Cortex®-M0+ 32-bit Microcontroller, 32 MHz, Up to 128 KB Code Flash, Up to 8 KB SRAM, 12-bit ADC

UM Rev. 1.03

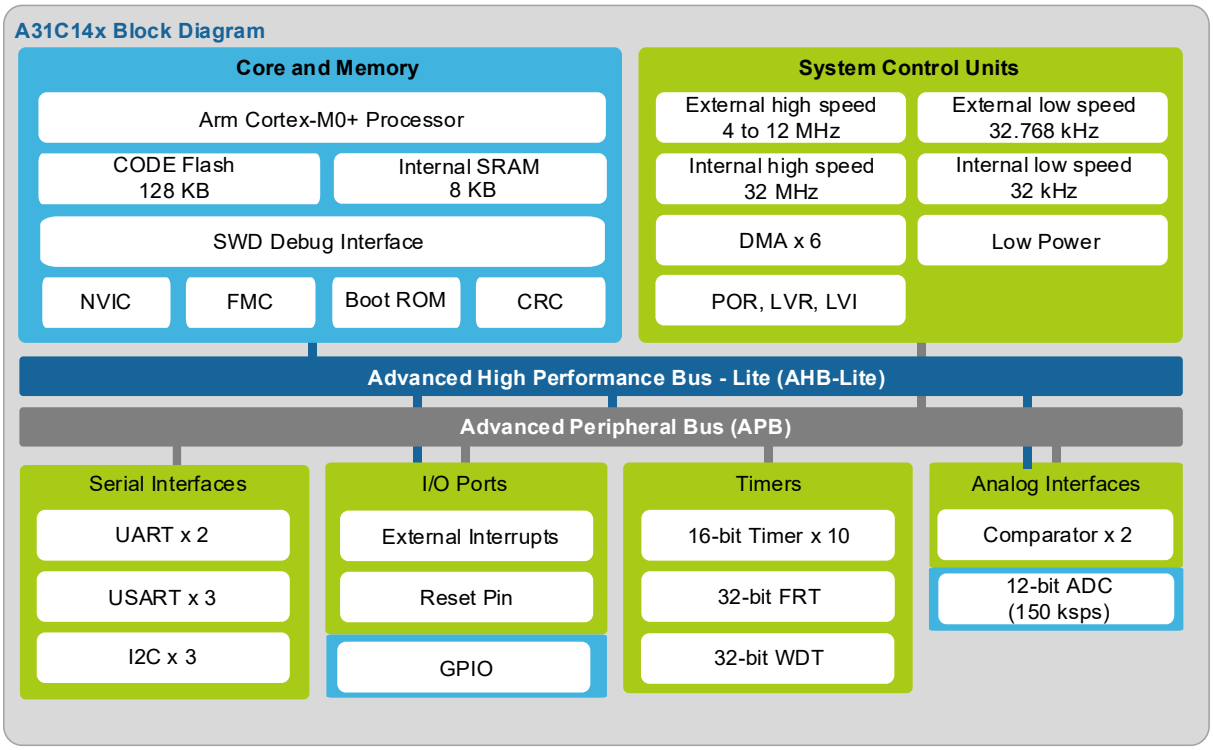
Introduction

This document describes the modules and peripherals of the A31C14x, A31C12x series of devices. Each description presents the module or peripheral in a general sense. Not all features and functions of all modules or peripherals are present on all devices. In addition, modules or peripherals can differ in their exact implementation on different devices. Pin functions, internal signal connections, and operational parameters differ from device to device.

The A31C14x and A31C12x are a 32-bit general purpose microcontroller for various appliances. The device supports both general-purpose and low-cost applications through simple, easy-to-use.

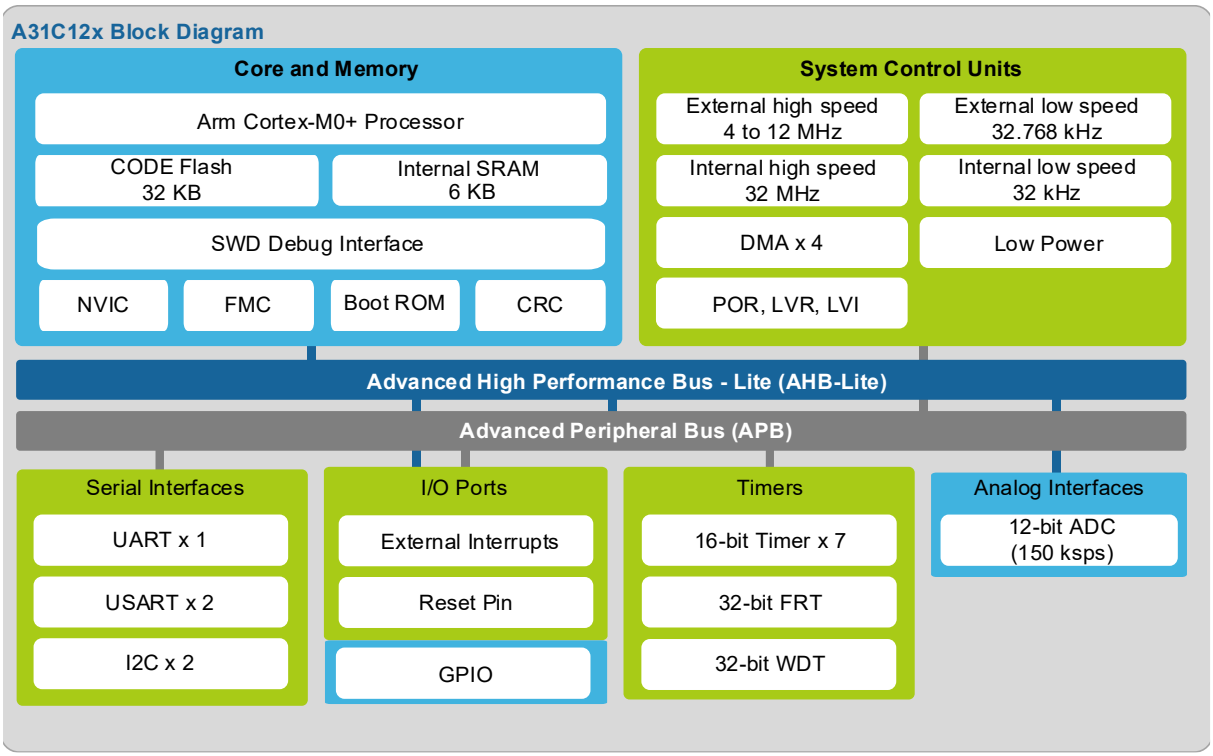
The A31C14x series has two comparators and 12-bit ADC with 18 analog channels. Multiple interfaces such as UART, USART, and I2C make external communications possible.

Figure 1. A31C14x Block Diagram



The A31C12x series has 12-bit ADC with 13 analog channels. Multiple interfaces such as UART, USART, and I2C make external communications possible.

Figure 2. A31C12x Block Diagram



Reference Documents

- DDI 0484C, published by Arm, contains information about the Cortex-M0+.
- The A31C14x and A31C12x Datasheets, published by ABOV, are available at www.abovsemi.com.

Contents

Introduction 1
Reference Documents 2
1. Description 22
1.1 Product Category Definition 22
1.2 Device Overview 23
1.3 Summary of A31C14x, A31C12x Peripherals 30
2. Pinouts and Pin Descriptions 31
2.1 Pinouts 31
2.1.1 A31C14x 31
2.1.1.1 A31C14xRL (64-LQFP 10X10) 31
2.1.1.2 A31C144RN (64-LQFP 14X14) 32
2.1.1.3 A31C14xCL (48-LQFP) 33
2.1.1.4 A31C14xCY (48-QFN) 34
2.1.1.5 A31C14xSN (44-LQFP) 35
2.1.1.6 A31C14xKY (32-QFN) 36
2.1.1.7 A31C14xGR (28-TSSOP) 37
2.1.1.8 A31C143FU (20-QFN) 37
2.1.1.9 A31C143FR (20-TSSOP) 38
2.1.2 A31C12x 39
2.1.2.1 A31C12xKY (32-QFN) 39
2.1.2.2 A31C12xKN (32-LQFP) 40
2.1.2.3 A31C12xGR (28-TSSOP) 41
2.1.2.4 A31C12xLU (24-QFN) 42
2.1.2.5 A31C12xFU (20-QFN) 43
2.1.2.6 A31C12xFR (20-TSSOP) 43
2.2 Pin Description 44
2.2.1 A31C14x 44
2.2.2 A31C12x 49
3. System and Memory Overview 52
3.1 System Architecture 52
3.1.1 System Block Diagram 52
3.1.1.1 A31C14x 52
3.1.1.2 A31C12x 53
3.1.2 Cortex-M0+ Core 54
3.1.3 Interrupt Controller 55
3.1.3.1 A31C14x 55
3.1.3.2 A31C12x 57
3.2 Memory Organization 59
3.2.1 Introduction 59
3.2.2 Memory Map and Register Boundary Address 60
3.2.2.1 A31C14x 60
3.2.2.2 A31C12x 61
3.3 Memory Map 62
3.3.1 Embedded SRAM 62
3.3.2 Flash Memory Overview 62
3.4 Boot Configuration 63
4. SCU 64

4.1 SCU Introduction 64
4.2 SCU Main Features 64
4.3 SCU Functional Description 65
4.3.1 SCU Block Diagram 65
4.3.2 SCU Pin and Type 65
4.4 Clocks 66
4.4.1 Overview 66
4.4.2 Clock Tree Configuration 67
4.4.3 HSE Clock 68
4.4.4 HSI Clock 69
4.4.5 LSE Clock 69
4.4.6 LSI Clock 70
4.4.7 Two Main Operating Clocks: HCLK and PCLK 70
4.4.7.1 HCLK Clock 70
4.4.7.2 PCLK Clock 70
4.4.8 Configuration of Miscellaneous Clocks 71
4.4.9 Peripheral Clocks 72
4.4.10 Clock Monitoring 73
4.4.10.1 Detection Range 73
4.4.10.2 Clock Monitoring Functions and Clock Sources 74
4.4.10.3 MCLK Monitoring 75
4.4.10.4 HSE Monitoring 77
4.4.10.5 LSE Monitoring 79
4.4.11 Clock Configuration Procedure 81
4.5 Reset 84
4.5.1 Overview 84
4.5.2 Reset Tree Configuration 85
4.5.3 Cold Reset 86
4.5.4 Warm Reset 87
4.5.5 LVD Reset (LVR) 88
4.5.6 nRESET Pin Reset 88
4.5.7 WDT Reset 89
4.5.8 Software Reset 89
4.5.9 HSE Fail Reset 89
4.5.10 LSE Fail Reset 89
4.5.11 CPU Request Reset 90
4.6 Power Supply 91
4.6.1 LVI Block Diagram 93
4.6.2 LVR Block Diagram 94
4.6.3 POR, LVR 94
4.7 Operation Modes 95
4.7.1 Overview 95
4.7.2 Transition of Operation Mode 96
4.7.3 RUN Mode 97
4.7.4 SLEEP Mode 97
4.7.5 DEEP-SLEEP (STOP1, STOP2) Mode 98
4.7.6 Operation Mode Summary 100
4.8 CHIPCONFIG and SCU Registers 101
4.8.1 CHIPCONFIG_VENDORID: Vendor ID Register 103
4.8.2 CHIPCONFIG_CHIPID: Chip ID Register 103

4.8.3 CHIPCONFIG_REVNR: Revision Number Register 104
4.8.4 SCU_SMR: System Mode Register 105
4.8.5 SCU_SRCR: System Reset Control Register 106
4.8.6 SCU_WUER: Wake-Up Source Enable Register 107
4.8.7 SCU_WUSR: Wake-Up Source Status Register 108
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 1 115
4.8.13 SCU_PER2: Peripheral Enable Register 2 117
4.8.14 SCU_PCER1: Peripheral Clock Enable Register 1 118
4.8.15 SCU_PCER2: Peripheral Clock Enable Register 2 120
4.8.16 SCU_CSCR: Clock Source Control Register 121
4.8.17 SCU_SCCR: System Clock Control Register 122
4.8.18 SCU_CMR: Clock Monitoring Register 123
4.8.19 SCU_COR: Clock Output Register 125
4.8.20 SCU_NMICR: NMI Control Register 126
4.8.21 SCU_NMISR: NMI Status Register 127
4.8.22 SCU_VDCCON: VDC Control Register 128
4.8.23 SCU_LVICR: LVI (Low-Voltage Indicator) Control Register 129
4.8.24 SCU_LVISR: LVI (Low-Voltage Indicator) Status Register 130
4.8.25 SCU_LVRCR: LVR (Low-Voltage Detect Reset) Control Register 131
4.8.26 SCU_EOSCR: External Oscillator Control Register 132
4.8.27 SCU_MCCR1: Miscellaneous Clock Control Register 1 133
4.8.28 SCU_MCCR3: Miscellaneous Clock Control Register 3 134
4.8.29 SCU_MCCR4: Miscellaneous Clock Control Register 4 135
4.8.30 SCU_MCCR5: Miscellaneous Clock Control Register 5 136
4.8.31 SCU_MCCR6: Miscellaneous Clock Control Register 6 137
4.8.32 SCU_MCCR7: Miscellaneous Clock Control Register 7 138
4.8.33 SCU_SYSTEN: System Access Enable Register 139
4.8.34 SCU Register Map Summary 140
5. PCU and GPIO 144
5.1 Introduction 144
5.1.1 PCU Introduction 144
5.1.2 GPIO Introduction 144
5.2 Main Features 145
5.2.1 PCU Features 145
5.2.2 GPIO Features 145
5.3 GPIO Functional Description 146
5.3.1 PCU and GPIO Pins and Internal Signals 150
5.3.2 I/O Port Control Registers 150
5.3.3 I/O Port Data Registers 151
5.3.4 I/O Data Bitwise Handling 151
5.3.5 I/O Alternate Function 152
5.3.5.1 A31C14x 153
5.3.5.2 A31C12x 155
5.3.6 Interrupt Functionality 156
5.3.7 Output Configuration 157
5.3.8 Input Configuration 158

5.3.9 Alternate Function Configuration 159
5.3.10 Debouncing Functionality 160
5.4 PCU and GPIO Registers 162
5.4.1 Pn_MR1: Port n MUX Control Register 1 164
5.4.2 Pn_MR2: Port n MUX Control Register 2 165
5.4.3 Pn_CR: Port n Control Register 166
5.4.4 Pn_PRCR: Port n Pull-Up/Pull-Down Control Register 166
5.4.5 Pn_DER: Port n Debounce Enable Register 167
5.4.6 Pn_IER: Port n Interrupt Enable Register 168
5.4.7 Pn_ISR: Port n Interrupt Status Register 169
5.4.8 Pn_ICR: Port n Interrupt Control Register 170
5.4.9 Pn_ODR: Port n Output Data Register 170
5.4.10 Pn_IDR: Port n Input Data Register 171
5.4.11 Pn_BSR: Port n Set/Reset Register 172
5.4.12 Pn_BCR: Port n Reset Register 173
5.4.13 PCU_RSTDBCR: Port Reset Debounce Control Register 174
5.4.14 PCU_PIN_REMAP: Pin Remapping Register 175
5.4.15 PCU_PORTEN: Port Access Enable Register 176
5.4.16 PCU and GPIO Register Map Summary 177
6. Embedded Flash Memory 179
6.1 Introduction 179
6.2 Flash Main Features 179
6.3 Flash Functional Description 180
6.3.1 Flash Memory Organization 180
6.3.1.1 Code Flash Memory (A31C14x) 180
6.3.1.2 Code Flash Memory (A31C12x) 181
6.3.2 Read Access Latency 182
6.3.2.1 Code Flash Memory Wait Time 182
6.3.3 Flash Program and Erase Operations 183
6.3.3.1 Access Control for Writing and Erasing Flash Memory 183
6.3.3.2 Access Control for Writing and Erasing User Info Memory 184
6.3.4 Flash Memory Erase Sequences 185
6.3.4.1 Page Erase 185
6.3.4.2 2 KB Sector Erase 186
6.3.4.3 Bulk Erase 187
6.3.5 Flash Memory Programming Sequences 188
6.3.5.1 Standard Programming 188
6.4 Flash Memory Protection 189
6.4.1 Read Protection 190
6.4.1.1 Read Protection: UNPROT (Unprotection) 190
6.4.1.2 Read Protection: LVL1 (Level-1) 191
6.4.1.3 Read Protection: LVL2 (Level-2) 191
6.4.1.4 Read Protection Setting 192
6.4.1.5 Read Protection Removal 192
6.4.1.6 Password Settings 192
6.4.2 Write Protection 193
6.4.2.1 Write Protection for A31C14x Code Flash Memory 194
6.4.2.2 Write Protection for A31C12x Code Flash Memory 195
6.4.3 Securable Memory Area (LVL1, LVL2) 196
6.4.4 Disabling Core Debug Access (LVL2) 196

6.5 Flash Memory Interrupts 197
6.6 Code Flash Memory Controller Registers 198
6.6.1 CFMC_CONF: Code Flash Control Register 199
6.6.2 CFMC_FLSKEY: Code Flash Access Key Register 200
6.6.3 CFMC_INFOKEY: Code Flash User Info Access Key Register 201
6.6.4 CFMC_FLSPROT: Code Flash Protection Register 202
6.6.5 CFMC_INFOPROT: Code Flash Info Protection Register 203
6.6.6 CFMC_CTRL: Code Flash Access Control Register 204
6.6.7 CFMC_STAT: Code Flash Access Status Register 205
6.6.8 CFMC_READPROT: Code Flash Read Protection Register 206
6.6.9 CFMC_PWIN: Code Flash Password Input Register 207
6.6.10 CFMC_WTIMEOUT: Code Flash Write Timeout Register 208
6.6.11 CFMC_PWPRST: Code Flash Password Preset Register 208
6.6.12 CFMC Register Map Summary 209
7. HSICMU 211
7.1 HSICMU Introduction 211
7.2 HSICMU Main Features 211
7.3 HSICMU Functional Description 212
7.3.1 HSICMU Block Diagram 212
7.3.2 Synchronization Input 212
7.3.3 Frequency Error Measurement 213
7.3.4 Frequency Error Evaluation and Automatic Trimming 214
7.3.5 HSICMU Initialization and Configuration 215
7.4 HSICMU Registers 219
7.4.1 HSICMU_CR: HSICMU Control Register 220
7.4.2 HSICMU_CFG: HSICMU Configure Register 221
7.4.3 HSICMU_ISR: HSICMU Interrupt and Status Register 222
7.4.4 HSICMU_ICR: HSICMU Control Interrupt and Status Register 223
7.4.5 HSICMU_ATRIM: HSICMU Control Auto-Trim Value Register 224
7.4.6 HSICMU_CAP: HSICMU Capture Value Register 224
7.4.7 HSICMU_THR: HSICMU Frequency Error Rate Threshold Register 225
7.4.8 HSICMU Register Map Summary 226
8. DMA 227
8.1 DMA Introduction 227
8.1.1 Modules Supported by DMA 227
8.2 DMA Functional Description 228
8.2.1 DMA Transfer Timing from Peripheral to Memory 229
8.2.2 DMA Transfer Timing from Memory to Peripheral 230
8.2.3 DMA Transfer 230
8.3 DMA Registers 231
8.3.1 DMAn_CR: DMA Channel n Control Register 232
8.3.2 DMAn_SR: DMA Channel n Status Register 234
8.3.3 DMAn_PAR: DMA Channel n Peripheral Address Register 235
8.3.4 DMAn_MAR: DMA Channel n Memory Address Register 235
8.3.5 DMA Register Map Summary 236
9. WDT 237
9.1 WDT Introduction 237
9.2 WDT Main Features 237
9.3 WDT Functional Description 238
9.3.1 WDT Block Diagram 238

9.3.2 Enabling WDT 238
9.3.3 Controlling Down-Counter 239
9.3.4 Interrupt Feature 240
9.3.4.1 Interrupt Mode Operation Timing Diagram 240
9.3.5 How to Program Watchdog Timeout 240
9.3.5.1 Basic Reset Operation 240
9.3.6 Debug Mode 241
9.4 WDT Registers 242
9.4.1 WDT_LR: WDT Load Register 243
9.4.2 WDT_CNT: WDT Current Count Register 244
9.4.3 WDT_CON: WDT Control Register 245
9.4.4 WDT_AEN: WDT Access Enable Register 246
9.4.5 WDT Register Map Summary 247
10. 16-bit Timer 248
10.1 16-bit Timer Introduction 248
10.2 16-bit Timer Main Features 249
10.3 16-bit Timer Functional Description 250
10.3.1 Block Diagram 250
10.3.2 Pins and Internal Signals 251
10.3.3 Clock Selection 253
10.3.3.1 Peripheral System Clock (PCLK) 253
10.3.3.2 Timer External Clock (SCU_MCCR3 Register) 253
10.3.3.3 Input to Pin TnIO for Timer Clock Source 254
10.3.4 Time-Base Unit 255
10.3.4.1 Prescaler Description 256
10.3.5 Periodic Mode 257
10.3.5.1 Operation According to TIMERn_GRA and TIMERn_GRB Settings 258
10.3.5.2 Operation According to UAO, OUTPOL, and STARTLVL 259
10.3.5.3 Notes on Counter Mode Operation 261
10.3.5.4 Counter Mode Register Setting Sequence 261
10.3.6 Capture Mode 262
10.3.6.1 Capture Mode Register Setting Sequence 263
10.3.7 PWM Mode 264
10.3.7.1 PWM Mode Operation Characteristics and Precautions 265
10.3.7.2 PWM Mode Register Setting Sequence 265
10.3.8 One-Shot Mode 266
10.3.8.1 One-Shot Mode Operation Characteristics and Precautions 267
10.3.8.2 One-Shot Mode Register Setting Sequence 268
10.3.9 ADC Trigger 269
10.3.9.1 ADC Trigger Operation Characteristics and Precautions 270
10.3.9.2 Register Setting Sequence for ADC Trigger Operation 271
10.3.10 Timer Synchronization 272
10.3.10.1 Delayed Timer Synchronization 274
10.3.10.2 Timer Synchronization Operation Characteristics and Precautions 275
10.3.10.3 Register Setting Sequence for Synchronous Function Operation 276
10.3.11 Direction Bit Output 277
10.3.12 Debug Mode 277
10.3.13 Interrupts 277
10.4 Timer Registers 278

10.4.1 TIMERn_CR1: Timer n Control Register 1 279
10.4.2 TIMERn_CR2: Timer n Control Register 2 281
10.4.3 TIMERn_PRS: Timer n Prescaler Register 282
10.4.4 TIMERn_GRA: Timer n General Data Register A 283
10.4.5 TIMERn_GRB: Timer n General Data Register B 284
10.4.6 TIMERn_CNT: Timer n Count Register 285
10.4.7 TIMERn_SR: Timer n Status Register 285
10.4.8 TIMERn_IER: Timer n Interrupt Enable Register 286
10.4.9 TIMERn_TRGPNT: Timer n Trigger Point Register 286
10.4.10 TIMERn_SYNC: Timer n Sync Configuration Register 287
10.4.11 Timer Register Map Summary 288
11. FRT 289
11.1 FRT Introduction 289
11.2 FRT Main Features 289
11.3 FRT Functional Description 290
11.3.1 FRT Block Diagram 290
11.3.2 FRT Clock Selection 290
11.3.3 FRT Operation Mode 291
11.3.3.1 Match Interrupt Operation 291
11.3.3.2 Overflow Interrupt Operation 292
11.3.3.3 Register Setting Procedure by FRT Operation Mode 293
11.3.4 FRT Low-Power Modes 294
11.3.4.1 Description of Steps of Low-Power Mode 294
11.4 FRT Registers 295
11.4.1 FRT_CTRL: FRT Control Register 296
11.4.2 FRT_MCNT: FRT Match Counter Register 296
11.4.3 FRT_CNT: FRT Counter Register 297
11.4.4 FRT_STAT: FRT Status Register 297
11.4.5 FRT Register Map Summary 298
12. UART 299
12.1 UART Introduction 299
12.2 Main Features 300
12.2.1 UART Implementation 300
12.3 UART Functional Description 301
12.3.1 UART Block Diagram 302
12.3.2 UART Pins and Signals 303
12.3.2.1 UART Bidirectional Communications 303
12.3.3 UART Character Description 304
12.3.4 UART Transmitter 307
12.3.4.1 Data Format 307
12.3.4.2 Transmit Interrupt 308
12.3.4.3 Character Transmission 308
12.3.4.4 Configurable Stop Bits 309
12.3.4.5 Character Transmission Procedure 310
12.3.4.6 Single Byte Communication 311
12.3.4.7 Break Condition 311
12.3.4.8 Inter-Frame Delay for Data Transmission 311
12.3.5 UART Receiver 312
12.3.5.1 Receive Data Sampling Timing 312
12.3.5.2 Start Bit Detection 313

12.3.5.3 Data Bit Sampling 315
12.3.5.4 Data Sampling Strategy 316
12.3.5.5 Character Reception 317
12.3.5.6 Break Condition 317
12.3.5.7 Overrun Error 317
12.3.5.8 Framing Error 318
12.3.5.9 Parity Error 318
12.3.5.10 Configurable Stop Bits During Reception 318
12.3.6 UART Baud-Rate Generation 319
12.3.7 Tolerance of UART Receiver to Clock Deviation 321
12.3.8 UART Parity Control 322
12.3.8.1 Even Parity 322
12.3.8.2 Odd Parity 322
12.3.8.3 Stick Parity 323
12.3.8.4 Various Configurations of Parity Bit 323
12.3.8.5 Parity Checking in Reception 323
12.3.8.6 Parity Generation in Transmission 323
12.3.9 Burst Transfer Using DMA and UART 324
12.3.9.1 Transmission using DMA 325
12.3.9.2 Reception using DMA 326
12.3.9.3 Error Flagging and Interrupt Generation in Communication 327
12.4 UART Interrupts 328
12.5 UART Registers 329
12.5.1 UARTn_RBR: UART n Receive Data Buffer Register 330
12.5.2 UARTn_THR: UART n Transmit Data Hold Register 330
12.5.3 UARTn_IER: UART n Interrupt Enable Register 331
12.5.4 UARTn_IIR: UART n Interrupt ID Register 332
12.5.5 UARTn_LCR: UART n Line Control Register 333
12.5.6 UARTn_DCR: UART n Data Control Register 335
12.5.7 UARTn_LSR: UART n Line Status Register 336
12.5.8 UARTn_ELSR: UART n Extended Line Status Register 338
12.5.9 UARTn_BDR: UART n Baud-Rate Divisor Latch Register 340
12.5.10 UARTn_BFR: UART n Baud-Rate Fraction Counter Register 340
12.5.11 UARTn_IDTR: UART n Inter-Frame Delay Time Register 341
12.5.12 UART Register Map Summary 342
13. I2C 343
13.1 I2C Introduction 343
13.2 I2C Main Features 343
13.3 I2C Implementation 344
13.4 I2C Functional Description 345
13.4.1 I2C Block Diagram 345
13.4.2 I2C Pins and Signals 346
13.4.3 I2C Mode Selection 346
13.4.4 I2C Initialization 347
13.4.4.1 Enabling and Disabling Peripherals 347
13.4.4.2 Noise Canceller 347
13.4.4.3 Using Noise Canceller in I2C Module 348
13.4.5 I2C Software Reset 349
13.4.6 I2C Module Protocol 350
13.4.6.1 I2C Bit Transfer 350

13.4.6.2 START/Repeated START/STOP 351
13.4.6.3 Acknowledge 352
13.4.6.4 Synchronization 353
13.4.6.5 Arbitration 354
13.4.6.6 Data Transfer 355
13.4.6.7 Master Transmit Timing 356
13.4.6.8 Master Receive Timing 357
13.4.6.9 Slave Transmit Timing 358
13.4.6.10 Slave Receive Timing 359
13.4.7 I2C Slave Mode 360
13.4.7.1 Slave Transmitter 360
13.4.7.2 Slave Receiver 362
13.4.8 I2C Master Mode 364
13.4.8.1 Master Transmitter 364
13.4.8.2 Master Receiver 367
13.5 I2C Interrupts 370
13.6 I2C Registers 371
13.6.1 I2Cn_DR: I2C Data Register 372
13.6.2 I2Cn_SR: I2C Status Register 373
13.6.3 I2Cn_SAR: I2C Slave Address Register 374
13.6.4 I2Cn_CR: I2C Control Register 375
13.6.5 I2Cn_SCLL: I2C SCL Low-Duration Register 376
13.6.6 I2Cn_SCLH: I2C SCL High-Duration Register 377
13.6.7 I2Cn_SDH: I2C SDA Hold Register 378
13.6.8 I2C Register Map Summary 379
14. USART 380
14.1 USART Introduction 380
14.2 USART Main Features 380
14.3 UART Functional Description 381
14.3.1 USART Block Diagram 382
14.3.2 USART Pins and Internal Signals 384
14.3.3 USART Clock Generation 385
14.3.4 External Clock (SCKn) 386
14.3.5 Synchronous Mode Operation 386
14.3.6 UART Data Format 387
14.3.7 UART Parity Bit 388
14.3.8 UART Transmitter 389
14.3.8.1 UART Sending TX Data 389
14.3.8.2 UART Transmitter Flag and Interrupt 389
14.3.8.3 UART Parity Generator 390
14.3.8.4 UART Disabling Transmitter 390
14.3.9 UART Receiver 391
14.3.9.1 UART Receiving RX Data 391
14.3.9.2 UART Receiver Flag and Interrupt 392
14.3.9.3 UART Parity Checker 392
14.3.9.4 UART Disabling Receiver 392
14.3.9.5 Asynchronous Data Reception 393
14.3.10 UART Low-Power Mode 395
14.3.10.1 Description of Steps of Low-Power Mode 395
14.3.11 SPI Mode 396

14.3.11.1 SPI Clock Formats and Timing 397
14.3.12 Baud Rate Settings (Example) 400
14.3.13 0% Error Baud Rate 401
14.3.14 Receive Time Out (RTO) 401
14.4 USART Registers 402
14.4.1 USARTn_CR1: USART n Control Register 1 403
14.4.2 USARTn_CR2: USART n Control Register 2 405
14.4.3 USARTn_ST: USART n Status Register 407
14.4.4 USARTn_BDR: USART n Baud Rate Generation Register 409
14.4.5 USARTn_DR: USART n Data Register 409
14.4.6 USARTn_BFR: USART n Baud-Rate Fraction Counter Register 410
14.4.7 USARTn_RTO: USART n RTO Register 410
14.4.8 USART Registers Map Summary 411
15. ADC 412
15.1 ADC Introduction 412
15.2 ADC Main Features 413
15.3 ADC Functional Description 414
15.3.1 ADC Block Diagram 414
15.3.2 ADC Pins and Signals 415
15.3.3 ADC Connectivity 417
15.3.3.1 Internal Wiring of ADC Channels 417
15.3.3.2 ADC Channels and External Input Pin 418
15.3.4 ADC Clocks (ACLK) 419
15.3.4.1 ADC Module Control 419
15.3.4.2 ADC Clock Sources 420
15.3.5 ADC On-Off Control (ADEN) 421
15.3.5.1 Software Procedure to Enable ADC 421
15.3.5.2 Software Procedure to Disable ADC 421
15.3.6 Constraints when Writing ADC Control Bits 422
15.3.6.1 When Writing ADC Control Bit 422
15.3.7 Channel Selection 423
15.3.7.1 ADC Analog Input Channel 423
15.3.7.2 ADC Conversion Channel Selection 424
15.3.7.3 Register Access when Changing A/D Conversion Channel 425
15.3.8 Single Mode Conversion (ADMOD[1:0] = '00', SEQCNT[2:0] = 0) 426
15.3.9 Sequential Mode Conversion (ADMOD[1:0] = '00', SEQCNT[2:0] > 0) 428
15.3.10 Burst Mode Conversion (ADMOD[1:0] = '01', SEQCNT > 0) 432
15.3.11 Multiple Mode Conversion (ADMOD[1:0] = '10', SEQCNT[2:0] > 0) 435
15.3.12 Starting Conversions (ASTART, Start Trigger, ARST) 437
15.3.12.1 Start A/D Conversion (ASTART) 437
15.3.12.2 Reset A/D Conversion (ARST) 437
15.3.12.3 Start A/D Conversion by Trigger Source 437
15.3.13 ADC Timing 438
15.3.14 Stopping Ongoing Conversion (ASTOP) 440
15.3.15 Conversion on External Trigger (TRGSEL[2:0], SEQTRGx[3:0] and TRGSRC[3:0], TRGIE, TRGIF) 441
15.3.15.1 ADC Trigger Source Selection (TRGSEL[2:0]) 441
15.3.15.2 ADC Trigger Channel Settings (SEQTRGx[3:0] and TRGSRC[3:0]) 442
15.3.15.3 ADC Trigger Start Control 442

15.3.15.4 ADC Trigger Interrupt and Flag (TRGIE, TRGIF) 443
15.3.16 End-of-Conversion, End-of-Sampling Phase (EOC) 444
15.3.17 End-of-Sequence Conversion (EOSIE, EOSIF) 445
15.3.18 Data Management and Information 447
15.3.18.1 Data Register and Data Alignment by Software 447
15.3.18.2 Conversion Data Offset Alignment 448
15.3.18.3 Channel and Trigger Information 449
15.3.19 Trigger Pending Function in Sequential Mode 450
15.3.20 ADC Low-Power Modes 450
15.3.21 A/D Conversion using DMA 451
15.3.22 ADC Overrun (OVR) in DMA Mode 453
15.3.23 Comparison of ADC Sensing Value 454
15.4 ADC Interrupts 456
15.5 ADC Registers 457
15.5.1 ADC_MR: ADC Mode Register 458
15.5.2 ADC_CSCR: ADC Current Channel Register 460
15.5.3 ADC_CCR: ADC Clock Control Register 461
15.5.4 ADC_TRG: ADC Trigger Select Register 462
15.5.5 ADC_SCSR1: ADC Channel Select 1 Register 463
15.5.6 ADC_SCSR2: ADC Channel Select 2 Register 464
15.5.7 ADC_CR: ADC Control Register 465
15.5.8 ADC_SR: ADC Status Register 466
15.5.9 ADC_IER: ADC Interrupt Enable Register 467
15.5.10 ADC_DDR: ADC DMA Data Register 468
15.5.11 ADC_DRx: ADC n Data 0 to 7 Register 469
15.5.12 ADC_CMP0R: ADC Channel Comparison 0 Register 470
15.5.13 ADC_CMP1R: ADC Channel Comparison 1 Register 471
15.5.14 ADC Register Map Summary 472
16. Comparator 474
16.1 Comparator Introduction 474
16.2 Comparator Main Features 474
16.3 Comparator Functional Description 475
16.3.1 Comparator Block Diagram 475
16.3.2 Comparator Pins and Internal Signals 476
16.3.3 Comparator Reference and Input Voltage Configuration 478
16.3.4 Comparator Reset and Clock 479
16.3.5 Comparator Hysteresis 480
16.3.6 Comparator Filter 481
16.4 Comparator Low-Power Modes 483
16.5 Comparator Interrupts 483
16.6 Comparator Registers 484
16.6.1 CMPn_CONF: Comparator Configure Register 485
16.6.2 CMPn_CTRL: Comparator Control Register 486
16.6.3 CMPn_STAT: Comparator Status Register 486
16.6.4 CMP_DBCR: Comparator Debounce Control Register 487
16.6.5 Comparator Register Map Summary 488
17. CRC (Cyclic Redundancy Check Calculation Module) 489
17.1 CRC Introduction 489
17.2 CRC Main Features 489
17.3 CRC Functional Description 490

17.3.1 CRC Block Diagram 490
17.3.2 CRC Internal Signals 490
17.3.3 Timing Diagram of CRC Internal Signals for Data Size 491
17.3.4 CRC Operation 493
17.3.4.1 How to Obtain CRC Calculation Result 494
17.3.5 CRC using DMA 495
17.4 CRC Registers 496
17.4.1 CRC_CTRL: CRC Control Register 497
17.4.2 CRC_INIT: CRC Initial Data Register 498
17.4.3 CRC_IDR: CRC Input Data Register 498
17.4.4 CRC_ODR: CRC Output Data Register 499
17.4.5 CRC_STAT: CRC Status Register 499
17.4.6 CRC Register Map Summary 500
18. DBGMCU (Microcontroller Debug Component) 501
18.1 DBGMCU Introduction 501
18.2 DBGMCU Main Features 501
18.3 DBGMCU Functional Description 502
18.3.1 Debug Support for STOP Mode 502
18.3.2 Debug Support for Timers, WDT and FRT 502
18.4 DBGMCU Registers 503
18.4.1 DBGMCU_CR: Debug Microcontroller Control Register 504
18.4.2 DBGMCU Register Map Summary 505
Abbreviation for Registers 506
Glossary 507
Revision History 508

List of Tables

Table 1. A31C14x, A31C12x Memory Capacity 22
Table 2. A31C14x, A31C12x Features and Peripherals 23
Table 3. Summary of A31C14x, A31C12x Peripherals 30
Table 4. Pin Description (A31C14x) 44
Table 5. Pin Description (A31C12x) 49
Table 6. Interrupt Vector Map (A31C14x) (1) 55
Table 7. Interrupt Vector Map (A31C12x) (1) 57
Table 8. Boot Mode Pin List 63
Table 9. SCU Pins 65
Table 10. Clock Sources 66
Table 11. Clock Generation Circuit Input/Output Pins 67
Table 12. HSE Clock Pins 68
Table 13. LSE Clock Pins 69
Table 14. Peripheral Clock Selection 72
Table 15. Clock Monitoring Function 74
Table 16. Clock Source for Clock Monitoring 74
Table 17. Flash Memory Wait Control Recommendation 82
Table 18. Reset Sources for Cold Reset and Warm Reset 84
Table 19. LVR Level 88
Table 20. nRESET Pin 88
Table 21. Power Supply of V_{DD} 91
Table 22. Operation Mode 96
Table 23. DEEP-SLEEP Mode Configuration 98
Table 24. Configurable Clock in Each RUN/SLEEP Mode 100
Table 25. Configurable Clocks in Each DEEP-SLEEP (STOP1, STOP2) Mode 100
Table 26. Base Address of CHIPCONFIG 101
Table 27. Register Map of CHIPCONFIG 101
Table 28. Base Address of SCU 102
Table 29. Register Map of SCU 102
Table 30. External Oscillator Control Settings 132
Table 31. SCU CHIPCONFIG Register Map Summary 140
Table 32. SCU Register Map Summary 140
Table 33. PCU and GPIO Internal Signal 150
Table 34. PCU and GPIO Pins 150
Table 35. GPIO Alternate Function (A31C14x) (1) 153
Table 36. GPIO Alternate Function (A31C12x) (1) 155
Table 37. Interrupt Sources and Corresponding Pins in GPIO Module 156
Table 38. Base Address of PCU and GPIO 162
Table 39. PCU and GPIO Register Map 163
Table 40. PCU and GPIO Register Map Summary 177
Table 41. Code Flash Memory Controller Features (A31C14x) 180
Table 42. Code Flash Memory Controller Features (A31C12x) 181
Table 43. Internal Code Flash Memory Wait Time by Operating Clock 182
Table 44. Available Operating Modes by Protection Level 190
Table 45. Read Protection Operation in UNPROT Level 190
Table 46. Read Protection Operation in Level-1 (LVL 1) 191
Table 47. Read Protection Operation in Level-2 (LVL2) 191

Table 48. Code Flash Memory Write Protection Area in 8 KB Units (A31C144) 194
Table 49. Code Flash Memory Unprotection Area (Top 4 KB) in 512 Bytes Units (A31C144) 194
Table 50. Code Flash Memory Write Protection Area in 8 KB Units (A31C143) 194
Table 51. Code Flash Memory Unprotection Area (Top 4 KB) in 512 Bytes Units (A31C143) 194
Table 52. Code Flash Memory Write Protection Area in 2 KB Units (A31C122) 195
Table 53. Code Flash Memory Unprotection Area (Top 4 KB) in 512 Bytes Units (A31C122) 195
Table 54. Code Flash Memory Write Protection Area in 2 KB Units (A31C121) 195
Table 55. Code Flash Memory Unprotection Area (Top 4 KB) in 512 Bytes Units (A31C121) 195
Table 56. Base Address of CFMC 198
Table 57. CFMC Register Map 198
Table 58. CFMC Register Map Summary 209
Table 59. Examples HSICMU (SYNCSRC[1:0] = 10, LSE) 218
Table 60. Examples HSICMU (SYNCSRC[1:0] = 01, HSE) 218
Table 61. Base Address of HSICMU 219
Table 62. HSICMU Register Map 219
Table 63. HSICMU Register Map Summary 226
Table 64. Base Address of DMA 231
Table 65. DMA Register Map 231
Table 66. DMA PERISEL[3:0] Selection (A31C14x) 232
Table 67. DMA PERISEL[3:0] Selection (A31C12x) 233
Table 68. DMA Register Map Summary 236
Table 69. Prescaled WDT Counter Clock Frequency 239
Table 70. Base Address of WDT 242
Table 71. WDT Register Map 242
Table 72. WDT Register Map Summary 247
Table 73. Input and Output Pins for TIMERn 251
Table 74. Internal Input and Output Signals for TIMERn 252
Table 75. PCLK Clock Selection by CKSEL[2:0] in TIMERn_CR1 253
Table 76. SCU_MCCR3 Register and CKSEL[2:0] in TIMERn_CR1 253
Table 77. TnIO and IOSEL in TIMERn_CR1 254
Table 78. TnIO and CKSEL[2:0] in TIMERn_CR1 254
Table 79. Operations Comparison According to TIMERn_GRA/GRB Settings 258
Table 80. Operation Status by Capture Mode 263
Table 81. One-Shot Mode Operation Setting Table 267
Table 82. Interrupt Requests 277
Table 83. Base Address of 16-bit Timer 278
Table 84. 16-bit Timer Register Map 278
Table 85. TIMER Register Map Summary 288
Table 86. Base Address of FRT 295
Table 87. FRT Register Map 295
Table 88. FRT Register Map Summary 298
Table 89. UART Features 300
Table 90. Pin Assignment of UART: External Pins 303
Table 91. Received Bit Value from Sampled Data 316
Table 92. Examples of Baud-Rate Calculation 319
Table 93. Examples of Calculation of Baud-Rate Using UART Fraction Counter 320
Table 94. Tolerance of UART Receiver 321
Table 95. UART Frame Formats 322
Table 96. Various Configurations to Create Parity Bit 323
Table 97. Base Address of UART 329

Table 98. UART Register Map 329
Table 99. Interrupt ID and Control 332
Table 100. UART Register Map Summary 342
Table 101. Features of I2C 344
Table 102. Pin Assignment of I2C: External Pins 346
Table 103. Base Address of I2C 371
Table 104. I2C Register Map 371
Table 105. I2C Register Map Summary 379
Table 106. Pin Assignment of USART External Pins 384
Table 107. Equations for Calculating USART Baud Rate Register Settings 385
Table 108. Effect of Low-Power Modes on UART 395
Table 109. CPOL Functionality 397
Table 110. Baud Rate Settings Example 400
Table 111. Base Address of USART 402
Table 112. USART Register Map 402
Table 113. USART Register Map Summary 411
Table 114. Pin Assignment of ADC External Pins 415
Table 115. ADC Input/Output Signals 416
Table 116. ADC Channels and External Input Pins (1) 418
Table 117. ADC Input Signal and Input Pin (1) 424
Table 118. ADC Frequency for Values of R_{AIN} 438
Table 119. External Trigger Source Configuration 441
Table 120. Trigger Source Table 442
Table 121. Start Control of AD Trigger Conversion 442
Table 122. Effect of Low-Power Modes on ADC 450
Table 123. Comparison and Flag Generation Conditions by CMPM 454
Table 124. Control Bits and Event Flags for ADC Interrupts 456
Table 125. Base Address of ADC 457
Table 126. ADC Register Map 457
Table 127. ADC Register Map Summary 472
Table 128. Pin Assignment of Comparator: External Pins 476
Table 129. Comparator Input Sources 477
Table 130. Comparator Reference Sources 477
Table 131. Comparator Interrupt Polarity 477
Table 132. Comparator Low-Power Modes 483
Table 133. Base Address of CMP 484
Table 134. CMP Register Map 484
Table 135. Comparator Register Map Summary 488
Table 136. CRC Internal Input/Output Signals 490
Table 137. Base Address of CRC 496
Table 138. CRC Register Map 496
Table 139. CRC Register Map Summary 500
Table 140. Base Address of DBGMCU 503
Table 141. DBGMCU Register Map 503
Table 142. DBGMCU Register Map Summary 505
Table 143. Abbreviation for Registers 506

List of Figures

Figure 1. A31C14x Block Diagram 1
Figure 2. A31C12x Block Diagram 2
Figure 3. A31C14xRL (64-LQFP 10X10) Pinouts 31
Figure 4. A31C144RN (64-LQFP 14X14) Pinouts 32
Figure 5. A31C14xCL (48-LQFP) Pinouts 33
Figure 6. A31C14xCY (48-QFN) Pinouts 34
Figure 7. A31C14xSN (44-LQFP) Pinouts 35
Figure 8. A31C14xKY (32-QFN) Pinouts 36
Figure 9. A31C14xGR (28-TSSOP) Pinouts 37
Figure 10. A31C143FU (20-QFN) Pinouts 37
Figure 11. A31C143FR (20-TSSOP) Pinouts 38
Figure 12. A31C12xKY (32-QFN) Pinouts 39
Figure 13. A31C12xKN (32-LQFP) Pinouts 40
Figure 14. A31C12xGR (28-TSSOP) Pinouts 41
Figure 15. A31C12xLU (20-QFN) Pinouts 42
Figure 16. A31C12xFU (20-QFN) Pinouts 43
Figure 17. A31C12xFR (20-TSSOP) Pinouts 43
Figure 18. System Block Diagram (A31C14x) 52
Figure 19. System Block Diagram (A31C12x) 53
Figure 20. Memory Map (A31C14x) 60
Figure 21. Memory Map (A31C12x) 61
Figure 22. SCU Block Diagram 65
Figure 23. Clock Configuration 67
Figure 24. HSE Crystal/Ceramic Oscillator 68
Figure 25. LSE External Clock 69
Figure 26. Miscellaneous Clock Configuration 71
Figure 27. Clock Monitoring Detection Range 73
Figure 28. MCLK Monitoring Workflow 76
Figure 29. HSE Monitoring Workflow 78
Figure 30. LSE Monitoring Workflow 80
Figure 31. Clock Change Procedure 81
Figure 32. Peripheral Clock Selection 83
Figure 33. Reset Tree Configuration 85
Figure 34. Timing Diagram of Power-Up Procedure (Cold Reset) 86
Figure 35. Warm Reset Timing Diagram 87
Figure 36. LVR Diagram 88
Figure 37. POR and LVR Timing Diagram 92
Figure 38. LVI Block Diagram 93
Figure 39. LVR Block Diagram 94
Figure 40. POR and LVR Block Diagram 94
Figure 41. Transition Between Operation Modes 96
Figure 42. SLEEP Mode Operation Sequence 97
Figure 43. DEEP-SLEEP Mode Operation Sequence 99
Figure 44. PCU and GPIO Block Diagram 147
Figure 45. I/O Port Block Diagram (GPIOs) 148
Figure 46. I/O Port Block Diagram (ADC, CMP, and External Oscillator Pins) 149
Figure 47. Output Port Block Diagram 157

Figure 48. Input Port Block Diagram 158
Figure 49. Port Function Block Diagram 159
Figure 50. Debouncing Logic Block Diagram 160
Figure 51. Example Timing Diagram of Port Debouncing 161
Figure 52. Code Flash Memory Map (A31C14x) 180
Figure 53. Code Flash Memory Map (A31C12x) 181
Figure 54. Flash Memory Page Erase Timing Diagram 185
Figure 55. Flash Memory 2 KB Sector Erase Timing Diagram 186
Figure 56. Flash Memory Bulk (Full-Chip) Erase Timing Diagram 187
Figure 57. Flash Memory Program Timing Diagram 188
Figure 58. HSICMU Block Diagram 212
Figure 59. HSICMU Counter Behavior 213
Figure 60. HSICMU Initialization Flowchart 216
Figure 61. DMA Block Diagram 227
Figure 62. Diagram of DMA Transfer Timing from Peripheral to Memory 229
Figure 63. Diagram of DMA Transfer Timing from Memory to Peripheral 230
Figure 64. N Number of DMA Transfers 230
Figure 65. WDT Block Diagram 238
Figure 66. Interrupt Mode Operation Timing Diagram with External Clock Set as WDT Clock 240
Figure 67. 16-bit Timer Block Diagram 250
Figure 68. Counter Timing Diagram with Basic Start and Match Operations 256
Figure 69. Normal Periodic Mode Operation 257
Figure 70. Operation Waveform Depending on UAO 259
Figure 71. Operation Waveform According to STARTLVL and OUTPOL 260
Figure 72. Timing Diagram of Capture Mode Operation 262
Figure 73. Timing Diagram of PWM Output Operation 264
Figure 74. Timing Diagram of One-Shot Mode Operation 266
Figure 75. Timing Diagram of ADC Triggering (TRGMOD[1:0] = ‘00’) 269
Figure 76. Timing Diagram of ADC Triggering (TRGMOD[1:0] = ‘10’) 269
Figure 77. Timing Diagram of ADC Triggering (TRGMOD[1:0] = ‘11’) 270
Figure 78. Timer Synchronization Example (SSYNC = ‘1’, CSYNC = ‘1’) 272
Figure 79. Timer Synchronization Example (CSYNC = ‘1’) 273
Figure 80. Delayed Timer Synchronization Example (SSYNC = ‘1’) 274
Figure 81. SSYNC Operation Example 275
Figure 82. CSYNC Operation Example 275
Figure 83. FRT Block Diagram 290
Figure 84. Match Interrupt Operation Timing Diagram 291
Figure 85. Overflow Interrupt Operation Timing Diagram 292
Figure 86. Timing Diagram in Low-Power Mode 294
Figure 87. UART Block Diagram 302
Figure 88. Data Inversion Control Diagram 304
Figure 89. Timing Diagram of 8-bit Data Length 305
Figure 90. Timing Diagram of 7-bit Data Length 305
Figure 91. Timing Diagram of 6-bit Data Length 306
Figure 92. Timing Diagram of 5-bit Data Length 306
Figure 93. Example Format of Transmit Data 307
Figure 94. Transmit Interrupt Timing Diagram 308
Figure 95. Example Format of Transmit Data 311
Figure 96. Sampling Timing of UART Receiver 312
Figure 97. Start Bit Detection of Single-Sampling Timing 313

Figure 98. Start Bit Detection of Multi-Sampling Timing 314
Figure 99. Single-Sampling Timing of Data Bit 315
Figure 100. Multi-Sampling Timing of Data Bit 315
Figure 101. Data Sampling Timing 316
Figure 102. Transmission using DMA 325
Figure 103. Reception using DMA 326
Figure 104. Error Flagging and Interrupt Generation Timing Diagram 327
Figure 105. I2C Block Diagram 345
Figure 106. Noise Canceller Timing 347
Figure 107. I2C Bus Bit Transfer Timing Diagram 350
Figure 108. START and STOP Conditions 351
Figure 109. I2C Bus Response 352
Figure 110. Clock Synchronization During Arbitration 353
Figure 111. Arbitration Process between Two Masters 354
Figure 112. I2C Bus Data Transfer 355
Figure 113. Master Transmit Timing 356
Figure 114. Master Receive Timing 357
Figure 115. Slave Transmit Timing 358
Figure 116. Slave Receive Timing 359
Figure 117. Slave Transmitter Flowchart 360
Figure 118. Slave Receiver Flowchart 362
Figure 119. Master Transmitter Flowchart 364
Figure 120. Master Receiver Flowchart 367
Figure 121. Internal Delay (INTDEL[1:0]) in Master Mode 375
Figure 122. SCL Low (SCLL[15:0]) Timing 376
Figure 123. SCL High (SCLH[15:0]) Timing 377
Figure 124. SDA Hold (SDH[14:0]) Timing 378
Figure 125. UART Block Diagram 382
Figure 126. SPI Block Diagram 383
Figure 127. Clock Generation Block Diagram 385
Figure 128. Synchronous Mode SCKn Timing 386
Figure 129. Frame Formant (USART) 387
Figure 130. Asynchronous Start Bit Sampling 393
Figure 131. Asynchronous Data and Parity Bit Sampling 394
Figure 132. Stop Bit Sampling and Next Start Bit Sampling 394
Figure 133. USART SPIn Clock Formats (CPHAn = 0) 398
Figure 134. USART SPIn Clock Formats (CPHAn = 1) 399
Figure 135. 0% Error Baud Rate Diagram 401
Figure 136. 12-bit ADC Block Diagram 414
Figure 137. Internal Wiring Diagram of 12-bit ADC Channels 417
Figure 138. Block Diagram of ADC Clock 420
Figure 139. ADC Single Mode Timing (ADMOD[1:0] = ‘00’) 427
Figure 140. ADC Sequential Mode Timing (ADMOD[1:0] = ‘00’ and SEQCNT[2:0] = 7) 430
Figure 141. ADC Trigger Timing in Sequential Mode (SEQCNT[2:0] = ‘111’, 8-Sequential Conversion) 431
Figure 142. ADC Burst Mode Timing (ADMOD[1:0] = ‘01’) 434
Figure 143. A/D Conversion Time 438
Figure 144. Block Diagram of Typical Connection using ADC 439
Figure 145. Stopping Ongoing A/D Conversions 440
Figure 146. Example of A/D Conversion Timing in Sequence Mode 446

Figure 147. Left Alignment (Unsigned Value) 447
Figure 148. Right Alignment (Unsigned Value) 448
Figure 149. Channel and Trigger Source Information of A/D Conversion Sequence 449
Figure 150. Block Diagram of ADC Internal Comparator 454
Figure 151. Timing Diagram of Comparison between A/D Conversion Results and Reference Value 455
Figure 152. Comparator Block Diagram 475
Figure 153. Schematic Diagram of Comparator Reset Signal 479
Figure 154. Comparator Hysteresis 480
Figure 155. Comparator Filter 481
Figure 156. Comparator Filter Timing 482
Figure 157. CRC Block Diagram 490
Figure 158. Timing Diagram of CRC Internal Signals: 8-bit Size Input and CRC-8 Outputs 491
Figure 159. Timing Diagram of CRC Internal Signals: 32-bit Size Input and CRC-8 Outputs 491
Figure 160. Timing Diagram of CRC Internal Signals: 32-bit Size Input and CRC-16 Outputs 492
Figure 161. Timing Diagram of CRC Internal Signals: 32-bit Size Input and CRC-32 Outputs 492
Figure 162. Timing Diagram of CRC Operation using DMA 495

1. Description

The A31C14x and A31C12x series is a high-performance and energy-efficient 32-bit microcontroller based on Arm Cortex-M0+ core.

1.1 Product Category Definition

Table 1 provides an overview of the memory capacity for available devices in the A31C14x and A31C12x.

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. A31C14x, A31C12x Memory Capacity

Device	Memory Capacity	
	Code Flash	RAM
A31C144	128 KB	8 KB
A31C143	64 KB	8 KB
A31C122	32 KB	6 KB
A31C121	16 KB	6 KB

1.2 Device Overview

Table 2 summarizes and lists product specific features available in the A31C14x and A31C12x considering the largest package.

Table 2. A31C14x, A31C12x Features and Peripherals

Peripheral		Description
Core	CPU	• Max. operating frequency: 32 MHz • 32-bit Arm Cortex-M0+ Core • Register settings in CPU: - Uses general-purpose registers specified by the 32-bit Thumb instruction set: R0-R12 - Main Stack Pointer (MSP) and Process Stack Pointer (PSP): R13 - Link Register (LR): R14 - Program Counter (PC): R15 • Data alignment in little-endian • AHB/APB
	Interrupt	• NVIC (Nested-Vectored Interrupt Controller) • Up to 32 peripheral interrupts available. • 2-bit width of group priority: four levels of priority

Table 2. A31C14x, A31C12x Features and Peripherals (continued)

Peripheral		Description
Memory	Code Flash	• Capacity - A31C144: 128 KB - A31C143: 64 KB - A31C122: 32 KB - A31C121: 16 KB • Erase unit: Page (512 bytes), sector (2 KB) or bulk (full chip) • Program unit: Word (4 bytes) • Read-Protection • Self-programming • Endurance: - A31C144: Page 0 to 251 (126 KB): 10,000 cycles Page 252 to 255 (2 KB): 100,000 cycles - A31C143: Page 0 to 123 (62 KB): 10,000 cycles Page 124 to 127 (2 KB): 100,000 cycles - A31C122: Page 0 to 59 (30 KB): 10,000 cycles Page 60 to 63 (2 KB): 100,000 cycles - A31C121: Page 0 to 27 (14 KB): 10,000 cycles Page 28 to 31 (2 KB): 100,000 cycles • Lifetime: 10 years
	Boot ROM	• Initiating the microcontroller's boot mode by receiving an external input signal at the nBOOT pin • UART mode of USART10 • In-system programming • Users can store information in the internal flash memory by using an application board to write data to the memory.
	SRAM	• Capacity: - A31C14x: 8 KB - A31C12x: 6 KB • Capable of functioning as a program's working area • High-speed execution for time-critical codes • Capability to remap into an interrupt vector area

Table 2. A31C14x, A31C12x Features and Peripherals (continued)

Peripheral		Description
SCU (System Control Unit)	Operating Frequency	• Up to 32 MHz
	Clock	• High-speed Internal (HSI) oscillator: 32 MHz - A31C14x: ±1.5% @ 0°C to +60°C ±3.0% @ −40°C to +85°C ±4.0% @ −40°C to +105°C - A31C12x: ±1.5% @ −20°C to +85°C ±1.0% @ 0°C to +65°C • Low-speed Internal Oscillator (LSI): 32 kHz - A31C14x: ±10% @ 0°C to +60°C, V_{DD} = 3.0V ±30% @ −40°C to +85°C, V_{DD} = 3.0V - A31C12x: ±30% @ −40°C to +85°C • High-speed External (HSE) main oscillator: - 4 MHz to 12 MHz • Low-speed External (LSE) sub-oscillator: 32.768 kHz
	Clock Monitoring	• System Fail-Safe function by clock monitoring - High-speed External (HSE) main oscillator - Low-speed External (LSE) sub-oscillator - Main system clock (MCLK)
	Operation Mode	• RUN mode • SLEEP mode • DEEP-SLEEP (STOP1, STOP2) mode
	Reset	• nRESET Pin reset (PC9) • Core reset • Software reset • POR (Power-On Reset) • LVR (Low-Voltage Reset) • WDTR (Watchdog Timer Reset) • Reset due to clock oscillating error
	POR	• POR generator: Capable of generating a reset signal by detecting an internal 1.48 V
	LVDI	• Four Low-Voltage detection levels • Supports interrupts • Supports wake-up from DEEP-SLEEP mode
	Wake-up	• Wake-up by GPIO pins • Wake-up by FRT • Wake-up by WDT • Wake-up by LVI • Wake-up by USART

Table 2. A31C14x, A31C12x Features and Peripherals (continued)

Peripheral	Description
GPIO	• Input/Output (I/O) port for General-purpose • 64-LQFP - I/O pins: 62 • 48-LQFP, 48-QFN - I/O pins: 46 • 44-LQFP - I/O pins: 42 • 32-QFN, LQFP - I/O pins: 30 • 24-QFN - I/O pins: 22 • 28-TSSOP - I/O pins: 26 • 20-QFN, 20-TSSOP - I/O pins: 18 • 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/pull-down resistors and debouncing • Configurable independently to be set or reset through its corresponding bit • Wake-up events triggered by external asynchronous inputs
DMA Controller	• Direct Memory Access (DMA) supporting peripherals - A31C14x: 6 channels - A31C12x: 4 channels • 8- /16- /32-bit data transfers • Operates with seven types of peripherals. - UART0, UART1 - USART10, USART11, USART12 - CRC - ADC

Table 2. A31C14x, A31C12x Features and Peripherals (continued)

Peripheral		Description
Timer	16-bit Timer	• General-purpose 16-bit up-count timer • A31C14x: 10 channels - 10 (TnIO) input capture channels - 10 (TnIO) output channels • A31C12x: 7 channels - 7 (TnIO) input capture channels - 7 (TnIO) output channels • Timer operating modes - Periodic timer mode - One-shot mode - PWM mode - Capture mode • Interrupt events - Timer/counter match interrupt - Timer overflow interrupt • Free selectable clock sources: - PCLK configurable through the peripheral clock enable register1 (SCU_PCER1) - LSI, LSE, MCLK, HSI, HSE configurable through the Miscellaneous Clock Control Register 3 (SCU_MCCR3) • Timer signals output through TnIO pins • 10-bit prescaler
	WDT	• 32-bit down-count watchdog timer • Reset and periodic interrupts • Free selectable clock sources: - LSI, LSE, HSI, HSE 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. A31C14x, A31C12x Features and Peripherals (continued)

Peripheral		Description
Serial Interface	UART	• 16450 asynchronous serial communication ports • Channel support - A31C14x: 2 channels - A31C12x: 1 channel • 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
	I2C	• Standard I2C communication protocol • Channel support - A31C14x: 3 channels - A31C12x: 2 channels • Master and slave modes • 7-bit addressing for slave mode • Configurable SCL signal's high-/low-periods and SDA signal's hold time
	USART	• Full Duplex Operation. (Independent Serial Receive and Transmit Registers) • Channel support - A31C14x: 3 channels - A31C12x: 2 channels • 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
CRC (Cyclic Redundancy Check)		• CRC operating modes: - CRC32 (0x04C1_1DB7) - CRC16_USB (0x8005) - CRC16_CCITT (0x1021) - CRC8_CCITT (0x07) • Support for byte reversal of input/output data • Supports DMA operation

Table 2. A31C14x, A31C12x Features and Peripherals (continued)

Peripheral		Description
Analog Block	12-bit ADC	• One independent ADC block • Analog input channels - A31C14x: 18 channels - A31C12x: 13 channels • Multiple operating modes: - Single conversion - Sequence conversion - Burst conversion - Multiple conversion • Up to eight sequential conversions • Software triggers • internal trigger sources (timers)
	Comparator	• Two internal comparators with two input sources • Various input and reference sources are available.
Operating voltage		• 1.8 V to 5.5 V
Operating temperature		• Commercial grade (−40°C to +85°C) • Industrial grade (−40°C to +105°C)
Package		• 11 types of package options - 64-LQFP-1010 - 64-LQFP-1414 - 48-LQFP-0707 - 48-QFN-0606 - 44-LQFP-1010 - 32-QFN-0505 - 32-LQFP-0707 - 28-TSSOP - 24-QFN-0404 - 20-TSSOP - 20-QFN-0404

1.3 Summary of A31C14x, A31C12x Peripherals

Descriptions are based on the representative package of the device. Available peripherals may differ depending on the package configuration and should be verified for each type.

Table 3. Summary of A31C14x, A31C12x Peripherals

Peripheral or Function		A31C14x	A31C12x
Clock	HSI	32 MHz	32 MHz
	LSI	32 kHz	32 kHz
	HSE	4 MHz to 12 MHz	4 MHz to 12 MHz
	LSE	32.768 kHz	32.768 kHz
Timers	16-bit GP	10	7
	SysTick	1	1
	WDT	1	1
	FRT	1	1
Communication Interfaces	UART	2	1
	I2C	3	2
	USART	3	2 (UART: 2, SPI: 1)
DMA		6	4
ADC		1	1
Comparator		2	N/A

2. Pinouts and Pin Descriptions

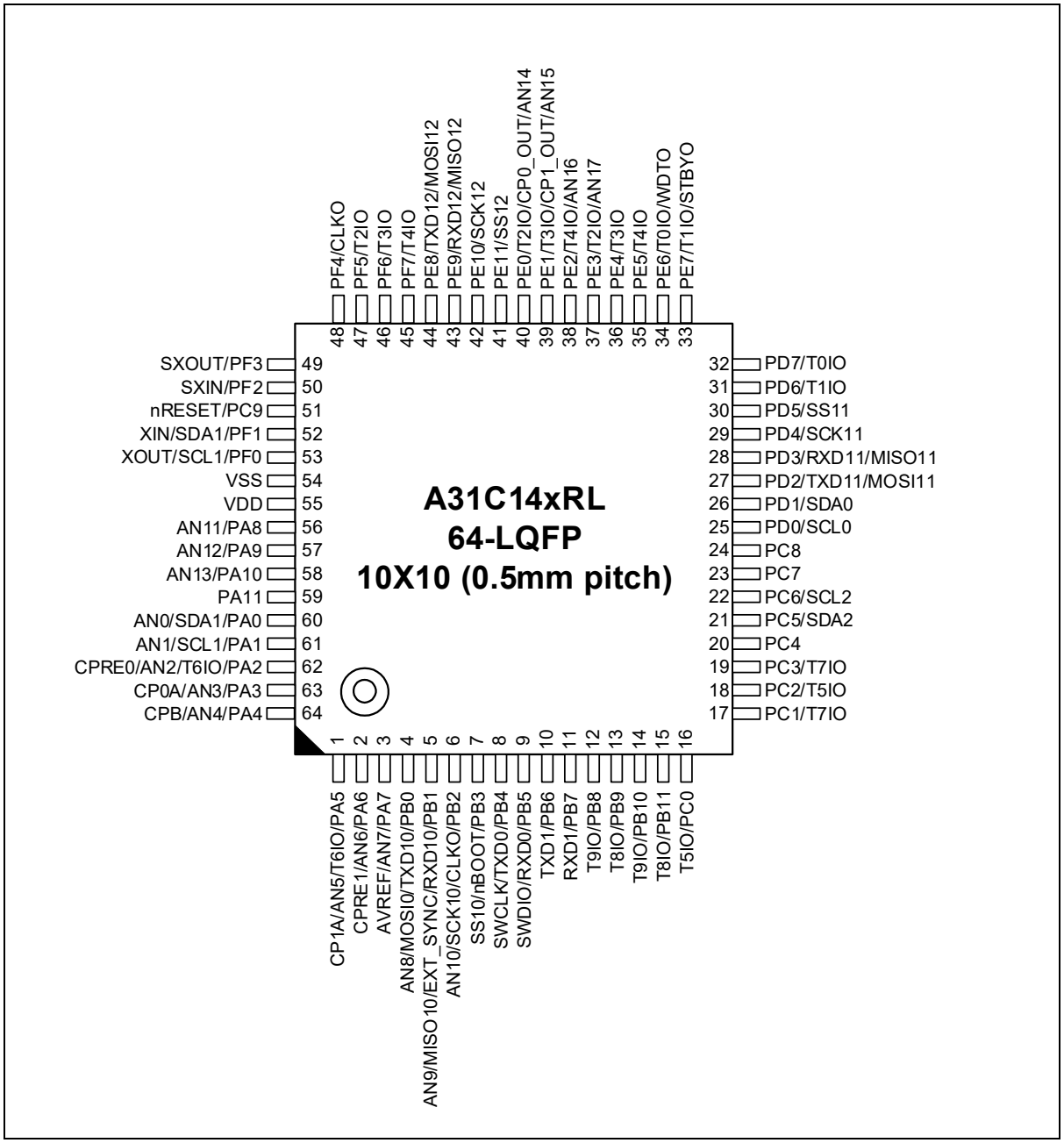
In this chapter, pinouts and pin descriptions of the product are described.

2.1 Pinouts

2.1.1 A31C14x

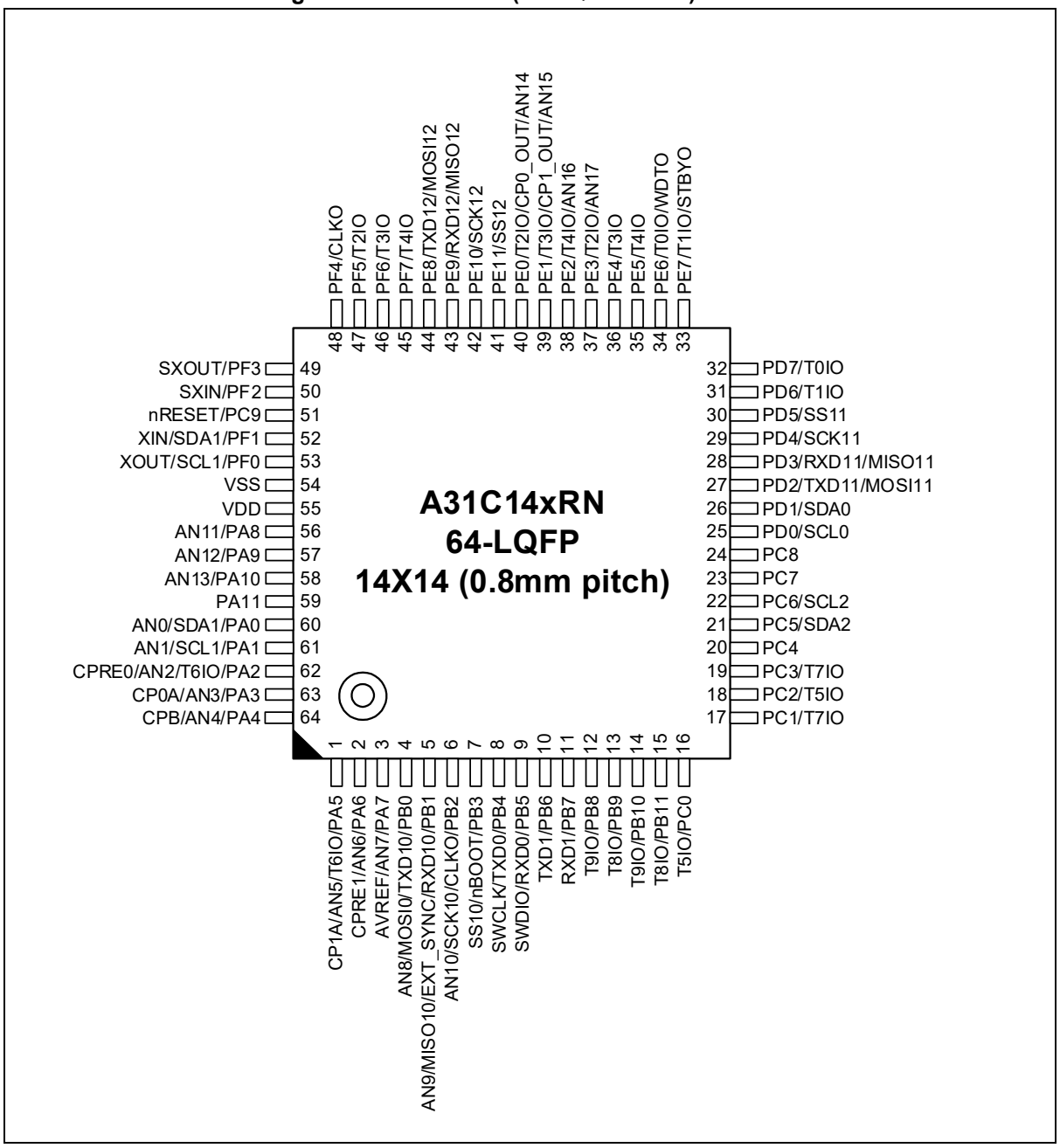
2.1.1.1 A31C14xRL (64-LQFP 10X10)

Figure 3. A31C14xRL (64-LQFP 10X10) Pinouts



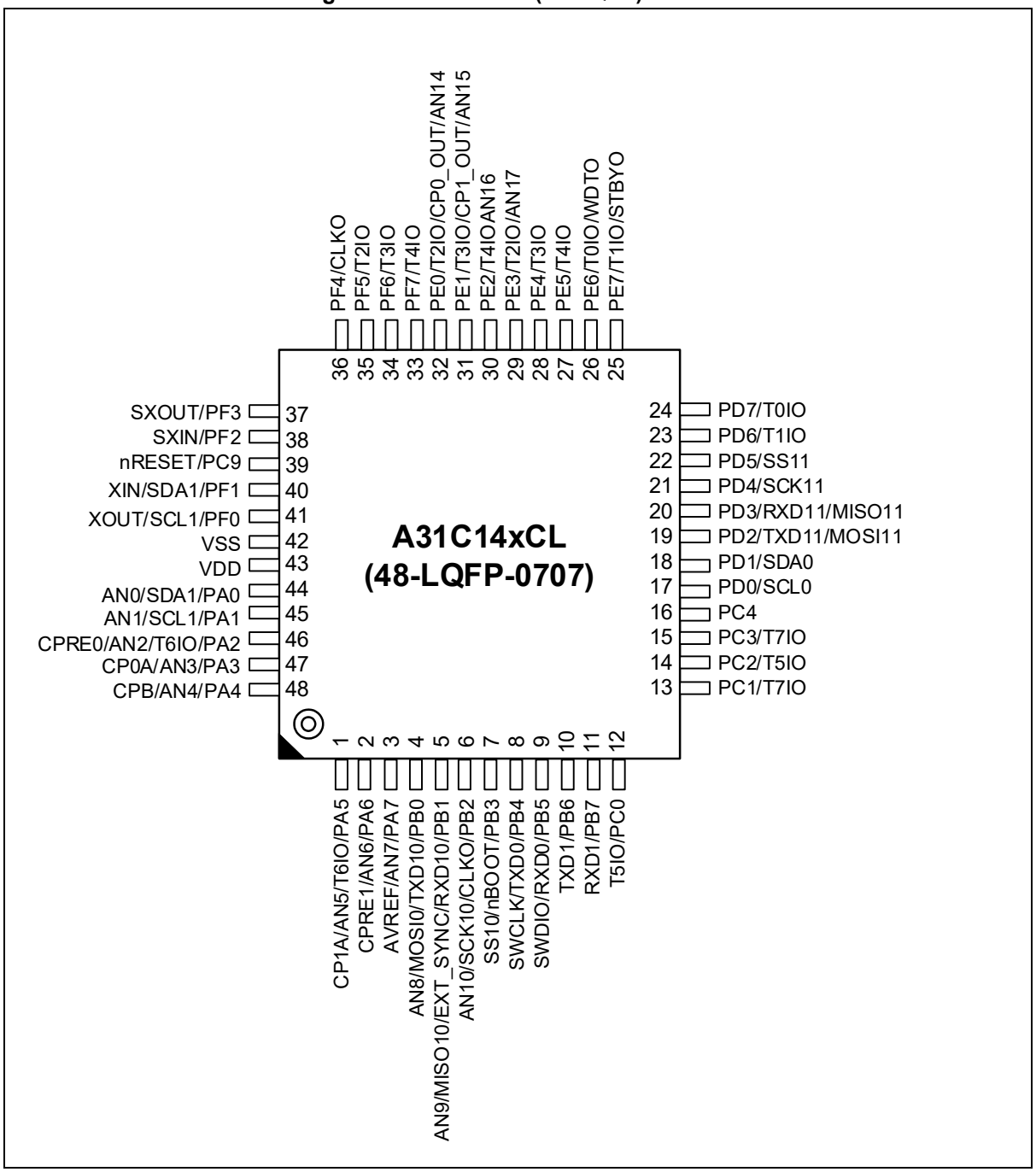
2.1.1.2 A31C144RN (64-LQFP 14X14)

Figure 4. A31C144RN (64-LQFP 14X14) Pinouts



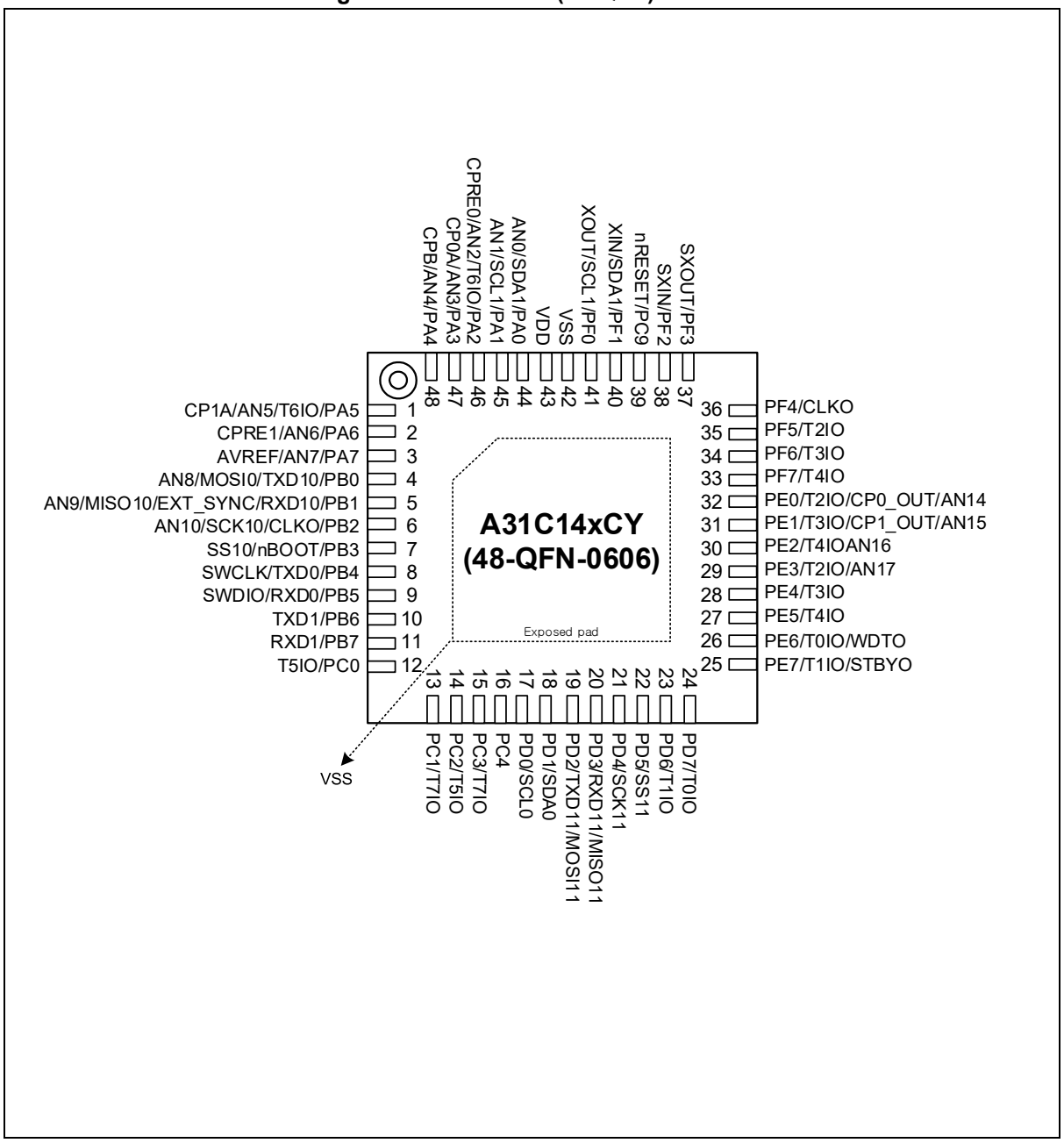
2.1.1.3 A31C14xCL (48-LQFP)

Figure 5. A31C14xCL (48-LQFP) Pinouts



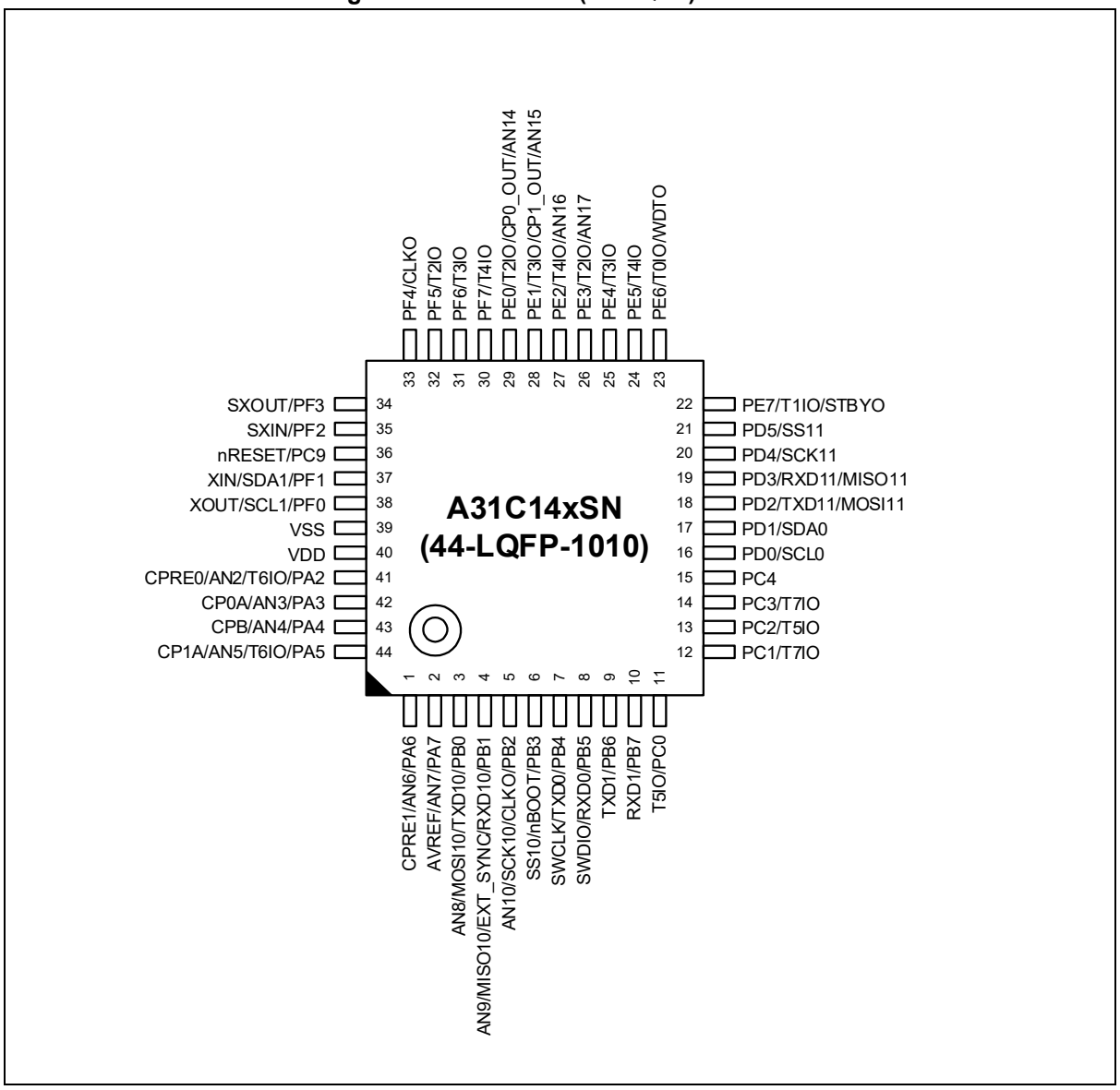
2.1.1.4 A31C14xCY (48-QFN)

Figure 6. A31C14xCY (48-QFN) Pinouts



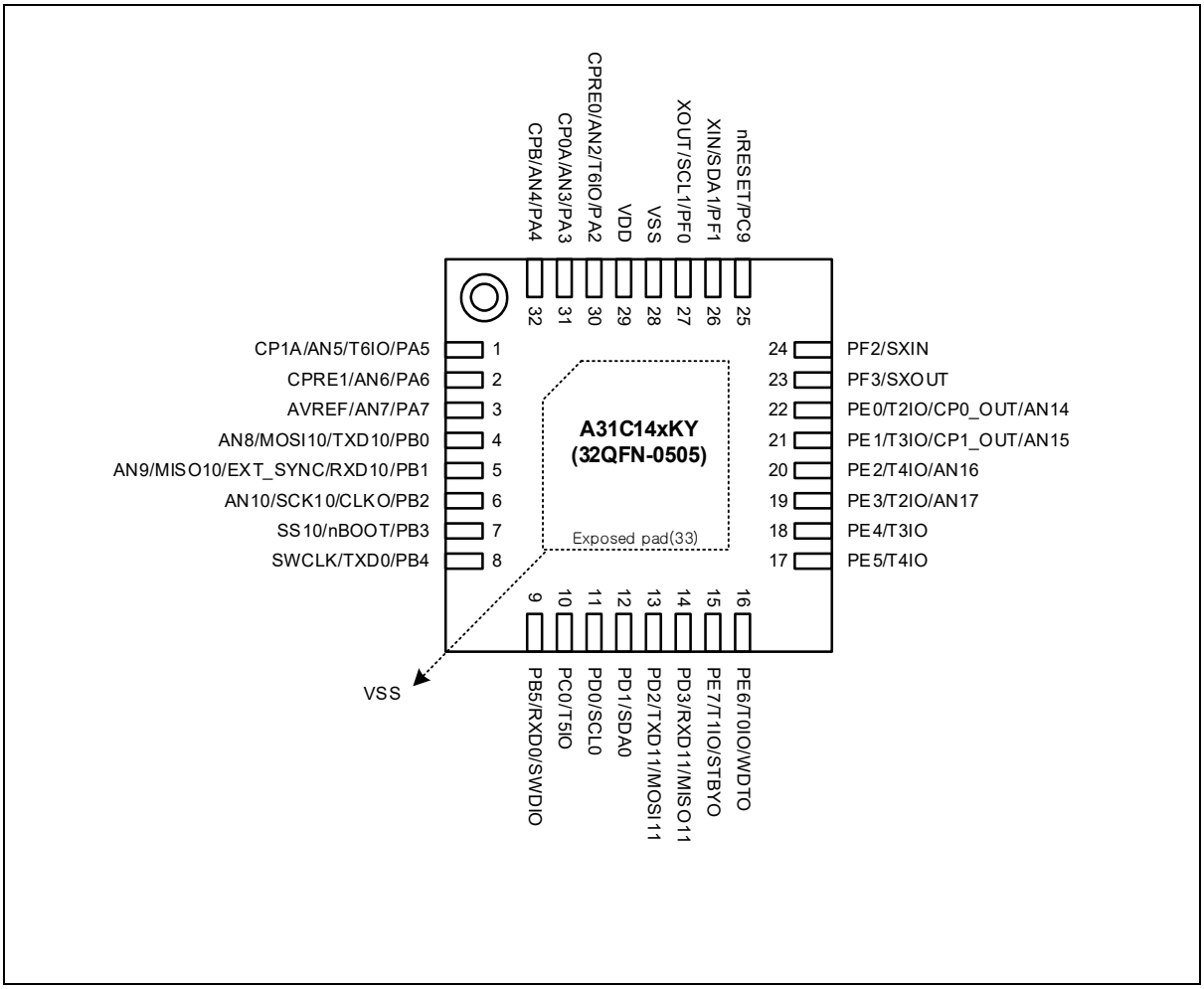
2.1.1.5 A31C14xSN (44-LQFP)

Figure 7. A31C14xSN (44-LQFP) Pinouts



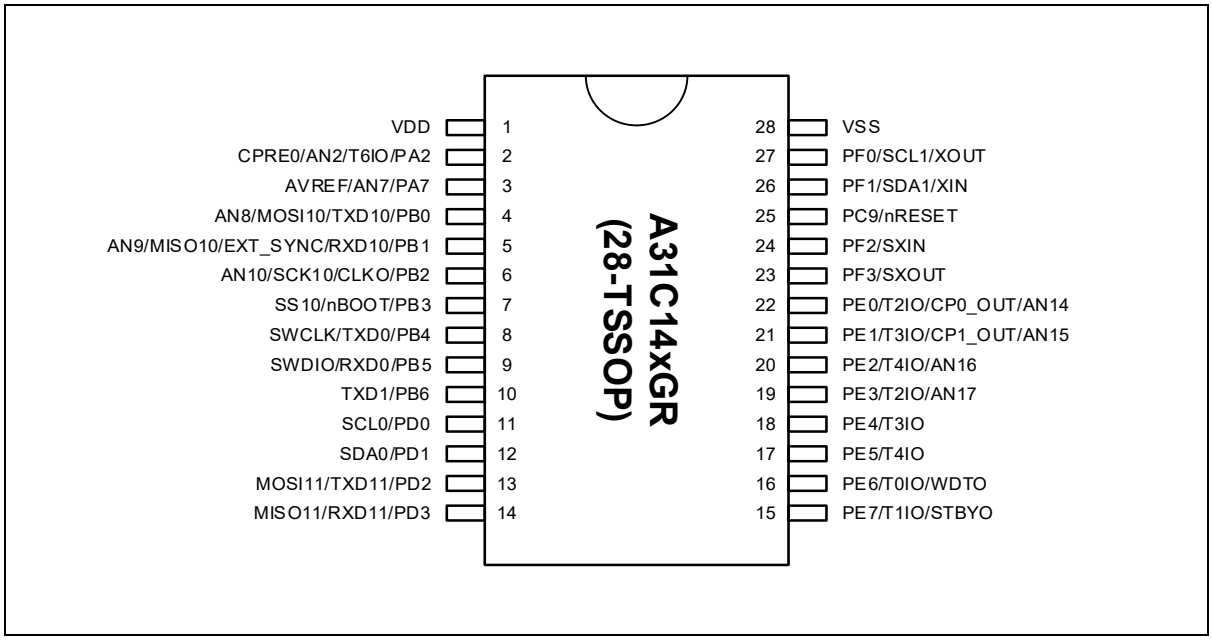
2.1.1.6 A31C14xKY (32-QFN)

Figure 8. A31C14xKY (32-QFN) Pinouts



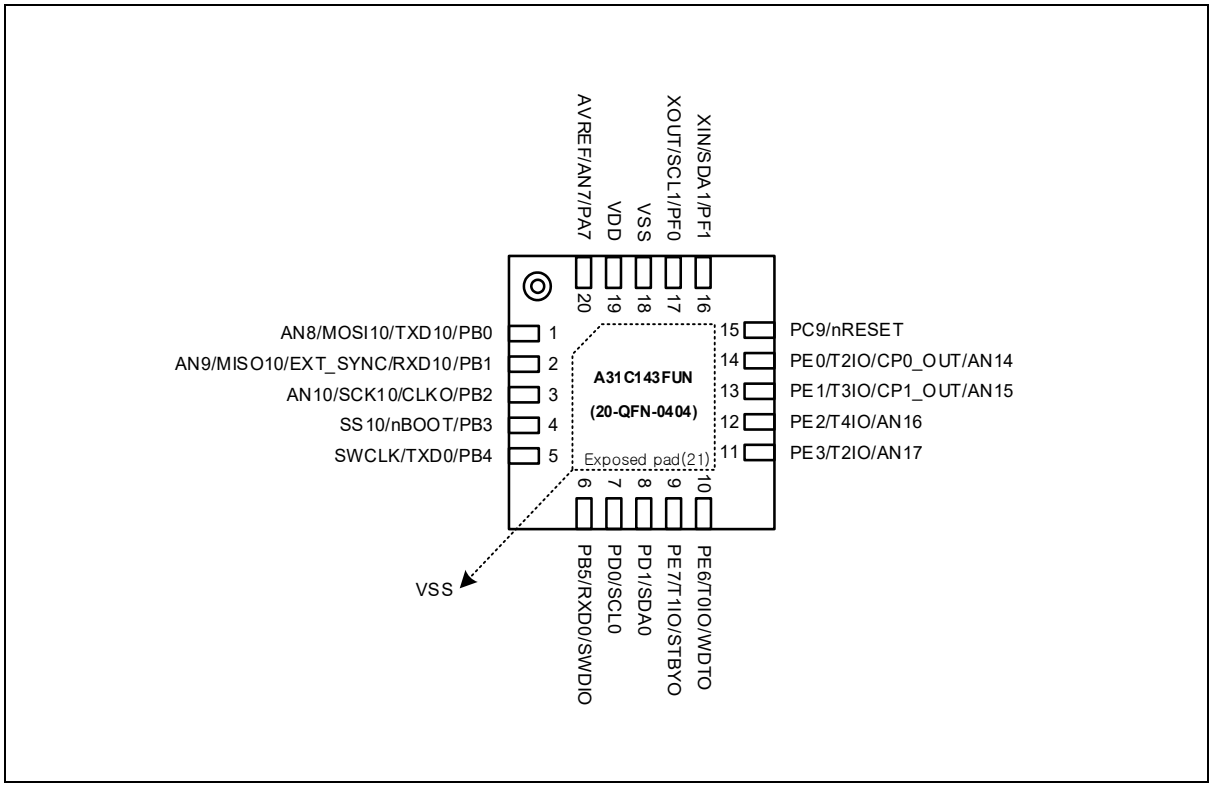
2.1.1.7 A31C14xGR (28-TSSOP)

Figure 9. A31C14xGR (28-TSSOP) Pinouts



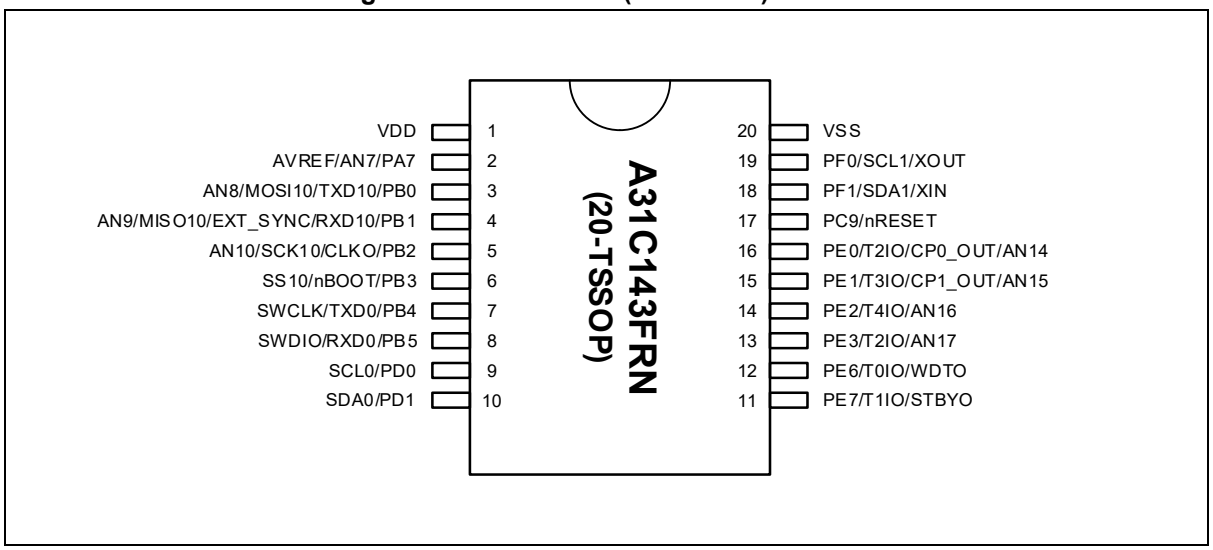
2.1.1.8 A31C143FU (20-QFN)

