MAKSA Dual-Core MCU
MKC40AT
User Manual

Rev1.5

© MAKSA LTD

Revision History

Version	Release Date	Key Changes
Rev1.0	2025/10/24	Initial release
Rev1.1	2025/11/25	1. Add registers and function descriptions. 2. Update the CMP/LVD chapter and add application notes. 3. Update the WDT chapter and add the WDT_STBY register. 4. Add the kernel module chapter. 5. Update the LCD chapter and modify the operation steps.
Rev1.2	2025/12/10	1. Update the DSP chapter: Modify descriptions of some registers and add interpolation and filter mode explanations. 2. Update the System Control chapter and add descriptions of the CORE1_CKE register. 3. Update the RTC chapter and modify the representable range of T0 in the RTC_XT0 register. 4. Update the GPIO chapter and modify the LVD multiplexing description of the PA04 pin. 5. Update the DMA chapter and revise descriptions of some functions.
Rev1.3	2025/12/19	1. Update the pin diagrams and pin definitions of MKC40AT: PIN64 is changed from PE00 to DGND. 2. Optimize the document format.
Rev1.4	2025/12/20	1. Update the KBI&INTC chapter: Rename the chapter from "KBI INTC" to "KBI&INTC", and add configuration notes for the KBI_MODE and CHx_FLT_TIME registers.
Rev1.5	2025/12/24	Update the Overview chapter and modify the description of the VREF pin definition.

Contents

REV1.5 ..... **1**

REVISION HISTORY **2**

TABLE OF CONTENTS **3**

1 OVERVIEW **24**

1.1 PRODUCT POSITIONING 24
1.2 PRODUCT FEATURES 24
1.3 SYSTEM BLOCK DIAGRA 27
1.4 PACKAGE AND MODEL SELECTION 28
1.5 PIN ARRANGEMENT 28
1.5.1 MKC40AT LQFP100L Pin Diagram 28
1.5.2 Pin Type Description 30
1.5.3 Pin Definition Description 31

2 ELECTRICAL CHARACTERISTICS **47**

3 SYSTEM ARCHITECTURE **51**

3.1 INTRODUCTION 51
3.2 RISC-V CORES 52
3.3 BUS ARCHITECTURE 53
3.3.1 Overall Block Diagram 53
3.3.2 Bus Matrix 55
3.4 MEMORY MAP 56
3.4.1 Memory Map Diagram 56
3.4.2 Detailed Address Mapping Tables 56
3.4.3 Bitband Region 63
3.4.4 Master Security Attributes 63

4 SYSTEM CONTROL **64**

4.1 POWER MANAGEMENT UNIT (PMU) 64
4.1.1 Power Scheme 64
4.1.2 Power Control 66
4.1.3 Power Monitoring 67
4.2 CLOCK MANAGEMENT UNIT (CMU) 67
4.3 RESET MANAGEMENT UNIT (RMU) 68
4.4 POWER CONSUMPTION MODES 68
4.4.1 Overview 68
4.4.2 HCM Mode 69
4.4.3 RCM Mode 70
4.4.4 LCM Mode 70
4.4.5 Sleep Mode 70
4.5 SLEEP AND WAKE-UP SCHEMES 70

4.5.1 Overview 70
4.5.2 Single Core Core0 Sleep/Wake-up Scheme 71
4.5.3 Single Core Core1 Sleep/Wakeup Scheme 73
4.5.4 Dual Core Sleep/Wakeup Scheme 74
4.6 CORE0 SYSTEM CONTROL REGISTERS 75
4.6.1 Password Class Registers 75
4.6.2 Clock Control Class Registers 76
4.6.3 Clock Calibration Class Registers 90
4.6.4 Reset Class Registers 93
4.6.5 Analog Function Class Registers 100
4.6.6 Digital Function Class Registers 101
4.6.7 Chip Version Class Registers 106
4.7 CORE1 SYSTEM CONTROL REGISTERS 107
4.7.1 Password Class Registers 107
4.7.2 Clock Control Class Registers 107
4.7.3 Reset Class Registers 116
4.7.4 Digital Function Class Registers 122
4.7.5 Chip Version Class Registers 128

5 KERNEL MODULES 129

5.1 TIMER 129
5.1.1 Features 129
5.1.2 Registers 129
5.1.3 Software Configuration Process 131
5.2 ECLIC 131
5.2.1 Features 131
5.2.2 Registers 136
5.2.3 Software Configuration Flow 138
5.3 PMP 138
5.3.1 Features 138
5.3.2 Registers 140
5.3.3 Software Configuration Flow 142

6 LOSC CLOCK SECURITY SYSTEM (LCSS) 143

6.1 LCSS_VCC 143
6.1.1 Introduction 143
6.2 LCSS_VBAT 143
6.2.1 Introduction 143

7 DMA 144

7.1 INTRODUCTION 144
7.1.1 Definition of Proprietary Terms 144
7.2 FEATURES 144
7.3 FUNCTIONAL DESCRIPTION 145

7.3.1 General-purpose DMA Functional Description 145
7.3.2 Special-purpose DMA Functional Description 148
7.3.3 SRAM DMA Arbitration Functional Description 148
7.4 General-purpose DMA Registers 151
7.4.1 GDMA0 Configuration Register GDMA0_SxCTRL (0x00 + 0x20*x) 152
7.4.2 GDMA0 Transfer Data Length Configuration Register GDMA0_SxNDTR (0x04 + 20*x) 152
7.4.3 GDMA0 Memory End Base Address Register GDMA0_SxMADR (0x08 + 20*x) 153
7.4.4 GDMA0 Peripheral End Base Address Register GDMA0_SxPADR (0x0C + 20*x) 153
7.4.5 GDMA0 Peripheral Interrupt Enable Configuration Register GDMA0 _SIE（0x200） 153
7.4.6 GDMA0 Peripheral Interrupt Flag Register GDMA0_SIF（0x204） 155
7.5 General GDMA1 Registers 156
7.5.1 GDMA1 Configuration Register GDMA1_SxCTRL（0x00+0x20*x） 156
7.5.2 GDMA1 Transfer Data Length Configuration Register GDMA1_SxNDTR（0x04+20*x） 157
7.5.3 GDMA1 Memory Base Address Register GDMA1_SxMADR（0x08+20*x） 157
7.5.4 GDMA1 Transfer Data Length Configuration Register GDMA1_SxPADR（0x0C+20*x） 158
7.5.5 GDMA M2M Channel i Configuration Register GDMA1_MiCTRL（0x100 +20*i） 158
7.5.6 GDMA M2M Channel i Transfer Data Length Configuration Register GDMA1_MiNDTR（0x104 +20*i） 159
7.5.7 GDMA M2M Channel i Destination Address Register GDMA1_MiDADR（0x108 +20*i） 159
7.5.8 GDMA M2M Channel i Source Address Register GDMA1_MiSADR（0x10C+20*i） 159
7.5.9 GDMA1 Peripheral Interrupt Enable Configuration Register GDMA1 _SIE（0x200） 159
7.5.10 GDMA1 Peripheral Interrupt Flag Register GDMA1_SIF（0x204） 160
7.5.11 GDMA M2M Channel Interrupt Enable Configuration Register GDMA1 _MIE（0x208） 162
7.5.12 GDMA M2M Channel Interrupt Flag Register GDMA1 _MIF（0x20C） 162
7.6 Software Configuration Flow and Application Notes 163

8 MAILBOX 164

8.1 Introduction 164
8.2 Features 164
8.3 Block Diagram 164
8.4 Clock and Reset 165
8.5 Functional Description 165
8.5.1 Connecting to Mailbox 165
8.5.2 Disconnecting from Mailbox 165
8.5.3 Memory Space 165
8.6 Interrupts 166
8.7 Registers 166
8.7.1 CSR（0x08） 167
8.7.2 RESET（0x0C） 168
8.7.3 LINKID（0x10） 168
8.7.4 LOCKOUT（0x18） 169
8.7.5 INTEN0（0x38） 169
8.7.6 INTEN1（0x38） 169
8.7.7 INTST0（0x3C） 170

8.7.8 INTST1（0x3C）.....170
8.7.9 OPT（0x5C）.....170
8.7.10 FROZE（0x60）.....171
8.7.11 ID_VAL（0x64）.....171
8.7.12 ID_MASK（0xE4）.....171
8.7.13 MBX_KEY0（0x300）.....172
8.7.14 MBX_KEY1（0x304）.....172
8.8 SOFTWARE OPERATION FLOW.....172
8.8.1 Module Initialization Flow.....172
8.8.2 Polling Mode Communication Flow.....173
8.8.3 Interrupt Mode Communication Flow.....174
8.8.4 Application Notes.....175
9 GIC.....176
9.1 INTRODUCTION.....176
9.2 FEATURES.....176
9.3 BLOCK DIAGRAM.....176
9.4 FUNCTIONAL DESCRIPTION.....176
9.4.1 External Interrupt Distribution.....176
9.4.2 Inter-Core Interrupt (ICI).....177
9.4.3 Interrupt Vector Table.....177
9.5 REGISTERS.....181
9.5.1 CORE0_ICI_STA0（0x00）.....182
9.5.2 CORE0_ICI_STA1（0x04）.....182
9.5.3 CORE1_ICI_STA0（0x10）.....182
9.5.4 CORE1_ICI_STA1（0x14）.....182
9.5.5 ICI_GEN（0x20）.....183
9.5.6 CORE_IND_LOCK（0x24）.....183
9.5.7 CORE_NUM（0x28）.....183
9.5.8 HART_NUM（0x2C）.....183
9.5.9 SOC_IRQ_NUM（0x30）.....183
9.5.10 SOC_INTn_INDICATOR, n=20~146（0x40~0x40+4*（n-20））.....183
9.6 SOFTWARE CONFIGURATION FLOW AND APPLICATION NOTES.....184
9.6.1 Inter-Core Interrupt Configuration Flow.....184
9.6.2 GIC Interrupt Distribution Configuration Flow.....184
10 HSEM.....185
10.1 INTRODUCTION.....185
10.2 FUNCTIONAL DESCRIPTION.....185
10.3 BLOCK DIAGRAM.....185
10.4 CLOCK AND RESET.....185
10.5 REGISTERS.....186
10.5.1 SEMAPHORE0~31（0x00~0x7C）.....186
10.6 SOFTWARE CONFIGURATION FLOW AND APPLICATION NOTES.....186

10.6.1 HSEM (Hardware Semaphore) Configuration Flow 186

11 SCU 187

11.1 INTRODUCTION 187
11.2 FEATURES 188
11.3 PERIPHERAL AND SECURITY MECHANISM LIST 188
11.4 STRUCTURE BLOCK DIAGRAM 190
11.5 PERIPHERAL ACCESS RULES 191
11.6 RESET 193
11.7 REGISTER DESCRIPTION 193
11.7.1 SCU_CFG Security Configuration Register Description 194
11.7.2 SCU_IC Interrupt Control Register Description 199
11.8 SOFTWARE USAGE FLOW 206

12 FLASH 208

12.1 INTRODUCTION 208
12.2 FEATURES 208
12.3 CLOCK 208
12.4 FUNCTIONAL DESCRIPTION 208
12.4.1 Storage Area Partitioning 208
12.5 REGISTER DEFINITION 209

13 KEY STORE 210

13.1 INTRODUCTION 210
13.2 FEATURES 210
13.3 BASIC PRINCIPLES 210
13.3.1 Storage Devices 210
13.3.2 Storage Format 211
13.3.3 Key Blocks 211
13.3.4 Attribute Parameters 211
13.4 REGISTER DEFINITION 212

14 M2M 213

14.1 INTRODUCTION 213
14.2 FEATURES 213
14.2.1 Input/Output Data Valid Bit Width Configuration 213
14.2.2 Configurable Dummy Value for Invalid Output Addresses 213
14.2.3 Output Data Reverse Placement 213
14.2.4 Configurable Source and Destination Addresses, Configurable Input Data Length 214
14.2.5 Data Arranged in Reverse Address Order 214
14.3 EXAMPLE DESCRIPTIONS 214
14.3.1 Input 4 Bytes Valid, Output 3 Bytes Valid 214
14.3.2 Input 2 Bytes Valid, Output 4 Bytes Valid 214
14.3.3 Input Data Stored in Reverse Byte Order (Input 4 Bytes Valid, Output 2 Bytes Valid) 214
14.3.4 Data Arranged in Reverse Address Order 215

14.4 Transfer Speed Evaluation 215
14.5 Design Detail Handling 215
14.6 Register Description 215
14.6.1 M2M_MODE 216
14.6.2 M2M_CTL 216
14.6.3 M2M_DUMMY 217
14.6.4 M2M_SADDR 217
14.6.5 M2M_DADDR 217
14.6.6 M2M_ILEN 217
14.6.7 M2M_IE 218
14.6.8 M2M_IF 218
14.7 Software Operation Flow 218

15 AOS 219

15.1 Introduction 219
15.2 Features 219
15.3 Functional Description 219
15.3.1 Trigger Source Selection Mechanism 219
15.3.2 Trigger Targets 219
15.3.3 TC Trigger Sources 220
15.3.4 SIMPTC Trigger Sources 221
15.3.5 D2F Trigger Sources 222
15.3.6 IOCNT Trigger Sources 222
15.3.7 CLK_CAL Trigger Sources 223
15.3.8 ADC0 Trigger Sources 223
15.3.9 DAC Trigger Sources 224
15.4 Registers 224
15.4.1 SEC_EN（0x00） 224
15.4.2 NSEC_EN（0x04） 224
15.4.3 PE_TRGSELx（0x10~0x88, 0x90~0x98） 225
15.5 Software Configuration Flow and Application Notes 225

16 WDT 226

16.1 Main Features 226
16.2 Register Description 226
16.2.1 Watchdog Enable Register WDT_EN（0x00） 226
16.2.2 Watchdog Configuration Register WDT_CTL (0x04) 227
16.2.3 Watchdog Password Register WDT_PASS(0x08) 228
16.2.4 Watchdog HALT Control Register WDT_HALT(0x14) 228
16.2.5 Watchdog STBY Control Register WDT_STBY(0x18) 229
16.2.6 Watchdog STOP Control Register WDT_STOP(0x1C) 229
16.2.7 Watchdog Counter Register WDT_CNT(0x20) 229

17 SIMPTC 230

17.1 INTRODUCTION 230
17.2 FUNCTIONAL DESCRIPTION 230
17.3 REGISTERS 230
17.3.1 SIMPTCx_y_CTRL（0x0） 231
17.3.2 SIMPTCx_ y_DN（0x4） 232
17.3.3 SIMPTCx_ y_VAL（0x8） 232
17.3.4 SIMPTCx_ y_IF（0xC） 232

18 UART 233

18.1 INTRODUCTION 233
18.2 FEATURES 233
18.3 BLOCK DIAGRAM 234
18.4 PIN DEFINITION 235
18.5 INTERFACE TIMING 235
18.6 FUNCTIONAL DESCRIPTION 235
18.6.1 Security Attribute Configuration Description 235
18.6.2 Baud Rate 236
18.6.3 Data Frame Format 237
18.6.4 Other Settings 237
18.6.5 Infrared Modulation Module 237
18.6.6 DMA Function 238
18.6.7 AOS Function 238
18.6.8 LSRX Linkage 238
18.6.9 UART Metering Mode 239
18.7 REGISTERS 242
18.7.1 Register List 243
18.7.2 UARTx_CTRL（0x00） 244
18.7.3 UARTx_BAUD（0x04） 247
18.7.4 UARTx_STA（0x08） 248
18.7.5 UARTx_TXD（0x0C） 249
18.7.6 UARTx_RXD（0x10） 249
18.7.7 UARTx_FDIV（0x14） 249
18.7.8 UARTx_DMA_CTL（0x18） 250
18.7.9 UARTx_DMA_TBADR（0x1C） 250
18.7.10 UARTx_DMA_RBADR（0x20） 250
18.7.11 UARTx_DMA_TLEN（0x24） 251
18.7.12 UARTx_DMA_RLEN（0x28） 251
18.7.13 UARTx_DMA_TADR（0x2C） 251
18.7.14 UARTx_DMA_RADR（0x30） 251
18.7.15 UARTx_DMA_IE（0x34） 251
18.7.16 UARTx_DMA_FLG（0x38） 252
18.7.17 UARTx_DMA_TO（0x3C） 252
18.7.18 UARTx_TX_DRLAY（UART0-5 Relative Offset: 0x40） 252
18.7.19 UART_CF_CTL（0x40） 253

18.7.20 UART_PFDAT（0x44）.....253
18.7.21 UART_QFDAT（0x48）.....253
18.7.22 UART_FPFDAT（0x4C）.....254
18.7.23 UART_IRQDAT（0x50）.....254
18.7.24 UART_CF_OV（0x54）.....254
18.7.25 UART_TW_CTL（0x58）.....254
18.7.26 UART_TW_BAUD（0x5C）.....255
18.7.27 UART_TW_EN（0x60）.....255
18.7.28 UART_CAL_CTL（0x64）.....255
18.7.29 UART_CAL_CNT（0x68）.....256
18.7.30 UARTx_TX_DRLAY（UART6-7 Relative Offset: 0x6C）.....256
18.8 Software Configuration Flow and Application Instructions.....256
18.8.1 UART Security Attribute Configuration.....256
18.8.2 UART Conventional Function Configuration Flow.....256
18.8.3 UART Infrared Modulation Function Configuration Flow.....257
18.8.4 UART Dedicated DMA Function.....257
18.8.5 UART AOS Function.....257
18.8.6 UART and LSRX Linkage Usage.....257

19 TC.....258

19.1 Introduction.....258
19.2 Features.....258
19.3 Functional Block Diagram.....259
19.4 Register Description.....260
19.4.1 TC_CNT（0x00）.....260
19.4.2 TC_PS（0x04）.....260
19.4.3 TC_DN（0x0C）.....261
19.4.4 TC_CCD0（0x14）.....261
19.4.5 TC_CCD1（0x18）.....261
19.4.6 TC_CCFG（0x1C）.....261
19.4.7 TC_CTRL（0x20）.....262
19.4.8 TC_CM0（0x24）.....263
19.4.9 TC_CM1（0x28）.....264
19.4.10 TC_IE（0x2C）.....265
19.4.11 TC_STA（0x30）.....265
19.4.12 TC_PWM_CFG（0x34）.....266
19.4.13 TC_PWM_CTL（0x38）.....267
19.4.14 TC_PWM_STA（0x3C）.....267
19.4.15 TC_PWM_CNT（0x40）.....268
19.4.16 TC_PWM_BADR（0x44）.....268
19.4.17 TC_PWM_DMA_LEN（0x48）.....269
19.4.18 TC_PWM_ADR（0x4C）.....269
19.5 Application Examples.....269
19.5.1 Interval Timing Event.....269

19.5.2 Square Wave Output 270
19.5.3 External Event Counting 271
19.5.4 External Clear and Start 271
19.5.5 Free-Running Mode 272
19.5.6 PWM Output 272
19.5.7 Pulse Width Measurement 274
19.5.8 Timer Cascade 275
19.5.9 Timer Parallel Connection 275

20 LSRX 276

20.1 INTRODUCTION 276
20.2 FEATURES 276
20.3 BASIC PRINCIPLES 276
20.3.1 Correction of Zero-Crossing Time Uploaded by LSTX by LSRX 276
20.3.2 Analysis of Frame Header Time Difference Between Channels 277
20.3.3 Handling of Phase Loss Issues 278
20.4 OVERALL IMPLEMENTATION SCHEME OF LSRX 286
20.5 REGISTER DESCRIPTION 288
20.6 SOFTWARE OPERATION FLOW 1
20.6.1 Calibration Waveform Output 1

21 LPUART 1

21.1 INTRODUCTION 1
21.2 FEATURES 1
21.3 BLOCK DIAGRAM 1
21.4 INTERFACE TIMING 1
21.5 FUNCTIONAL DESCRIPTION 1
21.5.1 FIFO Function of LPUART 1
21.5.2 Transmit Engine 1
21.5.3 Receive Engine 1
21.5.4 Baud Rate Generator 1
21.5.5 Parity Bit Configuration 1
21.5.6 LPUART Low-Power Wake-Up Mode 1
21.6 REGISTERS 1
21.6.1 LPUART_MODE（0x00） 1
21.6.2 LPUART_IE（0x04） 1
21.6.3 LPUART_STA（0x08） 1
21.6.4 LPUART_BAUD（0xC） 1
21.6.5 LPUART_TXD（0x10） 1
21.6.6 LPUART_RXD（0x14） 1
21.6.7 LPUART_DMR（0x18） 1
21.7 SOFTWARE CONFIGURATION FLOW AND APPLICATION NOTES 1
21.7.1 Transmit Flow 1
21.7.2 Receive Flow 1

22 7816 1

22.1 FEATURES 1
22.2 REGISTER DESCRIPTION 1
22.2.1 ISO7816x_CTL 1
22.2.2 ISO7816x_CLK 1
22.2.3 ISO7816x_BDDIV 1
22.2.4 ISO7816x_STAT 1
22.2.5 ISO7816x_DAT 1
22.3 SOFTWARE CONFIGURATION PROCESS AND APPLICATION NOTES 1

23 USART 1

23.1 INTRODUCTION 1
23.2 FEATURES 1
23.3 BLOCK DIAGRAM 1
23.4 PIN DEFINITIONS 1
23.5 FUNCTIONAL DESCRIPTION 1
23.5.1 Transmitter 1
23.5.2 Receiver 1
23.5.3 Baud Rate Generation 1
23.5.4 Automatic Baud Rate Detection 1
23.5.5 Parity Check Control 1
23.5.6 USART Synchronous Mode 1
23.5.7 Single-Wire Half-Duplex Communication 1
23.5.8 Continuous Communication Using DMA 1
23.5.9 Hardware Flow Control 1
23.5.10 Interrupts 1
23.6 REGISTERS 1
23.6.1 USARTx_TXDATA（0x00） 1
23.6.2 USARTx_RXDATA（0x04） 1
23.6.3 USARTx_TXCTRL（0x08） 1
23.6.4 USARTx_RXCTRL（0x0C） 1
23.6.5 USARTx_IE（0x10） 1
23.6.6 USARTx_STA（0x14） 1
23.6.7 USARTx_DIV（0x18） 1
23.6.8 USARTx_ SETUP（0x1C） 1
23.6.9 USARTx_ RX_SIZE（0x24） 1
23.6.10 USARTx_ TX_SIZE（0x30） 1
23.6.11 USARTx_ RX_IDLE（0x44） 1
23.6.12 USARTx_ RX_WM（0x48） 1
23.6.13 USARTx_ RX_FIFO_LEFT_ENTRY（0x4C） 1
23.6.14 USARTx _TX_FIFO_LEFT_ENTRY（0x50） 1
23.6.15 RX_LOW_LEVEL_CNT（0x5C） 1
23.7 SOFTWARE CONFIGURATION PROCESS AND APPLICATION NOTES 1
23.7.1 Asynchronous Communication 1

23.7.2 Synchronous Communication 1
23.7.3 Single-Wire Half-Duplex Communication 1
23.7.4 Closing Communication 1

24 IIC 1

24.1 Overview 1
24.2 Register Description 1
24.2.1 IIC_CTL 1
24.2.2 IIC_CLK 1
24.2.3 IIC_STAT 1
24.2.4 IIC_ADDR 1
24.2.5 IIC_DATA 1
24.3 Software Operation Flow 1
24.3.1 IIC as Master Communication 1
24.3.2 IIC as Slave Communication 1

25 SPI 1

25.1 Introduction 1
25.2 Features 1
25.3 Functional Description 1
25.4 Register Description 1
25.4.1 Register List 1
25.4.2 SPI_CTL（0x00） 1
25.4.3 SPI_STA（0x04） 1
25.4.4 SPI_TXDATA （0x08） 1
25.4.5 SPI_RXDATA （0x0C） 1
25.4.6 SPI_TXDFLT（0x10） 1
25.4.7 SPI_DMA_CTL（0x14） 1
25.4.8 SPI_DMA_TBADR（0x18） 1
25.4.9 SPI_DMA_RBADR（0x1C） 1
25.4.10 SPI_DMA_TLEN（0x20） 1
25.4.11 SPI_DMA_RLEN（0x24） 1
25.4.12 SPI_DMA_TADR（0x28） 1
25.4.13 SPI_DMA_RADR（0x2C） 1
25.4.14 SPI_DMA_IE（0x30） 1
25.4.15 SPI_DMA_FLG（0x34） 1
25.4.16 SPI_DLYSAMP（0x38） 1

26 QSPI 1

26.1 Introduction 1
26.2 Features 1
26.3 Signal Line Definition 1
26.4 Interface Timing 1
26.5 Functional Description 1

26.5.1 Command Sequence 1
26.5.2 Interface Modes 1
26.5.3 SDR/DDR Modes 1
26.5.4 Operation Modes 1
26.5.5 CS Mode Description 1
26.6 REGISTERS 1
26.6.1 QSPI_SCKDIV（0x00） 1
26.6.2 QSPI_SCKMODE（0x04） 1
26.6.3 SPI_SCKSAMPLE（0x08） 1
26.6.4 QSPI_FORCE（0x0C） 1
26.6.5 QSPI_CSID（0x10） 1
26.6.6 QSPI_CSDEF（0x14） 1
26.6.7 QSPI_CSMODE（0x18） 1
26.6.8 QSPI_VERSION（0x1C） 1
26.6.9 QSPI_ADDR_WRAP（0x20） 1
26.6.10 QSPI_BOUNDARY_CFG（0x24） 1
26.6.11 QSPI_DELAY0（0x28） 1
26.6.12 QSPI_DELAY1（0x2C） 1
26.6.13 QSPI_FIFO_NUM（0x30） 1
26.6.14 QSPI_TSIZE（0x34） 1
26.6.15 QSPI_RSIZE（0x38） 1
26.6.16 QSPI_FMT（0x40） 1
26.6.17 QSPI_TXDATA（0x48） 1
26.6.18 QSPI_RXDATA（0x4C） 1
26.6.19 QSPI_TX_MARK（0x50） 1
26.6.20 QSPI_RX_MARK（0x54） 1
26.6.21 QSPI_FCTRL（0x60，XIP） 1
26.6.22 QSPI_FFMT（0x64，XIP） 1
26.6.23 QSPI_IE（0x70） 1
26.6.24 QSPI_IP（0x74） 1
26.6.25 QSPI_FFMT1（0x78 ，XIP） 1
26.6.26 QSPI_STATUS（0x7C） 1
26.6.27 QSPI_SDR_SCKSAMPLE（0x80） 1
26.6.28 QSPI_CR（0x84） 1
26.7 SOFTWARE CONFIGURATION PROCESS AND APPLICATION NOTES 1
26.7.1 Register and DMA Modes 1
26.7.2 XIP Mode 1
26.7.3 Notes 1
27 GPIO 1
27.1 INTRODUCTION 1
27.2 FEATURES 1
27.3 BLOCK DIAGRAM 1
27.4 FUNCTIONAL DESCRIPTION 1

27.4.1 Functional Configuration Description 1
27.4.2 General I/O Description 1
27.4.3 Input Mode 1
27.4.4 Output Mode 1
27.4.5 Alternate Function Mode 1
27.4.6 Analog Mode 1
27.4.7 Forced Input Enable 1
27.4.8 Bitwise Operation 1
27.4.9 Secure Access 1
27.5 REGISTERS 1
27.5.1 Register List 1
27.5.2 GPIOx_MODE（0x00） 1
27.5.3 GPIOx_AEN（0x04） 1
27.5.4 GPIOx_ OTYPE（0x08） 1
27.5.5 GPIOx_OSPEED（0x0C） 1
27.5.6 GPIOx_FIE（0x10） 1
27.5.7 GPIOx_ITYPE（0x14） 1
27.5.8 GPIOx_UPDN（0x18） 1
27.5.9 GPIOx_IDATA（0x1C） 1
27.5.10 GPIOx_ODATA（0x20） 1
27.5.11 GPIOx_AFSEL0（0x24） 1
27.5.12 GPIOx_AFSEL1（0x28） 1
27.5.13 GPIOx_SET（0x2C） 1
27.5.14 GPIOx_CLR（0x30） 1
27.5.15 GPIOx_TG（0x34） 1
27.5.16 GPIO_DBG_KEY（0xB008 0070） 1
27.5.17 GPIO_DBG（0xB008 0074） 1
27.6 ALTERNATE FUNCTION LIST 1
27.7 SOFTWARE CONFIGURATION PROCESS AND APPLICATION NOTES 1
27.7.1 Configure and Enable the GPIO Module Clock According to the Content in the System Control Chapter 1
27.7.2 GPIO Configuration When the I/O is Used for JTAG Function 1
27.7.3 GPIO Configuration When the I/O is Used for General GPIO Function 1

28 LCD 1

28.1 INTRODUCTION 1
28.1.1 Scan Clock Frequency 1
28.1.2 Blink Modes 1
28.1.3 LCD Driving Waveforms 1
28.1.4 Bias Voltage 1
28.1.5 LCD BUFFER Mapping 1
28.2 REGISTER DESCRIPTION 1
28.2.1 LCD_CTL(0x00) 1
28.2.2 LCD_STATUS(0x04) 1

28.2.3 LCD_CLKDIV(0x08) 1
28.2.4 LCD_BLINK(0x0c) 1
28.2.5 LCD_PS(0x10) 1
28.2.6 LCD_RESCTL(0x14) 1
28.2.7 LCD_BUF[i](8-bit register) 1
28.3 LCD Operation Steps 1

29 KBI&INTC 1

29.1 Introduction 1
29.2 Features 1
29.3 Filtering 1
29.3.1 Filter Time 1
29.4 Registers 1
29.4.1 Register List 1
29.5 Software Configuration Process and Application Notes 1

30 IOCNT 1

30.1 Introduction 1
30.2 Features 1
30.3 Block Diagram 1
30.4 Functional Description 1
30.4.1 Input Signal Types 1
30.4.2 Pulse Counting 1
30.4.3 Pulse Division 1
30.4.4 Pulse Level Inversion Output 1
30.4.5 Pulse Forwarding Output Pulse Configuration 1
30.4.6 Software Pulse Special Command 1
30.5 Register Definition 1
30.5.1 IOCNT_CFG0~4（0x0~0x10） 1
30.5.2 IOCNT_OUT0~4（0x20~0x30） 1
30.5.3 IOCNT_CHNL（0x40） 1
30.5.4 IOCNT_CTRL（0x48） 1
30.5.5 IOCNT_CTL1（0x4C） 1
30.5.6 IOCNT_CMD（0x50） 1
30.5.7 IOCNT_REMD0~4（0x60~0x70） 1
30.6 Usage Flow 1
30.6.1 Internal Pulse Forwarding 1
30.6.2 High-Level Pulse Width Valid 1
30.6.3 Low-Level Pulse Width Valid 1
30.6.4 Application Notes 1

31 D2F 1

31.1 Introduction 1
31.2 Features 1

31.3 REGISTERS 1
31.3.1 HFCONST0/1/2（0x00~0x08） 1
31.3.2 D2F_CTRL（0x30） 1
31.3.3 D2F_ CFG（0x34） 1
31.3.4 D2F_CF_CFG（0x40） 1
31.3.5 D2F_IE（0x50） 1
31.3.6 D2F_IF（0x54） 1
31.3.7 D2F_NOLOAD（0x58） 1
31.3.8 D2F_P00~11（0x60~0x8C） 1
31.3.9 D2F_E00~11（0x120~0x14C） 1
31.3.10 D2F_StartTh00~02（0x1E0~0x1E8） 1
31.4 SOFTWARE CONFIGURATION PROCESS AND APPLICATION NOTES 1
31.4.1 Software Writing Integration 1
31.4.2 Application Notes 1

32 SECURE EXECUTION ENVIRONMENT (SPE) 1

32.1 FRAMEWORK 1
32.2 SEA ENGINE 1
32.2.1 SEA Atomic Operation Library Functions 1
32.2.2 Key Registration and Generation 1
32.3 KEYSTORE 1
32.4 SECURE BOOT 1

33 SEA 1

33.1 OVERVIEW 1
33.2 AES HARDWARE ACCELERATION UNIT 1
33.2.1 Features 1
33.2.2 Time Consumption Information 1
33.3 HASH HARDWARE ACCELERATION UNIT 1
33.3.1 Features 1
33.3.2 Time Consumption Information 1
33.4 TRNG TRUE RANDOM NUMBER GENERATOR 1
33.4.1 Features 1
33.4.2 Time Consumption Information 1
33.5 ECC HARDWARE ACCELERATION UNIT 1
33.5.1 Features 1
33.5.2 Time Consumption Information 1
33.6 RSA HARDWARE ACCELERATION UNIT 1
33.6.1 Features 1
33.6.2 Time Consumption Information 1
33.7 SOFTWARE CONFIGURATION PROCESS 1

34 AES 1

34.1 FEATURES 1

34.2 Time Consumption Information 1
34.3 Software Configuration Process 1

35 CRC 1

35.1 Overview 1
35.2 Features 1
35.3 Basic Principle 1
35.4 Function Description 1
35.4.1 Polynomial 1
35.4.2 Initial Remainder Value 1
35.4.3 Result XOR Value 1
35.4.4 DMA Function 1
35.4.5 Input Data Reversal 1
35.5 Registers 1
35.5.1 CRCx_DR（0x00）(x=0,1, same below) 1
35.5.2 CRCx_STA（0x04） 1
35.5.3 CRCx_CR（0x08） 1
35.5.4 CRCx_INIT（0x0C） 1
35.5.5 CRCx_POL（0x10） 1
35.5.6 CRCx_XOR（0x14） 1
35.5.7 CRC0_DMA_CTL（0x18） 1
35.6 Software Configuration Process and Application Description 1
35.6.1 Performing CRC Calculation through Software Operation 1
35.6.2 Performing Calculation Using DMA 1

36 DSP 1

36.1 Features 1
36.2 Basic Principles of the Calculation Engine 1
36.2.1 Single-Precision Floating-Point Representation 1
36.2.2 Special Values 1
36.2.3 Floating-Point Rounding Processing 1
36.2.4 IEEE754 Standard Rounding Modes 1
36.2.5 Rounding Modes of This Chip 1
36.2.6 Principle of Integer to Floating-Point Conversion 1
36.2.7 Principle of Floating-Point to Integer Conversion 1
36.2.8 Principle of Floating-Point Multiplication 1
36.2.9 Principle of Floating-Point Addition 1
36.2.10 Principle of Butterfly Operation 1
36.2.11 Principle of IIR Filter 1
36.2.12 Principle of FIR Filter 1
36.2.13 Linear Interpolation 1
36.2.14 Lagrange Interpolation 1
36.3 Operation Instructions 1
36.3.1 Integer to Floating-Point (int2fp/int2fp_dma) 1

36.3.2 Floating-Point to Integer (fp2int/fp2int_dma) 1
36.3.3 Floating-Point Multiplication (fp_mult) 1
36.3.4 Floating-Point Addition (fp_add) 1
36.3.5 Floating-Point Subtraction (fp_sub) 1
36.3.6 Floating-Point Multiply-Accumulate Operation (fp_mlad) 1
36.3.7 Floating-Point Butterfly Operation (Single) (btfy / btfy_dma) 1
36.3.8 Sine and Cosine Calculation (sin_cos) 1
36.3.9 Root Mean Square and Arctangent (fp_sqrt/ fp_atan) 1
36.3.10 Floating-Point Divider (fp_div) 1
36.3.11 IIR Filter 1
36.3.12 FIR Filter 1
36.3.13 Linear Interpolation 1
36.3.14 Lagrange Interpolation 1
36.4 IMPLEMENTATION DESCRIPTION 1
36.4.1 Integer to Floating-Point Conversion 1
36.4.2 Floating-Point to Integer Conversion 1
36.4.3 FFT Description 1
36.4.4 CORDIC Description 1
36.5 REGISTERS 1
36.5.1 Register List 1
36.5.2 MAC_CTL0（0x0） 1
36.5.3 MAC_CTL1（0x04） 1
36.5.4 MAC_CTL2（0x08） 1
36.5.5 MAC_IN0（0x0C） 1
36.5.6 MAC_IN1（0x10） 1
36.5.7 MAC_IN2（0x14） 1
36.5.8 MAC_IN3（0x18） 1
36.5.9 MAC_IN4（0x1C） 1
36.5.10 MAC_IN5（0x20） 1
36.5.11 MAC_OUT0（0x24） 1
36.5.12 MAC_OUT1（0x28） 1
36.5.13 MAC_OUT2（0x2C） 1
36.5.14 MAC_OUT3（0x30） 1
36.5.15 DIV_IN0（0x34） 1
36.5.16 DIV_IN1（0x38） 1
36.5.17 DIV_OUT0（0x3C） 1
36.5.18 DMA_SRBADR（0x40） 1
36.5.19 DMA_SIBADR（0x44） 1
36.5.20 DMA_PRBADR（0x48） 1
36.5.21 DMA_PIBADR（0x4C） 1
36.5.22 DMA_TRBADR（0x50） 1
36.5.23 DMA_TIBADR（0x54） 1
36.5.24 DMA_LEN（0x58） 1
36.5.25 DSP_IE（0x5C） 1

36.5.26 DSP_FLG（0x60） 1
36.5.27 ALU_STA0（0x64） 1
36.5.28 ALU_STA1（0x68） 1
36.5.29 CRD_CTL（0x6C） 1
36.5.30 CRD_XIN（0x70） 1
36.5.31 CRD_YIN（0x74） 1
36.5.32 CRD_AMP（0x78） 1
36.5.33 CRD_PHASE（0x7C） 1
36.5.34 CRD_ANGLE（0x80） 1
36.5.35 CRD_COSINE（0x84） 1
36.5.36 CRD_SINE（0x88） 1
36.5.37 CRD_IE（0x8C） 1
36.5.38 CRD_FLG（0x90） 1
36.5.39 INTP_LEN（0x94） 1
36.5.40 INTP_LOC（0x98） 1
36.5.41 INTP_STEP（0x9C） 1
36.5.42 PFMODE（0xA0） 1
36.6 SOFTWARE OPERATION PROCESS 1
36.6.1 Complete Solution Description 1
36.6.2 Operation Flow 1

37 RTC 1

37.1 OVERVIEW 1
37.2 FEATURES 1
37.3 REGISTERS 1
37.3.1 Register List 1
37.3.2 RTC Register Group 1
37.3.3 Quadratic Curve Compensation Register Group 1
37.3.4 Quartic Curve Compensation Register Group 1
37.3.5 Result Output Register Group 1
37.3.6 Secondary Compensation Register Group 1
37.3.7 RTC IO Control Register Group 1
37.3.8 Tamper Detection Freeze Time Register Group 1
37.3.9 User-defined Register Group 1
37.4 KEYOUT FUNCTION DESCRIPTION 1
37.5 RTC CLOCK READ/WRITE STEPS 1
37.6 RTC CALIBRATION STEPS 1
37.7 RTC TIMER OPERATION STEPS 1

38 ANALOG PERIPHERALS 1

38.1 OVERVIEW 1
38.2 SYSTEM BLOCK DIAGRAM 1
38.3 REGISTERS 1
38.3.1 Register List 1

38.3.2 AD_CTRL（0x0） 1
38.3.3 AD_START（0x4） 1
38.3.4 AD_STAT（0x8） 1
38.3.5 AD_DATA（0xC） 1
38.3.6 AD_CTRL1（0x18） 1
38.3.7 AD_DATA2（0x1C） 1
38.3.8 AD_CTRL3（0x20） 1
38.3.9 AD_DATA3（0x24） 1
38.3.10 ANA_RST_STAT（0x28） 1
38.3.11 ANA_PAD（0x3C） 1
38.3.12 ANA_RST（0x40） 1
38.4 ADC VOLTAGE DETECTION STEPS 1
38.5 VBAT VOLTAGE DETECTION 1
38.6 VBAT BOR RESET DETECTION 1

39 CMP&LVD 1

39.1 FEATURES 1
39.2 REGISTERS 1
39.2.1 LVD_CTL（0x00） 1
39.2.2 CMP_CTL（0x04） 1
39.2.3 LVDCMP_IE（0x08） 1
39.2.4 LVDCMP_IF（0x0C） 1
39.2.5 LVD_PWD（0x10） 1
39.2.6 CMP2_PWD（0x14） 1
39.2.7 CMP1_PWD（0x18） 1
39.3 SOFTWARE CONFIGURATION PROCESS AND APPLICATION DESCRIPTION 1

40 ADC0 1

40.1 INTRODUCTION 1
40.2 FEATURES 1
40.3 BLOCK DIAGRAM 1
40.4 PIN DEFINITION 1
40.5 CLOCK AND RESET 1
40.6 FUNCTIONAL DESCRIPTION 1
40.6.1 Switch Control 1
40.6.2 Sampling Rate 1
40.6.3 Channel Selection 1
40.6.4 Trigger Source Selection 1
40.6.5 Conversion Modes 1
40.6.6 Analog Watchdog 1
40.6.7 Programmable Sampling Time 1
40.6.8 Data Management 1
40.6.9 Calibration Function 1
40.6.10 Linkage with DAC 1

40.6.11 Dummy Conversion Function 1
40.6.12 Interrupts 1
40.7 REGISTERS 1
40.7.1 Register Mapping Table 1
40.7.2 Register Description 1
40.8 SOFTWARE CONFIGURATION PROCESS AND APPLICATION DESCRIPTION 1
40.8.1 Normal Usage Process of ADC0 1
41 DAC 1
41.1 INTRODUCTION 1
41.2 FEATURES 1
41.3 BLOCK DIAGRAM 1
41.4 PIN DEFINITION 1
41.5 CLOCK AND RESET 1
41.6 FUNCTIONAL DESCRIPTION 1
41.6.1 Data Alignment Format 1
41.6.2 DAC Output Voltage 1
41.6.3 DAC Output Buffer 1
41.6.4 DAC Output Modes 1
41.6.5 DMA Function 1
41.6.6 A/D Conversion Priority Function 1
41.6.7 Sample and Hold Mode 1
41.6.8 Interrupts 1
41.7 REGISTERS 1
41.7.1 Register Map Table 1
41.7.2 Register Description 1
41.8 SOFTWARE CONFIGURATION PROCESS AND APPLICATION DESCRIPTION 1
41.8.1 Continuous Mode Application Process 1
41.8.2 Software-Triggered Mode Application Process 1
41.8.3 Hardware-Triggered Mode Application Process 1
41.8.4 Output Buffer Calibration Process 1
41.8.5 Other Application Notes 1
42 DEBUG SUPPORT (DBG) 1
42.1 OVERVIEW 1
42.2 JTAG TAP CONTROLLER 1
42.3 DEBUG ARCHITECTURE 1
42.3.1 Instruction Register 1
42.3.2 Data Register 1
43 BOOTLOADER 1
43.1 OVERVIEW 1
43.2 NVR CONFIGURATION 1
43.3 IN-SYSTEM PROGRAMMING (ISP) 1

43.3.1 ISP State 1
43.3.2 Protection Levels 1
43.3.3 ISP Process 1
43.4 IN-APPLICATION PROGRAMMING (IAP) 1
43.4.1 IAP Return Values 1
43.4.2 IAP Commands 1
43.5 FLASH BOOT 1
43.5.1 Vector Table Checksum Verification 1
43.5.2 FLASH Signature Verification 1
43.6 BOOTLOADER RESOURCE USAGE 1
44 PACKAGE DIMENSIONS AND SOLDERING CONDITIONS 1
44.1 PACKAGE DIMENSIONS 1
44.1.1 LQFP100L 1
44.2 REFLOW OVEN TEMPERATURE PROFILE SETTINGS 1

1 Overview

1.1 Product Positioning

- Meet the functional safety certification requirements of electric meters, and can pass safety certifications such as CMVP (Cryptographic Module Validation Program) and PSA (Platform Security Architecture);
- Meet the independent legal metering requirements of electric meters, ensuring physical isolation between metering and management functions while maintaining sufficient flexibility. Any peripheral can work independently on different core buses to meet the needs of different customers and markets;
- MAKSA RISC-V dual-core series MCUs are hardware-compatible with MAKSA's AMR core versions V2/V3 (backward compatibility). The FLASH capacity is expanded to 1MB, and the RAM capacity is expanded to 160KB; it also supports FLASH and RAM expansion, enabling larger FLASH and RAM capacities;
- It can be used as either a dual-core MCU or a single-core MCU. The application core (Core1) has a maximum operating frequency of 58.9824MHz, with performance equivalent to that of ARM Cortex-M33; the security core (Core0) has a maximum operating frequency of 29.4912MHz, with performance equivalent to that of ARM Cortex-M23;
- The chip is equipped with an independent secure encryption core. Data security is crucial—the built-in security engine provides a reliable data security storage management mechanism to ensure that sensitive data is encrypted and stored, thus guaranteeing data storage security. The security core is located in the SPE (Secure Processing Environment), while the application core is in the NSPE (Non-Secure Processing Environment). The registers and cache data of the security core cannot be accessed by the application core, achieving physical isolation from the NSPE.

1.2 Product Features

Basic Features:

- High integration: 32-bit RISC-V dual-core MCU + 1MB embedded FLASH + 160KB embedded SRAM + independent power-supply hardware temperature-compensated RTC + LCD controller + AES/ECC hardware encryption meeting functional safety certification requirements + DSP accelerator;
- Core1 (application core): Performance equivalent to ARM Cortex-M33, with a maximum operating frequency of 58.9824MHz;
- Core0 (security core): Complies with the TEE (Trusted Execution Environment) architecture, with performance equivalent to ARM Cortex-M23;
- For single-core applications:If the operating frequency is sufficient, Core0 is recommended, as it is activated by default; if Core0 is not selected, MAKSA's secure boot firmware built in the FLASH NVR (Non-Volatile Register) space must be used to disable Core0;If a higher operating frequency and FPU (Floating-Point Unit) function are required for single-core applications, Core1 can be selected; when using Core1, a special secure boot code (running on Core0) must be programmed via a programmer;
- For dual-core applications: Customer application code is divided into non-upgradable secure code (recommended for secure encryption and legal metering, running on Core0) and upgradable application code (recommended for management/communication functions, running on Core1);
- Supports QSPI interface for FLASH expansion, with program space expandable to 4MB, reserving sufficient program space for future use;
- Supports QSPI interface for SRAM expansion, with SRAM expandable to 512KB, reserving sufficient SRAM space for future use;

- Wide voltage range:
  - Under typical conditions, the MCU works normally within a voltage range of 2.3V~5.5V; it can be configured to operate at a low speed within a voltage range of 1.8V~5.5V;
  - Under typical conditions, the RTC maintains normal temperature compensation and timing within a voltage range of 2.3V~5.5V;
  - Under typical conditions, the RTC maintains fixed temperature compensation and timing within a voltage range of 1.2V~5.5V;
- High performance:
  - With a 32.768KHz crystal oscillator (low-frequency) and an external high-frequency crystal oscillator, the maximum operating frequency of the large core (Core1) can reach 58.9824MHz (optional frequencies: 32.768KHz, 1.8432MHz, 7.3728MHz, 14.7456MHz, 29.4912MHz, 58.9824MHz), and the maximum operating frequency of the small core (Core0) is 29.491MHz;
  - The internal high-frequency RCH (Internal RC Oscillator) can operate at a maximum of 59MHz, with a full-temperature range accuracy of ±1%, and can be used as a backup clock;
- Low power consumption:
  - The MCU subsystem consumes less than 18μA when operating at 32.768KHz;
  - The overall chip power consumption in sleep mode is approximately 8μA (RTC with automatic temperature compensation; all RAM retained; MCU and digital peripherals not powered off; interrupt wake-up supported);
  - The typical power consumption of the VBAT domain is approximately 1.8μA;
  - The LCD power consumption is approximately 5μA, and the overall chip power consumption with continuous display is 13μA (resistor string voltage division mode, including 6COM display power consumption);
  - The LCD power consumption is approximately 10μA, and the overall chip power consumption with continuous display is 18μA (charge pump mode, including 8COM display power consumption);
- High precision:
  - The second pulse error of the RTC is less than ±5ppm within the full temperature range (-40°C~85°C), with a minimum calibration scale of 0.0339ppm;
- Processor-related:
  - RISC-V dual-core: The large core (Core1) is the application core, with performance equivalent to ARM Cortex-M33; the small core (Core0) is the security core, with performance equivalent to ARM Cortex-M23;
  - 1MB embedded FLASH memory, supporting QSPI interface for FLASH expansion up to 4MB, with reserved space for co-packaged FLASH chips as expanded program space;
  - 160KB embedded SRAM, which is non-volatile; supports QSPI interface for SRAM expansion up to 512KB, with reserved space for co-packaged SRAM chips as expanded RAM space;
- Built-in system timer;
- Supports multiple wake-up methods such as external interrupts;
- Provides a complete integrated development software and hardware environment;
- DSP (Digital Signal Processor)
  - Supports conversion between integers and floating-point numbers;
  - Supports floating-point addition, subtraction, multiplication, and division;
  - Supports single butterfly operation (complex numbers) and DMA-continuous butterfly operations;
  - Supports the entire base-2 FFT (Fast Fourier Transform) process, with optional point counts: 64, 128, 256,

512, 1024;

- Supports automatic bit-reverse data transfer, with optional point counts: 4, 8, 16, 32, 64, 128, 256, 512, 1024;
- Supports sine and cosine calculation;
- Supports root mean square (RMS) calculation;
- Supports arctangent calculation;
- Supports single IIR (Infinite Impulse Response) filter calculation and DMA-based IIR filter calculation;
- Supports FIR (Finite Impulse Response) filter operation;
- Supports linear interpolation operation;
- Supports Lagrange interpolation operation;

- RTC：
- Hardware automatic temperature compensation, meeting standard accuracy and power consumption requirements, with no need for MCU participation in temperature compensation;
- Temperature sensor: Temperature measurement accuracy of ±1°C within the range of -40°C~85°C;
- The RTC calendar and automatic temperature compensation circuit are independently powered by the VBAT pin;
- The typical power consumption of the VBAT domain is better than 1.8μA;
- The temperature compensation activation time is approximately 2ms, with a typical power consumption of 250μA during temperature measurement; if activated once every 30 seconds, the average power consumption is 0.016μA;
- LCD：
  - LQFP128 package: 4×40, 6×38, 8×36;
  - LQFP100 package: 4×34, 6×32, 8×30;
  - Supports optional ChargePUMP and built-in resistor string voltage division modes, with hardware compatibility between the two modes; if lower display power consumption is required, the resistor string voltage division mode can be selected;
  - The power consumption of the LCD module is better than 5μA (resistor string voltage division solution);
- D2F (Data to Frequency) Electric Energy Integration Unit:
  - Provides 12 D2F integrators, 3 of which support pulse output; suitable for special application scenarios such as three-phase manganin meters and single-phase dual-manganin meters;
- IOCNT (Pulse Forwarding) Function
  - Supports 5 channels; capable of forwarding pulses (input from GPIO or internal pulses) from GPIO after operations such as frequency division and level inversion; also records the number of input and output pulses, which is used for counting and forwarding electric energy pulses of metering chips;
- M2M (Memory-to-Memory) Data Transfer Unit:1 M2M module, enabling automatic data transfer between memories;
- SEA (Security Engine Accelerator) Encryption Unit
  - Hardware true random number generator (TRNG), complying with the U.S. NIST FIPS 140-2 standard;
  - AES hardware acceleration unit;
  - ECC (Elliptic Curve Cryptography) hardware acceleration unit;
  - RSA hardware acceleration unit;
  - HASH algorithm hardware acceleration unit;
- An independent AES encryption unit is provided in addition to the SEA unit;
- CRC (Cyclic Redundancy Check):2 CRC modules, one of which supports DMA, enabling hardware-automatic

CRC check calculation for data;

Other Peripherals

- High-speed GPIO, supporting interfaces with peripherals of different voltages;
- 12-bit low-speed ADC: Time-sharing multiplexing for temperature sensor/battery voltage detection/general ADC;
- LVD (Low Voltage Detector): Detects chip power supply voltage and external voltage;
- Two low-power comparators (CMP): Detect external voltage, used for power-on/off detection of electric meters;
- High-speed SAR ADC (ADC0): 12 channels with time-sharing multiplexing;
- DAC (Digital-to-Analog Converter): 2 output channels;
- Extended timers: 2 32-bit multi-function timers (TC), 4 32-bit simple timers (SIMPTC), 1 32-bit simple timer (SYS_TC);
- UART: 8 channels, supporting automatic baud rate detection, infrared modulation (also supported at 32.768KHz main frequency), UART wake-up, level inversion, and DMA;
- LPUART (Low-Power UART): 1 channel, supporting low-power serial communication;
- USART: 2 channels, supporting DMA;
- 7816 interfaces: 2 channels, which can be used independently;
- I2C: 2 channels;
- SPI: 4 channels, supporting DMA;
- QSPI: 2 channels, supporting DMA;
- Hardware watchdog timer (WDT);
- Key and external pin interrupts: 32 channels, with pin multiplexing.

1.3 System Block Diagra

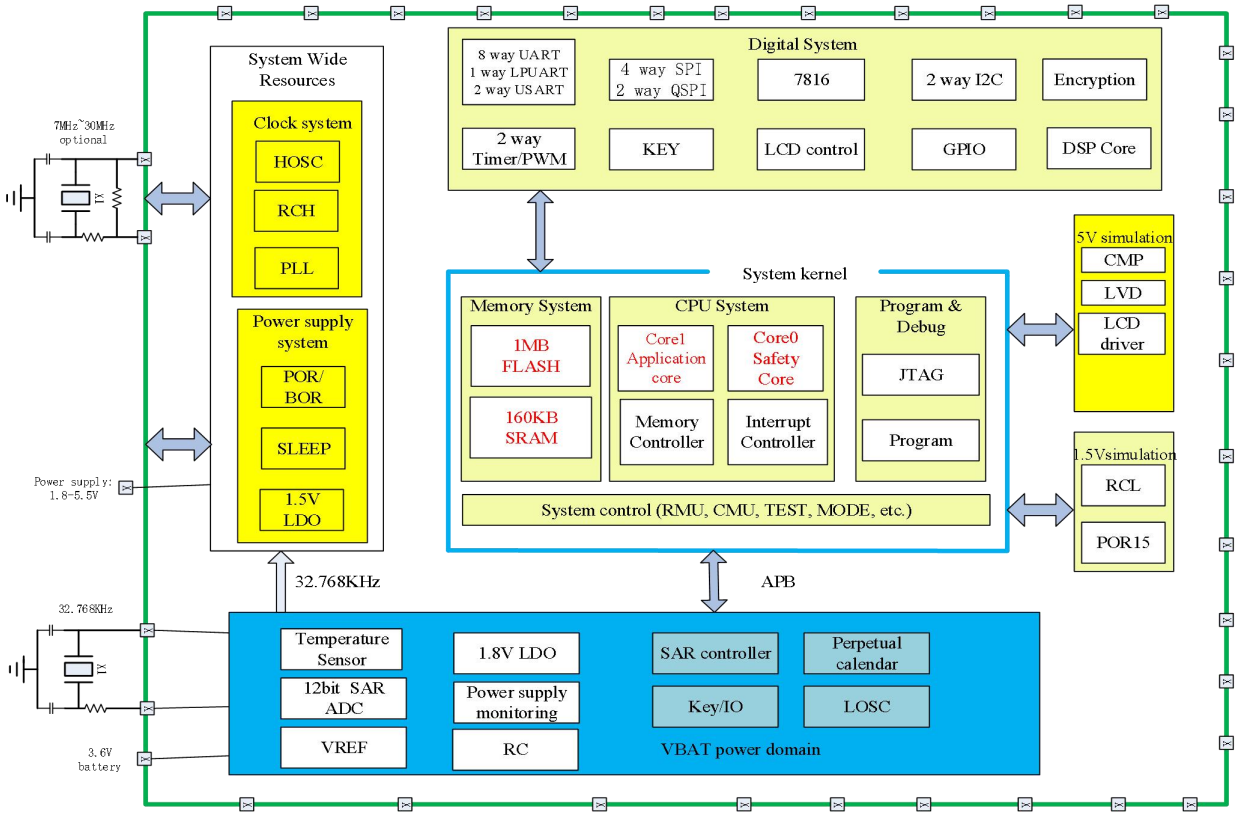


Figure 1-1 Block Diagram of the Dual-Core MCU

1.4 Package and Model Selection

Four models with three package types:

- LQFP100L: MKC40AT

1.5 Pin Arrangement

1.5.1 MKC40AT LQFP100L Pin Diagram

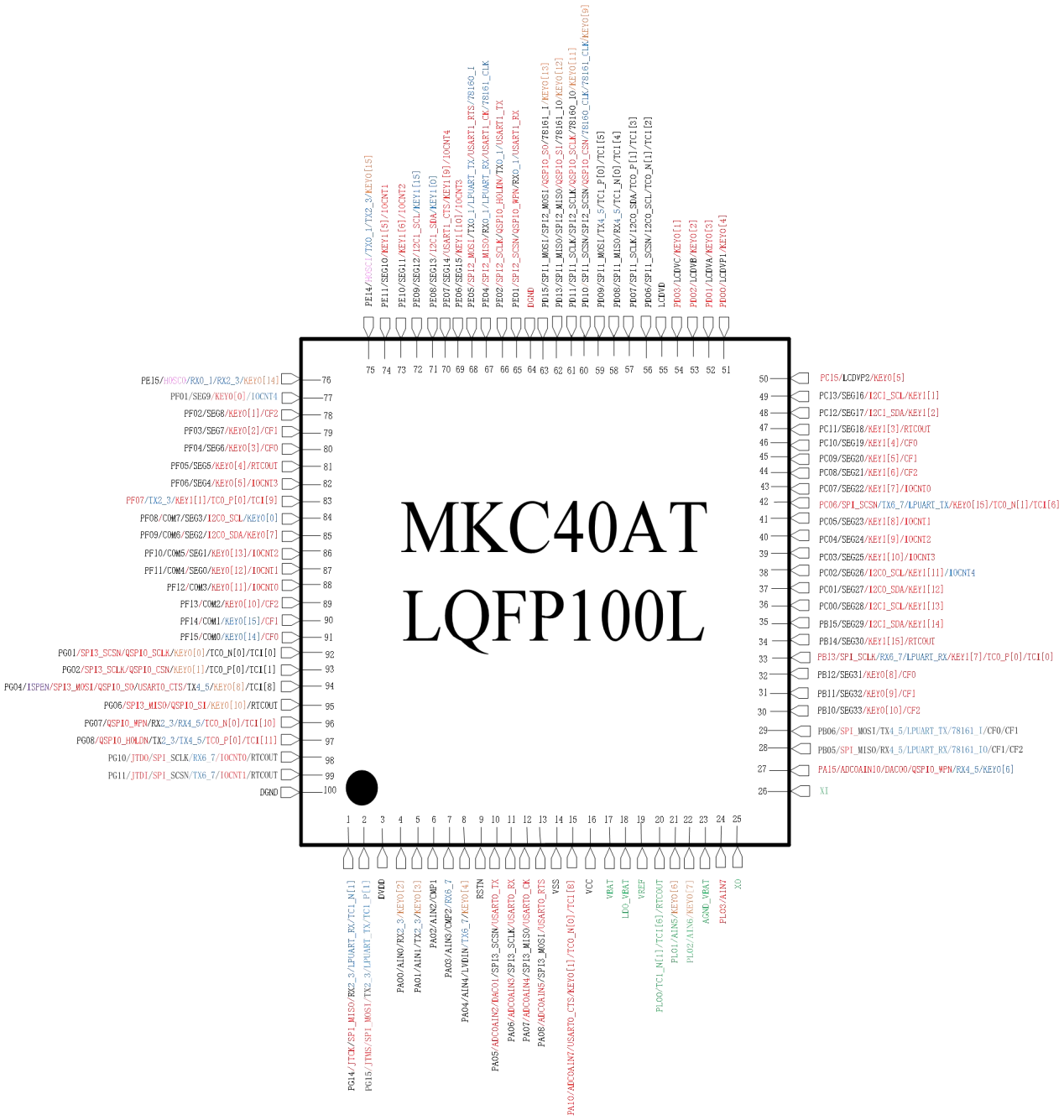


Figure 1-3 Pin Arrangement Diagram of MKC40AT

1.5.2 Pin Type Description

	Cell Type				Input Option				LCD Option	Output option		Description
Type	Analog	Bidirectional	Input	Output	Pull-Up	Pull-Down	Schmitt	TTL/CMOS	SEG/COM	Open-Drain	Drive	
PAB	√	√			√			√		√	√	
PABV	√	√			√			√		√		
PABV_A3	√	√			√			√		√		Drive:3mA
PAB_CMP	√	√			√			√		√	√	
PB		√			√	√		√		√	√	
PB_B		√			√	√		√		√	√	
PB_BI		√			√	√		√		√	√	
PB37		√			√	√		√		√	√	HOSCI
PB36_BI		√			√	√		√		√	√	HOSCO
PBC		√				√		√			√	Output: True open-drain output; Drive: Sink current configurable
PBL		√				√		√	√		√	Output: True open-drain output; Drive: Sink current configurable
PBL_B		√				√		√	√		√	Output: True open-drain output; Drive: Sink current configurable
PBL_BI		√			√	√		√	√		√	Output: True open-drain output; Drive: Sink current configurable
PLCDVD												LCDVD power pad
PIU			√		√		√					RSTN
PID			√			√	√					TEST

1.5.3 Pin Definition Description

MKC40AT	——	——	Pin Function	Type	Descrptions
LQFP100L	LQFP128L	LQFP64L			
Pin Number					
1	1	63	PG14	PB	GPIO
			JTCK		JTCK (JTAG Clock)
			SPI_MISO		SPIx master input slave output (x=0-3)
			RX2		UART2 receive
			RX3		UART3 receive
			LPUART_RX		LPUART Receive
			TC1_N[1]		Timer 1 Channel 1 compares the negative output
2	2	64	PG15	PB_BI	GPIO
			JTMS		JTAG mode selection
			SPI_MOSI		SPIx master output slave input (x=0-3)
			TX2		UART2 send
			TX3		UART3 send
			LPUART_TX		LPUART send
			TC1_P[1]		Timer 1 Channel 1 compares the forward output
NC	3	NC	PH00	PAB_CMP	GPIO
			ADC0AIN0		ADC0 external channel 0 input
			RX6		UART6 receive
			RX7		UART7 receive
			KEY0[3]		Key 0[3](originally KEY3)
			TC0_P[1]		Timer 0 channel 1 compares the forward output
			TCIN[3]		Timer input channel 3
			CF1		D2F1 output
NC	4	NC	PH01	PAB_CMP	GPIO
			ADC0AIN1		ADC0 external channel 1 input
			TX6		UART6 send
			TX7		UART7 send
			KEY0[11]		Key 0[11](originally INT3)
			TC0_N[1]		Timer 0 channel 1 compares the negative output
			TCIN[10]		Timer input channel 10
			CF0		D2F0 output
3	5	1	LDO15	power	The built-in 1.5VLDO output supplies power to the 1.5V digital domain of the chip. An external 1μf capacitor should be connected in parallel with a 0.1μf capacitor for decoupling
4	6	2	PA00	PAB	GPIO
			AIN0		SAR_ADC external channel 0 input
			RX2		UART2 receive

			RX3		UART3 receive
			KEY0[2]		Key 0[2](originally KEY2)
5	7	3	PA01	PAB	GPIO
			AIN1		SAR_ADC external channel 1 input
			TX2		UART2 send
			TX3		UART3 send
			KEY0[3]		Key 0[3] (originally KEY3)
6	8	NC	PA02	PAB_CMP	GPIO
			AIN2		SAR_ADC external channel 2 input
			CMP1		Comparator input 1
7	9	4	PA03	PAB_CMP	GPIO
			AIN3		SAR_ADC external channel 3 input
			CMP2		Comparator input 2
			RX6		UART6 receive
			RX7		UART7 receive
8	10	NC	PA04	PAB	GPIO
			AIN4		SAR_ADC external channel 4 input
			LVDIN		Power failure detection input
			TX6		UART6 Send
			TX7		UART7 Send
			KEY0[4]		Key 0[4](originally KEY4)
9	11	5	RSTN	PIU	PIN reset input, with an internal 10K pull-up resistor
10	12	6	PA05	PAB_CMP	GPIO
			ADC0AIN2		ADC0 external channel 2 input
			DACO1		DAC output channel 1
			SPI3_SCSN		Selection of SPI3 pieces
			USART0_TX		USART0 send
11	13	7	PA06	PAB_CMP	GPIO
			ADC0AIN3		ADC0 external channel 3 input
			SPI3_SCLK		SPI3 clock
			USART0_RX		USART0 receive
12	14	8	PA07	PAB_CMP	GPIO
			ADC0AIN4		ADC0 external channel 4 input
			SPI3_MISO		SPI3 master input slave output
			USART0_CK		USART0 synchronous mode clock
13	15	9	PA08	PAB_CMP	GPIO
			ADC0AIN5		ADC0 external channel 5 input
			SPI3_MOSI		The SPI3 master output and slave input
			USART0_RTS		USART0 hardware Flow Control (Sending requests)
NC	16	NC	PA09	PAB_CMP	GPIO
			ADC0AIN6		ADC0 external channel 6 input
			KEY0[2]		Key 0[2](originally KEY2)

			TC0_P[0]		The timer 0 channel 0 compares the forward output
14	17	10	VSS	GND	Ground
15	18	NC	PA10	PAB_CMP	GPIO
			ADC0AIN7		ADC0 external channel 7 input
			USART0_CTS		USART0 hardware flow control (send permission)
			KEY0[1]		Key 0[1](originally KEY1)
			TC0_N[0]		Timer 0 channel 0 is relatively negative output
			TCIN[8]		Timer input channel 8
16	19	11	VCC	Power	For the main power supply input, a 4.7uf capacitor should be externally connected and a 0.4uf capacitor in parallel for decoupling
17	20	12	VBAT	Power	The 3.6V battery or supercapacitor input pin only supplies power to the RTC section and also serves as the input for the SAR_ADC. When measuring this pin, there are two 300K resistors inside for voltage division, and a 0.5x gain is used. It is recommended to connect an external RC filter (R=10Ω, C=1μF).
18	21	13	LDO_VBAT	Power	The VBAT domain has a 2V LDO output, and an external 0.22μF capacitor needs to be connected
19	22	14	VREF	Power	Reference input for SAR_ADC,it is recommended to connect an external 0.22μF capacitor.
20	23	NC	PL00	PABV	GPIO
			TCIN[6]		Timer input channel 6
			TC1_N[1]		Timer 1 Channel 1 compares the negative output
			RTCOUT		RTC output
21	24	NC	PL01	PABV	GPIO
			AIN5		SAR_ADC external channel 5 input
			KEY0[6]		Key 0[6](originally KEY6)
22	25	NC	PL02	PABV_A3	GPIO
			AIN6		SAR_ADC external channel 6 input
			KEY0[7]		Key 0[7](original KEY7)
23	NC	15	AGND_VBAT	GND	VBAT domain simulation site
24	26	NC	PL03	PABV	GPIO
			AIN7		SAR_ADC external channel 7 input
25	27	16	XO	Clock	32.768KHz passive crystal oscillator output
26	28	17	XI	Clock	32.768KHz passive crystal oscillator input
NC	29	NC	PA11	PAB	GPIO
			AIN8		SAR_ADC external channel 8 input
			SPI_SCSN		SPIx film selection (x=0-3)
			QSPI0_CSN		QSPI0 chip selection
			KEY0[4]		Key 0[4](originally KEY4)
			TC1_N[0]		Timer 1 channel 0 is relatively negative output

			TCIN[4]		Timer input channel 4
NC	30	NC	PA12	PAB	GPIO
			AIN9		SAR_ADC external channel 9 input
			SPI_SCLK		SPIx clock (x=0-3)
			QSPI0_SCLK		QSPI0 clock
			KEY0[5]		Key 0[5](originally KEY5)
			TC1_P[0]		Timer 1 channel 0 compares the forward output
			TCIN[5]		Timer input channel 5
NC	31	NC	PA13	PAB_CMP	GPIO
			ADC0AIN8		ADC0 external channel 8 input
			SPI_MISO		SPIx master input slave output (x=0-3)
			QSPI0_SI (IO1)		QSPI0 serial input
			KEY0[6]		Key 0[6](originally KEY6)
			TC1_N[1]		Timer 1 Channel 1 compares the negative output
			TCIN[6]		Timer input channel 6
NC	32	18	PA14	PAB_CMP	GPIO
			ADC0AIN9		ADC0 external channel 9 input
			SPI_MOSI		SPIx master output slave input (x=0-3)
			QSPI0_S0 (IO0)		QSPI0 serial output
			KEY0[7]		Key 0[7](formerly KEY7)
			TC1_P[1]		Timer 1 Channel 1 compares the forward output
			TCIN[7]		Timer input channel 7
27	33	NC	PA15	PAB_CMP	GPIO
			ADC0AIN10		ADC0 external channel 10 input
			DACO0		DAC output channel 0
			QSPI0_WPN (IO2)		QSPI0 Write protection (Effective at low levels)
			RX4		UART4 receive
			RX5		UART5 receive
			KEY0[6]		Key 0[6](originally KEY6)
NC	34	19	PB00	PB_B	GPIO
			SPI0_SCSN		SPI0 chip selection
			QSPI0_CSN		QSPI0 chip selection
			TX4		Sent by UART4
			TX5		Sent by UART5
			78160_CLK		78160 clock
NC	35	20	PB01	PB	GPIO
			SPI0_SCLK		SPI0 clock
			QSPI0_SCLK		QSPI0 clock
			TX4		UART4 Send

			TX5		UART5 Send
			78160_IO		78160 two-way channel
NC	36	21	PB02	PB	GPIO
			SPI0_MISO		SPI0 host input slave output
			QSPI0_SI (IO0)		QSPI0 serial input
			RX4		UART4 receive
			RX5		UART5 receive
			78160_I		78160 one-way input channel
NC	37	22	PB03	PB	GPIO
			SPI0_MOSI		SPI0 host output slave input
			QSPI0_SO (IO0)		QSPI0 serial output
			TX6		UART6 Send
			TX7		UART7 Send
			78161_CLK		78161 clock
NC	38	23	PB04	PAB_CMP	GPIO
			ADC0AIN11		ADC0 external channel 11 input
			QSPI0_HOLDN (IO3)		QSPI0 retention (low effectiveness)
			RX6		UART6 receive
			RX7		UART7 receive
			KEY1[0]		Press Key 1[0]
28	39	24	PB05	PB_B	GPIO
			SPI_MISO		SPIx master input slave output (x=0-3)
			RX4		UART4 receive
			RX5		UART5 receive
			LPUART_RX		LPUART Receive
			78161_IO		78161 Two-way channel
			CF1		D2F1 output
			CF2		D2F2 output
29	40	25	PB06	PB	GPIO
			SPI_MOSI		SPIx master output slave input (x=0-3)
			TX4		UART4 Send
			TX5		UART5 Send
			LPUART_TX		LPUART send
			78161_I		78161 single-phase input channel
			CF0		D2F0 output
			CF1		D2F1 output
NC	41	NC	PB07	PB	GPIO
			TX6		UART6 send
			TX7		UART7 send

			KEY1[1]		Key 1[1]
			TC0_N[0]		Timer 0 channel 0 is relatively negative output
NC	42	NC	PB08	PBL	GPIO
			SEG35		LCD drives the SEG port
			I2C0_SDA		I2C0 data
			KEY0[12]		Key 0[12](originally INT4)
			IOCNT4		IOCNT pulse output channel 4
NC	43	NC	PB09	PBL	GPIO
			SEG34		LCD drives the SEG port
			I2C0_SCL		I2C0 clock
			KEY0[11]		Key 0[11](originally INT3)
			IOCNT0		IOCNT pulse output channel 0
30	44	NC	PB10	PBL_B	GPIO
			SEG33		LCD drives the SEG port
			KEY0[10]		Key 0[10](originally INT2)
			CF2		D2F2 output
31	45	NC	PB11	PBL	GPIO
			SEG32		LCD drives the SEG port
			KEY0[9]		Key 0[9](originally INT1)
			CF1		D2F1 output
32	46	NC	PB12	PBL	GPIO
			SEG31		LCD drives the SEG port
			KEY0[8]		Key 0[8](formerly INT0)
			CF0		D2F0 output
33	47	26	PB13	PB	GPIO
			SPI_SCLK		SPI clock (x=0-3)
			RX6		UART6 Receive
			RX7		UART7 Receive
			LPUART_RX		LPUART receive
			KEY1[7]		Key 1[7]
			TC0_P[0]		The timer 0 channel 0 compares the forward output
			TCIN[0]		Timer input channel 0
34	48	NC	PB14	PBL	GPIO
			SEG30		LCD drives the SEG port
			KEY1[15]		Key 1[15]
			RTCOUT		RTC output
35	49	NC	PB15	PBL	GPIO
			SEG29		LCD drives the SEG port
			I2C1_SDA		I2C1 data
			KEY1[14]		Key 1[14]
36	50	NC	PC00	PBL	GPIO
			SEG28		LCD drives the SEG port

			I2C1_SCL		I2C1 clock
			KEY1[13]		Key 1[13]
37	51	NC	PC01	PBL_B	GPIO
			SEG27		LCD drives the SEG port
			I2C0_SDA		I2C0 data
			KEY1[12]		Key 1[12]
38	52	NC	PC02	PBL	GPIO
			SEG26		LCD drives the SEG port
			I2C0_SCL		I2C0 clock
			KEY1[11]		Key 1[11]
			IOCNT4		IOCNT pulse output channel 4
39	53	NC	PC03	PBL	GPIO
			SEG25		LCD drives the SEG port
			KEY1[10]		Key 1[10]
			IOCNT3		IOCNT pulse output channel 3
40	54	27	PC04	PBL	GPIO
			SEG24		LCD drives the SEG port
			KEY1[9]		Key 1[9]
			IOCNT2		IOCNT pulse output channel 2
41	55	28	PC05	PBL	GPIO
			SEG23		LCD drives the SEG port
			KEY1[8]		Key 1[8]
			IOCNT1		IOCNT pulse output Channel 1
42	56	29	PC06	PB	GPIO
			SPI_SCSN		SPIx film selection (x=0-3)
			TX6		UART6 Send
			TX7		UART7 Send
			LPUART_TX		LPUART send
			KEY0[15]		Key 0[15](originally INT7)
			TC0_N[1]		Timer 0 channel 1 compares the negative output
			TCIN[1]		Timer input channel 1
43	57	30	PC07	PBL_B	GPIO
			SEG22		LCD drives the SEG port
			KEY1[7]		Key 1[7]
			IOCNT0		IOCNT pulse output channel 0
44	58	31	PC08	PBL	GPIO
			SEG21		LCD drives the SEG port
			KEY1[6]		Key 1[6]
			CF2		D2F2 output
45	59	32	PC09	PBL_B	GPIO
			SEG20		LCD drives the SEG port
			KEY1[5]		Key 1[5]

			CF1		D2F1 output
46	60	NC	PC10	PBL	GPIO
			SEG19		LCD drives the SEG port
			KEY1[4]		Key 1[4]
			CF0		D2F0 output
47	61	NC	PC11	PBL	GPIO
			SEG18		LCD drives the SEG port
			KEY1[3]		Key 1[3]
			RTCOUT		RTC output
48	62	NC	PC12	PBL	GPIO
			SEG17		LCD drives the SEG port
			I2C1_SDA		I2C1 data
			KEY1[2]		Key 1[2]
49	63	NC	PC13	PBL	GPIO
			SEG16		LCD drives the SEG port
			I2C1_SCL		I2C1 clock
			KEY1[1]		Key 1[1]
NC	64	NC	DGND	DGND	Ground
50	65	NC	PC15	PBC	GPIO
			LCDVP2		For the analog output, a 100nF capacitor should be connected between LCDVP2 and LCDVP1. If the LCD uses the resistor voltage division method, it can be left suspended.
			KEY0[5]		Key 0[5](originally KEY5)
51	66	NC	PD00	PBC	GPIO
			LCDVP1		For the analog output, a 100nF capacitor should be connected between LCDVP2 and LCDVP1. If the LCD uses the resistor voltage division method, it can be left suspended.
			KEY0[4]		Key 0[4](originally KEY4)
52	67	NC	PD01	PBC	GPIO
			LCDVA		For the LCD voltage output, an external 470nF capacitor is required. Whether a charge pump or a resistor voltage divider method is used, this capacitor needs to be externally connected.
			KEY0[3]		Key 0[3](originally KEY3)
53	68	NC	PD02	PBC	GPIO
			LCDVB		For the LCD voltage output, an external 470nF capacitor is required.
			KEY0[2]		Key 0[2](originally KEY2)
54	69	NC	PD03	PBC	GPIO

			LCDVC		For the LCD voltage output, an external 470nF capacitor is required.
			KEY0[1]		Key 0[1](originally KEY1)
55	70	NC	LCDVD	PLCDVD	For the LCD voltage output, an external 470nF capacitor is required. The default output voltage upon power-on is VCC.
NC	71	NC	PD05	PB_B	GPIO
			TX0		UART0 send
			TX1		UART1 Send
			USART1_TX		USART1 send
			KEY1[15]		Key 1[15]
			TC1_N[0]		Timer 1 channel 0 is relatively negative output
56	72	33	PD06	PB	GPIO
			SPI1_SCSN		SPI1 piece selection
			I2C0_SCL		I2C0 clock
			TCIN[2]		Timer input channel 2
			TC0_N[1]		Timer 0 channel 1 compares the negative output
57	73	34	PD07	PB	GPIO
			SPI1_SCLK		SPI1 clock
			I2C0_SDA		I2C0 data
			TCIN[3]		Timer input channel 3
			TC0_P[1]		Timer 0 channel 1 compares the forward output
58	74	35	PD08	PB	GPIO
			SPI1_MISO		SPI1 master input slave output
			RX4		UART4 Receive
			RX5		UART5 Receive
			TC1_N[0]		Timer 1 channel 0 is relatively negative output
			TCIN[4]		Timer input channel 4
59	75	36	PD09	PB	GPIO
			SPI1_MOSI		The SPI1 master output and slave input
			TX4		UART4 Send
			TX5		UART5 Send
			TC1_P[0]		Timer 1 channel 0 compares the forward output
			TCIN[5]		Timer input channel 5
60	76	37	PD10	PB_B	GPIO
			SPI1_SCSN		SPI1 Chip Select
			SPI2_SCSN		SPI2 Chip Select
			QSPI0_CSN		QSPI0 Chip Select(Active Low)
			78160_CLK		78160 Clock
			78161_CLK		78161 Clock
			KEY0[9]		Key 0[11], former INT3
61	77	38	PD11	PB	GPIO

			SPI1_SCLK		SPI1 Clock
			SPI2_SCLK		SPI2 Clock
			QSPI0_SCLK		QSPI0 Clock
			78160_IO		78160 Bidirectional Channel
			KEY0[11]		Key 0[11] (formerly INT3)
NC	78	39	PD12	PB	GPIO
			RX0		UART0 Receive
			RX1		UART1 Receive
			USART1_RX		USART1 Receive
			KEY1[14]		Key 1[14]
			TC1_P[0]		Timer 1 Channel 0 Compare Positive Output
62	79	40	PD13	PB	GPIO
			SPI1_MISO		SPI1 Master Input Slave Output
			SPI2_MISO		SPI2 Master Input Slave Output
			QSPI0_SI (IO1)		QSPI0 Serial Input (IO1)
			78161_IO		78161 Bidirectional Channel
			KEY0[12]		Key 0[12] (formerly KEY4)
NC	80	41	PD14	PB	GPIO
			TX0		UART0 Transmit
			TX1		UART1 Transmit
			USART1_CTS		USART1 Hardware Flow Control (Clear To Send)
			KEY1[13]		Key 1[13]
			TC0_N[0]		Timer 0 Channel 0 Compare Negative Output
63	81	42	PD15	PB_BI	GPIO
			SPI1_MOSI		SPI1 Master Output Slave Input
			SPI2_MOSI		SPI2 Master Output Slave Input
			QSPI0_SO (IO0)		QSPI0 Serial Output (IO0)
			78161_I		78161 Unidirectional Input Channel
			KEY0[13]		Key 0[13] (formerly INT5)
NC	82	NC	PE00	PB_BI	GPIO
			RX2		UART2 Receive
			RX3		UART3 Receive
			KEY1[12]		Key 1[12]
			TC1_P[0]		Timer 1 Channel 0 Compare Positive Output
			TCIN[9]		Timer Input Channel 9
65	83	44	PE01	PB	GPIO
			SPI2_SCSN		SPI2 Chip Select
			QSPI0_WPN (IO2)		QSPI0 Write Protect (Active Low) (IO2)
			RX0		UART0 Receive
			RX1		UART1 Receive
			USART1_RX		USART1 Receive
66	84	45	PE02	PB	GPIO

			SPI2_SCLK		SPI2 Clock
			QSPI0_HOLDN (IO3)		QSPI0 Hold (Active Low) (IO3)
			TX0		UART0 Transmit
			TX1		UART1 Transmit
			USART1_TX		USART1 Transmit
NC	85	NC	PE03	PB	GPIO
			RX0		UART0 Receive
			RX1		UART1 Receive
			KEY1[11]		Key 1[11]
			TC0_N[0]		Timer 0 Channel 0 Compare Negative Output
67	86	46	PE04	PB	GPIO
			SPI2_MISO		SPI2 Master Input Slave Output
			RX0		UART0 Receive
			RX1		UART1 Receive
			LPUART_RX		LPUART Receive
			USART1_CK		USART1 Clock
			78161_CLK		78161 Clock
68	87	47	PE05	PB	GPIO
			SPI2_MOSI		SPI2 Master Output Slave Input
			TX0		UART0 Transmit
			TX1		UART1 Transmit
			LPUART_TX		LPUART Transmit
			USART1_RTS		USART1 Hardware Flow Control (Request To Send)
			78160_I		78160 Unidirectional Input Channel
69	88	NC	PE06	PBL	GPIO
			SEG15		LCD Drive SEG Port
			KEY1[10]		Key 1[10]
			IOCNT3		IOCNT Pulse Output Channel 3
70	89	NC	PE07	PBL_BI	GPIO
			SEG14		LCD Drive SEG Port
			USART1_CTS		USART1 Hardware Flow Control (Clear To Send)
			KEY1[9]		Key 1[9]
			IOCNT4		IOCNT Pulse Output Channel 4
71	90	NC	PE08	PBL	GPIO
			SEG13		LCD Drive SEG Port
			I2C1_SDA		I2C1 Data
			KEY1[0]		Key 1[0]
72	91	NC	PE09	PBL	GPIO
			SEG12		LCD Drive SEG Port
			I2C1_SCL		I2C1 Clock
			KEY1[15]		Key 1[15]

73	92	NC	PE10	PBL	GPIO
			SEG11		LCD Drive SEG Port
			KEY1[6]		Key 1[6]
			IOCNT2		IOCNT Pulse Output Channel 2
74	93	NC	PE11	PBL	GPIO
			SEG10		LCD Drive SEG Port
			KEY1[5]		Key 1[5]
			IOCNT1		IOCNT Pulse Output Channel 1
NC	94	NC	PE12	PBL	GPIO
			SEG39		LCD Drive SEG Port
			KEY1[4]		Key 1[14]
NC	95	NC	PE13	PBL	GPIO
			SEG38		LCD Drive SEG Port
			KEY1[3]		Key 1[3]
75	96	48	PE14	PB37	GPIO
			HOSCI		High-Frequency Crystal Input (should be connected in series with a 10MΩ resistor and parallel 15pF capacitor to ground between HOSCI and HOSCO)
			TX0		UART0 Transmit
			TX1		UART1 Transmit
			TX2		UART2 Transmit
			TX3		UART3 Transmit
			KEY0[15]		Key 0[15] (formerly INT7)
76	97	49	PE15	PB36_BI	GPIO
			HOSCO		High-Frequency Crystal Output (should be connected in series with a 10MΩ resistor and parallel 15pF capacitor to ground between HOSCO and HOSCI)
			RX0		UART0 Receive
			RX1		UART1 Receive
			RX2		UART2 Receive
			RX3		UART3 Receive
			KEY0[14]		Key 0[14] (formerly INT6)
NC	98	NC	PF00	PBL	GPIO
			SEG37		LCD Drive SEG Port
			KEY1[2]		Key 1[2]
77	99	NC	PF01	PBL	GPIO
			SEG09		LCD Drive SEG Port
			KEY0[0]		Key 0[0] (formerly KEY0)
			IOCNT4		IOCNT Pulse Output Channel 4
78	100	NC	PF02	PBL	GPIO
			SEG8		LCD Drive SEG Port

			KEY0[1]		Key 0[1] (formerly KEY1)
			CF2		D2F2 Output
79	101	NC	PF03	PBL	GPIO
			SEG7		LCD Drive SEG Port
			KEY0[2]		Key 0[2] (formerly KEY2)
			CF1		D2F1 Output
80	102	NC	PF04	PBL	GPIO
			SEG6		LCD Drive SEG Port
			KEY0[3]		Key 0[3] (formerly KEY3)
			CF0		D2F0 Output
81	103	NC	PF05	PBL	GPIO
			SEG5		LCD Drive SEG Port
			KEY0[4]		Key 0[4] (formerly KEY4)
			RTCOUT		RTC Output
82	104	NC	PF06	PBL	GPIO
			SEG4		LCD Drive SEG Port
			KEY0[5]		Key 0[5] (formerly KEY5)
			IOCNT3		IOCNT Pulse Output Channel 3
83	105	NC	PF07	PB_BI	GPIO
			TX2		UART2 Transmit
			TX3		UART3 Transmit
			KEY1[1]		Key 1[1]
			TC0_P[0]		Timer 0 Channel 0 Compare Positive Output
			TCIN[9]		Timer Input Channel 9
84	106	NC	PF08	PBL	GPIO
			COM7		LCD Drive COM Port
			SEG3		LCD Drive SEG Port
			I2C0_SCL		I2C0 Clock
			KEY0[0]		Key 0[0] (formerly KEY0)
85	107	NC	PF09	PBL	GPIO
			COM6		LCD Drive COM Port
			SEG2		LCD Drive SEG Port
			I2C0_SDA		I2C0 Data
			KEY0[7]		Key 0[7] (formerly KEY7)
86	108	NC	PF10	PBL	GPIO
			COM5		LCD Drive COM Port
			SEG1		LCD Drive SEG Port
			KEY0[13]		Key 0[13] (formerly INT5)
			IOCNT2		IOCNT Pulse Output Channel 2
87	109	NC	PF11	PBL	GPIO
			COM4		LCD Drive COM Port
			SEG0		LCD Drive SEG Port

			KEY0[12]		Key 0[12] (formerly INT4)
			IOCNT1		IOCNT Pulse Output Channel 1
88	110	NC	PF12	PBL	GPIO
			COM3		LCD Drive COM Port
			KEY0[11]		Key 0[11] (formerly INT3)
			IOCNT0		IOCNT Pulse Output Channel 0
89	111	NC	PF13	PBL	GPIO
			COM2		LCD Drive COM Port
			KEY0[10]		Key 0[10] (formerly INT2)
			CF2		D2F2 Output
90	112	NC	PF14	PBL	GPIO
			COM1		LCD Drive COM Port
			KEY0[15]		Key 0[15] (formerly INT7)
			CF1		D2F1 Output
91	113	NC	PF15	PBL_BI	GPIO
			COM0		LCD Drive COM Port
			KEY0[14]		Key 0[14] (formerly INT6)
			CF0		D2F0 Output
NC	114	NC	PG00	PBL	GPIO
			SEG36		LCD Drive SEG Port
			KEY1[8]		Key 1[8]
92	115	50	PG01	PB	GPIO
			SPI3_SCSN		SPI3 Chip Select
			QSPI0_SCLK		QSPI0 Clock
			KEY0[0]		Key 0[0] (formerly KEY0)
			TC0_N[0]		Timer 0 Channel 0 Compare Negative Output
			TCIN[0]		Timer Input Channel 0
93	116	51	PG02	PB	GPIO
			SPI3_SCLK		SPI3 Clock
			QSPI0_CSN		QSPI0 Chip Select
			KEY0[1]		Key 0[1] (formerly KEY1)
			TC0_P[0]		Timer 0 Channel 0 Compare Positive Output
			TCIN[1]		Timer Input Channel 1
NC	117	52	PG03	PB	GPIO
			RX2		UART2 Receive
			RX3		UART3 Receive
			LPUART_RX		LPUART Receive
			KEY0[2]		Key 0[2] (formerly KEY2)
			TC0_N[1]		Timer 0 Channel 1 Compare Negative Output
			TCIN[2]		Timer Input Channel 2
94	118	53	PG04	PB_BI	GPIO
			ISPEN		After reset occurs, BOOTROM will detect the state

					of this port. If input is low level, the system will enter ISP. This issue should be noted in practical applications.
			SPI3_MOSI		SPI3 Master Output Slave Input
			QSPI0_SO (IO0)		QSPI0 Serial Output (IO0)
			USART0_CTS		USART0 Hardware Flow Control (Clear To Send)
			TX4		UART4 Transmit
			TX5		UART5 Transmit
			KEY0[8]		Key 0[8] (formerly INT0)
			TCIN[8]		Timer Input Channel 8
NC	119	54	PG05	PB	GPIO
			RX4		UART4 Receive
			RX5		UART5 Receive
			KEY0[9]		Key 0[9] (formerly INT1)
			TC0_P[1]		Timer 0 Channel 1 Compare Positive Output
			TCIN[9]		Timer Input Channel 9
95	120	55	PG06	PB	GPIO
			SPI3_MISO		SPI3 Master Input Slave Output
			QSPI0_SI (IO1)		QSPI0 Serial Input (IO1)
			KEY0[10]		Key 0[10] (formerly INT2)
			RTCOUT		RTC Output
96	121	56	PG07	PB	GPIO
			QSPI0_WPN (IO2)		QSPI0 Write Protect (Active Low) (IO2)
			RX2		UART2 Receive
			RX3		UART3 Receive
			RX4		UART4 Receive
			RX5		UART5 Receive
			TC0_N[0]		Timer 0 Channel 0 Compare Negative Output
			TCIN[10]		Timer Input Channel 10
97	122	57	PG08	PB_BI	GPIO
			QSPI0_HOLDN (IO3)		QSPI0 Hold (Active Low) (IO3)
			TX2		UART2 Transmit
			TX3		UART3 Transmit
			TX4		UART4 Transmit
			TX5		UART5 Transmit
			TC0_P[0]		Timer 0 Channel 0 Compare Positive Output
			TCIN[11]		Timer Input Channel 11
NC	123	58	PG09	PB_BI	GPIO
			TX2		UART2 Transmit
			TX3		UART3 Transmit
			LPUART_TX		LPUART Transmit

			KEY1[15]		Key 1[15]
			TCIN[7]		Timer Input Channel 7
			TC1_P[1]		Timer 1 Channel 1 Compare Positive Output
98	124	59	PG10	PB	GPIO
			JTDO		JTAG Data Output
			SPI_SCLK		SPIx Clock (x=0-3)
			RX6		UART6 Receive
			RX7		UART7 Receive
			IOCNT0		IOCNT Pulse Output Channel 0
			RTCOUT		RTC Output
99	125	60	PG11	PB	GPIO
			JTDI		JTAG Data Input
			SPI_SCSN		SPIx Chip Select (x=0-3)
			TX6		UART6 Transmit
			TX7		UART7 Transmit
			IOCNT1		IOCNT Pulse Output Channel 1
			RTCOUT		RTC Output
NC	126	61	PG12	PB	GPIO
			RX0		UART0 Receive
			RX1		UART1 Receive
			LPUART_RX		LPUART Receive
			KEY0[12]		Key 0[12] (formerly INT4)
			CF1		D2F1 Output
			IOCNT2		IOCNT Pulse Output Channel 2
NC	127	NC	PG13	PB_BI	GPIO
			TX0		UART0 Transmit
			TX1		UART1 Transmit
			LPUART_TX		LPUART Transmit
			KEY0[13]		Key 0[13] (formerly INT5)
			TCIN[11]		Timer Input Channel 11
			CF2		D2F2 Output
100	128	62	DGND	GND	Ground

2 Electrical Characteristics

Reference Voltage (VCC=3V~5.5V, Temperature Range: -40℃ to +85℃)						
		min.	typ.	max.	unit	Notes
Output Voltage	Vref		1.25		V	Dispersion less than ±1%
Temperature Coefficient	Tc		TBD		ppm/℃	
Analog Peripherals (Temperature Range: -40℃ to +85℃)						
Low-power Comparator CMP1/CMP2/LVDIN Threshold Vil1	CMP	1.23	1.28	1.33	V	Under default configuration, this threshold is the comparator output low-level comparison result threshold; the output high-level comparison result threshold is 220mV higher than this threshold.
Low-power Comparator CMP1/CMP2 Threshold Vil2	CMP	0.8	0.84	0.88	V	When the 0.9V range is selected and hysteresis is enabled, this threshold is the comparator output low-level comparison result threshold; the output high-level comparison result threshold is 140mV higher than this threshold.
SAR ADC Input Range	SAR-IN	0		VREF	V	VREF is the internal reference voltage, typical value is 1.26V
LCD Output Voltage (Resistor Divider)	LCDVD	2.7	3.0	3.6	V	Relative to the power supply, the drop voltage is 0.2V, output has 16 adjustable levels, step size 60mV
VBAT Measurement	VBATD	0	3.6	3.8	V	SAR ADC measurement range for VBAT
VREF Startup Stabilization Time 1	Tvref1		5		ms	3V; 25°C; External 1uF capacitor; Using original VREF circuit
VREF Startup Stabilization Time 2	Tvref2		0.2		ms	3V; 25°C; External 1uF capacitor; Using newly designed VREF circuit
SAR ADC Conversion Time	Tsar		1.5		ms	SAR ADC sampling clock is 32KHz; 3V; Using

						optimized algorithm
Temperature Compensation Time per Cycle 1	Ttps1		7		ms	3V; 25°C; Using original VREF circuit, external 1uF capacitor
Temperature Compensation Time per Cycle 2	Ttps2		1.7		ms	3V; 25°C; Using newly designed VREF circuit, external 1uF capacitor
Clock Parameters (Temperature Range: -40℃ to +85℃)						
Input Low Frequency Clock Frequency Range	XI		32.768		KHz	
Input High Frequency Clock Frequency Range	HOSI	3.6864	7.3728	29.4912	MHz	
Internal PLL Clock Frequency Range	PLL		7.3728*8		MHz	
Internal High Frequency RCH58M	RCH58M		58		MHz	Used as the default clock after chip reset
Internal Low Frequency RCL	RCL	20	32.768	40	KHz	Used for WDT clock
Power Supply (Temperature Range: -40℃ to +85℃)						
Main Supply Voltage	VCC	2.3	5/3.3	5.5	V	CPU normal operation, i.e., BOR reset point voltage
CPU Minimum Operating Voltage	Vil	2.1	2.2	2.35	V	Room temperature
		2	2.15	2.3		Low temperature -40°C
		2.2	2.3	2.45		High temperature 85°C
VBAT Supply	VBAT	2.2	3.6	5.5	V	RTC temperature compensation circuit normal operation, RTC accuracy guaranteed voltage operating range
		1.8	3.6	5.5		Calendar maintained voltage operating range
Digital Current	DIdd		1.5		mA	CPU running at 3.6864MHz
Sleep Power Consumption 1	SIdd1		6	10	μA	RTC auto temperature compensation; RAM retention; CPU and digital peripherals not powered down; WDT enabled; Power monitoring enabled; Interrupt wake-up. Typical condition: 3.6V; 25°C
Sleep Power	SIdd2		4	6	μA	Partial digital modules,

Consumption 2						FLASH powered down. Typical condition: 3.6V; 25°C
VBAT Supply Power 1	SIdd3		2	3	μA	Typical condition: 25°C; 3.6V; TT process
VBAT Supply Power 2	SIdd4		3	4.5	μA	Typical condition: 25°C; 5V; TT process
VREF Power 1	REFIdd1		590	800	μA	Typical condition: 25°C; 5V; TT process. Using original 9103C circuit
VREF Power 2	REFIdd2		100	150	μA	Typical condition: 25°C; 5V; TT process. Using newly designed circuit
SAR Power 1	SARIdd1		230	300	μA	Typical condition: 25°C; 5V; TT process
SAR Power 2	SARIdd2		150	200	μA	Typical condition: 25°C; 5V; TT process
RCH Power	RCHIdd		50	70	μA	Typical condition: 25°C; 5V; TT process
Instantaneous Temperature Compensation Power 1	TPSIdd1		870	1170	μA	Typical condition: 25°C; 5V; TT process. Using original 9103C circuit
Instantaneous Temperature Compensation Power 2	TPSIdd2		300	420	μA	Typical condition: 25°C; 5V; TT process. Using newly designed circuit
Instantaneous Temperature Compensation Power 3	TPSIdd3		250	350	μA	Typical condition: 25°C; 3.6V; TT process. Using newly designed circuit
LDO15	V1P5	1.4	1.5	1.6	V	Requirement 1: CPU core voltage, full temperature range ±7%. Before trim 1.4~1.6V
LDO15	V1P5	1.46	1.5	1.545		Requirement 2: LCD voltage reference full temperature range ±3%. Tightened range, mainly considering FLASH IP needs to provide BGR. After trim 1.46~1.54V
Absolute Maximum Ratings (Temperature Range: -40℃ to +85℃)						
Main Supply Voltage	Vvcc	-0.3	--	+7	V	

Battery Input Voltage	Vvbat	-0.3	--	+7	V	
DVDD to DGND		-0.3	--	+7	V	
DVDD to AVDD		-0.3		+0.3	V	
V1P,V1N,V2P,V2N, V3P,V3N		-6		+6	V	
Digital IO Output High Level	VOH		--	VCC+0.3	V	
Digital IO Output Low Level	VOL	-0.3	--		V	
Digital IO Input High Level	VIH		0.7VCC			CMOS
Digital IO Input Low Level	VIL		0.3VCC			CMOS
Digital IO Input High Level	VIH		0.4VCC			TTL
Digital IO Input Low Level	VIL		0.2VCC			TTL
Digital IO Isource	Isource	5		10	mA	6mA type
Digital IO Isink	Isink	7		15	mA	6mA type
Digital IO Isource	Isource	3		5	mA	3mA type
Digital IO Isink	Isink	5		10	mA	3mA type
Analog Input Voltage Relative to AGND	VINA	-0.3	--	AVDD +0.3	V	
Operating Temperature Range	TA	-40	--	85	℃	
Storage Temperature Range	Tstg	-65	--	150	℃	
Lead-free Soldering Temperature	Tsdr	--	260		℃	

Measurement Item	**Symbol**	**Test Conditions**	**Value**	**Unit**
Electrostatic Discharge	ESD	Human Body Model (HBM), according to JEDEC EIA/JESD22-A114, on all pins	4000	V
		Machine Model (MM), according to JEDEC EIA/JESD22-A115C, on all pins	200	V
		Charged Device Model (CDM), according to JEDEC EIA/JESD22-C101F, on all pins	1000	V
Latch-up Test	LatchUP	According to JEDEC STANDARD NO.78D NOVEMBER 2011, on all pins	200	mA
Moisture Sensitivity	MSD	Rated according to IPC/JEDEC J-STD-020D.1	Level 3	/

3 System Architecture

3.1 Introduction

- Dual-Core System: Adopts a RISC-V heterogeneous dual-core architecture.
  - Core0 uses the N200 core.
  - Core1 uses the N300 core.
  - Supports different application scenarios such as Core0 single-core applications, Core1 single-core applications, and dual-core applications with Core0 as the security core and Core1 as the application core.
- Security Architecture Design: Adopts TEE security architecture, supporting CMVP/PSA and other security certifications.
  - Core0 application and Core0 SRAM are located in the secure storage area, which Core1 cannot access.
  - The system composed of Core0 and its private components, including Core0 core, SEA encryption/decryption security engine, Core0 private peripherals, Core0 application software, and other hardware/software resources, can serve as an independent security certification subject.
  - The SEA encryption/decryption security engine supports an independent key storage module, KeyStore.
  - Core0 startup supports secure signature verification, meeting the secure boot process requirements.
  - Core0 provides software service interfaces for security modules to Core1; Core1 can call encryption/decryption services inside Core0.
- System Resource Allocation: Resources allocated to Core0 cannot be accessed by Core1; Core0 can access resources allocated to Core1.
  - FLASH supports Core0/Core1 allocation with a minimum granularity of 8KB. The entire 1MB FLASH can be allocated to Core0 or Core1.
  - SRAM supports Core0/Core1 allocation with a minimum granularity of 16KB. The entire 160KB SRAM can be allocated to Core0; Core1 can be allocated up to 144KB.
  - Most peripherals support Core0/Core1 allocation.
  - GPIO supports Core0/Core1 allocation with single-pin granularity.
  - System resource allocation can be locked; after locking, software cannot modify the allocation, meeting security application requirements.
- Operating Clock: Core0 and Core1 system clocks use the same clock source. Core0/Core1 can be configured with different division coefficients to support different operating frequencies. Core0 maximum system operating frequency is 29.4912MHz; Core1 maximum system operating frequency is 58.9824MHz.
- Reset: Supports different reset sources.
  - Power reset resets both Core0 and Core1.
  - Pin reset resets both Core0 and Core1.
  - Core0 WDT resets both Core0 and Core1; Core1 WDT resets Core1, software configurable whether to reset Core0.
  - Core0 software reset resets both Core0 and Core1; Core1 software reset resets Core1, software configurable whether to reset Core0.
- Interrupts: Global interrupt management. Each interrupt source supports distribution to specified

Core0/Core1 core or simultaneous distribution to both cores. Global interrupt management supports locking; after locking, software cannot modify it.

- Multiple Inter-Core Communication Methods:
  - Supports inter-core interrupts with attached messages.
  - Supports 32 hardware semaphores.
  - Supports a 256-byte depth mailbox.
  - Supports shared memory.

3.2 RISC-V Cores

The heterogeneous dual-core system consists of Core0 and Core1, where Core0 is the N200 core and Core1 is the N300 core.

- Core0 Core Features:
  - N200 core;
  - 2-stage pipeline;
  - RV32I standard instruction set;
  - Extended instruction set support: M/A/C/B/Zc standard extensions, supports unaligned access;
  - Machine Mode, User Mode;
  - Single-cycle multiplier;
  - Fast divider (17 cycles);
  - Branch prediction;
  - 8KB I-Cache;
  - 8 PMP Entries, 4KB granularity;
  - 64-bit timer (mtime);
  - 64-bit cycle counter (mcycle);
  - 64-bit instruction retirement counter (minstret);
  - ECLIC enhanced interrupt controller;
  - JTAG or cJTAG debug interface;
  - 8 hardware breakpoints;
  - Debug standard compliant with RISC-V debug specification v0.13.
- Core1 Core Features:
  - N300 core;
  - 3-stage pipeline;
  - RV32I standard instruction set;
  - Extended instruction set support: M/A/F/C/B/Zc standard extensions, supports unaligned access;
  - Single-precision floating-point unit;
  - Machine Mode, User Mode, Supervisor Mode;
  - Single-cycle multiplier;
  - Fast divider (17 cycles);
  - Branch prediction;
  - 8KB I-Cache, 4KB D-Cache;
  - Supports Stack check;
  - 8 PMP (Physical Memory Protection) Entries, 4KB granularity;
  - 64-bit timer (mtime);

- 64-bit cycle counter (mcycle);
- 64-bit instruction retirement counter (minstret);
- ECLIC enhanced interrupt controller;
- JTAG or cJTAG debug interface;
- 8 hardware breakpoints;
- Debug standard compliant with RISC-V debug specification v0.13.

3.3 Bus Architecture

3.3.1 Overall Block Diagram

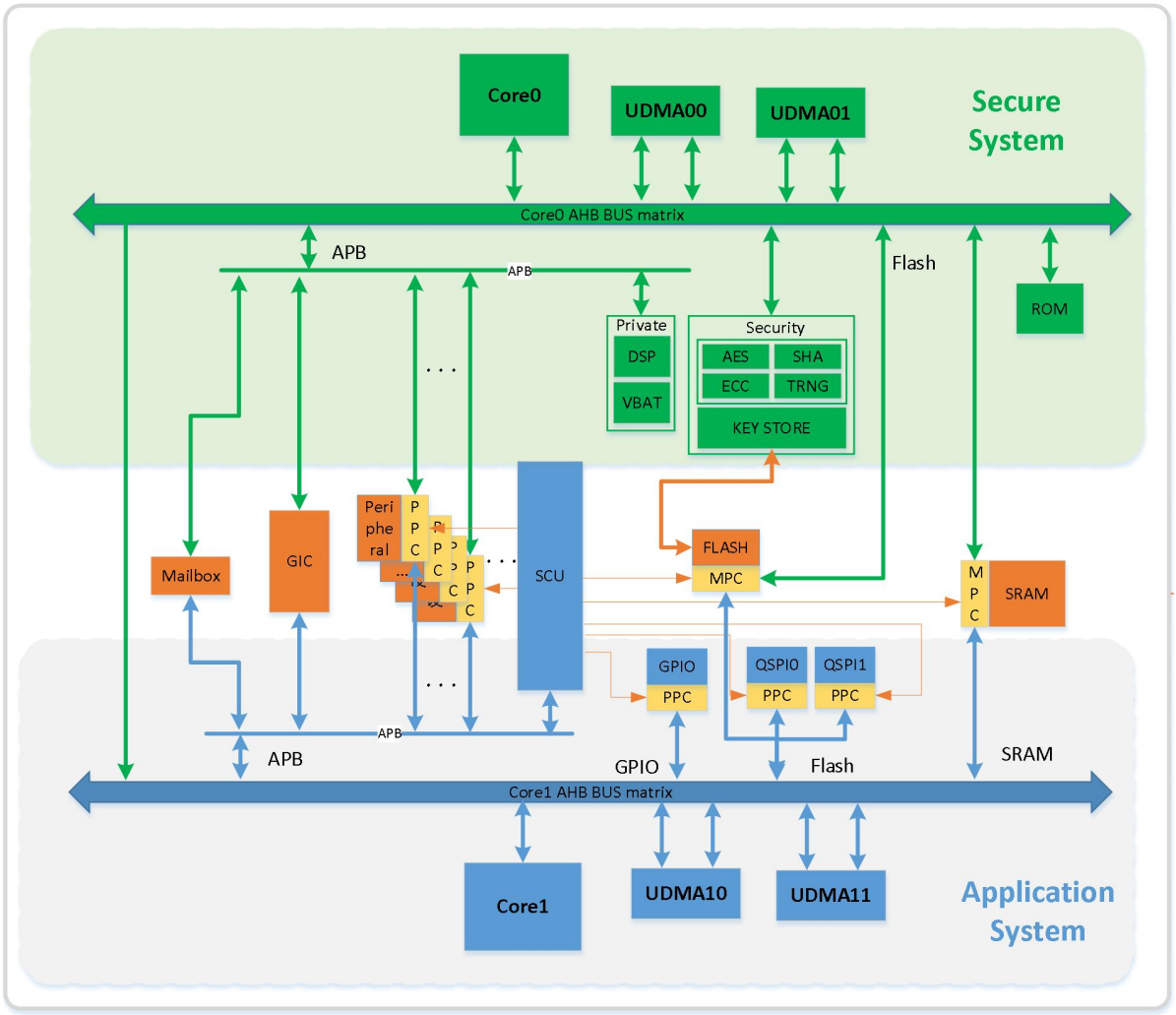


Figure 3-1 System Block Diagram

- Dual-bus system architecture; the chip contains 2 system bus groups: Core0 Bus and Core1 Bus.
- Core0 Bus:
  - 5 Masters: Core0, DMA00 Read, DMA00 Write, DMA01 Read, DMA01 Write.
  - Several Slaves.
- Core1 Bus:

- 6 Masters: Core1, DMA10 Read, DMA10 Write, DMA11 Read, DMA11 Write, HSM MASTER.
- Several Slaves.

- Resources such as peripherals, FLASH, and SRAM are uniformly managed by the SCU for security configuration management; peripheral PPC and FLASH/SRAM MPC implement security access management for peripherals and memory.
- Secure masters have channels to access the non-secure bus.
- Dual-core system design: Core0 and Core1 interact data through Mailbox, inter-core interrupts, and shared memory; interrupts are uniformly configured and managed by the GIC module.
- SRAM total capacity 160KB, can be allocated into secure and non-secure attribute contiguous memory areas with 16KB granularity.
- FLASH total capacity 1MB, can be allocated into secure and non-secure attribute contiguous memory areas with 8KB granularity.
- There are two QSPI interface modules on the Core1 bus, supporting XIP function, can be connected to external FLASH or SRAM memory chips, and can be used to store Core1 programs or data.
- Core0 does not support XIP function, but the QSPI module can be allocated to the Core0 system (using non-XIP mode).
- GPIO is connected to the Core1 bus; Core0 and Core1 can only access the GPIO module through the Core1 bus channel.

3.3.2 Bus Matrix

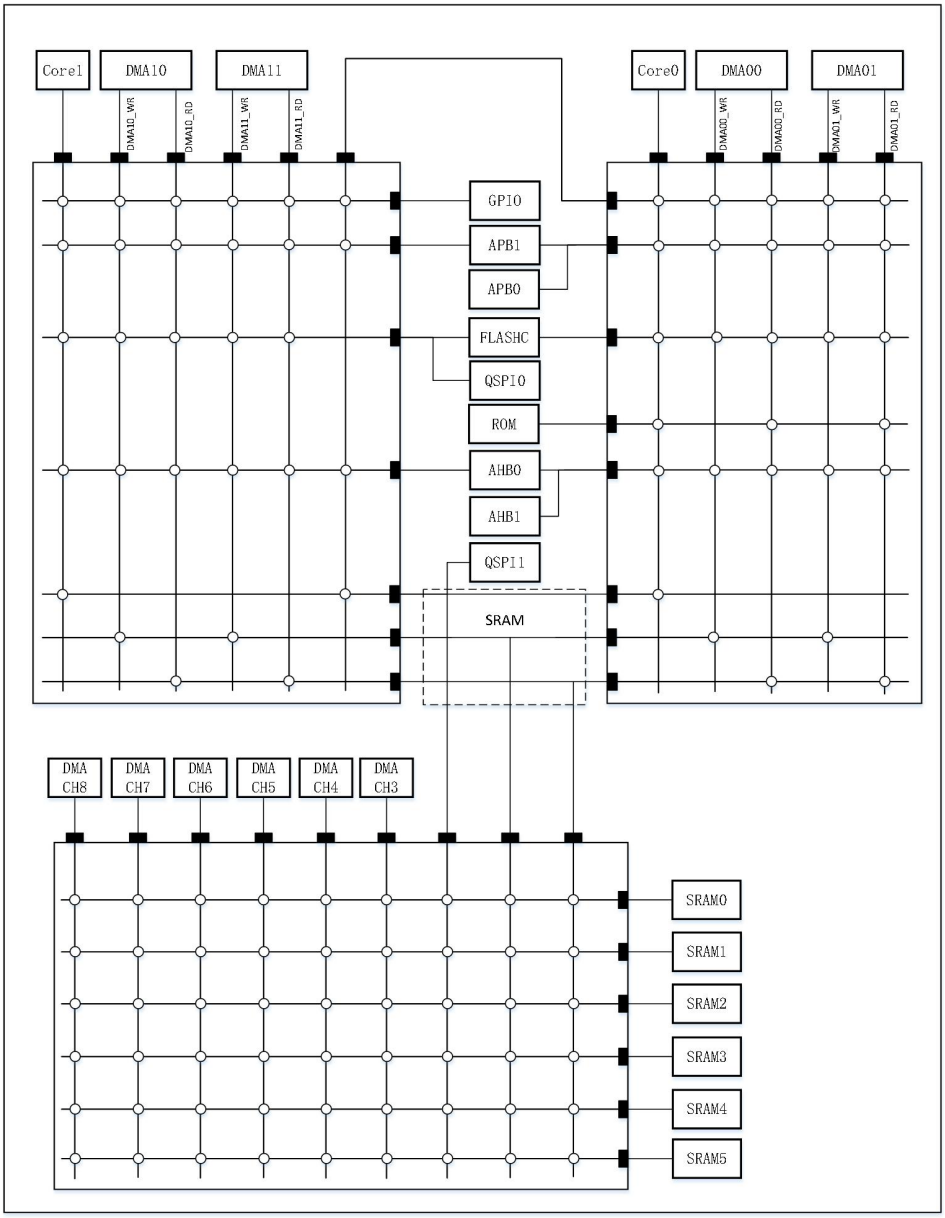


Figure 3-2 System Bus Matrix Diagram

- The bus matrix consists of the Core0 bus matrix and the Core1 bus matrix.
  - Core1 Bus Matrix, contains 6 masters: Core1, DMA10 (read/write separate), DMA11 (read/write separate), Core0 MASTER.
  - Core0 Bus Matrix, contains 5 masters: Core0, DMA00 (read/write separate), DMA01 (read/write separate).
- The slave end is located between the Core0 bus matrix and the Core1 bus matrix.
  - SRAM consists of 5 independent SRAM memory blocks to enable multi-channel simultaneous read/write, improving system parallel processing capability.

3.4 Memory Map

3.4.1 Memory Map Diagram

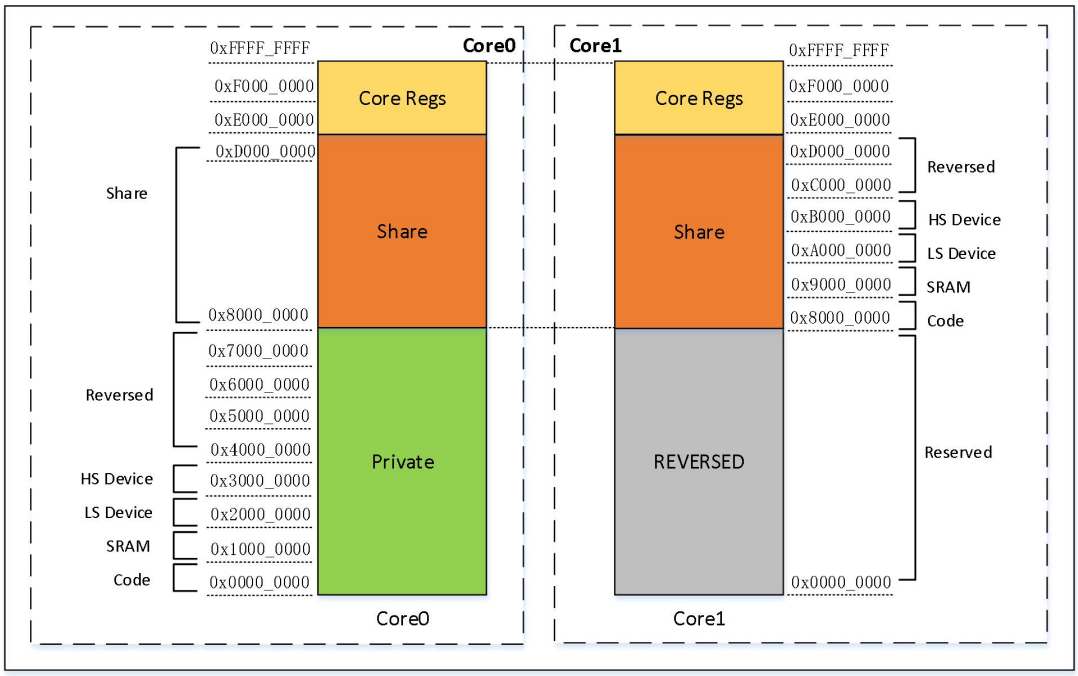


Figure 3-3 Bus Address Space Mapping

Core0 System Bus Space includes:

- Private Space: 0x0000_0000~0x7FFF_FFFF
- Shared Space: 0x8000_0000~0xDFFF_FFFF
- Kernel Space: 0xE000_0000~0xFFFF_FFFF

Core0 can access private peripherals on the Core0 bus through the private space address; simultaneously, it can access all memory and peripherals on the Core1 bus through the shared space.

Core1 System Bus Space includes:

- Reserved Undefined Space: 0x0000_0000~0x7FFF_FFFF
- Shared Space: 0x8000_0000~0xDFFF_FFFF
- Kernel Space: 0xE000_0000~0xFFFF_FFFF

Core1 has no private space. Core1 code and peripherals are located in the shared space area, supporting access by Core0.

3.4.2 Detailed Address Mapping Tables

3.4.2.1 Core0 Address Mapping Table

Core0 can access the secure system (Core0 Bus) and the non-secure system (Core1 Bus). When a memory area or peripheral is allocated to Core0, Core0 should access it through the Core0 secure system bus address (Core0 private address space) and cannot access it through the Core1 non-secure system bus address (Core0 shared address space). When a memory area or peripheral is allocated to Core1, Core0 should access it through the Core1 non-secure system bus address and cannot access it through the Core0 secure system bus address.

GPIO/HSEM/QSPI modules are different from other allocatable peripheral resources. These modules are located on the Core1 bus, and their access addresses are fixedly mapped to the Core1 non-secure system bus. The

access address is independent of the peripheral security attribute settings.

BUS	Secure Boundary Address	Non-Secure Boundary Address	Size (KB)	Slave
ROM	0x0000_0000~0x0000_BFFF	-	48	Rom
	0x0000_C000~0x000F_FFFF	-	976	Reserved
	0x0010_0000~0x001F_FFFF	-	1024	Reserved
FLASH（remap=0）	0x0020_0000~0x002F_FFFF	-	0~1024	Core0 FLASH
	0x0030_0000~0x0030_5FFF	-	24	FLASH NVR
	0x0030_6000~0x0FFF_FFFF	-	259048	Reserved
	-	0x8000_0000~0x800F_FFFF	0~1024	Core1 FLASH
	-	0x8010_0000~0x80FF_FFFF	15360	Reserved
	-	0x8100_0000~0x81FF_FFFF	16K	External FLASH
		0x8200_0000~0x8FFF_FFFF	-	Reserved
Sram Ch0 Sram Ch1 Sram Ch2 （remap=0)	0x1000_0000~0x1002_7FFF	-	0~160	Core0 Sram
	0x1002_8000~0x11FF_FFFF	-	32608	Reserved
	0x1200_0000~0x124F_FFFF	-	(0~160)*32	Core0 Sram bitband
	0x1250_0000~0x1FFF_FFFF	-	224256	Reserved
	-	0x9000_0000~0x9002_7FFF	0~160	Core1 Sram (noncacheable）
	-	0x9002_8000~0x903F_FFFF	3936	Reserved
	-	0x9040_0000~0x907F_FFFF	4096	External SRAM （noncacheable）
		0x9080_0000~0x91FF_FFFF	24576	Reserved
		0x9200_0000~0x924F_FFFF	(0~160)*32	Core1 Sram bitband （noncacheable）
		0x9250_0000~0x97FF_FFFF	93184	Reserved
		0x9800_0000~0x9802_7FFF	0~160	Core1 Sram（cacheable）
		0x9802_8000~0x983f_FFFF	3936	Reserved
		0x9840_0000~0x987F_FFFF	4096	External SRAM （cacheable）
		0x9880_0000~0x99FF_FFFF	24576	Reserved
		0x9A00_0000~0x9A4F_FFFF	(0~160)*32	Core1 Sram bitband （cacheable）
		0x9A50_0000~0x9FFF_FFFF	93184	Reserved
APB Device	0x2000_0000	0xA000_0000	16	UART0
	0x2000_4000	0xA000_4000	16	UART1
	0x2000_8000	0xA000_8000	16	UART2
	0x2000_C000	0xA000_C000	16	UART3

	0x2001_0000	0xA001_0000	16	UART4
	0x2001_4000	0xA001_4000	16	UART5
	0x2001_8000	0xA001_8000	16	UART6
	0x2001_C000	0xA001_C000	16	UART7
	0x2002_0000	0xA002_0000	16	LPUART
	0x2002_4000	0xA002_4000	16	USART0
	0x2002_8000	0xA002_8000	16	USART1
	0x2002_C000	0xA002_C000	16	LSRX
	0x2003_0000	0xA003_0000	16	78160
	0x2003_4000	0xA003_4000	16	78161
	0x2003_8000	0xA003_8000	16	SPI0
	0x2003_C000	0xA003_C000	16	SPI1
	0x2004_0000	0xA004_0000	16	SPI2
	0x2004_4000	0xA004_4000	16	SPI3
	0x2005_0000	0xA005_0000	16	I2C0
	0x2005_4000	0xA005_4000	16	I2C1
	0x2005_8000	0xA005_8000	16	Reserved
	0x2005_C000	0xA005_C000	16	IOCNT
	0x2006_0000	0xA006_0000	16	LCD
	0x2006_4000	0xA006_4000	16	KBI0
	0x2006_8000	0xA006_8000	16	KBI1
	0x2006_C000	0xA006_C000	16	DSP
	0x2007_0000	0xA007_0000	16	CRC0
	0x2007_8000	0xA007_8000	16	M2M
	0x2007_C000	0xA007_C000	16	D2F
	0x2008_0000	0xA008_0000	16	SYS_REG
	0x2008_8000	0xA008_8000	16	AOS
	0x2009_4000	0xA009_4000	16	ADC0
	0x2009_8000	0xA009_8000	16	DAC
	0x2009_C000	0xA009_C000	16	TC0
	0x200A_0000	0xA00A_0000	16	TC1
	0x200A_4000	0xA00A_4000	16	SIMPTC0
	0x200A_8000	0xA00A_8000	16	SIMPTC1
	0x200A_C000	0xA00A_C000	16	SIMPTC2
	0x200B_0000	0xA00B_0000	16	SIMPTC3
	0x200B_4000	0xA00B_4000	16	RTC
	0x200B_8000	0xA00B_8000	16	MADC
	0x200B_C000	0xA00B_C000	16	LVD/CMP0_1
	-	-		Reserved
	0x2010_0000	-	16	WDT0
	0x2010_4000	-	16	UDMA00

	0x2010_8000	-	16	UDMA01
	0x2010_C000	-	16	LCSS
	0x2011_0000	-	16	SYS_TC
	0x2011_4000	-	16	CRC1
	-	0xA010_0000	16	WDT1
	-	0xA010_4000	16	UDMA10
	-	0xA010_8000	16	UDMA11
	-	-		Reserved
	0x2200_0000~0x2FFF_FFFF	0xA200_0000~0xAFFF_FFFF		APB bitband
AHB Device	0x3000_0000	0xB000_0000	16	FLASH_REG
	0x3000_8000	0xB000_8000	16	SEA/KS
	0x3000_C000	0xB000_C000	16	MAILBOX
	0x3001_4000	0xB001_4000	16	AES
	0x3001_8000	0xB001_8000	16	SCU
	0x3001_C000	0xB001_C000	16	GIC
	-	-		Reserved
	-	0xB008_0000	16	GPIO
	-	0xB008_4000	16	HSEM
	-	0xB008_8000	16	QSPI0
	-	0xB008_C000	16	QSPI1
	-	-		Reserved
	0x3200_0000~0x3FFF_FFFF	0xB200_0000~0xBFFF_FFFF		AHB bitband
Core	0xE000_0000~0xE000_FFFF	-	64	CORE0 IINFO
	0xE001_0000~0xE001_FFFF	-	64	CORE0 DEBUG
	0xE002_0000~0xE002_FFFF	-	64	CORE0 ECLIC
	0xE003_0000~0xE003_FFFF	-	64	CORE0 TIMER

3.4.2.2 Core1 Address Mapping Table

Core1 can access the non-secure system (Core1 Bus) and cannot access the secure system (Core0 Bus). When a memory area or peripheral is allocated to Core1, Core1 can access it through the non-secure system bus address. When a memory area or peripheral is allocated to Core0, it cannot be accessed via the bus.

BUS	Non-Secure Boundary Address	Size (KB)	Slave
FLASH (remap=0)	0x8000_0000~0x800F_FFFF	0~1K	Core1 FLASH
	0x8010_0000~0x80FF_FFFF	15K	Reserved
	0x8100_0000~0x81FF_FFFF	16K	External FLASH
	0x8200_0000~0x8FFF_FFFF	4064K	Reserved
Sram Ch0 Sram Ch1 Sram Ch2 (remap=0)	0x9000_0000~0x9002_7FFF	0~160	Core1 Sram（noncacheable）
	0x9002_8000~0x903F_FFFF	3936	Reserved
	0x9040_0000~0x907F_FFFF	4096	External SRAM（noncacheable）
	0x9080_0000~0x91FF_FFFF	24576	Reserved

	0x9200_0000~0x924F_FFFF	(0~160)*32	Core1 Sram bitband（noncacheable）
	0x9250_0000~0x97FF_FFFF	93184	Reserved
	0x9800_0000~0x9802_7FFF	0~160	Core1 Sram（cacheable）
	0x9802_8000~0x983F_FFFF	3936	Reserved
	0x9840_0000~0x987F_FFFF	4096	External SRAM（cacheable）
	0x9880_0000~0x99FF_FFFF	24576	Reserved
	0x9A00_0000~0x9A4F_FFFF	(0~160)*32	NS Sram bitband（cacheable）
	0x9A50_0000~0x9FFF_FFFF	93184	Reserved
APB Device	0xA000_0000	16	UART0
	0xA000_4000	16	UART1
	0xA000_8000	16	UART2
	0xA000_C000	16	UART3
	0xA001_0000	16	UART4
	0xA001_4000	16	UART5
	0xA001_8000	16	UART6
	0xA001_C000	16	UART7
	0xA002_0000	16	LPUART
	0xA002_4000	16	USART0
	0xA002_8000	16	USART1
	0xA002_C000	16	LSRX
	0xA003_0000	16	78160
	0xA003_4000	16	78161
	0xA003_8000	16	SPI0
	0xA003_C000	16	SPI1
	0xA004_0000	16	SPI2
	0xA004_4000	16	SPI3
	0xA005_0000	16	I2C0
	0xA005_4000	16	I2C1
	0xA005_8000	16	Reserved
	0xA005_C000	16	IOCNT
	0xA006_0000	16	LCD
	0xA006_4000	16	KBI0
	0xA006_8000	16	KBI1
	0xA006_C000	16	DSP
	0xA007_0000	16	CRC0
	0xA007_8000	16	M2M
	0xA007_C000	16	D2F
	0xA008_0000	16	SYS_REG
	0xA008_8000	16	AOS
	0xA009_4000	16	ADC0
	0xA009_8000	16	DAC

	0xA009_C000	16	TC0
	0xA00A_0000	16	TC1
	0xA00A_4000	16	SIMPTC0
	0xA00A_8000	16	SIMPTC1
	0xA00A_C000	16	SIMPTC2
	0xA00B_0000	16	SIMPTC3
	0xA00B_4000	16	RTC
	0xA00B_8000	16	MADC
	0xA00B_C000	16	LVDCMP
	-		Reserved
	0xA010_0000	16	WDT1
	0xA010_4000	16	UDMA10
	0xA010_8000	16	UDMA11
	-		Reserved
	0xA200_0000~0xAFFF~FFFF		APB bitband
AHB Device	0xB000_0000	16	FLASH_REG
	0xB000_8000	16	SEA/KS
	0xB000_C000	16	MAILBOX
	0xB001_4000	16	AES
	0xB001_8000	16	SCU
	0xB001_C000	16	GIC
	-		Reserved
	0xB008_0000	16	GPIO
	0xB008_4000	16	HSEM
	0xB008_8000	16	QSPI0
	0xB008_C000	16	QSPI1
	-		Reserved
	0xB200_0000~0xBFFF_FFFF		AHB bitband
cpu device	0xE000_0000~0xE000_FFFF	64	CORE1 IINFO
	0xE001_0000~0xE001_FFFF	64	CORE1 DEBUG
	0xE002_0000~0xE002_FFFF	64	CORE1 ECLIC
	0xE003_0000~0xE003_FFFF	64	CORE1 TIMER

3.4.2.3 Memory Address Mapping under Various Remap Settings

The access address of memory is related to the remap setting, see the table below.

Memory Type		**core0_remap**		
		000/011/1xx	**001**	**010**
ROM	ROM	0x0000_0000 ~0x0000_7FFF	0x0020_0000 ~0x0020_7FFF	0x1000_0000 ~0x1000_7FFF
	Reserved	0x0000_8000 ~0x001F_FFFF	0x0020_8000 ~0x0FFF_FFFF	0x1000_8000 ~0x1FFF_FFFF

FLASH	Core0 FLASH	0x0020_0000 ~0x002F_FFFF	0x0000_0000 ~0x000F_FFFF	0x0020_0000 ~0x002F_FFFF
	FLASH NVR	0x0030_0000 ~0x0030_5FFF	0x0010_0000 ~0x0010_5FFF	0x0030_0000 ~0x0030_5FFF
	Reserved	0x0030_6000 ~0x0FFF_FFFF	0x0010_6000 ~0x001F_FFFF	0x0030_6000 ~0x0FFF_FFFF
SRAM	Core0 Sram	0x1000_0000 ~0x1002_7FFF	0x1000_0000 ~0x1002_7FFF	0x0000_0000 ~0x0002_7FFF
	Reserved	0x1002_8000 ~0x11FF_FFFF	0x1002_8000 ~0x11FF_FFFF	0x0002_8000 ~0x01FF_FFFF
	Core0 Sram bitband	0x1200_0000 ~0x124F_FFFF	0x1200_0000 ~0x124F_FFFF	0x0200_0000 ~0x024F_FFFF
	Reserved	0x1250_0000 ~0x1FFF_FFFF	0x1250_0000 ~0x1FFF_FFFF	0x0250_0000 ~0x0FFF_FFFF

Memory Type		**core1_remap**		
		000/011/1xx	**001**	**010**
FLASH	Core1 FLASH	0x8000_0000 ~0x800F_FFFF	0x9000_0000 ~0x900F_FFFF	0x8100_0000 ~0x810F_FFFF
	Reserved	0x8010_0000 ~0x80FF_FFFF	0x9010_0000 ~0x90FF_FFFF	0x8110_0000 ~0x81FF_FFFF
	External FLASH	0x8100_0000 ~0x81FF_FFFF	0x9100_0000 ~0x91FF_FFFF	0x8000_0000 ~0x80FF_FFFF
	Reserved	0x8200_0000 ~0x8FFF_FFFF	0x9200_0000 ~0x9FFF_FFFF	0x8200_0000 ~0x8FFF_FFFF
SRAM	Core1 Sram （noncacheable）	0x9000_0000 ~0x9002_7FFF	0x8000_0000 ~0x8002_7FFF	0x9000_0000 ~0x9002_7FFF
	Reserved	0x9002_8000 ~0x903F_FFFF	0x8002_8000 ~0x803F_FFFF	0x9002_8000 ~0x903F_FFFF
	External SRAM （noncacheable）	0x9040_0000 ~0x907F_FFFF	0x8040_0000 ~0x807F_FFFF	0x9040_0000 ~0x907F_FFFF
	Reserved	0x9080_0000 ~0x91FF_FFFF	0x8080_0000 ~0x81FF_FFFF	0x9080_0000 ~0x91FF_FFFF
	Core1 Sram bitband （noncacheable）	0x9200_0000 ~0x924F_FFFF	0x8200_0000 ~0x824F_FFFF	0x9200_0000 ~0x924F_FFFF
	Reserved	0x9250_0000 ~0x97FF_FFFF	0x8250_0000 ~0x87FF_FFFF	0x9250_0000 ~0x97FF_FFFF
	Core1 Sram（cacheable）	0x9800_0000 ~0x9802_7FFF	0x8800_0000 ~0x8802_7FFF	0x9800_0000 ~0x9802_7FFF

	Reserved	0x9802_8000 ~0x983f_FFFF	0x8802_8000 ~0x883f_FFFF	0x9802_8000 ~0x983f_FFFF
	External SRAM （cacheable）	0x9840_0000 ~0x987F_FFFF	0x8840_0000 ~0x887F_FFFF	0x9840_0000 ~0x987F_FFFF
	Reserved	0x9880_0000 ~0x99FF_FFFF	0x8880_0000 ~0x89FF_FFFF	0x9880_0000 ~0x99FF_FFFF
	Core1 Sram bitband （cacheable）	0x9A00_0000 ~0x9A4F_FFFF	0x8A00_0000 ~0x8A4F_FFFF	0x9A00_0000 ~0x9A4F_FFFF
	Reserved	0x9A50_0000 ~0x9FFF_FFFF	0x8A50_0000 ~0x8FFF_FFFF	0x9A50_0000 ~0x9FFF_FFFF

3.4.3 Bitband Region

There are three bitband address mapping spaces on the system bus, as follows:

- SRAM bitband: Starting address 0x1200_0000;
- AHB bitband: Starting address 0x2200_0000;
- APB bitband: Starting address 0x3200_0000;

The entire SRAM area supports bitband operations. Bitband operations for AHB and APB peripherals depend on the implementation of each peripheral module; refer to the peripheral module specifications for details.

3.4.4 Master Security Attributes

AHB bus masters include secure system masters and non-secure system masters. All master accesses to peripherals are controlled by the peripheral's security attribute configuration.

- Secure Masters
  - Core0
  - DMA00 Read
  - DMA00 Write
  - DMA01 Read
  - DMA01 Write
- Non-Secure Masters
  - Core1
  - DMA10 Read
  - DMA10 Write
  - DMA11 Read
  - DMA11 Write

4 System Control

4.1 Power Management Unit (PMU)

4.1.1 Power Scheme

4.1.1.1 Power Domain Partitioning

- DVCC Power Domain
- LDO15 Power Domain
- AVCC Power Domain
- VBAT Power Domain
- LDO_VBAT Power Domain

4.1.1.2 Power Structure Diagram

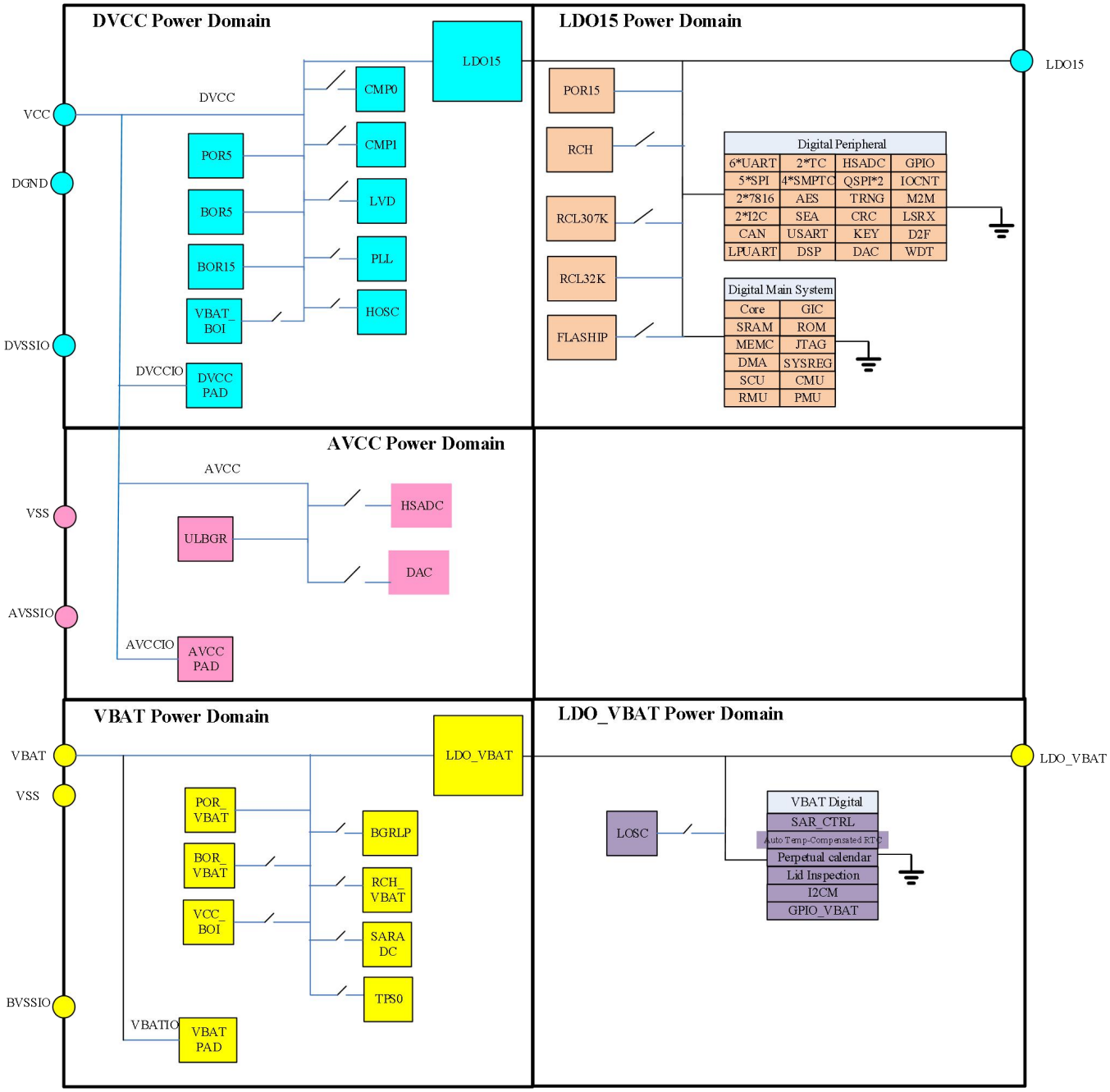


Figure 4-1 Power Structure Diagram

4.1.2 Power Control

Power Domain	Module	Module Power Switchable	Power-on Default State	Functional Description
DVCC	HOSC	√	OFF	System Clock Source
	PLL	√	OFF	System Clock Source
	LDO15	×, Always On	ON	Digital Domain Power Supply
	POR5	×, Always On	ON	DVCC First Power-on Reset
	BOR5	√	ON	DVCC Power-up/Power-down Reset
	BOR15	√	ON	LDO15 Power-up/Power-down Reset
	VBAT_BOI	√	ON	VBAT Domain Power Monitoring
	LVD	√	OFF	Comparator, can monitor DVCC power supply
	CMP0	√, Comparator and resistor bank controlled separately	ON	Low-power comparator, used for overall meter power loss detection
	CMP1	√	OFF	Low-power comparator, reserved
LDO15	DIG_VCC	×, Always On	ON	All digital except FLASH in DVCC domain
	FLASH	√	ON	FLASH
	POR15	×, Always On	ON	LDO15 First Power-on Reset
	RCH	√	ON	58.9824MHz internal system clock source, default 32-division to 1.8432MHz
	RCL307K	√	OFF	307KHz internal clock, source for IR 38K clock
	RCL32K	×, Always On	ON	32.768kHz backup clock, WDT clock, system clock in low-power mode, RTC backup clock
AVCC	ULBGR	×, Always On	ON	Low-power reference, poor performance
VBAT	BGRLP	√	ON	VBAT domain SARADC reference
	LDO_VBAT	×, Always On	ON	VBAT digital domain power supply
	POR_VBAT	×, Always On	ON	LDO_VBAT First Power-on Reset
	BOR_VBAT	√	ON	VBAT Power-up/Power-down Reset Module
	VCC_BOI	√	ON	Monitors DVCC domain power supply
	RCH_VBAT	√	OFF	VBAT domain high-frequency clock, for VBAT domain temperature compensation digital circuits, etc.
	SARADC	√	OFF	SARADC, for RTC temperature compensation, battery voltage monitoring,

				etc.
	TPS0	√	OFF	Temperature Sensor 1, can be used for RTC automatic temperature compensation
LDO_VBAT	LOSC	√	ON	32.768k crystal clock, RTC&PLL clock source
	DIG_VBAT	×, Always On	ON	VBAT domain digital modules, perpetual calendar, automatic temperature compensation, open cover detection, GPIO_VBAT, etc.
DVCCIO	DVCC domain PAD	×, Always On	ON	DVCC domain digital IO
AVCCIO	AVCC domain PAD	×, Always On	ON	AVCC domain analog input/output
VBATIO	VBAT domain PAD	×, Always On	ON	VBAT domain IO

4.1.3 Power Monitoring

4.1.3.1 External Power Monitoring

- CMP0/1, Mains detection
- Supports battery voltage detection

4.1.3.2 Internal Power Monitoring (DVCC Domain)

- LVD
- POR5
- BOR5
- POR15
- BOR15
- VBAT_BOI

4.1.3.3 Internal VBAT Power Monitoring (VBAT Domain)

- POR_VBAT
- BOR_VBAT
- VCC_BOI

4.2 Clock Management Unit (CMU)

The clock module is primarily used to generate the clocks required for dual-core operation. The dual cores operate using synchronous frequency division.

System clock switching and division control are handled by Core0 to ensure synchronization of the dual-core clock system. To ensure safe and reliable access from Core0 to Core1, the clock frequency of the Core0 system must be less than or equal to that of the Core1 system.

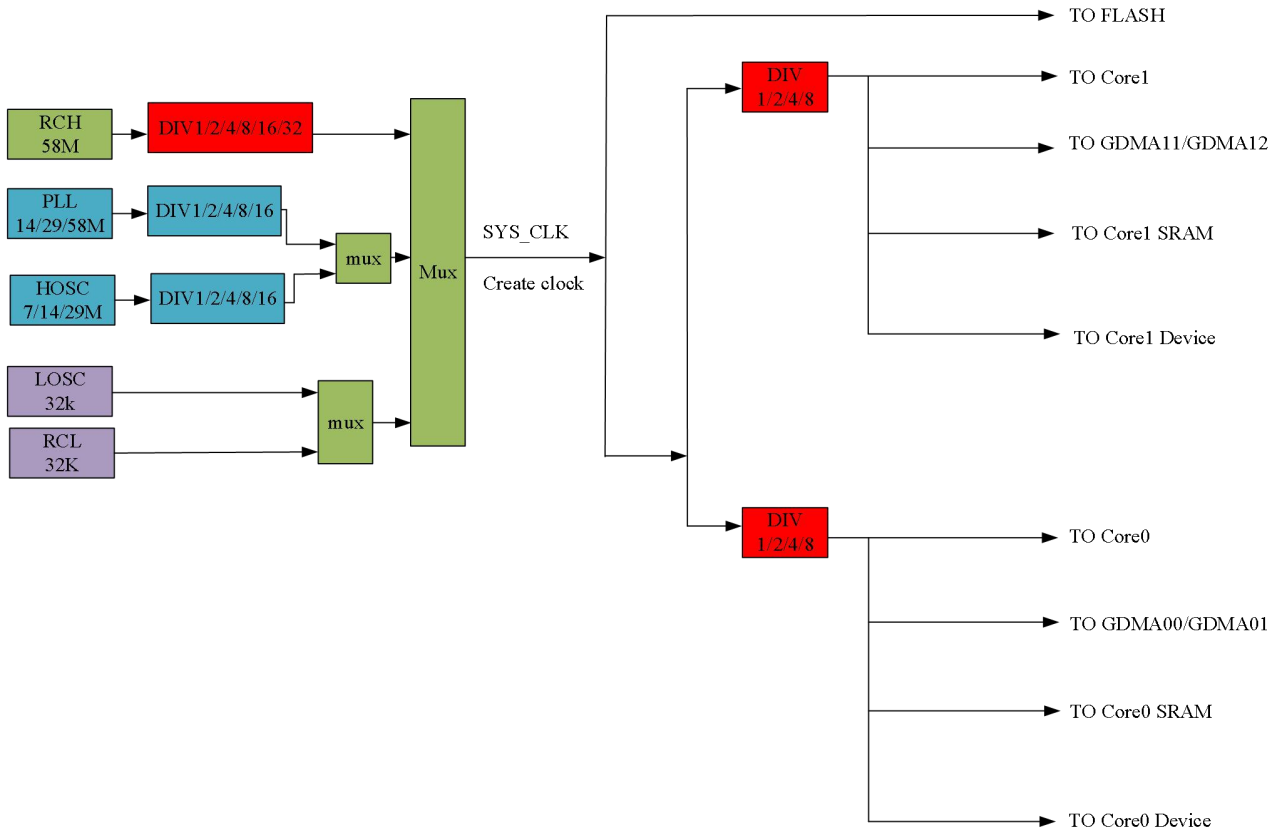


Figure 4-2 Clock Management Unit Block Diagram

4.3 Reset Management Unit (RMU)

The reset module is primarily used to generate the resets required for dual-core operation, including:

- VCC Power Domain Reset Sources mainly include:
  - Power Reset (POWEROK_RST)
  - PIN Reset (PIN_RST)
  - System Soft Reset (SOFT_RST, i.e., CPU Reset: Configure address 0xE0030FF0 value to 0x80000A5F for reset, other values release)
  - Watchdog Reset (WDT_RST)
  - Software Upgrade Reset (UPDATE_RST)
  - Peripheral Soft Reset (xx_RST_REQ)
- VBAT Power Domain Reset Sources mainly include:
  - Power-related Resets (BOI_VBAT_RST)

4.4 Power Consumption Modes

4.4.1 Overview

The chip's operating states are divided into two main categories: Run Modes and Low Power Modes:

- Run Modes can use different system operating clocks. Based on the system operating clock, they are divided into three working modes: HCM, RCM, and LCM. Users can switch working modes by operating the SYS_MODE register.

- Low Power Modes. Other peripheral clocks are turned off by the program.

Many modules have individual clock/power switches that can be turned off during run mode or before entering low power mode. The clock/power states are shown in the table below:

Main Module	Default Operating State
1.5V	
Core0 Kernel	On, clock can be turned off, never powered down
Core1 Kernel	On, clock can be turned off, never powered down
Interrupt System	On, clock can be turned off, never powered down
SRAM	On, clock can be turned off, never powered down
ROM	On, clock can be turned off, never powered down
FLASH	On, can be automatically powered down after CPU sleep
RTC	On, perpetual calendar cannot be turned off
Other Peripherals	Off, clock can be turned off, never powered down
5V	
ADC0	Off, power can be turned off
DAC	
BGRLP	Off, power can be turned off
ULBGR	
1.5V LDO	On, cannot be turned off
RCL	On, cannot be turned off
RCH	On, power can be turned off
CMP0	On, power can be turned off
SARADC	Turned on periodically
LVD	Off, power can be turned off
CMP1	Off, power can be turned off
BOR5 and BOR15	Always on
POR5 and POR15	Always on
PLL	Off, power can be turned off
HOSC	Off, power can be turned off
LOSC	Always on

4.4.2 HCM Mode

- HCM is the High-Frequency Mode. In this mode, the system operating clock uses either HOSC or PLL as the

clock source.

- In Run Mode, peripheral power and clocks can be turned on or off as needed.

4.4.3 RCM Mode

RCM is the RC Mode. This is the default state after system power-on, where the default system operating clock uses the 32-division of RCH58M.

4.4.4 LCM Mode

LCM is the Low-Frequency Mode. In this mode, the system operating clock source uses LOSC or RCL, and the system operating clock frequency is 32768Hz.

- Battery-powered operation requires working in low-frequency mode.
- System control (clock, reset, power management, mode control) and RTC timing sections are always powered, and the clock is always on and working.
- The power and clock for the RTC calibration section (temperature sensor, SAR ADC, RTC calibration calculation circuit) are turned on periodically according to configuration.
- The CPU minimal system (core, SRAM, or FLASH) is always powered; the clock is optional.
- IO ports are always powered to respond to wake-up events.
- Related circuits for peripheral circuits (such as analog parts, 1.5V digital parts) are optional. Since the system clock frequency is too low, these modules cannot work normally and are recommended to be turned off.
- The CPU cannot execute instructions until a reset or interrupt arrives; after wake-up, it runs at the system clock.

4.4.5 Sleep Mode

Executing the WFI/WFE instruction causes the system to enter the SLEEP low-power state.

4.5 Sleep and Wake-up Schemes

4.5.1 Overview

The typical application scenarios recommended for the dual-core architecture MCU are divided into the following three types: Single Core Core0 (Core0), Single Core Core1 (Core1), Dual Core Core0+Core1 (Core0+Core1). Corresponding to these three typical application scenarios, there are three sleep/wake-up schemes. This document details these three schemes.

4.5.2 Single Core Core0 Sleep/Wake-up Scheme

4.5.2.1 Block Diagram

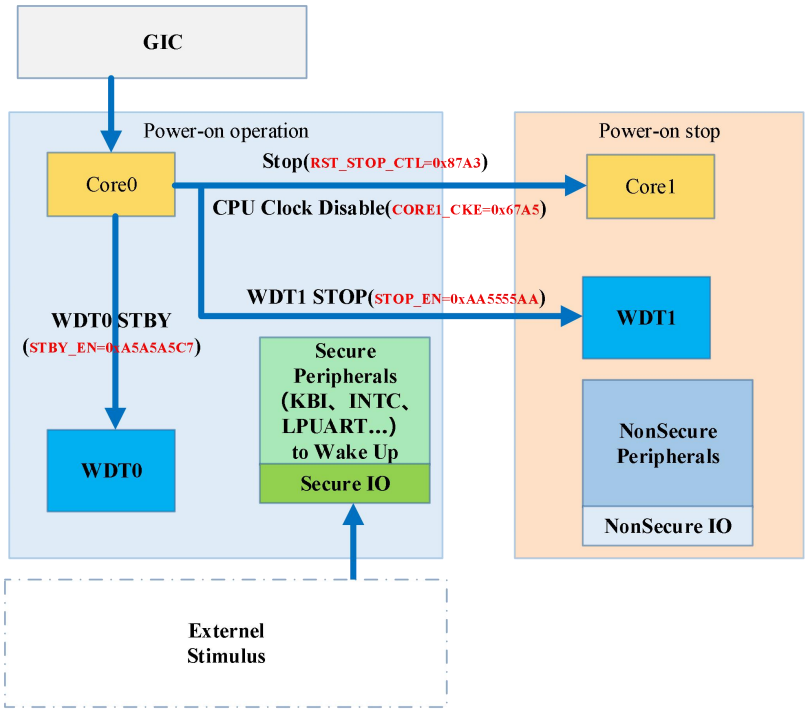


Figure 4-3 Single Core Core0 Sleep/Wake-up Scheme Block Diagram

4.5.2.2 Scheme Description

4.5.2.2.1 Pre-Sleep Preparation

- Core0 turns off all clock gating except for KBI0 (using KBI0 as an example, referring to the peripheral generating the wake-up interrupt), GPIO, and GIC.
- The GIC module assigns the KBI0 interrupt to Core0.
- GPIO PF01 pin is multiplexed as KBI0 (using KBI0 as an example, referring to the IO pin multiplexed for the peripheral generating the wake-up interrupt).
- Core0 enables the KBI0 module and KBI0 interrupt; Configure KBI module KBI_CTL.bit0=1, KBI_MASK.bit0=1.
- Core0 switches the system clock to 32KHz.

4.5.2.2.2 Entering Sleep

- After completing the above operations, the program executes "__WFI();" to enter sleep.

Table 4-1 Chip Module/CPU Status List during Single Core Core0 Sleep

Status Info	Core0	Core1	GPIO	WDT0	WDT1	FLASH	Peripherals Waking Core0
Module/CPU	Sleep	Stop	√	STBY (×)	STOP (×)	DeepSTBY	√
Clock	√	×	√	×	×	√	√

4.5.2.2.3 Wake-up

- Using KBI0 as an example, apply an external stimulus input on the PF01 pin (Note: Level duration must exceed the 32KHz clock period and filter duration).
- Set a software wake-up flag in the interrupt service routine (using KBI0 as an example, KBI0_HANDLER0).
- Query the software wake-up flag, switch from 32KHz to the operating clock frequency, enable each module's clock gating, and complete the wake-up operation.

Table 4-2 Chip Module/CPU Status List after Single Core Core0 Wake-up

Status Info	Core0	Core1	GPIO	WDT0	WDT1	FLASH	Peripherals Waking Core0
Module/CPU	√	Stop	√	√	STOP (×)	√	√
Clock	√	×	√	√	×	√	√

4.5.2.3 Sleep/Wake-up Flowchart

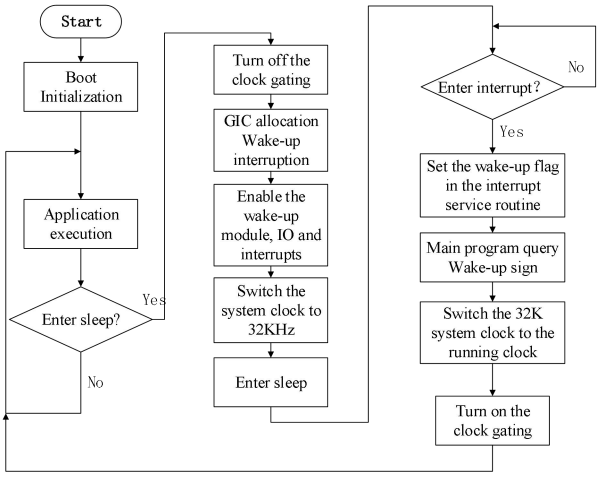


Figure 4-4 Single Core Core0 Sleep/Wake-up Flowchart

4.5.2.4 Interrupts

Table 4-3 Interrupt List Used in Single Core Core0 Sleep/Wake-up Scheme

Interrupt Event	Interrupt Target	Flag Bit	Enable Bit
KBI0 Channel 0 Interrupt (using KBI0 as example)	Core0	KBI_STA.bit0	KBI_MASK.bit0

4.5.3 Single Core Core1 Sleep/Wakeup Scheme

4.5.3.1 Structure Diagram

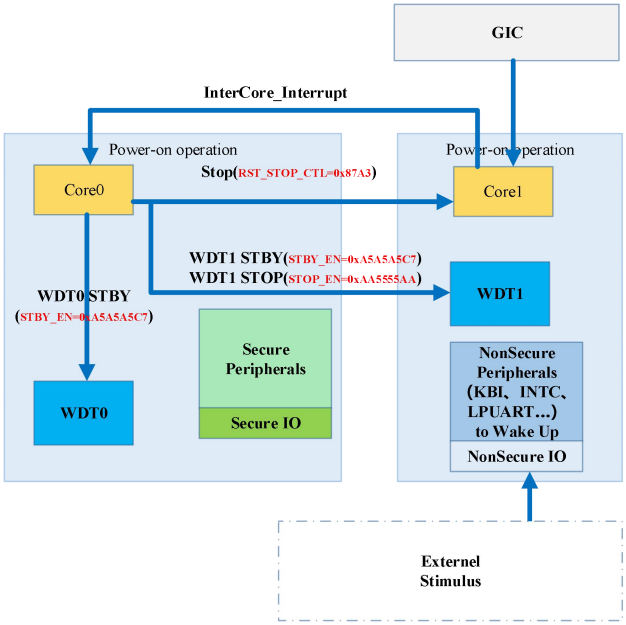


Figure 4-5 Single Core Core1 Sleep/Wakeup Scheme Structure Diagram

4.5.3.2 Scheme Description

4.5.3.2.1 Pre-Sleep Preparation

- Core1 turns off all clock gating except for KBI0 (using KBI0 as an example, referring to the peripheral generating the wakeup interrupt), GPIO, and GIC.
- The GIC module assigns the KBI0 interrupt to Core1.
- GPIO PF01 pin is multiplexed as KBI0 (using KBI0 as an example, referring to the IO pin 复用 for the peripheral generating the wakeup interrupt).
- Core1 enables the KBI0 module and the KBI0 interrupt.
  Configure the KBI module: KBI_CTL.bit0=1, KBI_MASK.bit0=1.
- After completing the above actions, Core1 notifies Core0 to perform clock switching and enter sleep.
- Core0 turns off all clock gating except for the GIC module.
- Core0 switches the system clock to 32KHz.
- After completing the system clock switch, Core0, before entering sleep, notifies Core1 that it has received the sleep instruction and is preparing to sleep.

4.5.3.2.2 Entering Sleep

- After completing the above operations, the Core1 program executes "__WFI();" to enter sleep.
- The Core0 program executes "__WFI();" to enter sleep.

Table 4-4: Chip Module/CPU Status List during Single Core Core1 Sleep

Status Info	Core0	Core1	GPIO	WDT0	WDT1	FLASH	Peripheral Waking Core1
Module/CPU	Sleep	Sleep	√	STBY（×）	STBY（×）	DeepSTBY	√
Clock	√	√	√	×	×	√	√

4.5.3.2.3 Wakeup

- Using KBI0 as an example, apply an external stimulus input on the PF01 pin (Note: The level duration must exceed the 32KHz clock period and the filter duration).
- In the Core1 Interrupt Service Routine (using KBI0 as an example, KBI0_HANDLER0), send an inter-core interrupt to Core0, enable the clock gating for various modules, and complete the wakeup operation.
- In the Core0 inter-core interrupt service routine, switch from 32KHz to the main operating clock frequency, notify Core1 via inter-core communication that it is awake, enable the clock gating for various modules, and complete the wakeup operation.

Table 4-5: Chip Module/CPU Status List after Single Core Core1 Wakeup

Status Info	Core0	Core1	GPIO	WDT0	WDT1	FLASH	Peripheral Waking Core1
Module/CPU	√	√	√	√	√	√	√
Clock	√	√	√	√	√	√	√

4.5.4 Dual Core Sleep/Wakeup Scheme

4.5.4.1 Structure Diagram

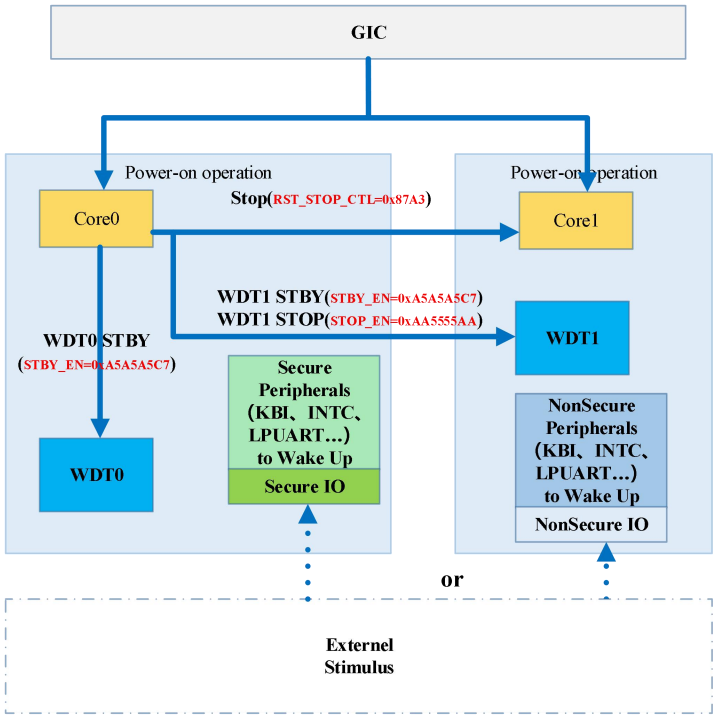


Figure 4-6 Dual Core Sleep/Wakeup Scheme Structure Diagram

4.5.4.2 Scheme Description

4.5.4.2.1 Pre-Sleep Preparation

- Core1 turns off all clock gating except for KBI0 (using KBI0 as an example, referring to the peripheral generating the wakeup interrupt), GPIO, and GIC.
- The GIC module assigns the KBI0 interrupt to both Core1 and Core0.

- GPIO PF01 pin is multiplexed as KBI0 (using KBI0 as an example, referring to the IO pin 复用 for the peripheral generating the wakeup interrupt).
- Core1 enables the KBI0 module and the KBI0 interrupt.
  Configure the KBI module: KBI_CTL.bit0=1, KBI_MASK.bit0=1.
- After completing the above actions, Core1 notifies Core0 to perform clock switching and enter sleep.
- Core0 turns off all clock gating except for the GIC module.
- Core0 switches the system clock to 32KHz.
- After completing the system clock switch, Core0, before entering sleep, notifies Core1 that it has received the sleep instruction and is preparing to sleep.

4.5.4.2.2 Entering Sleep

- After completing the above operations, the Core1 program executes "__WFI();" to enter sleep.
- The Core0 program executes "__WFI();" to enter sleep.

Table 4-6: Chip Module/CPU Status List during Dual Core Sleep

Status Info	Core0	Core1	GPIO	WDT0	WDT1	FLASH	Wakeup Peripheral
Module/CPU	Sleep	Sleep	√	STBY（×）	STBY（×）	DeepSTBY	√
Clock	√	√	√	×	×	√	√

4.5.4.2.3 Wakeup

- Using KBI0 as an example, apply an external stimulus input on the PF01 pin (Note: The level duration must exceed the 32KHz clock period and the filter duration).
- In the Core1 Interrupt Service Routine (using KBI0 as an example, KBI0_HANDLER0), enable the clock gating for various modules and complete the wakeup operation.
- Simultaneously with step 2, in the Core0 interrupt service routine, switch from 32KHz to the main operating clock frequency, enable the clock gating for various modules, and complete the wakeup operation.

Table 4-7: Chip Module/CPU Status List after Dual Core Wakeup

Status Info	Core0	Core1	GPIO	WDT0	WDT1	FLASH	Wakeup Peripheral
Module/CPU	√	√	√	√	√	√	√
Clock	√	√	√	√	√	√	√

4.5.4.3 Interrupts

Table 4-8: Interrupt List Used in Single Core Core1 Sleep/Wakeup Scheme

Interrupt Event	Interrupt Target	Flag Bit	Enable Bit
KBI0 Channel 0 Interrupt (using KBI0 as example)	Core1	KBI_STA.bit0	KBI_MASK.bit0

4.6 Core0 System Control Registers

4.6.1 Password Class Registers

Register Function Class	Physical/Mapped Address	Function Class Address Offset
PUB_PSW_TYPE	0x2008_0000	0x0000

Register Name	Register Address Offset	Description
SYS_PSW1	0x0000	Public Password Register

4.6.1.1 SYS_PSW1（0x0000）

Public Password Register

Bit	Name	Description	R/W Flag	Reset Value
31:8	---	Reserved	R	0
7:0	SYS_PSW1	When SYS_PSW1=8'h82, public registers are writable; When SYS_PSW1=other values, public registers are not writable; This register reads back the written value. It is recommended that the user disable write enable immediately after completing the write operation.	RW	0x00

4.6.2 Clock Control Class Registers

Register Function Class	Physical/Mapped Address	Function Class Address Offset
CLK_CTL_TYPE	0x2008_0000	0x0100
Register Name	**Register Address Offset**	**Description**
RCH_CTL	0x0100	RCH Control Register
PLL_CTL	0x0104	PLL Control Register
HOSC_CTL	0x0108	HOSC Control Register
RCL307K_CTL	0x010C	RCL307K Control Register
CLK_SEL	0x0120	Clock Selection Register
SYS_MODE	0x0124	System Mode Register
CORE_DIV	0x0128	CORE Clock Division Register
PMOD0_CKE	0x012C	Private Module Clock Enable Register
CORE1_CKE	0x0130	CORE1 System Clock Control Register
KBI0_CKE	0x0134	KBI0 Clock Enable Register
KBI1_CKE	0x0138	KBI1 Clock Enable Register
MOD0_CKE	0x013C	Module Clock Enable Register 0
MOD1_CKE	0x0140	Module Clock Enable Register 1
MOD2_CKE	0x0144	Module Clock Enable Register 2
SYSFREQ_STA	0x0170	System Clock Frequency Indication Register
SYSCLK_STA	0x0174	System Clock Status Register

4.6.2.1 RCH_CTL（0x0100）

RCH Control Register

Bit	Name	Description	R/W Flag	Reset Value
31:4	---	Reserved	R	0x0
3:1	RCH_DIV	RCH source clock division selection: 000: 1 division;	RW	0x5

		001: 2 division; 010: 4 division; 011: 8 division; 100: 16 division; 101: 32 division (default); Others: Reserved. Note: The operating frequency under RCH is only related to the configuration of this register, and supports changing the RCH_DIV value in RCM mode.		
0	RCH_PD	Internal RCH enable bit, main frequency 58.9824MHz: 0: On; 1: Off.	RW	0x0

4.6.2.2 PLL_CTL（0x0104）

PLL Control Register

Bit	Name	Description	R/W Flag	Reset Value
31:9	---	Reserved	R	0x0
8	PLL_LOCK	PLL lock status 0: Not locked 1: Locked Note: Regardless of PLL startup or PLL online frequency adjustment, application requires adding at least 2ms hard delay before detecting PLL lock status and switching clocks. Sample testing needs to confirm this value.	R	0x0
7	---	Reserved	R	0x0
6:4	PLL_DIV	PLL source clock division selection: 000: 1 division; 001: 2 division; 010: 4 division; 011: 8 division; 100: 16 division; Others: Reserved.	RW	0x0
3	---	Reserved	R	0x0
2:1	PLL_FREQ	00: PLL output frequency selection 7.3728MHz; 01: PLL output frequency selection 14.7456MHz; 10: PLL output frequency selection 29.4912MHz; 11: PLL output frequency selection 58.9824MHz;	RW	0x1
0	PLL_PD	PLL module enable bit 0: On 1: Off	RW	0x1

4.6.2.3 HOSC_CTL（0x0108）

HOSC Control Registe

Bit	Name	Description	R/W Flag	Reset Value
31:15	---	Reserved	R	0
14:12	HOSC_DSEL	HOSC analog filter time configuration: =000: 25.5ns, recommended configuration for main frequency 7.3728MHz =001: 10.5ns, recommended configuration for main frequency 14.7456MHz =01x: 5ns, recommended configuration for main frequency 29.4912MHz =1xx: HOSC clock without analog filtering	RW	0x0
11	---	Reserved	R	0x0
10:8	HOSC_PGA	HOSC gain selection 000: Recommended configuration for main frequency 7.3728MHz; 001: Recommended configuration for main frequency 14.7456MHz; 010: Recommended configuration for main frequency 29.4912MHz;	RW	0x0
7	---	Reserved	R	0x0
6:4	HOSC_DIV	HOSC source clock division selection: 000: 1 division; 001: 2 division; 010: 4 division; 011: 8 division; 100: 16 division; Others: Reserved.	RW	0x0
3	---	Reserved	R	0x0
2:1	HOSC_FREQ	00: External high-frequency crystal 7.3728MHz 01: External high-frequency crystal 14.7456MHz 10: External high-frequency crystal 29.4912MHz Others: Reserved	RW	0x0
0	HOSC_PD	External high-frequency oscillator enable bit: 0: On 1: Off	RW	0x1

4.6.2.4 RCL307K_CTL（0x010C）

RCL307K Control Register

Bit	Name	Description	R/W Flag	Reset Value
31:8	---	Reserved	R	0x0

7:0	RCL307K_PD	RCL307K enable switch 0x75: RCL307K off Others: RCL307K on	RW	0x75

4.6.2.5 CLK_SEL（0x0120）

Clock Selection Register

Bit	Name	Description	R/W Flag	Reset Value
31:12	---	Reserved	R	0
11	CORE1_RCL_LOSC_SEL	CORE1 low-frequency clock source selection 0: Filter clock selects LOSC; 1: Filter clock selects RCL32K;	RW	0x1
10	CORE0_RCL_LOSC_SEL	CORE0 low-frequency clock source selection 0: Filter clock selects LOSC; 1: Filter clock selects RCL32K;	RW	0x1
9	D2F_RCL_LOSC_SEL	D2F low-frequency clock source selection 0: Filter clock selects LOSC; 1: Filter clock selects RCL32K;	RW	0x0
8	VBAT_RCL_SEL	RCL32K clock channel switch to VBAT domain 0: RCL32K clock channel closed; 1: RCL32K clock channel opened; Note: First switch the VBAT domain circuit clock to RCL32K, then enable this register to prevent glitches during clock switching; When switching VBAT domain circuit clock to LOSC, need to close this register first.	RW	0x0
7	LPUART_RCL_LOSC_SEL	LPUART low-frequency clock source selection 0: Filter clock selects LOSC; 1: Filter clock selects RCL32K;	RW	0x0
6	KBI_RCL_LOSC_SEL	KBI filter clock source selection 0: Filter clock selects LOSC; 1: Filter clock selects RCL32K;	RW	0x0
5:4	IRFR_RCL_LOSC_SEL	00: IRFR selects LOSC external low-frequency crystal as clock source; 01: IRFR selects RCL32k internal low-frequency crystal as clock source; 10: IRFR selects RCL307k internal low-frequency crystal as clock source; 11: Reserved, not recommended configuration;	RW	0x0
3:2	LCD_RCL_LOSC_SEL	00: LCD selects LOSC external low-frequency crystal as clock source; 01: LCD selects RCL32k internal low-frequency crystal as clock source; Others: Reserved, not recommended configuration;	RW	0x0

1	SYS_PLL_HOSC_SEL	High-frequency runtime system main clock selection: 0: Select PLL output as system main clock; 1: Select high-frequency crystal as system main clock. This configuration can only be set in RC mode and low-frequency mode.	RW	0x0
0	SYS_RCL_LOSC_SEL	Low-frequency runtime system main clock selection: 0: Select LOSC external low-frequency crystal as clock source; 1: Select RCL32K internal low-frequency crystal as clock source; This register can only be modified in RC mode or high-frequency mode.	RW	0x0

4.6.2.6 SYS_MODE（0x0124）

System Mode Register

Bit	Name	Description	R/W Flag	Reset Value
31:5	---	Reserved	R	0x0
4	FLASH_BUSY	Flash busy status, cannot enter mode switch: 0: idle 1: busy	R	0x0
3:0	SYS_MODE	Write D, set to enter high-frequency mode HCM, read 4'b0100; Write E, set to enter RC mode RCM, read 4'b0010; Write F, set to enter 32.768KHz mode LCM, read 4'b0001; That is, the register read value is: {0,HCM,RCM,LCM} Note: flash busy, test mode 1 and test mode 2 cannot write this register	RW	0x2

Note: The current mode status indication should read the LOSC_ON, IRCH_ON, PLL_ON/HOSC_ON these three statuses. Not by reading this register, this register only represents the mode switch command written, does not represent that has switched to the expected mode.

