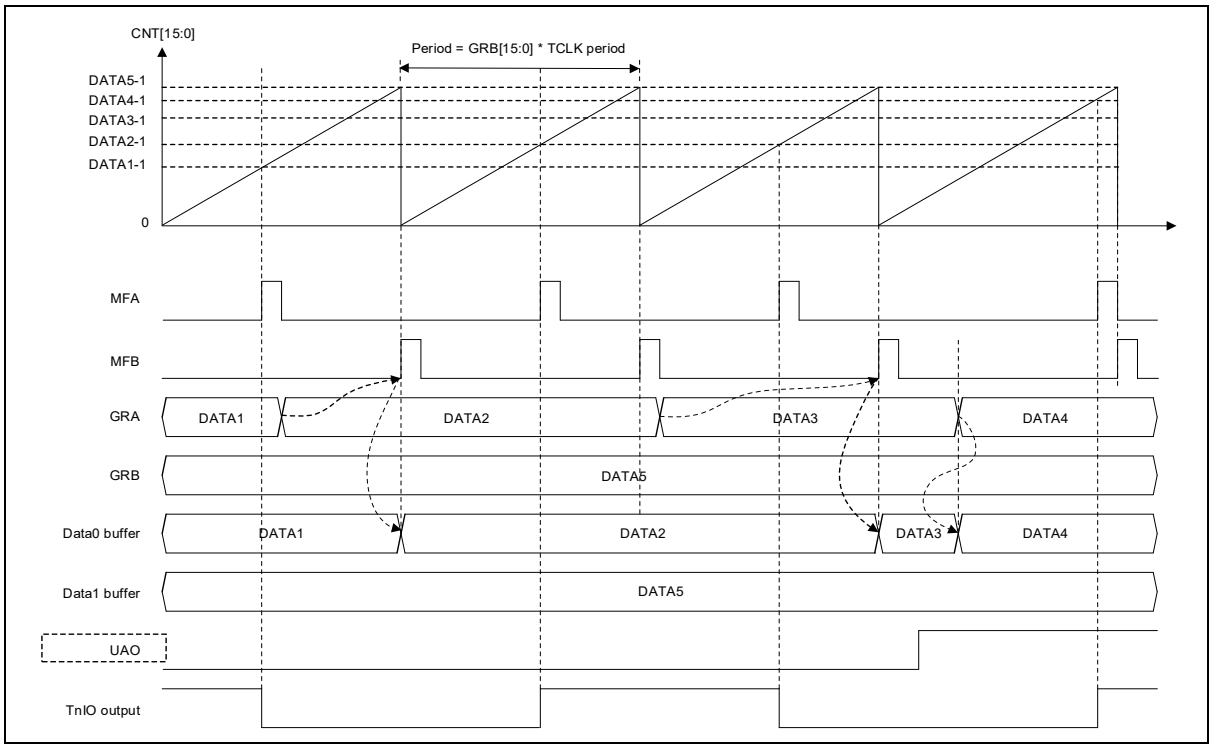
NOTE:

1. Internal counter clock period (A~B) is PCLK * (2* (2 + 1)), that is, 6 * PCLK.

10.3.5 Periodic Mode

Figure 57 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 57. Normal Periodic Mode Operation



The timer's count period can be calculated by using the formula below:

$$\boldsymbol{Period = TCLK\ Period \times TIMERn\_GRA\ value}$$

$$\boldsymbol{Match\ A\ Interrupt\ Time = TCLK\ Period \times TIMERn\_GRA\ value}$$

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 0.

The 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 in TIMERn_CR1 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 in TIMERn_CR2 register or the GBD match event occurred.

When the UAO in TIMERn_CR1 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 0 and the TIMERn_GRB value is not 0, 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 in TIMERn_CR1

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 76 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 76. Operations Comparison According to TIMERn_GRA/GRB Settings

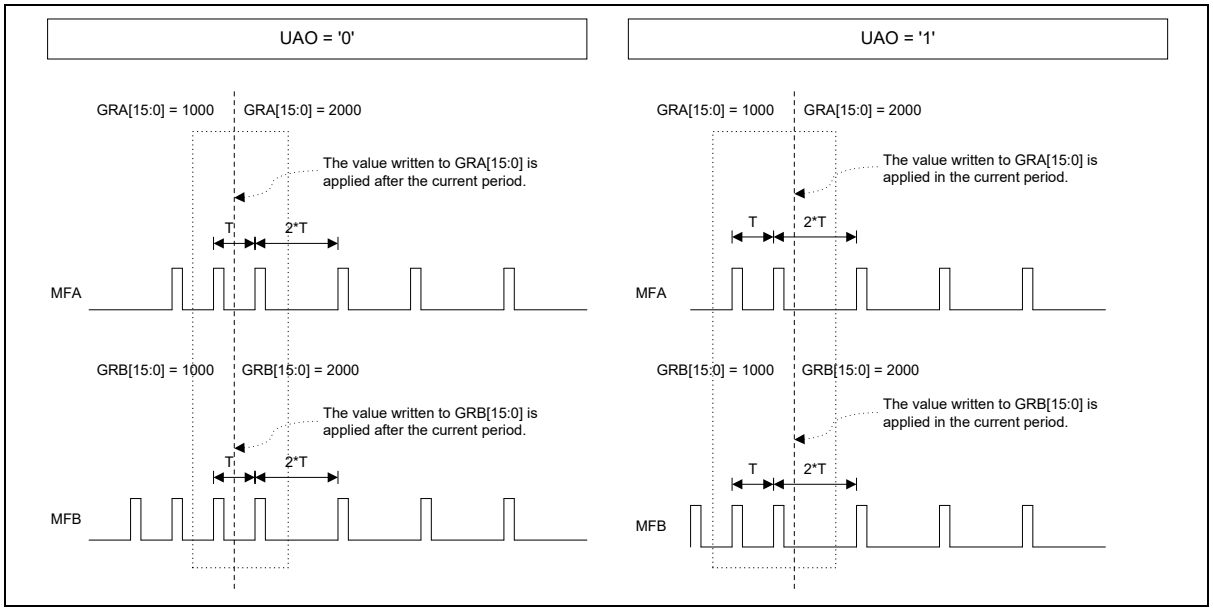
Case	GRA[15:0]	GRB[15:0]	UAO	OUTPOL	IOSEL	STARTLVL	TnIO Pin Output
1	1000	1000	0/1	0	1	0	Normal operation
2	1000	2000	0/1	0	1	0	Outputs the GRB[15:0] setting value.
3	1000	0	0/1	0	1	0	Outputs the GRB[15:0] setting value.
4	1000	1 to 999	0/1	0	1	0	No output
5	1000	65535	0/1	0	1	0	Outputs the GRB[15:0] setting value.

10.3.5.2 Operation According to UAO in TIMERn_CR1, OUTPOL in TIMERn_CR1, and STARTLVL in TIMERn_CR1 Settings

Refer to the Figure 58 for the operation waveform according to the UAO in TIMERn_CR1 settings.

- If the UAO 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 in TIMERn_CR1 value is ‘1’, the TIMERn_GRA / GRB value is applied within the current cycle operation.

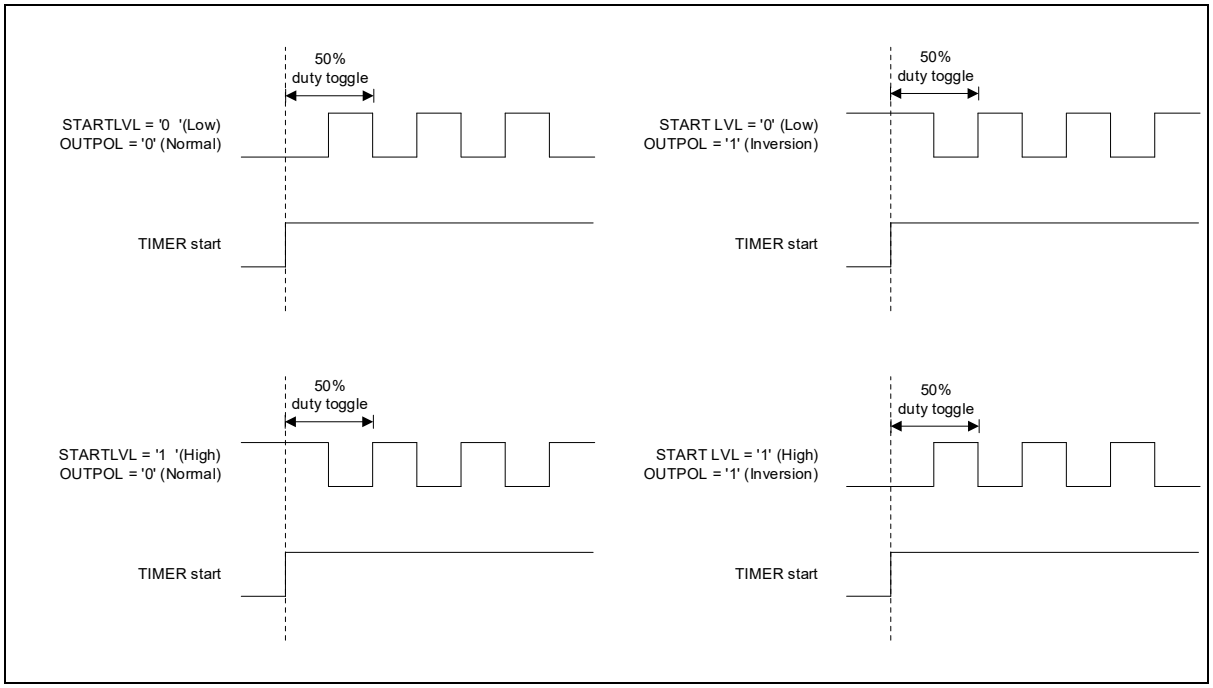
Figure 58. Operation Waveform Depending on UAO



Refer to the Figure 59 below for the operation waveform according to the STARTLVL in TIMERn_CR1 , OUTPOL in TIMERn_CR1 settings.

- The STARTLVL in TIMERn_CR1 is reflected in TnIO by TCLR in TIMERn_CR2 (n = 0 to 7).
- The OUTPOL in TIMERn_CR1 register is applied when the timer counter is cleared after setting. (Applied after the current cycle operation)

Figure 59. 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 7) 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 7) 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 in TIMERn_CR1.

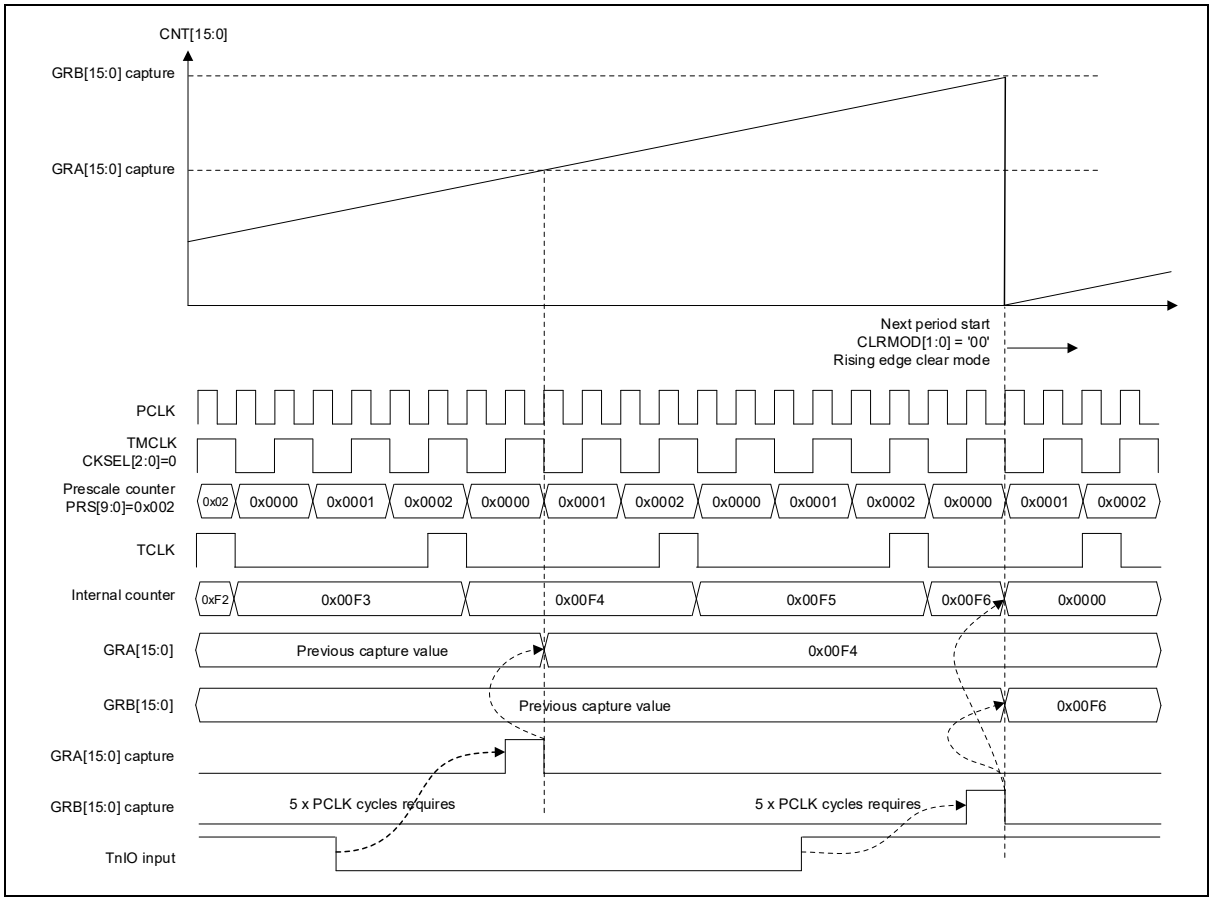
10.3.5.4 Counter Mode Register Setting Sequence

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 register1 (SCU_PER1) and the peripheral clock enable register1 (SCU_PCER1).
3. Write '0' to the TIMERn_CNT register to initialize the timer counter.
4. To set the timer counter mode, set the corresponding mode value to '00' in the MODE[1:0] field of TIMERn_CR1.
5. To set the timer clock source, enter the corresponding set value in the CKSEL[2:0] in TIMERn_CR1.
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 in TIMERn_CR1 to set the timer input / output (TnIO, n = 0 to 7) pin.
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 in TIMERn_IER to set the timer match interrupt.
11. Write '1' to the TLCR in TIMERn_CR2 to apply current timer setting.
12. Write '1' to the TEN in TIMERn_CR2 to start the timer operation.

10.3.6 Capture Mode

Figure 60 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 60. 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] in 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 60 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 function pin mode (TnIO, n = 0 to 7) and then set the IOSEL in TIMERn_CR1 to ‘0’.

Table 77. 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

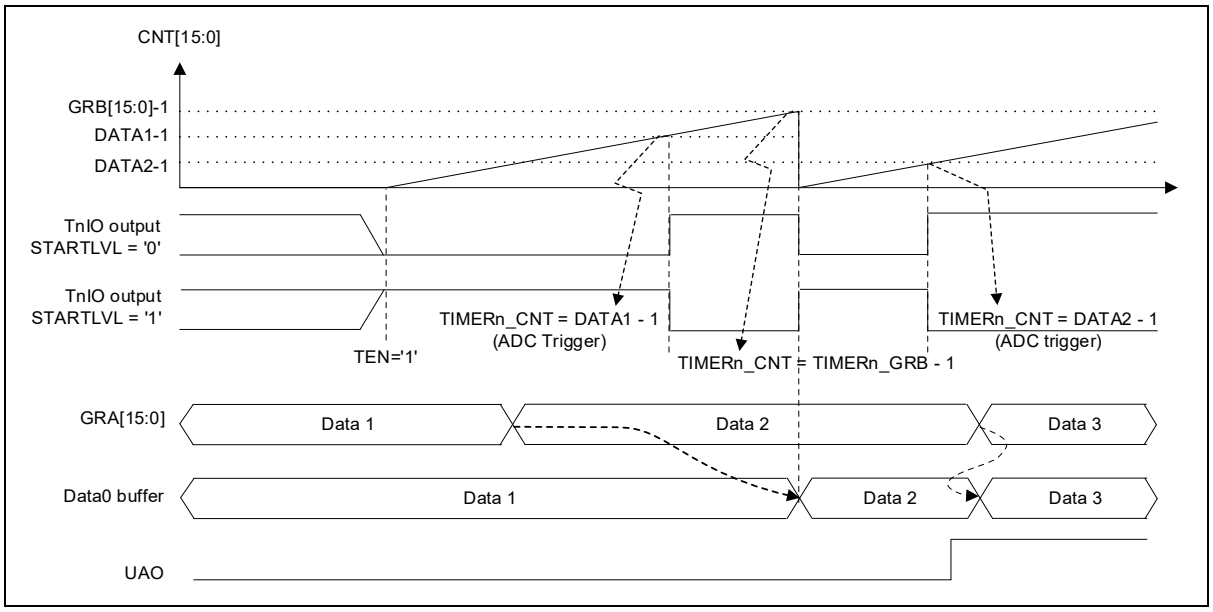
10.3.6.1 Capture Mode Register Setting Sequence

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 0.
4. Write ‘0’ to the TIMERn_CNT register to initialize the timer counter.
5. To set the timer capture mode, set the corresponding mode value to ‘11’ in the MODE[1:0] in TIMERn_CR1.
6. To set the timer clock source, enter the corresponding set value in the CKSEL[2:0] in TIMERn_CR1.
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 in TIMERn_CR1 to set the timer input / output (TnIO, n = 0 to 7) pin.
9. For the clear mode selection in capture mode, enter the corresponding set value in the CLRMOD[1:0] in TIMERn_CR1.
10. Write ‘1’ to the TLCR in TIMERn_CR2 to apply current timer setting.
11. Write ‘1’ to the TEN in TIMERn_CR2 to start the timer operation.

10.3.7 PWM Mode

Figure 61 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 7).

Figure 61. Timing Diagram of PWM Output Operation



The PWM pulse period can be calculated by using the formula below:

$$\boldsymbol{Period = TCLK\ Period \times TIMERn\_GRB\ value}$$

$$\boldsymbol{Duty = TCLK\ Period \times TIMERn\_GRA\ value}$$

If the TIMERn_GRB value is 0, the timer cannot be started even if the TEN in TIMERn_CR2 is set to '1' because the period is 0.

The 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 PWM mode with the UAO in TIMERn_CR1 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 in TIMERn_CR2 register or the GRB match event occurred.

When the UAO in TIMERn_CR1 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 in TIMERn_CR1. 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.

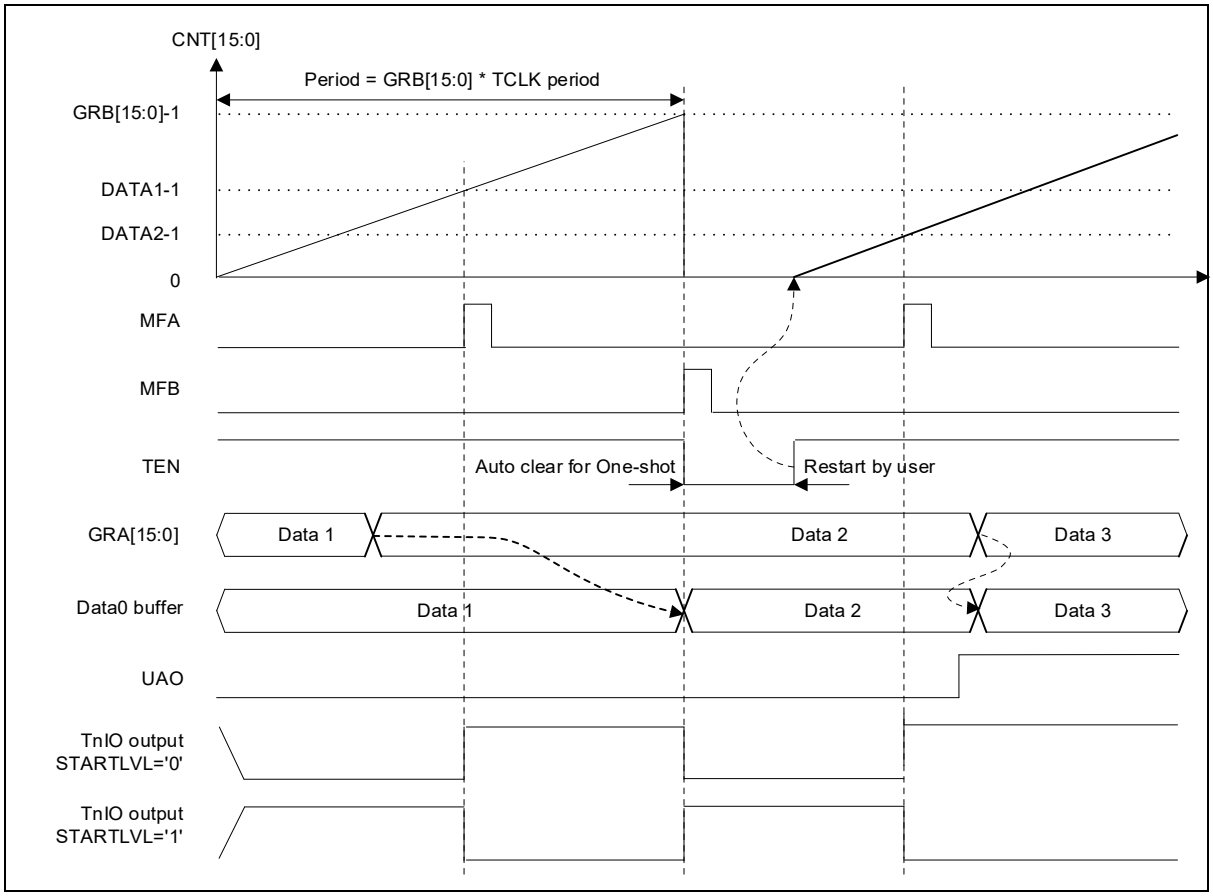
10.3.7.2 PWM Mode Register Setting Sequence

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 ‘0’ to the TIMERn_CNT register to initialize the timer counter.
4. To set the timer capture mode, set the corresponding mode value to ‘01’ in the MODE[1:0] in TIMERn_CR1.
5. To set the timer clock source, enter the corresponding set value in the CKSEL[2:0] in TIMERn_CR1.
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 in TIMERn_CR1 to set the timer input / output (TnIO, n = 0 to 7) pin.
8. Enter the set values in the TIMERn_GRA and TIMERn_GRB registers for PWM operation.
9. Write ‘1’ to the TLCR in TIMERn_CR2 to apply current timer setting.
10. Write ‘1’ to the TEN in TIMERn_CR2 to start the timer operation.

10.3.8 One-Shot Mode

Figure 62 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 62. Timing Diagram of One-Shot Mode Operation



The one-shot count period can be calculated by using the formula below:

$$\boldsymbol{Period = TCLK\ Period \times TIMERn\_GRB\ value}$$

$$\boldsymbol{Duty = TCLK\ Period \times TIMERn\_GRA\ value}$$

If the TIMERn_GRB value is 0, the timer cannot be started even if the TEN in TIMERn_CR2 is set to ‘1’ because the period is 0.

The 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 in TIMERn_CR1 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 in TIMERn_CR2 register or the GRB match event occurred.

When the UAO in TIMERn_CR1 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 are shown below:

- The TIMERn_GRA value must be greater than ‘0’ and less than or equal to the TIMERn_GRB value.

Operation sequence according to the STARTLVL in TIMERn_CR1 settings are shown below:

- 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 78. One-Shot Mode Operation View Table

Case	GRA[15:0]	GRB[15:0]	UAO	POL	IOSEL	STARTLVL	Remarks
#1	500	2000	0	0(N)	1(Out)	0	Initial output: Low High pulse output between GRA[15:0] and GRB[15:0]
#2	500	2000	0	0(N)	1(Out)	1	Initial output: High Low pulse output between GRA[15:0] and GRB[15:0]
#3	500	2000	0	1(I)	1(Out)	0	Initial output: High Low pulse output between GRA[15:0] and GRB[15:0]
#4	500	2000	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 ‘0’ to the TIMERn_CNT register to initialize the timer counter.
4. To set the timer capture mode, set the corresponding mode value to ‘10’ in the MODE[1:0] in TIMERn_CR1.
5. To set the timer clock source, enter the corresponding set value in the CKSEL[2:0] in TIMERn_CR1.
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 in TIMERn_CR1 to set the timer input / output (TnIO, n = 0 to 7) pin.
8. Enter the set values in the TIMERn_GRA and TIMERn_GRB registers for One-shot operation.
9. Write ‘1’ to the TLCR in TIMERn_CR2 to apply current timer setting.
10. Write ‘1’ to the TEN in TIMERn_CR2 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 63, Figure 64, and Figure 65 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 63. Timing Diagram of ADC Triggering (TRGMOD[1:0] = '00')

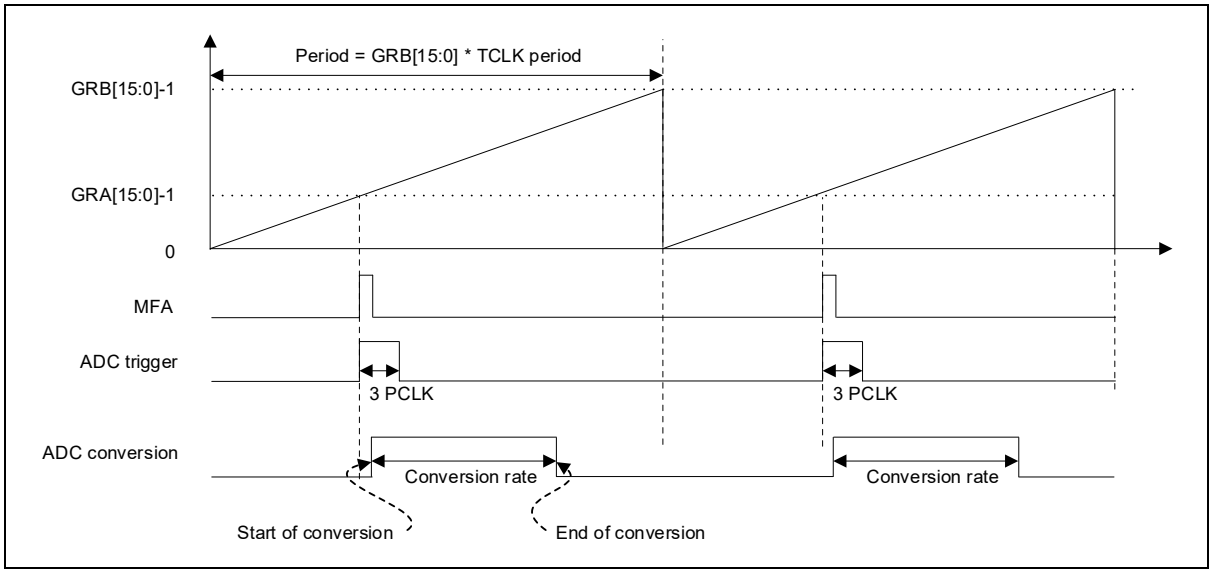


Figure 64. Timing Diagram of ADC Triggering (TRGMOD[1:0] = '10')

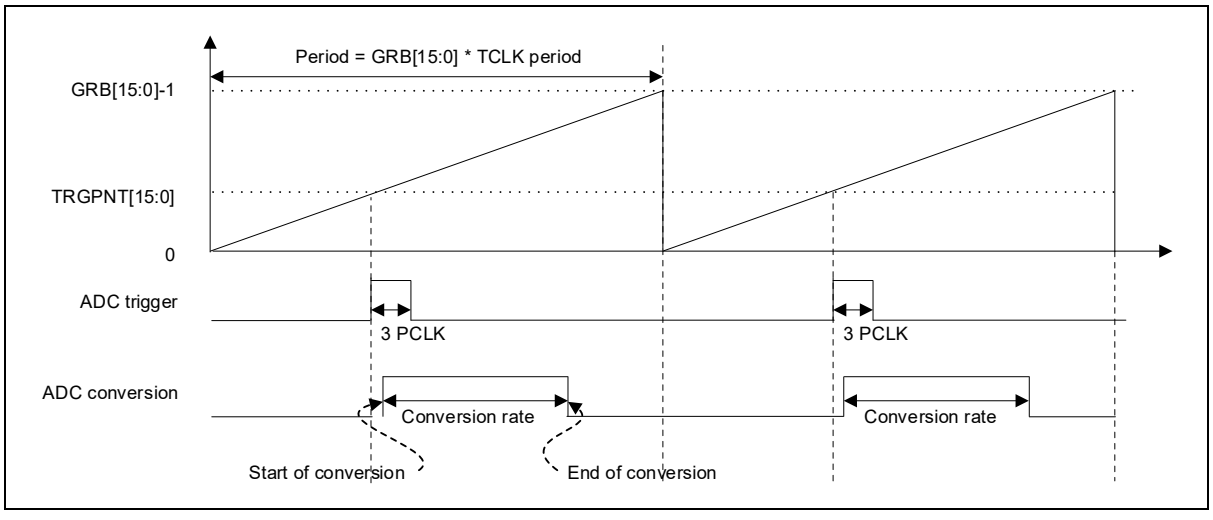
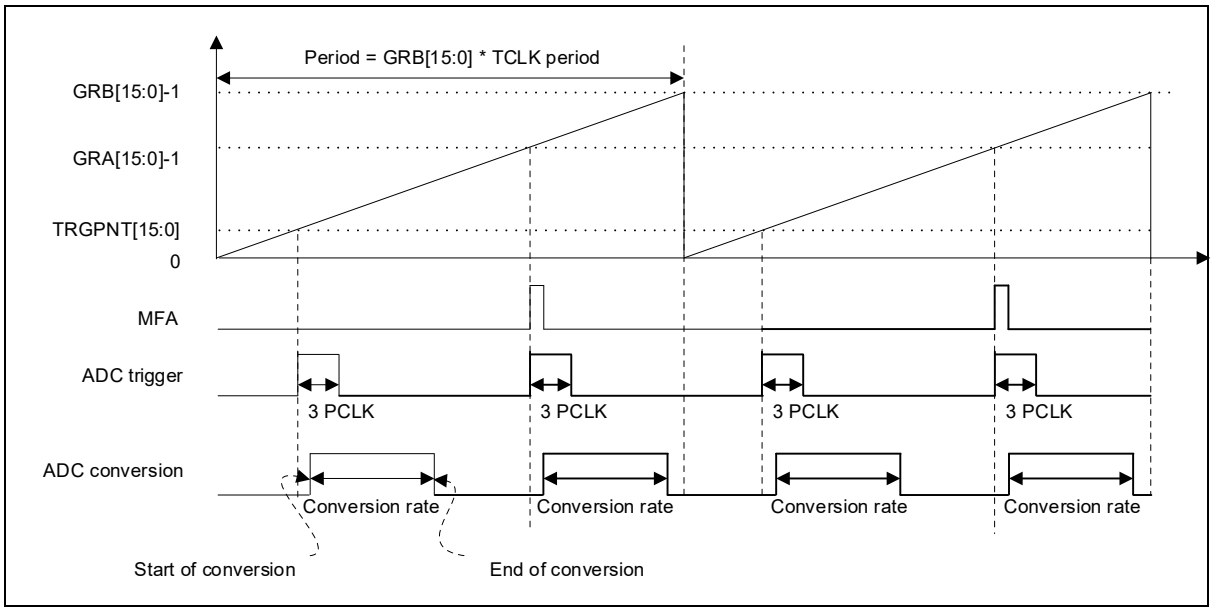


Figure 65. Timing Diagram of ADC Triggering (TRGMOD[1:0] = '11')



10.3.9.1 ADC Trigger Operation Characteristics and Precautions

If the TIMERn_GRB value is less than or equal to the TIMERn_GRA value or the TIMERn_TRGPNT value, the timer counter is cleared whenever timer counter matches with TIMERn_GRB value. So, the matching event does not occur.

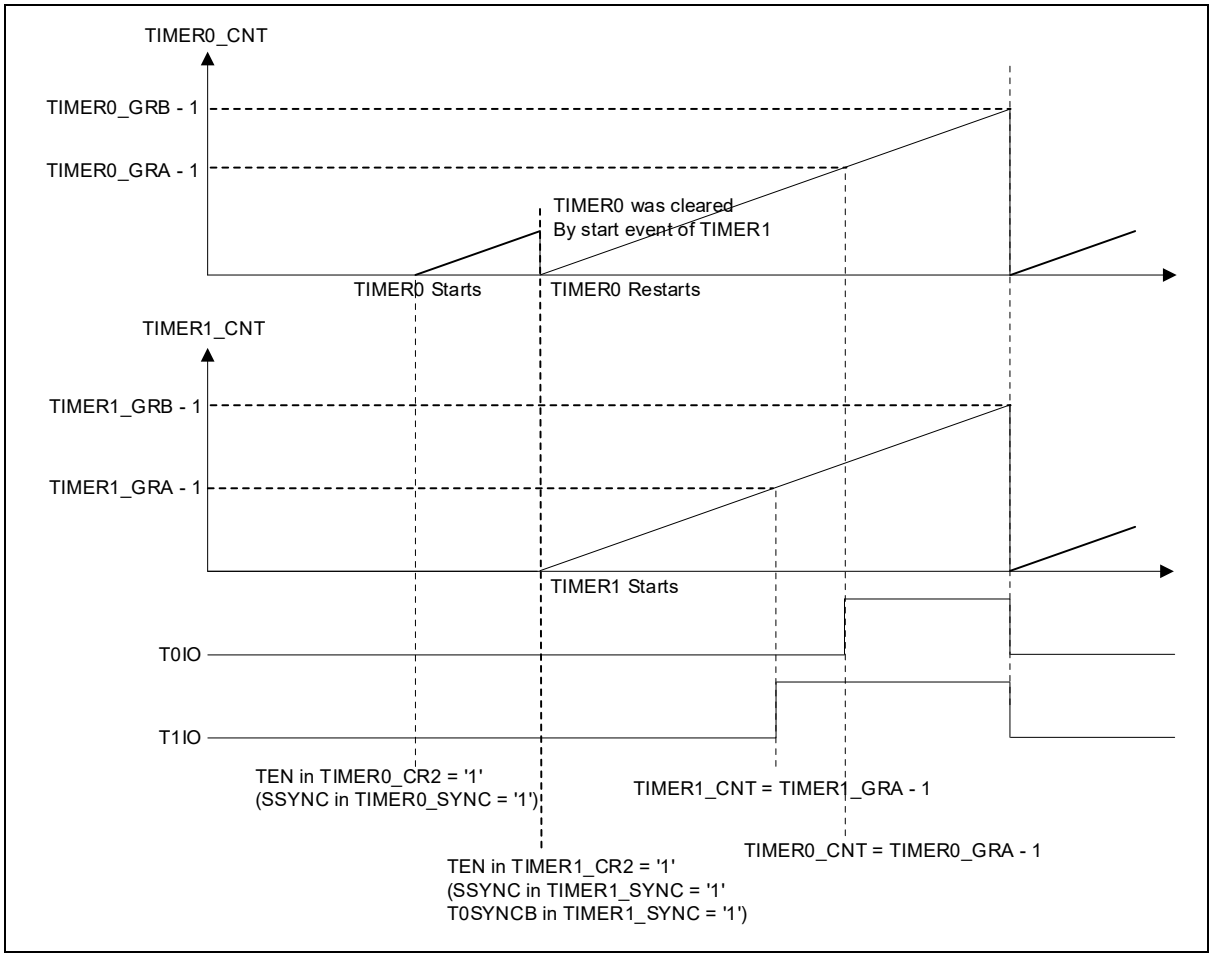
10.3.9.2 Register Setting Sequence for ADC Trigger Operation

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 ‘0’ 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] in TIMERn_CR1.
5. To set the timer clock source, enter the corresponding set value in the CKSEL[2:0] in TIMERn_CR1.
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 in TIMERn_CR1 to use the trigger source.
9. Enter the corresponding mode set value in the TRGMOD[1:0] in TIMERn_CR1 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] in TIMERn_CR1, 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] in ADC_MR to ‘01’.)
12. Write ‘1’ to the TLCR in TIMERn_CR2 to apply current timer setting.
13. Write ‘1’ to the TEN in TIMERn_CR2 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 66 shows synchronous PWM generation.

Figure 66. Timer Synchronization Example (when SSYNC = '1', CSYNC = '1')



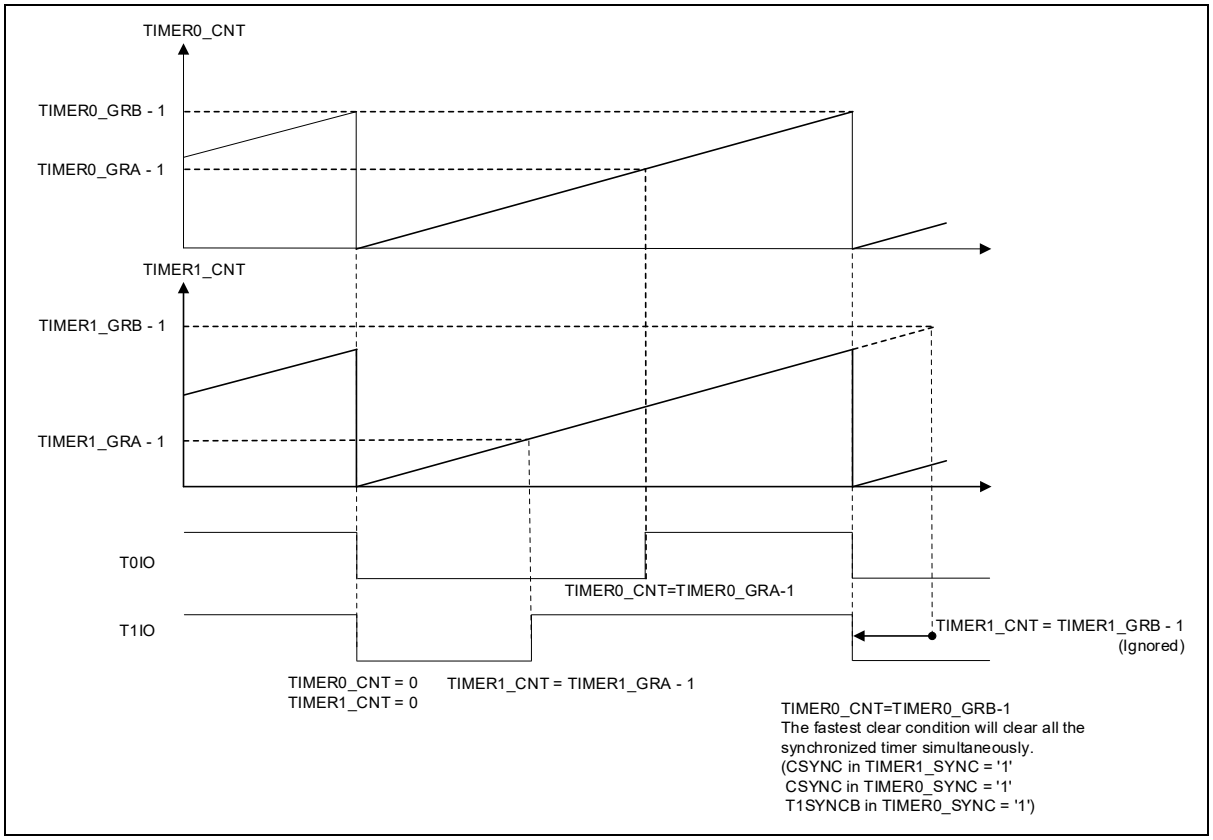
The SSYNC in TIMERn_SYNC controls start synchronization with other timer modules. In the above figure, the TIMER0 (Slave Timer) waveform shows that the timer's start is synchronized to the start of the signal from TIMER1 (Master Timer).

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 must be set to '1'. In the Slave Timer's TIMER0_SYNC register, the SSYNC 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 in TIMER0_CR2 is set to '0'.

Figure 67. Timer Synchronization Example (when CSYNC = '1')



The CSYNC in TIMERn_SYNC controls clear synchronization with other timer modules. In the above figure, the TIMER1 (Slave Timer) waveform shows that the timer's clarity 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 must be set to '1'. In the Slave Timer's TIMER1_SYNC register, the CSYNC must be set to '1'.

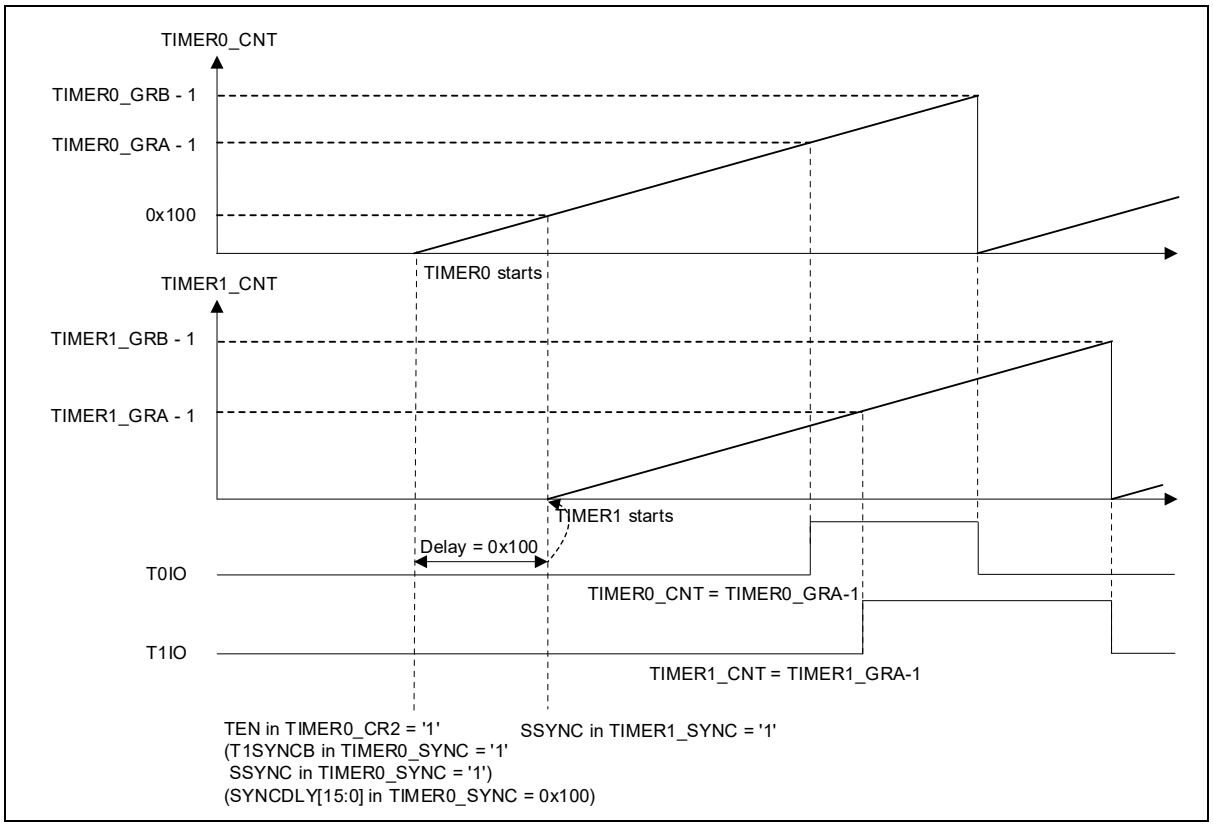
If only the Master Timer's synchronous clear bit is set enabled for the Slave Timer while the Slave's synchronous clear bit is disabled, the Slave Timer can run independently without being affected by the Master Timer's clear synchronization signal.

10.3.10.1 Delayed Timer Synchronization

The PWM delayed synchronization function is used between timers within the same group (Group 1: T0, T1, ..., T7); synchronization is delayed by the amount of time written to the SYNCDLY[15:0] in 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 68. Delayed Timer Synchronization Example (when SSYNC = '1')



For example as shown in Figure 68, 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 in TIMER1_SYNC to '1'.

Additionally, because the TIMER1 (Slave) operates as synchronized to the TIMER0's (Master's) start signal, it runs even if the TEN in TIMER0_CR2 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 in TIMERn_CR2 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 in TIMERn_CR2 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 69 and Figure 70 show the operational 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 69. SSYNC Operation Example

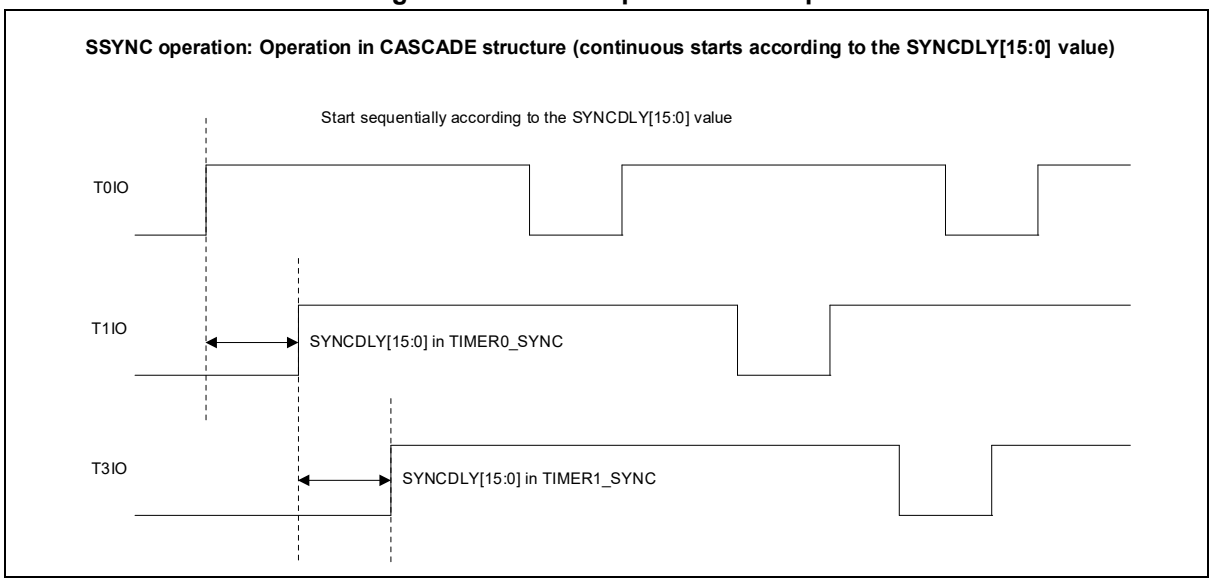
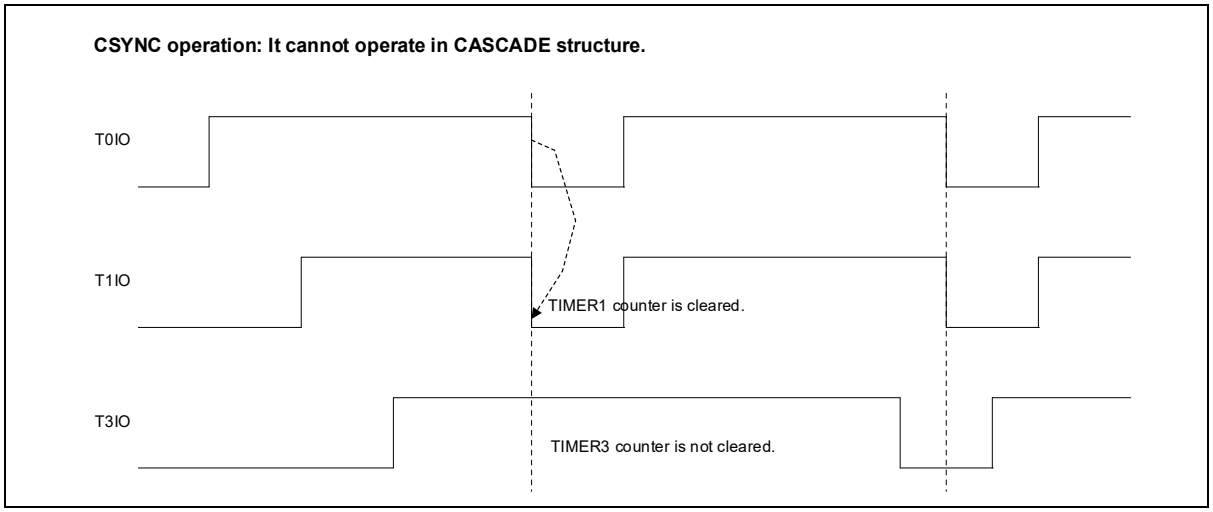


Figure 70. CSYNC Operation Example



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 '0' 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] TIMERn_CR1.
5. To set the timer clock source, enter the corresponding set value in the CKSEL[2:0] in TIMERn_CR1.
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 in TIMERn_CR1 to set the timer input / output (TnIO, n = 0 to 7) pin.
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 in TIMERn_SYNC (n = 0 to 7) or the CSYNC in TIMERn_SYNC.
    A. When using a synchronized counter start mode, set the SSYNC in TIMERn_SYNC.
    B. When using a synchronized counter clear mode, set the CSYNC in TIMERn_SYNC.
10. To use a sync delay, enter the corresponding set value in the SYNCDLY[15:0] in TIMERn_SYNC.
11. Repeat steps 1 through 10 above to set all timers that use the synchronization function. And write '1' to the TLCR in TIMERn_CR2 to apply current timer setting.
12. For the timer operating as the Master, enter the corresponding set value in the TnSYNCB (n = 0 to 7) in TIMERn_SYNC to select the Slave Timer.
13. Write '1' to the TLCR in TIMERn_CR2 to apply current timer setting.
14. Write '1' to the TEN in TIMERn_CR2 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 in TIMERn_CR1 register. Setting the IOSEL determines the timer input / output pin, and setting the OUTPOL selects the polarity between the normal output and inversion output. Setting the STARTLVL 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 7) are enabled, although the core halted, when the TEN in TIMERn_CR2 is set. When the TEN in TIMERn_CR2 is cleared, the outputs (TnIO, n = 0 to 7) are disabled.

10.3.13 Interrupts

The TIMERn can generate interrupts, as shown in Table 79. (n= 0 to 7)

Table 79. 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 addresses for the 16-bit timer are described in the tables below:

Table 80. 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

Table 81. 16-bit Timer Register Map

Name	Offset	Type	Description	Reset Value	Reference
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, 4, 5, 6 and 7.

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 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

31 30 29 28 27 26 25 24	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
	0	Sets the pin as an input port
	1	Sets the pin as an output port
8	ADCTRGEN	Whether or not to 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
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	MODE[1:0]	Timer operation mode control

0	00	Normal periodic mode
	01	PWM mode
	10	One-shot mode
	11	Capture mode

NOTE:

1. If the EXT_CLK is set in CKSEL[2:0] of TIMERn_CR 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

31 30 29 28 27 26 25 24	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[(1)]	Clearing of the timer count register	
		0	Does not clear the register.
		1	Initializes the timer. Writing a '1' to TCLR clears the counter register. (Automatically cleared to '0' after operation.)
0	TEN	Whether to enable or disable the timer	
		0	Stop the timer.
		1	Start the timer.

NOTE:

1. To prevent timing errors, the TCLR bit must be set to '1' before starting the timer.

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. The user 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

31 30 29 28 27 26 25 24	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

31 30 29 28 27 26 25 24	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

31 30 29 28 27 26 25 24	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 is a 16-bit register. 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

31 30 29 28 27 26 25 24	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

31 30 29 28 27 26 25 24	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 it.)
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 it.)
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 it.)

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_SR=0x4000_301C, TIMER1_SR=0x4000_305C
TIMER2_SR=0x4000_309C, TIMER3_SR=0x4000_30DC
TIMER4_SR=0x4000_311C, TIMER5_SR=0x4000_315C
TIMER6_SR=0x4000_319C, TIMER7_SR=0x4000_31DC

31 30 29 28 27 26 25 24	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
-				RWC1	RWC1	RWC1

2	MAIE	Whether to enable or disable the TIMERn_GRA match interrupt	
		0	Disables the TIMERn_GRA match interrupt.
		1	Enables the TIMERn_GRA match interrupt.
1	MBIE	Whether to enable or disable the TIMERn_GRB match interrupt	
		0	Disables the TIMERn_GRB match interrupt.
		1	Enables the TIMERn_GRB match interrupt.
0	OVIE	Whether to enable or disable 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

31 30 29 28 27 26 25 24	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] values are matched.

10.4.10 TIMERn_SYNC: Timer/Counter n Sync Configuration Register

TIMERn_SYNC is used to synchronize TIMER0 to TIMER7 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

31 30 29 28	27	26	25	24	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	T7SYNCB	T6SYNCB	T5SYNCB	T4SYNCB	T3SYNCB	T2SYNCB	T1SYNCB	T0SYNCB	Reserved	SSYNC	CSYNC	SYNCDLY[15:0]	
-	0	0	0	0	0	0	0	0	-	0	0	0x0000	
-	RW	RW	RW	RW	RW	RW	RW	RW	-	RW	RW	RW	

20+n	TnSYNCB (n = 0 to 7)	Synchronization settings to Tn: As a master, the timer must select a slave timer that is enabled to synchronize. The slave timer must not select a master timer.	
		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. When synchronization with TIMER1 is enabled in TIMER0_SYNC and synchronization with TIMER2 is enabled in TIMER1_SYNC, TIMER0, TIMER1, and TIMER2 start successively with the delay time set in SYNCDLY[15:0] between each timer.

10.4.11 Timer Register Map Summary

Table 82. 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	Res.	Res.	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.	Res.	Res.	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

11. Free-Run Timer (FRT)

11.1 FRT Introduction

A34L716 has one 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 A34L716 has the main features shown below:

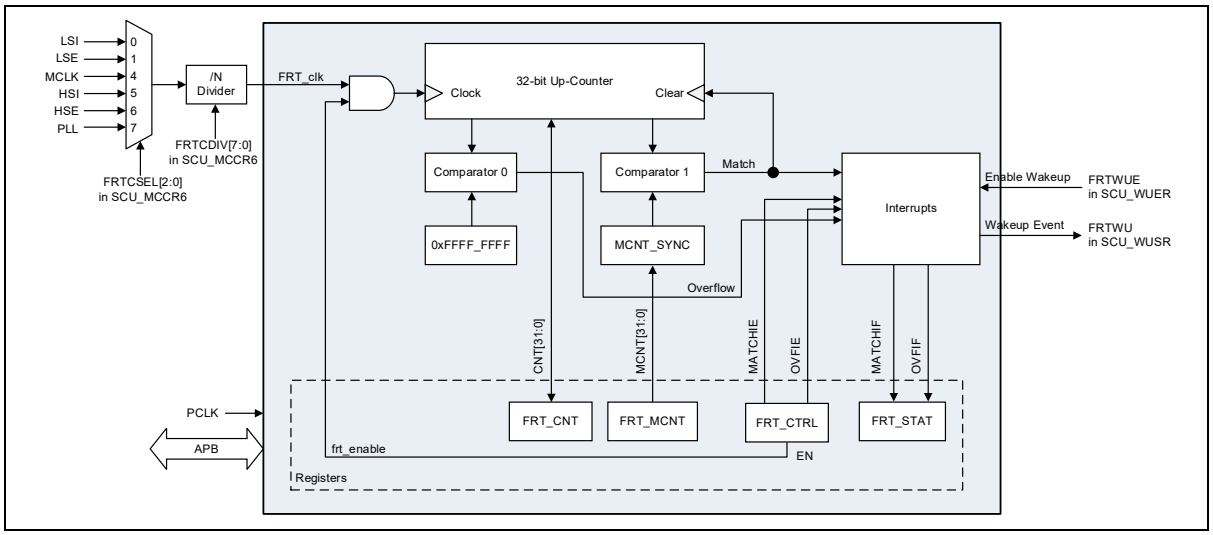
- 32-bit up count timer’s two operations
  - Free-run Timer Mode operation
  - Periodic Timer Mode operation based on the setup period
- FRT overflow interrupt and match interrupt functions
- Selectable FRT input clocks
  - Input clock source can be selected using the SCU_MCCR6 register: LSI, LSE, MCLK, HSI, HSE, or PLL

11.3 FRT Functional Description

11.3.1 FRT Block Diagram

Figure 71 shows a block diagram of the FRT.

Figure 71. FRT Block Diagram



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 FRT in SCU_PER1 and SCU_PCER1 register to ‘1’. Until the clock is provided to the FRT module, it is not reset and does not operate 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 FRTCDIV[7:0] in SCU_MCCR6 register is set to a certain value larger than ‘0’ and the FRTCSEL[2:0] in 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 A34L716 cannot wake up from Deep Sleep 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 in FRT_CTRL register determines whether the FRT operates in Periodic Timer Mode or Free-run Timer Mode.

When the FRT runs in Periodic Timer Mode, the MATCHIE in FRT_CTRL register determines whether to enable or disable the match interrupt. This bit does not affect the overflow interrupt.

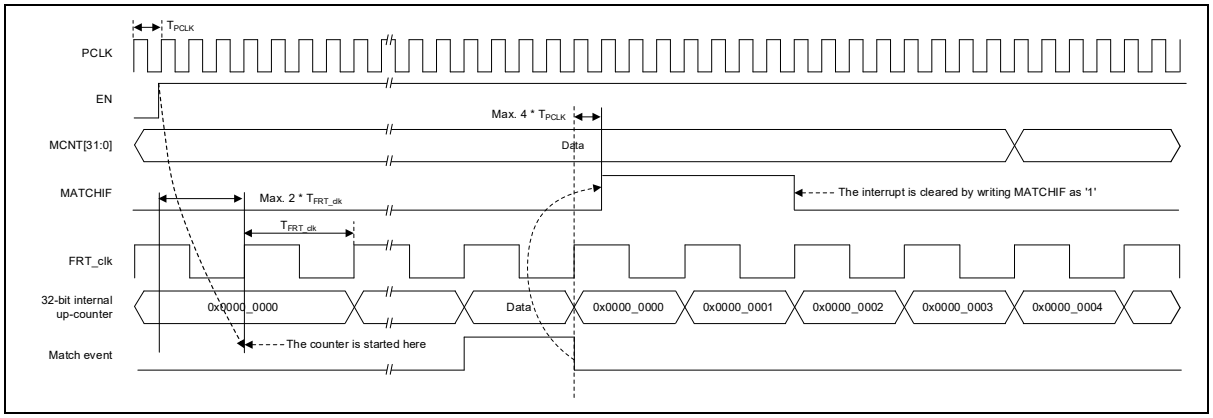
When the FRT runs in Free-run Timer Mode, the OVFIE in FRT_CTRL register determines whether to enable or disable the overflow interrupt. This bit does not affect the match interrupt.

11.3.3.1 Match Interrupt Operation

Figure 72 shows a timing diagram of the match interrupt operation. To enable the match interrupt, the MATCHIE in FRT_CTRL register must be set to ‘1’.

When the EN in FRT_CTRL register is set to ‘1’, the FRT counter starts counting. Once the counter value reaches the FRT_MCNT value, the FRT counter is clear t 0 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 72. Match Interrupt Operation Timing Diagram



The following formula calculates the FRT’s count period in Periodic Timer Mode:

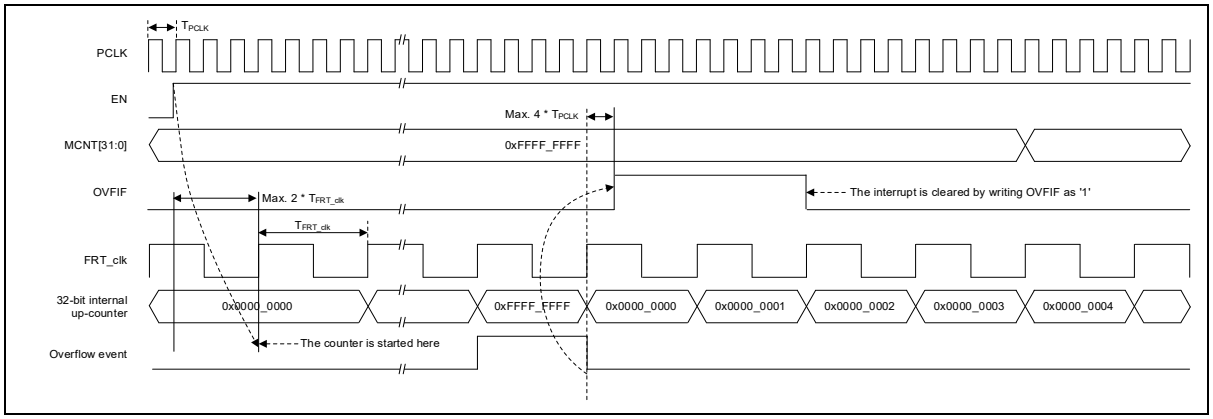
$$\boldsymbol{Period = FRT\_CLK\ Period \times (MCNT + 1)}$$

$$\boldsymbol{Match\ Interrupt\ Time = FRT\_CLK\ Period \times (MCNT + 1)}$$

11.3.3.2 Overflow Interrupt Operation

Figure 73 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 73. Overflow Interrupt Operation Timing Diagram



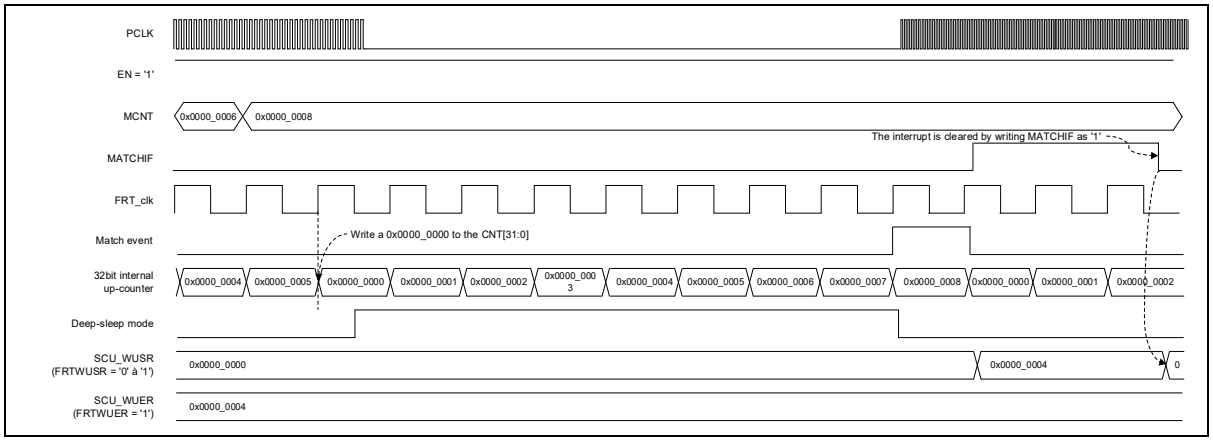
11.3.3.3 Register Setting Procedure by FRT Operation Mode

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 FRTCDIV[7:0] in SCU_MCCR6 register to a value larger than ‘0’. Then set the FRTCSEL[2:0] in SCU_MCCR6 register to a value of the desired clock.
4. Set the MODE in FRT_CTRL register to select a 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 FRT_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 FRT_MCNT register to specify the match counter value.

   Match interrupt time = FRT_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 FRT_CNT register.
9. To enable the FRT counter, set the EN in FRT_CTRL register to ‘1’.
10. Read the FRT_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 74. Timing Diagram in Low-Power Mode



11.3.4.1 Description of Steps of Low-Power Mode

1. To enable the signal to be used as a wake-up source, set FRTWUE in SCU_WUER register to '1'.
2. Set MCNT[31:0] in FRT_MCNT register, which triggers the match interrupt. (In the diagram above, it is set to 8 as an example.)
3. Clear CNT[31:0] in FRT_CNT register to 0, 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] in FRT_MCNT register, the Match Event is generated. Then MACHIF in FRT_STAT register and FRTWU in SCU_WUSR register are triggered.
6. And writing as '1' to MATHIF in FRT_STAT register to clear triggered flags. At this time FRTWU in SCU_WUSR register is also cleared.

11.4 FRT Registers

The base addresses and register map for the FRT are described in the tables below:

Table 83. Base Address of FRT

Name	Base Address
FRT	0x4000_0600

Table 84. FRT Register Map

Name	Offset	Type	Description	Reset Value	Reference
FRT_CTRL	0x0000	RW	FRT control register	0x0000_0000	11.4.1
FRT_MCNT	0x0004	RW	FRT match counter register	0xFFFF_FFFF	11.4.2
FRT_CNT	0x0008	RW	FRT counter register	0x0000_0000	11.4.3
FRT_STAT	0x000C	RWC1	FRT status register	0x0000_0000	11.4.4

11.4.1 FRT_CTRL: FRT Control Register

The FRT_CTRL register has control bits to the FRT operation. This register is 32 bits wide.

FRT_CTRL=0x4000_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
Reserved			OVFIE	MATCHIE	Reserved	MODE	EN
-			0	0	-	0	0
-			RW	RW	-	RW	RW

9	OVFIE	Whether to enable or disable the FRT counter overflow interrupt in Free-run timer mode (MODE = '0')	
		0	Disables the overflow interrupt.
		1	Enables the overflow interrupt.
8	MATCHIE	Whether to enable or disable the FRT counter match interrupt in Periodic timer mode (MODE = '1')	
		0	Disables the match interrupt.
		1	Enables the match interrupt.
1	MODE	FRT mode selection	
		0	Free-run timer mode
		1	Periodic timer mode
0	EN	Whether to enable or disable the FRT	
		0	Disables.
		1	Enables.

11.4.2 FRT_MCNT: FRT Match Counter Register

The FRT_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 FRT_CNT register counts up until it reaches the FRT_MCNT value, which triggers the match interrupt if it has been set enabled.

FRT_MCNT=0x4000_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
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 = FRT_clk period × (MCNT + 1)	

11.4.3 FRT_CNT: FRT Counter Register

The FRT_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 FRT_CTRL register is set to '1', this register can be read and written.

FRT_CNT=0x4000_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
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 FRT_STAT: FRT Status Register

The FRT_STAT is a 32-bit register that shows the FRT status. It can be read the status of each of FRT events and can be clear the status bit by writing '1' to the corresponding bit when the event occurs.

FRT_STAT=0x4000_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
Reserved			OVFIF	MATCHIF	Reserved
-			0	0	-
-			RW	RW	-

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 has occurred. (Writing a '1' to the bit clears this flag)

11.4.5 FRT Register Map Summary

Table 85. 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	FRT_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	FRT_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	FRT_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	FRT_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. UART (Universal Asynchronous Receiver/Transmitter)

12.1 UART Introduction

The A34L716 has one channel 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 65535) 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.

12.2 Main Features

The UART of A34L716 features the following:

- 16450 full-duplex asynchronous serial communication is supported.
- 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.

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 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 DMAn_CR register records the number of transfers. The counter decreases by 1 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.

Table 86 describes the summarized features of UART.

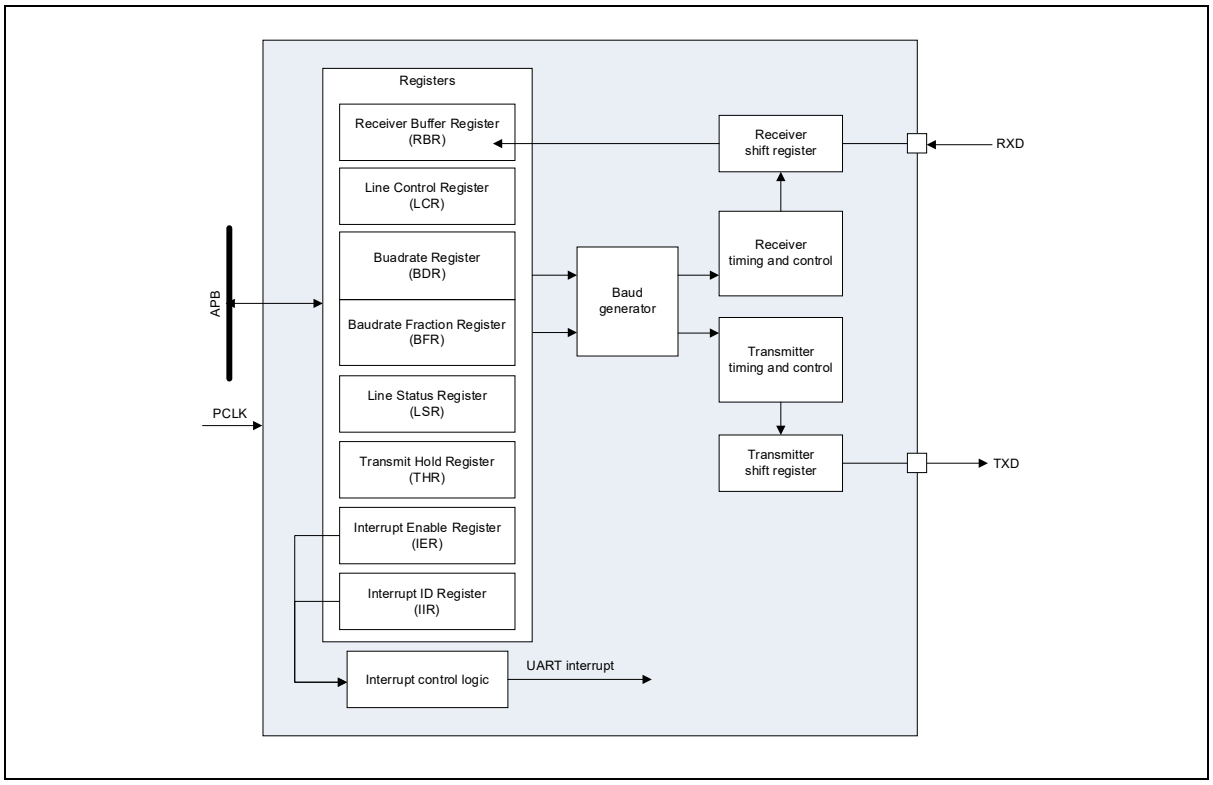
Table 86. Summarized Features of UART

UART Modes / Features	UART
Burst transfer using DMA	O
UART data length	5, 6, 7 and 8 bits

12.3.1 UART Block Diagram

Figure 75 shows a block diagram of the UART.

Figure 75. UART Block Diagram



12.3.2 UART Pins and Signals

Table 87 describes the external pins assigned for UART.

Table 87. Pin Assignment of UART: External Pins

Pin Name	Type	Description	Supported Packages		
			A34L716 (100-LQFP)	A34L716 (64-LQFP)	A34L716 (56-QFN)
TXD0	O	UART transmit output	O	O	O
RXD0	I	UART receive input	O	O	O

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).

RXD is a serial data input. Oversampling techniques are used for data recovery. They discriminate between valid incoming data and noise.

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 UART_LCR register as described below:

- For 5-bit character length: DLEN[1:0] = '00' (refer to Figure 80.)
- For 6-bit character length: DLEN[1:0] = '01' (refer to Figure 79.)
- For 7-bit character length: DLEN[1:0] = '10' (refer to Figure 78.)
- For 8-bit character length: DLEN[1:0] = '11' (refer to Figure 77.)

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 76 shows that the signal value (TXD and RXD) can be inverted, separately for each signal, through UART_DCR data inversion control.

Figure 76. Data Inversion Control Diagram

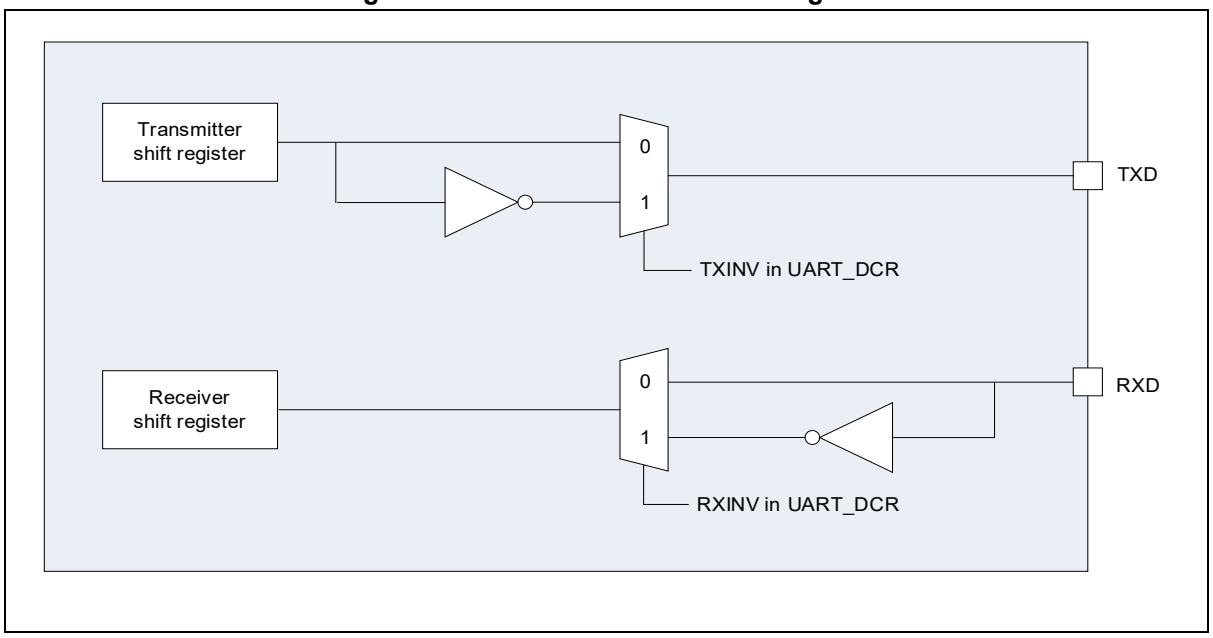


Figure 77 shows a timing diagram of data length programming. A Break condition is interpreted on receiving ‘0’s for longer than the time taken to receive the entire data word (i.e., the sum of the start, data, parity, and stop bits). The transmission and reception clocks are generated when the enable bit is set for the UART peripheral and clock, respectively.

Figure 77. Timing Diagram of 8-bit Data Length Programming

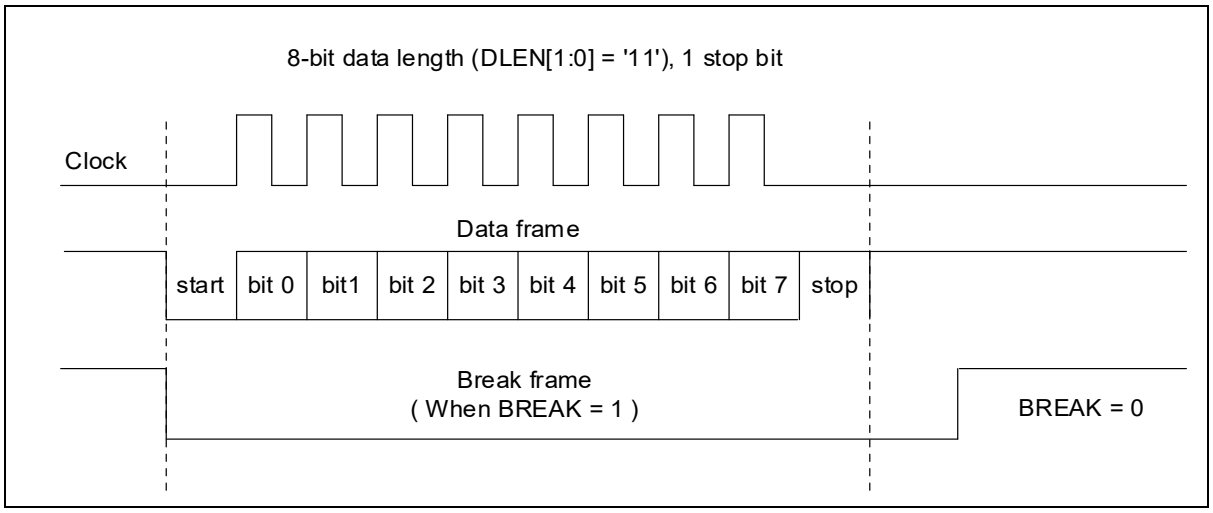


Figure 78. Timing Diagram of 7-bit Data Length Programming

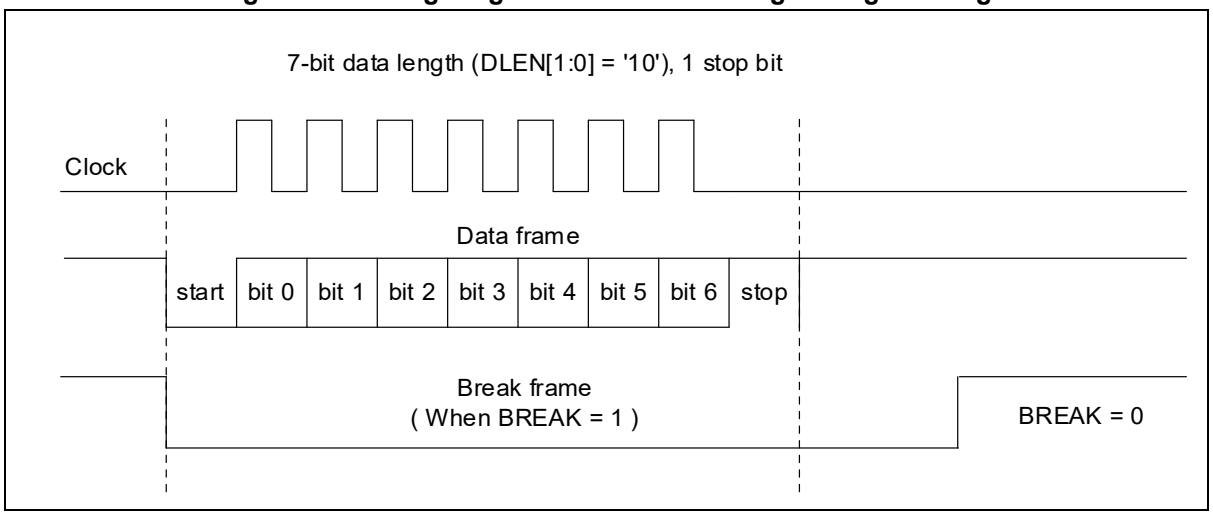


Figure 79. Timing Diagram of 6-bit Data Length Programming

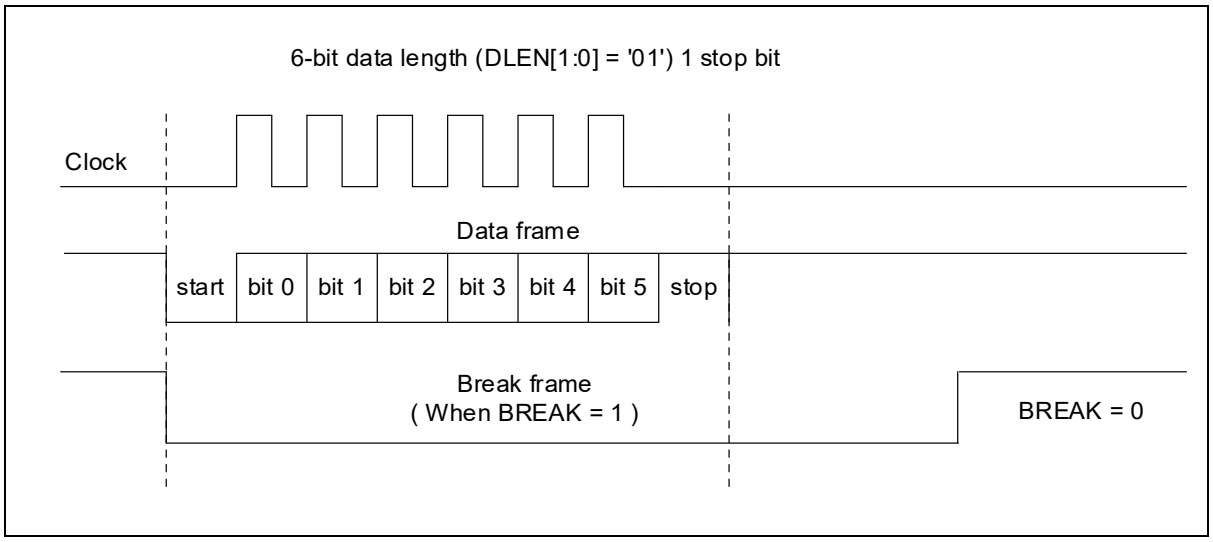
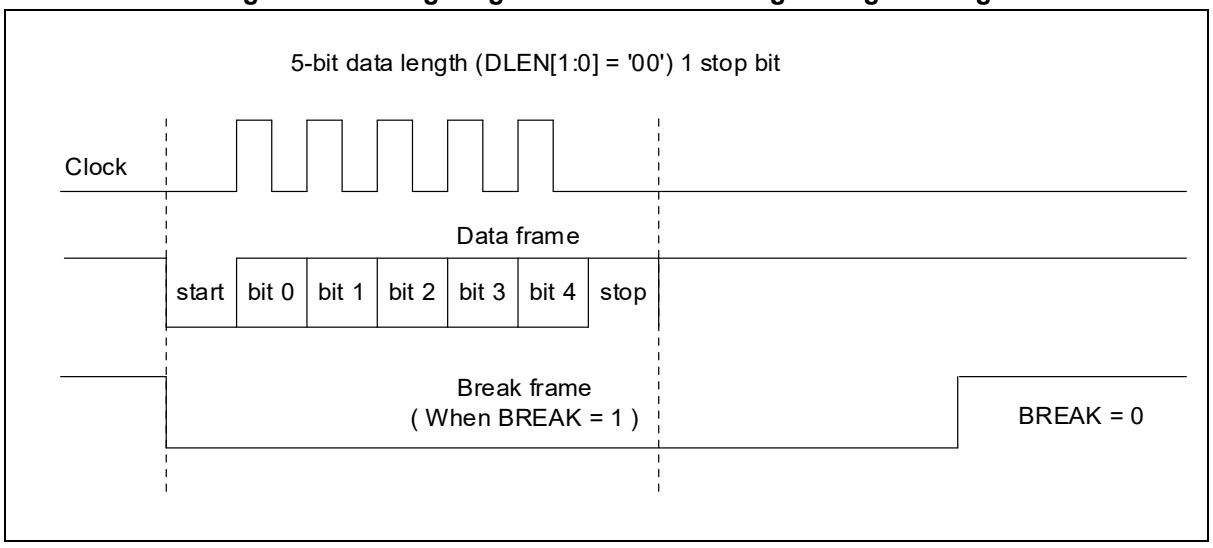


Figure 80. Timing Diagram of 5-bit Data Length Programming



12.3.4 UART Transmitter

12.3.4.1 Data Format

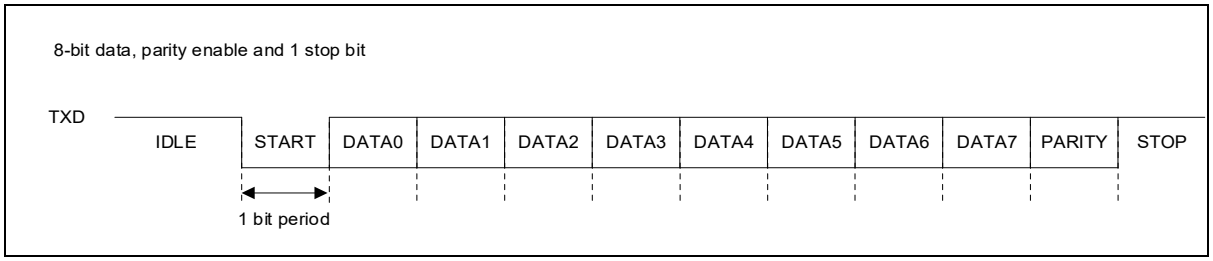
The UART transmitter is functioning to transmit data. For each data word, the start bit, data bits, optional bit, and stop bit are shifted serially in and out of the corresponding register, one bit at a time at the Least Significant Bit (LSB).

