

**ASR6601**

**Reference manual**

**Version 1.5.0**



**Issue date 2022-08-11**

**Copyright © 2022 ASR**

**About this document**

This document provides detailed and complete information of IoT LPWAN SoC-ASR6601 for application developers.

**Intended Readers**

This document is mainly for engineers who use this chip to develop their own platform and

products, for instance:

* Hardware development engineer
* Software engineer
* Technical support engineer

**Included Chip Models**

The product models corresponding to this document are as follows.

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **Model** | **Flash** | **SRAM** | **Processor** | **Package** | **Frequency** |
| ASR6601SE | 256 KB | 64 KB | 32-bit 48 MHz Arm China STAR-MC1 | QFN68,  8\*8 mm | 150 ~ 960 MHz |
| ASR6601CB | 128 KB | 16 KB | 32-bit 48 MHz Arm China STAR-MC1 | QFN48,  6\*6 mm | 150 ~ 960 MHz |

**Copyright Notice**

Copyright © 2022 ASR Microelectronics Co., Ltd. All rights reserved. No part or all of this document may be reproduced, transmitted, transcribed, stored or translated in any form or by any means without the written permission of ASR Microelectronics Co., Ltd.

**Trademark Statement**

 ASR and other ASR logos are trademarks of ASR Microelectronics Co., Ltd.  
Other trade names, trademarks and registered trademarks mentioned in this document are the property of their respective owners and are hereby declared.

**Disclaimer**

ASR does not give any warranty of any kind and may make improvements and/or changes in this

document or in the product described in this document at any time.

This document is only used as a guide, and no contents in the document constitute any form of

warranty. Information in this document is subject to change without notice.

All liability, including liability for infringement of any proprietary rights caused by using the

information in this document, is disclaimed.

**ASR Microelectronics Co., Ltd**

Address: 9F, Building 10, No. 399 Keyuan Road, Zhangjiang High-tech Park, Pudong New Area, Shanghai, 201203, China

Homepage: <http://www.asrmicro.com/>

**Revision History**

|  |  |  |
| --- | --- | --- |
| **Date** | **Version** | **Release notes** |
| 2021.03 | V1.0.0 | Initial Release. |
| 2021.05 | V1.1.0 | * Updated the overview and Table 6-1 of Chapter 6. * Updated some descriptions in Section 16.3, Section 16.9, and Section 16.14.1. * Corrected the description of LORAC\_SR register in Section 12.4.13. |
| 2021.07 | V1.2.0 | Updated CPU description.. |
| 2022.03 | V1.3.0 | * Added Chapter 21: DMA and Chapter 22: GPTIMER. * Fixed several typos. |
| 2022.05 | V1.4.0 | Modified RCO4M to RCO3.6M due to crystal frequency adjustment. |
| 2022.08 | V1.5.0 | * Updated register bits descriptions in Sections 7.5.3, 8.3.3, 8.3.4, 8.3.7, 8.3.12, and 8.3.13 * Updated Figure 8-1: Clock network diagram. |

# Table of contents

## 1. Overview ............................................................................................................................................... 1 2. ASR6601 Introduction ........................................................................................................................ 2

## 3. ASR6601 functions ............................................................................................................................. 3

3.1 ASR6601 SoC Diagram ................................................................................................................3

3.2 ASR6601 function..................................................................................................................... 4

## 4. Power Management Unit............................................................................................................... 9

4.1 Power Supply pins ....................................................................................................................... 9

4.2 Power Supply Architecture ............................................................................................................. 10

## 5. Access Control..............................................................................................................…................11

5.1 Simple Configuration ....................................................................................................................11

5.1.1 Recoverable Security Configuration...........................................................................................11

5.1.2 Unrecoverable Security Configuration........................................................................................11

5.2 Access Control ...............................................................................................................................…..11

5.2.1 Debug Level Rules ...............................................................................................................…...11

5.2.2 Secure and Non-Secure Operation..............................................................................................12

## 6. Operation Modes...........................................................................................................................….13

6.1 Run .........................................................................................................................…........................ 16

6.1.1 Enter and Exit ...............................................................................................……….................. 16

6.1.2 Wakeup Source.......................................................................................................................…..16

6.2 LpRun ...................................................................................................…….....................................…..16

6.2.1 Enter and Exit ..........................................................................…………............................…....….16

6.2.2 Wakeup Source ...........................................................................…................................….......... 16

6.3 Sleep ...................................................................................................……..................................…...... 17

6.3.1 Enter and Exit .....................................................................…………...................................…...... 17

6.3.2 Wakeup Source.....................................................................................................................…......17

6.4 LpSleep..................................................................................………...............................................…..…17

6.4.1 Enter and Exit ............................................…………...................................................................... 17

6.4.2 Wakeup Source...................…........................................................................................................ 17

6.5 Stop0 .............................................................................................................................................. 18

6.5.1 Enter and Exit .................................................................................................................. 18

6.5.2 Wakeup Source................................................................................................................. 18

6.6 Stop1 .............................................................................................................................................. 18

6.6.1 Enter and Exit .................................................................................................................. 18

6.6.2 Wakeup Source............................................................................................................................. 19

6.7 Stop2 .............................................................................................................................................. 19

6.7.1 Enter and Exit .................................................................................................................. 19

6.7.2 Wakeup Source ............................................................................................................................ 19

6.8 Stop3 .............................................................................................................................................. 20

6.8.1 Enter and Exit .................................................................................................................. 20

6.8.2 Wakeup Source ............................................................................................................................. 20

6.9 Standby........................................................................................................................................... 20

6.9.1 Enter and Exit .................................................................................................................. 20

6.9.2 Wakeup Source ....................................................................................................................... 20

## 7. System configuration........................................................................................................................ 21

7.1 System Architecture............................................................................................................................... 21

7.1.1 Arm China STAR-MC1 Processor ........................................................................................ 22 7.1.2 DMAC0 ............................................................................................................................. 22

7.1.3 DMAC1 ............................................................................................................................... 22

7.1.4 Master ................................................................................................................................. 22

7.2 Memory Mapping..................................................................................................................................... 23

7.2.1 AHB0 SFR .......................................................................................................................... 24

7.2.2 AHB1 SFR .......................................................................................................................... 24

7.2.3 APB0 SFR ........................................................................................................................... 24

7.2.4 APB1 SFR ........................................................................................................................... 25

7.3 SRAM ............................................................................................................................................. 25

7.4 Boot Mode ......................................................................................................................................... 26

7.5 SYSCFG Registers ................................................................................................................ 27

7.5.1 SYSCFG\_CR0 .................................................................................................................... 28

7.5.2 SYSCFG\_CR1 .................................................................................................................... 28

7.5.3 SYSCFG\_CR2 .................................................................................................................... 29

7.5.4 SYSCFG-CR3 ..................................................................................................................... 32

7.5.5 SYSCFG\_CR4 .................................................................................................................... 32

7.5.6 SYSCFG\_CR5 .................................................................................................................... 33

7.5.7 SYSCFG\_CR6 .................................................................................................................... 33

7.5.8 SYSCFG\_CR7 .................................................................................................................... 34

7.5.9 SYSCFG\_CR8 .................................................................................................................... 36

7.5.10 SYSCFG\_CR9 .................................................................................................................. 36

7.5.11 SYSCFG\_CR10 ................................................................................................................ 36

7.6 DMA Request MUX............................................................................................................................ 38

## 8. Reset and Clock Control (RCC)......................................................................................................... 40

8.1 Reset .............................................................................................................................................. 40

8.1.1 External Reset.......................................................................................................................... 40

8.1.2 Power-on Reset........................................................................................................................ 40

8.1.3 System Reset............................................................................................................................ 40

8.1.4 Low-power Reset......................................................................................................................... 40

8.2 Clock ................................................................................................................................................. 41

8.2.1 SYS\_CLK ........................................................................................................... 42

8.2.2 Clocks for the Modules............................................................................................................. 42

8.2.3 Clock-out Capability................................................................................................................... 43

8.3 RCC Registers ...................................................................................................................... 43

8.3.1 RCC\_CR0 ........................................................................................................................... 44

8.3.2 RCC\_CR1 ........................................................................................................................... 46

8.3.3 RCC\_CR2 ........................................................................................................................... 48

8.3.4 RCC\_CGR0 ........................................................................................................................ 50

8.3.5 RCC\_CGR1 ........................................................................................................................ 53

8.3.6 RCC\_CGR2 ........................................................................................................................ 54

8.3.7 RCC\_RST0 ......................................................................................................................... 55

8.3.8 RCC\_RST1 ......................................................................................................................... 58

8.3.9 RCC\_RST\_SR .................................................................................................................... 59

8.3.10 RCC\_RST\_CR .................................................................................................................. 60

8.3.11 RCC\_SR ........................................................................................................................... 61

8.3.12 RCC\_SR1 ......................................................................................................................... 62

8.3.13 RCC\_CR3 ......................................................................................................................... 64

## 9. Interrupts ................................................................................................................................... 66

9.1 Main Features................................................................................................................................ 66

9.2 SysTick.................................................................................................................................. 66

9.3 Interrupt Vector Table...................................................................................................................... 66

## 10. Embedded Flash.............................................................................................................................. 68

10.1 Introduction..................................................................................................................................... 68

10.2 Main Features................................................................................................................................. 68

10.3 Functional Description..................................................................................................................... 68

10.3.1 Flash Info Area Division......................................................................................................... 68

10.3.2 EFC\_CR Protection............................................................................................................... 69

10.3.3 Read Access Latency........................................................................................................... 69

10.3.4 Acceleration for Accessing the Flash Memory....................................................................... 69

10.3.5 Instruction Prefetch............................................................................................................... 70

10.3.6 Flash Program Operation ..................................................................................................... 70

10.3.7 Flash Erase Operation........................................................................................................... 71

10.4 Flash Option Bytes .......................................................................................................................... 72

10.4.1 Flash Option0 ................................................................................................................... 72

10.4.2 Flash Option1 ................................................................................................................... 74

10.5 Embedded Flash Registers ......................................................................................................... 75

10.5.1 EFC\_CR ............................................................................................................................ 76

10.5.2 EFC\_INT\_EN .................................................................................................................... 78

10.5.3 EFC\_SR ............................................................................................................................ 79

10.5.4 EFC\_PROG\_DATA0 ........................................................................................................ 80

10.5.5 EFC\_PROG\_DATA1 ........................................................................................................ 80

10.5.6 EFC\_TIMING\_CFG .......................................................................................................... 81

10.5.7 EFC\_PROTECT\_SEQ ...................................................................................................... 81

10.5.8 SERIAL\_NUM\_LOW ......................................................................................................... 82

10.5.9 SERIAL\_NUM\_HIGH ........................................................................................................ 82

10.5.10 OPTION\_CSR\_BYTES ................................................................................................... 82

10.5.11 OPTION\_EXE\_ONLY\_BYTES ....................................................................................... 83

10.5.12 OPTION\_WR\_PROTECT\_BYTES ................................................................................. 84

10.5.13 OPTION\_SECURE\_BYTES0 ......................................................................................... 84

10.5.14 OPTION\_SECURE\_BYTES1 ......................................................................................... 85

## 11. GPIO .......................................................................................................................... 86

11.1 Introduction................................................................................................................................. 86

11.2 Output Configuration.................................................................................................................... 86

11.3 Input Configuration....................................................................................................................... 86

11.4 Output Drive Strength................................................................................................................... 87

11.5 GPIO Interrupts............................................................................................................................. 87

11.6 Wakeup from Sleep/Stop0~2 Mode.............................................................................................. 87

11.7 Wakeup from Stop3 Mode............................................................................................................ 87

11.8 Alternate Function Configuration.................................................................................................. 87

11.9 Clock Reset.................................................................................................................................. 87

11.10 Power Domain............................................................................................................................ 88

11.11 Low-power Mode Operation and Wakeup.................................................................................. 88

11.12 SWD IO. ..................................................................................................................................... 88

11.13 BOOT0 Control........................................................................................................................... 88

11.14 GPIO Registers…....................................................................................................................... 89

11.14.1 GPIOx\_OER (x=A, B, C, D) ............................................................................................ 90

11.14.2 GPIOx\_OTYPER (x=A, B, C, D) ..................................................................................... 90

11.14.3 GPIOx\_IER (x=A, B, C, D) .............................................................................................. 90

11.14.4 GPIOx\_PER (x=A, B, C, D) ............................................................................................ 91

11.14.5 GPIOx\_PSR (x=A, B, C, D) ............................................................................................ 91

11.14.6 GPIOx\_IDR (x=A, B, C, D) ............................................................................................. 91

11.14.7 GPIOx\_ODR (x=A, B, C, D) ............................................................................................ 92

11.14.8 GPIOx\_BRR (x=A, B, C, D) ............................................................................................ 92

11.14.9 GPIOx\_BSRR (x=A, B, C, D) .......................................................................................... 92

11.14.10 GPIOx\_DSR (x=A, B, C, D) .......................................................................................... 93

11.14.11 GPIOx\_INT\_CR (x=A, B, C, D) ..................................................................................... 93

11.14.12 GPIOx\_FR (x=A, B, C, D) ............................................................................................. 94

11.14.13 GPIOx\_WU\_EN (x=A, B, C, D) ..................................................................................... 94

11.14.14 GPIOx\_WU\_LVL (x=A, B, C, D) ................................................................................... 94

11.14.15 GPIOx\_AFRL (x=A, B, C, D) ........................................................................................ 95

11.14.16 GPIOx\_AFRH (x=A, B, C) ............................................................................................ 97

11.14.17 GPIOD\_AFRH .............................................................................................................. 99

11.14.18 GPIOA\_STOP3\_WU\_CR ........................................................................................... 101

11.14.19 GPIOx\_STOP3\_WU\_CR (x=B, C).............................................................................. 103

11.14.20 GPIOD\_STOP3\_WU\_CR ........................................................................................... 105

## 12. LoRa Controller (LoRaC) ......................................................................................................... 106

12.1 Introduction.................................................................................................................................... 106

12.2 Main Features................................................................................................................................ 106

12.3 Functional Description................................................................................................................... 106

12.3.1 Internal SPI Interface.......................................................................................................... 106

12.3.2 Timing Sequence of Power-on............................................................................................ 107

12.3.3 Interrupts………………........................................................................................................ 107

12.4 LoRaC Registers………. ................................................................................................................ 108

12.4.1 SSP\_CR0........................................................................................................................ 109

12.4.2 SSP\_CR1........................................................................................................................ 110

12.4.3 SSP\_DR .......................................................................................................................... 110

12.4.4 SSP\_SR .......................................................................................................................... 111

12.4.5 SSP\_CPSR ..................................................................................................................... 111

12.4.6 SSP\_IMSC ...................................................................................................................... 112

12.4.7 SSP\_RIS ......................................................................................................................... 112

12.4.8 SSP\_MIS ........................................................................................................................ 113

12.4.9 SSP\_ICR......................................................................................................................... 113

12.4.10 SSP\_DMACR ............................................................................................................... 113

12.4.11 LORAC\_CR0 ................................................................................................................ 114

12.4.12 LORAC\_CR1 ................................................................................................................ 114

12.4.13 LORAC\_SR................................................................................................................... 115

12.4.14 LORAC\_NSS\_CR ......................................................................................................... 116

12.4.15 LORAC\_SCK\_CR ......................................................................................................... 116

12.4.16 LORAC\_MOSI\_CR ....................................................................................................... 116

12.4.17 LORAC\_MISO\_SR ....................................................................................................... 117

## 13. UART ................................................................................................................... 118

13.1 Introduction….................................................................................................................................. 118

13.2 Clock Reset..................................................................................................................................... 118

13.3 Reference Clock.............................................................................................................................. 118

13.4 Baud Rate Generator....................................................................................................................... 118

13.5 FIFO .............................................................................................................................................. 119

13.6 UART Operation.............................................................................................................................. 119

13.6.1 Baud Rate Divisor................................................................................................................. 119

13.6.2 Data Transmission………...................................................................................................... 119

13.6.3 Data Reception...................................................................................................................... 120

13.7 IrDA SIR Operation.......................................................................................................................... 120

13.7.1 Low-Power Divisor................................................................................................................. 120

13.7.2 IrDA SIR IrDA SIR Transmit Encoder .................................................................................... 121

13.7.3 IrDA SIR IrDA SIR Receive Decoder.................................................................................. 121

13.8 UART Character Frame................................................................................................................... 121

13.9 IrDA Data Modulation....................................................................................................................... 122

13.10 Hardware Flow Control.................................................................................................................. 122

13.11 Interrupts........................................................................................................................................ 122

13.12 DMA .............................................................................................................................................. 122

13.13 UART Registers……... .................................................................................................................. 123

13.13.1 UARTx\_DR (x=0, 1, 2, 3) .............................................................................................. 124

13.13.2 UARTx\_RSR\_ECR (x=0, 1, 2, 3) .................................................................................. 125

13.13.3 UARTx\_FR (x=0, 1, 2, 3) .............................................................................................. 126

13.13.4 UARTx\_ILPR (x=0, 1, 2, 3) ........................................................................................... 127

13.13.5 UARTx\_IBRD (x=0, 1, 2, 3) .......................................................................................... 127

13.13.6 UARTx\_FBRD (x=0, 1, 2, 3) ......................................................................................... 127

13.13.7 UARTx\_LCR\_H (x=0, 1, 2, 3) ....................................................................................... 128

13.13.8 UARTx\_CR (x=0, 1, 2, 3) .............................................................................................. 129

13.13.9 UARTx\_IFLS (x=0, 1, 2, 3) ........................................................................................... 130

13.13.10 UARTx\_IMSC (x=0, 1, 2, 3) ........................................................................................ 130

13.13.11 UARTx\_RIS (x=0, 1, 2, 3) ........................................................................................... 131

13.13.12 UARTx\_MIS (x=0, 1, 2, 3) .......................................................................................... 132

13.13.13 UARTx\_ICR (x=0, 1, 2, 3) ........................................................................................... 132

13.13.14 UARTx\_DMACR (x=0, 1, 2, 3) .................................................................................... 133

13.13.15 UARTx\_ID[8] (x=0, 1, 2, 3) ......................................................................................... 134

## 14. SSP ……………….......................................................................................................................... 137

14.1 Introduction.................................................................................................................................... 137

14.2 Main Features................................................................................................................................ 137

14.3 Functional Description..................................................................................................................... 137

14.3.1 Basic Information.................................................................................................................. 137

14.3.2 Clock Division...................................................................................................................... 138

14.3.3 Data Format......................................................................................................................... 139

14.3.4 DMA Transaction.................................................................................................................. 139

14.3.5 SSP Interrupts………............................................................................................................ 140

14.4 SSP Registers………....................................................................................................................... 140

14.4.1 SSP\_CR0........................................................................................................................ 141

14.4.2 SSP\_CR1........................................................................................................................ 142

14.4.3 SSP\_DR .......................................................................................................................... 142

14.4.4 SSP\_SR .......................................................................................................................... 143

14.4.5 SSP\_CPSR ..................................................................................................................... 143

14.4.6 SSP\_IMSC ...................................................................................................................... 144

14.4.7 SSP\_RIS ......................................................................................................................... 144

14.4.8 SSP\_MIS ........................................................................................................................ 145

14.4.9 SSP\_ICR......................................................................................................................... 145

14.4.10 SSP\_DMACR ............................................................................................................... 146

## 15. I2C ……...................................................................................……………………............................... 147

15.1 Introduction................................................................................................................................. 147

15.2 Start and Stop Conditions........................................................................................................... 148

15.3 Data Transmission Sequence..................................................................................................... 149

15.4 Data and Addressing................................................................................................................... 150

15.5 Acknowledge............................................................................................................................... 151

15.6 Arbitration.................................................................................................................................... 151

15.7 I2C Master Mode......................................................................................................................... 152

15.8 FIFO Mode.................................................................................................................................. 154

15.9 I2C Slave Mode........................................................................................................................... 156

15.10 I2C Clock Reset......................................................................................................................... 157

15.11 I2C Interrupts............................................................................................................................. 157

15.12 DMA Requests........................................................................................................................... 157

15.13 I2C Registers……...................................................................................................................... 158

15.13.1 I2Cx\_CR (x=0, 1, 2) ...................................................................................................... 159

15.13.2 I2Cx\_SR (x=0, 1, 2) ...................................................................................................... 162

15.13.3 I2Cx\_SAR (x=0, 1, 2) .................................................................................................... 163

15.13.4 I2Cx\_DBR (x=0, 1, 2).................................................................................................... 164

15.13.5 I2Cx\_LCR (x=0, 1, 2) .................................................................................................... 164

15.13.6 I2Cx\_WCR (x=0, 1, 2)................................................................................................... 164

15.13.7 I2Cx\_RST\_CYCL (x=0, 1, 2) ........................................................................................ 165

15.13.8 I2Cx\_BMR (x=0, 1, 2) ................................................................................................... 165

15.13.9 I2Cx\_WFIF0 (x=0, 1, 2) ................................................................................................ 165

15.13.10 I2Cx\_WFIFO\_WPTR (x=0, 1, 2) ................................................................................. 166

15.13.11 I2Cx\_WFIFO\_RPTR (x=0, 1, 2) .................................................................................. 166

15.13.12 I2Cx\_RFIFO (x=0, 1, 2) .............................................................................................. 166

15.13.13 I2Cx\_RFIFO\_WPTR (x=0, 1, 2) .................................................................................. 167

15.13.14 I2Cx\_RFIFO\_RPTR (x=0, 1, 2) .................................................................................. 167

15.13.15 I2Cx\_WFIFO\_STATUS (x=0, 1, 2) ............................................................................. 167

15.13.16 I2Cx\_RFIFO\_STATUS (x=0, 1, 2) .............................................................................. 168

## 16. ADC ....………….......................................................................................................................... 169

16.1 Introduction..................................................................................................…............................. 169

16.2 ADC Input Mode.......................................................................................................................... 169

16.3 Sampling Channels...................................................................................................................... 170

16.4 Trigger Source.............................................................................................................................. 170

16.5 Low-power Operation................................................................................................................... 171

16.6 ADC Overrun................................................................................................................................ 171

16.7 Conversion Modes........................................................................................................................ 171

16.8 Voltage Reference........................................................................................................................ 171

16.9 Data Buffer.................................................................................................................................... 172

16.10 DMA Request.............................................................................................................................. 172

16.11 Interrupts..................................................................................................................................... 173

16.12 Wakeup…………......................................................................................................................... 173

16.13 ADC Clock and Reset.................................................................................................................. 173

16.14 ADC Registers……. .................................................................................................................... 173

16.14.1 ADC\_CR ....................................................................................................................... 174

16.14.2 ADC\_CFGR .................................................................................................................. 175

16.14.3 ADC\_SEQR0 ................................................................................................................ 177

16.14.4 ADC\_SEQR1 ................................................................................................................ 178

16.14.5 ADC\_DIFFSEL ............................................................................................................. 179

16.14.6 ADC\_ISR ...................................................................................................................... 179

16.14.7 ADC\_IER ...................................................................................................................... 180

16.14.8 ADC\_DR ....................................................................................................................... 180

## 17. Basic timer (BSTIM) .................................................................................................................... 181

17.1 Introduction................................................................................................................................. 181

17.2 Main features............................................................................................................................... 181

17.3 Clock source................................................................................................................................. 182

17.4 Counter......................................................................................................................................... 182

17.5 Auto-reload.................................................................................................................................. 182

17.6 Prescaler…..................................................................................................................................... 182

17.7 DMA control................................................................................................................................ 183

17.8 Single pulse mode....................................................................................................................... 183

17.9 Main mode selection.................................................................................................................. 183

17.10 Update event management.................................................................................................... 184

17.11 Debug mode control............................................................................................................. 184

17.12 Interrupts.................................................................................................................................. 184

17.13 BSTIMER registers ……........................................................................................................... 185

17.13.1 BSTIM\_CR1 .................................................................................................................. 186

17.13.2 BSTIM\_CR2 .................................................................................................................. 187

17.13.3 BSTIM\_DIER ................................................................................................................ 187

17.13.4 BSTIM\_SR .................................................................................................................... 188

17.13.5 BSTIM\_EGR ................................................................................................................. 188

17.13.6 BSTIM\_CNT.................................................................................................................. 188

17.13.7 BSTIM\_PSC.................................................................................................................. 189

17.13.8 BSTIM\_ARR ................................................................................................................. 189

## 18. RTC ....................................…….......................................................................................... 190

18.1 Introduction......................................................................................................................... 190

18.2 Main Features..................................................................................................................... 190

18.3 Interface Clock.................................................................................................................... 190

18.4 Calendar.............................................................................................................................. 191

18.4.1 Reading the Calendar............................................................................................... 191

18.4.2 Setting the Calendar.................................................................................................. 191

18.5 RTC PPM Calibration............................................................................................................ 192

18.6 Wake-up from Low-power Mode............................................................................................ 192

18.7 Tamper/Wakeup IO Detection................................................................................................ 193

18.7.1 Tamper/Wakeup Initialization and Configuration.......................................................... 193

18.7.2 Erase Operation on Retention SRAM............................................................................. 193

18.8 Periodic Counter........................................................................................................................ 193

18.9 RTC Alarms................................................................................................................................ 194

18.10 Internal Signal Output through IO............................................................................................. 195

18.11 RTC Interrupts.......................................................................................................................... 195

18.12 RTC Registers....................................................................................................................... 196

18.12.1 RTC\_CR ....................................................................................................................... 197

18.12.2 RTC\_ALARM0 .............................................................................................................. 200

18.12.3 RTC\_ALARM1 .............................................................................................................. 201

18.12.4 RTC\_PPMADJUST ....................................................................................................... 202

18.12.5 RTC\_CALENDAR ......................................................................................................... 202

18.12.6 RTC\_CALENDAR\_H .................................................................................................... 203

18.12.7 RTC\_CYC\_MAX\_VALUE ............................................................................................. 203

18.12.8 RTC\_SR ........................................................................................................................ 204

18.12.9 RTC\_ASYNDATA ......................................................................................................... 205

18.12.10 RTC\_ASYNDATA\_H .................................................................................................. 205

18.12.11 RTC\_CR1 ................................................................................................................... 206

18.12.12 RTC\_SR1 ................................................................................................................... 207

18.12.13 RTC\_CR2 ................................................................................................................... 208

18.12.14 RTC\_SUB\_SECOND .................................................................................................. 209

18.12.15 RTC\_CYC\_CNT\_VALUE............................................................................................ 209

### 18.12.16 RTC\_ALARM0\_SUB ................................................................................................... 210

### 18.12.17 RTC\_ALARM1\_SUB ................................................................................................... 210

18.12.18 RTC\_CALENDAR\_R .................................................................................................. 211

18.12.19 RTC\_CALENDAR\_R\_H.............................................................................................. 211

## 19. LPUART ..................................………………………….................................................................. 212

19.1 Introduction.................................................................................................................................... 212

19.2 Main Features................................................................................................................................ 212

19.3 Functional Description.................................................................................................................... 212

19.3.1 Data Format..................................................................................................................... 212

19.3.2 Baud Rate Generation...................................................................................................... 213

19.3.3 CTS/RTS Flow Control...................................................................................................... 213

19.3.4 DMA Transaction............................................................................................................... 214

### 19.3.5 LPUART Interrupt Signals................................................................................................ 214

19.3.6 CPU Wakeup from Low-power Mode..............………….................................................. 215

19.4 LPUART Registers…………............................................................................................................ 215

19.4.1 LPUART\_CR0 ................................................................................................................ 216

19.4.2 LPUART\_CR1 ................................................................................................................ 217

19.4.3 LPUART\_SR0 ................................................................................................................. 218

19.4.4 LPUART\_SR1 ................................................................................................................. 219

19.4.5 LPUART\_DATA .............................................................................................................. 220

## 20. Low power timer (LPTIM) ....................................................................................................... 221

20.1 Introduction................................................................................................................................. 221

20.2 Main features.................................................................................................................................. 221

20.3 Interface clock................................................................................................................................ 222

20.4 Counter clock selection.................................................................................................................. 222

20.5 Counter ........................................................................................................................................... 223

20.6 Counting modes.............................................................................................................................. 223

20.7 Software trigger and external trigger............................................................................................... 224

20.8 Prescaler .................................................................................................................................... 224

20.9 PWM ............................................................................................................................................. 225

20.10 Single-pulse, Set-once, Timeout output mode ........................................................................... 225

20.11 Orthogonal coding ...................................................................................................................... 226

20.12 DEBUG mode control ................................................................................................................. 227

20.13 Wake-up signals ........................................................................................................................ 227

20.14 interrupt signals......................................................................................................................... 228

20.15 LPTIMER registers ............................................................................................................ 228

### 20.15.1 LPTIM\_ISR ................................................................................................................... 229

### 20.15.2 LPTIM\_ICR ................................................................................................................... 230

### 20.15.3 LPTIM\_IER ................................................................................................................... 231

20.15.4 LPTIM\_CFGR ............................................................................................................... 232

20.15.5 LPTIM\_CR .................................................................................................................... 234

20.15.6 LPTIM\_CMP ................................................................................................................. 235

20.15.7 LPTIM\_ARR .................................................................................................................. 235

20.15.8 LPTIM\_CNT .................................................................................................................. 235

20.15.9 LPTIM\_CSR .................................................................................................................. 236

20.15.10 LPTIM\_SR1 ................................................................................................................ 237

## 21. Direct memory access controller (DMA) ................................................................................. 238

21.1 Introduction................................................................................................................................. 238

21.2 Main features.............................................................................................................................. 238

21.3 Transfer data length configuration............................................................................................ 238

21.4 Data trasfer methods................................................................................................................. 239

21.5 LLI ................................................................................................................................................ 241

21.6 Auto-reloading .............................................................................................................................. 241

21.7 Interrupts .................................................................................................................................... 242

21.8 DMA registers………..................................................................................................................... 243

21.8.1 DMA\_SARx ..................................................................................................................... 244

21.8.2 DMA\_DARx..................................................................................................................... 244

21.8.3 DMA\_LLPx ...................................................................................................................... 244

21.8.4 DMA\_CTLx ..................................................................................................................... 245

21.8.5 DMA\_CFGx..................................................................................................................... 247

21.8.6 DMA\_StatusTfr ............................................................................................................... 249

21.8.7 DMA\_StatusBlock ........................................................................................................... 249

### 21.8.8 DMA\_StatusSrcTran ....................................................................................................... 250

### 21.8.9 DMA\_StatusDstTran ....................................................................................................... 250

21.8.10 DMA\_StatusErr ............................................................................................................. 251

21.8.11 DMA\_MaskTfr ............................................................................................................... 252

21.8.12 DMA\_MaskBlock .......................................................................................................... 253

### 21.8.13 DMA\_MaskSrcTran ...................................................................................................... 254

### 21.8.14 DMA\_MaskDstTran ...................................................................................................... 255

21.8.15 DMA\_MaskErr .............................................................................................................. 256

21.8.16 DMA\_ClearTfr ............................................................................................................... 257

21.8.17 DMA\_ClearBlock .......................................................................................................... 257

### 21.8.18 DMA\_ClearSrcTran ...................................................................................................... 258

### 21.8.19 DMA\_ClearDstTran ...................................................................................................... 258

21.8.20 DMA\_ClearErr .............................................................................................................. 259

21.8.21 DMA\_DmaCfgReg ........................................................................................................ 260