Figure 10. A31C143FU (20-QFN) Pinouts



2.1.1.9 A31C143FR (20-TSSOP)

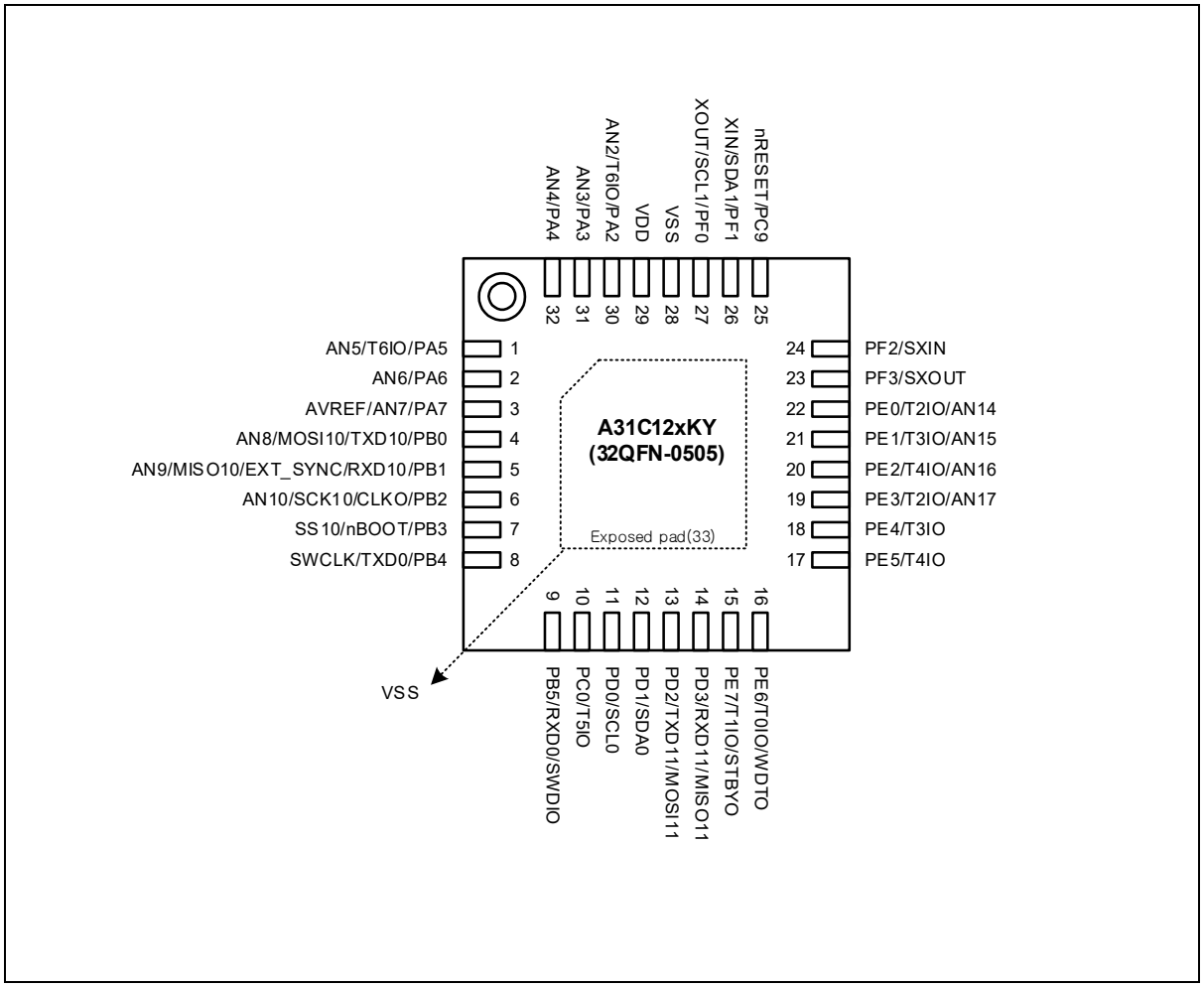
Figure 11. A31C143FR (20-TSSOP) Pinouts



2.1.2 A31C12x

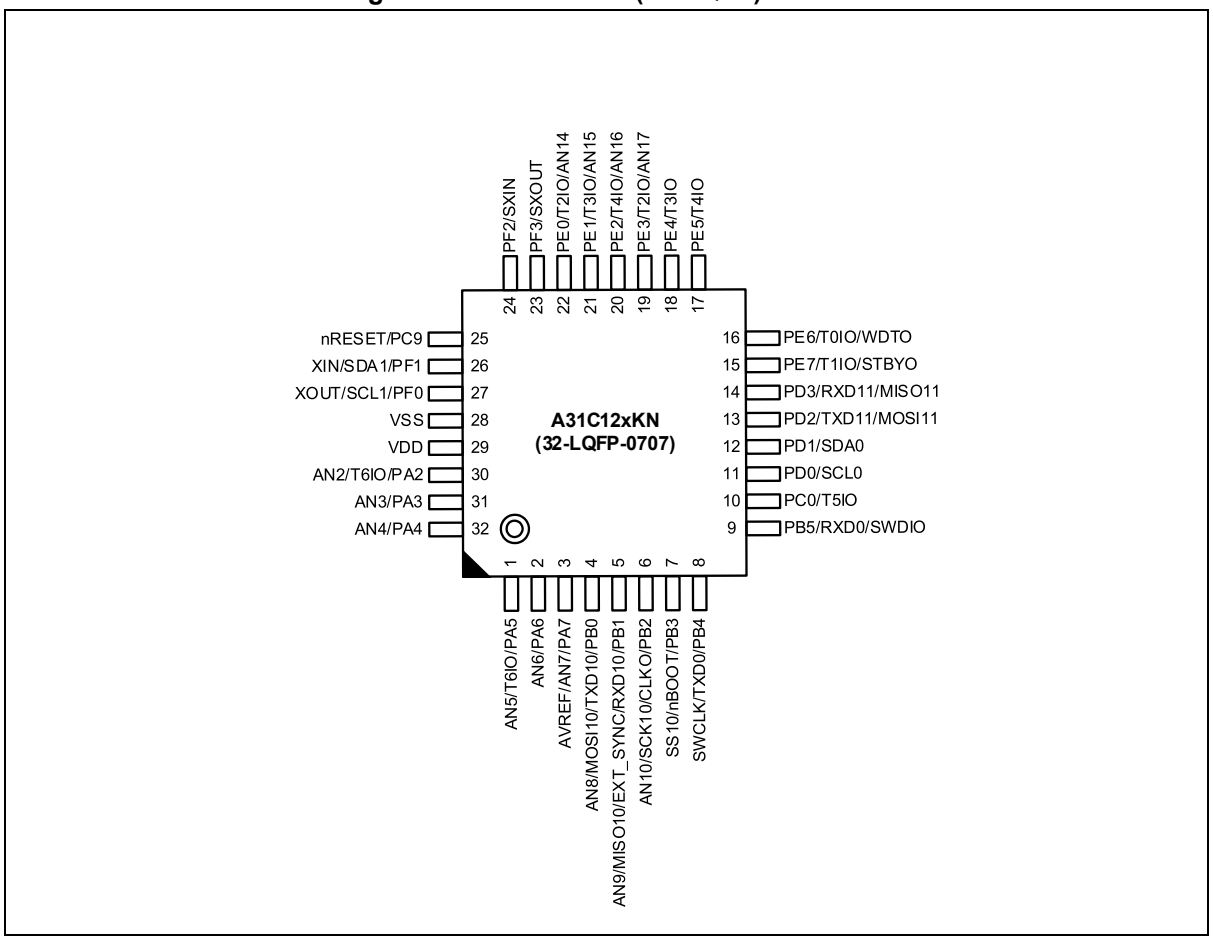
2.1.2.1 A31C12xKY (32-QFN)

Figure 12. A31C12xKY (32-QFN) Pinouts



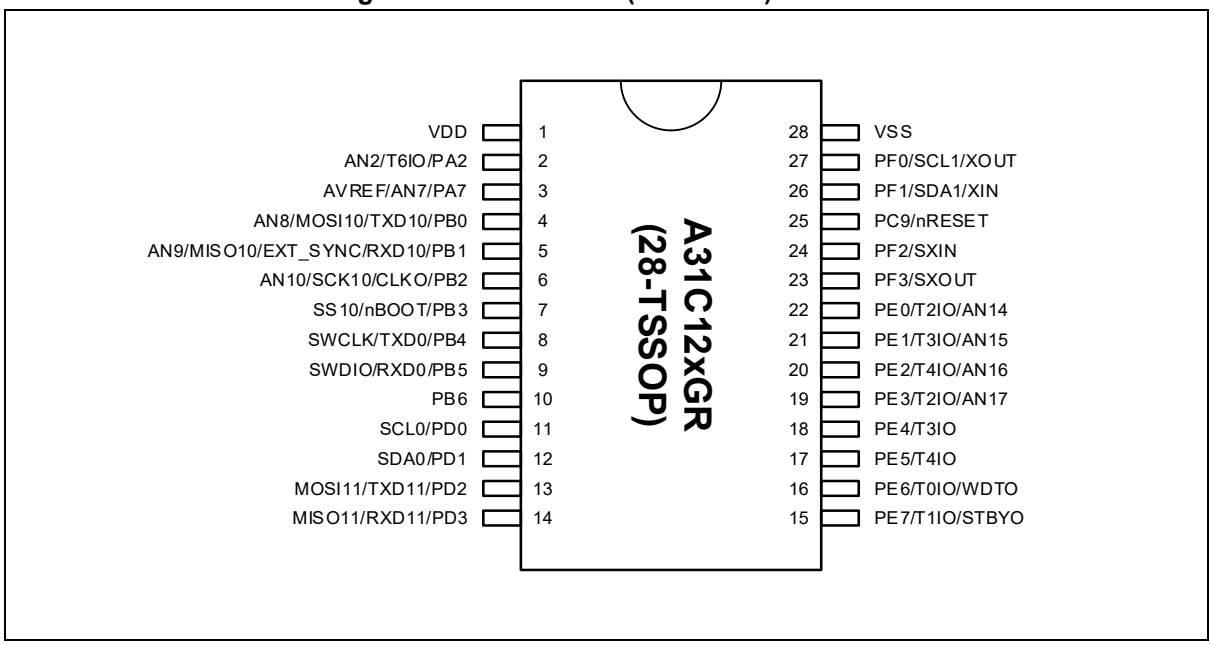
2.1.2.2 A31C12xKN (32-LQFP)

Figure 13. A31C12xKN (32-LQFP) Pinouts



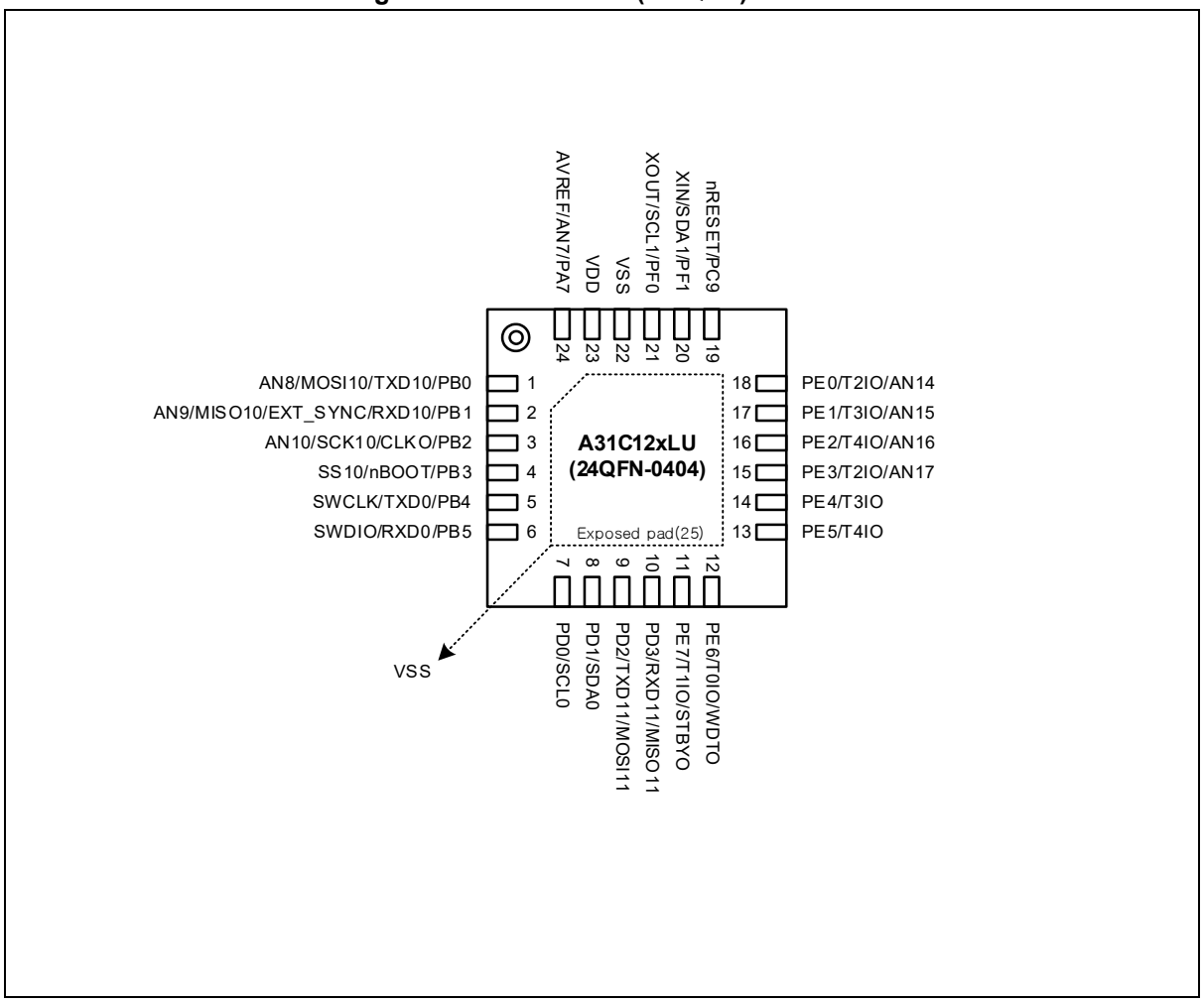
2.1.2.3 A31C12xGR (28-TSSOP)

Figure 14. A31C12xGR (28-TSSOP) Pinouts



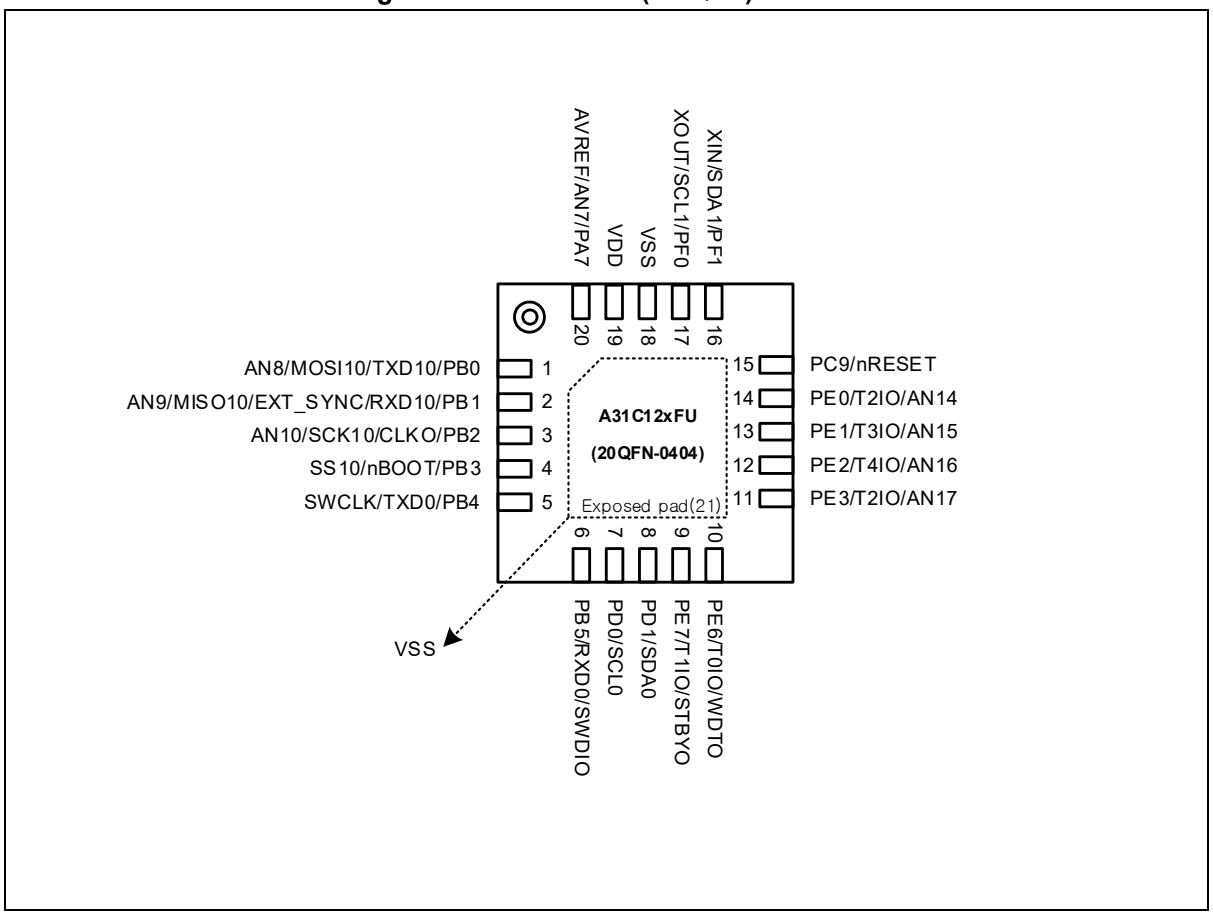
2.1.2.4 A31C12xLU (24-QFN)

Figure 15. A31C12xLU (20-QFN) Pinouts



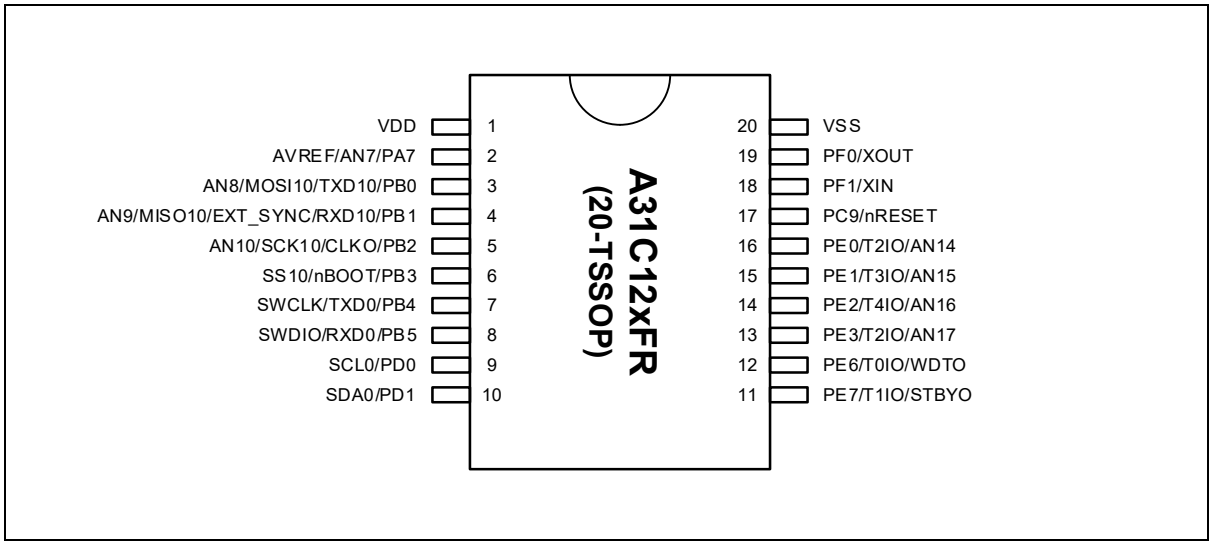
2.1.2.5 A31C12xFU (20-QFN)

Figure 16. A31C12xFU (20-QFN) Pinouts



2.1.2.6 A31C12xFR (20-TSSOP)

Figure 17. A31C12xFR (20-TSSOP) Pinouts



2.2 Pin Description

Pin configuration information in Table 4 contains two pairs of power ground and other dedicated pins. These multi-function pins have up to six selections of functions including GPIO. Configuration including pin ordering can be changed without notice.

2.2.1 A31C14x

Table 4. Pin Description (A31C14x)

Pin No.							Pin Name	Type	Description	Remark
64-pin	48-pin	44-pin	32-pin	28-pin	20-pin (QFN)	20-pin (TSSOP)				
1	1	44	1	-	-	-	PA5	IOUS	Port A bit 5 input/output	
							T6IO	IO	Timer 6 input/output	
							AN5(2)	IA	Analog Input 5	
							CP1A(2)	IA	COMP 1 Pos Input A	
2	2	1	2	-	-	-	PA6	IOUS	Port A bit 6 input/output	
							AN6(2)	IA	Analog input 6	
							CPRE1(2)	IA	Comparator 1 Reference Input	
3	3	2	3	3	20	2	PA7	IOUS	Port A bit 7 input/output	
							AVREF	IA	Analog Reference Input	
							AN7(2)	IA	Analog input 7	
4	4	3	4	4	1	3	PB0	IOUS	Port B bit 0 input/output	
							TXD10	O	UART(USART) CH0 data output	
							MOSI10	IO	SPI(USART) CH0 Master out/Slave input	
							AN8(2)	IA	Analog input 8	
5	5	4	5	5	2	4	PB1	IOUS	Port B bit 1 input/output	
							RXD10	I	UART(USART) CH0 data Input	
							EXT_SYNC	I	HSICMU External Sync Input	
							MISO10	IO	SPI(USART) CH0 Master input/Slave output	
							AN9(2)	IA	Analog input 9	
6	6	5	6	6	3	5	PB2	IOUS	Port B bit 2 input/output	
							CLKO	O	System clock output	
							SCK10	IO	SPI(USART) CH0 clock input/output	
							AN10(2)	IA	Analog input 10	
7	7	6	7	7	4	6	PB3	IOUS	Port B bit 3 input/output	
							nBOOT(2)(3)(5)	I	Boot mode selection input	Pull-Up
							SS10	IO	SPI(USART) CH0 slave select input	
8	8	7	8	8	5	7	PB4	IOUS	Port B bit 4 input/output	
							TXD0	O	UART CH0 TXD output	
							SWCLK(2)(3)(6)(7)	I	SWD clock input	Pull-Down

Table 4. Pin Description (A31C14x) (continued)

Pin No.							Pin Name	Type	Description	Remark
64-pin	48-pin	44-pin	32-pin	28-pin	20-pin (QFN)	20-pin (TSSOP)				
9	9	8	9	9	6	8	PB5	IOUS	Port B bit 5 input/output	
							RXD0	I	UART CH0 RXD input	
							SWDIO(2)(3)(6)(7)	IO	SWD data input/output	Pull-Up
10	10	9	-	10	-	-	PB6	IOUS	Port B bit 6 input/output	
							TXD1	O	UART CH1 data output	
11	11	10	-	-	-	-	PB7	IOUS	Port B bit 7 input/output	
							RXD1	I	UART CH1 data input	
12	-	-	-	-	-	-	PB8	IOUS	Port B bit 8 input/output	
							T9IO	IO	Timer 9 input/output	
13	-	-	-	-	-	-	PB9	IOUS	Port B bit 9 input/output	
							T8IO	IO	Timer 8 input/output	
14	-	-	-	-	-	-	PB10	IOUS	Port B bit 10 input/output	
							T9IO	IO	Timer 9 input/output	
15	-	-	-	-	-	-	PB11	IOUS	Port B bit 11 input/output	
							T8IO	IO	Timer 8 input/output	
16	12	11	10	-	-	-	PC0	IOUS	Port C bit 0 input/output	
							T5IO	IO	Timer 5 input/output	
17	13	12	-	-	-	-	PC1	IOUS	Port C bit 1 input/output	
							T7IO	IO	Timer 7 input/output	
18	14	13	-	-	-	-	PC2	IOUS	Port C bit 0 input/output	
							T5IO	IO	Timer 5 input/output	
19	15	14	-	-	-	-	PC3	IOUS	Port C bit 3 input/output	
							T7IO	IO	Timer 7 input/output	
20	16	15	-	-	-	-	PC4	IOUS	Port C bit 4 input/output	
21	-	-	-	-	-	-	PC5	IOUS	Port C bit 5 input/output	
							SDA2	IO	I2C CH2 SDA input/output	Open-Drain
22	-	-	-	-	-	-	PC6	IOUS	Port C bit 6 input/output	
							SCL2	IO	I2C CH2 SCL input/output	Open-Drain
23	-	-	-	-	-	-	PC7	IOUS	Port C bit 7 input/output	
24	-	-	-	-	-	-	PC8	IOUS	Port C bit 8 input/output	
25	17	16	11	11	7	9	PD0	IOUS	Port D bit 0 input/output	
							SCL0	IO	I2C CH0 SCL input/output	Open-Drain
26	18	17	12	12	8	10	PD1	IOUS	Port D bit 1 input/output	
							SDA0	IO	I2C CH0 SDA input/output	Open-Drain
27	19	18	13	13	-	-	PD2	IOUS	Port D bit 2 input/output	
							TXD11	O	UART(USART) CH1 data output	
							MOSI11	IO	SPI(USART) CH1 Master out/Slave input	
28	20	19	14	14	-	-	PD3	IOUS	Port D bit 3 input/output	
							RXD11	I	UART(USART) CH1 data input	
							MISO11	IO	SPI(USART) CH1 Master input/Slave output	

Table 4. Pin Description (A31C14x) (continued)

Pin No.							Pin Name	Type	Description	Remark
64-pin	48-pin	44-pin	32-pin	28-pin	20-pin (QFN)	20-pin (TSSOP)				
29	21	20	-	-	-	-	PD4	IOUS	Port D bit 4 input/output	
							SCK11	IO	SPI(USART) CH1 clock input/output	
30	22	21	-	-	-	-	PD5	IOUS	Port D bit 5 input/output	
							SS11	IO	SPI(USART) CH1 slave select input	
31	23	-	-	-	-	-	PD6	IOUS	Port D bit 6 input/output	
							T1IO	IO	Timer 1 input/output	
32	24	-	-	-	-	-	PD7	IOUS	Port D bit 7 input/output	
							T0IO	IO	Timer 0 input/output	
33	25	22	15	15	9	11	PE7	IOUS	Port E bit 7 input/output	
							T1IO	IO	Timer 1 input/output	
							STBYO	O	Deep Sleep mode output	
34	26	23	16	16	10	12	PE6	IOUS	Port E bit 6 input/output	
							T0IO	IO	Timer 0 input/output	
							WDTO	O	Watchdog timer overflow output	
35	27	24	17	17	-	-	PE5	IOUS	Port E bit 5 input/output	
							T4IO	IO	Timer 4 input/output	
36	28	25	18	18	-	-	PE4	IOUS	Port E bit 4 input/output	
							T3IO	IO	Timer 3 input/output	
37	29	26	19	19	11	13	PE3	IOUS	Port E bit 3 input/output	
							T2IO	IO	Timer 2 input/output	
							AN17[2]	IA	Analog input 17	
38	30	27	20	20	12	14	PE2	IOUS	Port E bit 2 input/output	
							T4IO	IO	Timer 4 input/output	
							AN16[2]	IA	Analog input 16	
39	31	28	21	21	13	15	PE1	IOUS	Port E bit 1 input/output	
							T3IO	IO	Timer 3 input/output	
							CP1OUT	O	Comparator 0 output	
							AN15[2]	IA	Analog input 15	
40	32	29	22	22	14	16	PE0	IOUS	Port E bit 0 input/output	
							T2IO	IO	Timer 2 input/output	
							CP0OUT	O	Comparator 0 output	
							AN14[2]	IA	Analog input 14	
41	-	-	-	-	-	-	PE11	IOUS	Port E bit 11 input/output	
							SS12	IO	SPI(USART) CH2 slave select input	
42	-	-	-	-	-	-	PE10	IOUS	Port E bit 10 input/output	
							SCK12	IO	SPI(USART) CH2 clock input/output	
43	-	-	-	-	-	-	PE9	IOUS	Port E bit 9 input/output	
							RXD12	I	UART(USART) CH2 data input	
							MISO12	IO	SPI(USART) CH2 Master input/Slave output	

Table 4. Pin Description (A31C14x) (continued)

Pin No.							Pin Name	Type	Description	Remark
64-pin	48-pin	44-pin	32-pin	28-pin	20-pin (QFN)	20-pin (TSSOP)				
44	-	-	-	-	-	-	PE8	IOUS	Port E bit 8 input/output	
							TXD12	O	UART(USART) CH2 data output	
							MOSI12	IO	SPI(USART) CH2 Master out/Slave input	
45	33	30	-	-	-	-	PF7	IOUS	Port F bit 7 input/output	
							T4IO	IO	Timer 4 input/output	
46	34	31	-	-	-	-	PF6	IOUS	Port F bit 6 input/output	
							T3IO	IO	Timer 3 input/output	
47	35	32	-	-	-	-	PF5	IOUS	Port F bit 5 input/output	
							T2IO	IO	Timer 2 input/output	
48	36	33	-	-	-	-	PF4	IOUS	Port F bit 4 input/output	
							CLKO	O	System clock output	
49	37	34	23	23	-	-	PF3	IOUS	Port F bit 3 input/output	
							SXOUT[2][8]	OA	Sub crystal oscillator output	
50	38	35	24	24	-	-	PF2	IOUS	Port F bit 2 input/output	
							SXIN[2][8]	IA	Sub crystal oscillator input	
51	39	36	25	25	15	17	PC9	IOUS	Port C bit 9 input/output	
							nRESET[2][3]	I	External reset input	Pull-Up
52	40	37	26	26	16	18	PF1	IOUS	Port F bit 1 input/output	
							SDA1	IO	I2C CH1 SDA input/output	Open-Drain
							XIN[2][8]	IA	External crystal oscillator input	
53	41	38	27	27	17	19	PF0	IOUS	Port F bit 0 input/output	
							SCL1	IO	I2C CH1 SCL input/output	Open-Drain
							XOUT[2][8]	IA	External crystal oscillator output	
54	42	39	28	28	18	20	VSS	P	Ground	
55	43	40	29	1	19	1	VDD	P	VDD	
56	-	-	-	-	-	-	PA8	IOUS	Port A bit 8 input/output	
							AN11[2]	IA	Analog input 11	
57	-	-	-	-	-	-	PA9	IOUS	Port A bit 9 input/output	
							AN12[2]	IA	Analog input 12	
58	-	-	-	-	-	-	PA10	IOUS	Port A bit 10 input/output	
							AN13[2]	IA	Analog input 13	
59	-	-	-	-	-	-	PA11	IOUS	Port A bit 11 input/output	
60	44	-	-	-	-	-	PA0	IOUS	Port A bit 0 input/output	
							SDA1	IO	I2C CH1 SDA input/output	Open-Drain
							AN0[2]	IA	Analog input 0	
61	45	-	-	-	-	-	PA1	IOUS	Port A bit 1 input/output	
							SCL1	IO	I2C CH1 SCL input/output	Open-Drain
							AN1[2]	IA	Analog input 1	
62	46	41	30	2	-	-	PA2	IOUS	Port A bit 2 input/output	
							T6IO	IO	Timer 2 input/output	
							AN2[2]	IA	Analog input 2	
							CPRE0[2]	IA	Comparator 0 reference	

Table 4. Pin Description (A31C14x) (continued)

Pin No.							Pin Name	Type	Description	Remark
64-pin	48-pin	44-pin	32-pin	28-pin	20-pin (QFN)	20-pin (TSSOP)				
63	47	42	31	-	-	-	PA3	IOUS	Port A bit 3 input/output	
							AN3(2)	IA	Analog input 3	
							CP0A(2)	IA	Comparator 0 plus input A	
64	48	43	32	-	-	-	PA4	IOUS	Port A bit 4 input/output	
							AN4(2)	IA	Analog input 4	
							CPB(2)	IA	Comparator 0/1 plus input B	

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.
3. PB3 (nBOOT), PC9 (nRESET), PB5 (SWDIO) and the default pull-up pins and PB4 (SWCLK) are default pull-down pin.
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 Data I/O (SWDIO) is enabled and the internal pull-down for the Serial Wire Clock (SWCLK) 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 PF1 (XIN), PF0 (XOUT), PF3 (SXOUT) and PF2 (SXIN) pins, other functions than the set clock will not operate normally if the clock is enabled by software.

2.2.2 A31C12x

Table 5. Pin Description (A31C12x)

Pin No.					Pin Name	Type	Description	Remark
32-pin	28-pin	24-pin	20-pin (QFN)	20-pin (TSSOP)				
1	-	-	-	-	PA5	IOUS	Port A bit 5 input/output	
					T6IO	IO	Timer 6 input/output	
2	-	-	-	-	PA6	IOUS	Port A bit 6 input/output	
3	3	24	19	2	PA7	IOUS	Port A bit 7 input/output	
					AVREF	IA	Analog Reference Input	
					AN7(2)	IA	Analog input 7	
4	4	1	20	3	PB0	IOUS	Port B bit 0 input/output	
					TXD10	O	UART(USART) CH0 data output	
					MOSI10	IO	SPI(USART) CH0 Master out/Slave input	
					AN8(2)	IA	Analog input 8	
5	5	2	1	4	PB1	IOUS	Port B bit 1 input/output	
					RXD10	I	UART(USART) CH0 data Input	
					EXT_SYNC	I	HSICMU External Sync Input	
					MISO10	IO	SPI(USART) CH0 Master input/Slave output	
					AN9(2)	IA	Analog input 9	
6	6	3	2	5	PB2	IOUS	Port B bit 2 input/output	
					CLKO	O	System clock output	
					SCK10	IO	SPI(USART) CH0 clock input/output	
					AN10(2)	IA	Analog input 10	
7	7	4	3	6	PB3	IOUS	Port B bit 3 input/output	
					nBOOT(2)(3)(5)	I	Boot mode selection input	Pull-Up
					SS10	IO	SPI(USART) CH0 slave select input	
8	8	5	4	7	PB4	IOUS	Port B bit 4 input/output	
					TXD0	O	UART CH0 TXD output	
					SWCLK(2)(3)(6)(7)	I	SWD clock input	Pull-Down
9	9	6	5	8	PB5	IOUS	Port B bit 5 input/output	
					RXD0	I	UART CH0 RXD input	
					SWDIO(2)(3)(6)(7)	IO	SWD data input/output	Pull-Up
-	10	-	-	-	PB6(9)	IOUS	Port B bit 6 input/output	
10	-	-	-	-	PC0(9)	IOUS	Port C bit 0 input/output	
					T5IO	IO	Timer 5 input/output	
11	11	7	6	9	PD0	IOUS	Port D bit 0 input/output	
					SCL0	IO	I2C CH0 SCL input/output	Open-Drain

Table 5. Pin Description (A31C12x) (continued)

Pin No.					Pin Name	Type	Description	Remark
32-pin	28-pin	24-pin	20-pin (QFN)	20-pin (TSSOP)				
12	12	8	7	10	PD1	IOUS	Port D bit 1 input/output	
					SDA0	IO	I2C CH0 SDA input/output	Open-Drain
13	13	9	-	-	PD2	IOUS	Port D bit 2 input/output	
					TXD11	O	UART(USART) CH1 data output	
					MOSI11	IO	SPI(USART) CH1 Master out/Slave input	
14	14	10	-	-	PD3	IOUS	Port D bit 3 input/output	
					RXD11	I	UART(USART) CH1 data input	
					MISO11	IO	SPI(USART) CH1 Master input/Slave output	
15	15	11	8	11	PE7	IOUS	Port E bit 7 input/output	
					T1IO	IO	Timer 1 input/output	
					STBYO	O	Deep Sleep mode output	
16	16	12	9	12	PE6	IOUS	Port E bit 6 input/output	
					T0IO	IO	Timer 0 input/output	
					WDTO	O	Watchdog timer overflow output	
17	17	13	-	-	PE5	IOUS	Port E bit 5 input/output	
					T4IO	IO	Timer 4 input/output	
18	18	14	-	-	PE4	IOUS	Port E bit 4 input/output	
					T3IO	IO	Timer 3 input/output	
19	19	15	10	13	PE3	IOUS	Port E bit 3 input/output	
					T2IO	IO	Timer 2 input/output	
					AN17[(2)]	IA	Analog input 17	
20	20	16	11	14	PE2	IOUS	Port E bit 2 input/output	
					T4IO	IO	Timer 4 input/output	
					AN16[(2)]	IA	Analog input 16	
21	21	17	12	15	PE1	IOUS	Port E bit 1 input/output	
					T3IO	IO	Timer 3 input/output	
					AN15[(2)]	IA	Analog input 15	
22	22	18	13	16	PE0	IOUS	Port E bit 0 input/output	
					T2IO	IO	Timer 2 input/output	
					AN14[(2)]	IA	Analog input 14	
23	23	-	-	-	PF3	IOUS	Port F bit 3 input/output	
					SXOUT[(2)(8)]	OA	Sub crystal oscillator output	
24	24	-	-	-	PF2	IOUS	Port F bit 2 input/output	
					SXIN[(2)(8)]	IA	Sub crystal oscillator input	
25	25	19	14	17	PC9	IOUS	Port C bit 9 input/output	
					nRESET[(2)(3)]	I	External reset input	Pull-Up
26	26	20	15	18	PF1	IOUS	Port F bit 1 input/output	
					SDA1	IO	I2C CH1 SDA input/output	Open-Drain
					XIN[(2)(8)]	IA	External crystal oscillator input	

Table 5. Pin Description (A31C12x) (continued)

Pin No.					Pin Name	Type	Description	Remark
32-pin	28-pin	24-pin	20-pin (QFN)	20-pin (TSSOP)				
27	27	21	16	19	PF0	IOUS	Port F bit 0 input/output	
					SCL1	IO	I2C CH1 SCL input/output	Open-Drain
					XOUT(2)(8)	IA	External crystal oscillator output	
28	28	22	17	20	VSS	P	Ground	
29	1	23	18	1	VDD	P	VDD	
30	2	-	-	-	PA2	IOUS	Port A bit 2 input/output	
					T6IO	IO	Timer 2 input/output	
					AN2(2)	IA	Analog input 2	
31	-	-	-	-	PA3	IOUS	Port A bit 3 input/output	
					AN3(2)	IA	Analog input 3	
32	-	-	-	-	PA4	IOUS	Port A bit 4 input/output	
					AN4(2)	IA	Analog input 4	

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 32-pin standard.
3. PB3 (nBOOT), PC9 (nRESET), PB5 (SWDIO) and the default pull-up pins and PB4 (SWCLK) are default pull-down pin.
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 Data I/O (SWDIO) is enabled and the internal pull-down for the Serial Wire Clock (SWCLK) 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 PF1 (XIN), PF0 (XOUT), PF3 (SXOUT) and PF2 (SXIN) pins, other functions than the set clock will not operate normally if the clock is enabled by software.
9. When using the 32-QFN or 28-TSSOP package of the A31C12x, the port configuration can be selected according to the value of the PIN_REMAP[0] bit in the PCU_PIN_REMAP register. Refer to Section 5.4.14 for more details.

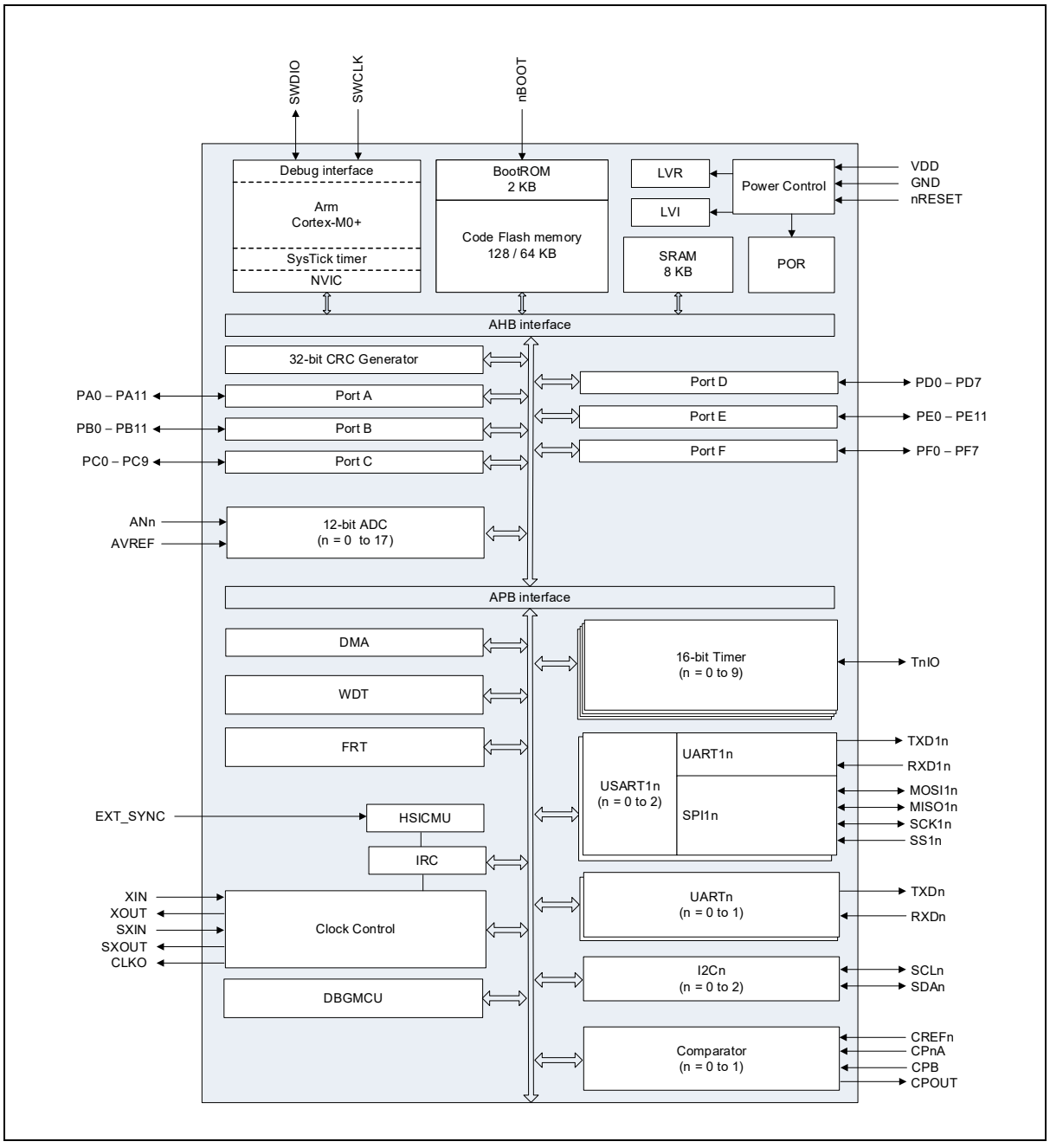
3. System and Memory Overview

3.1 System Architecture

3.1.1 System Block Diagram

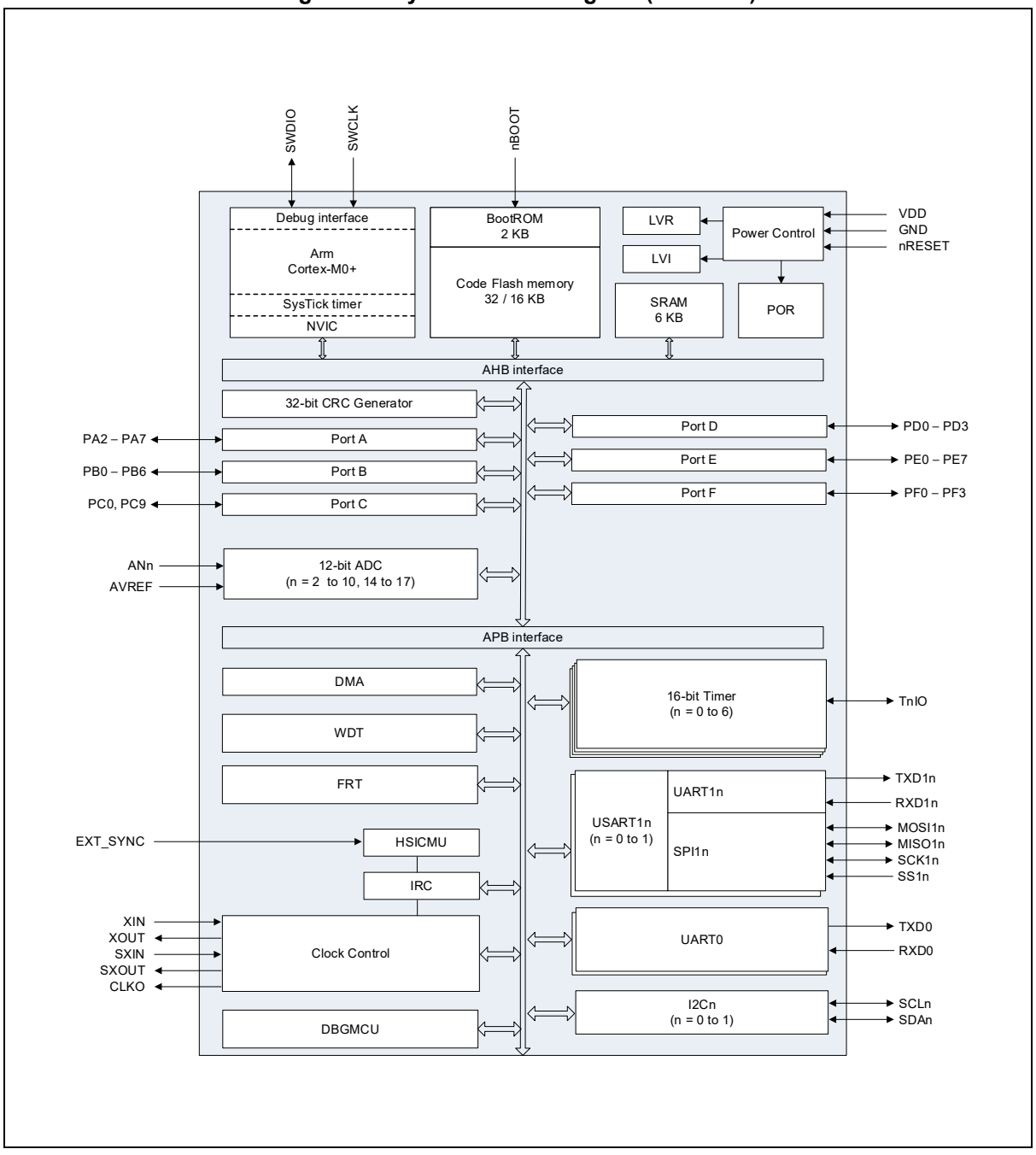
3.1.1.1 A31C14x

Figure 18. System Block Diagram (A31C14x)



3.1.1.2 A31C12x

Figure 19. System Block Diagram (A31C12x)



3.1.2 Cortex-M0+ Core

The A31C14x and A31C12x series adopts the energy-efficient 32-bit microcontroller core, Cortex-M0+ of Arm as its CPU core. The Cortex-M0+ designed for low-power consumption uses the ARMv6-M instruction set to improve performance and energy efficiency.

Refer to the technical reference manual “ARM DDI 0484C” for detailed information of Cortex-M0+.

3.1.3 Interrupt Controller

3.1.3.1 A31C14x

Table 6. Interrupt Vector Map (A31C14x) (1)

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	WDT
5	0x0000_0054	FRT
6	0x0000_0058	CFMC
7	0x0000_005C	TIMER0
8	0x0000_0060	TIMER1
9	0x0000_0064	TIMER2
10	0x0000_0068	TIMER3
11	0x0000_006C	GPIOA, GPIOB

Table 6. Interrupt Vector Map (A31C14x) (1) (continued)

Priority	Vector Address	Interrupt Source
12	0x0000_0070	GPIOC, GPIOD
13	0x0000_0074	GPIOE, GPIOF
14	0x0000_0078	TIMER4
15	0x0000_007C	TIMER5
16	0x0000_0080	USART10
17	0x0000_0084	USART11
18	0x0000_0088	USART12
19	0x0000_008C	I2C0
20	0x0000_0090	UART0
21	0x0000_0094	UART1
22	0x0000_0098	ADC
23	0x0000_009C	CRC
24	0x0000_00A0	CMP0
25	0x0000_00A4	CMP1
26	0x0000_00A8	I2C1
27	0x0000_00AC	I2C2
28	0x0000_00B0	TIMER6
29	0x0000_00B4	TIMER7
30	0x0000_00B8	TIMER8
31	0x0000_00BC	TIMER9

NOTE:

1. Each interrupt is linked to the priority level registers. Each interrupt priority register is 1-byte length and only the upper 2 bits of it are used for the priority level. NVIC registers of Cortex-M0+ processor use four interrupt priority bytes at a time because it supports the transmission in word unit.

   ** __NVIC_PRIO_BITS = 2

3.1.3.2 A31C12x

Table 7. Interrupt Vector Map (A31C12x) (1)

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	WDT
5	0x0000_0054	FRT
6	0x0000_0058	CFMC
7	0x0000_005C	TIMER0
8	0x0000_0060	TIMER1
9	0x0000_0064	TIMER2
10	0x0000_0068	TIMER3
11	0x0000_006C	GPIOA, GPIOB

Table 7. Interrupt Vector Map (A31C12x) [1] (continued)

Priority	Vector Address	Interrupt Source
12	0x0000_0070	GPIOC, GPIOD
13	0x0000_0074	GPIOE, GPIOF
14	0x0000_0078	TIMER4
15	0x0000_007C	TIMER5
16	0x0000_0080	USART10
17	0x0000_0084	USART11
18	0x0000_0088	Reserved
19	0x0000_008C	I2C0
20	0x0000_0090	UART0
21	0x0000_0094	Reserved
22	0x0000_0098	ADC
23	0x0000_009C	CRC
24	0x0000_00A0	Reserved
25	0x0000_00A4	Reserved
26	0x0000_00A8	I2C1
27	0x0000_00AC	Reserved
28	0x0000_00B0	TIMER6
29	0x0000_00B4	Reserved
30	0x0000_00B8	Reserved
31	0x0000_00BC	Reserved

NOTE:

1. Each interrupt is linked to the priority level registers. Each interrupt priority register is 1-byte length and only the upper 2 bits of it are used for the priority level. NVIC registers of Cortex-M0+ processor use four interrupt priority bytes at a time because it supports the transmission in word unit.

   ** __NVIC_PRIO_BITS = 2

3.2 Memory Organization

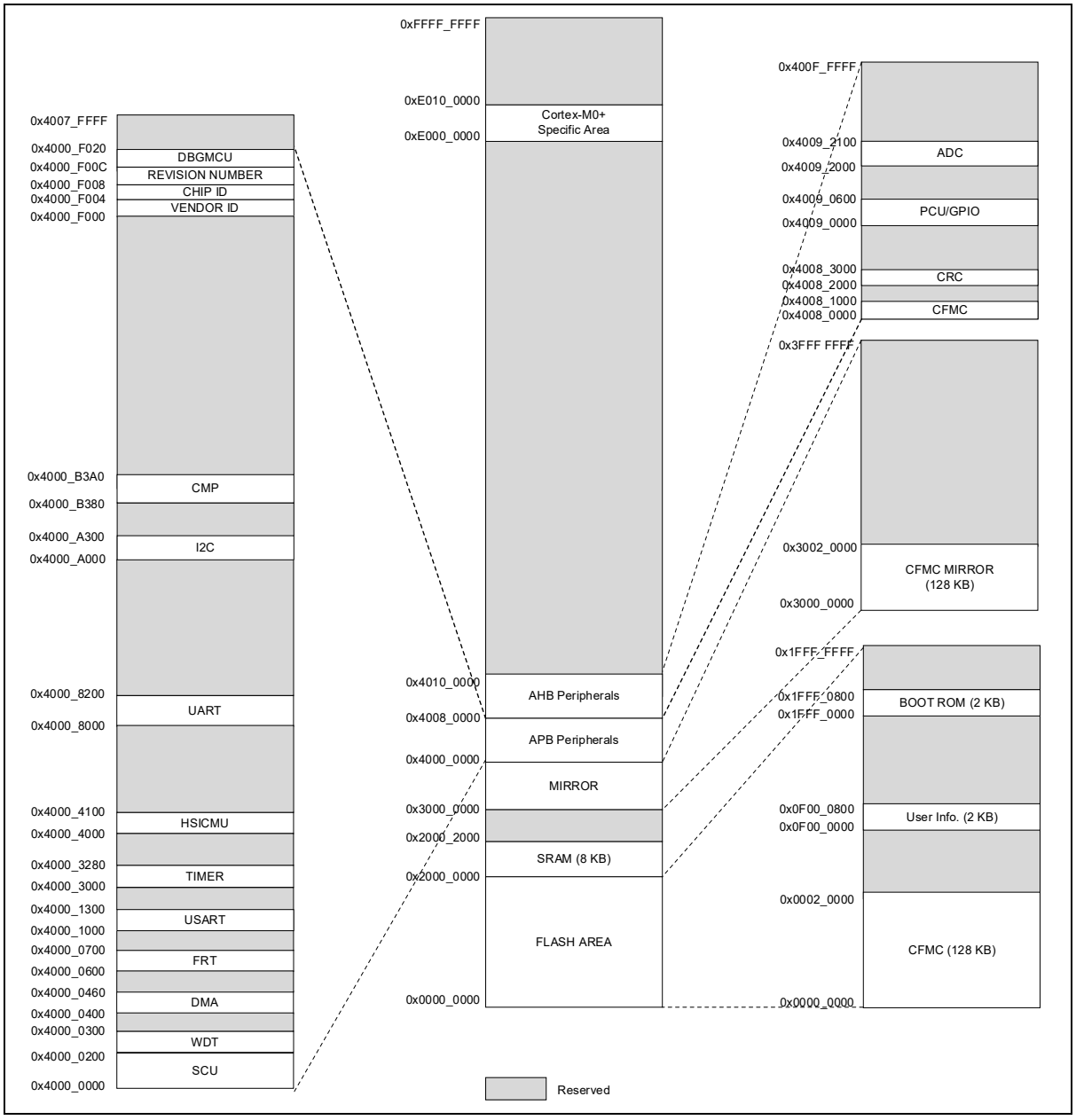
3.2.1 Introduction

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

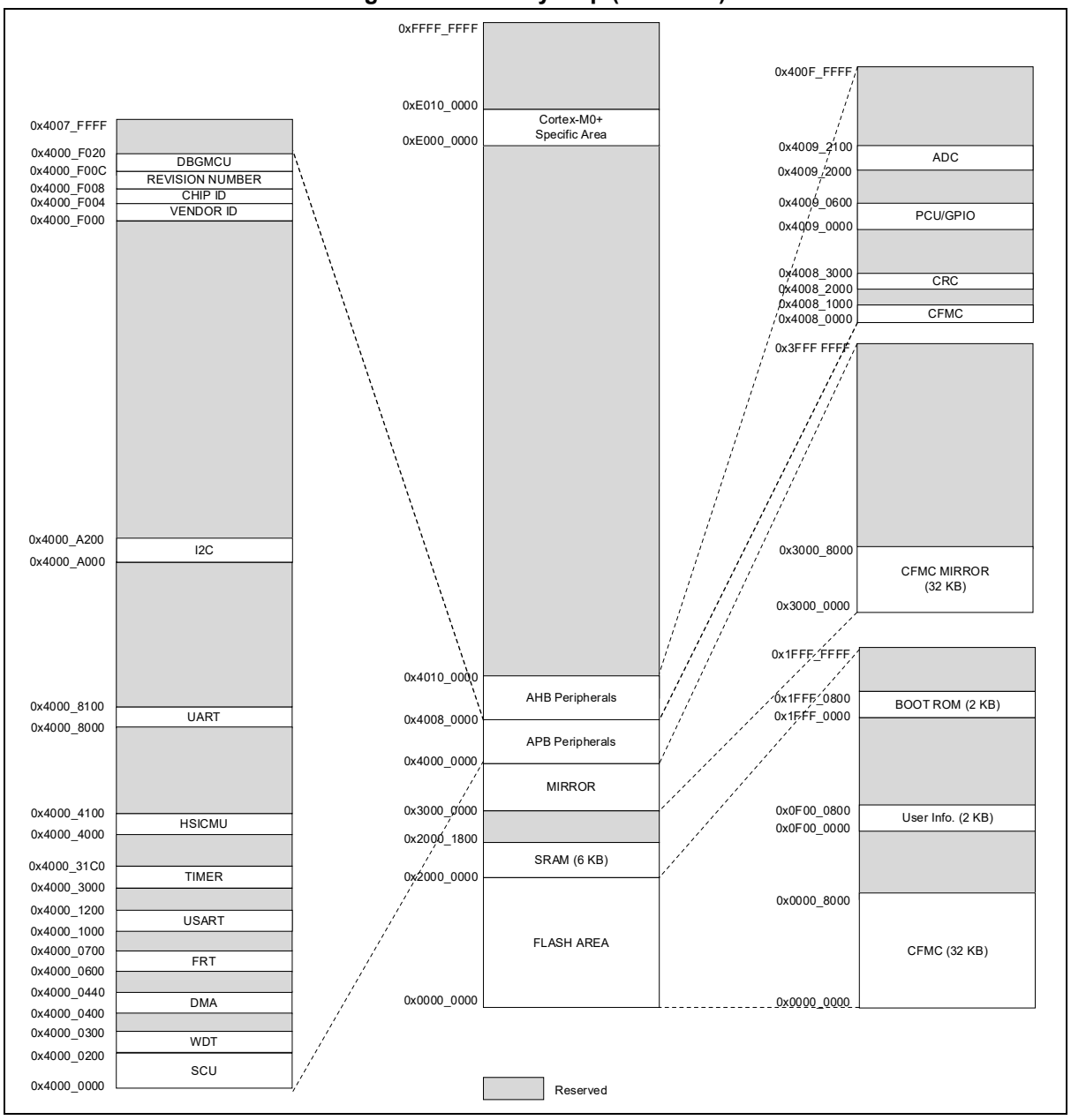
3.2.2.1 A31C14x

Figure 20. Memory Map (A31C14x)



3.2.2.2 A31C12x

Figure 21. Memory Map (A31C12x)



3.3 Memory Map

3.3.1 Embedded SRAM

The A31C14x and A31C12x series has a block of zero-wait on-chip SRAM. The SRAM size of A31C14x is 8 KB, A31C12x is 6 KB. The 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

A31C14x and A31C12x series each provides internal code flash memory and a flash controller. A31C14x includes 128 KB or 64 KB, while A31C12x includes 32 KB or 16 KB of code flash memory. 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 A31C14x and A31C12x series offers a boot mode for programming the 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 (TXD10/RXD10).

The pins for the boot mode are listed in Table 8.

Table 8. Boot Mode Pin List

Block	Pin Name	I/O	Pin Direction	Description
SYSTEM	nRESET	PC9	Input	Receive the reset signal
	nBOOT	PB3	Input	Set the boot mode
UART mode (USART10)	RXD10	PB1	Input	Receive the UART boot data
	TXD10	PB0	Output	Transmit the UART boot data

4. SCU

4.1 SCU Introduction

The A31C14x and A31C12x series 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 are as follows:

- Operating frequency: Up to 32 MHz
- High-speed internal oscillator (HSI): 32 MHz
- Low-speed internal oscillator (LSI): 32 kHz
- High-speed External Oscillator (HSE): 4 MHz to 12 MHz
- Low-speed External Oscillator (LSE): 32.768 kHz
- A31C14x and A31C12x series has clock monitoring as a system fail-safe function.

The reset features 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 modes are as follows:

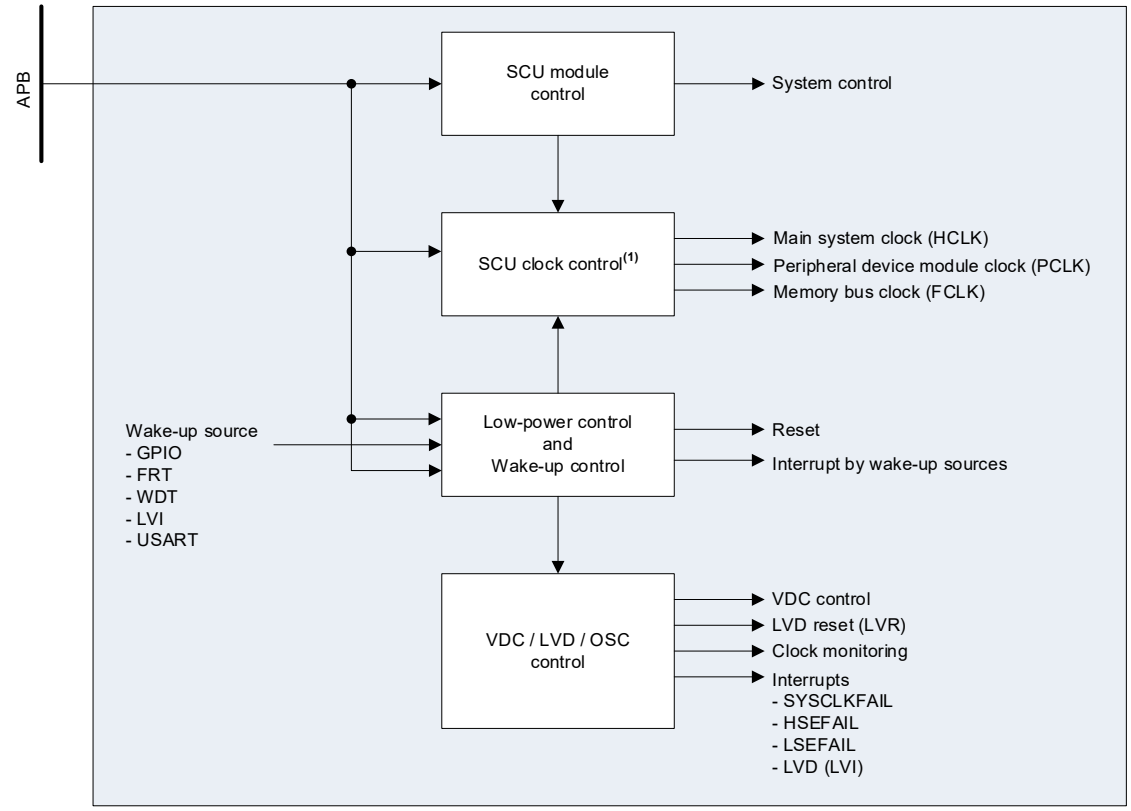
- RUN mode
- SLEEP mode
- DEEP-SLEEP (STOP1, STOP2) mode

4.3 SCU Functional Description

4.3.1 SCU Block Diagram

Figure 22 describes SCU in a block diagram.

Figure 22. SCU Block Diagram



NOTE:

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

4.3.2 SCU Pin and Type

Six pins in Table 9 are assigned for the SCU module.

Table 9. SCU Pins

Pin Name	Type	Description
nRESET	Input	External reset input
XIN, XOUT	OSC	External crystal oscillator (4 MHz to 12 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 A31C14x and A31C12x series has a clock generation circuit. The clock generation circuit consists of external oscillation circuit, and an internal oscillation circuit. The overview of each circuit is as follows:

- External oscillation circuit
  - High-Speed External Oscillator (HSE): 4 MHz to 12 MHz (It is recommended to use 8 MHz)
    - Generates main clocks by connecting oscillator to XIN pin and XOUT pin.
  - Low-Speed External Oscillator (LSE): 32.768 kHz
    - Generates sub clocks by connecting oscillator to SXIN pin and SXOUT pin.
- Internal oscillation circuit
  - High-Speed internal oscillator (HSI): 32 MHz(1)
  - Low-Speed internal oscillator (LSI): 32 kHz(1)

NOTE:

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

Four 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)

Table 10 describes the clock sources of the A31C14x and A31C12x series.

Table 10. Clock Sources

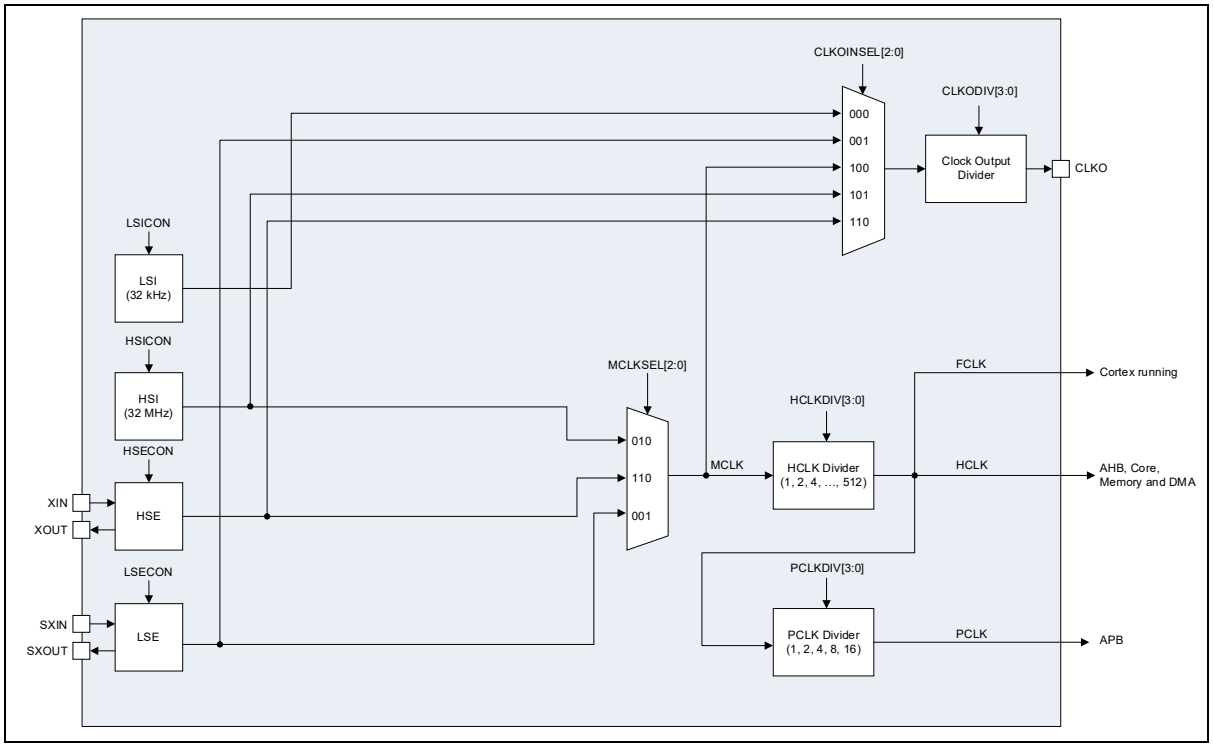
Clock Name	Frequency	Description
HSE	4 to 12 MHz	High-speed external oscillator
LSE	32.768 kHz	Low-speed external oscillator
HSI	32 MHz	High-speed internal oscillator
LSI	32 kHz	Low-speed internal oscillator

4.4.2 Clock Tree Configuration

Users can maintain the clock system variation under software control. Figure 23 allows the users to learn about the clock system on the A31C14x and A31C12x series.

The A31C4x and A31C12x series 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 23. Clock Configuration



Each clock multiplexer (selecting a clock source) has a glitch free circuit. So, a clock can be switched 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 11 are assigned for the Clock pins.

Table 11. Clock Generation Circuit Input/Output Pins

Pin Name	I/O	Description
XIN	Input	External crystal oscillator (4 MHz to 12 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:

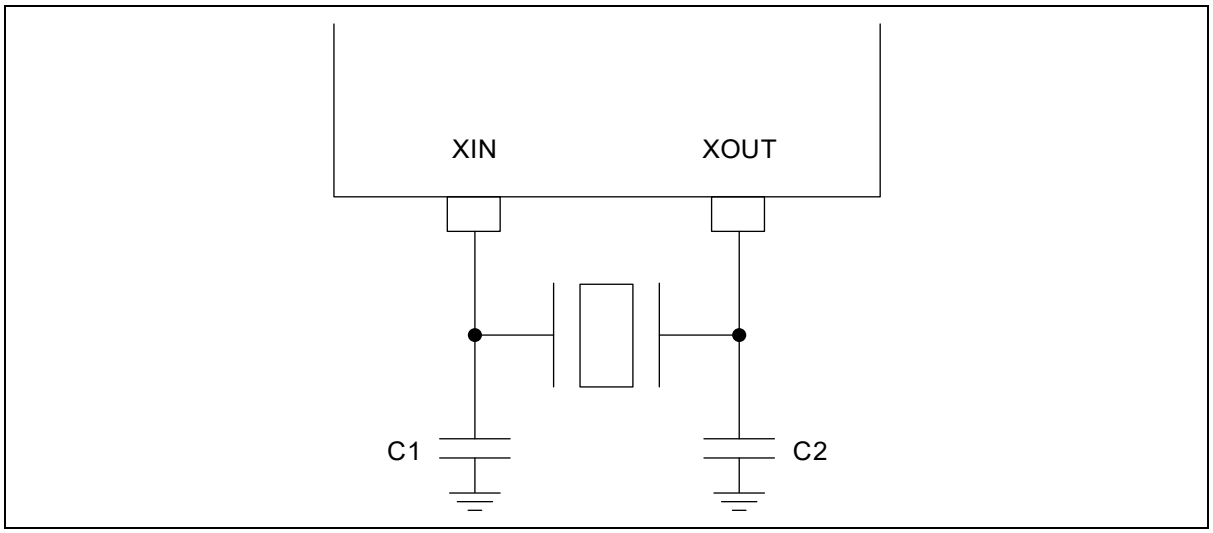
Two pins in Table 12 are assigned for the HSE clock pins block.

- HSE external crystal/ceramic oscillator

Table 12. HSE Clock Pins

Pin Name	I/O	Description
XIN	Input	4 MHz to 12 MHz These pins are used to connect a crystal/ceramic oscillator. For details, refer to Figure 24.
XOUT	Output	

Figure 24. HSE Crystal/Ceramic Oscillator



NOTE:

1. When not using HSE, 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, 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
  - A31C14x
    - ±1.5% @ 0°C to +60°C
    - ±3.0% @ −40°C to +85°C
    - ±4.0% @ −40°C to +105°C
  - A31C12x
    - ±1.5% @ −20°C to +85°C
    - ±1.0% @ 0°C to +65°C

4.4.5 LSE Clock

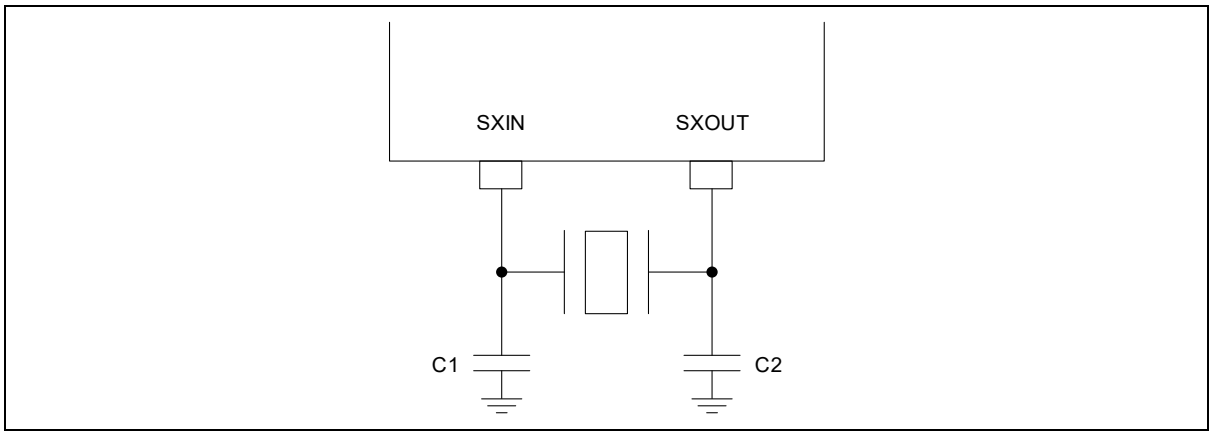
The Low-Speed External Oscillator (LSE) crystal is a 32.768 kHz crystal or ceramic resonator. It has the advantage of providing low power but highly accurate clock source to the Clock peripheral for clock/calendar or other timing functions.

- LSE: 32.768 kHz

Table 13. 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 25. LSE External Clock



NOTE:

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.

4.4.6 LSI Clock

After the microcontroller is powered on, the LSI (32 kHz) is initially enabled by default in the system operation sequence.

For more details, refer to the electrical characteristics chapter in A31C14x and A31C12x series.

- LSI: 32 kHz
  - A31C14x
    - ±10% @ 0°C to +60°C, V_{DD} = 3.0 V
    - ±30% @ −40°C to +85°C, V_{DD} = 3.0 V
  - A31C12x
    - ±30% @ −40°C to +85°C

4.4.7 Two Main Operating Clocks: HCLK and PCLK

The A31C14x and A31C12x series 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.

4.4.7.1 HCLK Clock

The Cortex-M0+ core 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 32 MHz, the HCLK frequency must not exceed 32 MHz.

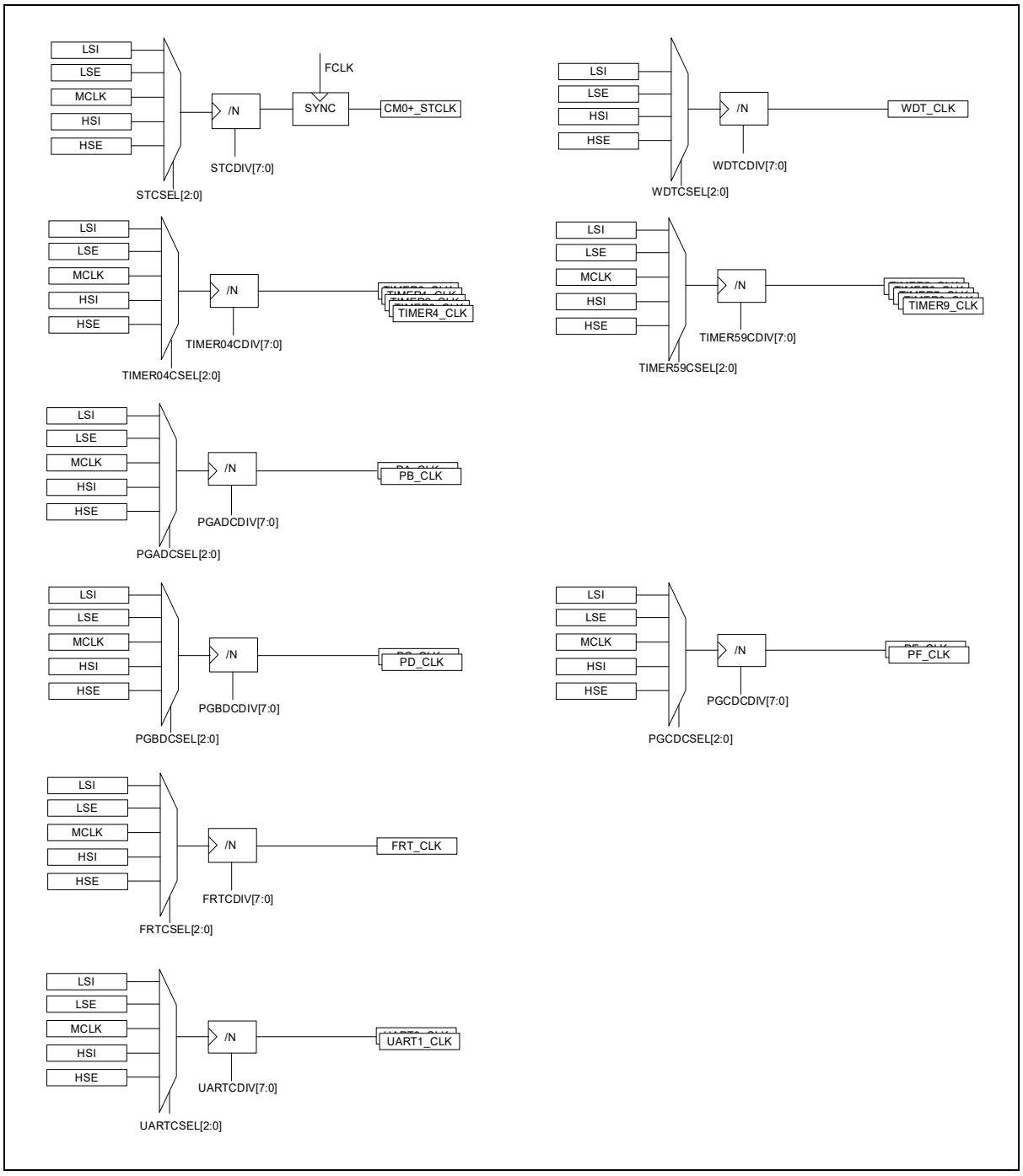
4.4.7.2 PCLK Clock

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.

4.4.8 Configuration of Miscellaneous Clocks

The A31C14x and A31C12x series supports the “miscellaneous clocks” feature, which allows each peripheral to be assigned with a different clock source (MCLK, HSE, LSE, LSI 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 26. Miscellaneous Clock Configuration



4.4.9 Peripheral Clocks

Table 14 describes the peripheral clock and the register selection:

Table 14. Peripheral Clock Selection

Peripheral	PCLK	Miscellaneous Clocks					
		SCU_MCCRx	MCLK	LSI	LSE	HSI	HSE
Systick	O	SCU_MCCR1	O	O	O	O	O
WDT	N/A		N/A	O	O	O	O
TIMERn (n = 0 to 4)	O	SCU_MCCR3	O	O	O	O	O
TIMERn (n = 5 to 9)	O		O	O	O	O	O
PGAD(1)	N/A	SCU_MCCR4	O	O	O	O	O
PGBD(2)	N/A	SCU_MCCR5	O	O	O	O	O
PGCD(3)	N/A		O	O	O	O	O
FRT	N/A	SCU_MCCR6	O	O	O	O	O
UART	N/A	SCU_MCCR7	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

4.4.10 Clock Monitoring

The A31C14x and A31C12x series 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 alternative internal clock source, HSI or reset the system. (For reference, when external crystal such as the HSE and LSE is damaged or an external clock signal is stopped, the A31C14x and A31C12x series generates error flags or interrupt events to check the status and enter recovery mode to put the application in a secure state.)

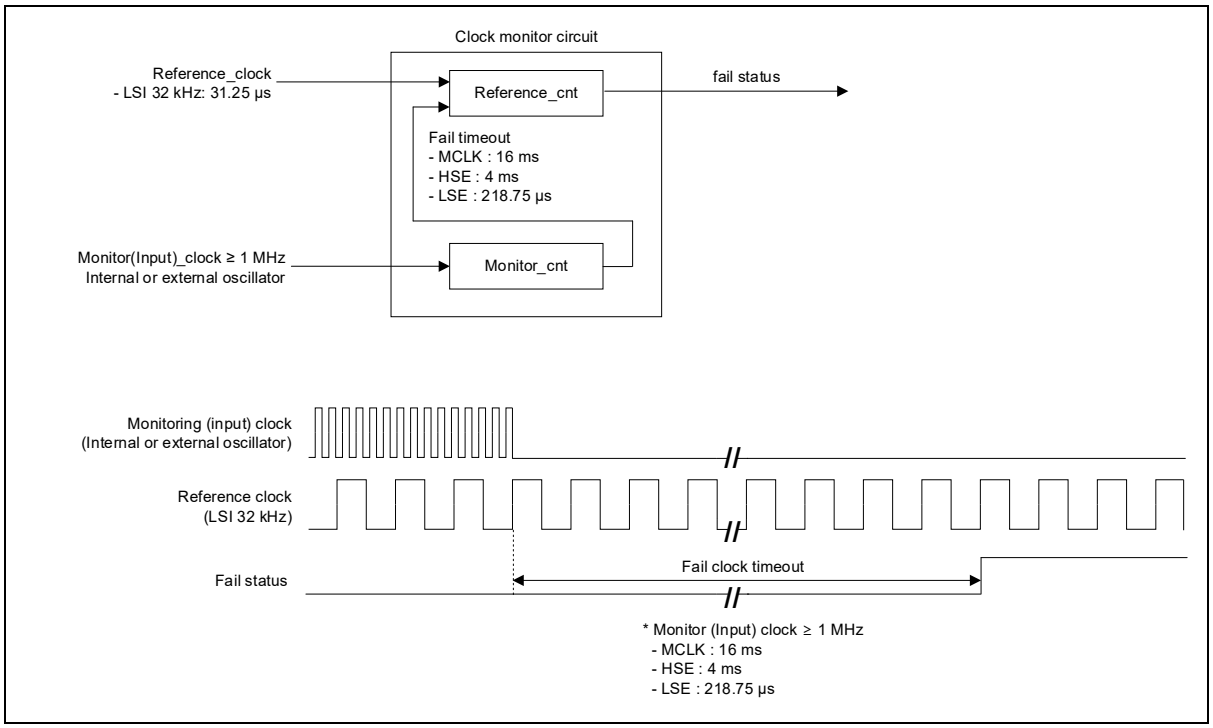
The speed of the internal clock and external clock may be different when using the clock monitoring function, so it is required to adjust the clock speed to meet the requirements for normal operation in the interrupt functions.

4.4.10.1 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 27 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 27. Clock Monitoring Detection Range



4.4.10.2 Clock Monitoring Functions and Clock Sources

Table 15 shows clock monitoring functions provided by the A31C14x and A31C12x series.

Table 15. 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 16 shows clock sources and their frequency characteristics that the A31C14x and A31C12x series can monitor. After each clock source is enabled, stabilization delay is required for stable frequency oscillation.

Table 16. Clock Source for Clock Monitoring

Clock Source	Frequency	Stabilization Delay
HSE	4 to 12 MHz	30 ms
LSE	32.768 kHz	1 s
LSI	32 kHz	600 µs
HSI	32 MHz	30 µs

4.4.10.3 MCLK Monitoring

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

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

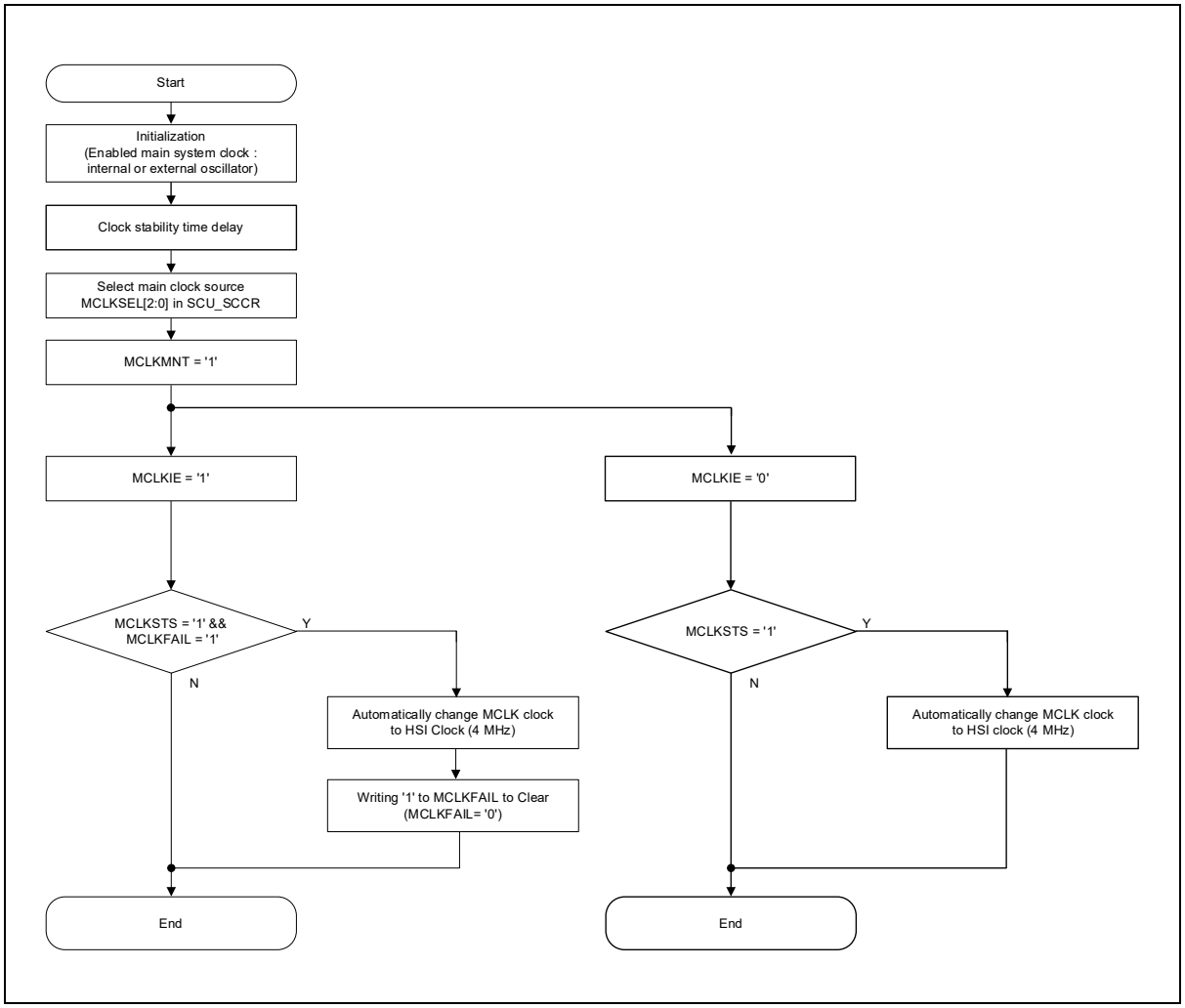
1. Set the HSECON and HSICON in SCU_CSCR register to enable a clock source set to MCLK and wait for a stable time. (HSE: 30 ms, HSI: 30 μs)
2. Select the main system clock source by configuring the MCLKSEL[2:0] in SCU_SCCR register. **(1)(2)(3)**
3. Set the MCLKMNT in SCU_CMR register to '1' to activate the MCLK monitoring function. MCLK errors can be detected by the clock monitoring circuit operating internally at LSI 32 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 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. If the MCLKSEL bits are set to use the LSE, ports must be set for the functions of SXIN and SXOUT respectively.

Figure 28 shows a workflow in which the clock monitoring circuit on the A31C14x and A31C12x series monitors MCLK.

Figure 28. MCLK Monitoring Workflow



4.4.10.4 HSE Monitoring

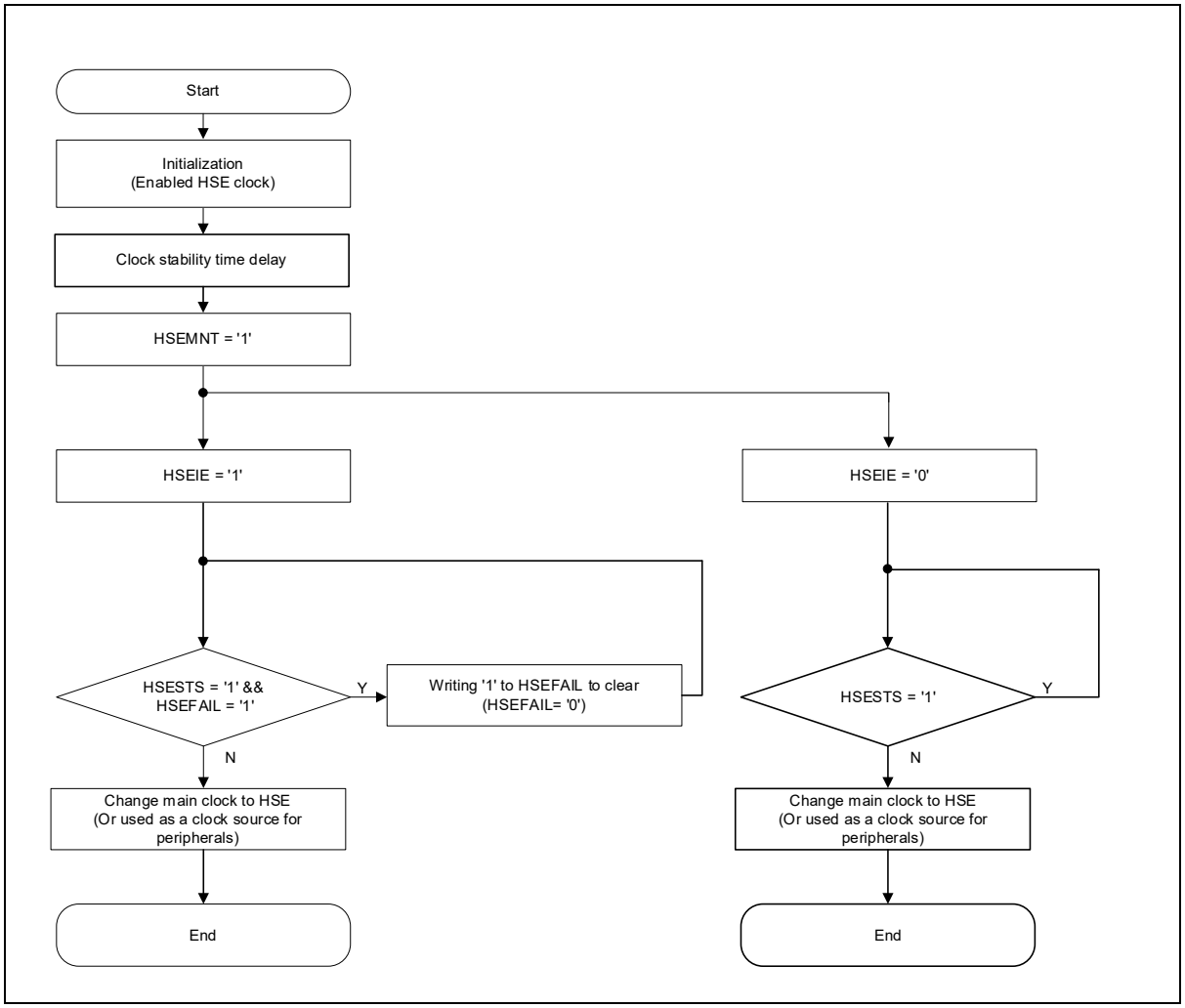
This section describes the HSE clock monitoring. For information on the HSE clock, see section 4.4.3.

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 32 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 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 29 shows a workflow in which the clock monitoring circuit on the A31C14x and A31C12x series monitors the HSE.

Figure 29. HSE Monitoring Workflow



4.4.10.5 LSE Monitoring

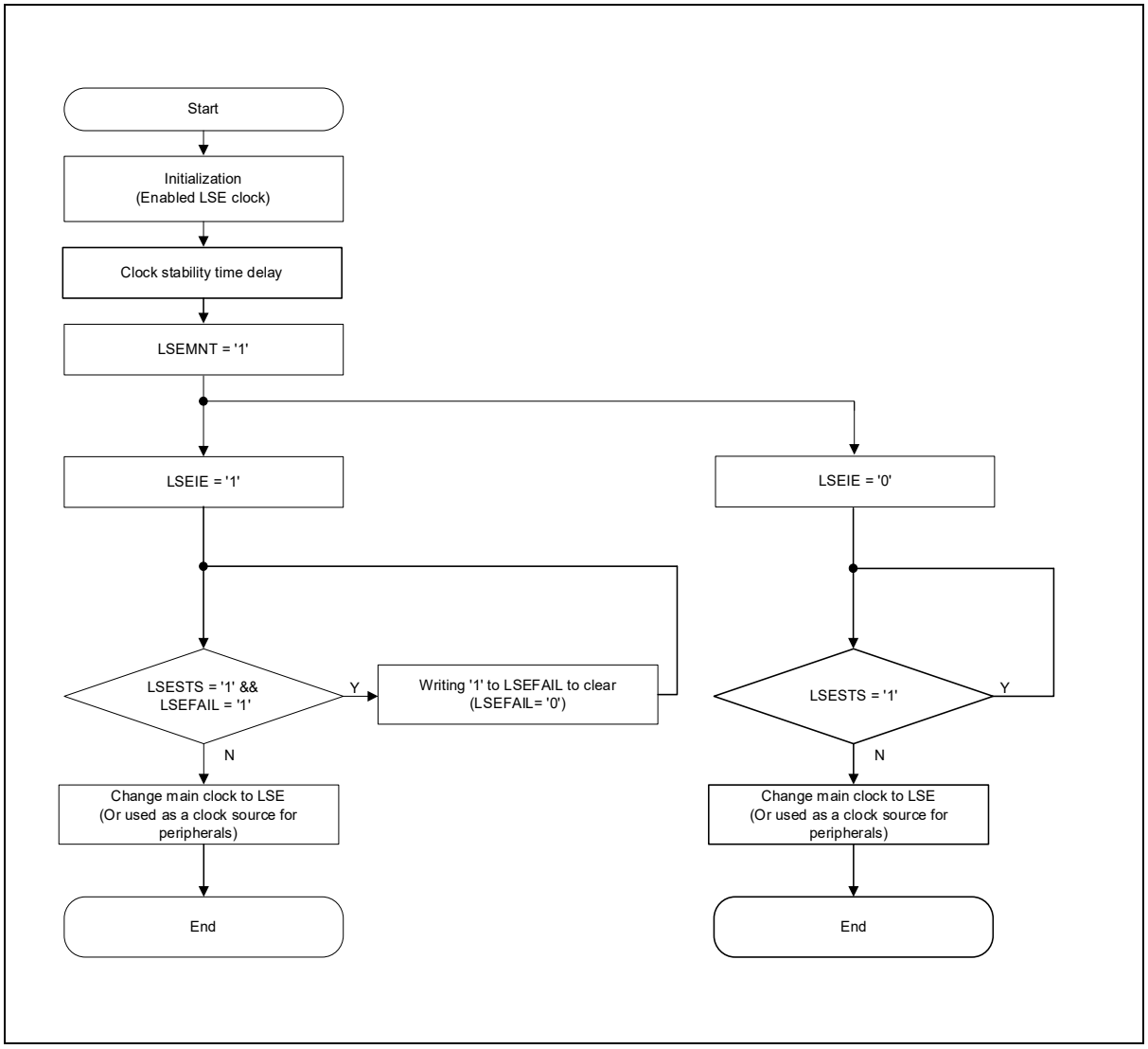
This section describes the LSE clock monitoring. For information on the LSE clock, see section 4.4.5.

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 32 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 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 30 shows a workflow in which the clock monitoring circuit on the A31C14x and A31C12x series monitors the LSE.

Figure 30. LSE Monitoring Workflow



4.4.11 Clock Configuration Procedure

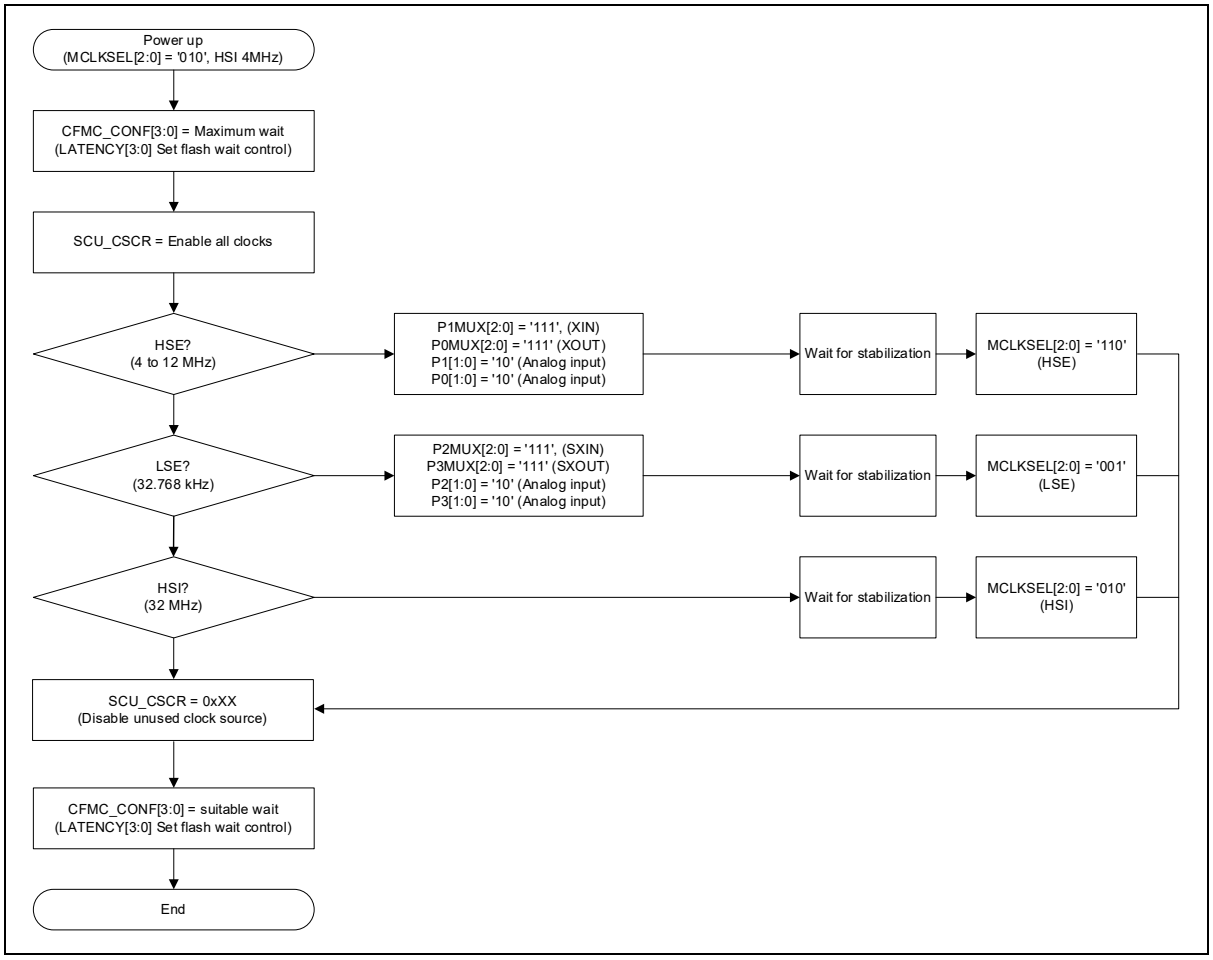
After power-on the device, 4 MHz HSI that 32 MHz HSI divided by 8, operates as a default system clock. The HSI is enabled during the power up sequence, and the HSE is configured by user controls with the HSI system clock.