The number of data bits can be determined by setting the DLEN[1:0] in UART_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] in UART_LCR register.

The parity bit type can be determined by setting the PARITY and PEN in UART_LCR register. If odd or even parity is selected, the parity bits are determined so that the bit sum of all data bits is odd or even. The number of stop bits is defined by configuring the STOPBIT in UART_LCR register.

Figure 81 shows an example format of transmit data.

Figure 81. Example Format of Transmit Data



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.

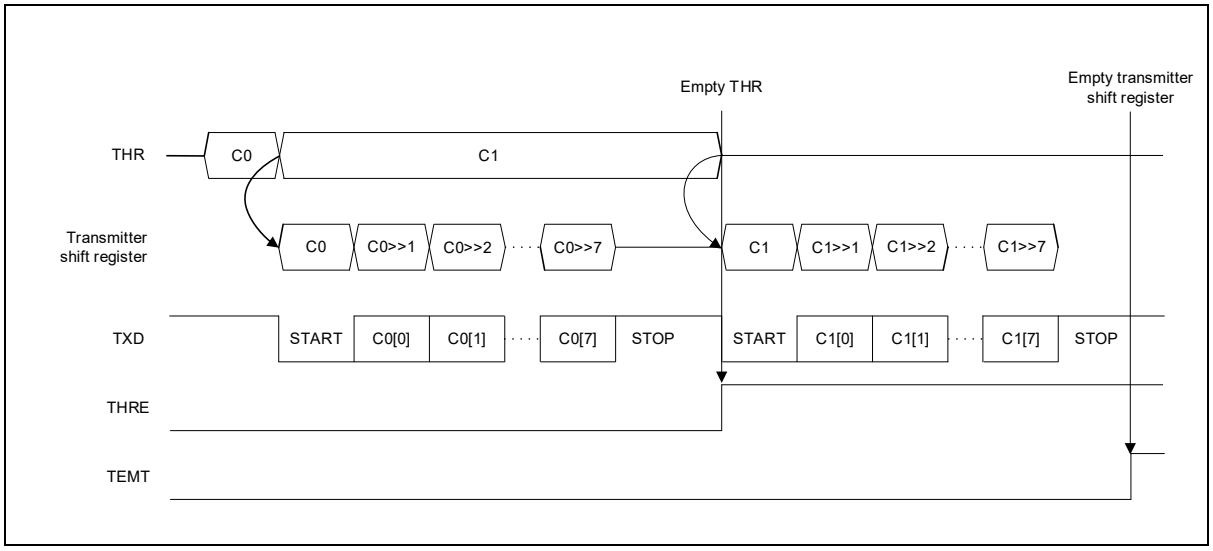
12.3.4.2 Transmit Interrupt

During UART transmission, some types of interrupt flags are used.

When the Transmit data Hold Resistor (UART_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 82.

Figure 82. 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 (UART_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.

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. These 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 is 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 in UART_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 UART_THR register starts UART communication.
2. Since data in the UART_THR register is transferred to the Transmit Shift Register for communication, updating the UART_THR register does not affect the data in communication.
    A. If the UART_THR register is updated, the Transmit Shift Register is automatically updated with the data in the UART_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 UART_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 UART_THR register is transferred to the Transmit Shift Register.
        ii. The DMA updates the memory data with the UART_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] 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 UART_THR register transfers data and the Transmit Shift Register outputs data. By checking the generation of the TXE interrupt, users can update the UART_THR register to restart the communication.
        ii. If the IID[2:0] in UART_IIR register is set to ‘001’ without the generation of the TXE interrupt, it means that data in the UART_THR register has been transferred to the Transmit Shift Register.
        iii. If the UART_THR register is not updated with new data, the TXE interrupt is generated when the Transmit Shift Register completes outputting its current data.
4. Polling the TEMT and THRE in UART_LSR register,
    A. The TEMT bit implies that the Transmit Shift Register is empty.
    B. The THRE bit implies that the UART_THR register is empty. (The UART_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 UART_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] in UART_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 UART_THR register is updated, the Transmit Shift Register that output data will be updated with new data to start the 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) in UART_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 BREAK 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.

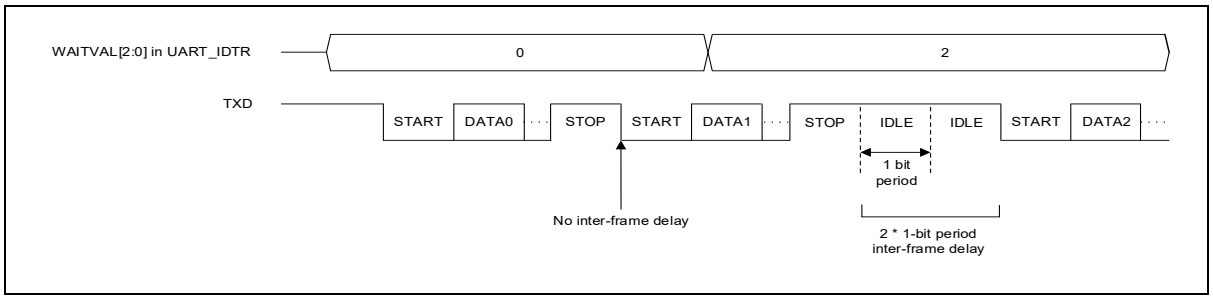
12.3.4.8 Inter-Frame Delay for Data Transmission

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] in UART_IDTR register.

If the WAITVAL[2:0] is 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 83 shows an example format of the transmit data.

Figure 83. Example Format of Transmit Data



12.3.5 UART Receiver

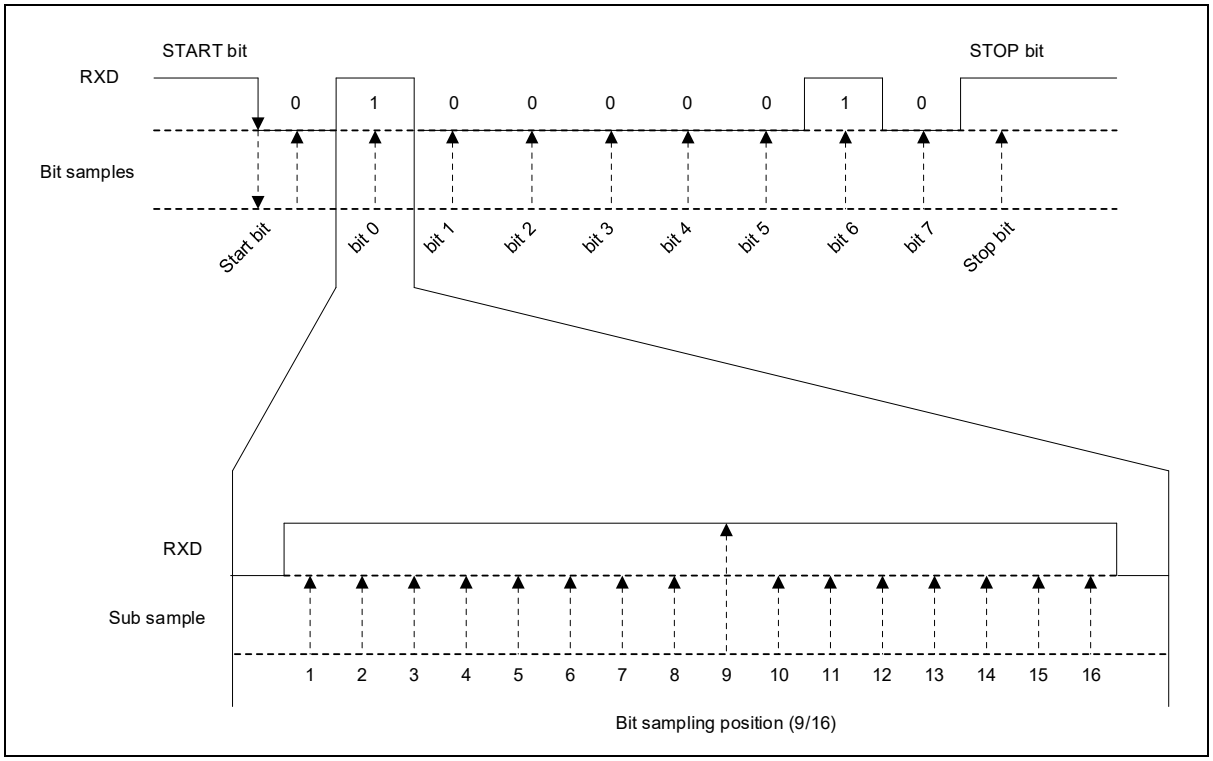
12.3.5.1 Receive Data Sampling Timing

The 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 9th 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] in UART_LCR register.

Figure 84. 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

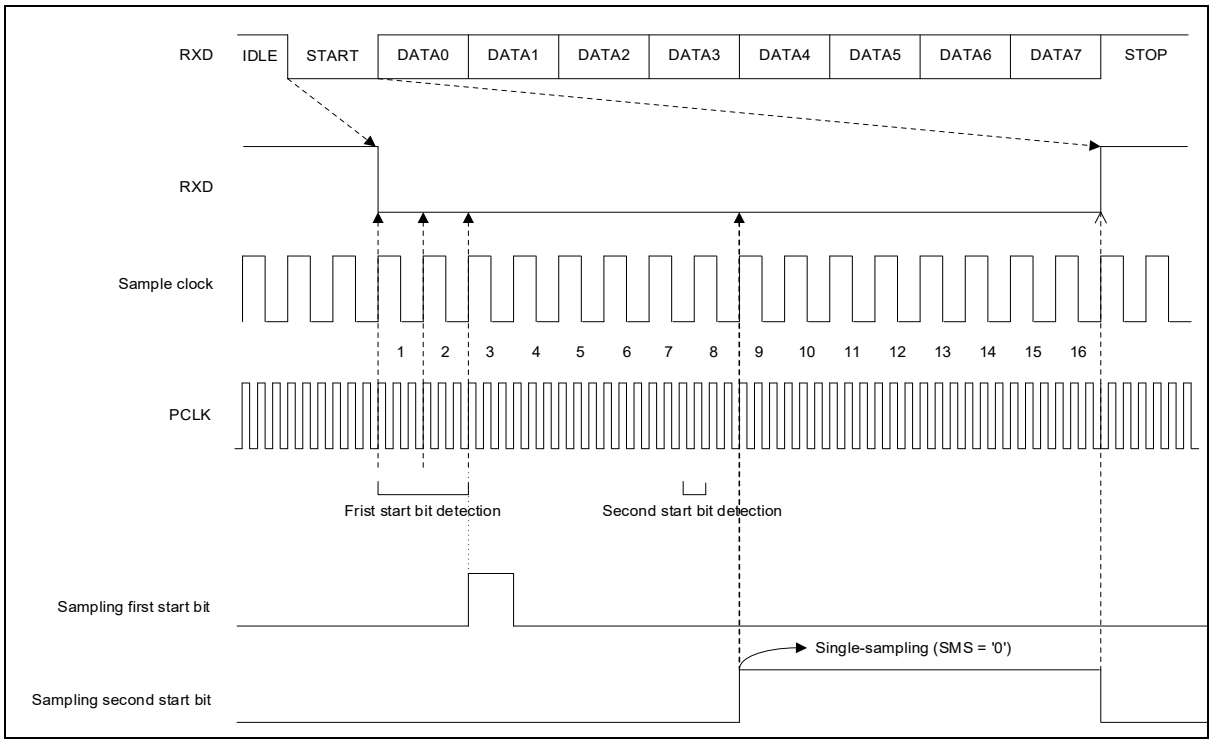
On the falling edge of the start bit, a digital filter rejects noise signals. A pulse shorter than ((3 × (1 / baud rate)) / 16) second is rejected by this filter. Even when a pulse is longer than ((3 × (1 / baud rate)) / 16) s, it can be rejected under the multi-sampling strategy during start-bit sampling.

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

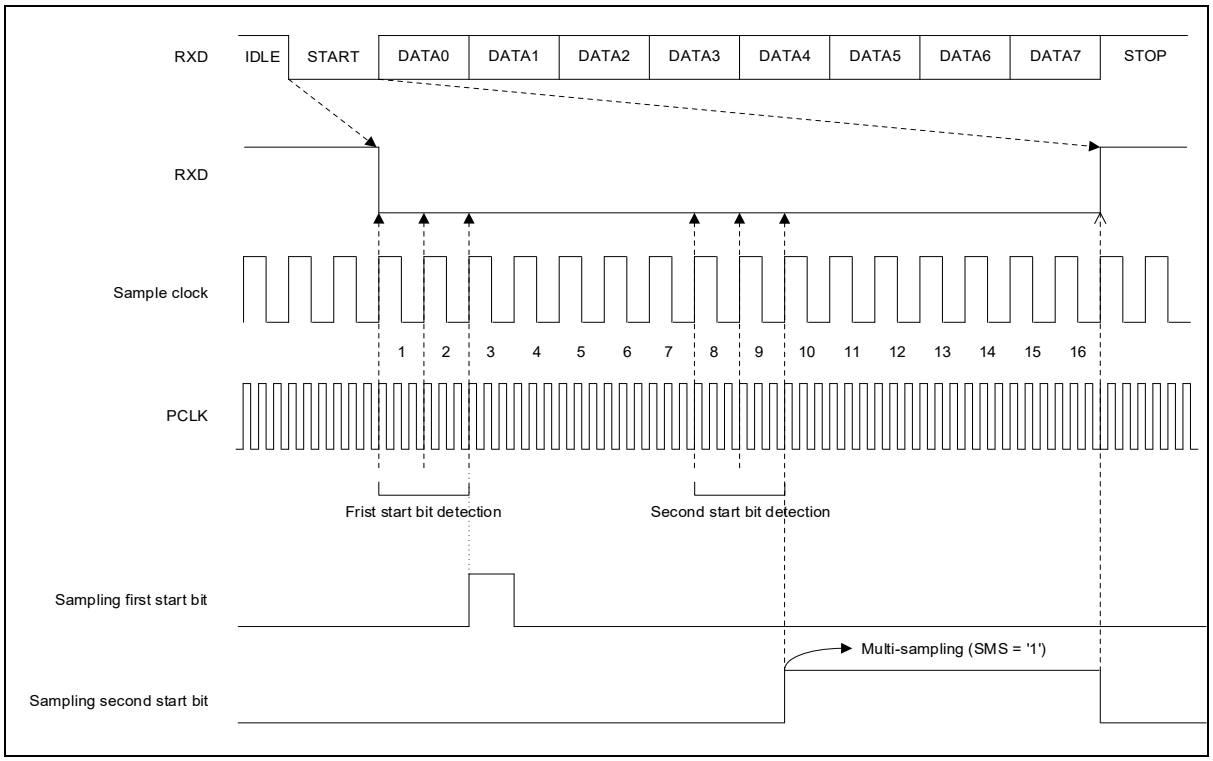
The start bit is confirmed (The DR in UART_LSR register set and an interrupt generated if the DRIE in UART_IER register is set to '1'), if the sampled bits are at '0' in single-sampling mode. The first sampling on the 1st, 2nd, and 3rd bits finds the three bits at '0' and the second sampling on the 9th bit also finds the bit at '0'.

Figure 85. Start Bit Detection of Single-Sampling Timing



Similarly, the start bit is confirmed (The DR in UART_LSR register set and an interrupt generated if the DRIE in UART_IER register is set to ‘1’), if the 3 sampled bits are at ‘0’ in multi-sampling mode. First sampling on the 1st, 2nd, and 3rd bits finds the 3 bits at ‘0’ and the second sampling on the 8th, 9th, and 10th bits finds the two bits at ‘0’.

Figure 86. 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 UART_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 87 shows the single sampling of data bits when the DMS is set to ‘0’.

Figure 87. Single-Sampling Timing of Data Bit

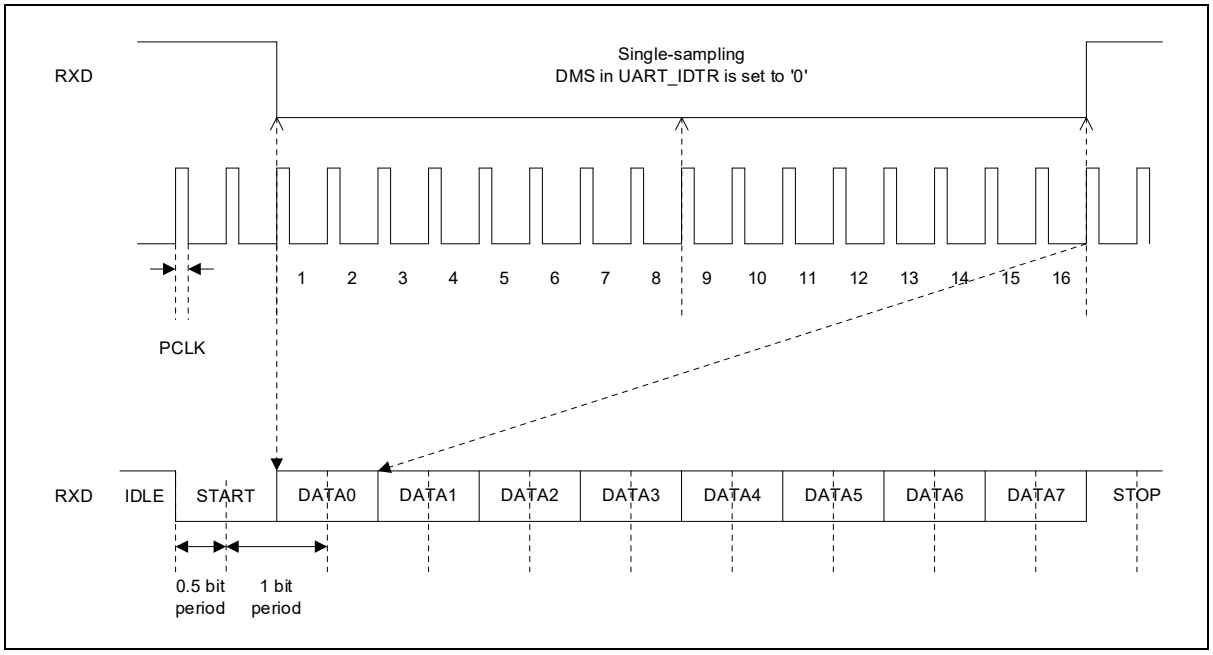
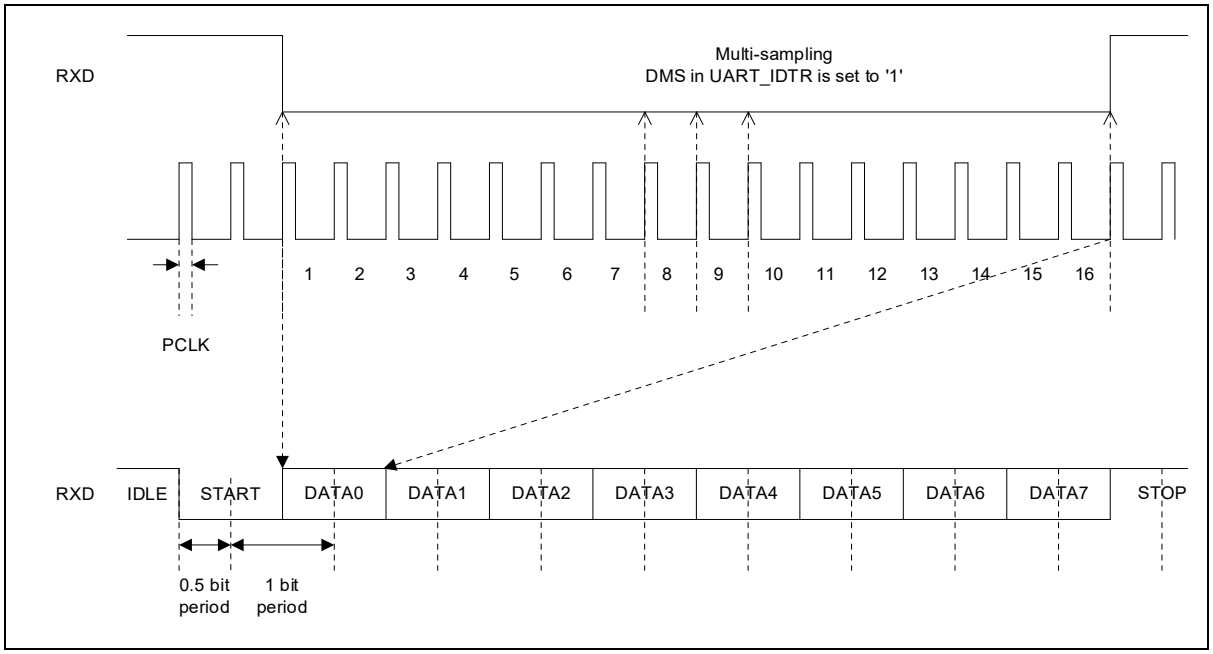


Figure 88 shows the multi-sampling of data bits when the DMS is set to ‘1’.

Figure 88. 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 noises in the RXD signal line. The start bit and data bits can be either single-sampled or multi-sampled. The SMS in UART_IDTR register defines the sampling strategy for the start bit, and the DMS in UART_IDTR register defines the sampling strategy for the data bits.

Figure 89. Data Sampling Timing

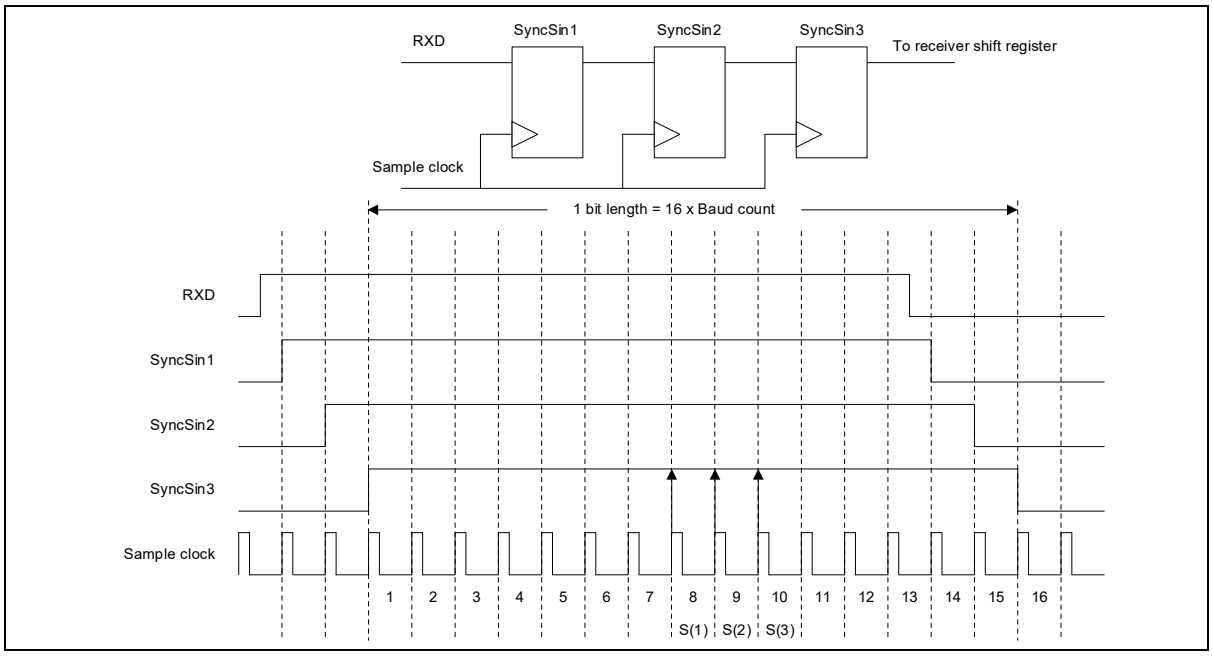


Figure 89 shows a block diagram and a sampling timing diagram of the anti-noise synchronous circuit.

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 to the Start bit.

Table 88. Received Bit Value from Sampled Data

Sampled Value	Received Bit Value
000	0
001	0
010	0
011	1
100	0
101	1
110	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 UART_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 UART_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 UART_RBR is updated. The DMA stores data of the UART_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] in UART_IIR register must be checked. If the IID[2:0] is set to ‘010’, this means that the Reception operation is completed and the UART_RBR register can be read.
- To confirm the Reception is completed using Polling, the DR in UART_LSR register must be checked. If data is received successfully, the DR bit is set to ‘1’ and data in the UART_RBR can be read. Reading the UART_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 in UART_LSR register indicates whether the Overrun Error has been detected. The Overrun Error occurs if the Receiver completes the new Reception operation when the UART_RBR data was not read (The DR in UART_LSR register is '1'). At this moment, although the Overrun Error occurs, data in the UART_RBR register is updated.

Reading the UART_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 in UART_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] in UART_LCR register. If the data size set in the DLEN[1:0] bits and the received data's bit size are different, problems may occur.

Reading the UART_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 in UART_LSR register is set to '1'.

Reading the UART_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 in UART_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 65535.

Users must write an appropriate divisor value to the 16-bit UART_BDR register to obtain the desired baud rate.

Below is the formula for calculating the baud rate:

$$\text{UARTn\_BDR} = \frac{(\text{UART clock set by SCU\_MCCR7})}{16 \times \ BaudRate}$$

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 89 lists the divisor and error rate for each baud rate, at the UART clock speed of 16 MHz.

Table 89. Examples of Baud Rate Calculation

UART Clock = 16 MHz		
Baud Rate	**Divisor (UART_BDR)**	**Error (%)**
1200	833	0.04%
2400	416	0.16%
4800	208	0.16%
9600	104	0.16%
19200	52	0.16%
38400	26	0.16%
57600	17	2.08%
115200	8	7.84%

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 UART_BFR register to the integer part indicated in the UART_BDR register, the correct divisor is obtained to calculate the accurate baud rate.

NOTE:

1. If the UART_BDR register value is equal to or less than 2, the UART_BFR register value is always 0 regardless of the UART_BFR register's set value.

For example, when the baud rate is 9600 bps:

$$\frac{UART\ clock}{16 \times\ BaudRate} = \frac{16000000}{16 \times\ 9600} = 104.1667, \qquad Divisor = 104, Float = 0.1667$$

$$CNT = Float \times 256\ = 0.1667 \times 256 = 42$$

$$UARTn\_BDR\ =\ 104, UARTn\_BFR\ =\ 42$$

Table 90. Examples Calculation of Baud Rate Using UART Fraction Counter

UART Clock = 16 MHz			
Baud Rate	**Divisor (UART_BDR)**	**FCNT (UART_BFR)**	**Error (%)**
1200	833	85	0.0%
2400	416	170	0.0%
4800	208	85	0.0%
9600	104	42	0.0%
19200	52	21	0.0%
38400	26	10	0.0%
57600	17	92	0.0%
115200	8	174	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 9th 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 91 shows an acceptable error that depends on the bit size of transmitted data.

Table 91. 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 UART_LCR register. Depending on the frame length defined in the DLEN[1:0] in UART_LCR register, the possible UART frame formats are as listed in Table 92.

Table 92. UART Frame Formats

DLEN[1:0] Bits	PEN Bits	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 in UART_LCR register = ‘1’ and PEN in UART_LCR register = ‘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 in UART_LCR register = ‘0’ and PEN in UART_LCR register = ‘1’).

12.3.8.3 Stick Parity

The STICKP in UART_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 in UART_LCR register.

Table 93 describes various configurations to create a parity bit.

Table 93. Various Configurations to Create a Parity Bit

Name	Offset	Type	Description
STICKP	PARITY	PEN	Parity
X	X	0	No parity
0	0	1	Odd-parity
0	1	1	Even-parity
1	0	1	Forced 1 stick-parity
1	1	1	Forced 0 stick-nrparity

12.3.8.5 Parity Checking in Reception

If the parity check fails, the PE flag is set by configuring the PE in UART_LSR register, and the corresponding interrupt is generated if the RLSIE in UART_IER register is set to '1'.

Reading the UART_LSR register clears the PE bit to '0'.

12.3.8.6 Parity Generation in Transmission

If the PEN in UART_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 UART_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 UART_IIR register.

Similarly, once the entire receive data is written to the target DMA memory, the UART_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] in UART_IIR register. Therefore, the UART RXD signal already becomes idle when the DMA Rx complete interrupt occurs.

12.3.9.1 Transmission Using DMA

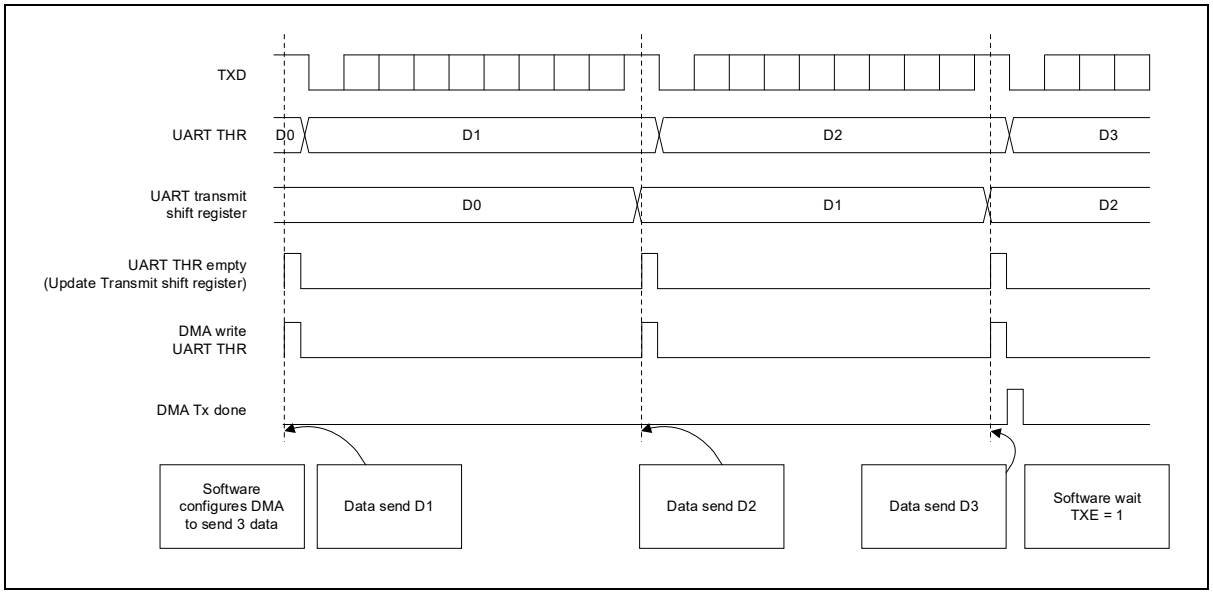
With DMA enabled, when the Transmit shift register is updated, the DMA Tx request signal is generated to allow the UART_THR register to store new data.

When the UART_THR register stores the new data using DMA transfer and the Transmit Shift Register completes to output data, the UART_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 90 is 3), the DMA generates the DMA Tx done signal. Upon receiving this signal, the 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 UART_THR register, the last data in the Transmit Shift Register is output and the Transmitter operation is complete. (TXE interrupt is generated.)

Figure 90. Transmission Using DMA



12.3.9.2 Reception Using DMA

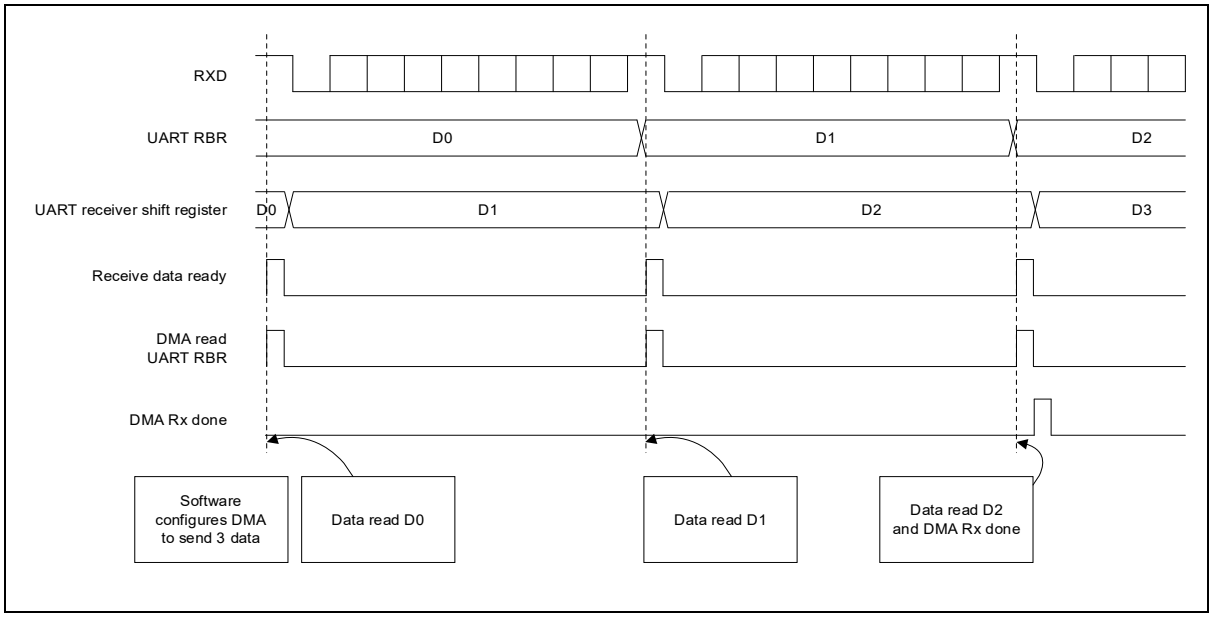
When the Receiver Shift Register is updated according to the reception operation, new data is stored in the UART_RBR register. With DMA enabled, when the UART_RBR register is updated with new data, the DMA Rx Request signal is generated.

DMA reads the UART_RBR register to store the data in memory.

After the DMA stores the data of the UART_RBR register into memory as many times as set in the DMA Rx count (The set number in Figure 91 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 91. Reception Using DMA



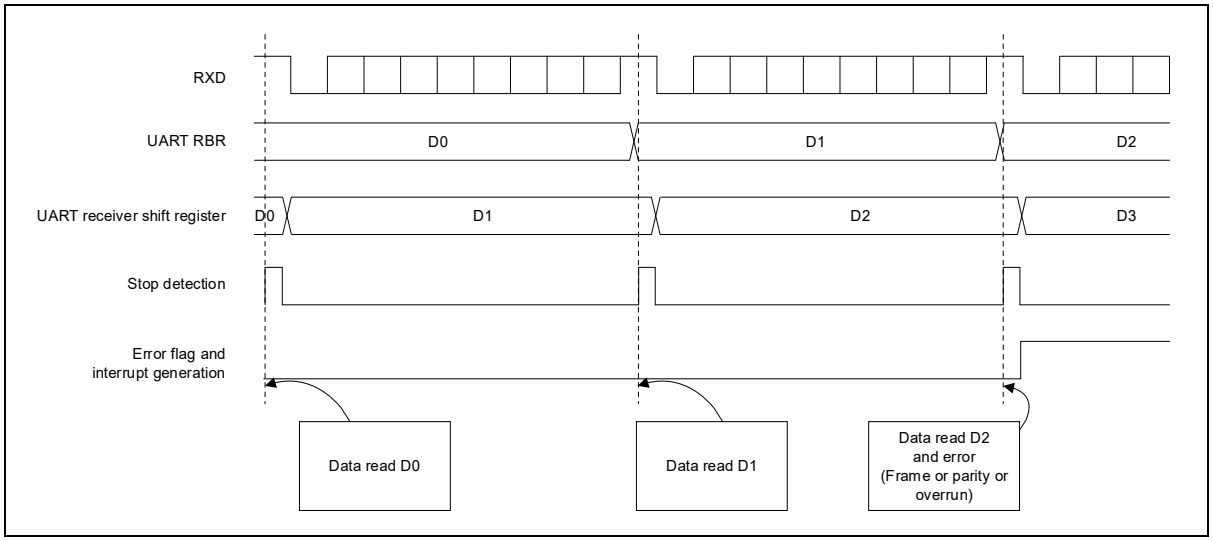
12.3.9.3 Error Flagging and Interrupt Generation in Communication

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 in UART_IER register.

At this point of time, the IID in UART_IIR register represents the interrupt source ID, and the UART_LSR register shows the type of error. The UART_LSR register is updated whenever an error occurs, even if the RLSIE in UART_IER register is disabled.

Although an error occurs, data in the Receiver Shift Register is transmitted to UART_RBR register and then UART Rx operation is maintained normally.

Figure 92. 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 UART_IIR register.

When the microcontroller’s core accesses the UART_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’s 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] in UART_IIR register.

The UART module uses a total of seven interrupts with different priorities. These interrupts include the following:

- Receive line status interrupt
- Receive data ready interrupt
- Transmit data hold register empty interrupt
- Transmit complete interrupt
- DMA Rx complete interrupts
- DMA Tx complete interrupts
- Transmit complete and DMA Tx complete interrupts

12.5 UART Registers

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

Table 94. Base Address of UART

Name	Base Address
UART	0x4000_8000

Table 95. UART Register Map

Name	Offset	Type	Description	Reset Value	Reference
UART_RBR	0x0000	RO	UART receive data buffer register	0x0000_0000	12.5.1
UART_THR	0x0000	WO	UART transmit data hold register	0x0000_0000	12.5.2
UART_IER	0x0004	RW	UART interrupt enable register	0x0000_0000	12.5.3
UART_IIR	0x0008	RO	UART interrupt ID register	0x0000_0001	12.5.4
UART_LCR	0x000C	RW	UART line control register	0x0000_0000	12.5.5
UART_DCR	0x0010	RW	UART data control register	0x0000_0000	12.5.6
UART_LSR	0x0014	RO	UART line status register	0x0000_0060	12.5.7
UART_BDR	0x0020	RW	UART baud rate divisor latch register	0x0000_0000	12.5.8
UART_BFR	0x0024	RW	UART baud rate fraction counter register	0x0000_0000	12.5.9
UART_IDTR	0x0030	RW	UART inter-frame delay time register	0x0000_0000	12.5.10

12.5.1 UARTn_RBR: UART n Receive Data Buffer Register

UART_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.

UART_RBR=0x4000_8000

31 30 29 28 27 26 25 24	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]
-			00
-			RO

7 0	RBR[7:0]	Receive buffer register

12.5.2 UARTn_THR: UART n Transmit Data Hold Register

UART_THR is an 8-bit write-only register. Data is stored in this register to be transmitted. However, data written to UART_THR cannot be read but sent to the Transmit Shift Register when this register is empty.

UART_THR=0x4000_8000

31 30 29 28 27 26 25 24	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

UART_IER enables six types of UART interrupts. Each interrupt generates an interrupt output signal. If users set all bits in the UART_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'.

UART_IER=0x4000_8004

31 30 29 28 27 26 25 24	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				DTXIEN	DRXIEN	TXEIE	RLSIE	THREIE	DRIE
-				0	0	0	0	0	0
-				RW	RW	RW	RW	RW	RW

5	DTXIEN	Whether to enable or disable the DMA transmit complete interrupt	
		0	Disables the DMA transmit complete interrupt.
		1	Enables the DMA transmit complete interrupt.
4	DRXIEN	Whether to enable or disable the DMA receive complete interrupt	
		0	Disables the DMA receive complete interrupt.
		1	Enables the DMA receive complete interrupt.
3	TXEIE	Whether to enable or disable the transmit complete interrupt	
		0	Disables the transmit complete interrupt.
		1	Enables the transmit complete interrupt.
2	RLSIE	Whether to enable or disable the receive line status interrupt	
		0	Disables the receive line status interrupt.
		1	Enables the receive line status interrupt.
1	THREIE	Whether to enable or disable the transmit data hold register empty (THRE) interrupt	
		0	Disables the THRE interrupt.
		1	Enables the THRE interrupt.
0	DRIE	Whether to 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

The UART_IIR register is a read-only register informing the operation of the interrupt that is set to enabled in the UART_IER register. When the CPU accesses the UART_IIR register, the UART locks all interrupts, and the highest-priority interrupt is read. Interrupts occur even while the register is being accessed by the CPU; however, its status remains unchanged until the current access is completed.

UART_IIR=0x4000_8008

31 30 29 28 27 26 25 24	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				TXE	IID[2:0]	IPEN
-				0	000	1
-				R0	R0	R0

4	TXE	Transmit complete (Refer to Table 96.)
		TXE Indicates whether the transmit shift register is empty. TXE = 0 indicates that UART_THR and transmit shift register is currently empty and thus new data can be written to the UART_THR register.
3 1	IID[2:0]	Interrupt source ID (Refer to Table 96.)
		IID[2:0] Displays the interrupt with the highest priority among those currently active.
0	IPEN	Presence of pending interrupts IPEN Indicates whether 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

Table 96. 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 UART_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 UART_IIR
4	1	X	X	X	X	Transmit register empty	Transmit shift register and transmit data hold register is empty	Write to the transmit data hold register or read UART_IIR
5	0	1	1	0	0	DMA Rx complete	DMA Rx complete	Read UART_IIR
6	0	1	0	1	0	DMA Tx complete	DMA Tx complete	Read UART_IIR
7	1	X	X	X	X	Transmit register empty and DMA complete	The transmit register is empty and DMA Tx has been completed.	Read UART_IIR

12.5.5 UARTn_LCR: UART n Line Control Register

The UART_LCR register is an 8-bit register.

UART_LCR=0x4000_800C

31 30 29 28 27 26 25 24	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	Whether or not to use stick parity. The setting of this bit is valid only when the PEN bit is set to 1. STICKP determines whether 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	Whether or not to enable parity PEN determines whether 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 one. 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	DLEN[1:0]	The length of data bits included by each data transfer

0	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 UART_DCR register is a 32-bit register that controls Tx or Rx data inversion.

UART0_DCR=0x4000_8010

31 30 29 28 27 26 25 24	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	Rx data inversion selection	
		0	Inputs normal Rx data.
		1	Inputs inverted Rx data.
2	TXINV	Tx data inversion selection	
		0	Outputs normal Tx data.
		1	Outputs inverted Tx data.

12.5.7 UARTn_LSR: UART n Line Status Register

The UART_LSR register is a read-only register that shows the status of data transmission and reception.

UART_LSR=0x4000_8014

31 30 29 28 27 26 25 24	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				TEMT	THRE	BI	FE	PE	OE	DR
-				1	1	0	0	0	0	0
-				RO	RO	RO	RO	RO	RO	RO

6	TEMT	Whether or not the transmit registers are empty TEMT indicates whether the transmit registers are empty. When both UART_THR and transmit shift register are empty, the bit is set to '1'. Once either UART_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	Whether or not the transmit data hold register is empty THRE indicates whether UART_THR is empty, which means the UART is ready to receive new data from the microcontroller's core for transmission. UART_THR empty interrupts can be generated if THREIE is set enabled in UART_IER. The THRE bit is set to '1' when new data can be written to UART_THR as the previous transmit data has been transferred from UART_THR to the transmit shift register. The bit is cleared to '0' when new data is written to UART_THR by the microcontroller's core (when UART_THR is no longer empty).
	0	The transmit data hold register is not empty.
	1	The transmit data hold register is empty.
4	BI	Whether or not a break condition has been detected BI shows the occurrence of 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 UART_LSR is read by the microcontroller's core, the bit is cleared to '0'.
	0	Normal.
	1	A break condition has been detected.
3	FE	Presence of a frame error FE Indicates whether 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 UART_LSR is read by the microcontroller's 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	Presence of a parity error PE indicates whether a parity error has been detected. If a received data word does not meet the odd / even parity condition defined in UART_LCR, the bit is set to '1'. Once UART_LSR is read by the microcontroller's 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	Presence of an overrun error OE indicates whether an overrun error has been detected. When a data word is transferred to UART_RBR before the register reads the previous data word, an error occurs, and the OE bit is set to '1'. Once UART_LSR is read by the microcontroller's core, the overrun error is not detected, and the bit is cleared to '0'.	
		0	No overrun error present.
		1	An overrun error has been detected. Additional data has arrived when the UART_RBR is already full.
0	DR	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 UART_RBR. Once the data is read by UART_RBR, the bit is cleared to '0'.	
		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_BDR: UART n Baud Rate Divisor Latch Register

The UART_BDR register is a 16-bit register.

UART_BDR=0x4000_8020

31 30 29 28 27 26 25 24	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)

12.5.9 UARTn_BFR: UART n Baud Rate Fraction Counter Register

The UART_BFR register is an 8-bit register.

UART_BFR=0x4000_8024

31 30 29 28 27 26 25 24	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)	Whether to enable or disable the fraction counter to compensate for error from the integer baud rate divisor	
		0	Disables the fraction counter.
		N	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 UART_BDR value is equal to or less than two, the UART_BFR is always zero regardless of the UART_BFR set value.

12.5.10 UARTn_IDTR: UART n Inter-Frame Delay Time Register

The UART_IDTR register is an 8-bit register. A dummy delay is inserted between two consecutive transmit data frames.

UART_IDTR=0x4000_8030

31 30 29 28 27 26 25 24	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]	Whether to enable or disable 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 9 (Among the 16 pulses in total).
	1	Enables multi-sampling the start 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.
6	DMS	Whether to enable or disable 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 of 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 powering on the microcontroller, if the UART bit in the SCU_PER2 register is disabled and then re-enabled, the SMS bit is set to '1'.

12.5.11 UART Register Map Summary

Table 97. 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.	RBR[7:0]								
	Reset value																									0	0	0	0	0	0	0	0
0x00	UARTn_THR	Res.	THR[7:0]								
	Reset value																									0	0	0	0	0	0	0	0
0x04	UARTn_IER	Res.	Res.	DTXIEN	DRXIEN	TXEIE	RLSIE	THREIE	DRIE
	Reset value																											0	0	0	0	0	0
0x08	UARTn_IIR	Res.	Res.	Res.	TXE	IID[2:0]		IPEN
	Reset value																												0	0	0	0	1
0x0C	UARTn_LCR	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.	RXINV	TXINV	Res.	Res.
	Reset value																													0	0		
0x14	UARTn_LSR	Res.	TEMT	THRE	BI	FE	PE	OE	DR
	Reset value																										1	1	0	0	0	0	0
0x20	UARTn_BDR	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.	BFR[7:0]								
	Reset value																									0	0	0	0	0	0	0	0
0x30	UARTn_IDTR	Res.	SMS	DMS	Res.	Res.	Res.	WAITVAL[2:0]		
	Reset value																									0	0				0	0	0

13. SPI (Serial Peripheral Interface)

13.1 SPI Introduction

The A34L716 has two built-in Serial Peripheral Interface (SPI) modules. The SPI module is 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.

13.2 SPI Main Features

The SPI of the A34L716 features the following:

- Selectable between the master and slave operations
- Full-duplex and four-wire synchronous transfers supported.
  - 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 bits
- Configurable interrupts triggered by the transmit status and SS signal
- Loop-back mode for internal checkups
- User-programmable start, burst, and stop delay times
- Port Drive Strength for High-Speed Communication
- Selectable between MSB first transfer or LSB first transfer
- DMA Tx, Rx transfers

13.3 SPI Functional Description

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 pin MOSI, the slave responds through the pin MISO.

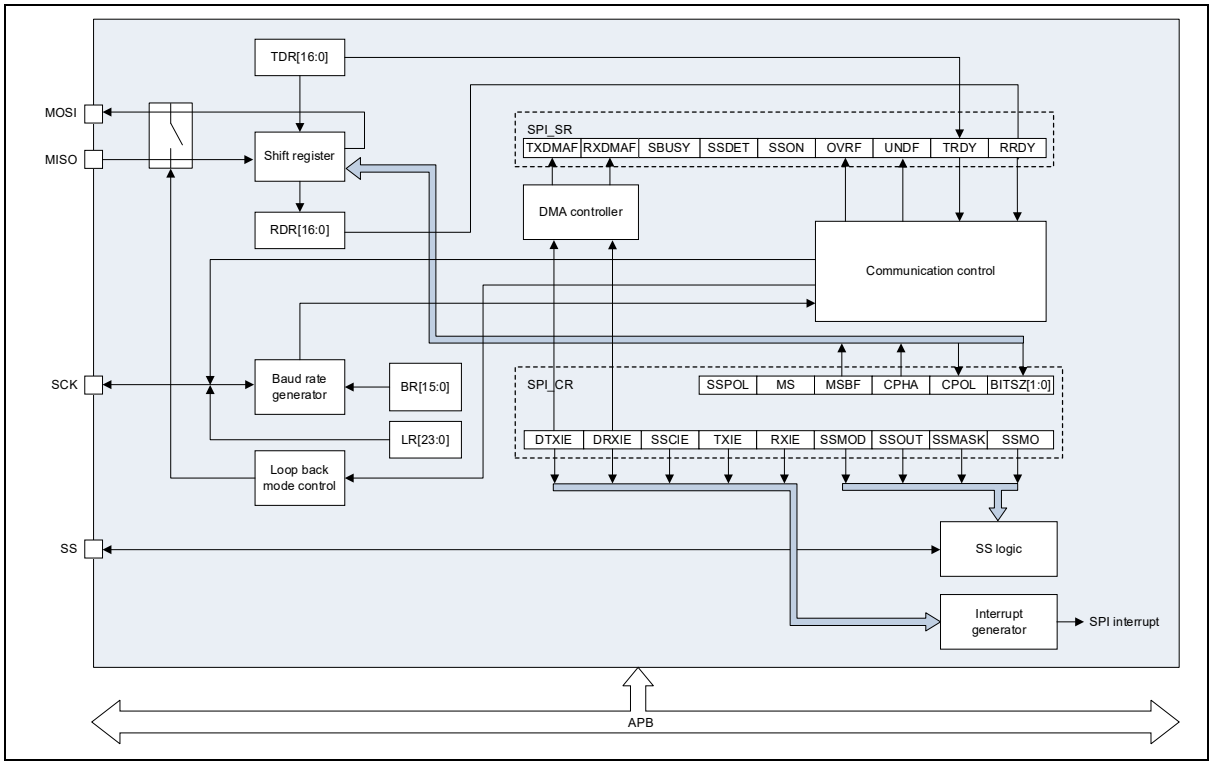
Table 98 describes the summarized features of the SPI.

Table 98. Summarized Features of SPI

SPI Feature	Description
DMA	DMA transmission and reception
Data size	Data size selection for transmission and reception (8-bit, 9-bit, 16-bit, 17-bit)
MSB / LSB selection	Transfer format selection (MSB or LSB)
Error flag	Error flags (Overrun, Underrun)
Latency	Programmable latency (Start, Burst, Stop)

13.3.1 SPI Block Diagram

Figure 93. SPI Block Diagram



13.3.2 SPI Pins and Internal Signals

Table 99 describes the pins assigned for the SPI.

Table 99. Pin Assignment of SPI: External Pins (n = 0 and 1)

Pin Name	Type	Description	Supported Packages		
			A34L716 (100-LQFP)	A34L716 (64-LQFP)	A34L716 (56-QFN)
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	O
SCK1	I/O	SPI1 clock I/O (Master: output / slave: input)	O	O	O
MOSI1	I/O	SPI1 transmit and receive data (Master: output / slave: input)	O	O	O
MISO1	I/O	SPI1 transmit and receive data (Master: input / slave: output)	O	O	O

13.3.3 Communication between Master and Single Slave

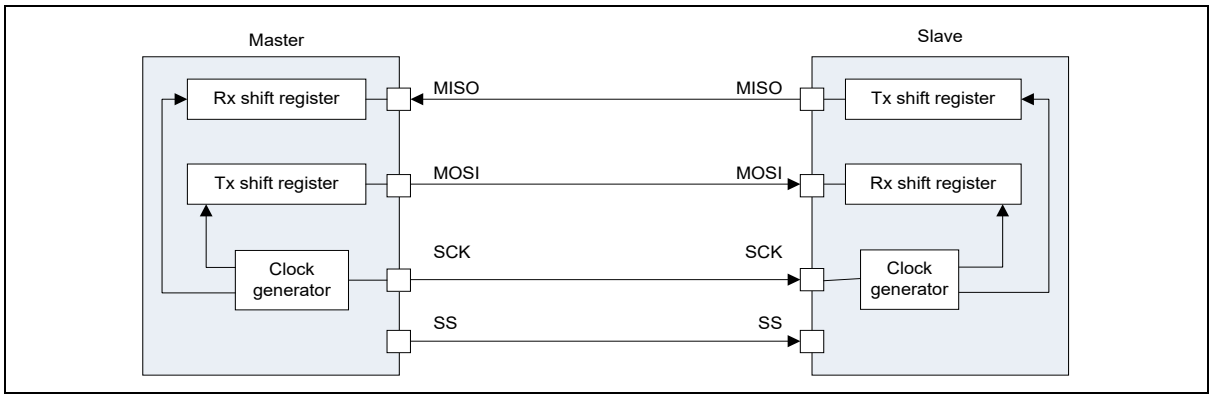
13.3.3.1 Full-Duplex Communication

Figure 94 shows a normal connection for the communication between the SPI-Master and a Slave. For the full-duplex communication, the SPI- Master and slave are connected as shown in Figure 94.

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 94. Block Diagram of SPI-Master to Slave Connections: Full-Duplex



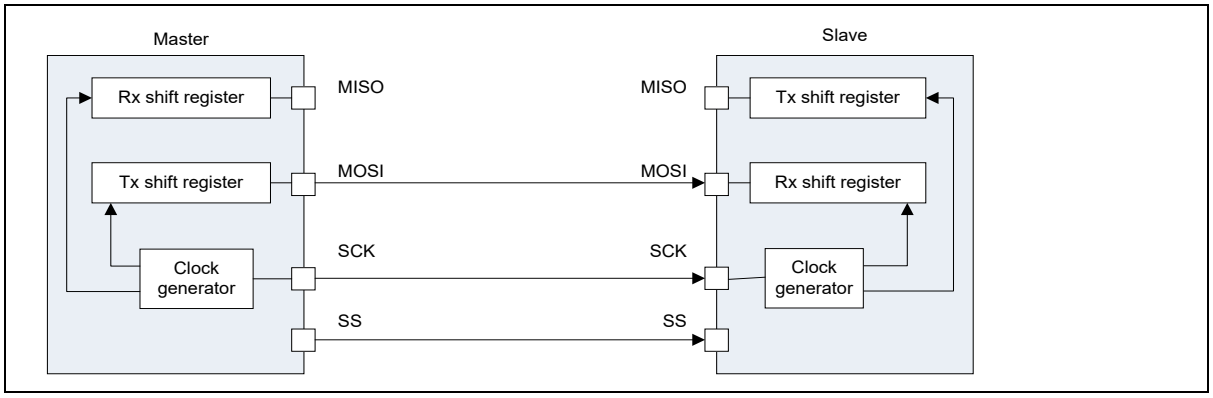
13.3.3.2 Simplex Communication

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 95.

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 95. Block Diagram of SPI-Master to Slave Connections: Simplex



13.3.4 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 SPIn_CR register.

If the SSMOD is set to ‘1’, the pin is driven internally according to the settings of the SSOUT in SPIn_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 SPIn_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.

13.3.5 Communication Formats

13.3.5.1 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 in 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 96 and Figure 97 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 96. SPI Transfer Timing 1 of 4 (CPHA = ‘0’, CPOL = ‘0’, MSBF = ‘0’)

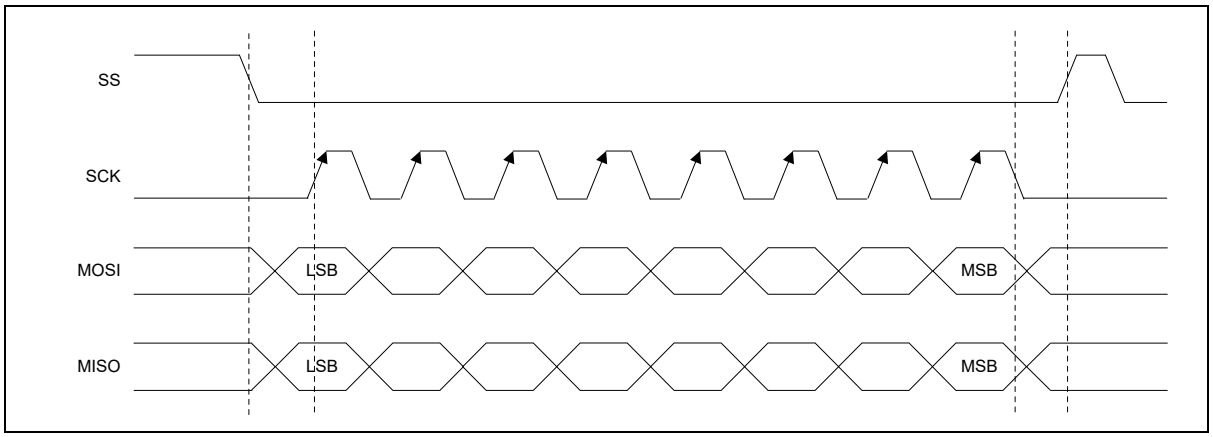


Figure 97. SPI Transfer Timing 2 of 4 (CPHA = ‘0’, CPOL = ‘1’, MSBF = ‘1’)

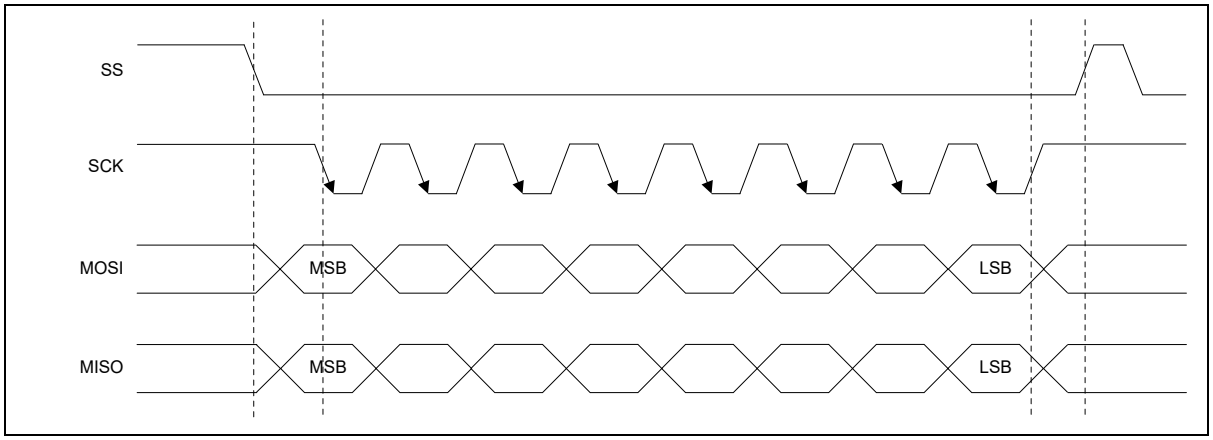


Figure 98 and Figure 99 illustrate SPI transfer timings when the CPHA is set to ‘1’.

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 98. SPI Transfer Timing 3 of 4 (CPHA = ‘1’, CPOL = ‘0’, MSBF = ‘0’)

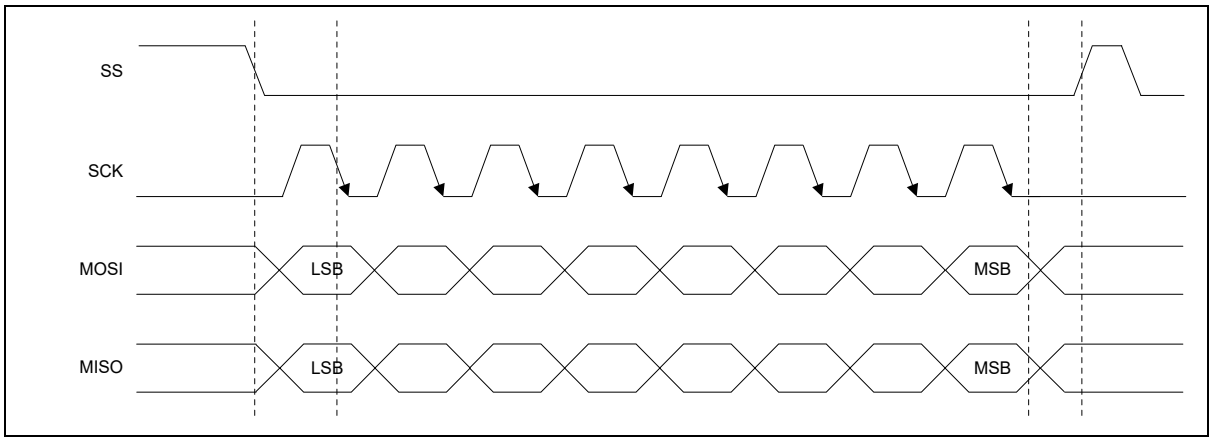
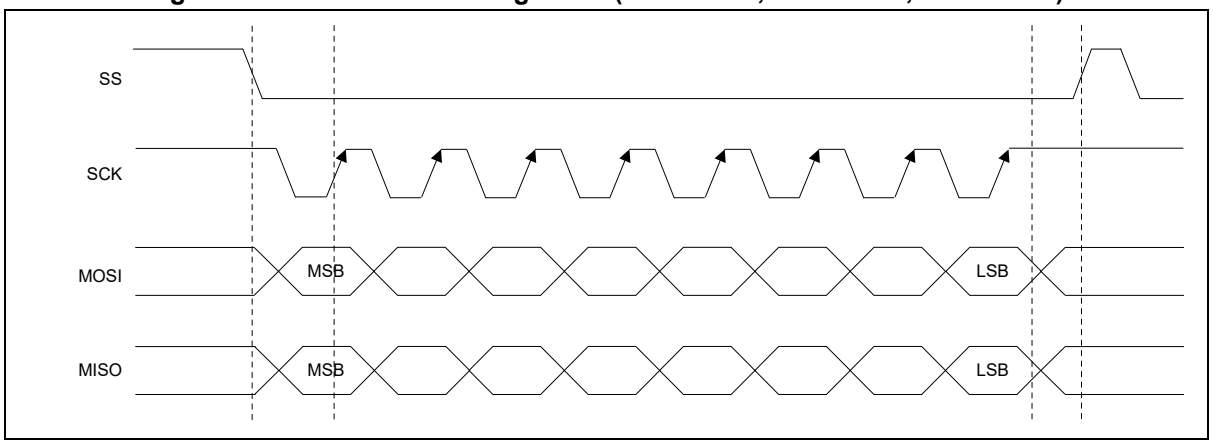


Figure 99. SPI Transfer Timing 4 of 4 (CPHA = ‘1’, CPOL = ‘1’, MSBF = ‘1’)



13.3.5.2 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.

13.3.6 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 SPIn_CR register to determine the data transfer size.
9. In Master mode, set the SSMO in SPIn_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 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 discussed in the later sections in this chapter.

For more information on data communication, refer to section 13.3.8

13.3.7 Procedure for Enabling SPI

The SPIn_EN register is used to enable or disable the SPI. When the ENABLE in 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.

13.3.8 Data Transmission and Reception Procedures

13.3.8.1 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 in 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 transfer data to the SPIn_TDR register.

[Polling Method]

1. When the SPI initial setup is complete, set the ENABLE 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 the transfer data 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 in SPIn_EN register to ‘1’.
2. Set the microcontroller core NVIC interrupts.
3. Write dummy data 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 dummy data 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 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 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.

13.3.8.2 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 in 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.

13.3.8.3 Data Packing

Users can designate the SPI transmission / reception data size. By setting BITSZ[1:0] in SPIn_CR register, users can select 17, 16, 9, 8-bit.

13.3.8.4 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.)

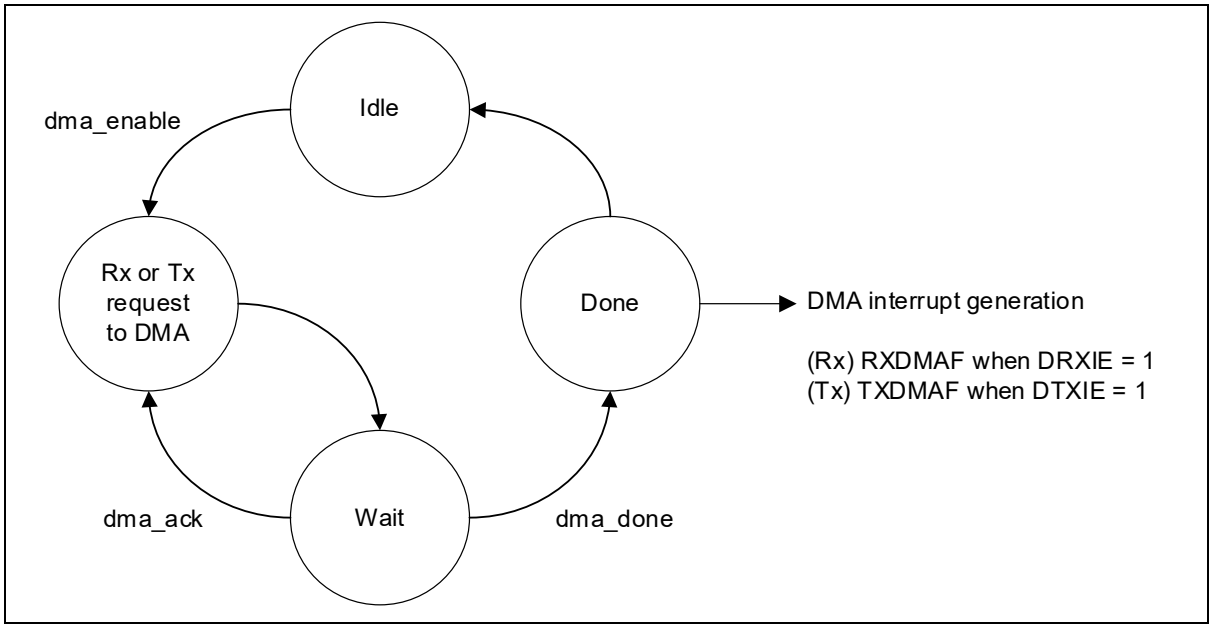
The SPI uses two DMA channels. As the transmitter and receiver are independent of each other, the SPI can operate the two channels at the same time. Once the DMA channel for the receiver is enabled and the Receive Buffer (SPI_RDR register) is filled, the SPI sends an Rx request to the DMA to empty the buffer and waits for an acknowledge signal from the DMA.

If the Receive Buffer (SPI_RDR register) is filled again after the acknowledge signal, the SPI sends an Rx request. If the DMA Rx DONE signal goes high, the RXDMAF bit in the SPI_SR register is set to '1', and an interrupt is serviced when the DRXIE bit in the SPI_CR register is set.

Similarly, if the Transmit Buffer (SPI_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 in SPI_SR register becomes '1' and an interrupt is serviced when the DTXIE in SPI_CR register is set. The Slave Transmitter sends dummy data at the first transfer (For 8–17 SCK pulses) in DMA handshake mode.

Figure 100. DMA Handshake State Diagram



13.3.8.5 Communication Diagrams with DMA

How to set up the SPI using DMA is shown below:

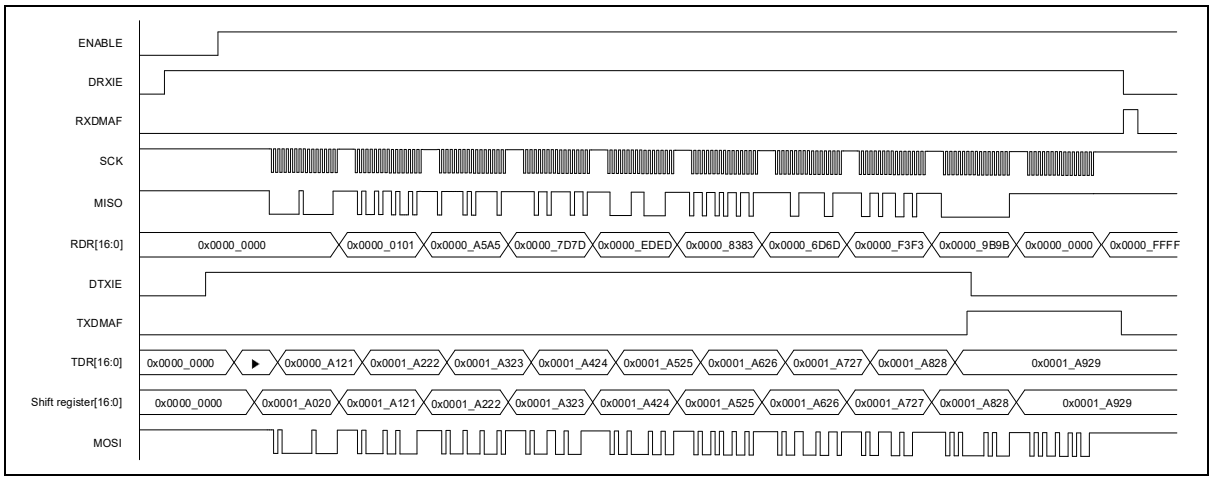
[DMA TX]

1. Set up the initial settings of DMA and SPI.
   A. Enable the DMAEN in 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[3:0] select the SPI_Tx.
      iii. The SIZE[1:0] selects 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 or 0x4000_9100, 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 DTXIE (DMA Tx Done Interrupt Enable) in SPI_CR register and SPI NVIC settings.
   A. Once the values are set in the previous steps, enable the DMAEN in 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] in DMAn_CR register reaches ‘0’, the TXDMAF in 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 in 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[3:0] select the SPI_Rx.
      iii. The SIZE[1:0] selects 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 or 0x4000_9100, the address of the SPI_RDR register.
   D. To the DMAn_MAR register, write the initial address of the memory to be read.
2. Enable DRXIE (DMA Rx Done Interrupt Enable) in SPI_CR register and SPI NVIC settings.
   A. Once the values are set in the previous steps, enable the DMAEN in 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] in DMA_CR register reaches '0', the RXDMAF in the DMAn_SR register indicates that the DMA transfer ends is set to '1'.
   C. Read SPI data into memory.

Figure 101. Example of Timing of SPI Transfer Operation Using DMA



13.3.9 SPI Status Flags

13.3.9.1 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.

13.3.9.2 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.

13.3.9.3 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.

13.3.9.4 Slave Select 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.

13.3.9.5 Rising or Falling Edge of Slave Select 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’.

13.3.9.6 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’.

13.3.9.7 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’.

13.3.10 SPI Error Flags

13.3.10.1 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.

13.3.10.2 Transmit Underrun Error Flag (UDRF)

The UDRF flag is set to ‘1’ when performing the SPI data transmitting operation without data ready to be transmitted to the SPI_TDR register. Writing data to the SPI_TDR register clears the UDRF flag.

13.4 SPI Registers

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

Table 100. Base Address of SPI

Name	Base Address
SPI0	0x4000_9000
SIP1	0x4000_9100

Table 101. SPI Register Map

Name	Offset	Type	Description	Reset Value	Reference
SPIn_TDR	0x0000	WO	SPI transmit data register	0x0000_0000	13.4.1
SPIn_RDR	0x0000	RO	SPI receive data register	0x0000_0000	13.4.2
SPIn_CR	0x0004	RW	SPI control register	0x0000_1020	13.4.3
SPIn_SR	0x0008	RC	SPI status register	0x0000_0006	13.4.4
SPIn_BR	0x000C	RW	SPI baud rate register	0x0000_FFFF	13.4.5
SPIn_EN	0x0010	RW	SPI enable register	0x0000_0000	13.4.6
SPIn_LR	0x0014	RW	SPI delay length register	0x0001_0101	13.4.7

NOTE:

1. n = 0 and 1.

13.4.1 SPIn_TDR: SPI Transmit Data Register

The SPIn_TDR register is a 17-bit sized read / write register. It contains serial transmit data.

SPI0_TDR=0x4000_9000, SPI1_TDR=0x4000_9100

31 30 29 28 27 26 25 24	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

13.4.2 SPIn_RDR: SPI Receive Data Register

The SPIn_RDR register is a 17-bit sized read / write register. It contains serial receive data.

SPI0_RDR=0x4000_9000, SPI1_RDR=0x4000_9100

31 30 29 28 27 26 25 24	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]	
-		-	
-		WO	

16 0	RDR[16:0]	Receive data register

13.4.3 SPIn_CR: SPI Control Register

The SPIn_CR register is a 20-bit sized read / write register and can configure SPI operation mode.

SPI0_CR=0x4000_9004, SPI1_CR=0x4000_9104

31 30 29 28 27 26 25 24	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 select 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 bits
		01	9 bits
		10	16 bits
		11	17 bits

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

13.4.4 SPIn_SR: SPI Status Register

The SPIn_SR register is a 10-bit sized read / write register. It contains the status of SPI interface.

SPI0_SR=0x4000_9008, SPI1_SR=0x4000_9108

31 30 29 28 27 26 25 24	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			TXDMAF	RXDMAF	SBUSY	SSDET	SSON	OVRF	UDRF	TXIDLE	TRDY	RRDY
-			0	0	0	0	0	0	0	1	1	0
-			RWC0	RWC0	RO	RWC0	RC	RC	RO	RO	RO	RO

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 .
3	UDRF	Transmit underrun error flag.
	0	Transmit underrun does not occur.
	1	Transmit underrun occurs. This bit is cleared by writing SPI_TDR.
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.

13.4.5 SPIn_BR: SPI Baud Rate Register

The SPIn_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

31 30 29 28 27 26 25 24	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 two or greater. (BR[15:0] ≥ 2)

NOTE:

1. For SPI speed, it is recommended to set the BR[15:0] value to 2 or higher so that the SPI input clock is divided by at least three.
   Example) PCLK = 24 MHz, BR = 2, SPI freq. = 24 MHz / (2 + 1) = 8 MHz

13.4.6 SPIn_EN: SPI Enable Register

The SPIn_EN register is a bit sized read / write register. It contains SPI enable bit.

SPI0_EN=0x4000_9010, SPI1_EN=0x4000_9110

31 30 29 28 27 26 25 24	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.

13.4.7 SPIn_LR: SPI Delay Length Register

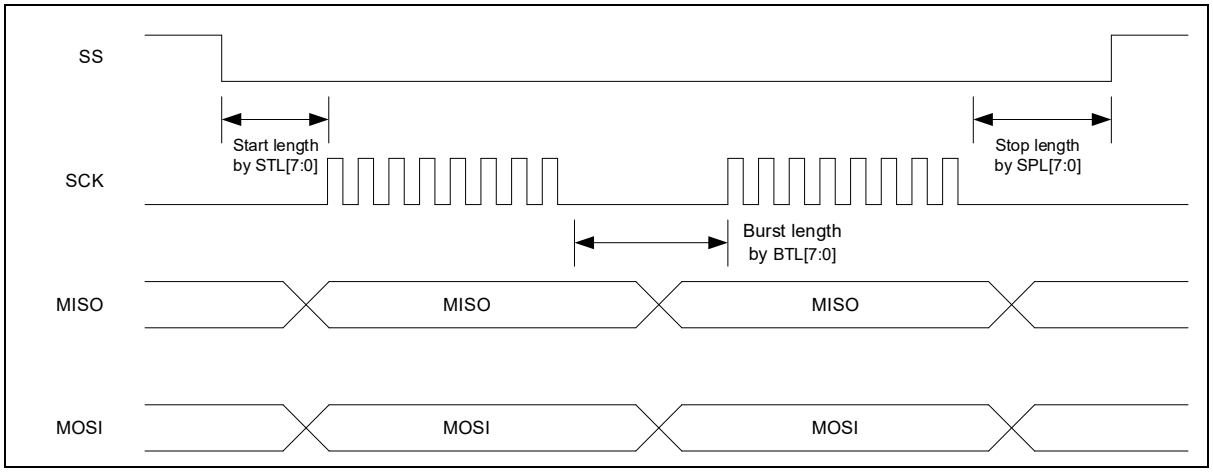
The SPIn_LR register contains the length values of Start, Burst, and Stop. The SPI must be disabled in the SPIn_EN register before configuring this register.

SPI0_LR=0x4000_9014, SPI1_LR=0x4000_9114

31 30 29 28 27 26 25 24	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 102. SPI Waveforms (STL[7:0], BTL[7:0], and SPL[7:0])(1)



NOTE:

1. Due to the nature of code flash, the access time is slower than that of RAM, resulting in flash waits. The data reception during SPI high speed communication performed in code flash may be faster than the data access time. Therefore, inserting a delay between SPI data transmission is recommended. By executing code in SRAM and using DMA, high speed communication can be performed without delay.

$$BTL(burst\ delay) \geq \frac{(Access\ wait + 1) \times SCK \times (xbit + 2)}{HCLK} + \frac{4}{(BR + 1)}$$

13.4.8 SPI Registers Map Summary

Table 102. 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	SPI_TDR	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	TDR[16:0]																
	Reset value																-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-
0x00	SPI_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	SPI_CR	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	TXBC	RXBC	DTXIE	DRXIE	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	0	0	0	0	0			0	0	0	0	0	0
0x08	SPI_SR	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	TXDMAF	RXDMAF	SBUSY	SSDET	SSON	OVRF	UDRF	TXIDLE	TRDY	RRDY
	Reset value																							0	0	0	0	0	0	0	0	0	0
0x0C	SPI_BR	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	BR[15:0]															
	Reset value																	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
0x10	SPI_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	SPI_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	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

14. I2C (Inter-Integrated Circuit)

14.1 I2C Introduction

The I2C (Inter-integrated Circuit) interface built in A34L716 satisfies the standard I2C communication protocol and is used for serial communication with internal and external devices via the I2C protocol. Equipped with two 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 A34L716 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 drive the output signals ‘HIGH’.

14.2 I2C Main Features

The I2C of the A34L716 features the following:

- Compliant with I2C protocol
- Two units supported.
- Master and slave modes
- Multi-slave mode
- General call
- Configurable transfer rates.
  - 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 summarized features of the I2C are shown in Table 103.