21.8.22 DMA\_ChEnReg ............................................................................................................ 260

## 22. General purpose (GPTIMER)………..…………………………..................................................... 262

22.1 Introduction................................................................................................................................. 262

22.2 Main features............................................................................................................................... 262

22.3 Counter......................................................................................................................................... 264

22.3.1 Counter clock ................................................................................................................. 264

22.3.2 Auto-reload ..................................................................................................................... 265

### 22.3.3 Up-count ......................................................................................................................... 265

### 22.3.4 Down-count.................................................................................................................... 266

22.3.5 Center-alignment count................................................................................................. 266

22.4 Prescaler .................................................................................................................................... 267

22.5 Capture mode............................................................................................................................ 267

22.6 通道 aisle...................................................................................................................................... 268

22.6.1 Input capture............................................................................................................... 269

22.6.2 Output comparison.................................................................................................... 269

22.7 Trigger input channel................................................................................................................. 272

22.8 Update event management....................................................................................................... 272

22.9 Encoding mode control.............................................................................................................. 273

22.10 Control from mode................................................................................................................... 274

22.11 Master mode control................................................................................................................ 275

22.12 Output control........................................................................................................................... 276

22.13 Channel remapping................................................................................................................... 276

22.14 Debug mode control................................................................................................................. 276

22.15 DMA control............................................................................................................................... 276

22.16 Interrupt...................................................................................................................................... 277

22.17 GPTIMER registers………............................................................................................................ 277

### 22.17.1 GPTIM\_CR1 ................................................................................................................. 279

### 22.17.2 GPTIM\_CR2 ................................................................................................................. 280

22.17.3 GPTIM\_SMCR .............................................................................................................. 281

22.17.4 GPTIM\_DIER ................................................................................................................ 282

22.17.5 GPTIM\_SR.................................................................................................................... 284

22.17.6 GPTIM\_EGR ................................................................................................................. 285

### 22.17.7 GPTIM\_CCMR1 ............................................................................................................ 286

### 22.17.8 GPTIM\_CCMR2 ............................................................................................................ 289

22.17.9 GPTIM\_CCER .............................................................................................................. 292

22.17.10 GPTIM\_CNT ............................................................................................................... 294

### 22.17.11 GPTIM\_PSC ............................................................................................................... 294

### 22.17.12 GPTIM\_ARR ............................................................................................................... 295

### 22.17.13 GPTIM\_CCR0 ............................................................................................................. 295

### 22.17.14 GPTIM\_CCR1 ............................................................................................................. 295

### 22.17.15 GPTIM\_CCR2 ............................................................................................................. 296

### 22.17.16 GPTIM\_CCR3 ............................................................................................................. 296

22.17.17 GPTIM\_DCR ............................................................................................................... 296

22.17.18 GPTIM\_DMAR ............................................................................................................ 298

22.17.19 GPTIM\_OR ................................................................................................................. 298

# List of tables

Table 3-1 ASR6601 functions............................................................................................................................ 4

Table 6-1 working status of different modules in each working mode............................................................... 13

Table 7-1 Master bus access range.................................................................................................................. 22

Table 7-2 Memory map.................................................................................................................................... 23

Table 7-3 AHB0 SFR address mapping ........................................................................................................... 24

Table 7-4 AHB1 SFR address mapping ........................................................................................................... 24

Table 7-5 APB0 SFR address mapping ........................................................................................................... 24

Table 7-6 APB1 SFR address mapping ........................................................................................................... 25

Table 7-7 ASR6601 Boot Mode Configuration.................................................................................................. 26

Table 7-8 SYSCFG Register Summary............................................................................................................ 27 Table 7-9 DMA Request MUX.......................................................................................................................... 38 Table 8-1 RCC Register Summary................................................................................................................... 43 Table 9-1 Interrupt Vectors......................................................................................................................... 66 Table 10-1 Flash Info Area Division.................................................................................................................. 68 Table 10-2 Flash Option0................................................................................................................................. 72 Table 10-3 ASR6601 Boot Mode Configuration............................................................................................... 73 Table 10-4 Flash Option1................................................................................................................................. 74 Table 10-5 Embedded Flash Register Summary.............................................................................................. 75 Table 11-1 GPIO Register Summary............................................................................................................... 89 Table 12-1 LORAC Register Summary.......................................................................................................... 108 Table 13-1 Receive FIFO Bit Functions.......................................................................................................... 119 Table 13-2 UART Register Summary............................................................................................................. 123 Table 14-1 SSP Register Summary............................................................................................................... 140 Table 15-1 Start and Stop Conditions............................................................................................................ 148 Table 15-2 Master Transactions...................................................................................................................... 152 Table 15-3 Slave Transactions........................................................................................................................ 156 Table 15-4 I2C Register Summary................................................................................................................. 158 Table 16-1 ADC Sampling Channels............................................................................................................. 170 Table 16-2 ADC Register Summary............................................................................................................... 173 Table 17-1 BSTIMER interrupts............................................................................................................. 184 Table 17-2 BSTIMER Register Summary...................................................................................................... 185 Table 18-1 RTC Wakeup Source.................................................................................................................. 192 Table 18-2 Bits to Enable Wake-up Signals.................................................................................................... 192 Table 18-3 RTC Interrupts ..................................................................................................................... 195 Table 18-4 RTC Register Summary............................................................................................................... 196 Table 19-1 LPUART Register Summary........................................................................................................ 215 Table 20-1 LPTIMER0 external trigger source............................................................................................ 224 Table 20-2 LPTIMER1 external trigger source............................................................................................ 224 Table 20-3 Orthogonal coded channel signals........................................................................................... 227 Table 20-4 LPTIMER interrupts………............................................................................................................ 228

Table 20-5 LPTIMER Register Summary....................................................................................................... 228 Table 21-1 Handshake value......................................................................................................................... 239 Table 21-2 DMA interrupts.........………....…................................................................................................... 242

Table 21-3 DMA Register Summary.............................................................................................................. 243 Table 22-1 GPTIMER module introduction................................................................................................ 263 Table 22-2 Input channels polarity configuration...................................................................................... 268 Table 22-3 Input channel mapping.............................................................................................................. 268 Table 22-4 Output comparison output description……......……............................................................. 270 Table 22-5 Encoder mode............................................................................................................................ 273 Table 22-6 GPTIMER internal trigger input mapping............................................................................... 274 Table 22-7 GPTIMER interrupts.....................................…..................................................................... 277 Table 22-8 GPTIMER Register Summary......................…............................................................................. 277

# List of Figures

Figure 3-1 ASR6601 SoC Diagram................................................................................................................... 3 Figure 4-1 ASR6601 Power Grid..........................................................................….....……………………......... 9

Figure 4-2 ASR6601 Power Supply Architecture.............................................................................................. 10 Figure 7-1 System Architecture Diagram.......................................................................................................... 21 Figure 8-1 Clock Tree............................................................................................................................... 41 Figure 12-1 Timing Sequence of Power-on.................................................................................................... 107 Figure 13-1 UART Character Frame.............................................................................................................. 121 Figure 13-2 IrDA Data Modulation (3/16)....................................................................................................... 122 Figure 14-1 Connection between a SSP Master and a SPI Slave................................................................. 138 Figure 14-2 Connection between a SPI Master and a SSP Slave................................................................. 138 Figure 14-3 MASTER mode clock output calculation............................................................................... 138 Figure 15-1 I2C Block Diagram...................................................................................................................... 147 Figure 15-2 SDA and SCL Signals During Start and Stop Conditions........................................................... 148 Figure 15-3 FIFO Mode Block Diagram........................................................................................................ 154 Figure 16-1 ADC Diagram........................................................................................................................... 169 Figure 17-1 BSTIMER Diagram.................................................................................................................... 181 Figure 17-2 Counting and Dividing Waveforms......................................................................................... 182 Figure 17-3 Single pulse waveform................................................................................................................ 183 Figure 19-1 LPUART Data Format........................................................................................................... 212 Figure 19-2 Connection between Two LPUART Devices...........................................................…................. 213

Figure 20-1 LPTIMER Diagram.....................................................….............................................................. 222

Figure 20-2 Counting mode conversion Diagram...................................................................................... 223 Figure 20-3 Single pulse counting.................................................................................................................. 225 Figure 20-4 Set-once count....................................................................................................................... 226 Figure 20-5 Timeout count........................................................................................................................... 226 Figure 21-1 Data transfer...................................................................................................................... 238 Figure 21-2 LLI chain table........................................................................................................................... 241 Figure 22-1 GPTIMER diagram................................................................................................................... 263 Figure 22-2 External clock mode 1 count.................................................................................................... 264 Figure 22-3 External clock mode 2 count................................................................................................... 264 Figure 22-4 Internal trigger signal for clock counting............................................................................... 265 Figure 22-5 Up-counting............................................................................................................................... 265 Figure 22-6 Down-counting.......................................................................................................................... 266 Figure 22-7 Center- alignment counting.................................................................................................... 266 Figure 22-8 Prescaler............................................................................................................................... 267 Figure 22-9 Input capture............................................................................................................................. 269 Figure 22-10 Output compare mode waveforms....................................................................................... 271 Figure 22-11 PWM2 edge alignment count.................................................................................................... 271 Figure 22-12 PWM2 center alignment count.................................................................................................. 271 Figure 22-13 Single pulse output waveform in fast mode ...................................................................... 272 Figure 22-14 External brake signal trigger................................................................................................. 272 Figure 22-15 Counting waveform of encoding mode 1............................................................................ 274 Figure 22-16 Reset mode waveform in slave mode................................................................................... 274 Figure 22-17 Gated mode waveform in slave mode.................................................................................. 274 Figure 22-18 Trigger mode waveform in slave mode.....…….......................................................................... 275

1 Overview

## 1. Overview

ASR6601 is a general LPWAN Wireless Communication SoC chip developed by ASR which supports LoRa modulation. The chip integrates Sub-1G RF transceiver, Arm China STAR-MC1 processor, embedded Flash memory and SRAM, as well as diverse analog modules. ASR6601 is designed for a wide variety of applications, such as smart meters, building automation, smart cities, agricultural sensors, safety and security sensors, supply chain and logistics, etc.

This manual provides detailed and complete information on the IoT LPWAN SoC-ASR6601 for application developers. Together with the API file in SDK, it helps developers solve various problems they may encounter during development. If any further support is needed, please contact us. We will keep this manual updated

2 ASR6601 Introduction

## 2. ASR6601 Introduction

ASR6601 SoC is a low-power wide area network wireless communication SoC chip that supports LoRa modulation. The ultra-low power transceiver integrated in the ASR6601 chip can support the full frequency band of 150 MHz ~ 960 MHz with the off-chip matching network. In addition to supporting LoRa modulation, it can also support FSK transceiver, MSK transceiver and BPSK transmission. When powered by 3.3 V power supply, the maximum output power of 22 dBm can be transmitted through the high-power PA. ASR6601 SoC mainly has Run, LpRun, Sleep, LpSleep, Stop0, Stop1, Stop2, Stop3, Standby working modes. Each mode supports different functions, working modules and power consumption. End users can choose the corresponding working mode according to their application scenarios. The two most commonly used low-power modes are Standby mode and Stop3 mode. When powered by 3.3 V, the Standby mode consumes as little as 0.9 uA; the Stop3 mode consumes as little as 1.3 uA (ASR6601CB) and 1.6 uA (ASR6601SE).

The ASR6601 SoC uses a 32-bit ARM STAR core with a maximum main frequency of 48 MHz, supports SWD debug interface, supports SysTick, MPU, FPU functions, and supports 37 IRQs with 8 interrupt priorities.

ASR6601 supports UART, I2C, I2S, LPUART, SSP, QSPI and other interfaces. With the peripherals of different types of corresponding interfaces, it can realize rich functions to meet customer needs. In addition to supporting rich number functions, ASR6601 also integrates rich analog functions, including ADC, DAC, OPA and LCD driver.  
  
ASR6601 implements AES encryption through hardware, greatly simplifying the efficiency of encryption and decryption. It also supports national encryption SM2/3/4.

### 3. Modules and functions

#### 3.1 ASR6601 SoC Diagram



**Figure 3-1 ASR6601 SoC Diagram**

#### 3.2 ASR6601 functional modules

**Table 3-1 ASR6601 functional modules**

|  |  |  |
| --- | --- | --- |
| Module Name | Functions supported by the module | |
| RCC | Clock and reset control | |
| SYSCFG | System function registers | |
| PWR | 1. | Chip low power mode control |
|  | 2. | Interrupt signal generation |
| SEC | 1. | Security IP Enable |
| 2. | Filtering Security IP alarm signal filtering |
|  | 3. | Alarm signal processing, support status generation, interrupt request, and reset bit request |
| CPU | 1. | SWD debug interface |
| 2. | Systick function |
| 3. | MPU function |
|  | 4. | FPU function |
|  | 5. | 37 IRQs, 8 interrupt priorities |
| MPU | Access Security control, including slave access operations of cpu, dma, and swd debug interfaces | |
| EFC | 1. | Power-on chip mode determination |
| 2. | Flash info area data is loaded at power-on |
| 3. | Basic flash operations, including read, program, page erase, mass erase |
| 4. | Flash operation key timing control, including reading beat number, program high-voltage time, erase high-voltage time |
| 5. | Flash instruction prefetch function, 1 depth prefetch buffer |
|  | 6. | Flash program operation supports single and continuous modes |
|  | 7. | Flash info area option bytes operation |
|  | 8. | Interrupt signals generating |
| I2S | 1. | Philips I2S serial protocol |
| 2. | Support Master and Slave modes |
| 3. | 1 RX channel, 1 TX channel, full duplex |
| 4. | Receive FIFO depth is 4 |
| 5. | Transmit FIFO depth is 4 |
| 6. | Receiver supports 12, 16, 20, 24, 32-bit resolution |
|  | 7. | Transmitter supports 12, 16, 20, 24, 32-bit resolution |
|  | 8. | Supports programmable DMA registers |
|  |  | Supports programmable FIFO Threshold |
| 10. | Support 1 interrupt signal generation |
| UART |  | Support IrDA, support 3/16 and low-power (1.41-2.23us) Bit width |
|  | Support FIFO transmission and reception, 16x8bits transmission FIFO, 16x10bits reception FIFO |
|  | Supports Buffer sending and receiving, 1 deep sending and receiving buffer |
|  | Baud rate generation, using 16 times oversampling, supports 16-bit integer division and 6-bit fractional division, and supports up to interface clock frequency/16 |
|  |  | UART data format configuration, including 1-2 bits Stop, 0-1 bits parity (odd, even, mark, space, none), 5-8 data bits |
|  | Support DMA transfer |
|  | Support false start detection |
|  | Support line break sending and detection |
|  | Support hardware flow control CTS and RTS |
|  | Support interrupt signal generation |
|  | *1.* | *Support low power wake-up* |
| LPUART | 2. | Baud rate generation, does not support oversampling, supports 4-bit fractional frequency division and 12-bit integer frequency division, the minimum integer frequency division is 3 |
|  | 3. | UART data format configuration, including 1-2 bits Stop, 0-1 bits parity (odd, even, mark, space, none), 5-8 data bits |
|  | 4. | Support hardware flow control CTS and RTS |
|  | 5. | Support DMA transfer |
|  | 6. | Interrupt signal generation |
| SSP | 1. | Support Master and Slave |
| 2. | Programmable baud rate and prescaler, Master supports up to 1/2 interface clock frequency, Slave supports up to 1/12 interface clock frequency |
| 3. | Supports 8\*16 Bit receiving and transmitting FIFO |
| 4. | Data length is configurable, 4-16 Bit |
|  | 5. | Supports DMA requests |
|  | 6. | Supports Motorola, Microwire (NS), TI formats |
|  | 7. | Motorola supports 4 polarity phase bit combinations |
|  | 8. | Interrupt signal generation |
| I2C | 1. | Support master mode and slave module support multi-master arbitration |
| 2. | Support multi-host arbitration |
| 3. | Support Standard Mode and Fast Mode |
| 4. | Support 7 Bit address mode |
|  | 5. | Support Clock Stretching |
|  | 6. | Supports generating interrupt signals |
|  | 7. | Support DMA requests |
| AFEC | 1. | Simulate IP status register |
| 2. | Simulate IP control register |
| 3. | Some registers support Safety lockControl |
|  | 4. | Interrupt signal generation |
| LORAC | 1. | LORA IP control register |
| 2. | LORA status register |
| 3. | LORA IP SPI interface source, supports ssp master control and reg control |
|  | 4. | Support DMA request and response |
|  | 5. | Interrupt signal generation |

|  |  |  |  |
| --- | --- | --- | --- |
| RTC |  | Calendar counting function, using BCD format, supports seconds, minutes, hours, days, months, years, and days of the week | |
|  | Support ppm adjustment, adjustment step size is 0.5ppm, support +/-1024 ppm adjustment | |
|  | Support low power wake-up | |
|  | Tamper/wakeup IO detection function, supports effective level selection, and the number of filter beats is configurable | |
|  | Cycle counting function, 32-bit counter | |
|  | Alarm clock function, supports two alarm clocks, supports Mask selection and calendar matching | |
|  | Tamper/wakeup alarm clear retention sram function | |
|  | Some registers support Safety lockControl | |
|  | Internal signal IO output, including alarm0 matching pulse, alarm1 matching pulse, cycle count configuration pulse, seconds signal output | |
|  | Support calendar count value reading | |
|  | Support sub-second count value reading | |
|  | Supports count value reading of cycle counting | |
|  | Support interrupt signal generation | |
| IWDG | 1. | | Watchdog counting function, subtraction counting, counting clock supports prescaler (4-256 divider) |
| 2. | | Watchdog exception status occurs when the count reaches 0 (feeding the dog too late) or when the count value when feeding the dog is greater than the counting window value (feeding the dog too early) |
| 3. | | Supports generating interrupt signals |
|  | 4. | | Support dog feeding window configuration |
|  | 5. | | Support count value reading |
|  | 6. | | Support low power wake-up |
| QSPI | 1. | | Supports master interface only |
| 2. | | Supports 1-wire, 2-wire, 4-wire modes |
| 3. | | Supports 3 working modes, including indirect access, status query and Memory-mapping |
|  | 4. | | Supports baud rate division, up to interface clock frequency/2 |
|  | 5. | | Supports generating interrupt signals |
| CRC | 1. | | Configurable polynomial bit width, supports 7, 8, 16, 32 bits |
| 2. | | Supports different hsize accesses, the lower byte is calculated first and can be edited |
| 3. | | Programmable crc initial value |
|  | 4. | | Support input data reverse, support byte, halfword and word |
|  | 5. | | Support output data reverse, support word |
| DMA | 1. | | Supports 1 master interface AHB bus |
| 2. | | The AHB interface only supports little-endian structure |
| 3. | | Support interrupt signal generation |
| 4. | | Transmission mode, supports M2M, P2M, M2P, P2P |
| 5. | | Support software triggering handshake signal |
|  | 6. | | Supports 4 sets of hardware handshake signals, including burst and single requests |
|  | 7. | | Supports hardware handshake signal sources, each group supports 64 source selections |
|  | 8. | | Supports 4 logical channels |
|  | 9. | | Channel 0 configutation:  (1) 8 bytes deep FIFO   1. Maximum burst length is 8 2. The maximum transfer length is 2047 3. Only supports dmac flow control 4. Source address data bit width configurable 5. Destination address data bit width configurable 6. The address supports increment, decrement, and unchanged 7. Support block transfer, including continuous address, automatic loading and linked list 8. Support scatter and gather |

|  |  |  |
| --- | --- | --- |
|  | 10. | Channel 1-3 configuration:   1. 8 bytes deep FIFO 2. Maximum burst length is 8 3. The maximum transfer length is 2047 4. Only supports dmac flow control 5. Source address data bit width configurable 6. Destination address data bit width configurable 7. The address supports increment, decrement, and unchanged 8. Supports block transfer, including continuous addresses and automatic loading, but does not support linked lists 9. Scatter and gather are not supported |
| GPIO | 1. | IO output configuration, supports push-pull, open drain, output high impedance |
| 2. | IO input configuration, supports floating, input pull-up, input pull-down, analog input |
| 3. | IO other configurations, pull-up configuration, pull-down configuration, drive capability Control |
|  | 4. | Supports generating interrupt signals, including rising edge interrupt, falling edge interrupt, and double edge interrupt |
|  | 5. | Supports generating wake-up signals, including high level and low level |
| SAE | 1. | Support AES128/192/256 |
| 2. | Supports DES and 3DES |
| 3. | Support SM2, SM3, SM4 (ASR6601SE) |
| 4. | Support RSA1024/2048 |
|  | 5. | Support ECC224/256/384/512 |
|  | 6. | Support SHA1, SHA-224, SHA256, SHA384, SHA512 |
|  | 7. | Support random number generator |
| BSTIMER | 1. | 32bits counter, supports automatic loading, supports addition, subtraction, addition and subtraction counting |
| 2. | 16bits count clock prescaler |
| 3. | Supports DMA requests |
|  | 4. | Supports generating interrupt signals |
| GPTIMER | 1. | 32bits counter, supports automatic loading, supports addition, subtraction, addition and subtraction counting |
| 2. | 16bits count clock prescaler |
| 3. | gptimer0 and gptimer1 support 4 Channels, gptimer2 and gptimer3 support 2 Channels, each Channel can support |
|  | 4. | Channel can support input capture, output comparison, PWM generation, single pulse output |
|  | 5. | Supports generating interrupt signals |
|  | 6. | Supports DMA requests |
| LPTIMER | 1. | Supports selecting internal clock and IO clock as counting clock |
| 2. | 16bits counter, additive counting, supports automatic loading |
| 3. | Support count clock prescaler |
| 4. | Supports counting clock prescaler and supports quadrature decoding |
|  | 5. | Support input capture, output comparison, PWM generation, single pulse output |
|  | 6. | Supports generating interrupt signals |
|  | 7. | Supports DMA requests |
| ADC | 1. | Sampling accuracy 12 bits |
| 2. | Configurable sampling speed up to 1 MHz |
| 3. | Supports single-ended and differential sampling |
| 4. | Data alignment only supports right alignment |
| 5. | Supports 8 external channels |
| 6. | Supports 7 internal channels, including DAC output, internal Vref, VDD/3 (battery power), Vts (internal temperature sensor), OPA output (3) |
|  | 7. | Trigger mode, supports software trigger and hardware trigger |
|  | 8. | Sampling mode, supports sampling sequence, continuous, single, and non-continuous |
|  |  | Supports analog watchdog function, 3 channels in total, configurable Channel selection and high and low thresholds |
|  | Supports DMA requests |
|  | Supports generating interrupt signals |
| DAC | 1. | Output accuracy 10 bits |
| 2. | Configurable output speed up to 1 MHz |
| 3. | Data alignment only supports right alignment |
| 4. | Special waveform output, supports triangle wave |
|  | 5. | Trigger mode, supports software trigger and hardware trigger |
|  | 6. | Supports DMA requests |
|  | 7. | Supports generating interrupt signals |
| LCDCTRL | 1. | Frame rate division control |
| 2. | Bias control, supports static, 1/2, 1/3, 1/4 |
| 3. | Duty Control, supports static (1comx27seg), 1/2 (2comx26seg), 1/3 (3comx25seg), 1/4 (4comx24seg), 1/8 (8comx20seg) |
| 4. | Dead frame control, supports dead frame of 0-7 shots, used to adjust contrast |
|  | 5. | blink control, supports the blinking function of 1, 2, 3, 4, 8 or all pixels, the blinking frequency is configurable |
|  | 6. | Supports large and small current selection control, including state machine dynamic control and register static control. During state machine dynamic control, high current can be configured to maintain the number of beats. |
|  | 7. | Interrupt signal generation |

4 power supply

### 4. Power Management Unit

#### 4.1 Power supply pins

ASR6601 has several independent power supply pins. By separating these power supply pins, the mutual influence between the various parts on the SoC can be well avoided, especially the interference of the SoC number part on the performance of the RF transceiver when it is working.

The power supply interface of ASRR6601 is shown in Figure 4-1:

* **VDD\_IN**:RF transceiver PA Power supply.
* **VBAT\_RF**:Power supply for RF transceiver.
* **VDCC\_RF**:The power supply for some modules in the RF transceiver must be connected to the VREG pin of the chip on the PCB.
* **VBAT\_ESD0**: digital IO Power supply.
* **VBAT\_ESD1**:digital IO Power supply.
* **VBAT\_ESD2**:digital IO Power supply.
* **VBAT\_ESD3**:digital IO Power supply.
* **VBAT\_DCC**: analog circuits DCDC separate power supply.
* **VBAT\_ESD\_RTC**: RTC domain IO power supply.
* **VBAT\_RTC**:RTC backup power supply.
* **VBAT\_ANA**: Analog Circuit Power supply.



**Figure 4-1 ASR6601 Power Grid**

#### 4.2 Chip internal power architecture

Internal power domains of the chip are mainly divided into *main* domain, *aon* domain and *aonr* domain. Please note that the power domains are divided according to functions, as shown in Figure 4-2.

1. **Main** domain contains most of the digital logic circuits of the SoC chip. In the frequentlyused low-power modes (Standby and Stop3), the power supply of main domain will be

turned off.

2. **Aon** (always on domain) means that the power supply for this domain is always available,

even in low-power mode. Most blocks in aon domain keep running in all power modes.

3. **Aonr** (Always on and retention) domain contains the modules that need to keep running

in Stop3 mode. These modules will be powered off in Standby mode. When aonr domain

modules remain in the current state without power off, the system can quickly recover and

continue to execute.



**Figure 4-2 ASR6601 Power Supply Architecture**

5 Permission control

### 5. Access Control

#### 5.1 Simple configuration

##### This section provides customers with commonly used simple configurations to meet their basic security needs.

##### 5.1.1 Recoverable Security Configuration

⚫ **Enable Security**

Configure FlashSecStart to 0 and FlashSecEnd to 0x3F in OPTION1 tab, and set the entire Flash\_main area as a secure area.

Consequently, the code in SWD (Serial Wire Debug) and non-secure area cannot read and write data into Flash\_main to guarantee security. Please note that code in non-secure SRAM area or non secure DMA will not be able to access Flash\_main.

⚫ **Disable Security**

Configure FlashSecStart to 0x3F and FlashSecEnd to 0 in OPTION1 tab, and set the entire Flash\_main area as a non-secure area.

The above configurations will erase the entire Flash\_main area, and then the program can be re-downloaded.

**5.1.2 Unrecoverable Security Configuration**

Configure the DebugLevel to 2 in Option0 tab. This operation is irreversible, and the code must be correct and strong.

**5.2 Access control**

Based on debug level rules, boot startup mode, exe-only access rules, write-protected access rules, info area access rules, and secure area access rules, access rights from the four main interfaces (cpucode, cpusw, dmac0, and dmac1) are controlled.

**5.2.1 Debug Level Rules**

Debuglevel mainly affects the access rights of cpu\_code (boot from SRAM and boot from bootloader), cpu\_sw, dmac0 and dmac1 to sensitive areas. Sensitive areas include flash\_main, otp area of flash\_info and retention SRAM. For details see "ASR6601 Access Rights Control Instructions".

##### 5.2.2 Secure and Non-Secure Operation

⚫ **Secure Operation**

The operations initiated by the code in the secure area include:

⬩ Operations initiated by DMAC0 configured as a secure area

⬩ Operations initiated by flash\_main configured as a secure area (CPU\_Code)

⬩ Operations initiated by system\_sram configured as a secure area (CPU\_Code)

⚫ **Non-secure Operation**

The operations initiated by the code in the non-secure area include:

⬩ Operations initiated by DMAC0 configured as a non-secure area

⬩ Operations initiated by DMAC1

⬩ Operations initiated by Debug Port (CPU\_SW)

⬩ Operations initiated by Bootloader (CPU\_Code)

⬩ Operations initiated by flash\_main configured as a non-secure area (CPU\_Code)

⬩ Operations initiated by system\_sram configured as a non-secure area (CPU\_Code)

### 6. Operation Mode

ASR6601 LPWAN SoC supports Run, LpRun, Sleep, LpSleep, Stop0, Stop1, Stop2, Stop3 and Standby modes. Each mode supports different functions with different working modules and power consumption. The user can choose the appropriate operation mode according to specific application scenarios. All modes are described detailedly in the contents below.

In addition, please note the following points:

1. When entering a low-power mode, peripherals marked as “O” (excluding GPIO) are turned

off by default. The functions used in low-power mode must be turned on before entering

low-power mode.

2. When entering a low-power mode, developers need to configure below items to achieve

specified power consumption:

(1) Configure unused GPIOs to ANALOG mode (high impedance)

(2) If the GPIOs are in input mode, users should configure them pull-up or pull-down

(3) In output mode, configure the connected peripherals pull-up or pull-down according

to the output level.

3. Use RCO48M/2 to enter or exit a low-power mode. If you use a clock other than RCO48M/2

before entering a low-power mode, you need to switch to RCO48M/2. After exiting the lowpower mode, you can switch to the previously used clock.

4. RCO32K/XO32K and some other analog functions can retain active in low-power modes.

If needed, turn on these functions before entering a low-power mode by software.

5. Clocks other than RCO48M/RCO32K/XO32K and the remaining analog function modules must be turned off by the software before entering a low-power mode.