4.6.2.7 CORE_DIV（0x0128）

CORE Clock Division Register

Bit	Name	Description	R/W Flag	Reset Value
31:4	--	Reserved	R	0x0
3:0	CORE1_CORE0_DIV	Generate CORE0/CORE1 clocks from system clock via CORE0/CORE1 master clock divider configuration (32kHz does not support division and frequency after division cannot be lower than 1.8432MHz) (CORE0 does not support 58.9824MHz frequency point) (To prevent illegal clock divider configuration, this register does not support bitband operation):	RW	0x0

		0000: CORE1 no division, CORE0 no division; 0001: CORE1 no division, CORE0 divide by 2; 0010: CORE1 no division, CORE0 divide by 4; 0011: CORE1 no division, CORE0 divide by 8; 0101: CORE1 divide by 2, CORE0 divide by 2; 0110: CORE1 divide by 2, CORE0 divide by 4; 0111: CORE1 divide by 2, CORE0 divide by 8; 1010: CORE1 divide by 4, CORE0 divide by 4; 1011: CORE1 divide by 4, CORE0 divide by 8; 1111: CORE1 divide by 8, CORE0 divide by 8; Others: Prohibited. Cannot write other values.		

4.6.2.8 PMOD0_CKE（0x012C）

Private Module Clock Enable Register

Bit	Name	Description	R/W Flag	Reset Value
31:15	--	Reserved	R	0x0
14	CRC1_CKE	CRC1 module clock enable: 0: Clock stopped; 1: Clock started	RW	0x0
13	SYS_TC_CKE	SYS_TC module clock enable: 0: Clock stopped; 1: Clock started	RW	0x0
12	LCSS_CKE	LCSS module enable, clock gating: 0: Clock stopped; 1: Clock started	RW	0x0
11	GPIO_CKE	GPIO module enable, clock gating: 0: Clock stopped; 1: Clock started	RW	0x0
10	UART38K_CKE	UART38K module clock enable: 0: Clock stopped; 1: Clock started	RW	0x0
9	GDMA01_CKE	GDMA01 module clock enable: 0: Clock stopped; 1: Clock started	RW	0x0
8	GDMA00_CKE	GDMA00 module clock enable: 0: Clock stopped; 1: Clock started	RW	0x0
7:5	--	Reserved	R	0x0
4	CORE0_GIC_CKE	CORE0_GIC module clock enable: 0: Clock stopped; 1: Clock started	RW	0x0
3	CORE0_SCU_CKE	CORE0_SCU module clock enable:	RW	0x0

		0: Clock stopped; 1: Clock started		
2	CORE0_AOS_CKE	CORE0_AOS module clock enable: 0: Clock stopped; 1: Clock started	RW	0x0
1	CORE0_MAILBOX_CKE	CORE0_MAILBOX module clock enable: 0: Clock stopped; 1: Clock started	RW	0x0
0	CORE0_WDT_CKE	CORE0_WDT apb clock enable, synchronously turned off when CORE0 enters sleep: 0: Clock stopped; 1: Clock started	RW	0x0

4.6.2.9 CORE1_CKE（0x0130）

CORE1 System Clock Control Register

Bit	Name	Description	R/W Flag	Reset Value
31:16	---	Reserved	R	0x0
15:0	CORE1_CKE	CORE1 System Clock Gating: 0x67a5: CORE1 core clock disabled; 0x675a: CORE1 core and CORE1 system bus disabled; Other: CORE1 system clock enabled Notes: 1. When using single-core CORE0, configure CORE1_CKE=0x67a5 to disable the core clock for power reduction; 2. In dual-core applications, enable CORE1_CKE before releasing CORE1 (RST_STOP_CTL.CORE1_STOP != 0x87a3).	RW	0x0

4.6.2.10 KBI0_CKE（0x0134）

KBI0 Clock Enable Register

Bit	Name	Description	R/W Flag	Reset Value
31	KBIC0_CKE	KBIC0 apb module clock enable: 0: Clock stopped; 1: Clock started	RW	0x0
30:16	--	Reserved	R	0x0
15	KBI015_CKE	KBI015 module clock enable: 0: Clock stopped; 1: Clock started	RW	0x0
14	KBI014_CKE	KBI014 module clock enable: 0: Clock stopped; 1: Clock started	RW	0x0

13	KBI013_CKE	KBI013 module clock enable: 0: Clock stopped; 1: Clock started	RW	0x0
12	KBI012_CKE	KBI012 module clock enable: 0: Clock stopped; 1: Clock started	RW	0x0
11	KBI011_CKE	KBI011 module clock enable: 0: Clock stopped; 1: Clock started	RW	0x0
10	KBI010_CKE	KBI010 module clock enable: 0: Clock stopped; 1: Clock started	RW	0x0
9	KBI09_CKE	KBI09 module clock enable: 0: Clock stopped; 1: Clock started	RW	0x0
8	KBI08_CKE	KBI08 module clock enable: 0: Clock stopped; 1: Clock started	RW	0x0
7	KBI07_CKE	KBI07 module clock enable: 0: Clock stopped; 1: Clock started	RW	0x0
6	KBI06_CKE	KBI06 module clock enable: 0: Clock stopped; 1: Clock started	RW	0x0
5	KBI05_CKE	KBI05 module clock enable: 0: Clock stopped; 1: Clock started	RW	0x0
4	KBI04_CKE	KBI04 module clock enable: 0: Clock stopped; 1: Clock started	RW	0x0
3	KBI03_CKE	KBI03 module clock enable: 0: Clock stopped; 1: Clock started	RW	0x0
2	KBI02_CKE	KBI02 module clock enable: 0: Clock stopped; 1: Clock started	RW	0x0
1	KBI01_CKE	KBI01 module clock enable: 0: Clock stopped; 1: Clock started	RW	0x0
0	KBI00_CKE	KBI00 module clock enable: 0: Clock stopped; 1: Clock started	RW	0x0

4.6.2.11 KBI1_CKE（0x0138）

KBI1 Clock Enable Register

Bit	Name	Description	R/W Flag	Reset Value
31	KBIC1_CKE	KBIC1 apb module clock enable: 0: Clock stopped; 1: Clock started	RW	0x0
30:16	--	Reserved	R	0x0
15	KBI115_CKE	KBI115 module clock enable: 0: Clock stopped; 1: Clock started	RW	0x0
14	KBI114_CKE	KBI114 module clock enable: 0: Clock stopped; 1: Clock started	RW	0x0
13	KBI113_CKE	KBI113 module clock enable: 0: Clock stopped; 1: Clock started	RW	0x0
12	KBI112_CKE	KBI112 module clock enable: 0: Clock stopped; 1: Clock started	RW	0x0
11	KBI111_CKE	KBI111 module clock enable: 0: Clock stopped; 1: Clock started	RW	0x0
10	KBI110_CKE	KBI110 module clock enable: 0: Clock stopped; 1: Clock started	RW	0x0
9	KBI19_CKE	KBI19 module clock enable: 0: Clock stopped; 1: Clock started	RW	0x0
8	KBI18_CKE	KBI18 module clock enable: 0: Clock stopped; 1: Clock started	RW	0x0
7	KBI17_CKE	KBI17 module clock enable: 0: Clock stopped; 1: Clock started	RW	0x0
6	KBI16_CKE	KBI16 module clock enable: 0: Clock stopped; 1: Clock started	RW	0x0
5	KBI15_CKE	KBI15 module clock enable: 0: Clock stopped; 1: Clock started	RW	0x0
4	KBI14_CKE	KBI14 module clock enable: 0: Clock stopped;	RW	0x0

		1: Clock started		
3	KBI13_CKE	KBI13 module clock enable: 0: Clock stopped; 1: Clock started	RW	0x0
2	KBI12_CKE	KBI12 module clock enable: 0: Clock stopped; 1: Clock started	RW	0x0
1	KBI11_CKE	KBI11 module clock enable: 0: Clock stopped; 1: Clock started	RW	0x0
0	KBI10_CKE	KBI10 module clock enable: 0: Clock stopped; 1: Clock started	RW	0x0

4.6.2.12 MOD0_CKE（0x013C）

Module Clock Enable Register 0

Bit	Name	Description	R/W Flag	Reset Value
31:30	--	Reserved	R	0x0
29	TC1_CKE	TC1 module clock enable: 0: Clock stopped; 1: Clock started	RW	0x0
28	TC0_CKE	TC0 module clock enable: 0: Clock stopped; 1: Clock started	RW	0x0
27	DAC_CKE	DAC module clock enable: 0: Clock stopped; 1: Clock started	RW	0x0
26	ADC0_CKE	ADC0 module clock enable: 0: Clock stopped; 1: Clock started	RW	0x0
25	D2F_CKE	D2F module clock enable: 0: Clock stopped; 1: Clock started	RW	0x0
24	M2M_CKE	M2M module clock enable: 0: Clock stopped; 1: Clock started	RW	0x0
23	--	Reserved	RW	0x0
22	DSP_CKE	DSP module clock enable: 0: Clock stopped; 1: Clock started	RW	0x0
21	LCD_CKE	LCD module clock enable:	RW	0x0

		0: Clock stopped; 1: Clock started		
20	IOCNT_CKE	IOCNT module clock enable: 0: Clock stopped; 1: Clock started	RW	0x0
19	--	Reserved	RW	0x0
18	I2C1_CKE	I2C1 module clock enable: 0: Clock stopped; 1: Clock started	RW	0x0
17	I2C0_CKE	I2C0 module clock enable: 0: Clock stopped; 1: Clock started	RW	0x0
16	SPI3_CKE	SPI3 module clock enable: 0: Clock stopped; 1: Clock started	RW	0x0
15	SPI2_CKE	SPI2 module clock enable: 0: Clock stopped; 1: Clock started	RW	0x0
14	SPI1_CKE	SPI1 module clock enable: 0: Clock stopped; 1: Clock started	RW	0x0
13	SPI0_CKE	SPI0 module clock enable: 0: Clock stopped; 1: Clock started	RW	0x0
12	ISO78161_CKE	ISO7816_1 module clock enable: 0: Clock stopped; 1: Clock started	RW	0x0
11	ISO78160_CKE	ISO7816_0 module clock enable: 0: Clock stopped; 1: Clock started	RW	0x0
10	USART1_CKE	USART1 module clock enable: 0: Clock stopped; 1: Clock started	RW	0x0
9	USART0_CKE	USART0 module clock enable: 0: Clock stopped; 1: Clock started	RW	0x0
8	LPUART_CKE	LPUART module clock enable: 0: Clock stopped; 1: Clock started	RW	0x0
7	UART7_CKE	UART5 module clock enable: 0: Clock stopped; 1: Clock started	RW	0x0
6	UART6_CKE	UART5 module clock enable:	RW	0x0

		0: Clock stopped; 1: Clock started		
5	UART5_CKE	UART5 module clock enable: 0: Clock stopped; 1: Clock started	RW	0x0
4	UART4_CKE	UART4 module clock enable: 0: Clock stopped; 1: Clock started	RW	0x0
3	UART3_CKE	UART3 module clock enable: 0: Clock stopped; 1: Clock started	RW	0x0
2	UART2_CKE	UART2 module clock enable: 0: Clock stopped; 1: Clock started	RW	0x0
1	UART1_CKE	UART1 module clock enable: 0: Clock stopped; 1: Clock started	RW	0x0
0	UART0_CKE	UART0 module clock enable: 0: Clock stopped; 1: Clock started	RW	0x0

4.6.2.13 MOD1_CKE（0x0140）

Module Clock Enable Register1

Bit	Name	Description	R/W Flag	Reset Value
31:11	--	Reserved	R	0x0
10	CRC0_CKE	CRC0 module clock enable: 0: Clock stopped; 1: Clock started	RW	0x0
9	LVDCMP_CKE	LVDCMP module clock enable: 0: Clock stopped; 1: Clock started	RW	0x0
8:7	--	Reserved	R	0x0
6	LSRX_CKE	LSRX module clock enable: 0: Clock stopped; 1: Clock started	RW	0x0
5	--	Reserved	R	0x0
4	ANA_CKE	RTC and SARADC bus clock enable: 0: Clock stopped; 1: Clock started	RW	0x0
3	SIMPTC3_CKE	SIMPTC3 module clock enable: 0: Clock stopped;	RW	0x0

		1: Clock started		
2	SIMPTC2_CKE	SIMPTC2 module clock enable: 0: Clock stopped; 1: Clock started	RW	0x0
1	SIMPTC1_CKE	SIMPTC1 module clock enable: 0: Clock stopped; 1: Clock started	RW	0x0
0	SIMPTC0_CKE	SIMPTC0 module clock enable: 0: Clock stopped; 1: Clock started	RW	0x0

4.6.2.14 MOD2_CKE（0x0144）

Module Clock Enable Register 2

Bit	Name	Description	R/W Flag	Reset Value
31:9	--	Reserved	R	0x0
8	AES_CKE	AES module clock enable: 0: Clock stopped; 1: Clock started	RW	0x0
7	KEY_STORE_CKE	KEY_STORE module clock enable: 0: Clock stopped; 1: Clock started	RW	0x0
6	SEA_SHA_CKE	SEA_SHA module clock enable: 0: Clock stopped; 1: Clock started	RW	0x0
5	SEA_TRNG_CKE	SEA_TRNG module clock enable: 0: Clock stopped; 1: Clock started	RW	0x0
4	SEA_AES_CKE	SEA_AES module clock enable: 0: Clock stopped; 1: Clock started	RW	0x0
3	SEA_PKE_CKE	SEA_PKE module clock enable: 0: Clock stopped; 1: Clock started	RW	0x0
2	SEA_BUS_CKE	SEA_BUS module clock enable: 0: Clock stopped; 1: Clock started	RW	0x0
1	QSPI1_CKE	QSPI1 module clock enable: 0: Clock stopped, module cleared; 1: Clock started, module enabled	RW	0x0
0	QSPI0_CKE	QSPI0 module clock enable: 0: Clock stopped, module cleared; 1: Clock started, module enabled	RW	0x0

4.6.2.15 SYSFREQ_STA（0x0170）

System Clock Frequency Indication Register

Bit	Name	Description	R/W Flag	Reset Value
31:11	--	Reserved	R	0x0
10:8	FLASH_CLK_FREQ	Flash system frequency indication: 000: Current system master clock is 32.768KHz; 001: Current system master clock is 1.8432MHz; 010: Current system master clock is 3.6864MHz; 011: Current system master clock is 7.3728MHz; 100: Current system master clock is 14.7456MHz; 101: Current system master clock is 29.4912MHz; 110: Current system master clock is 58.9824MHz; 111: Current system master clock configuration exceeds allowable operating frequency range; Others: Reserved;	R	0x1
7	--	Reserved	R	0x0
6:4	CORE1_SYS_FREQ	CORE1 core system master clock frequency indication: 000: Current system master clock is 32.768KHz; 001: Current system master clock is 1.8432MHz; 010: Current system master clock is 3.6864MHz; 011: Current system master clock is 7.3728MHz; 100: Current system master clock is 14.7456MHz; 101: Current system master clock is 29.4912MHz; 110: Current system master clock is 58.9824MHz; 111: Current system master clock configuration exceeds allowable operating frequency range; Others: Reserved;	R	0x1
3	--	Reserved	R	0x0
2:0	CORE0_SYS_FREQ	CORE0 core system master clock frequency indication: 000: Current system master clock is 32.768KHz; 001: Current system master clock is 1.8432MHz; 010: Current system master clock is 3.6864MHz; 011: Current system master clock is 7.3728MHz; 100: Current system master clock is 14.7456MHz; 101: Current system master clock is 29.4912MHz; 111: Current system master clock configuration exceeds allowable operating frequency range; Others: Reserved;	R	0x1

4.6.2.16 SYSCLK_STA（0x0174）

System Clock Status Register

Bit	Name	Description	R/W Flag	Reset Value
31:10	--	Reserved	R	0x0
9	RCL307K_ON_FLAG	RCL307K clock enable flag: 0: RCL307K clock not enabled; 1: RCL307K clock enabled	R	0x0
8	HOSC_ON_FLAG	HOSC clock enable flag: 0: HOSC clock not enabled; 1: HOSC clock enabled	R	0x0
7	RCL32K_ON_FLAG	RCL32K clock enable flag: 0: RCL32K clock not enabled; 1: RCL32K clock enabled	R	0x1
6	RCH_ON_FLAG	RCH clock enable flag: 0: RCH clock not enabled; 1: RCH clock enabled	R	0x1
5	PLL_ON_FLAG	PLL clock enable flag: 0: PLL clock not enabled; 1: PLL clock enabled	R	0x0
4	LOSC_ON_FLAG	LOSC clock enable flag: 0: LOSC clock not enabled; 1: LOSC clock enabled	R	0x1
3	--	Reserved	R	0x0
2	SYS_ON_PLL_HOSC	When system runs on external high frequency or internal PLL clock, this bit is 1; When system runs on other clocks, this bit is 0.	R	0x0
1	SYS_ON_RCH	When system runs on internal high frequency clock, this bit is 1; When system runs on other clocks, this bit is 0.	R	0x1
0	SYS_ON_LOSC_RCL	When system runs on external low frequency or internal RCL32K clock, this bit is 1; When system runs on other clocks, this bit is 0.	R	0x0

4.6.3 Clock Calibration Class Registers

Register Function Class	Physical/Mapped Address	Function Class Address Offset
CLK_CAL_TYPE	0x2008_0000	0x0200
Register Name	**Register Address Offset**	**Description**
CLK_CAL_CTL	0x0200	Clock Calibration Control Register
CLK_REF_CNT	0x0204	Target Count Register
CLK_CAL_CNT	0x0208	Return Count Register
CLK_CAL_STA	0x020C	Clock Calibration Status Register

4.6.3.1 CLK_CAL_CTL（0x0200）

Clock Calibration Control Register

Bit	Name	Description	R/W Flag	Reset Value
31:14	---	Reserved	R	0x0
13	CAL_OV_IE	Calibrated Clock Counter Overflow Flag Interrupt Enable: 0: Disable interrupt; 1: Enable interrupt;	RW	0x0
12	CAL_DONE_IE	Clock Calibration Complete Flag Interrupt Enable: 0: Disable interrupt; 1: Enable interrupt;	RW	0x0
11:8	CAL_CLK_SEL	> Clock Under Test Source Selection (All clocks are frequencies of analog clock sources): > 0000: Select LOSC; > 0001: Select RCL_32K; > 0010: Select HOSC; > 0011: Select RCH_VCC; > 0100: Select PLL; > 0101: Select RCH_VBAT; > 0110: Select RCL_307K; > Others: Reverse;	RW	0xF
7:4	REF_CLK_SEL	> Reference Clock Source Selection (All clocks are frequencies of analog clock sources): > 0000: Select LOSC; > 0001: Select RCL_32K; > 0010: Select HOSC; > 0011: Select RCH_VCC; > 0100: Select PLL; > 0101: Select RCH_VBAT; > 0110: Select RCL_307K; > Others: Reverse;	RW	0xF
3	---	Reserved	R	0x0
2	CAL_SEL	Clock Calibration Source Selection: 0: CORE0 software trigger clock calibration; 1: AOS trigger clock calibration;	RW	0x0
1	CAL_STOP	Clock Calibration Stop Bit: 0: No operation; 1: Stop clock calibration; Note: If stopping clock calibration, this bit must be written to 0 before restarting clock calibration.	RW	0x0
0	CAL_START	Clock Calibration Start Bit: 0: No operation; 1: Start clock calibration;	RW	0x0

		Note: This bit is automatically cleared when clock calibration completes or is stopped.		

4.6.3.2 CLK_REF_CNT（0x0204）

Target Count Register

Bit	Name	Description	R/W Flag	Reset Value
31:24	---	Reserved	R	0x0
23:0	REF_CLK_CNT	Reference Clock Count Value	RW	0x10000

4.6.3.3 CLK_CAL_CNT（0x0208）

Return Count Register

Bit	Name	Description	R/W Flag	Reset Value
31:24	---	Reserved	R	0x0
23:0	CAL_CLK_CNT	Calibrated Clock Returned Count Value	R	0x0

4.6.3.4 CLK_CAL_STA（0x020C）

Clock Calibration Status Register

Bit	Name	Description	R/W Flag	Reset Value
31:3	---	Reserved	R	0x0
2	AOS_CAL_BUSY	AOS Trigger Calibration Busy Flag: 0: No AOS calibration in progress; 1: AOS trigger calibration in progress.	R	0x0
1	CAL_OV	Calibrated Clock Counter Overflow Flag: 0: No overflow; 1: Overflow; Note: Write 1 to clear.	RW	0x0
0	CAL_DONE	Clock Calibration Complete Flag: 0: Not complete; 1: Complete; Note: Write 1 to clear.	RW	0x0

RC_CAL module actually only completes the clock period measurement of the calibrated clock. This module can select LOSC/HOSC/RCH/PLL as the reference clock (ref_clk), and can select RCL, RCH, RCH_VBAT as the calibrated clock (cal_clk). After selecting the clock sources, set the count value of the reference clock (REF_CLK_CNT[19:0]), then write TRIM_START.START to start clock calibration, subsequently query TRIM_START.CAL_DONE, if CAL_DONE=1, read the value of CAL_CLK_CNT[19:0] at this time, the period (frequency) of the calibrated clock can be calculated using the following formula:

$T_{CAL_CLK} = T_{REF_CLK} \times$（REF_CLK_CNT[19:0]＋1）÷（CAL_CLK_CNT[19:0]＋1）

After calculating the period of the calibrated clock, the system can perform calibration operations on the

calibrated clock.

The internal measurement process of the RC_CAL module is as follows: first assign the set value to REF_CLK_CNT, then perform down-counting in the REF_CLK clock domain until 0, generating a pulse width of length TREF_CLK × (REF_CLK_CNT[19:0] + 1). Then this pulse width is transmitted to the CAL_CLK clock domain. During the transmission between clock domains, the pulse width length may have an error, with a maximum error of ±(1 × TCAL_CLK). Then, in the CAL_CLK clock domain, perform up-counting starting from 0 during the valid period of this pulse width, generating CAL_CLK_CNT[19:0].

error=(Tcal_clk*1)/(Tref_clk*(ref_cnt+1)）

Taking the period measurement of RCH29M as an example, to achieve a measurement accuracy of 0.1%, what should the set value of REF_CNT be.

ref_clk(MHz)	LOSC	HOSC	HOSC/PLL	HOSC
	0.032768	29.4912	14.7456	7.3728
Tref_clk(ns)	30517.57813	33.90842014	67.81684028	135.6336806
REF_CNT[19:0]	1	1999	999	499
Total(ns)	61035.15625	67816.84028	67816.84028	67816.84028
Cal_clk(Mhz)	RCH29M	RCH29M	RCH29M	RCH29M
	29.4912	29.4912	29.4912	29.4912
Tcal_clk(ns)	33.90842014	33.90842014	33.90842014	33.90842014
CAL_CNT[19:0]	1799	1999	1999	1999
Error	0.11%	0.10%	0.10%	0.10%

4.6.4 Reset Class Registers

Register Function Class	Physical/Mapped Address	Function Class Address Offset
RST_TYPE	0x2008_0000	0x0300
Register Name	**Register Address Offset**	**Description**
RST_CTL	0x0300	Reset Control Register
RST_STOP_CTL	0x0304	Reset State Stop Control Register
SYS_RST_FLAG	0x0314	System Reset Flag Register
BOI_RST_CTL	0x0318	BOI_VBAT Reset Control Register
UPDATE_RST_CTL	0x031C	Update Reset Control Register
PMOD0_RST	0x0330	CORE0 Private Module Reset Register
MOD0_RST	0x0334	Assignable Module 0 Reset Register
MOD1_RST	0x0338	Assignable Module 1 Reset Register
MOD2_RST	0x033C	Assignable Module 2 Reset Register

4.6.4.1 RST_CTL（0x0300）

Reset Control Register

Bit	Name	Description	R/W Flag	Reset Value

31:16	RESERVED	Note: This register must maintain 0x785A	RW	0x785A
15:0	CORE0T1_RST_EN	CORE0 Reset CORE1 Enable: 0x82F3: CORE0 writes register to reset CORE1, CORE1 enters reset state; Other values: CORE1 operates normally;	RW	0x0

4.6.4.2 RST_STOP_CTL（0x0304）

Reset State Stop Control Register

Bit	Name	Description	R/W Flag	Reset Value
31:17	---	Reserved	R	0x0
15:0	CORE1_STOP	CORE1 STOP ON RESET Selection: 0x87A3: CORE1 STOP ON RESET (Logic 1); Other values: CORE1 normal startup (used during core1 update reset);	RW	0x87A3

4.6.4.3 SYS_RST_FLAG（0x0314）

System Reset Flag Register (Bits bit0~bit4 of this register can only be reset by power-on reset).

Bit	Name	Description	R/W Flag	Reset Value
31:7	---	Reserved	R	0x0
6	BOI_VBAT_RST_FLAG	VBAT BOI has experienced power-down reset: 0: This reset has not occurred; 1: This reset has occurred; Write 1 to clear Application note: If using this flag, upon first power-on, software should first clear this flag bit before application. When VBAT remains powered down, the flag cannot be cleared; software should clear the flag after detecting VBAT power-up again.	RW	0x0
5	---	Reserved	RW	0x0
4	CORE0_SOFT_RST_FLAG	CORE0 Software Reset Flag: 0: This reset has not occurred; 1: This reset has occurred; Write 1 to clear	RW	0x0
3	---	Reserved	RW	0x0
2	CORE0_WDT_RST_FLAG	CORE0 WDT Reset Flag: 0: This reset has not occurred; 1: This reset has occurred; Write 1 to clear	RW	0x0
1	PIN_RST_FLAG	External Pin Reset Flag:	RW	0x0

		0: This reset has not occurred; 1: This reset has occurred; Write 1 to clear		
0	POWEROK_RST_FLAG	Power Reset Flag: 0: This reset has not occurred; 1: This reset has occurred; Write 1 to clear	RW	0x1

4.6.4.4 BOI_RST_CTL（0x0318）

BOI_VBAT Reset Control Register

Bit	Name	Description	R/W Flag	Reset Value
31:7	---	Reserved	R	0x0
6	BOI_VBAT_RST_IE	BOI_VBAT_RST Interrupt Enable Bit: 0: Disable interrupt; 1: Enable interrupt; Application note: When enabling interrupt and after entering the interrupt, software should configure to 0 to disable the interrupt enable. After detecting VBAT has power again, clear the flag bit and re-enable the interrupt.	RW	0x0
5:0	---	Reserved	R	0x0

4.6.4.5 UPDATE_RST_CTL（0x031C）

Update Reset Control Register

Bit	Name	Description	R/W Flag	Reset Value
31:17	--	Reserved	R	0x0
16	CORE0_RST_FLAG	CORE0 Update Reset Flag =0 indicates has not occurred (this bit is 0 at power-on) =1 indicates has occurred this reset Write 1 to clear, except CORE0 update reset cannot clear this flag bit, other reset sources can reset this bit.	RW	0x0
15:0	CORE0_RST_EN	CORE0 Update Reset Enable (during update reset, prohibit dma access to flash) 0x35E2: Reset CORE0 core, hardware automatically clears after reset completion; Other values: Invalid;	RW	0x0

4.6.4.6 PMOD0_RST（0x0330）

CORE0 Private Module Reset Register

Note: For all module-level resets, first disable the module clock, then enable reset, then disable module reset, finally enable module clock for normal operation.

Bit	Name	Description	R/W Flag	Reset Value
31:7	--	Reserved	R	0x0
6	CRC1_RST_REQ	Software Reset CRC1 Module: Write 1 to reset CRC1 module; Write 0 to cancel CRC1 module reset.	RW	0x0
5	SYS_TC_RST_REQ	Software Reset SYS_TC Module: Write 1 to reset SYS_TC module; Write 0 to cancel SYS_TC module reset.	RW	0x0
4	GPIO_RST_REQ	Software Reset GPIO Module: Write 1 to reset GPIO module; Write 0 to cancel GPIO module reset.	RW	0x0
3	GDMA01_RST_REQ	Software Reset GDMA01 Module: Write 1 to reset GDMA01 module; Write 0 to cancel GDMA01 module reset.	RW	0x0
2	GDMA00_RST_REQ	Software Reset GDMA00 Module: Write 1 to reset GDMA00 module; Write 0 to cancel GDMA00 module reset.	RW	0x0
1	AOS_RST_REQ	Software Reset AOS Module: Write 1 to reset AOS module; Write 0 to cancel AOS module reset.	RW	0x0
0	MAILBOX_RST_REQ	Software Reset MAILBOX Module: Write 1 to reset MAILBOX module; Write 0 to cancel MAILBOX module reset.	RW	0x0

4.6.4.7 MOD0_RST（0x0334）

Assignable Module 0 Reset Register

Note: For all module-level resets, first disable the module clock, then enable reset, then disable module reset, finally enable module clock for normal operation.

Bit	Name	Description	R/W Flag	Reset Value
31:30	--	Reserved	R	0x0
29	TC1_RST_REQ	Software Reset TC1 Module: Write 1 to reset TC1 module; Write 0 to cancel TC1 module reset.	RW	0x0
28	TC0_RST_REQ	Software Reset TC0 Module: Write 1 to reset TC0 module; Write 0 to cancel TC0 module reset.	RW	0x0
27	DAC_RST_REQ	Software Reset DAC Module: Write 1 to reset DAC module; Write 0 to cancel DAC module reset.	RW	0x0
26	ADC0_RST_REQ	Software Reset ADC0 Module: Write 1 to reset ADC0 module;	RW	0x0

		Write 0 to cancel ADC0 module reset.		
25	D2F_RST_REQ	Software Reset D2F Module: Write 1 to reset D2F module; Write 0 to cancel D2F module reset.	RW	0x0
24	M2M_RST_REQ	Software Reset M2M Module: Write 1 to reset M2M module; Write 0 to cancel M2M module reset.	RW	0x0
23	--	Reserved	RW	0x0
22	DSP_RST_REQ	Software Reset DSP Module: Write 1 to reset DSP module; Write 0 to cancel DSP module reset.	RW	0x0
21	LCD_RST_REQ	Software Reset LCD Module: Write 1 to reset LCD module; Write 0 to cancel LCD module reset.	RW	0x0
20	IOCNT_RST_REQ	Software Reset IOCNT Module: Write 1 to reset IOCNT module; Write 0 to cancel IOCNT module reset.	RW	0x0
19	--	Reserved	RW	0x0
18	I2C1_RST_REQ	Software Reset I2C1 Module: Write 1 to reset I2C1 module; Write 0 to cancel I2C1 module reset.	RW	0x0
17	I2C0_RST_REQ	Software Reset I2C0 Module: Write 1 to reset I2C0 module; Write 0 to cancel I2C0 module reset.	RW	0x0
16	SPI3_RST_REQ	Software Reset SPI3 Module: Write 1 to reset SPI3 module; Write 0 to cancel SPI3 module reset.	RW	0x0
15	SPI2_RST_REQ	Software Reset SPI2 Module: Write 1 to reset SPI2 module; Write 0 to cancel SPI2 module reset.	RW	0x0
14	SPI1_RST_REQ	Software Reset SPI1 Module: Write 1 to reset SPI1 module; Write 0 to cancel SPI1 module reset.	RW	0x0
13	SPI0_RST_REQ	Software Reset SPI0 Module: Write 1 to reset SPI0 module; Write 0 to cancel SPI0 module reset.	RW	0x0
12	ISO78161_RST_REQ	Software Reset ISO78161 Module: Write 1 to reset ISO78161 module; Write 0 to cancel ISO78161 module reset.	RW	0x0
11	ISO78160_RST_REQ	Software Reset ISO78160 Module: Write 1 to reset ISO78160 module; Write 0 to cancel ISO78160 module reset.	RW	0x0
10	USART1_RST_REQ	Software Reset USART1 Module:	RW	0x0

		Write 1 to reset USART1 module; Write 0 to cancel USART1 module reset.		
9	USART0_RST_REQ	Software Reset USART0 Module: Write 1 to reset USART0 module; Write 0 to cancel USART0 module reset.	RW	0x0
8	LPUART_RST_REQ	Software Reset LPUART Module: Write 1 to reset LPUART module; Write 0 to cancel LPUART module reset.	RW	0x0
7	UART7_RST_REQ	Software Reset UART7 Module: Write 1 to reset UART7 module; Write 0 to cancel UART7 module reset.	RW	0x0
6	UART6_RST_REQ	Software Reset UART6 Module: Write 1 to reset UART6 module; Write 0 to cancel UART6 module reset.	RW	0x0
5	UART5_RST_REQ	Software Reset UART5 Module: Write 1 to reset UART5 module; Write 0 to cancel UART5 module reset.	RW	0x0
4	UART4_RST_REQ	Software Reset UART4 Module: Write 1 to reset UART4 module; Write 0 to cancel UART4 module reset.	RW	0x0
3	UART3_RST_REQ	Software Reset UART3 Module: Write 1 to reset UART3 module; Write 0 to cancel UART3 module reset.	RW	0x0
2	UART2_RST_REQ	Software Reset UART2 Module: Write 1 to reset UART2 module; Write 0 to cancel UART2 module reset.	RW	0x0
1	UART1_RST_REQ	Software Reset UART1 Module: Write 1 to reset UART1 module; Write 0 to cancel UART1 module reset.	RW	0x0
0	UART0_RST_REQ	Software Reset UART0 Module: Write 1 to reset UART0 module; Write 0 to cancel UART0 module reset.	RW	0x0

4.6.4.8 MOD1_RST（0x0338）

Assignable Module 1 Reset Register

Note: For all module-level resets, first disable the module clock, then enable reset, then disable module reset, finally enable module clock for normal operation

Bit	Name	Description	R/W Flag	Reset Value
31:11	--	Reserved	R	0x0
10	CRC0_RST_REQ	Software Reset CRC0 Module: Write 1 to reset CRC0 module;	RW	0x0

		Write 0 to cancel CRC0 module reset.		
9	LVDCMP_RST_REQ	Software Reset LVDCMP Module: Write 1 to reset LVDCMP module; Write 0 to cancel LVDCMP module reset.	RW	0x0
8	KBI1_RST_REQ	Software Reset KBI1 Module: Write 1 to reset KBI1 module; Write 0 to cancel KBI1 module reset.	RW	0x0
7	KBI0_RST_REQ	Software Reset KBI0 Module: Write 1 to reset KBI0 module; Write 0 to cancel KBI0 module reset.	RW	0x0
6	LSRX_RST_REQ	Software Reset LSRX Module: Write 1 to reset LSRX module; Write 0 to cancel LSRX module reset.	RW	0x0
5	--	Reserved	R	0x0
4	ANA_RST_REQ	Software Reset ANA Module (includes RTC & SAR): Write 1 to reset ANA module; Write 0 to cancel ANA module reset.	RW	0x0
3	SIMPTC3_RST_REQ	Software Reset SIMPTC3 Module: Write 1 to reset SIMPTC3 module; Write 0 to cancel SIMPTC3 module reset.	RW	0x0
2	SIMPTC2_RST_REQ	Software Reset SIMPTC2 Module: Write 1 to reset SIMPTC2 module; Write 0 to cancel SIMPTC2 module reset.	RW	0x0
1	SIMPTC1_RST_REQ	Software Reset SIMPTC1 Module: Write 1 to reset SIMPTC1 module; Write 0 to cancel SIMPTC1 module reset.	RW	0x0
0	SIMPTC0_RST_REQ	Software Reset SIMPTC0 Module: Write 1 to reset SIMPTC0 module; Write 0 to cancel SIMPTC0 module reset.	RW	0x0

4.6.4.9 MOD2_RST（0x033C）

Assignable Module 2 Reset Register

Note: For all module-level resets, first disable the module clock, then enable reset, then disable module reset, finally enable module clock for normal operation.

Bit	Name	Description	R/W Flag	Reset Value
31:9	--	Reserved	R	0x0
8	AES_RST_REQ	Software Reset AES Module: Write 1 to reset AES module; Write 0 to cancel AES module reset.	RW	0x0
7:3	--	Reserved	R	0x0
2	SEA_RST_REQ	Software Reset SEA Module:	RW	0x0

		Write 1 to reset SEA module; Write 0 to cancel SEA module reset.		
1	QSPI1_RST_REQ	Software Reset QSPI1 Module: Write 1 to reset QSPI1 module; Write 0 to cancel QSPI1 module reset.	RW	0x0
0	QSPI0_RST_REQ	Software Reset QSPI0 Module: Write 1 to reset QSPI0 module; Write 0 to cancel QSPI0 module reset.	RW	0x0

4.6.5 Analog Function Class Registers

Register Function Class	Physical/Mapped Address	Function Class Address Offset
A_FUN_TYPE	0x2008_0000	0x0400
Register Name	**Register Address Offset**	**Description**
BOR5_CTL	0x0400	BOR5 Control Register
PLVD5_CTL	0x0404	PLVD5 Control Register

4.6.5.1 BOR5_CTL（0x0400）

BOR5 Control Register

Bit	Name	Description	R/W Flag	Reset Value
31:16	---	Reserved	R	0x0
15:0	BOR5_SEL	BOR5 Threshold Selection Signal, Power-on threshold/Power-down threshold 0x75B6: 2.1v/1.8v Other values: 2.5v/2.2v, default	RW	0x0

4.6.5.2 PLVD5_CTL（0x0404）

PLVD5 Control Register

Bit	Name	Description	R/W Flag	Reset Value
31:2	---	Reserved	R	0x0
1	PLVD5_FLAG	PLVD5 has experienced low voltage event: 0: No flag indicating power voltage below 2.8v; 1: Flag indicating power voltage below 2.8v generated, write 1 to clear. Filtered by system clock for 16 cycles.	RW	0x0
0	PLVD5_IE	PLVD5 Interrupt Enable Bit: 0: Disable interrupt; 1: Enable interrupt; Application note: When enabling interrupt and after entering the interrupt, software should configure to 0 to disable the interrupt enable, then clear the flag bit. After detecting PLVD5 voltage is above 2.8v again, re-enable the interrupt.	RW	0x0

4.6.6Digital Function Class Registers

Register Function Class	Physical/Mapped Address	Function Class Address Offset
D_FUN_TYPE	0x2008_0000	0x0500
Register Name	Register Address Offset	Description
ADDR_MAP_CTL	0x0500	Address Mapping Control Register
FLASH_MAP_CTL	0x0504	FLASH Mapping Control Register
DMA_PRI0_CTL	0x0508	DMA Access Priority Control Register 0
DMA_PRI1_CTL	0x050C	DMA Access Priority Control Register 1
DMA_PRI2_CTL	0x0510	DMA Access Priority Control Register 2
IRFR_CTL	0x0514	Infrared Control Register

4.6.6.1 ADDR_MAP_CTL（0x0500）

Address Mapping Control Register

Note: The CORE0_REMAP register actually resets to 000 after reset occurs. After Bootrom runs, it will be changed to 001. The reset value seen by the user is 001.

Bit	Name	Description	R/W Flag	Reset Value
31:7	---	Reserved	R	0x0
6:4	CORE1_REMAP	Address Mapping: 000: CORE1 FLASH mapped at 0x8000_0000 address; CORE1 FLASH NVR mapped at 0x8010_0000 address; CORE1 SRAM mapped at 0x9000_0000 address; CORE1 QSPI XPM mapped at 0x8100_0000 address; 001: CORE1 FLASH mapped at 0x9000_0000 address; CORE1 FLASH NVR mapped at 0x9010_0000 address; CORE1 SRAM mapped at 0x8000_0000 address; CORE1 QSPI XPM mapped at 0x9100_0000 address; 010: CORE1 FLASH mapped at 0x8100_0000 address; CORE1 FLASH NVR mapped at 0x8110_0000 address; CORE1 SRAM mapped at 0x9000_0000 address; CORE1 QSPI XPM mapped at 0x8000_0000 address; Others: Equivalent to 000.	R	0x0
3	---	Reserved	R	0x0
2:0	CORE0_REMAP	Address Mapping: 000: CORE0 FLASH mapped at 0x0020_0000 address; CORE0 FLASH NVR mapped at 0x0030_0000 address; CORE0 SRAM mapped at 0x1000_0000 address; ROM mapped at 0x0000_0000 address; 001: CORE0 FLASH mapped at 0x0000_0000 address; CORE0 FLASH NVR mapped at 0x0010_0000 address;	RW	0x0

		CORE0 SRAM mapped at 0x1000_0000 address; ROM mapped at 0x0020_0000 address; 010: CORE0 FLASH mapped at 0x1020_0000 address; CORE0 FLASH NVR mapped at 0x1030_0000 address CORE0 SRAM mapped at 0x0000_0000 address; ROM mapped at 0x1000_0000 address; Others: Equivalent to 000.		

4.6.6.2 FLASH_MAP_CTL（0x0504）

FLASH Mapping Control Register

Bit	Name	Description	R/W Flag	Reset Value
31:2	---	Reserved	R	0x0
1	CORE1_FLASH_REMAP	Address Mapping: 0: CORE1 program start address mapped in CORE1 FLASH lower half; 1: CORE1 program start address mapped in CORE1 FLASH upper half;	R	0x0
0	CORE0_FLASH_REMAP	Address Mapping: 0: CORE0 program start address mapped in CORE0 FLASH lower half; 1: CORE0 program start address mapped in CORE0 FLASH upper half;	RW	0x0

4.6.6.3 DMA_PRI0_CTL（0x0508）

DMA Access SRAM Priority Configuration Register 0, acts on first-level arbitration

Bit	Name	Description	R/W Flag	Reset Value
31:30	DMA_CH15_PRI0	Channel 15: TCO_PWM Channel Access SRAM Priority Configuration Priority configuration same as DMA_CH0_PRI0 (0x0 lowest, 0x3 highest); When priority is the same, smaller channel number has higher priority.	RW	0x0
29:28	---	Reserved	RW	0x0
27:26	DMA_CH13_PRI0	Channel 13: UART7 Channel Access SRAM Priority Configuration Priority configuration same as DMA_CH0_PRI0 (0x0 lowest, 0x3 highest); When priority is the same, smaller channel number has higher priority.	RW	0x0

25:24	DMA_CH12_PRI0	Channel 12: UART6 Channel Access SRAM Priority Configuration Priority configuration same as DMA_CH0_PRI0 (0x0 lowest, 0x3 highest); When priority is the same, smaller channel number has higher priority.	RW	0x0
23:22	DMA_CH11_PRI0	Channel 11: UART5 Channel Access SRAM Priority Configuration Priority configuration same as DMA_CH0_PRI0 (0x0 lowest, 0x3 highest); When priority is the same, smaller channel number has higher priority.	RW	0x0
21:20	DMA_CH10_PRI0	Channel 10: UART4 Channel Access SRAM Priority Configuration Priority configuration same as DMA_CH0_PRI0 (0x0 lowest, 0x3 highest); When priority is the same, smaller channel number has higher priority.	RW	0x0
19:18	DMA_CH9_PRI0	Channel 9: UART3 Channel Access SRAM Priority Configuration Priority configuration same as DMA_CH0_PRI0 (0x0 lowest, 0x3 highest); When priority is the same, smaller channel number has higher priority.	RW	0x0
17:16	DMA_CH8_PRI0	Channel 8: UART2 Channel Access SRAM Priority Configuration Priority configuration same as DMA_CH0_PRI0 (0x0 lowest, 0x3 highest); When priority is the same, smaller channel number has higher priority.	RW	0x0
15:14	DMA_CH7_PRI0	Channel 7: UART1 Channel Access SRAM Priority Configuration Priority configuration same as DMA_CH0_PRI0 (0x0 lowest, 0x3 highest); When priority is the same, smaller channel number has higher priority.	RW	0x0
13:12	DMA_CH6_PRI0	Channel 6: UART0 Channel Access SRAM Priority Configuration Priority configuration same as DMA_CH0_PRI0 (0x0 lowest, 0x3 highest); When priority is the same, smaller channel number has higher priority.	RW	0x0
11:10	DMA_CH5_PRI0	Channel 5: DAC Channel Access SRAM Priority Configuration Priority configuration same as DMA_CH0_PRI0 (0x0 lowest, 0x3 highest); When priority is the same, smaller channel number has higher priority.	RW	0x0

9:8	DMA_CH4_PRI0	Channel 4: ADC0 Channel Access SRAM Priority Configuration Priority configuration same as DMA_CH0_PRI0 (0x0 lowest, 0x3 highest); When priority is the same, smaller channel number has higher priority.	RW	0x0
7:6	DMA_CH3_PRI0	Channel 3: SPI3 Channel Access SRAM Priority Configuration Priority configuration same as DMA_CH0_PRI0 (0x0 lowest, 0x3 highest); When priority is the same, smaller channel number has higher priority.	RW	0x0
5:4	DMA_CH2_PRI0	Channel 2: SPI2 Channel Access SRAM Priority Configuration Priority configuration same as DMA_CH0_PRI0 (0x0 lowest, 0x3 highest); When priority is the same, smaller channel number has higher priority.	RW	0x0
3:2	DMA_CH1_PRI0	Channel 1: SPI1 Channel Access SRAM Priority Configuration Priority configuration same as DMA_CH0_PRI0 (0x0 lowest, 0x3 highest); When priority is the same, smaller channel number has higher priority.	RW	0x0
1:0	DMA_CH0_PRI0	Channel 0: SPI0 Channel Access SRAM Priority Configuration. 0x0: Lowest priority 0x1: Second lowest priority 0x2: Second highest priority 0x3: Highest priority When priority is the same, smaller channel number has higher priority.	RW	0x0

4.6.6.4 DMA_PRI1_CTL（0x050C）

DMA Access SRAM Priority Configuration Register 1, acts on first-level arbitration

Bit	Name	Description	R/W Flag	Reset Value
31:6	--	Reserved	RO	0x0
5:4	DMA_CH18_PRI0	Channel 18: M2M Channel Access SRAM Priority Configuration. 0x0: Lowest priority 0x1: Second lowest priority 0x2: Second highest priority 0x3: Highest priority When priority is the same, smaller channel number has higher priority.	RW	0x0
3:2	DMA_CH17_PRI0	Channel 17: AES Channel Access SRAM Priority Configuration. 0x0: Lowest priority 0x1: Second lowest priority	RW	0x0

		0x2: Second highest priority 0x3: Highest priority When priority is the same, smaller channel number has higher priority.		
1:0	DMA_CH16_PRI0	Channel 16: TC1_PWM Channel Access SRAM Priority Configuration. 0x0: Lowest priority 0x1: Second lowest priority 0x2: Second highest priority 0x3: Highest priority When priority is the same, smaller channel number has higher priority.	RW	0x0

4.6.6.5 DMA_PRI2_CTL（0x0510）

DMA Access SRAM Priority Configuration Register 2, acts on second-level arbitration

Bit	Name	Description	R/W Flag	Reset Value
31:10	--	Reserved	RO	0x0
9:8	DMA_CH4_PRI1	Channel 4: CORE0/CORE1 Channel Access SRAM Priority Configuration. Priority configuration same as DMA_CH0_PRI1 (0x0 lowest, 0x3 highest); When priority is the same, smaller channel number has higher priority.	RW	0x0
7:6	DMA_CH3_PRI1	Channel 3: DSP Channel Access SRAM Priority Configuration. Priority configuration same as DMA_CH0_PRI1 (0x0 lowest, 0x3 highest); When priority is the same, smaller channel number has higher priority.	RW	0x0
5:4	DMA_CH2_PRI1	Channel 2: General DMA Write Channel (GDMA0x_WR/GDMA1x_WR) Access SRAM Priority Configuration. Priority configuration same as DMA_CH0_PRI1 (0x0 lowest, 0x3 highest); When priority is the same, smaller channel number has higher priority.	RW	0x0
3:2	DMA_CH1_PRI1	Channel 1: General DMA Read Channel (GDMA0x_RD/GDMA1x_RD) Access SRAM Priority Configuration. Priority configuration same as DMA_CH0_PRI1 (0x0 lowest, 0x3 highest); When priority is the same, smaller channel number has higher priority.	RW	0x0

1:0	DMA_CH0_PRI1	Channel 0: First-level Arbitration Result Channel Access SRAM Priority Configuration. 0x0: Lowest priority 0x1: Second lowest priority 0x2: Second highest priority 0x3: Highest priority When priority is the same, smaller channel number has higher priority.	RW	0x0

4.6.6.6 IRFR_CTL（0x0514）

Infrared Control Register

Bit	Name	Description	R/W Flag	Reset Value
31:8	---	Reserved	R	0x0
7:0	INFRARED_CYCLE	Infrared clock division factor based on 7.3728MHZ, in HCMM and RCMM modes. ... 0xBE: Infrared output clock 38.6K; 0xBF: Infrared output clock 38.4K; 0xC0: Infrared output clock 38.2K; ... Calculation formula: RCH58M/(8*INFRARED_CYCLE) Cannot be configured to 0.	RW	0xBF

4.6.7Chip Version Class Registers

Register Function Class	Physical/Mapped Address	Function Class Address Offset
CHIP_ID_TYPE	0x2008_0000	0x0800
Register Name	**Register Address Offset**	**Description**
CHIP_ID	0x0800	CHIP ID Register
DIE_VER	0x0804	Chip Version Register
FAB_UID0	0x0808	Chip Unique Code 0 Register
FAB_UID1	0x080C	Chip Unique Code 1 Register

4.6.7.1 CHIP_ID（0x0800）

CHIP ID Register

Bit	Name	Description	R/W Flag	Reset Value
15:0	CHIP_ID	CHIP_ID: C248	R	0xC248

4.6.7.2 DIE_VER（0x0804）

Chip Version Register

Bit	Name	Description	R/W Flag	Reset Value
31:4	Chip Model	Chip Model	R	0x6568FEC
3:0	Version Number	Version Number	R	0x0

4.6.7.3 FAB_UID0（0x0808）

Chip Unique Code 0 Register

Bit	Name	Description	R/W Flag	Reset Value
31:0	LOT_NUM	Wafer LOT NUM Information	R	0x0

4.6.7.4 FAB_UID1（0x080C）

Chip Unique Code 1 Register

Bit	Name	Description	R/W Flag	Reset Value
31:20	Y_LABEL	DIE Y Coordinate Information	R	0x0
19:8	X_LABEL	DIE X Coordinate Information	R	0x0
7:0	WAFER_ID	Wafer WAFER ID Information	R	0x0

4.7 Core1 System Control Registers

4.7.1 Password Class Registers

Register Function Class	Physical/Mapped Address	Function Class Address Offset
PUB_PSW_TYPE	0xA008_0000	0x0000
Register Name	Register Address Offset	Description
SYS_PSW1	0x0000	Public Password Register

4.7.1.1 SYS_PSW（0x0000）

Public Password Register

Bit	Name	Description	R/W Flag	Reset Value
31:8	---	Reserved	R	0
7:0	SYS_PSW1	When SYS_PSW1=8'h82, public registers are writable; When SYS_PSW1=other values, public registers are not writable; This register read value is the written value. It is recommended that the user disable write enable immediately after write operation completes.	RW	0x00

4.7.2 Clock Control Class Registers

Register Function Class	Physical/Mapped Address	Function Class Address Offset
CLK_CTL_TYPE	0xA008_0000	0x0100

Register Name	Register Address Offset	Description
SYS_MODE	0x0124	System Mode Register
PMOD0_CKE	0x012C	Private Module Clock Enable Register
KBI0_CKE	0x0134	KBI0 Clock Enable Register
KBI1_CKE	0x0138	KBI1 Clock Enable Register
MOD0_CKE	0x013C	Module Clock Enable Register 0
MOD1_CKE	0x0140	Module Clock Enable Register 1
MOD2_CKE	0x0144	Module Clock Enable Register 2
SYSFREQ_STA	0x0170	System Clock Frequency Indication Register

4.7.2.1 SYS_MODE（0x0124）

System Mode Register

Bit	Name	Description	R/W Flag	Reset Value
31:5	---	Reserved	R	0x0
4	FLASH_BUSY	Flash busy status, cannot enter mode switch: 0: idle 1: busy	R	0x0
3:0	SYS_MODE	Register read value: {0,HCM,RCM,LCM}	R	0x2

4.7.2.2 PMOD0_CKE（0x012C）

Private Module Clock Enable Register

Bit	Name	Description	R/W Flag	Reset Value
31:13	---	Reserved	R	0x0
12	HSEM_CKE	HSEM module enable, clock gating: 0: clock stopped 1: clock started	RW	0x0
11	GPIO_CKE	GPIO module enable, clock gating: 0: clock stopped 1: clock started	RW	0x0
10	UART38K_CKE	UART38K module clock enable: 0: clock stopped 1: clock started	RW	0x0
9	GDMA11_CKE	GDMA11 module clock enable: 0: clock stopped 1: clock started	RW	0x0
8	GDMA10_CKE	GDMA10 module clock enable: 0: clock stopped 1: clock started	RW	0x0
7:5	---	Reserved	R	0x0
4	CORE1_GIC_CKE	CORE1_GIC module clock enable:	RW	0x0

		0: clock stopped 1: clock started		
3	CORE1_SCU_CKE	CORE1_SCU module clock enable: 0: clock stopped 1: clock started	RW	0x0
2	CORE1_AOS_CKE	CORE1_AOS module enable, clock gating: 0: clock stopped 1: clock started	RW	0x0
1	CORE1_MAILBOX_CKE	CORE1_MAILBOX module clock enable: 0: clock stopped 1: clock started	RW	0x0
0	CORE1_WDT_CKE	CORE1_WDT apb clock enable, CORE1 enters sleep synchronously closes this clock: 0: clock stopped 1: clock started	RW	0x0

4.7.2.3 KBI0_CKE（0x0134）

KBI0 Clock Enable Register

Bit	Name	Description	R/W Flag	Reset Value
31	KBIC0_CKE	KBIC0 apb module clock enable: 0: clock stopped 1: clock started	RW	0x0
30:16	---	Reserved	R	0x0
15	KBI015_CKE	KBI015 module clock enable: 0: clock stopped 1: clock started	RW	0x0
14	KBI014_CKE	KBI014 module clock enable: 0: clock stopped 1: clock started	RW	0x0
13	KBI013_CKE	KBI013 module clock enable: 0: clock stopped 1: clock started	RW	0x0
12	KBI012_CKE	KBI012 module clock enable: 0: clock stopped 1: clock started	RW	0x0
11	KBI011_CKE	KBI011 module clock enable: 0: clock stopped 1: clock started	RW	0x0
10	KBI010_CKE	KBI010 module clock enable: 0: clock stopped 1: clock started	RW	0x0
9	KBI09_CKE	KBI09 module clock enable:	RW	0x0

		0: clock stopped 1: clock started		
8	KBI08_CKE	KBI08 module clock enable: 0: clock stopped 1: clock started	RW	0x0
7	KBI07_CKE	KBI07 module clock enable: 0: clock stopped 1: clock started	RW	0x0
6	KBI06_CKE	KBI06 module clock enable: 0: clock stopped 1: clock started	RW	0x0
5	KBI05_CKE	KBI05 module clock enable: 0: clock stopped 1: clock started	RW	0x0
4	KBI04_CKE	KBI04 module clock enable: 0: clock stopped 1: clock started	RW	0x0
3	KBI03_CKE	KBI03 module clock enable: 0: clock stopped 1: clock started	RW	0x0
2	KBI02_CKE	KBI02 module clock enable: 0: clock stopped 1: clock started	RW	0x0
1	KBI01_CKE	KBI01 module clock enable: 0: clock stopped 1: clock started	RW	0x0
0	KBI00_CKE	KBI00 module clock enable: 0: clock stopped 1: clock started	RW	0x0

4.7.2.4 KBI1_CKE（0x0138）

KBI1 Clock Enable Register

Bit	Name	Description	R/W Flag	Reset Value
31	KBIC1_CKE	KBIC1 apb module clock enable: 0: clock stopped 1: clock started	RW	0x0
30:16	---	Reserved	R	0x0
15	KBI115_CKE	KBI115 module clock enable: 0: clock stopped 1: clock started	RW	0x0
14	KBI114_CKE	KBI114 module clock enable: 0: clock stopped	RW	0x0

		1: clock started		
13	KBI113_CKE	KBI113 module clock enable: 0: clock stopped 1: clock started	RW	0x0
12	KBI112_CKE	KBI112 module clock enable: 0: clock stopped 1: clock started	RW	0x0
11	KBI111_CKE	KBI111 module clock enable: 0: clock stopped 1: clock started	RW	0x0
10	KBI110_CKE	KBI110 module clock enable: 0: clock stopped 1: clock started	RW	0x0
9	KBI19_CKE	KBI19 module clock enable: 0: clock stopped 1: clock started	RW	0x0
8	KBI18_CKE	KBI18 module clock enable: 0: clock stopped 1: clock started	RW	0x0
7	KBI17_CKE	KBI17 module clock enable: 0: clock stopped 1: clock started	RW	0x0
6	KBI16_CKE	KBI16 module clock enable: 0: clock stopped 1: clock started	RW	0x0
5	KBI15_CKE	KBI15 module clock enable: 0: clock stopped 1: clock started	RW	0x0
4	KBI14_CKE	KBI14 module clock enable: 0: clock stopped 1: clock started	RW	0x0
3	KBI13_CKE	KBI13 module clock enable: 0: clock stopped 1: clock started	RW	0x0
2	KBI12_CKE	KBI12 module clock enable: 0: clock stopped 1: clock started	RW	0x0
1	KBI11_CKE	KBI11 module clock enable: 0: clock stopped 1: clock started	RW	0x0
0	KBI10_CKE	KBI10 module clock enable: 0: clock stopped 1: clock started	RW	0x0

4.7.2.5 MOD0_CKE（0x013C）

Module Clock Enable Register 0

Bit	Name	Description	R/W Flag	Reset Value
31:30	---	Reserved	R	0x0
29	TC1_CKE	TC1 module clock enable: 0: clock stopped 1: clock started	RW	0x0
28	TC0_CKE	TC0 module clock enable: 0: clock stopped 1: clock started	RW	0x0
27	DAC_CKE	DAC module clock enable: 0: clock stopped 1: clock started	RW	0x0
26	ADC0_CKE	ADC0 module clock enable: 0: clock stopped 1: clock started	RW	0x0
25	D2F_CKE	D2F module clock enable: 0: clock stopped 1: clock started	RW	0x0
24	M2M_CKE	M2M module clock enable: 0: clock stopped 1: clock started	RW	0x0
23	---	Reserved	RW	0x0
22	DSP_CKE	DSP module clock enable: 0: clock stopped 1: clock started	RW	0x0
21	LCD_CKE	LCD module clock enable: 0: clock stopped 1: clock started	RW	0x0
20	IOCNT_CKE	IOCNT module clock enable: 0: clock stopped 1: clock started	RW	0x0
19	---	Reserved	RW	0x0
18	I2C1_CKE	I2C1 module clock enable: 0: clock stopped 1: clock started	RW	0x0
17	I2C0_CKE	I2C0 module clock enable: 0: clock stopped 1: clock started	RW	0x0
16	SPI3_CKE	SPI3 module clock enable: 0: clock stopped	RW	0x0

		1: clock started		
15	SPI2_CKE	SPI2 module clock enable: 0: clock stopped 1: clock started	RW	0x0
14	SPI1_CKE	SPI1 module clock enable: 0: clock stopped 1: clock started	RW	0x0
13	SPI0_CKE	SPI0 module clock enable: 0: clock stopped 1: clock started	RW	0x0
12	ISO78161_CKE	ISO7816_1 module clock enable: 0: clock stopped 1: clock started	RW	0x0
11	ISO78160_CKE	ISO7816_0 module clock enable: 0: clock stopped 1: clock started	RW	0x0
10	USART1_CKE	USART1 module clock enable: 0: clock stopped 1: clock started	RW	0x0
9	USART0_CKE	USART0 module clock enable: 0: clock stopped 1: clock started	RW	0x0
8	LPUART_CKE	LPUART module clock enable: 0: clock stopped 1: clock started	RW	0x0
7	UART7_CKE	UART5 module clock enable: 0: clock stopped 1: clock started	RW	0x0
6	UART6_CKE	UART5 module clock enable: 0: clock stopped 1: clock started	RW	0x0
5	UART5_CKE	UART5 module clock enable: 0: clock stopped 1: clock started	RW	0x0
4	UART4_CKE	UART4 module clock enable: 0: clock stopped 1: clock started	RW	0x0
3	UART3_CKE	UART3 module clock enable: 0: clock stopped 1: clock started	RW	0x0
2	UART2_CKE	UART2 module clock enable: 0: clock stopped 1: clock started	RW	0x0

1	UART1_CKE	UART1 module clock enable: 0: clock stopped 1: clock started	RW	0x0
0	UART0_CKE	UART0 module clock enable: 0: clock stopped 1: clock started	RW	0x0

4.7.2.6 MOD1_CKE（0x0140）

Module Clock Enable Register 1

Bit	Name	Description	R/W Flag	Reset Value
31:11	---	Reserved	R	0x0
10	CRC0_CKE	CRC0 module clock enable: 0: clock stopped 1: clock started	RW	0x0
9	LVDCMP_CKE	LVDCMP module clock enable: 0: clock stopped 1: clock started	RW	0x0
8:7	---	Reserved	R	0x0
6	LSRX_CKE	LSRX module clock enable: 0: clock stopped 1: clock started	RW	0x0
5	---	Reserved	R	0x0
4	RTC_SARADC_CKE	RTC and SARADC bus clock enable: 0: clock stopped 1: clock started	RW	0x0
3	SIMPTC3_CKE	SIMPTC3 module clock enable: 0: clock stopped 1: clock started	RW	0x0
2	SIMPTC2_CKE	SIMPTC2 module clock enable: 0: clock stopped 1: clock started	RW	0x0
1	SIMPTC1_CKE	SIMPTC1 module clock enable: 0: clock stopped 1: clock started	RW	0x0
0	SIMPTC0_CKE	SIMPTC0 module clock enable: 0: clock stopped 1: clock started	RW	0x0

4.7.2.7 MOD2_CKE（0x0144）

Module Clock Enable Register 2

Bit	Name	Description	R/W	Reset

			Flag	Value
31:8	---	Reserved	R	0x0
8	AES_CKE	AES module clock enable: 0: clock stopped 1: clock started	RW	0x0
7	KEY_STORE_CKE	KEY_STORE module clock enable: 0: clock stopped 1: clock started	RW	0x0
6	SEA_SHA_CKE	SEA_SHA module clock enable: 0: clock stopped 1: clock started	RW	0x0
5	SEA_TRNG_CKE	SEA_TRNG module clock enable: 0: clock stopped 1: clock started	RW	0x0
4	SEA_AES_CKE	SEA_AES module clock enable: 0: clock stopped 1: clock started	RW	0x0
3	SEA_PKE_CKE	SEA_PKE module clock enable: 0: clock stopped 1: clock started	RW	0x0
2	SEA_BUS_CKE	SEA_BUS module clock enable: 0: clock stopped 1: clock started	RW	0x0
1	QSPI1_CKE	QSPI1 module clock enable: 0: clock stopped, module cleared 1: clock started, module enabled	RW	0x0
0	QSPI0_CKE	QSPI0 module clock enable: 0: clock stopped, module cleared 1: clock started, module enabled	RW	0x0

4.7.2.8 SYSFREQ_STA（0x0170）

System Clock Frequency Indication Register

Bit	Name	Description	R/W Flag	Reset Value
31:11	---	Reserved	R	0x0
10:8	FLASH_CLK_FREQ	Flash system frequency indication: 000: Current system main clock is 32.768KHz; 001: Current system main clock is 1.8432MHz; 010: Current system main clock is 3.6864MHz; 011: Current system main clock is 7.3728MHz; 100: Current system main clock is 14.7456Mhz; 101: Current system main clock is 29.4912Mhz;	R	0x1

		110: Current system main clock is 58.9824Mhz; Others: No meaning.		
7	---	Reserved	R	0x0
6:4	CORE1_SYS_FREQ	CORE1 core system main clock frequency indication: 000: Current system main clock is 32.768KHz; 001: Current system main clock is 1.8432MHz; 010: Current system main clock is 3.6864MHz; 011: Current system main clock is 7.3728MHz; 100: Current system main clock is 14.7456Mhz; 101: Current system main clock is 29.4912Mhz; 110: Current system main clock is 58.9824Mhz; Others: No meaning.	R	0x1
3	---	Reserved	R	0x0
2:0	CORE0_SYS_FREQ	CORE0 core system main clock frequency indication: 000: Current system main clock is 32.768KHz; 001: Current system main clock is 1.8432MHz; 010: Current system main clock is 3.6864MHz; 011: Current system main clock is 7.3728MHz; 100: Current system main clock is 14.7456Mhz; 101: Current system main clock is 29.4912Mhz; Others: No meaning.	R	0x1

4.7.3 Reset Class Registers

Register Function Class	Physical/Mapped Address	Function Class Address Offset
RST_TYPE	0xA008_0000	0x0300
Register Name	**Register Address Offset**	**Description**
SYS_RST_FLAG	0x0314	System Reset Flag Register
BOI_RST_CTL	0x0318	BOI_VBAT Reset Control Register
UPDATE_RST_CTL	0x031C	Update Reset Control Register
PMOD0_RST	0x0330	CORE1 Private Module Reset Register
MOD0_RST	0x0334	Assignable Module 0 Reset Register
MOD1_RST	0x0338	Assignable Module 1 Reset Register
MOD2_RST	0x033C	Assignable Module 2 Reset Register

4.7.3.1 SYS_RST_FLAG（0x0314）

System Reset Flag Register

Bit	Name	Description	R/W Flag	Reset Value
31:10	---	Reserved	R	0x0
9	CORE0T1_SOFT_RST_FLAG	CORE0 software reset CORE1 flag (reset only affects core1): 0: This reset has not occurred;	RW	0x0

		1: This reset has occurred; Write 1 to clear		
8	CORE1_SOFT_RST_FLAG	CORE1 software reset flag (reset only affects core1): 0: This reset has not occurred; 1: This reset has occurred; Write 1 to clear	RW	0x0
7	CORE1_WDT_RST_FLAG	CORE1 WDT reset flag (reset only affects core1): 0: This reset has not occurred; 1: This reset has occurred; Write 1 to clear	RW	0x0
6	BOI_VBAT_RST_FLAG	VBAT BOI has experienced power-down reset: 0: This reset has not occurred; 1: This reset has occurred; Write 1 to clear Application note: If using this flag, upon first power-on, software should first clear this flag bit before application. When VBAT remains powered down, the flag cannot be cleared; software should clear the flag after detecting VBAT power-up again.	RW	0x0
5	---	Reserved	RW	0x0
4	CORE0_SOFT_RST_FLAG	CORE0 software reset flag: 0: This reset has not occurred; 1: This reset has occurred; Write 1 to clear	RW	0x0
3	---	Reserved	RW	0x0
2	CORE0_WDT_RST_FLAG	CORE0 WDT reset flag: 0: This reset has not occurred; 1: This reset has occurred; Write 1 to clear	RW	0x0
1	PIN_RST_FLAG	External pin reset flag: 0: This reset has not occurred; 1: This reset has occurred; Write 1 to clear	RW	0x0
0	POWEROK_RST_FLAG	Power reset flag: 0: This reset has not occurred; 1: This reset has occurred; Write 1 to clear	RW	0x1

4.7.3.2 BOI_RST_CTL（0x0318）

BOI_VBAT Reset Control Register

Bit	Name	Description	R/W Flag	Reset Value
31:5	---	Reserved	R	0x0

4	BOI_VBAT_RST_IE	BOI_VBAT_RST interrupt enable bit: 0: Disable interrupt; 1: Enable interrupt; Application note: When enabling interrupt and after entering the interrupt, software should configure to 0 to disable the interrupt enable. After detecting VBAT has power again, clear the flag bit and re-enable the interrupt.	RW	0x0
3:2	---	Reserved	R	0x0
1	CORE1_RST_IE	CORE1 reset interrupt enable bit (this register only has power-on/power-down reset): 0: Disable interrupt; 1: Enable interrupt.	RW	0x1
0	CORE1_RST_PENDING	CORE1 reset PENDING bit: =0 indicates has not occurred; =1 indicates has occurred this reset (including update reset). Write 1 to clear.	RW	0x1

4.7.3.3 UPDATE_RST_CTL（0x031C）

Update Reset Control Register

Bit	Name	Description	R/W Flag	Reset Value
31:17	--	Reserved	R	0x0
16	CORE1_RST_FLAG	CORE1 update reset flag =0 indicates has not occurred (this bit is 0 at power-on) =1 indicates has occurred this reset Write 1 to clear, except CORE1 update reset cannot clear this flag bit, other reset sources can reset this bit.	RW	0x0
15:0	CORE1_RST_EN	CORE1 update reset enable (during update reset, prohibit dma access to flash) 0x35E2: Reset CORE1 core, hardware automatically clears after reset completion; Other values: Invalid;	RW	0x0

4.7.3.4 PMOD0_RST（0x0330）

CORE1 Private Module Reset Register

Note: For all module-level resets, first disable the module clock, then enable reset, then disable module reset, finally enable module clock for normal operation.

Bit	Name	Description	R/W Flag	Reset Value
31:5	--	Reserved	R	0x0
4	HSEM_RST_REQ	Software reset HSEM module:	RW	0x0

		Write 1 to reset HSEM module; Write 0 to cancel HSEM module reset.		
3	GDMA11_RST_REQ	Software reset GDMA11 module: Write 1 to reset GDMA11 module; Write 0 to cancel GDMA11 module reset.	RW	0x0
2	GDMA10_RST_REQ	Software reset GDMA10 module: Write 1 to reset GDMA10 module; Write 0 to cancel GDMA10 module reset.	RW	0x0
1	AOS_RST_REQ	Software reset AOS module: Write 1 to reset AOS module; Write 0 to cancel AOS module reset.	RW	0x0
0	MAILBOX_RST_REQ	Software reset MAILBOX module: Write 1 to reset MAILBOX module; Write 0 to cancel MAILBOX module reset.	RW	0x0

4.7.3.5 MOD0_RST（0x0334）

Assignable Module 0 Reset Register

Note: For all module-level resets, first disable the module clock, then enable reset, then disable module reset, finally enable module clock for normal operation.

Bit	Name	Description	R/W Flag	Reset Value
31:30	--	Reserved	R	0x0
29	TC1_RST_REQ	Software reset TC1 module: Write 1 to reset TC1 module; Write 0 to cancel TC1 module reset.	RW	0x0
28	TC0_RST_REQ	Software reset TC0 module: Write 1 to reset TC0 module; Write 0 to cancel TC0 module reset.	RW	0x0
27	DAC_RST_REQ	Software reset DAC module: Write 1 to reset DAC module; Write 0 to cancel DAC module reset.	RW	0x0
26	ADC0_RST_REQ	Software reset ADC0 module: Write 1 to reset ADC0 module; Write 0 to cancel ADC0 module reset.	RW	0x0
25	D2F_RST_REQ	Software reset D2F module: Write 1 to reset D2F module; Write 0 to cancel D2F module reset.	RW	0x0
24	M2M_RST_REQ	Software reset M2M module: Write 1 to reset M2M module; Write 0 to cancel M2M module reset.	RW	0x0
23	--	Reserved	RW	0x0
22	DSP_RST_REQ	Software reset DSP module: Write 1 to reset DSP module;	RW	0x0

		Write 0 to cancel DSP module reset.		
21	LCD_RST_REQ	Software reset LCD module: Write 1 to reset LCD module; Write 0 to cancel LCD module reset.	RW	0x0
20	IOCNT_RST_REQ	Software reset IOCNT module: Write 1 to reset IOCNT module; Write 0 to cancel IOCNT module reset.	RW	0x0
19	---	Reserved	RW	0x0
18	I2C1_RST_REQ	Software reset I2C1 module: Write 1 to reset I2C1 module; Write 0 to cancel I2C1 module reset.	RW	0x0
17	I2C0_RST_REQ	Software reset I2C0 module: Write 1 to reset I2C0 module; Write 0 to cancel I2C0 module reset.	RW	0x0
16	SPI3_RST_REQ	Software reset SPI3 module: Write 1 to reset SPI3 module; Write 0 to cancel SPI3 module reset.	RW	0x0
15	SPI2_RST_REQ	Software reset SPI2 module: Write 1 to reset SPI2 module; Write 0 to cancel SPI2 module reset.	RW	0x0
14	SPI1_RST_REQ	Software reset SPI1 module: Write 1 to reset SPI1 module; Write 0 to cancel SPI1 module reset.	RW	0x0
13	SPI0_RST_REQ	Software reset SPI0 module: Write 1 to reset SPI0 module; Write 0 to cancel SPI0 module reset.	RW	0x0
12	ISO78161_RST_REQ	Software reset ISO78161 module: Write 1 to reset ISO78161 module; Write 0 to cancel ISO78161 module reset.	RW	0x0
11	ISO78160_RST_REQ	Software reset ISO78160 module: Write 1 to reset ISO78160 module; Write 0 to cancel ISO78160 module reset.	RW	0x0
10	USART1_RST_REQ	Software reset USART0 module: Write 1 to reset USART0 module; Write 0 to cancel USART0 module reset.	RW	0x0
9	USART0_RST_REQ	Software reset USART0 module: Write 1 to reset USART0 module; Write 0 to cancel USART0 module reset.	RW	0x0
8	LPUART_RST_REQ	Software reset LPUART module: Write 1 to reset LPUART module; Write 0 to cancel LPUART module reset.	RW	0x0
7	UART7_RST_REQ	Software reset UART7 module: Write 1 to reset UART7 module;	RW	0x0

		Write 0 to cancel UART7 module reset.		
6	UART6_RST_REQ	Software reset UART6 module: Write 1 to reset UART6 module; Write 0 to cancel UART6 module reset.	RW	0x0
5	UART5_RST_REQ	Software reset UART5 module: Write 1 to reset UART5 module; Write 0 to cancel UART5 module reset.	RW	0x0
4	UART4_RST_REQ	Software reset UART4 module: Write 1 to reset UART4 module; Write 0 to cancel UART4 module reset.	RW	0x0
3	UART3_RST_REQ	Software reset UART3 module: Write 1 to reset UART3 module; Write 0 to cancel UART3 module reset.	RW	0x0
2	UART2_RST_REQ	Software reset UART2 module: Write 1 to reset UART2 module; Write 0 to cancel UART2 module reset.	RW	0x0
1	UART1_RST_REQ	Software reset UART1 module: Write 1 to reset UART1 module; Write 0 to cancel UART1 module reset.	RW	0x0
0	UART0_RST_REQ	Software reset UART0 module: Write 1 to reset UART0 module; Write 0 to cancel UART0 module reset.	RW	0x0

4.7.3.6 MOD1_RST（0x0338）

Assignable Module 1 Reset Register

Note: For all module-level resets, first disable the module clock, then enable reset, then disable module reset, finally enable module clock for normal operation.

Bit	Name	Description	R/W Flag	Reset Value
31:11	--	Reserved	R	0x0
10	CRC0_RST_REQ	Software reset CRC0 module: Write 1 to reset CRC0 module; Write 0 to cancel CRC0 module reset.	RW	0x0
9	LVDCMP_RST_REQ	Software reset LVDCMP module: Write 1 to reset LVDCMP module; Write 0 to cancel LVDCMP module reset.	RW	0x0
8	KBI1_RST_REQ	Software reset KBI1 module: Write 1 to reset KBI1 module; Write 0 to cancel KBI1 module reset.	RW	0x0
7	KBI0_RST_REQ	Software reset KBI0 module: Write 1 to reset KBI0 module; Write 0 to cancel KBI0 module reset.	RW	0x0
6	LSRX_RST_REQ	Software reset LSRX module:	RW	0x0

		Write 1 to reset LSRX module; Write 0 to cancel LSRX module reset.		
5	--	Reserved	R	0x0
4	RTC_SARADC_RST_REQ	Software reset RTC_SARADC module: Write 1 to reset RTC_SARADC module; Write 0 to cancel RTC_SARADC module reset.	RW	0x0
3	SIMPTC3_RST_REQ	Software reset SIMPTC3 module: Write 1 to reset SIMPTC3 module; Write 0 to cancel SIMPTC3 module reset.	RW	0x0
2	SIMPTC2_RST_REQ	Software reset SIMPTC2 module: Write 1 to reset SIMPTC2 module; Write 0 to cancel SIMPTC2 module reset.	RW	0x0
1	SIMPTC1_RST_REQ	Software reset SIMPTC1 module: Write 1 to reset SIMPTC1 module; Write 0 to cancel SIMPTC1 module reset.	RW	0x0
0	SIMPTC0_RST_REQ	Software reset SIMPTC0 module: Write 1 to reset SIMPTC0 module; Write 0 to cancel SIMPTC0 module reset.	RW	0x0

4.7.3.7 MOD2_RST（0x033C）

Assignable Module 2 Reset Register

Note: For all module-level resets, first disable the module clock, then enable reset, then disable module reset, finally enable module clock for normal operation.

Bit	Name	Description	R/W Flag	Reset Value
31:8	--	Reserved	R	0x0
8	AES_RST_REQ	Software reset AES module: Write 1 to reset AES module; Write 0 to cancel AES module reset.	RW	0x0
7:3	--	Reserved	R	0x0
2	SEA_RST_REQ	Software reset SEA module: Write 1 to reset SEA module; Write 0 to cancel SEA module reset.	RW	0x0
1	QSPI1_RST_REQ	Software reset QSPI1 module: Write 1 to reset QSPI1 module; Write 0 to cancel QSPI1 module reset.	RW	0x0
0	QSPI0_RST_REQ	Software reset QSPI0 module: Write 1 to reset QSPI0 module; Write 0 to cancel QSPI0 module reset.	RW	0x0

4.7.4 Digital Function Class Registers

Register Function Class	Physical/Mapped Address	Function Class Address Offset

D_FUN_TYPE	0xA008_0000	0x0500
Register Name	Register Address Offset	Description
ADDR_MAP_CTL	0x0500	Address Mapping Control Register
FLASH_MAP_CTL	0x0504	FLASH Mapping Control Register
DMA_PRI0_CTL	0x0508	DMA Access Priority Control Register 0
DMA_PRI1_CTL	0x050C	DMA Access Priority Control Register 1
DMA_PRI2_CTL	0x0510	DMA Access Priority Control Register 2

4.7.4.1 ADDR_MAP_CTL（0x0500）

Address Mapping Control Register

Note: The CORE0_REMAP register actually resets to 000 after reset occurs. After Bootrom runs, it will be changed to 001. The reset value seen by the user is 001.

Bit	Name	Description	R/W Flag	Reset Value
31:7	---	Reserved	R	0x0
6:4	CORE1_REMAP	Address Mapping: 000: CORE1 FLASH mapped at 0x8000_0000 address; CORE1 FLASH NVR mapped at 0x8010_0000 address; CORE1 SRAM mapped at 0x9000_0000 address; CORE1 QSPI XPM mapped at 0x8100_0000 address; 001: CORE1 FLASH mapped at 0x9000_0000 address; CORE1 FLASH NVR mapped at 0x9010_0000 address; CORE1 SRAM mapped at 0x8000_0000 address; CORE1 QSPI XPM mapped at 0x9100_0000 address; 010: CORE1 FLASH mapped at 0x8100_0000 address; CORE1 FLASH NVR mapped at 0x8110_0000 address; CORE1 SRAM mapped at 0x9000_0000 address; CORE1 QSPI XPM mapped at 0x8000_0000 address; Others: Equivalent to 000.	RW	0x0
3	---	Reserved	R	0x0
2:0	CORE0_REMAP	Address Mapping: 000: CORE0 FLASH mapped at 0x0020_0000 address; CORE0 FLASH NVR mapped at 0x0030_0000 address; CORE0 SRAM mapped at 0x1000_0000 address; ROM mapped at 0x0000_0000 address; 001: CORE0 FLASH mapped at 0x0000_0000 address; CORE0 FLASH NVR mapped at 0x0010_0000 address; CORE0 SRAM mapped at 0x1000_0000 address; ROM mapped at 0x0020_0000 address; 010: CORE0 FLASH mapped at 0x0020_0000 address; CORE0 FLASH NVR mapped at 0x0030_0000 address CORE0 SRAM mapped at 0x0000_0000 address;	R	0x0

		ROM mapped at 0x1000_0000 address; Others: Equivalent to 000.		

4.7.4.2 FLASH_MAP_CTL（0x0504）

FLASH Mapping Control Register

Bit	Name	Description	R/W Flag	Reset Value
31:2	---	Reserved	R	0x0
1	CORE1_FLASH_REMAP	Address Mapping: 0: CORE1 program start address mapped in CORE1 FLASH lower half; 1: CORE1 program start address mapped in CORE1 FLASH upper half;	RW	0x0
0	CORE0_FLASH_REMAP	Address Mapping: 0: CORE0 program start address mapped in CORE0 FLASH lower half; 1: CORE0 program start address mapped in CORE0 FLASH upper half;	R	0x0

4.7.4.3 DMA_PRI0_CTL（0x0508）

DMA Access SRAM Priority Configuration Register 0, acts on first-level arbitration

Bit	Name	Description	R/W Flag	Reset Value
31:30	DMA_CH15_PRI0	Channel 15: TCO_PWM Channel Access SRAM Priority Configuration Priority configuration same as DMA_CH0_PRI0 (0x0 lowest, 0x3 highest); When priority is the same, smaller channel number has higher priority.	RW	0x0
29:28	---	Reserved	RW	0x0
27:26	DMA_CH13_PRI0	Channel 13: UART7 Channel Access SRAM Priority Configuration Priority configuration same as DMA_CH0_PRI0 (0x0 lowest, 0x3 highest); When priority is the same, smaller channel number has higher priority.	RW	0x0
25:24	DMA_CH12_PRI0	Channel 12: UART6 Channel Access SRAM Priority Configuration Priority configuration same as DMA_CH0_PRI0 (0x0 lowest, 0x3 highest); When priority is the same, smaller channel number has higher priority.	RW	0x0

23:22	DMA_CH11_PRI0	Channel 11: UART5 Channel Access SRAM Priority Configuration Priority configuration same as DMA_CH0_PRI0 (0x0 lowest, 0x3 highest); When priority is the same, smaller channel number has higher priority.	RW	0x0
21:20	DMA_CH10_PRI0	Channel 10: UART4 Channel Access SRAM Priority Configuration Priority configuration same as DMA_CH0_PRI0 (0x0 lowest, 0x3 highest); When priority is the same, smaller channel number has higher priority.	RW	0x0
19:18	DMA_CH9_PRI0	Channel 9: UART3 Channel Access SRAM Priority Configuration Priority configuration same as DMA_CH0_PRI0 (0x0 lowest, 0x3 highest); When priority is the same, smaller channel number has higher priority.	RW	0x0
17:16	DMA_CH8_PRI0	Channel 8: UART2 Channel Access SRAM Priority Configuration Priority configuration same as DMA_CH0_PRI0 (0x0 lowest, 0x3 highest); When priority is the same, smaller channel number has higher priority.	RW	0x0
15:14	DMA_CH7_PRI0	Channel 7: UART1 Channel Access SRAM Priority Configuration Priority configuration same as DMA_CH0_PRI0 (0x0 lowest, 0x3 highest); When priority is the same, smaller channel number has higher priority.	RW	0x0
13:12	DMA_CH6_PRI0	Channel 6: UART0 Channel Access SRAM Priority Configuration Priority configuration same as DMA_CH0_PRI0 (0x0 lowest, 0x3 highest); When priority is the same, smaller channel number has higher priority.	RW	0x0
11:10	DMA_CH5_PRI0	Channel 5: DAC Channel Access SRAM Priority Configuration Priority configuration same as DMA_CH0_PRI0 (0x0 lowest, 0x3 highest); When priority is the same, smaller channel number has higher priority.	RW	0x0
9:8	DMA_CH4_PRI0	Channel 4: ADC0 Channel Access SRAM Priority Configuration Priority configuration same as DMA_CH0_PRI0 (0x0 lowest, 0x3 highest); When priority is the same, smaller channel number has higher priority.	RW	0x0
7:6	DMA_CH3_PRI0	Channel 3: SPI3 Channel Access SRAM Priority Configuration	RW	0x0

		Priority configuration same as DMA_CH0_PRI0 (0x0 lowest, 0x3 highest); When priority is the same, smaller channel number has higher priority.		
5:4	DMA_CH2_PRI0	Channel 2: SPI2 Channel Access SRAM Priority Configuration Priority configuration same as DMA_CH0_PRI0 (0x0 lowest, 0x3 highest); When priority is the same, smaller channel number has higher priority.	RW	0x0
3:2	DMA_CH1_PRI0	Channel 1: SPI1 Channel Access SRAM Priority Configuration Priority configuration same as DMA_CH0_PRI0 (0x0 lowest, 0x3 highest); When priority is the same, smaller channel number has higher priority.	RW	0x0
1:0	DMA_CH0_PRI0	Channel 0: SPI0 Channel Access SRAM Priority Configuration. 0x0: Lowest priority 0x1: Second lowest priority 0x2: Second highest priority 0x3: Highest priority When priority is the same, smaller channel number has higher priority.	RW	0x0

4.7.4.4 DMA_PRI1_CTL（0x050C）

DMA Access SRAM Priority Configuration Register 1, acts on first-level arbitration

Bit	Name	Description	R/W Flag	Reset Value
31:6	--	Reserved	RO	0x0
5:4	DMA_CH18_PRI0	Channel 18: M2M Channel Access SRAM Priority Configuration. 0x0: Lowest priority 0x1: Second lowest priority 0x2: Second highest priority 0x3: Highest priority When priority is the same, smaller channel number has higher priority.	RW	0x0
3:2	DMA_CH17_PRI0	Channel 17: AES Channel Access SRAM Priority Configuration. 0x0: Lowest priority 0x1: Second lowest priority 0x2: Second highest priority 0x3: Highest priority When priority is the same, smaller channel number has higher priority.	RW	0x0
1:0	DMA_CH16_PRI0	Channel 16: TC1_PWM Channel Access SRAM Priority Configuration.	RW	0x0

		0x0: Lowest priority 0x1: Second lowest priority 0x2: Second highest priority 0x3: Highest priority When priority is the same, smaller channel number has higher priority.		

4.7.4.5 DMA_PRI2_CTL（0x0510）

DMA Access SRAM Priority Configuration Register 2, acts on second-level arbitration

Bit	Name	Description	R/W Flag	Reset Value
31:10	--	Reserved	RO	0x0
9:8	DMA_CH4_PRI1	Channel 4: CORE0/CORE1 Channel Access SRAM Priority Configuration. Priority configuration same as DMA_CH0_PRI1 (0x0 lowest, 0x3 highest); When priority is the same, smaller channel number has higher priority.	RW	0x0
7:6	DMA_CH3_PRI1	Channel 3: DSP Channel Access SRAM Priority Configuration. Priority configuration same as DMA_CH0_PRI1 (0x0 lowest, 0x3 highest); When priority is the same, smaller channel number has higher priority.	RW	0x0
5:4	DMA_CH2_PRI1	Channel 2: General DMA Write Channel (MDMA0x_WR/MDMA1x_WR) Access SRAM Priority Configuration. Priority configuration same as DMA_CH0_PRI1 (0x0 lowest, 0x3 highest); When priority is the same, smaller channel number has higher priority.	RW	0x0
3:2	DMA_CH1_PRI1	Channel 1: General DMA Read Channel (MDMA0x_RD/MDMA1x_RD) Access SRAM Priority Configuration. Priority configuration same as DMA_CH0_PRI1 (0x0 lowest, 0x3 highest); When priority is the same, smaller channel number has higher priority.	RW	0x0
1:0	DMA_CH0_PRI1	Channel 0: First-level Arbitration Result Channel Access SRAM Priority Configuration. 0x0: Lowest priority 0x1: Second lowest priority 0x2: Second highest priority 0x3: Highest priority	RW	0x0

		When priority is the same, smaller channel number has higher priority.		

4.7.5Chip Version Class Registers

Register Function Class	Physical/Mapped Address	Function Class Address Offset
CHIP_ID_TYPE	0xA008_0000	0x0800
Register Name	**Register Address Offset**	**Description**
CHIP_ID	0x0800	CHIP ID Register
DIE_VER	0x0804	Chip Version Register
FAB_UID0	0x0808	Chip Unique Code 0 Register
FAB_UID1	0x080C	Chip Unique Code 1 Register

4.7.5.1 CHIP_ID（0x0800）

CHIP ID Register

Bit	Name	Description	R/W Flag	Reset Value
15:0	CHIP_ID	CHIP_ID: C248	R	0xC248

4.7.5.2 DIE_VER（0x0804）

Chip Version Register

Bit	Name	Description	R/W Flag	Reset Value
31:4	Chip Model	Chip Model	R	0x6568FEC
3:0	Version Number	Version Number	R	0x0

4.7.5.3 FAB_UID0（0x0808）

Chip Unique Code 0 Register

Bit	Name	Description	R/W Flag	Reset Value
31:0	LOT_NUM	Wafer LOT NUM	R	0x0

4.7.5.4 FAB_UID1（0x080C）

Chip Unique Code 1 Register

Bit	Name	Description	R/W Flag	Reset Value
31:20	Y_LABEL	DIE Y Coordinate	R	0x0
19:8	X_LABEL	DIE X Coordinate	R	0x0
7:0	WAFER_ID	Wafer WAFER ID	R	0x0

5 Kernel Modules

5.1 TIMER

5.1.1 Features

The Timer Unit (TIMER) is a kernel peripheral, and each core is equipped with one TIMER peripheral. The TIMER peripheral is typically used to generate timer interrupts, software interrupts, and perform software reset of the system.

The TIMER module is a memory-mapped module, through which the timer clock source can be configured and the timer counting can be paused.

5.1.2 Registers

TIMER Register Base Addresses

Module Name	Core0 Base Address	Core1 Base Address
TIMER	0xE003_0000	0xE003_0000

TIMER Register Offset Address

Register Name	Address Offset	Default Value	Attribute	Description
mtime_lo	0x00	0x0	RW	Reflects the lower 32-bit value of timer mtime
mtime_hi	0x04	0x0	RW	Reflects the higher 32-bit value of timer mtime
mtimecmp_lo	0x08	0xFFFFFFFF	RW	Configures the lower 32 bits of the timer compare value mtimecmp
mtimecmp_hi	0x0C	0xFFFFFFFF	RW	Configures the higher 32 bits of the timer compare value mtimecmp
msftrst	0xFF0	0x0	RW	Generates a chip software reset request
mtimectl	0xFF8	0x0	RW	Controls the timer counting
msip	0xFFC	0x0	RW	Generates a System Timer software interrupt

Note: The registers of the TIMER module only support word-aligned read and write access.

5.1.2.1 mtime（0x00）

Bit	Name	Description	R/W Flag	Reset Value
63:0	mtime	This register is a 64-bit count register, containing {mtime_hi, mtime_lo}. The timer is enabled by default, so it starts counting automatically after reset.	RW	0

5.1.2.2 mtimecmp（0x08）

After a timer interrupt is generated when mtime > mtimecmp, the timer interrupt can be cleared by software writing a larger value to the mtimecmp register or a smaller value to the mtime register.

Bit	Name	Description	R/W Flag	Reset Value

63:0	mtimecmp	This register is a 64-bit compare register, containing {mtimecmp_hi, mtimecmp_lo}. This register is used as the compare value for the timer. If mtime > mtimecmp, a timer interrupt is generated. The state of the mtime register is affected by the value of mtimectl.CMPCLREN: - If mtimectl.CMPCLREN = 1, then when mtime > mtimecmp, mtime is automatically cleared and starts counting again. - If mtimectl.CMPCLREN = 0, then when mtime > mtimecmp, mtime increments normally.	RW	0

5.1.2.3 msftrst（0xFF0）

Bit	Name	Description	R/W Flag	Reset Value
31	MSFTRST	Only the highest bit of this register is valid, and this valid bit can only be cleared by a chip reset. Write 0x80000a5f to this register to generate a chip software reset request.	RW	0
30:0	-	Reserved	R	0

5.1.2.4 mtimectl（0xFF8）

Bit	Name	Description	R/W Flag	Reset Value
31:3	-	Reserved	R	0
2	CLKSRC	Selects the TIMER clock frequency source. 0: Low-frequency clock. This chip supports two low-frequency clock sources, LOSC and RCL32K, **default is RCL32K** (system register CLK_SEL.CORE0_RCL_LOSC_SEL). 1: System clock.	RW	0
1	CMPCLREN	Configures whether the timer mtime count is cleared: 0: The mtime register increments normally. 1: After generating a timer interrupt, the mtime register is cleared and then continues counting.	RW	0
0	TIMESTOP	Controls the pause of the timer mtime count: 0: The mtime register increments normally. 1: The mtime register pauses counting.	RW	0

Note: In ECLIC mode, if mtimectl.CMPCLREN = 1 is set, the TIMER interrupt request can only maintain one pulse, so the TIMER interrupt needs to be set to edge-triggered mode.

5.1.2.5 msip（0xFFC）

The System Timer software interrupt needs to be actively triggered by configuring the msip register.

Bit	Name	Description	R/W Flag	Reset Value
31:1	-	Reserved	R	0
0	MSIP	Used to generate a System Timer software interrupt: 0: Clear the software interrupt. 1: Generate a software interrupt.	RW	0

5.1.3 Software Configuration Process

Configuration Process of System Timer Interrupt (Interrupt ID 7):

- Implement the eclic_mtip_handler timer interrupt handler function;
- Call the ECLIC_Register_IRQ function, which is used to configure the interrupt vector mode, interrupt trigger mode, interrupt level, interrupt priority, and then enable the interrupt;
- Call the SysTick_Config function to set the required number of Ticks;
- The interrupt is triggered when the mtime count meets the specified Ticks.

Configuration Process of System Timer Software Interrupt (Interrupt ID 3):

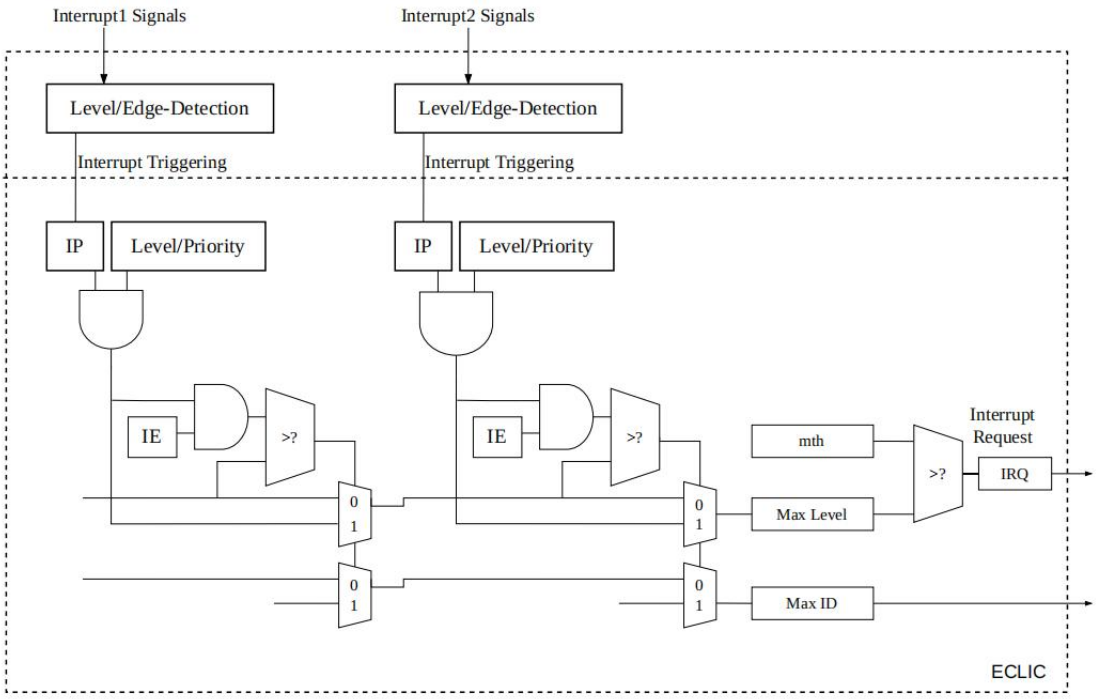
- Implement the eclic_msip_handler timer interrupt handler function;
- Call the ECLIC_Register_IRQ function, which is used to configure the software interrupt vector mode, interrupt trigger mode, interrupt level, interrupt priority, and then enable the interrupt;
- Configure msip.MSIP = 1 to trigger the interrupt.

5.2 ECLIC

5.2.1 Features

Each core integrates an ECLIC interrupt controller inside, which arbitrates external interrupt sources and core internal interrupt sources, and supports interrupt preemption.

The ECLIC module of this chip supports 147 interrupt sources, and defines the following characteristics and parameters for each interrupt source: ID number, enable bit IE, pending flag bit IP, level or edge-triggered attribute, level and priority, vector or non-vector processing.



Main Features:

- Interrupt IDs 0~18 are core internal interrupts, and currently only ID3 (Machine Software Interrupt) and ID7 (Machine Timer Interrupt) are supported. Note that mie/mip of CSR is meaningless in ECLIC mode;
- Interrupt IDs 19~146 are peripheral interrupts, which need to be assigned to the specified core through the GIC module except for inter-core interrupts and core internal interrupts;
- In ECLIC mode, mie/mip CSR is meaningless; global interrupt management is set through mstatus.MIE CSR, and each interrupt source is set by ECLIC.clicintie register;
- The ECLIC module registers are accessed in a memory-mapped manner;
- Each ECLIC interrupt supports the setting of level and edge interrupt trigger modes:
  - In level-triggered mode, software cannot clear the IP bit, and can only clear the IP bit by clearing the source of the interrupt;
  - In edge-triggered mode, the ECLIC automatically clears the IP bit after the core jumps to the interrupt service routine, without the need for software to clear it.
- Interrupt Level and Priority:
  - The Level and Priority of each interrupt are controlled by an 8-bit clicintctl register, whose actual valid bit width is the upper 3 bits, and the lower bits are all 1;
  - The bit width of Level is set globally by the nlbits field of the cliccfg register. When nlbits=0, all interrupt Levels=255; when nlbits>3, the actual level bit width is limited to 3;
  - Refer to the description of the clicintctl register for the Level and Priority values of each interrupt source.
- Supports the setting of global interrupt threshold mth; only interrupts with Level>mth will be processed;
- Interrupt preemption and arbitration:
  - Preemption: Interrupts with a higher Level can preempt interrupts with a lower Level;
  - Arbitration: Arbitration is performed in the order of Level, Priority, and ID, with larger values indicating higher priority.
- Each interrupt source can be set to vector mode or non-vector mode.

  Note: In this chip, the hardware hardwires clicinfo.CLICINTCTLBITS = 3, and the software configures

cliccfg.nlbits = 3. In this setting, the Level configuration is valid and the Priority configuration is invalid. The Priority configuration can be made valid by modifying the value of the macro __ECLIC_INTCTLBITS.

5.2.1.1 Level or Edge Attribute of Interrupt Sources

Each interrupt source of ECLIC can be set to level-triggered or edge-triggered attribute (configure clicintattr[i].trig), with the following key points:

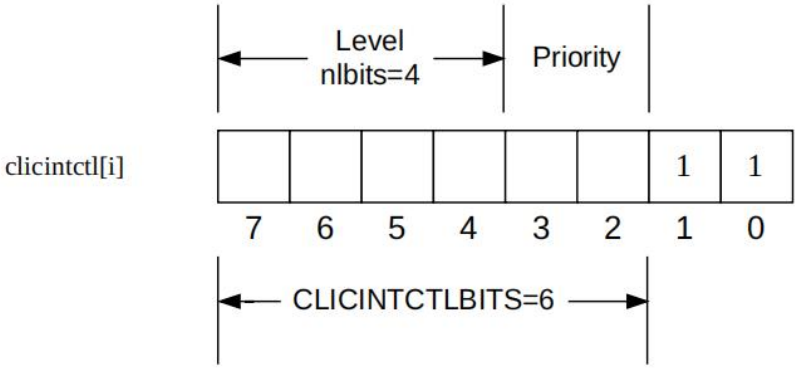
- When clicintattr[i].trig[0] = 0, the interrupt is set to level-triggered:
  - The IP bit of the interrupt source will reflect the level value of the interrupt source in real time;
  - Software cannot clear the IP bit by writing. If software needs to clear the interrupt, it can only do so by clearing the ultimate source of the interrupt.
- When clicintattr[i].trig[0] = 1 and clicintattr[i].trig[1] = 0, the interrupt is set to rising edge-triggered:
  - The interrupt source is triggered when the ECLIC detects a rising edge of the interrupt source, and the IP bit of the interrupt source is set high;
  - Software writing to the IP bit of the interrupt source is valid, that is, software can clear the IP bit by writing.
- When clicintattr[i].trig[0] = 1 and clicintattr[i].trig[1] = 1, the interrupt is set to falling edge-triggered:
  - The interrupt source is triggered when the ECLIC detects a falling edge of the interrupt source, and the IP bit of the interrupt source is set high;
  - Software writing to the IP bit of the interrupt source is valid, that is, software can clear the IP bit by writing.

For edge-triggered interrupts, to improve the efficiency of interrupt processing, when the interrupt is responded to and the core jumps to the Interrupt Service Routines (ISR), the ECLIC hardware will automatically clear the IP bit of the interrupt, eliminating the need for software to clear the IP bit inside the interrupt service routine.

5.2.1.2 Interrupt Level and Priority

5.2.1.2.1 Interrupt Level

The value of the interrupt Level is interpreted in a left-aligned manner. The lower bits outside the effective bit width (specified by cliccfg.nlbits) are all filled with constant 1, as shown in the figure below:



If cliccfg.nlbits > clicinfo.CLICINTCTLBITS, it means the number of bits indicated by nlbits exceeds the effective bits of the clicintctl[i] register. The exceeding bits are invalid and are all filled with constant 1.

If cliccfg.nlbits = 0, the Level value is considered fixed at 255.

The range of Level values corresponding to different nlbits values is shown in the figure below:

#nlbits	Encoding	The value of level
1	L..... (= L1111111)	127, 255
2	LL..... (= LL111111)	63, 127, 191, 255
3	LLL..... (= LLL11111)	31, 63, 95, 127, 159, 191, 223, 255
4	LLLL.... (= LLLL1111)	15, 31, 47, 63, 79, 95, 111, 127, 143, 159, 175, 191, 207, 223, 239, 255

"L" indicates the field of level
"." indicates the rest bits and are filled with the constant 1

Examples of cliccfg settings:

CLICINTCTLBITS	nlbits	clicintctl[i]	the value of level
0	2	.....	255
1	2	L.....	127, 255
2	2	LL.....	63, 127, 191, 255
3	3	LLL.....	31, 63, 95, 127, 159, 191, 223, 255
4	1	LPPP....	127, 255

Larger Level values indicate a higher level. Higher-level interrupts can preempt the processing of lower-level interrupts, thus forming interrupt nesting.

5.2.1.2.2 Interrupt Priority

The value of the interrupt Priority is also interpreted in a left-aligned manner. The lower bits outside the effective bit width (clicinfo.CLICINTCTLBITS - cliccfg nlbits) are all filled with constant 1.

Larger Priority values indicate a higher priority.

Interrupt Priority does not participate in the judgment of interrupt nesting. That is, whether an interrupt can nest is not related to the numerical size of the interrupt priority, but to the numerical size of the interrupt level. When multiple interrupts are pending (IP bit of interrupt source is high), the ECLIC needs to arbitrate to decide which interrupt is sent to the core for processing. During arbitration, the Level value, Priority value, and ID value of each interrupt source need to be referenced.

5.2.1.3 Interrupt Threshold

The software writes to the mth register to set the interrupt threshold. During the final arbitration process of the ECLIC, only if the interrupt Level value of an interrupt source is greater than the interrupt threshold set in the mth register, can that interrupt request be sent to the core.

This threshold represents a Level value.

5.2.1.4 Interrupt Arbitration

Only interrupt sources that meet all the following conditions can participate in arbitration: the enable bit (clicintie[i].IE) and the pending bit (clicintip[i].IP) of the interrupt source must both be

Arbitration rules are as follows:

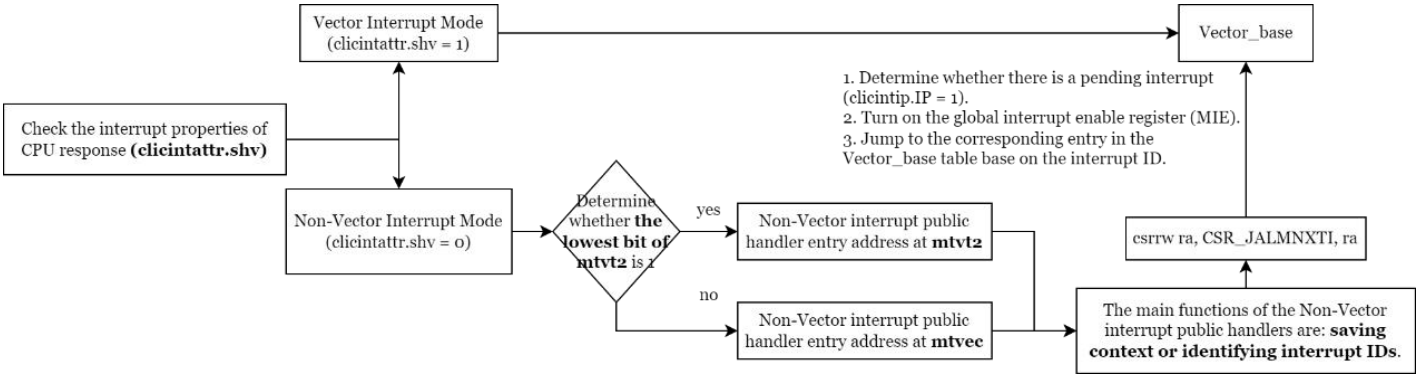
- First, compare the interrupt Level values; the larger the value, the higher the arbitration priority.
- If the Levels are equal, compare the interrupt Priority values; the larger the value, the higher the arbitration priority.
- If both Level and Priority values are equal, compare the interrupt ID numbers; the larger the value, the higher the arbitration priority.
- Finally, check if the Level value of the winning interrupt source in the arbitration is greater than the value in the mth register. If greater, the final interrupt request is generated and sent to the core.

5.2.1.5 Interrupt Preemption

When the core is processing a low-priority interrupt, if a new interrupt with a higher priority arrives, whose Level value is greater than the current interrupt's Level value, the hardware saves the current context and then jumps to the new interrupt's handler.

Vector mode does not support interrupt preemption. Therefore, if interrupt preemption or nesting is needed, the user must configure the interrupt source to non-vector mode.

5.2.1.6 Interrupt Vector Mode and Non-Vector Mode



The trigger flow for different modes is as follows:

- Interrupt configured as vector mode (clicintattr[i].shv = 1): When the interrupt is triggered, the kernel will read the interrupt vector table address stored in mtvt, then jump to the target interrupt handler address stored in the interrupt vector table.
- Interrupt configured as non-vector mode (clicintattr[i].shv = 0): When the interrupt is triggered, the kernel will first read the lowest bit of the mtvt2 register to see if it is 1:
  - mtvt2 lowest bit = 0: Jump via the non-vector interrupt common handler entry in mtvt2 to the interrupt vector table, i.e., mtvt2 specifies an independent entry address dedicated to non-vector interrupts, and mtvec specifies the exception entry address.
  - mtvt2 lowest bit = 1: Jump via the non-vector interrupt common handler entry in mtvec, where exceptions and non-vector interrupts share the entry address.

5.2.1.7 Vector Table Remapping

The vector table can be remapped by modifying the value of the mtvt register.

5.2.2 Registers

ECLIC Register Base Address

Module Name	Core0 Base Address	Core1 Base Address
ECLIC	0xE002_0000	0xE002_0000

ECLIC Register Offset Address

Register Name	Address Offset	Attribute	Description
cliccfg	0x00	RW	8-bit ECLIC global configuration register
clicinfo	0x04	R	32-bit ECLIC global information register
mth	0x0B	RW	8-bit ECLIC interrupt threshold configuration register
clicintip[i]	0x1000+4*i	RW	8-bit ECLIC interrupt source pending flag register
clicintie[i]	0x1001+4*i	RW	8-bit ECLIC interrupt source enable register
clicintattr[i]	0x1002+4*i	RW	8-bit ECLIC interrupt source attribute register
clicintctl[i]	0x1003+4*i	RW	8-bit ECLIC interrupt source control register

Note: i represents the interrupt id. Each interrupt has corresponding clicintip, clicintie, clicintattr, and clicintctl registers. The ECLIC address space size is 0xFFFF, and values at other addresses are always 0.

5.2.2.1 cliccfg（0x00）

Bit	Name	Description	R/W Flag	Reset Value
7:5	-	Reserved	R	0
4:1	nlbits	Used to specify the number of bits for the Level field in the clicintctl register.	RW	0
0	-	Reserved	R	1

5.2.2.2 clicinfo（0x04）

Bit	Name	Description	R/W Flag	Reset Value
31:25	-	Reserved	R	0
24:21	CLICINTCTLBITS	Used to specify the number of effective bits in the clicintctl register.	R	0x3
20:13	VERSION	Version number of the ECLIC hardware implementation.	R	0x1
12:0	NUM_INTERRUPT	Number of interrupt sources supported by the ECLIC hardware.	R	0x93

5.2.2.3 mth（0x0B）

Bit	Name	Description	R/W Flag	Reset Value

7:0	mth	Configures the threshold level for interrupt targeting.	RW	0

5.2.2.4 clicintip（0x1000+4*i）

Bit	Name	Description	R/W Flag	Reset Value
7:1	-	Reserved	R	0
0	IP	Pending flag for interrupt source i: 0: This interrupt is not triggered. 1: This interrupt has been triggered.	RW	0

5.2.2.5 clicintie（0x1001+4*i）

Bit	Name	Description	R/W Flag	Reset Value
7:1	-	Reserved	R	0
0	IE	Enable bit for interrupt source i: 0: Mask this interrupt source. 1: Enable this interrupt source.	RW	0

5.2.2.6 clicintattr（0x1002+4*i）

Bit	Name	Description	R/W Flag	Reset Value
7:6	-	Reserved	R	0x11
5:3	-	Reserved	R	0
2:1	trig	Used to specify the trigger condition for interrupt source i: 2'bx0: Level-triggered. 2'b01: Rising-edge triggered. 2'b11: Falling-edge triggered.	RW	0
0	shv	Used to specify the interrupt handling mode for interrupt source i: 0: Non-vector mode. 1: Vector mode.	RW	0

5.2.2.7 clicintctl（0x1003+4*i）

The Level and Priority fields of this register change dynamically based on the value of cliccfg.nlbits.

Bit	Name	Description	R/W Flag	Reset Value
7:(8 - nlbits)	Level	Configures the interrupt level for interrupt source i. A larger level indicates higher priority and supports interrupt preemption.	RW	0

(8 - nlbits - 1):5	Priority	Configures the interrupt priority for interrupt source i. A larger value indicates higher priority but does not support interrupt preemption.	RW	0
4:0	-	Reserved	R	0x1F

5.2.3 Software Configuration Flow

- Implement the interrupt handler entry in the interrupt vector table, note to add the keyword "__INTERRUPT".
- Enable the GIC module clock, allocate the specified interrupt to CORE0.
- Configure and enable the interrupt via the ECLIC_Register_IRQ function:

  int32_t ECLIC_Register_IRQ(IRQn_Type IRQn, uint8_t shv, ECLIC_TRIGGER_Type trig_mode, uint8_t lvl, uint8_t priority, void handler);*
  - IRQn: Specifies the ID of the interrupt source.
  - shv: Configures whether the interrupt source handling mode is vector mode or non-vector mode.
  - trig_mode: Configures the interrupt trigger mode for the interrupt source.
  - lvl: Configures the interrupt level for the interrupt source.
  - priority: Configures the interrupt priority for the interrupt source.
  - handler: Configures the interrupt handler entry for the interrupt source (pass NULL if not remapped).
- Alternatively, call the encapsulated kernel interrupt functions separately for configuration:

  /* set interrupt vector mode */

  ECLIC_SetShvIRQ(IRQn, shv);

  /* set interrupt trigger mode and polarity */

  ECLIC_SetTrigIRQ(IRQn, trig_mode);

  /* set interrupt level */

  ECLIC_SetLevelIRQ(IRQn, lvl);

  /* set interrupt priority */

  ECLIC_SetPriorityIRQ(IRQn, priority);

  /* enable interrupt */

  ECLIC_EnableIRQ(IRQn);

  Except for the ECLIC_EnableIRQ function which should be placed last, the order of the other functions does not matter.

5.3 PMP

5.3.1 Features

The PMP unit (Physical Memory Protection Unit) is used to provide security processing and fault isolation functions. By modifying the corresponding registers in M-mode, access permissions such as read, write, and execute for each physical memory region can be specified. Memory access operations that violate the PMP mechanism will be captured by the kernel and trigger an exception.

PMP features:

- PMP supports 8 entries. A PMP entry consists of a configuration register pmpcfg and an address register pmpaddr, which together define the access permissions for a memory region.
- PMP can support permission configuration for address ranges as small as 4 bytes.
- PMP allows configuration of coarser-grained protection regions, with a minimum granularity of 4KB.
- PMP registers can only be accessed in M-mode.

5.3.1.1 Address Matching Modes

The pmpaddr register only holds an address, and pmpcfg.A determines the address matching mode. Together they determine the physical address range controlled by the PMP entry. The specific modes are shown in the figure below:

A	Name	Description
0	OFF	Null region (disabled)
1	TOR	Top of range
2	NA4	Naturally aligned four-byte region
3	NAPOT	Naturally aligned power-of-two region, ≥8 bytes

5.3.1.1.1 OFF

This entry is disabled and does not match any address.

5.3.1.1.2 TOR

The address saved in pmpaddr~i~ of entry i forms the top of the address range, and the address saved in pmpaddr~i-1~ of the previous entry forms the bottom of the address range, i.e., the address region is represented as [$pmpaddr_{i-1}$, $pmpaddr_i$).

Specifically, **PMP entry 0 represents the address region [0, pmpaddr0).**

If $pmpaddr_{i-1} \geq pmpaddr_i$, then PMP entry i will not match any address, i.e., the region is empty.

5.3.1.1.3 NA4

The region is a fixed size of 4 bytes, generally used to contain a single register or word-sized data.

5.3.1.1.4 NAPOT

The region is a power-of-2 sized region, with a minimum region size of 8 bytes, requiring natural alignment, meaning the starting address of the region must be an integer multiple of its own size.

Under this rule, **the lower bits of the pmpaddr register indicate the address range, depending on the number of consecutive 1s at the lowest bits,** the higher bits represent the base address aligned to 4 bytes, separated by zeros in the middle. Assuming G is the number of consecutive 1s in the lower bits of the pmpaddr register, the actual size of the protected region in NAPOT mode is $2^{(G+3)}$ bytes. The pattern is shown in the table below:

pmpaddr in bits	pmpcfg.A	Physical Address Range Set by Entry
yyyy...yyyy	NA4	Region of size 2^2 bytes, base address is "yyyy...yyyy00".
yyyy...yyy0	NAPOT	Region of size 2^3 bytes, base address is "yyyy...yyy000".
yyyy...yy01	NAPOT	Region of size 2^4 bytes, base address is "yyyy...yy0000".
yyyy...y011	NAPOT	Region of size 2^5 bytes, base address is "yyyy...y00000".
...	...	...

yy01...1111	NAPOT	Region of size 2^{XLEN} bytes, base address is "0000...000000". The core address space of this chip is 32-bit, meaning under this configuration the entire 32-bit address space is fully covered.
y011...1111	NAPOT	Region of size 2^{XLEN+1} bytes, base address same as above.
0111...1111	NAPOT	Region of size 2^{XLEN+2} bytes, base address same as above.
1111...1111	NAPOT	Region of size 2^{XLEN+3} bytes, base address same as above.
Note: y represents 0 or 1.		

Example: Define a 4KB region starting from address 0x8000_0000.

- Calculate G: 4KB = 2^{12} bytes, from G + 3 = 12 we get G = 9.
- Construct the pmpaddr value:
  - High bits: 0x8000_0000 >> 2 = 0x2000_0000.
  - Low bits: G consecutive 1s, i.e., 0x1FF.
  - Combined result: 0x2000_0000 | 0x1FF = 0x2000_01FF.

Write to the pmpaddr register: The defined region is [0x2000_0000, 0x2000_01FF], size 4KB.

5.3.1.2 PMP Grain

The PMP grain defines the minimum size and basic alignment unit of the PMP protected address range.

The minimum grain is defined as 4KB, meaning any PMP protected region must be at least 4KB in size, and the starting address must be 4-byte aligned.

When in TOR mode, the address range between the value written to pmpaddr and the previous entry must not be less than 4KB. The value written to pmpaddr in NAPOT mode should also pay attention to the actual region size.

Note: Mode NA4 is not available in this chip, because NA4 aims to define a precise 4-byte protection region, but it is only available when the PMP grain is less than or equal to 4 bytes.

5.3.1.3 Entry Locking

Configure pmpcfg.L to control whether the PMP entry is locked. The locking mechanism is as follows:

- If pmpcfg.L = 1, it indicates that the entry is locked.
- When an entry is locked, the corresponding permission configuration register pmpcfg and address configuration register pmpaddr cannot be modified.
- When an entry is locked, it cannot be unlocked by software modifying pmpcfg.L; it can only be reset by a core reset.
- When an entry is not locked, the permissions and address range protection set by this entry are only effective for operations in S-mode/U-mode. When entry x is locked, it also applies to operations in M-mode.

5.3.2 Registers

The PMP unit is not a memory-mapped module.

5.3.2.1 pmpcfgi

Each $pmpcfg_i$ manages 4 PMP entries.

Bit	Name	Description	R/W Flag	Reset Value
31:24	$pmp_{x+3}cfg$	PMP configuration register for the x+3 entry.	RW	0

23:16	$pmp_{x+2}cfg$	PMP configuration register for the x+2 entry.	RW	0
15:8	$pmp_{x+1}cfg$	PMP configuration register for the x+2 entry.	RW	0
7:0	$pmp_{x}cfg$	PMP configuration register for the x entry.	RW	0

5.3.2.1.1 pmpicfg

When one of these bits is 0, the corresponding access type will be denied. For example:

- Attempting to fetch an instruction from a PMP region without execute permission causes an instruction access-fault exception.
- A load instruction accessing a physical address in a PMP region without read permission causes a load access-fault exception.
- Attempting to execute a store or AMO instruction that accesses a physical address in a PMP region without write permission causes a store access-fault exception.

Bit	Name	Description	R/W Flag	Reset Value
7	L	Configures whether the PMP entry is locked: 0: Do not lock the PMP entry. 1: Lock the PMP entry. The locked PMP entry remains locked until the core is reset. If PMP entry i is locked and pmp<i>cfg.A = TOR, then writes to pmpaddr<i-1> will also be ignored. Even if pmp<i>cfg.A = OFF, setting the L bit locks the PMP entry.	RW	0
6:5	-	Reserved	R	0
4:3	A	Configures the address matching mode: 2'b00: OFF, this entry is disabled, provides no protection. 2'b01: TOR, defines an arbitrary upper boundary. 2'b10: NA4, defines a 4-byte region. 2'b11: NAPOT, defines a naturally aligned region with a size that is a power of two and at least 8 bytes.	RW	0
2	X	Controls code execute permission: 0: Execution not permitted in the address range. 1: Execution permitted in the address range.	RW	0
1	W	Controls data write permission: 0: Write not permitted in the address range. 1: Write permitted in the address range.	RW	0
0	R	Controls data read permission: 0: Read not permitted in the address range. 1: Read permitted in the address range.	RW	0

5.3.2.2 pmpaddri

Bit	Name	Description	R/W Flag	Reset Value
31:0	address	PMP address for the i-th entry; the value written is the actual address shifted right by 2 bits.	RW	0

5.3.3 Software Configuration Flow

Related functions are in NMSIS/Core/Include/core_feature_pmp h. Usage example:

- Configure pmpaddr[i], calculate the pmpaddr value and write it.
  __set_PMPADDRx(1, (rv_csr_t)(&memory_pmp1[4]) >> 2);
- Configure pmp<i>cfg: Set the desired permissions RWX and write.
  __set_PMPxCFG(0, PMP_A_NA4 | PMP_R | PMP_W);
- If modification is not allowed, write 1 to pmpcfg.L.
  __set_PMPxCFG(1, PMP_A_NA4 | PMP_R | PMP_W);

6 LOSC Clock Security System (LCSS)

LOSC serves as an important internal clock source within the system, used for PLL multiplication source, RTC timekeeping, LPUART communication, LCD flicker, KBI filtering, infrared communication, and other functions. LCSS (LOSC Clock Security System) is a controller for clock monitoring and anomaly handling, detecting the LOSC clock frequency and performing corresponding clock switching processing based on a relatively reliable internal clock within the system. The LCSS clock security circuit includes two domains: VCC domain and VBAT domain. When both VCC and VBAT are powered, LOSC clock security monitoring primarily relies on LCSS_VCC; when only the VBAT domain is powered, it relies on LCSS_VBAT for LOSC clock security monitoring.

6.1 LCSS_VCC

6.1.1 Introduction

The Clock Security System LCSS_VCC (LOSC Clock Security System) is used to monitor the LOSC clock frequency in real-time, featuring anomaly reporting and automatic switching functions. When the LOSC frequency becomes abnormal, it generates an anomaly interrupt and can optionally automatically switch modules using LOSC as the clock source to a backup clock source via hardware; LCSS supports LOSC anomaly self-recovery function. If this function is enabled, when the LOSC frequency is detected to be normal, the relevant clock sources automatically switch back to LOSC or PLL. It also supports software-controlled clock switching, which can be performed within the interrupt service routine.

6.2 LCSS_VBAT

6.2.1 Introduction

The Clock Security System LCSS_VBAT (LOSC Clock Security System) is used to monitor in real-time whether the LOSC clock and RCL32K clock in the VBAT domain have stopped oscillating, generating corresponding LOSC and RCL32K clock status flags and stop oscillation flags. It features anomaly reporting and automatic switching of the RTC clock source function.

7 DMA

7.1 Introduction

The DMA controller is used for high-speed data transfer between peripherals and memory. Data is rapidly moved between peripherals and memory without CPU involvement, freeing up the CPU to handle other tasks.

Based on the requirements of different modules and the system architecture, the chip adopts a structure combining General-purpose DMA (GDMA) and Special-purpose DMA (SDMA). The entire DMA system includes 4 General-purpose DMA hosts, 20 Special-purpose DMA hosts, and one DMA arbiter. The system's 160KB SRAM is physically composed of 6 IP blocks, supporting up to 6 DMA channels accessing 6 different SRAM blocks simultaneously in parallel.

7.1.1 Definition of Proprietary Terms

GDMA : General Direct Memory Access

SDMA : Special Direct Memory Access

7.2 Features

- Utilizes a General-purpose DMA (GDMA) + Special-purpose DMA (SDMA) architecture;
- The system contains a total of 4 General-purpose DMA hosts, among which 2 are secure hosts for data transfer between secure peripherals and memory, and 2 are non-secure hosts for data transfer between non-secure peripherals and memory;
- Each General-purpose DMA host
  - The source or target address for peripheral channels is in fixed mode when interfacing with the peripheral end, and in increment mode when interfacing with the memory end;
  - Depending on different requirements, some peripheral channels can only access SRAM, while some peripheral channels can access Flash or SRAM;
- The system contains 20 Special-purpose DMA hosts, each fixedly owned by a specific proprietary peripheral, with security attributes following the peripheral's security configuration;
- Special-purpose DMA host
  - The source or target address is in fixed mode when interfacing with the peripheral end, and in increment mode when interfacing with the memory end;
  - Different Special-purpose DMA hosts have different data channels: some are Peripheral -> Memory, some are Memory -> Peripheral, and some support both channel types;
- DMA arbitration is divided into 2 levels,
  - Primary arbitration is performed among all channels of each General-purpose DMA host, outputting 2 sets of bus access interfaces, one for read and one for write;
  - For low-speed Special-purpose DMA hosts, primary arbitration is performed first, outputting one set of bus access interfaces, followed by timing interleaving;
  - Secondary arbitration is performed among the post-arbitration buses of General-purpose DMA, the post-arbitration bus of low-speed Special-purpose DMA, and the high-speed Special-purpose DMA hosts.

7.3 Functional Description

7.3.1 General-purpose DMA Functional Description

7.3.1.1 Host Channels

General-purpose DMA GDMAx_0 host contains 5 peripheral channels (P2M, M2P)

Host	Channel	Mode	Source Address	Destination Address
GDMA0_0 GDMA1_0	0	P2M	QSPI0 RX	SRAM
	1	P2M	USART0 RX	SRAM
	2	M2P	SRAM	QSPI0 TX
	3	M2P	SRAM	USART0 TX
	4	M2P/P2M	SRAM	SEA

General-purpose DMA GDMAx_1 host contains 5 channels, which are 4 peripheral channels (P2M, M2P) and 1 M2M (Memory to Memory) channel.

Host	Channel	Mode	Source Address	Destination Address
GDMA0_1 GDMA1_1	0	P2M	QSPI1 RX	SRAM
	1	P2M	USART1 RX	SRAM
	2	M2P	SRAM	QSPI1 TX
	3	M2P	SRAM	USART1 TX
	4	M2P	SRAM	SRAM
	5	M2M	SRAM	CRC0_DR

7.3.1.2 DMA Transfer Event

A single DMA event consists of a series of data transfers. The data width for transfer is configured by software.

Each DMA transfer includes the following steps:

- Load data from the source based on the value in the DMA_SxPAR or DMA_MxPAR register;
- Write the loaded data to the destination address based on the value in the DMA_SxPAR or DMA_MxPAR register;
- Decrement the value in the DMA_SxNDTR register.

7.3.1.3 Handshake Process Between Peripheral and DMA Controller

Each time a DMA event occurs, the peripheral sends a DMA request signal to the DMA controller; the DMA controller responds to the DMA request signals sequentially based on pre-configured priorities. Once the DMA controller completes the access for the corresponding peripheral channel, it sends an acknowledgment signal back to the peripheral. Upon receiving the DMA acknowledgment signal, the peripheral releases the DMA request signal, indicating the current transfer is complete. If the next DMA event occurs, the peripheral raises the DMA request

signal again.

Handshake of req and ack: When the enable for this peripheral channel is turned on, after req is raised, the DMA can start data transfer, first read then write (the DMA reads one piece of data from the data source via the AHB bus, then writes this data to the destination address). After completing one read-write operation (after the write feedback returns, not just after the write instruction is issued), the DMA's ack is pulled high. The ack signal causes the current req to be pulled low, completing one handshake. The peripheral can decide whether to raise the next req based on its own situation.

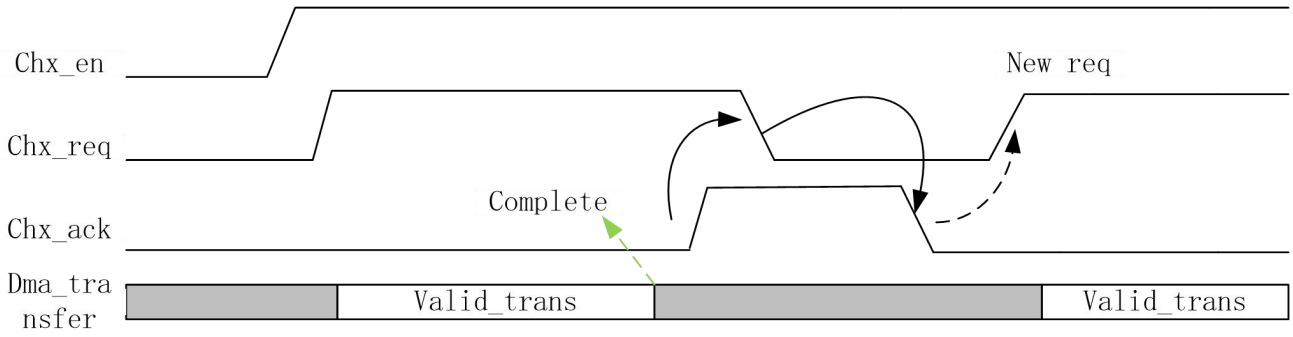


Figure 7-1 DMA Controller and Peripheral Handshake Process Schematic

7.3.1.4 Arbiter

DMA arbitration supports priorities consisting of two parts: configurable + fixed priority.

Configurable Priority: Each channel can be independently configured to one of the following 4 levels:

- very high
- high
- midium
- low

Fixed Priority: If two channels have the same configured priority, the channel with the smaller channel number has higher priority.

7.3.1.5 Transfer Direction

- peripheral-to-memory

When DIR is 0, the source address can be configured as a peripheral register address, and the destination address can be configured as an SRAM or Flash address (not supported by some channels, for area saving). Each time the peripheral initiates a DMA request, data is read from the source address and written to the destination address. Each time the transfer from source to destination is completed, the DMA_SxNDR register decrements accordingly. Once the DMA_SxNDR register reaches 0 or when the DMA enable bit is cleared by software, the DMA transfer ends immediately after the current AHB transaction completes.

- memory-to-peripheral

When DIR is 1, the source address can be configured as an SRAM or Flash address (not supported by some channels, for area saving), and the destination address can be configured as a peripheral register address. Each time the peripheral initiates a DMA request, data is read from the source address and written to the destination address. Each time the transfer from source to destination is completed, the DMA_SxNDR register decrements accordingly. Once the DMA_SxNDR register reaches 0 or when the DMA enable bit is cleared by software, the DMA transfer

ends immediately after the current AHB transaction completes.

- memory-to-memory

This mode is only supported by GDMA1. In this mode, both the source and destination addresses can be configured as SRAM, Flash, or register addresses.

When DMA works in this mode, it does not require a peripheral to trigger the DMA transfer; instead, it automatically starts the DMA transfer work after being enabled by software.

When configuring the EN bit of register DMA_SxCR to 1, the stream controller immediately fetches data from the memory source address (DMA_SxPAR) to fill the internal data register. When the amount of data in the FIFO reaches the threshold, the data in the data register is transferred to the memory destination address (DMA_SxM0AR). Repeat the above work until the specified length of data transfer is completed. Once the DMA_SxNDR register value reaches 0, or software configures EN to 0, the DMA transfer ends immediately after the current AHB transaction completes.

7.3.1.6 Circular Mode

Circular mode is used to handle circular buffers and continuous data streams. It is enabled by configuring the CIRC bit in the DMA_SxCR register.

The module includes shadow registers. When configured in circular mode, after each DMA transfer of the specified data length is completed, the data count configuration value DMA_SxNDR, DMA address pointers, etc., are automatically reloaded.

7.3.1.7 DMA Transfer Completion

Different events can trigger the end of DMA and set the TCIFx flag.

- In Normal mode, when the counter DMA_SxNDR equals 0;
- Software configures EN to 0, the stream controller is disabled, and the current transfer completes;

7.3.1.8 Address Pointer Increment

The address pointer increment or non-increment for each channel can be configured according to parameterized configuration.

7.3.1.9 Error Management

When a bus error occurs during the transfer process, the error flag TEIFx is set.

7.3.1.10 Peripheral Attributes

- GDMA0 Peripheral Attribute Table

Channel x	CH0	CH1	CH2	CH3	CH4
Peripheral	QSPI0_RX	USART0_RX	QSPI0_TX	USART0_TX	sea
Periph Addr Inc	No	No	No	No	No
MEM Addr Inc	Yes	Yes	Yes	Yes	Yes
Data Count Bit Width N	18	18	18	18	20

- GDMA1 Peripheral Attribute Table

Channel x	CH0	CH1	CH2	CH3	CH4	CH5
Peripheral	QSPI1_RX	USART1_RX	QSPI1_TX	USART1_TX	SRAM	CRC0_DR
Periph Addr Inc	No	No	No	No	No	Yes
MEM Addr Inc	Yes	Yes	Yes	Yes	Yes	Yes
Data Count Bit Width N	18	18	18	18	20	20

- GDMA1 M2M Channel High Attribute Table

M2M Channel i	CPU(i=0)	CRC0(i=1)
Source Addr Inc	No	Yes
MEM Addr Inc	Yes	Yes
Data Count Bit Width N	20	20

7.3.2 Special-purpose DMA Functional Description

Special-purpose DMA includes a total of 20 Special-purpose DMA hosts:

Host	Quantity
UART	8
SPI	4
TC_PWM	2
M2M	1
DSP	1
AES	1
DAC	1
ADC0	1
sum	20

7.3.3 SRAM DMA Arbitration Functional Description

A total of 24 hosts participate in SRAM access arbitration. The system has 6 SRAM blocks. To improve SRAM access bandwidth, all input channels are first decoded to output the currently accessed SRAM number, and then arbitration is performed for multiple access requests to each SRAM.

7.3.3.1 SRAM Security Attributes

As it needs to support configurable granularity of 16kB, and the addresses of allocated secure SRAM and non-secure SRAM are both contiguous, it is physically implemented using 4 pieces of 32kB SRAM + 2 pieces of 16kB SRAM, and SRAM address remapping is performed through different configurations. There are a total of 10 configurations and mapping relationships, as shown in the figure below:

SRAM security attributes are configured via the waterline address configuration WMSTAR[3:0] (WMSTAR can be configured from 0x1 to 0xA). WMSTAR[3:0]=n, indicates that n*16kB of SRAM is secure SRAM, and (10-n)*16kB of SRAM is non-secure SRAM.

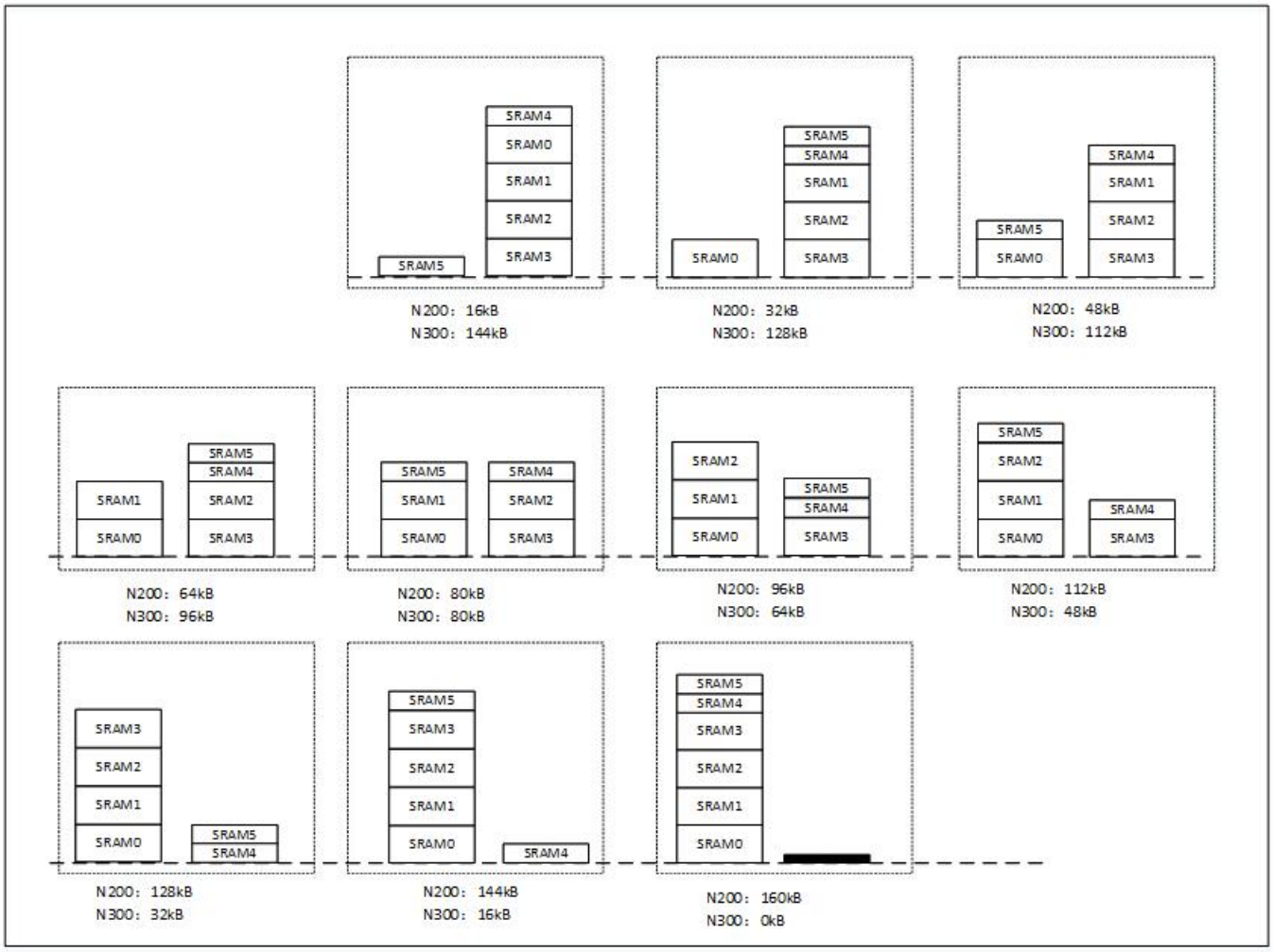


Figure 7-2 SRAM Security Attribute Configuration

As shown in the table above, for different configuration values, each SRAM has different mapping addresses, which can be divided into two categories: one is addresses below 0x8000_0000, and the other is addresses above 0x8000_0000. When mapped below 0x8000_0000, the 3 slave channels on the HSM bus can access this SRAM; when mapped above 0x8000_0000, the 3 slave channels on the HOST bus can access this SRAM. Furthermore, depending on the configuration, each SRAM only supports access by either the HSM or HOST bus at the same time, not simultaneous access.

7.3.3.2 Arbiter Structure

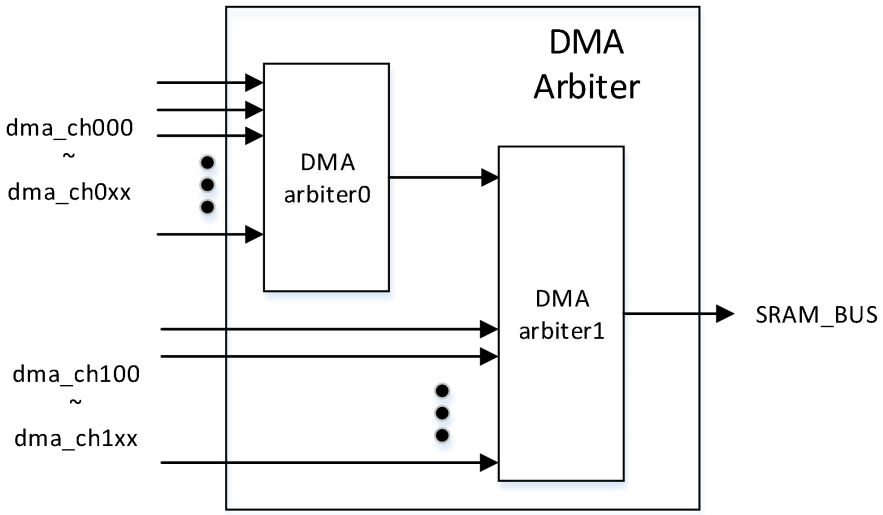


Figure 7-3 Arbiter Structure Block Diagram

To optimize SRAM interface timing, a two-level arbitration structure is adopted. Input channels for Arbiter0 require one more clock cycle to access SRAM compared to input channels for Arbiter1. The input channels are as follows:

Arbiter	Channel Number	Host
Arbiter0	0	SPI0
	1	SPI1
	2	SPI2
	3	SPI3
	4	ADC0
	5	DAC
	6	UART0
	7	UART1
	8	UART2
	9	UART3
	10	UART4
	11	UART5
	12	UART6
	13	UART7
	14	Reserved
	15	TC0_PWM
	16	TC1_PWM
	17	AES
	18	M2M
Arbiter1	0	Arbiter0_out
	1	GDMA0x_RD/GDMA1x_RD
	2	GDMA0x_WR/GDMA1x_WR

	3	DSP
	4	CORE0/CORE1

7.4 General-purpose DMA Registers

Base addresses for GDMA module registers

Module Name	Core0 Base Address	Core1 Base Address
GDMA0	0x2010_4000	0xA010_4000
GDMA1	0x2010_8000	0xA010_8000

Register offset addresses for GDMA0 interface

Register Name	Address Offset	Description
GDMA0_SxCTRL	0x000 + 0x20*x	Peripheral Channel x DMA Configuration
GDMA0_SxNDTR	0x004 + 0x20*x	Peripheral Channel x DMA Data Length Configuration
GDMA0_SxMADR	0x008 + 0x20*x	Peripheral Channel x DMA Memory End Base Address
GDMA0_SxPAR	0x00C + 0x20*x	Peripheral Channel x DMA Peripheral End Base Address
GDMA0_SIE	0x200	Peripheral Channel Interrupt Enable
GDMA0_SIF	0x204	Peripheral Channel Interrupt Flag

Where x ranges from 0 to 4 in GDMA0

Register offset addresses for GDMA1 interface

Register Name	Address Offset	Description
GDMA1_SxCTRL	0x000 + 0x20*x	Peripheral Channel x DMA Configuration
GDMA1_SxNDTR	0x004 + 0x20*x	Peripheral Channel x DMA Data Length Configuration
GDMA1_SxMADR	0x008 + 0x20*x	Peripheral Channel x DMA Memory End Base Address
GDMA1_SxPAR	0x00C + 0x20*x	Peripheral Channel x DMA Peripheral End Base Address
GDMA1_MiCTRL	0x100 + 0x20*i	M2M Channel i DMA Configuration, applicable to GDMA1
GDMA1_MiNDTR	0x104 + 0x20*i	M2M Channel i DMA Data Length Configuration, applicable to GDMA1
GDMA1_MiMADR	0x108 + 0x20*i	M2M Channel i DMA Memory End Base Address, applicable to GDMA1
GDMA1_MiPAR	0x10C + 0x20*i	M2M Channel i DMA Peripheral End Base Address, applicable to GDMA1
GDMA1_SIE	0x200	Peripheral Channel Interrupt Enable
GDMA1_SIF	0x204	Peripheral Channel Interrupt Flag
GDMA1_MIE	0x208	M2M Channel Interrupt Enable, applicable to

		GDMA1
GDMA1_MIF	0x210	M2M Channel Interrupt Flag, applicable to GDMA1

Where x ranges from 0 to 3 in GDMA1, and i ranges from 0 to 1 (i=0 is CPU, i=1 is CRC0).

7.4.1 GDMA0 Configuration Register GDMA0_SxCTRL (0x00 + 0x20*x)

Bit	Name	Description	R/W	Reset
31:7	---	Reserved	RO	0x0
6	DIR	Transfer Direction: 0: Data transfer from peripheral to memory; 1: Data transfer from memory to peripheral.	RW	0x0
5:4	MSIZE	Memory Data Size: 00: Byte 01: Half_word 10: Word 11: Reserved (Same as Byte) Configurable only when EN=0	RW	0x0
3:2	PL	Priority Level: 00: Lowest 01: Medium 10: High 11: Very High Configurable only when EN=0	RW	0x0
1	CIRC	Circular Mode: 0=Disable 1=Enable Configurable only when EN=0	RW	0x0
0	EN	DMA Channel Enable, configured by software: 0=Disable 1=Enable Hardware can also automatically clear this bit when the following occurs: - DMA transfer ends - DMA transfer error occurs - When source and destination address width configuration does not meet spec definition; - Transfer data amount is less than transfer data width.	RW	0x0

7.4.2 GDMA0 Transfer Data Length Configuration Register GDMA0_SxNDTR (0x04 + 20*x)

Bit	Name	Description	R/W	Reset
31:N	---	Reserved	RO	0x0
N-1:0	NDT	Number of data items to transfer (in Bytes).	RW	0x0

		Configurable only when EN is 0. When EN=1, this bit becomes read-only, indicating the number of data items yet to be transferred. Decrements with each transfer. Once the transfer is complete, this bit can remain 0, or in the following modes, it is automatically reloaded to the previously configured value; If the peripheral's DMA channel can access Flash, then N=20, otherwise N=18.		

7.4.3 GDMA0 Memory End Base Address Register GDMA0_SxMADR (0x08 + 20*x)

Bit	Name	Description	R/W	Reset
31:0	MADR	Memory base address (Byte), configurable when EN=0. Based on the current system, when this channel accesses flash, MADR[27:22] cannot be accessed, so MADR[27:22] configuration is invalid, read as 0; when this channel does not access flash, MADR[27:18] cannot be accessed, so MADR[27:18] configuration is invalid, read as 0;	RW	0x0

7.4.4 GDMA0 Peripheral End Base Address Register GDMA0_SxPADR (0x0C + 20*x)

Bit	Name	Description	R/W	Reset
31:0	PADR	Peripheral interface base address (Byte), configurable when EN=0. Based on the current system, PADR[27:22] cannot be accessed, so PADR[27:22] configuration is invalid, read as 0;	RW	0x0

7.4.5 GDMA0 Peripheral Interrupt Enable Configuration Register GDMA0 _SIE（0x200）

Bit	Name	Description	R/W	Reset Value
31:19	--	Reserved	RO	0x0
18	TEIE4	Peripheral Channel 4 Data Transfer Error Interrupt Enable 0=Disable 1=Enable	RW	0x0
17	TCIE4	Peripheral Channel 4 Data Transfer Complete Interrupt Enable 0=Disable 1=Enable	RW	0x0
16	HTIE4	Peripheral Channel 4 Data Transfer Half Interrupt Enable 0=Disable 1=Enable	RW	0x0

15	--	Reserved	RO	0x0
14	TEIE3	Peripheral Channel 3 Data Transfer Error Interrupt Enable 0=Disable 1=Enable	RW	0x0
13	TCIE3	Peripheral Channel 3 Data Transfer Complete Interrupt Enable 0=Disable 1=Enable	RW	0x0
12	HTIE3	Peripheral Channel 3 Data Transfer Half Interrupt Enable 0=Disable 1=Enable	RW	0x0
11	--	Reserved	RO	0x0
10	TEIE2	Peripheral Channel 2 Data Transfer Error Interrupt Enable 0=Disable 1=Enable	RW	0x0
9	TCIE2	Peripheral Channel 2 Data Transfer Complete Interrupt Enable 0=Disable 1=Enable	RW	0x0
8	HTIE2	Peripheral Channel 2 Data Transfer Half Interrupt Enable 0=Disable 1=Enable	RW	0x0
7	--	Reserved	RO	0x0
6	TEIE1	Peripheral Channel 1 Data Transfer Error Interrupt Enable 0=Disable 1=Enable	RW	0x0
5	TCIE1	Peripheral Channel 1 Data Transfer Complete Interrupt Enable 0=Disable 1=Enable	RW	0x0
4	HTIE1	Peripheral Channel 1 Data Transfer Half Interrupt Enable 0=Disable 1=Enable	RW	0x0
3	--	Reserved	RO	0x0
2	TEIE0	Peripheral Channel 0 Data Transfer Error Interrupt Enable 0=Disable 1=Enable	RW	0x0
1	TCIE0	Peripheral Channel 0 Data Transfer Complete Interrupt Enable 0=Disable 1=Enable	RW	0x0
0	HTIE0	Peripheral Channel 0 Data Transfer Half Interrupt Enable 0=Disable 1=Enable	RW	0x0

7.4.6 GDMA0 Peripheral Interrupt Flag Register GDMA0_SIF（0x204）

Bit	Name	Description	R/W	Reset Value
31:19	--	Reserved	RO	0x0
18	TEIF4	Peripheral Channel 4 Data Transfer Error Flag 0=No data transfer error 1=Data transfer error event occurred Software writes '1' to clear this bit	RW1C	0x0
17	TCIF4	Peripheral Channel 4 Data Transfer Complete Flag 0=Transfer not complete 1=Transfer complete event occurred Software writes '1' to clear this bit	RW1C	0x0
16	HTIF4	Peripheral Channel 4 Data Transfer Half Interrupt Flag 0=Transfer not half 1=Transfer half event occurred Software writes '1' to clear this bit	RW1C	0x0
15	--	Reserved	RO	0x0
14	TEIF3	Peripheral Channel 3 Data Transfer Error Flag 0=No data transfer error 1=Data transfer error event occurred Software writes '1' to clear this bit	RW1C	0x0
13	TCIF3	Peripheral Channel 3 Data Transfer Complete Flag 0=Transfer not complete 1=Transfer complete event occurred Software writes '1' to clear this bit	RW1C	0x0
12	HTIF3	Peripheral Channel 3 Data Transfer Half Interrupt Flag 0=Transfer not half 1=Transfer half event occurred Software writes '1' to clear this bit	RW1C	0x0
11	--	Reserved	RO	0x0
10	TEIF2	Peripheral Channel 2 Data Transfer Error Flag 0=No data transfer error 1=Data transfer error event occurred Software writes '1' to clear this bit	RW1C	0x0
9	TCIF2	Peripheral Channel 2 Data Transfer Complete Flag 0=Transfer not complete 1=Transfer complete event occurred Software writes '1' to clear this bit	RW1C	0x0
8	HTIF2	Peripheral Channel 2 Data Transfer Half Interrupt Flag 0=Transfer not half 1=Transfer half event occurred Software writes '1' to clear this bit	RW1C	0x0
7	--	Reserved	RO	0x0

6	TEIF1	Peripheral Channel 1 Data Transfer Error Flag 0=No data transfer error 1=Data transfer error event occurred Software writes '1' to clear this bit	RW1C	0x0
5	TCIF1	Peripheral Channel 1 Data Transfer Complete Flag 0=Transfer not complete 1=Transfer complete event occurred Software writes '1' to clear this bit	RW1C	0x0
4	HTIF1	Peripheral Channel 1 Data Transfer Half Interrupt Flag 0=Transfer not half 1=Transfer half event occurred Software writes '1' to clear this bit	RW1C	0x0
3	--	Reserved	RO	0x0
2	TEIF0	Peripheral Channel 0 Data Transfer Error Flag 0=No data transfer error 1=Data transfer error event occurred Software writes '1' to clear this bit	RW1C	0x0
1	TCIF0	Peripheral Channel 0 Data Transfer Complete Flag 0=Transfer not complete 1=Transfer complete event occurred Software writes '1' to clear this bit	RW1C	0x0
0	HTIF0	Peripheral Channel 0 Data Transfer Half Interrupt Flag 0=Transfer not half 1=Transfer half event occurred Software writes '1' to clear this bit	RW1C	0x0

7.5 General GDMA1 Registers

7.5.1 GDMA1 Configuration Register GDMA1_SxCTRL（0x00+0x20*x）

Bit	Name	Description	R/W	Reset Value
31:7	--	Reserved	RO	0x0
6	DIR	Transfer Direction: 0: Data transfer from peripheral to memory; 1: Data transfer from memory to peripheral.	RW	0x0
5:4	MSIZE	Transfer Data Size: 00: Byte 01: Half_word 10: Word 11: Reserved (Same as Byte) Configurable only when EN=0	RW	0x0
3:2	PL	Priority Level Configuration: 00: Lowest 01: Medium	RW	0x0

		10: High 11: Highest Configurable only when EN=0		
1	CIRC	Circular Mode: 0=Disable 1=Enable Configurable only when EN=0	RW	0x0
0	EN	DMA Channel Enable, configured by software: 0=Disable 1=Enable Hardware can also automatically clear this bit when the following occurs: - DMA transfer ends - DMA transfer error - When source and destination address bit width configuration does not meet specification; - Transfer data amount is less than transfer bit width	RW	0x0

7.5.2 GDMA1 Transfer Data Length Configuration Register GDMA1_SxNDTR（0x04+20*x）

Bit	Name	Description	R/W	Reset Value
31:18	--	Reserved	RO	0x0
17:0	NDT	Number of data items to transfer (in Bytes) Configurable only when EN is 0. When EN=1, this bit becomes read-only, indicating the number of data items remaining to be transferred. Decrements with each transfer. Once the transfer is complete, this bit remains 0, or in the following modes, automatically reloads to the previously configured value;	RW	0x0

7.5.3 GDMA1 Memory Base Address Register GDMA1_SxMADR（0x08+20*x）

Bit	Name	Description	R/W	Reset Value
31:0	MADR	Memory base address (Byte), configurable when EN=0. Based on the current system, when this channel accesses flash, MADR[27:22] is inaccessible, so MADR[27:22] configuration is invalid, reads as 0; when this channel does not access flash, MADR[27:18] is inaccessible, so MADR[27:18] configuration is invalid, reads as 0;	RW	0x0

7.5.4 GDMA1 Transfer Data Length Configuration Register GDMA1_SxPADR（0x0C+20*x）

Bit	Name	Description	R/W	Reset Value
31:0	PADR	Peripheral interface base address (Byte), configurable when EN=0. Based on the current system, PADR[27:22] is inaccessible, so PADR[27:22] configuration is invalid, reads as 0;	RW	0x0

7.5.5 GDMA M2M Channel i Configuration Register GDMA1_MiCTRL（0x100 +20*i）

Bit	Name	Description	R/W	Reset Value
31:8	--	Reserved	RO	0x0
7:6	SSIZE	Source area port data size: 00: Byte 01: Half_word 10: Word 11: Reserved (Same as Byte) Configurable only when EN=0	RW	0x0
5:4	DSIZE	Destination area port data size: 00: Byte 01: Half_word 10: Word 11: Reserved (Same as Byte) Configurable only when EN=0	RW	0x0
3:2	PL	Priority Level Configuration: 00: Lowest 01: Medium 10: High 11: Highest Configurable only when EN=0	RW	0x0
1	CIRC	Circular Mode: 0=Disable 1=Enable Configurable only when EN=0	RW	0x0
0	EN	DMA Channel Enable, configured by software: 0=Disable 1=Enable Hardware can also automatically clear this bit when the following occurs: - DMA transfer ends - DMA transfer error - When source and destination address bit width configuration	RW	0x0

		does not meet specification; - Configured transfer data amount is less than one transfer data amount		

7.5.6 GDMA M2M Channel i Transfer Data Length Configuration Register GDMA1_MiNDTR（0x104 +20*i）

Bit	Name	Description	R/W	Reset Value
31:20	--	Reserved	RO	0x0
19:0	NDT	Number of data items to transfer (in Bytes) Configurable only when EN is 0. When EN=1, this bit becomes read-only, indicating the number of data items remaining to be transferred. Decrements with each transfer. Once the transfer is complete, this bit remains 0, or in the following modes, automatically reloads to the previously configured value; - When configured in Circular Mode	RW	0x0

7.5.7 GDMA M2M Channel i Destination Address Register GDMA1_MiDADR（0x108 +20*i）

Bit	Name	Description	R/W	Reset Value
31:0	DADR	Destination base address (Byte), configurable when EN=0. Based on the current system, DADR[27:22] configuration is invalid, so DADR[27:22] reads as 0;	RW	0x0

7.5.8 GDMA M2M Channel i Source Address Register GDMA1_MiSADR（0x10C+20*i）

Offset Address=0x10C+20*i

Bit	Name	Description	R/W	Reset Value
31:0	SADR	Source base address (Byte), configurable when EN=0. Based on the current system, SADR[27:22] configuration is invalid, so SADR[27:22] reads as 0;	RW	0x0

7.5.9 GDMA1 Peripheral Interrupt Enable Configuration Register GDMA1 _SIE（0x200）

Bit	Name	Description	R/W	Reset Value
31:15	--	Reserved	RO	0x0
14	TEIE3	Peripheral Channel 3 Data Transfer Error Interrupt Enable 0=Disable 1=Enable	RW	0x0
13	TCIE3	Peripheral Channel 3 Data Transfer Complete Interrupt Enable 0=Disable	RW	0x0

		1=Enable		
12	HTIE3	Peripheral Channel 3 Data Transfer Half Interrupt Enable 0=Disable 1=Enable	RW	0x0
11	--	Reserved	RO	0x0
10	TEIE2	Peripheral Channel 2 Data Transfer Error Interrupt Enable 0=Disable 1=Enable	RW	0x0
9	TCIE2	Peripheral Channel 2 Data Transfer Complete Interrupt Enable 0=Disable 1=Enable	RW	0x0
8	HTIE2	Peripheral Channel 2 Data Transfer Half Interrupt Enable 0=Disable 1=Enable	RW	0x0
7	--	Reserved	RO	0x0
6	TEIE1	Peripheral Channel 1 Data Transfer Error Interrupt Enable 0=Disable 1=Enable	RW	0x0
5	TCIE1	Peripheral Channel 1 Data Transfer Complete Interrupt Enable 0=Disable 1=Enable	RW	0x0
4	HTIE1	Peripheral Channel 1 Data Transfer Half Interrupt Enable 0=Disable 1=Enable	RW	0x0
3	--	Reserved	RO	0x0
2	TEIE0	Peripheral Channel 0 Data Transfer Error Interrupt Enable 0=Disable 1=Enable	RW	0x0
1	TCIE0	Peripheral Channel 0 Data Transfer Complete Interrupt Enable 0=Disable 1=Enable	RW	0x0
0	HTIE0	Peripheral Channel 0 Data Transfer Half Interrupt Enable 0=Disable 1=Enable	RW	0x0

7.5.10 GDMA1 Peripheral Interrupt Flag Register GDMA1_SIF（0x204）

Bit	Name	Description	R/W	Reset Value
31:15	--	Reserved	RO	0x0
14	TEIF3	Peripheral Channel 3 Data Transfer Error Flag 0=No data transfer error 1=Data transfer error event occurred	RW1C	0x0

		Software writes '1' to clear this bit		
13	TCIF3	Peripheral Channel 3 Data Transfer Complete Flag 0=Transfer not complete 1=Transfer complete event occurred Software writes '1' to clear this bit	RW1C	0x0
12	HTIF3	Peripheral Channel 3 Data Transfer Half Interrupt Flag 0=Transfer not half 1=Transfer half event occurred Software writes '1' to clear this bit	RW1C	0x0
11	--	Reserved	RO	0x0
10	TEIF2	Peripheral Channel 2 Data Transfer Error Flag 0=No data transfer error 1=Data transfer error event occurred Software writes '1' to clear this bit	RW1C	0x0
9	TCIF2	Peripheral Channel 2 Data Transfer Complete Flag 0=Transfer not complete 1=Transfer complete event occurred Software writes '1' to clear this bit	RW1C	0x0
8	HTIF2	Peripheral Channel 2 Data Transfer Half Interrupt Flag 0=Transfer not half 1=Transfer half event occurred Software writes '1' to clear this bit	RW1C	0x0
7	--	Reserved	RO	0x0
6	TEIF1	Peripheral Channel 1 Data Transfer Error Flag 0=No data transfer error 1=Data transfer error event occurred Software writes '1' to clear this bit	RW1C	0x0
5	TCIF1	Peripheral Channel 1 Data Transfer Complete Flag 0=Transfer not complete 1=Transfer complete event occurred Software writes '1' to clear this bit	RW1C	0x0
4	HTIF1	Peripheral Channel 1 Data Transfer Half Interrupt Flag 0=Transfer not half 1=Transfer half event occurred Software writes '1' to clear this bit	RW1C	0x0
3	--	Reserved	RO	0x0
2	TEIF0	Peripheral Channel 0 Data Transfer Error Flag 0=No data transfer error 1=Data transfer error event occurred Software writes '1' to clear this bit	RW1C	0x0
1	TCIF0	Peripheral Channel 0 Data Transfer Complete Flag 0=Transfer not complete 1=Transfer complete event occurred	RW1C	0x0

		Software writes '1' to clear this bit		
0	HTIF0	Peripheral Channel 0 Data Transfer Half Interrupt Flag 0=Transfer not half 1=Transfer half event occurred Software writes '1' to clear this bit	RW1C	0x0

7.5.11 GDMA M2M Channel Interrupt Enable Configuration Register GDMA1 _MIE（0x208）

Bit	Name	Description	R/W	Reset Value
31:7	--	Reserved	RO	0x0
6	M2M_ERRIE1	M2M Channel 1 Data Transfer Error Interrupt Enable 0=Disable 1=Enable	RW	0x0
5	M2M_TCIE1	M2M Channel 1 Data Transfer Complete Interrupt Enable 0=Disable 1=Enable	RW	0x0
4	M2M_HTIE1	M2M Channel 1 Data Transfer Half Interrupt Enable 0=Disable 1=Enable	RW	0x0
3	--	Reserved	RO	0x0
2	M2M_ERRIE0	M2M Channel 0 Data Transfer Error Interrupt Enable 0=Disable 1=Enable	RW	0x0
1	M2M_TCIE0	M2M Channel 0 Data Transfer Complete Interrupt Enable 0=Disable 1=Enable	RW	0x0
0	M2M_HTIE0	M2M Channel 0 Data Transfer Half Interrupt Enable 0=Disable 1=Enable	RW	0x0

7.5.12 GDMA M2M Channel Interrupt Flag Register GDMA1 _MIF（0x20C）

Bit	Name	Description	R/W	Reset Value
31:7	--	Reserved	RO	0x0
6	M2M_ERRIF1	M2M Channel 1 Data Transfer Error Flag 0=Disable 1=Enable	RW	0x0
5	M2M_TCIF1	M2M Channel 1 Data Transfer Complete Flag 0=Transfer not complete 1=Transfer complete event occurred Software writes '1' to clear this bit	RW1C	0x0
4	M2M_HTIF1	M2M Channel 1 Data Transfer Half Interrupt Flag	RW1C	0x0

		0=Transfer not half 1=Transfer half event occurred Software writes '1' to clear this bit		
3	--	Reserved	RO	0x0
2	M2M_ERRIF0	M2M Channel 0 Data Transfer Error Flag 0=Disable 1=Enable	RW	0x0
1	M2M_TCIF0	M2M Channel 0 Data Transfer Complete Flag 0=Transfer not complete 1=Transfer complete event occurred Software writes '1' to clear this bit	RW1C	0x0
0	M2M_HTIF0	M2M Channel 0 Data Transfer Half Interrupt Flag 0=Transfer not half 1=Transfer half event occurred Software writes '1' to clear this bit	RW1C	0x0

7.6 Software Configuration Flow and Application Notes

Configuration flow is as follows:

- Configure GDMA_SxNDTR, GDMA_SxMADR, GDMA_SxPADR registers;
- Configure the interrupt enable register GDMA_SIE based on whether interrupts need to be generated;
- Configure the GDMA_SxCTRL register.

For channel x, when enable is turned on, multiple registers cannot be configured. Therefore, note the following during software configuration:

- Configure other registers first, then configure the GDMA_SxCTRL register, and enable the channel within GDMA_SxCTRL;
- The configured transfer width, base addresses, and transfer data length GDMA_SxNDTR should be byte-aligned; otherwise, the GDMA transfer will be abnormal.

8 MAILBOX

8.1 Introduction

The Mailbox is a communication channel between the CORE0 (Secure Core) kernel and the CORE1 (Application Core) kernel. The Mailbox has two interfaces, which are the AHB interfaces of CORE1_FAB and CORE0_FAB respectively. It communicates with CORE1 through CORE1_FAB and with CORE0 through CORE0_FAB. The CORE IDs of the CORE0 kernel and CORE1 kernel are different. The mailbox to be accessed is specified by the CPU ID and each mailbox status.。

8.2 Features

- Number of valid mailbox groups: 1
- Each mailbox group is divided into Receive/Transmit mailboxes; the size of a single mailbox is 256 bytes.

8.3 Block Diagram

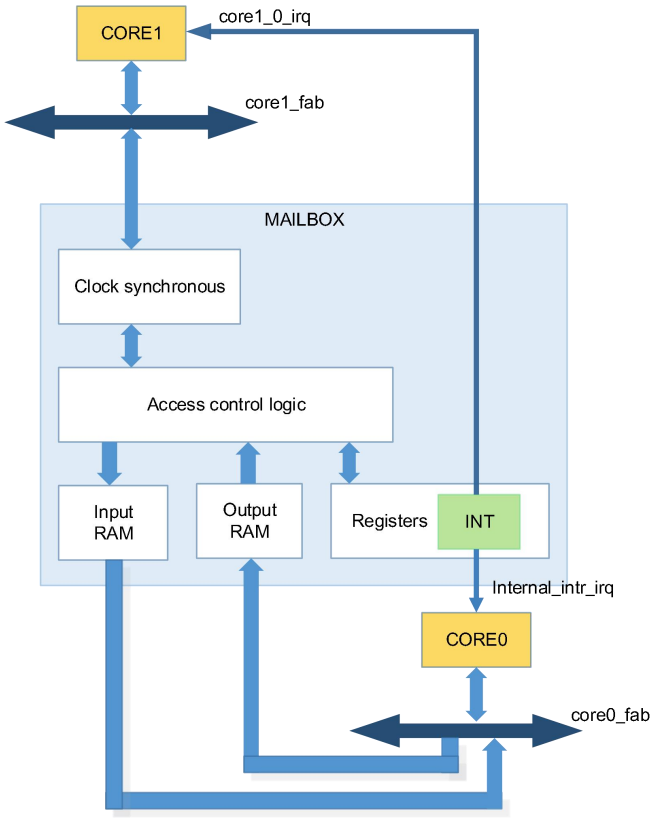


Figure 8-1 MAILBOX Module Block Diagram

Table 8-1 Function Description List

Name	Function Description
CORE0	Controls the MAILBOX channel for CORE0;
CORE1	CORE1 that can be connected to the MAILBOX channel;
Clock synchronous	Synchronizes the CORE1 bus to the CORE0 clock;

Access control logic	Controls the access of CORE0 and CORE1 to the mailbox SRAM and registers;
Input RAM	Input mailbox; single mailbox size is 256 bytes;
Output RAM	Output mailbox; single mailbox size is 256 bytes;
Registers	Register read/write and interrupt module;

8.4 Clock and Reset

The MAILBOX module has two clocks, coming from Core0 and Core1 respectively. The MAILBOX module has three reset signals: CORE0 bus reset, CORE1 bus reset, and module reset.

8.5 Functional Description

8.5.1 Connecting to Mailbox

When using the MAILBOX module for dual-core communication with CORE0, the host needs to be connected to the mailbox channel first. Currently, the only host that can connect to the mailbox channel in the system is CORE1.

CORE1 reads the CSR register. If CSR.MBX_ALB_UNLINK = 1, it indicates that CORE1 can connect to the mailbox channel and there is no pending data in the mailbox. If CSR.MBX_ALB_UNLINK = 0, it indicates that the mailbox is already connected to CORE1 or there is pending data in the mailbox.

If CSR.MBX_ALB_UNLINK = 1, CORE1 can connect to the mailbox channel by configuring the CSR.MBX_LINK bit. After a successful connection, the CSR.MBX_LINK bit is automatically set to 1.

8.5.2 Disconnecting from Mailbox

CORE1 writing 1 to the CSR.MBX_ALB_UNLINK bit can disconnect CORE1 from the mailbox. Writing to the CSR register to disconnect from the mailbox can simultaneously write 1 to the MBX_IN_FULL bit. At this time, CORE0 can read and process the data in the input mailbox, and CORE1 is disconnected from the mailbox.

CORE0 can disconnect CORE1 from the mailbox by writing to the RESET register MBX_UNLINK bit.

8.5.3 Memory Space

Both the input mailbox and output mailbox have a register set and a local SRAM. Each mailbox has a dedicated bus interface with an independent base address. The local SRAM is used to store input data, intermediate data, and results.

Input mailbox MAILBOX_IN offset address: 0x2000~0x20FF;

Output mailbox MAILBOX_OUT offset address: 0x2000~0x20FF;

For the input mailbox, only the connected CORE1 can write, and only CORE0 can read when the input mailbox is full; otherwise, read operations by CORE1 will read 0. CORE1 can only write when the input mailbox is not set to full by the CSR register. Illegal write operations are ignored.

For the output mailbox, CORE0 can only write when the output mailbox is not set to full by the CSR register; CORE1 can only read when the output mailbox is full; otherwise, read operations by CORE0 will read 0. Illegal write operations are ignored.

For CORE0, it can only write to the output mailbox when it is not set to full, and can only read from the input mailbox when it is set to full; reading the output mailbox will read 0.

For CORE1, it can only write to the input mailbox when it is not set to full, and can only read from the output mailbox when it is set to full; reading the input mailbox will read 0.