Table 103. Features of I2C

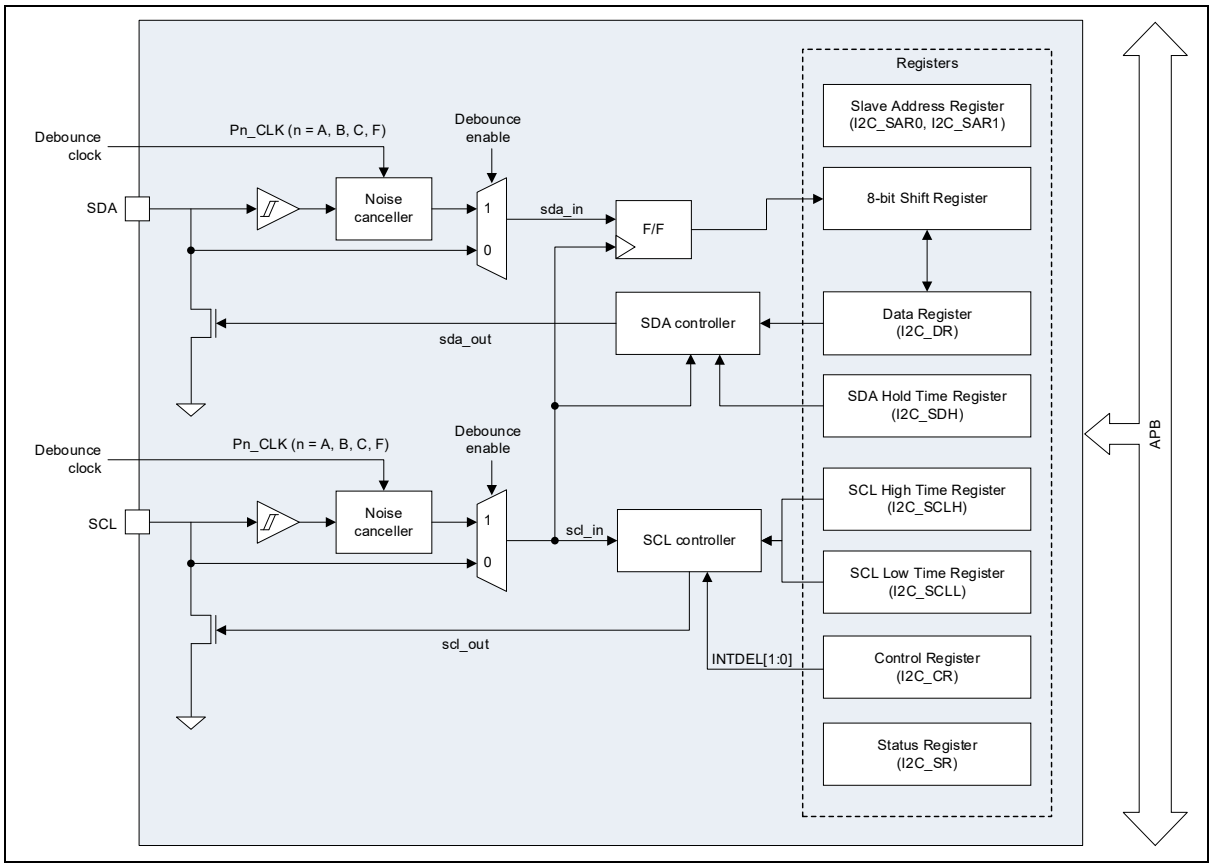
I2C Features	Availability
7-bit addressing mode	O
Standard-mode (up to 100 kbit/s)	O
Fast-mode (up to 400 kbit/s)	O
General call	O

14.4 I2C Functional Description

14.4.1 I2C Block Diagram

Figure 103 shows a block diagram of the I2C module.

Figure 103. 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 set to the 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 104 describes pins assigned for the I2C interface.

Table 104. Pin Assignment of I2C: External Pins

Pin Name	Type	Description	Package Type		
			A34L716 (100-LQFP)	A34L716 (64-LQFP)	A34L716 (56-QFN)
SCL0	I/O	I2C0 serial clock bus line	O	O	O
SDA0	I/O	I2C0 serial data bus line	O	O	O
SCL1	I/O	I2C1 serial clock bus line	O	O	O
SDA1	I/O	I2C1 serial data bus line	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 in I2Cn_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

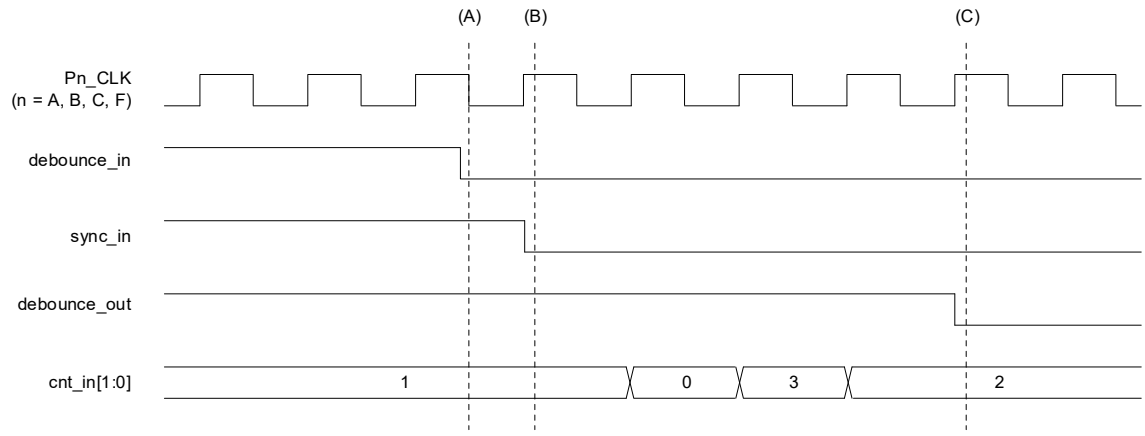
14.4.4.1 Enabling or Disabling I2C

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 I2Cn_CR register to control the I2C communication.

14.4.4.2 Noise Canceller

Figure 104. Noise Canceller Timing



Pn_CLK: The clock selected by Port n (A, B, C, F) for using debounce.
debounce_in: Noise canceller input signal
sync_in: The Internal signal that synchronizes the debounce_in signal to the rising of Pn_CLK.
debounce_out: Debounced output signal
cnt_in: Internal 2-bit counter

To apply 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 the 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, F)

14.4.4.3 Using Noise Canceller in I2C Module

Here’s how to use a noise canceller.

1. The I2C module uses PA, PB, PC, PF. So, select PGADCSEL (PA, PB) in SCU_MCCR4 register or PGCDCSEL(PC), PGBDCSEL (PF) in SCU_MCCR5 register. According to the port to be used and set the clock source for debouncing.
2. Set the PGADCDIV (PA, PB) in SCU_MCCR4 register or PGCDCDIV (PF), PGBDCDIV (PC) in SCU_MCCR5 register to divide the clock source for debouncing.
3. Enable debouncing by setting the PDEn (n = 0 to 15) in Pn_DER (n = A, B, C, F) register 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] in 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 I2C in SCU_PER2 register to ‘1’ again.

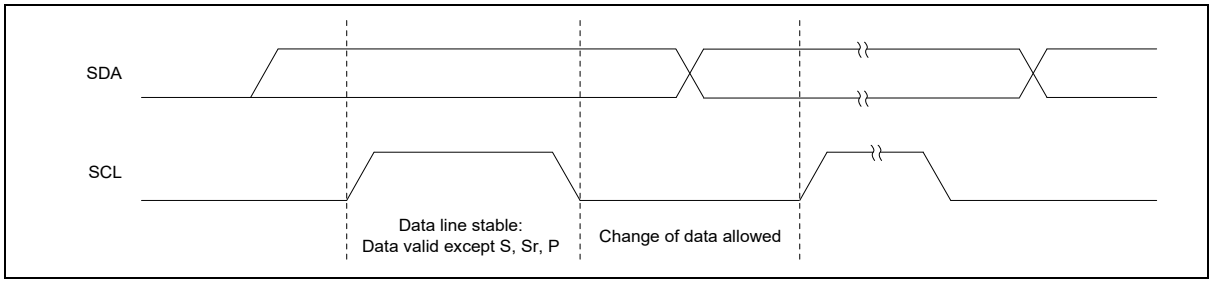
Setting the SOFTRST bit in the I2Cn_CR register applies software reset on the I2C block. The value of the SOFTRST bit is only valid when 0x1756 is simultaneously written to the bits [31:16] in the I2Cn_CR register.

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** 105 shows a timing diagram of the I2C Bus Bit Transfer.

Figure 105. I2C Bus Bit Transfer Timing Diagram



14.4.6.2 START/Repeated START/STOP

During the operation of the I2C bus, unique situations arise, which are defined as the START (S) and STOP (P) conditions (See Figure 106):

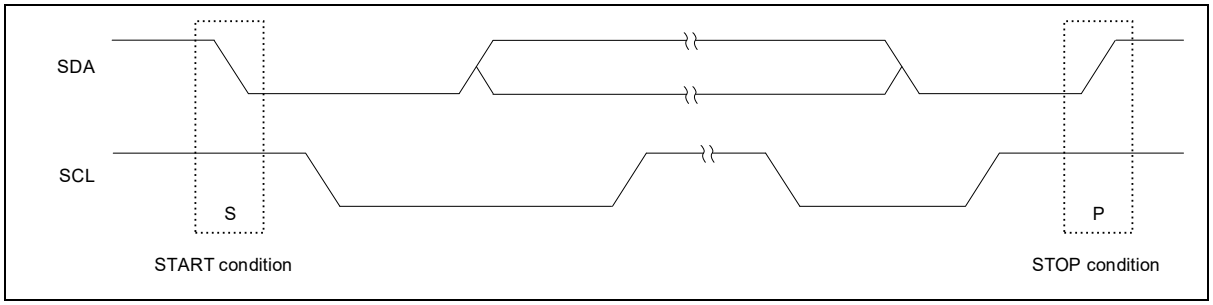
- START (S) condition: An “H” to “L” transition on the SDA line while SCL is “H”.
- STOP (P) condition: An “L” to “H” transition on the SDA line while SCL is “H”.

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 must sample the SDA line at least twice per clock period to sense the transition.

Figure 106. START and STOP Conditions



14.4.6.3 Acknowledge

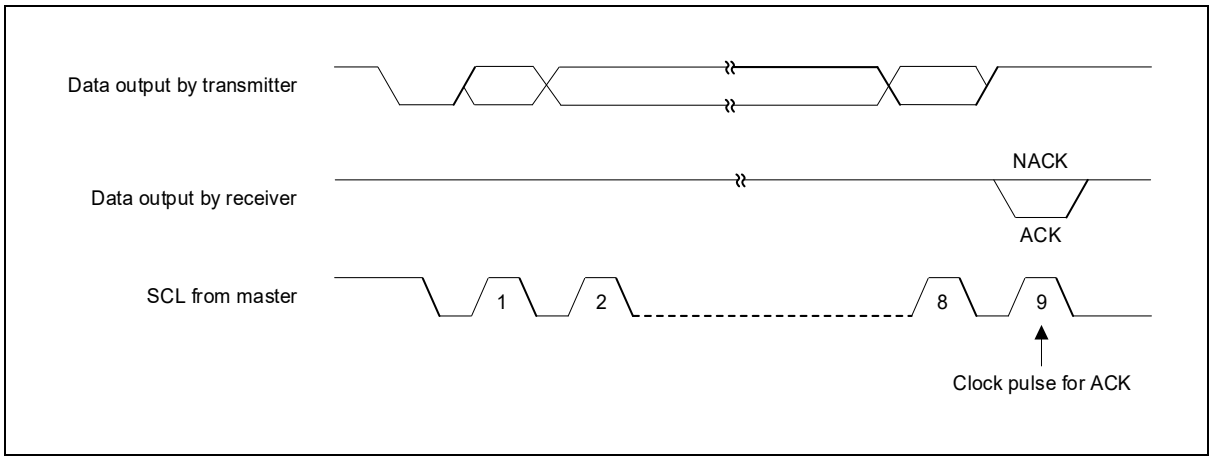
Data transfer with acknowledgement is necessary. The acknowledge-related clock pulse is generated by the master. The transmitter releases the SDA line (“H”) during the acknowledge clock pulse. The receiver must pull down the SDA line during the acknowledge clock pulse so that it remains at stable “L” during the “H” period of this clock pulse. (See Figure 107)

Set-up and hold times must also be considered. When a slave doesn’t 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 be generating the STOP or repeated START condition.

Figure 107. I2C Bus Response

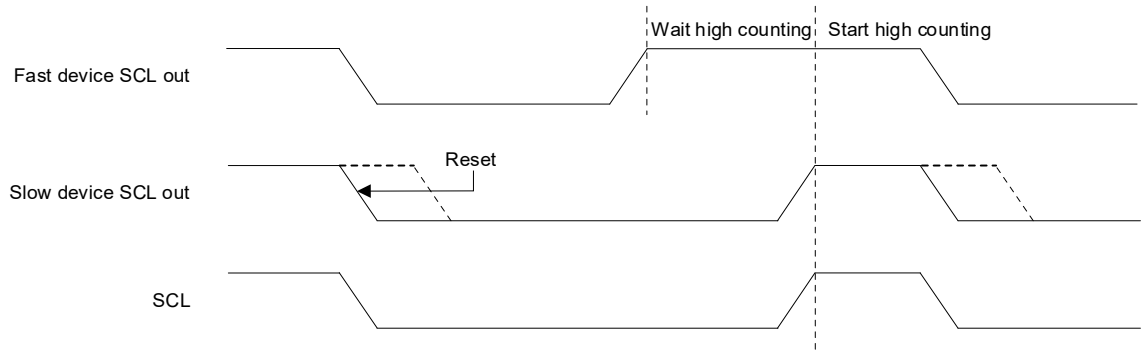


14.4.6.4 Synchronization

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 108. Clock Synchronization during Arbitration



NOTE:

1. However, 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.

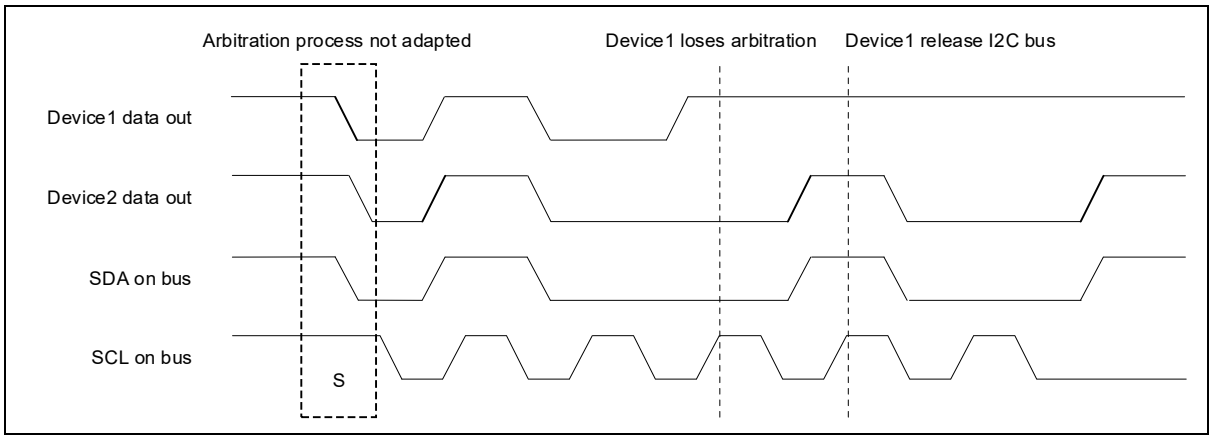
Arbitration can continue for many bits. Its first stage is the comparison of the address bits. If the masters are trying to address the same device, arbitration proceeds with comparing either the data-bits (If they are master-transmitters) or the acknowledge-bits (If they are master-receivers).

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 109 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 109. Arbitration Process between Two Masters

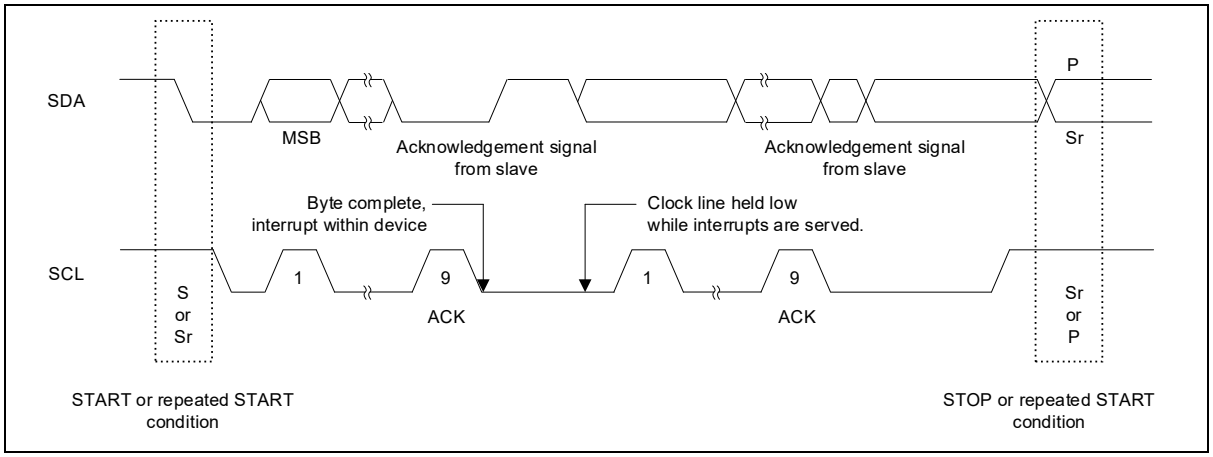


14.4.6.6 Data Transfer

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. (See Figure 110). If a slave can't 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.

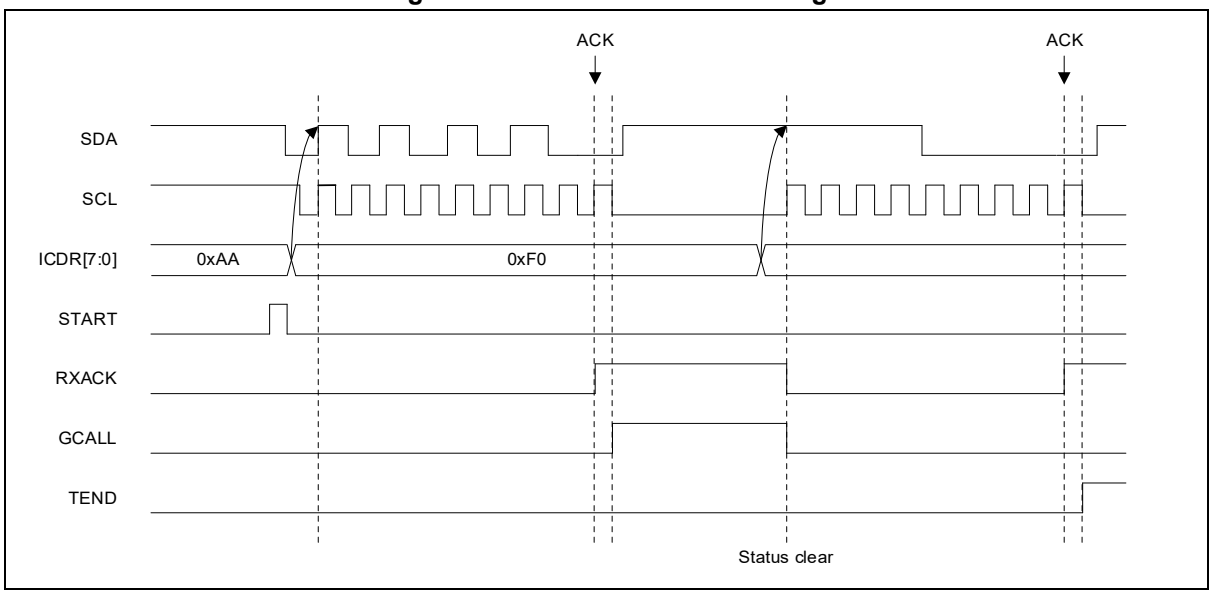
A message starting with such an address can be terminated by an occurrence of the STOP condition, even during the transmission of a byte. In this case, no acknowledgement is generated.

Figure 110. I2C Bus Data Transfer



14.4.6.7 Master Transmit Timing

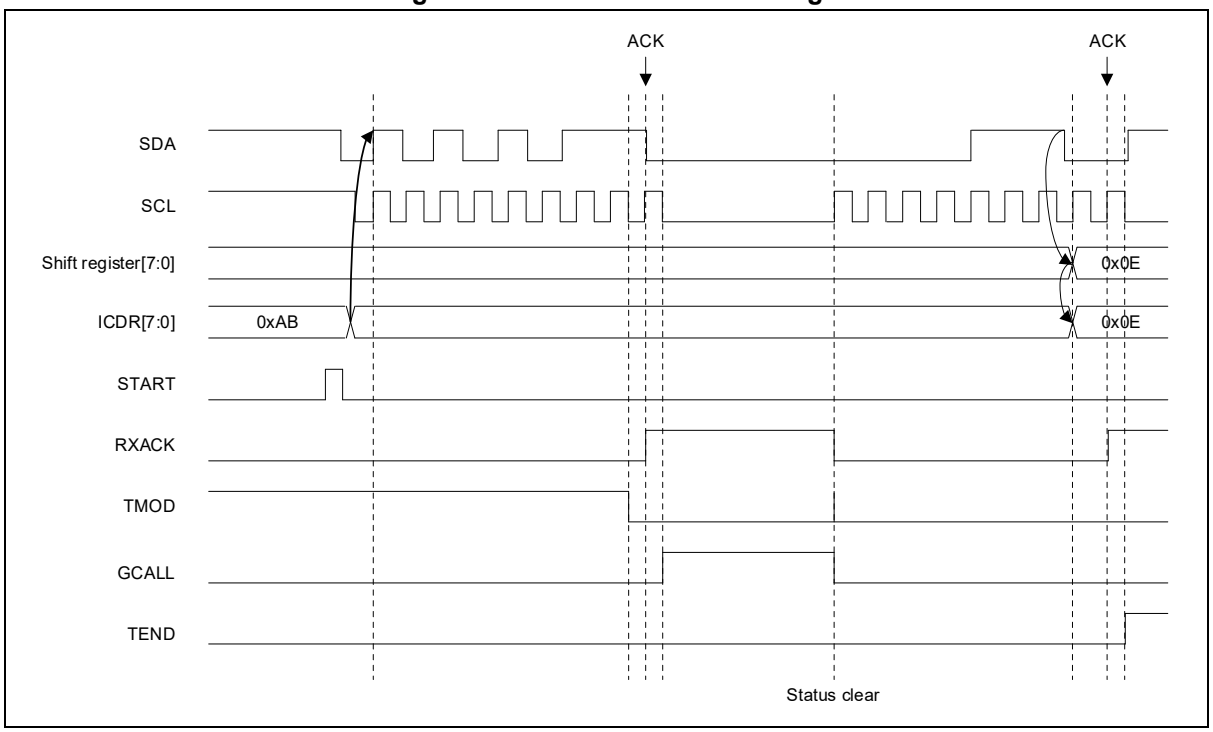
Figure 111. Master Transmit Timing



1. When bit 0 stored in I2Cn_DR register is ‘0’, if START in 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 SCL 8 pulses and ACK is received at the rising edge of the 9th pulse.
3. If ACK is normally received, the RXACK in I2Cn_SR register becomes ‘1’ and the GCALL in I2Cn_SR register becomes ‘1’ at the falling edge of the 9th pulse. In Master mode, GCALL in 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 in I2Cn_SR register becomes 1.

14.4.6.8 Master Receive Timing

Figure 112. Master Receive Timing

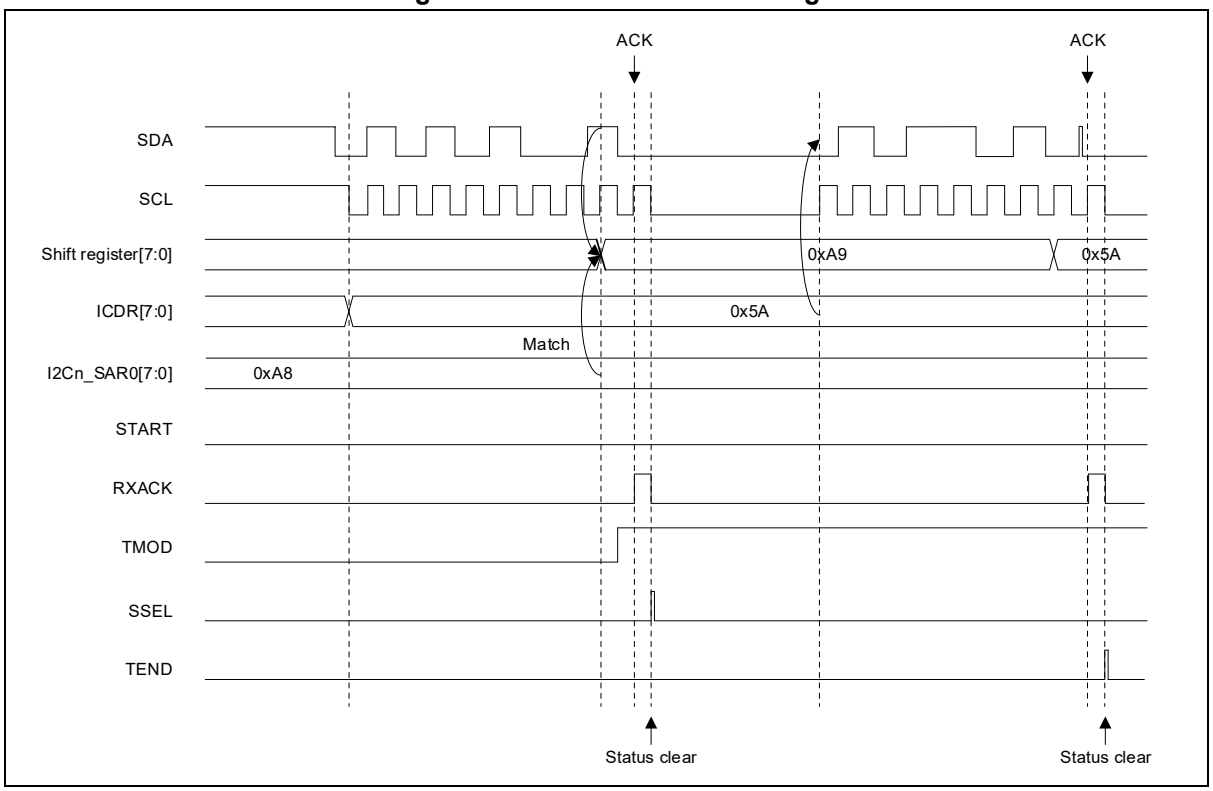


1. When bit 0 stored in I2Cn_DR register is ‘1’, if START in I2Cn_CR register is written as ‘1’, Master operates in Receive mode.
2. 8-bit data stored in I2Cn_DR register is transmitted during SCL 8 pulses, and TMODE in I2Cn_SR register becomes ‘0’ (Receive mode).
3. If ACK is normally received at the rising edge of the 9th pulse, the RXACK in I2Cn_SR register becomes ‘1’ and the GCALL in I2Cn_SR register becomes ‘1’ at the falling edge of the 9th pulse. In Master mode, GCALL in I2Cn_SR register is used as AACK (Address ACK judgment) bit.
4. If AACK is confirmed after sending address, clear 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 in I2C_SR register becomes ‘1’. When data reception is completed, the TEND in I2Cn_SR register becomes 1.

14.4.6.9 Slave Transmit Timing

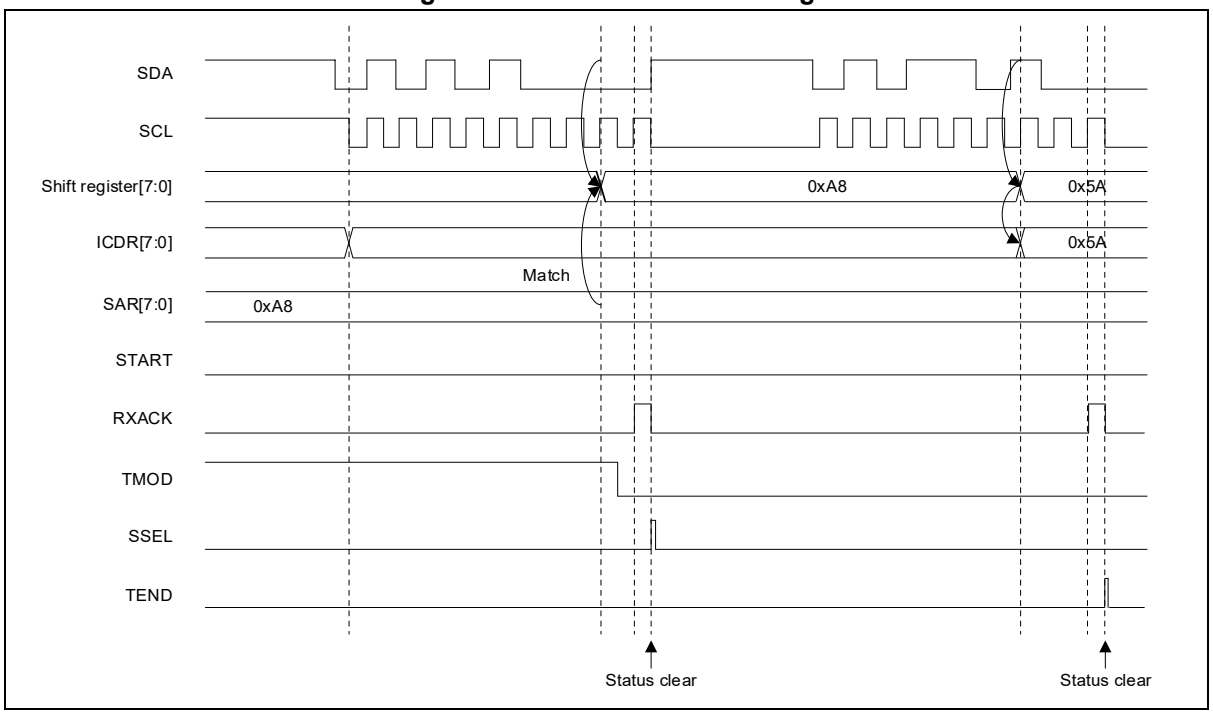
Figure 113. Slave Transmit Timing



1. If START in 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_SAR0 or I2Cn_SAR1 register value and the Address (SHFTR[7:1]) value stored in the shift register (SHFTR) match, it is Transmit mode, so the TMODE in I2Cn_SR register is '1'.
4. When the Address Match is reached, the SSEL in 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, TEND bit 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 114. Slave Receive Timing



1. If START in 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_SAR0 or I2Cn_SAR1 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 in I2Cn_SR register is '0'.
4. When the Address Match is reached, SSEL in 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.

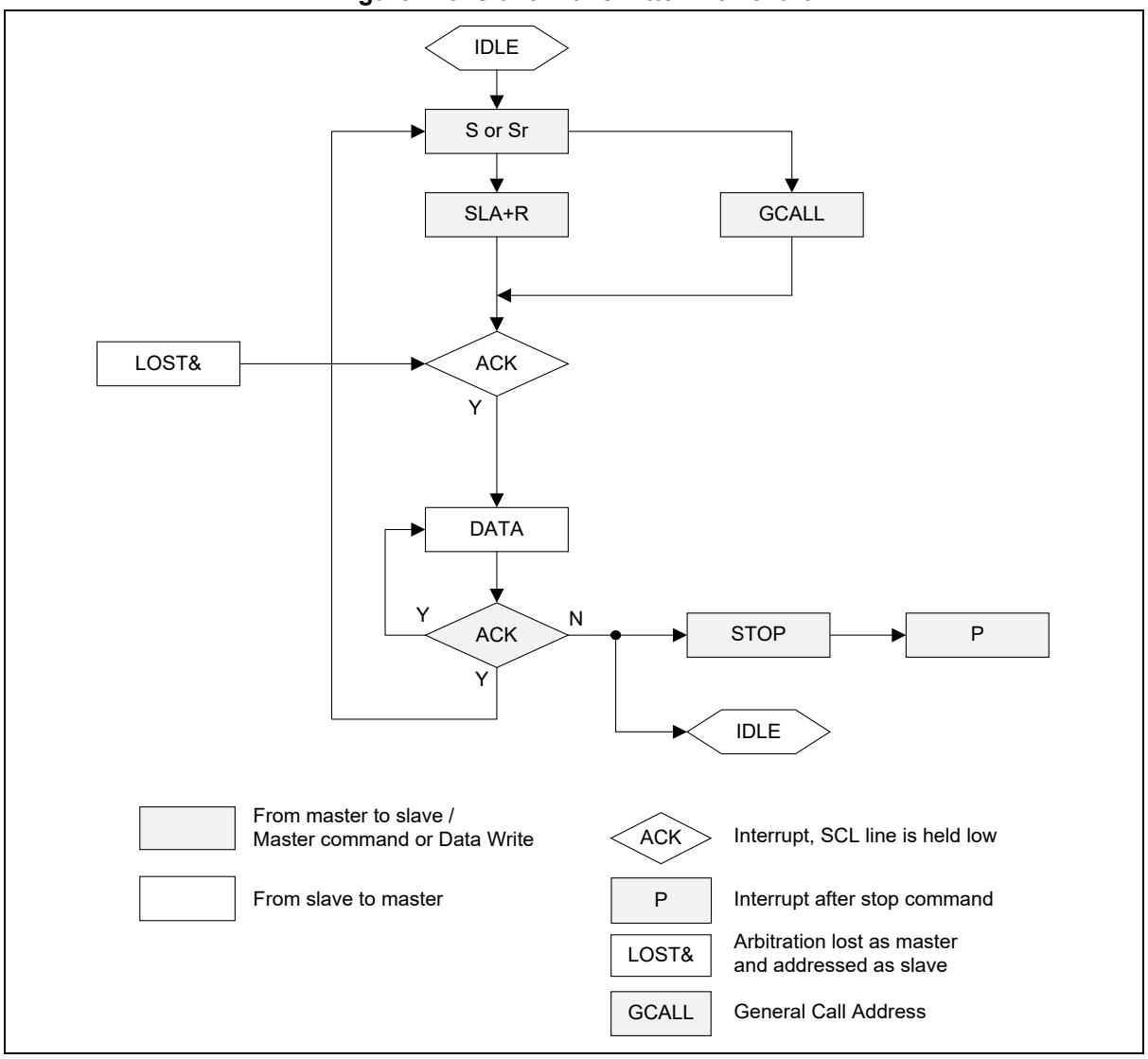
At the 9th pulse of SCL, the RXACK in I2Cn_SR register becomes ‘1’. When data reception is completed, the TEND in I2Cn_SR register becomes ‘1’.

14.4.7 I2C Slave Mode

14.4.7.1 Slave Transmitter

Figure 115 shows a transmitter's flowchart in slave mode.

Figure 115. 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 I2C_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 I2C_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 in SCU_PER2 and SCU_PCER2 registers to enable and clock the I2C.
3. Set the I2CEN in I2Cn_CR register to enable the I2C.

4. Set the INTEN in I2Cn_CR register to enable the I2C interrupts.
5. When the START condition is detected, the I2C receives one byte of data and compares it with value of the SVAD[6:0] in I2Cn_SAR0 or I2Cn_SAR1 register. In the case that the GCEN in I2Cn_SAR0 or I2Cn_SAR1 register is set enabled, the I2C raises the “general call” flag if the received data is 0x00.
6. If the received address does not match the SVAD[6:0] in I2Cn_SAR0 or I2Cn_SAR1 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.
7. One byte of data is transmitted.
8. 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.

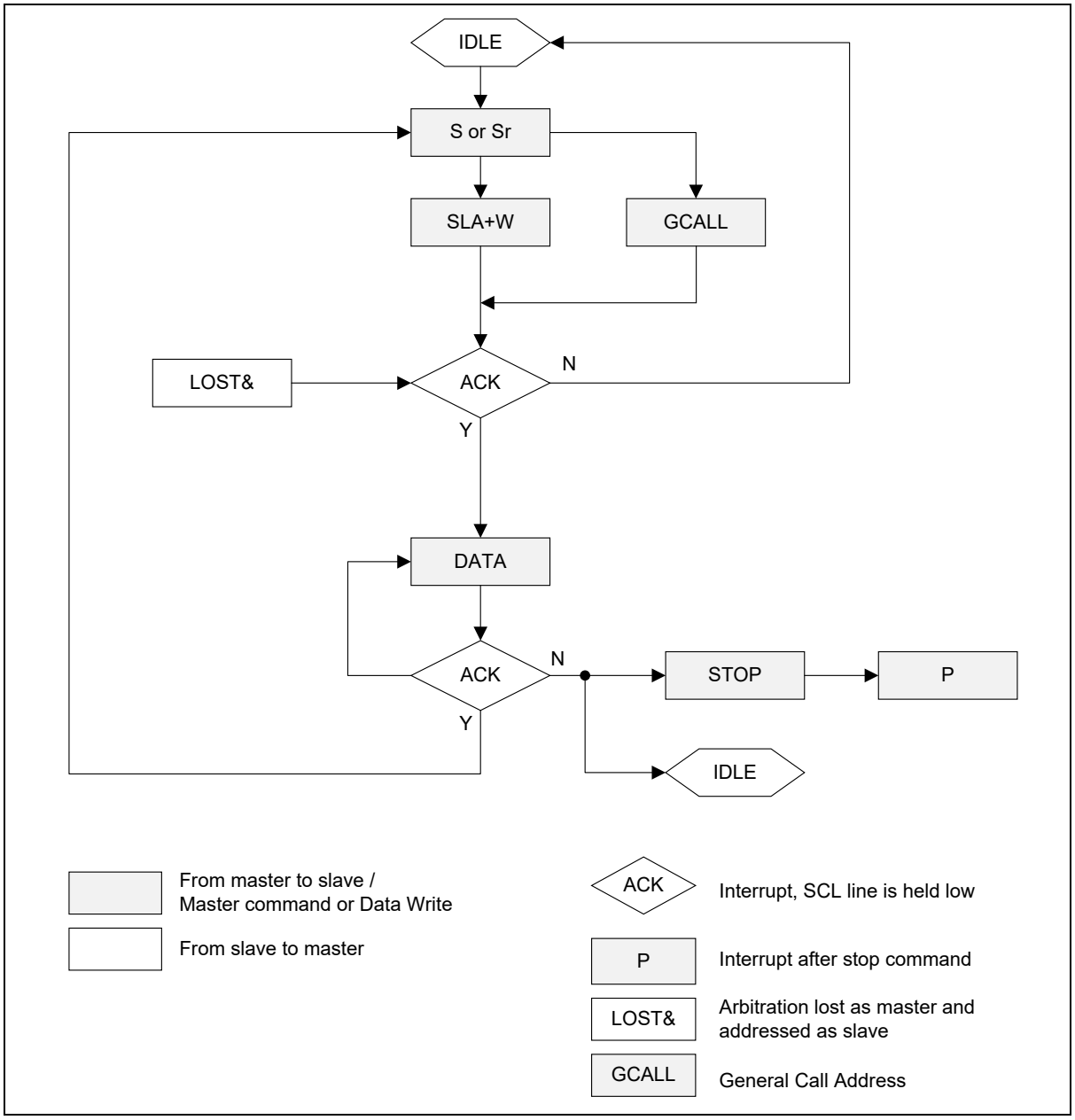
   Case 2. If an ACK signal from the master is detected, the I2C loads the next data in the I2Cn_DR register.

   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.
9. 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.

14.4.7.2 Slave Receiver

Figure 116 shows a flowchart of the receiver in slave mode.

Figure 116. 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 I2C_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 I2C_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 in SCU_PER2 and SCU_PCER2 registers to enable and clock the I2C.

3. Set the I2CEN in I2Cn_CR register to enable the I2C.
4. Set the INTEN in I2Cn_CR register to enable the I2C interrupts.
5. When the START condition is detected, the I2C receives one byte of data and compares it with the SVAD[6:0] in I2Cn_SAR0 or I2Cn_SAR1 register. In the case that the GCEN in I2Cn_SAR0 or I2Cn_SAR1 register is set enabled, the I2C raises the “general call” flag if the received data is 0x00.
6. If the received address does not match the SVAD[6:0] in I2Cn_SAR0 or I2Cn_SAR1 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.
7. One byte of data is received.
8. 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.

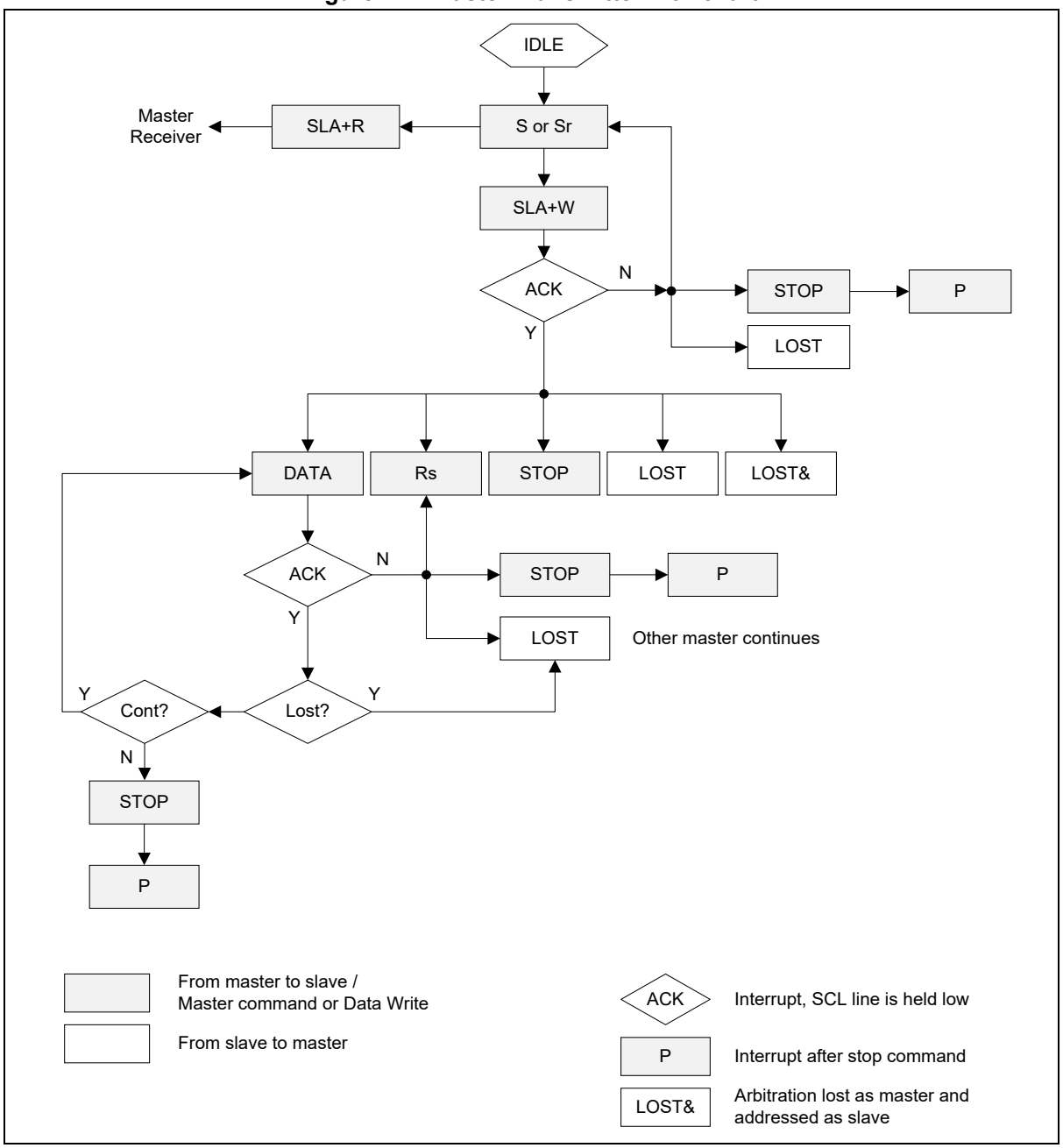
   Once one of the above operations is performed, release the SCL line by writing a value 0xFF to the I2C_SR register. For Case 1, go to step 8 and end the communication. For Case 2, go to step 6. In either case, the consecutive START condition can be detected; then, go to step 4.
9. The last step is to 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 117 shows a flowchart of the transmitter in master mode.

Figure 117. Master Transmitter Flowchart



To operate the I2C in master transmitter mode, users must follow the steps below:

1. Set the I2C in SCU_PER2 and SCU_PCER2 registers to enable and clock the I2C.
2. Set the I2CEN in I2Cn_CR register to enable the I2C.

3. Set the INTEN in I2Cn_CR register. to enable the I2C interrupts.
4. 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.
5. Set the I2Cn_SCLL and I2Cn_SCLH registers to configure the SCL transfer speed.
6. Set the I2Cn_SDH register, which determines the time SCL maintains the SDA value during the “L” period.
7. Write a ‘1’ to the START in 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.
8. 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 interrupt occurs, regardless of whether an ACK has been received. If the I2C loses its bus mastership, the MLOST in 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 in I2Cn_CR register must be set to ‘1’ and the SVAD[6:0] in I2Cn_SAR0 or I2Cn_SAR1 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 whether or not it has received 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 in 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.
9. One byte of data is transmitted. Bus arbitration is valid during data transmission.

10. 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 in 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 whether it has received 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 in 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 in I2Cn_CR register must be set to ‘1’.

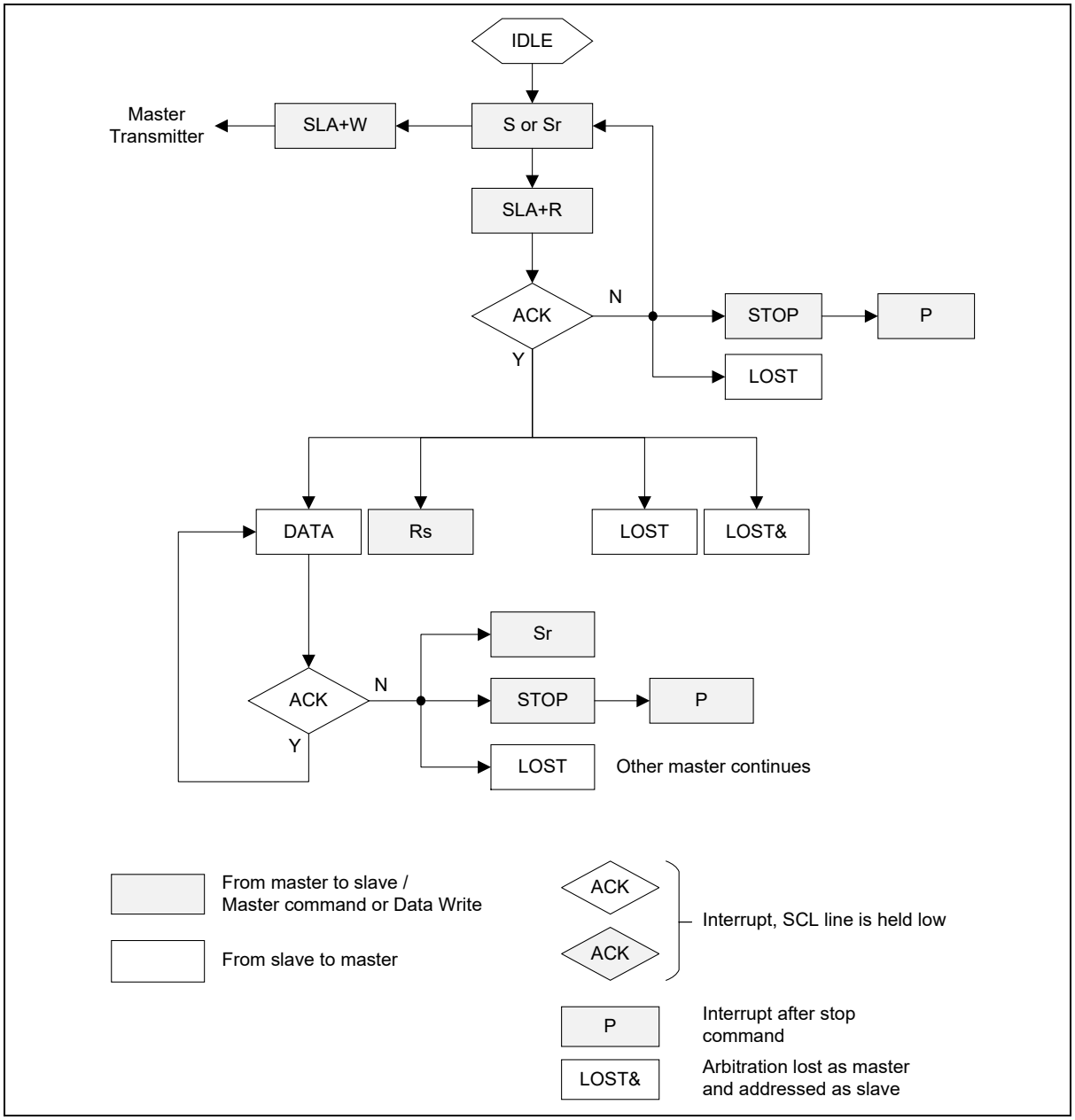
    The SCL line is released when one of the above is performed and a value 0xFF is written to the I2Cn_SR register. For Case A, go to step 8; for Case B, go to step 10; and for Case C, write the data in the I2Cn_DR register and go to step 6. If the transfer direction bit in SLA + R / W that is written to the I2Cn_DR register is set to ‘1’, go to the section on master receiver.

11. 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 118 shows a flowchart of the receiver in master mode.

Figure 118. Master Receiver Flowchart



To operate the I2C in master receiver mode, users must follow the steps below:

1. Set the I2C in SCU_PER2 and SCU_PCER2 registers to enable and clock the I2C.
2. Set the I2CEN in I2Cn_CR register to enable the I2C.
3. Set the INTEN in I2Cn_CR register to enable the I2C interrupts.

4. Enter “SLA + R” in the I2Cn_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.
5. Set the I2Cn_SCLL and I2Cn_SCLH registers to configure the SCL transfer speed.
6. Set the I2Cn_SDH register, which determines the time SCL maintains the SDA value during the “L” period.
7. Write a ‘1’ to the START in 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.
8. 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 interrupt occurs, regardless of whether an ACK has been received. If the I2C loses its bus mastership, the MLOST in 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 in 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 whether it has received 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 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 in 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.
9. One byte of data is received.

10. 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, TEND interrupt occurs in the I2C.

    The I2C (Master) performs one of the following operations regardless of the RXACK in I2C_SR register:

    Case A. The master continues to receive data from the slave. The ACKEN in 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 in 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 in 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 I2Cn_DR register; and the START in 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.

11. 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.

14.4.9 I2C Low-Power Mode

Table 105. Effect of Low-Power Modes on I2C

Mode	Description
SLEEP	No effect
DEEP-SLEEP	Slave mode operates as a wake-up source.

14.4.9.1 Low-Power Mode Sequence

1. To enable the signal to be used as a wake-up source, set I2CnWUE in SCU_WUER register to ‘1’.
2. Set the I2C slave mode. (Refer to 14.4.7 I2C Slave .)
3. Enter low power mode (DEEP-SLEEP mode).
4. When slave address is matched, the Match Event is generated. Then, the SSEL in the I2Cn_SR register and the I2CnWU in the SCU_WUSR register are triggered.
5. Writing ‘1’ to the I2CnWU in the SCU_WUSR register clears the triggered flags.

14.5 I2C Interrupts

Setting the INTEN in 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 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 section 14.4.8.

14.6 I2C Registers

The base address and register map of the I2C are described in the following tables:

Table 106. Base Address of I2C

Name	Base Address
I2C0	0x4000_A000
I2C1	0x4000_A100

Table 107. I2C Register Map

Name	Offset	Type	Description	Reset Value	Reference
I2Cn_DR	0x0000	RW	I2C data register	0x0000_00FF	14.6.1
I2Cn_SR	0x0008	RW	I2C status register	0x0000_0000	14.6.2
I2Cn_SAR0	0x000C	RW	I2C slave address register	0x0000_0000	14.6.3
I2Cn_SAR1	0x0010	RW	I2C slave address register	0x0000_0000	14.6.4
I2Cn_CR	0x0014	RW	I2C control register	0x0000_0000	14.6.5
I2Cn_SCLL	0x0018	RW	I2C SCL low duration register	0x0000_FFFF	14.6.6
I2Cn_SCLH	0x001C	RW	I2C SCL high duration register	0x0000_FFFF	14.6.7
I2Cn_SDH	0x0020	RW	I2C SDA hold register	0x0000_7FFF	14.6.8

NOTE:

1. n = 0 and 1.

14.6.1 I2Cn_DR: I2C Data Register

The I2Cn_DR register is an 8-bit register. It stores received byte-sized data or serial data for transmission.

I2C0_DR=0x4000_A000, I2C1_DR=0x4000_A100

31 30 29 28 27 26 25 24	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.

14.6.2 I2Cn_SR: I2C Status Register

The I2Cn_SR register is a 8-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

31 30 29 28 27 26 25 24	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.

14.6.3 I2Cn_SAR0: I2C Slave Address Register

The I2Cn_SAR0 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_SAR0=0x4000_A00C, I2C1_SAR0=0x4000_A10C

31 30 29 28 27 26 25 24	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	SVAD[6:0](1)	7-bit slave address	
0	GCEN	Whether to enable or disable general call	
		0	Dismisses the general call address.
		1	Receives the general call address.

NOTE:

1. In master mode, the SVAD value must not be equal to the slave address.

14.6.4 I2Cn_SAR1: I2C Slave Address Register

The I2Cn_SAR1 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_SAR1=0x4000_A010, I2C1_SAR1=0x4000_A11C

31 30 29 28 27 26 25 24	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	SVAD[6:0](1)	7-bit slave address	
0	GCEN	Whether to enable or disable general call	
		0	Dismisses the general call address.
		1	Receives the general call address.

NOTE:

1. In master mode, the SVAD value must not be equal to the slave address.

14.6.5 I2Cn_CR: I2C Control Register

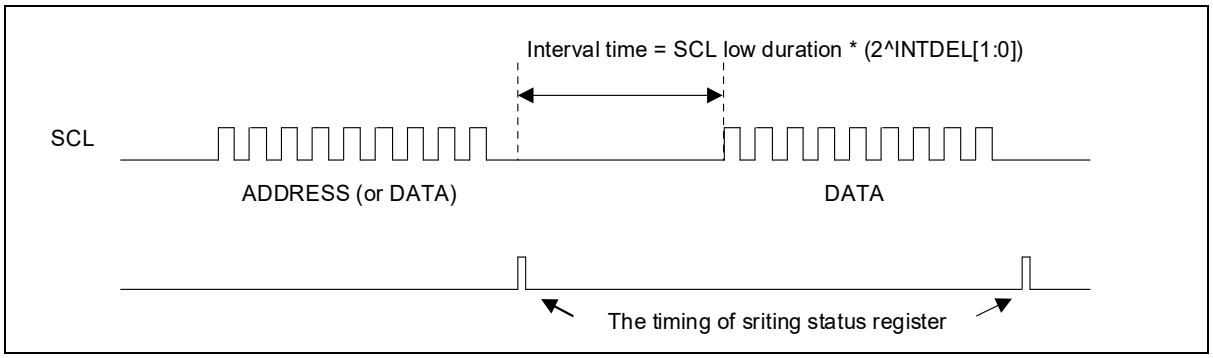
The I2Cn_CR register is a 32-bit read/write accessible register. It sets the operating modes and enablement of the I2C module.

I2C0_CR=0x4000_A014, I2C1_CR=0x4000_A114

31 30 29 28 27 26 25 24	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	INTDEL[1:0]	IIF	I2CEN	SOFTRST	INTEN	ACKEN	IMASTER	STOP	START
-		-	00	0	0	0	0	0	0	0	0
-		-	RW	RO	RW	RW	RW	RW	RO	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.
6	I2CEN	I2C Enable
	0	Disable
	1	Enable
5	SOFTRST	I2C software reset To enable this bit, write bit[15] with '1' and bit[31:16] with 0x1756 together.
	0	No effect
	1	Apply Software Reset on I2C block
4	INTEN	Whether to enable or disable interrupts
	0	Disables interrupts.
	1	Enables interrupts.
3	ACKEN	Whether to enable or disable ACK in receive mode
	0	No ACK signal is generated.
	1	ACK signal is generated.
2	IMASTER	Master mode flag
	0	Slave mode
	1	Master mode
1	STOP	Whether to 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 119. Internal Delay (INTDEL[1:0]) in Master Mode



14.6.6 I2Cn_SCLL: I2C SCL Low-Duration Register

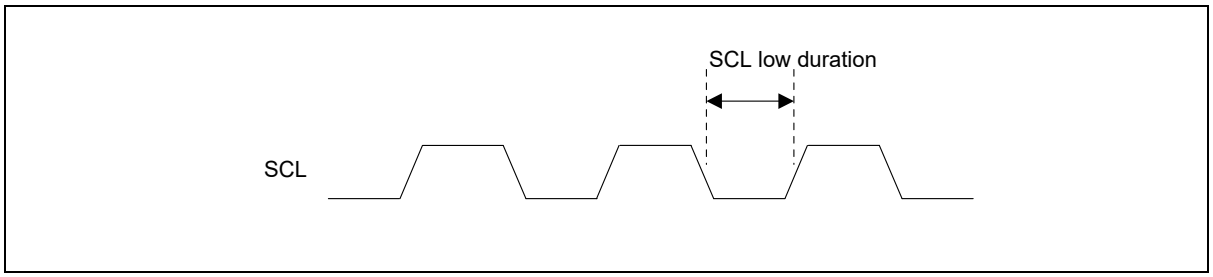
The I2Cn_SCLL register is a 32-bit register. The SCL low duration time can be set in master mode.

I2C0_SCLL=0x4000_A018, I2C1_SCLL=0x4000_A118

31 30 29 28 27 26 25 24	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 120. SCL Low (SCLL[15:0]) Timing



14.6.7 I2Cn_SCLH: I2C SCL High-Duration Register

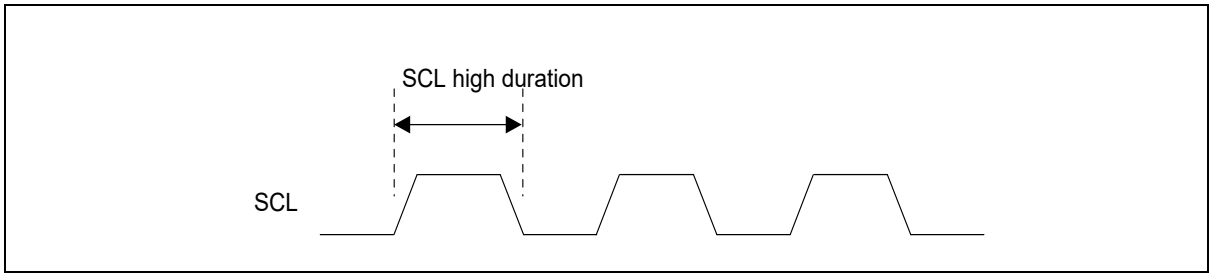
The I2Cn_SCLH register is a 32-bit register. The SCL high duration time can be set in master mode.

I2C0_SCLH=0x4000_A01C, I2C1_SCLH=0x4000_A11C

31 30 29 28 27 26 25 24	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

Figure 121. SCL High (SCLH[15:0]) Timing



NOTE:

1. In case of SCLH, a delay that counts the SCLH section may occur depending on external input conditions (Pull-up resistance, VDDEXT).

14.6.8 I2Cn_SDH: I2C SDA Hold Register

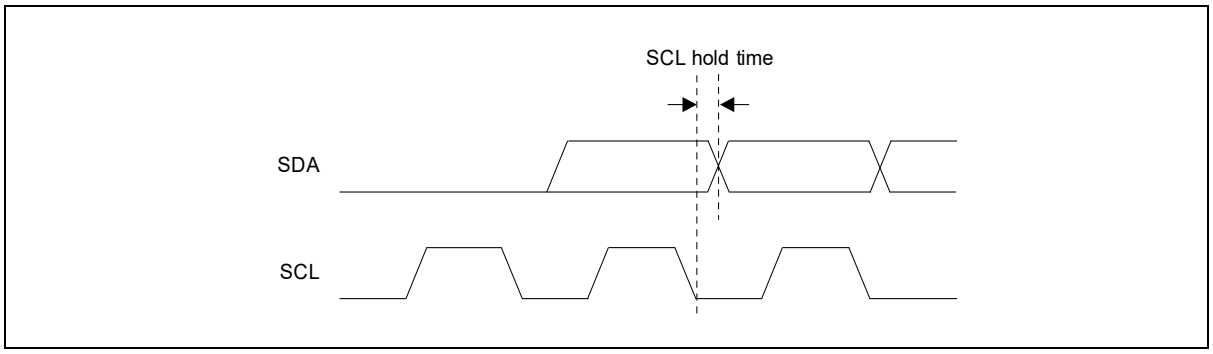
The I2Cn_SDH register is a 15-bit R/W register. The SCL hold time can be set in master mode.

I2C0_SDH=0x4000_A020, I2C1_SDH=0x4000_A120

31 30 29 28 27 26 25 24	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 122. SDA Hold (SDH[14:0]) Timing



14.6.9 I2C Register Map Summary

Table 108. 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	I2C_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	I2C_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	I2C_SAR0	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
0x10	I2C_SAR1	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	I2C_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	I2CEN	SOFTRST	INTEN	ACKEN	IMASTER	STOP	START
	Reset value																							0	0	0	0	0	0	0	0	0	0
0x18	I2C_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	I2C_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	I2C_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. USART (Universal Synchronous/Asynchronous Receiver/Transmitter)

15.1 USART Introduction

Universal Synchronous and Asynchronous serial Receiver and Transmitter (USART) is a highly flexible serial communication device.

15.2 USART Main Features

The USART of the A34L716 features the following:

- Full Duplex operation. (Independent Serial Receive and Transmit Registers)
- Asynchronous or Synchronous operation.
- Baud Rate Generator.
- Supports serial frames with 5,6,7,8 or 9 data bits and 1 or 2 stop bits.
- Odd or Even parity generation and parity check are supported by hardware.
- Data over run detection.
- Framing error detection.
- Three separate interrupts on TX completion, TX data register empty and RX completion.
- Double speed asynchronous communication mode.

Additional features are :

- 0% error baud rate by floating point count register.
- Supports receive time out interrupt.
- Supports direct memory access and interrupt.

15.3 USART Functional Description

USART includes a clock generator, transmitter, and receiver.

The clock generation logic consists of a synchronization logic for external clock input used by synchronizing or SPI slave operation, and the baud rate generator for asynchronous or master (synchronous or SPI) operation.

The transmitter consists of a single write buffer, a serial shift register, parity generator and control logic for handling different serial frame formats. The write buffer allows continuous transfer of data without any delay between frames.

The receiver is the most complex part of the UART module due to its clock and data recovery units. The recovery unit is used for asynchronous data reception. In addition to the recovery unit, the receiver includes a parity checker, a shift register, a two-level receive FIFO (USARTn_DR) and control logic. The receiver supports the same frame formats as the transmitter and can detect frame error, data overrun and parity errors (n = 10 and 11).

15.3.1 USART Block Diagram

Figure 123 shows a block diagram of the USART block.

Figure 123. UART Block Diagram (n = 10 and 11)

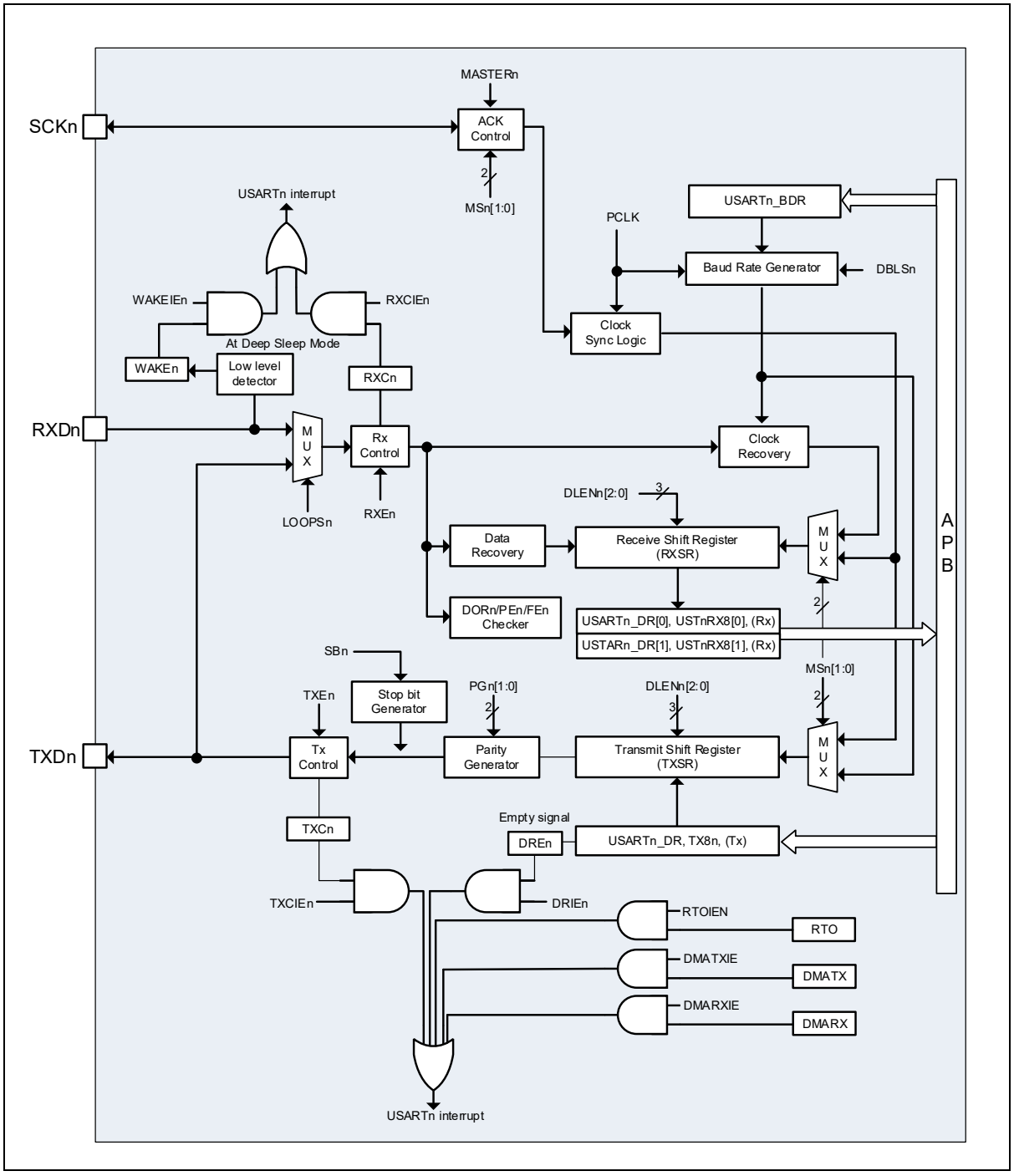
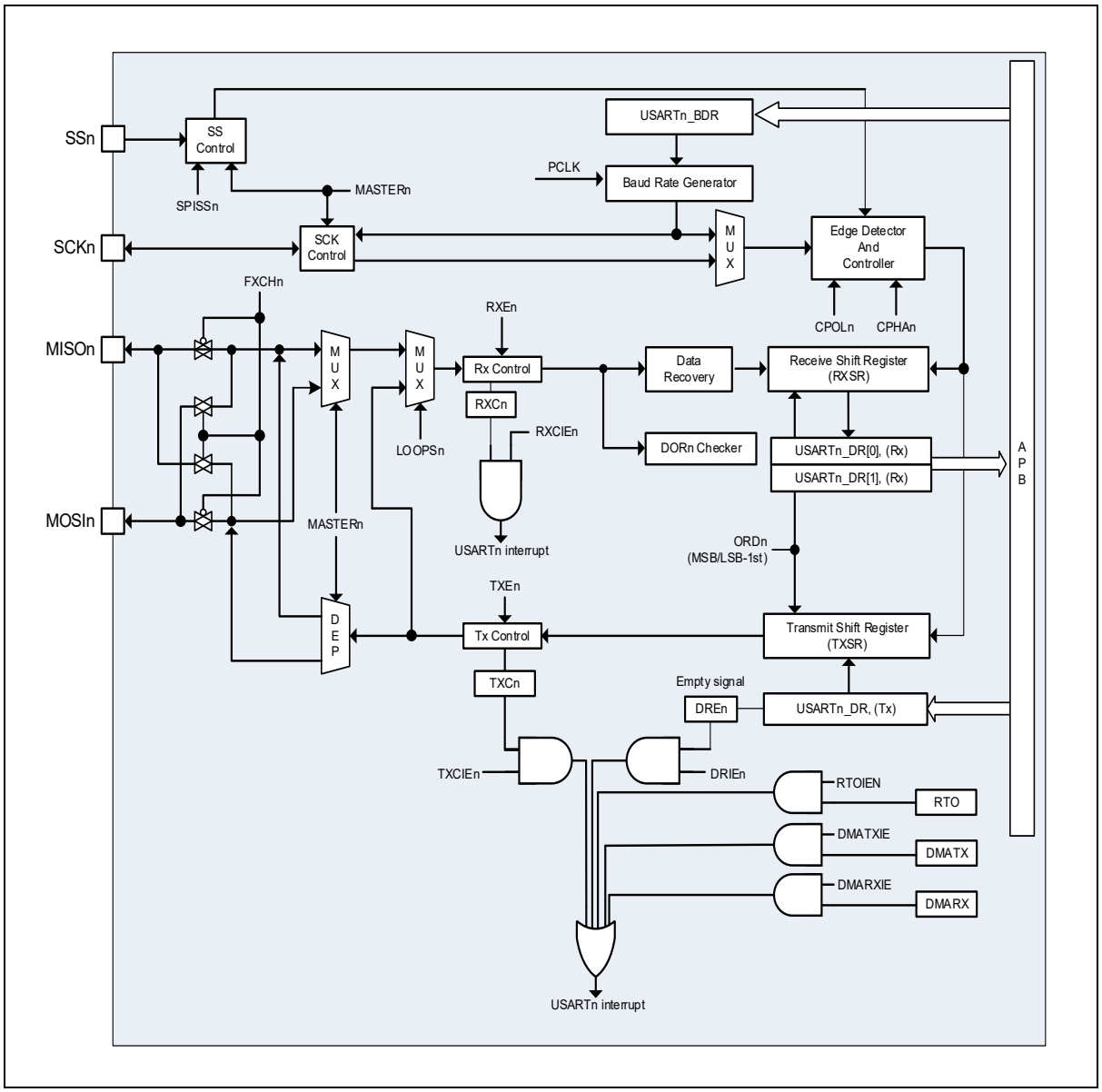


Figure 124. SPI Block Diagram (n = 10 and 11)



15.3.2 USART Pins and Signals

Table 109 describes the external pins assigned for USART.

Table 109. Pin Assignment of USART: External Pins

Pin Name	Type	Description	Supported Packages		
			A34L716 (100-LQFP)	A34L716 (64-LQFP)	A34L716 (56-QFN)
TXD10	O	UART transmit output	O	O	O
RXD10	I	UART transmit input	O	O	O
SS10	I/O	SPI serial port I/O signal for slave selection	O	O	O
SCK10	I/O	SPI clock I/O (Master: output / slave: input)	O	O	O
MOSI10	I/O	SPI transmit and receive data (Master: output / slave: input)	O	O	O
MISO10	I/O	SPI transmit and receive data (Master: input / slave: output)	O	O	O
TXD11	O	UART transmit output	O	O	X
RXD11	I	UART transmit input	O	O	O
SS11	I/O	SPI serial port I/O signal for slave selection	O	O	X
SCK11	I/O	SPI clock I/O (Master: output / slave: input)	O	O	O
MOSI11	I/O	SPI transmit and receive data (Master: output / slave: input)	O	O	X
MISO11	I/O	SPI transmit and receive data (Master: input / slave: output)	O	O	O

15.3.3 USART Clock Generation

The clock generation logic generates the base clock for the transmitter and receiver. The USART supports four modes of clock operation and those are normal asynchronous, double speed asynchronous, master synchronous and slave synchronous modes.

The clock generation scheme for master SPI and slave SPI mode is the same as master synchronous and slave synchronous operation mode.

MS[1:0] bits in USARTn_CR1 register selects asynchronous or synchronous operation. The asynchronous double speed mode is controlled by the DBLS[8] bit in the USARTn_CR2 register.

MASTER[7] bit in USARTn_CR2 register controls whether the clock source is internal (master mode, output pin) or external (slave mode, input pin). The SCKn pin is active only when the USART operates in synchronous or SPI mode.

Figure 125. Clock Generation Block Diagram (n = 10 and 11)

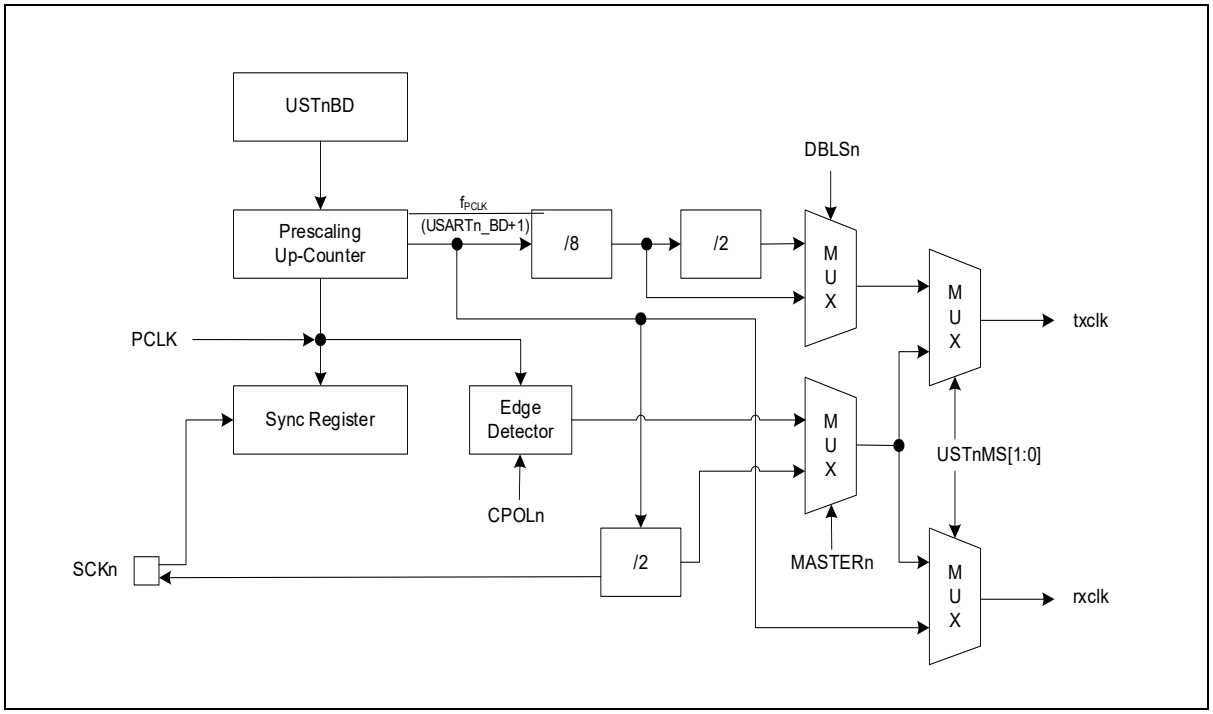


Table 110 shows the equations for calculating baud rate (in bps.)

Table 110. Equations for Calculating USART Baud Rate Register Settings (n= 10 and 11)

Operating Mode	Equation for Calculating Baud Rate
Asynchronous Normal Mode (DBLSn=0)	Baud Rate = PCLK/(16(USARTn_BDR+1))
Asynchronous Double Speed Mode (DBLSn=1)	Baud Rate = PCLK/(8(USARTn_BDR+1))
Synchronous or SPI Master Mode	Baud Rate = PCLK/(2(USARTn_BDR+1))

15.3.4 External Clock (SCKn)

External clocking is used in the synchronous or SPI slave mode of operation.

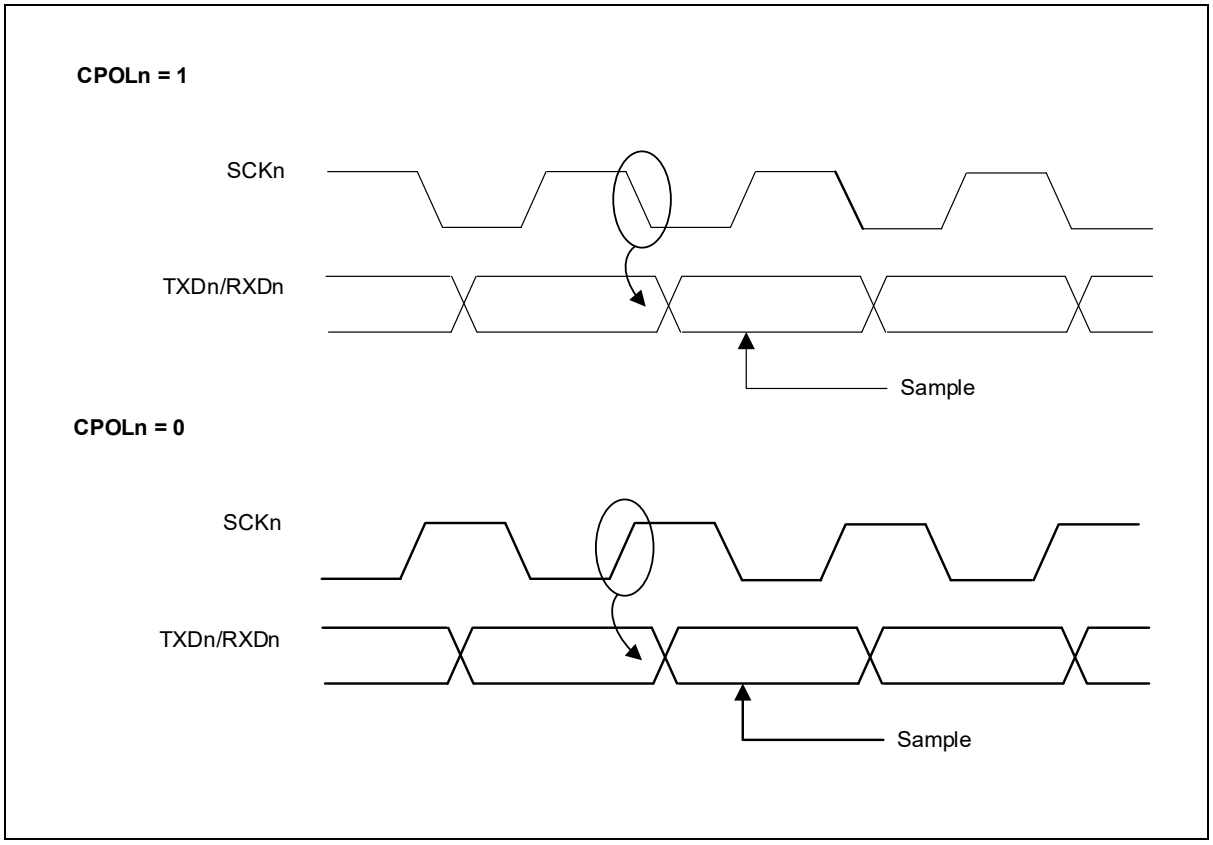
External clock input from the SCKn pin is sampled by synchronization logic to remove meta-stability. Output from the synchronization logic must be passed through an edge detector before it is used by the transmitter and receiver. This process introduces a two CPU clock period delay. The maximum frequency of the external SCKn pin is limited by 1 MHz.

15.3.5 Synchronous Mode Operation

External clocking is used in the synchronous or SPI slave mode of operation.

When synchronous or SPI mode is used, the SCKn pin will be used as either clock input (slave) or clock output (master). Data sampling and transmitter are issued on the different edge of SCKn clock respectively. For example, if data input on RXDn (MISOn in SPI mode) pin is sampled on the rising edge of SCKn clock, data output on TXDn (MOSIn in SPI mode) pin is altered on the falling edge.
CPOL[7] bit in USARTn_CR1 register selects which SCKn clock edge is used for data sampling and which is used for data change. As shown in Figure 126, when CPOL[7] is zero, the data will be changed at rising edge of SCKn and sampled at falling edge of SCKn.