#### Table 6-1 Modules Working Status in Various Operation Modes

|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
|  | Run | LpRun | Sleep | LpSleep | Stop0 | Stop1 | Stop2 |  | Stop3 | Standby |  | Stop0-2  Wakeup | Stop3  Wakeup | Standby  Wakeup |
| cpu | Y | Y | NA | NA | NA | NA | NA | NA |  | NA |  |  |  |  |
| efc | Y | Y | O | O | NA | NA | NA | NA |  | NA |  |  |  |  |
| sysramc | Y | Y | O | O | NA | NA | NA | NA |  | NA |  |  |  |  |
| retramc | Y | Y | O | O | NA | NA | NA | NA |  | NA |  |  |  |  |
| i2s | O | O | O | O | NA | NA | NA | NA |  | NA |  |  |  |  |
| uart0 | O | O | O | O | NA | NA | NA | NA |  | NA |  |  |  |  |
| uart1 | O | O | O | O | NA | NA | NA | NA |  | NA |  |  |  |  |
| uart2 | O | O | O | O | NA | NA | NA | NA |  | NA |  |  |  |  |
| uart3 | O | O | O | O | NA | NA | NA | NA |  | NA |  |  |  |  |
| ssp0 | O | O | O | O | NA | NA | NA | NA |  | NA |  |  |  |  |
| ssp1 | O | O | O | O | NA | NA | NA | NA |  | NA |  |  |  |  |
| ssp2 | O | O | O | O | NA | NA | NA | NA |  | NA |  |  |  |  |
| qspi | O | O | O | O | NA | NA | NA | NA |  | NA |  |  |  |  |
| i2c0 | O | O | O | O | NA | NA | NA | NA |  | NA |  |  |  |  |

|  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
|  |  |  |  |  |  |  |  |  | |  |  |  |  |
| i2c1 | O | O | O | O | NA | NA | NA | NA | | NA |  |  |  |
| i2c2 | O | O | O | O | NA | NA | NA | NA | | NA |  |  |  |
| adcctrl | O | O | O | O | NA | NA | NA | NA | | NA |  |  |  |
| dacctrl | O | O | O | O | NA | NA | NA | NA | | NA |  |  |  |
| gptim0 | O | O | O | O | NA | NA | NA | NA | | NA |  |  |  |
| gptim1 | O | O | O | O | NA | NA | NA | NA | | NA |  |  |  |
| gptim2 | O | O | O | O | NA | NA | NA | NA | | NA |  |  |  |
| gptim3 | O | O | O | O | NA | NA | NA | NA | | NA |  |  |  |
| basictim0 | O | O | O | O | NA | NA | NA | NA | | NA |  |  |  |
| basictim1 | O | O | O | O | NA | NA | NA | NA | | NA |  |  |  |
| wwdg | O | O | O | O | NA | NA | NA | NA | | NA |  |  |  |
| crc | O | O | O | O | NA | NA | NA | NA | | NA |  |  |  |
| sec | O | O | O | O | NA | NA | NA | NA | | NA |  |  |  |
| sac | O | O | O | O | NA | NA | NA | NA | | NA |  |  |  |
| mpu | O | O | O | O | NA | NA | NA | NA | | NA |  |  |  |
| dmac0 | O | O | O | O | NA | NA | NA | NA | | NA |  |  |  |
| dmac1 | O | O | O | O | NA | NA | NA | NA | | NA |  |  |  |
| syscfg | O | O | O | O | NA | NA | NA | NA | | NA |  |  |  |
| afec | O | O | O | O | NA | NA | NA | NA | | NA |  |  |  |
| lorac | O | O | O | O | NA | NA | NA | NA | | NA |  |  |  |
| gpio | O | O | O | O | NA | NA | NA | GPIO0~55: Y3  GPIO56~63:Y4 | | GPIO0~55: NA3  GPIO56~63: Y4 | Y | Y |  |
| rcc | Y | Y | Y | Y | Y | Y | Y | Y | | Y |  |  |  |
| pwr | Y | Y | Y | Y | Y | Y | Y | Y | | Y |  |  |  |
| lpuart | O | O | O | O | O | O | O | O (RX only) | | O (RX only) | Y | Y | Y |
| lcdctrl | O | O | O | O | O | O | O | O | | O |  |  |  |
| lptim0 | O | O | O | O | O | O | O | O | | O | Y | Y | Y |
| lptim1 | O | O | O | O | O | O | O | O | | O | Y | Y | Y |
| iwdg | O | O | O | O | O | O | O | O | | O | Y1 | Y | Y |
| rtc | O | O | O | O | O | O | O | O | | O | Y | Y | Y |
| ADC | O | O | O | O | NA | NA | NA | NA | | NA |  |  |  |
| RCO48M | O | O | O | O | NA | NA | NA | NA | | NA |  |  |  |
| XO24M | O | O | O | O | NA | NA | NA | NA | | NA |  |  |  |
| PLL48M | O | O | O | O | NA | NA | NA | NA | | NA |  |  |  |
| RNG | O | O | O | O | NA | NA | NA | NA | | NA |  |  |  |
| DAC | O | O | O | O | O3 | O3 | O3 | NA | | NA |  |  |  |
| OPA | O | O | O | O | O | O | O | NA | | NA |  |  |  |
| COMP | O | O | O | O | O | O | O | O | | O | Y | Y | Y |
| VD | O | O | O | O | O | O | O | O | | O | Y | Y | Y |
| RCO3.6M | O | O | O | O | O | O | O | O | | O |  |  |  |
|  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| RCO32K | O | O | O | O | O | O | O | O |  | O |  |  |  |
| XO32K | O | O | O | O | O | O | O | O |  | O |  |  |  |
| LCD | O | O | O | O | O | O | O | O |  | O |  |  |  |
| BOR | Y | Y | Y | Y | Y | Y | Y | Y |  | Y | Y2 | Y2 | Y2 |
| FLASH | Y | Y | Y | Y | SLM | SLM | SLM | PDM |  | PDM |  |  |  |
| SRAM | Y | Y | Y | Y | NA | NA | NA | NA1 |  | NA2 |  |  |  |
| IO | Y | Y | Y | Y | Y | Y | Y | Y |  | Y |  |  |  |
| RF | O | O | O | O | O | O | O | O |  | O | Y | Y | Y |

Notes and symbol annotations for the above table:

* **stop0-2**: all GPIOs can be configured to wake up the CPU; all GPIOs retain the previous

state in Stop0-2 mode.

* **stop3**: 56 GPIOs in the main domain can be configured to wake up the CPU; all GPIOs

retain the previous state in Stop3 mode.

* **standby**: 8 GPIOs in the AON domain retain the previous state in Standby mode; 56

GPIOs in the main domain are used as analog functions (such as LCD, COMP) and

cannot be used to wake up the CPU. The LPUART only supports RX in Standby/Stop3

mode.

* **Y**: Work normally.
* **O**: Optional, configured by software.
* **O3**: Data update is not supported, but the output retains current voltage level.
* **Y1**:Generate system reset to wake up the system indirectly
* **Y2**: Generate BOR reset to wake up the system indirectly.
* **Y3**: Retain the state before entering low-power mode, and can be used to wake up the

CPU.

* **Y4**: IO MUX Function=1 function NA; other multiplexing functions work normally.
* **NA1**: Retention and algorithm contents are kept. System content can be configured to be

kept or not.

* **NA2**: Retention content is kept.
* **NA3**: Analog Output Only

#### 6.1 Run

##### 6.1.1 Enter and Exit

***Run*** mode is the default operation mode after power-on or system reset.

ASR6601 can enter Sleep, LpRun, Stop0, Stop1, Stop2, Stop3 or Standby mode from ***Run***

mode.

ASR6601 can return to ***Run*** mode from Sleep, LpRun, Stop0, Stop1, Stop2, Stop3 or Standby

mode.

For detailed mode switching conditions, please refer to the descriptions of other operation

modes.

**6.1.2 Wakeup Source**

N/A

#### 6.2 LpRun

##### 6.2.1 Enter and Exit

Enter ***LpRun*** mode from Run mode in the following way:

Turn off all high-speed clocks to make CPU run at 32K clock frequency. Then switch the working

state of LDO by software.

LpRun config register is used to switch LDO working state:

(1) Set bits[3:3] of the register (address 0x05) to 1, and the other bits remain unchanged.

(2) Set bits[21:20] of the register (address 0x06) to 1, and the other bits remain unchanged.

Return to ***Run*** mode from LpRun mode in the following way:

Switch the working state of LDO by software. Then turn on the high-speed clock

LpRun config register is used to switch LDO working state:

(1) Clear bits[21:20] of the register (address 0x06) to 0, and the other bits remain unchanged.

(2) Clear bits[3:3] of the register (address 0x05) to 0, and the other bits remain unchanged.

**6.2.2 Wakeup Source**

N/A

#### 6.3 Sleep

##### 6.3.1 Enter and Exit

Enter ***Sleep*** mode from Run mode in the following way:

CPU executes WFI/WFE instruction SLEEPDEEP=0, or isr returns SLEEPONEXIT=1 and

SLEEPDEEP=0.

Return to ***Run*** mode from Sleep mode in the following ways:

 If WFI instruction is used to enter Sleep mode, then the system is waked-up by interrupts.

 If WFE instruction is used to enter Sleep mode, then the system is waked-up by events.

***Notice:*** *Since there is no dedicated wake-up event signal, the interrupt signal is used to*

*generate wake-up events by instruction SVONPEND=1 and turning off the corresponding NVIC.*

**6.3.2 Wakeup Source**

Interrupt signal generated by each module.

#### 6.4 LpSleep

##### 6.4.1 Enter and Exit

Enter ***LpSleep*** mode from LpRun mode in the following way:

CPU executes WFI/WFE instruction SLEEPDEEP=0, or isr returns SLEEPONEXIT=1 and

SLEEPDEEP=0.

Return to ***LpRun*** mode from LpSleep mode in the following ways:

 If WFI instruction is used to enter LpSleep mode, then the system returns LpRun mode

by interrupts.

 If WFE instruction is used to enter LpSleep mode, then the system returns LpRun mode

by wake-up events.

***Notice:*** *Since there is no dedicated event wake-up signal, the interrupt signal is used to*

*generate wake-up events by instruction SVONPEND=1 and turning off the corresponding NVIC.*

**6.4.2 Wakeup Source**

Interrupt signal of each module.

#### 6.5 Stop0

##### 6.5.1 Enter and Exit

Enter ***Stop0*** mode from Run mode in the following way:

Configure lp\_mode to 2’b00, then CPU executes WFI/WFE instruction SLEEPDEEP=1, or isr

returns SLEEPONEXIT=1 and SLEEPDEEP=1.

Return to ***Run*** mode from Stop0 mode in the following ways:

 If WFI instruction is used to enter Stop0 mode, then the system is waked-up by interrupts.

 If WFE instruction is used to enter Stop0 mode, then the system is waked-up by events.

The pwr module manages the status of the wake-up sources and outputs the *pwr\_wakeup\_int*

signal and the *pwr\_wakeup\_event* signal to wake up the CPU.

##### 6.5.2 Wakeup Source

* GPIO00-GPIO63 can all be used to wake up the CPU, 4 IOs make up a group, and each

group can select any of the 4 IOs for wake-up. A group generates a wake-up signal, and

any of the IOs can wake up the CPU at high or low level. The wake-up sources other than

GPIOs are listed below.

* PVM Alarm  Wakeup/Tamper IO
* VD Alarm  RTC Alarm
* TD Alarm  RTC CYC Timer
* LD Alarm  LPUART RX Status
* Comparator  LORA BUSY
* LPTIM0/1  LORA IRQ
* FD\_32K Alarm

#### 6.6 Stop1

##### 6.6.1 Enter and Exit

Enter ***Stop1*** mode from Run mode in the following way:

Configure lp\_mode to 2’b01, then CPU executes WFI/WFE instruction SLEEPDEEP=1, or isr

returns SLEEPONEXIT=1 and SLEEPDEEP=1.

Return to ***Run*** mode from Stop1 mode in the following ways:

 If WFI instruction is used to enter Stop1 mode, then the system is waked-up by interrupts.

 If WFE instruction is used to enter Stop1 mode, then the system is waked-up by events.

The pwr module manages the status of the wake-up sources and outputs the *pwr\_wakeup\_int*

signal and the *pwr\_wakeup\_event* signal to wake up the CPU.

###### 6.6.2 Wakeup Source

GPIO00-GPIO63 can all be used to wake up the CPU, 4 IOs make up a group, and each

group can select any of the 4 IOs for wake-up. A group generates a wake-up signal, and

any of the IOs can wake up the CPU at high or low level. The wake-up sources other than

GPIOs are listed below.

* PVM Alarm  Wakeup/Tamper IO
* VD Alarm  RTC Alarm
* TD Alarm  RTC CYC Timer
* LD Alarm  LPUART RX Status
* Comparator  LORA BUSY
* LPTIM0/1  LORA IRQ
* FD\_32K Alarm

#### 6.7 Stop2

##### 6.7.1 Enter and Exit

Enter ***Stop2*** mode from Run mode in the following way:

Configure lp\_mode to 2’b10, then CPU executes WFI/WFE instruction SLEEPDEEP=1, or isr

returns SLEEPONEXIT=1 and SLEEPDEEP=1.

Return to ***Run*** mode from Stop2 mode in the following ways:

 If WFI instruction is used to enter Stop2 mode, then the system is waked-up by interrupts.

 If WFE instruction is used to enter Stop2 mode, then the system is waked-up by events.

The pwr module manages the status of the wake-up sources, and outputs *pwr\_wakeup\_int*

signal and *pwr\_wakeup\_event* signal to wake up the CPU.

###### 6.7.2 Wakeup Source

GPIO00-GPIO63 can all be used to wake up the CPU, 4 IOs make up a group, and each

group can select any of the 4 IOs for wake-up. A group generates a wake-up signal, and

any of the IOs can wake up the CPU at high or low level. The wake-up sources other than

GPIOs are listed below.

* PVM Alarm  Wakeup/Tamper IO
* VD Alarm  RTC Alarm
* TD Alarm  RTC CYC Timer
* LD Alarm  LPUART RX Status
* Comparator  LORA BUSY
* LPTIM0/1  LORA IRQ
* FD\_32K Alarm

#### 6.8 Stop3

##### 6.8.1 Enter and Exit

Enter ***Stop3*** mode from Run mode in the following way:

Configure lp\_mode to 2’b11 and lp\_mode\_ext to 1’b1, then CPU executes WFI/WFE instruction

SLEEPDEEP=1, or isr returns SLEEPONEXIT=1 and SLEEPDEEP=1.

The system returns to ***Run*** mode if a wake-up event occurred in Stop3 mode.

###### 6.8.2 Wakeup Source

GPIO00-GPIO55 can all be used to wake up the CPU, 4 IOs make up a group, and each

group can select any of the 4 IOs for wake-up. A group generates a wake-up signal, and

any of the IOs can wake up the CPU at high or low level. The wake-up sources other than

GPIOs are listed below.

* PVM Alarm  RTC Alarm
* VD Alarm  RTC CYC Timer
* Comparator  LPUART RX Status
* LPTIM0/1  LORA BUSY
* FD\_32K Alarm  LORA IRQ
* Wakeup/Tamper IO  IWDG Timeout

#### 6.9 Standby

##### 6.9.1 Enter and Exit

Enter ***Standby*** mode from Run mode in the following way:

Configure lp\_mode to 2’b11 and lp\_mode\_ext to 1’b0, then CPU executes WFI/WFE instruction

SLEEPDEEP=1, or isr returns SLEEPONEXIT=1 and SLEEPDEEP=1.

The system returns to ***Run*** mode if a wake-up event occurred in Standby mode.

***Notice:***

*1. When the power supply is switched between DCDC and VBAT, the CPU will return to Run*

*mode immediately after entering Standby mode without any wake-up event.*

*2. When dbg\_standby=1, the switch between DCDC and VBAT is disabled.*

###### 6.9.2 Wakeup Source

* PVM Alarm  RTC Alarm
* VD Alarm  RTC CYC Timer
* Comparator  LPUART RX Status
* LPTIM0/1  LORA BUSY
* FD\_32K Alarm  LORA IRQ
* Wakeup/Tamper IO  IWDG Timeout

### 7. System Configuration

#### 7.1 System Architecture

#### 



**Figure 7-1 System Architecture Diagram**

##### 7.1.1 Arm China STAR-MC1 Processor

Arm China STAR-MC1 Processor consists of three master buses, including icode AHB bus,

dcode AHB bus and system AHB bus, which are used for program access, data access and

register access.

**7.1.2 DMAC0**

DMAC0 has a master bus, which can assist the CPU to transfer data.

##### 7.1.3 DMAC1

DMAC1 has a master bus, which can assist the CPU to transfer data.

**7.1.4 Master**

The addresses accessible by each master bus is shown in the table below.

***(1)*** *Only accessible when boot from Bootloader.*

**Table 7-1 Master Bus Access Scope**

|  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- |
| Start  Address | End Address | Description | Execut- able | icode  Access | dcode  Access | system  Access | dmac0 Access | dmac1 Access |
| 0xE0100000 | 0xFFFFFFFFF | Reserved |  |  |  |  |  |  |
| 0xE0000000 | 0xE00FFFFF | ARM STAR peripherals |  |  |  |  |  |  |
| 0xA0000000 | 0xDFFFFFFF | Reserved |  |  |  |  |  |  |
| 0x70000000 | 0x9FFFFFFF | Reserved |  |  |  |  |  |  |
| 0x60000000 | 0x6FFFFFFF | Qspi Flash Bank | Y |  |  | Y | Y | Y |
| 0x50000000 | 0x5FFFFFFF | Reserved |  |  |  |  |  |  |
| 0x40030000 | 0x4FFFFFFF | AHB1 SFR |  |  |  | Y | Y | Y |
| 0x40020000 | 0x4002FFFF | AHB0 SFR |  |  |  | Y | Y | Y |
| 0x40010000 | 0x4001FFFF | APB1 SFR |  |  |  | Y | Y | Y |
| 0x40000000 | 0x4000FFFF | APB0 SFR |  |  |  | Y | Y | Y |
| 0x30000400 | 0x3FFFFFFF | Reserved |  |  |  |  |  |  |
| 0x30000000 | 0x300003FF | Retention SRAM |  |  |  | Y | Y | Y |
| 0x20010000 | 0x2FFFFFFF | Reserved |  |  |  |  |  |  |
| 0x20000000 | 0x2000FFFF | System SRAM | Y |  |  | Y | Y | Y |
| 0x18010000 | 0x1FFFFFFF | Reserved |  |  |  |  |  |  |
| 0x18000000 | 0x1800FFFF | System SRAM | Y | Y | Y |  |  |  |
| 0x10004000 | 0x17FFFFFF | Reserved |  |  |  |  |  |  |
| 0x10003000 | 0x10003FFF | Option Bytes |  |  | Y |  |  |  |
| 0x10002000 | 0x10002FFF | Factory Bytes |  |  | Y |  |  |  |
| 0x10001C00 | 0x10001FFF | OTP |  |  | Y |  |  |  |
| 0x10000000 | 0x10001BFF | BootLoader |  | Y**(1)** | Y**(1)** |  |  |  |
| 0x08040000 | 0x0FFFFFFF | Reserved |  |  |  |  |  |  |
| 0x08000000 | 0x0803FFFF | Flash Main | Y | Y | Y |  | Y | Y |
| 0x00040000 | 0x07FFFFFF | Reserved |  |  |  |  |  |  |
| 0x00000000 | 0x0003FFFF | Flash Main/BootLoader/  System SRAM**(1)** | Y | Y | Y |  |  |  |

#### 7.2 Memory Mapping

#### The Memory Mapping table is shown below. The bytes are coded in memory in Little Endian format, i.e. the least significant byte is in the lowest address.

##### Table 7-2 Memory Mapping

##### 

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Category | Start Address | End Address | Description | Size |
| SYSTEM | 0xE0100000 | 0xFFFFFFFFF | Reserved |  |
| PPB | 0xE0000000 | 0xE00FFFFF | ARM STAR peripherals |  |
| EXT PERIPHERAL | 0xA0000000 | 0xDFFFFFFF | Reserved |  |
| EXT SRAM | 0x70000000 | 0x9FFFFFFF | Reserved |  |
| 0x60000000 | 0x6FFFFFFF | Qspi Flash Bank | 256MB |
| PERIPHERAL | 0x50000000 | 0x5FFFFFFF | Reserved |  |
| 0x40030000 | 0x4FFFFFFF | AHB1 SFR |  |
| 0x40020000 | 0x4002FFFF | AHB0 SFR |  |
| 0x40010000 | 0x4001FFFF | APB1 SFR |  |
| 0x40000000 | 0x4000FFFF | APB0 SFR |  |
| SRAM | 0x30000400 | 0x3FFFFFFF | Reserved |  |
| 0x30000000 | 0x300003FF | Retention SRAM | 1KB |
| 0x20010000 | 0x2FFFFFFF | Reserved |  |
| 0x20000000 | 0x2000FFFF | System SRAM | 64KB |
| CODE | 0x18010000 | 0x1FFFFFFF | Reserved |  |
| 0x18000000 | 0x1800FFFF | System SRAM | 64KB |
| 0x10004000 | 0x17FFFFFF | Reserved |  |
| 0x10003000 | 0x10003FFF | Option Bytes | 4KB |
| 0x10002000 | 0x10002FFF | Factory Bytes | 4KB |
| 0x10001C00 | 0x10001FFF | OTP | 1KB |
| 0x10000000 | 0x10001BFF | BootLoader | 7KB |
| 0x08040000 | 0x0FFFFFFF | Reserved |  |
| 0x08000000 | 0x0803FFFF | Flash Main | 256KB |
| 0x00040000 | 0x07FFFFFF | Reserved |  |
| 0x00000000 | 0x0003FFFF | Flash Main/BootLoader/  System SRAM**(1)** | 256KB |

***(1)*** *The memory corresponding to address 0x00000000 is determined by the boot mode.*

##### 7.2.1 AHB0 SFR

See the table below for AHB0 SFR Internal Address Mapping.

**Table 7-3 AHB0 SFR Internal Address mapping**

|  |  |  |  |
| --- | --- | --- | --- |
| Start Address | End Address | Description | Size |
| 0x40025000 | 0x4002FFFF | Reserved |  |
| 0x40024000 | 0x40024FFF | DMAC1 | 4KB |
| 0x40023000 | 0x40023FFF | DMAC0 | 4KB |
| 0x40022000 | 0x40022FFF | CRC | 4KB |
| 0x40021000 | 0x40021FFF | QSPI | 4KB |
| 0x40020000 | 0x40020FFF | EFC | 4KB |

##### 7.2.2 AHB1 SFR

See the table below for AHB1 SFR Internal Address Mapping.

**Table 7-4 AHB1 SFR Internal Address mapping**

|  |  |  |  |
| --- | --- | --- | --- |
| Start Address | End Address | Description | Size |
| 0x40034000 | 0x4003FFFF | Reserved |  |
| 0x40033000 | 0x40033FFF | RNGC | 4KB |
| 0x40030000 | 0x40032FFF | SAC | 12KB**(1)(2)** |

1. *Low 8KB is ARAM space, and high 4KB is for registers.*
2. *ARAM space can only be accessed in word.*

##### 7.2.3 APB0 SFR

See the table below for APB0 SFR Internal Address Mapping.

**Table 7-5 APB0 SFR** Internal Address **mapping**

|  |  |  |  |
| --- | --- | --- | --- |
| Start Address | End Address | Description | Size |
| 0x4000f000 | 0x4000FFFF | SEC | 4KB |
| 0x4000e000 | 0x4000EFFF | RTC | 4KB |
| 0x4000d800 | 0x4000DFFF | LPTIM1 | 2KB |
| 0x4000d000 | 0x4000D7FF | LPTIM0 | 2KB |
| 0x4000c000 | 0x4000CFFF | BASICTIM0 | 4KB |
| 0x4000b000 | 0x4000BFFF | GPTIM2 | 4KB |
| 0x4000a000 | 0x4000AFFF | GPTIM0 | 4KB |
| 0x40009000 | 0x40009FFF | LORAC | 4KB |
| 0x40008000 | 0x40008FFF | AFEC | 4KB |
| Start Address | End Address | Description | Size |
| 0x40007000 | 0x40007FFF | I2C0 | 4KB |
| 0x40006000 | 0x40006FFF | SSP0 | 4KB |
| 0x40005000 | 0x40005FFF | LPUART | 4KB |
| 0x40004000 | 0x40004FFF | UART1 | 4KB |
| 0x40003000 | 0x40003FFF | UART0 | 4KB |
| 0x40002000 | 0x40002FFF | I2S | 4KB |
| 0x40001800 | 0x40001FFF | PWR | 2KB |
| 0x40001000 | 0x400017FF | SYSCFG | 2KB |
| 0x40000000 | 0x40000FFF | RCC | 4KB |

##### 7.2.4 APB1 SFR

See the table below for APB1 SFR Internal Address Mapping.

**Table 7-6 APB1 SFR Internal Address mapping**

|  |  |  |  |
| --- | --- | --- | --- |
| Start Address | End Address | Description | Size |
| 0x4001fc00 | 0x4001FFFF | PortD | 1KB |
| 0x4001f800 | 0x4001FBFF | PortC | 1KB |
| 0x4001f400 | 0x4001F7FF | PortB | 1KB |
| 0x4001f000 | 0x4001F3FF | PortA | 1KB |
| 0x4001e000 | 0x4001EFFF | WWDG | 4KB |
| 0x4001d000 | 0x4001DFFF | IWDG | 4KB |
| 0x4001c000 | 0x4001CFFF | BASICTIM1 | 4KB |
| 0x4001b000 | 0x4001BFFF | GPTIM3 | 4KB |
| 0x4001a000 | 0x4001AFFF | GPTIM1 | 4KB |
| 0x40019000 | 0x40019FFF | DACCTRL | 4KB |
| 0x40018000 | 0x40018FFF | LCDCTRL | 4KB |
| 0x40017000 | 0x40017FFF | ADCCTRL | 4KB |
| 0x40016000 | 0x40016FFF | Reserved | 4KB |
| 0x40015000 | 0x40015FFF | I2C2 | 4KB |
| 0x40014000 | 0x40014FFF | I2C1 | 4KB |
| 0x40013000 | 0x40013FFF | SSP2 | 4KB |
| 0x40012000 | 0x40012FFF | SSP1 | 4KB |
| 0x40011000 | 0x40011FFF | UART3 | 4KB |
| 0x40010000 | 0x40010FFF | UART2 | 4KB |

##### 7.3 SRAM

The SRAM in ASR6601 includes system SRAM, retention SRAM and SAC SRAM. SAC SRAM

only supports word access, and system SRAM and retention SRAM support word, halfword,

and byte access.

#### 7.4 Boot Mode

The boot mode can be configured by the levels of BOOT0 pin (GPIO02) and the data in the

Flash.

##### Table 7-7 ASR6601 Boot Mode Configuration

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| DEBUG\_  LEVEL | USE\_FLASH\_  BOOT0 | FLASH\_  BOOT0 | BOOT0  PIN | FLASH\_  BOOT1 | MAIN\_FLASH\_  EMPTY | Boot Config |
| 2 | X | X | X | X | X | Boot from Flash Main |
| <2 | 0 | X | 0 | X | 0 | Boot from Flash Main |
| <2 | 0 | X | 0 | X | 1 | Boot from Flash Bootloader |
| <2 | 0 | X | 1 | 1 | X | Boot from Flash Bootloader |
| <2 | 0 | X | 1 | 0 | X | Boot from System SRAM |
| <2 | 1 | 1 | X | X | 0 | Boot from Flash Main |
| <2 | 1 | 1 | X | X | 1 | Boot from Flash Bootloader |
| <2 | 1 | 0 | X | 1 | X | Boot from Flash Bootloader |
| <2 | 1 | 0 | X | 0 | X | Boot from System SRAM |

DebugLevel, UseFlashBoot0, FlashBoot0 and FlashBoot1 is the information area of the

Flash, they can be modified according to the application.

MainFlashEmpty is determined by the data of address 0 in the Flash Main area. If the data

in the address 0 of Flash Main area is 0xFFFFFFFF, the value of MainFlashEmpty is 1,

otherwise the value of MainFlashEmpty is 0. BOOT0 pin is GPIO02 in the package.

The boot mode is selected according to the configurations when the system is in these

status: first powered up, exit the Standby mode or reset.

#### 7.5 System Configuration Registers

Base Address:0x40001000

**Table 7-8 SYSCFG Register Summary**

|  |  |  |
| --- | --- | --- |
| Register | Offset | Description |
| SYSCFG\_CR0 | 0x000 | Control Register 0, DMA handshake |
| SYSCFG\_CR1 | 0x004 | Control Register 1, DMA handshake |
| SYSCFG\_CR2 | 0x008 | Control Register 2 |
| SYSCFG\_CR3 | 0x00C | Control Register 3,Low power debug connection control |
| SYSCFG\_CR4 | 0x010 | Control Register 4 |
| SYSCFG\_CR5 | 0x014 | Control Register 5 |
| SYSCFG\_CR6 | 0x018 | Control Register 6,secure lock control |
| SYSCFG\_CR7 | 0x01C | Control Register 7,secure lock control |
| SYSCFG\_CR8 | 0x020 | Control Register 8,QSPI memory encryption key |
| SYSCFG\_CR9 | 0x024 | Control Register 9,QSPI REMAP |
| SYSCFG\_CR10 | 0x028 | Control Register 10 |

##### 7.5.1 SYSCFG\_CR0

Offset:0x000

Reset Value:0x00000000

|  |  |  |  |
| --- | --- | --- | --- |
| **31-30** | **29-24** | **23-22** | **21-16** |
| RESERVED | DMAC0\_HANDSHAKE0\_SEL | RESERVED | DMAC0\_HANDSHAKE1\_SEL |
| r | r/w | r | r/w |
| **15-14** | **13-8** | **7-6** | **5-0** |
| RESERVED | DMAC0\_HANDSHAKE2\_SEL | RESERVED | DMAC0\_HANDSHAKE3\_SEL |
| r | r | r | r/w |

**Bits 31-30 RESERVED:** Must be kept, and can't be modified.

**Bits 29-24 DMAC0\_HANDSHAKE0\_SEL:**DMAC0 HANDSHAKE0 selection. For details, see Table*7-9 DMA Request MUX.*

**Bits 23-22 RESERVED:** Must be kept, and can't be modified.

**Bits 21-16 DMAC0\_HANDSHAKE1\_SEL:**DMAC0 HANDSHAKE1 selection. For details, see Table*7-9 DMA Request MUX.*

**Bits 15-14 RESERVED:** Must be kept, and can't be modified.

**Bits 13-8 DMAC0\_HANDSHAKE2\_SEL:**DMAC0 HANDSHAKE2 selection. For details, see Table*7-9 DMA Request MUX.*

**Bits 7-6 RESERVED:** Must be kept, and can't be modified.

**Bits 5-0 DMAC0\_HANDSHAKE3\_SEL:**DMAC0 HANDSHAKE3 selection. For details, see Table*7-9 DMA Request MUX.*

##### 7.5.2 SYSCFG\_CR1

Offset:0x004

Reset Value:0x00000000

|  |  |  |  |
| --- | --- | --- | --- |
| **31-30** | **29-24** | **23-22** | **21-16** |
| RESERVED | DMAC1\_HANDSHAKE0\_SEL | RESERVED | DMAC1\_HANDSHAKE1\_SEL |
| r | r/w | r | r/w |
| **15-14** | **13-8** | **7-6** | **5-0** |
| RESERVED | DMAC1\_HANDSHAKE2\_SEL | RESERVED | DMAC1\_HANDSHAKE3\_SEL |
| r | r | r | r/w |

**Bits 31-30 RESERVED:** Must be kept, and can't be modified.

**Bits 29-24 DMAC1\_HANDSHAKE0\_SEL:**DMAC1 HANDSHAKE0 selection. For details, see Table*7-9 DMA Request MUX.*

**Bits 23-22 RESERVED:**Must be kept, and can't be modified.

**Bits 21-16 DMAC1\_HANDSHAKE1\_SEL:**DMAC1 HANDSHAKE1 selection. For details, see Table*7-9 DMA Request MUX.*

**Bits 15-14 RESERVED:** Must be kept, and can't be modified.

**Bits 13-8 DMAC1\_HANDSHAKE2\_SEL:**DMAC1 HANDSHAKE2 selection. For details, see Table*7-9 DMA Request MUX.*

**Bits 7-6 RESERVED:** Must be kept, and can't be modified.

**Bits 5-0 DMAC1\_HANDSHAKE3\_SEL:**DMAC1 HANDSHAKE3 selection. For details, see Table*7-9 DMA Request MUX.*

##### 7.5.3 SYSCFG\_CR2

Offset:0x008

Reset Value:0x00000000

|  |  |  |  |
| --- | --- | --- | --- |
| **31** | **30** | **29-28** | **27** |
| RESERVED | SYSCFG\_HALTED\_IPTI  M1\_EN | RESERVED | SYSCFG\_HALTED\_LPT  IM0\_EN |
| r | r/w | r | r/w |
| **26** | **25** | **24** | **23** |
| SYSCFG\_HALTED\_IW  DG\_EN | SYSCFG\_HALTED\_WW  DG\_EN | SYSCFG\_HALTED\_GP  TIM0\_EN | SYSCFG\_HALTED\_GP  TIM1\_EN |
| r/w | r/w | r/w | r/w |
| **22** | **21** | **20** | **19** |
| SYSCFG\_HALTED\_GP  TIM2\_EN | SYSCFG\_HALTED\_GP  TIM3\_EN | SYSCFG\_HALTED\_BA  SICTIM0\_EN | SYSCFG\_HALTED\_BA  SICTIM1\_EN |
| r/w | r/w | r/w | r/w |
| **18** | **17** | **16-12** | |
| QSPI\_MEM\_ENCRYPT  \_EN | QSPI\_REMAP\_ENABLE | RESERVED | |
| r/w | r/w | r | |
| **11** | **10** | **9-8** | |
| CPU\_STCALIB\_SKEW | SYSCFG\_DBG\_SLEEP | RESERVED | |
| r/w | r/w | r | |
| **7** | **6** | **5** | **4** |
| UART0\_DMA\_CLR\_SEL | UART1\_DMA\_CLR\_SEL | UART2\_DMA\_CLR\_SEL | UART3\_DMA\_CLR\_SEL |
| r/w | r/w | r/w | r/w |
| **3** | **2** | **1** | **0** |
| SSP0\_DMA\_CLR\_SEL | SSP1\_DMA\_CLR\_SEL | SSP2\_DMA\_CLR\_SEL | SSP\_AFEC\_DMA\_CLR  \_SEL |
| r/w | r/w | r/w | r/w |

**Bits 31 RESERVED:** Must be kept, and can't be modified.

**Bit 30 SYSCFG\_HALTED\_LPTIM1\_EN:** Whether the LPTIM1 counter is stopped when the core is halted.

* 0: Disable
* 1: Enable

**Bits 29-28 RESERVED:** Must be kept, and can't be modified.

**Bit 27 SYSCFG\_HALTED\_LPTIM0\_EN:** Whether the LPTIM0 counter is stopped when the core is halted.

* 0: Disable
* 1: Enable

**Bit 26 SYSCFG\_HALTED\_IWDG\_EN:** Whether the independent watchdog counter is stopped

when the core is halted

* 0: Disable
* 1: Enable

**Bit 25 SYSCFG\_HALTED\_WWDG\_EN:** Whether the window watchdog counter is stopped when the core is halted

* 0: Disable
* 1: Enable

**Bit 24 SYSCFG\_HALTED\_GPTIM0\_EN:** Whether the GPTIM0 counter is stopped when the core is halted

* 0: Disable
* 1: Enable

**Bit 23 SYSCFG\_HALTED\_GPTIM1\_EN:** Whether the GPTIM1 counter is stopped when the core is halted

* 0: Disable
* 1: Enable

**Bit 22 SYSCFG\_HALTED\_GPTIM2\_EN:** Whether the GPTIM2 counter is stopped when the core is halted

* 0: Disable
* 1: Enable

**Bit 21 SYSCFG\_HALTED\_GPTIM3\_EN:** Whether the GPTIM3 counter is stopped when the core is halted

* 0: Disable
* 1: Enable

**Bit 20 SYSCFG\_HALTED\_BASICTIM0\_EN:** Whether the BASICTIM0 counter is stopped when the core is halted

* 0: Disable
* 1: Enable

**Bit 19 SYSCFG\_HALTED\_BASICTIM1\_EN:** Whether the BASICTIM1 counter is stopped when the core is halted

* 0: Disable
* 1: Enable

**Bit 18 QSPI\_MEM\_ENCRYPT\_EN:** QSPI memory encryption enable

* 0: Disable
* 1: Enable

**Bit 17 QSPI\_REMAP\_ENABLE:** QSPI remap function enable

* 0: Disable
* 1: Enable

**Bits 16-12 RESERVED:** Must be kept, and can't be modified.

**Bit 11 CPU\_STCALIB\_SKEW:** CPU SysTick skew configuration. Affects CPU STCALIB[24]Bits.

* 0: no skew
* 1: skew

**Bit 10 SYSCFG\_DBG\_SLEEP:** Sleep Whether debug connections are allowed in low power mode.  
Only used during debugging, it will affect the implementation of sleep low-power mode.