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 31 shows an example sequence of how the system clock changes.

Figure 31. Clock Change Procedure



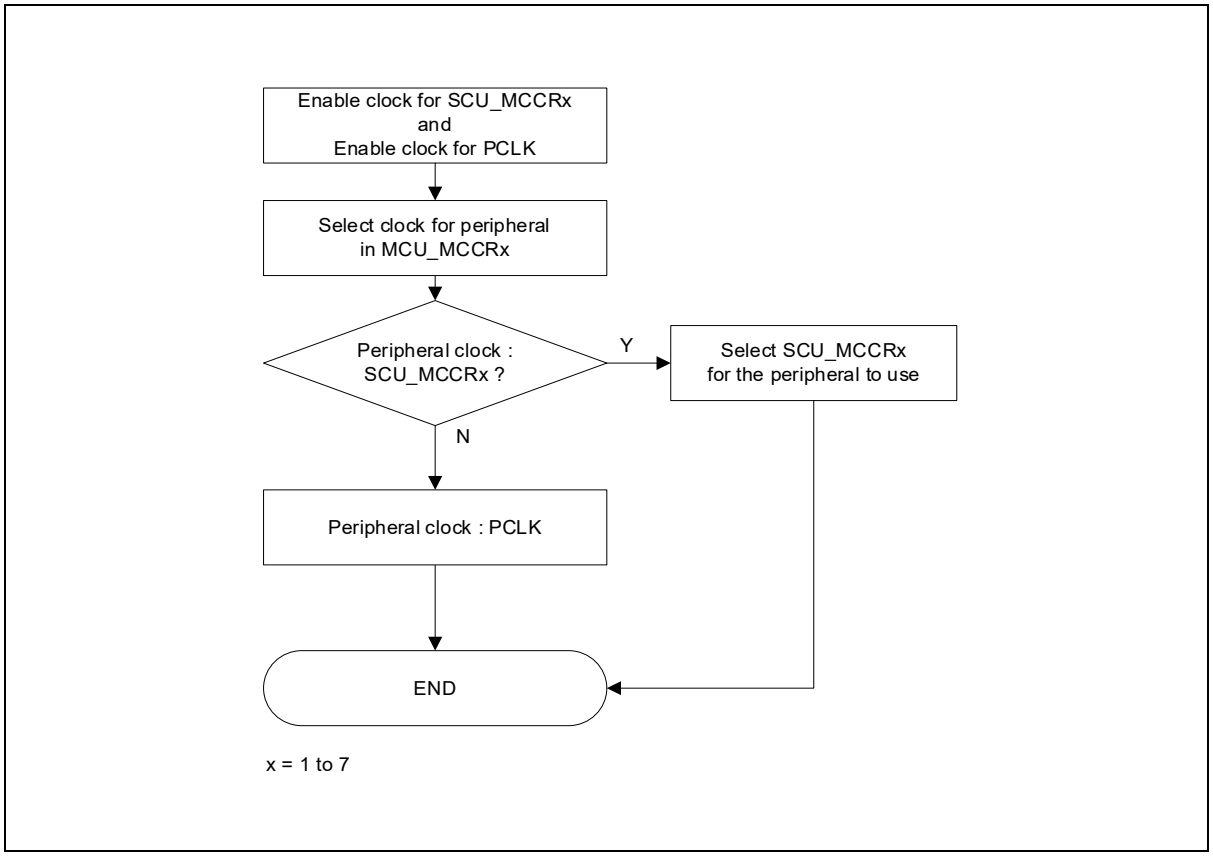
To speed up the system clock to the maximum operating frequency, users must check the flash memory wait control configuration. Flash read access time is one of limitation factors for performance. The wait control recommendation is suggested in Table 17.

Table 17. Flash Memory Wait Control Recommendation

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

Figure 32 shows how to set the peripheral clock. Typical selection of peripheral clocks is to use SCU_MCCRx (x = 1 to 7) and PCLK.

Figure 32. 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
- WDT Reset (WDTR)
- Software Reset
- Clock Oscillating Error Reset
- CPU Request Reset

The A31C14x and A31C12x series 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.)

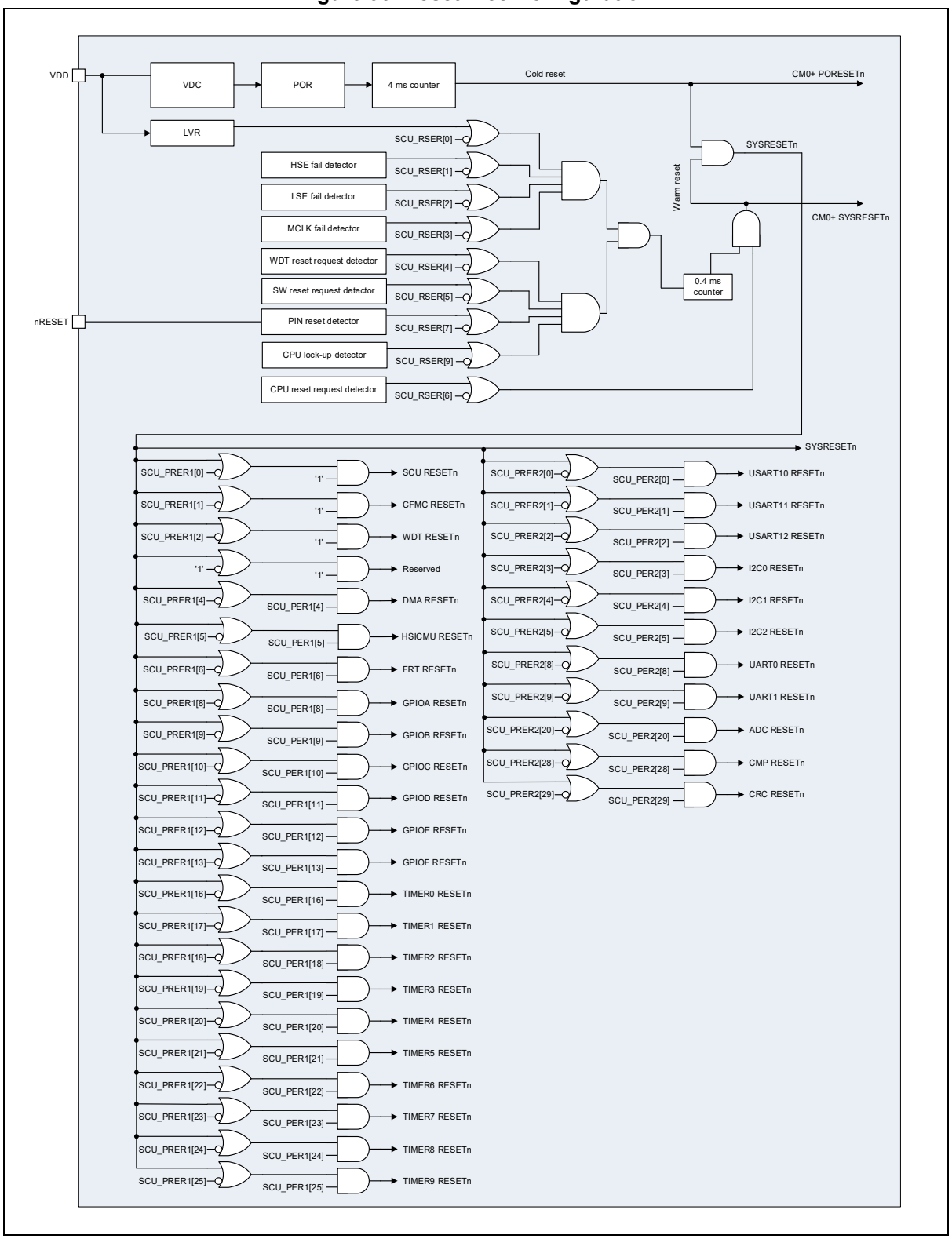
A reset event makes a chip to turn to an initial state. Reset sources of the cold reset and the warm reset are listed in Table 18.

Table 18. 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 33. Reset Tree Configuration



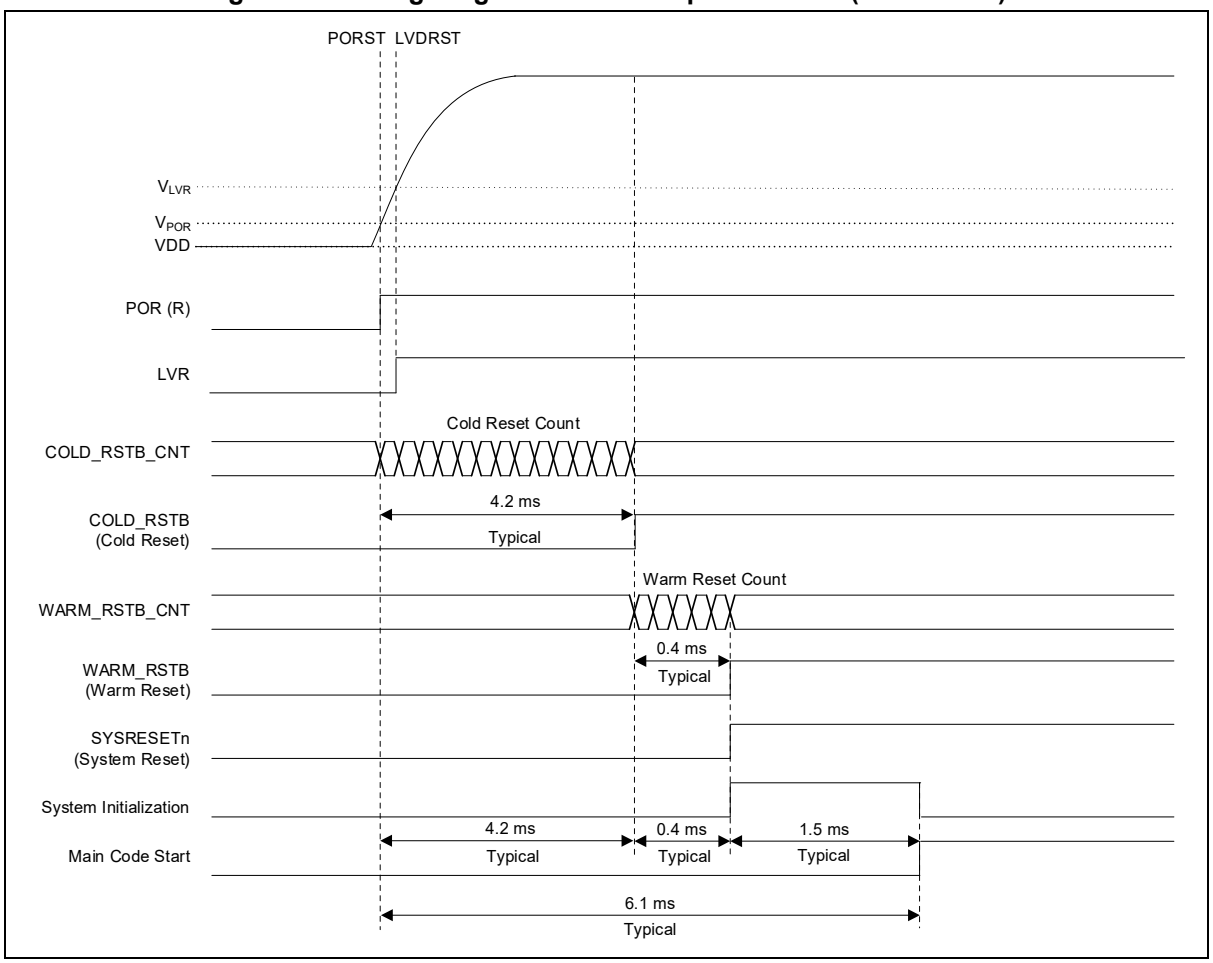
4.5.3 Cold Reset

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

During the cold reset process, when the applied voltage exceeds 1.48 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 (1.70 V), the cold reset is released and, after a waiting time of 0.4 ms for warm reset synchronization, booting begins.

Figure 34 shows waveform of power-up sequence and initial reset.

Figure 34. Timing Diagram of Power-Up Procedure (Cold Reset)



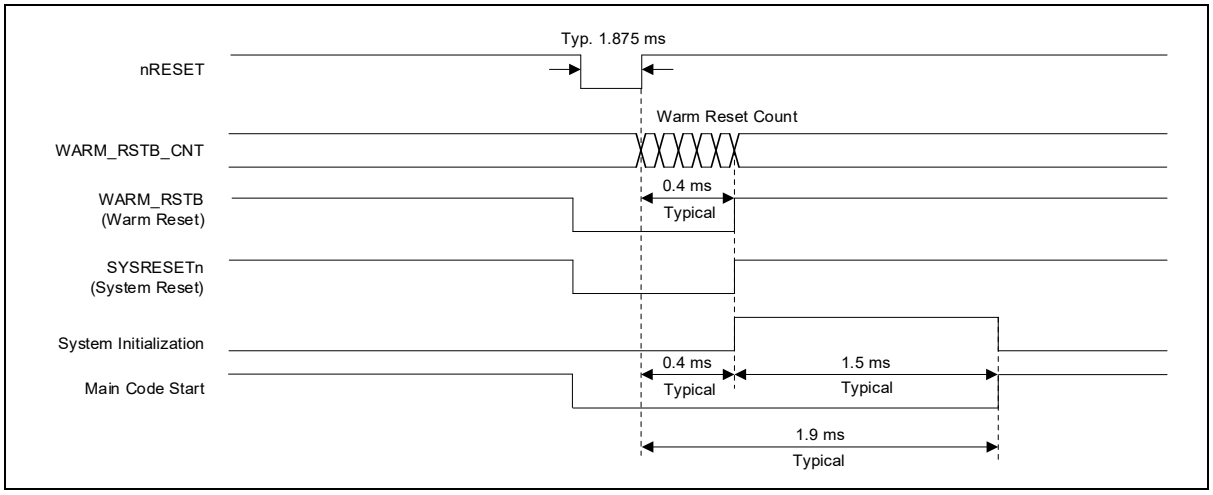
4.5.4 Warm Reset

A warm reset event triggers 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 35. Warm Reset Timing Diagram



NOTE:

1. The nRESET pin requires a stable low pulse of at least 1.875 ms to ensure proper internal reset operation. For additional information, refer to Section 4.5.6.

4.5.5 LVD Reset (LVR)

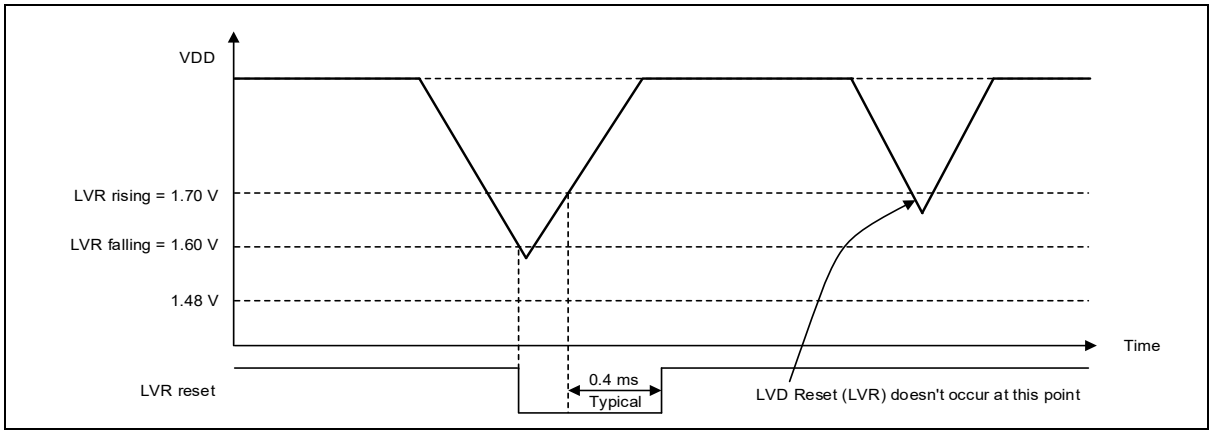
An LVR (Low-Voltage Detect Reset) event is triggered when the operating voltage drops below a certain level during the microcontroller's operation. Users can set the microcontroller whether to perform a reset or to perform an interrupt, when an LVR event is triggered.

Table 19 describes the range of LVR level and the register for LVR level control.

Table 19. LVR Level

LVR Level Range	Description
1.6 V to 5.5 V	LVR level with four steps This LVR level is set by LVRVS[1:0] in SCU_LVRCR.

Figure 36. LVR Diagram



4.5.6 nRESET Pin Reset

This reset signal is generated through the nRESET pin.

When the nRESET pin is driven low, all ongoing processes are aborted, and the microcontroller enters the reset state.

The nRESET pin requires a stable low pulse of at least 1.875 ms to ensure proper internal reset operation. The duration of the low pulse of reset can be configured via the PCU_RSTDBCR register.

Note that reset debounce is valid only when the LSI clock is enabled. This register is initialized only by a cold reset and retains its value during a warm reset.

The nRESET pin shown in Table 20 is assigned for the reset function.

Table 20. nRESET Pin

Pin Name	Type	Description
nRESET	Input	External reset input

4.5.7 WDT Reset

WDT (Watchdog timer) 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 A31C14x and A31C12x series 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 9. WDT.

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 operates with a supply voltage of 1.8 V to 5.5 V (V_{DD}). The analog peripherals are also powered through V_{DD}, internally connected to AV_{DD} within the microcontroller.

Table 21. Power Supply of V_{DD}

Parameter	Symbol	Min.	Max.	Unit
Supply voltage	V_{DD}	1.8	5.5	V

During a power-up process, a reset plays an important role and affects the entire process of system booting. A31C14x and A31C12x series has two power-related reset options as described below:

- POR_RST (Power-On Reset) that controls the voltage less than 1.48 V
- LVD_RST (Low-Voltage Detect Reset) that controls the voltage less than 1.70 V (Default)

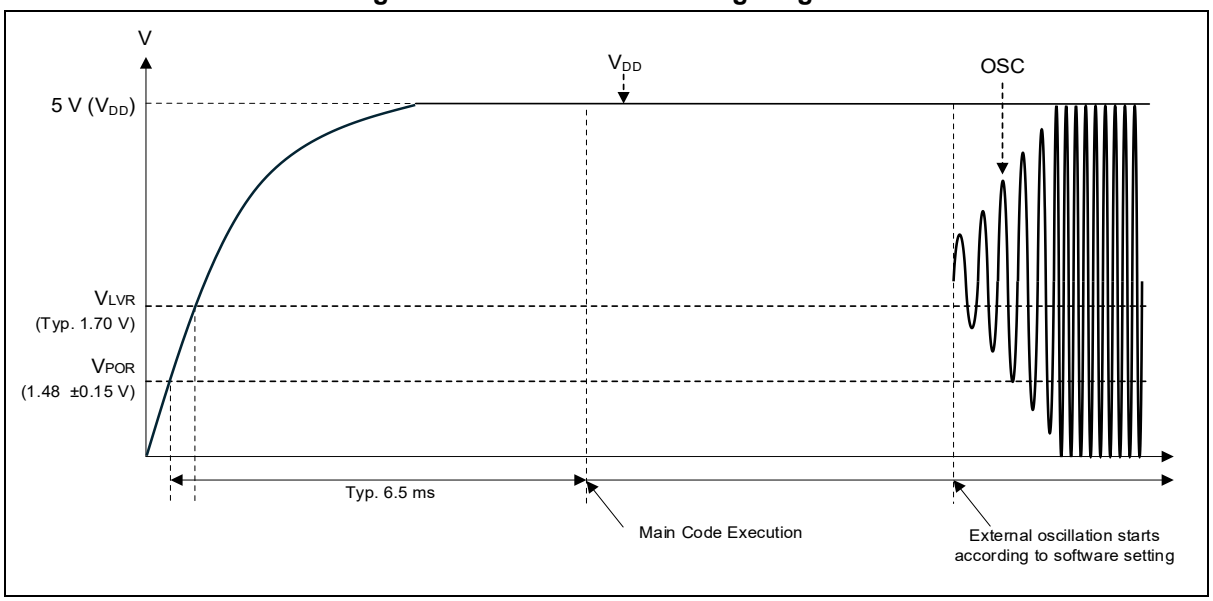
If the power level is higher than the POR and below the flash operating voltage code read operation may malfunction. To prevent this abnormal code read operation, the LVD_RST operates to generate the SYSRESETn internal signal and enter the microcontroller into reset mode to prevent abnormal operation.

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 37 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 A31C14x and A31C12x series, it notes that external oscillation is set to operate by software setting.

Figure 37. POR and LVR Timing Diagram

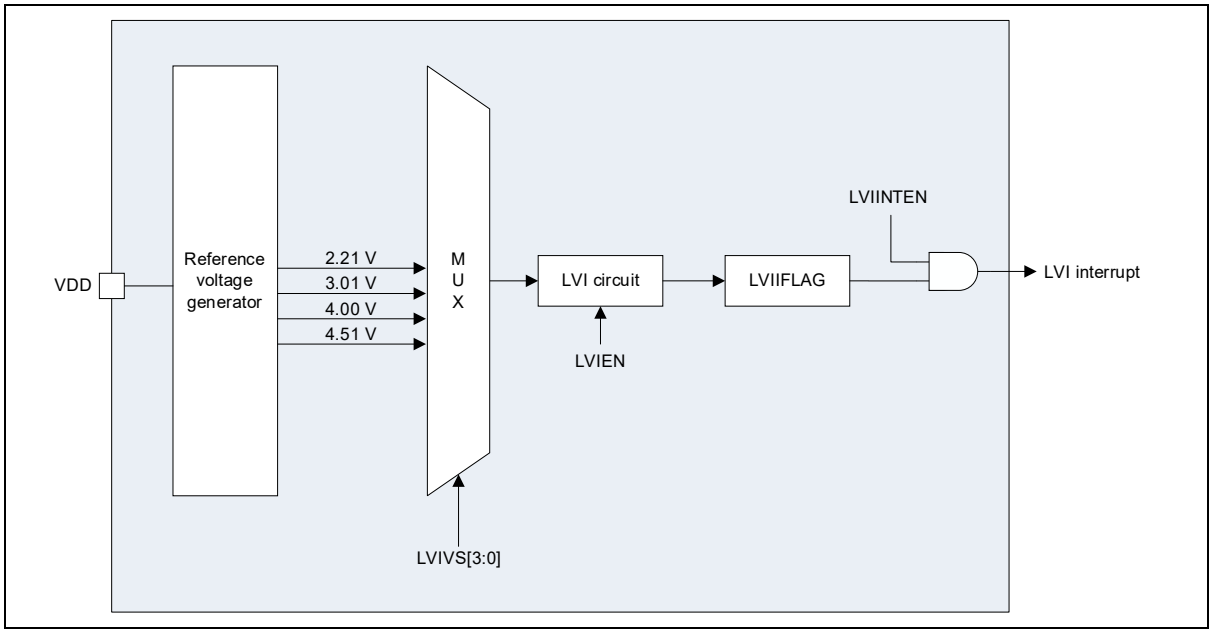


4.6.1 LVI Block Diagram

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

Figure 38 shows a block diagram of the LVI. For information on the LVI signal operation, refer to SCU_LVICR.

Figure 38. LVI Block Diagram

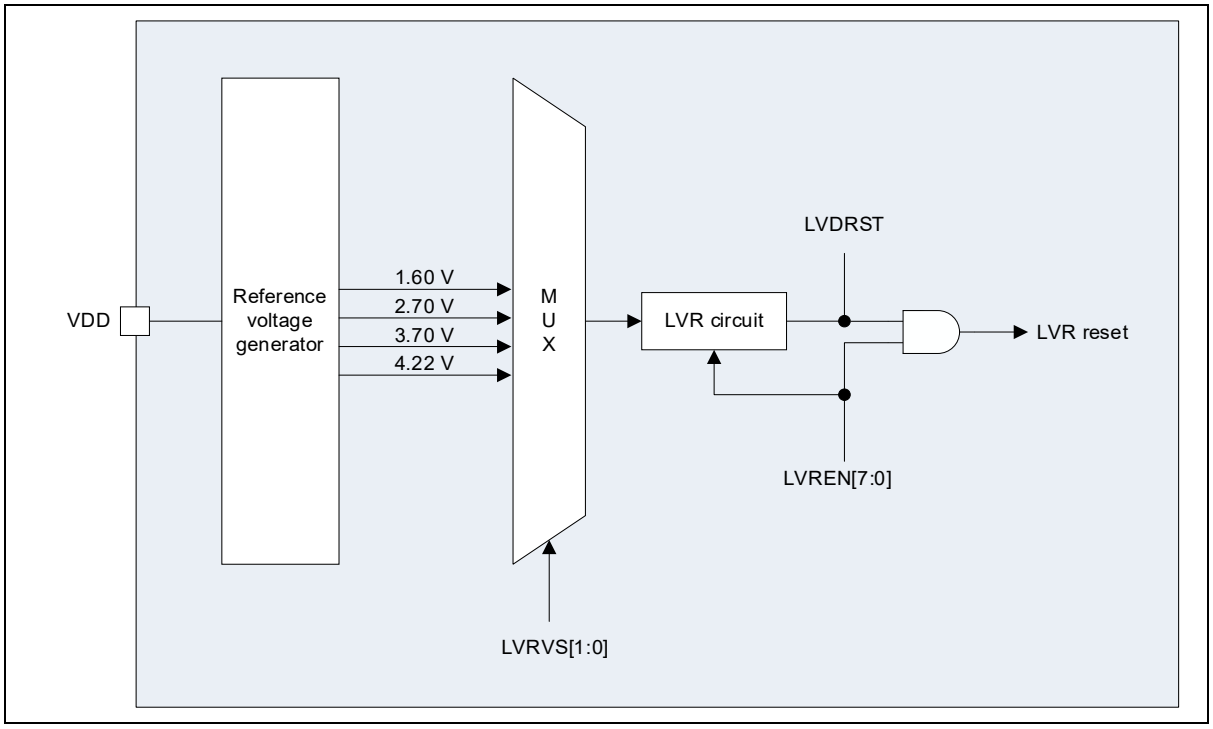


4.6.2 LVR Block Diagram

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

Figure 39 shows a block diagram of the LVR. For information on the LVR signal operation refer to SCU_LVRCR.

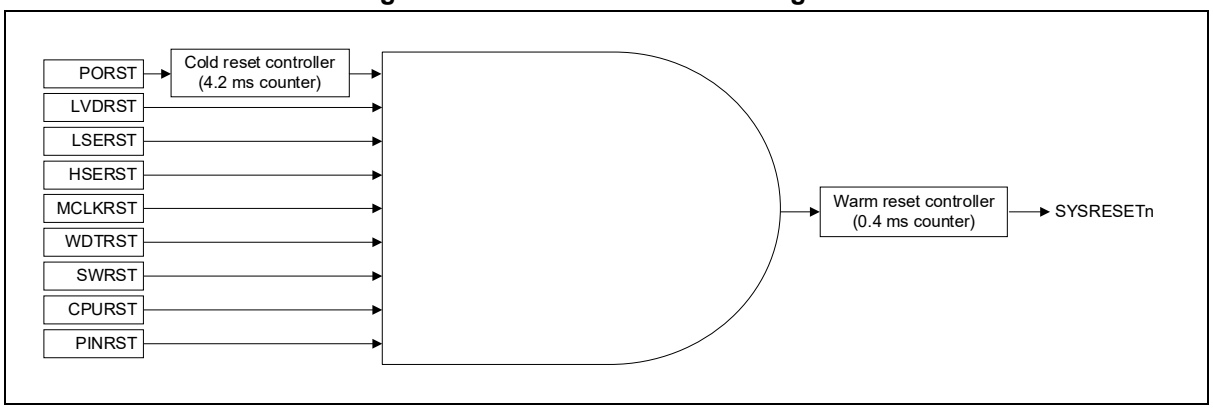
Figure 39. LVR Block Diagram



4.6.3 POR, LVR

For the case where the VDD input level is higher than the POR and lower than the minimum voltage (1.48 V) required for the Flash operation, the LVDRST (Min. 1.70 V) is set to improve stability and prevent operation errors such as code read. For more information, refer to section 4.5.

Figure 40. 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 A31C14x and A31C12x series operates in three modes below:

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

4.7.2 Transition of Operation Mode

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

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

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

Figure 41 describes transition between the operation modes.

Figure 41. Transition Between Operation Modes

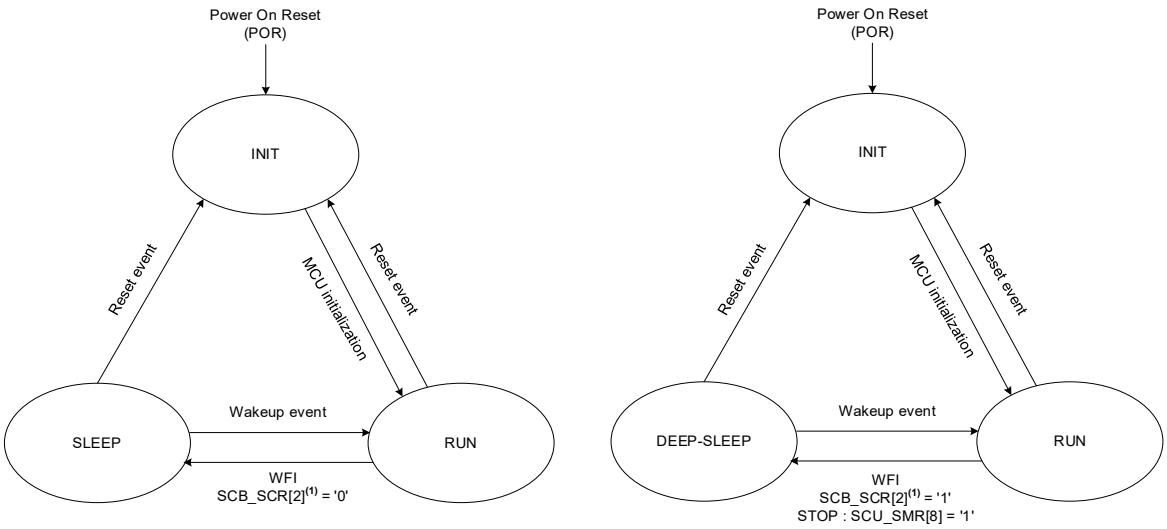


Table 22. Operation Mode

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.

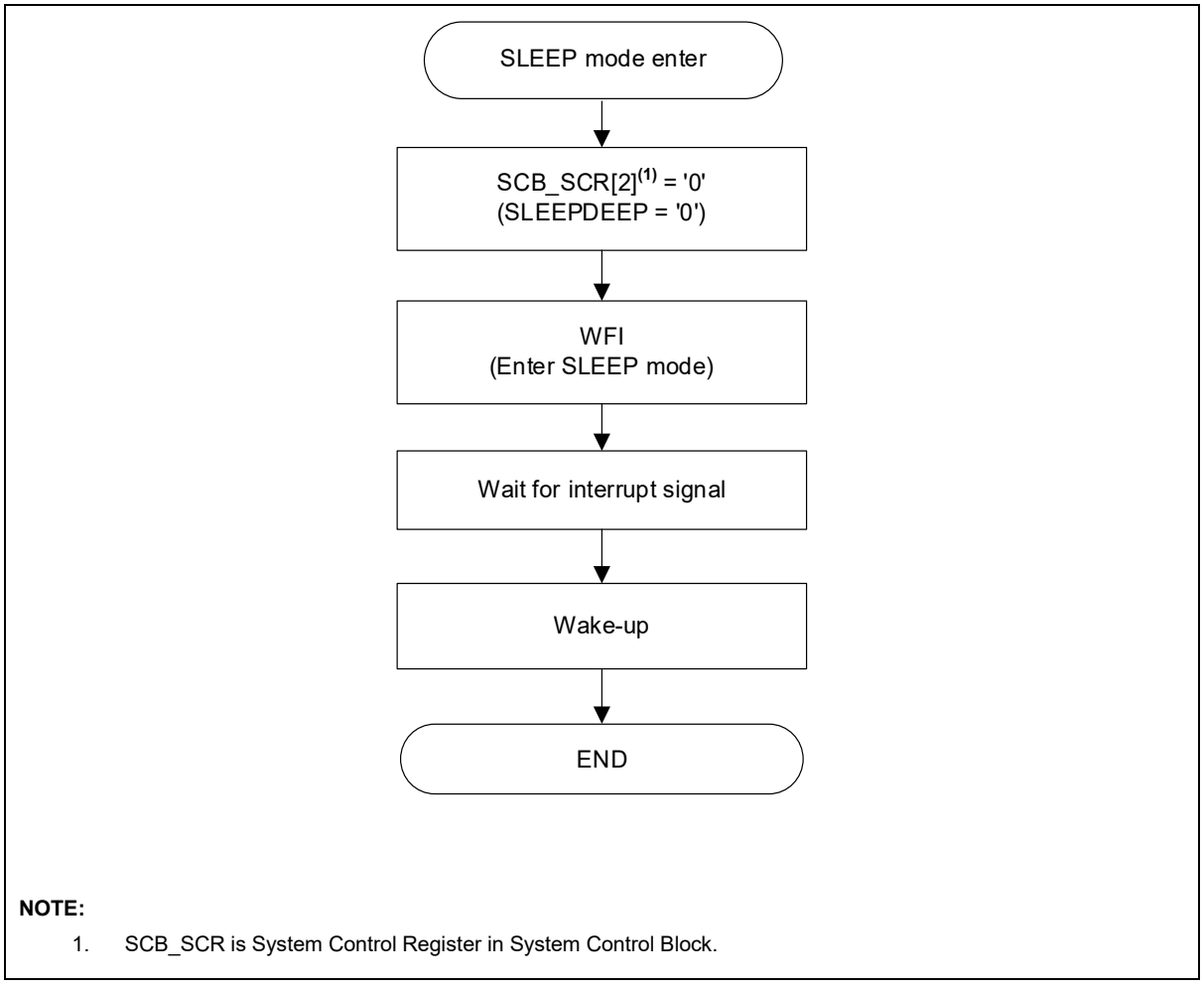
4.7.3 RUN Mode

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 42. SLEEP Mode Operation Sequence



NOTE:

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

4.7.5 DEEP-SLEEP (STOP1, STOP2) Mode

When the core goes under stop state using WFI instruction, the chip enters in DEEP-SLEEP mode.

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 WDT and FRT 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, 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, HSI, and LSI may not be used.

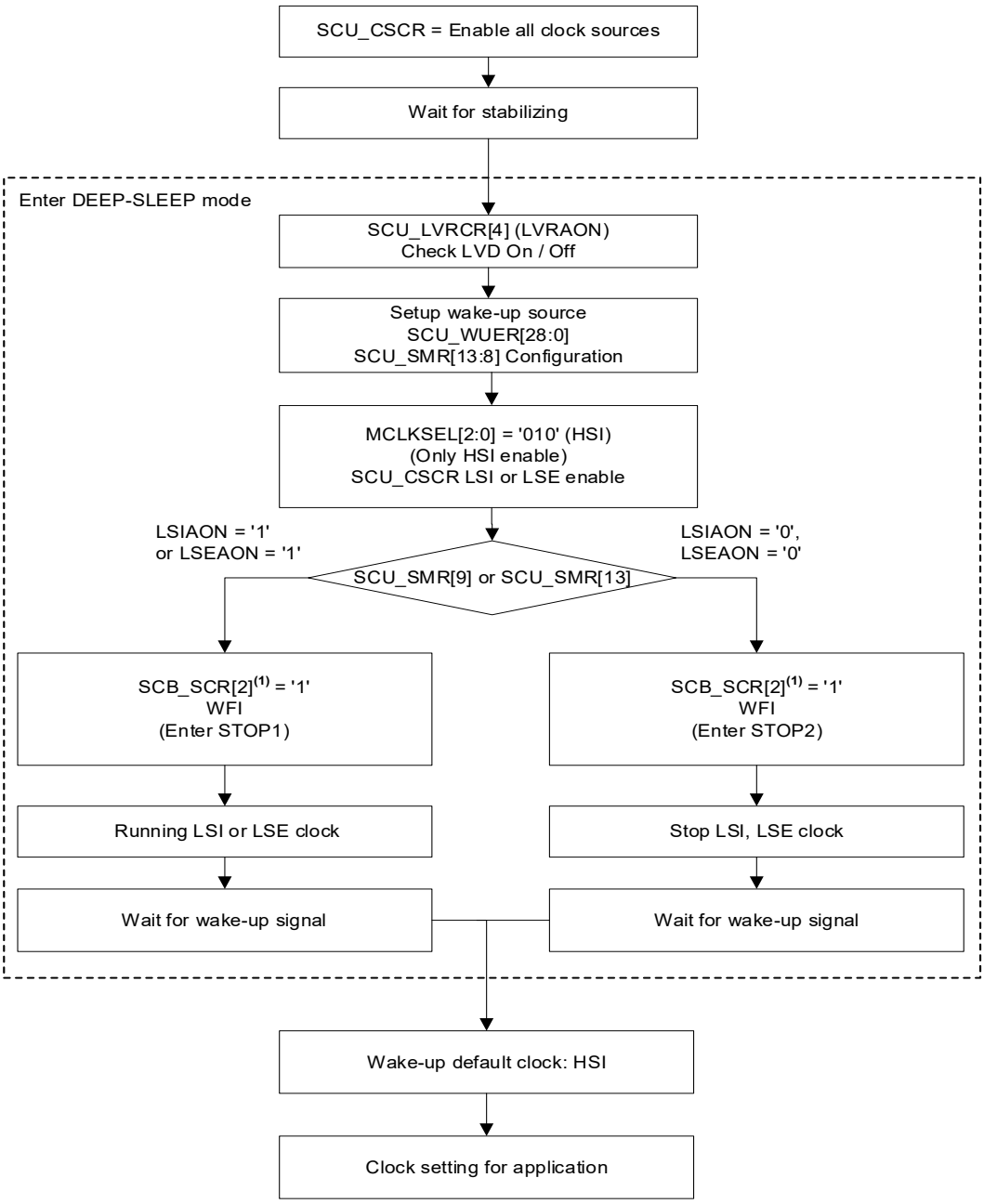
Table 23. DEEP-SLEEP Mode Configuration

Mode	Type	Condition	Wake-up Source
DEEP-SLEEP	STOP1	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
	STOP2	WFI SCB_SCR[2][(1)] = '1' (SLEEPDEEP = '1') SCU_SMR[9] = '0' (LSIAON = '0') or 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, 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 43. DEEP-SLEEP Mode Operation Sequence



NOTE:

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

4.7.6 Operation Mode Summary

Table 24 and Table 25 are configurable clocks according to each operating mode.

Table 24. Configurable Clock in Each RUN/SLEEP Mode

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

Table 25. Configurable Clocks in Each DEEP-SLEEP (STOP1, STOP2) Mode

Mode	Core	Peripheral	SCU_SMR				
			VDCAON	LSIAON	HSIAON	HSEAON	LSEAON
RUN	Active	User-defined	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
DEEP-SLEEP (STOP1)	Inactive	Only GPIO, USART10/11/12, WDT, FRT, LVI	OFF	User-defined	OFF	OFF	User-defined
DEEP-SLEEP (STOP2)	Inactive	Only GPIO, USART10/11/12, WDT, FRT, LVI	OFF	OFF	OFF	OFF	OFF

4.8 CHIPCONFIG and SCU Registers

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

Table 26. Base Address of CHIPCONFIG

Name	Base Address
CHIPCONFIG	0x4000_F000

Table 27. 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	0x4331_1440(2)	4.8.2
CHIPCONFIG_REVNR	0x0008	RO(1)	Revision Number Register	0x0000_00xx	4.8.3

NOTES:

1. 'RO' means 'Read-Only'.
2. The reset value of the CHIPCONFIG_CHIPID register in the A31C122 is 0x4331_1220.

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

Table 28. Base Address of SCU

Name	Base Address
SCU	0x4000_0000

Table 29. 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	0x03FF_3F77	4.8.10
SCU_PRER2	0x0024	RW	Peripheral Reset Enable Register 2	0x3010_033F	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_0001	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_0001	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_VDCCON	0x0064	RW	VDC Control Register	0x0000_007F	4.8.22
SCU_LVICR	0x0068	RW	LVI Control Register	0x0000_0000	4.8.23
SCU_LVISR	0x006C	RC	LVI Status Register	0x0000_0000	4.8.24
SCU_LVRCR	0x0070	RW	LVR Control Register	0x0000_0000	4.8.25
SCU_EOSCR	0x0080	RW	External Oscillator Control Register	0x0100_0102	4.8.26
SCU_MCCR1	0x0090	RW	Misc. Clock Control Register 1	0x0001_0000	4.8.27
SCU_MCCR3	0x0098	RW	Misc. Clock Control Register 3	0x0000_0000	4.8.28
SCU_MCCR4	0x009C	RW	Misc. Clock Control Register 4	0x0000_0000	4.8.29
SCU_MCCR5	0x00A0	RW	Misc. Clock Control Register 5	0x0000_0000	4.8.30
SCU_MCCR6	0x00A4	RW	Misc. Clock Control Register 6	0x0000_0000	4.8.31
SCU_MCCR7	0x00A8	RW	Misc. Clock Control Register 7	0x0000_0000	4.8.32
SCU_SYSTEN	0x00F0	WO	System Access Enable	0x0000_0000	4.8.33

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 0	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]																															
0xXXXX_XXXX																															
RO																															

31 0	CHIPID[31:0]	Chip ID bit.	
		0x4331_1440 0x4331_1430	128 KB flash memory/8 KB RAM/A31C144 64 KB flash memory/8 KB RAM/A31C143
		0x4331_1220 0x4331_1221	32 KB flash memory/6 KB RAM/A31C122 16 KB flash memory/6 KB RAM/A31C121

4.8.3 CHIPCONFIG_REVNR: Revision Number Register

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

CHIPCONFIG_REVNR=0x4000_F008

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

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

4.8.4 SCU_SMR: System Mode Register

The SCU_SMR contains bits controllable in DEEP-SLEEP[1] mode and informs 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	Reserved	HSIAON	LSIAON	VDCAON	Reserved
-			0	0	-	0	0	0	-
-			RW	RW	-	RW	RW	RW	-

13	LSEAON	LSE enablement in 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.
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.

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 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			GPIOFWUE	GPIOEWUE	GPIODWUE	GPIOCWUE	GPIOBWUE	GPIOAWUE	USART12WUE	USART11WUE	USART10WUE	Reserved	FRTWUE	WDTWUE	LVIWUE
-			0	0	0	0	0	0	0	0	0	-	0	0	0
-			RW	RW	RW	RW	RW	RW	RW	RW	RW	-	RW	RW	RW

13	GPIOFWUE	Use the GPIOF port event as a wake-up source
		0 Does not use the event as a wake-up source.
		1 Uses the event as a wake-up source.
12	GPIOEWUE	Use the GPIOE port event as a wake-up source
		0 Does not use the event as a wake-up source.
		1 Uses the event as a wake-up source.
11	GPIODWUE	Use the GPIOD port event as a wake-up source
		0 Does not use the event as a wake-up source.
		1 Uses the event as a wake-up source.
10	GPIOCWUE	Use the GPIOC port event as a wake-up source
		0 Does not use the event as a wake-up source.
		1 Uses the event as a wake-up source.
9	GPIOBWUE	Use the GPIOB port event as a wake-up source
		0 Does not use the event as a wake-up source.
		1 Uses the event as a wake-up source.
8	GPIOAWUE	Use the GPIOA port event as a wake-up source
		0 Does not use the event as a wake-up source.
		1 Uses the event as a wake-up source.
7	USART12WUE	Use the USART12 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	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	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.
2	FRTWUE	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	Use the WDT event as a wake-up source
		0 Does not use the event as a wake-up source.
		1 Uses the event as a wake-up source.
0	LVIWUE	Use the LVI event as a wake-up source
		0 Does not use the event as a wake-up source.
		1 Uses the event as a wake-up source.

4.8.7 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			GPIOFWU	GPIOEWU	GPIODWU	GPIOCWU	GPIOBWU	GPIOAWU	USART12WU	USART11WU	USART10WU	Reserved	FRTWU	WDTWU	LVIWU
-			0	0	0	0	0	0	0	0	0	-	0	0	0
-			RO	RO	RO	RO	RO	RO	RO	RO	RO	-	RO	RO	RO

13	GPIOFWU	GPIOF port wake-up event status
	0	Wake-up event has not occurred.
	1	Wake-up event occurs.
12	GPIOEWU	GPIOE port Wake-up event status
	0	Wake-up event has not occurred.
	1	Wake-up event occurs.
11	GPIODWU	GPIOD port Wake-up event status
	0	Wake-up event has not occurred.
	1	Wake-up event occurs.
10	GPIOCWU	GPIOC port Wake-up event status
	0	Wake-up event has not occurred.
	1	Wake-up event occurs.
9	GPIOBWU	GPIOB port Wake-up event status
	0	Wake-up event has not occurred.
	1	Wake-up event occurs.
8	GPIOAWU	GPIOA port Wake-up event status
	0	Wake-up event has not occurred.
	1	Wake-up event occurs.
7	USART12WU	USART12 Wake-up event status
	0	Wake-up event has not occurred.
	1	Wake-up event occurs.
6	USART11WU	USART11 Wake-up event status
	0	Wake-up event has not occurred.
	1	Wake-up event occurs.
5	USART10WU	USART10 Wake-up event status
	0	Wake-up event has not occurred.
	1	Wake-up event occurs.
2	FRTWU	FRT Wake-up event status
	0	Wake-up event has not occurred.
	1	Wake-up event occurs.
1	WDTWU	Watchdog Timer (WDT) Wake-up event status
	0	Wake-up event has not occurred.
	1	Wake-up event occurs.
0	LVIWU	LVI Wake-up event status
	0	Wake-up event has not occurred.
	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	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	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	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	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	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	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	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	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	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	MCLKRST	LSERST	HSERST	LVDRST
-			0	1	0	0	0	0	0	0	0	0
-			RWC1	RWC1	RWC1	RWC1	RWC1	RWC1	RWC1	RWC1	RWC1	RWC1

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

1	HSERST	HSE error reset flag	
		0	Read: The reset has not occurred. Write: N/A
		1	Read: The reset has occurred. Write: Clears the flag.
0	LVDRST	LVD reset flag	
		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. The bit values of this register are initialized when a cold reset occurs. Similarly, the SCU_PERx and SCU_PCERx registers are also initialized on cold reset. Once a reset event occurs, the peripherals whose corresponding bits are set to '1' are initialized and those set to '0' remain uninitialized. However, the POR reset initializes all peripherals.

SCU_PRER1=0x4000_0020

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

25	TIMER9	TIMER9 reset mask
		0 Masked
		1 Not masked
24	TIMER8	TIMER8 reset mask
		0 Masked
		1 Not masked
23	TIMER7	TIMER7 reset mask
		0 Masked
		1 Not masked
22	TIMER6	TIMER6 reset mask
		0 Masked
		1 Not masked
21	TIMER5	TIMER5 reset mask
		0 Masked
		1 Not masked
20	TIMER4	TIMER4 reset mask
		0 Masked
		1 Not masked
19	TIMER3	TIMER3 reset mask
		0 Masked
		1 Not masked
18	TIMER2	TIMER2 reset mask
		0 Masked
		1 Not masked
17	TIMER1	TIMER1 reset mask
		0 Masked
		1 Not masked
16	TIMER0	TIMER0 reset mask
		0 Masked
		1 Not masked
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
5	HSICMU	HSICMU reset mask
		0 Masked
		1 Not masked
4	DMA	DMA reset mask
		0 Masked
		1 Not masked
2	WDT	WDT reset mask
		0 Masked
		1 Not masked
1	CFMC	CFMC reset mask
		0 Masked
		1 Not masked
0	SCU	SCU reset mask
		0 Masked
		1 Not masked

4.8.11 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	CMP	Reserved							ADC	Reserved										UART1	UART0	Reserved		I2C2	I2C1	I2C0	USART12	USART11	USART10	
-		1	1	-							1	-										1	1	-		1	1	1	1	1	1	
-		RW	RW	-							RW	-										RW	RW	-		RW	RW	RW	RW	RW	RW	

29	CRC	CRC reset mask
		0 Masked
		1 Not masked
28	CMP	CMP reset mask
		0 Masked
		1 Not masked
20	ADC	ADC reset mask
		0 Masked
		1 Not masked
9	UART1	UART1 reset mask
		0 Masked
		1 Not masked
8	UART0	UART0 reset mask
		0 Masked
		1 Not masked
5	I2C2	I2C2 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
2	USART12	USART12 reset mask
		0 Masked
		1 Not masked
1	USART11	USART11 reset mask
		0 Masked
		1 Not masked
0	USART10	USART10 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 are enabled at the initial reset, additional action is required to have other peripherals enabled if users want to use them. By writing a '0' to the bit representing the peripheral the users will not use, the users can reset it.

SCU_PER1=0x4000_0028

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

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

12	GPIOE	Enable or disable GPIOE
		0 Disabled
		1 Enabled
11	GPIOD	Enable or disable GPIOD
		0 Disabled
		1 Enabled
10	GPIOC	Enable or disable GPIOC
		0 Disabled
		1 Enabled
9	GPIOB	Enable or disable GPIOB
		0 Disabled
		1 Enabled
8	GPIOA	Enable or disable GPIOA
		0 Disabled
		1 Enabled
6	FRT	Enable or disable FRT
		0 Disabled
		1 Enabled
5	HSICMU	Enable or disable HSICMU
		0 Disabled
		1 Enabled
4	DMA	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	CMP	Reserved							ADC	Reserved										UART1	UART0	Reserved		I2C2	I2C1	I2C0	USART12	USART11	USART10	
-		0	0	-							0	-										0	0	-		0	0	0	0	0	1	
-		RW	RW	-							RW	-										RW	RW	-		RW	RW	RW	RW	RW	RW	

29	CRC	Enable or disable CRC
		0 Disabled
		1 Enabled
28	CMP	Enable or disable CMP
		0 Disabled
		1 Enabled
20	ADC	Enable or disable ADC
		0 Disabled
		1 Enabled
9	UART1	Enable or disable UART1
		0 Disabled
		1 Enabled
8	UART0	Enable or disable UART0
		0 Disabled
		1 Enabled
5	I2C2	Enable or disable I2C2
		0 Disabled
		1 Enabled
4	I2C1	Enable or disable I2C1
		0 Disabled
		1 Enabled
3	I2C0	Enable or disable I2C0
		0 Disabled
		1 Enabled
2	USART12	Enable or disable USART12
		0 Disabled
		1 Enabled
1	USART11	Enable or disable USART11
		0 Disabled
		1 Enabled
0	USART10	Enable or disable USART10
		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 it is clocked.

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
Reserved						TIMER9	TIMER8	TIMER7	TIMER6	TIMER5	TIMER4	TIMER3	TIMER2	TIMER1	TIMER0	Reserved		GPIOF	GPIOE	GPIOD	GPIOC	GPIOB	GPIOA	Reserved		FRT	HSICMU	DMA	Reserved		
-						0	0	0	0	0	0	0	0	0	0	-		0	0	0	0	0	0	-		0	0	0	1111		
-						RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	-		RW	RW	RW	RW	RW	RW	-		RW	RW	RW	RO		

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

12	GPIOE	Enable or disable the clock to GPIOE
		0 Disabled
		1 Enabled
11	GPIOD	Enable or disable the clock to GPIOD
		0 Disabled
		1 Enabled
10	GPIOC	Enable or disable the clock to GPIOC
		0 Disabled
		1 Enabled
9	GPIOB	Enable or disable the clock to GPIOB
		0 Disabled
		1 Enabled
8	GPIOA	Enable or disable the clock to GPIOA
		0 Disabled
		1 Enabled
6	FRT	Enable or disable the clock to FRT
		0 Disabled
		1 Enabled
5	HSICMU	Enable or disable the clock to HSICMU
		0 Disabled
		1 Enabled
4	DMA	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	CMP	Reserved		ADC		Reserved	UART1	UART0	Reserved	I2C2	I2C1	I2C0	USART12	USART11	USART10
-	0	0	-		0		-	0	0	-	0	0	0	0	0	1
-	RW	RW	-		RW		-	RW	RW	-	RW	RW	RW	RW	RW	RW

29	CRC	Enable or disable the clock to CRC
		0 Disabled
		1 Enabled
28	CMP	Enable or disable the clock to CMP
		0 Disabled
		1 Enabled
20	ADC	Enable or disable the clock to ADC
		0 Disabled
		1 Enabled
9	UART1	Enable or disable the clock to UART1
		0 Disabled
		1 Enabled
8	UART0	Enable or disable the clock to UART0
		0 Disabled
		1 Enabled
5	I2C2	Enable or disable the clock to I2C2
		0 Disabled
		1 Enabled
4	I2C1	Enable or disable the clock to I2C1
		0 Disabled
		1 Enabled
3	I2C0	Enable or disable the clock to I2C0
		0 Disabled
		1 Enabled
2	USART12	Enable or disable the clock to USART12
		0 Disabled
		1 Enabled
1	USART11	Enable or disable the clock to USART11
		0 Disabled
		1 Enabled
0	USART10	Enable or disable the clock to USART10
		0 Disabled
		1 Enabled

4.8.16 SCU_CSCR: Clock Source Control Register

A31C14x 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 (PF2, PF3).
2. For the HSE operation, the corresponding Pn_MR registers of the XIN, XOUT ports must be set to ‘111’. HSE (PF0, PF1).

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	MCLKSEL[2:0]
-	0011	-	000	-	010
-	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
2 0	MCLKSEL[2:0](1)	Main system clock selection	
		001	LSE(3)
		010	HSI
		110	HSE
		Others	Reserved

NOTES:

1. If the MCLKSEL[2:0] bits are set to use the HSE, PF0 and PF1 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, PF2 and PF3 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
		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		WDTINTEN	MCLKFAILIEN	LVIEN
-	0x00	0	-		0	0	0
-	RW	RW	-		RW	RW	RW

23 16	NMISRC[7:0]	NMI source selection bit One of the interrupt priority levels can be used as the NMI source.
15	NMIINEN	Enable or disable the NMIs (Interrupts that have been specified as NMI sources) 0 Disables. 1 Enables.
2	WDTINTEN	Enable or disable the WDT interrupt as an NMI 0 Disables 1 Enables.
1	MCLKFAILEN	Enable or disable the MCLK error interrupt as an NMI 0 Disables 1 Enables.
0	LVIEN	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		WDTINTSTS	MCLKFAILST	LVISTS
0x00	-	0	-		0	0	0
WO	-	RO	-		RWC1	RWC1	RWC1

31 24	WTIDKY[7:0]	Write identification key	
		Writing 0x8C to the bit field enables clearing flags in this register.	
15	NMIINTSTS	Whether an NMI has occurred (The flag can be cleared by writing to the triggered NMI's bit.)	
		0	No NMI has occurred.
		1	An NMI has occurred.
2	WDTINTSTS	WDT interrupt flag (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	MCLK error interrupt flag	
		0	The interrupt has not occurred.
		1	The interrupt has occurred (writing a '1' to the bit clears the flag).
0	LVISTS	LVI interrupt flag	
		0	The interrupt has not occurred.
		1	The interrupt has occurred (writing a '1' to the bit clears the flag).

4.8.22 SCU_VDCCON: VDC Control Register

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

SCU_VDCCON=0x4000_0064

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

7 0	VDCWDLY[7:0]	VDC Warm-up delay count value When the SCU wakes up from DEEP-SLEEP mode, the Warm-up delay is inserted after waiting until the VDC output is stabilized. Default delay value is 0x7F (254us) The delay value can be set to the max. value of 510us in the VDCWDLY bits (Max: 0xFF).

4.8.23 SCU_LVICR: LVI (Low-Voltage Indicator) 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[1:0]
-			0	-	0	0	-	00
-			RW	-	RW	RW	-	RW

7	LVIEN	Enable or disable the LVI	
		0	Disables.
		1	Enables.
5	LVIINTEN	Enable or disable the LVI interrupt.	
		0	Disables.
		1	Enables.
4	LVIAON	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.
1 0	LVIVS[1:0]	LVI interrupt voltage selection bits	
		00	2.21 V (default)
		01	3.01 V
		10	4.00 V
		11	4.51 V

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

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

SCU_LVISR=0x4000_006C

31 30 29 28 27 26 25 24	23 22 21 20 19 18 17 16	15 14 13 12 11 10 9 8	7 6	5	4 3 2 1	0
WTIDKY[7:0]	Reserved			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	LVI interrupt status (raw data)	
		0	The LVI interrupt has ended.
		1	The LVI interrupt is active.

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

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

SCU_LVRCR=0x4000_0070

31 30 29 28 27 26 25 24	23 22 21 20 19 18 17 16	15 14 13 12 11 10 9 8	7 6 5	4	3 2	1 0
Reserved		LVREN[7:0]	Reserved	LVRAON	Reserved	LVRVS[1:0]
-		0x00	-	0	-	00
-		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	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.
1 0	LVRVS [1:0]	LVR voltage selection bits	
		00	1.60 V
		01	2.70 V
		10	3.70 V
		11	4.22 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.

4.8.26 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
Reserved	LSEISEL[1:0]	Reserved	LSENFEN	Reserved	HSEISEL[1:0]	Reserved	HSENFEN	Reserved	HSENFSEL[1:0]
-	01	-	0	-	01	-	0	-	10
-	RW	-	RW	-	RW	-	RW	-	RW

25 24	LSEISEL[1:0](1)	LSE current capability selection	
		00	170 nA
		01	210 nA
		10	250 nA
		11	290 nA
16	LSENFEN	LSE noise filter enable bit	
		0	Disable
		1	Enable
9 8	HSEISEL[1:0](2)	HSE current capability selection	
		00	Reserved
		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	Reserved

NOTE:

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

Table 30. External Oscillator Control Settings

Freq. (MHz)	HSENFSEL[1:0]	HSEISEL[9:8]	NC Delay (ns)
4	01	11	18
8	10	10	13
12	10	01	13

4.8.27 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](1)	WDT clock source selection bit	
		000	LSI (32 kHz)
		001	LSE (32.768 kHz)
		101	HSI (32 MHz)
		110	HSE
		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 (32 kHz)
		001	LSE (32.768 kHz)
		100	MCLK
		101	HSI (32 MHz)
		110	HSE
		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.

NOTE:

1. It is not recommended to use the same clock source for both the WDT and MCLK.

4.8.28 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 2. 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	TIMER59CSEL[2:0]	TIMER59CDIV[7:0]	Reserved	TIMER04CSEL[2:0]	TIMER04CDIV[7:0]
-	000	0x00	-	000	0x00
-	RW	RW	-	RW	RW

26 24	TIMER59CSEL[2:0]	Clock source selection for TIMER5 through TIMER9	
		000	LSI (32 kHz)
		001	LSE (32.768 kHz)
		100	MCLK
		101	HSI (32 MHz)
		110	HSE
		Other	Reserved
23 16	TIMER59CDIV[7:0](1)	Clock divider (N) for TIMER5 through TIMER9	
		0x00	Disables the divider
		0xN	Clock source / N, N must be larger than zero. If the timer clock (TMCLK) is set in this register, the clock speed must satisfy PCLK ≥ TMCLK × 3.
10 8	TIMER04CSEL[2:0]	Clock source selection for TIMER0 through TIMER4	
		000	LSI (32 kHz)
		001	LSE (32.768 kHz)
		100	MCLK
		101	HSI (32 MHz)
		110	HSE
		Other	Reserved
7 0	TIMER04CDIV[7:0](1)	Clock divider (N) for TIMER0 through TIMER4	
		0x00	Disables the divider
		0xN	Clock source / N, N must be larger than zero. 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 is 1 / 2 slower than the clock source set by SCU_MCCRx register, frequency should be divided more than six times.

4.8.29 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	
-	000	0x00	-	
-	RW	RW	-	

26 24	PGADCSEL[2:0] (PA, PB)	Clock source for port group A debouncing	
		000	LSI (32 kHz)
		001	LSE (32.768 kHz)
		100	MCLK
		101	HSI (32 MHz)
		110	HSE
		Other	Reserved
23 16	PGADCDIV[7:0] (PA, PB)	Clock divider N for port group A debouncing	
		0x00	Disables the divider
		N	Clock source / N, N must be larger than zero.

4.8.30 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

25 24	PGCDCSEL[2:0] (PE, PF)	Clock source for port group C debouncing	
		000	LSI (32 kHz)
		001	LSE (32.768 kHz)
		100	MCLK
		101	HSI (32 MHz)
		110	HSE
		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 (32 kHz)
		001	LSE (32.768 kHz)
		100	MCLK
		101	HSI (32 MHz)
		110	HSE
		Other	Reserved
7 0	PGBDCDIV[7:0] (PC, PD)	Clock divider (N) for port group B debouncing	
		0x00	Disables the divider
		N	Clock source / N, N must be larger than zero.

4.8.31 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		FRTCSEL[2:0]	FRTCDIV[7:0]
-		000	0x00
-		RW	RW

10 8	FRTCSEL[2:0]	FRT clock source selection bit	
		000	LSI (32 kHz)
		001	LSE (32.768 kHz)
		100	MCLK
		101	HSI (32 MHz)
		110	HSE
		Other	Reserved
7 0	FRTCDIV[7:0]	FRT clock divider (N)	
		0x00	Disables the divider
		N	Clock source / N, N must be larger than zero.

4.8.32 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_MCCR7=0x4000_00A8

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

26 24	UARTCSEL[2:0]	UART clock source selection bit (UART0 through UART1)	
		000	LSI (32 kHz)
		001	LSE (32.768 kHz)
		100	MCLK
		101	HSI (32 MHz)
		110	HSE
		Other	Reserved
23 16	UARTCDIV[7:0]	UART clock divider (N)	
		0x00	Disables the divider
		N	Clock source / N, N must be larger than zero. If the UART clock is set in this register, the clock speed must satisfy PCLK ≥ UART clock set by SCU_MCCR7 / 2.

4.8.33 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.

4.8.34 SCU Register Map Summary

Table 31. SCU CHIPCONFIG Register Map Summary

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

Table 32. 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		HSIAON	LSIAON	VDCAON			PREVMODE[1:0]					
	Reset value																			0	0		0	0	0			1	1				
0x08	SCU_SRCR																												STBYO				SWRST
	Reset value																												0				0
0x10	SCU_WUER																			GPIOFWUE	GPIOEWUE	GPIODWUE	GPIOCWUE	GPIOBWUE	GPIOAWUE	USART12WUE	USART11WUE	USART10WUE			FRTWUE	WDTWUE	LVIWUE
	Reset value																			0	0	0	0	0	0	0	0	0			0	0	0
0x14	SCU_WUSR																			GPIOFWU	GPIOEWU	GPIODWU	GPIOCWU	GPIOBWU	GPIOAWU	USART12WU	USART11WU	USART10WU			FRTWU	WDTWU	LVIWU
	Reset value																			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

Table 32. 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							TIMER9	TIMER9	TIMER7	TIMER6	TIMER5	TIMER4	TIMER3	TIMER2	TIMER1	TIMER0			GPIOF	GPIOE	GPIOD	GPIOC	GPIOB	GPIOA	Res	FRT	HSICMU	DMA		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	CMP								ADC											UART1	UART0			I2C2	I2C1	I2C0	USART12	USART11	USART10
	Reset value			1	1								1											1	1			1	1	1	1	1	1
0x28	SCU_PER1							TIMER9	TIMER8	TIMER7	TIMER6	TIMER5	TIMER4	TIMER3	TIMER2	TIMER1	TIMER0			GPIOF	GPIOE	GPIOD	GPIOC	GPIOB	GPIOA		FRT	HSICMU	DMA				
	Reset value							0	0	0	0	0	0	0	0	0	0			0	0	0	0	0	0	0	0	0	0	1	1	1	1
0x2C	SCU_PER2			CRC	CMP								ADC											UART1	UART0			I2C2	I2C1	I2C0	USART12	USART11	USART10
	Reset value			0	0								0											0	0			0	0	0	0	0	1
0x30	SCU_PCER1							TIMER9	TIMER8	TIMER7	TIMER6	TIMER5	TIMER4	TIMER3	TIMER2	TIMER1	TIMER0			GPIOF	GPIOE	GPIOD	GPIOC	GPIOB	GPIOA		FRT	HSICMU	DMA				
	Reset value							0	0	0	0	0	0	0	0	0	0			0	0	0	0	0	0	0	0	0	0	1	1	1	1
0x34	SCU_PCER2			CRC	CMP								ADC											UART1	UART0			I2C2	I2C1	I2C0	USART12	USART11	USART10
	Reset value			0	0								0											0	0			0	0	0	0	0	1
0x40	SCU_CSCR																									LSECON		LSICON		HSICON		HSECON	
	Reset value																									0		1		1		0	
0x44	SCU_SCCR					HCLKDIV[3:0]									PCLKDIV[2:0]																MCLKSEL[2:0]		
	Reset value					0	0	1	1						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

Table 32. 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													WDTINTEN	MCLKFAILEN	LVIEN
	Reset value									0	0	0	0	0	0	0	0	0													0	0	0
0x58	SCU_NMISR	WTIDKY[7:0]																NMIINTSTS													WDTINTSTS	MCLKFAILSTS	LVISTS
	Reset value	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[1:0]	
	Reset value																										0		0	0		0	0
0x6C	SCU_LVISR	WTIDKY[7:0]																										LVIIFLAG					LVIINTSTS
	Reset value	0	0	0	0	0	0	0																				0					0
0x70	SCU_LVRCR																	LVREN[7:0]											LVRAON			LVRVS[1:0]	
	Reset value																	0	0	0	0	0	0	0	0				0			0	0
0x80	SCU_EOSCR							LSEISEL[1:0]									LSENFEN							HSEISEL[1:0]					HSENFEN			HSENFSEL[1:0]	
	Reset value							0	1								1							0	0				0			1	1
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

Table 32. SCU Register Map Summary (continued)

Offset	Register name	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0x98	SCU_MCCR3						TIMER59CSEL[2:0]			TIMER59CDIV[7:0]													TIMER04CSEL[2:0]			TIMER04CDIV[7:0]							
	Reset value						0	0	0	0	0	0	0	0	0	0	0						0	0	0	0	0	0	0	0	0	0	0
0x9C	SCU_MCCR4						PGADCSEL[2:0]			PGADCDIV[7:0]																							
	Reset value						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																						FRTCSEL[2:0]			FRTCDIV[7:0]							
	Reset value																						0	0	0	0	0	0	0	0	0	0	0
0xA8	SCU_MCCR7						UARTCSE[2:0]			UARTCDIV[7:0]																							
	Reset value						0	0	0	0	0	0	0	0	0	0	1																
0xF0	SCU_SYSTEN																								ENS	SYSTEN[7:0]							
	Reset value																								0	0	0	0	0	0	0	0	0

5. PCU and GPIO

5.1 Introduction

5.1.1 PCU Introduction

A31C14x and A31C12x series has a Port Control Unit (PCU) module that controls the external 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 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 six 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 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:

- PB4: SWCLK with pull-down
- PB5: SWDIO with pull-up
- PC9: nRESET with pull-up
- PB3: nBOOT with pull-up

Following figures (Figure 44, Figure 45, and Figure 46) show block diagrams of GPIO and external interrupt I/O pins, respectively.

Figure 44. PCU and GPIO Block Diagram

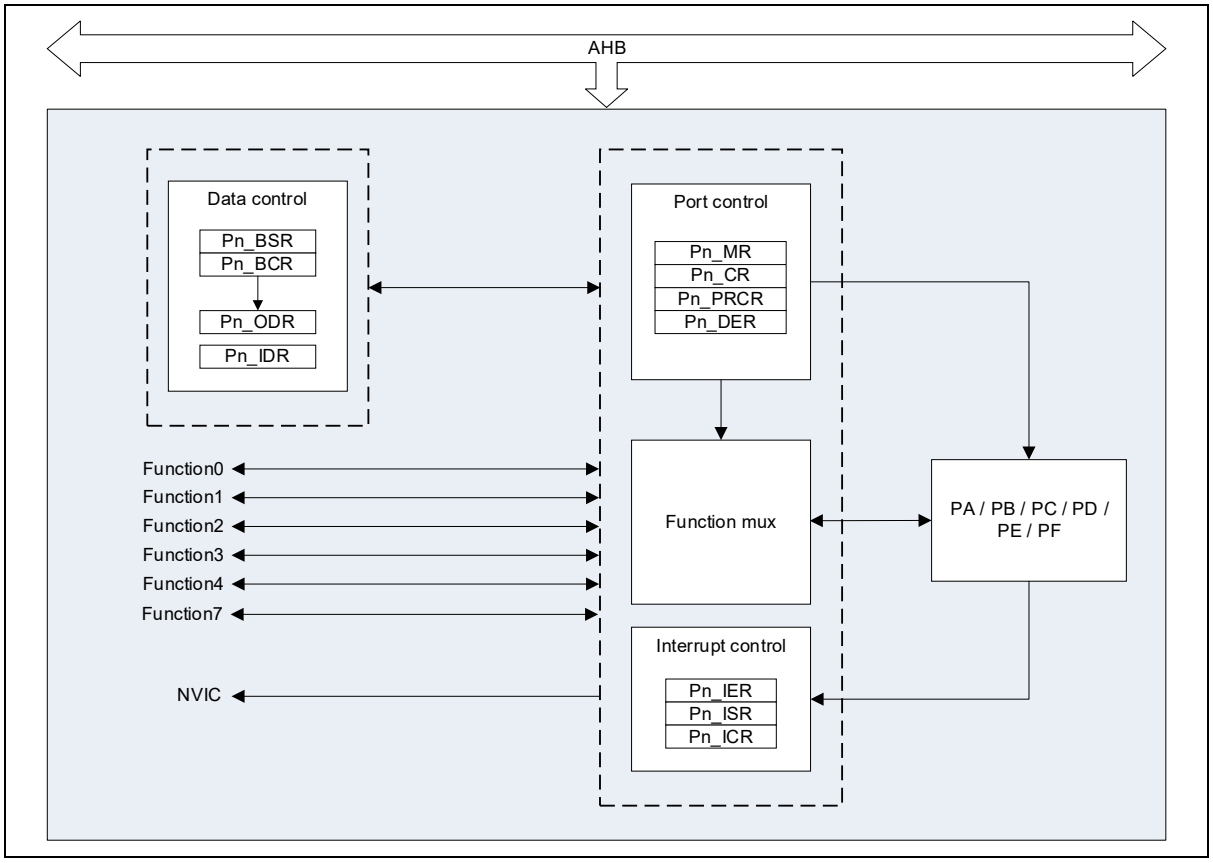


Figure 45. I/O Port Block Diagram (GPIOs)

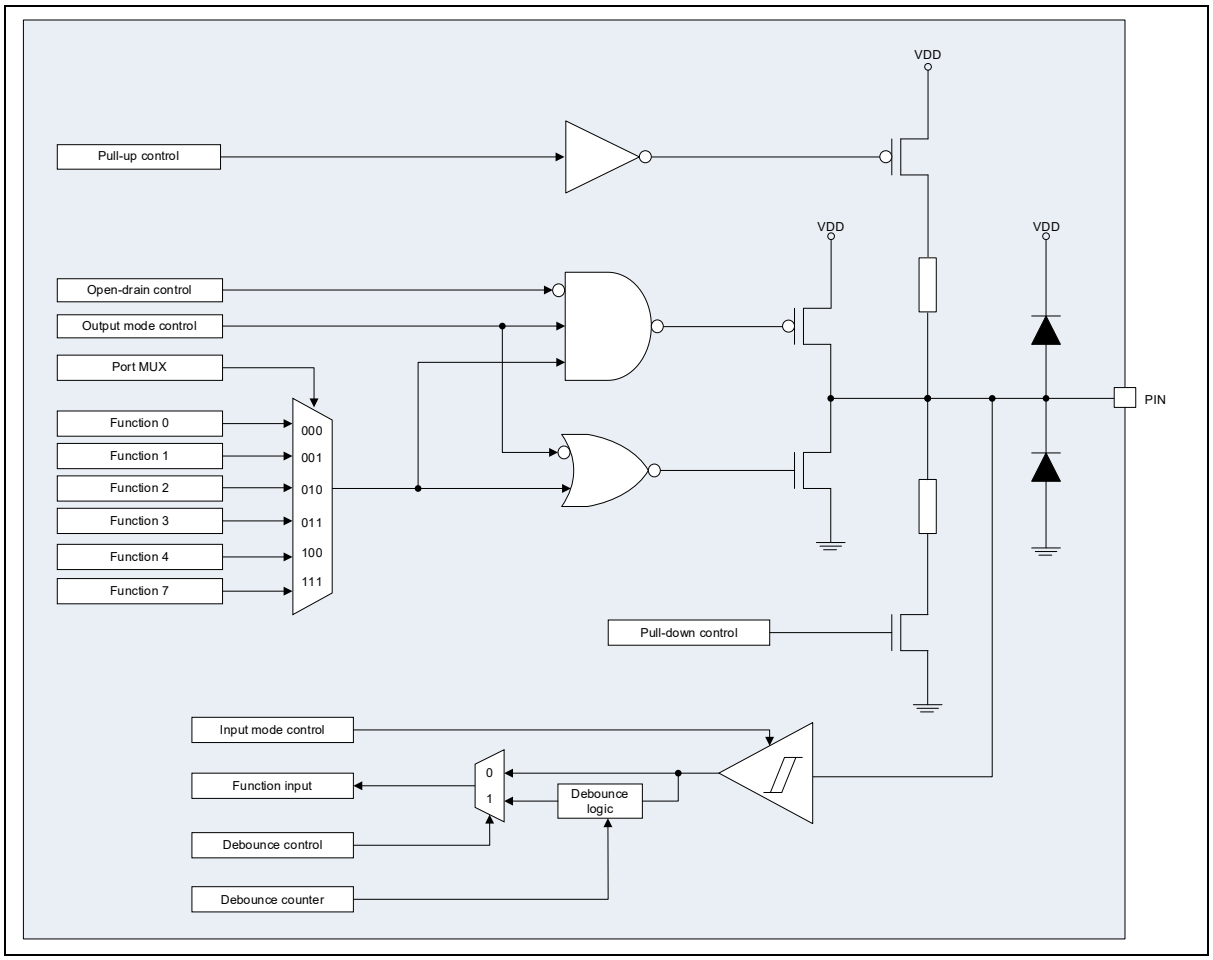
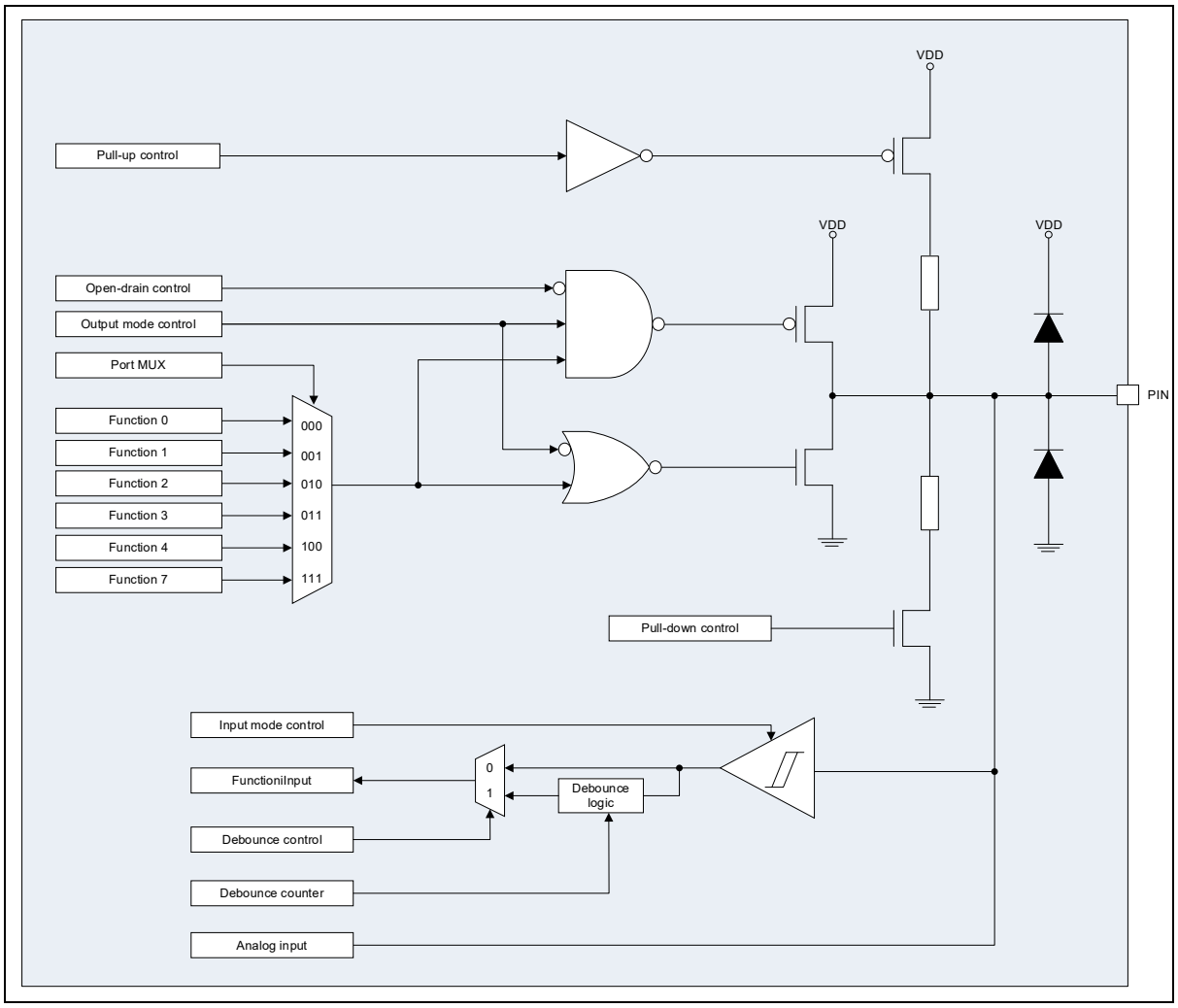


Figure 46. I/O Port Block Diagram (ADC, CMP, and External Oscillator Pins)



5.3.1 PCU and GPIO Pins and Internal Signals

Table 33 describes PCU and GPIO internal signals.

Table 33. PCU and GPIO Internal Signal

Signal Name	Signal Type	Description
HCLK	Clock	AHB Clock

Table 34 describes seven pins assigned for PCU and GPIO module.

Table 34. PCU and GPIO Pins

Pin Name	Type	A31C14x	A31C12x
PA	IO	PA0 to PA11	PA2 to PA7
PB	IO	PB0 to PB11	PB0 to PB5
PC	IO	PC0 to PC9	PC0, PC9
PD	IO	PD0 to PD7	PD0 to PD3
PE	IO	PE0 to PE11	PE0 to PE7
PF	IO	PF0 to PF7	PF0, PF1

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 and pull-down resistors of all ports I/Os.

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

NOTE:

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

5.3.3 I/O Port Data Registers

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

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

5.3.5.1 A31C14x

Table 35 describes the GPIO alternate functions.

Table 35. GPIO Alternate Function (A31C14x) (1)

Pin Name	Alternate Function					
	AF0	AF1	AF2	AF3	AF4	AF7
PA0	PA0	SDA1				AN0*
PA1	PA1	SCL1				AN1*
PA2	PA2		T6IO			AN2/CPRE0*
PA3	PA3					AN3/CP0A*
PA4	PA4					AN4/CPB*
PA5	PA5		T6IO			AN5/CP1A*
PA6	PA6					AN6/CPRE1*
PA7	PA7					AN7/AVREF*
PA8	PA8					AN11*
PA9	PA9					AN12*
PA10	PA10					AN13*
PA11	PA11					*
PB0	PB0	TXD10			MOSI10	AN8*
PB1	PB1	RXD10		EXT_SYNC	MISO10	AN9*
PB2	PB2			CLKO	SCK10	AN10*
PB3	PB3			nBOOT*	SS10	
PB4	PB4	TXD0		SWCLK*		
PB5	PB5	RXD0		SWDIO*		
PB6	PB6	TXD1				*
PB7	PB7	RXD1				*
PB8	PB8		T9IO			*
PB9	PB9		T8IO			*
PB10	PB10		T9IO			*
PB11	PB11		T8IO			*
PC0	PC0		T5IO			*
PC1	PC1		T7IO			*
PC2	PC2		T5IO			*
PC3	PC3		T7IO			*
PC4	PC4					*
PC5	PC5	SDA2				*
PC6	PC6	SCL2				*
PC7	PC7					*
PC8	PC8					*
PC9	PC9			nRESET*		
PD0	PD0	SCL0				
PD1	PD1	SDA0				
PD2	PD2	TXD11			MOSI11	*
PD3	PD3	RXD11			MISO11	*
PD4	PD4				SCK11	*
PD5	PD5				SS11	*
PD6	PD6		T1IO			*
PD7	PD7		T0IO			*

Table 35. GPIO Alternate Function (A31C14x) [1] (continued)

Pin Name	Alternate Function					
	AF0	AF1	AF2	AF3	AF4	AF7
PE0	PE0		T2IO		CP0OUT	AN14*
PE1	PE1		T3IO		CP1OUT	AN15*
PE2	PE2		T4IO			AN16
PE3	PE3		T2IO			AN17
PE4	PE4		T3IO			*
PE5	PE5		T4IO			*
PE6	PE6		T0IO	WDTO		*
PE7	PE7		T1IO	STBYO		*
PE8	PE8	TXD12			MOSI12	*
PE9	PE9	RXD12			MISO12	*
PE10	PE10				SCK12	*
PE11	PE11				SS12	*
PF0	PF0	SCL1				XOUT*
PF1	PF1	SDA1				XIN*
PF2	PF2					SXIN*
PF3	PF3					SXOUT*
PF4	PF4			CLKO		*
PF5	PF5		T2IO			*
PF6	PF6		T3IO			*
PF7	PF7		T4IO			*

NOTES:

1. Unused pins are set to output from firmware (low output is recommended).

* 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 64-pin standard. Therefore, care must be taken when using these pins to ensure proper operation.

5.3.5.2 A31C12x

Table 36 describes the GPIO alternate functions.

Table 36. GPIO Alternate Function (A31C12x) (1)

Pin Name	Alternate Function					
	AF0	AF1	AF2	AF3	AF4	AF7
PA2	PA2		T6IO			AN2*
PA3	PA3					AN3*
PA4	PA4					AN4*
PA5	PA5		T6IO			AN5*
PA6	PA6					AN6*
PA7	PA7					AN7/AVREF*
PB0	PB0	TXD10			MOSI10	AN8*
PB1	PB1	RXD10		EXT_SYNC	MISO10	AN9*
PB2	PB2			CLKO	SCK10	AN10*
PB3	PB3			nBOOT*	SS10	
PB4	PB4	TXD0		SWCLK*		
PB5	PB5	RXD0		SWDIO*		
PB6	PB6	TXD1				*
PC0	PC0		T5IO			*
PC9	PC9			nRESET*		
PD0	PD0	SCL0				*
PD1	PD1	SDA0				*
PD2	PD2	TXD11			MOSI11	*
PD3	PD3	RXD11			MISO11	*
PE0	PE0		T2IO			AN14*
PE1	PE1		T3IO			AN15*
PE2	PE2		T4IO			AN16*
PE3	PE3		T2IO			AN17*
PE4	PE4		T3IO			*
PE5	PE5		T4IO			*
PE6	PE6		T0IO	WDTO		*
PE7	PE7		T1IO	STBYO		*
PF0	PF0	SCL1				XOUT*
PF1	PF1	SDA1				XIN*
PF2	PF2					SXIN*
PF3	PF3					SXOUT*

NOTES:

1. Unused pins are set to output from firmware (low output is recommended).

* 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 32-pin standard.

5.3.6 Interrupt Functionality

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

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

Table 37. Interrupt Sources and Corresponding Pins in GPIO Module

Interrupt Name	Configurable Pins	
	A31C14x	A31C12x
GPIOA, GPIOB	Port A[11:0], Port B[11:0]	Port A[7:2], Port B[5:0]
GPIOC, GPIOD	Port C[9:0], Port D[7:0]	Port C[0],[9], Port D[3:0]
GPIOE, GPIOF	Port E[11:0], Port F[7:0]	Port E[11:7], Port F[1:0]

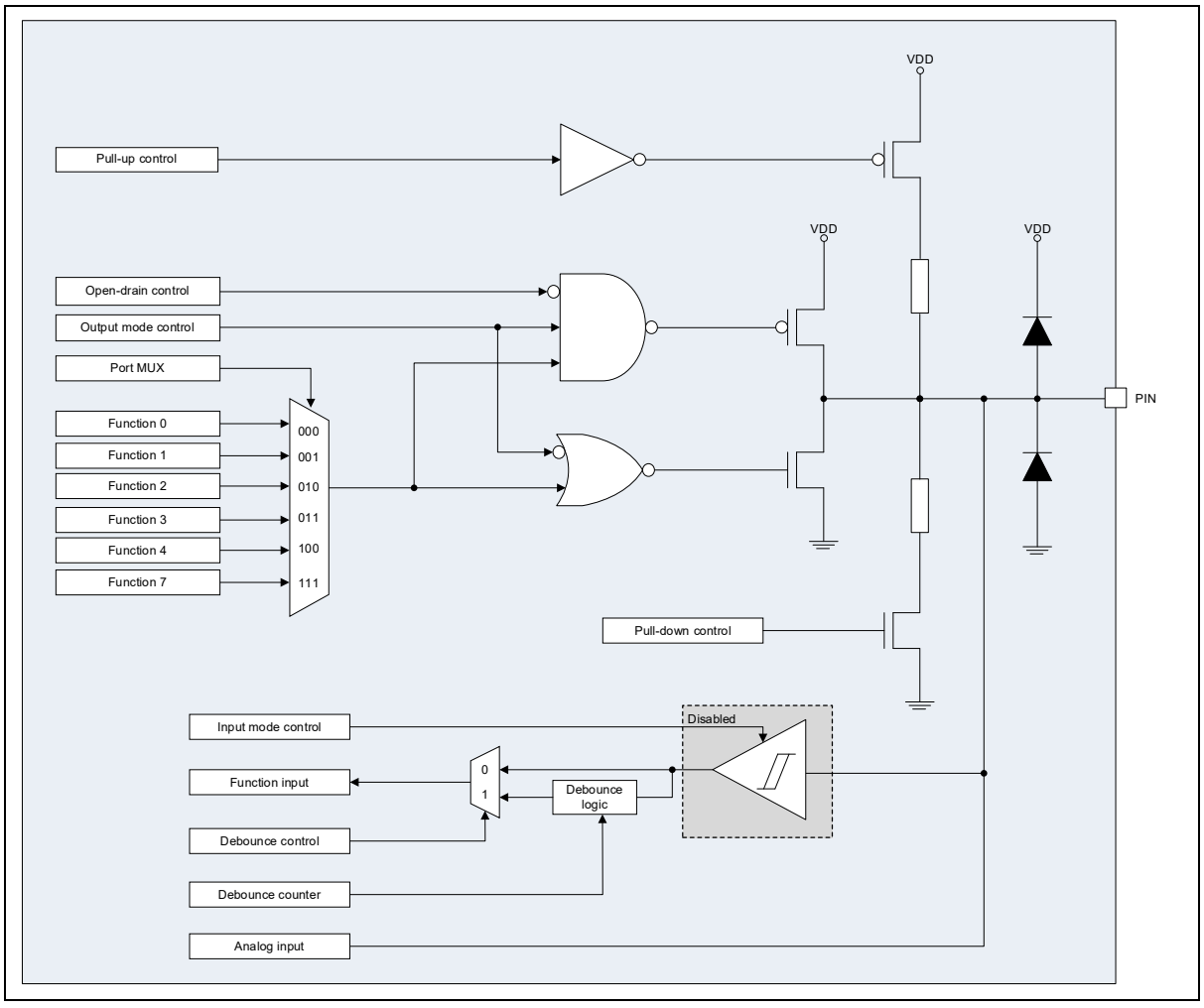
5.3.7 Output Configuration

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.
- 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 47 shows a block diagram of an output port.

Figure 47. Output Port Block Diagram



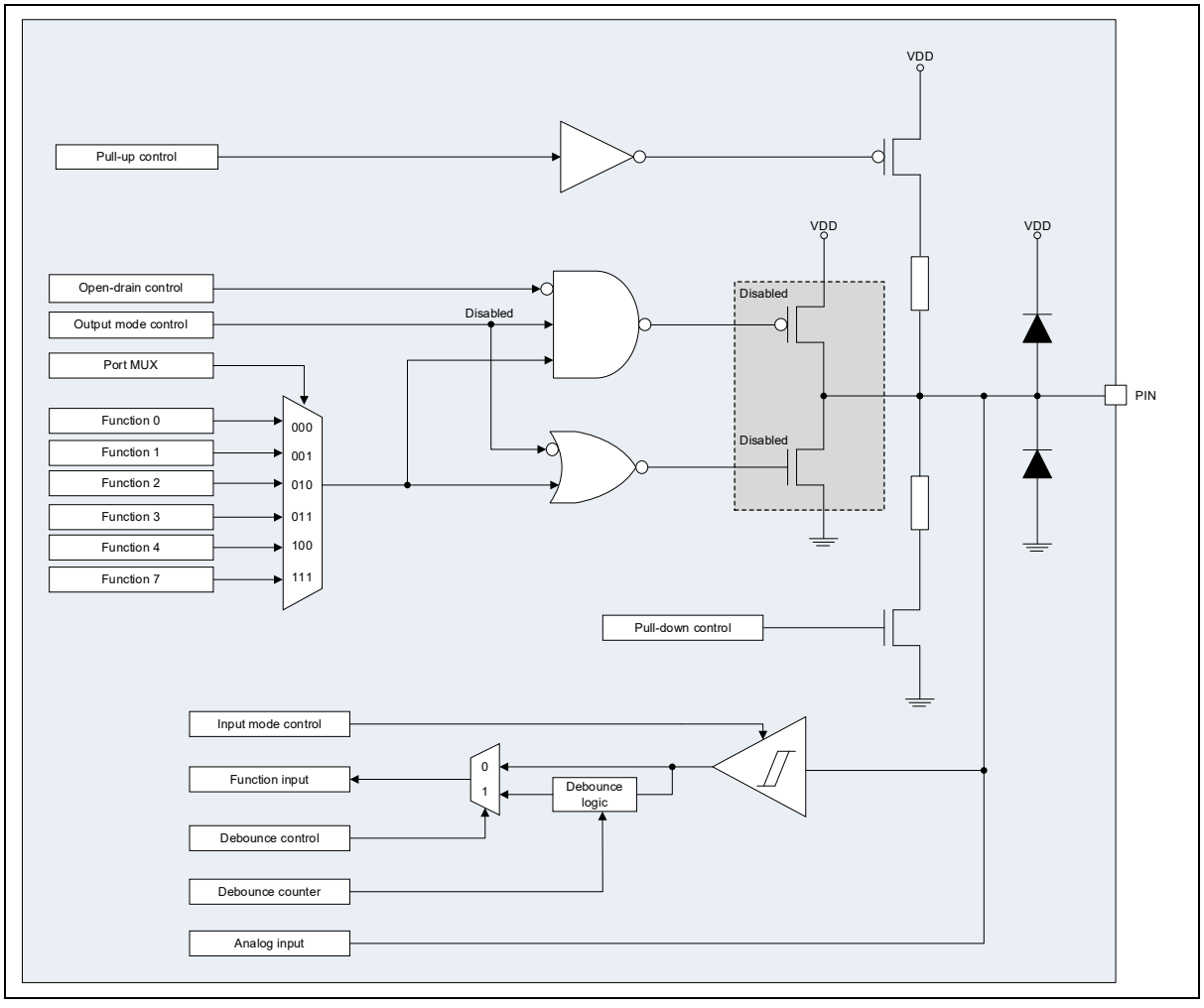
5.3.8 Input Configuration

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 48 shows a block diagram of an input port.

Figure 48. Input Port Block Diagram



5.3.9 Alternate Function Configuration

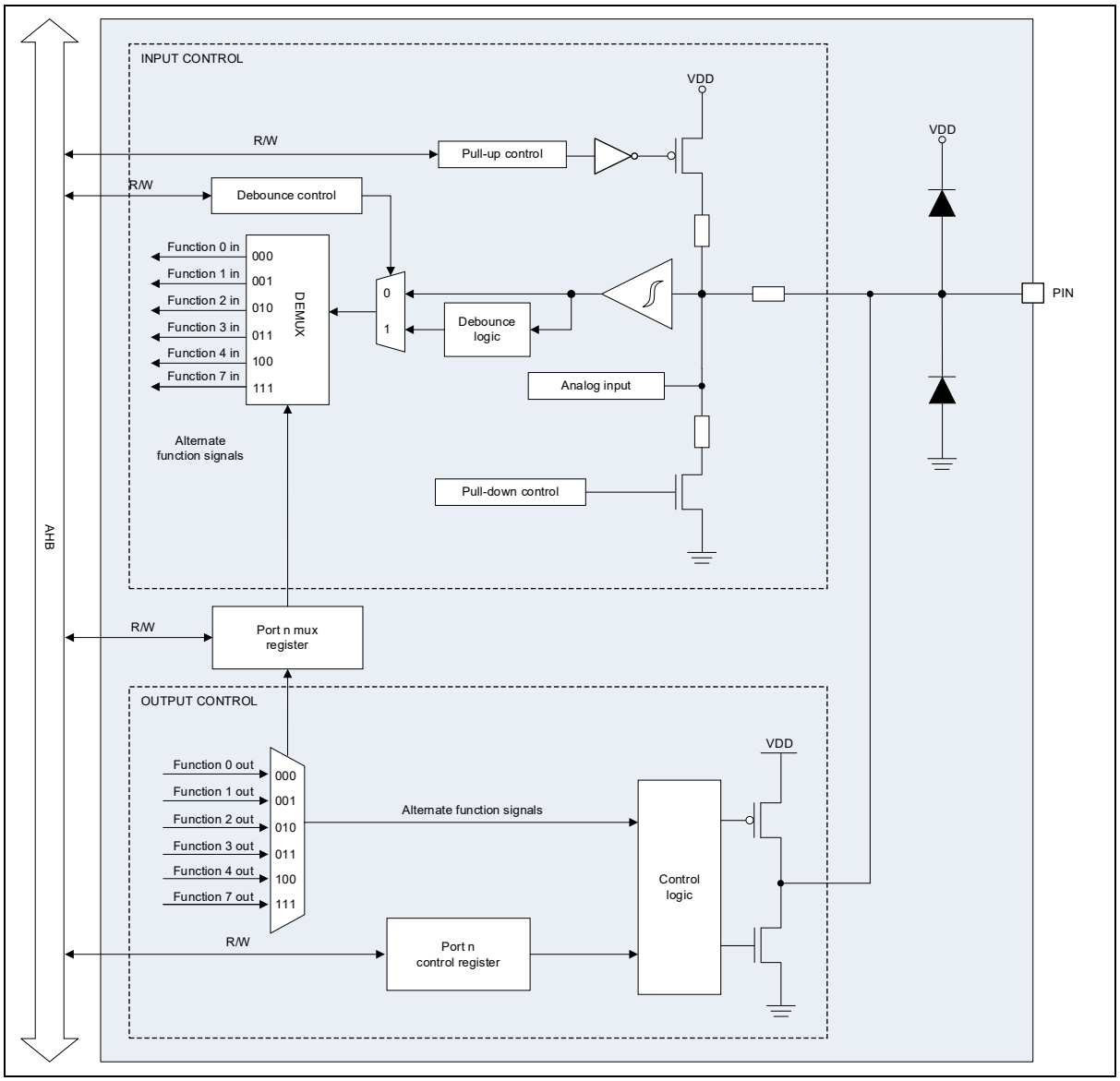
For the A31C14x and A31C12x series, all pins except those with specific purposes can be used as GPIO pins, and the GPIO function can be set by the pin mux control register on each port.

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

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

Figure 49 shows a block diagram that describes alternate function selection on a port.