Figure 126. Synchronous Mode SCKn Timing (USARTn, n = 10 and 11)



15.3.6 UART Data Format

A serial frame is defined as one character of data bits with synchronization bits (start and stop bits), and optionally a parity bit for error detection. USART supports all 30 combinations of the following as valid frame formats.

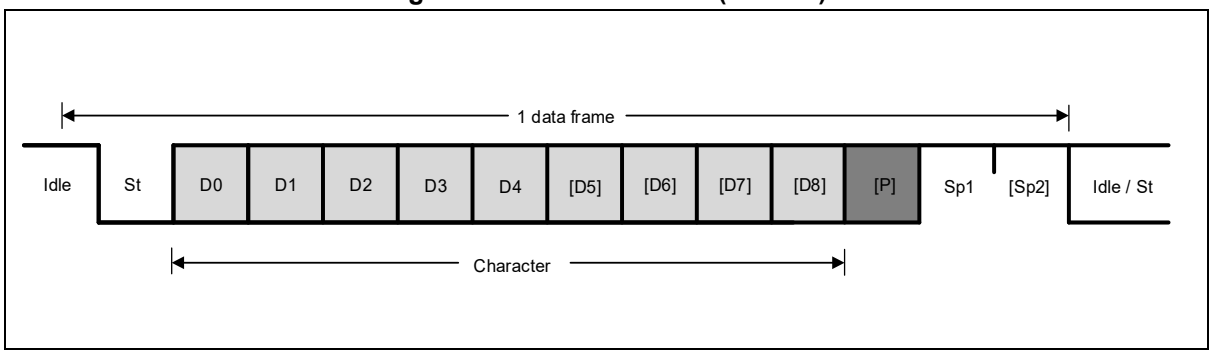
- 1 start bit
- 5, 6, 7, 8 or 9 data bits
- None, even or odd parity bit.
- 1 or 2 stop bits.

A frame starts with a start bit followed by the least significant data bit (LSB). Next data bits, up to nine, are succeeding, ending with the most significant bit (MSB). If the parity function is enabled, the parity bit is inserted between the last data bit and the stop bit.

A high-to-low transition on data pin is considered as a start bit. When a complete frame is transmitted, it can be directly followed by a new frame, or the communication line can be set to an idle state. The idle means high state of data pin.

Figure 127. Frame Format (USART) shows possible combinations of the frame formats. Bits inside brackets are optional.

Figure 127. Frame Format (USART)



Single data frame consists of the following bits:

- Idle: No signal on communication line (TXDn/RXDn)
- St: Start bit (Low)
- Dm: Data bits (0 to 8)
- Parity bit: Event parity, odd parity, no parity
- Stop bit(s): 1 bit or 2 bits

Frame format of UART is set by configuring DLEN[11:9], PE[13:12] bits in USARTn_CR1 register and SB[2] bit in USARTn_CR2 register. Transmitter and receiver use the same figures (n = 10 and 11).

15.3.7 UART Parity Bit

Parity bit is calculated by doing an exclusive-OR of all data bits. If odd-parity is used, the result of the exclusive-OR is inverted. Parity bit is located between the MSB and first stop bit of a serial frame.

- Peven = Dm-1 ^ … ^ D3 ^ D2 ^ D1 ^ D0 ^ 0
- Podd = Dm-1 ^ … ^ D3 ^ D2 ^ D1 ^ D0 ^ 0
- Peven: Parity bit using even parity
- Podd: Parity bit using odd parity
- Dm: Data bit n of the character

15.3.8 UART Transmitter

UART transmitter is enabled by configuring TXE[1] bit in USARTn_CR1 register. When the transmitter is enabled, TXDn pin should be set to TXDn function for the serial output pin of UART by the Pn_MR1/Pn_MR2. Baud rate, operation mode and frame format must be set up once before starting any transmission.

In synchronous operation mode, SCKn pin is used as a transmission clock, so it should be selected to function SCKn by Pn_MR1/Pn_MR2 (n = 10 and 11).

15.3.8.1 UART Sending TX Data

A data transmission is initiated by loading the transmit buffer (USARTn_DR register I/O location) with the data to be transmitted. The data written in transmit buffer is moved to the shift register when the shift register is ready to send a new frame. The shift register is loaded with the new data if it is in idle state or immediately after the last stop bit of the previous frame is transmitted.

When the shift register is loaded with new data, it will transfer one complete frame according to the settings of control registers. If the 9-bit characters are used in asynchronous or synchronous operation mode, the ninth bit must be written to the TX8[1] bit in USARTn_CR2 register before it is loaded to the transmit buffer USARTn_DR register (n = 10 and 11).

15.3.8.2 UART Transmitter Flag and Interrupt

The USART transmitter has two flags which indicate its state. One is USART data register empty flag (DRE[7]) and the other is transmit completion flag (TXC[6]). Both flags can be interrupt sources.

DRE[7] flag indicates whether the transmit buffer is ready to receive new data. This bit is set when the transmit buffer is empty and cleared when the transmit buffer contains data to be transmitted but has not yet been moved into the shift register. And this flag can be cleared by writing ‘0’ to this bit position. Writing ‘1’ to this bit position is prevented.

When the data register empty interrupt enable (DRIE[5]) bit in USARTn_CR1 register is set and the global interrupt is enabled, USARTn_ST status register empty interrupt is generated while DRE[7] flag is set.

The transmit complete (TXC[6]) flag bit is set when the entire frame in the transmit shift register has been shifted out and there is no more data in the transmit buffer. The TXC[6] flag is automatically cleared when the transmit complete interrupt serve routine is executed, or it can be cleared by writing ‘0’ to TXC[6] bit in USARTn_ST register.

When the transmit complete interrupt enable (TXCIE[4]) bit in USARTn_CR1 register is set and the global interrupt is enabled, UART transmit complete interrupt is generated while TXC[6] flag in USARTn_ST register is set (n = 10 and 11).

15.3.8.3 UART Parity Generator

The parity generator calculates the parity bit for the serial frame data to be sent. When parity bit is enabled by setting PG[13:12] bit value to 10b (even parity) or 11b (odd parity) in USARTn_CR1 register, the transmitter control logic inserts the parity bit between the MSB and the first stop bit of the frame to be sent.

15.3.8.4 UART Disabling Transmitter

Disabling the transmitter by clearing the TXE[1] bit in USARTn_CR1 will not become effective until ongoing transmission is completed. When the transmitter is disabled, the TXDn pin can be used as a normal general purpose I/O (n = 10 and 11).

15.3.9 UART Receiver

USART receiver is enabled by setting the RXE[0] bit in the USARTn_CR1 register. When the receiver is enabled, the RXDn pin should be set to RXDn function for the serial input pin of UART by Pn_MR1/Pn_MR2. The baud rate, mode of operation and frame format must be set before serial reception. In synchronous or SPI operation mode the SCKn pin is used as transfer clock input, so it should be selected to do SCKn function by Pn_MR1/Pn_MR2. In SPI operation mode the SSn input pin in slave mode can be configured as SSn output pin in master mode. This can be done by setting SSEN[4] bit in USARTn_CR2 register (n = 10 and 11).

15.3.9.1 UART Receiving RX Data

When UART is in synchronous or asynchronous operation mode, the receiver starts data reception when it detects a valid start bit (LOW) on RXDn pin. Each bit after start bit is sampled at pre-defined baud rate (asynchronous) or sampling edge of SCKn (synchronous) and shifted into the receive shift register until the first stop bit of a frame is received. Even if there's the second stop bit in the frame, the second stop bit is ignored by the receiver. That is, receiving the first stop bit means that a complete serial frame is presented in the receiver shift register and contents of the shift register are to be moved into the receive buffer. The receive buffer is read by reading the USARTn_DR register.

If 9-bit characters are used (DLEN[2:0] = "111"), the ninth bit is stored in the RX8[0] bit position in the USARTn_CR2 register. The ninth bit must be read from the RX8[0] bit before reading the low 8 bits from the USARTn_TDR register. Likewise, the error flags PE[0], FE[1], DOR[2] bit must be read before reading the data from USARTn_DR register. It's because the error flags are stored in the same FIFO position of the receive buffer (n = 10 and 11).

15.3.9.2 UART Receiver Flag and Interrupt

The UART receiver has one flag that indicates the receiver state.

A receive complete (RXC[5]) flag indicates whether there is unread data in the receive buffer. This flag is set when there is unread data in the receive buffer and cleared when the receive buffer is empty. If the receiver is disabled (RXE[0] = 0), the receiver buffer is flushed and the RXC[5] flag is cleared.

When a receive complete interrupt enable (RXCIE[3]) bit in the USARTn_CR1 register is set and global interrupt is enabled, the UART receiver complete interrupt is generated while RXC[5] flag is set.

The UART receiver has three error flags which are frame error (FE[2]), data overrun (DOR[2]) and parity error (PE[0]). These error flags can be read from the USARTn_ST register. As received data is stored in the 2-level receive buffer, these error flags are also stored in the same position of receive buffer. So, before reading received data from USARTn_DR register, read the USARTn_ST register first which contains error flags.

The frame error (FE[1]) flag indicates the state of the first stop bit. The FE flag is ‘0’ when the stop bit was correctly detected as ‘1’, and the FE flag is ‘1’ when the stop bit was incorrect, i.e. detected as ‘0’. This flag can be used for detecting out-of-sync conditions between data frames.

The data overrun (DOR[2]) flag indicates data loss due to a receive buffer full condition. DOR[2] occurs when the receive buffer is full, and another new data is presented in the receive shift register which is to be stored into the receive buffer. After the DOR[2] flag is set, all the incoming data is lost. To avoid data loss or clear this flag, read the receive buffer.

The parity error (PE[0]) flag indicates that the frame in the receive buffer had a parity error when received. If parity check function is not enabled (PG[13:12] = 0 in USARTn_CR1 register), the PE[0] bit in USARTn_ST register is always read ‘0’ (n = 10 and 11).

15.3.9.3 UART Parity Checker

if parity bit is enabled (PG[13:12]=1 in USARTn_CR1 register), the Parity Checker calculates the parity of the data bits in incoming frame and compares the result with the parity bit from the received serial frame (n = 10 and 11).

15.3.9.4 UART Disabling Receiver

In contrast to transmitter, disabling the Receiver by clearing RXE[0] bit makes the Receiver inactive immediately.

When the receiver is disabled, the receiver flushes the receive buffer, the remaining data in the buffer is all reset, and the RXDn pin can be used as a normal general purpose I/O (n = 10 and 11).

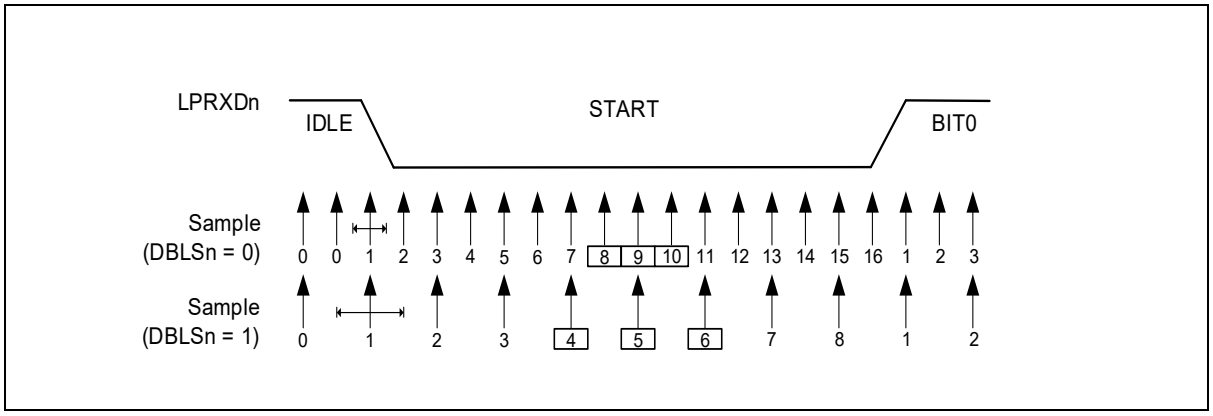
15.3.9.5 Asynchronous Data Reception

To receive an asynchronous data frame, the USART includes a clock and data recovery unit. The clock recovery logic is used for synchronizing the internally generated baud rate clock to the incoming asynchronous serial frame on the RXDn pin.

The data recovery logic does sample and low pass filtering the incoming bits and removing the noise of RXDn pin.

The next figure illustrates the sampling process of the start bit of an incoming frame. The sampling rate is 16 times the baud rate in normal mode and 8 times the baud rate for double speed mode (DBLS[8] = 1 in USARTn_CR2 register). The horizontal arrows show the synchronization variation due to the asynchronous sampling process. Note that larger time variation is shown when using the double speed mode (n = 10 and 11).

Figure 128. Asynchronous Start Bit Sampling (n = 10 and 11)

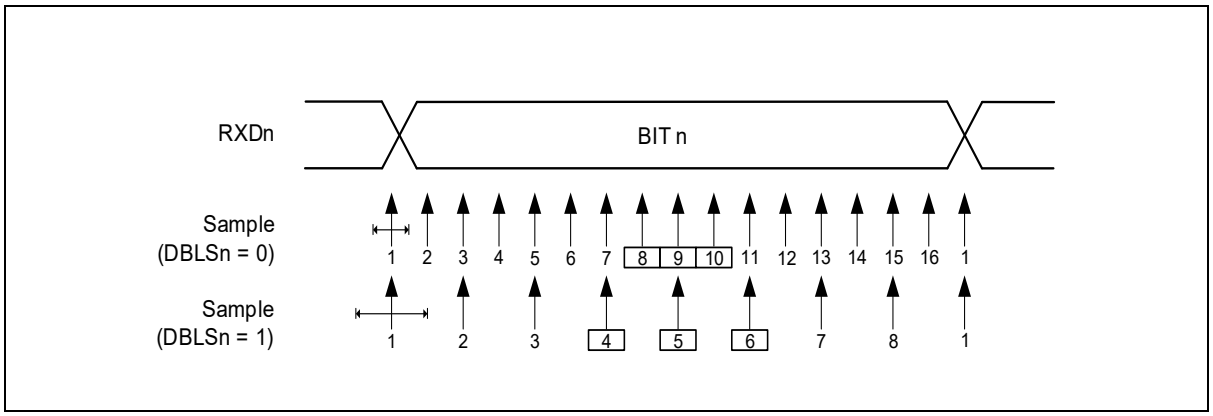


When the receiver is enabled (RXE[0] = 1), the clock recovery logic tries to find a high-to-low transition on the RXDn line, the start bit condition. After detecting high to low transition on RXDn line, the clock recovery logic uses samples 8, 9 and 10 for normal mode to decide if a valid start bit is received. If more than 2 samples have a logical low level, it is considered that a valid start bit is detected, and the internally generated clock is synchronized to the incoming data frame. And the data recovery can begin. The synchronization process is repeated for each start bit.

As described above, when the receiver clock is synchronized to the start bit, the data recovery can begin. Data recovery process is almost the same as the clock recovery process. The data recovery logic samples 16 times for each incoming bits for normal mode and 8 times for double speed mode, and uses sample 4, 5 and 6 to decide data value. If more than 2 samples have low levels, the received bit is considered to a logic ‘0’ and if more than two samples have high levels, the received bit is considered to a logic ‘1’.

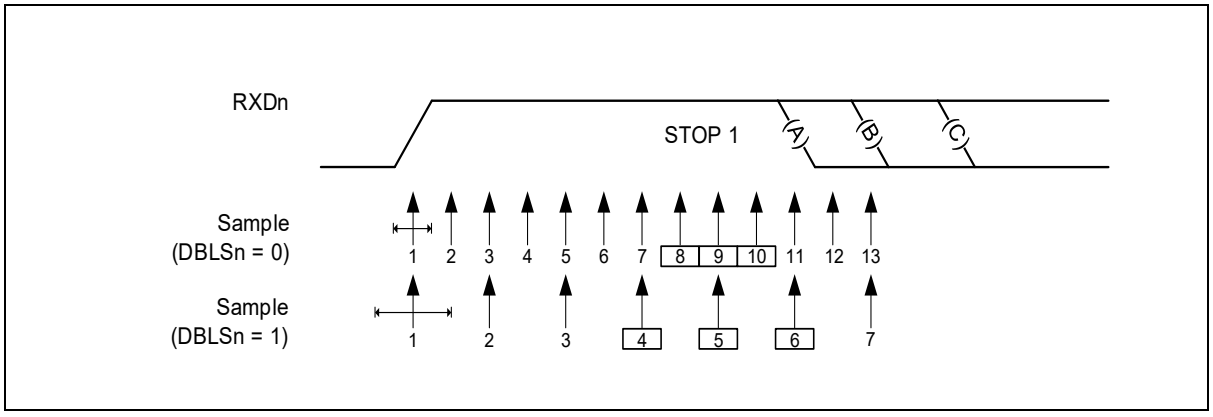
The data recovery process is then repeated until a complete frame is received including the first stop bit. The decided bit value is stored in the receive shift register in order. Note that the Receiver only uses the first stop bit of a frame. Internally, after receiving the first stop bit, the Receiver is in an idle state and waiting to find start bit.

Figure 129. Asynchronous Data and Parity Bit Sampling (n = 10 and 11)



The process for detecting stop bit is same as clock and data recovery process. That is, if two or more samples of three center values have high level, correct stop bit is detected, else a frame error (FE[1]) flag is set. After deciding whether the first stop bit is valid or not, the Receiver goes to idle state and monitors the RXDn line to check if a valid high to low transition is detected (start bit detection). (n = 10 and 11).

Figure 130. Stop Bit Sampling and Next Start Bit Sampling (n = 10 and 11)



15.3.10 UART Low-Power Mode

Table 111. Effect of Low-Power Mode on UART

Mode	Description
SLEEP	No effect
DEEP-SLEEP	Only UART mode operates as a wakeup source.

15.3.10.1 Description of Steps of Low-Power Mode

1. To enable the signal to be used as a wake-up source, set the USARTnWUE in the SCU_WUER register to ‘1’ (n = 10 and 11). Before setting the USARTnWUE bits, set UART mode first.
2. Enter low power mode (DEEP-SLEEP mode).
3. When the RXn pin goes low, the microcontroller wakes up from DEEP-SLEEP mode. The WAKE bit in the USARTn_ST register is set to ‘1’ and must be cleared after the wake up.
4. Writing a ‘1’ to the USARTnWU in the SCU_WUSR register clears the triggered flags.
5. Discard the USARTn_DR value received immediately after the wake up and use the next incoming data as a valid value.

15.3.11 SPI Mode

The USART can be set to operate in industrial standard SPI compliant mode.

The SPI mode features the following:

- Full Duplex, Three-wire synchronous data transfer.
- Master and Slave Operation
- Supports all four SPI modes of operation (mode 0, and 1).
- Selectable LSB first or MSB first data transfer.
- Double buffered transmit and receive.
- Programmable transmit bit rate.

When the SPI mode is enabled by configuring MS[15:14] as ”11”, the slave select (SSn) pin becomes active LOW input in slave mode operation or can be output in master mode operation if USARTn_CR2<SSEN[4]> bit is set to ‘0’. so, it should be selected to do MISOn, MOSIn function by Pn_MR1/Pn_MR2. (n = 10 and 11).

15.3.11.1 SPI Clock Formats and Timing

To accommodate a wide variety if synchronous serial peripherals from different manufacturers, the USART has a clock polarity bit (CPOL[7]) and a clock phase control bit (CPHA[6]) to select one of four clock formats for data transfers. CPOL[7] selectively insert an inverter in series with the clock. CPHA[6] chooses between two different clock phase relationships between the clock and data. Note that CPHA[6] and CPOL[7] bits in USARTn_CR1 register have different meanings according to the USARTn_CR1<MS[15:14]> bits which decides the operating mode of USART(MS[15:14] = 01).

Table 112 shows four combinations of CPOL and CPHA for SPI modes 0, 1, 2 and 3 (n = 10 and 11).

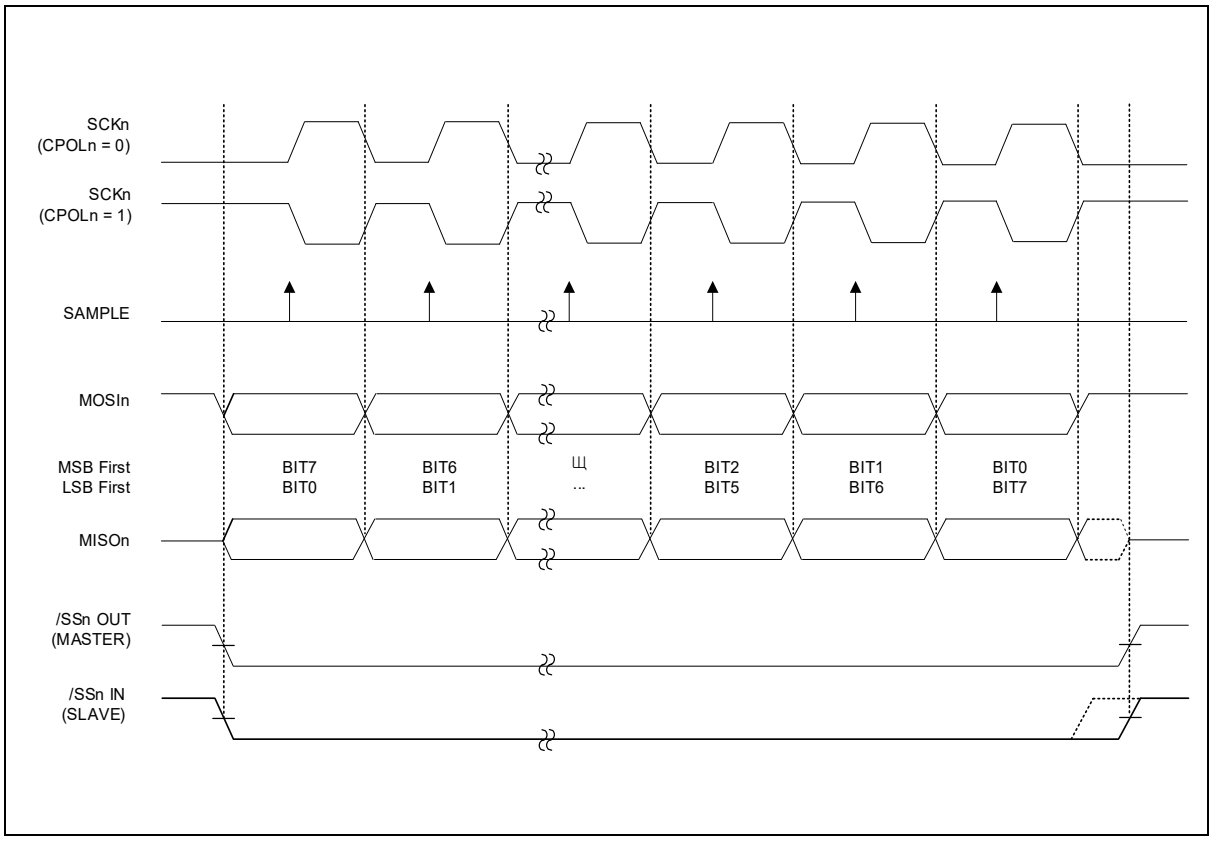
Table 112. CPOL Functionality (n = 10 and 11)

SPIn Mode	CPOLn	CPHAn	Leading Edge	Trailing Edge
0	0	0	Sample (Rising)	Setup (Falling)
1	0	1	Setup (Rising)	Sample (Falling)
2	1	0	Sample (Falling)	Setup (Rising)
3	1	1	Setup (Falling)	Sample (Rising)

When CPHA[6]=0, the slave begins to drive its MISOn output with the first data bit value when SSn goes to active low. The first SCKn edge causes both the master and the slave to sample the data bit value on their MISOn and MOSIn inputs, respectively.

At the second SCKn edge, the USART shifts the second data bit value out to the MOSIn and MISOn outputs of the master and slave, respectively. Unlike the case of CPHA[6] = 1, when CPHA[6] = 0, the slave's SSn input must go to its inactive high level between transfers. This is because the slave can prepare the first data bit when it detects falling edge of SSn input (n = 10 and 11).

Figure 131. USART SPIn Clock Formats when CPHAn = 0 (n = 10 and 11)

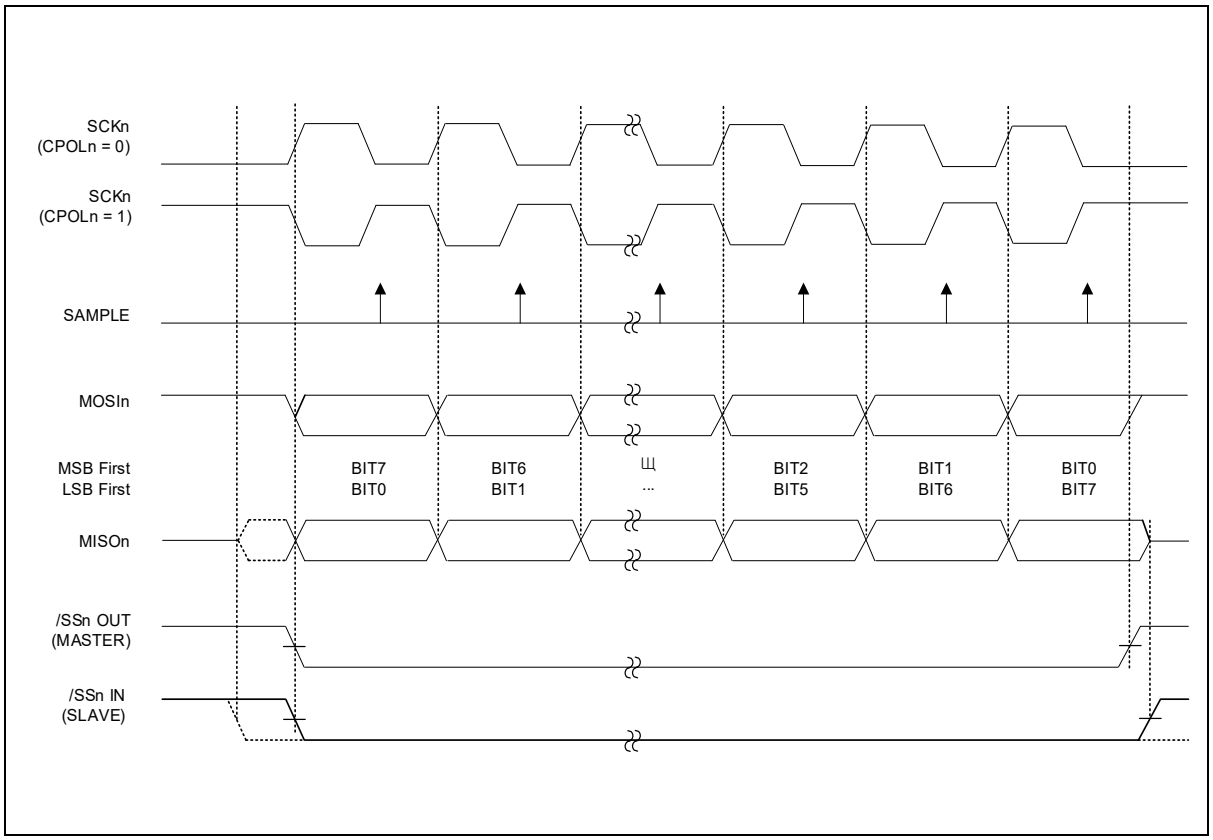


When CPHA[6] = 1, the slave begins to drive its MISOn output when SSn goes active low, but the data is not defined until the first SCKn edge. The first SCKn edge shifts the first bit of data from the shifter onto the MOSIn output of the master and the MISOn output of the slave.

The next SCKn edge causes both the master and slave to sample the data bit value on their MISOn and MOSIn inputs, respectively. At the third SCKn edge, the USART shifts the second data bit value out to the MOSIn and MISOn output of the master and slave respectively. When CPHA[6] = 1, the slave's SSn input is not required to go to its inactive high level between transfers.

Because the SPIn logic reuses the USART resources, SPIn mode of operation is similar to that of synchronous or asynchronous operation. SPIn transfer is initiated by checking for the USART Data Register Empty flag (DRE[7] = 1) in USARTn_ST register and then writing a byte of data to the USARTn_DR register. In master mode of operation, even if transmission is not enabled (TXE[0] = 0), writing data to the USARTn_DR register is necessary because the clock SCKn is generated from transmitter block.

Figure 132. USART SPIn Clock Formats when CPHAn = 1 (n = 10 and 11)



15.3.12 Baud Rate Settings (Example)

Table 113. Baud Rate Settings Example

Baud Rate	fOSC=1.00 MHz				fOSC=1.8432 MHz				fOSC=2.00 MHz			
	DBLSn=0		DBLSn=1		DBLSn=0		DBLSn=1		DBLSn=0		DBLSn=1	
	UBAUD	ERROR	UBAUD	ERROR	UBAUD	ERROR	UBAUD	ERROR	UBAUD	ERROR	UBAUD	ERROR
2400	25	0.2%	51	0.2%	47	0.0%	95	0.0%	51	0.2%	103	0.2%
4800	12	0.2%	25	0.2%	23	0.0%	47	0.0%	25	0.2%	51	0.2%
9600	6	-7.0%	12	0.2%	11	0.0%	23	0.0%	12	0.2%	25	0.2%
14.4K	3	8.5%	8	-3.5%	7	0.0%	15	0.0%	8	-3.5%	16	2.1%
19.2K	2	8.5%	6	-7.0%	5	0.0%	11	0.0%	6	-7.0%	12	0.2%
28.8K	1	8.5%	3	8.5%	3	0.0%	7	0.0%	3	8.5%	8	-3.5%
38.4K	1	-18.6%	2	8.5%	2	0.0%	5	0.0%	2	8.5%	6	-7.0%
57.6K	-	-	1	8.5%	1	-25.0%	3	0.0%	1	8.5%	3	8.5%
76.8K	-	-	1	-18.6%	1	0.0%	2	0.0%	1	-18.6%	2	8.5%
115.2K	-	-	-	-	-	-	1	0.0%	-	-	1	8.5%
230.4K	-	-	-	-	-	-	-	-	-	-	-	-

Baud Rate	fOSC=3.6864 MHz				fOSC=4.00 MHz				fOSC=7.3728 MHz			
	DBLSn=0		DBLSn=1		DBLSn=0		DBLSn=1		DBLSn=0		DBLSn=1	
	UBAUD	ERROR	UBAUD	ERROR	UBAUD	ERROR	UBAUD	ERROR	UBAUD	ERROR	UBAUD	ERROR
2400	95	0.0%	191	0.0%	103	0.2%	207	0.2%	191	0.0%	-	-
4800	47	0.0%	95	0.0%	51	0.2%	103	0.2%	95	0.0%	191	0.0%
9600	23	0.0%	47	0.0%	25	0.2%	51	0.2%	47	0.0%	95	0.0%
14.4K	15	0.0%	31	0.0%	16	2.1%	34	-0.8%	31	0.0%	63	0.0%
19.2K	11	0.0%	23	0.0%	12	0.2%	25	0.2%	23	0.0%	47	0.0%
28.8K	7	0.0%	15	0.0%	8	-3.5%	16	2.1%	15	0.0%	31	0.0%
38.4K	5	0.0%	11	0.0%	6	-7.0%	12	0.2%	11	0.0%	23	0.0%
57.6K	3	0.0%	7	0.0%	3	8.5%	8	-3.5%	7	0.0%	15	0.0%
76.8K	2	0.0%	5	0.0%	2	8.5%	6	-7.0%	5	0.0%	11	0.0%
115.2K	1	0.0%	3	0.0%	1	8.5%	3	8.5%	3	0.0%	7	0.0%
230.4K	-	-	1	0.0%	-	-	1	8.5%	1	0.0%	3	0.0%
250K	-	-	1	-7.8%	-	-	1	0.0%	1	-7.8%	3	-7.8%
0.5M	-	-	-	-	-	-	-	-	-	-	1	-7.8%

Baud Rate	fOSC=8.00 MHz				fOSC=11.0592 MHz				fOSC=14.7456 MHz			
	DBLSn=0		DBLSn=1		DBLSn=0		DBLSn=1		DBLSn=0		DBLSn=1	
	UBAUD	ERROR	UBAUD	ERROR	UBAUD	ERROR	UBAUD	ERROR	UBAUD	ERROR	UBAUD	ERROR
2400	207	0.2%	-	-	-	-	-	-	-	-	-	-
4800	103	0.2%	207	0.2%	143	0.0%	-	-	191	0.0%	-	-
9600	51	0.2%	103	0.2%	71	0.0%	143	0.0%	95	0.0%	191	0.0%
14.4K	34	-0.8%	68	0.6%	47	0.0%	95	0.0%	63	0.0%	127	0.0%
19.2K	25	0.2%	51	0.2%	35	0.0%	71	0.0%	47	0.0%	95	0.0%
28.8K	16	2.1%	34	-0.8%	23	0.0%	47	0.0%	31	0.0%	63	0.0%
38.4K	12	0.2%	25	0.2%	17	0.0%	35	0.0%	23	0.0%	47	0.0%
57.6K	8	-3.5%	16	2.1%	11	0.0%	23	0.0%	15	0.0%	31	0.0%
76.8K	6	-7.0%	12	0.2%	8	0.0%	17	0.0%	11	0.0%	23	0.0%
115.2K	3	8.5%	8	-3.5%	5	0.0%	11	0.0%	7	0.0%	15	0.0%
230.4K	1	8.5%	3	8.5%	2	0.0%	5	0.0%	3	0.0%	7	0.0%
250K	1	0.0%	3	0.0%	2	-7.8%	5	-7.8%	3	-7.8%	6	5.3%
0.5M	-	-	1	0.0%	-	-	2	-7.8%	1	-7.8%	3	-7.8%
1M	-	-	-	-	-	-	-	-	-	-	1	-7.8%

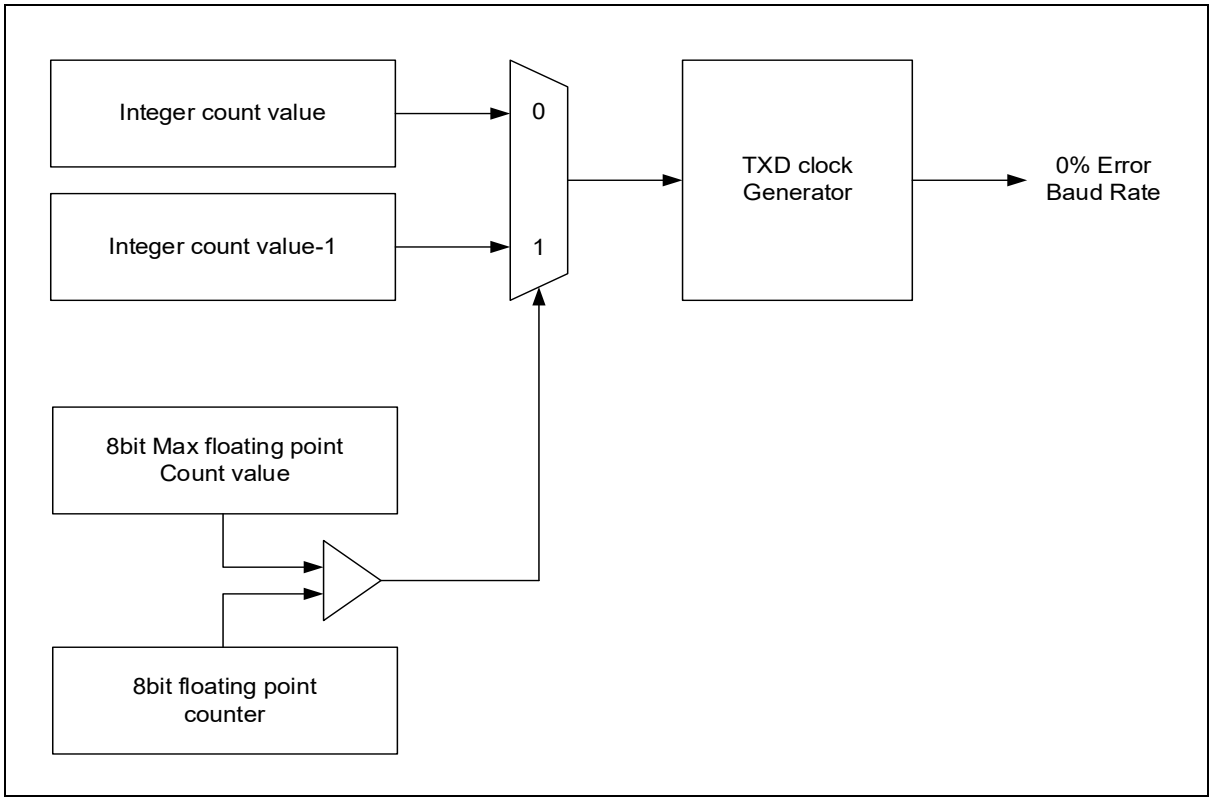
15.3.13 0% Error Baud Rate

This USART system supports the fraction counter logic for the 0% error of baud rate. By using the 8 bits floating point counter logic, the cumulative error to below the decimal point can be removed.

The fraction counter value is defined by baud rate error. In the baud rate formula, USARTn_BDR is presented the integer count value. For example, when the 57600 baud rate (fXIN = 16 MHz) is used, the calculated integer count value must be 17.36 (USARTn_BDR+1 = 16000000/(16×57600) = 17.36).

Here, the USARTn_BDR which can be set is the nearest big integer number 17. To realize 0% error of baud rate, fraction counter value must be 92 ((0.36) x 256 ≒ 92). In other words, 92 (decimal number) must be written to the USARTn_BFR register and 17 (decimal number) to the USART_BDR register.

Figure 133. 0% Error Baud Rate Diagram



15.3.14 Receive Time-Out (RTO)

This USART system supports the receive time-out (RTO) interrupt. The RTO counter uses the system clock and continues counting while the RXD input is not present. If the RTO counter matches USARTn_RTO, RTOF becomes '1'. RTOEN is set to '1' to enable this operation, and RTOEN is automatically set to 0 when an RTO match occurs. The RTO interrupt can be used with RTOIE.

15.4 USART Registers

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

Table 114. Base Addresses of USART

Name	Base Address
USART10	0x4000_8800
USART11	0x4000_8900

Table 115. UART Register Map

Name	Offset	Type	Description	Reset Value	Reference
USARTn_CR1	0x0000	RW	USART n control register 1	0x0000_0000	15.4.1
USARTn_CR2	0x0004	RW	USART n control register 2	0x0000_0000	15.4.2
USARTn_ST	0x000C	RW	USART n status register	0x0000_0080	15.4.3
USARTn_BDR	0x0010	RW	USART n baud rate generation register	0x0000_0FFF	15.4.4
USARTn_DR	0x0014	RO	USART n data register	0x0000_0000	15.4.5
USARTn_BFR	0x0018	RW	USART n baud rate fraction counter register	0x0000_0000	15.4.6
USARTn_RTO	0x001C	RW	USART n receive time out register	0x00FF_FFFF	15.4.7

NOTE:

1. n = 10 and 11.

15.4.1 USARTn_CR1: USART n Control Register 1

USARTn module should be configured properly before running.

USARTn_CR1 register is 32-bit size and accessible in 32/16/8-bit (n=10 and n=11)

USART0_CR1=0x4000_8800, USART1_CR1=0x4000_8900

31 30 29 28 27 26 25 24	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		MS[1:0]		PG[1:0]		DLEN[2:0]			ORD	CPOL	CPHA	DRIE	TXCIE	RXCIE	WAKEIE	TXE	RXE
-		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 14	MS[1:0]	USARTn operation mode selection bits.			
		00	Asynchronous mode (UART).		
		01	Synchronous mode.		
		10	Reserved.		
		11	SPI mode.		
13 12	PG[1:0]	Selects parity generation and check method (only UART mode)			
		00	No parity.		
		01	Reserved.		
		10	Even parity.		
		11	Odd parity.		
11 9	DLEN[2:0]	Selects the length of data bit in a frame when Asynchronous or Synchronous mode.			
		000	5 bits		
		001	6 bits		
		010	7 bits		
		011	8 bits		
		111	9 bits		
		others	Reserved.		
8	ORD	Selects the first data bit to be transmitted (only SPI mode)			
		0	LSB-first.		
		1	MSB-first.		
7	CPOL	Selects the clock polarity of ACK in synchronous or SPI mode.			
		0	TXD change @rising edge. RXD change @falling edge		
		1	TXD change @falling edge. RXD change @rising edge		
6	CPHA	The CPOLn and this bit determine if data are sampled on the leading or trailing edge of SCK (only SPI mode)			
		CPOLn	CPHAn	Leading edge	Trailing edge
		0	0	Sample (Rising)	Setup (Falling)
		0	1	Setup (Rising)	Sample (Falling)
		1	0	Sample (Falling)	Setup (Rising)
		1	1	Setup (Falling)	Sample (Rising)
5	DRIE	Transmit data register empty interrupt enable bit.			
		0	Disable the transmit data empty interrupt.		
		1	Enable the transmit data empty interrupt.		
4	TXCIE	Transmit complete interrupt enable bit.			
		0	Disable transmit complete interrupt.		
		1	Enable transmit complete interrupt.		

3	RXCIE	Receive complete interrupt enable bit.	
		0	Disable receive complete interrupt.
		1	Enable receive complete interrupt.
2	WAKEIE	Asynchronous wake-up interrupt enable bit in DEEP SLEEP mode. When device is in DEEP SLEEP mode, if RXDn goes to low level, an interrupt can be requested to wake-up system. (only UART mode). Must be set '1' in DEEP SLEEP mode, set to '0' if wake up from DEEP SLEEP mode.	
		0	Disable Asynchronous wake-up interrupt.
		1	Enable Asynchronous wake-up interrupt.
1	TXE	Enables the transmitter unit.	
		0	Transmitter is disabled.
		1	Transmitter is enabled.
0	RXE	Enables the receiver unit.	
		0	Receiver is disabled.
		1	Receiver is enabled.

NOTE:

1. The CPOLn and CPHAn bits should be changed during the TXEn and RXEn bits are both '0'.

15.4.2 USARTn_CR2: USART n Control Register 2

USARTn module should be configured properly before running.

USARTn_CR2 register is 32-bit size and accessible in 32/16/8-bit (n=10 and n=11)

USART0_CR2=0x4000_8804, USART1_CR2=0x4000_8904

31 30 29 28 27 26 25 24	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			DMATXIE	DMARXIE	RTOIE	RTOEN	BFREN	EN	DBLS	MASTER	LOOPS	DISCCK	SSEN	FXCH	SB	TX8	RX8
-			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

14	DMATXIE	DMA TX interrupt bit
	0	Disable DMA TX interrupt.
	1	Enable DMA TX interrupt.
13	DMARXIE	DMA RX interrupt bit
	0	Disable DMA RX interrupt.
	1	Enable DMA RX interrupt.
12	RTOIE	RTO interrupt bit.
	0	Disable RTO interrupt.
	1	Enable RTO interrupt.
11	RTOEN	Activate RTO block by supplying
	0	Disable RTO.
	1	Enable RTO.
10	BFREN	Activate baud rate fraction counter register.
	0	Disable fraction counter register.
	1	Enable fraction counter register.
9	EN	Activate USARTn block by supplying.
	0	Disable USARTn block.
	1	Enable USARTn block.
8	DBLS	Selects receiver sampling rate. (only UART mode)
	0	Normal Asynchronous operation.
	1	Double speed Asynchronous operation.
7	MASTER	Selects master or slave in SPIn or synchronous mode and controls the direction of SCKn pin.
	0	Slave operation (External clock for SCKn)
	1	Master operation (Internal clock for SCKn)
6	LOOPS	Control the loop back mode of USARTn for test mode.
	0	Normal operation.
	1	Loop back mode.
5	DISSCK	In synchronous mode operation, selects the waveform of SCKn output.
	0	SCKn is free-running while USARTn is enabled in synchronous master mode.
	1	SCKn active while any frame is on transferring.
4	SSEN	Enables the receiver unit.
	0	Receiver is disabled.
	1	Receiver is enabled.
3	FXCH	SPIn port function exchange control bit (only SPI mode).
	0	No effect.

		1 Exchange MOSIn and MISOn function.
2	SB	Selects the length of stop bit in Asynchronous or Synchronous mode
		0 1 stop bit.
		1 2 stop bit.
1	TX8	The 9th bit of data frame in Asynchronous or Synchronous mode of operation. Write this bit first before loading the USTnDR register.
		0 MSB (9th bit) to be transmitted is ‘0’.
		1 MSB (9th bit) to be transmitted is ‘1’.
0	RX8	The 9th bit of data frame in Asynchronous or Synchronous mode of operation. Read this bit first before reading the receiver buffer (only UART mode)
		0 MSB (9th bit) to be received is ‘0’.
		1 MSB (9th bit) to be received is ‘1’.

15.4.3 USARTn_ST: USART n Status Register

USARTn_ST register is 32-bit size and accessible in 32/16/8-bit (n=10 and n=11)

USART0_ST=0x4000_880C, USART1_ST=0x4000_890C

31 30 29 28 27 26 25 24	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			DMATXF	DMARXF	DRE	TXC	RXC	WAKE	RTOF	DOR	FE	PE
-			0	0	1	0	0	0	0	0	0	0
.			RW	RW	RW	RW	RW	RW	RW	RW	RW	RW

9	DMATXF[1]	DMA transmit operation complete flag (DMA to USART).
	0	DMA transmit operation is working or disabled.
	1	DMA transmit operation is done.
8	DMARXF	DMA receive operation complete flag (USART to DMA).
	0	DMA receive operation is working or disabled.
	1	DMA receive operation is done.
7	DRE	Transmit data register empty interrupt flag. The DRE flag indicates if the transmit data register (USARTn_DR) is ready to receive new data. If DRE is '1', the data register is empty and ready to be written.
	0	Transmit buffer is not empty.
	1	Transmit buffer is empty. This bit is cleared to '0' when write '1'
6	TXC	Transmit complete interrupt flag. This flag is set when the entire frame in the transmit shift register has been shifted out and there is no new data currently present in the transmit buffer.
	0	No request occurred.
	1	Transmit buffer is empty and the data in transmit shift register are shifted out completely. This bit is cleared to '0' when write '1'.
5	RXC	Receive complete interrupt flag. This flag is set when there are unread data in the receive buffer and cleared when all the data in the receive buffer are read.
	0	There is no data unread in the receive buffer.
	1	There are more than 1 data in the receive buffer.
4	WAKE	Asynchronous wake-up interrupt flag. This flag is set when the RXD pin is detected low while the CPU is in DEEP SLEEP mode. (only UART mode)
	0	No request occurred.
	1	Request occurred. This bit is cleared to '0' when write '1'.
3	RTOF	Receive time out interrupt flag. This bit is cleared to '0' when write '1'
	0	Receive time out is not generated.
	1	Receive time out is generated.
2	DOR	This bit is set if data over-run occurs. While this bit is set, the incoming data frame is ignored. This flag is valid until the receive buffer is read.
	0	No data overrun.
	1	Data overrun detected.
1	FE	This bit is set if the first stop bit of next character in the receive buffer is detected as '0'. This bit is valid until the receive buffer is read. (only UART mode)
	0	No frame error.
	1	Frame error detected.

0	PE	This bit is set if the next character in the receive buffer has a parity error while parity is checked. This bit is valid until the receive buffer is read. (only UART mode)	
		0	No parity error.
		1	Parity error detected.

NOTE:

1. When the USART is configured in DMA transmit mode, a 1-byte transmission completes normally, but the DMA Transfer Complete flag (DMATX) is not generated.

15.4.4 USARTn_BDR: USART n Baud Rate Generation Register

USART_BDR register is 32-bit size and accessible in 32/16/8-bit (n=10 and n=11)

USART0_BDR=0x4000_8810, USART1_BDR=0x4000_8910

31 30 29 28 27 26 25 24	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			BDATA[11:0]
-			1 1 1 1 1 1 1 1 1 1 1 1
-			RW RW RW RW RW RW RW RW RW RW RW RW

11 0	BDATA[11:0]	The value in this register is used to generate internal baud rate in UART mode or to generate SCK clock in SPI mode. To prevent malfunction, do not write '0' in UART mode and do not write '0' or '1' in synchronous or SPI mode.

15.4.5 USARTn_DR: USART n Data Register

USART_DR register is 32-bit size and accessible in 32/16/8-bit (n=10 and n=11)

USART0_DR=0x4000_8814, USART1_DR=0x4000_8914

31 30 29 28 27 26 25 24	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			DATA[7:0]
-			0 0 0 0 0 0 0 0
-			RW RW RW RW RW RW RW RW

7 0	DATA[7:0]	The USART transmit buffer and receive buffer share the same I/O address with this DATA register. The transmit data buffer is the destination for data written to the USARTn_DR register. Reading the USTnDR register returns the contents of the receive buffer. Write to this register only when the DRE flag is set. In SPI master mode, the SCK clock is generated when data are written to this register.

NOTE:

1. The byte won't be written when the block is disabled or both of TXEn and RXEn bits are '0'.

15.4.6 USARTn_BFR: USART n Baud Rate Fraction Count Register

USART_BFR register is 32-bit size and accessible in 32/16/8-bit (n=10 and n=11)

USART0_BFR=0x4000_8818, USART1_BFR=0x4000_8918

31 30 29 28 27 26 25 24	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			BFC[7:0]
-			0 0 0 0 0 0 0 0
.			RW RW RW RW RW RW RW RW

7 0	BFC[7:0]	USARTn baud rate fraction counter. 8-bit floating point counter.

15.4.7 USARTn_RTO: USART n RTO Register

USART_RTO register is 32-bit size and accessible in 32/16/8-bit (n=10 and n=11)

USART0_RTO=0x4000_881C, USART1_RTO=0x4000_891C

31 30 29 28 27 26 25 24	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	RTO[23:0]		
-	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

23 0	RTO[23:0]	USART receive time out register.

15.4.8 USART Register Map Summary

Table 116. USART 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	USARTn_CR1	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	MS[1:0]		PG[1:0]		DLEN[2:0]			ORD	CPOL	CPHA	DRIE	TXCIE	RXCIE	WAKEIE	TXE	RXE
	Reset value																	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
0x04	USARTn_CR2	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.		DMATXIE	DMARXIE	RTOIE	RTOEN	BFREN	EN	DBLS	MASTER	LOOPS	DISCCK	SSEN	FXCH	SB	TX8	RX8
	Reset value																		0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
0x0C	USARTn_ST	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.							DMATXF	DMARXF	DRE	TXC	RXC	WAKE	RTOF	DOR	FE	PE
	Reset value																							0	0	1	0	0	0	0	0	0	0
0x10	USARTn_BDR	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.					BDATA[11:0]											
	Reset value																					1	1	1	1	1	1	1	1	1	1	1	1
0x14	USARTn_DR	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.									DATA[7:0]							
	Reset value																									0	0	0	0	0	0	0	0
0x18	USARTn_BFR	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.									BFC[7:0]							
	Reset value																									0	0	0	0	0	0	0	0
0x1C	USARTn_RTO	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	RTO[23:0]																							
	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

16. LPUART (Low-Power Universal Asynchronous Receiver/Transmitter)

16.1 LPUART Introduction

The A34L716 series has a built-in two-channel Low Power UART module (Universal Asynchronous Receiver/Transmitter).

This Low Power UART (LPUART) supports asynchronous serial communication up to 9600 bps in DEEP SLEEP mode with 32.768 kHz sub-oscillator. It also supports 1-wire half-duplex communication.

16.2 LPUART Main Features

LPUART 0/1 of the A34L716 series features the following:

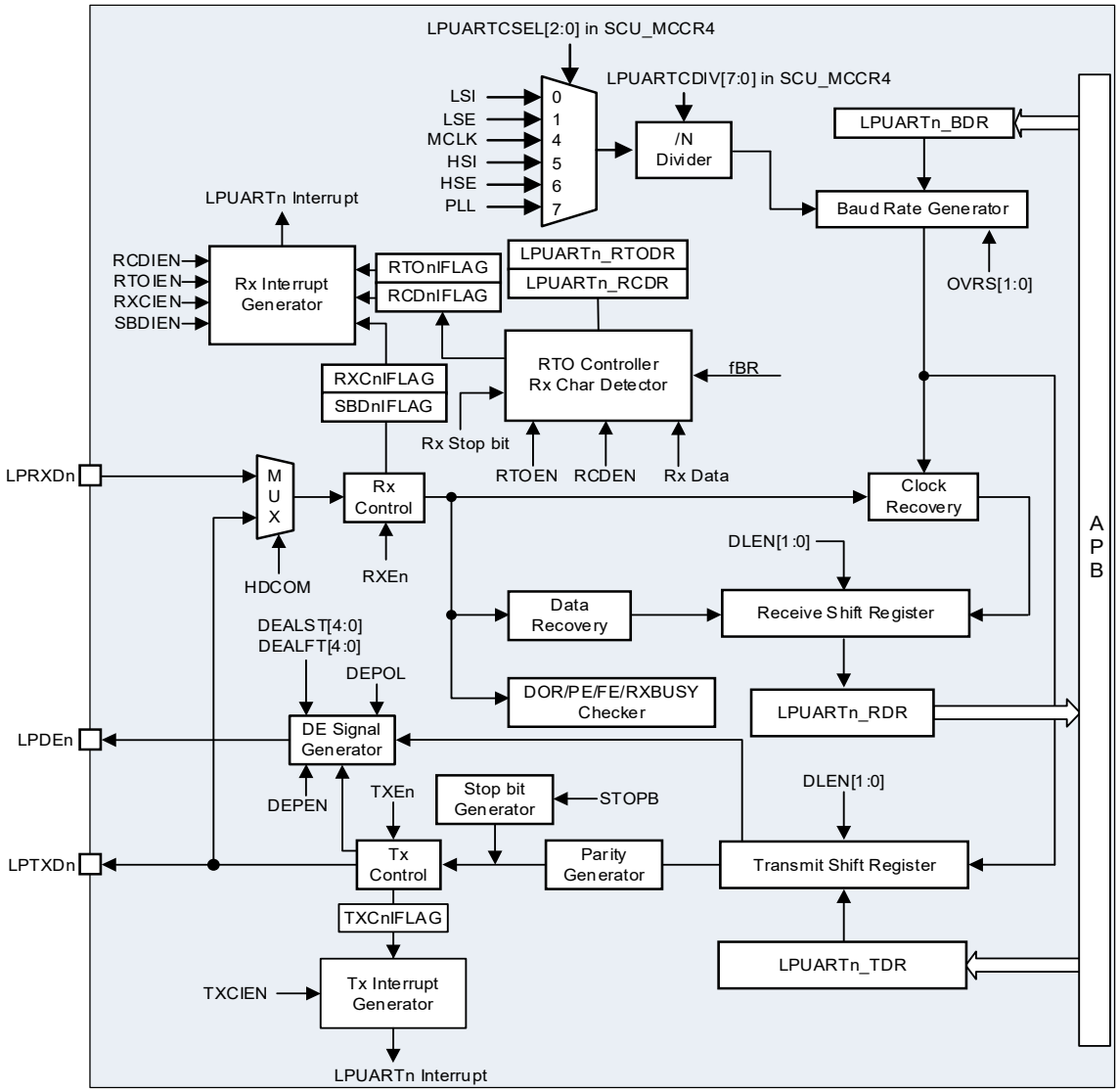
- Full-Duplex and Half-Duplex operations
- Baud rate generator
- Serial frames with 5,6,7, or 8 data bits and 1 or 2 stop bits supported.
- Odd or even parity generation, and parity check supported by hardware.
- Receive character detection and receive time-out function supported.
- Baud rate compensation function
- Up to 9600 bps with 32.768 kHz sub-oscillator supported.
- Data overrun detection
- Framing error detection
- Double-speed asynchronous communication mode

16.3 LPUART Functional Description

16.3.1 LPUART Block Diagram

Figure 134 shows a block diagram of the LPUART block.

Figure 134. LPUART Block Diagram (n = 0 and 1)



NOTES:

1. If LSE (32.768 kHz) is used for a clock of the LPUARTn, the maximum baud rate may be up to 9600 bps.
2. A clock of PCLK should be faster than or equal to a clock of the LPUARTn.
3. Data to be transmitted should be written to the LPUARTn_TDR register after checking if the TXCIFLAG bit is set to '1'.

16.3.2 LPUART Pins and Signals

Table 117 describes external pins assigned for LPUART.

Table 117. Pin Assignment of LPUART: External Pins (n = 0 and 1)

Pin Name	Type	Description	Supported packages		
			A34L716 (100-LQFP)	A34L716 (64-LQFP)	A34L716 (56-QFN)
LPTXD0	O	Low Power UART0 transmit output	O	O	O
LPRXD0	I	Low Power UART0 receive input	O	O	O
LPDE0	O	Low Power UART0 DE signal output	O	O	O
LPTXD1	O	Low Power UART1 transmit output	O	O	X
LPRXD1	I	Low Power UART1 receive input	O	O	O
LPDE1	O	Low Power UART1 DE signal output	O	O	O

16.3.3 LPUARTn Clock Generation

The clock generation logic generates clocks for the transmitter and the receiver. The LPUARTn baud rate generator supports three modes of clock operation, which are 16-oversampling mode, 8-oversampling mode, and only 1 sampling mode. Only one sampling mode can be used with LSE (32.768 kHz).

Table 118 shows equations for baud rate calculation (in bps).

Table 118. Equations for Calculating Baud Rate Register Settings

Oversampling	Equation for Calculating Baud Rate
16-oversampling mode (OVRS = 00)	Baud Rate = f_{LPUART}/(16 × (LPUARTn_BDR+1))
8-oversampling mode (OVRS = 01)	Baud Rate = f_{LPUART}/(8 × (LPUARTn_BDR+1))
Only 1 sampling mode (OVRS = 10)	Baud Rate = f_{LPUART}/(LPUARTn_BDR+1)

16.3.4 LPUARTn Baud Rate Compensation

The baud rate compensation is used to optimize the precision in each bit. There is a sign (BCMPS bit of LPUARTn_BCMP register) bit to define the positive or negative compensation in each bit. If the sign bit is '0', one clock of $f_{LPUARTn}$ will be appended to the compensated bit. If the sign bit is '1', one clock of $f_{LPUARTn}$ will be taken out from the compensated bit.

There are nine bits to define whether relative compensation is required for each bit. The bits are BCMP[7:0] for data and BCMP8 for parity.

For example, when the baud rate is $f_{LPUARTn}$ 32.768 kHz, No oversampling, Baud rate 9600 bps:

$$\frac{fLPUARTn\ clock}{1 \times\ BaudRate} = \frac{32768}{1 \times\ 9600} = 3.413, \qquad Divisor = 3, Float = 0.413$$

$$LPUARTn\_BDR = \ 3 - 1 = 2$$

32.768 kHz/(1 × 9600) = 3.413, LPUARTn_BDR = 3 - 1 = 2, and "Baud rate clock"/bit = 3 × 1

"Clock error"/bit: 3.413 × 1(OVRS) – 3(Divisor) × 1(OVRS) = 0.413 clock → "1 clock compensation"/bit if a BCMPx bit is '1'.

The result is that the sign bit, BCMPS, is '0' for positive compensation and the baud rate compensation bits, BCMP[8:0], are "010100101". (CEPB: "clock error"/bit)

Table 119. Baud Rate Compensation Example 1

Rx/Tx bit	BCMPx Bit	Clock Error	Compensation Bit	Final Clock Error
Start bit	–	-0.413 (CEPB)	x	-0.413
D0	bit 0	-0.827 (CEPB+ before compensation)	1	0.173
D1	bit 1	-0.240 (CEPB+ before compensation)	0	-0.240
D2	bit 2	-0.653 (CEPB+ before compensation)	1	0.347
D3	bit 3	-0.067 (CEPB+ before compensation)	0	-0.067
D4	bit 4	-0.480 (CEPB+ before compensation)	0	-0.480
D5	bit 5	-0.893 (CEPB+ before compensation)	1	0.107
D6	bit 6	-0.307 (CEPB+ before compensation)	0	-0.307
D7	bit 7	-0.720 (CEPB+ before compensation)	1	0.280
Parity bit	bit 8	-0.133 (CEPB+ before compensation)	0	-0.133

16.3.5 LPUARTn Interface Data Format

The parity bit is calculated by doing an exclusive-OR of all data bits. If odd-parity is used, the result of the exclusive-OR is inverted. The parity bit is located between the last data bit and first stop bit of a serial frame.

- Peven = Dm-1 ^ … ^ D3 ^ D2 ^ D1 ^ D0 ^ 0
- Podd = Dm-1 ^ … ^ D3 ^ D2 ^ D1 ^ D0 ^ 1
- Peven: Parity bit using even parity
- Podd: Parity bit using odd parity
- Dm: Data bit n of the character

16.3.6 LPUARTn Transmitter

The LPUARTn transmitter is enabled by setting the TXE bit in LPUARTn_CR1 register. When the transmitter is enabled, the LPTXDn pin should be set to LPTXDn function for the serial output pin by the GPIO registers. The configuration for the LPTXDn pin is set in the PCU_MR1/PCU_MR2 registers.

Baud rate, operation mode and frame format must be set up before doing any transmission.

16.3.7 LPUARTn Sending TX Data

A data transmission is initiated by loading data to the transmit data register (LPUARTn_TDR register). The data to be written in transmit data register is moved to the shift register when the shift register is ready to send a new frame.

The shift register is loaded with the new data if it is in idle state or immediately after the last stop bit of the previous frame is transmitted. When the shift register is loaded to the new data, it will transfer one complete frame according to the settings of control registers. (n = 0 and 1)

16.3.8 LPUARTn Parity Generator

The parity generator calculates parity bit for the serial frame data to be sent. When the parity bit is enabled (PENn = 1), the transmitter control logic inserts the parity bit between the last data bit and the first stop bit of the frame to be sent.

16.3.9 LPUARTn Receiver

The LPUARTn receiver is enabled by setting the RXE bit in the LPUARTn_CR1 register. When the receiver is enabled, the LPRXDn pin should be set to LPRXDn function for the serial input pin by the GPIO registers. The configuration for the LPRXDn pin is set in the Pn_MR1/Pn_MR2 registers.

Baud rate, operation mode, and frame format must be set before the serial reception.

16.3.10 LPUARTn Receiving RX Data

The receiver starts data reception when it detects a valid start bit (LOW) on LPRXDn pin. Each bit after start bit is sampled at predefined baud rate and shifted into the receive shift register until the first stop bit of a frame is received.

Even if there is the second stop bit in the frame, the second stop bit is ignored by the receiver. That is, receiving the first stop bit means that a complete serial frame is presented in the receiver shift register and contents of the shift register are to be moved into the receive data register.

16.3.11 LPUARTn Parity Checker

If the parity bit is enabled (PEN = 1), the Parity Checker calculates the parity of the data bits in incoming frame and compares the result with the parity bit from the received serial frame.

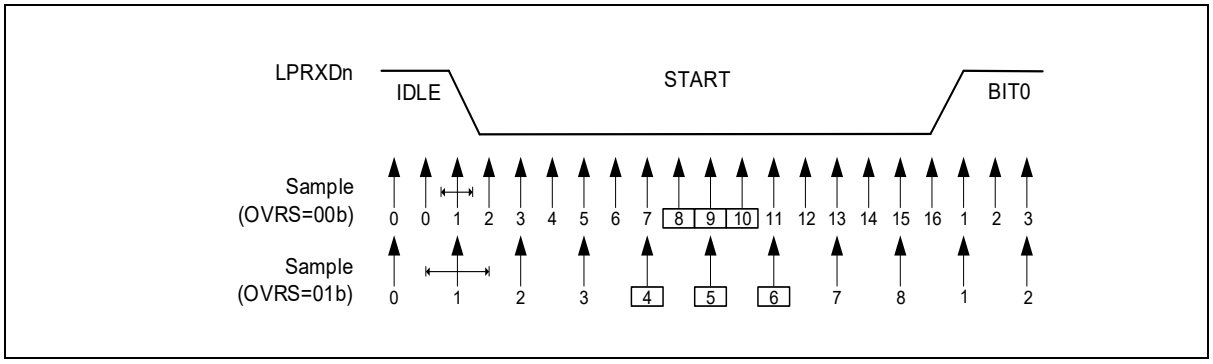
16.3.12 LPUARTn Data Reception

To receive data frame, the receiver includes a clock and data recovery unit. The clock recovery logic is used for synchronizing the internally generated baud rate clock to the incoming asynchronous serial frame on the LPRXDn pin.

The data recovery logic samples and filters the incoming bits with a low pass filter and removes the noise of the receive pin.

Figure 135 illustrates the sampling process of a start bit of an incoming frame. The sampling rate is 16 times the baud rate in 16-oversampling mode and 8 times the baud rate for 8-oversampling mode. The horizontal arrows show the synchronization variation due to the asynchronous sampling process. Note that larger time variation is seen when using 8-oversampling mode.

Figure 135. Start Bit Sampling



When the receiver is enabled (RXEn=1), the clock recovery logic tries to find a high-to-low transition on the LPRXDn line, the start bit condition. After detecting high to low transition on the line, the clock recovery logic uses samples 8, 9 and 10 for 16-oversampling mode to detect whether valid start bit is received.

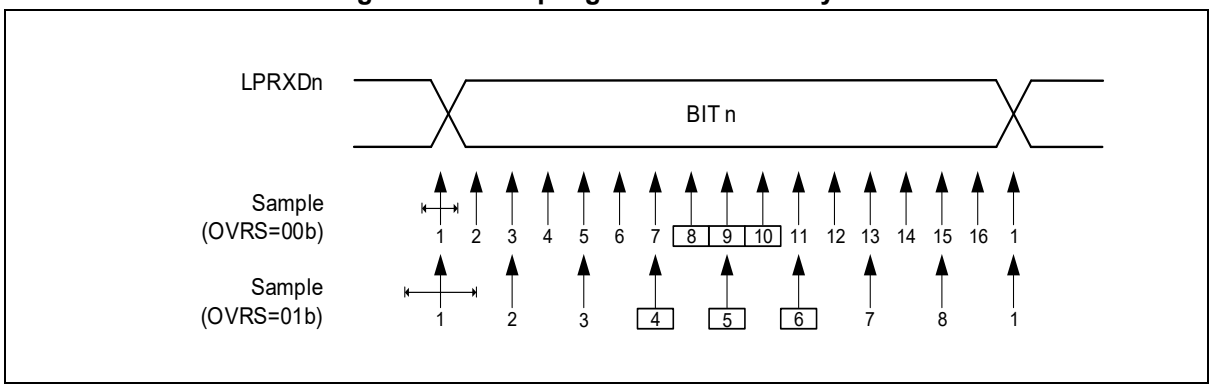
If more than two samples have a logical low level, it is considered that a valid start bit is detected, and the internally generated clock is synchronized to the incoming data frame. Then the data recovery can begin. The synchronization process is repeated for each start bit.

As described above, when the receiver clock is synchronized to the start bit, the data recovery can begin. Data recovery process is almost the same as clock recovery process. The data recovery logic samples each incoming bit 16 times for 16-oversampling mode and 8 times for 8-oversampling mode, and uses sample 8, 9 and 10 to decide data value. If more than two samples have low levels, the received bit is considered as a logic ‘0’ and if more than two samples have high levels, the received bit is considered as a logic ‘1’.

The data recovery process is then repeated until a complete frame is received, including the first stop bit. The decided bit value is stored in the receive shift register in order.

Note that the receiver only uses the first stop bit of a frame. Internally, after receiving the first stop bit, the receiver is in an idle state and waits to find the start bit.

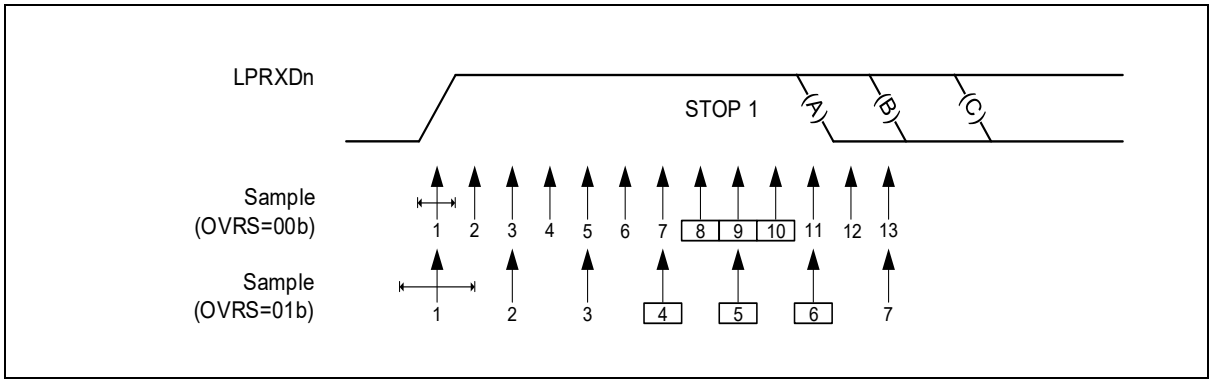
Figure 136. Sampling of Data and Parity Bit



The process for detecting stop bit is the same as clock and data recovery process. That is, if two or more samples of three center values have high level, correct stop bit is detected, or else a frame error (FE) flag is set.

After deciding whether the first stop bit is valid or not, the receiver goes to idle state and monitors the LPRXDn line to check whether a valid high to low transition is detected (start bit detection).

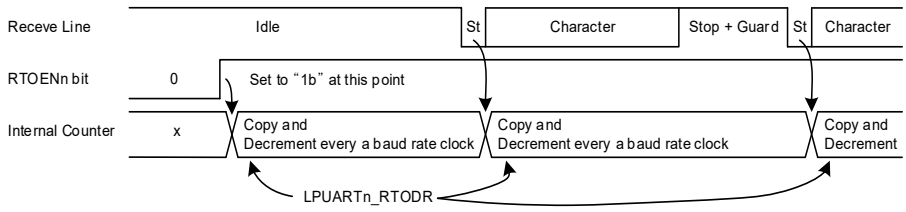
Figure 137. Stop Bit Sampling and Next Start Bit Sampling



16.3.13 LPUARTn Receive Time-Out Function

The receive time out function is used for checking the frame finish. This function is to count time with baud rate unit between the last start bit and a new start bit, and between setting the RTOEN bit of the LPUARTn_CR register and a new start bit. The LPUARTn_RTODR register should have duration time value before using the receive time out function.

Figure 138. Start Bit Sampling



NOTE:

1. The value of LPUARTn_RTODR register is transferred to an internal counter when the RTOENn bit is set to "1b" or when a start bit is received to. If the RTOEN bit is set to "1b", the internal counter counts down until underflow. The RTOnIFLAG bit of LPUARTn_IFSR register is set to "1b" on the internal counter underflow and the receive time out function is finished. So, to re-use the receive time out function, the RTOEN bit must be cleared to "0b" and then set to "1b" again.

16.3.14 1-Wire Half-Duplex Communication

1-wire half-duplex mode is selected by configuring HDCOM bit in the LPUARTn_CR1 register. The TXDn and the RXDn lines are internally connected, the RXDn pin is not used, and the TXDn pin is always an input when not transmitted. So, the TXDn pin must be configured to open-drain with an external pull-up resistor.

16.3.15 Continuous Communication Using DMA and LPUART

The LPUART 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.

The TXCIFLAGn and RXCIFLAGn bits of LPUARTn_IFSR register are the request signal for Tx and Rx of DMA.

16.3.16 LPUART Low-Power Mode

Table 120. Effect of Low-Power Mode on LPUART

Mode	Description
SLEEP	No effect
DEEP-SLEEP	DEEP-SLEEP mode operates as a wake-up source.

16.3.16.1 Wake-Up Source

There are three types of LPUART wake-up sources, SBDIEN, RXCIEN, and RCDIEN, which can be set in the LPUARTn_IER register.

The SBDIEN wakes up the system when the start bit is detected, the RXCIEN wakes up the system when a character is received, and the RCDIEN wakes up the system when the received data matches the value set in the LPUARTn_RCDR register.

After entering DEEP-SLEEP mode, if the activated event was detected, the corresponding flag of the event in the LPUARTn_IFSR register occurs: SBDIEN – SBDIFLAG/ RXCIEN – RXCIFLAG/ RCDIEN – RCDIFLAG. Writing ‘1’ to the bits clears the corresponding interrupt flag.

16.3.16.2 Description of Steps of Low-Power Mode

1. Set the LPUARTn clock to LSE or LSI in the SCU_MCCR4 register.
2. To enable the signal to be used as a wake-up source, set the LPUARTnWUE in the SCU_WUER register to ‘1’.
3. Write proper value to the WAKEN bit in the LPUARTn_CR1 register.
4. Set the wake-up event in the LPUARTn_IER register. (RCDIEN, SBDIEN, RXCIEN)
5. Enter low power mode (DEEP-SLEEP mode).
6. The system wakes up when the event set in step 4 is matched, and a flag corresponding to the event set in the LPUARTn_IFSR register occurs (RCDIFLAG, RXCIFLAG, SBDIFLAG), and the LPUARTnWU in SCU_WUSR register is triggered.
7. Write 1 to clear the event flag in the LPUARTn_IFSR register (RCDIFLAG, RTOIFLAG, SBDIFLAG) and clear the WAKEN bit in the LPUARTn_CR1 register. At this time, the LPUARTnWU in the SCU_WUSR register is also cleared.

16.4 LPUART Registers

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

Table 121. Base Addresses of LPUART

Name	Base Address
LPUART0	0x4000_8C00
LPUART1	0x4000_8D00

Table 122. LPUART Register Map

Name	Offset	Type	Description	Reset Value	Reference
LPUARTn_CR1	0x0000	RW	LPUART n Control Register 1	0x0000_0000	16.4.1
LPUARTn_CR2	0x0004	RW	LPUART n Control Register 2	0x0000_0000	16.4.2
LPUARTn_IER	0x0010	RW	LPUART n Interrupt Enable Register	0x0000_0000	16.4.3
LPUARTn_IFSR	0x0014	RW	LPUART n Interrupt Flag and Status Register	0x0000_0004	16.4.4
LPUARTn_RDR	0x0018	RO	LPUART n Receive Data Register	0x0000_0000	16.4.5
LPUARTn_TDR	0x001C	RW	LPUART n Transmit Data Register	0x0000_0000	16.4.6
LPUARTn_BDR	0x0020	RW	LPUART n Baud Rate Data Register	0x0000_FFFF	16.4.7
LPUARTn_BCMP	0x0024	RW	LPUART n Baud Rate Compensation Register	0x0000_0000	16.4.8
LPUARTn_RTODR	0x0028	RW	LPUART n Receive Time Out Data Register	0x0000_FFFF	16.4.9
LPUARTn_RCDR	0x002C	RW	LPUART n Receive Character Detection Data Register	0x0000_0000	16.4.10
LPUARTn_DLYDR	0x0030	RW	LPUART n Tx Delay Time Data Register	0x0000_0000	16.4.11

NOTE:

1. n = 0 and 1

16.4.1 LPUARTn_CR1: LPUART n Control Register1

The low power UARTn module should be configured properly before running.

LPUARTn_CR1 register is 32-bit size and accessible in 32/16/8-bit. (n=0 and 1)

LPUART0_CR1=0x4000_8C00, LPUART1_CR1=0x4000_8D00

31 30 29 28 27 26 25 24	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			PEN	STKPEN	PSEL	Reserved	DLEN[1:0]	Reserved	STOPB	OVRS[1:0]	HDCOM	TXE	RXE	WAKEN	LPUEN
-			0	0	0	0	0	-	0	0	0	0	0	0	0
-			RW	RW	RW	RW	RW	-	RW	RW	RW	RW	RW	RW	RW

14	PEN	Parity Enable	
		0	Disable parity bit generation and detection
		1	Enable parity bit generation and detection
13	STKPEN(1)	Stick Parity Enable	
		0	Disable stick parity
		1	Enable stick parity
12	PSEL	Parity Selection	
		0	Odd parity (Odd number of logic ‘1’)
		1	Even parity (Even number of logic ‘1’)
10 9	DLEN[1:0]	Data Length Selection	
		00	5-bit (Start, D0, D1, D2, D3, D4, Parity or not, Stop1, Stop2, or not)
		01	6-bit (Start, D0, D1, D2, D3, D4, D5, Parity or not, Stop1, Stop2 or not)
		10	7-bit (Start, D0, D1, D2, D3, D4, D5, D6, Parity or not, Stop1, Stop2 or not)
		11	8-bit (Start, D0, D1, D2, D3, D4, D5, D6, D7, Parity or not, Stop1, Stop2 or not)
7	STOPB	Stop bit	
		0	1 Stop bit
		1	2 Stop bit
6 5	OVRS[1:0]	Oversampling Selection	
		00	16 oversampling
		01	8 oversampling
		10	No oversampling (only 1 sampling)
		11	Reserved (no oversampling)
4	HDCOM	1-wire Half-Duplex Communication	
		0	Normal operation
		1	1-wire half-duplex communication (The TXD and RXD lines are internally connected, the RXD pin is not used, and the TXD pin is always an input when no transmitted. So, the TXD pin must be configured to pen-drain with an external pull-up resistor
3	TXE	Enable the Transmitter unit	
		0	Transmitter is disabled
		1	Transmitter is enabled
2	RXE	Enable the Receiver unit	
		0	Receiver is disabled

		1	Receiver is enabled
1	WAKEN	Wake-up function bit in DEEP SLEEP mode. The LPUARTn clock to wake-up from DEEP SLEEP mode must be selected as LSE or LSI by SCU_MCCR4 register. This bit should be set just before entering DEEP SLEEP mode and cleared on exit.	
		0	Disable wake-up function in DEEP SLEEP mode
		1	Enable wake-up function in DEEP SLEEP mode
0	LPUEN(2)	Low power UARTn enable bit. This bit can be cleared to ‘0’ during the corresponding TXE and RXE bits are all ‘0’	
		0	Disable LPUARTn block
		1	Enable LPUARTn block

NOTES:

1. On PEN=1 and STKPEN=1, the parity bit is 0 if PSEL=0 and 1 if PSEL=1.
2. If this bit is cleared, the LPUARTn current operations are discarded, the configuration is kept, and all the status flags are set to reset values.

16.4.2 LPUARTn_CR2: LPUART n Control Register2

The low power UARTn module should be configured properly before running.

LPUARTn_CR2 register is 32-bit size and accessible in 32/16/8-bit. (n=0 and 1)

LPUART0_CR2=0x4000_8C04, LPUART1_CR2=0x4000_8D04

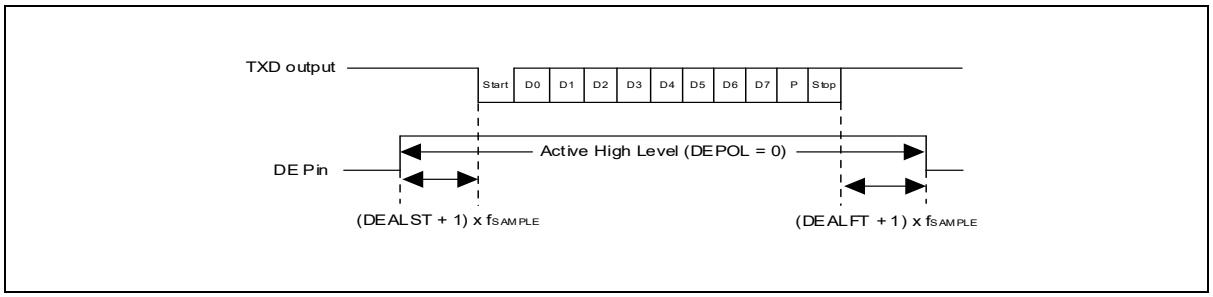
31 30 29 28 27 26 25 24	23 22 21 20 19 18 17 16	15 14 13 12 11 10 9 8	7 6	5	4 3	2 1 0
Reserved	DEALST[4:0]	Reserved	DEALFT[4:0]	DEPOL DEPEN	Reserved	RCDEN RTOEN	Reserved
-	0 0 0 0 0	-	0 0 0 0 0	0 0	-	0 0	-
-	RW RW RW RW RW	-	RW RW RW RW RW	RW RW	-	RW RW	-

20 16	DEALST[4:0]	DE Pin Active Level Start Time. The range is 0x00 to 0x1F. These bits define the time in low power UARTn clock from the active level of DE signal to the beginning of the start bit.
12 8	DEALFT[4:0]	DE Pin Active Level Finish Time. The range is 0x00 to 0x1F. These bits define the time in low power UARTn clock from the end of the stop bit to the inactive level of DE signal.
7	DEPOL(1)	DE Pin Polarity Selection
		0 Active high level. The DE pin is a high level during transmit a frame, else low level.
		1 Active low level. The DE pin is low during transmit a frame, else high level.
6	DEPEN	DE Pin Function Enable
		0 Disable DE pin function
		1 Enable DE pin function
4	RCDEN	Receive Character Detection Function Enable. This function is to compare the value of LPUARTn_RCDR register with the value just received.
		0 Disable receive detection function
		1 Enable receive detection function
3	RTOEN	Receive Time Out Function Enable. This function is to count time with baud rate units from the leading edge of start bit to a new start bit. The receive time out controller counts down from the value of LPUARTn_RTODR register very start bit and set this bit. The RTOIFLAG bit is set to '1' at the counter underflow. The counter clock is a baud rate bit unit.
		0 Disable receive time out function
		1 Enable receive time out function

NOTE:

1. TXCIFLAG bit will be set to '1b' at stop bit and the transmission of next character may start after the end of active level.