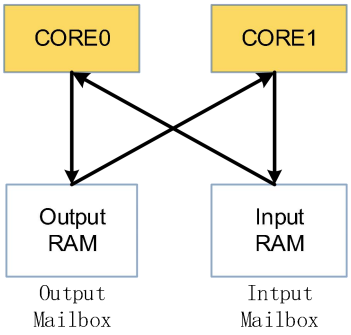


Figure 8-2 MAILBOX Access Diagram

8.6 Interrupts

Table 8-2 MAILBOX Interrupt List

Interrupt Event	Interrupt Target	Flag Bit	Enable Bit
Mailbox Linkable Interrupt	CORE1	INTST1.MBX_LINKABLE	INTEN1.MBX_LINKABLE
Output Mailbox Full Interrupt		INTST1.MBX_DONE	INTEN1.MBX_DONE
Mailbox Linked and Input Mailbox Empty Interrupt		INTST1.MBX_IFRE	INTEN1.MBX_IFRE
Output Mailbox Empty Interrupt	CORE0	INTST0.MBX_DONE	INTEN0.MBX_DONE
Input Mailbox Full Interrupt		INTST0.MBX_IFRE	INTEN0.MBX_IFRE

8.7 Registers

Table 8-3 MAILBOX Module Register Base Addresses

Module Name	CORE0 Address	CORE1 Address
MAILBOX	0x3000_C000	0xB000_C000

Table 8-4 MAILBOX Register Offset Addresses

Register Name	Offset Address	Default Value	Description	Read/Write Permissions	
				CORE0	CORE1
CSR	0x8	0x8	Mailbox Control and Status Register	RW	RW
RESET	0xC	0x0	Mailbox Reset Register	RW	RO
LINKID	0x10	0x0	Mailbox Connected Host ID Register	RO	RO
LOCKOUT	0x18	0x0	Mailbox Lock Register	RO	RW
INTEN0	0x38	0x0	Interrupt Enable Control Register for CORE0 access	RW	Inaccessible
INTEN1	0x38	0x0	Interrupt Enable Control Register for CORE1 access	Inaccessible	RW
INTST0	0x3C	0x0	Interrupt Status Register for CORE0 access	W1C	Inaccessible
INTST1	0x3C	0x0	Interrupt Status Register for CORE1	Inaccessible	W1C

			access		
OPT	0x5C	0x51	Mailbox Internal Configuration Status Register	RO	RO
FROZE	0x60	0x0	Freeze Host ID Access Configuration Register	RW	Inaccessible
ID_VAL	0x64	0x0	Accessible Host ID Number Register	RW	Inaccessible
ID_MASK	0xE4	0x0	Accessible Host ID Number Mask Register	RW	Inaccessible
MBX_KEY0	0x300	0x0	Mailbox Password Register 0	RW	Inaccessible
MBX_KEY1	0x304	0x0	Mailbox Password Register 1	RO	RW

8.7.1 CSR（0x08）

Mailbox Control/Status Register

Offset Address = 0x08

Bit	Name	Description	R/W	Reset Value
31:4	-	Reserved	-	0x0
3	MBX_ALB_UNLINK	This bit is readable and writable by CORE1, read-only by CORE0; 1. Write 1 to disconnect from the mailbox and clear the CSR.MBX_LINK bit. 2. Read back the linkable status of the mailbox: 0: Mailbox not linkable, indicating this mailbox is already connected to CORE1 or is in input mailbox full state; 1: Mailbox linkable;	RW	0x1
2	MBX_LINK	This bit is readable and writable by CORE1, read-only by CORE0; 1. Write 1 to connect to the mailbox channel. If successful, this bit is automatically set to 1. 2. Read back the linked status of the mailbox: 0: Mailbox not linked; 1: Mailbox linked;	RW	0x0
1	MBX_OUT_FULL	Output mailbox status bit, accessible by CORE0 and CORE1; For CORE0: 1. Write 1 indicates output mailbox full, INTST.MBX_DONE is set; 2. Read back output mailbox status: 0: Output mailbox empty; 1: Output mailbox full; For CORE1: 1. Write 1 clears output mailbox status; 2. Read back output mailbox status:	RW	0x0

		0: Output mailbox empty; 1: Output mailbox full;		
0	MBX_IN_FULL	Input mailbox status bit, accessible by CORE0 and CORE1; For CORE0: 1. Write 1 clears input mailbox status; 2. Read back input mailbox status: 0: Input mailbox empty; 1: Input mailbox full; For CORE1: 1. Write 1 indicates input mailbox full, INTST.MBX_IFRE is set; 2. Read back input mailbox status: 0: Input mailbox empty; 1: Input mailbox full;	RW	0x0

8.7.2 RESET（0x0C）

Mailbox Reset Register. This register is only accessible by CORE0.

Offset Address = 0x0C

Bit	Name	Description	R/W	Reset Value
31:4	-	Reserved	-	0x0
3	MBX_UNLINK	CORE0 writes 1 to this bit to clear the CSR.MBX_LINK bit and disconnect CORE1 from the mailbox. Reads back as 0.	W1C	0x0
2	-	Reserved	-	0x0
1	MBX_OUT_EMPTY	CORE0 writes 1 to this bit to clear CSR.MBX_OUT_FULL. Reads back as 0.	W1C	0x0
0	-	Reserved	-	0x0

8.7.3 LINKID（0x10）

Mailbox Connected Host ID Register

Offset Address = 0x10

Bit	Name	Description	R/W	Reset Value
31:1	-	Reserved	-	0x0
0	MBX_LINK_ID	Reads back the host ID number connected to the input mailbox. Since there is currently only one host in the system with host_id 0, it always reads back 0 and cannot be used to determine if CORE1 is connected to the mailbox.	RO	0x0

8.7.4LOCKOUT（0x18）

Mailbox Lock Register

Offset Address = 0x18

Bit	Name	Description	R/W	Reset Value
31:1	-	Reserved	-	0x0
0	MBX	This bit is only accessible by CORE1; Mailbox lock status, i.e., whether CORE1 can access the mailbox: 0: Mailbox unlocked, CORE1 can connect to the mailbox; 1: Mailbox locked, CORE1 cannot connect to the mailbox;	RW	0x0

8.7.5INTEN0（0x38）

CORE0 Mailbox Interrupt Enable Register. This register is only accessible by CORE0.

Offset Address = 0x38

Bit	Name	Description	R/W	Reset Value
31:2	-	Reserved	-	0x0
1	MBX_DONE	CORE0 Output Mailbox Empty Interrupt Enable; 0: Disable 1: Enable	RW	0x0
0	MBX_IFRE	CORE0 Input Mailbox Full Interrupt Enable; 0: Disable 1: Enable	RW	0x0

8.7.6INTEN1（0x38）

CORE1 Mailbox Interrupt Enable Register. This register is only accessible by CORE1.

Offset Address = 0x38

Bit	Name	Description	R/W	Reset Value
31:17	-	Reserved	-	0x0
16	MBX_LINKABLE	CORE1 Mailbox Linkable Interrupt Enable; 0: Disable 1: Enable	RW	0x0
15:2	-	Reserved	-	0x0
1	MBX_DONE	CORE1 Output Mailbox Full Interrupt Enable; 0: Disable 1: Enable	RW	0x0
0	MBX_IFRE	CORE1 Input Mailbox Empty Interrupt Enable; 0: Disable 1: Enable	RW	0x0

8.7.7INTST0（0x3C）

CORE0 Mailbox Interrupt Status Register. This register is only accessible by CORE0.

Offset Address = 0x3C

Bit	Name	Description	R/W	Reset Value
31:2	-	Reserved	-	0x0
1	MBX_DONE	Output Mailbox Empty Interrupt Flag 1. Set when CORE1 writes CSR.MBX_OUT_FULL, indicating CORE1 has finished reading data from the mailbox, output mailbox is empty; 2. Cleared by CORE0 writing 1;	W1C	0x0
0	MBX_IFRE	Input Mailbox Full Interrupt Flag 1. Set when CORE1 writes CSR.MBX_IN_FULL, indicating CORE1 has finished writing data to the input mailbox, input mailbox is full; 2. Cleared by CORE0 writing 1;	W1C	0x0

8.7.8INTST1（0x3C）

CORE1 Mailbox Interrupt Status Register. This register is only accessible by CORE1.

Offset Address = 0x3C

Bit	Name	Description	R/W	Reset Value
31:17	-	Reserved	-	0x0
16	MBX_LINKABLE	CORE1 Mailbox Linkable Interrupt Flag; 0: CORE1 cannot connect to mailbox 1: CORE1 can connect to mailbox	W1C	0x1
15:2	-	Reserved	-	0x0
1	MBX_DONE	Output Mailbox Full Interrupt Flag 1. Set when CORE0 writes CSR.MBX_OUT_FULL, indicating output mailbox is full; 2. Cleared by CORE1 writing 1 or when CORE1 disconnects from the mailbox;	W1C	0x0
0	MBX_IFRE	Input Mailbox Empty Interrupt Flag 1. Set when CORE0 writes CSR.MBX_IN_FULL, indicating input mailbox is empty; 2. Cleared by CORE1 writing 1 or when CORE1 disconnects from the mailbox;	W1C	0x0

8.7.9OPT（0x5C）

Mailbox Internal Configuration Status Register

Offset Address = 0x5C

Bit	Name	Description	R/W	Reset Value

31:8	-	Reserved	-	0x0
7:6	ACTIVE_HOST_N	Indicates the number of cores that can connect to the mailbox; Currently, only CORE1 can connect to the mailbox in the system, reads back 1;	RO	0x1
5:4	MBX_SIZE	Hardware specifies the actual implemented size of each input/output mailbox: 00: 128Bytes 01: 256Bytes (default) 10: 512Bytes 11: 1024Bytes	RO	0x1
3:0	MBX_NUM	Indicates the number of existing input/output mailbox pairs, range 1~8; Currently, there is only 1 mailbox group in the system, hardware fixed to 1;	RO	0x1

8.7.10 FROZE（0x60）

Freeze Host ID Access Configuration Register. This register is only accessible by CORE0.

Offset Address = 0x60

Bit	Name	Description	R/W	Reset Value
31:0	FROZE	CORE0 writes 1 to lock the core number that can access the MAILBOX registers; can only be cleared by system reset; writing 0 is invalid; If FROZE is written, reads back 0xffff_ffff; If FROZE is not written, reads back 0x0;	RW	0x0

8.7.11 ID_VAL（0x64）

Accessible Host ID Number Register. This register is only accessible by CORE0.

Offset Address = 0x64

Bit	Name	Description	R/W	Reset Value
31:1	-	Reserved	-	0x0
0	ID_VAL	Written by CORE0 with the host ID number of the core that can access the MAILBOX registers (CORE1's core host_id is 0); If FROZE is written, reads back 0x0;	RW	0x0

8.7.12 ID_MASK（0xE4）

Accessible Host ID Number Mask Register. This register is only accessible by CORE0.

Offset Address = 0xE4

Bit	Name	Description	R/W	Reset Value

31:1	-	Reserved	-	0x0
0	MASK	CORE0 writes the mask for the host ID of the core that can access the MAILBOX registers; 0: Do not check if CORE1's host_id equals ID_VAL; 1: Check if CORE1's host_id equals ID_VAL; If FROZE is written, reads back 0x0;	RW	0x0

8.7.13 MBX_KEY0（0x300）

Mailbox Password Register 0. This register is only accessible by CORE0.

Offset Address = 0x3000

Bit	Name	Description	R/W	Reset Value
31:0	MBX_KEY0	CORE0 writes the password register; the password is 0xA538_E827; =0xA538_E827, writes to FROZE and RESET registers are valid; =Other, writes to FROZE and RESET registers are invalid;	RW	0x0

8.7.14 MBX_KEY1（0x304）

Mailbox Password Register 1. This register is only accessible by CORE1.

Offset Address = 0x304

Bit	Name	Description	R/W	Reset Value
31:0	MBX_KEY1	CORE1 writes the password register; the password is 0xA538_EC63; =0xA538_EC63, writes to CSR.MBX_ALB_UNLINK and CSR.MBX_LINK are valid; =Other, writes to CSR.MBX_ALB_UNLINK and CSR.MBX_LINK are invalid;	RW	0x0

8.8 Software Operation Flow

8.8.1 Module Initialization Flow

The MAILBOX module must be initialized; otherwise, any host access to the MAILBOX module's registers and SRAM is prohibited (if CORE1 operates the MAILBOX module's registers and SRAM before CORE0 finishes initializing MAILBOX, it will cause an exception), and an error flag is returned to the system. The specific flow is as follows:

- (Optional) CORE0 writes MAILBOX->ID_VAL = 0x0;
- (Optional) CORE0 writes MAILBOX->ID_MASK = 0x1;
- (Mandatory) CORE0 writes MAILBOX->MBX_KEY0 = 0xA538_E827;
- (Mandatory) CORE0 writes MAILBOX->FROZE |= 1;

8.8.2 Polling Mode Communication Flow

8.8.2.1 CORE1 Sends Data to CORE0 via Input Mailbox

Table 8-5 CORE1 Sends Data to CORE0 via Input Mailbox Flow

Step	CORE0 Software Operation Flow	CORE1 Software Operation Flow
1	MAILBOX Module Initialization: 1. (Optional) Write MAILBOX->ID_VAL =0x0; 2. (Optional) Write MAILBOX->ID_MASK = 0x1; 3. (Mandatory) Write mailbox password MBX_KEY0->MBX_KEY0 =0xA538_E827; 4. (Mandatory) Write MAILBOX->FROZE |= 1;	Cannot access MAILBOX module before initialization is complete, otherwise CORE1 will enter an exception;
2	(Mandatory!) Wait for CORE1 to connect to the mailbox channel, read CSR.MBX_LINK to determine if CORE1 is connected to the mailbox;	Connect to mailbox channel: 1. Write mailbox password MBX_KEY1->MBX_KEY1 = 0xA538_EC63; 2. Check if the mailbox channel is locked, read LOCKOUT register to check if the mailbox channel allows CORE1 to connect; if not connectable, write LOCKOUT->MBX = 0x0 to unlock the mailbox; 3. Check if the mailbox channel is free, read CSR.MBX_ALB_UNLINK to determine if the mailbox channel can be connected by CORE1; 4. Configure CSR.MBX_LINK=1 to connect CORE1 to the mailbox channel; 5. Read CSR.MBX_LINK to check successful connection to the mailbox channel;
3	Wait for CORE1 to send input mailbox data, read CSR.MBX_IN_FULL to determine if data has been written to the input mailbox;	Write data to the input mailbox, after writing, configure CSR.MBX_IN_FULL=1;
4	Read data from the input mailbox, after reading, write 1 to CSR.MBX_IN_FULL to clear the flag;	Wait for CORE0 to read data from the input mailbox;

8.8.2.2 CORE1 Reads Data from CORE0 via Output Mailbox

Table 8-6 CORE1 Reads Data from CORE0 via Output Mailbox Flow

Step	CORE0 Software Operation Flow	CORE1 Software Operation Flow
1	MAILBOX Module Initialization: 1. (Optional) Write MAILBOX->ID_VAL =0x0; 2. (Optional) Write MAILBOX->ID_MASK = 0x1; 3. (Mandatory) Write mailbox password MBX_KEY0->MBX_KEY0 =0xA538_E827; 4. (Mandatory) Write MAILBOX->FROZE |= 1;	Cannot access MAILBOX module before initialization is complete;
2	(Mandatory!) Wait for CORE1 to connect to the mailbox channel, read CSR.MBX_LINK to determine if CORE1 is connected to the mailbox;	Connect to mailbox channel: 1. Write mailbox password MBX_KEY1->MBX_KEY1 = 0xA538_EC63;

		2. Check if the mailbox channel is locked, read LOCKOUT register to check if the mailbox channel allows CORE1 to connect; if not connectable, write LOCKOUT->MBX = 0x0 to unlock the mailbox; 3. Check if the mailbox channel is free, read CSR.MBX_ALB_UNLINK to determine if the mailbox channel can be connected by CORE1; 4. Configure CSR.MBX_LINK=1 to connect CORE1 to the mailbox channel; 5. Read CSR.MBX_LINK to check successful connection to the mailbox channel;
3	Write data to the output mailbox, after writing, configure CSR.MBX_OUT_FULL=1;	Wait for CORE0 to send output mailbox data, read CSR.MBX_OUT_FULL to determine if data has been written to the output mailbox;
4	Wait for CORE1 to read data;	Read data from the output mailbox, after reading, write 1 to CSR.MBX_OUT_FULL to clear the flag;

8.8.3 Interrupt Mode Communication Flow

Table 8-7 Interrupt Mode Communication Flow 1

Step	CORE0 Software Operation Flow	CORE1 Software Operation Flow
1	MAILBOX Module Initialization: 1. (Optional) Write MAILBOX->ID_VAL = 0x0; 2. (Optional) Write MAILBOX->ID_MASK = 0x1; 3. (Mandatory) Write Mailbox Password MBX_KEY0->MBX_KEY0 = 0xA538_E827; 4. (Mandatory) Write MAILBOX->FROZE |= 1;	MAILBOX module cannot be accessed before initialization is complete;
2	Enable Input Mailbox Full Interrupt, Output Mailbox Empty Interrupt; Wait for CORE1 to write data to Input Mailbox, triggering CORE0's Input Mailbox Full Interrupt;	Write Mailbox Password MBX_KEY1->MBX_KEY1 = 0xA538_EC63; Enable Mailbox Connectable Interrupt (optional), wait to enter Mailbox Connectable Interrupt;
3		Connect Mailbox Channel (refer to the above process); After successful connection, enable Input Mailbox Empty Interrupt, Output Mailbox Full Interrupt; Write data to Input Mailbox, after writing set CSR.MBX_IN_FULL=1;
4	Upon receiving Input Mailbox Full Interrupt, read data from Input Mailbox, after reading write 1 to CSR.MBX_IN_FULL to clear the flag, triggering CORE1's Input Mailbox Empty Interrupt;	Wait for CORE0 to read data, wait for CORE1's Input Mailbox Interrupt;
5	Write data to Output Mailbox, after writing set CSR.MBX_OUT_FULL=1;	Upon receiving Input Mailbox Empty Interrupt, indicating Input Mailbox is ready for new data; Wait for CORE0 to write data to Output Mailbox,

		triggering CORE1's Output Mailbox Full Interrupt;
6	Wait for CORE1 to read data, wait for CORE0's Output Mailbox Empty Interrupt;	Upon receiving Output Mailbox Full Interrupt, read data from Output Mailbox, after reading write 1 to CSR.MBX_OUT_FULL to clear the flag;
7	Upon receiving Output Mailbox Empty Interrupt, indicating Output Mailbox is ready for new data;	

Table 8-8 Interrupt Mode Communication Flow 2

Step	CORE0 Software Operation Flow	CORE1 Software Operation Flow
1	MAILBOX Module Initialization: 1. (Optional) Write MAILBOX->ID_VAL = 0x0; 2. (Optional) Write MAILBOX->ID_MASK = 0x1; 3. (Mandatory) Write Mailbox Password MBX_KEY0->MBX_KEY0 = 0xA538_E827; 4. (Mandatory) Write MAILBOX->FROZE |= 1;	MAILBOX module cannot be accessed before initialization is complete;
2	Enable Input Mailbox Full Interrupt; Wait for CORE1 to write data to Input Mailbox, triggering CORE0's Input Mailbox Full Interrupt;	Write Mailbox Password MBX_KEY1->MBX_KEY1 = 0xA538_EC63; Enable Mailbox Connectable Interrupt (optional), wait to enter Mailbox Connectable Interrupt;
3		Connect Mailbox Channel (refer to the above process); After successful connection, enable Output Mailbox Full Interrupt; Write data to Input Mailbox, after writing set CSR.MBX_IN_FULL=1;
4	Upon receiving Input Mailbox Full Interrupt, read data from Input Mailbox, after reading write 1 to CSR.MBX_IN_FULL to clear the flag;	
5	Write data to Output Mailbox, after writing set CSR.MBX_OUT_FULL=1;	Wait for CORE0 to write data to Output Mailbox, triggering CORE1's Output Mailbox Full Interrupt;
6		Upon receiving Output Mailbox Interrupt, read data from Output Mailbox, after reading write 1 to CSR.MBX_OUT_FULL to clear the flag;

8.8.4 Application Notes

After CORE1 is reset individually, CORE0 needs to perform the following operations to clear the mailbox status:

- Write RESET.MBX_UNLINK register to disconnect CORE1 from the mailbox channel;
- Write RESET.MBX_OUT_EMPTY register to clear the Output Mailbox Full status;
- Write CSR.MBX_IN_FULL register to clear the Input Mailbox Full status;

9 GIC

9.1 Introduction

The Global Interrupt Controller (GIC) can handle all interrupts from the system, distributes each interrupt to the corresponding core according to configuration, and provides an inter-core interrupt mechanism.

9.2 Features

The features of this module are as follows:

- Supports distribution of 128 SOC IRQ interrupts; all SOC IRQs are level signals.
- Supports distribution of Inter-Core Interrupts (ICI).
- Supports transferring 8-bit ICI information from the sending core to the target core.
- Inter-Core Interrupts (ICI) are distributed through the GIC; other interrupts need to be distributed through the GIC.
- GIC interrupt distribution can only be operated by CORE0.

9.3 Block Diagram

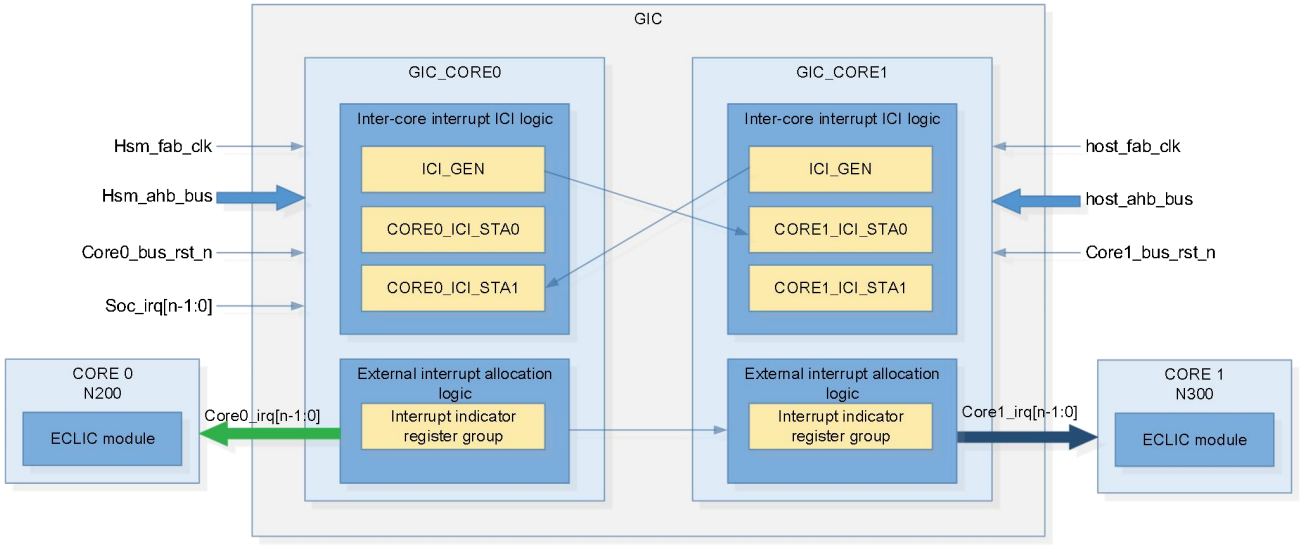


Figure 9-1 GIC Module Block Diagram

9.4 Functional Description

9.4.1 External Interrupt Distribution

The GIC acts as an interrupt distributor in the system. Configuring the SOC_INTn_INDICATOR register via CORE0 enables the following functions:

- Broadcast all external interrupts to a specific core or all cores.
- Allow a specific core to respond to a specific interrupt, while other cores cannot respond to this interrupt.
- The SOC_INTn_INDICATOR register allocates interrupts for the interrupt with sequence number n in the interrupt vector table.
- Inter-Core Interrupts (ICI) do not use the SOC_INTn_INDICATOR register for interrupt allocation.

9.4.2 Inter-Core Interrupt (ICI)

Inter-Core Interrupt (ICI) refers to the ability of one core to send an interrupt to another core in a multi-core system.

- CORE0 supports configuring CORE0 to send ICIs to CORE1/CORE0.
- CORE1 supports configuring CORE1 to send ICIs to CORE1/CORE0.

Usage flow for ICI (CORE n sending interrupt to CORE m):

- CORE n writes the target core CORE_id to ICI_GEN[0].
- Then the COREm_INT_STAn register updates, triggering an ICI from CORE n to CORE m.
- When CORE m receives the ICI and enters the Interrupt Service Routine (ISR), it can query the COREm_INI_STAn register flag to determine that the current ICI was triggered by CORE n; it must write 1 to clear the interrupt flag before exiting the ISR.
- GIC supports transferring 8-bit interrupt information via ICI_GEN [31:24]. The target core COREm can obtain this information by reading the value of COREm_ICI_STAn[31:24].

9.4.3 Interrupt Vector Table

The list of interrupts and events output by the GIC module is as follows:

Table 9-1 Interrupt Vector Table

Int Vec Num	SOC IRQ Num	Peripheral	Interrupt Source	Note
146	127	MAILBOX_HOST	Large Core Interrupt	CORE1 Exclusive
145	126	MAILBOX_INTERNAL	Small Core Interrupt	CORE0 Exclusive
144	125	CORE0_SYSREG	System Control (rccal\boi\plvd5) Interrupt	CORE0 Exclusive
143	124	CORE1_SYSREG	System Control (boi) Interrupt	CORE1 Exclusive
142	123	CORE1_RST_IRQ	CORE1 Reset Interrupt	CORE0 Exclusive
141	122	SCU	Single Interrupt	CORE0 Exclusive
140	121	LCSS	VCC Domain LCSS Interrupt	CORE0 Exclusive
139	120	LVDCMP	LVD, CMP1, CMP2 Interrupt	
138	119	MADC	SAR External Measurement Interrupt, VBAT POR or BOR Reset Interrupt, Lid Open Detection Event Interrupt, Software Freeze Time Interrupt, VBAT Domain LCSS Interrupt	
137	118	RTC	Timer Interrupt and Month-Day-Hour-Minute-Second	

			Interrupt	
136	117	D2F	Single Interrupt	
135	116	ADC0	DMA\Comparison\Measurement Interrupt	
134	115	DAC	Single Interrupt	
133	114	KEYSTORE	Single Interrupt	
132	113	SEA	Single Interrupt	
131	112	AES	Single Interrupt	
130	111	M2M	Single Interrupt	
129	110	DSP	Single Interrupt	
128	109	QSPI1	Single Interrupt	
127	108	QSPI0	Single Interrupt	
126	107	SPI3	Single Interrupt	
125	106	SPI2	Single Interrupt	
124	105	SPI1	Single Interrupt	
123	104	SPI0	Single Interrupt	
122	103	Reserved	--	
121	102	USART1	Single Interrupt	
120	101	USART0	Single Interrupt	
119	100	UART7	Single Interrupt	
118	99	UART6	Single Interrupt	
117	98	UART5	Single Interrupt	
116	97	UART4	Single Interrupt	
115	96	UART3	Single Interrupt	
114	95	UART2	Single Interrupt	
113	94	UART1	Single Interrupt	
112	93	UART0	Single Interrupt	
111	92	LSRX	Single Interrupt	
110	91	LPUART	Single Interrupt	
109	90	I2C1	Single Interrupt	
108	89	I2C0	Single Interrupt	
107	88	7816_1	Single Interrupt	
106	87	7816_0	Single Interrupt	
105	86	TC1	Single Interrupt	
104	85	TC0	Single Interrupt	
103	84	SIMPTC3_3	Single Interrupt	
102	83	SIMPTC3_2	Single Interrupt	
101	82	SIMPTC3_1	Single Interrupt	
100	81	SIMPTC3_0	Single Interrupt	
99	80	SIMPTC2_3	Single Interrupt	
98	79	SIMPTC2_2	Single Interrupt	
97	78	SIMPTC2_1	Single Interrupt	

96	77	SIMPTC2_0	Single Interrupt	
95	76	SIMPTC1_3	Single Interrupt	
94	75	SIMPTC1_2	Single Interrupt	
93	74	SIMPTC1_1	Single Interrupt	
92	73	SIMPTC1_0	Single Interrupt	
91	72	SIMPTC0_3	Single Interrupt	
90	71	SIMPTC0_2	Single Interrupt	
89	70	SIMPTC0_1	Single Interrupt	
88	69	SIMPTC0_0	Single Interrupt	
87	68	SYS_TC	Single Interrupt	CORE0 Exclusive
86	67	General DMA00	Single Interrupt	CORE0 Exclusive
85	66	General DMA01	Single Interrupt	CORE0 Exclusive
84	65	General DMA10	Single Interrupt	CORE1 Exclusive
83	64	General DMA11	Single Interrupt	CORE1 Exclusive
82	63	WDT1	Single Interrupt	
81	62	WDT0	Single Interrupt	
80	61	KBI0	Multi-channel Combined Interrupt	
79	60	KBI0_15	Single Interrupt	
78	59	KBI0_14	Single Interrupt	
77	58	KBI0_13	Single Interrupt	
76	57	KBI0_12	Single Interrupt	
75	56	KBI0_11	Single Interrupt	
74	55	KBI0_10	Single Interrupt	
73	54	KBI0_9	Single Interrupt	
72	53	KBI0_8	Single Interrupt	
71	52	KBI0_7	Single Interrupt	
70	51	KBI0_6	Single Interrupt	
69	50	KBI0_5	Single Interrupt	
68	49	KBI0_4	Single Interrupt	
67	48	KBI0_3	Single Interrupt	
66	47	KBI0_2	Single Interrupt	
65	46	KBI0_1	Single Interrupt	
64	45	KBI0_0	Single Interrupt	
63	44	KBI1	Multi-channel Combined Interrupt	
62	43	KBI1_15	Single Interrupt	
61	42	KBI1_14	Single Interrupt	
60	41	KBI1_13	Single Interrupt	

59	40	KBI1_12	Single Interrupt	
58	39	KBI1_11	Single Interrupt	
57	38	KBI1_10	Single Interrupt	
56	37	KBI1_9	Single Interrupt	
55	36	KBI1_8	Single Interrupt	
54	35	KBI1_7	Single Interrupt	
53	34	KBI1_6	Single Interrupt	
52	33	KBI1_5	Single Interrupt	
51	32	KBI1_4	Single Interrupt	
50	31	KBI1_3	Single Interrupt	
49	30	KBI1_2	Single Interrupt	
48	29	KBI1_1	Single Interrupt	
47	28	KBI1_0	Single Interrupt	
46	27	LCD	Single Interrupt	
45	26	--	Reserved	
44	25	--	Reserved	
43	24	--	Reserved	
42	23	--	Reserved	
41	22	--	Reserved	
40	21	--	Reserved	
39	20	--	Reserved	
38	19	--	Reserved	
37	18	--	Reserved	
36	17	--	Reserved	
35	16	--	Reserved	
34	15	--	Reserved	
33	14	--	Reserved	
32	13	--	Reserved	
31	12	--	Reserved	
30	11	--	Reserved	
29	10	--	Reserved	
28	9	--	Reserved	
27	8	--	Reserved	
26	7	--	Reserved	
25	6	--	Reserved	
24	5	--	Reserved	
23	4	--	Reserved	
22	3	--	Reserved	
21	2	--	Reserved	
20	1	--	Reserved	
19	0	INTER_CORE	Inter-Core Interrupt	
18	--	--	Reserved	

17	--	--	Reserved	
16	--	--	Reserved	
15	--	--	Reserved	
14	--	--	Reserved	
13	--	--	Reserved	
12	--	--	Reserved	
11	--	--	Reserved	
10	--	--	Reserved	
9	--	--	Reserved	
8	--	--	Reserved	
7	--	SysTimer	Core-internal Timer Interrupt	
6	--	--	Reserved	
5	--	--	Reserved	
4	--	--	Reserved	
3	--	SysTimerSW	Core-internal Timer Software Interrupt	
2	--	--	Reserved	
1	--	--	Reserved	
0	--	--	Reserved	

9.5 Registers

Table 9-2 GIC Module Register Base Addresses

Module Name	CORE0 Address	CORE1 Address
GIC	0x3001_C000	0xB001_C000

Table 9-3 GIC Module Register Offset Addresses

Register Name	Offset Address	Default Value	Description	R/W Permission	
				CORE0	CORE1
CORE0_ICI_STA0	0x0	0x0000_0000	CORE0 ICI Status Register 0	W1C	Inaccessible
CORE0_ICI_STA1	0x4	0x0000_0000	CORE0 ICI Status Register 1	W1C	Inaccessible
CORE1_ICI_STA0	0x10	0x0000_0000	CORE1 ICI Status Register 0	Inaccessible	W1C
CORE1_ICI_STA1	0x14	0x0000_0000	CORE1 ICI Status Register 1	Inaccessible	W1C
ICI_GEN	0x20	0x0000_0000	Inter-Core Interrupt Generation Register	WO	WO
CORE_IND_LOCK	0x24	0x0000_0000	CORE Interrupt Lock Register	RW	Inaccessible
CORE_NUM	0x28	0x0000_0002	CORE Number Register	RO	RO
HART_NUM	0x2C	0x0000_0002	HART Number Register	RO	RO

SOC_IRQ_NUM	0x30	0x0000_004F	External Interrupt Number Register	RO	RO
SOC_INTn_INDICATOR, n=20~146	0x40~0x40+4*(n-20)	0x0000_0000	SOC Interrupt Indicator Register	RW	RO

Note: W1C means Write 1 to Clear, RW means Read/Write, RO means Read Only, WO means Write Only.

9.5.1 CORE0_ICI_STA0（0x00）

CORE0 Inter-Core Interrupt Status Register 0

Offset Address = 0x00

Only the CORE0 host can read/write this register.

Bit	Name	Description	R/W	Reset
31:24	ICI_INFO	8-bit message received by CORE0 from CORE0 along with the ICI.	RO	0x0
23:1	-	Reserved	RO	0x0
0	ICI_STATUS	ICI status bit for CORE0 sending ICI to CORE0;	W1C	0x0

9.5.2 CORE0_ICI_STA1（0x04）

CORE0 Inter-Core Interrupt Status Register 1

Offset Address = 0x04

Only the CORE0 host can read/write this register.

Bit	Name	Description	R/W	Reset
31:24	ICI_INFO	8-bit message received by CORE0 from CORE1 along with the ICI.	RO	0x0
23:1	-	Reserved	RO	0x0
0	ICI_STATUS	ICI status bit for CORE1 sending ICI to CORE0;	W1C	0x0

9.5.3 CORE1_ICI_STA0（0x10）

CORE1 Inter-Core Interrupt Status Register 0

Offset Address = 0x10

Only the CORE1 host can read/write this register.

Bit	Name	Description	R/W	Reset
31:24	ICI_INFO	8-bit message received by CORE1 from CORE0 along with the ICI.	RO	0x0
23:1	-	Reserved	RO	0x0
0	ICI_STATUS	ICI status bit for CORE0 sending ICI to CORE1;	W1C	0x0

9.5.4 CORE1_ICI_STA1（0x14）

CORE1 Inter-Core Interrupt Status Register 1

Offset Address = 0x14

Only the CORE1 host can read/write this register.

Bit	Name	Description	R/W	Reset
31:24	ICI_INFO	8-bit message received by CORE1 from CORE1 along with the ICI.	RO	0x0

23:1	-	Reserved	RO	0x0
0	ICI_STATUS	ICI status bit for CORE1 sending ICI to CORE1;	W1C	0x0

9.5.5ICI_GEN（0x20）

Inter-Core Interrupt Generation Register

Offset Address = 0x20

Bit	Name	Description	R/W	Reset
31:24	ICI_INT_INFO	8-bit message sent to the core receiving the ICI.	WO	0x0
23:1	-	Reserved	RO	0x0
0	DESTINATION	Write the CORE_id of the core receiving the ICI;	WO	0x0

(Note: ICI_INT_INFO and DESTINATION must be written simultaneously)

9.5.6CORE_IND_LOCK（0x24）

CORE Interrupt Lock Register

Offset Address = 0x24

Only the CORE0 host can read/write this register.

Bit	Name	Description	R/W	Reset
31:0	CORE_IND_LOCK	CORE Interrupt Lock Configuration Register =0x90CD_EFAB: Locked =Other values: Not locked Note: Once locked, it cannot be unlocked by writing other values; it can only be unlocked by reset.	RW	0x0

9.5.7CORE_NUM（0x28）

CORE Number Register

Offset Address = 0x28

Bit	Name	Description	R/W	Reset
31:0	CORE_NUM	Number of cores in the system;	RO	0x2

9.5.8HART_NUM（0x2C）

HART Number Register

Offset Address = 0x2C

Bit	Name	Description	R/W	Reset
31:0	HART_NUM	Number of HART buses in the system;	RO	0x2

9.5.9SOC_IRQ_NUM（0x30）

Interrupt Number Register

Offset Address = 0x30

Bit	Name	Description	R/W	Reset
31:0	SOC_IRQ_NUM	Number of external interrupts in the system;	RO	0x7F

9.5.10 SOC_INTn_INDICATOR, n=20~146（0x40~0x40+4*（n-20））

SOC Interrupt Indicator Register

Offset Address = 0x40~0x40+4*(n-20)

Bit	Name	Description	R/W	Reset
31:2	-	Reserved	RO	0x0
1	SOC_INT1_INDICATOR	Whether external interrupt n is allocated to CORE1; 1: Allocated to CORE1; 0: Not allocated to CORE1; CORE0 can read/write this bit, CORE1 can read this bit;	RW	0x0
0	SOC_INT0_INDICATOR	Whether external interrupt n is allocated to CORE0 1: Allocated to CORE0; 0: Not allocated to CORE0; CORE0 can read/write this bit, CORE1 cannot read this bit;	RW	0x0

9.6 Software Configuration Flow and Application Notes

9.6.1 Inter-Core Interrupt Configuration Flow

- Initialize the ECLIC module in the receiving core(s) and enable Inter-Core Interrupts (Both CORE0 and CORE1 require ECLIC module initialization).
- Sending core triggers the ICI:
  - If needing to send an ICI from CORE0 to CORE0 or CORE1, then CORE0 writes the receiving core's CORE_id to ICI_GEN[1:0], and simultaneously writes the 8-bit message for the interrupt transfer to ICI_GEN[31:24], immediately triggering an ICI.
  - If needing to send an ICI from CORE1 to CORE0 or CORE1, then CORE1 writes the receiving core's CORE_id to ICI_GEN[1:0], and simultaneously writes the 8-bit message for the interrupt transfer to ICI_GEN[31:24], immediately triggering an ICI.
- The receiving core enters the ISR, reads the COREx_ICI_STAx register to obtain the core that sent the ICI and the corresponding 8-bit interrupt message.
- The receiving core performs subsequent interrupt processing based on the information from the COREx_ICI_STAx register.

9.6.2 GIC Interrupt Distribution Configuration Flow

- Initialize the ECLIC module in the receiving core(s) and enable the interrupt module (Both CORE0 and CORE1 require ECLIC module initialization).
- According to the interrupt vector table, select the corresponding SOC Interrupt Indicator Register SOC_INTn_INDICATOR for configuration, distributing the interrupt with the corresponding number to CORE0/CORE1 or to both cores simultaneously (This operation can only be performed by CORE0).
- Enter the corresponding ISR for subsequent interrupt processing.

10 HSEM

10.1 Introduction

The Hardware Semaphore (HSEM) module provides a hardware semaphore mechanism for resource allocation in a dual-core system. When a core needs exclusive access to a resource, it can request a semaphore and release the semaphore after use. The chip supports a mechanism of 32 semaphore registers.

10.2 Functional Description

HSEM supports 32 hardware semaphores for user configuration and use. The registers related to semaphores are SEMAPHOREn.

The software usage requirements for using semaphores are as follows:

- All cores in the system agree to use the SEMAPHOREn semaphore register to protect critical resources (such as UART devices, shared memory, etc.); the correspondence between core ID, critical resource, and semaphore is defined by software.
- If a core COREn (n=0,1 similarly hereinafter) wants to access a critical resource, it should attempt to acquire the semaphore resource by writing to the semaphore register, then read the value of the semaphore and compare it with the written value; if they match, it means COREn successfully owns the semaphore and can access the critical resource; if they do not match, the semaphore acquisition fails and the critical resource cannot be accessed.
- When COREn owns the semaphore and completes work related to the critical resource, it must release the semaphore by writing all 1s (0x3FF) to the register; writing all 1s by other cores is invalid.
- In the same core, writing a non-0x3FF value multiple times to the same semaphore register is not allowed. For example, it is an illegal operation to write 0x001 to SEMAPHORE0, then without writing 0x3FF to release the semaphore, write 0x003 to SEMAPHORE0 again.

10.3 Block Diagram

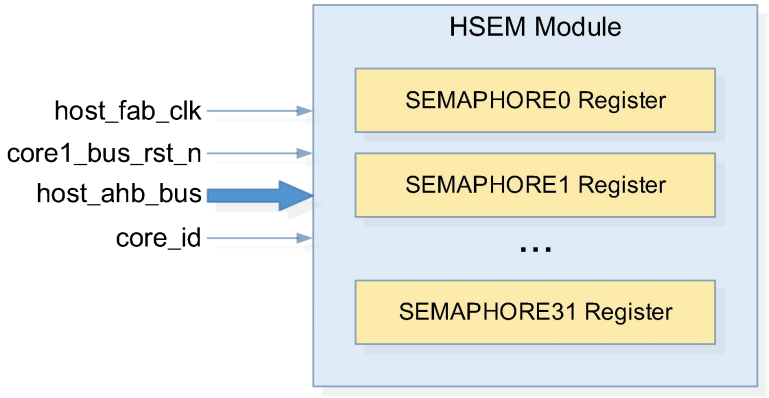


Figure 10-1 HSEM Module Block Diagram

10.4 Clock and Reset

The HSEM clock is the CORE1 system clock. Reset sources include external PIN reset, power-on/power-down reset, CPU reset, watchdog reset, and module soft reset.

10.5 Registers

Table 10-1 HSEM Module Register Base Addresses

Module Name	CORE0 Address	CORE1 Address
HSEM	/	0xB008_4000

Table 10-2 HSEM Module Register Offset Addresses

Register Name	Offset Address	Default Value	Description	R/W Permission	
				N200	N300
SEMAPHORE0~31	0x0~0x7C	0x3FF	Semaphore Register 0~31	RW	RW

10.5.1 SEMAPHORE0~31（0x00~0x7C）

Semaphore Register 0~31

Offset Address = 0x00~0x7C

Bit	Name	Description	R/W	Reset
31:10	-	Reserved	R	0x0
9:0	SEMA_STATUS	Writing a non-0x3FF value indicates a specific CORE has acquired the semaphore.	RW	0x3FF

10.6 Software Configuration Flow and Application Notes

10.6.1 HSEM (Hardware Semaphore) Configuration Flow

- HSEM module initialization (operated only by CORE1): Enable the HSEM clock gating in the system register chapter.
- If a core COREn (n=0,1 similarly hereinafter) wants to access a critical resource, it attempts to acquire the semaphore resource by writing to the semaphore register, then reads the value of the semaphore and compares it with the written value; if they match, it means COREn successfully owns the semaphore and can access the critical resource; if they do not match, the semaphore acquisition fails and the critical resource cannot be accessed.
- If one core has acquired the semaphore, another core attempting to acquire the semaphore resource will fail.
- When COREn owns the semaphore and completes work related to the critical resource, it must release the semaphore by writing all 1s (0x3FF) to the register; writing all 1s by other cores is invalid (other cores cannot release the semaphore).

11 SCU

11.1 Introduction

SCU (Security Configuration Unit) is a security management component.

In a dual-core system architecture (CORE0 as security core, CORE1 as application core), the SCU is used to configure peripherals with dynamically assignable security attributes after power-on. These peripherals include: SRAM, GPIO, and peripherals with a PPC (Peripheral Protection Controller) interface as a security check mechanism.

Additionally, the SCU outputs the security attributes of SRAM and peripherals with a dual-bus port PPC (DPPC) interface to the CMU and RMU. The CMU and RMU select the clock domain and reset corresponding to CORE0 SOC or CORE1 SOC based on the peripheral's security attributes.

Simultaneously, the SCU outputs the security attributes of peripherals capable of initiating requests to the DMA to the DMA, for the DMA to perform security checks on peripherals.

Finally, the SCU also aggregates illegal security access events for peripherals with a PPC interface and generates an interrupt to the GIC, which the GIC software allocates to CORE0.

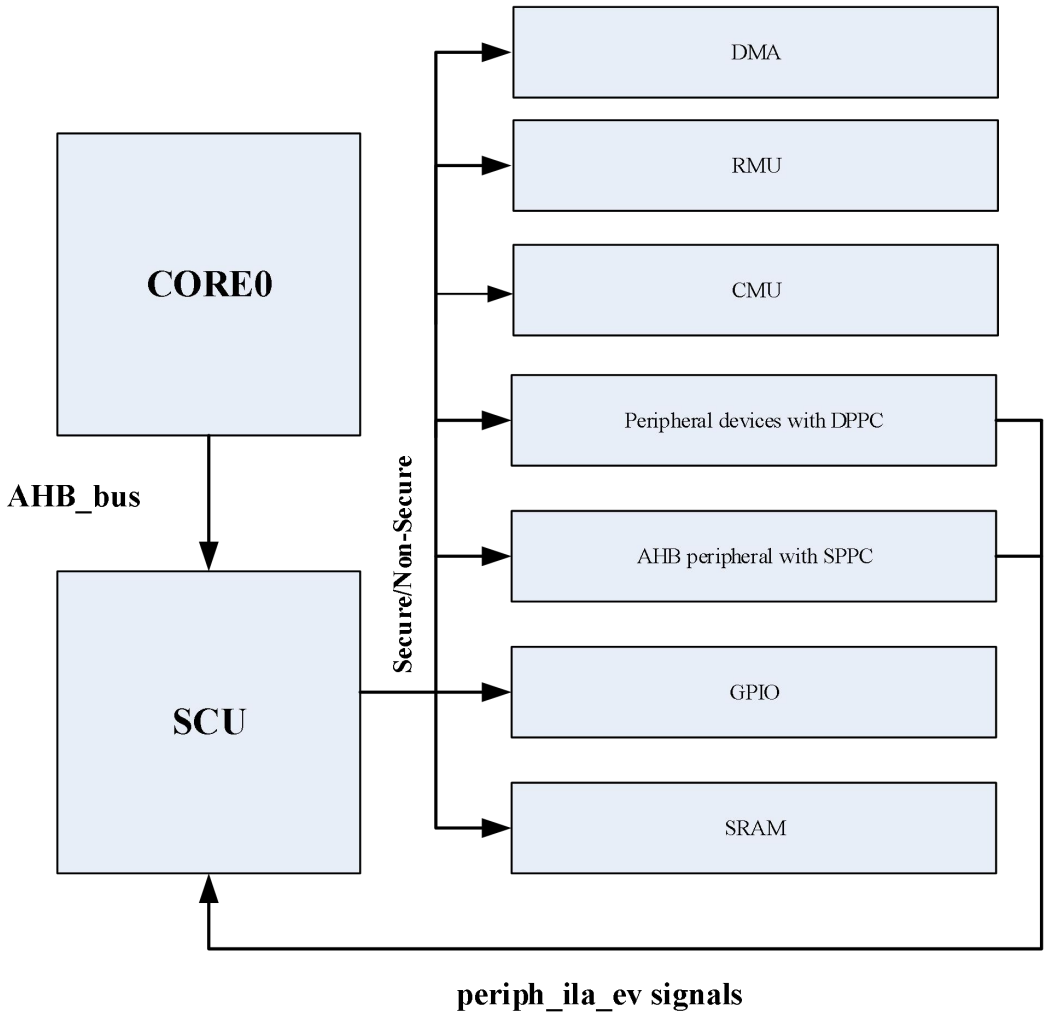


Figure 11-1 SCU System Block Diagram

11.2 Features

- All SCU registers have write password protection (SCU_WKEY) functionality.
- For internal security configuration registers, CORE0 has read and write permissions, CORE1 has only read permission.
- For internal interrupt control registers, only CORE0 has read and write access permissions, CORE1 has no read or write access permissions.
- SCU security attribute configuration supports a lock function; after locking, configuration cannot be changed until reset unlocks it.
- Upon power-up, all peripherals with dynamically assignable security attributes default to secure peripherals.
- Supports different security configuration granularities for peripherals:
  - Peripherals with PPC: Support security attribute configuration for individual peripherals with PPC.
  - GPIO: Support security attribute configuration for individual GPIO pins.
  - SRAM: Supports splitting the 160 KB SRAM into two parts: 16 KB * n of secure SRAM and 16 KB * (10 - n) of non-secure SRAM from the contiguous 160 KB SRAM.
- Supports illegal security access events for peripherals with PPC, generating interrupts to the GIC.
- Illegal security access handling for other peripherals (such as GPIO, SRAM, SCU, GIC, etc.) is RAZWI (Read as zero, Write ignore).
- Outputs security attributes of SRAM and peripherals with DPPC to CMU and RMU for selecting clock domain, clock gating, and reset source. Secure peripherals select CORE0's clock domain and reset; non-secure peripherals select CORE1's clock domain and reset source.
- Outputs peripheral security attributes to the DMA for the DMA to perform peripheral security checks.

11.3 Peripheral and Security Mechanism List

Table 11-1 Peripheral and Security Mechanism Table

Peripheral	Peripheral Type	Security Identification Mechanism	Security Attribute Configurable via SCU	Illegal Access Handling	Peripheral Bus Interface
WDT0	S	Physical Isolation	Not Configurable	RAZ/WI	APB
WDT1	NS	Physical Isolation	Not Configurable	RAZ/WI	APB
SYS_REG_CORE0	S	Physical Isolation	Not Configurable	RAZ/WI	APB
SYS_REG_CORE1	NS	Physical Isolation	Not Configurable	RAZ/WI	APB
GDMA00	S	Physical Isolation	Not Configurable	RAZ/WI	APB
GDMA01	NS	Physical Isolation	Not Configurable	RAZ/WI	APB
GDMA10	S	Physical Isolation	Not Configurable	RAZ/WI	APB
GDMA11	NS	Physical Isolation	Not Configurable	RAZ/WI	APB
LCSS	S	Physical Isolation	Not Configurable	RAZ/WI	APB
SYSTC	S	Physical Isolation	Not Configurable	RAZ/WI	APB
CRC1	S	Physical Isolation	Not Configurable	RAZ/WI	APB
AES	O(NS/S)	PPC(DPPC_AHB)	SCU Configurable	S Event and RAZ/WI	AHB
SEA	O(NS/S)	PPC(DPPC_AHB)	SCU Configurable	S Event and RAZ/WI	AHB
QSPI1	O(NS/S)	PPC(SPPC_AHB)	SCU Configurable	S Event and RAZ/WI	AHB
QSPI0	O(NS/S)	PPC(SPPC_AHB)	SCU Configurable	S Event and RAZ/WI	AHB

KBI1	O(NS/S)	PPC(DPPC_APB)	SCU Configurable	S Event and RAZ/WI	APB
KBI0	O(NS/S)	PPC(DPPC_APB)	SCU Configurable	S Event and RAZ/WI	APB
CRC0	O(NS/S)	PPC(DPPC_APB)	SCU Configurable	S Event and RAZ/WI	APB
LVD_CMP	O(NS/S)	PPC(DPPC_APB)	SCU Configurable	S Event and RAZ/WI	APB
KBI1	O(NS/S)	PPC(DPPC_APB)	SCU Configurable	S Event and RAZ/WI	APB
KBI0	O(NS/S)	PPC(DPPC_APB)	SCU Configurable	S Event and RAZ/WI	APB
LSRX	O(NS/S)	PPC(DPPC_APB)	SCU Configurable	S Event and RAZ/WI	APB
ANA(RTC&SAR&GPIOL)	O(NS/S)	PPC(DPPC_APB)	SCU Configurable	S Event and RAZ/WI	APB
SIMPTC3	O(NS/S)	PPC(DPPC_APB)	SCU Configurable	S Event and RAZ/WI	APB
SIMPTC2	O(NS/S)	PPC(DPPC_APB)	SCU Configurable	S Event and RAZ/WI	APB
SIMPTC1	O(NS/S)	PPC(DPPC_APB)	SCU Configurable	S Event and RAZ/WI	APB
SIMPTC0	O(NS/S)	PPC(DPPC_APB)	SCU Configurable	S Event and RAZ/WI	APB
TC1	O(NS/S)	PPC(DPPC_APB)	SCU Configurable	S Event and RAZ/WI	APB
TC0	O(NS/S)	PPC(DPPC_APB)	SCU Configurable	S Event and RAZ/WI	APB
DAC	O(NS/S)	PPC(DPPC_APB)	SCU Configurable	S Event and RAZ/WI	APB
ADC0	O(NS/S)	PPC(DPPC_APB)	SCU Configurable	S Event and RAZ/WI	APB
D2F	O(NS/S)	PPC(DPPC_APB)	SCU Configurable	S Event and RAZ/WI	APB
M2M	O(NS/S)	PPC(DPPC_APB)	SCU Configurable	S Event and RAZ/WI	APB
DSP	O(NS/S)	PPC(DPPC_APB)	SCU Configurable	S Event and RAZ/WI	APB
LCD	O(NS/S)	PPC(DPPC_APB)	SCU Configurable	S Event and RAZ/WI	APB
IOCNT	O(NS/S)	PPC(DPPC_APB)	SCU Configurable	S Event and RAZ/WI	APB
Reserved	O(NS/S)	PPC(DPPC_APB)	SCU Configurable	S Event and RAZ/WI	APB
I2C1	O(NS/S)	PPC(DPPC_APB)	SCU Configurable	S Event and RAZ/WI	APB
I2C0	O(NS/S)	PPC(DPPC_APB)	SCU Configurable	S Event and RAZ/WI	APB
SPI3	O(NS/S)	PPC(DPPC_APB)	SCU Configurable	S Event and RAZ/WI	APB
SPI2	O(NS/S)	PPC(DPPC_APB)	SCU Configurable	S Event and RAZ/WI	APB
SPI1	O(NS/S)	PPC(DPPC_APB)	SCU Configurable	S Event and RAZ/WI	APB
SPI0	O(NS/S)	PPC(DPPC_APB)	SCU Configurable	S Event and RAZ/WI	APB
ISO78161	O(NS/S)	PPC(DPPC_APB)	SCU Configurable	S Event and RAZ/WI	APB
ISO78160	O(NS/S)	PPC(DPPC_APB)	SCU Configurable	S Event and RAZ/WI	APB
USART1	O(NS/S)	PPC(DPPC_APB)	SCU Configurable	S Event and RAZ/WI	APB
USART0	O(NS/S)	PPC(DPPC_APB)	SCU Configurable	S Event and RAZ/WI	APB
LPUART	O(NS/S)	PPC(DPPC_APB)	SCU Configurable	S Event and RAZ/WI	APB
UART7	O(NS/S)	PPC(DPPC_APB)	SCU Configurable	S Event and RAZ/WI	APB
UART6	O(NS/S)	PPC(DPPC_APB)	SCU Configurable	S Event and RAZ/WI	APB
UART5	O(NS/S)	PPC(DPPC_APB)	SCU Configurable	S Event and RAZ/WI	APB
UART4	O(NS/S)	PPC(DPPC_APB)	SCU Configurable	S Event and RAZ/WI	APB
UART3	O(NS/S)	PPC(DPPC_APB)	SCU Configurable	S Event and RAZ/WI	APB
UART2	O(NS/S)	PPC(DPPC_APB)	SCU Configurable	S Event and RAZ/WI	APB
UART1	O(NS/S)	PPC(DPPC_APB)	SCU Configurable	S Event and RAZ/WI	APB
UART0	O(NS/S)	PPC(DPPC_APB)	SCU Configurable	S Event and RAZ/WI	APB
GPIO	O(P)	Secure-aware	SCU Configurable	RAZ/WI	APB

SRAM	O(P)	Secure-aware	SCU Configurable	RAZ/WI	APB
GIC	P	Secure-aware	Not Configurable	RAZ/WI	AHB
HSEM	P	Secure-aware	Not Configurable	RAZ/WI	AHB
FLASH_REG	P	Secure-aware	Not Configurable	RAZ/WI	AHB
MAILBOX	P	Secure-aware	Not Configurable	RAZ/WI	AHB
AOS	P	Secure-aware	Not Configurable	RAZ/WI	APB
FLASH	P	Secure-aware	Not Configurable	RAZ/WI	AHB
ROM	S	Secure-aware	Not Configurable	RAZ/WI	AHB

Note 1:

- **S: Secure Peripheral;**
- **NS: Non-Secure Peripheral;**
- **P: Partially Secure and Partially Non-Secure Peripheral;**
- **O(NS/S): Configurable as Secure or Non-Secure Peripheral;**
- **O(P): Configurable as Partially Secure and Partially Non-Secure Peripheral**

Note 2:

- **Physical Isolation: Isolated on the bus;**
- **PPC: Peripheral Protection Controller;**
- **Secure-aware: Peripheral module with built-in security attribute checking;**

11.4 Structure Block Diagram

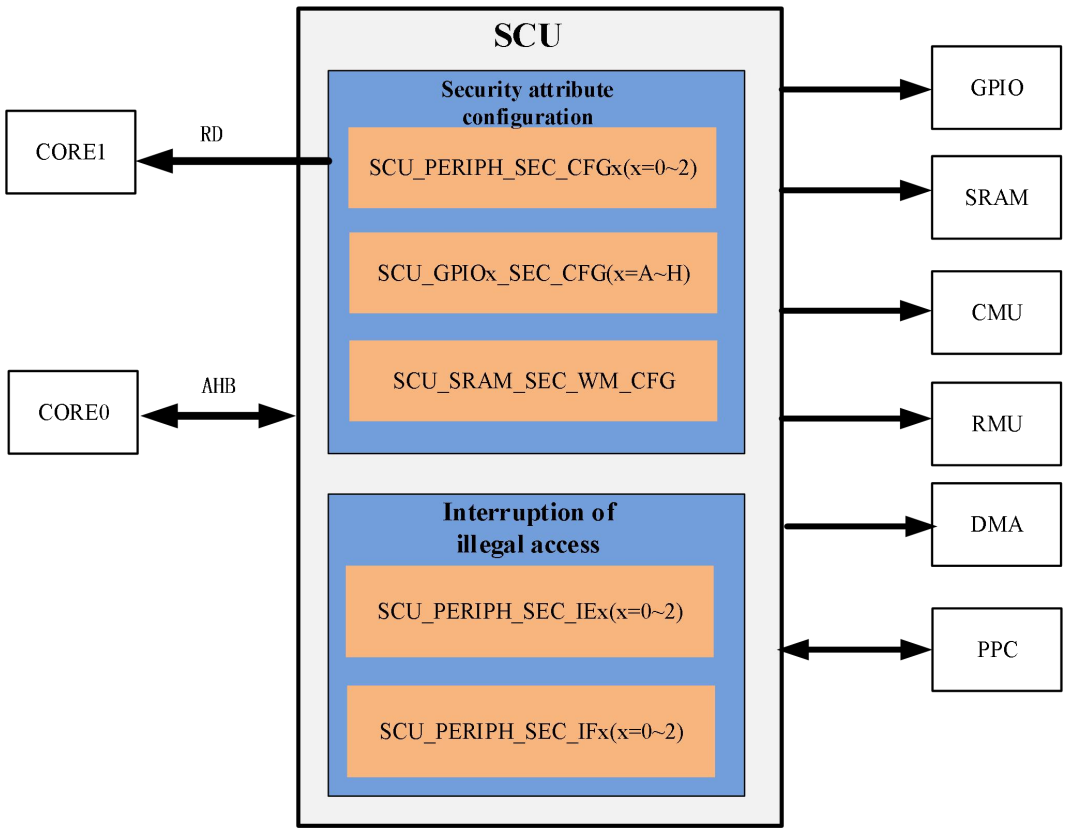


Figure 11-2 SCU Structure Block Diagram

The SCU configures the security attributes of peripherals with PPC by setting SCU_PERIPH_SEC_CFG0, SCU_PERIPH_SEC_CFG1, and SCU_PERIPH_SEC_CFG2. Each bit corresponds to the security attribute of one peripheral.

The SCU configures the security attribute of any GPIO pin by setting SCU_GPIOx_SEC_CFG. Each bit corresponds to the security attribute of one GPIO pin.

The SCU divides the non-secure SRAM and secure SRAM by configuring the watermark (dividing line) in SCU_SRAM_SEC_WM_CFG, splitting the contiguous 160 KB SRAM into 16 KB * n of secure SRAM and 16 KB * (10 - n) of non-secure SRAM.

The SCU integrates illegal access events for peripherals with PPC. The corresponding peripheral's illegal access status can be queried from SCU_PERIPH_SEC_IFx. The illegal access interrupt for the corresponding peripheral is enabled via SCU_PERIPH_SEC_IEx.

Illegal accesses are handled using the RAZWI (Read as zero, Write ignore) method. Illegal accesses to peripherals with PPC also generate an interrupt. (See section 10.5 Peripheral Access Rules for details)

11.5 Peripheral Access Rules

- For peripherals using DPPC (Dual Peripheral Protection Controller) for security monitoring:

Accessing Bus Port ID	Access Address (Corresponding Access Bus)	DPPC Peripheral Security Attribute	Validity	Peripheral Response
Secure Host (CORE0/DMA00/DMA01)	Secure Address haddr[31]=0 (AHB0/APB0)	Secure Peripheral	Legal	Responds with Secure Host Access Timing
		Non-secure Peripheral	Illegal	Blocks Secure Host Access, Generates Illegal Access Event
	Non-secure Address haddr[31]=1 (AHB1/APB1)	Secure Peripheral	Illegal	Blocks Secure Host Access, Generates Illegal Access Event
		Non-secure Peripheral	Legal	Responds with Secure Host Access Timing
Non-secure Host (CORE1/DMA10/DMA11)	Secure Address haddr[31]=0 (AHB0/APB0)	Secure Peripheral	Illegal	Bus Isolation, Enters Exception Handler
		Non-secure Peripheral	Illegal	Bus Isolation, Enters Exception Handler
	Non-secure Address haddr[31]=1 (AHB1/APB1)	Secure Peripheral	Illegal	Blocks Non-secure Host Access, Generates Illegal Access Event
		Non-secure Peripheral	Legal	Responds with Non-secure Host Access Timing

Note: RAZWI means Read As Zero, Write Ignore.

- For peripherals using SPPC (Single Peripheral Protection Controller) for security monitoring:

Accessing Bus Port ID	Access Address (Corresponding Access Bus)	SPPC Peripheral Security Attribute	Validity	Peripheral Response
Secure Host (CORE0/DMA00/DMA01)	Secure Address haddr[31]=0 (AHB0)	Secure Peripheral	Illegal	Bus Isolation, RAZWI
		Non-secure Peripheral	Illegal	Bus Isolation, RAZWI
	Non-secure Address haddr[31]=1 (AHB1)	Secure Peripheral	Legal	Responds with AHB1 Timing
		Non-secure Peripheral	Legal	Responds with AHB1 Timing
Non-secure Host (CORE1/DMA10/DMA11)	Secure Address haddr[31]=0 (AHB0)	Secure Peripheral	Illegal	Bus Isolation, Enters Exception Handler
		Non-secure Peripheral	Illegal	Bus Isolation, Enters Exception Handler
	Non-secure Address haddr[31]=1 (AHB1)	Secure Peripheral	Illegal	Blocks AHB1 Access, Generates Illegal Access Event
		Non-secure Peripheral	Legal	Responds with AHB1 Timing

- For GPIO:
  - When GPIOx[n] security attribute is configured as Secure, only the secure host is allowed to access the register group related to this GPIO pin via the non-secure address (bit[31] of bus haddr = 1).
  - When GPIOx[n] security attribute is configured as Non-Secure, only the non-secure host is allowed to access the register group related to this GPIO pin via the non-secure address (bit[31] of bus haddr = 1).

  Note: For the other two cases of mismatch between GPIO pin and host security attributes, all illegal access responses for GPIOx[n] are RAZWI. (See GPIO chapter for details)
- For SRAM, configure SCU_SRAM_SEC_WM_CFG[3:0] to divide the 160KB SRAM into secure SRAM and non-secure SRAM regions. The corresponding access rules are:
  - When SRAM_WM[3:0] = n (n≤10), the secure SRAM region is 0x1000_0000 ~ (0x1000_0000 + 16KB * n), the non-secure SRAM region is (0x9000_0000 + 16 KB * n) ~ (0x9000_0000 + 160 KB).
  - Secure host accesses secure SRAM via the SRAM secure region address (0x1000_0000 ~ (0x1000_0000 + 16KB * n)).
  - Secure host or non-secure host accesses non-secure SRAM via the SRAM non-secure region address ((0x9000_0000 + 16 KB * n) ~ (0x9000_0000 + 160 KB)).

  Note: All illegal access responses for SRAM are RAZWI.
- Illegal security access responses from CORE0 and CORE1 to other peripherals are RAZWI. Non-secure host (CORE1/DMA11) access to secure addresses will enter the exception service routine.

11.6 Reset

- Clock: Register configuration clock

  CORE0 reads/writes SCU via secure address, its clock is the CORE0 clock domain.

  Both CORE0 and CORE1 can read SCU security configuration via non-secure address, their clock is the CORE1 clock domain.

- Reset: CORE0 Reset

  All SCU registers can only be reset by CORE0, not by CORE1.

11.7 Register Description

Table 11-2 SCU Module Register Base Address

Module Name	CORE0 Address	CORE1 Address
SCU	0x3001_8000	0xB001_8000

Table 11-3 SCU Module Register Offset Addresses

Register Name	Offset Address	Default Value	Description	Read/Write	
				CORE0	CORE1
SCU_WKEY	0x00	0x0000_0000	SCU Write Password Protection Register	RW	-
SCU_LCK	0x04	0x0000_0000	SCU_LCK Control Register	RW	-
SCU_PERIPH_SEC_CFG0	0x08	0x3FFF_FFFF	Peripheral Security Configuration Register 0	RW	RO
SCU_PERIPH_SEC_CFG1	0x0C	0x0000_03FF	Peripheral Security Configuration Register 1	RW	RO
SCU_PERIPH_SEC_CFG2	0x10	0x0000_0107	Peripheral Security Configuration Register 2	RW	RO
SCU_GPIOA_SEC_CFG	0x1C	0x0000_FFFF	GPIOA Security Configuration Register	RW	RO
SCU_GPIOB_SEC_CFG	0x20	0x0000_FFFF	GPIOB Security Configuration Register	RW	RO
SCU_GPIOC_SEC_CFG	0x24	0x0000_FFFF	GPIOC Security Configuration Register	RW	RO
SCU_GPIOD_SEC_CFG	0x28	0x0000_FFFF	GPIOD Security Configuration Register	RW	RO
SCU_GPIOE_SEC_CFG	0x2C	0x0000_FFFF	GPIOE Security Configuration Register	RW	RO
SCU_GPIOF_SEC_CFG	0x30	0x0000_FFFF	GPIOF Security Configuration Register	RW	RO
SCU_GPIOG_SEC_CFG	0x34	0x0000_FFFF	GPIOG Security Configuration Register	RW	RO
SCU_GPIOH_SEC_CFG	0x38	0x0000_0003	GPIOH Security Configuration Register	RW	RO
SCU_SRAM_SEC_WM_CFG	0x5C	0x0000_000A	SRAM Security Watermark Configuration Register	RW	RO

SCU_PERIPH_SEC_IE0	0x60	0x0000_0000	SCU Peripheral Illegal Access Interrupt Enable Register 0	RW	-
SCU_PERIPH_SEC_IE1	0x64	0x0000_0000	SCU Peripheral Illegal Access Interrupt Enable Register 1	RW	-
SCU_PERIPH_SEC_IE2	0x68	0x0000_0000	SCU Peripheral Illegal Access Interrupt Enable Register 2	RW	-
SCU_PERIPH_SEC_IF0	0x6C	0x0000_0000	SCU Peripheral Illegal Access Interrupt Flag Register 0	RW1C	-
SCU_PERIPH_SEC_IF1	0x70	0x0000_0000	SCU Peripheral Illegal Access Interrupt Flag Register 1	RW1C	-
SCU_PERIPH_SEC_IF2	0x74	0x0000_0000	SCU Peripheral Illegal Access Interrupt Flag Register 2	RW1C	-

11.7.1 SCU_CFG Security Configuration Register Description

11.7.1.1 SCU_WKEY（0x00）

SCU Write Password Protection Register

Offset Address = 0x00

Bit	Name	Description	Read/Write	Reset Value
31:0	WKEY	Password protection bits for 0~15 bits, password is 0xA538_7492. =0xA538_7492: Write to SCU related registers (04H~68H) is effective; =Other values: Write to SCU related registers is invalid;	RW	0x0

11.7.1.2 SCU_LCK（0x04）

SCU_LCK Control Register

Offset Address = 0x04

Bit	Name	Description	Read/Write	Reset Value
31:0	LCK	SCU security attribute (08H~5CH) configuration lock register. =32'hF52A_B874: Locked =Other values: Not locked Note: After locking, it cannot be unlocked by writing other values, only reset unlocks.	RW	0x0

11.7.1.3 SCU_PERIPH_SEC_CFG0（0x08）

Peripheral Security Configuration Register 0

Offset Address = 0x08

Bit	Name	Description	Read/Write	Reset Value
31:30	---	Reserved	RO	0x0
29	TC1SEC	TC1 Security Attribute Configuration 0: Non-secure 1: Secure	RW	0x1
28	TC0SEC	TC0 Security Attribute Configuration 0: Non-secure 1: Secure	RW	0x1
27	DACSEC	DAC Security Attribute Configuration 0: Non-secure 1: Secure	RW	0x1
26	ADC0SEC	ADC0 Security Attribute Configuration 0: Non-secure 1: Secure	RW	0x1
25	D2FSEC	D2F Security Attribute Configuration 0: Non-secure 1: Secure	RW	0x1
24	M2MSEC	M2M Security Attribute Configuration 0: Non-secure 1: Secure	RW	0x1
23	---	Reserved	RO	0x0
22	DSPSEC	DSP Security Attribute Configuration 0: Non-secure 1: Secure	RW	0x1
21	LCDSEC	LCD Security Attribute Configuration 0: Non-secure 1: Secure	RW	0x1
20	IOCNTSEC	IOCNT Security Attribute Configuration 0: Non-secure 1: Secure	RW	0x1
19	Reserved	Reserved	RW	0x1
18	I2C1SEC	I2C1 Security Attribute Configuration 0: Non-secure 1: Secure	RW	0x1
17	I2C0SEC	I2C0 Security Attribute Configuration 0: Non-secure 1: Secure	RW	0x1

16	SPI3SEC	SPI3 Security Attribute Configuration 0: Non-secure 1: Secure	RW	0x1
15	SPI2SEC	SPI2 Security Attribute Configuration 0: Non-secure 1: Secure	RW	0x1
14	SPI1SEC	SPI1 Security Attribute Configuration 0: Non-secure 1: Secure	RW	0x1
13	SPI0SEC	SPI0 Security Attribute Configuration 0: Non-secure 1: Secure	RW	0x1
12	ISO78161SEC	ISO7816-1 Security Attribute Configuration 0: Non-secure 1: Secure	RW	0x1
11	ISO78160SEC	ISO7816-0 Security Attribute Configuration 0: Non-secure 1: Secure	RW	0x1
10	USART1SEC	USART1 Security Attribute Configuration 0: Non-secure 1: Secure	RW	0x1
9	USART0SEC	USART0 Security Attribute Configuration 0: Non-secure 1: Secure	RW	0x1
8	LPUARTSEC	LPUART Security Attribute Configuration 0: Non-secure 1: Secure	RW	0x1
7	UART7SEC	UART7 Security Attribute Configuration 0: Non-secure 1: Secure	RW	0x1
6	UART6SEC	UART6 Security Attribute Configuration 0: Non-secure 1: Secure	RW	0x1
5	UART5SEC	UART5 Security Attribute Configuration 0: Non-secure 1: Secure	RW	0x1
4	UART4SEC	UART4 Security Attribute Configuration 0: Non-secure 1: Secure	RW	0x1
3	UART3SEC	UART3 Security Attribute Configuration 0: Non-secure 1: Secure	RW	0x1
2	UART2SEC	UART2 Security Attribute Configuration	RW	0x1

		0: Non-secure 1: Secure		
1	UART1SEC	UART1 Security Attribute Configuration 0: Non-secure 1: Secure	RW	0x1
0	UART0SEC	UART0 Security Attribute Configuration 0: Non-secure 1: Secure	RW	0x1

11.7.1.4 SCU_PERIPH_SEC_CFG1（0x0C）

Peripheral Security Configuration Register 1

Offset Address = 0x0C

Bit	Name	Description	Read/Write	Reset Value
31:11	---	Reserved	RO	0x0
10	CRC0SEC	CRC0 Security Attribute Configuration 0: Non-secure 1: Secure	RW	0x1
9	LVD_CMPSEC	LVD, CMP Security Attribute Configuration 0: Non-secure 1: Secure *Note: LVD, CMP1 and CMP2 share the LC_VDD security attribute configuration.*	RW	0x1
8	KBI1SEC	KBI1 Security Attribute Configuration 0: Non-secure 1: Secure	RW	0x1
7	KBI0SEC	KBI0 Security Attribute Configuration 0: Non-secure 1: Secure	RW	0x1
6	LSRXSEC	LSRX Security Attribute Configuration 0: Non-secure 1: Secure	RW	0x1
5	---	Reserved	RO	0x0
4	ANASEC	ANA Security Attribute Configuration 0: Non-secure 1: Secure *Note: RTC, SARADC and GPIOL[3:0] share the ANASEC security attribute configuration.*	RW	0x1
3	SIMPTC3SEC	SIMPTC3 Security Attribute Configuration 0: Non-secure 1: Secure	RW	0x1
2	SIMPTC2SEC	SIMPTC2 Security Attribute Configuration 0: Non-secure 1: Secure	RW	0x1

1	SIMPTC1SEC	SIMPTC1 Security Attribute Configuration 0: Non-secure 1: Secure	RW	0x1
0	SIMPTC0SEC	SIMPTC0 Security Attribute Configuration 0: Non-secure 1: Secure	RW	0x1

11.7.1.5 SCU_PERIPH_SEC_CFG2（0x10）

Peripheral Security Configuration Register 2

Offset Address = 0x10

Bit	Name	Description	Read/Write	Reset Value
31:9	---	Reserved	RO	0x0
8	AESSEC	AES Security Attribute Configuration 0: Non-secure 1: Secure	RW	0x1
7:3	---	Reserved	RO	0x0
2	SEASEC	SEA Security Attribute Configuration 0: Non-secure 1: Secure *Note: SEA, KEY_STORE share the SEA security attribute configuration.*	RW	0x1
1	QSPI1SEC	QSPI1 Security Attribute Configuration 0: Non-secure 1: Secure	RW	0x1
0	QSPI0SEC	QSPI0 Security Attribute Configuration 0: Non-secure 1: Secure	RW	0x1

11.7.1.6 SCU_GPIOx_SEC_CFG(x = A~G) （0x1C~0x34）

GPIOx Security Configuration Register

Offset Address = 0x1C~0x34

Bit	Name	Description	Read/ Write	Reset Value
31:16	---	Reserved	RO	0x0
15:0	Px_SEC	Px00~Px15 Security Attribute Configuration 0: Non-secure 1: Secure *Note: PD04 is the LCDVD pin, has no GPIO function, the PD_SEC[4] security attribute configuration is invalid, the read/write attribute of PD_SEC[4] is retained.*	RW	0xffff

11.7.1.7 SCU_GPIOH_SEC_CFG （0x38）

GPIOH Security Configuration Register

Offset Address = 0x38

Bit	Name	Description	Read/ Write	Reset Value
31:2	---	Reserved	RO	0x0
1:0	PH_SEC	PH00~PH01 Security Attribute Configuration 0: Non-secure 1: Secure	RW	0x3

11.7.1.8 SCU_SRAM_SEC_WM_CFG（0x5C）

SRAM Security Watermark Configuration Register

Offset Address = 0x5C

Bit	Name	Description	Read/ Write	Reset Value
31:4	---	Reserved	RO	0x0
3:0	SRAM_SEC_LEN	Secure SRAM Region Length Configuration =1~4'ha: Secure SRAM region length is SRAM_SEC_LEN*16KB, Non-secure SRAM region length is (10-SRAM_SEC_LEN) * 16KB. =0 or 4'hb~4'hf: This configuration is invalid.	RW	0xa

11.7.2 SCU_IC Interrupt Control Register Description

11.7.2.1 SCU_PERIPH_SEC_IE0（0x60）

SCU Peripheral Illegal Access Interrupt Enable Register 1

Offset Address = 0x60

Bit	Name	Description	Read/Write	Reset Value
31:30	---	Reserved	RO	0x0
29	TC1IE	TC1 Illegal Access Interrupt Enable Configuration 0: Disable 1: Enable	RW	0x0
28	TC0IE	TC0 Illegal Access Interrupt Enable Configuration 0: Disable 1: Enable	RW	0x0
27	DACIE	DAC Illegal Access Interrupt Enable Configuration 0: Disable 1: Enable	RW	0x0
26	ADC0IE	ADC0 Illegal Access Interrupt Enable Configuration 0: Disable 1: Enable	RW	0x0
25	D2FIE	D2F Illegal Access Interrupt Enable Configuration	RW	0x0

		0: Disable 1: Enable		
24	M2MIE	M2M Illegal Access Interrupt Enable Configuration 0: Disable 1: Enable	RW	0x0
23	---	Reserved	RO	0x0
22	DSPIE	DSP Illegal Access Interrupt Enable Configuration 0: Disable 1: Enable	RW	0x0
21	LCDIE	LCD Illegal Access Interrupt Enable Configuration 0: Disable 1: Enable	RW	0x0
20	IOCNTIE	IOCNT Illegal Access Interrupt Enable Configuration 0: Disable 1: Enable	RW	0x0
19	Reserved	Reserved	RW	0x0
18	I2C1IE	I2C1 Illegal Access Interrupt Enable Configuration 0: Disable 1: Enable	RW	0x0
17	I2C0IE	I2C0 Illegal Access Interrupt Enable Configuration 0: Disable 1: Enable	RW	0x0
16	SPI3IE	SPI3 Illegal Access Interrupt Enable Configuration 0: Disable 1: Enable	RW	0x0
15	SPI2IE	SPI2 Illegal Access Interrupt Enable Configuration 0: Disable 1: Enable	RW	0x0
14	SPI1IE	SPI1 Illegal Access Interrupt Enable Configuration 0: Disable 1: Enable	RW	0x0
13	SPI0IE	SPI0 Illegal Access Interrupt Enable Configuration 0: Disable 1: Enable	RW	0x0
12	ISO78161IE	ISO7816_1 Illegal Access Interrupt Enable Configuration 0: Disable 1: Enable	RW	0x0
11	ISO78160IE	ISO7816_0 Illegal Access Interrupt Enable Configuration 0: Disable 1: Enable	RW	0x0
10	USART1IE	USART1 Illegal Access Interrupt Enable Configuration 0: Disable 1: Enable	RW	0x0

9	USART0IE	USART0 Illegal Access Interrupt Enable Configuration 0: Disable 1: Enable	RW	0x0
8	LPUARTIE	LPUART Illegal Access Interrupt Enable Configuration 0: Disable 1: Enable	RW	0x0
7	UART7IE	UART7 Illegal Access Interrupt Enable Configuration 0: Disable 1: Enable	RW	0x0
6	UART6IE	UART6 Illegal Access Interrupt Enable Configuration 0: Disable 1: Enable	RW	0x0
5	UART5IE	UART5 Illegal Access Interrupt Enable Configuration 0: Disable 1: Enable	RW	0x0
4	UART4IE	UART4 Illegal Access Interrupt Enable Configuration 0: Disable 1: Enable	RW	0x0
3	UART3IE	UART3 Illegal Access Interrupt Enable Configuration 0: Disable 1: Enable	RW	0x0
2	UART2IE	UART2 Illegal Access Interrupt Enable Configuration 0: Disable 1: Enable	RW	0x0
1	UART1IE	UART1 Illegal Access Interrupt Enable Configuration 0: Disable 1: Enable	RW	0x0
0	UART0IE	UART0 Illegal Access Interrupt Enable Configuration 0: Disable 1: Enable	RW	0x0

11.7.2.2 SCU_PERIPH_SEC_IE1（0x64）

SCU Peripheral Illegal Access Interrupt Enable Register 2

Offset Address = 0x10

Bit	Name	Description	Read/Write	Reset Value
31:11	---	Reserved	RO	0x0
10	CRC0IE	CRC0 Illegal Access Interrupt Enable Configuration 0: Disable 1: Enable	RW	0x0
9	LVD_CMPIE	LVD_CMP Illegal Access Interrupt Enable Configuration 0: Disable 1: Enable	RW	0x0

8	KBI1IE	KBI1 Illegal Access Interrupt Enable Configuration 0: Disable 1: Enable	RW	0x0
7	KBI0IE	KBI0 Illegal Access Interrupt Enable Configuration 0: Disable 1: Enable	RW	0x0
6	LSRXIE	LSRX Illegal Access Interrupt Enable Configuration 0: Disable 1: Enable	RW	0x0
5	---	Reserved	RO	0x0
4	ANAIE	ANA Illegal Access Interrupt Enable Configuration 0: Disable 1: Enable *Note: RTC, SARADC and GPIOL[3:0] share the ANASEC security attribute configuration.*	RW	0x0
3	SIMPTC3IE	SIMPTC3 Illegal Access Interrupt Enable Configuration 0: Disable 1: Enable	RW	0x0
2	SIMPTC2IE	SIMPTC2 Illegal Access Interrupt Enable Configuration 0: Disable 1: Enable	RW	0x0
1	SIMPTC1IE	SIMPTC1 Illegal Access Interrupt Enable Configuration 0: Disable 1: Enable	RW	0x0
0	SIMPTC0IE	SIMPTC0 Illegal Access Interrupt Enable Configuration 0: Disable 1: Enable	RW	0x0

11.7.2.3 SCU_PERIPH_SEC_IE2（0x68）

SCU Peripheral Illegal Access Interrupt Enable Register 3

Offset Address = 0x68

Bit	Name	Description	Read/Write	Reset Value
31:9	---	Reserved	RO	0x0
8	AESIE	AES Illegal Access Interrupt Enable Configuration 0: Disable 1: Enable	RW	0x0
7:3	---	Reserved	RO	0x0
2	SEAIE	SEA Illegal Access Interrupt Enable Configuration 0: Disable 1: Enable	RW	0x0
1	QSPI1IE	QSPI1 Illegal Access Interrupt Enable Configuration 0: Disable	RW	0x0

		1: Enable		
0	QSPI0IE	QSPI0 Illegal Access Interrupt Enable Configuration 0: Disable 1: Enable	RW	0x0

11.7.2.4 SCU_PERIPH_SEC_ IF0（0x6C）

SCU Peripheral Illegal Access Interrupt Flag Register 1

Offset Address = 0x6C

Bit	Name	Description	Read/Write	Reset Value
31:30	---	Reserved	RO	0x0
29	TC1IF	TC1 Illegal Access Interrupt Flag Bit 0: Legal Access 1: Illegal Access	RW1C	0x0
28	TC0IF	TC0 Illegal Access Interrupt Flag Bit 0: Legal Access 1: Illegal Access	RW1C	0x0
27	DACIF	DAC Illegal Access Interrupt Flag Bit 0: Legal Access 1: Illegal Access	RW1C	0x0
26	ADC0IF	ADC0 Illegal Access Interrupt Flag Bit 0: Legal Access 1: Illegal Access	RW1C	0x0
25	D2FIF	D2F Illegal Access Interrupt Flag Bit 0: Legal Access 1: Illegal Access	RW1C	0x0
24	M2MIF	M2M Illegal Access Interrupt Flag Bit 0: Legal Access 1: Illegal Access	RW1C	0x0
23	---	Reserved	RO	0x0
22	DSPIF	DSP Illegal Access Interrupt Flag Bit 0: Legal Access 1: Illegal Access	RW1C	0x0
21	LCDIF	LCD Illegal Access Interrupt Flag Bit 0: Legal Access 1: Illegal Access	RW1C	0x0
20	IOCNTIF	IOCNT Illegal Access Interrupt Flag Bit 0: Legal Access 1: Illegal Access	RW1C	0x0
19	Reserved	Reserved	RW1C	0x0
18	I2C1IF	I2C1 Illegal Access Interrupt Flag Bit 0: Legal Access	RW1C	0x0

		1: Illegal Access		
17	I2C0IF	I2C0 Illegal Access Interrupt Flag Bit 0: Legal Access 1: Illegal Access	RW1C	0x0
16	SPI3IF	SPI3 Illegal Access Interrupt Flag Bit 0: Legal Access 1: Illegal Access	RW1C	0x0
15	SPI2IF	SPI2 Illegal Access Interrupt Flag Bit 0: Legal Access 1: Illegal Access	RW1C	0x0
14	SPI1IF	SPI1 Illegal Access Interrupt Flag Bit 0: Legal Access 1: Illegal Access	RW1C	0x0
13	SPI0IF	SPI0 Illegal Access Interrupt Flag Bit 0: Legal Access 1: Illegal Access	RW1C	0x0
12	ISO78161IF	ISO7816_1 Illegal Access Interrupt Flag Bit 0: Legal Access 1: Illegal Access	RW1C	0x0
11	ISO78160IF	ISO7816_0 Illegal Access Interrupt Flag Bit 0: Legal Access 1: Illegal Access	RW1C	0x0
10	USART1IF	USART1 Illegal Access Interrupt Flag Bit 0: Legal Access 1: Illegal Access	RW1C	0x0
9	USART0IF	USART0 Illegal Access Interrupt Flag Bit 0: Legal Access 1: Illegal Access	RW1C	0x0
8	LPUARTIF	LPUART Illegal Access Interrupt Flag Bit 0: Legal Access 1: Illegal Access	RW1C	0x0
7	UART7IF	UART7 Illegal Access Interrupt Flag Bit 0: Legal Access 1: Illegal Access	RW1C	0x0
6	UART6IF	UART6 Illegal Access Interrupt Flag Bit 0: Legal Access 1: Illegal Access	RW1C	0x0
5	UART5IF	UART5 Illegal Access Interrupt Flag Bit 0: Legal Access 1: Illegal Access	RW1C	0x0
4	UART4IF	UART4 Illegal Access Interrupt Flag Bit 0: Legal Access	RW1C	0x0

		1: Illegal Access		
3	UART3IF	UART3 Illegal Access Interrupt Flag Bit 0: Legal Access 1: Illegal Access	RW1C	0x0
2	UART2IF	UART2 Illegal Access Interrupt Flag Bit 0: Legal Access 1: Illegal Access	RW1C	0x0
1	UART1IF	UART1 Illegal Access Interrupt Flag Bit 0: Legal Access 1: Illegal Access	RW1C	0x0
0	UART0IF	UART0 Illegal Access Interrupt Flag Bit 0: Legal Access 1: Illegal Access	RW1C	0x0

11.7.2.5 SCU_PERIPH_SEC_ IF1（0x70）

SCU Peripheral Illegal Access Interrupt Flag Register 2

Offset Address = 0x70

Bit	Name	Description	Read/Write	Reset Value
31:11	---	Reserved	RO	0x0
10	CRC0IF	CRC0 Illegal Access Interrupt Flag Bit 0: Legal Access 1: Illegal Access	RW1C	0x0
9	LVD_CMPIF	LVD_CMP Illegal Access Interrupt Flag Bit 0: Legal Access 1: Illegal Access	RW1C	0x0
8	KBI1IF	KBI1 Illegal Access Interrupt Flag Bit 0: Legal Access 1: Illegal Access	RW1C	0x0
7	KBI0IF	KBI0 Illegal Access Interrupt Flag Bit 0: Legal Access 1: Illegal Access	RW1C	0x0
6	LSRXIF	LSRX Illegal Access Interrupt Flag Bit 0: Legal Access 1: Illegal Access	RW1C	0x0
5	---	Reserved	RO	0x0
4	ANAIF	ANA Illegal Access Interrupt Flag Bit 0: Legal Access 1: Illegal Access	RW1C	0x0
3	SIMPTC3IF	SIMPTC3 Illegal Access Interrupt Flag Bit 0: Legal Access 1: Illegal Access	RW1C	0x0

2	SIMPTC2IF	SIMPTC2 Illegal Access Interrupt Flag Bit 0: Legal Access 1: Illegal Access	RW1C	0x0
1	SIMPTC1IF	SIMPTC1 Illegal Access Interrupt Flag Bit 0: Legal Access 1: Illegal Access	RW1C	0x0
0	SIMPTC0IF	SIMPTC0 Illegal Access Interrupt Flag Bit 0: Legal Access 1: Illegal Access	RW1C	0x0

11.7.2.6 SCU_PERIPH_SEC_IF2（0x74）

SCU Peripheral Illegal Access Interrupt Flag Register 3

Offset Address = 0x74

Bit	Name	Description	Read/Write	Reset Value
31:9	---	Reserved	RO	0x0
8	AESIF	AES Illegal Access Interrupt Flag Bit 0: Legal Access 1: Illegal Access	RW1C	0x0
7:3	---	Reserved	RO	0x0
2	SEAIF	SEA Illegal Access Interrupt Flag Bit 0: Legal Access 1: Illegal Access	RW1C	0x0
1	QSPI1IF	QSPI1 Illegal Access Interrupt Flag Bit 0: Legal Access 1: Illegal Access	RW1C	0x0
0	QSPI0IF	QSPI0 Illegal Access Interrupt Flag Bit 0: Legal Access 1: Illegal Access	RW1C	0x0

11.8 Software Usage Flow

The following software flow is performed on CORE0.

- After power-up, all peripherals with dynamically assignable security attributes are secure peripherals.
- Enable the SCU clock gating in the system register chapter.
- Configure SCU_WKEY[31:0] = 0xA538_7492.
- Configure SCU_PERIPH_SEC_CFG0~2 to assign security attributes to peripherals with PPC.
- Configure SCU_GPIOx_SEC_CFG (x=A~P) to assign security attributes to GPIO pins.
- Configure SCU_SRAM_SEC_WM_CFG to divide the 160 KB SRAM into secure SRAM (0x1000_0000 ~ (0x1000_0000 + 16KB * n)) and non-secure SRAM ((0x9000_0000 + 16 KB * n) ~ (0x9000_0000 + 160 KB)). To use CORE1, first allocate SRAM space for CORE1.
- Configure SCU_LCK[31:0] = 0xF52A_B874 to lock the security attribute configuration for peripherals with PPC, GPIO, and SRAM. (Optional)

- Enable the corresponding interrupts for peripherals with PPC.

12 FLASH

12.1 Introduction

The chip has a built-in 1MB FLASH for storing Core0/Core1 programs and data. In addition to the 1MB main storage area, the FLASH also provides an NVR storage area for storing configuration information. The NVR storage area is not affected by erasure and programming of the main storage area.

12.2 Features

- Minimum 100,000 erase/write cycles
- Minimum data retention time of 100 years @25°C
- Main storage area contains 2048 sectors, each sector size is 512 Bytes
- Minimum erase unit is one sector; sector erase time is approximately 3ms
- Supports configurable storage areas for dual cores
- Dual-core FLASH internal storage address remapping
- Supports dual-core configuration for erase/write protection of the main storage area
- Supports Core0 NVR area read/write protection configuration
- Supports auto standby function
- Supports KeyStore module directly accessing FLASH
- Supports Lock mechanism for managing NVR area read/write permissions

12.3 Clock

The Core1 system maximum operating clock is 58.9824MHz, the Core0 system maximum operating clock is 29.4912MHz, and the FLASH operating clock is the clock before dividing the Core0/Core1 system clock, with a maximum of 58.9824MHz.

12.4 Functional Description

12.4.1 Storage Area Partitioning

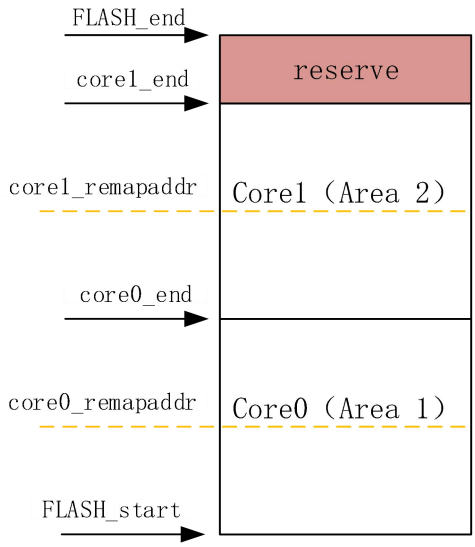


Figure 12-1 FLASH Internal Storage Allocation

The internal FLASH storage allocation is as shown above. The Core0 configuration register FLASH_ALLOCATE1 defines the FLASH address ranges accessible by the two cores (i.e., configuring core0_end and core1_end in the figure), with a granularity of 8KB. Core0 and Core1 can each define their own area's remap address (remap_addr). The read/write permissions for each area are as follows:

- Area 1: Core0 can read and write, Core1 cannot read or write (0 =< core0_addr <= core0_end);
- Area 2: Core1 can read and write, Core0 can read but not write (core1_end < core1_addr <= core1_end);

Note 1: The specific allocation within each area, such as boot area and app area for Area 1 and Area 2, is defined by software, and the hardware configures erase/write protection permissions.

Note 2: When core0_end = core1_end, it means Core1 has no readable/writable space.

Note 3: When core1_end is not configured to 0x3f, the FLASH has a reserved space, which is inaccessible to both Core0 and Core1 (access returns 0x0). This is a feature to reduce the FLASH storage space.

Note 4: When reading an illegal address range, the bus feedback data is all zeros.

12.5 Register Definition

All FLASH operations are encapsulated in standard library functions. Please refer to the library function usage instructions.

13 Key Store

13.1 Introduction

The Key Store module is a key storage module. The CPU can manage keys in the FLASH NVR area through this module. The SEA encryption/decryption module can interact with this module, for example, to generate or extract keys for operations. If a key becomes invalid or is tampered with, the CPU can erase it via this module. Furthermore, the Key Store module ensures key security through methods such as encrypted storage and integrity checks.

13.2 Features

- Supports Core0 configuration for key generation.
- Supports storing keys with a minimum length of 128 bits and a maximum length of 576 bits.
- Supports storing keys in the FLASH area or the OTP area.
- Supports storing up to 80 FLASH keys and 8 OTP keys.
- Supports the SEA module storing and retrieving keys through this module.
- Supports key erasure (FLASH).
- Supports encrypted key storage.
- Supports key integrity checks and key parameter verification.
- Supports key block queries.

13.3 Basic Principles

13.3.1 Storage Devices

The storage devices for keys in this module can be divided into FLASH and OTP types. The OTP is constructed from FLASH_NVR with monitoring circuitry. The FLASH_NVR area occupied by this module is accessible only by this module; this block is internally partitioned within the FLASH_NVR area by hardware circuitry. The figure below is a schematic diagram of the dedicated area allocation for this module within FLASH_NVR.

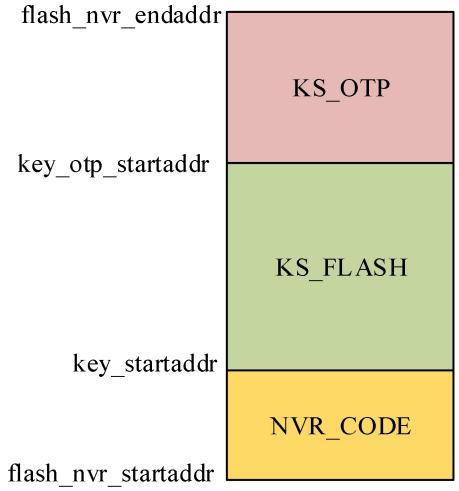


Figure13-1 Schematic of the module's dedicated area partition in FLASH_NVR

As shown in the figure, keys stored at addresses between key_startaddr and key_otp_startaddr reside in KS_FLASH, while keys stored at addresses between key_otp_startaddr and flash_nvr_endaddr reside in KS_OTP.

When a key is stored in the KS_OTP area, this module restricts it from being erased.

13.3.2 Storage Format

This module stores keys supporting a maximum length of 576 bits and a minimum length of 128 bits. It supports encrypted storage, erasure, and being invisible to the CPU.

This module allows key writing and reading via the SEA module, and there is no mandatory requirement for the key write order (i.e., endianness). The write order and read order are the same and are strongly related to the actual application.

When an erase request is initiated by the module, it will erase all keys in KS_FLASH. Use this function with caution, as keys in KS_OTP are not affected.

13.3.3 Key Blocks

This module contains 88 key blocks: 80 key blocks stored in KS_FLASH (key block indices 0-79), and 8 key blocks stored in KS_OTP (key block indices 80-87). For keys stored in KS_OTP, the module writes three copies into 3 SECTORs to prevent storage errors.

13.3.4 Attribute Parameters

The attribute parameters for a key block are as follows:

- Readable:
  - 0: Invisible to the CPU, meaning the key cannot be read from the block.
  - 1: Visible to the CPU, allowing only Core0 to read the key from the block.
- Encrypted Storage:
  - 0: Stored in plain text within the block.
  - 1: Stored as cipher text within the block; XOR encryption or AES_ECB 128-bit encryption can be selected.
- Programmed:
  - 0: The key block is not programmed, meaning a key can be written.
  - 1: The key block is programmed, meaning key writing fails, but key reading is possible.
- Key Length: Only the following key lengths are supported.

Table13-1 Supported Key Length List

Key Length	Key Length (bits)
8'b0	128
8'b1	160
8'b10	192
8'b11	224
8'b100	256
8'b101	384
8'b110	512
8'b111	521

8’b1000	576
Else	Reserved

- Encryption Key Index: When selecting AES_ECB encrypted storage for the key, a key index must be specified (generally selecting a root key stored in the KS_OTP area).

Attribute parameters are verified every time a key is read.

13.4 Register Definition

All Key Store operations are encapsulated in standard library functions. Please refer to the library function usage instructions.

14 M2M

14.1 Introduction

The Mem2Mem module uses a DMA method to automatically transfer data within the SRAM without CPU involvement, moving data from a specified source address to a destination address. The configurable minimum data unit for transfer is 1 Byte. It also supports configuration of valid bits for source and destination data. This module is only applicable to systems with an SRAM width of 32 bits.

14.2 Features

- Configurable source and destination addresses; addresses can be Byte-aligned.
- Configurable source data length, minimum unit is Byte.
- Configurable input data valid bits.
- Configurable output data valid bits.
- Configurable output data stored in reverse Byte order at the destination address.
- Configurable output data stored in reverse address order at the destination address.
- Generates a flag upon data transfer completion; can be configured to generate an interrupt.

14.2.1 Input/Output Data Valid Bit Width Configuration

Input/output data valid bit width configuration refers to how many Bytes are valid within every 4 Bytes of data.

Input and output data can be configured via the registers ivld[3:0] and ovld[3:0] respectively. For every 4 Bytes of data, 1 to 4 Bytes can be configured as valid.

An invalid input data Byte means that the corresponding Byte in every 32-bit data from the source will be discarded during the data transfer process.

An invalid output data Byte means that in the destination address's SRAM, the corresponding Byte of every 32-bit word will not store valid data.

- When both input and output data valid bit widths are configured to 4 Bytes, direct data transfer within the SRAM can be achieved.
- When the input data valid bit width is less than 4 Bytes, invalid data in the input source can be discarded.
- When the output data valid bit width is less than 4 Bytes, the function of inserting dummy data into the target data can be achieved.

  See the example descriptions for specific functions.

14.2.2 Configurable Dummy Value for Invalid Output Addresses

When the bit corresponding to the output address is invalid, the configured register value is filled in. By default, no write operation is performed on invalid addresses.

Note that the register configuration here is 32 bits, corresponding to each bit of the 32-bit SRAM.

14.2.3 Output Data Reverse Placement

Setting the register bit orv_en=1 enables output data reversal. The reversal result depends on the output data width. For example, if the output data width is 3 Bytes, the 3 Bytes of data (A0, A1, A2) fetched from the source data are reversed (A2, A1, A0) and then stored in the SRAM.

14.2.4 Configurable Source and Destination Addresses, Configurable Input Data Length

The source data address and destination data address are configured via registers, supporting Byte alignment. The input data length is configurable via registers, with the minimum unit being Byte. The output data length is determined by the input/output data length and the input/output data valid bits. Software can calculate the output data length based on the configuration; this function is not implemented in hardware.

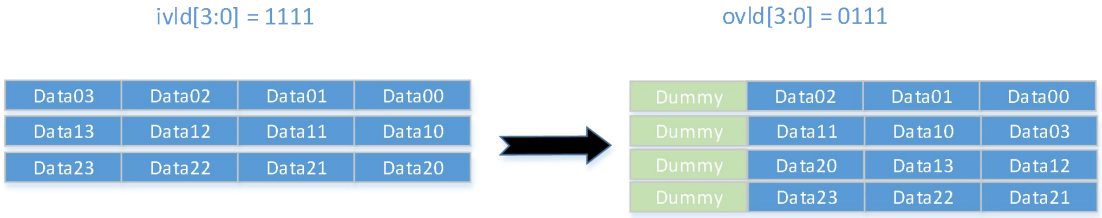
14.2.5 Data Arranged in Reverse Address Order

Setting the register bit addr_rven=1 enables output data to be placed in reverse address order. This address reversal function only supports cases where both input and output data lengths are 4 bytes and are word-aligned. Each time 4 Bytes (A3, A2, A1, A0) are fetched from the source, (A3, A2, A1, A0) is written to the destination address, then the destination address is decremented by 4 to store the next 4 Bytes fetched from the source. This continues until the last 4 bytes are stored.

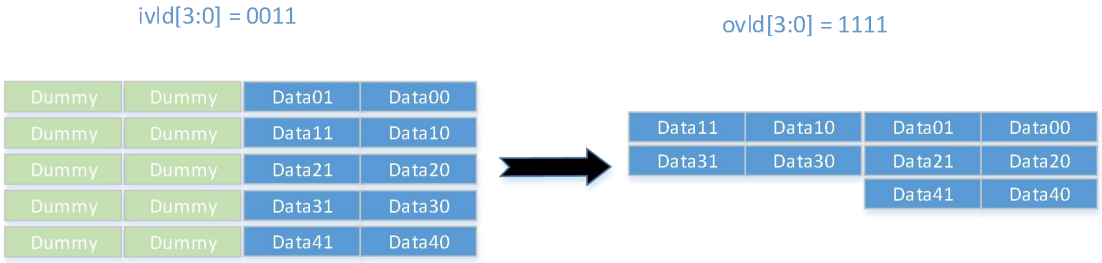
When using address reverse placement, it must be ensured that ivld=4'b1111, ovld=4'b1111, saddr[1:0]=0, daddr[1:0]=0; ilen=4*n (n is an integer greater than 0).

14.3 Example Descriptions

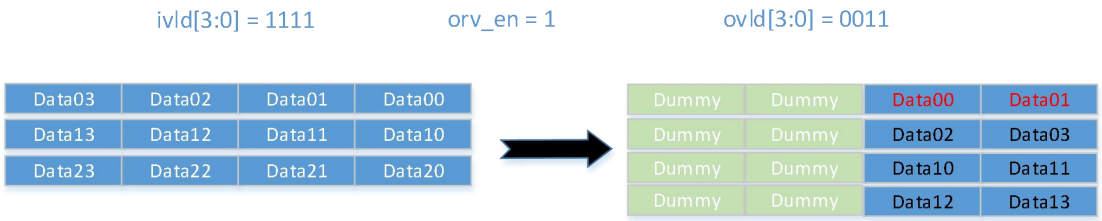
14.3.1 Input 4 Bytes Valid, Output 3 Bytes Valid



14.3.2 Input 2 Bytes Valid, Output 4 Bytes Valid



14.3.3 Input Data Stored in Reverse Byte Order (Input 4 Bytes Valid, Output 2 Bytes Valid)



From the figure above, it can be seen that the output address is in reverse order compared to the input address; data00 from the source address is placed at the lowest address of a Word, while in the destination address, data00 is placed at the highest address of a Word.

14.3.4 Data Arranged in Reverse Address Order

From the figure above, it can be seen that the first 4 Bytes of source data (data03, data02, data01, data00) are written to the destination address daddrn, then the destination address is decremented by 4 to store the next 4 Bytes fetched from the source. This continues until the last byte is stored.

14.4 Transfer Speed Evaluation

It can achieve a maximum speed of 6 cycles to complete one 32-bit data transfer. The actual speed also depends on the configured ivld and ovld, as well as SRAM contention.

14.5 Design Detail Handling

For source data length ilen, source data valid bit width ivld, and destination data valid bit width ovld, situations may arise where ilen * ivld / ovld is not an integer; this is an unconventional application. During implementation, the last transfer is handled in two cases:

- If DUMMY_EN=0, valid bytes are written with corresponding valid data, and extra bytes are not written.
- If DUMMY_EN=1, valid bytes are written with corresponding valid data, and extra bytes are written with the dummy value.

14.6 Register Description

M2M Register Base Address

Module Name	Core0 Base Address	Core1 Base Address
M2M	0x20078000	0xA0078000

Register List

Offset Address	Name	RW	Valid Word Length	Reset Value	Function Description
0x00	M2M_MODE	RW	4	0xFF	M2M Mode Configuration Register
0x04	M2M_CTL	RW	4	0x0	M2M Control Register
0x08	M2M_DUMMY	RW	4	0x0	M2M Invalid Address DUMMY Value Configuration Register
0x0C	M2M_SADDR	RW	4	0x0	M2M Source Data Address Configuration Register
0x10	M2M_DADDR	RW	4	0x0	M2M Destination Data Address Configuration Register
0x14	M2M_ILEN	RW	4	0x0	M2M Source Data Length Configuration

					Register
0x18	M2M_IE	RW	4	0x0	M2M Interrupt Enable Register
0x1C	M2M_IF	RW	4	0x0	M2M Interrupt Flag Register

The above registers do not support bit-band operation.

14.6.1 M2M_MODE

M2M Mode Configuration Register Address: 0x0；

Bit	Bit Name	Function Description	Read/Write	Reset Value
31:11	--	Reserved	R	0
10	ADDR_RVEN	Data arranged in reverse address order. =0: No reversal =1: Arrange in reverse address order	RW	0
9	DUMMY_EN	=0: For invalid Byte positions in the destination address, no write operation is performed. =1: For invalid Byte positions in the destination address, write the dummy value.	RW	0
8	ORV_EN	Output data placed in reverse Byte order. =0: No reversal =1: Reverse order	RW	0
7:4	OVLD	Output data valid Byte configuration. =0: Byte invalid =1: Byte valid A bit set to 0 indicates that the corresponding Byte address in the destination address is invalid, or the dummy value is written if DUMMY_EN is effective.	RW	0xF
3:0	IVLD	Input data valid Byte configuration. =0: Byte invalid =1: Byte valid Each configured bit sequentially corresponds to each Byte in the 32-bit data; 1 indicates the corresponding Byte is valid.	RW	0xF

14.6.2 M2M_CTL

M2M Control Register Address: 0x4；

Bit	Bit Name	Function Description	Read/Write	Reset Value
31:1	--	Reserved	R	0
0	M2M_EN	=0: No action =1: Start data transfer. Automatically clears after the specified length data transfer is complete. When this bit is 1, writing 0 can terminate the current transfer process.	RW	0

14.6.3 M2M_DUMMY

M2M Invalid Address DUMMY Value Configuration Register Address: 0x8；

Bit	Bit Name	Function Description	Read/Write	Reset Value
31:0	DUMMY	Effective when M2M_MODE.DUMMY_EN=1. This 32-bit value corresponds Byte-wise to the Bytes of the invalid addresses configured by M2M_MODE.OVLD.	RW	0

14.6.4 M2M_SADDR

M2M Source Data Address Configuration Register Address: 0xC；

Bit	Bit Name	Function Description	Read/Write	Reset Value
31:18	--	Reserved	R	0
17:0	SADDR	Source data address. When using address reverse order, must be word-aligned; otherwise Byte-aligned. This address is the SRAM offset address. Source data address + source data length must not exceed the SRAM address range.	RW	0

14.6.5 M2M_DADDR

M2M Destination Data Address Configuration Register Address: 0x10；

Bit	Bit Name	Function Description	Read/Write	Reset Value
31:18	--	Reserved	R	0
17:0	DADDR	Destination data address. When using address reverse order, must be word-aligned; otherwise Byte-aligned. This address is the SRAM offset address. Destination data address + destination data length must not exceed the SRAM address range.	RW	0

14.6.6 M2M_ILEN

M2M Source Data Length Configuration Register Address: 0x14；

Bit	Bit Name	Function Description	Read/Write	Reset Value
31:18	--	Reserved	R	0
17:0	ILEN	Input data length configuration, Byte-aligned. Source data length must not exceed the SRAM size. Note: Configuring ILEN=0 and M2M_EN=1 is not allowed. If done, the DONE flag of M2M will remain	RW	0

		0, and M2M_EN will also remain 1. If ILEN=0 and M2M_EN=1 are configured, M2M_EN must be written to 0 before the next transfer.		

14.6.7 M2M_IE

M2M Interrupt Enable Register Address: 0x18；

Bit	Bit Name	Function Description	Read/Write	Reset Value
31:1	--	Reserved	R	0
0	DONE_IE	Transfer completion interrupt enable. =0: Disable =1: Enable interrupt	RW	0

14.6.8 M2M_IF

M2M Interrupt Flag Register Address: 0x1C；

Bit	Bit Name	Function Description	Read/Write	Reset Value
31:1	--	Reserved	R	0
0	DONE	Transfer completion flag, write 1 to clear. =0: Transfer not complete =1: Transfer complete This bit is automatically cleared each time M2M_CTL.START=1 is configured.	RW	0

14.7 Software Operation Flow

- Configure the source data start address M2M_SADDR.
- Configure the destination data start address M2M_DADDR.
- Configure the source data length M2M_ILEN.
- Configure the mode register M2M_MODE.
- Configure the interrupt enable M2M_IE.
- Configure and start M2M_CTL.
- Wait for the interrupt, read data from the destination address, and clear the flag.

15 AOS

15.1 Introduction

The Automatic Operation System (AOS) enables linkage between different peripherals without CPU intervention. Hardware prohibits non-secure peripherals from triggering secure peripherals, and secure peripherals from triggering non-secure peripherals. The trigger signals provided by peripherals are called Trigger Sources (e.g., timer compare match signal, timer overflow signal, RTC periodic signal, etc.), used to trigger actions in other peripheral circuits. The triggered peripheral actions are called Trigger Targets (e.g., ADC0 start conversion, DAC start conversion, etc.).

15.2 Features

- Linkage between different peripherals without CPU intervention.
- Hardware prohibits secure peripherals from triggering non-secure peripherals.
- Hardware prohibits non-secure peripherals from triggering secure peripherals.

15.3 Functional Description

15.3.1 Trigger Source Selection Mechanism

Each trigger target peripheral circuit action (hereinafter referred to as the target peripheral) has a trigger source selection register PE_TRGSELx. This register has a 6-bit field TRGSEL, which can select a trigger source event number. When TRGSEL=0, it means no trigger source is selected, effectively disabling the linkage for that target peripheral. When TRGSEL is set to a defined trigger source number, that trigger source can trigger the target peripheral.

The AOS module also has two enable bits: a secure peripheral enable bit and a non-secure peripheral enable bit. Only when the corresponding enable bit is set, the 6-bit TRGSEL configuration for the target peripheral is valid; otherwise, the target peripheral cannot receive trigger signals.

15.3.2 Trigger Targets

Table 15-1 AOS Trigger Target List

Target Number	Target Peripheral	Purpose
0	TC0	External Input Clock Source
1	TC0	Slave Mode External Input Event
2	TC0	Channel 0 Capture External Input Event
3	TC0	Channel 1 Capture External Input Event
4	TC1	External Input Clock Source
5	TC1	Slave Mode External Input Event
6	TC1	Channel 0 Capture External Input Event
7	TC1	Channel 1 Capture External Input Event
8	SIMPTC0	Hardware Trigger Channel 0 Start Count
9	SIMPTC0	Hardware Trigger Channel 1 Start Count
10	SIMPTC0	Hardware Trigger Channel 2 Start Count
11	SIMPTC0	Hardware Trigger Channel 3 Start Count
12	SIMPTC1	Hardware Trigger Channel 0 Start Count

13	SIMPTC1	Hardware Trigger Channel 1 Start Count
14	SIMPTC1	Hardware Trigger Channel 2 Start Count
15	SIMPTC1	Hardware Trigger Channel 3 Start Count
16	SIMPTC2	Hardware Trigger Channel 0 Start Count
17	SIMPTC2	Hardware Trigger Channel 1 Start Count
18	SIMPTC2	Hardware Trigger Channel 2 Start Count
19	SIMPTC2	Hardware Trigger Channel 3 Start Count
20	SIMPTC3	Hardware Trigger Channel 0 Start Count
21	SIMPTC3	Hardware Trigger Channel 1 Start Count
22	SIMPTC3	Hardware Trigger Channel 2 Start Count
23	SIMPTC3	Hardware Trigger Channel 3 Start Count
24	D2F	Hardware Trigger D2F
25	IOCNT	Channel 0 Count Source
26	IOCNT	Channel 1 Count Source
27	IOCNT	Channel 2 Count Source
28	IOCNT	Channel 3 Count Source
29	IOCNT	Channel 4 Count Source
30	CLK_CAL	Hardware Trigger Clock Calibration
31	Reserved	Reserved
32	ADC0	Hardware Trigger Sequence A Start Conversion
33	ADC0	Hardware Trigger Sequence B Start Conversion
34	DAC	Hardware Trigger DAC Start Conversion

15.3.3 TC Trigger Sources

Table 15-2 TC0/TC1 Trigger Source Selection

Trigger Source Number	Source Peripheral	TC0 (Target 0~3) Trigger Signal	TC1 (Target 4~7) Trigger Signal
0	No Source Peripheral	No Trigger Signal	
1	UART0	RX Input Signal	
2	UART1	RX Input Signal	
3	UART2	RX Input Signal	
4	UART3	RX Input Signal	
5	UART4	RX Input Signal	
6	UART5	RX Input Signal	
7	UART6	RX Input Signal	
8	UART7	RX Input Signal	
9	LPUART	RX Input Signal	
10	USART0	RX Input Signal	
11	USART0	TX Input Signal	
12	USART1	RX Input Signal	
13	USART1	TX Input Signal	

14	ISO78160	IO Input Signal	
15	ISO78160	IN Input Signal	
16	ISO78161	IO Input Signal	
17	ISO78161	IN Input Signal	
18	Reserved	Reserved	
19	D2F	CF0 Output Signal	CF1 Output Signal
20	D2F	CF1 Output Signal	CF2 Output Signal
21	TC1/TC0	TC1 Channel 0 Positive Output	TC0 Channel 0 Positive Output
22	TC1/TC0	TC1 Channel 0 Negative Output	TC0 Channel 0 Negative Output
23	TC1/TC0	TC1 Channel 1 Positive Output	TC0 Channel 1 Positive Output
24	TC1/TC0	TC1 Channel 1 Negative Output	TC0 Channel 1 Negative Output
25	RTC	RTCOUT Output Signal	
26	GPIO	TCIN[0]	
27	GPIO	TCIN[1]	
28	GPIO	TCIN[2]	
29	GPIO	TCIN[3]	
30	GPIO	TCIN[4]	
31	GPIO	TCIN[5]	
32	GPIO	TCIN[6]	
33	GPIO	TCIN[7]	
34	GPIO	TCIN[8]	
35	GPIO	TCIN[9]	
36	GPIO	TCIN[10]	
37	GPIO	TCIN[11]	

15.3.4 SIMPTC Trigger Sources

Table 15-3 SIMPTC0~1 (8 channels) Trigger Source Selection

Trigger Source Number	Source Peripheral	SIMPTC0~1 Trigger Signal			
		Channel 0 (Target 8/12)	Channel 1 (Target 9/13)	Channel 2 (Target 10/14)	Channel 3 (Target 11/15)
0	No Source Peripheral	No Trigger Signal			
1	KBI0 Output Signal	Channel 0	Channel 1	Channel 2	Channel 3
2	KBI0 Output Signal	Channel 4	Channel 5	Channel 6	Channel 7
3	KBI1 Output Signal	Channel 0	Channel 1	Channel 2	Channel 3
4	KBI1 Output Signal	Channel 4	Channel 5	Channel 6	Channel 7

Table 15-4 SIMPTC2~3 (8 channels) Trigger Source Selection

Trigger Source Number	Source Peripheral	SIMPTC2~3 Trigger Signal			
		Channel 0 (Target 16/20)	Channel 1 (Target 17/21)	Channel 2 (Target 18/22)	Channel 3 (Target 19/23)
0	No Source	No Trigger Signal			

	Peripheral				
1	KBI0 Output Signal	Channel 8	Channel 9	Channel 10	Channel 11
2	KBI0 Output Signal	Channel 12	Channel 13	Channel 14	Channel 15
3	KBI1 Output Signal	Channel 8	Channel 9	Channel 10	Channel 11
4	KBI1 Output Signal	Channel 12	Channel 13	Channel 14	Channel 15

15.3.5 D2F Trigger Sources

Table 15-5 D2F Trigger Source Selection

Trigger Source Number	Source Peripheral	D2F (Target 24) Trigger Signal
0	No Source Peripheral	No Trigger Signal
1	SIMPTC0	Channel 0 Count Overflow Signal
2	SIMPTC1	Channel 0 Count Overflow Signal
3	SIMPTC2	Channel 0 Count Overflow Signal
4	SIMPTC3	Channel 0 Count Overflow Signal

15.3.6 IOCNT Trigger Sources

Table 15-6 IOCNT0~4 Trigger Source Selection

Trigger Source Number	Source Peripheral	IOCNT Trigger Signal				
		Channel 0 (Target 25)	Channel 1 (Target 26)	Channel 2 (Target 27)	Channel 3 (Target 28)	Channel 4 (Target 29)
0	No Source Peripheral	No Trigger Signal				
1	UART6	PF Output Signal (Single-phase Application)				
2	UART6	QF Output Signal (Single-phase Application)				
3	UART6	FPF Output Signal (Single-phase Application)				
4	UART7	PF Output Signal (Single-phase Application)				
5	UART7	QF Output Signal (Single-phase Application)				
6	UART7	FPF Output Signal (Single-phase Application)				
7	D2F	CF0 Output Signal				
8	D2F	CF1 Output Signal				
9	D2F	CF2 Output Signal				
10	KBI0 Input Signal	Channel 0	Channel 4	Channel 8	Channel 12	Channel 0
11	KBI0 Input Signal	Channel 1	Channel 5	Channel 9	Channel 13	Channel 1
12	KBI0 Input Signal	Channel 2	Channel 6	Channel 10	Channel 14	Channel 2
13	KBI0 Input Signal	Channel 3	Channel 7	Channel 11	Channel 15	Channel 3
14	KBI1 Input Signal	Channel 0	Channel 4	Channel 8	Channel 12	Channel 0
15	KBI1 Input Signal	Channel 1	Channel 5	Channel 9	Channel 13	Channel 1
16	KBI1 Input Signal	Channel 2	Channel 6	Channel 10	Channel 14	Channel 2
17	KBI1 Input Signal	Channel 3	Channel 7	Channel 11	Channel 15	Channel 3

15.3.7 CLK_CAL Trigger Sources

Table 15-7 CLK_CAL Trigger Source Selection

Trigger Source Number	Source Peripheral	CLK_CAL (Target 30) Trigger Signal
0	No Source Peripheral	No Trigger Signal
1	KBI0 Output Signal	Channel 0
2	KBI0 Output Signal	Channel 1
3	KBI0 Output Signal	Channel 2
4	KBI0 Output Signal	Channel 3
5	KBI1 Output Signal	Channel 0
6	KBI1 Output Signal	Channel 1
7	KBI1 Output Signal	Channel 2
8	KBI1 Output Signal	Channel 3
9	SIMPTC0	Channel 0 Count Overflow Signal
10	SIMPTC1	Channel 0 Count Overflow Signal
11	SIMPTC2	Channel 0 Count Overflow Signal
12	SIMPTC3	Channel 0 Count Overflow Signal

15.3.8 ADC0 Trigger Sources

Table 15-8 ADC0 Trigger Source Selection

Trigger Source Number	Source Peripheral	ADC0 Trigger Signal	
		Sequence A (Target 32)	Sequence B (Target 33)
0	No Source Peripheral	No Trigger Signal	
1	KBI0 Output Signal	Channel 0	Channel 4
2	KBI0 Output Signal	Channel 1	Channel 5
3	KBI0 Output Signal	Channel 2	Channel 6
4	KBI0 Output Signal	Channel 3	Channel 7
5	KBI1 Output Signal	Channel 0	Channel 4
6	KBI1 Output Signal	Channel 1	Channel 5
7	KBI1 Output Signal	Channel 2	Channel 6
8	KBI1 Output Signal	Channel 3	Channel 7
9	SIMPTC0 Count Overflow Signal	Channel 0	Channel 1
10	SIMPTC1 Count Overflow Signal	Channel 0	Channel 1
11	SIMPTC2 Count Overflow Signal	Channel 0	Channel 1
12	SIMPTC3 Count Overflow Signal	Channel 0	Channel 1

15.3.9 DAC Trigger Sources

Table 15-9 DAC Trigger Source Selection

Trigger Source Number	Source Peripheral	DAC (Target 34) Trigger Signal
0	No Source Peripheral	No Trigger Signal
1	KBI0 Output Signal	Channel 8
2	KBI0 Output Signal	Channel 9
3	KBI0 Output Signal	Channel 10
4	KBI0 Output Signal	Channel 11
5	KBI1 Output Signal	Channel 8
6	KBI1 Output Signal	Channel 9
7	KBI1 Output Signal	Channel 10
8	KBI1 Output Signal	Channel 11
9	SIMPTC0 Count Overflow Signal	Channel 2
10	SIMPTC1 Count Overflow Signal	Channel 2
11	SIMPTC2 Count Overflow Signal	Channel 2
12	SIMPTC3 Count Overflow Signal	Channel 2

15.4 Registers

AOS Module Register Base Addresses

Module Name	Secure Mapping Address	Non-secure Mapping Address
AOS	0x2008_8000	0xA008_8000

AOS Interface Register Offset Addresses

Register Name	Address Offset	Description
SEC_EN	0x00	Secure Peripheral Linkage Enable Register
NSEC_EN	0x04	Non-secure Peripheral Linkage Enable Register
PE_TRGSELx (x=0~34)	0x10~0x98	Target Peripheral Trigger Source Selection Register

15.4.1 SEC_EN（0x00）

Secure Peripheral Linkage Enable Register, accessible only via address 0x2008_8000 (Only Core0 can access this secure address).

Bit	Name	Description	R/W	Reset Value
31:8	----	Reserved	RO	0x0
7:0	SEC_EN	Secure Peripheral Linkage Enable Bit 0xa5: Enable Other values: Disable	RW	0x0

15.4.2 NSEC_EN（0x04）

Non-secure Peripheral Linkage Enable Register, accessible only via address 0xA008_8004.

Bit	Name	Description	R/W	Reset Value
31:8	----	Reserved	RO	0x0
7:0	NSEC_EN	Non-secure Peripheral Linkage Enable Bit 0x8E: Enable Other values: Disable	RW	0x0

15.4.3 PE_TRGSELx（0x10~0x88, 0x90~0x98）

Target Peripheral Trigger Source Selection Register x (x=0,1,...,30, 32,33,34)

Secure target peripheral registers can only be accessed via 0x2008_8000 + offset address.

Non-secure target peripheral registers can only be accessed via 0xA008_8000 + offset address.

Bit	Name	Description	R/W	Reset Value
31:6	----	Reserved	RO	0x0
5:0	TRGSEL	Trigger Source Event Number. Writing 0 means no trigger source event is selected.	RW	0x0

15.5 Software Configuration Flow and Application Notes

Secure Core Core0 configuring Peripheral 0 to trigger Peripheral 1:

- Core0 configures both Peripheral 0 and Peripheral 1 as secure peripherals.
- Core0 configures Peripheral 0 related function registers and starts Peripheral 0.
- Core0 configures Peripheral 1 related function registers and starts Peripheral 1.
- Core0 writes SYSCTL->PMOD0_CKE |= (1 << 2) to enable the CORE0_AOS module clock, then configures AOS->PE_TRGSEL1 to select Peripheral 0's trigger signal as the trigger source for Peripheral 1.
- Core0 configures AOS->SEC_EN=0xa5 to enable secure peripheral linkage.

Application Core Core1 configuring Peripheral 0 to trigger Peripheral 1:

- Core0 configures both Peripheral 0 and Peripheral 1 as non-secure peripherals.
- Core1 configures Peripheral 0 related function registers and starts Peripheral 0.
- Core1 configures Peripheral 1 related function registers and starts Peripheral 1.
- Core1 writes SYSCTL->PMOD0_CKE |= (1 << 2) to enable the CORE1_AOS module clock, then configures AOS->PE_TRGSEL1 to select Peripheral 0's trigger signal as the trigger source for Peripheral 1.
- Core1 configures AOS->NSEC_EN=0x8E to enable non-secure peripheral linkage.

16 WDT

16.1 Main Features

- Watchdog Timer uses the internal low-speed oscillator circuit.
- Count clock can be configured with 2/4/8/16/32/64/128 division.
- Watchdog Timer overflow time settings: 16ms, 32ms, 128ms, 512ms, 1s, 2s, 4s, 8s.
- Watchdog Timer is used to detect unexpected program loops.
- When the Watchdog Timer generates a reset, the WDT_RST bit in the Reset Flag Register is set to 1.
- An interval interrupt can be generated when 75% of the overflow event is reached.
- Wake-up interrupt count value can be configured.
- When a WDT0 reset occurs, both Core0 and Core1 are reset.
- WDT1 only resets Core1.

Watchdog reset is generated under the following conditions:

- If the Watchdog Timer counter overflows.
- If data other than "BB" is written to WDT_EN.
- If data is written to WDT_EN during the window closed period.
- If data is written to WDT_EN in the WDT Bitband space.

16.2 Register Description

WDT Module Register Base Addresses

Module Name	Core0 Base Address	Core1 Base Address
WDT0	0x2010_0000	--
WDT1	--	0xA010_0000

WDT Module Register Offset Addresses

Register Name	Address Offset	Description
WDT_EN	Offset+0x0	WDT Enable Register
WDT_CTL	Offset+0x4	WDT Configuration Register
WDT_PASS	Offset+0x8	WDT Password Register
WDT_HALT	Offset+0x14	Controls whether WDT stops counting when CPU is in HALT (spl1)
WDT_STBY	Offset+0x18	Controls whether WDT stops counting when CPU is in SLEEP (spl1)
WDT_STOP	Offset+0x1C	Controls whether WDT stops counting when CPU is in STOP or no clock
WDT_CNT	Offset+0x20	Watchdog Count Register

16.2.1 Watchdog Enable Register WDT_EN（0x00）

Bit	Name	Description	R/W Flag	Reset Value

31:9	----	Reserved	R	0
8	WR_BUSY	WDT register write operation busy. When WR_BUSY =1, WDT_EN, WDT_CTL are not writable. Only when WR_BUSY =0, can WDT_EN, WDT_CTL registers be written. (WDT_PASS does not get BUSY)	R	0
7:0	WDTE	Writing BBH to WDT_EN clears the Watchdog Timer and starts the counting operation again. Reset signal generation sets this register to 55H or AAH. When WDTON bit=0 (Watchdog operation disabled), WDT_EN reset value is 55H. When WDTON bit=1 (Watchdog operation enabled), WDTE reset value is AAH.	RW	55H or AAH

16.2.2 Watchdog Configuration Register WDT_CTL (0x04)

Bit	Name	Description	R/W Flag	Reset Value
31	----	Reserved	R	0
30:12	WAKE_CNT	Wake Configuration Counter Value	WR	0x0
11	RAD_WAKE	Arbitrary Wake Configuration Enable =1: Enable =0: Disable	RW	0
10:8	CLKDIV	Clock Division Configuration =0: No division =1: Divide by 2 =2: Divide by 4 =3: Divide by 8 =4: Divide by 16 =5: Divide by 32 =6: Divide by 64 =7: Divide by 128	RW	0
7	WDTINT	Watchdog Timer Interval Interrupt =0: Do not use interval interrupt =1: Generate interval interrupt when 75% of overflow event is reached.	RW	0
6:5	WINDOW	Window Open Period: 00 25% 01 50% 10 75% 11 100% If BBH is written to WDTE during the window open period, the watchdog is cleared and restarts counting. If BBH is written to WDTE during the window closed period, an internal reset signal is generated.	RW	11
4:3	--	Reserved	R	0

2:0	WDCS	Watchdog Timer Overflow Time: 000=15.625ms*(CLKDIV+1) 001=31.25ms*(CLKDIV+1) 010=125ms*(CLKDIV+1) 011=500ms*(CLKDIV+1) 100=1s*(CLKDIV+1) 101=2s*(CLKDIV+1) 110=4s*(CLKDIV+1) 111=8s*(CLKDIV+1)	RW	100

When RAD_WAKE is configured to 1, the 75% overflow event wake-up becomes invalid; wake-up occurs when the counter is greater than or equal to WAKE_CNT.

16.2.3 Watchdog Password Register WDT_PASS(0x08)

Bit	Name	Description	R/W Flag	Reset Value
31:8	----	Reserved	R	0
7:4	PASS	Password is 0x7 Note: Only when this password is correct can WDT_PASS[3:0] be written.	RW	0
3:0	PASS	Password is 0xE Note: WDT_CTL can only be written when PASS[7:0]=0x7E. Steps to configure WDT_PASS: WDT->PASS = 0x70; WDT->PASS = 0x7e;	RW	0

16.2.4 Watchdog HALT Control Register WDT_HALT(0x14)

Bit	Name	Description	R/W Flag	Reset Value
31:0	HALT_EN	When the CPU is in HALT (used during debugging), if HALT_EN=32'hA5A5_A581, then pause WDT counting. If HALT_EN is any other value, then do not pause WDT counting.	RW	0x5A5A_5A7E

Notes:

1. HALT function is not enabled by default after reset (needs to be enabled by bootrom).
2. The purpose of this register is to reduce the probability of turning off the WDT in case of unsuccessful power-on reset, allowing the WDT to reset the system.
3. This register is protected by a password protection bit.

16.2.5 Watchdog STBY Control Register WDT_STBY(0x18)

Bit	Name	Description	R/W Flag	Reset Value
31:0	STBY_EN	STBY_EN = 32'hA5A5_A5C7: Disable WDT when the CPU is in standby mode; STBY_EN = Other values: Do not disable WDT;	RW	0x5A5A_5A38

16.2.6 Watchdog STOP Control Register WDT_STOP(0x1C)

Bit	Name	Description	R/W Flag	Reset Value
31:0	STOP_EN	STOP_EN=32'Haa55_55AA: Turn off WDT when CPU is in STOP or clock is off. STOP_EN=Other values: Do not turn off WDT.	RW	0x55AA_AA55

16.2.7 Watchdog Counter Register WDT_CNT(0x20)

Bit	Name	Description	R/W Flag	Reset Value
31:19	----	Reserved	R	0
18:0	CNT	Watchdog Counter Value	R	0x0

17 SIMPTC

17.1 Introduction

The SIMPTC module is a simple system tick counter that counts using the system clock of the CPU.

There are four SIMPTC modules (SIMPTC0, SIMPTC1, SIMPTC2, SIMPTC3). Each SIMPTC module integrates four independently controlled 32-bit counters. Each counter has two operating modes:

- One-shot mode;
- Cycle mode;

17.2 Functional Description

The SIMPTC module is a simple system tick counter that counts using the system clock of the CPU. It generates an overflow flag when the count value reaches the preset target value.

The counter supports software-enabled counting and hardware-enabled counting. Software counting refers to controlling the counter to start counting by configuring the count enable bit of the register. Hardware counting refers to the synchronous count enable pulse forwarded by AOS controlling the counter to start counting. The counter is cleared each time software counting or hardware counting is exited.

The hardware-enabled counter only supports one-shot counting mode. The software-enabled counter supports two different counting modes: one-shot and cycle. One-shot mode means that after starting the count, the counter generates an overflow flag and stops counting when the count value reaches the preset target value. Cycle mode means that after starting the count, the counter generates an overflow flag and continues counting from 0 when the count value reaches the preset target value.

During the counting process, when the CPU debug is halted, the counter pauses counting by default. When the CPU resumes running, the counter continues counting. The behavior of the counter pausing counting due to CPU halt control can be disabled by software configuration.

17.3 Registers

SIMPTC Module Register Base Addresses

Module Name	Core0 Base Address	Core1 Base Address
SIMPTC0		
SIMPTC0_0	0x200A_4000	0xA00A_4000
SIMPTC0_1	0x200A_4010	0xA00A_4010
SIMPTC0_2	0x200A_4020	0xA00A_4020
SIMPTC0_3	0x200A_4030	0xA00A_4030
SIMPTC1		
SIMPTC1_0	0x200A_8000	0xA00A_8000
SIMPTC1_1	0x200A_8010	0xA00A_8010
SIMPTC1_2	0x200A_8020	0xA00A_8020
SIMPTC1_3	0x200A_8030	0xA00A_8030
SIMPTC2		
SIMPTC2_0	0x200A_C000	0xA00A_C000
SIMPTC2_1	0x200A_C010	0xA00A_C010
SIMPTC2_2	0x200A_C020	0xA00A_C020

SIMPTC2_3	0x200A_C030	0xA00A_C030
SIMPTC3		
SIMPTC3_0	0x200B_0000	0xA00B_0000
SIMPTC3_1	0x200B_0010	0xA00B_0010
SIMPTC3_2	0x200B_0020	0xA00B_0020
SIMPTC3_3	0x200B_0030	0xA00B_0030

SIMPTC Interface Register Offset Addresses

Register Name (x=0~3, y=0~3)	Address Offset	Description
SIMPTCx_y_CTRL	0x0	SIMPTCx_y Control Register
SIMPTCx_y_LOAD	0x4	SIMPTCx_y Target Count Value Register
SIMPTCx_y_VAL	0x8	SIMPTCx_y Current Count Value Register
SIMPTCx_y_IF	0xc	SIMPTCx_y Interrupt Flag Register

17.3.1 SIMPTCx_y_CTRL（0x0）

SIMPTCx_y Control Register

Offset=0x0

Bit	Name	Description	R/W	Reset Value
31:5	---	Reserved	R	0x0
4	IRQEN	Interrupt Enable: 0: Disable interrupt; 1: Enable interrupt;	RW	0x0
3	MODE	Cycle Count Mode Configuration Bit: (Only valid for software-enabled counting; AOS pulse enable counting only has one-shot counting mode) 0: One-shot counting; 1: Cycle counting;	RW	0x0
2	DBGSTBDIS	CPU Debug State Counter Count Enable: 0: Not enabled; (Counter pauses counting when CPU is in debug state) 1: Enabled; (Counter continues counting when CPU is in debug state)	RW	0x0
1	CNT_SEL	Counter Start Enable Source Selection AOS/Software Configuration: 0: Software enable counting; 1: AOS pulse enable counting;	RW	0x0
0	CNT_SFT_EN	Counter Software Enable Bit: 0: Stop counting; 1: Start counting;	RW	0x0

17.3.2 SIMPTCx_ y_DN（0x4）

SIMPTCx_y Target Count Value Register

Offset=0x4

Bit	Name	Description	R/W	Reset Value
31:0	DN	Target Count Value: DN + 1	RW	0x0

17.3.3 SIMPTCx_ y_VAL（0x8）

SIMPTCx_y Current Count Value Register

Offset=0x8

Bit	Name	Description	R/W	Reset Value
31:0	VAL	Counter Current Count Value	R	0x0

17.3.4 SIMPTCx_ y_IF（0xC）

SIMPTCx_y Interrupt Flag Register

Offset=0x0c

Bit	Name	Description	R/W	Reset Value
31:1	---	Reserved	R	0x0
0	OVIF	Count Overflow Flag Bit 0: Counter has not overflowed 1: Counter has overflowed Note: Write 1 to clear	RW	0x0

18 UART

18.1 Introduction

This module is a UART module (Universal Asynchronous Receiver/Transmitter, UART), used for asynchronous serial communication with external devices.

The chip has a total of 8 UART interfaces. UART0-5 are conventional UART communication interfaces. In addition to conventional UART interface functions, UART6-7 additionally integrate metering mode UART functions for use with the HSTX module of MAKSA single-phase metering chips. In metering mode, when UART6-7 communicate with single-phase metering chips, the peripheral isolation components required for communication between the MCU and the metering chip can be reduced, achieving the goal of cost reduction.

The HSTX/LSTX module interface is a communication interface designed by MAKSA single-phase chips to reduce the number of optocoupler isolation components required in single-phase electricity meter implementation solutions while ensuring real-time waveform transmission. This interface achieves communication that converts pulse signals and zero-crossing signals into pulse data, zero-crossing data, and other metering data at a relatively low cost. For details, please refer to the HSTX/LSTX section of the MAKSA single-phase metering chip documentation.

18.2 Features

The chip integrates a total of 8 UART interfaces for external asynchronous serial communication, with the following features:

- There are 8 UART interfaces in total;
- Supports peripheral security management; each UART resource can be independently configured with security attributes to meet communication security requirements;
- UART6-7 support automatic operation system linkage with the AOS module to implement waveform forwarding via the IOCNT module;
- UART6-7 support cooperation with the HSTX module in MAKSA single-phase metering chips to achieve DMA transmission of zero-crossing data, active power, and reactive power data, as well as waveform forwarding;
- UART6 and 7 support cooperation with the HSTX module in MAKSA Micro single-phase metering chips, can automatically identify the data frame header sent by HSTX, and achieve high-speed data transmission and waveform forwarding;
- Built-in baud rate generator, minimum 300 baud rate, maximum baud rate supports fcpu/16, supports fractional baud rate division;
- UART module RX and TX support two GPIO multiplexing methods: independent multiplexing and bound multiplexing;
- The UART module supports dedicated DMA function, enabling high-speed data communication between UART and SRAM without CPU intervention;
- Supports infrared debugging output; all serial ports support 38kHz infrared modulation output; infrared demodulation needs to be implemented through external components;
- Supports UART wake-up; when the CPU is in SLEEP mode, it can be woken up via UART;
- Supports transmit FIFO with a depth of 2; receive does not support FIFO;
- Data bit width supports configurable 5/6/7/8 bits;
- Stop bit width supports configurable 1/2 bits;

- Parity bit supports no parity, 1-bit odd/even parity, fixed 0/1 parity;
- Supports positive and negative polarity data transmission;
- Supports internal loopback function;
- Supports independent configuration of peripheral security attributes to achieve peripheral security management;
- Supports AOS automatic operation system, enabling linkage with other peripherals without CPU intervention;
- Supports cooperation with the LSRX module to achieve correction of zero-crossing data from the metering chip's LSTX module;
- UART6 and UART7 support hardware parsing of communication with the metering chip's HSTX module, adding three new modes: UART_CF, UART_TW, UART_CAL, adding functions to receive, identify, and store data sent by the metering chip's HSTX module, send calibration waveforms to the metering chip, and detect the metering chip's operating frequency;
- Baud rate configuration parameter UARTx_BAUD is expanded to 14 bits, enabling 300 baud rate communication under a 58MHz clock;
- RX/TX support two multiplexing methods: independent multiplexing and bound multiplexing;
- Added configurable function for 1-8 baud rate cycle delay between data frames in transmit mode.

18.3 Block Diagram

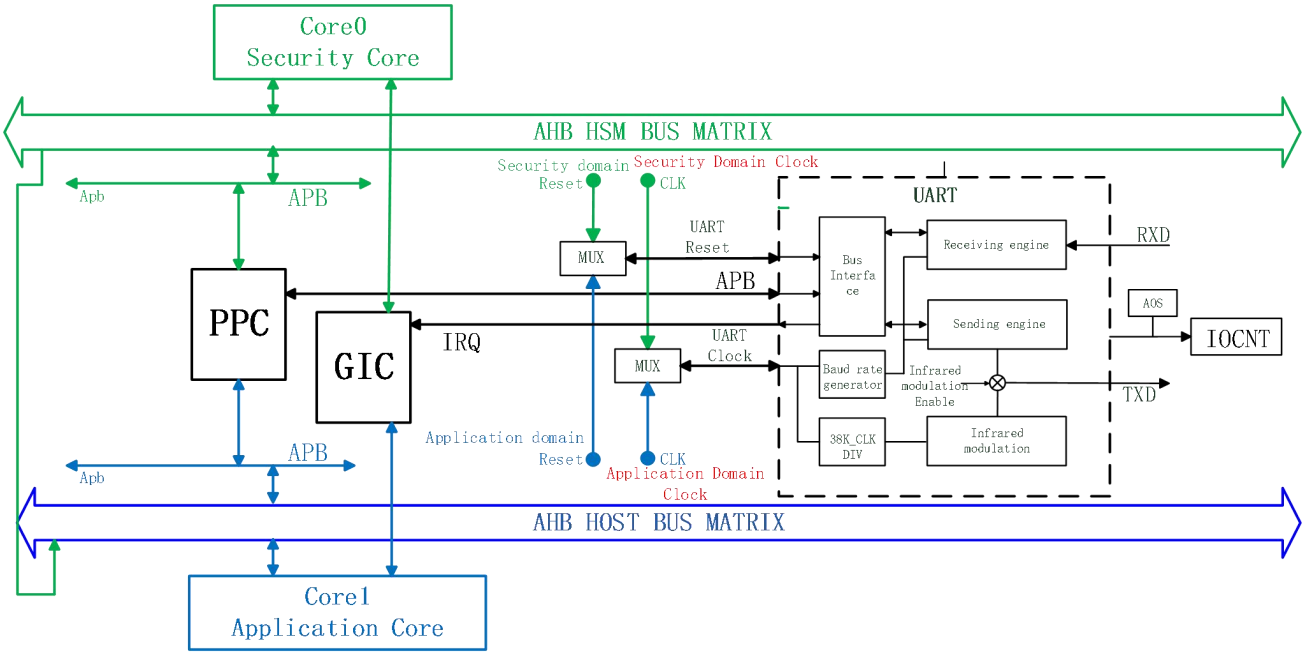


Figure 18-1 UART Module Structure Diagram

The block diagram of the UART module is shown in the figure above. When a secure or non-secure chip accesses the UART, it undergoes security attribute checking through the PPC gatekeeper. Access is only successful if the current access complies with SCU access rules.

The internal UART baud rate generator and the 38k infrared modulation divider are controlled by gated clocks (UART clock gate and 38k infrared modulation clock gate). When infrared modulation enable is turned on, the data from the transmit engine is modulated by the 38k carrier signal before output.

When the AOS function of the UART is enabled, after the UART as a trigger source triggers IOCNT, it

directly sends PF and QF data to the IOCNT module.

18.4 Pin Definition

The signals of the UART module consist of two lines: RXD and TXD. RXD is the input for communication data, and TXD is the output for data. When the module operates in positive polarity mode, a high level represents logic "1" and a low level represents logic "0" (data polarity can be configured via register UARTx_CTRL[14] NEG bit). The multiplexing relationships for UART are as follows:

Peripheral	IO	
UART0	RX0	PD12、PE01、PE15、PE03、PE04、PG12
	TX0	PD05、PD14、PE14、PE02、PE05、PG13
UART1	RX1	PD12、PE01、PE15、PE03、PE04、PG12
	TX1	PD05、PD14、PE14、PE02、PE05、PG13
UART2	RX2	PA00、PE00、PE15、PG03、PG07、PG14
	TX2	PA01、PE14、PF07、PG09、PG08、PG15
UART3	RX3	PA00、PE00、PE15、PG03、PG07、PG14
	TX3	PA01、PE14、PF07、PG09、PG08、PG15
UART4	RX4	PA15、PB02、PB05、PD08、PG05、PG07
	TX4	PB00、PB01、PB06、PD09、PG04、PG08
UART5	RX5	PA15、PB02、PB05、PD08、PG05、PG07
	TX5	PB00、PB01、PB06、PD09、PG04、PG08
UART6	RX6	PA03、PB04、PB13、PC14、PG10、PH00
	TX6	PA04、PB03、PB07、PC06、PG11、PH01
UART7	RX7	PA03、PB04、PB13、PC14、PG10、PH00
	TX7	PA04、PB03、PB07、PC06、PG11、PH01

Table 18-1 UART 0~7 Multiplexing Relationship List

18.5 Interface Timing

During UART transmission, a 1-bit wide low level is first maintained as the start frame, followed by the data frame content (5-8 bits), then according to the frame format configuration, the parity bit and stop bit are sent. The stop bit is a high level maintained for 1-bit width. During transmission, the transmit engine sets and holds a busy state high starting from the entry of the first data, returning to low level when the stop bit is transmitted.

During UART reception, RXD remains high before valid data arrives. The first falling edge of the start bit appears, maintaining a 1-bit low level. Then, according to the frame format configuration, the corresponding length of data bits, parity bit, and finally the 1-bit high level stop bit are received. During reception, the receiver performs three samples at the middle position of each bit as signal sampling. The receive busy flag is set upon detecting the RXD falling edge and returns to low level after the stop bit is sampled.

18.6 Functional Description

18.6.1 Security Attribute Configuration Description

The security attributes of the UART module must comply with SCU access rules. Here is a brief overview of SCU detection of peripherals; for detailed content, please refer to the SCU section.

UART is a dual-bus peripheral, managed by DPPC security access rules:

- When the UART is set as a secure peripheral, it can only be accessed via the secure address.
- When the UART is set as a non-secure peripheral, Core0 and Core1 can access it via the non-secure address.

SCU Access Rule / UART Security Attribute	CORE0 Secure Host Access		CORE1 Non-Secure Host Access	
		Secure Address	Non-Secure Address	Secure Address
UART - Secure Peripheral	Legal	Illegal, S Event	Illegal, RAZWI	Illegal, S Event
UART - Non-Secure Peripheral	Illegal, S Event	Legal	Illegal, RAZWI	Legal

Table 18-2 UART 0~7 Security Access Rules

When UART is multiplexed with GPIO, it is also important to note that GPIO access must also comply with SCU security rules. In actual configuration, the GPIO multiplexed by UART must correspond to the security domain of the peripheral, otherwise illegal access (RAZWI) will occur.

Specifically, the security attribute configuration of UART only supports the following 4 scenarios:

Secure Host	Secure UART	Secure Address	Secure GPIO
	Non-Secure UART	Non-Secure Address	Non-Secure GPIO
Non-Secure Host	Non-Secure UART	Non-Secure Address	Secure GPIO
	Non-Secure UART	Non-Secure Address	Non-Secure GPIO

Table 18-3 UART 0~7 Legal Configuration Scenarios

18.6.2 Baud Rate

The baud rate generator produces the 16x baud rate clock required for transmission and reception. Taking a system clock of 3.6864MHz as an example, typical baud rate configurations are as follows:

Target Baud Rate	Baud Rate Config	Baud Rate Error	Target Baud Rate	Baud Rate Config	Baud Rate Error
1200	0xBF	0%	9600	0x17	0%
2400	0x5F	0%	19200	0xB	0%
4800	0x2F	0%	38400	0x5	0%

Table 18-4 Typical Baud Rate Configuration under 3.6864MHz Clock

Baud rate can be configured by setting UARTx_BAUD and UARTx_FDIV. UARTx_BAUD is the integer part of the baud rate division preset value, calculated according to the formula UARTx_BAUD = System Clock / [16*(CLKDIV+1)]; UARTx_FDIV is the fractional part of the baud rate division preset value, calculated according to the following formula (where fi is the system clock frequency, fo is the baud rate frequency):

$$F = \left[\left(\frac{fi}{16 \times fo} - \left[\frac{fi}{16 \times fo} \right] \right) \times 2^{14} + 0.5 \right]$$

Note that the UARTx_FDIV register bit width is [13:0]. In actual calculation, a situation where FDIV=0x4000 might occur. In this case, setting UARTx_FDIV = 0x4000 results in UARTx_FDIV = 0, causing an error (this can be avoided by approximating UARTx_FDIV = 0x4000 as UARTx_FDIV = 0x3FFF).

18.6.3 Data Frame Format

The conventional serial communication function allows configuration of the serial communication data frame format through registers, including data bits, parity bits, and stop bits.

Specific settings are as follows:

- Parity Bit Setting: UARTx_CTRL[9:7] (000 - No parity; 001 - Odd parity; 010 - Even parity; 011 - Fixed 0 parity; 100 - Fixed 1 parity);
- Data Bit Setting: UARTx_CTRL[6:5] (00 - 5 bits; 01 - 6 bits; 10 - 7 bits; 11 - 8 bits);
- Stop Bit Setting: UARTx_CTRL[4] (0 - 1-bit stop bit; 1 - 2-bit stop bits);

18.6.4 Other Settings

- LSB/MSB:
  UARTx_CTRL[13]: 0: LSB first; 1: MSB first.
- Internal Loopback:
  UARTx_CTRL[11]: 0: Disable loopback; 1: Enable loopback.
- Data Polarity Selection:
  UARTx_CTRL[14]: 0: Positive polarity; 1: Negative polarity.

18.6.5 Infrared Modulation Module

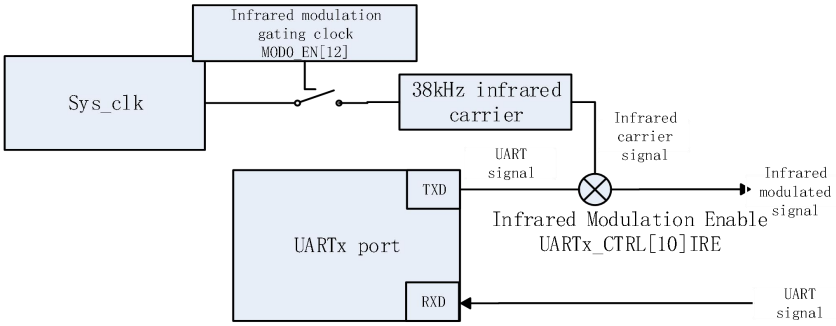


Figure 18-2 UART Infrared Modulation Function Diagram

To meet the purpose of sending and receiving data through wireless infrared communication, a 38kHz infrared frequency modulation module is loaded onto the UART module. As shown in Figure 18-4, this module modulates the original UART serial signal onto a 38kHz carrier signal and sends it out in the form of an infrared signal through the infrared module. Specifically, through modulation, the original signal "0" is sent outward in the form of a 38kHz carrier, while the signal "1" does not send any carrier and is sent outward in its original signal form, as shown in Figure 18-5.

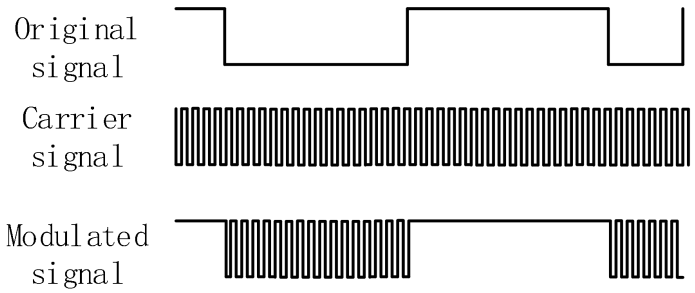


Figure 18-3 Signal Changes Before and After Infrared Modulation

In specific use, in addition to setting the UART communication settings, it is also necessary to configure

registers related to the infrared modulation part, including enabling the infrared module, enabling the infrared module gated clock, configuring the infrared module data polarity, etc.

18.6.6 DMA Function

UART shares the dedicated DMA function with other peripherals. The dedicated DMA can transfer data between the peripheral port and memory without CPU intervention. The dedicated DMA uses time-division multiplexing to respond to peripherals requesting dedicated DMA. When using dedicated DMA with UART, it is necessary to establish the relationship between the UART and the dedicated DMA peripheral channel to set the source and destination of UART data.

18.6.7 AOS Function

The AOS automatic operation system enables linkage between peripherals without CPU intervention. The AOS system follows security access rules, prohibiting peripheral devices from triggering across security domains. The implementation of the UART module's AOS function supports UART in metering mode as a trigger source to trigger IOCNT, achieving pulse waveform forwarding.

18.6.8 LSRX Linkage

UART can cooperate with the LSRX module to achieve communication with the metering chip's LSTX module. The application solutions are divided into single-phase and three-phase, as shown in the following two schemes.

18.6.8.1 Single-Phase Connection Scheme

The connection scheme under single-phase conditions is shown in the figure below. In conventional applications, the UART is first used to configure the registers and read related data of the single-phase chip. When the metering chip enables the LSTX module, the LSTX module periodically sends register data, pulse data, and zero-crossing data to the host through the metering chip's TX pin. After receiving this data, the host stores it in SRAM, and simultaneously, the LSRX part parses the pulse data and zero-crossing data, performs pulse forwarding and zero-crossing event upload, achieving the purpose of sending multiple types of data through a single pin.

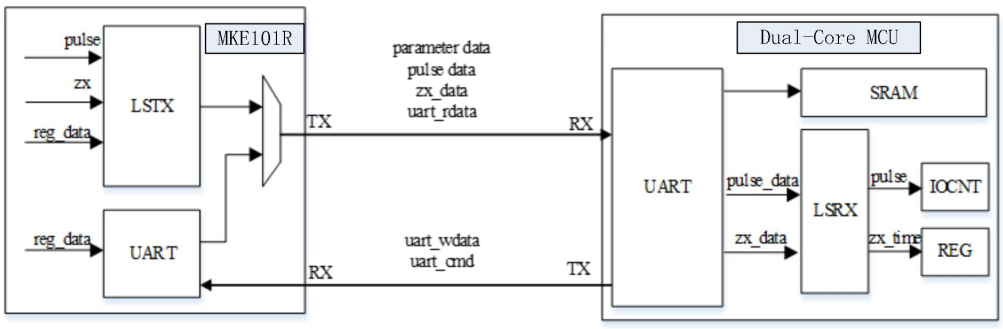


Figure 18-4 Single-Phase Connection Scheme

18.6.8.2 Three-Phase Scheme

The connection scheme under three-phase conditions is shown in the figure below. Only the TX pin of the

metering chip is connected to the MCU, and the RX pin is pulled up. Three metering chips are connected to the host. After power-on reset, the metering chip's LSTX starts automatically, uploading parameter register data and zero-crossing data to the host through the TX pin. After storing the data in SRAM via DMA, the host simultaneously parses and calculates the zero-crossing data through LSRX to obtain the zero-crossing time. By obtaining the zero-crossing times of the three single-phase chips, the phase angles between the three channels can be calculated.

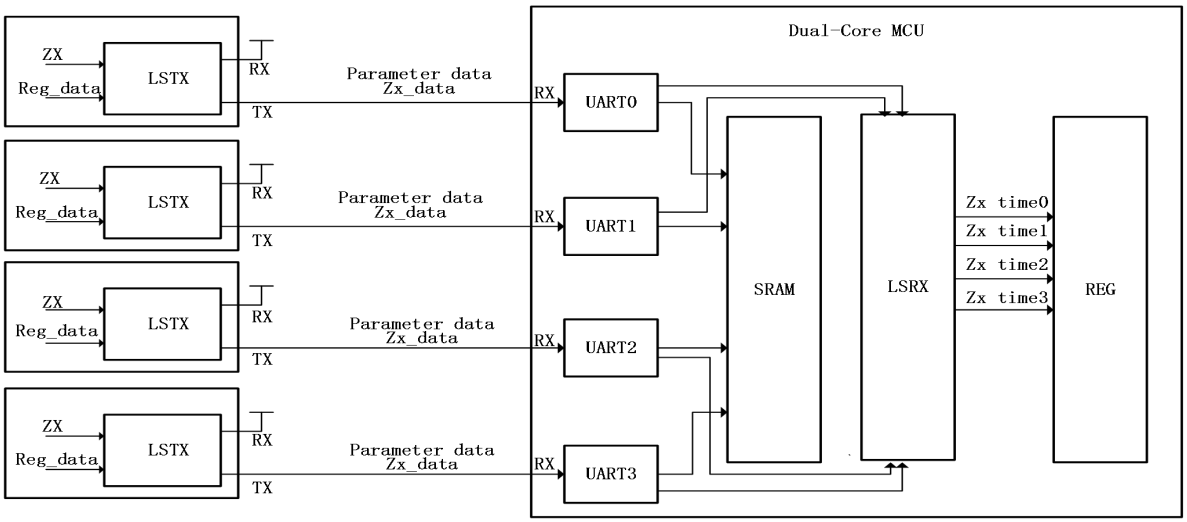


Figure 18-5 Three-Phase Connection Scheme

18.6.9 UART Metering Mode

This mode applies to UART6 and UART7. In this mode, the UART interface can cooperate with MAKSA single-phase metering chips equipped with an HSTX module to implement decoding functions for communication with the single-phase metering chip. By providing a special UART metering mode, the MCU can decode data sent by the HSTX module of the single-phase metering chip when received. This data includes pulse data, parameter register data, interrupt data, and waveform data. Pulse data and interrupt data have their own feature codes and can be customized according to user requirements. For details, refer to the LSTX module introduction of the relevant MAKSA single-phase metering chip.

18.6.9.1 UART Metering Modes

UART metering mode has three modes: UART_CF mode, UART_TW mode, UART_CAL mode.

- UART_CF Mode: This mode is used for the metering chip MKE101R to send data to the MCU, and the MCU receives and parses the data from the metering chip. The priority of the three types of data received by UART is: Pulse data > Interrupt data > Parameter register data = Waveform data. The data transmission rules are as follows:
  - Parameter register data, waveform data: Sent in a fixed form during each half-full interrupt. After receiving parameter register data and waveform data, the dual-core MCU stores the data in SRAM via DMA for further processing.
  - Pulse data: Sent at any time (sent as soon as it is generated; if the interface is sending other data, wait for the current byte (or word) to finish sending before sending). After the dual-core MCU receives pulse data, it is not stored in SRAM. It generates a corresponding pulse for the next stage and raises the corresponding pulse interrupt flag.

- Interrupt data: Sent at any time (the single-phase metering chip sends interrupt data each time an interrupt flag is raised; if the interface is sending other data, wait for the current byte (or word) to finish sending before sending). Priority is lower than pulse data. The interrupt data can be chosen whether to store in SRAM. After receiving interrupt data, an interrupt flag is raised once.
- If a pulse signal is generated by the single-phase metering chip during the transmission of interrupt data, wait for the interrupt data to finish sending before sending the pulse data.

To prevent data confusion, four bytes of fixed data (this fixed data does not repeat with parameter register data or waveform data, the last byte is a checksum) are used to represent the output of pulse signals and interrupt signals. Among them, the last byte of each data inserts an incorrect check code to increase distinction. (If communication is configured for odd parity, send even parity code; if configured for even parity, send odd parity code; if for zero parity, send check code 1; if for 1 parity, send check code 0). Three different sets of fixed data represent PF, QF, FPF pulses respectively. The specific output method of pulse data is as shown in the table below:

Pulse Type	Data 0	Data 1	Data 2	Data 3
PF Pulse	80H	0AH	20H	AAH
QF Data	80H	0CH	40H	CCH
FPF Pulse	80H	0EH	60H	EEH
Interrupt Signal	80H	08H	20H	A8H

Table 18-5 Pulse Data Output Signals

Features of this mode:

- UART_CF mode is used to receive pulse data. The mode is turned on or off by the enable signal; when the mode is off, UART returns to conventional functional mode.
- UART_CF mode can identify pulse data and, through the AOS module, forward the pulse to the IOCNT module and generate corresponding interrupts.
- After a check code error occurs in the data received by UART_CF mode, an error interrupt is generated, but the data is still written to SRAM. Only after receiving and identifying pulse/interrupt data, if a check bit error interrupt occurs four times consecutively, will the data not be written to SRAM.
- UART_CF mode has a timeout detection mechanism. If no new data is received for a period after receiving part of the pulse data, an error is reported. The timeout time is configurable. When timeout cache clearing is enabled, upon a timeout event, the hardware directly writes the identified part of the pulse data to SRAM.
- UART mode identifies UART_IRQ_DATA and raises the interrupt flag, deciding whether to write the interrupt data to SRAM according to the configuration.

- UART_TW Mode

This mode is used for the MCU chip's UART module TX to separately send a low level to the metering chip. After this mode is enabled, the UART mode TX switches to sending a configurable number of fixed-spacing low-level signals. After exiting this mode, TX returns to the UART_TX function.

The UART module TX outputs low levels at certain intervals. The low-level length is about 2ms, equivalent to 10 bits at 4800 baud rate. The interval high-level length is 4 times the low-level length. A single waveform is shown in the figure below. The user configures the UART_BAUD register according to the module clock cycle to generate a 4800 baud rate.

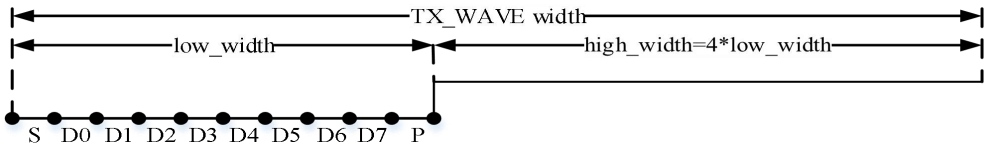


Figure 18-6 UART_TW Mode Transmit Waveform Diagram

- UART_CAL Mode

This is the baud rate measurement mode. In this mode, the MCU's UART can measure the baud rate of the signal received on the RX pin. There are continuous measurement mode and fixed data measurement mode. The measurement scheme is: the single-phase metering chip sends a signal of a certain bit length (baud rate close to the agreed baud rate) to the MCU chip. After receiving it, the MCU chip measures the signal length using the system clock. After measurement is complete, an interrupt is sent to the user. The user software calculates the actual baud rate of the metering chip by reading the measurement value register. The trigger mechanisms for the single-phase metering chip to send a signal of a certain bit length mainly include two types:

- After switching to a specific mode (such as switching to HSTX output), the single-phase metering chip mainly sends a signal of a certain bit length through the TX pin;
- The MCU host accesses the single-phase metering chip's 0x00 address register through a register read instruction.

Features of this mode:

- ✓ Set a signal measurement switch; after enabling, use the system clock to measure the length of the signal received by RX.
- ✓ The signal measurement start point and end point can be configured as the rising edge or falling edge of the RX signal.
- ✓ For the two trigger mechanisms of the single-phase chip, the signal length measurement mode is designed as two types: continuous measurement mode and fixed data measurement mode.
- ✓ Continuous measurement mode: When each signal arrives, start measuring the signal length; after one measurement ends, send an interrupt and update the measurement value register.
- ✓ Fixed data measurement mode: When the signal arrives, start measuring the signal length; after one measurement ends, if the received data is consistent with the value configured in the fixed data configuration register, send an interrupt, stop measuring, and update the measurement value register.

18.6.9.2 UART Metering Mode Usage Scenarios

The main usage scenarios for UART metering mode are as follows:

- MCU Communication with Metering HSTX Module:

After the metering chip MKE101R receives an instruction, it sends pulse data, parameter register data, interrupt data, and waveform data to the MCU chip. The default transmission speed is 894.886KHz. MKS20AT sends ms-level precise low-level signals, with a low-level length of about 2ms and an interval of 8ms.

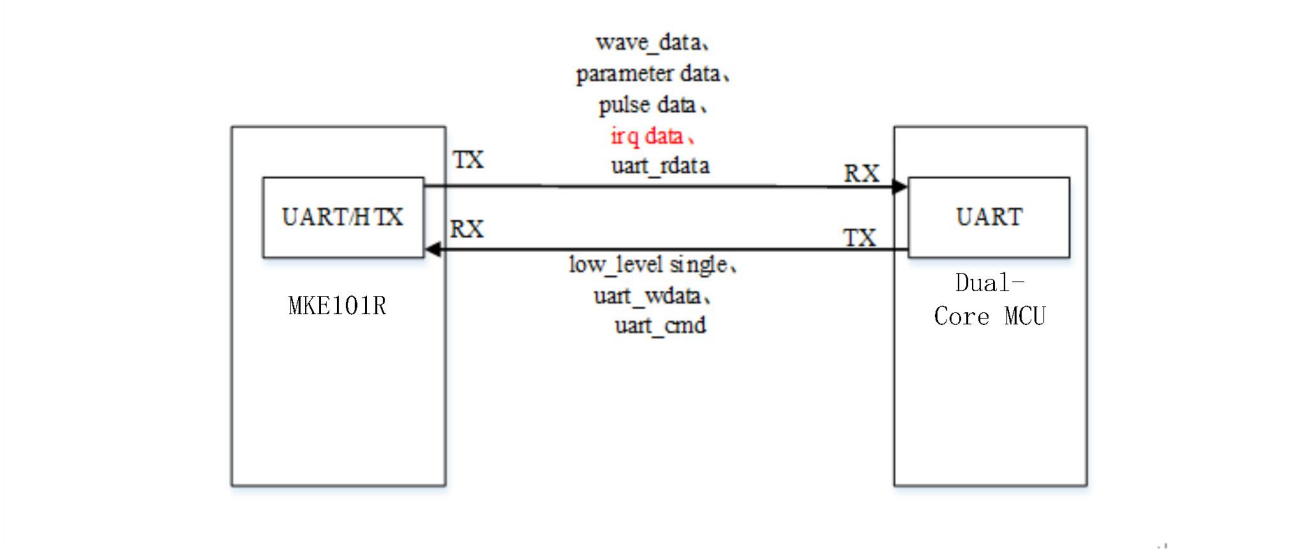


Figure 18-7 UART Metering Mode Application Scenario 1

- Metering Chip Actual Baud Rate Measurement

The MCU's UART can measure the baud rate of the signal received on the RX pin. There are continuous measurement mode and fixed data measurement mode. After the measurement ends, the baud rate count value is saved. The software uses this count value to calculate the baud rate of the UART at the other end of the communication.