Figure 49. Port Function Block Diagram



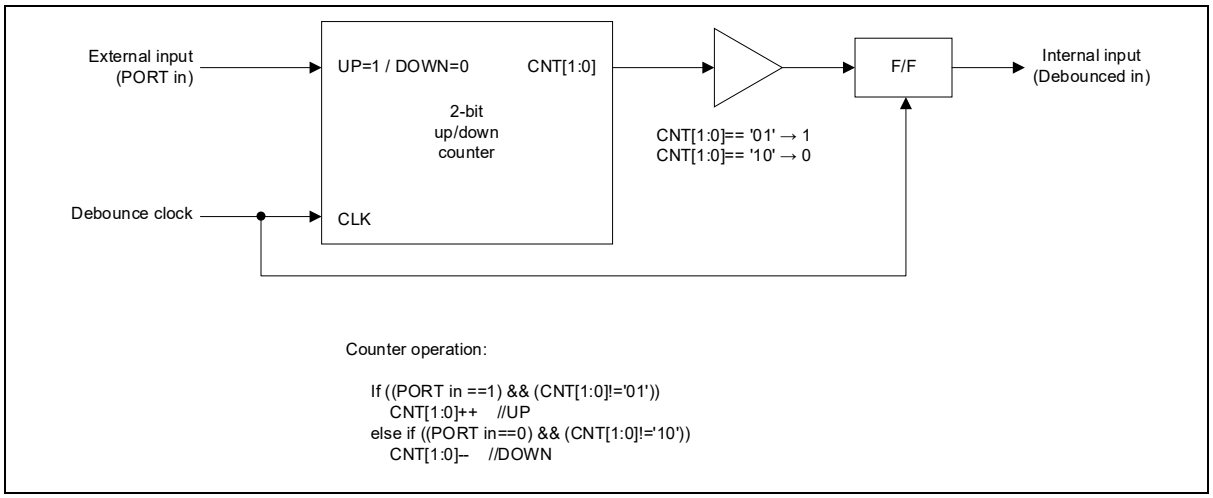
5.3.10 Debouncing Functionality

For the A31C14x and A31C12x series, 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 50 shows a block diagram of simplified debouncing logic.

Figure 50. Debouncing Logic Block Diagram

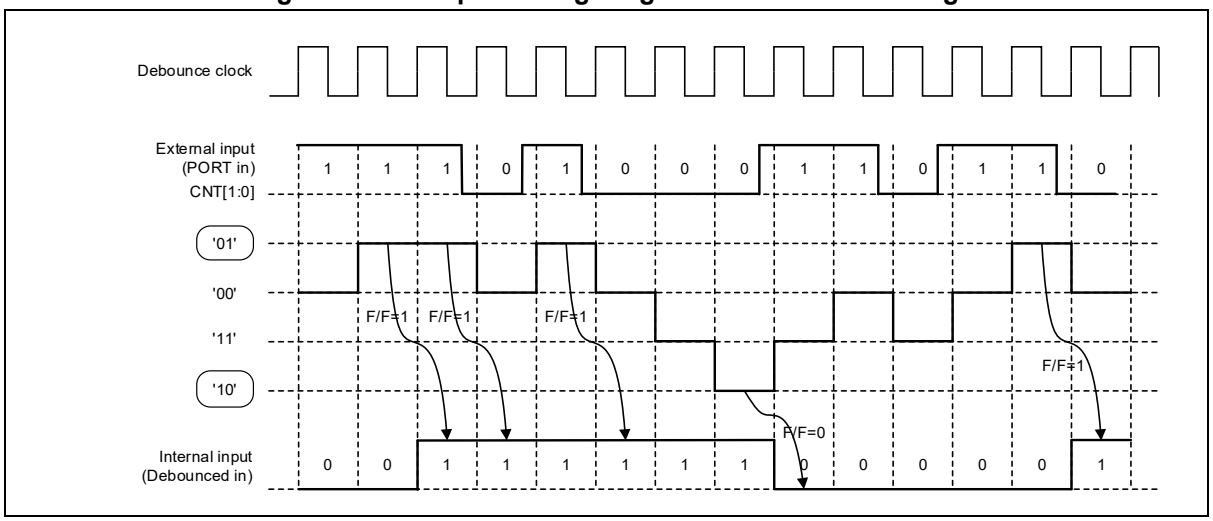


When enabled, the debouncing logic is clocked at the frequency division ratio set in the SCU_MCCR4 or SCU_MCCR5 registers, depending on which register the port belongs to.

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 51 shows a timing diagram of a port debouncing.

Figure 51. Example Timing Diagram of Port Debouncing



5.4 PCU and GPIO Registers

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

Table 38. Base Address of PCU and GPIO

Name	Base Address	Description
PA	0x4009_0000	Port A
PB	0x4009_0100	Port B
PC	0x4009_0200	Port C
PD	0x4009_0300	Port D
PE	0x4009_0400	Port E
PF	0x4009_0500	Port F

Table 39. PCU and GPIO Register Map

Name	Offset	Type	Description	Reset value	Ref.
Pn_MR1	0x0000	RW	Port n MUX1 Select Register	0x7777_7777 PB: 0x7733_3777 PD: 0x7777_7700 PE: 0x7777_0077	5.4.1
Pn_MR2	0x0004	RW	Port n MUX2 Select Register	PA, PB, PE: 0x0000_7777 PC : 0x0000_0037 PD, PF: 0x0000_0000	5.4.2
Pn_CR	0x0008	RW	Port n Control Register	PA, PB, PE: 0x00FF_FFFF PC: 0x000F_FFFF PD, PF: 0x0000_FFFF	5.4.3
Pn_PRCR	0x000C	RW	Port n Pull-up/pull-down Control Register	0x0000_0000 PB: 0x0000_0B80 PC: 0x0008_0000 PD: 0x0000_000A PE: 0x0000_00A0	5.4.4
Pn_DER	0x0010	RW	Port n Debouncing Enable Register	0x0000_0000	5.4.5
Pn_IER	0x0020	RW	Port n Interrupt Enable Register	0x0000_0000	5.4.6
Pn_ISR	0x0024	RWC 1	Port n Interrupt Status Register	0x0000_0000	5.4.7
Pn_ICR	0x0028	RW	Port n Interrupt Control Register	0x0000_0000	5.4.8
Pn_ODR	0x0030	RW	Port n Output Data Register	0x0000_0000	5.4.9
Pn_IDR(2)	0x0034	RO	Port n Input Data Register	PA, PB, PE: 0x0000_0FFF PC: 0x0000_03FF PD, PF: 0x0000_00FF	5.4.10
Pn_BSR	0x0038	WO	Port n Set/Reset Register	0x0000_0000	5.4.11
Pn_BCR	0x003C	WO	Port n Reset Register	0x0000_0000	5.4.12
PCU_RSTDBCR	0x0F0C	RW	Pin Reset Debounce Control Register	0x0000_3C01	5.4.13
PCU_PORTEN	0x0FF0	RW	Port Access Enable Register	0x0000_0000	5.4.15

NOTES:

1. n = A, B, C, D, E and F.
2. The initial values for the GPIO of A31C12x are as follows. PD_MR1/PE_MR1: 0x7777_7777, PD_PRCR/PE_PRCR: 0x0000_0000, PA_IDR: 0x0000_00FC, PB_IDR: 0x0000_006F, PC_IDR: 0x0000_0201, PD_IDR/PF_IDR: 0x0000_000F, PE_IDR: 0x0000_00FF. All other initial values are the same as above.

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=0x4009_0000, PB_MR1=0x4009_0100
PC_MR1=0x4009_0200, PD_MR1=0x4009_0300
PE_MR1=0x4009_0400, PF_MR1=0x4009_0500

31	30 29 28	27	26 25 24	23	22 21 20	19	18 17 16	15	14 13 12	11	10 9 8	7	6 5 4	3	2 1 0
Reserved	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 multiplexer selection bit, x=0 to 7.	
		000	Alternate function 0 (AF0)
		001	Alternate function 1 (AF1)
		010	Alternate function 2 (AF2)
		011	Alternate function 3 (AF3)
		100	Alternate function 4 (AF4)
		111	Alternate function 5 (AF7)
		Others	Reserved

NOTES:

1. The PxMUX[2:0] (x = 1 to 15) bits of PF0, PF1, PF2 and PF3 cannot be modified while the HSE or LSE is set as the system clock (MCLK).
2. If users select pin multiplexing for a pin, the corresponding I/O settings are automatically modified to meet the users' selection. (If Alternate Function is applied according to the Pn_MR register settings, I/O is automatically set as the input or output of the function.)
3. When pin multiplexing is set to AF1, AF2, AF3, AF4 and AF7 it cannot be set to open-drain output or input even if users use the Pn_CR register.
4. When pin multiplexing is set to AF0, users can use the Pn_CR register to set a port.
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=0x4009_0004, PB_MR2=0x4009_0104
PC_MR2=0x4009_0204, PD_MR2=0x4009_0304
PE_MR2=0x4009_0404, PF_MR2=0x4009_0504

31	30 29 28	27	26 25 24	23	22 21 20	19	18 17 16	15	14 13 12	11	10 9 8	7	6 5 4	3	2 1 0
Reserved	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 multiplexer 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)
		111	Alternate function 5 (AF7)
		Others	Reserved

NOTES:

1. The PxMUX[2:0] (x = 1 to 15) bits of PF0, PF1, PF2 and PF3 cannot be modified while the HSE or LSE is set as the system clock (MCLK).
2. If users select pin multiplexing for a pin, the corresponding I/O settings are automatically modified to meet the users' selection. (If Alternate Function is applied according to the Pn_MR register settings, I/O is automatically set as the input or output of the function.)
3. When pin multiplexing is set to AF1, AF2, AF3, AF4 and AF7 it cannot be set to open-drain output or input even if users use the Pn_CR register.
4. When pin multiplexing is set to AF0, users can use the Pn_CR register to set a port.
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=0x4009_0008, PB_CR=0x4009_0108
PC_CR=0x4009_0208, PD_CR=0x4009_0308
PE_CR=0x4009_0408, PF_CR=0x4009_0508

31 30	29 28	27 26	25 24	23 22	21 20	19 18	17 16	15 14	13 12	11 10	9 8	7 6	5 4	3 2	1 0
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=0x4009_000C, PB_PRCR =0x4009_410C
PC_PRCR =0x4009_420C, PD_PRCR =0x4009_430C
PE_PRCR =0x4009_440C, PF_PRCR =0x4009_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 and pull-down resistors.
		10	Enables the pull-up resistor and disables the pull-down resistor.
		11	Disables the pull-up resistor and enables the pull-down resistor.

NOTES:

1. 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[7:0].

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=0x4009_0010, PB_DER=0x4009_0110
PC_DER=0x4009_0210, PD_DER=0x4009_0310
PE_DER=0x4009_0410, PF_DER=0x4009_0510

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

x	PDEx	Enable 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.
3. When PC9 is configured as AF3 (nRESET), the settings in the PCU_RSTDBCR register are applied. When configured as AF0 (GPIO), the settings in the PC_DER register are applied.

5.4.6 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=0x4009_0020, PB_IER=0x4009_0120
PC_IER=0x4009_0220, PD_IER=0x4009_0320
PE_IER=0x4009_0420, PF_IER=0x4009_0520

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

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

NOTES:

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

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

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

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

3. Pn_DER, Pn_STR, Pn_IER, Pn_ISR, Pn_ICR, Pn_ODR, Pn_IDR, Pn_BSR and Pn_BCR registers can change the value of the register without activating PORTEN.

5.4.7 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=0x4009_0024, PB_ISR=0x4009_0124
PC_ISR=0x4009_0224, PD_ISR=0x4009_0324
PE_ISR=0x4009_0424, PF_ISR=0x4009_0524

31 30	29 28	27 26	25 24	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].

5.4.8 Pn_ICR: Port n Interrupt Control Register

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

PA_ICR=0x4009_0028, PB_ICR=0x4009_0128
PC_ICR=0x4009_0228, PD_ICR=0x4009_0328
PE_ICR=0x4009_0428, PF_ICR=0x4009_0528

31 30	29 28	27 26	25 24	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].

5.4.9 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=0x4009_0030, PB_ODR=0x4009_0130
PC_ODR=0x4009_0230, PD_ODR=0x4009_0330
PE_ODR=0x4009_0430, PF_ODR=0x4009_0530

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

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

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_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=0x4009_0034, PB_IDR=0x4009_0134
PC_IDR=0x4009_0234, PD_IDR=0x4009_0334
PE_IDR=0x4009_0434, PF_IDR=0x4009_0534

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

15 0	PIDx	The pin's input data (represents the port's 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].

5.4.11 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=0x4009_0038, PB_BSR=0x4009_0138
PC_BSR=0x4009_0238, PD_BSR=0x4009_0338
PE_BSR=0x4009_0438, PF_BSR=0x4009_0538

31	30	29	28	27	26	25	24	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].

5.4.12 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=0x4009_003C, PB_BSR=0x4009_013C
PC_BSR=0x4009_023C, PD_BSR=0x4009_033C
PE_BSR=0x4009_043C, PF_BSR=0x4009_053C

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

5.4.13 PCU_RSTDBCR: Port Reset Debounce Control Register

Pin Reset Debounce Control Register.

PCU_RSTDBCR=0x4009_0F0C

31 30 29 28 27 26 25 24	23 22 21 20 19 18 17 16	15 14 13 12 11 10 9 8	7 6 5 4 3 2 1	0
WTDIKEY[31:16]		CLK_CNT[7:0]	Reserved	EN
-		0x3C	-	1
RW		RW	-	RW

31 16	WTDIKEY[31:16]	Write Identification Key On Writes, write 0x0514 to these bits. Otherwise, writing is ignored.
15 8	CLK_CNT[7:0]	Noise cancelling delay option for selected debounce clock N N clock checking for debounce by selected debounce clock. Default filter is set '1.875ms' with LSI (32kHz) debounce clock.
0	EN	Pin reset debounce enable 0 Disables 1 Enables

NOTES:

1. If users want to operate pin reset debounce, user must enable LSI (32kHz). Because pin reset debounce use LSI clock source.
2. When PC9 is configured as AF3 (nRESET), the settings in the PCU_RSTDBCR register are applied. When configured as AF0 (GPIO), the settings in the PC_DER register are applied.
3. The pin reset debounce function is not applied when the A31C14x enters DEEP-SLEEP mode.

5.4.14 PCU_PIN_REMAP: Pin Remapping Register

The PCU_PIN_REMAP register controls remapping. The PCU_PIN_REMAP register controls pin remapping and is available only for product A31C12x. This register is required when using the 32-QFN, 32-LQFP or 28-TSSOP package of product A31C12x. When using the 32-QFN or 32-LQFP package, the PIN_REMAP bit must be set to 0 to enable the use of port PC0. When using the 28-TSSOP package, the PIN_REMAP bit must be set to '1' to enable the use of port PB6.

PCU_PIN_REMAP=0x4009_0F10

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

31 16	PIN_REMAP_EN [15:0]	Writing '0x5A0F' to the bit field enables an update to the PIN_REMAP bit.
0	PIN_REMAP	Remapping selection *To use it, 0x5A0F must also be written to bit [31:16] of PCU_PIN_REMAP
		0 Use PC0 pin
		1 Use PB6 pin

5.4.15 PCU_PORTEN: Port Access Enable Register

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

PCU_PORTEN=0x4009_0FF0

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

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

NOTE:

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

5.4.16 PCU and GPIO Register Map Summary

Table 40. 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_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 40. 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_RSTDBCR	WTDIKEY[31:16]																CLK_CNT[7:0]								Res.	Res.	Res.	Res.	Res.	Res.	Res.	EN
Reset value	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	1	1	1	0	0								1
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 A31C14x and A31C12x series 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 A31C14x and A31C12x series has several key features as listed below:

- A31C14x series
  - 128 KB or 64 KB of code flash memory with write protection bits.
- A31C12x series
  - 32 KB or 16 KB of code flash memory with write protection bits.
- Erase unit:
  - Page (512 bytes)
  - Sector (2 KB)
  - Bulk (full chip)
- Program unit:
  - Word (4 bytes)
- Zero wait (less than 25 MHz), 1 to 15-wait
- Code flash memory read protection
- Code flash memory write protection

6.3 Flash Functional Description

6.3.1 Flash Memory Organization

Flash memory consists of code flash memory. Code flash stores the application code or constant data.

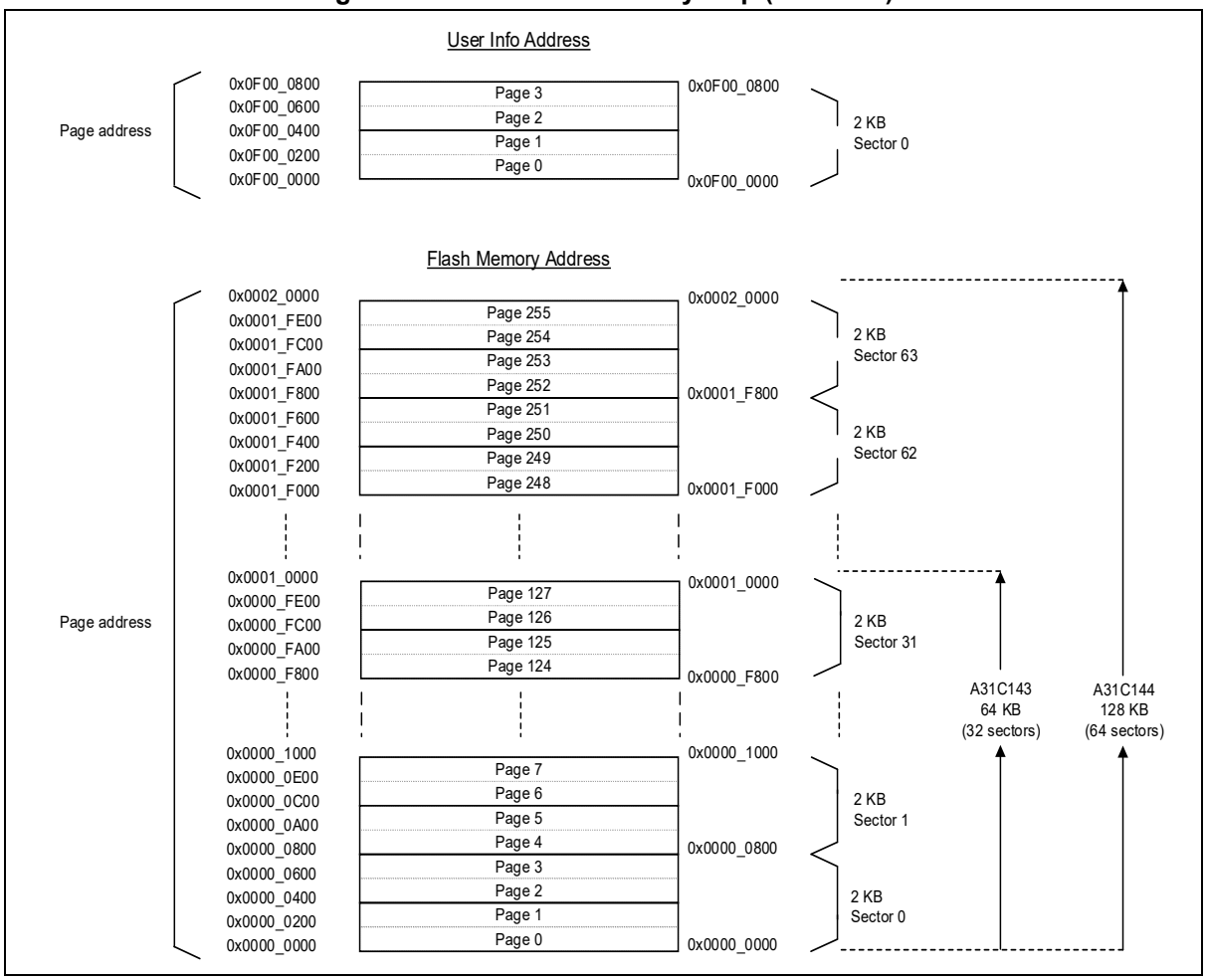
6.3.1.1 Code Flash Memory (A31C14x)

Code flash memory has key features as shown below:

Table 41. Code Flash Memory Controller Features (A31C14x)

Item	A31C143	A31C144
Size	64 KB	128 KB
Start address	0x0000_0000	0x0000_0000
End address	0x0000_FFFF	0x0001_FFFF
Page size	512 bytes	
Total page count	128 page	256 pages
Program unit	4 bytes (1 word)	
Erase unit	512 bytes, 2 KB, Bulk (full chip)	

Figure 52. Code Flash Memory Map (A31C14x)



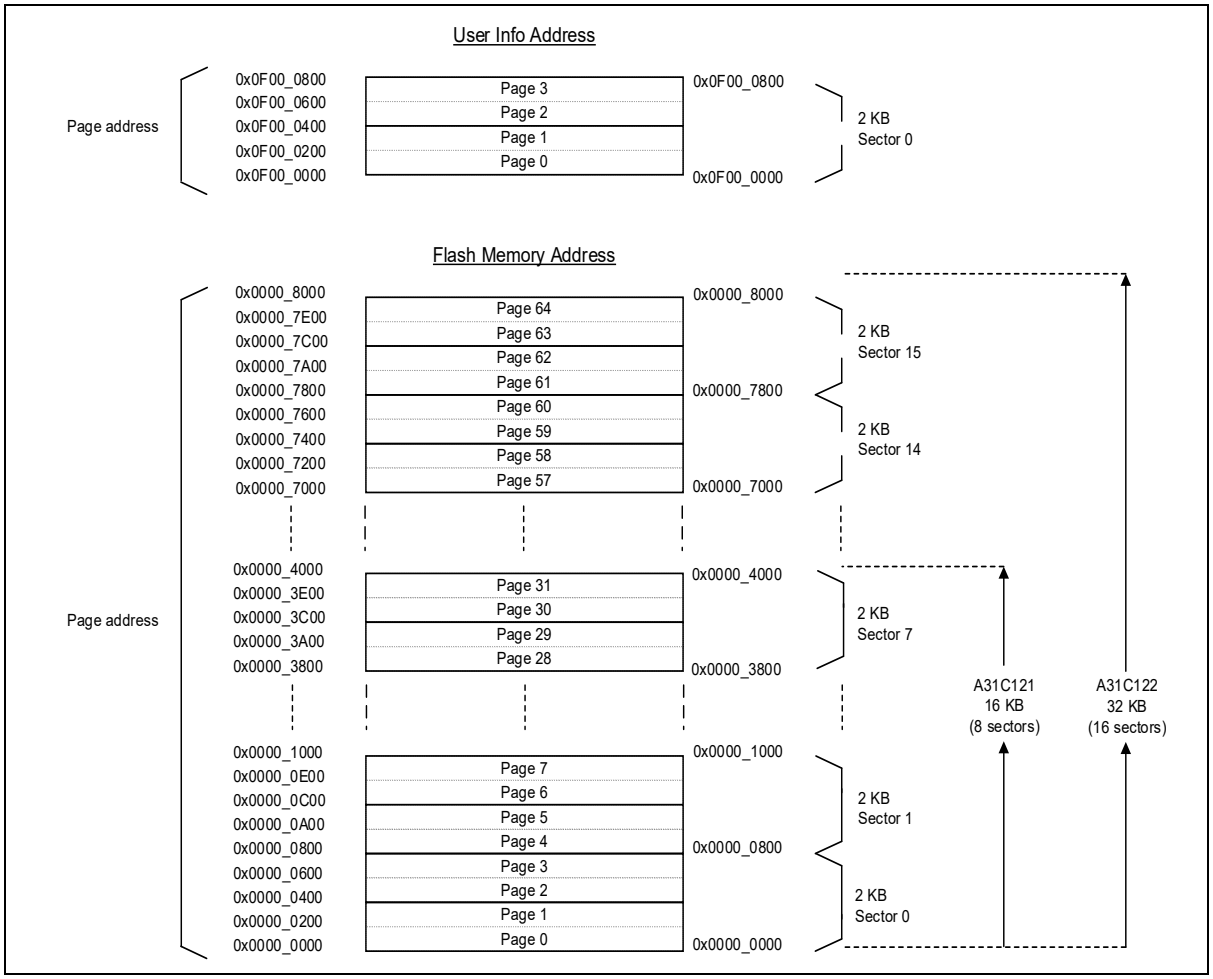
6.3.1.2 Code Flash Memory (A31C12x)

Code flash memory has key features as shown below:

Table 42. Code Flash Memory Controller Features (A31C12x)

Item	A31C121	A31C122
Size	16 KB	32 KB
Start address	0x0000_0000	0x0000_0000
End address	0x0000_3FFF	0x0000_7FFF
Page size	512 bytes	
Total page count	32 pages	64 pages
Program unit	4 bytes (1 word)	
Erase unit	512 bytes, 2 KB, Bulk (full chip)	

Figure 53. Code Flash Memory Map (A31C12x)



6.3.2 Read Access Latency

Flash memory has a read latency that is the 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 43.
2. Program the LATENCY[3:0] in CFMC_CONF register to have a new wait time.

Table 43. Internal Code Flash Memory Wait Time by Operating Clock

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

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. Two 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 disable 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 user 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. To enable access to the user info memory, enter the key values into the CFMC_INFOKEY register in the order shown below:
   A. In normal input case:
      i. KEY1 → KEY2 → KEY3
   B. In non-consecutive input case:
      i. KEY1 → APB (No KEY) write → KEY2 → APB (No KEY) write → KEY3
2. To disable access to the user info memory, set the INFOLOCK in CFMC_CTRL register to ‘1’.

Read access to the user info 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.

6.3.4 Flash Memory Erase Sequences

6.3.4.1 Page Erase

Figure 54 shows a timing diagram for page erase operations in the flash memory.

Figure 54. Flash Memory Page Erase Timing Diagram

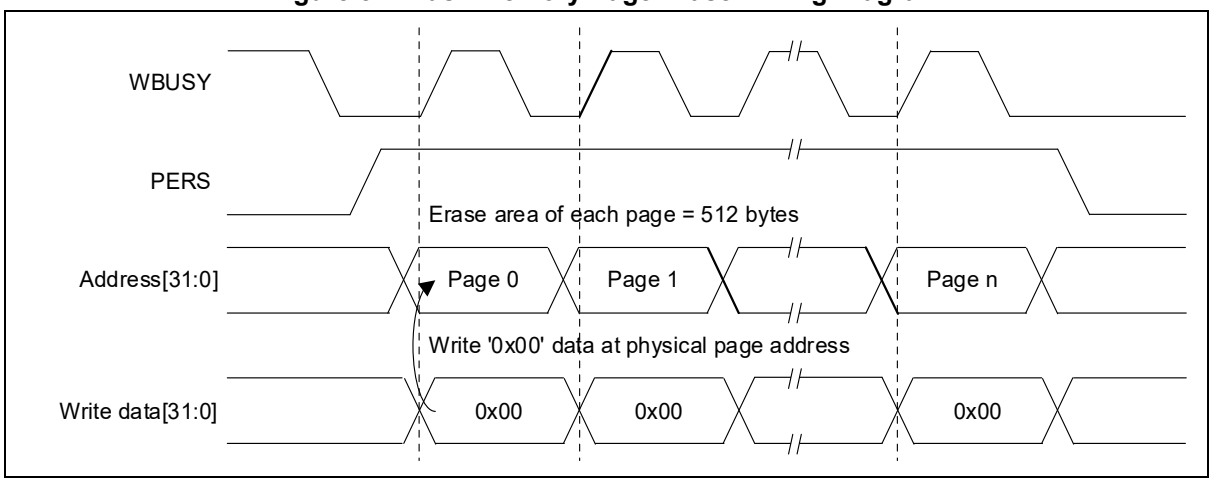


Figure 54 describes the following procedure for the page erase:

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

6.3.4.2 2 KB Sector Erase

Figure 55 shows a timing diagram for 2 KB sector erase operations in the flash memory.

Figure 55. Flash Memory 2 KB Sector Erase Timing Diagram

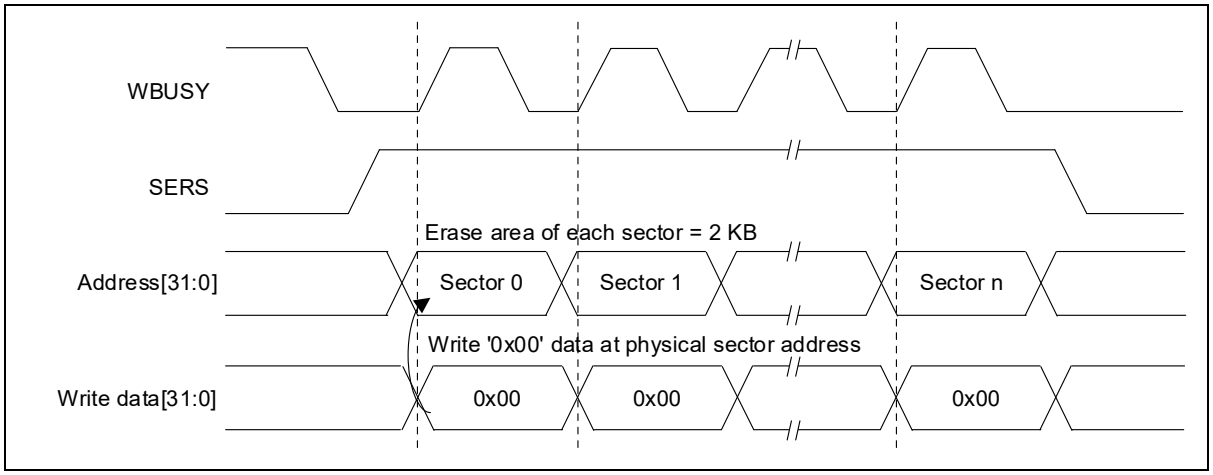


Figure 55 describes the following procedure for the 2 KB sector erase:

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

6.3.4.3 Bulk Erase

Figure 56 shows a timing diagram for bulk (full chip) erase operations in the flash memory.

Figure 56. Flash Memory Bulk (Full-Chip) Erase Timing Diagram

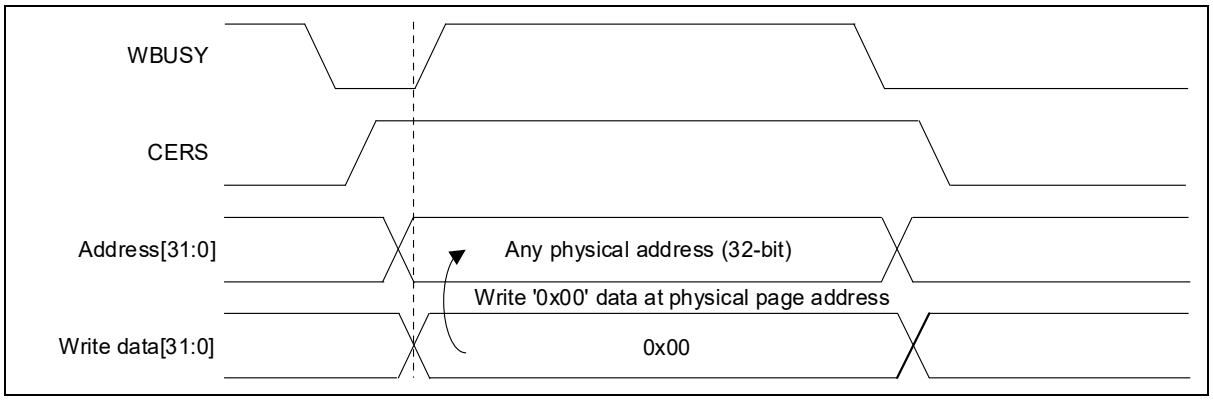


Figure 56 describes the following procedure for the bulk erase.

1. Check the Check the WBUSY flag in the CFMC_STAT register.
2. Set the CERS bit in the CFMC_CTRL register.
3. Write ‘0x00’ data at any physical address.
4. Check the WBUSY flag in the CFMC_STAT register at polling mode or the WDONE flag in the CFMC_STAT register at interrupt mode.
5. Clear the CERS bit in the CFMC_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. Note that lock, protection must be disabled during the writing is complete.

6.3.5.1 Standard Programming

Figure 57. Flash Memory Program Timing Diagram

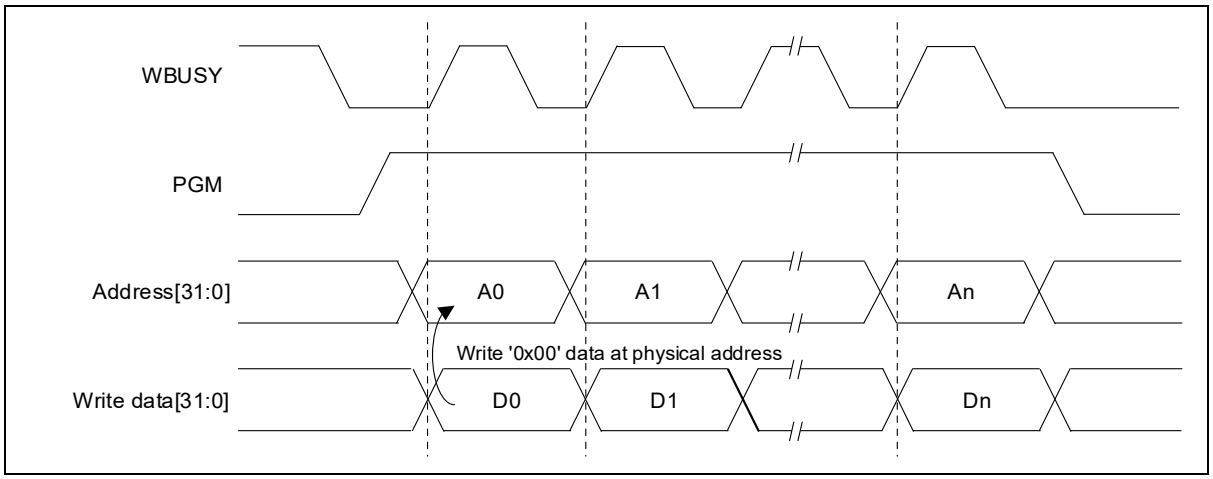


Figure 57 describes the following procedure for the standard programming:

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

6.4 Flash Memory Protection

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

6.4.1 Read Protection

Table 44 describes three different protection mode levels of the read protection, such as UNPROT, LVL1, and LVL2.

Table 44. Available Operating Modes by Protection Level

Protection Level	Operation Mode	Main (Code Flash Memory)		User Info 0		User Info 1/2/3	
		Read	Write	Read	Write	Read	Write
UNPROT	Normal mode	O	O	O	O	O	O
	Debug mode	O	O	O	O	O	O
	Boot mode	O	O	O	O	O	O
LVL1	Normal mode	O	O	O	N/A	O	O
	Debug mode	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
LVL2	Normal mode	O	O	O	N/A	O	O
	Debug mode	This mode cannot be entered					
	Boot mode	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.,) procedure 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 45. Read Protection Operation in UNPROT Level

Normal Mode		Destination Position (DEST)			
		BootROM	Code Flash	SRAM	User Info
Execution position (SRC)	Boot/Code/SRAM	R	All	All	All
	User info	X	X	X	X
	DMA	X	X	R/W	X
	Debug	R	All	All	All

NOTE:

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

6.4.1.2 Read Protection: LVL1 (Level-1)

Debug access is available, and 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 46. Read Protection Operation in Level-1 (LVL 1)

Read Protection Level-1		Destination Position (DEST)			
		BootROM	Code Flash	SRAM	User Info
Execution Position (SRC)	Boot/Code/SRAM	R	All	All	R/W/(E)
	User Info	X	X	X	X
	DMA	X	X	R/W	X
	Debug	X	X	R/W	-/W/(E)

NOTE:

1. 'R': read access, 'W': write access, 'E': erase is possible only by bulk erase the of code flash area (64 KB), 'All' = read, write, and erase access.

6.4.1.3 Read Protection: LVL2 (Level-2)

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

Table 47. Read Protection Operation in Level-2 (LVL2)

Read Protection Level-2		Destination Position (DEST)			
		BootROM	Code Flash	SRAM	User Info
Execution Position (SRC)	Boot/Code/SRAM	R	All	All	R/W/(E)
	User Info	X	X	X	X
	DMA	X	X	R/W	X
	Debug	X	X	X	X

NOTE:

1. 'R': read access, 'W': write access, 'E': erase is possible only by bulk erase of the code flash area (64 KB), 'All' = read, write, and erase access.

6.4.1.4 Read Protection Setting

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

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

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

6.4.1.5 Read Protection Removal

To remove the read protection, users must reset after deleting the related data in the User 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 memory is stored and maintained until reset occurs.

6.4.1.6 Password Settings

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

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

```
CFMC_PWPRST = 0x1234_5678;

CFMC_PWPRST = 0x1234_5678;
```

6.4.2 Write Protection

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

- If any of the FPBY16_n in CFMC_FLSPROT register is ‘1’ (protection), the chip erase does not work because the protected area is included in the erase area.
- Similarly, if some unprotection area is set protection (FUP512_x in CFMC_FLSPROT register is ‘0’) when the FPBY16_15 bit of A31C14x is '1’, the erase commands such as ‘Sector erase’, and ‘Chip erase’ do not work for the area that includes protection area. Likewise, when the FPBY16_14 or FPBY16_15 bit of A31C12x is '1’, the erase commands such as ‘Sector erase’, and ‘Chip erase’ do not work for the area that includes protection area.

6.4.2.1 Write Protection for A31C14x Code Flash Memory

The CFMC_FLSPROT register is used to set the Protection and Unprotection as described in Table 48, Table 49, Table 50, and Table 51.

Table 48. Code Flash Memory Write Protection Area in 8 KB Units (A31C144)

n	Address	n	Address
0	0x0000_0000 ~ 0x0000_1FFF	1	0x0000_2000 ~ 0x0000_3FFF
2	0x0000_4000 ~ 0x0000_5FFF	3	0x0000_6000 ~ 0x0000_7FFF
4	0x0000_8000 ~ 0x0000_9FFF	5	0x0000_A000 ~ 0x0000_BFFF
6	0x0000_C000 ~ 0x0000_DFFF	7	0x0000_E000 ~ 0x0000_FFFF
8	0x0001_0000 ~ 0x0001_1FFF	9	0x0001_2000 ~ 0x0001_3FFF
10	0x0001_4000 ~ 0x0001_5FFF	11	0x0001_6000 ~ 0x0001_7FFF
12	0x0001_8000 ~ 0x0001_9FFF	13	0x0001_A000 ~ 0x0001_BFFF
14	0x0001_C000 ~ 0x0001_DFFF	15	0x0001_E000 ~ 0x0001_FFFF

Table 49. Code Flash Memory Unprotection Area (Top 4 KB) in 512 Bytes Units (A31C144)

x	Address	x	Address
0	0x0001_F000 ~ 0x0001_F1FF	1	0x0001_F200 ~ 0x0001_F3FF
2	0x0001_F400 ~ 0x0001_F5FF	3	0x0001_F600 ~ 0x0001_F7FF
4	0x0001_F800 ~ 0x0001_F9FF	5	0x0001_FA00 ~ 0x0001_FBFF
6	0x0001_FC00 ~ 0x0001_FDFF	7	0x0001_FE00 ~ 0x0001_FFFF

Table 50. Code Flash Memory Write Protection Area in 8 KB Units (A31C143)

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

Table 51. Code Flash Memory Unprotection Area (Top 4 KB) in 512 Bytes Units (A31C143)

x	Address	x	Address
0	0x0000_F000 ~ 0x0000_F1FF	1	0x0000_F200 ~ 0x0000_F3FF
2	0x0000_F400 ~ 0x0000_F5FF	3	0x0000_F600 ~ 0x0000_F7FF
4	0x0000_F800 ~ 0x0000_F9FF	5	0x0000_FA00 ~ 0x0000_FBFF
6	0x0000_FC00 ~ 0x0000_FDFF	7	0x0000_FE00 ~ 0x0000_FFFF

6.4.2.2 Write Protection for A31C12x Code Flash Memory

The CFMC_FLSPROT register is used to set the Protection and Unprotection as described in Table 52, Table 53, Table 54, and Table 55.

Table 52. Code Flash Memory Write Protection Area in 2 KB Units (A31C122)

n	Address	n	Address
0	0x0000_0000 ~ 0x0000_07FF	1	0x0000_0800 ~ 0x0000_0FFF
2	0x0000_1000 ~ 0x0000_17FF	3	0x0000_1800 ~ 0x0000_1FFF
4	0x0000_2000 ~ 0x0000_27FF	5	0x0000_2800 ~ 0x0000_2FFF
6	0x0000_3000 ~ 0x0000_37FF	7	0x0000_3800 ~ 0x0000_3FFF
8	0x0000_4000 ~ 0x0000_47FF	9	0x0000_4800 ~ 0x0000_4FFF
10	0x0000_5000 ~ 0x0000_57FF	11	0x0000_5800 ~ 0x0000_5FFF
12	0x0000_6000 ~ 0x0000_67FF	13	0x0000_6800 ~ 0x0000_6FFF
14	0x0000_7000 ~ 0x0000_77FF	15	0x0000_7800 ~ 0x0000_7FFF

Table 53. Code Flash Memory Unprotection Area (Top 4 KB) in 512 Bytes Units (A31C122)

x	Address	x	Address
0	0x0000_7000 ~ 0x0000_71FF	1	0x0000_7200 ~ 0x0000_73FF
2	0x0000_7400 ~ 0x0000_75FF	3	0x0000_7600 ~ 0x0000_77FF
4	0x0000_7800 ~ 0x0000_79FF	5	0x0000_7A00 ~ 0x0000_7BFF
6	0x0000_7C00 ~ 0x0000_7DFF	7	0x0000_7E00 ~ 0x0000_7FFF

Table 54. Code Flash Memory Write Protection Area in 2 KB Units (A31C121)

n	Address	n	Address
0	0x0000_0000 ~ 0x0000_07FF	1	0x0000_0800 ~ 0x0000_0FFF
2	0x0000_1000 ~ 0x0000_17FF	3	0x0000_1800 ~ 0x0000_1FFF
4	0x0000_2000 ~ 0x0000_27FF	5	0x0000_2800 ~ 0x0000_2FFF
6	0x0000_3000 ~ 0x0000_37FF	7	0x0000_3800 ~ 0x0000_3FFF

Table 55. Code Flash Memory Unprotection Area (Top 4 KB) in 512 Bytes Units (A31C121)

x	Address	x	Address
0	0x0000_3000 ~ 0x0000_31FF	1	0x0000_3200 ~ 0x0000_33FF
2	0x0000_3400 ~ 0x0000_35FF	3	0x0000_3600 ~ 0x0000_37FF
4	0x0000_3800 ~ 0x0000_39FF	5	0x0000_3A00 ~ 0x0000_3BFF
6	0x0000_3C00 ~ 0x0000_3DFF	7	0x0000_3E00 ~ 0x0000_3FFF

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 chapter 6.4.1).

6.5 Flash Memory Interrupts

For the code flash memory control, A31C14x and A31C12x series supports the write done interrupt. This interrupt can be activated by setting the WDIEN bit in the CFMC_CTRL register to ‘1’.

6.6 Code Flash Memory Controller Registers

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

Table 56. Base Address of CFMC

Name	Base Address
CFMC	0x4008_0000

Table 57. CFMC Register Map

Name	Offset	Type	Description	Reset Value	Ref.
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	0x0010	RW	Code Flash User Info Protection Register	0x0000_0000	6.6.5
CFMC_CTRL	0x0014	RW	Code Flash Access Control Register	0xC000_0000	6.6.6
CFMC_STAT	0x0018	RW	Code Flash Access Status Register	0x0000_0000	6.6.7
CFMC_READPROT	0x001C	RW	Code Flash Read Protection Register	0x0000_00FF	6.6.8
CFMC_PWIN	0x0020	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_PWPRST	0x0F18	WO	Code Flash Password Reset Register	0x0000_0000	6.6.11

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=0x4008_0000

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

25	WRITE	Code flash write enable bit
		0 Disables.
		1 Enables.
3 0	LATENCY[3:0](1)	Flash memory wait value bits
		Number of flash wait cycle value between 0 to 15.

NOTE:

1. For detailed information on 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=0x4008_0004

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

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

NOTE:

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

6.6.3 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=0x4008_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. User info access control bit must be disabled before controlling user info memory. To this end, the KEY values must be written in the CFMC_INFOKEY register in the order of KEY1 -> KEY2 -> KEY3. Once the User info access control bit is disabled, the access enable status is maintained until the INFOLOCK bit of the CFMC_CTRL register is written as '1'.

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=0x4008_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	Reserved	FPBY16_15	FPBY16_14	FPBY16_13	FPBY16_12	FPBY16_11	FPBY16_10	FPBY16_9	FPBY16_8	FPBY16_7	FPBY16_6	FPBY16_5	FPBY16_4	FPBY16_3	FPBY16_2	FPBY16_1	FPBY16_0
0	0	0	0	0	0	0	0	-	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
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 - The settings of these bits have a higher priority level than the protection selection bits. - A31C144) This bit is only meaningful if the FPBY16_15 bit is '1'. - A31C122) When FPBY16_14 bit is '1' meaningful and x=0~3 and FPBY16_15 bit is '1', x=4~7 is meaningful.	
		0	Disable (protection).
		1	Enable (unprotection).
n	FPBY16_n (n = 0 to 15)	Flash area protection selection bit	
		0	Unprotection
		1	Protection

NOTES:

1. When any of the FPBY16_n bit is enabled, chip erase does not work because the protected area is included in the erase.
2. Likewise, as each unprotection sector is 512 bytes, other erase commands (2 KB and chip erases) do not work if any of the FUP512B_x bit is '0' when the FPBY16_15 bit of A31C14x is '1' and when the FPBY16_14 or FPBY16_15 bit of A31C12x is '1', the erase commands such as ‘Sector erase’, and ‘Chip erase’ do not work for the area that includes protection area.
3. For detailed address and description of each area, refer to section 6.4.2.
4. FPBY16_n bit has 1/16 size of the code flash size.

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=0x4008_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				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 (Read protection area) 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 32 bits wide.

CFMC_CTRL=0x4008_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
FLOCK	INFOLOCK	Reserved			WDIEN	WABORT	Reserved	CERS	SERS	PERS	PGM
1	1	-			0	0	-	0	0	0	0
RW	RW	-			RW	RW	-	RW	RW	RW	RW

31	FLOCK[1]	Code flash access control bit Flash access control bit must be disabled before controlling flash memory. To this end, the KEY values must be written in the CFMC_FLSKEY register in the order of KEY1 -> KEY2 -> KEY3. Once the access control bit is disabled, the access enable status is maintained until the FLOCK bit of the CFMC_CTRL register is written as '1'.
	0	Unlock status
	1	Lock status
30	INFOLOCK	User info access control bit User info access control bit must be disabled before controlling 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 access control bit is disabled, the access enable status is maintained until the INFOLOCK bit of the CFMC_CTRL register is written as '1'.
	0	Unlock status
	1	Lock status
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=0x4008_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	INFOPERR	FPERR	INFOLERR	FLERR	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 protection 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 protection error	
		0	Not error
		1	Error detection (cleared by writing a ‘1’)
18	FPERR	Flash memory protection 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 memory lock error	
		0	Not error
		1	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=0x4008_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
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=0x4008_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
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 Timeout 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=0x4008_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]																															
0x00000000																															
RW																															

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

6.6.11 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=0x4008_0F18

31	30	29	28	27	26	25	24	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]																															
-																															
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	

6.6.12 CFMC Register Map Summary

Table 58. 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							WRITE																						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									FPBY16_15	FPBY16_14	FPBY16_13	FPBY16_12	FPBY16_11	FPBY16_10	FPBY16_9	FPBY16_8	FPBY16_7	FPBY16_6	FPBY16_5	FPBY16_4	FPBY16_3	FPBY16_2	FPBY16_1	FPBY16_0
	Reset value	0	0	0	0	0	0	0	0									0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
0x10	CFMC_INFOPROT																													INFO3	INFO2	INFO1	INFO0
	Reset value																													0	0	0	0
0x14	CFMC_CTRL	FLOCK	INFOLOCK																						WDIEN	WABOART				CERS	SERS	PERS	PGM
	Reset value	1	1																						0	0				0	0	0	0
0x18	CFMC_STAT										WTERR	RPERR	WSERR	INFOPERR	FPERR	INFOLERR	FLERR								WDONE								WBUSY
	Reset value										0	0	0	0	0	0	0								0								0
0x1C	CFMC_READPROT	DBGMOD	SRBOOT				PWMATCH	INFO0ERSD	CERSD							LVL2_STS	LVL1_STS							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 58. 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
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

7. HSICMU

7.1 HSICMU Introduction

The High-Speed Internal OSC Clock Management Unit (HSICMU) integrated into the A31C14x, and A31C12x series is designed for the auto-trimming of the Internal OSC clock. The synchronization signal can also be derived from the LSE or HSE oscillator output, or it can be generated by user software.

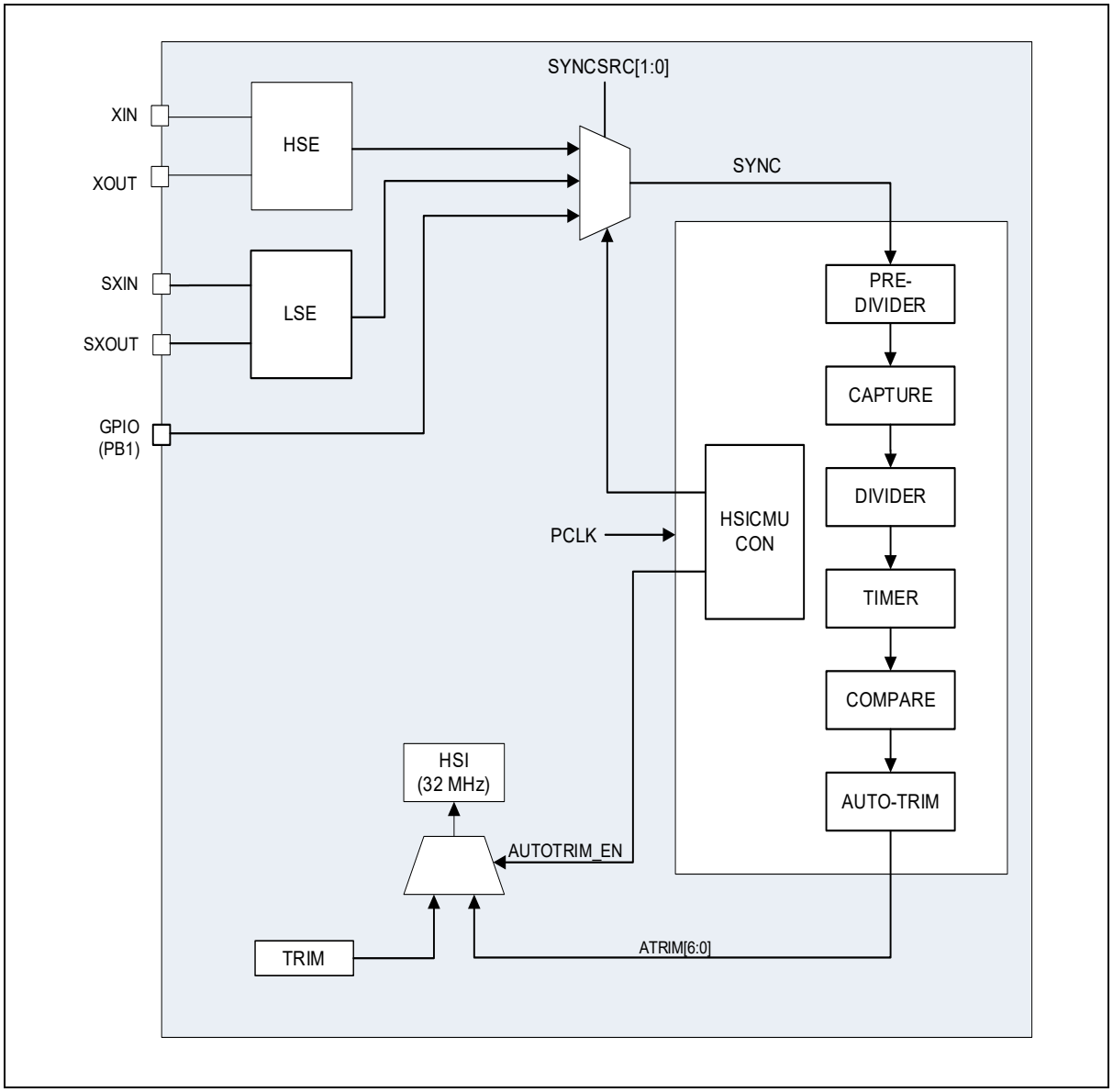
7.2 HSICMU Main Features

- Selectable synchronization source with programmable prescaler and polarity:
  - HSE (4 MHz to 12 MHz)
  - LSE (32.768kHz)
  - External SYNC pin (PB1)
- 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 58. 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 HSE clock signal.

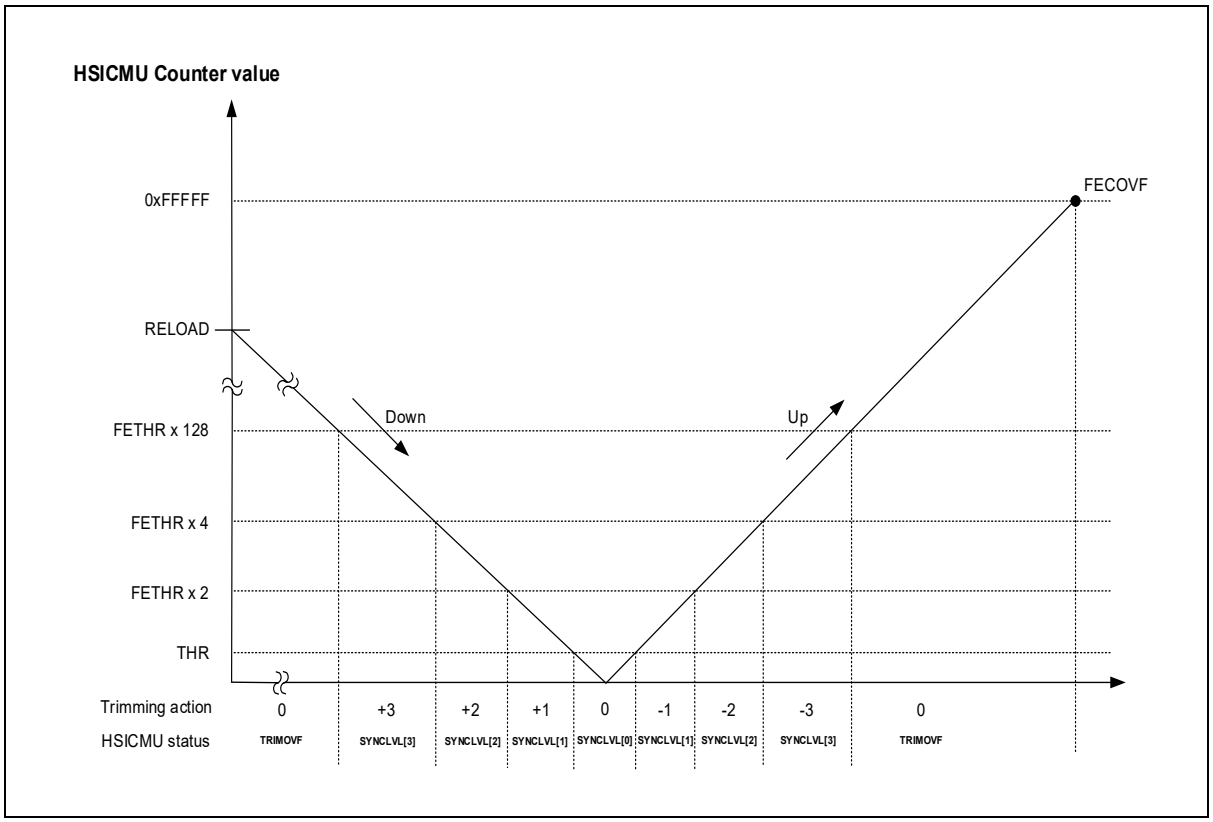
For more information on the HSICMU synchronization source configuration, refer to 7.4.2.

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 0xFFFFF 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 59. 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.
  - 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 32.768 kHz (LSE)

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:

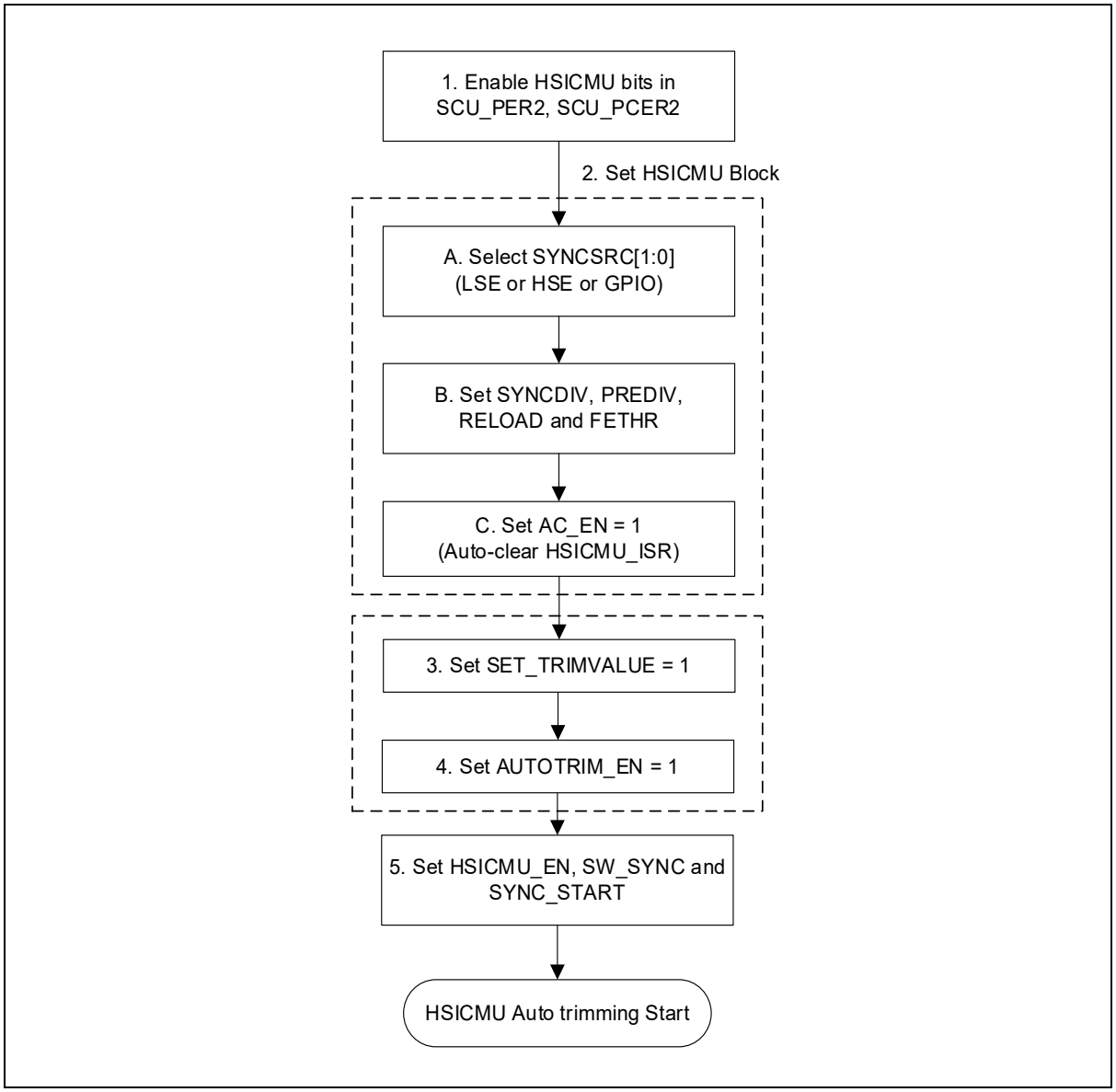
$$\mathrm{FETHR} = \left(\frac{\mathrm{f_{TARGET}}}{\mathrm{f_{SYNC}}}\right) \times \mathrm{HSI\ TRIM\ STEP[\%]/100\%}$$

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.

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 60. HSICMU Initialization Flowchart



1. Enable the HSICMU-related bits in the SCU_PER2 and SCU_PCER2 registers.

2. The following sequences (A, B, and C) describe the procedure for configuring 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 HSE or GPIO.

   B. Calculate and set SYNCDIV[2:0] and PREDIV[3:0] so that RELOAD[15:0] is not set too small. Examples of appropriate SYNCDIV[2:0], PREDIV[3:0], RELOAD[15:0], and FETHR[15:0] settings for each SYNCSRC[1:0] can be found in Table 59 and Table 60. 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. 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 immediately after enabling the SET_TRIMVALUE bit. 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_0x2000; // Enable SET_TRIMVALUE bit

      HSICMU_CR = 0x0000_0x0200; // 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. Down-count is performed based on the value written to the RELOAD register. When the count reaches zero, it continues counting up to 0xFFFFF. However, reaching 0xFFFFF typically indicates a malfunction, in which case the FECOVF flag is triggered. For more details on this operation, refer to Figure 59.

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

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

Table 59 shows examples of RELOAD[15:0] and FETHR[15:0] values according to PCLK for each SYNCSRC[1:0]. When fTARGET was LSE(32.768 kHz), the calculation was made assuming the threshold was 0.12%.

Table 59. Examples HSICMU (SYNCSRC[1:0] = 10, LSE)

SYNCSRC[1:0]	PREDIV[3:0]	SYNCDIV[2:0]	PCLK	RELOAD[15:0]	FETHR[15:0]
10 (LSE, 32.768kHz)	1001	010	32 MHz	2000000	2400
	1001	011	16 MHz	2000000	2400
	1001	100	8 MHz	2000000	2400
	1001	101	4 MHz	2000000	2400
	1001	110	2 MHz	2000000	2400

Table 60 provides an example when SYNCSRC[1:0] is set to HSE (8 MHz).

When fTARGET is 8 MHz, it is recommended to divide it using SYNCDIV[2:0], PREDIV[3:0] to increase the RELOAD[15:0] value, since a small RELOAD value may result in lower accuracy.

For this reason, a threshold of 0.12% is assumed for the calculation.

Table 60. Examples HSICMU (SYNCSRC[1:0] = 01, HSE)

SYNCSRC[1:0]	PREDIV[3:0]	SYNCDIV[2:0]	PCLK	RELOAD[15:0]	FETHR[15:0]
01 (HSE, 8MHz)	1001	111	32 MHz	262144	314
	1001	111	16 MHz	131072	157
	1001	111	8 MHz	65536	78
	1001	111	4 MHz	32768	39
	1001	111	2 MHz	16384	19

7.4 HSICMU Registers

The base address of HSICMU is described in the following tables:

Table 61. Base Address of HSICMU

Name	Base Address
HSICMU	0x4000_4000

Table 62. HSICMU Register Map

Name	Offset	Type	Description	Reset Value	Ref.
HSICMU_CR	0x0000	RW	HSICMU Control Register	0x0000_1000	7.4.1
HSICMU_CFG	0x0004	RW	HSICMU Configure register	0x000F_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_ATRIM	0x0014	RW	HSICMU Auto trim register	0x0000_0000	7.4.5
HSICMU_CAP	0x0018	RW	HSICMU Capture register	0x0000_0000	7.4.6
HSICMU_THR	0x001C	RW	HSICMU Frequency error threshold register	0x0000_0000	7.4.7

7.4.1 HSICMU_CR: HSICMU Control Register

The HSICMU_CR register has control bits to the HSICMU operation. This register is 32 bits 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	Sets the trim value to HSICMU_ATRIM block.
	0	Disables.
	1	Enables.
12	SYNC_START	Starts synchronized timer by SYNC 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	Generates 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 32 bits 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]	PREDIV[3:0]	RELOAD[19:0]	
00	00	0	000	0000	0xFFFFF	
RW	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	Reserved.
		01	HSE
		10	LSE
		11	GPIO (PB1)
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.
23 20	PREDIV[3:0]	Pre divider	
		0000	Sync not divided
		N	Pre divided by 2^N (N max. 9)
19 0	RELOAD[19:0]	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_ISR=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 an 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 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	-		RO	RO	RO	RO	RO

14	SYNC_FLAG	Sync Signal Flag.
		0 None.
		1 Clear when set to ‘1’.
15	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.
		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_ATRIM: HSICMU Control Auto-Trim Value Register

The HSICMU_ATRIM is used to check auto-trim value of HSICMU. This register is 32 bits 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[6:0]
-			0x00
-			R0

6 0	ATRIM[6:0]	Current trim value during auto trimming.

7.4.6 HSICMU_CAP: HSICMU Capture Value Register

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

HSICMU_CAP=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[19:0]
-			0x0000
-			R0

19 0	CAPTURE[19:0]	Capture value of the frequency error rate counter.

7.4.7 HSICMU_THR: HSICMU Frequency Error Rate Threshold Register

The HSICMU_THR is used to frequency error rate threshold. This register is 32 bits 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															
-																RW															

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

7.4.8 HSICMU Register Map Summary

Table 63. 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]			PREDIV[3:0]				RELOAD[19:0]																			
	Reset value	0	0	0	0	0	0	0	0	0	0	0	0	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	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.	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
0x14	HSICMU_ATRIM	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.	ATRIM[6:0]						
	Reset value																										0	0	0	0	0	0	0
0x18	HSICMU_CAP	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	CAPTURE[19:0]																			
	Reset value													0	0	0	0	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

8.1 DMA Introduction

The DMA (Direct Memory Access) 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.

- A31C14x series has 6 channels, and A31C12x series has 4 channels
- 8-/16-/32-bit data transfers supported
- Various buffers with the same size supported
- DMA transfers are triggered through peripheral interrupts

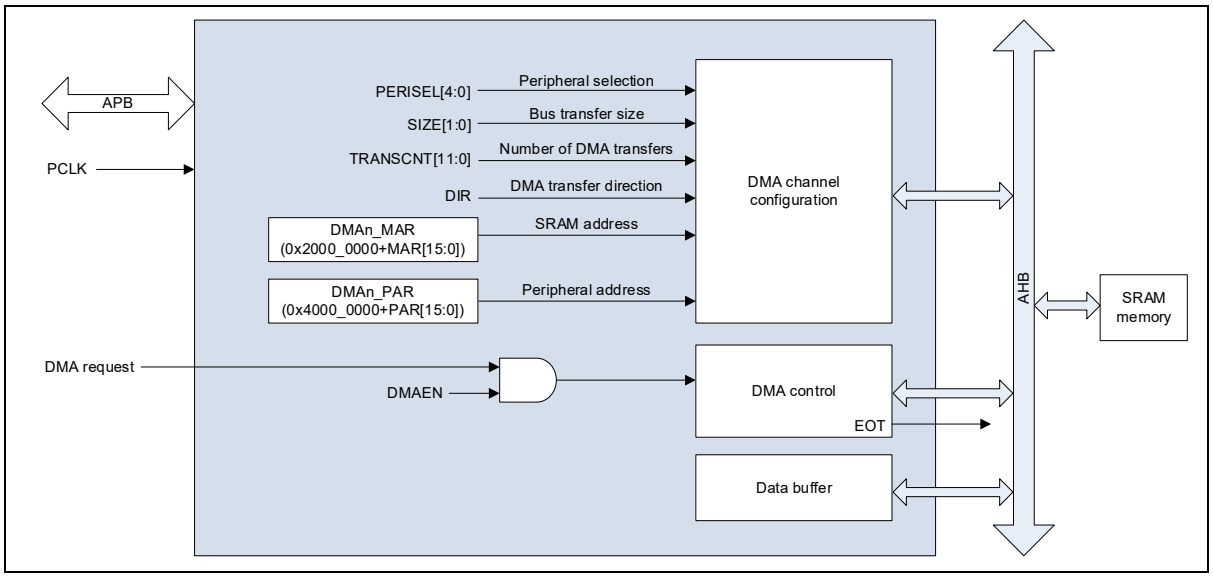
8.1.1 Modules Supported by DMA

DMA supports the following four peripherals.

- UART
- USART
- ADC
- CRC

Figure 61 shows a DMA block diagram.

Figure 61. DMA Block Diagram



8.2 DMA Functional Description

The DMA controller directly transfers data to memories by sharing the AHB with the microcontroller core. The AHB shares two masters operating in a round-robin fashion. Therefore, the DMA controller shares only half the system bandwidth with the microcontroller core.

The DMA controller is triggered only by requests made by peripherals. Once a peripheral requests a transfer to the DMA controller, the connected channel becomes enabled. Then, the bus is accessed to transfer the requested data from the memory buffer to the peripheral's data buffer (or the other way around).

The DMA controller operates in the following sequence:

1. The programmer sets the address of the peripheral to use in the DMAn_PAR register and the memory address in the DMAn_MAR register.
2. In the DMAn_CR register, the programmer sets the DMA transfer count (0 to 4,095), 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[3:0] 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:

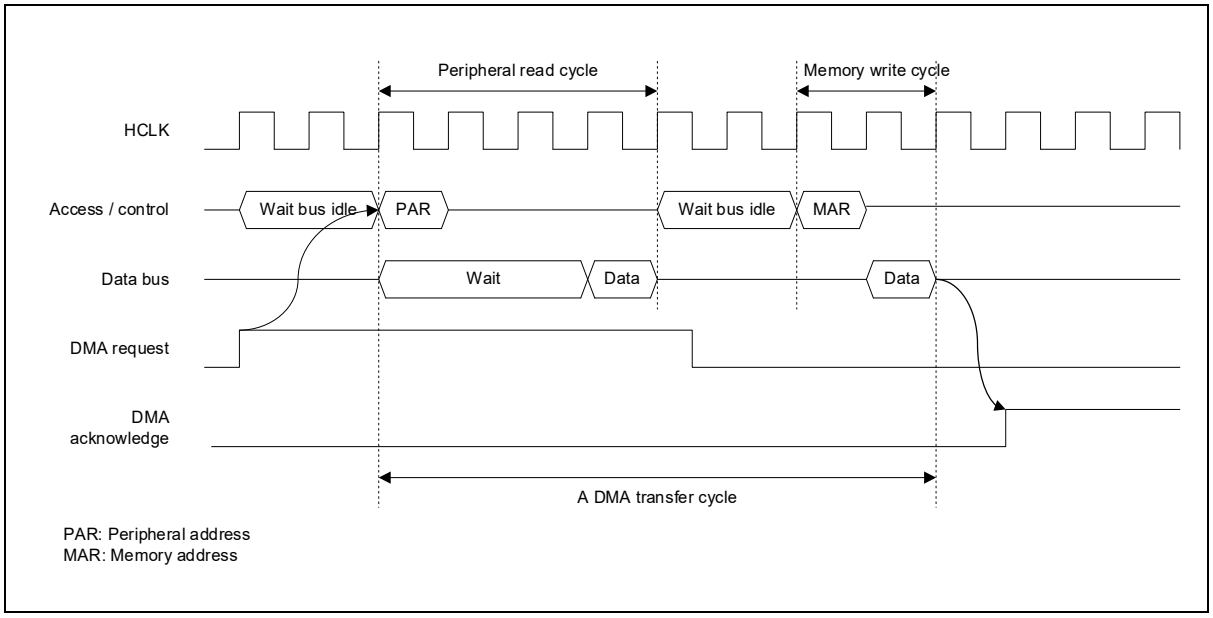
1. DMA itself has no interrupt sources; instead, each peripheral has flag bits that display the status of the connected DMA channel's transfer interrupts.

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 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.

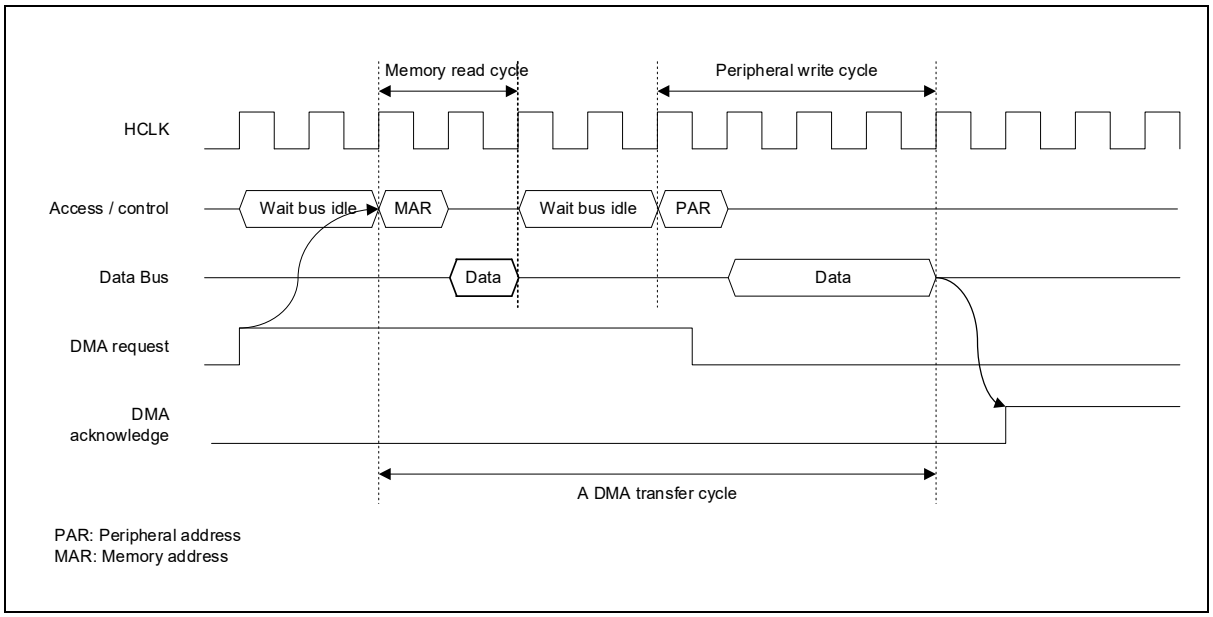
Figure 62. Diagram of DMA Transfer Timing from Peripheral to Memory



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.

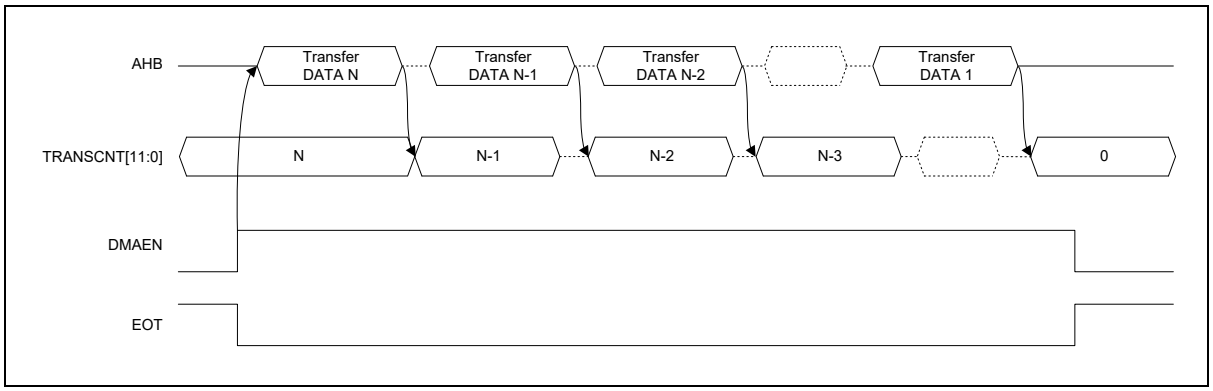
Figure 63. Diagram of DMA Transfer Timing from Memory to Peripheral



8.2.3 DMA Transfer

The following diagram shows an example of DMA transfers in a queue. DMA transferring starts when a DMAEN bit of DMAn_SR is set to ‘1’. And the bit is cleared to zero when all transfers have been completed.

Figure 64. N Number of DMA Transfers



8.3 DMA Registers

The base address of DMA is described in the following tables:

Table 64. Base Address of DMA

Name	Base Address
DMA0	0x4000_0400
DMA1	0x4000_0410
DMA2	0x4000_0420
DMA3	0x4000_0430
DMA4	0x4000_0440
DMA5	0x4000_0450

Table 65. DMA Register Map

Name	Offset	Type	Description	Reset Value	Ref.
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

NOTES:

1. A31C14x has a DMA 6 channels (n = 0 to 5).
2. A31C12x has a DMA 4 channels (n = 0 to 3).

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

31 30 29 28 27 26 25 24	23 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[3:0]	Reserved	SIZE[1:0]	DIR	Reserved
-	0x000	-	0x0	-	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.
11 8	PERISEL[3:0](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 Selects the peripheral to use. Refer to Table 66.
3 2	SIZE[1:0]	Bus transfer size
		00 Sets the DMA transfer size to 1 byte.
		01 Sets the DMA transfer size to half-word (2 bytes) size.
		10 Sets the DMA transfer size to one-word (4 bytes) size.
		11 Holds off the size setting.
1	DIR	Transfer direction selection
		0 Memory to peripheral (TX)
		1 Peripheral to memory (RX)

NOTE:

1. A DMA channel should be connected to selected peripheral. Below table shows peripheral selection numbers. This PERISEL[3:0] field should be set with proper number of peripherals which will be connected with DMA interface.

Table 66. DMA PERISEL[3:0] Selection (A31C14x)

PERISEL[3:0]	Peripheral	PERISEL[3:0]	Peripheral
0	CHANNEL IDLE	7	USART11 RX
1	UART0 RX	8	USART11 TX
2	UART0 TX	9	USART12 RX
3	UART1 RX	10	USART12 TX
4	UART1 TX	11	ADC RX
5	USART10 RX	12	CRC TX
6	USART10 TX	13 to 15	Reserved

Table 67. DMA PERISEL[3:0] Selection (A31C12x)

PERISEL[3:0]	Peripheral	PERISEL[3:0]	Peripheral
0	CHANNEL IDLE	7	USART11 RX
1	UART0 RX	8	USART11 TX
2	UART0 TX	9	Reserved
3	Reserved	10	Reserved
4	Reserved	11	ADC RX
5	USART10 RX	12	CRC TX
6	USART10 TX	13 to 15	Reserved

PERISEL[3:0] cannot have the same value on different channels. If the same PERISEL[3:0] value is written in more than one channel, proper operation cannot be guaranteed. Unused channels should have CHANNEL IDLE value in PERISEL[3:0] bit positions.

8.3.2 DMAn_SR: DMA Channel n Status Register

DMAn_SR is a 32-bit register. It shows the status of the DMA controller and whether the DMA channel is enabled or disabled.

DMA0_SR=0x4000_0404, DMA1_SR=0x4000_0414
DMA2_SR=0x4000_0424, DMA3_SR=0x4000_0434
DMA4_SR=0x4000_0444, DMA5_SR=0x4000_0454

31 30 29 28 27 26 25 24	23 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. The 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

31 30 29 28 27 26 25 24	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

31 30 29 28 27 26 25 24	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 68. 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.	Res.	PERISEL[34:0]				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																									0			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

9.1 WDT Introduction

Watchdog Timer (WDT) is used to detect microcontroller errors due to external interference or logically unexpected 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 A31C14x and A31C12x series is a 32-bit down counter. If the WDT is set as a reset source, the microcontroller restarts when the down counter reaches zero.

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
  - The clock source can be selected by configuring the WDTCSEL[2:0] in SCU_MCCR1 register: LSI, LSE, HSI, and HSE
- 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

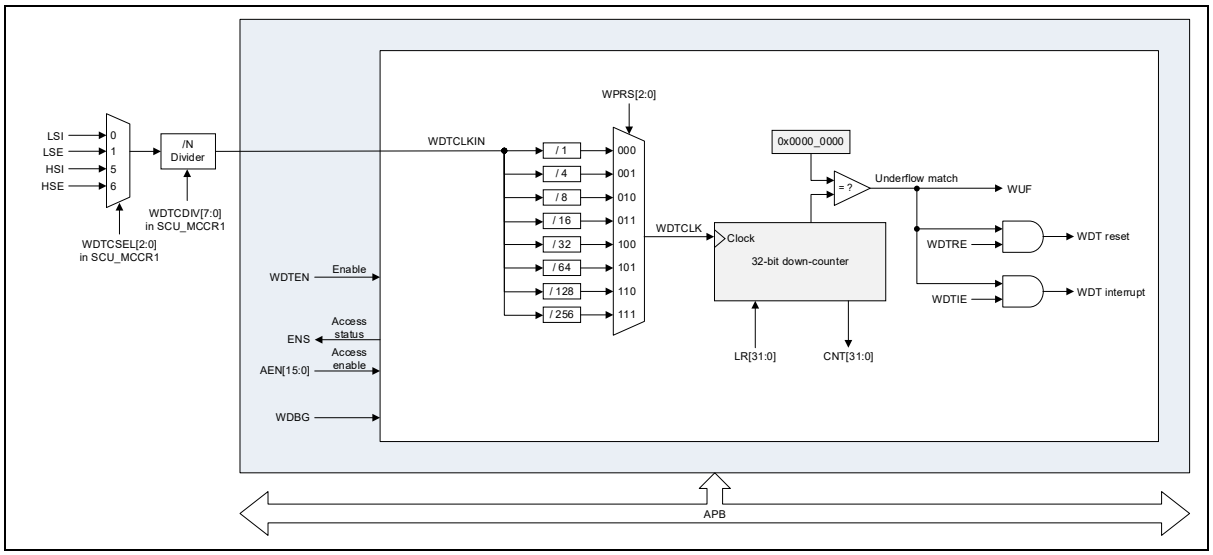
Once the WDT module is enabled and the WDTEN in WDT_CON register is set to '1', the 32-bit down counter is decremented. The application program writes a value in the WDT_LR register regularly to prevent an instant reset or an interrupt. Users must write certain values in the WDT_LR register to update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

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

9.3.1 WDT Block Diagram

Figure 65 shows a block diagram for the WDT.

Figure 65. WDT Block Diagram



9.3.2 Enabling WDT

The WDT is always enabled after the power is on and operates in WDT reset mode.

An external clock source (SCU_MCCR1) can be set as a clock source for the WDT. By default, the WDT clock source is set to LSI (32 kHz). It is recommended to use a clock source other than MCLK for the WDT.

When changing the clock source, disable WDTEN in the WDT_CON register before making the change.

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 69 describes the prescaled frequencies of various clock sources in the WDT.

Table 69. Prescaled WDT Counter Clock Frequency

Clock Source	WDTCLKIN / 1	WDTCLKIN / 4	WDTCLKIN / 16	WDTCLKIN / 64	WDTCLKIN / 256
LSI	32 kHz	8 kHz	2 kHz	500 Hz	125 Hz
LSE	32.768 kHz	8.192 kHz	2.048 kHz	512 Hz	128 Hz
HSI	32 MHz	8 MHz	2 MHz	500 kHz	125 kHz
HSE	XTAL	XTAL / 4	XTAL / 16	XTAL / 64	XTAL / 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’

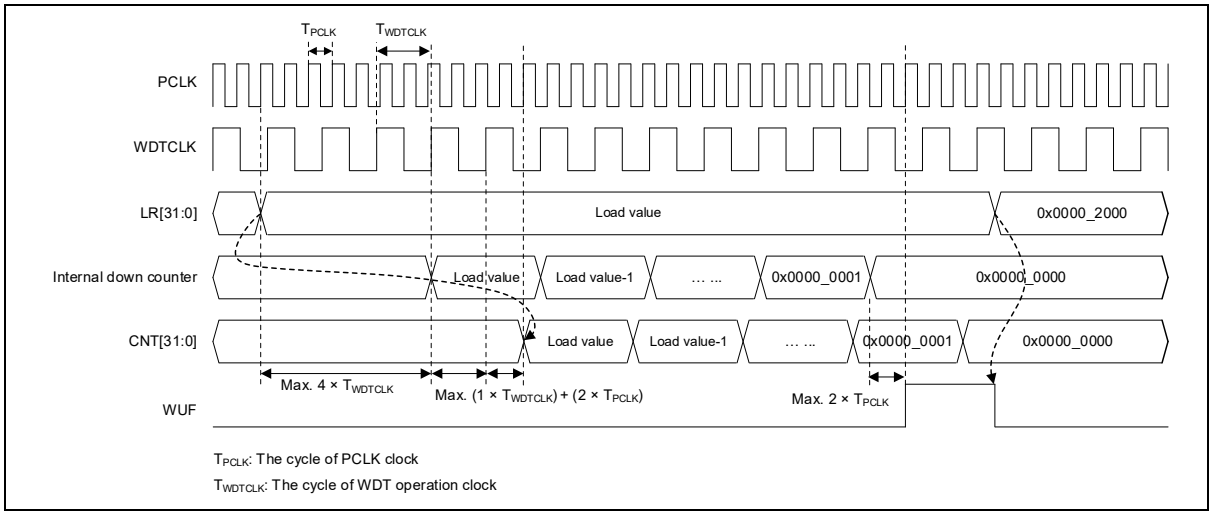
9.3.4 Interrupt Feature

The WDT interrupt can be used to perform safety tasks or data logging before triggering an actual reset.

The WDT interrupt is enabled by setting the WDTIE in WDT_CON register to ‘1’. 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 66. 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’. When the down-counter reaches zero (the moment the value of the down counter changes from ‘1’ to ‘0’), the WDT generates the reset signal to reset microcontroller.

9.3.6 Debug Mode

If the DBG_IWDT_STOP in DBG_CR 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 DBG_IWDT_STOP in DBG_CR 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 address of WDT and register map are described in the following. Initial setting of WDT's timeout period is 2,000 ms.

Table 70. Base Address of WDT

Name	Base Address
WDT	0x4000_0200

Table 71. 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_07D0	9.4.2
WDT_CON	0x0008	RW	WDT Control Register	0x0000_0054	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 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_07D0																															
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 '0'.

WDT_CON=0x4000_0208

31 30 29 28 27 26 25 24	23 22 21 20 19 18 17 16	15 14 13 12 11 10 9	8	7	6	5	4	3	2 1 0
Reserved		Reserved	WUF	WDTIE	WDTRE	Reserved	WDTEN	Reserved	WPRS[2:0]
-		-	0	0	1	-	1	-	100
-		-	RW	RW	RW	-	RW	-	RW

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.
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	W0	

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.

NOTES:

1. Example code for using the WDT_AEN register

   WDT_AEN = 0xA55A;// Enables WDT access.

   // WDT_LR and WDT_CON, etc. become settable.

   WDT_AEN = 0; // Disables WDT access.

2. To use the reload feature, a value greater than 0 must be set in the WDT_LR register.

   WDT_AEN = 0xA55A; // Enable WDT access.

   WDT_LR = 0x7A12; // Set timer load value.

   WDT_AEN = 0; // Disable WDT access.

3. The instant reload feature

   WDT_AEN = 0x555A; // Triggers an instant reload.

   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 72. WDT Register Map Summary

Offset	Register name	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0x00	WDT_LR	LR[31:0]																															
	Reset value	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
0x04	WDT_CNT	CNT[31:0]																															
	Reset value	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	1	1	1	0	1	0	0	0	0	1	0	0	1	0
0x08	WDT_CON	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	WUF	WDTIE	WDTRE	Res	WDTEN	Res	WPRS[2:0]		
	Reset value																								0	0	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

This 16-bit Timer supports four operating modes such as Periodic mode, PWM mode, One-shot mode, and Capture mode.

Users can use a divided PCLK or an external clock as an input clock source for the 16-bit Timer. An internal 10-bit prescaler allows to generate a variety of timer base clocks.

Interrupts can be triggered at regular intervals when the timer is used in Periodic Mode. Users can set the period and duty to form a PWM signal that is used in PWM Mode. In One-shot Modes, the timer can generate one PWM waveform. In Capture Mode, the external input signal's pulse intervals can be measured based on the preset condition. Moreover, the timer can export signals to other devices to control them. This timer is primarily used as periodic tick timer or wake-up source.

10.2 16-bit Timer Main Features

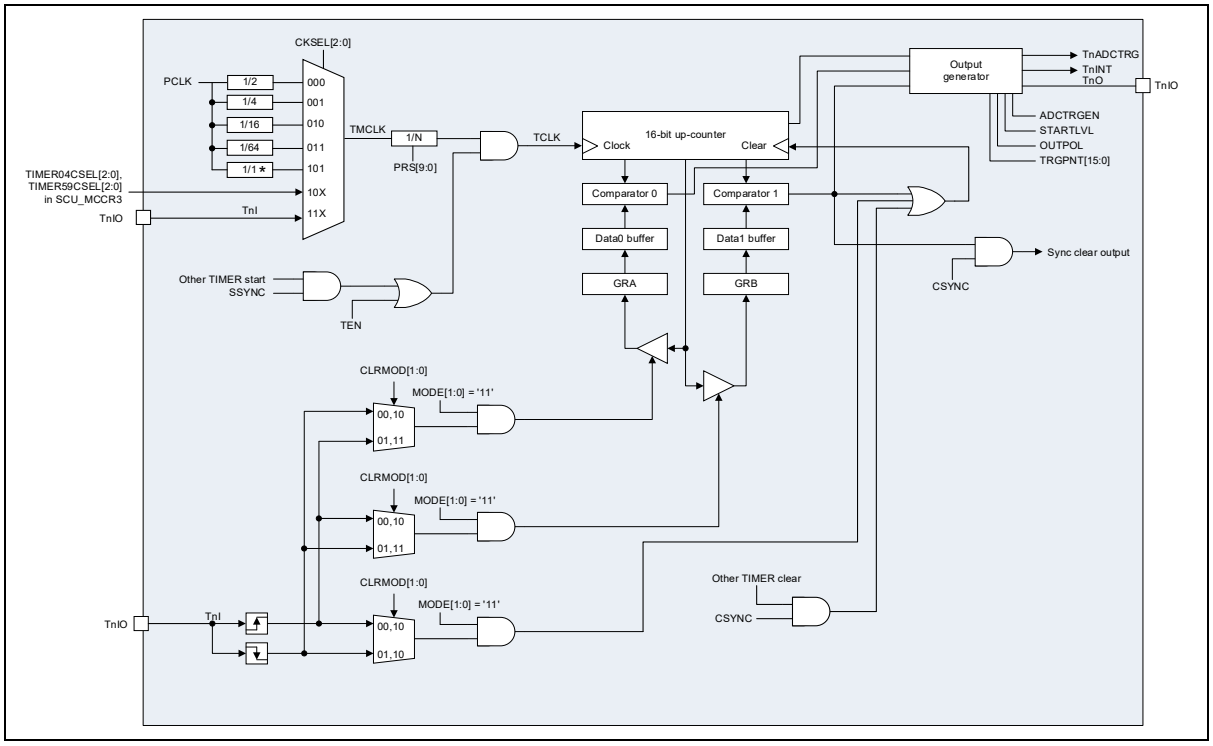
TIMERn includes the features below:

- 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:
  - A31C14x: Four PCLK prescaler levels (1/2, 1/4, 1/16, and 1/64)
  - A31C12x: Five PCLK prescaler levels (1/1, 1/2, 1/4, 1/16, and 1/64)
  - External clock sources selectable using the SCU_MCCR3 register: LSI, LSE, MCLK, HSI, and HSE
  - Timer clock source on port TnIO used as input.
- 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 67. 16-bit Timer Block Diagram



NOTE:

* PCLK/1 is supported only on the A31C12x series. For the A31C14x series, EXT_CLK is selected when CKSEL[2:0] is set to ‘1’00 or 101.

10.3.2 Pins and Internal Signals

Table 73 summarizes the I/O information on the TIMERn

Table 73. Input and Output Pins for TIMERn

Pin Name	Type	Description	Package Type							
			64-LQFP	48-LQFP, QFN	44-LQFP	32-QFN, LQFP	28-TSSOP	24-QFN	20-QFN	20-TSSOP
T0IO	IO	TIMER0 capture input signal external clock input TIMER0 periodic/pwm/one-shot output	O	O	O	O	O	O	O	O
T1IO	IO	TIMER1 capture input signal external clock input TIMER1 periodic/pwm/one-shot output	O	O	O	O	O	O	O	O
T2IO	IO	TIMER2 capture input signal external clock input TIMER2 periodic/pwm/one-shot output	O	O	O	O	O	O	O	O
T3IO	IO	TIMER3 capture input signal external clock input TIMER3 periodic/pwm/one-shot output	O	O	O	O	O	O	O	O
T4IO	IO	TIMER4 capture input signal external clock input TIMER4 periodic/pwm/one-shot output	O	O	O	O	O	O	O	O
T5IO	IO	TIMER5 capture input signal external clock input TIMER5 periodic/pwm/one-shot output	O	O	O	O	-	-	-	-
T6IO	IO	TIMER6 capture input signal external clock input TIMER6 periodic/pwm/one-shot output	O	O	O	O	O	-	-	-
T7IO	IO	TIMER7 capture input signal external clock input TIMER7 periodic/pwm/one-shot output	O	O	O	-	-	-	-	-
T8IO	IO	TIMER8 capture input signal external clock input TIMER8 periodic/pwm/one-shot output	O	-	--	-	-	-	-	-
T9IO	IO	TIMER9 capture input signal external clock input TIMER9 periodic/pwm/one-shot output	O	-	-	-	-	-	-	-

Table 74. Internal Input and Output Signals for TIMERn

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

10.3.3 Clock Selection

The counter clock can be provided by the following clock sources:

- PCLK
- Timer external clock (SCU_MCCR3 register)
- Input to pin TnIO for timer clock source

Users can select a clock source by configuring the CKSEL[2:0] in TIMERn_CR

10.3.3.1 Peripheral System Clock (PCLK)

Users can select a timer input clock source using five PCLK divide levels (PCLK / 1, PCLK / 2, PCLK / 4, PCLK / 16, PCLK / 64). If CKSEL[2:0] in the TIMERn_CR1 is set to a number between 0 and 3, the timer clock is provided by the PCLK divide level.

PCLK / 1 is supported only for A31C12x. In A31C14x, CLKSEL[2:0]=101 selects EXT_CLK instead.

Table 75. PCLK Clock Selection by CKSEL[2:0] in TIMERn_CR1

CKSEL[2:0] in TIMERn_CR1	TIMERn Clock Source Selection
000	PCLK / 2
001	PCLK / 4
010	PCLK / 16
011	PCLK / 64
101	PCLK / 1

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 and HSE. If the CKSEL[2:0] in TIMERn_CR1 is set to '10X', the timer clock is provided by the SCU_MCCR3 register. A31C12x and A31C11x are set to EXT_CLK when CLKSEL[2:0]=100.

Table 76. 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 port TnIO. If the IOSEL in TIMERn_CR1 is set to 0, the TnIO pin is set as input.