Figure 139. USART DEALST, DEALFT, DEPOL



16.4.3 LPUARTn_IER: LPUART n Interrupt Enable Register

LPUARTn_IER register is 32-bit size and accessible in 32/16/8-bit. (n=0 and 1)

LPUART0_IER=0x4000_8C10, LPUART1_IER=0x4000_8D10

31 30 29 28 27 26 25 24	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			DTXIEN	DRXIEN	RCDIEN	RTOIEN	Reserved	SBDIEN	Reserved	TXCIEN	Reserved	RXCIEN
-			0	0	0	0	-	0	-	0	-	0
-			RW	RW	RW	RW	-	RW	-	RW	-	RW

9	DTXIEN	Whether to enable or disable the DMA transmit complete interrupt
	0	Disables the DMA transmit complete interrupt.
	1	Enables the DMA transmit complete interrupt.
8	DRXIEN	Whether to enable or disable the DMA receive complete interrupt
	0	Disables the DMA receive complete interrupt.
	1	Enables the DMA receive complete interrupt.
7	RCDIEN	Receive Character Detection Interrupt Enable. On DEEP SLEEP mode, the receive character detection can wake-up system.
	0	Disable receive character detection interrupt
	1	Enable receive character detection interrupt
6	RTOIEN	Receive Time Out Interrupt Enable
	0	Disable receive time out interrupt
	1	Enable receive time out interrupt
4	SBDIEN	Start bit detection Interrupt enable bit in DEEP SLEEP mode On DEEP SLEEP mode, the detection of start bit can wake-up system.
	0	Disable start bit detection interrupt
	1	Enable start bit detection interrupt
2	TXCIEN	Transmit Complete Interrupt Enable
	0	Disable transmit complete interrupt
	1	Enable transmit complete interrupt
0	RXCIEN	Receive Data Register not empty interrupt enable bit. On DEEP SLEEP mode, it can wake-up system if there is a received character.
	0	Disable receive data not empty interrupt
	1	Enable receive data not empty interrupt

16.4.4 LPUARTn_IFSR: LPUART n Interrupt Flag and Status Register

LPUARTn_IFSR register is 32-bit size and accessible in 32/16/8-bit. (n=0 and 1)

LPUART0_IFSR=0x4000_8C14, LPUART1_IFSR=0x4000_8D14

31 30 29 28 27 26 25 24	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		DOR	FE	PE	RXBUSY	Reserved	DTXFLAG	DRXFLAG	RCDIFLAG	RTOIFLAG	Reserved	SBDIFLAG	Reserved	TXCIFLAG	Reserved	RXCIFLAG
-		0	0	0	0	-	0	0	0	0	-	0	-	1	-	0
-		RW	RW	RW	RW	-	RW	RW	RW	RW	-	RW	-	RW	-	RW

15	DOR	Data OverRun This bit is set when the receive shift register is transferred to the LPUARTn_RDR register while the RXCnIFLAG=1. The data of the shift register is ignored. This bit must be cleared by software to receive new data.
	0	No data overrun
	1	Data overrun detected. This bit is cleared to ‘0’ when write ‘1’.
14	FE	Frame Error bit This bit is set when the receive data has not a valid stop bit (That is the stop bit following the last data bit is detected as ‘0’.) The bit will be cleared by hardware if new data are received.
	0	No frame error
	1	frame error detected. This bit is cleared to ‘0’ when write ‘1’.
13	PE	Parity Error bit This bit is set when the receive data has a parity error on parity enable. The bit will be cleared by hardware if new data are received.
	0	No parity error
	1	Parity error detected. This bit is cleared to ‘0’ when write ‘1’.
12	RXBUSY	RXD Line Busy bit
	0	Receive line (RXD) is not busy
	1	Reception on going
9	DTXFLAG	DMA TX complete
	0	No request occurred
	1	Request occurred. This bit is cleared to ‘0’ when write ‘1’.
8	DRXFLAG	DMA RX complete
	0	No request occurred
	1	Request occurred. This bit is cleared to ‘0’ when write ‘1’.
7	RCDIFLAG	Receive Character detection Interrupt lag This bit is set to ‘1’ when the value of LPUARTn_RCDR register matches the value received in the non-error state of frame and parity. On match of them, the bit may be set even if data overrun occurs.
	0	No request occurred
	1	Request occurred. This bit is cleared to ‘0’ when write ‘1’.
6	RTOIFLAG	Receive Time Out Interrupt Flag. This bit is set to ‘1’ at the counter underflow of the receive time out controller.
	0	No request occurred

		1	Request occurred. This bit is cleared to ‘0’ when write ‘1’.
4	SBDIFLAG	Start Bit Detection Interrupt Flag This bit is set to ‘1’ when a start bit is detected in DEEP SLEEP mode	
		0	No request occurred
		1	Request occurred. This bit is cleared to ‘0’ when write ‘1’.
2	TXCIFLAG	Transmit Complete Interrupt Flag This bit is set to ‘1’ when the data in the transmit shift register 몬 been shifted out.	
		0	No request occurred
		1	Request occurred. This bit is cleared to ‘0’ when write ‘1’.
0	RXCIFLAG	Receive data register not empty interrupt flag. This bit is set to ‘1’ when the data in the receive shift register has been transferred to the LPUARTn_RDR register. This bit is cleared by a read to the LPUARTn_RDR register.	
		0	No request occurred
		1	Request occurred. This bit is cleared to ‘0’ when write ‘1’.

16.4.5 LPUARTn_RDR: LPUART n Receive Data Register

LPUARTn_RDR register is 32-bit size and accessible in 32/16/8-bit. (n=0 and 1)

LPUART0_RDR=0x4000_8C18, LPUART1_RDR=0x4000_8D18

31 30 29 28 27 26 25 24	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			RDATA[7:0]
-			0 0 0 0 0 0 0 0
-			RO RO RO RO RO RO RO RO

7 0	RDATA[7:0]	Receive Data. A receive shift register is moved to this register after stop bit.

16.4.6 LPUARTn_TDR: LPUART n Transmit Data Register

LPUARTn_TDR register is 32-bit size and accessible in 32/16/8-bit. (n=0 and 1)

LPUART0_TDR=0x4000_8C1C, LPUART1_TDR=0x4000_8D1C

31 30 29 28 27 26 25 24	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			TDATA[7:0]
-			0 0 0 0 0 0 0 0
-			RO RO RO RO RO RO RO RO

7 0	TDATA[7:0]	Transmit Data bits. This register is moved to the transmit shift register after a previous character is completely shifted out.

16.4.7 LPUARTn_BDR: LPUART n Baud Rate Generation Register

LPUARTn_BDR register is 32-bit size and accessible in 32/16/8-bit. (n=0 and 1)

LPUART0_BDR=0x4000_8C20, LPUART1_BDR=0x4000_8D20

31 30 29 28 27 26 25 24	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		BDATA[15:0]	
-		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

15 0	BDATA[15:0]	These bits are used to generate baud rate. 16-oversampling: Baud rate = fLPUART/{16 × (BDATA[15:0] + 1)} BDATA[15:0] range: 0x0 to 0xFFFF 8-oversampling: Baud rate: fLPUART/{8 × (BDATA[15:0] + 1)} BDATA[15:0] range: 0x0 to 0xFFFF No oversampling: This can be used with XSOSC (32.768 kHz) Baud rate: fLPUART/{8 × (BDATA[15:0] + 1)} BDATA[15:0] range: 0x0 to 0xFFFF If this register is 0x0002 on the no oversampling, The LPUARTn_BCMP[15] bit (BCMPS) shouldn't be set to '1' for minus compensation

16.4.8 LPUARTn_BCMP: LPUART n Baud Rate Compensation Register

LPUARTn_BCMP register is 32-bit size and accessible in 32/16/8-bit. (n=0 and 1)

LPUART0_BCMP=0x4000_8C24, LPUART1_BCMP=0x4000_8D24

31 30 29 28 27 26 25 24	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		BCMPS	Reserved	BCMP8	BCMP7	BCMP6	BCMP5	BCMP4	BCMP3	BCMP2	BCMP1	BCMP0
-		0	-	0	0	0	0	0	0	0	0	0
-		RW	-	RW	RW	RW	RW	RW	RW	RW	RW	RW

15	BCMPS	Baud Rate Compensation Sign	
		0	Plus one clock for compensation
		1	Minus one clock for compensation
8 0	BCMPx	Baud Rate Compensation bits. x = 0 to 8.	
		0	No compensation
		1	One clock compensation with sign bit (BCMPS).

16.4.9 LPUARTn_RTODR: LPUART n Receive Time-Out Data Register

LPUARTn_RTODR register is 32-bit size and accessible in 32/16/8-bit. (n=0 and 1)

LPUART0_RTODR=0x4000_8C28, LPUART1_RTODR=0x4000_8D28

31	30	29	28	27	26	25	24	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								RTOD[23:0]																							
-								0	0	0	0	0	0	0	0	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1
-								RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW

23 0	RTOD[23:0]	LPUARTn Receive Time Out data

16.4.10 LPUARTn_RCDR: LPUART n Receive Character Detection Register

LPUARTn_RCDR register is 32-bit size and accessible in 32/16/8-bit. (n=0 and 1)

LPUART0_RCDR=0x4000_8C2C, LPUART1_RCDR=0x4000_8D2C

31	30	29	28	27	26	25	24	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																								RCDD[7:0]							
-																								0	0	0	0	0	0	0	0
-																								RW	RW	RW	RW	RW	RW	RW	RW

7 0	RCDD[7:0]	LPUARTn Receive Character Detection Data

16.4.11 LPUARTn_DLYDR: LPUART n Delay Time Data Register

LPUARTn_DLYDR register is 32-bit size and accessible in 32/16/8-bit. (n=0 and 1)

LPUART0_DLYDR=0x4000_8C30, LPUART1_DLYDR=0x4000_8D30

31 30 29 28 27 26 25 24	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			DLYD[7:0]

-							0	0	0	0	0	0	0	0
-							RW	RW	RW	RW	RW	RW	RW	RW

7 0	DLYD[7:0]	LPUARTn Tx Delay Data. This register is used for transmitting delay time between the last stop bit and the next start bit with baud rate unit. The data in the LPUARTn_TDR register will be transferred to the transmit shift register after a delay time. Delay time: DLYD[7:0] × “Baud rate clock period”, No delay on DLYD[7:0] = 0

16.4.12 LPUART Register Map Summary

Table 123. LPUART 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	LPUARTn_CR1	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	PEN	STKPEN	PSEL	Res.	DLEN[1:0]		Res.	STOPB	OVRS[1:0]		HDCOM	TXE	RXE	WAKEN	LPUEN
	Reset value																		0	0	0		0	0		0	0	0	0	0	0	0	0
0x04	LPUARTn_CR2	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	DEALST[4:0]					Res.	Res.	Res.	DEALFT[4:0]					DEPOL	DEPEN	Res.	RCDEN	RTOEN	Res.	Res.	Res.
	Reset value												0	0	0	0	0				0	0	0	0	0	0	0		0	0			
0x10	LPUARTn_IER	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	DTXIEN	DRXIEN	RCDIEN	RTOIEN	Res.	SBDIEN	Res.	TXCIEN	Res.	RXCIEN
	Reset value																							0	0	0	0		0		0		0
0x14	LPUARTn_IFSR	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	DOR	FE	PE	RXBUSY	Res.	Res.	DTXFLAG	DRXFLAG	RCDIFLAG	RTOIFLAG	Res.	SBDIFLAG	Res.	TXCIFLAG	Res.	RG
	Reset value																	0	0	0	0			0	0	0	0		0		1		0
0x18	LPUARTn_RDR	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	RDATA[7:0]							
	Reset value																									0	0	0	0	0	0	0	0
0x1C	LPUARTn_TDR	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	TDATA[7:0]							
	Reset value																									0	0	0	0	0	0	0	0
0x20	LPUARTn_BDR	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	BDATA[15:0]															
	Reset value																	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1
0x24	LPUARTn_BCMP	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	BCMPS	Res.	Res.	Res.	Res.	Res.	Res.	BCMP8	BCMP7	BCMP6	BCMP5	BCMP4	BCMP3	BCMP2	BCMP1	BCMP0
	Reset value																	0							0	0	0	0	0	0	0	0	0
0x28	LPUARTn_RTODR	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	DR[23:0]																							
	Reset value									0	0	0	0	0	0	0	0	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1
0x2C	LPUARTn_RCDR	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	RCDD[7:0]							
	Reset value																									0	0	0	0	0	0	0	0
0x30	LPUARTn_DLYDR	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	DLYD[7:0]							
	Reset value																									0	0	0	0	0	0	0	0

17. USB Full Speed Device Interface

17.1 USB Introduction

The USB block of the A34L716 series controls an interface between a full-speed USB bus and the AHB bus.

There are five endpoints in this controller. Each endpoint can be configured as IN or OUT endpoint. It is compliant with USB 2.0 full-speed device specification and supports control/bulk/interrupt transfer types. The isochronous transfer works, but it is practically difficult to use because the buffer size is small. USB suspend/resume are supported, which allows to stop the device clocks for low-power consumption. Please refer to *Universal Serial Bus Specification Revision 1.1* for details.

17.2 USB Main Features

The USB of the A34L716 series features the following:

- Compliant with USB 2.0 Full-Speed specification
- Provides one interrupt vector with three different interrupt events (WAKEUP, USB and BUS)
- Supports Control/Bulk/Interrupt transfer type.
- Supports suspend function when no bus activity existing for 3 ms.
- Provides remote wake-up capability.
- Support Only Full-Speed (FS, 12-Mbps)
- Configurable number of endpoints from 1 to 5.
- Two USB DMA channels, 64 bytes buffer size.

17.3 USB Implementation

Table 124 describes the USB implementation in the devices.

Table 124. A34L716 USB Implementation

USB Features	USB Implementation
Number of endpoints	5
Size of dedicated packet buffer memory SRAM	1024 bytes
Dedicated packet buffer memory SRAM access scheme	4 x 8 bits / word
USB 2.0 Link Power Management (LPM) support	X
Battery Charging Detection (BCD) support	X
Embedded pull-up/down resistor on USB_DP/DM line	O

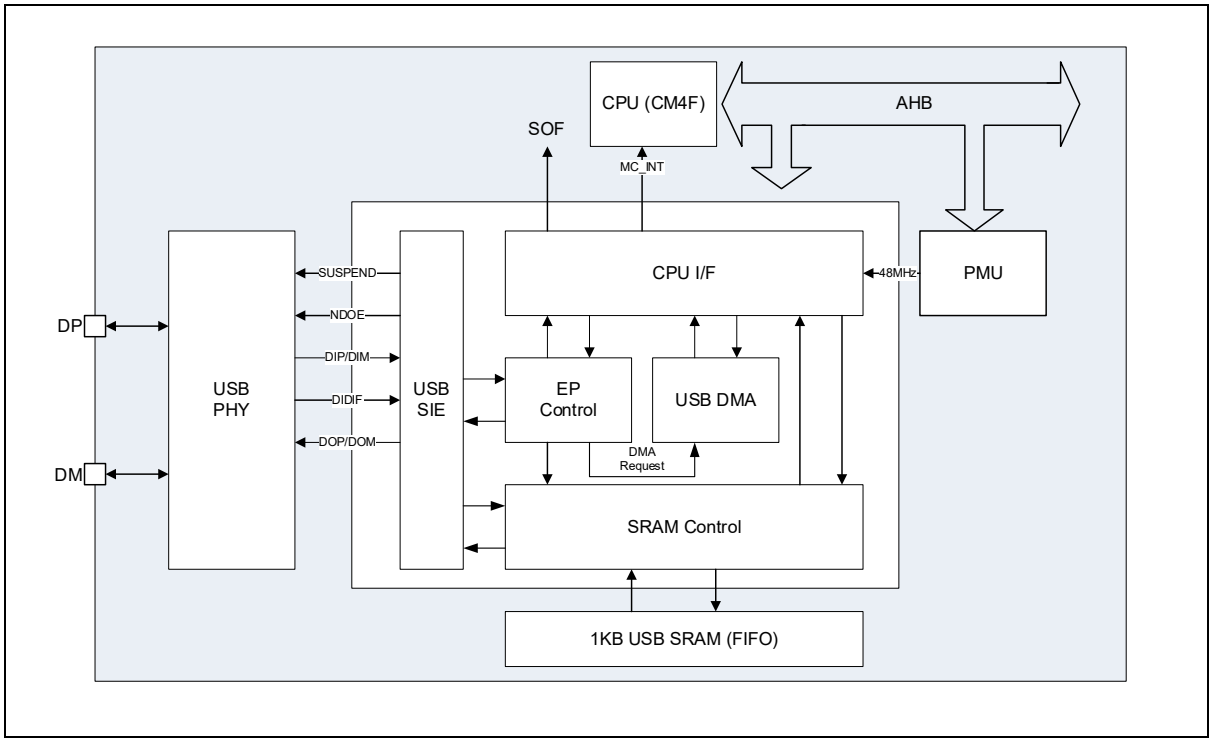
NOTE:

1. The maximum size of the FIFO is fixed at 64/128 bytes or less.

17.4 USB Functional Description

17.4.1 USB Block Diagram

Figure 140. USB Block Diagram



The USB peripheral provides a USB-compliant connection between the host PC and the function implemented by the microcontroller. Data transfer between the host PC and the system memory occurs through a dedicated packet buffer memory accessed directly by the USB peripheral. This dedicated memory size is up to 128 bytes, and up to 10 mono-directional or 5 bidirectional endpoints can be used. The USB peripheral interfaces with the USB host, detecting token packets, handling data transmission/reception, and processing handshake packets as required by the USB standard. Transaction formatting is performed by the hardware, including CRC generation and checking.

Each endpoint is associated with a buffer description block indicating where the endpoint-related memory area is located, how large it is or how many bytes must be transmitted. When a token for a valid function/endpoint pair is recognized by the USB peripheral, the related data transfer (if required and if the endpoint is configured) takes place. The data buffered by the USB peripheral is loaded in an internal 16-bit register and memory access to the dedicated buffer is performed. When all the data has been transferred, if needed, the proper handshake packet over the USB is generated or expected according to the direction of the transfer.

At the end of the transaction, an endpoint-specific interrupt is generated, reading status registers and/or using different interrupt response routines. The microcontroller can determine:

- Compliant with USB 2.0 Full-Speed specification
- Provides one interrupt vector with three different interrupt events (WAKEUP, USB and BUS)

The unit can be placed in Low-Power mode (SUSPEND mode), by writing in the control register, whenever required. At this time, all static power dissipation is avoided, and the USB clock can be slowed down or stopped. The detection of activity at the USB inputs, while in Low-Power mode, wakes the device up asynchronously. A special interrupt source can be connected directly to a wake-up line to allow the system to immediately restart the normal clock generation and/or support direct clock start/stop.

17.4.2 USB Pins and Signals

The external pins assigned for USB are described in Table 125.

Table 125. Pin Assignment of USB: External Pins

Pin Name	Type	Description	Supported Packages		
			A34L716 (100-LQFP)	A34L716 (64-LQFP)	A34L716 (56-QFN)
USBDP	I/O	USB data line plus	O	O	O
USBDM	I/O	USB data line minus	O	O	O

17.4.3 USB Description of USB Block

The USB peripheral implements all the features related to USB interfacing, which include the following blocks:

- USB Physical Interface (USB PHY): This block maintains the electrical interface to an external USB host. It contains the differential analog transceiver itself, controllable embedded pull-up / pull-down resistor (connected to USB_DP / USB_DM line), multiplexed on same USB_DP and USB_DM lines.
- Serial Interface Engine (SIE): The SIE handles NRZI encoding/decoding, bit stuffing/unstuffing, and CRC generation/checking. It generates a 12 MHz USB clock from the 48 MHz input clock to the core and synchronizes this clock to the incoming data stream when receiving data from the USB. It generates headers for packets to be transmitted and decodes the headers of received packets. This unit also generates signals according to USB peripheral events, such as Start of Frame (SOF), USB_Reset, Data errors etc. and to Endpoint related events like end of transmission or correct reception of a packet; these signals are then used to generate interrupts.
- Endpoint Controllers: Two controller state machines are used: one for control transfers over Endpoint 0, and one for Bulk, Interrupt transactions over Endpoints 1 to 5. Each endpoint has an associated register containing the endpoint type and its current status. For mono-directional/single-buffer endpoints, a single register can be used to implement two distinct endpoints. The number of endpoints is 5, allowing up to 10 mono-directional/single-buffer in any combination.
- SRAM Controllers: The SRAM controller provides an interface to a single block of synchronous single-port RAM which is used to buffer packets between the CPU and USB. It takes the FIFO pointers from the endpoint controllers, converts them to address pointers within the SRAM block and generates the SRAM access control signals.
- CPU Interface: The CPU Interface allows access to the control/status registers and the FIFOs for each endpoint. It also generates an interrupt to the CPU when a packet has been successfully transmitted or received, or when the core enters Suspend mode or resumes from Suspend mode. Also, it contains information about the status of the whole USB peripheral and used to force some USB events, such as resume and power-down.
- USB DMA: The USB DMA provides two channels to allow FIFOs for IN Endpoints 1 to 5 and OUT Endpoints 1 to 5 to be accessed by dedicated DMA controller. The DMA request lines are individually enabled by a bit in the appropriate INCSR2 control register or OUTCSR2 control register. These DMA Modes are primarily designed to be used where large blocks of data are transferred to a Bulk endpoint. The USB protocol requires such packets to be split into a series of packets of maximum packet size = 8, 16, 32 or 64 bytes. The last packet in the series will be less than the maximum packet size (or a null packet if the total size of the transfer is an exact multiple of the maximum packet size) and the receiver may use this ‘short’ packet to signal the end of the transfer. DMA Mode 1 can be used to avoid the overhead of having to interrupt the processor after each individual packet – instead just interrupting the processor after the transfer has completed. If the endpoint is configured for isochronous transfers, DMA Mode 0 should always be selected. However, the use of isochronous transfers is greatly limited because the buffer size in each FIFO is small. (FIFO buffers are supported up to 128 bytes and do not meet to 1023 bytes.)

17.5 Programming Considerations

The following sections describe the main tasks required for the application software to obtain USB compliant behavior and to ease application software development.

17.5.1 Select Clock Source

For USB communication, the 48 MHz clock output from the PLL is used. The clock source for the PLL can use either the HSE or the HSI. If the HSE is used as the PLL source clock, the PLL output clock can be used for USB peripheral without calibration. However, if HSI is used, the HSI output clock must be calibrated to 32 MHz using the HSICMU before using it as the PLL source clock.

The HSICMU is ideally suited to provide a precise clock to the USB peripheral. In such case, the synchronization signal can be derived from the start-of-frame (SOF) packet signalization on the USB bus, which is sent by a USB host at 1 ms intervals. The synchronization signal can also be derived from the LSE oscillator output or from an external pin.

17.5.2 Endpoint Initialization

The first step to initialize an endpoint is to write appropriate values to the registers so that the USB peripheral finds the data to be transmitted already available and the data to be received can be buffered.

Typically, unidirectional endpoints need to initialize only bits and registers related to the supported direction. Once the transmission and/or reception are enabled, setting register should not be modified by the application software, as the hardware can change their value on the fly. When the data transfer operation is completed, notified by an interrupt event (if enabled), they can be accessed again to re-enable a new operation.

17.5.3 Endpoint Control

There are five endpoints in this controller. Each of the endpoints can be configured as Control, Bulk, Interrupt or Isochronous transfer type. All the operations including Control, Bulk, Interrupt and Isochronous transfer are implemented in this block. It is also used to manage data sequential synchronization, endpoint state control, current endpoint start address, current transaction status, and data buffer status in each endpoint.

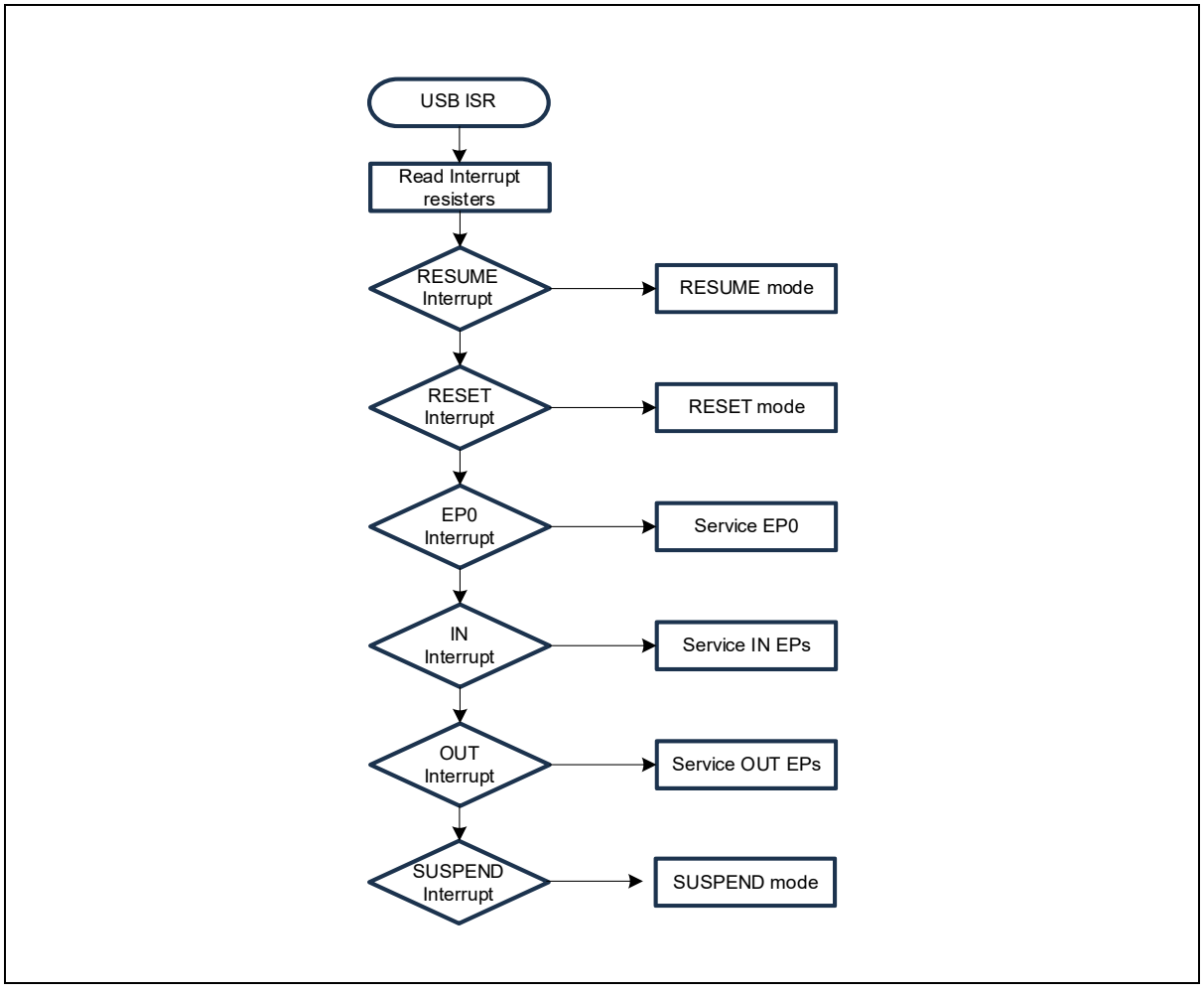
17.5.4 USB Interrupt Handling

This USB provides one interrupt vector with USB and BUS interrupt events. The USB event notifies users of some USB requests, such as IN ACK, OUT ACK., and the BUS event notifies users of some bus events, such as suspend and, resume. The related bits must be set in the interrupt enable register of USB Device Controller to enable USB interrupts.

USB interrupt is used to notify users of any USB event on the bus, and users can read INTRIN1 and INTROUT1 to take necessary responses.

Same as USB interrupt, BUS interrupt notifies users of some bus events, like SOF, USB reset, suspend, and resume. A user can read INTRUSB to acknowledge bus events.

Figure 141. USB Interrupt Service Routine



17.5.5 Power Saving

If set to support Suspend mode, the USB automatically turns off the PHY transceiver to save power while the chip enters Suspend mode. To further reduce current consumption in the application, sleep / deep sleep mode can be used, and the clock source must be changed to either LSE or LSI to ensure that the USB Peripheral operates to a minimum. When an application wakes up in sleep mode or deep sleep mode by an interrupt, the clock source must be restored to 48 MHz PLL output to resume USB communication.

17.5.6 Buffer Control

The USB Specification defines an endpoint as a source or sink of data. Since USB is a serial bus, a device endpoint is a FIFO which sequentially empties or fills with USB data bytes. The A34L716 USB can uniquely address five endpoints, IN1 through IN5 and OUT1 through OUT5.

There are 960 bytes of SRAM in the controller and the five endpoints share this buffer. The seven 64-byte buffers are designated EP0, EP1IN/OUT, EP2IN/OUT and EP3IN/OUT and the four 128-byte buffers are designated EP4IN/OUT and EP5IN/OUT.

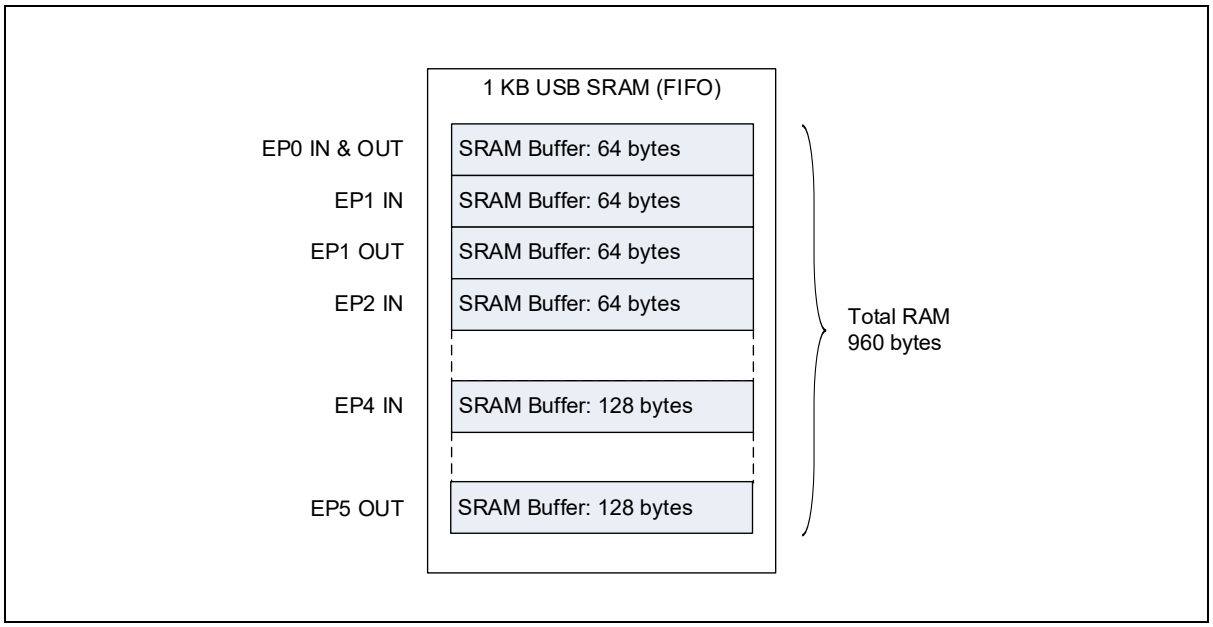
EP0 is the default CONTROL endpoint, a bidirectional endpoint that uses a single 64-byte buffer for both IN and OUT data. Firmware reads or fills the EP0 buffer when the data stage of a CONTROL transfer is required.(1)

Other EPn(2) IN/OUT use separate 64-byte or 128-byte buffers. Firmware can configure these endpoints as BULK or INTERRUPT. Also, EP4 and EP5 have 128-byte buffers, they can be used as double buffers to improve speed when using BULK transfer. But the buffer size is predetermined and cannot be changed by the user to suit bandwidth requirements. FIFO is divided into read and write, so the written value cannot be read again. To check the written value, users need to send data to USB.

NOTES:

1. The eight SETUP bytes in a CONTROL transfer do not appear in the 64-byte EP0 endpoint buffer. Instead, to simplify programming, the A34L716 USB automatically stores the eight SETUP bytes in a separate buffer.
2. n = 1 to 5

Figure 142. USB Buffer Control

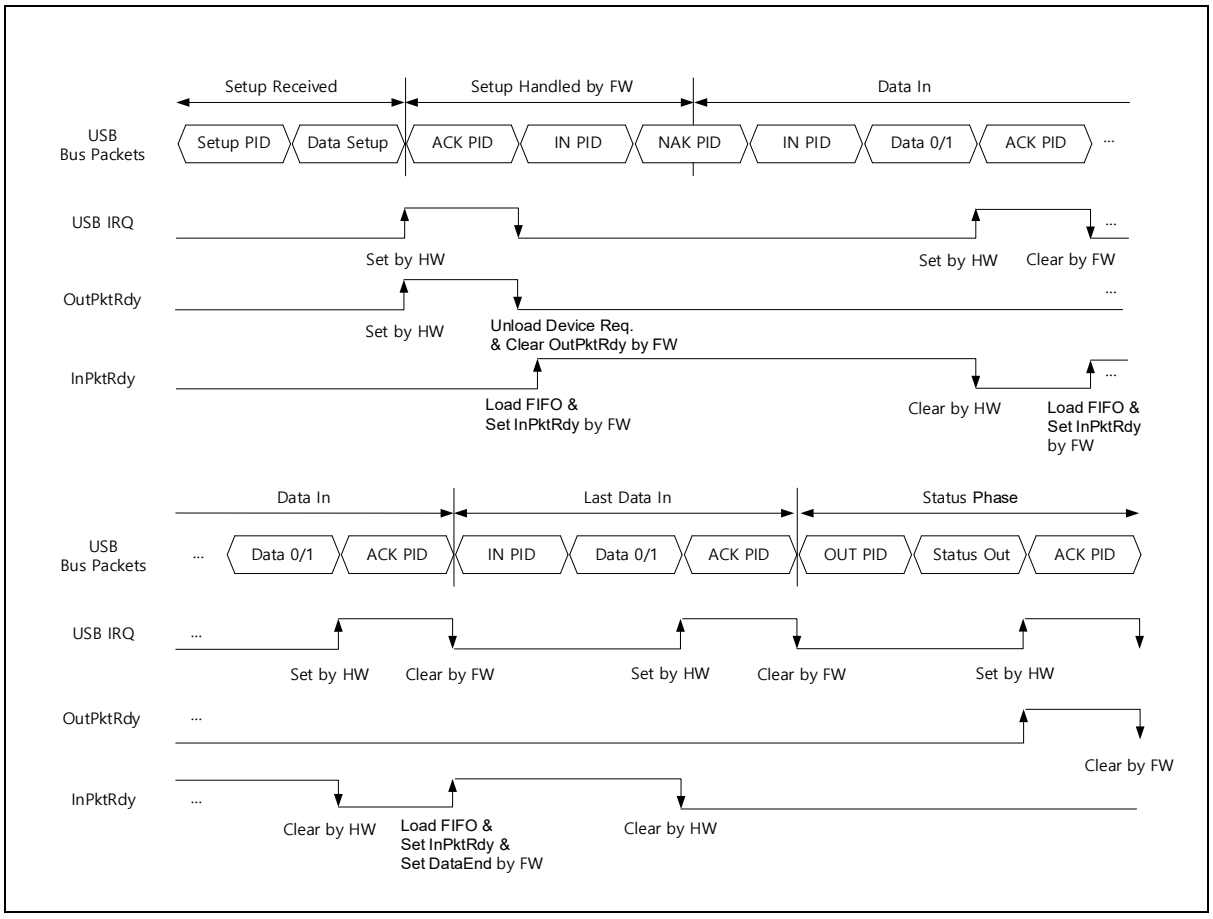


17.5.7 Handling Transactions with USB Device Peripheral

Users can use interrupt to monitor the USB transactions. When transactions occur, USB_IRQHandler will be set by hardware and send an interrupt request to CPU (if related interrupt enabled). The following is the control flow with interrupt enabled.

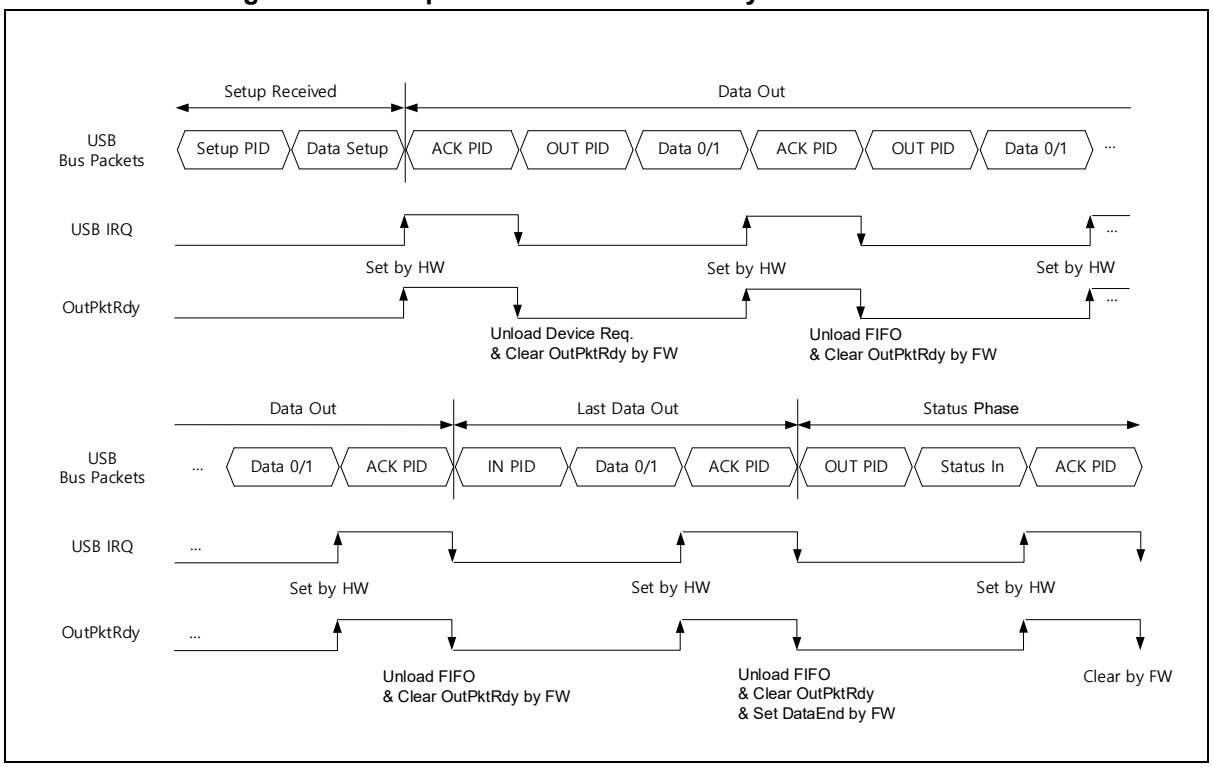
When USB host has requested data from a device controller, users need to prepare related data in the specified endpoint buffer in advance. After buffering the required data, users need to write the actual data length in the specified INMAXP register. Once this register is written, the internal signal “InPktRdy” will be asserted, and the buffering data will be transmitted immediately after receiving associated IN token from Host. Note that after transferring the specified data, the signal “InPktRdy” will be de-asserted automatically by hardware.

Figure 143. Setup Transaction Followed by Data IN Transaction



Alternatively, when USB host wants to transmit data to the OUT endpoint in the device controller, hardware will buffer these data to the specified endpoint buffer. After this transaction is completed, hardware will record the data length in specified OUTMAXP register and de-assert the internal signal “OutPktRdy”. This will prevent hardware accepting next transaction until users move out the current data in the related endpoint buffer. Once users have processed this transaction, the specified OUTMAXP register needs to be written by firmware to assert the signal “OutPktRdy” again to accept the next transaction.

Figure 144. Setup Transaction Followed by Data Out Transfer



17.6 USB Registers

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

Table 126. Base Address of USB

Name	Base Address
USB	0x4100_6000

Table 127. USB Register Map

Name	Offset	Type	Description	Reset value	Reference
USB_FADDR	0x0000	RW	Function Address register	0x0000_0000	17.6.1
USB_POWER	0x0004	RW	Power management register	0x0000_0000	17.6.2
USB_INTRIN1	0x0008	RO	Interrupt Flag Register for Endpoint 0 plus IN Endpoints 1 to 5	0x0000_0000	17.6.3
USB_INTROUT1	0x0010	RO	Interrupt Flag Register for OUT Endpoints 1 to 5	0x0000_0000	17.6.4
USB_INTRUSB	0x0018	RO	Interrupt Flag Register for Common USB Interrupts	0x0000_0000	17.6.5
USB_INTRIN1E	0x001C	RW	Interrupt Enable Register for Endpoint 0 plus IN Endpoints 1 to 5	0x0000_003F	17.6.6
USB_INTROUT1E	0x0024	RW	Interrupt Enable Register for OUT Endpoints 1 to 5	0x0000_003E	17.6.7
USB_INTRUSBE	0x002C	RW	Interrupt Enable Register for Common USB Interrupts	0x0000_0006	17.6.8
USB_FRAME1	0x0030	RO	Frame number bit 0 to 7	0x0000_0000	17.6.9
USB_FRAME2	0x0034	RO	Frame number bit 8 to 10	0x0000_0000	17.6.10
USB_INDEX	0x0038	RW	Index Register for Selecting the Endpoint Status and Control Registers	0x0000_0000	17.6.11
USB_PUPDCR	0x0040	RW	DP/DN Pins Pull-up/Down Resistor Control Register	0x0000_0000	17.6.12
USB_PHYCR	0x0044	RW	PHY IP SUSPEND mode control register	0x0000_0001	17.6.13
USB_INMAXP	0x0050	RW	Maximum Packet Size for IN Endpoint. (Index Register set to select Endpoints 1 – 5 only)	0x0000_0000	17.6.14
USB_CSR0	0x0054	RW	Control Status Register for Endpoint 0. (Index Register set to select Endpoint 0)	0x0000_0000	17.6.15
USB_INCSR1	0x0054	RW	Control Status Register 1 for IN Endpoint. (Index Register set to select Endpoints 1 – 5)	0x0000_0000	17.6.16
USB_INCSR2	0x0058	RW	Control Status Register 2 for IN Endpoint. (Index Register set to select Endpoints 1 – 5 only)	0x0000_0020	17.6.17

Table 127. USB Register Map (continued)

Name	Offset	Type	Description	Reset value	Reference
USB_OUTMAXP	0x005C	RW	Maximum Packet Size for OUT Endpoint. (Index Register set to select Endpoints 1 – 5 only)	0x0000_0000	17.6.18
USB_OUTCSR1	0x0060	RW	Control Status Register 1 for OUT Endpoint. (Index Register set to select Endpoints 1 – 5 only)	0x0000_0000	17.6.19
USB_OUTCSR2	0x0064	RW	Control Status Register 2 for OUT Endpoint. (Index Register set to select Endpoints 1 – 5 only)	0x0000_0000	17.6.20
USB_COUNT0	0x0068	RO	Number of Received Bytes in Endpoint 0 FIFO. (Index Register set to select Endpoint 0)	0x0000_0000	17.6.21
USB_OUTCOUNT1	0x0068	RO	Number of Bytes in OUT Endpoint FIFO (Lower Byte). (Index register set to select Endpoints 1 – 5 only)	0x0000_0000	17.6.22
USB_OUTCOUNT2	0x006C	RO	Number of Bytes in OUT Endpoint FIFO (Upper Byte). (Index Register set to select Endpoints 1 – 5 only)	0x0000_0000	17.6.23
USB_FIFO0	0x0080-0x008C	RW	FIFO for EP0	0x0000_0000	17.6.24
USB_FIFO1	0x0090-0x009C	RW	FIFO for EP1	0x0000_0000	17.6.24
USB_FIFO2	0x00A0-0x00AC	RW	FIFO for EP2	0x0000_0000	17.6.24
USB_FIFO3	0x00B0-0x00BC	RW	FIFO for EP3	0x0000_0000	17.6.24
USB_FIFO4	0x00C0-0x00CC	RW	FIFO for EP4	0x0000_0000	17.6.24
USB_FIFO5	0x00D0-0x00DC	RW	FIFO for EP5	0x0000_0000	17.6.24
USB_DMAINTR	0x0200	RW	USB DMA Interrupt Pending Register	0x0000_0000	17.6.25
USB_DMA1CNTL	0x0204	RW	USB DMA Channel 1 Control Register	0x0000_0000	17.6.26
USB_DMA1ADDR	0x0208	RW	USB DMA Channel 1 Memory Address Register	0x0000_0000	17.6.27
USB_DMA1COUNT	0x020C	RW	USB DMA Channel 1 Transfer Byte Count Register	0x0000_0000	17.6.28
USB_DMA2CNTL	0x0214	RW	USB DMA Channel 2 Control Register	0x0000_0000	17.6.29
USB_DMA2ADDR	0x0218	RW	USB DMA Channel 2 Memory Address Register	0x0000_0000	17.6.30
USB_DMA2COUNT	0x021C	RW	USB DMA Channel 2 Transfer Byte Count Register	0x0000_0000	17.6.31

17.6.1 USB_FADDR: Function Address Register

FADDR is an 8-bit register that should be written with the function's 7-bit address. It is used for decoding the function address in subsequent token packets.

USB_FADDR=0x4100_6000

31 30 29 28 27 26 25 24	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	FADDR[6:0]						
-			0	0	0	0	0	0	0	0
-			RO	RW						

7	UPDATE	Set when FADR is written. Cleared when the new address takes effect (at the end of the current transfer).	
		0	Clear by USB when end of the current transfer.
		1	Updated Function address. (FADDR)
6 0	FADDR[6:0]	The function address	

17.6.2 USB_POWER: Power Management Register

POWER is used for controlling Suspend and Resume signaling.

USB_POWER=0x4100_6004

31 30 29 28 27 26 25 24	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			ISOUPDATE	Reserved	RESET	RESUME	SUSPENDMD	SUSPENDEN
-			0	-	0	0	0	0
-			RW	-	RO	RW	RO	RW

7	ISOUPDATE	When set by the CPU, this Device will wait for an SOF token from the time InPktRdy is set before sending the packet. If an IN token is received before an SOF token, then a zero-length data packet will be sent. This bit is only used by endpoints performing Isochronous transfers.	
		0	Disables.
		1	Wait SOF token in isochronous (ISO) mode
3	RESET	set while reset signaling is present on the bus	
		0	Cleared by USB
		1	Detect Reset status on the bus
2	RESUME	Set by the CPU to generate resume signaling when the function is in suspend mode.	
		0	clear this bit by CPU after 10 ms (a maximum of 15ms) to end Resume signaling.
		1	Generate Resume signal in suspend mode
1	SUSPENDMD	Set by the USB when suspend mode is entered.	
		0	Cleared when the CPU reads the interrupt register, or sets the resume bit of this register
		1	Detect Suspend mode
0	SUSPENDEN	Set by the CPU to enable entry into suspend mode when suspend signaling is received on the bus	
		0	Disables.
		1	Enables.

17.6.3 USB_INTRIN1: Interrupt for EP0 + IN EP1 to EP5 Register

INTRIN1 indicates which of the interrupts for IN Endpoints 1 to 5 are currently active. It also indicates whether the Endpoint 0 interrupt is currently active. All active interrupts will be cleared when this register is read.

USB_INTRIN1=0x4100_6008

31 30 29 28 27 26 25 24	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				EP5IFG	EP4IFG	EP3IFG	EP2IFG	EP1IFG	EP0IFG
-				0	0	0	0	0	0
-				RO	RO	RO	RO	RO	RO

5	EP5IFG	IN Endpoint 5 interrupt	
		0	-
		1	Issued interrupt IN EP5
4	EP4IFG	IN Endpoint 4 interrupt	
		0	-
		1	Issued interrupt IN EP5
3	EP3IFG	IN Endpoint 3 interrupt	
		0	-
		1	Issued interrupt IN EP3
2	EP2IFG	IN Endpoint 2 interrupt	
		0	-
		1	Issued interrupt IN EP2
1	EP1IFG	IN Endpoint 1 interrupt	
		0	-
		1	Issued interrupt IN EP1
0	EP0IFG	IN Endpoint 0 interrupt	
		0	-
		1	Issued interrupt IN EP0

17.6.4 USB_INTROUT1: Interrupt for OUT EP1 to EP5 Register

INTROUT1 indicates which of the interrupts for OUT Endpoints 1-5 are currently active. All active interrupts will be cleared when this register is read.

USB_INTROUT1=0x4100_6010

31 30 29 28 27 26 25 24	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				EP5IFG	EP4IFG	EP3IFG	EP2IFG	EP1IFG	Reserved
-				0	0	0	0	0	-
-				RO	RO	RO	RO	RO	-

5	EP5IFG	IN Endpoint 5 interrupt	
		0	-
		1	Issued interrupt IN EP5
4	EP4IFG	IN Endpoint 4 interrupt	
		0	-
		1	Issued interrupt IN EP5
3	EP3IFG	IN Endpoint 3 interrupt	
		0	-
		1	Issued interrupt IN EP3
2	EP2IFG	IN Endpoint 2 interrupt	
		0	-
		1	Issued interrupt IN EP2
1	EP1IFG	IN Endpoint 1 interrupt	
		0	-
		1	Issued interrupt IN EP1

17.6.5 USB_INTRUSB: Interrupt for Common USB

INTRUSB indicates which USB interrupts are currently active. All active interrupts will be cleared when this register is read.

USB_INTRUSB=0x4100_6018

31 30 29 28 27 26 25 24	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				SOFIFG	RESETIFG	RESUMEIFG	SUSPENDIFG
-				0	0	0	0
-				RO	RO	RO	RO

3	SOFIFG	Set at the start of each frame.	
		0	-
		1	Detect SOF
2	RESETIFG	Set when reset signaling is detected on the bus.	
		0	-
		1	Detect Reset
1	RESUMEIFG	Set when resume signaling is detected on the bus while the Device is in suspend mode.	
		0	-
		1	Detect Resume
0	SUSPENDIFG	Set when suspend signaling is detected on the bus.	
		0	-
		1	Detect Suspend

17.6.6 USB_INTRIN1E: Interrupt Enable for INTIN

INTRIN1E provides interrupt enable bits for the interrupts. On reset, the bits corresponding to End point 0 and the IN endpoints included in the design are set to 1, while the remaining bits are set to 0.

USB_INTRIN1E=0x4100_601C

31 30 29 28 27 26 25 24	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				EP5IEN	EP4IEN	EP3IEN	EP2IEN	EP1IEN	EP0IEN
-				1	1	1	1	1	1
-				RW	RW	RW	RW	RW	RW

5	EP5IEN	Interrupt enable for EP5 IN	
		0	Disables.
		1	Enables.
4	EP4IEN	Interrupt enable for EP4 IN	
		0	Disables.
		1	Enables.
3	EP3IEN	Interrupt enable for EP3 IN	
		0	Disables.
		1	Enables.
2	EP2IEN	Interrupt enable for EP2 IN	
		0	Disables.
		1	Enables.
1	EP1IEN	Interrupt enable for EP1 IN	
		0	Disables.
		1	Enables.
0	EP0IEN	Interrupt enable for EP0 IN	
		0	Disables.
		1	Enables.

17.6.7 USB_INTROUT1E: Interrupt Enable for INTOUT

INTROUT1E provides interrupt enable bits for the interrupts. On reset, the bits corresponding to the OUT endpoints included in the design are set to 1, while the remaining bits are set to 0.

USB_INTROUT1E=0x4100_6024

31 30 29 28 27 26 25 24	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				EP5IEN	EP4IEN	EP3IEN	EP2IEN	EP1IEN	Reserved
-				1	1	1	1	1	-
-				RW	RW	RW	RW	RW	-

5	EP5IEN	Interrupt enable for EP5 OUT
	0	Disables.
	1	Enables.
4	EP4IEN	Interrupt enable for EP4 OUT
	0	Disables.
	1	Enables.
3	EP3IEN	Interrupt enable for EP3 OUT
	0	Disables.
	1	Enables.
2	EP2IEN	Interrupt enable for EP2 OUT
	0	Disables.
	1	Enables.
1	EP1IEN	Interrupt enable for EP1 OUT
	0	Disables.
	1	Enables.

17.6.8 USB_INTRUSBE: Interrupt Enable for INTUSB

INTRUSBE provides interrupt enable bits for each of the interrupts in INTRUSBE register.

USB_INTRUSBE=0x4100_602C

31 30 29 28 27 26 25 24	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				SOFIEN	RESETIEN	RESUMEIEN	SUSPENDIEN
-				0	1	1	0
-				RW	RW	RW	RW

3	SOFIEN	Interrupt enable for SOF	
		0	Disables.
		1	Enables.
2	RESETIEN	Interrupt enable for Rest	
		0	Disables.
		1	Enables.
1	RESUMEIEN	Interrupt enable for Resume	
		0	Disables.
		1	Enables.
0	SUSPENDIEN	Interrupt enable for Suspend	
		0	Disables.
		1	Enables.

17.6.9 USB_FRAME1: Frame Number Bit 0 to 7

FRAME1 holds the lower eight bits of the last received frame number.

USB_FRAME1=0x4100_6030

31 30 29 28 27 26 25 24	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			FRMNO[7:0]
-			0x00
-			R0

7 0	FRMNO[7:0]	Lower 8 bits of Frame Number

17.6.10 USB_FRAME2: Frame Number Bit 8 to 10

FRAME2 holds the upper three bits of the last received frame number.

USB_FRAME2=0x4100_6034

31 30 29 28 27 26 25 24	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				FRMNO[2:0]		
-				0	0	0
-				R0		

2 0	FRMNO[2:0]	Lower 3 bits of Frame Number

17.6.11 USB_INDEX: Index for Selecting Endpoint Register

INEDX determines which endpoint control/status registers are accessed at addresses 0x4100_6040 (USB_INMAXP) to 0x4100_605C (USB_OUTCOUNT2).

USB_INDEX=0x4100_6038

31 30 29 28 27 26 25 24	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				INDEX[3:0]
-				0 0 0 0
-				RW

3 0	INDEX[3:0]	Upper 3 bits of Frame Number

17.6.12 USB_PUPDCR: DP/DN Pins Pull-Up/Pull-Down Resistor Control Register

USB_PUPDCR control Pull-up or Pull-down for DP/DN.

USB_PUPDCR=0x4100_6040

31 30 29 28 27 26 25 24	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				DNPDEN	DNPUEN	Reserved	DPPDEN	DPPUEN
-				0	0	-	0	0
-				RW	RW	-	RW	RW

5	DNPDEN	DN Pin Pull-down resistor enable bit	
		0	Disables.
		1	Enables.
4	DNPUEN	DN Pin Pull-up resistor enable bit	
		0	Disables.
		1	Enables.
1	DPPDEN	DP Pin Pull-down resistor enable bit	
		0	Disables.
		1	Enables.
0	DPPUEN	DP Pin Pull-up resistor enable bit	
		0	Disables.
		1	Enables.

17.6.13 USB_PHYCR: USB Power Control Register

USB_PHYCR control power.

USB_PHYCR=0x4100_6044

31 30 29 28 27 26 25 24	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			PHYSUSP_MODE
-			1
-			RW

0	PHYSUSP_MODE	USB PHY IP Suspend mode enable bit	
		0	Disables.
		1	Enables.

17.6.14 USB_INMAXP: Maximum Package Size for IN Register

INMAXP holds the maximum packet size for transactions through the currently-selected IN endpoint – in units of 8 bytes, except that a value of 128 sets the maximum packet size to 1023 (the maximum size for an Isochronous packet transferred in a full-speed transaction) rather than 1024. When setting this value, the constraints imposed by the USB Specification on packet sizes for Bulk, Interrupt, and Isochronous transactions in Full-speed operations should be considered.

USB_INMAXP=0x4100_6050

31 30 29 28 27 26 25 24	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			INMAXPS[7:0]
-			0x00
-			RW

7 0	INMAXPS[7:0]	Maximum Packet Size / transaction for IN endpoint. 8 bytes unit. (ex. 0x04 = 32 bytes, but 0x80 = 1023 bytes)

17.6.15 USB_CSR0: Control Status Register 0 for EP0 Register

CSR0 provides control and status bits for Endpoint 0.

USB_CSR0=0x4100_6054

31 30 29 28 27 26 25 24	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			SERVICEDSETUPEND	SERVICEDOUTPKTRDY	SENDSTALL	SETUPEND	DATAEND	SENTSTALL	INPKTRDY	OUTPKTRDY
-			0	0	0	0	0	0	0	0
-			WS	WS	WS	RO	WS	RC	RWS	RO

7	SERVICEDSETUPEND	Clear the SetupEnd bit.
		0 -
		1 Clear SetupEnd
6	SERVICEDOUTPKTRDY	Clear the OutPktRdy bit.
		0 -
		1 Clear OutPktRdy
5	SENDSTALL	set to terminate the current transaction.
		0 -
		1 Terminate transaction.
4	SETUPEND	Detected when a control transaction ends before the DataEnd bit has been set. An interrupt will be generated, and the FIFO flushed at this time.
		0 Clear by SvSetupEnd.
		1 Detect control transaction ends.
3	DATAEND	The CPU sets this bit when: setting InPktRdy for the last data packet clearing OutPktRdy after unloading the last data packet setting InPktRdy for a zero-length data packet
		0 -
		1 Set when setting InPktRdy and OutPktRdy.
2	SENTSTALL	Set when a STALL handshake is transmitted. The CPU should clear this bit.
		0 -
		1 Detect STALL handshake
1	INPKTRDY	set after loading a data packet into the FIFO. An interrupt is generated when the bit is cleared.
		0 Self-clearing after data packet has been transmitted.
		1 Indicate loading a data packet into the FIFO.
0	OUTPKTRDY	This bit is set when a data packet has been received. An interrupt is generated when this bit is set.
		0 Clear by SvOutPktRdy
		1 Detect a data packet has been received.

17.6.16 USB_INCSR1: Control Status Register 1 for IN Register

INCSR1 provides control and status bits for IN transactions through the currently selected endpoint. There is an INSCR1 register for each IN endpoint (not including Endpoint 0).

USB_INCSR1=0x4100_6054

31 30 29 28 27 26 25 24	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				CLRDATATOG	SENTSTALL	SENDSTALL	FLUSHFIFO	UNDERRUN	FIFONOTEMPTY	INPKTRDY
-				0	0	0	0	0	0	0
-				WS	RC	RW	WS	RC	RC	RS

6	CLRDATATOG	Reset the endpoint IN data toggle to 0.
		0 -
		1 Reset the endpoint IN data toggle
5	SENTSTALL	Detect when a STALL handshake is transmitted. The FIFO is flushed and the InPktRdy bit is cleared. The CPU should clear this bit.
		0 -
		1 Detect STALL handshakes are transmitted.
4	SENDSTALL	Set to issue a STALL handshake to an IN token. The CPU clears this bit to terminate the stall condition. This bit has no effect if the IN endpoint is in ISO mode.
		0 Clear to terminate the stall condition.
		1 Issue a STALL handshake.
3	FLUSHFIFO	Set to flush the next packet to be transmitted from the endpoint IN FIFO. The FIFO pointer is reset and the InPktRdy bit is cleared. **NOTE:** 1. If the FIFO contains two packets, FlushFIFO will need to be set twice to completely clear the FIFO.
		0 -
		1 Set to flush FIFO.
2	UNDERRUN	In ISO mode, this bit is set when a zero-length data packet is sent after receiving an IN token with the InPktRdy bit not set. In Bulk/Interrupt mode, this bit is set when a NAK is returned in response to an IN token. The CPU should clear this bit.
		0 Clear
		1 Detect zero length data packet or NAK
1	FIFONOTEMPTY	This bit is set when there is a lease 1 packet in the IN FIFO.
		0 Clear.
		1 Detect FIFO is not empty.
0	INPKTRDY	Set after loading a data packet into the FIFO. It is cleared automatically when a data packet has been transmitted. An interrupt is generated (if enabled) when the bit is cleared.
		0 Auto cleared after data packet has been transmitted.
		1 Loaded a data packet into the FIFO.

17.6.17 USB_INCSR2: Control Status Register 2 for IN Register

INCSR2 provides further control bit for IN transactions through the currently selected endpoint. There is an INCSR2 register for each IN endpoint (not including Endpoint 0)

USB_INCSR2=0x4100_6058

31 30 29 28 27 26 25 24	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			AUTOSET	ISO	MODE	DMAENAB	FRCDATATOG	Reserved
-			0	0	1	0	0	-
-			RW	RW	RW	RW	RW	-

7	AUTOSET	If the CPU sets this bit, InPktRdy will be automatically set when data of the maximum packet size (value in InMaxP) is loaded into the IN FIFO. If a packet of less than the maximum packet size is loaded, then InPktRdy will have to be set manually.
		0 Disable.
		1 Enable Automatically set InPktRdy.
6	ISO	Enable the IN endpoint for isochronous transfers (ISO mode) and clears it to enable the IN endpoint for bulk/interrupt transfers.
		0 Enable Bulk or Interrupt mode.
		1 Enable ISO mode
5	MODE	Set the endpoint direction. Valid only when the same endpoint FIFO is used for both IN and OUT transactions.
		0 OUT direction.
		1 IN direction.
4	DMAENAB	Set to enable the DMA request for the IN endpoint.
		0 Disable.
		1 Enable.
3	FRCDATATOG	Set to force the endpoint's IN data toggle to switch after each data packet is sent regardless of whether an ACK was received. This can be used by interrupt IN endpoints which are used to communicate feedback for Isochronous endpoints.
		0 Disable.
		1 Enable.

17.6.18 USB_OUTMAXP: Maximum Package Size for OUT Register

OUTMAXP holds the maximum packet size for transactions through the currently selected OUT endpoint - in units of 8 bytes, except that a value of 128 sets the maximum packet size to 1023 (the maximum size for an isochronous packet) rather than 1024. When setting this value, the constraints imposed by the USB Specification on packet sizes for Bulk, Interrupt, and Isochronous transactions in Full-speed operation should be considered.

There is an OUTMAXP register for each OUT endpoint (except Endpoint 0).

The value written to this register should match the MaxPacketSize field of the Standard Endpoint Descriptor for the associated endpoint. A mismatch could cause unexpected results.

The Total amount of data represented by the value written to this register must not exceed the FIFO size for the OUT endpoint and should not exceed half the FIFO size if double-buffering is required. If a value greater than the configured OUT FIFO size for the endpoint is written to this register, the value will be automatically changed to the OUT FIFO size. If the value written to this register is less than, or equal to, half the OUT FIFO size, two OUT packets can be buffered.

USB_OUTMAXP=0x4100_605C

31 30 29 28 27 26 25 24	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			OUTMAXPS[7:0]
-			0x00
-			RW

7 0	OUTMAXPS[7:0]	Maximum Packet Size / transaction

17.6.19 USB_OUTCSR1: Control Status Register 1 for OUT Register

OUTCSR1 provides control and status bits for OUT transactions through the currently selected endpoint. It is reset to 0.

USB_OUTCSR1=0x4100_6060

31 30 29 28 27 26 25 24	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			CLRDATATOG	SENTSTALL	SENDSTALL	FLUSHFIFO	DATAERROR	OVERRUN	FIFOFULL	OUTPKTRDY
-			0	0	0	0	0	0	0	0
-			WO	RC	RW	WO	RO	RC	RO	RC

7	CLRDATATOG	Reset the Endpoint data toggle to '0'
	0	Auto clear by USB
	1	Reset Data toggle
6	SENTSTALL	Detect STALL handshake
	0	Clear by CPU
	1	Received STALL
5	SENDSTALL	Issue a STALL handshake. Ignored in ISO mode
	0	Clear by CPU to terminate the stall condition
	1	transmit STALL
4	FLUSHFIFO	Flush the next packet to be read from the endpoint OUT FIFO. **NOTE:** 1. if the FIFO contains two packets, FlushFIFO will need to be set twice to completely clear the FIFO
	0	Self-clearing
	1	Flush
3	DATAERROR	Set when OutPktRdy is set if the data packet has a CRC or bit-stuff error. This bit is only valid in ISO mode.
	0	Clear when OutPktRdy is cleared.
	1	Detect CRC or bit-stuff error in ISO mode
2	OVERRUN	Set if an OUT packet cannot be loaded into the OUT FIFO. The CPU should clear this bit. The bit is only valid is ISO mode
	0	Clear by CPU
	1	Over OUT FIFO in ISO mode
1	FIFOFULL	Set to '1' when no more packets can be loaded into the OUT FIFO
	0	Self-clearing
	1	Full OUT FIFO
0	OUTPKTRDY	Set when a data packet has been received. The CPU should clear this bit when the packet has been unloaded from the OUT FIFO. An interrupt is generated when the bit is set.
	0	Clear by CPU.
	1	Received data packet.

17.6.20 USB_OUTCSR2: Control Status Register 2 for OUT Register

OUTCSR2 provides control and status bits for OUT transactions through the currently selected endpoint. It is reset to 0.

USB_OUTCSR2=0x4100_6064

31 30 29 28 27 26 25 24	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			AUTOCLEAR	ISO	DMAENAB	DMAMODE	Reserved
-			0	0	0	0	-
-			RW	RW	RW	RW	-

7	AUTOCLEAR	If the CPU sets this bit then the OutPktRdy bit will be automatically cleared when a packet of OutMaxP bytes has been unloaded from the OUT FIFO. When packets of less than the maximum packet size are unloaded, OutPktRdy will have to be cleared manually.	
		0	Disables.
		1	Enables Auto clear OutPktRdy.
6	ISO	Enable the OUT endpoint for Isochronous transfers and clears it to enable the OUT endpoint for Bulk/Interrupt transfers.	
		0	Disable.
		1	Enable ISO mode for OUT.
5	DMAENAB	Enable the DMA request for the OUT endpoints	
		0	Disables.
		1	Enables DMA
4	DMAMODE	Two modes of DMA operation are supported like below. DMA mode 0 in which a DMA request is generated for all received packets, together with an interrupt (if enabled). DMA mode 1 in which a DMA request (but no interrupt) is generated for OUT packets of size OutMaxP bytes, and an interrupt (but no DMA request) is generated for OUT packets of any other size.	
		0	DMA mode 0 (every packet)
		1	DMA mode 1 (every OutMaxP)
		0	Disable
		1	Enable

17.6.21 USB_COUNT0: Number of Bytes for EP0 Register

COUNT0 indicates the number of data bytes received in the Endpoint 0 FIFO. The value returned is valid while OutPktRdy(INCSR0[0]) is set.

USB_COUNT0=0x4100_6068

31 30 29 28 27 26 25 24	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			EP0OUTCNT[6:0]
-			0 0 0 0 0 0 0
-			R0

6 0	EP0OUTCNT[6:0]	Endpoint 0 Out Count

17.6.22 USB_OUTCOUNT1: Number of Bytes for Lower Byte Register

OUTCOUNT1 holds the lower eight bits of the number of received data bytes in the packet in the FIFO associated with the currently selected OUT endpoint. The value returned is valid while OutPktRdy is set.

USB_OUTCOUNT1=0x4100_6068

31 30 29 28 27 26 25 24	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			EPOUTCNT[7:0]
-			0x00
-			R0

c

7 0	EPOUTCNT[7:0]	Endpoint Out count – lower eight bits

17.6.23 USB_OUTCOUNT2: Number of Bytes for Upper Byte Register

OUTCOUNT2 holds the upper three bits of the number of received data bytes in the packet in the FIFO associated with the currently selected OUT endpoint. The value returned is valid while OutPktRdy is set.

USB_OUTCOUNT2=0x4100_606C

31 30 29 28 27 26 25 24	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				EPOUTCNT[2:0]		
-				0	0	0
-				RO		

c

2 0	EPOUTCNT[2:0]	Endpoint Out count – upper three bits

17.6.24 USB_FIFOn: FIFO for EP0 to EP5 Register

This address range provides six addresses for CPU access to the FIFOs for each endpoint. Writing to these addresses loads data into the IN FIFO for the corresponding endpoint. Reading from these addresses' unloads data from the OUT FIFO for the corresponding endpoint.

USB_FIFO0=0x4100_6080, USB_FIFO1=0x4100_6090, USB_FIFO2=0x4100_60A0
USB_FIFO3=0x4100_60B0, USB_FIFO4=0x4100_60C0, USB_FIFO5=0x4100_60D0

31 30 29 28 27 26 25 24	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			FIFO[7:0]
-			0x00
-			RW

c

7 0	FIFO[7:0]	IN/OUT FIFO for each endpoint n

17.6.25 USB_DMAINTR: USB DMA Interrupt Pending Register

USB_DMAINTR indicates whether the USB DMA x interrupt is currently active.

USB_INTROUT1=0x4100_6200

31 30 29 28 27 26 25 24	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				UDMA2IFG	UDMA1IFG
-				0	0
-				R0	R0

1 0	UDMAxIFG	USB DMA x Interrupt flag bit x: 1 and 2

17.6.26 USB_DMA1CNTL: USB DMA Channel 1 Control Register

USB_DMA1CNTL provides control and status bits for DMA channel 1.

USB_DMAINTR=0x4100_6204

31 30 29 28 27 26 25 24	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		BUSERR	MAXPS[6:0]	EPNO[3:0]	DMAIEN	MODE	DIR	DMAENAB
-		0	0 0 0 0 0 0 0	0 0 0 0	0	0	0	0
-		RO	RW	RW	RW	RW	RW	RW

15	BUSERR	BUS Error bit
		0 No bus error occurred
		1 Bus error occurred. The DMA controller will immediately terminate the DMA transfer and interrupt the processor with this bit set.
14 8	MAXPS[6:0]	DMA 1 Maximum Packet Size in units of 8 bytes. Required for Mode 1 only
7 4	EPNO[3:0]	DMA 1 Endpoint Number bits
3	DMAIEN	DMA 1 Interrupt Enable bit
		0 Disable
		1 Enable
2	MODE	DMA 1 Interrupt Enable bit
		0 DMA Mode 0 (DMA requests every packet)
		1 DMA Mode 1 (DMA requests every OutMaxP)
1	DIR	DMA 1 Transfer Direction Select bit
		0 OUT endpoint. Transfer is from FIFO to memory through DMA buffer
		1 IN endpoint. Transfer is from memory to FIFO through DMA buffer
0	DMAENAB	DMA 1 Enable bit
		0 Disable
		1 Enable

17.6.27 USB_DMA1ADDR: USB DMA Channel 1 Memory Address Register

USB_DMA1ADDR provides Memory Address for DMA channel 1.

USB_DMA1ADDR=0x4100_6208

31 30 29 28 27 26 25 24	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]			
0x0000_0000			
RW			

c

31 0	ADDR[31:0]	DMA Channel 1 Memory Address bits

17.6.28 USB_DMA1COUNT: USB DMA Channel 1 Transfer Byte Count Register

USB_DMA1COUNT provides Transfer byte count value for DMA channel 1.

USB_DMA1COUNT=0x4100_620C

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

c

31 0	TRSCNT[31:0]	DMA Channel 1 Transfer Byte Count bits

17.6.29 USB_DMA2CNTL: USB DMA Channel 2 Control Register

USB_DMA2CNTL provides control and status bits for DMA channel 1.

USB_DMA2CNTL=0x4100_6214

31 30 29 28 27 26 25 24	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		BUSERR	MAXPS[6:0]	EPNO[3:0]	DMAIEN	MODE	DIR	DMAENAB
-		0	0 0 0 0 0 0 0	0 0 0 0	0	0	0	0
-		RO	RW	RW	RW	RW	RW	RW

15	BUSERR	BUS Error bit
		0 No bus error occurred
		1 Bus error occurred. The DMA controller will immediately terminate the DMA transfer and interrupt the processor with this bit set.
14 8	MAXPS[6:0]	DMA 2 Maximum Packet Size in units of 8 bytes. Required for Mode 1 only
7 3	EPNO[3:0]	DMA 2 Endpoint Number bits
3	DMAIEN	DMA 2 Interrupt Enable bit
		0 Disable
		1 Enable
2	MODE	DMA 2 Interrupt Enable bit
		0 DMA Mode 0 (DMA requests every packet)
		1 DMA Mode 1 (DMA requests every OutMaxP)
1	DIR	DMA 1 Transfer Direction Select bit
		0 OUT endpoint. Transfer is from FIFO to memory through DMA buffer
		1 IN endpoint. Transfer is from memory to FIFO through DMA buffer
0	DMAENAB	DMA 1 Enable bit
		0 Disable
		1 Enable

17.6.30 USB_DMA2ADDR: USB DMA Channel 2 Memory Address Register

USB_DMA2ADDR provides Memory Address for DMA channel 2.

USB_DMA2ADDR=0x4100_6218

31 30 29 28 27 26 25 24	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]			
0x0000_0000			
RW			

c

31 0	ADDR[31:0]	DMA Channel 2 Memory Address bits

17.6.31 USB_DMA2COUNT: USB DMA Channel 2 Transfer Byte Count Register

USB_DMA2COUNT provides Transfer byte count value for DMA channel 2.

USB_DMA1COUNT=0x4100_621C

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

c

31 0	TRSCNT[31:0]	DMA Channel 2 Transfer Byte Count bits

17.6.32 USB Register Map Summary

Table 128. USB 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	USB_FADDR	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	UPDATE	FADDR[6:0]						
	Reset value																									0	0	0	0	0	0	0	0
0x04	USB_POWER	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	ISOUPDATE				RESET	RESUME	SUSPENDMD	SUSPENDEN
	Reset value																									0				0	0	0	0
0x08	USB_INTRIN1	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.			EP5IFG	EP4IFG	EP3IFG	EP2IFG	EP1IFG	EP0IFG
	Reset value																											0	0	0	0	0	0
0x10	USB_INTROUT1	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.			EP5IFG	EP4IFG	EP3IFG	EP2IFG	EP1IFG	
	Reset value																											0	0	0	0	0	
0x18	USB_INTRUSB	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.					SOFIFG	RESETIFG	RESUMEIFG	SUSPENDIF
	Reset value																													0	0	0	0
0x1C	USB_INTRIN1E	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.			EP5IEN	EP4IEN	EP3IEN	EP2IEN	EP1IEN	EP0IEN
	Reset value																											1	1	1	1	1	1
0x24	USB_INTROUT1E	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.			EP5IEN	EP4IEN	EP3IEN	EP2IEN	EP1IEN	
	Reset value																											1	1	1	1	1	
0x2C	USB_INTRUSBE	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.					SOFIEN	RESETIEN	RESUMEIEN	SUSPENDIEN
	Reset value																													0	1	1	0

Table 128. USB 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	USB_FRAME1	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	FRMNO[7:0]							
	Reset value																									0	0	0	0	0	0	0	0
0x34	USB_FRAME2	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.						FRMNO[2:0]		
	Reset value																														0	0	0
0x38	USB_INDEX	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.					INDEX[3:0]			
	Reset value																													0	0	0	0
0x40	USB_PUPDCR	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.			DNPDEN	DNPUEN			DPPDEN	DPPUEN
	Reset value																											0	0			0	0
0x44	USB_PHYCR	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.								PHYSUSP_MODE
	Reset value																																1
0x50	USB_INMAXP	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	INMAXPS[7:0]							
	Reset value																									0	0	0	0	0	0	0	0
0x54	USB_CSR0	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.															SERVICEDSETUPEND	SERVICEDOUTPKTRDY	SENDSTALL	SETUPEND	DATAEND	SENTSTALL	INPKTRDY	OUTPKTRDY
	Reset value																									0	0	0	0	0	0	0	0
0x54	USB_INCSR1	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.																CLRDATATO	SENTSTALL	SENDSTALL	FLUSHFIFO	UNDERRUN	FIFONOTEM	INPKTRDY
	Reset value																										0	0	0	0	0	0	0
0x58	USB_INCSR2	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.															AUTOSET	ISO	MODE	DMAENAB	FRCDATATOG			
	Reset value																									0	0	1	0	0			

Table 128. USB 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
0x5C	USB_OUTMAXP	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	OUTMAXPS[7:0]							
	Reset value																									0	0	0	0	0	0	0	0
0x60	USB_OUTCSR1	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	CLRDATATO	SENTSTALL	SENDSTALL	FLUSHFIFO	DATAERROR	OVERRUN	FIFOFULL	OUTPKTRDY
	Reset value																									0	0	0	0	0	0	0	0
0x64	USB_OUTCSR2	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	AUTOCLEAR	ISO	DMAENAB	DMAMODE				
	Reset value																									0	0	0	0				
0x68	USB_COUNT0	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.		OUTMAXPS[6:0]						
	Reset value																										0	0	0	0	0	0	0
0x68	USB_OUTCOUNT1	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	EPOUTCNT[7:0]							
	Reset value																									0	0	0	0	0	0	0	0
0x6C	USB_OUTCOUNT2	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.						EPOUTCNT[2:0]		
	Reset value																														0	0	0
0x80	USB_FIFO0	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	FIFO[7:0]							
	Reset value																									0	0	0	0	0	0	0	0
0x90	USB_FIFO1	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	FIFO[7:0]							
	Reset value																									0	0	0	0	0	0	0	0
0xA0	USB_FIFO2	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	FIFO[7:0]							
	Reset value																									0	0	0	0	0	0	0	0
0xB0	USB_FIFO3	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	FIFO[7:0]							
	Reset value																									0	0	0	0	0	0	0	0

Table 128. USB 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
0xC0	USB_FIFO4	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	FIFO[7:0]							
	Reset value																									0	0	0	0	0	0	0	0
0xD0	USB_FIFO5	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	FIFO[7:0]							
	Reset value																									0	0	0	0	0	0	0	0
0x200	USB_DMAINTR	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.																					UDMA2IFG	UDMA1IFG
	Reset value																															0	0
0x204	USB_DMA1CNTL	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.							BUSERR	MAXPS[6:0]							EPNO[3:0]				DMAIEN	MODE	DIR	DMAENAB
	Reset value																	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
0x208	USB_DMA1ADDR	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
0x20C	USB_DMA1COUNT	TRSCNT[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
0x214	USB_DMA2CNTL	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.							BUSERR	MAXPS[6:0]							EPNO[3:0]				DMAIEN	MODE	DIR	DMAENAB
	Reset value																	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
0x218	USB_DMA2ADDR	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
0x21C	USB_DMA2COUNT	TRSCNT[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

18. LCD Driver/Controller

18.1 LCD Introduction

The LCD drives up to 8 common terminals and 44 segment terminals to drive up to 320 pixels.

LCD display can continue its operation even during SLEEP mode and DEEP-SLEEP mode if it uses a selected clock for LCD driver.

A clock and duty of the LCD driver is initialized by hardware whenever a value is written to the control register. So, it is recommended not to rewrite the LCD_CR frequently.