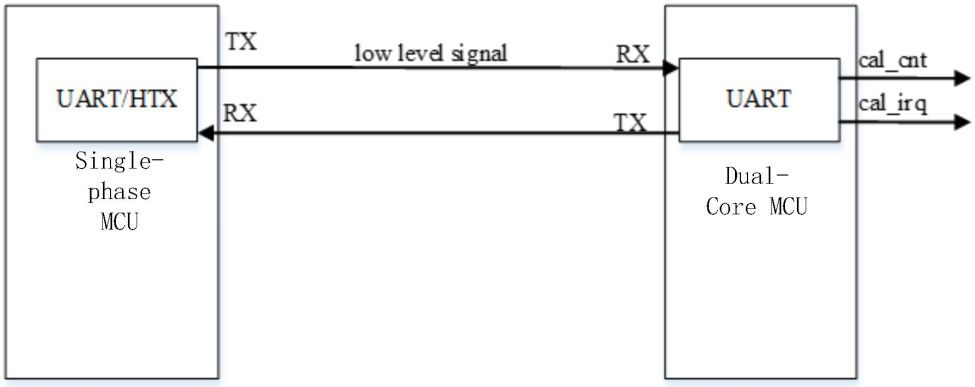


Figure 18-8 UART Metering Mode Application Scenario 2

18.7 Registers

Module Name	Secure Base Address	Non-Secure Base Address
UART0	0x2000_0000	0xA000_0000
UART1	0x2000_4000	0xA000_4000
UART2	0x2000_8000	0xA000_8000
UART3	0x2000_C000	0xA000_C000
UART4	0x2001_0000	0xA001_0000

UART5	0x2001_4000	0xA001_4000
UART6	0x2001_8000	0xA001_8000
UART7	0x2001_C000	0xA001_C000

18.7.1 Register List

Register Name (x=0-7)	Address Offset	Description
UARTx_CTRL	0x00	UARTx Control Register
UARTx_BAUD	0x04	UARTx Baud Rate Configuration Register
UARTx_STA	0x08	UARTx Status Indicator Register
UARTx_TXD	0x0C	UARTx Transmit Data Register
UARTx_RXD	0x10	UARTx Receive Data Register
UARTx_FDIV	0x14	UARTx Baud Rate Fractional Divider Configuration Register
UARTx_DMA_CTRL	0x18	UARTx DMA Control Register
UARTx_DMA_TBADR	0x1C	UARTx DMA Transmit Start Address Register
UARTx_DMA_RBADR	0x20	UARTx DMA Receive Start Address Register
UARTx_DMA_TLEN	0x24	UARTx DMA Transmit Length Register
UARTx_DMA_RLEN	0x28	UARTx DMA Receive Length Register
UARTx_DMA_TADR	0x2C	UARTx Current Transmit DMA Address Register
UARTx_DMA_RADR	0x30	UARTx Current Receive DMA Address Register
UARTx_DMA_IE	0x34	UARTx DMA Interrupt Enable Register
UARTx_DMA_FLG	0x38	UARTx DMA Interrupt Flag Register
UARTx_DMA_TO	0x3C	UARTx Receive Timeout Configuration Register
UART_TX_DELAY	0x40	UARTx Transmit Delay Configuration Register (This register offset address is 0x40 for UART0-5, 0x6C for UART6-7)

The following registers are only supported by UART6, UART7

Register Name (i=6-7)	Address Offset	Description
UARTi_CF_CTRL	0x40	UART_CF Mode Control Register
UARTi_PFDATA	0x44	UART_CF Mode PF Pulse Sequence Register
UARTi_QFDATA	0x48	UART_CF Mode QF Pulse Sequence Register
UARTi_FPFDATA	0x4C	UART_CF Mode FPF Pulse Sequence Register
UARTi_IRQDATA	0x50	UART_CF Mode Interrupt Sequence Register

UARTi_CF_OV	0x54	UART_CF Mode Timeout Control Register
UARTi_TW_CTRL	0x58	UART_TW Mode Control Register
UARTi_TW_BAUD	0x5C	UART_TW Mode Baud Rate Setting Register
UARTi_TW_EN	0x60	UART_TW Mode Start Register
UARTi_CAL_CTRL	0x64	UART_CAL Low Level Length Count Configuration Register
UARTi_CAL_CNT	0x68	UART_CAL Low Level Length Count Value Register
UART_TX_DELAY	0x6C	UARTi Transmit Delay Configuration Register (This register offset address is 0x40 for UART0-5, 0x6C for UART6-7)

18.7.2 UARTx_CTRL（0x00）

UART0-UART5 UART_CTRL Register (Supports Bitband Operation)

Bit	Name	Description	R/W	Reset Value
31:23	---	Reserved	R	0
16	RX_TO_IE	Receive Timeout Interrupt Enable 0: Disable interrupt 1: Enable interrupt	RW	0
15	PAR_MEM	Configuration for data frame not stored in MEM upon receive parity error 1: When receive parity error occurs, data frame is not stored in MEM 0: Data storage in MEM is independent of receive parity bit error	RW	0
14	NEG	Data Polarity Selection Bit 0: Positive polarity, i.e., default drive level is high, transmit/receive data polarity unchanged 1: Negative polarity, i.e., default drive level is low, transmit/receive data inverted	RW	0
13	LMSB	LSB/MSB Transmission Mechanism Selection 0: LSB first 1: MSB first Note: When PARS selects user-defined parity, the parity bit is treated as the extended highest bit of data. If MSB mode is selected at this time, the parity bit will be transmitted first.	RW	0
12	IRSEL	Infrared Modulation Polarity Selection Bit: 0: Positive polarity, i.e., default drive level remains high, low level modulation output 1: Negative polarity, i.e., default drive level remains low, high level modulation output	RW	0

		Note: IRSEL only determines the level during idle output (invalid level) and does not affect the level during valid data periods.		
11	ILBE	Internal Loopback Enable Bit 0: Internal loopback disabled 1: Internal loopback enabled, TXD and RXD signals short-circuited internally within the module	RW	0
10	IRE	Infrared Modulation Enable Bit 0: Disable infrared modulation output 1: Enable infrared modulation output, modulate the low level of output data with 38k carrier Note: Before enabling this bit, the UART_CLK of the system control module must be correctly configured to provide the 38k infrared modulation clock.	RW	0
9:7	PARS	Parity Bit Selection Bit 000: No parity 001: Odd parity 010: Even parity 011: Fixed 0 parity 100: Fixed 1 parity Others: User-defined parity	RW	0
6:5	DATLEN	Transmit Data Width Bit 00: 5-bit 01: 6-bit 10: 7-bit 11: 8-bit	RW	0
4	STOPS	Stop Bit Width Selection 0: 1-bit stop bit 1: 2-bit stop bits	RW	0
3	ERRIE	Error Interrupt Enable Bit, corresponding flags are bits 5~2 in the Status Indicator Register. 0: Disable interrupt 1: Enable interrupt	RW	0
2	RXIE	Receive Data Interrupt Enable Bit, corresponding flag is bit 1 in the Status Indicator Register. 0: Disable interrupt 1: Enable interrupt	RW	0
1	TXIE	Transmit Data Interrupt Enable Bit, corresponding flag is bit 0 in the Status Indicator Register. 0: Disable interrupt 1: Enable interrupt	RW	0
0	EN	UARTx Module Enable Bit 0: Disable	RW	0

		1: Enable		

UART6,7 UART_CTRL Register (Supports Bitband)

Bit	Name	Description	R/W	Reset Value
31:24	---	Reserved	R	0
23	PAR_MEM	Setting for data frame not stored in MEM upon receive parity error 1: Receive parity error, current frame data not stored in MEM 0: Data storage in MEM is independent of receive parity error	RW	0
22	RX_TO_IE	Receive Timeout Interrupt Enable 0: Disable interrupt 1: Enable interrupt	RW	0
21	RX_IRQ_IE	Interrupt Data Received Interrupt Enable	RW	0
20	UART_CAL_IE	Signal Measurement Completion Interrupt Enable	RW	0
19	UART_TW_IE	Low Level Signal Transmission Completion Interrupt Enable	RW	0
18	UART_CF_Err_IE	UART_CF Mode Timeout Error Interrupt Enable	RW	0
17	UART_FPF_IE	FPF Pulse Received Interrupt Enable	RW	0
16	UART_QF_IE	QF Pulse Received Interrupt Enable	RW	0
15	UART_PF_IE	PF Pulse Received Interrupt Enable	RW	0
14	NEG	Data Polarity Selection Bit 0: Positive polarity, i.e., default drive level is high, transmit/receive data polarity unchanged 1: Negative polarity, i.e., default drive level is low, transmit/receive data inverted	RW	0
13	LMSB	LSB/MSB Transmission Mechanism Selection 0: LSB first 1: MSB first Note: When PARS selects user-defined parity, the parity bit is treated as the extended highest bit of data. If MSB mode is selected at this time, the parity bit will be transmitted first.	RW	0
12	IRSEL	Infrared Modulation Polarity Selection Bit: 0: Positive polarity, i.e., default drive level remains high, low level modulation output 1: Negative polarity, i.e., default drive level remains low, high level modulation output Note: IRSEL only determines the level during idle output (invalid level) and does not affect the level during valid data periods.	RW	0
11	ILBE	Internal Loopback Enable Bit 0: Internal loopback disabled 1: Internal loopback enabled, TXD and RXD signals short-circuited internally within the module	RW	0

10	IRE	Infrared Modulation Enable Bit 0: Disable infrared modulation output 1: Enable infrared modulation output, modulate the low level of output data with 38k carrier Note: Before enabling this bit, the UART_CLK of the system control module must be correctly configured to provide the 38k infrared modulation clock.	RW	0
9:7	PARS	Parity Bit Selection Bit 000: No parity 001: Odd parity 010: Even parity 011: Fixed 0 parity 100: Fixed 1 parity Others: User-defined parity	RW	0
6:5	DATLEN	Transmit Data Width Bit 00: 5-bit 01: 6-bit 10: 7-bit 11: 8-bit	RW	0
4	STOPS	Stop Bit Width Selection 0: 1-bit stop bit 1: 2-bit stop bits	RW	0
3	ERRIE	Error Interrupt Enable Bit, corresponding flags are bits 5~2 in the Status Indicator Register. 0: Disable interrupt 1: Enable interrupt	RW	0
2	RXIE	Receive Data Interrupt Enable Bit, corresponding flag is bit 1 in the Status Indicator Register. 0: Disable interrupt 1: Enable interrupt	RW	0
1	TXIE	Transmit Data Interrupt Enable Bit, corresponding flag is bit 0 in the Status Indicator Register. 0: Disable interrupt 1: Enable interrupt	RW	0
0	EN	UARTx Module Enable Bit 0: Disable 1: Enable	RW	0

18.7.3 UARTx_BAUD（0x04）

Baud Rate Configuration Register (Supports Bitband Operation)

Bit	Name	Description	R/W	Reset Value
31:14	---	Reserved	R	0

13:0	CLKDIV	UARTx Clock Division Number Baud rate calculation formula is: System Clock / [16*(CLKDIV+1)] Supports 300 baud rate at 58M.	RW	0

18.7.4 UARTx_STA（0x08）

UARTx Status Indicator Register (x=0-7)

Bit15-Bit21 only supported by UART6,7

Bit	Name	Description	R/W	Reset Value
31:22	---	Reserved	R	0
21	RX_IRQ_IF	Interrupt Data Received Interrupt Flag, write 1 to clear	RW	0
20	UART_CAL_IF	Signal Measurement Completion Interrupt Flag, write 1 to clear	RW	0
19	UART_TW_IF	Low Level Signal Transmission Completion Interrupt Flag, write 1 to clear	RW	0
18	UART_CF_Err_IE	UART_CF Mode Timeout Error Interrupt Flag, write 1 to clear	RW	0
17	UART_FPF_IF	FPF Pulse Received Interrupt Flag, write 1 to clear	RW	0
16	UART_QF_IF	QF Pulse Received Interrupt Flag, write 1 to clear	RW	0
15	UART_PF_IF	PF Pulse Received Interrupt Flag, write 1 to clear	RW	0
14:11	---	Reserved	R	0
10	Rx_to	Receive Timeout Flag. If UART receives a timeout, this bit is set. Write "1" to clear this bit. 0: No timeout occurs 1: Timeout occurs	RW1c	0
9	tx_fifo_full	Transmit FIFO Full: 0: Not full 1: Full	R	0
8	tx_fifo_empty	Transmit FIFO Empty: 0: Not empty 1: Empty	R	1
7	TB	Transmit Status Flag Bit 0: Not transmitting 1: Transmitting data	R	0
6	RB	Receive Status Flag Bit 0: Not receiving 1: Receiving data	R	0
5	DE	Data Error Flag Bit, write 1 to clear UART Transmit FIFO is full but UART transmit register is written again, or new transmit data is written during transmission, this bit will be set. 0: No error	RW	0

		1: Error exists		
4	FE	Frame Format Error Flag Bit, write 1 to clear UART received data does not conform to frame format flag, i.e., the received stop bit is 0 instead of 1, this bit will be set. 0: No error 1: Error exists	RW	0
3	OE	Overflow Error Flag Bit, write 1 to clear UART receive data register is not read in time causing receive overflow, this bit will be set. 0: No error 1: Error exists	RW	0
2	PE	Parity Error Flag Bit, write 1 to clear UART received data parity bit error, this bit will be set. 0: No error 1: Error exists	RW	0
1	TX	UART Data Transmission Completion Flag Bit, write 1 to clear 0: Data not yet transmitted or no pending data to transmit 1: Data has been transmitted	RW	0
0	RX	UART Data Reception Completion Flag Bit, write 1 to clear 0: No received data 1: Data has been received	RW	0

Note: UARTx_CTRL and UARTx_BAUD should not be modified during transmission (TXBUSY/RXBUSY).

18.7.5 UARTx_TXD（0x0C）

UARTx Transmit Data Register (x=0-7)

Bit	Name	Description	R/W	Reset Value
31:9	---	Reserved	R	0
8	UP	User-defined Parity Bit	RW	0
7:0	TXDATA	Transmit Data Register	RW	0

18.7.6 UARTx_RXD（0x10）

UARTx Receive Data Register(x=0-7)

Bit	Name	Description	R/W	Reset Value
31:9	---	Reserved	R	0
8	UP	Parity Bit	R	0
7:0	RXDATA	Receive Data Register	R	0

18.7.7 UARTx_FDIV（0x14）

UARTx Baud Rate Fractional Divider Configuration Register (x=0-7) (Supports Bitband Operation)

Bit	Name	Description	R/W	Reset Value
31:14	---	Reserved	R	0
13:0	FDIV	Fractional Division Coefficient. Calculation formula is: $F = \left[\left(\frac{fi}{16 \times fo} - \left[\frac{fi}{16 \times fo}\right]\right) \times 2^{14} + 0.5\right]$ Where fi is the system clock, fo is the output clock, "[ ]" is the floor function. For example, system clock is 16.384MHz, output clock is 115200Hz, then: $\mathrm{FDIV} = \left[\left(\frac{16384000}{16\times115200} - \left[\frac{16384000}{16\times115200}\right]\right) \times 2^{14} + 0.5\right]$ Result FDIV=14564. Used in conjunction with the BAUD register, the integer part is configured via BAUD. Can achieve any baud rate output under the same system clock, and output fixed baud rate when the system clock changes.	RW	0

18.7.8 UARTx_DMA_CTL（0x18）

UARTx Module DMA Enable Control Register (x=0-7) (Bit0-1 support Bitband operation)

Bit	Name	Description	R/W	Reset Value
31:5	Reserved	Reserved	R	0
4	DMA_PASS	DMA Parity Bit	RW	0
3	RX_CYC_MODE	Receive Cycle Mode Enable	RW	0
2	TX_CYC_MODE	Transmit Cycle Mode Enable	RW	0
1	RX_DMA_EN	Receive DMA Enable	RW	0
0	TX_DMA_EN	Transmit DMA Enable	RW	0

18.7.9 UARTx_DMA_TBADR（0x1C）

UARTx Module DMA Transmit Start Address Register(x=0-7)

Bit	Name	Description	R/W	Reset Value
31:18	Reserved	Reserved	R	0
17:0	DMA_TBADR	DMA Transmit Start Address (Byte address) The lowest two bits must be configured according to the specific interface data bit width.	RW	0

18.7.10 UARTx_DMA_RBADR（0x20）

UARTx Module DMA Receive Start Address Register(x=0-7)

Bit	Name	Description	R/W	Reset Value
31:18	Reserved	Reserved	R	0

17:0	DMA_RBADR	DMA Receive Start Address (Byte address) The lowest two bits must be configured according to the specific interface data bit width.	RW	0

18.7.11 UARTx_DMA_TLEN（0x24）

UARTx Module DMA Transmit Length Register(x=0-7)

Bit	Name	Description	R/W	Reset Value
31:18	Reserved	Reserved	R	0
17:0	DMA_TLEN	DMA Transmit Length = (n) Byte The lowest two bits must be configured according to the specific interface data bit width.	RW	0

18.7.12 UARTx_DMA_RLEN（0x28）

UARTx Module DMA Receive Length Register(x=0-7)

Bit	Name	Description	R/W	Reset Value
31:18	Reserved	Reserved	R	0
17:0	DMA_RLEN	DMA Receive Length = (n) Byte The lowest two bits must be configured according to the specific interface data bit width.	RW	0

18.7.13 UARTx_DMA_TADR（0x2C）

UARTx Module Current Transmit DMA Address Register (x=0-7)

Bit	Name	Description	R/W	Reset Value
31:18	Reserved	Reserved	R	0
17:0	DMA_TADR	Current Transmit DMA Address (Byte address)	RO	0

18.7.14 UARTx_DMA_RADR（0x30）

UARTx Module Current Receive DMA Address Register (x=0-7)

Bit	Name	Description	R/W	Reset Value
31:18	Reserved	Reserved	R	0
17:0	DMA_RADR	Current Receive DMA Address (Byte address)	RO	0

18.7.15 UARTx_DMA_IE（0x34）

UARTx DMA Interrupt Enable Register(x=0-7)

Bit	Name	Description	R/W	Reset Value
31:7	Reserved	Reserved	R	0
6	RX_DONE_IE	DMA Receive Completion Interrupt Enable	RW	0
5	RX_ERR_IE	Receive Data Error Interrupt Enable	RW	0

4	TX_ERR_IE	Transmit Data Error Interrupt Enable	RW	0
3	RX_FIE	DMA Receive Full Interrupt Enable	RW	0
2	RX_HIE	DMA Receive Half Full Interrupt Enable	RW	0
1	TX_FIE	DMA Transmit Empty Interrupt Enable	RW	0
0	TX_HIE	DMA Transmit Half Empty Interrupt Enable	RW	0

18.7.16 UARTx_DMA_FLG（0x38）

UARTx DMA Interrupt Flag Register (x=0-7)

Bit	Name	Description	R/W	Reset Value
31:7	Reserved	Reserved	R	0
6	RX_DONE	DMA Receive Completion, this flag is set, write "1" to clear	RW	0
5	RX_ERR	Receive Data Error Flag, write "1" to clear	RW	0
4	--	--Reserved--	RW	0
3	RX_FDONE	DMA Receive Full Interrupt Flag, write "1" to clear	RW	0
2	RX_HDONE	DMA Receive Half Full Interrupt Flag, write "1" to clear	RW	0
1	TX_FDONE	DMA Transmit Empty Interrupt Flag, write "1" to clear	RW	0
0	TX_HDONE	DMA Transmit Half Empty Interrupt Flag, write "1" to clear	RW	0

18.7.17 UARTx_DMA_TO（0x3C）

UARTx Receive Timeout Configuration Register(x=0-7)

Bit	Name	Description	R/W	Reset Value
31:8	Reserved	Reserved	R	0
7:0	TIMEOUT_CNT	After receiving the UART start bit, if UART does not receive data for the time length defined by this register, it indicates a receive timeout. n = n UART data bit width times. Configuration of 0 is invalid.	RW	0

18.7.18 UARTx_TX_DRLAY（UART0-5 Relative Offset: 0x40）

UARTx Transmit Delay Configuration Register (x=0-5)

Bit	Name	Description	R/W	Reset Value
31:3	Reserved	Reserved	R	0
2	TX_DELAY_EN	Transmit Delay Function Enable 1: Enable this function 0: Disable this function	RW	0
1:0	TIMEOUT_CNT	2'd0: Insert 1 UART data bit width time 2'd1: Insert 2 UART data bit width times 2'd2: Insert 4 UART data bit width times	RW	0

		2'd3: Insert 8 UART data bit width times Valid when TX_DELAY_EN=1		

Note: Insert delay between each data frame.

18.7.19 UART_CF_CTL（0x40）

UART_CF Mode Control Register

Bit	Name	Description	R/W	Reset Value
31:3	Reserved	Reserved	R	0
2	PULSE_OE	UART_CF Mode Pulse Output Enable (Configurable when UART_CF_EN is 0) 0: Receive metering pulse data interrupt set but no pulse sent 1: Receive metering pulse data interrupt set and pulse sent to IOCNT	RW	1
1	IRQ_WR_EN	UART_CF Mode Interrupt Data Write to SRAM Enable (Configurable when UART_CF_EN is 0) 0: Receive metering interrupt data interrupt set but data not written to SRAM 1: Receive metering interrupt data interrupt set and data written to SRAM via DMA	RW	0
0	UART_CF_EN	UART_CF Mode Enable 0: Disable 1: Enter UART_CF mode	RW	0

18.7.20 UART_PFDAT（0x44）

UART_CF Mode PF Pulse Sequence Register

Bit	Name	Description	R/W	Reset Value
31:0	PF_DATA	PF Pulse Sequence Value that UART needs to recognize	RW	0x800A20AA

Configurable when UART_CF_EN is 0. In UART_CF mode, receiving this data (requires each byte parity bit error) is recognized as a PF pulse.

18.7.21 UART_QFDAT（0x48）

UART_CF Mode QF Pulse Sequence Register

Bit	Name	Description	R/W	Reset Value
31:0	QF_DATA	QF Pulse Sequence Value that UART needs to recognize	RW	0x800C40CC

Configurable when UART_CF_EN is 0. In UART_CF mode, receiving this data (requires each byte parity bit error) is recognized as a QF pulse.

18.7.22 UART_FPFDAT（0x4C）

UART_CF Mode FPF Pulse Sequence Register

Bit	Name	Description	R/W	Reset Value
31:0	FPF_DATA	FPF Pulse Sequence Value that UART needs to recognize	RW	0x800E60EE

This register can only be configured when UART_CF_EN is 0. In UART_CF mode, receiving this data (requires each byte parity bit error) is recognized as an FPF pulse.

Read/Write Attribute

18.7.23 UART_IRQDAT（0x50）

UART_CF Mode Interrupt Sequence Register

Bit	Name	Description	R/W	Reset Value
31:0	IRQ_DATA	The interrupt data values to be identified by UART	RW	0x800820A8

Configurable when UART_CF_EN is 0. In UART_CF mode, receiving this data (requires each byte parity bit error) recognizes the single-phase chip interrupt pull-up, and this module generates the corresponding interrupt.
UART_IRQDAT value cannot be equal to UART_PFDAT, UART_QFDAT, UART_FPFDAT register values.

18.7.24 UART_CF_OV（0x54）

UART_CF Mode Timeout Control Register. This register can only be configured when UART_CF_EN is 0.

Bit	Name	Description	R/W	Reset Value
31:3	Reserved	Reserved	R	0
2	OV_BUF_CLR	Timeout Cache Clear Enable. When UART_CF mode timeout occurs, if this register configuration value is 1, the identified partial pulse data is written to SRAM.	RW	0
1:0	CF_OV_TIME	UART_CF Mode Timeout Configuration, time calculated using 894k baud rate: =00: Do not start timeout mechanism =01: Timeout 22 bits error (approx. 24.584us); =10: Timeout 44 bits error (approx. 49.168us); =11: Timeout 88 bits error (approx. 98.336us);	RW	0

18.7.25 UART_TW_CTL（0x58）

UART_TW Mode Control Register

Bit	Name	Description	R/W	Reset Value
31:5	Reserved	Reserved	R	0
4:1	TW_NUM	Number of low levels to send is (TW_NUM+1)	RW	0
0	UART_TW_MODE	UART_TW Mode Enable Configuration	RW	0

After configuring UART_TW mode enable, switch UART TX function to transmit low-level waveform.

18.7.26 UART_TW_BAUD（0x5C）

UART_TW Mode Baud Rate Setting Register

Bit	Name	Description	R/W	Reset Value
31:14	Reserved	Reserved	R	0
13:0	TW_BAUD	TW Mode Transmit Each Bit Length Configuration	RW	0xBFF

This register cannot be modified during transmission. Default value is the 4800 baud rate configuration value under 14M system clock.

18.7.27 UART_TW_EN（0x60）

UART_TW Mode Start Register

Bit	Name	Description	R/W	Reset Value
31:1	Reserved	Reserved	R	0
0	UART_TW_KICK	TX Waveform Transmission Start Signal. Write 1 to start transmitting waveform when in use, writing during transmission is invalid. This register reads as 1 during transmission, and automatically clears to 0 after transmission is complete.	RW	0

18.7.28 UART_CAL_CTL（0x64）

Receive Low Level Length Count Configuration Register

Bit	Name	Description	R/W	Reset Value
31:13	Reserved	Reserved	R	0
12:4	CAL_DATA	Data required for measurement value register update in fixed data measurement mode	RW	0
3	POS_END	=1; Measurement end point is RX rising edge =0: Measurement end point is RX falling edge	RW	0
2	POS_START	=1; Measurement start point is RX rising edge =0: Measurement start point is RX falling edge	RW	0
1	CAL_MODE	Measurement Mode Selection: =0: Continuous measurement mode; =1: Fixed data measurement mode	RW	0
0	UART_CAL_EN	Measurement Mode Enable	RW	0

After UART_CAL_EN is enabled, signal measurement on the RX pin input starts.

When CAL_MODE is 0, after measuring each signal that meets the measurement requirements, update the measurement value register and generate an interrupt. UART_CAL_EN remains 1. Write 0 to UART_CAL_EN to stop measurement.

When CAL_MODE is 1, after measuring the signal that meets the measurement requirements and receiving the communication stop bit, if the received data is consistent with CAL_DATA, then update the measurement value register, generate an interrupt, stop measurement, and UART_CAL_EN automatically resets to 0. If the received data is inconsistent with CAL_DATA, continue measurement, and UART_CAL_EN remains 1. This mode is only

valid in 8-bit communication.

18.7.29 UART_CAL_CNT（0x68）

Receive Low Level Length Count Value Register

Bit	Name	Description	R/W	Reset Value
31:0	UART_CAL_CNT	Measurement Result Register Continuous measurement mode: Update UART_CAL_CNT register after each low-level measurement completion; Fixed bit measurement mode: Update UART_CAL_CNT register after receiving and measuring the configured CAL_bit_CNT length low level.	R	0

18.7.30 UARTx_TX_DRLAY（UART6-7 Relative Offset: 0x6C）

UARTx Transmit Delay Configuration Register(x=6、7)

Bit	Name	Description	R/W	Reset Value
31:3	Reserved	Reserved	R	0
2	TX_DELAY_EN	Transmit Delay Function Enable 1: Enable this function 0: Disable this function	RW	0
1:0	TIMEOUT_CNT	2'd0: Insert 1 UART data bit width time 2'd1: Insert 2 UART data bit width times 2'd2: Insert 4 UART data bit width times 2'd3: Insert 8 UART data bit width times Valid when TX_DELAY_EN=1	RW	0

Note: Insert delay between each data frame.

18.8 Software Configuration Flow and Application Instructions

18.8.1 UART Security Attribute Configuration

- Write the password 0xA5A5 to the SCU password protection bit to make other registers writable;
- Set the security attributes of the peripheral (PERIPH_SEC_CFG1 register);
- Set the security attributes of GPIOi (GPIOi_SEC_CFG register);
- Lock SCU writes; after locking, the SCU security attribute configuration registers cannot be written until the next reset.

18.8.2 UART Conventional Function Configuration Flow

- Enable the gated clock for UART and the gated clock for the corresponding multiplexed GPIO;
- Perform port multiplexing; note that the security attributes of UART and GPIO must be consistent during multiplexing;
- Configure the corresponding attributes of UART (UARTx_CTRL register) and set the baud rate;
- Enable the corresponding UARTx interrupts;

- Enable UARTx;
- Transmit or receive data;
- All three types of UART interrupts enter the UARTx_HANDLER interrupt. After entering the interrupt, the cause of the interrupt can be determined by querying the corresponding flag bits.

18.8.3 UART Infrared Modulation Function Configuration Flow

- Enable the UARTx infrared modulation function;
- Configure infrared modulation attributes and turn on the gated clock for infrared modulation;
- Refer to part 2 for conventional UART operations.

18.8.4 UART Dedicated DMA Function

- Refer to part 2 for conventional UART operations;
- Set the UART_DMA_CTL register to enable/disable DMA transmission/reception, cycle mode, and checksum function;
- Fill the source address of the transmit data and the length of the transmit data into the UARTx_DMA_TBADR and UARTx_DMA_TLEN registers;
- Fill the destination address of the receive data and the length of the receive data into the UARTx_DMA_RBADR and UARTx_DMA_RLEN registers;
- Configure DMA interrupt enable and DMA timeout settings as needed;
- Enable UARTx and UARTx interrupts.

18.8.5 UART AOS Function

- UART AOS Function Configuration Flow:
- Configure the security attributes of UART according to the requirements, and enable the linkage enable for the corresponding security area;
- Set up according to the UART conventional communication and IOCNT sections respectively;
- Set the number corresponding to UART6-7 as the trigger source, and set the trigger source of the AOS IOCNT corresponding register (PE_TRGSEL25-TRGSEL29) to the specified UARTx trigger signal;
- Set trigger signal detection;
- Enable AOS;

18.8.6 UART and LSRX Linkage Usage

18.8.6.1 Calibration Waveform Output

- Configure the LSRX calibration output configuration register according to requirements, select the output port, configure the output special character and low-level length;
- Configure the LSRX interrupt enable register to enable the calibration waveform transmission completion interrupt;
- Configure the LSRX enable register to enable the LSRX module;
- Configure the LSRX calibration output configuration register; after configuration is complete, start transmission;
- Wait for the interrupt to be raised, completing one calibration waveform output;
- Read the Single Phase Meter IC's RCFreq register or obtain the calibration result through LSTX reporting; based on this result, the internal RC frequency of a special-chip can be calculated.

19 TC

19.1 Introduction

The MCU has two built-in 32-bit timers, each capable of operating completely independently. The timers do not share any resources and can operate synchronously. The counting timer primarily has four uses: timing tracking and action triggering, event counting and action triggering, signal demodulation, and signal modulation and output. The timer has multiple functions, such as being used as an interval timer, square wave output, external/internal event counter, single pulse output, PWM output, and pulse width measurement, etc.

19.2 Features

- Two 32-bit timers, each timer:
  - Has one 32-bit incrementing auto-reload counter;
  - Has a 16-bit programmable prescaler with selectable division factor from 1 to 65535;
  - Supports 32-bit timing function;
  - Supports dynamic access to count value;
  - Supports 32-bit counting function;
  - Supports free-running mode;
  - Supports single-run operation.
- Each timer has 2 capture/compare channels, each channel can be independently configured as:
  - Input capture;
  - Output compare;
  - PWM generation (single slope);
  - Single pulse output;
  - Complementary PWM with configurable dead-time insertion length;
  - PWM of Channel 1 supports general-purpose DMA.
- Programmable dead-time length:
  - Dead-time length for both edges can be set independently:
  - 6-bit timing length;
  - Output polarity configurable.
- Configurable fault handling:
  - Output disable;
  - Output clear;
  - Output tri-state.
- Slave mode support:
  - External reset and restart;
  - External gating.
- Input capture support:
  - Rising edge capture;
  - Falling edge capture;
  - Dual-edge capture;
  - Period measurement;
  - Pulse width measurement;
  - Optional filtering.

- Output compare support:
  - Tri-state output;
  - Inverted output;
  - Fixed level output;
  - Pulse output with configurable width;
  - Compare register update anytime.
- Interrupt support:
  - Count overflow;
  - Input capture;
  - Output compare.

19.3 Functional Block Diagram

Please refer to the figure below for the functional block diagram of the counting timer. Each counting timer contains a 32-bit counter and four 32-bit capture/compare channels.

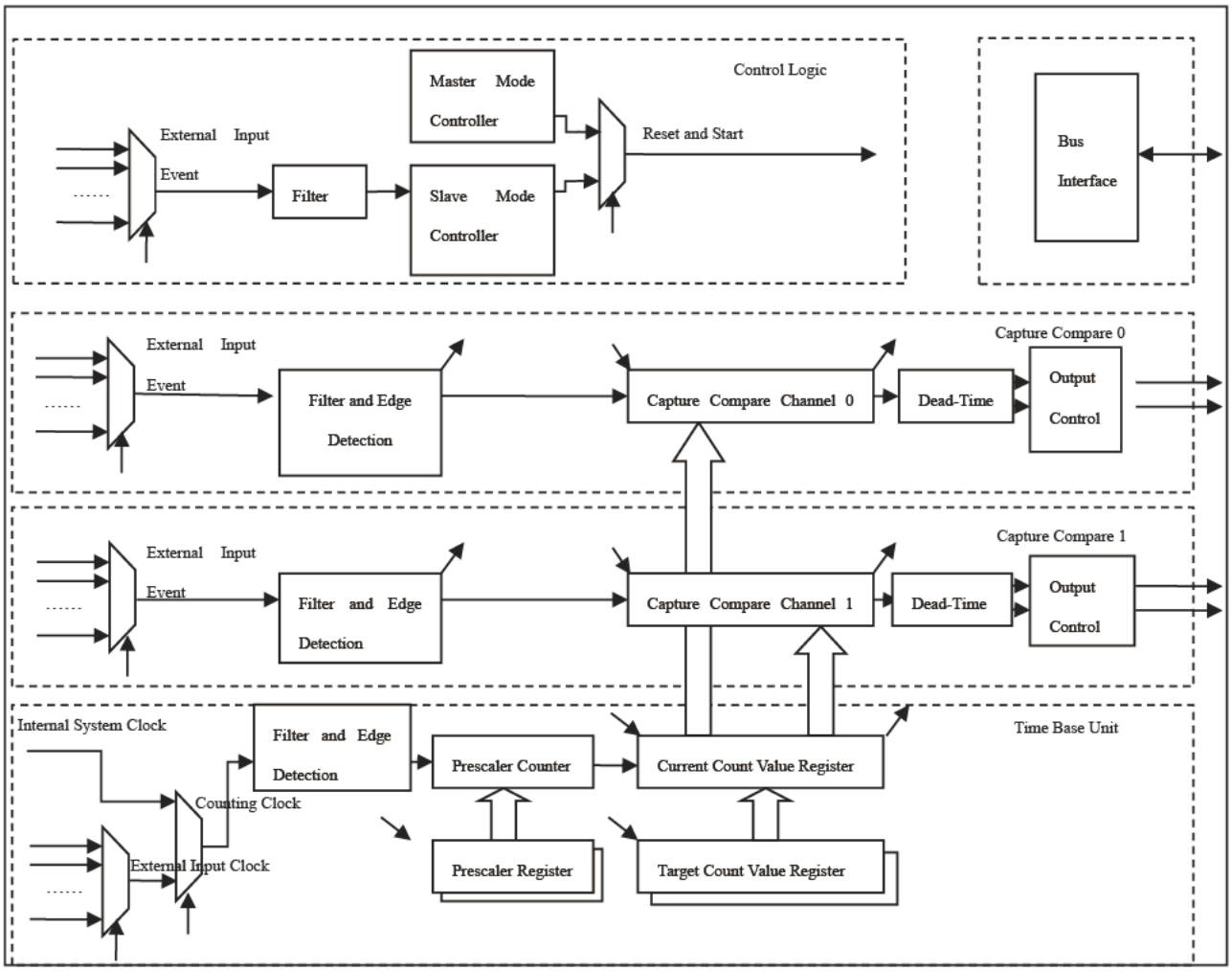


Figure 19-1 Functional Block Diagram of Counting Timer

19.4 Register Description

Register Base Address of TC Module

Module Name	Core0 Base Address	Core1 Base Address
TC0	0x2009_C000	0xA009_C000
TC1	0x200A_0000	0xA00A_0000

Register Offset Address of TC Module

Register Name	Address Offset	Description
TC_CNT	0x0	Current Count Value Indicator
TC_PS	0x4	Prescaler Register
TC_DN	0xC	Target Count Value Register
TC_CCD0	0x14	Capture Compare Channel 0 Data Register
TC_CCD1	0x18	Capture Compare Channel 1 Data Register
TC_CCFG	0x1C	Clock Configuration Register
TC_CTRL	0x20	Control Register
TC_CM0	0x24	Capture Compare Channel 0 Mode Register
TC_CM1	0x28	Capture Compare Channel 1 Mode Register
TC_IE	0x2C	Interrupt Enable Register
TC_STA	0x30	Status Register
TC_PWM_CFG	0x34	PWM Configuration Register
TC_PWM_CTL	0x38	PWM Control Register
TC_PWM_STA	0x3C	PWM Status Register
TC_PWM_CNT	0x40	PWM Parameter Register
TC_PWM_BADR	0x44	PWM DMA Start Address Register
TC_PWM_DMA_LEN	0x48	PWM DMA Data Length Register
TC_PWM_ADR	0x4C	PWM DMA Current Address Register

19.4.1 TC_CNT（0x00）

Current Count Value Register

Offset Address = 0x00

Bit	Name	Description	R/W	Reset Value
31:0	CNT	Current Count Value	RO	0

19.4.2 TC_PS（0x04）

Prescaler Register

Offset Address = 0x04

Bit	Name	Description	R/W	Reset Value

31:16	--	Reserved	RO	0
15:0	PS	Division factor, division value (PS+1), 0 means no division	RW	0

19.4.3 TC_DN（0x0C）

Target Count Value Register

Offset Address = 0x0C

Bit	Name	Description	R/W	Reset Value
31:0	DN	Target count value, actual counting clock cycles is DN+1	RW	0

19.4.4 TC_CCD0（0x14）

Capture Compare Channel 0 Data Register

Offset Address = 0x14

Bit	Name	Description	R/W	Reset Value
31:0	CCD	Capture Compare Data	RW	0

Note: When Channel 0 is configured for capture function (i.e., the CCM bit field of the TC_CM0 register is 0), the TC_CCD0 register is not writable.

19.4.5 TC_CCD1（0x18）

Capture Compare Channel 1 Data Register

Offset Address = 0x18

Bit	Name	Description	R/W	Reset Value
31:0	CCD	Capture Compare Data	RW	0

Note: When Channel 1 is configured for capture function (i.e., the CCM bit field of the TC_CM1 register is 0), the TC_CCD1 register is not writable.

19.4.6 TC_CCFG（0x1C）

Clock Configuration Register

Offset Address = 0x1C

Bit	Name	Description	R/W	Reset Value
31:12	--	Reserved	R	0
11:4	FLTOPT	External input clock filter parameter setting, sets the number of clock cycles for filtering.	RW	0
3:2	ECLKMODE	External input clock mode: 00: Rising edge 01: Falling edge 10: Both edges 11: Reserved (equivalent to both edges)	RW	0
1	FLTEN	External input clock filter enable: 0: Disable	RW	0

		1: Enable		
0	CM	Counting clock source selection: 0: Internal system clock 1: External input clock	RW	0

19.4.7 TC_CTRL（0x20）

Control Register

Offset Address = 0x20

Bit	Name	Description	R/W	Reset Value
31:18	--	Reserved	R	0
17	DBGSTBDIS	Counter count enable during debugging: 0: Disable (counter pauses counting when CPU is in debug state) 1: Enable (counter continues counting when CPU is in debug state) Note: CPU in debug state refers to the user halting the CPU via the debug interface (PC pointer stops incrementing).	RW	0
16	SLVGATELVL	Slave mode gate mode active level: 0: Active level is low 1: Active level is high	RW	0
15	--	Reserved	R	0
14:7	SLVFLTOPT	Slave mode input filter parameter	RW	0
6:5	SLVTRGMODE	Slave mode control mode selection: 00: Rising edge clears internal counter 01: Falling edge clears internal counter 10: Both edges clear internal counter 11: Gate mode (internal counter counts when external input signal is at active level)	RW	0
4	OPS	Single-shot count mode selection: 0: Disable single-shot count mode (counts cyclically after overflow) 1: Enable single-shot count mode (stops after overflow)	RW	0
3	SLVFLTEN	Slave mode external input event filter enable: 0: Disable 1: Enable	RW	0
2	SLVEN	Slave mode enable: 0: Disable 1: Enable	RW	0
1	--	Reserved	R	0
0	START	Timer start: 0: Stop	RW	0

		1: Start		

19.4.8 TC_CM0（0x24）

Capture Compare Channel 0 Mode Register

Offset Address = 0x24

Bit	Name	Description	R/W	Reset Value
31:26	--	Reserved	R	0
25	DFTLVL	Compare output default level: 0: Low level 1: High level	RW	0
24	EFELVL	Compare output effective level: 0: Low level 1: High level	RW	0
23:21	OM	Compare output mode: 000: No output (tri-state) 001: Set to effective level 010: Set to inactive level 011: Toggle 100: Force to effective level 101: Force to inactive level 110: PWM mode 1 111: PWM mode 2	RW	0
20	FLTEN	Capture external input event filter enable: 0: Disable 1: Enable	RW	0
19	--	Reserved	R	0
18:11	FLTOPT	Capture external input event filter parameter	RW	0
10:9	CPOL	Capture external input event polarity selection: 00: Rising edge 01: Falling edge 10: Both edges 11: Reserved	RW	0
8:3	DL	Compare output dead-time length (only valid for PWM mode 1 and PWM mode 2, invalid in other modes)	RW	0
2	DIEN	Compare output dead-time insertion enable (only valid for PWM mode 1 and PWM mode 2, invalid in other modes): 0: Disable 1: Enable	RW	0
1	CCM	Capture compare mode selection: 0: Capture 1: Compare	RW	0
0	ENABLE	Channel enable:	RW	0

		0: Disable 1: Enable		

19.4.9 TC_CM1（0x28）

Capture Compare Channel 1 Mode Register

Offset Address = 0x28

Bit	Name	Description	R/W	Reset Value
31:26	--	Reserved	R	0
25	DFTLVL	Compare output default level: 0: Low level 1: High level	RW	0
24	EFELVL	Compare output effective level: 0: Low level 1: High level	RW	0
23:21	OM	Compare output mode: 000: No output (tri-state) 001: Set to effective level 010: Set to inactive level 011: Toggle 100: Force to effective level 101: Force to inactive level 110: PWM mode 1 111: PWM mode 2	RW	0
20	FLTEN	Capture external input event filter enable: 0: Disable 1: Enable	RW	0
19	--	Reserved	R	0
18:11	FLTOPT	Capture external input event filter parameter	RW	0
10:9	CPOL	Capture external input event polarity selection: 00: Rising edge 01: Falling edge 10: Both edges 11: Reserved	RW	0
8:3	DL	Compare output dead-time length (only valid for PWM mode 1 and PWM mode 2, invalid in other modes)	RW	0
2	DIEN	Compare output dead-time insertion enable (only valid for PWM mode 1 and PWM mode 2, invalid in other modes): 0: Disable 1: Enable	RW	0
1	CCM	Capture compare mode selection: 0: Capture 1: Compare	RW	0

0	ENABLE	Channel enable: 0: Disable 1: Enable	RW	0

19.4.10 TC_IE（0x2C）

Interrupt Enable Register

Offset Address = 0x2C

Bit	Name	Description	R/W	Reset Value
31:4	--	Reserved	R	0
3	SLVIE	Slave mode interrupt enable: 0: Disable 1: Enable	RW	0
2	CC1IE	Capture compare channel 1 interrupt enable: 0: Disable 1: Enable	RW	0
1	CC0IE	Capture compare channel 0 interrupt enable: 0: Disable 1: Enable	RW	0
0	OVIE	Overflow interrupt enable: 0: Disable 1: Enable	RW	0

19.4.11 TC_STA（0x30）

Status Register

Offset Address = 0x30

Bit	Name	Description	R/W	Reset Value
31:4	--	Reserved	R	0
3	SLVF	Slave mode event flag: (Write 1 to clear) 0: No slave mode event 1: Slave mode event occurred	RW	0
2	CC1F	Capture compare channel 1 event flag: (Write 1 to clear) 0: No capture or compare event 1: Capture or compare event occurred	RW	0
1	CC0F	Capture compare channel 0 event flag: (Write 1 to clear) 0: No capture or compare event 1: Capture or compare event occurred	RW	0
0	OVF	Overflow interrupt flag: (Write 1 to clear) 0: No overflow event 1: Overflow event occurred	RW	0

19.4.12 TC_PWM_CFG（0x34）

PWM Control Register, supports bit-band operation. If the control bits of the following registers are configured during waveform generation, the output result will change accordingly. Therefore, do not configure the following registers during waveform generation.

Offset Address = 0x34

Bit	Name	Description	R/W	Reset Value
31:17	--	Reserved	R	0
16:7	WL	PWM wave length, PWM output signal length configuration parameter. Only valid in length-configurable mode. Represents outputting WL+1 complete signal cycles. Example: In DMA mode, using PWM to simulate a sine wave, it means outputting WL+1 cycles of the sine wave, i.e., traversing the DMA data table counts as one complete signal cycle; in software mode, when outputting a square wave, it means outputting WL+1 square waves, i.e., each use of the PWM parameter counts as one complete signal cycle.	RW	0
6	ORV	Output Remain Valid, output signal is always valid. 0: Length-configurable mode 1: Always-valid mode Length-configurable means using the WL parameter; always-valid means outputting the waveform continuously until disabled.	RW	0
5	CSS	Complete Signal selection, signal integrity output selection. 0: After software disables the module, it stops waveform output after completing one full signal output. In DMA mode, DMA length end indicates one complete signal; in software mode, one PWM wave indicates one complete signal. 1: After software disables the module, PWM output stops immediately.	RW	0
4	CMS	Control mode selection, PWM modulation control mode. 0: DMA mode 1: Software control mode	RW	0
3	CYCDONEIE	Output signal cycle completion interrupt enable. 0: Disable 1: Enable	RW	0
2	BUFVLDIE	PWM parameter buffer empty interrupt enable (only used in software mode). 0: Disable 1: Enable	RW	0
1	DATERRIE	Data error interrupt enable. 0: Disable	RW	0

		1: Enable		
0	DONEIE	PWM output completion interrupt enable. 0: Disable 1: Enable	RW	0

19.4.13 TC_PWM_CTL（0x38）

PWM Control Register

Offset Address = 0x38

Bit	Name	Description	R/W	Reset Value
31:1	--	Reserved	R	0
0	EN	PWM module enable bit. 0: PWM modulation off 1: PWM modulation on Only enabled/disabled by software. In DMA mode, when dma_len is 0, after enabling, remember to disable the enable, otherwise it will affect TC function.	RW	0

19.4.14 TC_PWM_STA（0x3C）

PWM Status Indicator Register

Offset Address = 0x3C

Bit	Name	Description	R/W	Reset Value
31:4	--	Reserved	R	0
4	CYCDONE	Output signal cycle completion interrupt. 0: Has not completed one cycle signal output 1: Completed one cycle signal output Write 1 to clear. In DMA mode, set after traversing the DMA table, e.g., when simulating sine wave output, set after one sine wave cycle ends. In software control mode, set after each PWM square wave output completes, i.e., set after each parameter count ends.	RW	0
3	BUFVLD	PWM parameter buffer empty interrupt (only used in software mode). 0: In software control mode, PWM parameters exist in the buffer 1: In software control mode, new PWM parameters can be configured Write 1 to clear. Set at the start of waveform output; set after each PWM parameter count ends.	RW	0
2	DATERR	Data empty error interrupt.	RW	0

		0: Normal data reading 1: In DMA mode, data cannot be read in time due to competition; in software control mode, the user did not write new data in time. Write 1 to clear. In DMA mode, set when the competition time is greater than the set PWM parameter. In software control mode, set if new data is not written before the previous PWM parameter count ends, set after count ends; set after startup if no PWM parameter is written before startup.		
1	DONE	PWM output completion interrupt. 0: Waveform output not completed 1: Waveform output completed Write 1 to clear. Set after WL count completes; when CSS=0, set after waveform output completes following disable; when CSS=1, not set after disable.	RW	0
0	BUSY	PWM signal transmission status. 0: Idle state 1: Waveform generation state	R	0

19.4.15 TC_PWM_CNT（0x40）

PWM Parameter Register

Offset Address = 0x40

Bit	Name	Description	R/W	Reset Value
31:16	--	Reserved	R	0
15:0	N_Pwm	Represents the system clock cycle length of the current PWM period. Actual counting clock cycles is N_Pwm+1. In software control mode, configurable when buffer is empty, configuration overwrites the original value when not empty. In DMA control mode, do not configure this register by software.	RW	0

19.4.16 TC_PWM_BADR（0x44）

PWM Parameter Register

Offset Address = 0x44

Bit	Name	Description	R/W	Reset Value
31:16	--	Reserved	R	0
15:0	BADR	DMA Start Address (Word address)	RW	0

19.4.17 TC_PWM_DMA_LEN（0x48）

PWM DMA Data Length

Offset Address = 0x48

Bit	Name	Description	R/W	Reset Value
31:16	---	Reserved	R	0
15:0	LEN	DMA Data Length, i.e., the length of the PWM parameter table. If configured as n, it indicates length = n Words, and the size of the table is n. It is recommended that the input length LEN > 1; if LEN = 1 is required, the software control mode is recommended. In DMA mode, when LEN = 1, if the length-configurable function is used, the final output length is WL + 2.	RW	0

19.4.18 TC_PWM_ADR（0x4C）

PWM DMA Current Address

Offset Address = 0x4C

Bit	Name	Description	R/W	Reset Value
31:16	--	Reserved	R	0
15:0	ADR	DMA Current Address (Word address)	R	0

19.5 Application Examples

19.5.1 Interval Timing Event

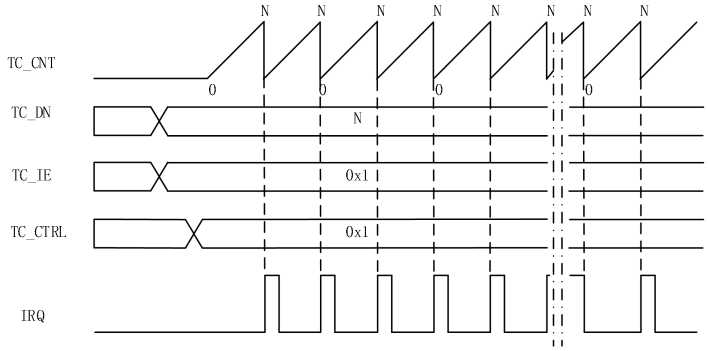


Figure 19-2 Interval Timing Event (Free-Running Mode)

Auto-run mode is the interval timing function.

For the basic timing function, only the following registers need to be set:

- Target Count Value Register, which represents the timing duration, counted by the counting clock.
- Enable the overflow interrupt in the Interrupt Enable Register.
- Control Register, start the timer.

The timer will generate interrupts periodically with the target count value as the cycle.

Common Optional Function Configuration Description:

- The prescaler register value can be modified to change the frequency of the timer counting clock.
- The Clock Configuration Register can be configured, and CM can be modified to external input clock, and the corresponding trigger source of TC in the AOS chapter can be configured. The frequency of the external input clock cannot be higher than half the frequency of the internal system clock.
- Single-count mode: the timer stops after overflow; set the OPS bit in the Control Register to 1 for single-count mode.
- If the external input clock source is the output of another timer, the two timers can be connected in cascade mode to increase the bit width of the timer registers.

19.5.2 Square Wave Output

Square Wave Output

The square wave output function is the function of frequency-dividing the counting clock of TC for output. Each timer has two output channels, and each channel has two output terminals: P and N. Among them, P is the positive output terminal, and N is the inverted output terminal of P.

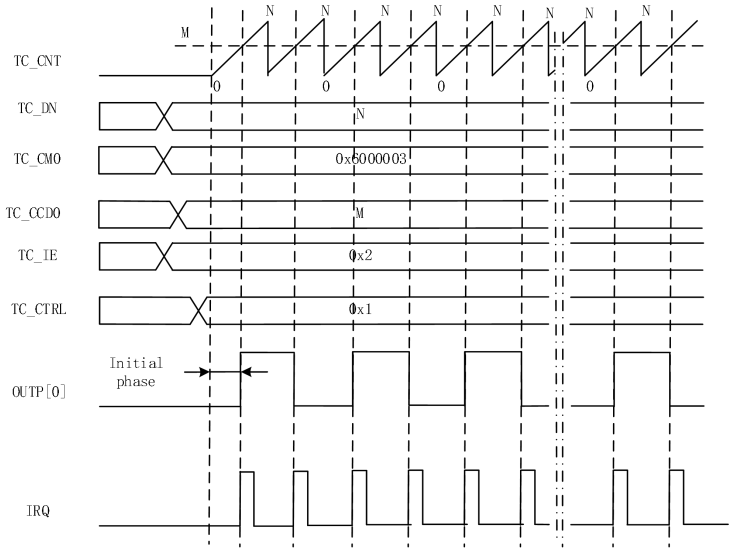


Figure 19-3 Square Wave Output

For the square wave output function, only the following registers need to be set:

- Target Count Value Register, counted by the counting clock, which can be set to the maximum value;
- Capture Compare Channel 0/1 Mode Register setting: enable ENABLE Channel 0/1, configure CCM to compare mode, configure DFTLVL to default level, configure EFELVL to valid level, and configure OM output to toggle function;
- Set Capture Compare Channel 0/1 Data Register (the set value shall not be greater than the Target Count Value Register);
- Enable Capture Compare Channel 0/1 interrupt in the Interrupt Enable Register;
- Control Register, start the timer.

The value of the Target Count Value Register determines the period of the output square wave, and the value of the Capture Compare Channel 0/1 Data Register is the output toggle point.

Common Optional Function Configuration Description:

- The prescaler register value can be modified to change the frequency of the timer counting clock.

- The Clock Configuration Register and the corresponding trigger source of TC in the AOS chapter can be configured. The frequency of the external input clock source cannot be higher than half the frequency of the internal system clock.
- External input clock filtering function: enable the FLTEN filtering function in the Clock Configuration Register, and set the filtering cycle number through configuring FLTOPT.

19.5.3 External Event Counting

Adopt external clock mode to count the valid edges of external signals. At the same time, set the compare channel to the preset number of events.

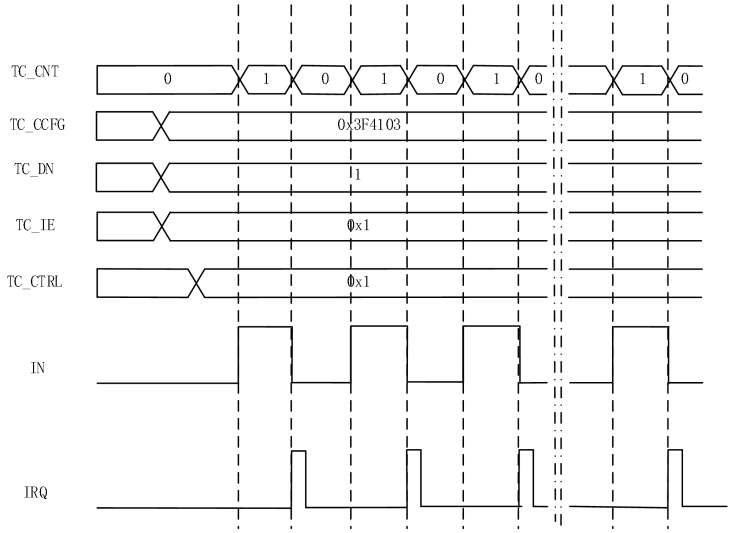


Figure 19-4 External Event Counting

19.5.4 External Clear and Start

The slave mode adds the function of controlling the internal counter's clear and gating through external input events to the original functions.

For the external clear function, only the following registers need to be set:

- Target Count Value Register, counted by the counting clock;
- Enable the overflow interrupt in the Interrupt Enable Register;
- Control Register: enable SLVEN slave mode, and configure the corresponding trigger source of TC in the AOS chapter;
- Control Register, start the timer.

In this way, an external clear function for the internal CNT is added in the free-running mode.

Common Optional Function Configuration Description:

- The prescaler register value can be modified to change the frequency of the timer counting clock;
- The Clock Configuration Register can be configured, CM can be modified to external input clock, and the corresponding trigger source of TC in the AOS chapter can be configured at the same time. The frequency of the external input clock source cannot be higher than half the frequency of the internal system clock;
- Single-count mode: the timer stops after overflow; set the OPS bit in the Control Register to 1 for single-count mode;

- If the external input clock source is the output of another timer, the two timers can be connected in cascade mode to increase the bit width of the timer registers;
- In the slave mode control mode selection, if gating mode is selected, configure SLVGATELVL slave mode gating valid level; the internal counter counts only when the input of the slave mode is at the gating valid level.

When the slave mode is configured as input capture mode, and the external input event of the slave mode is configured to be the same as the captured external input event, select one edge for capture and the other edge for slave mode polarity, then the pulse width can be directly obtained through the Capture Compare Channel Data Register.

19.5.5 Free-Running Mode

Auto-run mode is the interval timing function.

For the basic timing function, only the following registers need to be set:

- Target Count Value Register, which represents the timing duration, counted by the counting clock;
- Enable the overflow interrupt in the Interrupt Enable Register;
- Control Register, start the timer.

The timer will generate interrupts periodically with the target count value as the cycle.

Common Optional Function Configuration Description:

- The prescaler register value can be modified to change the frequency of the timer counting clock.
- The Clock Configuration Register and the corresponding trigger source of TC in the AOS chapter can be configured. The frequency of the external input clock cannot be higher than half the frequency of the internal system clock.
- Single-count mode: the timer stops after overflow; set the OPS bit in the Control Register to 1 for single-count mode.
- If the external input clock source is the output of another timer, the two timers can be connected in cascade mode to increase the bit width of the timer registers.

19.5.6 PWM Output

The Pulse Width Modulation (PWM) mode can generate a signal whose frequency is determined by the TC_DN register and whose duty cycle is determined by the TC_CCDx register. Two PWM modes are supported: PWM Mode 1 and PWM Mode 2.

PWM Mode 1: If TC_CNT < TC_CCDx, the output is at valid level; otherwise, it is at invalid level.

PWM Mode 2: If TC_CNT >= TC_CCDx, the output is at valid level; otherwise, it is at invalid level.

The following figure is a typical application diagram of PWM Mode 1.

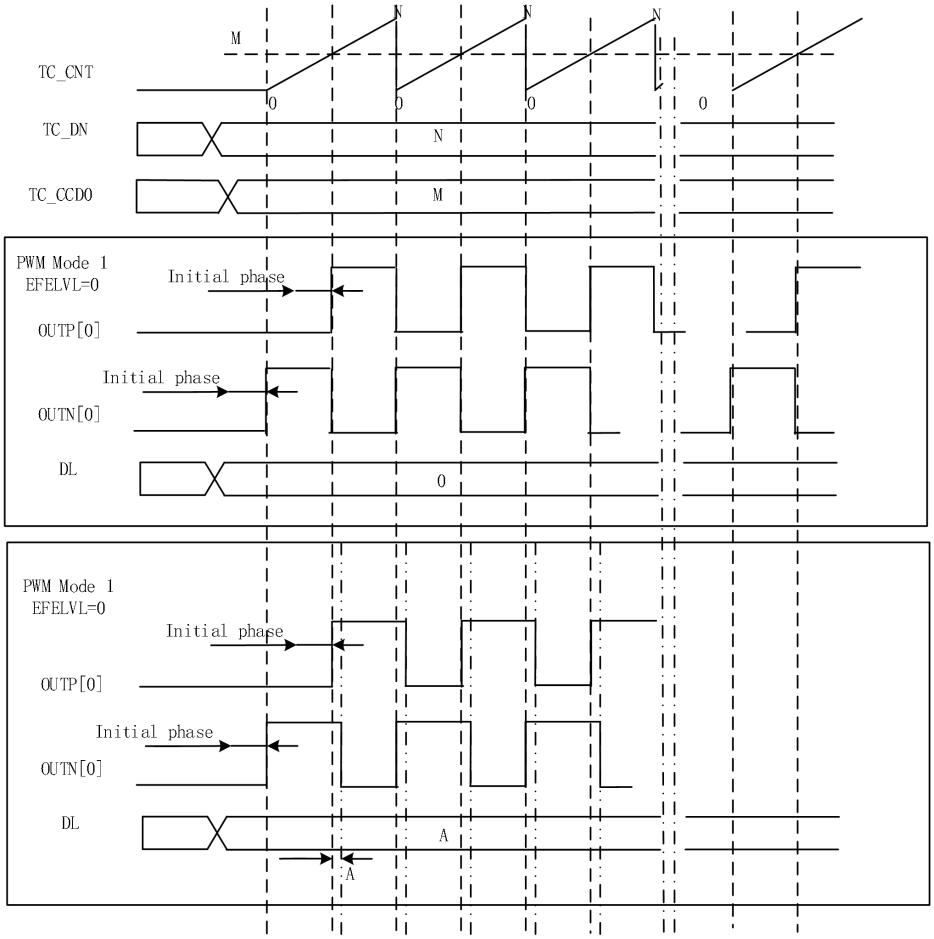


Figure 19-5 PWM Mode 1

For the PWM output function, only the following registers need to be set:

- Target Count Value Register, counted by the counting clock;
- Capture Compare Channel 0/1 Mode Register setting: enable ENABLE Channel 0/1, configure CCM to compare mode, configure DFTLVL to default level, configure EFELVL to valid level, and configure OM output to PWM Mode 1 or PWM Mode 2;
- Set Capture Compare Channel 0/1 Data Register, which must be smaller than the Target Count Value Register;
- Control Register, start the timer.

The P terminal of the channel outputs the positive waveform of PWM Mode 1/PWM Mode 2, and the N terminal of the channel outputs the inverted waveform of the P terminal.

PWM Mode 1: The period is (Target Count Value Register value + 1), and the valid level period is (Channel Data Register value + 1) cycles.

PWM Mode 2: The period is (Target Count Value Register value + 1), and the invalid level period is (Channel Data Register value + 1) cycles.

Common Optional Function Configuration Description:

- The prescaler register value can be modified to change the frequency of the timer counting clock.
- The Clock Configuration Register and the corresponding trigger source of TC in the AOS chapter can be configured. The frequency of the external input clock source cannot be higher than half the frequency of the internal system clock.

- Complementary output with dead-time insertion: enable DIEN dead-time insertion, and configure DL to set the dead-time insertion length. A delay is added between the two edge transitions of P and N, that is, the two edges are not toggled at the same time.

When the valid level EFELVL is low level: the falling edges of P and N outputs are delayed by DL cycles.

When the valid level EFELVL is high level: the rising edges of P and N outputs are delayed by DL cycles.

19.5.7 Pulse Width Measurement

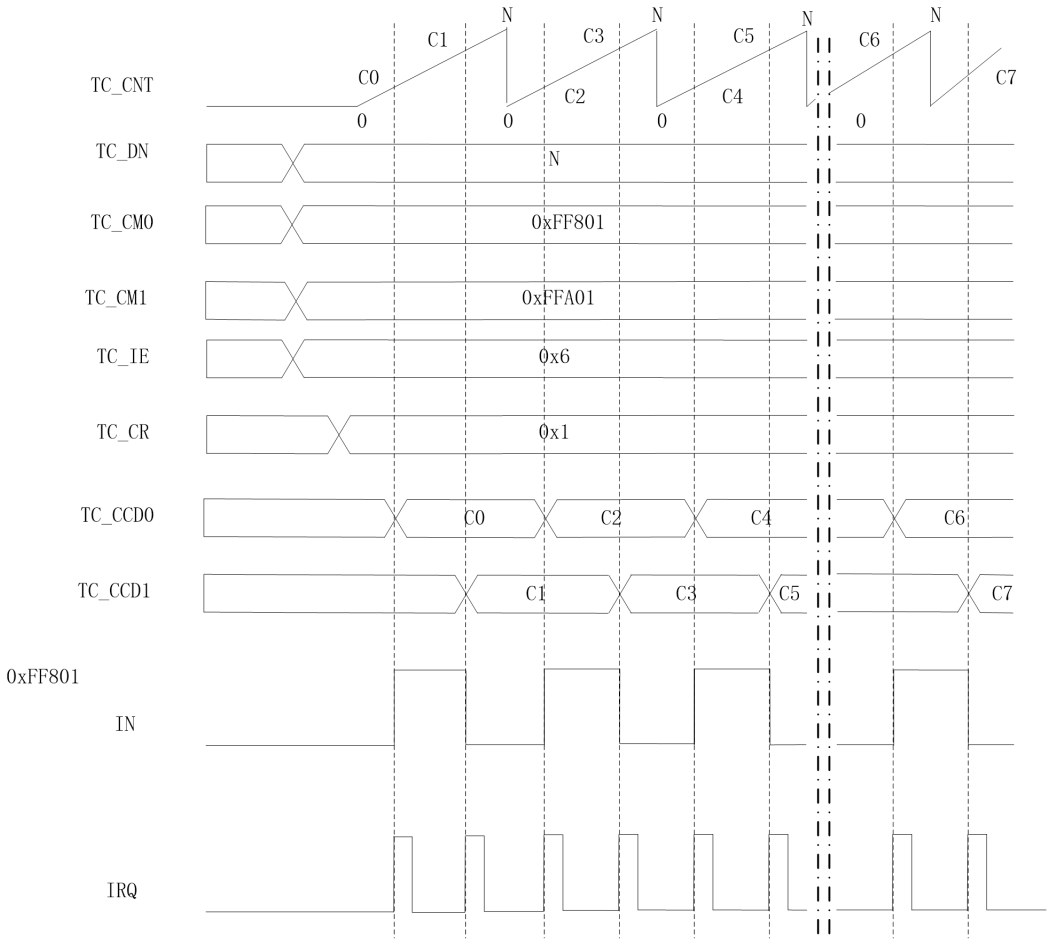


Figure 19-6 Pulse Width Measurement

The main function of the input capture mode is to test the pulse width.

For the basic pulse width test function, only the following registers need to be set:

- Target Count Value Register, counted by the counting clock, which can be set to the maximum value.
- Capture Compare Channel 0/1 Mode Register setting: enable ENABLE Channel 0/1, configure CCM to capture mode, select CPOL for capture polarity, and select CS for external input event.
- Enable Capture Compare Channel 0/1 interrupt in the Interrupt Enable Register.
- Control Register, start the timer.

When the timer captures the capture polarity of the external input event, an interrupt is generated, and the current count value is saved in the Capture Compare Channel Data Register. If two channels are used, one channel captures the rising edge and the other captures the falling edge, the pulse width can be calculated through the data registers of the two channels.

Common Optional Function Configuration Description:

- The prescaler register value can be modified to change the frequency of the timer counting clock.
- The Clock Configuration Register can be configured, CM can be modified to external input clock, and the external input clock source configured by the CS bit can be modified at the same time. The frequency of the external input clock source cannot be higher than half the frequency of the internal system clock.
- Filtering function for external input events: enable the FLTEN filtering function in the Capture Compare Channel Mode Register, and set the filtering cycle number through configuring FLTOPT.
- If the external input event is the output of another timer, the two timers can be connected in cascade mode.

19.5.8 Timer Cascade

The master timer adopts free-running mode, and the slave timer adopts single-pulse slave mode.

19.5.9 Timer Parallel Connection

The synchronous operation of all or part of the timers can be realized by reset mode or gating mode.

20 LSRX

20.1 Introduction

This module is used to interface with the LSTX module of MAKSA single-phase metering chips. It can be used in single-phase meter metering solutions and three-phase manganese copper meter metering solutions. The module parses the LSTX data received via UART, judges the legality of the data frame, extracts the time information from the data frame, calculates the frame interval, which can be used to estimate the clock cycle of the metering chip, the reset time difference between chips, and achieve correction of the zero-crossing time.

Furthermore, this module interacts with the chip UART module, outputs waveforms through the UART-TX pin for clock calibration compensation of single-phase chips in crystal-less mode; it can also detect the UART-RX pin, and the UART receiver module officially starts data reception only after detecting LSTX IDLE, preventing communication errors.

20.2 Features

Has the following features:

- Supports multiplexing all UART interfaces;
- Supports automatic recognition of LSTX _IDLE before starting UART reception;
- Supports output of calibration waveforms, supports configurable output waveforms;
- Supports real-time upload of frame header time;
- Supports data reception timeout detection;
- Supports 4 data channels;
- LSRX can be configured to be called by CORE0 or CORE1, but can only interact with the UART configured under the same core.

20.3 Basic Principles

20.3.1 Correction of Zero-Crossing Time Uploaded by LSTX by LSRX

The error in the zero-crossing time uploaded by LSTX mainly comes from the error caused by the deviation of the working clock RCH of the metering chip special-chip itself. Here, the situation of data transmission for three-phase channels is explained using the case of two channels as an example.

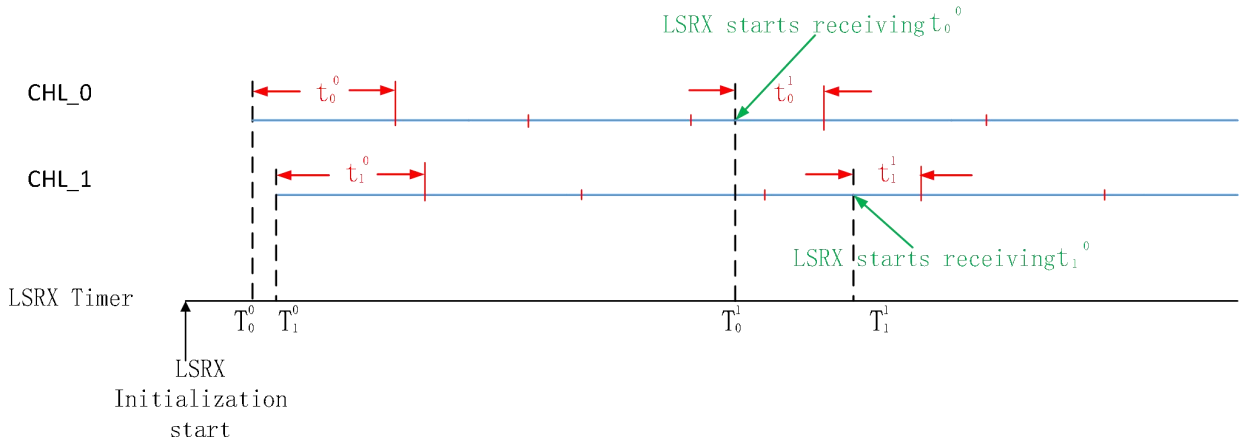


Figure 20-1 LSRX Reception Timing Diagram

As shown in the figure above, the variables identified in the figure are as follows:

- T_n^m represents the frame header time of the m-th frame recorded by channel n. This time is recorded by the timer in the LSRX module of the dual-core MCU and is the accurate time.
- t_n^m represents the zero-crossing time in the m-th frame data sent by the metering chip LSTX of channel n. The zero-crossing time sent by LSTX at time t_n^m starts timing from the first zero-crossing time of the previous metering communication cycle, i.e., starting from T_n^{m-1}, until the first zero-crossing count appears thereafter. This count is based on the inaccurate RCH count of the metering chip and needs correction.

Additionally, within one metering communication cycle, there may be 3-4 zero-crossings, but LSTX only sends the first zero-crossing after the frame start. This means the update period of the zero-crossing time is consistent with the metering communication interval.

To correct the zero-crossing time of a channel, i.e., to correct t_n^m, take channel CHL_0 as an example for analysis.

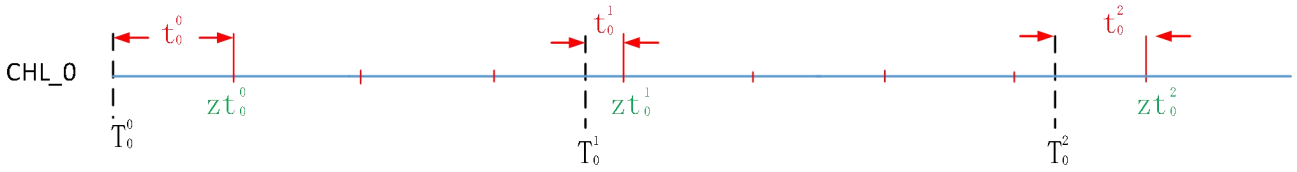


Figure 20-2 Single Channel Zero-Crossing Time Correction Diagram

As shown in the figure, the variable zt_0^n represents the real time of the zero-crossing time for the n-th frame of channel 0. The following relationship can be derived:

$$\frac{zt_0^n - T_0^n}{T_0^{n+1} - T_0^n} = \frac{t_0^n}{T_T}$$

zt_0^n represents the accurate time point of the zero-crossing time based on the LSRX timer;

T_0^n represents the arrival time of the n-th frame data header of channel 0, which is the accurate time recorded by the LSRX timer;

t_0^n is the n-th frame zero-crossing time sent by LSTX to LSRX, which is the inaccurate time recorded by the metering chip timer;

T_T is the COUNT value when the single-phase metering chip TC counts a full cycle (35/71ms).

Then the accurate zero-crossing time zt_0^n should be expressed as:

$$zt_0^n = T_0^n + T_{tn} = T_0^n + \frac{t_0^n}{T_T} * (T_0^{n+1} - T_0^n)$$

Where t_0^n、T_0^n、T_0^{n+1}、T_T are all known quantities.

Through the above calculation, the accurate time of the first zero-crossing in each channel's frame communication cycle can be obtained. Since the zero-crossing sent by LSTX always lags by one frame cycle, the correction of zero-crossing data can only start after two consecutive valid data frames are received. For details on the zero-crossing data correction flow, refer to "19.3.3.8 Zero-Crossing Data Correction Scheme".

20.3.2 Analysis of Frame Header Time Difference Between Channels

In the Single-Phase Energy Metering Chip three-phase solution, due to the RCH deviation of each Chip metering chip, the frame header interval of data frames from different channels will gradually increase. As shown in the figure, analysis shows that the frame header interval between different channels consists of two parts: 1. The

reset time difference caused by different power-on initialization times between the two channels. This is reflected as the frame header interval of the first frame data of each channel, and this reset time difference does not change until a channel undergoes power-on reset again; 2. The difference in the communication cycle (71ms/35ms) generated by the RCH deviation of each channel's metering chip. This difference accumulates with the increase in the number of communication data frames. Analyzing the frame header time difference here, we can get its expression:

$$\Delta 0_{01}^{n} = \Delta 0_{01}^{0} + \sum_{m=1}^{n} (T_0^m - T_1^m)$$

Where $\Delta 0_{01}^{n}$ represents the frame header interval between CHL_0 and CHL_1 for the n-th frame.

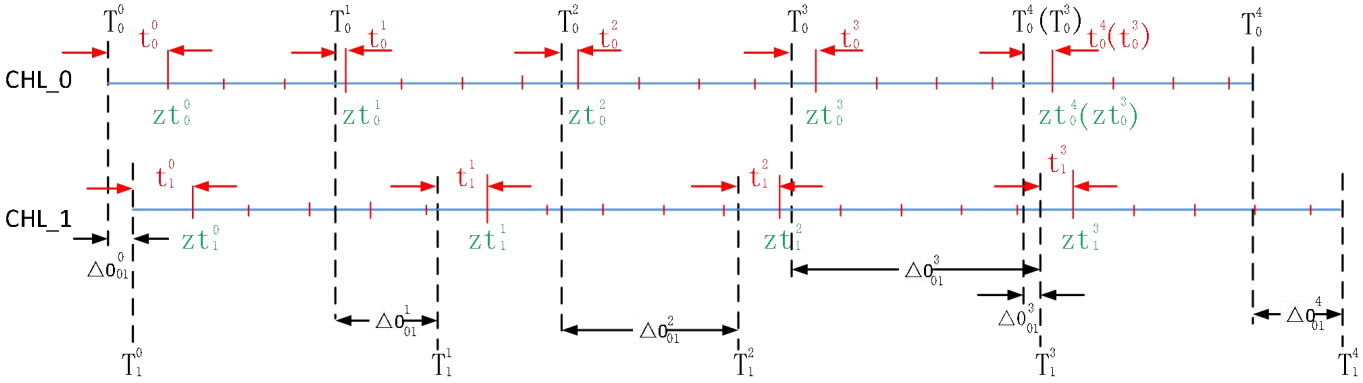


Figure 20-3 Frame Header Time Difference Diagram

If no processing is performed on the RCH deviation accumulation, it will inevitably lead to misalignment of frame data between the two channels. As shown in the figure, at time T_1^3, CHL1 has just completed metering and communication for 3 cycles, while CHL0 has already completed metering and communication for 4 cycles. During software processing, if one channel has one more frame of data than other channels within the same time range, this frame of data can either be ignored or used to calculate the phase angle with the last frame of data received by other channels.

It should be noted here that the zero-crossing data of each channel does not completely correspond. Taking the situation in the figure as an example, zt_1^2 calculated together with zt_0^2 is actually the data lagging behind $zt_0^2$by 3 voltage signal cycles. The red scales in the figure refer to each zero-crossing time (with a cycle of 20ms), the long red scales indicate the first zero-crossing after the frame starts, and LSTX will send its relative time to LSRX. The short red scales indicate the zero-crossings of the voltage signal that occur within the frame cycle; LSTX does not collect these zero-crossing points, and these zero-crossing times are unknown. During calculation, the zero-crossing time point of CHL1 channel corresponding to zt_0^2 is deduced through zt_1^2 to realize the phase angle calculation between CHL0 and CHL1.

The alignment of zero-crossing times of each channel can be reflected by the frame header time difference, and also by the zero-crossing time difference used to calculate the channel phase angle.

20.3.3 Handling of Phase Loss Issues

For the handling of phase loss issues, the phase loss situations are divided into two types: phase loss with data and phase loss without data, and the two situations are analyzed respectively.

20.3.3.1 Phase Loss with Data

Phase loss with data means that after a phase loss occurs in a certain channel, the power supply of the metering chip is normal and it can still work normally. At this time, the metering chip LSTX continues to send metering data to the MCU, but the metering data at this time is invalid. This type of phase loss can be distinguished by detecting the voltage PEAK value in each frame of data. By setting an effective threshold for the voltage PEAK value, when the voltage PEAK value in the data frame received by LSRX is lower than the effective threshold, it is determined that phase loss occurs in this channel's data, and this frame of data is invalid (for detailed software detection steps, refer to "19.3.3.3 Data Validity Judgment" later in the document).

20.3.3.2 Phase loss without data

Phase loss without data means that after a phase loss occurs in a certain channel, the chip power supply stops, the chip stops working at this time, and the metering chip LSTX cannot send metering data to the MCU. The situation at this time is as shown in the following figure.

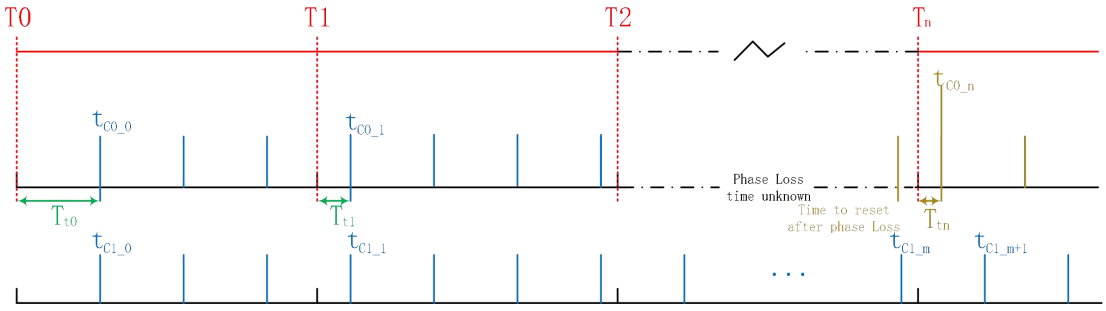


Figure 20-4 Schematic Diagram of Phase Loss Without Data

As shown in the figure, CHL0 data has phase loss without data at time T2 and recovers after T_1^n. When phase loss without data occurs, LSRX cannot receive data from this channel all the time. At this time, a reception timeout threshold is set for each channel to detect the occurrence of phase loss without data. Here, 105% of the communication cycle (71.1ms/35.5ms) is used as the timeout threshold. This setting is because the deviation caused by inaccurate RCH is usually within ±3%, and UART data reception will have errors when the deviation exceeds ±5%. Therefore, if the next frame of data is still not received within T*105% time after the arrival of the previous frame header, this frame of data is considered lost (even if this frame of data arrives after the time threshold, the UART communication is unreliable when the deviation is greater than 5%, and the data is still considered invalid). At this time, the LSRX module will end the data reception operation of this frame in this channel, set the data update flag to 1, and set the data reception timeout flag to 1 (timeout).

When CHL0 resumes work after T_0^n, the state of this channel is equivalent to a re-reset and power-on. After LSRX successfully receives two consecutive frames of data, the RCH deviation and reset time difference of CHL0 can be recalculated through the frame header times of these two frames, and then the work of each channel returns to the normal working state.

It should be additionally noted that when CHL0 resumes work after T_0^n, a beat difference may occur when the corrected zt_0^n is used to calculate the phase angle with the zero-crossing times of other channels. This situation is equivalent to the situation where the frame header time interval between channels is greater than one metering communication cycle in the analysis of frame header time difference between channels in Section 19.3.2. For relevant analysis, please refer to the analysis in Section 19.3.2.

20.3.3.3 Data Validity Judgment

After the four channels of data are received, the software corrects the zero-crossing data of each channel. During each correction calculation of the zero-crossing data, the software needs to first judge the validity of the data. The software judgment is divided into three parts:

- Query the data reception timeout bit of the channel. When the flag bit is 0 (no timeout), it indicates that no phase loss without data has occurred in this channel's data;
- Detect the voltage PEAK value in the frame data stored in SRAM. If the 4 consecutive 2-byte voltage PEAK values after the frame header are all less than the effective threshold of the voltage PEAK value, it is determined that the frame data in SRAM is invalid. (There are two situations that cause the 4 voltage PEAK values to be less than the effective threshold of the voltage PEAK value: phase loss without data occurs, or the frame data is not stored in SRAM due to frame check failure).
- Detect the check of the entire frame of data in SRAM. Due to the phenomenon that DMA may store the data of the previous and next frames in an overlapping manner as mentioned in Section 19.3.3.5, the software needs to perform a check detection on the overall data when calling the data in SRAM.

20.3.3.4 Phase Angle Calculation During Phase Loss

When phase loss occurs in a certain channel's data, the software selects the phase angle reference channel according to the following rules:

- When Channel A data is received successfully and is valid, other channels with successfully received and valid data all use Channel A as the reference to calculate the phase angle;
- When phase loss occurs in Channel A data, if Channel B data is received successfully and is valid, the other two channels use Channel B as the reference to calculate the phase angle (Channel A data is invalid, and the phase angle between it and other channels is not calculated);
- When phase loss occurs in both Channels A and B, if Channel C data is received successfully and is valid, the neutral line uses Channel C as the reference to calculate the phase angle;
- When phase loss occurs in Channels A, B, and C, only the neutral line metering data is processed, and the phase angles between channels are not calculated;

20.3.3.5 Detection of Frame Data Reception Completion

Currently, the single-phase metering chip LSTX module sends 28-byte or 60-byte data in one metering communication cycle. During data communication, the UART module is responsible for data reception. The check and detection mechanism of the UART part during the process is based on bytes, and the UART cannot detect whether all data of the entire 28-byte or 60-byte communication process is correctly received. The frame header detection of LSRX only detects whether the first 4 bytes of the 28-byte data are correct. In the actual transceiving process, if several bytes are received first and then communication fails due to strong external interference, the UART of the dual-core MCU will never receive the start bit of the data frame, and the phenomenon of overlapping data of the previous and next frames occurs in this channel. To avoid this situation, the software needs to perform a check on the data when DMA extracts the data, referring to the software judgment (3) in Section 19.3.3.3.

20.3.3.6 Single-Phase Solution

The single-phase chip is mainly used in single-phase solutions, and the application connection scheme is as shown in the following figure. In conventional applications, the dual-core MCU uses the UART interface for

register configuration and data reading. After the LSTX module is enabled, the LSTX module uses the TX pin to periodically send data frames containing configuration register data, pulse data, and zero-crossing data to the dual-core MCU. After the UART receives the data, the LSRX module is responsible for detecting the frame header of the data. If the frame header detection is valid, the time points of the previous and next frames of this channel are recorded, and an interrupt and a flag bit are provided each time the data is updated to remind the software to calculate the frame interval. In addition, through configuration, a calibration waveform can be output to single-phase chip through the TX of the UART to realize clock calibration compensation for the single-phase chip in the crystal-free mode.

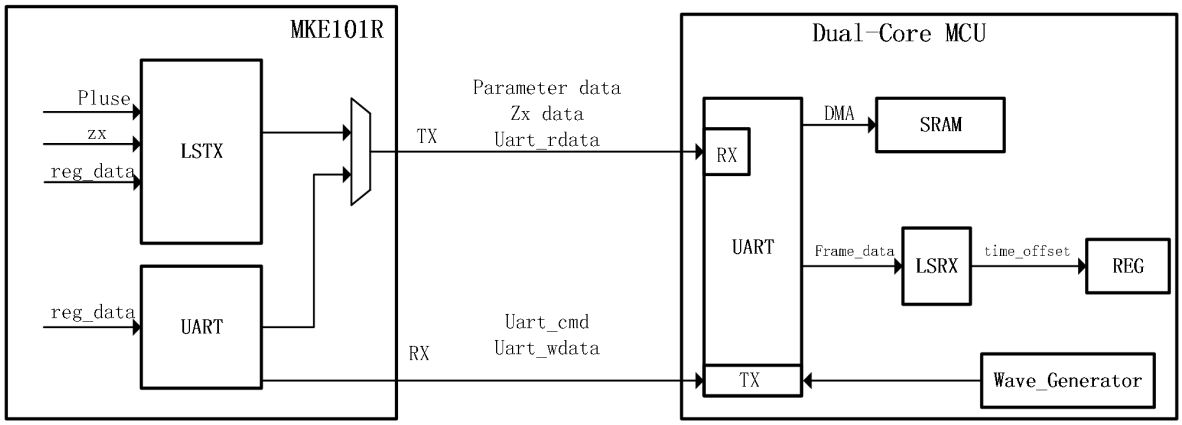


Figure 20-5 Single-Phase Measurement Solution

The roles played by LSRX in this solution include:

- Data frame header identification: if the identification is successful, the relevant data is updated, and an interrupt is provided to remind the software to calculate;
- Calibration waveform output: supports configurable waveforms to realize clock compensation for the single-phase chip in the crystal-free mode.

20.3.3.7 Three-Phase Manganin Meter Solution

The single-phase chip is mainly used in three-phase manganin meter solutions, and the application connection scheme is as shown in the following figure. Only the TX pin of single-phase Chip is connected to the MCU, the RX pin is pulled up, and four the Special Single-Phase Metering IC chips are connected to the dual-core MCU. After the single-phase chip is powered on and reset, LSTX starts automatically and uploads parameter register data and zero-crossing data to the dual-core MCU through the TX pin. The dual-core MCU stores the data into SRAM through DMA, and at the same time, LSRX marks the data frame time and records the time points when the data frame headers of each channel arrive, for the software to calculate the correction value for the zero-crossing time of each channel.

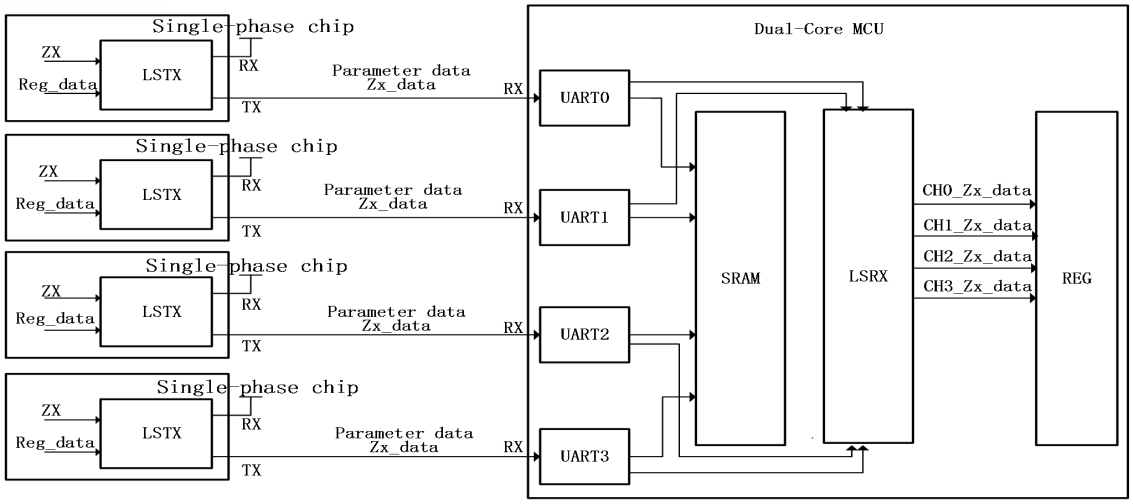


Figure 20-6 Three-Phase Manganin Meter Measurement Solution

In conventional applications of the single-phase chip, the UART RX pin is pulled up, and clock calibration compensation cannot be performed. Therefore, the LSTX transmission cycle is related to the internal RC frequency of the single-phase chip.

The roles played by the LSRX module in this solution include:

- Enable data reception only when LSTX IDLE is detected to prevent communication errors. This function is difficult to implement with software and has high overhead;
- Provide the time points when the frame headers of the previous and next frames of data in the same channel arrive, and provide an interrupt and a flag after data update to remind the software to calculate the clock cycle of the metering chip in this channel;
- Detect the frame header reception interval of each channel. If the frame header of the next frame is not received within the set time threshold, it is determined that channel phase loss or data loss occurs, and a corresponding flag is given to remind the software;

20.3.3.8 Correction Scheme Flow for Zero-Crossing Data Uploaded by LSTX

Based on the previous content, the LSTX hardware workflow and the software processing flow for calculating zero-crossing time and phase angles of each channel are given respectively.

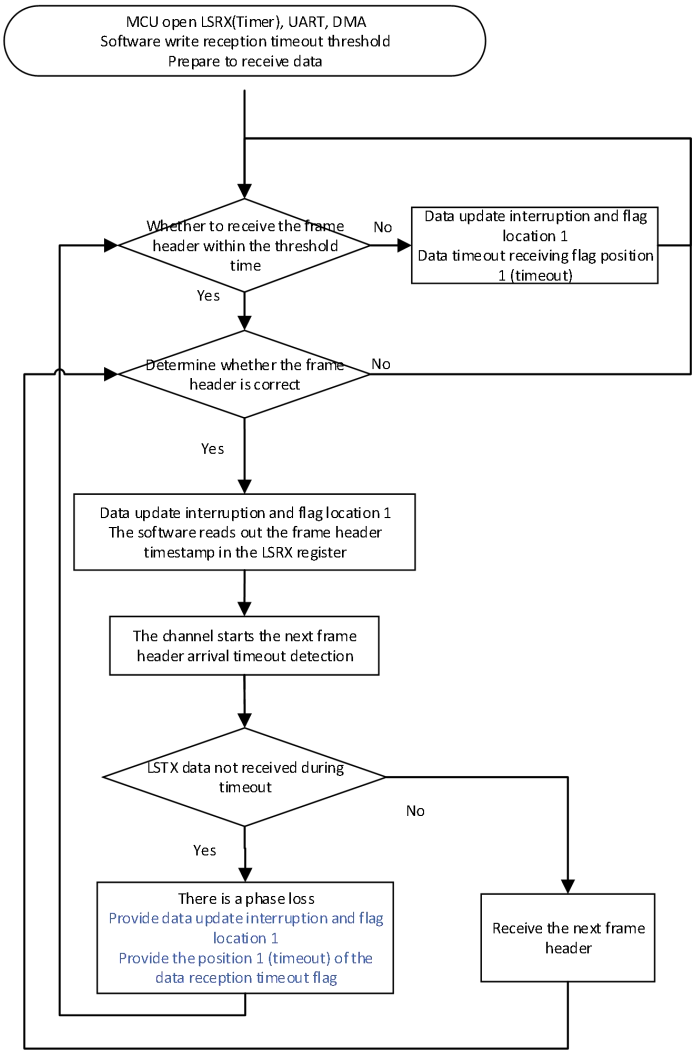


Figure 20-7 LSTX Hardware Workflow

- LSRX Hardware Workflow

The LSRX hardware workflow is given first. After LSRX is enabled, the timeout detection of each channel starts timing. The timing threshold is set by the software in [21:25]Time_TH in the LSRX_CTL register. The timeout threshold is equal to (Time_TH +1)*8.8888ms (the default value is 9, i.e., 79.9ms~87ms, which is greater than the minimum configurable value of 71*1.05). When a timeout occurs, LSRX will generate a data update interrupt and the corresponding channel's data update flag and reception timeout flag.

When LSRX receives a data frame, LSRX will perform validity detection on the data frame header (the specific detection method can be configured through the frame header identification function enable register LSRX_FHD). (1) When the LSRX frame header validity detection passes, LSRX will record the time of the frame header, generate a channel data update interrupt and a flag to notify the software, and at the same time, LSRX will start timing from the frame header arrival time to detect data reception timeout; (2) If LSRX fails to receive a valid frame header within the timeout threshold, which may be due to no frame header received (phase loss) or invalid frame header received (communication interfered, data error), LSRX will generate a data update interrupt, a flag, and a data reception timeout flag.

The two status flags and two hardware judgment thresholds in the hardware workflow are explained below:

- Two Status Flags in the Workflow

During the entire hardware execution process, the CHLx_UPDATA_IF (LSRX_STA[4:1]) data update flag bit

is used to inform the software of the channel communication progress, and the data timeout reception flag bit CHLx_Valid_IF (LSRX_STA[8:5]) is used to inform the software whether the channel has detected a valid data frame header within the specified time. Both flag bits are cleared by software writing 1.

✓ CHLx_UPDATA_IF Data Update Flag (x=0-3)

It is used to indicate that the channel has completed the collection of frame header time once. The data update flags of the four channels will all trigger a data update interrupt. After entering the data update interrupt, the software first detects which channel's data has been updated, then queries whether the channel that provides the data update flag has a reception timeout flag. If a reception timeout flag appears, it indicates that phase loss occurs in this channel during this cycle, and no processing is performed; otherwise, the frame header time register data of the data update channel is read and stored.

✓ CHLx_Valid_IF Data Reception Timeout Flag (x=0-3)

It is used to match the CHLx data update flag to inform the software whether phase loss occurs in the channel. This flag is provided by hardware and does not include the detection of data validity in SRAM. In addition, when the software detects data validity, in addition to detecting the flag provided by this hardware, it also needs to check the data in SRAM by itself (refer to "19.3.3.3 Data Validity Detection").

① When the reception is successful within the timeout threshold and the frame header check passes, the data reception timeout flag is set to 0 (no timeout), and the data update flag is set to 1 (completed);

② When the channel has a timeout of 8.888*(N+1) ms or a frame check error, it is determined that the data reception of this channel is timed out, the data reception timeout flag is set to 1 (timeout), and the data update flag is set to 1 (completed).

■ Hardware Judgment Thresholds

✓ Timeout Threshold (Hardware Execution)

Each channel has an exclusive timer for timeout detection. Timeout detection refers to the timeout detection of the arrival of the next frame header. The deviation of the channel caused by RCH offset is about ±3%. Therefore, if the frame header is not successfully received again within (72*105%) ms after the arrival of the previous frame header, it is considered that phase loss occurs in the channel. At this time, it is considered that the MCU fails to receive the zero-crossing data, and the data reception timeout flag is set to 1 (timeout), and the data update flag is set to 1.

✓ Frame Header Judgment (Hardware Execution)

To distinguish between the data sent by LSTX and the noise data between channels, it is necessary to perform validity judgment on the frame header first when recording the frame header time. Only when the frame header is valid will the arrival time of this frame header be recorded, and this frame header time is considered valid.

● Software Correction of Zero-Crossing Data and Calculation of Phase Angles

Figure 19-8 below shows the calculation flow of the software correcting the zero-crossing data and calculating the phase angles of each channel based on the flags and data provided by LSRX.

According to the calculation requirements, the software first divides the corresponding space for storing frame data and frame header time. After LSRX generates a data update interrupt, the software enters the interrupt function. In the interrupt function, the software first determines which channel's frame header time has been updated, and the software queries whether the corresponding channel has a reception timeout. If there is no reception timeout, the frame header time in the LSRX register is promptly taken out and stored in the pre-divided space, the stored frame header time is counted, and then the corresponding bit flag is cleared by bit. Here, the flag must be cleared by bit, and the entire register cannot be cleared to avoid missing data of a certain channel when multiple channels of data arrive simultaneously; if a timeout occurs, the frame header count is also increased by 1, but the corresponding frame data needs to be marked as invalid, and this frame data is invalid in subsequent calculations.

Outside the interrupt function, the function detects whether the frame header times of each channel currently stored have all reached N+1. If all have reached N+1, then N frames of data must have been received, and then the software processes each frame of data in sequence. When the software processes the data, it proceeds according to the following steps:

Step 1: Move out N frames of data and the frame header times of N frames of data, clear the data reception address, and prepare to receive new data;

Step 2: Detect whether the data is invalid data corresponding to a timeout. The timeout invalid data is not corrected for zero-crossing, and the channel does not participate in the phase angle calculation of the corresponding cycle. The valid data continues with the subsequent operations;

Step 3: Detect the data in SRAM, check whether the voltage PEAK value is greater than the threshold, and check whether the check data of the entire frame of data is correct. The data that fails the detection is considered invalid, and its zero-crossing data is not corrected, and the phase angle between this channel and other channels is not calculated;

Step 4: Correct the zero-crossing data according to the theoretical calculation, and calculate the phase angles of each channel;

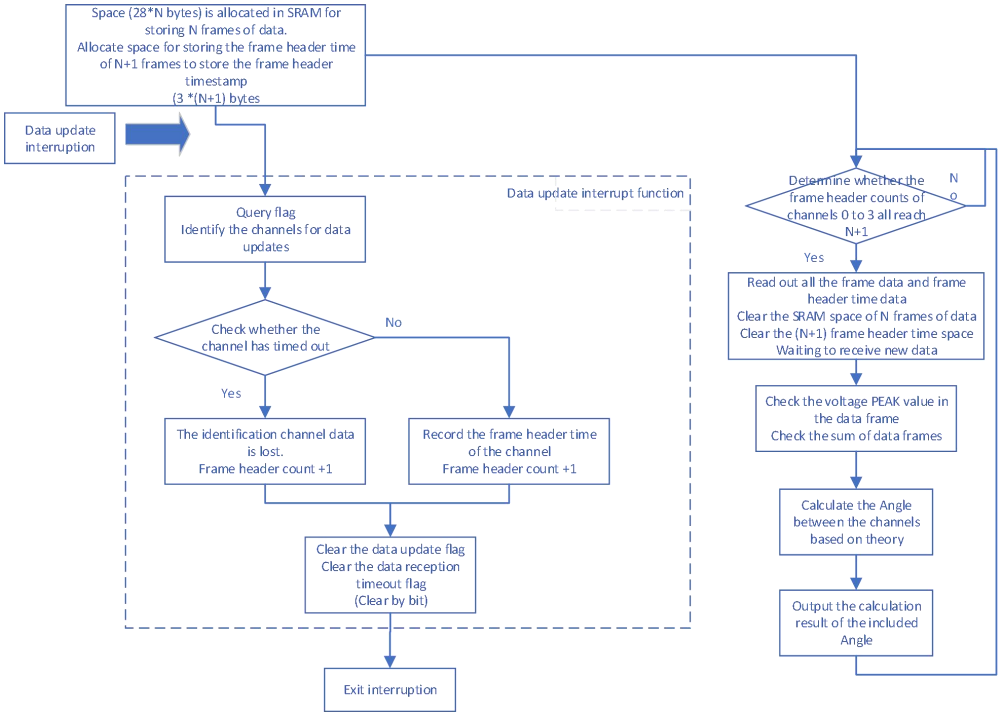


Figure 20-8 Software Processing Flow for Zero-Crossing Time Correction

20.4 Overall Implementation Scheme of LSRX

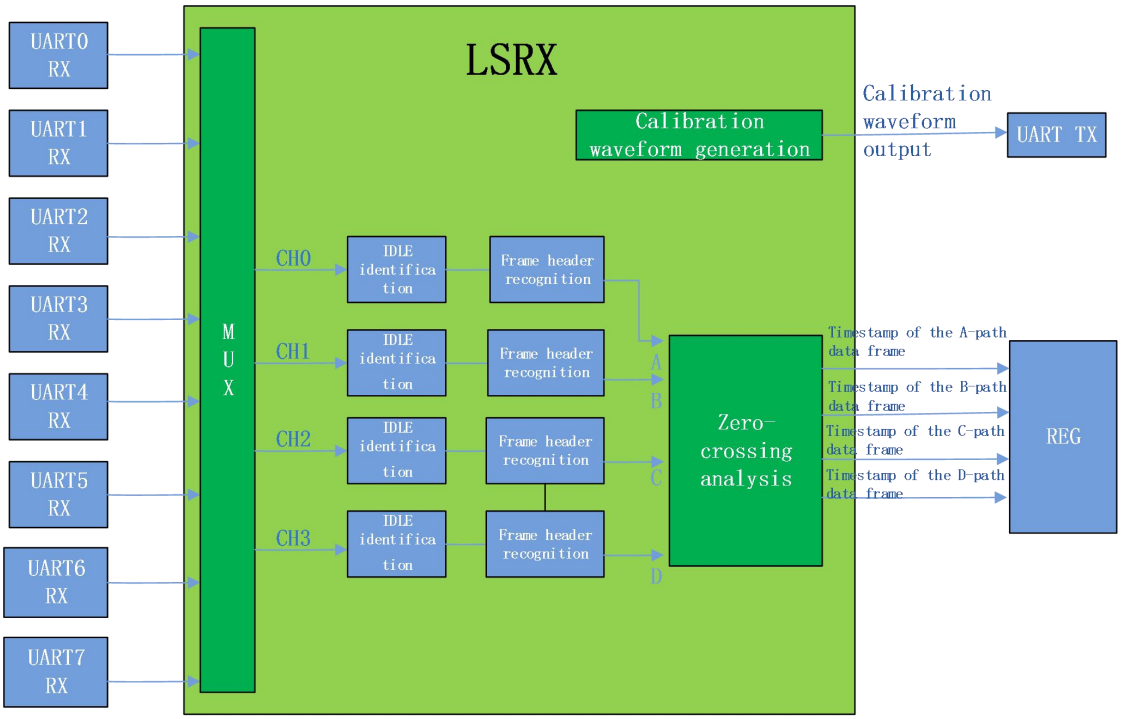


Figure 20-9 Schematic Diagram of LSRX Scheme

The above figure shows the schematic diagram of the implementation scheme of this module. The LSRX module can reuse all UART ports. The LSRX module uses the system clock as the module clock, and the time calculation is adaptive to the frequency. This module sets four input channels (Channel 0/1/2/3) for extracting the time of data frames.

After the LSRX module is enabled, it performs IDLE identification on the selected input UART RX. If it is detected as high level for 22 consecutive times, it indicates that LSTX is in the IDLE state, and UART-RX and DMA are enabled to start receiving data. When the frame header (special character + odd parity) is identified for the first time, timing starts, and the current frame header arrival time and the previous frame header arrival time of each channel are recorded respectively. At the same time, the LSRX module can output a clock calibration waveform through the UART-TX pin according to the configuration.

When docking with sinle-phase chip, it is mainly used for pulse forwarding of a single UART port and zero-crossing time uploading. At the same time of communicating with LSTX, RC clock calibration can be realized. The hardware structure when docking with the single-phase chip is as shown in the following figure:

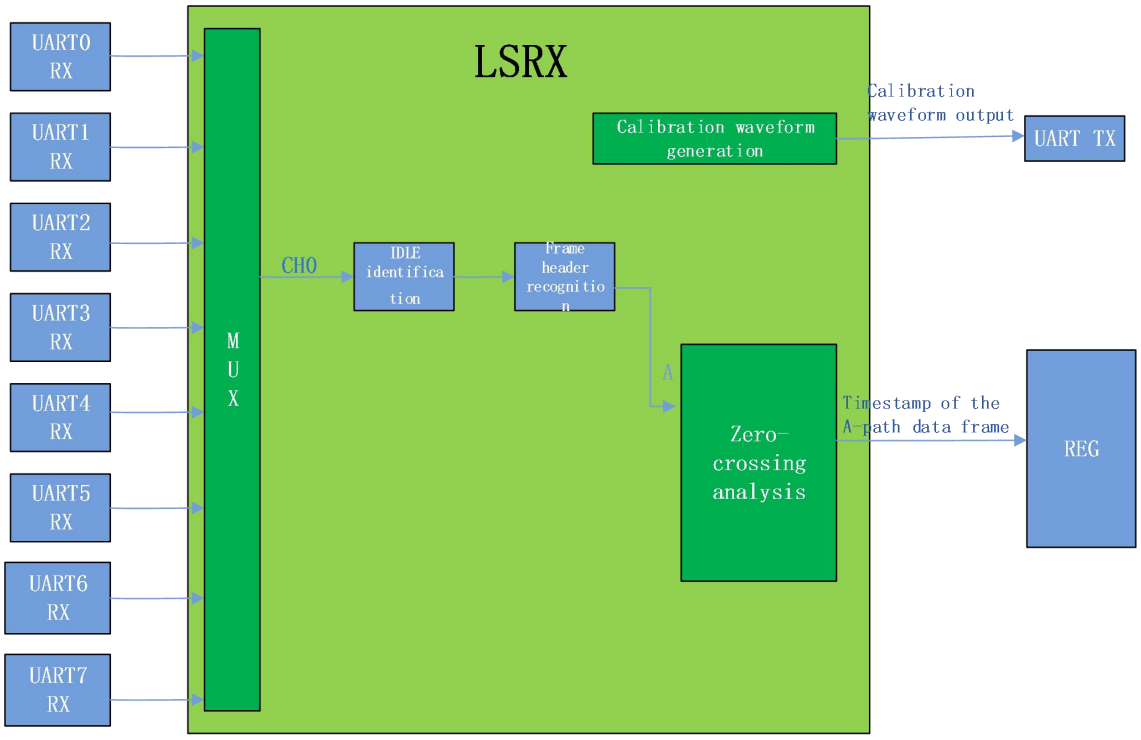


Figure 20-10 Schematic Diagram of LSRX Single-Phase Pulse Forwarding Scheme

When docking with the Special Single-Phase Metering IC, it is mainly used for three-phase zero-crossing synchronization. The hardware structure when docking with the Special Single-Phase Metering IC is as shown in the following figure. Due to the inconsistent power-on times of the four the Special Single-Phase Metering IC chips, when analyzing the zero-crossing times of the four the Special Single-Phase Metering IC chips, it is also necessary to calibrate the reference times of the four the Special Single-Phase Metering IC chips once. The transmission reference time difference of the four the Special Single-Phase Metering IC chips is obtained by calculating the time difference of the frame headers received by the four UARTs, and this time difference is compensated to the zero-crossing time after analysis and calculation, so as to obtain the real zero-crossing time difference of the four the Special Single-Phase Metering IC chips.

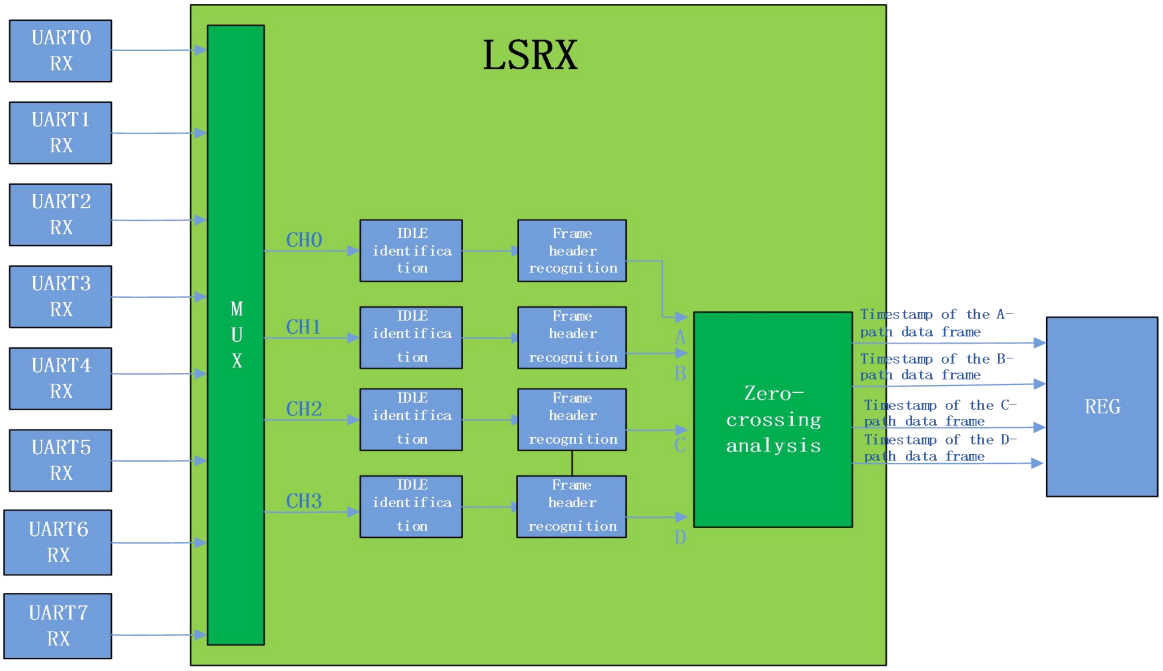


Figure 20-11 Schematic Diagram of LSRX Three-Phase Manganin Meter Zero-Crossing Synchronization Scheme

20.5 Register Description

Module Name	Secure Base Address	Non-Secure Base Address
LSRX	0x2002_C000	0xA002_C000

Register List

Register Name	Address Offset	Description
LSRX_FTOP	0x00	LSRX Frame Header Identification Register
LSRX_FHD	0x04	LSRX Frame Header Identification Function Enable Register
LSRX_CTL	0x08	LSRX Parsing Parameter Register
LSRX_CAL_CFG	0x0C	LSRX Calibration Output Configuration Register
LSRX_EN	0x10	LSRX Enable Register
LSRX_IE	0x14	LSRX Interrupt Enable Register
LSRX_IF	0x18	LSRX Interrupt Flag Register
LSRX_CH0FH_TIME	0x1C	LSRX Channel 0 Frame Header Time Register
LSRX_CH1FH_TIME	0x20	LSRX Channel 1 Frame Header Time Register
LSRX_CH2FH_TIME	0x24	LSRX Channel 2 Frame Header Time Register
LSRX_CH3FH_TIME	0x28	LSRX Channel 3 Frame Header Time Register

20.5.1.1 LSRX_FTOP（0x00）

LSRX Frame Header Identification Register

Bit	Bit Name	Function Description	R/W	Default Value
31:0	LSRX_FTOP	Frame header identified by the LSRX module	WR	0x800A8000

This register cannot be modified when LSRX_EN is 1.

20.5.1.2 LSRX_FHD（0x04）

LSRX Frame Header Identification Function Enable Register

Bit	Bit Name	Function Description	Read/Write	Default Value
31:8	Reserved	Reserved	R	0
7	LSRX_B3PRTEN	Fourth-byte check enable of frame header	WR	0b1
6	LSRX_B3CHKEN	Fourth-byte detection enable of frame header	WR	0b1
5	LSRX_B2PRTEN	Third-byte check enable of frame header	WR	0b1
4	LSRX_B2CHKEN	Third-byte detection enable of frame header	WR	0b1
3	LSRX_B1PRTEN	Second-byte check enable of frame header	WR	0b1
2	LSRX_B1CHKEN	Second-byte detection enable of frame header	WR	0b1
1	LSRX_B0PRTEN	First-byte check enable of frame header	WR	0b1
0	LSRX_B0CHKEN	First-byte detection enable of frame header	WR	0b0

Byte detection refers to detecting whether the corresponding byte can be equal to the corresponding byte of

LSRX_FTOP;

Byte check refers to performing check detection in bytes. (The byte order is from high to low)

20.5.1.3 LSRX_CTL（0x08）

LSRX Interface Control Register

Bit	Bit Name	Function Description	Read/Write	Default Value
31:26	Reserved	Reserved	R	0
25:21	TIME_TH	Used to judge whether the data frame is not received due to timeout1. 1. When LSRX_CHLx does not receive data, timing starts from the enable of the LSRX module. If the frame header is not detected within the threshold time, the CHLx reception timeout flag and interrupt are generated; 2. When LSRX_CHLx receives the frame header, timing starts from the arrival time of this frame header. If the frame header is not detected within the threshold time, the CHLx reception timeout flag and interrupt are generated; Threshold time = (TIMR_TH + 1) * 8.888888ms When the metering communication cycle = 71ms, the threshold is configured as 0x08, which is greater than 71*105% = 74.55ms, and the actual timeout threshold is set between 79-87ms; When the metering communication cycle = 35ms, the threshold is configured as 0x04, which is greater than 35*105% = 36.75ms, and the actual timeout threshold is set between 40-48ms;	WR	0x09
20	LSRX_CHL3EN	Channel 3 Reception Enable	WR	0b0
19	LSRX_CHL2EN	Channel 2 Reception Enable	WR	0b0
18	LSRX_CHL1EN	Channel 1 Reception Enable	WR	0b0
17	LSRX_CHL0EN	Channel 0 Reception Enable	WR	0b0
16	IDLE_DET	LSTX Idle Detection Enable After enabling, after the UART port RX configuration is turned on, the UART RX is turned on only when LSTX is detected to be in the idle state. When this configuration is 0, the UART port RX is turned on directly after the RX configuration is turned on.	WR	0
15:12	CHL3_SEL	=0x0: Channel 3 selects UART0; =0x1: Channel 3 selects UART1; =0x2: Channel 3 selects UART2; =0x3: Channel 3 selects UART3; =0x4: Channel 3 selects UART4; =0x5: Channel 3 selects UART5;	WR	0x3

		=0x6: Channel 3 selects UART6； =0x7: Channel 3 selects UART7 ELSE: Reserved		
11:8	CHL2_SEL	=0x0: Channel 2 selects UART0; =0x1: Channel 2 selects UART1; =0x2: Channel 2 selects UART2; =0x3: Channel 2 selects UART3; =0x4: Channel 2 selects UART4; =0x5: Channel 2 selects UART5; =0x6: Channel 2 selects UART6 =0x7: Channel 2 selects UART7 ELSE: Reserved	WR	0x2
7:4	CHL1_SEL	=0x0: Channel 1 selects UART0; =0x1: Channel 1 selects UART1; =0x2: Channel 1 selects UART2; =0x3: Channel 1 selects UART3; =0x4: Channel 1 selects UART4; =0x5: Channel 1 selects UART5; =0x6: Channel 1 selects UART6 =0x7: Channel 1 selects UART7 ELSE: Reserved	WR	0x1
3:0	CHL0_SEL	=0x0: Channel 0 selects UART0; =0x1: Channel 0 selects UART1; =0x2: Channel 0 selects UART2; =0x3: Channel 0 selects UART3; =0x4: Channel 0 selects UART4; =0x5: Channel 0 selects UART5; =0x6: Channel 0 selects UART6 =0x7: Channel 0 selects UART7 ELSE: Reserved	WR	0x0

This register cannot be modified when LSRX_EN is 1.

20.5.1.4 LSRX_CAL_CFG（0x0C）

LSRX Calibration Output Configuration Register

Bit	Bit Name	Function Description	Read/Write	Default Value
31	Reserved	Reserved	R	0
30:23	PAR_DATA	Output calibration character, sent according to the communication baud rate	WR	0x2F
22:0	CAL_CNT	Low-level length between output characters (CAL_CNT + 1) cycles	WR	0x1FFFF

Each time this register is written, the register outputs a waveform once. The output calibration waveform is as shown in the following figure, where PAR_DATA uses even parity.

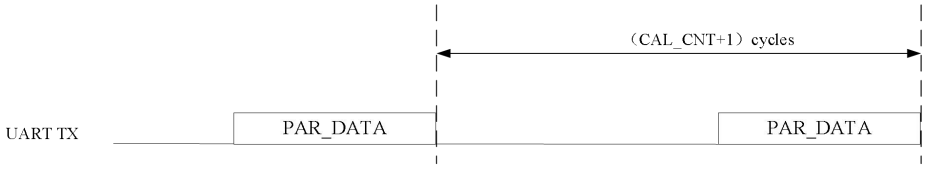


Figure 20-12 Schematic Diagram of Output Calibration Waveform

20.5.1.5 LSRX_EN（0x10）

LSRX Enable Register

Bit	Bit Name	Function Description	Read/Write	Default Value
31:1	Reserved	Reserved	R	0x0
0	LSRX_EN	Module Enable	WR	0b0

20.5.1.6 LSRX_IE（0x14）

LSRX Interrupt Enable Register

Bit	Bit Name	Function Description	Read/Write	Default Value
31:2	Reserved	Reserved	R	0
1	CAL_UPDATA_IE	LSRX Data Update Interrupt	WR	0
0	CAL_FINISH_IE	Calibration Waveform Transmission Completion Interrupt Enable	WR	0

20.5.1.7 LSRX_STA（0x18）

LSRX Status Flag Register

Bit	Bit Name	Function Description	Read/Write	Default Value
31:9	Reserved	Reserved	R	0x0
8	CHL3_Valid_IF	Channel 3 Reception Timeout Flag, cleared by writing 1	WR	0b0
7	CHL2_Valid_IF	Channel 2 Reception Timeout Flag, cleared by writing 1	WR	0b0
6	CHL1_Valid_IF	Channel 1 Reception Timeout Flag, cleared by writing 1	WR	0b0

5	CHL0_Valid_IF	Channel 0 Reception Timeout Flag, cleared by writing 1	WR	0b0
4	CHL3_UPDATA_IF	Channel 3 Data Update Flag, cleared by writing 1	WR	0b0
3	CHL2_UPDATA_IF	Channel 2 Data Update Flag, cleared by writing 1	WR	0b0
2	CHL1_UPDATA_IF	Channel 1 Data Update Flag, cleared by writing 1	WR	0b0
1	CHL0_UPDATA_IF	Channel 0 Data Update Flag, cleared by writing 1	WR	0b0
0	CAL_FINISH_IF	Calibration Waveform Transmission Completion Interrupt Flag, cleared by writing 1	WR	0b0

20.5.1.8 LSRX_CHL0FH_TIME（0x1C）

LSRX Channel 0 Frame Header Time Register

Bit	Bit Name	Function Description	Read/Write	Default Value
24:0	CHL0FH_TIME0	Last frame header time of Channel 0	R	0x0

When phase loss is detected, the frame header time registers of Channels 0-3 will be cleared synchronously.

20.5.1.9 LSRX_CHL1FH_TIME（0x20）

LSRX Channel 1 Frame Header Time Register

Bit	Bit Name	Function Description	Read/Write	Default Value
24:0	CHL1FH_TIME0	Last frame header time of Channel 1	R	0x0

When phase loss is detected, the frame header time registers of Channels 0-3 will be cleared synchronously.

20.5.1.10 LSRX_CHL2FH_TIME（0x24）

LSRX Channel 2 Frame Header Time Register

Bit	Bit Name	Function Description	Read/Write	Default Value
24:0	CHL2FH_TIME0	Last frame header time of Channel 2	R	0x0

When phase loss is detected, the frame header time registers of Channels 0-3 will be cleared synchronously.

20.5.1.11 LSRX_CHL3FH_TIME（0x28）

LSRX Channel 3 Frame Header Time Register

Bit	Bit Name	Function Description	Read/Write	Default Value
24:0	CHL3FH_TIME0	Last frame header time of Channel 3	R	0x0

When phase loss is detected, the frame header time registers of Channels 0-3 will be cleared synchronously.

20.6 Software Operation Flow

20.6.1 Calibration Waveform Output

- Configure the LSRX Calibration Output Configuration Register according to requirements: select the output port, and set the output special character and low-level duration;
- Configure the LSRX Interrupt Enable Register to enable the calibration waveform transmission completion interrupt;
- Configure the LSRX Enable Register to turn on the LSRX module;
- Configure the LSRX Calibration Output Configuration Register; transmission starts after configuration is completed;
- Wait for the interrupt to be asserted to complete one calibration waveform output;

21 LPUART

21.1 Introduction

The Low-Power Universal Asynchronous Receiver/Transmitter (LPUART) module enables full-duplex UART communication with limited power consumption. A working clock configured to 32.768kHz is sufficient to achieve 9600 baud rate communication. LPUART can operate in low-frequency mode to implement low-power asynchronous data transmission and reception. When the CPU is in SLEEP mode, LPUART can generate a wake-up interrupt to quickly wake the CPU upon detecting a wake-up event.

The LPUART protocol includes two data lines: LPUART_TX (transmit data line) and LPUART_RX (receive data line). The module's functions mainly consist of three parts: configuring the data frame format for serial data transmission; detecting the start bit for data reception; and waking up from low-power mode.

21.2 Features

- Maximum 9600 baud rate at 32.768kHz working clock, with a baud rate range of 300~9600bps
- Transmit and receive buffers each support independent 4-byte FIFO
- Supports data bit widths of 5/6/7/8 bits, and 1 or 2 stop bits
- Parity bit supports odd parity, even parity, all 0s/all 1s, no parity bit, or user-defined parity bit
- Supports 4 low-power wake-up modes: RX falling edge wake-up, start bit detection wake-up, data reception completion wake-up (single frame or multiple frames), and one-frame data matching wake-up
- Independent enable for transmit and receive interrupts
- Supports 38kHz infrared modulation output
- Reception supports idle frame detection

21.3 Block Diagram

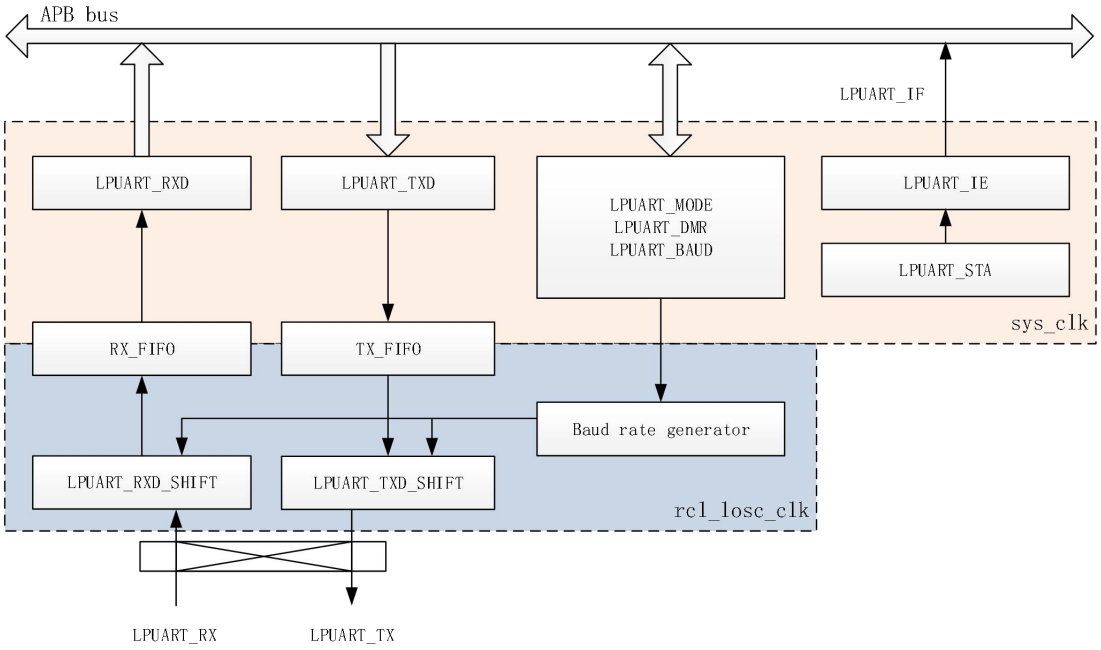


Figure 21-1 LPUART Module Structure Diagram

21.4 Interface Timing

By programming the DATLEN bit in the LPUART_MODE register, the data frame length can be set to 5~8 bits. A data frame format consists of: start bit + data bits + parity bit + stop bit.

By default, the start bit of TX and RX is low level, and the stop bit is high level. The polarity of each bit signal can also be inverted through polarity configuration. The parity bit is set at the last bit of the data, and the stop bit can be set to 1 bit or 2 bits. If the user-defined parity bit is selected and high-order data is sent first, the parity bit is transmitted first.

An idle frame is a continuous high level of one data frame length including the start bit and stop bit, calculated from the end of the stop bit of the previous frame. The length of the idle frame varies with the data length.

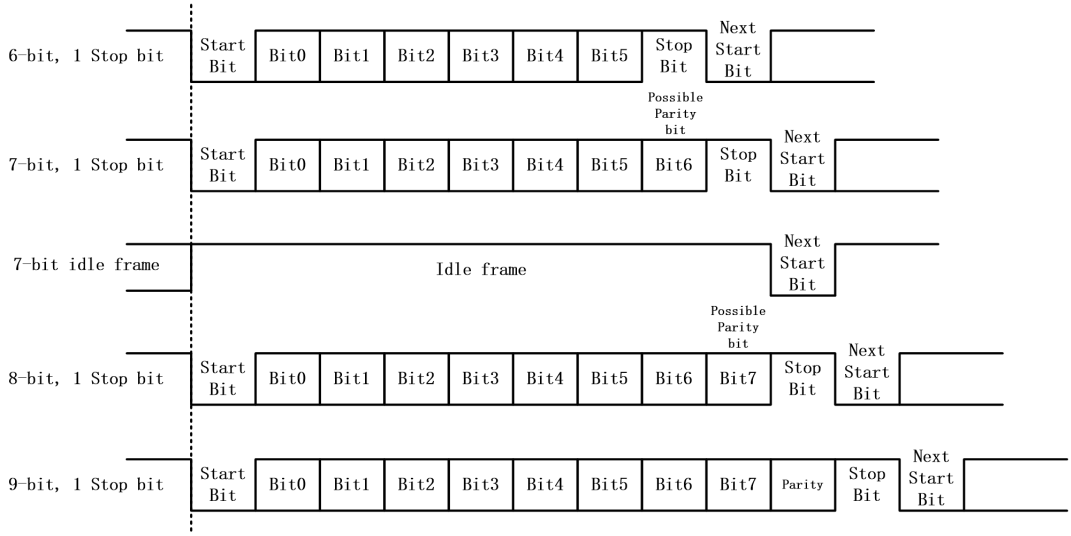


Figure 21-2 LPUART Data Frame Structure

21.5 Functional Description

21.5.1 FIFO Function of LPUART

LPUART has a 9-bit wide transmit FIFO and receive FIFO, both with a depth of 4. Interrupt flags are set when the transmit FIFO is empty/half-empty and the receive FIFO is non-empty/full/half-full, and interrupt enable can be configured. The transmit and receive FIFOs are reset when transmit enable and receive enable are turned on, respectively.

When the transmitter is enabled, writing data to the LPUART_TXD register adds a data to the TXFIFO, and the written data queues in the TXFIFO. When the TXFIFO empty and half-empty flags are set, write operations can continue without checking if the TXFIFO is completely empty each time:

① When the TXFIFO empty flag is generated, the software controls writing 4 bytes to the TXFIFO;

② When the TXFIFO half-empty flag is generated, the software controls writing 2 bytes to the TXFIFO.

The non-empty/half-full/full flags of the receiver RXFIFO all indicate that data can be read. Reading LPUART_RXD returns the earliest data input to the RXFIFO.

① When the RXFIFO is full, an interrupt can be generated to allow the software to read 4 bytes of data at a time;

② When the RXFIFO is half-full, an interrupt can be generated to allow the software to read 2 bytes of data at a time;

③ When the receiver detects an idle frame, it is deemed that the current data packet transmission is completed, and the baud rate generator enable is automatically turned off. If the interrupt signal is enabled, an idle interrupt can

be generated for the software to read data from the RXFIFO.

21.5.2 Transmit Engine

21.5.2.1 Transmit Characteristics

The transmit engine includes a transmit control register, a transmit buffer register, and a transmit shift register. The transmit engine can transmit data length, parity bit, and stop bit width according to the configuration of the control register.

Set the transmit enable bit TE in the transmit control register during the first transmission; TE cannot be reset during data transmission. During LPUART transmission, the TX pin transmits the least significant bit (LSB) first by default.

Writing data to be transmitted to LPUART_TXD adds a data to the TXFIFO, and the written data queues in the TXFIFO. Reading the LPUART_TXD register returns the earliest data in the TXFIFO.

After all data is written, it is necessary to wait until both the TXIF bit and TXFFIF bit of LPUART_STA are equal to 1, indicating that all frames have been transmitted and both the TXFIFO and shift register are empty, before entering SLEEP mode.

21.5.2.2 Single-Byte Communication Process

The TXFNFIF flag is set to 1 by hardware, indicating that the TXFIFO is not full, and the next data can be written to the LPUART_TXD register without overwriting old data.

During transmission, writing instructions to the LPUART_TXD register store data in the TXFIFO. When the current ongoing transmission ends, the data is copied to the shift register. When the TXFIFO is empty, TXFEIF is 1, indicating that 4 bytes can be written to the TXFIFO; when half-empty, TXFHEIF is 1, indicating that 2 bytes can be continuously written. If the TXFIFO interrupt is enabled, empty/half-empty interrupts can be generated.

If the current frame has been transmitted, indicating that the buffer register is empty, TXIF will become high level at the last stop bit of a frame. If TXIE is set, an interrupt can be generated. After writing the last data to the register, it is necessary to wait until TXIF=1 and TXFEIF=1 before disabling LPUART or the MCU entering SLEEP mode.

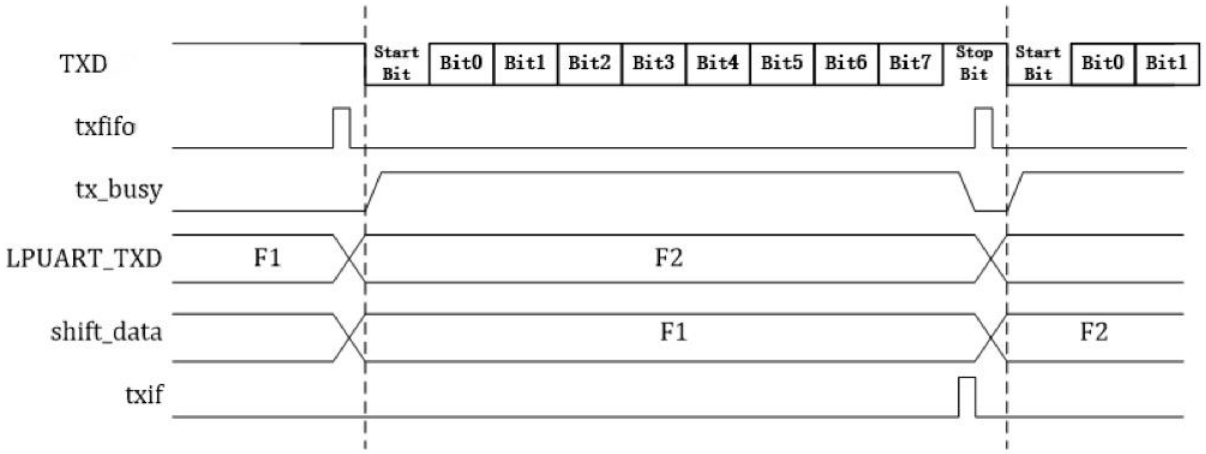


Figure 21-3 Single-Byte Transmit Timing Diagram

21.5.3 Receive Engine

21.5.3.1 Transmission Characteristics

Data bit width is 5/6/7/8 bits, set by the DATLEN bit and stop bit length STOPS in the LPUART_MODE register.

During LPUART reception, data first shifts the least significant bit through the RX pin. If internal loopback is enabled, the transmitter TX and receiver RX are short-circuited internally in the module.

The RXFNEIF bit is set to 1, indicating that the RXFIFO is non-empty. Reading LPUART_RXD returns the earliest data input to the RXFIFO. When the RXFIFO is empty, the RXFNE bit is cleared. When the RXFIFO is full, the first data in the RXFIFO must be read before the next character is received to avoid overflow errors.

If a frame error, noise error, or overflow error is detected during reception, the corresponding error flag bit is set.

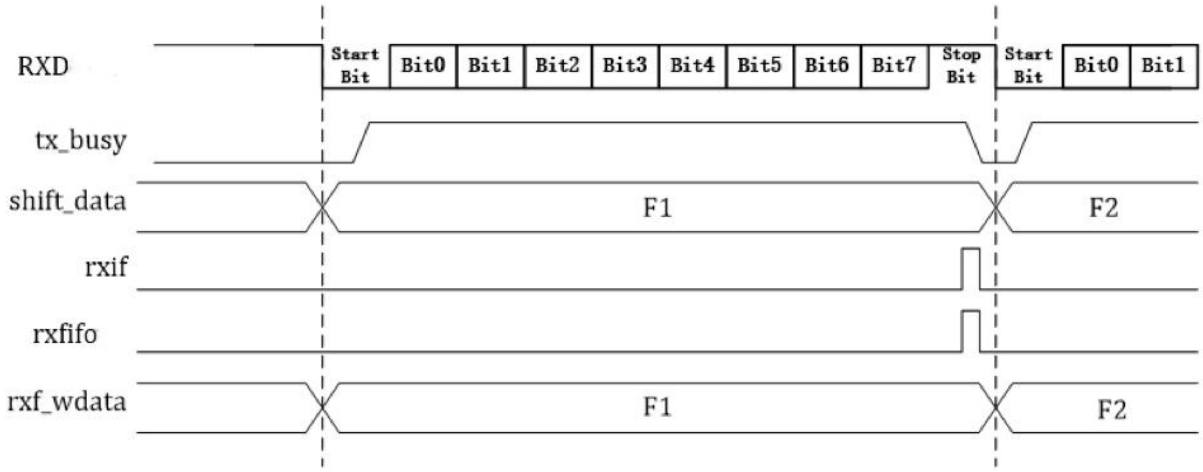


Figure 21-4 Single-Byte Receive Timing Diagram

21.5.3.2 Start Bit Detection

Detect the falling edge of the RX line, then sample at the middle position of the start bit to confirm it remains 0. If the start bit sample is judged to be 1, the noise error flag NEIF is set, the start bit is discarded, and the receiver waits for a new start bit.

21.5.3.3 Stop Bit Configuration

1 stop bit: Sample at the stop bit sampling position

2 stop bits: Sample at the position of the second stop bit; the first stop bit is not used to check frame format errors

21.5.3.4 Idle Frame Detection

When the RXFIFO is enabled, if the RXFIFO sets the half-full or full flag but does not reach the FIFO threshold after receiving the last frame, the FIFO may keep waiting for data reception, resulting in cached data not being read in time. Idle frame detection needs to be used for judgment.

Software usage example: When receive enable is turned on and data is received, idle frame detection is enabled. When a high level on the RX line exceeding one data frame length is detected, an idle frame interrupt flag is generated. If the interrupt is enabled, the corresponding interrupt service routine is entered. When the flag is set,

idle frame detection is stopped until data is received next time and restarted. If the idle frame interrupt is enabled, an interrupt flag can be generated.

21.5.3.5 Receive Error Judgment

- Overflow Error

An overflow error occurs when the receive shift register is ready to transfer data to the FIFO and the receive FIFO is full. Data cannot be transferred from the shift register to the LPUART_RXD register until there is a free position in the RXFIFO. The RXFIFO sets full and half-full flags. If new data is received when RXFFIF=1, an overflow error is generated. When an overflow error occurs: the OEIF bit is set to 1, data in the RXFIFO is not lost, the shift register is overwritten, and all data received during the overflow is lost.

- Frame Format Error

When the stop bit is not recognized at the expected reception time (i.e., the stop bit sample value = 0). When a frame error is detected: FEIF is set by hardware, and invalid data is transferred from the shift register to LPUART_RXD. During single-byte communication, the module generates an interrupt by the RXFNE bit. If a frame format error occurs, FEIF is set.

21.5.4 Baud Rate Generator

Since LPUART cannot directly obtain the required baud rate through integer division, fractional division is required to achieve 9600bps. In the LPUART module, it is generated using the pre-stage dual-mode fractional division method. The register bitwise division configuration corresponding to different baud rates needs to be configured separately, and the bit modulation number of Bit0 must be set to 1.

Table 21-1 LPUART Bitwise Division Configuration

Baud Rate/bps	LPUART_BAUD.BAUD	LPUART_BAUD.MCTL											
		Bit 0	Bit 1	Bit 2	Bit 3	Bit 4	Bit 5	Bit 6	Bit 7	Bit 8	Bit 9	Bit 10	Bit 11
300	101/110/111	1	1	0	0	0	0	0	0	0	0	0	1
600	100	1	1	1	1	1	1	1	0	0	0	0	0
1200	011	1	1	1	0	0	0	0	0	0	0	1	0
2400	010	1	1	0	1	1	0	1	1	0	1	1	0
4800	001	1	0	1	1	1	1	1	0	1	1	1	1
9600	000	1	0	1	0	0	1	0	1	0	0	1	0

21.5.5 Parity Bit Configuration

Set the PARS parity bit in the LPUART_MODE register. The parity bit can be configured during data transmission with lengths of 5/6/7/8 bits.

Parity check during reception: If the parity check fails, the PEIF bit of the LPUART_STA register is set to 1; if the PEIE bit is 1, an interrupt is generated. The receiver does not check the user-defined parity bit.

Parity generation during transmission: If the PARS bit is not 000 or the user-defined parity bit is selected, the data written in the data register is transmitted first, followed by a parity bit (even parity: the number of 1s is even; odd parity: the number of 1s is odd). If the user-defined parity bit is selected at the transmitter, the parity bit is transmitted first when LMSB=1.

21.5.6 LPUART Low-Power Wake-Up Mode

LPUART receives data and wakes up the CPU in Sleep mode. Before the system core clock is turned off, it is necessary to use registers to configure the wake-up mode and wake-up interrupt enable until a specific event occurs to wake the chip out of sleep mode. LPUART supports 4 low-power wake-up events, which can be configured in registers: RX falling edge wake-up, start bit detection wake-up, data reception completion wake-up, and one-frame data matching wake-up.

Before entering low-power mode, LPUART ensures that no data transmission or reception is in progress (check the BUSY flag). After entering sleep mode, the core clock is turned off. The LPUART module clock is not turned off during sleep, and data reception is automatically turned on when an RX falling edge is detected. It is also necessary to configure the WUSEL of LPUART_MODE in advance to select the wake-up interrupt event, set the WUFIE bit to 1 to enable the LPUART low-power wake-up interrupt, set RE to 1 to turn on receive enable, and then enter SLEEP mode.

21.5.6.1 RX Falling Edge Detection Wake-Up

The chip remains monitoring the RX pin in sleep mode. Configure the RX falling edge detection interrupt enable WUSEL of the LPUART_MODE register. When the receiver detects an RX falling edge event in sleep mode, it turns on the module clock, automatically starts data reception, and requests to wake the CPU. The falling edge wake-up does not perform noise detection on the start bit。

21.5.6.2 Start Bit Detection Wake-Up

Configure the register for start bit detection wake-up. After monitoring the RX falling edge, sample at the middle position of the start bit. If the sample value is 0, it is judged as the start bit, and a wake-up event is sent to the MCU at the end of the start bit.

21.5.6.3 Data Reception Completion Wake-Up

In sleep mode, after monitoring the falling edge of the RX terminal, start receiving data and buffering it. After the data reception is completed, an interrupt is directly generated to wake the CPU without data matching.

The wake-up interrupt sources of LPUART can be set to:

RXFNEIF flag (RXFIFO is non-empty, 1 byte has been received), RXFNEIE must be set to 1 before sleep.

RXFHFIF flag (RXFIFO is half-full, 2 bytes have been received), RXFHFIE must be set to 1 before sleep.

RXFFIF flag (RXFIFO is full, 4 bytes have been received), RXFFIE must be set to 1 before sleep.

21.5.6.4 One-Frame Data Matching Wake-Up

Configure the MDATA data in the data matching register in advance. During the first frame reception in sleep mode, compare the first frame data with the MDATA data bit by bit. If the data matching is successful, the MDATIF flag bit is set to 1, triggering an interrupt, which can be used for data reception wake-up in sleep mode; if the data matching fails, the received data is ignored. In sleep mode, if RXFNEIF=0, the next received data frame is determined as the first frame for data matching.

21.6 Registers

Register Base Address of LPUART Module

Module Name	Core0 Base Address	Core1 Base Address
LPUART	0x2002_0000	0xA002_0000

Register Offset Address of LPUART Interface

Register Name	Address Offset	Description
LPUART_MODE	0x0	LPUART Mode Configuration Register
LPUART_IE	0x4	LPUART Interrupt Enable Register
LPUART_STA	0x8	LPUART Status Indication Register
LPUART_BAUD	0xC	LPUART Baud Rate Modulation Register
LPUART_TXD	0x10	LPUART Transmit Data Buffer Register
LPUART_RXD	0x14	LPUART Receive Data Buffer Register
LPUART_DMR	0x18	LPUART Data Matching Register

Note: LPUART_MODE & LPUART_BAUD are prohibited from being modified during transmission.

21.6.1 LPUART_MODE（0x00）

LPUART Mode Configuration Register

Reset Value: 0x0

Bit	Bit Name	Function Description	Read/Write	Reset Value
31:15	---	Reserved	R	0x0
14:13	WUSEL	Wake-Up Interrupt Event Configuration 00: START start bit detection 01: RX falling edge detection 10: Data reception completion 11: One-frame data matching success	RW	0x0
12	ILBE	Internal Loopback Enable Bit 0: Internal loopback disabled 1: Internal loopback enabled, TX and RX are short-circuited internally in the module	RW	0
11	LMSB	LSB/MSB Transmission Mechanism Selection Mode 0: LSB transmitted first 1: MSB transmitted first Note: When PARS is selected as user-defined parity, the parity bit is treated as the highest extended bit of data. If the selection mode is MSB at this time, the parity bit is transmitted first	RW	0x0
10	STOPS	Stop Bit Width Selection 0: 1-bit stop bit 1: 2-bit stop bit	RW	0x0
9:7	PARS[2:0]	Parity Bit Selection Bit 000: No parity001: Odd parity 010: Even parity 011: Fixed 0 parity	RW	0x0

		100: Fixed 1 parity 101/110/111: User-defined parity		
6:5	DATLEN[1:0]	Transmission Data Width Bit (excluding parity bit) 00: 5-bit 01: 6-bit 10: 7-bit 11: 8-bit	RW	0x0
4	IRPOL	Infrared Modulation Output Polarity Selection 0: Positive polarity, i.e., the default drive level remains high, low-level modulation output 1: Negative polarity, i.e., the default drive level remains low, high-level modulation output Note: IRPOL only determines the level during idle output (invalid level) and does not affect the level during valid data	RW	0x0
3	NEG	Transmit and Receive Data Polarity Configuration 0: Positive polarity, default drive level is high 1: Negative polarity, default drive level is low	RW	0x0
2	IRE	Infrared Modulation Output Function Configuration 0: Disabled 1: Enabled	RW	0x0
1	RE	LPUART Receiver Enable Bit 0: Disabled 1: Enabled	RW	0x0
0	TE	LPUART Transmitter Enable Bit 0: Disabled 1: Enabled	RW	0x0

21.6.2 LPUART_IE（0x04）

LPUART Interrupt Enable Register

Default Reset Value: 0x0

Bit	Bit Name	Function Description	Read/Write	Reset Value
31:14	---	Reserved	R	0x0
13	RXFFIE	Receive FIFO Full Interrupt Flag Enable 0: Not enabled 1: Enabled	RW	0x0
12	RXFHFIE	Receive FIFO Half-Full Interrupt Flag Enable 0: Not enabled 1: Enabled	RW	0x0
11	RXFNEIE	Receive FIFO Non-Empty Interrupt Flag Enable 0: Not enabled 1: Enabled	RW	0x0
10	TXFHEIE	Transmit FIFO Half-Empty Interrupt Flag Enable	RW	0x0

		0: Not enabled 1: Enabled		
9	TXFEIE	Transmit FIFO Empty Interrupt Flag Enable 0: Not enabled 1: Enabled	RW	0x0
8	IDLEIE	Idle Frame Detection Interrupt Enable 0: Interrupt disabled 1: Interrupt enabled	RW	0x0
7	NEIE	Start Bit Noise Detection Enable 0: Interrupt disabled 1: Interrupt enabled	RW	0x0
6	DEIE	Data Error Enable 0: Interrupt disabled 1: Interrupt enabled	RW	0x0
5	FEIE	Frame Format Error Enable 0: Interrupt disabled 1: Interrupt enabled	RW	0x0
4	OEIE	Overflow Error Enable 0: Interrupt disabled 1: Interrupt enabled	RW	0x0
3	PEIE	Parity Error Enable 0: Interrupt disabled 1: Interrupt enabled	RW	0x0
2	WUFIE	Wake-Up Interrupt Enable from Sleep Mode 0: Interrupt disabled 1: Interrupt enabled Note: The interrupt enable must be turned on before entering sleep mode, and WUF is only valid in sleep mode. The software disables the interrupt enable after waking up.	RW	0x0
1	RXIE	Receive Data Completion Interrupt Enable 0: Interrupt disabled 1: Interrupt enabled	RW	0x0
0	TXIE	Transmit Data Completion Interrupt Enable 0: Interrupt disabled 1: Interrupt enabled	RW	0x0

21.6.3 LPUART_STA（0x08）

LPUART Status Indication Register

Reset Value: 0x0

Bit Name	Function Description	Read/Write	Reset Value	
31:23	---	Reserved	R	0x0

22:20	RXFCNT[2:0]	Number of Data in Receive FIFO	R	0x0
19:17	TXFCNT[2:0]	Number of Data in Transmit FIFO	R	0x0
16	RXFFIF	Receive FIFO Full Interrupt Flag, cleared by writing 1 0: Not full 1: Full	RW	0x0
15	RXFHFIF	Receive FIFO Half-Full Interrupt Flag, indicating 2 data in the receive FIFO, cleared by writing 1 0: Not half-full 1: Half-full	RW	0x0
14	RXFNEIF	Receive FIFO Non-Empty Interrupt Flag, cleared by writing 1 0: Empty 1: Non-empty	RW	0x0
13	TXFF	Transmit FIFO Full Flag, cleared by writing 1 or automatically cleared by hardware 0: Not full 1: Full	RW	0x0
12	TXFHEIF	Transmit FIFO Half-Empty Interrupt Flag, indicating 2 free positions in the transmit FIFO, cleared by writing 1 0: Not half-empty 1: Half-empty	RW	0x0
11	TXFEIF	Transmit FIFO Empty Interrupt Flag, cleared by writing 1 0: Not empty 1: Empty	RW	0x1
10	RBUSY	Receive Status Flag 0: No reception in progress 1: Receiving data	R	0x0
9	TBUSY	Transmit Status Flag 0: No transmission in progress 1: Transmitting flag	R	0x0
8	IDLEIF	Idle Frame Detection Interrupt Flag, cleared by writing 1 0: No idle frame detected 1: Idle frame detected	RW	0x0
7	NEIF	Start Bit Noise Detection Flag, cleared by writing 1 0: No noise 1: Noise present	RW	0x0
6	DEIF	Data Error Flag, cleared by writing 1 This bit is set when writing to the LPUART transmit register after the LPUART transmit FIFO is full 0: No error 1: Error present	RW	0x0
5	FEIF	Frame Format Error Flag, cleared by writing 1 Flag bit indicating that the data received by LPUART does not conform to the frame format, i.e., the received stop bit is 0 instead	RW	0x0

		of 1. This bit is set 0: No error 1: Error present		
4	OEIF	Overflow Error Flag, cleared by writing 1 Reception overflow caused by failure to read the LPUART receive data buffer register in time 0: No error 1: Error present	RW	0x0
3	PEIF	Parity Check Error Flag, cleared by writing 1 0: No error 1: Error present	RW	0x0
2	WUF	Receive Wake-up Event Flag, cleared by writing 1 0: Wake-up event not occurred 1: Wake-up event occurred Wake-up event selected by WUSEL: RX falling edge wake-up, start bit detection wake-up, data reception complete wake-up, one frame data match wake-up.	RW	0x0
1	RXIF	Receive Complete Interrupt Flag, cleared by writing 1 0: Receive data not complete 1: Receive data complete, can be read from data buffer register Triggered upon completion of each frame reception.	RW	0x0
0	TXIF	Transmit Complete Interrupt Flag, cleared by writing 1 0: Transmit not complete 1: Transmit complete Triggered after the STOP bit of the current transmitted byte has been sent.	RW	0x0

Note: Software should read/write data first upon entering the FIFO interrupt, then clear the flag bit.

21.6.4 LPUART_BAUD（0xC）

LPUART Baud Rate Modulation Register

Reset Value: 0x0

Bit	Bit Name	Functional Description	R/W	Reset Value
31:28	---	Reserved	R	0x0
27:16	MCTL	Corresponds to the bit width modulation signal for each LPUART bit.	RW	0x0
15:3	-----	Reserved		0x0
2:0	BAUD	Baud Rate Control (bps) 000: 9600 001: 4800 010: 2400 011: 1200 100: 600	RW	0x0

		101/110/111: 300		

LPUART_BAUD is prohibited from modification during transmission. LPUART_BAUD[16] must be set to 1.

21.6.5 LPUART_TXD（0x10）

LPUART Transmit Data Buffer Register

Reset Value: 0x0

Bit	Bit Name	Functional Description	R/W	Reset Value
31:9	---	Reserved	R	0x0
8	UP	User-defined Parity Bit	RW	0x0
7:0	TXDATA	Transmit Data Register	RW	0x0

21.6.6 LPUART_RXD（0x14）

LPUART Receive Data Buffer Register

Reset Value:0x0

Bit	Bit Name	Functional Description	R/W	Reset Value
31:9	---	Reserved	R	0x0
8	UP	Parity Bit	R	0x0
7:0	RXDATA	Receive Data Register	R	0x0

21.6.7 LPUART_DMR（0x18）

LPUART Data Match Register

Reset Value:0x0

Bit	Bit Name	Functional Description	R/W	Reset Value
31:8	---	Reserved	R	0x0
7:0	MDATA	Compares with the first frame of received data. If a data match is detected, triggers a wake-up interrupt, used for receive wake-up in sleep mode.	RW	0x0

Note: This register can only be written when reception is disabled.

21.7 Software Configuration Flow and Application Notes

21.7.1 Transmit Flow

- Configure the LPUART configuration register LURT_CFG[2:0] to select the LPUART and UARTx multiplexed IO configuration, and configure the multiplexed pins;
- Write 1 to the LPUART_EN bit of the module enable register MOD0_EN[25] to enable the LPUART module;
- Set the character length, stop bit width, parity mode, baud rate, interrupt enable, and other configurations;
- Write 0x1ffff to clear the LPUART_STA register status;
- Configure LPUART interrupt enable and priority, and write the interrupt service routine. LPUART receive, transmit, and error interrupts share the same interrupt entry. It is necessary to judge the type of interrupt according to the interrupt enable bit and status flag.
- Set the TE bit of LPUART_MODE to 1 to enable the transmitter during the first transmission;

- Write data to the transmit data buffer register;
- Wait for the interrupt event and continue writing data to the transmit data buffer register.

21.7.2 Receive Flow

- Configure the LPUART configuration register LURT_CFG[2:0] to select the LPUART and UARTx multiplexed IO configuration, and configure the multiplexed pins;
- Write 1 to the LPUART_EN bit of the module enable register MOD0_EN[25] to enable the LPUART module;
- Set the character length, stop bit width, parity mode, baud rate, interrupt enable, and other configurations;
- Write 0x1ff to clear the LPUART_STA register status;
- Configure LPUART interrupt enable and priority, and write the interrupt service routine. LPUART receive, transmit, and error interrupts share the same interrupt entry. It is necessary to judge the type of interrupt according to the interrupt enable bit and status flag;
- Set the RE bit of LPUART_MODE to 1 to enable the receiver before the first reception;
- Wait for the interrupt event and process the received data and errors.

22 7816

Built-in two ISO7816 channels, supporting connection of 2 external devices with 7816 protocol interfaces.

22.1 Features

The ISO7816 interface controller has the following characteristics:

- Supports the standard ISO7816 protocol and operates in master mode;
- Provides card clock output with a frequency configurable between 1～5MHz;
- Dual data channel interfaces, supporting 2 external 7816 protocol interface devices;
- Supports multiple division ratio settings for 7816;
- Supports data encoding modes: low logic with MSB first output and high logic with LSB first output;
- Supports error signal width settings of 1 or 2 ETU;
- Supports EGT settings with a width of 0～254 ETU;
- Supports automatic retransmission mechanism for transmission errors, with retransmission times configurable between 0～3;
- The 7816 card protocol stack supports two interfaces (Esam and card);
- When interfacing with a card and using optocoupler isolation, it is recommended to separate reception and transmission.

22.2 Register Description

Base Addresses of ISO7816 Interface Registers

Module Name	Core0 Base Address	Core1 Base Address
ISO78160	0x20030000	0xA0030000
ISO78161	0x20034000	0xA0034000

Offset Addresses of ISO7816 Interface Registers

Register Name	Address Offset	Description
ISO7816_CTL	Offset+0x0	ISO7816 Control Register
ISO7816_CLK	Offset+0x4	ISO7816 Clock Configuration Register
ISO7816_BDDIV	Offset+0x8	ISO7816 Baud Rate Configuration Register
ISO7816_STAT	Offset+0xc	ISO7816 Status Indication Register
ISO7816_DAT	Offset+0x10	ISO7816 Data Register

22.2.1 ISO7816x_CTL

ISO7816 Control Register

Bit	Name	Description	R/W	Reset Value
31	CARD_CHECK_EN	Card Removal Detection Enable Bit. Valid only after OLD detection function is enabled. 1: Enable card removal detection interrupt function. 0: Disable card removal detection interrupt function.	RW	0
30	OLD_IRQ_EN	OLD Detection Interrupt Function Enable. Valid only after OLD detection function is enabled.	RW	0

		1: Enable OLD detection interrupt function. 0: Disable OLD detection interrupt function.		
29	OLD_EN	OLD Detection Function Enable. 1: Enable OLD detection function. 0: Disable OLD detection function.	RW	0
28	RX_GT0	Receive Data GT Selection Bit. 1: Receive data GT is 1 etu. 0: Receive data GT is 2 etu.	RW	0
27	TX_GT0	Transmit Data GT Selection Bit. 1: Transmit data GT is 1 etu. 0: Transmit data GT is 2 etu.	RW	0
26	RESERVED	Reserved	R	0
25	IO_EN	Data Bidirectional Enable Signal. 1: 7816_IO port is bidirectional. 0: 7816_IO port is unidirectional, output only; data input comes from the 7816_I port.	RW	1
24:17	EGT	EGT Width Selection Value (0~255), Extra Guard Time N. Default value N=0. In the range 0 to 254, N represents the following delay from the leading edge of the previous character (also sent by the card or interface device) required by the card before being ready to receive the next character: 12 etu + (Q×(N/f)) In the formula, Q should take one of the two values: F/D, i.e., the value used to calculate the etu, when T=15 is not present in the ATR; Fi/Di, when T=15 is present in the ATR. N=255 indicates that during the transmission protocol, the minimum delay between the leading edges of two consecutive characters is the same in both transmission directions. This minimum delay value is: 12 etu for T=0 11 etu for T=1	RW	0
16:14	REP_CNT	Automatic Retransmission Count Control for Data Parity Errors. 000: 0 times; 001: 1 time; 010: 2 times; 011: 3 times; 100: 4 times; 101: 5 times; 110: 6 times; 111: 7 times;	RW	011
13	RXPAR_ESEL	Receive Data Parity Error Handling Mode Selection. 1: Parity error, send error signal back according to T=0 protocol. Set RX_PAR_ERR flag, generate interrupt. 0: Parity error, do not send error signal, set RX_PAR_ERR	RW	1

		flag, generate interrupt directly.		
12:11	ERRWTH	Error Signal Width Selection Bits. 00: 2 etu 01: 1 etu 10: 1.5 etu 11: 2 etu	RW	01
10:8	PARSEL	Parity Bit Selection Bits. 000: No parity 001: Odd parity 010: Even parity 011: Fixed 0 parity 100: Fixed 1 parity Others: User-defined parity	RW	010
7	BGT_EN	BGT Control Bit between Data Reception and Transmission. 0: Disable BGT function, no BGT inserted between data reception and transmission. 1: Enable BGT function, insert BGT between data reception and transmission.	RW	0
6	ERR_IRQ_EN	Transmission Error Interrupt Enable Bit. Includes data collision during transmission, data overflow during reception, and receive data frame format error. 0: Disable interrupt generation on transmission error. 1: Enable interrupt generation on transmission error.	RW	0
5	RX_IRQ_EN	Data Receive Interrupt Enable Bit. Enables interrupt generation when data is shifted from the shift register into the receive buffer register. 0: Disable interrupt generation on data reception. 1: Enable interrupt generation on data reception.	RW	0
4	TX_IRQ_EN	Data Transmit Interrupt Enable Bit. Enables interrupt generation when data is shifted from the transmit buffer register into the shift register. 0: Disable interrupt generation on data transmission. 1: Enable interrupt generation on data transmission.	RW	0
3	RX_EN	Receive Data Enable. 0: Disable data reception. 1: Enable data reception.	RW	0
2	TX_EN	Transmit Data Enable. 0: Disable data transmission. 1: Enable data transmission.	RW	0
1	DIRSEL	Data Encoding Mode Selection Bit. 0: LSB first, positive logic data encoding. 1: MSB first, negative logic data encoding.	RW	0

0	EN	ISO7816 Controller Enable Bit. 0: Controller disabled. 1: Controller enabled.	RW	0

22.2.2 ISO7816x_CLK

ISO7816 Clock Control Register

Bit	Name	Description	R/W	Reset Value
31:4	---	Reserved	R	0
3	CLKO_EN	Card Clock Output Enable Bit. 0: Disable card clock output. 1: Enable card clock output.	RW	0
2:0	CLKDIV	ISO7816 Clock Output CLK_O Divider Ratio. The source clock for the ISO7816 module is directly derived from the system clock fsyspll. 000: No division; 001: Divide by 2; 010: Divide by 4; 011: Divide by 8; 100: Divide by 16; 101: Divide by 32; Other configuration values: Divide by 32;	RW	010

22.2.3 ISO7816x_BDDIV

ISO7816 Baud Rate Divider Register

Bit	Name	Description	R/W	Reset Value
31:23	---	Reserved	R	0
22	FDS_EN	Enable software configuration of F/D coefficient. 1: Baud rate coefficient is determined by software writing to FDS. 0: Baud rate coefficient is determined by FD.	RW	0
21:8	FDS	Software-configured baud rate coefficient value. This bit is writable only when FDS_EN=1; otherwise, it is 14'd372.	RW	14'd372
7:0	FD	8-bit FI and DI conveyed by the Answer To Reset.	RW	8'h01

22.2.4 ISO7816x_STAT

ISO7816 Status Register

Bit	Name	Description	R/W	Reset Value
31:14	---	Reserved	R	0
13	CARD_OUT_FLAG	Valid after CARD_CHECK_EN is enabled. Cleared by writing 1. 1: Card removal detected (detected high-level pulse width at input port greater than 50ms). 0: Card removal not detected (detected high-level pulse width at	R	0

		input port not greater than 50ms).		
12	OLD_FLAG	Valid after OLD_EN is enabled. Cleared by writing 1. This flag corresponds to the interrupt flag for receiving OLD signal in the special-chip . 1: OLD signal received. 0: OLD signal not received.	RW	0
11	FRAME_ERR	Receive Data Frame Format Error Interrupt Flag. Cleared by writing 1. 1: Transmit/Receive data frame format error, generates interrupt when transmission error interrupt enable is active. 0: No transmit/receive data frame format error.	RW	0
10	BDDIV_R	Baud Rate Matching Indication. Indicates whether FI and DI match. FD defaults to 8'h01, clock matches. Set to 1 when the written FD value does not match. 1: Match. 0: Mismatch.	R	1
9	TX_FLAG	Transmit Buffer Empty Flag. Automatically set after power-on reset, indicating buffer is empty and data can be written. Automatically cleared after MCU writes data, set to 1 after data is shifted from transmit buffer register to shift register. 1: Data transmit buffer empty. 0: Data present in transmit buffer waiting to be sent.	R	1
8	RX_FLAG	Data Receive Complete Flag. The 7816 interface controller generates an interrupt for each received byte, according to the receiving channel. Set by hardware, cleared by reading the data receive buffer register. 1: 1 byte received, data receive buffer full. 0: No data received, data receive buffer empty.	R	0
7	RXBUSY	Data Receive Busy Flag. Set by hardware, cleared by software. Automatically set/cleared by hardware. 0: Data reception idle. 1: Receive shift register is receiving data, automatically set upon receiving start bit, automatically cleared upon receiving stop bit.	R	0
6	TXBUSY	Data Transmit Busy Flag. Set by hardware, cleared by software. Automatically set/cleared by hardware. 0: Data transmission idle. 1: Transmit shift register is transmitting data, set when start bit is transmitted, automatically cleared when stop bit is transmitted.	R	0
5	TXPAR_ERRIF	Transmit Data Parity Error Flag. Set by hardware, cleared by software. Writing 1 clears this bit.	R	0

		1: Parity error occurred during data transmission. 0: No parity error occurred during data transmission.		
4	RXPAR_ERRIF	Receive Data Parity Error Flag. Set by hardware, cleared by software. Writing 1 clears this bit. 1: Parity error occurred during data reception. 0: No parity error occurred during data reception.	RW	0
3	COL_IF	Transmit Data Collision Error Interrupt Flag. Set by hardware, cleared by software. Writing 1 clears this bit. 0: No interrupt. 1: Interrupt occurred.	RW	0
2	OVL_IF	Receive Data Overflow Error Flag. Set by hardware, cleared by software. Writing 1 clears this bit. 0: No overflow error. 1: Interrupt occurred; receive buffer register was not read and new data was received, overflow error flag is valid.	RW	0
1	RXIF	Data Receive Interrupt Flag. Set to 1 after data is shifted from shift register into receive buffer register. Set by hardware, cleared by software. Writing 1 clears this bit. 0: No interrupt. 1: Interrupt occurred.	RW	0
0	TXIF	Data Transmit Interrupt Flag. Set to 1 after data is shifted from transmit buffer register into shift register. Set by hardware, cleared by software. Writing 1 clears this bit. 0: No interrupt. 1: Interrupt occurred.	RW	0

22.2.5 ISO7816x_DAT

ISO7816 Data Register

Bit	Name	Description	R/W	Reset Value
31:9	--	Reserved	R	0
8	DATA[8]	When parse is in user-defined mode, this is the PARITY bit in the data frame. Write data to be transmitted to this register. Reading this register yields the received data.	RW	0
7:0	DAT	Data Register. Write data to be transmitted to this register. Reading this register yields the received data.	RW	0

22.3 Software Configuration Process and Application Notes

- Set or query the security attributes of ISO7816x through the SCU module; refer to the SCU module for details.
- Enable the corresponding bit in the MOD0_CKE register in the System Control chapter; MOD0_CKE[11] corresponds to ISO78160, MOD0_CKE[12] corresponds to ISO78161.
- Configure GPIO according to the security attributes of ISO7816x, ensuring the module's multiplexed I/O ports comply with SCU security rules.
- Configure the ISO7816x_CTL register to set the enabling functions of ISO7816x.
- Configure the ISO7816x_CLK register to set the clock division for ISO7816x.
- Configure the ISO7816x_BDDIV register to set the F/D coefficient and baud rate coefficient.
- Enable interrupts, set GIC to configure interrupt distribution; refer to the GIC module for details.
- Enable the module by setting ISO7816x_CTL[0], and use ISO7816x for communication.

23 USART

23.1 Introduction

USART (Universal Synchronous Asynchronous Receiver Transmitter) provides a flexible full-duplex data exchange method, suitable for external devices requiring the use of industry-standard NRZ asynchronous serial data format. USART offers more precise baud rates through a fractional baud rate generator. It supports synchronous simplex communication, half-duplex single-wire communication, and modem operations (CTS/RTS). It also supports DMA channels of the DMA controller for high-speed data communication.

23.2 Features

The main features of USART are as follows:

- Full-duplex asynchronous communication.
- Two internal FIFOs for transmitting and receiving data:
  - FIFO width: 9 bits,
  - FIFO depth: 16.
- Universal programmable transmit and receive baud rates:
  - Integer baud rate generation system,
  - Fractional baud rate generation system.
- Programmable data word length (5/6/7/8/9 bits).
- Configurable stop bits in TX and RX (0.5/1/1.5/2 stop bits).
- Parity check control:
  - Transmit parity bit,
  - Check parity of received data bytes.
- Single-wire half-duplex communication.
- Transmitter clock output for synchronous transmission.
- DMA access supported via DMA controller for high-speed data communication.
- Separate enable bits for transmitter and receiver:
  - TX and RX can operate in DMA or CPU mode independently.
- Hardware flow control mode supported.
- Interrupt sources with flags:
  - RX IDLE timeout interrupt supported,
  - RX_FIFO/TX_FIFO watermark interrupt supported.

All functions implemented in the USART peripheral are shown in the following table.

Table 23-1 USART Implementation

USART Features	USART
Modem hardware flow control	Supported
Continuous communication using DMA	Supported
Synchronous mode	Supported
Single-wire half-duplex communication	Supported
Receiver timeout interrupt	Supported
Automatic baud rate detection	Supported

USART data length	5, 6, 7, 8, and 9 bits
Tx/Rx FIFO	Supported
Tx/Rx FIFO bit width	9 bits
Tx/Rx FIFO depth	16

23.3 Block Diagram

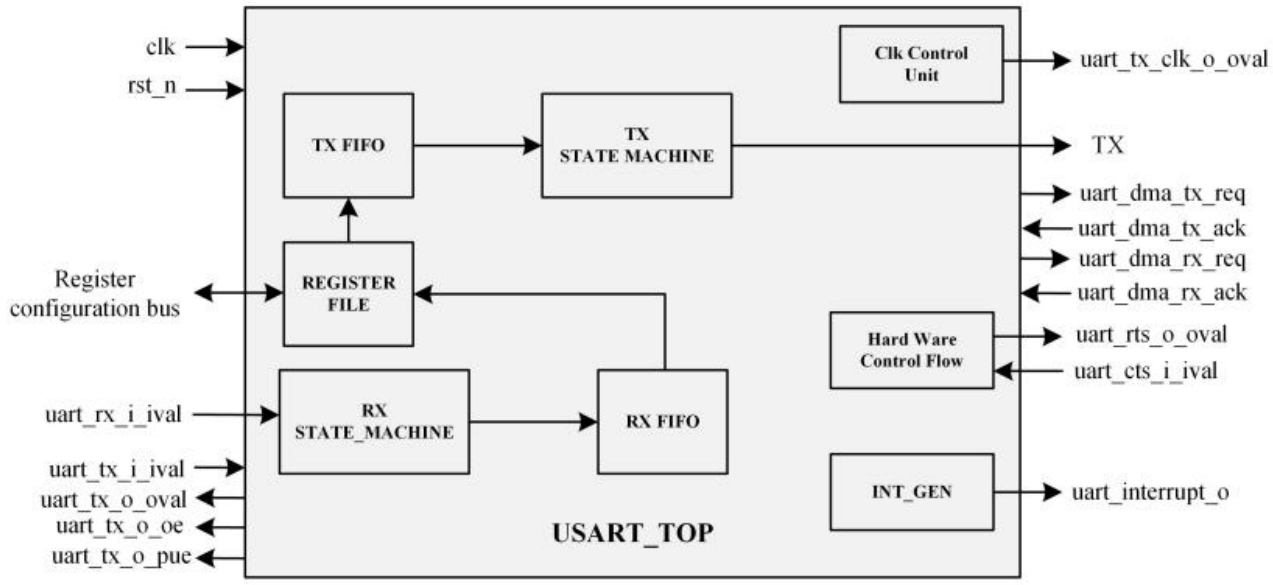


Figure 23-1 USART Block Diagram

23.4 Pin Definitions

The interface connects to another device through three pins. USART asynchronous bidirectional communication requires at least two pins: Receive Data Input (RX) and Transmit Data Output (TX). Synchronous bidirectional communication additionally requires a Clock Output (CK).

RX: Receive Data Input is the serial data input. Data recovery is performed using majority voting sampling technology to distinguish between valid incoming data and noise.

TX: Transmit Data Output. When the transmitter is disabled, this output pin returns to its I/O port configuration value. When the transmitter is enabled and no data needs to be sent, the TX pin is at a high level. In single-wire mode, this I/O is used for both transmitting and receiving data.

Through these pins, serial data frames are transmitted and received in normal USART mode, including:

- An idle line before transmission or reception
- A start bit
- A data word (5/6/7/8/9 bits), with the least significant bit transmitted first
- 0.5, 1, 1.5, or 2 stop bits indicating the end of the frame
- This interface uses a fractional baud rate generator with 13-bit integer and 3-bit fractional parts
- A status register (USARTx_STA)
- Data registers (USARTx_TXDATA/USARTx_RXDATA)
- A baud rate register (USARTx_DIV) - (18-bit integer plus configurable 3-bit fractional part)

The following pins are required in synchronous master/slave mode:

- CK: This pin serves as a clock output in synchronous master mode. In synchronous master mode, this pin outputs the transmitter data clock corresponding to SPI master mode (no clock pulses on start and stop bits, and a software option to configure whether to send a clock pulse on the last data bit). Data can be received synchronously on RX in parallel. This can be used to control peripherals with shift registers (e.g., LCD drivers). Clock phase and polarity are software-programmable.

The following pins are required in hardware flow control mode:

- CTS: Clear To Send, allows transmission. When driven high, this signal prevents data transmission at the end of the current transmission.
- RTS: Request To Send, requests transmission. When low, this signal indicates that the USART is ready to receive data.

23.5 Functional Description

23.5.1 Transmitter

The transmitter can send data words of 5, 6, 7, 8, or 9 bits, depending on the USARTx_SETUP[6:4] register. When the transmit enable bit (USARTx_TXCTRL[0]) is set, the data in the transmit shift register is output through the TX pin, and the corresponding clock pulses are output through the CK pin.

23.5.1.1 Character Transmission

During USART transmission, data is shifted out through the TX pin with the least significant bit first. In this mode, the USARTx_TXDATA register of the TXFIFO buffer acts as FIFO entries, and the transmit state machine shifts these entries out of the FIFO one by one.

Each character frame starts with a start bit, which is a low level corresponding to one bit period. The character frame ends with a configurable number of stop bits.

USART supports the following stop bits: 0.5, 1, 1.5, and 2 stop bits.

23.5.1.2 Configurable Stop Bits

The number of stop bits transmitted per character frame can be programmed in USARTx_TXCTRL[2:1].

- 0.5 stop bits: This is the default value for the number of stop bits
- 1 stop bit
- 1.5 stop bits
- 2 stop bits: Supported by normal USART mode and single-wire mode

Idle frame transmission also includes stop bits.

23.5.2 Receiver

The USART can receive data words of 5, 6, 7, 8, or 9 bits, depending on the USARTx_SETUP[6:4] register.

23.5.2.1 Start Bit Detection

The start bit detection sequence is the same for 16x or 8x oversampling.

In USART, the start bit is detected when specific bit samples are detected. The start bit is detected on a falling edge, followed by the second bit being 0, and at least two of the seventh, eighth, and ninth bits being 0. The

sequence is: 1110X0XXX000XXXXXX.