Table 77. 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 78. TnIO and CKSEL[2:0] in TIMERn_CR1

CKSEL[2:0] in TIMERn_CR1	Counter clock source selection
11X	Input to pin TnIO

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

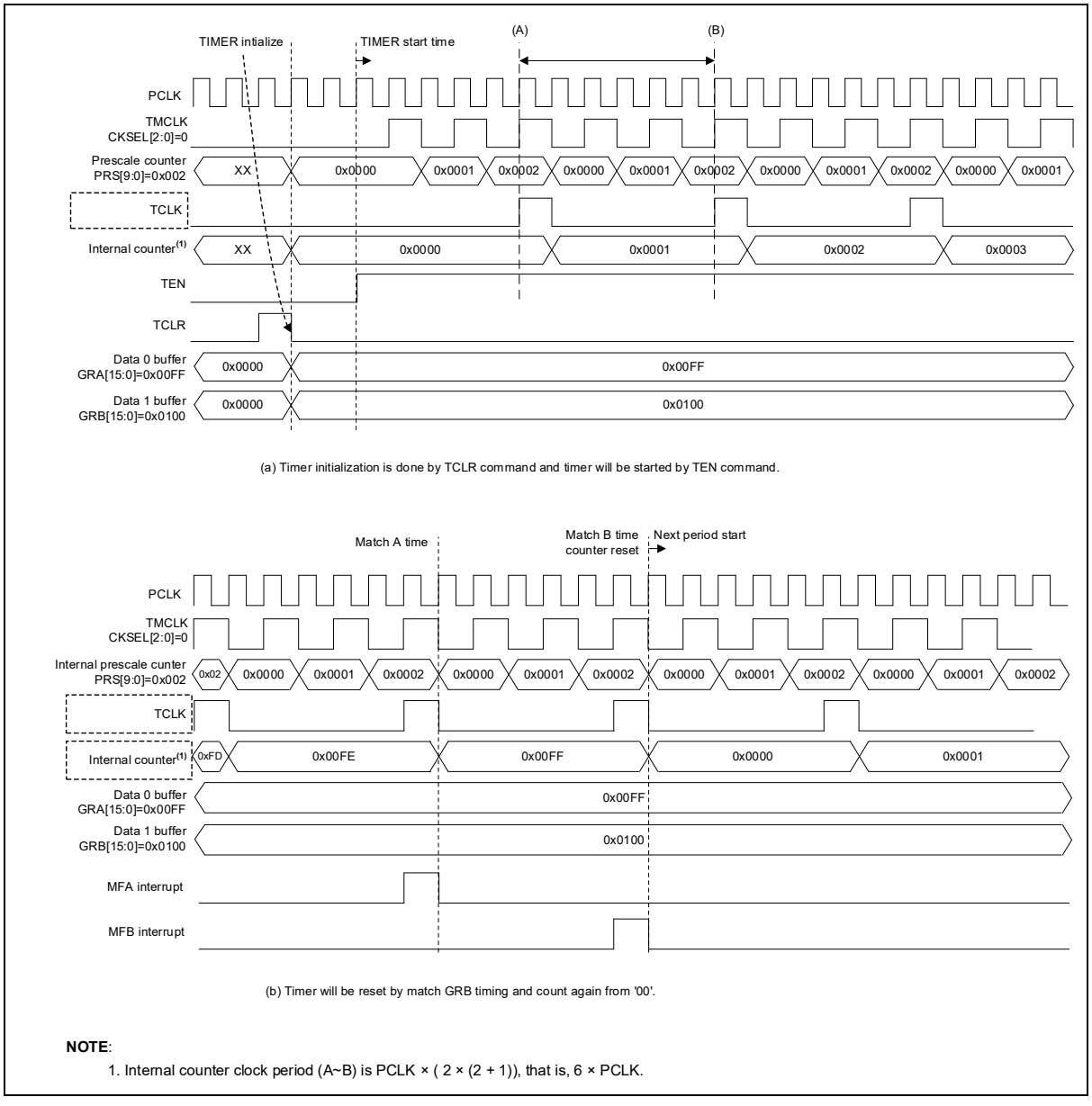
- Timer counter register: TIMERn_CNT
- Timer prescaler register: TIMERn_PRS
- Timer general data register A: TIMERn_GRA
- Timer general data register B: TIMERn_GRB
- Miscellaneous clock control register: SCU_MCCR3

10.3.4.1 Prescaler Description

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

The TMCLK shown in Figure 68 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 68. Counter Timing Diagram with Basic Start and Match Operations



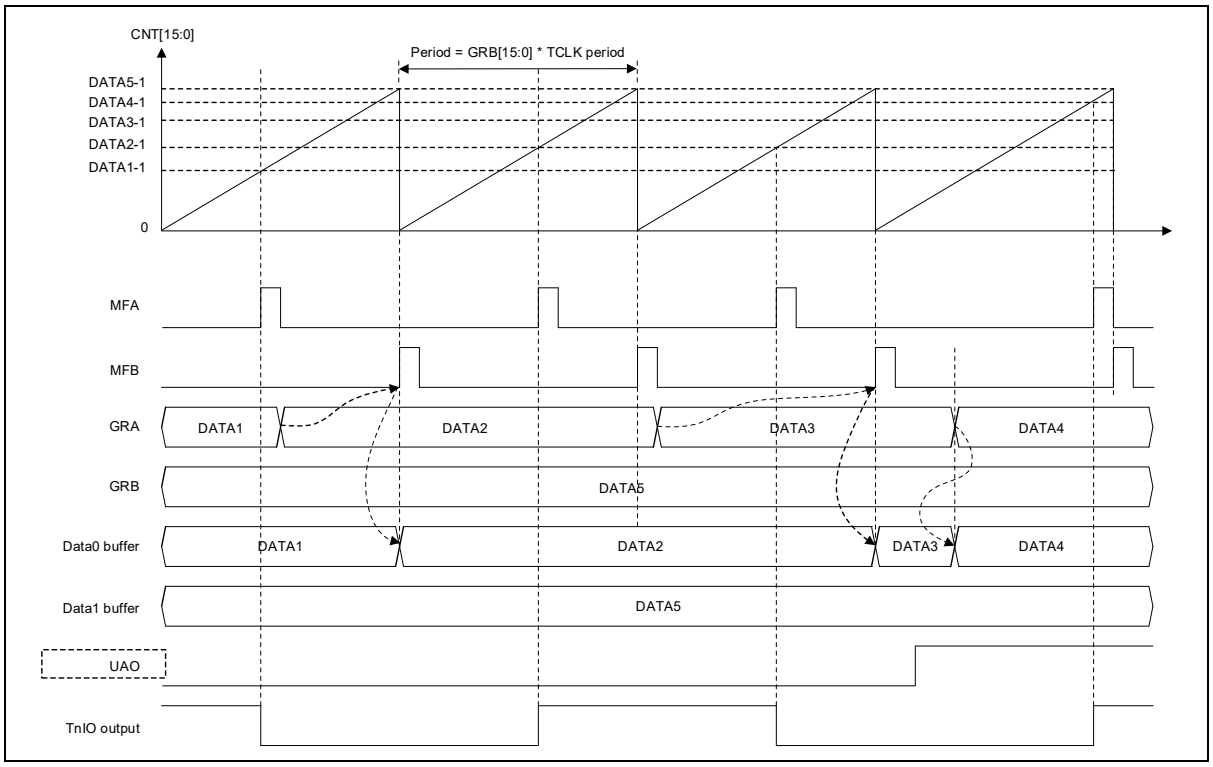
NOTE:

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

10.3.5 Periodic Mode

Figure 69 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 69. Normal Periodic Mode Operation



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

- Period = TCLK period × TIMERn_GRA value
- Match A interrupt time = TCLK period × TIMERn_GRA value

If the TIMERn_GRA value is '0, the timer cannot be started even if the TEN in TIMERn_CR2 is set to '1' because the period is 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 zero and the TIMERn_GRB value is not zero, the TnIO output does not change its previous level. If the TIMERn_GRA value is the same as the TIMERn_GRB value, the TnIO output will toggle simultaneously depending on the counter clear time. The initial level of the TnIO signal is decided by the STARTLVL 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 79 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.

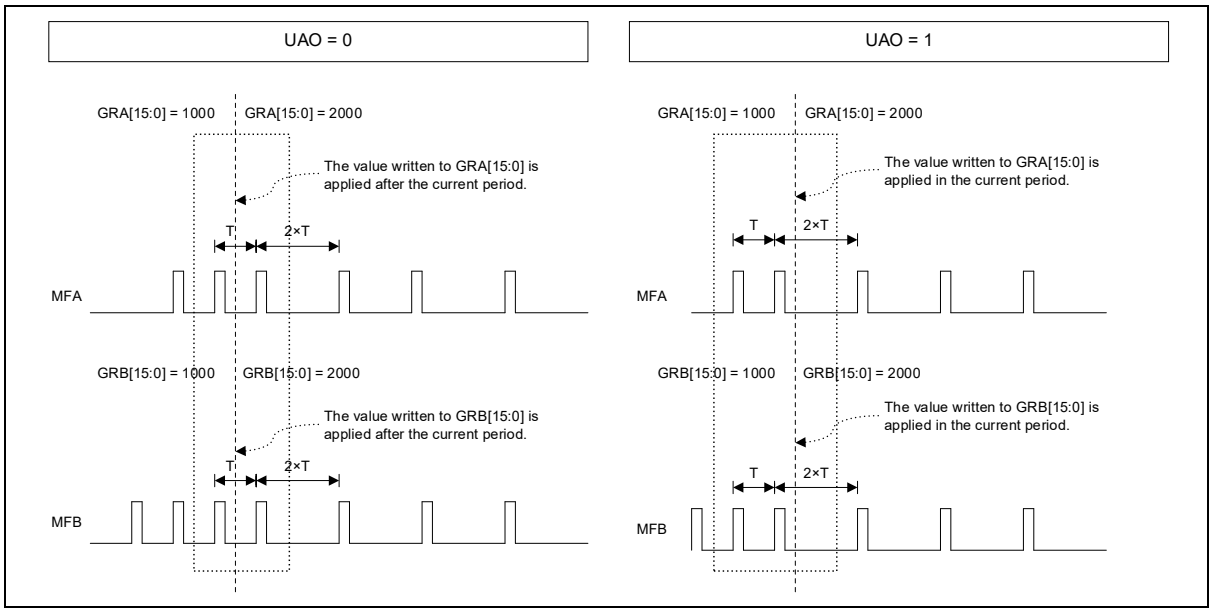
Table 79. Operations Comparison According to TIMERn_GRA/GRB Settings

No	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, OUTPOL, and STARTLVL

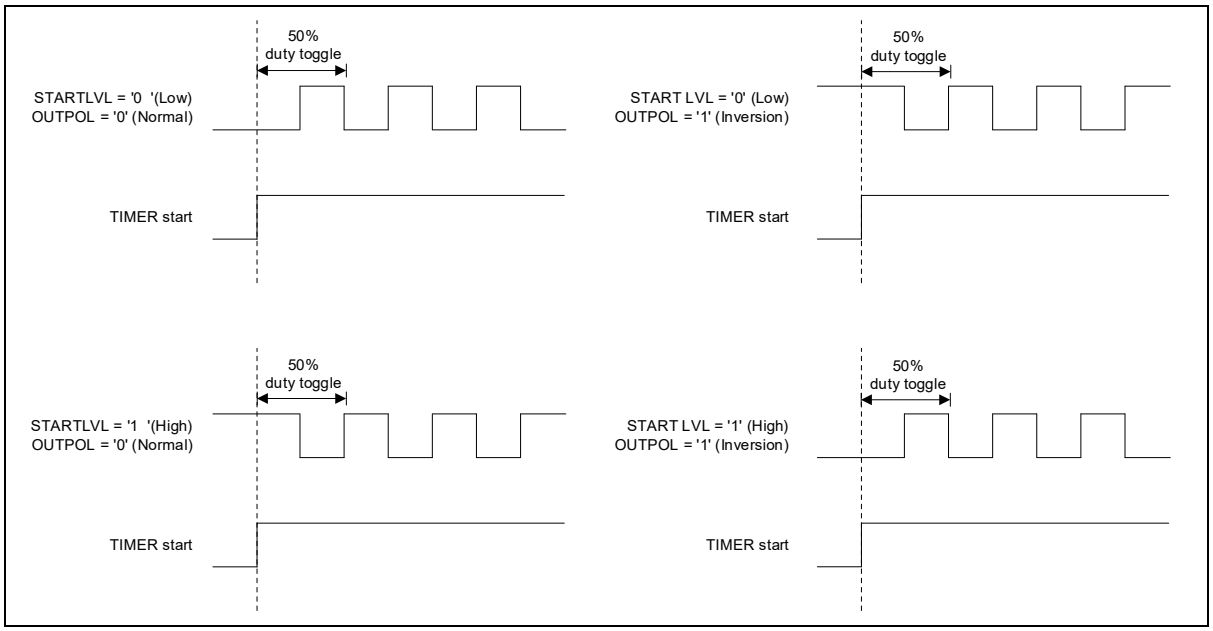
- Figure 70 shows the operation waveform according to the UAO in TIMERn_CR1 settings.
  - If the UAO in TIMERn_CR1 value is ‘0’, the TIMERn_GRA/GRB value which has been updated after the current cycle operation is applied.
  - If the UAO in TIMERn_CR1 value is ‘1’, the TIMERn_GRA/GRB value is applied within the current cycle operation.

Figure 70. Operation Waveform Depending on UAO



- Figure 71 shows 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.
  - The OUTPOL in TIMERn_CR1 register is applied when the timer counter is cleared after setting. (Applied after the current cycle operation)

Figure 71. Operation Waveform According to STARTLVL and OUTPOL



10.3.5.3 Notes on Counter Mode Operation

1. If the timer counter is cleared during timer operation, the output of the timer pin (TnIO) 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) 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.

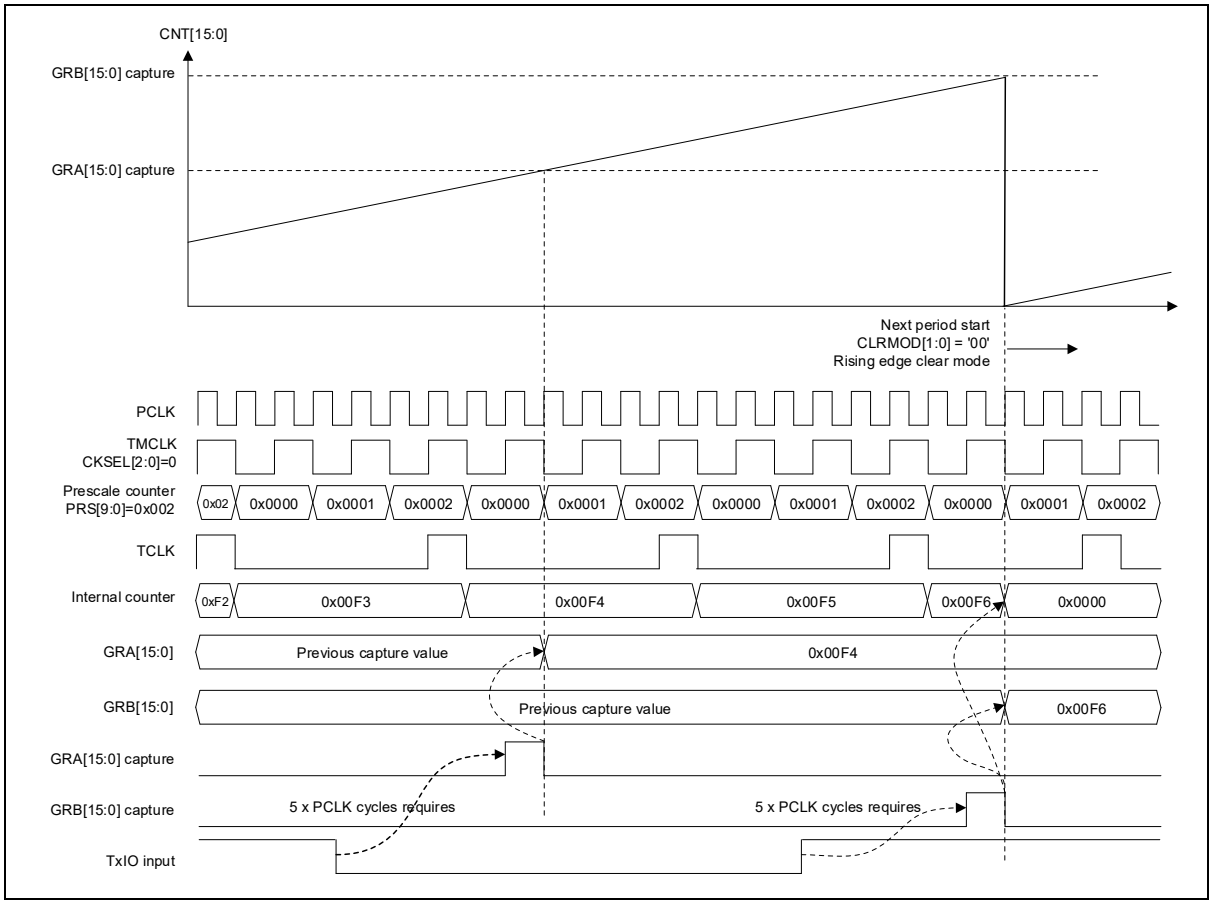
10.3.5.4 Counter Mode Register Setting Sequence

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 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. Write '1' to the IOSEL in TIMERn_CR1 to set the timer input/output (TnIO) 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 72 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 72. 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.

The example timing diagram in Figure 72 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) and then set the IOSEL in TIMERn_CR1 to ‘0’.

Table 80. Operation Status by Capture Mode

CLRMOD[1:0] in TIMERn_CR1	Clear Mode	TIMERn_GRA Data	TIMERn_GRB Data
00	Rising edge clear	Falling edge capture	Rising edge capture
01	Falling edge clear	Rising edge capture	Falling edge capture
10	Both edge clear	Falling edge capture	Rising edge capture
11	Disable clear	Rising edge capture	Falling edge capture

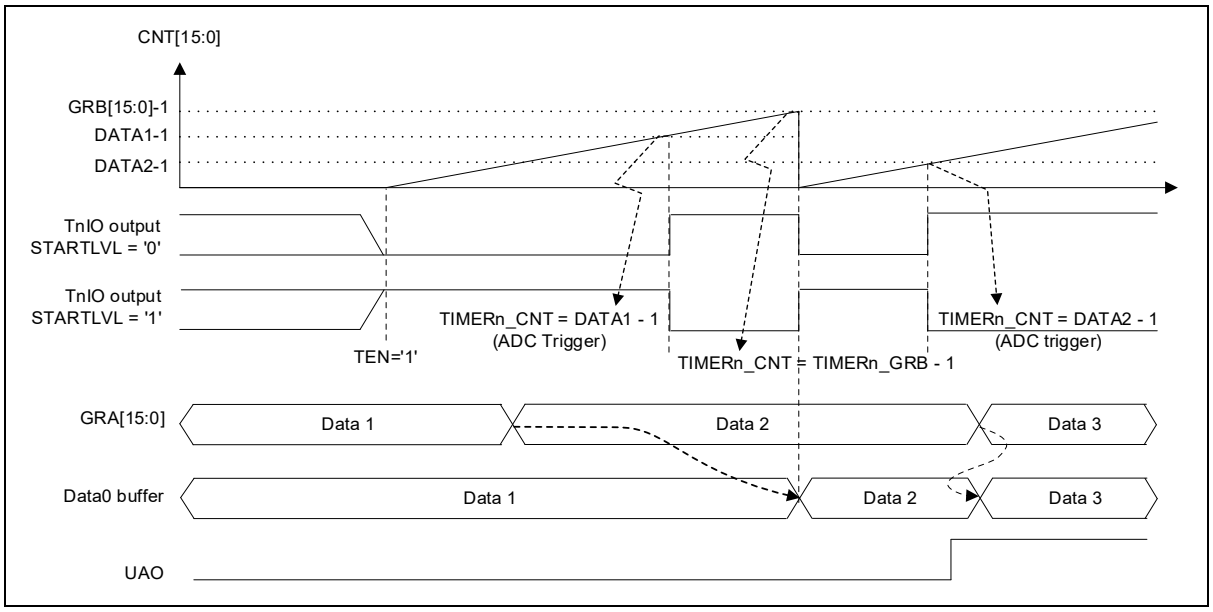
10.3.6.1 Capture Mode Register Setting Sequence

1. To enable the system access, write a reserved value to the corresponding bit of the System access enable register (SCU_SYSTEN).
2. To enable the timer module, write ‘1’ to the corresponding bit of the peripheral enable register1 (SCU_PER1) and the peripheral clock enable register1 (SCU_PCER1).
3. Initialize the TIMERn_GRA and TIMERn_GRB registers to zero.
4. Write ‘0’ to the TIMERn_CNT register to initialize the timer counter.
5. To set the timer mode, enter the corresponding mode set value in the MODE[1:0] 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) pin.
9. For the clear 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 73 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.

Figure 73. Timing Diagram of PWM Output Operation



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

- Period = TCLK period × TIMERn_GRB value
- Duty = TCLK period × TIMERn_GRA value

If the TIMERn_GRB value is 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 zero and less than or equal to the TIMERn_GRB value.

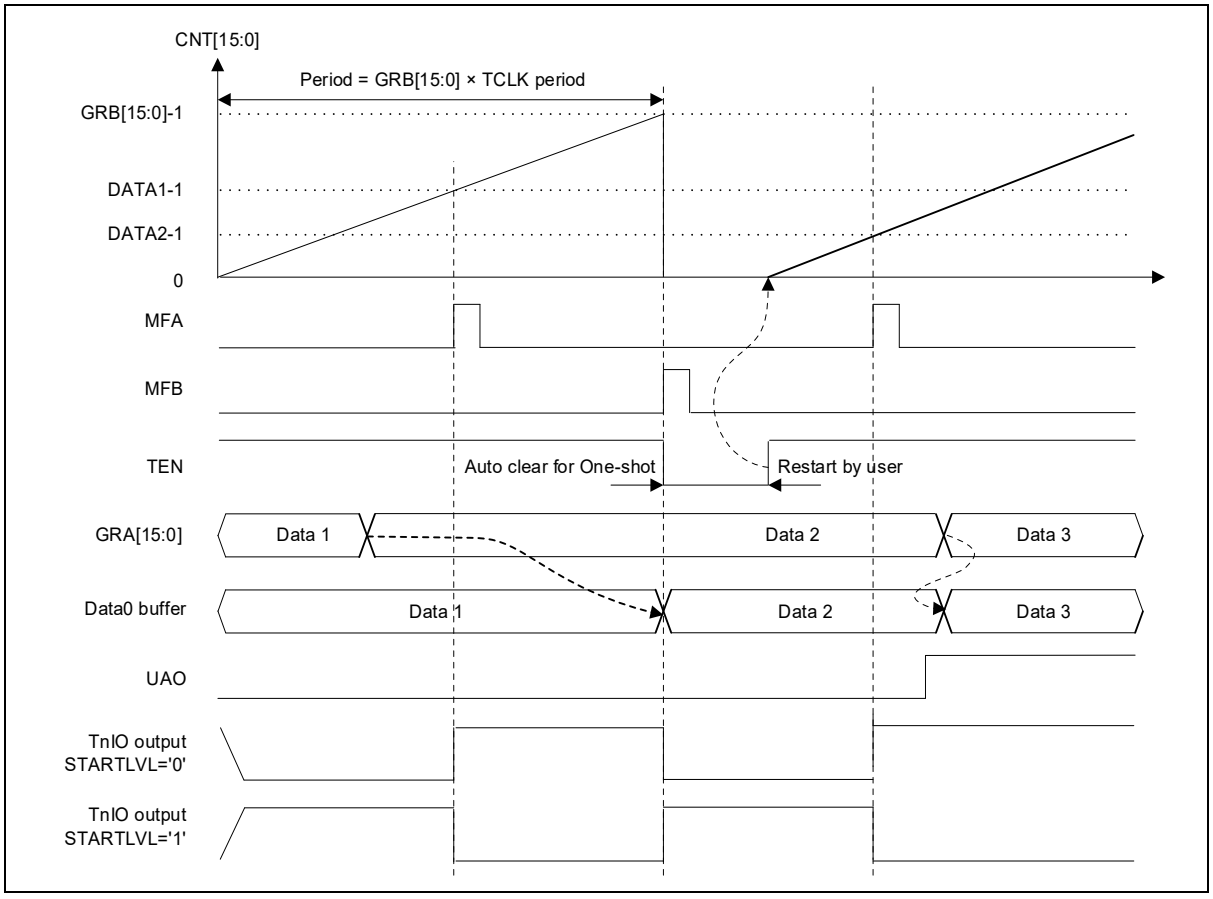
10.3.7.2 PWM Mode Register Setting Sequence

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 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. Write '1' to the IOSEL in TIMERn_CR1 to set the timer input/output (TnIO) 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 74 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 74. Timing Diagram of One-Shot Mode Operation



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

- Period = TCLK period × TIMERn_GRB value
- Duty = TCLK period × TIMERn_GRA value

If the TIMERn_GRB value is 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:
  - 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:
  - 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 81. One-Shot Mode Operation Setting 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 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. Write ‘1’ to the IOSEL in TIMERn_CR1 to set the timer input/output (TnIO) 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 75, Figure 76, and Figure 77 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 75. Timing Diagram of ADC Triggering (TRGMOD[1:0] = '00')

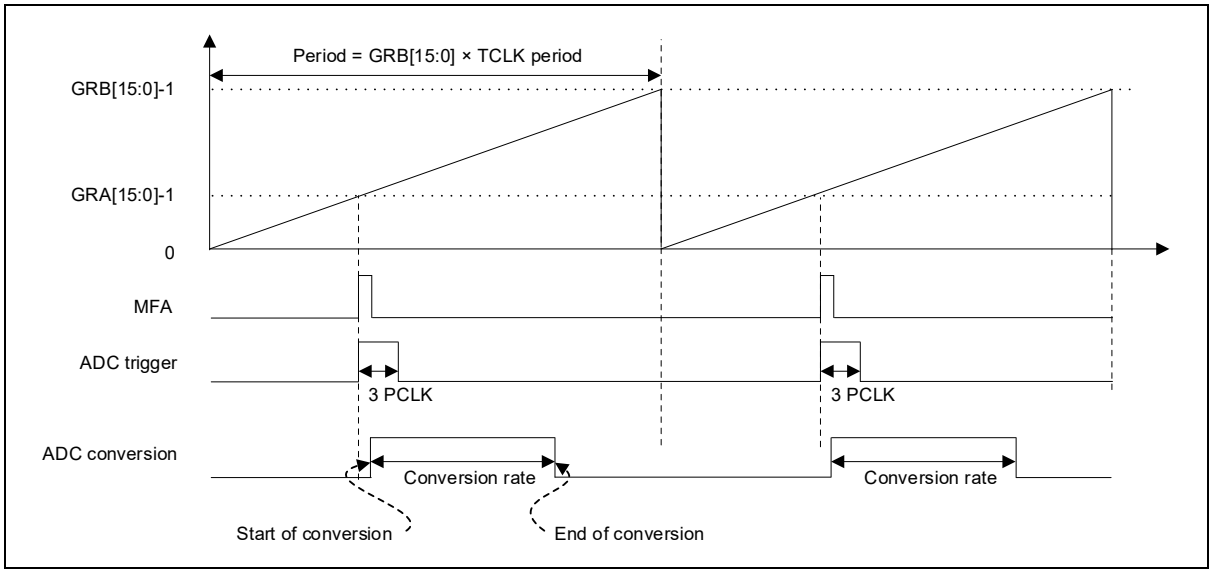


Figure 76. Timing Diagram of ADC Triggering (TRGMOD[1:0] = '10')

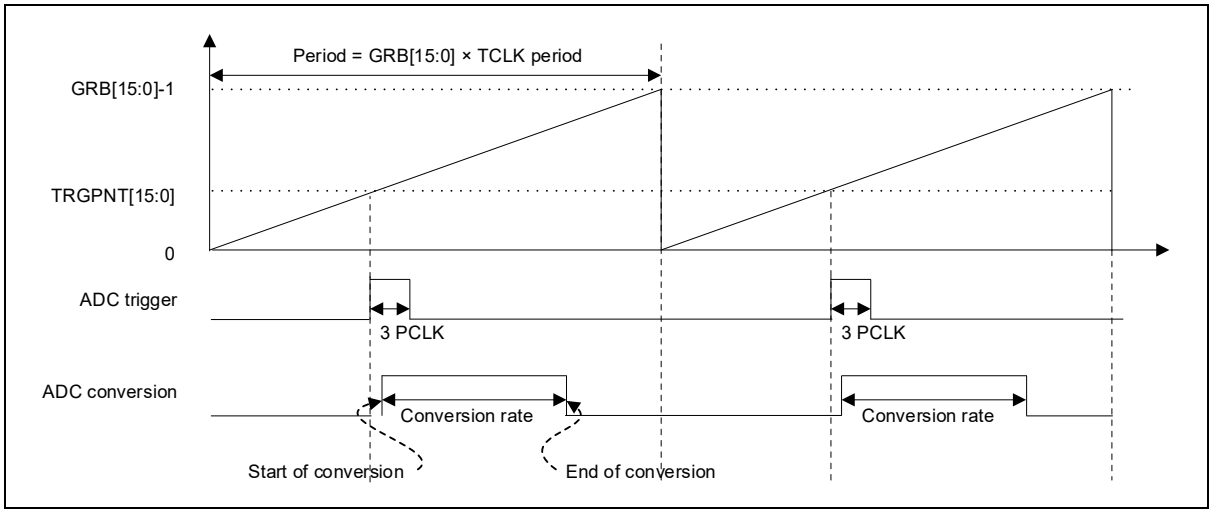
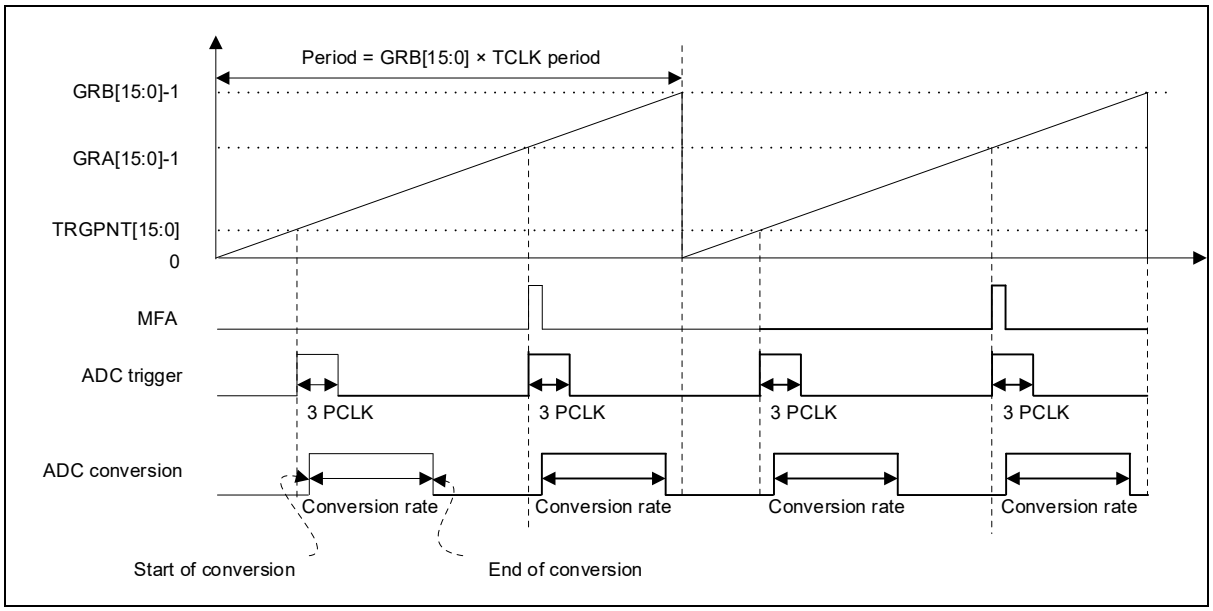


Figure 77. Timing Diagram of ADC Triggering (TRGMOD[1:0] = '11')



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.

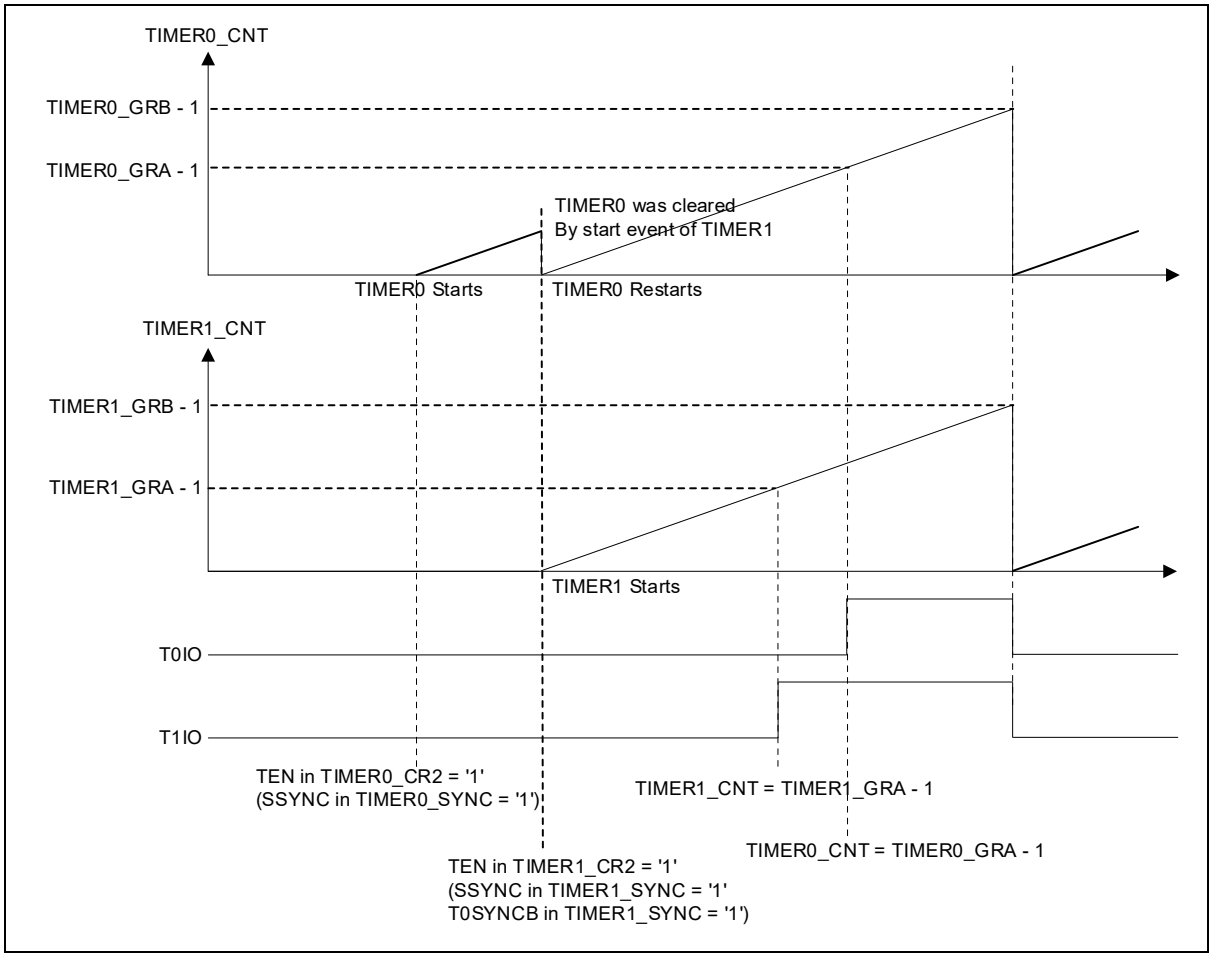
10.3.9.2 Register Setting Sequence for ADC Trigger Operation

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 78 shows synchronous PWM generation.

Figure 78. Timer Synchronization Example (SSYNC = '1', CSYNC = '1')



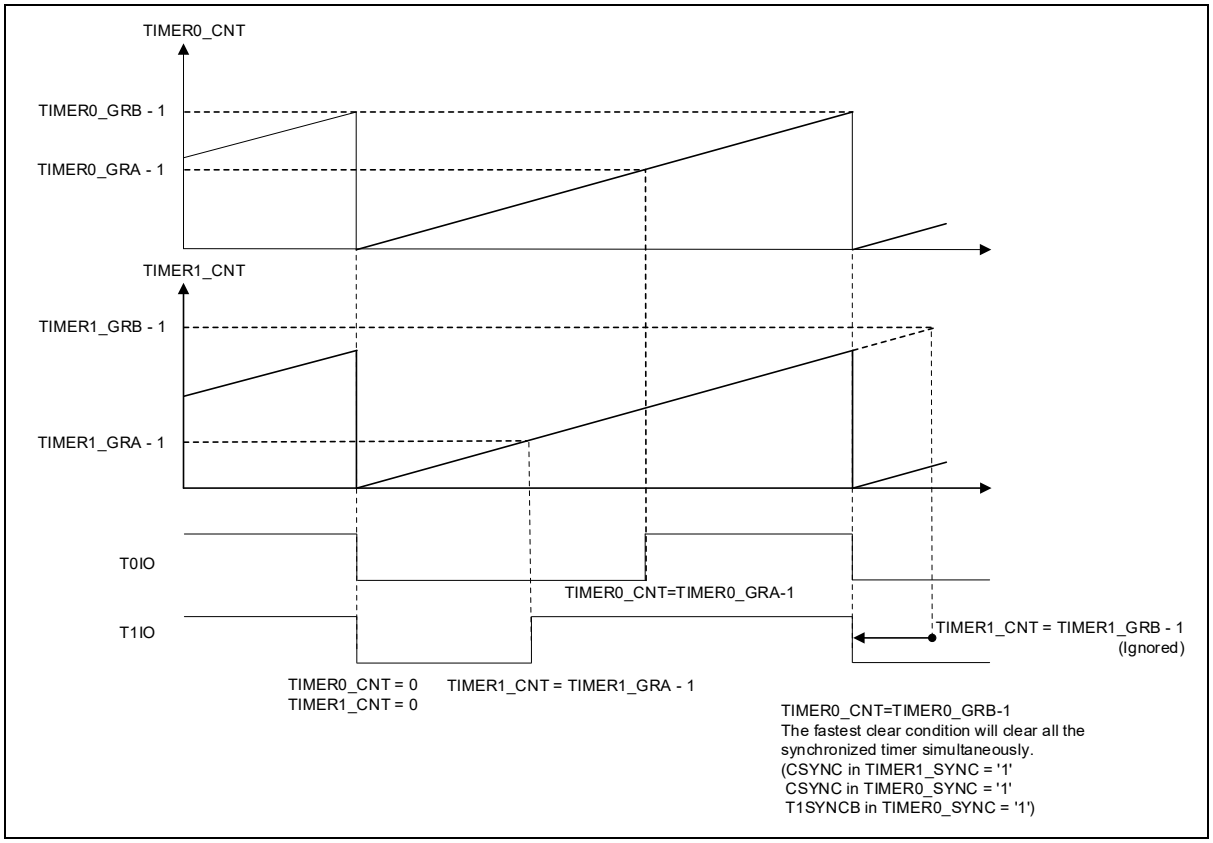
The SSYNC in TIMERn_SYNC controls start synchronization with other timer modules. In the above figure, the TIMER0 (Slave Timer) waveform shows that the timer's start is synchronized to the start of the signal from TIMER1 (Master Timer).

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 79. Timer Synchronization Example (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 clear is synchronized to the clear of the signal from the TIMER0 (Master Timer).

For this CSYNC function in the figure above, both the Master and Slave Timers' synchronous clear bits must be set enabled: In the Master Timer's TIMER0_SYNC register, the T1SYNCB and CSYNC must be set to '1'. In the Slave Timer's TIMER1_SYNC register, the CSYNC must be set to '1'.

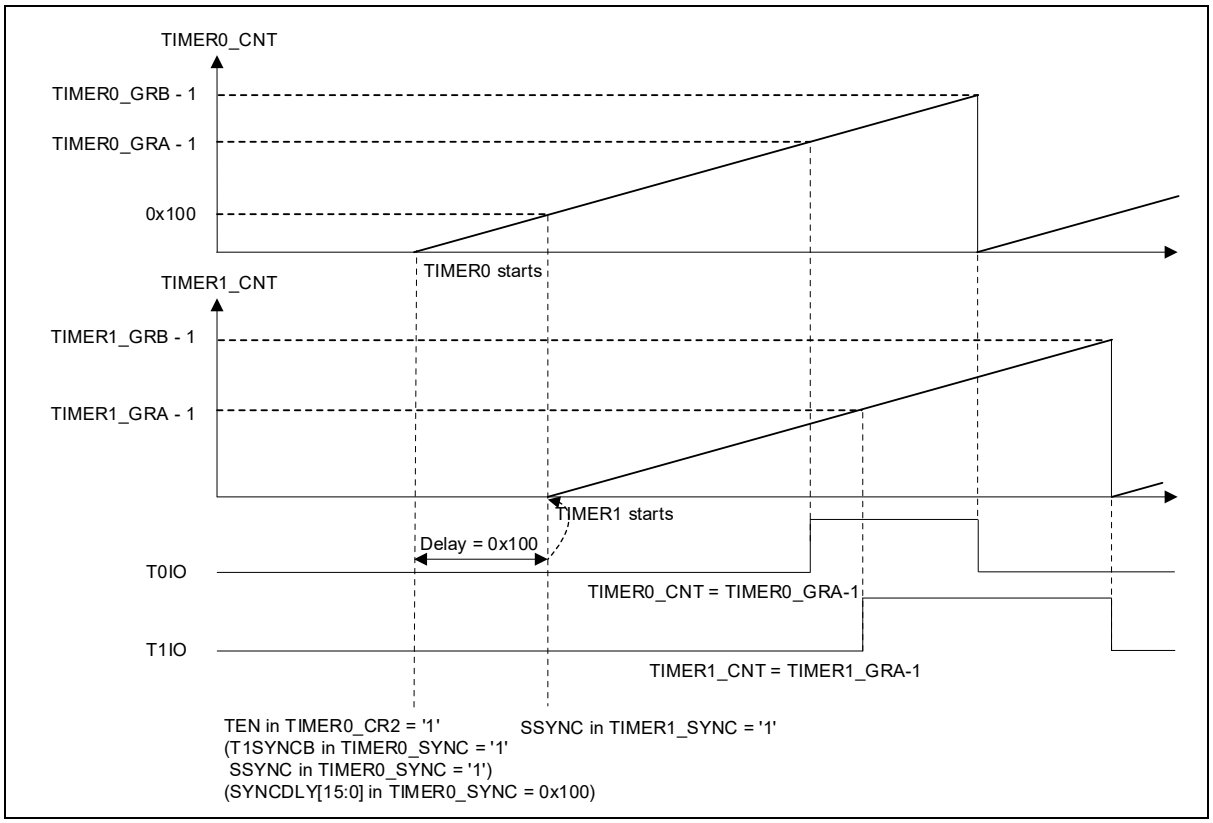
If only the Master Timer's synchronous clear bit is set enabled for the Slave Timer while the Slave's synchronous clear bit is disabled, the Slave Timer can run independently without being affected by the Master Timer's clear synchronization signal.

10.3.10.1 Delayed Timer Synchronization

The PWM delayed synchronization function is used between timers within the same group (Group 1: T0, T1, ..., T9); 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 80. Delayed Timer Synchronization Example (SSYNC = '1')



For example as shown in Figure 80, 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 TIMERn_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 81 and Figure 82 show the operation waveform of the SSYNC and CSYNC signals in cascaded structure. The timers in the figures below are cascaded in the order of TIMER0 → TIMER1 → TIMER3.

Figure 81. SSYNC Operation Example

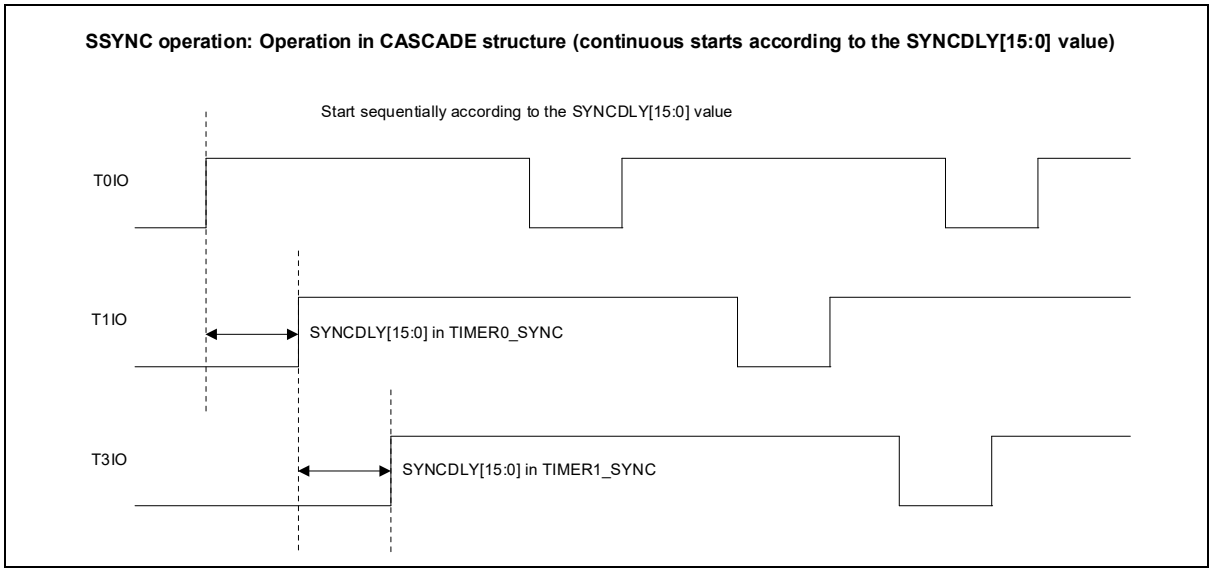
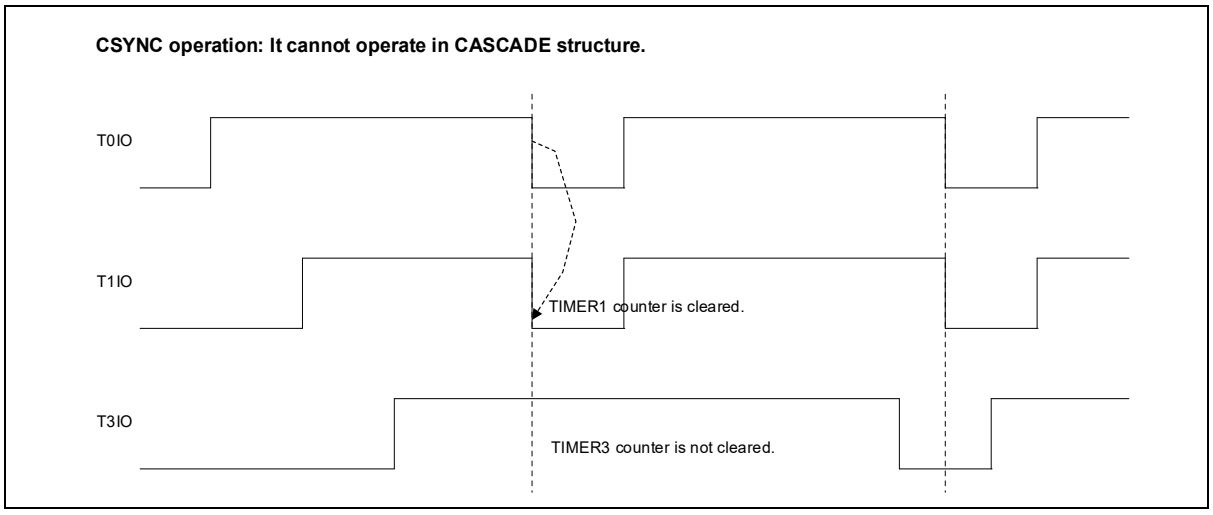


Figure 82. CSYNC Operation Example



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) 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 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 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) 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) are disabled.

10.3.13 Interrupts

The TIMERn can generate interrupts, as shown in Table 82.

Table 82. Interrupt Requests

Interrupt event	Event flag	Enable control bit	Interrupt clear method
Overflow interrupt	OVF in TIMERn_SR	OVIE in TIMERn_IER	Writing a '1' to OVF in TIMERn_SR
GRB match interrupt	MFB in TIMERn_SR	MBIE in TIMERn_IER	Writing a '1' to MFB in TIMERn_SR
GRA match interrupt	MFA in TIMERn_SR	MAIE in TIMERn_IER	Writing a '1' to MFA in TIMERn_SR

NOTE:

1. When using overflow interrupt, set the TIMERn_GRA value and TIMERn_GRB value to zero. (Especially if the TIMERn_GRB value is not set to zero, OVF in TIMERn_SR will not be generated.)

10.4 Timer Registers

The base address of 16-bit timer is described in the following:

Table 83. Base Address of 16-bit Timer

Name	Base Address
TIMER0	0x4000_3000
TIMER1	0x4000_3040
TIMER2	0x4000_3080
TIMER3	0x4000_30C0
TIMER4	0x4000_3100
TIMER5	0x4000_3140
TIMER6	0x4000_3180
TIMER7	0x4000_31C0
TIMER8	0x4000_3200
TIMER9	0x4000_3240

Table 84. 16-bit Timer Register Map

Name	Offset	Type	Description	Reset Value	Ref.
TIMERn_CR1	0x0000	RW	Timer n Control Register 1	0x0000_0000	10.4.1
TIMERn_CR2	0x0004	RW	Timer n Control Register 2	0x0000_0000	10.4.2
TIMERn_PRS	0x0008	RW	Timer n Prescaler Register	0x0000_0000	10.4.3
TIMERn_GRA	0x000C	RW	Timer n General Data Register A	0x0000_0000	10.4.4
TIMERn_GRB	0x0010	RW	Timer n General Data Register B	0x0000_0000	10.4.5
TIMERn_CNT	0x0014	RW	Timer n Counter Register	0x0000_0000	10.4.6
TIMERn_SR	0x0018	RWC1	Timer n Status Register	0x0000_0000	10.4.7
TIMERn_IER	0x001C	RW	Timer n Interrupt Enable Register	0x0000_0000	10.4.8
TIMERn_TRGPNT	0x0020	RW	Timer n Trigger Point Register	0x0000_0000	10.4.9
TIMERn_SYNC	0x0024	RW	Timer n Sync Configuration Register	0x0000_0000	10.4.10

NOTES:

1. A31C14x has 10 channels of Timer (n = 0 to 9).
2. A31C12x has 7 channels of Timer (n = 0 to 6).

10.4.1 TIMERn_CR1: Timer n Control Register 1

TIMERn_CR1 is a 32-bit register. The timer module must be set appropriately before running the timer. After the use of the timer is specified, the timer can be set in TIMERn_CR1. After this register is set, users can enable or disable the timer by setting the TIMERn_CR2.

TIMER0_CR1=0x4000_3000, TIMER1_CR1=0x4000_3040
TIMER2_CR1=0x4000_3080, TIMER3_CR1=0x4000_30C0
TIMER4_CR1=0x4000_3100, TIMER5_CR1=0x4000_3140
TIMER6_CR1=0x4000_3180, TIMER7_CR1=0x4000_31C0
TIMER8_CR1=0x4000_3200, TIMER9_CR1=0x4000_3240

31 30 29 28 27 26 25 24	23 22 21 20 19 18 17 16	15 14	13	12	11	10 9	8	7	6 5 4	3 2	1 0
Reserved		TRGMOD[1:0]	UAO	OUTPOL	IOSEL	Reserved	ADCTRGEN	STARTLVL	CKSEL[2:0]	CLRMOD[1:0]	MODE[1:0]
-		00	0	0	0	-	0	0	000	00	00
-		RW	RW	RW	RW	-	RW	RW	RW	RW	RW

15 14	TRGMOD[1:0]	ADC trigger mode selection	
		0X	Selects TIMERn_GRA value trigger mode (normal mode).
		10	Selects TIMERn_TRGPNT value trigger mode.
		11	Triggers both TIMERn_GRA and TIMERn_TRGPNT values.
13	UAO	GRA/GRB update mode selection	
		0	The value written to TIMERn_GRA or TIMERn_GRB is applied after the current period.
		1	The value written to TIMERn_GRA or TIMERn_GRB is applied in the current period.
12	OUTPOL	Timer output polarity	
		0	General output
		1	General output inversion
11	IOSEL	TnIO pin configuration	
		0	Sets the pin as an input port
		1	Sets the pin as an output port
8	ADCTRGEN	Whether 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
		100	EXT_CLK (SCU_MCCR3)(1)(2)
		101	PCLK/1(2)
		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 0	MODE[1:0]	Timer operation mode control	
		00	Normal periodic mode
		01	PWM mode
		10	One-shot mode
		11	Capture mode

NOTES:

1. If the EXT_CLK is set in CKSEL[2:0] of TIMERn_CR register, the clock speed must satisfy PCLK ≥ TMCLK × 3.
2. PCLK/1 is supported only on the A31C12x series. On the A31C14x series, EXT_CLK is selected when CKSEL[2:0] is set to ‘1’00 or 101.

10.4.2 TIMERn_CR2: Timer n Control Register 2

TIMERn_CR2 is a 32-bit register and is used to control the timer's operation. Before starting the timer, users must set the TCLR in TIMERn_CR2 to '1' to clear the timer count register.

TIMER0_CR2=0x4000_3004, TIMER1_CR2=0x4000_3044
TIMER2_CR2=0x4000_3084, TIMER3_CR2=0x4000_30C4
TIMER4_CR2=0x4000_3104, TIMER5_CR2=0x4000_3144
TIMER6_CR2=0x4000_3184, TIMER7_CR2=0x4000_31C4
TIMER8_CR2=0x4000_3204, TIMER9_CR2=0x4000_3244

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

1	TCLR	Clearing of the timer count register(1)	
		0	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. Users must set the TCLR bit to '1' before starting the timer to prevent a time error.

10.4.3 TIMERn_PRS: Timer n Prescaler Register

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

TIMER0_PRS=0x4000_3008, TIMER1_PRS=0x4000_3048
TIMER2_PRS=0x4000_3088, TIMER3_PRS=0x4000_30C8
TIMER4_PRS=0x4000_3108, TIMER5_PRS=0x4000_3148
TIMER6_PRS=0x4000_3188, TIMER7_PRS=0x4000_31C8
TIMER8_PRS=0x4000_3208, TIMER9_PRS=0x4000_3248

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

9 0	PRS[9:0]	Prescaler value for the counter clock TCLK = TMCLK / (PRS[9:0] + 1) (TMCLK = Timer input clock selected by the setting of CKSEL[2:0] in TIMERn_CR1)

10.4.4 TIMERn_GRA: Timer n General Data Register A

The TIMERn_GRA register is 16 bits wide.

TIMER0_GRA=0x4000_300C, TIMER1_GRA=0x4000_304C
TIMER2_GRA=0x4000_308C, TIMER3_GRA=0x4000_30CC
TIMER4_GRA=0x4000_310C, TIMER5_GRA=0x4000_314C
TIMER6_GRA=0x4000_318C, TIMER7_GRA=0x4000_31CC
TIMER8_GRA=0x4000_320C, TIMER9_GRA=0x4000_324C

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

15 0	GRA[15:0]	Timer general register A This register is used for different purposes depending on the operating mode. **Period/PWM/One-shot Modes:** The target count value is stored in the register. If the counter value matches the TIMERn_GRA value, the counter is cleared to restart or stop running. When the timer is cleared, TnIO outputs the STARTLVL in TIMERn_CR1 value. If the counter value matches the register's value, the MFA in TIMERn_SR is triggered. When the GRA interrupt is triggered or clearing is demanded, the counter value is copied to internal data buffer 0. In PWM mode, the GRA[15:0] value represents the duty value. **Capture mode:** - In rising-edge clear mode, the register stores the counter value captured on the falling edge of the signal at port TnIO. - In falling-edge clear mode, the register stores the counter value captured on the rising edge of the signal at port TnIO.

10.4.5 TIMERn_GRB: Timer n General Data Register B

The TIMERn_GRB register is 16 bits wide.

TIMER0_GRA=0x4000_3010, TIMER1_GRA=0x4000_3050
TIMER2_GRA=0x4000_3090, TIMER3_GRA=0x4000_30D0
TIMER4_GRA=0x4000_3110, TIMER5_GRA=0x4000_3150
TIMER6_GRA=0x4000_3190, TIMER7_GRA=0x4000_31D0
TIMER8_GRA=0x4000_3210, TIMER9_GRA=0x4000_3250

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

15 0	GRB[15:0]	Timer general register B This register is used for different purposes depending on the operating mode. **Periodic mode:** It should be set the same as the TIMERn_GRA **PWM/One-shot mode:** The target count value is stored in the register. If the counter value matches the register's value, the MFB in TIMERn_SR is triggered. In PWM mode, the TIMERn_GRB value represents the period value. **Capture mode:** In rising-edge clear mode, the register stores the counter value captured on the rising edge of the signal at port TnIO. (The opposite edge to that of TIMERn_GRA) In falling-edge clear mode, the register stores the counter value captured on the falling edge of the signal at port TnIO. (The opposite edge to that of TIMERn_GRA)

10.4.6 TIMERn_CNT: Timer 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
TIMER8_CNT=0x4000_3214, TIMER9_CNT=0x4000_3254

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

15 0	CNT[15:0]	Timer count value	
		R	Reads the current timer count.
		W	Sets the count value.

10.4.7 TIMERn_SR: Timer n Status Register

The TIMERn_SR register shows current timer status. This register is 8 bits wide.

TIMER0_SR=0x4000_3018, TIMER1_SR=0x4000_3058
TIMER2_SR=0x4000_3098, TIMER3_SR=0x4000_30D8
TIMER4_SR=0x4000_3118, TIMER5_SR=0x4000_3158
TIMER6_SR=0x4000_3198, TIMER7_SR=0x4000_31D8
TIMER8_SR=0x4000_3218, TIMER9_SR=0x4000_3258

31 30 29 28 27 26 25 24	23 22 21 20 19 18 17 16	15 14 13 12 11 10 9 8	7 6 5 4 3	2	1	0
Reserved				MFA	MFB	OVF
-				0	0	0
-				RWC1	RWC1	RWC1

2	MFA	TIMERn_GRA match flag	
		0	No TIMERn_GRA match has been detected.
		1	A TIMERn_GRA match has been flagged.
1	MFB	TIMERn_GRB match flag	
		0	No TIMERn_GRB match has been detected.
		1	A TIMERn_GRB match has been flagged.
0	OVF	Counter overflow flag	
		0	No overflow event has occurred.
		1	A counter overflow event has been flagged.

10.4.8 TIMERn_IER: Timer 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
TIMER8_SR=0x4000_321C, TIMER9_SR=0x4000_325C

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

2	MAIE	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 n Trigger Point Register

The TIMERn_TRGPNT register is used to trigger the ADC at the desired timer counter point. This register is 16 bits wide.

TIMER0_TRGPNT=0x4000_3020, TIMER1_TRGPNT=0x4000_3060
TIMER2_TRGPNT=0x4000_30A0, TIMER3_TRGPNT=0x4000_30E0
TIMER4_TRGPNT=0x4000_3120, TIMER5_TRGPNT=0x4000_3160
TIMER6_TRGPNT=0x4000_31A0, TIMER7_TRGPNT=0x4000_31E0
TIMER8_TRGPNT=0x4000_3220, TIMER9_TRGPNT=0x4000_3260

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

15 0	TRGPNT[15:0]	Timer trigger point value
		Read: Reads the current trigger point value.
		Write: Sets the trigger point value.

10.4.10 TIMERn_SYNC: Timer n Sync Configuration Register

TIMERn_SYNC is used to synchronize TIMER0 to TIMER4 to start or clear the respective timers. For example, it needs to set '15' to SYNCDLY[15:0] in TIMER0_SYNC register and set '1' to T1SYNCB in TIMER0_SYNC register if user wants to start TIMER1 after 15 counts after starting TIMER0.

It has no effect to set '1' to T0SYNCB in TIMER0_SYNC register. And Syncing TIMER0 from TIMER1 after TIMER0 to TIMER1 has started syncing has no effect.

TIMER0_SYNC=0x4000_3024, TIMER1_SYNC=0x4000_3064
TIMER2_SYNC=0x4000_30A4, TIMER3_SYNC=0x4000_30E4
TIMER4_SYNC=0x4000_3124, TIMER5_SYNC=0x4000_3164
TIMER6_SYNC=0x4000_31A4, TIMER7_SYNC=0x4000_31E4
TIMER8_SYNC=0x4000_3224, TIMER9_SYNC=0x4000_3264

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved		T9SYNCB	T8SYNCB	T7SYNCB	T6SYNCB	T5SYNCB	T4SYNCB	T3SYNCB	T2SYNCB	T1SYNCB	T0SYNCB	Reserved		SSYNC	CSYNC	SYNCDLY[15:0]															
-		0	0	0	0	0	0	0	0	0	0	-		0	0	0x0000															
-		RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	-		RW	RW	RW															

20+x	TxSYNCB[3]	Synchronization settings to Tx: 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. If users enable synchronization with TIMER1 in TIMER0_SYNC and synchronization with TIMER2 in TIMER1_SYNC, TIMER0, TIMER1 and TIMER2 will start successively with the delay time set at SYNCDLY[15:0] between them.

10.4.11 Timer Register Map Summary

Table 85. 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.	T9SYNCB	T8SYNCB	T7SYNCB	T6SYNCB	T5SYNCB	T4SYNCB	T3SYNCB	T2SYNCB	T1SYNCB	T0SYNCB	Res.	Res.	SSYNC	CSYNC	SYNCDLY[15:0]															
	Reset value			0	0	0	0	0	0	0	0	0	0			0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

11. FRT

11.1 FRT Introduction

A31C14x and A31C12x series 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 mode.

11.2 FRT Main Features

The FRT has the main features shown below:

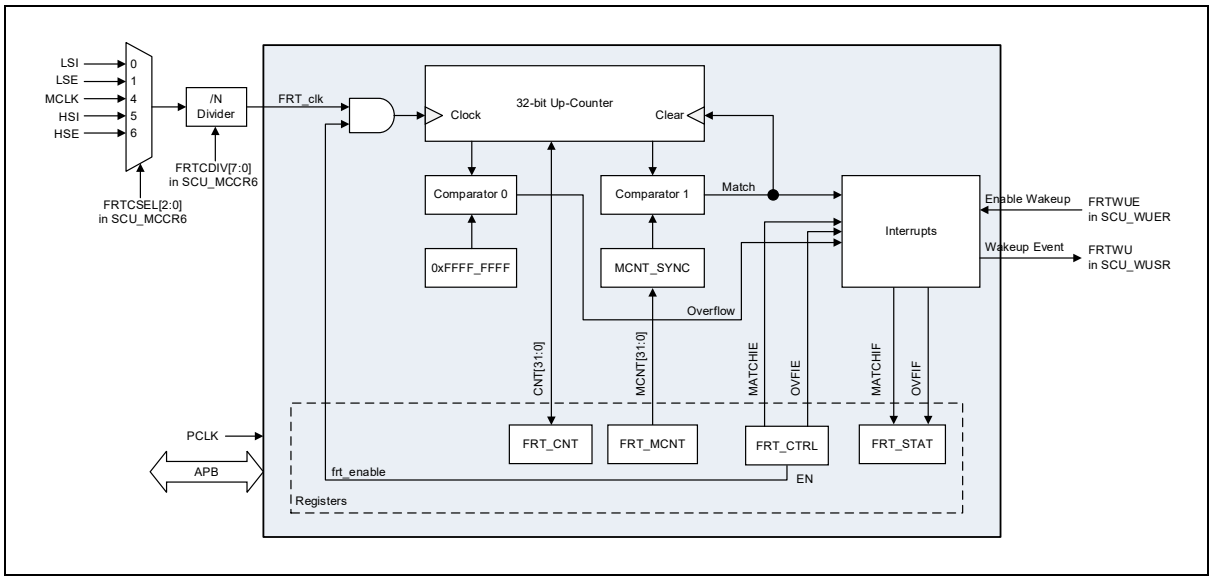
- 32-bit up count timer‘s two operations
  - Free-Run Timer Mode operation
  - Periodic Timer Mode operation based on the setup period
- FRT overflow interrupt and match interrupt functions
- Selectable FRT input clocks
  - Input clock source can be selected using the SCU_MCCR6 register: LSI, LSE, MCLK,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

11.3 FRT Functional Description

11.3.1 FRT Block Diagram

Figure 83 shows a block diagram of the FRT.

Figure 83. 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 nor operates normally.

An external clock source can be set as a clock source for the FRT. The external clock source can be selected as the FRT’s clock source when the 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 A31C14x and A31C12x series 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.

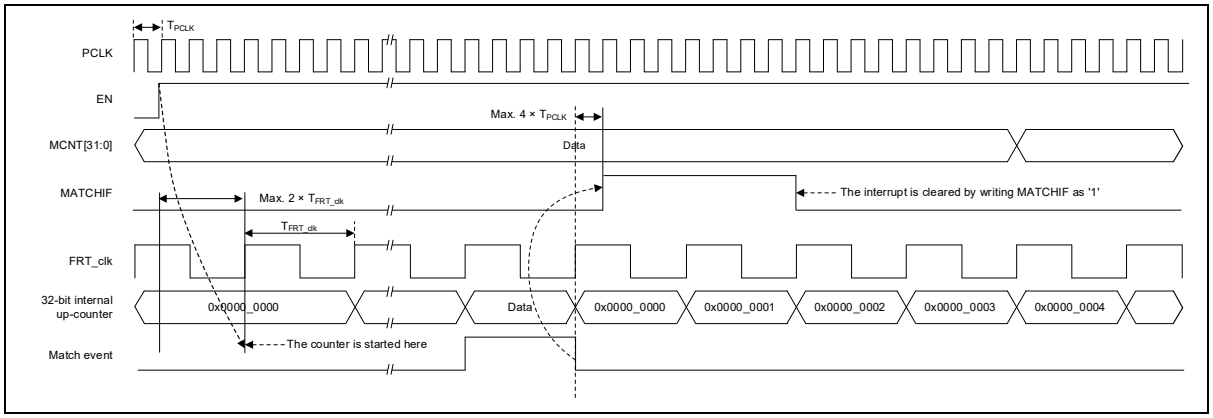
When the FRT runs in Free Run Timer Mode, the OVFIE in FRT_CTRL register determines whether to enable or disable the overflow interrupt. This bit does not affect the match interrupt.

11.3.3.1 Match Interrupt Operation

Figure 84 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 to ‘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 84. Match Interrupt Operation Timing Diagram



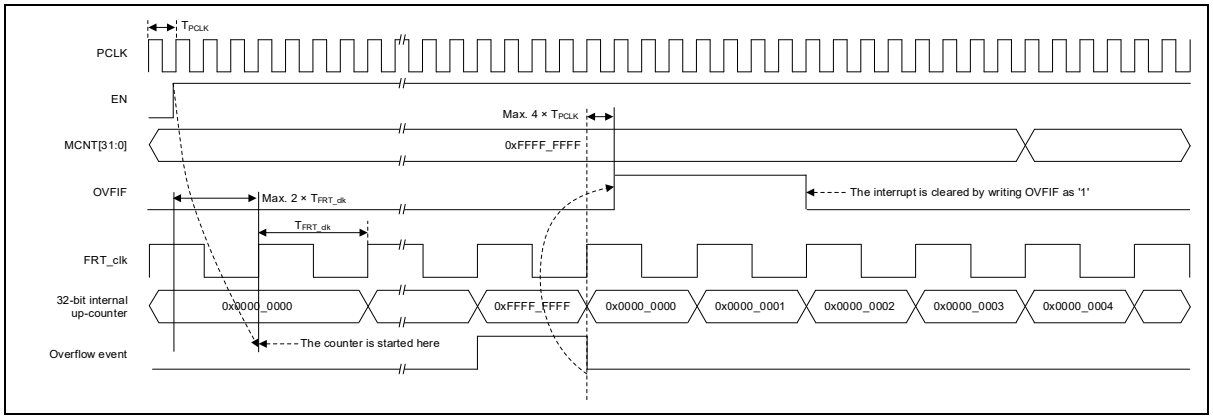
The following formula calculates the FRT’s count period in Periodic Timer Mode:

- Period = FRT_clk period × (MCNT + 1)
- Match interrupt time = FRT_clk period × (MCNT + 1)

11.3.3.2 Overflow Interrupt Operation

Figure 85 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 85. Overflow Interrupt Operation Timing Diagram



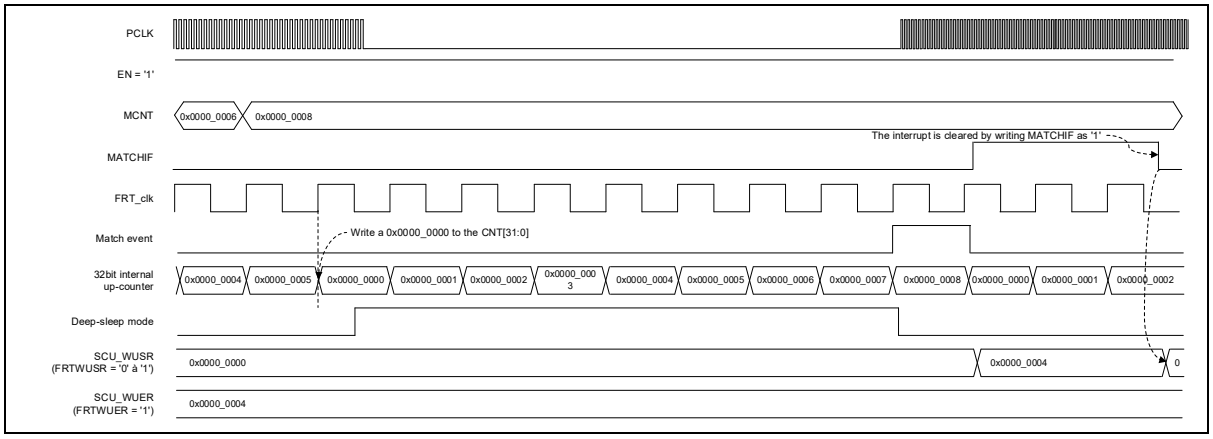
11.3.3.3 Register Setting Procedure by FRT Operation Mode

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 86. 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 zero, which initialize 32-bit internal up-counter.
4. Enter low power mode (DEEP-SLEEP mode)
5. When 32-bit internal up-counter reaches the MCNT[31:0] 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 the same time FRTWU in SCU_WUSR register is also cleared.

11.4 FRT Registers

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

Table 86. Base Address of FRT

Name	Base Address
FRT	0x4000_0600

Table 87. 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.	

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 88. 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	0	0	0	0	0	0	0	0	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	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

12.1 UART Introduction

This built-in UART (Universal Asynchronous Receiver/Transmitter) 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 A31C14x and A31C12x series 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
  - Notification about communication 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.

12.2.1 UART Implementation

Table 89 describes the UART implementation in the A31C14x and A31C12x series.

Table 89. UART Features

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

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.

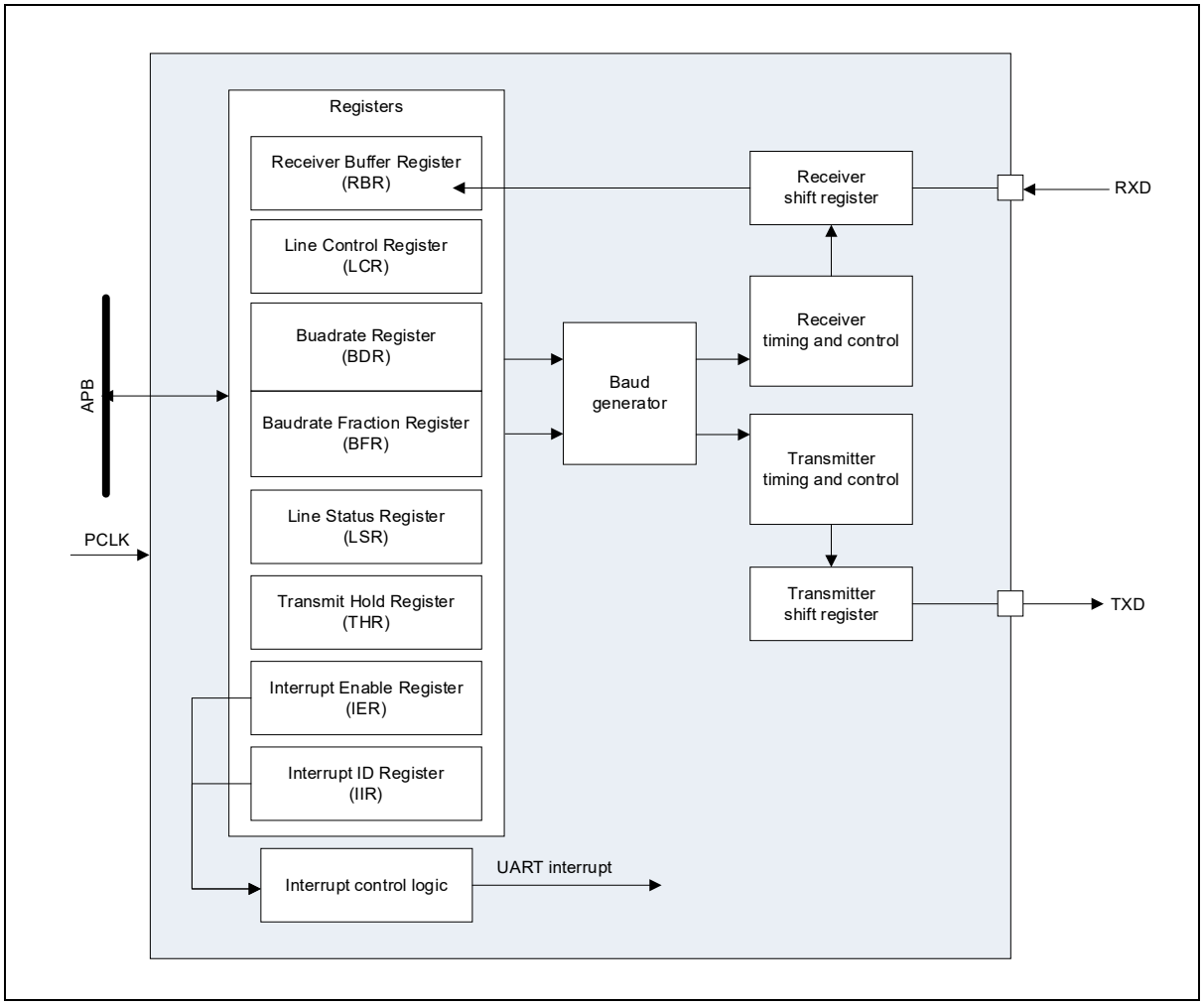
The memory address for the Rx transfers is the target address, whereas the address for Tx transfers is the source address.

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.

12.3.1 UART Block Diagram

Figure 87 shows a block diagram of the UART.

Figure 87. UART Block Diagram



12.3.2 UART Pins and Signals

Table 90 describes external pins assigned for UART.

Table 90. Pin Assignment of UART: External Pins

Pin Name	Type	Description	Package Type							
			64-LQFP	48-LQFP, QFN	44-LQFP	32-QFN, LQFP	28-TSSOP	24-QFN	20-QFN	20-TSSOP
TXD0	O	UART channel 0 transmit output	O	O	O	O	O	O	O	O
RXD0	I	UART channel 0 receive input	O	O	O	O	O	O	O	O
TXD1	O	UART channel 1 transmit output	O	O	O	-	-	-	-	-
RXD1	I	UART channel 1 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 the UARTn_LCR register as described below:

- For 5-bit character length: DLEN[1:0] = '00' (refer to Figure 92)
- For 6-bit character length: DLEN[1:0] = '01' (refer to Figure 91)
- For 7-bit character length: DLEN[1:0] = '10' (refer to Figure 90)
- For 8-bit character length: DLEN[1:0] = '11' (refer to Figure 89)

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 88 shows that the signal value (TXD and RXD) can be inverted, separately for each signal, through UARTn_DCR's data inversion control.

Figure 88. Data Inversion Control Diagram

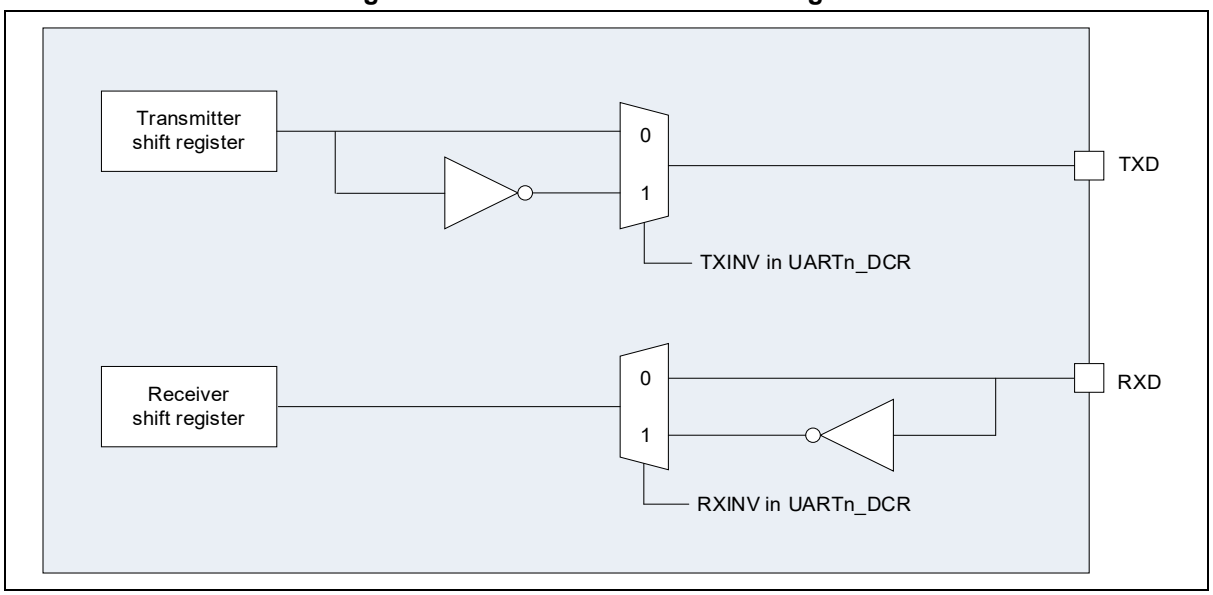


Figure 89 shows a timing diagram of data length programming. A Break condition is interpreted on receiving ‘0’s for longer than the time taken to receive the entire data word (i.e., the sum of the start, data, parity, and stop bits). The transmission and reception clocks are generated when the enable bit is set for the UART peripheral and clock, respectively.

Figure 89. Timing Diagram of 8-bit Data Length

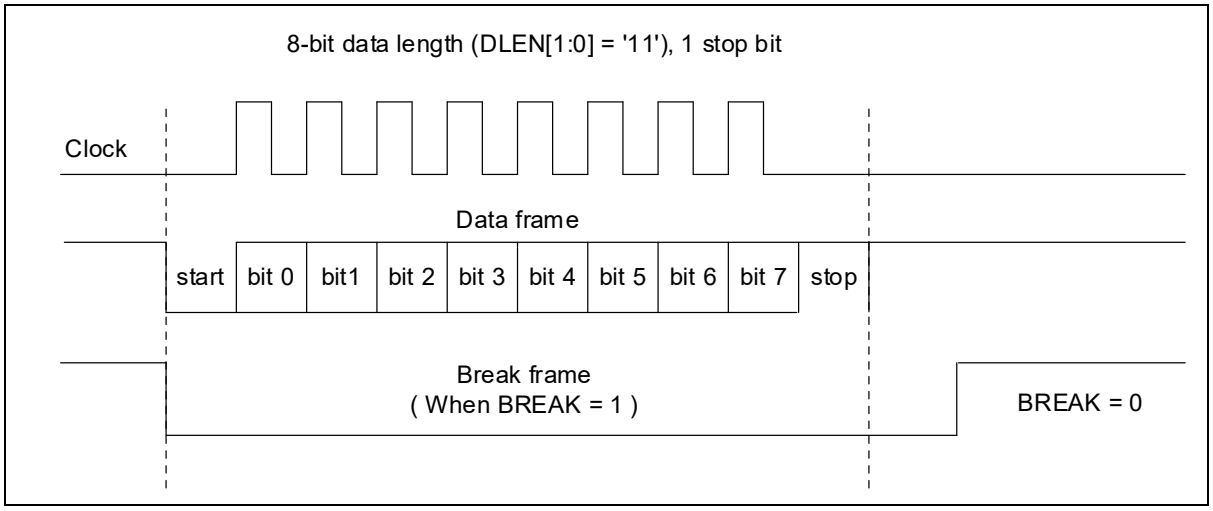


Figure 90. Timing Diagram of 7-bit Data Length

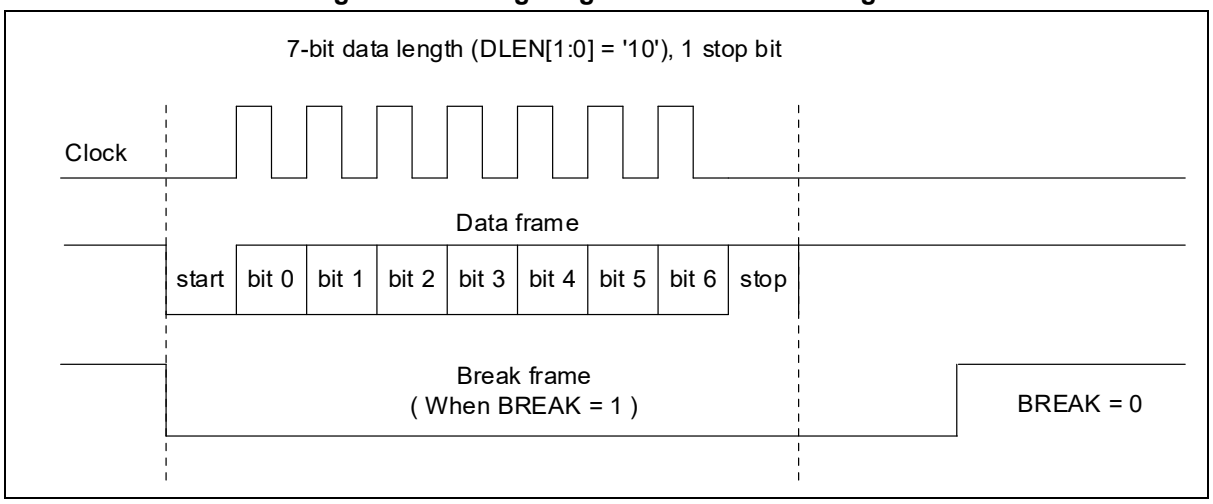


Figure 91. Timing Diagram of 6-bit Data Length

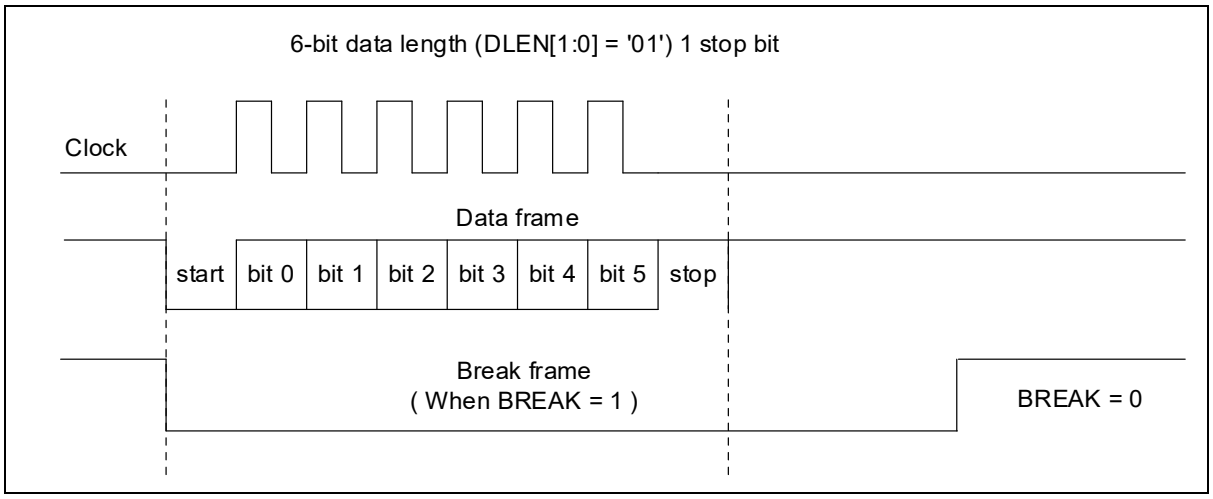
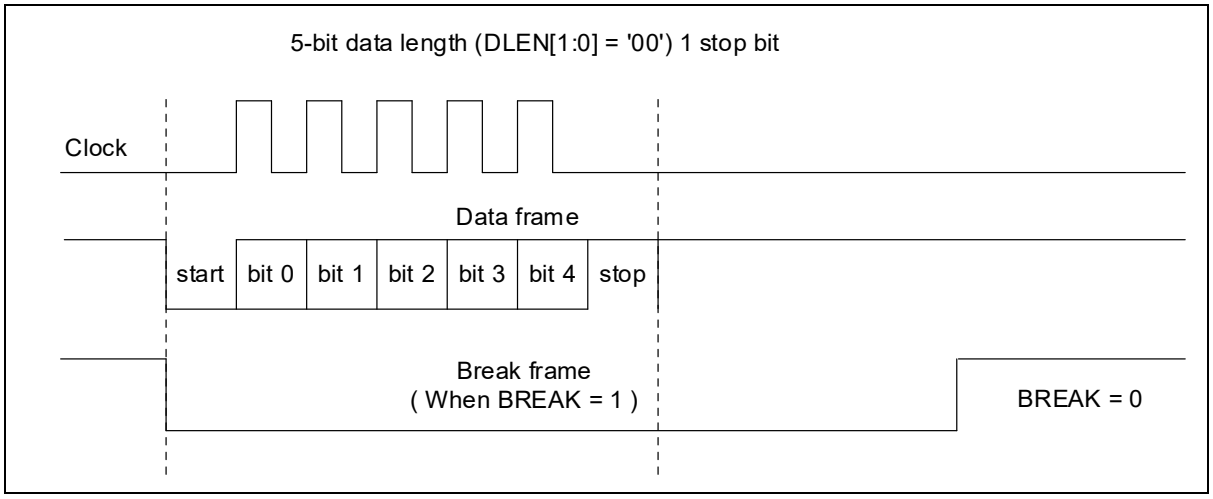


Figure 92. Timing Diagram of 5-bit Data Length



12.3.4 UART Transmitter

12.3.4.1 Data Format

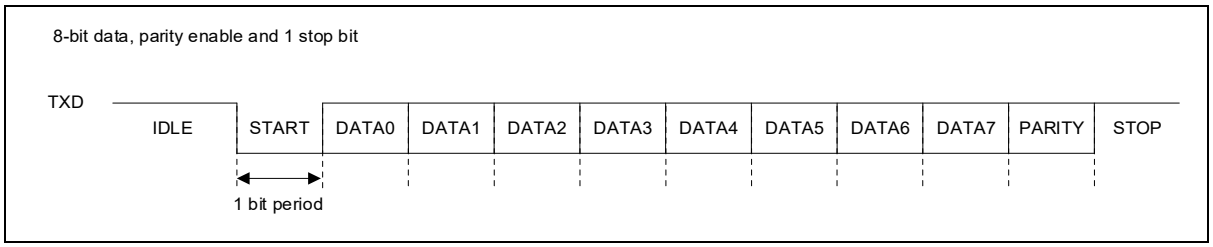
The UART transmitter is a block for transmitting data. For each data word, the start bit, data bits, optional bit, and stop bit are shifted serially in and out of the corresponding register, one bit at a time at the Least Significant Bit (LSB).

The number of data bits can be determined by setting the DLEN[1:0] in UARTn_LCR register. The UART transmitter sends data words of either 5, 6, 7, or 8 bits, based on set value of the DLEN[1:0] in UARTn_LCR register.

The parity bit type can be determined by setting the PARITY and PEN in UARTn_LCR register. If even parity is selected, the parity bit is determined by the bit sum of all the data bits. For odd parity, the parity bit takes the opposite value to that of even parity. The number of stop bits is defined by configuring the STOPBIT in UARTn_LCR register.

Figure 93 shows an example format of transmitting data.

Figure 93. 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.

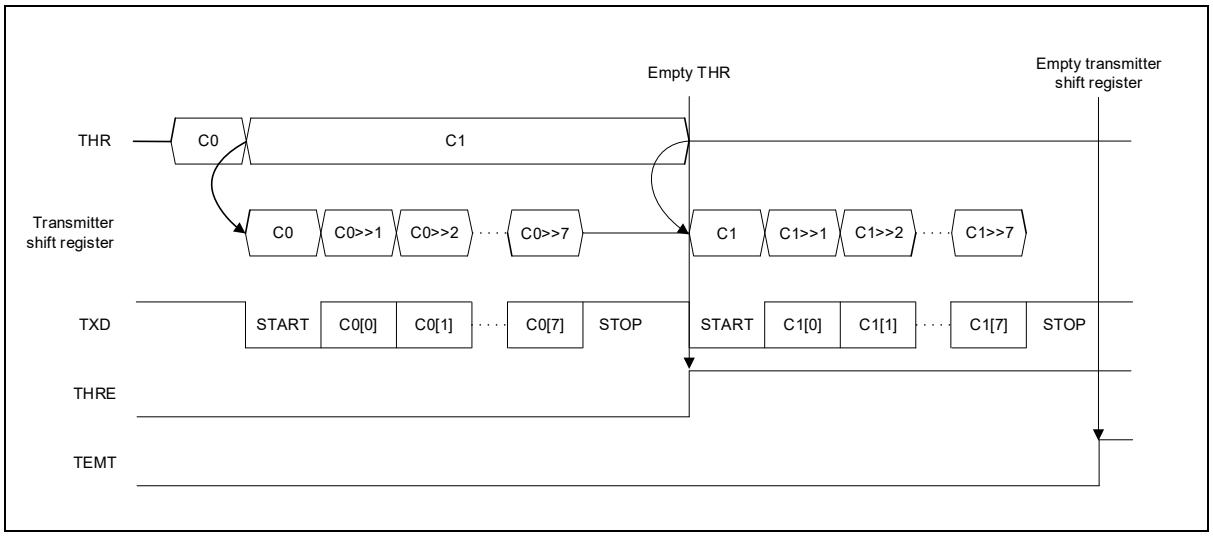
12.3.4.2 Transmit Interrupt

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

When the Transmit data Hold Resistor (UARTn_THR) is empty, the THRE interrupt flag is set to '1'; when the Transmit Shift Register is empty, the TEMT interrupt flag is set to '1'.

Users can select the best interrupt timing for their applications, by referring Figure 94.

Figure 94. Transmit Interrupt Timing Diagram



12.3.4.3 Character Transmission

During UART transmission, data is shifted out at the Least Significant Bit first on the TXD pin. In this mode, the UART transmission register consists of a buffer (UARTn_THR) between the internal bus and the Transmit Shift Register.(1)

Every character is preceded by a start bit which corresponds to a low logic level for one bit period. The character is terminated by a configurable number of stop bits. The number of stop bits can be configured to 1, 1.5 or 2, depending on the STOPBIT bit status.

NOTE:

1. The UART peripheral and clock enable bits must be set in the SCU_PER2 and SCU_PCER2 registers before writing the data to be transmitted.

The UART peripheral and clock enable bits must not be reset during data transmission. Resetting the UART peripheral and clock enable bits during a transfer will freeze the baud rate count and stick the data of the TXD pin. Then, the data frame being transmitted will be broken.

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 UARTn_LCR register.

- 1 stop bit: Default number of stop bits.
- 1.5, 2 stop bits: Number of stop bits that users can configure.

A break transmission is possible to transmit long breaks (Break of length greater than the time of entire data word - i.e., The sum of the start bit, data bits, parity bit, and stop bits).

12.3.4.5 Character Transmission Procedure

Users can use DMA, Interrupts, and Polling to transmit the UART communication packets following the procedure below:

1. Entering data in the UARTn_THR register starts UART communication.
2. Since data in the UARTn_THR register is transferred to the Transmit Shift Register for the communication, updating the UARTn_THR register does not affect the data in communication.
   A. If the UARTn_THR register is updated, the Transmit Shift Register is automatically updated with the data in the UARTn_THR register after entire data in the Transmit Shift Register is output.
   B. The Transmit Shift Register continues to output its updated data for communication.
3. DMA, Interrupts, and Polling can be used to update the UARTn_THR register after confirming the communication is completed.
   A. With the DMA Tx enabled,
      i. The UART block generates the DMA Request signal immediately after data in the UARTn_THR register is transferred to the Transmit Shift Register.
      ii. The DMA updates the memory data with the UARTn_THR register. When this operation is repeated as many times as set in the DMA counter, the DMA Done signal is generated, the IID[2:0] is set to '101', and the DMA Tx interrupt is generated.
   B. Checking if the TXE interrupt is generated or verifying that the IID[2:0] value is '001',
      i. The TXE interrupt is generated when the communication is completed after the UARTn_THR register transfers data and the Transmit Shift Register outputs data. By checking the generation of the TXE interrupt, users can update the UARTn_THR register to restart communication.
      ii. If the IID[2:0] in UARTn_IIR register is set to '001' without the generation of the TXE interrupt, it means that data in the UARTn_THR register has been transferred to the Transmit Shift Register.
      iii. If the UARTn_THR register is not updated with new data, the TXE interrupt is generated when the Transmit Shift Register completes to output its current data.
4. Polling the TEMT and THRE in UARTn_LSR register,
   A. The TEMT bit implies that the Transmit Shift Register is empty.
   B. The THRE bit implies that the UARTn_THR register is empty. (The UARTn_THR register can be updated only when TEMT and THRE bits are enabled.)

12.3.4.6 Single Byte Communication

When data is stored in the UARTn_THR (Transmitter Hold Register) register for transmission, it is automatically transferred to the Transmit Shift Register.

Before the Transmit Shift Register transfers data, the Transmitter outputs a start bit. After the start bit, the Transmit Shift Register shifts one bit at a time to the output of the Transmitter.

After data in the Transmit Shift Register is output by the bit size set in the DLEN[1:0] in UARTn_LCR register, the Transmitter outputs a stop bit to complete the communication.

During the data transmission or after the communication is completed, if data in the UARTn_THR register is updated, the Transmit Shift Register that output data will be updated with new data to start next communication.

The Transmitter, which outputs a stop bit, restarts communication outputting a start bit.

12.3.4.7 Break Condition

Setting the BREAK(1) in UARTn_LCR register transmits a long break (Break of length greater than the time of entire data word - i.e., The sum of the start, data, parity, and stop bits). Whereas writing a '0' to the bit disables the break condition.

NOTE:

1. If a '1' is written to the BREAK bit, a break is sent on the Tx line immediately. Therefore, users must confirm the completion of Tx transmission and set the BREAK bit.

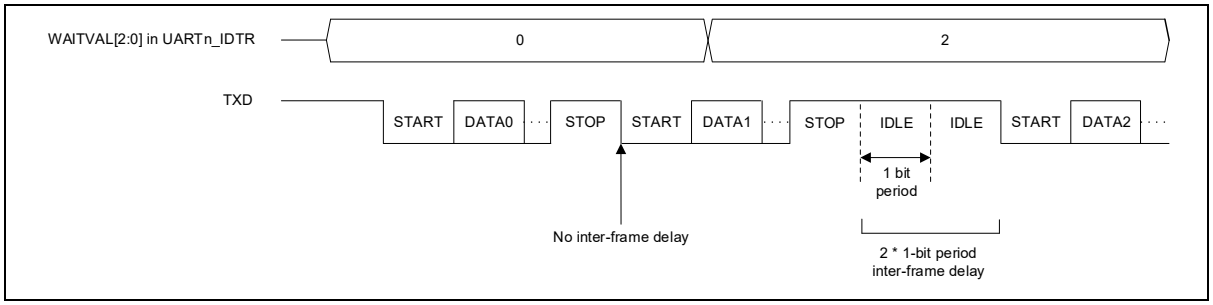
12.3.4.8 Inter-Frame Delay for Data Transmission

The inter-frame delay functionality creates an idle state between two characters on the TXD line. The length of the idle state can be determined by configuring the WAITVAL[2:0] in UARTn_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 95 shows an example format of the transmit data.

Figure 95. Example Format of Transmit Data



12.3.5 UART Receiver

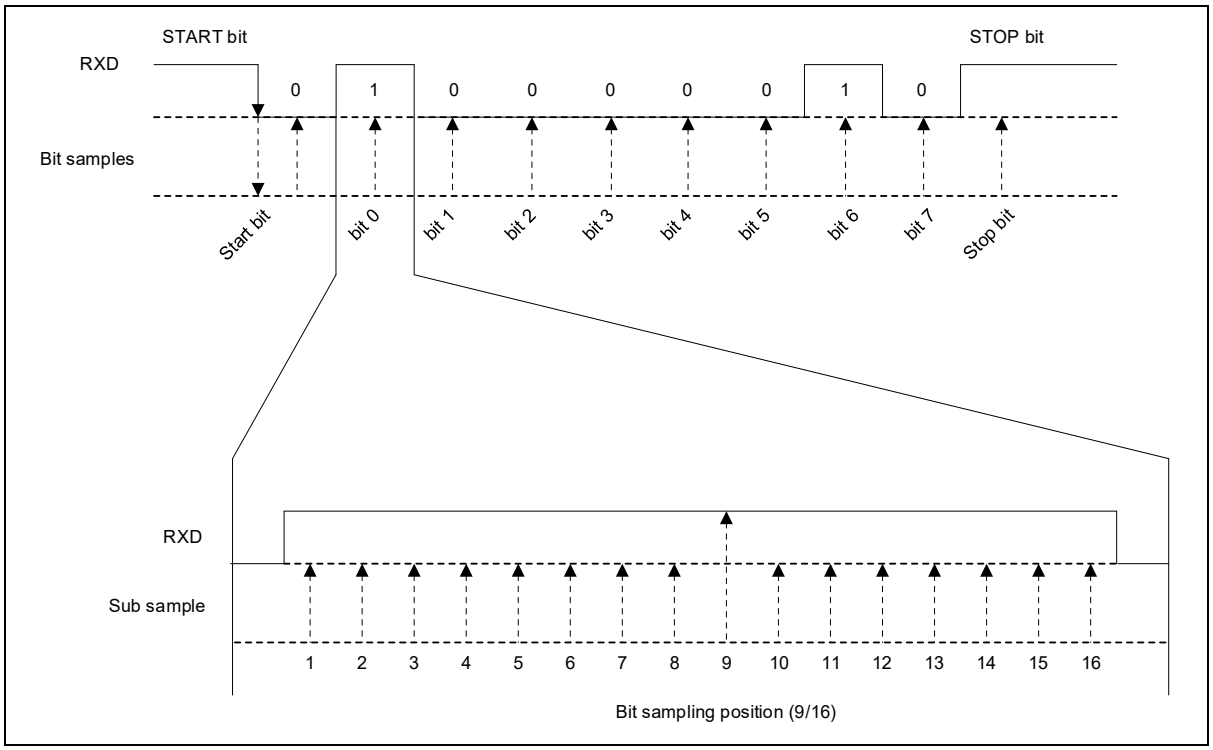
12.3.5.1 Receive Data Sampling Timing

Timing of the UART operation is described below:

Once a falling edge is detected in the receive line, the UART determines that a start bit is being received. UART oversamples the received data 16 times per bit beginning from the start bit. Among the 16 samples, the value at the 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 UARTn_LCR register.