* 0: not allowed
* 1: allowed

**Bits 9-8 RESERVED:** Must be kept, and can't be modified.

**Bit 7 UART0\_DMA\_CLR\_SEL:** UART0 DMA\_CLR signal selection.  
It is recommended to configure it to 1 to improve DMAC data transfer efficiency. UART IP uses the synchronized DMA\_CLR signal by default.

* 0: use the DMA\_CLR signal after 2 cycles
* 1: directly use the DMA\_CLR signal output by DMAC

**Bit 6** UART1\_DMA\_CLR\_SEL: UART1 DMA\_CLR signal selection.  
It is recommended to configure it to 1 to improve DMAC data transfer efficiency. UART IP uses the synchronized DMA\_CLR signal by default.

* 0: use the DMA\_CLR signal after 2 cycles
* 1: directly use the DMA\_CLR signal output by DMAC

**Bit 5** UART2\_DMA\_CLR\_SEL: UART2 DMA\_CLR signal selection.  
It is recommended to configure it to 1 to improve DMAC data transfer efficiency. UART IP uses the synchronized DMA\_CLR signal by default.

* 0: use the DMA\_CLR signal after 2 cycles
* 1: directly use the DMA\_CLR signal output by DMAC

**Bit 4** UART3\_DMA\_CLR\_SEL: UART3 DMA\_CLR signal selection.  
It is recommended to configure it to 1 to improve DMAC data transfer efficiency. UART IP uses the synchronized DMA\_CLR signal by default.

* 0: use the DMA\_CLR signal after 2 cycles
* 1: directly use the DMA\_CLR signal output by DMAC

**Bit 3 SSP0\_DMA\_CLR\_SEL:** SSP0 DMA\_CLR signal selection

It is recommended to set this bit to improve DMAC transfer efficiency. SSP module uses the synchronized DMA\_CLR signal by default.

* 0: use the DMA\_CLR signal after 2 cycles
* 1: directly use the DMA\_CLR signal output by DMAC

**Bit 2 SSP1\_DMA\_CLR\_SEL:** SSP1 DMA\_CLR signal selection

It is recommended to set this bit to improve DMAC transfer efficiency. SSP module uses the synchronized DMA\_CLR signal by default.

* 0: use the DMA\_CLR signal after 2 cycles
* 1: directly use the DMA\_CLR signal output by DMAC

**Bit 1 SSP2\_DMA\_CLR\_SEL:** SSP1 DMA\_CLR signal selection

It is recommended to set this bit to improve DMAC transfer efficiency. SSP module uses the synchronized DMA\_CLR signal by default.

* 0: use the DMA\_CLR signal after 2 cycles
* 1: directly use the DMA\_CLR signal output by DMAC

**Bit 0 SSP\_AFEC\_DMA\_CLR\_SEL:** AFECSSP DMA\_CLR signal selection

It is recommended to set this bit to improve DMAC transfer efficiency. SSP module uses the synchronized DMA\_CLR signal by default.

* 0: use the DMA\_CLR signal after 2 cycles
* 1: directly use the DMA\_CLR signal output by DMAC

##### 7.5.4 SYSCFG-CR3

Offset: 0x00C

Reset Value: 0x00000000

This register is in the AON domain.

|  |  |  |
| --- | --- | --- |
| **31-2** | **1** | **0** |
| RESERVED | SYSCFG\_DBG\_STOP | SYSCFG\_DBG\_STANDBY |
| r | r/w | r/w |

**Bits 31-2 RESERVED:** Must be kept, and can't be modified.

**Bit 1 SYSCFG\_DBG\_STOP:** Whether to allow a debug connection in Stop mode

It is only used in debug and it will affect the implementation of Stop mode.

* 0: not allowed
* 1: allowed

**Bit 0 SYSCFG\_DBG\_STANDBY:** Whether to allow a debug connection in Standby mode. It is only used in debug and it will affect the implementation of Standby mode.

* 0: not allowed  1: allowed

##### 7.5.5 SYSCFG\_CR4

Offset: 0x010

Reset Value:0x00000000

This register is in the AON domain.

|  |  |
| --- | --- |
| **31** | **30-0** |
| SYSCFG\_CR4\_REG | USER-DEFINED |
| r/w | r/w |

**Bit 31:** LPTIM1 IN2 remapping enable.

* 0: disabled, LPTIM1\_IN2 is determined by GPIO AFR
* 1: enabled, LPTIM1\_IN2 is derived from LPTIM0\_IN1

**Bits 30-0:** These bits are user-defined and can be used to store a small amount of data by software.

##### 7.5.6 SYSCFG\_CR5

Offset: 0x014

Reset Value: 0x00000000

This register is in the AON domain.

|  |
| --- |
| **31-0** |
| SYSCFG\_CR5\_REG |
| r/w |

**Bits 31-0 SYSCFG\_CR5\_REG:** These bits are user-defined and can be used to store a small amount of data by software.

##### 7.5.7 SYSCFG\_CR6

Offset: 0x018

Reset Value: 0x00000000

|  |  |  |  |
| --- | --- | --- | --- |
| **31-16** | **15** | **14-5** | **4** |
| RESERVED | RNGC\_SECURE\_LOCK | ANALOG\_MAIN\_SECU  RE\_LOCK | RESERVED |
| r | r/w | r/w | r |
| **3** | **2** | **1** | **0** |
| SEC\_SECURE\_LOCK | SAC\_SECURE\_LOCK | DMAC0\_SLAVE\_SECU  RE\_LOCK | DMAC0\_MASTER\_SEC  URE\_LOCK |
| r/w | r/w | r/w | r/w |

**Bits 31-16 RESERVED:** Must be kept, and can't be modified.

**Bit 15 RNGC\_SECURE\_LOCK:** RNGC security lock bits.

* 0: no security lock
* 1: security lock

**Bits 14-5 ANALOG\_MAIN\_SECURE\_LOCK:** Security lock for main domain configuration of AFEC

1. Correspond to VD
   * 0: no security lock
   * 1: security lock
2. Correspond to TD
   * 0: no security lock
   * 1: security lock
3. Correspond to LD
   * 0: no security lock
   * 1: security lock
4. Correspond to FD24M
   * 0: no security lock
   * 1: security lock
5. Correspond to FD32M
   * 0: no security lock
   * 1: security lock
6. Correspond to RNG
   * 0: no security lock
   * 1: security lock
7. Correspond to TEST
   * 0: no security lock
   * 1: security lock

[14:12] unused

* + 0: no security lock
  + 1: security lock

**Bit 4 RESERVED:** Must be kept, and can't be modified.

**Bit 3 SEC\_SECURE\_LOCK:** SEC security lock bits.

* + 0: no security lock
  + 1: security lock

**Bit 2 SAC\_SECURE\_LOCK:** SAC security lock bits.

* + 0: no security lock
  + 1: security lock

**Bit 1 DMAC0\_SLAVE\_SECURE\_LOCK:** DMAC0 slave interface security lock bits.

* + 0: no security lock
  + 1: security lock

**Bit 0 DMAC0\_MASTER\_SECURE\_LOCK:** DMAC0 master interface security lock bits.

* + 0: no security lock
  + 1: security lock

##### 7.5.8 SYSCFG\_CR7

Offset: 0x01C

Reset Value: 0x00000000

This register is in the AON domain.

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **31-15** | **14-5** | | **4** | |
| RESERVED | ANALOG\_AON\_SECURE\_LOCK | | RTC\_CALENDAR\_SECURE\_LOCK | |
| r | r/w | | r/w | |
| **3** | **2** | **1** | | **0** |
| RTC\_WAKEUP2\_SEC  URE\_LOCK | RTC\_WAKEUP1\_SECU  RE\_LOCK | RTC\_WAKEUP0\_SECU  RE\_LOCK | | RTC\_TAMPER\_SECUR  E\_LOCK |
| r/w | r/w | r/w | | r/w |

**Bits 31-16 RESERVED:** Must be kept, and can't be modified.

**Bits 14-5 ANALOG\_AON\_SECURE\_LOCK:** Security lock for AON domain configuration of AFEC.

[5] Correspond to LPLDO

* + 0: no security lock
  + 1: security lock

1. Correspond to RCO3.6M
   * 0: no security lock
   * 1: security lock
2. Correspond to PWRSW
   * 0: no security lock
   * 1: security lock
3. Correspond to RCO32K
   * 0: no security lock
   * 1: security lock
4. Correspond to XO32K
   * 0: no security lock
   * 1: security lock
5. Correspond to LDO12
   * 0: no security lock
   * 1: security lock
6. Correspond to FD32K
   * 0: no security lock
   * 1: security lock

[14:12] unused

* + 0: no security lock
  + 1: security lock

**Bit 4 RTC\_CALENDAR\_SECURE\_LOCK:** RTC Calendar security lock configuration.

* + 0: no security lock
  + 1: security lock

**Bit 3 RTC\_WAKEUP2\_SECURE\_LOCK:** RTC Wakeup2 security lock configuration.

* + 0: no security lock
  + 1: security lock

**Bit 2 RTC\_WAKEUP1\_SECURE\_LOCK:** RTC Wakeup1 security lock configuration.

* + 0: no security lock
  + 1: security lock

**Bit 1 RTC\_WAKEUP0\_SECURE\_LOCK:** RTC Wakeup0 security lock configuration.

* + 0: no security lock
  + 1: security lock

**Bit 0 RTC\_TAMPER\_SECURE\_LOCK:** RTC Tamper security lock configuration.

* + 0: no security lock
  + 1: security lock

##### 7.5.9 SYSCFG\_CR8

Offset: 0x020

Reset Value: 0x00000000

|  |
| --- |
| **31-0** |
| QSPI\_MEM\_ENCRYPT\_KEY |
| r/w |

**Bits 31-0 QSPI\_MEM\_ENCRYPT\_KEY:** QSPI memory encryption key.

##### 7.5.10 SYSCFG\_CR9

Offset:0x024

Reset Value:0x00000000

|  |  |  |
| --- | --- | --- |
| **31-28** | **27-14** | **13-0** |
| RESERVED | QSPI\_REMAP\_SRC\_ADDR | QSPI\_REMAP\_DST\_ADDR |
| r | r/w | r/w |

**Bits 31-28 RESERVED:**Must be kept, and can't be modified.

**Bits 27-14 QSPI\_REMAP\_SRC\_ADDR:** QSPI remap source address, aligned in 1KB.

**Bits 13-0** QSPI\_REMAP\_DST\_ADDR: QSPI remap destination address, aligned in 1KB.

##### 7.5.11 SYSCFG\_CR10

Offset:0x028

Reset Value:0x00000000

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **31-24** | **23** | **22** | **21-15** | **14** | **13-0** |
| RESERVED | I2S\_WS\_SEL | I2S\_WS\_EN | I2S\_WS\_LEN | I2S\_MODE\_SEL | QSPI\_REMAP\_  SIZE |
| r | r/w | r/w | r/w | r/w | r/w |

**Bits 31-24 RESERVED:** Must be kept, and can't be modified.

**Bit 23 I2S\_WS\_SEL:** I2S output delay enable.

* 0: Disable
* 1: Enable

***Notice:*** *This bit can only be configured when the I2S acts as master interface. When enabled, the WS signal is output one cycle later than the data transmission.*

**Bit 22 I2S\_WS\_EN:** I2S WS Enable.

* 0: Disable
* 1: Enable

***Notice:*** *This bit can only be configured when the I2S acts as master interface. When enabled, the WS signal is generated based on the I2S\_WS\_LEN configuration.*

**Bits 21-15 I2S\_WS\_LEN:** I2S master interface resolution configuration.

N: WS frequency=I2S interface clock frequency/[(N+1)\*2]

The I2S interface clock frequency is determined by the I2S\_CLK\_DIV and I2S\_CLK\_SEL in RCC\_CR3 and RCC\_CR2 registers.

**Bit 14 I2S\_MODE\_SEL:** I2S mode select.

* 0: slave mode
* 1: master mode

***Notice:*** In addition to this Register, you also need to configure I2S\_CLK\_DIV and I2S\_CLK\_SEL in RCC\_CR3 and RCC\_CR2 registers, as well as GPIO multiplexing configuration.

**Bit 13-0 QSPI\_REMAP\_SIZE:** Address space for QSPI remapping, aligned in 1KB.

##### 7.6 DMA Request MUX

**Table 7-9 DMA Request MUX**

|  |  |
| --- | --- |
| No. | Source |
| 63 |  |
| 62 |  |
| 61 |  |
| 60 |  |
| 59 |  |
| 58 |  |
| 57 |  |
| 56 |  |
| 55 |  |
| 54 |  |
| 53 | basictim0\_up |
| 52 | basictim1\_up |
| 51 | gptim3\_up |
| 50 | gptim3\_trg |
| 49 | gptim3\_ch0 |
| 48 | gptim3\_ch1 |
| 47 | gptim2\_up |
| 46 | gptim2\_trg |
| 45 | gptim2\_ch0 |
| 44 | gptim2\_ch1 |
| 43 | gptim1\_up |
| 42 | gptim1\_trg |
| 41 | gptim1\_ch0 |
| 40 | gptim1\_ch1 |
| 39 | gptim1\_ch2 |
| 38 | gptim1\_ch3 |
| 37 | gptim0\_up |
| 36 | gptim0\_trg |
| 35 | gptim0\_ch0 |
| 34 | gptim0\_ch1 |
| 33 | gptim0\_ch2 |
| 32 | gptim0\_ch3 |
| 31 | uart0\_rx |
| 30 | uart0\_tx |
| 29 | uart1\_rx |
| 28 | uart1\_tx |
| 27 | uart2\_rx |
| No. | Source |
| 26 | uart2\_tx |
| 25 | uart3\_rx |
| 24 | uart3\_tx |
| 23 | lpuart\_rx |
| 22 | lpuart\_tx |
| 21 | ssp0\_rx |
| 20 | ssp0\_tx |
| 19 | ssp1\_rx |
| 18 | ssp1\_tx |
| 17 | ssp2\_rx |
| 16 | ssp2\_tx |
| 15 | i2c0\_rx |
| 14 | i2c0\_tx |
| 13 | i2c1\_rx |
| 12 | i2c1\_tx |
| 11 | i2c2\_rx |
| 10 | i2c2\_tx |
| 9 |  |
| 8 |  |
| 7 | adcctrl |
| 6 | dacctrl |
| 5 | lorac\_rx |
| 4 | lorac\_tx |
| 3 |  |
| 2 |  |
| 1 |  |
| 0 |  |

### 8. Reset and clock control (RCC)

#### 8.1 Reset

There are four types of reset: external reset, power reset, system reset and low-power reset.

##### 8.1.1 External Reset

The external reset is triggered by RSTN IO input (active at low level).

The external reset is used to reset all digital logic.

##### 8.1.2 Power-on Reset

The power-on reset is generated by the BOR (Brownout reset) circuitry. The BOR circuitry

monitors VBAT to ensure that the internal reset is released when the voltage is greater than

1.8V.

Power-on reset is used to reset all digital logic.

##### 8.1.3 System Reset

System reset sources include IWDG Reset, WWDG Reset, Option Byte Load Reset, Software

Reset, Sec Reset, Power-on Reset, and External Reset.

*  IWDG Reset: generated by the IWDG module for exception recovery
*  WWDG Reset: generated by the WWDG module for exception recovery
*  Option Byte Load Reset: generated by the EFC module and used to start option byte
* reloading
*  Software Reset: generated by the CPU
*  Sec Reset: generated by the Sec module and used for system reset after security alarm
* The system reset is used to reset most of the data logic in the Main domain excluding the Reset
* Source Status Register (*RCC\_RST\_SR*) which is used to record the latest system reset source.

##### 8.1.4 Low-power Reset

The low-power reset is generated by the low-power state machine and is used to reset the logic

of the main domain when the CPU exits Standby or Stop3 mode.

The low-power reset is used to reset all digital logic in the main domain.

#### 8.2 Clock

System clock structure:

iwdg, gpio

**Figure 8-1 Clock Tree**

##### 8.2.1 SYS\_CLK

RCO48M divided by 2, RCO32K, XO32K, PLL, XO24M, XO32M, RCO3.6M or RCO48M can

be selected as the SYS\_CLK clock source. The default is RCO48M divided by 2.

 RCO48M (48MHz) is generated from the internal clock circuit.

 RCO32K (32kHz) is generated from the internal clock circuit.

 RCO3.6M (3.6MHz) is generated from the internal clock circuit.

 XO32K (32.768kHz) is generated from an external crystal oscillator.

 XO32M (32MHz) is generated from an external crystal oscillator.

 XO24M (24MHz) is generated from an external crystal oscillator.

 PLL is an internal clock circuit, RCO48M, XO32M, XO24M or RCO3.6M can be selected

as PLL clock source, and the PLL clock output supports up to 48MHz

AHB bus clock HCLK is generated from SYS\_CLK divided by 2^N (N ranges from 0 to 9).

The system includes two APB buses, the APB bus clock PCLK1 and PCLK2 are generated

from HCLK divided by 2^M (M ranges from 0 to 4). The clock division factor for the two APB

buses can be configured independently.

##### 8.2.2 **Clocks for the Modules**

##### The clocks for the modules consist of bus clocks and interface clocks.

The bus clock is generated by HCLK or PCLK gating and is used for modules to access bus.

In addition to a bus clock, some modules also have an independent interface clock, which is

different from the bus clock, and is used to realize the function of the module.

The interface clock source for each module is selectable by software:

 LPTIM: PCLK0, RCO3.6M, XO32K, RCO32K, IO input clock

 LCDCTRL: XO32K, RCO32K, RCO3.6M

 LPUART: XO32K, RCO32K, RCO3.6M

 RTC: XO32K, RCO32K

 IWDG: XO32K, RCO32K

 UART: PCLK0/PCLK1, RCO3.6M, XO32K, XO24M

 ADCCTRL: PCLK1, SYS\_CLK, PLL, RCO48M

 I2S: PCLK0, XO24M, PLL, XO32M, IO input clock

 QSPI: HCLK, SYS\_CLK, PLL

ADCCTRL and I2S also support interface clock division, which is used to generate low frequency interface clocks.

##### 8.2.3 **Clock-out Capability** (MCO)

The microcontroller clock output (MCO) capability allows the internal clock to be output by IO.

MCO clock source can be RCO32K, XO32K, RCO3.6M, XO24M, XO32M, RCO48M, PLL or

SYS\_CLK.

The clock can be output with a frequency divided by software configuration.

#### 8.3 RCC **Registers**

#### 

Base Address: 0x40000000

##### Table 8-1 RCC Register Summary

|  |  |  |
| --- | --- | --- |
| Register | Offset | Description |
| RCC\_CR0 | 0x000 | Control Register 0 |
| RCC\_CR1 | 0x004 | Control Register 1, interface clock source selection |
| RCC\_CR2 | 0x008 | Control Register 2, interface clock source selection |
| RCC\_CGR0 | 0x00C | Module clock configure register 0 |
| RCC\_CGR1 | 0x010 | Module clock configure register 1 |
| RCC\_CGR2 | 0x014 | Module clock configure register 2 |
| RCC\_RST0 | 0x018 | Module reset control register 0 |
| RCC\_RST1 | 0x01C | Module reset control register 1 |
| RCC\_RST\_SR | 0x020 | System reset source status register |
| RCC\_RST\_CR | 0x024 | System reset source control register |
| RCC\_SR | 0x028 | Status register, RCC\_CGR2 configuration status |
| RCC\_SR1 | 0x02C | Status register 1, module clock configuration status |
| RCC\_CR3 | 0x030 | Control register 3, interface clock division |

##### 8.3.1 RCC\_CR0

Offset: 0x000

Reset Value: 0x00000000

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **31-26** | **25** | **24-22** | **21-19** | **18** |
| RESERVED | STCLKEN\_SEL | MCO\_CLK\_DIV\_NUM | MCO\_CLK\_SEL | MCO\_CLK\_OUT\_EN |
| r | r/w | r/w | r/w | r/w |
| **17-15** | **14-12** | **11-8** | **7-5** | **4-0** |
| PCLK1\_DIV | SYS\_CLK\_SEL | HCLK\_DIV | PCLK0\_DIV | RESERVED |
| r/w | r/w | r/w | r/w | r |

**Bits 31-26 RESERVED:** Must be kept, and can't be modified.

**Bits 25 STCLKEN\_SEL:** CPU SysTick clock source selection.

* 0:XO32K
* 1:RCO32K

**Bits 24-22 MCO\_CLK\_DIV\_NUM:** MCO division factor.

* <4: division factor is 1
* 4: division factor is 2
* 5: division factor is 4
* 6: division factor is 8
* 7: division factor is 16

***Notice*:** *Make sure to configure this bit when MCO\_CLK\_OUT\_EN=0. If the MCO\_CLK\_OUT\_EN bit is enabled, users must disable it by software first, wait for at least 2 current clock cycles or query the RCC\_SR1 register, and then configure the MCO division factor.*

**Bits 21-19 MCO\_CLK\_SEL:** MCO clock source selection.

* 0: RCO32K
* 1: XO32K
* 2: RCO3.6M
* 3: XO24M
* 4: XO32M
* 5: RCO48M
* 6: PLL
* 7: SYS\_CLK

***Notice*:** *Make sure to configure this bit when MCO\_CLK\_OUT\_EN=0. If the MCO\_CLK\_OUT\_EN bit is enabled, users must disable it by software first, wait for at least 2 current clock cycles or query the RCC\_SR1 register, and then configure the MCO clock source.*

**Bit 18 MCO\_CLK\_OUT\_EN:** MCO output enable.

* 0: Disable
* 1: Enable

**Bits 17-15 PCLK1\_DIV:** PCLK1 division factor.

* 0: PCLK1 clock frequency = HCLK clock frequency
* 1: PCLK1 clock frequency = 1/2 HCLK clock frequency
* 2: PCLK1 clock frequency = 1/4 HCLK clock frequency
* 3: PCLK1 clock frequency = 1/8 HCLK clock frequency
* >3: PCLK1 clock frequency = 1/16 HCLK clock frequency

**Bits 14-12 SYS\_CLK\_SEL:** SYS\_CLK clock source selection.

* 0: RCO48M divided by 2
* 1: RCO32K
* 2: XO32K
* 3: PLL
* 4: XO24M
* 5: XO32M
* 6: RCO3.6M
* 7: RCO48M

**Bits 11-8 HCLK\_DIV:** HCLK division factor.

* 0: HCLK clock frequency = SYS\_CLK clock frequency
* 1: HCLK clock frequency = 1/2 SYS\_CLK clock frequency
* 2: HCLK clock frequency = 1/4 SYS\_CLK clock frequency
* 3: HCLK clock frequency = 1/8 SYS\_CLK clock frequency
* 4: HCLK clock frequency = 1/16 SYS\_CLK clock frequency
* 5: HCLK clock frequency = 1/32 SYS\_CLK clock frequency
* 6: HCLK clock frequency = 1/64 SYS\_CLK clock frequency
* 7: HCLK clock frequency = 1/128 SYS\_CLK clock frequency
* 8: HCLK clock frequency = 1/256 SYS\_CLK clock frequency
* >8: HCLK clock frequency = 1/512 SYS\_CLK clock frequency

**Bits 7-5 PCLK0\_DIV:** PCLK0 division factor.

* 0: PCLK0 clock frequency = HCLK clock frequency
* 1: PCLK0 clock frequency = 1/2 HCLK clock frequency
* 2: PCLK0 clock frequency = 1/4 HCLK clock frequency
* 3: PCLK0 clock frequency = 1/8 HCLK clock frequency
* >3: PCLK0 clock frequency = 1/16 HCLK clock frequency

**Bits 4-0 RESERVED:** Must be kept, and can't be modified.

##### 8.3.2 RCC\_CR1

Offset: 0x004

Reset Value: 0x00000000

This register is in the AON domain.