18.2 LCD Main Features

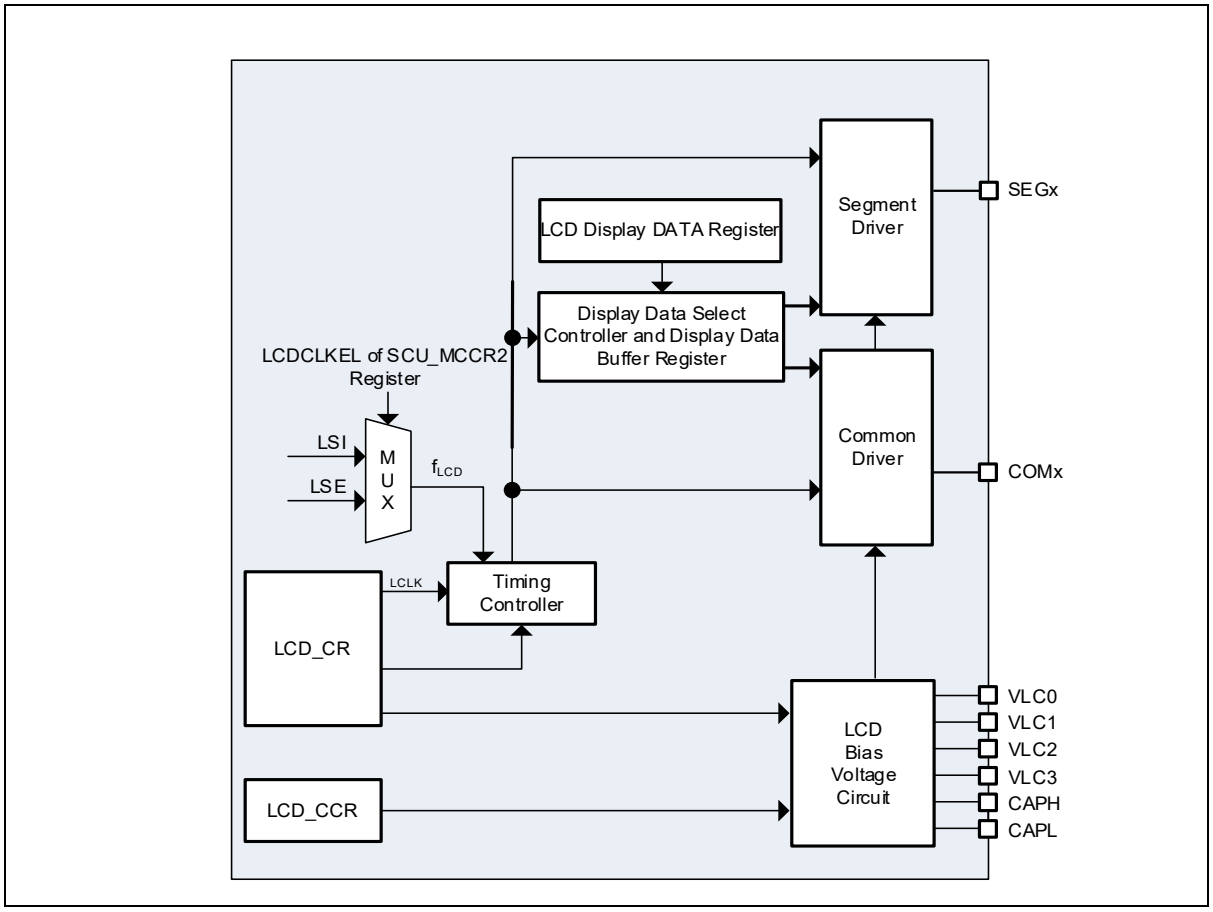
The LCD of the A34L716 series features the following:

- Supports resistor bias, capacitor bias.
- Supports 1/2, 1/3, 1/4, 1/5, 1/6 and 1/8 duty.
- Supports 1/2, 1/3, and 1/4 bias.

18.3 LCD Functional Description

18.3.1 LCD Block Diagram

Figure 145. LCD Block Diagram



18.3.2 LCD Pins and Signals

The external pins assigned for LCD are described in Table 129.

Table 129. Pin Assignment of LCD: External Pins

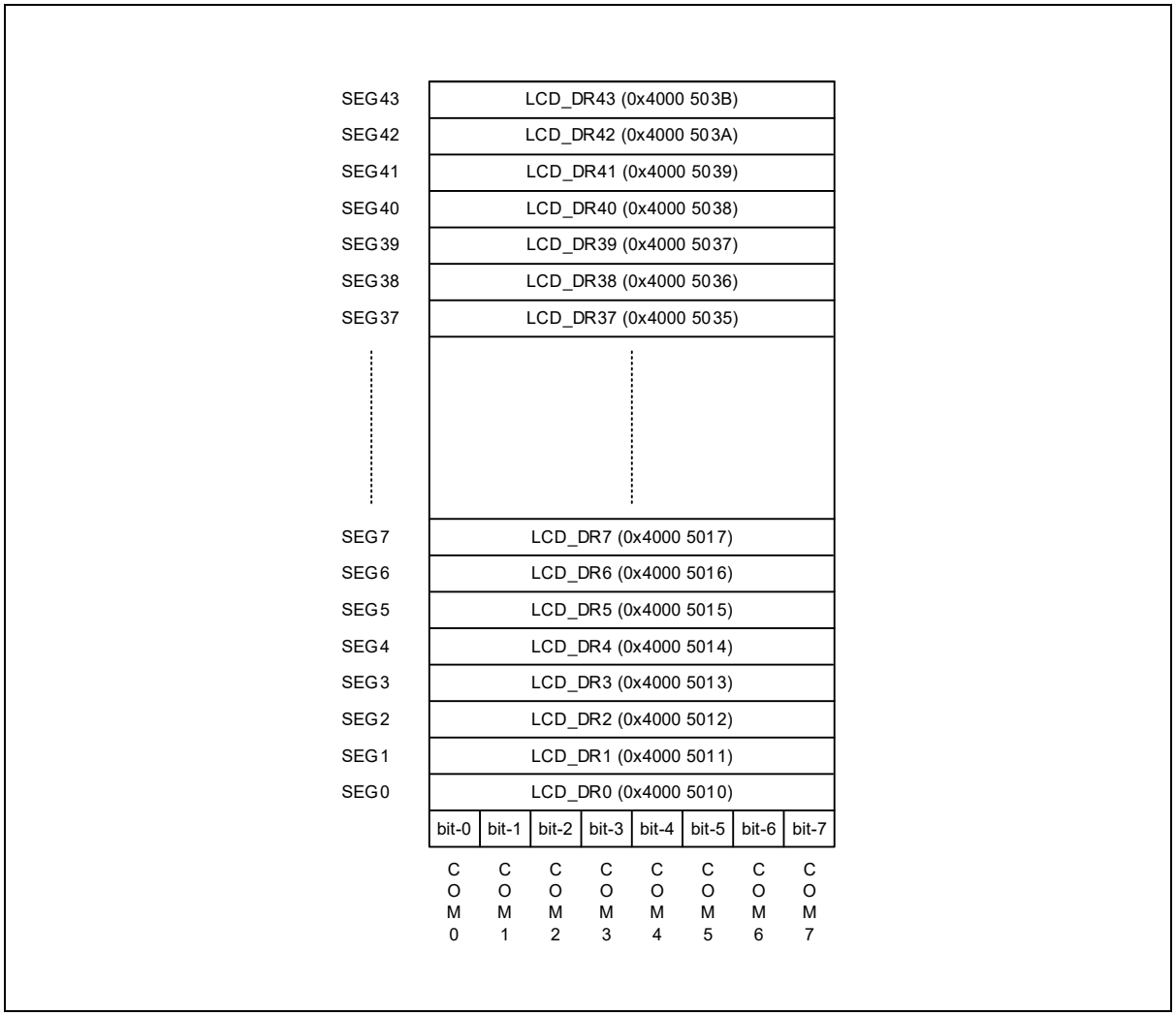
Pin name	Type	Description	Supported packages		
			A34L716 (100-LQFP)	A34L716 (64-LQFP)	A34L716 (56-QFN)
COM0 to COM7	O	LCD common signal outputs	O	O	O
VLCD0/1/2/3	I/O	LCD bias voltage input/output	O	O	O
CAPH/L	O	Capacitor terminal for voltage booster	O	O	O
SEG0	O	LCD segment signal outputs	O	O	O
SEG1			O	O	O
SEG2			O	O	O
SEG3			O	O	O
SEG4			O	O	O
SEG5			O	O	O
SEG6			O	O	O
SEG7			O	O	O
SEG8			O	O	O
SEG9			O	O	O
SEG10			O	O	O
SEG11			O	O	O
SEG12			O	O	O
SEG13			O	O	O
SEG14			O	O	X
SEG15			O	O	X
SEG16			O	O	X
SEG17			O	O	O
SEG18			O	O	O
SEG19			O	O	X
SEG20			O	O	O
SEG21			O	O	X
SEG22			O	O	X
SEG23 to SEG43			O	X	X

18.3.3 LCD Display RAM Organization

Display data are stored in display data area. The display data stored in the display data area (address 0x4000_5010-0x4000_5038) are read automatically and are sent to the LCD driver by hardware. The LCD driver generates the segment and common signals in accordance with the display data and driving method. Therefore, display patterns can be changed by simply overwriting the contents of the display data area with a program.

Figure 146 shows the correspondence between the display data area and the COM/SEG pins. The LCD is turned on when the display data is '1' and turned off when it is '0'.

Figure 146. LCD Display Ram Organization



18.3.4 LCD Signal Waveform

Figure 147. LCD Signal Waveforms (1/2-Duty, 1/2-Bias)

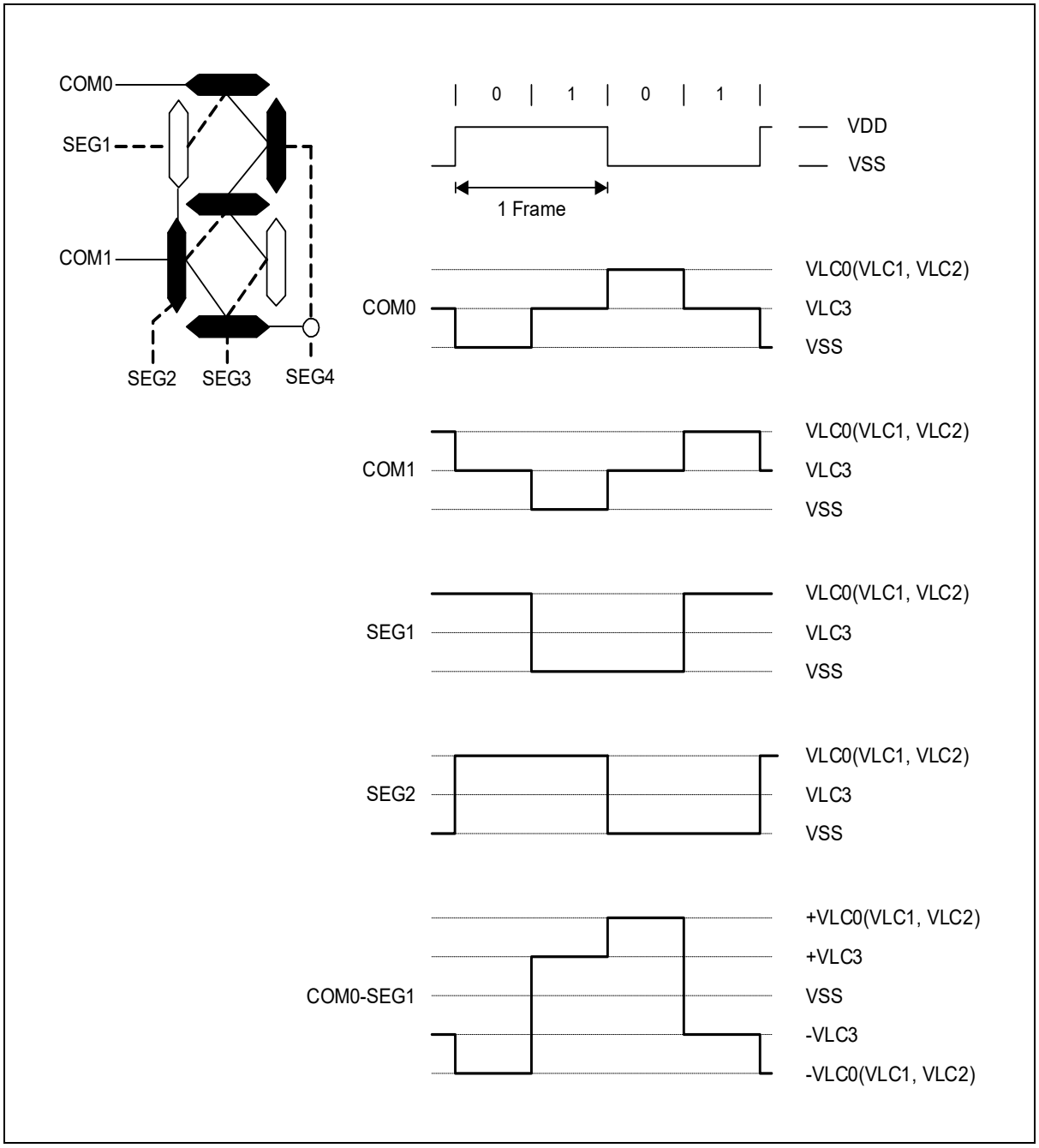


Figure 148. LCD Signal Waveforms (1/3-Duty, 1/3-Bias)

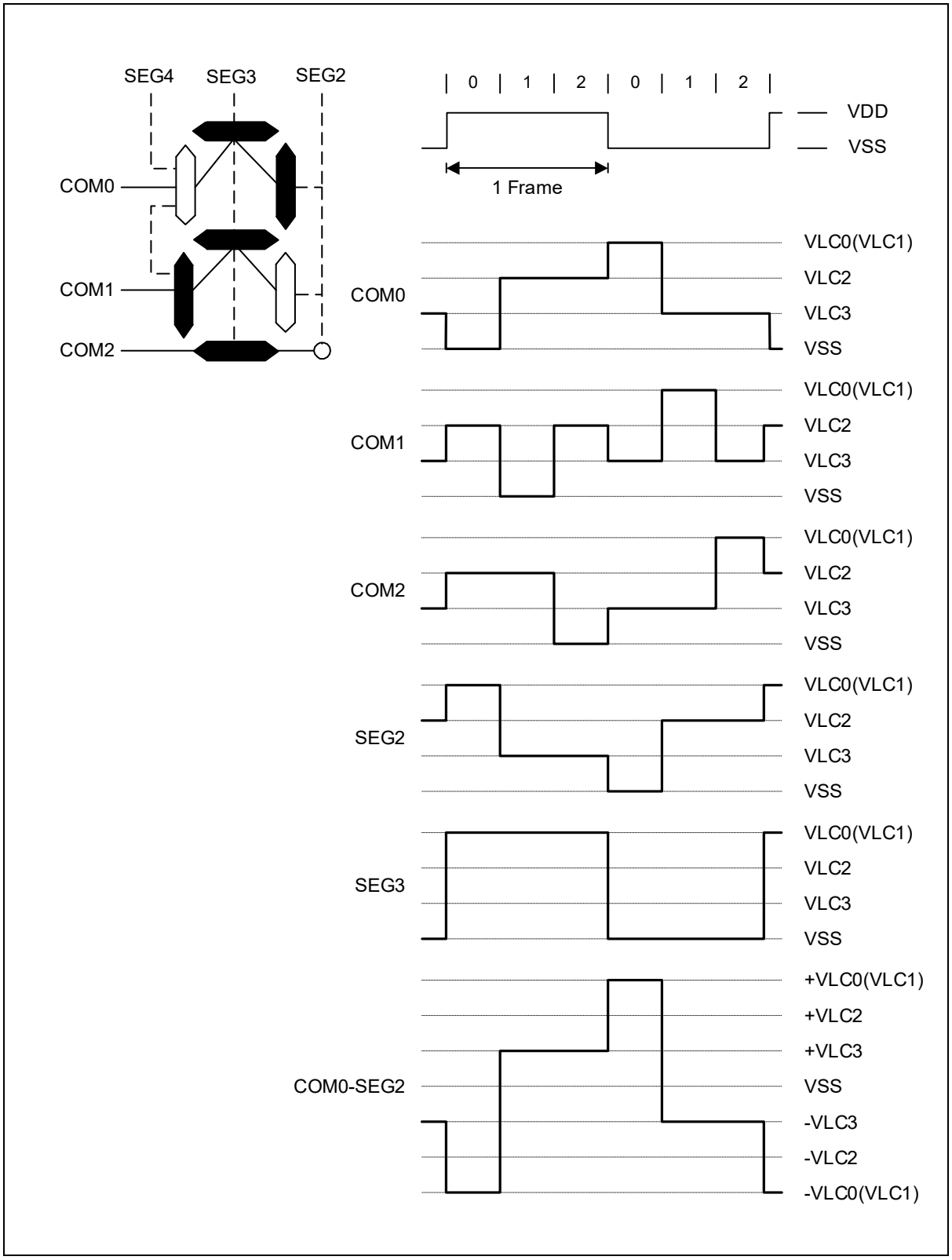


Figure 149. LCD Signal Waveforms (1/4-Duty, 1/3-Bias)

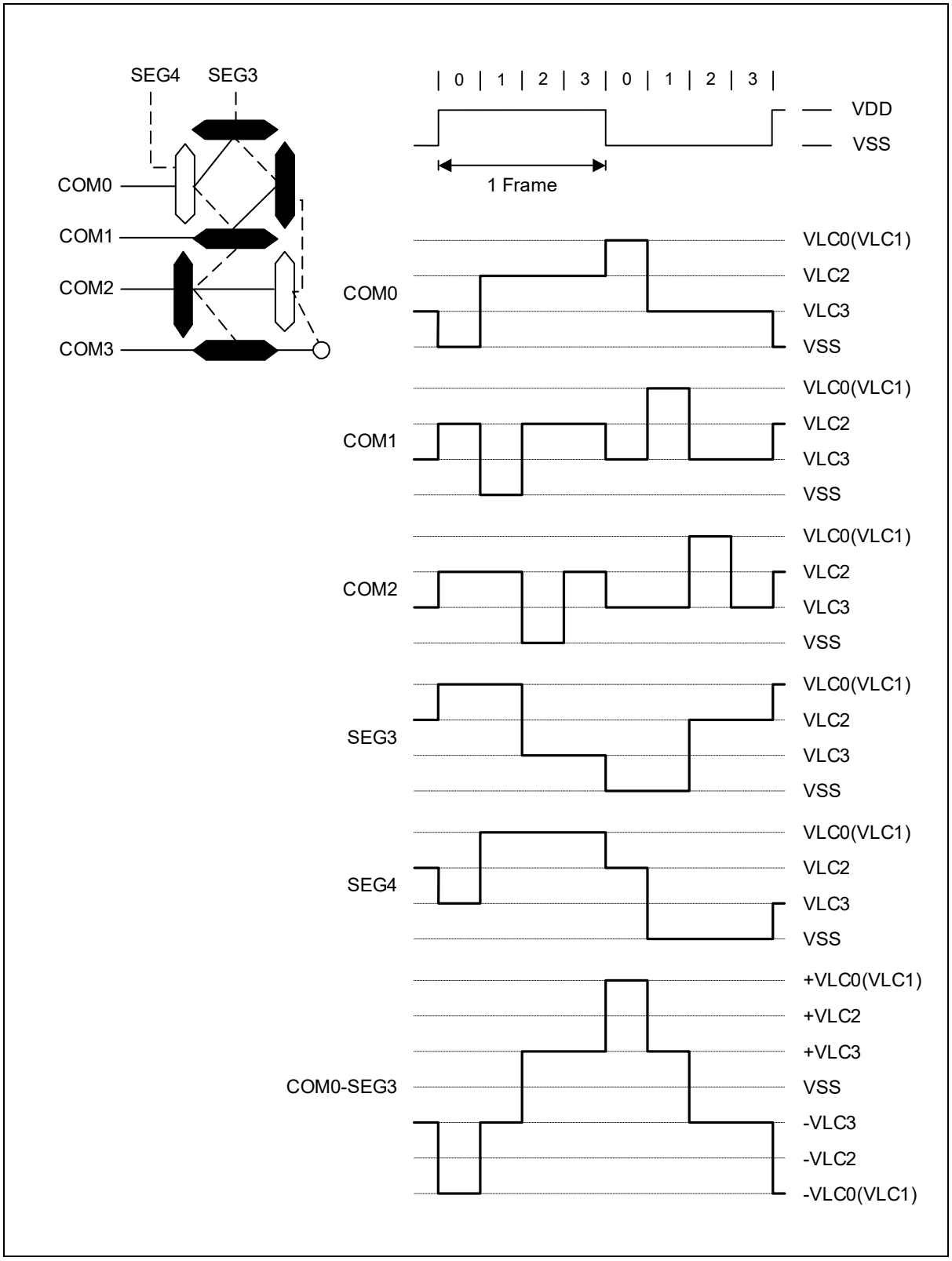
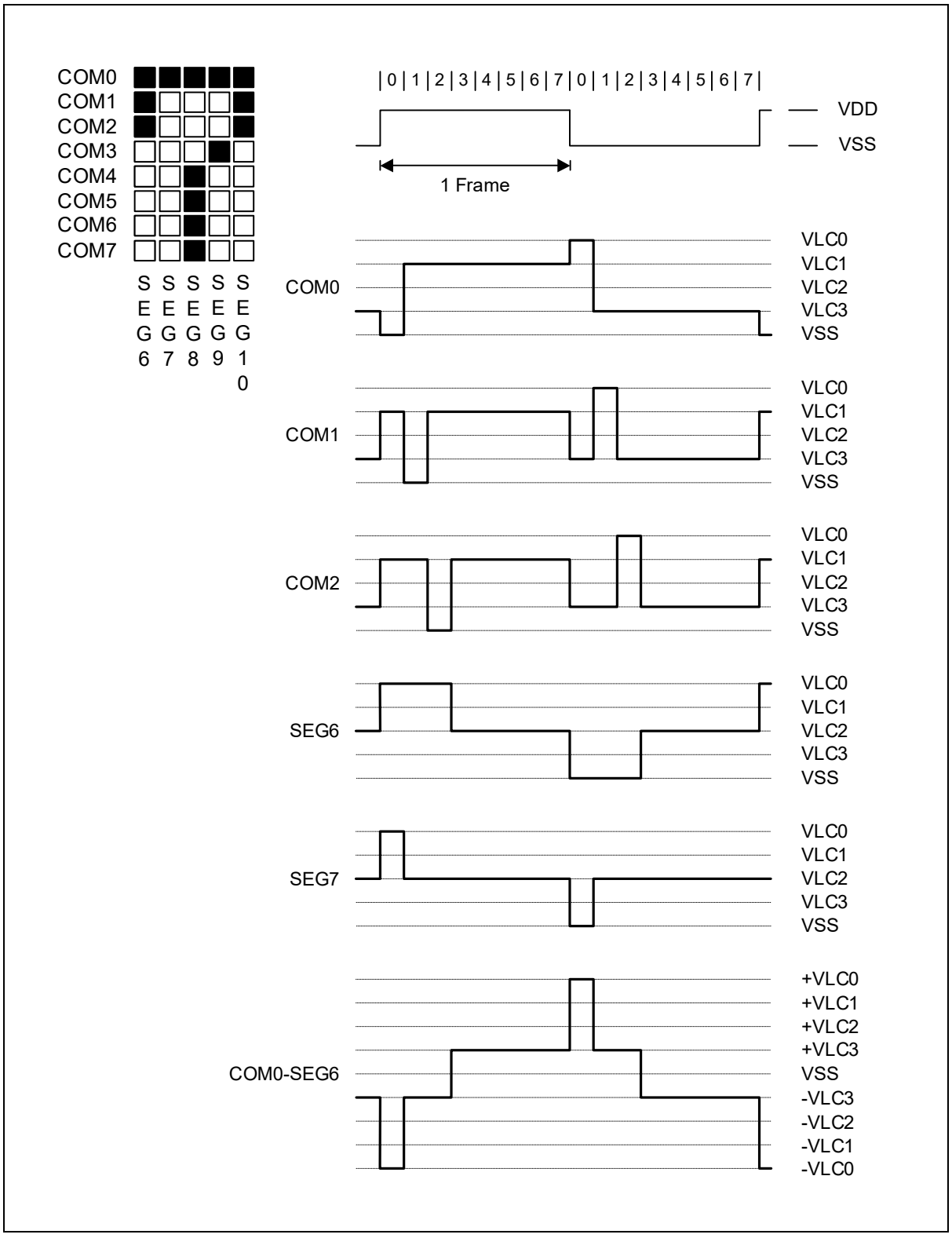
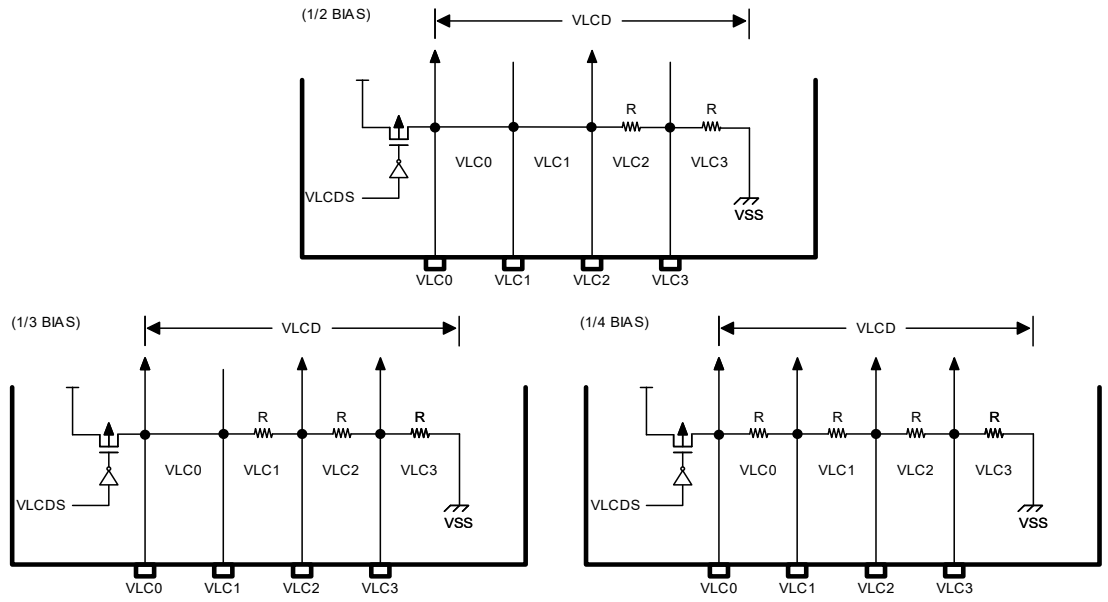


Figure 150. LCD Signal Waveforms (1/8-Duty, 1/4-Bias)



18.3.5 Internal Resistor Bias Connection

Figure 151. Internal Resistor Bias Connection

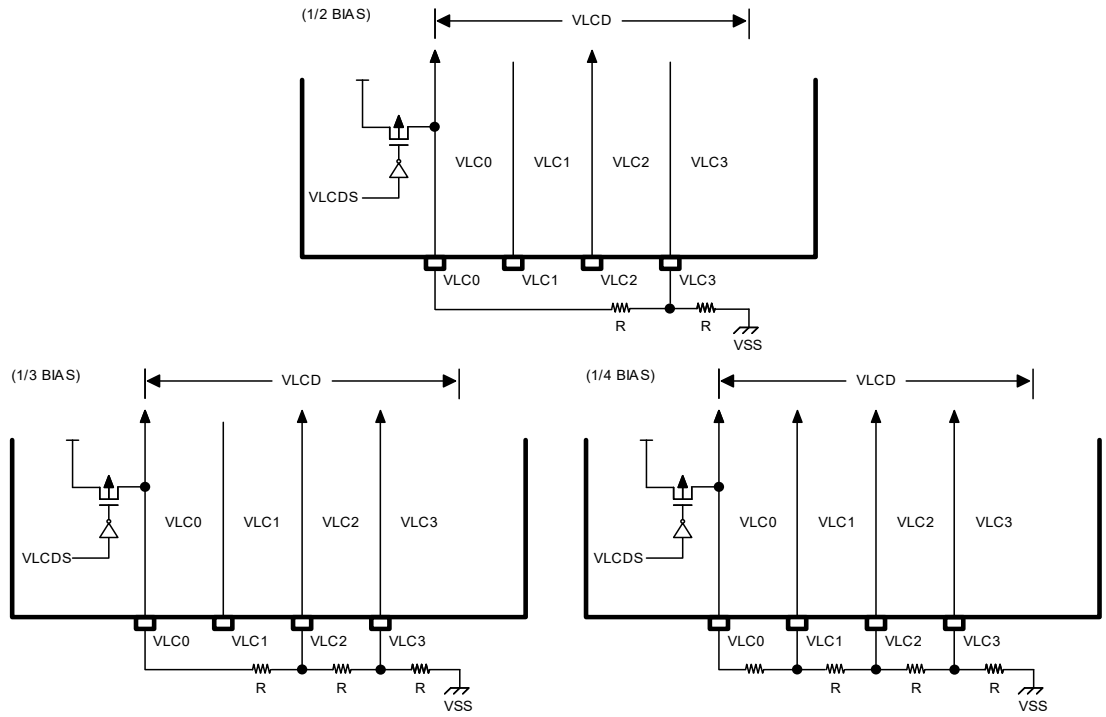


NOTES:

1. When 1/2 bias is selected, the VLC0/VLC1 and VLCD2 are internally connected like the above figure.
2. When 1/3 bias is selected, the VLC0 and VLC1 are internally connected.
3. When the internal resistors are selected, all the VLC0, VLC1, VLC2, VLC3, CAPH, and CAPL pins can be used for normal I/O.

18.3.6 External Resistor Bias Connection

Figure 152. External Resistor Bias Connection

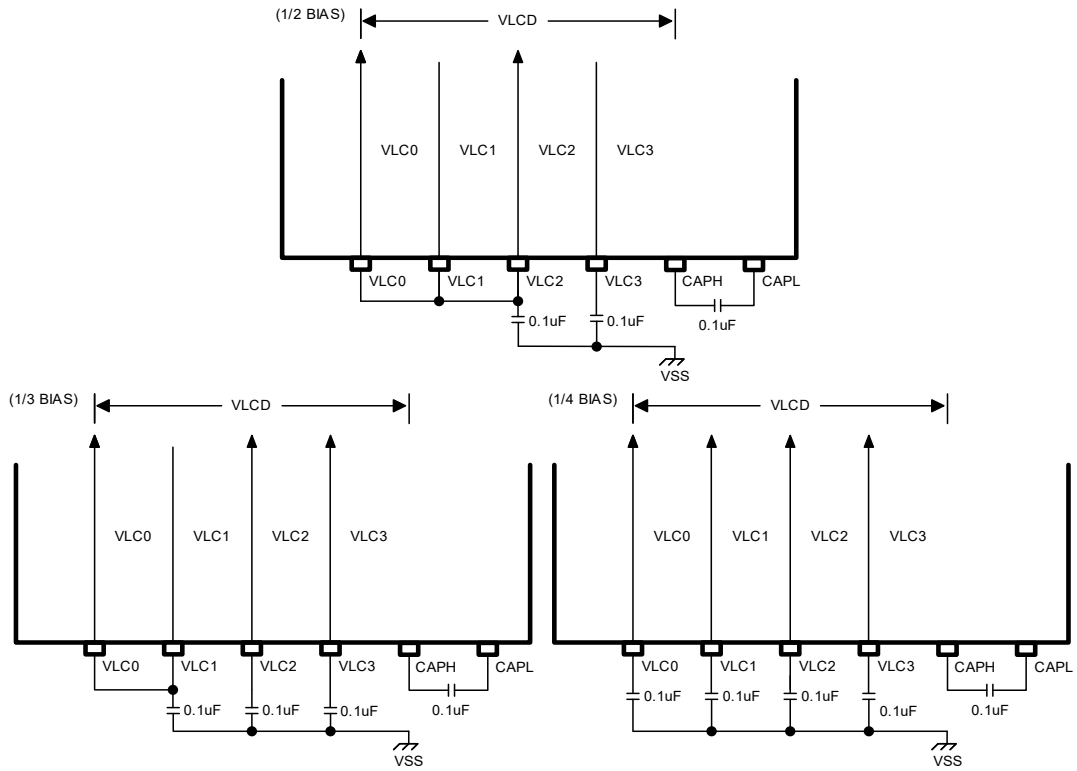


NOTES:

1. When the external resistor bias is selected, the internal resistors for bias are disconnected.
2. When the external resistor bias is selected, the dividing resistors should be connected like the above figure and the needed bias pins should be selected as the LCD bias function.
3. When it is 1/2 bias, the VLC0 and VLC3 pins should be selected as the LCD bias function. The other pins can be used for normal I/O.
4. When it is 1/3 bias, the VLC0, VLC2, and VLC3 pins should be selected as the LCD bias function. The other pins can be used for normal I/O.
5. When it is 1/4 bias, the VLC0, VLC1, VLC2, and VLC3 pins should be selected as the LCD bias function.

18.3.7 Capacitor Bias Connection

Figure 153. Capacitor Bias Connection



NOTE:

1. When the capacitor bias is selected for the VLCD voltage, all the VLC0, VLC1, VLC2, VLC3, CAPH, and CAPL pins should be selected for LCD bias function and should be connected like the above figure.

18.4 LCD Registers

The base address and register map of the LCD module are described in the following tables:

Table 130. Base Address of LCD

Name	Base Address
LCD	0x4000_5000

Table 131. LCD Register Map

Name	Offset	Type	Description	Reset Value	Reference
LCD_CR	0x0000	RW	LCD driver control register	0x0000_0000	18.4.1
LCD_CCR	0x0004	RW	LCD contrast control register	0x0000_0000	18.4.2
LCD_DR0 to LCD_DR43	0x0010 to 0x0038	RW	LCD display data register 0	0x0000_0000	18.4.3

18.4.1 LCD_CR: LCD Driver Control Register

LCD_CR register is 32-bit size and accessible in 32/16/8-bit.

LCD_CR=0x4000_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		Reserved	VLCDS	BTYPE[1:0]	IRSEL[1:0]	DBS[2:0]	LCLK[1:0]	DISP
-		-	0	0 0	0 0	0 0 0	0 0	0
-		-	RW	RW	RW	RW	RW	RW

10	VLCDS	Selection VLCD Power (only when a resistor bias type is selected)	
		0	Internal Power for VLCD
		1	External Power for VLCD
9 8	BTYPE[1:0] (1)	LCD Bias Type Selection	
		00	Internal Resistor Bias
		01	External Resistor Bias
		10	Capacitor Bias (Voltage Booster)
		11	Not available
7 6	IRSEL[1:0]	Internal LCD Bias Dividing Resistor Selection	
		00	RLCD3: 100 kΩ
		01	RLCD1: 10 kΩ
		10	RLCD2: 50 kΩ
		11	Not available.
5 3	DBS[2:0]	LCD Duty and Bias Selection	
		000	1/8 duty, 1/4 bias.
		001	1/6 duty, 1/4 bias.
		010	1/5 duty, 1/3 bias.
		011	1/4 duty, 1/3 bias.
		100	1/3 duty, 1/3 bias.
		101	1/3 duty, 1/2 bias
		110	1/2 duty, 1/2 bias
		Others	reserved.
	LCLK[1:0]	LCD Clock Selection (When fLCD = 32.768 kHz)	
		00	128 Hz.
		01	256 Hz.
		10	512 Hz.
		11	1024 Hz.
	DISP	LCD Display Control	
		0	Display off (The LCD block and voltage booster off).
		1	Normal display on (When the BTYPE[1:0] = “10”, the voltage booster is turn on).

NOTE:

1. All the VLC0 – VLC3, CAPH, and CAPL pins must be used as bias functions when the capacitor bias is selected for the LCD bias type.

18.4.2 LCD_CCR: LCD Contrast Control Register

LCD_CCR register is 32-bit size and accessible in 32/16/8-bit.

LCD_CR=0x4000_5004

31 30 29 28 27 26 25 24	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				VLCD[2:0]
-				0 0 0
.				RW

2 0	VLCD[2:0]	VLC3 Voltage Control when the capacitor bias is selected by the BTYPE[1:0]			
			1/2-bias	1/3-bias	1/4-bias
		000	1.23 V	0.96 V	0.77 V
		001	1.27 V	0.99 V	0.79 V
		010	1.30 V	1.03 V	0.82 V
		011	1.34 V	1.06 V	0.84 V
		100	1.37 V	1.10 V	0.86 V
		101	1.41 V	1.13 V	0.89 V
		110	1.44 V	1.17 V	0.91 V
		111	1.47 V	1.20 V	0.94 V

NOTE:

1. The VLC0 voltage can be calculated by the below formulas.
   VLC0 = VLC3 × 2 when 1/2 bias
   VLC0 = VLC3 × 3 when 1/3 bias
   VLC0 = VLC3 × 4 when 1/4 bias

18.4.3 LCD_DRx: LCD Display Data Register

LCD_DRx register is 8-bit size and accessible in 32/16/8-bit. (n = 0 to 10)

LCD_DRx=0x4000_5010 to 0x4000_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
LCD_DR(4×n+3) [7:0]	LCD_DR(4×n+2) [7:0]	LCD_DR(4×n+1) [7:0]	LCD_DR(4×n+0) [7:0]
0x00	0x00	0x00	0x00
RW	RW	RW	RW

31 24	LCD_DR(4×n+3)	LCD Display Data
23 16	LCD_DR(4×n+2)	LCD Display Data
15 8	LCD_DR (4×n+1)	LCD Display Data
7 0	LCD_DR(4×n+0)	LCD Display Data

18.4.4 LCD Register Map Summary

Table 132. LCD 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	LCD_CR	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	VLCDS	BTYPE[1:0]		IRSEL[1:0]		DBS[2:0]			LCK[1:0]		DISP
	Reset value																						0	0	0	0	0	0	0	0	0	0	0
0x04	LCD_CCR	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.	VLCD[2:0]		
	Reset value																														0	0	0
0x10-0x38	LCD_DRx	LCD_DR4xn+3)[7:0]								LCD_DR(4xn+2)[7:0]								LCD_DR(4xn+1)[7:0]								LCD_DR(4xn+0)[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

19. ADC (Analog-to-Digital Converter)

19.1 ADC Introduction

The A34L716 series features a 12-bit resolution Analog-to-Digital Converter (ADC) module. It consists of two independently controllable units and supports up to 24 different analog inputs.

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 times, which is required for various applications and can be achieved using the triggering functionality in association with TIMER.

Once the A/D conversion is complete, the result of the ADC is stored in a left-aligned 16-bit data register.

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’s 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.

19.2 ADC Main Features

Main features of the ADC module are listed below:

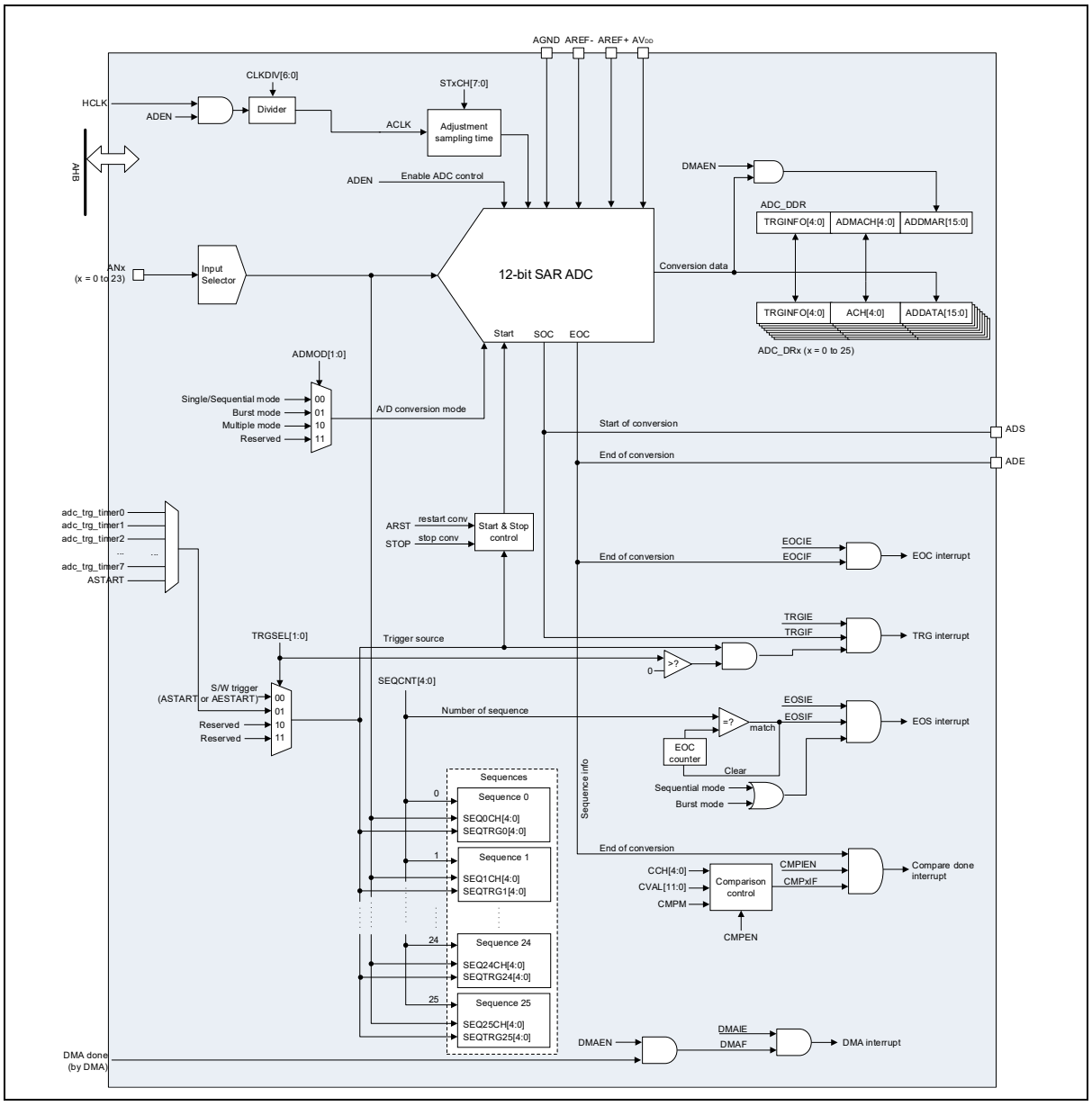
- Composed of two independent ADC units.
- 12-bit analog to digital resolution
- Manage single-ended or differential inputs.
- 1.0 Msps maximum conversion rate with full resolution
- 1.8 V minimum to 3.6 V operating voltage range
- Data can be managed by DMA for high-performance conversion.
- Analog Input channels
  - ADC is connected to 24 external channels.
- Internally dedicated channels
  - AREF+ internal reference voltage outputs connected to ADC.
- Flexible sampling time control
- Five conversion modes
  - Single mode
  - Sequential mode
  - Multiple mode
  - Burst mode
    - In Burst mode, support for Continuous mode
- Up to 26 sequential conversions can be supported.
- Software trigger sources per channel
  - Internal software trigger source
  - TIMERn (n = 0, 1, 2, 3, 4, 5, 6, 7) 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: GND ≤ ANx (x = 0 to 23) ≤ AREF+ ≤ A_{VDD} = V_{DD}
  - AV_{DD} must always be supplied at the exact same voltage level as V_{DD} without exception. (AV_{DD} = V_{DD})

19.3 ADC Functional Description

19.3.1 ADC Block Diagram

Figure 154 shows a block diagram of the 12-bit ADC.

Figure 154. 12-bit ADC Block Diagram



19.3.2 ADC Pins and Signals

Table 133 describes the pins assigned for the ADC.

Table 133. Pin Assignment of ADC External Pins

Pin Name	Signal Type(1)	Description	Supported Packages		
			A34L716 (100-LQFP)	A34L716 (64-LQFP)	A34L716 (56-QFN)
AREF+	P	Input, analog reference positive The higher/positive reference voltage for the DAC, AREF+ ≤ AV_{DD} (refer to datasheet)	O	X	X
AREF-	P	Input, analog reference negative AREF- = AGND (refer to datasheet)	O	X	X
AV_{DD}	P	Input, analog supply AV_{DD} = V_{DD} (refer to datasheet)	O	O	O
AGND	P	Input, analog supply ground	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 /ANN0	A	ADC external analog input 16 ADC negative external analog input	O	O	O
AN17	A	ADC external analog input 17	O	X	X
AN18	A	ADC external analog input 18	O	X	X
AN19	A	ADC external analog input 19	O	X	X
AN20	A	ADC external analog input 20	O	X	X
AN21	A	ADC external analog input 21	O	X	X
AN22	A	ADC external analog input 22	O	X	X
AN23	A	ADC external analog input 23	O	X	X

NOTE:

1. P: Power, O: Output, A: Analog Input.

Table 134 describes the input and output signals for the ADC.

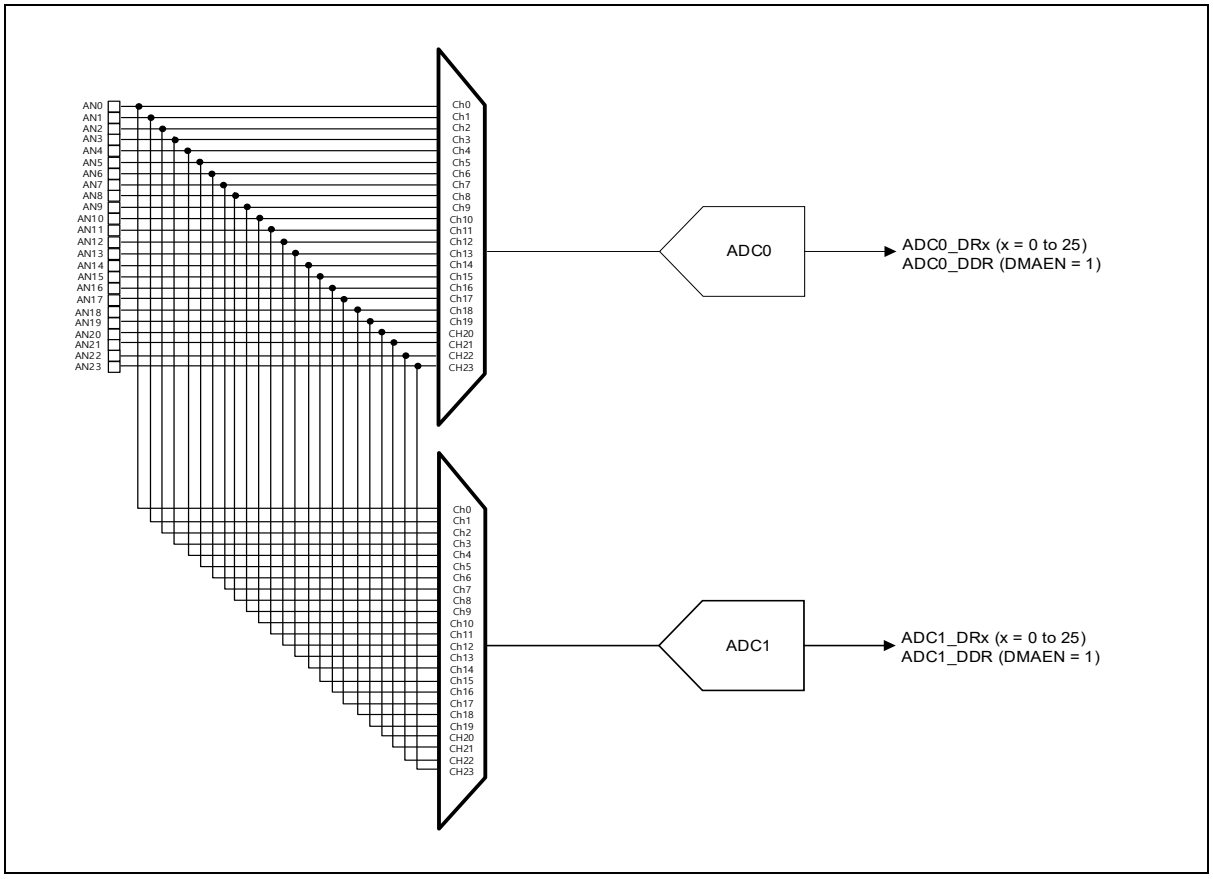
Table 134. ADC Input/Output Signals

Signal Name	Signal Type	Description
ACLK	clock	ADC operating clock
adc_trg_timerx	Inputs	The internal trigger source of TIMERn (n = 0 to 7)
adc_trg_astart	Inputs	The internal clock of ADC
adc_ch[23:0]	Analog Input	ADC external input signal
adc_out	Digital Output	A/D Conversion data by ADC

19.3.3 ADC Connectivity

19.3.3.1 ADC Internal Channel Wiring

Figure 155. 12-bit ADC Internal Channel Wiring



19.3.4 ADC Clocks (ACLK)

19.3.4.1 ADC Module Control

Before enabling the ADCn clock, users must select an ADCn unit to be enabled by setting a bit of the ADCn in SCU_PER2 register to '1' which corresponds to the ADCn unit. Then, enable the clock of the selected ADCn unit by setting a bit of the ADCn in SCU_PCER2 register to '1' which corresponds to the clock of the selected ADCn unit.

To initialize the registers of each ADCn unit, set each corresponding bit of the ADCn in SCU_PER2 register and the ADCn in SCU_PCER2 register to '0' and then set back to '1'.

Each bit of the ADCn in SCU_PER2 register controls the corresponding ADCn unit. To enable or disable all ADCn units at a time, set the ADCn in SCU_PCER2 register to the number shown below:

- If the ADCn in SCU_PER2 register = '0', ADCn unit is disabled.
- If the ADCn in SCU_PER2 register = '1', ADCn unit is enabled.

To provide the ADCn unit with the ADCn clock, configure the ADCn in SCU_PCER2 register referring to the settings shown below:

- If the ADCn in SCU_PCER2 register = '0', ADCn clock is disabled.
- If the ADCn in SCU_PCER2 register = '1', ADCn clock is enabled.

19.3.4.2 ADC Clock Sources

The HCLK operates as the ADC operation clock (ACLK) based on the AHB domain.

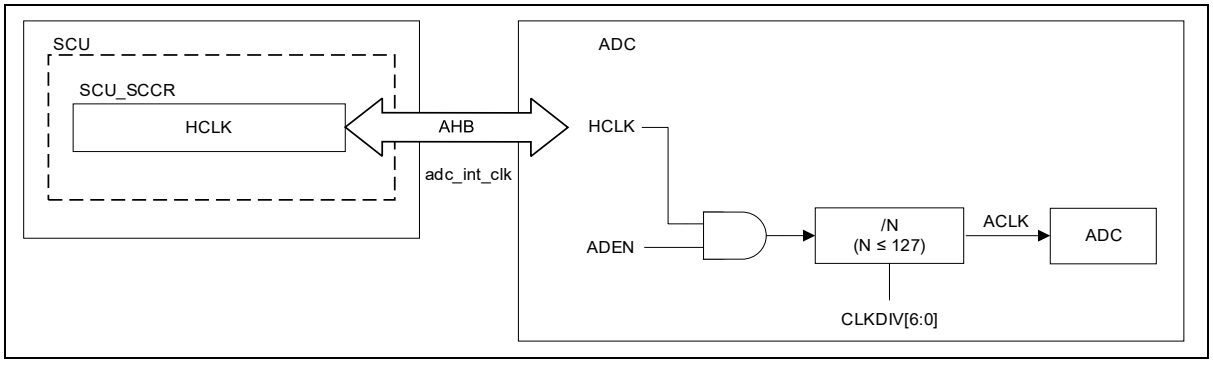
Internal Clock Source

The programmable divider coefficient can be selected to range from 0 to 127 by configuring the CLKDIV[6:0](1) in ADCn_CCR register, and the HCLK is divided by the coefficient to be used as the ADCn operation clock.

NOTE:

1. The CLKDIV[6:0] in ADCn_CCR register is used only when the HCLK, the ADC's internal system clock is used.

Figure 156. Block Diagram of ADC Clock



19.3.5 ADC On and Off Control (ADEN)

Before starting the A/D conversion, users must enable the ADCn module using the ADEN in ADCn_MR register.

- Setting the ADEN = '1' enables the ADCn unit. When the ADCn unit is enabled, the ADCn functionality and the ASTART bit control are possible.
- Setting the ADEN = '0' disables the ADCn unit. When the ADCn unit is disabled, the A/D conversion cannot be performed.

VREFBUF also needs to be set up before using the ADC. For information on how to set up VREFBUF, see section 20.2 VREFBUF Functional Description.

19.3.5.1 Software Procedure to Enable ADC

1. Clear the EOCIF in ADCn_SR register by writing '1'.
2. Setting the ADEN in ADCn_MR register to '1' enables the ADCn module.

19.3.5.2 Software Procedure to Disable ADC

1. Confirm that the ASTART and the ASTOP in ADCn_CR register are set to '0', respectively. Th is ensures that no conversion is ongoing. If required, set the ASTOP to '1' to stop the convers ion and then wait until the ASTOP bit becomes '0'(1).
2. Disable the ADCn module by setting the ADEN in ADCn_MR register to '0.'

NOTE:

1. This ASTOP bit is auto cleared at the next ADC clock cycle.

19.3.6 Constraints when Writing ADC Control Bits

19.3.6.1 When Writing ADC Control Bit

While accessing the SCU_PER2, SCU_PCER2, and ADCn_CCR registers to enable the ADC operation clock source and ADC module, the ADC module must be disabled. For this, set the ADEN in ADCn_MR register to '0'.

During the A/D conversion operations, if users write to each ADCn control bit of the SCU_PER2, SCU_PCER2, and ADCn_CCR registers, the ADC clock bus and ADCn registers are not controllable and become Unknown state.

Software can write to the control bits configuring the A/D conversion, only when the ADCn module is enabled (ADEN = '1') and the A/D conversion is not performed (e.g., ASTART = '0').

- Set the ADEN in ADCn_MR register to '0'.
- Clear all bits in ADCn_CR register.

19.3.7 Channel Selection

19.3.7.1 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 the mode of the desired channel to ADC using the Pn_MR1 or Pn_MR2 registers.
2. Set the pin direction to ‘input’ using the Pn_CR register.
3. Disable the internal Pull-up or Pull-down resistance functionality using the Pn_PRCR register.

NOTE:

1. n = A or B

19.3.7.2 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 in ADCn_SCSR1 to ADCn_SCSR7 registers. (x = 0 to 25)

For the A/D conversion operation in Single mode, use the SEQ0CH in ADCn_SCSR1 register to set the channel.

For the multi-channel sampling, up to 26 channels can be input and the number of conversions is set in the SEQCNT[4:0] in ADCn_MR register.

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.

19.3.7.3 Single Ended and Differential Input Channels

The SINGLE_EN bit in the ADCn_MR register allows the user to select between Single-ended input mode and Differential input mode. When Differential input mode is selected, the pin can be used as a negative external analog input pin is PB0/AN16.

19.3.7.4 Register Access when Changing A/D Conversion Channel

The ADCn_SCSRx registers cannot be modified or changed during the A/D conversion. To change the channel to be sampled, write ‘1’ to ASTOP in the ADCn_CR register to stop the A/D conversion.

The user can select a number of conversions in a sequence by configuring the SEQCNT[4:0] 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[4:0] is set to 0x03 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[4:0] in ADCn_CSCR register is set to 0x02, 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’.

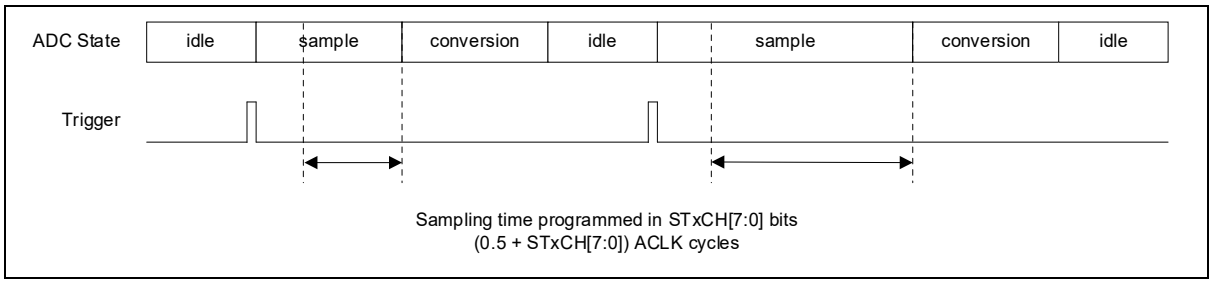
19.3.8 Programmable Sampling Time

The STxCH[7:0] in ADCn_CSTRy 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 (0.5 + STxCH[7:0]) ACLK cycles. The minimum sampling time is 0.5 ACLK cycles, and the sampling channel is always active when the STxCH[7:0] in the ADCn_CSTRy register are set to '11111'.

- If STxCH[7:0] = '00000000': 0.5 ACLK cycles.
- If STxCH[7:0] = '00000001': 1.5 ACLK cycles.
- If STxCH[7:0] = '00000010': 2.5 ACLK cycles.
- …
- If STxCH[7:0] = '11111101': 253.5 ACLK cycles.
- If STxCH[7:0] = '11111110': 254.5 ACLK cycles.
- If STxCH[7:0] = '11111111': sampling channel is always active.

Figure 157. Example of ADC Conversion Timing



19.3.9 Single Mode Conversion (ADMOD[1:0] = '00', SEQCNT[4:0] = 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] in ADCn_SCSR1 register.

- If ADMOD[1:0] = '00' and SEQCNT[4:0] = '00000', A/D conversion in Single mode.

There are two types of trigger sources in Single mode:

- If TRGSEL[1:0] = '00', Software trigger event (ACLK).
- If TRGSEL[1:0] = '01', TIMERn (n = 0 to 7) trigger event.

The SEQCNT[4:0] in ADCn_MR Register in Single Mode:

- If SEQCNT[4:0] = '00000', 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:

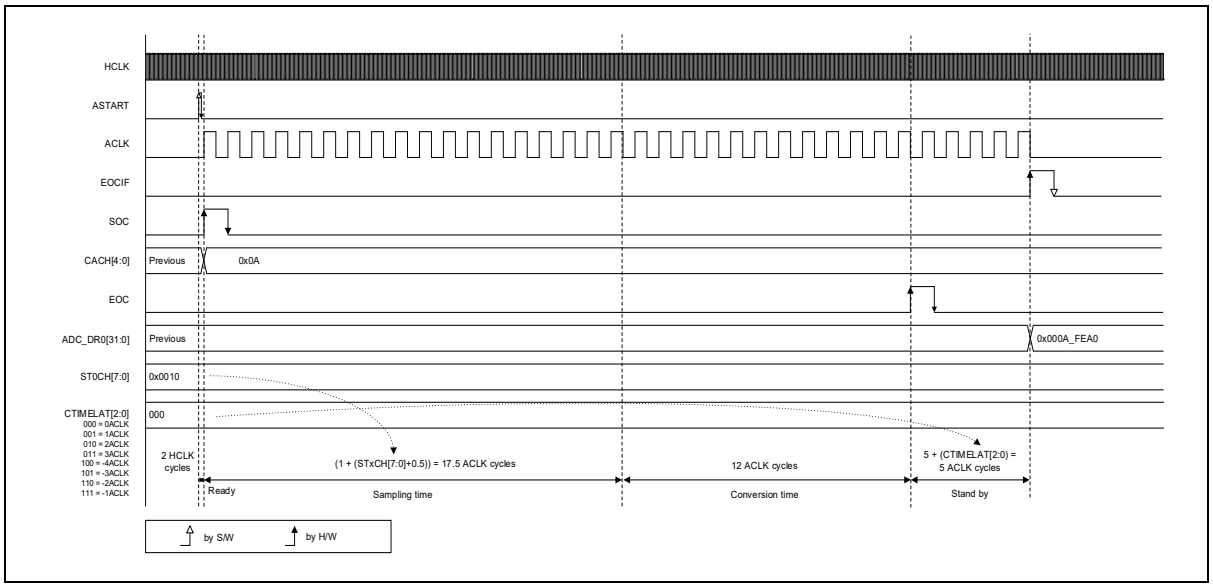
- If SEQTRG0[3:0] = '0000' to '1111', ADC trigger source.

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 (x = 0 to 25), and the EOCIF in the ADCn_SR register is set to '1'.

Users can obtain the A/D conversion result of the analog input by polling the EOCIF in ADCn_SR register using software or read-accessing the ADCn_DRx register (x = 0 to 25) at the point of time when the EOCIF in the ADCn_SR register interrupt occurs.

Figure 158. ADC Single Mode Timing (when ADMOD[1:0] = '00')



19.3.10 Sequential Mode Conversion (ADMOD[1:0] = '00', SEQCNT[4:0] > 0)

Sequential mode is used when the A/D conversion of the analog inputs is sequentially performed by each trigger source. The A/D conversion of the analog inputs is executed as many times as set in the SEQCNT[4:0] in the ADCn_MR register.

- If ADMOD[1:0] = '00' and SEQCNT[4:0] > '00000', A/D conversion in Sequential mode.

There are two types of trigger sources in Sequential mode:

- If TRGSEL[1:0] = '00', Software trigger event (ACLK)
- If TRGSEL[1:0] = '01', TIMERn (n = 0 to 7) trigger event

The SEQCNT[2:0] in ADC_MR Register in Sequential Mode:

Since the A/D conversion is executed for multiple channels, the SEQCNT[4:0] in the ADCn_MR register must be set to a number other than '0'.

- If SEQCNT[4:0] = '00001' to '11001', Multi-channel sampling in Sequential mode

Analog Input Channel in Sequential Mode:

The number of sequence channels that can be configured is set in the SEQCNT[4:0] in the ADCn_MR register.

- If SEQxCH[4:0] = y (x = 0 to 25, y = 0 to 23), sequence's channel is adc_chy.

Trigger Event Source Settings:

The trigger sources in each sequence can be configured is set in the SEQTRGx[3:0] in the ADCn_TRG register. In sequence mode, the 1st sequence has the highest priority and the 26th sequence has the lowest priority.

- If SEQTRGx[3:0] = y (x = 0 to 25, y = 0 to 7 or 15), channel's trigger source is y.

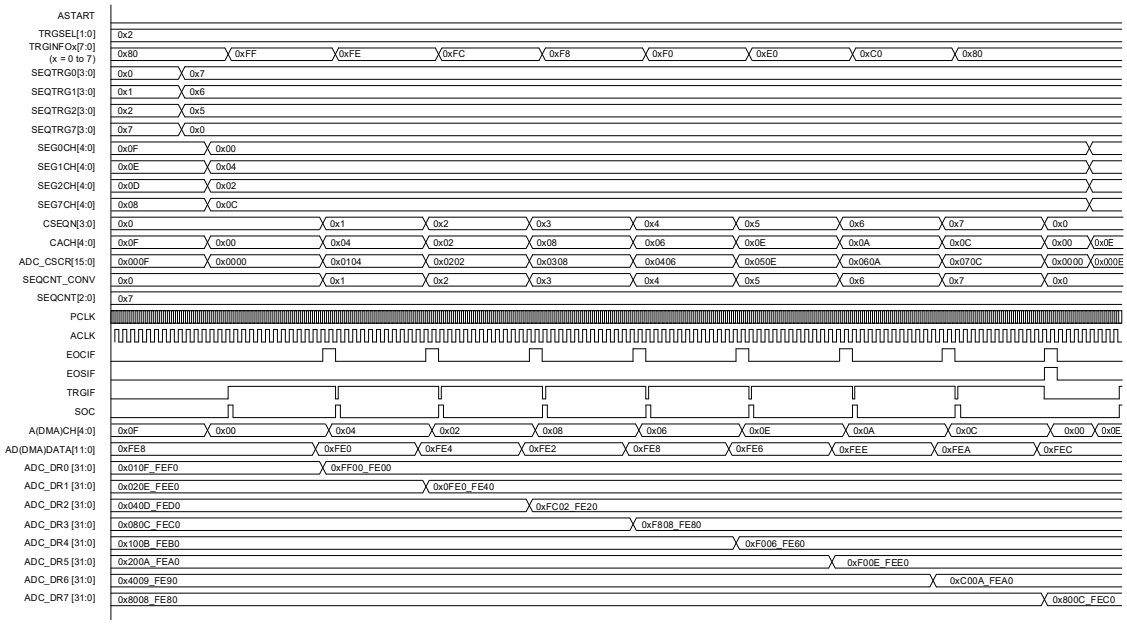
After the options for the ADC operation mode and trigger event sources, A/D conversion channel, and trigger event source are set, once the trigger event starts, sampling for the analog input starts.

Digitized data of the analog inputs is stored in the ADCn_DRx register (x = 0 to 25), and the EOCIF in ADCn_SR register is set to '1'.

The EOCIF in the ADCn_SR register is generated as many as the number of sequences set by the SEQCNT[4:0] 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 in the ADCn_SR register using software or read accessing the ADCn_DRx register (x = 0 to 25) 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 25) trigger, followed by as many conversions as defined by the SEQCNT[4:0] in the ADCn_MR register.

Figure 159. ADC Sequential Mode Timing (when ADMOD[1:0] = '00' and SEQCNT[4:0] = 7)



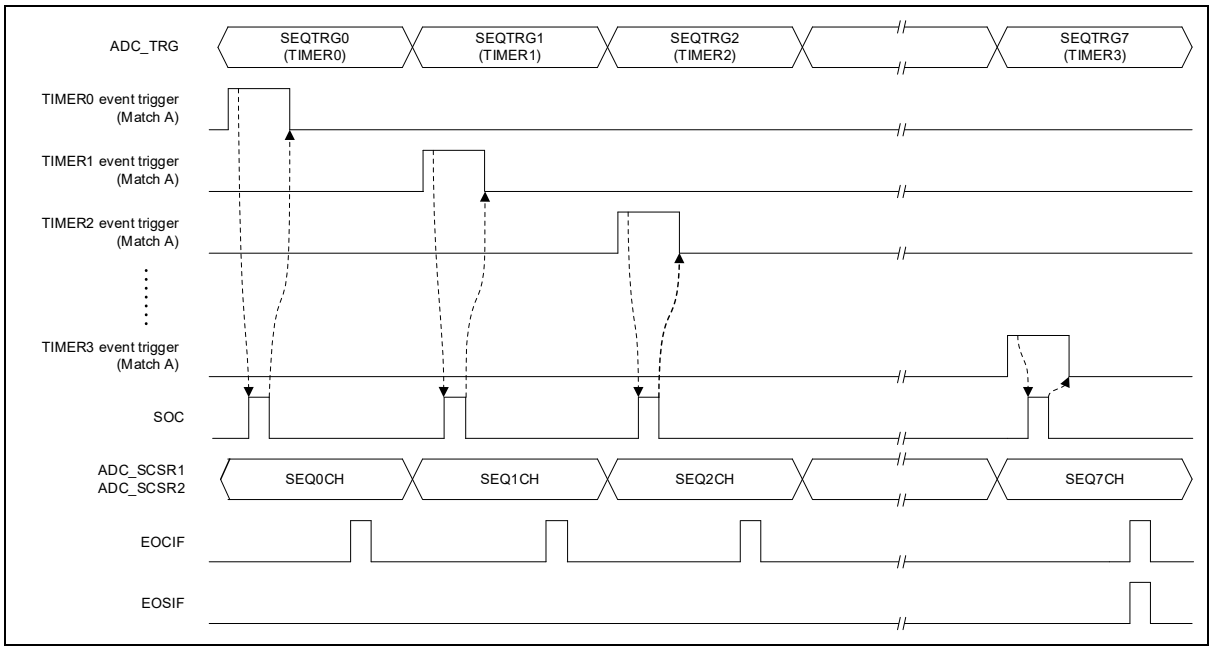
NOTE:

1. HCLK = 32 MHz, ACLK = 8 MHz (CLKDIV = 4)

Figure 160 shows a timing diagram that uses eight 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[4:0], and the sequence end flag EOS is generated when the EOC occurs during the last sequence.

Figure 160. ADC Trigger Timing in Sequential Mode (SEQCNT[4:0] = '00111', 8-Sequential Conversion)



19.3.11 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 two types of trigger sources in Burst mode:

- If TRGSEL[1:0] = '00', Software trigger event (ACLK).
- If TRGSEL[1:0] = '01', TIMERn trigger event (n = 0 to 7).

The SEQCNT[4:0] in ADC_MR Register in Burst Mode:

Since the A/D conversion is executed for the multi-channels, the SEQCNT[4:0] in the ADCn_MR register must be set to a number other than '0' in Burst mode.

- If SEQCNT[4:0] = '00001' to '11001', Multi-channel sampling in Burst mode

Analog Input Channel in Burst Mode:

The number of sequence channels that can be configured is set in the SEQCNT[4:0] in the 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), the 1st sequence's channel is adc_chx.
- If SEQ1CH[4:0] = x (x = 0 to 23), the 2nd sequence's channel is adc_chx.
- If SEQ2CH[4:0] = x (x = 0 to 23), the 3rd sequence's channel is adc_chx.
- If SEQ3CH[4:0] = x (x = 0 to 23), the 4th sequence's channel is adc_chx.
- If SEQ4CH[4:0] = x (x = 0 to 23), the 5th sequence's channel is adc_chx.
- If SEQ5CH[4:0] = x (x = 0 to 23), the 6th sequence's channel is adc_chx.
- If SEQ6CH[4:0] = x (x = 0 to 23), the 7th sequence's channel is adc_chx.
- If SEQ7CH[4:0] = x (x = 0 to 23), the 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 25) is triggered by the trigger source event assigned in the BSTTRG in the ADCn_TRG register.

If the TRGSEL[1:0] in the ADCn_MR register is set for the TIMER event triggers, SOC (Start of Conversion) signals are triggered by the BSTTRG in the ADCn_TRG set trigger.

If the BSTTRG in the ADCn_TRG register is set for the TIMER 3, for example, the ADC conversion gets started by the TIMER 3 trigger. To start the ADC conversion using Timer trigger, the ASTART bit in the ADCn_CR register must be set to '1' before starting the conversion.

Once an event is triggered by the BSTTRG 's set trigger source, the ADC shifts the ADC channels as many times as set in the SEQCNT[4:0] in the ADCn_MR register.

- If BSTTRG = x (x = 0 to 7 or 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.

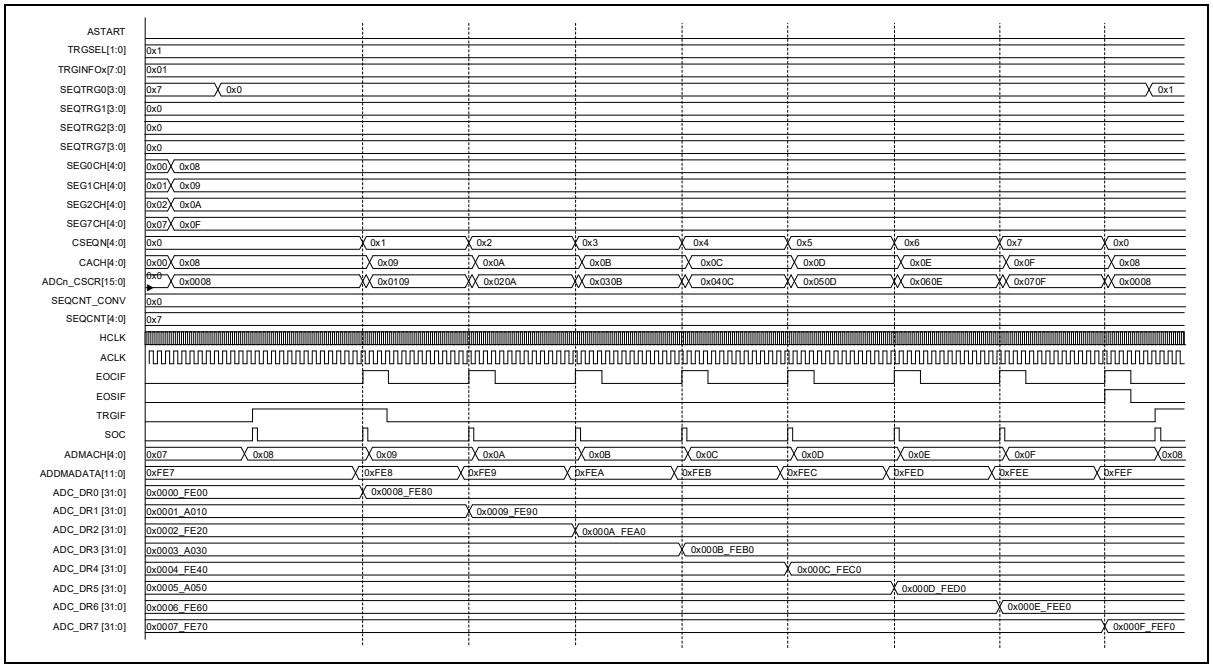
Digitized data of the analog inputs is stored in the ADCn_DRx register (x = 0 to 25), and the EOCIF in the ADCn_SR register is set to '1'.

The EOCIF in the ADCn_SR register is generated as many times as set in the SEQCNT[4:0] 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 in the ADCn_SR register using software or read accessing the ADCn_DRx register (x = 0 to 25) of the corresponding sequence at the point when the EOSIF in the ADCn_SR register interrupt occurs.

In Burst timer trigger mode, if the ARST bit in the ADCn_MR register is enabled, the ADC will continue to operate for the conversion even if the timer becomes disabled. To stop the ADC, the ASTOP bit in th e ADCn_CR register must be set to '1'.

Figure 161. ADC Burst Mode Timing (when ADMOD[1:0] = '01')



19.3.11.1 Burst Continuous Mode

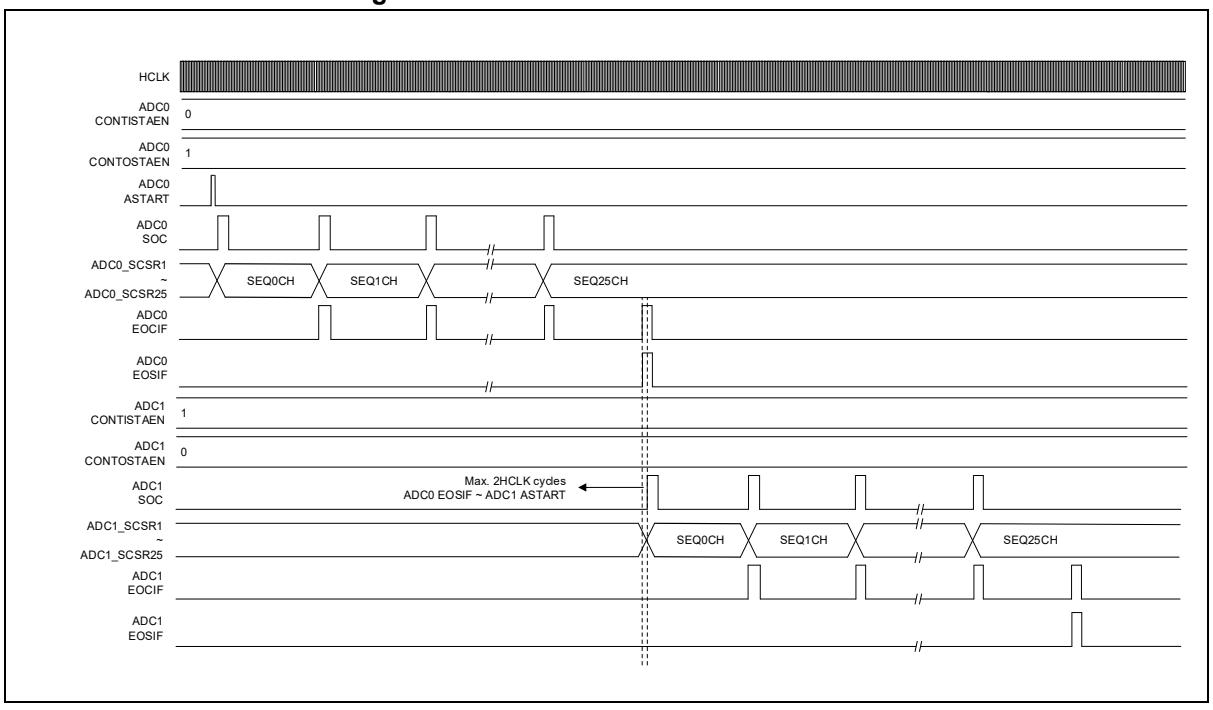
Continuous mode is a function to operate two ADCs sequentially. If the ADC1 conversion needs to be performed after ADC0 conversion, just set the CONTISTAEN and CONTOSTAEN bits before ADC0 start. Then, ADC1 conversion will be automatically performed after ADC0 conversion. Set the CONTOSTAEN bit for the first ADC block to start and set the CONTISTAEN for the second ADC block start.

See examples of case-specific settings below:

- Case1: ADC0 conversion → ADC1 conversion
  - ADC0 CONTOSTAEN = 1, ADC1 CONTOSTAEN = 0
  - ADC0 CONTISTAEN = 0, ADC1 CONTISTAEN = 1
- Case 0: ADC1 conversion → ADC0 conversion
  - ADC0 CONTOSTAEN = 0 ADC1 CONTOSTAEN = 1
  - ADC0 CONTISTAEN = 1 ADC1 CONTISTAEN = 0

The subsequent process is the same as in the Burst mode conversion (section 19.3.11). Continuous mode is only possible in Burst mode.

Figure 162. ADC Burst Continuons Mode



19.3.12 Multiple Mode Conversion (ADMOD[1:0] = '10', SEQCNT[4:0] > 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 two types of SOC trigger sources in Multiple mode:

- If TRGSEL[1:0] = '00', Software trigger event (ACLK).
- If TRGSEL[1:0] = '01', TIMERn (n = 0 to 7) trigger event.

The SEQCNT[4:0] in the ADCn_MR register in Multiple mode:

Since the A/D conversion is executed for multiple channels, the SEQCNT[4:0] in the ADCn_MR register must be set to a number other than '0'.

- If SEQCNT[4:0] = '00001' to '11001', Multi-channel sampling in Multiple mode.

Trigger Event Source Settings:

The number of trigger sources in each sequence that can be configured is set in the SEQCNT[4:0] in the ADCn_MR register.

Users can set the desired trigger sources in the SEQTRGx[3:0] in the ADCn_TRG register (x = 0 to 7 or 15). Each of these sources triggers an SOC when the triggering conditions are met, regardless of the sequence set in the SEQTRGx[3:0] in the ADCn_TRG register (x = 0 to 7 or 15).

For example, if the SEQCNT[4:0] in the ADCn_MR register is set to '11' and four trigger sources are selected from among the TIMER 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 7 or 15), the 1st channel's trigger source is x.
- If SEQTRG1[3:0] = x (x = 0 to 7 or 15), the 2nd channel's trigger source is x.
- If SEQTRG2[3:0] = x (x = 0 to 7 or 15), the 3rd channel's trigger source is x.
- If SEQTRG3[3:0] = x (x = 0 to 7 or 15), the 4th channel's trigger source is x.
- If SEQTRG4[3:0] = x (x = 0 to 7 or 15), the 5th channel's trigger source is x.
- If SEQTRG5[3:0] = x (x = 0 to 7 or 15), the 6th channel's trigger source is x.
- If SEQTRG6[3:0] = x (x = 0 to 7 or 15), the 7th channel's trigger source is x.
- If SEQTRG7[3:0] = x (x = 0 to 7 or 15), the 8th channel's trigger source is x.

NOTE:

1. The unused trigger source of the SEQTRGx[3:0] should be set to '14' to 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 (x = 0 to 25), and the EOCIF in the ADCn_SR register is set to '1'.

The EOCIF ADCn_SR register is generated as many as the number of sequences set in the SEQCNT[4:0], and the End of Sequence Interrupt Flag EOSIF 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 in the ADCn_SR register using software or read accessing the ADCn_DRx register (x = 0 to 25) of the corresponding sequence at the point when the EOCIF in the ADCn_SR register interrupt occurs.

19.3.13 Starting Conversions (ASTART, AESTART, Start Trigger, ARST)

19.3.13.1 Start A/D Conversion (ASTART)

The A/D conversion starts when the ASTART 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)

19.3.13.2 All Start A/D Conversion (AESTART)

AESTART is a function to start ADC0 and ADC1 simultaneously. Both ADC0 and ADC1 must be configured before running AESTART and will start simultaneously even if both ASTART and AESTART bits are set on one of those two ADCs.

The following shows ADC operation with the combination of ASTART and AESTART:

- If AESTART = 0
  - 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)
- If AESTART = 1: ADC0 and ADC1 simultaneous start is enabled.
  - If ASTART = 0, ADC conversion does not start. (Software Trigger)
  - If ASTART = 1, ADC0 and ADC1 simultaneous conversion starts. This bit is cleared at the next ADC clock cycle. (Software Trigger)

19.3.13.3 Reset A/D Conversion (ARST)

If the ARST in ADCn_MR register is set to '0' and the software trigger is used as a trigger source, users must set the ASTART in ADCn_CR register to '1' using software to restart the A/D conversion at the end of the A/D conversion sequence.

If the ARST bit in the ADCn_MR register is set to '1' before triggering ASTART, the A/D conversion will commence automatically when EOSIF of the ADCn_SR register occurs subsequent to the completion of the A/D conversion.

- If ARST = 0, A/D conversion stops at EOS. (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).