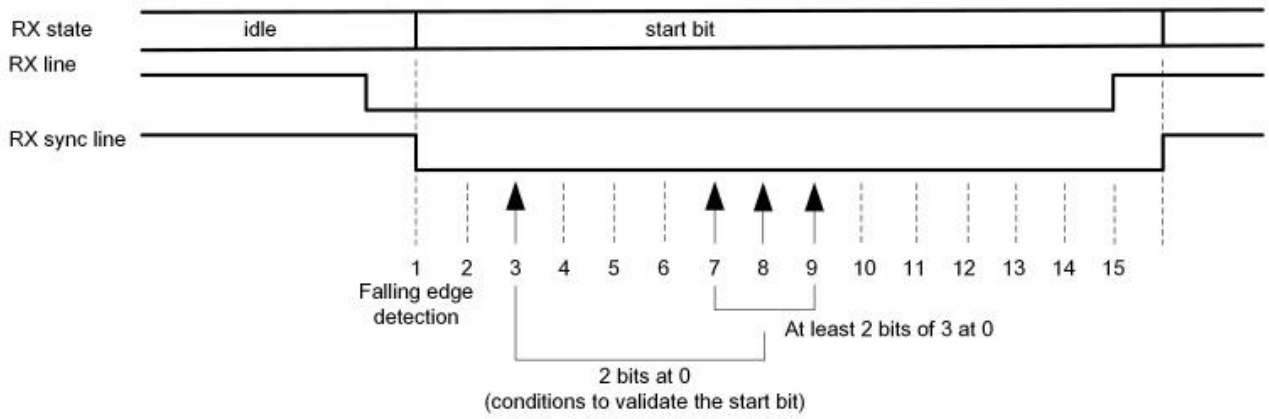


Figure 23-2 Start Bit Detection

If the specific bits are incomplete, start bit detection is aborted, the receiver returns to the idle state (no flag set), and then waits for a falling edge.

Start bit confirmation is as follows:

- First sample: bit 3 is 0.
- Second sample: two of bits 7, 8, and 9 are also 0.。

23.5.2.2 Character Reception

During USART reception, data is shifted into the RX pin with the least significant bit first (default configuration).

In this mode, the RXFIFO buffer receives data as FIFO entries, and the ICB bus can sequentially retrieve these data by accessing the USARTx_RXDATA register.

23.5.2.3 Data Sampling

Data sampling technology performs data recovery by distinguishing between valid incoming data and noise. When the actual baud rate divider is 15, the samples of bits 7, 8, and 9 determine whether the bit frame is 0 or 1. Data sampling is shown in the following figure.

When the actual baud rate divider is even, the sampling position of each data frame (USARTx_SETUP[18:17]) can be set.

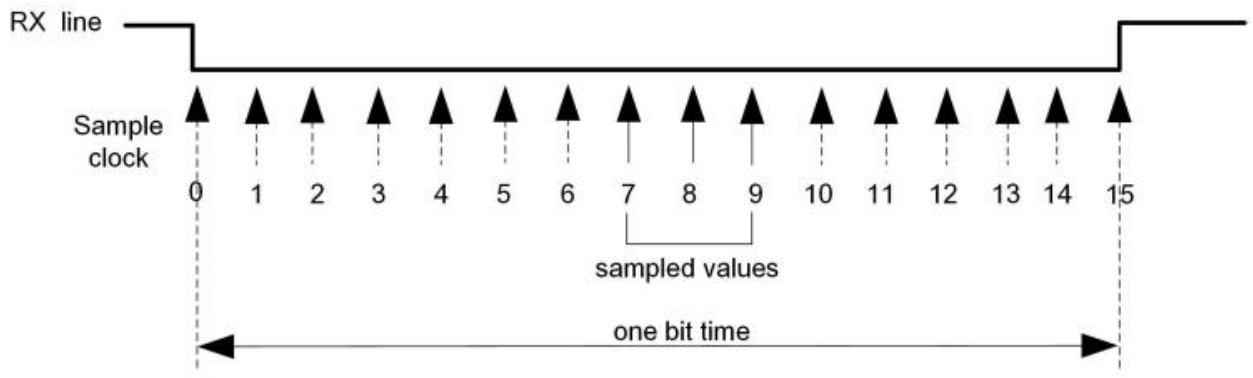


Figure 23-3 Data Sampling

Table 23-2 Noise Detection from Sampled Data

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

The number of stop bits to be received can be configured through the control bits of USARTx_TXCTRL[2:1]. If there is a low level in the middle of the stop bit frame, the RX_FRAME_ERROR (USARTx_STA[8]) flag will be set, which is prohibited in the UART protocol.

23.5.3 Baud Rate Generation

The baud rates of the receiver and transmitter (RX and TX) are both set to the same integer and fractional values in the USARTx_DIV register.

Tx/Rx baud = fclk/(USART_DIV + 1)

fclk: Input clock (icb_clk) to the peripheral.

If the fractional enable bit (USARTx_SETUP[19]) is set, the actual division ratio is USARTx_DIV[20:0], where bits 0~2 are the fractional part and bits 20~3 are the integer part. When the fractional enable bit is disabled, the actual division ratio is USARTx_DIV[18:0], all of which are integer parts.

23.5.4 Automatic Baud Rate Detection

The USART can detect and automatically set the value of the RX_LOW_LEVEL_CNT[31:0] register based on receiving a character frame. Automatic baud rate detection is very useful in the following two cases:

- The system communication speed is unknown in advance.
- The system uses a relatively low-precision clock source. This mechanism allows obtaining the correct baud rate without measuring clock deviation.

No configuration is required to enable it. When the module detects the falling edge of the start bit, it starts counting. It stops counting when the first rising edge of the RX signal is detected. It sets RX_LOW_LEVEL_CNT[31] (RX_HIGH_LEVEL_DETECT) to 1, and stores the count value during the low-level duration in RX_LOW_LEVEL_CNT[19:0] (RX_LOW_LEVEL_LENGTH).

23.5.5 Parity Check Control

Parity check control (generating parity bits in transmission and checking parity in reception) can be enabled by setting the parity enable bit in the USARTx_SETUP register. The frame length is defined according to the length bits and parity selection bits (even or odd parity).

23.5.5.1 Even Parity

The parity bit is calculated such that the 5, 6, 7, 8, or 9 LSB bits (depending on the frame length) in the frame

and the parity bit itself have an even number of "1"s.

23.5.5.2 Odd Parity

The parity bit is calculated such that the 5, 6, 7, 8, or 9 LSB bits (depending on the frame length) in the frame and the parity bit itself have an odd number of "1"s.

23.5.6 USART Synchronous Mode

Synchronous mode is set by writing 1 to the CLKEN bit in the USARTx_SETUP register. USART allows users to control bidirectional synchronous serial communication in master mode. The CK pin is the output of the USART transmitter clock. During start and stop bits, no clock pulses are sent to the CK pin, and there is a software option to configure whether to send a clock pulse on the last data bit. Data can be received synchronously on RX in parallel. This can be used to control peripherals with shift registers (e.g., LCD drivers). Clock phase and polarity are software-programmable. When idle, the external CK clock is not activated. In synchronous mode, the USART transmitter works exactly the same as in asynchronous mode. However, since CK is synchronized with TX (according to CPOL and CPHA), the data on TX is synchronous. In this mode, the USART receiver works differently from asynchronous mode. If RXEN=1, data is sampled on CK (rising or falling edge, depending on CPOL and CPHA).

USART only supports master mode, that is, the CK signal of USART is an output signal, and it does not support configuration as a slave receiver clock signal.

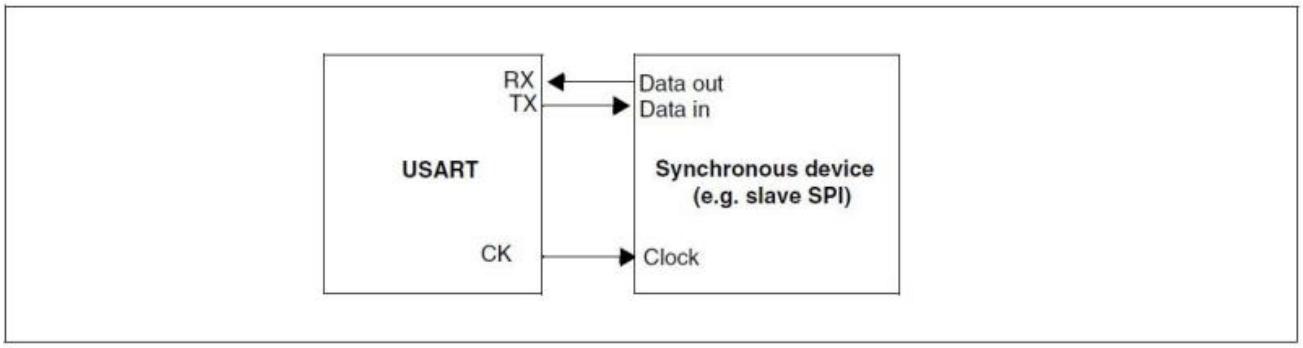


Figure 23-4 USART Synchronous Transmission Example

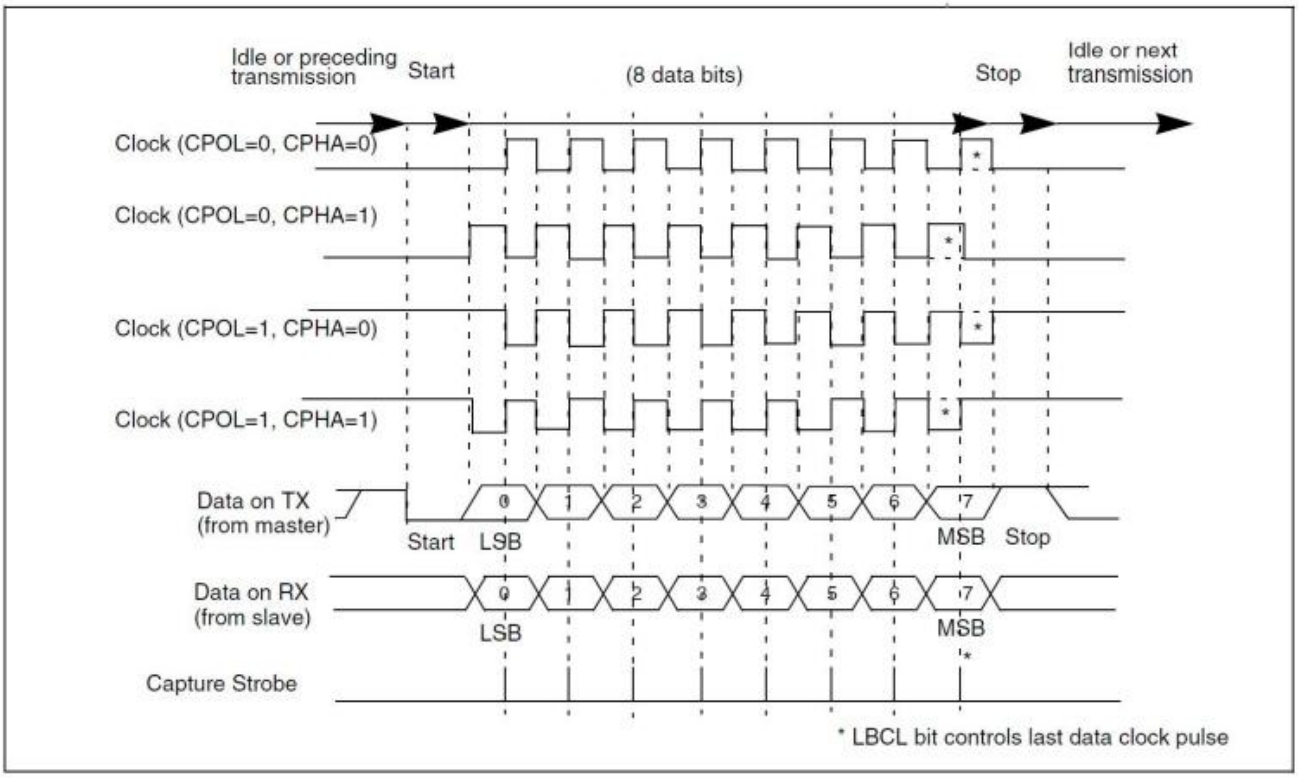


Figure 23-5 USART Data Clock Timing Diagram (8-bit)

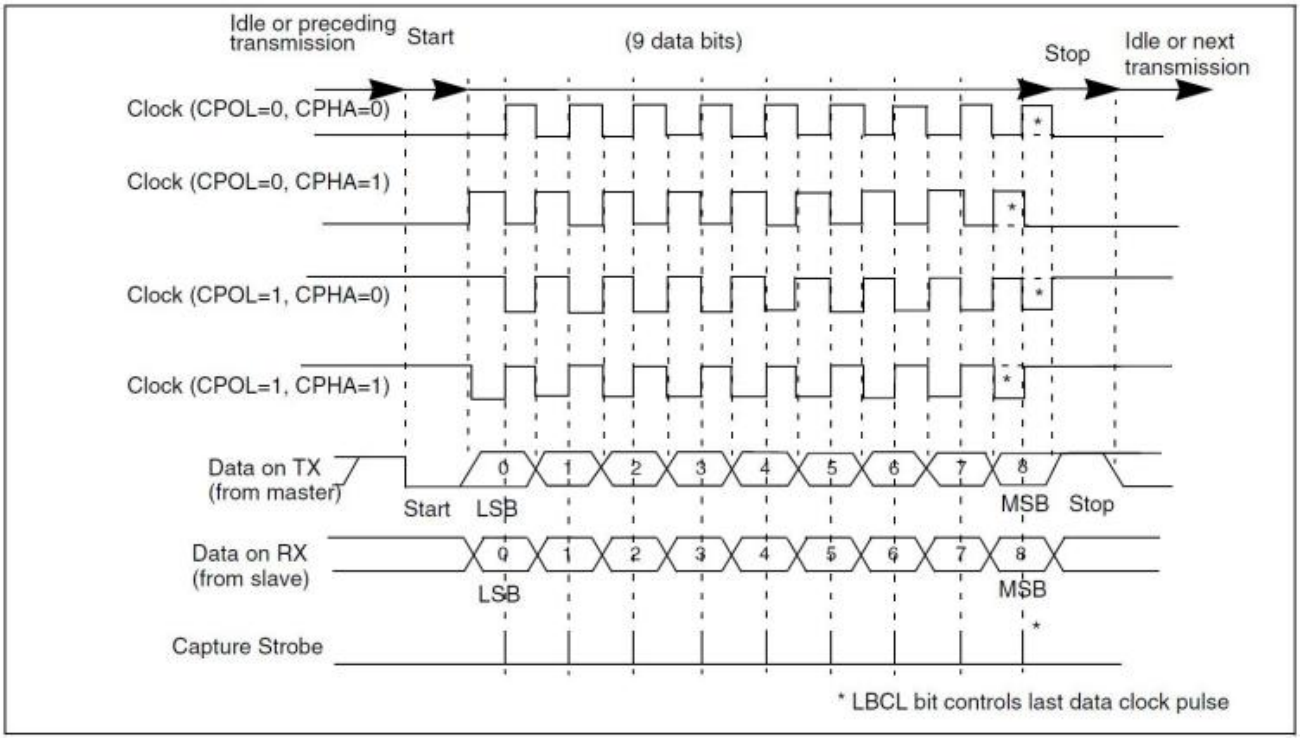


Figure 23-6 USART Data Clock Timing Diagram (9-bit)

23.5.7 Single-Wire Half-Duplex Communication

USART can be configured to follow the single-wire half-duplex protocol. In single-wire half-duplex mode, the TX and RX pins are internally connected. The HDSEL control bit in the USARTx_SETUP register is used to select

half-duplex and full-duplex communication.

Once HDSEL is written to '1':

- The TX and RX lines are internally shorted,
- The RX pin is invalid,
- When there is no data transmission, the TX pin is always in a released state. Therefore, it is used as a standard I/O during idle or reception. This means that the I/O must be configured to set TX as an open-drain multiplex function with an external pull-up resistor.

In addition, the communication protocol is similar to normal USART mode. Any conflicts on the line must be managed by software (e.g., using a centralized arbiter). It is particularly important to note that during communication, the software needs to control the enable of the transmitter and receiver (TXEN, RXEN) to achieve the direction of data. Enable TXEN and disable RXEN when sending data; enable RXEN and disable TXEN when receiving data.

23.5.8 Continuous Communication Using DMA

USART is capable of continuous communication using DMA. DMA requests for the Rx buffer and Tx buffer are generated independently.

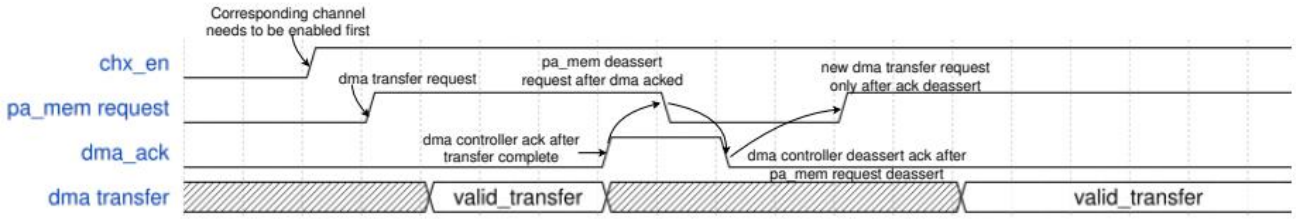


Figure 23-7 USART DMA Transmission Flow Chart

23.5.8.1 Transmission Using DMA

DMA mode for transmission can be enabled by setting the DMA_EN bit in the USARTx_SETUP[15] register. When the FIFO is empty, data is loaded from the SRAM area configured using the DMA peripheral (refer to the corresponding DMA chapter) into the UART_TXFIFO. The DMA-Core accesses the transmitted data through the ICB bus.

23.5.8.2 Reception Using DMA

DMA mode for reception can be enabled by setting the DMA_EN bit in the USARTx_SETUP[15] register. When a byte of data is received, the data is transferred from the UART_RXFIFO to the SRAM area configured using the DMA peripheral (refer to the corresponding DMA chapter). The DMA-Core accesses the received data through the ICB bus.

23.5.8.3 TX and RX Can Operate in CPU Mode or DMA Mode Independently

Transmission and reception can be performed in CPU or DMA independently by setting the SEPARATE EN bit in the USARTx_SETUP[2] register. In this mode, the DMA_EN bit in the USARTx_SETUP[15] register enables transmission, and the RX DMA EN bit in USARTx_SETUP[20] enables reception.

Note: You should refer to the subsystem specifications to understand the availability of the DMA controller. If DMA is not available in the subsystem, you should use the USART explained in the "Transmitter" or "Receiver" sections, i.e., the transmitter or receiver in CPU mode.

23.5.9 Hardware Flow Control

Serial data flow between two devices can be controlled by using CTS input and RTS output. The following figure shows how to connect two USART devices in this mode:

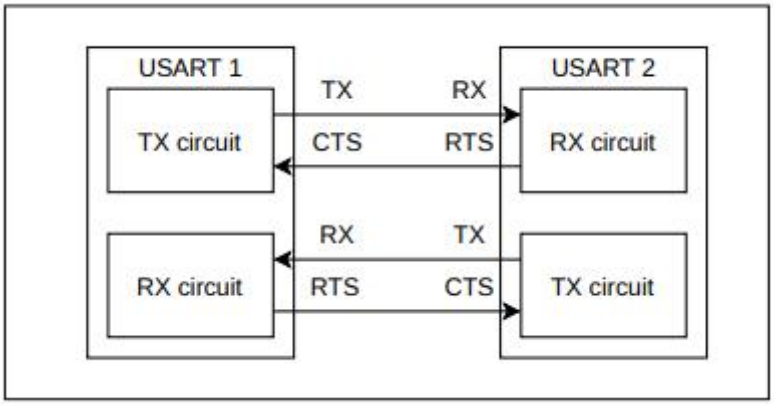


Figure 23-8 Hardware Flow Control Between Two USARTs

RTS and CTS flow control can be enabled separately by writing '1' to the RTSE and CTSE bits in the USARTx_SETUP register.

23.5.9.1 RTS Flow Control

If RTS flow control is enabled (RTSE=1), RTS is active (kept low) as long as the USART receiver is ready to receive new data. When the receive register is full, RTS is inactive, indicating that transmission is expected to stop at the end of the current frame. The following figure shows a communication example with RTS flow control enabled.

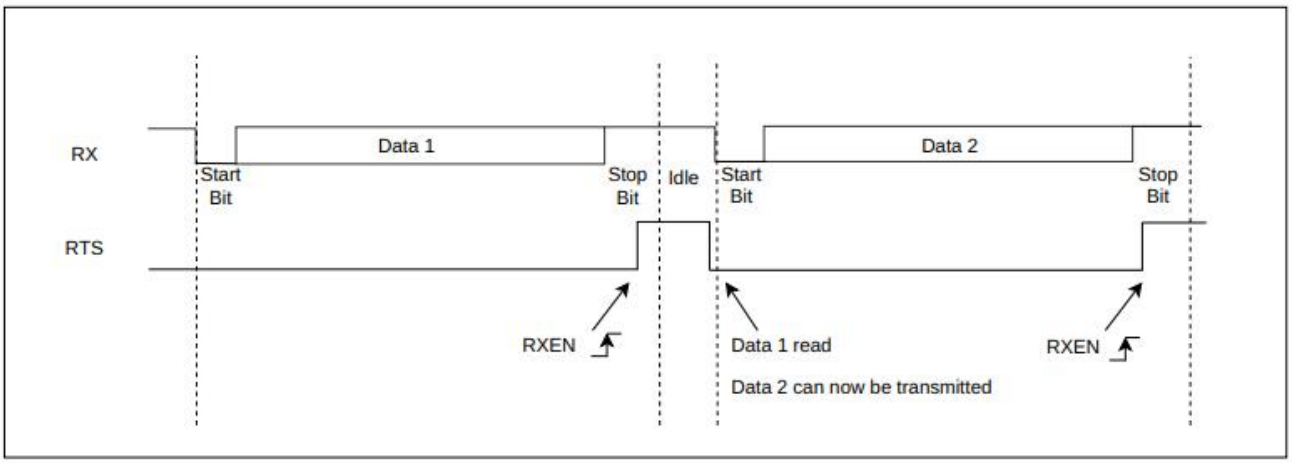


Figure 23-9 RTS Flow Control

23.5.9.2 CTS Flow Control

If CTS flow control is enabled (CTSE=1), the transmitter checks the CTS input before sending the next frame.

If CTS is active (kept low), then the next data is sent (assuming there is data to send, in other words, if TXE=0); otherwise, no transmission is performed. If CTS becomes inactive during transmission, the transmitter stops only after the current transmission is completed.

When CTSE=1, the status bits (CTS_RISE_FLAG and CTS_FALL_FLAG) are automatically set by hardware when the CTS input switches. It indicates when the receiver is ready or not ready for communication. If the interrupt enable bit in the USART_INT_EN register is set to 1, a corresponding interrupt is generated. The following figure shows a communication example with CTS flow control enabled.

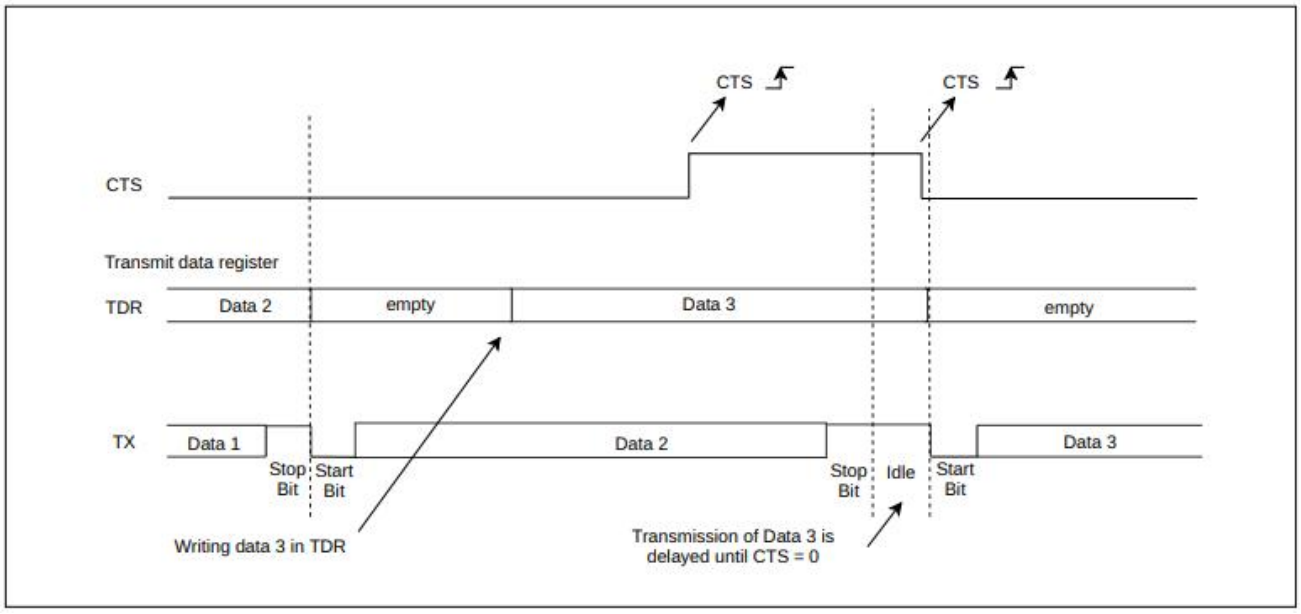


Figure 23-10 CTS Flow Control

23.5.10 Interrupts

The following table shows USART interrupt requests.

Table 23-3 USART Interrupt Requests

Interrupt Event	Event Flag	Enable Control Bit	Description
RX FIFO Watermark	rxip	rxie	Generates an interrupt when the RX FIFO depth exceeds the watermark.
TX FIFO Watermark	txip	txie	Generates an interrupt when the TX FIFO depth is less than the watermark.
Parity Error	rx_error_parity	parity error en	Generates an interrupt when a data parity error occurs in receive mode.
RX FIFO Overflow	rx_error_overflow	overflow error en	Generates an interrupt when the RX FIFO is full and data is still being transmitted.
CTS Rising Edge Flag	Cts rise	ctsie rise en	Generates an interrupt when a rising edge change occurs on the CTS signal.
CTS Falling Edge Flag	Cts fall	ctsie fall en	Generates an interrupt when a falling edge change occurs on the CTS signal.
RX Frame Error	RX frame error	rx frame error en	Generates an interrupt when the RX line is low during the stop bit phase.
RX Stop Bit	RX stop bit error	rx stop bit error en	Generates an interrupt when an error is detected

Error			in the stop bit of the received data.
Rx Idle Timeout	rx idle tout status	rx idle tout irq en	Generates an interrupt when the receive FIFO is not empty and the RX IDLE time exceeds the RX_IDLE_TOUT_NUM register.
Rx Watermark Timeout	rx watermark tout	rx watermark tout irq en	Generates an interrupt when the time of the RX watermark interrupt exceeds the RX_WM_TOUT_NUM register.
TX Transmission Complete	tx eot status	tx eot irq en	Generates an interrupt when data transmission is completed.
RX Transmission Complete	rx eot status	rx eot irq en	Generates an interrupt when data reception is completed.

23.6 Registers

Module Name	Secure Base Address	Non-Secure Base Address
USART0	0x2002_4000	0xA002_4000
USART1	0x2002_8000	0xA002_8000

USARTx Register Offset Addresses

Register Name (x=0,1)	Address Offset	Description
USARTx_TXDATA	0x00	Transmit Data
USARTx_RXDATA	0x04	Receive Data
USARTx_TXCTRL	0x08	Transmit Control
USARTx_RXCTRL	0x0C	Receive Control
USARTx_IE	0x10	Interrupt Control
USARTx_STA	0x14	Interrupt Status
USARTx_DIV	0x18	Baud Rate Divider
USARTx_SETUP	0x1C	USART Configuration and Setup
USARTx_RX_SIZE	0x24	Total Data Size in Receive Transmission Mode
USARTx_TX_SIZE	0x30	Total Data Size in Transmit Transmission Mode
USARTx_RX_IDLE	0x44	RX Line Idle Timeout Setup
USARTx_RX_WM	0x48	RX Watermark Timeout Setup
USARTx_RX_FIFO_LEFT_ENTRY	0x4C	RX FIFO Remaining Entry Count
USARTx_TX_FIFO_LEFT_ENTRY	0x50	TX FIFO Remaining Entry Count
RX_LOW_LEVEL_CNT	0x5C	RX Low Level Time Measurement

23.6.1 USARTx_TXDATA（0x00）

Transmit Data Register

Offset Address = 0x00

Bit	Name	Description	Read/Wr	Reset

Position			ite	Value
31:9	---	Reserved	RO	0x0
8:0	TX_DATA	Transmit Data	RW	0x0

23.6.2 USARTx_RXDATA（0x04）

Receive Data Register

Offset Address = 0x04

Bit Position	Name	Description	Read/Write	Reset Value
31:9	---	Reserved	RO	0x0
8:0	RX_DATA	Receive Data	RO	0x0

23.6.3 USARTx_TXCTRL（0x08）

Transmit Control Register

Offset Address = 0x08

Bit	Name	Description	R/W	Reset Value
31:21	---	Reserved	RO	0x0
20:16	TXCNT	Transmit Watermark Level	RW	0x0
15:5	---	Reserved	RO	0x0
4	CLR	TX Channel Clear and Stop Transmission: - 1'b0: Disable - 1'b1: Enable, stop and clear ongoing transmission	WO	0x0
3	CONTINUOUS	TX Channel Continuous Mode Bit Field: - 1'b0: Disable uDMA reload - 1'b1: Enable uDMA reload and start new transfer at buffer transfer end. At buffer transfer end, uDMA reloads address/buffer size and starts new transfer.	RW	0x0
2:1	CFG_STOP_BIT	Stop Bit Configuration: - 2'b00: 0.5 bit - 2'b01: 1 bit - 2'b10: 1.5 bit - 2'b11: 2 bit	RW	0x0
0	TXEN	Transmit Enable: - 1'b0: Disable - 1'b1: Enable	RW	0x0

23.6.4 USARTx_RXCTRL（0x0C）

Receive Control Register

Offset Address = 0x0C

Bit	Name	Description	R/W	Reset Value
31:21	---	Reserved	RO	0x0

20:16	RXCNT	Receive Watermark Level	RW	0x0
15:3	---	Reserved	RO	0x0
2	CLR	RX Channel Clear and Stop Transmission: - 1'b0: Disable - 1'b1: Enable, stop and clear ongoing transmission	WO	0x0
1	CONTINUOUS	RX Channel Continuous Mode Bit Field: - 1'b0: Disable uDMA reload - 1'b1: Enable uDMA reload and start new transfer at buffer transfer end. At buffer transfer end, uDMA reloads address/buffer size and starts new transfer.	RW	0x0
0	RXEN	Receive Enable: - 1'b0: Disable - 1'b1: Enable	RW	0x0

23.6.5 USARTx_IE（0x10）

Interrupt Control Register

Offset Address = 0x10

Bit	Name	Description	R/W	Reset Value
31:14	---	Reserved	RO	0x0
13	RX_EOT_IE	RX End of Transmission Interrupt Enable: - 1'b0: Disable - 1'b1: Enable	RW	0x0
12	TX_EOT_IE	TX End of Transmission Interrupt Enable: - 1'b0: Disable - 1'b1: Enable	RW	0x0
11	RX_WM_TOUT_IE	RX Watermark Timeout Interrupt Enable: - 1'b0: Disable - 1'b1: Enable	RW	0x0
10	RX_IDLE_TOUT_IE	RX Idle Timeout Interrupt Enable: - 1'b0: Disable - 1'b1: Enable	RW	0x0
9:8	---	Reserved	RO	0x0
7	STOP_BIT_ERR_IE	Stop Bit Error Interrupt Enable: - 1'b0: Disable - 1'b1: Enable	RW	0x0
6	RX_FRAME_ERR_IE	RX Frame Error Interrupt Enable: - 1'b0: Disable - 1'b1: Enable	RW	0x0
5	CTSIE_FALL_IE	CTS Falling Edge Interrupt Enable: - 1'b0: Disable - 1'b1: Enable	RW	0x0
4	CTSIE_RISE_IE	CTS Rising Edge Interrupt Enable:	RW	0x0

		- 1'b0: Disable - 1'b1: Enable		
3	PARITY_ERR_IE	RX Parity Error Interrupt Enable: - 1'b0: Disable - 1'b1: Enable	RW	0x0
2	OWIE	RX Overflow Interrupt Enable: - 1'b0: Disable - 1'b1: Enable	RW	0x0
1	RXIE	Receive Watermark Interrupt Enable: - 1'b0: Disable - 1'b1: Enable	RW	0x0
0	TXIE	Transmit Watermark Interrupt Enable: - 1'b0: Disable - 1'b1: Enable	RW	0x0

23.6.6 USARTx_STA（0x14）

Interrupt Status Register

Offset Address = 0x14

Bit	Name	Description	R/W	Reset Value
31:18	---	Reserved	RO	0x0
17	RX_EOT	RX End of Transmission Status Set when RX_SIZE register value becomes 0 (Receiver decrements RX_SIZE register value by 1 for each frame received)	W1C	0x0
16	TX_EOT	TX End of Transmission Status Set when TX_SIZE register value becomes 0 (Transmitter decrements TX_SIZE register value by 1 for each frame transmitted)	W1C	0x0
15	EMPTY	Receive FIFO Empty	RO	0x1
14	FULL	Transmit FIFO Full	RO	0x0
13	RX_WM_TOUT_STATUS	Set when the trigger time of the RX watermark interrupt exceeds the value in the RX_WM_TOUT_NUM register.	W1C	0x0
12	RX_IDLE_TOUT_STATUS	Set when the Receive FIFO is not empty and no new data is received during the RX_IDLE_TOUT_NUM period.	W1C	0x0
11:10	---	Reserved	RO	0x0
9	STOP_BIT_ERR_FLAG	Stop bit error occurs if the RX line stop bit length does not equal the stop bit configuration bits in USARTx_TXCTRL[2:1] register.	W1C	0x0
8	RX_FRAME_ERR_FLAG	RX frame error status occurs if the RX line is low during the stop bit mode. Write 1 to clear this	W1C	0x0

		status.		
7	CTS_FALL_FLAG	CTS Falling Edge Flag: 0: No falling edge change (from 1 to 0) occurred on CTS status line. 1: A falling edge change (from 1 to 0) occurred on CTS status line.	W1C	0x0
6	CTS_RISE_FLAG	CTS Rising Edge Flag: 0: No rising edge change (from 0 to 1) occurred on CTS status line. 1: A rising edge change (from 0 to 1) occurred on CTS status line.	W1C	0x0
5	RX_ERROR_PARITY	RX Parity Error Status Flag (Write 1 to clear): 0: No error 1: RX parity error occurred	W1C	0x0
4	RX_ERROR_FLOW	RX Overflow Error Status Flag (Write 1 to clear): 0: No error 1: RX overflow error generated	W1C	0x0
3	RX_BUSY	RX Busy Status Flag: 0: No RX transmission in progress 1: RX transmission in progress	RO	0x0
2	TX_BUSY	TX Busy Status Flag: 0: No TX transmission in progress 1: TX transmission in progress	RO	0x0
1	RXIP	Receive Watermark Interrupt, set when the amount of data in the receiver FIFO is greater than the RXCNT value set in the RXCTRL register.	RO	0x0
0	TXIP	Transmit Watermark Interrupt, set when the amount of data in the transmitter FIFO is less than the TXCNT value set in the TXCTRL register.	RO	0x0

23.6.7 USARTx_DIV（0x18）

Baud Rate Divider Register

Offset Address = 0x18

Bit	Name	Description	R/W	Reset Value
31:21	---	Reserved	RO	0x0
20:0	BAUD_DIV	Baud_rate = fusart_clk/(BAUD_DIV+1). If the fractional baud rate generator is disabled, BAUD_DIV[18:0] is the actual divider. If the fractional baud rate generator is enabled, BAUD_DIV[20:3] is the integer part of the actual divider, and BAUD_DIV[2:0] is the fractional part, configurable as follows:	RW	0x0

		- 3'b000: 0 - 3'b001: 0.125 - 3'b010: 0.25 - 3'b011: 0.375 - 3'b100: 0.5 - 3'b101: 0.625 - 3'b110: 0.75 - 3'b111: 0.875		

Note: Whether the fractional baud rate generator is enabled or not, BAUD_DIV[20:0] should not be set to 0 or 1. There are two main reasons for this:

1. When the BAUD_DIV[21:0] register is set to 0, the counter in the USART will underflow.
2. There is noise detection logic for sampled data. Therefore, the RX length should be greater than 2 USART clocks, and the BAUD_DIV[21:0] register should not be set to 1.

23.6.8 USARTx_ SETUP（0x1C）

USART Setup Register

Offset Address = 0x1C

Bit	Name	Description	R/W	Reset Value
31:21	---	Reserved	RO	0x0
20	RX_DMA_EN	Rx Channel DMA Mode Enable. This bit is determined effective by software when the SEPARATE_EN bit is enabled; when the SEPARATE_EN bit is disabled, this bit's effective state is determined by register USARTx_SETUP[15]. - 1'b0: CPU mode - 1'b1: DMA mode	RW	0x0
19	FRACTION_BAUD	Fractional Baud Rate - 1'b0: Disable fractional baud rate generator - 1'b1: Enable fractional baud rate generator	RW	0x0
18:17	SAMPLE_POSITION	When the actual divider value is even, this sampling position can be selected within the data frame: - 2'b00: Scattered: one data frame up, another adjacent frame down - 2'b01: Down - 2'b10: Up	RW	0x2
16	VOTE_MAJORITY_EN	Enable 2/3 majority logic in the data frame: - 1'b0: Disable 2/3 majority logic - 1'b1: Enable 2/3 majority logic	RW	0x1
15	DMA_EN	DMA_EN: Mode Selection - 1'b0: CPU mode, TX data comes from ICB or other bus - 1'b1: DMA mode, TX data comes from DMA. Note: When the SEPARATE_EN register is enabled, this	RW	0x0

		bit is used only for the TX channel.		
14	CTSE	CTS Enable: - 1'b0: CTS hardware flow control disabled - 1'b1: CTS mode enabled, data is transmitted only when the CTS input is activated (set to 0). If the CTS input becomes inactive during data transmission, transmission will stop after the current data transmission is completed. If data is written to the RX FIFO when CTS is inactive, transmission will be postponed until CTS becomes active.	RW	0x0
13	RTSE	RTS Enable: - 1'b0: RTS hardware flow control disabled - 1'b1: RTS mode enabled, data is requested only when the RX FIFO has space, expecting to stop data transmission after the current character transmission is completed. The RTS output is activated (set to 0) when data can be received.	RW	0x0
12	---	Reserved	RO	0x0
11	HDSEL	HDSEL: Half-Duplex Selection - 1'b0: Disable, do not select half-duplex mode - 1'b1: Enable, select half-duplex mode	RW	0x0
10	LBCL	Last Bit Clock Pulse: This bit allows the user to select whether the clock pulse associated with the last data bit (MSB) transmitted needs to be output to the CK pin in synchronous mode. - 1'b0: No output: The clock pulse for the last data bit is not output to the CK pin. - 1'b1: Output: The clock pulse for the last data bit is output to the CK pin.	RW	0x0
9	CPHA	Clock Phase: This bit allows the user to select the clock output phase on the CK pin in synchronous mode. It works together with the CPOL bit to produce the desired clock/data relationship. - 1'b0: The first data is sampled on the first clock edge. - 1'b1: The first data is sampled on the second clock edge.	RW	0x0
8	CPOL	Clock Polarity: This bit allows the user to select the clock output polarity on the CK pin in synchronous mode. It works together with the CPHA bit to produce the desired clock/data relationship. - 1'b0: The CK pin remains low when not transmitting externally. - 1'b1: The CK pin remains high when not transmitting externally.	RW	0x0

7	CLK_EN	Clock Enable: This bit allows the user to enable the CK pin. - 1'b0: CK pin disabled - 1'b1: CK pin enabled tx_clk_o can only be output when TX mode and clk_en=1.	RW	0x0
6:4	BIT_LENGTH	Character Length Bit Field: - 3'b000: 5 bits - 3'b001: 6 bits - 3'b010: 7 bits - 3'b011: 8 bits - 3'b100: 9 bits	RW	0x3
3	CLEAN_FIFO	Clear RX FIFO in all modes, set 1 then set 0 to achieve FIFO reset: - 1'b0: Disable, stop clearing RX FIFO. - 1'b1: Enable, clear RX FIFO.	RW	0x0
2	SEPARATE_EN	Enabling this bit allows TX and RX to work in CPU or DMA mode separately. - 1'b0: TX and RX are both controlled by the DMA_EN bit in the USARTx_SETUP[15] register. - 1'b1: RX is controlled by the RX_DMA_EN bit in the USARTx_SETUP[20] register, TX is controlled by the DMA_EN bit in the USARTx_SETUP[15] register.	RW	0x0
1	PARITY_SEL	Parity Selection: - 1'b0: Even parity - 1'b1: Odd parity	RW	0x0
0	PARITY_EN	Parity Bit Generation and Check Configuration Bit Field: - 1'b0: Disable parity bit generation and check - 1'b1: Enable parity bit check	RW	0x0

23.6.9 USARTx_ RX_SIZE（0x24）

Receive Total Data Size Register

Offset Address = 0x24

Bit	Name	Description	R/W	Reset Value
31:20	---	Reserved	RO	0x0
19:0	RX_SIZE	Total data size in receive transfer mode. The register value decrements by 1 for each frame received by the receiver. When this register value becomes 0, the RX_EOT flag is set to "1". - Read: Returns the remaining size of the receive transfer. - Write: Sets the total data size in receive	RW	0x0

		transfer mode.		

23.6.10 USARTx_ TX_SIZE（0x30）

Transmit Total Data Size Register

Offset Address = 0x30

Bit	Name	Description	R/W	Reset Value
31:20	---	Reserved	RO	0x0
19:0	TX_SIZE	Total data size in transmit transfer mode. The register value decrements by 1 for each frame transmitted by the transmitter. When this register value becomes 0, the TX_EOT flag is set to "1". - Read: Returns the remaining size of the transmit transfer. - Write: Sets the total data size in transmit transfer mode.	RW	0x0

23.6.11 USARTx_ RX_IDLE（0x44）

RX Line Idle Timeout Setup Register

Offset Address = 0x44

Bit	Name	Description	R/W	Reset Value
31:0	IDLE_TOUT_NUM	RX Line Idle Timeout Value	RW	0xffff

23.6.12 USARTx_ RX_WM（0x48）

RX Watermark Timeout Setup Register

Offset Address = 0x48

Bit	Name	Description	R/W	Reset Value
31:0	WM_TOUT_NUM	RX Watermark Timeout Value	RW	0xffff

23.6.13 USARTx_ RX_FIFO_LEFT_ENTRY（0x4C）

RX FIFO Remaining Depth Register

Offset Address = 0x4C

Bit	Name	Description	R/W	Reset Value
31:5	---	Reserved	RO	0x0
4:0	LEFT_ENTRY	RX FIFO Remaining Depth	RO	0x0

23.6.14 USARTx _TX_FIFO_LEFT_ENTRY（0x50）

TX FIFO Remaining Depth Register

Offset Address = 0x50

Bit	Name	Description	R/W	Reset Value
31:5	---	Reserved	RO	0x0
4:0	LEFT_ENTRY	TX FIFO Remaining Depth	RO	0x0

23.6.15 RX_LOW_LEVEL_CNT（0x5C）

RX Low Level Time Measurement Register

Offset Address = 0x5C

Bit	Name	Description	R/W	Reset Value
31	RX_HIGH_LEVEL_DETECT	This bit becomes valid if a rising edge of the RX low level is detected.	W0C	0x0
30:20	---	Reserved	RO	0x0
19:0	RX_LOW_LEVEL_LENGTH	RX Low-Level Duration, counted by icb_clk	RO	0x0

To clear this register, write 0 to the entire register.

23.7 Software Configuration Process and Application Notes

23.7.1 Asynchronous Communication

23.7.1.1 USART Data Reception and Transmission Flow

- Configure the relevant ports in the GPIO register as RX and TX of USARTx (x=0,1);
- Set the character length, stop bit width, parity mode, baud rate, interrupt enable and other configurations;
- Write 0x333F0 to clear the status of the USARTx_STA register;
- Configure USART module interrupt enable and write the interrupt service routine. USART reception, transmission, and error interrupts share the same interrupt entry. It is necessary to determine the type of interrupt based on the interrupt enable bit and status flag.
  - RX Transmission Complete Interrupt: Triggered when the receiver receives data of the size set by the USARTx_RX_SIZE register. After each trigger, the USARTx_RX_SIZE needs to be refilled to set the data size for the next communication to trigger again;
  - TX Transmission Complete Interrupt: Triggered when the transmitter sends data of the size set by the USARTx_TX_SIZE register. After each trigger, the USARTx_TX_SIZE needs to be refilled to set the data size for the next communication to trigger again;
  - RX Watermark Interrupt: Triggered when the data stored in the receive FIFO is greater than the value set by USARTx_RXCTRL[20:16] RXCNT, and the flag is set to 1;
  - TX Watermark Interrupt: Triggered when the remaining data in the transmit FIFO is less than the value set by USARTx_TXCTRL[20:16] TXCNT, and the flag is set to 1;
- Set the TXEN bit of the USARTx_TXCTRL register to 1 to enable the transmitter during the first transmission;
- Set the RXEN bit of the USARTx_RXCTRL register to 1 to enable the receiver before the first reception;
- Process the received or transmitted data and errors.

23.7.1.2 USART Data Reception and Transmission Using DMA

Initialize USART according to steps 1)-6) above, then perform the following configurations.

Transmission Using DMA Channel:

- Set the DMA_EN bit in the USARTx_SETUP[15] register to 1 to enable transmission in DMA mode;
- Write the USARTx_TXDATA register address into the DMA control register and configure it as the target address of the transmission;
- Write the memory address into the DMA control register and configure it as the source address of the transmission;
- Configure the total number of bytes to be transmitted in the DMA control register;
- Configure the channel priority, the number of the multiplexed peripheral, etc., in the DMA register;
- In the DMA interrupt enable register, configure the half-transmission or full-transmission interrupt enable according to application requirements, and write the interrupt service routine;
- Enable the DMA channel in the DMA register;
- Wait for the interrupt event. If transmission is to be performed again, update the memory data, configure the total number of bytes to be transmitted in the DMA register, and send the data.

23.7.1.3 Reception Using DMA Channel:

- Set the DMA_EN bit in the USARTx_SETUP[15] register to 1 to enable reception in DMA mode;
- Write the USARTx_RXDATA register address into the DMA control register and configure it as the source address of the transmission;
- Write the memory address into the DMA control register and configure it as the destination address of the transmission;
- Configure the total number of bytes to be transmitted in the DMA control register;
- Configure the channel priority, the number of the multiplexed peripheral, etc., in the DMA register;
- In the DMA interrupt enable register, configure the half-transmission or full-transmission interrupt enable according to application requirements, and write the interrupt service routine;
- Enable the DMA channel for this purpose in the DMA register;
- Wait for the interrupt event and process the received data and errors.

After configuration, the DMA peripheral loads data from the configured SRAM area into the USARTx_TXDATA register for USART transmission; in addition, when receiving data bytes, the DMA peripheral loads data from the USARTx_RADATA register into the configured SARAM area.

23.7.2 Synchronous Communication

USART supports synchronous master mode communication. An example of the data transmission and reception flow is as follows:

- Configure the relevant ports in the GPIO register as RX, TX, and CK pins of USARTx (x=0,1);
- Set the CLK_EN bit of the USARTx_SETUP register to 1 to enable the CK pin output, and configure the clock polarity CPOL, clock phase CPHA, and last bit clock pulse LBCL according to application requirements;
- Set the character length, stop bit width, parity mode, baud rate, interrupt enable and other configurations;
- Write 0x333F0 to clear the status of the USARTx_STA register;
- Configure USART module interrupt enable and write the interrupt service routine. USART reception, transmission, and error interrupts share the same interrupt entry. It is necessary to determine the type of interrupt based on the interrupt enable bit and status flag.
- Set the TXEN bit of the USARTx_TXCTRL register to 1 to enable the transmitter during the first

transmission;

- Set the RXEN bit of the USARTx_RXCTRL register to 1 to enable the receiver before the first reception;
- Process the received or transmitted data and errors.

It should be noted that in synchronous mode, the data frame format of USART still has start and stop bits, but no clock is sent on the CK signal during start and stop bits. When using USART synchronous mode for communication, the assertion of the RX transmission complete flag (interrupt) requires detecting the stop bit of the received data. This means that when USART communicates with a synchronous peripheral that does not send stop bits (such as SPI) in synchronous mode, its RX transmission complete flag (interrupt) cannot be asserted because the stop bit of the data cannot be detected. At this time, the completion of data reception can be detected through the DMA transmission complete flag or the RX watermark interrupt.

For the flow of USART synchronous master mode transmission and reception using DMA, USART can be initialized according to steps 1)-4) above, and then the configuration of DMA transmission and reception in the "Asynchronous Communication" chapter can be referred to for operation.

23.7.3 Single-Wire Half-Duplex Communication

Set the HDSEL bit of the USARTx_SETUP register to 1 to select single-wire half-duplex mode. In this mode, the RX pin is not used, and the software program needs to manage conflicts in bidirectional communication on the TX line.

During communication, the software needs to perform the following controls:

- For the party sending data, disable the receiver enable RXEN and enable the transmitter TXEN;
- For the party receiving data, disable the transmitter enable TXEN and enable the receiver RXEN.

23.7.4 Closing Communication

During the process of using USART for communication, if you want to close USART communication, you need to close USART according to the following steps:

- Detect the current communication status of USART, and wait for TX_BUSY and RX_BUSY of USARTx_STA[3:2] to be 0 before closing the corresponding TXEN and RXEN in TX_CTRL and RX_CTRL;
- Set TXCTRL[4] and RXCTRL[2] to 1 to clear the TX_FIFO and RX_FIFO;

24 IIC

24.1 Overview

- 2-channel IIC interface module;
- Supports Master and Slave modes;
- 7-bit address bits;
- External pull-up resistors are required for the IIC external bus;
- Supports transmission speeds: Standard mode (100kbps) and Fast mode (400kbps);

24.2 Register Description

Base Addresses of IIC Module

Module Name	Core0 Base Address	Core1 Base Address
IIC0	0x20050000	0xA0050000
IIC1	0x20054000	0xA0054000

Offset Addresses of IIC Module Registers

Register Name	Address Offset	Description
IIC_CTL	0x0	IIC Control Register
IIC_CLK	0x4	IIC Clock Configuration Register
IIC_STAT	0x8	IIC Status Indication Register
IIC_ADDR	0xC	IIC Slave Address Register
IIC_DATA	0x10	IIC Transmit/Receive Data Register

24.2.1 IIC_CTL

IIC Control Register

Offset = 0x0

Bit	Name	Description	R/W Flag	Reset Value
31:6	--	Reserved	R	0x0
5	MODE	Mode Selection Bit 1: Master Mode 0: Slave Mode	RW	0x0
4	ACK	ACK Signal Receive and Transmit Bit When receiving data: 1: Generate ACK signal upon receiving the 9th SCL 0: Do not generate ACK signal upon receiving the 9th SCL	RW	0x0
3	IRQE	IIC Interrupt Enable Bit 0: Disable Interrupt 1: Enable Interrupt	RW	0x0
2:1	BUSCON	Bus Control Generation Bit. Start command is valid when the bus is idle or the master is in transmit state. Stop command is valid when the master is in transmit state.	RW	0x0

		Command bits are cleared when start or stop sequence is detected. 00: No action 01: Generate START sequence 10: Generate STOP sequence 11: Reserved		
0	EN	IIC Module Enable Bit 1: IIC ON 0: IIC OFF	RW	0x0

24.2.2 IIC_CLK

IIC Clock Configuration Register

Offset = 0x4

<table>
<tr><th>Bit</th><th>Name</th><th>Description</th><th>R/W Flag</th><th>Reset Value</th></tr>
<tr><td>31:3</td><td>--</td><td>Reserved</td><td>R</td><td>0x0</td></tr>
<tr><td>2:0</td><td>CLKDIV</td><td>IIC Clock Division Parameter Selection Bit:
IIC communication clock rate calculation formula: SCL = APBCLK / m, where m is generated by CLKDIV, as shown in the table below. Generates IIC high-speed/normal mode communication clock according to different system main frequencies and division parameter selection bits.
<table>
<tr><td rowspan="2">System Main Frequency (MHz)</td><td colspan="2">Division Parameter / CLKDIV (m)</td></tr>
<tr><td>High-Speed Mode</td><td>Normal Mode</td></tr>
<tr><td>1.8432</td><td>000（6）Not Supported</td><td>010（20）</td></tr>
<tr><td>3.6864</td><td>001（10）Not Supported</td><td>011（38）</td></tr>
<tr><td>7.3728</td><td>010（20）</td><td>100（76）</td></tr>
<tr><td>14.7456</td><td>011（38）</td><td>101（152）</td></tr>
<tr><td>29.4912</td><td>100（76）</td><td>110（304）</td></tr>
<tr><td>58.9824</td><td>101（152）</td><td>111（608）</td></tr>
</table></td><td>RW</td><td>0x3</td></tr>
</table>

24.2.3 IIC_STAT

IIC Status Indication Register

Offset = 0x8

Bit	Name	Description	R/W Flag	Reset Value

31:9	--	Reserved	R	0x0
8	DIR	Read/Write Direction Flag Bit 1: Read. 0: Write.	R	0x0
7	MATCH	Address Match Bit, cleared after detecting start or stop sequence. 0: Address Mismatch 1: Address Match	R	0x0
6	BUSY	Communication Status Flag Bit 0: IIC is idle 1: IIC is in normal communication state	R	0x0
5	COL	Transmit Collision Interrupt Flag Bit In Slave mode, when transmitting data, if the transmit data register is not empty and the CPU writes new data to the data register, a transmit collision interrupt is triggered. When receiving data, if the CPU writes data to the data register, a transmit collision interrupt is triggered. Writing 1 to this bit clears it. 0: No transmit collision interrupt triggered 1: Transmit collision interrupt triggered	RW	0x0
4	OVERF	Receive Overflow Interrupt Flag Bit In Slave mode, when receiving data, if a new byte is received before the previous byte has been read, an overflow interrupt is triggered. Writing 1 to this bit clears it. 0: No overflow interrupt triggered 1: Overflow interrupt triggered	RW	0x0
3	TXEMPT	Transmit Data Register Empty Error Flag Bit In Slave mode, when writing data, after the data is written out, if the host sends SCL requesting new data from the slave before new data arrives, a transmit data register empty error interrupt is triggered. Writing 1 to this bit clears it. 0: No transmit data register empty error occurred 1: Transmit data register empty error interrupt triggered	RW	0x0
2	TRANC	Transfer Complete Interrupt Flag Bit Writing 1 to this bit clears it; Triggered when transmit buffer is empty during data transmission or receive buffer is full during data reception. 0: Transfer not complete 1: Transfer complete	RW	0x0
1	RX_NACK	Received NACK Interrupt Flag Bit Writing 1 clears this bit.	RW	0x0

		1: NACK received 0: NACK not received		
0	STPD	STOP Sequence Detection Interrupt Flag Bit Writing 1 to this bit clears it. Automatically cleared when IIC module is disabled or upon receiving START sequence. 0: STOP sequence not detected 1: STOP sequence detected	RW	0x0

24.2.4 IIC_ADDR

IIC Slave Address Register

Offset = 0xC

Bit	Name	Description	R/W Flag	Reset Value
31:8	--	Reserved	R	0x0
7:1	SADR	Device Address Bit. Address cannot be written to this register during address transfer period. In Master mode, represents the slave device address; In Slave mode, this address is used to compare with the address sent by the master.	RW	0x0
0	RW	Master Read/Write Direction Control Bit 0: Write 1: Read	RW	0x0

24.2.5 IIC_DATA

IIC Transmit/Receive Data Register

Offset = 0x10

Bit	Name	Description	R/W Flag	Reset Value
31:8	--	Reserved	R	0x0
7:0	TRDAT	When the master is transmitting data, data can only be written when the transmit buffer is empty. Receive/Transmit data.	RW	0x0

24.3 Software Operation Flow

24.3.1 IIC as Master Communication

- Configure IIC clock enable and GPIO pin multiplexing;
- Configure the IIC_CTL register to select Master mode, and configure whether to enable ACK response and interrupt as needed;
- Configure the IIC_CLK register to set the division factor according to the system operating clock and communication rate requirements, and enable the IIC module;
- Configure the IIC_ADDR register, fill the slave address into IIC_ADDR[7:1], and configure IIC_ADDR[0] according to read/write requirements;
- Configure IIC_CTL[2:1] to 0b'01 to send START timing. At this time, IIC will send the START timing, as

well as the slave address and read/write command;

- Wait for the IIC_STA[2] transmission completion flag to be set, and clear the flag bit (at this time, IIC has completed sending the START timing and address data);
- Master performs data communication
  - Master read: Wait for the IIC_STA[2] flag to be set again, and the master reads the data sent by the slave from IIC_DATA;
  - Master write: The master fills the data to be written to the slave into IIC_DATA, and waits for the IIC_STA[2] flag to be set;
- During the master's reading from the slave, it is necessary to resend information to the slave (the following operations are not required when writing to the slave):
  - Before the last read/write, modify IIC_CTL[4] to not reply with ACK (i.e., reply with NACK) after receiving the 9th bit of data, notifying the slave to stop sending and release SDA;
  - Wait for the IIC_STA[2] transmission completion flag to be set (i.e., the slave completes the last data transmission, releases SDA), and clear the flag bit;
  - The master sends information;
  - If the master needs to switch read/write commands, the master must modify IIC_ADDR[0] according to the read/write requirements, repeat step 5, and resend the START timing and address to the slave;
- End communication:
  - Master writes to slave: The master modifies the IIC_CTL[2:1] register to 0b'10 to send STOP timing, detects that IIC_STA[0] is set, clears the flag bit, and ends communication;
  - Master reads from slave: Before the slave sends the last data, the master modifies IIC_CTL[4] to not reply with ACK after receiving the 9th bit of data, waits for the IIC_STA[2] transmission completion flag to be set, clears the flag bit, the master modifies the IIC_CTL[2:1] register to 0b'10 to send STOP timing, detects that IIC_STA[0] is set, clears the flag bit, and ends communication.

24.3.2 IIC as Slave Communication

- Configure IIC clock enable and GPIO pin multiplexing;
- Configure the IIC_CTL register to select Slave mode, and configure whether to enable ACK response and interrupt as needed;
- Configure the IIC_CLK register to set the division factor according to the system operating clock and communication rate requirements, and enable the IIC module;
- Wait for the master to send the address and read/write command, detect if the address matches, and clear the flag when IIC_STA[7] is set;
- During communication
  - When the master writes to the slave: After detecting that data transmission is completed (IIC_STA[2] is set), clear the flag bit and read the data from IIC_DATA;
  - When the master reads from the slave: After detecting that the address matches (IIC_STA[7] is set), clear the flag bit and write the data to respond to the master into the IIC_DATA register.
- After the master sends the STOP timing, the slave detects that IIC_STA[0] is set, clears the flag bit, and ends communication.

25 SPI

25.1 Introduction

The chip integrates 4 SPI M/S interfaces with dedicated DMA functions, supporting SPI full-duplex mode for communication with external SPI interface devices. It can be programmed to work in Master mode and Slave mode.

25.2 Features

- Supports SPI full-duplex mode, and can be programmed to work in Master mode and Slave mode;
- Supports SPI Clock polarity and phase setting;
- Supports LSB and MSB transmission modes with configurable 8-bit, 16-bit, and 32-bit data widths;
- The master has 256 configurable baud rates (fcpu/(2*(1~256)), up to 29.4912MHz (fcpu=58.984MHz));
- The slave supports configurable low-speed and high-speed modes. The slave in low-speed mode supports up to fcpu/8, and the high-speed mode supports fcpu/4;
- Supports data transmission completion interrupt;
- Supports data transmission collision interrupt;
- Supports SCSN mode error interrupt;
- Supports general DMA function;
- Adds transmit delay function. In DMA master mode, insert 1/2/4/8 SPI_CLK cycles of delay after each 8-bit data transmission;
- Adds sampling delay function, supporting 1/2/3/4 spi_clk clock cycle delay sampling.

25.3 Functional Description

Complies with the standard SPI HOST protocol. The SPI Clock sets different working modes through CPOL (Clock Polarity) and CPHA (Clock Phase) parameters. CPOL determines whether the leading edge of the SPI Clock is a rising edge or a falling edge, and CPHA determines whether the leading edge of the SPI Clock is for data sampling or data setup. The detailed working modes are as follows:

SPI Mode	CPOL/CPHA	Leading Edge	Trailing Edge
0	0/0	Rising edge, data sampling	Falling edge, data setup
1	0/1	Rising edge, data setup	Falling edge, data sampling
2	1/0	Falling edge, data sampling	Rising edge, data setup
3	1/1	Falling edge, data setup	Rising edge, data sampling

Table 25-1 SPI Working Mode Table

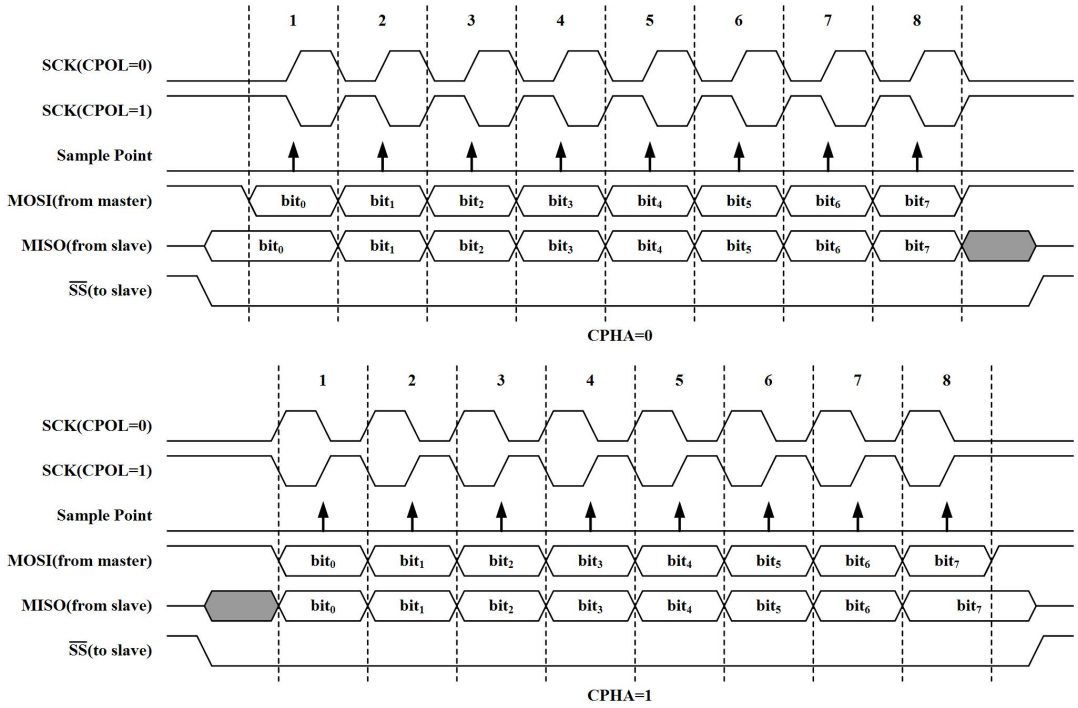


Figure 25-1 Timing Diagrams of Different SPI Working Modes

Data transmission supports two directions: LSB and MSB. The data transmission size supports 8/16/32-bit widths. The SPI clock source is derived from the APB bus clock, and the SPI module clock is generated after a division factor.

Supports four types of interrupts: data transmit collision interrupt, data receive overflow interrupt, transmission end interrupt, and SS mode error interrupt.

Write data collision: When the BUF is full and the bus sends a write data command to SPI, the TXCOLIF bit is set to 1, and SPI will not respond to this write command. If the TXCOL_IRQ_EN interrupt is enabled, an interrupt is generated.

Receive data overflow: If the RXDATA register is not read before the next complete received data enters the shift register, a receive data overflow is generated, and RXCOLIF is set to 1. If COL_IRQ_EN=1, an interrupt is generated, and the newly received data is stored in the receive data register, overwriting the previously unread data.

Transmission end interrupt: When the transmission ends (sck_end), if TR_IRQ_EN=1, an interrupt is generated, and TRIF is set to 1.

SCSN mode error interrupt: In Slave mode, SCSN must be used as an input. If SCSN goes high during data transmission, the SCSN mode error flag is set to 1; In Master mode, only when the Master mode SCSN mode error detection is enabled (SCSN_EN=1) and the SCSN input is low, the SCSN mode error flag is set to 1. Once the SCSN mode error flag is 1, the ongoing transmission is terminated and the SPI module is reset. If ERR_IRQ_EN=1, an interrupt is also generated.

Adopts single transmission completion interrupt + CSN pull-down/pull-up interrupt. The software fills the txbuf in the CSN pull-down interrupt and transmission completion interrupt service routines, and the CSN pull-up interrupt indicates the end of the frame. Realize transmitting one frame of data during one CSN pull-down process.

Supports configuring to transmit the last transmitted data or a register-configurable default value when the TX BUFFER is empty.

For the general DMA function, refer to Chapter 6.

25.4 Register Description

25.4.1 Register List

Base Addresses of SPI Module Registers

Module Name	Core0 Base Address	Core1 Base Address
SPI0	0x2003_8000	0xA003_8000
SPI1	0x2003_C000	0xA003_C000
SPI2	0x2004_0000	0xA004_0000
SPI3	0x2004_4000	0xA004_4000

Offset Addresses of SPI Interface Registers

Register Name	Address Offset	Description
SPI_CTL	0x0	SPI Control Register
SPI_STAT	0x4	SPI Status Indication Register
SPI_TXDATA	0x8	SPI Data Transmit Register
SPI_RXDATA	0xc	SPI Data Receive Register
SPI_TXDFLT	0x10	SPI Default Transmit Data Configuration Register (when transmit BUF is empty)
SPI_DMA_CTL	0x14	SPI DMA Control Register
SPI_DMA_TBADR	0x18	SPI DMA Transmit Start Address Register
SPI_DMA_RBADR	0x1c	SPI DMA Receive Start Address Register
SPI_DMA_TLEN	0x20	SPI DMA Transmit Length Register
SPI_DMA_RLEN	0x24	SPI DMA Receive Length Register
SPI_DMA_TADR	0x28	SPI Current Transmit DMA Address Register
SPI_DMA_RADR	0x2c	SPI Current Receive DMA Address Register
SPI_DMA_IE	0x30	SPI DMA Interrupt Enable Register
SPI_DMA_FLG	0x34	SPI DMA Interrupt Flag Register
SPI_DLYSAMP	0x38	SPI Sampling Delay Register

25.4.2 SPI_CTL（0x00）

SPI Control Register

Bit	Name	Description	R/W	Reset Value
31:30	--	Reserved	RO	0x0
29:28	DLY_CYC_SEL	Data Delay Selection In DMA Master mode, effective after delay enable bit is set. 00: Insert 1 SPI_CLK delay between every 8bit data 01: Insert 2 SPI_CLK delay between every 8bit data 10: Insert 4 SPI_CLK delay between every 8bit data 11: Insert 8 SPI_CLK delay between every 8bit data	RW	0x0
27	DLY_EN	Delay Enable Bit In DMA Master mode, insert delay between every 8bit data. 0: Disable 1: Enable	RW	0x0

26	SLV_TX_ADV	SPI Slave High-Speed Mode Enable Bit =0, SPI Slave Low-Speed Mode, fcpu/8, backward compatible =1, SPI Slave High-Speed Mode, slave rate can reach fcpu/4. Note: In slave high-speed mode, without considering path delay, slave rate can reach fcpu/2.	RW	0x0
25	SCSN_POS_IRQEN	SCSN_POS Interrupt Enable 0, Disable interrupt 1, Enable interrupt	RW	0x0
24	SCSN_NEG_IRQEN	SCSN_NEG Interrupt Enable 0, Disable interrupt 1, Enable interrupt	RW	0x0
23	TXEMPT_IRQEN	TXEMPT Interrupt Enable 0, Disable interrupt 1, Enable interrupt	RW	0x0
22	TX_DFLT_EN	Data content sent when Transmit BUF is empty. 1: Send SPI_TXDFLT register value 0: Send last transmitted SPI_TXDATA value	RW	0x0
21	--	Reserved bit, default is 0, do not write 1	RW	0x0
20	--	Reserved bit, default is 0, do not write 1	RW	0x0
19:12	CLKDIV	SCK Clock Division Factor SCK = APBClock / (2 * (CLKDIV + 1)) Note: APBClock is the system clock	RW	0x0
11:10	WIDTH	Data Width Selection 00: 8bit 01: 16bit 10: 32bit 11: Reserved, 8bit	RW	0x0
9	SCSN_EN	SCSN Mode Error Detection Enable, only applicable to Master mode. 0: Disable Master mode SCSN mode error detection, SCSN is general purpose IO. 1: Enable Master mode SCSN mode error detection, SCSN serves as SPI input. Note: Must keep this bit as 0, do not enable this function.	RW	0x0
8	CPHA	Clock Phase Selection 0: Sample data on front edge 1: Setup data on front edge	RW	0x0
7	CPOL	Clock Polarity Selection 0: "SCK" is set to low level in idle state 1: "SCK" is set to high level in idle state	RW	0x0
6	LMSB	LSB/MSB Selection Method 0: MSB transmitted first 1: LSB transmitted first	RW	0x0

5	TXCOL_IRQ_EN	Transmit Data Collision Interrupt Enable Signal 0: Disable write collision interrupt 1: Enable write collision interrupt Note: Transmit data collision refers to the abnormal situation that occurs when new data is written to the SPI data transmit register before the data in it has been completely sent.	RW	0x0
4	RXCOL_IRQ_EN	Receive Data Collision Interrupt Enable Signal 0: Disable read collision interrupt 1: Enable read collision interrupt Note: Receive data collision refers to the abnormal situation that occurs when new data is received before the data in the SPI data receive register has been read.	RW	0x0
3	ERR_IRQ_EN	SCSN Mode Error Interrupt Enable Signal 0: Disable mode error interrupt 1: Enable mode error interrupt	RW	0x0
2	TR_IRQ_EN	Data Transfer Interrupt Enable Signal 0: Disable data transfer completion interrupt 1: Enable data transfer completion interrupt	RW	0x0
1	MAST/SLAV	Master/Slave Selection 1: MASTER 0: SLAVE	RW	0x1
0	EN	Enable Signal 0: Disable SPI interface 1: Enable SPI interface	RW	0x0

25.4.3 SPI_STA（0x04）

SPI Status Indicator Register

Bit	Name	Description	R/W	Reset Value
31:8	--	Reserved	RO	0x0
7	SCSN_POS	As a slave, CSN pull-up event flag. 0: CSN pull-up event not occurred 1: CSN pull-up event occurred	RWC	0x0
6	SCSN_NEG	As a slave, CSN pull-down event flag. 0: CSN pull-down event not occurred 1: CSN pull-down event occurred	RWC	0x0
5	TXEMPT	Transmit Data Empty Collision Flag Bit. When BUF is empty and SPI bus needs to transmit data, a write data empty collision occurs. 0: No transmit data empty collision interrupt 1: Transmit data empty collision interrupt generated	RWC	0x0
4	TXBUF_VLD	TXBUF Not Empty Flag Bit. 0: TXBUF is empty, software can write new data to TXBUF 1: TXBUF is not empty, software cannot write new data to TXBUF	RO	0x0

3	TXCOLIF	Write Data Collision Flag Bit. When the BUF is full and the bus sends a write data command to SPI, this bit is set to 1, and the current write command is not responded to. If TXCOL_IRQ_EN interrupt is enabled, an interrupt is generated. Writing 1 clears this bit. 0: No write data collision interrupt occurred 1: Write data collision interrupt generated	RWC	0x0
2	RXCOLIF	Receive Data Overflow: If the RXDATA register is not read before the next complete received data enters the shift register, a receive data overflow occurs, then RXCOLIF is set to 1. If COL_IRQ_EN=1, an interrupt is generated, and the newly received data will overwrite the previous data. Writing 1 clears this bit. 0: No receive data overflow interrupt 1: Receive data overflow interrupt generated	RWC	0x0
1	ERRIF	SCSN Mode Error Interrupt Flag Bit. In Slave mode, SCSN must be used as an input. If SCSN goes high during data transmission, the SCSN mode error flag is set to 1; In Master mode, only if Master mode SCSN mode error detection is enabled (SCSN_EN=1) and SCSN input is low, the SCSN mode error flag is set to 1. Once the SCSN mode error flag is 1, it will terminate the ongoing transmission and reset the SPI module. If ERR_IRQ_EN=1, an interrupt will also be generated. Writing 1 clears this bit. 0: No mode error interrupt 1: Mode error interrupt generated	RWC	0x0
0	TRIF	Transfer End Interrupt Flag Bit. When the transfer ends (sck_end), if TR_IRQ_EN=1, an interrupt is generated and TRIF is set to 1. Writing 1 clears this bit. 0: Transfer not complete 1: Transfer complete	RWC	0x0

25.4.4 SPI_TXDATA （0x08）

SPI Data Transmit Register

Bit	Name	Description	R/W	Reset Value
31:0	TXDATA	Data Transmit Register, write valid after enable	RW	0x0

25.4.5 SPI_RXDATA （0x0C）

SPI Data Receive Register

Bit	Name	Description	R/W	Reset Value
31:0	RXDATA	Data Receive Register	RO	0x0

25.4.6 SPI_TXDFLT（0x10）

SPI Default Transmit Data Configuration Register

Bit	Name	Description	R/W	Reset Value
31:0	TXDFLT	Default Transmit Data Register	RW	0x0

25.4.7 SPI_DMA_CTL（0x14）

SPI DMA Control Register

Bit	Name	Description	R/W	Reset Value
31:5	--	Reserved	RO	0x0
4	CS_STOP_RXDMA	When SCSN detects high level, close RX_DMA_EN 0: Disable 1: Enable Note: After this bit is configured as 1, in slave mode, even if configured as DMA cycle mode, after SCSN detects high level, DMA will also close.	RW	0x0
3	RX_CYC_MODE	Receive Cycle Mode Enable	RW	0x0
2	TX_CYC_MODE	Transmit Cycle Mode Enable	RW	0x0
1	RX_DMA_EN	Receive DMA Enable	RW	0x0
0	TX_DMA_EN	Transmit DMA Enable	RW	0x0

25.4.8 SPI_DMA_TBADR（0x18）

SPI DMA Transmit Base Address Register

Bit	Name	Description	R/W	Reset Value
31:18	--	Reserved	RO	0x0
17:0	DMA_TBADR	DMA Transmit Base Address (Byte address) Lowest two bits must be configured according to specific interface data bit width configuration.	RW	0x0

25.4.9 SPI_DMA_RBADR（0x1C）

SPI DMA Receive Base Address Register

Bit	Name	Description	R/W	Reset Value
31:18	--	Reserved	RO	0x0
17:0	DMA_RBADR	DMA Receive Base Address (Byte address) Lowest two bits must be configured according to specific interface data bit width configuration.	RW	0x0

25.4.10 SPI_DMA_TLEN（0x20）

SPI DMA Transmit Length Register

Bit	Name	Description	R/W	Reset Value
31:18	--	Reserved	RO	0x0
17:0	DMA_TLEN	DMA Transmit Length (Byte count) = (n) Byte Lowest two bits must be configured according to specific interface data bit width configuration.	RW	0x0

25.4.11 SPI_DMA_RLEN（0x24）

SPI DMA Receive Length Register

Bit	Name	Description	R/W	Reset Value
31:18	--	Reserved	RO	0x0
17:0	DMA_RLEN	DMA Receive Length (Byte count) = (n) Byte Lowest two bits must be configured according to specific interface data bit width configuration.	RW	0x0

25.4.12 SPI_DMA_TADR（0x28）

SPI Current Transmit DMA Address Register

Bit	Name	Description	R/W	Reset Value
31:18	--	Reserved	RO	0x0
17:0	DMA_TADR	Current Transmit DMA Address (Byte address)	RO	0x0

25.4.13 SPI_DMA_RADR（0x2C）

SPI Current Receive DMA Address Register

Bit	Name	Description	R/W	Reset Value
31:18	--	Reserved	RO	0x0
17:0	DMA_RADR	Current Receive DMA Address (Byte address)	RO	0x0

25.4.14 SPI_DMA_IE（0x30）

SPI DMA Interrupt Enable Register

Bit	Name	Description	R/W	Reset Value
31:6	--	Reserved	RO	0x0
5	RX_ERR_IE	Receive Data Overwrite Interrupt Enable =0, Not enable =1, Enable	RW	0x0
4	TX_ERR_IE	Transmit Data Error Interrupt Enable =0, Not enable	RW	0x0

		=1, Enable		
3	RX_FIE	DMA Receive Full Interrupt Enable =0, Not enable =1, Enable	RW	0x0
2	RX_HIE	DMA Receive Half Full Interrupt Enable =0, Not enable =1, Enable	RW	0x0
1	TX_FIE	DMA Transmit Full Interrupt Enable =0, Not enable =1, Enable	RW	0x0
0	TX_HIE	DMA Transmit Half Full Interrupt Enable =0, Not enable =1, Enable	RW	0x0

25.4.15 SPI_DMA_FLG（0x34）

SPI DMA Interrupt Flag Register

it	Name	Description	R/W	Reset Value
31:6	--	Reserved	RO	0x0
5	RX_ERR	Receive Data Overwrite Flag, write "1" clears this bit	RWC	0x0
4	--	Reserved	RO	0x0
3	RX_FDONE	DMA Receive Full Interrupt Flag, write "1" clears this bit	RWC	0x0
2	RX_HDONE	DMA Receive Half Full Interrupt Flag, write "1" clears this bit	RWC	0x0
1	TX_FDONE	DMA Transmit Full Interrupt Flag, write "1" clears this bit	RWC	0x0
0	TX_HDONE	DMA Transmit Half Full Interrupt Flag, write "1" clears this bit	RWC	0x0

25.4.16 SPI_DLYSAMP（0x38）

SPI Sampling Delay Register

Bit	Name	Description	R/W	Reset Value
31:3	--	Reserved	RO	0x0
2:1	DLYSMP_SEL	Master mode, SPI receive data sampling delay selection 0x0: Delay 1 sys_clk cycle sampling 0x1: Delay 2 sys_clk cycle sampling 0x2: Delay 3 sys_clk cycle sampling 0x3: Delay 4 sys_clk cycle sampling	RW	0x0
0	DLYSMP_EN	SPI Receive Data Sampling Delay Enable 0: Disable 1: Enable	RW	0x0

26 QSPI

26.1 Introduction

QSPI (Quad Serial Peripheral Interface) is an interface dedicated to communicating with external storage devices. It supports communication with single-wire, dual-wire, and quad-wire SPI-Flash and SPI-SRAM, and can operate in the following three operation modes:

- Register mode: Transmit and receive by configuring the corresponding QSPI registers;
- DMA mode: Based on register mode, combine with DMA data stream to simplify the transmission process and improve transmission efficiency;
- Memory-mapped mode (XIP): The external Flash is mapped to the device address space and regarded by the system as equivalent to internal storage space.

26.2 Features

- The main features of QSPI are as follows:
- Supports Master/Slave modes;
- Supports four interface modes: Single-wire half-duplex/single-wire/dual-wire/quad-wire modes;
- Supports three operation modes: Register/DMA/XIP modes;
- Supports fully programmable command format;
- Supports SDR and DDR;
- Programmable clock frequency, polarity, and phase, with a maximum frequency of FQSPI_CLK/2;
- Programmable MSB and LSB transmission;
- Integrated 4*32 bits transmit and receive FIFO;
- Supports 8/16/32-bit data access;
- Supports interrupts such as operation completion, overflow error, and FIFO threshold.

26.3 Signal Line Definition

QSPI connects to external SPI Flash devices through 6 signal lines. The pin information is shown in the "QSPI Pin Description" table.

Table 26-1 QSPI Pin Description

Pin Name	Direction	Description
SCK	Output/Input	Clock signal
CS#	Output/Input	Chip select signal, active low
SO (IO0)	Input/Output	Single-wire half-duplex mode: Data input or output Single-wire mode: Data output Dual-wire mode: Data input or output Quad-wire mode: Data input or output
SI (IO1)	Input/Output	Single-wire half-duplex mode: No SI (IO1) Single-wire mode: Data input Dual-wire mode: Data input or output Quad-wire mode: Data input or output
WP# (IO2)	Input/Output	Single-wire half-duplex mode: Flash write protection function

		Single-wire mode: Flash write protection function Dual-wire mode: Flash write protection function Quad-wire mode: Data input or output
HOLD# (IO3)	Input/Output	Single-wire half-duplex mode: Flash hold function Single-wire mode: Flash hold function Dual-wire mode: Flash hold function Quad-wire mode: Data input or output

26.4 Interface Timing

QSPI communicates with Flash through commands:

- Each command includes 5 phases: Instruction, Address, Mode, Dummy Cycle, and Data phases;
- Any phase can be configured to be skipped, but at least one of the 5 phases must exist in a command;
- Any phase can be configured into single-wire half-duplex/single-wire/dual-wire/quad-wire mode;
- nCS is pulled low before the start of each command and pulled high after the end of each command.

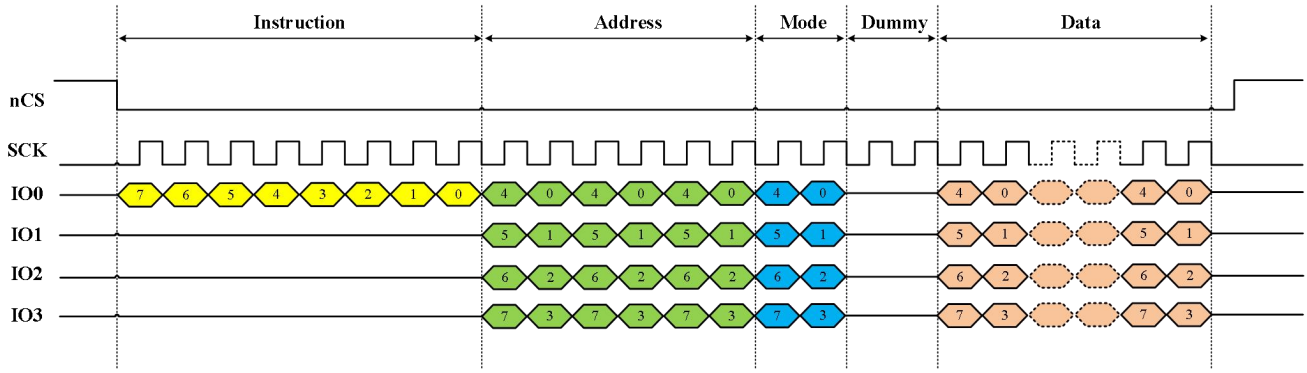


Figure 26-1 Read Timing in SDR Quad-Wire Mode

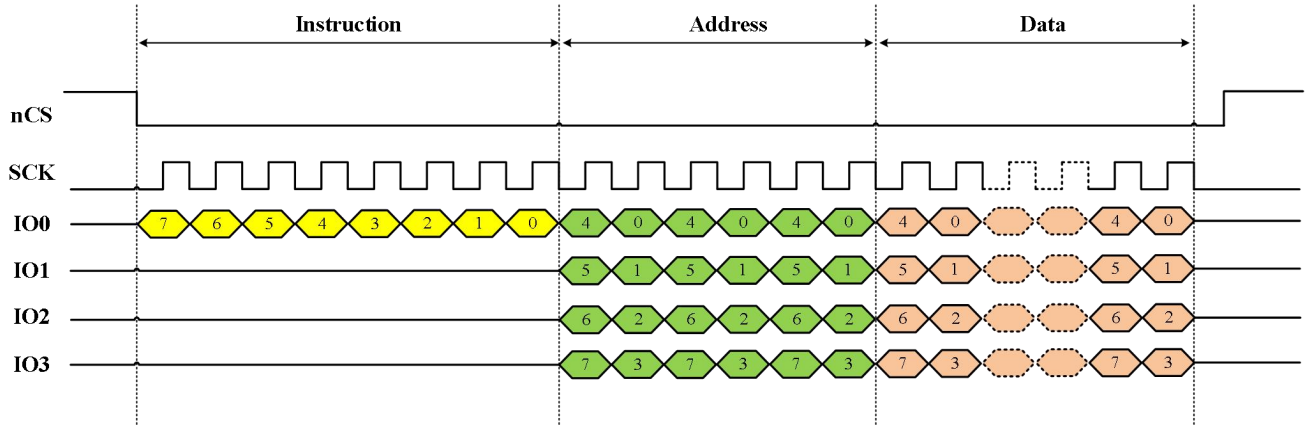


Figure 26-2 Write Timing in SDR Quad-Wire Mode

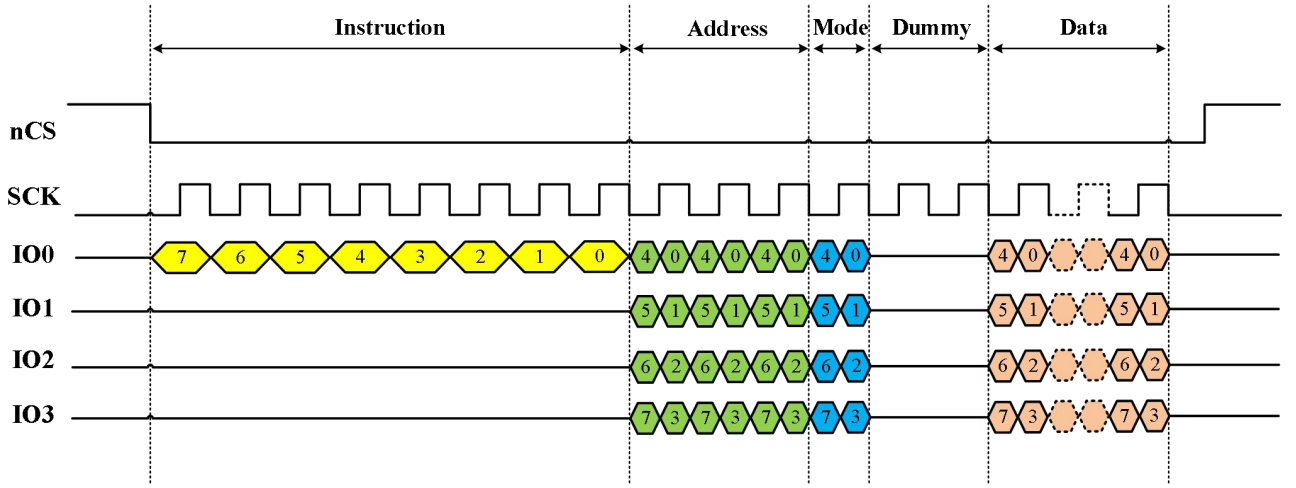


Figure 26-3 Read Timing in DDR Quad-Wire Mode

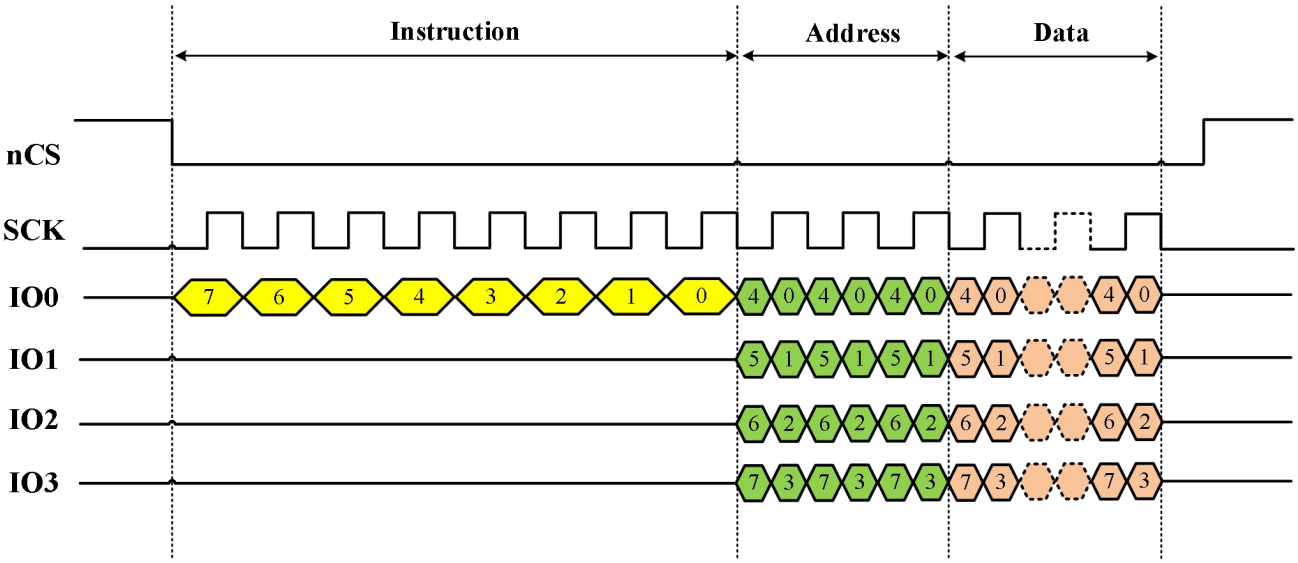


Figure 26-4 Write Timing in DDR Quad-Wire Mode

The QSPI clock can be combined into four timings by configuring SCKMODE.CPOL (clock phase) and SCKMODE.CPHA (clock polarity) to adapt to the slave:

- CPOL = 0, CPHA = 0: SCK is low in idle state, and received data is sampled on the rising edge of SCK;
- CPOL = 0, CPHA = 1: SCK is low in idle state, and received data is sampled on the falling edge of SCK;
- CPOL = 1, CPHA = 0: SCK is high in idle state, and received data is sampled on the falling edge of SCK;
- CPOL = 1, CPHA = 1: SCK is high in idle state, and received data is sampled on the rising edge of SCK.

Note: For SPI Flash, the configurations of CPOL = 0, CPHA = 0 (mode 0) and CPOL = 1, CPHA = 1 (mode 3) are commonly used.

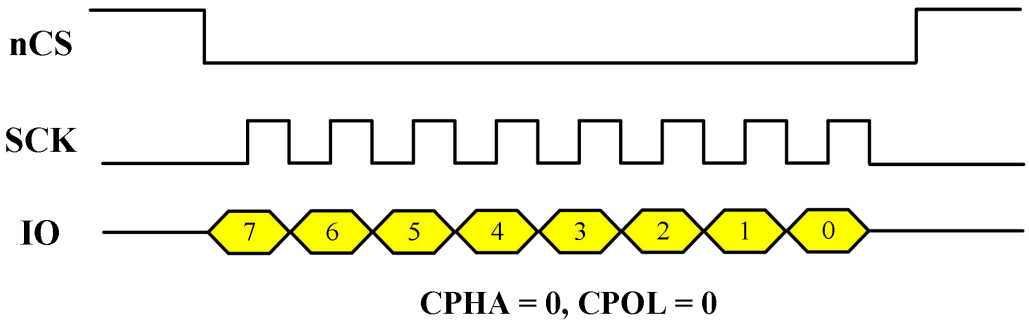


Figure 26-5 Mode 0 Timing Diagram

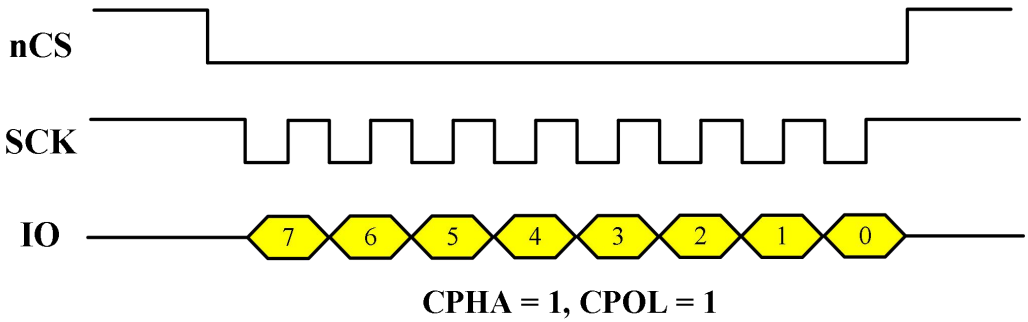


Figure 26-6 Mode 3 Timing Diagram

26.5 Functional Description

26.5.1 Command Sequence

A command sequence includes 5 phases: Instruction, Address, Mode, Dummy Cycle, and Data phases. The following describes the five phases in XIP mode.

26.5.1.1 Instruction Phase

In the instruction phase, an 8-bit instruction is sent to the Flash to specify the type of operation to be performed:

- The read instruction is configured in the FFMT.RCMD_CODE register;
- The write instruction is configured in the FFMT1.WCMD_CODE register;
- The single-wire/dual-wire/quad-wire mode of the instruction phase is configured in the FFMT[9:8].CMD_PROTO register;
- When FFMT.CMD_EN = 0, the instruction phase is skipped, and the command sequence starts from the address phase.

Note: Most SPI Flashes only support single-wire mode when receiving instructions, i.e., transmitting 1 bit at a time through the SO (IO0) pin.

26.5.1.2 Address Phase

In the address phase, 1-4 bytes of address are sent to the Flash to indicate the operation address:

The address length is configured in the FFMT.ADDR_LEN register;

The single-wire/dual-wire/quad-wire mode of the address phase can be configured in the FFMT.ADDR_PROTO register;

Directly access the address space in the SPI Flash through the XIP bus;

When FFMT.ADDR_LEN = 0, the address phase is skipped, and the command sequence directly enters the

next phase.

26.5.1.3 Mode Phase

In the mode phase, 1-4 bytes of mode bytes are sent to the Flash, usually used for Flash mode configuration:

- The length of mode bytes can be configured in the FFMT1.MODE_CNT register;
- The bytes to be sent can be configured in the FFMT1.MODE_CODE register;
- The single-wire/dual-wire/quad-wire mode of the mode phase is configured in the FFMT1.MODE_PROTO register;
- When FFMT1.MODE_CNT = 0, the mode phase is skipped, and the command sequence directly proceeds to the next phase.

26.5.1.4 Dummy Cycle Phase

In the dummy cycle phase, no data is transmitted or received, allowing the Flash to have sufficient time to prepare data under high-frequency clocks:

- For read operations, the dummy cycle length (1~31) is configured in the FFMT.RPAD_CNT and FFMT1.RPAD_CNT_H registers;
- For write operations, the dummy cycle length (1~31) is configured in the FFMT1.WPAD_CNT register;
- When the corresponding dummy cycle length is 0, the dummy cycle phase is skipped, and the command sequence directly enters the data phase.

26.5.1.5 Data Phase

The data phase supports 8/16/32-bit data access and burst transmission:

- The single-wire/dual-wire/quad-wire mode of the data phase is configured in the FFMT.DATA_PROTO register;
- Enable the corresponding burst transmission function in the FCTRL register;
- Read/write data is directly transmitted through the XIP bus.

26.5.2 Interface Modes

26.5.2.1 Single-Wire Half-Duplex Mode

Single-wire half-duplex mode only allows single-bit serial data transmission/reception. This mode can be entered by configuring the SPI_FMT.PROTO (non-XIP mode) or DATA_PROTO (XIP mode) register to 3'b111. In this mode, during any phase of the command:

- IO0 is in inout mode;
- IO1 is in output mode and forced to 1;
- IO2 is in output mode and forced to 0 (disable "write protection" function);
- IO3 is in output mode and forced to 1 (disable "hold" function).

26.5.2.2 Single-Wire Mode

Single-wire mode only allows single-bit serial data transmission and reception. Different phases can be configured to CMD_PROTO/ADDR_PROTO/MODE_PROTO/DATA_PROTO as 00 to enter this mode. In this mode, during the command phase:

- IO0 is in output mode;
- IO1 is in input mode (high impedance);
- IO2 is in output mode and forced to 0 (disable "write protection" function);
- IO3 is in output mode and forced to 1 (disable "hold" function).

26.5.2.3 Dual-Wire Mode

Dual-wire mode allows 2 bits of data to be transmitted/received simultaneously. Different phases can be configured to CMD_PROTO/ADDR_PROTO/MODE_PROTO/DATA_PROTO as 01 to enter this mode. In this mode, during the command phase:

- IO0/IO1 are in high impedance (input mode) only during the data phase of read operations, and in output mode in other cases;
- IO2 is in output mode and forced to 0 (disable "write protection" function);
- IO3 is in output mode and forced to 1 (disable "hold" function).

26.5.2.4 Quad-Wire Mode

Quad-wire mode allows 4 bits of data to be transmitted/received simultaneously. Different phases can be configured to CMD_PROTO/ADDR_PROTO/MODE_PROTO/DATA_PROTO as 10 to enter this mode. In this mode, during the command phase:

- IO2/IO3 are multiplexed as data transmission IOs;
- IO0/IO1/IO2/IO3 are in high impedance (input mode) only during the data phase of read operations, and in output mode in other cases.

26.5.3 SDR/DDR Modes

26.5.3.1 SDR Mode

In SDR mode, data on IO0/IO1/IO2/IO3 is driven only on the falling edge of SCK. When receiving data, sampling is performed on the rising edge of SCK (default), and the sampling point can be adjusted by configuring the SDR_SCKSAMPLE register.

- In register and DMA modes, enter SDR mode by configuring the SPI_CR.DDR_EN register to 0 (default).
- In XIP mode, enter SDR mode for the data/mode/addr/cmd phases respectively by configuring the FFMT1.DDR_EN[3:0] register to 0 (default).

26.5.3.2 DDR Mode

In DDR mode, data on IO0/IO1/IO2/IO3 is driven on both the falling and rising edges of SCK, and sampling is performed on the next edge (rising and falling edges). The sampling point can be adjusted by configuring the SCKSAMPLE register.

- In register and DMA modes, enter DDR mode by configuring the SPI_CR.DDR_EN register to 1.
- In XIP mode, enter DDR mode for the data/mode/addr/cmd phases respectively by configuring the FFMT1.DDR_EN[3:0] register to 1.

26.5.4 Operation Modes

26.5.4.1 Register Mode

In register mode, data transmission and reception are performed by configuring registers:

- Configure the data direction through FMT.SPI_DIR;
- Start a transmission by writing a value to the QSPI_TXDATA register through software to write into the transmit FIFO:
  - During transmission, this data is the transmitted data;
  - During reception, this data is invalid data, and the received data will be stored in the receive FIFO.
- When the transmitted/received data length reaches SPI_TSIZE/SPI_RSIZE, the transmission/reception completion interrupt flag is set to 1, and the interrupt is raised if the interrupt is enabled.

26.5.4.2 DMA Mode

In DMA mode, data transmission and reception are performed by configuring the following registers:

- Configure the data direction through SPI_FMT.SPI_DIR;
- Perform transmission by enabling the transmit DMA to transfer data to the transmit FIFO:
  - During transmission, the data in the transmit FIFO is the transmitted data, and a transmit DMA request is raised if the transmit FIFO is not full;
  - During reception, the data in the transmit FIFO is invalid data, and the received data will be stored in the receive FIFO. If the receive DMA is enabled, a receive DMA request is raised if there is data in the receive FIFO.
- The transmit and receive DMAs can enter continuous transmission mode respectively by configuring the CR.DMA_TXCT_MODE and CR.DMA_RXCT_MODE registers to 1;
- The transmission/reception completion interrupt flag is set to 1, and the interrupt is raised if the interrupt is enabled, in the following three cases:
  - The transmitted/received data length reaches SPI_TSIZE/SPI_RSIZE;
  - During continuous transmission, disable the transmit/receive DMA enable, and the data in the transmit FIFO is transmitted;
  - During continuous transmission, disable the continuous transmit/receive enable, and the data in the transmit FIFO is transmitted.

Note: For continuous transmission mode, parameter configuration synchronization with uDMA is required for normal transmission.

26.5.4.3 XIP Mode

In XIP mode, the Flash can be accessed through the XIP bus, and the external Flash is treated as internal storage access:

- Enter XIP mode by configuring the QSPI_FCTRL.FLASH_EN register to 1;
- Stop transmission when the maximum access address reaches the address boundary configured by QSPI_BOUNDARY_CFG;
- The start address of the SPI Flash can be configured through the SPI_ADDR_WRAP register;
- To perform write, write mask, burst transmission, and loopback transmission operations on the SPI Flash in XIP mode, the corresponding bits of the QSPI_FCTRL register need to be configured to enable these functions;

- Supports 8/16/32-bit access.

26.5.5 CS Mode Description

In register mode and DMA mode, software can control the CS pin level through three registers: CSID, CSDEF, and CSMODE. Among them, select the slave through the CSID register; configure the default idle level value of the CS output through CSDEF; configure the CS mode through CSMODE. There are three CS modes:

- Automatic mode:
  - Enter automatic mode when the software configures the CSMODE register to 00;
  - CS is pulled low before the start of each frame of data and pulled high after the end of each frame of data;
- Hold mode:
  - Enter hold mode when the software configures the CSMODE register to 10;
  - In hold mode, CS remains low after being pulled low until the software performs the following operations to pull CS high:
    - ✓ Configure the CSMODE register to exit hold mode;
    - ✓ Configure to enter XIP mode;
    - ✓ Disable DMA-related enables.
- Disable mode:
  - Enter disable mode when the software configures the CSMODE register to 11;
  - In disable mode, CS maintains the idle level value.

Note: XIP mode is not affected by the CSMODE register.

26.6 Registers

Base Addresses of QSPI Module Registers

Module Name	Physical Address	Mapped Address
QSPI0	0xB008 8000	0xB008 8000
QSPI1	0xB008 C000	0xB008 C000

Offset Addresses of QSPI Registers

Register Name	Address Offset	Description
QSPI_SCKDIV	0x00	QSPI Clock Division Register
QSPI_SCKMODE	0x04	QSPI Clock Mode Register (Master/Slave)
QSPI_SCKSAMPLE	0x08	QSPI Sampling Point Configuration Register (DDR Mode)
QSPI_FORCE	0x0C	QSPI Pin Output Enable Register
QSPI_CSID	0x10	QSPI Chip Select ID Register
QSPI_CSDEF	0x14	QSPI Chip Select Default Value Register
QSPI_CSMODE	0x18	QSPI Chip Select Mode Register
QSPI_VERSION	0x1C	QSPI Version Register
QSPI_ADDR_WRAP	0x20	SPI Flash Start Address Offset Register
QSPI_BOUNDARY_CFG	0x24	SPI Flash Address Boundary Configuration Register
QSPI_DELAY0	0x28	QSPI Delay Control Register 0
QSPI_DELAY1	0x2C	QSPI Delay Control Register 1

QSPI_FIFO_NUM	0x30	QSPI FIFO Valid Data Count Register
QSPI_TSIZE	0x34	QSPI Transmit Data Length Register
QSPI_RSIZE	0x38	QSPI Receive Data Length Register
QSPI_FMT	0x40	QSPI Frame Format Register (Master/Slave)
QSPI_TXDATA	0x48	QSPI Transmit Data Register (Master/Slave)
QSPI_RXDATA	0x4C	QSPI Receive Data Register (Master/Slave)
QSPI_TX_MARK	0x50	QSPI Transmit FIFO Threshold Register (Master/Slave)
QSPI_RX_MARK	0x54	QSPI Receive FIFO Threshold Register (Master/Slave)
QSPI_FCTRL	0x60	SPI Flash Interface Control Register (XIP)
QSPI_FFMT	0x64	SPI Flash Command Format Register (XIP)
QSPI_IE	0x70	QSPI Interrupt Enable Register (Master/Slave)
QSPI_IP	0x74	QSPI Interrupt Pending Register (Master/Slave)
QSPI_FFMT1	0x78	SPI Flash Command Format Register 1 (XIP)
QSPI_STATUS	0x7C	QSPI Status Register
QSPI_SDR_SCKSAMPLE	0x80	QSPI Sampling Point Configuration Register (SDR Mode)
QSPI_CR	0x84	QSPI Control Register

26.6.1 QSPI_SCKDIV（0x00）

QSPI Clock Division Register

Offset Address = 0x00

Bit	Name	Description	R/W	Reset Value
31:12	--	Reserved	RO	0x0
11:0	SCKDIV	Clock Division Factor Selection Bit Calculation Formula FSCK = FQSPI_CLK / (2 * (SCKDIV + 1)) 0x000: FQSPI_CLK /2 0x001: FQSPI_CLK /4 0x002: FQSPI_CLK /6 0x003: FQSPI_CLK /8 ..... Note: DDR mode does not support SCKDIV=0	RW	0x4

26.6.2 QSPI_SCKMODE（0x04）

QSPI Clock Mode Register (Master/Slave)

Offset Address = 0x04

Bit	Name	Description	R/W	Reset Value
31:2	--	Reserved	RO	0x0
1	CPOL	QSPI Clock SCK Polarity Selection Bit 0: SCK maintains low level when idle 1: SCK maintains high level when idle	RW	0x0
0	CPHA	QSPI Clock SCK Phase Selection Bit 0: Receive data on the first edge after SCK starts 1: Receive data on the second edge after SCK starts Note: In DDR mode, only CPHA configuration of 0 is supported	RW	0x0

26.6.3 SPI_SCKSAMPLE（0x08）

QSPI Sampling Point Configuration Register (DDR Mode)

Offset Address = 0x08

Bit	Name	Description	R/W	Reset Value
31:2	--	Reserved	RO	0x0
11:0	SCKSMP	DDR Mode, SCK Sampling Point Selection Bit 0x000: No delay 0x001: Delay 1 bus clock cycle for sampling 0x002: Delay 2 bus clock cycles for sampling ..... Note: Sampling delay must be less than 1 QSPI_SCK cycle	RW	0x0

26.6.4 QSPI_FORCE（0x0C）

QSPI Pin Output Enable Register

Offset Address = 0x0C

Bit	Name	Description	R/W	Reset Value
31:2	--	Reserved	RO	0x0
1	FORCE_WP	WP (IO2) Port Output State Selection Bit 0: Output low level 1: Output high level Note: This bit is invalid in quad-wire mode	RW	0x0
0	FORCE_EN	Force output enable when pins serve as non-data pins in different interface modes: 0: Invalid 1: Force enable	RW	0x1

26.6.5 QSPI_CSID（0x10）

QSPI Chip Select ID Register

Offset Address = 0x10

Bit	Name	Description	R/W	Reset Value
31:3	--	Reserved	RO	0x0
2:0	CSID	Chip Select Device ID Selection Bit 0x0: Disable 0x1: Select CS0 Others: Reserved	RW	0x1

26.6.6 QSPI_CSDEF（0x14）

QSPI Chip Select Default Value Register

Offset Address = 0x14

Bit	Name	Description	R/W	Reset Value
31:1	--	Reserved	RO	0x0
0	CSDEF	CS0 Default (Idle) State Selection Bit (Master Mode) 0: Low level 1: High level	RW	0x1

26.6.7 QSPI_CSMODE（0x18）

QSPI Chip Select Mode Register

Offset Address = 0x18

Bit	Name	Description	R/W	Reset Value
31:2	--	Reserved	RO	0x0
1:0	CSMODE	Chip Select Mode Selection Bit 00: Automatic mode 01: Reserved 10: Hold mode 11: Disable mode Note: It is recommended to use hold mode for continuous data transmission, and configure to disable mode after data transmission is completed	RW	0x0

26.6.8 QSPI_VERSION（0x1C）

QSPI Version Register

Offset Address = 0x1C

Bit	Name	Description	R/W	Reset Value
31:0	VERSION	QSPI Version 2.1.0	RO	0x00020100

26.6.9 QSPI_ADDR_WRAP（0x20）

SPI Flash Start Address Offset Register

Offset Address = 0x20

Bit	Name	Description	R/W	Reset Value
31:0	OFFSET	SPI Flash Start Address Offset	RW	0x0

26.6.10 QSPI_BOUNDARY_CFG（0x24）

SPI Flash Address Boundary Configuration Register

Offset Address = 0x24

Bit	Name	Description	R/W	Reset Value
31:0	BOUNDARY	XIP Boundary Address Configuration Register, configured as SPI FLASH address boundary minus 1, enable bit is QSPI_FCTRL.BOUNDRY_EN register. Application scenario: The address auto-increment of some SPI FLASH depends on its boundary. In XIP Burst mode, when the burst length is long, software needs to configure this boundary through this register. The hardware will resend the address when reaching the boundary, preventing data overflow, i.e., XIP splits the long burst into multiple short bursts for transmission.	RW	0x3ff

26.6.11 QSPI_DELAY0（0x28）

QSPI Delay Control Register 0

Offset Address = 0x28

Bit	Name	Description	R/W	Reset Value
31:24	--	Reserved	RO	0x0
23:16	SCKCS	Delay cycle count selection bit from the last valid edge of SCK to the rising edge of CS 0x00: No delay 0x01: Delay 1 bus clock cycle 0x02: Delay 2 bus clock cycles	RW	0x1
15:8	--	Reserved	RO	0x0
7:0	CSSCK	Delay cycle count selection bit from CS falling edge to the first valid rising edge of SCK 0x00: No delay 0x01: Delay 1 bus clock cycle 0x02: Delay 2 bus clock cycles	RW	0x1

26.6.12 QSPI_DELAY1（0x2C）

QSPI Delay Control Register 1

Offset Address = 0x2C

Bit	Name	Description	R/W	Reset Value
31:24	--	Reserved	RO	0x0
23:16	INTERXFR	Delay cycle count selection bit between two frames of data 0x00: No delay 0x01: Delay 1 bus clock cycle 0x02: Delay 2 bus clock cycles ..... Note: This register is only valid in Register and DMA operation modes	RW	0x0
15:8	--	Reserved	RO	0x0
7:0	INTERCS	Minimum delay cycle count selection bit between CS rising edge and CS falling edge 0x00: No delay 0x01: At least delay 1 bus clock cycle 0x02: At least delay 2 bus clock cycles	RW	0x3

26.6.13 QSPI_FIFO_NUM（0x30）

QSPI FIFO Valid Data Count Register

Offset Address = 0x30

Bit	Name	Description	R/W	Reset Value
31:19	--	Reserved	RO	0x0
18:16	RX_NUM	RX FIFO Valid Data Count 0: 0 count 1: 1 count 2: 2 counts	RO	0x0
15:3	--	Reserved	RO	0x0
2:0	TX_NUM	TX FIFO Valid Data Count 0: 0 count 1: 1 count 2: 2 counts	RO	0x0

26.6.14 QSPI_TSIZE（0x34）

QSPI Transmit Data Length Register

Offset Address = 0x34

Bit	Name	Description	R/W	Reset

				Value
31:0	TSIZE	QSPI Transmit Data Length Selection Bit 0x0000 0000: 0 beat 0x0000 0001: 1 beat 0x0000 0002: 2 beats	RW	0xffff ffff

26.6.15 QSPI_RSIZE（0x38）

QSPI Receive Data Length Register

Offset address =0x38

Bit	Name	Description	R/W	Reset Value
31:0	RSIZE	QSPI Receive Data Length Selection Bit 0x0000 0000: 0 beat 0x0000 0001: 1 beat 0x0000 0002: 2 beats	RW	0xffff ffff

26.6.16 QSPI_FMT（0x40）

QSPI Frame Format Register (Master/Slave, Non-XIP Mode)

Offset Address = 0x40

Bit	Name	Description	R/W	Reset Value
31:22	--	Reserved	RO	0x0
21:16	LEN	QSPI Per Frame Transmission Data Bit Width Selection Bit 6'b001000: 8 bits (default) 6'b010000: 16 bits 6'b100000: 32 bits Note: Master mode supports 4~32 bits, Slave mode supports 8/16/32 bits	RW	0x8
15:6	--	Reserved	RO	0x0
5	ENDIAN_BYTE	QSPI Byte Transmission Order Selection Bit 0: Lower byte transmitted first 1: Higher byte transmitted first Note: This bit only changes the order of the 4 bytes within 32 bits	RW	0x0
4	PROTO_HI	QSPI_FMT.PROTO[1:0] High Bit Extension Bit	RW	0x0
3	DIR	QSPI Pin Data Transmission Direction Selection Bit 0: Receive 1: Transmit Note: This definition only applies to data transmission pins with input/output in different wire modes, and is invalid for pins with fixed transmission directions, such as IO0 and IO1 in	RW	0x0

		single-wire mode which can only serve as output and input pins respectively		
2	ENDIAN	QSPI Transmission Data Order Selection Bit 0: MSB, higher bit transmitted first 1: LSB, lower bit transmitted first Note: This bit does not change byte order, only changes the transmission order of 8 bits within 1 byte of data	RW	0x0
1:0	PROTO	QSPI Wire Mode Selection Bit 000: Single-wire mode 001: Dual-wire mode 010: Quad-wire mode 011: Eight-wire mode (Reserved) 100: X16 mode (Reserved) 111: Single-wire half-duplex mode Note: Hardware does not support eight-wire and X16 modes, do not configure in software	RW	0x0

26.6.17 QSPI_TXDATA（0x48）

QSPI Transmit Data Register (Master/Slave)

Offset Address = 0x48

Bit	Name	Description	R/W	Reset Value
31:0	TXDATA	Transmit Data Buffer	WO	0x0

26.6.18 QSPI_RXDATA（0x4C）

QSPI Receive Data Register (Master/Slave)

Offset Address = 0x4C

Bit	Name	Description	R/W	Reset Value
31:0	RXDATA	Receive Data Buffer	RO	0x0

Note: Valid data bit width depends on QSPI_FMT.LEN register configuration.

26.6.19 QSPI_TX_MARK（0x50）

QSPI Transmit FIFO Threshold Register (Master/Slave)

Offset Address = 0x50

Bit	Name	Description	R/W	Reset Value
31:3	--	Reserved	RO	0x0
2:0	TX_MARK	Transmit FIFO Threshold Selection Bit. Interrupt flag tx_irq is raised when transmit FIFO valid data count does not reach the threshold 0x0: Invalid 0x1: 1 valid data	RW	0x0

		0x2: 2 valid data 0x3: 3 valid data 0x4: 4 valid data		

26.6.20 QSPI_RX_MARK（0x54）

QSPI Receive FIFO Threshold Register (Master/Slave)

Offset Address = 0x54

Bit	Name	Description	R/W	Reset Value
31:3	--	Reserved	RO	0x0
2:0	RX_MARK	Receive FIFO Threshold Selection Bit. Interrupt flag rx_irq is raised when receive FIFO valid data count exceeds the threshold 0x0: 0 valid data 0x1: 1 valid data 0x2: 2 valid data 0x3: 3 valid data 0x4: Invalid	RW	0x0

26.6.21 QSPI_FCTRL（0x60，XIP）

SPI Flash Interface Control Register (XIP)

Offset Address = 0x60

Bit	Name	Description	R/W	Reset Value
31:24	--	Reserved	RO	0x0
5	BOUNDRY_EN	XIP Mode Boundary Address Enable Bit 0: Disable 1: Enable	RW	0x0
4	FLASH_WRAP_EN	XIP Mode WRAP Function Enable Bit 0: Disable (default) 1: Enable Note: The slave needs to support and enable the WRAP function, otherwise it can only be configured to 0	RW	0x0
3	FLASH_BURST_EN	XIP Mode Burst Transmission Enable Bit 0: Disable (default) 1: Enable	RW	0x0
2	FLASH_WEN	XIP Mode Write Enable Bit 0: Disable (default) 1: Enable	RW	0x0
1	FLASH_WMASK_EN	XIP Mode Write Data Mask Function Enable Bit 0: Disable (default) 1: Enable	RW	0x0
0	FLASH_EN	XIP Mode Enable Bit 0: Disable	RW	0x1

		1: Enable (default)		

26.6.22 QSPI_FFMT（0x64，XIP）

SPI Flash Command Format Register (XIP)

Offset Address = 0x64

Bit	Name	Description	R/W	Reset Value
31:24	PAD_CODE	First 8 bits data transmitted during dummy cycle	RW	0x0
23:16	RCMD_CODE	SPI Flash Read Instruction Byte	RW	0x03
15	DATA_PROTO_HI	QSPI_FFMT.DATA_PROTO High Bit Extension Bit	RW	0x0
14	ENDIAN_F	XIP Mode, QSPI Transmission Data Order Selection Bit 0: MSB, higher bit transmitted first 1: LSB, lower bit transmitted first Note: This bit does not change byte order, only changes the transmission order of 8 bits within 1 byte of data	RW	0x0
13:12	DATA_PROTO	QSPI Data Phase Interface Mode Selection Bit 000: Single-wire mode 001: Dual-wire mode 010: Quad-wire mode 011: Eight-wire mode (Reserved) 100: X16 mode (Reserved) 111: Single-wire half-duplex mode Note: Hardware does not support eight-wire and X16 modes, do not configure in software	RW	0x0
11:10	ADDR_PROTO	QSPI Address Phase Interface Mode Selection Bit 00: Single-wire mode 01: Dual-wire mode 10: Quad-wire mode 11: Eight-wire mode (Reserved) Note: Hardware does not support eight-wire mode, do not configure in software	RW	0x0
9:8	CMD_PROTO	QSPI Instruction Phase Interface Mode Selection Bit 00: Single-wire mode 01: Dual-wire mode 10: Quad-wire mode 11: Eight-wire mode (Reserved) Note: Hardware does not support eight-wire mode, do not configure in software	RW	0x0
7:4	RPAD_CNT	Read Instruction, Dummy Cycle Count Selection Bit 5'b0 0000: No dummy cycle (default) 5'b0 0001: 1 dummy cycle	RW	0x0

		5'b 1 1111: 31 dummy cycles Note: The 5th bit of RPAD_CNT is configured in FFMT1.RPAD_CNT_H		
3:1	ADDR_LEN	Address Length Selection Bit 000: Skip address phase 001: 8 bits 010: 16 bits 011: 24 bits (default) 100: 32 bits	RW	0x3
0	CMD_EN	Instruction Send Selection Bit 0: Do not send instruction 1: Send instruction (default)	RW	0x1

26.6.23 QSPI_IE（0x70）

QSPI Interrupt Enable Register (Master/Slave)

Offset Address = 0x70

Bit	Name	Description	R/W	Reset Value
31:15	--	Reserved	RO	0x0
14	CFG_ERR_EN	QSPI Configuration Frame Error Interrupt Enable Bit 0: Disable 1: Enable Note: This interrupt includes normal QSPI mode and XIP mode	RW	0x0
13	--	Reserved	RO	0x0
12	RX_DONE_EN	Receive Completion Interrupt Enable Bit 0: Disable 1: Enable	RW	0x0
11	TX_DONE_EN	Transmit Completion Interrupt Enable Bit 0: Disable 1: Enable	RW	0x0
10:8	--	Reserved	RO	0x0
7	DONE_EN	Transmission Completion Interrupt Enable Bit 0: Disable 1: Enable	RW	0x0
6	--	Reserved	RO	0x0
5	TX_OVR_EN	Transmit FIFO Overflow Interrupt Enable Bit 0: Disable 1: Enable	RW	0x0
4	RX_UDR_EN	Receive FIFO Underflow Interrupt Enable Bit 0: Disable 1: Enable	RW	0x0
3	RX_OVR_EN	Receive FIFO Overflow Interrupt Enable Bit 0: Disable	RW	0x0

		1: Enable		
2	TX_UDR_EN	Transmit FIFO Underflow Interrupt Enable Bit 0: Disable 1: Enable	RW	0x0
1	RX_IRQ_EN	Receive FIFO Threshold Interrupt Enable Bit 0: Disable 1: Enable	RW	0x0
0	TX_IRQ_EN	Transmit FIFO Threshold Interrupt Enable Bit 0: Disable 1: Enable	RW	0x0

26.6.24 QSPI_IP（0x74）

QSPI Interrupt Pending Register (Master/Slave)

Offset Address = 0x74

Bit	Name	Description	R/W	Reset Value
31:2	--	Reserved	RO	0x0
1	RX_IRQ	Receive FIFO Valid Data Count Exceeds Receive Threshold Interrupt Flag Bit 0: Not exceeded 1: Exceeded	RO	0x0
0	TX_IRQ	Transmit FIFO Valid Data Count Does Not Reach Transmit Threshold Interrupt Flag Bit 0: Reached 1: Not reached	RO	0x0

26.6.25 QSPI_FFMT1（0x78 ， XIP）

SPI Flash Command Format Register 1 (XIP)

Offset Address = 0x78

Bit	Name	Description	R/W	Reset Value
31:28	MODE_CNT	Mode Byte Length Selection Bit 0x00: Skip mode phase (default) 0x08: 8 bits, recommended value Others: Not recommended	RW	0x0
27:20	MODE_CODE	SPI Flash Mode Byte	RW	0x0
19:18	MODE_PROTO	QSPI Mode Byte Phase Interface Mode Selection Bit 00: Single-wire mode 01: Dual-wire mode 10: Quad-wire mode 11: Eight-wire mode (Reserved) Note: Hardware does not support eight-wire mode, do not configure in software	RW	0x0

17:14	DDR_EN	Instruction [14], Address [15], Mode [16] and Data [17] Phase DDR Enable Bits 0: SDR mode (default) 1: DDR mode	RW	0x0
13	RPAD_CNT_H	Read Instruction, Dummy Cycle Count Selection Bit 5th bit, lower 4 bits configured in FFMT.RPAD_CNT register	RW	0x0
12:8	WPAD_CNT	Write Instruction, Dummy Cycle Count Selection Bit 5'b0 0000: No dummy cycle (default) 5'b0 0001: 1 dummy cycle ..... 5'b 1 1111: 31 dummy cycles	RW	0x0
7:0	WCMD_CODE	SPI Flash Write Instruction Byte	RW	0x02

26.6.26 QSPI_STATUS（0x7C）

QSPI Status Register

Offset Address = 0x7C

Bit	Name	Description	R/W	Reset Value
31:21	--	Reserved	RO	0x0
20	CFG_ERR	Frame Configuration Error Flag Bit. Flag is raised when frame configuration error occurs in normal QSPI or XIP mode, cleared by software writing 1 0: Not completed 1: Completed	RW	0x0
19	--	Reserved	RO	0x0
18	RX_DONE	Receive Completion Flag Bit. Flag is raised when received data reaches receive length, cleared by software writing 1 0: Not completed 1: Completed Note: Detailed description see "Operation Modes" chapter	RW	0x0
17	TX_DONE	Transmit Completion Flag Bit. Flag is raised when transmitted data reaches transmit length, cleared by software writing 1 0: Not completed 1: Completed Note: Detailed description see "Operation Modes" chapter	RW	0x0
16:14	--	Reserved	RO	0x0
13	DONE	Transmission Completion Flag Bit. Flag is raised when transmission/reception of one data is completed and transmit FIFO is empty, cleared by software writing 1 0: Not completed 1: Completed	RW	0x0
12	RX_FULL	Receive FIFO Full Flag Bit 0: Not full	RO	0x0

		1: Full		
11	TX_EMPTY	Transmit FIFO Empty Flag Bit 0: Not empty 1: Empty	RO	0x1
10	TX_OVR_ERR	Transmit FIFO Overflow Error Flag Bit. Flag is raised when continuing to write QSPI_RXDATA register while transmit FIFO is full, cleared by software writing 1 0: Not overflowed 1: Overflowed	RW	0x0
9	RX_UDR_ERR	Receive FIFO Underflow Error Flag Bit. Flag is raised when reading QSPI_RXDATA register while receive FIFO is empty, cleared by software writing 1 0: Not underflowed 1: Underflowed	RW	0x0
8:6	--	Reserved	RO	0x0
5	RX_EMPTY	Receive FIFO Empty Flag Bit 0: Not empty 1: Empty	RO	0x1
4	TX_FULL	Transmit FIFO Full Flag Bit 0: Not full 1: Full	RO	0x0
3	TX_UDR_ERR	Transmit FIFO Underflow Error Flag Bit. Flag is raised when there is no data in transmit FIFO during transmission, cleared by software writing 1 0: Not underflowed 1: Underflowed	RW	0x0
2	RX_OVR_ERR	Receive FIFO Overflow Error Flag Bit. Flag is raised when receive FIFO is full after receiving one data, cleared by software writing 1 0: Not overflowed 1: Overflowed	RW	0x0
1	--	Reserved	RO	0x0
0	BUSY	QSPI Transmission Busy Flag Bit (Master/Slave) 0: Idle 1: Transmitting	RO	0x0

26.6.27 QSPI_SDR_SCKSAMPLE（0x80）

QSPI Sampling Point Configuration Register (SDR Mode)

Offset Address = 0x80

Bit	Name	Description	R/W	Reset Value
31:12	--	Reserved	RO	0x0
11:0	SDR_SCKSA	SDR Mode, SCK Sampling Point Selection Bit	RW	0x0

	MPLE	0x000: No delay 0x001: Delay 1 bus clock cycle for sampling 0x002: Delay 2 bus clock cycles for sampling ..... Note: Sampling delay must be less than 1 QSPI_SCK cycle		

26.6.28 QSPI_CR（0x84）

QSPI Control Register

Offset Address = 0x84

Bit	Name	Description	R/W	Reset Value
31:14	--	Reserved	RO	0x0
13	SPI_RXFIFO_EN	Receive FIFO Enable Bit 0: Disable 1: Enable (default)	RW	0x1
12	--	Reserved	RO	0x0
11	DMA_TXCT_MODE	DMA Continuous Transmit Mode Enable Bit 0: Disable 1: Enable	RW	0x0
10	DMA_RXCT_MODE	DMA Continuous Receive Mode Enable Bit 0: Disable 1: Enable	RW	0x0
9	DMA_RX_EN	Receive DMA Mode Enable Bit 0: Disable 1: Enable	RW	0x0
8	DMA_TX_EN	Transmit DMA Mode Enable Bit 0: Disable 1: Enable	RW	0x0
7:6	--	Reserved	RO	0x0
5	SSM	Software Slave Management CS Enable Bit 0: Hardware management 1: Software management	RW	0x0
4	CSOE	Master Mode, CS Output Enable Bit 0: Disable 1: Enable	RW	0x1
3	CSI	Software Management Slave Mode, CS Enable Bit 0: Software enable CS 1: Software disable CS Note: Effective when SSM bit is configured to 1	RW	0x0
2	DDR_EN	DDR Mode Enable Bit, only supported in Master mode 0: SDR mode 1: DDR mode	RW	0x0
1	DMA_EN	DMA Mode Enable Bit	RW	0x0

		0: Register mode 1: DMA mode		
0	MSTER	Master/Slave Mode Selection Bit 0: Slave mode 1: Master mode	RW	0x1

26.7 Software Configuration Process and Application Notes

26.7.1 Register and DMA Modes

- Correctly configure QSPI pin multiplexing according to the "Multiplexing Function List" and "GPIO Register Configuration Process";
- Configure the QSPI_CR register to set Master mode, Register/DMA mode, and SDR/DDR mode respectively.
- Configure the QSPI_SCKDIV and QSPI_SCKMODE registers to select the division factor, phase, and polarity of the QSPI output clock;
- Configure the QSPI_SCKSAMPLE and QSPI_SDR_SCKSAMPLE registers to set the sampling point;
- Configure the QSPI_FMT register to select the interface mode, LMSB, transmission direction, and data bit width of the QSPI data line;
- Configure the QSPI_TSIZE and QSPI_RSIZE registers to set the transmit and receive data lengths.
- Configure the QSPI_IE register to enable interrupts. All interrupts use the same interrupt entry. Determine which interrupt is raised by reading the QSPI_STATUS register, and some interrupts can be cleared by writing the QSPI_STATUS register;
- Write the data to be transmitted into the transmit FIFO through the QSPI_TXDATA register. For data reception, start a transmission by writing to the transmit FIFO as well;
- The data in the receive FIFO can be obtained by reading the QSPI_TXDATA register;
- Wait for the interrupt event and repeat the above steps according to application requirements.

26.7.2 XIP Mode

- Correctly configure QSPI pin multiplexing according to the "Multiplexing Function List" and "GPIO Register Configuration Process";
- Correctly configure the QSPI_ADDR_WRAP and QSPI_BOUNDARY_CFG registers according to the start address and capacity of the SPI Flash slave;
- Configure the QSPI_SCKDIV and QSPI_SCKMODE registers to select the division factor, phase, and polarity of the QSPI output clock;
- Configure the QSPI_SCKSAMPLE and QSPI_SDR_SCKSAMPLE registers to set the sampling point;
- Configure the QSPI_FFMT and QSPI_FFMT1 registers to set basic parameters such as the length and interface mode of the instruction, address, mode, dummy cycle, and data phases;
- Configure the QSPI_FCTRL register to enable XIP functions;
- Then the CPU can access the SPI Flash as internal storage through the XIP bus.

26.7.3 Notes

- When QSPI is used in register and DMA modes, the following two points need to be paid special attention to:
  - The QSPI_FCTRL register must be configured first to disable XIP;
  - The QSPI_FMT register must be configured first to refresh the transmission frame and line mode, then configure the QSPI_CR register;

- Recommended software configuration sequence: Configure other registers → Clear STATUS flags → Configure CR register to ensure that the residual flags of the previous transmission are cleared.

- Usage scenarios and notes for the QSPI_SDR_SCKSAMPLE and QSPI_SCKSAMPLE registers:
  - When PAD and IO delays are large, the sampling can be delayed by configuring this register. The configured delay must be less than 1 QSPI SCK cycle;
  - When the configured value of this register is large, it is recommended that the interrupt function query the QSPI_STATUS.busy flag to determine whether the transmission is completed.
- When FMT.DIR is 1 (i.e., transmitting), the hardware disables data reception, and no data transmission will occur after the RX DMA is enabled.
- After the transmission ends, the residual data in the RX FIFO can be cleared by reading the RXDATA register multiple times, and the residual data in the TX FIFO can only be reset by resetting the QSPI module.
- In the slave 4-wire QSPI mode, it is recommended to define the QSPI frame as 16/32 bits, as the underflow risk is higher for 8-bit frames.
- Notes for XIP mode:
  - When connecting external storage such as FLASH or SRAM, QSPI must be configured as Master mode;
  - For XIP read operations, SPI_RXFIFO_EN must be configured to 1; otherwise, the read data will be 0;
  - In XIP mode, it is prohibited to configure the instruction, address, and mode phases to 0 simultaneously.
- Notes for DDR mode:
  - The instruction phase of XIP does not support being configured into DDR mode, i.e., bit 14 of FFMT1 is prohibited from being configured to 1;
  - In XIP DDR mode, the instruction and dummy cycle phases must exist;
  - In DDR mode, only CPHA=0 is supported.
  - In terms of transmission efficiency, the minimum SCKDIV can be configured to 0 (2x division) in SDR mode, and the minimum SCKDIV can be configured to 1 (4x division) in DDR mode. Therefore, their ultimate transmission efficiencies are the same. It is recommended not to use DDR mode unless necessary.
- Notes for GDMA usage:
  - It is recommended to configure and enable GDMA first, then configure QSPI DMA enable;
  - The DMA transmission length can be configured only on the GDMA side. The TSIZE and RSIZE on the QSPI side do not need to be configured, and the default value 0xffff ffff can be used. GDMA will disable the enable after reaching the DMA transmission length and no longer respond to QSPI DMA requests;
  - Software uses the GDMA transmission completion flag, and combines the QSPI busy and tx/rx num flags in the GDMA interrupt function.

27 GPIO

27.1 Introduction

GPIO (General Purpose I/O) is a general-purpose input/output port. Each general-purpose I/O port includes 6 control registers (GPIOx_MODE, GPIOx_AEN, GPIOx_OTYPE, GPIOx_OSPEED, GPIOx_ITYPE, and GPIOx_UPDN), 2 data registers (GPIOx_IDATA and GPIOx_ODATA), 2 alternate function registers (GPIOx_AFSEL0 and GPIOx_AFSEL1), and 3 bitwise operation registers (GPIOx_SET, GPIOx_CLR, and GPIOx_TG).

There are 118 GPIOs in total, including ports PA, PB, PC, PD, PE, PF, PG, PH, and PL:

- Ports PA, PB, PC, PD, PE, PF, and PG each contain 16 I/O pins, namely Px00~Px15 (x=A-G);
- Port PH contains 2 I/O pins, namely PH00 and PH01;
- Port PL contains 4 I/O pins, namely PL00~PL03, which are I/Os in the Vbat domain. The configuration priority of the Vbat domain is higher than that of the VCC domain.

27.2 Features

- Configurable modes: Input, Output, Alternate Function, and Analog function modes;
- Configurable input states: CMOS/TTL input + Pull-up/Pull-down;
- Configurable output states: Normal/Open-drain output + Driving capability;
- Supports bitwise operation function for GPIOx_ODATA output data;
- Supports fast toggling;
- Refer to the "System Control" chapter for the I/O usage permission register;
- Supports secure access;
- GPIO is an AHB peripheral and supports the Bitband function.

27.3 Block Diagram

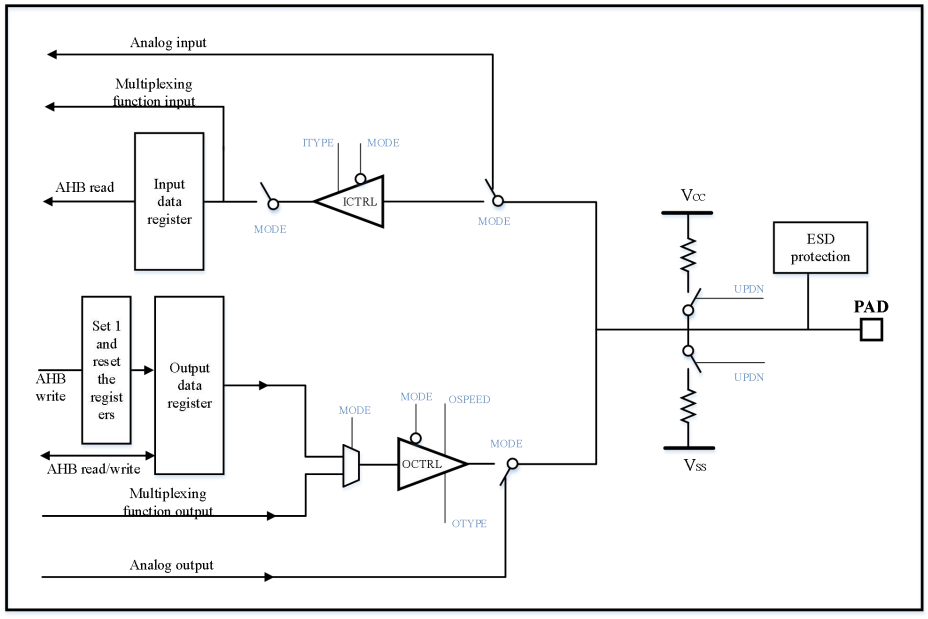


Figure 27-1 GPIO Block Diagram

27.4 Functional Description

27.4.1 Functional Configuration Description

- Each I/O supports the following configurations:
- Supports four working modes: Input/Output/Alternate Function/Analog, controlled by the GPIOx_MODE register;
- Supports two input types: CMOS/TTL, controlled by the GPIOx_ITYPE register;
- Supports two output types: Normal output/Open-drain output, controlled by the GPIOx_OTYPE register;
- Supports two output drives: 3.0/6.0 mA, controlled by the GPIOx_OSPEED register;
- Supports Floating/Pull-up/Pull-down functions, controlled by the GPIOx_UPDN register;
- Supports 16 alternate function configurations, controlled by the GPIOx_AFSEL0 and GPIOx_AFSEL1 registers;
- Supports I/Os with analog functions, and the analog function enable is controlled by the GPIOx_AEN register.

Table 27-1 GPIO Configuration Table

	Register Configuration					
MODE	AEN	ITYPE	OTYPE	OSPEED	UPDN	AFSEL0/1
Input Mode 00b	×	CMOS/ TTL 0/1	×	×	Floating/Pull Up/Pull Down 00b/01b/10b	×
Output Mode 01b	×		Normal/Open-Drain Output 0/1	Drive current 3.0/6.0 mA 0/1	Floating/×/× 00b/01b/10b	×
Alternate Function 10b	×				Floating/Pull Up/Pull Down00b/01b/10b	AF00~AF15 0000b~1111b
Analog Mode 11b	Analog function Disable/Enable 0/1	×	×	×	Floating/×/× 00b/01b/10b	×

Note 1: "×" indicates that the configuration is invalid. Different types of PADs support different configurations. See the PAD type description list for details.Note 2: When the alternate function is output, pull-up and pull-down are invalid.

27.4.2 General I/O Description

During and after reset, most I/O ports are configured as Analog mode (high impedance) by default. The following are special I/Os:

- PG04: Configured as Analog mode (high impedance) by default. After chip reset, the Bootloader program in the ROM will configure this pin as Pull-up Input mode:
  - When PG04 is low, the Bootloader program will enter the ISP process and will not boot the FLASH application program;
  - When PG04 is high, the Bootloader program will boot the FLASH application program and restore the PG04 pin to the default Analog mode (high impedance).

Note: Applications should try to avoid connecting an external pull-down resistor or external driving low level to PG04, otherwise the chip will not boot the FLASH application program correctly after reset.

- PG14: JTCK, configured through GPIO_DBG[0], the function is enabled by default. After enabling, the I/O is

Pull-down Input;

- PG15: JTMS, configured through GPIO_DBG[1], the function is enabled by default. After enabling, the I/O is Pull-up Input;
- PD04: LCDVD pin, no GPIO function;
- PA02, PA03, PA04, and PE15: I/O ports with analog hardware linkage functions. When the following analog functions are enabled, they are forced to be analog pins.
  - PA02: Defaults to CMP1 function after power-on, GPIOx_MODE configuration is invalid;
  - PA03: Defaults to CMP2 function after power-on, GPIOx_MODE configuration is invalid;
  - PA04: Defaults to Analog mode (high impedance) after power-on. When LVD is enabled (i.e., LVD_PWD!=0x55aaaa55), the GPIOx_MODE configuration is invalid;
  - PE14, PE15: Default to Analog mode (high impedance) after power-on, serving as HOSC I/Os. That is, when HOSC_PD is configured to 0, they are forced to Analog mode (high impedance), and the GPIOx_MODE configuration is invalid.
- PL00~PL03: Vbat domain I/Os. The I/O function configuration priority of the Vbat domain is higher than the general GPIO configuration of the Vcc domain. Their analog functions are only controlled by the Vbat domain. After the Vcc domain is powered down, if the I/O is configured as a digital/analog function of Vbat, it is not affected by the Vcc domain power-down. If the I/O is configured as a general GPIO function, it is in high impedance. For Vbat domain configuration, refer to the RTC chapter.

Note: The GPIO_DBG register is an IO enable register for chip JTAG debugging, with higher priority than the GPIOx register. After this register is configured to enable the JTAG function, the corresponding IO's GPIOx register configuration becomes invalid. For detailed description, see the register section.

27.4.3 Input Mode

When the I/O pin is configured as Input:

- The Schmitt trigger input is enabled, and the input type is selected as CMOS or TTL through GPIOx_ITYPE;
- The I/O pin is selected as Floating, Pull-up, or Pull-down through GPIOx_UPDN;
- The output buffer is disabled, and both GPIOx_OTYPE and GPIOx_OSPEED configurations are invalid;
- The I/O input is sampled at each fclk rising edge and stored in GPIOx_IDATA;
- The I/O input state can be obtained by reading GPIOx_IDATA.

27.4.4 Output Mode

When the I/O pin is configured as Output:

- The input Schmitt trigger is disabled;
- The output buffer is enabled. The output type is selected as Normal output or Open-drain output through GPIOx_OTYPE, and the I/O output driving capability is selected through GPIOx_OSPEED;
- The return value of reading GPIOx_IDATA is "0";
- The last written value can be returned by reading GPIOx_ODATA.

27.4.5 Alternate Function Mode

When the I/O pin is configured as Alternate Function:

- The peripheral function of the I/O port multiplexing is selected through GPIOx_AFSEL0 or GPIOx_AFSEL1;
- The I/O pin is selected as Floating, Pull-up, or Pull-down through GPIOx_UPDN;
- The Schmitt trigger input enable is driven by the selected peripheral;
- The output buffer enable is driven by the selected peripheral.

Note 1: If the I/O alternate function is LCD, do not apply the input/output configuration of the alternate function. Refer to the input/output description of the Analog mode.

Note 2: For alternate functions, refer to the "Alternate Function List" chapter.

27.4.6 Analog Mode

When the I/O pin is configured as Analog mode:

- The input Schmitt trigger is disabled;
- The output buffer is disabled;
- Pull-up and pull-down are hardware-closed;
- The return value of reading GPIOx_IDATA is "0";
- When no analog function is connected to the pin, it is in high impedance;
- When an analog function is connected to the pin, the analog function enable of the pin is controlled by GPIOx_AEN.

Note 1: For specific analog function configuration, refer to the registers in each analog function chapter.

Note 2: For analog functions, refer to the "Pin Definition" chapter.

27.4.7 Forced Input Enable

When the I/O is in Output mode, the input Schmitt trigger is disabled; when the I/O is in Alternate Function mode, the input Schmitt trigger enable is controlled by the multiplexed peripheral.

In these two modes, if there is a need to continuously turn on the pin input, the corresponding bit of the GPIOx_FIE register can be configured to 0 to keep the pin's input Schmitt trigger enabled. At this time, the I/O input state can be obtained by reading GPIOx_IDATA.

27.4.8 Bitwise Operation

The bitwise operations of GPIO include Set, Clear, and Toggle:

- Writing 1 to the GPIOx_SET register can set the corresponding bit of GPIOx_ODATA;
- Writing 1 to the GPIOx_CLR register can clear the corresponding bit of GPIOx_ODATA;
- Writing 1 to the GPIOx_TG register can toggle the level of the corresponding bit of GPIOx_ODATA;
- AHB Bitband direct access to GPIOx_ODATA can implement bit clear/set operations:
  - The Bitband function is enabled when the 25th bit of the AHB bus address is 1;
  - Bits 2~6 of the AHB bus address indicate bits 0~31 of the access register;
  - Bits 2~13 of the original AHB bus address are mapped to bits 7~18;
  - Bit 0 of the AHB write data is the write value.

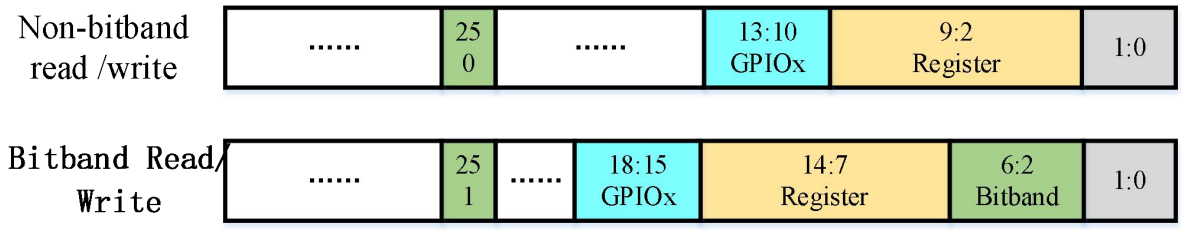


Figure 27-2 Bitband Read/Write

27.4.9 Secure Access

1.4.9.1 Register Secure Access

Each I/O has a 1-bit security attribute register. See the SCU chapter for details. For general GPIOx registers, when the security attribute of the accessing host matches the I/O attribute, it is a legal access and the access is valid. In the following cases, it is considered an illegal access and the access is invalid:

- A secure host (core0) accesses a non-secure I/O: write access is invalid, and read returns "0";
- A non-secure host (core1) accesses a secure I/O: write access is invalid, and read returns "0".

Among them, the registers GPIO_DBG_KEY and GPIO_DBG related to JTAG debugging can only be accessed by the secure host core0.

27.4.9.1 Peripheral Security

Each I/O can be multiplexed with up to 16 different digital peripheral functions. According to the security attributes of the I/O and the multiplexed peripheral, the following security protection measures are taken for the peripheral input/output:

- When the I/O attribute is secure, the input/output function of the multiplexed peripheral is normal;
- When the I/O attribute is non-secure and the multiplexed peripheral attribute is also non-secure, the input/output of the multiplexed peripheral is normal;
- When the I/O attribute is non-secure and the multiplexed peripheral attribute is secure, the input/output of the multiplexed peripheral is invalid.

27.5 Registers

Module Name	Physical Address	Mapped Address
GPIOA	0xB008 0000	0xB008 0000
GPIOB	0xB008 0400	0xB008 0400
GPIOC	0xB008 0800	0xB008 0800
GPIOD	0xB008 0C00	0xB008 0C00
GPIOE	0xB008 1000	0xB008 1000
GPIOF	0xB008 1400	0xB008 1400
GPIOG	0xB008 1800	0xB008 1800
GPIOH	0xB008 1C00	0xB008 1C00
GPIOL（VBAT）	0xB008 2C00	0xB008 2C00

27.5.1 Register List

Register Name	Address Offset	Description
GPIOx_MODE	0x00	GPIO Mode Register
GPIOx_AEN	0x04	GPIO Analog Function Enable Register
GPIOx_OTYPE	0x08	GPIO Output Type Register
GPIOx_OSPEED	0x0C	GPIO Output Driving Capability Register
GPIOx_FIE	0x10	GPIO Forced Input Enable Register
GPIOx_ITYPE	0x14	GPIO Input Type Register
GPIOx_UPDN	0x18	GPIO Pull-up/Pull-down Register
GPIOx_IDATA	0x1C	GPIO Input Data Register
GPIOx_ODATA	0x20	GPIO Output Data Register

GPIOx_AFSEL0	0x24	GPIO Alternate Function Selection Register 0
GPIOx_AFSEL1	0x28	GPIO Alternate Function Selection Register 1
GPIOx_SET	0x2C	GPIO Bit Set Register
GPIOx_CLR	0x30	GPIO Bit Clear Register
GPIOx_TG	0x34	GPIO Bit Toggle Register

Addresses of GPIO JTAG Debug Registers

Register Name	Physical Address	Description
GPIO_DBG_KEY	0xB008 0070	Password Register for GPIO_DBG Register
GPIO_DBG	0xB008 0074	GPIO JTAG Debug Configuration Register

27.5.2 GPIOx_MODE（0x00）

GPIOx Mode Register (x = A, B, C, D, E, F, G, H, L)

Offset Address = 0x00

Supports Bitband write operation

Reset Value:

- Port D reset value: 0xFFFF FCFF (PD04 reserved)
- Port H reset value: 0x0000 000F
- Port L reset value: 0x0000 00FF
- Reset value of other ports: 0xFFFF FFFF

Reserved Ports: Port D[9:8], Port H[31:4], and Port L[31:8]

Bits	Name	Description	R/W	Reset
31:0	MODE[15:0][1:0]	Px00~Px15 Mode Control Bits 00: Input Mode 01: Output Mode 10: Alternate Function Mode 11: Analog Mode (High-Z) (Default)	RW	0x3

27.5.3 GPIOx_AEN（0x04）

GPIOx Analog Function Enable Register (x = A, B, C, D, E, F, G, H, L)

Offset Address = 0x04

Supports Bitband write operation

Reset Value: 0x0000 0000

Reserved Ports: Port D[4], Port H[31:2], and Port L[31:0]

Bits	Name	Description	R/W	Reset
31:16	---	Reserved	RO	0x0
15:0	ANA_EN[15:0]	Px00~Px15 Analog Function Enable Bits 0: Disable 1: Enable	WR	0x0

		Note: This register configuration takes effect only when the corresponding I/O is in Analog Mode.		

27.5.4 GPIOx_ OTYPE（0x08）

GPIOx Output Type Register (x = A, B, C, D, E, F, G, H, L)

Offset Address = 0x08

Supports Bitband write operation

Reset Value: 0x0000 0000

Reserved Ports: Port D[4], Port H[31:2], and Port L[31:4]

Bits	Name	Description	R/W	Reset
31:16	---	Reserved	RO	0x0
15:0	OTYPE[15:0]	Px00~Px15 Output Type Control Bits 0: Normal Output (Default) 1: N-ch Open Drain Output	RW	0x0

27.5.5 GPIOx_OSPEED（0x0C）

GPIOx Output Driving Capability Register (x = A, B, C, D, E, F, G, H, L)

Offset Address = 0x0C

Supports Bitband write operation

Reset Value: 0x0000 0000

Reserved Ports: Port D[4], Port H[31:2], and Port L[31:4]

Bits	Name	Description	R/W	Reset
31:16	---	Reserved	RO	0x0
15:0	OSPEED[15:0]	Px00~Px15 Output Drive Capability Control Bits 0: 3.0 mA (Default) 1: 6.0 mA	RW	0x0

27.5.6 GPIOx_FIE（0x10）

GPIOx Forced Input Enable Register (x = A, B, C, D, E, F, G, H, L)

Offset Address = 0x10

Supports Bitband write operation

Reset Value:

- Port D reset value: 0x0000 FFEF
- Port H reset value: 0x0000 0003
- Port L reset value: 0x0000 000F
- Reset value of other ports: 0x0000 FFFF

Reserved Ports: Port D[4], Port H[31:2], and Port L[31:4]

Bits	Name	Description	R/W	Reset
31:16	---	Reserved	RO	0x0
15:0	FIE[15:0]	Px00~Px15 Input Force Enable Bits 0: Input Force Enable 1: Invalid	RW	0x1

		Note: This register is valid only when the I/O is in Output Mode or Alternate Function Mode.		

27.5.7 GPIOx_ITYPE（0x14）

GPIOx Input Type Register (x = A, B, C, D, E, F, G, H, L)

Offset Address = 0x14

Supports Bitband write operation

Reset Value: 0x0000 0000Reserved Ports: Port D[4], Port H[31:2], and Port L[31:4]

Bits	Name	Description	R/W	Reset
31:16	---	Reserved	RO	0x0
15:0	ITYPE[15:0]	Px00~Px15 Input Type Control Bits 0: CMOS Buffer, Vil=0.3VCC, Vih=0.7VCC (Default) 1: TTL Buffer, Vil=0.16VCC, Vih=0.4VCC	RW	0x0

27.5.8 GPIOx_UPDN（0x18）

GPIOx Pull-up/Pull-down Register (x = A, B, C, D, E, F, G, H, L)

Offset Address = 0x18

Supports Bitband write operation

Reset Value: 0x0000 0000Reserved Ports: Port D[9:8], Port H[31:4], and Port L[31:8]

Bits	Name	Description	R/W	Reset
31:0	UPDN[15:0][1:0]	Px00~Px15 Pull-up/Pull-down Control Bits, Invalid in Analog Mode 00: Floating (Default) 01: Pull-up 10: Pull-down 11: Reserved	RW	0x0

27.5.9 GPIOx_IDATA（0x1C）

GPIOx Input Data Register (x = A, B, C, D, E, F, G, H, L)

Offset Address = 0x1C

Reserved Ports: Port D[4], Port H[31:2], and Port L[31:4]

Bits	Name	Description	R/W	Reset
31:16	---	Reserved	RO	0x0
15:0	IDATA[15:0]	Px00~Px15 Input Data 0: Low Level 1: High Level	RO	0x0

Note: This register supports Bitband read operations.

27.5.10 GPIOx_ODATA（0x20）

GPIOx Output Data Register (x = A, B, C, D, E, F, G, H, L)

Offset Address = 0x20

Supports Bitband write operation

Reset Value: 0x0000 0000

Reserved Ports: Port D[4], Port H[31:2], and Port L[31:4]

Bits	Name	Description	R/W	Reset
31:16	---	Reserved	RO	0x0
15:0	ODATA[15:0]	Px00~Px15 Output Data 0: Low Level 1: High Level	RW	0x0

27.5.11 GPIOx_AFSEL0（0x24）

GPIOx Alternate Function Selection Register 0 (x = A, B, C, D, E, F, G, H, L)

Offset Address = 0x24

Supports Bitband write operation

Reset Value: 0x0000 0000

Reserved Ports: Port D[19:16], Port H[31:8], and Port L[31:16]

Bits	Name	Description	R/W	Reset
31:0	SEL[7:0][3:0]	Px00~Px07 Alternate Function Selection Bits 0000: Select AF00 Function 0001: Select AF01 Function ..... 1110: Select AF14 Function 1111: Select AF15 Function	RW	0x0

27.5.12 GPIOx_AFSEL1（0x28）

GPIOx Alternate Function Selection Register 1 (x = A, B, C, D, E, F, G)

Offset Address = 0x28

Supports Bitband write operation

Reset Value: 0x0000 0000

Bits	Name	Description	R/W	Reset
31:0	SEL[7:0][3:0]	Px08~Px15 Alternate Function Selection Bits 0000: Select AF00 Function 0001: Select AF01 Function ..... 1110: Select AF14 Function 1111: Select AF15 Function	RW	0x0

27.5.13 GPIOx_SET（0x2C）

GPIOx Bit Set Register (x = A, B, C, D, E, F, G, H, L)

Offset Address = 0x2C

Reset Value: 0x0000 0000

Reserved Ports: Port D[4], Port H[31:2], and Port L[31:4]

Bits	Name	Description	R/W	Reset
31:16	---	Reserved	RO	0x0
15:0	SET[15:0]	Px00~Px15 Bit Set Bits 0: Invalid 1: Corresponding ODATA bit set to 1 Note: This bit is valid only when the corresponding I/O is in Output Mode.	WO	0x0

27.5.14 GPIOx_CLR（0x30）

GPIOx Bit Clear Register (x = A, B, C, D, E, F, G, H, L)

Offset Address = 0x30

Reset Value: 0x0000 0000

Reserved Ports: Port D[4], Port H[31:2], and Port L[31:4]

Bits	Name	Description	R/W	Reset
31:16	---	Reserved	RO	0x0
15:0	CLR[15:0]	Px00~Px15 Bit Clear Bits 0: Invalid 1: Corresponding ODATA bit cleared to 0 Note: This bit is valid only when the corresponding I/O is in Output Mode.	WO	0x0

27.5.15 GPIOx_TG（0x34）

GPIOx Bit Toggle Register (x = A, B, C, D, E, F, G, H, L)

Offset Address = 0x34

Reset Value: 0x0000 0000

Reserved Ports: Port D[4], Port H[31:2], and Port L[31:4]]

Bits	Name	Description	R/W	Reset
31:16	---	Reserved	RO	0x0
15:0	TG[15:0]	Px00~Px15 Bit Toggle Function 0: Invalid 1: Corresponding ODATA bit toggled Note: This bit is valid only when the corresponding I/O is in Output Mode.	WO	0x0

27.5.16 GPIO_DBG_KEY（0xB008 0070）

Password Register for GPIO_DBG Register

Address = 0xB008 0070

Reset Value: 0x0000 0000

Bits	Name	Description	R/W	Reset
31:0	KEY	GPIO_DBG Register Password Register Before each write to the GPIO_DBG register, the	WO	0x0

		following sequence must be written to this bit in order: KEY = 0x55dd 77ee KEY = 0x6688 3399 KEY = 0xddaa bb11 If the password sequence is correct, the next write to the GPIO_DBG register is valid; if an incorrect value is written or other registers are read/written during the process, the password sequence must be restarted.		

27.5.17 GPIO_DBG（0xB008 0074）

GPIO JTAG Debug Configuration Register

Address = 0xB008 0074Reset Value: 0x0000 0003

The password for this register is shown in the GPIO_DBG_KEY register

Bits [3:2] are power-on reset and CORE0 WDT reset domain registers

Bits	Name	Description	R/W	Reset
31:4	---	Reserved	RO	0x0
3	JTDO_EN	JTAG JTDO Function Configuration Bit 0: PG10 used as General-purpose GPIO 1: PG10 used as JTDO Function	RW	0x0
2	JTDI_EN	JTAG JTDI Function Configuration Bit 0: PG11 used as General-purpose GPIO 1: PG11 used as JTDI Function, pin pull-up	RW	0x0
1	JTMS_EN	JTAG JTMS Function Configuration Bit 0: PG15 used as General-purpose GPIO 1: PG15 used as JTMS Function, pin pull-up	RW	0x1
0	JTCK_EN	JTAG JTCK Function Configuration Bit 0: PG14 used as General-purpose GPIO 1: PG14 used as JTCK Function, pin pull-down	RW	0x1

Note: This register is an IO enable register for chip JTAG debugging, with higher priority than the GPIOx register. After this register is configured to enable the JTAG function, the corresponding IO's GPIOx register configuration becomes invalid.

27.6 Alternate Function List

Appendix Table 1 Port A Alternate Function List

Port A Alternate Function																
Port	AF00	AF01	AF02	AF03	AF04	AF05	AF06	AF07	AF08	AF09	AF10	AF11	AF12	AF13	AF14	AF15
	SYS	QSPI0	SPI0 QSPI1	SPI1	SPI2 UART0/2	SPI3 UART1/3	UART0-7	UART0-7	LPUART CAN RTC1S	I2C0-1 ISO78160-1	KEY0-1	USART0-1 TCO	TCI TCO	CF0-2 IOCNT0-4	Reserve	LCD
PA00	-	-	-	-	-	-	UART2_RX	UART3_RX	-	-	KEY0[2]	-	-	-	-	-
PA01	-	-	-	-	-	-	UART2_TX	UART3_TX	-	-	KEY0[3]	-	-	-	-	-
PA02	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-
PA03	-	-	-	-	-	-	UART6_RX	UART7_RX	-	-	-	-	-	-	-	-
PA04	-	-	-	-	-	-	UART6_TX	UART7_TX	-	-	KEY0[4]	-	-	-	-	-
PA05	-	-	-	-	-	SPI3_SCSN	-	-	-	-	-	USART0_TX	-	-	-	-
PA06	-	-	-	-	-	SPI3_SCLK	-	-	-	-	-	USART0_RX	-	-	-	-
PA07	-	-	-	-	-	SPI3_MISO	-	-	-	-	-	USART0_CK	-	-	-	-
PA08	-	-	-	-	-	SPI3_MOSI	-	-	-	-	-	USART0_RTS	-	-	-	-
PA09	-	-	-	-	-	-	-	-	-	-	KEY0[2]	-	TC0_P[0]	-	-	-
PA10	TC0_N[0]	-	-	-	-	-	-	-	-	-	KEY0[1]	USART0_CTS	TCIN[8]	-	-	-
PA11	-	QSPI0_CSN	SPI0_SCSN	SPI1_SCSN	SPI2_SCSN	SPI3_SCSN	-	-	-	-	KEY0[4]	TC1_N[0]	TCIN[4]	-	-	-
PA12	-	QSPI0_SCLK	SPI0_SCLK	SPI1_SCLK	SPI2_SCLK	SPI3_SCLK	-	-	-	-	KEY0[5]	TC1_P[0]	TCIN[5]	-	-	-
PA13	-	QSPI0_SI	SPI0_MISO	SPI1_MISO	SPI2_MISO	SPI3_MISO	-	-	-	-	KEY0[6]	TC1_N[1]	TCIN[6]	-	-	-
PA14	-	QSPI0_SO	SPI0_MOSI	SPI1_MOSI	SPI2_MOSI	SPI3_MOSI	-	-	-	-	KEY0[7]	TC1_P[1]	TCIN[7]	-	-	-
PA15	-	QSPI0_WPN	-	-	-	-	UART4_RX	UART5_RX	-	-	KEY0[6]	-	-	-	-	-

Appendix Table 2 Port B Alternate Function List

Port B Alternate Function																
Port	AF00	AF01	AF02	AF03	AF04	AF05	AF06	AF07	AF08	AF09	AF10	AF11	AF12	AF13	AF14	AF15
	SYS	QSPI0	SPI0 QSPI1	SPI1	SPI2 UART0/2	SPI3 UART1/3	UART0-7	UART0-7	LPUART CAN RTC1S	I2C0-1 ISO78160-1	KEY0-1	USART0-1 TCO	TCI TCO	CF0-2 IOCNT0-4	Reserve	LCD
PB00	-	QSPI0_CSN	SPI0_SCSN	-	-	-	UART4_TX	UART5_TX	CAN_RX	78160_CLK	-	-	-	-	-	-
PB01	-	QSPI0_SCLK	SPI0_SCLK	-	-	-	UART4_TX	UART5_TX	CAN_TX	78160_IO	-	-	-	-	-	-
PB02	-	QSPI0_SI	SPI0_MISO	-	-	-	UART4_RX	UART5_RX	CAN_RX	78160_I	-	-	-	-	-	-
PB03	-	QSPI0_SO	SPI0_MOSI	-	-	-	UART6_TX	UART7_TX	CAN_TX	78161_CLK	-	-	-	-	-	-
PB04	-	QSPI0_HOLDN	-	-	-	-	UART6_RX	UART7_RX	-	-	KEY1[0]	-	-	-	-	-
PB05	CF1	-	SPI0_MISO	SPI1_MISO	SPI2_MISO	SPI3_MISO	UART4_RX	UART5_RX	LPUART_RX	78161_IO	-	-	-	CF2	-	-
PB06	CF0	-	SPI0_MOSI	SPI1_MOSI	SPI2_MOSI	SPI3_MOSI	UART4_TX	UART5_TX	LPUART_TX	78161_I	-	-	-	CF1	-	-
PB07	-	-	-	-	-	-	UART6_TX	UART7_TX	-	-	KEY1[1]	TC0_N[0]	-	-	-	-
PB08	-	-	-	-	-	-	-	-	-	I2C0_SDA	KEY0[12]	-	-	IOCNT4	-	SEG35
PB09	-	-	-	-	-	-	-	-	-	I2C0_SCL	KEY0[11]	-	-	IOCNT0	-	SEG34
PB10	-	-	-	-	-	-	-	-	-	-	KEY0[10]	-	-	CF2	-	SEG33
PB11	-	-	-	-	-	-	-	-	-	-	KEY0[9]	-	-	CF1	-	SEG32
PB12	-	-	-	-	-	-	-	-	-	-	KEY0[8]	-	-	CF0	-	SEG31
PB13	-	-	SPI0_SCLK	SPI1_SCLK	SPI2_SCLK	SPI3_SCLK	UART6_RX	UART7_RX	LPUART_RX	-	KEY1[7]	TC0_P[0]	TCIN[0]	-	-	-
PB14	RTCOUT	-	-	-	-	-	-	-	RTC1S	-	KEY1[15]	-	-	-	-	SEG30
PB15	-	-	-	-	-	-	-	-	-	I2C1_SDA	KEY1[14]	-	-	-	-	SEG29

Appendix Table 3 Port C Alternate Function List

Port C Alternate Function																
Port	AF00	AF01	AF02	AF03	AF04	AF05	AF06	AF07	AF08	AF09	AF10	AF11	AF12	AF13	AF14	AF15
	SYS	QSPI0	SPI0 QSPI1	SPI1	SPI2 UART0/2	SPI3 UART1/3	UART0-7	UART0-7	LPUART CAN RTC1S	I2C0-1 ISO78160-1	KEY0-1	USART0-1 TCO	TCI TCO	CF0-2 IOCNT0-4	Reserve	LCD
PC00	-	-	-	-	-	-	-	-	-	I2C1_SCL	KEY1[13]	-	-	-	-	SEG28
PC01	-	-	-	-	-	-	-	-	-	I2C0_SDA	KEY1[12]	-	-	-	-	SEG27
PC02	-	-	-	-	-	-	-	-	-	I2C0_SCL	KEY1[11]	-	-	IOCNT4	-	SEG26
PC03	-	-	-	-	-	-	-	-	-	-	KEY1[10]	-	-	IOCNT3	-	SEG25
PC04	-	-	-	-	-	-	-	-	-	-	KEY1[9]	-	-	IOCNT2	-	SEG24
PC05	-	-	-	-	-	-	-	-	-	-	KEY1[8]	-	-	IOCNT1	-	SEG23
PC06	-	-	SPI0_SCSN	SPI1_SCSN	SPI2_SCSN	SPI3_SCSN	UART6_TX	UART7_TX	LPUART_TX	-	KEY0[15]	TC0_N[1]	TCIN[1]	-	-	-
PC07	-	-	-	-	-	-	-	-	-	-	KEY1[7]	-	-	IOCNT0	-	SEG22
PC08	-	-	-	-	-	-	-	-	-	-	KEY1[6]	-	-	CF2	-	SEG21
PC09	-	-	-	-	-	-	-	-	-	-	KEY1[5]	-	-	CF1	-	SEG20
PC10	-	-	-	-	-	-	-	-	-	-	KEY1[4]	-	-	CF0	-	SEG19
PC11	RTCOUT	-	-	-	-	-	-	-	RTC1S	-	KEY1[3]	-	-	-	-	SEG18
PC12	-	-	-	-	-	-	-	-	-	I2C1_SDA	KEY1[2]	-	-	-	-	SEG17
PC13	-	-	-	-	-	-	-	-	-	I2C1_SCL	KEY1[1]	-	-	-	-	SEG16
PC14	-	-	-	-	-	-	UART6_RX	UART7_RX	-	-	KEY1[0]	TC0_P[1]	-	-	-	-
PC15	-	-	-	-	-	-	-	-	-	-	KEY0[5]	-	-	-	-	LCDVP2

Appendix Table 4 Port D Alternate Function List

Port C Alternate Function																
Port	AF00	AF01	AF02	AF03	AF04	AF05	AF06	AF07	AF08	AF09	AF10	AF11	AF12	AF13	AF14	AF15
	SYS	QSPI0	SPI0 QSPI1	SPI1	SPI2 UART0/2	SPI3 UART1/3	UART0-7	UART0-7	LPUART CAN RTC1S	I2C0-1 ISO78160-1	KEY0-1	USART0-1 TCO	TCI TCO	CF0-2 IOCNT0 -4	Reser ve	LCD
PD00	-	-	-	-	-	-	-	-	-	-	KEY0[4]	-	-	-	-	LCDVP1
PD01	-	-	-	-	-	-	-	-	-	-	KEY0[3]	-	-	-	-	LCDVA
PD02	-	-	-	-	-	-	-	-	-	-	KEY0[2]	-	-	-	-	LCDVB
PD03	-	-	-	-	-	-	-	-	-	-	KEY0[1]	-	-	-	-	LCDVC
PD04	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	LCDVD
PD05	-	-	-	-	-	-	UART0_TX	UART1_TX	-	-	KEY1[15]	USART1_TX	TC1_N[0]	-	-	-
PD06	-	-	-	SPI1_SCSN	-	-	-	-	-	I2C0_SCL	-	TC0_N[1]	TCIN[2]	-	-	-
PD07	-	-	-	SPI1_SCLK	-	-	-	-	-	I2C0_SDA	-	TC0_P[1]	TCIN[3]	-	-	-
PD08	-	-	-	SPI1_MISO	-	-	UART4_RX	UART5_RX	-	-	-	TC1_N[0]	TCIN[4]	-	-	-
PD09	-	-	-	SPI1_MOSI	-	-	UART4_TX	UART5_TX	-	-	-	TC1_P[0]	TCIN[5]	-	-	-
PD10	78161_CLK	QSPI0_CSN	QSPI1_CSN	SPI1_SCSN	SPI2_SCSN	-	-	-	-	78160_CLK	KEY0[9]	-	-	-	-	-
PD11	-	QSPI0_SCLK	QSPI1_SCLK	SPI1_SCLK	SPI2_SCLK	-	-	-	-	78160_IO	KEY0[11]	-	-	-	-	-
PD12	-	-	-	-	-	-	UART0_RX	UART1_RX	-	-	KEY1[14]	USART1_RX	TC1_P[0]	-	-	-
PD13	-	QSPI0_SI	QSPI1_SI	SPI1_MISO	SPI2_MISO	-	-	-	-	78161_IO	KEY0[12]	-	-	-	-	-
PD14	-	-	-	-	-	-	UART0_TX	UART1_TX	-	-	KEY1[13]	USART1_CTS	TC0_N[0]	-	-	-
PD15	-	QSPI0_SO	QSPI1_SO	SPI1_MOSI	SPI2_MOSI	-	-	-	-	78161_I	KEY0[13]	-	-	-	-	-

Appendix Table 5 Port E Alternate Function List

Port E Alternate Function																
Port	AF00	AF01	AF02	AF03	AF04	AF05	AF06	AF07	AF08	AF09	AF10	AF11	AF12	AF13	AF14	AF15
	SYS	QSPI0	SPI0 QSPI1	SPI1	SPI2 UART0/2	SPI3 UART1/3	UART0-7	UART0-7	LPUART CAN RTC1S	I2C0-1 ISO78160-1	KEY0-1	USART0-1 TCO	TCI TCO	CF0-2 IOCNT 0-4	Reser ve	LCD
PE00	-	-	-	-	-	-	UART2_RX	UART3_RX	CAN_RX	-	KEY1[12]	TC1_P[0]	TCIN[9]	IOCNT1	-	-
PE01	-	QSPI0_WPN	QSPI1_WPN	-	SPI2_SCSN	-	UART0_RX	UART1_RX	CAN_RX	-	-	USART1_RX	-	-	-	-
PE02	-	QSPI0_HOLDN	QSPI1_HOLDN	-	SPI2_SCLK	-	UART0_TX	UART1_TX	CAN_TX	-	-	USART1_TX	-	-	-	-
PE03	-	-	-	-	-	-	UART0_RX	UART1_RX	-	-	KEY1[11]	TC0_N[0]	-	-	-	-
PE04	-	-	-	-	SPI2_MISO	-	UART0_RX	UART1_RX	LPUART_RX	78161_CLK	-	USART1_CK	-	-	-	-
PE05	-	-	-	-	SPI2_MOSI	-	UART0_TX	UART1_TX	LPUART_TX	78160_I	-	USART1_RTS	-	-	-	-
PE06	-	-	-	-	-	-	-	-	-	-	KEY1[10]	-	-	IOCNT3	-	SEG15
PE07	-	-	-	-	-	-	-	-	-	-	KEY1[9]	USART1_CTS	-	IOCNT4	-	SEG14
PE08	-	-	-	-	-	-	-	-	-	I2C1_SDA	KEY1[0]	-	-	-	-	SEG13
PE09	-	-	-	-	-	-	-	-	-	I2C1_SCL	KEY1[15]	-	-	-	-	SEG12
PE10	-	-	-	-	-	-	-	-	-	-	KEY1[6]	-	-	IOCNT2	-	SEG11
PE11	-	-	-	-	-	-	-	-	-	-	KEY1[5]	-	-	IOCNT1	-	SEG10
PE12	-	-	-	-	-	-	-	-	-	-	KEY1[4]	-	-	-	-	SEG39
PE13	-	-	-	-	-	-	-	-	-	-	KEY1[3]	-	-	-	-	SEG38
PE14	-	-	-	-	UART0_TX	UART1_TX	UART2_TX	UART3_TX	CAN_TX	-	KEY0[15]	-	-	-	-	-
PE15	-	-	-	-	UART0_RX	UART1_RX	UART2_RX	UART3_RX	CAN_RX	-	KEY0[14]	-	-	-	-	-

Appendix Table 6 Port F Alternate Function List

Port F Alternate Function																
Port	AF00	AF01	AF02	AF03	AF04	AF05	AF06	AF07	AF08	AF09	AF10	AF11	AF12	AF13	AF14	AF15
	SYS	QSPI0	SPI0 QSPI1	SPI1	SPI2 UART0/2	SPI3 UART1/3	UART0-7	UART0-7	LPUART CAN RTC1S	I2C0-1 ISO78160-1	KEY0-1	USART0-1 TCO	TCI TCO	CF0-2 IOCNT0-4	Reserve	LCD
PF00	-	-	-	-	-	-	-	-	-	-	KEY1[2]	-	-	-	-	SEG37
PF01	-	-	-	-	-	-	-	-	-	-	KEY0[0]	-	-	IOCNT4	-	SEG9
PF02	-	-	-	-	-	-	-	-	-	-	KEY0[1]	-	-	CF2	-	SEG8
PF03	-	-	-	-	-	-	-	-	-	-	KEY0[2]	-	-	CF1	-	SEG7
PF04	-	-	-	-	-	-	-	-	-	-	KEY0[3]	-	-	CF0	-	SEG6
PF05	RTCOUT	-	-	-	-	-	-	-	RTC1S	-	KEY0[4]	-	-	-	-	SEG5
PF06	-	-	-	-	-	-	-	-	-	-	KEY0[5]	-	-	IOCNT3	-	SEG4
PF07	-	-	-	-	-	-	UART2_TX	UART3_TX	CAN_TX	-	KEY1[1]	TC0_P[0]	TCIN[9]	IOCNT2	-	-
PF08	-	-	-	-	-	-	-	-	-	I2C0_SCL	KEY0[0]	-	-	-	-	SEG3 /COM7
PF09	-	-	-	-	-	-	-	-	-	I2C0_SDA	KEY0[7]	-	-	-	-	SEG2 /COM6
PF10	-	-	-	-	-	-	-	-	-	-	KEY0[13]	-	-	IOCNT2	-	SEG1 /COM5
PF11	-	-	-	-	-	-	-	-	-	-	KEY0[12]	-	-	IOCNT1	-	SEG0 /COM4
PF12	-	-	-	-	-	-	-	-	-	-	KEY0[11]	-	-	IOCNT0	-	COM3
PF13	-	-	-	-	-	-	-	-	-	-	KEY0[10]	-	-	CF2	-	COM2
PF14	-	-	-	-	-	-	-	-	-	-	KEY0[15]	-	-	CF1	-	COM1
PF15	-	-	-	-	-	-	-	-	-	-	KEY0[14]	-	-	CF0	-	COM0

Appendix Table 7 Port G Alternate Function List

Port G Alternate Function																
Port	AF00	AF01	AF02	AF03	AF04	AF05	AF06	AF07	AF08	AF09	AF10	AF11	AF12	AF13	AF14	AF15
	SYS	QSPI0	SPI0 QSPI1	SPI1	SPI2 UART0/2	SPI3 UART1/3	UART0-7	UART0-7	LPUART CAN RTC1S	I2C0-1 ISO78160-1	KEY0-1	USART0-1 TCO	TCI TCO	CF0-2 IOCNT 0-4	Reser ve	LCD
PG00	-	-	-	-	-	-	-	-	-	-	KEY1[8]	-	-	-	-	SEG36
PG01	-	QSPI0_SCLK	QSPI1_SCLK	-	-	SPI3_SCSN	-	-	-	-	KEY0[0]	TC0_N[0]	TCIN[0]	-	-	-
PG02	-	QSPI0_CSN	QSPI1_CSN	-	-	SPI3_SCLK	-	-	-	-	KEY0[1]	TC0_P[0]	TCIN[1]	-	-	-
PG03	-	-	-	-	-	-	UART2_RX	UART3_RX	LPUART_RX	-	KEY0[2]	TC0_N[1]	TCIN[2]	-	-	-
PG04	-	QSPI0_SO	QSPI1_SO	-	-	SPI3_MOSI	UART4_TX	UART5_TX	-	-	KEY0[8]	USART0_CTS	TCIN[8]	-	-	-
PG05	-	-	-	-	-	-	UART4_RX	UART5_RX	-	-	KEY0[9]	TC0_P[1]	TCIN[9]	-	-	-
PG06	RTCOUT	QSPI0_SI	QSPI1_SI	-	-	SPI3_MISO	-	-	RTC1S	-	KEY0[10]	-	-	-	-	-
PG07	-	QSPI0_WPN	QSPI1_WPN	-	UART2_RX	UART3_RX	UART4_RX	UART5_RX	CAN_RX	-	-	TC0_N[0]	TCIN[10]	-	-	-
PG08	-	QSPI0_HOLDN	QSPI1_HOLDN	-	UART2_TX	UART3_TX	UART4_TX	UART5_TX	CAN_TX	-	-	TC0_P[0]	TCIN[11]	-	-	-
PG09	-	-	-	-	-	-	UART2_TX	UART3_TX	LPUART_TX	-	KEY1[15]	TC1_P[1]	TCIN[7]	-	-	-
PG10	RTCOUT	-	SPI0_SCLK	SPI1_SCLK	SPI2_SCLK	SPI3_SCLK	UART6_RX	UART7_RX	RTC1S	-	-	-	-	IOCNT0	-	-
PG11	RTCOUT	-	SPI0_SCSN	SPI1_SCSN	SPI2_SCSN	SPI3_SCSN	UART6_TX	UART7_TX	RTC1S	-	-	-	-	IOCNT1	-	-
PG12	IOCNT2	-	-	-	-	-	UART0_RX	UART1_RX	LPUART_RX	-	KEY0[12]	-	-	CF1	-	-
PG13	-	-	-	-	-	-	UART0_TX	UART1_TX	LPUART_TX	-	KEY0[13]	-	TCIN[11]	CF2	-	-
PG14	-	-	SPI0_MISO	SPI1_MISO	SPI2_MISO	SPI3_MISO	UART2_RX	UART3_RX	LPUART_RX	-	-	-	TC1_N[1]	-	-	-
PG15	-	-	SPI0_MOSI	SPI1_MOSI	SPI2_MOSI	SPI3_MOSI	UART2_TX	UART3_TX	LPUART_TX	-	-	-	TC1_P[1]	-	-	-

Appendix Table 8 Port H Alternate Function List

Port H Alternate Function																
Port	AF00	AF01	AF02	AF03	AF04	AF05	AF06	AF07	AF08	AF09	AF10	AF11	AF12	AF13	AF14	AF15
	SYS	QSPI0	SPI0 QSPI1	SPI1	SPI2 UART0/2	SPI3 UART1/3	UART0-7	UART0-7	LPUART CAN RTC1S	I2C0-1 ISO78160-1	KEY0-1	USART0-1 TCO	TCI TCO	CF0-2 IOCNT0-4	Reserve	LCD
PH00	-	-	-	-	-	-	UART6_RX	UART7_RX	-	-	KEY0[3]	TC0_P[1]	TCIN[3]	CF1	-	-
PH01	-	-	-	-	-	-	UART6_TX	UART7_TX	-	-	KEY0[11]	TC0_N[1]	TCIN[10]	CF0	-	-

Appendix Table 9 Port L Alternate Function List

Port L Alternate Function																
Port	AF00	AF01	AF02	AF03	AF04	AF05	AF06	AF07	AF08	AF09	AF10	AF11	AF12	AF13	AF14	AF15
	SYS	QSPI0	SPI0 QSPI1	SPI1	SPI2 UART0/2	SPI3 UART1/3	UART0-7	UART0-7	LPUART CAN RTC1S	I2C0-1 ISO78160-1	KEY0-1	USART0-1 TCO	TCI TCO	CF0-2 IOCNT0-4	Reserve	LCD
PL00	RTCOUT	-	-	-	-	-	-	-	RTC1S	-	-	TC1_N[1]	TCIN[6]	-	-	-
PL01	-	-	-	-	-	-	-	-	-	-	KEY0[6]	-	-	-	-	-
PL02	-	-	-	-	-	-	-	-	-	-	KEY0[7]	-	-	-	-	-
PL03	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-

27.7 Software Configuration Process and Application Notes

27.7.1 Configure and Enable the GPIO Module Clock According to the Content in the System Control Chapter

- The GPIO clock can be enabled through the core0 system control register PMOD0_CKE[11];
- The GPIO clock can also be enabled through the core1 system control register PMOD0_CKE[11].