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **31-12** | **11** | **10** | **9-8** | |
| RESERVED | LPTIM1\_EXT\_CLK\_SEL | LPTIM0\_EXT\_CLK\_SEL | LPTIM1\_CLK\_SEL | |
| r | r/w | r/w | r/w | |
| **7-6** | **5-4** | **3-2** | **1** | **0** |
| LPTIM0\_CLK\_SEL | LCDCTRL\_CLK\_SEL | LPUART\_CLK\_SEL | RTC\_CLK\_  SEL | IWDG\_CLK\_  SEL |
| r/w | r/w | r/w | r/w | r/w |

**Bits 31-12 RESERVED:** Must be kept, and can't be modified.

**Bit 11 LPTIM1\_EXT\_CLK\_SEL:** LPTIM1 interface clock source selection.

* 0: decided by the LPTIM1\_CLK\_SEL bit
* 1: use external clock from IN1

***Notice*:**

***1.*** *Make sure to configure this bit when LPTIM1\_CLK\_EN=0. If the LPTIM1\_CLK\_EN bit is enabled, the user must disable it by software first, wait or at least 2 current clock cycles or query the RCC\_SR1 register, and then configure the LPTIM1 interface clock source.*

*2. This bit and the LPTIM1\_CLK\_SEL bit jointly determine the LPTIM1 interface clock source.*

**Bit 10 LPTIM0\_EXT\_CLK\_SEL:** LPTIM0 interface clock source selection

* 0: decided by the LPTIM0\_CLK\_SEL bit
* 1: use external clock from IN1

***Notice*:**

*1. Make sure to configure this bit when LPTIM0\_CLK\_EN=0. If the LPTIM0\_CLK\_EN bit is enabled, the user must disable it by software first, wait for at least 2 current clock cycles or query the RCC\_SR1 register, and then configure the LPTIM0 interface clock source.*

*2. This bit and the LPTIM0\_CLK\_SEL bit jointly determine the LPTIM0 interface clock source.*

**Bits 9-8 LPTIM1\_CLK\_SEL:** LPTIM1 interface clock source selection

* 0: PCLK0
* 1: RCO3.6M
* 2: XO32K
* 3: RCO32K

***Notice***:

*1. Make sure to configure this bit when LPTIM1\_CLK\_EN=0. If the LPTIM1\_CLK\_EN bit is enabled, the user must disable it by software first, wait for at least 2 current clock cycles or query the RCC\_SR1 register, and then configure the LPTIM1 interface clock source.*

*2. This bit and the LPTIM1\_EXT\_CLK\_SEL bit jointly determine the LPTIM1 interface clock source.*

*3. To select PCLK0 as clock source, the LPTIM1\_INF\_CLK\_EN bit in the RCC\_CGR1 register must have been enabled.*

**Bits 7-6 LPTIM0\_CLK\_SEL:** LPTIM0 LPTIM0 interface clock source selection.

* 0: PCLK0
* 1: RCO3.6M
* 2: XO32K
* 3: RCO32K

***Notice***:

*1. Make sure to configure this bit when LPTIM0\_CLK\_EN=0. If the LPTIM0\_CLK\_EN bit is enabled, the user must disable it by software first, wait for at least 2 current clock cycles or query the RCC\_SR1 register, and then configure the LPTIM0 interface clock source.*

*2. This bit and the LPTIM0\_EXT\_CLK\_SEL bit jointly determine the LPTIM0 interface clock source.*

*3. To select PCLK0 as clock source, the LPTIM0\_INF\_CLK\_EN bit in the RCC\_CGR1 register must have been enabled.*

**Bits 5-4 LCDCTRL\_CLK\_SEL:** LCDCTRL LCDCTRL interface clock source selection.

* 0: XO32K
* 1: RCO32K
* >1: RCO3.6M

**Bit 3-2 LPUART\_CLK\_SEL:** LPUART interface clock source selection.

* 0: XO32K
* 1: RCO32K
* >1: RCO3.6M

**Bit 1 RTC\_CLK\_SEL:** interface clock source selection.

* 0: XO32K
* 1: RCO32K

**Bit 0 IWDG\_CLK\_SEL:** IWDG interface clock source selection.

* 0: XO32K
* 1: RCO32K

##### 8.3.3 RCC\_CR2

Offset: 0x008

Reset Value: 0x00000000

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **31-17** | **16-15** | **14-13** | **12-11** | |
| RESERVED | UART0\_CLK\_SEL | UART1\_CLK\_SEL | UART2\_CLK\_SEL | |
| r | r/w | r/w | r/w | |
| **10-9** | **8-7** | **6-5** | **4-2** | **1-0** |
| UART3\_CLK\_SEL | RESERVED | ADCCTRL\_CLK\_SEL | I2S\_CLK\_SEL | QSPI\_CLK\_SEL |
| r/w | r | r/w | r/w | r/w |

**Bits 31-17 RESERVED:** Must be kept, and can't be modified.

**Bits 16-15 UART0\_CLK\_SEL:** UART0 interface clock source selection.

* 0: PCLK0
* 1: RCO3.6M
* 2: XO32K
* 3: XO24M

***Notice:*** *Make sure to configure this bit when UART0\_CLK\_EN=0. If the UART0\_CLK\_EN bit is enabled, the user must disable it by software first, wait for at least 2 current clock cycles or query the RCC\_SR1 register, and then configure the UART0 interface clock source.*

**Bits 14-13 UART1\_CLK\_SEL:** UART1 interface clock source selection.

* 0: PCLK0
* 1: RCO3.6M
* 2: XO32K
* 3: XO24M

**Notice:** *Make sure to configure this bit when UART1\_CLK\_EN=0. If the UART1\_CLK\_EN bit is enabled, the user must disable it by software first, wait for at least 2 current clock cycles or query the RCC\_SR1 register, and then configure the UART1 interface clock source.*

**Bits 12-11 UART2\_CLK\_SEL:** interface clock source selection.

* 0: PCLK1
* 1: RCO3.6M
* 2: XO32K
* 3: XO24M

***Notice:*** *Make sure to configure this bit when UART2\_CLK\_EN=0. If the UART2\_CLK\_EN bit is enabled, the user must disable it by software first, wait for at least 2 current clock cycles or query the RCC\_SR1 register, and then configure the UART2 interface clock source.*

**Bits 10-9 UART3\_CLK\_SEL:**UART3 interface clock source selection.

* 0: PCLK1
* 1: RCO3.6M
* 2: XO32K
* 3: XO24M

**Notice:** *Make sure to configure this bit when UART3\_CLK\_EN=0. If the UART3\_CLK\_EN bit is enabled, the user must disable it by software first, wait for at least 2 current clock cycles or query the RCC\_SR1 register, and then configure the UART3 interface clock source.*

**Bits 8-7 RESERVED:** Must be kept, and can't be modified.

**Bits 6-5 ADCCTRL\_CLK\_SEL:** ADCCTRL interface clock source selection.

* 0:PCLK1
* 1:SYS\_CLK
* 2:PLL
* 3:RCO48M

**Notice:** *Make sure to configure this bit when ADCCTRL\_CLK\_EN=0. If the ADCCTRL\_CLK\_EN bit is enabled, the user must disable it by software first, wait for at least 2 current clock cycles or query the RCC\_SR1 register, and then configure the ADCCTRL interface clock source.*

**Bits 4-2 I2S\_CLK\_SEL:** I2S interface clock source selection.

* 0: PCLK0
* 1: XO24M
* 2: PLL
* 3: XO32M
* >3: external IOM\_I2S\_CLK

###### *Notice:*

###### *1. Make sure to configure this bit when I2S\_CLK\_EN=0. If the I2S\_CLK\_EN bit is enabled, the user must disable it by software first, wait for at least 2 current clock cycles or query the RCC\_SR1 register, and then configure the I2S interface clock source.*

###### *2. When I2S acts as a slave, the clock source must be configured to external IOM\_I2S\_CLK; when I2S acts as a master, the clock source is selected according to functional requirements.*

**Bits 1-0 QSPI\_CLK\_SEL:** interface clock source selection.

* 0: HCLK
* 1: SYS\_CLK
* >1: PLL

***Notice:*** *Make sure to configure this bit when QSPI\_CLK\_EN=0. If the QSPI\_CLK\_EN bit is enabled, the user must disable it by software first, wait for at least 2 current clock cycles or query the RCC\_SR1 register, and then configure the QSPI interface clock source.*

**8.3.4 RCC\_CGR0**

Offset:0x00C

Reset Value:0x00000000

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| **31** | **30** | **29** | **28** | **27** | **26** | **25** | **24** |
| PWR\_CLK  \_EN | DMAC0\_C  LK\_EN | DMAC1\_C  LK\_EN | CRC\_CLK  \_EN | BASICTIM0  \_CLK\_EN | BASICTIM1  \_CLK\_EN | IOM0\_CL  K\_EN | IOM1\_CL  K\_EN |
| r/w | r/w | r/w | r/w | r/w | r/w | r/w | r/w |
| **23** | **22** | **21** | **20** | **19** | **18** | **17** | **16** |
| IOM2\_CLK  \_EN | IOM3\_CL  K\_EN | SYSCFG\_  CLK\_EN | UART0\_C  LK\_EN | UART1\_CL  K\_EN | UART2\_CL  K\_EN | UART3\_C  LK\_EN | LPUART\_  CLK\_EN |
| r/w | r/w | r/w | r/w | r/w | r/w | r/w | r/w |
| **15** | **14** | **13** | **12** | **11** | **10** | **9** | **8** |
| SSP0\_CLK  \_EN | SSP1\_CL  K\_EN | SSP2\_CL  K\_EN | I2C0\_CLK  \_EN | I2C1\_CLK\_  EN | I2C2\_CLK\_  EN | RESERVE  D | ADCCTRL \_CLK\_EN |
| r/w | r/w | r/w | r/w | r/w | r/w | r | r/w |
| **7** | **6** | **5** | **4** | **3** | **2** | **1** | **0** |
| AFEC\_CL  K\_EN | LCDCTRL \_CLK\_EN | DACCTRL \_CLK\_EN | LORAC\_C  LK\_EN | GPTIM0\_C  LK\_EN | GPTIM1\_C  LK\_EN | GPTIM2\_ CLK\_EN | GPTIM3\_ CLK\_EN |
| r/w | r/w | r/w | r/w | r/w | r/w | r/w | r/w |

**Bits 31 PWR\_CLK\_EN:**PWR clock Enable.

* 0: Disable
* 1: Enable

**Bit 30 DMAC0\_CLK\_EN:**DMAC0 clock Enable.

* 0: Disable
* 1: Enable

**Bit 29 DMAC1\_CLK\_EN:**DMAC1 clock Enable.

* 0: Disable
* 1: Enable

**Bit 28 CRC\_CLK\_EN:**CRC clock Enable.

* 0: Disable
* 1: Enable

**Bit 27 BASICTIM0\_CLK\_EN:**BASICTIM0 clock Enable.

* 0: Disable
* 1: Enable

**Bit 26 BASICTIM1\_CLK\_EN:**BASICTIM1 clock Enable.

* 0: Disable
* 1: Enable

**Bit 25 IOM0\_CLK\_EN:**IOM0 clock Enable.

* 0: Disable
* 1: Enable

**Bit 24 IOM1\_CLK\_EN:**IOM1 clock Enable.

* 0: Disable
* 1: Enable

**Bit 23 IOM2\_CLK\_EN:**IOM2 clock Enable.

* 0: Disable
* 1: Enable

**Bit 22 IOM3\_CLK\_EN:**IOM3 clock Enable.

* 0:Disable
* 1:Enable

**Bit 21 SYSCFG\_CLK\_EN:**SYSCFG clock Enable.

* 0: Disable
* 1: Enable

**Bit 20 UART0\_CLK\_EN:**UART0 clock Enable.

* 0: Disable
* 1: Enable

**Bit 19 UART1\_CLK\_EN:**UART1 clock Enable.

* 0: Disable
* 1: Enable

**Bit 18 UART2\_CLK\_EN:**UART2 clock Enable.

* 0: Disable
* 1: Enable

**Bit 17 UART3\_CLK\_EN:**UART3 clock Enable.

* 0: Disable
* 1: Enable

**Bit 16 LPUART\_CLK\_EN:**LPUART clock Enable.

* 0: Disable
* 1: Enable

**Bit 15 SSP0\_CLK\_EN:**SSP0 clock Enable.

* 0: Disable
* 1: Enable

**Bit 14 SSP1\_CLK\_EN:**SSP1 clock Enable.

* 0: Disable
* 1: Enable

**Bit 13 SSP2\_CLK\_EN:**SSP2 clock Enable.

* 0: Disable
* 1: Enable

**Bit 12 I2C0\_CLK\_EN:**I2C0 clock Enable.

* 0: Disable
* 1: Enable

**Bit 11 I2C1\_CLK\_EN:**I2C1 clock Enable.

* 0: Disable
* 1: Enable

**Bit 10 I2C2\_CLK\_EN:**I2C2 clock Enable.

* 0: Disable
* 1: Enable

**Bits 9 RESERVED:** Must be kept, and can't be modified.

**Bit 8 ADCCTRL\_CLK\_EN:**ADCCTRL clock Enable.

* 0: Disable
* 1: Enable

**Bit 7 AFEC\_CLK\_EN:**AFEC clock Enable.

* 0: Disable
* 1: Enable

**Bit 6 LCDCTRL\_CLK\_EN:**LCDCTRL clock Enable.

* 0: Disable
* 1: Enable

**Bit 5 DACCTRL\_CLK\_EN:**DACCTRL clock Enable.

* 0: Disable
* 1: Enable

**Bit 4 LORAC\_CLK\_EN:**LORAC clock Enable.

* 0: Disable
* 1: Enable

**Bit 3 GPTIM0\_CLK\_EN:**GPTIM0 clock Enable.

* 0: Disable
* 1: Enable

**Bit 2 GPTIM1\_CLK\_EN:**GPTIM1 clock Enable.

* 0: Disable
* 1: Enable

**Bit 1 GPTIM2\_CLK\_EN:**GPTIM2 clock Enable.

* 0: Disable
* 1: Enable

**Bit 0 GPTIM3\_CLK\_EN:**GPTIM3 clock Enable.

* 0: Disable
* 1: Enable

##### 8.3.5 RCC\_CGR1

Offset:0x010

Reset Value:0x00000000

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| **31-13** | **12** | **11** | **10** | **9** | **8** | **7** |
| RESERVED | LPTIM1\_INF  \_CLK\_EN | LPTIM1\_CLK  \_EN | RNGC\_CLK  \_EN | LPTIM0\_INF  \_CLK\_EN | I2S\_CLK\_  EN | SAC\_CLK\_  EN |
| r | r/w | r/w | r/w | r/w | r/w | r/w |
| **6** | **5** | **4** | **3** | **2** | **1** | **0** |
| WWDG\_CN  T\_CLK\_EN | QSPI\_CLK\_  EN | LPTIM0\_CLK  \_EN | IWDG\_CLK  \_EN | WWDG\_CL  K\_EN | RTC\_CLK\_  EN | SEC\_CLK\_  EN |
| r/w | r/w | r/w | r/w | r/w | r/w | r/w |

**Bits 31-13 RESERVED:** Must be kept, and can't be modified.

**Bit 12 LPTIM1\_INF\_CLK\_EN:**LPTIM1 interface PCLK0 clock Enable.

* 0: Disable
* 1: Enable

**Bit 11 LPTIM1\_CLK\_EN:** LPTIM1 clock Enable.

* 0: Disable
* 1: Enable

**Notice:** *If PCLK0 is selected as the clock source, the LPTIM1\_INF\_CLK\_EN bit must be enabled before enabling the LPTIM1 clock, while it must be disabled after the LPTIM1 clock is disabled.*

**Bit 10 RNGC\_CLK\_EN:** RNGC clock Enable.

* 0: Disable
* 1: Enable

**Bit 9 LPTIM0\_INF\_CLK\_EN:** LPTIM0 interface PCLK0 clock Enable.

* 0: Disable
* 1: Enable

**Bit 8 I2S\_CLK\_EN:** I2S clock Enable.

* 0: Disable
* 1: Enable

**Bit 7 SAC\_CLK\_EN:** SAC clock Enable.

* 0: Disable
* 1: Enable

**Bit 6 WWDG\_CNT\_CLK\_EN:** WWDG counter clock Enable.

* 0: Disable
* 1: Enable

**Bit 5 QSPI\_CLK\_EN:** QSPI clock Enable.

* 0: Disable
* 1: Enable

**Bit 4 LPTIM0\_CLK\_EN:** LPTIM0 clock Enable.

* 0: Disable
* 1: Enable

**Notice:** *If PCLK0 is selected as the clock source, the LPTIM0\_INF\_CLK\_EN bit must be enabled before enabling the LPTIM0 clock, while it must be disabled after the LPTIM0 clock is disabled.*

**Bit 3 IWDG\_CLK\_EN:**IWDG clock Enable.

* 0: Disable
* 1: Enable

**Bit 2 WWDG\_CLK\_EN:** WWDG clock Enable.

* 0: Disable
* 1: Enable

**Bit 1 RTC\_CLK\_EN:** RTC clock Enable.

* 0: Disable
* 1: Enable

**Bit 0 SEC\_CLK\_EN:** SEC clock Enable.

* 0: Disable
* 1: Enable

##### 8.3.6 RCC\_CGR2

Offset:0x014

Reset Value:0x00000000

This register is in the AON power domain. Read the *RCC\_SR* register before configuring this register.

When the corresponding bit is set in the *RCC\_SR* register, this register can be read; when all the

bits are set in the *RCC\_SR* register, this register can be written.

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| **31-6** | **5** | **4** | **3** | **2** | **1** | **0** |
| RESERVED | LPTIM1\_AO N\_CLK\_EN | LPTIM\_AON  \_CLK\_EN | LCDCTRL\_A ON\_CLK\_EN | LPUART\_AO  N\_CLK\_EN | RTC\_AON \_CLK\_EN | IWDG\_AON  \_CLK\_EN |
| r | r/w | r/w | r/w | r/w | r/w | r/w |

**Bits 31-6 RESERVED:** Must be kept, and can't be modified.

**Bit 5 LPTIM1\_AON\_CLK\_EN:** Enable the LPTIM1 interface clock in AON domain

* 0: Disable
* 1: Enable

**Bit 4 LPTIM\_AON\_CLK\_EN:** Enable the LPTIM interface clock in AON domain.

* 0: Disable
* 1: Enable

**Bit 3 LCDCTRL\_AON\_CLK\_EN:** Enable the LCDCTRL interface clock in AON domain.

* 0: Disable
* 1: Enable

**Bit 2 LPUART\_AON\_CLK\_EN:** Enable the LPUART interface clock in AON domain.

* 0: Disable
* 1: Enable

**Bit 1 RTC\_AON\_CLK\_EN:** Enable the RTC interface clock in AON domain.

* 0: Disable
* 1:Enable

**Bit 0 IWDG\_AON\_CLK\_EN:** Enable the IWDG interface clock in AON domain.

* 0: Disable
* 1: Enable

##### 8.3.7 RCC\_RST0

Offset:0x018

Reset Value:0xffffffff

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| **31** | **30** | **29** | **28** | **27** | **26** | **25** | **24** |
| UART0\_R  ST\_N | UART1\_R  ST\_N | UART2\_R  ST\_N | UART3\_R  ST\_N | LPUART\_  RST\_N | SSP0\_RS  T\_N | SSP1\_RS  T\_N | SSP2\_RS  T\_N |
| r/w | r/w | r/w | r/w | r/w | r/w | r/w | r/w |
| **23** | **22** | **21** | **20** | **19** | **18** | **17** | **16** |
| QSPI\_RST  \_N | I2C0\_RST  \_N | I2C1\_RST  \_N | I2C2\_RST  \_N | RESERVE  D | ADCCTRL  \_RST\_N | AFEC\_RS  T\_N | LCDCTRL  \_RST\_N |
| r/w | r/w | r/w | r/w | r | r/w | r/w | r/w |
| **15** | **14** | **13** | **12** | **11** | **10** | **9** | **8** |
| DACCTRL  \_RST\_N | LORAC\_R  ST\_N | IOM\_RST  \_N | GPTIM0\_  RST\_N | GPTIM1\_  RST\_N | GPTIM2\_  RST\_N | GPTIM3\_  RST\_N | BASICTIM 0\_RST\_N |
| r/w | r/w | r/w | r/w | r/w | r/w | r/w | r/w |
| **7** | **6** | **5** | **4** | **3** | **2** | **1** | **0** |
| BASICTIM 1\_RST\_N | LPTIM\_R  ST\_N | IWDG\_RS  T\_N | WWDG\_R  ST\_N | RTC\_RST  \_N | CRC\_RST  \_N | SEC\_RST  \_N | SAC\_RST  \_N |
| r/w | r/w | r/w | r/w | r/w | r/w | r/w | r/w |

**Bit 31 UART0\_RST\_N:** UART0 reset control.

* 0: reset
* 1: no reset

**Bit 30 UART1\_RST\_N:** UART1 reset control.

* 0: reset
* 1: no reset

**Bits 29 UART2\_RST\_N:** UART1reset control**.**

* 0: reset
* 1: no reset

**Bit 28 UART3\_RST\_N:** UART3 reset control**.**

* 0: reset
* 1: no reset

**Bit 27 LPUART\_RST\_N:** reset control**.**

* 0: reset
* 1: no reset

**Bit 26 SSP0\_RST\_N:** reset control**.**

* 0: reset
* 1: no reset

**Bit 25 SSP1\_RST\_N:** SSP1 reset control**.**

* 0: reset
* 1: no reset

**Bit 24 SSP2\_RST\_N:** SSP2 reset control**.**

* 0: reset
* 1: no reset

**Bit 23 QSPI\_RST\_N:** QSPI reset control**.**

* 0: reset
* 1: no reset

**Bit 22 I2C0\_RST\_N:** I2C0 reset control**.**

* 0: reset
* 1: no reset

**Bits 21 I2C1\_RST\_N:** I2C1 复Bits控制.

* 0: reset
* 1: no reset

**Bit 20 I2C2\_RST\_N:** I2C2 reset control**.**

* 0: reset
* 1: no reset

**Bit 19 RESERVED:** Must be kept, and can't be modified.

**Bit 18 ADCCTRL\_RST\_N:** ADCCTRL reset control**.**

* 0: reset
* 1: no reset

**Bit 17 AFEC\_RST\_N:** AFEC reset control**.**

* 0: reset
* 1: no reset

**Bit 16 LCDCTRL\_RST\_N:** LCDCTRL reset control**.**

* 0: reset
* 1: no reset

**Bit 15 DACCTRL\_RST\_N:** DACCTRL reset control**.**

* 0: reset
* 1: no reset

**Bit 14 LORAC\_RST\_N:** LORAC reset control**.**

* 0: reset
* 1: no reset

**Bit 13 IOM\_RST\_N:** IOM reset control**.**

* 0: reset
* 1: no reset

**Bit 12 GPTIM0\_RST\_N:** GPTIM0 reset control**.**

* 0: reset
* 1: no reset

**Bit 11 GPTIM1\_RST\_N:** GPTIM1 reset control**.**

* 0: reset
* 1: no reset

**Bit 10 GPTIM2\_RST\_N:** GPTIM2 reset control**.**

* 0: reset
* 1: no reset

**Bit 9 GPTIM3\_RST\_N:**GPTIM3 reset control**.**

* 0: reset
* 1: no reset

**Bit 8 BASICTIM0\_RST\_N:** BASICTIM0 reset control**.**

* 0: reset
* 1: no reset

**Bit 7 BASICTIM1\_RST\_N:** BASICTIM1 reset control**.**

* 0: reset
* 1: no reset

**Bit 6 LPTIM\_RST\_N:** LPTIM reset control**.**

* 0: reset
* 1: no reset

**Bit 5 IWDG\_RST\_N:** IWDG reset control**.**

* 0: reset
* 1: no reset

**Bit 4 WWDG\_RST\_N:**WWDG reset control**.**

* 0: reset
* 1: no reset

**Bit 3 RTC\_RST\_N:** RTC reset control**.**

* 0: reset
* 1: no reset

**Bit 2 CRC\_RST\_N:** CRC reset control**.**

* 0: reset
* 1: no reset

**Bit 1 SEC\_RST\_N:** SEC reset control**.**

* 0: reset
* 1: no reset

**Bits 0 SAC\_RST\_N:** SAC reset control**.**

* 0: reset
* 1: no reset

##### 8.3.8 RCC\_RST1

Offset:0x01C

Reset Value:0x0000001f

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **31-5** | **4** | **3** | **2** | **1** | **0** |
| RESERVED | LPTIM1\_RST\_N | RNGC\_RST\_N | I2S\_RST\_N | DMAC0\_RST\_N | DMAC1\_RST\_N |
| r | r/w | r/w | r/w | r/w | r/w |

**Bits 31-5 RESERVED:** Must be kept, and can't be modified.

**Bit 4 LPTIM1\_RST\_N:** LPTIM1 reset control**.**

* 0: reset
* 1: no reset

**Bit 3 RNGC\_RST\_N:** RNGC reset control**.**

* 0: reset
* 1: no reset

**Bit 2 I2S\_RST\_N:** I2S reset control**.**

* 0: reset
* 1: no reset

**Bit 1 DMAC0\_RST\_N:** DMAC0 reset control**.**

* 0: reset
* 1: no reset

**Bit 0 DMAC1\_RST\_N:** DMAC1 reset control**.**

* 0: reset
* 1: no reset

##### 8.3.9 RCC\_RST\_SR

Offset:0x020

Reset Value:0x00000040

***Notice:*** *The BOR\_RESET\_SR and STANDBY\_RESET\_SR are in the AON domain.*

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| **31-7** | **6** | **5** | **4** | **3** | **2** | **1** | **0** |
| RESERVED | BOR\_RE SET\_SR | IWDG\_RE  SET\_SR | WWDG\_RE  SET\_SR | EFC\_RE SET\_SR | CPU\_RE SET\_SR | SEC\_RE SET\_SR | STANDBY\_ RESET\_SR |
| r | r/w | r/w | r/w | r/w | r/w | r/w | r/w |

**Bits 31-7 RESERVED:** Must be kept, and can't be modified.

**Bit 6 BOR\_RESET\_SR:** BOR reset status. This bit is set by hardware and cleared by software writing 1 to it.

* 0: no BOR reset occurred
* an BOR reset occurred

**Bit 5 IWDG\_RESET\_SR:** IWDG reset status. This bit is set by hardware and cleared by software writing 1 to it.

* 0: no IWDG reset occurred
* an IWDG reset occurred

**Bit 4 WWDG\_RESET\_SR:** WWDG reset status. This bit is set by hardware and cleared by software writing 1 to it.

* 0: no WWDG reset occurred
* an WWDG reset occurred

**Bit 3 EFC\_RESET\_SR:** EFC reset status. This bit is set by hardware and cleared by software writing 1 to it.

* 0: no EFC reset occurred
* an EFC reset occurred

**Bit 2 CPU\_RESET\_SR:** CPU reset status. This bit is set by hardware and cleared by software writing 1 to it.

* 0: no CPU reset occurred
* an CPU reset occurred

**Bit 1 SEC\_RESET\_SR:** SEC reset status. This bit is set by hardware and cleared by software writing 1 to it.

* 0: no SEC reset occurred
* an SEC reset occurred

**Bit 0 STANDBY\_RESET\_SR:** MPU reset status. This bit is set by hardware and cleared by software writing 1 to it.

* 0: no MPU reset occurred
* an MPU reset occurred

##### 8.3.10 RCC\_RST\_CR

Offset:0x024

Reset Value:0x00000004

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| **31-6** | **5** | **4** | **3** | **2** | **1** | **0** |
| RESERVED | IWDG\_RESE  T\_REQ\_EN | WWDG\_RES ET\_REQ\_EN | EFC\_RESE  T\_REQ\_EN | CPU\_RESE T\_REQ\_EN | SEC\_RESE T\_REQ\_EN | RESERVED |
| r | r/w | r/w | r/w | r/w | r/w | r |

**Bits 31-6 RESERVED:** Must be kept, and can't be modified.

**Bit 5 IWDG\_RESET\_REQ\_EN:** IWDG reset enable.

* 0: Disable
* 1: Enable

**Bit 4 WWDG\_RESET\_REQ\_EN:** WWDG reset enable.

* 0: Disable
* 1: Enable

**Bit 3 EFC\_RESET\_REQ\_EN:** EFC reset enable.

* 0: Disable
* 1: Enable

**Bit 2 CPU\_RESET\_REQ\_EN:** CPU reset enable.

* 0: Disable
* 1: Enable

**Bit 1 SEC\_RESET\_REQ\_EN:** SEC reset enable.

* 0: Disable
* 1: Enable

**Bit 0 RESERVED:** Must be kept, and can't be modified.

##### 8.3.11 RCC\_SR

Offset:0x028

Reset Value:0x0000003f

|  |  |  |  |
| --- | --- | --- | --- |
| **31-6** | | **5** | **4** |
| RESERVED | | SET\_LPTIM1\_AON\_CL  K\_EN\_DONE | SET\_LPTIM\_AON\_CLK  \_EN\_DONE |
| r | | r | r |
| **3** | **2** | **1** | **0** |
| SET\_LCDCTRL\_AON\_  CLK\_EN\_DONE | SET\_LPUART\_AON\_C  LK\_EN\_DONE | SET\_RTC\_AON\_CLK\_  EN\_DONE | SET\_IWDG\_AON\_CLK  \_EN\_DONE |
| r | r | r | r |

**Bits 31-6 RESERVED:** Must be kept, and can't be modified.

**Bit 5 SET\_LPTIM1\_AON\_CLK\_EN\_DONE:** LPTIM1\_AON\_CLK\_EN configuration status. This bit is set and cleared by hardware.

* 0: configuration is in progress
* 1: configuration is complete

**Bit 4 SET\_LPTIM\_AON\_CLK\_EN\_DONE:** LPTIM\_AON\_CLK\_EN configuration status. This bit is set and cleared by hardware.

* 0: configuration is in progress
* 1: configuration is complete

**Bit 3 SET\_LCDCTRL\_AON\_CLK\_EN\_DONE:** LCDCTRL\_AON\_CLK\_EN configuration status. This bit is set and cleared by hardware.

* 0: configuration is in progress
* 1: configuration is complete

**Bit 2 SET\_LPUART\_AON\_CLK\_EN\_DONE:** LPUART\_AON\_CLK\_EN configuration status. This bit is set and cleared by hardware.

* 0: configuration is in progress
* 1: configuration is complete

**Bit 1 SET\_RTC\_AON\_CLK\_EN\_DONE:** RTC\_AON\_CLK\_EN configuration status. This bit is set and cleared by hardware.

* 0: configuration is in progress
* 1: configuration is complete

**Bit 0 SET\_IWDG\_AON\_CLK\_EN\_DONE:**IWDG\_AON\_CLK\_EN configuration status. This bit is set and cleared by hardware.

* 0: configuration is in progress
* 1: configuration is complete

##### 8.3.12 RCC\_SR1

Offset:0x02C

Reset Value:0x00000000

The clock should be disabled before the clock source is switched or the frequency division changes to avoid glitches. This register is used to determine whether the clock is disabled.

|  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- |
| **31-21** | **20** | | **19** |  | **18** | **17** | | **16** |
| RESERVED | LPTIM1\_CLK  \_EN\_SYNC | | LPTIM1\_AON\_C  LK\_EN\_SYNC |  | UART0\_CLK\_  EN\_SYNC | UART1\_CLK \_EN\_SYNC | | UART2\_CLK\_E  N\_SYNC |
| r | r | | r |  | r | r | | r |
| **15** | **14** | | **13** |  | **12** | **11** | | **10** |
| UART3\_CLK\_  EN\_SYNC | RESERVED | | ADCCTRL\_CLK  \_EN\_SYNC |  | LPTIM\_CLK\_  EN\_SYNC | QSPI\_CLK\_E  N\_SYNC | | LPUART\_CLK\_  EN\_SYNC |
| r | r | | r |  | r | r | | r |
| **9** | **8** | | **7** |  | **6** | **5** | | **4** |
| LCDCTRL\_CL K\_EN\_SYNC | IWDG\_CLK\_  EN\_SYNC | | RTC\_CLK\_EN\_S  YNC |  | MCO\_CLK\_E  N\_SYNC | I2S\_CLK\_EN  \_SYNC | | LPTIM\_AON\_C LK\_EN\_SYNC |
| r | r | | r |  | r | r | | r |
| **3** | | **2** | | **1** | | | **0** | |
| LCDCTRL\_AON\_CLK\_  EN\_SYNC | | LPUART\_AON\_CLK\_E  N\_SYNC | | RTC\_AON\_CLK\_EN\_S  YNC | | | IWDG\_AON\_CLK\_EN\_  SYNC | |
| r | | r | | r | | | r | |

**Bits 31-21 RESERVED:** Must be kept, and can't be modified.

**Bit 20 LPTIM1\_CLK\_EN\_SYNC:** LPTIM1\_CLK\_EN actual status.

* 0: disabled
* 1: enabled

**Bit 19 LPTIM1\_AON\_CLK\_EN\_SYNC:** LPTIM1\_AON\_CLK\_EN actual status.

* 0: disabled
* 1: enabled

**Bit 18 UART0\_CLK\_EN\_SYNC:** UART0\_CLK\_EN actual status.

* 0: disabled
* 1: enabled

**Bit 17 UART1\_CLK\_EN\_SYNC:** UART1\_CLK\_EN actual status.

* 0: disabled
* 1: enabled

**Bit 16 UART2\_CLK\_EN\_SYNC:** UART2\_CLK\_EN actual status.

* 0: disabled
* 1: enabled

**Bit 15 UART3\_CLK\_EN\_SYNC:** UART3\_CLK\_EN actual status.

* 0: disabled
* 1: enabled

**Bit 14 RESERVED:** Must be kept, and can't be modified.

**Bit 13 ADCCTRL\_CLK\_EN\_SYNC:** ADCCTRL\_CLK\_EN actual status.

* 0: disabled
* 1: enabled

**Bit 12 LPTIM\_CLK\_EN\_SYNC:** LPTIM\_CLK\_EN actual status.

* 0: disabled
* 1: enabled

**Bit 11 QSPI\_CLK\_EN\_SYNC:** QSPI\_CLK\_EN actual status.

* 0: disabled
* 1: enabled

**Bit 10 LPUART\_CLK\_EN\_SYNC:** LPUART\_CLK\_EN actual status.

* 0: disabled
* 1: enabled

**Bit 9 LCDCTRL\_CLK\_EN\_SYNC:** LCDCTRL\_CLK\_EN actual status.

* 0: disabled
* 1: enabled

**Bit 8 IWDG\_CLK\_EN\_SYNC:** IWDG\_CLK\_EN actual status.

* 0: disabled
* 1: enabled

**Bit 7 RTC\_CLK\_EN\_SYNC:** RTC\_CLK\_EN actual status.

* 0: disabled
* 1: enabled

**Bit 6 MCO\_CLK\_EN\_SYNC:** MCO\_CLK\_EN actual status.

* 0: disabled
* 1: enabled

**Bit 5 I2S\_CLK\_EN\_SYNC:** I2S\_CLK\_EN actual status.

* 0: disabled
* 1: enabled

**Bit 4 LPTIM\_AON\_CLK\_EN\_SYNC:** LPTIM\_AON\_CLK\_EN actual status.

* 0: disabled
* 1: enabled

**Bit 3 LCDCTRL\_AON\_CLK\_EN\_SYNC:** LCDCTRL\_AON\_CLK\_EN actual status.

* 0: disabled
* 1: enabled

**Bit 2 LPUART\_AON\_CLK\_EN\_SYNC:** LPUART\_AON\_CLK\_EN actual status.

* 0: disabled
* 1: enabled

**Bit 1 RTC\_AON\_CLK\_EN\_SYNC:** RTC\_AON\_CLK\_EN actual status.

* 0: disabled
* 1: enabled

**Bit 0 IWDG\_AON\_CLK\_EN\_SYNC:** IWDG\_AON\_CLK\_EN actual status.

* 0: disabled
* 1: enabled

##### 8.3.13 RCC\_CR3

Offset:0x030

Reset Value:0x00000000

|  |  |  |
| --- | --- | --- |
| **31-16** | **15-8** | **7-0** |
| RESERVED | I2S\_MCLK\_DIV | I2S\_SCLK\_DIV |
| r | r/w | r/w |

**Bits 31-16 RESERVED:** Must be kept, and can't be modified.

**Bits 15-8 I2S\_MCLK\_DIV:** I2S interface clock MCLK frequency division.

* 0: not divided
* 1: not divided
* 2: divided by 2
* 3: divided by 3
* N: divided by N

***Notice:***

*1. Make sure to configure I2S\_MCLK\_DIV when I2S\_CLK\_EN=0. If the I2S\_CLK\_EN bit is enabled, the user must disable it by software first, wait for at least 2 current clock cycles or query the RCC\_SR1 register, and then configure I2S\_MCLK\_DIV.*

*2. When I2S acts as a slave, this bit must be configured to 0 or 1; when I2S acts as a master, this bit is configured according to functional requirements.*

*3. The duty cycle of the output clock is 50%.*

**Bits 7-0 I2S\_SCLK\_DIV:** I2S interface clock SCLK frequency division.

* 0: not divided
* 1: not divided
* 2: divided by 2
* 3: divided by 3
* N: divided by N

**Notice:**

***1.*** *Make sure to configure I2S\_SCLK\_DIV when I2S\_CLK\_EN=0. If the I2S\_CLK\_EN bit is enabled, the user must disable it by software first, wait for at least 2 current clock cycles or query the RCC\_SR1 register, and then configure I2S\_SCLK\_DIV.*

*2. When I2S acts as a slave, this bit must be configured to 0 or 1; when I2S acts as a master, this bit is configured according to functional requirements.*

*3. The duty cycle of the output clock is 50%.*

9 interrupt (Interrupt)

## 9. Interrupts

### 9.1 Main Features

* Support 37 IRQ interrupts.
* Configurable 0~7 priority levels for each IRQ interrupt

#### 9.2 SysTick

SysTick calibration value is 0x147. Using a 32.768 kHz clock source for SysTick counting gives

a reference time base of 10 ms.