Figure 96. Sampling Timing of UART Receiver



To enhance protection against external glitch noise, it is recommended to enable port debouncing[1] in the PCU module.

NOTE:

1. To use the port debounce function, the debounce clock in the SCU_MCCR4 and SCU_MCCR5 registers must be enabled first. If the debounce function is activated without setting the debounce clock, the port may malfunction.

12.3.5.2 Start Bit Detection

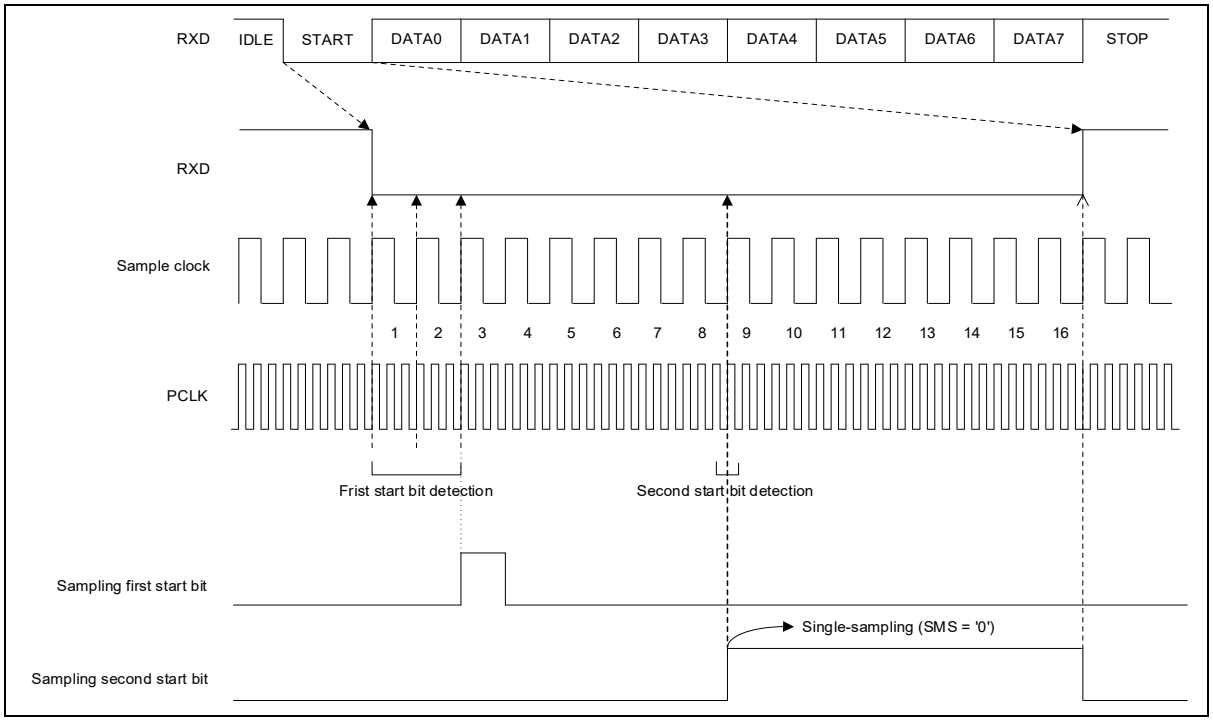
On the falling edge of the start bit, a digital filter rejects noise signal. A pulse shorter than ((3 × (1 / baud-rate)) / 16) seconds is rejected by this filter. Even when a pulse is longer than ((3 × (1 / baud-rate)) / 16) seconds, it can be rejected under the multi-sampling strategy during start-bit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

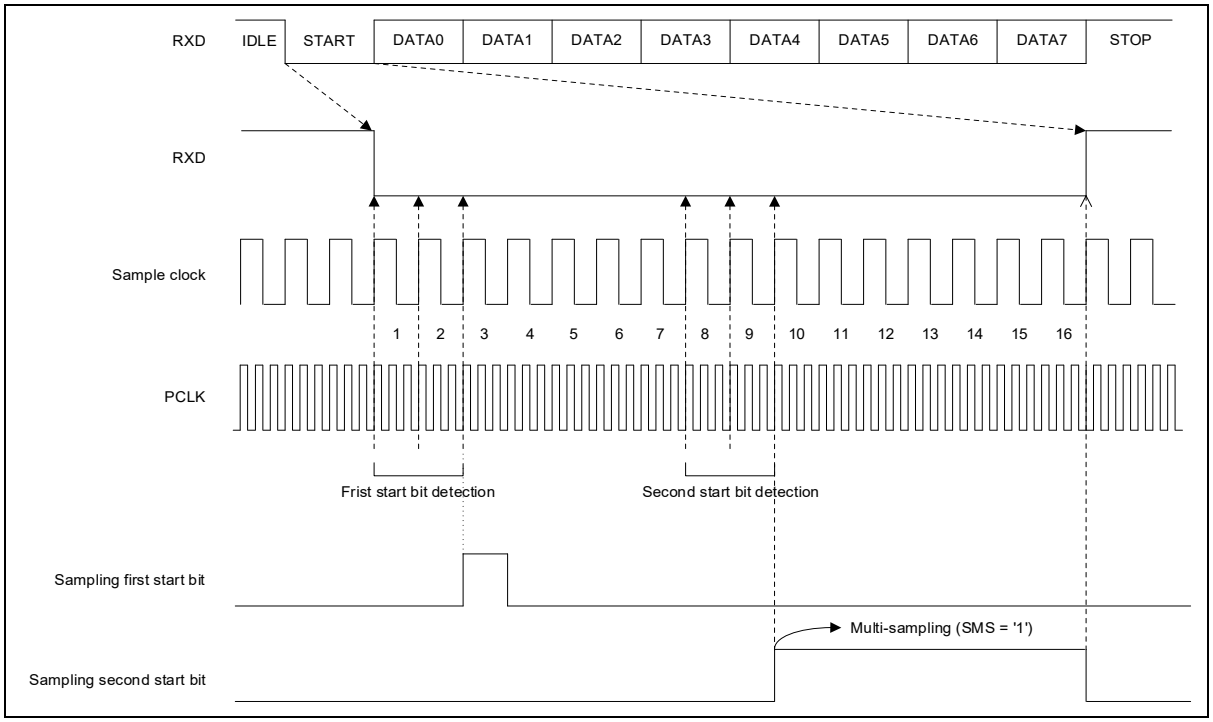
The start bit is confirmed (The DR in UARTn_LSR register set and an interrupt generated if the DRIE in UARTn_IER register is set to '1'), if the sampled bits are at '0' in single-sampling mode. The first sampling on the 1st, 2nd, and 3rd bits finds the 3 bits at '0' and the second sampling on the 9th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

Figure 97. Start Bit Detection of Single-Sampling Timing



Similarly, the start bit is confirmed (The DR in UARTn_LSR register set and an interrupt generated if the DRIE in UARTn_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 2 bits at ‘0’.

Figure 98. Start Bit Detection of Multi-Sampling Timing



NOTE:

1. If the sequence is not complete, the start bit detection stops and the receiver returns to the idle state, where it waits for a falling edge on RXD line.

12.3.5.3 Data Bit Sampling

If the DMS in UARTn_IDTR register is set to ‘1’, the values sampled at the 8th, 9th, and 10th clock pulses are determined to represent the value of the receive data bit. The multi-sampling technique is used for data recovery by discriminating between valid incoming data and noise.

Figure 99 shows the single sampling of data bits when the DMS is set to ‘0’.

Figure 99. Single-Sampling Timing of Data Bit

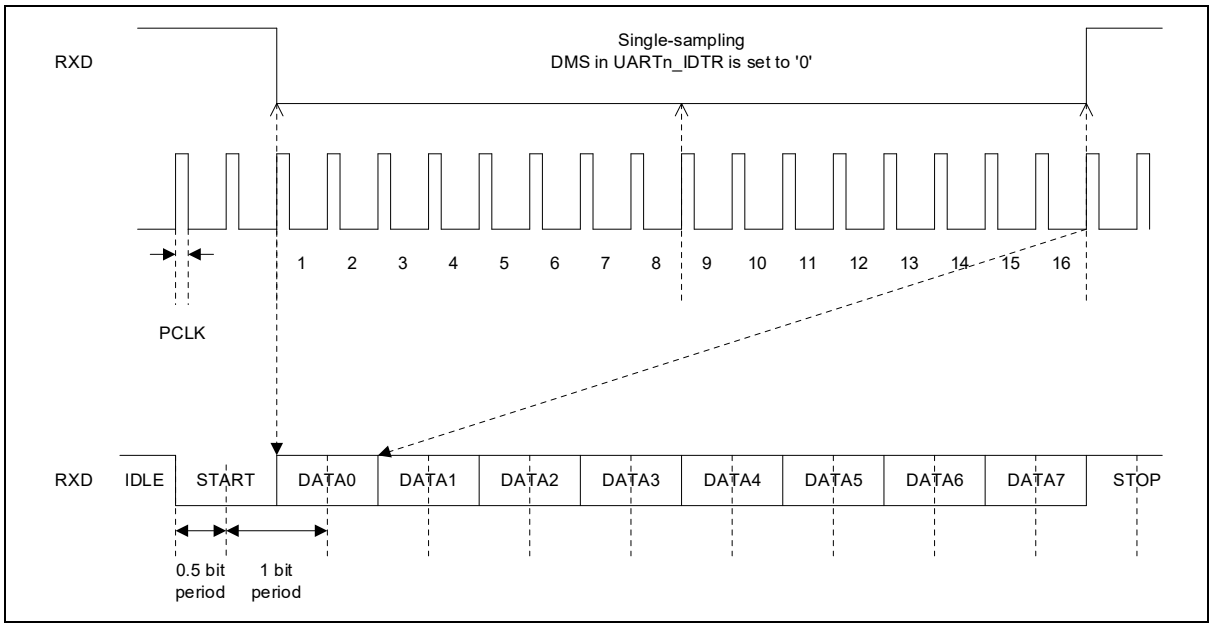
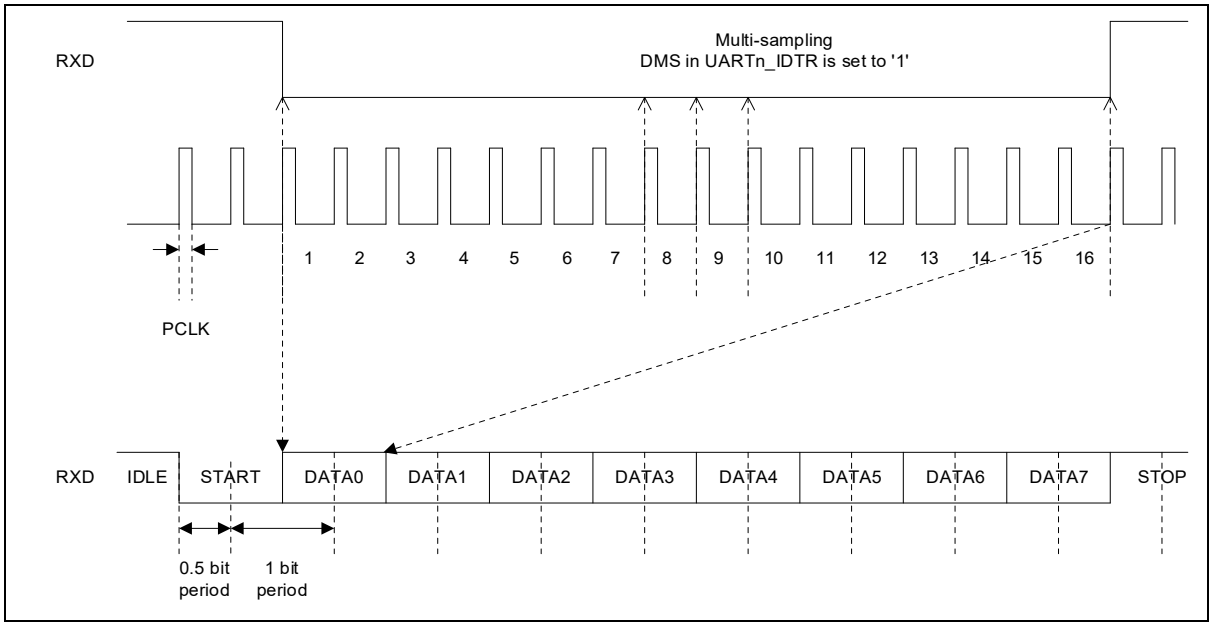


Figure 100 shows the multi-sampling of data bits when the DMS is set to ‘1’.

Figure 100. 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 UARTn_IDTR register defines the sampling strategy for the start bit, and the DMS in UARTn_IDTR register defines the sampling strategy for the data bits.

Figure 101. Data Sampling Timing

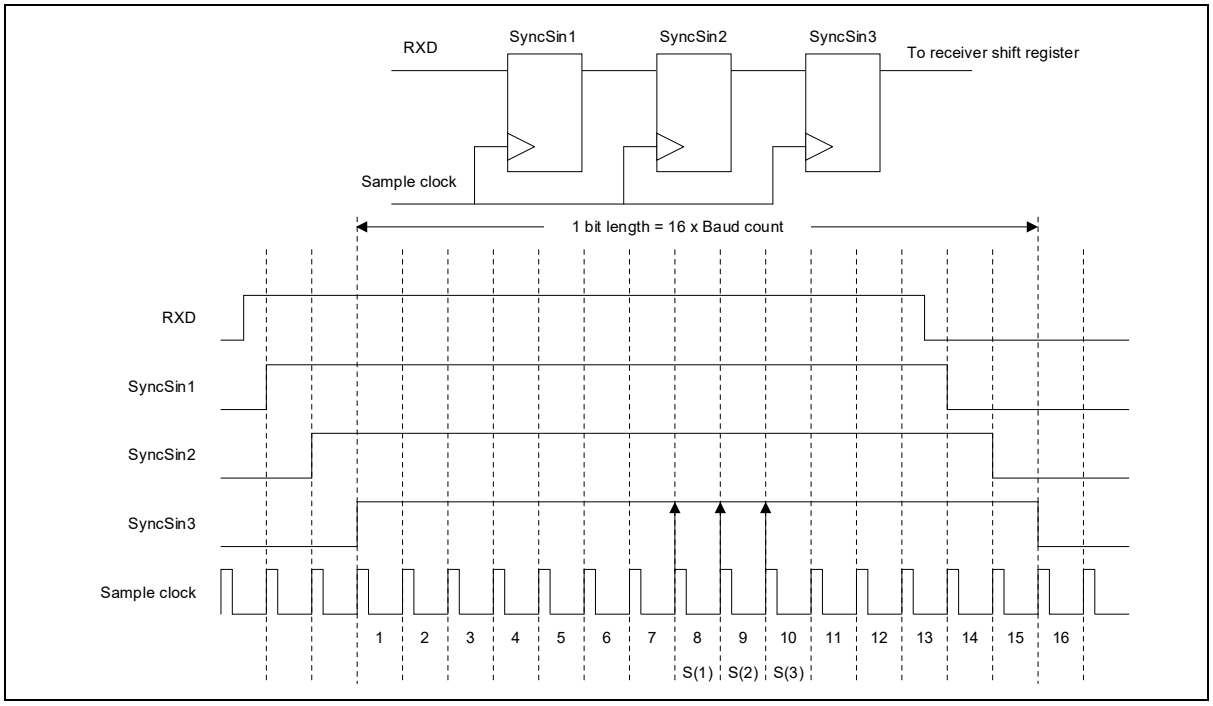


Figure 101 shows a block diagram and a sampling timing diagram of the anti-noise synchronous circuit.

If the DMS bit is set enabled, the value of each bit is sampled at multiple clock pulses. For the length of each data bit, values at S(1), S(2), and S(3) are taken out of 16 samples to determine the bit's value by taking the value indicated by a majority of the samples.

If the DMS bit is set disabled, on the other hand, the single-sampling process takes the value at S(2) only. The DMS bit does not affect the start bit, but the SMS bit works in a similar way for the Start bit.

Table 91. Received Bit Value from Sampled Data

Sampled Value	Received Bit Value	Sampled Value	Received Bit Value
000	0	100	0
001	0	101	1
010	0	110	1
011	1	111	1

12.3.5.5 Character Reception

When the falling edge is detected on the RXD Pin, the Receiver determines it as a start bit and starts the receive operation. The Data frame following the start bit is transferred one bit at a time at the LSB and stored in the Receiver Shift Register.

Receiving the stop bit means that a set of communication is completed. If the UARTn_RBR (Receive data Buffer Register) register is empty, data in the Receiver Shift Register is transferred into this register and kept until it is read out over the Bus.

[Character Reception Procedure]

DMA, Interrupt, and Polling can be used to read data from the UARTn_RBR register after confirming that the communication is completed, as described below:

- With DMA enabled, the DMA Rx Request signal is generated when data in the UARTn_RBR is updated. The DMA stores data of the UARTn_RBR into Memory as many times as set in the DMA counter and then sends the DMA Done signal to the UART. The UART outputs the DMA Rx Complete interrupt upon receiving the DMA Done signal.
- If the interrupt is generated, the IID[2:0] in UARTn_IIR register must be checked. If the IID[2:0] is set to ‘010’, this means that the reception operation is completed and the UARTn_RBR register can be read.
- To confirm the receive is completed using Polling, the DR in UARTn_LSR register must be checked. If data is received successfully, the DR bit is set to ‘1’ and data in the UARTn_RBR can be read. Reading the UARTn_RBR register clears the DR bit.

12.3.5.6 Break Condition

When a break condition is received, the UART generates break interrupt.

12.3.5.7 Overrun Error

The OE in UARTn_LSR register indicates whether the Overrun Error has been detected. The Overrun Error occurs if the Receiver completes the new Reception operation when the UARTn_RBR data was not read (The DR in UARTn_LSR register is '1'). At this moment, although the Overrun Error occurs, data in the UARTn_RBR register is updated.

Reading the UARTn_LSR register clears the OE bit.

12.3.5.8 Framing Error

When the Frame data transmission is finished, if the sampled value is read as '0' at the point of time expected to receive the stop bit, the FE in UARTn_LSR register is set to '1' (Even when multi-sampling, only the first sampled value is read). This means that the Frame data transmission is not completed on time.

The Frame data size can be determined by configuring the DLEN[1:0] in UARTn_LCR register. If the data size which is set in the DLEN[1:0] bits and the received data's bit size are different, problems may occur.

Reading the UARTn_LSR register clears the FE bit.

12.3.5.9 Parity Error

The parity bit is appended to the data frame and provides a simple way to check whether the Receiver received correct data frame. If the parity bit is incorrect, the PE in UARTn_LSR register is set to '1'.

Reading the UARTn_LSR register clears the PE bit.

12.3.5.10 Configurable Stop Bits During Reception

During Reception operations, the first stop bit is sampled regardless of the STOPBIT in UARTn_LCR register value. Users can change the STOPBIT value while the Receiver is operating.

12.3.6 UART Baud-Rate Generation

To establish communication via a UART channel, users must set an appropriate baud-rate. To this end, a programmable baud-rate generator is built in, providing a baud-rate divisor ranging from 1 to 65535.

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

Below is the formula for calculating the baud-rate:

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

If the UART clock has been set in the SCU_MCCR7(1) register, the PCLK clock speed must satisfy the formula below:

$$\text{PCLK} \geq (\text{UART clock set by SCU\_MCCR7}) \times 2$$

NOTE:

1. The UART only uses a clock source set by MCCR7. It does not use PCLK.

Table 92 lists the divisor and error rate for each baud-rate, at the UART clock speed of 16 MHz.

Table 92. Examples of Baud-Rate Calculation

UART Clock = 16 MHz		
Baud-Rate	**Divisor (UARTn_BDR)**	**Error (%)**
1,200	833	0.04%
2,400	416	0.16%
4,800	208	0.16%
9,600	104	0.16%
19,200	52	0.16%
38,400	26	0.16%
57,600	17	2.08%
115,200	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 UARTn_BFR register to the integer part indicated in the UARTn_BDR register, the correct divisor is obtained to calculate the accurate baud-rate.

NOTE:

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

For example, when the baud-rate is 9,600 bps:

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

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

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

Table 93. Examples of Calculation of Baud-Rate Using UART Fraction Counter

UART Clock = 16 MHz			
Baud-Rate	**Divisor (UARTn_BDR)**	**FCNT (UARTn_BFR)**	**Error (%)**
1,200	833	85	0.0%
2,400	416	170	0.0%
4,800	208	85	0.0%
9,600	104	42	0.0%
19,200	52	21	0.0%
38,400	26	10	0.0%
57,600	17	92	0.0%
115,200	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 94 shows an acceptable error that depends on the bit size of transmitted data.

Table 94. Tolerance of UART Receiver

Sampling	DLEN[1:0] bits	UART Frame(1)	Tolerance of UART Receive
Single-Sampling	00	| St | 5-bit data | Sp |	7 / (16×7) = 6.25%
	01	| St | 6-bit data | Sp |	7 / (16×8) = 5.46%
	10	| St | 7-bit data | Sp |	7 / (16×9) = 4.86%
	11	| St | 8-bit data | Sp |	7 / (16×10) = 4.37%
Multi-Sampling	00	| St | 5-bit data | Sp |	6 / (16×7) = 5.35%
	01	| St | 6-bit data | Sp |	6 / (16×8) = 4.68%
	10	| St | 7-bit data | Sp |	6 / (16×9) = 4.16%
	11	| St | 8-bit data | Sp |	6 / (16×10) = 3.75%

NOTE:

1. St: Start bit, Sp: Stop bit

12.3.8 UART Parity Control

Parity control (Generation of parity bit in transmission and parity checking in reception) can be enabled by setting the PEN in UARTn_LCR register. Depending on the frame length defined in the DLEN[1:0] in UARTn_LCR register, the possible UART frame formats are as listed in Table 95.

Table 95. 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 UARTn_LCR register = ‘1’ and PEN in UARTn_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 4 bits are set, then the parity bit will be ‘1’ if odd parity is selected (PARITY in UARTn_LCR register = ‘0’ and PEN in UARTn_LCR register = ‘1’).

12.3.8.3 Stick Parity

The STICKP in UARTn_LCR register determines whether to use stick parity.

- If PEN = '1', PARITY = '1', and STICKP = '1', then the parity bit is always '0'.
- If PEN = '1', PARITY = '0', and STICKP = '1', then the parity bit is always '1'.
- If STICKP = '0', stick parity is disabled.

12.3.8.4 Various Configurations of Parity Bit

A parity bit is generated based on the settings of the PEN, PARITY and STICKP in UARTn_LCR register.

Table 96 describes various configurations to create a parity bit.

Table 96. Various Configurations to Create 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 parity

12.3.8.5 Parity Checking in Reception

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

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

12.3.8.6 Parity Generation in Transmission

If the PEN in UARTn_LCR register is set to '1', the Transmitter generates a parity bit between the last data bit and the stop bit. The parity bit is set to '0' or '1' to ensure that the total number of '1's in the data and parity bits is either even or odd depending on how the PARITY and STICKP bits are set.

12.3.9 Burst Transfer Using DMA and UART

The UART module supports DMA transfers using the DMA device built in the microcontroller. The start of the memory address for data transfer and the length of the data to be transferred are determined by the programming registers in the DMA module.

Completion of the data transfer is related to a notification by the transfer complete flag. Once the entire transmit data is written to the Transmit data Hold Register (THR), the UARTn_IIR register's DMA Tx complete flag (TXE bit) is set, and the interrupt source ID of the complete interrupt is written to the IID[2:0] in UARTn_IIR register.

Similarly, once the entire receive data is written to the target DMA memory, the UARTn_IIR register's DMA Rx complete flag is set, and the interrupt source ID of the complete interrupt is written to the IID[2:0] in UARTn_IIR register. Therefore, the UART RXD signal already becomes idle when the DMA Rx complete interrupt occurs.

12.3.9.1 Transmission using DMA

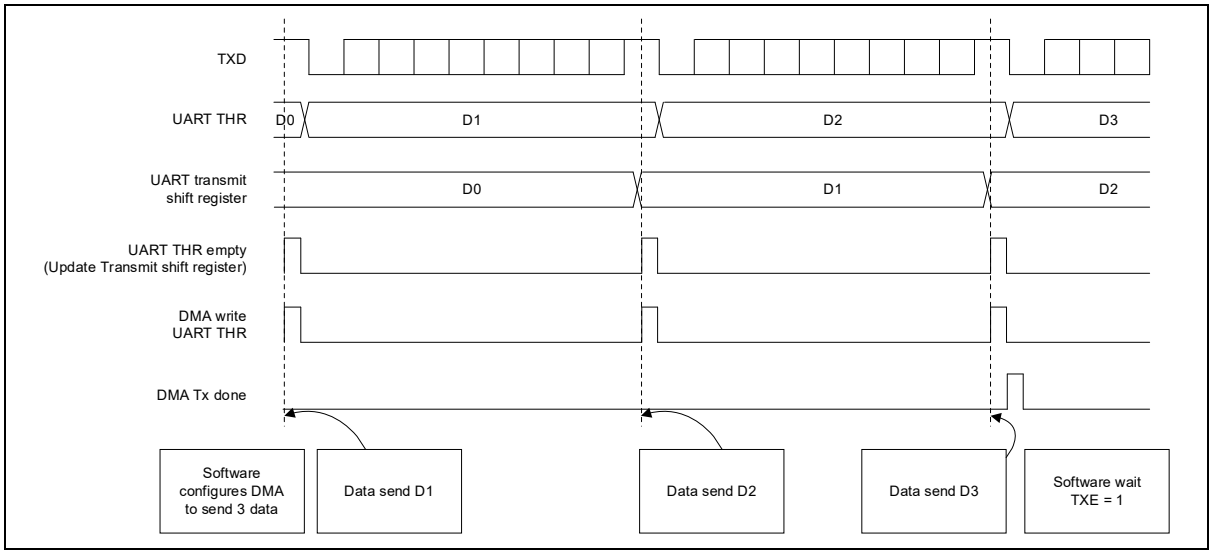
With DMA enabled, when the Transmit shift register is updated, the DMA Tx request signal is generated to allow the UARTn_THR register to store new data.

When the UARTn_THR register stores, the new data using DMA transfer and the Transmit Shift Register completes output data, the UARTn_THR register transmits the new data to the Transmit Shift Register again. The data in the Transmit Shift Register is output to the TXD Line according to the Transmitter operation.

After the DMA transmits data as many times as set in the DMA Tx Count (The set number in Figure 102 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 UARTn_THR register, the last data in the Transmit Shift Register is output and the Transmitter operation is complete. (TXE interrupt is generated.)

Figure 102. Transmission using DMA



12.3.9.2 Reception using DMA

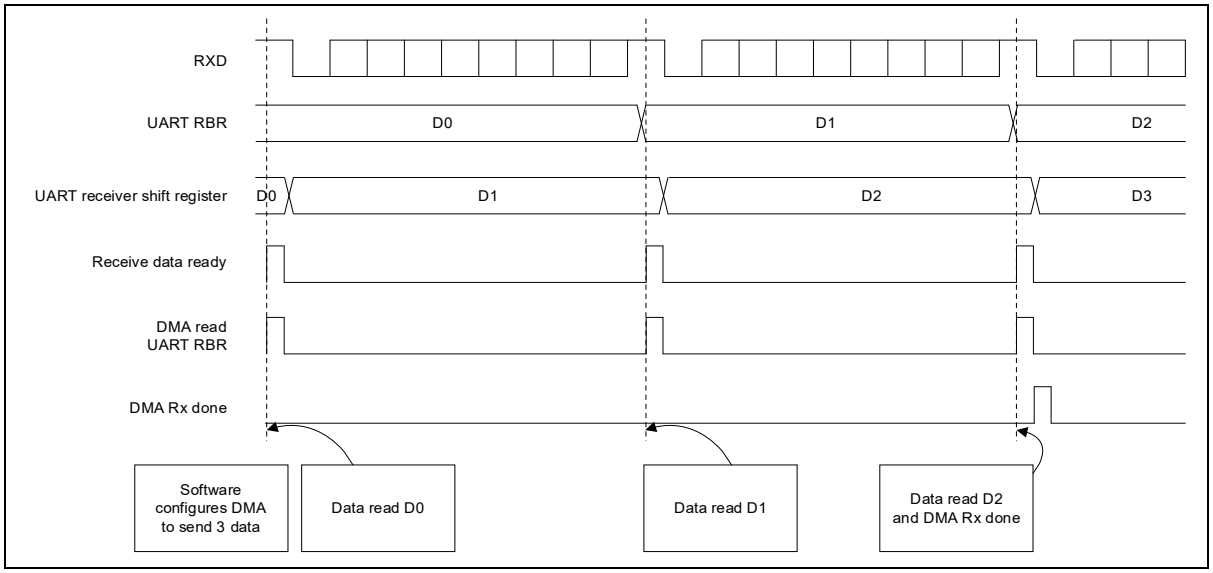
When the Receiver Shift Register is updated according to the reception operation, new data is stored in the UARTn_RBR register. With DMA enabled, when the UARTn_RBR register is updated with new data, the DMA Rx Request signal is generated.

DMA reads the UARTn_RBR register to store the data into memory.

After the DMA stores the data of the UARTn_RBR register into memory as many times as set in the DMA Rx count (The set number in Figure 103 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 103. Reception using DMA



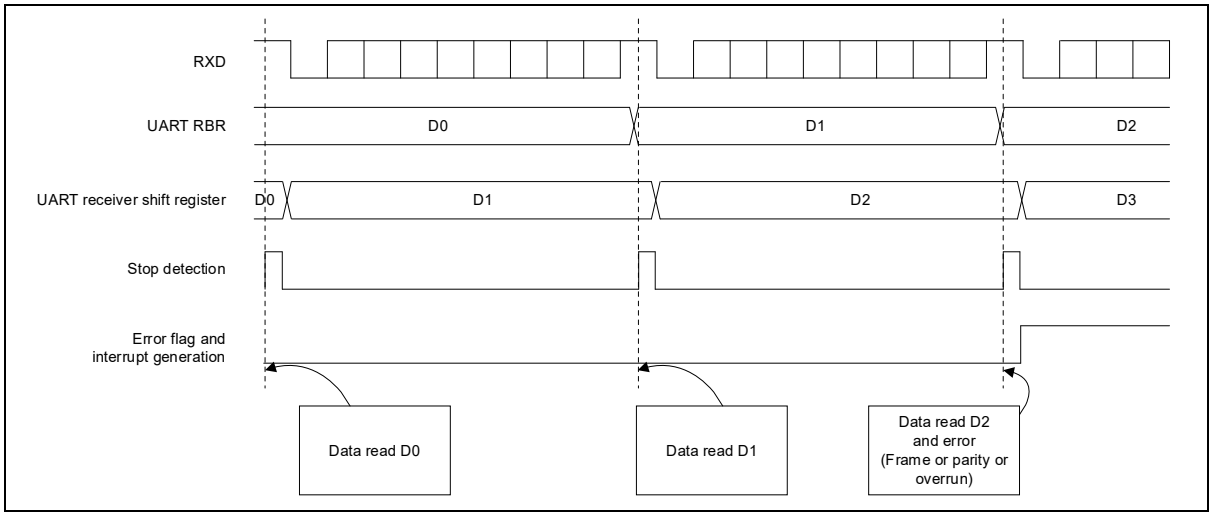
12.3.9.3 Error Flagging and Interrupt Generation in Communication

Frame errors, Parity errors, and Overrun errors may occur after the data receiving is completed, during Rx operation. If one of these errors occurs in the receive operation, the Receive Line Status Error Interrupt is generated depending on the settings of the RLSIE in UARTn_IER register.

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

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

Figure 104. Error Flagging and Interrupt Generation Timing Diagram



12.4 UART Interrupts

During UART communications, various interrupts can be generated by different events. They can be checked by reading the UARTn_IIR register.

When the microcontroller core accesses the UARTn_IIR register, the UART locks all interrupts, and the highest-priority interrupt is read.

Interrupts can occur even while the register is being accessed by the microcontroller core; however, the interrupt status remains unchanged until the current access is completed.

Among the pending interrupts, the highest-priority interrupt's source ID is represented to the IID[2:0] in UARTn_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 97. Base Address of UART

Name	Base Address
UART0	0x4000_8000
UART1	0x4000_8100

Table 98. UART Register Map

Name	Offset	Type	Description	Reset Value	Ref.
UARTn_RBR	0x0000	RO	UART n Receive Data Buffer Register	0x0000_0000	12.5.1
UARTn_THR	0x0000	WO	UART n Transmit Data Hold Register	0x0000_0000	12.5.2
UARTn_IER	0x0004	RW	UART n Interrupt Enable Register	0x0000_0000	12.5.3
UARTn_IIR	0x0008	RO	UART n Interrupt ID Register	0x0000_0001	12.5.4
UARTn_LCR	0x000C	RW	UART n Line Control Register	0x0000_0000	12.5.5
UARTn_DCR	0x0010	RW	UART n Data Control Register	0x0000_0000	12.5.6
UARTn_LSR	0x0014	RO	UART n Line Status Register	0x0000_0060	12.5.7
UARTn_ELSR	0x0014	RW	UART n Line Status Register	0x0000_0060	12.5.8
UARTn_BDR	0x0020	RW	UART n Baud rate Divisor Latch Register	0x0000_0000	12.5.9
UARTn_BFR	0x0024	RW	UART n Baud rate Fraction Counter Register	0x0000_0000	12.5.10
UARTn_IDTR	0x0030	RW	UART n Inter-frame Delay Time Register	0x0000_0000	12.5.11

NOTES:

1. A31C14x has 2 channels of UART (n = 0 to 1).
2. A31C12x has a channel of UART (n = 0).

12.5.1 UARTn_RBR: UART n Receive Data Buffer Register

UARTn_RBR is an 8-bit read-only register. Received data is read from this register, and the maximum length of data is 8 bits. The last data received is retained until a new byte is received.

UART0_RBR=0x4000_8000, UART1_RBR=0x4000_8100

31 30 29 28 27 26 25 24	23 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

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

UART0_THR=0x4000_8000, UART1_THR=0x4000_8100

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

7 0	THR[7:0]	Transmit data hold register

12.5.3 UARTn_IER: UART n Interrupt Enable Register

UARTn_IER enables six types of UART interrupts. Each interrupt generates an interrupt output signal. If users set all bits in the UARTn_IER register to 0, all UART interrupts become disabled. A particular interrupt can be set by setting the corresponding bit in this register to '1'.

UART0_IER=0x4000_8004, UART1_IER=0x4000_8104

31 30 29 28 27 26 25 24	23 22 21 20 19 18 17 16	15	14 13 12	11 10 9 8	7 6	5	4	3	2	1	0
Reserved		MODE	KEY	Reserved		DTXIEN	DRXIEN	TXEIE	RLSIE	THREIE	DRIE
-		0	000	-		0	0	0	0	0	0
-		RW	WO	-		RW	RW	RW	RW	RW	RW

15	MODE	Enable or disable IIR extended mode bit
		0 IIR legacy mode
		1 IIR extended mode
14 12	KEY	Mode change Key
		To change the mode, write 0x05 to the KEY bit and the desired value to the MODE bit simultaneously.
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 UARTn_IIR register is a read-only register informing the operation of the interrupt that is set to enabled in the UARTn_IER register. When the CPU accesses the UARTn_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.

UART0_IIR=0x4000_8008, UART1_IIR=0x4000_8108

31 30 29 28 27 26 25 24	23 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 99.)
		TXE Indicates whether the transmit shift register is empty. TXE = 0 indicates that UARTn_THR and transmit shift register is currently empty, and thus new data can be written to the UARTn_THR register.
3 1	IID[2:0]	Interrupt source ID (Refer to Table 99.)
		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 99. Interrupt ID and Control

Prio. Level	TXE	IID[2:0]			IPEN	Interrupt source		
	bit 4	bit 3	bit 2	bit 1	bit 0	Interrupt Name	Interrupt Condition	Interrupt Clear
-	0	0	0	0	1	N/A	-	-
1	0	0	1	1	0	Receive line status	Overrun, parity, framing, break error, etc.	Read UARTn_LSR
2	0	0	1	0	0	Receive data present	There is data received	Read the receive register
3	0	0	0	1	0	Transmit hold register empty	Transmit data hold register empty	Write to the transmit data hold register or UARTn_IIR
4	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 UARTn_IIR
5	0	1	1	0	0	DMA Rx complete	DMA Rx complete	Read UARTn_IIR
6	0	1	0	1	0	DMA Tx complete	DMA Tx complete	Read UARTn_IIR
7	1	1	X	X	X	Transmit register empty and DMA complete	The transmit register is empty and DMA Tx has been completed.	Read UARTn_IIR

12.5.5 UARTn_LCR: UART n Line Control Register

The UARTn_LCR register is an 8-bit register.

UART0_LCR=0x4000_800C, UART1_LCR=0x4000_810C

31 30 29 28 27 26 25 24	23 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 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 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 1.
	1	The number of stop bits is 1.5 or 2. If the data word is 5 bits long, 1.5 stop bits are added. If the data word is 6, 7, or 8 bits long, 2 stop bits are added.

1 0	DLEN[1:0]	The length of data bits included by each data transfer	
		00	5-bit data
		01	6-bit data
		10	7-bit data
		11	8-bit data

12.5.6 UARTn_DCR: UART n Data Control Register

The UARTn_DCR register is a 32-bit register that controls Tx or Rx data inversion.

UART0_DCR=0x4000_8010, UART1_DCR=0x4000_8110

31 30 29 28 27 26 25 24	23 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

[legacy mode]

The UARTn_LSR register is supported when MODE bit in UARTn_IER register is set to '0'.

The UARTn_LSR register is a read-only register that shows the status of data transmission and reception. This register reports the status of data transfers between the transmitter and receiver. Through this register, users can check the status of the UART lines and set interrupts as follows: Status interrupts for bits 1, 2, 3, and 4 can be called if they are set enabled in the RLSIE bit in the UARTn_IER register. Other bits enable their corresponding interrupts to be called if they are set enabled in the UARTn_IER register.

UART0_LSR=0x4000_8014, UART1_LSR=0x4000_8114

31 30 29 28 27 26 25 24	23 22 21 20 19 18 17 16	15 14 13 12 11	10	9	8	7	6	5	4	3	2	1	0
Reserved			DTXBUSY	DRXBUSY	Reserved	TEMT	THRE	BI	FE	PE	OE	DR	
-			0	0	-	1	1	0	0	0	0	0	
-			RO	RO	-	RO	RO	RO	RO	RO	RO	RO	

10	DTXBUSY	DMA transmit busy check. (DTXBUSY indicates whether the DMA transmit data is busy)
	0	Not busy
	1	Busy
9	DRXBUSY	DMA receive busy check (DRXBUSY indicates whether the DMA receive data is busy)
	0	Not busy
	1	Busy
6	TEMT	Indicates the transmit registers are empty TEMT indicates whether the transmit registers are empty. When both UARTn_THR and transmit shift register are empty, the bit is set to '1'. Once either UARTn_THR or transmit shift register holds any data, the bit is cleared to '0'.
	0	Data is being transmitted through the transmit registers.
	1	The transmit registers are empty.
5	THRE	Indicates the transmit data hold register is empty THRE indicates whether UARTn_THR is empty, which means the UART is ready to receive new data from the microcontroller core for transmission. UARTn_THR empty interrupts can be generated if THREIE is set enabled in UARTn_IER. The THRE bit is set to '1' when new data can be written to UARTn_THR as the previous transmit data has been transferred from UARTn_THR to the transmit shift register. The bit is cleared to '0' when new data is written to UARTn_THR by the microcontroller core (When UARTn_THR is no longer empty).
	0	The transmit data hold register is not empty.
	1	The transmit data hold register is empty.
4	BI	Indicates a break condition has been detected flag BI shows the occurrence of break interrupts. If the incoming receive data remains in low state for longer than the time taken to receive the entire data word (i.e., The sum of the start, data, parity, and stop

		bits), the bit is set to '1'. Once UARTn_LSR is read by software, the bit is cleared to '0'.
		0 Normal.
		1 A break condition has been detected.
3	FE	Indicates presence of a frame error flag 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'. The bit is cleared to '0'.
		0 No frame error present.
		1 A frame error has been detected. The data received does not have appropriate stop bits.
2	PEF	Indicates presence of a parity error flag PE indicates whether a parity error has been detected. If a received data word does not meet the odd/even parity condition defined in UARTn_LCR, the bit is set to '1'. The bit is cleared to '0'.
		0 No parity error present
		1 A parity error has been detected. The data received does not satisfy parity conditions.
1	OEF	Indicates presence of an overrun error flag OE indicates whether an overrun error has been detected. When a data word is transferred to UARTn_RBR before the register reads the previous data word, an error occurs, and the OE bit is set to '1'. 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 UARTn_RBR is already full.
0	DR	Indicates data reception status DR indicates that data reception has been completed. The bit is set to '1' when a data word has completely been received and transferred to UARTn_RBR. Once the data is read by UARTn_RBR, the bit is cleared to '0'.
		0 There is no data in the receive data buffer register.
		1 Data is received and stored in the receive data buffer register.

12.5.8 UARTn_ELSR: UART n Extended Line Status Register

[Extended Mode]

The UARTn_ELSR register is supported when MODE bit in UARTn_IER register is set to '1'.

In Extended mode, the UARTn_IIR Register cannot be used.

The UARTn_ELSR register is status register that shows the status of data transmission and reception. This register reports the status of data transfers between the transmitter and receiver. Through this register, users can check the status of the UART lines and set interrupts as follows: if they are set enabled in the RLSIE bit in the UARTn_IER register. Other bits enable their corresponding interrupts to be called if they are set enabled in the UARTn_IER register.

UART0_ELSR=0x4000_8014, UART1_ELSR=0x4000_8114

31 30 29 28 27 26 25 24	23 22 21 20 19 18 17 16	15 14 13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved			DTXIF	DRXIF	DTXBUSY	DRXBUSY	TFMTIF	THRFIF	TEMT	THRE	BIIF	FEIF	PEIF	OEIF	DR
-			0	0	0	0	0	0	1	1	0	0	0	0	0
-			RWC1	RWC1	RO	RO	RWC1	RWC1	RO	RO	RWC1	RWC1	RWC1	RWC1	RO

12	DTXIF	DMA Transmit complete interrupt flag
	0	The DMA transmit data has not completed.
	1	The DMA transmit data has been completed. (Writing a '1' to the bit clears this flag)
11	DRXIF	DMA Receiver complete interrupt flag
	0	The DMA receive data has not completed.
	1	The DMA receive data has been completed. (Writing a '1' to the bit clears this flag)
10	DTXBUSY	DMA transmit busy check. (DTXBUSY indicates whether the DMA transmit data is busy)
	0	Not busy
	1	Busy
9	DRXBUSY	DMA receive busy check (DRXBUSY indicates whether the DMA receive data is busy)
	0	Not busy
	1	Busy
8	TEMTIF	Transmit complete interrupt flag TEMTIF indicates whether the transmit shift register is empty. TEMT = 0 indicates that UARTn_THR and transmit shift register is currently empty and thus new data can be written to the UARTn_THR register. Write '1' to clear
7	THREIF	Transmit data hold register empty flag Write to the transmit data hold register. Write '1' to clear
6	TEMT	Indicates the transmit registers are empty TEMT indicates whether the transmit registers are empty. When both UARTn_THR and transmit shift register are empty, the bit is set to '1'. Once either UARTn_THR or transmit shift register holds any data, the bit is cleared to '0'.
	0	Data is being transmitted through the transmit registers.

		1 The transmit registers are empty.
5	THRE	Indicates the transmit data hold register is empty THRE indicates whether UARTn_THR is empty, which means the UART is ready to receive new data from the microcontroller core for transmission. UARTn_THR empty interrupts can be generated if THREIE is set enabled in UARTn_IER. The THRE bit is set to '1' when new data can be written to UARTn_THR as the previous transmit data has been transferred from UARTn_THR to the transmit shift register. The bit is cleared to '0' when new data is written to UARTn_THR by the microcontroller core (When UARTn_THR is no longer empty).
		0 The transmit data hold register is not empty.
		1 The transmit data hold register is empty.
4	BIIF	Indicates a break condition has been detected flag 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 UARTn_LSR is read by the microcontroller core, the bit is cleared to '0'.
		0 Normal.
		1 A break condition has been detected.
3	FEIF	Indicates presence of a frame error flag 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. The bit is cleared to '0'.
		0 No frame error present.
		1 A frame error has been detected. The data received does not have appropriate stop bits.
2	PEIF	Indicates presence of a parity error flag PE indicates whether a parity error has been detected. If a received data word does not meet the odd/even parity condition defined in UARTn_LCR, the bit is set to '1'. The bit is cleared to '0'.
		0 No parity error present
		1 A parity error has been detected. The data received does not satisfy parity conditions.
1	OEIF	Indicates presence of an overrun error flag OE indicates whether an overrun error has been detected. When a data word is transferred to UARTn_RBR before the register reads the previous data word, an error occurs, and the OE bit is set to '1'. 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 UARTn_RBR is already full.
0	DR	Indicates data reception status DR indicates that data reception has been completed. The bit is set to '1' when a data word has completely been received and transferred to UARTn_RBR. Once the data is read by UARTn_RBR, the bit is cleared to '0'.
		0 There is no data in the receive data buffer register.
		1 Data is received and stored in the receive data buffer register.

12.5.9 UARTn_BDR: UART n Baud-Rate Divisor Latch Register

The UARTn_BDR register is a 16-bit register.

UART0_BDR=0x4000_8020, UART1_BDR=0x4000_8120

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

15 0	BDR[15:0][1]	Baud-rate divisor latch (1 to 65535)

NOTE:

1. If the UARTn_BDR value is equal to or less than 0x0002, the UARTn_BFR must be 0x00 regardless of the UARTn_BFR set value.

12.5.10 UARTn_BFR: UART n Baud-Rate Fraction Counter Register

The UARTn_BFR register is an 8-bit register.

UART0_BFR=0x4000_8024, UART1_BFR=0x4000_8124

31 30 29 28 27 26 25 24	23 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 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 UARTn_BDR value is equal to or less than 0x0002, the UARTn_BFR must be 0x00 regardless of the UARTn_BFR set value.

12.5.11 UARTn_IDTR: UART n Inter-Frame Delay Time Register

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

UART0_IDTR=0x4000_8030, UART1_IDTR=0x4000_8130

31 30 29 28 27 26 25 24	23 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 the 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 the microcontroller's power-on, if users disable the SCU_PER2 register's UART bit and then enable it again, the SMS bit is set to '1'.

12.5.12 UART Register Map Summary

Table 100. UART Register Map Summary

Offset	Register name	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0x00	UARTn_RBR	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	RBR[7:0]							
	Reset value																									0	0	0	0	0	0	0	0
0x00	UARTn_THR	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	THR[7:0]							
	Reset value																									0	0	0	0	0	0	0	0
0x04	UARTn_IER	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	MODE	Res.	Res.	KEY	Res.	Res.	Res.	Res.	Res.	Res.	DTXIEN	DRXIEN	TXEIE	RLSIE	THREIE	DRIE
	Reset value																	0			0							0	0	0	0	0	0
0x08	UARTn_IIR	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	TXE	IID[2:0]			IPEN
	Reset value																												0	0	0	0	1
0x0C	UARTn_LCR	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.	BREAK	STICKP	PARITY	PEN	STOPBIT	DLEN[1:0]	
	Reset value																										0	0	0	0	0	0	0
0x10	UARTn_DCR	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	RXINV	TXINV	Res.	Res.
	Reset value																													0	0		
0x14	UARTn_LSR [Legacy mode]	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.		DTXBUSY	DRXBUSY	Res.	Res.	TEMT	THRE	BI	FE	PE	OE	DR
	Reset value																						0	0			1	1	0	0	0	0	0
0x14	UARTn_ELSR [Extended mode]	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	DTXIF	DRXIF	DTXBUSY	DRXBUSY	TFMTIF	THRFIF	TEMT	THRE	BIIF	FEIF	PEIF	OEIF	DR
	Reset value																				0	0	0	0	0	0	1	1	0	0	0	0	0
0x20	UARTn_BDR	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	BDR[15:0]															
	Reset value																	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
0x24	UARTn_BFR	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	BFR[7:0]							
	Reset value																									0	0	0	0	0	0	0	0
0x30	UARTn_IDTR	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	SMS	DMS	Res.	Res.	Res.	WAITVAL[2:0]		
	Reset value																									0	0				0	0	0

13. I2C

13.1 I2C Introduction

The I2C (Inter-integrated Circuit) interface built in A31C14x, and A31C12x series satisfies the standard I2C communication protocol and is used for serial communication with internal and external devices via the I2C protocol.

The I2C block, which is equipped with one unit, supports both master and slave modes, and can transmit and receiving data in bytes by using interrupts or polling.

The I2C of the A31C14x and A31C12x series operates in Standard mode (100 kHz) or Fast mode (400 kHz) and supports General call.

It helps communicate with various peripherals that have the same bus type. To use the I2C, it is recommended to set the SCL and SDA pins to open-drain and then connect external pull-up resistors to render their output signals ‘HIGH’.

13.2 I2C Main Features

The I2C of the A31C14x and A31C12x series features the following:

- Compliant with I2C protocol
- Master and slave modes
- General call
- Transfer rates configurable
  - Maximum transfer rate: 400 kHz
- I2C interrupts
- 7-bit addressing
- Delay time can be set for pin SCL’s high- or low-period
- Hold time can be set for previous data
- Generates and detects STOP, START, and ACK signals
- Noise canceller

13.3 I2C Implementation

I2C module implemented in the A31C14x and A31C12x series has all features of an I2C peripheral as shown in Table 101.

Table 101. Features of I2C

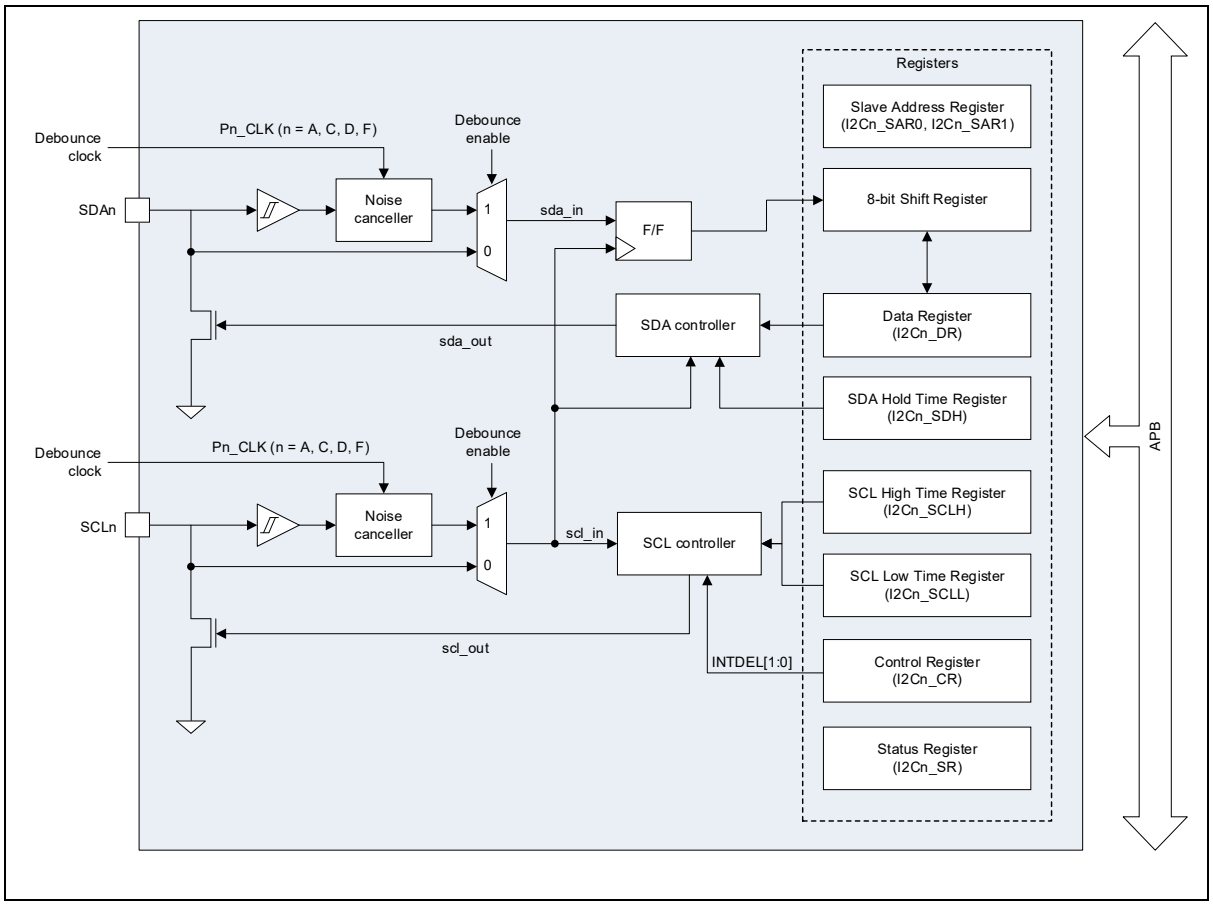
I2C Features	I2C
7-bit addressing mode	O
Standard-mode (up to 100 kbit/s)	O
Fast-mode (up to 400 kbit/s)	O
General call	O

13.4 I2C Functional Description

13.4.1 I2C Block Diagram

Figure 105 shows a block diagram of the I2C module.

Figure 105. I2C Block Diagram



NOTES:

1. When the corresponding port of the I2C is a sub-function for SCL and SDA pin, the SCL and SDA pins are automatically set to the N-channel 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 canceller, refer to section 4.8.29 or section 4.8.30.

13.4.2 I2C Pins and Signals

Table 102 describes pins assigned for the I2C interface.

Table 102. Pin Assignment of I2C: External Pins

Pin Name	Type	Description	Package Type							
			64-LQFP	48-LQFP, QFN	44-LQFP	32-QFN, LQFP	28-TSSOP	24-QFN	20-QFN	20-TSSOP
SCL0	I/O	I2C channel 0 serial clock bus line (open-drain)	O	O	O	O	O	O	O	O
SDA0	I/O	I2C channel 0 serial data bus line (open-drain)	O	O	O	O	O	O	O	O
SCL1	I/O	I2C channel 1 serial clock bus line (open-drain)	O	O	O	O	O	O	O	O
SDA1	I/O	I2C channel 1 serial data bus line (open-drain)	O	O	O	O	O	O	O	O
SCL2	I/O	I2C channel 2 serial clock bus line (open-drain)	O	-	-	-	-	-	-	-
SDA2	I/O	I2C channel 2 serial data bus line (open-drain)	O	-	-	-	-	-	-	-

13.4.3 I2C Mode Selection

The I2C module operates in one of the following modes:

- Slave transmitter
- Slave receiver
- Master transmitter
- Master receiver

The I2C module supports the four operation modes above and operates as a slave by default. It can operate as a master by configuring the START bit in the 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.

13.4.4 I2C Initialization

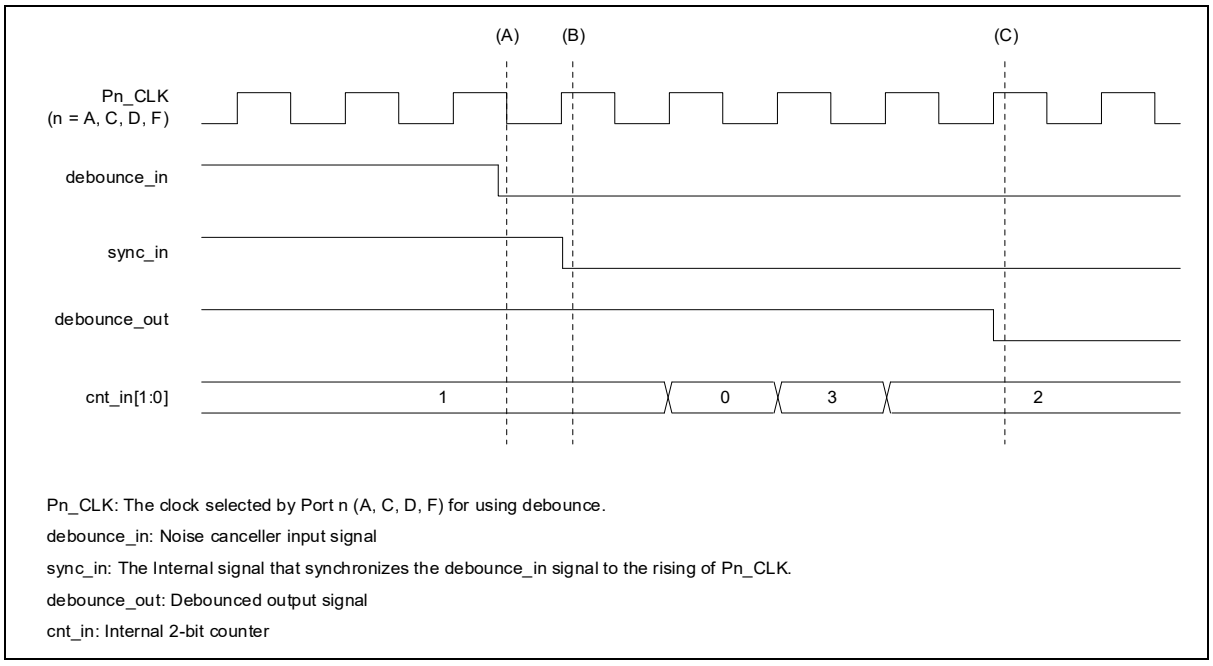
13.4.4.1 Enabling and Disabling Peripherals

To use the I2C, follow the procedure below:

1. Set the I2C in SCU_PER2 register to ‘1’. This enables the I2C module.
2. Set the I2C in SCU_PCER2 register to ‘1’. This enables the I2C module clock.
3. Configure the I2Cn_CR register to control the I2C communication.

13.4.4.2 Noise Canceller

Figure 106. Noise Canceller Timing



Pn_CLK: The clock selected by Port n (A, C, D, 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 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, C, D, F)

13.4.4.3 Using Noise Canceller in I2C Module

The following describes how to use noise-canceller.

1. I2C module uses PA, PC, PD, PF. So, select PGADCSEL (PA) in SCU_MCCR4 register or PGBDCSEL (PC, PD), PGCDCSEL (PF) in SCU_MCCR5 register. According to the port to be used and set the clock source for debouncing.
2. Set the PGBDCDIV (PC, PD), PGCDCDIV (PF) in SCU_MCCR5 register or PGADCDIV (PA) in SCU_MCCR4 register to divide the clock source for debouncing.
3. Enable debouncing by setting the PDEx (x = 0 to 15) in Px_DER (x = A, C, D, 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.

13.4.5 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.

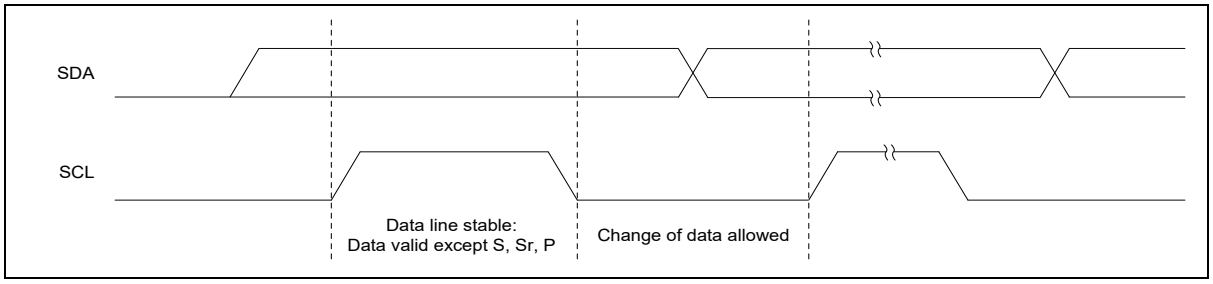
13.4.6 I2C Module Protocol

13.4.6.1 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 107 shows a timing diagram of the I2C Bus Bit Transfer.

Figure 107. I2C Bus Bit Transfer Timing Diagram



13.4.6.2 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 108):

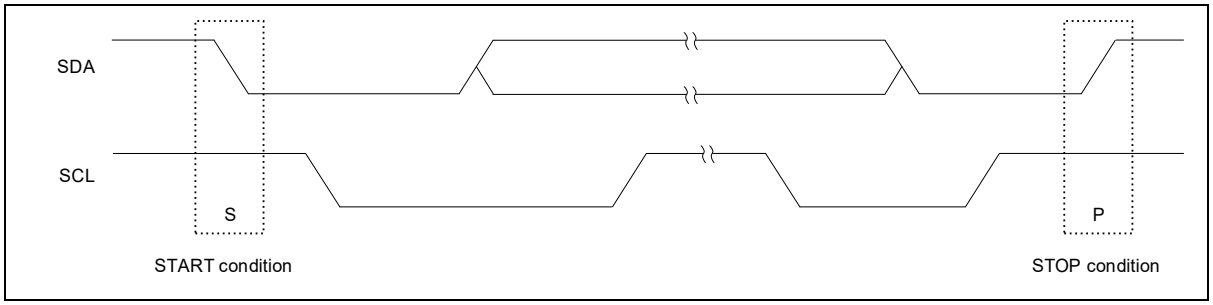
- START (S) condition: ‘H’ to ‘L’ transition on the SDA line while SCL is ‘H’
- STOP (P) condition: ‘L’ to ‘H’ transition on the SDA line while SCL is ‘H’

These START and STOP conditions are always generated by the master. The bus is considered busy once the START condition occurs and is considered as free again after the STOP condition occurs.

Thus, the bus stays busy between a START and a STOP. If a repeated START (Sr) is generated instead of a STOP, the bus remains busy. In this respect, the START (S) and repeated START (Sr) conditions are functionally identical. For the remainder of this document, therefore, an S will be used to represent both the START and repeated START conditions, unless Sr is particularly relevant.

The detection of the START and STOP conditions by devices connected to the bus is easy if the necessary interfacing hardware is incorporated. However, microcontrollers with no such interface have to sample the SDA line at least twice per clock period to sense the transition.

Figure 108. START and STOP Conditions



13.4.6.3 Acknowledge

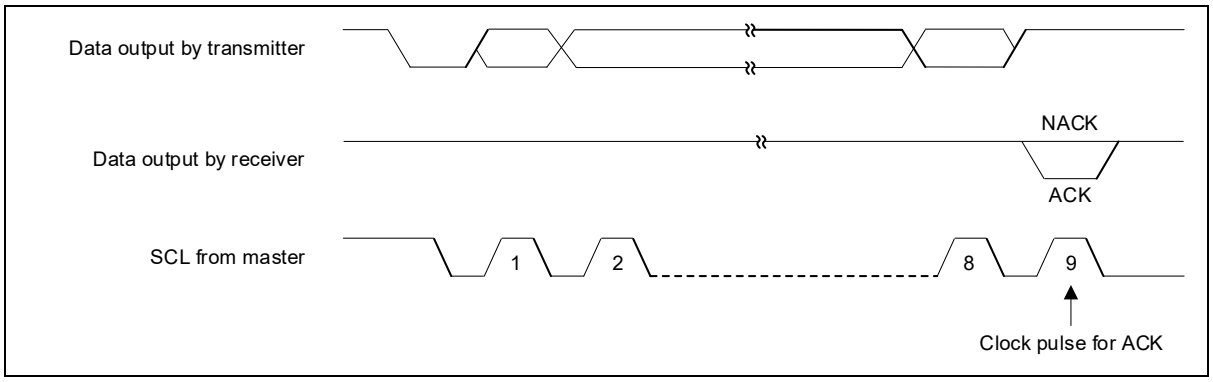
Data transfer with acknowledgement is necessary. The acknowledge-related clock pulse is generated by the master. The transmitter releases the SDA line ('H') during the Acknowledge clock pulse. The receiver must pull down the SDA line during the Acknowledge clock pulse so that it remains at stable 'L' during the 'H' period of this clock pulse. (See Figure 109)

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 generate the STOP or repeated START condition.

Figure 109. I2C Bus Response

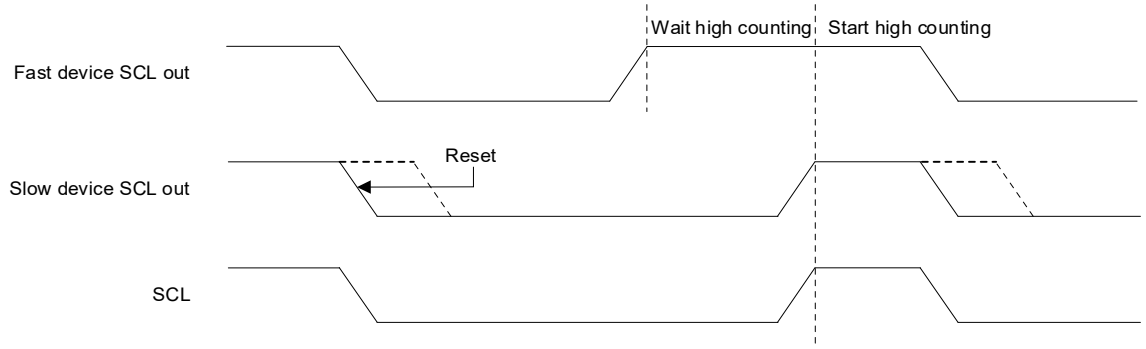


13.4.6.4 Synchronization

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 110. Clock Synchronization During Arbitration



NOTE:

1. The ‘L’ to ‘H’ transition of this clock may not change the state of the SCL line if another clock is still within its ‘L’ by the device with the longest ‘L’ period. Devices with shorter ‘L’ periods enter an ‘H’ wait-state during this time.

When all devices concerned have counted off their ‘L’ period, the clock line will be released and go ‘H’. Then, there will be no difference between the device clocks and the state of the SCL line, and the devices will start counting their ‘H’ periods. The first device to complete its ‘H’ period will again pull the SCL line ‘L’.

13.4.6.5 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.

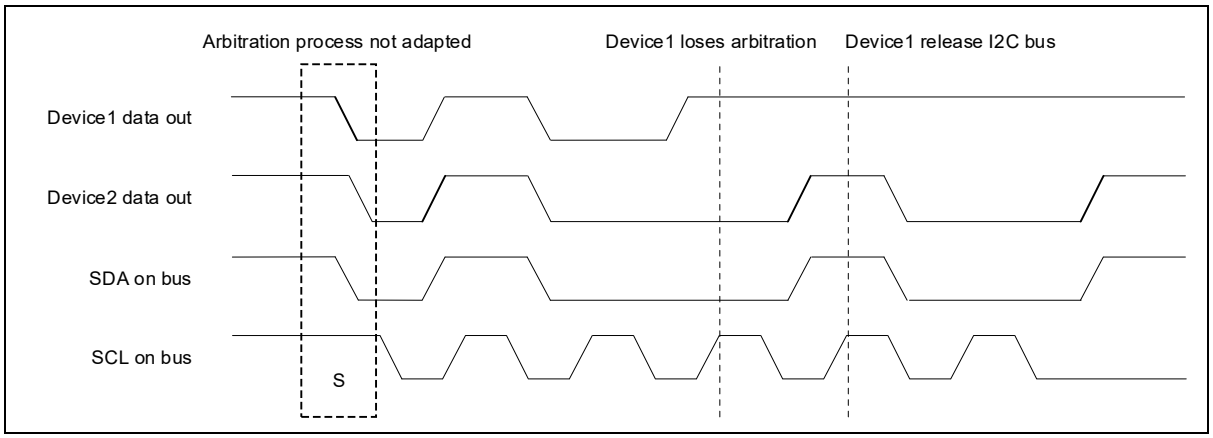
Arbitration can continue for many bits. Its first stage is the comparison of the address bits. If the masters are trying to address the same device, arbitration proceeds with comparing either the data-bits (If they are master-transmitters) or the acknowledge-bits (If they are master-receivers).

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 111 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 111. Arbitration Process between Two Masters



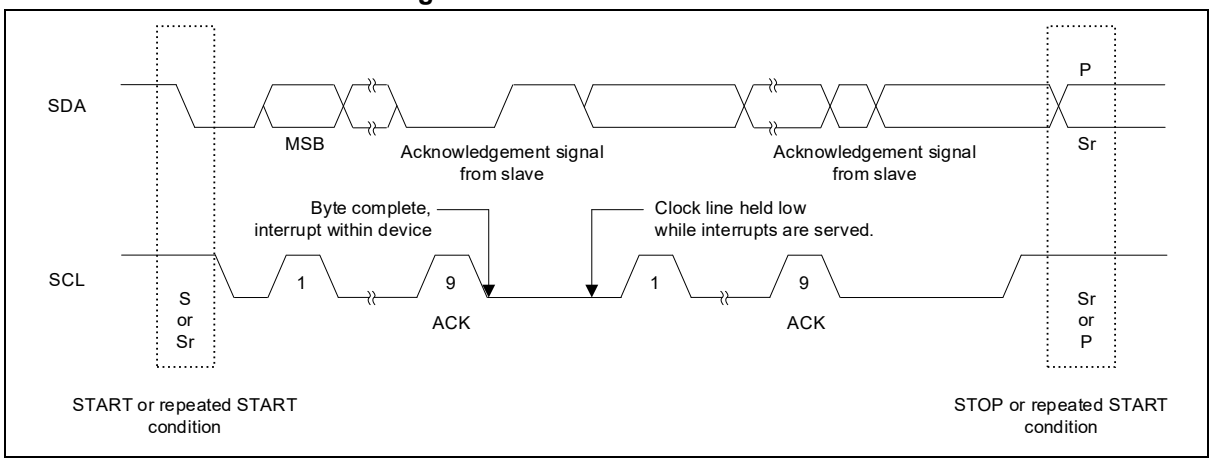
13.4.6.6 Data Transfer

Every byte put on the SDA line must be 8-bits long. The number of bytes that can be transmitted per transfer is unrestricted. Each byte must be followed by an Acknowledge bit and is transferred MSB first. (Refer to Figure 112)

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, release clock line SCL and master continues data transfer.

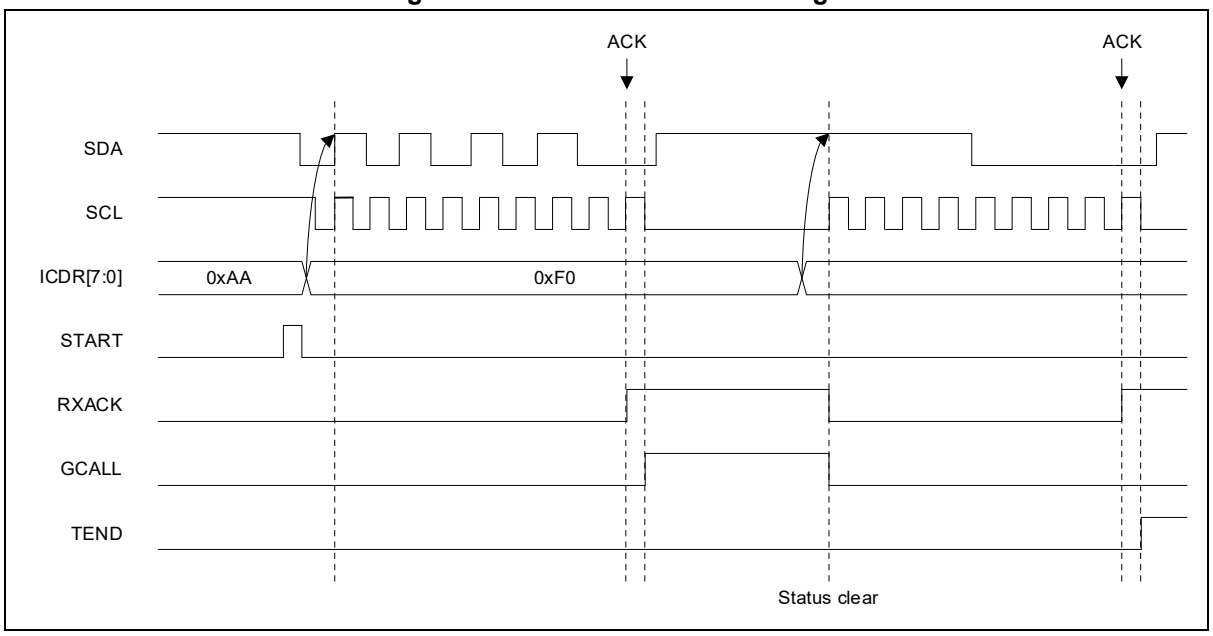
A message starting with such an address can be terminated by an occurrence of the STOP condition, even during the transmission of a byte. In this case, no acknowledgement is generated.

Figure 112. I2C Bus Data Transfer



13.4.6.7 Master Transmit Timing

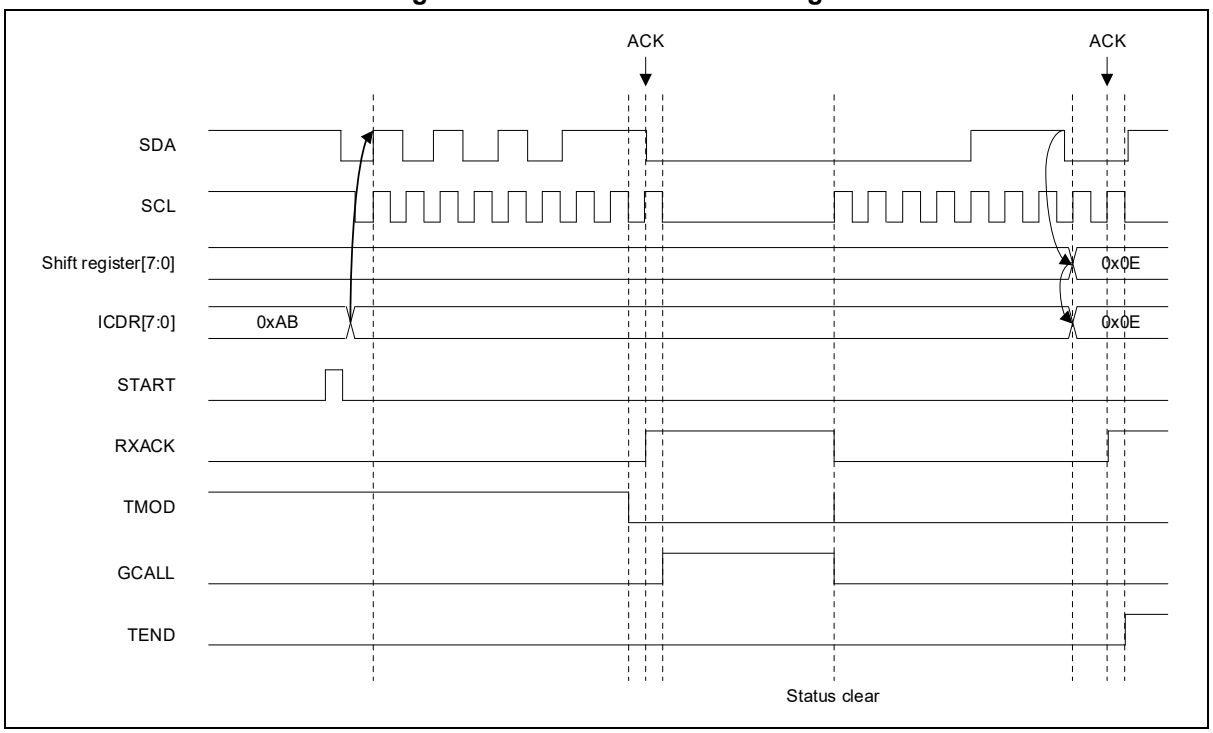
Figure 113. Master Transmit Timing



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 8 SCL 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’.

13.4.6.8 Master Receive Timing

Figure 114. Master Receive Timing

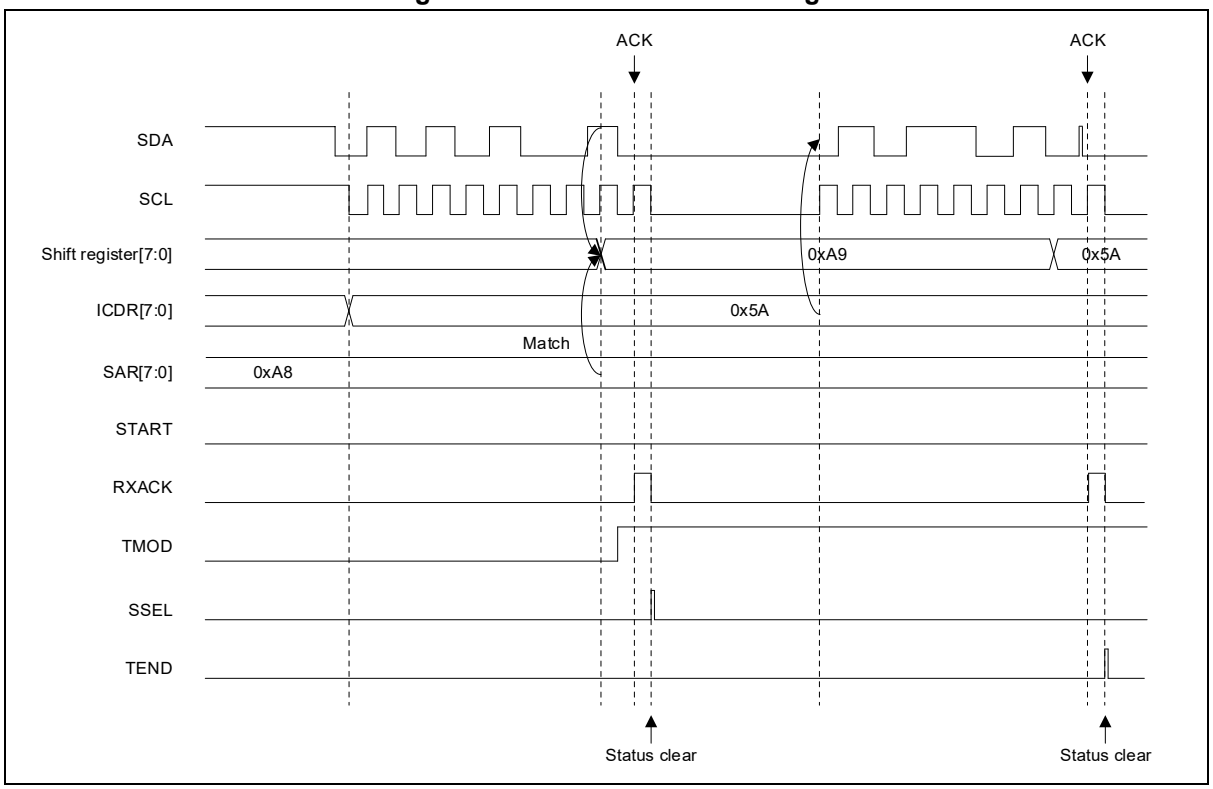


1. When bit-0 stored in I2Cn_DR register is ‘1’, if START in I2Cn_CR register is written as ‘1’, Master operates in Receive mode.
2. 8-bit data stored in I2Cn_DR register is transmitted during 8 SCL pulses, and TMODE in I2Cn_SR register becomes ‘0’ (Receive mode).
3. If ACK is normally received at the 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 I2Cn_SR register becomes ‘1’. When data reception is completed, the TEND in I2Cn_SR register becomes ‘1’.

13.4.6.9 Slave Transmit Timing

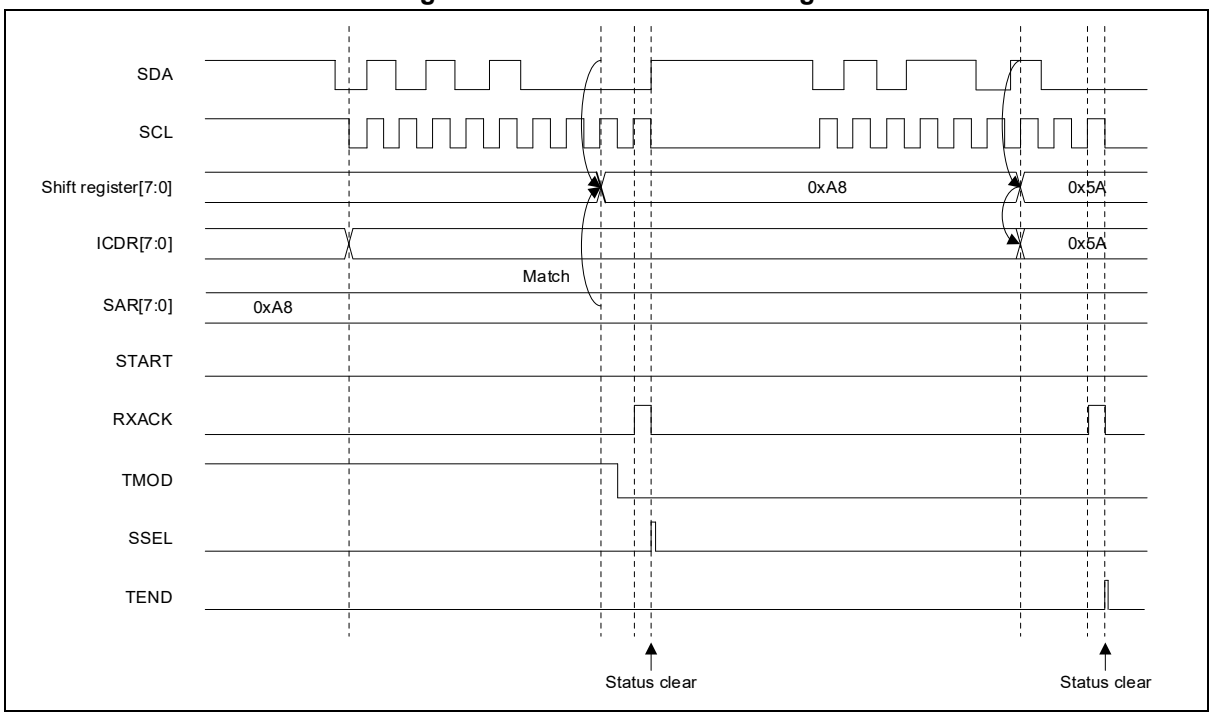
Figure 115. Slave Transmit Timing



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 I2C_SR register is '1'.
4. When Address Match is reached, the SSEL in I2C_SR register changes to ‘1’ and it is selected as Slave, and the data stored in the I2C_DR register is transmitted according to the SCL pulse.
5. When all 8-bit data is transmitted, TEND in I2C_SR register becomes ‘1’ at the falling edge of the 9th pulse of SCL.

13.4.6.10 Slave Receive Timing

Figure 116. 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 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 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.

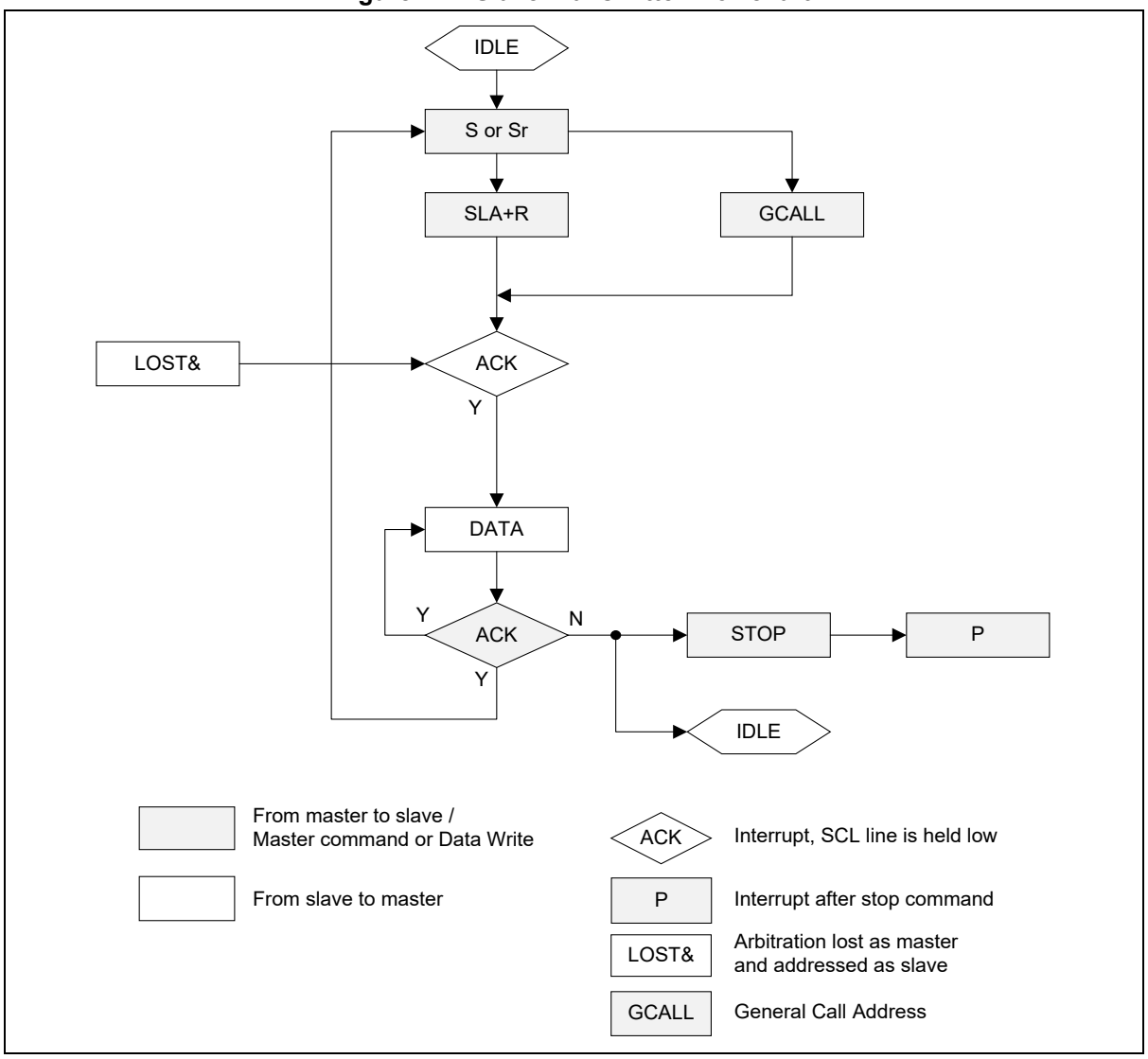
At the 9th pulse of SCL, the RXACK in I2Cn_SR register becomes ‘1’. When data reception is completed, the TEND in I2Cn_SR register becomes ‘1’.

13.4.7 I2C Slave Mode

13.4.7.1 Slave Transmitter

Figure 117 shows a flowchart of the transmitter in slave mode.

Figure 117. Slave Transmitter Flowchart



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

1. If the main operating clock (PCLK) is slower than the SCL, users must set the I2Cn_SDH register to 0x00, so that the SDA is changed at the falling edge of the SCL. The SDA hold time is obtained by multiplying the I2Cn_SDH register value by the PCLK period. If the SDA hold time is longer than the SCL period, data cannot be properly transmitted.
2. Set the I2C in SCU_PER2 and SCU_PCER2 registers to enable and clock the I2C.

3. 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_SAR register. In the case that the GCEN in I2C_SAR register is set enabled, the I2C raises the “general call” flag if the received data is 0x00.

4. If the received address does not match the SVAD[6:0] in I2Cn_SAR0 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.

5. One byte of data is transmitted.

6. 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.

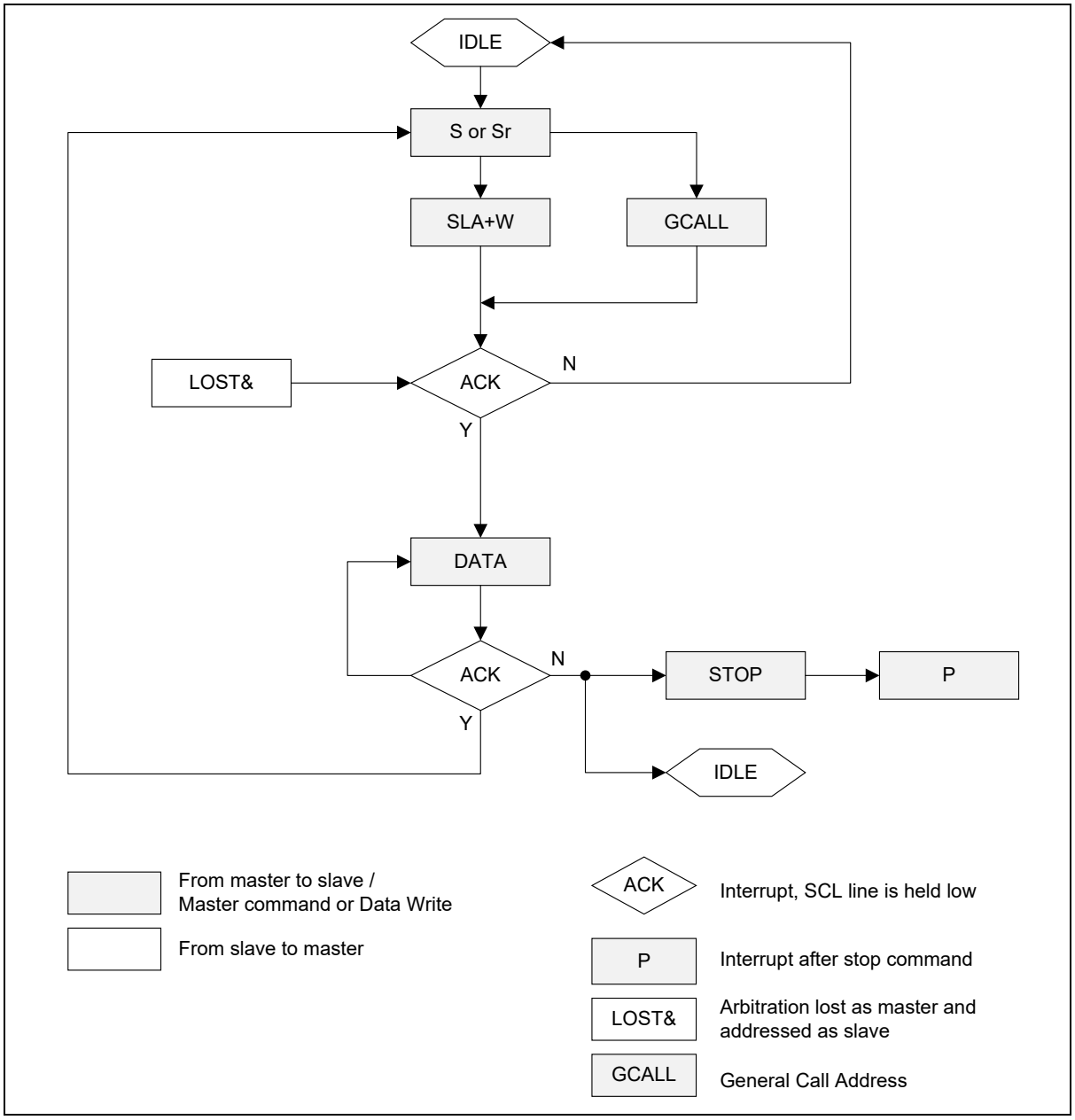
   Once one of the above operations is performed, release the SCL line by writing a value 0xFF to the I2Cn_SR register. For Case 1, go to step 8 and end the communication. For Case 2, go to step 6. In either case, the consecutive START condition can be detected; then, go to step 4.

7. 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.

13.4.7.2 Slave Receiver

Figure 118 shows a flowchart of the receiver in slave mode.

Figure 118. Slave Receiver Flowchart



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

1. If the main operating clock (PCLK) is slower than the SCL, users must set the I2Cn_SDH register to 0x00, so that the SDA is changed at the falling edge of the SCL. The SDA hold time is obtained by multiplying the I2Cn_SDH register value by the PCLK period. If the SDA hold time is longer than the SCL period, data cannot be properly transmitted.

2. Set the I2C in SCU_PER2 and SCU_PCER2 registers to enable and clock the I2C.
3. When the START condition is detected, the I2C receives one byte of data and compares it with the SVAD[6:0] in I2Cn_SAR register. In the case that the GCEN in I2Cn_SAR register is set enabled, the I2C raises the “general call” flag if the received data is 0x00.
4. If the received address does not match the SVAD[6:0] in I2Cn_SAR register, the I2C becomes idle and waits until another START bit is detected. If the ACKEN bit is set enabled and the received address matches the address in the SVAD[6:0] bit field, the I2C triggers the SSEL interrupt and maintains the SCL line in the ‘L’ level. If the ACKEN bit is set disabled, the I2C becomes idle even though the received address matches the SVAD[6:0] bit field. If the SSEL interrupt occurs, the I2C is ready to receive data. Release the SCL line by writing a value 0xFF to the I2Cn_SR register.
5. One byte of data is received.
6. Once the reception is completed, the I2C triggers the TEND interrupt regardless of an ACK signal from the slave, and it maintains the SCL line in the ‘L’ level. The slave performs one of the following operations:

   Case 1: If an ACKEN = ‘0’, a NACK signal is generated and the I2C waits for the STOP condition or consecutive START condition.

   Case 2: If an ACKEN = ‘1’, an ACK signal is generated and the I2C can continue receiving data from the master.

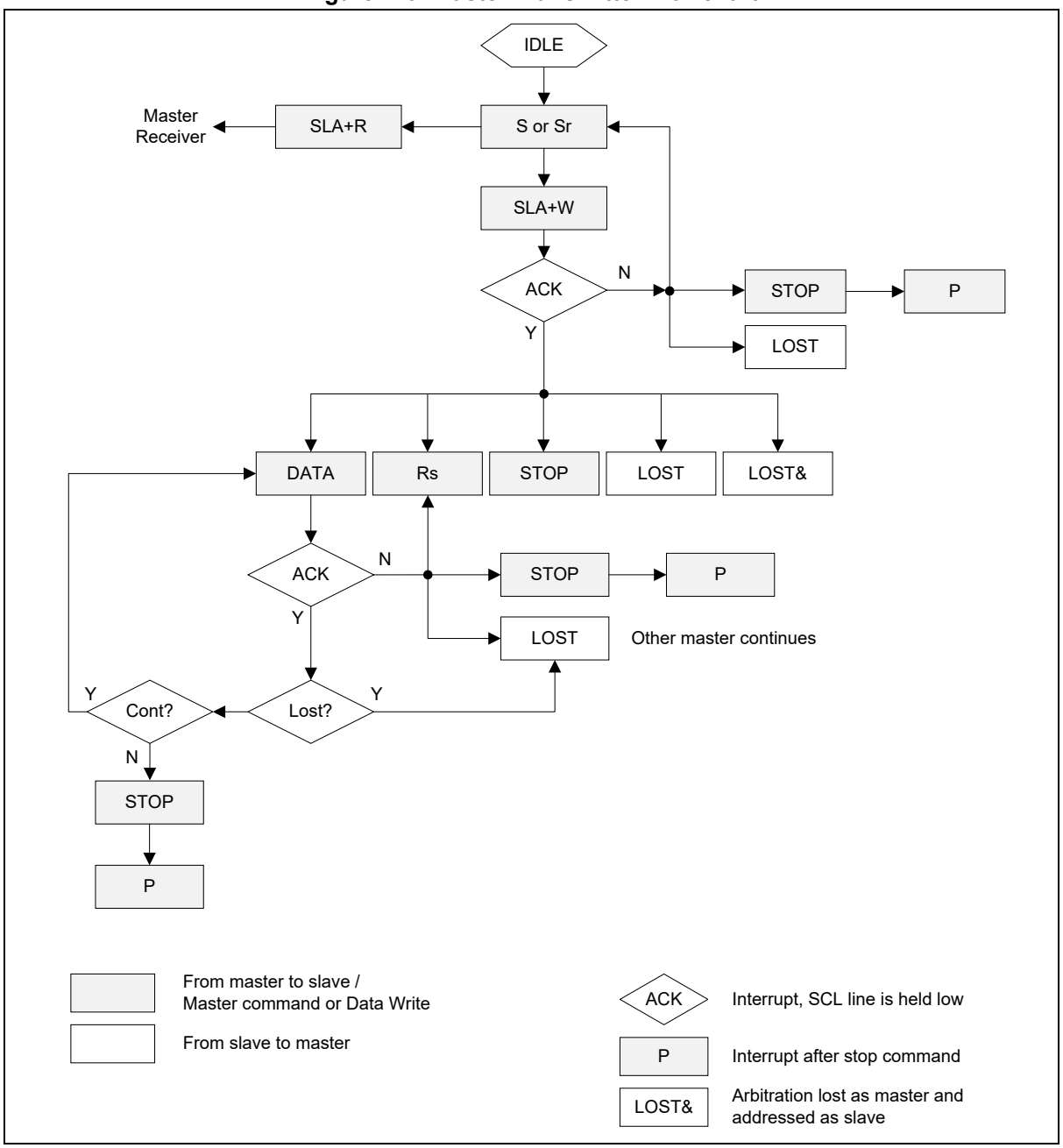
   Once one of the above operations is performed, release the SCL line by writing a value 0xFF to the I2Cn_SR register. For Case 1, go to step 8 and end the communication. For Case 2, go to step 6. In either case, the consecutive START condition can be detected; then, go to step 4.
7. The last step is to 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.