27.7.2 GPIO Configuration When the I/O is Used for JTAG Function

- Write 0x55DD77EE, 0x66883399, 0xDDAABB11 to the GPIO_DBG_KEY register in turn;
- Write the GPIO_DBG register to configure the JTAG function enable or disable of the corresponding pin;

27.7.3 GPIO Configuration When the I/O is Used for General GPIO Function

- If the pin is used as output, the GPIOx_OTYPE register can be configured to select the output type as normal or open-drain output, and the GPIOx_OSPEED register can be configured to select the driving capability of the pin as fast or slow;
- If the pin is used as input, the GPIOx_ITYPE register can be configured to select the input type as CMOS or TTL;
- When the I/O is used as output, the output data register GPIOx_ODATA can be operated bitwise by configuring the GPIOx_SET, GPIOx_CLR, and GPIOx_TG registers;
- If the I/O is used for digital alternate function, set GPIOx_MODE to alternate mode, and configure the pin to reset to which digital alternate function by configuring GPIOx_AFSEL0 and GPIOx_AFSEL1;
- If the I/O has an analog function, the GPIOx_AEN can be configured to enable the analog function of the corresponding bit pin. The analog function of GPIOL is only controlled by the VBAT domain register. For the function configuration of the analog module, refer to the analog chapter;
- If the pin needs pull-up or pull-down, it can be implemented by configuring the GPIOx_UPDN register;
- Configure the GPIOx_MODE register to select the working mode of the pin, including input, output, alternate, and analog functions;
- By enabling the GPIOx_FIE register, the input detection function of the pin can be forced to be enabled. Even if the pin is configured as output mode, after enabling the FIE of the corresponding bit, the current level value on the pin can still be obtained through GPIOx_IDATA.

28 LCD

28.1 Introduction

The LCD has the following characteristics:

- MKC40AT: 434, 632, 830;
- Supports two driving waveforms: Type A and Type B;
- Supports 1/3 bias and 1/4 bias;
- Supports static, 1/2, 1/3, 1/4, 1/6, and 1/8 duty;
- Supports LCD driving mode with 16-level contrast adjustment;
- After the LCD module is turned off, the configured COM and SEG pins are automatically pulled down to ground (Note: Keep the COM and SEG states unchanged when the display is stopped. Do not switch the pin multiplexing relationship to IO ports, nor change the bias and duty configurations).
- MKC40AT support charge pump and internal resistor string voltage division to achieve LCD Bias voltage.

28.1.1 Scan Clock Frequency

The clock of the LCD unit comes from fosc or RCL (selected by the OSC_CTL2 register of the system control) with a frequency of 32768Hz. Fosc or RCL is divided to serve as the LCD waveform scan frequency flcd. The waveform scan frequency flcd can be configured through the register LCD_CLKDIV[7:0].

Fframe = flcd/(number of COM). Generally, the refresh frequency of the LCD screen is required to be slightly higher than 60Hz.

Table 28-1 shows the clock settings corresponding to different duties. (The green-marked ones are the normally used frame rates)

LCD_CLKDIV	Scan Frequency	Static Duty	1/2 Duty	1/3 Duty	1/4 Duty	1/6 Duty	1/8 Duty
0xff	64Hz	64Hz	32Hz	21.3Hz	16Hz	10.7Hz	8Hz
0x7f	128Hz	128Hz	64Hz	42.7Hz	32Hz	21.3Hz	16Hz
0x54	192.8Hz	192.8Hz	96.4Hz	64.3Hz	48.2Hz	32.1Hz	24.0Hz
0x3f	256Hz	256Hz	128Hz	85.3Hz	64Hz	42.7Hz	32Hz
0x2a	381.3Hz	381.3Hz	190.5Hz	127.0Hz	95.3Hz	63.5Hz	47.6Hz
0x1f	512Hz	512Hz	256Hz	170.7Hz	128Hz	85.3Hz	64Hz

28.1.2 Blink Modes

The LCD supports two blink modes: Internal Blink and External Blink. Both modes can be enabled

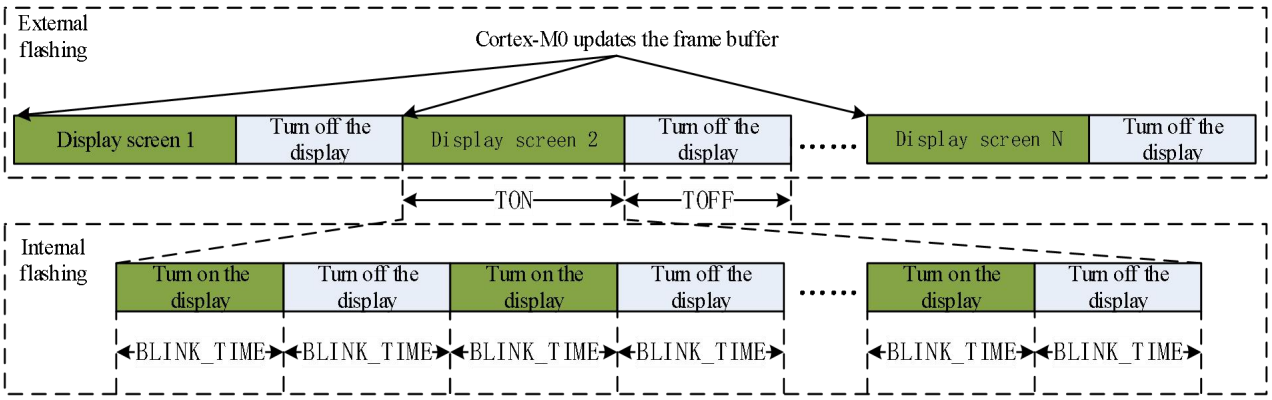


Figure 28-1 LCD Blink Modes

As shown in Figure 27-1, after the LCD is enabled, it will turn on the display for the time length defined by the TON bit field of the LCD_BLINK register, and then turn off the display for the time length defined by the TON bit field of the LCD_BLINK register.

The LCD can issue interrupt requests when the display is turned on and off. Users can use these events to update the frame buffer.

28.1.2.1 Internal Blink Mode

The LCD supports inserting a blink mode during the display period specified by the TON bit field of the LCD_BLINK register. The blink interval is given by the BLINK_TIME bit field of the LCD_BLINK register. When BLINK_TIME is 0, the internal blink mode is disabled; when BLINK_TIME is not 0, TON must be an even multiple of BLINK_TIME.

28.1.2.2 External Blink Mode

When TOFF in the LCD_BLINK register is not 0, the blink function is enabled. After Blink Mode is enabled, the blink frequency is determined by the values of TON and TOFF in the LCD_BLINK register.

28.1.3 LCD Driving Waveforms

The LCD driving waveform is related to the LCD Type, Duty number, and Bias number set by software (LCD_CTL register).

The MCU supports two driving waveforms: Type A and Type B. Type A driving is a row inversion method, that is, completing an alternation of positive and negative driving within the time of 1 frame; Type B driving is a frame inversion method, that is, requiring 2 frames to complete an alternation of positive and negative driving. For details, refer to the LCD driving waveforms described in this chapter. When the driving Duty number is high, the Type B driving mode will have a better display effect.

Users need to select the duty cycle of the LCD output waveform according to the number of COMs required by the application:

- 1 COM: Select static duty cycle, only use COM0;
- 2 COMs: Select 1/2 duty cycle, use COM0 and COM1;
- 3 COMs: Select 1/3 duty cycle, use COM0 ~ COM2;
- 4 COMs: Select 1/4 duty cycle, use COM0 ~ COM3;
- 6 COMs: Select 1/6 duty cycle, use COM0 ~ COM5;
- 8 COMs: Select 1/8 duty cycle, use COM0 ~ COM7;

28.1.3.1 LCD Type A Driving Waveforms

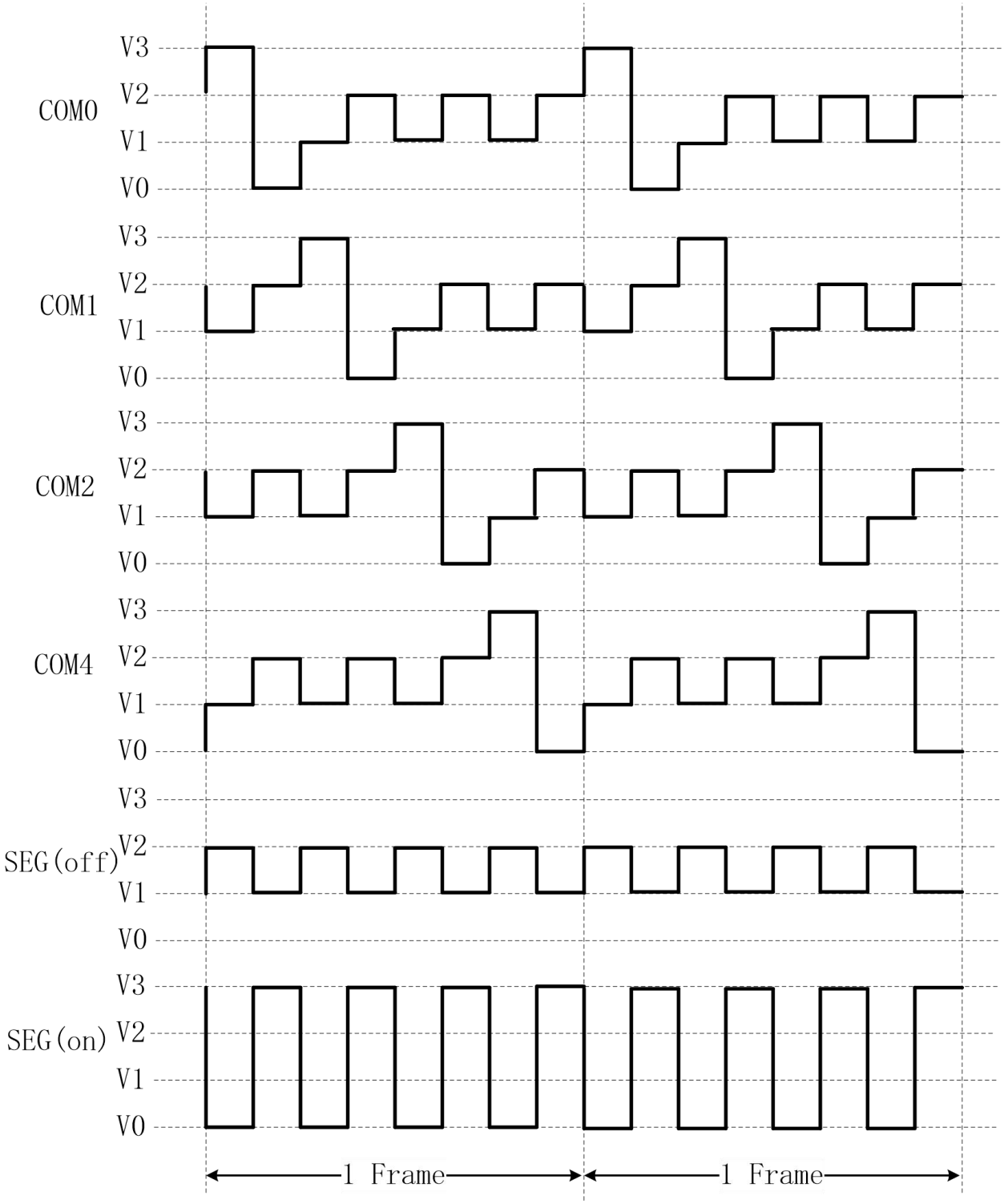


Figure 28-2 LCD Driving Waveform (1/4 Duty, 1/3 Bias, Type A)

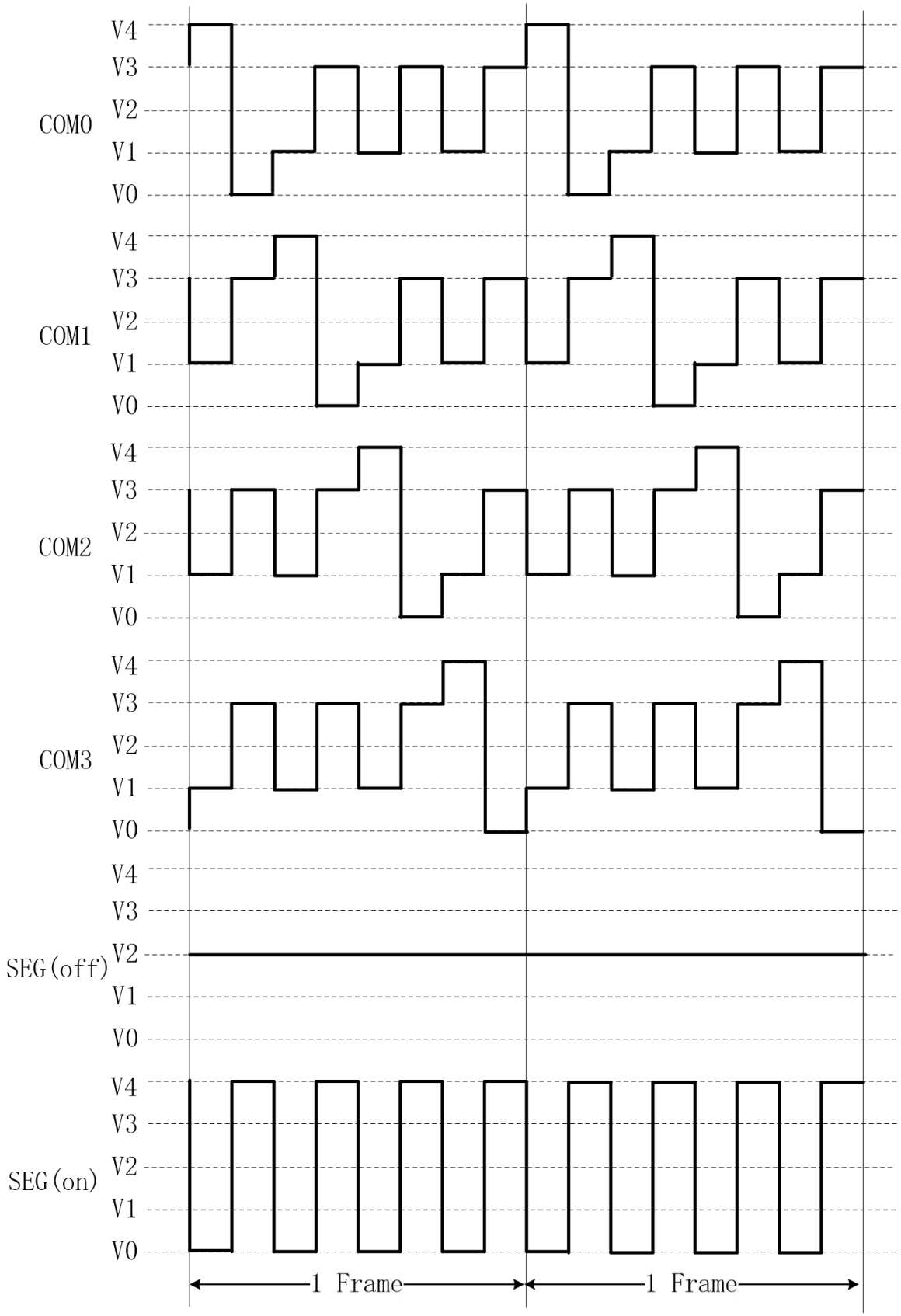


Figure 28-3 LCD Driving Waveform (1/4 Duty, 1/4 Bias, Type A)

28.1.3.2 Type B Driving Waveforms

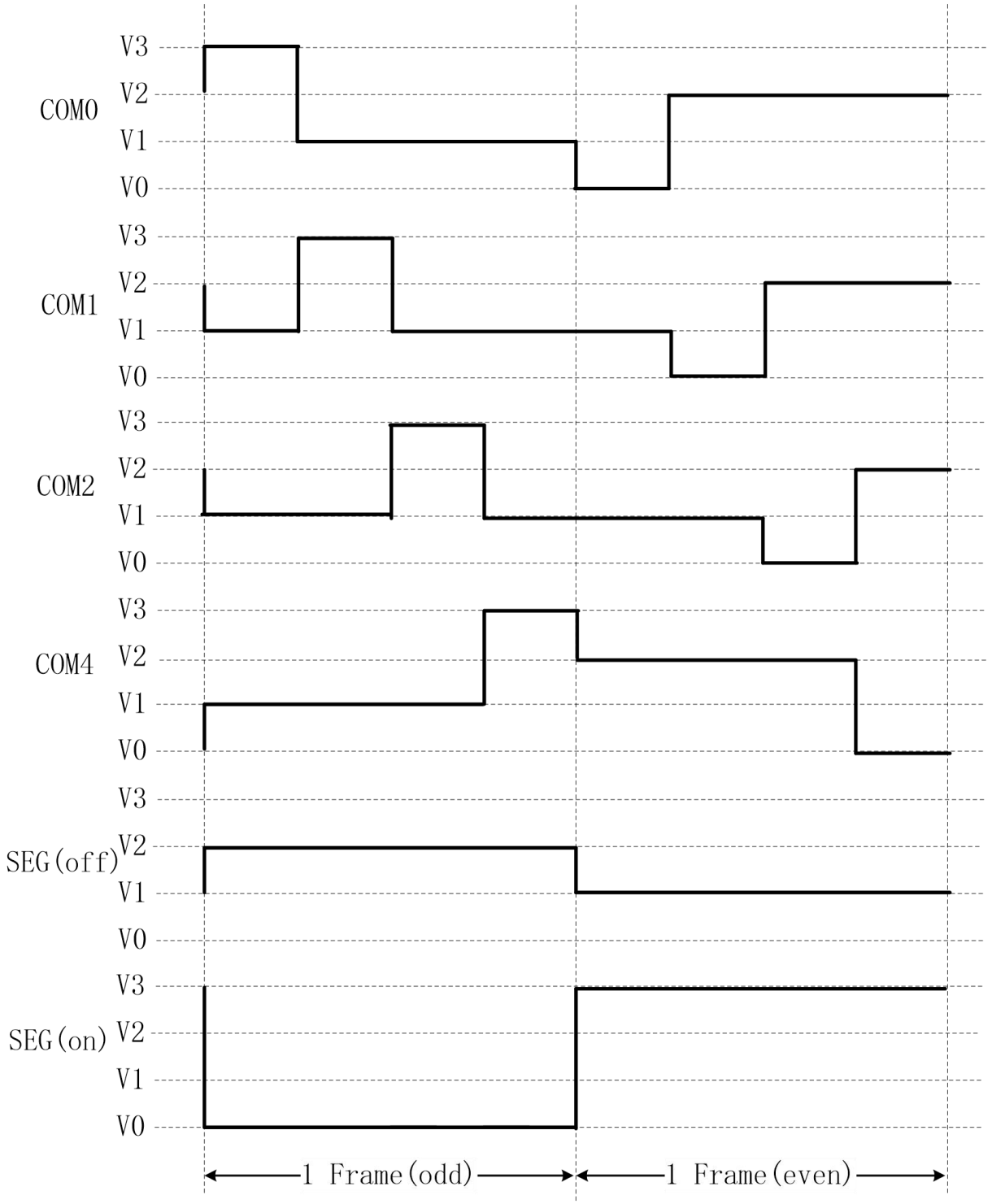


Figure 28-4 LCD Driving Waveform （1/4 Duty，1/3 Bias，Type B）

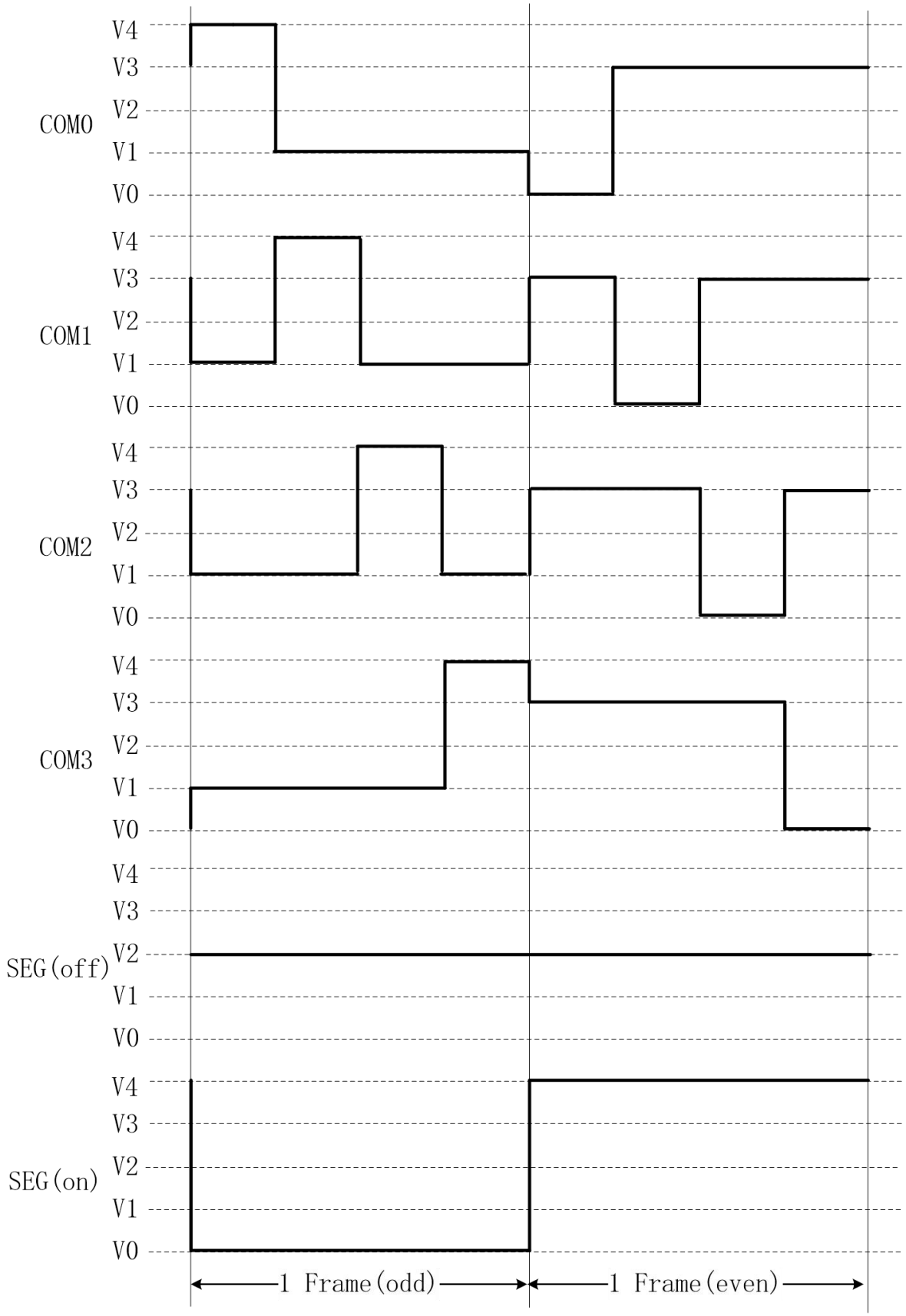


Figure 28-5 LCD Driving Waveform （1/4 Duty，1/4 Bias，Type B）

28.1.4 Bias Voltage

28.1.4.1 Providing LCD Bias Voltage via Charge Pump

The bias voltage of the LCD can be provided by the Charge Pump method. The Charge Pump needs to generate 4 voltages (Va, Vb, Vc, Vd) to meet the application of 1/4 bias ratio. For different bias ratio settings, the voltage modes output by the Charge Pump are different, as shown in Table 28-2.

Table 28-2 Relationship Between LCD Driving Voltage and Bias Ratio

Bias Ratio	Gray Scale Selection	Va	Vb	Vc	Vd	Vd （MAX）
1/3 Bias Ratio	BIASLVL[5] = 0	Va = Vref*(32+BIASLVL[4:0])/63	Vb = Va	Vc = 2*Va	Vd = 3*Va	3.75 V
	BIASLVL[5] = 1	Va = Vref*(1 + BIASLVL[4:0]/63)	Vb = Va	Vc = 2*Va	Vd = 3*Va	5.59 V
1/4 Bias Ratio	BIASLVL[5] = 0	Va = Vref*(32+BIASLVL[4:0])/63	Vb = 2*Va	Vc = 3*Va	Vd = 4*Va	5.0 V
	BIASLVL[5] = 1	Va = Vref*(1 + BIASLVL[4:0]/63)	Vb = 2*Va	Vc = 3*Va	Vd = 4*Va	6.032V

Example: The maximum Vd required by the LCD is 5.2V. When the 1/4 bias ratio is selected, if BIASLVL[5:0] is set greater than 6'h2d, the LCD controller will automatically clamp BIASLVL[5:0] to 6'h2d.

The LCD selects the internal Vref as the reference, and the typical value of the Vref output is 1.25V.

The voltage selection for 1/3 and 1/4 bias ratio applications is shown in Figure 28-6.

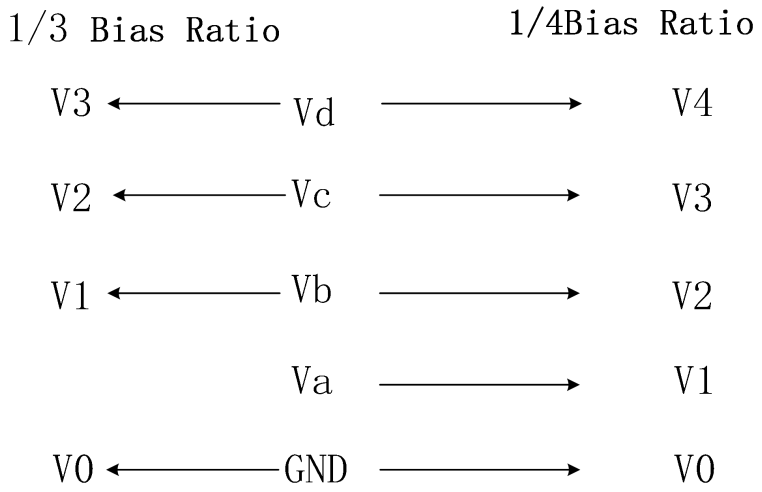


Figure 28-6 Bias Voltage Selection

28.1.4.2 Providing LCD Bias Voltage via Internal Resistor String Voltage Division

Built-in LDO with adjustable output from 2.7~3.6V and a step size of 60mV, supporting 3.0V and 3.3V LCD screens; refer to the LCD Internal Resistor String Control Register (LCD_RESCTL) for specific configuration.

The voltage selection for 1/3 and 1/4 bias ratio applications is shown in Figure 28-6.

The LCD internal resistor string mode supports two modes: large/small resistor string time-sharing enable mode and small resistor/open circuit time-sharing enable mode. It is recommended to use the small resistor/open circuit mode, which requires an external 470nf capacitor for filtering;

Fast charging only works when the screen is first started and has no impact on normal display. The purpose of fast charging is to allow the external capacitor to be fully charged as soon as possible. It is recommended to enable this bit by default.

28.1.5 LCD BUFFER Mapping

The mapping relationship between the LCD_BUFx register and LCD screens of different segment specifications is as follows.

i. When using 8COMs, 36 LCD_BUFs are required, which can support up to an 8*36 LCD screen

LCD_BUF[i] i=0~35 Max SEG:36	SEG[i+4] COM7	SEG[i+4] COM6	SEG[i+4] COM5	SEG[i+4] COM4	SEG[i+4] COM3	SEG[i+4] COM2	SEG[i+4] COM1	SEG[i+4] COM0

ii. When using 6COMs, 38 LCD_BUFs are required, which can support up to a 6*38 LCD screen

LCD_BUF[i] i=0~37 Max SEG:38	-	-	SEG[i+2] COM5	SEG[i+2] COM4	SEG[i+2] COM3	SEG[i+2] COM2	SEG[i+2] COM1	SEG[i+2] COM0

iii. When using 4COM/3COM/2COM/1COM, 20 LCD_BUFs are required, which can support up to a 4*40 LCD screen

LCD_BUF[i] i=0~19 Max SEG:40	SEG[2*i+1] COM3	SEG[2*i+1] COM2	SEG[2*i+1] COM1	SEG[2*i+1] COM0	SEG[2*i] COM3	SEG[2*i] COM2	SEG[2*i] COM1	SEG[2*i] COM0

28.2 Register Description

Base Addresses of LCD Module

Module Name	Core0 Base Address	Core1 Base Address
LCD	0x2006_0000	0xA006_0000

Offset Addresses of LCD Module Registers

Register Name	Address Offset	Description
LCD_CTL	0x0	LCD Control Register
LCD_STATUS	0x4	LCD Status Register
LCD_CLKDIV	0x8	LCD Clock Control Register
LCD_BLINK	0xc	LCD Blink Control Register
LCD_PS	0x10	LCD Fast Charging Time Register
LCD_RESCTL	0x14	LCD Internal Resistor String Control Register
LCD_BUF[i]	0x20+i*1 (i=0-37)	LCD Data Registers (38 8-bit registers in total)

28.2.1 LCD_CTL(0x00)

LCD Control Register

Offset Address: 0x00

Bits	Name	Description	R/W	Reset
31:13	Reserved	Reserved	R	0
16:13	SEG3/COM7 ~	SEG3/COM7~SEG0/COM4 define port multiplexing configuration:	RW	0x0

	SEG0/COM4	0: Select as SEG 1: Select as COM		
12	PWD_PUMP	LCD PUMP Switch: 0: Turn on PUMP, LCD voltage generated by internal PUMP. 1: Turn off PUMP, use resistor string voltage division scheme.	RW	0
11	TYPE	LCD Drive Type Select 0: Type A 1: Type B	RW	0
10:5	BIASLVL	LCD Bias Voltage Adjustment Controls Charge Pump output different amplitude voltages to control LCD contrast.	RW	0
4	BIAS	LCD Bias Control 0: 1/3 Bias 1: 1/4 Bias	RW	0
3:1	DUTY	LCD Duty Cycle Control 000: Static Output (COM0) 001: 1/2 Duty (COM0~1) 010: 1/3 Duty (COM0~2) 011: 1/4 Duty (COM0~3) 100: 1/6 Duty (COM0~5) 101: 1/8 Duty (COM0~7) Other: Reserved	RW	0
0	EN	LCD Module Enable 0: LCD Module Disabled 1: LCD Module Enabled Note: LCD.EN=1 about 250ms later, LCD_BUSY=1; and LCD_BUSY will remain 1; When LCD_BUSY is 1, LCD_CTRL (except EN Bit), LCD_CLKDIV, LCD_BLINK, LCD_PS registers cannot be modified; After LCD.EN=0, the duration of LCD_BUSY is at most 32 LCD refresh cycles, e.g., if LCD_CLKDIV configured as 64Hz, max LCD_BUSY duration is 32*(1/64)=0.5s; Therefore, in low-power handling flow, after LCD.EN=0, wait at least 0.5s before turning off LCD APB clock; Practical applications may not turn off LCD APB clock. 4. In low-power handling flow, before turning off LCD module, relevant analog circuits must be turned off first to ensure low power requirements.	RW	0

28.2.2 LCD_STATUS(0x04)

LCD Status Register

Offset Address: 0x04

Bits	Name	Description	R/W	Reset
31:7	Reserved	Reserved	R	0
6	LCD_BUSY	LCD Busy Bit 0: Not Busy 1: Busy Note: LCD.EN=1 about 250ms later, LCD_BUSY=1; and LCD_BUSY will remain 1; When LCD_BUSY is 1, LCD_CTRL (except EN Bit), LCD_CLKDIV, LCD_BLINK, LCD_PS registers cannot be modified; 3. After LCD.EN=0, the duration of LCD_BUSY is at most 32 LCD refresh cycles, e.g., if LCD_CLKDIV configured as 64Hz, max LCD_BUSY duration is 32*(1/64)=0.5s; Therefore, in low-power handling flow, after LCD.EN=0, wait at least 0.5s before turning off LCD APB clock; Practical applications may not turn off LCD APB clock.	R	0
5	Reserved	Reserved register bit, user should not operate.		
4	Reserved	Reserved register bit, user should not operate.		
3	IRQOFFEN	Display Off IRQ Enable Bit 0: Disabled 1: Enabled	RW	0
2	IRQONEN	Display On IRQ Enable Bit 0: Disabled 1: Enabled	RW	0
1	DOFF	Display Off Pending Bit 0: No interrupt event 1: Set when display changes from bright to dark Note: Write 1 to clear	RW	0
0	DON	Display On Pending Bit 0: No interrupt event 1: Set when display changes from dark to bright Note: Write 1 to clear	RW	0

28.2.3 LCD_CLKDIV(0x08)

LCD Clock Control Register

Offset Address: 0x08

Bits	Name	Description	R/W	Reset
31:8	Reserved	Reserved	R	0

7:0	CLKDIV	LCD Clock Divider Coefficient LCD_CLK=fosc/(2*(CLKDIV+1)) (fosc is 32768Hz)	RW	0

28.2.4 LCD_BLINK(0x0c)

LCD Blink Control Register

Offset Address: 0x0c

Bits	Name	Description	R/W	Reset
31:24	Reserved	Reserved	R	0
25:18	BLINK_TIME	Step size is 0.25s, supports 0~63.75s Sets the bright and dark time within the TON display period = 0.25 * BLINK_TIME. Note: When set to 0, represents constant bright within TON display period, no blinking. When set value > 0, TON must be an even multiple of BLINK_TIME (n is an integer > 0).	RW	0
17:9	TOFF	Step size is 0.25s, supports 0~127.5s, please set >3s 0.25 * TOFF	RW	0
8:0	TON	Step size is 0.25s, supports 0~127.5s, please set >3s 0.25 * TON	RW	0

28.2.5 LCD_PS(0x10)

LCD PUMP Setup Time Register

Offset=0x10

Bits	Name	Description	R/W	Reset
31:16	---	Reserved	R	0
15:0	PS	LCD PUMP Setup time Time = Tosc * (PS+4) (Tosc is 30.5uS)	RW	0xccc

28.2.6 LCD_RESCTL(0x14)

LCD Internal Resistor String Control Register

Offset Address 0x14

Bits	Name	Description	R/W	Reset
31:13	Reserved	Reserved	R	0
15:13	OFF_LVL	Controls pin output signal value after LCD is turned off. 0: 0 (LCDV0) 1: 1 (LCDVA) 2: 2 (LCDVB) 3: 3 (LCDVC) 4: 4 (LCDVD) 5: 7 (HZ) Other: 0 (LCDV0)	RW	0x0

<table>
<tr><td>12:9</td><td>LDOS</td><td>LDO Output Level Selection Signal
LDO output serves as LCDVD voltage, and generates LCDVC, LCDVB, LCDVA through resistor division;
LDO output level from 2.7V~3.6V, total 16 levels, step 0.06V;
<table>
<tr><th>LDOS Configuration</th><th>LDO Output Voltage V</th></tr>
<tr><td>0000</td><td>2.7</td></tr>
<tr><td>0001</td><td>2.76</td></tr>
<tr><td>0010</td><td>2.82</td></tr>
<tr><td>0011</td><td>2.88</td></tr>
<tr><td>0100</td><td>2.94</td></tr>
<tr><td>0101</td><td>3</td></tr>
<tr><td>0110</td><td>3.06</td></tr>
<tr><td>0111</td><td>3.12</td></tr>
<tr><td>1000</td><td>3.18</td></tr>
<tr><td>1001</td><td>3.24</td></tr>
<tr><td>1010</td><td>3.3</td></tr>
<tr><td>1011</td><td>3.36</td></tr>
<tr><td>1100</td><td>3.42</td></tr>
<tr><td>1101</td><td>3.48</td></tr>
<tr><td>1110</td><td>3.54</td></tr>
<tr><td>1111</td><td>3.6</td></tr>
</table>
When Reset, the LDO outputs 3.0V</td><td>RW</td><td>0101</td></tr>
<tr><td>8</td><td>FCC</td><td>Fast Charge Control:
0: Disable fast charge function
1: Enable fast charge function;
Select internal resistor division, node voltage externally connected with 470nf capacitor, when LCD module turns on, internal resistor adjusted to 5k, and maintained for 100ms, to complete fast charging of the capacitor.</td><td>RW</td><td>1</td></tr>
<tr><td>7</td><td>RES_AO</td><td>Resistor String Time-Division Drive Switch
0: Resistor string time-division drive, generate drive signal according to RES_DT and RES_FT
1: Resistor string always driven, RES_DT and RES_FT configuration invalid.
When selecting TYPE A, RES_AO is always 1.</td><td>RW</td><td>1</td></tr>
<tr><td>6:4</td><td>RES_DT</td><td>Resistor String Voltage Division Mode, within screen refresh cycle, 20k resistor drive time configuration
Drive time Td=(BIAS_DT[2:0]+1)* Tosc (Tosc is 30.5uS) .</td><td>RW</td><td>000</td></tr>
<tr><td>3:2</td><td>RES_FT</td><td>Resistor String Voltage Division Mode, within screen</td><td>RW</td><td>00</td></tr>
</table>

		refresh cycle, 20k resistor drive count configuration 00: Drive 1 time 01: Drive 2 times 10: Drive 3 times 11: Drive 4 times		
1	RSM	Internal Resistor Division Resistor Mode Selection Signal 0 Small resistor + open circuit, 20k resistor time-division enabled, 200k resistor always shorted 1 Large/Small resistor switching, 20k and 200k resistors time-division switched	RW	0
0	IRSN	Resistor Division Mode Selection 0 Internal resistor division mode 1 Reserved If using resistor division mode, this register can only be configured as 0, configuring as 1 is meaningless.	RW	1

If selecting resistor division mode, suggested configuration for LCD_RESCTL as follows:

Typical 3V screen, LCD->RES_CTRL register configuration value: 0xb14;

Typical 3.3V screen, LCD->RES_CTRL register configuration value: 0x1514.

28.2.7 LCD_BUF[i](8-bit register)

Offset 0x20– 0x45

Bits	Name	Description	R/W	Reset
31:8	Reserved	Reserved	R	0
7:0	LCD_BUFx	LCD Screen SEG Display Data, physical meaning of each bit as follows: 0: Corresponding display unit not lit 1: Corresponding display unit lit	RW	0

28.3 LCD Operation Steps

- Initialize GPIO

a. Enable register write access, and configure SYS_PSW1 (the public password register of the Core0 system control module) to 0x82;

b. Enable the GPIO module clock gating, and configure bit 11 of PMOD0_CKE (the clock control register of the Core0 system control module) to 1;

c. Based on the required SEG/COM pins, configure the output mode of the corresponding PIN to alternate function mode (0x3). Refer to Section 27.5.1 GPIOx_MODE in the GPIO chapter;

d. Based on the required SEG/COM pins, configure the alternate function of the corresponding PIN to LCD function (AF15). Refer to Section 27.5.11 GPIOx_AFSEL0 and Section 27.5.12 GPIOx_AFSEL1 in the GPIO chapter. For the alternate function configuration of LCD function PINs, refer to the Alternate Function List in Section 27.6 of the GPIO chapter.

- Initialize LCD

a. Enable the LCD module clock gating, and configure bit 21 of MOD0_CKE (the clock control register) to 1;

b. Configure bit 0 of LCD_CTL to 0 to disable the LCD module. Note: Except for bit 0 of LCD_CTL, the

configuration of registers LCD_CTL, LCD_CLKDIV, LCD_BLINK, and LCD_PS all requires disabling the LCD first;

c. Configure the LCD clock control register LCD_CLKDIV; refer to Section 28.1.1 Scan Clock Frequency for the configuration of this register;

d. Configure the LCD blink control register LCD_BLINK; if the blink function is not required, it can be directly configured to 0x01;

e. The LCD Bias voltage can be implemented via the PUMP scheme or the resistor string voltage division scheme. When using the PUMP scheme, only LCD_CTL needs to be configured; when selecting the resistor string voltage division scheme, both LCD_CTL and LCD_RESCTL need to be configured simultaneously. For example, for an 8-COM LCD screen with a display voltage of 5V, if the PUMP driving method is selected, LCD_CTL can be optionally configured to 0x3FB;

f. Configure bit 0 of LCD_CTL to 1 to enable the LCD module. Approximately 250ms after the LCD module is enabled, LCD_BUSY will be set to 1; at this point, except for bit 0 of LCD_CTL, the configurations of registers LCD_CTL, LCD_CLKDIV, LCD_BLINK, and LCD_PS will be locked.

- Complete the initialization settings, and send the display data to LCD_BUF[i].
- Power-Down Handling Processa.

It is recommended that the power-down handler disable the LCD immediately after detecting mains power failure, i.e., configure bit 0 of LCD_CTL to 0.b.

It is recommended not to disable the LCD's APB clock in the power-down process. If it is necessary to disable the APB clock, a certain period of time should be waited after disabling the LCD module. This is because after the LCD module is disabled, LCD_BUSY will remain active for up to 32 LCD refresh cycles. Taking the LCD_CLKDIV configured to 64Hz as an example, the maximum duration of LCD_BUSY is 32*(1/64) = 0.5s; thus, the reserved waiting time should be at least 0.5s, and it is recommended to extend the waiting time to 1s.

29 KBI&INTC

29.1 Introduction

The KBI&INTC module is a key detection and external interrupt module. By setting the length of the filtering time, key detection and external interrupt triggering are achieved respectively. The chip integrates two KBI&INTCs, each with 16 channels, supporting independent and shared interrupts for 16 channels. The channel filtering time, security attributes, and channel enable are independently configurable.

29.2 Features

- The system has 2 KBI&INTCs in total, and 1 KBI&INTC supports 16-channel detection.
- The security attributes of KBI&INTC are configurable.
- Each channel of KBI&INTC supports the following functions:
- Independent channel enable.
- Filter enable and filter time can be independently configured.
- Three edge detection modes: rising edge, falling edge, and both edges.
- Can generate flag bits.
- Supports output of 16-channel independent interrupts and shared interrupts.
- Supports GPIO multiplexing input security check.

29.3 Filtering

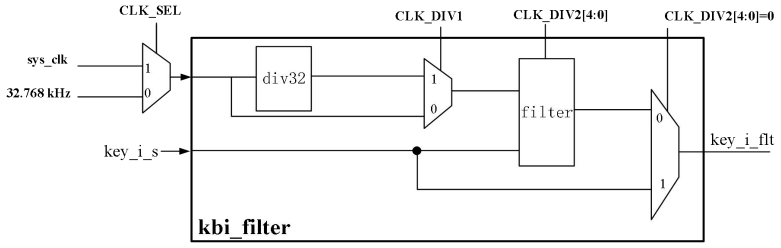


Figure 29-1 Single-Channel Filter Configuration Block Diagram

29.3.1 Filter Time

The filter time is jointly determined by CLK_SEL, CLK_DIV1, and CLK_DIV2[4:0] of the KBIx_FTL_TIME register, with the basic configuration block diagram as above. If the configuration value of CLK_DIV2[4:0] is 0, filter_in is considered unfiltered. In other cases, the filter time can be calculated by the following formula:

$T_{filter} = T_{filter_clk} * CLK_DIV1 * CLK_DIV2$;

Among them, T_{filter_clk} selects the 32.768 kHz clock when CLK_SEL=0, and sys_clk when CLK_SEL=1. CLK_DIV1 is 1x division or 32x division. CLK_DIV can be 1~31. (For specific definitions, refer to the description of the register KBIx_FTL_TIME)

29.4 Registers

Module Name	Secure Base Address	Non-Secure Base Address
KBI0	0x2006_4000	0xA006_4000
KBI1	0x2006_8000	0xA006_8000

29.4.1 Register List

Register Name	Address Offset	Description
KBI_CTL	0x0	Enable control configuration register for KBI channels 15~0
KBI_MODE	0x4	Edge detection mode register for KBI channels 15~0
KBI_MASK	0xC	Interrupt enable register for KBI channels 15~0
KBI_STA	0x10	Status register for KBI channels 15~0
CH0_FLT_TIME	0x14	Filter configuration register for KBI channel 0
CH1_FLT_TIME	0x18	Filter configuration register for KBI channel 1
CH2_FLT_TIME	0x1C	Filter configuration register for KBI channel 2
CH3_FLT_TIME	0x20	Filter configuration register for KBI channel 3
CH4_FLT_TIME	0x24	Filter configuration register for KBI channel 4
CH5_FLT_TIME	0x28	Filter configuration register for KBI channel 5
CH6_FLT_TIME	0x2C	Filter configuration register for KBI channel 6
CH7_FLT_TIME	0x30	Filter configuration register for KBI channel 7
CH8_FLT_TIME	0x34	Filter configuration register for KBI channel 8
CH9_FLT_TIME	0x38	Filter configuration register for KBI channel 9
CH10_FLT_TIME	0x3C	Filter configuration register for KBI channel 10
CH11_FLT_TIME	0x40	Filter configuration register for KBI channel 11
CH12_FLT_TIME	0x44	Filter configuration register for KBI channel 12
CH13_FLT_TIME	0x48	Filter configuration register for KBI channel 13
CH14_FLT_TIME	0x4C	Filter configuration register for KBI channel 14
CH15_FLT_TIME	0x50	Filter configuration register for KBI channel 15

29.4.1.1 KBI_CTL（0x00）

KBI Control Register

Offset Address = 0x00

Bits	Name	Description	R/W	Reset
31:16	---	Reserved	R	0
15:0	Enable	Enable signal, Enable[15:0] corresponds to KBI Channel 15~Channel 0 0: Disable corresponding KBI channel 1: Enable corresponding KBI channel	RW	0

29.4.1.2 KBI_MODE（0x04）

KBI Mode Register

Offset Address = 0x04

Note: When modifying the configuration of the KBI_MODE register, you must first configure the KBI_CTL register to disable all channels before making the configuration. Otherwise, it will result in abnormal configuration of the MODE for the enabled channels.

Bits	Name	Description	R/W	Reset
31:30	MODE15	KBI Channel 15 Interrupt Input Valid Mode Selection 00: Rising Edge 01: Falling Edge 10: Both Edges Others: Reserve	RW	0
29:28	MODE14	KBI Channel 14 Interrupt Input Valid Mode Selection 00: Rising Edge 01: Falling Edge 10: Both Edges Others: Reserve	RW	0
27:26	MODE13	KBI Channel 13 Interrupt Input Valid Mode Selection 00: Rising Edge 01: Falling Edge 10: Both Edges Others: Reserve	RW	0
25:24	MODE12	KBI Channel 12 Interrupt Input Valid Mode Selection 00: Rising Edge 01: Falling Edge 10: Both Edges Others: Reserve	RW	0
23:22	MODE11	KBI Channel 11 Interrupt Input Valid Mode Selection 00: Rising Edge 01: Falling Edge 10: Both Edges Others: Reserve	RW	0

21:20	MODE10	KBI Channel 10 Interrupt Input Valid Mode Selection 00: Rising Edge 01: Falling Edge 10: Both Edges Others: Reserve	RW	0
19:18	MODE9	KBI Channel 9 Interrupt Input Valid Mode Selection 00: Rising Edge 01: Falling Edge 10: Both Edges Others: Reserve	RW	0
17:16	MODE8	KBI Channel 8 Interrupt Input Valid Mode Selection 00: Rising Edge 01: Falling Edge 10: Both Edges Others: Reserve	RW	0
15:14	MODE7	KBI Channel 7 Interrupt Input Valid Mode Selection 00: Rising Edge 01: Falling Edge 10: Both Edges Others: Reserve	RW	0
13:12	MODE6	KBI Channel 6 Interrupt Input Valid Mode Selection 00: Rising Edge 01: Falling Edge 10: Both Edges Others: Reserve	RW	0
11:10	MODE5	KBI Channel 5 Interrupt Input Valid Mode Selection 00: Rising Edge 01: Falling Edge 10: Both Edges Others: Reserve	RW	0
9:8	MODE4	KBI Channel 4 Interrupt Input Valid Mode Selection 00: Rising Edge 01: Falling Edge 10: Both Edges Others: Reserve	RW	0
7:6	MODE3	KBI Channel 3 Interrupt Input Valid Mode Selection 00: Rising Edge 01: Falling Edge 10: Both Edges Others: Reserve	RW	0
5:4	MODE2	KBI Channel 2 Interrupt Input Valid Mode Selection 00: Rising Edge 01: Falling Edge	RW	0

		10: Both Edges Others: Reserve		
3:2	MODE1	KBI Channel 1 Interrupt Input Valid Mode Selection 00: Rising Edge 01: Falling Edge 10: Both Edges Others: Reserve	RW	0
1:0	MODE0	KBI Channel 0 Interrupt Input Valid Mode Selection 00: Rising Edge 01: Falling Edge 10: Both Edges Others: Reserve	RW	0

Note: The KBI mode register can only be configured when the KBI channel is not enabled.

29.4.1.3 KBI_MASK（0x0C）

KBI MASK Register

Offset Address = 0x0C

Bits	Name	Description	R/W	Reset
31:16	---	Reserved	R	0
15:0	MASK	MASK[15:0] corresponds to KBI Channel 15~Channel 0 0: Interrupt Disabled 1: Interrupt Enabled	RW	0

29.4.1.4 KBI_STA（0x10）

KBI Interrupt Flag Register

Offset Address = 0x68

Bits	Name	Description	R/W	Reset
31:16	---	Reserved	R	0
15:0	STA	STA[15:0] corresponds to KBI Channel 15~Channel 0 0: KBI interrupt event has not occurred 1: KBI interrupt event has occurred Note: Write 1 to clear	RW1C	0

29.4.1.5 CHx_FLT_TIME（0x14~0x50）(x=0~15)

KBI Channel x Filter Configuration Register (x=0~15)

Offset Address = 0x14~0x50

Note: When modifying the configuration of the CHx_FLT_TIME register, you must first configure the KBI_CTL register to disable the corresponding channel before making the configuration. Otherwise, it will result in abnormal configuration of the CHx_FLT_TIME register.

Bits	Name	Description	R/W	Reset
31:7	---	Reserved	R	0
6	CLK_SEL	=0: Filter clock is 32.768 kHz clock;	RW	0

		=1: Filter clock is sys_clk;		
5	CLK_DIV1	Filter Clock First Stage Divider =0: No division; =1: 32 division	RW	1
4:0	CLK_DIV2	Filter Clock Second Stage Divider =0: Turn off channel input filter; =N: Filter time is: T_filter = Tfilter_clk * CLK_DIV1 * (N+1);	RW	5'h13

Reference typical filter time configurations:

External interrupt: CLK_SEL=1; CLK_DIV1=0; CLK_DIV2=5'h7 (LCM mode: 5'h4);

Key interrupt: CLK_SEL=0; CLK_DIV=1; CLK_DIV2=5'h13;

Note: The KBI channel x filter configuration register can only be configured when the KBI channel is not enabled.。

29.5 Software Configuration Process and Application Notes

- Enable the register configuration clock gating of KBI (system register KBIx_CKE.KBIC0_CKE Bit);
- Configure the mode, interrupt enable, and filtering (filter enable, filter time) registers of KBI channel x (x=0~15);
- Enable the module clock gating of the KBI channel (system register KBIx_CKE.KBIxx_CKE Bit);
- Configure GPIO multiplexing input to KBI channel x (x=0~15);
- Enable the corresponding KBI channel x (x=0~15);

Note: Before configuring CHx_FLT_TIME.CLK_SEL of the KBI filter channel register to switch the filter clock source, the software must disable the clock gating (system register KBIx_CKE.KBIxx_CKE Bit) and enable (KBI_CTL.Enable[xx]) of the corresponding channel.

Important Notes:

- When modifying the configuration of the KBI_MODE register, you must first configure the KBI_CTL register to disable all channels before proceeding with the configuration. Otherwise, it will result in abnormal configuration of the MODE for the enabled channels. Enable KBI only after configuring the MODE.
- When modifying the configuration of the CHx_FLT_TIME register, you must first configure the KBI_CTL register to disable the corresponding channel before proceeding with the configuration. Otherwise, it will result in abnormal configuration of the CHx_FLT_TIME register.

30 IOCNT

30.1 Introduction

The pulse forwarding function, also known as the IOCNT function, refers to the function of forwarding the pulses input on GPIO or internal pulses from the GPIO port after operations such as division and level inversion. It also has the function of recording the number of input and output pulses.

30.2 Features

- 5 independent pulse output channels, each can be configured with different division ratios and level inversion output functions;
- Supports input pulse counting and output pulse counting;
- Supports forwarding and outputting input pulses after level inversion;
- Supports direct forwarding output of input pulses or forwarding output after 2~2048x division;
- Configurable pulse rising edge counting and falling edge counting;
- Configurable clear after reading and non-clear after reading of pulse count value;
- Supports filter bypass function and flexible configuration of 1~128 cycle filter count;
- Supports configurable input channels as external input or internal software pulses; external channels are configured by the AOS module;
- Supports writing initial values to the input pulse counter;
- Supports software configuration to trigger one input pulse of IOCNTx;
- The output pulse can be configured to not output the first pulse when receiving the first input pulse.

30.3 Block Diagram

The module block diagram is as follows.

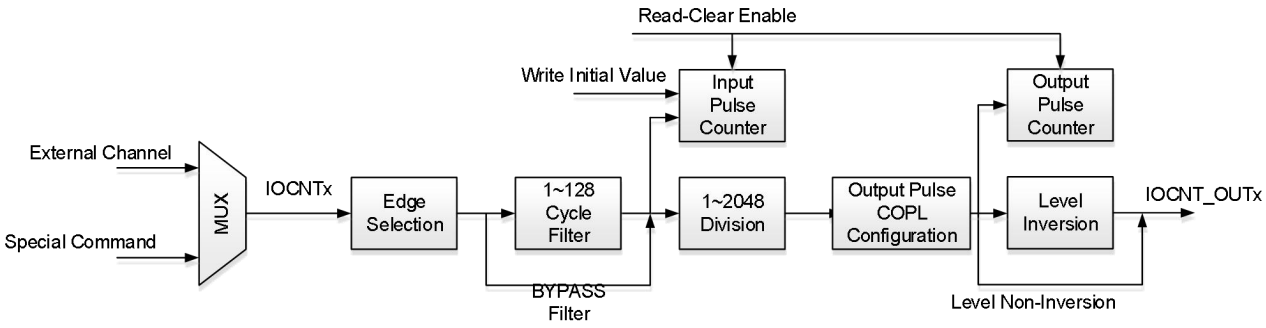


Figure 30-1 Pulse Forwarding Function Block Diagram

30.4 Functional Description

30.4.1 Input Signal Types

Input signals are divided into periodically uniformly distributed and periodically non-uniformly distributed. Periodically uniformly distributed signals are further divided into equal duty cycle and unequal duty cycle, as shown below:

Periodically Uniformly Distributed Pulses

Figure 30-2 Equal Duty Signal

Figure 30-3 Unequal Duty Signal

Periodically Non-Uniformly Distributed Pulses

Figure 30-4 Non-Uniform Period Signal

30.4.2 Pulse Counting

Pulse counting is the edge counting of the input signal io_in. Rising edge counting or falling edge counting can be configured through the control register.

Signal Definitions:

io_in: PAD input;

io_out: Signal between the frequency divider and inverter in the above figure;

30.4.2.1 Rising Edge Counting

When the input signal io_in is low by default, it is recommended to use rising edge counting. In this mode, io_out outputs low by default.

Start Time Point:

Start when io_in is low, counting starts from the first rising edge, as shown in Figure 5;

Start when io_in is high, the count value is incremented by 1, as shown in Figure 6.

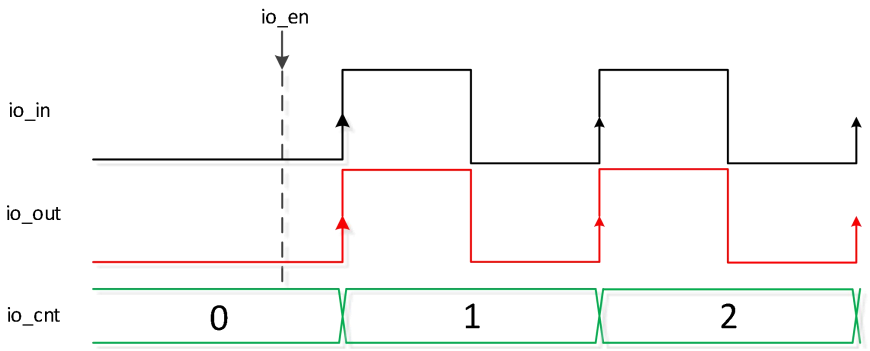


Figure 30-5 Rising Edge Counting Started from Low Level (Pulse Not Divided)

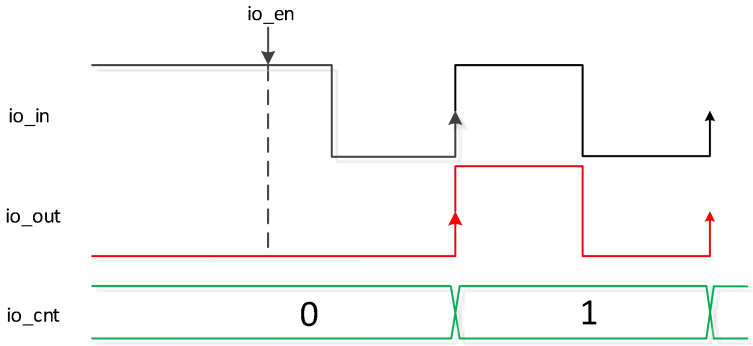


Figure 30-6 Rising Edge Counting Started from High Level (Pulse Not Divided)

30.4.2.2 Falling Edge Counting

When the input signal io_in is high by default, it is recommended to use falling edge counting. In this mode, io_out outputs high by default.

When starting with io_in in high state, the first falling edge is valid and counting starts, as shown in Figure 7 below.When starting with io_in in low state, the count value is incremented by 1, as shown in Figure 8 below.

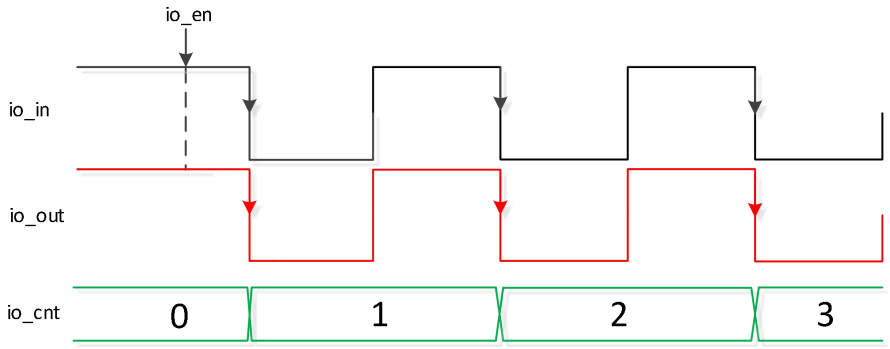


Figure 30-7 Falling Edge Counting Started from High Level (Pulse Not Divided)）

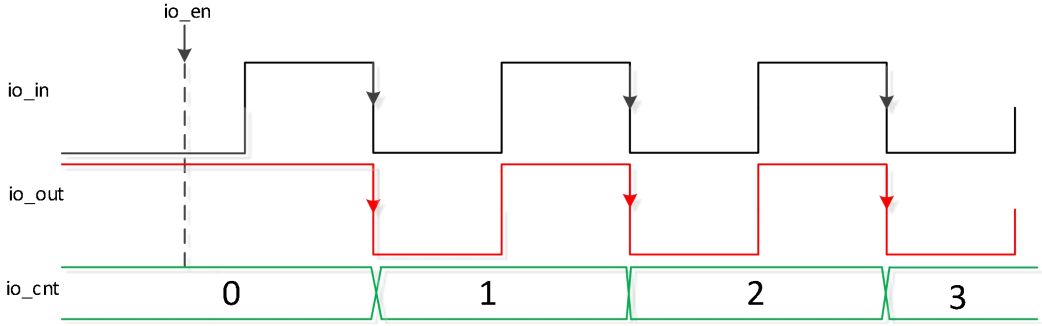


Figure 30-8 Falling Edge Counting Started from Low Level (Pulse Not Divided)

30.4.2.3 Count Clear and Count Overflow

Each time the io_cnt register is read, the historical count value is cleared by default. It can also be configured as an accumulative type that is not cleared after reading through the CNT_CLR bit of the IOCNT_CTRL register.

When the read operation comes before or after the edge change of the input signal, the edge of this time is accumulated to the next io_cnt, as shown in Figure 9 and Figure 10.

。

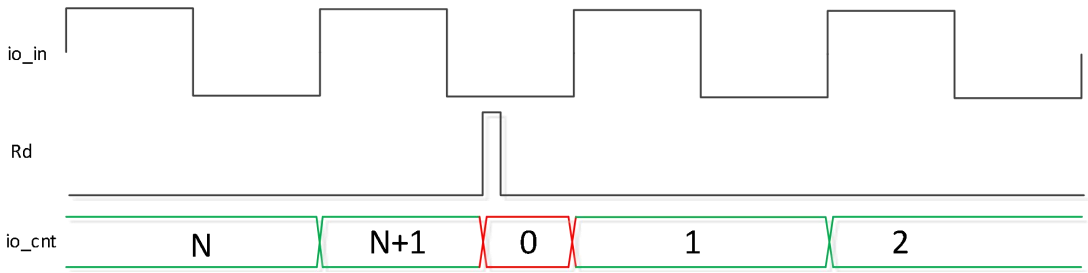


Figure 30-9 Clear Operation Before io Edge (Rising Edge Counting）

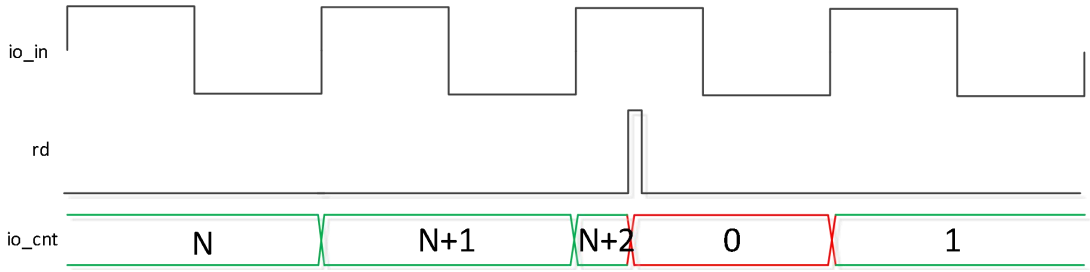


Figure 30-10 Clear Operation After io Edge (Rising Edge Counting)

The read operation occurs simultaneously with the edge of the input signal. The value read this time is the previous count value, and the edge of this time is accumulated to the next io_cnt, as shown in Figure 11.

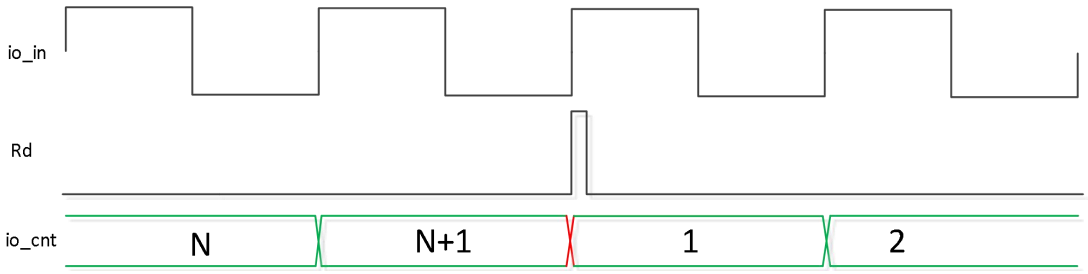


Figure 30-11 Clear Operation Simultaneous with io Edge (Rising Edge Counting)

Count overflow design: When io_cnt = 0xffffff, and another pulse comes, the count overflows, io_cnt = 0, counting starts from the beginning, and io_cnt = io_cnt + 1 for subsequent pulses, as shown in Figure 12.

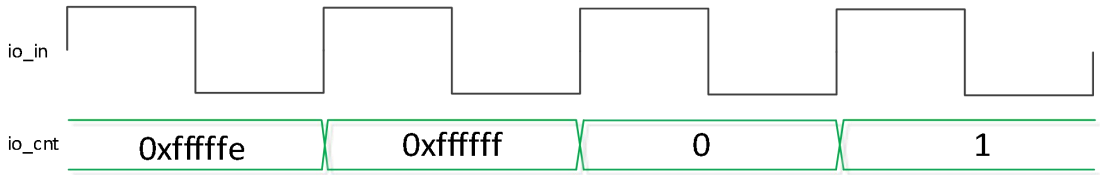


Figure 30-12 io_cnt Overflow (Rising Edge Counting)

Overflow and read occur simultaneously. At this time, 0xffffff is read, and the count value becomes 1. When overflow and read clear occur simultaneously, io_cnt = 1. As shown in Figure 13.

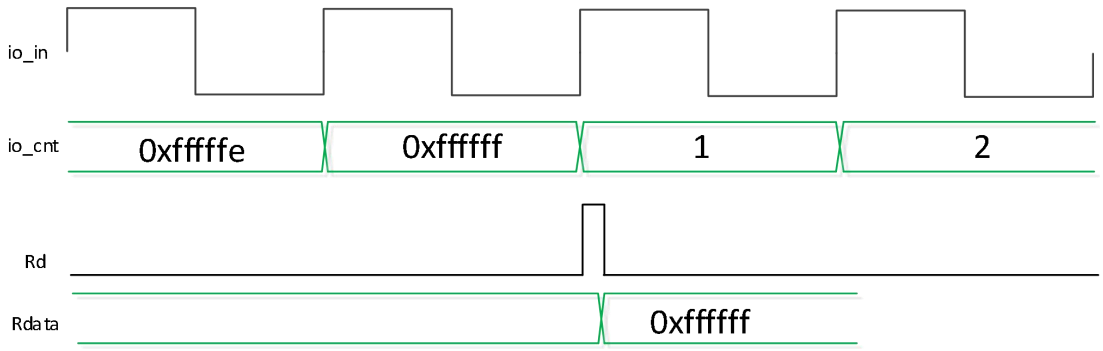


Figure 30-13 Overflow and Read Coming Simultaneously (Rising Edge Counting)

30.4.3 Pulse Division

30.4.3.1 No Division (Pulse Following)

Without division, io_out follows io_in changes without processing.When the high level of io_in exceeds 80ms, io_out does not perform any special processing and follows io_in.

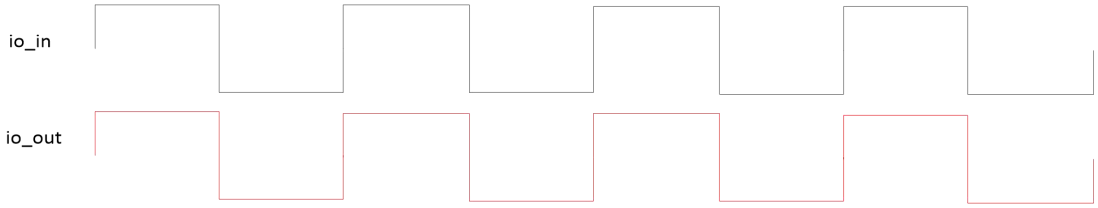


Figure 30-14 Equal Duty Signal

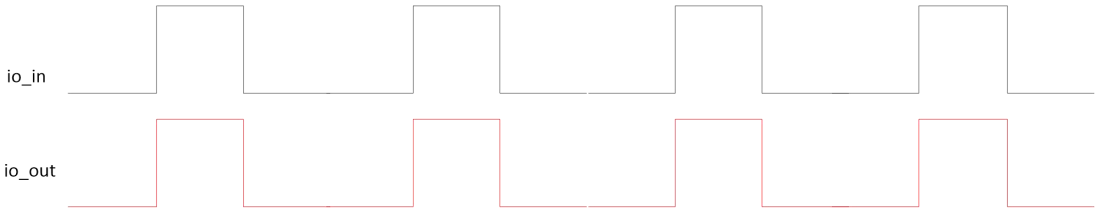


Figure 30-15 Unequal Duty Signal

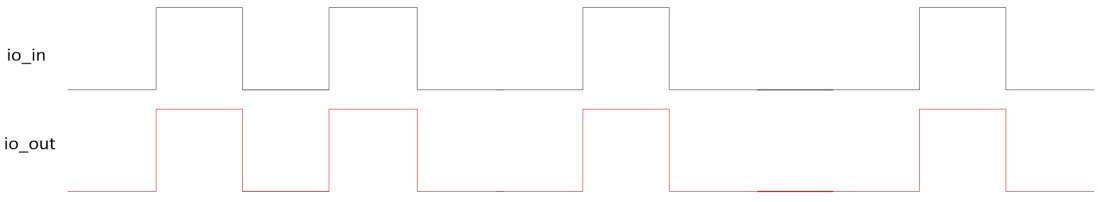


Figure 30-16 Non-Uniform Period Signal

Figure 30-17 Input Signal Pulse Width Greater Than 80ms

30.4.3.2 Even Division (io_out High Level < 80ms)

io_out is the division of io_in, the division ratio N = n+1, where n is the register division factor configuration value.

io_out is edge-aligned with io_in, determined by the edge configuration mode.

When io_in is periodically uniformly distributed, io_out maintains an equal duty waveform after even division, as shown in Figure 18, Figure 19, and Figure 20.

When io_in is periodically non-uniformly distributed, io_out is an unequal duty waveform after even division, as shown in Figure 21.

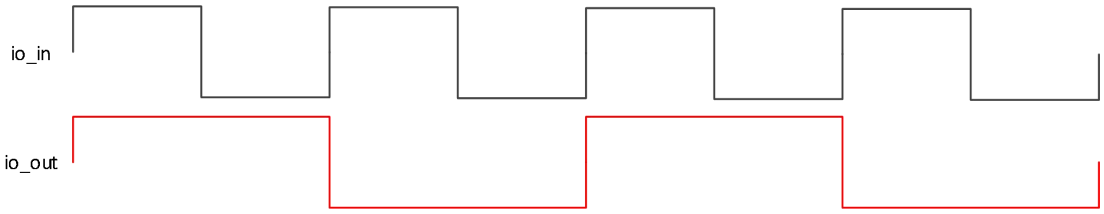


Figure 30-18 (n+1) Division

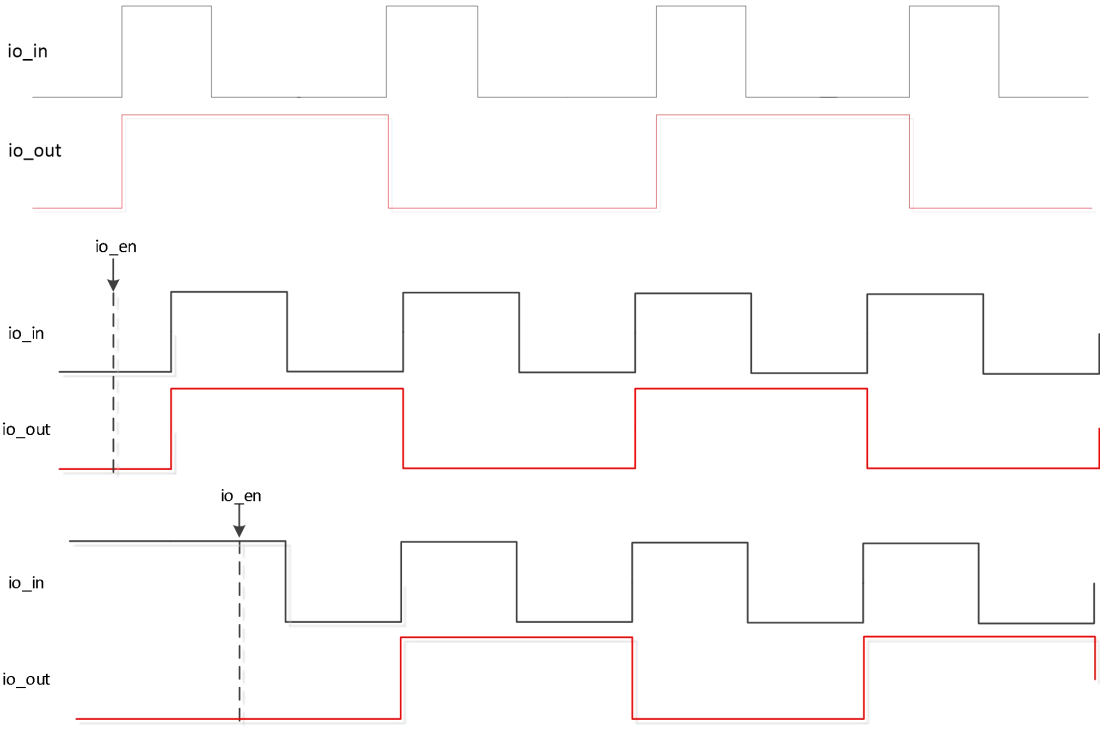


Figure 30-19 Rising Edge Alignment

Started at abnormal point, the first output pulse has unequal duty.

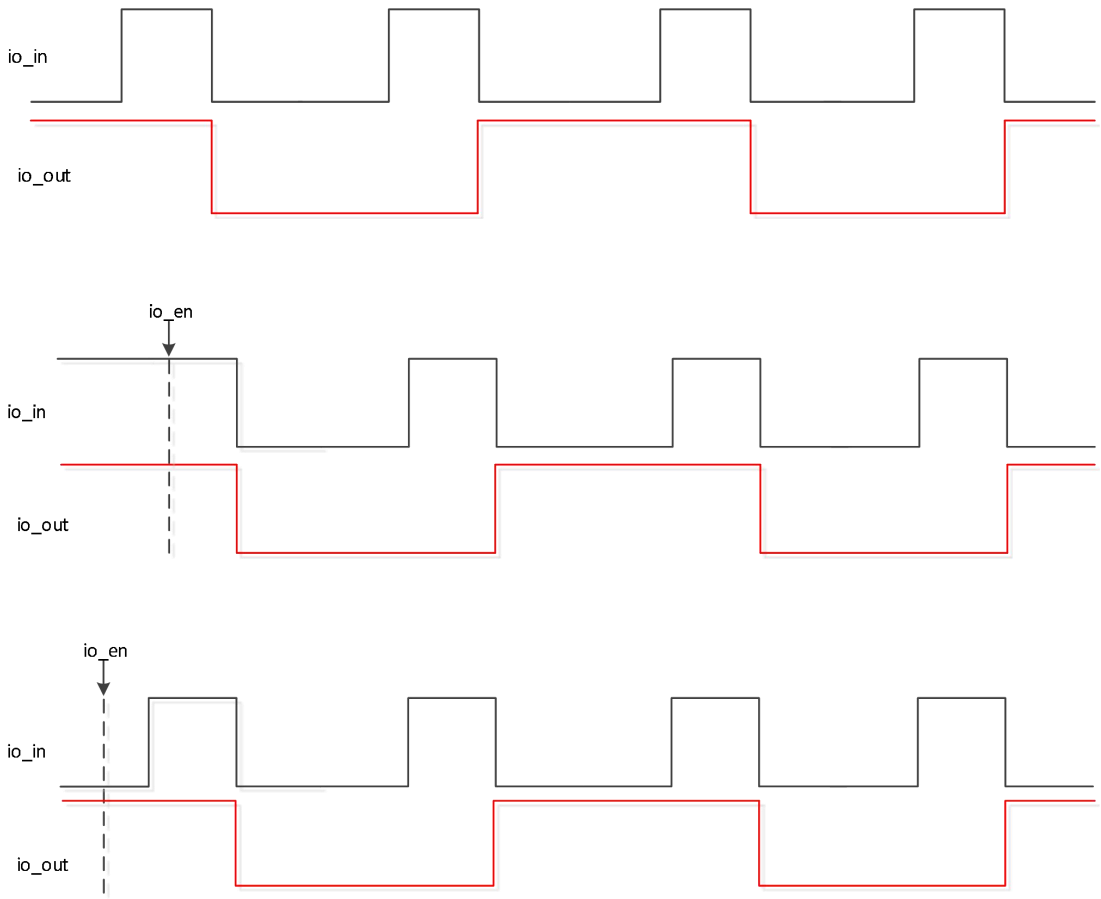


Figure 30-20 Falling Edge Alignment

Started at abnormal point, the first output pulse has unequal duty.

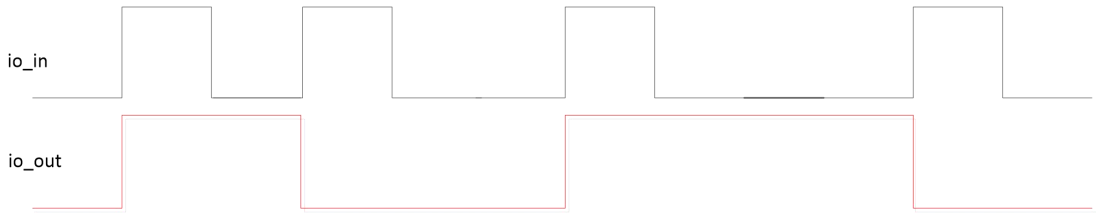


Figure 30-21 Non-Uniform Period Division

30.4.3.3 Odd Division (io_out High Level < 80ms)

io_out is the division of io_in, and the division ratio is configured by the register n+1.

io_out is edge-aligned with io_in, determined by the edge counting mode.

When io_in is periodically uniformly distributed, io_out is an unequal duty waveform after odd division, as shown in Figure 22 and Figure 23.

When io_in is periodically non-uniformly distributed, io_out is an unequal duty waveform after odd division, as shown in Figure 24.

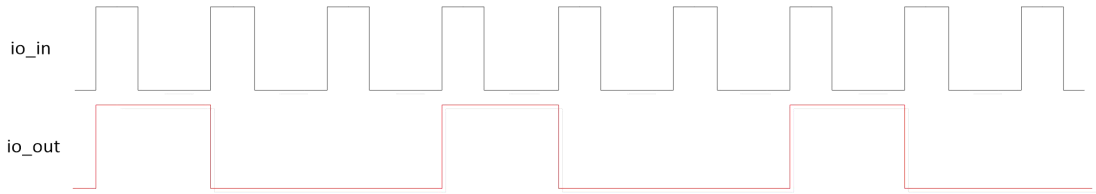


Figure 30-22 Rising Edge Alignment

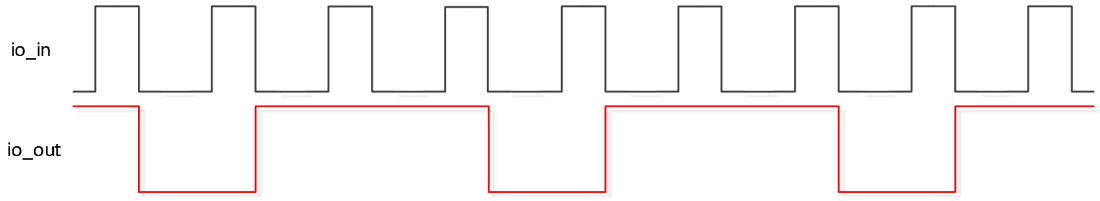


Figure 30-23 Falling Edge Alignment

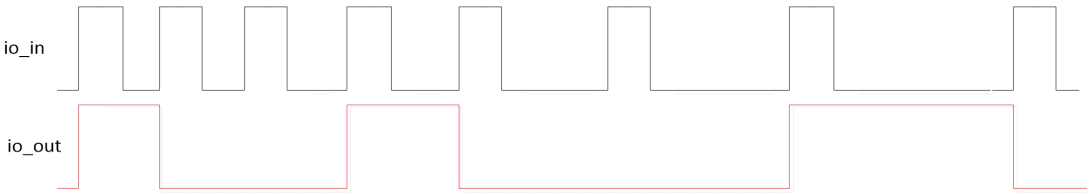


Figure 30-24 Non-Uniform Period

30.4.3.4 The Pulse Width of the Effective Level (High or Low) After Division ≥ 80ms

When configured for rising edge counting, the effective level is high level;

When configured for falling edge counting, the effective level is low level;

For rising edge counting, when the high-level pulse width after division ≥ 80ms, the high level is pulled low, maintaining the high level for 80ms, and the low level is maintained until the end of the division cycle.

Conversely, the same processing applies to falling edge counting.

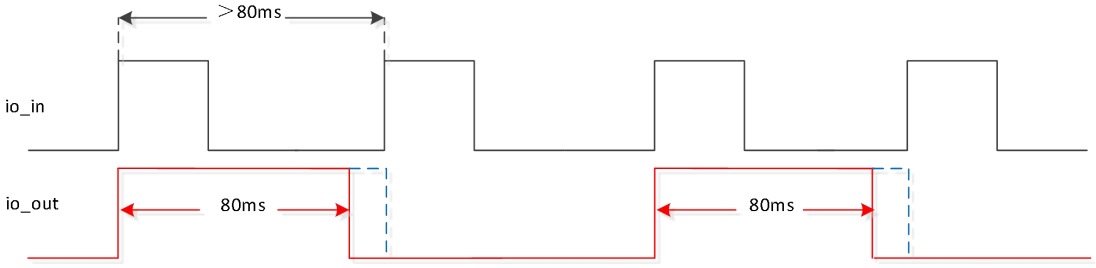


Figure 30-25 Unequal Duty (2x Division)

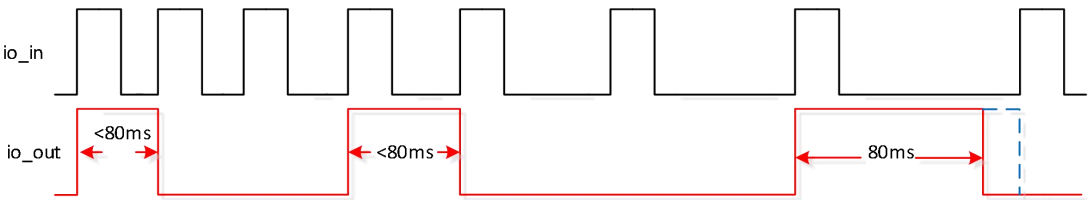


Figure 30-26 Non-Uniform Period (3x Division)

Figure 30-27 Input Effective Level Greater Than 80ms (2x Division)

30.4.4 Pulse Level Inversion Output

Supports inverting the level of the io_out signal before output, where io_out' is the output after division.

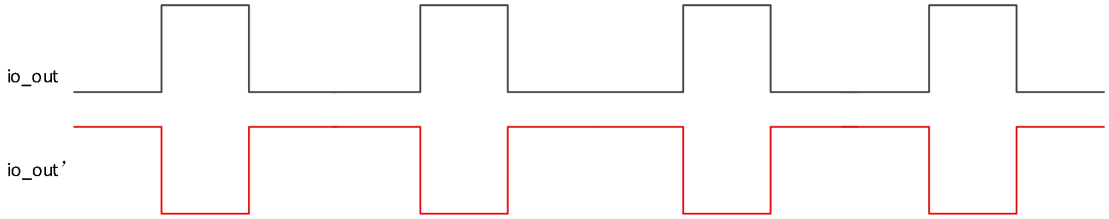


Figure 30-28 Pulse Level Inversion Output

30.4.5 Pulse Forwarding Output Pulse Configuration

When COPL[i] = 0, IOCNTi outputs the first low-frequency pulse when receiving the first input pulse, and outputs the (n+1)th low-frequency pulse when receiving the (n*division factor + 1)th input pulse;

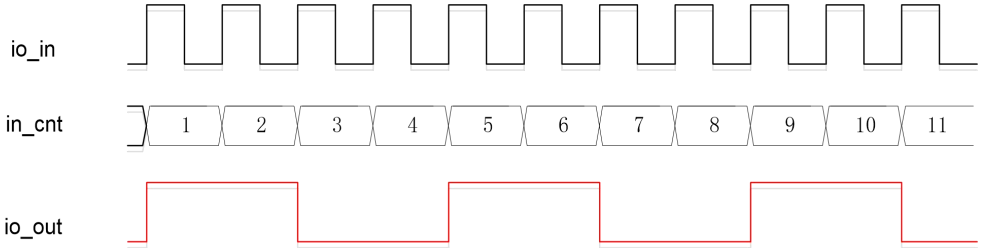


Figure 30-29 Output the First Low-Frequency Pulse at the First Input Pulse During 4x Division

When COPL[i] = 1, IOCNTi outputs the nth low-frequency pulse when receiving the (n*CNT)th input pulse;

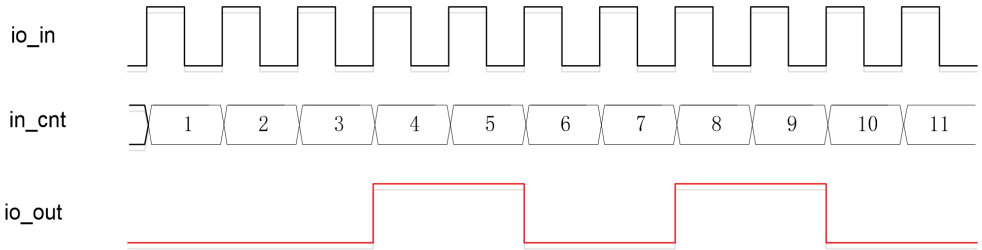


Figure 30-30 Output the First Low-Frequency Pulse at the 4th Input Pulse During 4x Division

30.4.6 Software Pulse Special Command

When the MCU writes to the corresponding IOCNT channel of the pulse forwarding special command register IOCNT_CMD, it immediately triggers one input pulse with a pulse width of 1 clock cycle. When enabling the special command for pulse forwarding, the filter function must be bypassed.

30.5 Register Definition

Application Notes:

1. Does not support modifying register configuration while the module is enabled.

2. Supports online system clock frequency adjustment; mode switching will clear the 80ms counter and restart counting.

3. Disabling the module enable will reset the module, and all count values will be cleared.

IOCNT Module Register Base Address offset

Module Name	Core0 Base Address	Core1 Base Address
IOCNT	0x2005_C000	0xA005_C000

IOCNT Interface Register Offset Addresses

Register Name	Address Offset	Description
IOCNT_CFG0	Offset+0x0	Pulse Forwarding Configuration Register 0
IOCNT_CFG1	Offset+0x4	Pulse Forwarding Configuration Register 1
IOCNT_CFG2	Offset+0x8	Pulse Forwarding Configuration Register 2
IOCNT_CFG3	Offset+0xC	Pulse Forwarding Configuration Register 3
IOCNT_CFG4	Offset+0x10	Pulse Forwarding Configuration Register 4
IOCNT_OUT0	Offset+0x20	Pulse Forwarding Output Pulse Counter 0
IOCNT_OUT1	Offset+0x24	Pulse Forwarding Output Pulse Counter 1
IOCNT_OUT2	Offset+0x28	Pulse Forwarding Output Pulse Counter 2
IOCNT_OUT3	Offset+0x2C	Pulse Forwarding Output Pulse Counter 3
IOCNT_OUT4	Offset+0x30	Pulse Forwarding Output Pulse Counter 4
IOCNT_CHNL	Offset+0x40	Pulse Forwarding Input Channel Selection Register
IOCNT_CTL	Offset+0x48	IOCNT Pulse Forwarding Control Register
IOCNT_CTL1	Offset+0x4C	Pulse Forwarding Output Pulse Configuration Register
IOCNT_CMD	Offset+0x50	Pulse Forwarding Special Command Register
IOCNT_REMD0	Offset+0x60	Pulse Forwarding Input Pulse Remainder Register 0
IOCNT_REMD1	Offset+0x64	Pulse Forwarding Input Pulse Remainder Register 1
IOCNT_REMD2	Offset+0x68	Pulse Forwarding Input Pulse Remainder Register 2
IOCNT_REMD3	Offset+0x6C	Pulse Forwarding Input Pulse Remainder Register 3
IOCNT_REMD4	Offset+0x70	Pulse Forwarding Input Pulse Remainder Register 4

30.5.1 IOCNT_CFG0~4（0x0~0x10）

Pulse Forwarding Configuration Register 0~4

Reset Value: 0000_0000h

Bits	Name	Description	R/W	Reset
31	INT	Pulse Inverted Output Enable: =0: Output pulse maintains same level direction as input pulse; =1: Output pulse is inverted before output.	RW	0
30:20	CFG	Pulse Division Factor Configuration: Division Factor = (CFG+1), Where CFG=0 means no division, supports up to 2048 division. Note: The divided pulse requires the high level to be maintained for 80ms; if faster than 80ms, it maintains equal duty output; It is recommended to configure CFG as an odd number for easier equal duty implementation.	RW	0
19:0	CNT[19:0]	When the IOCNT channel is enabled, CNT[19:0] acts as a 20-bit read-only pulse counter, recording the number of input pulses before division according to the configured trigger edge. The CNT register is cleared after read by default, but can be configured as cumulative type; refer to the CNT_CLR configuration in IOCNT_CTRL[16].	R	0

30.5.2 IOCNT_OUT0~4（0x20~0x30）

Pulse Forwarding Output Pulse Counter 0

Reset Value: 0000_0000h

Bits	Name	Description	R/W	Reset
31:20	---	Reserved	R	00
19:0	CNT	Pulse Counter, records the number of output pulses after division according to the configured trigger edge. The CNT register is cleared after read, design references the energy register.	R	00

30.5.3 IOCNT_CHNL（0x40）

Pulse Forwarding Input Channel Selection Register 0

Reset Value: 0000_0000h

Bits	Name	Description	R/W	Reset
31:30	-	Reserved	R	0
29:24	IOCNT4_SEL	Same as IOCNT0_SEL	RW	0
23:18	IOCNT3_SEL	Same as IOCNT0_SEL	RW	0
17:12	IOCNT2_SEL	Same as IOCNT0_SEL	RW	0
11:6	IOCNT1_SEL	Same as IOCNT0_SEL	RW	0
5:0	IOCNT0_SEL	=0x0: External Input =0x1: Internal Input Special Command =Other: Reserved	RW	0

		Note: When enabling special command for pulse forwarding, the filter function must be bypassed.		

30.5.4 IOCNT_CTRL（0x48）

Pulse Forwarding Control Register

Bits	Name	Description	R/W	Reset
31:25	FLT_CFG	IOCNT4~0 Filter Cycle Count Configuration Filter Cycle = FLT_CFG +1 clock cycles, maximum configurable 128 cycles. Only when the width is less than the filter cycle, it is filtered out. Greater than or equal to the filter cycle, it will not be filtered out. E.g., FLT_CFG=3, then filter cycle is 4 clock cycles, only waveform width less than 4 clock cycles will be filtered out; greater than or equal to 4 clock cycles will not be filtered out.	RW	0x1f
24:23	--	Reserved	R	0
21	FLT_BYPASS4	IOCNT4 Filter Function Bypass Enable =0, Filter function enabled, filter according to flt_cfg configuration =1, Bypass filter function	RW	0
20	FLT_BYPASS3	IOCNT3 Filter Function Bypass Enable =0, Filter function enabled, filter according to flt_cfg configuration =1, Bypass filter function	RW	0
19	FLT_BYPASS2	IOCNT2 Filter Function Bypass Enable =0, Filter function enabled, filter according to flt_cfg configuration =1, Bypass filter function	RW	0
18	FLT_BYPASS1	IOCNT1 Filter Function Bypass Enable =0, Filter function enabled, filter according to flt_cfg configuration =1, Bypass filter function	RW	0
17	FLT_BYPASS0	IOCNT0 Filter Function Bypass Enable =0, Filter function enabled, filter according to flt_cfg configuration =1, Bypass filter function	RW	0
16	CNT_CLR	Pulse Counter Type Configuration =0, All pulse counters are cleared after read =1, All pulse counters are not cleared after read	RW	0
15:13	--	Reserved	R	0
12:8	IOCNT_MODE	IOCNT4~IOCNT0 Pulse Count Mode Selection: =0, Rising Edge Trigger =1, Falling Edge Trigger	RW	0

		Default Rising Edge Trigger		
7:5	---	Reserved	R	0
4:0	IOCNTx_EN	IOCNT4~IOCNT0 Pulse Forwarding Function Enable Configuration: =0, Disabled =1, Enable IOCNT	RW	0

30.5.5 IOCNT_CTL1（0x4C）

Pulse Forwarding Output Pulse Configuration Register

Reset Value: 0000_0000h

Bits	Name	Description	R/W	Reset
31:5	---	Reserved	R	0
4:0	COPL	Output Pulse Configuration: bit0~4 correspond to IOCNT0~4 When COPL[i]=0, IOCNTi outputs the 1st low-frequency pulse at the 1st input pulse, and outputs the n+1th low-frequency pulse at the (n*division factor)+1th input pulse; When COPL[i]=1, IOCNTi outputs the nth low-frequency pulse at the (n*division factor)th input pulse; When COPL=0, IOCNT output pulse first follows the input pulse to output one pulse, then accumulates pulses according to the division factor, and outputs the next pulse when the pulse count reaches the division factor value +1.	RW	0

30.5.6 IOCNT_CMD（0x50）

Pulse Forwarding Control Register

Reset Value: 0000_0000h

Bits	Name	Description	R/W	Reset
31:5	-	Reserved	R	0
4	IOCNT4_KICK	IOCNT4 Pulse Forwarding Special Command Generation Bit Software writes 1, immediately triggers one input pulse for IOCNTx; Software write 0 is invalid, automatically cleared by hardware. Note: When enabling special command for pulse forwarding, the filter function must be bypassed.	W	0
3	IOCNT3_KICK	IOCNT3 Pulse Forwarding Special Command Generation Bit Software writes 1, immediately triggers one input pulse for IOCNTx; Software write 0 is invalid, automatically cleared by hardware. Note: When enabling special command for pulse	W	0

		forwarding, the filter function must be bypassed.		
2	IOCNT2_KICK	IOCNT2 Pulse Forwarding Special Command Generation Bit Software writes 1, immediately triggers one input pulse for IOCNTx; Software write 0 is invalid, automatically cleared by hardware. Note: When enabling special command for pulse forwarding, the filter function must be bypassed.	W	0
1	IOCNT1_KICK	IOCNT1 Pulse Forwarding Special Command Generation Bit Software writes 1, immediately triggers one input pulse for IOCNTx; Software write 0 is invalid, automatically cleared by hardware. Note: When enabling special command for pulse forwarding, the filter function must be bypassed.	W	0
0	IOCNT0_KICK	IOCNT0 Pulse Forwarding Special Command Generation Bit Software writes 1, immediately triggers one input pulse for IOCNTx; Software write 0 is invalid, automatically cleared by hardware. Note: When enabling special command for pulse forwarding, the filter function must be bypassed.	W	0

30.5.7 IOCNT_REMD0~4（0x60~0x70）

Pulse Forwarding Control Register

Reset Value: 0000_0000h

Bits	Name	Description	R/W	Reset
31:11	-	Reserved	R	0
10:0	IOCNT_REMD	IOCNT Input Pulse Remainder Register, total 11 bits When IOCNT is enabled, this register is read-only, not writable, storing the remainder under the current division configuration; it will be automatically cleared after IOCNT is turned off; When IOCNT is turned off, this register is readable and writable. When a remainder is written, after restarting IOCNT, the next division output will accumulate based on this remainder. For example, if the chip detects a power-down event, the software can read this remainder and save it as the initial value for the next power-on, then turn off	RW	0

		IOCNT, and after the next power-on, manually fill the saved remainder into this register, then start IOCNT.		

30.6 Usage Flow

30.6.1 Internal Pulse Forwarding

When the input signal is high-level valid, use rising edge counting, the steps are as follows;

- Configure IOCNT_CTL register IOCNTx_MODE to select rising edge counting;
- Configure the pulse forwarding input channel register IOCNT_CHNL to select the input source. If external input is selected, configure the channel selection register of the AOS module;
- Optionally configure division counting and level inversion;
- Configure pulse forwarding enable.

30.6.2 High-Level Pulse Width Valid

When the input signal is high-level valid, it is recommended to connect an external pull-down resistor to the INTx hardware circuit so that the default input of INTx is low level, and use rising edge counting. The steps are as follows;

- Configure IOCNT_CTL register IOCNTx_MODE to select rising edge counting;
- Optionally configure IOCNT_CFG register for division counting and level inversion;
- Configure GPIO multiplexing relationship;
- Configure pulse forwarding enable.

30.6.3 Low-Level Pulse Width Valid

When the input signal is low-level valid, it is recommended to connect an external pull-up resistor to the INTx hardware circuit so that the default input of INTx is high level, and use falling edge counting. The steps are as follows:

- Configure IOCNT_CTL register IOCNTx_MODE to select falling edge counting;
- Optionally configure IOCNT_CFG register for division counting and level inversion;
- Configure GPIO multiplexing relationship;
- Configure pulse forwarding enable.

30.6.4 Application Notes

- Modifying register configuration when the module is enabled is not supported.
- Online system clock frequency adjustment is supported. After mode switching, the 80ms counter is cleared and restarts counting.
- Disabling the module enable will reset the module and clear all count values.
- During laboratory testing, the input is stable (the so-called static electric meter), and the input is of equal duty.
- The input pulse remainder value IOCNT_REMD can be written when the enable is disabled. The written value cannot exceed the division factor value; otherwise, it cannot be written.

31 D2F

31.1 Introduction

D2F (Customized Energy/Pulse Integration) is a crucial function in energy management. It can integrate the measured power value over time into energy values and energy verification pulses. It can be applied in scenarios and fields such as power system monitoring, energy management, electric vehicle charging monitoring, equipment operation monitoring, and intelligent meter reading systems.

The input of this module is the power register code value, and the output is the energy register code value and the energy verification pulse output through the IO port.

31.2 Features

- Supports 12 channels of D2F integrators, all of which support verification pulse output. Every 4 channels of D2F share 1 verification pulse port for output, supporting a maximum of 3 verification pulse ports for simultaneous output;
  (CF0: D2F00~D2F03; CF1: D2F04~D2F07; CF2: D2F08~D2F11)
- Supports grouping of 4 D2F channels into one group, with 3 groups of pulse constants configurable separately;
  (HFCONST0: D2F00~D2F03; HFCONST1: D2F04~D2F07; HFCONST2: D2F08~D2F11)
- Supports filling the power register with 32-bit signed numbers, with a code value range of ±2^31;
- Supports grouping of 4 D2F channels into one group, with 3 groups of creep start thresholds configurable separately;
- Supports independent interrupt enable and interrupt status for each D2F channel;
- Supports configurable energy register modes: accumulative type or clear-after-read type;

31.3 Registers

Base Addresses of D2F Module Registers

Module Name	Core0 Bus Mapped Address	Core1 Bus Mapped Address
D2F	0x2007_C000	0xA007_C000

Offset Addresses of D2F Interface Registers

Register Name	Address Offset	Description
HFCONST0	0x0	High-frequency pulse constant register for D2F00~D2F03
HFCONST1	0x4	High-frequency pulse constant register for D2F04~D2F07
HFCONST2	0x8	High-frequency pulse constant register for D2F08~D2F11
RESERVED	0x18~0x2C	Reserved
D2F_CTRL	0x30	D2F module control register
D2F_CFG	0x34	D2F module configuration register
RESERVED1	0x38~0x3C	Reserved
D2F_CF_CFG	0x40	D2F pulse output configuration register
RESERVED2	0x44~0x4C	Reserved
D2F_IE	0x50	D2F interrupt enable register

D2F_IF	0x54	D2F interrupt flag register
D2F_NOLOAD	0x58	D2F creep status register
RESERVED3	0x5C	Reserved
D2F_P00~D2F_P11	0x60~0x8C	D2F power input registers
RESERVED4	0x90~0x11C	Reserved
D2F_E00~D2F_E11	0x120~0x14C	D2F energy registers
RESERVED5	0x150~0x1DC	Reserved
D2F_StartTh00~D2F_StartTh02	0x1E0~0x1E8	D2F start threshold registers

31.3.1 HFCONST0/1/2（0x00~0x08）

D2F High-frequency Pulse Constant Registers

Offset Address = 0x00

Bits	Name	Description	R/W	Reset Value
31:16	--	Reserved	RO	0x0
15:0	HFCONST0	Pulse constant for custom energy registers D2FE00~D2FE03 and custom pulse output CF0. High-frequency pulse constant is a 16-bit unsigned number.	RW	0x1000

Offset Address = 0x04

Bits	Name	Description	R/W	Reset Value
31:16	--	Reserved	RO	0x0
15:0	HFCONST1	Pulse constant for custom energy registers D2FE04~D2FE07 and custom pulse output CF1. High-frequency pulse constant is a 16-bit unsigned number.	RW	0x1000

Offset Address = 0x08

Bits	Name	Description	R/W	Reset Value
31:16	--	Reserved	RO	0x0
15:0	HFCONST2	Pulse constant for custom energy registers D2FE08~D2FE11 and custom pulse output CF2. High-frequency pulse constant is a 16-bit unsigned number.	RW	0x1000

Note 1:

Based on PF=1.0, the theoretical HFConst value is calculated using the standard active power value P, meter constant EC and other parameters according to the following formula:

HFConst＝INT[P*3.6*10^6*femu / (EC*Un*Ib*2^33)]

At rated input and PF=1.0, the standard active power value P = INT(U_standard * I_standard / 223),

Where U_standard is the standard voltage RMS register value at rated voltage input, and I_standard is the standard current RMS register value at rated current input;

femu: 460.8KHz/32.768KHz;

Un: Rated input voltage;

Ib: Rated input current;

EC: Meter constant;

Configure the corresponding HFConst0~2 registers according to the calculation result.

The clock for D2F integration of the three-phase metering chip is 1.024MHz, and the clock for D2F integration of the MCU chip is 460.8KHz/32.768KHz. If the dual-core MCU (HFconst) is connected to the MAKSA three-phase metering chip (HFconst1), assuming the meter pulse constants EC defined by both are completely consistent, then as long as the following condition is met:

HFconst=(0.4608/1.024)*HFconst1

Directly read the power register value (32-bit, such as PowerP) of the MAKSA three-phase metering chip, fill it into the power register D2FP, and then the pulse output CF can maintain the same pulse output speed as the corresponding CF pin of the three-phase metering chip.

If the defined pulse constants are different, both the HFCONST adjustment and the filling value of the customized power register can be used.

Note 2:

Whether the D2F integration clock of the MCU chip is 460.8KHz or 32.768KHz depends on the D2F_CFG.CLK_SEL configuration, as shown in the following table:

- When D2F_CFG.CLK_SEL is configured to default D2F clock mode:

The corresponding relationship between the system main frequency sys_clk and the integrator working clock d2f_clk is as follows:

sys_clk	d2f_clk
32.768KHz	32.768KHz
1.8432MHz	460.8KHz
3.6864MHz	460.8KHz
7.3728MHz	460.8KHz
14.7456MHz	460.8KHz
29.4912MHz	460.8KHz
58.9824MHz	460.8KHz

- When D2F_CFG.CLK_SEL is configured to constant D2F clock mode (Note: Only affects D2F08~D2F11; other D2F channels still use the default D2F clock mode):

The corresponding relationship between the system main frequency sys_clk and the integrator working clock d2f_clk is as follows:

sys_clk	d2f_clk
32.768KHz	32.768KHz
1.8432MHz	32.768KHz
3.6864MHz	32.768KHz
7.3728MHz	32.768KHz
14.7456MHz	32.768KHz
29.4912MHz	32.768KHz
58.9824MHz	32.768KHz

31.3.2 D2F_CTRL（0x30）

Offset Address =0x30

Bits	Name	Description	R/W	Reset Value
31:1	--	Reserved	RO	0x0
0	EN	Module Enable 0: Enable D2F module; 1: Stop D2F operation and reset all counters and digital operation units, but do not reset registers; (Default)	RW	0x1

31.3.3 D2F_ CFG（0x34）

Offset Address =0x34

Bits	Name	Description	R/W	Reset Value
31:3	--	Reserved	RO	0x0
2	CLK_SEL	Integration Clock Mode Selection 0: Integration clock is default D2F clock mode; 1: Integration clock is constant D2F clock mode.	RW	0x0
1	Syn_SimTC	Synchronized loading configuration selection linked to SimTC 0: Default loading mode; 1: Synchronized loading mode linked to SimTC. Note: AOS module needs to be configured to select SIMPTCx to trigger D2F.	RW	0x0
0	ERegCAR	Energy Register Type Selection 0: Read-after-clear type; (Default) 1: Accumulative type;	RW	0x0

31.3.4 D2F_CF_CFG（0x40）

Offset Address =0x40

Bits	Name	Description	R/W	Reset Value
31:10	--	Reserved	RO	0x0
9:8	CF2_SEL	CF2 Port Output D2F Selection 0: D2F08; 1: D2F09; 2: D2F10; 3: D2F11;	RW	0x0
7:6	--	Reserved	RO	0x0
5:4	CF1_SEL	CF1 Port Output D2F Selection 0: D2F04; 1: D2F05;	RW	0x0

		2: D2F06; 3: D2F07;		
3:2	--	Reserved	RO	0x0
1:0	CF0_SEL	CF0 Port Output D2F Selection 0: D2F00; 1: D2F01; 2: D2F02; 3: D2F03;	RW	0x0

31.3.5 D2F_IE（0x50）

Offset Address =0x50

Bits	Name	Description	R/W	Reset Value
31:25	--	Reserved	RO	0x0
24	PCOV_IE	D2F Power Coverage Interrupt Enable	RW	0x0
23:12	--	Reserved	RO	0x0
11	IE11	D2F11 Interrupt Enable	RW	0x0
10	IE10	D2F10 Interrupt Enable	RW	0x0
9	IE09	D2F09 Interrupt Enable	RW	0x0
8	IE08	D2F08 Interrupt Enable	RW	0x0
7	IE07	D2F07 Interrupt Enable	RW	0x0
6	IE06	D2F06 Interrupt Enable	RW	0x0
5	IE05	D2F05 Interrupt Enable	RW	0x0
4	IE04	D2F04 Interrupt Enable	RW	0x0
3	IE03	D2F03 Interrupt Enable	RW	0x0
2	IE02	D2F02 Interrupt Enable	RW	0x0
1	IE01	D2F01 Interrupt Enable	RW	0x0
0	IE00	D2F00 Interrupt Enable 0: Disable; (Default) 1: Enable; Other bits are configured in the same way.	RW	0x0

31.3.6 D2F_IF（0x54）

Offset Address =0x54

Bits	Name	Description	R/W	Reset Value
31:25	--	Reserved	RO	0x0
24	PCOV_IF	D2F Power Coverage Interrupt Flag, indicates that in synchronized loading mode, a new power value is written to the D2FP00~11 registers before the previously written power value is loaded into the integrator; Write 1 to clear 0; if the corresponding PCOV_IE is enabled, writing 1 clears the flag and the interrupt simultaneously.	W1C	--

23:12	--	Reserved	RO	0x0
11	IF11	D2F11 Energy Pulse Interrupt Flag, write 1 to clear 0; if the corresponding IE is enabled, writing 1 clears the flag and the interrupt simultaneously.	W1C	--
10	IF10	D2F10 Energy Pulse Interrupt Flag, write 1 to clear 0; if the corresponding IE is enabled, writing 1 clears the flag and the interrupt simultaneously.	W1C	--
9	IF09	D2F09 Energy Pulse Interrupt Flag, write 1 to clear 0; if the corresponding IE is enabled, writing 1 clears the flag and the interrupt simultaneously.	W1C	--
8	IF08	D2F08 Energy Pulse Interrupt Flag, write 1 to clear 0; if the corresponding IE is enabled, writing 1 clears the flag and the interrupt simultaneously.	W1C	--
7	IF07	D2F07 Energy Pulse Interrupt Flag, write 1 to clear 0; if the corresponding IE is enabled, writing 1 clears the flag and the interrupt simultaneously.	W1C	--
6	IF06	D2F06 Energy Pulse Interrupt Flag, write 1 to clear 0; if the corresponding IE is enabled, writing 1 clears the flag and the interrupt simultaneously.	W1C	--
5	IF05	D2F05 Energy Pulse Interrupt Flag, write 1 to clear 0; if the corresponding IE is enabled, writing 1 clears the flag and the interrupt simultaneously.	W1C	--
4	IF04	D2F04 Energy Pulse Interrupt Flag, write 1 to clear 0; if the corresponding IE is enabled, writing 1 clears the flag and the interrupt simultaneously.	W1C	--
3	IF03	D2F03 Energy Pulse Interrupt Flag, write 1 to clear 0; if the corresponding IE is enabled, writing 1 clears the flag and the interrupt simultaneously.	W1C	--
2	IF02	D2F02 Energy Pulse Interrupt Flag, write 1 to clear 0; if the corresponding IE is enabled, writing 1 clears the flag and the interrupt simultaneously.	W1C	--
1	IF01	D2F01 Energy Pulse Interrupt Flag, write 1 to clear 0; if the corresponding IE is enabled, writing 1 clears the flag and the interrupt simultaneously.	W1C	--
0	IF00	D2F00 Energy Pulse Interrupt Flag, write 1 to clear 0; if the corresponding IE is enabled, writing 1 clears the flag and the interrupt simultaneously.	W1C	--

31.3.7 D2F_NOLOAD（0x58）

Offset Address =0x58

Bits	Name	Description	R/W	Reset Value
31:12	--	Reserved	RO	0x0

11	D2F11_NLD	D2F11 No-load Status Flag, indicates whether the D2F channel is disabled or in no-load state 0: Not in no-load state; 1: In no-load state;	RO	0x1
10	D2F10_NLD	D2F10 No-load Status Flag 0: Not in no-load state; 1: In no-load state;	RO	0x1
9	D2F09_NLD	D2F09 No-load Status Flag 0: Not in no-load state; 1: In no-load state;	RO	0x1
8	D2F08_NLD	D2F08 No-load Status Flag 0: Not in no-load state; 1: In no-load state;	RO	0x1
7	D2F07_NLD	D2F07 No-load Status Flag 0: Not in no-load state; 1: In no-load state;	RO	0x1
6	D2F06_NLD	D2F06 No-load Status Flag 0: Not in no-load state; 1: In no-load state;	RO	0x1
5	D2F05_NLD	D2F05 No-load Status Flag 0: Not in no-load state; 1: In no-load state;	RO	0x1
4	D2F04_NLD	D2F04 No-load Status Flag 0: Not in no-load state; 1: In no-load state;	RO	0x1
3	D2F03_NLD	D2F03 No-load Status Flag 0: Not in no-load state; 1: In no-load state;	RO	0x1
2	D2F02_NLD	D2F02 No-load Status Flag 0: Not in no-load state; 1: In no-load state;	RO	0x1
1	D2F01_NLD	D2F01 No-load Status Flag 0: Not in no-load state; 1: In no-load state;	RO	0x1
0	D2F00_NLD	D2F00 No-load Status Flag 0: Not in no-load state; 1: In no-load state;	RO	0x1

31.3.8 D2F_P00~11（0x60~0x8C）

Taking D2F_P00 as an example, other registers are the same.

Offset Address = 0x60~0x8C

Bits	Name	Description	R/W	Reset

				Value
31:0	D2F_P00	Custom power input registers D2F_P00~ D2F_P11 In two's complement format, 32-bit signed number, where the highest bit is the sign bit. Readable and Writable.	RW	0x0

31.3.9 D2F_E00~11（0x120~0x14C）

Taking D2F_E00 as an example, other registers are the same.

Offset Address = 0x120~0x14C

Bits	Name	Description	R/W	Reset Value
31:0	D2F_E00	Custom energy registers D2F_E00~ D2F_E11 32-bit unsigned number, read-only. Represents the accumulated number of corresponding reverse pulses. The minimum unit of the register represents energy of 1/EC KWh, where EC is the meter constant. The energy register overflows from 0xFFFFFFFF to 0x00000000, users need to manage this themselves. The energy register can be selected as accumulative type or read-after-clear type, configured by D2F_CFG.	RO	0x0

31.3.10 D2F_StartTh00~02（0x1E0~0x1E8）

Taking D2F_StartTh00 as an example, other registers are the same.

Offset Address = 0x1E0~0x1E8

Bits	Name	Description	R/W	Reset Value
31	Reserved	Reserved	RO	0x0
30:0	D2F_StartTh00	Custom start threshold register is a 32-bit unsigned number, the highest bit is always 0. When D2F_Pxx is updated, the D2F module takes the value for comparison. When the absolute value of D2F_Pxx is less than or equal to the corresponding D2F_StartThxx, this power does not participate in D2F operation, and the D2F module automatically adds 0 power integration. (No-load) Next time D2F_Pxx is updated, when the absolute value of D2F_Pxx is greater than the corresponding D2F_StartThxx, this power participates in D2F operation. (Start)	RW	0x0

31.4 Software Configuration Process and Application Notes

31.4.1 Software Writing Integration

- Enable the D2F module clock gating;
- Configure the HFCONST0/1/2 registers as required;
- Configure the D2F_StartTh0/1/2 start creep thresholds as required;
- Configure the D2F_IE to enable interrupts as required;

- Configure the D2F_CF_CFG to enable pulse output as required;
- Configure D2F_CFG to select the energy register type and whether to link the SIMPTC mode;
- Configure D2F_CTRL to enable the D2F module.
- Configure D2F_P00~D2FP11 to write the power to be integrated as required;

31.4.2 Application Notes

- Do not switch the configurations of D2F_CFG and D2F_CF_CFG after D2F integration is enabled;
- When enabling the synchronized loading function linked to SIMPTCx, the AOS module needs to be configured to select SIMPTCx to trigger D2F.

32 Secure Execution Environment (SPE)

Data security is of paramount importance. The security engine inside the chip provides a reliable data security storage and management mechanism to ensure that sensitive data is encrypted and stored, thereby guaranteeing the security of data storage.

The chip is equipped with an independent secure encryption core. The secure core resides in the SPE, while the application core is located in the NSPE. The registers and cache data of the secure core cannot be accessed by the application core, as it is physically isolated from the NSPE. The SPE accesses the NSPE through a one-way bus, and the NSPE cannot access the SPE in reverse. The NSPE sends commands via the Mailbox to request services and functions in the SPE, and the SPE returns responses or status flags through the Mailbox.

SPE components include: secure core, encryption/decryption engine, secure core-exclusive SRAM, FLASH area accessible only to the secure core, ROM, Mailbox, encryption/decryption firmware, and secure boot code.

The secure system and non-secure system adopt physical isolation to meet the system architecture security standards, as shown in the following Block Diagram 31-1.

- The Core0 secure system includes a private bus, private memory space (ROM, Flash, SRAM), private security modules (encryption/decryption), and private peripherals.
- The program of Core0 runs in the secure program space, and Core1 has no channel to access the program area of Core0, ensuring the program security of Core0.
- Core1 has no direct data channel to access Core0, and communication between the two cores can only be realized through security mechanisms (software and hardware).
- The Core0 secure system provides secure service interfaces for Core1, enabling interaction through Mailbox, shared memory, interrupts, etc.
- The Core0 kernel is equipped with PMP function (MPU), which can configure the program execution space.

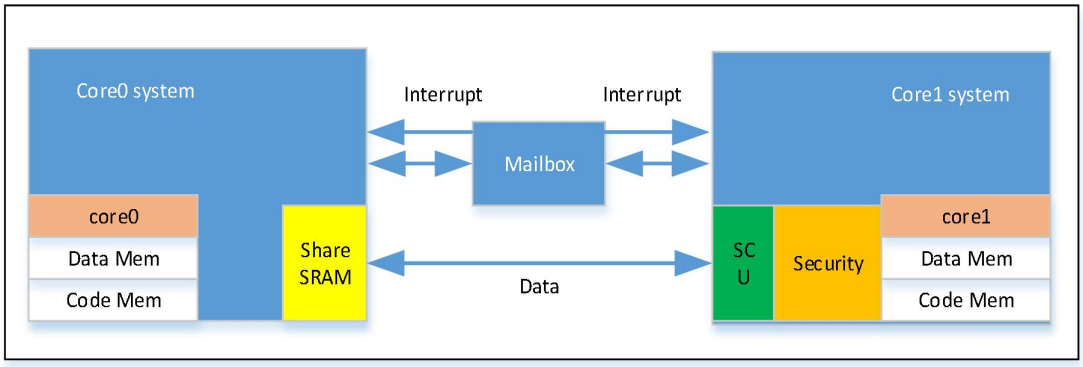


Figure 32-1 Block Diagram of Secure System and Non-Secure System

32.1 Framework

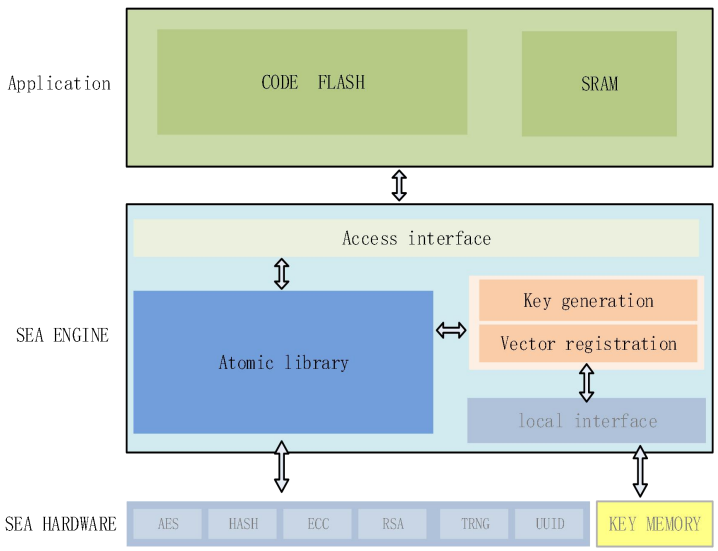
The data security storage mechanism is composed of components such as the Security Encryption Accelerator (SEA) hardware module, SEA ENGINE secure encryption firmware, and a dedicated encrypted storage area inside the chip.

The Security Encryption Accelerator (SEA) hardware module provides underlying data encryption and decryption algorithms for the chip, including symmetric encryption algorithms (AES) certified by the FIPS140-3 standard, public key cryptography algorithms (ECDSA, ECDH, etc.) modules, and a True Random Number Generator (TRNG) module using Fibonacci rings as entropy sources.

SEA ENGINE is a complete secure encryption firmware in the chip's ROM and is permanently non-tamperable. It provides atomic operation mechanisms for the SEA hardware module, calculation key generation algorithms, and an interface access management unit.

The chip internally integrates a FLASH. In addition to the 512Kbit data area, this built-in FLASH has a 1Kbit KEY storage area for storing sensitive data. Unlike the data area, the KEY storage area only supports one-time programming. After the one-time programming operation is completed, the hardware completely shuts off the write operation path to ensure that the written sensitive data cannot be tampered with.

The initial vector storage, UUID storage, etc., of the key management system of SEA ENGINE are all stored in the KEY storage area. This storage area is not open to the application system, cannot be accessed through the system bus, and is managed by SEA ENGINE.



Block Diagram 32-2 Data Security Storage Framework

32.2 SEA ENGINE

SEA ENGINE is a complete secure encryption firmware in the chip's ROM and is permanently non-tamperable. It provides dedicated atomic operation control library functions for the SEA hardware module, calculation key generation algorithms, and an interface access management unit.

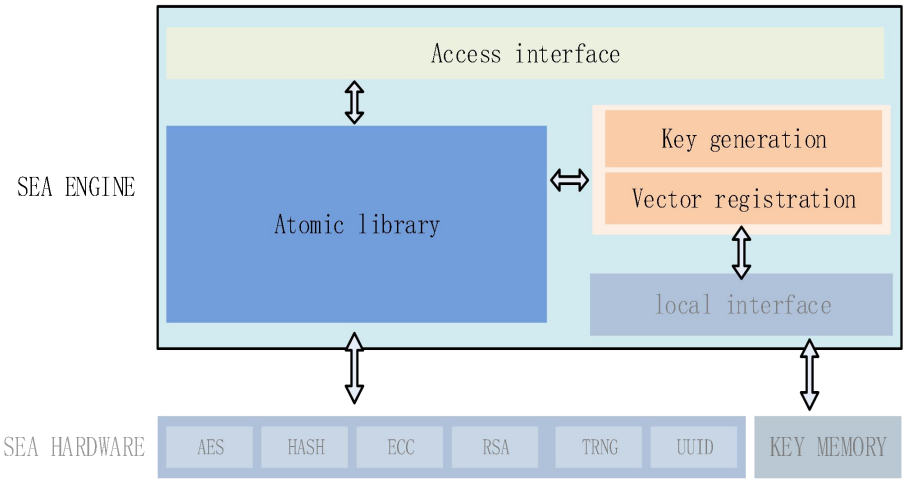


Figure 32-3 SEA ENGINE Block Diagram

32.2.1 SEA Atomic Operation Library Functions

The SEA atomic operation library functions provide all operational capabilities of the SEA hardware module, encrypting plaintext data into ciphertext data or decrypting ciphertext data into plaintext data. To prevent malicious access and tampering of data, all encryption/decryption operations inside the library functions are atomic operations. Any modification to the data during the intermediate process of encryption/decryption or the internal software flow of the library functions will cause the current encryption/decryption to be terminated.

Any process of encrypting plaintext data or decrypting ciphertext data is regarded as an atomic operation, and the library functions perform program switching and data processing according to the specified state internally.

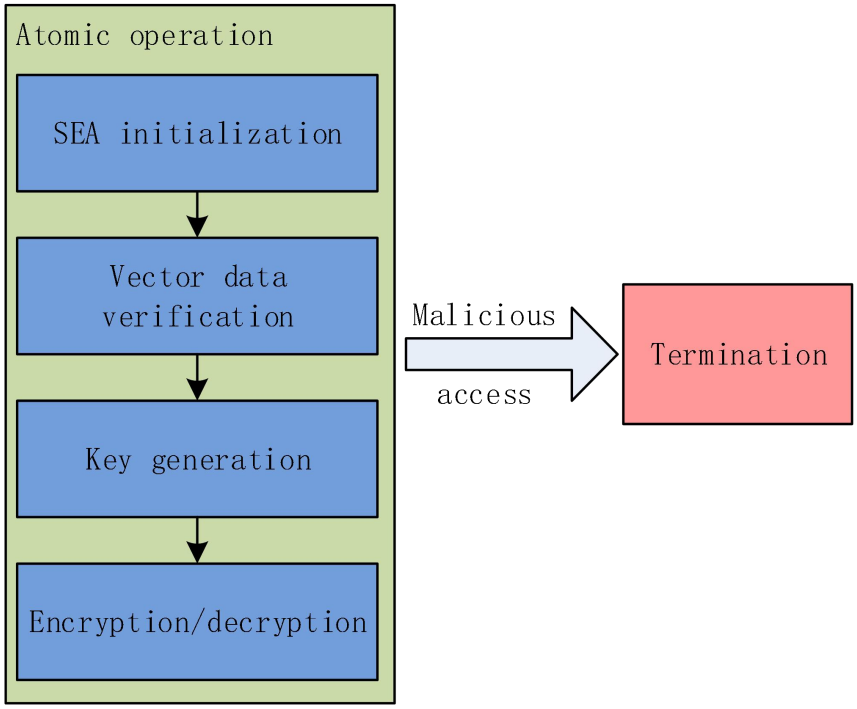


Figure 32-4 SEA Atomic Operation Library Functions

SEA initialization: Initializes the SEA hardware circuit controller and obtains the plaintext/ciphertext data to be encrypted/decrypted according to the data pointer passed by the access interface.

Vector data verification: Obtains the initial vector of the encryption/decryption key generation algorithm from the

KEY storage area of the built-in FLASH and performs data and issuance verification. If the initial vector in the KEY storage area is invalid, the current encryption/decryption process is terminated.

Key generation: Uses the valid initial vector in the KEY storage area and the chip's UUID to generate an encryption/decryption key according to a specific algorithm. The encryption/decryption key is not stored in any storage device and is destroyed immediately after the current encryption/decryption operation is completed.

Encryption/Decryption: Uses SEA HARDWARE to encrypt/decrypt the input plaintext/ciphertext with the key generated by key generation, ensuring high performance of data processing.

The atomic operation library functions ensure that each sub-operation process is an indispensable part of the entire operation and has strict timing requirements. This ensures that any forged access is identified as malicious access, and the current operation is terminated to ensure data security.

32.2.2 Key Registration and Generation

The keys used by SEA ENGINE in the data encryption/decryption process are not saved in any form. Key management consists of key registration and key generation processes. Key registration uses TRNG and the unique UUID inside each chip to generate the initial vector required for key generation through a specified algorithm, and the initial vector is stored in the KEY storage area. Key generation calculates and generates a temporary encryption/decryption key suitable for the current operation based on the initial vector and UUID.

32.2.2.1 KEY Storage Area (Secure Storage Area)

The chip internally integrates a 512Kbit FLASH. In addition to the 512Kbit data area, this built-in FLASH has a 1Kbit special storage area. Unlike the data area, this area only supports one-time programming. After the one-time programming operation is completed, the hardware completely shuts off the write operation path to ensure that the written data cannot be tampered with.

The KEY storage area is the special storage area within the built-in FLASH. The initial vector required for key generation is stored in the KEY storage area and generated by the key registration process. The data storage in the KEY storage area has a data integrity verification mechanism to ensure data security and credibility.

32.2.2.1.1 Isolation of the KEY Storage Area

The read/write operations of the KEY storage area and the FLASH main storage area are isolated from each other. The main storage area supports two access methods: directly accessing main storage area data through the API interface (data is stored without encryption) and accessing the encryption engine through the API to obtain main storage area data (data is stored with encryption). The KEY storage area only supports one-time programming, which is completed during the chip mass production stage and is permanently non-tamperable. The read interface of the KEY storage area is different from that of the main storage area. The KEY storage area only supports reading by the internal encryption engine firmware and cannot be accessed through the API interface.

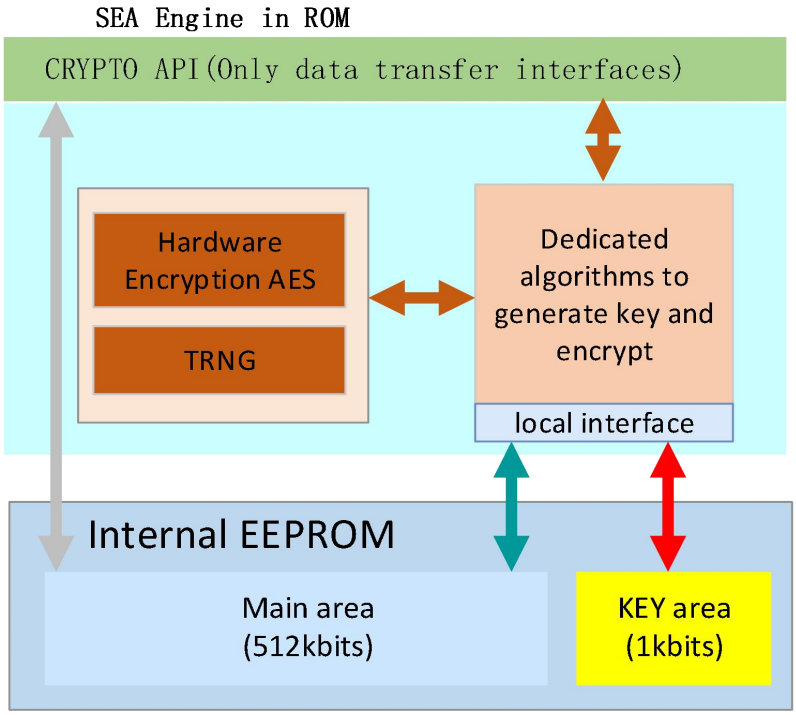


Figure 32-5 Access Methods of the KEY Storage Area and FLASH Main Storage Area

32.2.2.2 Key Registration

Key registration is implemented during the chip mass production test phase (FT test).

Key registration is based on the TRNG hardware module inside the chip. This hardware module uses multiple Fibonacci rings as entropy sources internally, complies with the FIPS-140-3 standard, and passes the NIST standard SP800-90B test.

Key registration generates multiple sets of initial key vectors, which are stored in the KEY storage area. The key registration process is as follows:

- Obtain two sets of UUID1 and UUID2 inside the chip (UUID is the unique identification code of the chip, which cannot be modified, is unique for each chip, and has random characteristics).
- Generate a random number RAND1 using TRNG.
- UUID1 ⊕ RAND1=IVECTOR1, UUID2 ⊕ RAND1=IVECTOR2.
- Repeat steps 2-3 above to generate IVECTOR3~IVECTOR4.
- Write the generated IVECTOR1~IVECTOR4 into the KEY storage area of the built-in FLASH.

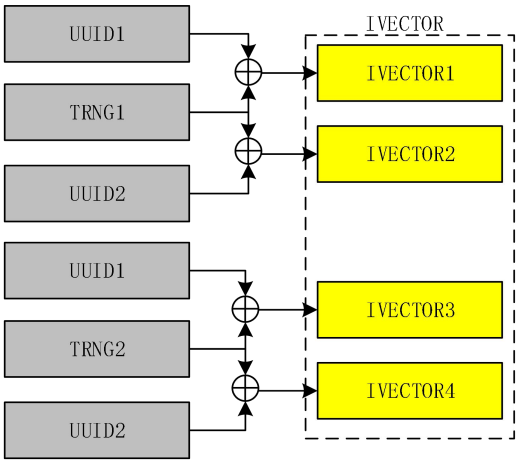


Figure 32-6 Key Registration Process

The initial key vector has random characteristics. Through logical operations based on the initial key vector, the validity of the initial key vector can be verified. The verification process of the initial key vector is as follows:

- Obtain two sets of UUID1 and UUID2 inside the chip.
- Read IVECTOR1~IVECTOR4.
- IVECTOR1 ⊕ IVECTOR2 == UUID1 ⊕ UUID2, and verify the second set of IVECTOR similarly.

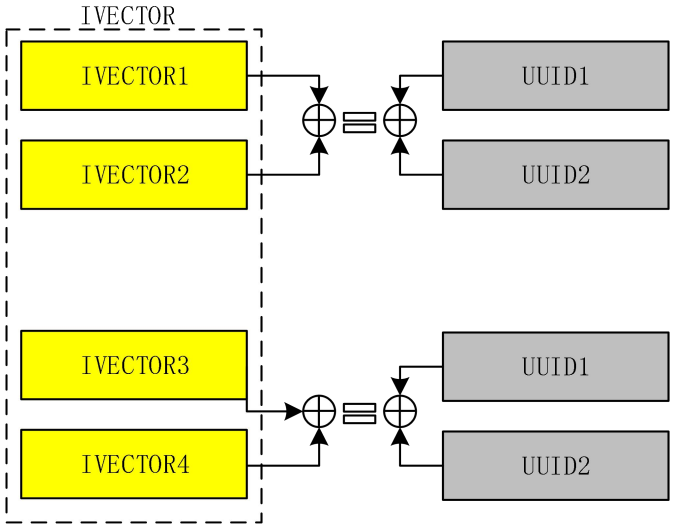


Figure 32-7 Initial Key Vector Verification Process

32.2.2.3 Key Generation

Key generation is implemented during each data encryption/decryption operation phase.

Key generation occurs during each data encryption/decryption operation phase and is obtained by operating on multiple sets of initial key vectors in the built-in FLASH KEY storage area. The key is only used as temporary data during each operation phase, is not persistently stored, is generated during the interface call process, and is destroyed after the interface call is completed.

Multiple sets of initial key vectors have random characteristics, but the validity of the initial key vectors can be verified. For the detailed verification process, refer to the key registration chapter.

The key generation process is as follows:

- Read IVECTOR1~IVECTOR4 and perform initial key vector validity verification. If the verification fails, data

read/write operations are not allowed.

- Obtain two sets of UUID1 and UUID2 inside the chip.
- UUID2 ⊕ IVECTOR1=KEY1, UUID1 ⊕ IVECTOR2=KEY2.
- Repeat step 3 to generate KEY3~KEY4.

Use KEY1~KEY4 as the 128-bit key for AES.

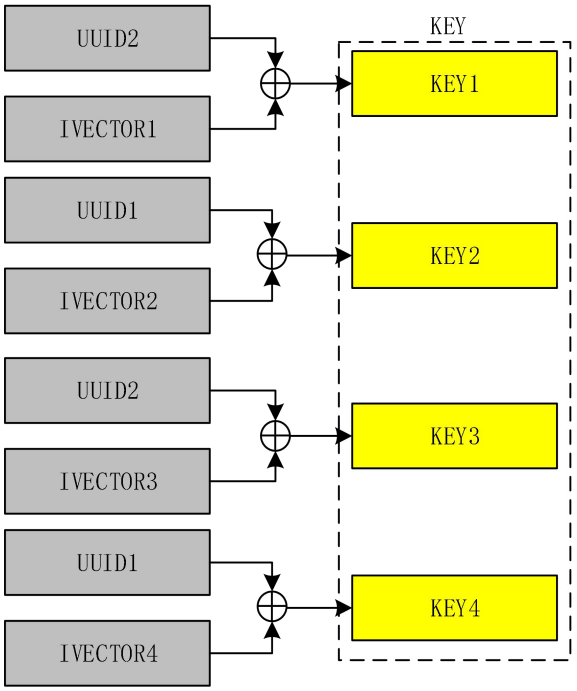
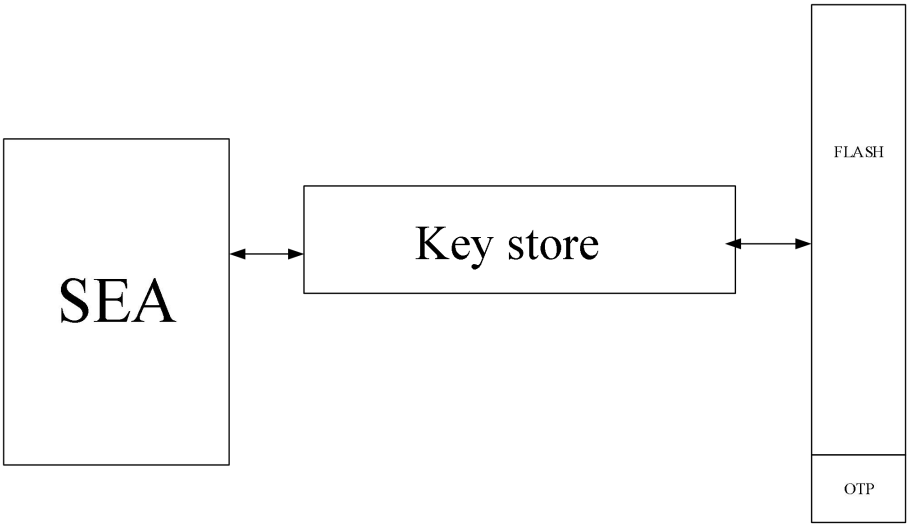


Figure 32-8 Key Generation Process

32.3 KeyStore



The physical scope of KeyStore only includes the stacked area. Any use of the encryption/decryption engine will utilize the encrypted storage function, adopting a 256-bit private encryption algorithm.

There is a special NVR space in the FLASH of the chip's SPE area. This area has an access control mechanism and can only be accessed by the encryption/decryption engine for storing Random and UUID. Both Random and UUID are 128-bit random numbers programmed and solidified in the NVR during the chip production process, generated by the

hardware true random number generator to ensure the uniqueness of the UUID. HUK is derived from Random and UUID, and during operation, the encryption/decryption engine only accesses the derived HUK.

The 256*2 bits ECDSA public key used for secure boot verification of the encryption/decryption firmware is stored in the NVR and programmed and solidified through a dedicated process during chip production.

- The chip supports remote upgrade through the UART port. The firmware to be upgraded is encrypted (session key AES-128 GCM) and transmitted to the NSPE shared SRAM of the chip via UART.
- The NSPE application program of the chip obtains superuser mode permission (CO) and sends the encrypted ciphertext of the session key (encrypted by the root key) to the SPE through the MAILBOX. The encryption/decryption firmware of the SPE provides session key storage service and stores the session key in the FLASH of the SPE for subsequent firmware upgrade calls.
- After the NSPE application program of the chip passes identity authentication, it applies for firmware upgrade service from the encryption/decryption firmware in the SPE through the MAILBOX:
  - The SPE secure core reads the version number of the firmware to be upgraded in the shared SRAM. If it is less than the version number of the currently running firmware, the SPE returns a firmware upgrade failure status through the MAILBOX and does not perform the firmware upgrade.
  - The SPE secure core uses the decrypted session key (decrypted by the root key) to verify the MAC value of the firmware to be upgraded in the shared SRAM. If the verification fails, the SPE returns a firmware upgrade failure status through the MAILBOX and does not perform the firmware upgrade.
  - The SPE secure core uses the decrypted session key (decrypted by the root key) to verify the MAC value of the firmware to be upgraded in the shared SRAM. If the verification passes, it decrypts the firmware code to be upgraded and saves the firmware version number to the Flash area of the SPE. The data stored in this area can only change from 0 to 1 bit by bit, not vice versa, meaning the next stored firmware version number must be higher than the previous one.
  - The SPE secure core writes the decrypted firmware code into the FLASH code area of the SPE and then jumps to this address to execute the new upgraded firmware.

The special Flash area of the SPE stores firmware version information. The data stored in this area can only change from 0 to 1 bit by bit, not vice versa, meaning the next stored firmware version number must be higher than the previous one. The SPE secure core reads the version number of the firmware to be upgraded in the shared SRAM. If it is less than the version number of the currently running firmware, the SPE returns a firmware upgrade failure status through the MAILBOX and does not perform the firmware upgrade. Authorized rollback is not supported.

All provided sensitive parameters are stored in the SRAM, ROM, and Flash of the SPE. Applications in the NSPE cannot directly read the storage area of the SPE through the application core due to physical isolation. The injection and reset operations of sensitive parameters require the application core of the NSPE to obtain superuser privileges. The original credentials required for superuser privileges are provided by the supplier to the manufacturer. No third party other than these two parties can access or tamper with sensitive parameters. Without obtaining superuser privileges, NSPE applications can only request services from the SPE with normal user privileges, and the sensitive parameters used by the services with normal user privileges are not open to NSPE applications.

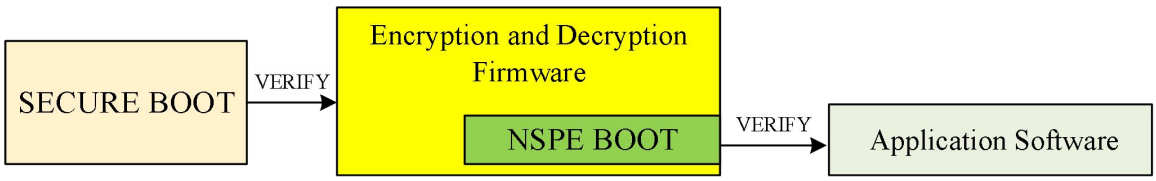
32.4 Secure BOOT

The chip's ROM mask program provides a secure boot function, supporting different security levels and the trusted boot function of the user's boot program.

The ROM in the SPE stores the secure boot code, which is programmed into the ROM during the production

process and cannot be modified. Before the secure boot guides the encryption/decryption firmware in the SPE, it performs signature verification using the ECDSA-256 algorithm (SHA-256, NIST-P256 ECC). The public key is stored in the NVR area of the SPE Flash (only allowing one-time programming), and the digital signature of the encryption/decryption firmware is stored in the SPE Flash, which can only be accessed by the secure core of the SPE. Before the software in the NSPE is loaded, it is guided by the encryption/decryption firmware in the SPE and also performs signature verification using the ECDSA algorithm (SHA-256, NIST-P256 ECC). The public key and digital signature are stored in the SPE Flash, which can only be accessed by the secure core of the SPE.

The block diagrams of the secure boot process of the SPE and the non-secure boot process of the NSPE are shown below:



If the secure boot fails to verify the encryption/decryption firmware, both the secure core and the application core are reset simultaneously; if the NSPE boot fails to verify the application program, the application core is reset. In the Vendor phase, all sensitive parameters are in the production-injected state. From the Vendor phase to the OEM phase, the supplier provides the superuser privilege credentials to the manufacturer. In theory, no third party other than these two parties can access or tamper with sensitive parameters. In the OEM phase, only the manufacturer in superuser mode can add normal users and modify some sensitive parameters. From the OEM phase to the Deployed phase, the manufacturer creates a list of normal users (including normal user credentials), and other users not on the list cannot call security services to access sensitive parameters. In the Deployed phase, normal users can only send security service requests to the secure core through the Mailbox, and only after passing identity authentication can the encryption/decryption firmware in the SPE access sensitive parameters. From the Deployed phase to the RMA phase, the manufacturer should take the device back for scrapping. In the RMA phase, after the manufacturer enters superuser mode, it performs the reset operation of sensitive parameters and locks the Flash.

33 SEA

33.1 Overview

The Security Encryption Accelerator (SEA: SHA, ECC, AES) is a module that provides hardware acceleration for symmetric encryption algorithms (AES) and public key cryptography algorithms (ECDSA, ECDH, etc.). It includes an AES hardware acceleration unit, an ECC hardware acceleration unit, an RSA hardware acceleration unit, a hash algorithm accelerator, and a true random number generator. The above encryption/decryption operation units share the data storage RAM space and registers. Four 1KB dedicated RAMs are used as the data memory of the SEA module.

The symmetric encryption algorithm (AES) hardware acceleration module supports six modes: ECB, CBC, CTR, GCM, CFB128, and OFB. It supports 128-bit finite field multiplication.

The ECC hardware acceleration unit accelerates elliptic curve (EC) operations. Combined with software, it can complete elliptic curve protocols including NIST(P), SEC(p), SEC(k), Brainpool, etc., and signature authentication algorithms including ECDSA (EC Digital Signature Algorithm), ECDH (EC Diffie-Hellman), and their variant algorithms.

The RSA hardware acceleration unit provides hardware acceleration for RSA encryption/decryption operations, which can significantly improve the implementation efficiency of RSA encryption/decryption protocols. This module supports RSA encryption/decryption with word lengths ranging from 32 bits to 1024 bits.

The Hash Algorithm Accelerator (HASH) module supports five operation modes, including SHA-1, SHA-224, SHA-256, SHA384, and SHA512.

The True Random Number Generator (TRNG) uses Fibonacci rings as entropy sources, including four digital combination rings internally, which can generate random numbers at the ns-level scale. It can generate 128-bit random numbers each time.

The Security Encryption Accelerator can obtain or create keys through the Key Store module according to the algorithm.

The firmware RS-SEA based on this module has passed the FIPS140-3 algorithm certification. The list of certified algorithms is as follows. For details, refer to the NIST official website: *Cryptographic Algorithm Validation Program | CSRC (nist.gov)*。

Module	Sub-item	Specific Content	Compliance Standard
AES	ECB	Key Length-128、192、256	SP800-38A
	CBC	Key Length-128、192、256	SP800-38A
	CTR	Key Length-128、192、256	SP800-38A
	CFB128	Key Length-128、192、256	SP800-38A
	OFB	Key Length-128、192、256	SP800-38A
	GMAC	Key Length-128、192、256	SP800-38D
	GCM	Key Length-128、192、256	SP800-38D
DRBG	CTR	Mode：AES-128、AES-192、AES-256	SP800-90A
HASH	SHA-1	Message Length：0~65536 increment 8	FIPS 180-4
	SHA-224	Message Length：0~65536 increment 8	FIPS 180-4
	SHA-256	Message Length：0~65536 increment 8	FIPS 180-4
	SHA-384	Message Length：0~65536 increment 8	FIPS 180-4
	SHA-512	Message Length：0~65536 increment 8	FIPS 180-4
ECDSA	KeyGen	P-192、P-224、P-256、P-384、P-521	FIPS 186-4

	KeyVer	P-192、P-224、P-256、P-384、P-521	FIPS 186-4
	SigGen	P-192、P-224、P-256、P-384、P-521	FIPS 186-4
	SigVer	P-192、P-224、P-256、P-384、P-521	FIPS 186-4

33.2 AES Hardware Acceleration Unit

33.2.1 Features

- Supports key lengths of 128bits/192bits/256bits;
- Supports KeyExp key expansion;
- Supports six data stream processing modes: ECB, CBC, CTR, GCM, CFB128, and OFB;
- Supports multiplication under the GF(2^128) field, which can accelerate the identity authentication process in the GCM algorithm when combined with software;
- Supports AES key security mode. Automatically obtains keys from the Key Store according to configuration. During this process, the keys are not readable by the CPU, and the SRAM data (intermediate data) is not readable.

33.2.2 Time Consumption Information

Hardware Acceleration Unit in Module	**Core0 29M**		**Core1 58M**	
	Time Consumption (Cycles)	Fastest Time (us)	Time Consumption (Cycles)	Fastest Time (us)
128-bit Key Expansion	126	4.3	130	2.2
192-bit Key Expansion	141	4.8	142	2.4
256-bit Key Expansion	162	5.5	165	2.8
128-bit Encryption/Decryption Unit	55	1.9	57	0.96
192-bit Encryption/Decryption Unit	63	2.2	65	1.1
256-bit Encryption/Decryption Unit	71	2.5	71	1.2
128-bit Finite Field Multiplication Unit	128	4.4	130	2.2

Note: Time consumption refers to the number of AHB clock cycles (Cycles) elapsed from the start of the operation to the end of the operation.

33.3 HASH Hardware Acceleration Unit

33.3.1 Features

- Supports acceleration of SHA-1, SHA-224, and SHA-256 secure hash algorithms, processing 512bits of data each time;
- Supports acceleration of SHA384 and SHA512 secure hash algorithms, processing 1024bits of data each time;
- Supports DMA continuous mode, which can continuously write data through the general-purpose DMA and perform automatic operations.

33.3.2 Time Consumption Information

Hardware Acceleration Unit in Module	Core0 29M		Core1 58M	
	Time Consumption (Cycles)	Fastest Time (us)	Time Consumption (Cycles)	Fastest Time (us)
SHA-256	95	3.2	95	1.6
SHA-512	111	3.7	111	1.9

Note: Time consumption refers to the number of AHB clock cycles (Cycles) elapsed from the start of the operation to the end of the operation.

33.4 TRNG True Random Number Generator

33.4.1 Features

- Supports outputting random number bit streams at a frequency of the current system clock frequency/N (1~6536);
- Supports random number randomness detection and error reporting functions;
- Supports the LFSR pseudo-random algorithm;
- Supports the address interleaving randomness enhancement algorithm;
- Supports outputting 128bits of random numbers each time in interrupt mode;
- Supports the TRNG security mode. After generating random numbers and completing automatic operations, they are automatically stored in the pre-configured position of the Key Store, and the CPU cannot read them during this period.

33.4.2 Time Consumption Information

Generated Data Type	Core0 29M		Core1 58M	
	Time Consumption (Cycles)	Fastest Time (us)	Time Consumption (Cycles)	Fastest Time (us)
First set of 128-bit data after enabling 8 ring oscillators	335	7.1	206	3.6
Other sets of 128-bit data	128	4.3	123	2.2

Note: Time consumption refers to the number of AHB clock cycles (Cycles) elapsed from the start of the operation to the generation of 128bits.

33.5 ECC Hardware Acceleration Unit

33.5.1 Features

- Supports modular addition and subtraction operations with a maximum word length of 521bits;
- Supports Montgomery parameter precomputation with a maximum word length of 521bits;
- Supports Montgomery modular multiplication and modular inverse operations with a maximum word length of 521bits;
- Supports domain conversion of coordinates from the integer domain to the Montgomery domain before point addition and point doubling operations;
- Supports two basic point operations: point doubling and point addition with word lengths ranging from 160bits to 521bits;
- Supports domain conversion of coordinates from the Montgomery domain to the integer domain after point operations;
- Combined with software, it can complete elliptic curve protocols including NIST(P), SEC(p), SEC(k), and

Brainpool;

- Combined with software, it can complete signature authentication algorithms including ECDSA (EC Digital Signature Algorithm), ECDH (EC Diffie-Hellman), and their variant algorithms;
- Supports the ECC security mode. During operation, keys are obtained from the Key Store according to the configured address and directly filled into the specified area in the mapping. The CPU cannot read them during this process.

33.5.2 Time Consumption Information

Algorithm	Core0 29M Time Consumption (Reference)	Core1 58M Time Consumption (Reference)
ECDSA_NIST-P256	TBD	12.09ms (Public/Private Key Generation)39.45ms (Signature + Verification)
ECDSA_NIST-P384	TBD	30.73ms (Public/Private Key Generation)99.96ms (Signature + Verification)
ECDSA_Brainpoolp512r1	342.41ms	61.13ms (Public/Private Key Generation)200.57ms (Signature + Verification)
ECDSA_NIST_P521	TBD	73.68ms (Public/Private Key Generation)235.14ms (Signature + Verification)

Note: Time consumption refers to the range of AHB clock cycles (Cycles) elapsed from the start of the operation to the end of the operation.

33.6 RSA Hardware Acceleration Unit

33.6.1 Features

- Supports the RSA encryption/decryption protocol based on Euler's function φ(N)=(P-1)(Q-1);
- Supports modular exponentiation operations with word lengths ranging from 32bits to 576bits;
- Supports modular multiplication operations with word lengths ranging from 32bits to 2047bits;

33.6.2 Time Consumption Information

Hardware Acceleration Unit	Time Consumption (Cycles)
J Parameter Calculation	TBD
H Parameter Calculation	TBD
Modular Multiplication	TBD
Modular Exponentiation	TBD

33.7 Software Configuration Process

Please refer to the relevant chapters of the "MAKSA Library Functions" regarding the firmware RS-SEA v1.0.0 of this module. For the usage process and application solutions, please refer to the "MAKSA Application Notes".

34 AES

34.1 Features

- Supports key lengths of 128bits/192bits/256bits;
- Supports KeyExp key expansion;
- Supports six data stream processing modes: ECB, CBC, CTR, GCM, OFB, and CFB128;
- Supports multiplication under the GF(2^128) field, which can accelerate the identity authentication process in the GCM algorithm when combined with software.

34.2 Time Consumption Information

Hardware Acceleration Unit in Module	Core0 29M
	Time Consumption (Cycles)
128-bit Key Expansion	246
192-bit Key Expansion	282
256-bit Key Expansion	319
128-bit Encryption/Decryption Unit	162
192-bit Encryption/Decryption Unit	186
256-bit Encryption/Decryption Unit	211
128-bit Finite Field Multiplication Unit	128

Note: Time consumption refers to the number of AHB clock cycles (Cycles) elapsed from the start of the operation to the end of the operation.

34.3 Software Configuration Process

Please refer to the relevant chapters of the "MAKSA Library Functions" regarding this module. For the usage process and application solutions, please refer to the "MAKSA Application Notes".

35 CRC

35.1 Overview

Cyclic Redundancy Check (CRC) is mainly used to detect or verify the integrity of data transmission or storage. The CRC calculation unit uses a polynomial generator to generate a CRC code from 8bit/16bit/32bit data.

Through software control, CRC calculation of data from 7816, I2C, UART, and SPI modules can be performed. In addition, verification of Flash integrity can also be performed.

35.2 Features

- Supports 2 channels of CRC. CRC0 supports DMA, while CRC1 does not.
- CRC1 is a private peripheral of Core0, and CRC0 can be configured to be available for both cores.
- Supports 7/8/16/32-bit CRC and arbitrary polynomials.
- Processes data sizes of 8 bits, 16 bits, and 32 bits.
- Programmable CRC initial value.
- Supports input data reversal by byte/half-word/full-word.
- Supports output result reversal and XOR.
- Completes 8-bit CRC operation in 1 cycle and 32-bit operation in 4 cycles.
- CRC0 supports general-purpose DMA function.

35.3 Basic Principle

The basic idea of CRC is to use the transmitted data divided by another number (polynomial) to obtain the remainder, which is the CRC code.

The most basic CRC operation adopts modulo-2 division. The result of each bit division in modulo-2 division does not affect other bits, that is, no borrow is made to the higher bit. Therefore, it is actually a bitwise XOR. Each calculation requires a left shift by 1 bit. When the number of bits of the dividend is less than that of the divisor, the result is the remainder.

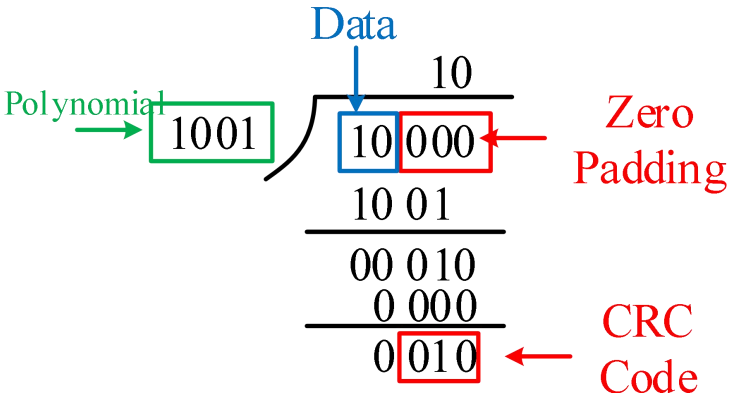


Figure 35-1 Modulo-2 Division

The most basic CRC division has an obvious defect: adding some zeros at the beginning of the data does not affect the result of the final check word, and the complete data is required to calculate the CRC code. In fact, the real CRC calculation is a slight modification based on the original CRC algorithm. Two concepts are added: the first is the initial remainder value, and the second is the result XOR value.

35.4 Function Description

35.4.1 Polynomial

According to the standard, the highest and lowest bits of the polynomial are 1. Common polynomials are as follows:

CRC8 $= X^8 + X^5 + X^4 + 1$

CRC-CCITT $= X^{16} + X^{12} + X^5 + 1$

CRC16 $= X^{16} + X^{15} + X^2 + 1$

CRC32 $= X^{32} + X^{26} + X^{23} + X^{22} + X^{16} + X^{12} + X^{11} + X^{10} + X^8 + X^7 + X^5 + X^4 + X^2 + X^1 + 1$

The bit width in the polynomial refers to the number of binary bits minus 1. For example, the bit width of CRC8 is 8, but the binary number has nine bits. To abbreviate the value of the polynomial, a hexadecimal notation is used. Since the highest bit of the polynomial must be 1, the highest bit is omitted in the abbreviation. For example, the abbreviation of CRC-CCITT is 0x1021, which actually represents 0x11021.

35.4.2 Initial Remainder Value

The initial remainder value is an initial value given to the CRC register at the beginning of calculating the CRC value. In the segmental calculation of CRC, the result of each calculation is used as the initial remainder value of the next calculation. For example, to calculate the 32-bit data 0xddccbbaa, first calculate the CRC value of the lower 8 bits 0xaa, then use this value as the initial remainder value for calculating 0xbb, and so on until the last byte. The final result is used as the CRC value of 0xddccbbaa.

35.4.3 Result XOR Value

The result XOR value is the value that the value of the CRC register is XORed with after the rest of the calculations are completed to obtain the final check value.

35.4.4 DMA Function

CRC supports the general-purpose DMA function. The M2M channel is used to transfer data from SRAM to the CRC_DR register for CRC/checksum calculation. The data transmission direction configured by the general-purpose DMA host must be SRAM→CRC.

A CRC/checksum calculation is performed each time DMA is enabled. The calculation starts when DMA is enabled and ends when DMA is disabled. The enable must be closed by software. After DMA is disabled, the calculation result can be read from the CRC_DR register. During the CRC/checksum calculation process, the calculation method cannot be directly switched. DMA must be disabled first to end the current calculation.

35.4.4.1 DMA CRC Calculation

When the starting address of SRAM is unaligned, the DMA data acquisition of CRC is shown in the following figure. Assuming the starting address of SRAM is 0x1, the DMA transmission length is 10 bytes, and the transmission data width is word. Then, the first received data is {4, 3, 2, 1}, the second data is {8, 7, 6, 5}, the third data is {x, x, x, 9}, and the fourth data is {x, x, x, a}, where x represents invalid data. That is, when the remaining data length is less than the transmission data width, DMA transmits data by byte.

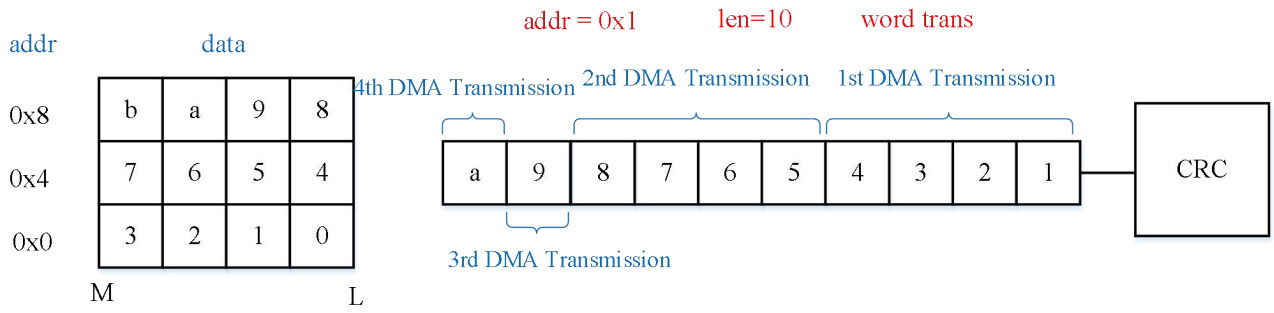


Figure 35-2 DMA Unaligned Transmission

Regardless of whether DMA transmits data by byte, half-word, or word, the CRC calculation is performed by byte, in the order from the lowest byte to the highest byte, and only the valid data part is calculated. For example, in the above figure, the CRC of "1" is calculated first, then the CRC of "2", and so on until the CRC of "a" is calculated last.

35.4.4.2 DMA Checksum Calculation

The checksum calculation is only used when the CHKSUM mode DMA is enabled. The accumulation bit width can be determined as 8bit/16bit/32bit by setting CRC_CR.OPDW, and the calculation result is stored as 32bit in the CRC_DR register. For different accumulation bit widths, there are corresponding restrictions on DMA transmission configuration, as shown in the following table. Software configuration other than the cases in the table is prohibited.

Table 35-1 DMA Checksum Transmission Configuration Restrictions

Accumulation Bit Width Selection	DMA Transmission Configuration		
	Transmission Bit Width of SRAM→CRC	Lower Two Bits of SRAM Starting Address	Transmission Length
8 bit	8/16/32 bit	11/10/01/00	n byte
16 bit	16/32 bit	10/00	2*n byte
32 bit	32 bit	00	4*n byte

Note: n is a positive integer, and the transmission length does not exceed the SRAM capacity.

The result of the DMA checksum calculation is stored as 32bit in the CRC_DR register. It supports setting the initial calculation value, output reversal and XOR, and input data reversal. Among them:

Output reversal: reverses the 32bit result;

Input data reversal: if the valid data bit width is not a multiple of the reversed bit width, it will not be executed. For details, refer to the "Input Data Reversal" chapter.

35.4.5 Input Data Reversal

Supports input data reversal by byte, half-word, or word. For example, if the input data is 0x1A2B3C4D:

- Reversal by byte: the data becomes 0x58D43CB2; // [31:0] -> [24,…,31,16,…,23,8..,15,0,…,7]
- Reversal by half-word: the data becomes 0xD458B23C; // [31:0] -> [16,15…...,31,0,1….15]
- Reversal by word: the data becomes 0xB23CD458. // [31:0] -> [0,1,2,3,……,31]

If the valid data bit width is not a multiple of the reversed bit width, the reversal will not be executed. For example, for valid data data[7:0], setting to perform reversal by half-word/word will not execute the reversal operation.