### 9.3 Interrupt Vector Table

The interrupt vector Table is as follows:

#### Table 9-1 Interrupt Vector Table

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Position | Priority | Type of priority | Acronym | Description | Address |
|  | - | - | - | Reserved | 0x0000\_0000 |
|  | -3 | fixed | Reset | Reset | 0x0000\_0004 |
|  | -2 | fixed | NMI\_Handler | Secure area check error | 0x0000\_0008 |
|  | -1 | fixed | HardFault\_Handler | fault | 0x0000\_000C |
|  | 0 | settable | MemManage Handler | fault | 0x0000\_0010 |
|  | 1 | settable | BusFault Handler | fault | 0x0000\_0014 |
|  | 2 | settable | UsageFault Handler | fault | 0x0000\_0018 |
|  | - | - | - | Reserved | 0x0000\_001C -  0x0000\_002B |
|  | 3 | settable | SVC\_Handler | System service call via SWI instruction | 0x0000\_002C |
|  | - | - | - | Reserved | 0x0000\_0030 -  0x0000\_0037 |
|  | 5 | settable | PendSV\_Handler | Pendable request for system service | 0x0000\_0038 |
|  | 6 | settable | SysTick\_Handler | System tick timer | 0x0000\_003C |
| 0 | 7 | settable | sec | Include mpu | 0x0000\_0040 |
| 1 | 8 | settable | rtc | Include tamper io, cyc, wakeup io | 0x0000\_0044 |

9 interrupt (Interrupt)

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Position | Priority | Type of priority | Acronym | Description | Address |
| 2 | 9 | settable | wwdg |  | 0x0000\_0048 |
| 3 | 10 | settable | efc |  | 0x0000\_004C |
| 4 | 11 | settable | uart3 |  | 0x0000\_0050 |
| 5 | 12 | settable | i2c2 |  | 0x0000\_0054 |
| 6 | 13 | settable | uart0 |  | 0x0000\_0058 |
| 7 | 14 | settable | uart1 |  | 0x0000\_005C |
| 8 | 15 | settable | uart2 |  | 0x0000\_0060 |
| 9 | 16 | settable | lpuart |  | 0x0000\_0064 |
| 10 | 17 | settable | ssp0 |  | 0x0000\_0068 |
| 11 | 18 | settable | ssp1 |  | 0x0000\_006C |
| 12 | 19 | settable | qspi |  | 0x0000\_0070 |
| 13 | 20 | settable | i2c0 |  | 0x0000\_0074 |
| 14 | 21 | settable | i2c1 |  | 0x0000\_0078 |
| 15 | 22 | settable | - |  | 0x0000\_007C |
| 16 | 23 | settable | adcctrl |  | 0x0000\_0080 |
| 17 | 24 | settable | afec |  | 0x0000\_0084 |
| 18 | 25 | settable | ssp2 |  | 0x0000\_0088 |
| 19 | 26 | settable | dmac1 |  | 0x0000\_008C |
| 20 | 27 | settable | dacctrl |  | 0x0000\_0090 |
| 21 | 28 | settable | lorac |  | 0x0000\_0094 |
| 22 | 29 | settable | iom |  | 0x0000\_0098 |
| 23 | 30 | settable | gptim0 |  | 0x0000\_009C |
| 24 | 31 | settable | gptim1 |  | 0x0000\_00A0 |
| 25 | 32 | settable | gptim2 |  | 0x0000\_00A4 |
| 26 | 33 | settable | gptim3 |  | 0x0000\_00A8 |
| 27 | 34 | settable | basictim0 |  | 0x0000\_00AC |
| 28 | 35 | settable | basictim1 |  | 0x0000\_00B0 |
| 29 | 36 | settable | lptim0 |  | 0x0000\_00B4 |
| 30 | 37 | settable | sac |  | 0x0000\_00B8 |
| 31 | 38 | settable | dmac0 |  | 0x0000\_00BC |
| 32 | 39 | settable | i2s |  | 0x0000\_00C0 |
| 33 | 40 | settable | lcdctrl |  | 0x0000\_00C4 |
| 34 | 41 | settable | pwr |  | 0x0000\_00C8 |
| 35 | 42 | settable | lptim1 |  | 0x0000\_00CC |
| 36 | 43 | settable | iwdg |  | 0x0000\_00D0 |

## 10. Embedded Flash

### 10.1 Introduction

* The whole Flash is divided into Flash info area and Flash main area
* Flash size:
  + Flash info area: 16 KB in total
  + Flash main area: 256 KB for ASR6601SE, 128 KB for ASR6601CB
* Page erase (4KB), Mass erase (all flash main)

### 10.2 Main features

* Flash operations include read, program, page erase and mass erase
* Read access latency
* Acceleration for accessing the Flash memory
* Support instruction prefetch with 1 deep buffer
* Flash program operation supports single and continuous mode
* Option bytes in Flash info area
* Can be used to generate interrupt signals

### 10.3 Functional Description

#### 10.3.1 Flash info Area Division

The Flash info area is divided into four parts: Option Bytes, Factory Bytes, OTP and BootLoader.

See the table below for details.

**Table 10-1 Flash info** Area Division

|  |  |  |
| --- | --- | --- |
| Start Address | Description | Size |
| 0x10003000 | Option Bytes | 4KB |
| 0x10002000 | Factory Bytes | 4KB |
| 0x10001C00 | OTP | 1KB |
| 0x10000000 | BootLoader | 7KB |

#### 10.3.2 EFC\_CR **Protection**

By default, the EFC\_CR register cannot be modified, to modify it, the user must configure the

protection sequence correctly through the *EFC\_PROTECT\_SEQ* register in the following order.

If there is an error in the configuration, then the configuration is invalid, and the protection

sequence should be reconfigured.

1. First write “0x8C9DAEBF” to EFC\_PROTECT\_SEQ register
2. Then write “0x13141516” to EFC\_PROTECT\_SEQ register

#### 10.3.3 **Read Access Latency**

In order to improve Flash read performance, the number of wait states (READ\_NUM[19:16])

should be correctly programmed in *EFC\_TIMING\_CFG* register according to the frequency of

SYS\_CLK. The number of wait states (READ\_NUM) equals to (READ\_NUM+1) multiplied by

SYS\_CLK clock period. Details are as follows:

* When SYS\_CLK is 48MHz frequency, READ\_NUM must ≥ 2.
* When SYS\_CLK is 32MHz frequency, READ\_NUM must ≥ 1.
* When SYS\_CLK is 24MHz frequency, READ\_NUM must ≥ 1.
* When SYS\_CLK is 3.6MHz frequency, READ\_NUM must ≥ 0.
* When SYS\_CLK is 32KHz frequency, READ\_NUM must ≥ 0.

**The operations to switch to a high-frequency clock source for SYS\_CLK:**

1. Modify the READ\_NUM value in *EFC\_TIMING\_CFG* register to match the SYS\_CL after its clock source is switched.
2. Wait for the READ\_NUM\_DONE status bit in *EFC\_SR* register to be set.
3. Modify the SYS\_CLK\_SEL field in *RCC\_CR0* register to switch to the target clock source.

**The operations to switch to a low-frequency clock source for SYS\_CLK:**

(1) Modify the SYS\_CLK\_SEL field in *RCC\_CR0* register to switch to the target clock source.

(2) Modify the READ\_NUM value in *EFC\_TIMING\_CFG* register to match the SYS\_CLK after its clock source is switched.

(3) Wait for the READ\_NUM\_DONE status bit in *EFC\_SR* register to be set.

***Notice:*** *When the user wants to switch to a high-frequency clock source, first increase the READ\_NUM, and then configure the clock source selection bit; otherwise, first configure the clock source selection bit, and then decrease the READ\_NUM.*

#### 10.3.4 **Acceleration for Accessing the Flash Memory**

Read acceleration is disabled by default. If READ\_NUM < (2^HCLK\_DIV), read acceleration

can be enabled to achieve the maximum bus access efficiency. Note that read acceleration

must be enabled after READ\_NUM and HCLK\_DIV are configured.

***Notice:*** *Read acceleration and instruction prefetch can’t be enabled at the same time.*

#### 10.3.5 **Instruction Prefetch**

It is disabled by default. If READ\_NUM ≥ (2^HCLK\_DIV), read acceleration cannot be enabled.

You can choose to enable instruction prefetch to improve access efficiency.

***Notice:*** *Read acceleration and instruction prefetch can’t be enabled at the same time.*

#### 10.3.6 Flash Program Operation

There are two modes for Flash programming:

 **Single Programming Mode**

In single mode, it programs 2 words (8 Bytes) each time.

#####  **Continuous Programming Mode**

In continuous mode, it programs a complete word line (512 Bytes) each time.

During continuous programming, Flash cannot be read or executed, so the continuous

programming code must be executed in RAM.

**Steps for single programming:**

1. Set the PROG\_EN bit in register *EFC\_CR*.
2. Write the low 4 Bytes data into register *EFC\_PROG\_DATA0*.
3. Write the high 4 Bytes data into register *EFC\_PROG\_DATA1*.
4. Write any value to the Flash address to be written into.
5. Wait for the OPERATION\_DONE bit in register *EFC\_SR* to be set.
6. Write 1 to the OPERATION\_DONE bit in register *EFC\_SR* to clear the flag.

**Steps for continuous programming:**

1. Set the PROG\_EN, WRITE\_RELEASE\_EN and PROG\_MODE bits in register *EFC\_CR*.
2. Wait for the PROG\_DATA\_WAIT bit in register *EFC\_SR* to be set.
3. Write the low 4 Bytes data into register *EFC\_PROG\_DATA0*.
4. Write the high 4 Bytes data into register *EFC\_PROG\_DATA1*.
5. Write any value to the Flash address to be written into.
6. Wait for the PROG\_DATA\_WAIT bit in register *EFC\_SR* to be set.
7. Continue to write data to the *EFC\_PROG\_DATA0* and *EFC\_PROG\_DATA1 registers*.
8. Repeat **Step 6** and **Step 7** until 512 Bytes are written.
9. Wait for the OPERATION\_DONE bit in register *EFC\_SR* to be set.
10. Write 1 to the OPERATION\_DONE bit in register *EFC\_SR* to clear the flag.

#### 10.3.7 Flash Erase Operation

The Flash memory erase operation can be performed at page level (page erase) or on the

whole memory (mass erase).

* **Page erase**

Page erase unit size is 4KB.

* **Mass erase**

After a mass erase, the entire Flash main area will be 0xFF.

**Steps for page erase:**

(1) Set the PAGE\_ERASE\_EN bit in register *EFC\_CR*.

(2) Write any value to the Flash address to be erased.

(3) Wait for the OPERATION\_DONE bit in register *EFC\_SR* to be set.

(4) Write 1 to the OPERATION\_DONE bit in register *EFC\_SR* to clear the flag.

**Steps for mass erase:**

(1) Set the MASS\_ERASE\_EN bit in register *EFC\_CR*.

(2) Write any value to the Flash address 0x08000000.

(3) Wait for the OPERATION\_DONE bit in register *EFC\_SR* to be set.

(4) Write 1 to the OPERATION\_DONE bit in register *EFC\_SR* to clear the flag.

#### 10.4 Flash Option Bytes

Flash option bytes is divided into option0 and option1.

##### 10.4.1 Flash Option0

Option0 has 64 bits in total, and its format is as follows:

**Table 10-2 Flash Option0**

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| **63-50** | **49-44** | **43-38** | **37-32** | **31-26** | **25** | **24-19** |
| RESERVED | WR\_PROT ECT\_END | WR\_PROTE  CT\_START | EXE\_ONLY2  \_END | EXE\_ONLY2  \_START | EXE\_ONLY  \_KEEEP | EXE\_ONLY1  \_END |
| **18-13** | **12-5** | **4** | **3** | **2** | **1** | **0** |
| EXE\_ONLY  1\_START | DEBUG\_L  EVEL | RESERVED | SYS\_SRAM  \_RESET | FLASH\_BOO  T1 | USE\_FLAS H\_BOOT0 | FLASH\_BOO  T0 |

**Bits 63-50 Reserved:** Must be kept, and can't be modified.

**Bits 49-44 WR\_PROTECT\_END:** 写保护区域的结束 Page 地址.当 *WR\_PROTECT\_START > WR\_PROTECT\_END* 时,写保护区域Disable；默认Disable.

**Bits 43-38 WR\_PROTECT\_START:** 写保护区域的开始 Page 地址.当 *WR\_PROTECT\_START > WR\_PROTECT\_END* 时,写保护区域Disable；默认Disable.

**Bits 37-32 EXE\_ONLY2\_END:** ExeOnly 区域 2 的结束 Page 地址.当 *EXE\_ONLY2\_START > EXE\_ONLY2\_END* 时,ExeOnly 区域 2 Disable；默认Disable,修改时只能Enable或扩大区域,不能减少或关闭.

**Bits 31-26 EXE\_ONLY2\_START:** ExeOnly 区域 2 的开始 Page 地址.

当 *EXE\_ONLY2\_START > EXE\_ONLY2\_END* 时,ExeOnly 区域 2 Disable；默认Disable,修改时只能Enable或扩大区域,不能减少或关闭.

**Bit 25 EXE\_ONLY\_KEEP:** ExeOnly 区域在 DebugLevel 由 1 变为 0 时,是否保留:

* 0: 不保留 ExeOnly 区域
* 1: 保留 ExeOnly 区域支持修改,但只能写 0；DebugLevel 从 1 变为 0 时,EXE\_ONLY\_KEEP 自动变 1.

**Bits 24-19 EXE\_ONLY1\_END:**ExeOnly 区域 1 的结束 Page 地址.当 *EXE\_ONLY1\_START > EXE\_ONLY1\_END* 时,ExeOnly 区域 1 Disable；默认Disable,修改时只能Enable或扩大区域,不能减少或关闭.

**Bits 18-13 EXE\_ONLY1\_START:** ExeOnly 区域 1 的开始 Page 地址.当 *EXE\_ONLY1\_START > EXE\_ONLY1\_END* 时,ExeOnly 区域 1 Disable；默认Disable,修改时只能Enable或扩大区域,不能减少或关闭.

**Bits 12-5 DEBUG\_LEVEL:** debug\_level 配置.

* AA: Level0
* CC: Level2
* 其它值: Level1

**Bit 4 Reserved:** Must be kept, and can't be modified.

**Bit 3 SYS\_SRAM\_RESET**: System 复Bits后 Startup 过程中是否清除系统 SRAM:

* 1: 清除系统 SRAM
* 0: 不清除系统 SRAM

**Bit 2 FLASH\_BOOT1:** 用于启动模式判断.

**Bits 1 USE\_FLASH\_BOOT0:** 用于启动模式判断.

**Bit 0 FLASH\_BOOT0:** 用于启动模式判断.

启动模式判断如下:

**Table 10-3 SoC 启动方式配置Table**

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| DEBUG\_  LEVEL | USE\_FLASH\_  BOOT0 | FLASH\_  BOOT0 | BOOT0  PIN | FLASH\_  BOOT1 | MAIN\_FLASH\_  EMPTY | Boot Config |
| 2 | X | X | X | X | X | Boot from Flash Main |
| <2 | 0 | X | 0 | X | 0 | Boot from Flash Main |
| <2 | 0 | X | 0 | X | 1 | Boot from Flash Bootloader |
| <2 | 0 | X | 1 | 1 | X | Boot from Flash Bootloader |
| <2 | 0 | X | 1 | 0 | X | Boot from System SRAM |
| <2 | 1 | 1 | X | X | 0 | Boot from Flash Main |
| <2 | 1 | 1 | X | X | 1 | Boot from Flash Bootloader |
| <2 | 1 | 0 | X | 1 | X | Boot from Flash Bootloader |
| <2 | 1 | 0 | X | 0 | X | Boot from System SRAM |

##### 10.4.2 Flash Option1

Option1 共 64 Bits,其格式如下:

**Table 10-4 Flash Option1**

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **63-56** | **55** | **54-49** | **48** | **47-42** | **41-37** |
| RESERVED | SYSRAM\_HID  E\_EN | SYSRAM\_HID  E\_START | FLASH\_HIDE\_  EN | FLASH\_HIDE  \_START | RETRAM\_SEC  URE\_END |
| **36-32** | **31-24** | **23-18** | **17-12** | **11-6** | **5-0** |
| RETRAM\_SEC  URE\_START | RESERVED | SYSRAM\_SEC  URE\_END | SYSRAM\_SEC  URE\_START | FLASH\_SEC  URE\_END | FLASH\_SECU  RE\_START |

**Bits 63-56 Reserved:** Must be kept, and can't be modified.

**Bits 55 SYSRAM\_HIDE\_EN:**SysRamHide 区域Enable控制.

* 0:SysRamHide 区域Enable
* 1:SysRamHide 区域Disable仅当 FlashSecure 区域Enable时才有效.

**Bits 54-49 SYSRAM\_HIDE\_START:**SysRamHide 区域的开始 Page 地址.

必须在 SysRamSecure 区域内才Enable,区域范围由 SysRamHideStart 开始一直到 SysRamSecureEnd；仅当 FlashSecure 区域Enable时才有效.

**Bits 48 FLASH\_HIDE\_EN:**FlashHide 区域Enable控制.

* 0:FlashHide 区域Enable
* 1:FlashHide 区域Disable仅当 FlashSecure 区域Enable时才有效.

**Bits 47-42 FLASH\_HIDE\_START:**FlashHide 区域的开始 Page 地址.

必须在 FlashSecure 区域内才Enable,区域范围由 FlashHideStart 开始一直到 FlashSecureEnd；仅当 FlashSecure 区域Enable时才有效.

**Bits 41-37 RETRAM\_SECURE\_END:**RetRam Secure 区域的结束地址.

当 *RETRAM\_SECURE\_START > RETRAM\_SECURE\_END* 时,则该区域安全Disable； 仅当 FlashSecure 区域Enable时才有效.

**Bits 36-32 RETRAM\_SECURE\_START:**RetRam Secure 区域的开始地址.

当 *RETRAM\_SECURE\_START > RETRAM\_SECURE\_END* 时,则该区域安全Disable； 仅当 FlashSecure 区域Enable时才有效.**Bits 31-24 Reserved:** Must be kept, and can't be modified.

**Bits 23-18 SYSRAM\_SECURE\_END:**SysRam Secure 区域的结束地址.

当 *SYSRAM\_SECURE\_START > SYSRAM\_SECURE\_END* 时,则该区域安全Disable； 仅当 FlashSecure 区域Enable时才有效.

**Bits 17-12 SYSRAM\_SECURE\_START:**SysRam Secure 区域的开始地址.

当 *SYSRAM\_SECURE\_START > SYSRAM\_SECURE\_END* 时,则该区域安全Disable； 仅当 FlashSecure 区域Enable时才有效.

**Bits 11-6 FLASH\_SECURE\_END:**Flash Secure 区域的结束地址.

当 *FLASH \_SECURE\_START > FLASH \_SECURE\_END* 时,则该区域安全Disable；

FlashSecure 区域Enable也是安全区域总Enable开关,当 FlashSecure 去Enable时会触发擦除操作.

**Bits 5-0 FLASH \_SECURE\_START:**Flash Secure 区域的开始地址.

当 *FLASH\_SECURE\_START > FLASH \_SECURE\_END* 时,则该区域安全Disable；

FlashSecure 区域Enable也是安全区域总Enable开关,当 FlashSecure 去Enable时会触发擦除操作.

### 10.5 嵌入式 Flash 相关RegisterDescription

Base Address:0x40020000

**Table 10-5 嵌入式 Flash Register Summary**

|  |  |  |
| --- | --- | --- |
| Register | Offset | Description |
| EFC\_CR | 0x00 | Control Register |
| EFC\_INT\_EN | 0x04 | interruptEnableRegister |
| EFC\_SR | 0x08 | 状态Register |
| EFC\_PROG\_DATA0 | 0x0C | Program 编程数据 0 |
| EFC\_PROG\_DATA1 | 0x10 | Program 编程数据 1 |
| EFC\_TIMING\_CFG | 0x14 | clock 配置Register |
| EFC\_PROTECT\_SEQ | 0x18 | 保护序列 |
| RESERVED | 0x1C-0x28 | 保留 |
| SERIAL\_NUM\_LOW | 0x2C | 芯片序列号低 32 Bits |
| SERIAL\_NUM\_HIGH | 0x30 | 芯片序列号高 32 Bits |
| RESERVED | 0x34-0x38 | 保留 |
| OPTION\_CSR\_BYTES | 0x3C | OPTION 控制及状态数据 |
| OPTION\_EXE\_ONLY\_BYTES | 0x40 | OPTION 只执行数据 |
| OPTION\_WR\_PROTECT\_BYTES | 0x44 | OPTION 写保护数据 |
| OPTION\_SECURE\_BYTES0 | 0x48 | OPTION 安全数据 0 |
| OPTION\_SECURE\_BYTES1 | 0x4C | OPTION 安全数据 1 |

#### 10.5.1 EFC\_CR

Offset:0x00 Reset Value:0x00000000

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **31** | **30-10** | **9** | **8** | **7** | **6** |
| INFO\_BYTE\_LO  AD | RESERVED | ECC\_DIS | OPTION\_OPR  \_EN | RESERVED | WRITE\_RELEA  SE\_EN |
| w | r | r/w | r/w | r | r/w |
| **5** | **4** | **3** | **2** | **1** | **0** |
| PREFETCH\_EN | READ\_ACC\_EN | PROG\_MODE | PROG\_EN | PAGE\_ERA  SE\_EN | MASS\_ERASE  \_EN |
| r/w | r/w | r/w | r/w | r/w | r/w |

**Bits 31 INFO\_BYTE\_LOAD:**Info byte load 复BitsRequest .

* 写 0:无效
* 写 1:将引起系统复Bits,重新加载 Flash info 区中的信息,如 option 信息等,该Bits由硬件自动清零**Bits 30-10 Reserved:**Must be kept, and can't be modified.**Bits 9 ECC\_DIS:**ECC 编码除能控制.

**Bits 8 OPTION\_OPR\_EN:**Option 操作Enable.

* 0:Option 操作Disable
* 1:Option 操作Enable**Notice:**

1. *OPTION\_OPR\_EN, PROG\_EN*和*PAGE\_ERASE\_EN*任意两个不能同时配置Enable.
2. 每执行完一次*option*操作后,需要执行一次复Bits操作,配置才能真正生效.

**Bits 7 Reserved:** Must be kept, and can't be modified.

**Bits 6 WRITE\_RELEASE\_EN:**Flash 执行编程、擦除（包括 Mass）、Option 操作时,AHB 总线 hold/release 模式选择.

* 0:hold 模式
* 1:release 模式

**Notice:** 一旦配置为*release* 模式,则编程*/*擦除过程中不能读*Flash* 区域,也不能在*Flash* 中执行程序,否则操作会被拦截,返回数据未知,并且会起*FLASHBUSY\_ERR* 错误标志；但可以访问*SR* Register,等待操作完成.

**Bits 5 PREFETCH\_EN:**Flash 指令预取功能Enable.

* 0:Disable预取
* 1:Enable预取**Notice:** 预取Enable和读加速Enable控制不能同时开启.**Bits 4 READ\_ACC\_EN:**Flash 读操作加速Enable控制.
* 0:Disable读加速（hold 模式）
* 1:Enable读加速（release 模式）

##### Notice:

1. 当 *READ\_NUM < (2^HCLK\_DIV)*时,可以配置为 *1*,开启读加速功能；配置读加速Enable,需要在 *READ\_NUM*和*HCLK\_DIV*配置完成后进行.
2. 预取Enable和读加速Enable控制不要同时开启.

**Bits 3 PROG\_MODE:**flash program 模式控制.

* 0:单次编程模式,每次 program 将 EFC\_PROG\_DATA1 与 EFC\_PROG\_DATA0 写入指定地址.
* 1:WL 连续编程模式,该模式下会自动将 1 个 WL（512byte）中的连续地址均编程,软件需要查询

PROG\_DATA\_WAIT 标志来决定是否要将新的数据写入 EFC\_PROG\_DATA1 与 EFC\_PROG\_DATA0,以便于连续编程.

##### Notice:

1. *Flash*中的*ECC*编码格式为*64+8*,因此每次编程为偶数个*Word*.
2. *WL* 连续编程模式下需要将 *WRITE\_RELEASE\_EN* 配置为 *1*,并且编程过程中只能读写 *EFC\_SR, EFC\_PROG\_DATA1*与*EFC\_PROG\_DATA0*,不能读取*Flash*区域,也不能在*Flash*中执行程序.

**Bits 2 PROG\_EN:**flash program Enable控制.

* 0:对 Flash 存储空间的写操作不产生 flash program
* 1:对 Flash 存储空间的写操作产生 flash program **Notice:**

1. 单次编程模式下通过向*Flash*地址写入数据启动编程,*EFC\_PROG\_DATA0*的数据会写入该地址*8*字节对齐的低地址空间,*EFC\_PROG\_DATA1*写入该地址*8*字节对齐的高地址空间.
2. *WL* 连续编程模式下通过向*Flash* 地址写入数据启动编程,编程地址依次累加,直到一条*WL* 编程结束.

**Bits 1 PAGE\_ERASE\_EN:**flash page erase Enable控制.

* 0:对 Flash 存储空间的写操作不产生 flash page erase
* 1:对 Flash 存储空间的写操作产生 flash page erase,page 地址为写入地址所在的 PAGE **Bits 0 MASS\_ERASE\_EN:**flash mass erase Enable控制.
* 0:对 Flash 存储空间的写操作不产生 flash mass erase
* 1:对 Flash 存储空间的写操作产生 flash mass erase **Notice:**

1. 写操作的地址为*flash main*区地址则仅执行*main*区的*mass erase*；若写操作的地址为*flash info*区,则*main+info*区均执行*mass erase*.
2. 请不要对*info*区进行*mass erase*,否则芯片将被毁掉.

#### 10.5.2 EFC\_INT\_EN

Offset:0x04 Reset Value:0x00000000

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **31-9** | **8** | **7** | **6** | **5** |
| RESERVED | TWO\_BIT\_ERROR  \_INT\_EN | ONE\_BIT\_CORRE  CT\_INT\_EN | PROG\_ERR\_INT\_  EN | PAGE\_ERASE\_  ERR\_INT\_EN |
| r | r/w | r/w | r/w | r/w |
| **4** | **3** | **2** | **1** | **0** |
| OPTION\_WR\_ERR  \_INT\_EN | FLASHBUSY\_ERR  \_INT\_EN | PROG\_DATA\_WAI  T\_INT\_EN | RESERVED | OPERATION\_D  ONE\_INT\_EN |
| r/w | r/w | r/w | r | r/w |

**Bits 31-9 Reserved:** Must be kept, and can't be modified.

**Bits 8 TWO\_BIT\_ERROR\_INT\_EN:**ECC TWO\_BIT\_ERROR interruptEnable.

* 0:Disable
* 1:Enable**Bits 7 ONE\_BIT\_CORRECT\_INT\_EN:**ECC ONE\_BIT\_CORRECT interruptEnable.
* 0:Disable
* 1:Enable

**Bits 6 PROG\_ERR\_INT\_EN:**PROG\_ERR interruptEnable.

* 0:Disable
* 1:Enable

**Bits 5 PAGE\_ERASE\_ERR\_INT\_EN:**PAGE\_ERASE\_ERR interruptEnable.

* 0:Disable
* 1:Enable

**Bits 4 OPTION\_WR\_ERR\_INT\_EN:**OPTION\_WR\_ERR interruptEnable.

* 0:Disable
* 1:Enable

**Bits 3 FLASHBUSY\_ERR\_INT\_EN:**FLASHBUSY\_ERR interruptEnable.

* 0:Disable
* 1:Enable

**Bits 2 PROG\_DATA\_WAIT\_INT\_EN:**PROG\_DATA\_WAIT interruptEnable.

* 0:Disable
* 1:Enable**Bits 1 Reserved:** Must be kept, and can't be modified.