19.3.13.4 Start A/D Conversion by Trigger Sources

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 19.3.16)

- The TEN = 1 in the TIMERn_CR2 (n = 0 to 7)
- The ASTART = 1 in the ADCn_MR

When the A/D conversion is completed for one channel, the EOCIF flag is generated and the ASTART in the ADCn_CR register is cleared by hardware automatically.

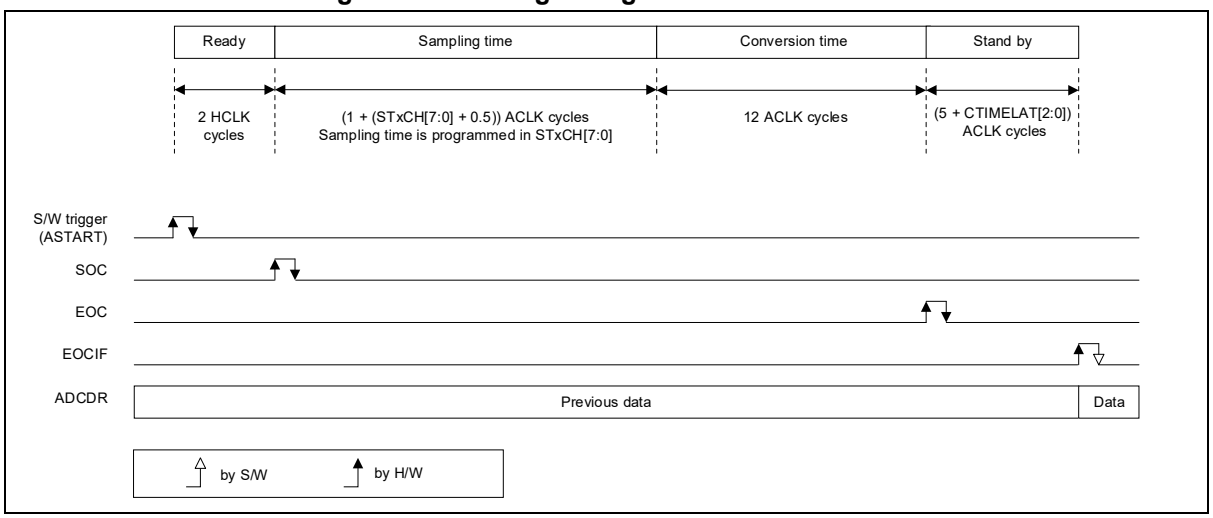
In Sequential, Multiple, and Burst mode, the EOCIF flags are generated as many channels as set in a sequence, and the EOSIF flag is generated when sampling of the last channel is complete except Multiple mode.

19.3.14 ADC Timing

The A/D conversion time consists of the following:

t_{ADC} = 2 HCLK cycles [Delay for sync until SOC] + (1 + (STxCH[7:0]+0.5)) ACLK cycles [Sampling time] + 12 ACLK cycles [Conversion time (EOC)] + (5 + CTIMELAT[2:0]) ACLK cycles [EOCIF]

Figure 163. Analog-to-Digital Conversion Time



The total conversion time is calculated as follows:

Example)

With f_{HCLK} = 32 MHz, f_{ACLK} = 16 MHz and a STxCH[7:0] = 2, Sampling time is 3.5 ACLK cycles, CTIMELAT[2:0] = 0, Conversion time latency cycle is 0 ACLK cycles

Then, t_{ADC} = (2)HCLK + {(1 + (2 + 0.5)) + 12 + (5 + 0)} ACLK cycles = 2 HCLK + 20.5 ACLK = 1.3437 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 RAIN connected to the analog input and STxCH[7:0] in the ADC_MR register value.

Table 135 shows the recommended values for RAIN and STxCH[7:0] in the ADC_MR register for A/D conversion when the ADC's reference voltage is 3.0 V and ACLK 16 MHz.

Table 135. ADC Frequency for R_{AIN} (1)

(2.4 V＜AV_{DD}＜3.6 V, fHCLK=32 MHz, fACLK=16 MHz, CTIMELAT[2:0] = 4]

RAIN Max. [kΩ]	Ready		Sampling time			Conversion time		Stand by		t_{ADC} [µs]	f_{ADC} [Msps]
	[Cycle]	[µs]	STxCH	[Cycle]	[µs]	[Cycle]	[µs]	[Cycle]	[µs]		
0.1	2	0.0625	2	3.5	0.219	12	0.75	1	0.0625	1.094	0.914
1.5			6	7.5	0.469					1.344	0.744
3.5			14	15.5	0.969					1.844	0.542
5.0			22	23.5	1.469					2.344	0.427
10.0			46	47.5	2.969					3.844	0.26
20.0			94	95.5	5.969					6.844	0.146
35.0			190	191.5	11.969					12.844	0.078
50.0			254	255.5	15.969					16.844	0.059

NOTES:

1. The parameters are guaranteed by design.
2. f_{ADC} is an A/D conversion rate from SOC to EOC.

Table 136. ADC Frequency for R_{AIN} (2)

(1.8 V＜AV_{DD}＜2.4 V, fHCLK=32 MHz, fACLK=8 MHz, CTIMELAT[2:0] = 4)

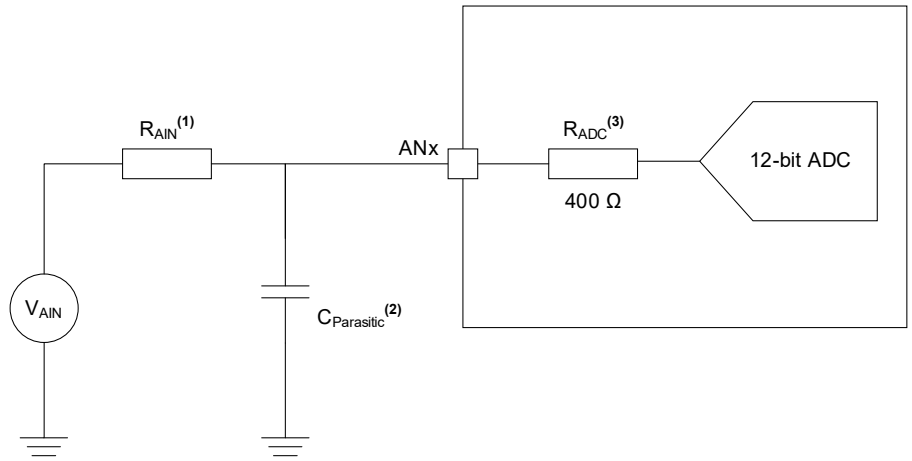
RAIN Max. [kΩ]	Ready		Sampling time			Conversion time		Stand by		t_{ADC} [µs]	f_{ADC} [Msps]
	[Cycle]	[µs]	STxCH	[Cycle]	[µs]	[Cycle]	[µs]	[Cycle]	[µs]		
0.1	2	0.0625	2	3.5	0.438	12	1.5	1	0.125	2.125	0.471
1.5			6	7.5	0.938					2.625	0.381
3.5			14	15.5	1.938					3.625	0.276
5.0			22	23.5	2.938					4.625	0.216
10.0			46	47.5	5.938					7.625	0.131
20.0			94	95.5	11.938					13.625	0.073
35.0			190	191.5	23.938					25.625	0.039
50.0			254	255.5	31.938					33.625	0.03

NOTES:

1. The parameters are guaranteed by design.
2. f_{ADC} is an A/D conversion rate from SOC to EOC.

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} must be considered when inputting an external analog source. (See Figure 164 for how to design the R_{AIN} resistor circuit for the ANx analog input.)

Figure 164. Typical Connection Diagram with ADC



NOTES:

1. Refer to Table 135 for more information on 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.

19.3.15 Stopping an Ongoing Conversion (ASTOP)

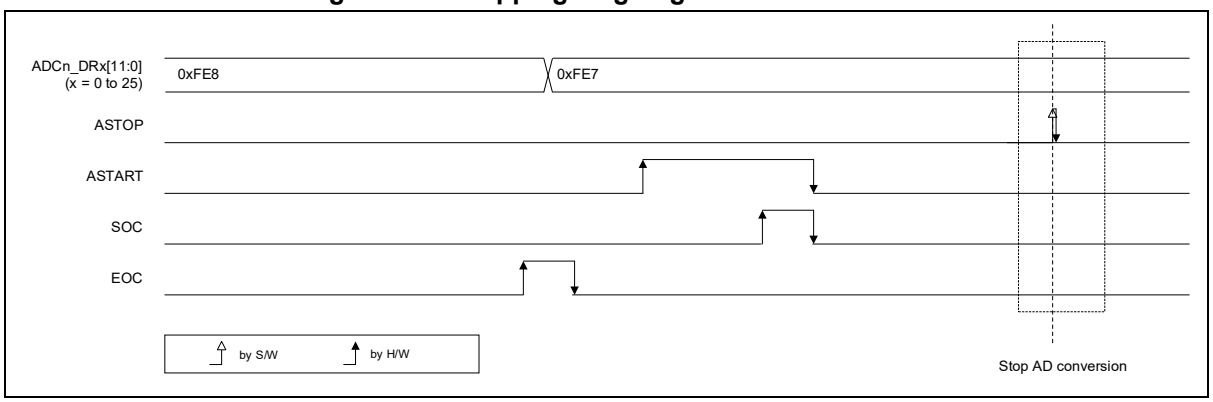
The A/D conversion in progress can be stopped by setting the ASTOP bit in the ADCn_CR register to '1' using software. When the A/D conversion stops, the ADC operation in progress is reset. The ADC can be reconfigured (e.g., channel selection, trigger change) for new input data.

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 (x = 0 to 25) 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 165. Stopping Ongoing A/D Conversions



19.3.16 Conversion on External Trigger (TRGSEL[1:0], SEQTRGx[3:0] and TRGSRC[3:0], TRGIE, TRGIF)

The A/D conversion can be triggered by the software or the events of the TIMER 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'.

19.3.16.1 ADC Trigger Source Selection (TRGSEL[1:0])

Users can select a trigger source by configuring the TRGSEL[1:0] in the ADC_MR register as described in Table 137.

Table 137. External Trigger Source Configuration

TRGSEL[1:0]	Trigger Source	Description
00	ASTART	Disables event triggers and enables the soft trigger only.
01	TIMERn (n = 0 to 7)	Enables the timer event trigger and soft trigger.

To use the TIMER as the ADC trigger source, select the event and polarity of the TIMER channel, then set the ADCTRGEN bit in the TIMERn_CR1 register bit to '1'. (n = 0 to 7).

19.3.16.2 ADC Trigger Channel Settings (SEQTRGx[3:0] and TRGSRC[3:0])

Once a type of the trigger source is determined, up to 9 trigger sources (As a single trigger source or a sequence of trigger sources per ADC unit) can be selected by configuring the ADCn_TRG register. If a trigger source for a single A/D channel is used, users can use the SEQTRG0[3:0] in the ADCn_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] in ADCn_TRG. (x = 0 to 25)

Table 138. Trigger Source Table

Trigger Source (SEQTRGx)	TIMER (TRGSEL[1:0] = '01')
0	TIMER 0
1	TIMER 1
2	TIMER 2
3	TIMER 3
4	TIMER 4
5	TIMER 5
6	TIMER 6
7	TIMER 7
8~14	Reserved
15	ASTART(1)

NOTE:

1. ASTART bit in ADCn_CR register is a software trigger.

19.3.16.3 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 139. Start Control of AD Trigger Conversion

TRGSEL[1:0]	Start Control Bit
ASTART	ASTART = '1' in ADCn_CR register ASTART = '1' in ADCn_ICR register
TIMERn (n = 0 to 7)	TEN = '1' in TIMERn_CR2 register

19.3.16.4 ADC Trigger Interrupt and Flag (TRGIE, TRGIF)

Interrupts for the ADC trigger events can be enabled by configuring the TRGIE bit in the ADC_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 TIMER is not enabled, TRGIF flag of ADC is generated when trigger source event happens. 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.

19.3.17 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 flag 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 (x = 0 to 25) 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.

19.3.18 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[4:0] in the ADCn_MR register as shown below:

- If SEQCNT[4:0] = '00000': 1 Sequence ADC (Single mode conversion).
- If SEQCNT[4:0] = '00001': 2 Sequences ADC.
- If SEQCNT[4:0] = '00010': 3 Sequences ADC.
- If SEQCNT[4:0] = '00011': 4 Sequences ADC.
- If SEQCNT[4:0] = '00100': 5 Sequences ADC.
- If SEQCNT[4:0] = '00101': 6 Sequences ADC.
- If SEQCNT[4:0] = '00110': 7 Sequences ADC.
- If SEQCNT[4:0] = '00111': 8 Sequences ADC.

After setting the number of channels, users can assign the channels that are performed in each sequence using the ADCn_SCSRx register. Refer to section 19.3.7.

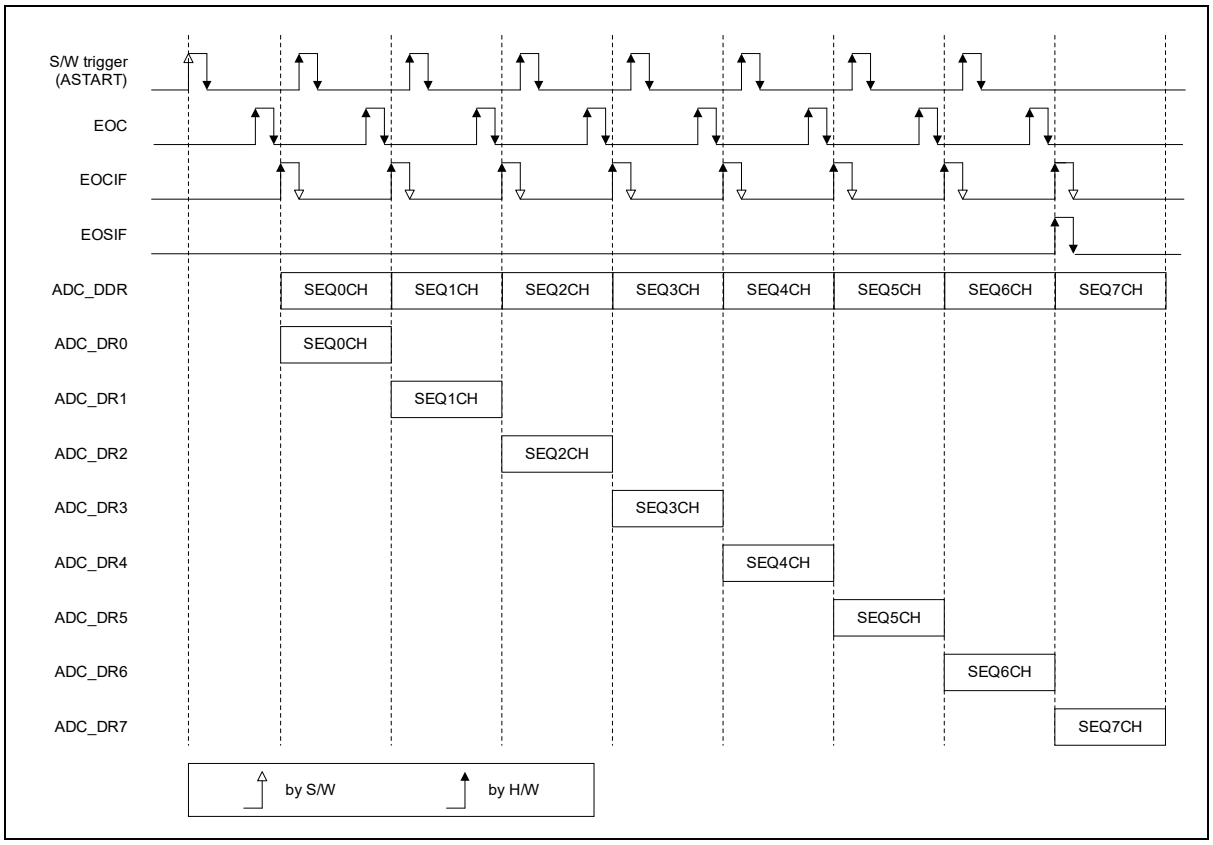
Sequence in Sequential mode is performed by following the ascending order of the sequence trigger source number set in the SEQTRGx[3:0] in ADC_TRG register. In Multiple mode, the sampling order is determined by the timing of each trigger source's event.

Users can adjust the trigger timing to set the sampling period for the ADC sequence. Refer to section 19.3.16.

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 166 shows a timing diagram in which the A/D conversion is performed by a software trigger on the analog inputs on 8 channels at the same interval in Sequential mode.

Figure 166. Example of A/D Conversion Timing in Sequence Mode



19.3.19 Data Management and Information

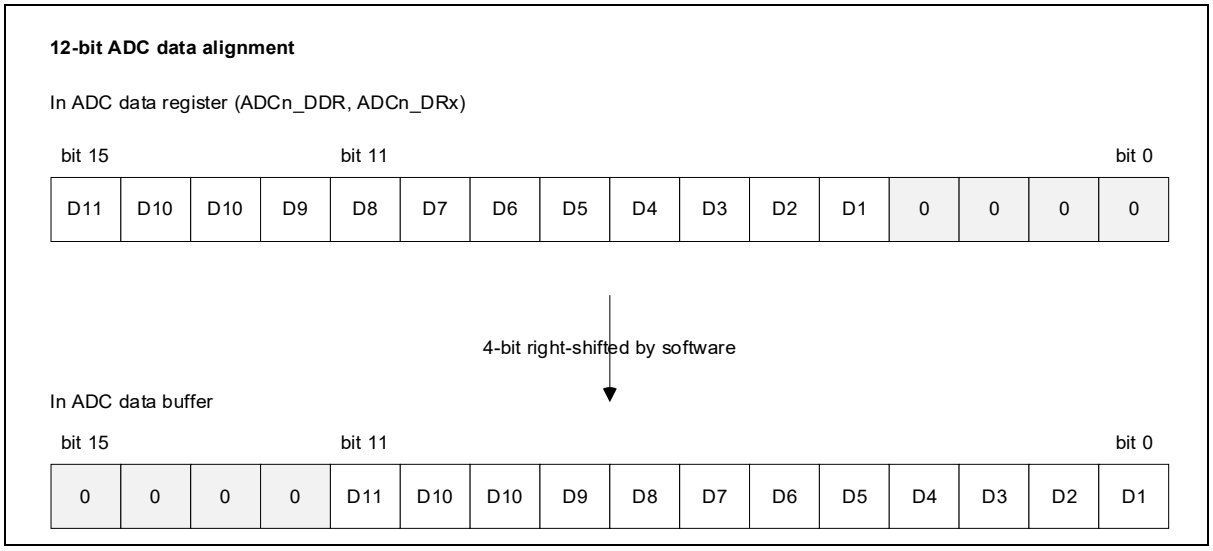
19.3.19.1 Data Register and Data Alignment by Software

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, …, ADCn_DR25.

- 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 167, 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 four bits, generating a 12-bit value:

Figure 167. Left Alignment (Unsigned Value)

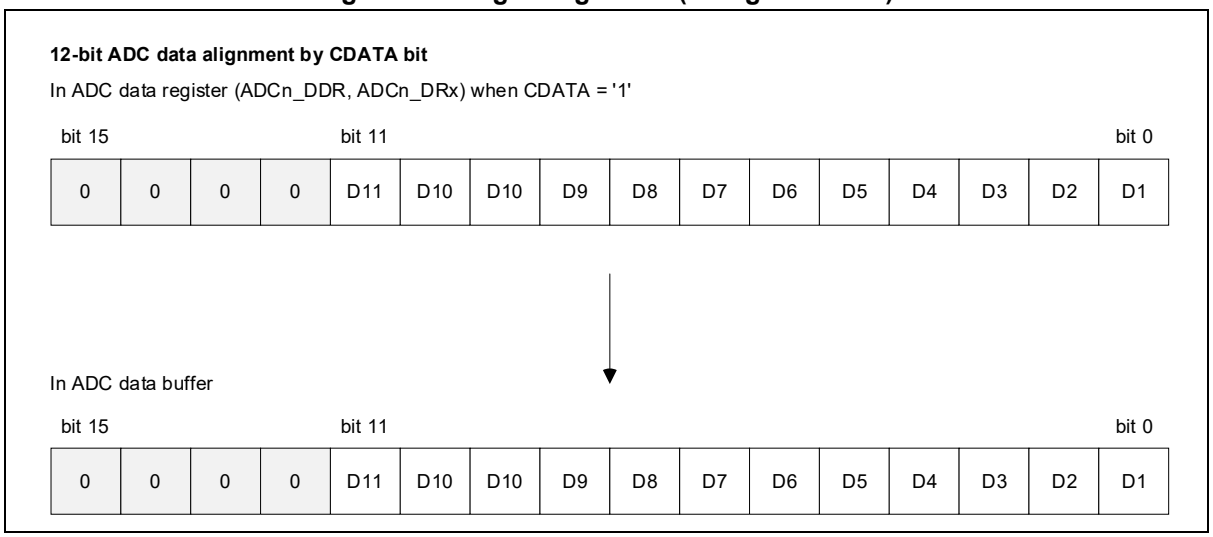


19.3.19.2 Conversion Data Offset Alignment

ADC must be calibrated to ensure the accuracy of digitally converted values. During calibration, the offset of the A/D conversion data value is adjusted. The CDATA bit function automatically aligns the offset with a 4-bit right-shift. Using this function can directly utilize the A/D conversion data obtained from the ADCn_DRx register, without requiring 4-bit right-shift processing by software. Therefore, Enabling the CDATA bit is recommended to reduce the software processing time for calibration when acquiring A/D conversion data.

- When the CDATA bit is '0', the A/D conversion value is stored in the [15:4] bit field of ADCn_DDR and ADCn_DRx registers.
- When the CDATA bit is '1', the A/D conversion value is stored in the [11:0] bit field of ADCn_DDR and ADCn_DRx registers.

Figure 168. Right Alignment (Unsigned Value)



19.3.19.3 Channel and Trigger Information

Users can get information of the A/D conversion data channel and the sequence information by checking the upper 16 bits of the ADC data registers.

- The A/D conversion channel information to the ACH[4:0] in the ADCn_DRx register.
- The sequence information for the channel where the A/D conversion is performed to the TRGINFO in the ADCn_DRx register.

The TRGINFO in the ADCn_DDR register shows that the lower the channel number set in a sequence, the higher the channel priority is executed first.

Figure 169 shows the configuration of the channel and sequence information in the ADCn_DDR and ADCn_DRx registers. As shown in the figure, the SEQxCH (x = 0 to 25) sequence is configured with 24 channels adc_ch0 to adc_ch23, allowing users to check the channel and trigger information in the ADCn_DDR and ADCn_DRx registers.

Figure 169. Channel and Trigger Source Information of A/D Conversion Sequence

	SEQ0CH	SEQ1CH	SEQ2CH	SEQ3CH	SEQ4CH	SEQ5CH	SEQ6CH	SEQ7CH	...	SEQ24CH	SEQ25CH
	adc_ch0	adc_ch1	adc_ch2	adc_ch3	adc_ch4	adc_ch5	adc_ch6	adc_ch7		adc_ch23	adc_ch23
ACH[4:0] (Channel number of AD conversion)	0x00	0x01	0x02	0x03	0x04	0x05	0x06	0x07		0x17	0x17
TRGINFO[4:0] (Trigger source priority of AD conversion)	0x00	0x01	0x02	0x03	0x04	0x05	0x06	0x07		0x18	0x19

19.3.20 Trigger Pending Function in Sequential Mode

In sequential mode of the ADC module, the SEQTRGT bit in the ADCn_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 = 0: when a second trigger event occurs while waiting for the first trigger event, the second trigger event is ignored, and the first trigger event of the next sequence is processed.
- SEQTRGT is set to 1, when waiting for the first trigger event and the second trigger event occurs, the ADC waits for the first trigger source event after pending the second trigger event until the first trigger source event of next sequence. Then, the first trigger source event of the next sequence is processed.

19.3.21 ADC Low-Power Modes

Table 140. Effect of Low-Power Modes 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.

19.3.22 A/D Conversion Using the 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 in DMAn_SR register to '1' with the ADEN bit in the ADCn_MR register to '1' as listed below:

- If DMAEN = '0', ADC DMA disabled.
- If DMAEN = '1', ADC DMA enabled.

To connect the ADC to the DMA channel, set the PERISEL[4:0] in the DMA_CR register to select the ADC, referring to the number below:

- PERISEL[4:0] = '01111' or '10000' (0x0F or 0x10, Selected ADC module of DMA transferring) in the DMAn_CR register.

Set the SIZE[1:0] 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] in the DMAn_CR register (n = 0 to 4). 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 PAR[15:0] in the DMAn_PAR register and the DMA transfer's destination memory (SRAM) address in the MAR[15:0] in the DMAn_MAR register.

- PAR[15:0] = 0x202C (0x4200_202C, ADC_DDR register address)
- MAR[15:0] = 0x1000 (0x2000_1000, buffer data address)

The direction of the DMA transfer is determined based on the memory and can be controlled by setting the DIR in 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 ADC_IER register.

- If DMAIE = '0', Disable ADC DMA done interrupt.
- If DMAIE = '1', Enable ADC DMA done interrupt.

When the DMAEN 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] in the DMAn_CR register is decreased by 1 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] 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 ADC_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 in the DMAn_SR register of the DMA module is enabled when the transmission is complete and the TRANSCNT[11:0] value becomes zero. The EOT flag is cleared by writing a number larger than zero to the TRANSCNT[11:0] 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 ADC_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.

NOTE:

1. Applications using ADC modules must clear the EOC at initialization, when using DEEP-SLEEP mode. Please refer to section 19.3.17.

19.3.23 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:

- When the ADC completes the A/D conversion and the EOCIF in the ADCn_SR register is set to '1', the ADC sends a data transfer request to the DMA.
- The DMA transfers the A/D conversion results to the specified SRAM memory address and sends the ACK signal to the ADC.
- When the TRANSCNT[11:0] in the DMAn_CR register value becomes zero, the EOT 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 (DMAOF) in the ADCn_SR register) is set to '1'. The DMAOF does not support the interrupt function.

If the overrun flag is generated since 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 DMAOF in the ADCn_SR register.

Even if the overrun flag is generated, the ADC maintains operating. Unless software sets the ASTOP 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 in the ADCn_SR register is generated.

19.3.24 Comparison of ADC Sensing Value

The ADC has a built-in comparator that compares the sampling results of the A/D 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] in the ADCn_CMPnR register to select a target ADC channel for comparison and enter the comparison value in the CVAL[11:0] in the ADCn_CMPnR register.

- If CMPEN = 0, ADC comparator operation is disabled.
- If CMPEN = 1, ADC comparator operation is enabled.

Figure 170. Block Diagram of ADC Internal Comparator

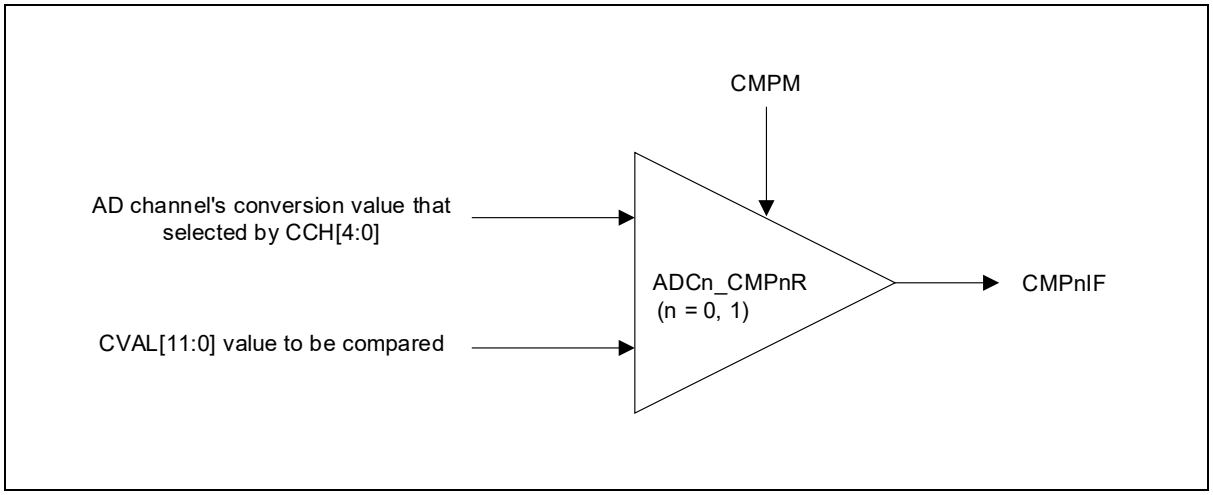


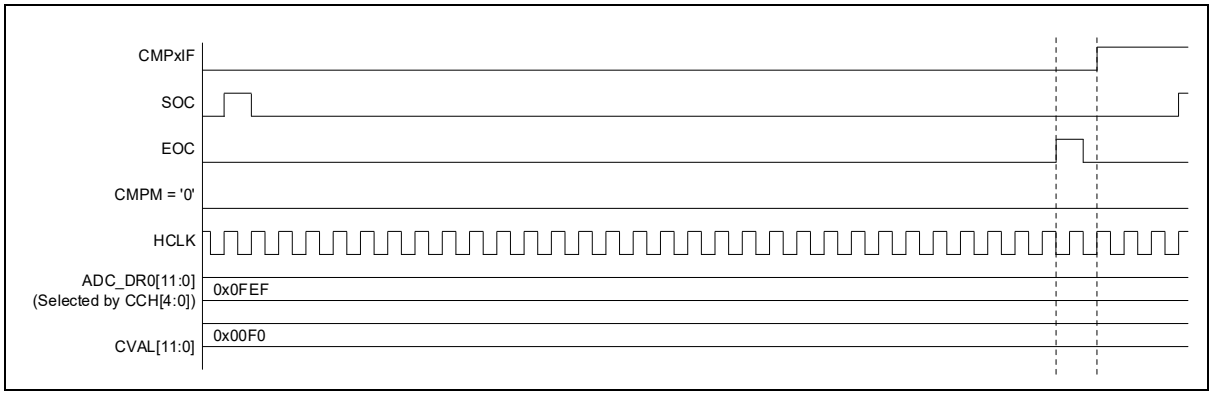
Table 141 shows the conditions under which the CMPnIF flag occurs by comparing the CVAL[11:0] and the ADC channel values set in CCH[4:0] according to the CMPM settings.

Table 141. Comparison and Flag Generation Conditions by CMPM

CMPM	Comparison Condition	CMPxIF 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 CMPxIF in the ADCn_SR register indicating the comparison completion is generated (CMPxIF = '1').

Figure 171. Timing Diagram of Comparison Between A/D Conversion Results and Reference Value



The comparison results based on the CMPM in the ADCn_CMPxR register value affect the flag setting condition of the CMPnIF 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 in the ADCn_CMPxR register to ‘1’ as shown below:

- If CMPIEN = 0, ADC compare interrupt is disabled.
- If CMPIEN = 1, ADC compare interrupt is enabled.

19.4 ADC Interrupts

For each ADC unit, an interrupt can be generated by setting below:

- End of single data conversion (EOCIE)
- End of a sequence mode conversion (EOSIE)
- Interrupt by trigger source (TRGIE)
- End of DMA transfer of A/D conversion data (DMAIE)
- ADC conversion data comparison (CMPnIE)

Each Interrupt Enable bits are available for flexibility.

Table 142. Control Bits and Event Flags for ADC Interrupts (n = 0 and 1)

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	DMAF	DMAIE
ADC conversion data comparison	CMPnIF	CMPnIE

19.5 ADC Registers

The base address and register map of the ADC module are described in the following tables:

Table 143. Base Address of ADC

Name	Base Address
ADC0	0x4100_5000
ADC1	0x4100_5100

Table 144. ADC Register Map

Name	Offset	Type	Description	Reset Value	Reference
ADCn_MR	0x0000	RW	ADC mode register	0x0000_0000	19.5.1
ADCn_CSCR	0x0004	RW	ADC current sequence/channel register	0x0000_0000	19.5.2
ADCn_CCR	0x0008	RW	ADC clock control register	0x0000_0000	19.5.3
ADCn_TRG1	0x000C	RW	ADC trigger select 1 register	0x0000_0000	19.5.4
ADCn_TRG2	0x0010	RW	ADC trigger select 2 register	0x0000_0000	19.5.5
ADCn_TRG3	0x0014	RW	ADC trigger select 3 register	0x0000_0000	19.5.6
ADCn_TRG4	0x0018	RW	ADC trigger select 4 register	0x0000_0000	19.5.7
ADCn_CSTR1	0x001C	RW	ADC channel sampling time 1 register	0x0000_0000	19.5.8
ADCn_CSTR2	0x0020	RW	ADC channel sampling time 2 register	0x0000_0000	19.5.9
ADCn_CSTR3	0x0024	RW	ADC channel sampling time 3 register	0x0000_0000	19.5.10
ADCn_CSTR4	0x0028	RW	ADC channel sampling time 4 register	0x0000_0000	19.5.11
ADCn_CSTR5	0x002C	RW	ADC channel sampling time 5 register	0x0000_0000	19.5.12
ADCn_CSTR6	0x0030	RW	ADC channel sampling time 6 register	0x0000_0000	19.5.13
ADCn_CSTR7	0x0034	RW	ADC channel sampling time 7 register	0x0000_0000	19.5.14
ADCn_SCSR1	0x0038	RW	ADC channel select 1 register	0x0000_0000	19.5.15
ADCn_SCSR2	0x003C	RW	ADC channel select 2 register	0x0000_0000	19.5.16
ADCn_SCSR3	0x0040	RW	ADC channel select 3 register	0x0000_0000	19.5.17
ADCn_SCSR4	0x0044	RW	ADC channel select 4 register	0x0000_0000	19.5.18
ADCn_SCSR5	0x0048	RW	ADC channel select 5 register	0x0000_0000	19.5.19
ADCn_SCSR6	0x004C	RW	ADC channel select 6 register	0x0000_0000	19.5.20
ADCn_SCSR7	0x0050	RW	ADC channel select 7 register	0x0000_0000	19.5.21
ADCn_CR	0x0054	RW	ADC control register	0x0000_0000	19.5.22
ADCn_SR	0x0058	RW	ADC status register	0x0000_0000	19.5.23

Table 144. ADC Register Map (continued)

Name	Offset	Type	Description	Reset Value	Ref.
ADCn_IER	0x005C	RW	ADC interrupt enable register	0x0000_0000	19.5.24
ADCn_DDR	0x0060	RO	ADC DMA data register	0x0000_0000	19.5.25
ADCn_DR0	0x0064	RO	ADC Sequence 0 data register	0x0000_0000	19.5.26
ADCn_DR1	0x0068	RO	ADC Sequence 1 data register	0x0000_0000	19.5.26
ADCn_DR2	0x006C	RO	ADC Sequence 2 data register	0x0000_0000	19.5.26
ADCn_DR3	0x0070	RO	ADC Sequence 3 data register	0x0000_0000	19.5.26
ADCn_DR4	0x0074	RO	ADC Sequence 4 data register	0x0000_0000	19.5.26
ADCn_DR5	0x0078	RO	ADC Sequence 5 data register	0x0000_0000	19.5.26
ADCn_DR6	0x007C	RO	ADC Sequence 6 data register	0x0000_0000	19.5.26
ADCn_DR7	0x0080	RO	ADC Sequence 7 data register	0x0000_0000	19.5.26
ADCn_DR8	0x0084	RO	ADC Sequence 8 data register	0x0000_0000	19.5.26
ADCn_DR9	0x0088	RO	ADC Sequence 9 data register	0x0000_0000	19.5.26
ADCn_DR10	0x008C	RO	ADC Sequence 10 data register	0x0000_0000	19.5.26
ADCn_DR11	0x0090	RO	ADC Sequence 11 data register	0x0000_0000	19.5.26
ADCn_DR12	0x0094	RO	ADC Sequence 12 data register	0x0000_0000	19.5.26
ADCn_DR13	0x0098	RO	ADC Sequence 13 data register	0x0000_0000	19.5.26
ADCn_DR14	0x009C	RO	ADC Sequence 14 data register	0x0000_0000	19.5.26
ADCn_DR15	0x00A0	RO	ADC Sequence 15 data register	0x0000_0000	19.5.26
ADCn_DR16	0x00A4	RO	ADC Sequence 16 data register	0x0000_0000	19.5.26
ADCn_DR17	0x00A8	RO	ADC Sequence 17 data register	0x0000_0000	19.5.26
ADCn_DR18	0x00AC	RO	ADC Sequence 18 data register	0x0000_0000	19.5.26
ADCn_DR19	0x00B0	RO	ADC Sequence 19 data register	0x0000_0000	19.5.26
ADCn_DR20	0x00B4	RO	ADC Sequence 20 data register	0x0000_0000	19.5.26
ADCn_DR21	0x00B8	RO	ADC Sequence 21 data register	0x0000_0000	19.5.26
ADCn_DR22	0x00BC	RO	ADC Sequence 22 data register	0x0000_0000	19.5.26
ADCn_DR23	0x00C0	RO	ADC Sequence 23 data register	0x0000_0000	19.5.26
ADCn_DR24	0x00C4	RO	ADC Sequence 24 data register	0x0000_0000	19.5.26
ADCn_DR25	0x00C8	RO	ADC Sequence 25 data register	0x0000_0000	19.5.26
ADCn_CMP0R	0x00E0	RW	ADC channel comparison 0 register	0x0000_0000	19.5.27
ADCn_CMP1R	0x00E4	RW	ADC channel comparison 1 register	0x0000_0000	19.5.28

19.5.1 ADCn_MR: ADC Mode Register

ADCn_MR configures the modes of the ADC module. This register must be set prior to all other ADC registers according to the intended use of the ADC module.

ADC0_MR=0x4100_5000, ADC1_MR=0x4100_5100

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
ACLKAON	Reserved		SINGLE_EN	SEL_REF	CTIMELAT[2:0]			Reserved	EXTCONSTAEN	CONTISTAEN	CONTOSTAEN	CDATA	Reserved	DMAEN	Reserved			SEQTRGT	SEQCNT[4:0]					ADEN	ARST	ADMOD[1:0]		Reserved		TRGSEL[1:0]	
0	-		0	0	000			-	0	0	0	0	-	0	-			0	0	0	0	0	0	0	0	00		-		00	
RW	-		RW	RW	RW			-	RW	RW	RW	RW	-	RW	-			RW	RW					RW	RW	RW		-		RW	

31	ACLKAON	ADC clock always on control	
		0	ADC clock auto control
		1	ADC clock always on enable
28	SINGLE_EN	Select input mode (not to change during conversion)	
		0	Differential input mode
		1	Single-end input mode
27	SEL_REF	ADC SEL_REF control	
		0	Disable (AVDD, analog input range(max))
		1	Enable (VREFP, analog input range(max))
26 24	CTIMELAT[2:0]	Conversion time latency cycle parameter	
		000	0
		001	+1
		010	+2
		011	+3
		100	-4
		101	-3
		110	-2
		111	-1
22	EXTCONSTAEN	ADC dual mode enable bit	
		0	Disable
		1	Enable
21	CONTISTAEN	Continuous start enable (input mode)	
		0	Disable
		1	Enable
20	CONTOSTAEN	Continuous start enable (output mode)	
		0	Disable
		1	Enable
19	CDATA	Conversion data offset alignment	
		0	[15:4] offset alignment
		1	[11:0] offset alignment
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 19.3.20.)	
		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[4:0]	Number of conversions in a sequence (a maximum of 26) Sequence = SEQCNT + 1 The setting of the SEQCNT is valid only in 0~25. If ADMOD[1:0] is 0 and SEQCNT[4:0] is not 0, the CSEQN value increments to the SEQCNT[4:0] value by a trigger event. (The setting of the bit field is valid only in single / Sequential modes.)	
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	Reserved
1 0	TRGSEL[1:0]	Trigger selection bit	
		00	Event trigger disable / Software trigger only
		01	Timer event trigger / Software trigger
		10	Reserved
		11	Reserved

NOTES:

1. If ADMOD[1:0] has been set for burst conversion mode, the ADC channels are assigned as BST0CH through BST25CH. 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.)
2. 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.

19.5.2 ADCn_CSCR: ADC Current Sequence/Channel Register

ADCn_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[4:0] in ADCn_CSCR register enables the user to instantly read the current sequence number.

Before setting this register, ADEN in the ADCn_MR register must be enabled.

ADC0_CSCR=0x4100_5004, ADC1_CSCR=0x4100_5104

31 30 29 28 27 26 25 24	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[4:0]	Reserved	CACH[4:0]
-		00000	-	00000
-		R0	-	R0

12 8	CSEQN[4:0]	Current sequence number	
		00000	Current sequence is 0
		00001	Current sequence is 1
		00010	Current sequence is 2
		…	…
		11001	Current sequence is 25
		Others	Reserved
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

19.5.3 ADCn_CCR: ADC Clock Control Register

The ADCn_CCR register controls the clock of the ADC module. This register is 16 bits wide.

ADC0_CCR=0x4100_5008, ADC1_CCR=0x4100_5108

31 30 29 28 27 26 25 24	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 ADC clock = HCLK / (CLKDIV[6:0] + 1) The ADC clock's frequency must not exceed 16 MHz.

19.5.4 ADCn_TRG1: ADC Trigger Select 1 Register

The ADCn_TRG1 register selects trigger sources for the ADC module.

ADC0_TRG1=0x4100_500C, ADC1_TRG=0x4100_510C

31 30 29 28 27 26 25 24	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]
0000	0000	0000	0000
RW	RW	RW	RW

SEQTRG3[3:0]	SEQTRG2[3:0]	SEQTRG1[3:0]	SEQTRG0[3:0] BSTTRG[3:0]
0000	0000	0000	0000
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

19.5.5 ADCn_TRG2: ADC Trigger Select 2 Register

The ADCn_TRG2 register selects trigger sources for the ADC module.

ADC0_TRG2=0x4100_5010, ADC1_TRG=0x4100_5110

31 30 29 28	27 26 25 24	23 22 21 20	19 18 17 16	15 14 13 12	11 10 9 8	7 6 5 4	3 2 1 0
SEQTRG15[3:0]	SEQTRG14[3:0]	SEQTRG13[3:0]	SEQTRG12[3:0]	SEQTRG11[3:0]	SEQTRG10[3:0]	SEQTRG9[3:0]	SEQTRG8[3:0]
0000	0000	0000	0000	0000	0000	0000	0000
RW	RW	RW	RW	RW	RW	RW	RW

30 28	SEQTRG15[3:0]	16th sequence trigger source
27 24	SEQTRG14[3:0]	15th sequence trigger source
23 20	SEQTRG13[3:0]	14th sequence trigger source
19 16	SEQTRG12[3:0]	13th sequence trigger source
15 12	SEQTRG11[3:0]	12th sequence trigger source
11 8	SEQTRG10[3:0]	11th sequence trigger source
7 4	SEQTRG9[3:0]	10th sequence trigger source
3 0	SEQTRG8[3:0]	9th sequence trigger source

19.5.6 ADCn_TRG3: ADC Trigger Select 3 Register

The ADCn_TRG3 register selects trigger sources for the ADC module.

ADC0_TRG3=0x4100_5014, ADC1_TRG=0x4100_5114

31 30 29 28	27 26 25 24	23 22 21 20	19 18 17 16	15 14 13 12	11 10 9 8	7 6 5 4	3 2 1 0
SEQTRG23[3:0]	SEQTRG22[3:0]	SEQTRG21[3:0]	SEQTRG20[3:0]	SEQTRG19[3:0]	SEQTRG18[3:0]	SEQTRG17[3:0]	SEQTRG16[3:0]
0000	0000	0000	0000	0000	0000	0000	0000
RW	RW	RW	RW	RW	RW	RW	RW

30 28	SEQTRG23[3:0]	24th sequence trigger source
27 24	SEQTRG22[3:0]	23rd sequence trigger source
23 20	SEQTRG21[3:0]	22nd sequence trigger source
19 16	SEQTRG20[3:0]	21st sequence trigger source
15 12	SEQTRG19[3:0]	20th sequence trigger source
11 8	SEQTRG18[3:0]	19th sequence trigger source
7 4	SEQTRG17[3:0]	18th sequence trigger source
3 0	SEQTRG16[3:0]	17th sequence trigger source

19.5.7 ADCn_TRG4: ADC Trigger Select 4 Register

The ADCn_TRG4 register selects trigger sources for the ADC module.

ADC0_TRG4=0x4100_5018, ADC1_TRG=0x4100_5118

31 30 29 28 27 26 25 24	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			SEQTRG25[3:0]	SEQTRG24[3:0]
-			0000	0000
.			RW	RW

7 4	SEQTRG25[3:0]	26th sequence trigger source
3 0	SEQTRG24[3:0]	25th sequence trigger source

NOTE:

1. In Multiple mode, the 1st sequence trigger source is given the highest priority, and the 26th sequence trigger source is given the lowest priority. Table 138 shows trigger sources represented by each value. (Refer to section 19.3.16)

19.5.8 ADCn_CSTR1: ADC Channel Sampling Time 1 Register

The ADCn_CSTR1 register is used to set the channel sampling time 1 for the ADC module.

ADC0_CSTR1=0x4100_501C, ADC1_CSTR1=0x4100_511C

31 30 29 28 27 26 25 24	23 22 21 20 19 18 17 16	15 14 13 12 11 10 9 8	7 6 5 4 3 2 1 0
ST3CH[7:0]	ST2CH[7:0]	ST1CH[7:0]	ST0CH[7:0]
0x0000	0x0000	0x0000	0x0000
RW	RW	RW	RW

21 24	ST3CH[7:0]	4th conversion sequence channel sampling time
23 16	ST2CH[7:0]	3rd conversion sequence channel sampling time
15 8	ST1CH[7:0]	2nd conversion sequence channel sampling time
7 0	ST0CH[7:0]	1st conversion sequence channel sampling time

19.5.9 ADCn_CSTR2: ADC Channel Sampling Time 2 Register

The ADCn_CSTR2 register is used to select channel sampling time 2 for the ADC module.

ADC0_CSTR2=0x4100_5020, ADC1_CSTR2=0x4100_5120

31 30 29 28 27 26 25 24	23 22 21 20 19 18 17 16	15 14 13 12 11 10 9 8	7 6 5 4 3 2 1 0
ST7CH[7:0]	ST6CH[7:0]	ST5CH[7:0]	ST4CH[7:0]
0x00	0x00	0x00	0x00
RW	RW	RW	RW

31 24	ST7CH[7:0]	8th conversion sequence channel sampling time
23 16	ST6CH[7:0]	7th conversion sequence channel sampling time
15 8	ST5CH[7:0]	6th conversion sequence channel sampling time
7 0	ST4CH[7:0]	5th conversion sequence channel sampling time

19.5.10 ADCn_CSTR3: ADC Channel Sampling Time 3 Register

The ADCn_CSTR3 register is used to select channel sampling time 3 for the ADC module.

ADC0_CSTR3=0x4100_5024, ADC1_CSTR3=0x4100_5124

31 30 29 28 27 26 25 24	23 22 21 20 19 18 17 16	15 14 13 12 11 10 9 8	7 6 5 4 3 2 1 0
ST11CH[7:0]	ST10CH[7:0]	ST9CH[7:0]	ST8CH[7:0]
0x00	0x00	0x00	0x00
RW	RW	RW	RW

31 24	ST11CH[7:0]	12th conversion sequence channel sampling time
23 16	ST10CH[7:0]	11th conversion sequence channel sampling time
15 8	ST9CH[7:0]	10th conversion sequence channel sampling time
7 0	ST8CH[7:0]	9th conversion sequence channel sampling time

19.5.11 ADCn_CSTR4: ADC Channel Sampling Time 4 Register

The ADCn_CSTR4 register is used to select channel sampling time 4 for the ADC module.

ADC0_CSTR4=0x4100_5028, ADC1_CSTR4=0x4100_5128

31 30 29 28 27 26 25 24	23 22 21 20 19 18 17 16	15 14 13 12 11 10 9 8	7 6 5 4 3 2 1 0
ST15CH[7:0]	ST14CH[7:0]	ST13CH[7:0]	ST12CH[7:0]
0x00	0x00	0x00	0x00
RW	RW	RW	RW

31 24	ST15CH[7:0]	16th conversion sequence channel sampling time
23 16	ST14CH[7:0]	15th conversion sequence channel sampling time
15 8	ST13CH[7:0]	14th conversion sequence channel sampling time
7 0	ST12CH[7:0]	13th conversion sequence channel sampling time

19.5.12 ADCn_CSTR5: ADC Channel Sampling Time 5 Register

The ADCn_CSTR5 register is used to select channel sampling time 5 for the ADC module.

ADC0_CSTR5=0x4100_502C, ADC1_CSTR5=0x4100_512C

31 30 29 28 27 26 25 24	23 22 21 20 19 18 17 16	15 14 13 12 11 10 9 8	7 6 5 4 3 2 1 0
ST19CH[7:0]	ST18CH[7:0]	ST17CH[7:0]	ST16CH[7:0]
0x00	0x00	0x00	0x00
RW	RW	RW	RW

31 24	ST19CH[7:0]	20th conversion sequence channel sampling time
23 16	ST18CH[7:0]	19th conversion sequence channel sampling time
15 8	ST17CH[7:0]	18th conversion sequence channel sampling time
7 0	ST16CH[7:0]	17th conversion sequence channel sampling time

19.5.13 ADCn_CSTR6: ADC Channel Sampling Time 6 Register

The ADCn_CSTR6 register is used to select channel sampling time 6 for the ADC module.

ADC0_CSTR6=0x4100_5030, ADC1_CSTR6=0x4100_5130

31 30 29 28 27 26 25 24	23 22 21 20 19 18 17 16	15 14 13 12 11 10 9 8	7 6 5 4 3 2 1 0
ST23CH[7:0]	ST22CH[7:0]	ST21CH[7:0]	ST20CH[7:0]
0x00	0x00	0x00	0x00
RW	RW	RW	RW

31 24	ST23CH[7:0]	24th conversion sequence channel sampling time
23 16	ST22CH[7:0]	23rd conversion sequence channel sampling time
15 8	ST21CH[7:0]	22nd conversion sequence channel sampling time
7 0	ST20CH[7:0]	21st conversion sequence channel sampling time

19.5.14 ADCn_CSTR7: ADC Channel Sampling Time 7 Register

The ADCn_CSTR7 register is used to select channel sampling time 7 for the ADC module.

ADC0_CSTR7=0x4100_5034, ADC1_CSTR7=0x4100_5134

31 30 29 28 27 26 25 24	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		ST25CH[7:0]	ST24CH[7:0]
-		0x00	0x00
-		RW	RW

15 8	ST25CH[7:0]	26th conversion sequence channel sampling time
7 0	ST24CH[7:0]	25th conversion sequence channel sampling time

19.5.15 ADCn_SCSR1: ADC 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, users must first enable the ADEN in the ADCn_MR register.

ADC0_SCSR1=0x4100_5038, ADC1_SCSR1=0x4100_5138

31 30 29 28 27 26 25 24		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].

19.5.16 ADCn_SCSR2: ADC 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, users must first enable the ADEN in the ADCn_MR register.

ADC0_SCSR2=0x4100_503C, ADC1_SCSR2=0x4100_513C

31 30 29 28 27 26 25 24		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]	5th conversion sequence channel selection

19.5.17 ADCn_SCSR3: ADC Channel Select 3 Register

ADCn_SCSR3 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, users must first enable the ADEN in the ADCn_MR register.

ADC0_SCSR3=0x4100_5040, ADC1_SCSR3=0x4100_5140

31 30 29 28 27 26 25 24		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	SEQ11CH[4:0]	Reserved	SEQ10CH[4:0]	Reserved	SEQ9CH[4:0]	Reserved	SEQ8CH[4:0]
-	00000	-	00000	-	00000	-	00000
-	RW	-	RW	-	RW	-	RW

28 24	SEQ11CH[4:0]	12th conversion sequence channel selection
20 16	SEQ10CH[4:0]	11th conversion sequence channel selection
12 8	SEQ9CH[4:0]	10th conversion sequence channel selection
4 0	SEQ8CH[4:0]	9th conversion sequence channel selection

19.5.18 ADCn_SCSR4: ADC Channel Select 4 Register

ADCn_SCSR4 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, users must first enable the ADEN in the ADCn_MR register.

ADC0_SCSR4=0x4100_5044, ADC1_SCSR4=0x4100_5144

31 30 29 28 27 26 25 24		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	SEQ15CH[4:0]	Reserved	SEQ14CH[4:0]	Reserved	SEQ13CH[4:0]	Reserved	SEQ12CH[4:0]
-	00000	-	00000	-	00000	-	00000
-	RW	-	RW	-	RW	-	RW

28 24	SEQ15CH[4:0]	16th conversion sequence channel selection
20 16	SEQ14CH[4:0]	15th conversion sequence channel selection
12 8	SEQ13CH[4:0]	14th conversion sequence channel selection
4 0	SEQ12CH[4:0]	13th conversion sequence channel selection

19.5.19 ADCn_SCSR5: ADC Channel Select 5 Register

ADCn_SCSR5 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, users must first enable the ADEN in the ADCn_MR register.

ADC0_SCSR5=0x4100_5048, ADC1_SCSR5=0x4100_5148

31 30 29 28 27 26 25 24		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	SEQ19CH[4:0]	Reserved	SEQ18CH[4:0]	Reserved	SEQ17CH[4:0]	Reserved	SEQ16CH[4:0]
-	00000	-	00000	-	00000	-	00000
-	RW	-	RW	-	RW	-	RW

28 24	SEQ19CH[4:0]	20th conversion sequence channel selection
20 16	SEQ18CH[4:0]	19th conversion sequence channel selection
12 8	SEQ17CH[4:0]	18th conversion sequence channel selection
4 0	SEQ16CH[4:0]	17th conversion sequence channel selection

19.5.20 ADCn_SCSR6: ADC Channel Select 6 Register

ADCn_SCSR6 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, users must first enable the ADEN in the ADCn_MR register.

ADC0_SCSR6=0x4100_504C, ADC1_SCSR6=0x4100_514C

31 30 29 28 27 26 25 24		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	SEQ23CH[4:0]	Reserved	SEQ22CH[4:0]	Reserved	SEQ21CH[4:0]	Reserved	SEQ20CH[4:0]
-	00000	-	00000	-	00000	-	00000
-	RW	-	RW	-	RW	-	RW

28 24	SEQ23CH[4:0]	24th conversion sequence channel selection
20 16	SEQ22CH[4:0]	23rd conversion sequence channel selection
12 8	SEQ21CH[4:0]	22nd conversion sequence channel selection
4 0	SEQ20CH[4:0]	21st conversion sequence channel selection

19.5.21 ADCn_SCSR7: ADC Channel Select 7 Register

ADCn_SCSR7 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, users must first enable the ADEN in the ADCn_MR register.

ADC0_SCSR7=0x4100_5050, ADC1_SCSR7=0x4100_5150

31 30 29 28 27 26 25 24	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		SEQ25CH[4:0]	Reserved	SEQ24CH[4:0]
-		00000	-	00000
-		RW	-	RW

12 8	SEQ25CH[4:0]	26th conversion sequence channel selection
4 0	SEQ24CH[4:0]	25th conversion sequence channel selection

19.5.22 ADCn_CR: ADC Control Register

ADCn_CR is used to control the operation of ADC module.

ADC0_CR=0x4100_5054, ADC1_CR=0x4100_5154

31 30 29 28 27 26 25 24	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	AESTART	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	AESTART	ADC all 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[4:0] in ADC_MR register.

19.5.23 ADCn_SR: ADC Status Register

The ADCn_SR register displays the status of the ADC.

ADC0_SR=0x4100_5058, ADC1_SR=0x4100_5158

31	30	29 28 27 26 25 24	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	ACLKBUSY	Reserved		Reserved	CMP1IF	CMP0IF	Reserved	DOVRUN	DMAF	TRGIF	EOSIF	Reserved	EOCIF
0	0	-		-	0	0	-	0	0	0	0	-	0
RO	RO	-		-	RWC1	RWC1	-	RO	RO	RWC1	RWC1	-	RWC1

31	ABUSY	ADC busy status
		0 ADC idle
		1 ADC busy
30	ACLKBUSY	ADC Clock busy status
		0 ADC Clock idle
		1 ADC Clock 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	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, the EOCIF bit must be used.

```
Example)
while ( (ADCn_SR & 0x01) == 1 ) // EOCIF flag checking
{
    ADCn_SR = 0x1;   // EOCIF flag clear
}
…
```

19.5.24 ADCn_IER: ADC Interrupt Enable Register

The ADCn_IER register determines whether to enable the ADC interrupts.

ADC0_IER=0x4100_505C, ADC1_IER=0x4100_515C

31 30 29 28 27 26 25 24	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	DMAOE	DMAIE	TRGIE	EOSIE	Reserved	EOCIE
-			0	0	-	0	0	0	0	-	0
-			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	DMAOE	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.
	0	Disables.
	1	Enables.
0	EOCIE	ADC single conversion interrupt
	0	Disables.
	1	Enables.

NOTE:

1. Burst mode sets the EOSIE = '1' and checks the EOSIF flag in the ADC_SR register.

19.5.25 ADCn_DDR: ADC 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=0x4100_5060, ADC1_DDR=0x4100_5160

31 30 29 28 27 26 25 24	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	TRGINFO[4:0]	Reserved	ADMACH[4:0]	ADDMAR[15:0] ADC DMA temporary data
-	00000	-	00000	0x0000
-	R0	-	R0	R0

28 24	TRGINFO[4:0] (0 to 25)	ADC sequential number information Indicates whether each trigger source has been captured until the end of conversion (EOC). The trigger source currently being processed can be identified by referring to CSEQN[4:0] in the ADC_CSCR register. 0 Disables. 1 Enables.
20 16	ADMACH[4:0]	DMA ADC channel indicator
15 0	ADDMAR[15:0]	DMA ADC conversion result data(12-bit)

NOTE:

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.

19.5.26 ADCn_DRx: ADC Data 0 to 25 Register

The ADCn_DRx register displays the results of ADC conversions. There are eight of these registers.

ADC0_DR0=0x4100_5064, ADC0_DR1=0x4100_5068, ADC0_DR2=0x4100_506C, ADC0_DR3=0x4100_5070, ADC0_DR4=0x4100_5074, ADC0_DR6=0x4100_5078, ADC0_DR6=0x4100_507C, ADC0_DR7=0x4100_5080, ADC0_DR8=0x4100_5084, ADC0_DR9=0x4100_5088, ADC0_DR10=0x4100_508C, ADC0_DR11=0x4100_5090, ADC0_DR12=0x4100_5094, ADC0_DR13=0x4100_5098, ADC0_DR14=0x4100_509C, ADC0_DR15=0x4100_50A0, ADC0_DR16=0x4100_50A4, ADC0_DR17=0x4100_50A8, ADC0_DR18=0x4100_50AC, ADC0_DR19=0x4100_50B0, ADC0_DR20=0x4100_50B4, ADC0_DR21=0x4100_50B8, ADC0_DR22=0x4100_50BC, ADC0_DR23=0x4100_50C0, ADC0_DR24=0x4100_50C4, ADC0_DR25=0x4100_50C8

ADC1_DR0=0x4100_5164, ADC1_DR1=0x4100_5168, ADC1_DR2=0x4100_516C, ADC1_DR3=0x4100_5170, ADC1_DR4=0x4100_5174, ADC1_DR6=0x4100_5178, ADC1_DR6=0x4100_517C, ADC1_DR7=0x4100_5180, ADC1_DR8=0x4100_5184, ADC1_DR9=0x4100_5188, ADC1_DR10=0x4100_518C, ADC1_DR11=0x4100_5190, ADC1_DR12=0x4100_5194, ADC1_DR13=0x4100_5198, ADC1_DR14=0x4100_519C, ADC1_DR15=0x4100_51A0, ADC1_DR16=0x4100_51A4, ADC1_DR17=0x4100_51A8, ADC1_DR18=0x4100_51AC, ADC1_DR19=0x4100_51B0, ADC1_DR20=0x4100_51B4, ADC1_DR21=0x4100_51B8, ADC1_DR22=0x4100_51BC, ADC1_DR23=0x4100_51C0, ADC1_DR24=0x4100_51C4, ADC1_DR25=0x4100_51C8

31 30 29 28 27 26 25 24		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	TRGINFO[4:0]	Reserved	ACH[4:0]	ADDATA[15:0]	
-	00000	-	00000	0x0000	
-	R0	-	R0	R0	

28 24	TRGINFO[4:0] (0 to 25)	ADC sequential number information Indicates whether each trigger source has been captured until the end of conversion (EOC). The trigger source currently being processed can be identified by referring to CSEQN[4:0] in the ADC_CSCR register.
20 16	ACH[4:0]	ADC channel indicator
15 0	ADDATA [15:0]	ADC conversion result data (12-bit)

19.5.27 ADCn_CMP0R: ADC Channel Comparison 0 Register

The ADCn_CMP0R register controls the comparison between the ADC channels.

ADC0_CMP0R=0x4100_50E0, ADC1_CMP0R=0x4100_51E0

31 30 29 28 27 26	25	24	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]	Reserved	CMPIEN	CMPEN	Reserved	CMPM	CCH[4:0]	CVAL[11:0]	Reserved
000000	-	0	0	-	0	00000	0x000	-
RW	-	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.	
24	CMPIEN	Compare interrupt enable bit	
		0	Disables.
		1	Enables.
23	CMPEN	Compare operation enable bit	
		0	Disables.
		1	Enables.
21	CMPM	ADC compare mode bit	
		0	ADC value is higher than CVAL[11:0] value.
		1	ADC value is lower 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
		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	

19.5.28 ADCn_CMP1R: ADC Channel Comparison 1 Register

The ADCn_CMP1R register controls the comparison between the ADC channels.

ADC0_CMP1R=0x4100_50E4, ADC1_CMP1R=0x4100_51E4

31 30 29 28 27 26	25	24	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]	Reserved	CMPIEN	CMPEN	Reserved	CMPM	CCH[4:0]	CVAL[11:0]	Reserved
000000	-	0	0	-	0	00000	0x000	-
RW	-	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.	
24	CMPIEN	Compare interrupt enable bit	
		0	Disable.
		1	Enable.
23	CMPEN	Compare operation enable bit	
		0	Disable.
		1	Enable.
21	CMPM	ADC compare mode bit	
		0	Output when ADC value is higher than CVAL[11:0] value.
		1	Output when ADC value is lower 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
		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	

19.5.29 ADC Register Map Summary

Table 145. 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	ACLKAON	Res.	Res.	SINGLE_EN	SEL_REF	CTIMEFLAT[2:0]			Res.	EXTCONSTAEN	CONTISTAEN	CONTOSTAEN	CDATAT	Res.	DMAEN	Res.	Res.	Res.	SEQTRGT	SEQCNT[4:0]					ADEN	ARST	ADMOD[1:0]		Res.	Res.	TRGSEL[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
0x04	ADCn_CSCR	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	CSEQN[4:0]					Res.	Res.	Res.	CACH[4:0]				
	Reset value																				0	0	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_TRG1	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_TRG2	SEQTRG15[3:0]				SEQTRG14[3:0]				SEQTRG13[3:0]				SEQTRG12[3:0]				SEQTRG11[3:0]			SEQTRG10[3:0]				SEQTRG9[3:0]				SEQTRG8[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
0x14	ADCn_TRG3	SEQTRG23[3:0]				SEQTRG22[3:0]				SEQTRG21[3:0]				SEQTRG20[3:0]				SEQTRG19[3:0]			SEQTRG18[3:0]				SEQTRG17[3:0]				SEQTRG16[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
0x18	ADCn_TRG4	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	SEQTRG25[3:0]				SEQTRG24[3:0]			
	Reset value																									0	0	0	0	0	0	0	0
0x1C	ADCn_CSTR1	ST3CH[7:0]								ST2CH[7:0]								ST1CH[7:0]								ST0CH[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
0x20	ADCn_CSTR2	ST7CH[7:0]								ST6CH[7:0]								ST5CH[7:0]								ST4CH[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
0x24	ADCn_CSTR3	ST11CH[7:0]								ST10CH[7:0]								ST9CH[7:0]								ST8CH[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

Table 145. 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
0x28	ADCn_CSTR4	ST15CH[7:0]								ST14CH[7:0]								ST13CH[7:0]								ST12CH[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
0x2C	ADCn_CSTR5	ST19CH[7:0]								ST18CH[7:0]								ST17CH[7:0]								ST16CH[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
0x30	ADCn_CSTR6	ST23CH[7:0]								ST22CH[7:0]								ST21CH[7:0]								ST20CH[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
0x34	ADCn_CSTR7	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	ST25CH[7:0]								ST24CH[7:0]							
	Reset value																	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
0x38	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
0x3C	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
0x40	ADCn_SCSR3	Res	Res	Res	SEQ11CH[4:0]					Res	Res	Res	SEQ10CH[4:0]					Res	Res	Res	SEQ9CH[4:0]					Res	Res	Res	SEQ8CH[4:0]				
	Reset value				0	0	0	0	0				0	0	0	0	0				0	0	0	0	0				0	0	0	0	0
0x44	ADCn_SCSR4	Res	Res	Res	SEQ15CH[4:0]					Res	Res	Res	SEQ14CH[4:0]					Res	Res	Res	SEQ13CH[4:0]					Res	Res	Res	SEQ12CH[4:0]				
	Reset value				0	0	0	0	0				0	0	0	0	0				0	0	0	0	0				0	0	0	0	0
0x48	ADCn_SCSR5	Res	Res	Res	SEQ19CH[4:0]					Res	Res	Res	SEQ18CH[4:0]					Res	Res	Res	SEQ17CH[4:0]					Res	Res	Res	SEQ16CH[4:0]				
	Reset value				0	0	0	0	0				0	0	0	0	0				0	0	0	0	0				0	0	0	0	0
0x4C	ADCn_SCSR6	Res	Res	Res	SEQ23CH[4:0]					Res	Res	Res	SEQ22CH[4:0]					Res	Res	Res	SEQ21CH[4:0]					Res	Res	Res	SEQ20CH[4:0]				
	Reset value				0	0	0	0	0				0	0	0	0	0				0	0	0	0	0				0	0	0	0	0

Table 145. 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
0x50	ADCn_SCSR7	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	SEQ25CH[4:0]					Res.	Res.	Res.	SEQ24CH[4:0]				
	Reset value																				0	0	0	0	0				0	0	0	0	0
0x54	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.	AESTART	Res.	Res.	TRGCLR	ASTART
	Reset value																									0			0			0	0
0x58	ADCn_SR	ABUSY	ACLKBUS	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	CMP1IF	CMP0IF	Res.	Res.	DOVRUN	DMAF	TRGIF	EOSIF	Res.	EOCIF
	Reset value	0	0																					0	0			0	0	0	0		0
0x5C	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	Res.	Res.	DMAOE	DMAIE	TRGIE	EOSIE	Res.	EOCIE
	Reset value																							0	0			0	0	0	0		0
0x60	ADCn_DDR	Res.	Res.	Res.	TRGINFO[4:0]					Res.	Res.	Res.	ADMACH[4:0]					ADDMAR[15:0] ADC DMA temporary data															
	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
0x64-0xC8	ADCn_DRx	Res.	Res.	Res.	TRGINFO[4:0]					Res.	Res.	Res.	ACH[4:0]					ADDATA[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
0xE0	ADCn_CMP0R	MATCH_CNT[5:0]						Res.	CMPIEN	CMPEN	Res.	CMPM	CCH[4:0]					CVAL[11:0]													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				
0xE4	ADCn_CMP1R	MATCH_CNT[5:0]						Res.	CMPIEN	CMPEN	Res.	CMPM	CCH[4:0]					CVAL[11:0]													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				

20. VREFBUF (Voltage Reference Buffer)

20.1 VREFBUF Introduction

The devices embed a voltage reference buffer which can be used as voltage reference for ADCs, DACs and also as voltage reference for external components through the AREF+ pin.

An external voltage reference can be provided through the AREF+ pin when the internal voltage reference buffer is off. Even when external voltage reference is used, it is needed to select reference source via VREFBUF_CR register.

VREFBUF must be set before using ADC, DAC, and OPAMP.

20.2 VREFBUF Functional Description

20.2.1 VREFBUF Reset and Clock

The clocks of the VREFBUF operate based on the PCLK clock in the APB domain.

To enable the VREFBUF, the VREFBUF in SCU_PER1 register must be set to ‘1’ as shown below:

- If the VREFBUF in SCU_PER1 register = ‘0’, the VREFBUF peripheral is disabled.
- If the VREFBUF in SCU_PER1 register = ‘1’, the VREFBUF peripheral is enabled.

To provide the clock to the VREFBUF, the VREFBUF in SCU_PCER1 register must be set to ‘1’ as shown below:

- If the VREFBUF in SCU_PCER1 register = ‘0’, the VREFBUF peripheral clock is disabled.
- If the VREFBUF in SCU_PCER1 register = ‘1’, the VREFBUF peripheral clock is enabled.

To initialize the VREFBUF peripheral registers, each VREFBUF in the SCU_PER1 and SCU_PCER1 registers must be set to ‘0’ and then set to ‘1’.

20.2.2 VREFBUF Mode

The internal voltage reference buffer supports two voltages(1), which are configured with VREF_SEL bits in the VREFBUF_CR register:

- VREFPSEL= 0: VREFP_OUT1 around 2.0 V.
- VREFPSEL= 1: VREFP_OUT2 around 2.5 V.

The internal voltage reference can be configured in four different modes depending on MODE_SEL bits configuration. These modes are provided in the table below:

Table 146. VREFBUF Modes

MODESEL	Description
000	Reserved
001	• External voltage reference mode - VREFBUF buffer off - AREF+ pin input mode: AREF+ ≤ (AV_{DD} = V_{DD}) - Implementation of 0.1uF capacitor on the AREF+ pin
010	• Bypass voltage reference mode - VREFBUF buffer off - AV_{DD} pin input mode (AV_{DD} = V_{DD}) - Because the AREF+ pin is internally shorted to AV_{DD}, the AREF+ pin must be left unconnected.
100	• Internal voltage reference mode - VREFBUF buffer on (2V or 2.5V) - AREF+ pin connected load capacitor (0.5 to 1.5 μF) - V_{DD} = AV_{DD}

NOTE:

1. The minimum AV_{DD} voltage depends on VREF_SEL setting, refer to the product datasheet.

20.2.3 Voltage Reference Output Initialization

The stabilization time for the voltage reference output is required for each mode, as shown below:

- External: No stabilization time required
- Bypass mode: 2.5 ms
- Internal mode: 2 ms

VREFBUF has an up-count Timer function that can count the stabilization time of the voltage reference output.

When the MODESEL in VREFBUF_CR register is set to start the VREFBUF counter, if the counter value reaches the VREFSTIME value, the VREFACT is set to 1 and the interrupt occurs.

Internal mode stabilization time = PCLK period × VREF_STIME

For example, when the PCLK is 32 MHz:

$$\frac{1}{f_{PCLK}} \times \text{VREF\_STIME} = 0.03125 \times 64000 = 2000, \qquad \text{VREF\_STIME} = 64000,$$

$$\text{Internal Mode Stabilization Time} = 2\ \text{ms}$$

Bypass mode stabilization time = PCLK period × VREF_STIME × 1.25

For example, when the PCLK is 32 MHz:

$$\frac{1}{f_{PCLK}} \times \text{VREF\_STIME} = 0.03125 \times 64000 \times 1.25 = 2500, \qquad \text{VREF\_STIME} = 64000,$$

$$\text{Bypass Mode stabilization time} = 2.5\ \text{ms}$$

When bypass mode was selected, the PCLK period is automatically multiplied by 1.25 internally, so the user only needs to set the 2 ms count based on Internal mode.

20.3 VREFBUF Interrupt

For VREFBUF, an interrupt can be generated by setting below:

- VREF is stable (INTEN)

20.4 VREFBUF Registers

The base address and register map of the VREFBUF block are described in the following tables:

Table 147. Base Address of VREFBUF

Name	Base Address
VREFBUF	0x4000_B000

Table 148. VREFBUF Register Map

Name	Offset	Type	Description	Reset Value	Reference
VREFBUF_SR	0x0000	RW	Status Register	0x0000_0000	20.4.1
VREFBUF_CR	0x0004	RW	Configure Register	0x0000_0000	20.4.2
VREFBUF_TCR	0x0008	RW	Time Control Register	0x0001_FFFF	20.4.3

20.4.1 VREFBUF_SR: VREFBUF Status Register

The VREFBUF_SR register displays the status of the VREFBUF.

VREFBUF_SR=0x4000_B000

31 30 29 28 27 26 25 24	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			VREF_ACT
-			0
-			RWC1

0	VREF_ACT	VREF is stable	
		0	VREF is unstable.
		1	VREF is stable.

20.4.2 VREFBUF_CR: VREFBUF Control Register

VREFBUF_CR is used to control VREFBUF.

VREFBUF_CR=0x4000_B004

31 30 29 28 27 26 25 24	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			VREFP_SEL	Reserved	INTEN	Reserved	MODE_SEL[2:0]
-			0	-	0	-	0 0 0
-			RW	-	RW	-	RW

8	VREFP_SEL	Selects internal ref voltage if MODE_SEL[2:0] = 100.	
		0	2.0 V
		1	2.5 V
4	INTEN	Uses interrupts.	
		0	Disables.
		1	Enables.
2 0	MODE_SEL[(1)]	Selects VREF mode.	
		000	Reserved
		001	External source (External mode; It can be used immediately.)
		010	VDDEXT (Bypass mode, 2.5 ms)
		100	Internal (2.0 V or 2.5 V) source (Buf mode, 2.0 ms)

NOTE:

1. Using VREF voltage, users can select between REF voltage (External mode), VDDEXT voltage (Bypass mode), and internal REF voltage (2.0 V/ 2.5 V: Buf mode). Since Bypass mode is implemented by connecting VDDEXT and AREF+ outside the IC and then selecting External mode, only External and Buf modes are actually meaningful.
   External mode can be used immediately after VREFBUF block is enabled, Buf mode requires about 2 ms stabilization time after VREFBUF block is enabled, and Bypass mode requires 2.5 ms stabilization time.
   In DEEP-SLEEP mode, MODE_SEL bit[2:0] is automatically cleared to 000.

20.4.3 VREFBUF_TCR: VREFBUF Voltage Control Register

VREFBUF_TCR register.

VREFBUF_TCR=0x4000_B008

31 30 29 28 27 26 25 24	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		VREF_STIME[16:0]	
-		0 0 0 0 0 0 0 0	0 0 0 0 0 0 0 0
-		RW	

16 0	VREF_STIME[16:0]	VREF stable time Stable time = fPCLK_VREFBUF × VREF_STIME

20.4.4 VREFBUF Register Map Summary

Table 149. VREFBUF 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	VREFBUF_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.	Res.	Res.	VREF_ACT
	Reset value																																0
0x04	VREFBUF_CR	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	VREF_SEL	Res.	Res.	Res.	INTEN	Res.	MODE_SEL[2:0]		
	Reset value																								0				0		0	0	0
0x08	VREFBUF_TCR	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	VREF_STIME[16:0]																
	Reset value																0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

21. OPAMP (Operational Amplifier)

21.1 OPAMP Introduction

The following are characteristics of the operational amplifiers (OPAMP) built in the A34L716. In the OPAMPn_CR register, the OPAMP's control signals can be set.

21.2 OPAMP Main Features

The OPAMP of the A34L716 features the following:

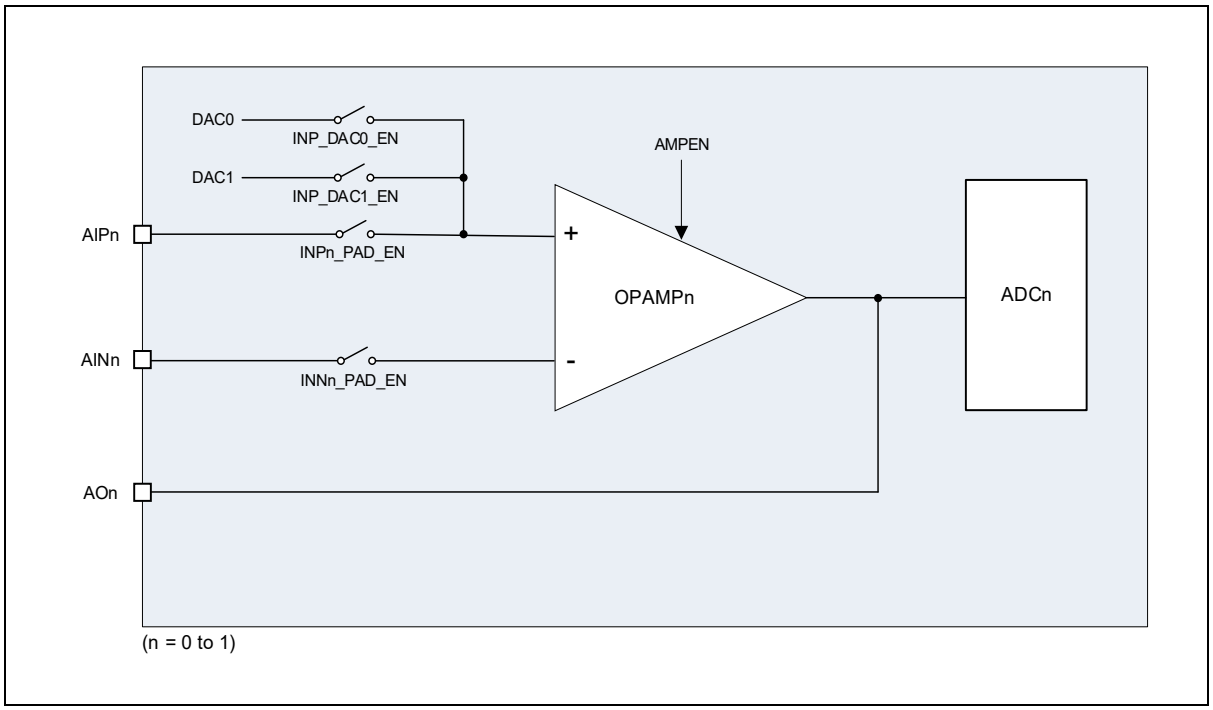
- Two OPAMP units are available.
- OPAMP outputs are used in connection with ADC channels.

21.3 OPAMP Functional Description

21.3.1 OPAMP Block Diagram

Figure 172 shows a block diagram of the OPAMPs.

Figure 172. OPAMP Block Diagram



21.3.2 OPAMP Pins and Signals

Table 150 shows pins assigned for the OPAMP.

Table 150. Pin Assignment of OPAMP External Pins

Pin Name	Signal Type(1)	Description	Supported packages		
			A34L716 (100-LQFP)	A34L716 (64-LQFP)	A34L716 (56-QFN)
AIN0	A	Analog Input 0 (-)	O	O	O
AIP0	A	Analog Input 0 (+)	O	O	O
AO0	O	Opamp Output 0	O	O	O
AIN1	A	Analog Input 1 (-)	O	O	O
AIP1	A	Analog Input 1 (+)	O	O	O
AO1	O	Opamp Output 1	O	O	O

NOTE:

1. P: Power, O: Output, A: Analog Input.

Table 151. OPAMP Input/Output Signals

Signal Name	Pin	Internal	Description
INP0_PAD_EN	PA1 (AN1)	adc_ch1	OPAMP0 input voltage from PA1
INN0_PAD_EN	PA2 (AN2)	adc_ch2	OPAMP0 input voltage from PA2
AO0	PA3 (AN3)	adc_ch3	OPAMP0 output voltage is connected to PA3
INP1_PAD_EN	PA4 (AN4)	adc_ch4	OPAMP1 input voltage from PA4
INN1_PAD_EN	PA5 (AN5)	adc_ch5	OPAMP1 input voltage from PA5
AO1	PA6 (AN6)	adc_ch6	OPAMP1 output voltage is connected to PA6

21.3.3 OPAMP Reset and Clock

The clocks of the OPAMP operate based on the PCLK clock in the APB domain, and all units of the OPAMP use a common clock source.

To enable the OPAMP, the OPAMP in SCU_PER2 register must be set to '1' as shown below:

- If the OPAMP in SCU_PER2 register = '0', the OPAMP peripheral is disabled.
- If the OPAMP in SCU_PER2 register = '1', the OPAMP peripheral is enabled.

To provide the clock to the OPAMP, the OPAMP in SCU_PCER2 register must be set to '1' as shown below:

- If the OPAMP in SCU_PCER2 register = '0', the OPAMP peripheral clock is disabled.
- If the OPAMP in SCU_PCER2 register = '1', the OPAMP peripheral clock is enabled.

To initialize the OPAMP peripheral registers, each OPAMP in the SCU_PER2 and SCU_PCER2 registers must be set to '0' and then set to '1'.