When DMA transmits data, there may be cases where reversal is not performed. As shown in the following figure,

the starting address of SRAM is 0x0, the transmission length is 10 bytes, and DMA transmits with 32bit bit width. Then, 4 DMA transmissions are required, which are "0~3", "4~7", "8", and "9" in sequence.

- Reversal by byte: all transmitted data is reversed successfully;
- Reversal by half-word: all transmitted data is reversed successfully, where "8~9" are combined to perform half-word reversal;
- Reversal by word: only "0~3" and "4~7" are reversed successfully.

Therefore, when configuring input data reversal by byte/half-word/word in DMA mode, it is necessary to ensure that all DMA transmitted data can be reversed. If improper software operation causes the tail data reversal to fail, the calculation result will be incorrect. At the same time, software is prohibited from configuring the DMA transmission bit width to be smaller than the reversed bit width. For example, DMA transmits by byte and data input is reversed by half-word. The above illegal configuration will lead to incorrect calculation results.

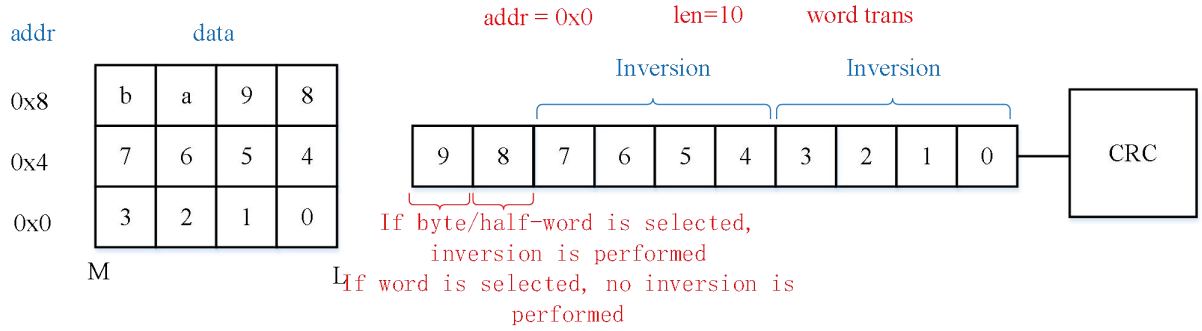


Figure 35-3 Input Data Reversal

35.5 Registers

Base Addresses of Registers in the CRC Module

Module Name	Core0 Base Address	Core1 Base Address
CRC0	0x2007_0000	0xA007_0000
CRC1	0x2011_4000	--

CRC Interface Register Offset Address

Register Name	Address Offset	Description
CRCx_DR	Offset+0x0	CRCx Data Register (x=0,1, same below)
CRCx_STA	Offset+0x4	CRC Status Register
CRCx_CR	Offset+0x8	CRC Control Register
CRCx_INIT	Offset+0xC	CRC Initial Value Register
CRCx_POL	Offset+0x10	CRC Polynomial Register
CRCx_XOR	Offset+0x14	CRC Output XOR Register
CRC0_DMA_CTL	Offset+0x18	CRC0 DMA Control Register

35.5.1 CRCx_DR（0x00）(x=0,1, same below)

CRCx Data Register

Offset Address = 0x00

Bits	Name	Description	R/W	Reset Value
31:0	DR	CRC Calculation:	RW	0x0

		This register is used as the data input register and saves the CRC calculation result after the operation. 1. As input: 1) DMA not enabled: If OPDW==00, written data DR[31:0] is valid; If OPDW==01, written data DR[15:0] is valid; If OPDW==1x, written data DR[7:0] is valid. Each data write starts a CRC calculation. 2) DMA enabled: The valid bits of DR are determined based on the bus wdata valid bit signal. Each DMA enable performs one CRC / checksum calculation. Starts with DMA enable and ends with DMA enable disable. 2. Saving result: If 7-bit polynomial, result is saved in DR[6:0]; If 8-bit polynomial, result is saved in DR[7:0]; If 16-bit polynomial, result is saved in DR[15:0]; If 32-bit polynomial, result is saved in DR[31:0]; If checksum calculation, result is saved in DR[31:0]. Note: CRC calculates in byte form, first calculating DR[7:0], then DR[15:8] until the last valid byte.		

35.5.2 CRCx_STA（0x04）

CRCx Status Register

Offset Address = 0x04

Bits	Name	Description	R/W	Reset Value
31:3	--	Reserved	RO	0x0
0	DONE	Operation Completion Flag 1: CRC/Checksum operation completed 0: CRC/Checksum operation in progress When using DMA, the CRC/checksum operation is considered complete only after all DMA data has been processed.	RO	0x1

35.5.3 CRCx_CR（0x08）

CRCx Control Register, supports bitband operation

Offset Address = 0x08

Bits	Name	Description	R/W	Reset Value
31:9	--	Reserved	RO	0x0
7:6	OPDW	Calculation Data Width Selection (Operation by data width) 00: 32-bit data calculation 01: 16-bit data calculation	RW	0x0

		1x: 8-bit data calculation Note: When not enabling DMA, selects the calculation data width for writing to the register for CRC calculation; When CRC DMA is enabled, the hardware adapts the CRC calculation data width based on the bus wdata write valid bit signal, configuration is invalid; When CHKSUM DMA is enabled, selects the checksum accumulation width. Cannot be configured during checksum operation, configuration is invalid.		
5:4	REV_IN	Reverse Input Data 00: No reversal 01: Bit reversal by byte 10: Bit reversal by half-word 11: Bit reversal by word Example: Calculation data is 0x1A2B3C4D REV_IN==00, use data 0x1A2B3C4D directly for calculation; REV_IN==01, change data to 0x58D43CB2, then calculate; REV_IN==10, change data to 0xD458B23C, then calculate; REV_IN==11, change data to 0xB23CD458, then calculate. Reversal is not performed if the effective data width is not a multiple of the reversal width (see "Input Data Reversal" chapter). Configuration is invalid during operation.	RW	0x0
3	REV_OUT	Reverse Output Data 0: Output not reversed 1: Output reversed by CRC/checksum result bit length Example: CRC result is 0xAABB REV_OUT==1, then output result is 0xDD55; REV_OUT==0, then output result is 0xAABB. Checksum calculation reverses by 32 bits. Configuration is invalid during operation.	RW	0x0
2	XOR	Output Data Bitwise XOR Enable 0: No operation on output data 1: Output data is XORed with the value in the CRC_XOR register Configuration is invalid during operation.	RW	0x0
1:0	POLYSIZE	Polynomial Size 00: 32-bit polynomial 01: 16-bit polynomial 10: 8-bit polynomial 11: 7-bit polynomial Configuration is invalid during CRC operation. This configuration bit is not used during checksum calculation.	RW	0x0

35.5.4 CRCx_INIT（0x0C）

CRCx Initial Value Register

Offset Address = 0x0C

Bits	Name	Description	R/W	Reset Value
31:0	INIT	CRC Initial Value. If 32-bit polynomial size is selected, INIT[31:0] is valid; If 16-bit polynomial size is selected, INIT[15:0] is valid; If 8-bit polynomial size is selected, INIT[7:0] is valid; If 7-bit polynomial size is selected, INIT[6:0] is valid. Configuration is invalid during operation. CRC Calculation: When using the configuration register for CRC calculation, if this register is not configured before the CRC calculation, the remainder initial value will use the result of the previous data calculation. When using DMA, the value of this register is automatically used as the remainder initial value, no need to reconfigure. If the remainder initial value needs to be changed, it must be configured before calculation. Checksum Calculation: Used as the initial value for accumulation.	RW	0x0

35.5.5 CRCx_POL（0x10）

CRCx Polynomial Register

Offset Address = 0x10

Bits	Name	Description	R/W	Reset Value
31:0	POLY	Polynomial used for CRC calculation. If the polynomial is less than 32 bits, the least significant bits are used. The value written for the polynomial ignores the highest bit value. For example, the polynomial for CRC-CCITT is x^16 + x^12 + x^5 + 1, which is 0x11021 in hex. Ignoring the highest bit, the value written is 0x1021. If 32-bit polynomial size is selected, POLY [31:0] is valid; If 16-bit polynomial size is selected, POLY [15:0] is valid; If 8-bit polynomial size is selected, POLY [7:0] is valid; If 7-bit polynomial size is selected, POLY [6:0] is valid. Configuration is invalid during operation.	RW	0x0

35.5.6 CRCx_XOR（0x14）

CRCx Output XOR Register

Offset Address = 0x14

Bits	Name	Description	R/W	Reset Value
31:0	XOR	CRC Operation Result XOR Register. When CRC_CR.XOR is 1, the CRC/checksum result is XORed with the data in this register before output. If 32-bit polynomial size is selected, XOR [31:0] is valid; If 16-bit polynomial size is selected, XOR [15:0] is valid; If 8-bit polynomial size is selected, XOR [7:0] is valid; If 7-bit polynomial size is selected, XOR [6:0] is valid. Configuration is invalid during operation.	RW	0x0

35.5.7 CRC0_DMA_CTL（0x18）

CRC0 DMA Control Register

Offset Address = 0x18

Bits	Name	Description	R/W	Reset Value
31:2	--	Reserved	RO	0x0
1	DMA_CAL_MOD	DMA Data Calculation Mode Selection 0: DMA transfers data for CRC calculation 1: DMA transfers data for Checksum calculation Configuration is invalid after DMA is enabled.	RW	0x0
0	DMA_EN	DMA Enable Bit 0: Disable 1: Enable	RW	0x0

35.6 Software Configuration Process and Application Description

35.6.1 Performing CRC Calculation through Software Operation

35.6.1.1 Calculating Single Data

- Without Using the Flag Bit

```
CRCx_INIT = 0xxxxx; // The CRCx_INIT register must be reconfigured for new data calculation
CRCx_CR = 0xxxxx; // Configure relevant controls: OPDW, REV_IN, REV_OUT, XOR, POLYSIZE
CRCx_POL = 0xxxxx; // Configure the polynomial
CRCx_DR = 0xxxxx; // Configure the calculation data — data must be configured last
// Wait for the calculation to complete
// No waiting is required for 8-bit data calculation
// NOP; // Wait for 1 cycle for 16-bit data calculation
// NOP; NOP; NOP; // Wait for 3 cycles for 32-bit data calculation;
CRCx_result = CRCx_DR[x:0]; // Read the CRC calculation result
// Read the result according to the configured POLYSIZE
// For example, if CRC7 is configured, crc_result = CRCx_DR[6:0];
// CRC8, crc_result = CRCx_DR[7:0];
// CRC16, crc_result = CRCx_DR[15:0];
// CRC32, crc_result = CRCx_DR[31:0];
```

- Using the Flag Bit

```
CRCx_INIT = 0xxxxx; // The CRCx_INIT register must be reconfigured for new data calculation
CRCx_CR = 0xxxxx; // Configure relevant controls: OPDW, REV_IN, REV_OUT, XOR, POLYSIZE
CRCx_POL = 0xxxxx; // Configure the polynomial
CRCx_DR = 0xxxxx; // Configure the calculation data — data must be configured last
while(!(CRCx_STA & 0x1)); // Wait for the flag bit
crc_result = CRCx_DR[x:0]; // Read the CRC calculation result
// Read the result according to the configured POLYSIZE
// For example, if CRC7 is configured, crc_result = CRCx_DR[6:0];
// CRC8, crc_result = CRCx_DR[7:0];
// CRC16, crc_result = CRCx_DR[15:0];
// CRC32, crc_result = CRCx_DR[31:0];
```

35.6.1.2 Calculating Multiple Data

- Example: Calculating 6-byte data:

```
CRCx_INIT = 0xxxxx; // The CRCx_INIT register must be reconfigured for new data calculation
CRCx_CR = 0xxxxx; // Configure relevant controls: OPDW, REV_IN, REV_OUT, XOR, POLYSIZE
CRCx_POL = 0xxxxx; // Configure the polynomial
// OPDW = 2'b00; First calculate 4-byte data
CRCx_DR = 0xxxxx; // Configure the calculation data and write the upper 4-byte data
CRCx_CR = 0xxxxx; // Configure relevant controls: OPDW, REV_IN, REV_OUT, XOR, POLYSIZE
// OPDW = 2'b01; Then calculate 2-byte data
CRCx_DR = 0xxxxx; // Configure the calculation data and write the last 2-byte data
NOP(); // Wait for the calculation to complete
crc_result = CRCx_DR[x:0]; // Read the CRC calculation result
// Read the result according to the configured POLYSIZE
// For example, if CRC7 is configured, crc_result = CRCx_DR[6:0];
// CRC8, crc_result = CRCx_DR[7:0];
// CRC16, crc_result = CRCx_DR[15:0];
// CRC32, crc_result = CRCx_DR[31:0];
```

- Example: Calculating 8-byte data:

```
CRCx_INIT = 0xxxxx; // The CRCx_INIT register must be reconfigured for new data calculation
CRCx_CR = 0xxxxx; // Configure relevant controls: OPDW, REV_IN, REV_OUT, XOR, POLYSIZE
CRCx_POL = 0xxxxx; // Configure the polynomial
// OPDW = 2'b00; Calculate 4-byte data
CRCx_DR = 0xxxxx; // Configure the calculation data and write the upper 4-byte data
NOP();
CRCx_DR = 0xxxxx; // Configure the calculation data and write the lower 4-byte data
NOP();
NOP();
NOP();
```

crc_result = CRCx_DR[x:0]; // Read the CRC calculation result
// Read the result according to the configured POLYSIZE
// For example, if CRC7 is configured, crc_result = CRCx_DR[6:0];
// CRC8, crc_result = CRCx_DR[7:0];
// CRC16, crc_result = CRCx_DR[15:0];
// CRC32, crc_result = CRCx_DR[31:0];

35.6.2 Performing Calculation Using DMA

35.6.2.1 DMA CRC Calculation

- DMA CRC Calculation of a Segment of Data

CRC0_INIT = 0xxxxx; // The CRC0_INIT register must be reconfigured for new data calculation
CRC0_CR = 0xxxxx; // Configure relevant controls: OPDW, REV_IN, REV_OUT, XOR, POLYSIZE
CRC0_POL = 0xxxxx; // Configure the polynomial
// General-purpose DMA register configuration
GDMA_M1SADR = 0xxxxx; // Configure the base address of the DMA data source, i.e., the starting address of the data to be calculated
GDMA_M1DADR = 0xxxxx; // Configure the base address of the DMA target, i.e., configure it as the address of the CRC0_DR register
GDMA_M1NDTR = 0xxxxx; // Configure the DMA transmission length
GDMA_MIE = 0xxxxx; // Configure the DMA CRC M2M channel transmission completion interrupt enable, etc.
GDMA_M1CTRL = 0xxxxx; // Configure the DMA enable, source transmission bit width, target transmission bit width, etc.
while(1); // Wait for the DMA interrupt
---- Enter the interrupt program
// Check the DMA interrupt flag. If the CRC M2M transmission completion flag is set, then
CRC0_DMA_CTL = 0x0; // Software disables DMA CRC calculation enable
crc_result = CRC0_DR[x:0]; // Read the CRC calculation result
// Read the result according to the configured POLYSIZE
// For example, if CRC7 is configured, crc_result = CRC0_DR[6:0];
// CRC8, crc_result = CRC0_DR[7:0];
// CRC16, crc_result = CRC0_DR[15:0];
// CRC32, crc_result = CRC0_DR[31:0];

- DMA CRC Calculation of Multiple Segments of Data

CRC0_INIT = 0xxxxx; // The CRC0_INIT register must be reconfigured for new data calculation
CRC0_CR = 0xxxxx; // Configure relevant controls: OPDW, REV_IN, REV_OUT, XOR, POLYSIZE
CRC0_POL = 0xxxxx; // Configure the polynomial
// General-purpose DMA register configuration
GDMA_M1SADR = 0xxxxx; // Configure the base address of the DMA data source, i.e., the starting address of the data to be calculated
GDMA_M1DADR = 0xxxxx; // Configure the base address of the DMA target, i.e., configure it as the address of the CRC0_DR register
GDMA_M1NDTR = 0xxxxx; // Configure the DMA transmission length

GDMA_MIE = 0xxxxx; // Configure the DMA CRC M2M channel transmission completion interrupt enable, etc.

GDMA_M1CTRL = 0xxxxx; // Configure the DMA enable, source transmission bit width, target transmission bit width, etc.

while(1); // Wait for the DMA transmission completion interrupt enable

---- Enter the interrupt program

// Check the DMA interrupt flag. If the CRC M2M transmission completion flag is set, repeat the above "general-purpose DMA register configuration operation"

while(1); // Wait for the DMA transmission completion interrupt enable

---- Enter the interrupt program

// Check the DMA interrupt flag. If the CRC M2M transmission completion flag is set, then

CRC0_DMA_CTL = 0x0; // Software disables DMA CRC calculation enable

crc_result = CRC0_DR[x:0]; // Read the CRC calculation result

// Read the result according to the configured POLYSIZE

// For example, if CRC7 is configured, crc_result = CRC0_DR[6:0];

// CRC8, crc_result = CRC0_DR[7:0];

// CRC16, crc_result = CRC0_DR[15:0];

// CRC32, crc_result = CRC0_DR[31:0];

35.6.2.2 DMA Checksum Calculation

// Except for the following operations, refer to the DMA CRC calculation operation process for other operations.

// Configure CRC0_DMA_CTL as follows to start the checksum calculation

CRC0_DMA_CTL = 0x3; // Configure DMA checksum calculation enable

……

chksum_result = CRC0_DR[31:0]; // Read the checksum calculation result. The checksum result bit width is always 32bit.

35.6.2.3 Disabling DMA During Calculation

After disabling DMA, the CRC/checksum operation will stop immediately.

36 DSP

This module is a Digital Signal Processing (DSP) module, including FFT operations in floating-point format, conversion units between integers and floating-point numbers, basic floating-point operation units, CORDIC algorithm for calculating sine, cosine and root mean square (RMS), IIR acceleration operations, FIR acceleration operations, linear interpolation operations, and Lagrange interpolation operations.

All floating-point numbers mentioned here are single-precision floating-point numbers (23 valid bits after the decimal point in binary).

36.1 Features

- Supports conversion between integers and floating-point numbers;
- Supports floating-point addition, subtraction, multiplication, and division;
- Supports single butterfly operation (complex number) and DMA continuous butterfly operations;
- Supports the entire process of radix-2 FFT, with configurable points of 64, 128, 256, 512, and 1024, all supporting the use of built-in parameter tables;
- Supports automatic BITREV data transfer, with configurable points of 4, 8, 16, 32, 64, 128, 256, 512, and 1024;
- Supports sine and cosine calculations;
- Supports RMS calculation;
- Supports arctangent calculation;
- Supports single IIR calculation and IIR DMA operations;
- Supports FIR filtering operations;
- Supports linear interpolation operations;
- Supports Lagrange interpolation operations.

36.2 Basic Principles of the Calculation Engine

36.2.1 Single-Precision Floating-Point Representation

The figure above shows the floating-point representation method. The value of a floating-point number = $\pm 2^{n}(1+f)$, where n is the exponent value and f is the fractional part.

The sign bit indicates the positive or negative value of the floating-point number: 0 for positive and 1 for negative.

The exponent part is represented by biased notation, that is, the highest bit of the 8-bit "exponent bias" is the exponent sign bit, and the lower 7 bits are the actual exponent value. For example, the exponent value 1 is represented as 1+127=128 in biased notation; the exponent value -10 is represented as -10+127=117.

The mantissa fractional part represents the number after the decimal point.

36.2.2 Special Values

Table 36-1 Floating-Point Representation of Special Values

Category	Sign Bit (1bit)	Exponent Bias (8bit)	Mantissa Fractional Part (23bit)
Zero	0/1	0	0
Denormal	0/1	0	Non-zero
Infinity	0/1	255	0

NaN	0/1	255	Non-zero
Normal	0/1	1~254	Arbitrary

The table above lists some special values in floating-point representation. When performing floating-point operations below, it is necessary to determine whether the input and output data are special values, perform special processing, and set flag bits.

36.2.3 Floating-Point Rounding Processing

When the bit width of the mantissa fractional part exceeds the actually representable bit width during floating-point operations, rounding processing is required.

First, define the following concepts:

As shown in the figure above, when truncating m-bit data to n-bit:

- Rounding bit: The (n+1)-th bit starting from the highest bit on the left;
- Sticky bit: All bits after the (n+2)-th bit starting from the highest bit on the left.

36.2.4 IEEE754 Standard Rounding Modes

The IEEE754 standard defines 4 rounding methods:

- Nearest: Round to the nearest value; round up if the rounding bit is 1 and the sticky bit is non-zero.
- +∞: For positive numbers, round up if the rounding bit is 1 or the sticky bit is non-zero; otherwise, no carry.
- -∞: For negative numbers, round up if the rounding bit is 1 or the sticky bit is non-zero; otherwise, no carry.
- Zero: Directly discard the rounding bit and sticky bit.

36.2.5 Rounding Modes of This Chip

This chip supports 6 rounding methods, selectable by configuration:

- When rnd=000: Nearest rounding;
- When rnd=001: Zero rounding;
- When rnd=010: +∞ rounding;
- When rnd=011: -∞ rounding;
- When rnd=100: Up rounding (always carry);
- When rnd=101: Away rounding (direct discard).

36.2.6 Principle of Integer to Floating-Point Conversion

The following briefly describes the idea and principle of integer to floating-point conversion:

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

- Sign bit:
  - The highest bit is the floating-point sign bit: fp_sign = bit(31);
  - If the original data is negative, convert it to two's complement before further processing.
- Exponent:
  - Starting from the highest bit, find the first "1" and record its position as n. The exponent value is n (27 in the

table above);

- According to the encoding rules, the exponent representation is fp_exp = n + 127.

- Fraction:
  - If n > 23: Starting from bit n-1, truncate 24 bits as m; perform rounding on the remaining bits of the original data to determine the least significant bit of m. m is the fractional part;
  - If n = 23: Starting from bit n-1, truncate 24 bits directly as the fractional part;
  - If n < 23: Starting from bit n-1, truncate all remaining bits and pad zeros at the lower bits as the fractional part.

36.2.7 Principle of Floating-Point to Integer Conversion

- Determine whether the original data is a special value (0, NaN, +∞, -∞);
- Based on the exponent part e of the original data, determine if the conversion result overflows. If e > 32 + 127, the conversion result overflows; otherwise, proceed to the next step;
- Extract the 23-bit fractional part m, pad "1" at the highest bit to form a 24-bit fraction m;
- Left-shift m by e bits, pad "0" at the higher bits ((32-e) bits) and lower bits (e bits) to get 56-bit data;
- Right-shift the result by 24 bits to get the final integer.

36.2.8 Principle of Floating-Point Multiplication

- Multiplication of special data:
  - NaN multiplied by any number is NaN;
  - 0 multiplied by infinity is NaN;
  - 0 multiplied by any other number is 0.
- Multiplication of normal data:
  - XOR the sign bits to get the result sign bit;
  - Add the exponent parts and subtract 127（$e = e_0 + e_1 - 127$）;
  - Expand each mantissa part by 1 bit (total 24 bits) and multiply to get a 48-bit data m. Check for overflow: if overflow occurs, set e = e + 1, check if the exponent e overflows, and right-shift m by 3 bits; otherwise, right-shift m by 2 bits. Take the upper 23 bits as the mantissa result.
- Overflow handling: Supports 4 overflow handling methods specified by IEEE754, configurable via registers.

36.2.9 Principle of Floating-Point Addition

- Check for special data: NaN added to any number is NaN;
- Addition of normal numbers:
  - Use the sign bit of the number with the larger absolute value as the output sign bit;
  - Compare the two exponent parts, calculate the difference F = larger exponent (e_0) − smaller exponent (e_1);
  - Right-shift the mantissa part of the number with the smaller exponent by F bits, then expand both mantissa parts by 1 bit and add;
  - Exponent: If overflow occurs, set e = e_0 + 1; otherwise, e = e_0;
  - Remove the upper 2 expanded bits of the mantissa part after addition. If overflow occurs, right-shift the mantissa part by 1 bit (equivalent to dividing by 2) to get the result mantissa.

36.2.10 Principle of Butterfly Operation

$$X_1^{'}(k) = X_1(k) + W_N^k X_2(k)$$
$$X_2^{'}(k) = X_1(k) - W_N^k X_2(k)$$

Where $X_1(k)$ and X_2(k)are inputs, and $X_1^{'}(k)$ and $X_2^{'}(k)$ are outputs of the butterfly operation.

If the original data are complex numbers:

$$X_1(\mathrm{k}) = X_{1r}(k) + jX_{1i}(k)$$
$$X_2(\mathrm{k}) = X_{2r}(k) + jX_{2i}(k)$$
$$W_N^k = W_{Nr}^k + jW_{Ni}^k$$
$$X_1^{'}(k) = X_{1r}^{'}(k) + jX_{1i}^{'}$$

Thus:

$$X_1^{'}(k) = X_1(k) + W_N^k X_2(k)$$
$$= (X_{1r}(k) + jX_{1i}(k)) + \left(W_{Nr}^k + jW_{Ni}^k\right) * (X_{2r}(k) + jX_{2i}(k))$$

Extract the real and imaginary parts:

$$X_{1r}^{'} = X_{1r} + W_{Nr}^k X_{2r} + W_{Ni}^k X_{2i}$$
$$X_{1i}^{'} = X_{1i} + W_{Nr}^k X_{2i} - W_{Ni}^k X_{2r}$$

Similarly:

$$X_{2r}^{'} = X_{1r} - W_{Nr}^k X_{2r} - W_{Ni}^k X_{2i}$$
$$X_{2i}^{'} = X_{1i} - W_{Nr}^k X_{2i} + W_{Ni}^k X_{2r}$$

36.2.11 Principle of IIR Filter

IIR (Infinite Impulse Response) filter is one of the most common types of linear digital filters. At a given time, the output of an IIR filter depends on its input and previously calculated input values.

IIR filters are recursive because the difference equation includes feedback. Therefore, their settling time is longer than that of FIR filters, and the impulse response may have infinite width. Attention should be paid to the stability of the filter.

The transfer function (z-domain) of an IIR filter is as follows:

$$H(z) = \frac{\sum_{k=0}^{M} b_k z^{-k}}{a_0 + \sum_{k=1}^{N} a_k z^{-k}} = \frac{Y(z)}{X(z)}$$

The time-domain calculation formula of an IIR filter (direct I-type) is as follows:

$$y[n] = \sum_{k=0}^{M} b_k x[n-k] - \sum_{k=1}^{N} a_k y[n-k]$$

For a second-order IIR filter (where M and N in the above formula are both 2, and generally designed as 1), the formula becomes:

$$y[n] = b_0 \times x(n) + b_1 \times x(n-1) + b_2 \times x(n-2) - a_1 \times y(n-1) - a_2 \times y(n-2)$$

Where x(n) is the current input data, x(n-1) is the previous input data (lag by one unit in time), x(n-2) is the input data before the previous one (lag by two units in time); y(n) is the current output data, y(n-1) is the previous output data (lag by one unit in time), and y(n-2) is the output data before the previous one (lag by two units in time).

36.2.12 Principle of FIR Filter

FIR (Finite Impulse Response) filter is the abbreviation of non-recursive filter. An FIR filter with constant coefficients is an LTI (Linear Time-Invariant) digital filter. The finite impulse response means there is no feedback in the filter. The output of an FIR filter of length N is given by a finite convolution sum with the input time sequence x[n], as

follows:

In the formula above, a[k] is the filter coefficient, and x[n-k] is x[n] delayed by k cycles.

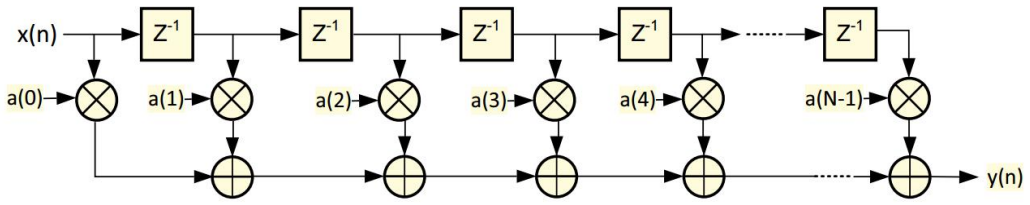


Figure 36-1 Diagram of Conventional Tapped Delay Line FIR Filter

The formula above represents an (N-1)-th order FIR filter with N taps (coefficients). Therefore, it consists of N multipliers and (N-1) adders. The input signal is time-varying. The final output of the FIR filter is the sum of the inputs at various times multiplied by the corresponding weights (coefficients).

Taking a 4th-order FIR filter as an example, there are 5 coefficients. Implemented in a serial manner, it requires 5 multiplier calls and 4 adder calls to complete one calculation:y[n] = a0*x[n] + a1*x[n-1] + a2*x[n-2]+ a3*x[n-3]+ a4*x[n-4]

36.2.13 Linear Interpolation

Given n points: $(1, y_1), \ldots, (n, y_n)$, find y(i), where x is the abscissa of the i-th output data.

Formula: $y(i) = y_{[x]} + (x - [x]) * (y_{[x]+1} - y_{[x]})$, [x] = floor(x) (take the floor of x)

Calculate output data point by point, increasing the interpolation position by step each cycle:

```
for i=1：1：out_len
   n=floor(t);
   y[i]= x[n]+ (t-n) * (x[n+1]-x[n]);
   t=t+step;
end
```

36.2.14 Lagrange Interpolation

Given n points: $(1, y_1), \ldots, (n, y_n)$, find y(i), where x is the abscissa of the i-th output data.

3-point Lagrange interpolation uses a quadratic curve for each interval, with the formula:

$$cur = round(x);$$

$$y(i) = \frac{x - cur}{2} * \left[(x - (cur + 1)) * y_{cur-1} + (x - (cur - 1) * y_{cur+1}\right] - (x - (cur - 1)) * (x - (cur + 1)) * y_{cur}$$

Calculate output data point by point, increasing the interpolation position by step each cycle:

```
for i=1：1：out_len
    n=floor(t);
    y[i]=(t-n)*0.5*((t-n-1)*x[n-1]+(t-n+1)*x[n+1]) - (t-n+1)*(t-n-1)*x[n]
    t=t+step;
end
```

36.3 Operation Instructions

36.3.1 Integer to Floating-Point (int2fp/int2fp_dma)

36.3.1.1 Single Data Conversion

Figure 36-2 Integer to Floating-Point

This mode converts a 32-bit (or 24-bit/16-bit/8-bit) signed integer input from a register into a 32-bit single-precision floating-point format, outputs it to a register, and is readable by the CPU.

Supports normalization of the converted floating-point number, i.e., dividing by 2^n. n can be configured as 0~32; when configured as 0, no normalization is performed.

By default, the input register integer is a 32-bit signed number. For special applications, if the input integer is a 24-bit signed number, configure MAC_CTL0[4:0] = 5'd1 to select this mode.

Software Operation Flow:

- Configure MAC_CTL0[4:0] = 5'd1 to select the integer-to-floating-point single-point mode;
- Configure MAC_CTL0[9:7] to select the floating-point rounding mode;
- Configure MAC_CTL0[14:10] to select the normalization order;
- For 24-bit input data, configure MAC_CTL[22:21] = 1; for other special applications, configure the I2F_PRE_EN register according to the number of valid bytes;
- Write the original integer to be converted into the MAC_IN0 register;
- Read the MAC_OUT0 register to obtain the converted floating-point value.

After configuring the mode, the conversion starts as soon as the MAC_IN0 register is configured. The calculation result is stored in MAC_OUT0. Therefore, the converted result can be obtained by directly reading the result register after configuring the original value.

Note: If the software needs to perform continuous conversions, the above mode configuration only needs to be executed once. Before switching modes, each write to the MAC_IN0 register will trigger the conversion.

36.3.1.2 Continuous Conversion of Multiple Data

This mode supports converting a segment of integers in SRAM into floating-point numbers and writing the results back to SRAM. Note that the original data stored in SRAM are signed integers. For special applications, if 24-bit integers occupy 4 bytes, valid data are stored in the lower 3 bytes, and the highest byte is invalid, configure MAC_CTL.I2F_PRE_EN = 1; for other special applications, configure the I2F_PRE_EN register according to the number of valid bytes.

The target address can be configured to be the same as the source data address to overwrite the original data, thereby saving SRAM space. Otherwise, configure the target address to be different from the source data address to write the converted data to another area in SRAM.

Software Operation Flow:

- Configure MAC_CTL0[4:0] = 5'd2 to select the integer-to-floating-point DMA mode;
- Configure MAC_CTL0[9:7] to select the floating-point rounding mode;
- Configure MAC_CTL0[14:10] to select the normalization order;

- Configure the DMA_SRBADR register to select the DMA start address of the source data;
- Configure the DMA_TRBADR register to select the DMA start address of the target data;
- Configure the DMA_LEN register to select the DMA length;
- Configure MAC_CTL2[0] = 1 to start the conversion;
- Wait for the DSP_FLG[3] flag to be set, then the CPU can obtain the converted result from SRAM.

After converting the specified length of data, a completion flag is generated. If the interrupt enable signal is configured, a completion interrupt is generated. The software needs to clear the flag bit in the interrupt.

36.3.2 Floating-Point to Integer (fp2int/fp2int_dma)

36.3.2.1 Single Data Conversion

Figure 36-3 Floating-Point to Integer

This mode converts a 32-bit floating-point number input from a register into a 32-bit signed integer format, outputs it to a register, and is readable by the CPU.

Supports scaling the floating-point number by 2^n before conversion. n can be configured as 0~32; when configured as 0, no scaling is performed.

Software Operation Flow:

- Configure MAC_CTL0[4:0] = 5'd3 to select the floating-point-to-integer single-point mode;
- Configure MAC_CTL0[9:7] to select the floating-point rounding mode;
- If scaling of the floating-point number before conversion is required, configure MAC_CTL0[19:15] = n;
- Write the original floating-point number to be converted into the MAC_IN0 register;
- Read the MAC_OUT0 register to obtain the converted integer value.

After configuring the mode, the conversion starts as soon as the MAC_IN0 register is configured. The calculation result is stored in MAC_OUT0. Therefore, the converted result can be obtained by directly reading the result register after configuring the original value.

Note: If the software needs to perform continuous conversions, the above mode configuration only needs to be executed once. Before switching modes, each write to the MAC_IN0 register will trigger the conversion.

36.3.2.2 Continuous Conversion of Multiple Data

This mode supports converting a segment of floating-point numbers in SRAM into integers and writing the results back to SRAM. The target address can be configured to be the same as the source data address to overwrite the original data, thereby saving SRAM space. Otherwise, configure the target address to be different from the source data address to write the converted data to another area in SRAM.

Software Operation Flow:

- Configure MAC_CTL0[4:0] = 5'd4 to select the floating-point-to-integer DMA mode;
- Configure MAC_CTL0[9:7] to select the floating-point rounding mode;
- If scaling of the floating-point number before conversion is required, configure MAC_CTL0[19:15] = n;
- Configure the DMA_SRBADR register to select the DMA start address of the source data;
- Configure the DMA_TRBADR register to select the DMA start address of the target data;
- Configure the DMA_LEN register to select the DMA length;
- Configure MAC_CTL2[0] = 1 to start the conversion;

- Wait for the DSP_FLG[4] flag, then the CPU can obtain the converted result from SRAM.

After converting the specified length of data, a completion flag is generated. If the interrupt enable signal is configured, a completion interrupt is generated. The software needs to clear the flag bit in the interrupt.

36.3.3 Floating-Point Multiplication (fp_mult)

Figure 36-4 Floating-Point Multiplication

This mode performs a floating-point multiplication on the data in the input registers MAC_IN0 and MAC_IN1. It supports the following four modes for input and output data formats:

- Input: Integer, Output: Integer;
- Input: Integer, Output: Floating-point;
- Input: Floating-point, Output: Integer;
- Input: Floating-point, Output: Floating-point.

If the input is an integer, the hardware first converts the integer into a floating-point format, then performs floating-point multiplication. The product obtained from the operation is in floating-point format. If the output integer format is selected, the hardware converts the floating-point product into an integer format.

Software Operation Flow:

- Configure MAC_CTL0[4:0] = 5'd5 to select the floating-point multiplication mode;
- Configure MAC_CTL0[9:7] to select the floating-point rounding mode;
- Configure MAC_CTL0[6:5] to select the input and output data mode;
- Configure the MAC_IN0 and MAC_IN1 registers (multiplier and multiplicand);
- Wait for the DSP_FLG[9] = 1 flag, indicating that the multiplication is completed;
- Read the MAC_OUT0 register to obtain the product.

In this mode, each configuration of the MAC_IN1 register automatically starts the multiplication operation. Therefore, the software should first configure the MAC_IN0 register, then the MAC_IN1 register. Each configuration of the MAC_IN1 register automatically clears the previous flag bit, and the flag bit is set only after the current calculation is completed.

Due to the need for format conversion for different input and output data formats, the time required for one floating-point multiplication operation is related to the mode.

36.3.4 Floating-Point Addition (fp_add)

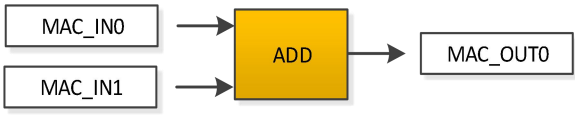


Figure 36-5 Floating-Point Addition

This mode performs a floating-point addition on the data in the input registers MAC_IN0 and MAC_IN1. The data format of the input registers must be floating-point, and the output result is also in floating-point format.

Software Operation Flow:

- Configure MAC_CTL0[4:0] = 5'd6 to select the floating-point addition mode;
- Configure MAC_CTL0[9:7] to select the floating-point rounding mode;
- Configure the MAC_IN0 and MAC_IN1 registers (augend and addend);
- Read the MAC_OUT0 register to obtain the sum.

In this mode, each configuration of the MAC_IN1 register automatically starts the addition operation. Therefore, the software should first configure the MAC_IN0 register, then the MAC_IN1 register.

36.3.5 Floating-Point Subtraction (fp_sub)

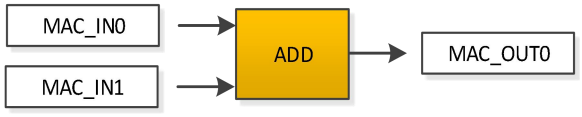


Figure 36-6 Floating-Point Subtraction

This mode performs a floating-point subtraction on the data in the input registers MAC_IN0 and MAC_IN1. The data format of the input registers must be floating-point, and the output result is also in floating-point format.

Software Operation Flow:

- Configure MAC_CTL0[4:0] = 5'd7 to select the floating-point subtraction mode;
- Configure MAC_CTL0[9:7] to select the floating-point rounding mode;
- Configure the MAC_IN0 and MAC_IN1 registers (minuend and subtrahend);
- Read the MAC_OUT0 register to obtain the difference.

In this mode, each configuration of the MAC_IN1 register automatically starts the subtraction operation. Therefore, the software should first configure the MAC_IN0 register, then the MAC_IN1 register.

36.3.6 Floating-Point Multiply-Accumulate Operation (fp_mlad)

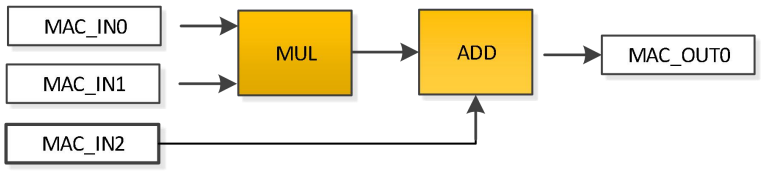


Figure 36-7 Floating-Point Multiply-Accumulate Operation

This mode first performs a floating-point multiplication on the data in the input registers MAC_IN0 and MAC_IN1, then adds the result to MAC_IN2, and finally outputs the result to MAC_OUT0. The data format of the input registers must be floating-point, and the output result is also in floating-point format.

Software Operation Flow:

- Configure MAC_CTL0[4:0] = 5'd8 to select the floating-point multiply-accumulate mode;
- Configure MAC_CTL0[9:7] to select the floating-point rounding mode;
- Configure the MAC_IN0, MAC_IN1, and MAC_IN2 registers (multiplier, multiplicand, and augend);
- Read the MAC_OUT0 register to obtain the multiply-accumulate result.

In this mode, each configuration of the MAC_IN2 register automatically starts the multiply-accumulate operation. Therefore, the software should first configure the MAC_IN0 and MAC_IN1 registers, then the MAC_IN2 register.

36.3.7 Floating-Point Butterfly Operation (Single) (btfy / btfy_dma)

36.3.7.1 Single Data Conversion

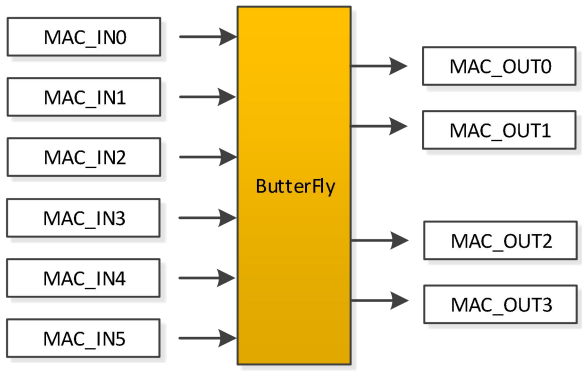


Figure 36-8 Butterfly Operation

This mode performs a single butterfly operation on two original data and one parameter from the input registers, stores the result in the registers for CPU reading. In this mode, the input original data and parameters must be in floating-point format.

$$X'_{1r} = X_{1r} + W^k_{Nr}X_{2r} + W^k_{Ni}X_{2i}$$
$$X'_{1i} = X_{1i} + W^k_{Nr}X_{2i} - W^k_{Ni}X_{2r}$$
$$X'_{2r} = X_{1r} - W^k_{Nr}X_{2r} - W^k_{Ni}X_{2i}$$
$$X'_{2i} = X_{1i} - W^k_{Nr}X_{2i} + W^k_{Ni}X_{2r}$$

One butterfly operation executes the above formulas. In the formulas, X_{1r}、$X_{1i}$are the real and imaginary parts of the first input original data;X_{2r}、X_{2i} are the real and imaginary parts of the second input original data; W^k_{Nr} and $W^k_{Ni}$are the real and imaginary parts of the input parameter.

Software Operation Flow:

Configure MAC_CTL0[4:0] = 5'd9 to select the single butterfly mode;

Configure MAC_CTL0[9:7] to select the floating-point rounding mode;

Configure the MAC_IN0/1/2/3/4/5 registers to input data:

MAC_IN0=X_{1r}

MAC_IN1=X_{1i}

MAC_IN2=X_{2r}

MAC_IN3=X_{2i}

MAC_IN4=W^k_{Nr}

MAC_IN5=W^k_{Ni}

Configure MAC_CTL1[0] = 1 to start the butterfly operation;

Wait for the DSP_FLG[5] flag to be set, then read the results from MAC_OUT0/1/2/3:

MAC_OUT0=X'_{1r}

MAC_OUT1=X'_{1i}

MAC_OUT2=X'_{2r}

MAC_OUT3=X'_{2i}

If the interrupt enable signal is configured, an interrupt is generated after the single butterfly operation. Each configuration of the register to start the butterfly operation automatically clears the flag bit.

36.3.7.2 Continuous Conversion of Multiple Data

This mode calculates the butterfly operation for multiple groups of data in SRAM. Two numbers at adjacent addresses perform one butterfly operation, and the result is written back to SRAM.

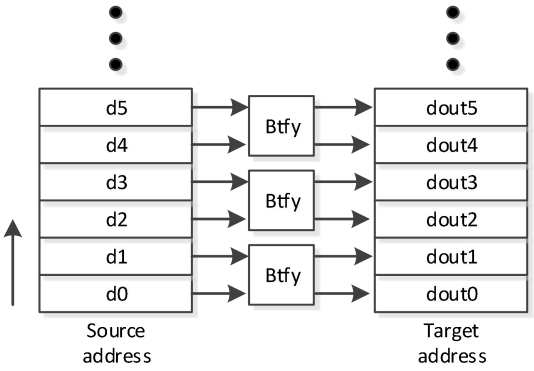


Figure 36-9 Continuous Conversion Mode for Multiple Data

As shown in the figure above, starting from the low address to the high address, two data are taken from the source data area each time to perform one butterfly operation, then written to the target SRAM.

Supports the same source DMA address and target DMA address to overwrite the source data, saving SRAM.

Software Configuration Flow:

- Configure MAC_CTL0[4:0] = 5'd10 to select the butterfly DMA mode;
- Configure MAC_CTL0[9:7] to select the floating-point rounding mode;
- Configure the DMA_SRBADR register to select the DMA start address of the real part of the source data;
- Configure the DMA_SIBADR register to select the DMA start address of the imaginary part of the source data;
- Configure the DMA_PRBADR register to select the DMA start address of the real part of the parameter table;
- Configure the DMA_PIBADR register to select the DMA start address of the imaginary part of the parameter table;
- Configure the DMA_TRBADR register to select the DMA start address of the real part of the target data;
- Configure the DMA_TIBADR register to select the DMA start address of the imaginary part of the target data;
- Configure the DMA_LEN register to select the number of data points (a power of 2);
- Clear the DSP_FLG[6] flag, configure MAC_CTL2[0] = 1 to start the calculation;
- Wait for the DSP_FLG[6] flag, then obtain the calculated data from SRAM.

After all data calculations are completed, a flag bit is generated. If the interrupt enable is configured, an interrupt is generated. The flag bit needs to be cleared by the software.

36.3.7.3 FFT (fp_fft)

In this mode, the hardware automatically performs radix-2 FFT calculation on the specified data (floating-point numbers) in SRAM, stores the result in SRAM, and generates a completion flag. The data length can be configured as 64, 128, 256, 512, or 1024.

Since FFT operation requires a parameter table, to speed up the FFT process, the hardware has a built-in parameter table for FFT of 64~1024 points. Thus, the table values are directly retrieved for each calculation, reducing the time for accessing SRAM.

Supports the same source data and target data addresses to achieve data overwriting, reducing SRAM overhead.

Software Configuration Flow:

- Configure MAC_CTL0[4:0] = 5'd11 to select the FFT mode;
- Configure MAC_CTL0[9:7] to select the floating-point rounding mode;
- If using the internal parameter table, configure MAC_CTL0[20] = 1; otherwise, if the table is stored in SRAM, configure the DMA_PRBADR register to select the DMA start address of the real part of the table and the DMA_PIBADR register to select the DMA start address of the imaginary part of the table;
- Configure the DMA_SRBADR register to select the DMA start address of the real part of the source data; configure the DMA_SIBADR register to select the DMA start address of the imaginary part of the source data;
- Configure the DMA_TRBADR register to select the DMA start address of the real part of the target data; configure the DMA_TIBADR register to select the DMA start address of the imaginary part of the target data;
- Configure the DMA_LEN register to select the data length;
- Clear the DSP_FLG[7] flag, configure MAC_CTL2[0] = 1 to start the calculation;
- Wait for the completion flag DSP_FLG[7] = 1, indicating that the FFT is completed.

36.3.7.4 Bit Reverse Data Transfer (bitrev_move)

In this mode, the hardware automatically reorders a segment of original data in SRAM according to the BITREV algorithm and writes it to the specified position in SRAM.

The data length supports 4, 8, 16, 32, 64, 128, 256, 512, and 1024. Each data occupies 4 bytes of SRAM space and is stored in Word-aligned format in SRAM.

Software Configuration Flow:

- Configure MAC_CTL0[4:0] = 5'd12 to select the BITREV mode;
- Configure the DMA_SRBADR register to select the DMA start address of the source data;
- Configure the DMA_TRBADR register to select the DMA start address of the target;
- Configure the DMA_LEN register to select the data length;
- Clear the DSP_FLG[8] flag, configure MAC_CTL2[0] = 1 to start the calculation;
- Wait for the completion flag DSP_FLG[8] = 1, indicating completion.

36.3.8 Sine and Cosine Calculation (sin_cos)

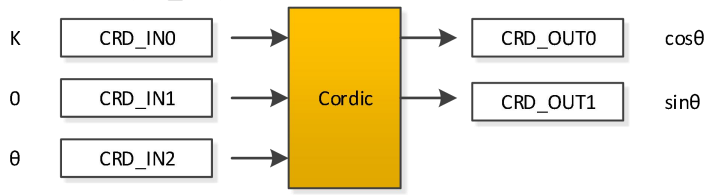


Figure 36-10 Sine and Cosine Calculation

This mode supports inputting an angle, and the hardware calculates the sin and cos functions.

Adopting the CORDIC rotation mode, configure $X_0 = \frac{1}{P} = K = 0.60725$, Y_0=0, $Z_0 = \theta$, and the output result is $[\cos\theta, \sin\theta, 0]$.

Software Configuration Flow:

- Configure the CRD_ANGLE register to input the angle (register value = radian/pi*(2^{31}));
- Clear the CRD_FLG[1] flag, configure CRD_CTL[1] = 1 to start the operation;
- Wait for the completion flag CRD_FLG[1], then read the CRD_COSINE/CRD_SINE registers to obtain the calculation result.

36.3.9 Root Mean Square and Arctangent (fp_sqrt/ fp_atan)

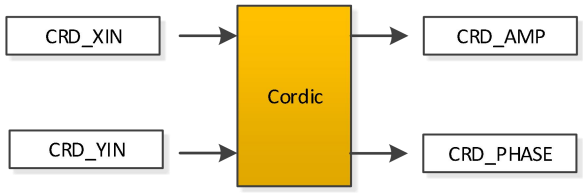


Figure 36-11 Root Mean Square

This mode calculates the root mean square of the input data CRD_XIN and CRD_YIN.

Adopting the CORDIC vector mode, configure X_0=X，Y_0=Y，$Z_0 = 0$, and the output is.

Software Configuration Flow:

- Configure CRD_XIN = X;
- Configure CRD_YIN = Y;$\left[P\sqrt{X_i^2 + Y_i^2},\ 0,\ \arctan(\frac{Y_i}{X_i})\right]$
- Clear the CRD_FLG[0] flag, configure CRD_CTL[0] = 1 to start the operation;
- Wait for the completion flag CRD_FLG[0], then read the CRD_AMP/CRD_PHASE registers to obtain the calculation result;CRD_AMP = $\sqrt{X_0^2 + Y_0^2}$/4*P=$\sqrt{X_0^2 + Y_0^2}$/（$4 * 0.60725$），CRD_PHASE = $\arctan\left(\frac{y_0}{x_0}\right) * pi/2\text{^}31$

If the interrupt enable signal is configured, an interrupt is generated after the calculation is completed.

36.3.10 Floating-Point Divider (fp_div)

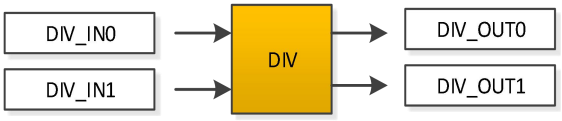


Figure 36-12 Division

This mode performs a floating-point division operation. Input two floating-point numbers, perform division, and output the quotient and remainder.

Software Configuration Flow:

- Configure the dividend into the DIV_IN0 register;
- Configure the divisor into the DIV_IN1 register;
- Clear the DSP_FLG[2] flag, configure MAC_CTL2[2] = 1 to start the division operation;
- Wait for the completion flag DSP_FLG[2], then read the DIV_OUT0 register to obtain the quotient.

36.3.11 IIR Filter

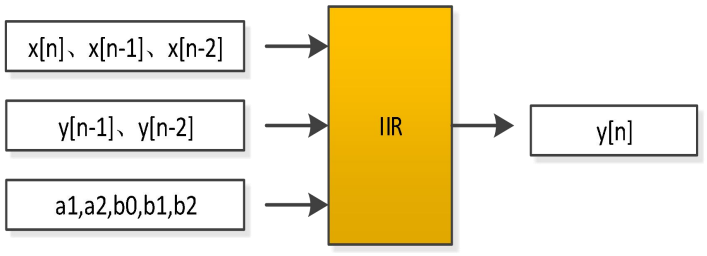


Figure 36-13 IIR Calculation

This mode performs IIR filter calculation. After inputting floating-point data a_1、a_2、b_0、b_1、b_2、y[n-2]，y[n-1]，x[n]，x[n-1]，x[n-2], it performs IIR operation and outputs y[n].

36.3.11.1 Single Data Conversion

- Configure MAC_CTL0[4:0] = 13 (IIR_ONCE_MODE) to select the single IIR mode;
- Configure MAC_CTL0[9:7] to select the floating-point rounding mode;
- Configure the MAC_IN01/2/3/4/5, DIV_IN0/1, MAC_OUT0/1 registers to input data:

(MAC_IN0 must be configured last; after writing to MAC_IN0, the hardware automatically starts the IIR operation)

MAC_IN1=x[n-1]
MAC_IN2=x[n-2]
MAC_IN3=b_0
MAC_IN4=b_1
MAC_IN5=b_2
DIV_IN0=a_1
DIV_IN1=a_2
MAC_OUT0=y[n]
MAC_OUT1=y[n-1]
MAC_IN0= x[n]

- Wait for the DSP_FLG[10] (IIR_ONCE_DONE) flag to be set, then read the result from MAC_OUT0;

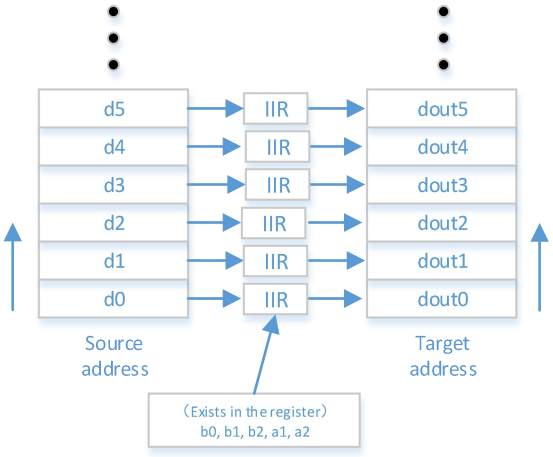
MAC_OUT0 = y[n]

If the interrupt enable signal (DSP_IE[10].IIR_ONCE_IE = 1) is configured, an interrupt is generated after the IIR operation. The flag bit needs to be cleared by the software.

(Because IIR uses DIV_IN0/1 to configure parameters a1 and a2, DIV_IN0/1 cannot be changed during the IIR operation. That is, IIR and division cannot be used simultaneously.)

36.3.11.2 Continuous Conversion of Multiple Data

This mode calculates the IIR operation for multiple groups of data in SRAM. Three numbers at adjacent addresses perform one IIR operation, and the result is written back to SRAM.



As shown in the figure above, starting from the low address to the high address, three data are taken from the source data area each time to perform IIR calculation, then written to the target SRAM.

Software Configuration Flow:

- Configure MAC_CTL0[4:0] = 14 to select the IIR DMA mode;
- Configure MAC_CTL0[9:7] to select the floating-point rounding mode;
- Configure the MAC_IN1/2/3/4/5, DIV_IN0/1, MAC_OUT0/1 registers to input data:

  MAC_IN1=x[n-1]

  MAC_IN2=x[n-2]

  MAC_IN3= b_0

  MAC_IN4= b_1

  MAC_IN5= b_2

  DIV_IN0= a_1

  DIV_IN1= a_2

  MAC_OUT0=y[n]

  MAC_OUT1=y[n-1]
- Configure the start address of data x[n] (DMA_SRBADR), target address of data y[n] (DMA_TRBADR), data length (DMA_LEN), and SRAM read step of DMA (MAC_CTL1[3:1]);
- Clear the DSP_FLG[11] flag, configure MAC_CTL2[0] = 1 (DMA_EN) to start the operation (it is recommended to clear to 0 first, then write 1);
- Wait for the DSP_FLG[11] (IIR_DMA_DONE) flag to be set, then read y[n] from the target address in SRAM.

If the interrupt enable signal (DSP_IE[11].IIR_DMA_IE = 1) is configured, an interrupt is generated after the IIR DMA operation. The flag bit needs to be cleared by the software.

36.3.12 FIR Filter

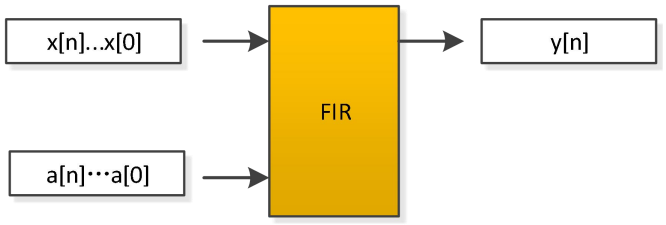
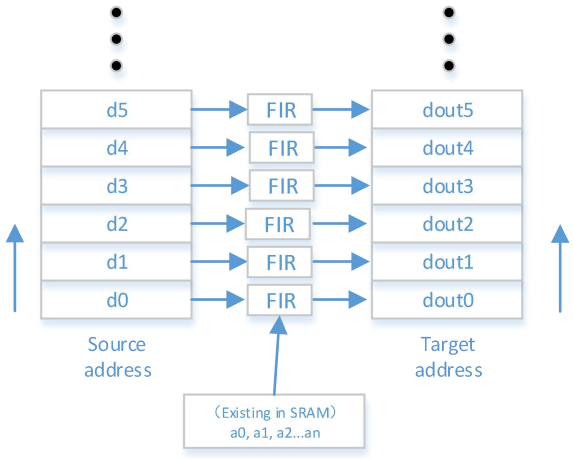


Figure 36-14 FIR Operation

This mode performs FIR filter calculation. After inputting floating-point data x[n]…x[0] and parameters a[n]…a[0], it performs FIR operation and outputs y[n].

This mode calculates the FIR operation for multiple data in SRAM. N numbers at adjacent addresses perform one FIR operation, and the result is written back to SRAM.



Starting from the low address to the high address, three data are taken from the source data area each time to perform FIR calculation, then written to the target SRAM.

Software Configuration Flow:

- Configure MAC_CTL0[4:0] = 15 to select the FIR DMA mode;
- Configure MAC_CTL0[9:7] to select the floating-point rounding mode;
- Configure the order of FIR, start address of DMA data x[n] (DMA_SRBADR), start address of parameter a (DMA_PRBADR), target address of data y[n] (DMA_TRBADR), data length (DMA_LEN), and SRAM read step of DMA (MAC_CTL1[3:1]);
- Clear the DSP_FLG[11] flag, configure MAC_CTL2[0] = 1 (DMA_EN) to start the operation (it is recommended to clear to 0 first, then write 1);
- Wait for the DSP_FLG[11] (FIR_DMA_DONE) flag to be set, then read y[n] from the target address in SRAM.

If the interrupt enable signal (DSP_IE[11].FIR_DMA_IE = 1) is configured, an interrupt is generated after the FIR DMA operation. The flag bit needs to be cleared by the software.

(It is not allowed to configure dma_len less than fir_order; dma_len >= fir_order.)

36.3.13 Linear Interpolation

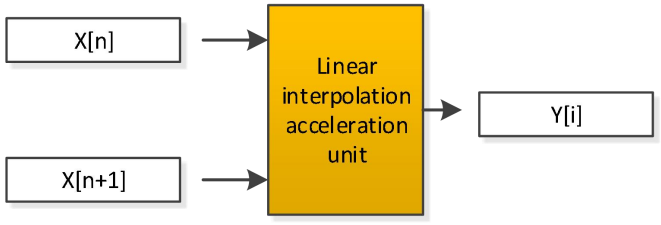


Figure 36-15 Linear Interpolation Acceleration Algorithm

This mode implements the linear interpolation acceleration algorithm. One y[i] value is output from each two adjacent points according to the formula.

This mode calculates the linear interpolation operation for multiple data in SRAM. Two numbers at adjacent addresses perform one linear interpolation operation, and the result is written back to SRAM.

Starting from the low address to the high address, two data are taken from the source data area each time to perform linear interpolation calculation, then written to the target SRAM.

Software Configuration Flow:

- The software calculates various parameters, including the length of actually available input data (len), the length of interpolated data (out_len), and the current interpolation position (t);
- Configure MAC_CTL0[4:0] = 16 (LINE_INTP) to select the linear interpolation mode;
- Configure MAC_CTL0[9:7] to select the floating-point rounding mode;
- Configure the start address of data x[n] (DMA_SRBADR) and target address of data y[i] (DMA_TRBADR);
- Configure the MAC_IN5[31:0] register (the last data of the previous cycle);
- Configure MAC_CTL1[0] to select whether to use the last data of the previous cycle;
- Configure the INTP_LEN[12:0] register to select the length of actually available input data;
- Configure the DMA_LEN[10:0] register to select the length of interpolated data;
- Configure the INTP_LOC[31:0] register to select the current interpolation position;
- Configure the INTP_STEP[31:0] register to select the step;
- Clear the DSP_FLG[13] flag, configure MAC_CTL2[0] = 1 (DMA_EN) to start the operation (it is recommended to clear to 0 first, then write 1);
- Wait for the DSP_FLG[13] (LINE_INTP_DONE) flag to be set, then read y[n] from the target address in SRAM;
- The software reads INTP_LOC[31:0], sets t = INTP_LOC[31:0], and calculates the redundant length after the completion of this interpolation calculation: red_len = len - t + step;
- According to specific conditions, the software selects whether to save the last data. If saving is required, write the value to the MAC_IN5[31:0] register (the last data of the previous cycle).

If the interrupt enable signal (DSP_IE[13].LINE_INTP_IE = 1) is configured, an interrupt is generated after the linear interpolation operation. The flag bit needs to be cleared by the software.

36.3.14 Lagrange Interpolation

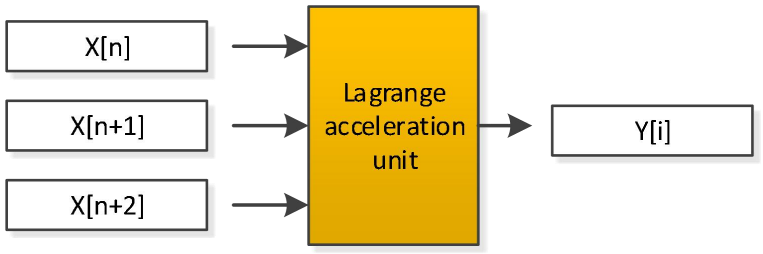


Figure 36-16 Lagrange Interpolation Acceleration Algorithm

This mode implements the Lagrange interpolation acceleration algorithm. One y[i] value is output from each three adjacent points according to the formula.

This mode calculates the Lagrange interpolation operation for multiple data in SRAM. Three numbers at adjacent addresses perform one Lagrange interpolation operation, and the result is written back to SRAM.

Starting from the low address to the high address, three data are taken from the source data area each time to perform Lagrange calculation, then written to the target SRAM.

Software Configuration Flow:

- The software calculates various parameters, including the length of actually available input data (len), the length of interpolated data (out_len), and the current interpolation position (t);
- Configure MAC_CTL0[4:0] = 17 (LAGR_INTP) to select the Lagrange interpolation mode;
- Configure MAC_CTL0[9:7] to select the floating-point rounding mode;
- Configure the start address of DMA data x[n] (DMA_SRBADR) and target address of data y[i] (DMA_TRBADR);
- Configure the MAC_IN5[31:0] register (the last data of the previous cycle);
- Configure MAC_CTL1[0] to select whether to use the last data of the previous cycle;
- Configure the INTP_LEN[12:0] register to select the length of actually available input data;
- Configure the DMA_LEN[9:0] register to select the length of interpolated data;
- Configure the INTP_LOC[31:0] register to select the current interpolation position;
- Configure the INTP_STEP[31:0] register to select the step;
- Clear the DSP_FLG[14] flag, configure MAC_CTL2[0] = 1 (DMA_EN) to start the operation (it is recommended to clear to 0 first, then write 1);
- Wait for the DSP_FLG[14] (LAGRINTP_DONE) flag to be set, then read y[n] from the target address in SRAM;
- The software reads INTP_LOC[31:0], sets t = INTP_LOC[31:0], and calculates the redundant length after the completion of this interpolation calculation: red_len = len - t + step;
- According to specific conditions, the software selects whether to save the last data. If saving is required, write the value to the MAC_IN5[31:0] register (the last data of the previous cycle).

If the interrupt enable signal (DSP_IE[14].LAGR_INTP_IE = 1) is configured, an interrupt is generated after the Lagrange interpolation operation. The flag bit needs to be cleared by the software.

36.4 Implementation Description

36.4.1 Integer to Floating-Point Conversion

Since the input integer bit width is 32 bits, the exponent range of the converted floating-point number is small, so there will be no overflow. When performing normalization, the maximum support is 32th power, and no overflow

handling is required.

Normalization means subtracting n from the exponent part while keeping the fractional part unchanged.

36.4.2 Floating-Point to Integer Conversion

It is necessary to scale the floating-point number first to make the integer part large enough, then perform the conversion to ensure more valid bits.

Scaling by 2^n means adding n to the exponent part while keeping the fractional part unchanged.

36.4.3 FFT Description

The software only needs to store the original data in SRAM in advance, and configure the DMA start address, target address, and data length. The hardware automatically fetches data and performs operations to obtain the butterfly result. If the data length is 64, there are 64/2 = 32 butterfly operations per stage, and a total of $\log_2 64 = 6$ stages. Therefore, the hardware automatically completes 32*6 butterfly operations, stores the result in the specified position in SRAM, and generates a completion flag or interrupt.

For each butterfly operation, the hardware needs to read SRAM 4 times to obtain the original data (the imaginary part of the original input data is 0 in the first round), and write SRAM 4 times to save the result of this butterfly operation. (If the parameter table is stored in SRAM, it also needs to read SRAM 2 times to obtain parameters.) Therefore, the number of SRAM access cycles required to complete a specified length of data is:

$k_0 = \frac{n}{2} * log_2 n * (4 + 4) - \frac{n}{2} * 2$, with hardware-embedded parameter table;

$k_0 = \frac{n}{2} * log_2 n * (4 + 4 + 2) - \frac{n}{2} * 2$, with parameter table stored in SRAM.

Number of SRAM Read/Write Cycles for FFT Completion

Number of Points	SRAM Read/Write Cycles	
	Hardware Table	SRAM Table
64	1472	1856
128	3456	4352
256	7936	9984
512	17920	22528
1024	39936	50176

Including SRAM read/write and butterfly operations, the fastest time to complete one butterfly is 10 clock cycles (hardware-embedded parameter table) or 12 clock cycles (parameter table stored in SRAM).

Clock Cycle	1	2	3	4	5	6	7	8	9	10	11	12
Hardware Behavior	Read	Read	Read	Read	Read	Read						
		Latch	Latch	Latch	Latch	Latch	Latch					
								Multiply	Multiply	Multiply	Multiply	
								Add	Add	Add	Add	
								Add	Add	Add	Add	
									Write	Write	Write	Write

Thus, the total time is:

$k_0 = \frac{n}{2} * log_2 n * 10$, with hardware-embedded parameter table;

$k_0 = \frac{n}{2} * log_2 n * 12$, with parameter table stored in SRAM.

Table: FFT Clock Cycles and Time

Number of Points	FFT Clock Cycles		FFT Time (us) (16M Clock)	
	Hardware Table	SRAM Table	Hardware Table	SRAM Table
64	1920	2304	120	144
128	4480	5376	280	336
256	10240	12288	640	768
512	23040	27648	1440	1728
1024	51200	61440	3200	3840

Thus, the fastest time to complete FFT is the calculated time in the table plus the time to configure DMA and other mode registers. The time in the table is calculated based on idle SRAM with only FFT module access. In actual situations, other modes may seize SRAM, so the FFT time will be longer.

36.4.4 CORDIC Description

Arctangent parameter table: from arctan(1) (45 degrees), arctan(1/2)… to arctan($1/2^n$).

All parameters need to be scaled proportionally. Taking arctan(1)=2^n as the standard, other parameters are scaled by the same factor a. The scaling factor a is a quantization process that introduces errors.

In each iteration, x and y are shifted (approximated downward), which introduces errors. The error can be reduced by using rounding instead.

The number of iterations is determined by the scaled parameter table. If the parameter table value is 0, the iteration ends.

36.4.4.1 Vector Mode

Determine the bit width:

- First, since the bit width of the input data x and y is 32 bits;
- In this mode, the input z = 0, so the scaling factor of arctangent has nothing to do with the input and can be arbitrarily selected. At this time, the scaling factor n of arctangent is only determined by the required output angle precision.
- The number of iterations is determined by the scaling factor of arctangent.
- The final output angle precision is related to the scaling factor n of arctangent. The output angle result must be divided by n to get the actual angle (because this angle is accumulated, $z(i+1) = z(i) - \arctan(1/2^i)$; both sides are scaled by n simultaneously). It is also possible to output the upper bits (the operation process is relatively accurate, and the output result is relatively short).

In vector mode, the input x and y can be scaled proportionally to achieve higher calculation accuracy. Therefore, for relatively small x and y, higher precision results can be obtained by scaling first and then substituting into the

CORDIC operation.

Preprocessing is first required to move all angles to the first quadrant. If it is in the second quadrant, move to the first quadrant and set the initial angle z = 90 degrees. For the third quadrant, set the initial angle z = 180 degrees; for the fourth quadrant, set the initial angle z = 270 degrees.

```
if(x<0){
    x=-x;
    y=-y;
    z=z-(1~3)pi/2*n;    // n is the scaling factor
}
```

36.4.4.2 Rotation Mode

Determine the bit width:

- The module input is in radians, ranging from−π：π. Assuming the input radians are represented by n bits, the input radian scaling factor is p0=(2^n)/2pi.
- According to the formula z(i+1)=z(i)-arctan(1/2^i), the scaling factor of arctangent must be the same as that of the input angle z during addition and subtraction operations. atan_t(n)=p0*atan(n), and other parameter values are scaled by the same factor p0.
- Under normal circumstances, atan_t(1)=p0*atan(n)=2^(n-3), so the parameter table bit width is n-3. To reduce quantization errors, the number of bits of the parameter table can be set longer. During calculation, it is only necessary to right-shift the input data and scale by the same factor to participate in addition and subtraction.

  For example, if the bit width of atan during quantization is n-3+5, then scale before operation:

  ```
  x_in = {x_in,5'h0};
  y_in ={y_in,5'h0};
  z_in ={z_in,5'h0};
  ```

  After iteration, take the upper bits of the output.
- The output result after iteration is also scaled by p0. The output scaling factor can be adjusted according to the bit width of the output data.

Implementation process of rotation mode:

- Input $X_0 = k$, $Y_0=0$, $Z_0 = \theta$; the input angle range is [-pi, pi];
- First, preprocess the angle to flip it to the [0, pi/2] range;
- Iterate according to the formula to get the result;
- Multiply the result by the gain coefficient K;
- The bit width of the input x and y is related to the number of iterations, because it is shifted by 1 bit each time.

36.5 Registers

36.5.1 Register List

Base Addresses of Registers in the DSP Module

Module Name	Core0 Base Address	Core1 Base Address
DSP	0x2006_C000	0xA006_C000

Register List of the DSP Module

Offset Address	Name	RW	Valid Word	Reset Value	Function Description

			Length		
0x0	MAC_CTL0	RW	4	0x0	MAC Unit Control Register 0
0x4	MAC_CTL1	RW	4	0x0	MAC Unit Control Register 1
0x8	MAC_CTL2	RW	4	0x0	MAC Unit Control Register 2
0xC	MAC_IN0	RW	4	0x0	MAC Unit Data Input Register 0
0x10	MAC_IN1	RW	4	0x0	MAC Unit Data Input Register 1
0x14	MAC_IN2	RW	4	0x0	MAC Unit Data Input Register 2
0x18	MAC_IN3	RW	4	0x0	MAC Unit Data Input Register 3
0x1C	MAC_IN4	RW	4	0x0	MAC Unit Data Input Register 4
0x20	MAC_IN5	RW	4	0x0	MAC Unit Data Input Register 5
0x24	MAC_OUT0	RW	4	0x0	MAC Unit Data Output Register 0
0x28	MAC_OUT1	RW	4	0x0	MAC Unit Data Output Register 1
0x2C	MAC_OUT2	R	4	0x0	MAC Unit Data Output Register 2
0x30	MAC_OUT3	R	4	0x0	MAC Unit Data Output Register 3
0x34	DIV_IN0	RW	4	0x0	Division Unit Data Input 0
0x38	DIV_IN1	RW	4	0x3F800000 (1.0)	Division Unit Data Input 1
0x3C	DIV_OUT0	R	4	0x0	Division Unit Data Output
0x40	DMA_SRBADR	RW	4	0x0	DMA Source Data FFT Real Part Start Address
0x44	DMA_SIBADR	RW	4	0x0	DMA Source Data FFT Imaginary Part Start Address
0x48	DMA_PRBADR	RW	4	0x0	DMA Twiddle Factor Table Real Part Start Address
0x4C	DMA_PIBADR	RW	4	0x0	DMA Twiddle Factor Table Imaginary Part Start Address
0x50	DMA_TRBADR	RW	4	0x0	DMA Target Data FFT Real Part Start Address
0x54	DMA_TIBADR	RW	4	0x0	DMA Target Data FFT Imaginary Part Start Address
0x58	DMA_LEN	RW	2	0x0	DMA Length Configuration
0x5C	DSP_IE	RW	2	0x0	FFT Interrupt Enable Register
0x60	DSP_FLG	RW	2	0x0	FFT Flag Register
0x64	ALU_STA0	R	4	0x0	ALU Status Register 0
0x68	ALU_STA1	R	2	0x0	ALU Status Register 1
0x6C	CRD_CTL	WO	1	0x0	CORDIC Control Register
0x70	CRD_XIN	RW	4	0x0	CORDIC Vector Mode Data X Input
0x74	CRD_YIN	RW	4	0x0	CORDIC Vector Mode Data Y Input
0x78	CRD_AMP	R	4	0x0	CORDIC Vector Mode Amplitude Output
0x7C	CRD_PHASE	R	4	0x0	CORDIC Vector Mode Phase Output
0x80	CRD_ANGLE	RW	4	0x0	CORDIC Rotation Mode Angle

					Input
0x84	CRD_COS	R	4	0x0	CORDIC Rotation Mode Cosine Output
0x88	CRD_SIN	R	4	0x0	CORDIC Rotation Mode Sine Output
0x8C	CRD_IE	RW	1	0x0	CORDIC Interrupt Enable
0x90	CRD_FLG	RW	1	0x0	CORDIC Completion Flag Register
0x94	INTP_LEN	RW	2	0x0	Interpolation Input Data Length Register
0x98	INTP_LOC	RW	4	0x0	Current Interpolation Position Register
0x9C	INTP_STEP	RW	4	0x0	Interpolation Algorithm Step Register
0xA0	PFMODE	RW	1	0x1	DSP Module High-Performance Mode Selection Register

36.5.2 MAC_CTL0（0x0）

Offset Address: 00H; Default Value: 0x0

Bits	Bit Name	Function Description	R/W Flag	Reset Value
31:23	Reserved	Reserved	R	0
22:21	I2F_PRE_EN	Integer to Floating-point Preprocessing 2'b00: All 4 Bytes are valid; 2'b01: In each 4 Bytes, the highest Byte data is invalid. Valid data is only 3 Bytes; 2'b10: In each 4 Bytes, the highest 2 Bytes data is invalid. Valid data is only 2 Bytes; 2'b11: In each 4 Bytes, the highest 3 Bytes data is invalid. Valid data is only 1 Byte	RW	0
20	FFT_TB_EN	FFT uses internal parameter table, valid for 64/128/256/512/1024 points. 0 = Disable 1 = Enable	RW	0
19:15	F2I_MUL	Amplify floating-point number before converting to integer 0 means no amplification n means amplify by 2^n	RW	0
14:10	I2F_DIV	Normalization order configuration after integer to floating-point conversion 0 means no normalization n means divide by 2^n	RW	0
9:7	ROUND_MODE	Floating-point operation rounding mode configuration: rnd=000: nearest rounding;	RW	0

		rnd=001: zero rounding;
		rnd=010: +∞ rounding;
		rnd=011: -∞ rounding;
		rnd=100: up rounding;
		rnd=101: away rounding;
		Others: Reserved.		

6	MUL_OUT_FM	Floating-point multiplication output data format 0 = Floating-point 1 = Integer	RW	0
5	MUL_IN_FM	Floating-point multiplication input data format 0 = Floating-point 1 = Integer	RW	0
4:0	MODE_SEL	5'd0: All modes disabled; 5'd1: Integer to floating-point, single enable; 5'd2: Integer to floating-point, DMA enable; 5'd3: Floating-point to integer, single enable; 5'd4: Floating-point to integer, DMA enable; 5'd5: Floating-point multiplication, mode enable; 5'd6: Floating-point addition, mode enable; 5'd7: Floating-point subtraction, mode enable; 5'd8: Floating-point multiply-accumulate, mode enable (y=ax+b); 5'd9: Butterfly operation, single enable; 5'd10: Butterfly operation, DMA enable; 5'd11: FFT mode enable; 5'd12: BITREV mode enable; 5'd13: Second-order IIR single operation, mode enable; 5'd14: Second-order IIR DMA operation, mode enable; 5'd15: N-order FIR DMA operation, mode enable; 5'd16: Linear interpolation operation, mode enable; 5'd17: Lagrange interpolation operation, mode enable; 5'd18~31: All modes disabled;	RW	0

36.5.3 MAC_CTL1（0x04）

Offset Address: 04H; Default Value: 0x0

Bits	Bit Name	Function Description	R/W Flag	Reset Value
31:12	Reserved	Reserved.	R	0
11:4	FIR_ORDER	FIR Order Configuration; 0~3: 3rd order; 4: 4th order; 5: 5th order; ...	RW	0

		254: 254th order; 255: 255th order;		
3:1	DMA_STEP	DMA Read SRAM Step, only effective for IIR, FIR DMA modes. 0: Step 1; 1: Step 2; ... 7: Step 8;	RW	0
0	INTP_LAST_EN	Interpolation mode uses the last data from the previous period, enable signal	RW	0

36.5.4 MAC_CTL2（0x08）

Offset Address: 08H; Default Value: 0x0

Bits	Bit Name	Function Description	R/W Flag	Reset Value
31:3	Reserved	Reserved.	R	0
2	DIV_KICK	Divider Start Enable Signal	WO	0
1	BTFY_ONCE_KICK	Single Butterfly Operation Mode Start Signal, active high	WO	0
0	DMA_EN	DMA Enable Signal, all DMA-related operations are started by configuring this bit. Automatically cleared to 0 after DMA completion.	RW	0

Note: Bits 1 and 2 are write-only bits.

36.5.5 MAC_IN0（0x0C）

Offset Address: 0CH; Default Value: 0x0

Bits	Bit Name	Function Description	R/W Flag	Reset Value
31:0	MAC_IN0	MAC Unit Data Input Port	RW	0

36.5.6 MAC_IN1（0x10）

Offset Address: 10H; Default Value: 0x0x0

Bits	Bit Name	Function Description	R/W Flag	Reset Value
31:0	MAC_IN1	MAC Unit Data Input Port	RW	0

36.5.7 MAC_IN2（0x14）

Offset Address: 14H; Default Value: 0x0

Bits	Bit Name	Function Description	R/W Flag	Reset Value
31:0	MAC_IN2	MAC Unit Data Input Port	RW	0

36.5.8 MAC_IN3（0x18）

Offset Address: 18H; Default Value: 0x0

Bits	Bit Name	Function Description	R/W Flag	Reset Value
31:0	MAC_IN3	MAC Unit Data Input Port	RW	0

36.5.9 MAC_IN4（0x1C）

Offset Address: 1CH; Default Value: 0

Bits	Bit Name	Function Description	R/W Flag	Reset Value
31:0	MAC_IN4	MAC Unit Data Input Port	RW	0

36.5.10 MAC_IN5（0x20）

Offset Address: 20H; Default Value: 0

Bits	Bit Name	Function Description	R/W Flag	Reset Value
31:0	MAC_IN5	MAC Unit Data Input Port	RW	0

36.5.11 MAC_OUT0（0x24）

Offset Address: 24H; Default Value: 0x0

Bits	Bit Name	Function Description	R/W Flag	Reset Value
31:0	MAC_OUT0	MAC Unit Data Output Port	RW	0

36.5.12 MAC_OUT1（0x28）

Offset Address: 28H; Default Value: 0x0

Bits	Bit Name	Function Description	R/W Flag	Reset Value
31:0	MAC_OUT1	MAC Unit Data Output Port	RW	0

36.5.13 MAC_OUT2（0x2C）

Offset Address: 2CH; Default Value:0x0

Bits	Bit Name	Function Description	R/W Flag	Reset Value
31:0	MAC_OUT2	MAC Unit Data Output Port	R	0

36.5.14 MAC_OUT3（0x30）

Offset Address: 30H; Default Value: 0x0

Bits	Bit Name	Function Description	R/W Flag	Reset Value
31:0	MAC_OUT3	MAC Unit Data Output Port	R	0

36.5.15 DIV_IN0（0x34）

Offset Address: 34H; Default Value: 0x0

Bits	Bit Name	Function Description	R/W Flag	Reset Value
31:0	DIV_IN0	Division Unit Data Input Port (Dividend)	RW	0

36.5.16 DIV_IN1（0x38）

Offset Address: 38H; Default Value: 0x3F800000

Bits	Bit Name	Function Description	R/W Flag	Reset Value
31:0	DIV_IN1	Division Unit Data Input Port (Divisor)	RW	0x3F800000 (1.0)

36.5.17 DIV_OUT0（0x3C）

Offset Address: 3CH; Default Value: 0x0

its	Bit Name	Function Description	R/W Flag	Reset Value
31:0	DIV_OUT0	Division Unit Data Output Port (Quotient)	R	0

36.5.18 DMA_SRBADR（0x40）

Offset Address: 40H; Default Value: 0x0

Bits	Bit Name	Function Description	R/W Flag	Reset Value
31:16	Reserved	Reserved.	R	0
15:0	DMA_SRBADR	DMA Source Data Start Address: (Word Address) FFT/BTFY_DMA Mode: DMA Source Data FFT Real Part Start Address I2F_DMA/F2I_DMA/BIT_REV Mode: DMA Source Data Start Address Linear interpolation/Lagrange interpolation mode: Starting address of data before DMA interpolation; IIR/FIR filter mode: Starting address of data before DMA filtering	RW	0

36.5.19 DMA_SIBADR（0x44）

Offset Address: 44H; Default Value:0x0

Bits	Bit Name	Function Description	R/W Flag	Reset Value
31:16	Reserved	Reserved.	R	0
15:0	DMA_SIBADR	DMA Source Data Start Address: (Word Address) FFT/BTFY_DMA Mode: DMA Source Data FFT Imaginary Part Start Address	RW	0

36.5.20 DMA_PRBADR（0x48）

Offset Address: 48H; Default Value:0x0

Bits	Bit Name	Function Description	R/W Flag	Reset Value
31:16	Reserved	Reserved.	R	0
15:0	DMA_PRBADR	DMA Source Data Start Address: (Word Address) FFT or BTFY Mode: DMA Twiddle Factor Parameter Table Real Part Start Address	RW	0

36.5.21 DMA_PIBADR（0x4C）

Offset Address: 4CH; Default Value: 0x0

Bits	Bit Name	Function Description	R/W Flag	Reset Value
31:16	Reserved	Reserved.	R	0
15:0	DMA_PIBADR	DMA Source Data Start Address: (Word Address) FFT or BTFY Mode: DMA Twiddle Factor Parameter Table Imaginary Part Start Address	RW	0

36.5.22 DMA_TRBADR（0x50）

Offset Address: 50H; Default Value:0x0

Bits	Bit Name	Function Description	R/W Flag	Reset Value
31:16	Reserved	Reserved.	R	0
15:0	DMA_TRBADR	DMA Target Data Start Address: (Word Address) FFT/BTFY_DMA Mode: DMA Target Data FFT Real Part Start Address I2F_DMA/F2I_DMA/BIT_REV Mode: DMA Source Data Target Address Linear interpolation/Lagrange interpolation mode: Starting address of data after DMA interpolation; IIR/FIR filter mode: Starting address of data after DMA filtering	RW	0

36.5.23 DMA_TIBADR（0x54）

Offset Address: 54H; Default Value: 0x0

Bits	Bit Name	Function Description	R/W Flag	Reset Value
31:16	Reserved	Reserved.	R	0
15:0	DMA_TIBADR	DMA Target Data Start Address: (Word Address) FFT/BTFY_DMA Mode: DMA Target Data FFT Imaginary Part Start Address	RW	0

36.5.24 DMA_LEN（0x58）

Offset Address: 58H; Default Value: 0x0

Bits	Bit Name	Function Description	R/W Flag	Reset Value
31:10	Reserved	Reserved.	R	0
9:0	DMA_LEN	Point Configuration n, Length = (n+1) Word I2F_DMA/F2I_DMA/BTFY_DMA Mode: Supports any configuration BITREV Mode: Only supports points 4, 8, 16, 32, 64, 128, 256, 512, 1024 FFT Mode: Only supports 64, 128, 256, 512, 1024 points Linear interpolation/Lagrange interpolation mode: length of data after interpolation; IIR_DMA mode: length of data before filtering, supports any number of configurable points; FIR_DMA mode: actually usable data length before filtering, set the length of new data updated each time as L, configure the filter order fir_order as N, then the register value DMA_LEN=L N (N historical input data need to be stored at the lower address of the input data)	RW	0

36.5.25 DSP_IE（0x5C）

Offset Address: 5CH; Default Value: 0x0

Bits	Bit Name	Function Description	R/W Flag	Reset Value
31:15	Reserved	Reserved.	R	0
14	LAGR_DMA_IE	=0, Disable =1, Lagrange Interpolation Completion Interrupt Enable	RW	0
13	LINE_DMA_IE	=0, Disable =1, Linear Interpolation Completion Interrupt Enable	RW	0
12	FIR_DMA_IE	=0, Disable =1, FIR DMA Operation Completion Interrupt Enable	RW	0
11	IIR_DMA_IE	=0, Disable =1, IIR DMA Operation Completion Interrupt Enable	RW	0
10	IIR_ONCE_IE	=0, Disable =1, Single IIR Operation Completion Interrupt Enable	RW	0
9	MULT_IE	=0, Disable =1, Floating-point Multiplication Interrupt Enable	RW	0
8	BITREV_IE	=0, Disable =1, BITREV Mode Interrupt Enable	RW	0
7	DSP_IE	=0, Disable =1, FFT Mode Interrupt Enable	RW	0
6	BTFY_DMA_IE	=0, Disable	RW	0

		=1, Butterfly Operation DMA Interrupt Enable		
5	BTFY_ONCE_IE	=0, Disable =1, Single Butterfly Operation Interrupt Enable	RW	0
4	F2I_DMA_IE	=0, Disable =1, Floating-point to Integer DMA Interrupt Enable	RW	0
3	I2F_DMA_IE	=0, Disable =1, Integer to Floating-point DMA Interrupt Enable	RW	0
2	DIV_IE	=0, Disable =1, Division Unit Interrupt Enable	RW	0
1	DMA_IE	=0, Disable =1, DMA Interrupt Enable	RW	0
0	MAC_IE	=0, Disable =1, MAC Unit Interrupt Enable	RW	0

36.5.26 DSP_FLG（0x60）

Offset Address: 60H; Default Value:0x0

Bits	Bit Name	Function Description	R/W Flag	Reset Value
31:15	Reserved	Reserved.	R	0
14	LAGR_DMA_DONE	Lagrange Interpolation Completion Interrupt Flag Write 1 to clear 0	RWC	0
13	LINE_DMA_DONE	Linear Interpolation Completion Interrupt Flag Write 1 to clear 0	RWC	0
12	FIR_DMA_DONE	FIR DMA Operation Completion Interrupt Flag Write 1 to clear 0	RWC	0
11	IIR_DMA_DONE	IIR DMA Operation Completion Interrupt Flag Write 1 to clear 0	RWC	0
10	IIR_ONCE_DONE	Single IIR Operation Completion Interrupt Flag Write 1 to clear 0	RWC	0
9	MULT_DONE	Floating-point Multiplication Completion Interrupt Flag Write 1 to clear 0	RWC	0
8	BITREV_DONE	BITREV Mode Interrupt Flag Write 1 to clear 0	RWC	0
7	FFT_DONE	FFT Mode Interrupt Flag Write 1 to clear 0	RWC	0
6	BTFY_DMA_DONE	Butterfly Operation DMA Interrupt Flag Write 1 to clear 0	RWC	0
5	BTFY_ONCE_DONE	Single Butterfly Operation Interrupt Flag Write 1 to clear 0	RWC	0
4	F2I_DMA_DONE	Floating-point to Integer DMA Interrupt Flag Write 1 to clear 0	RWC	0
3	I2F_DMA_DONE	Integer to Floating-point DMA Interrupt Flag	RWC	0

		Write 1 to clear 0		
2	DIV_DONE	Division Unit Interrupt Flag Write 1 to clear 0	RWC	0
1	DMA_DONE	DMA Completion Signal Interrupt Flag Write 1 to clear 0	RWC	0
0	MAC_DONE	MAC Unit Calculation (MUL or BTFY_ONCE) Completion, write "1" to clear this bit	RWC	0

36.5.27 ALU_STA0（0x64）

Offset Address: 64H;

Bits	Bit Name	Function Description	R/W Flag	Reset Value
31:24	F2I_STATUS	Floating-point to Integer Unit Operation Status Flag Definition same as "MAC Unit 0 Operation Status Flag"	R	1
23:16	I2F_STATUS	Integer to Floating-point Unit Operation Status Flag Definition same as "MAC Unit 0 Operation Status Flag"	R	1
15:8	ADDSUB1_STATUS	MAC Unit 1 Operation Status Flag Definition same as "MAC Unit 0 Operation Status Flag"	R	1
7:0	ADDSUB0_STATUS	MAC Unit 0 Operation Status Flag bit0: Integer or floating-point output is zero. bit1: Floating-point output is infinity. bit2: Floating-point operation is invalid. Also set to 1 when one input is NaN. bit3: Floating-point number is less than the minimum normalized number. bit4: Floating-point number is greater than the maximum normalized number. bit5: Integer or floating-point output is not equal to the infinitely precise result. bit6: The magnitude of the rounded integer result is greater than the largest representable two's complement integer with the same sign. bit7: Reserved.	R	1

36.5.28 ALU_STA1（0x68）

Offset Address: 68H;

Bits	Bit Name	Function Description	R/W Flag	Reset Value
31:16	Reserved	Reserved.	R	0
15:8	DIV_STATUS	Division Unit Operation Status, definition same as "MAC Unit 0 Operation Status Flag"	R	1
7:0	MUL_STATUS	Multiplication Unit Operation Status, definition same as "MAC Unit 0 Operation Status Flag"	R	1

36.5.29 CRD_CTL（0x6C）

Offset Address: 6CH; Default Value: 0x0

Bits	Bit Name	Function Description	R/W Flag	Reset Value
31:2	Reserved	Reserved.	R	0
1	CRD_ROT_KICK	Cordic Rotation Mode Start Signal	WO	0
0	CRD_VEC_KICK	Cordic Vector Mode Start Signal	WO	0

This register has no read function, write-only.

36.5.30 CRD_XIN（0x70）

Offset Address: 70H; Default Value: 0x0

Bits	Bit Name	Function Description	R/W Flag	Reset Value
31:0	X_IN	Cordic Vector Mode Data X Input Port	RW	0

36.5.31 CRD_YIN（0x74）

Offset Address: 74H; Default Value: 0x0

Bits	Bit Name	Function Description	R/W Flag	Reset Value
31:0	Y_IN	Cordic Vector Mode Data Y Input Port	RW	0

36.5.32 CRD_AMP（0x78）

Offset Address: 78H; Default Value: 0x0

Bits	Bit Name	Function Description	R/W Flag	Reset Value
31：0	AMP_OUT	Cordic Vector Mode Data Output Port $\sqrt{X_0^2 + Y_0^2}$ /4*P (P=0.60725)	R	0

36.5.33 CRD_PHASE（0x7C）

Offset Address: 7CH; Default Value: 0x0

Bits	Bit Name	Function Description	R/W Flag	Reset Value
31：0	PHASE_OUT	Cordic Vector Mode Data Output Port Register Value =$\arctan(\frac{Y_0}{X_0})$	R	0

36.5.34 CRD_ANGLE（0x80）

Offset Address: 80H; Default Value: 0x0

Bits	Bit Name	Function Description	R/W Flag	Reset Value
31:0	ANGLE_IN	Cordic Rotation Mode Radian Input, range [-π : π]	RW	0

		Input -1*2^31 represents -π, input 1*2^31 represents π Register Value = radian * 2^31 / π		

36.5.35 CRD_COSINE（0x84）

Offset Address: 84H; Default Value: 0x0

Bits	Bit Name	Function Description	R/W Flag	Reset Value
31:0	COSINE_OUT	Cordic Rotation Mode Cosine Output Register Value = cos(angle) * (2^(31-2)) / (2*π) angle = radian * 180 / π	R	0

36.5.36 CRD_SINE（0x88）

Offset Address: 88H; Default Value: 0x0

Bits	Bit Name	Function Description	R/W Flag	Reset Value
31:0	SINE_OUT	Cordic Rotation Mode Sine Output Register Value = sin(angle) * (2^(31-2)) / (2*π)	R	0

36.5.37 CRD_IE（0x8C）

Offset Address: 8CH; Default Value: 0x0

Bits	Bit Name	Function Description	R/W Flag	Reset Value
31:2	Reserved	Reserved.	R	0
1	CRD_ROT_IE	Cordic Rotation Mode Completion Interrupt Enable Bit	RW	0
0	CRD_VEC_IE	Cordic Vector Mode Completion Interrupt Enable Bit	RW	0

36.5.38 CRD_FLG（0x90）

Offset Address: 90H; Default Value: 0x0

Bits	Bit Name	Function Description	R/W Flag	Reset Value
31:2	Reserved	Reserved.	R	0
1	CRD_ROT_DONE	Cordic Rotation Mode Completion Flag Write 1 to clear 0	RW	0
0	CRD_VEC_DONE	Cordic Vector Mode Completion Flag Write 1 to clear 0	RW	0

36.5.39 INTP_LEN（0x94）

Offset Address: 94H; Default Value: 0x0

Bits	Bit Name	Function Description	R/W Flag	Reset Value
31:13	Reserved	Reserved.	R	0
12:0	INTP_LEN	Interpolation Algorithm Actual Available Input Data Length, i.e., number of x[n]	RW	0

		Point Configuration n, Length = (n+1) Word. (Integer type)		

36.5.40 INTP_LOC（0x98）

Offset Address: 98H; Default Value: 0x0

Bits	Bit Name	Function Description	R/W Flag	Reset Value
31:0	INTP_LOC	Interpolation Algorithm Current Interpolation Location (Floating-point type)	RW	0

36.5.41 INTP_STEP（0x9C）

Offset Address: 9CH; Default Value:0x0

Bits	Bit Name	Function Description	R/W Flag	Reset Value
31:0	INTP_STEP	Interpolation Algorithm Step (Floating-point type)	RW	0

36.5.42 PFMODE（0xA0）

Offset Address: A0H; Default Value: 0x1

Bits	Bit Name	Function Description	R/W Flag	Reset Value
31:1	Reserved	Reserved.	R	0
0	PFM_SEL	DSP Module High-Performance Mode Selection (Clock Down Configuration) =0, Clock down frequency closed, DSP module calculation speed faster; =1, Clock down frequency opened (DSP module system clock divided by 2), DSP module calculation speed slower. Note: 1. When the DSP module is allocated to Core1, and the Core1 system clock is 59Mhz, this register can only be configured as 1, enabling DSP module frequency reduction. 2. When the DSP module is allocated to Core0, this register configuration is invalid, the DSP module clock is always equal to the Core0 system clock.	RW	1

36.6 Software Operation Process

For specific implementation, please refer to the routines provided by MAKSA. The following descriptions are for learning reference only.

36.6.1 Complete Solution Description

- ADC sampling data (integer d0) is cached in SRAM, with a bit width of 24-bit two's complement format.
- Data preprocessing: Normalize the ADC data (two's complement format) and convert it to signed floating-point format d1, which ranges from -1 to 1.
- FFT operation: Use all converted numbers d1 as inputs for FFT operation in complex number format, where the

real part is d1 and the imaginary part is 0. The FFT operation yields FFT output results, which are complex numbers with real part Re and imaginary part Im both represented in floating-point format.

- Calculate harmonic content:

repeat(6){

$Y_0 = \sqrt{Re0 * Re0 + Im0 * Im0}0$,

for(n=1;n<42;n++)

{

$Y_n = \sqrt{Ren * Ren + Imn * Imn}$,

Yin=k*Yn/Y0,

}

}

Calculate the 41st harmonic content. Separate harmonic contents Yu (voltage) and Yi (current) need to be calculated.

- Phase angle calculation and harmonic power:

repeat(3)

{

for(n=1;n<42;n++)

{

Aun=atan(Ren/Imn), // For voltage data

Ain=atan*(Ren/Imn), // For current data

A=((Au-Ai)+A0*n)*180/pi,

PFn=cos(A*pi/180)

}

}

Calculate 40 phase angles (excluding the fundamental wave), requiring an accuracy of 0.01 degrees.

- Calculate harmonic power: Phn=FSA*Yun*Yin*PFn. Accumulate after calculating 41 harmonics. FSA is obtained from external sources.

36.6.2 Operation Flow

- Cache ADC sampling data sdata0 to SRAM, with each data point being 24-bit;
- Use the i2f_dma mode to convert sdata1 in SRAM to floating-point format and normalize to obtain sdata2. Note that for each word address in SRAM, only the 3rd byte is valid data, so configure MAC_CTL0[22:21] = 1 and ignore the highest byte;
- Use the bitrev_mode to perform BITREV on sdata2 to get sdata3;
- Use the FFT calculation module to perform FFT on sdata3 to obtain results (the software needs to write the FFT calculation table to SRAM in advance);
- Use the hardware CORDIC module and divider module to calculate harmonic content and harmonic power.

37 RTC

37.1 Overview

The RTC (Real-Time Clock) module provides functions including real-time clock, oscillator temperature compensation, calendar, alarm, and clock pulse output.

The real-time clock tracks time using independent second, minute, and hour registers. The calendar includes year, month, day, and weekday registers with automatic leap year and leap month correction. The clock pulse output offers multiple selectable frequencies for clock calibration and provides alarm functionality.

It integrates a temperature sensor to deliver digital temperature measurement results.

The module is powered independently by VBAT.

37.2 Features

- Provides accurate temperature values with a measurement accuracy of ±1℃ within the range of -25℃～70℃;
- Implements initial RTC calibration at room temperature;
- Automatically performs RTC temperature compensation without CPU involvement;
- Low-power design;
- Frequency adjustment accuracy up to 0.01697ppm;
- High-stability oscillator;
- RTC remains active in all modes and operates normally under low power;
- Offers clock and calendar functions: output registers include seconds, minutes, hours, date, month, year, weekday, and sub-seconds;
- Automatic leap year and leap month adjustment with a timekeeping range of 100 years (00-99);
- Supports timestamp function;
- 1 alarm interrupt, 2 timer periodic interrupts, and 5 time interrupts (second, minute, hour, month, day);
- Configurable timer and CPU linkage, when the CPU is halted, timer pause count;
- Output of uncalibrated 32768Hz frequency;
- Output of calibrated 1Hz/4Hz/8Hz/16Hz/(1/30)Hz frequencies from LOSC clock;
- Output of calibrated 1Hz/2Hz/4Hz/8Hz/16Hz frequencies from PLL clock;
- Supports RTC fourth-order curve temperature compensation; call Raynen Micro library functions to modify compensation parameters in practical use.
- Supports 2 independent tamper detection circuits (TAMPER1 and TAMPER2);
- Tamper detection circuits support millisecond/second-level filtering with a millisecond range of 34~64ms and a second range of 1~8s;
- Supports tamper detection flag query and interrupt enable;
- Supports KEYOUT function: triggers KEYOUT pin to output low level when a tamper event occurs, used to control the power supply of external EEPROM.
- Supports timed KEYOUT trigger function with selectable timing periods of 1~64s or 1~64min, and timed trigger flag query;
- KEYOUT maintains low level output for 2s after being triggered before release, with support for software-initiated early release of KEYOUT;
- Adds 32-byte readable and writable registers for storing power-related data, which will not be reset by CPU or VBAT BOR.

37.3 Registers

Base Addresses of the RTC Module

Module Name	Secure Mapping Address	Non-Secure Mapping Address
RTC	0x200B4000	0xA00B4000

37.3.1 Register List

Register Name	Address Offset	Register Name
RTC Register Group		
RTC_CTRL	Offset+0x00	RTC Control Register
RTC_SC	Offset+0x04	Second Register, Write-Protected
RTC_MN	Offset+0x08	Minute Register, Write-Protected
RTC_HR	Offset+0x0C	Hour Register, Write-Protected
RTC_DT	Offset+0x10	Date Register, Write-Protected
RTC_MO	Offset+0x14	Month Register, Write-Protected
RTC_YR	Offset+0x18	Year Register, Write-Protected
RTC_DW	Offset+0x1C	Weekday Register, Write-Protected
RTC_SSC	Offset+0x158	
Note: After the CPU writes to the calendar register group, it needs to wait 300uS to read the written value.		
RTC_CNT1	Offset+0x20	Timer 1 Register
RTC_CNT2	Offset+0x24	Timer 2 Register
RTC_SCA	Offset+0x28	Second Alarm Register
RTC_MNA	Offset+0x2C	Minute Alarm Register
RTC_HRA	Offset+0x30	Hour Alarm Register
RTC_IE	Offset+0x34	RTC Interrupt Enable Register
RTC_IF	Offset+0x38	RTC Status Register
RTC_TEMP	Offset+0x3C	Current Temperature Register, Readable and Writable, Write-Protected
RTC_TEMP2	Offset+0xF8	Current Temperature Register 2, 12-bit, Read-Only
TPS_START	Offset+0x150	Temperature Measurement Start Register
RTC_TEMP3	Offset+0x154	Current Temperature Register 3, 12-bit, Read-Only
RTC_TEMPOS	Offset+0xC4	RTC Temperature Offset Calibration Register, Write-Protected
RTC_TPSIN	Offset+0xC8	ADC Measurement Value Input for RTC Automatic Temperature Compensation
PLLOUT_CFG	Offset_0x168	PLL 1Hz Configuration Register
LOSC_CFG1	Offset+0x6C	LOSC Configuration Register 1
Compensation Register Group		
RTC_PS	Offset+0x40	Compensation Register Group Password Protection
RTC_DOTA0	Offset+0x48	Initial Frequency Deviation Calibration

		Register
RTC_ALPHAL	Offset+0x4C	Oscillator Low-Temperature Range Temperature Coefficient Register
RTC_ALPHAH	Offset+0x50	Oscillator High-Temperature Range Temperature Coefficient Register
RTC_XT0	Offset+0x54	Oscillator Temperature Curve Vertex Temperature Register
RTC_TADJ	Offset+0x58	Mass Production Temperature Single-Point Calibration Register
RTC_ZT	Offset+0x60	0°C Calibration Value, ADC Offset Calibration Value
RTC_MODE1	Offset+0x130	=16'hEA65: Enable fourth-order curve compensation, use registers 0x134~0x144=Other values: Use second-order curve compensation, effective registers are 0x4C/0x50/0x54
RTC_XT1	Offset+0x134	Oscillator Temperature Curve Vertex Temperature Register
RTC_ALPHA	Offset+0x138	Oscillator First-Order Temperature Coefficient Register
RTC_BETA	Offset+0x13C	Oscillator Second-Order Temperature Coefficient Register
RTC_GAMMA	Offset+0x140	Oscillator Third-Order Temperature Coefficient Register
RTC_ZETA	Offset+0x144	Oscillator Fourth-Order Temperature Coefficient Lower Register
RTC_ZETA_H	Offset+0x148	Oscillator Fourth-Order Temperature Coefficient Upper Register
Result Output Register Group		
RTC_DOTAT	Offset+0x64	dota(T)=dota0 + alpha*(temp-xt0)^2
RTC_FPTR	Offset+0x74	PLL Compensation Register
RTC_FDTR1S	Offset+0x78	Per 1S Compensation Register
RTC_FDTR30S	Offset+0x7C	Per 30S Compensation Register
RTC_FDTR120S	Offset+0x84	Per 120S Compensation Register
Secondary Compensation Register Group		
RTC_CALPS	Offset+0xCC	RTC Secondary Compensation Enable Register; write 8'hA8 to activate registers T0~T9
RTC_CAL_T0	Offset+0xD0	T0~T9 are 8-bit registers for secondary compensation of RTC errors based on hardware automatic temperature compensation, with a scale of 0.25ppm;Compensation Temperature Range:

		T < -30℃
RTC_CAL_T1	Offset+0xD4	Compensation Temperature Range: -30℃<= T <-20℃
RTC_CAL_T2	Offset+0xD8	Compensation Temperature Range: -20℃<= T < -10℃
RTC_CAL_T3	Offset+0xDC	Compensation Temperature Range: -10℃<= T < 0℃
RTC_CAL_T4	Offset+0xE0	Compensation Temperature Range: 0℃<= T <= 10℃
RTC_CAL_T5	Offset+0xE4	Compensation Temperature Range: 35℃< T <= 45℃
RTC_CAL_T6	Offset+0xE8	Compensation Temperature Range: 45℃< T <= 55℃
RTC_CAL_T7	Offset+0xEC	Compensation Temperature Range: 55℃< T <= 65℃
RTC_CAL_T8	Offset+0xF0	Compensation Temperature Range: 65℃< T <= 75℃
RTC_CAL_T9	Offset+0xF4	Compensation Temperature Range: 75℃< T <=85℃
RTC IO Port Control Register Group		
RTC_IOEN	Offset+0x88	IO Port Function Enable Register
RTC_IOMODE	Offset+0x8C	IO Port Mode Selection Register
TAMPERIE	Offset+0x160	Tamper Detection Interrupt Enable Register
TAMPERIF	Offset_0x164	Tamper Detection Interrupt Flag Register
KEYOUT_TK_CFG	Offset_0x1D4	KEYOUT Timed Trigger Configuration Register
KEYOUT_SREL	Offset_0x1D8	Software Release KEYOUT Register
Tamper Detection Freeze Time Register Group		
TAMPER1N0_TIME0	Offset+0x90	TAMPER1 First Falling Edge Frozen Time (Minute, Second) Register
TAMPER1N0_TIME1	Offset+0x94	TAMPER1 First Falling Edge Frozen Time (Date, Hour) Register
TAMPER1N0_TIME2	Offset+0x98	TAMPER1 First Falling Edge Frozen Time (Year, Month) Register
TAMPER1P0_TIME0	Offset+0x9C	TAMPER1 First Rising Edge Frozen Time (Minute, Second) Register
TAMPER1P0_TIME1	Offset+0xA0	TAMPER1 First Rising Edge Frozen Time (Date, Hour) Register
TAMPER1P0_TIME2	Offset+0xA4	TAMPER1 First Rising Edge Frozen Time (Year, Month) Register
TAMPER1N1_TIME0	Offset+0xA8	TAMPER1 Second Falling Edge Frozen Time (Minute, Second) Register
TAMPER1N1_TIME1	Offset+0xAC	TAMPER1 Second Falling Edge Frozen

		Time (Date, Hour) Register
TAMPER1N1_TIME2	Offset+0xB0	TAMPER1 Second Falling Edge Frozen Time (Year, Month) Register
TAMPER1P1_TIME0	Offset+0xB4	TAMPER1 Second Rising Edge Frozen Time (Minute, Second) Register
TAMPER1P1_TIME1	Offset+0xB8	TAMPER1 Second Rising Edge Frozen Time (Date, Hour) Register
TAMPER1P1_TIME2	Offset+0xBC	TAMPER1 Second Rising Edge Frozen Time (Year, Month) Register
TAMPER2N0_TIME0	Offset+0x100	TAMPER2 First Falling Edge Frozen Time (Minute, Second) Register
TAMPER2N0_TIME1	Offset+0x104	TAMPER2 First Falling Edge Frozen Time (Date, Hour) Register
TAMPER2N0_TIME2	Offset+0x108	TAMPER2 First Falling Edge Frozen Time (Year, Month) Register
TAMPER2P0_TIME0	Offset+0x10C	TAMPER2 First Rising Edge Frozen Time (Minute, Second) Register
TAMPER2P0_TIME1	Offset+0x110	TAMPER2 First Rising Edge Frozen Time (Date, Hour) Register
TAMPER2P0_TIME2	Offset+0x114	TAMPER2 First Rising Edge Frozen Time (Year, Month) Register
TAMPER2N1_TIME0	Offset+0x118	TAMPER2 Second Falling Edge Frozen Time (Minute, Second) Register
TAMPER2N1_TIME1	Offset+0x11C	TAMPER2 Second Falling Edge Frozen Time (Date, Hour) Register
TAMPER2N1_TIME2	Offset+0x120	TAMPER2 Second Falling Edge Frozen Time (Year, Month) Register
TAMPER2P1_TIME0	Offset+0x124	TAMPER2 Second Rising Edge Frozen Time (Minute, Second) Register
TAMPER2P1_TIME1	Offset+0x128	TAMPER2 Second Rising Edge Frozen Time (Date, Hour) Register
TAMPER2P1_TIME2	Offset+0x12C	TAMPER2 Second Rising Edge Frozen Time (Year, Month) Register
User-Defined Register Group		
BKPPS	Offset+0x190	User-Defined Register Write Protection Register
BKP0	Offset+0x194	User-Defined Register 0, 16-bit
BKP1	Offset+0x198	User-Defined Register 1, 16-bit
BKP2	Offset+0x19C	User-Defined Register 2, 16-bit
BKP3	Offset+0x1A0	User-Defined Register 3, 16-bit
BKP4	Offset+0x1A4	User-Defined Register 4, 16-bit
BKP5	Offset+0x1A8	User-Defined Register 5, 16-bit
BKP6	Offset+0x1AC	User-Defined Register 6, 16-bit

BKP7	Offset+0x1B0	User-Defined Register 7, 16-bit
BKP8	Offset+0x1B4	User-Defined Register 8, 16-bit
BKP9	Offset+0x1B8	User-Defined Register 9, 16-bit
BKP10	Offset+0x1BC	User-Defined Register 10, 16-bit
BKP11	Offset+0x1C0	User-Defined Register 11, 16-bit
BKP12	Offset+0x1C4	User-Defined Register 12, 16-bit
BKP13	Offset+0x1C8	User-Defined Register 13, 16-bit
BKP14	Offset+0x1CC	User-Defined Register 14, 16-bit
BKP15	Offset+0x1D0	User-Defined Register 15, 16-bit

37.3.2 RTC Register Group

37.3.2.1 RTC_CTRL（0x00）

Bit	Name	Description	R/W	Reset Value
31:13	---	Reserved	R	0
12	STF_TRIG	Software Time Freeze Function 0: Invalid 1: Trigger time freeze function, hardware automatically clears; Triggering the time freeze function stores the current year, month, day, hour, minute, second, sub-second values in registers TAMPER2N1_TIME2 ~TAMPER2P1_TIME2 (0x120~0x12C); If the second tamper detection function is enabled, the STF_TRIG function is invalid; After enabling software freeze function, need to wait 100us or read TAMPER_IF[4].STFIF as 1 before reading 0x120~0x12C registers; RTC_IOEN[11:8]. PL02EN write 0xE will also clear the frozen time;	WO	0
11	---	Reserved, do not configure	RW	0
10	CALBUSY	RTC Correction Calculation Busy. =1 indicates RTC is performing correction calculation; =0 indicates RTC correction calculation is complete. This bit is read-only, it also sets high during each correction, no practical significance.	R	0
9	WRBUSY	RTC Register Write Operation Busy. Writing multiple registers does not require waiting. But after writing and reading back, need to wait for WRBUSY bit to change from 1 to 0 before reading the correct value. Therefore, it is recommended that the application uniformly waits 300us after writing the perpetual calendar before reading (including RTC_SSC register).	R	0
8	WRTC	RTC Register Group Write Allow: 0: Disable RTC register write operation; 1: Allow RTC register write operation.	RW	0

		Note: This bit is effective for RTC register group 00~1C/3C/C4, and also for RTC_CTRL[7:0], RTC_CTRL[12:11], PLLOUT_CFG. Two methods for writing perpetual calendar time registers: 1) Write in the order "year, month, day, hour, minute, second". When the second register is written, time starts accumulating from the written moment. Note that this method has the possibility of minute rollover before the second is written, so read back for verification after writing; 2) Write in the order "second, minute, hour, year, month, day". Write the second register first, the perpetual calendar counter is cleared. As long as other values are written within one second, successful writing can be guaranteed. Note 1: Hardware has legality judgment for "year, month, day". Cannot write in the order "day, month, year", must write "year, month, day" consecutively. Note 2: Writing to the perpetual calendar time registers must follow the above order, and one time write must write all 6 registers (year, month, day, hour, minute, second), cannot update only part (e.g., only write minute and second).		
7:6	TSE	Temperature Sensor Enable Bit 00: Disable automatic temperature compensation. Only when TSE=00, compensation register group 44~60H are writable. 01: Start automatic temperature compensation. Perform periodic temperature compensation according to TCP settings. 10: Start user temperature compensation mode 0, temperature register can be changed, filled by the user, each write to the temperature register starts one temperature compensation; 11: Start user temperature compensation mode 1, temperature register cannot be changed, each write to the temperature register starts one temperature compensation operation, the temperature register value is obtained by SOC measurement. Note 1: In user temperature compensation mode 0, need to wait until RTC correction is not busy to start temperature compensation, i.e., wait until bit10 of RTC_CTRL register is 0 before writing the temperature register. Note 2: In user temperature compensation mode 1, need to wait until SAR ADC is not busy to start temperature compensation, i.e., wait until both bit10 of RTC_CTRL register and bit1 of SAR_STATUS register are 0 before writing the temperature register.	RW	00
5:3	TCP	Temperature Compensation Period: 000: 2s 001: 10s (default) 010: 20s 011: 30s 100: 1 minute 101: 2 minutes	RW	001

		110: 5 minutes 111: 10 minutes		
2:0	FOUT	RTC Output Signal Control Bit 000: No output 001: Output PLL divided signal, supports output 1Hz/2Hz/4Hz/8Hz/16Hz, specific frequency determined by PLLOUT_CFG; 010: Output LOSC divided 1/30Hz 011: Output LOSC 32768Hz 100: Output LOSC divided 16Hz 101: Output LOSC divided 8Hz 110: Output LOSC divided 4Hz 111: Output LOSC divided 1Hz. Note: When the relevant GPIO is configured as RTCOUT function, the GPIO output is controlled by the FOUT configuration bit.	RW	000

37.3.2.2 RTC_SC（0x04）

Bit	Name	Description	R/W	Reset Value
31:07	---	Reserved	R	0
6:0	SC	Stores the second value of the clock BCD code format, SC[6:4] is the tens digit of the second value, SC[3:0] is the units digit of the second value, second value range is 0~59	RW	-

37.3.2.3 RTC_MN（0x08）

Bit	Name	Description	R/W	Reset Value
31:07	---	Reserved	R	0
6:0	MN	Stores the minute value of the clock BCD code format, MN[6:4] is the tens digit of the minute value, MN[3:0] is the units digit of the minute value, minute value range is 0~59	RW	-

37.3.2.4 RTC_HR（0x0C）

Bit	Name	Description	R/W	Reset Value
31:06	---	Reserved	R	0
5:0	HR	Stores the hour value of the clock BCD code format, HR[5:4] is the tens digit of the hour value, HR[3:0] is the units digit of the hour value, hour value range is 0~23.	RW	-

37.3.2.5 RTC_DT（0x10）

Bit	Name	Description	R/W	Reset Value
31:06	---	Reserved	R	0
5:0	DT	Stores the date value of the clock BCD code format, DT[5:4] is the tens digit of the date value, DT[3:0] is the units digit of the date value, date value range is 1~31.	RW	-

37.3.2.6 RTC_MO（0x14）

Bit	Name	Description	R/W	Reset Value
31:05	---	Reserved	R	0
4:0	MO	Stores the month value of the clock BCD code format, MO[4] is the tens digit of the month value, M0[3:0] is the units digit of the month value, month value range is 1~12	RW	-

37.3.2.7 RTC_YR（0x18）

Bit	Name	Description	R/W	Reset Value
31:08	---	Reserved	R	0
7:0	YR	Stores the year value of the clock BCD code format, YR[7:4] is the tens digit of the year value, YR[3:0] is the units digit of the year value, year value range is 0~99.	RW	-

37.3.2.8 RTC_DW（0x1C）

Bit	Name	Description	R/W	Reset Value
31:03	---	Reserved	R	0
2:0	DW	Stores the day of the week corresponding to the current date. DW[2:0] counting cycle is 0-1-2-3-4-5-6-0-1-2-...	RW	-

37.3.2.9 RTC_SSC （0x158）

Bit	Name	Description	R/W	Reset Value
31:16	---	Reserved	R	0
15:0	SSC	Stores the sub-second value of the clock Binary value, 1LSB=30.518us When the system clock is 32K, directly reading the sub-second register will cause errors, so it is recommended not to read the sub-second register under this system clock.	R	-

37.3.2.10 RTC_CNT1（0x20）

Bit	Name	Description	R/W	Reset Value
31:11	---	Reserved	R	0
10	DBGSTB	Timer 1 and CPU Linkage Halt Function 0: CPU entering halted state does not affect Timer 1 counting 1: CPU entering halted state, Timer 1 pauses counting; CPU exiting halted state, Timer 1 continues counting;	RW	0
9	--	Reserved	R	0
8	CNT1PD	Timer 1 Switch Signal 0: Enable Timer 1 1: Disable Timer 1 Note: 1-second timing comes from the perpetual calendar second interrupt. After disabling and re-enabling, the counter will restart counting (scale 1s)	RW	0
7:0	CNT1	Timer 1 Counter Preset Value Unsigned number, counting unit is 1s. When count value = (CNT+1), set RTCCNT1F flag. (Minimum can generate an interrupt every 1 second, maximum can generate an interrupt every 256 seconds) Note1: This timer is accurate after RTC correction. Note2: The 1s source is determined by the configuration of RTC_IE.RTC_1S_SEL bit. Note3: Because the timer count is synchronized with seconds, after starting the timer, the first timing has an error, maximum error is 1s, subsequent timing is accurate.	RW	0

37.3.2.11 RTC_CNT2（0x24）

Bit	Name	Description	R/W	Reset Value
31:11	---	Reserved	R	0
10	DBGSTB	Timer 2 and CPU Linkage Halt Function 0: CPU entering halted state does not affect Timer 2 counting 1: CPU entering halted state, Timer 2 pauses counting; CPU exiting halted state, Timer 2 continues counting;	RW	0
9	CNT2_CLK_SEL	Timer 2 Clock Source Selection 0: 32768Hz 1: RCL	RW	0
8	CNT2PD	Timer 2 Switch Signal 0: Enable Timer 2 1: Disable Timer 2, re-enabling will restart the counter (scale	RW	0

		1/256s)		
7:0	CNT2	Timer 2 Counter Preset Value Unsigned number, counting unit is 1/256s. When count value = (CNT+1), set RTCCNT2F flag. (Minimum can generate an interrupt every 1/256 second, maximum can generate an interrupt every 1 second) Note1: This timer originates from the 32768Hz crystal,uncorrect has some error. Note2: Because the timer count is synchronized with seconds, after starting the timer, the first timing has an error, maximum error is 1/256s, subsequent timing is accurate.	RW	0

37.3.2.12 RTC_SCA（0x28）

Bit	Name	Description	R/W	Reset Value
31:07	---	Reserved	R	0
6:0	SCA	Second Alarm Value BCD code format, SCA[6:4] is the tens digit of the second value, SCA[3:0] is the units digit of the second value, second value range is 0~59	RW	0

37.3.2.13 RTC_MNA（0x2C）

Bit	Name	Description	R/W	Reset Value
31:07	---	Reserved	R	0
6:0	MNA	Minute Alarm Value BCD code format, MNA[6:4] is the tens digit of the minute value, MNA[3:0] is the units digit of the minute value, minute value range is 0~59	RW	0

37.3.2.14 RTC_HRA（0x30）

Bit	Name	Description	R/W	Reset Value
31:06	---	Reserved	R	0
5:0	HRA	Hour Alarm Value BCD code format, HRA[5:4] is the tens digit of the hour value, HRA[3:0] is the units digit of the hour value, hour value range is 0~23.	RW	0

37.3.2.15 RTC_IE（0x34）

Bit	Name	Description	R/W	Reset

				Value
31:10	---	Reserved	R	0
9	RTC_1S_SEL	Second Interrupt and CNT1 Counting Source Selection: =0: According to system clock mode: select pll_1hz in high frequency mode, select RTC second pulse in non-high frequency mode =1: Select RTC second pulse Note1: pll_1hz and second register update are not synchronized, RTC second pulse is synchronized with the second register. It is recommended that customers configure this register to 1, select RTC second pulse. Note2: The second interrupt and CNT1 count source are both configured by RTC_1S_SEL, but the two functions are independent functions;	RW	0
8	IECLKEN	RTC Interrupt Generation Clock Enable; When any bit of RTC_IE[8:0] is high, the interrupt module clock is turned on; When all bits of RTC_IE[8:0] are low, the interrupt module clock is turned off; =0, not enable =1, enable Note: Clearing the interrupt flag bits of RTC_IF also requires turning on the interrupt module clock;	RW	0
7	MOIE	Month Interrupt Enable 0 : Disable 1 : Enable	RW	0
6	DTIE	Date Interrupt Enable 0 : Disable 1 : Enable	RW	0
5	HRIE	Hour Interrupt Enable 0 : Disable 1 : Enable	RW	0
4	MNIE	Minute Interrupt Enable 0 : Disable 1 : Enable	RW	0
3	SCIE	Second Interrupt Enable 0 : Disable 1 : Enable Note: When RTC-IE.RTC_1S_SEL is 0, i.e., select pll_1hz as the second interrupt source, the second interrupt is not synchronized with the second register update, the second interrupt may not be generated at the second count starting point, can be generated at any moment during the second count. When RTC_IE.	RW	0

		RTC_1S_SEL is 1, i.e., select RTC second pulse as the second interrupt source, the second interrupt is consistent with the second register update. It is recommended to select RTC_IE. RTC_1S_SEL as 1.		
2	RTCCNT2IE	RTC Timer 2 Interrupt Enable 0 : Disable 1 : Enable	RW	0
1	RTCCNT1IE	RTC Timer 1 Interrupt Enable 0 : Disable 1 : Enable	RW	0
0	ALMIE	Alarm Event Interrupt Enable 0 : Disable 1 : Enable	RW	0

37.3.2.16 RTC_IF（0x38）

Bit	Name	Description	R/W	Reset Value
31:12	---	Reserved	R	0
7	MOF	Month Interrupt Flag Bit 0 : Month counter did not increment by 1 1 : Month counter incremented by 1 Note: Write 1 to clear	RW	0
6	DTF	Date Interrupt Flag Bit 0 : Date counter did not increment by 1 1 : Date counter incremented by 1 Note: Write 1 to clear	RW	0
5	HRF	Hour Interrupt Flag Bit 0 : Hour counter did not increment by 1 1 : Hour counter incremented by 1 Note: Write 1 to clear	RW	0
4	MNF	Minute Interrupt Flag Bit 0 : Minute counter did not increment by 1 1 : Minute counter incremented by 1 Note: Write 1 to clear	RW	0
3	SCF	Second Interrupt Flag Bit 0 : Second counter did not increment by 1 1 : Second counter incremented by 1 Note: Write 1 to clear	RW	0
2	RTCCNT2F	RTC Timer 2 Interrupt Flag Bit 0 : Timer 2 interrupt did not occur 1 : Timer 2 interrupt occurred Note: Write 1 to clear	RW	0
1	RTCCNT1F	RTC Timer 1 Interrupt Flag Bit	RW	0

		0 : Timer 1 interrupt did not occur 1 : Timer 1 interrupt occurred Note: Write 1 to clear		
0	ALMF	Alarm Event Flag Bit, alarm event matching real-time clock occurred 0 : Alarm event did not occur 1 : Alarm event occurred Note: Write 1 to clear	RW	0

When the interrupt module clock is turned on, the interrupt flag register flags take effect. Refer to the configuration of RTC_IE register to choose whether to turn on the clock.

When the interrupt module clock is turned off, the interrupt flag bits cannot be cleared by writing 1; please clear the interrupt flag bits by writing 1 before turning off the interrupt module clock.

37.3.2.17 RTC_TEMP（0x3C）

Bit	Name	Description	R/W	Reset Value
31:10	---	Reserved	R	0

15:00	TEMP	Current temperature value.
		Bit9 is the sign bit; Bit8~2 are the integer bits; Bit1~0 are the fractional bits.
		// Temp[9] Temp[8:2] Temp[1] Temp[0]
		// Sign -128℃~127℃ 0.5℃ 0.25℃
		Representation range: -128℃~+127.75℃
		Temperature conversion formula: If sign bit is 0, then temperature = TEMP/4; If sign bit is 1, then temperature = (TEMP-2^10)/4
		TSE=00: Disable automatic temperature compensation. This register is invalid, read value is meaningless.
		TSE=01: Periodic automatic temperature compensation according to TCP settings. This register displays the temperature value of the measurement cycle.
		TSE=10: Start user temperature compensation mode 0, temperature register can be changed, filled by the user, each write to the temperature register starts one temperature compensation.
		TSE=11: Start user temperature compensation mode 1, temperature register cannot be changed, each write to the temperature register starts one temperature compensation operation, the temperature register value is obtained by MCU measurement.
		Note1: When bit10 of RTC_CTRL register is 0, then RTC_TEMP register can be written.
		Note2: In user temperature compensation mode 0, and when RTC_MODE1=16'hEA65, enable quartic curve compensation, then TEMP is extended to 16bit number, bit15 is the sign bit, minimum scale is 0.25℃/2^6=0.00390625℃; Otherwise still defaults to 10bit
			RW	-

		writable register. After TEMP is extended to 16bit, temperature conversion formula: If sign bit is 0, then temperature = TEMP/2^8; If sign bit is 1, then temperature = (TEMP-2^16)/2^8。		

37.3.2.18 RTC_TEMP2（0xF8）

Bit	Name	Description	R/W	Reset Value
31:12	---	Reserved	R	0
11:0	TEMP2	Current temperature value. Bit11 is the sign bit; Bit10~4 are the integer bits; Bit3~0 are the fractional bits. // Sign -128℃~127℃ 0.5℃ 0.25℃ 0.125℃ 0.0625℃ Representation range: -128℃~+127.75℃ Temperature conversion formula: If sign bit is 0, then temperature = TEMP2/16; If sign bit is 1, then temperature = (TEMP2-2^12)/16	R	-

37.3.2.19 RTC_TEMPOS（0xC4）

Bit	Name	Description	R/W	Reset Value
31:06	---	Reserved	R	0
5:0	TEMPOS	When TSE equals 01 or 11, the value read from temperature register TEMP = actual calculated value + TEMPOS/4. TEMPOS is a signed number, used for temperature offset correction. This register is in two's complement format, Bit5 is the sign bit, correction scale 0.25℃, to change the temperature value by t (℃), should fill in TEMPOS = t*4; Correction range is: -8℃~+7.75℃.	RW	0

37.3.2.20 RTC_TPSIN（0xC8）

Bit	Name	Description	R/W	Reset Value
31:13	---	Reserved	R	0
12:0	TPSIN	This register stores the measurement value input by SAR ADC to RTC. Refer to the description about TPSIN in the analog peripheral module. RTC automatic temperature compensation temperature measurement method (changing bias current) selection: 000: First select bias current as 2uA, test 5 values and sum to get D1; then select bias current as 4uA, test 5 values and sum to get D2; get measurement value ADCO=2*D1-D2 passed to RTC; RTC divides ADCO by 5 to get TPSIN, then calculates temperature value.	R	-

		001: Select bias current as 2uA, test 5 values and sum to get ADCO; ADCO passed to RTC; RTC divides ADCO by 5 to get TPSIN, then calculates temperature value. 010: Select bias current as 4uA, test 5 values and sum to get ADCO; ADCO passed to RTC; RTC divides ADCO by 5 to get TPSIN, then calculates temperature value. 011: Same as 00 100: Select ibias current as 2uA, test 10 values, average to get TPSIN; 101: Select ibias current as 4uA, test 10 values, average to get TPSIN; 110: Select ibias current as 2uA, test 20 values, average to get TPSIN; 111: Select ibias current as 4uA, test 20 values, average to get TPSIN; Note: The specific value of TPSIN can be read in the RTC module.		

37.3.2.21 TPS_START（0x150）

Bit	Name	Description	R/W	Reset Value
31:1	---	Reserved	R	0
0	START	1: Start SAR for TPS temperature measurement, and the digital circuit calculates the temperature value according to the temperature formula, output to temperature register RTC_TEMP3. 0: Invalid Note1: This register is only valid in user temperature compensation mode 0, only starts temperature measurement without temperature compensation, convenient for the application to obtain temperature for software averaging. Note2: After temperature measurement starts, need to wait 4.5ms to complete temperature measurement; need to query RTC->CTRL bit10 (Cal_busy bit) (cannot query immediately) after starting measurement for 4.5ms, when it is 0 then obtain RTC_TEMP3 temperature register 3 value; query timeout time, recommended to be above 10ms.	WO	0

37.3.2.22 RTC_TEMP3（0x154）

Bit	Name	Description	R/W	Reset Value
31:12	---	Reserved	R	0
11:0	TEMP	Temperature Register	R	0

		In user temperature compensation mode 0, after software configures TPS_START signal to start temperature measurement, wait for CALBUSY=0, then the temperature value calculated from this temperature measurement can be obtained from this register.
		Bit11 is the sign bit; Bit10~4 are the integer bits; Bit3~0 are the fractional bits.
		// Temp[11] Temp[10:4] Temp[3] Temp[2] Temp[1] Temp[0]
		// Sign -128℃~127℃ 0.5℃ 0.25℃ 0.125℃ 0.0625℃
		Representation range: -128℃~+127.75℃
		Temperature conversion formula: If sign bit is 0, then temperature = TEMP2/16; If sign bit is 1, then temperature = (TEMP2-2^12)/16		

37.3.2.23 PLLOUT_CFG（0x168）

PLLOUT Configuration Register Offset Address=0x168

This register is a VCC domain register, uses VCC domain clock and reset, affected by CPU reset, not affected by VBAT reset;

Register is write protected by RTC_CTRL[8];

Bit	Name	Description	R/W	Reset Value
31:3	---	Reserved	R	0
2:0	FREQ_SEL	PLL Divided Signal Output Selection 3'b000: Output 1Hz 3'b001: Output 2Hz 3'b010: Output 4Hz 3'b011: Output 8Hz 3'b100: Output 16Hz Others: Output 1Hz; Note: This configuration bit selects the PLL divided signal, controlled by the FOUT configuration bit of RTC_CTRL register whether to output.	RW	0

37.3.2.24 LOSC_CFG1（0x6C）

Bit	Name	Description	R/W	Reset Value
31:16	---	Reserved	R	0
10	VCC_BOI_FLAG	=0: Normal; =1, VCC has experienced power loss; Write 1 to clear When using this flag, during power-on, software needs to clear this flag first, then proceed with the	RW	0

		application.		
9	LOSC_FAIL	=0: LOSC normal =1: LOSC has experienced oscillation stop Write 1 to clear. When using this flag, during power-on, software needs to clear this flag first, then proceed with the application.	RW	0
8	LOSC_WEN	=0: LOSC_PD not writable =1: LOSC_PD writable When LOSC_WEN is 1, bit[7:0] are writable.	RW	0
7:0	LOSC_PD	LOSC Enable Bit: =Others: LOSC on; =8'ha8: LOSC off. Note: Also the external clock injection enable signal, when LOSC_PD is effective, enable external clock injection. Design idea: Write: When LOSC_PD is written, the 8bit written value of LOSC_PD is copied 3 times and stored in a 24bit register. When this 24bit register equals 24'ha8a8a8, pwd_losc is effective, turn off LOSC, otherwise do not turn off LOSC. Read back: When the three 8bit backup values in the 24bit register are equal, read back the low 8bit value, otherwise read 0. Steps to turn off LOSC and enable external clock injection function: LOSC_CFG1 = (1<<8); LOSC_CFG1 = (0xa8<<0);	RW	0

Note: If using external injected 32.768kHz clock, need to inject from XI, and need to turn off the chip's oscillation circuit (i.e., turn off LOSC enable bit).

37.3.3 Quadratic Curve Compensation Register Group

37.3.3.1 RTC_PS（0x40）

Bit	Name	Description	R/W	Reset Value
31:08	---	Reserved	R	0
7:0	RTC_PS	When RTC_PS=8'h8E, compensation registers 44H~60H, 80H, 130H~148H are writable; When RTC_PS=other values, compensation registers 44H~60H, 80H, 130H~148H are not writable.	RW	0

Quadratic curve compensation register group：

37.3.3.2 RTC_DOTA0（0x48）（boot configuration）

Bit	Name	Description	R/W	Reset Value
31:16	---	Reserved	R	0
15:0	DOTA0	Initial frequency deviation correction value DOTA0 is in signed two's complement format, bit12 is the sign bit DOTA0 = round(σ0*32), where σ0 unit is ppm, representable range: -1024ppm --1023.96785ppm. Resolution is 0.03125ppm. Example: Initial frequency deviation σ0=-6ppm, correction value DOTA0=0xff40 Note: Most 32768 crystal typical value is ±20ppm, maximum value is ±30ppm)	RW	0

Note: Only VBAT power-on/power-off reset will restore this register to default value, other resets will not restore.

When the user calibrates the initial frequency deviation of RTC, it is recommended to call the MAKSA library function "void RtcWriteDota(uint16_t dota);", this library function not only has the calibration function for the DOTA0 register, but also has the function of effectively managing the vbat domain configuration registers, which can ensure that the VBAT domain related parameters are correctly written, which is especially important for applications where VBAT powers up slower than VCC.

37.3.3.3 RTC_ALPHAL（0x4C）（boot configuration）

Bit	Name	Description	R/W	Reset Value
31:11	---	Reserved	R	0
10:00	ALPHAL	Crystal frequency temperature coefficient low temperature part temperature coefficient, i.e., the temperature coefficient from -40℃ to T0 this segment. (Format is unsigned binary) ALPHAL = round(α *2048*16) (Unit is ppm/℃², generally temperature coefficient is -0.034ppm/℃²) (α representable range: 0 -- 0.0625 ppm/℃²)	RW	0

Only VBAT power-on/power-off reset can take effect, bootrom will modify this value.

0.034*2048*16*（T-T0）^2=0.034*2048*16*（-40-25）^2=4707123.2 approximately 143.65ppm

round(0.034*2048*16)*（T-T0）^2=round(0.034*2048*16)*（-40-25）^2=1114*4225=4706650 approximately 143.635ppm

The error is 0.02ppm

37.3.3.4 RTC_ALPHAH（0x50）（boot configuration）

Bit	Name	Description	R/W	Reset Value
31:11	---	Reserved	R	0
10:00	ALPHAH	Crystal frequency temperature coefficient high temperature part temperature coefficient, i.e., the temperature coefficient from T0 to 85℃ this segment. (Format is unsigned binary)	RW	0

		ALPHAH = round(α *2048*16) (Unit is ppm/℃², generally temperature coefficient is -0.034ppm/℃²) (α representable range: 0 -- 0.0625 ppm/℃²)		

Only VBAT power-on/power-off reset can take effect, bootrom will modify this value.

37.3.3.5 RTC_XT0（0x54）（boot configuration）

Bit	Name	Description	R/W	Reset Value
31:07	---	Reserved	R	0
6:0	XT0	Crystal temperature curve (quadratic curve) vertex temperature value (Format is unsigned number) XT0 = round(T_0*4) (T_0 unit is ℃) (T_0 representable range: 0 -- 31.75℃) (Resolution is 0.25 degree. Actually should write value of 25±5.	RW	-

37.3.3.6 RTC_TADJ（0x58）（boot configuration）

Bit	Name	Description	R/W	Reset Value
31:10	---	Reserved	R	0
10:0	TADJ	Mass production temperature single point correction parameter (Format is unsigned binary) Mass production platform calculates correction parameter according to TADJ = $\frac{T_0 + 273}{D_0 + 1021}$. DATA = round(2^11*TADJ*4)(TADJ unit is ℃/LSB) (TADJ representable range: 0 -- 0.249. The slope k of the temperature curve is in 0.1859 -- 0.2055 corresponds to 0℃ temperature deviation of 260-290k) (1℃ change is about 1ppm. TADJ accuracy is 0.0001, can guarantee temperature calculation result accuracy of 0.125℃)	RW	-

37.3.3.7 RTC_ZT（0x60）（boot configuration）

Bit	Name	Description	R/W	Reset Value
31:15	---	Reserved	R	-
14:09	ADCOS[5:0]	This register performs offset correction on the ADC sampling value for calculating temperature, ADC sampling value is 10bit, ADCOS is added aligned with its low bits, ADCOS[5] is the sign bit.	RW	-
08:00	ZT	Reserved option for design, generally, this register should be written with 273 (decimal).	RW	-

37.3.4 Quartic Curve Compensation Register Group

Quartic curve compensation register group register configuration, need to first configure RTC_PS=8'h8E, after write enable takes effect then can configure.

Quartic curve compensation initial deviation correction is the same as quadratic curve compensation register RTC_DOTA0, application needs attention.

37.3.4.1 RTC_MODE1（0x130）

Bit	Name	Description	R/W	Reset Value
31:16	---	Reserved	R	0
15:0	CURVE_SEL	Temperature Compensation Curve Selection =16'hEA65: Enable quartic curve compensation, use registers 0x134~0x144 =Other values: Use quadratic curve compensation, effective registers are 0x4C/0x50/0x54.	RW	0

37.3.4.2 RTC_XT1（0x134）

Bit	Name	Description	R/W	Reset Value
31:13	---	Reserved	R	0
12:0	XT1	Crystal temperature curve (quadratic curve) vertex temperature value (Format is unsigned number) XT0 = round(T0*256) (T0 unit is ℃) (T0 representable range: 0 -- 31.99609375℃) (Resolution is 0.00390625 degree). Typical value is 25℃.	RW	-

37.3.4.3 RTC_ALPHA（0x138）

Bit	Name	Description	R/W	Reset Value
31:16	---	Reserved	R	0
15:0	ALPHA	Crystal frequency temperature coefficient first order coefficient α, typical value is -0.017ppm/℃, maximum 0.012, minimum -0.052. (Format is signed binary, note first order coefficient may be positive or negative. Unit is ppm/℃) If α is positive, then: ALPHA = round(α *2^15*8) If α is negative, then: ALPHA = round(2^16+α *2^15*8) α is represented by 16-bit signed number, normalized value amplified by 8 times, representable range is: ±0~0.125ppm/℃ One LSB represents: 1/2^18=3.8*10^-6 ppm/℃, Quantization error maximum impact: LSB*(40+25)=0.00025ppm, can be ignored.	RW	0

37.3.4.4 RTC_BETA（0x13C）

Bit	Name	Description	R/W	Reset Value
31:15	---	Reserved	R	0
14:0	BETA	Crystal frequency temperature coefficient second order coefficient β, typical value is -0.0334ppm/℃², maximum -0.0332, minimum -0.0336. (Second order coefficients are all negative, format defined as unsigned binary. Unit is ppm/℃²) β is normally negative, register should write: BETA = round(-β*2^15*16) β is represented by 16-bit unsigned number, normalized value amplified by 16 times, representable range is: 0~-0.0625 ppm/℃, note bit15 can only be 0. One LSB represents: 1/2^19=1.9073*10^-6 ppm/℃, Quantization error maximum impact: LSB*(40+25)^2=0.00806ppm, can be ignored.	RW	0

37.3.4.5 RTC_GAMMA（0x140）

Bit	Name	Description	R/W	Reset Value
31:16	---	Reserved	R	0
15:0	GAMMA	Crystal frequency temperature coefficient third order coefficient γ, Seiko crystal parameter table typical value is -4.9*10^-5ppm/℃³, maximum -4.4*10^-5ppm/℃³, minimum -5.2*10^-5ppm/℃³. Measured value is -6.3*10^-5ppm/℃³. (Format still uses signed binary representation) γ is normally negative, then register should write: GAMMA = round(2^16+γ*2^15*2048*4) γ is represented by 16-bit signed number, normalized value amplified by 8192 times, representable range is: ±0~12.207*10^-5 ppm/℃ One LSB represents: 1/2^28=0.37253*10^-8 ppm/℃, Quantization error maximum impact: LSB*(40+25)^3=0.001023ppm, can be ignored.	RW	0

37.3.4.6 RTC_ZETA（0x144）

Bit	Name	Description	R/W	Reset Value

31:16	---	Reserved	R	0
15:0	ZETA	{ZETA_H[1:0],ZETA[15:0]} is the crystal frequency temperature coefficient fourth order coefficient ζ, measured value is 4.692437*10^-7ppm/°C⁴. Seiko crystal parameter table has no such parameter. (Format still uses signed binary representation) If ζ is positive, then register should write: ZETA = round(ζ*2^15*65536*8) If ζ is negative, then register should write: ZETA = round(2^16+ζ*2^15*65536*8) ζ is represented by 18-bit signed number, normalized value amplified by 65536*8 times, representable range is: ±0~7.6294*10^-6 ppm/°C. Note: Actually quartic curve does not need such large representation range, filling too large value will cause DotaT register overflow, no practical significance. One LSB represents: 1/2^34=5.8208*10^-11 ppm/°C, Quantization error maximum impact: LSB*(40+25)^4=0.001039ppm, can be ignored.	RW	0

Note: Reference material listed parameter calculation is 1.22*10^-8

37.3.4.7 RTC_ZETA_H（0x148）

Bit	Name	Description	R/W	Reset Value
31:2	---	Reserved	R	0
1:0	ZETA_H	Crystal frequency temperature coefficient fourth order coefficient ζ high 2bit value, specific description please see RTC_ZETA	RW	0

Note: Reference material listed parameter calculation is 1.22*10^-8

37.3.5 Result Output Register Group

37.3.5.1 RTC_DOTAT（0x64）

Bit	Name	Description	R/W	Reset Value
31:16	---	Reserved	R	0

15:0	DOTAT	Quadratic curve calculation principle:
		dotat[15:0]/32=dota0[12:0] /32+
		alpha[10:0]/ (2048*16)* (temp[9:0]/4-{3'h0,xt0[6:0] }/4)^2
		
		// dota0[12:0]=round[dota00*32] Range:-128ppm –127.96875ppm
		// alph[10:0]=round[alph*2048*16] Range:alph=0~0.034~0.0625(no sign)
		// temp[9:0]=T*4(sign) Range:-128°C~+127.75°C
		// xt0=T0*4(no sign) Range:0 – 31.5°C
			R	-

		Data format:
		15 14~5 4:0
		Sign ppm 0.03125ppm
		Representation range:-1024ppm ~ 1023.96785ppm
		Actual representation range:
		Dota0 is generally 20~30ppm, alpha is about 0.034, temp range is -40~85℃, xt0 is generally about 25℃.
		Dotat actual range is generally within 128ppm.
		
		Quartic curve calculation principle:
		dotat[15:0]/32=dota0[12:0]/32 +
		alpha[15:0]/(2^15*8) * (temp[9:0]/4- xt1[12:0]/256)+
		beta[14:0]/(-2^15*16)* (temp[9:0]/4-xt1[12:0] / 256)^2+
		gamma[15:0]/(2^15*2048*4) * (temp[9:0]/4-{xt1[12:0]/256 })^3+
		zeta[17:0]/(2^15*65536*8) * (temp[9:0]/4-{xt1[12:0]/256})^4
		
		// dota0[12:0]=round[dota00*32] Range:-128ppm –127.96875ppm
		// alph[15:0]=round[alph*2^15*8] Range:±0~0.125 ppm/℃ (sign)
		//beta[14:0]=round(-beta*2^15*16) Range:0~-0.0625 ppm/℃（no sign）
		//gamma[15:0]=round(2^16+gamma*2^15*2048*4) Range:±0~12.207^-5 ppm/℃(sign)
		//zeta[15:0]=round(2^16+zeta*2^15*65536*8) Range:±0~7.6294^-6 ppm/℃(sign)
		// temp[9:0]=T*4(sign) Range:-128℃~+127.75℃
		// xt1=T0*256(no sign) Range:0 – 31.99609375℃）		

37.3.5.2 RTC_ FPTR（0x74）

Bit	Name	Description	R/W	Reset Value
31:17	---	Reserved	R	0
16:0	FPTR	PLL Compensation Register fptr = fix(dota(T)/0.01697ppm) Signed number, represents the number of clock cycles that need to be compensated every 1S in PLL mode.	R	0

37.3.5.3 RTC_ FDTR1S（0x78）

Bit	Name	Description	R/W	Reset Value
31:6	---	Reserved	R	0
5:0	RTC_FDTR1S	Per 1S Compensation Register RTC_FDTR1S = fix(dotat/30.5175)	R	0

		Negative values need to be converted to two's complement format.		

37.3.5.4 RTC_ FDTR30S（0x7C）

Bit	Name	Description	R/W	Reset Value
31:7	---	Reserved	R	0
6:0	RTC_FDTR30S	Digital compensation amount every 30s: Per 30S Compensation Register 1S compensation remainder A = (dotat - FDTR1S*30.5175) Additional compensation amount needed every 30S B = fix(A/1.017257) Every 30S compensation register value FDTR30S = FDRT1S + B Example: If dotat=50ppm, then FDTR1S=1, 1S compensation remainder A = (50 - fix(50/30.5)*30.5)=19.5 Additional compensation amount needed every 30S B = fix(19.5/1.01725)=19 Every 30S compensation register value FDTR30S = FDRT1S +19=20 If dotat=-50ppm, then FDTR1S=-1, 1S compensation remainder A = (-50 - fix(-50/30.5)*30.5)=-19.5 Additional compensation amount needed every 30S B = fix(-19.5/1.01725)=-19 Every 30S compensation register value FDTR30S = FDRT1S -19=-20 Negative values need to be converted to two's complement format.	R	0

37.3.5.5 RTC_ FDTR120S（0x84）

Bit	Name	Description	R/W	Reset Value
31:9	---	Reserved	R	0
8:0	RTC_FDTR120S	Digital compensation amount every 120s: Per 120S Compensation Register 1S and 30S compensation remainder A = (dotat - FDTR1S*30.5 - (FDTR30S - FDTR1S)*1.01725) Additional compensation amount needed every 120S B = fix(A/0.254313) Every 120S compensation register value FDTR120S = FDTR30S+B	R	-

37.3.6 Secondary Compensation Register Group

37.3.6.1 RTC_CALPS（0xCC）

Secondary Compensation Enable Register Offset Address = 0xCC

Bit	Name	Description	R/W	Reset Value
31:8	---	Reserved	R	0
7:0	CALPS	RTC secondary compensation enable bit =8 'HA8, temperature compensation calculation is added to registers T0 to T9 For other values, T0 to T9 will not be included in the temperature compensation calculation	RW	0

37.3.6.2 RTC_CAL_T0~T9（0xD0~0xF4）

Bit	Name	Description	R/W	Reset Value
31:8	---	Reserved	R	0
7:0	CAL_T0~T9	Perform secondary compensation on RTC error based on hardware automatic temperature compensation, T0~T9 registers are in two's complement format, Bit7 is the sign bit, scale is 0.25ppm; T0 compensation temperature range: T < -30℃ T1 compensation temperature range: -30℃<= T <-20℃ T2 compensation temperature range: -20℃<= T < -10℃ T3 compensation temperature range: -10℃<= T < 0℃ T4 compensation temperature range: 0℃<= T <= 10℃ T5 compensation temperature range: 35℃< T <= 45℃ T6 compensation temperature range: 45℃< T <= 55℃ T7 compensation temperature range: 55℃< T <= 65℃ T8 compensation temperature range: 65℃< T <= 75℃ T9 compensation temperature range: 75℃<="85℃	RW	0

37.3.7 RTC IO Control Register Group

37.3.7.1 RTC_IOEN（0x88）

RTC IO Function Enable Register Offset Address=0x88

This register is writable when RTC_IOMODE.PADEN_PS=4'hB.

Bit	Name	Description	R/W	Reset Value
31:16	--	Reserved	R	0
15:12	PL03EN	PL03 VBAT Domain Multiplex Function Configuration Determined by {ANA_PAD[2],RTC_IOEN[15:12]}, priority higher than VCC domain configuration =5'h1x: Select as SARADC AIN7 input =5'h0b: Select as KEYOUT output =Other values: Used as general GPIOL, function determined by	RW	0

		GPIOLx registers, refer to GPIO chapter description for details		
11:8	PL02EN	PL02 VBAT Domain Multiplex Function Configuration =4'h5: Select TAMPER2 input, and enable TAMPER2 tamper detection function, simultaneously supports GPIOL configuration as KEY0[7] =4'hE: Used as general GPIOL, and disable TAMPER2 tamper detection function, clear TAMPER2 register group 0x100~0x12C =4'hB: Select KEYOUT output, and disable TAMPER2 tamper detection function =Other values: Used as general GPIOL, and disable TAMPER2 tamper detection function Note: ANA_PAD[1] has higher priority, when ANA_PAD[1] is 1, PL02 selects SARADC AIN6 input; when ANA_PAD[1] is 0, PL02 is determined by PL02EN Note: When PL02 used as general GPIOL, function determined by GPIOLx registers, refer to GPIO chapter description for details	RW	0x0
7:4	PL00EN	PL00 VBAT Domain Multiplex Function Configuration, priority higher than VCC domain configuration =4'ha: Select as RTC_OUT output =4'hb: Select as KEYOUT output =Other values: Used as general GPIOL, function determined by GPIOLx registers, refer to GPIO chapter description for details	RW	0x0
3:0	PL01EN	PL01 VBAT Domain Multiplex Function Configuration =4'h5: Select TAMPER1 input, and enable TAMPER1 tamper detection function, simultaneously supports GPIOL configuration as KEY0[6] =4'hE: Used as general GPIOL, and disable TAMPER1 tamper detection function, clear TAMPER1 register group 0x90~0xBC =4'hB: Select KEYOUT output, and disable TAMPER1 tamper detection function =Other values: Used as general GPIOL, and disable TAMPER1 tamper detection function Note: ANA_PAD[0] has higher priority, when ANA_PAD[0] is 1, PL01 selects SARADC AIN5 input; when ANA_PAD[0] is 0, PL01 is determined by PL01EN Note: When PL01 used as general GPIOL, function determined by GPIOLx registers, refer to GPIO chapter description for details	RW	0x0

37.3.7.2 RTC_IOMODE（0x8C）

Bit	Name	Description	R/W	Reset Value
31:15	--	Reserved	R	0
14	TAMPER_FLT_PS	TAMPER_FLT_TS and TAMPER_FLT_CNT write password bit	RW	0x0

		=0: TAMPER_FLT_TS and TAMPER_FLT_CNT not writable, readable =1: TAMPER_FLT_TS and TAMPER_FLT_CNT writable, readable		
13	TAMPER_FLT_TS	TAMPER1/TAMPER2 Filter Accuracy Selection =0: Select 2ms filter accuracy =1: Select 250ms filter accuracy	RW	0x0
12:8	TAMPER_FLT_CNT	TAMPER1/TAMPER2 Filter Length Selection When TAMPER_FLT_TS is 0, TAMPER1/TAMPER2 filter length is (TAMPER_FLT_CNT[3:0]+1)*2ms+32ms, range 34~64ms, step 2ms When TAMPER_FLT_TS is 1, TAMPER1/TAMPER2 filter length is (TAMPER_FLT_CNT[4:2]+1)*1s, range 1~8s, step 1s	RW	0x0
7:4	PADEN_PS	RTC_IOEN Register Write Protection =4'hB: Enable RTC_IOEN write operation =Other values: Not enable	RW	0x0
3	--	Reserved	RO	0x0
2	KEYOUT_MODE	PL00~PL03 Multiplex Output KEYOUT Mode Configuration =0: KEYOUT outputs low level when triggered, PAD is high impedance when released =1: KEYOUT outputs low level when triggered, outputs high level when released	RW	0x0
1	TAMPER2_MODE	PL02/TAMPER2 Tamper Detection Mode Configuration =0: Record the earliest two tamper events after IOEN enable =1: Record every tamper event after IOEN enable, recording does not distinguish time sequence in cache, should be judged by software Before enabling RTC_IOEN, should first configure TAMPER2 mode Tamper detection filter time can be configured, refer to TAMPER_FLT_TS and TAMPER_FLT_CNT for details When configured as 0, the earliest two tamper events will link trigger KEYOUT output; when configured as 1, every tamper event will not link trigger KEYOUT	RW	0x0
0	TAMPER1_MODE	PL01/TAMPER1 Tamper Detection Mode Configuration =0: Record the earliest two tamper events after IOEN enable =1: Record every tamper event after IOEN enable, recording does not distinguish time sequence in cache, should be judged by software Before enabling RTC_IOEN, should first configure TAMPER1 mode Tamper detection filter time can be configured, refer to TAMPER_FLT_TS and TAMPER_FLT_CNT for details	RW	0x0

		When configured as 0, the earliest two tamper events will link trigger KEYOUT output; when configured as 1, every tamper event will not link trigger KEYOUT		

37.3.7.3 TAMPERIE（0x160）

Bit	Name	Description	R/W	Reset Value
31:6	--	Reserved	R	0
5	KEYOTKIE	KEYOUT Timer Trigger Interrupt Enable 0: Not enable 1: Enable	RW	0
4	STFIE	Software Time Freeze Interrupt Enable 0: Not enable 1: Enable	RW	0
3	TAMPER2NIE	TAMPER2 Falling Edge Interrupt Enable 0: Not enable 1: Enable	RW	0
2	TAMPER2PIE	TAMPER2 Rising Edge Interrupt Enable 0: Not enable 1: Enable	RW	0
1	TAMPER1NIE	TAMPER1 Falling Edge Interrupt Enable 0: Not enable 1: Enable	RW	0
0	TAMPER1PIE	TAMPER1 Rising Edge Interrupt Enable 0: Not enable 1: Enable	RW	0

37.3.7.4 TAMPERIF（0x164）

Bit	Name	Description	R/W	Reset Value
31:6	--	Reserved	R	0
5	KEYOTKIF	KEYOUT Timer Trigger Interrupt Flag Bit 0: Timer trigger event has not occurred 1: Timer trigger event has occurred This flag bit will maintain for 30us after being set, and the flag bit needs to wait for 2 bus clocks before release, recommended software execution in this interrupt program: If no write to RTC or analog peripheral registers occurred within 1ms, then first perform write operation to any register (ensure bus clock is enabled); if write to RTC or analog peripheral registers occurred within 1ms, then skip step 1) 2) Wait at least 35us then write 1 to clear this flag bit	RW1C	0

4	STFIF	Software Time Freeze Interrupt Flag Bit 0: This event has not occurred 1: This event has occurred Write 1 to clear this flag bit	RW1C	0
3	TAMPER2NIF	TAMPER2 Falling Edge Interrupt Flag Bit 0: This event has not occurred 1: This event has occurred This flag bit will maintain for 30us after being set, and the flag bit needs to wait for 2 bus clocks before release, recommended software execution in this interrupt program: If no write to RTC or analog peripheral registers occurred within 1ms, then first perform write operation to any register (ensure bus clock is enabled); if write to RTC or analog peripheral registers occurred within 1ms, then skip step 1) Wait at least 35us then write 1 to clear this flag bit	RW1C	0
2	TEMPER2PIF	TAMPER2 Rising Edge Interrupt Flag Bit 0: This event has not occurred 1: This event has occurred Clear processing same as TAMPER2NIF	RW1C	0
1	TEMPER1NIF	TAMPER1 Falling Edge Interrupt Flag Bit 0: This event has not occurred 1: This event has occurred Clear processing same as TAMPER2NIF	RW1C	0
0	TEMPER1PIF	TAMPER1 Rising Edge Interrupt Flag Bit 0: This event has not occurred 1: This event has occurred Clear processing same as TAMPER2NIF	RW1C	0

37.3.7.5 KEYOUT_TK_CFG（0x1D4）

31:17	--	Reserved	R	0
16	PS	KEYOUT_TK_CFG Other Configuration Bits Password Protection Bit =1: Register bits readable writable =0: Register bits readable not writable	RW	0
15	--	Reserved	R	0
14:9	CNT	KEYOUT Timer Count Value KEYOUT timer trigger time is (CNT+1)*TB	RW	0
8	TB	KEYOUT Timer Time Base =0: 1min =1: 1s	RW	0
7:0	EN	KEYOUT Timer Trigger Enable =0xA5: Enable KEYOUT timer trigger =Other values: Disable KEYOUT timer trigger	RW	0

		Note: After disabling KEYOUT timer trigger, KEYOUT will restore to default state Note: Timer trigger module detects enable configuration bit every 1ms, so after disabling KEYOUT timer trigger, delay at least 2ms before re-enabling KEYOUT timer trigger function		
31:17	--	Reserved	R	0

37.3.7.6 KEYOUT_SREL（0x1D8）

Bit	Name	Description	R/W	Reset Value
31:16	--	Reserved	R	0
15:8	KEYO_PS	KEYO_SREL Password Protection Bit =0xEC: KEYO_SREL writable readable =Other values: KEYO_SREL not writable readable When KEYOUT releases, KEYO_PS becomes 0	RW	0x0
7:0	KEYO_SREL	{KEYO_PS, KEYO_SREL} forms software release KEYOUT control bit =0xECF8: Software release KEYOUT =Other values: Invalid configuration When KEYOUT releases, KEYO_STOP becomes 0 Note: If software does not configure KEYO_SREL, then KEYOUT will automatically release after being triggered for 2s Note: When tamper trigger KEYOUT and software release KEYOUT occur simultaneously, tamper trigger KEYOUT priority higher than software release KEYOUT, KEYOUT will be in triggered state Note: When timer trigger KEYOUT and software release KEYOUT occur simultaneously, software release KEYOUT priority higher than timer trigger KEYOUT, KEYOUT will be in released state Note: After software configures register to release KEYOUT, need to read interrupt flag bit again to judge whether new tamper event has occurred	RW	0x0

37.3.8 Tamper Detection Freeze Time Register Group

37.3.8.1 TAMPER1 First Freeze Time

Freeze time data 16bit valid, all in BCD code format.

Offset Address	Name	Description	R/W	Reset Value
0x90	TAMPER1N0_TIME0	TAMPER1 First Falling Edge Freeze Time Minute, Second Register [15:8]: Minute [7:0]: Second	R	0

0x94	TAMPER1N0_TIME1	TAMPER1 First Falling Edge Freeze Time Day, Hour Register [15:8]: Day [7:0]: Hour	R	0
0x98	TAMPER1N0_TIME2	TAMPER1 First Falling Edge Freeze Time Year, Month Register [15:8]: Year [7:0]: Month	R	0
0x9C	TAMPER1P0_TIME0	TAMPER1 First Rising Edge Freeze Time Minute, Second Register [15:8]: Minute [7:0]: Second	R	0
0xA0	TAMPER1P0_TIME1	TAMPER1 First Rising Edge Freeze Time Day, Hour Register [15:8]: Day [7:0]: Hour	R	0
0xA4	TAMPER1P0_TIME2	TAMPER1 First Rising Edge Freeze Time Year, Month Register [15:8]: Year [7:0]: Month	R	0

37.3.8.2 TAMPER1 Second Freeze Time

Freeze time data 16bit valid, all in BCD code.

Offset Address	Name	Description	R/W	Reset Value
0xA8	TAMPER1N1_TIME0	TAMPER1 Second Falling Edge Freeze Time Minute, Second Register [15:8]: Minute [7:0]: Second	R	0
0xAC	TAMPER1N1_TIME1	TAMPER1 Second Falling Edge Freeze Time Day, Hour Register [15:8]: Day [7:0]: Hour	R	0
0xB0	TAMPER1N1_TIME2	TAMPER1 Second Falling Edge Freeze Time Year, Month Register [15:8]: Year [7:0]: Month	R	0
0xB4	TAMPER1P1_TIME0	TAMPER1 Second Rising Edge Freeze Time Minute, Second Register [15:8]: Minute [7:0]: Second	R	0
0xB8	TAMPER1P1_TIME1	TAMPER1 Second Rising Edge Freeze Time Day, Hour Register	R	0

		[15:8]: Day [7:0]: Hour		
0xBC	TAMPER1P1_TIME2	TAMPER1 Second Rising Edge Freeze Time Year, Month Register [15:8]: Year [7:0]: Month	R	0

37.3.8.3 TAMPER2 First Freeze Time

Freeze time data 16bit valid, all in BCD code.

Offset Address	Name	Description	R/W	Reset Value
0x100	TAMPER2N0_TIME0	TAMPER2 First Falling Edge Freeze Time Minute, Second Register [15:8]: Minute [7:0]: Second	R	0
0x104	TAMPER2N0_TIME1	TAMPER2 First Falling Edge Freeze Time Day, Hour Register [15:8]: Day [7:0]: Hour	R	0
0x108	TAMPER2N0_TIME2	TAMPER2 First Falling Edge Freeze Time Year, Month Register [15:8]: Year [7:0]: Month	R	0
0x10C	TAMPER2P0_TIME0	TAMPER2 First Rising Edge Freeze Time Minute, Second Register [15:8]: Minute [7:0]: Second	R	0
0x110	TAMPER2P0_TIME1	TAMPER2 First Rising Edge Freeze Time Day, Hour Register [15:8]: Day [7:0]: Hour	R	0
0x114	TAMPER2P0_TIME2	TAMPER2 First Rising Edge Freeze Time Year, Month Register [15:8]: Year [7:0]: Month	R	0

37.3.8.4 TAMPER2 Second Freeze Time

Freeze time data 16bit valid, all in BCD code.

Offset Address	Name	Description	R/W	Reset Value
0x118	TAMPER2N1_TIME0	TAMPER2 Second Falling Edge Freeze Time Minute, Second Register	R	0

		[15:8]: Minute [7:0]: Second		
0x11C	TAMPER2N1_TIME1	TAMPER2 Second Falling Edge Freeze Time Day, Hour Register [15:8]: Day [7:0]: Hour	R	0
0x120	TAMPER2N1_TIME2	TAMPER2 Second Falling Edge Freeze Time Year, Month Register [15:8]: Year [7:0]: Month When TAMPERIF[4] is 1, TAMPER2N1_TIME2 refers to the sub-second value of software triggered time freeze;	R	0
0x124	TAMPER2P1_TIME0	TAMPER2 Second Rising Edge Freeze Time Minute, Second Register [15:8]: Minute [7:0]: Second When TAMPERIF[4] is 1, TAMPER2P1_TIME0 refers to the minute and second values of software triggered time freeze;	R	0
0x128	TAMPER2P1_TIME1	TAMPER2 Second Rising Edge Freeze Time Day, Hour Register [15:8]: Day [7:0]: Hour When TAMPERIF[4] is 1, TAMPER2P1_TIME1 refers to the date and hour values of software triggered time freeze;	R	0
0x12C	TAMPER2P1_TIME2	TAMPER2 Second Rising Edge Freeze Time Year, Month Register [15:8]: Year [7:0]: Month When TAMPERIF[4] is 1, TAMPER2N1_TIME2 refers to the year and month values of software triggered time freeze;	R	0

37.3.9 User-defined Register Group

37.3.9.1 BKPPS（0x190）

User-defined Register Group Write Protection Register, this register is only affected by VBAT POR reset;

Bit	Name	Description	R/W	Reset Value
31:8	---	Reserved	R	0
7:0	BKP_PS	User-defined Register Group Write Protection =8'hFA, registers 0x194~0x1D0 readable writable	RW	0x0

		=Other values, registers 0x194~0x1D0 read only		

37.3.9.2 BKP0~BKP15（0x194~0x1D0）

User-defined Register 0~15, only affected by VBAT POR reset;

Bit	Name	Description	R/W	Reset Value
31:16	---	Reserved	R	0
15:0	BKP_DATA	User-defined Data 0~15	RW	0x0

37.4 KEYOUT Function Description

The KEYOUT function means that PL00/PL01/PL02/PL03 can be configured to output an indicator signal (KEYOUT) for PL01/TAMPER1 and PL02/TAMPER2 to detect the first two tamper events. The specific function description is as follows:

- Trigger Modes

  Tamper detection trigger and timed trigger

  In the tamper detection trigger mode, only when the tamper detection mode is set to record the first two events, and the first two events occurring after enabling will trigger KEYOUT to output a low level for 2 seconds. If the tamper detection mode is set to record every event, or for the third, fourth, and subsequent events after enabling, KEYOUT will not be triggered.

  In the timed trigger mode, the software configures the KEYOUT_TK_CFG register to enable the timed trigger function. After that, KEYOUT is triggered once every cycle.

- Output

  When PL00/PL01/PL02/PL03 is configured for the KEYOUT function, it is in the released state by default, outputting a high-impedance state. The KEYOUT_MODE configuration bit in the RTC_IOMODE register can be used to set the output to a high level when released.

  When KEYOUT is triggered, it outputs a low level and automatically releases to a high-impedance state (or high level, depending on the RTC_IOMODE.KEYOUT_MODE configuration bit) after 2 seconds by default. The software can configure the KEYOUT_SREL register to immediately release KEYOUT to a high-impedance state (or high level).

  Note that if KEYOUT is enabled to be triggered periodically every 1 second or 2 seconds, KEYOUT will remain in the triggered state and will not be released automatically.

37.5 RTC Clock Read/Write Steps

Core0 configures the 4th bit RTC_SARADC_SEC of the SCU_PERIPH_SEC_CFG1 register, which is used to set the security attribute of RTC. When RTC_SARADC_SEC = 1, RTC is a secure peripheral; when RTC_SARADC_SEC = 0, RTC is a non-secure peripheral.

- The following configurations are common to both Core0 and Core1:
- Set the 4th bit RTC_SARADC_CKE of the MOD1_CKE register in the system control chapter to 1 to enable the RTC bus clock.
- Clock reading: Read the time registers of RTC, such as second, minute, and hour registers.
- Clock writing:
  - Set the 8th bit WRTC of RTC_CTL to 1 to enable write operations.

- Write in the order of "year-month-day-hour-minute-second". After writing the second register, the time starts to accumulate from the writing moment. Note that there is a possibility of minute rollover before writing the second register, so it is necessary to read back and verify after writing.
- Or write in the order of "second-minute-hour-year-month-day". Write the second register first to clear the calendar counter. As long as other values are written within one second, successful writing can be ensured.

Note: The hardware performs validity checks on "year-month-day". Writing in the order of "day-month-year" is not allowed; "year-month-day" must be written consecutively. For improved software reliability, it is recommended to read back and confirm after writing.

37.6 RTC Calibration Steps

The user only needs to calibrate the initial deviation of the 32.768KHz (clock signal). The error is written through the Raynen Micro programmer interface or by using library functions.

37.7 RTC Timer Operation Steps

Take configuring Timer 1 to generate a 1-second interrupt as an example.

First, Core0 configures the security attribute of RTC according to application requirements and assigns the RTC interrupt to either Core0 or Core1:

- Write the 4th bit RTC_SARADC_SEC of the SCU_PERIPH_SEC_CFG1 register. When RTC_SARADC_SEC = 1, RTC is a secure peripheral; when RTC_SARADC_SEC = 0, RTC is a non-secure peripheral.
- Execute set_gic_irq(RTC_IRQn, 1, 0); // The second parameter is set to 1 to assign the interrupt to Core0; the third parameter can also be set to 1 to assign the interrupt to Core1.

The following operation steps are common to both Core0 and Core1:

- Set the 4th bit RTC_SARADC_CKE of the MOD1_CKE register in the system control chapter to 1 to enable the RTC bus clock.
- Set the 8th bit WRTC of RTC_CTL to 1 to enable write operations.
- Set RTC->CNT1 = 0x00; to generate an interrupt every 1 second.
- Set RTC->IE = 0x02; to enable the RTC Timer 1 interrupt.
- Enable the RTC interrupt:

ECLIC_Register_IRQ(RTC_IRQn, ECLIC_NON_VECTOR_INTERRUPT, ECLIC_LEVEL_TRIGGER, 1, 0, RTC_HANDLER);

- Write the interrupt service routine:

```
void RTC_HANDLER(void)
{
    if(RTC->IF & 0x02) // Check Timer 1
    {
        /* Start adding user code. Do not edit comment generated here */
    }
}
```

- After the configuration is completed, a 1-second interrupt can be generated.

38 Analog Peripherals

38.1 Overview

The multi-channel general-purpose ADC module includes a 16-channel analog multiplexer, an internal 1.25V reference voltage source (using an independent REF), and a 12-bit SAR ADC. It is used for temperature sensor measurement, battery voltage detection, main power supply voltage detection, general-purpose analog input, etc.

The 16-channel analog multiplexer includes channels AIN0–AIN9 for external analog signal measurement, VBAT for battery voltage sampling, Temp Sensor for measuring the output of the temperature sensor, and 4 reserved channels. Both AIN6 and VBAT adopt 1/2 voltage division input for sampling.