**Bits 0 OPERATION\_DONE\_INT\_EN:**OPERATION\_DONE interruptEnable.

* 0:Disable
* 1:Enable

#### 10.5.3 EFC\_SR

Offset:0x08 Reset Value:0x00000006

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **31-9** | **8** | **7** | **6** | **5** |
| RESERVED | TWO\_BIT\_ERROR | ONE\_BIT\_CORRECT | PROG\_ERR | PAGE\_ERASE\_ERR |
| r | r/w | r/w | r/w | r/w |
| **4** | **3** | **2** | **1** | **0** |
| OPTION\_WR\_  ERR | FLASHBUSY\_ERR | PROG\_DATA\_WAIT | READ\_NUM\_  DONE | OPERATION\_DONE |
| r/w | r/w | r/w | r | r/w |

**Bits 31-9 Reserved:** Must be kept, and can't be modified.

**Bits 8 TWO\_BIT\_ERROR:**Flash 读数据 two bit ECC 错误标志.

* 0:无 two bit 错误发生
* 1:Flash 读数据发生 two bit 错误,ECC 不纠正**Bits 7 ONE\_BIT\_CORRECT:**Flash 读数据单 bit ECC 错误纠正标志.
* 0:无错误发生
* 1:Flash 读数据发生单 bit 错误,ECC 进行了纠正**Bits 6 PROG\_ERR:**Flash info 部分区域不支持编程操作（PROG\_EN）,对这些区域的编程操作会被拦截,并且起该标志,硬件置 1,软件写 1 清 0.
* 0:无 Program 错误产生
* 1:产生 Program 错误**Notice:** *option*区域不能通过直接编程的方式实现写,需要选择Correspond to 操作；*bootloader*区域不能编程.

**Bits 5 PAGE\_ERASE\_ERR:**Flash info 区域不支持擦除操作,对 info 的擦除操作会被拦截,并且起该标志,硬件置 1,软件写 1 清 0.

* 0:无 Page erase 错误
* 1:发生了 Page erase 错误**Bits 4 OPTION\_WR\_ERR:**Option 区域的配置需要满足一定条件,不满足的 option 操作会被拦截,并且起该标志,硬件置 1,软件写 1 清 0.
* 0:无 Option byte 写权限错误
* 1:发生 Option byte 写权限错误

Option 区域配置约束条件为:

1. Flash exe\_only1/2 区域Enable时不能关闭或缩小；
2. exe\_only\_keep 不能由 0 写 1
3. secure\_area\_en Enable时非安全操作只能配置 option 将 secure\_area\_en 关闭

**Bits 3 FLASHBUSY\_ERR:**Flash 在执行编程、擦除（包括 mass）、option 操作时,软件发起了

Flash 空间读操作,则读操作会拦截,总线返回数据不确定,为异常状态,硬件置 1,软件写 1 清 0.

* 0:无错误发生
* 1:Flash 操作期间发生读操作错误

**Bits 2 PROG\_DATA\_WAIT:**Flash 连续编程模式下等待数据写入状态,硬件置 1,软件写 EFC\_PROG\_DATA0 和 EFC\_PROG\_DATA1 后硬件自动清零,或者软件写 1 清零.

* 0:EFC\_PROG\_DATA0 和 EFC\_PROG\_DATA1 数据写入完毕
* 1:等待 EFC\_PROG\_DATA0 和 EFC\_PROG\_DATA1 数据写入**Bits 1 READ\_NUM\_DONE:**READ\_NUM\_DONE 状态,指示 READ\_NUM 配置是否已完成.硬件控制置 1 与清 0.
* 0:未完成
* 1:完成**Bits 0 OPERATION\_DONE:**OPERATION\_DONE 状态,用于指示 mass erase、page erase、 program、option 操作是否完成.硬件置 1,软件写 1 清 0.
* 0:未完成
* 1:完成

#### 10.5.4 EFC\_PROG\_DATA0

Offset:0x0C

Reset Value:0x00000000

|  |
| --- |
| **31-0** |
| PROG\_DATA0 |
| r/w |

**Bits 31-0 PROG\_DATA0:**program 编程数据 0.**Notice:** *program*时先写入*EFC\_PROG\_DATA0*,再写入*EFC\_PROG\_DATA1*

#### 10.5.5 EFC\_PROG\_DATA1

Offset:0x10 Reset Value:0x00000000

|  |
| --- |
| **31-0** |
| PROG\_DATA1 |
| r/w |

**Bits 31-0 PROG\_DATA1:**program 编程数据 1.

**Notice:** *program*时先写入*EFC\_PROG\_DATA0*,再写入*EFC\_PROG\_DATA1*

#### 10.5.6 EFC\_TIMING\_CFG

Offset:0x14 Reset Value:0x00031D1D

|  |  |  |
| --- | --- | --- |
| **31-20** | **19-16** | **15-0** |
| RESERVED | READ\_NUM | RESERVED |
| r | r/w | r |

**Bits 31-20 Reserved:** Must be kept, and can't be modified.

**Bits 19-16 READ\_NUM:**flash 读操作读等待拍数控制,读等待拍数等于（READ\_NUM+1）个 SYS\_CLK clock 周期.

* SYS\_CLK 频率为 48 MHz 时,READ\_NUM 要大于等于 2  SYS\_CLK 频率为 32 MHz 时,READ\_NUM 要大于等于 1
* SYS\_CLK 频率为 24 MHz 时,READ\_NUM 要大于等于 1
* SYS\_CLK 频率为 4 MHz 时,READ\_NUM 要大于等于 0
* SYS\_CLK 频率为 32 KHz 时,READ\_NUM 要大于等于 0

**Notice:** 当修改 *RCC\_CR0* 中的 *SYS\_CLK* clock 源选择时,需要注意操作顺序:若切为快clock ,则先配置 *READ\_NUM* 变大,再配置clock 源选择；反之若切为慢clock 时,先配置clock 源选择,再配置*READ\_NUM* 变小.

**Bits 15-0 Reserved:** Must be kept, and can't be modified.

#### 10.5.7 EFC\_PROTECT\_SEQ

Offset:0x18 Reset Value:0x00000000

|  |
| --- |
| **31-0** |
| PROTECT\_SEQ |
| w |

**Bits 31-0 PROTECT\_SEQ:**EFC\_CR 配置的保护序列.

软件在操作 EFC\_CR 前需要按照以下顺序配置保护序列,才能够正确操作 EFC\_CR Register,中间插入了错误序列,则操作无效,需要按以下步骤重新配置:

1. 先写 0x8C9DAEBF
2. 再写 0x13141516
3. 然后才能操作 EFC\_CR

##### 10.5.8 SERIAL\_NUM\_LOW

Offset:0x2C

|  |
| --- |
| **31-0** |
| SERIAL\_NUM\_LOW |
| r |

**Bits 31-0 SERIAL\_NUM\_LOW:**芯片序列号的低 32 Bits.

##### 10.5.9 SERIAL\_NUM\_HIGH

Offset:0x30

|  |
| --- |
| **31-0** |
| SERIAL\_NUM\_HIGH |
| r |

**Bits 31-0 SERIAL\_NUM\_HIGH:**芯片序列号的高 32 Bits.

#### 10.5.10 OPTION\_CSR\_BYTES

Offset:0x3C Reset Value:0x000000BD

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| **31-7** | **6-5** | **4** | **3** | **2** | **1** | **0** |
| RESERVED | DEBUG\_LEVEL | SECURE\_AREA  \_EN | SYS\_SRAM  \_RST | FLASH\_ BOOT1 | USE\_FLASH  \_BOOT0 | FLASH\_ BOOT0 |
| r | r | r | r | r | r | r |

**Bits 31-7 Reserved:**Must be kept, and can't be modified.**Bits 6-5 DEBUG\_LEVEL:**Debug level 配置.

* 0:Level0
* 1:Level1
* 2:Level2

**Bits 4 SECURE\_AREA\_EN:**安全区域状态指示Register.

* 0:Disable
* 1:Enable**Bits 3 SYS\_SRAM\_RST:**System 复Bits后 Startup 过程中是否清除系统 SRAM.
* 1:清除系统 SRAM
* 0:不清除系统 SRAM **Bits 2 FLASH\_BOOT1:**用于 Boot 方式判断.
* 1:BootLoader 启动  0:系统 SRAM 启动

**Bits 1 USE\_FLASH\_BOOT0:**控制使用 option bit 或 IO 进行 Boot 方式判断.

* 0:使用 BOOT0 pin
* 1:使用 option bit FLASH\_BOOT0

**Bits 0 FLASH\_BOOT0:**USE\_FLASH\_BOOT0=1 时生效,用于 Boot 方式判断.

* 0:由 FLASH\_BOOT1 控制启动方式
* 1:Flash Main 启动

#### 10.5.11 OPTION\_EXE\_ONLY\_BYTES

Offset:0x40 Reset Value:0x00FC0FC0

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **31-25** | **24** | **23-18** | **17-12** | **11-6** | **5-0** |
| RESERVED | EXE\_ONLY\_K  EEP | EXE\_ONLY2\_  END | EXE\_ONLY2\_  START | EXE\_ONLY1\_  END | EXE\_ONLY1\_  START |
| r | r | r | r | r | r |

**Bits 31-25 Reserved:** Must be kept, and can't be modified.

**Bits 24 EXE\_ONLY\_KEEP:**Exe\_Only 区域在 Debug\_Level 由 1 变为 0 时,是否保留.

* 0:不保留 ExeOnly 区域
* 1:保留 ExeOnly 区域支持修改,但只能写 0.

**Bits 23-18 EXE\_ONLY2\_END:**Exe\_Only 区域 2 的结束 Page 地址.当 *EXEONLY2\_START > EXEONLY2\_END*,ExeOnly 区域 2 Disable.**Bits 17-12 EXE\_ONLY2\_START:**Exe\_Only 区域 2 的开始 Page 地址.

当 *EXEONLY2\_START > EXEONLY2\_END*,ExeOnly 区域 2 Disable.

ExeOnly 区域 2 支持修改,但只能Enable或扩大区域,不能减少或关闭.**Bits 11-6 EXE\_ONLY1\_END:**Exe\_Only 区域 1 的结束 Page 地址.当 *EXEONLY1\_START > EXEONLY1\_END*,ExeOnly 区域 1 Disable.**Bits 5-0 EXE\_ONLY1\_START:**Exe\_Only 区域 1 的开始 Page 地址.当 *EXEONLY1\_START > EXEONLY1\_END*,ExeOnly 区域 1 Disable.

ExeOnly 区域 1 支持修改,但只能Enable或扩大区域,不能减少或关闭.

#### 10.5.12 OPTION\_WR\_PROTECT\_BYTES

Offset:0x44 Reset Value:0x0003F03F

|  |  |  |
| --- | --- | --- |
| **31-12** | **11-6** | **5-0** |
| RESERVED | WRPROTECT\_END | WRPROTECT\_START |
| r | r | r |

**Bits 31-12 Reserved:** Must be kept, and can't be modified.

**Bits 11-6 WRPROTECT\_END:**WrProtect 区域的结束 Page 地址.当 *WRPROTECT\_START > WRPROTECT\_END*,WrProtect 区域Disable.**Bits 5-0 WRPROTECT\_START:**WrProtect 区域的开始 Page 地址.当 *WRPROTECT\_START > WRPROTECT\_END*,WrProtect 区域Disable.

#### 10.5.13 OPTION\_SECURE\_BYTES0

Offset:0x48 Reset Value:0x00FC0FC0

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **31-24** | **23-18** | **17-12** | **11-6** | **5-0** |
| RESERVED | SYSRAM\_SECURE\_  END | SYSRAM\_SECURE\_  START | FLASH\_SECURE\_  END | FLASH\_SECURE\_  START |
| r | r | r | r | r |

**Bits 31-24 Reserved:** Must be kept, and can't be modified.

**Bits 23-18 SYSRAM\_SECURE\_END:**SysRam Secure 区域的结束地址.当 *SYSRAM\_SECURE\_START > SYSRAM\_SECURE\_END* 时,该区域安全Disable；且仅在 SECURE\_ AREA\_EN=1 才有效.

**Bits 17-12 SYSRAM\_SECURE\_START:**SysRam Secure 区域的起始地址.当 *SYSRAM\_SECURE\_START > SYSRAM\_SECURE\_END* 时,该区域安全Disable；且仅在 SECURE\_ AREA\_EN=1 才有效.

**Bits 11-6 FLASH\_SECURE\_END:**Flash Secure 区域的结束地址.当 *FLASH\_SECURE\_START > FLASH\_SECURE\_END* 时,该区域安全Disable.**Bits 5-0 FLASH\_SECURE\_START:**FLASH Secure 区域的起始地址.

当 *FLASH\_SECURE\_START > FLASH\_SECURE\_END* 时,该区域安全Disable. FLASH 区域安全Enable为整个安全区域的总Enable:当 FLASH 区域安全Enable时,SECURE\_AREA\_EN 为 1,Enable整个安全区域；当 FLASH 区域安全去Enable时,SECURE\_AREA\_EN 由 1 变为 0,将触发擦除操作.

#### 10.5.14 OPTION\_SECURE\_BYTES1

Offset:0x4C Reset Value:0x008103E0

|  |  |  |  |
| --- | --- | --- | --- |
| **31-24** | **23** | | **22-17** |
| RESERVED | SYSRAM\_HIDE\_ENABLE | | SYSRAM\_HIDE\_START |
| r | r | | r |
| **16** | **15-10** | **9-5** | **4-0** |
| FLASH\_HIDE\_ENABLE | FLASH\_HIDE\_START | RETRAM\_SECURE\_  END | RETRAM\_SECURE\_START |
| r | r | r | r |

**Bits 31-24 Reserved:** Must be kept, and can't be modified.

**Bits 23 SYSRAM\_HIDE\_ENABLE:**SysRamHide 区域Enable控制.

* 0:SysRamHide 区域Enable
* 1:SysRamHide 区域Disable仅当 SECURE\_AREA\_EN=1 时才有效.

**Bits 22-17 SYSRAM\_HIDE\_START:**SysRamHide 区域的开始 Page 地址.必须在 SysRamSecure 区域内才Enable,区域范围由 SYSRAM\_HIDE\_START 开始一直到 SYSRAM\_

SECURE\_END； 仅当 SECURE\_AREA\_EN=1 时才有效.**Bits 16 FLASH\_HIDE\_ENABLE:**FlashHide 区域Enable控制.

* 0:FlashHide 区域Enable
* 1:FlashHide 区域Disable仅当 SECURE\_AREA\_EN=1 时才有效.

**Bits 15-10 FLASH\_HIDE\_START:**FlashHide 区域的开始 Page 地址.必须在 FlashSecure 区域内才Enable,区域范围由 FLASH\_HIDE\_START开始一直到 FLASH\_SECURE\_END；仅当 SECURE\_AREA\_EN=1 时才有效.

**Bits 9-5 RETRAM\_SECURE\_END:**RetRam Secure 区域的结束地址.当 *RETRAM\_SECURE\_START > RETRAM\_SECURE\_END* 时,该区域安全Disable；仅当 SECURE\_AREA\_EN=1 时才有效.

**Bits 4-0 RETRAM\_SECURE\_START:**RetRam Secure 区域的开始地址.当 *RETRAM\_SECURE\_START > RETRAM\_SECURE\_END* 时,该区域安全Disable；仅当 SECURE\_AREA\_EN=1 时才有效.

### 11. 通用输入输出 (GPIO)

#### 11.1 Introduction

共 A、B、C、D 四组 GPIO,每组的 SFR Register分配一致,利用不同基地址进行区分；PortD

Pin8~PortD Pin15 Bits于 AlwayOn 域,其他 IO Bits于 Main 域.

支持输入输出功能,支持上下拉功能,支持推挽输出和开漏输出,可以配置输出驱动电流为 4mA/8mA, 所有 IO 都支持interrupt功能,上升沿、下降沿或双沿可配,Sleep/Stop0~2 模式所有

IO 都可以唤醒,Stop3 模式部分 IO 可唤醒.所有 GPIO 均支持功能复用配置.

#### 11.2 输出配置

支持输出数据配置,通过配置输出EnableRegister GPIOx\_OER 和输出数据Register GPIOx\_ODR 实现.

支持输出数据置Bits和清零两种Bits操作, 通过向输出数据清零Register GPIOx\_BRR 写 1 来清除 GPIOx\_ODR 的相应Bits,或者向输出数据置Bits清零Register GPIOx\_BSRR 低 16 Bits写 1 置Bits GPIOx\_ODR 的相应Bits,向 GPIOx\_BSRR 高 16 Bits写 1 清零 GPIOx\_ODR 的相应Bits,只操作指定Bits,避免读后写.

支持推挽输出,通过配置输出类型Register GPIOx\_OTYPER 实现.支持开漏输出,其中 PortD Pin8~PortD Pin15 通过配置 GPIOx\_IER/GPIOx\_OER/GPIOx\_ODR/GPIOx\_PSR 实现,其他的 IO 通过配置 GPIOx\_OER/GPIOx\_IER/GPIOx\_ODR/GPIOx\_OTYPER 实现.不支持真的

Open Drain 结构,通过配置 GPIOx\_OER 和 GPIOx\_ODR 实现.

支持配置为模拟输出.

#### 11.3 输入配置

支持输入数据配置,通过配置输入EnableRegister GPIOx\_IER Enable输入功能,读取输入数据Register GPIOx\_IDR 获取输入状态.

支持 Floating 输入方式,通过配置上下拉EnableRegister GPIOx\_PER 关闭上下拉功能实现.

支持上拉输入和下拉输入,配置 GPIOx\_PER Enable上下拉功能,配置上下拉选择Register

GPIOx\_PSR 选择上拉或者下拉功能.

支持配置为模拟输入.

#### 11.4 输出驱动能力

可配为低驱动能力 4mA 或高驱动能力 8mA,通过配置输出驱动能力Register GPIOx\_DSR 实现.

##### 11.5 interrupt

所有 IO 都支持上升沿、下降沿和双沿interrupt,interruptEnable可配,通过配置interruptEnableRegister GPIOx\_

INT\_CR 实现.

##### 11.6 Sleep/Stop0~2 唤醒Request

支持高电平与低电平,输出唤醒信号为高电平.GPIO00~GPIO63 均可用于唤醒,每 4 个 IO为一组,一组可以产生一个唤醒信号,同组内的每个IO均支持配置高电平或者低电平唤醒.通过配置 Sleep/Stop0~2 唤醒EnableRegister GPIOx\_WU\_EN Enable唤醒功能,通过配置 Sleep/Stop0~2 唤醒电平Control Register选择高电平或者低电平唤醒.

##### 11.7 Stop3 唤醒Request

Main 域的 GPIO00~GPIO55,每 4 个 IO MUX 出一个唤醒信号,共 14 个唤醒信号.支持高电平唤醒或低电平唤醒和唤醒Enable控制,通过配置 Stop3 唤醒EnableRegister GPIOx\_STOP3\_WU\_

CR 实现.

#### 11.8 复用功能控制

可在 GPIO 以及不同外设之间选择,GPIO 下的输入输出Enable由 GPIO Register控制,外设下的输入输出控制由外设实现,上下拉由 GPIO Register实现.

每个 IO 均有 4bit 的复用功能控制（PortD Pin8~PortD Pin15 为每个 3bit）,除 PortA Pin6 和

PortA Pin7 默认选择 SWD 外,其余 IO 均默认选择 GPIO 功能.

通过低 8 Pin 功能 MUX 选择Register GPIOx\_AFRL 配置 Portx Pin[7:0] 的功能,通过高 8 Pin 功能 MUX 选择Register GPIOx\_AFRH 配置 Portx Pin[15:8] 的功能.

#### 11.9 clock 复Bits

APB 总线clock ,共 4 组,每组均有独立的总线clock .APB 总线复Bits,共 4 组,每组均有独立的总线复Bits.

##### 11.10 电源域

**Main 域:**除了 PortD Pin8~PortD Pin15 外相应的 PAD 均在 Main 域.

###### AlwayOn 域:

PortD Pin8~PortD Pin15 Correspond to 的 PAD Bits于 AlwaysOn 域,如果配置为 Function 模式,那么直接由外设去控制,如果没有配置为 Function 模式,需要由 AlwaysOn 的 GPIO Register控制.

#### 11.11 低功耗工作与唤醒

1. Sleep 下所有 GPIO 均可工作,可输出唤醒信号.
2. Stop0~2 下所有 GPIO 均可工作,可输出唤醒信号.
3. Stop3 下 GPIO00~GPIO55 工作状态可保持,并且均可配置为唤醒信号.
4. Stop3 下 AlwaysOn 域的 PortD Pin8~PortD Pin15 可保持,也可以通过 RTC 唤醒.
5. Standby 下 PortD Pin8~PortD Pin15 可工作,其他 IO 不可工作.

##### 11.12 SWD IO

**默认控制:**控制 GPIO MUX 的Register默认应选择 SWD,并且上下拉默认值应为 SWC-下拉

（PortA Pin7）,SWD-上拉（PortA Pin6）.

**封口控制:**上电时由Register默认态控制 IO 状态,直到 DebugLevel 判断完毕,若发现需要封口,则执行永久封口；否则,继续由Register控制.

**软件配置:**软件运行过程中,可通过复用Register控制关闭 SWD,注意是单向封口的,即只能控制关闭,不能关闭后再Enable.

##### 11.13 BOOT0 的控制

**默认控制:**由于除了 SWC 和 SWD 两个 IO 外,其他 IO 默认都是模拟 IO,因此上电时这三个 IO 需要特殊控制.

**BOOT0 (GPIO02):**BOOT0 在 io\_lock 前为输入下拉,待 EFC 锁定后,切换为 GPIO 控制.

##### 11.14 GPIO 相关RegisterDescription

GPIO Port A 基地址:0x4001F000

GPIO Port B 基地址:0x4001F400

GPIO Port C 基地址:0x4001F800 GPIO Port D 基地址:0x4001FC00

**Table 11-1 GPIO Register Summary**

|  |  |  |
| --- | --- | --- |
| Register | Offset | Description |
| GPIOx\_OER | 0x00 | 通用输出EnableRegister |
| GPIOx\_OTYPER | 0x04 | 通用输出类型Control Register |
| GPIOx\_IER | 0x08 | 通用输入EnableRegister |
| GPIOx\_PER | 0x0C | 上下拉EnableRegister |
| GPIOx\_PSR | 0x10 | 上下拉选择Register |
| GPIOx\_IDR | 0x14 | 输入数据Register |
| GPIOx\_ODR | 0x18 | 输出数据Register |
| GPIOx\_BRR | 0x1C | 输出数据清零Register |
| GPIOx\_BSRR | 0x20 | 输出数据置Bits和清零Register |
| GPIOx\_DSR | 0x24 | 输出驱动能力Register |
| GPIOx\_INT\_CR | 0x28 | interruptEnableRegister |
| GPIOx\_FR | 0x2C | interrupt沿标志Register |
| GPIOx\_WU\_EN | 0x30 | Sleep/Stop0~2 唤醒EnableRegister |
| GPIOx\_WU\_LVL | 0x34 | Sleep/Stop0~2 唤醒电平Control Register |
| GPIOx\_AFRL | 0x38 | 低 8 Pin 功能 MUX 选择Register |
| GPIOx\_AFRH | 0x3C | 高 8 Pin 功能 MUX 选择Register |
| GPIOx\_STOP3\_WU\_CR | 0x40 | Stop3 唤醒EnableControl Register |

###### 11.14.1 GPIOx\_OER (x=A, B, C, D)

Offset:0x00 Reset Value:0x0000FFFF

|  |  |
| --- | --- |
| **31-16** | **15-0** |
| RESERVED | OEN |
| r-0h | rw-ffffh |

**Bits 31-16 RESERVED:**保留,不可更改.**Bits 15-0 OEN:**Portx Pin[15:0] 输出Enable.

* 0:输出Enable有效
* 1:输出Enable无效

###### 11.14.2 GPIOx\_OTYPER (x=A, B, C, D)

Offset:0x04 Reset Value:0x00000000

|  |  |
| --- | --- |
| **31-16** | **15-0** |
| RESERVED | OTYPE |
| r-0h | rw-0h |

**Bits 31-16 RESERVED:**保留,不可更改.**Bits 15-0 OTYPE:**Portx Pin[15:0] 输出类型控制.

* 0:推挽输出
* 1:开漏输出**说明:***AON*的*PAD*（*PortD\_Pin[15:8]*）不支持该Register的配置,而是通过*GPIOx\_OER*直接控制*PAD*的 *OEN*端.其他*PAD*的开漏模式可以通过配置*GPIOx\_ODR/GPIOx\_IER/GPIOx\_OER/GPIOx\_OTYPER*的组合去实现.

###### 11.14.3 GPIOx\_IER (x=A, B, C, D)

Offset:0x08 Reset Value:0x00000000

|  |  |
| --- | --- |
| **31-16** | **15-0** |
| RESERVED | IE |
| r-0h | rw-0h |

**Bits 31-16 RESERVED:**保留,不可更改.**Bits 15-0 IE:**Portx Pin[15:0] 输入Enable.

* 0:输入Enable无效
* 1:输入Enable有效

###### 11.14.4 GPIOx\_PER (x=A, B, C, D)

Offset:0x0C

Reset Value:0x00000000

|  |  |
| --- | --- |
| **31-16** | **15-0** |
| RESERVED | PE |
| r-0h | rw-0h |

**Bits 31-16 RESERVED:**保留,不可更改.**Bits 15-0 PE:**Portx Pin[15:0] 上下拉Enable.

* 0:上下拉Enable无效
* 1:上下拉Enable有效上下拉直接由Register控制.默认上下拉Disable,IO 处于模拟模式,PortA\_Pin[7:6] 例外,为 SWD 接口.

###### 11.14.5 GPIOx\_PSR (x=A, B, C, D)

Offset:0x10 Reset Value:0x00000000

|  |  |
| --- | --- |
| **31-16** | **15-0** |
| RESERVED | PS |
| r-0h | rw-0h |

**Bits 31-16 RESERVED:**保留,不可更改.**Bits 15-0 PS:**Portx Pin[15:0] 上下拉选择.

* 0:选择下拉
* 1:选择上拉

###### 11.14.6 GPIOx\_IDR (x=A, B, C, D)

Offset:0x14 Reset Value:0x00000000

|  |  |
| --- | --- |
| **31-16** | **15-0** |
| RESERVED | ID |
| r-0h | r-0h |

**Bits 31-16 RESERVED:**保留,不可更改.**Bits 15-0 ID:**Portx Pin[15:0]输入数据.

* 0:输入低
* 1:输入高

###### 11.14.7 GPIOx\_ODR (x=A, B, C, D)

Offset:0x18 Reset Value:0x00000000

|  |  |
| --- | --- |
| **31-16** | **15-0** |
| RESERVED | OD |
| r-0h | rw-0h |

**Bits 31-16 RESERVED:**保留,不可更改.**Bits 15-0 OD:**Portx Pin[15:0]输出数据.

* 0:输出低
* 1:输出高

###### 11.14.8 GPIOx\_BRR (x=A, B, C, D)

Offset:0x1C

Reset Value:0x00000000

|  |  |
| --- | --- |
| **31-16** | **15-0** |
| RESERVED | BR |
| r-0h | w-0h |

**Bits 31-16 RESERVED:**保留,不可更改.**Bits 15-0 BR:**Portx Pin[15:0] 输出数据清零.

* 写 0:无效
* 写 1:清除 GPIOx\_ODR 相应Bits

###### 11.14.9 GPIOx\_BSRR (x=A, B, C, D)

Offset:0x20 Reset Value:0x00000000

|  |  |
| --- | --- |
| **31-16** | **15-0** |
| BR | BSR |
| w-0h | w-0h |

**Bits 31-16 BR:**Portx Pin[15:0] 输出数据清零.

* 写 0:无效
* 写 1:清除 GPIOx\_ODR 相应Bits**说明:**若*BSR*和*BR*同时有效,则*BSR*优先级高.

**Bits 15-0 BSR:**Portx Pin[15:0] 输出数据置Bits.

* 写 0:无效
* 写 1:置Bits GPIOx\_ODR 相应Bits**说明:**若*BSR*和*BR*同时有效,则*BSR*优先级高.