13.4.8 I2C Master Mode

13.4.8.1 Master Transmitter

Figure 119 shows a flowchart of the transmitter in master mode.

Figure 119. 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. Enter “SLA+W” in the I2C_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.
3. Set the I2Cn_SCLL and I2Cn_SCLH registers to configure the SCL transfer speed.
4. Set the I2Cn_SDH register, which determines the time SCL maintains the SDA value during the ‘L’ period.
5. 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.
6. 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_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 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.
7. One byte of data is transmitted. Bus arbitration is valid during data transmission.

8. 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’.

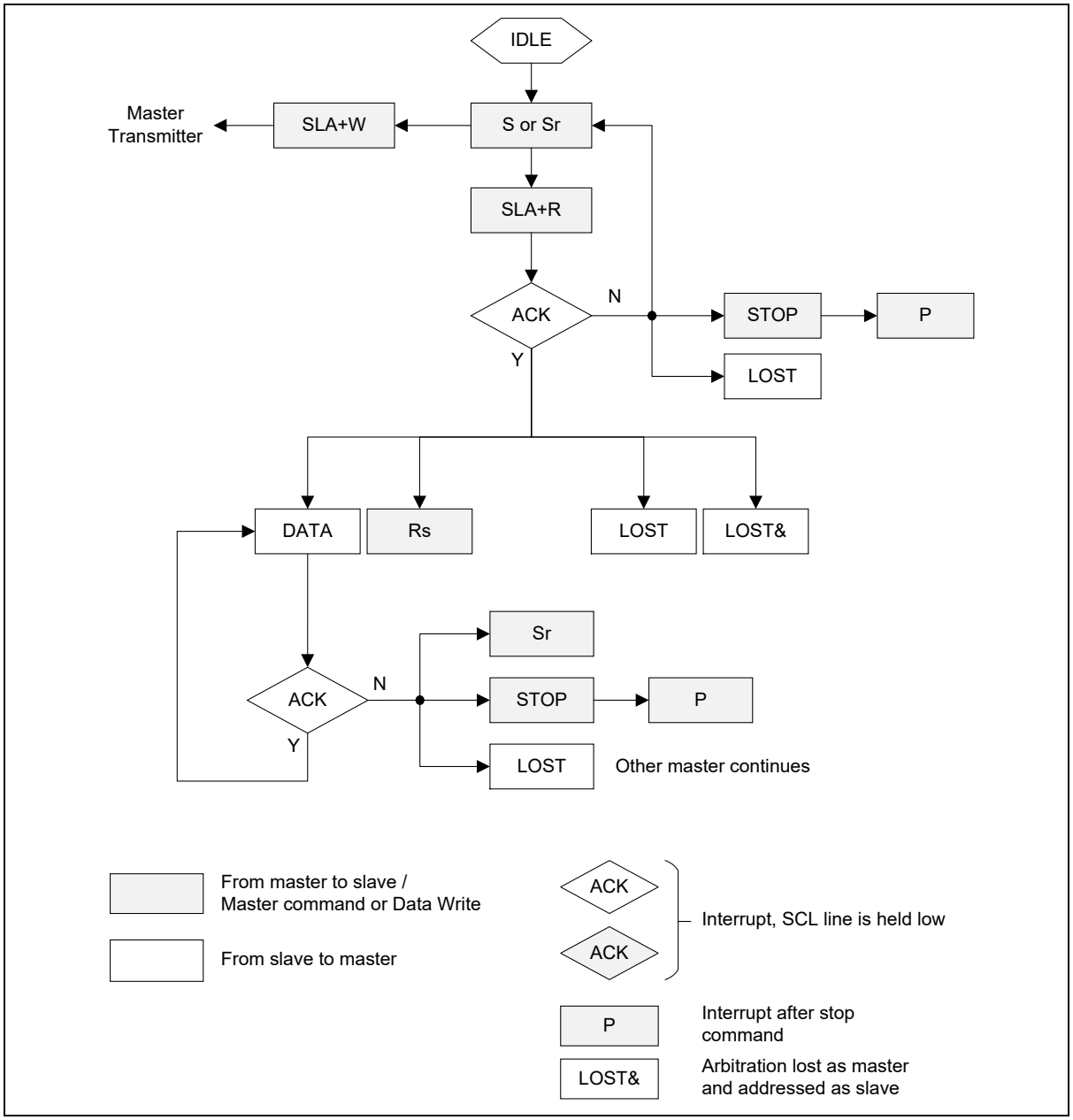
   The SCL line is released when one of the above is performed and a value 0xFF is written to the I2Cn_SR register. For Case A, go to step 8; for Case B, go to step 10; and for Case C, write the data in the I2Cn_DR register and go to step 6. If the transfer direction bit in SLA+R/W that is written to the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

9. 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.

13.4.8.2 Master Receiver

Figure 120 shows a flowchart of the receiver in master mode.

Figure 120. 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. 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.

3. Set the I2Cn_SCLL and I2Cn_SCLH registers to configure the SCL transfer speed.

4. Set the I2Cn_SDH register, which determines the time SCL maintains the SDA value during the ‘L’ period.

5. 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.

6. 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. Therefore, the I2C is ready to receive data. In this case, the receive data is read from the I2C_DR register.

   Case 2: Even if an ACK signal has been received, the master can stop reception. In this case, the STOP in I2C_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 I2C_DR register; and the START in I2C_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 I2C_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 I2C_DR register and go to step 6. If the transfer direction bit in SLA+R/W that is written to the I2C_DR register is set to ‘0’, go to the section on master transmitter.

7. One byte of data is received.

8. 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 I2C_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 I2C_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 I2C_CR register must be set enabled.

   Case D: ACK signals are not generated, and the master sends the consecutive START condition. In this case, “SLA+R/W” must be loaded to the I2C_DR register; and the START in I2C_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 I2C_SR register. For Cases A, go to step 8; for Case B and Case C, go to step 10; and for Case D, write the data in the I2C_DR register and go to step 6. If the transfer direction bit in SLA+R/W that is written to the I2C_DR register is set to ‘0’, go to the section on master transmitter.

9. 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 I2C_SR register, write a value 0xFF to the register. This causes the I2C to become idle.

13.5 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 13.4.7 and section13.4.8.

13.6 I2C Registers

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

Table 103. Base Address of I2C

Name	Base Address
I2C0	0x4000_A000
I2C1	0x4000_A100
I2C2	0x4000_A200

Table 104. I2C Register Map

Name	Offset	Type	Description	Reset Value	Reference
I2Cn_DR	0x0000	RW	I2Cn Data Register	0x0000_00FF	13.6.1
I2Cn_SR	0x0008	RW	I2Cn Status Register	0x0000_0000	13.6.2
I2Cn_SAR	0x000C	RW	I2Cn Slave Address Register 0	0x0000_0000	13.6.3
I2Cn_CR	0x0014	RW	I2Cn Control Register	0x0000_0000	13.6.4
I2Cn_SCLL	0x0018	RW	I2Cn SCL Low-Duration Register	0x0000_FFFF	13.6.5
I2Cn_SCLH	0x001C	RW	I2Cn SCL High-Duration Register	0x0000_FFFF	13.6.6
I2Cn_SDH	0x0020	RW	I2Cn SDA Hold Register	0x0000_7FFF	13.6.7

NOTES:

1. A31C14x has 3 channels of I2C (n = 0 to 2).
2. A31C12x has 2 channels of I2C (n = 0 to 1).

13.6.1 I2Cn_DR: I2C Data Register

The I2Cn_DR register is a 32-bit register. It stores received byte-size data or serial data for transmission.

I2C0_DR=0x4000_A000, I2C1_DR=0x4000_A100
I2C2_DR=0x4000_A200

31 30 29 28 27 26 25 24	23 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.

13.6.2 I2Cn_SR: I2C Status Register

The I2Cn_SR register is a 32-bit read/write accessible register. It displays the status of the I2C bus interface. Writing to the register clears the status bits. Once an I2C interrupt other than the stop interrupt occurs, the SCL line is set low. To release this SCL setting, a value 0xFF must be written to the SR register. This will clear the status of the GCALL, TEND, STOP, SSEL, MLOST, and RXACK bits.

I2C0_SR=0x4000_A008, I2C1_SR=0x4000_A108
I2C2_SR=0x4000_A208

31 30 29 28 27 26 25 24	23 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 the ACK address (AACK) has been received from a slave. If the I2C module is a slave, it indicates whether 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.

13.6.3 I2Cn_SAR: I2C Slave Address Register

The I2Cn_SAR register is an 8-bit readable and writable register. The first seven bits store the address selected when the I2C module operates as a slave.

I2C0_SAR=0x4000_A00C, I2C1_SAR=0x4000_A10C
I2C2_SAR=0x4000_A20C

31 30 29 28 27 26 25 24	23 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]	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.

13.6.4 I2Cn_CR: I2C Control Register

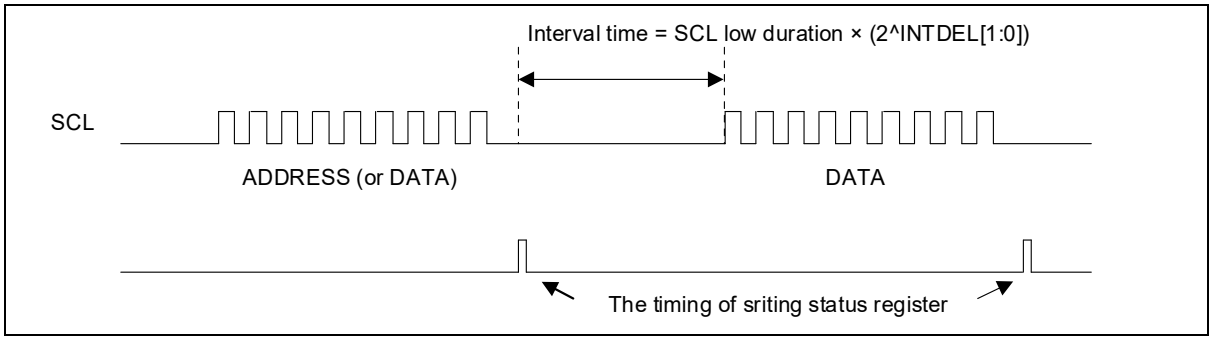
The I2Cn_CR register is a 32-bit read/write accessible register. It sets the operating modes and enablement of the I2C module.

I2C0_CR=0x4000_A014, I2C1_CR=0x4000_A114
I2C2_CR=0x4000_A214

31 30 29 28 27 26 25 24	23 22 21 20 19 18 17 16	15 14 13 12 11 10 9 8	7	6 5	4	3	2	1	0
Reserved			INTDEL[1:0]	IIF	Reserved	INTEN	ACKEN	Reserved	STOP	START
-			00	0	-	0	0	-	0	0
-			RW	RO	-	RW	RW	-	RW	RW

9 8	INTDEL[1:0]	Internal delay between address and data transfers (Or between two data transfers)
	0	1 × SCL low duration
	1	2 × SCL low duration
	2	4 × SCL low duration
	3	8 × SCL low duration
7	IIF	Interrupt flag
	0	No interrupt has occurred, or the flagged interrupt has been cleared.
	1	An interrupt has occurred.
4	INTEN	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.
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 start in master mode
	0	The I2C waits in slave mode.
	1	The I2C starts transmitting in master mode.

Figure 121. Internal Delay (INTDEL[1:0]) in Master Mode



13.6.5 I2Cn_SCLL: I2C SCL Low-Duration Register

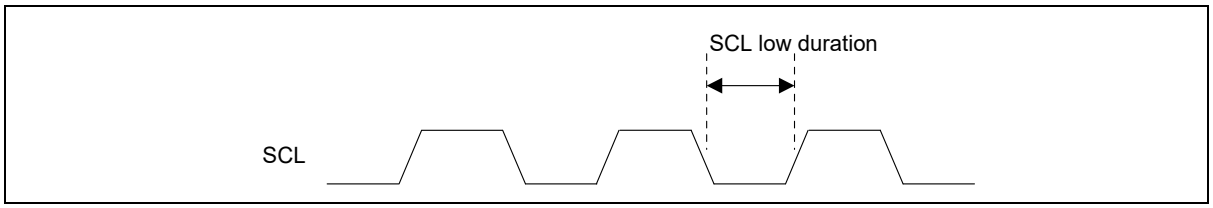
The I2Cn_SCLL register is a 32-bit register. The SCL low duration time can be set in master mode.

I2C0_SCLL=0x4000_A018, I2C1_SCLL=0x4000_A118
I2C2_SCLL=0x4000_A218

31 30 29 28 27 26 25 24	23 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 122. SCL Low (SCLL[15:0]) Timing



13.6.6 I2Cn_SCLH: I2C SCL High-Duration Register

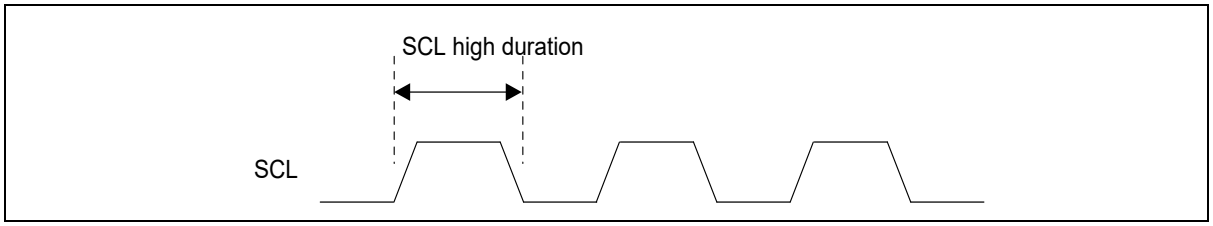
The I2Cn_SCLH register is a 32-bit register. The SCL high duration time can be set in master mode.

I2C0_SCLH=0x4000_A01C, I2C1_SCLH=0x4000_A11C
I2C2_SCLH=0x4000_A21C

31 30 29 28 27 26 25 24	23 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 123. 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).

13.6.7 I2Cn_SDH: I2C SDA Hold Register

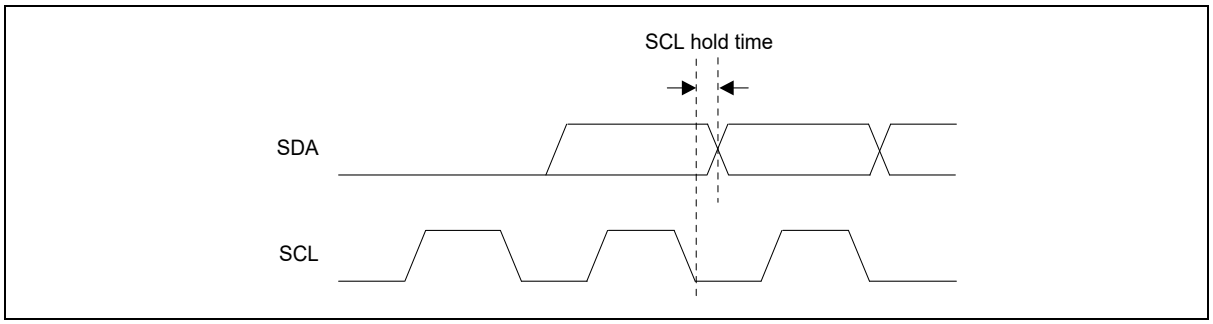
The I2Cn_SDH register is a 15-bit R/W register. The SCL hold time can be set in master mode.

I2C0_SDH=0x4000_A020, I2C1_SDH=0x4000_A120
I2C2_SDH=0x4000_A220

31 30 29 28 27 26 25 24	23 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 124. SDA Hold (SDH[14:0]) Timing



13.6.8 I2C Register Map Summary

Table 105. 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_SAR	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	SVAD[6:0]							GCEN
	Reset value																									0	0	0	0	0	0	0	0
0x14	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	Res.	Res.	INTEN	ACKEN	IMASTER	STOP	START
	Reset value																							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

14. USART

14.1 USART Introduction

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

14.2 USART Main Features

The USART of the A31C14x and A31C12x series 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.

14.3 UART Functional Description

The 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.

14.3.1 USART Block Diagram

Figure 125 shows the block diagram for the USART of the A31C14x and A31C12x series.

Figure 125. UART Block Diagram

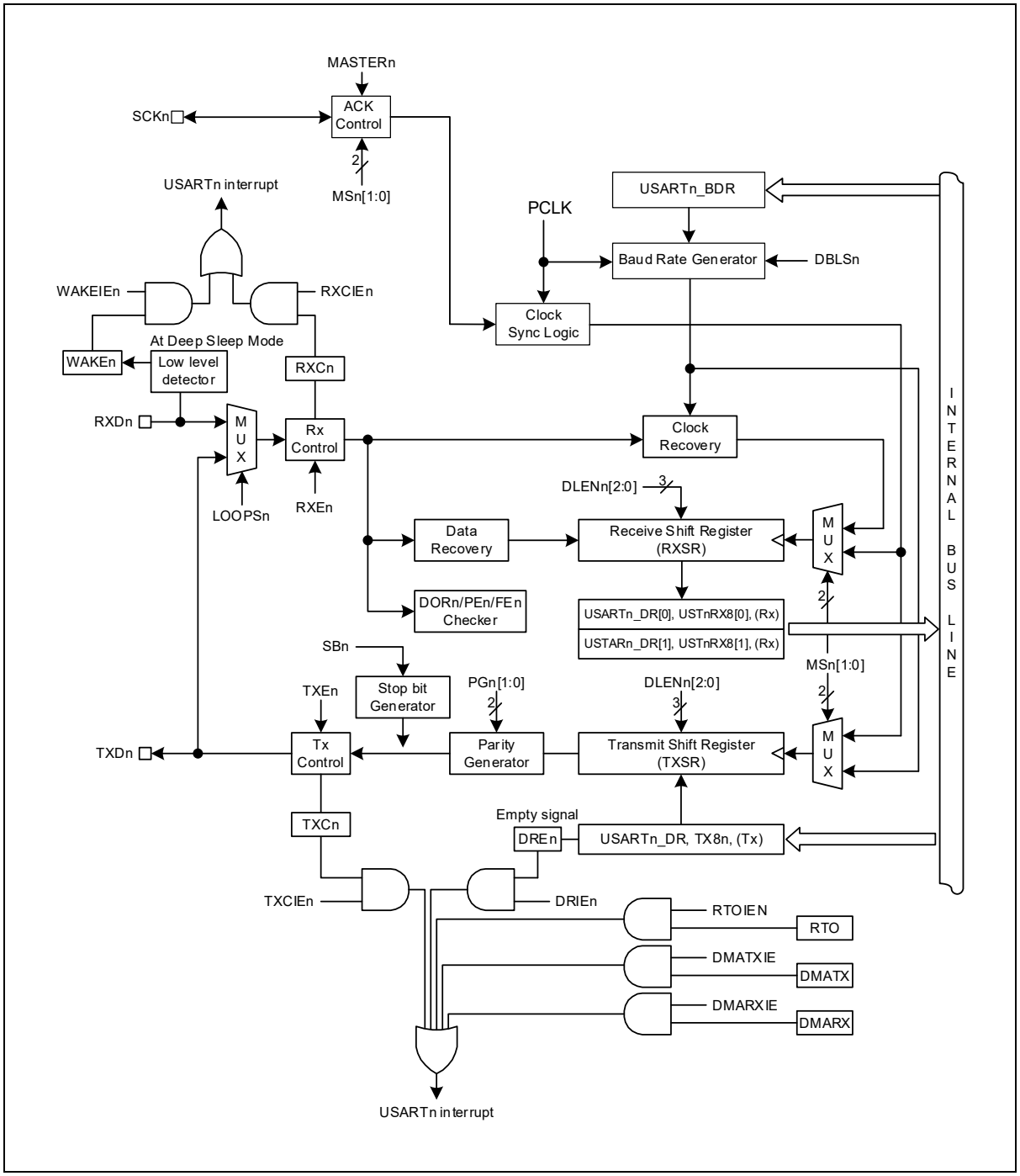
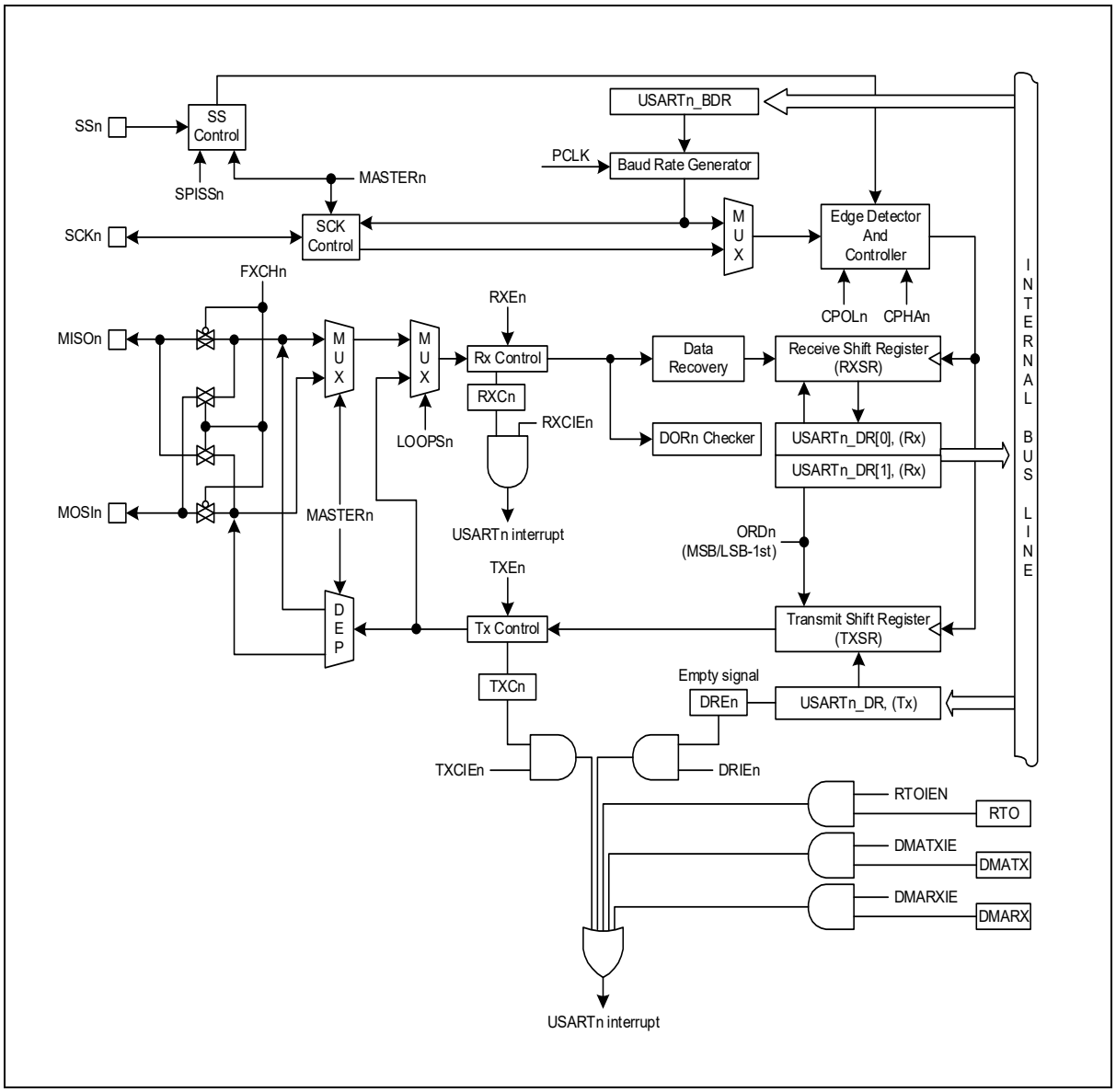


Figure 126. SPI Block Diagram



14.3.2 USART Pins and Internal Signals

Table 106 describes the pins assigned for the USART.

Table 106. Pin Assignment of USART External Pins

Pin Name	Type	Description	Package Type							
			64-LQFP	48-QFN, LQFP	44-LQFP	32-QFN, LQFP	28-TSSOP	24-QFN	20-QFN	20-TSSOP
TXD10	O	UART transmit output	O	O	O	O	O	O	O	O
RXD10	I	UART transmit input	O	O	O	O	O	O	O	O
SS10	I/O	SPI serial port I/O signal for slave selection	O	O	O	O	O	O	O	O
SCK10	I/O	SPI clock I/O (Master: output/slave: input)	O	O	O	O	O	O	O	O
MOSI10	I/O	SPI transmit and receive data (Master: output/slave: input)	O	O	O	O	O	O	O	O
MISO10	I/O	SPI transmit and receive data (Master: input/slave: output)	O	O	O	O	O	O	O	O
TXD11	O	UART transmit output	O	O	O	O	O	O	-	-
RXD11	I	UART transmit input	O	O	O	O	O	O	-	-
SS11	I/O	SPI serial port I/O signal for slave selection	O	O	O	-	-	-	-	-
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	O	O	O	O	-	-
MISO11	I/O	SPI transmit and receive data (Master: input/slave: output)	O	O	O	O	O	O	-	-
TXD12	O	UART transmit output	O	-	-	-	-	-	-	-
RXD12	I	UART transmit input	O	-	-	-	-	-	-	-
SS12	I/O	SPI serial port I/O signal for slave selection	O	-	-	-	-	-	-	-
SCK12	I/O	SPI clock I/O (Master: output/slave: input)	O	-	-	-	-	-	-	-
MOSI12	I/O	SPI transmit and receive data (Master: output/slave: input)	O	-	-	-	-	-	-	-
MISO12	I/O	SPI transmit and receive data (Master: input/slave: output)	O	-	-	-	-	-	-	-

14.3.3 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. 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 127. Clock Generation Block Diagram

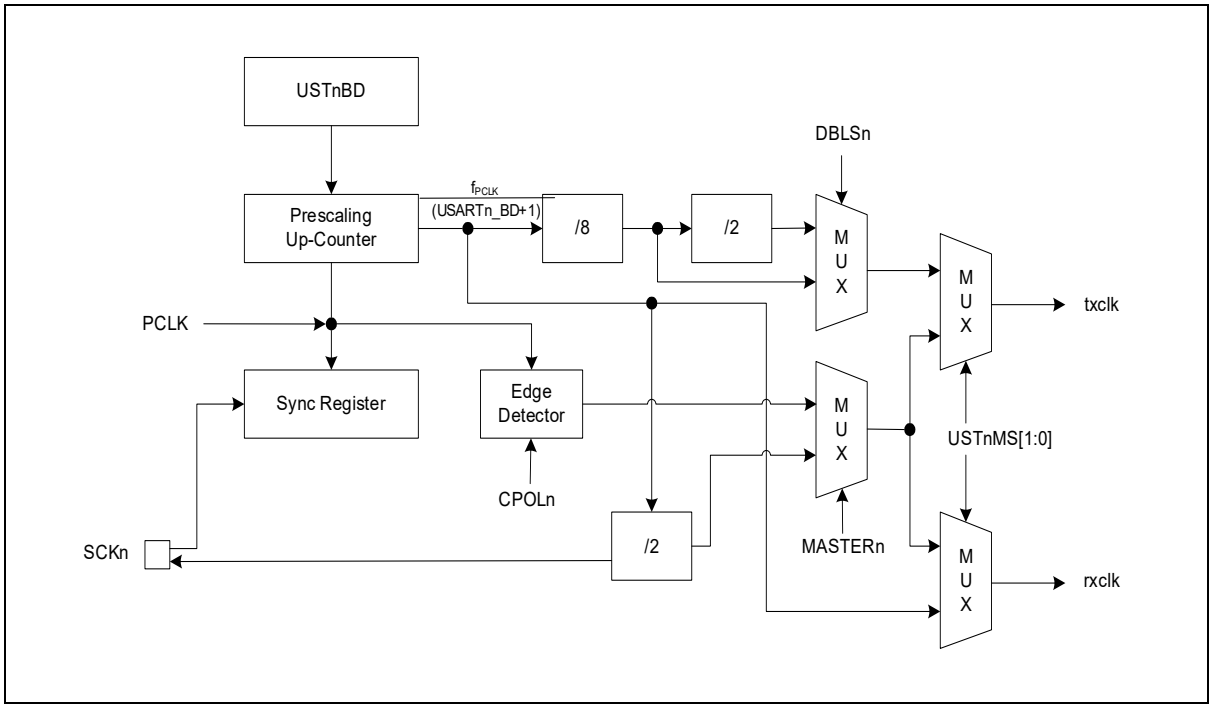


Table 107 shows the equations for calculating baud rate (in bps.)

Table 107. Equations for Calculating USART Baud Rate Register Settings

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))

14.3.4 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.

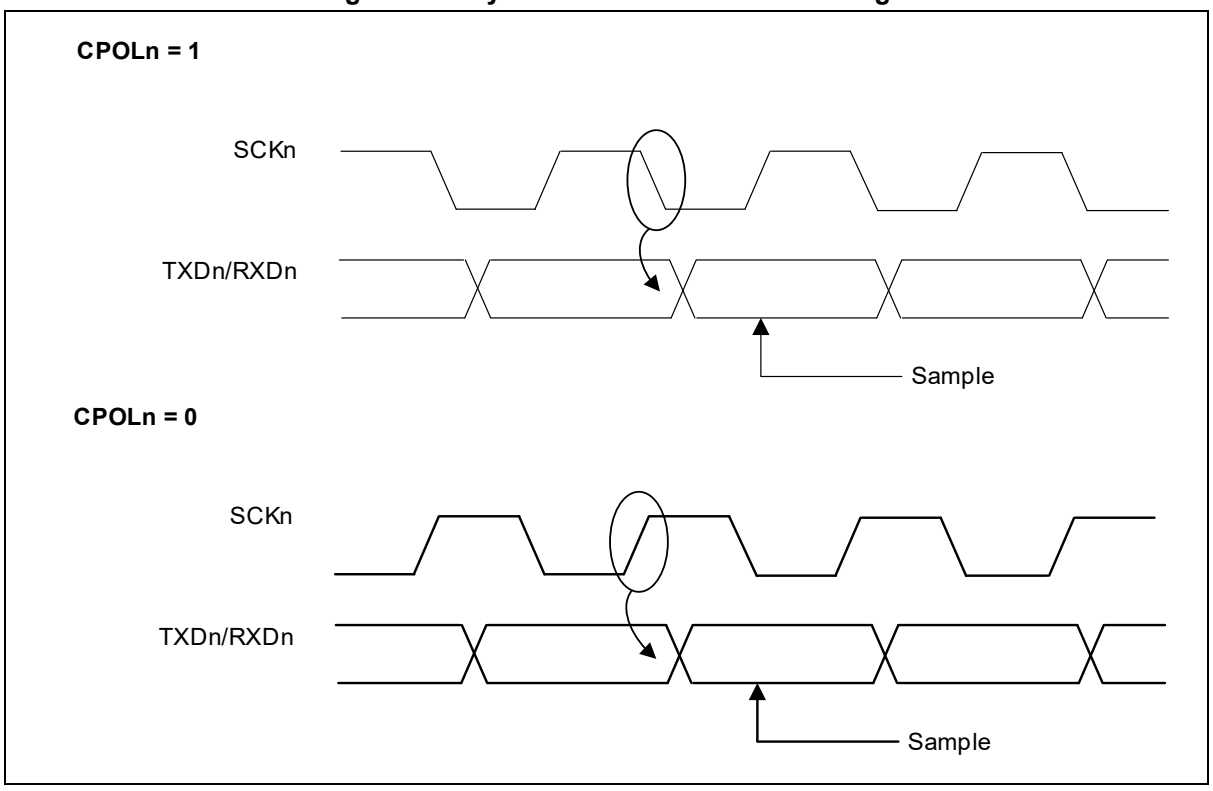
14.3.5 Synchronous Mode Operation

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 128, when CPOL[7] is zero, the data will be changed at rising edge of SCKn and sampled at falling edge of SCKn.

Figure 128. Synchronous Mode SCKn Timing



14.3.6 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.

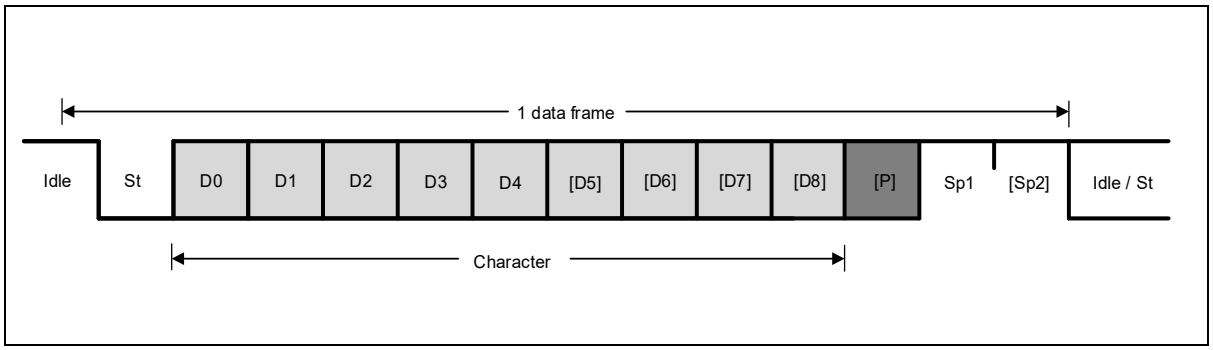
- 1 start bit
- 5, 6, 7, 8 or 9 data bits
- None, even or odd parity bit.
- 1 or 2 stop bits.

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 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 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 129 shows possible combinations of the frame formats. Bits inside brackets are optional.

Figure 129. Frame Formant (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: Even 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.

14.3.7 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

14.3.8 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.

14.3.8.1 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.

14.3.8.2 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.

14.3.8.3 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.

14.3.8.4 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.

14.3.9 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.

14.3.9.1 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.

14.3.9.2 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”.

14.3.9.3 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.

14.3.9.4 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.

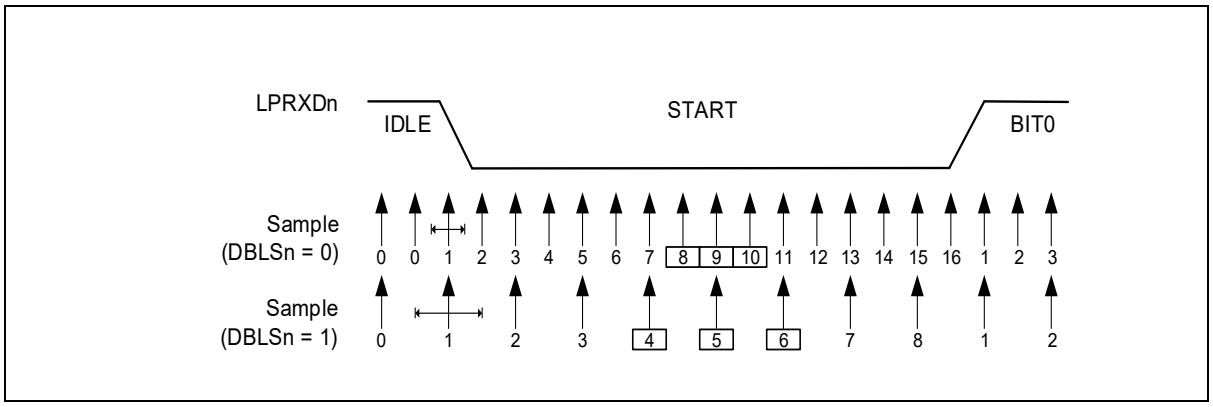
14.3.9.5 Asynchronous Data Reception

To receive 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.

Figure 130. Asynchronous Start Bit Sampling

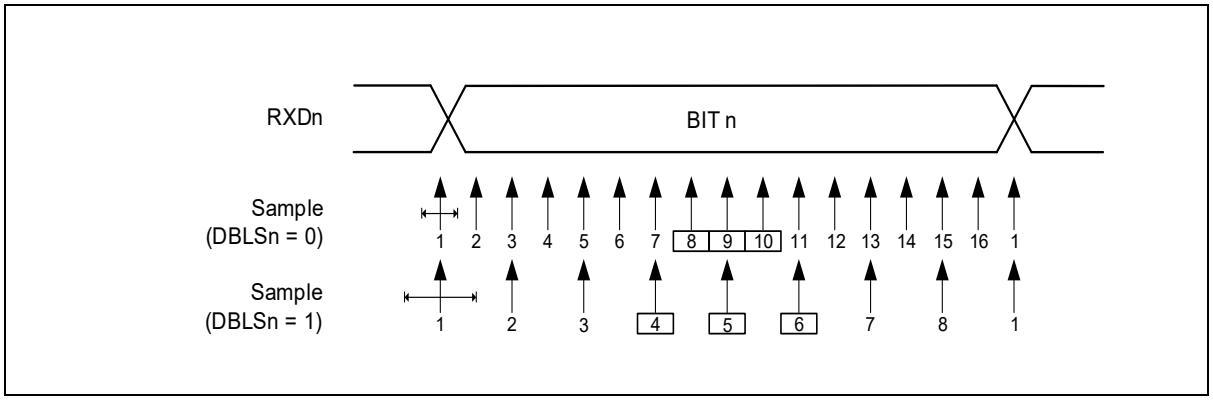


When the receiver is enabled (RXE[0] = 1), the clock recovery logic tries to find a high-to-low transition on the RXDn line, the start bit condition. After detecting high to low transition on RXDn line, the clock recovery logic uses samples 8, 9 and 10 for normal mode to decide if a valid start bit is 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 to 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 2 samples have high levels, the received bit is considered to logic ‘1’.

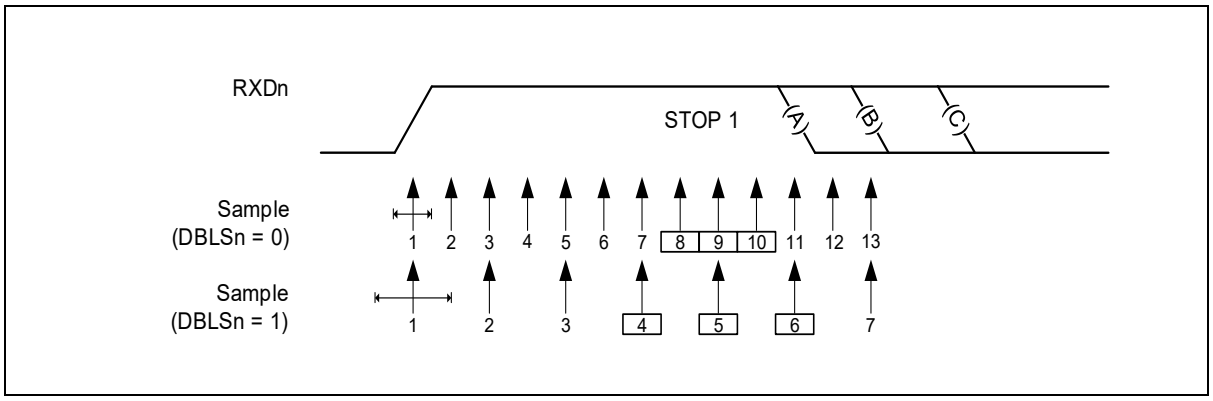
The data recovery process is then repeated until a complete frame is received including the first stop bit. The decided bit value is stored in the receive shift register in order. Note that the Receiver only uses the first stop bit of a frame. Internally, after receiving the first stop bit, the Receiver is in idle state and waiting to find start bit.

Figure 131. Asynchronous Data and Parity Bit Sampling



The process for detecting stop bit is same as clock and data recovery process. That is, if 2 or more samples of 3 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).

Figure 132. Stop Bit Sampling and Next Start Bit Sampling



14.3.10 UART Low-Power Mode

Table 108. Effect of Low-Power Modes on UART

Mode	Description
SLEEP	No effect
DEEP-SLEEP	Only UART mode operates as a wakeup source.

14.3.10.1 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’. 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.

14.3.11 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.

14.3.11.1 SPI Clock Formats and Timing

To accommodate a wide variety of 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 109 shows four combinations of CPOL and CPHA for SPI modes 0, 1, 2 and 3.

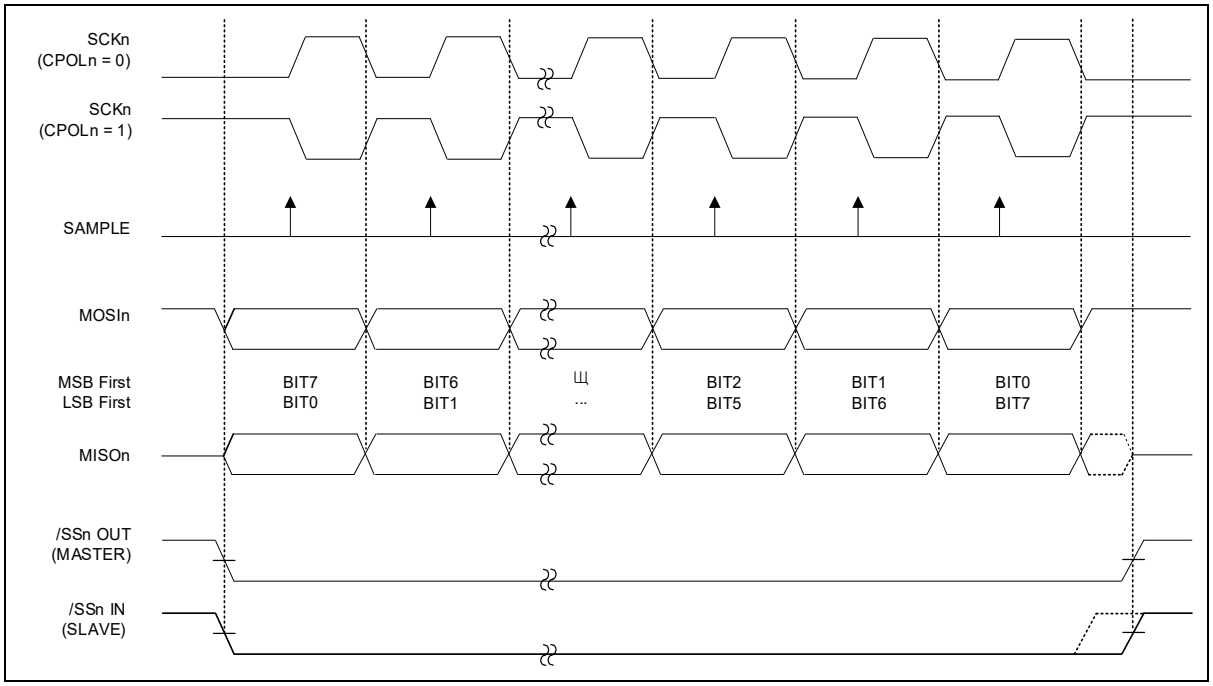
Table 109. CPOL Functionality

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.

Figure 133. USART SPIn Clock Formats (CPHAn = 0)

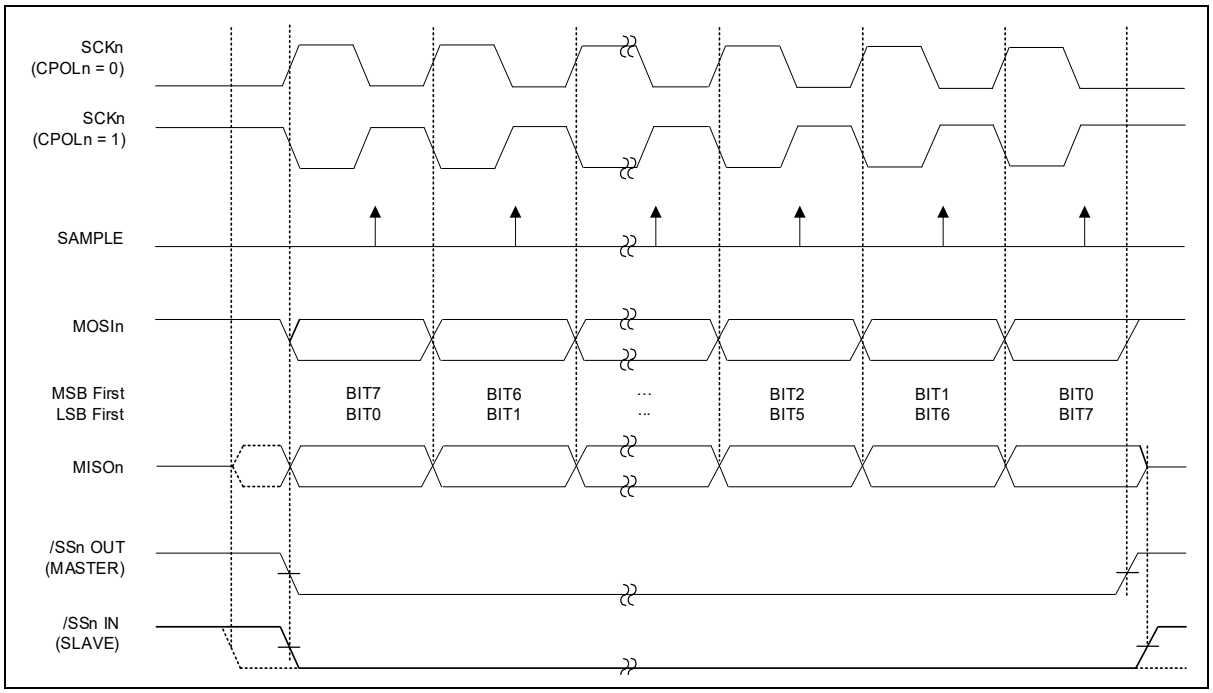


When CPHA[6] = 1, the slave begins to drive its MISOn output when SSn goes active low, but the data is not defined until the first SCKn edge. The first SCKn edge shifts the first bit of data from the shifter onto the MOSIn output of the master and the MISOn output of the slave.

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 134. USART SPIn Clock Formats (CPHAn = 1)



14.3.12 Baud Rate Settings (Example)

Table 110. Baud Rate Settings Example

Baud Rate	fOSC=1.00MHz				fOSC=1.8432MHz				fOSC=2.00MHz			
	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=3.6864MHz				fOSC=4.00MHz				fOSC=7.3728MHz			
	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=8.00MHz				fOSC=11.0592MHz				fOSC=14.7456MHz			
	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%

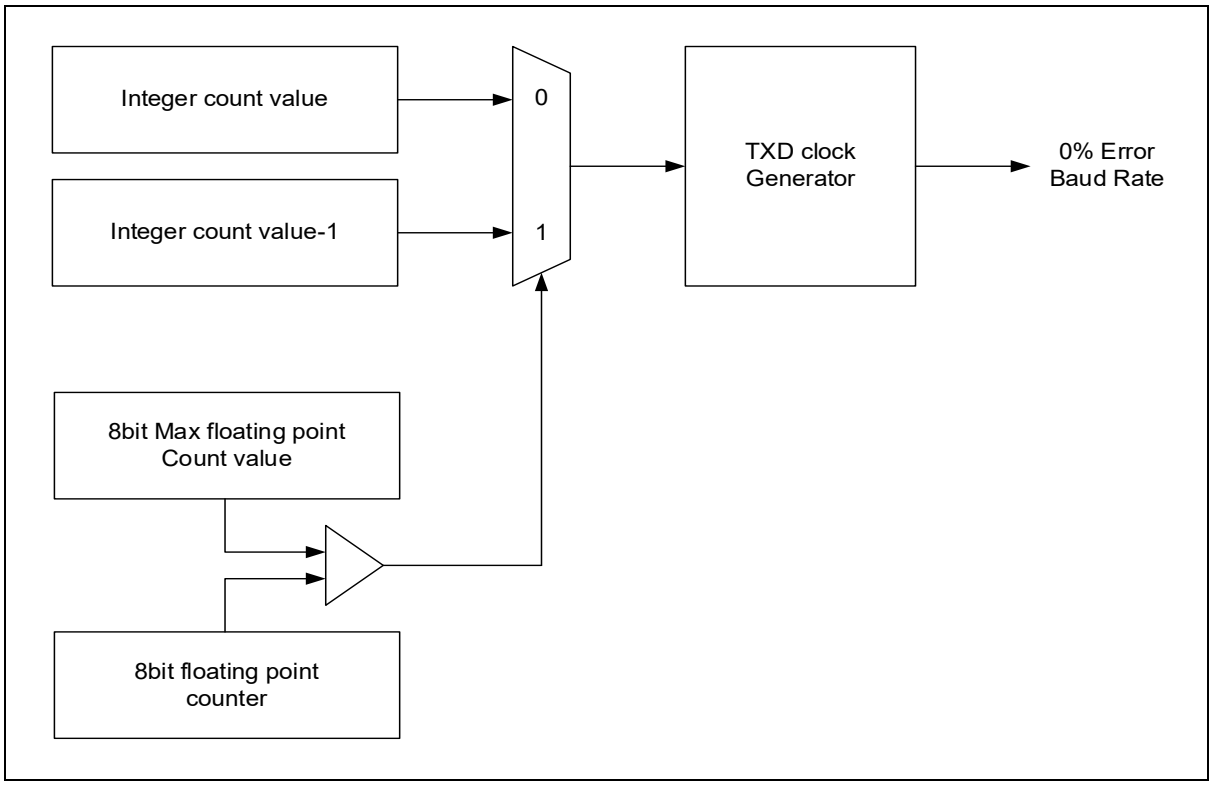
14.3.13 0% Error Baud Rate

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, If users want to use the 57600 baud rate (fXIN = 16 MHz), a calculated integer count value must be 17.36 value (USARTn_BDR+1 = 16000000/(16×57600) = 17.36).

Here, 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) × 256 ≒ 92). Namely, the users must write 92 (decimal number) to USARTn_BFR and 16 (decimal number) to USART_BDR register.

Figure 135. 0% Error Baud Rate Diagram



14.3.14 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. Users can use RTO interrupts with RTOIE.

14.4 USART Registers

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

Table 111. Base Address of USART

Name	Base Address
USART10	0x4000_1000
USART11	0x4000_1100
USART12	0x4000_1200

Table 112. USART Register Map

Name	Offset	Type	Description	Reset Value	Reference
USARTn_CR1	0x0000	RW	USART n control register 1	0x0000_0000	14.4.1
USARTn_CR2	0x0004	RW	USART n control register 2	0x0000_0000	14.4.2
USARTn_ST	0x000C	RW	USART n status register	0x0000_0080	14.4.3
USARTn_BDR	0x0010	RW	USART n baud rate generation register	0x0000_0FFF	14.4.4
USARTn_DR	0x0014	RO	USART n data register	0x0000_0000	14.4.5
USARTn_BFR	0x0018	RW	USART n baud-rate fraction counter register	0x0000_0000	14.4.6
USARTn_RTO	0x001C	RW	USART n receiver time out register	0x00FF_FFFF	14.4.7

NOTES:

1. A31C14x has 3 channels of USART (n = 10 to 12).
2. A31C12x has 2 channels of USART (n = 10 to 11).

14.4.1 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, 11 and 12)

USART10_CR1=0x4000_1000, USART11_CR1=0x4000_1100
USART12_CR1=0x4000_1200

31 30 29 28 27 26 25 24	23 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	Select 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 CPOL and this bit determine if data are sampled on the leading or trailing edge of SCK (only SPI mode).
		CPOL CPHA 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 CPOL and CPHA bits should be changed during the TXE and RXE bits are both '0's.

14.4.2 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, 11 and 12)

USART10_CR2=0x4000_1004, USART11_CR2=0x4000_1104
USART12_CR2=0x4000_1204

31 30 29 28 27 26 25 24	23 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	DISSCK	SREN	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	SREN	Enables the receiver unit. (Only SPI slave mode)
	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	Select the length of stop bit in Asynchronous or Synchronous mode.	
		0	1 stop bit.
		1	2 stop bits.
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’.

14.4.3 USARTn_ST: USART n Status Register

USARTn_ST register is 32-bit size and accessible in 32-/16-/8-bit (n=10, 11 and 12).

USART10_ST=0x4000_100C, USART11_ST=0x4000_110C
USART12_ST=0x4000_120C

31 30 29 28 27 26 25 24	23 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 (DMA 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 writing '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
5	RXC	Receive complete interrupt flag. This flag is set when there is 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 is 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 writing '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.

14.4.4 USARTn_BDR: USART n Baud Rate Generation Register

USARTn_BDR register is 32-bit size and accessible in 32-/16-/8-bit (n=10, 11 and 12).

USART10_BDR=0x4000_1010, USART11_BDR=0x4000_1110
USART12_BDR=0x4000_1210

31	30	29	28	27	26	25	24	23	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.

14.4.5 USARTn_DR: USART n Data Register

USARTn_DR register is 32-bit size and accessible in 32-/16-/8-bit (n=10, 11 and 12).

USART10_DR=0x4000_1014, USART11_DR=0x4000_1114
USART12_DR=0x4000_1214

31	30	29	28	27	26	25	24	23	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 is 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's.

14.4.6 USARTn_BFR: USART n Baud-Rate Fraction Counter Register

USARTn_BFR register is 32-bit size and accessible in 32-/16-/8-bit (n=10, 11 and 12).

USART10_BFR=0x4000_1018, USART11_BFR=0x4000_1118
USART12_BFR=0x4000_1218

31	30	29	28	27	26	25	24	23	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]	USART baud rate fraction counter. 8-bit floating point counter.

14.4.7 USARTn_RTO: USART n RTO Register

USARTn_RTO register is 32-bit size and accessible in 32-/16-/8-bit (n=10, 11 and 12).

USART10_RTO=0x4000_101C, USART11_RTO=0x4000_111C
USART12_RTO=0x4000_121C

31	30	29	28	27	26	25	24	23	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.

14.4.8 USART Registers Map Summary

Table 113. 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.	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	1

15. ADC

15.1 ADC Introduction

The ADC module of the A31C14x and A31C12x series is a 12-bit resolution Analog-to-Digital Converter (ADC) that consists of an independently controllable unit.

This ADC module can process A/D conversion of many different channels in Single mode, Sequential mode, Multiple mode and Burst mode. It can sample the voltage inputs of multiple channels at critical time, which is required for various applications and can be achieved using the triggering functionality in association with TIMER.

Once the A/D conversion is complete, the result is left aligned in 16-bit data registers.

The ADC is mapped to the AHB and capable of high-speed operation based on the internal clock source. In addition, it uses built-in DMA to reduce microcontroller core load while enabling fast data sampling.

The analog input monitoring and comparison capabilities of the ADC allow the applications to determine whether the input voltage is above or below the predefined level.

15.2 ADC Main Features

Main features of the ADC module are listed below:

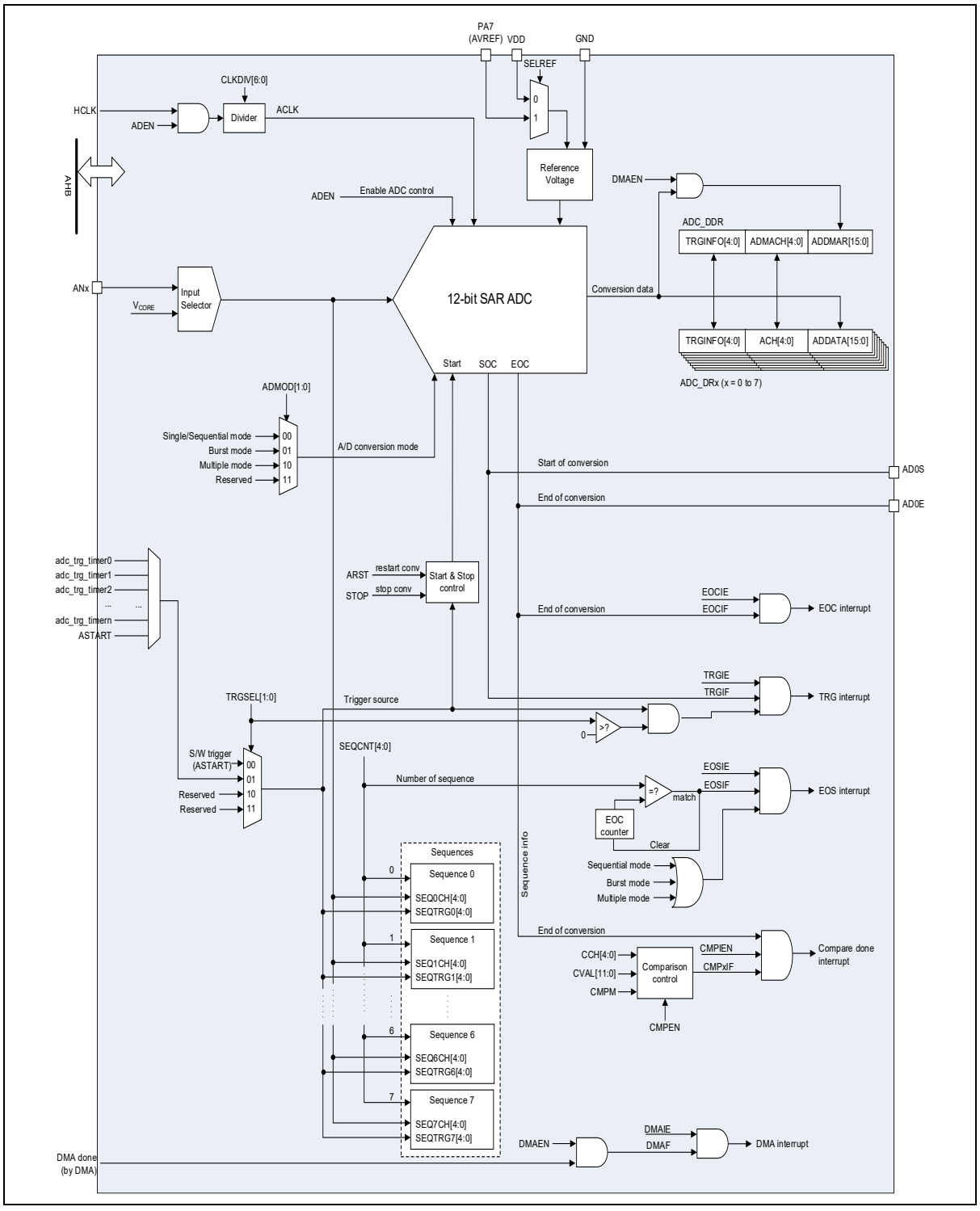
- Composed of one independent ADC unit
- 12-bit analog to digital resolution
- 150 ksps maximum conversion rate with full resolution
- 2.4 V minimum to 5.5 V operating voltage range
- Data can be managed by DMA for high-performance conversion.
- Analog Input channels
  - A31C14x: 18 external channels
  - A31C12x: 13 external channels
- Internally dedicated channels
  - AVREF internal reference voltage outputs connected to ADC
- Four conversion modes
  - Single mode
  - Sequential mode
  - Multiple mode
  - Burst mode
- Up to 8 sequential conversions can be supported.
- Software trigger sources per channel
  - Internal software trigger source
  - TIMERn 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 ≤ AVREF ≤ VDD

15.3 ADC Functional Description

15.3.1 ADC Block Diagram

Figure 136 shows a block diagram of the 12-bit ADC.

Figure 136. 12-bit ADC Block Diagram



15.3.2 ADC Pins and Signals

Table 114 describes pins assigned for the ADC.

Table 114. Pin Assignment of ADC External Pins

Pin Name	Signal Type[1]	Description	Package Type							
			64-LQFP	48-LQFP, QFN	44-LQFP	32-QFN, LQFP	28-TSSOP	24-QFN	20-QFN	20-TSSOP
AVREF	P	Input, analog reference voltage AVREF ≤ VDD (refer to datasheet)	O	O	O	O	O	O	O	O
AN0	A	ADC external analog input 0	O	O	-	-	-	-	-	-
AN1	A	ADC external analog input 1	O	O	-	-	-	-	-	-
AN2	A	ADC external analog input 2	O	O	O	O	O	-	-	-
AN3	A	ADC external analog input 3	O	O	O	O	-	-	-	-
AN4	A	ADC external analog input 4	O	O	O	O	-	-	-	-
AN5	A	ADC external analog input 5	O	O	O	O	-	-	-	-
AN6	A	ADC external analog input 6	O	O	O	O	-	-	-	-
AN7	A	ADC external analog input 7	O	O	O	O	O	O	O	O
AN8	A	ADC external analog input 8	O	O	O	O	O	O	O	O
AN9	A	ADC external analog input 9	O	O	O	O	O	O	O	O
AN10	A	ADC external analog input 10	O	O	O	O	O	O	O	O
AN11	A	ADC external analog input 11	O	-	-	-	-	-	-	-
AN12	A	ADC external analog input 12	O	-	-	-	-	-	-	-
AN13	A	ADC external analog input 13	O	-	-	-	-	-	-	-
AN14	A	ADC external analog input 14	O	O	O	O	O	O	O	O
AN15	A	ADC external analog input 15	O	O	O	O	O	O	O	O
AN16	A	ADC external analog input 16	O	O	O	O	O	O	O	O
AN17	A	ADC external analog input 17	O	O	O	O	O	O	O	O

NOTE:

1. P: Power, O: Output, A: Analog Input

Table 115 describes the input and output signals of the ADC.

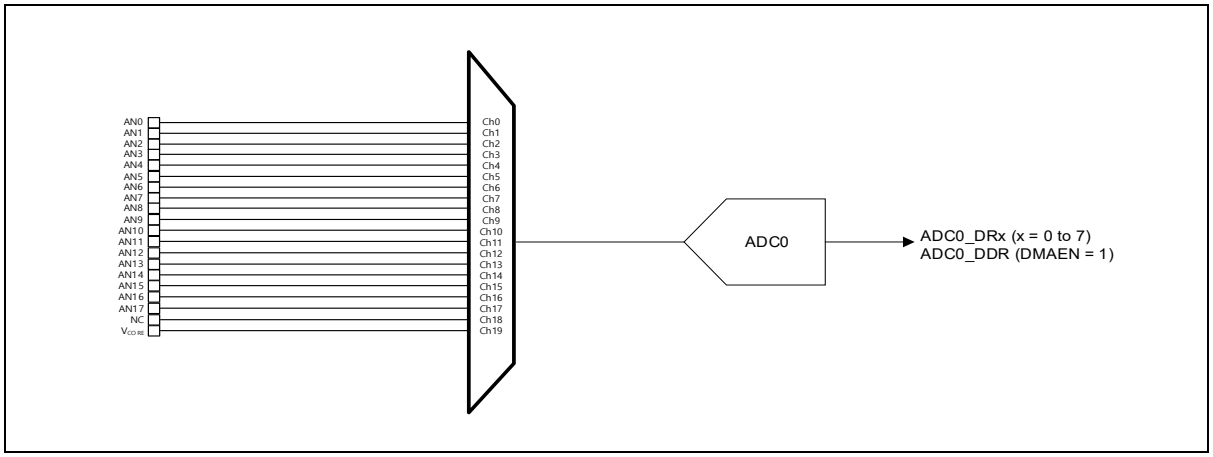
Table 115. ADC Input/Output Signals

Signal Name	Signal Type	Description
ACLK	clock	ADC operating clock
adc_trg_timerx	Inputs	The internal trigger source of TIMERn
adc_trg_astart	Inputs	The internal clock of ADC
adc_ch[17:0]	Analog Input	ADC external input signal
V_{CORE}	Analog Input	Output voltage from internal reference voltage (1.59V)
adc_out	Digital Output	A/D conversion data by ADC

15.3.3 ADC Connectivity

15.3.3.1 Internal Wiring of ADC Channels

Figure 137. Internal Wiring Diagram of 12-bit ADC Channels



15.3.3.2 ADC Channels and External Input Pin

Table 116. ADC Channels and External Input Pins (1)

ADC Channel	SEQxCH (x = 0 to 7)	External Input Pin
adc_ch0	0	AN0
adc_ch1	1	AN1
adc_ch2	2	AN2
adc_ch3	3	AN3
adc_ch4	4	AN4
adc_ch5	5	AN5
adc_ch6	6	AN6
adc_ch7	7	AN7
adc_ch8	8	AN8
adc_ch9	9	AN9
adc_ch10	10	AN10
adc_ch11	11	AN11
adc_ch12	12	AN12
adc_ch13	13	AN13
adc_ch14	14	AN14
adc_ch15	15	AN15
adc_ch16	16	AN16
adc_ch17	17	AN17
adc_ch18	18	NC
adc_ch19	19	V_{CORE}

NOTES:

1. SEQxCH (x = 0 to 7) are configured in the ADC_SCSR registers.
2. Single mode conversion uses the SEQ0CH bit field in the ADC_SCSR register and the ADC_DR0 register only.

15.3.4 ADC Clocks (ACLK)

15.3.4.1 ADC Module Control

Before enabling the ADC clock, users must select an ADC unit to be enabled by setting a bit of the ADC in the SCU_PER2 register to ‘1’ which is corresponding to the ADC unit. Then, enable the clock of the selected ADC unit by setting a bit of the ADC bit in the SCU_PCER2 register to ‘1’ which corresponds to the clock of the selected ADC unit.

To initialize the registers of each ADC unit, set each corresponding bit of the ADC in the SCU_PER2 register and the ADC bit in the SCU_PCER2 register to ‘0’ and then set back to ‘1’.

Each bit of the ADC in SCU_PER2 register controls the corresponding ADC unit. To enable or disable all ADC units at a time, set the ADC in SCU_PCER2 register to the number shown below:

- If the ADC bit in the SCU_PER2 register is ‘0’, ADC unit is disabled.
- If the ADC bit in the SCU_PER2 register is ‘1’, ADC unit is enabled.

To provide the ADC unit with the ADC clock, configure the ADC in SCU_PCER2 register referring to the settings shown below:

- If the ADC bit in the SCU_PCER2 register is ‘0’, ADC clock is disabled.
- If the ADC bit in the SCU_PCER2 register is ‘1’, ADC clock is enabled.

15.3.4.2 ADC Clock Sources

The HCLK operates as the ADC operation clock (ACLK) based on the AHB domain.

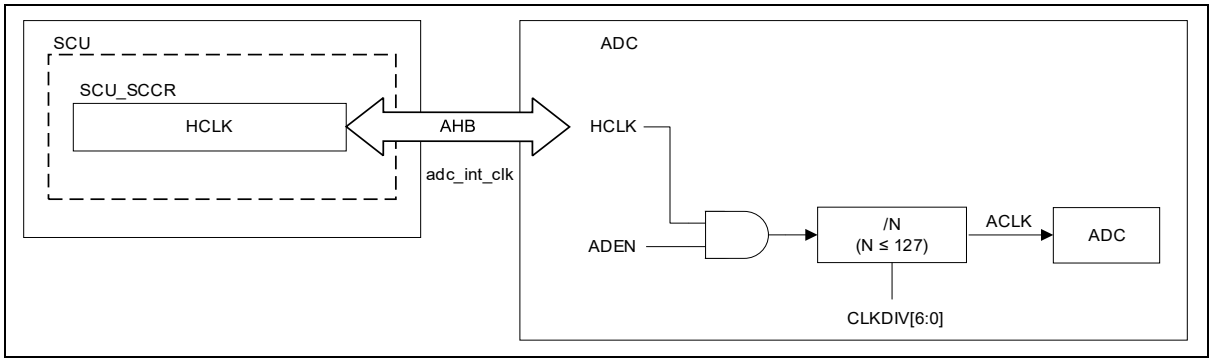
[Internal Clock Source]

The programmable divider coefficient can be selected to range from 0 to 127 by configuring the CLKDIV[6:0](1) value in the ADC_CCR register, and the HCLK is divided by the coefficient to be used as the ADC operation clock.

NOTE:

1. CLKDIV[6:0] value in the ADC_CCR register is used only when the HCLK, the ADC's internal system clock is used.

Figure 138. Block Diagram of ADC Clock



15.3.5 ADC On-Off Control (ADEN)

Before starting the A/D conversion, users must enable the ADC module using the ADEN in ADC_MR register.

- Setting the ADEN = '1' enables the ADC unit. When the ADC unit is enabled, the ADC functionality and the ASTART bit control are possible.
- Setting the ADEN = '0' disables the ADC unit. When the ADC unit is disabled, the A/D conversion cannot be performed.

15.3.5.1 Software Procedure to Enable ADC

1. Clear the EOCIF flag in the ADC_SR register by writing '1'.
2. Setting the ADEN bit in the ADC_MR register to '1' enables the ADC module.

15.3.5.2 Software Procedure to Disable ADC

1. Confirm that the ASTART and the ASTOP bits in the ADC_CR register are set to '0', respecti vely. This ensures that no conversion is ongoing. If required, set the ASTOP to '1' to stop the conversion and then wait until the ASTOP bit becomes '0'(1).
2. Disable the ADC module by setting the ADEN bit in the ADC_MR register to '0'

NOTE:

1. This ASTOP bit is auto cleared at the next ADC clock cycle.

15.3.6 Constraints when Writing ADC Control Bits

15.3.6.1 When Writing ADC Control Bit

While accessing the SCU_PER2, SCU_PCER2, and ADC_CCR registers to enable the ADC operation clock source and ADC module, the ADC module must be disabled. For this, set the ADEN bit in the ADC_MR register to ‘0’.

During the A/D conversion operations, if users write to each ADC control bit of the SCU_PER2, SCU_PCER2, and ADC_CCR registers, the ADC clock bus and ADC registers are not controllable and become an unknown state.

Software can write to the control bits configuring the A/D conversion, only when the ADC module is enabled (ADEN = ‘1’) and the A/D conversion is not performed (e.g. ASTART = ‘0’).

- Set the ADEN bit in the ADC_MR register to ‘0’.
- Clear all bits in the ADC_CR register.

15.3.7 Channel Selection

15.3.7.1 ADC Analog Input Channel

Each ADC has up to 18 multiplexed channels.

When the analog inputs come from GPIO ports:

1. For the analog inputs on the ADC channel, set the mode of the channel of 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.

15.3.7.2 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 bit field in the ADC_SCSR1 or ADC_SCSR2 registers. (x = 0 to 7)

For the A/D conversion operation in Single mode, use the SEQ0CH field in the ADCn_SCSR1 register to set the channel.

Table 117. ADC Input Signal and Input Pin [1]

ADC input signal	ADC_SCSR1				ADC_SCSR2			
	SEQ0CH[4:0][2]	SEQ1CH[4:0]	SEQ2CH[4:0]	SEQ3CH[4:0]	SEQ4CH[4:0]	SEQ5CH[4:0]	SEQ6CH[4:0]	SEQ7CH[4:0]
adc_ch0	0	0	0	0	0	0	0	0
adc_ch1	1	1	1	1	1	1	1	1
adc_ch2	2	2	2	2	2	2	2	2
adc_ch3	3	3	3	3	3	3	3	3
adc_ch4	4	4	4	4	4	4	4	4
adc_ch5	5	5	5	5	5	5	5	5
adc_ch6	6	6	6	6	6	6	6	6
adc_ch7	7	7	7	7	7	7	7	7
adc_ch8	8	8	8	8	8	8	8	8
adc_ch9	9	9	9	9	9	9	9	9
adc_ch10	10	10	10	10	10	10	10	10
adc_ch11	11	11	11	11	11	11	11	11
adc_ch12	12	12	12	12	12	12	12	12
adc_ch13	13	13	13	13	13	13	13	13
adc_ch14	14	14	14	14	14	14	14	14
adc_ch15	15	15	15	15	15	15	15	15
adc_ch16	16	16	16	16	16	16	16	16
adc_ch17	17	17	17	17	17	17	17	17

NOTES:

1. For the multi-channel sampling, up to 8 channels can be input and the number of conversions is set in the SEQCNT[2:0] in ADC_MR register.
2. For the single-channel sampling, the channel must be selected by configuring the SEQ0CH[4:0] in ADC_SCSR1 register. Once the A/D conversion starts with the ASTART in ADC_CR register set to '1', information of the channel currently converting is stored in the CACH[4:0] in ADC_CSCR register and the most recently sampled channel value is retained.

15.3.7.3 Register Access when Changing A/D Conversion Channel

The ADC_SCSR registers cannot be modified or changed during the A/D conversion. To change the channel to sample, write ‘1’ to the ASTOP bit in the ADC_CR register to stop the A/D conversion.

The user can select a number of conversions in a sequence by configuring the SEQCNT[2:0] in ADC_MR register. Updates are available before the Sequence A/D conversion, or immediately after the End of Sequence (EOS) before a new sequence is executed.

Assuming that the SEQCNT[2:0] is set to 3 and the sequence is configured in the order of adc_ch0, adc_ch1, adc_ch2, and adc_ch3 by using the SEQ0CH, SEQ1CH, SEQ2CH, and SEQ3CH, and the CSEQN[4:0] in ADC_CSCR register is set to 2, then the A/D conversion is performed in the order of adc_ch2 and adc_ch3, and the Sequence End Interrupt Flag, EOSIF in ADCn_SR register is set to ‘1’.

15.3.8 Single Mode Conversion (ADMOD[1:0] = ‘00’, SEQCNT[2:0] = 0)

Single mode conversion means that the A/D conversion is performed in a single channel. The Single mode performs the sampling and the A/D conversion of analog inputs on the channel selected in the SEQ0CH[4:0] bits in the ADC_SCSR1 register.

- If ADMOD[1:0] = ‘00’ and SEQCNT[2:0] = ‘000’, 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 trigger event

The SEQCNT[2:0] in ADC_MR register in Single mode:

- If SEQCNT[2:0] = ‘000’, Single channel sampling

Analog input channel in Single mode:

- If SEQ0CH[4:0] = x, adc_chx channel for A/D conversion

Trigger Event Source Settings:

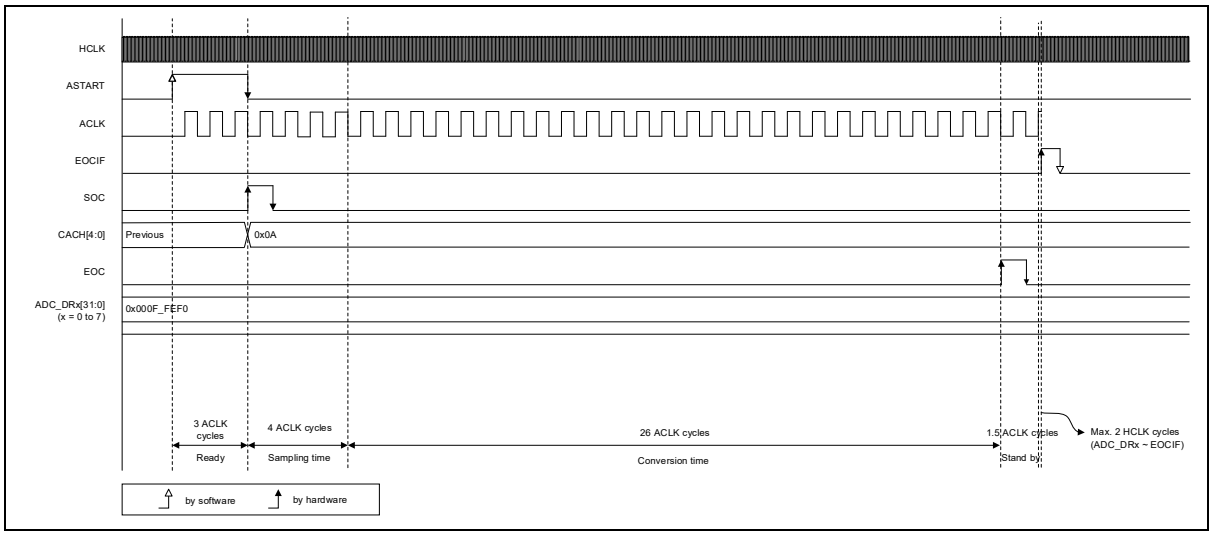
- If SEQTRG0[3:0] = ‘0000’ to ‘1111’, ADC trigger source

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 ADC_DRx register (x = 0 to 7), and the EOCIF flag in the ADC_SR register is set to ‘1’.

Users can obtain the A/D conversion result of the analog input by polling the EOCIF flag in the ADC_SR register using software or read-accessing the ADC_DRx register (x = 0 to 7) at the point of time when the EOCIF flag in the ADC_SR register interrupt occurs.

Figure 139. ADC Single Mode Timing (ADMOD[1:0] = '00')



15.3.9 Sequential Mode Conversion (ADMOD[1:0] = '00', SEQCNT[2:0] > 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[2:0] in ADC_MR register.

- If ADMOD[1:0] = '00' and SEQCNT[2:0] > '000', 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 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[2:0] in ADC_MR register must be set to a number other than '0'.

- If SEQCNT[2:0] = '001' to '111', Multi-channel sampling in Sequential mode

Analog Input Channel in Sequential Mode:

The number of sequence channels that can be configured is set in the SEQCNT[2:0] in ADC_MR register.

- If SEQ0CH[4:0] = x, sequence's channel is adc_chx.
- If SEQ1CH[4:0] = x, sequence's channel is adc_chx.
- If SEQ2CH[4:0] = x, sequence's channel is adc_chx.
- If SEQ3CH[4:0] = x, sequence's channel is adc_chx.
- If SEQ4CH[4:0] = x, sequence's channel is adc_chx.
- If SEQ5CH[4:0] = x, sequence's channel is adc_chx.
- If SEQ6CH[4:0] = x, sequence's channel is adc_chx.
- If SEQ7CH[4:0] = x, sequence's channel is adc_chx.

Trigger Event Source Settings:

The trigger sources in each sequence can be configured is set in the SEQTRGx[3:0] in ADC_TRG register. In sequence mode, the 1st sequence has the highest priority, and the 8th sequence has the lowest priority.

- If SEQTRG0[3:0] = x, 1st channel's trigger source is x.
- If SEQTRG1[3:0] = x, 2nd channel's trigger source is x.
- If SEQTRG2[3:0] = x, 3rd channel's trigger source is x.
- If SEQTRG3[3:0] = x, 4th channel's trigger source is x.
- If SEQTRG4[3:0] = x, 5th channel's trigger source is x.
- If SEQTRG5[3:0] = x, 6th channel's trigger source is x.
- If SEQTRG6[3:0] = x, 7th channel's trigger source is x.
- If SEQTRG7[3:0] = x, 8th channel's trigger source is x.

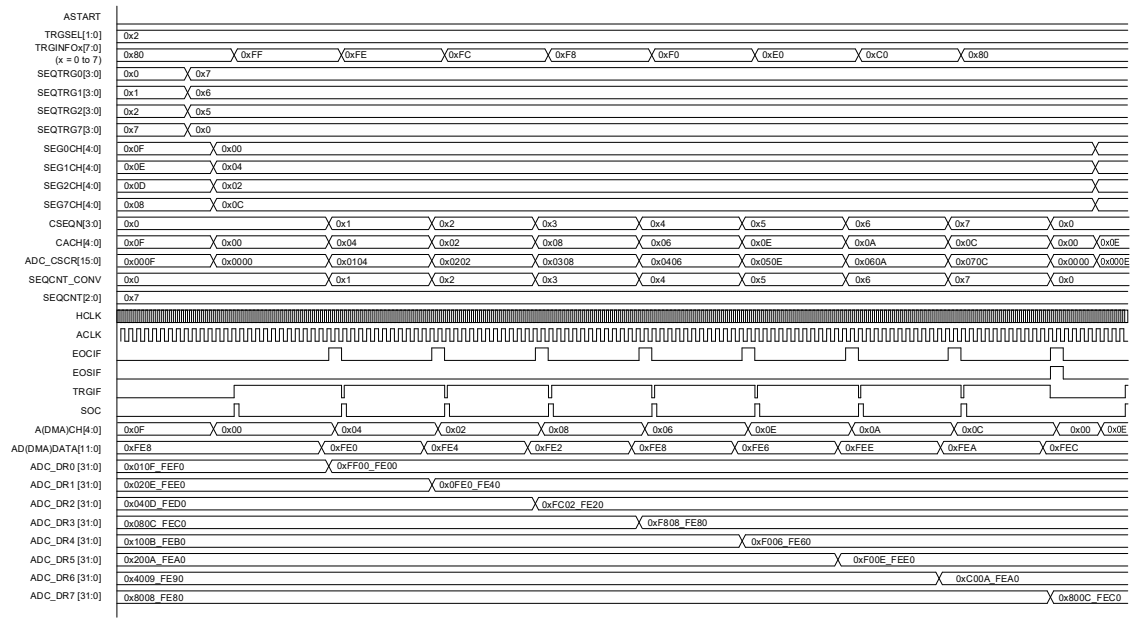
After the options for the ADC operation mode and trigger event sources, A/D conversion channel, and trigger event source are set, once the trigger event starts, sampling for the analog input starts.

Digitized data of the analog inputs is stored in the ADC_DRx register, and the EOCIF in ADC_SR register is set to '1'.

The EOCIF in ADC_SR register is generated as many as the number of sequences set by the SEQCNT[2:0] in ADC_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 ADC_SR register using software or read accessing the ADC_DRx register 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] trigger, followed by as many conversions as defined by the SEQCNT[2:0] in ADC_MR register.

Figure 140. ADC Sequential Mode Timing (ADMOD[1:0] = '00' and SEQCNT[2:0] = 7)



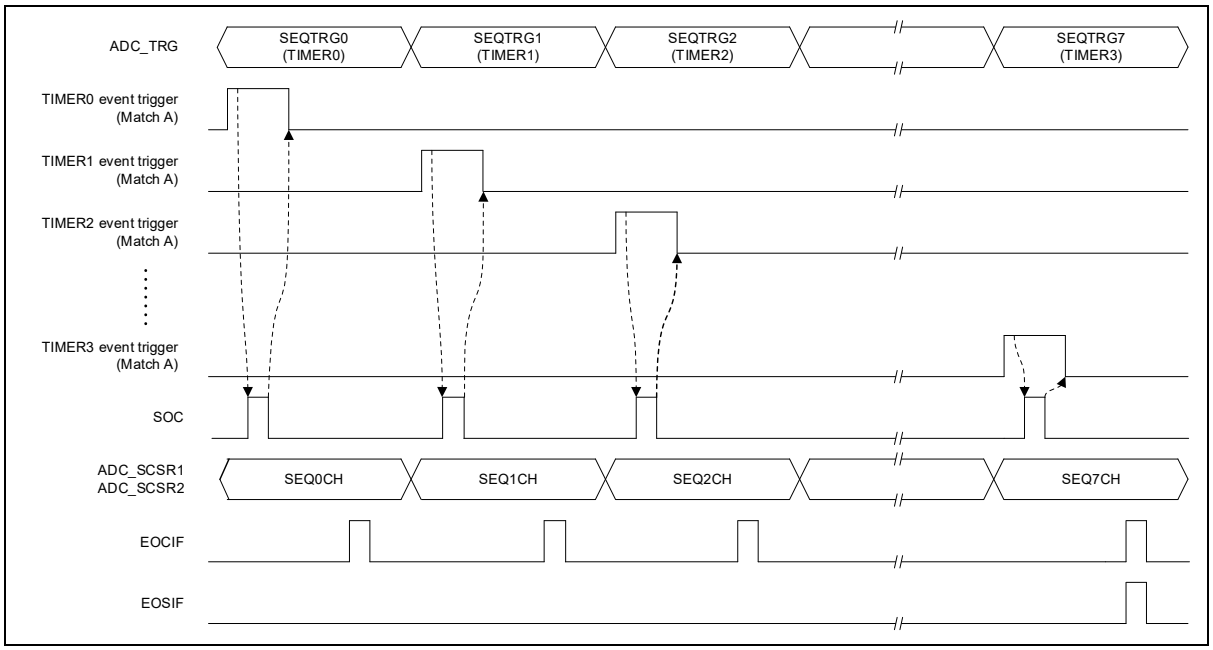
NOTE:

1. HCLK = 32 MHz, ACLK = 4.57 MHz (CLKDIV = 6)

Figure 141 shows a timing diagram that uses 8 sequences in Sequential mode, where each sequence is set to have TIMER trigger source. Based on the sequence' priority, the A/D conversion is sequentially executed for each trigger event.

When the trigger event of each channel occurs, the internal SOC signal is set to 'H' and becomes 'L' when the trigger event ends. Then the EOC flag is generated when the A/D conversion of the corresponding analog input signal is complete. The SOC and EOC occur as many times as set in the SEQCNT[2:0], and the sequence end flag EOS is generated when the EOC occurs during the last sequence.

Figure 141. ADC Trigger Timing in Sequential Mode (SEQCNT[2:0] = '111', 8-Sequential Conversion)



15.3.10 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

The SEQCNT[2:0] in ADC_MR Register in Burst Mode:

Since the A/D conversion is executed for multiple channels, the SEQCNT[2:0] in ADCn_MR register must be set to a number other than '0' in Burst mode.

- If SEQCNT[2:0] = '001' to '111', multi-channel sampling in Burst mode

Analog Input Channel in Burst Mode:

The number of sequence channels that can be configured is set in the SEQCNT[2:0] in ADCn_MR register. In Burst mode, when the trigger event occurs, the A/D conversions are performed in the order starting from SEQ0CH as shown below:

- If SEQ0CH[4:0] = x, 1st sequence's channel is adc_chx.
- If SEQ1CH[4:0] = x, 2nd sequence's channel is adc_chx.
- If SEQ2CH[4:0] = x, 3rd sequence's channel is adc_chx.
- If SEQ3CH[4:0] = x, 4th sequence's channel is adc_chx.
- If SEQ4CH[4:0] = x, 5th sequence's channel is adc_chx.
- If SEQ5CH[4:0] = x, 6th sequence's channel is adc_chx.
- If SEQ6CH[4:0] = x, 7th sequence's channel is adc_chx.
- If SEQ7CH[4:0] = x, 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 is triggered by the trigger source event assigned in the BSTTRG in ADC_TRG register.

If the TRGSEL[1:0] in ADC_MR register is set for the TIMER event triggers, SOC (Start of Conversion) signals are triggered by the BSTTRG in the ADC_TRG set trigger.

If the BSTTRG in ADC_TRG register is set for 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 ADC_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[2:0] in ADC_MR.

- If BSTTRG = x, 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.

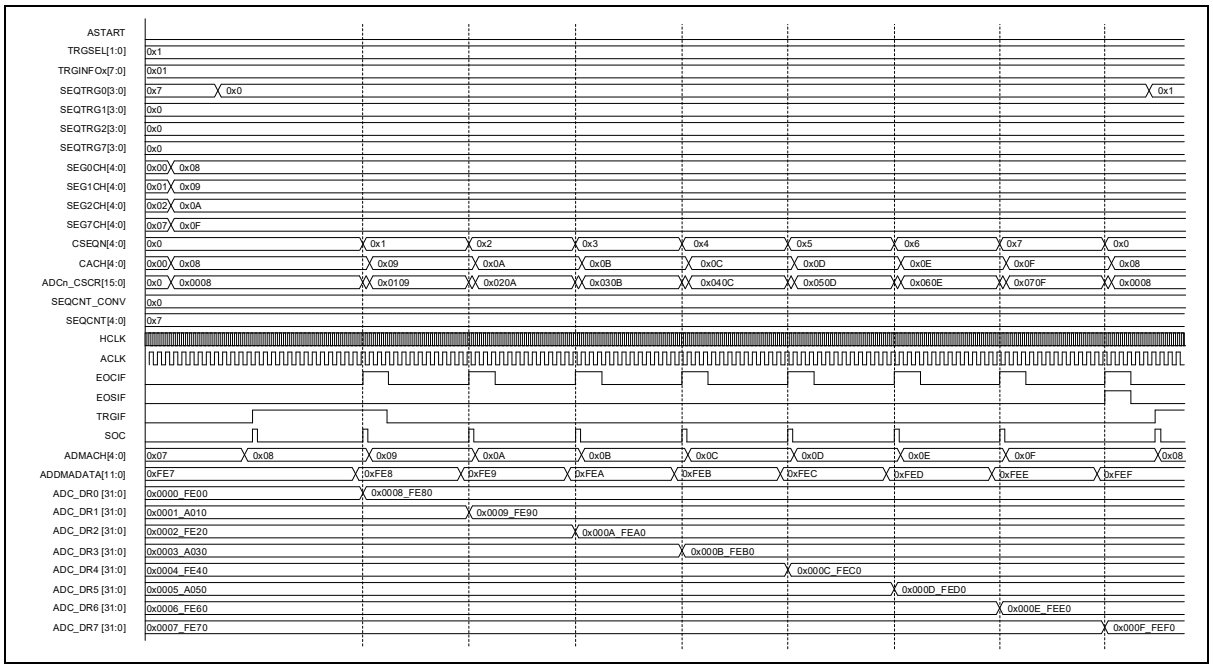
Digitized data of the analog inputs is stored in the ADC_DRx register (x = 0 to 7), and the EOCIF in ADC_SR register is set to ‘1’.

The EOCIF in ADC_SR register is generated as many times as set in the SEQCNT[2:0] in ADC_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 ADC_SR register using software or read accessing the ADC_DRx register (x = 0 to 7) of the corresponding sequence at the point when the EOSIF in ADC_SR register interrupt occurs.

In Burst timer trigger mode, if the ARST bit in ADC_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 ADC_CR register must be set to ‘1’.

Figure 142. ADC Burst Mode Timing (ADMOD[1:0] = '01')



15.3.11 Multiple Mode Conversion (ADMOD[1:0] = '10', SEQCNT[2:0] > 0)

In Multiple mode, the A/D conversions are performed in the order in which trigger events occur for each analog input. Since the A/D conversion starts for the analog inputs in the corresponding sequence at the point when the trigger event occurs, the EOS signal indicating end of the sequence is ignored.

- If ADMOD[1:0] = '10', A/D conversion in Multiple mode

There are two types of SOC trigger sources in Multiple mode:

- If TRGSEL[1:0] = '00', Software trigger event (ACLK)
- If TRGSEL[1:0] = '01', TIMERn trigger event

The SEQCNT[2:0] in ADC_MR register in Multiple mode:

Since the A/D conversion is executed for multiple channels, the SEQCNT[2:0] in ADC_MR register must be set to a number other than '0'.

- If SEQCNT[2:0] = '001' to '111', multi-channel sampling in Multiple mode

Trigger Event Source Settings:

The number of trigger sources in each sequence that can be configured is set in the SEQCNT[2:0] in ADC_MR register.

Users can set the desired trigger sources in the SEQTRGx[3:0] in ADC_TRG register. Each of these sources triggers an SOC when the triggering conditions are met, regardless of the sequence set in the SEQTRGx[3:0] in ADC_TRG register. For example, if the SEQCNT[2:0] in ADC_MR register is set to '11' and two 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, 1st channel's trigger source is x.
- If SEQTRG1[3:0] = x, 2nd channel's trigger source is x.
- If SEQTRG2[3:0] = x, 3rd channel's trigger source is x.
- If SEQTRG3[3:0] = x, 4th channel's trigger source is x.
- If SEQTRG4[3:0] = x, 5th channel's trigger source is x.
- If SEQTRG5[3:0] = x, 6th channel's trigger source is x.
- If SEQTRG6[3:0] = x, 7th channel's trigger source is x.
- If SEQTRG7[3:0] = x, 8th channel's trigger source is x.

NOTE:

1. 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 ADC_DRx register (x = 0 to 7), and the EOCIF in ADC_SR register is set to '1'.

The EOCIF ADC_SR register is generated as many as the number of sequences set in the SEQCNT[2:0], and the End of Sequence Interrupt Flag EOSIF in ADC_SR register has not occurred 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 ADC_SR register using software or read-accessing the ADC_DRx register (x = 0 to 7) of the corresponding sequence at the point when the EOCIF in ADC_SR register interrupt occurs.

15.3.12 Starting Conversions (ASTART, Start Trigger, ARST)

15.3.12.1 Start A/D Conversion (ASTART)

The A/D conversion starts when the ASTART bit in the ADC_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)

15.3.12.2 Reset A/D Conversion (ARST)

If the ARST in ADC_MR register is set to '0' and the software trigger is used as a trigger source, users must set the ASTART bit in the ADC_CR register to '1' using software to restart the A/D conversion at the end of the A/D conversion sequence.

If the A/D conversions are performed per sequence by the trigger source in Sequential mode, set the ARST bit in the ADC_MR register to '1' to restart the A/D conversion when the EOSIF flag in the ADC_SR register is generated at the end 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 has occurred after End-of-Sequence (EOS).

15.3.12.3 Start A/D Conversion by Trigger Source

If the ADC timer trigger source is used, software allows the A/D conversion to start when the timer trigger source occurs after timer operation. (Refer to section 15.3.15)

- The TEN is '1' in TIMERn_CR2
- The ASTART is '1' in ADC_MR

When the A/D conversion is complete for one channel, the EOCIF flag is generated and the ASTART bit in the ADC_CR register is cleared by hardware automatically.

In Sequential, Multiple, and Burst mode, the EOCIF flags are generated as many channels as set in a sequence, and the EOSIF flag is generated when sampling of the last channel is complete except Multiple mode.

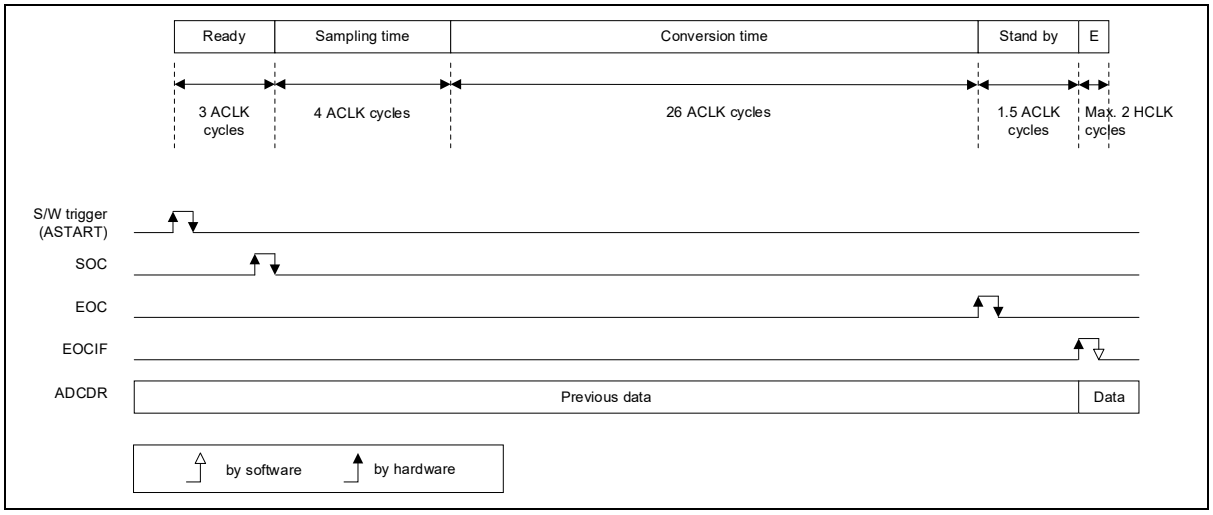
15.3.13 ADC Timing

The A/D conversion time consists of the following:

t_{ADC} = 3 ACLK cycles [Delay for sync until SOC] + 4 ACLK cycles [Sampling time]

+ 26 ACLK cycles [Conversion time] + 1.5 ACLK cycles (EOC) + Max. 2 HCLK cycles [EOCIF]

Figure 143. A/D Conversion Time



The total conversion time is calculated as follows:

Example)

When f_{HCLK} = 32 MHz, f_{ACLK} = 4.57 MHz

Then, t_{ADC} = {3 + 4 + 26 + 1.5} ACLK cycles + 2 HCLK = 34.5 ACLK + 2 HCLK = 7.6 μs

Table 118. ADC Frequency for Values of R_{AIN}.