An external clock source can be set as a clock source for the OPAMP. The external clock source can be selected as the OPAMP's clock source when the OPAMPCDIV[7:0] in SCU_MCCR7 register is set to a certain value larger than '0' and the OPAMPCSEL[2:0] in SCU_MCCR7 register is set to select the external clock source.

IMPORTANT:

1. The input clock of OPAMP must be 50 kHz and must be set by users.
2. LSE (32.768 kHz), LSI (40 kHz), and PLL cannot be used as a clock.

21.3.4 Initial Configuration

The OPAMP analog inputs must be configured to analog mode using the corresponding bits of PA_MR1 register.

OPAMP's reference voltage operates based on the VDD and GND signals. OPAMP's analog output results must be set up by software to perform the A/D conversion in the ADC module.

21.3.5 OPAMP Modes

Immediately after the AMPEN in the OPAMPn_CR register is set to '1', the input pin and output pin of the OPAMP is connected to the ADC. If the OPAMP is enabled in the initialization stage, it requires the stabilization time of Typ. 20 us:

- If the AMPEN in OPAMPn_CR register = '0', OPAMP operation is disabled.
- If the AMPEN in OPAMPn_CR register = '1', OPAMP operation is enabled.

21.4 OPAMP Low-Power Mode

Table 152. Effect of Low-Power Mode on OPAMP

Mode	Description
SLEEP	No effect
DEEP-SLEEP	The OPAMP peripheral and clock are disabled. OPAMP peripheral must be re-initialized after waking from DEEP-SLEEP mode.

21.5 OPAMP Registers

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

Table 153. Base Address of OPAMP

Name	Base Address
OPAMP	0x4000_B300

Table 154. OPAMP Register Map

Name	Offset	Type	Description	Reset Value	Reference
OPAMP_MR	0x0000	RW	OPAMP mode control register	0x0000_0000	21.5.1
OPAMP0_CR	0x0004	RW	OPAMP0 control register	0x0000_0300	21.5.2
OPAMP1_CR	0x0008	RW	OPAMP1 control register	0x0000_0300	21.5.2

21.5.1 OPAMP_MR: OPAMP Mode Control Register

OPAMP_MR is used to control OPAMP0,1

OPAMP_MR=0x4000_B300

31 30 29 28 27 26 25 24	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			LP_MODE_EN
-			0
-			RW

0	LP_MODE_EN	Low-power mode enable Only '0' should be used.	
		0	Disables.
		1	Enables.

21.5.2 OPAMPn_CR: OPAMP n Control Register

OPAMPn_CR is used to control OPAMP0/1. Reserved bits must be '0'.

OPAMP0_CR=0x4000_B304
OPAMP1_CR=0x4000_B308

31 30 29 28 27 26 25 24	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		INN_PAD_EN	INP_PAD_EN	INP_DAC1_EN	INP_DAC0_EN	Reserved		AMPEN
-		0	0	0	0	-		0
-		RW	RW	RW	RW	-		RW

19	INN_PAD_EN	OPAMP INN is connected to PAD
		0 Disables.
		1 Enables.
18	INP_PAD_EN	OPAMP INP is connected to PAD
		0 Disables.
		1 Enables.
17	INP_DAC1_EN	OPAMP INP is connected to DAC1
		0 Disables.
		1 Enables.
16	INP_DAC0_EN	OPAMP INP is connected to DAC0
		0 Disables.
		1 Enables.
0	AMPEN	Whether to enable or disable the OPAMP
		0 Disables.
		1 Enables.

21.5.3 OPAMP Register Map Summary

Table 155. OPAMP 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	OPAMP_MR	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.	LP_MODE_EN
	Reset value																																0
0x04	OPAMP0_CR	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	INN_PAD_EN	INP_PAD_EN	INP_DAC1_EN	INP_DAC0_EN	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	AMPEN
	Reset value													0	0	0	0																0
0x08	OPAMP1_CR	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	INN_PAD_EN	INP_PAD_EN	INP_DAC1_EN	INP_DAC0_EN	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	AMPEN
	Reset value													0	0	0	0																0

22. DAC (Digital-to-Analog Converter)

22.1 DAC Introduction

The DAC module is a 12-bit, voltage-output digital-to-analog converter. The DAC features two output channels, each with its own converter. An input reference pin, AREF+ (shared with others analog peripherals) is available for better resolution. An internal reference can also be set on the same input. Refer to voltage reference buffer (VREFBUF) section.

The DAOn pin can be used as general-purpose input/output (GPIO) when the DAC output is disconnected from output pad and connected to on-chip peripheral.

The DAC output buffer can be optionally enabled to allow a high drive output current.

22.2 DAC Main Features

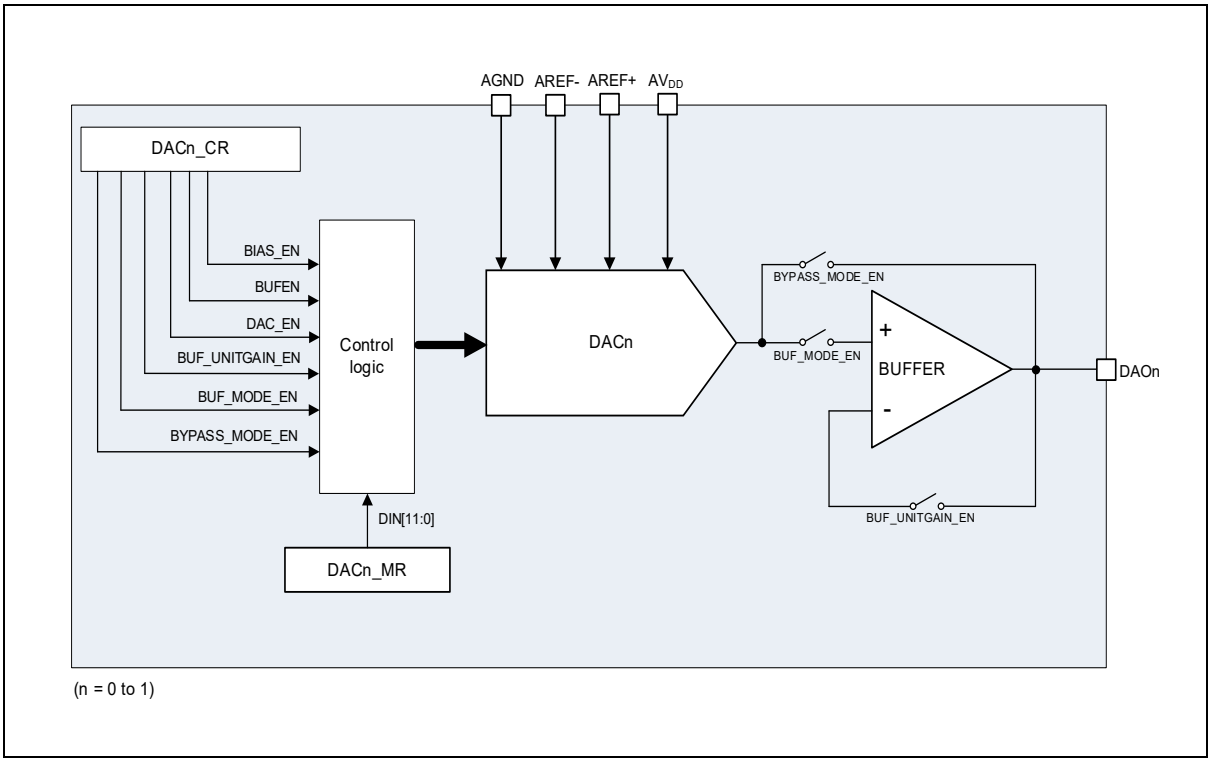
The DAC of A34L716 has the following features.

- Two DAC converters: one output channel each.
- Synchronized update capability.
- Dual DAC channel for independent conversions.
- DAC output channel buffered/unbuffered modes.
- Each DAC output can be disconnected from the DAOn output pin.
- DAC output connection to on chip peripherals.
- Input voltage reference, AREF+

22.3 DAC Functional Description

22.3.1 DAC Block Diagram

Figure 173. DAC Block Diagram



22.3.2 DAC Pins and Signals

Table 156 describes the pins assigned for the DAC.

Table 156. Pin Assignment of DAC External Pins

Pin Name	Signal Type(1)	Description	Supported Packages		
			A34L716 (100-LQFP)	A34L716 (64-LQFP)	A34L716 (56-QFN)
AREF+	P	Input, analog reference positive The higher/positive reference voltage for the DAC, AREF+ ≤ AVDD (refer to datasheet)	O	X	X
AREF-	P	Input, analog reference negative AREF- = AGND (refer to datasheet)	O	X	X
AVDD	P	Input, analog supply AVDD = VDD (refer to datasheet)	O	O	O
AGND	P	Input, analog supply ground	O	O	O
DAO0	O	Digital to analog output 0	O	O	O
DAO1	O	Digital to analog output 1	O	O	O

NOTE:

1. P: Power, O: Output, A: Analog Input

22.3.3 DAC Reset and Clock

The clocks of the DACn operate based on the PCLK clock in the APB domain, and all units of the DACn use a common clock source.

To enable the DACn, the DACn in the SCU_PER2 register must be set to '1' as shown below:

- If the DACn in SCU_PER2 register = '0', the DACn peripheral is disabled.
- If the DACn in SCU_PER2 register = '1', the DACn peripheral is enabled.

To provide the clock to the DACn, the DACn in the SCU_PCER2 register must be set to '1' as shown below:

- If the DACn in SCU_PCER2 register = '0', the DACn peripheral clock is disabled.
- If the DACn in SCU_PCER2 register = '1', the DACn peripheral clock is enabled.

To initialize the DACn peripheral registers, each DACn in the SCU_PER2 and SCU_PCER2 registers must be set to '0' and then set to '1'.

22.3.4 DAC Channel Enable

Each DAC channel can be powered on by setting its corresponding DAC_EN and BIAS_EN bit in the DACn_CR register. The DAC channel is then enabled after a startup time T_{WAKEUP}.

VREFBUF settings are required to use DAC. See chapter 20.2.

22.3.5 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, BUF_MODE_EN bit in the DACn_CR register.

DAC Output Voltage

Digital inputs are converted to output voltages on a linear conversion between 0 and AREF+.

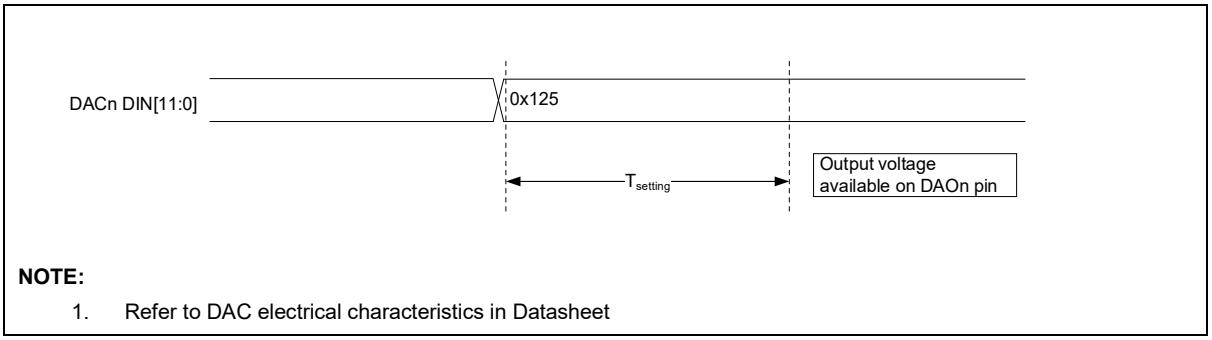
The analog output voltages on each DAC channel pin are determined by the following equation:

$$\mathrm{DAOn} = AREF + \times \left(\frac{\mathrm{DIN}}{4096}\right)$$

22.3.6 DAC Conversion

When a value is written to the DIN in DACn_MR register, the analog output voltage becomes available after a settling time $T_{setting}$, which varies depending on the power supply voltage and the analog output load.

Figure 174. DAC Conversion



NOTE:

1. Refer to DAC electrical characteristics in Datasheet

22.4 DAC Registers

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

Table 157. Base Addresses of DAC

Name	Base Address
DAC0	0x4000_B800
DAC1	0x4000_B900

Table 158. DAC Register Map

Name	Offset	Type	Description	Reset Value	Reference
DACn_MR	0x0000	RW	DACn mode register	0x0000_0000	22.4.1
DACn_CR	0x0004	RW	DACn control register	0x0000_0000	22.4.2

NOTE:

1. n = 0, 1.

22.4.1 DACn_MR: DACn Mode Register

The DAC_MR register is a 12-bit input data register of DACn.

DAC0_MR=0x4000_B800, DAC1_MR=0x4000_B900

31	30	29	28	27	26	25	24	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																DIN[11:0]															
-																0	0	0	0	0	0	0	0	0	0	0	0				
-																RW															

11 0	DIN[11:0]	DAC input data bits

22.4.2 DACn_CR: DACn Control Register

The DAC_CR register is used to control the DAC. Reserved bits must be '0'.

DAC0_CR=0x4000_B804, DAC1_CR=0x4000_B904

31 30 29 28 27 26 25 24	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			BIAS_EN	Reserved	BUFEN	DAC_EN	Reserved	BUF_UNITGAIN_EN	Reserved	BUF_MODE_EN	BYPASS_MODE_EN
-			0	-	0	0	-	0	-	0	0
-			RW	-	RW	RW	-	RW	-	RW	RW

14	BIAS_EN	DAC IBIAS REF Enable Bit	
		0	Disable
		1	Enable
9	BUFEN	DAC Buffer Enable Bit	
		0	Disable
		1	Enable
8	DAC_EN	DAC Enable Bit	
		0	Disable
		1	Enable
5	BUF_UNITGAIN_EN	DAC buffer unit gain mode	
		0	Disable
		1	Enable
1	BUF_MODE_EN	DAC is connected to buffer	
		0	Disable
		1	Enable
0	BYPASS_MODE_EN	DAC Buffer bypass mode (Buffer must be turned off)	
		0	Disable
		1	Enable

22.4.3 DAC Register Map Summary

Table 159. 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	DAC_MR	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	DIN[11:0]											
	Reset value																					0	0	0	0	0	0	0	0	0	0	0	0
0x04	DAC_CR	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	BIAS_EN	Res	Res	Res	Res	BUFEN	DAC_EN	Res	Res	BUF_UNITGAIN_EN	Res	Res	Res	BUF_MODE_EN	BYPASS_MODE_EN
	Reset value																		0					0	0			0				0	0

23. CRC (Cyclic Redundancy Check 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 A34L716 supports DMA.

23.2 CRC Main Features

The CRC of the A34L716 series features the following:

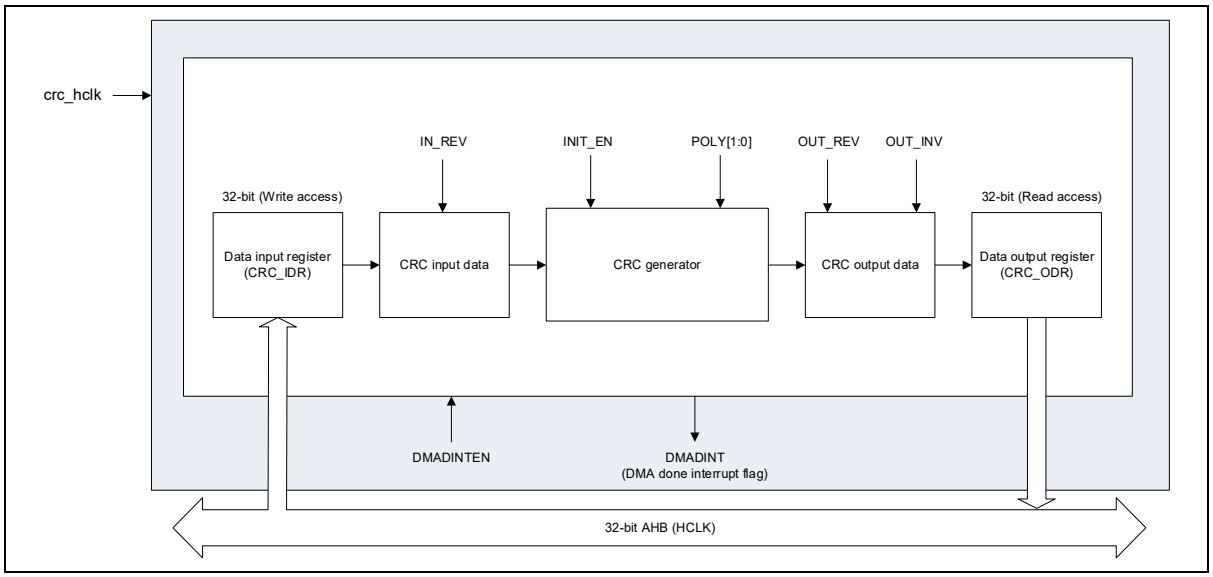
- Handles 8-bit, 32-bit data size.
- Input buffer to avoid bus stall during calculation.
- CRC calculation automatically done for word / byte input data size.
  - Four AHB clock cycles (HCLK) for 32-bit
  - Two AHB clock cycles (HCLK) for 16-bit
  - One AHB clock cycles (HCLK) for 8-bit
- CRC code selectable 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 175 shows a block diagram of the A34L716 CRC module.

Figure 175. CRC Block Diagram



23.3.2 CRC Internal Signals

Table 160. 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 176 shows a timing diagram of 8-bit size input data and CRC-8 calculation (outputs). Signals in Figure 176 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 AHB clock cycle.

Figure 176. Timing Diagram of CRC Internal Signals: 8-bit Size Input and CRC-8 Outputs

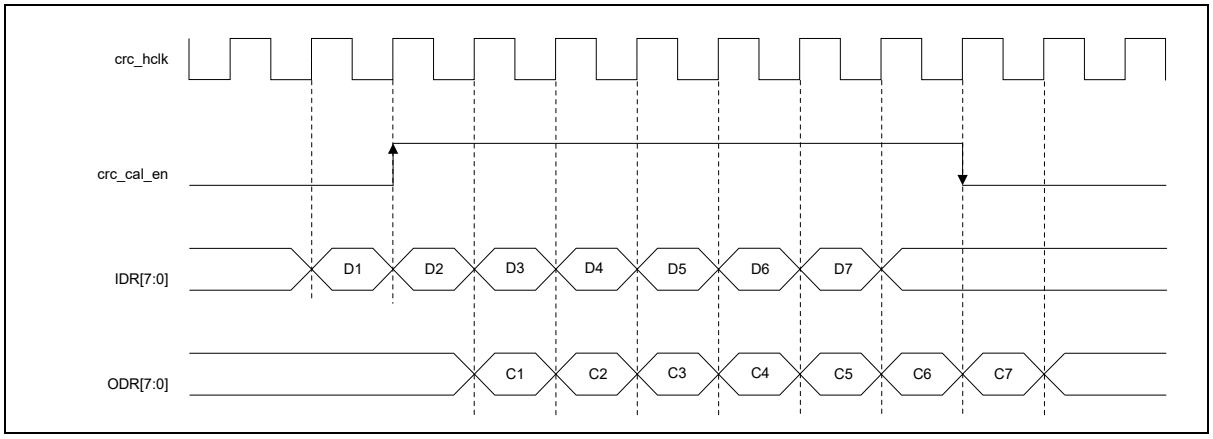


Figure 177 shows a timing diagram of 32-bit size input data and CRC-8 calculation (outputs).

Signals in Figure 177 show that the 32-bit size input data enters at once and the CRC-8 calculated values are output from the point of one AHB clock cycle.

Figure 177. Timing Diagram of CRC Internal Signals: 32-bit Size Input and CRC-8 Outputs

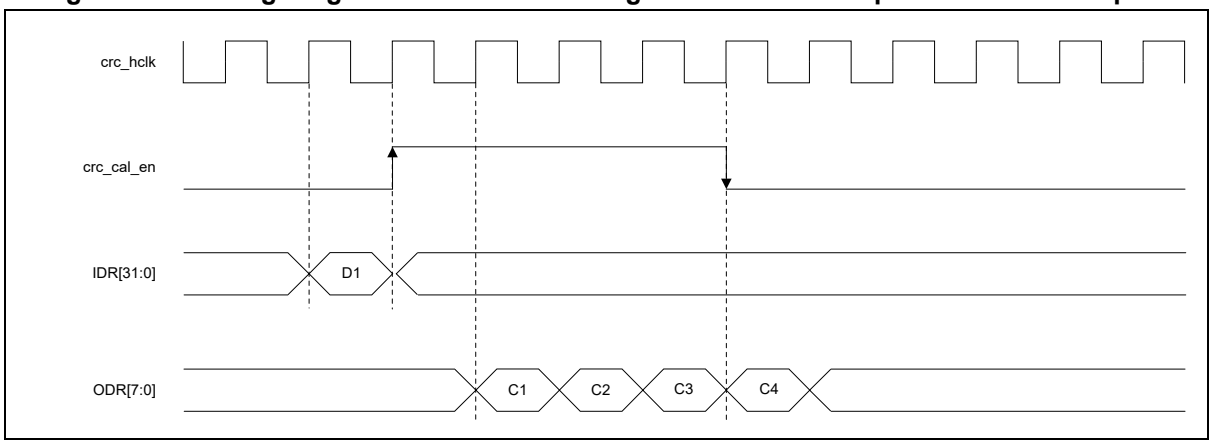


Figure 178 shows a timing diagram of 32-bit size input data and CRC-16 calculation (outputs).

Signals in Figure 178 show that a 32-bit size input datum enters 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 178. Timing Diagram of CRC Internal Signals: 32-bit Size Input and CRC-16 Outputs

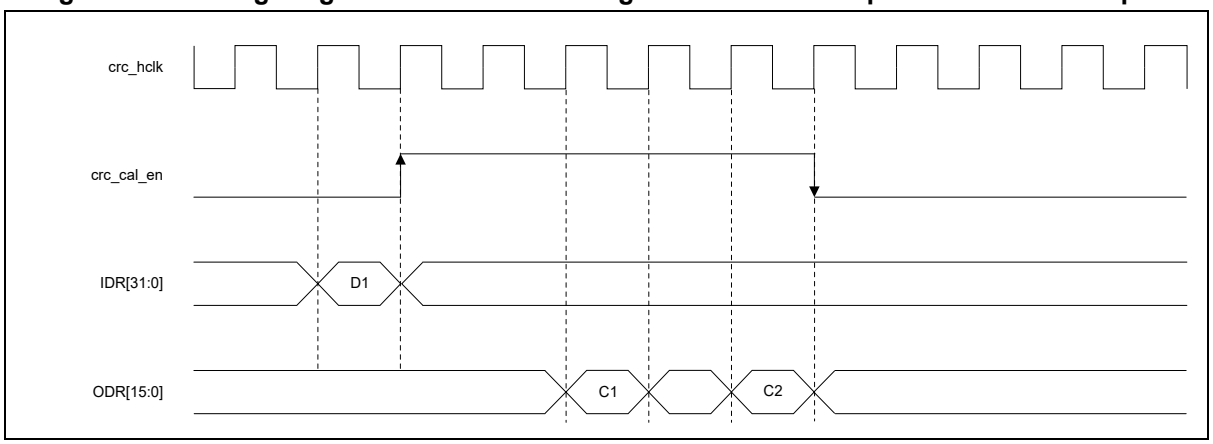
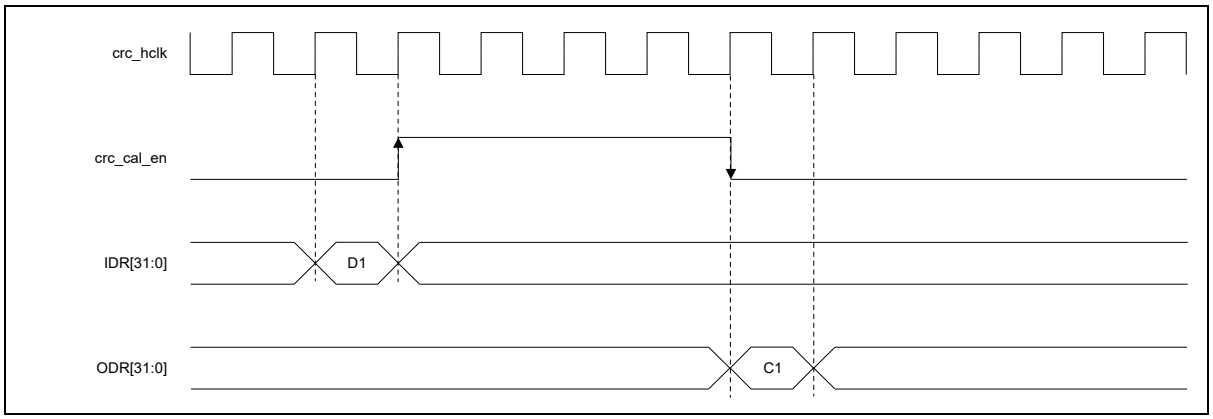


Figure 179 shows a timing diagram of 32-bit size input data and CRC-32 calculation (outputs).

Signals in Figure 179 show that a 32-bit size input datum enters at once and the CRC-32 calculated values are output from the point of four AHB clock cycles.

Figure 179. 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 in 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 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 in 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 in CRC_CTRL register is converted using bit reversal. For example, '0x1122_3344' is converted to '0x22CC_4488'.

The OUT_INV in 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 is set to ‘1’, each byte of CRC output data is XORed with ‘0xFF’ before it is output; while if the OUT_INV 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] 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’.

23.3.5 CRC Using DMA

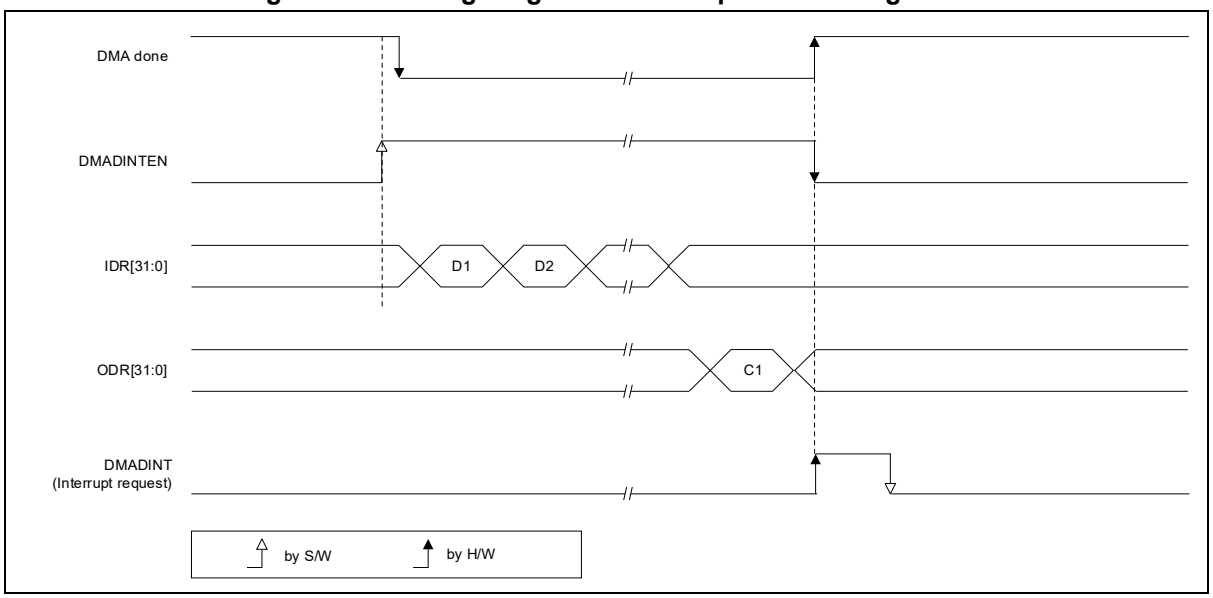
The CRC module supports DMA function.

The 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 to the following.
   A. The TRANSCNT[11:0] bits determine the number of DMA transfers.
   B. The PERISEL[3:0] bits select the CRC_Tx (8).
   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 0x4000_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 in the DMAn_SR register t o transfer the CRC value using DMA.
6. Once the DMA transfer is started and when value of the TRANSCNT[11:0] in the DMAn_CR r egister reaches ‘0’, the EOT in DMAn_SR register is enabled indicating that the DMA transfe r ends.
7. For detailed information on the DMA operation, refer to chapter 8.

Figure 180 shows a timing diagram of the CRC operation using DMA.

Figure 180. 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 161. Base Address of CRC

Name	Base Address
CRC	0x4100_2000

Table 162. 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	0x0000_0000	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 is used to control the CRC.

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	Whether to enable or disable CRC output data inversion	
		0	Disables.
		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	Whether to enable or disable the DMA done interrupt	
		0	Disables.
		1	Enables.
2 1	POLY[1:0]	Polynomial selection bit	
		00	CRC32 (0x04C1_1DB7)
		01	CRC16_USB (0x8005)
		10	CRC16_CCITT (0x1021)
		11	CRC8_CCITT (0x07)
0	INIT_EN	Whether or not 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]																															
0xFFFFFFFF																															
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 in CRC_CTRL register must be enabled. For example, writing '0x8005' to CRC_INIT register and writing a '1' to the INIT_EN in CRC_CTRL register changes the internal CRC initial register to '0x8005'.

23.4.3 CRC_IDR: CRC Input Data Register

The CRC_IDR register is able to 32, 16 and 8-bit access. 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 data is entered in this bit field, its polynomial result is automatically written to the CRC_ODR register.

23.4.4 CRC_ODR: CRC Output Data Register

The CRC_ODR register is available for 32-, 16- and 8-bit access. 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]			
0xFFFFFFFF			
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 163. 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	0	0	0	0	0	0	0	0	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	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
0x0C	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								

24. RTCC (Real Time Clock and Calendar)

24.1 RTCC Introduction

Real-Time Clock (RTCC) of the A34L716 series provides an automatic wakeup function to manage all low-power modes.

The RTCC is an independent BCD timer/counter. It provides a time-of-day clock/calendar with programmable alarm interrupt. In addition, it includes a periodic programmable wakeup flag with interrupt capability. Eight RTCC counters contain the sub-seconds, seconds, minutes, hours, week, day, month, and year expressed in binary coded decimal format (BCD). Compensations for 28-, 29-(leap year), 30-, and 31-day months are performed automatically. An error correction is available to compensate for variations in crystal oscillator accuracy.

In any mode of operation, the sub-second counter is never accessible. In normal operating mode, seven RTCC counters are protected from read and write access. Read and write operation is possible only in RW mode. If the supply voltage remains in the operating range, the RTCC never stops, regardless of the status of the device.

24.2 RTC Main Features

The RTCC of the A34L716 series has the following features.

- Using MCCR2 clock as RTCC clock (f_{RTCC})
- Counters of year, month, week, day, hour, minute, and second (up to 99 years)
- Constant-period matching interrupt function (period: 0.5 seconds, 1 second, 1 minute, 1 hour, 1 day, 1 month)
- Alarm interrupt function (alarm: week, hour, minute)
- Timestamp interrupt function.
- Pin output of 1 Hz clock.
- Error correction by 1.53 ppm resolution when using 32,768 Hz LSE oscillator.

24.3 RTCC Functional Description

RTCC clock (f_{RTCC}) input to the sub-second counter is determined by MCCR2. By setting the SUBDR register value corresponding to the RTC clock, the sub-second counter output, that is, input clock of the SEC, can be 1 Hz.

In Standby mode or backup power mode, only LSE and LSI40kHz operate, so the clock source must be changed.

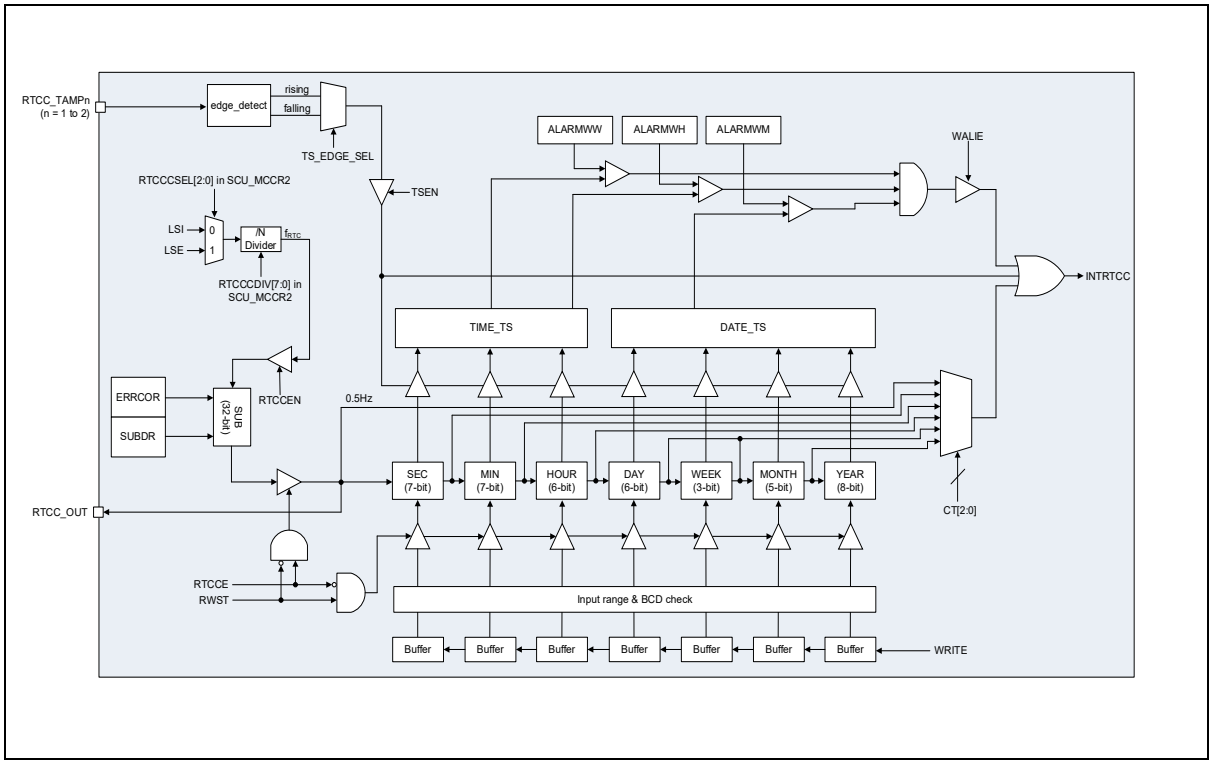
Table 164. Base Address of RTCC

RTCCCSSEL	RTCDIV	f_{RTC}	SUBDR	ERROR
LSI	1	40 kHz	0x0000_9C40 – 1	-
LSE	1	32.768 kHz	0x0000_8000 – 1	-

24.3.1 RTCC Block Diagram

Figure 181 shows a block diagram of the A34L716 RTCC module.

Figure 181. RTCC Block Diagram



24.3.2 RTCC Pins and Signals

The external pins assigned for RTCC are described in Table 165.

Table 165. Pin Assignment of RTCC: External Pins

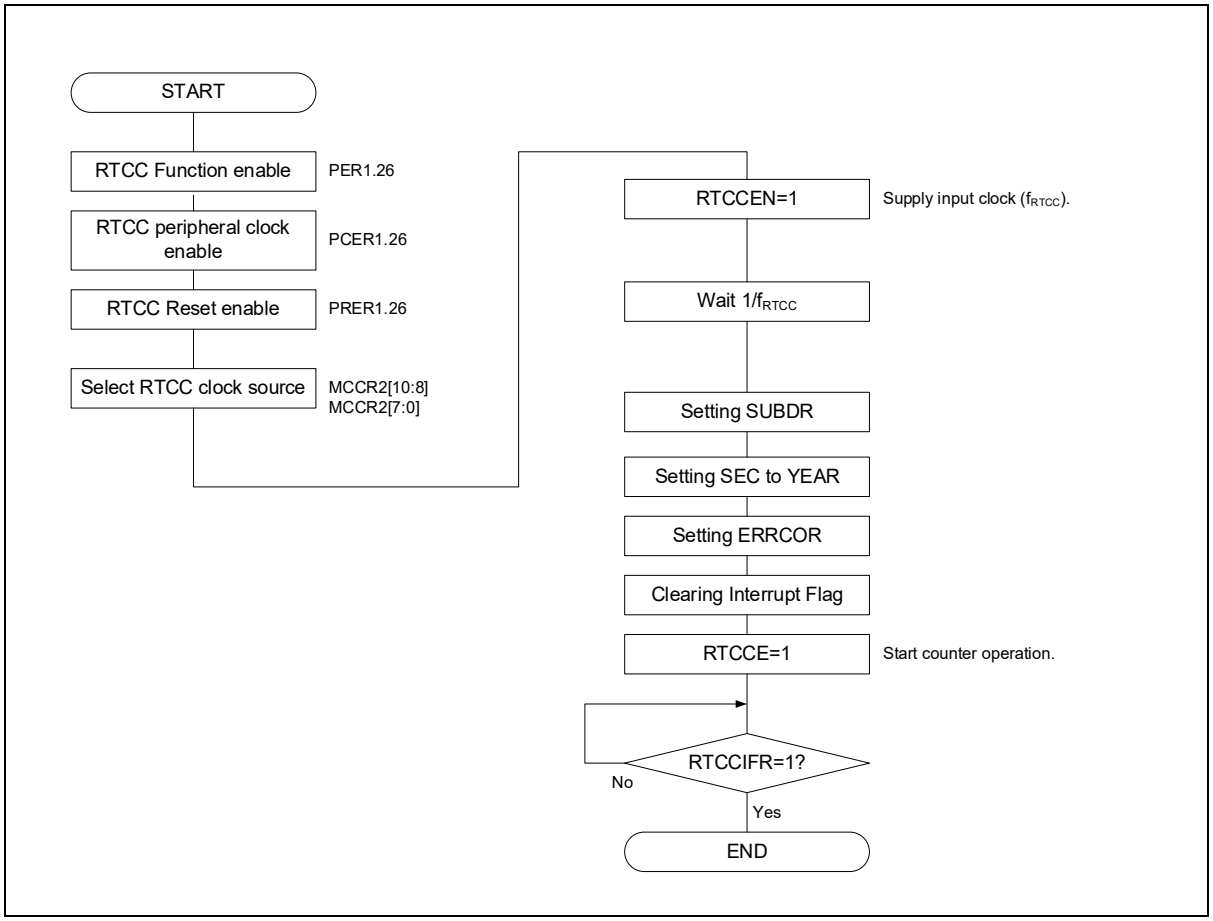
Pin Name	Type	Description	Supported Packages		
			A34L716 (100-LQFP)	A34L716 (64-LQFP)	A34L716 (56-QFN)
RTCC_OUT	O	RTCC outputs	O	O	O
RTCC_TAMP1	I	RTCC timestamp input 1	O	O	O
RTCC_TAMP2	I	RTCC timestamp input 2	O	O	X

24.3.3 Starting the RTCC Operation

To start the RTCC, a user should follow the procedure below:

1. Set the system configure register to enable RTCC peripherals (PER1.26, PCER1.26, PRER1.26).
2. Select the RTCC clock source by configuring MCCR2. (LSI or LSE)
3. Set the RTCCEN bit to '1', while the oscillation of the clock is stable.
4. Set the necessary control registers such as alarm, timestamp, and constant period matching etc. before activating the RTCC counter.
5. Set the AMPM register for the specified time system. If error correction is required, set the ERRCOR register.
6. After all the settings are completed, when the RTCC counter is read for run, activate RTCC by setting RTCCE to '1'.
7. Wait at least for the clock of RTCC after setting the counter when RTCCEN = 1 and RTCCE = 0.

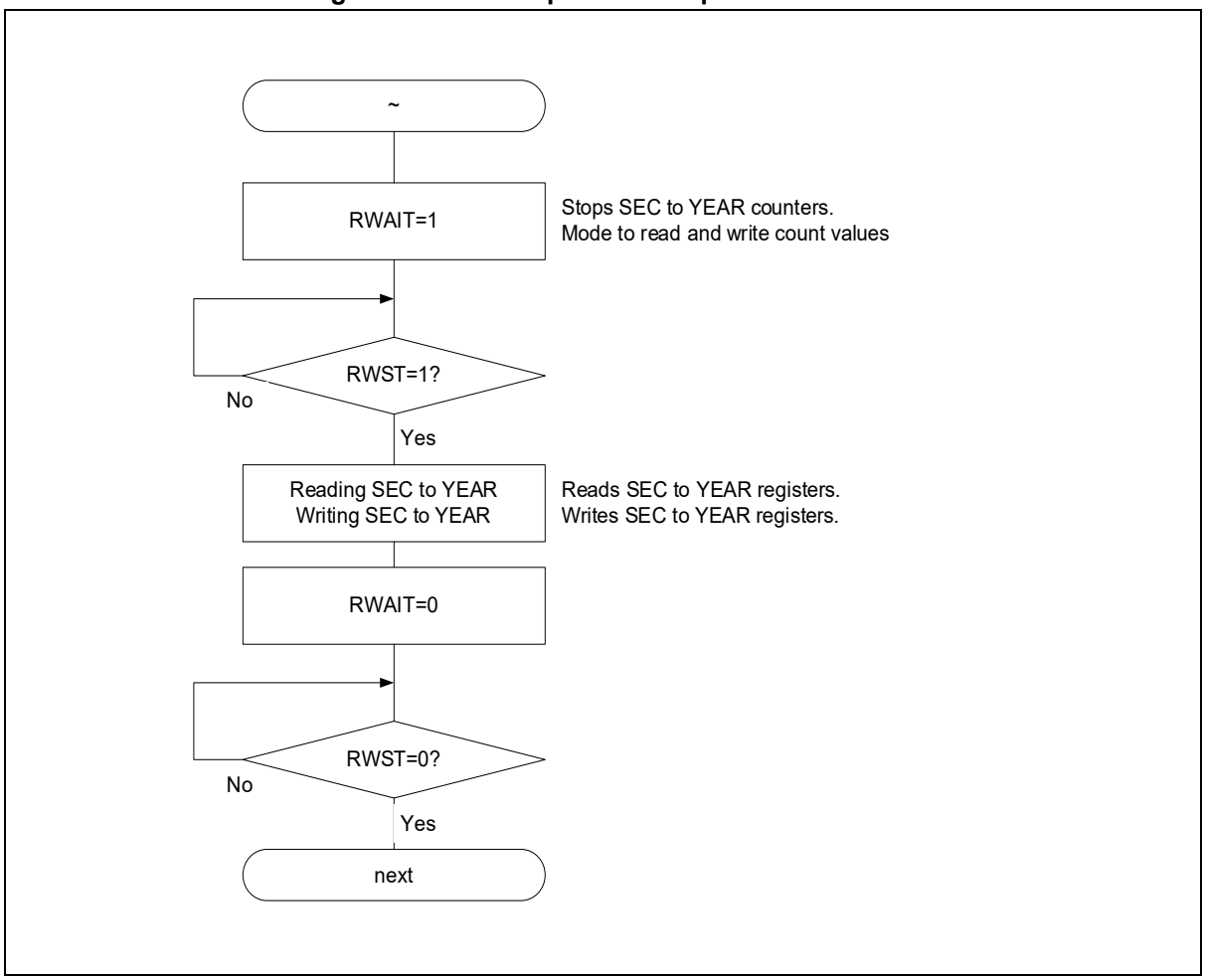
Figure 182. Start Operation Sequence of RTCC



24.3.4 RTCC RW Mode (Read and Write Mode)

RTCC counter read and write operations are not allowed in the RTCC run state. Only RTCC counters can be read and written in RW mode only. The RWAIT bit must be set to 1 to enter RW mode. RWST indicates that it is in RW mode and synchronized to the RTCC clock. To perform a read or write operation, be sure to check RWST = '1' before proceeding. To exit RW mode, set RWAIT to '0'.

Figure 183. Read Operation Sequence of RTCC



The value written in RW mode is loaded into the RTCC counter at the end of RW mode. In RW mode, the SEC to YEAR counters stop counting and only the sub-second counter continues to count.

Therefore, even if the sub-second counter match is repeated several times in RW mode, it will be applied only once at the end of RW mode. For this reason, RW mode operation should be done in 1 second. When writing to the SEC in RW mode, even if a sub-second counter overflow occurs, it is ignored.

NOTE:

1. Complete the series of processes of setting the RWAIT bit to '1' to clearing the RWAIT bit to '0' within '1' second.

The second counter register (SEC), minute counter register (MIN), hour counter register (HOUR), week counter register (WEEK), day counter register (DAY), month counter register (MONTH), and year counter register (YEAR) may be read/written in any sequence.

All registers do not have to be read and only some registers may be read or written.

When changing the values of the SEC, MIN, HOUR, WEEK, DAY, MONTH, and YEAR register while the counter operates (RTCCE = 1), rewrite the values of the MIN register after disabling interrupt servicing INTRTC by using the interrupt mask flag register. Furthermore, clear the WAFG, RIFG and RTCCIF flags after rewriting the MIN register.

Figure 184. Sub-second Counter Match in RW Mode

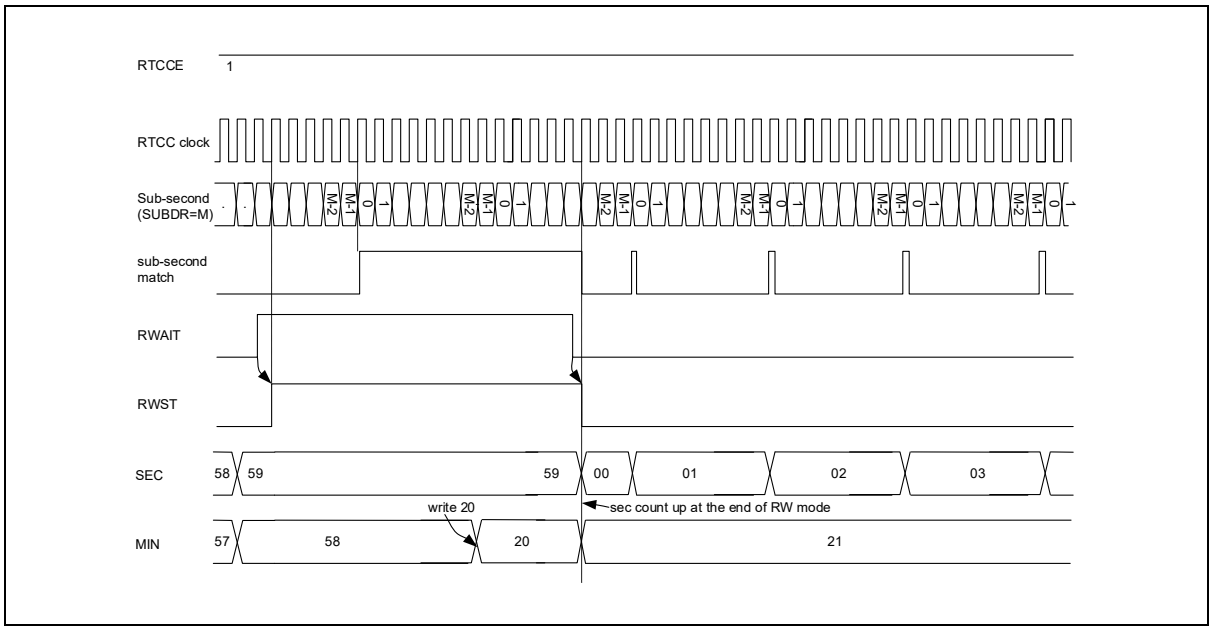
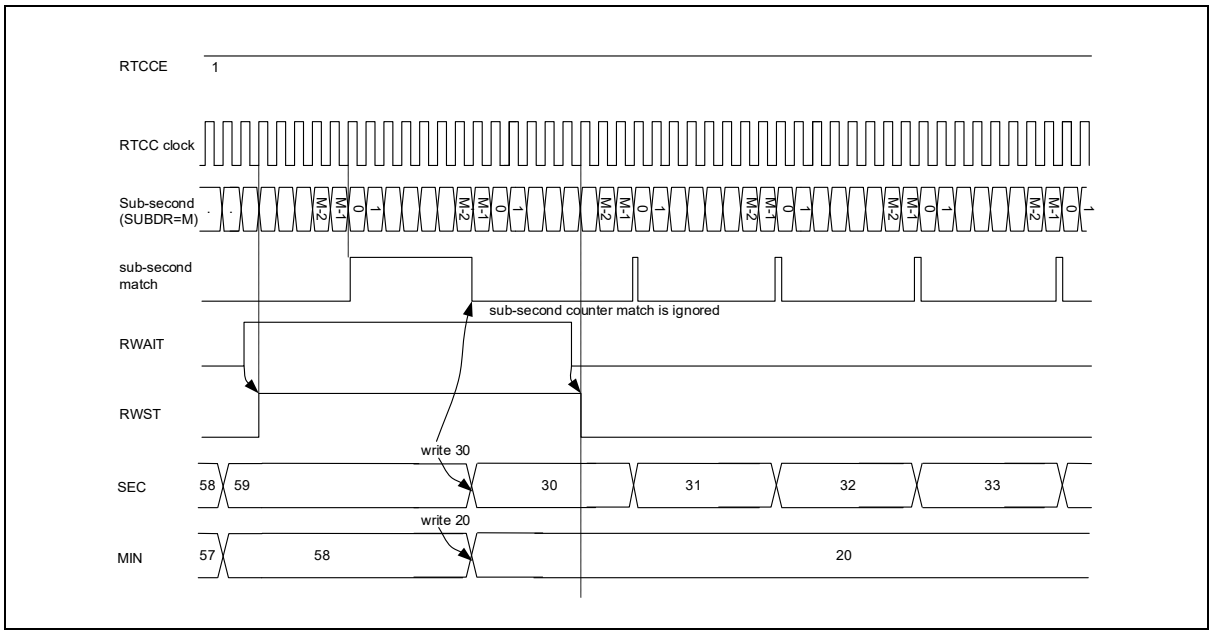


Figure 185. Sub-second Counter Match in RW Mode when Writing to SEC

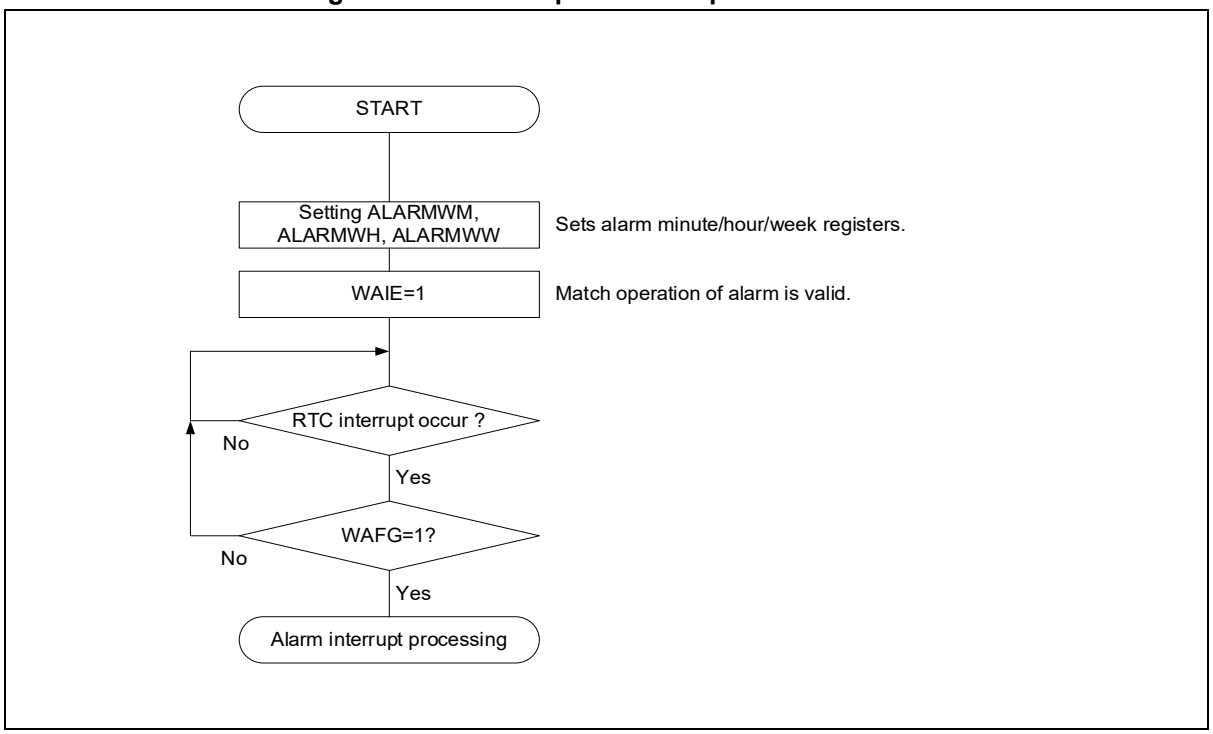


24.3.5 RTCC Alarm Setting

To activate the alarm function of RTCC, write '1' to WAIE of RTCC_CR. Alarm time can be set with three registers ALARMWM, ALARMWH and ALARMWW. The alarm week register (ALARMWW), alarm hour register (ALARMWH), and alarm minute register (ALARMWM) may be written in any sequence.

Check the WAFG of RTCCR to see that an alarm has occurred. Constant-period matching interrupts, alarm match interrupts and timestamp interrupts use the same interrupt source. When using these three types of interrupts at the same time, which interrupt occurred can be judged by checking the Constant-period matching flag (RIFG), the alarm detection status flag (WAFG), and the timestamp event flag (TSFG) upon RTCC interrupt occurrence.

Figure 186. Alarm Operation Sequence of RTCC



There is an ALARMWW bit corresponding to the Week counter register, and each bit of ALARMWW indicates the day of the week. Therefore, it is possible to set an alarm for a duplicate day of the week. (See Table 166 and Table 167.)

Table 166. WEEK and ALARMWW Registers

WEEK	ALARMWW						
	Bit6	Bit5	Bit4	Bit3	Bit2	Bit1	Bit0
0	0	0	0	0	0	0	1
1	0	0	0	0	0	1	0
2	0	0	0	0	1	0	0
3	0	0	0	1	0	0	0
4	0	0	1	0	0	0	0
5	0	1	0	0	0	0	0
6	1	0	0	0	0	0	0

Table 167. Example of Setting Alarm

Time of Alarm	Day (ALARMWW)							12-hour Display		24-hour Display	
	Sunday WW0	Monday WW1	Tuesday WW2	Wednesday WW3	Thursday WW4	Friday WW5	Saturday WW6	Hour ALARMWH	Minute ALARMWM	Hour ALARMWH	Minute ALARMWM
Every day, 0:00 a.m.	1	1	1	1	1	1	1	12	00	00	00
Every day, 1:30 a.m.	1	1	1	1	1	1	1	01	30	01	30
Every day, 11:59 a.m.	1	1	1	1	1	1	1	11	59	11	59
Monday through Friday, 0:00 p.m.	0	1	1	1	1	1	0	32	00	12	00
Sunday, 1:30 p.m.	1	0	0	0	0	0	0	21	30	13	30
Monday, Wednesday, 11:59 p.m.	0	1	0	1	0	0	0	31	59	23	59

24.3.6 RTCC Constant-Period Matching

The flag (RIFG) can be generated at regular intervals. Depending on the CT value of RTCC_CR, a flag is generated once every 0.5 second, 1 second, 1 minute, 1 hour, 1 day, 1month. However, this period is the value when the sub-second counter outputs 1 Hz. Constant-period matching does not occur when the RTCC value is zero. If the RTCC interrupt is enabled on the NVIC, the Constant-period matching flag is used as the interrupt source. In this case, periodic wake up in standby mode is possible. Since the RTCC interrupt and RTCC alarm and RTCC time stamp are also assigned together with the constant-period matching, if the RTCC interrupt occurs, the interrupt handler can check the FLAG (RIFG, WAFG, TSFG) to see which interrupt has occurred.

24.3.7 RTCC Timestamp

The timestamp function is provided when an external pin input occurs. The timestamp can be activated by writing ‘1’ to TSEN in the RTCC_CR register, and TS_EDGE_SEL determines the edge of the timestamp input.

When a timestamp input is received, the value of the RTCC counter register at that time is stored in RTCC_TIME_TS and RTCC_DATA_TS, and the RTCC continues counting.

However, since the external timestamp pin input (GPIO) and the internal RTCC counter (based on the LSE or LSI clock) operate asynchronously, incorrect time and date values may be captured due to timing issues.

To prevent this issue and obtain stable values, a debounce filter using the LSE or LSI clock as the source clock must be applied to the timestamp input pin. (SCU_MCCR5, PGBDCDIV[7:0] = 1)

It should be noted that the GPIO debounce configuration is shared on a port-group basis. Therefore, when debounce is set for the timestamp pins (PC7 and PC8), other GPIOC/D pins belonging to the same group are also affected by the debounce operation. This should be carefully considered during hardware pin assignment and firmware design.

24.3.8 RTCC 1 Hz Output

RTCC supports 1 Hz clock output to RTCC_OUT pin. The 1 Hz clock is the frequency when using LSE 32.768 kHz as RTCC clock and SUBDR is written 0x00007FFF. The RTCC 1 Hz output can be used for error correction of the RTCC counter.

24.3.9 Error Correction of RTCC

The watch can be corrected with high accuracy when it is slow or fast, by configuring a value to the error correction register (ERRCOR). The correction value is 1, 2, 3, 4, … 2^32-3, 2^32-2, 2^32-1 or -1, -2, -3, -4, … -2^32-3, -2^32-2, -2^32-1.

Target frequency is the results after performing the correction. The range of value that can be corrected by using the watch error correction register (ERRCOR) is shown below in Table 168.

Table 168. Correction Resolution by Frequency

Correction Data	f_{RTCC} = 32,768 Hz		f_{RTCC} = 40,000 Hz	
	DEV = 0	DEV = 1	DEV = 0	DEV = 1
Minimum resolution	±1.53 ppm	±0.51 ppm	±1.25 ppm	±0.41 ppm
1	±1.53 ppm	±0.51 ppm	±1.25 ppm	±0.41 ppm
2	±3.05 ppm	±1.02 ppm	±2.50 ppm	±0.83 ppm
3	±4.58 ppm	±1.52 ppm	±3.75 ppm	±1.25 ppm
122	±186.2 ppm	±62.1 ppm	±152.5 ppm	±50.8 ppm
123	±187.7 ppm	±62.6 ppm	±153.7 ppm	±51.2 ppm
124	±189.2 ppm	±63.0 ppm	±155.0 ppm	±51.7 ppm
1,024	±1,562.5 ppm	±520 ppm	±1,280.0 ppm	±427 ppm
32,768	±50,000 ppm	±16,667 ppm	±40,960 ppm	±13,653 ppm

24.3.9.1 Example of Calculating Correction Value

The following expression is used for calculation of the correction value which is used when correcting the count value of internal counter (32-bit):

- (DEV = 0): Correction value = Number of corrections counts in 1 minute ÷ 3 = (f_{RTCC} ÷ 32768 - 1) × (32768 x 60 ÷ 3)
- (DEV = 1): Correction value = Number of corrections counts in 1 minute = (f_{RTCC} ÷ 32768 - 1) × (32768 x 60)

24.3.9.2 Correction Example

Example procedure to correct from 32772.3 Hz to 32768 Hz (32772.3 Hz – 131.2 ppm)[1] is shown below:

1. Measuring the oscillation frequency: The f_{RTCC} is measured by outputting to RTCC_OUT pin when the ERRCOR register is set to its initial value (0x00000000).
2. Calculating the correction value: Assume the target frequency to be 32768 Hz (32772.3 Hz–131.2 ppm) and DEV to be zero.

   Correction value = (f_{RTCC} ÷ 32768 – 1) × 32768 × 60 ÷ 3 = (32772.3 ÷ 32768 – 1) × 32768 × 60 ÷ 3 = 86
3. Calculating the values to be set to (ERRCOR): If the correction value is 0 or larger (when slowing), assume DIR to be '1'.

   Consequently, when correcting from 32772.3 Hz to 32768 Hz (32772.3 Hz – 131.2 ppm), setting the error correction register such that DEV is 0 and the correction value is 86 results in 32768 Hz (0 ppm)

   ERRCOR = 0x5000_0056

NOTE:

1. If the CORDATA value exceeds the clock frequency range, it may malfunction.

24.4 RTCC Registers

The base address and register map of the RTCC module are described in the following tables.

Table 169. Base Address of RTCC

Name	Base Address
RTCC	0x4000_0900

Table 170. CRC Register Map

Name	Offset	Type	Description	Reset Value	Reference
RTCC_CR	0x0000	RW	RTCC Control Register	0x0000_0000	24.4.1
RTCC_ERRCOR	0x0008	RW	Watch Error Correction Register	0x0000_0000	24.4.2
RTCC_SEC	0x000C	RW	Second Counter Register	0x0000_0000	24.4.3
RTCC_MIN	0x0010	RW	Minute Counter Register	0x0000_0000	24.4.4
RTCC_HOUR	0x0014	RW	Hour Counter Register	0x0000_0000	24.4.5
RTCC_DAY	0x0018	RW	Day Counter Register	0x0000_0000	24.4.6
RTCC_WEEK	0x001C	RW	Week Counter Register	0x0000_0000	24.4.7
RTCC_MONTH	0x0020	RW	Month Counter Register	0x0000_0000	24.4.8
RTCC_YEAR	0x0024	RW	Year Counter Register	0x0000_0000	24.4.9
RTCC_ALARMW	0x0028	RW	Alarm Minute Register	0x0000_1200	24.4.10
RTCC_SUBDR	0x002C	RW	Sub-second counter compare register	0x0001_FFFF	24.4.11
RTCC_TIME_TS	0x0030	RW	Timestamp time register	0x0000_0000	24.4.12
RTCC_DATE_TS	0x0034	RW	Timestamp date register	0x0000_0000	24.4.13

24.4.1 RTCC_CR: RTCC Control Register

The RTCC_CR register is used to control the RTCC.

RTCC_CR=0x4000_0900

31	30	29 28 27 26 25 24	23 22 21	20	19 18 17	16	15	14 13	12	11	10 9	8	7 6 5 4 3 2	1	0
RTCCEN	RTCCE	Reserved		AMPM	CT[2:0]	RIFG	TSEN	TS_EDGE_SEL[1:0]	TSFG	WAIE	Reserved	WAFG	Reserved	RWST	RWAIT
0	0	-		0	0	0	0	0	0	0	-	0	-	0	0
RW	RW	-		RW	RW	RW	RW	RW	RW	RW	-	RW	-	RO	RW

31	RTCCEN[1]	RTCC clock input enable
		0 RTCC clock input disabled
		1 RTCC clock input enabled
30	RTCCE	RTCC counter run
		0 RTCC counter stop
		1 RTCC counter run
20	AMPM[2]	Hour system selection
		0 12-hour system (a.m. and p.m. are displayed.)
		1 24-hour system
19 17	CT[2:0]	Period selection of constant-period matching
		000 Constant-period matching disabled
		001 Once per 0.5s (synchronized with second count up)
		010 Once per 1s (same time as second count up)
		011 Once per 1m (second 0 of every minute)
		100 Once per 1 hour (minute 0 and second 0 of every hour)
		101 Once per 1 day (hour 0, minute 0, and second 0 of every day)
		110 Once per 1 month (Day 1, hour 0 a.m., minute 0 and second 0 of every month)
		111 reserved
16	RIFG[3]	Constant-period matching flag
		0 Constant-period matching is not generated
		1 Constant-period matching is generated
15	TSEN[4]	Timestamp enable
		0 Timestamp is disabled
		1 Timestamp is enabled
14 13	TS_EDGE_SEL [1:0]	Timestamp event active edge selection
		00 Falling edge
		01 Rising edge
		10 Falling/rising
		11 reserved
12	TSFG[5]	Timestamp event flag
		0 Timestamp event is not generated
		1 Timestamp event is generated
11	WAIE	Alarm enable
		0 Alarm is disabled
		1 Alarm is enabled
8	WAFG[5]	Alarm flag

		0	Alarm is not generated
		1	Alarm is generated
1	RWST[6]	RW mode status	
		0	Run mode
		1	RW mode
0	RWAIT[7]	RW mode set	
		0	RW mode is unset
		1	RW mode is set

NOTES:

1. If RTCCEN = 0, writing to RTCC registers is ignored. If the registers are read, only default values are read. (SEC to YEAR, ALARMWx, ERRCOR). AMPM can only be changed in RW mode. HOUR value for the 12-hour system and the 24-hour system are shown in Table 171.
2. This flag is reset when '0' is written to it. Writing '1' to it is invalid.
3. If timestamp wakeup is used in standby mode, PC7, PC8 must be set to timestamp function in run mode.
4. This flag is reset when '0' is written to it. Writing '1' to it is invalid.
5. RWST is a signal that synchronizes RWAIT to RTCC clock(f_{RTCC}). This indicates that it is in RW mode for reading and writing the RTCC counter. Before reading and writing the RTCC counter, confirm that the value of this flag is '1'.
6. Constant-period matching interrupts, alarm match interrupts and timestamp interrupts use the same interrupt source.

   When using these three types of interrupts at the same time, which interrupt occurred can be judged by checking the Constant-period matching flag (RIFG), the alarm detection status flag (WAFG), and the timestamp event flag (TSFG) upon RTCC interrupt occurrence.

   25us delay required after RTCCEN enable.
   Example code)
   RTCCCR =| 0x01 (RTCCEN enable)
   NOP; … NOP; // 25 us

24.4.2 RTCC_ERRCOR: RTCC Error Correction Register

ERRCOR is a 32-bit register. This register is used to correct the watch with high accuracy when it is slow or fast by changing the value that overflows from the sub-second counter register to the second counter register (SEC).

RTCC_ERRCOR=0x4000_0908

31	30	29	28	27 26 25 24	23 22 21 20 19 18 17 16	15 14 13 12 11 10 9 8	7 6 5 4 3 2 1 0
DEV	DIR	Reserved	COREN	Reserved		CORDATA [14:0]	
0	0	-	0	-		0x0000	
RW	RW	-	RW	-		RW	

31	DEV	Whether to enable or disable CRC output data inversion
	0	Corrects watch error when the second digits are at 00, 20, or 40 (every 20 seconds).
	1	Corrects watch error only when the second digits are at 00 (every 60 seconds)
30	DIR	Setting of watch error correction direction
	0	Decreases by CORDATA
	1	Increases by CORDATA
28	COREN	Enable watch error correction
	0	Disable
	1	Enable
14 0	CORDATA[14:0]	Setting of watch error correction data When CORDATA=0, the watch error is not corrected

24.4.3 RTCC_SEC: RTCC Second Counter Register

SEC is a 7-bit register that takes a value of '00' to '59' (decimal) and indicates the count value of seconds. It counts up when the sub-second counter register overflows. When data is written to this register, it is written to a buffer and then to the counter when RWST bit is changed to '0'.

Set a decimal value of '00' to '59' to this register in BCD code. If a value that is not a BCD code is written, it is ignored. If a value outside the range is written, the reset value is written.

RTCC_SEC=0x4000_090C

31 30 29 28 27 26 25 24	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			SEC[6:0]
-			0 0 0 0 0 0 0
.			RW

6 0	SEC[6:0]	Second Counter Register

24.4.4 RTCC_MIN: RTCC Minute Counter Register

MIN is a 7-bit register that takes a value of '00' to '59' (decimal) and indicates the count value of minutes. It counts up when the second counter overflows. When data is written to this register, it is written to a buffer and then to the counter when RWST bit is changed to '0'. Even if the second counter register overflows while this register is being written, the overflow is ignored, and MIN register is set to the value written.

Set a decimal value of '00' to '59' to this register in BCD code. If a value that is not a BCD code is written, it is ignored. If a value outside the range is written, the reset value is written.

RTCC_MIN=0x4000_0910

31 30 29 28 27 26 25 24	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			MIN[6:0]
-			0 0 0 0 0 0 0
.			RW

6 0	MIN[6:0]	Minute Counter Register

24.4.5 RTCC_HOUR: RTCC Hour Counter Register

HOUR is a 6-bit register that takes a value of ‘00’ to ‘23’ (‘01’ to ‘12’ and ‘21’ to ‘32’ when AMPM=0, decimal) and indicates the count value of hours. It counts up when the minute counter overflows. When data is written to this register, it is written to a buffer and then to the counter when RWST bit is changed to ‘0’. Even if the minute counter register overflows while this register is being written, the overflow is ignored, and HOUR register is set to the value written.

Specify a decimal value of ‘00’ to ‘23’ (‘01’ to ‘12’ and ‘21’ to ‘32’ when AMPM=0) by using BCD code according to the time system specified using bit 20 (AMPM) of real-time clock control register (RTCCCR).

If the AMPM bit value is changed, the values of the HOUR register and ALARMWH register change according to the specified time system. If a value that is not a BCD code is written, it is ignored. If a value outside the range is written, the reset value is written. Reset signal clears this register to 12H.

RTCC_HOUR=0x4000_0914

31 30 29 28 27 26 25 24	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				HOUR[5:0]					
-				0	0	0	0	0	0
-				RW					

5 0	HOUR[5:0]	Hour Counter Register

Table 171. Specified Time System by AMPM

24-hour Display (AMPM=1)				12-hour Display (AMPM=0)			
Time	**Hour**	**Time**	**Hour**	**Time**	**Hour**	**Time**	**Hour**
12 a.m.	00H	12 p.m.	12H	12 a.m.	12H	12 p.m.	32H
1 a.m.	01H	1 p.m.	13H	1 a.m.	01H	1 p.m.	21H
2 a.m.	02H	2 p.m.	14H	2 a.m.	02H	2 p.m.	22H
3 a.m.	03H	3 p.m.	15H	3 a.m.	03H	3 p.m.	23H
4 a.m.	04H	4 p.m.	16H	4 a.m.	04H	4 p.m.	24H
5 a.m.	05H	5 p.m.	17H	5 a.m.	05H	5 p.m.	25H
6 a.m.	06H	6 p.m.	18H	6 a.m.	06H	6 p.m.	26H
7 a.m.	07H	7 p.m.	19H	7 a.m.	07H	7 p.m.	27H
8 a.m.	08H	8 p.m.	20H	8 a.m.	08H	8 p.m.	28H
9 a.m.	09H	9 p.m.	21H	9 a.m.	09H	9 p.m.	29H
10 a.m.	10H	10 p.m.	22H	10 a.m.	10H	10 p.m.	30H
11 a.m.	11H	11 p.m.	23H	11 a.m.	11H	11 p.m.	31H

24.4.6 RTCC_DAY: RTCC Day Counter Register

DAY is a 6-bit register that takes a value ranging from '01' to '31' (decimal) and indicates the count value of days. It counts up when the hour counter overflows.

This counter counts the following:

- '01' to '31' (January, March, May, July, August, October, December)
- '01' to '30' (April, June, September, November)
- '01' to '29' (February, leap year)
- '01' to '28' (February, normal year)

When data is written to this register, it is written to a buffer and then to the counter when RWST bit is changed to '0'. Even if the hour counter register overflows while this register is being written, this overflow is ignored, and DAY register is set to the value written. Set a decimal value of '01' to '31' (corresponding to the month) to this register in BCD code. If a value that is not a BCD code is written, it is ignored. If a value outside the range is written, the reset value is written.

RTCC_DAY=0x4000_0918

31 30 29 28 27 26 25 24	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				DAY[5:0]
-				0 0 0 0 0 0
.				RW

5 0	DAY[5:0]	Day Counter Register

24.4.7 RTCC_WEEK: RTCC Week Counter Register

WEEK is a 3-bit register that takes a value ranging from ‘0’ to ‘6’ (decimal) and indicates the count value of weekdays.

It counts up in synchronization with the day counter. When data is written to this register, it is written to a buffer and then to the counter when RWST bit is changed to ‘0’. Set a decimal value of 0 to 6 to this register. If a value outside the range is written, the reset value is written.

There is no corresponding WEEK value according to DAY. Select a value from ‘0’ to ‘6’ that corresponds to the day of the week and use it sequentially. After reset release, set the week counter register to the desired value.

RTCC_WEEK=0x4000_091C

31 30 29 28 27 26 25 24	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				WEEK[2:0]		
-				0	0	0
				RW		

2 0	WEEK[2:0]	Week Counter Register

24.4.8 RTCC_MONTH: RTCC Month Counter Register

MONTH is a 5-bit register that takes a value ranging from ‘01’ to ‘12’ (decimal) and indicates the count value of months. It counts up when the day counter overflows. When data is written to this register, it is written to a buffer and then to the counter when RWST bit is changed to ‘0’. Even if the day counter register overflows while this register is being written, the overflow is ignored, and MONTH register is set to the value written.

Set a decimal value of ‘01’ to ‘12’ to this register in BCD code. If a value that is not a BCD code is written, it is ignored. If a value outside the range is written, the reset value is written 0x0.

RTCC_MONTH=0x4000_0920

31 30 29 28 27 26 25 24	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				MONTH [4:0]				
-				0	0	0	0	1
.				RW				

4 0	MONTH[4:0]	Month Counter Register

24.4.9 RTCC_YEAR: RTCC Year Counter

YEAR is an 8-bit register that takes a value ranging from '00' to '99' (decimal) and indicates the count value for years. It counts up when the month counter register overflows. Values 00, 04, 08, …, 92, and 96 indicate a leap year.

When data is written to this register, it is written to a buffer and then to the counter when RWST bit is changed to '0'. Even if the MONTH register overflows while this register is being written, the overflow is ignored, and YEAR register is set to the value written.

Set a decimal value of '00' to '99' to this register in BCD code. If a value that is not a BCD code is written, it is ignored.

RTCC_YEAR=0x4000_0924

31 30 29 28 27 26 25 24	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			YEAR[7:0]
-			0 0 0 0 0 0 0 0
-			RW

7 0	YEAR[7:0]	Year Counter Register

24.4.10 RTCC_ALARMW: RTCC Alarm Register

ALARMW sets alarm time by configuring the minute, hour and week of alarm.

RTCC_ALARMW=0x4000_0928

31 30 29 28 27 26 25 24	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	ALARMWW[6:0]	Reserved	ALARMWH[5:0]	Reserved	ALARMWM[6:0]
-	0 0 0 0 0 0 0	-	0 1 0 0 1 0	-	0 0 0 0 0 0 0
-	RW	-	RW	-	RW

22 16	ALARMWW[6:0]	This register is used to set the day of the week for the alarm. Each of the seven bits represents a day of the week, so that duplicate days of the week can be selected.
13 8	ALARMWH[5:0]	This register is used to set hours of the alarm. Set a decimal value of 00 to 23, 01 to 12, or 21 to 32 to this register according to AMPM in BCD code. If a value outside the range is set, the alarm is not detected.
6 0	ALARMWM[6:0]	This register is used to set minutes of the alarm. Set a decimal value of 00 to 59 to this register in BCD code. If a value outside the range is set, the alarm is not detected.

24.4.11 RTCC_SUBDR: RTCC Sub-Second Counter Compare Register

SUBDR is a 32-bit register. This register determines the sub-second counter matching value and provides the desired frequency clock to the second counter register.

RTCC_SUBDR=0x4000_092C

31 30 29 28 27 26 25 24	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		SUBDR[15:0]	
-		0xFFFF	
-		RW	

15 0	SUBDR[15:0]	Sub-second counter Compare Register.

24.4.12 RTCC_TIME_TS: RTCC TimeStamp Time Register

TIME_TS automatically saves the current time when a timestamp event occurs.

RTCC_TIME_TS=0x4000_0930

31 30 29 28 27 26 25 24	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	HOUR_TS[7:0]	MIM_TS[7:0]	SEC_TS[7:0]
-	0x00	0x00	0x00
-	RO	RO	RO

23 16	HOUR_TS[7:0]	HOUR stored when the timestamp event occurred.
15 8	MIN_TS[7:0]	MIN stored when the timestamp event occurred
7 0	SEC_TS[7:0]	SEC stored when the timestamp event occurred

24.4.13 RTCC_DATE_TS: RTCC TimeStamp Date Register

DATE_TS automatically saves the current date when a timestamp event occurs.

RTCC_DATE_TS=0x4000_0934

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

31 24	YEAR_TS[7:0]	YEAR stored when the timestamp event occurred.
23 16	MONTH_TS[7:0]	MONTH stored when the timestamp event occurred.
15 8	WEEK_TS[7:0]	WEEK stored when the timestamp event occurred
7 0	DAY_TS[7:0]	DAY stored when the timestamp event occurred

24.4.14 RTCC Register Map Summary

Table 172. RTCC 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	RTC_CR	RTCEN	RTCE	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	AMPM	CT[2:0]			RIFG	TSEN	TS_EDGE_SEL[1:0]		TSFG	WAIE	Res.	Res.	WAFG	Res.	Res.	Res.	Res.	Res.	Res.	RWST	RWAIT
	Reset value	0	0										0	0	0	0	0	0	0	0	0	0			0							0	0
0x08	RTC_ERRCOR	DEV	DIR	Res.	COREN	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	CORDATA [14:0]															
	Reset value	0	0		0													0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
0x0C	RTC_SEC	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	SEC[6:0]							
	Reset value																									0	0	0	0	0	0	0	0
0x10	RTC_MIN	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	MIN[6:0]							
	Reset value																									0	0	0	0	0	0	0	0
0x14	RTC_HOUR	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	HOUR[6:0]							
	Reset value																									0	0	0	0	0	0	0	0
0x18	RTC_DAY	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	DAY[6:0]							
	Reset value																									0	0	0	0	0	0	0	0
0x1C	RTC_WEEK	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.	WEEK[2:0]		
	Reset value																														0	0	0
0x20	RTC_MONTH	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.	MONTH[4:0]				
	Reset value																												0	0	0	0	1
0x24	RTC_YEAR	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	YEAR[7:0]							
	Reset value																									0	0	0	0	0	0	0	0

Table 172. RTCC 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	RTC_ALARMW	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	ALARMWW[6:0]							Res.	Res.	ALARMWH[5:0]						Res.	ALARMWM[6:0]						
	Reset value										0	0	0	0	0	0	0			0	1	0	0	1	0		0	0	0	0	0	0	0
0x2C	RTC_SUBDR	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	SUBDR[15:0]															
	Reset value																	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1
0x30	RTC_TIME_TS	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	HOUR_TS[7:0]								MIM_TS[7:0]								SEC_TS[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
0x34	RTC_DATE_TS	YEAR_TS[7:0]								MONTH_TS[7:0]								WEEK_TS[7:0]								DAY_TS[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

Abbreviation for Registers

The following abbreviations are used in register descriptions:

Table 173. Abbreviation

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
- ADC: Analog to Digital Converter
- AHB: Advanced High-performance Bus
- APB: Advanced Peripheral Bus
- CRC: Cyclic Redundancy Check
- DAC: Digital to Analog Converter
- DMA: Direct Memory Access
- FRT: Free-Run Timer
- HSE: High-Speed External
- HSI: High-Speed Internal
- HSICMU: High-Speed Internal OSC Clock Management Unit
- I2C: Inter-Integrated Circuit
- LCD: Liquid Crystal Display
- LSB: Least Significant Bit
- LPUART: Low-Power Universal Asynchronous Receiver / Transmitter
- LQFP: Low-profile Quad Flat Package
- LSE: Low-Speed External
- LSI: Low-Speed Internal
- LVI: Low-Voltage Indicator
- LVR: Low-Voltage Reset
- MSB: Most Significant Bit
- OPAMP: Operational Amplifier
- PLL: Phase Locked Loop
- POR: Power On Reset
- QFN: Quad Flat Non-Leaded Package
- RTCC: Real Time Clock
- VREFBUF: Voltage Reference Buffer
- SCU: System Control Unit

- SPI: Serial Peripheral Interface
- UART: Universal Asynchronous Receiver Transmitter
- USART: Universal Synchronous / Asynchronous Receiver Transmitter
- USB: Universal Serial Bus
- WDT: Watchdog Timer

Revision History

Revision	Date	Notes
1.00	Sep. 20, 2024	Initial release
1.01	Dec. 02, 2024	Updated the disclaimer
1.02	Mar. 28, 2025	• Updated Figure 11, Section 4.4.5.1, Section 4.8.30, Section 10.3.3.2, and Section 15.4.2, and Figure 166 • Added Section 7.3.5, Table 58, Section 9.4.3, and Section 10.4.1
1.03	Nov. 20, 2025	• Added a note about unsupported internal pull-up and port strength on PB5-PB10 • Added a note about DMATX flag not occurring during 1-byte transmission in USART DMA TX mode
1.04	Mar. 23, 2026	• Updated the HSICMU Block diagram • Added precautions for RTC Timestamp and HSICMU usage

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
Tel: +81-6-6225-7758

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.

© 2024 - 2026 ABOV Semiconductor – All rights reserved