The 12-bit successive approximation ADC completes one conversion in 16 cycles, with the sampling clock derived from the divided clock of RCM 1.8M; it supports a maximum input of 1.25V. Gain options include 0.25x, 0.5x, 1x, 1.5x, and 2x.

The ADC automatically enters power-saving mode when not sampling.

38.2 System Block Diagram

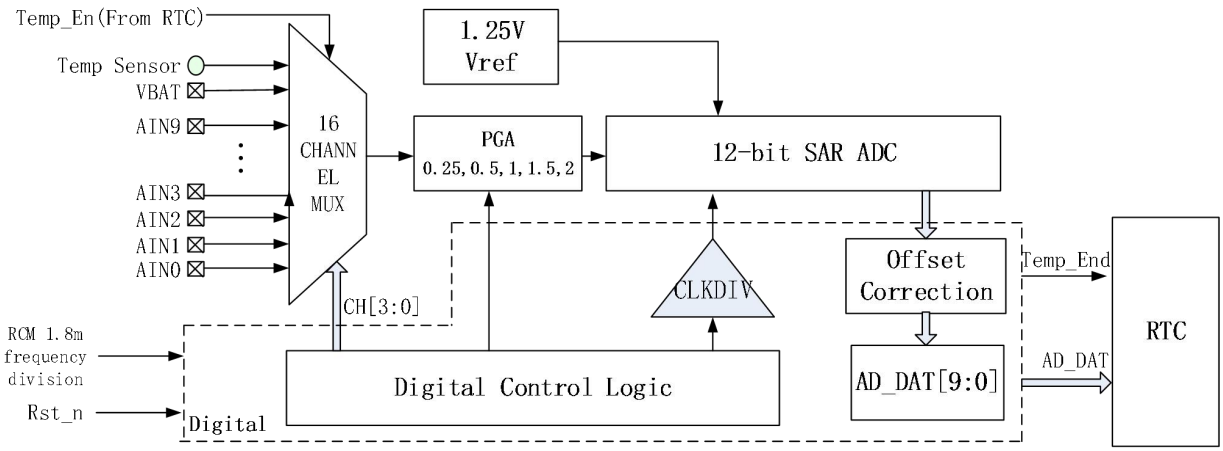


Figure 38-1 System Block Diagram

38.3 Registers

Base Addresses of the Analog Peripheral Module

Module Name	Secure Mapped Address	Non-secure Mapped Address
MADC	0x200B8000	0xA00B8000

38.3.1 Register List

Register Name	Address Offset	Description
AD_CTRL	Offset+0x0	SAR-ADC Control Register
AD_START	Offset+0x4	SAR-ADC Start Register
AD_STAT	Offset+0x8	SAR-ADC Status Register
AD_DATA	Offset+0xC	SAR-ADC Data Register, 10bit
AD_CTRL1	Offset+0x18	SAR-ADC Control Register 1
AD_DATA2	Offset+0x1C	SAR-ADC Data Register, 12bit

AD_CTRL3	Offset+0x20	SAR-ADC Control Register 3
AD_DATA3	Offset+0x24	SAR-ADC Data Register, 16bit
ANA_RST_STAT	Offset+0x28	VBAT Domain Reset Status Register
ANA_PAD	Offset+0x3C	VBAT Domain Analog PAD Configuration Register
ANA_RST	Offset+0x40	VBAT Domain Software Reset Register

38.3.2 AD_CTRL（0x0）

Bit	Name	Description	R/W Flag	Reset Value
31:17	---	Reserved	R	0
16:12	REF_WAIT	Wait time from turning on REF to turning on ADC: Minimum time unit: REF_WAIT_T0 (TBD) Wait time value: (REF_WAIT+1) * REF_WAIT_T0 (TBD) 5'd0: REF_WAIT_T0 (TBD) 5'd31: (31+1) * REF_WAIT_T0 (TBD)	RW	0x5
11:7	SAR_WAIT	Wait time required after SAR ADC is turned on and before starting sampling conversion: 5'd0: 30.5μs 5'd31: (31+1) * 30.5μs Wait time = (SAR_WAIT+1) * 30.5μs Note: Steps to start ADC measurement: Turn on REF, wait for REF_WAIT time; Turn on ADC and temperature sensor, wait for SAR_WAIT time; The above steps are automatically controlled by hardware. Software judges whether the ADC measurement is completed via the flag bit. Due to internal clock deviation, if software sets a timeout wait, the timeout wait time is recommended to have a 3-5 times margin. See the example routine.	RW	0xE
6	SAR_CH3	SAR-ADC Channel selection, forms a 4-bit register together with SAR_CH[2:0]. Refer to SAR_CH[2:0] description.	RW	0
5	SAR_IE	SAR-ADC Interrupt Control: 1: Enable ADC interrupt output; 0: Disable ADC interrupt output.	RW	0
4:3	SAR_PGA	SAR-ADC Gain Control: 00: 0.5x 01: 1x 10: 1.5x 11: 2x When AD_CTRL1.SAR_PGA_SEL=0, SAR-ADC gain is determined by SAR_PGA[1:0]; when AD_CTRL1.SAR_PGA_SEL=1, SAR-ADC gain is determined by AD_CTRL1.SAR_PGA_2[2:0].	RW	0

2:0	SAR_CH	SAR-ADC Channel selection, forms a 4-bit value with bit6 SAR_CH3, i.e., {SAR_CH3, SAR_CH[2:0]} for SAR channel selection. 0000: Temp Sensor 0001: VBAT (After 1/2 divider gets 1.8V, PGA uses 0.5x, measuring input 0.9V) 0010: AIN0, input from PA00 0011: AIN1, input from PA01 0100: AIN2, input from PA02 0101: AIN3, input from PA03 0110: AIN4, input from PA04 0111: AIN5, input from PL01 1000: AIN6, input from PL02 after 1/2 divider then input to SARADC 1001: AIN7, input from PL03 1010: AIN8, input from PA11 1011: AIN9, input from PA12 1100: Reserved 1101: Reserved 1110: Reserved 1111: Reserved Note: Regardless of the channel set, automatic temperature measurement has the highest priority.	RW	0
Note: The above register bits are writable only when ST in AD_START is 0.				

38.3.3 AD_START（0x4）

Bit	Name	Description	R/W Flag	Reset Value
31:01	---	Reserved	R	0
0	ST	SAR-ADC Start Bit 0: SAR-ADC no operation 1: Start one SAR-ADC sampling, automatically cleared after sampling is completed. Note1: Automatic temperature measurement controlled by RTC is not controlled by this bit and has higher priority than this configuration bit; when TPS_BUSY=1, this register is not writable. Note2: When this bit is 1, software is prohibited from writing 1 again to start SAR-ADC measurement; software needs to query this flag bit, and can only start a new ADC measurement operation when this flag bit is 0; recommended timeout wait time = 2*(REF_WAIT + SAR_WAIT) + 3ms.	RW	0

38.3.4 AD_STAT（0x8）

Bit	Name	Description	R/W Flag	Reset Value
31:2	---	Reserved	R	0
1	TPS_BUSY	Automatic temperature measurement busy, =1: Automatic temperature measurement is in progress; =0: Automatic temperature measurement is not in progress. When TPS_BUSY=1, SAR_START register is not writable.	R	0
0	DREADY	ADC Date Ready Pending Bit 0: ADC conversion result not completed 1: ADC conversion result completed Note: Write 1 to clear; Automatic temperature measurement controlled by RTC is not indicated in this status.	RW	0

38.3.5 AD_DATA（0xC）

Bit	Name	Description	R/W Flag	Reset Value
15:10	---	Reserved	R	0
9:0	SAR_DAT	ADC Conversion Result	R	0

Keep 10bit register unchanged.

38.3.6 AD_CTRL1（0x18）

Bit	Name	Description	R/W Flag	Reset Value
31:15	---	Reserved	R	0x0
14:8	SAR_CONVERT	Wait time required from SARADC sampling conversion to digital sample DOUT output: 6'd0: 3.906μs ... 6'd127: (127+1) * 3.906μs Wait time = (SAR_CONVERT +1) * 3.906μs Note: Must comply with SAR_CONVERT[6:0] >= 16 * 2 * 3.906μs; modifying this configuration bit is not recommended.	RW	0x29
7	--	Reserved	R	0x0
6:4	Reserved	Reserved, not recommended to operate	RW	0x0
3	SAR_PGA_SEL	SAR-ADC Gain Control Register Selection =0, selects SAR_CTL[4:3] as the SAR-ADC gain register =1, selects SAR_CTL1[2:0] as the SAR-ADC gain register	RW	0x0
2:0	SAR_PGA	SAR-ADC Gain Control Bits 000: 0.5x 001: 1x 010: 1.5x	RW	0x0

		011: 2x 1xx: 0.25x		

38.3.7 AD_DATA2（0x1C）

Bit	Name	Description	R/W Flag	Reset Value
15:12	---	Reserved	R	0
11:0	SAR_DAT2	ADC Conversion Result, 12bit	R	0

38.3.8 AD_CTRL3（0x20）

Bit	Name	Description	R/W Flag	Reset Value
31:8	---	Reserved	R	0x0
7:0	ACC_CNT	SAR ADC output data accumulation point configuration =0, 1 point =1, 2 points accumulation ... =N, N+1 points accumulation Maximum support 256 points accumulation	RW	0x0

38.3.9 AD_DATA3（0x24）

Bit	Name	Description	R/W Flag	Reset Value
31:16	---	Reserved	R	0
15:0	SAR_DAT3	ADC sampling data conversion result after accumulation points (ACC_CNT+1)	R	0

38.3.10 ANA_RST_STAT（0x28）

Bit	Name	Description	R/W Flag	Reset Value
31:5	---	Reserved	R	0
4	BORIE	VBAT BOR Reset Interrupt Enable Bit =0, Disable =1, Enable This bit can be reset by VBAT POR reset and VBAT soft reset.	RW	0
3:2	---	Reserved	R	0
1	BORIF	VBAT BOR Reset Flag Register; write 1 to clear. =0, No reset =1, Reset occurred If BORIE=1 and BORIF=1, an interrupt will be triggered; writing 1 to BORIF clears this flag.	RW	-
0	PORIF	VBAT POR Reset Flag Register; write 1 to clear.	RW	-

		=0, No reset =1, Reset occurred If PORIF=1, an interrupt will be triggered; writing 1 to this bit clears the flag; PORIF is a non-maskable interrupt;		

38.3.11 ANA_PAD（0x3C）

Bit	Name	Description	R/W Flag	Reset Value
31:3	---	Reserved	R	0
2	PL03_AE	SAR AIN7 Analog Input Enable Signal 0: Disable 1: Enable	RW	1'b0
1	PL02_AE	SAR AIN6 Analog Input Enable Signal 0: Disable; 1: Enable;	RW	1'b0
0	PL01_AE	SAR AIN5 Analog Input Enable Signal 0: Disable; 1: Enable;	RW	1'b0

PL01~PL03 multiplexing function configuration priority: ANA_PAD > RTC_IOEN > GPIOL_AFSEL0/GPIOL_AFSEL1;

PL00 multiplexing function configuration priority: RTC_IOEN > GPIOL_AFSEL0/GPIOL_AFSEL1;

38.3.12 ANA_RST（0x40）

Bit	Name	Description	R/W Flag	Reset Value
31:16	---	Reserved	R	0
15:8	PS	Write Password Protection Bit for [7:0] =0xa5, [7:0] can be written =Other values, [7:0] cannot be written	RW	0
7:0	ANA_RST	Force reset VBAT power domain all registers with reset terminal, excluding perpetual calendar registers. =0xa8, Reset VBAT power domain all registers with reset terminal =Other, No effect	RW	0

Note: To reset the registers in the VBAT power domain, follow these steps:

Open the password protection bit of the ANA RST register: ANA RST = 0xa500;

Put the VBAT power domain in reset state: ANA RST = 0xa5a8;

Delay 300us;

Release the VBAT power domain reset state: ANA RST = 0xa500;

Close the password protection bit of the ANA RST register: ANA RST = 0x0000

38.4 ADC Voltage Detection Steps

Core0 configures the 4th bit RTC_SARADC_SEC of the SCU_PERIPH_SEC_CFG1 register, which is used to set

the security attribute of the SAR ADC. When RTC_SARADC_SEC=1, the SAR ADC is a secure peripheral; when RTC_SARADC_SEC=0, the SAR ADC is a non-secure peripheral.

The following configurations are common to both Core0 and Core1:

- Set the 4th bit RTC_SARADC_CKE of the MOD1_CKE register in the system control chapter to 1 to enable the bus clock of the analog peripheral;
- Determine whether the 1st bit of the SAR-ADC status register AD_STAT is 0 (no automatic temperature measurement in progress). If it is 0, proceed to the next step; if it is 1, wait;
- Configure the ADC control register AD_CTRL, set the wait time, gain, and select the corresponding ADC channel;
- Configure the SAR-ADC start register AD_START to 1 to start ADC conversion;
- Determine the 0th bit of the SAR-ADC status register AD_STAT and wait for the conversion to complete;
- Read the ADC conversion data register AD_DATA;
- Calculation: The ADC reference voltage source is 1.25V, and the value of the AD_DATA register is 1024 when the ADC is full-scale. The calculation formula is (AD_DATA *1.25)/ 1024. When the voltage exceeds the full-scale of the ADC, the conversion value is 1024;
- Conversion and calculation are completed.

38.5 VBAT Voltage Detection

Core0 configures the 4th bit RTC_SARADC_SEC of the SCU_PERIPH_SEC_CFG1 register to set the security attribute of the SAR ADC. When RTC_SARADC_SEC=1, the SAR ADC is a secure peripheral; when RTC_SARADC_SEC=0, it is a non-secure peripheral.

The following configurations are common to both Core0 and Core1:

- Set the 4th bit RTC_SARADC_CKE of the MOD1_CKE register in the system control chapter to 1 to enable the bus clock of the analog peripheral;
- Determine whether the 1st bit of the SAR-ADC status register AD_STAT is 0 (no automatic temperature measurement in progress). If it is 0, proceed to the next step; if it is 1, wait;
- Configure the ADC control register AD_CTRL to select VBAT as the channel and set the wait time;
- Configure the SAR-ADC start register AD_START to 1 to start ADC conversion;
- Determine the 0th bit of the SAR-ADC status register AD_STAT and wait for the conversion to complete;
- Read the ADC conversion data register AD_DATA;
- Calculation: The ADC reference voltage source is 1.25V, and the value of the AD_DATA register is 1024 when the ADC is full-scale. A 3.6V battery is directly connected to the VBAT pin, and AD_CTRL.SAR_PGA=0 (gain is 0.5x). The voltage value is (AD_DATA1.254)/1024. Since the gain is 0.5x and the internal VBAT input has a 1/2 voltage division, the actual voltage needs to be multiplied by 4;
- Conversion and calculation are completed.

38.6 VBAT BOR Reset Detection

According to application requirements, Core0 configures the security attribute of the SAR ADC and assigns the VBAT interrupt to either Core0 or Core1:

- Write the 4th bit RTC_SARADC_SEC of the SCU_PERIPH_SEC_CFG1 register. When RTC_SARADC_SEC=1, the SAR ADC is a secure peripheral; when RTC_SARADC_SEC=0, it is a non-secure peripheral;
- Execute set_gic_irq(VBAT_IRQn, 1, 0); // The second parameter is set to 1 to assign the interrupt to Core0; the third parameter can also be set to 1 to assign the interrupt to Core1;

The following operation steps are common to both Core0 and Core1:

- Set the 4th bit RTC_SARADC_CKE of the MOD1_CKE register in the system control chapter to 1 to enable the bus clock of the analog peripheral.
- First write ANA_RST_STAT=3 to clear the flag bits. Since VBAT POR reset is a non-maskable interrupt, the PORIF flag needs to be cleared before enabling the VBAT interrupt to avoid the program executing the interrupt function unexpectedly.
- Write ANA_RST_STAT=(1 << 4) to enable the VBAT BOR reset interrupt.
- Enable the VBAT interrupt:

ECLIC_Register_IRQ(VBAT_IRQn, ECLIC_NON_VECTOR_INTERRUPT, ECLIC_LEVEL_TRIGGER, 1, 0, VBAT_HANDLER);

- Write the interrupt service routine:

```
void VBAT_HANDLER(void)
{
    if(MADC->ANA_RST_STAT & (1 << 4)) // VBAT BOR reset
    {
        /* Start adding user code. Do not edit comment generated here */
    }
}
```

39 CMP&LVD

39.1 Features

One LVD (Low Voltage Detection) circuit with the following main features:

- LVD input can be selected as the chip power supply or external PIN input;
- LVD threshold is adjustable, divided into 14 levels from 1.9V to 4.6V;
- When selected as external PIN input, the threshold is fixed at approximately 1.25V with an internal resistance of about 1M ohms;
- Filter time is configurable;
- Module power can be turned off;
- Power consumption is approximately 5μA.

Two comparator circuits (CMP1 and CMP2) with the following main features:

- External PIN input with a threshold of approximately 1.28V;
- CMP1 and CMP2 have an internal 600K pull-down resistor, which can be turned off through the register (CMP_CTL(0x04));
- Filter time is configurable;
- Module power can be turned off;
- Power consumption is approximately 0.3μA, and the comparator can be used for main power on/off monitoring.

39.2 Registers

Base Addresses of CMP_LVD Registers

Module Name	Core0 Base Address	Core1 Base Address
CMP_LVD	0x200B_C000	0xA00B_C000

Offset Addresses of CMP Registers

Register Name	Address Offset	Description
LVD_CTL	0x00	LVD Control Register
CMP_CTL	0x04	CMP Control Register
LVDCMP_IE	0x08	CMP and LVD Interrupt Enable Register
LVDCMP_IF	0x0C	CMP and LVD Interrupt Flag Register
LVD_PWD	0x10	LVD Switch Register
CMP2_PWD	0x14	CMP2 Switch Register
CMP1_PWD	0x18	CMP1 Switch Register

39.2.1 LVD_CTL（0x00）

Bit	Name	Description	R/W Flag	Reset Value
31:12	---	Reserved	R	0x0
11:4	LVD_FL_CNT	LVD Filter Time Configuration: Configure filter time (number of module clock cycles), =0, indicates no filtering; =Other values, indicates filtering is active; Filter time calculation formula: Filter time = LVD_FL_CNT * T_sys_clk, Where: LVD_FL_CNT is the user-configured filter count value (unit: system clock cycles); T_sys_clk is the system clock period. Example default filter times in different operating modes: HCM Mode (High Frequency Clock Mode): Assume system clock frequency is 29.4912 MHz (corresponding period ≈ 33.91 ns), default LVD_FL_CNT = 0x64 (i.e., 100), Filter time = 100 × 33.91 ns ≈ 3.391 μs. LCM Mode (Low Frequency Clock Mode): System clock frequency is 32.768 kHz (corresponding period ≈ 30.52 μs), default LVD_FL_CNT = 0x64 (i.e., 100), Filter time = 100 × 30.52 μs ≈ 3.052 ms. Clock Gating Description: The clock used for LVD filtering is controlled by the LVDCMP module clock gating - they are switched on/off synchronously. That is: as long as the LVDCMP module's clock gating is enabled, the LVD filtering function is active (because the LVD filtering clock shares the same clock gating signal with the LVDCMP module). When the LVDCMP module clock is turned off, the LVD filtering logic will also stop working.	RW	0x64

<table>
<tr>
<td>3:0</td>
<td>LVDS</td>
<td>LVD Threshold Voltage Setting:
<table>
<tr><th>LVDS[3:0]</th><th>Vil(V)</th><th>Vih(V)</th></tr>
<tr><td>0000</td><td>1.9</td><td>2.0</td></tr>
<tr><td>0001</td><td>2.1</td><td>2.2</td></tr>
<tr><td>0010</td><td>2.3</td><td>2.4</td></tr>
<tr><td>0011</td><td>2.5</td><td>2.6</td></tr>
<tr><td>0100</td><td>2.7</td><td>2.8</td></tr>
<tr><td>0101</td><td>2.9</td><td>3.0</td></tr>
<tr><td>0110</td><td>3.1</td><td>3.2</td></tr>
<tr><td>0111</td><td>3.3</td><td>3.4</td></tr>
<tr><td>1000</td><td>3.5</td><td>3.6</td></tr>
<tr><td>1001</td><td>3.7</td><td>3.8</td></tr>
<tr><td>1010</td><td>3.9</td><td>4.0</td></tr>
<tr><td>1011</td><td>4.1</td><td>4.2</td></tr>
<tr><td>1100</td><td>4.3</td><td>4.4</td></tr>
<tr><td>1101</td><td>4.5</td><td>4.6</td></tr>
<tr><td>111x(LVDIN)</td><td>1.25</td><td>1.45</td></tr>
</table>
Note: When LVDS[3:0]=4'b111x, it detects the external LVDIN pin input voltage and compares it with BGR (1.25V). When equal to other values, it compares with the VCC input voltage.</td>
<td>RW</td>
<td>0x0</td>
</tr>
</table>

39.2.2 CMP_CTL（0x04）

Bit	Name	Description	R/W Flag	Reset Value
31:20	---	Reserved	R	0x0
19:12	CMP2_FLT_CNT	CMP2 Filter Time Configuration: Configure filter time (number of module clock cycles), =0, indicates no filtering; =Other values, indicates filtering is active; Filter time calculation formula: Filter time = CMP2_FLT_CNT * T_sys_clk, Where: CMP2_FLT_CNT is the user-configured filter count value (unit: system clock cycles); T_sys_clk is the system clock period. Example default filter times in different operating modes: HCM Mode (High Frequency Clock Mode): Assume system clock frequency is 29.4912 MHz (corresponding period ≈ 33.91 ns), default CMP2_FLT_CNT = 0x64 (i.e., 100), Filter time = 100 × 33.91 ns ≈ 3.391 μs. LCM Mode (Low Frequency Clock Mode): System clock frequency is 32.768 kHz (corresponding period ≈ 30.52 μs), default CMP2_FLT_CNT = 0x64 (i.e., 100), Filter time = 100 × 30.52 μs ≈ 3.052 ms. Clock Gating Description: The clock used for filtering is always on and is not controlled by the LVDCMP module's clock gating.	RW	0x64

11:4	CMP1_FLT_CNT	CMP1 Filter Time Configuration: Configure filter time (number of module clock cycles), =0, indicates no filtering; =Other values, indicates filtering is active; Filter time calculation formula: Filter time = CMP1_FLT_CNT * T_sys_clk, Where: CMP1_FLT_CNT is the user-configured filter count value (unit: system clock cycles); T_sys_clk is the system clock period. Example default filter times in different operating modes: HCM Mode (High Frequency Clock Mode): Assume system clock frequency is 29.4912 MHz (corresponding period ≈ 33.91 ns), default CMP1_FLT_CNT = 0x64 (i.e., 100), Filter time = 100 × 33.91 ns ≈ 3.391 μs. LCM Mode (Low Frequency Clock Mode): System clock frequency is 32.768 kHz (corresponding period ≈ 30.52 μs), default CMP1_FLT_CNT = 0x64 (i.e., 100), Filter time = 100 × 30.52 μs ≈ 3.052 ms. Clock Gating Description: The clock used for filtering is always on and is not controlled by the LVDCMP module's clock gating.	RW	0x64
3	HYSEN_CMP2	CMP2 Internal Hysteresis Comparator Hysteresis Switch 0x1: Enable hysteresis 0x0: Disable hysteresis	RW	0x0
2	HYSEN_CMP1	CMP1 Internal Hysteresis Comparator Hysteresis Switch 0x1: Enable hysteresis 0x0: Disable hysteresis	RW	0x0
1	MODE_CMP2R	CMP2 Internal 600K Resistor Sampling Mode Switch 0: CMP2 internal resistor sampling mode enabled, threshold VIL=1.25V, VIH=1.45V (HYSEN_CMP2 must be configured as 0) 1: CMP2 internal resistor sampling mode disabled, threshold VIL=0.83V, VIH=0.95V (HYSEN_CMP2 must be configured as 1)	RW	0x0

0	MODE_CMP1R	CMP1 Internal 600K Resistor Sampling Mode Switch 0: CMP1 internal resistor sampling mode enabled, threshold VIL=1.25V, VIH=1.45V (HYSEN_CMP1 must be configured as 0) 1: CMP1 internal resistor sampling mode disabled, threshold VIL=0.83V, VIH=0.95V (HYSEN_CMP1 must be configured as 1)	RW	0x0

39.2.3 LVDCMP_IE（0x08）

Bit	Name	Description	R/W Flag	Reset Value
31:3	---	Reserved	R	0x0
2	LVDIE	LVD Interrupt Enable Register: =0: Disable interrupt; =1: Enable interrupt;	RW	0x0
1	CMP2_IE	Comparator 1 Interrupt Enable: =0: Disable interrupt; =1: Enable interrupt;	RW	0x0
0	CMP1_IE	Comparator 0 Interrupt Enable: =0: Disable interrupt; =1: Enable interrupt;	RW	0x0

39.2.4 LVDCMP_IF（0x0C）

it	Name	Description	R/W Flag	Reset Value
31:3	---	Reserved	R	0x0
2	LVDIF	LVD Status Flag (Write 1 to Clear) =0: No interrupt generated; =1: Interrupt generated; Interrupt is generated when the input voltage becomes lower or higher relative to the threshold, write 1 to clear;	RW	0x0
1	CMP2_IF	Comparator 1 Interrupt Flag (Write 1 to Clear) =0: No interrupt generated; =1: Interrupt generated; Interrupt is generated when the input voltage becomes lower or higher relative to the threshold, write 1 to clear;	RW	0x0
0	CMP1_IF	Comparator 0 Interrupt Flag (Write 1 to Clear) =0: No interrupt generated; =1: Interrupt generated; Interrupt is generated when the input voltage becomes lower or higher relative to the threshold, write 1 to clear;	RW	0x0

39.2.5 LVD_PWD（0x10）

Bit	Name	Description	R/W Flag	Reset Value
31:0	LVD_PWD	LVD Switch =0x55AA_AA55: LVD Off =Other: LVD On Default Off	RW	0x55AA_AA55

39.2.6 CMP2_PWD（0x14）

Bit	Name	Description	R/W Flag	Reset Value
31:0	CMP2_PWD	Comparator 2 Switch =0xAA55_55AA: Comparator 1 Off =Other: Comparator 2 On Default On	RW	0x0

39.2.7 CMP1_PWD（0x18）

Bit	Name	Description	R/W Flag	Reset Value
31:0	CMP1_PWD	Comparator 1 Switch =0xA5A5_5A5A: Comparator 0 Off =Other: Comparator 1 On Default On	RW	0x0

39.3 Software Configuration Process and Application Description

The interrupt flag bits of CMP/LVD can only indicate that the current input voltage has changed relative to the threshold, but cannot indicate whether the current input voltage is higher or lower than the threshold. After detecting the interrupt flag bit, the application program needs to further confirm the current level state through the interrupt flag bit after configuring the CMP/LVD switch; the application program can also periodically query the current level state through the interrupt flag bit after configuring the CMP/LVD switch. The following is a typical software processing flow for monitoring the mains power state through a comparator (CMPx):

To accurately distinguish between "power failure" and "power-on" events, after detecting the flag bit through software query or interrupt, to further confirm the current level state, the following steps need to be executed in the interrupt service routine or flag query function:

- Turn off the comparator

  Write 0xAA55_55AA to the CMPx_PWD register to turn off CMP2; or write 0xA5A5_5A5A to turn off CMP1.
- Delay to wait for filtering completion (shutdown phase)

  Software delay time: Tdelay1 =(CMP2_FLT_CNT or CMP1_FLT_CNT +3)×Tsys_clk
- Clear the interrupt flag (after shutdown)

Read and clear the CMPx_IF flag bit. This step is to avoid confusion in judging the flag bit after restarting the comparator:

If CMPx_IF == 1, it indicates that there was a valid interrupt event before shutdown; clear the flag bit (clear by writing 1);

If it is 0, there is no event; proceed with the subsequent process.

- Restart the comparator

Write 0x0 or another value to the CMPx_PWD register to restart CMP2; alternatively, write 0x0 or another value to restart CMP1.

- Delay to wait for filtering completion (startup phase)

Software delay time: Tdelay2 =(CMP2_FLT_CNT or CMP1_FLT_CNT +3)×Tsys_clk +1.3μs

The additional 1.3 μs is used to ensure the stability of the comparator flag. When the comparator is turned on from off, the flag bit will rise later, requiring a delay of 1.3μs. If the software waits for the flag instead of an interrupt, the waiting time needs to be extended by 1.3μs.

- Finally judge the mains power state

Read the CMPx_IF flag bit again:

If CMPx_IF == 1: It indicates that a mains power failure is currently detected; clear the flag bit (clear by writing 1);

If CMPx_IF == 0: It indicates that the mains power is currently in the power-on state.

Notes:

- All delays need to be converted into specific instruction cycles according to the actual system clock frequency (e.g., 29.4912 MHz);
- The filter cycle count refers to the filter count value configured in the CMPx_CTL register (e.g., CMP1_FLT_CNT);
- If the application program does not perform the above level query operation in the interrupt service routine (i.e., through software query), it is recommended to disable the interrupt enable configuration register corresponding to the CMP/LVD module during the above operations to prevent frequent interrupt triggers caused by flag bit changes during the comparator switch operation.

It is recommended that the application program execute the above operations in multiple cycles to obtain a reliable level indication;

40 ADC0

40.1 Introduction

The chip integrates ADC0 (High-Speed Analog-to-Digital Converter) with a maximum resolution of 12 bits. It features 14 multiplexed channels that can measure signals from 12 external sources, 1 internal reference channel, and 1 DAC channel. External sources can be directly input to ADC0 or through a BUFFER before input to ADC0. Analog-to-digital conversion of these channels can be performed in single-shot, continuous, full-scan, or interval-scan modes. The conversion result is stored in a 16-bit data register with optional left-aligned or right-aligned configuration.

ADC0 has an analog watchdog feature that allows the application to detect if the input voltage exceeds a user-defined threshold range.

The oversampling mechanism of ADC0 can improve performance by reducing the relevant computational burden on the MCU.

ADC0 supports a built-in reference source calibration scheme.

40.2 Features

- Default 12-bit resolution; software-configurable to 10-bit, 8-bit, or 6-bit resolution
- Up to 0.9216 MSPS sampling rate at 12-bit resolution; lower resolution enables higher sampling rates
- 14 input channels, including 12 external channels (ADC0AIN0~11), 1 BGRLP channel, and 1 DAC channel;
- Reference voltage derived from VREFP/VREFN, default connected to VCC/VSS;
- Channel input range: VREFN ＜ VIN ＜ VREFP
- Power-on stabilization time, conversion interval time, and sampling time for each channel can be independently configured
- Supports single-shot conversion, continuous conversion, and software-customized sequence scan conversion or interval conversion modes
- Supports oversampling operation (multiple conversion averaging function) to achieve 16-bit resolution
- Supports left-aligned or right-aligned data registers
- Supports KBI or SIMPTC to automatically trigger ADC0 conversion, improving conversion real-time performance;
- Conversion result threshold monitoring function: analog watchdog function
- Built-in BUFFER for converting high-impedance signals
- Supports built-in reference source calibration scheme to eliminate ADC0 OFFSET
- Supports DMA function
- Generates interrupts upon conversion completion, analog watchdog events, or overflow events
- Supports a minimum operating voltage of 1.8V (when the operating voltage is 1.8~2.4V, the maximum operating clock frequency is 8MHz; when the operating voltage is 1.8V, the maximum input signal frequency is less than 1KHz; 2.2V when the BUFFER is enabled)

40.3 Block Diagram

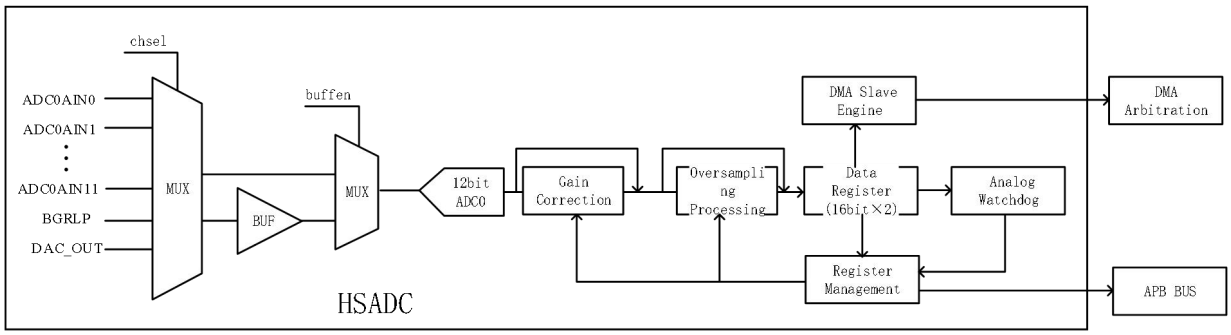


Figure 40-1 Block Diagram

40.4 Pin Definition

Pin Name	Pin Number	Description
ADC0AIN0	PH[0]	External Input Channel 0
ADC0AIN1	PH[1]	External Input Channel 1
ADC0AIN2	PA[5]	External Input Channel 2
ADC0AIN3	PA[6]	External Input Channel 3
ADC0AIN4	PA[7]	External Input Channel 4
ADC0AIN5	PA[8]	External Input Channel 5
ADC0AIN6	PA[9]	External Input Channel 6
ADC0AIN7	PA[10]	External Input Channel 7
ADC0AIN8	PA[13]	External Input Channel 8
ADC0AIN9	PA[14]	External Input Channel 9
ADC0AIN10	PA[15]	External Input Channel 10
ADC0AIN11	PB[4]	External Input Channel 11
VREFP	VREFP	Reference Voltage Positive Input
VREFN	VREFN	Reference Voltage Negative Input

Note: When an external signal is directly connected to ADC0 without passing through the buffer, to ensure measurement accuracy, requirements are imposed on the external impedance. To make the sampling error less than 0.5 LSB, the maximum value of the internal resistance of the external signal source is expressed as follows:

$$R_{AIN\max} = \frac{T_S}{f_{ADC} \times C_{ADC} \times \ln\left(2^{N+1}\right)} - R_{ADC}$$

Where N is the resolution of ADC0, Ts is the number of sampling cycles, fADC is the main clock frequency, Cadc is the sampling capacitor, and Radc is the signal path impedance, including the internal resistance of ADC0 and the pad channel impedance. Note: Simulation shows that the value of Cadc does not change when ADC0 is configured with different resolutions.

40.5 Clock and Reset

The clock of ADC0 follows the bus clock.

The reset signal of ADC0 takes effect upon core reset and PIN reset, and ADC0 also supports independent software reset.

40.6 Functional Description

40.6.1 Switch Control

ADC0 can be selected as always-on or interval-on through the power-down switch control register ADC0_PWDCTL.ADC0_AON.

The BUFFER can be selected as always-on or interval-on through the power-down switch control register ADC0_PWDCTL.BUFF_AON.

If ADC0 is selected as always-on, if conversion is started immediately after enabling, it needs to wait for 32~35 cycles. During this period, ADC0 will not start any conversion. After the wait, conversion can begin. No additional wait time is required for subsequent conversions, but keeping ADC0 always-on will increase chip power consumption.

If ADC0 is selected as interval-on, ADC0 will power down when idle. Whenever a conversion trigger occurs during idle time, ADC0 needs a stabilization time of 32~35 cycles before starting conversion, and will enter power-down state again after conversion is completed.

If the BUFFER is selected as always-on, it needs a stabilization time of at least 7.5us after power-on. During this period, ADC0 cannot start any conversion.

40.6.2 Sampling Rate

The sampling rate of ADC0 is calculated as follows:

$$F_{ADCsamp} = \frac{f_{\text{clk}}}{N_{\text{div}} \times \left(T_{samp} + T_{\text{conv}} + T_{\text{wconv}}\right)}$$

Where,$f_{\text{clk}}$is the CPU system bus clock,N_{div} is the division factor, which can be configured by software through the register ADC0_CFG.CLKDIV (configurable as 2~128); T_{samp} is the sampling time of ADC0, which can be configured by software through the channel sampling time configuration register ADC0_SAMPTn (configurable as 4~256 cycles); T_{conv} is the conversion time of ADC0, which is proportional to the resolution (specifically 6/8/10/12 cycles); T_{wconv} is the inter-conversion wait time of ADC0, which can be configured as 0~31 cycles by software through the register ADC0_CFG.INTV_TIME.The normal operating sampling rate of ADC0 is 30 KSPS~1 MSPS. Attention should be paid during application not to exceed this range; otherwise, ADC0 will not work properly. This is because there are a series of internal capacitors and resistors. If the sampling clock is too fast, charging will be incomplete; if the sampling clock is too slow, the charge on the capacitor will decrease, and the sampling capacitor cannot maintain the charge.

Example: When f_{clk}=58.98MHz，N_{div}=4，T_{samp}=4，T_{conv}=12，T_{wconv}=0, the sampling rate of 12-bit ADC0 is 0.9216 MSPS.

40.6.3 Channel Selection

The number of channels is selected through ADC0_CHSELx.CHSEL.

Whether the input channel passes through the BUFFER before input is selected through ADC0_CFG.BUFF_EN.

Channel Number	Channel Input Signal	
ADC0_CHSELx	BUFF_EN=0	BUFF_EN =1
0	ADC0AIN0 Direct Input	ADC0AIN0 Input through BUFFER
1	ADC0AIN1 Direct Input	ADC0AIN1 Input through BUFFER
2	ADC0AIN2 Direct Input	ADC0AIN2 Input through BUFFER
3	ADC0AIN3 Direct Input	ADC0AIN3 Input through BUFFER
4	ADC0AIN4 Direct Input	ADC0AIN4 Input through BUFFER
5	ADC0AIN5 Direct Input	ADC0AIN5 Input through BUFFER
6	ADC0AIN6 Direct Input	ADC0AIN6 Input through BUFFER
7	ADC0AIN7 Direct Input	ADC0AIN7 Input through BUFFER
8	ADC0AIN8 Direct Input	ADC0AIN8 Input through BUFFER
9	ADC0AIN9 Direct Input	ADC0AIN9 Input through BUFFER
10	ADC0AIN10 Direct Input	ADC0AIN10 Input through BUFFER
11	ADC0AIN11 Direct Input	ADC0AIN11 Input through BUFFER
12	BGRLP Direct Input	BGRLP Input through BUFFER
13	DAC Voltage Input	DAC Voltage Input through BUFFER

Software can select any one or more channels to form Sequence A for continuous conversion, and any one or more channels to form Sequence B for continuous conversion. If conversion requirements occur simultaneously during the conversion of Sequence A and Sequence B, the following principles are followed:：

A/D Conversion	Trigger Signal Occurrence	Processing Method (Priority: Sequence A ＞ Sequence B)
During Sequence A conversion	Sequence A trigger	Trigger signal is invalid
	Sequence B trigger	Start Sequence B conversion after all channels of Sequence A are converted
During Sequence B conversion	Sequence A trigger	Sequence B conversion is interrupted, and Sequence A conversion starts;After Sequence A conversion is completed, Sequence B restarts conversion from the interrupted channel;
	Sequence B trigger	Trigger signal is invalid
ADC0 idle	Sequence A and B triggers simultaneously	Sequence A starts first; start Sequence B conversion after all channels of Sequence A are converted;

40.6.4 Trigger Source Selection

Sequence A and Sequence B can independently select trigger sources.

Sequence A and Sequence B can be triggered by software, or the output of KBI or SIMPTC in the AOS module can be selected as the trigger source. That is, write the PE_TRGSEL32/PE_TRGSEL33 of the AOS module to enable KBI or SIMPTC triggering, wait for the hardware trigger condition, and start conversion.

40.6.5 Conversion Modes

- Single-Channel Single-Shot Conversion Mode

This mode can operate in Sequence A or Sequence B. Software selects a single channel to be converted. When triggered by software or the selected corresponding trigger source, ADC0 will sample and convert the channel. After completion, the flag bit EOC is pulled high, indicating the end of sampling and conversion. The data is stored in the corresponding data register of Sequence A or Sequence B. A new round of sampling and conversion will be performed only when the trigger source is triggered again.

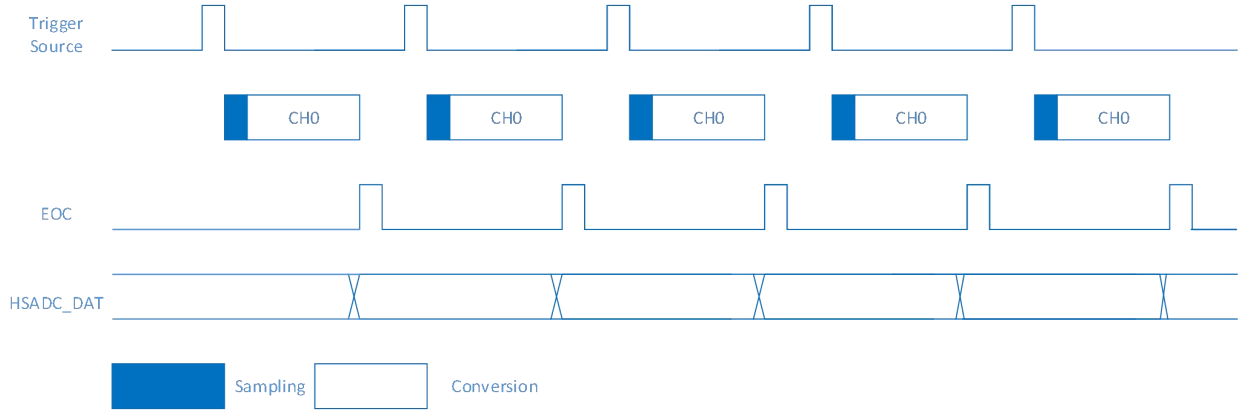


Figure 40-2 Single-Channel Single-Shot Conversion Mode

- Single-Channel Continuous Conversion Mode

On the basis of the single-shot conversion mode, enabling bit[1] CONTA in the register ADC0_CTLx enters the continuous conversion mode.

This mode can operate in Sequence A or Sequence B. Software selects a single channel to be converted. When triggered by software or the selected corresponding trigger source, ADC0 will continuously sample and convert the specified channel, and the data is stored in the corresponding data register of Sequence A or Sequence B.

Software can start DMA to transfer data to SRAM.

ADC0 will stop sampling and conversion only when software turns off ADC0 or disables the continuous conversion function.

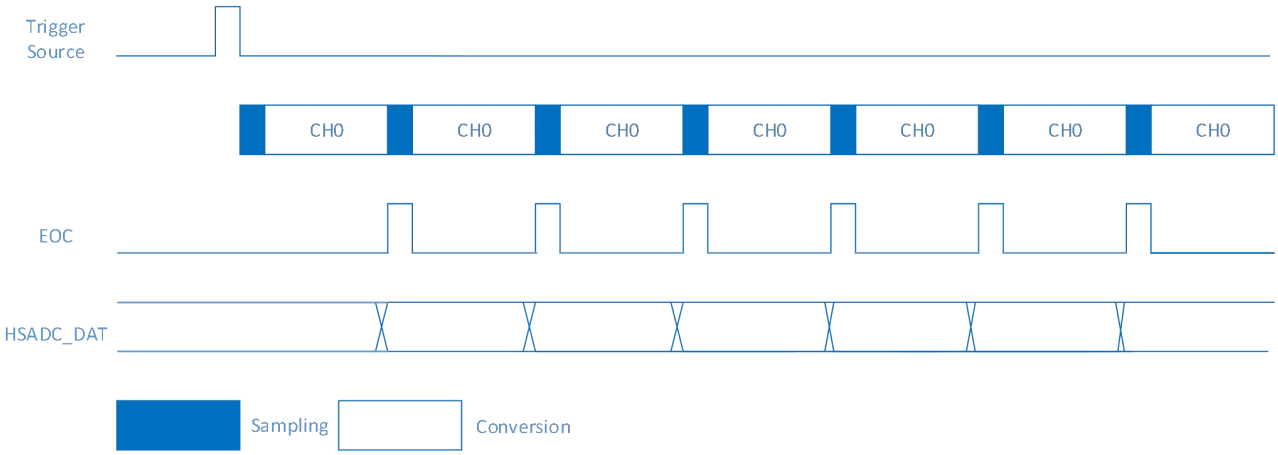


Figure 40-3 Single-Channel Continuous Conversion Mode

- Multi-Channel Single-Scan Conversion Mode

Software selects the channels of Sequence A or Sequence B. When triggered by software or the selected corresponding trigger source, ADC0 will sample and convert all enabled channels in Sequence A or Sequence B. After completion, the flag bit EOCAF is pulled high, indicating the end of sampling and conversion. The data is stored in the corresponding data register of Sequence A or Sequence B. A new round of sampling and conversion will be performed

only when the trigger source is triggered again.

Software can start DMA to transfer data to SRAM.

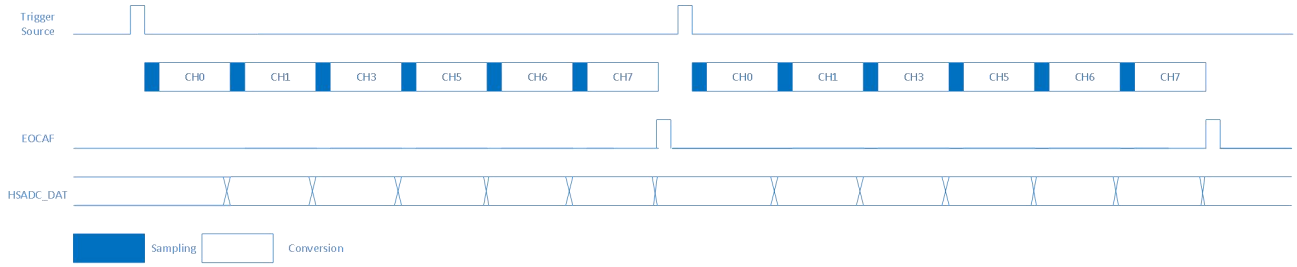


Figure 40-4 Multi-Channel Single-Scan Conversion Mode

- Multi-Channel Continuous-Scan Conversion Mode

On the basis of the multi-channel single-shot conversion mode, enabling bit[1] CONTA in the register ADC0_CTLx enters the multi-channel continuous conversion mode.

In this mode, after all channels in Sequence A or Sequence B are converted, the conversion will automatically restart without waiting for a new trigger source.

Software can start DMA to transfer data to SRAM.

ADC0 will stop sampling and conversion only when software turns off ADC0 or disables the conversion function.

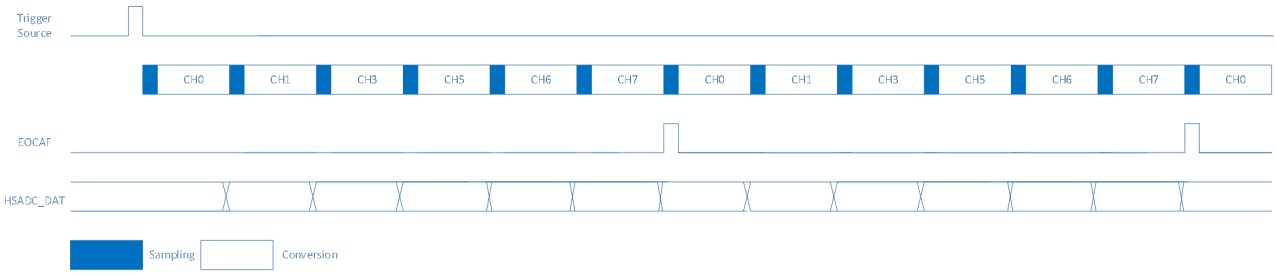


Figure 40-5 Multi-Channel Continuous-Scan Conversion Mode

- Multi-Channel Interval Conversion Mode

Software first selects the channels of Sequence A or Sequence B, then selects the number of channels n to be converted each time during interval conversion. When triggered by software or the selected corresponding trigger source, ADC0 will sample and convert n channels of Sequence A or Sequence B. Upon re-triggering, it will continue to sample and convert n channels of the remaining channels, and so on until all enabled channels are sampled and converted.

Software can start DMA to transfer data to SRAM.

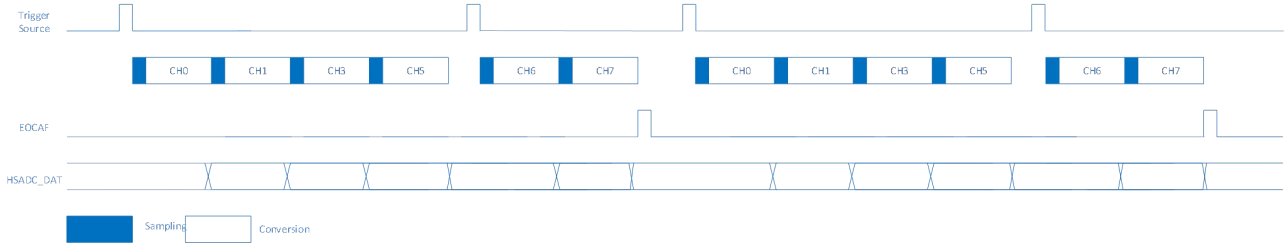


Figure 40-6 Multi-Channel Interval Conversion Mode

The above figure shows the interval conversion mode with n=4.

40.6.6 Analog Watchdog

Software can independently enable the watchdog function for Sequence A and Sequence B. If the ADC0 conversion

result exceeds the threshold range, an analog watchdog event is generated. If the interrupt is enabled, an interrupt will be generated.

Software can select the protected range as 0~lower threshold, lower threshold~upper threshold, or upper threshold~full scale.

Software can select to monitor one or any multiple channels of Sequence A and/or Sequence B, not all enabled channels.

40.6.7 Programmable Sampling Time

Software can select the clock cycles for each channel during sampling. Each external channel or internal channel can have independent register bits to configure the sampling time as 4~256 cycles. Software can set the number of sampling cycles for each channel from the register ADC0_SAMPTn.

40.6.8 Data Management

- Data Resolution

The conversion cycle of ADC0 is related to the number of resolution bits; the lower the resolution, the faster the conversion speed.

Software can select the resolution of ADC0 during the conversion of Sequence A or Sequence B as 12-bit, 10-bit, 8-bit, or 6-bit from bits[1:0] RESC of the register ADC0_CFG.

- Data Compensation

The register compensates the conversion result. The ADC0 conversion result needs to be added to or subtracted from the value in the register before being stored in the corresponding register.

This function takes effect only when ADC0_CAL.CAL_EN is configured to 1;

- Data Auto-Clear Function

The conversion results of Sequence A or Sequence B are stored in one register. To facilitate software to determine whether the current conversion is completed, software can enable the data auto-clear function. After the data is read by software or DMA, the data register will be automatically cleared. If the data read again is 0, it indicates that the current conversion is not completed, and the read data is invalid;

- Oversampling Processing

Software can configure the number of oversampling accumulations N and the averaging factor M through bits[3:1] OVSR and bits[7:4] OVSS in the ADC0_OVSCFG register to obtain 16-bit data;

If the number of valid bits of the calculation result exceeds 16, the lower bits will be truncated. The specific truncation mechanism is as follows:

Resolution (RESC)	Number of Accumulations (OVSR)	Effective Bit Width of Accumulated Sum	Number of Right Shifts (OVSS)	Hardware Automatic Right Shifts	Final Effective Output Bit Width
12 bits	4	17 bits	0	1	16 bits
12 bits	5	18 bits	0 ~ 1	2 ~ 1	16 bits
12 bits	6	19 bits	0 ~ 2	3 ~ 1	16 bits
12 bits	7	20 bits	0 ~ 3	4 ~ 1	16 bits
10 bits	6	17 bits	0	1	16 bits
10 bits	7	18 bits	0~1	2 ~ 1	16 bits

- Alignment Function

The data alignment function is only valid when oversampling processing is not enabled. Software can select the

conversion result to be left-aligned or right-aligned and stored in ADC0_DATx or in the memory through DMA by DFMT;

Data alignment for 12-bit resolution:

Left-Aligned Data:

D11	D10	D9	D8	D7	D6	D5	D4	D3	D2	D1	D0	0	0	0	0

Right-Aligned Data:

0	0	0	0	D11	D10	D9	D8	D7	D6	D5	D4	D3	D2	D1	D0

Data alignment for 10-bit resolution:

Left-Aligned Data:

D9	D8	D7	D6	D5	D4	D3	D2	D1	D0	0	0	0	0	0	0

Right-Aligned Data:

0	0	0	0	0	0	D9	D8	D7	D6	D5	D4	D3	D2	D1	D0

Data alignment for 8-bit resolution:

Left-Aligned Data:

D7	D6	D5	D4	D3	D2	D1	D0	0	0	0	0	0	0	0	0

Right-Aligned Data:

0	0	0	0	0	0	0	0	D7	D6	D5	D4	D3	D2	D1	D0

Data alignment for 6-bit resolution:

Left-Aligned Data:

D5	D4	D3	D2	D1	D0	0	0	0	0	0	0	0	0	0	0

Right-Aligned Data:

0	0	0	0	0	0	0	0	0	0	D5	D4	D3	D2	D1	D0

40.6.9 Calibration Function

Since ADC0 has VREF error, gain error, and offset error, a calibration function for ADC0 is provided;

- Code Value Calibration

Assume that the output code value of a certain channel when the BUFFER is not enabled is DIN, and the output code value when the BUFFER is enabled is DINB. The corresponding code values output after hardware calibration are DOUT and DOUTB, using the following calibration methods:

BUFFER not enabled: DOUT = (DIN + ZTG * ZT) * GAINERR + DOS

BUFFER enabled: DOUTB = (DINB + ZTBG * ZTB) * GAINERR + DOS

Where, GAINERR, ZT, and ZTB are values determined during FT testing, managed by BOOT, and automatically loaded after power-on;

ZTG, ZTBG, and DOS are calibration coefficients written by the user; the calculation formulas for ZTG and ZTBG are as follows: where DR0 and DRB0 are the accumulated sum of 16 conversion results of the BGRLP channel when the BUFFER is enabled and not enabled under an ideal power supply (the default input reference voltage is 3.3V) (values determined during FT testing); DR1 and DRB1 are the results measured for the BGRLP channel when the BUFFER is enabled and not enabled under the current environment (n is the number of oversampling accumulations, m is the number of oversampling right shifts);

$$ZTG = \frac{DR1 \times 2^{m+4}}{DR0 \times n} \qquad ZTBG = \frac{DRB1 \times 2^{m+4}}{DRB0 \times n}$$

- Calculating the Current Input Voltage Value

For the case where the buffer is not enabled, the output code value of the BGRLP channel under the reference voltage VREF0 (default 3.3V) is DR0, and the output code value of the BGRLP channel under the current reference voltage VREF1 is DR1. Since the reference voltage used as internal input is constant, we have:

VREF0 * DR0 = VREF1 * DR1

Then VREF1 = VREF0 * DR0 / DR1

From this, the current reference voltage VREF1 can be obtained, and then the voltage value of the current input channel can be calculated as:

$$Vin = \frac{Dout \times VREF1}{2^{12}}$$

If the input channel does not enable the BUFFER, then $Vin = \frac{Dout \times VREF0}{2^{12} \times ZTG}$

If the input channel enables the BUFFER, then $Vin = \frac{Dout \times VREF0}{2^{12} \times ZTBG}$

Where, VREF0 is the externally injected reference voltage during FT testing, usually connected to VCC with a default value of 3.3V. This value should be determined according to the actual input reference voltage of the calibration scheme during FT testing;

40.6.10 Linkage with DAC

A transient surge current may occur on the analog power supply when the DAC starts conversion, which may interfere with the ADC0 that is working. Therefore, the ADC0 conversion priority function can be enabled in the DAC. ADC0 will output an idle state to the DAC. By changing the update timing of the DAC conversion data, this interference can be effectively avoided.

It also supports enabling the DAC conversion priority function in ADC0. When the ADC0_CFG.HALT_2DAC signal is configured to 1, when ADC0 starts from the idle state, it will first wait for the DAC to be in the idle state before starting conversion;

40.6.11 Dummy Conversion Function

Since the first conversion result of the ADC0 IP after power-on has an error of 4 code values, a dummy conversion function is added; by default, after ADC0 is powered on, a dummy conversion is performed on the selected channel, and the conversion result is discarded; software can configure the number of dummy conversions as 0~3 through ADC0_DCONV;

40.6.12 Interrupts

When ADC0 conversion is completed, if the interrupt is enabled, an interrupt will be generated;

When a watchdog event occurs, if the interrupt is enabled, an interrupt will be generated;

When DMA is enabled and DMA interrupt is enabled, if a half-full, full-full, or data overwrite event occurs, an interrupt will be generated;

The above interrupts share one interrupt number.

40.7 Registers

40.7.1 Register Mapping Table

Module Name	Physical Address	Mapped Address

ADC0 (Small Core Area)	0x2009_4000	0x2009_4000
ADC0 (Large Core Area)	0xA009_4000	0xA009_4000

Register Name	Address Offset	Description
ADC0_CFG	0x00	Configuration Register
ADC0_OVSCFG	0x04	Oversampling Configuration Register
ADC0_OVSCHSEL	0x08	Oversampling Channel Select Register
ADC0_PWDCTL	0x0C	Power Down Switch Control Register
ADC0_SAMPT0	0x10	Input Channel 0~3 Sampling Time Configuration Register
ADC0_SAMPT1	0x14	Input Channel 4~7 Sampling Time Configuration Register
ADC0_SAMPT2	0x18	Input Channel 8~11 Sampling Time Configuration Register
ADC0_SAMPT3	0x1C	Input Channel 12~14 Sampling Time Configuration Register
ADC0_CTRLA	0x20	Sequence A Control Register
ADC0_CHSELA	0x24	Sequence A Channel Select Register
ADC0_STARTA	0x28	Sequence A Start Register
ADC0_DATA	0x2C	Sequence A Data Register
ADC0_WDTHLA	0x30	Sequence A Watchdog Threshold Register
ADC0_WCHSELA	0x34	Sequence A Watchdog Channel Select Register
ADC0_DMA_RBADRA	0x38	Sequence A DMA Receive Base Address Register
ADC0_DMA_RLENA	0x3C	Sequence A DMA Receive Length Register
ADC0_DMA_RADRA	0x40	Sequence A DMA Receive Current Address Register
ADC0_IEA	0x44	Sequence A Interrupt Enable Register
ADC0_IFA	0x48	Sequence A Interrupt Flag Register
--	0x4C	Reserved
ADC0_CTRLB	0x50	Sequence B Control Register
ADC0_CHSELB	0x54	Sequence B Channel Select Register
ADC0_STARTB	0x58	Sequence B Start Register
ADC0_DATB	0x5C	Sequence B Data Register
ADC0_WDTHLB	0x60	Sequence B Watchdog Threshold Register
ADC0_WCHSELB	0x64	Sequence B Watchdog Channel Select Register
ADC0_DMA_RBADRB	0x68	Sequence B DMA Receive Base Address Register
ADC0_DMA_RLENB	0x6C	Sequence B DMA Receive Length Register
ADC0_DMA_RADRB	0x70	Sequence B DMA Receive Current Address Register
ADC0_IEB	0x74	Sequence B Interrupt Enable Register
ADC0_IFB	0x78	Sequence B Interrupt Flag Register
--	0x7C	Reserved
ADC0_CAL	0x80	Gain Calibration Register

ADC0_DOS	0x84	Code Value Calibration Register
ADC0_BGRLP	0x88	BGRLP Channel Ideal Code Value Register
ADC0_DCONV	0x94	Dummy Conversion Configuration Register

40.7.2 Register Description

40.7.2.1 Configuration Register ADC0_CFG(0x00)

<table>
<tr><th>Bit</th><th>Name</th><th>Description</th><th>R/W</th><th>Reset Value</th></tr>
<tr><td>31:30</td><td>--</td><td>Reserved Bits</td><td>RW</td><td>0x0</td></tr>
<tr><td>29:23</td><td>CLKDIV</td><td>ADC0 module operating clock is the bus clock divided by (CLKDIV+1)
0: Bus clock divided by 2
1: Bus clock divided by 2
.....
127: Bus clock divided by 128</td><td>RW</td><td>0x0</td></tr>
<tr><td>22:18</td><td>INTV_TIME</td><td>Wait time between conversions is INTV_TIME cycles
0: Start next conversion immediately after current conversion ends;
1: Wait 1 cycle after current conversion ends before starting next conversion;
2: Wait 2 cycles after current conversion ends before starting next conversion;
3: Wait 3 cycles after current conversion ends before starting next conversion;
.....
31: Wait 31 cycles after current conversion ends before starting next conversion;</td><td>RW</td><td>0x0</td></tr>
<tr><td>17:16</td><td>PWRUP1_TIME</td><td>ADC0 analog circuit power-up stabilization time is (PWRUP1_TIME+32) cycles
0: Power-up stabilization time is 32 cycles
1: Power-up stabilization time is 33 cycles
2: Power-up stabilization time is 34 cycles
3: Power-up stabilization time is 35 cycles</td><td>RW</td><td>0x1</td></tr>
<tr><td>15:8</td><td>PWRUP2_TIME</td><td>BUFFER power-up stabilization time is (PWRUP2_TIME+32) cycles
<table>
<tr><th>ADC0 Operating Clock</th><th>Recommended Config</th><th>ADC0 Operating Clock</th><th>Recommended Config</th></tr>
<tr><td>14.7456MHz</td><td>0x4F</td><td>5.89824MHz</td><td>0xD</td></tr>
<tr><td>11.7965MHz</td><td>0x39</td><td>5.362036MHz</td><td>0x9</td></tr>
<tr><td>9.8304MHz</td><td>0x2A</td><td>4.9152MHz</td><td>0x5</td></tr>
<tr><td>8.4261MHz</td><td>0x20</td><td>4.537108MHz</td><td>0x3</td></tr>
<tr><td>7.3728MHz</td><td>0x18</td><td>≤4.213029MH</td><td>0x1</td></tr>
</table></td><td>RW</td><td>0x4F</td></tr>
</table>

		<table><tr><td></td><td></td><td>z</td><td></td></tr><tr><td>6.5536MHz</td><td>0x12</td><td></td><td></td></tr></table> Note 1: BUFFER power-up stabilization time is recommended to be greater than 7.5us; Note 2: When BUFFER and ADC0 are not configured to be always on, enabling BUFFER and starting ADC0 will power up both BUFFER and ADC0 simultaneously, so PWRUP2_TIME cannot be less than PWRUP1_TIME.		
7:6	--	Reserved Bits	RO	0x0
5	HALT_2DAC	DAC Link Configuration Bit 0: ADC0 works normally regardless of DAC state; 1: ADC0 pauses conversion when DAC is in conversion state;	RW	0x0
4	BUFF_EN	BUFFER Switch Control 0: Turn off BUFFER, channel input signal is directly connected to ADC0; 1: Turn on BUFFER, channel input signal is connected to ADC0 after being amplified by BUFFER, used for measuring high-impedance signals; This bit can only be configured when ADC0 is idle, i.e., when both ADC0_STARTA and ADC0_STARTB are 0;	RW	0x0
3	DCLREN	Data Register Auto Clear Enable 0: Data register is not automatically cleared 1: Data register is automatically cleared after software read or DMA transfer	RW	0x0
2	DFMT	Data Format 0: Data right-aligned 1: Data left-aligned	RW	0x0
1:0	RESC	Resolution 00: 12-bit 01: 10-bit 10: 8-bit 11: 6-bit This bit can only be configured when ADC0 is idle, i.e., when both ADC0_STARTA and ADC0_STARTB are 0;	RW	0x0

40.7.2.2 Oversampling Configuration Register ADC0_OVSCFG(0x04)

Bit	Name	Description	R/W	Reset Value
31:8	--	Reserved Bits	RO	0x0
7:4	OVSS	Oversampling Right Shift Count 0000: No shift	RW	0x0

		0001: Shift 1 bit 0010: Shift 2 bits 0011: Shift 3 bits 0100: Shift 4 bits 0101: Shift 5 bits 0110: Shift 6 bits 0111: Shift 7 bits 1000: Shift 8 bits Others: No shift		
3:1	OVSR	Oversampling Accumulation Count 000: Accumulate 2 times 001: Accumulate 4 times 010: Accumulate 8 times 011: Accumulate 16 times 100: Accumulate 32 times 101: Accumulate 64 times 110: Accumulate 128 times 111: Accumulate 256 times	RW	0x0
0	OVSEN	Oversampling Enable 0: Disable 1: Enable This bit can only be configured when ADC0 is idle, i.e., when both ADC0_STARTA and ADC0_STARTB are 0;	RW	0x0

40.7.2.3 Oversampling Channel Select Register ADC0_OVSCHSEL(0x08)

Bit	Name	Description	R/W	Reset Value
31:15	--	Reserved Bits	RO	0x0
14:0	OVSC CHSEL	Oversampling Channel Select, each bit represents a channel, any combination can be selected. 0: The corresponding channel is not selected, then only one conversion is performed per sequence conversion 1: The corresponding channel is selected, then the accumulated sum and right shift of multiple conversion results are performed per sequence conversion, and then the data is placed into the data register; This bit can only be configured when ADC0 is idle, i.e., when both ADC0_STARTA and ADC0_STARTB are 0;	RW	0x0

40.7.2.4 Power Down Switch Control Register ADC0_PWDCTL(0x0C)

Bit	Name	Description	R/W	Reset Value
31:17	--	Reserved Bits	RO	0x0

15:8	ADC0_AON	ADC0 Analog Circuit Switch Configuration Bit =0xA5, ADC0 analog circuit is always on (must wait at least 33 cycles before starting ADC0 operation) =Other values, ADC0 analog circuit is only turned on during operation	RW	0x0
7:0	BUFF_AON	ADC0 BUFFER Switch Configuration Bit =0xAC, BUFFER is always on (must wait at least 7.5us before starting ADC0 conversion) =Other values, BUFFER is only turned on during operation;	RW	0x0

40.7.2.5 Channel Sampling Time Configuration Register ADC0_SAMPTn(0x10~0x1C)

n=0~3

Bit	Name	Description	R/W	Reset Value
31:24	CHN3_SMPT	Sampling cycle count for channel number 4n+3 is (CHN3_SMPT+1), optional 4~256 cycles, configuring 1~3 cycles is too short and will cause conversion error; In ADC0_SAMPT3, this bit is reserved;	RW	0x3
23:16	CHN2_SMPT	Sampling cycle count for channel number 4n+2 is (CHN2_SMPT+1), optional 4~256 cycles, configuring 1~3 cycles is too short and will cause conversion error;	RW	0x3
15:8	CHN1_SMPT	Sampling cycle count for channel number 4n+1 is (CHN1_SMPT+1), optional 4~256 cycles, configuring 1~3 cycles is too short and will cause conversion error;	RW	0x3
7:0	CHN0_SMPT	Sampling cycle count for channel number 4n is (CHN0_SMPT+1), optional 4~256 cycles, configuring 1~3 cycles is too short and will cause conversion error;	RW	0x3

40.7.2.6 Sequence x Control Register ADC0_CTLx(0x20/0x50)

x = A/B.

Bit	Name	Description	R/W	Reset Value
31:15	--	Reserved	RO	0x0
14	SMODE	Sequence B Preemption Handling Method 0: Forcefully abort conversion and switch to sequence A 1: Wait for current conversion to complete before switching to sequence A SMODE is only valid in ADC0_CTLB register, invalid in ADC0_CTLA;	RW	0x0
13:12	WDTMOD	Watchdog Protection Area 00: Threshold lower limit < data < threshold upper limit 01: data < threshold lower limit	RW	0x0

		10: data > threshold upper limit 11: Reserved		
11	WDTEN	Watchdog Enable 0: Disable 1: Enable	RW	0x0
10	OVRMOD	Data Overflow Handling Method 0: Do not overwrite data 1: Overwrite data	RW	0x0
9	DMA_MODE	DMA Mode Configuration Bit 0: Single mode, DMA stops after current specified length data transfer is completed; 1: Circular mode, after current specified length data transfer is completed, automatically reload DMA start address and data length;	RW	0x0
8	DMAEN	Start DMA transfer data, only supports 16-bit transfer;	RW	0x0
7	--	Reserved	RO	0x0
6:3	DSICNUM	ADC0 Discontinuous Scan Count is (DSICNUM+1)	RW	0x0
2	DSICEN	ADC0 Discontinuous Scan Enable 0: ADC0 does not perform discontinuous scan on channels enabled by ADC0_CHSELx 1: ADC0 performs discontinuous scan on channels enabled by ADC0_CHSELx This bit can only be configured when sequence x is idle, i.e., when ADC0_STARTx is 0; Note: When DSICEN=1, if CHMOD=0 or CONT=1, the conversion mode is invalid, writing START or hardware trigger at this time is invalid, ADC will not start conversion;	RW	0x0
1	CONTA	ADC0 Continuous Conversion Enable 0: When software/hardware triggers, ADC0 converts the specified one or more channels once 1: When software/hardware triggers, ADC0 continuously converts the specified one or more channels until software writes 0 to STOP to stop conversion This bit can only be configured when sequence x is idle, i.e., when ADC0_STARTx is 0;	RW	0x0
0	CHMOD	ADC0 Sequence x Channel Enable 0: ADC0 converts the channel specified by ADC0_DATx.CHN 1: ADC0 performs scan conversion on channels enabled by ADC0_CHSELx This bit can only be configured when sequence x is idle, i.e., when ADC0_STARTx is 0;	RW	0x0

40.7.2.7 Sequence x Channel Select Register ADC0_CHSELx(0x24/0x54)

x = A/B.

Bit	Name	Description	R/W	Reset Value
31:15	--	Reserved Bits	RO	0x0
14:0	CHSEL	Sequence x Channel Select, each bit represents a channel, any combination can be selected. 0: The corresponding channel is not selected 1: The corresponding channel is selected	RW	0x0

40.7.2.8 Sequence x Channel Start Register ADC0_STARTx(0x28/0x58)

x = A/B.

Bit	Name	Description	R/W	Reset Value
31:1	--	Reserved Bits	RO	0x0
0	START	ADC0 Conversion Sequence x Start 0: Stop conversion 1: Start conversion This bit becomes 1 when software writes 1 or hardware triggers ADC0 conversion sequence x, becomes 0 when software writes 0 or ADC0 conversion completes Note: During sequence scan, writing 0 to START will clear the remaining unscanned channels; next start will restart scanning the sequence;	RW	0x0

40.7.2.9 Sequence x Data Register ADC0_DATx(0x2C/0x5C)

x = A/B.

Bit	Name	Description	R/W	Reset Value
31:20	--	Reserved	RO	0x0
19:16	CHN	Sequence x Current Conversion Channel Number Writable only when ADC0_CTRLx.CHMOD is 0, read-only otherwise	WR	0x0
15:0	DAT	Sequence x Current Channel Conversion Data	RO	0x0

40.7.2.10 Sequence x Watchdog Threshold Register ADC0_WDTHLx(0x30/0x60)

x = A/B.

Bit	Name	Description	R/W	Reset Value
31:16	WDHT	Watchdog Threshold Upper Limit	RW	0xFFFF
15:0	WDLT	Watchdog Threshold Lower Limit Note: This threshold is directly compared with the ADC0_DATx	RW	0x0

		value; if configured for left alignment, the high bits of WDHT, WDLT are used for comparison;		

40.7.2.11 Sequence x Watchdog Channel Select Register ADC0_WCHSELx(0x34/0x64)

x = A/B.

Bit	Name	Description	R/W	Reset Value
31:15	--	Reserved Bits	RO	0x0
14:0	WCHSEL	Sequence x Watchdog Channel Select, each bit represents a channel	RW	0x0

40.7.2.12 Sequence x DMA Receive Base Address Register ADC0_DMA_RBADRx(0x38/0x68)

x = A/B.

Bit	Name	Description	R/W	Reset Value
31:18	--	Reserved Bits	RO	0x0
17:0	DMA_RBADR	DMA Receive Base Address, lowest bits fixed to 0	RW	0x0

40.7.2.13 Sequence x DMA Receive Length Register ADC0_DMA_RLENx(0x3C/0x6C)

x = A/B.

Bit	Name	Description	R/W	Reset Value
31:18	--	Reserved Bits	RO	0x0
17:0	DMA_RLEN	DMA Receive Length, lowest bits fixed to 0	RW	0x0

40.7.2.14 Sequence x DMA Receive Current Address Register ADC0_DMA_RADRx(0x40/0x70)

x = A/B.

Bit	Name	Description	R/W	Reset Value
31:18	--	Reserved Bits	RO	0x0
17:0	DMA_RADR	DMA Receive Current Address, lowest bits fixed to 0	RO	0x0

40.7.2.15 Sequence x Interrupt Enable Register ADC0_IEx(0x44/0x74)

x = A/B.

Bit	Name	Description	R/W	Reset Value
31:6	--	Reserved Bits	RO	0x0
5	DMA_ERR	DMA Receive Data Overwrite Interrupt Enable 0: Disable 1: Enable	RW	0x0
4	DMA_FIE	DMA Receive Data Full Interrupt Enable 0: Disable	RW	0x0

		1: Enable		
3	DMA_HIE	DMA Receive Data Half Full Interrupt Enable 0: Disable 1: Enable	RW	0x0
2	WDTE	Sequence x Watchdog Event Interrupt Enable Bit 0: Disable 1: Enable	RW	0x0
1	EOCAE	Sequence x Conversion Complete Interrupt Enable Bit 0: Disable 1: Enable	RW	0x0
0	EOCE	Sequence x Current Channel Conversion Complete Interrupt Enable Bit 0: Disable 1: Enable	RW	0x0

40.7.2.16 Sequence x Flag Register ADC0_IFx(0x48/0x78)

x = A/B.

Bit	Name	Description	R/W	Reset Value
31:6	--	Reserved Bits	RO	0x0
5	DMA_ERR	DMA Receive Data Overwrite Flag Bit 0: No data overwrite error occurred 1: Data overwrite error occurred Data overwrite error means the previous data was not written to SRAM via DMA, and new data was generated, overwriting the previous data;	W1C	0x0
4	DMA_FDONE	DMA Receive Full Interrupt Flag Bit 0: DMA receive data not full 1: DMA receive data full	W1C	0x0
3	DMA_HDONE	DMA Receive Half Full Interrupt Flag Bit 0: DMA receive data not half full 1: DMA receive data half full	W1C	0x0
2	WDTF	Sequence x Watchdog Event Flag Bit 0: No watchdog event occurred 1: Watchdog event occurred	W1C	0x0
1	EOCAF	Sequence x Conversion Complete Flag Bit 0: All enabled channels conversion not completed 1: All enabled channels conversion completed	W1C	0x0
0	EOC	Sequence x Current Channel Conversion Complete Flag Bit 0: Current channel conversion not completed 1: Current channel conversion completed	W1C	0x0

40.7.2.17 Calibration Parameter Register ADC0_CAL(0x80)

Bit	Name	Description	R/W	Reset Value
31:25	--	Reserved Bits	RO	0x0
24	CAL_EN	Calibration Enable =0: Do not use hardware calibration, output code value is ADC0 raw conversion code value =1: Use hardware calibration, output code value is the calibrated code value This bit can only be configured when ADC0 is idle, i.e., when both ADC0_STARTA and ADC0_STARTB are 0; Note: After configuring this register, wait at least 16 cycles before starting ADC conversion operation;	RW	0x0
23:12	ZTBG	ZEROT Calibration Coefficient for BUFFER enabled, unsigned number Theoretically 0.5 ≤ ZTBG ≤ 2, so ZTBG = round(gain*2048), ZTBG selectable range is 0.000488~1.999512 This bit can only be configured when ADC0 is idle, i.e., when both ADC0_STARTA and ADC0_STARTB are 0;	RW	0x800
11:0	ZTG	ZEROT Calibration Coefficient for BUFFER disabled, unsigned number Theoretically 0.5 ≤ ZTG ≤ 2, so ZTG = round(gain*2048), ZTG selectable range is 0.000488~1.999512 This bit can only be configured when ADC0 is idle, i.e., when both ADC0_STARTA and ADC0_STARTB are 0;	RW	0x800

40.7.2.18 Gain Compensation Register ADC0_DOS(0x84)

Bit	Name	Description	R/W	Reset Value
31:12	--	Reserved Bits	RO	0x0
11:0	DOS	Code Value Compensation Amount, highest bit is sign bit, selectable range -2048~2047 When starting the calibration mechanism, DOS needs to be set to 0, after calibration is completed, fine-tune according to the output result;	RW	0x0

40.7.2.19 BGRLP Channel Ideal Code Value Register ADC0_BGRLP(0x88)

Bit	Name	Description	R/W	Reset Value
31:16	BGRLP_COREB	When reference voltage is 3.3V, the accumulated 16 times code value of BGRLP channel with BUFFER enabled, unsigned number	RO	0x0
15:0	BGRLP_CORE	When reference voltage is 3.3V, the accumulated 16 times	RO	0x0

		code value of BGRLP channel with BUFFER disabled, unsigned number		

40.7.2.20 Dummy Conversion Configuration Register ADC0_DCONV(0x94)

Bit	Name	Description	R/W	Reset Value
31:2	--	Reserved Bits	RO	0x0
1:0	DUMMY_NUM	Post-power-up Dummy Conversion Count Configuration Post-power-up dummy conversion count is (DUMMY_NUM), configurable as 0~3 times;	RW	0x1

40.8 Software Configuration Process and Application Description

40.8.1 Normal Usage Process of ADC0

- Write to ADC0_CFG to configure the ADC0 clock frequency, ADC0 resolution, data alignment and register auto-clear function, BUFFER enable (optional), and parameters for each time period;
- Write to ADC0_OVSCHSEL to enable channels that require oversampling processing (optional);
- Write to ADC0_SAMPT0~3 to configure the sampling time for each channel of ADC0;
- Write to ADC0_CTRLx and ADC0_CHSELx to configure the conversion mode and conversion channels for the two sequences;
- Write to ADC0_WDTx and ADC0_WCHSEL to configure the analog watchdog function for the sequences;
- Write to ADC0_DMA_RBADRx, ADC0_DMA_RLENx, and ADC0_DMA_RADRx to configure the DMA function;
- Write to ADC0_IEx to enable interrupts;
- Write to ADC0_STARTx to start conversion;

  or

  Write to PE_TRGSEL32/PE_TRGSEL33 of the AOS module to enable KBI or SIMPTC triggering, wait for the hardware trigger condition, and start conversion;
- Read the ADC0_IFx flag bit to determine whether the conversion is completed;
- Read the ADC0_DATx data or read the corresponding location in SRAM;

41 DAC

41.1 Introduction

Digital-to-Analog Converter (DAC) is a device that converts discrete digital signals into continuous analog signals. The DAC can convert 12-bit digital data into analog voltage output, and the data can be configured in left-aligned or right-aligned format. When external triggering is enabled, DMA can be used to update digital input data. The DAC module is equipped with an output buffer, which can provide greater driving capability when enabled.

41.2 Features

The main features of the DAC are as follows:

- Supports 12-bit DAC output with a maximum output rate of 200 Ksps
- 12-bit conversion data can be configured in left-aligned or right-aligned format
- Supports output buffer to improve voltage driving capability
- Supports output buffer offset calibration
- Supports continuous output and triggered output; trigger sources include software trigger and external trigger
- External trigger sources include KBI0 (channels 8~11), KBI1 (channels 8~11), and SIMPTC0~3 (channel 2), managed by the AOS module
- Supports DMA function when using external trigger
- In continuous output mode, supports A/D conversion priority function to reduce interference with A/D conversion
- Supports sample and hold mode to reduce power consumption
- Supports selecting the output channel as PAD or ADC0

41.3 Block Diagram

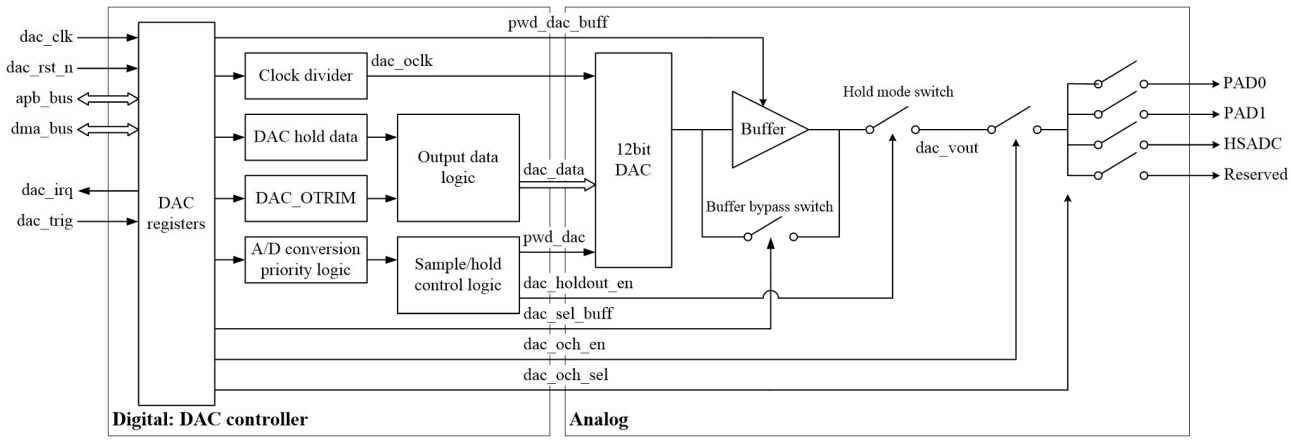


Figure 41-1 DAC Block Diagram

41.4 Pin Definition

Pin Name	Pin Number	Description
DACO0	PA15	DAC Analog Output 0
DACO1	PA05	DAC Analog Output 1

41.5 Clock and Reset

The DAC module only requires a bus clock, which follows the system main frequency.

The reset signal of the DAC module takes effect upon core reset and PIN reset, and the DAC also supports independent software reset.

41.6 Functional Description

41.6.1 Data Alignment Format

Software configures the data format through the DAC control register DAC_CFG.DPSEL:

When DAC_CFG.DPSEL=0, 12-bit data right-alignment is selected, and the user must write data to bits DAC_DHOLD[11:0] of the 12-bit data hold register.

When DAC_CFG.DPSEL=1, 12-bit data left-alignment is selected, and the user must write data to bits DAC_DHOLD[15:4] of the 12-bit data hold register.

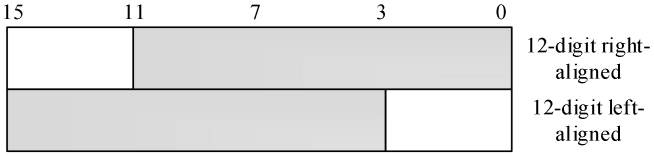


Figure 41-2 Data Alignment Format

Based on the operation of the data hold register, after corresponding shifting, the written data is transferred to the internal data storage register. Subsequently, the content of the internal register is either automatically transferred to the DOUT register or triggered to be transferred to the DOUT register by software trigger or external event trigger.

41.6.2 DAC Output Voltage

The analog output voltage on the DAC pin is determined by the following equation:

Vout = ((Data／4096) × Vref)

Where Vout is the analog output voltage, Data is the digital input value, and Vref is the reference voltage VREFP.

41.6.3 DAC Output Buffer

To reduce output impedance and drive external loads without an external operational amplifier, the DAC module integrates an output buffer operational amplifier internally. By default, the output buffer is disabled. The user can select whether to enable the buffer by setting the control register DAC_CFG.BUFF_EN, and select whether to output through the buffer by setting DAC_CFG.SEL_BUFF.

When the DAC uses the output buffer, due to the characteristics of the output buffer, the output voltage may introduce an offset voltage Voffset. The analog output voltage is as follows:

Vout = ((Data／4096) × Vref) + Voffset

This offset voltage can be compensated for the output voltage by writing a voltage calibration value to the module calibration register DAC_OTRIM.

The user can compensate the DAC output by modifying the module calibration register DAC_OTRIM. Software calibration is only valid when the DAC uses the output buffer. During calibration, the software configures the buffer output to be connected to the ADC0 input, writes the middle code value Ddac=0x800 to the DAC, and starts D/A conversion. Start ADC0 to convert the output signal of the DAC into a 12-bit digital code value Dadc. The software writes a voltage calibration value to DAC_OTRIM according to the difference between Ddac and Dadc, and starts the DAC and ADC0 again until Ddac=Dadc.

41.6.4 DAC Output Modes

41.6.4.1 Continuous Output Mode

Software writes to the control register DAC_CFG.TRIG_CTRL=2'b00 to configure the DAC into continuous output mode. In continuous output mode, the DAC performs digital-to-analog conversion immediately after the software writes to the data hold register. The data update frequency must not be faster than the conversion clock of the DAC; otherwise, the output voltage cannot be guaranteed to be correctly established.

41.6.4.2 Software-Triggered Output Mode

Software writes to the control register DAC_CFG.TRIG_CTRL=2'b01 to configure the DAC into software-triggered output mode. When software triggering is selected, writing DAC_SWT.SWT=1 triggers one D/A conversion.

41.6.4.3 External-Triggered Output Mode

Software writes to the control register DAC_CFG.TRIG_CTRL=2'b1x to configure the DAC into external-triggered output mode. The trigger source is configured for the DAC through the AOS module, and one D/A conversion is triggered when the rising edge of the trigger signal arrives.

41.6.5 DMA Function

When external triggering is enabled, the DMA function is enabled by setting the control register DAC_CFG.DMA_EN bit. The DAC supports the word-aligned transmission mode of dedicated DMA. After the DMA controller receives the DMA request, it automatically updates the data to the internal buffer register.

When the external-triggered output mode is enabled, the first external trigger signal after the DAC is enabled will generate a DMA request to read 32-bit data and cache it in the internal buffer register. The first output voltage of the DAC is the pre-written DAC_DHOLD value before DMA is enabled; the second external trigger signal updates the lower 16-bit data of the internal buffer register to the DAC_DOUT register and outputs it through the DAC; the third external trigger signal updates the upper 16-bit data of the internal buffer register to the DAC_DOUT register and outputs it through the DAC, while generating a DMA request.

If the second external trigger arrives before the data in the internal buffer register is read, the new trigger request is not responded to, and a DMA underrun error event occurs. If the DERR_IE bit in the DAC_IE register is set to 1, an interrupt is generated.

DMA can be configured in single mode or circular mode:

- When configured in DMA single mode, after the conversion of the specified length of DMA data is completed, a DMA completion flag DDONE_IF is generated, DMA_EN is automatically disabled, no more DMA requests are generated, and the DAC_DHOLD value is output when subsequent external trigger signals arrive;
- When configured in DMA circular mode, after the conversion of the specified length of DMA data is completed, a DMA completion flag DDONE_IF is generated, the internal DMA start address and data length information are reloaded, and the DMA data is output when an external trigger arrives.

41.6.6 A/D Conversion Priority Function

A transient surge current may occur on the analog power supply when the DAC starts D/A conversion, which may interfere with the ADC0 that is working. The A/D conversion priority function can effectively avoid this interference by changing the update timing of the D/A conversion data.

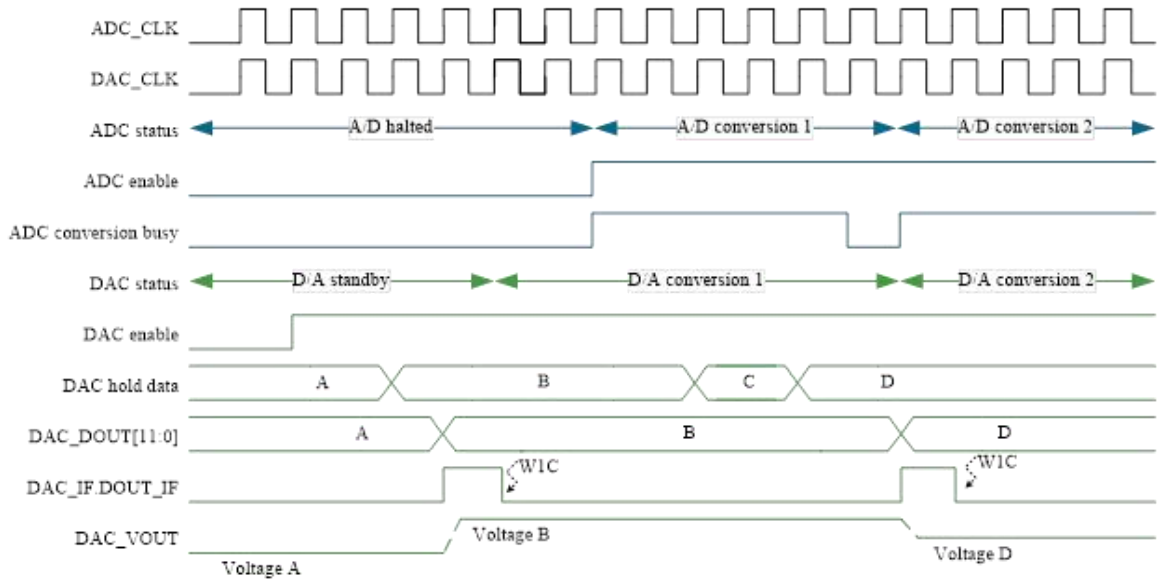


Figure 41-3 A/D Conversion Priority Function Timing Diagram

Set DAC_ADP.ADPEN=1 to enter the A/D conversion priority function. If ADC0 is performing conversion when writing to the data hold register, the digital-to-analog conversion will not start immediately. It can be configured to start after the current A/D conversion is completed or after the ADC0 conversion times out. If ADC0 is in a stopped state when the data hold register is rewritten, the D/A conversion starts at the next conversion clock after writing.

To avoid loss of conversion data, first check the value of the interrupt flag register DAC_IF.DOUT_IF to confirm that the data conversion is completed before writing the next data.

41.6.7 Sample and Hold Mode

The DAC output supports sample and hold mode. The sample and hold mode is enabled by configuring the control register DAC_CFG.SH_MODE.

When sample and hold is not enabled, the DAC analog output is continuously driven by the DAC, resulting in stable current consumption during the operation of the DAC.

After enabling the sample and hold mode, the DAC only drives the DAC output for a period of time after completing the conversion of the data written to the DAC_DOUT register this time, then automatically turns off the DAC. The DAC output signal becomes high-impedance, and an off-chip capacitor is required to maintain the previous voltage state; the internal timer counts the conversion clock, and re-drives the DAC output after an interval of programmable time. The sample and hold mode can be divided into 2 phases:

- Sample Phase

Charge the sample and hold capacitor to the desired voltage. The charging time depends on the capacitor value selected by the user. The time of the sample phase is configured through the TSMP bit of the sample and hold time register DAC_SHT.

- Hold Phase

Turn off the DAC and output buffer, set the output to high-impedance, and the sample and hold capacitor maintains the voltage state. The time of the hold phase is configured through the THOLD bit of the sample and hold time register DAC_SHT.

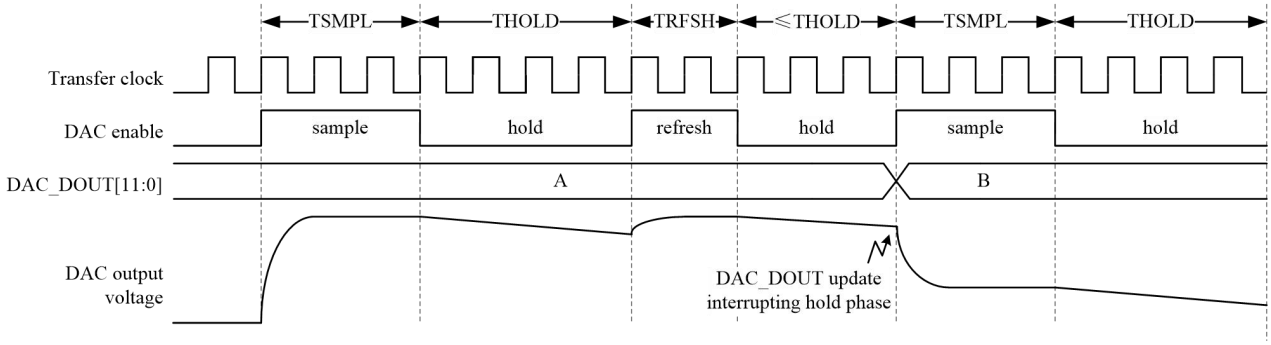


Figure 41-4 Hold Phase Timing Diagram

41.6.8 Interrupts

The DAC module supports the following interrupt events.

Interrupt Event	Flag Bit	Enable Bit
Data Output Register Update Interrupt	DOUT_IF	DOUT_IE
DMA Completion Interrupt	DDONE_IF	DDONE_IE
DMA Error Interrupt	DERR_IF	DERR_IE
A/D Conversion Wait Timeout Interrupt	ADPOV_IF	ADPOV_IE

Table 41-1 DAC Module Interrupt Events Table

41.7 Registers

41.7.1 Register Map Table

Module Name	Physical Address	Mapped Address
DAC (Small Core Area)	0x2009_8000	0x2009_8000
DAC (Large Core Area)	0xA009_8000	0xA009_8000

Register Name	Address Offset	Description
DAC_CFG	0x0	DAC Control Register
DAC_ADPOT	0x4	DAC A/D Conversion Priority Timeout Register
DAC_SHT	0x8	DAC Sample and Hold Time Register
DAC_OTRIM	0xC	DAC Buffer Calibration Register
DAC_SWT	0x10	DAC Software Trigger Register
DAC_IE	0x14	DAC Interrupt Enable Register
DAC_IF	0x18	DAC Interrupt Flag Register
DAC_DHOLD	0x20	DAC Data Hold Register
DAC_DOUT	0x24	DAC Data Output Register
DAC_DMA_CFG	0x30	DAC DMA Configuration Register
DAC_DMA_BADR	0x34	DAC DMA Base Address Register
DAC_DMA_LEN	0x38	DAC DMA Data Length Register

DAC_DMA_ADR	0x3C	DAC DMA Current Address Register

41.7.2 Register Description

41.7.2.1 Control Register DAC_CFG(0x00)

<table>
<tr><th>Bit</th><th>Name</th><th>Description</th><th>R/W</th><th>Reset Value</th></tr>
<tr><td>31:16</td><td>CONV_DIV</td><td>Output Conversion Clock Division Factor Configuration, conversion clock frequency as follows:
$$f_{conversion_clock} = \frac{f_{system_clock}}{CONV_DIV + 1}$$
Under different system clock frequencies, the conversion clock division factor should not be less than the minimum configuration number in the table below.
<table>
<tr><th>System Frequency</th><th>Min Config Number</th><th>System Frequency</th><th>Min Config Number</th></tr>
<tr><td>58.9824 MHz</td><td>294</td><td>3.6864 MHz</td><td>18</td></tr>
<tr><td>29.4912 MHz</td><td>147</td><td>1.8432 MHz</td><td>9</td></tr>
<tr><td>14.7456 MHz</td><td>73</td><td>32.768 KHz</td><td>1</td></tr>
<tr><td>7.3728 MHz</td><td>36</td><td>-</td><td>-</td></tr>
</table></td><td>RW</td><td>0x0</td></tr>
<tr><td>15</td><td>-</td><td>Reserved</td><td>RO</td><td>0x0</td></tr>
<tr><td>14:13</td><td>OCH_SEL</td><td>Analog Output Channel Selection
0x0: Output to DACO0 (PA15)
0x1: Output to DACO1 (PA05)
0x2: Output to ADC0
0x3: Reserved</td><td>RW</td><td>0x0</td></tr>
<tr><td>12</td><td>OCH_EN</td><td>Analog Output Channel Enable;
0: All output channels are turned off;
1: DAC outputs to PAD or ADC0 according to OCH_SEL selection;</td><td>RW</td><td>0x0</td></tr>
<tr><td>11:10</td><td>TBUFF</td><td>Wait for output Buffer power-up stabilization time configuration, unit is conversion clock cycle Toclk;
0x0: 1.5Toclk
0x1: 2.5Toclk (default)
0x2: 3.5Toclk
0x3: 4.5Toclk
Note: If the output Buffer is enabled and the output channel is turned on, it is necessary to wait for the DAC's output Buffer to power up and stabilize. The first output voltage lasts for half a DAC conversion clock cycle, and subsequent conversions last for a full conversion clock cycle.</td><td>RW</td><td>0x1</td></tr>
<tr><td>9</td><td>ADP_CTRL</td><td>When A/D conversion priority function is enabled, wait timeout handling method
0: After timeout, generate A/D conversion wait timeout interrupt,</td><td>RW</td><td>0x0</td></tr>
</table>

		continue waiting until A/D conversion completes, then perform next D/A conversion; 1: After timeout, generate A/D conversion wait timeout interrupt, do not wait for A/D conversion to complete, immediately perform next D/A conversion;		
8	ADPEN	A/D Conversion Priority Function Enable Bit 0: Disable A/D conversion priority function; 1: Enable A/D conversion priority function;	RW	0x0
7	DPSEL	Data Alignment Format Configuration 0: 12-bit data right-aligned; 1: 12-bit data left-aligned;	RW	0x0
6	SH_MODE	Sample and Hold Mode Configuration 0: Disable sample and hold mode; 1: Enable sample and hold mode;	RW	0x0
5:4	TRIG_CTRL	DAC Trigger Mode Configuration 2'b00: Continuous output mode; 2'b01: Software trigger, write DAC_SWT register to trigger one conversion; 2'b1x: External trigger, external trigger source configured by AOS module;	RW	0x0
3	INIT_MODE	Initial Output State 0: Analog output high-impedance state when channel is disabled; 1: Analog output grounded when channel is disabled;	RW	0x0
2	SEL_BUFF	Output Buffer Selection Enable 0: Output without passing through output Buffer; 1: Output through output Buffer;	RW	0x0
1	BUFF_EN	Output Buffer Enable Control 0: Disable output Buffer; 1: Enable output Buffer;	RW	0x0
0	DAC_EN	DAC Channel Conversion Enable 0: Disable DAC conversion; 1: Enable DAC conversion;	RW	0x0

41.7.2.2 A/D Conversion Priority Timeout Register DAC_ADPOT(0x04)

Bit	Name	Description	R/W	Reset Value
31:20	-	Reserved	RO	0x0
19:16	DAC_BUSYT	DAC Conversion Busy Flag Assertion Time Setting, unit is system clock cycle Tsys_clk; Configured by software according to current system clock frequency, recommended time is 0.1us; When DAC conversion is busy, ADC0 will wait for DAC conversion to complete before starting; 0x0: 1 Tsys_clk	RO	0x0

<table>
<tr><td></td><td></td><td>0x1: 2 Tsys_clk
.....
0xF: 16 Tsys_clk
<table><tr><td>System Clock Frequency</td><td>Recommended Value</td><td>System Clock Frequency</td><td>Recommended Value</td></tr><tr><td>58.9824MHz</td><td>0x5</td><td>3.6864MHz</td><td>0x0</td></tr><tr><td>29.4912MHz</td><td>0x2</td><td>1.8432MHz</td><td>0x0</td></tr><tr><td>14.7456MHz</td><td>0x1</td><td>32.768KHZ</td><td>0x0</td></tr><tr><td>7.3728MHz</td><td>0x0</td><td>-</td><td>-</td></tr></table>Note: The conversion busy flag assertion time DAC_BUSYT needs to be less than or equal to the output conversion clock division factor CONV_DIV;</td><td></td><td></td></tr>
<tr><td>15:0</td><td>ADP_OVT</td><td>A/D Conversion Priority Function Timeout Configuration, unit is conversion clock cycle Toclk;
0x0: 1Toclk
0x1: 2Toclk
.....
0xFFFF: 65536Toclk</td><td>RW</td><td>0x0</td></tr>
</table>

41.7.2.3 Sample and Hold Time Register DAC_SHT(0x08)

Bit	Name	Description	R/W	Reset Value
31:16	THOLD	Hold time in sample and hold mode, unit is conversion clock cycle Toclk; 0x0: 1Toclk 0x1: 2Toclk 0xFFFF: 65536Toclk	RW	0x0
15:0	TSMPL	Sampling time in sample and hold mode, unit is conversion clock cycle Toclk; 0x0: 1Toclk 0x1: 2Toclk 0xFFFF: 65536Toclk	RW	0x0

41.7.2.4 Module Calibration Register DAC_OTRIM(0x0C)

Bit	Name	Description	R/W	Reset Value
31:8	-	Reserved	RO	0x0
7:0	OTRIM	When output buffer is enabled, the compensation value for the output voltage, the compensation value is an 8-bit signed number	RW	0x0

		in two's complement form; Compensates the output code value after enabling the output buffer.		

41.7.2.5 Software Trigger Register DAC_SWT(0x10)

Bit	Name	Description	R/W	Reset Value
31:1	-	Reserved	RO	0x0
0	SWT	Software trigger one D/A conversion, automatically cleared by hardware. Write 1 to generate one software trigger signal, write 0 is invalid.	WO	0x0

41.7.2.6 Interrupt Enable Register DAC_IE(0x14)

Bit	Name	Description	R/W	Reset Value
31:4	-	Reserved	RO	0x0
3	ADPOV_IE	A/D Conversion Wait Timeout Interrupt Enable 0: Disable 1: Enable	RW	0x0
2	DERR_IE	DMA Error Interrupt Enable 0: Disable 1: Enable	RW	0x0
1	DDONE_IE	DMA Complete Interrupt Enable 0: Disable 1: Enable	RW	0x0
0	DOUT_IE	Data Output Register Update Interrupt Enable 0: Disable 1: Enable	RW	0x0

41.7.2.7 Interrupt Flag Register DAC_IF(0x18)

Bit	Name	Description	R/W	Reset Value
31:5	-	Reserved	RO	0x0
4	CONV_BUSY	DAC Busy Flag, set by DAC_ADPOT.DAC_BUSYT for the assertion time during each D/A conversion; Set and cleared automatically by hardware;	RO	0x0
3	ADPOV_IF	A/D Conversion Wait Timeout Interrupt Flag, write 1 to clear;	W1C	0x0
2	DERR_IF	DMA Error Interrupt Flag, write 1 to clear; Occurs when a new DMA request is externally triggered before the current DMA operation is completed; Or when a new DMA request is externally triggered before the current DMA data conversion is completed;	W1C	0x0

1	DDONE_IF	DMA Complete Interrupt Flag, write 1 to clear;	W1C	0x0
0	DOUT_IF	Data Output Register Update Interrupt Flag, write 1 to clear;	W1C	0x0

41.7.2.8 Data Hold Register DAC_DHOLD(0x20)

Bit	Name	Description	R/W	Reset Value
31:16	-	Reserved	RO	0x0
15:0	DHOLD	When DAC_CFG.DPSEL=0 (data right-aligned), write the data to be converted to DHOLD[11:0], DHOLD[15:12] are reserved bits; When DAC_CFG.DPSEL=1 (data left-aligned), write the data to be converted to DHOLD[15:4], DHOLD[3:0] are reserved bits;	RW	0x0

41.7.2.9 Data Output Register DAC_DOUT(0x24)

Bit	Name	Description	R/W	Reset Value
31:12	-	Reserved	RO	0x0
11:0	DOUT	DAC's 12-bit output data, read-only	RO	0x0

41.7.2.10 DMA Configuration Register DAC_DMA_CFG（0x30）

Bit	Name	Description	R/W	Reset Value
31:2	--	Reserved	RO	0x0
1	CIRC	DMA Circular Mode Configuration 0: Single mode, stop after current DMA specified length data transfer is completed; 1: Circular mode, after current DMA specified length data transfer is completed, automatically reload DMA start address and data length;	RW	0x0
0	DMA_EN	DMA Enable Configuration, automatically cleared after single mode transfer completion, needs software configuration to turn off in circular mode; 0: Disable DMA; 1: Enable DMA (external trigger must be enabled);	RW	0x0

41.7.2.11 DAC_DMA_BADR（0x34）

Bit	Name	Description	R/W	Reset Value
31:16	--	Reserved	RO	0x0
15:0	BADR	DMA Base Address (Word address)	RW	0x0

41.7.2.12 DAC_DMA_LEN（0x38）

Bit	Name	Description	R/W	Reset Value
31:16	--	Reserved	RO	0x0
15:0	LEN	DMA Data Length, unit is Word.	RW	0x0

41.7.2.13 DAC_DMA_ADR（0x3C）

Bit	Name	Description	R/W	Reset Value
31:16	--	Reserved	RO	0x0
15:0	ADR	DMA Current Address (Word address)	RO	0x0

41.8 Software Configuration Process and Application Description

41.8.1 Continuous Mode Application Process

- Enable the DAC module clock gating；
- Write to the control register DAC_CFG.TRIG_CTRL=0 for continuous output mode, and set the conversion clock frequency, output buffer enable, sample and hold mode, data alignment format, A/D conversion priority function and other configurations；
- Write to the A/D conversion priority timeout register DAC_ADPOT to configure the waiting time for ADC0 conversion and the DAC conversion busy time；
- If the sample and hold mode is enabled, write to the sample and hold register DAC_SHT to configure the sample time TSMPL and hold time THOLD；
- Write to the interrupt enable register DAC_IE to configure interrupt enable；
- Write to the control register DAC_CFG.DAC_EN=1 to enable D/A conversion；
- According to the data alignment format configuration, write to the data hold register DAC_DHOLD；
- Read the interrupt flag register DAC_IF to check the data update flag and write 1 to clear it；
- Repeat steps 7 and 8 until all data conversions are completed；
- After the DAC conversion is completed, write to the control register DAC_CFG.DAC_EN=0；
- Disable the DAC module clock；

41.8.2 Software-Triggered Mode Application Process

- Enable the DAC module clock gating；
- Write to the control register DAC_CFG.TRIG_CTRL=1 to select software-triggered output mode, and set the conversion clock frequency, output buffer enable, sample and hold mode, data alignment format；
- Write to the A/D conversion priority timeout register DAC_ADPOT to configure the waiting time for ADC0 conversion and the DAC conversion busy time；
- If the sample and hold mode is enabled, write to the sample and hold time register DAC_SHT to configure the sample time TSMPL and hold time THOLD；
- Write to the interrupt enable register DAC_IE to configure interrupt enable；
- Write to the control register DAC_CFG.DAC_EN=1 to enable D/A conversion；
- Repeat the following steps until all data conversions are completed：
  - According to the data alignment format configuration, write to the data hold register DAC_DHOLD；

- Write to the software trigger register DAC_SWT to trigger one data conversion;
- Read the interrupt flag register DAC_IF to check the data update flag and write 1 to clear it;

- After the DAC conversion is completed, write to the control register DAC_CFG.DAC_EN=0;
- Disable the DAC module clock;

41.8.3 Hardware-Triggered Mode Application Process

- Enable the DAC module clock gating;
- Write to the control register DAC_CFG.TRIG_CTRL=2 to select hardware-triggered output mode, and set the conversion clock frequency, output buffer enable, sample and hold mode, data alignment format;
- Write to the A/D conversion priority timeout register DAC_ADPOT to configure the waiting time for ADC0 conversion and the DAC conversion busy time;
- If DMA needs to be used, write to the DAC_DMA_CFG register to configure the DMA circular mode;
- If DMA is enabled, configure the external trigger source in the PE_TRGSEL register corresponding to the DAC in the AOS module;
- If the DMA function is enabled, it is necessary to configure the DMA channel, data source address, destination address, amount of data to be transferred, transmission control configuration, interrupt enable of the DMA controller module, and enable the DMA channel;
- If the DMA function is enabled, the data hold register DAC_DHOLD can be written to configure the initial output preset value before enabling the DAC conversion;
- Write to the DAC_DMA_CFG register to enable DMA;
- If the sample and hold mode is enabled, write to the sample and hold time register DAC_SHT to configure the sample time TSMPL and hold time THOLD;
- Write to the interrupt enable register DAC_IE to configure interrupt enable;
- Write to the control register DAC_CFG.DAC_EN=1 to enable D/A conversion;
- Repeat the following steps until all data conversions are completed:
  - If DMA is enabled, the rising edge of the external trigger signal generates a DMA request and triggers one data conversion;
  - If DMA is not enabled, according to the data alignment format configuration, write to the data hold register DAC_DHOLD, and the external trigger signal triggers one data conversion;
  - Read the interrupt flag register DAC_IF to check the data update flag and write 1 to clear it;
- After the DAC conversion is completed, write to the control register DAC_CFG.DAC_EN=0;
- Disable the DAC module clock;

41.8.4 Output Buffer Calibration Process

- Enable the ADC0 control module gating;
- Configure the ADC0 control module and set the ADC0 input source to DAC output;
- Enable the DAC module clock gating;
- Write to the control register DAC_CFG to set the conversion clock frequency and enable the output buffer;
- Write to the data hold register DAC_DHOLD=0x800 intermediate value;
- Write to the control register DAC_CFG.DAC_EN=1 to start DAC conversion;
- Execute the calibration algorithm:
  - Start writing to the module calibration register DAC_OTRIM from 0x0;
  - Read the ADC0 completion flag to check whether the conversion is completed, and read whether the output code value of ADC0 is 0x800;

- When the code value written to the DAC is equal to the output code value of ADC0, it indicates that the correct calibration value has been found; otherwise, increment the code value until the correct calibration value is found.

41.8.5 Other Application Notes

- When DAC_EN=1, that is, the DAC channel conversion is enabled, do not modify the configuration of DAC_CFG[31:1].
- When using the sample and hold function, it is recommended to connect an external capacitor not exceeding 0.1uF.
- When connecting an external resistive load, the resistance value should not be less than 1Mohm.
- When using the output Buffer, to reduce errors, the output voltage amplitude should be controlled between 0.2V and Vcc-0.2V.
- The DAC output and ADC0 input cannot be multiplexed to the same pin.
- If the DAC module and ADC0 module are enabled at the same time, the DAC module output will be interfered by ADC0, and the maximum jitter of the DAC output voltage is 4mV.
- If the output Buffer is enabled and the output channel OCH_EN is turned on, the first output voltage after the DAC channel conversion is enabled is half a DAC conversion clock cycle, and subsequent conversions last for a complete 1 conversion clock cycle.

42 Debug Support (DBG)

42.1 Overview

The Core0/Core1 kernels integrate a hardware debug module. Dual-core debug control is implemented through the chain-linked TAP (Test Access Port) controllers in the DTM (Debug Transport Module). When debugging is not disabled, connecting a debugger allows the use of the hardware debug module to debug the chip.

A hardware debug module is integrated in both the Core0 and Core1 kernels. Dual-core debug control is implemented via the daisy-chained TAP (Test Access Port) controllers in the DTM (Debug Transport Module). When debugging is not disabled, the hardware debug module can be used to debug the chip by connecting a debugger.

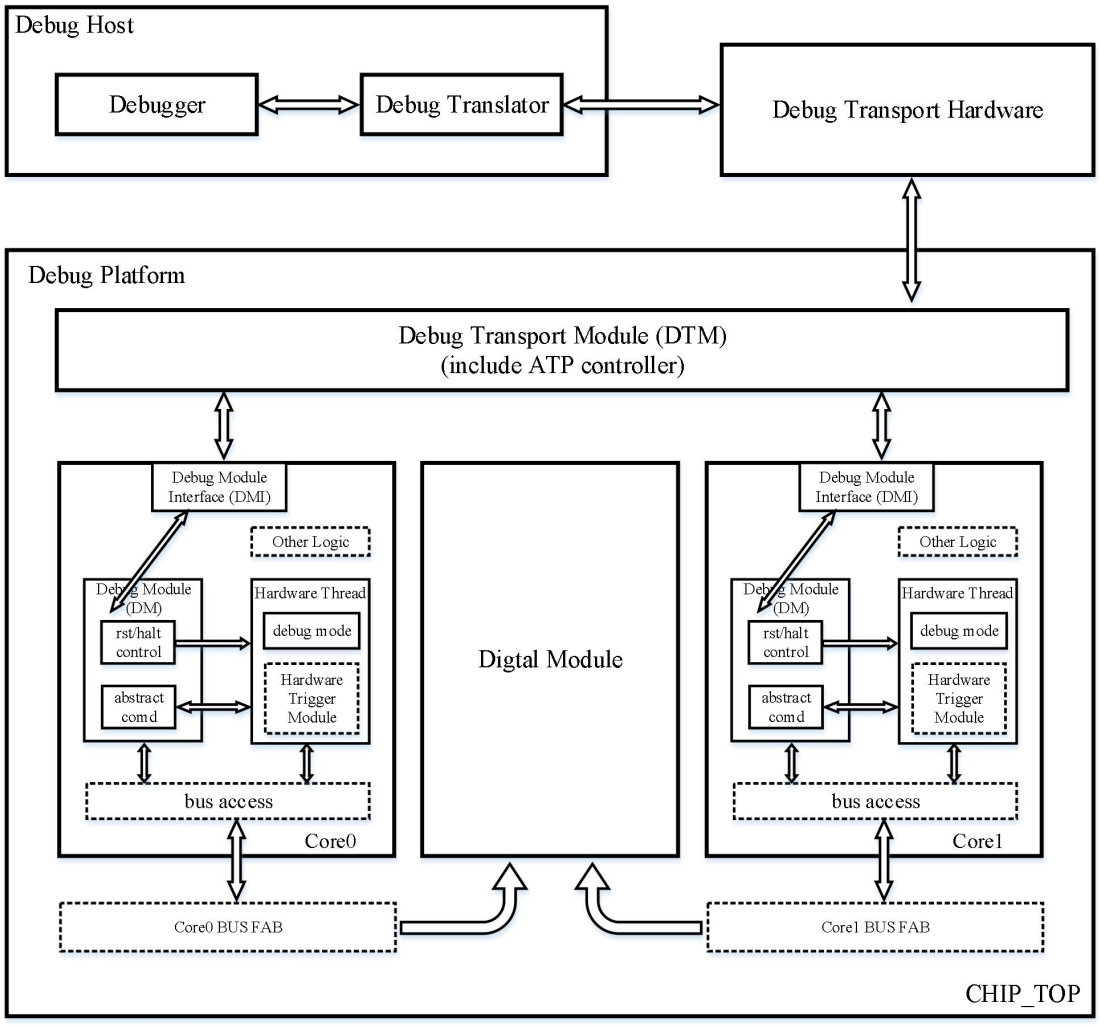


Figure 42-1 Block Diagram of the Debug Module Structure

The debug interface supports 2-wire cJTAG and 4-wire JTAG, with the cJTAG interface as the default. Different debug interfaces can be selected through NVR configuration.

Pin Name	2-wire cJTAG（Default）		4-wire JTAG		Pin Assignment	Reset State
	Type	Description	Type	Description		
JTMS	IO	Data Input / JTAG Test Mode Select / Data Output	I	JTAG Test Mode Select	PG15	Input Pull-Up
JTCK	I	JTAG Test Clock	I	JTAG Test Clock	PG14	Input Pull-Down
JTDI	-	-	I	JTAG Test Data Input	PG11	General-Purpose GPIO
JTDO	-	-	O	JTAG Test Data Output	PG10	General-Purpose GPIO

42.2 JTAG TAP Controller

The JTAG TAP controller is based on the IEEE 1149.1 standard. The following figure shows the state machine of the TAP controller. State transitions are affected by the JTMS input, and there are two main scan paths: the DR scan path and the IR scan path. Its working principle is as follows:

- IR Scan Path:
  - When the state jumps to Capture-IR, the IR value is transmitted to the scan path, which is connected between JTDI and JTDO.
  - When the state jumps to Shift-IR, the IR scan path shifts one bit on each rising edge of JTCK. At this time:
  - ✓ The JTDO output value is the LSB (Least Significant Bit) of the IR scan path.
  - ✓ The nth bit of the IR scan path shifts to the (n-1)th bit.
  - ✓ The MSB (Most Significant Bit) of the IR scan path is the JTDI input value.
  - When the state jumps to Update-IR, the cached value of the IR scan path is updated to the IR.
- DR Scan Path:
  - When the state jumps to Capture-DR, the DR value corresponding to the current IR is transmitted to the scan path, which is connected between JTDI and JTDO.
  - When the state jumps to Shift-DR, the DR scan path shifts one bit on each rising edge of JTCK. At this time:
  - ✓ The JTDO output value is the LSB of the DR scan path.
  - ✓ The nth bit of the DR scan path shifts to the (n-1)th bit.
  - ✓ The MSB of the DR scan path is the JTDI input value.
  - When the state jumps to Update-DR, the cached value of the DR scan path is updated to the selected DR.

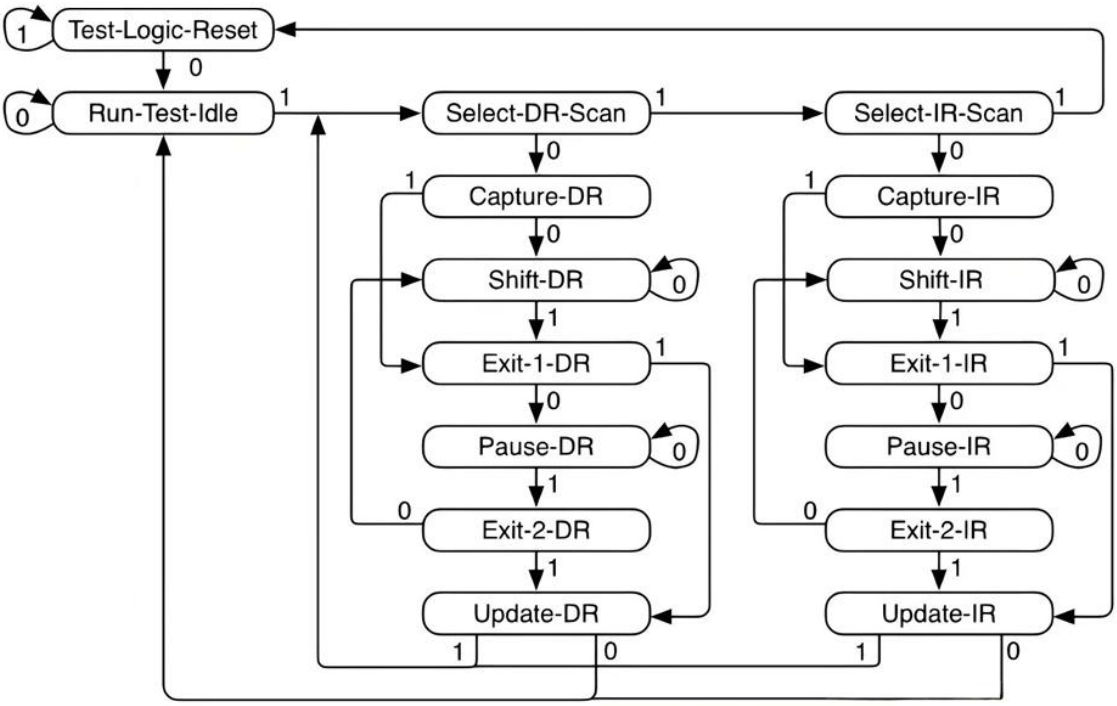


Figure 42-2 State Machine of the TAP Controller

The 2-wire cJTAG supports OScan0 and OScan1 (default) modes. The Scan Packet of TMSC (i.e., TMS under 4-wire JTAG) is shown in the following figure.

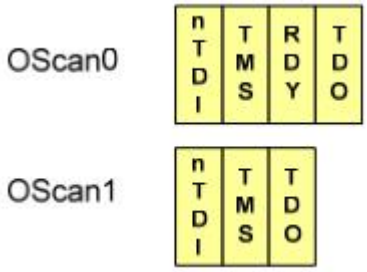


Figure 42-3 Scan Packet of TMSC (i.e., TMS under 4-wire JTAG)

Among them, nTDI is the inverse logic of TDI in 4-wire JTAG (i.e., nTDI = ~TDI), TMS is the TMS of 4-wire JTAG, and TDO is the TDO of 4-wire JTAG. The following figures show the transmission timing diagrams of OScan0 and OScan1 respectively.

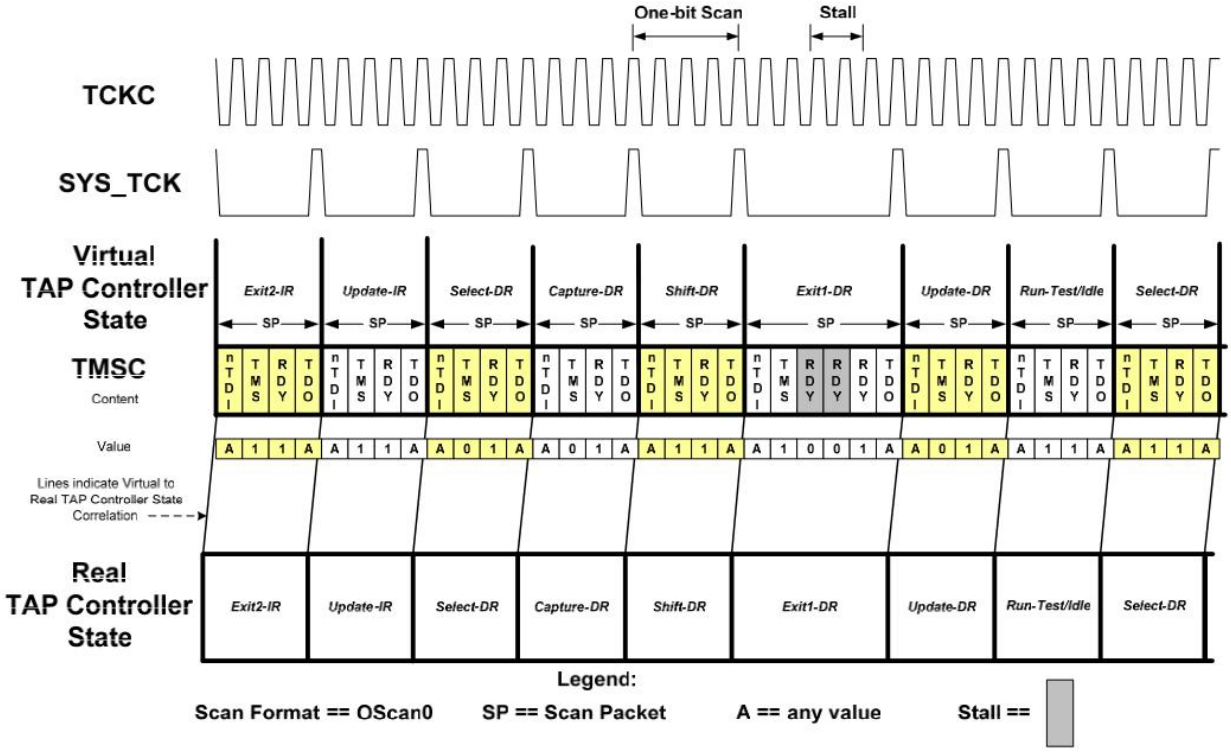


Figure 42-4 Transmission Timing Diagram of OScan0

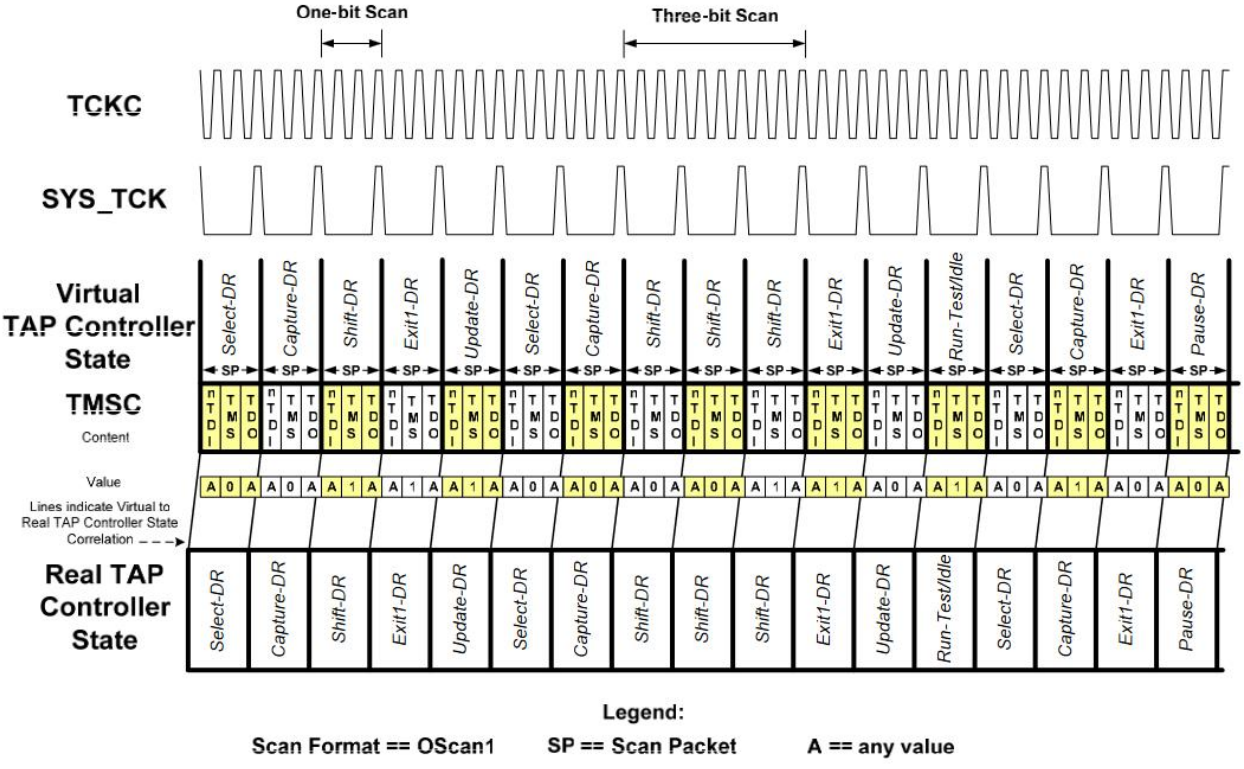


Figure 42-5 Transmission Timing Diagram of OScan1

42.3 Debug Architecture

The debug architecture complies with the RISC-V standard RISC-V Debug Specification v0.13 and supports dual-core debugging through a JTAG-CHAIN structure. The JTAG TAPs of Core0 and Core1 are connected in series: Core1 is at the front end of the chain, and Core0 is at the back end.

The Instruction Register (IR) of the Core0/Core1 JTAG chain has a width of 5 bits, and the Data Register (DR) has a width of 41 bits. It supports bypassing one of the cores using the BYPASS instruction.

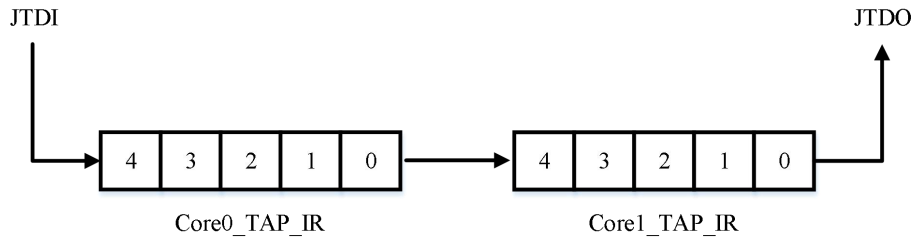


Figure 42-6 Instruction Register (IR) of the Core0/Core1 JTAG Chain

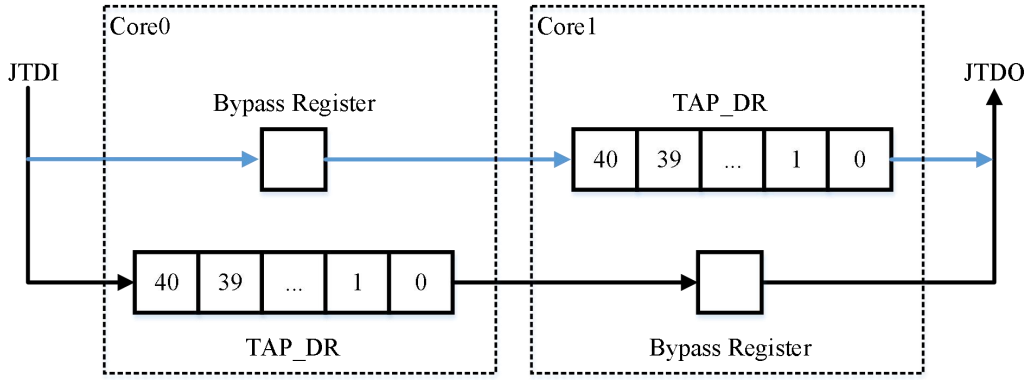


Figure 42-7 Data Register (DR) of the Core0/Core1 JTAG Chain

42.3.1 Instruction Register

Both Core0 and Core1 have an instruction register with a width of 5 bits, and the reset value is 0x1 (IDCODE). Different IR values correspond to different Data Registers (DR), including BYPASS, IDCODE, DTMCS, DMI, etc. The following figure lists the main commands supported by the instruction register.

Address	Name	Description	Page
0x00	BYPASS	JTAG recommends this encoding	
0x01	IDCODE	JTAG recommends this encoding	
0x10	DTM Control and Status (`dtmcs`)	For Debugging	64
0x11	Debug Module Interface Access (`dmi`)	For Debugging	65
0x12	Reserved (BYPASS)	Reserved for future RISC-V debugging	
0x13	Reserved (BYPASS)	Reserved for future RISC-V debugging	
0x14	Reserved (BYPASS)	Reserved for future RISC-V debugging	
0x15	Reserved (BYPASS)	Reserved for future RISC-V standards	
0x16	Reserved (BYPASS)	Reserved for future RISC-V standards	
0x17	Reserved (BYPASS)	Reserved for future RISC-V standards	
0x1f	BYPASS	JTAG requires this encoding	

Figure 42-8 Main Commands Supported by the Instruction Register

42.3.2 Data Register

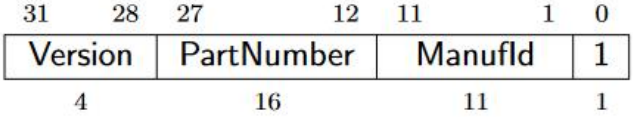
Both Core0 and Core1 have an instruction register with a width of 41 bits. The width of the Data Register (DR) corresponding to different IR values varies.

42.3.2.1 BYPASS (IR=0x1f …)

When the debugger does not need to communicate with this TAP, set the IR value to BYPASS. This register is read-only.

42.3.2.2 IDCODE（IR=0x01）

A read-only register that complies with the IEEE 1149.1-2013 standard.

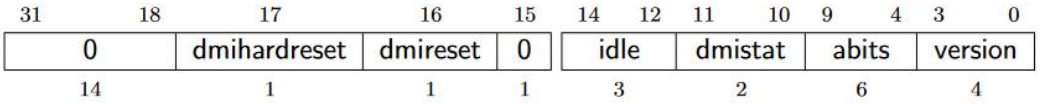


Field	Description	Access	Reset
Version	Identifies the release version of this part.	R	Preset

PartNumber	Identifies the designer's part number of this part.	R	Preset
ManufId	Identifies the designer/manufacturer of this part. Bits 6:0 must be bits 6:0 of the designer/manufacturer's Identification Code as assigned by JEDEC Standard JEP106. Bits 10:7 contain the modulo-16 count of the number of continuation characters (0x7f) in that same Identification Code.	R	Preset

42.3.2.3 DTM Control and Status（dtmcs，IR=0x10）

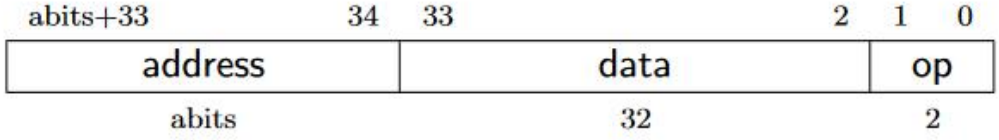
The upper two bits are writable and used to terminate/retry transmission; the remaining bits are readable and used to confirm the completion status of the previous operation. The initial value is 0x0_7071.



Field	Description	Access	Reset
dmihardreset	Writing 1 to this bit does a hard reset of the DTM, causing the DTM to forget about any outstanding DMI transactions. In general this should only be used when the Debugger has reason to expect that the outstanding DMI transaction will never complete (e.g. a reset condition caused an inflight DMI transaction to be cancelled).	W1	-
dmireset	Writing 1 to this bit clears the sticky error state and allows the DTM to retry or complete the previous transaction.	W1	-
idle	This is a hint to the debugger of the minimum number of cycles a debugger should spend in Run-Test/Idle after every DMI scan to avoid a 'busy' return code (dmistat of 3). A debugger must still check dmistat when necessary. 0: It is not necessary to enter Run-Test/Idle at all. 1: Enter Run-Test/Idle and leave it immediately. 2: Enter Run-Test/Idle and stay there for 1 cycle before leaving. And so on.	R	Preset
dmistat	0: No error. 1: Reserved. Interpret the same as 2. 2: An operation failed (resulted in op of 2). 3: An operation was attempted while a DMI access was still in progress (resulted in op of 3).	R	0
abits	The size of address in dmi.	R	Preset
version	0: Version described in spec version 0.11. 1: Version described in spec version 0.13. 15: Version not described in any available version of this spec.	R	1

42.3.2.4 Debug Module Interface Access（dmi，IR=0x11）

This register is used to access the Debug Module Interface (DMI). After the TAP state machine jumps to Capture-DR, the DTM uses the operation result to update the data. After the TAP state machine jumps to Update-DR, this DR register is updated and a corresponding DMI access request is initiated—unless it is in a sticky state (where the previous operation is incomplete). The address width is 7 bits.



Field	Description	Access	Reset
address	Address used for DMI access. In Update-DR this value is used to access the DM over the DMI.	R/W	0
data	The data to send to the DM over the DMI during Update-DR, and the data returned from the DM as a result of the previous operation.	R/W	0
op	When the debugger writes this field, it has the following meaning: 0: Ignore data and address. (nop) Don't send anything over the DMI during Update-DR. This operation should never result in a busy or error response. The address and data reported in the following Capture-DR are undefined. 1: Read from address. (read) 2: Write data to address. (write) 3: Reserved. When the debugger reads this field, it means the following: 0: The previous operation completed successfully. 1: Reserved. 2: A previous operation failed. The data scanned into dmi in this access will be ignored. This status is sticky and can be cleared by writing dmireset in dtmcs. This indicates that the DM itself responded with an error. There are no specified cases in which the DM would respond with an error, and DMI is not required to support returning errors. 3: An operation was attempted while a DMI request is still in progress. The data scanned into dmi in this access will be ignored. This status is sticky and can be cleared by writing dmireset in dtmcs. If a debugger sees this status, it needs to give the target more TCK edges between Update-DR and Capture-DR. The simplest way to do that is to add extra transitions in Run-Test/Idle.	R/W	0

43 Bootloader

43.1 Overview

After reset, the chip executes the bootloader program from the ROM. After completing system initialization and NVR configuration loading, the bootloader executes different processes based on the ISP state: the ISP process or the FLASH boot process.

ISP Process: After initializing the ISP communication port according to the NVR configuration, it waits for baud rate synchronization. Upon successful baud rate synchronization, it enters the ISP task processing flow (waiting to execute ISP tasks issued by the user). During the ISP process, the ISP state is continuously detected. If an exit from the ISP state is detected, the chip resets and re-executes the bootloader program.

FLASH Boot Process: It determines whether to verify the signature of the N200 firmware code based on the NVR configuration. If the N200 firmware is detected as valid, it modifies the system REMAP register and jumps to the FLASH to execute the N200 firmware code.

In addition, the bootloader provides a series of IAP functions for application program calls.

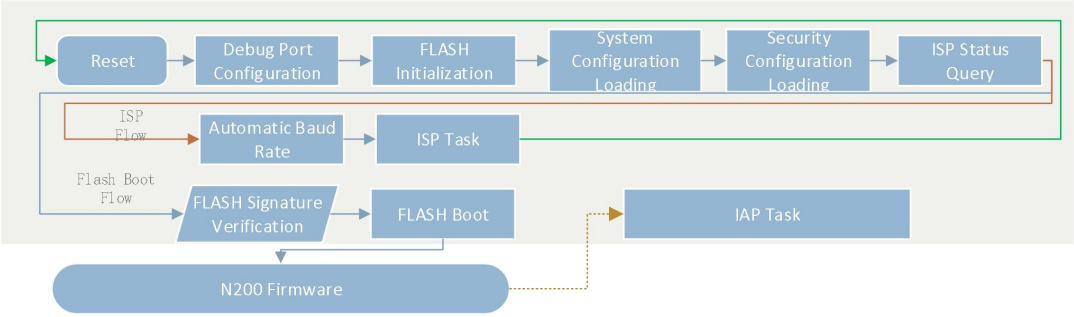


Figure 43-1 Bootloader Flow Chart

43.2 NVR Configuration

NVR (Non-Volatile Register) refers to a special storage space in the FLASH memory, used to store system configurations and application program configurations. Operations on the FLASH memory (such as erasing or programming the FLASH) do not affect the NVR. The NVR can only be modified through special IAP interface functions or programmers.

The NVR area is managed in pages (basic unit), with a page size of 512 bytes:

- NVR0/1: Not open for use.
- NVR1: Used to store key parameters of the chip's factory configuration, programmed before the chip leaves the factory.
- NVR2: Used to store application program configuration parameters. It is recommended to program it via a programmer during the whole-machine production stage or modify it via the IAP interface in the application program.
- NVR3/NVR4: Used to store the signature information of the N200 kernel firmware, modified by the application program via the IAP interface during remote upgrades.
- NVR5: Used to store security configuration information. If the application program has security-related requirements, NVR5 must be configured correctly. It is recommended to program it via a programmer during the whole-machine production stage.

The validity of NVR data is checked when it is used. The validity of NVR data consists of two parts: a data validity flag (TAG) and a data checksum (CKSUM). Only when both parts are valid can the NVR data be used. The valid value

of TAG is 0x12345678. The CKSUM is calculated as 0 - sum(0~0x1f8), where "sum" means accumulating and summing in word units.

The NVR is only accessible by the N200 kernel, with a base address of <FLASH Base Address> + 1MB. For example, in REMAP0 mode, the NVR base address = 0x0020_0000 + 0x0010_0000 = 0x0030_0000. Each NVR is stored and managed in 16-bit units. For detailed definitions, refer to the following table.

Offset Address	NVR0/1	NVR2	NVR3/4	NVR5
0x00	Not Open	ISP_CP_L	VERSION_L	CFG0_L
0x02		ISP_CP_H	VERSION_H	CFG0_H
0x04		ISP_PASS_L	SIGN_LEN_L	CFG1_L
0x06		ISP_PASS_H	SIGN_LEN_H	CFG1_H
0x08		DOTA0	SIGN0	CFG2_L
0x0a		ALPHAL	SIGN1	CFG2_H
0x0c		ALPHAH	SIGN2-SIGN30	GPIOA_CFG
0x0e		XT0		GPIOB_CFG
0x10		WDT0_CTRL_L		GPIOC_CFG
0x12		WDT0_CTRL_H		GPIOD_CFG
0x14		WDT0_HALT_L		GPIOE_CFG
0x16		WDT0_HALT_H		GPIOF_CFG
0x18		WDT0_STBY_L		GPIOG_CFG
0x1a		WDT0_STBY_H		GPIOH_CFG
0x1c		DOTA0_BYPASS		SRAM_CFG
0x1e		ALPHAL_BYPASS		FLASH_CFG
0x20		ALPHAH_BYPASS		reserved
0x22		XT0_BYPASS		SIGN_EN
0x24		ISP_BYPASS		SIGN_MODE
0x26		ISP_UART_SEL		SIGN_PK
0x28		DELAY_SEL		LOCK
0x2a		JTAG_MODE		SIGN_SPEEDUP
0x2c		JTAG0_CTRL		UPDATE_SIGN_EN
0x2e		JTAG1_CTRL		reserved
0x30		SYSCHECK0_L		
0x32		SYSCHECK0_H		
0x34		SYSCHECK1_L		
0x36		SYSCHECK1_H		
0x38		SYSCHECK2_L		
0x3a		SYSCHECK2_H		
0x3c		SYSCHECK3_L		
0x3e		SYSCHECK3_H		
0x40		SYSCHECK4_L		
0x42		SYSCHECK4_H		
0x44		N200APP_NVR_L		
0x46		N200APP_NVR_H	SIGN31	

0x48		WDT1_CTRL_L	N200_REMAP	
0x4a		WDT1_CTRL_H	reserved	
0x4c		WDT1_HALT_L		
0x4e		WDT1_HALT_H		
0x50		WDT1_STBY_L		
0x52		WDT1_STBY_H		
0x54		WDT1_STOP_L		
0x56		WDT1_STOP_H		
0x58		N300_RELEASE		
0x5a		reserved		
0x5c		N300_REMAP		
0x5e		ISP_FREQ		
0x60		SOFT_ISP_EN_L		
0x62		SOFT_ISP_EN_H		
0x64		SOFT_ISP_PROTECT_L		
0x66		SOFT_ISP_PROTECT_H		
0x68		PRE_JTAG_CFG_L		
0x6a		PRE_JTAG_CFG_H		
...		reserved		
0x1f8		TAG_L	TAG_L	TAG_L
0x1fa		TAG_H	TAG_H	TAG_H
0x1fc		CKSUM_L	CKSUM_L	CKSUM_L
0x1fe		CKSUM_H	CKSUM_H	CKSUM_H

43.3 In-System Programming (ISP)

ISP supports using a serial port to perform operations such as programming and erasing the FLASH via the ISP protocol provided by the bootloader, without the need for application program participation. After the bootloader initialization phase is completed, it checks the ISP state. If the ISP state meets the conditions, it jumps to execute the ISP process.

43.3.1 ISP State

Entering the ISP process requires the ISP state to meet specific conditions. The ISP state is divided into hardware ISP pin state and software ISP state:

- Hardware ISP Pin: The hardware ISP pin is PG04. After the bootloader resets, it configures PG04 as a GPIO input and enables the built-in pull-up resistor of the PG04 pin. If the PG04 pin is at a low level, the ISP state condition is met, and the bootloader enters the ISP process.
- Software ISP State: The software ISP state is determined by the SOFT_ISP_EN and SOFT_ISP_PROTECT values in NVR2. If SOFT_ISP_PROTECT is not 0x69789687 and SOFT_ISP_EN is 0xe7a5185a, the ISP state condition is met, and the bootloader enters the ISP process.

The exit conditions differ between entering the bootloader via the hardware ISP pin and via software ISP configuration:

- When entering the ISP process via the hardware ISP pin: The ISP process exits and the bootloader re-executes once PG04 returns to a high level.
- When entering the ISP process via software ISP configuration: The ISP process exits and the bootloader

re-executes when the application program calls a library function to modify SOFT_ISP_EN to a value other than 0xe7a5185a.

43.3.2 Protection Levels

Protection levels are used for permission management of the ISP communication port, supporting 4 different ISP permissions (see the table below). By default, the permission management of the debug port is configured together with the protection level. The debug port permission and protection level are configured via ISP_CP and JTAG_CTRL in NVR2, respectively. It is recommended to configure JTAG_CTRL when configuring ISP_CP.

Protection Level	ISP_CP Value	Definition	Debug Port Permission (JTAG_CTRL)
0	Other Values	No protection for the ISP communication port	Debug Allowed
1	0x12345678	Password required for accessing the ISP communication port	Debug Allowed
2	0x4e647370	The ISP communication port only supports chip erasure; other ISP communication commands are prohibited	Debug Prohibited (needs configuration)
3	0x9d61629e	All ISP communication commands are prohibited	Debug Prohibited (needs configuration)

For chip erasure at Protection Level 2: In addition to erasing the user application program in the FLASH storage area, it also clears the ISP_CP and JTAG_CTRL values in NVR2—restoring the protection level from 2 to 0 and enabling debugging.

43.3.3 ISP Process

The physical interface of the ISP protocol is a serial port. After successful baud rate synchronization, communication with the bootloader is conducted via the ISP protocol. The serial port for ISP supports configuring different UART pins via ISP_UART_SEL in NVR2.

ISP_UART_SEL	UART_ID	TX_PORT	TX_PIN	RX_PORT	RX_PIN
0	**2**	**A**	**1**	**A**	**0**
1	**6**	**A**	**4**	**A**	**3**
2	**4**	**B**	**0**	**A**	**15**
3	**4**	**B**	**1**	**B**	**2**
4	**6**	**B**	**3**	**B**	**4**
5	**4**	**B**	**6**	**B**	**5**
6	**6**	**B**	**7**	**B**	**13**
7	**6**	**C**	**6**	**C**	**14**
8	**4**	**D**	**9**	**D**	**8**
9	**0**	**D**	**14**	**D**	**12**
10	**0**	**E**	**2**	**E**	**1**
11	**0**	**E**	**5**	**E**	**4**
12	**2**	**E**	**14**	**E**	**15**
13	**2**	**F**	**7**	**G**	**3**

14	**4**	**G**	**8**	**G**	**7**
15	**0**	**G**	**13**	**G**	**12**
16	**6**	**H**	**1**	**H**	**0**

43.3.3.1 Baud Rate Synchronization

Baud rate synchronization uses the synchronization character "e" by convention. The synchronization process is as follows:

- The upper computer sends the character "e" (note: a single character, no newline character added).
- The bootloader automatically detects the baud rate of the synchronization character. If detection is successful, it returns the response string "Synchronized" (note: no newline character at the end); if detection fails, it continues to wait for the synchronization character.
- After receiving the response string, the upper computer sends back the response string with a newline character, i.e., "Synchronized\n".
- If the bootloader correctly receives the response string, it returns the current operating frequency (e.g., "1843200\n") and ends baud rate synchronization; if an error is detected, it returns "ES" and continues to wait for the response string.

Note: The default operating frequency of the bootloader is 1843200 Hz, and the maximum supported baud rate is 115200 bps. Do not use a baud rate exceeding 115200 bps for communication.

43.3.3.2 ISP Commands

After completing baud rate synchronization, the ISP process enters the ISP command transmission/reception state. At this point, if the ISP pin changes from low to high, a system reset will occur, and the bootloader will re-execute to boot the FLASH. ISP commands uniformly use ASCII codes, consisting of <Command> + <Parameters> + <Newline Character>. The following table lists some commonly used ISP commands; for a complete list of ISP commands, refer to the IAP Function Interface chapter.

Command	Usage
Set Baud Rate	BS <Baud Rate> <Stop Bit>
Echo	RD <Switch Setting>
Write Memory	WM <Address> <Byte Length> <Mode>
Read Memory (including FLASH, SRAM)	RM <Address> <Byte Length> <Mode>
FLASH Page Erase	FP <Page Address>
FLASH Chip Erase	FC
FLASH Programming	FW <FLASH Address> <RAM Address> <Byte Length>
Memory Compare	MC <Address1> <Address2> <Byte Length>
Run	GO <Address>
Unlock	UN <Password>
Get Protection Level	AL
Software Reset	RS
Enable NVM (FLASH)	NV <NVM Option>

Note: Numbers in parameters are in decimal format.

43.3.3.3 ISP Data Transmission

The ISP command communication protocol uses ASCII codes. For some ISP commands that support subsequent data transmission, the protocol switches to raw data (hex format) during data transmission to improve communication efficiency. Raw data transmission follows a fixed protocol format:

- Data Line: Raw data is transmitted in "lines" as the basic unit. The line format is <0x7e> + <num> + <data> + <0x7e>, where:
  - The first and last characters must be 0x7e.
  - "num" represents the byte length of the currently transmitted data.
  - A maximum of 64 bytes of raw data can be transmitted per line.
  - Escape rules: If the data in the line is 0x7e, it is escaped to 0x7d, 0x5e for transmission; if the data is 0x7d, it is escaped to 0x7d, 0x5d (to ensure no 0x7e is contained in the UART-transmitted data).
  - Note: "num" refers to the length of the raw data, not the escaped length.
- Block and Checksum Line: 32 lines form a "block"; if the data is less than 32 lines, the actual number of lines is used. After transmitting a block of data, a checksum line must be sent. The checksum line format is fixed as <0x7e> + <0xff> + <checksum> + <0x7e>, where the checksum is the two's complement of the sum of all data in the current block (calculated in word units). The checksum also follows the data escape rules.

Data Line Format (B represents transmitted data bytes, in hexadecimal):

Line Start	1	2	3	4	5	6	……	64	65	Line End
0x7e	Num	B0	B1	B2	B3	B4	……	B62	B63	0x7e

Checksum Line Format (ASCII code; S represents the accumulated checksum SUM):

Line Start	1	2	3	4	5	Line End
0x7e	0xff	S0	S1	S2	S3	0x7e

43.4 In-Application Programming (IAP)

IAP provides interfaces that can be called by application programs, supporting erasing and programming operations on the FLASH. The IAP interface adopts a unique interface function, and different interface functions are implemented through parameters.

The function pointer address of the IAP interface is stored at the offset address 0x20 of the ROM. The IAP interface has two parameters: the first parameter is the input parameter pointer of the IAP interface, which stores the IAP interface command code value, the first parameter, the second parameter, etc., in word units; the second parameter is the return value pointer of the IAP interface, which stores the command execution status return of the IAP interface, result 1, result 2, etc., in word units.

The IAP interface is called in the application program and uses the stack space of the application program. The IAP interface uses a maximum of 1KB of stack space, and the application program needs to reserve sufficient stack space.

43.4.1 IAP Return Values

The return values of the IAP interface are shown in the following table:

IAP Return Value	Return	Return Value Description

	Value Code	
IAP_CMD_SUCCESS	0x0	Success
IAP_INVALID_COMMAND	0x1	Invalid Command
IAP_INVALID_PARAM	0x2	Invalid Parameter
IAP_INVALID_BAUD_RATE	0x3	Invalid Baud Rate
IAP_INVALID_STOP_BIT	0x4	Invalid Stop Bit
IAP_ADDR_NOT_ALIGN	0x5	Address Misalignment
IAP_COUNT_ERROR	0x6	Length Error
IAP_ADDR_NOT_MAPPED	0x7	Address Error
IAP_INVALID_PAGES	0x8	FLASH Page Error
IAP_SECTOR_NOT_BLANK	0x9	FLASH Not Erased
IAP_SRC_ADDR_NOT_ALIGN	0xA	Source Address Misalignment
IAP_SRC_ADDR_NOT_MAPPED	0xB	Source Address Error
IAP_DST_ADDR_NOT_ALIGN	0xC	Destination Address Misalignment
IAP_DST_ADDR_NOT_MAPPED	0xD	Destination Address Error
IAP_COMPARE_ERROR	0xE	Data Comparison Error
IAP_LOWPWR_ERROR	0x14	FLASH Erase/Program Error in Low Power Mode

43.4.2 IAP Commands

The following table lists all currently supported IAP commands. For the description of the IAP interface related to encryption and decryption functions, refer to the encryption and decryption library function description document released by MAKSA, which is not expanded here.

IAP Command Name	Command Code Value	Parameters	IAP Function Description
IAP_GET_VERSION	0	Parameter 0: Pointer to store the returned version number, requires 32-bit alignment. Return value format: 0x00<<24 | x<<16 | y<<8 | z<<0, e.g., 0x00010100	Get Version Number
IAP_FLASH_UNLOCK	1	Parameter 0: FLASH type to operate, 0 indicates N200 FLASH, 1 indicates N300 FLASH, 2 indicates NVR Parameter 1: Target address for operation Parameter 2: Operation byte length, for NVR, the length must be page length	Unlock FLASH
IAP_FLASH_ERASE	2	Parameter 0: FLASH type to operate Parameter 1: Target address for operation Parameter 2: Operation byte length	Erase FLASH
IAP_FLASH_PROGRAM	3	Parameter 0: FLASH type to operate Parameter 1: Target address for operation Parameter 2: Source address of data to be	Program FLASH

		programmed Parameter 3: Operation byte length	
IAP_MODE_CHG	4	Parameter 0: Target mode to switch to Clock_Losc = 0x00 Clock_RCL_32K = 0x01 Clock_RC_1M8 = 0x02 Clock_RC_59M = 0x03 Clock_PLL_7M4 = 0x04 Clock_PLL_14M7 = 0x05 Clock_PLL_29M5 = 0x06 Clock_PLL_59M = 0x07 Clock_HOSC_7M4 = 0x08 Clock_HOSC_14M7 = 0x09 Clock_HOSC_29M5 = 0x0a Parameter 1: Clock source division value for the target mode to switch to Clock_Div1 = 0 Clock_Div2 = 1 Clock_Div4 = 2 Clock_Div8 = 3 Clock_Div16 = 4 Clock_Div32 = 5 Parameter 2: Clock division value for N300 and N200 in the target mode to switch to (Clock_N300_N200) Clock_Div1_Div1 = 0x00 Clock_Div1_Div2 = 0x01 Clock_Div1_Div4 = 0x02 Clock_Div1_Div8 = 0x03 Clock_Div2_Div2 = 0x05 Clock_Div2_Div4 = 0x06 Clock_Div2_Div8 = 0x07 Clock_Div4_Div4 = 0x0a Clock_Div4_Div8 = 0x0b Clock_Div8_Div8 = 0x0f	Switch System Clock Mode
IAP_FAST_MODE_CHG	5	Parameter 0: Target mode to switch to, only supports fast switching between LOSC and RC1.8M modes Clock_Losc = 0x00 Clock_RC_1M8 = 0x02	Fast Clock Mode Switching

IAP_PLL_CTL	7	Parameter 0: Select PLL clock source, supports: Clock_PLL_7M4 = 0x04 Clock_PLL_14M7 = 0x05 Clock_PLL_29M5 = 0x06 Clock_PLL_59M = 0x07 Parameter 1: PLL clock switch, 0 indicates on, 1 indicates off	PLL Clock Control
IAP_SYSCFG_CHK	8	None	System Key Register Check
IAP_KS_RD_KEY	9	Parameter 0: Pointer to store the returned key value Parameter 1: Key length Parameter 2: Key slot index number	Read KS Key
IAP_KS_RD_KEYINFO	10	Parameter 0: Pointer to the structure storing returned key slot information Parameter 1: Key slot index number	Read KS Key Information
IAP_KS_WR_KEY	11	Parameter 0: Pointer to the structure storing key information Parameter 1: Pointer storing the key Parameter 2: Key slot index number	Write KS Key
IAP_KS_ERASE_ALL	13	Parameter 0: Password	Erase All Keys in KS Area
IAP_TRNG_GEN_KEY	15	Parameter 0: Pointer to store the returned random number, 32-bit aligned Parameter 1: Random number length, in word units	Generate Random Number
IAP_RSA_GEN_KEY	16	Refer to the library function documentation provided by MAKSA	
IAP_RSA_GEN_SIGN	17		
IAP_RSA_VER_SIGN	18		
IAP_RSA_PUB_ENCRYPT	19		
IAP_RSA_PUB_DECRYPT	20		
IAP_RSA_PVT_ENCRYPT	21		
IAP_RSA_PVT_DECRYPT	22		
IAP_HASH_INIT	23		
IAP_HASH_DEINIT	24		
IAP_HASH	25		
IAP_HASH_LATCH	26		
IAP_HASH_RECOVER	27		
IAP_ECC_GEN_SIGN	28		
IAP_ECC_GEN_SIGN_WITH_PKEY	29		
IAP_ECC_VER_SIGN	30		

IAP_ECC_GEN_DHKEY	31	
IAP_ECC_GEN_KEY	32	
IAP_ECC_GEN_PUBKEY	33	
IAP_ECC_VER_KEY	34	
IAP_ECC_GEN_KEY_WITH_SEC	35	
IAP_ECC_GEN_SIGN_WITH_SEC	36	
IAP_ECC_VER_SIGN_WITH_SEC	37	
IAP_AES_GF128_MULT	38	
IAP_AES_CRYPT_INIT	39	
IAP_AES_CRYPT_DEINIT	40	
IAP_AES_SIGN_CRYPT	41	
IAP_AES_VER_DECRYPT	42	
IAP_AES_GMAC	43	
IAP_AES_LATCH	44	
IAP_AES_RECOVER	45	
IAP_AES_ONCE	46	
IAP_AES_key_wrap	47	
IAP_AES_key_unwrap	48	
IAP_AES_update_aad	49	

43.4.2.1 Clock Switching Related Library Functions

Different clock modes support different frequency settings and division coefficients at the clock source end. Among them:

- RCL/LOSC does not support source division, and the frequency is fixed.
- RC supports 1/2/4/8/16/32 division, and the frequency is fixed (59M).
- PLL supports 1/2/4/8/16 division, and the frequency supports 7/14/29/59M.
- HOSC supports 1/2/4/8/16 division, and the frequency supports 7/14/29M.

System Clock Mode Design Requirements:

- The working modes of N200 and N300 must be from the same source.
- The frequency of N200 cannot be greater than that of N300.
- N200 must be ≤29M.
- The software ensures that the frequency of N200 and N300 after division is 32K/1M/3M/7M/14M/29M/59M.

The clock mode switching library function HAL_SYSTEM_ClockModeConfig adopts a unified library function interface. The difference from other products is the addition of a parameter for source division.

HAL_StatusTypeDef HAL_SYSTEM_ClockModeConfig(HAL_SYSTEM_ClockModeTypeDef ClockMode, HAL_SYSTEM_SourceDivTypeDef SourceDivison, HAL_SYSTEM_CoreDivTypeDef CoreDivison);

To support the default clock mode after reset, in addition to the RC59M mode, a new RC1M mode is added.

In addition to the unified clock switching library function interface, a fast clock mode switching library function HAL_SYSTEM_FastClockModeConfig is also provided, which supports fast switching between RC1M and LOSC.

Note: Fast clock switching does not operate the PLL/HOSC clock configuration, only the LDO mode configuration.

HAL_StatusTypeDef HAL_SYSTEM_FastClockModeConfig(HAL_SYSTEM_ClockModeTypeDef ClockMode)

Since the second pulse is generated by the PLL, the PLL is enabled by default in all modes except 32K and RC1.8M modes. The library provides a dedicated PLL configuration function HAL_SYSTEM_PLLConfig, which is used to enable/disable the PLL and set the PLL frequency. If the current system operating mode is PLL, this function will not execute and will return an error.

HAL_StatusTypeDef HAL_SYSTEM_PLLConfig(HAL_SYSTEM_ClockModeTypeDef PLLFrequency, HAL_SYSTEM_PLLSwitchTypeDef PLLState)

The following table lists the supported configurations for different clock modes and the status of each clock source after switching to the corresponding mode:

Mode	Clock Source Division	System Clock Division	RC	PLL	HOSC	Description
LOSC	-	-	off	off	off	
RCL	-	-	off	off	off	
RC1M8	-(Internally fixed at 32)	-	on	off	off	
RC59M	1/2/4/8/16/32	1/2/4/8	on	on	off	1. The software evaluates the validity of frequency division coefficients to ensure no unsupported frequencies appear. 2. RC mode is enabled to support dynamic clock measurement by user code. 3. PLL mode is enabled to ensure accurate generation of the second pulse.
PLL7M4	1/2/4/8/16	1/2/4/8	on	on	off	
PLL14M7	1/2/4/8/16	1/2/4/8	on	on	off	
PLL29M5	1/2/4/8/16	1/2/4/8	on	on	off	
PLL59M	1/2/4/8/16	1/2/4/8	on	on	off	
HOSC7M	1/2/4/8/16	1/2/4/8	on	on	on	
HOSC14M	1/2/4/8/16	1/2/4/8	on	on	on	
HOSC29M	1/2/4/8/16	1/2/4/8	on	on	on	

43.4.2.2 FLASH Operation Related Library Functions

The FLASH operation library functions provide interfaces for erasing and programming the FLASH. Due to security design requirements, FLASH operations for both N200 and N300 can only be performed via N200. The FLASH is divided into three regions: FLASH0 allocated to N200, FLASH1 allocated to N300, and the NVR region.

All target address parameters for FLASH interfaces must be page-aligned.

An unlock operation must be performed before each call to the FLASH erase/program interface. This operation enables the FLASH partition protection register based on the input FLASH region, starting address, and length parameters. The unlock operation uses single-execution mode, meaning the command register automatically disables operations after executing a single command.

HAL_FLASH_StatusTypeDef HAL_FLASH_Unlock(HAL_FLASH_IDTypeDef FlashID, uint32_t Address, uint32_t ByteLength)

FLASH Erase Interface: Input parameters are the FLASH region, target start address, and target length. The FLASH region supports N200 FLASH, N300 FLASH, and NVR. The target address must be a valid logical address

within the corresponding region and be page-aligned. The target length must be within the valid FLASH region.

HAL_FLASH_StatusTypeDef HAL_FLASH_Erase(HAL_FLASH_IDTypeDef FlashID, uint32_t Address, uint32_t ByteLength)

FLASH Programming Interface: Input parameters are the FLASH region, target start address, data source address, and target length. The FLASH region supports N200 FLASH, N300 FLASH, and NVR. The target start address must be a valid logical address within the corresponding region and must be page-aligned. The data source address is the start address of the data to be programmed and does not require word alignment. The target length must be within the valid FLASH region.

HAL_FLASH_StatusTypeDef HAL_FLASH_Program(HAL_FLASH_IDTypeDef FlashID, uint32_t address, uint32_t data, uint32_t byteLength)

43.5 FLASH Boot

When the ISP state is not satisfied, the bootloader attempts to boot the FLASH. Before booting the FLASH, the bootloader verifies the validity of the data in the FLASH. Data validity includes vector table checksum verification and optional firmware signature verification. If the FLASH data validity check fails, the bootloader will not boot the FLASH and will reset the chip after a delay of approximately 1 second.

43.5.1 Vector Table Checksum Verification

The first step in FLASH data validity verification is to check the vector table checksum, i.e., word6 = 0 - sum(word0~word5, word7). Here, word6 corresponds to the FLASH offset address 0x18, and "sum" means accumulating and summing in word units.

43.5.2 FLASH Signature Verification

FLASH signature verification supports two methods (ECDSA and HASH) for verifying the signature of the N200 firmware:

- ECDSA: Supports the NIST-P256 curve, with a public key length of 64 bytes and a signature length of 64 bytes.
- HASH: Supports SHA256, with a signature length of 32 bytes.

The public key of the firmware is managed through KS, and the signature information is stored in NVR3/4. The configuration information for signature verification is stored in NVR5:

- SIGN_EN: If SIGN_EN = 0xabcd, the Core0 firmware signature verification function is enabled during boot; other values mean no signature verification is performed on the firmware.
- SIGN_MODE (Upper 8 bits): Defines the signature storage location. If the value is 0xfe, the signature is stored in NVR3/4; if 0xef, the signature is stored at the end of the firmware (corresponding to <FLASH Start Address> + <SIGN_LEN> address); other values are not supported, and no signature verification is performed.
- SIGN_MODE (Lower 8 bits): Defines the signature type. If 0x0, SHA256 is used; if 0x1, ECDSA is used; other values are not supported, and no signature verification is performed.

The signature information stored in NVR3/4 includes:

- Software Version Number: Defined as {VERSION_H, VERSION_L}, a 32-bit little-endian unsigned integer. A larger value indicates a higher version number. Users can define the version number as long as this constraint is met (no specific format required).
- SIGN_LEN_L/H: Represents the firmware length in bytes. The default start address of the firmware is the FLASH start address. If SIGN_LEN_L/H = 0 or exceeds the FLASH allocation size of Core0, a signature verification error is triggered.

The signature verification time is related to the verification length. To avoid excessively long bootloader startup time due to signature verification, the bootloader supports a signature verification acceleration configuration. When signature verification acceleration is enabled (SIGN_SPEEDUP = 0x7e01), the bootloader switches to RC29M mode during the signature verification phase and reverts to RC1M8 mode after verification is completed.

43.6 Bootloader Resource Usage

After reset, the chip first executes the bootloader program, which consumes execution time and RAM resources. The default bootloader execution time is approximately 119ms. Following an upgrade reset, since the configuration loading process is unnecessary, the reset time is approximately 109ms. The bootloader defaults to a 100ms software delay to allow the power system to achieve more reliable stabilization. Disabling this 100ms delay reduces bootloader execution to approximately 19ms, and post-update reset execution to about 9ms.

The bootloader utilizes the [13-16KB] range of SRAM (i.e., SRAM offset addresses 0x3400-0x4000) for its runtime stack and global variables. Applications should avoid using this region as a noinit segment.

44 Package Dimensions and Soldering Conditions

44.1 Package Dimensions

44.1.1 LQFP100L

LQFP100L (1414×1.4)		14.00×14.00×1.40	e=0.50

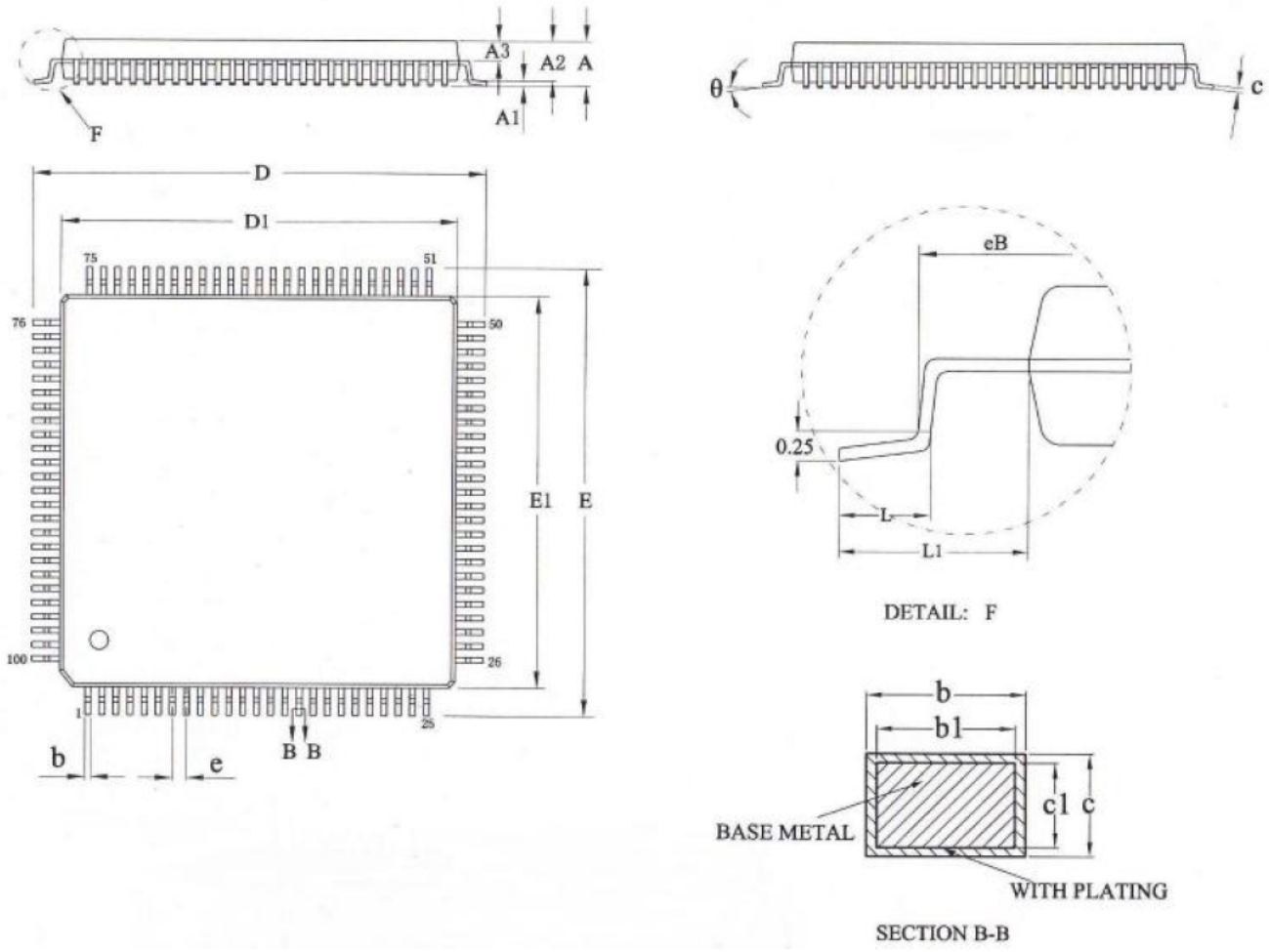


Figure 44-2 LQFP100L Package Dimension Diagram

SYMBOL	MILLIMETER		
	MIN	NOM	MAX
A	---	---	1.6
A1	0.05	---	0.15
A2	1.35	1.40	1.45
A3	0.59	0.64	0.69
b	0.18	---	0.26
b1	0.17	0.20	0.23

c	0.13	---	0.17
c1	0.12	0.13	0.14
D	15.80	16.00	16.20
D1	13.90	14.00	14.10
E	15.80	16.00	16.20
E1	13.90	14.00	14.10
eB	15.05	---	15.35
e	0.50BSC		
L	0.45	---	0.75
L1	1.00BSC		
θ	0	-----	7º

θ	0	-----	7º

44.2 Reflow Oven Temperature Profile Settings

Distribution	Characteristic Value
Soak Temperature (TL) Peak Temperature (Tp) Average Ramp Rate (TL to Tp)	217°C 260°C Max. 3°C/sec
Preheat Minimum Temperature (Ts min) Maximum Temperature (Ts max) Time (Min-Max) (ts)	 150°C 200°C 60-180 sec
Ramp Rate (Ts max to TL)	Max. 3°C/sec