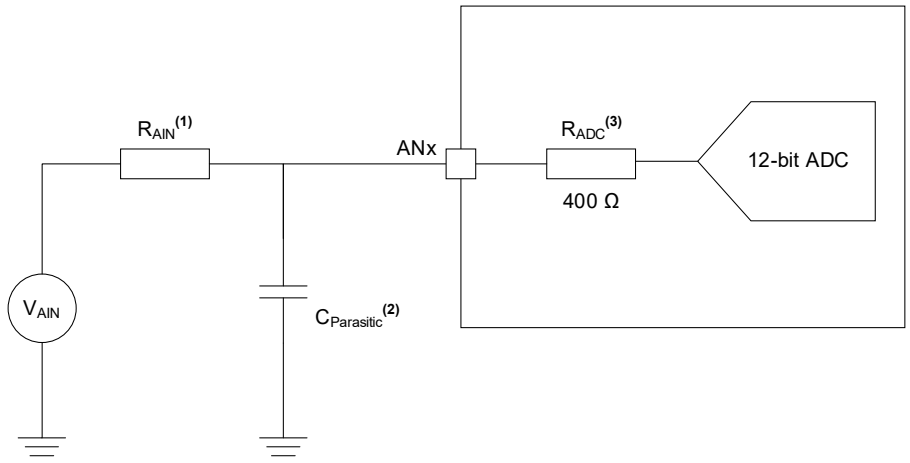
f_{ACLK} [MHz]	Ready		Sampling Time		Conversion Time		Stand by		End	t_{ADC} [μs]	f_{ADC} [ksps]	R_{AIN} Max. [kΩ]
	[Cycle]	[μs]	[Cycle]	[μs]	[Cycle]	[μs]	[Cycle]	[μs]	[μs]			
4.57	3	0.656	4	0.875	26	5.687	1.5	0.328	0.0625	7.6	131.4	6632.47
4		0.75		1		6.5		0.375		8.68	115.1	7700.85
3.2		0.937		1.25		8.125		0.468		10.84	92.2	9837.60
2		1.5		2		13		0.75		17.31	57.7	16247.86
1.6		1.875		2.5		16.25		0.937		21.62	46.2	20521.36
1		3		4		26		1.5		34.56	28.9	33341.88

NOTES:

1. The parameters are guaranteed by design.
2. See Figure 144 for how to design the RAIN resistor circuit for the ANx analog input.
3. The f_{ACLK} must not exceed 4.57 MHz.

The circuit of the analog input ANx for ADC to which the source is supplied can be designed as follows. When using I/O set to analog input mode, the ratio of the internal resistance R_{ADC} to the external resistance R_{AIN} has to be considered when inputting an external analog source.

Figure 144. Block Diagram of Typical Connection using ADC



NOTES:

1. Refer to Table 118 for more information about 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.

15.3.14 Stopping Ongoing Conversion (ASTOP)

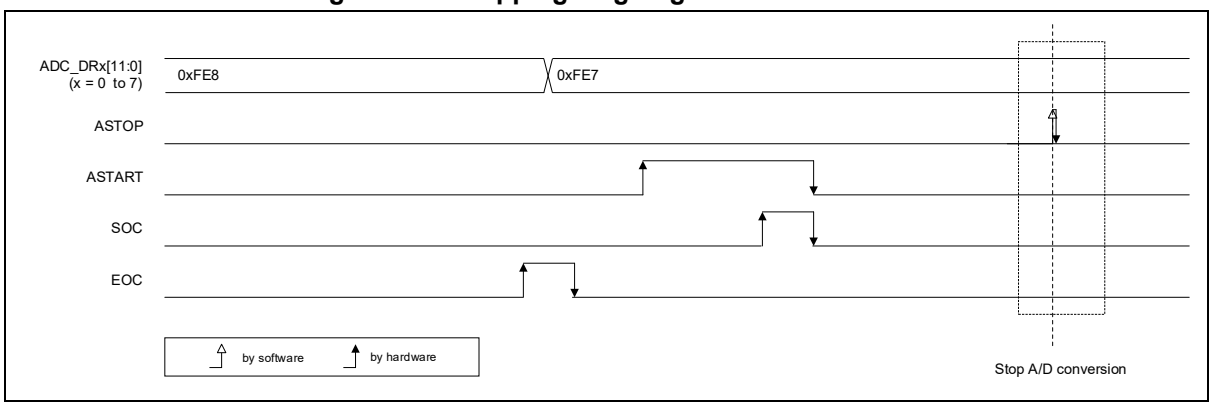
The A/D conversion in progress can be stopped by setting the ASTOP bit in the ADC_CR register to '1' using software. When the A/D conversion stops, the ADC operation in progress is reset. The ADC can be reconfigured (e.g. channel selection, trigger change) for new input data.

Once the ASTOP bit in the ADC_CR register is set to '1' by software, all A/D conversions in progress stop and partial results are deleted. (However, the ADC_DRx (x = 0 to 7) register is not updated due to this deletion.)

During the sequence conversion of the multi-channel, if the ASTOP in the ADC_CR register is set to '1', then the A/D conversion of a sequence stops.

When this procedure is completed, hardware clears the ASTOP in the ADC_CR register, and software needs to clear the EOCIF flag in the ADC_SR register.

Figure 145. Stopping Ongoing A/D Conversions



15.3.15 Conversion on External Trigger (TRGSEL[2:0], 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'.

15.3.15.1 ADC Trigger Source Selection (TRGSEL[2:0])

Users can select a trigger source by configuring the TRGSEL[2:0] in the ADC_MR register as described in Table 119.

Table 119. External Trigger Source Configuration

TRGSEL[2:0]	Trigger Source	Description
000	ASTART	Disables event triggers and enables the soft trigger only.
001	TIMERn	Enables the timer event trigger and soft trigger.
Others	-	Reserved

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'.

15.3.15.2 ADC Trigger Channel Settings (SEQTRGx[3:0] and TRGSRC[3:0])

Once a type of the trigger source is determined, up to 11 trigger sources (as a single trigger source or a sequence of trigger sources per ADC unit) can be selected by configuring the ADC_TRG register. If a trigger source for a single AD channel is used, users can use the SEQTRG0[3:0] in ADC_TRG register to select a trigger source.

For each sequence sampling in Sequential/Burst/Multiple mode, assign the corresponding trigger source for the A/D conversion channel using the SEQTRGx[3:0] in ADC_TRG. (x = 0 to 7)

By configuring the TRGSRC[3:0] bits in ADC_IMR register, users can select one from two types of trigger sources in Injection mode.

Table 120. Trigger Source Table

Trigger Source (SEQTRGx)	TIMER (TRGSEL[2:0] = '1')
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	TIMER 8
9	TIMER 9
10~14	Reserved
15	ASTART(1)

NOTE:

1. ASTART bit in ADC_CR register is a software trigger.

15.3.15.3 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 121. Start Control of AD Trigger Conversion

TRGSEL	Start Control Bit
ASTART	ASTART = '1' in ADC_CR register
TIMERn	TEN = '1' in TIMERn_CR2 register

15.3.15.4 ADC Trigger Interrupt and Flag (TRGIE, TRGIF)

Interrupts for the ADC trigger events can be enabled by configuring the TRGIE in 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.

15.3.16 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 ADC_DDR or ADC_DRx (x = 0 to 7) register to obtain the A/D conversion data.

For the use of interrupts, setting the EOCIE bit in the ADC_IER register to ‘1’ generates an interrupt when the EOCIF in ADC_SR register is set to ‘1’ after the A/D conversion is completed. Then read access the ADC_DDR or ADC_DRx (x = 0 to 7) register to obtain the A/D conversion data. The EOCIF flag in the ADC_SR register is cleared by writing ‘1’ to this bit using software.

15.3.17 End-of-Sequence Conversion (EOSIE, EOSIF)

The ADC module allows the sequence setup for the A/D conversion of analog inputs in multi-channels. A sequence can be configured with up to 8 channels using the SEQCNT[2:0] in ADC_MR register as shown below:

- If SEQCNT[2:0] = '000': 1 Sequence ADC (Single mode conversion)
- If SEQCNT[2:0] = '001': 2 Sequences ADC
- If SEQCNT[2:0] = '010': 3 Sequences ADC
- If SEQCNT[2:0] = '011': 4 Sequences ADC
- If SEQCNT[2:0] = '100': 5 Sequences ADC
- If SEQCNT[2:0] = '101': 6 Sequences ADC
- If SEQCNT[2:0] = '110': 7 Sequences ADC
- If SEQCNT[2:0] = '111': 8 Sequences ADC

After setting the number of channels, users can assign the channels that are performed in each sequence using the ADC_SCSR1 register (SEQ0CH to SEQ3CH) and the ADC_SCSR2 (SEQ4CH to SEQ7CH) register. Refer to section 15.3.7.

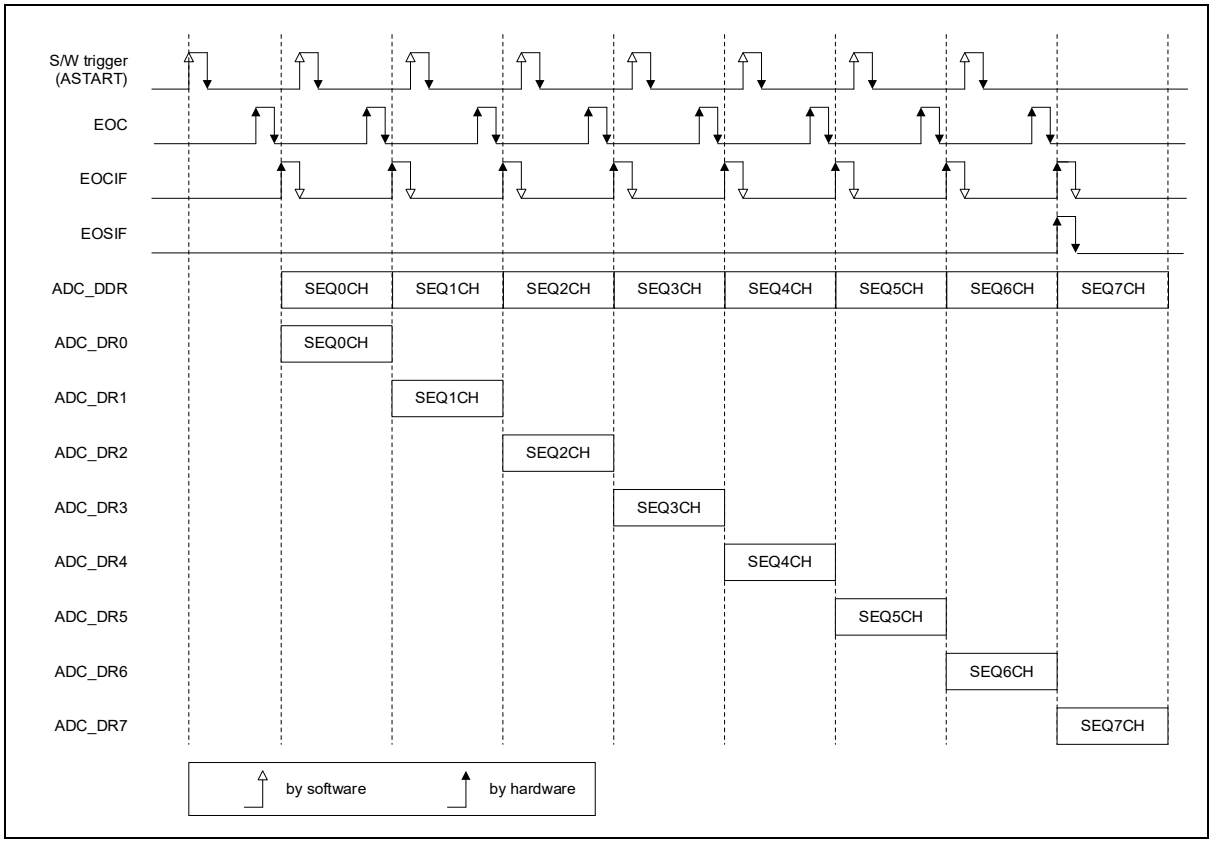
Sequence in Sequential mode is performed by following the ascending order of the sequence trigger source number set in the SEQTRGx[3:0] in ADC_TRG register. In Multiple mode, the sampling order is determined by the timing of each trigger source's event.

Users can adjust the trigger timing to set the sampling period for the ADC sequence. Refer to section 15.3.15.

For the ADC sequence conversion interrupt operation, enable the EOSIE in ADC_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 in ADC_SR register is set to '1'.

Figure 146 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 146. Example of A/D Conversion Timing in Sequence Mode



15.3.18 Data Management and Information

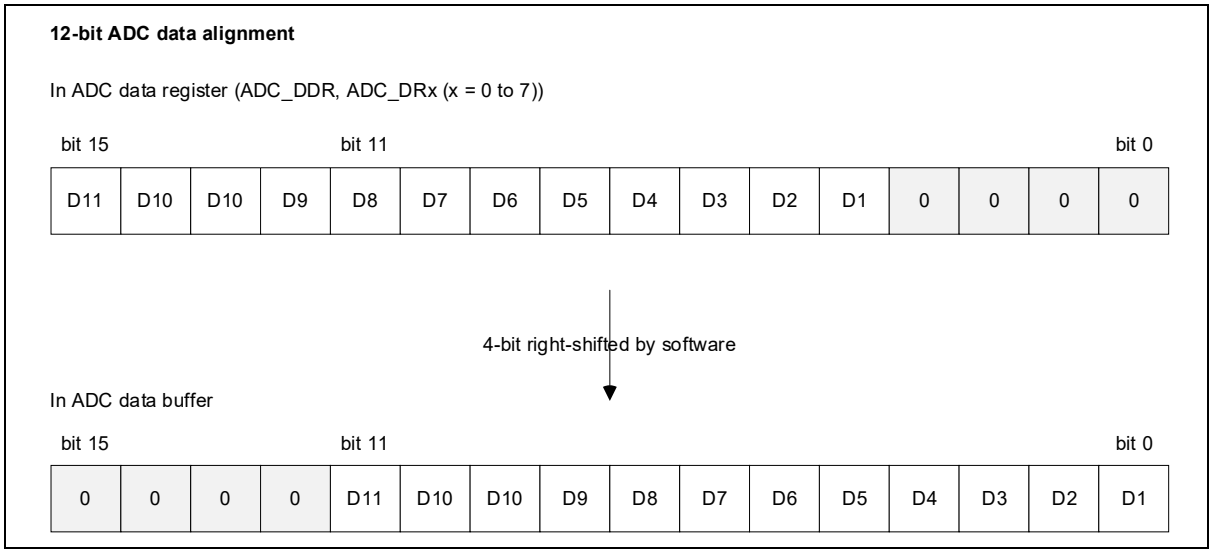
15.3.18.1 Data Register and Data Alignment by Software

At the EOC event of each A/D conversion channel, the result is stored in ADC data registers such as ADC_DDR, ADC_DR0, ADC_DR1, …, ADC_DR7.

- ADC_DDR: This register stores the AD data value of a channel where the A/D conversion was completed most recently.
- ADC_DRx: This register stores the A/D conversion result of a channel set in each sequence. (x = 0 to 7)

As shown in Figure 147, 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 147. Left Alignment (Unsigned Value)



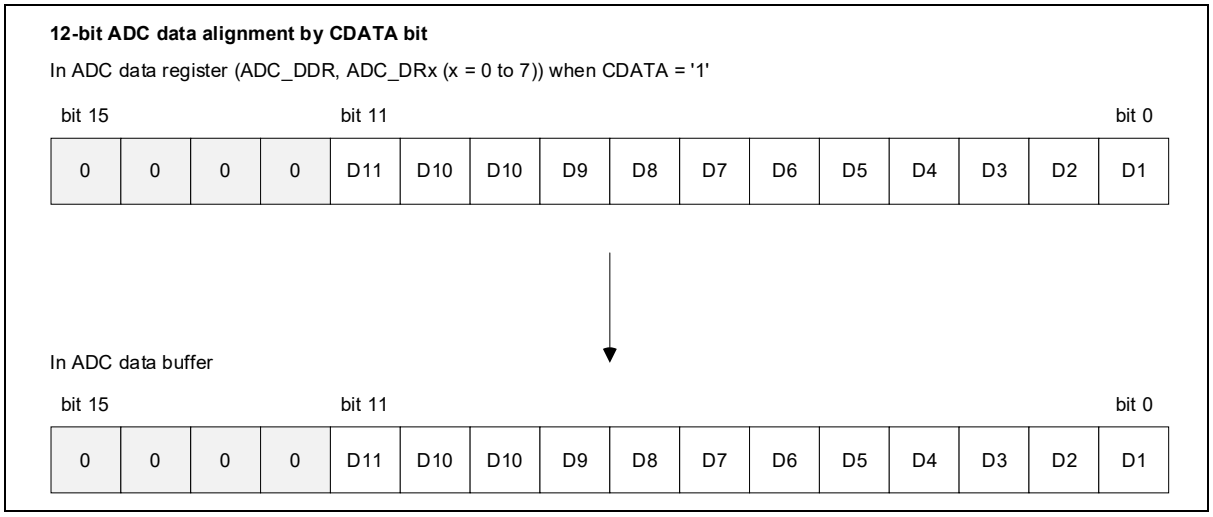
15.3.18.2 Conversion Data Offset Alignment

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 ADC_DRx (x = 0 to 7) 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 in the ADC_DDR and ADC_DRx (x = 0 to 7) registers.
- When the CDATA bit is 1, the A/D conversion value is stored in the [11:0] bit field in the ADC_DDR and ADC_DRx (x = 0 to 7) registers.

Figure 148. Right Alignment (Unsigned Value)



15.3.18.3 Channel and Trigger Information

Users can get information of the A/D conversion data channel by checking the upper 16 bits of the ADC data registers.

- The A/D conversion channel information to the ACH[4:0] in the ADC_DRx register.
- The sequence information for the channel where the A/D conversion is performed to the TRGINFOx in ADC_DRx register. (x = 0 to 7)

The TRGINFOx in ADC_DDR register (x = 0 to 7) shows that the lower the channel number set in a sequence, the higher the channel priority is executed first.

Figure 149 shows the configuration of the channel and sequence information in the ADC_DDR and ADC_DRx registers. As shown in the figure, the SEQxCH (x = 0 to 7) sequence is configured with 8 channels adc_ch0 to adc_ch7, allowing users to check the channel and trigger information in the in the ADC_DDR and ADC_DRx registers.

Figure 149. Channel and Trigger Source Information of A/D Conversion Sequence

	SEQ0CH	SEQ1CH	SEQ2CH	SEQ3CH	SEQ4CH	SEQ5CH	SEQ6CH	SEQ7CH
	adc_ch0	adc_ch1	adc_ch2	adc_ch3	adc_ch4	adc_ch5	adc_ch6	adc_ch7
ACH[4:0] (Channel number of AD conversion)	0x00	0x01	0x02	0x03	0x04	0x05	0x06	0x07
TRGINFO[4:0] (Trigger source priority of AD conversion)	0x00	0x01	0x02	0x03	0x04	0x05	0x06	0x07

15.3.19 Trigger Pending Function in Sequential Mode

In sequential mode of the ADC module, the SEQTRGT bit in ADC_MR determines how the ADC processes trigger events when multiple analog input channels are operated in sequential mode.

For example, when two analog input channels are operated in sequential mode, the method of processing trigger events is different depending on the value of the SEQTRGT bit.

- SEQTRGT = 0: when a second trigger event occurs while waiting for the first trigger event, the second trigger event is ignored, and 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.

15.3.20 ADC Low-Power Modes

Table 122. 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.

15.3.21 A/D Conversion using DMA

The ADC module sends a request for data transfer to the DMA module when the A/D conversion of the analog inputs is completed and the EOCIF flag in the ADC_SR register is set to '1'. The DMA module transfers data to the specified memory area upon receiving this request.

To use the A/D conversion functionality, enable the ADC and DMA using the SCU and then activate the DMAEN bit in the DMAn_SR register to '1' with the ADEN in ADC_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[3:0] in the DMA_CR register to select the ADC, referring to the number below:

- PERISEL[3:0] = '1011' (0xB, 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 5). For example, if users want to transfer 64 A/D conversion data values to the DMA, the DMA data width is set to '1'6-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 is 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 (0x4100_5060, 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 in ADC_IER.

- If DMAIE = ‘0’, Disable ADC DMA-done interrupt
- If DMAIE = ‘1’, Enable ADC DMA-done interrupt

When the DMAEN bit in the DMAn_SR is set to ‘1’, the DMA is enabled, and the A/D conversion results are transferred to the address specified by the DMA controller using software. The value in the TRANSCNT[11:0] in the DMAn_CR register is decreased by one each time a DMA transfer of a specified size is completed.

When the DMA transfer is completed by the data size requested to the DMA, and the TRANSCNT[11:0] value 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 flag in the DMAn_SR register of the DMA module is enabled when the transmission is complete and the TRANSCNT[11:0] value becomes 0. The EOT flag is cleared by writing a number larger than 0 to the TRANSCNT[11:0] in DMAn_CR register.

The A/D conversion results before the generation of the EOCIF can be referenced only by accessing the ADC_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.

15.3.22 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 flag in the ADC_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 DMAn_CR register value becomes ‘0’, the EOT flag in the DMAn_SR register signal is sent to the ADC module to inform that all DMA data transmissions are complete.

When the DMA does not complete the transmission yet and cannot send the ACK signal to the ADC, if it receives another transfer request from the ADC, the DMA overrun flag (DMAOF) in the ADC_SR register) is set to ‘1’. The DMAOF does not support interrupt function.

If the overrun flag is generated because the DMA cannot process the DMA transfer requests on time, the ADC stops requesting a 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 flag in the ADC_SR register.

Even if the overrun flag is generated, the ADC maintains operating. Unless software sets the ASTOP bit in the ADC_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 ADC_DDR register until the EOCIF flag in the ADC_SR register is generated.

15.3.23 Comparison of ADC Sensing Value

The ADC has a built-in comparator that compares the sampling results of the AD inputs and software set values.

Users can compare the value set by CVAL[11:0] with the value A/D conversion channel specified by CCH[4:0] based on the VDD reference voltage.

Set the CCH[4:0] in ADC_CMPxR register to select a target ADC channel for comparison and enter the comparison value in the CVAL[11:0] in ADC_CMPxR register

- If CMPEN = 0, ADC comparator operation is disabled.
- If CMPEN = 1, ADC comparator operation is enabled.

Figure 150. Block Diagram of ADC Internal Comparator

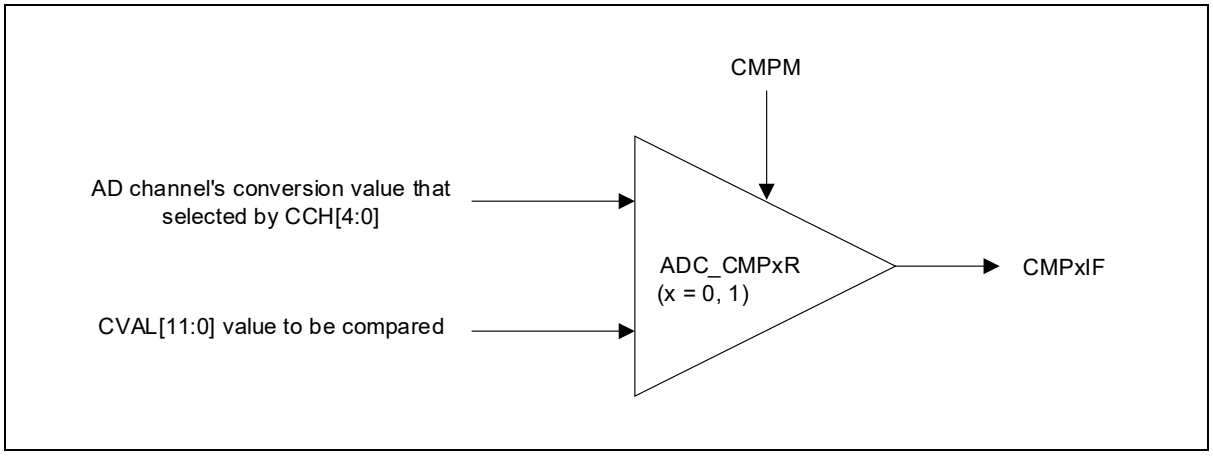


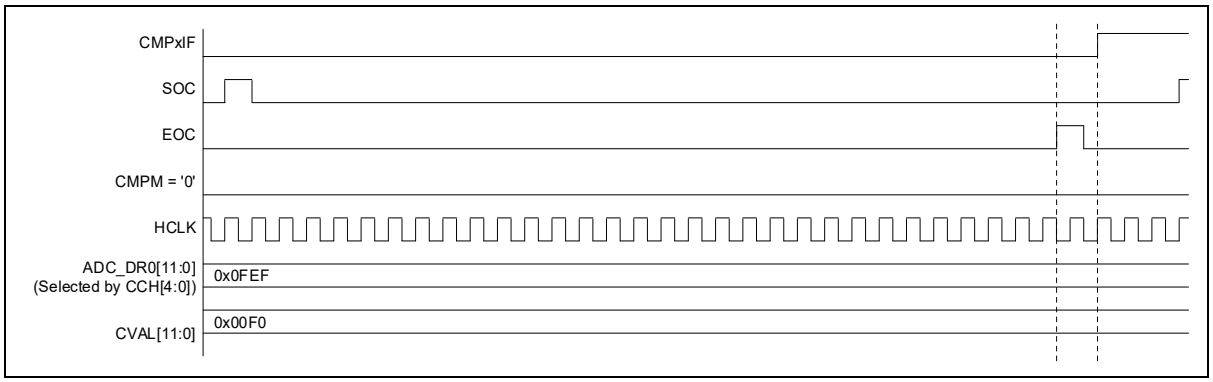
Table 123 shows the conditions under which the CMPxIF flag occurs by comparing the CVAL[11:0] and the ADC channel values set in CCH[4:0] according to the CMPM settings.

Table 123. 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 in ADC_CR register is set to ‘1’, when the EOCIF in ADC_SR register indicating the A/D conversion completion is set to ‘1’, two input signals are compared internally and the CMPxIF in ADC_SR register indicating the comparison completion is generated (CMPxIF = 1)

Figure 151. Timing Diagram of Comparison between A/D Conversion Results and Reference Value



The comparison results based on the CMPM in ADC_CMPxR register value affect the flag setting condition of the CMPxIF in ADC_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 ADC_CMPxR register to ‘1’ as shown below:

- If CMPIEN = 0, ADC compare interrupt is disabled.
- If CMPM = 1, ADC compare interrupt is enabled.

15.4 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 (CMPxIE) (x = 0, 1)

Each interrupt enable bits are available for flexibility.

Table 124. Control Bits and Event Flags for ADC Interrupts

Interrupt Event	Event Flag	Enable Control Bit
End of conversion of single data	EOCIF	EOCIE
End of sequence conversion	EOSIF	EOSIE
End of conversion by timer trigger source	TRGIF	TRGIE
End of DMA transfer of ADC conversion data	DMAF	DMAIE
ADC conversion data comparison	CMPxIF	CMPxIE

15.5 ADC Registers

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

Table 125. Base Address of ADC

Name	Base address
ADC	0x4009_2000

Table 126. ADC Register Map

Name	Offset	Type	Description	Reset Value	Ref.
ADC_MR	0x0000	RW	ADC Mode Register	0x0000_0000	15.5.1
ADC_CSCR	0x0004	RW	ADC Current Sequence/Channel Register	0x0000_0000	15.5.2
ADC_CCR	0x0008	RW	ADC Clock Control Register	0x0000_0000	15.5.3
ADC_TRG	0x000C	RW	ADC Trigger Select Register	0x0000_0000	15.5.4
ADC_SCSR1	0x0018	RW	ADC Channel Select 1 Register	0x0000_0000	15.5.5
ADC_SCSR2	0x001C	RW	ADC Channel Select 2 Register	0x0000_0000	15.5.6
ADC_CR	0x0020	RW	ADC Control Register	0x0000_0000	15.5.7
ADC_SR	0x0024	RW	ADC Status Register	0x0000_0000	15.5.8
ADC_IER	0x0028	RW	ADC Interrupt Enable Register	0x0000_0000	15.5.9
ADC_DDR	0x002C	RO	ADC DMA Data Register	0x0000_0000	15.5.10
ADC_DR0	0x0030	RO	ADC Sequence Data 0 Register	0x0000_0000	15.5.11
ADC_DR1	0x0034	RO	ADC Sequence Data 1 Register	0x0000_0000	15.5.11
ADC_DR2	0x0038	RO	ADC Sequence Data 2 Register	0x0000_0000	15.5.11
ADC_DR3	0x003C	RO	ADC Sequence Data 3 Register	0x0000_0000	15.5.11
ADC_DR4	0x0040	RO	ADC Sequence Data 4 Register	0x0000_0000	15.5.11
ADC_DR5	0x0044	RO	ADC Sequence Data 5 Register	0x0000_0000	15.5.11
ADC_DR6	0x0048	RO	ADC Sequence Data 6 Register	0x0000_0000	15.5.11
ADC_DR7	0x004C	RO	ADC Sequence Data 7 Register	0x0000_0000	15.5.11
ADC_CMP0R	0x0060	RW	ADC channel compare 0 register	0x0000_0000	15.5.12
ADC_CMP1R	0x0064	RW	ADC channel compare 1 register	0x0000_0000	15.5.13

15.5.1 ADC_MR: ADC Mode Register

ADC_MR configures the modes of the ADC module. Users should set this register prior to all other ADC registers according to the intended use of the ADC module.

ADC_MR=0x4009_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	SEL_REF	Reserved		CDATA	Reserved	DMAEN		Reserved	SEQTRGT	SEQCNT[2:0]	ADEN	ARST	ADMOD[1:0]	Reserved	TRGSEL[2:0]
-	0	-		0	-	0		-	0	000	0	0	00	-	000
-	RW	-		RW	-	RW		-	RW	RW	RW	RW	RW	-	RW

27	SEL_REF	ADC SEL_REF control	
		0	Disable (VDD, analog input range (max))
		1	Enable (AVREF, analog input range(max))
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 0)	
		0	Trigger that is not related to current count in sequential is pending.
		1	Trigger that is not related to current count in sequential is not pending.
10 8	SEQCNT[2:0]	Number of conversions in a sequence (a maximum of 8) Sequence = SEQCNT + 1 The setting of the SEQCNT is valid only in 0~7. If ADMOD[1:0] is '0' and SEQCNT[2:0] is not '0', the CSEQN value increments to the SEQCNT[2:0] value by a trigger event. (The setting of the bit field is valid only in single/Sequential modes.)	
		000	1-sequence ADC
		001	2-sequence ADC
		010	3-sequence ADC
		011	4-sequence ADC
		100	5-sequence ADC
		101	6-sequence ADC
		110	7-sequence ADC
		111	8-sequence ADC
7	ADEN	ADC enable bit	
		0	Disable
		1	Enable
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]	ADC mode selection bit	
		00	Single or Sequential conversion mode
		01	Burst conversion mode
		10	Multiple conversion mode
		11	Reserved
2 0	TRGSEL[2:0]	Trigger selection bit	
		000	Event trigger disable/Software trigger only
		001	Timer event trigger/Software trigger
		others	Reserved

NOTES:

1. If ADMOD[1:0] has been set for burst conversion mode, the ADC channels are assigned as BST0CH through BST7CH. Burst mode always begins with BST0CH. (In three-burst mode, for example, the analog inputs to the BST0CH, BST1CH, and BST2CH channels are converted in the order of channel number.)
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.

15.5.2 ADC_CSCR: ADC Current 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, users must set the ADEN in ADC_MR register.

ADC_CSCR=0x4009_2004

31 30 29 28 27 26 25 24	23 22 21 20 19 18 17 16	15 14 13 12 11 10 9 8	7 6 5 4 3 2 1 0
Reserved		CSEQN[2:0]	Reserved	CACH[4:0]
-		000	-	00000
-		R0	-	R0

10 8	CSEQN[2:0]	Current sequence number	
		000	Current sequence number is 0.
		001	Current sequence number is 1.
		010	Current sequence number is 2.
		011	Current sequence number is 3.
		100	Current sequence number is 4.
		101	Current sequence number is 5.
		110	Current sequence number is 6.
		111	Current sequence number is 7.
4 0	CACH[4:0]	Current active channel	
		00000	ADC channel 0 is active.
		00001	ADC channel 1 is active.
		00010	ADC channel 2 is active.
		00011	ADC channel 3 is active.
		00100	ADC channel 4 is active.
		00101	ADC channel 5 is active.
		00110	ADC channel 6 is active.
		00111	ADC channel 7 is active.
		01000	ADC channel 8 is active.
		01001	ADC channel 9 is active.
		01010	ADC channel 10 is active.
		01011	ADC channel 11 is active.
		01100	ADC channel 12 is active.
		01101	ADC channel 13 is active.
		01110	ADC channel 14 is active.
		01111	ADC channel 15 is active.
		10000	ADC channel 16 is active.
		10001	ADC channel 17 is active.
		10011	V_{CORE}
		Others	Reserved

15.5.3 ADC_CCR: ADC Clock Control Register

The ADC_CCR register controls the clock of the ADC module. This register is 16 bits wide.

ADC_CCR=0x4009_2008

31 30 29 28 27 26 25 24	23 22 21 20 19 18 17 16	15 14 13 12 11 10 9 8	7 6 5 4 3 2 1 0
Reserved		CLKDIV[6:0]	Reserved
-		0000000	-
-		RW	-

14 8	CLKDIV[6:0]	ADC clock division ratio value bit CLKDIV = 0, ADC clock = HCLK (bypass) CLKDIV ≥ 1, ADC clock = HCLK / (CLKDIV[6:0] + 1) The ADC clock's frequency must not exceed 4.57 MHz.

15.5.4 ADC_TRG: ADC Trigger Select Register

The ADC_TRG register selects trigger sources for the ADC module.

ADC_TRG=0x4009_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
SEQTRG7[3:0]	SEQTRG6[3:0]	SEQTRG5[3:0]	SEQTRG4[3:0]	SEQTRG3[3:0]	SEQTRG2[3:0]	SEQTRG1[3:0]	SEQTRG0[3:0] BSTTRG[3:0]
0000	0000	0000	0000	0000	0000	0000	0000
RW	RW	RW	RW	RW	RW	RW	RW

30 28	SEQTRG7[3:0]	8th sequence trigger source
27 24	SEQTRG6[3:0]	7th sequence trigger source
23 20	SEQTRG5[3:0]	6th sequence trigger source
19 16	SEQTRG4[3:0]	5th sequence trigger source
15 12	SEQTRG3[3:0]	4th sequence trigger source
11 8	SEQTRG2[3:0]	3rd sequence trigger source
7 4	SEQTRG1[3:0]	2nd sequence trigger source
3 0	SEQTRG0[3:0] BSTTRG[3:0]	1st sequence trigger source Burst conversion trigger source

NOTE:

1. In Multiple mode, the 1st sequence trigger source is given the highest priority, and the 8th sequence trigger source is given the lowest priority. Table 120 shows trigger sources represented by each value. (Refer to section 15.3.15)

15.5.5 ADC_SCSR1: ADC Channel Select 1 Register

ADC_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 ADC_TRG register. To write to this register, users must first enable the ADEN in ADC_MR register.

ADC_SCSR1=0x4009_2018

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

15.5.6 ADC_SCSR2: ADC Channel Select 2 Register

ADC_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 ADC_TRG register. To write to this register, users must first enable the ADEN in ADC_MR register.

ADC_SCSR2=0x4009_201C

31 30 29 28 27 26 25 24	23 22 21 20 19 18 17 16	15 14 13 12 11 10 9 8	7 6 5 4 3 2 1 0
Reserved / SEQ7CH[4:0]	Reserved / SEQ6CH[4:0]	Reserved / SEQ5CH[4:0]	Reserved / SEQ4CH[4:0]
- 00000	- 00000	- 00000	- 00000
- RW	. RW	. RW	. RW

28 24	SEQ7CH [4:0]	8th conversion sequence channel selection
20 16	SEQ6CH [4:0]	7th conversion sequence channel selection
12 8	SEQ5CH [4:0]	6th conversion sequence channel selection
4 0	SEQ4CH[4:0](1)	5th conversion sequence channel selection

15.5.7 ADC_CR: ADC Control Register

ADC_CR is used to control the operation of ADC module.

ADC_CR=0x4009_2020

31 30 29 28 27 26 25 24	23 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	TRGCLR	ASTART
-			0	-	0	0
-			WO	-	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.
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.

15.5.8 ADC_SR: ADC Status Register

The ADC_SR register displays the status of the ADC.

ADC_SR=0x4009_2024

31	30 29 28 27 26 25 24	23 22 21 20 19 18 17 16	15 14 13 12 11 10	9	8	7 6	5	4	3	2	1	0
ABUSY	Reserved			CMP1IF	CMP0IF	Reserved	DOVRUN	DMAF	TRGIF	EOSIF	Reserved	EOCIF
0	-			0	0	-	0	0	0	0	-	0
RO	-			RWC1	RWC1	-	RO	RO	RWC1	RWC1	-	RWC1

31	ABUSY	ADC busy status
		0 Not busy
		1 Busy
9	CMP1IF	Compare 1 interrupt flag bit
		0 Interrupt is not generated.
		1 Interrupt is generated (writing ‘1’ clears the flag).
8	CMP0IF	Compare 0 interrupt flag bit
		0 Interrupt is not generated.
		1 Interrupt is generated (writing ‘1’ clears the flag).
5	DOVRUN	DMA overrun flag
		0 Flag is not detected.
		1 Flag is detected.
4	DMAF	DMA done received flag (DMA transfer is completed.)
		0 Flag is not detected.
		1 Flag is detected.
3	TRGIF	ADC trigger interrupt flag
		0 Flag is not detected.
		1 Flag is detected (writing ‘1’ clears the flag).
2	EOSIF	Sequence end interrupt flag
		0 Flag is not detected.
		1 Flag is detected (writing ‘1’ clears the flag).
0	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, users must use the EOCIF bit in ADC0_SR register.

```
Example)
while ( (ADC_SR & 0x01) == 0 )  // EOCIF flag checking
ADC_SR = 0x1;                   // EOCIF flag clear
…
```

15.5.9 ADC_IER: ADC Interrupt Enable Register

The ADC_IER register determines whether to enable the ADC interrupts.

ADC_IER=0x4009_2028

31 30 29 28 27 26 25 24	23 22 21 20 19 18 17 16	15 14 13 12 11 10	9	8	7 6	5	4	3	2	1	0
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	EOCIE	ADC single conversion interrupt
		0 Disables.
		1 Enables.

NOTE:

1. Burst mode sets the EOSIE = 1 and check the EOSIF in ADC_SR register.

15.5.10 ADC_DDR: ADC DMA Data Register

ADC_DDR manages the ADC module's DMA data. It displays the results of the ADC module's DMA conversions.

ADC_DDR=0x4009_202C

31	30	29	28	27	26	25	24	23 22 21	20 19 18 17 16	15 14 13 12 11 10 9 8	7 6 5 4 3 2 1 0
TRGINFO7	TRGINFO6	TRGINFO5	TRGINFO4	TRGINFO3	TRGINFO2	TRGINFO1	TRGINFO0	Reserved	ADMACH[4:0]	ADDMAR[11:0] ADC DMA temporary data	
0	0	0	0	0	0	0	0	-	0x00	0x0000	
R0	R0	R0	R0	R0	R0	R0	R0	-	R0	R0	

31 24	TRGINFOx (x= 0 to 7)	ADC sequential number information Indicates whether each trigger source has been captured until the end of conversion (EOC). Users can find out which trigger source is being currently processed by referring to the CSEQN[4:0] in ADC_CSCR register.
20 16	ADMACH[4:0]	DMA ADC channel indicator
15 0	ADDMAR[15:0]	DMA ADC conversion result data (12-bit)

NOTES:

1. Even when DMA is inactive, data is temporarily stored in this register before stored in the designated buffer. Additionally, the register also displays which channel the data belongs to.
2. TRGINFOx indicates the flag state in which a trigger occurred, and the corresponding trigger is cleared by matching the SOC after the sequence event is processed. However, in burst mode, it is masked as TRGINFO0 = 1 and other TRGINFOx bits are cleared.

15.5.11 ADC_DRx: ADC n Data 0 to 7 Register

The ADC_DR0 register displays the results of ADC conversions. There are eight of these registers.

ADC_DR0=0x4009_2030, ADC_DR1=0x4009_2034, ADC_DR2=0x4009_2038, ADC_DR3=0x4009_203C
ADC_DR4=0x4009_2040, ADC_DR5=0x4009_2044, ADC_DR6=0x4009_2048, ADC_DR7=0x4009_204C

31	30	29	28	27	26	25	24	23 22 21 20 19 18 17 16	15 14 13 12 11 10 9 8	7 6 5 4 3 2 1 0
TRGINFO7	TRGINFO6	TRGINFO5	TRGINFO4	TRGINFO3	TRGINFO2	TRGINFO1	TRGINFO0	Reserved	ACH[4:0]	ADDATA[15:0]
0	0	0	0	0	0	0	0	-	0x00	0x0000
R0	R0	R0	R0	R0	R0	R0	R0	-	R0	R0

31 24	TRGINFOx (x= 0 to 7)	ADC sequential number information Indicates whether each trigger source has been captured until the end of conversion (EOC). Users can find out which trigger source is being currently processed by referring to the CSEQN[4:0] in ADC_CSCR register.
20 16	ACH[4:0]	ADC channel indicator
15 0	ADDATA[11:0]	ADC conversion result data (12-bit)

NOTE:

1. TRGINFOx indicates the flag state in which a trigger occurred, and the corresponding trigger is cleared by matching the SOC after the sequence event is processed. However, in burst mode, it is masked as TRGINFO0 = 1 and other TRGINFOx bits are cleared.

15.5.12 ADC_CMP0R: ADC Channel Comparison 0 Register

The ADC_CMP0R register controls the comparison between the ADC channels.

ADC_CMP0R=0x4009_2060

31 30 29 28 27 26 25 24	23	22	21	20 19 18 17 16	15 14 13 12 11 10 9 8	7 6 5 4 3 2 1 0
MATCH_CNT[5:0]	Reserved	CMPEN	Reserved	CMPM	CCH[4:0]	CVAL[11:0]	Reserved
000000	-	0	-	0	00000	0x000	-
RW	-	RW	-	RW	RW	RW	-

31 26	MATCH_CNT[5:0]	Interrupt is generated only when ADC data compare data meets the condition (MATCHCNT + 1) consecutively.	
23	CMPEN	Compare operation enable bit	
		0	Disables.
		1	Enables.
21	CMPM	ADC compare mode bit	
		0	ADC value is 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
		01100	Compare channel is ADC channel 12
		01101	Compare channel is ADC channel 13
		01110	Compare channel is ADC channel 14
		01111	Compare channel is ADC channel 15
		10000	Compare channel is ADC channel 16
		10001	Compare channel is ADC channel 17
		10011	V_{CORE}
		others	Reserved
15 4	CVAL[11:0]	Compare value bit	

15.5.13 ADC_CMP1R: ADC Channel Comparison 1 Register

The ADC_CMP1R register controls the comparison between the ADC channels.

ADC_CMP0R=0x4009_2064

31 30 29 28 27 26 25 24	23	22	21	20 19 18 17 16	15 14 13 12 11 10 9 8	7 6 5 4 3 2 1 0
MATCH_CNT[5:0]	CMPEN	Reserved	CMPM	CCH[4:0]	CVAL[11:0]	Reserved
000000	0	-	0	00000	0x000	-
RW	RW	-	RW	RW	RW	-

31 26	MATCH_CNT[5:0]	Interrupt is generated only when ADC data compare data meets the condition (MATCHCNT + 1) consecutively.	
23	CMPEN	Channel sampling time	
		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
		01100	Compare channel is ADC channel 12
		01101	Compare channel is ADC channel 13
		01110	Compare channel is ADC channel 14
		01111	Compare channel is ADC channel 15
		01000	Compare channel is ADC channel 16
		01001	Compare channel is ADC channel 17
		10011	V_{CORE}
		others	Reserved
15 4	CVAL[11:0]	Compare value bit	

15.5.14 ADC Register Map Summary

Table 127. 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	ADC_MR	Res.	Res.	Res.	Res.	SEL_REF	Res.	Res.	Res.	Res.	Res.	Res.	Res.	CDATA	Res.	DMAEN	Res.	Res.	Res.	Res.	Res.	SEQTRGT	SEQCNT[2:0]			ADEN	ARST	ADMOD[1:0]		Res.	TRGSEL[2:0]		
	Reset value					0								0		0						0	0	0	0	0	0	0	0		0	0	0
0x04	ADC_CSCR	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	CSEQN[2:0]			Res.	Res.	Res.	CACH[4:0]				
	Reset value																						0	0	0				0	0	0	0	0
0x08	ADC_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	ADC_TRG	SEQTRG7[3:0]				SEQTRG6[3:0]				SEQTRG5[3:0]				SEQTRG4[3:0]				SEQTRG3[3:0]				SEQTRG2[3:0]				SEQTRG1[3:0]				SEQTRG0[3:0] BSTTRG[3:0]			
	Reset value	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
0x38	ADC_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	ADC_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
0x54	ADC_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.	Res.	Res.	Res.	TRGCLR	ASTART
	Reset value																									0						0	0

Table 127. 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
0x58	ADC_SR	ABUSY	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	CMP1IF	CMP0IF	Res.	Res.	DOVRUN	DMAF	TRGIF	EOSIF	Res.	EOCIF
	Reset value	0																						0	0			0	0	0	0		0
0x5C	ADC_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	ADC_DDR	TRGINFO7	TRGINFO6	TRGINFO5	TRGINFO4	TRGINFO3	TRGINFO2	TRGINFO1	TRGINFO0	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	0	0	0
0x64- 0xC8	ADC_DRx	TRGINFO7	TRGINFO6	TRGINFO5	TRGINFO4	TRGINFO3	TRGINFO2	TRGINFO1	TRGINFO0	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	0	0	0
0xE0	ADC_CMP0R	MATCH_CNT[5:0]						Res.	Res.	CMPEN	Res.	CMPM	CCH[4:0]					CVAL[11:0]												Res.	Res.	Res.	Res.
	Reset value	0	0	0	0	0	0			0		0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0				
0xE4	ADC_CMP1R	MATCH_CNT[5:0]						Res.	Res.	CMPEN	Res.	CMPM	CCH[4:0]					CVAL[11:0]												Res.	Res.	Res.	Res.
	Reset value	0	0	0	0	0	0			0		0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0				

16. Comparator

16.1 Comparator Introduction

This section applies to A31C14x devices only.

A31C14x has two comparators. The comparator outputs the comparison result of two analog voltages via an I/O pin or generates interrupts.

Users can control the comparators by configuring the CMP_CONF register. The HYSSEL[1:0] in CMP_CONF register is used to select the comparator hysteresis voltage, and the HYSEN enables the hysteresis.

16.2 Comparator Main Features

The comparators of the A31C14x have key features as listed below.

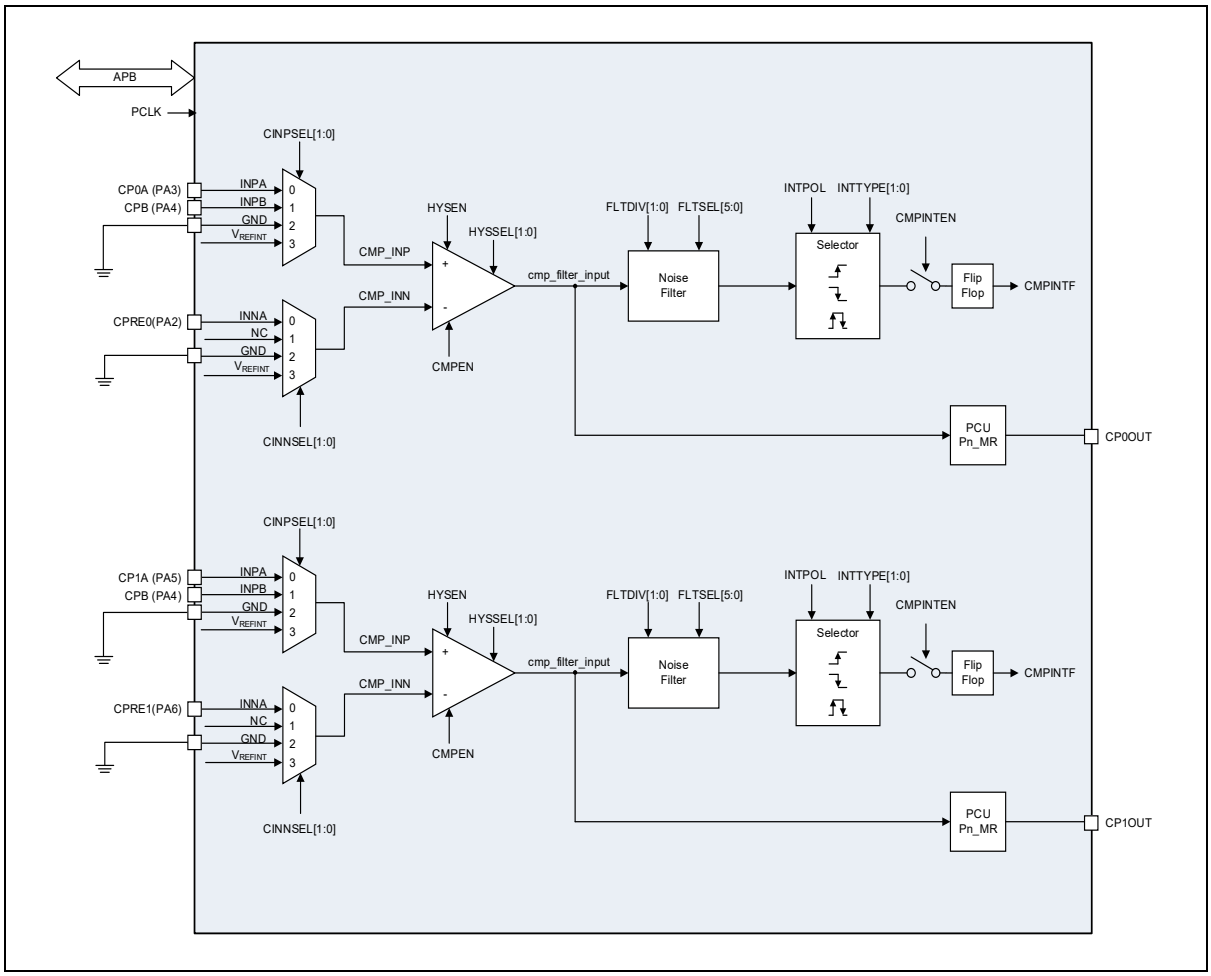
- Two built-in comparators
- Comparator has two input sources (CPn (n=A, B), CPRE/Internal VDD 1.0 V)
- The selection between falling and rising of the interrupt signal outputs is available.
- Noise filtering capability
- Four hysteresis voltages (40 mV, 80 mV, 120 mV and 160 mV) available

16.3 Comparator Functional Description

16.3.1 Comparator Block Diagram

The comparator is described in a block diagram as shown in Figure 152.

Figure 152. Comparator Block Diagram



16.3.2 Comparator Pins and Internal Signals

To use I/O ports for the comparators, users must change the mode using the Pn_MR1 and Pn_MR2 (n = A) register in the Port Control Unit (PCU). More information about the PCU, refer to chapter 5.

The Comparator input pins of the Pin Mux are set to have the same value as the ADC (refer to section 5.4.1, section 5.4.2, and Table 35). Functions can work together to allow users to use a combination of the comparator and the ADC.

Table 128 describes pins assigned for the comparator.

Table 128. Pin Assignment of Comparator: External Pins

Pin Name	Type	Description	Package Type				
			64-LQFP	48-LQFP, QFN	44-LQFP	32-LQFP	28-TSSOP
CP0A	A	Comparator 0 input A	O	O	O	O	-
CP1A	A	Comparator 1 input A	O	O	O	O	-
CPB	A	Comparator input B	O	O	O	O	-
CPRE0	A	Comparator 0 reference input	O	O	O	O	O
CPRE1	A	Comparator 1 reference input	O	O	O	O	-
CP0OUT	O	Comparator 0 output	O	O	O	O	-
CP1OUT	O	Comparator 1 output	O	O	O	O	-

Users can refer to the values shown in Table 129 and Table 130, when selecting the input signal and reference signal.

NOTE:

1. During flash erase or program operation, it is not allowed to change the input to V_{REFINT} or from V_{REFINT} to another input by using the CINNSEL[1:0] or CINPSEL[2:0] bits.

Table 129. Comparator Input Sources

CINPSEL[1:0]	Input (+) CMP_INP
00	CP0A (PA3, INPA), CP1A (PA5, INPA)
01	CPB (PA4, INPB)
10	GND (GND)
11	V_{REFINT} (INPC)

Table 130. Comparator Reference Sources

CINNSEL[1:0]	Reference (-) CMP_INN
00	CPRE0 (PA2, INNA), CPRE1 (PA6, INNA)
01	Reserved
10	GND (GND)
11	V_{REFINT} (INNB)

For interrupt polarity settings, refer to Table 131.

If INTTYPE[1:0] is ‘01’ (Level), INTPOL is selected as Low or High. If INTTYPE[1:0] is ‘10’ (single edge), INTPOL is selected as Falling or Rising. The setting of INTPOL is ignored when INTTYPE[1:0] is selected as ’11’ (both edges).

Table 131. Comparator Interrupt Polarity

Interrupt Type	Interrupt Polarity	
INTTYPE[1:0]	INTPOL = ‘0’	INTPOL = ‘1’
00	Disable	Disable
01 (Level)	Low	High
10 (Single Edge)	Falling	Rising
11 (Both Edges)	don’t care	don’t care

16.3.3 Comparator Reference and Input Voltage Configuration

A comparator generates a digital output signal based on a voltage comparison between an input signal and a reference signal. If the input voltage is higher than the reference voltage, the digital output signal is ‘1’. Otherwise, the output signal is ‘0’.

The comparator module must be enabled before working, so the CMP in SCU_PER2 and SCU_PCER2 register must be set to ‘1’. The comparator in A31C14x is enabled when the CMPEN in CMP_CONF register is set to ‘1’.

Users can confirm the comparator’s output by checking the CMPFLAG in CMP_STAT register. If an interrupt is enabled, users can use the CMPINTF in CMP_STAT register to check whether the interrupt is generated. If CMPINTEN is enabled before CMPEN while the comparator output is low, continuous interrupts may occur. To prevent this, enable CMPEN first, clear the CMPINTF status flag, and then enable CMPINTEN.

The reference voltage of the comparator can be decided by configuring the CINNSEL[1:0] in CMP_CONF register. If the CINNSEL[1:0] is set to ‘00’, the input voltage via an external I/O pin is set as the reference voltage (CPREn). If the CINNSEL[1:0] is set to ‘11’, the V_{REFINT} voltage in the microcontroller is set as the reference voltage.

The input voltage of the comparator can be set by configuring the CINPSEL[1:0] in CMP_CONF register. Users can select one from four input sources by writing a number ranging from 0 to 3 in the CINPSEL[1:0].

The Comparator module in the A31C14x provides a function that adds hysteresis voltage to the input pin. Users can select a hysteresis voltage value by writing a proper number in the HYSSEL[1:0] in CMP_CONF register after setting the HYSEN to ‘1’ to enable the hysteresis function.

16.3.4 Comparator Reset and Clock

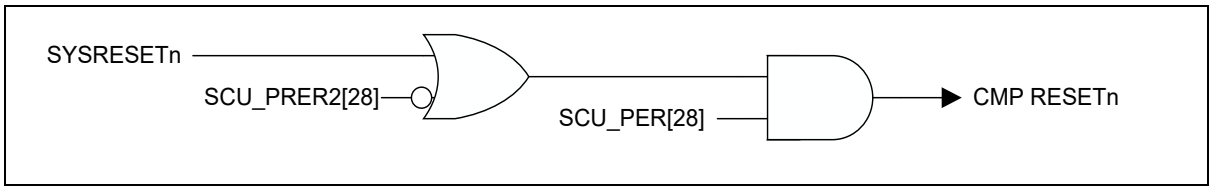
The comparator module must be enabled before working, so the CMP in SCU_PER2 and SCU_PCER2 register must be set to ‘1’.

The comparator module can be reset by disabling and enabling again the CMP in SCU_PER2 and SCU_PCER2 register, respectively.

If the corresponding reset mask bit in the SCU_PRER1 register is set to ‘0’, the settings for the comparator module can be maintained when the Warm Reset occurs.

Figure 153 shows a schematic diagram regarding the Comparator Reset signal. For more information about the comparator reset and clocks, please refer to section 4.4 and section 4.5.

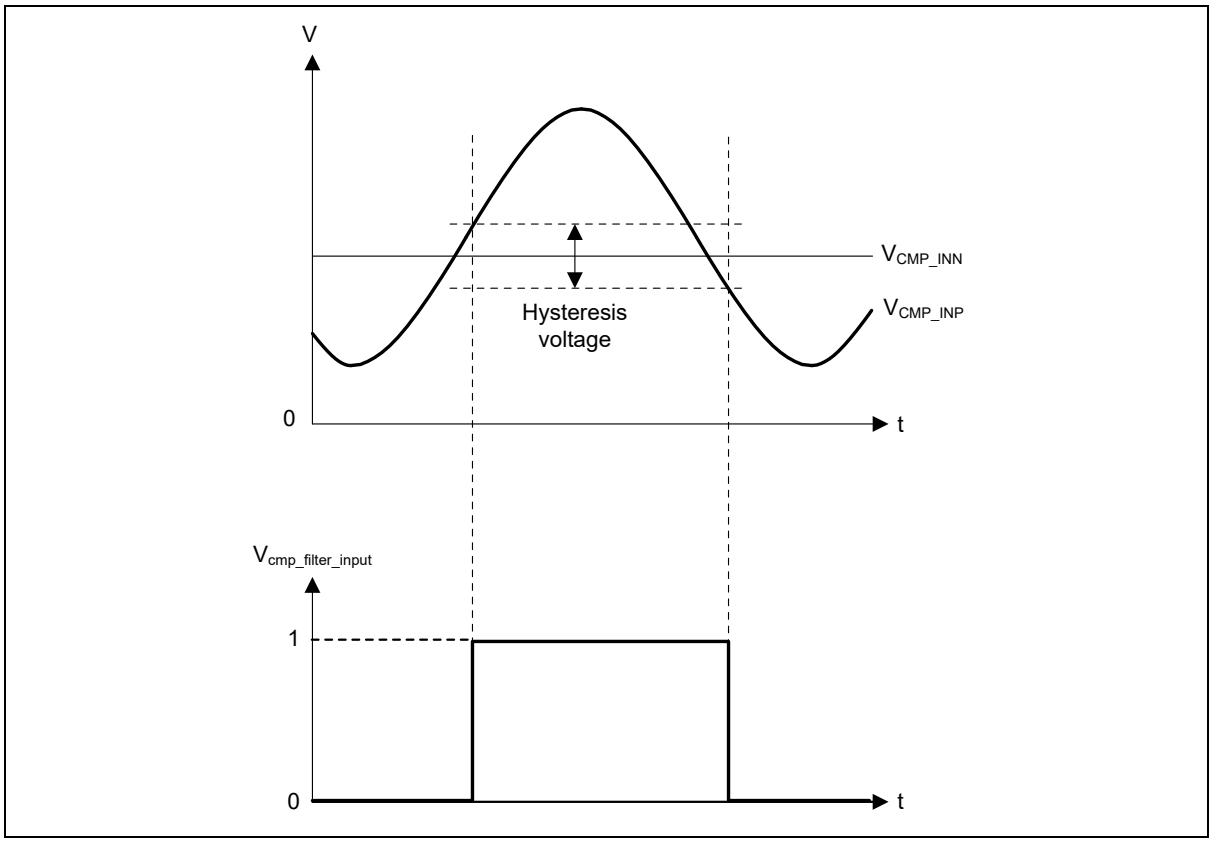
Figure 153. Schematic Diagram of Comparator Reset Signal



16.3.5 Comparator Hysteresis

If the comparator's input signal contains noise, the comparator's output may be changed at the unwanted point of time. To prevent this behavior, the comparator in A31C14x has Hysteresis function and can select Comparator Hysteresis voltage between 40mV, 80mV, 120mV and 160mV using the HYSSEL[1:0] bit. It is enabled or disabled by configuring the HYSEN bit.

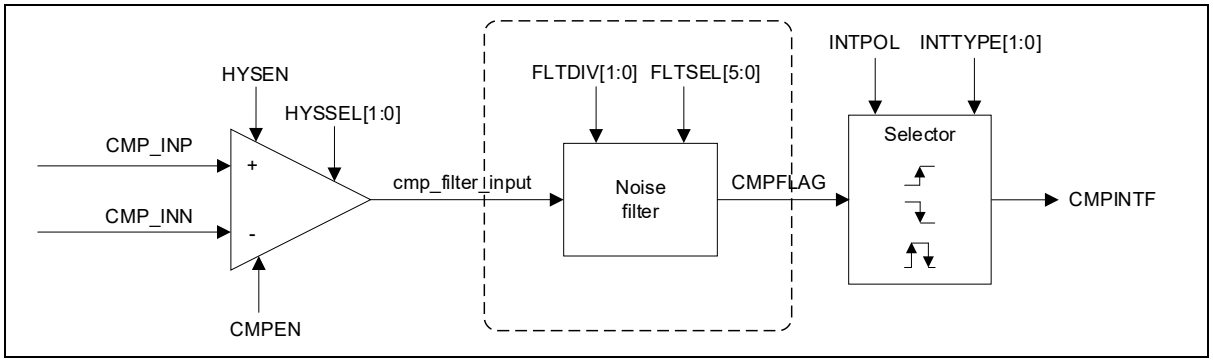
Figure 154. Comparator Hysteresis



16.3.6 Comparator Filter

A31C14x comparator has a Noise Filter inside. Figure 155 shows a block diagram of the comparator filter that operates based on the PCLK.

Figure 155. Comparator Filter



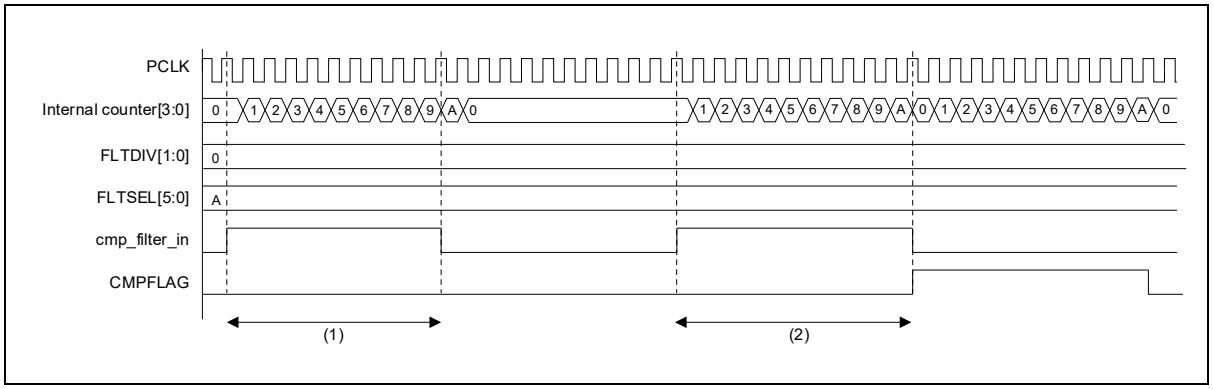
The comparator filter uses the value of the FLTSEL[5:0] multiplied by the PCLK / (2^FLTDIV[1:0]) time for filtering.

According to the example simulation shown in Figure 156, the input signal requires delay time that is longer than one tenth of the PCLK (800 kHz or higher), if the PCLK period is 8 MHz and the FLTSEL[5:0] is ‘0x0A’.

1. This section shows that the cmp_filter_input (Input signal for Noise Filter) has 1250 ns (FLTSEL[5:0] × ((1 / (PCLK / 2^FLTDIV[1:0]))) of input pulse width. In this section, the input value becomes the same as the noise filter time width, therefore the filter outputs (CMPFLAG) are canceled to generate ‘0’.
2. This section shows that the cmp_filter_input has 1,250 ns ((FLTSEL[5:0] + 1) × (1 / (PCLK / 2^FLTDIV[1:0]))) of input pulse width. In this section, the input value becomes greater than the values of the noise filter time width, therefore the filter outputs (CMPFLAG) are generated. This means that the cmp_filter_input must be maintained for at least the time of ((FLTSEL[5:0] + 1) × (1 / (PCLK/2^FLTDIV[1:0]))) to generate outputs normally.

If the “cmp_filter_input” changes while the input signal is entering, an internal counter is reset and counts from the beginning to filter the noise that occurs while the input signal enters.

Figure 156. Comparator Filter Timing



16.4 Comparator Low-Power Modes

Table 132 describes 2 low power modes that A31C14x supports for the comparator.

Table 132. Comparator Low-Power Modes

Mode	Description
SLEEP	No effect
DEEP-SLEEP	In this mode, incoming clocks to the comparator is blocked, and the corresponding comparator device is disabled.

16.5 Comparator Interrupts

Comparator interrupts occur at every rising/falling edge or high/low level of the comparator output. When the INTTYPE[1:0] in CMP_CONF register is set to ‘10’, the comparator of A31C14x detects rising edges if the INTPOL in CMP_CONF register is set to ‘1’; while it detects falling edges if the INTPOL bit is set to ‘0’. When the INTTYPE[1:0] in CMP_CONF register is set to ‘01’, the comparator of A31C14x detects high level if the INTPOL in CMP_CONF register is set to ‘1’; while it detects low level if the INTPOL bit is set to ‘0’.

Users can confirm this interrupt’s status by checking the CMPINTF in CMP_STAT register. Writing ‘1’ to the bit field clears the corresponding interrupt flag.

If the comparator is enabled while the input (+) and reference (−) voltages differ, an unintended interrupt may occur.

For example, if the rising edge interrupt is configured before enabling the comparator and the input (+) is higher than the reference (−), enabling the comparator causes CMP_OUT to transition from Low to High. This transition may result in an unintended single rising edge interrupt.

16.6 Comparator Registers

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

Table 133. Base Address of CMP

Name	Base Address
CMP0	0x4000_B380
CMP1	0x4000_B390

Table 134. CMP Register Map

Name	Offset	Type	Description	Reset Value	Reference
CMPn_CONF	0x0000	RW	Comparator n Configure Register	0x0000_0000	16.6.1
CMPn_CTRL	0x0004	RW	Comparator n Control Register	0x0000_0000	16.6.2
CMPn_STAT	0x0008	RW	Comparator n Status Register	0x0000_0000	16.6.3
CMP_DBCR	0x0040	WO	Comparator Debounce Control Register	0x0000_0000	16.6.4

16.6.1 CMPn_CONF: Comparator Configure Register

The CMPn_CONF register is used for overall settings of the Comparator.

CMP0_CONF=0x4000_B380, CMP1_CONF=0x4000_B390

31 30	29 28 27 26 25 24	23 22 21	20	19 18	17 16	15 14 13 12 11	10	9 8	7 6	5 4	3 2	1 0
FLTDIV[1:0]	FLTSEL[5:0]	Reserved	HYSEN	Reserved	HYSSEL[1:0]	Reserved	INTPOL	INTTYPE[1:0]	Reserved	CINNSEL[1:0]	Reserved	CINPSEL[1:0]
00	000000	-	0	-	00	-	0	00	-	00	-	00
RW	RW	-	RW	-	RW	-	RW	RW	-	RW	-	RW

31 30	FLTDIV[1:0]	Digital filter clock divider Filter Clock = (1 / PCLK) × (2 ^ FLTDIV[1:0])
29 24	FLTSEL[5:0]	Digital filter count Filtered by the amount of FLTSEL[5:0] value multiplied by the comparator filter clock value. If the FLTSEL[5:0] value = '2' and the comparator filter clock value = 1 MHz, then the corresponding time value of 2 MHz is 0.5 µs and the signal must be maintained for 0.5 µs or longer to generate the flag.
20	HYSEN	Hysteresis enable bit 0 Disables. 1 Enables.
17 16	HYSSEL[1:0]	Hysteresis selection bit 00 40 mV Hysteresis 01 80 mV Hysteresis 10 120 mV Hysteresis 11 160 mV Hysteresis
10	INTPOL	Interrupt polarity selection bit (The settings are irrespective of the selection of both edge.) 0 Low (Falling) 1 High (Rising)
9 8	INTTYPE[1:0]	Interrupt type selection bit 00 Disable 01 Level 10 Single edge 11 Both edge
5 4	CINNSEL[1:0](1)	Reference (input -) selection bit 00 External reference (CPREn) 01 Reserved 10 Ground 11 Internal Reference (V_{REFINT})
1 0	CINPSEL[1:0]	Source (Input +) selection bit 00 CPnA 01 CPB 10 Ground 11 Internal Reference

NOTE:

1. During flash erase or program operation, it is not allowed to change the input to V_{REFINT} or from V_{REFINT} to another input by using the CINNSEL[1:0] or CINPSEL[2:0] bits.

16.6.2 CMPn_CTRL: Comparator Control Register

The CMPn_CTRL register controls the comparator operation.

CMP0_CTRL=0x4000_B384, CMP1_CTRL=0x4000_B394

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

8	CMPINTEN	Comparator interrupt enable bit
		0 Disable
		1 Enable
0	CMPEN	Comparator enable bit
		0 Disable
		1 Enable

16.6.3 CMPn_STAT: Comparator Status Register

The CMPn_STAT register displays the operation status of the Comparator.

CMP0_STAT=0x4000_B388, CMP1_STAT=0x4000_B398

31 30 29 28 27 26 25 24	23 22 21 20 19 18 17 16	15 14 13 12 11 10 9 8	7 6 5 4 3 2 1 0
Reserved		CMPINTF	Reserved CMPFLAG
-		0	- 0
-		RWC1	- RO

8	CMPINTF	Comparator interrupt flag bit
		0 Not occurred
		1 Occurred (Writing ‘1’ to the bit clears it to ‘0’)
0	CMPFLAG	Comparator output status
		0 Low
		1 High

16.6.4 CMP_DBCR: Comparator Debounce Control Register

The CMP_DBCR register is used to apply the debounce filter to the CP0_OUT(PE0), CP1_OUT(PE1) pin output.

CMP_DBCR=0x4000_B3C0

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

0	DBEN	Comparator output pin debounce filter enable bit	
		0	disable
		1	enable

16.6.5 Comparator Register Map Summary

Table 135. Comparator Register Map Summary

Offset	Register name	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0x00	CMPn_CONF	FLTDIV[1:0]		FLTSEL[5:0]						Res.	Res.	Res.	HYSEN	Res.	Res.	HYSSEL[1:0]		Res.	Res.	Res.	Res.	Res.	INTPOL	INTTYPE[1:0]		Res.	Res.	CINNSEL[1:0]		Res.	Res.	CINPSEL[1:0]	
	Reset value	0	0	0	0	0	0	0	0				0			0	0						0	0	0			0	0			0	0
0x04	CMPn_CTRL	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	CMPINTEN	Res.	Res.	Res.	Res.	Res.	Res.	Res.	CMPEN
	Reset value																								0								0
0x08	CMPn_STAT	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	CMPINTF	Res.	Res.	Res.	Res.	Res.	Res.	Res.	CMPFLAG
	Reset value																								0								0
0x40	CMP_DBCR	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.	DBEN
	Reset value																																0

17. CRC (Cyclic Redundancy Check Calculation Module)

17.1 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 A31C14x and A31C12x series supports DMA.

17.2 CRC Main Features

The CRC of the A31C14x and A31C12x series features the following:

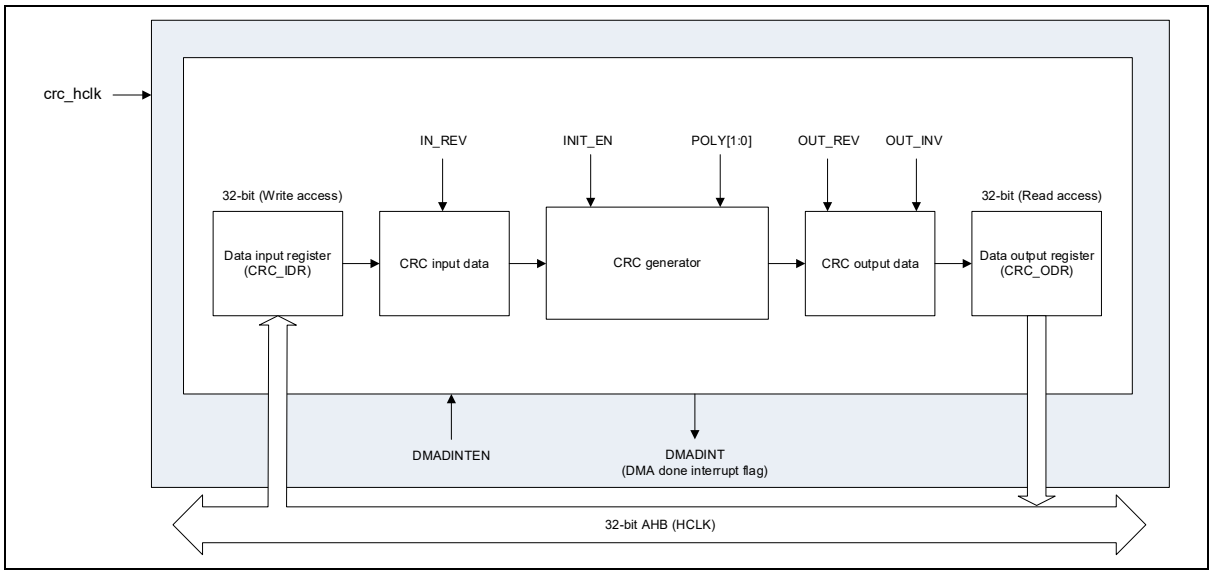
- Handles 8-, 32-bit data size
- Input buffer to avoid bus stall during calculation
- CRC calculation is automatically done for word/byte input data size
  - 4 AHB clock cycles (HCLK) for 32-bit
  - 2 AHB clock cycles (HCLK) for 16-bit
  - 1 AHB clock cycles (HCLK) for 8-bit
- CRC code selectable by 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

17.3 CRC Functional Description

17.3.1 CRC Block Diagram

Figure 157 shows a block diagram of the A31C14x and A31C12x series CRC module.

Figure 157. CRC Block Diagram



17.3.2 CRC Internal Signals

Table 136. CRC Internal Input/Output Signals

Signal Name	Signal Type	Description
crc_hclk	clock	AHB clock

17.3.3 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 158 shows a timing diagram of 8-bit size input data and CRC-8 calculation (outputs). Signals in Figure 158 such as the HCLK (crc_hclk), CRC enable bit (crc_cal_en), input data (IDR[7:0]), and output data (ODR[7:0] are internal signals, and show the cycles required for internal calculation.

The CRC-8 calculated values are output every 1 AHB clock cycle.

Figure 158. Timing Diagram of CRC Internal Signals: 8-bit Size Input and CRC-8 Outputs

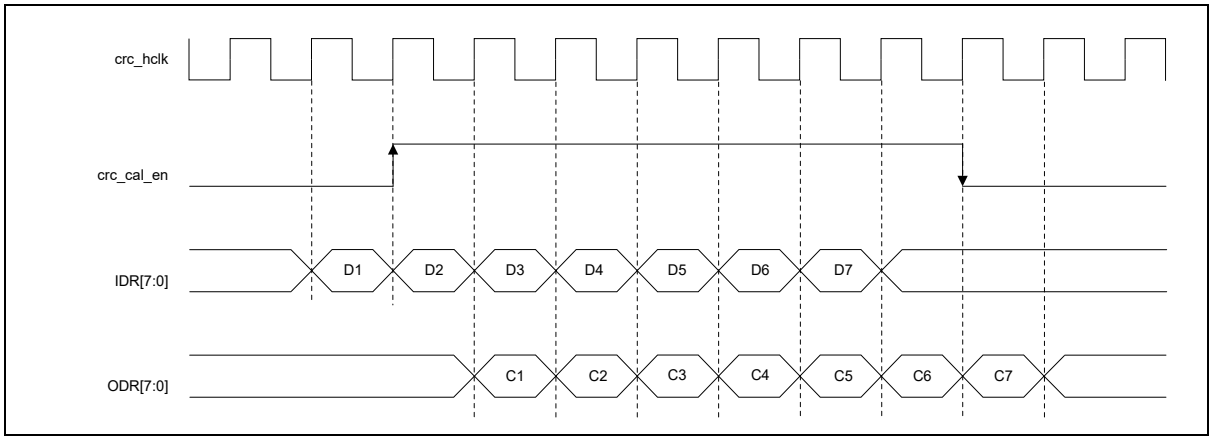


Figure 159 shows a timing diagram of 32-bit size input data and CRC-8 calculation (outputs).

Signals in Figure 159 show that the 32-bit size input data enter at once and the CRC-8 calculated values are output from the point of single AHB clock cycle.

Figure 159. Timing Diagram of CRC Internal Signals: 32-bit Size Input and CRC-8 Outputs

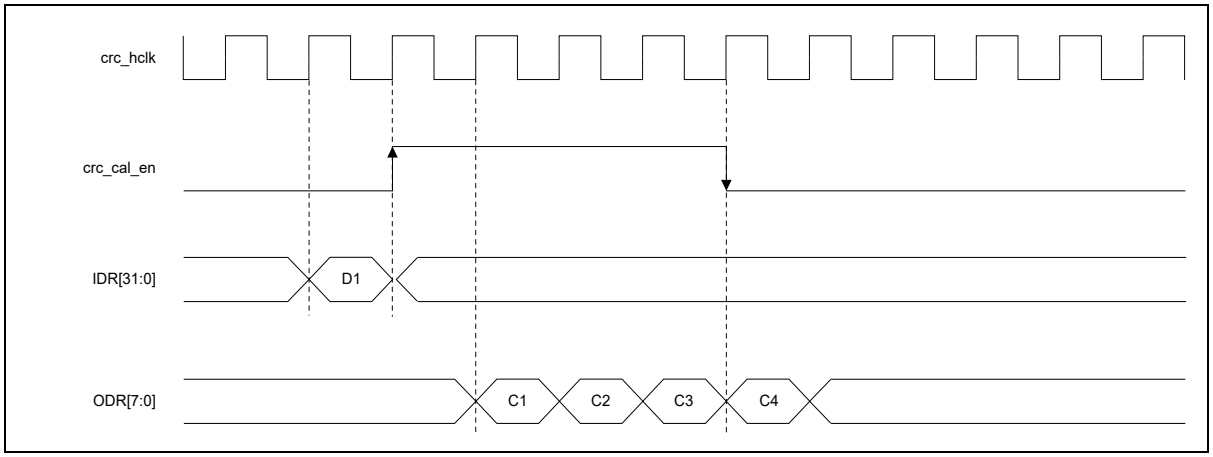


Figure 160 shows a timing diagram of 32-bit size input data and CRC-16 calculation (outputs).

Signals in Figure 160 show that the 32-bit size input data enter at once and two CRC-16 values are output. The first CRC-16 calculated value is output at the point of 2 AHB clock cycle and the second CRC-16 calculated in parallel is output at the final point of 2 + 2 AHB clock cycles.

Figure 160. Timing Diagram of CRC Internal Signals: 32-bit Size Input and CRC-16 Outputs

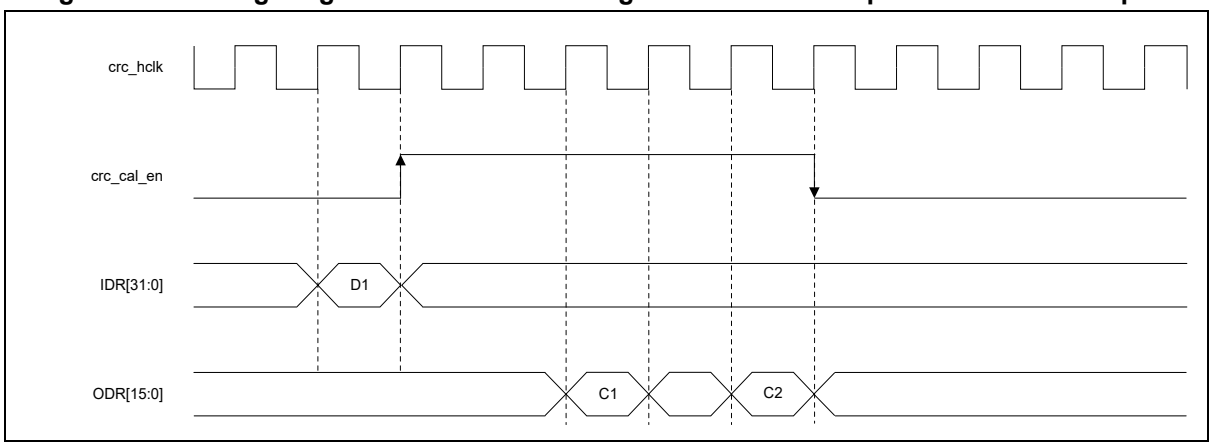
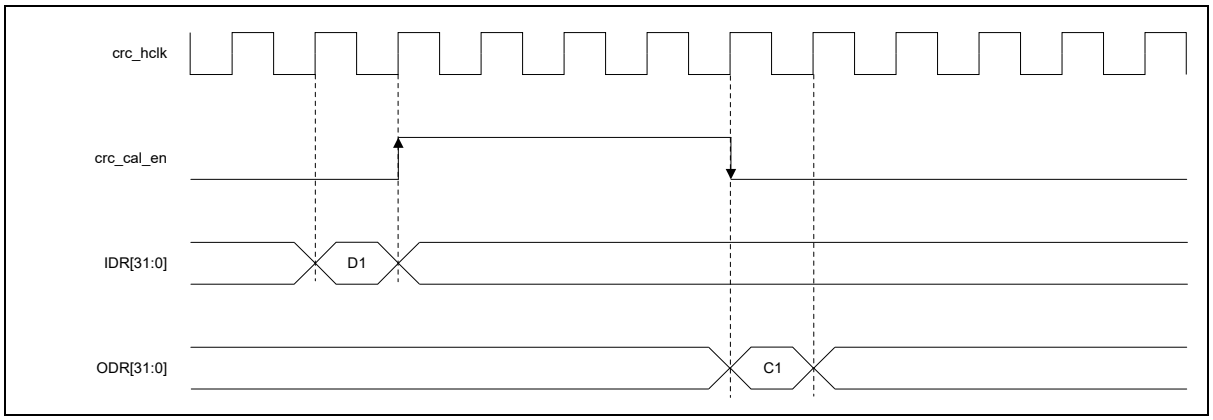


Figure 161 shows a timing diagram of 32-bit size input data and CRC-32 calculation (outputs).

Signals in Figure 161 show that the 32-bit size input data enter at once and the CRC-32 calculated values are output from the point of four AHB clock cycles.

Figure 161. Timing Diagram of CRC Internal Signals: 32-bit Size Input and CRC-32 Outputs



17.3.4 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 a 32-bit input data and internally processes the CRC calculation of 16-bit data two times.

17.3.4.1 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, user 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. 4 AHB clock cycles for 32-bit
    B. 2 AHB clock cycles for 16-bit
    C. 1 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'.

17.3.5 CRC using DMA

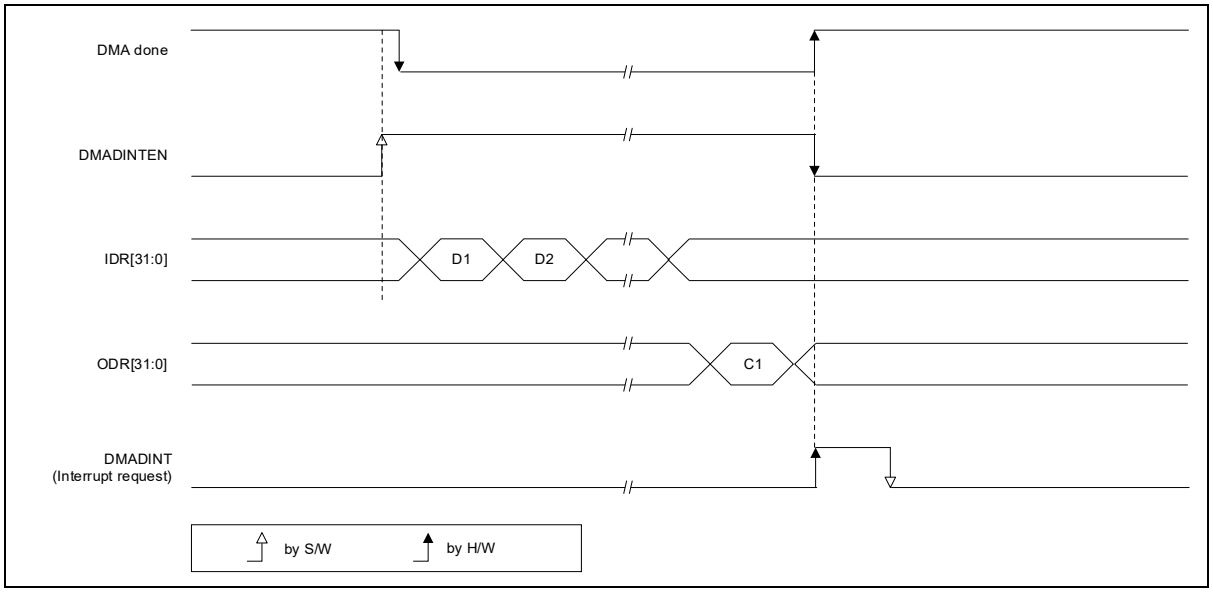
The CRC module supports DMA function.

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 DMAn_SR register to tr ansfer the CRC value using DMA.
6. Once the DMA transfer is started and when value of the TRANSCNT[11:0] in DMAn_CR regi ster reaches ‘0’, the EOT in DMAn_SR register is enabled indicating that the DMA transfer e nds.
7. For detailed information on the DMA operation, refer to chapter 7.

Figure 162 shows a timing diagram of the CRC operation using DMA.

Figure 162. Timing Diagram of CRC Operation using DMA



17.4 CRC Registers

The base address and register map of the CRC module are described in the following tables.

Table 137. Base Address of CRC

Name	Base Address
CRC	0x4008_2000

Table 138. CRC Register Map

Name	Offset	Type	Description	Reset value	Reference
CRC_CTRL	0x0000	RW	CRC Control Register	0x0000_0000	17.4.1
CRC_INIT	0x0004	RW	CRC Initial data Register	0xFFFF_FFFF	17.4.2
CRC_IDR	0x0008	WO	CRC Input data Register	-	17.4.3
CRC_ODR	0x0008	RO	CRC Output data Register	0xFFFF_FFFF	17.4.4
CRC_STAT	0x000C	RWC1	CRC Status Register	0x0000_0000	17.4.5

17.4.1 CRC_CTRL: CRC Control Register

The CRC_CTRL register is used to control the CRC.

CRC_CTRL=0x4008_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 to apply the CRC initial value register (auto clear)	
		0	No Effect
		1	Applies the CRC_INIT register value.

17.4.2 CRC_INIT: CRC Initial Data Register

The CRC initial value data is written to the CRC_INIT register.

CRC_INIT=0x4008_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'.

17.4.3 CRC_IDR: CRC Input Data Register

The CRC_IDR register can be accessed by 32-, 16- and 8-bit size. This register is 32 bits wide.

CRC_IDR=0x4008_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.

17.4.4 CRC_ODR: CRC Output Data Register

The CRC_ODR register can be accessed by 32-, 16- and 8-bit size. This register is 32 bits wide.

CRC_ODR=0x4008_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

17.4.5 CRC_STAT: CRC Status Register

The CRC_STAT register displays the operating status of the CRC.

CRC_STAT = 0x4008_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.)

17.4.6 CRC Register Map Summary

Table 139. 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	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-
0x0C	CRC_ODR	ODR[31:0]																															
	Reset value	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1
0x10	CRC_STAT	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	DMADINT	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.
	Reset value																								0								

18. DBGMCU (Microcontroller Debug Component)

18.1 DBGMCU Introduction

The A31C14x and A31C12x series supports maintaining the debugging interface (SWD) even when the microcontroller enters STOP mode, allowing continuous software debugging without loss of connection.

It also provides clock control function that enables the Timer, WDT, and FRT to continue operating during debugging.

18.2 DBGMCU Main Features

The DBGMCU of the A31C14x and A31C12x series includes the following features:

- The debugger connection can be maintained during STOP mode
- Clock control is available for Timers, WDT and FRT

18.3 DBGMCU Functional Description

18.3.1 Debug Support for STOP Mode

To enter STOP mode, the WFI instruction must be executed.

To enable debug connectivity during STOP mode, the DBG_STOP bit in the DBGMCU_CR register must be set to ‘1’ in advance. This setting ensures that the internal HSI oscillator provides clock to FCLK and HCLK during STOP mode, allowing the debugger connection to remain active.

18.3.2 Debug Support for Timers, WDT and FRT

The operation of the Timer counter, WDT, and FRT during breakpoints must be configured by the user.

By default, these counters stop when a breakpoint occurs. To enable continuous operation during breakpoints, clear the DBG_TIMER_STOP, DBG_IWDT_STOP, and DBG_FRT_STOP bits to ‘0’. The user should set these bits based on the intended debug behavior.

For the A31C14x and A31C12x series devices, if the FRT or Timer clock source is set to PCLK, the counter continues to operate during a core halt regardless of the corresponding bit settings. Use caution when debugging under this configuration. When the clock source is set to EXT_CLK(MCCR), the counter behavior follows the bit settings as expected.

18.4 DBGMCU Registers

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

Table 140. Base Address of DBGMCU

Name	Base Address
DBGMCU	0x4000_F020

Table 141. DBGMCU Register Map

Name	Offset	Type	Description	Reset Value	Reference
DBGMCU_CR	0x0000	RW	Debug Microcontroller Control Register.	0x003F_F500	18.4.1

18.4.1 DBGMCU_CR: Debug Microcontroller Control Register

The Debug Microcontroller Control register controls the comparator operation.

DBGMCU_CR=0x4000_F020

31 30 29 28 27 26 25 24	23 22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7 6 5 4 3 2	1	0
Reserved		DBG_TIMER9_STOP	DBG_TIMER8_STOP	DBG_TIMER7_STOP	DBG_TIMER6_STOP	DBG_TIMER5_STOP	DBG_TIMER4_STOP	DBG_TIMER3_STOP	DBG_TIMER2_STOP	DBG_TIMER1_STOP	DBG_TIMER0_STOP	Reserved	DBG_FRT_STOP	Reserved	DBG_IWDT_STOP	Reserved	DBG_STOP	Reserved
-		1	1	1	1	1	1	1	1	1	1	-	1	-	1	-	0	-
-		RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	-	RW	-	RW	-	RW	-

21 12	DBG_TIMER_STOP[9:0](1)	Debug Timer[9:0] stopped when core was halted. This bit only works if TIMERn CR1 CKSEL bit is 10x(MCCR3)
		0 The timer counter clock continues even if the core is halted.
		1 The timer counter clock is stopped when the core is halted.
10	DBG_FRT_STOP(1)	Debug FRT stopped when core is halted.
		0 The FRT counter clock continues even if the core is halted.
		1 The FRT counter clock is stopped when the core is halted.
8	DBG_IWDT_STOP(1)	Debug independent watchdog stopped when core is halted.
		0 The watchdog counter clock continues even if the core is halted.
		1 The watchdog counter clock is stopped when the core is halted.
1	DBG_STOP	Debug stop mode.
		0 (FCLK=Off, HCLK=Off) In STOP mode, the clock controller disables all clocks (including HCLK and FCLK). When exiting from STOP mode, the clock configuration is identical to the one after RESET(CPU clocked by the 4 MHz HSI). Consequently, the software must reprogram the clock controller to enable the Xtal, etc.
		1 (FCLK=On, HCLK=On) In this case, when entering STOP mode, FCLK and HCLK are provided by HSI which remain active in STOP mode. When exiting STOP mode, the software must reprogram the clock controller to enable the Xtal, etc. (in the same way it would do in case of DBG_STOP=0)

NOTE:

1. If the FRT, Timer clock source is set to PCLK, the counter continues operating during core halt, regardless of the corresponding bit setting.

18.4.2 DBGMCU Register Map Summary

Table 142. DBGMCU 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
0x20	DBGMCU_CR	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	DBG_TIMER_STOP[9]	DBG_TIMER_STOP[8]	DBG_TIMER_STOP[7]	DBG_TIMER_STOP[6]	DBG_TIMER_STOP[5]	DBG_TIMER_STOP[4]	DBG_TIMER_STOP[3]	DBG_TIMER_STOP[2]	DBG_TIMER_STOP[1]	DBG_TIMER_STOP[0]	Res.	DBG_FRT_STOP	Res.	DBG_IWDT_STOP	Res.	Res.	Res.	Res.	Res.	Res.	DBG_STOP	Res.
	Reset value											1	1	1	1	1	1	1	1	1	1		1		1							0	

Abbreviation for Registers

The following abbreviations are used in register descriptions:

Table 143. Abbreviation for Registers

Abbreviation	Description
RW (Read/write)	Software can read and write to this bit.
RO (Read only)	Software can only read this bit.
WO (Write-only)	Software can only write to this bit. Reading this bit returns the reset value.
RC (Read/clear)	Software can read to clear this bit.
RWC0 (Read/write 0 clear)	Software can read and clear this bit by writing ‘0’.
RWC1 (Read/write 1 clear)	Software can read and clear this bit by writing ‘1’.
Res. (Reserved)	Reserved bit, must be kept at reset value.

Glossary

This section gives a brief definition of acronyms, abbreviations, and terminology used in this document:

- Word: Data of 32-bit length
- Half-word: Data of 16-bit length
- Byte: Data of 8-bit length
- PGM: Flash Programming
- AHB: Advanced High-performance Bus
- APB: Advanced Peripheral Bus
- CRC: Cyclic Redundancy Check
- DMA: Direct Memory Access
- FRT: Free Run Timer
- HSE: High-Speed External
- HSI: High-Speed Internal
- I2C: Inter-Integrated Circuit
- LSB: Least Significant Bit
- LQFP: Low-profile Quad Flat Package
- TSSOP: Thin Shrink Small Outline Package
- LSE: Low-Speed External
- LSI: Low-Speed Internal
- LVI: Low-Voltage Indicator
- LVR: Low-Voltage Reset
- MSB: Most Significant Bit
- POR: Power On Reset
- SCU: System Control Unit
- UART: Universal Asynchronous Receiver Transmitter
- USART: Universal Synchronous Asynchronous Receiver Transmitter
- WDT: Watchdog Timer

Revision History

Revision	Date	Notes
1.00	Sep. 25, 2025	Initial release
1.01	Jan. 30, 2026	• Updated the Flash memory organization details in section 6.3.1. • Updated the write protection configuration and descriptions in section 6.4.2. • Updated the HSICMU initialization and configuration sequences in section 7.3.5. • Added the VCORE (Internal reference voltage) as an ADC input channel. • Added a cautionary statement that the comparator input must not be changed from VREFINT to any other input while a Flash erase or write operation is in progress. • Added a note specifying that the DMATXF flag is not asserted when sending a single byte via USART DMA, even though the waveform is generated normally. • Corrected miscellaneous typos.
1.02	Aug. 20, 2026	• Added New Package (20-QFN/20-TSSOP/32-LQFP) • Modified Flash Endurance Cycles in Table 2. • Modified the default configuration of PD0, PD1, PE2, and PE3 pins in Table 4. • Corrected miscellaneous typos.
1.03	Sep. 16, 2026	Corrected Wake-up Source in Table 25.

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.abovsemi.com/en/about/corporate_identity.php. All other products or service names are the property of their respective owners. Information in this document supersedes and replaces the information previously supplied in any former versions of this document.

© 2025 - 2026 ABOV Semiconductor – All rights reserved