11.14.10 GPIOx\_DSR (x=A, B, C, D)

Offset:0x24 Reset Value:0x00000000

|  |  |
| --- | --- |
| **31-16** | **15-0** |
| RESERVED | DS |
| r-0h | w-0h |

**Bits 31-16 RESERVED:**保留,不可更改.**Bits 15-0 DS:**Portx Pin[15:0] 输出驱动能力配置.

* 写 0:低驱动能力,4mA
* 写 1:高驱动能力,8mA

###### 11.14.11 GPIOx\_INT\_CR (x=A, B, C, D)

Offset:0x28 Reset Value:0x00000000

|  |  |
| --- | --- |
| **2\*n + 1** | **2\*n** |
| NEG\_INT\_EN | POS\_INT\_EN |
| rw-0h | rw-0h |

**Bits 2\*n + 1 NEG\_INT\_EN:**Portx Pin[15:0] 下降沿interruptEnable.

* 0:Disable下降沿interrupt
* 1:Enable下降沿interrupt

**Bits 2\*n POS\_INT\_EN:**Portx Pin[15:0] 上升沿interruptEnable.

* 0:Disable上升沿interrupt
* 1:Enable上升沿interrupt

###### 11.14.12 GPIOx\_FR (x=A, B, C, D)

Offset:0x2C

Reset Value:0x00000000

|  |  |
| --- | --- |
| **2\*n + 1** | **2\*n** |
| NEG\_F | POS\_F |
| rw1c-0h | rw1c-0h |

**Bits 2\*n + 1 NEG\_INT\_EN:**Portx Pin[15:0] 下降沿interrupt标志.

* 0:未发生下降沿interrupt
* 1:发生下降沿interrupt

**Bits 2\*n POS\_INT\_EN:**Portx Pin[15:0] 上升沿interrupt标志.

* 0:未发生上升沿interrupt
* 1:发生上升沿interrupt

###### 11.14.13 GPIOx\_WU\_EN (x=A, B, C, D)

Offset:0x30 Reset Value:0x00000000

|  |  |
| --- | --- |
| **31-16** | **15-0** |
| RESERVED | WU\_EN |
| r-0h | rw-0h |

**Bits 31-16 RESERVED:**保留,不可更改.

**Bits 15-0 WU\_EN:**Portx Pin[15:0] Sleep/Stop0~2 唤醒Enable.

* 0:Disable Sleep/Stop0~2 唤醒
* 1:Enable Sleep/Stop0~2 唤醒

###### 11.14.14 GPIOx\_WU\_LVL (x=A, B, C, D)

Offset:0x34 Reset Value:0x00000000

|  |  |
| --- | --- |
| **31-16** | **15-0** |
| RESERVED | WU\_LVL |
| r-0h | rw-0h |

**Bits 31-16 RESERVED:**保留,不可更改.**Bits 15-0 WU\_LVL:**Portx Pin[15:0] Sleep/Stop0~2 电平控制.

* 0:低电平唤醒
* 1:高电平唤醒

###### 11.14.15 GPIOx\_AFRL (x=A, B, C, D)

Offset:0x38 Reset Value:0x00000000

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| **31-28** | **27-24** | **23-20** | **19-16** | **15-12** | **11-8** | **7-4** | **3-0** |
| AF7 | AF6 | AF5 | AF4 | AF3 | AF2 | AF1 | AF0 |
| rw-0h | rw-0h | rw-0h | rw-0h | rw-0h | rw-0h | rw-0h | rw-0h |

**Bits 31-28 AF7:**Portx Pin7 功能 MUX 选择.

* 0000:Function0
* 0001:Function1
* 0010:Function2
* 0011:Function3
* 0100:Function4
* 0101:Function5
* 0110:Function6
* 0111:Function7
* others:Reserved

**Bits 27-24 AF6:**Portx Pin6 功能 MUX 选择.

* 0000:Function0
* 0001:Function1
* 0010:Function2
* 0011:Function3
* 0100:Function4
* 0101:Function5
* 0110:Function6
* 0111:Function7
* others:Reserved

**Bits 23-20 AF5:**Portx Pin5 功能 MUX 选择.

* 0000:Function0
* 0001:Function1
* 0010:Function2
* 0011:Function3
* 0100:Function4
* 0101:Function5
* 0110:Function6
* 0111:Function7
* others:Reserved

**Bits 19-16 AF4:**Portx Pin4 功能 MUX 选择.

* 0000:Function0
* 0001:Function1
* 0010:Function2**（未完,接下页）**
* 0011:Function3
* 0100:Function4
* 0101:Function5
* 0110:Function6
* 0111:Function7
* others:Reserved

**Bits 15-12 AF3:**Portx Pin3 功能 MUX 选择.

* 0000:Function0
* 0001:Function1
* 0010:Function2
* 0011:Function3
* 0100:Function4
* 0101:Function5
* 0110:Function6
* 0111:Function7
* others:Reserved

**Bits 11-8 AF2:**Portx Pin2 功能 MUX 选择.

* 0000:Function0
* 0001:Function1
* 0010:Function2
* 0011:Function3
* 0100:Function4
* 0101:Function5
* 0110:Function6
* 0111:Function7
* others:Reserved

**Bits 7-4 AF1:**Portx Pin1 功能 MUX 选择.

* 0000:Function0
* 0001:Function1
* 0010:Function2
* 0011:Function3
* 0100:Function4
* 0101:Function5
* 0110:Function6
* 0111:Function7
* others:Reserved

**Bits 3-0 AF0:**Portx Pin0 功能 MUX 选择.

* 0000:Function0
* 0001:Function1
* 0010:Function2
* 0011:Function3
* 0100:Function4
* 0101:Function5**（未完,接下页）**

0110:Function6 0111:Function7 others:Reserved

###### 11.14.16 GPIOx\_AFRH (x=A, B, C)

Offset:0x3C

Reset Value:0x00000000

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| **31-28** | **27-24** | **23-20** | **19-16** | **15-12** | **11-8** | **7-4** | **3-0** |
| AF15 | AF14 | AF13 | AF12 | AF11 | AF10 | AF9 | AF8 |
| rw-0h | rw-0h | rw-0h | rw-0h | rw-0h | rw-0h | rw-0h | rw-0h |

**Bits 31-28 AF15:**Portx Pin15 功能 MUX 选择.

* 0000:Function0
* 0001:Function1
* 0010:Function2
* 0011:Function3
* 0100:Function4
* 0101:Function5
* 0110:Function6
* 0111:Function7
* others:Reserved

**Bits 27-24 AF14:**Portx Pin14 功能 MUX 选择.

* 0000:Function0
* 0001:Function1
* 0010:Function2
* 0011:Function3
* 0100:Function4
* 0101:Function5
* 0110:Function6
* 0111:Function7
* others:Reserved

**Bits 23-20 AF13:**Portx Pin13 功能 MUX 选择.

* 0000:Function0
* 0001:Function1
* 0010:Function2
* 0011:Function3
* 0100:Function4
* 0101:Function5
* 0110:Function6
* 0111:Function7
* others:Reserved

**Bits 19-16 AF12:**Portx Pin12 功能 MUX 选择.

* 0000:Function0
* 0001:Function1
* 0010:Function2
* 0011:Function3
* 0100:Function4
* 0101:Function5
* 0110:Function6
* 0111:Function7
* others:Reserved

**Bits 15-12 AF11:**Portx Pin11 功能 MUX 选择.

* 0000:Function0
* 0001:Function1
* 0010:Function2
* 0011:Function3
* 0100:Function4
* 0101:Function5
* 0110:Function6
* 0111:Function7
* others:Reserved

**Bits 11-8 AF10:**Portx Pin10 功能 MUX 选择.

* 0000:Function0
* 0001:Function1
* 0010:Function2
* 0011:Function3
* 0100:Function4
* 0101:Function5
* 0110:Function6
* 0111:Function7
* others:Reserved

**Bits 7-4 AF9:**Portx Pin9 功能 MUX 选择.

* 0000:Function0
* 0001:Function1
* 0010:Function2
* 0011:Function3
* 0100:Function4
* 0101:Function5
* 0110:Function6
* 0111:Function7
* others:Reserved

**Bits 3-0 AF8:**Portx Pin8 功能 MUX 选择.

* 0000:Function0
* 0001:Function1**（未完,接下页）** 0010:Function2

0011:Function3

0100:Function4

0101:Function5

* 0110:Function6
* 0111:Function7
* others:Reserved

###### 11.14.17 GPIOD\_AFRH

Offset:0x3C

Reset Value:0x00000000

|  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- |
| **31-24** | **23-21** | **20-18** | **17-15** | **14-12** | **11-9** | **8-6** | **5-3** | **2-0** |
| RESERVED | AF15 | AF14 | AF13 | AF12 | AF11 | AF10 | AF9 | AF8 |
| r-0h | rw-0h | rw-0h | rw-0h | rw-0h | rw-0h | rw-0h | rw-0h | rw-0h |

**Bits 31-24 RESERVED:**保留,不可更改.

* 001:Function1
* 010:Function2
* 011:Function3
* 100:Function4
* 101:Function5
* 110:Function6
* 111:Function7

**Bits 23-21 AF15:**PortD Pin15 功能 MUX 选择.

* 001:Function1
* 010:Function2
* 011:Function3
* 100:Function4
* 101:Function5
* 110:Function6
* 111:Function7

**Bits 20-18 AF14:**PortD Pin14 功能 MUX 选择.

* 001:Function1
* 010:Function2
* 011:Function3
* 100:Function4
* 101:Function5
* 110:Function6
* 111:Function7

**Bits 17-15 AF13:**PortD Pin13 功能 MUX 选择.

* 001:Function1**（未完,接下页）**

010:Function2

011:Function3

100:Function4

101:Function5

* 110:Function6
* 111:Function7

**Bits 14-12 AF12:**PortD Pin12 功能 MUX 选择.

* 001:Function1
* 010:Function2
* 011:Function3
* 100:Function4
* 101:Function5
* 110:Function6
* 111:Function7

**Bits 11-9 AF11:**PortD Pin11 功能 MUX 选择.

* 001:Function1
* 010:Function2
* 011:Function3
* 100:Function4
* 101:Function5
* 110:Function6
* 111:Function7

**Bits 8-6 AF10:**PortD Pin10 功能 MUX 选择.

* 001:Function1
* 010:Function2
* 011:Function3
* 100:Function4
* 101:Function5
* 110:Function6
* 111:Function7

**Bits 5-3 AF9:**PortD Pin9 功能 MUX 选择.

* 001:Function1
* 010:Function2
* 011:Function3
* 100:Function4
* 101:Function5
* 110:Function6
* 111:Function7

**Bits 2-0 AF8:**PortD Pin8 功能 MUX 选择.

* 001:Function1
* 010:Function2
* 011:Function3**（未完,接下页）** 100:Function4

101:Function5

110:Function6

111:Function7

###### 11.14.18 GPIOA\_STOP3\_WU\_CR

Offset:0x40 Reset Value:0x00000000

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **31-16** | **15** | **14** | **13-12** | **11** |
| RESERVED | STOP3\_WU\_EN  \_G1 | STOP3\_WU\_LVL\_G3 | STOP3\_WU\_SEL\_G  3 | STOP3\_WU\_EN\_G2 |
| r-0h | rw-0h | rw-0h | rw-0h | rw-0h |
| **10** | | **9-8** | **7** | **6** |
| STOP3\_WU\_LVL\_G2 | | STOP3\_WU\_SEL\_G  2 | STOP3\_WU\_EN\_G1 | STOP3\_WU\_LVL\_G  1 |
| rw-0h | | rw-0h | rw-0h | rw-0h |
| **5-4** | | **3** | **2** | **1-0** |
| STOP3\_WU\_SEL\_G1 | | STOP3\_WU\_EN\_G0 | STOP3\_WU\_LVL\_G  0 | STOP3\_WU\_SEL\_G  0 |
| rw-0h | | rw-0h | rw-0h | rw-0h |

**Bits 31-16 RESERVED:**保留,不可更改.

**Bits 15 STOP3\_WU\_EN\_G3:**PortA Pin Group3 Stop3 唤醒Enable控制.

* 0:PortA Pin[15/14/7/6] 唤醒Disable
* 1:PortA Pin[15/14/7/6] 唤醒Enable

**Bits 14 STOP3\_WU\_LVL\_G3:**PortA Pin Group3 Stop3 唤醒电平选择.

* 0:PortA Pin[15/14/7/6] 低电平唤醒
* 1:PortA Pin[15/14/7/6] 高电平唤醒

**Bits 13-12 STOP3\_WU\_SEL\_G3:**PortA Pin Group3 Stop3 Wakeup Source选择.

* 00:选择 PortA Pin6
* 01:选择 PortA Pin7
* 10:选择 PortA Pin14
* 11:选择 PortA Pin15

**Bits 11 STOP3\_WU\_EN\_G2:**PortA Pin Group2 Stop3 唤醒Enable控制.

* 0:PortA Pin[11:8] 唤醒Disable
* 1:PortA Pin[11:8] 唤醒Enable**Bits 10 STOP3\_WU\_LVL\_G2:**PortA Pin Group2 Stop3 唤醒电平选择.
* 0:PortA Pin[11:8] 低电平唤醒
* 1:PortA Pin[11:8] 高电平唤醒

**Bits 9-8 STOP3\_WU\_SEL\_G2:**PortA Pin Group2 Stop3 Wakeup Source选择.

* 00:选择 PortA Pin8
* 01:选择 PortA Pin9
* 10:选择 PortA Pin10
* 11:选择 PortA Pin11

**Bits 7 STOP3\_WU\_EN\_G1:**PortA Pin Group1 Stop3 唤醒Enable控制.

* 0:PortA Pin[13/12/5/4] 唤醒Disable
* 1:PortA Pin[13/12/5/4] 唤醒Enable

**Bits 6 STOP3\_WU\_LVL\_G1:**PortA Pin Group1 Stop3 唤醒电平选择.

* 0:PortA Pin[13/12/5/4] 低电平唤醒
* 1:PortA Pin[13/12/5/4] 高电平唤醒

**Bits 5-4 STOP3\_WU\_SEL\_G1:**PortA Pin Group1 Stop3 Wakeup Source选择.

* 00:选择 PortA Pin4
* 01:选择 PortA Pin5
* 10:选择 PortA Pin12
* 11:选择 PortA Pin13

**Bits 3 STOP3\_WU\_EN\_G0:**PortA Pin Group0 Stop3 唤醒Enable控制.

* 0:PortA Pin[3:0] 唤醒Disable
* 1:PortA Pin[3:0] 唤醒Enable**Bits 2 STOP3\_WU\_LVL\_G0:**PortA Pin Group0 Stop3 唤醒电平选择.
* 0:PortA Pin[3:0] 低电平唤醒
* 1:PortA Pin[3:0] 高电平唤醒**Bits 1-0 STOP3\_WU\_SEL\_G0:**PortA Pin Group0 Stop3 Wakeup Source选择.
* 00:选择 PortA Pin0
* 01:选择 PortA Pin1
* 10:选择 PortA Pin2
* 11:选择 PortA Pin3

###### 11.14.19 GPIOx\_STOP3\_WU\_CR (x=B, C)

Offset:0x40 Reset Value:0x00000000

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **31-16** | **15** | **14** | **13-12** | **11** |
| RESERVED | STOP3\_WU\_EN  \_G3 | STOP3\_WU\_LVL\_G3 | STOP3\_WU\_SEL\_G  3 | STOP3\_WU\_EN\_G2 |
| r-0h | rw-0h | rw-0h | rw-0h | rw-0h |
| **10** | | **9-8** | **7** | **6** |
| STOP3\_WU\_LVL\_G2 | | STOP3\_WU\_SEL\_G  2 | STOP3\_WU\_EN\_G1 | STOP3\_WU\_LVL\_G  1 |
| rw-0h | | rw-0h | rw-0h | rw-0h |
| **5-4** | | **3** | **2** | **1-0** |
| STOP3\_WU\_SEL\_G1 | | STOP3\_WU\_EN\_G0 | STOP3\_WU\_LVL\_G  0 | STOP3\_WU\_SEL\_G  0 |
| rw-0h | | rw-0h | rw-0h | rw-0h |

**Bits 31-16 RESERVED:**保留,不可更改.

**Bits 15 STOP3\_WU\_EN\_G3:**Portx Pin Group3 Stop3 唤醒Enable控制.

* 0:Portx Pin[15:12] 唤醒Disable
* 1:Portx Pin[15:12] 唤醒Enable**Bits 14 STOP3\_WU\_LVL\_G3:**Portx Pin Group3 Stop3 唤醒电平选择.
* 0:Portx Pin[15:12] 低电平唤醒
* 1:Portx Pin[15:12] 高电平唤醒**Bits 13-12 STOP3\_WU\_SEL\_G3:**Portx Pin Group3 Stop3 Wakeup Source选择.
* 00:选择 Portx Pin12
* 01:选择 Portx Pin13
* 10:选择 Portx Pin14
* 11:选择 Portx Pin15

**Bits 11 STOP3\_WU\_EN\_G2:**Portx Pin Group2 Stop3 唤醒Enable控制.

* 0:Portx Pin[11:8] 唤醒Disable
* 1:Portx Pin[11:8] 唤醒Enable**Bits 10 STOP3\_WU\_LVL\_G2:**Portx Pin Group2 Stop3 唤醒电平选择.
* 0:Portx Pin[11:8] 低电平唤醒
* 1:Portx Pin[11:8] 高电平唤醒**Bits 9-8 STOP3\_WU\_SEL\_G2:**Portx Pin Group2 Stop3 Wakeup Source选择.
* 00:选择 Portx Pin8
* 01:选择 Portx Pin9
* 10:选择 Portx Pin10
* 11:选择 Portx Pin11

**Bits 7 STOP3\_WU\_EN\_G1:**Portx Pin Group1 Stop3 唤醒Enable控制.

* 0:Portx Pin[7:4] 唤醒Disable
* 1:Portx Pin[7:4] 唤醒Enable**Bits 6 STOP3\_WU\_LVL\_G1:**Portx Pin Group1 Stop3 唤醒电平选择.
* 0:Portx Pin[7:4] 低电平唤醒
* 1:Portx Pin[7:4] 高电平唤醒**Bits 5-4 STOP3\_WU\_SEL\_G1:**Portx Pin Group1 Stop3 Wakeup Source选择.
* 00:选择 Portx Pin4
* 01:选择 Portx Pin5
* 10:选择 Portx Pin6
* 11:选择 Portx Pin7

**Bits 3 STOP3\_WU\_EN\_G0:**Portx Pin Group0 Stop3 唤醒Enable控制.

* 0:Portx Pin[3:0] 唤醒Disable
* 1:Portx Pin[3:0] 唤醒Enable**Bits 2 STOP3\_WU\_LVL\_G0:**Portx Pin Group0 Stop3 唤醒电平选择.
* 0:Portx Pin[3:0] 低电平唤醒
* 1:Portx Pin[3:0] 高电平唤醒**Bits 1-0 STOP3\_WU\_SEL\_G0:**Portx Pin Group0 Stop3 Wakeup Source选择.
* 00:选择 Portx Pin0
* 01:选择 Portx Pin1
* 10:选择 Portx Pin2
* 11:选择 Portx Pin3

###### 11.14.20 GPIOD\_STOP3\_WU\_CR

Offset:0x40 Reset Value:0x00000000

|  |  |  |  |
| --- | --- | --- | --- |
| **31-8** | | **7** | **6** |
| RESERVED | | STOP3\_WU\_EN\_G1 | STOP3\_WU\_LVL\_G1 |
| r-0h | | rw-0h | rw-0h |
| **5-4** | **3** | **2** | **1-0** |
| STOP3\_WU\_SEL\_G1 | STOP3\_WU\_EN\_G0 | STOP3\_WU\_LVL\_G0 | STOP3\_WU\_SEL\_G0 |
| rw-0h | rw-0h | rw-0h | rw-0h |

**Bits 31-8 RESERVED:**保留,不可更改.

**Bits 7 STOP3\_WU\_EN\_G1:**PortD Pin Group1 Stop3 唤醒Enable控制.

* 0:PortD Pin[7:4] 唤醒Disable
* 1:PortD Pin[7:4] 唤醒Enable**Bits 6 STOP3\_WU\_LVL\_G1:**PortD Pin Group1 Stop3 唤醒电平选择.
* 0:PortD Pin[7:4] 低电平唤醒
* 1:PortD Pin[7:4] 高电平唤醒**Bits 5-4 STOP3\_WU\_SEL\_G1:**PortD Pin Group1 Stop3 Wakeup Source选择.
* 00:选择 PortD Pin4
* 01:选择 PortD Pin5
* 10:选择 PortD Pin6
* 11:选择 PortD Pin7

**Bits 3 STOP3\_WU\_EN\_G0:**PortD Pin Group0 Stop3 唤醒Enable控制.

* 0:PortD Pin[3:0] 唤醒Disable
* 1:PortD Pin[3:0] 唤醒Enable**Bits 2 STOP3\_WU\_LVL\_G0:**PortD Pin Group0 Stop3 唤醒电平选择.
* 0:PortD Pin[3:0] 低电平唤醒
* 1:PortD Pin[3:0] 高电平唤醒**Bits 1-0 STOP3\_WU\_SEL\_G0:**PortD Pin Group0 Stop3 Wakeup Source选择.
* 00:选择 PortD Pin0
* 01:选择 PortD Pin1
* 10:选择 PortD Pin2
* 11:选择 PortD Pin3

## 12. LoRa 控制器 (LoRaC)

**12.1 Introduction**

LoRa Controller 主要用来控制内部的 RF TRX 实现 LoRa 的发送和接收.

### 12.2 Main features

* 支持 SPI 接口连接 RF TRX
* Interrupt signal generation

### 12.3 功能Description

#### 12.3.1 内部 SPI 接口

LoRa Controller 拥有一路内部 SPI 接口,可以通过Register直接控制 RF TRX.与 RF TRX 之间的通信如下:

1. 初始化 LoRa Controller 内部 SSP.
2. 检查 LORAC\_SR Register的 BUSY\_DIG\_SR 是否为 0,若为 0 则说明当前 RF TRX 空闲,可以进行通信.
3. 将Register LORAC\_NSS\_CR 写 0.
4. 将数据写入 LoRa Controller 内部 SSP 的Register SSP\_DR.
5. 等待发送完成.
6. 通过Register SSP\_DR 读回数据.
7. 根据需求重复执行步骤(4) - (6).
8. 将Register LORAC\_NSS\_CR 写 1.

|  |  |
| --- | --- |
| **12.3.2** | **上电初始化** |



##### Figure 12-1 上电初始化时序

如上Figure所示,上电初始化的流程为:

1. 将Register LORAC\_CR1 的 NRESET\_BAT 置 1.
2. 将Register LORAC\_CR1 的 POR\_BAT 置 0.
3. 等待 BUSY\_DIG\_SR 为 0.其中,Tpor\_min 为 100us,Tnrst\_min 为 50us.

#### 12.3.3 interrupt信号

LoRa Controller 的interrupt信号主要是透传 RF TRX 的interruptRequest ,需注意的是当 LoRa Controller 的interruptRequest 触发时,需要向 RF TRX 发送 ClearIrqStatus 命令清除interrupt,不然interruptRequest 会一直触发.

#### 12.4 LoRaC 相关RegisterDescription

LORAC Base Address:0x40009000

**Table 12-1 LORAC Register Summary**

|  |  |  |
| --- | --- | --- |
| Register | Offset | Description |
| SSP\_CR0 | 0x00 | LORAC 内部 SSP Control Register 0 |
| SSP\_CR1 | 0x04 | LORAC 内部 SSP Control Register 1 |
| SSP\_DR | 0x08 | LORAC 内部 SSP 数据Register |
| SSP\_SR | 0x0C | LORAC 内部 SSP 状态Register |
| SSP\_CPSR | 0x10 | LORAC 内部 SSP clock 分频Register |
| SSP\_IMSC | 0x14 | LORAC 内部 SSP interrupt设置Register |
| SSP\_RIS | 0x18 | LORAC 内部 SSP 原始interrupt状态Register |
| SSP\_MIS | 0x1C | LORAC 内部 SSP 屏蔽interrupt状态Register |
| SSP\_ICR | 0x20 | LORAC 内部 SSP interrupt清除Register |
| SSP\_DMACR | 0x24 | LORAC 内部 SSP DMA Control Register |
| RESERVED | 0x28-0xFC | 保留 |
| LORAC\_CR0 | 0x100 | LORAC Control Register 0 |
| LORAC\_CR1 | 0x104 | LORAC Control Register 1 |
| LORAC\_SR | 0x108 | LORAC 状态Register |
| LORAC\_NSS\_CR | 0x10C | LORAC NSS Control Register |
| LORAC\_SCK\_CR | 0x110 | LORAC SCK Control Register |
| LORAC\_MOSI\_CR | 0x114 | LORAC MOSI Control Register |
| LORAC\_MISO\_SR | 0x118 | LORAC MISO 状态Register |

##### 12.4.1 SSP\_CR0

Offset:0x00

Reset Value:0x00000000

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **31-16** | **15-8** | **7** | **6** | **5-4** | **3-0** |
| RESERVED | SCR | SPH | SPO | FRF | DSS |
| r | r/w | r/w | r/w | r/w | r/w |

**Bits 31-16 RESERVED:**Must be kept, and can't be modified.**Bits 15-8 SCR:**串行clock 速率,用于设置 SSP 传输的数据速率.



SSP 的数据速率计算公式如上,其中 CPSDVR 是取值 2 到 254 的偶数.**Bits 7 SPH:**SSP 相Bits设置,仅应用于 Motorola SPI 格式.**Bits 6 SPO:**SSP 极性设置,仅应用于 Motorola SPI 格式.

**Bits 5-4 FRF:**SSP 帧格式设置.

* 0:Motorola SPI 格式
* 1:Texas Instruments SPI 格式
* 2:National Semiconductor Microwire 格式
* 3:保留**Bits 3-0 DSS:**数据Bits宽设置.
* 0:保留
* 1:保留
* 2:保留
* 3:4 bit  4:5 bit  5:6 bit  6:7 bit  7:8 bit  8:9 bit
* 9:10 bit
* 10:11 bit  11:12 bit
* 12:13 bit  13:14 bit  14:15 bit
* 15:16 bit

##### 12.4.2 SSP\_CR1

Offset:0x04

Reset Value:0x00000000

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **31-4** | **3** | **2** | **1** | **0** |
| RESERVED | SOD | MS | SSE | LBM |
| r | r/w | r/w | r/w | r/w |

**Bits 31-4 RESERVED:**Must be kept, and can't be modified.**Bits 3 SOD:**从模式输出禁止.

* 0:从模式下,SSP 可以输出
* 1:从模式下,SSP 不可输出**Bits 2 MS:**主从模式选择.
* 0:主模式
* 1:从模式**Bits 1 SSE:**SSP Enable.
* 0:Disable
* 1:Enable**Bits 0 LBM:**回环模式.
* 0:正常模式
* 1:回环模式

##### 12.4.3 SSP\_DR

Offset:0x08 Reset Value:0x00000000

|  |  |
| --- | --- |
| **31-16** | **15-0** |
| RESERVED | DATA |
| r | r/w |

**Bits 31-16 RESERVED:**Must be kept, and can't be modified.**Bits 15-0 DATA:**SSP TX/RX 数据.

##### 12.4.4 SSP\_SR

Offset:0x0C

Reset Value:0x00000003

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **31-5** | **4** | **3** | **2** | **1** | **0** |
| RESERVED | BSY | RFF | RNE | TNF | TFE |
| r | r | r | r | r | r |

**Bits 31-5 RESERVED:** Must be kept, and can't be modified.

**Bits 4 BSY:**SSP 忙标识.

* 0:SSP 空闲
* 1:SSP 正在传输中**Bits 3 RFF:**RX FIFO 满标识.
* 0:RX FIFO 未满
* 1:RX FIFO 满**Bits 2 RNE:**RX FIFO 非空标识.
* 0:RX FIFO 为空
* 1:RX FIFO 不为空**Bits 1 TNF:**TX FIFO 非满标识.
* 0:TX FIFO 满
* 1:TX FIFO 未满**Bits 0 TFE:**TX FIFO 空标识.
* 0:TX FIFO 不为空
* 1:TX FIFO 为空

##### 12.4.5 SSP\_CPSR

Offset:0x0C

Reset Value:0x00000000

|  |  |
| --- | --- |
| **31-8** | **7-0** |
| RESERVED | CPSDVSR |
| r | r/w |

**Bits 31-8 RESERVED:** Must be kept, and can't be modified.

**Bits 7-0 CPSDVSR:**clock 分频因子,必须为 2-254 之间的偶数.

###### 12.4.6 SSP\_IMSC

Offset:0x00

Reset Value:0x00000000

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **31-4** | **3** | **2** | **1** | **0** |
| RESERVED | TXIM | RXIM | RTIM | RORIM |
| r | r/w | r/w | r/w | r/w |

**Bits 31-4 RESERVED:**Must be kept, and can't be modified.**Bits 3 TXIM:**TX interrupt屏蔽Bits.

* 0:not allowed产生 TX interrupt
* 1:allowed产生 TX interrupt**Bits 2 RXIM:**RX interrupt屏蔽Bits.
* 0:not allowed产生 RX interrupt
* 1:allowed产生 RX interrupt**Bits 1 RTIM:**RX TIMEOUT interrupt屏蔽Bits.
* 0:not allowed产生 RX TIMEOUT interrupt
* 1:allowed产生 RX TIMEOUT interrupt**Bits 0 RORIM:**RX OVERRUN interrupt屏蔽Bits.
* 0:not allowed产生 RX OVERRUN interrupt
* 1:allowed产生 RX OVERRUN interrupt

##### 12.4.7 SSP\_RIS

Offset:0x00 Reset Value:0x00000008

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **31-4** | **3** | **2** | **1** | **0** |
| RESERVED | TXRIS | RXRIS | RTRIS | RORRIS |
| r | r | r | r | r |

**Bits 31-4 RESERVED:**Must be kept, and can't be modified.**Bits 3 TXRIS:**TX 原始interrupt状态.**Bits 2 RXRIS:**RX 原始interrupt状态.**Bits 1 RTRIS:**RX TIMEOUT 原始interrupt状态.**Bits 0 RORRIS:**RX OVERRUN 原始interrupt状态.

##### 12.4.8 SSP\_MIS

Offset:0x00

Reset Value:0x00000000

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **31-4** | **3** | **2** | **1** | **0** |
| RESERVED | TXMIS | RXMIS | RTMIS | RORMIS |
| r | r | r | r | r |

**Bits 31-4 RESERVED:**Must be kept, and can't be modified.**Bits 3 TXMIS:**TX 屏蔽interrupt状态.**Bits 2 RXMIS:**RX 屏蔽interrupt状态.**Bits 1 RTMIS:**RX TIMEOUT 屏蔽interrupt状态.**Bits 0 RORMIS:**RX OVERRUN 屏蔽interrupt状态.

##### 12.4.9 SSP\_ICR

Offset:0x00 Reset Value:0x00000000

|  |  |  |
| --- | --- | --- |
| **31-2** | **1** | **0** |
| RESERVED | RTIC | RORIC |
| r | w | w |

**Bits 31-2 RESERVED:** Must be kept, and can't be modified.

**Bits 1 RTIC:**RX TIMEOUT interrupt清除,写 1 清除,写 0 无效.

**Bits 0 RORIC:**RX OVERRUN interrupt清除,写 1 清除,写 0 无效.

##### 12.4.10 SSP\_DMACR

Offset:0x00 Reset Value:0x00000000

|  |  |  |
| --- | --- | --- |
| **31-2** | **1** | **0** |
| RESERVED | TXDMAE | RXDMAE |
| r | r/w | r/w |

**Bits 31-2 RESERVED:**Must be kept, and can't be modified.**Bits 1 TXDMAE:**DMA TX Enable.

* 0:关闭 DMA TX
* 1:Enable DMA TX **Bits 0 RXDMAE:**DMA RX Enable.
* 0:关闭 DMA RX
* 1:Enable DMA RX

##### 12.4.11 LORAC\_CR0

Offset:0x100 Reset Value:0x00000000

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **31-11** | **10** | **9** | **8** | **7-5** | **4-0** |
| RESERVED | NSS\_SEL | SCK\_MOSI\_SEL | RESERVED | IRQ\_DIG\_INT\_EN | RESERVED |
| r | r/w | r/w | r | r/w | r |

**Bits 31-11 RESERVED:**Must be kept, and can't be modified.**Bits 10 NSS\_SEL:**RF TRX 的 nss 来源选择.

* 0:来自Register LORAC\_NSS\_CR
* 1:来自 LORAC 的内部 SSP **Bits 9 SCK\_MOSI\_SEL:**RF TRX 的 sck/mosi/miso 来源选择.
* 0:来自 LORAC\_SCK\_CR,LORAC\_MOSI\_CR,LORA\_MISO\_SR
* 1:来自 LORAC 的内部 SSP **Bits 8 RESERVED:** Must be kept, and can't be modified.

**Bits 7-5 IRQ\_DIG\_INT\_EN:**IRQ\_DIG\_INT 高电平interruptEnable, [5] Correspond to IRQ\_DIG[0],[6] Correspond to IRQ\_DIG[1],[7] Correspond to IRQ\_DIG[2].

* 0:Disable
* 1:Enable**Bits 4-0 RESERVED:** Must be kept, and can't be modified.

##### 12.4.12 LORAC\_CR1

Offset:0x104 Reset Value:0x00000080

|  |  |  |  |
| --- | --- | --- | --- |
| **31-8** | **7** | **6** | **5** |
| RESERVED | POR\_BAT | RESERVED | NRESET\_BAT |
| r | r/w | r | r/w |
| **4-3** | **2** | **1** | **0** |
| RESERVED | CLK\_32M\_EN\_BAT | TCXO\_EN\_BAT | PWRTCXO\_EN\_BAT |
| r | r/w | r/w | r/w |

**Bits 31-8 RESERVED:**Must be kept, and can't be modified.**Bits 7 POR\_BAT:**RF TRX 的 POR\_BAT 控制.

* 0:不复Bits
* 1:复Bits

**Bits 6 RESERVED:** Must be kept, and can't be modified.

**Bits 5 NRESET\_BAT:**RF TRX 的 NRESET\_BAT 控制.

* 0:复Bits
* 1:不复Bits**Bits 4-3 RESERVED:** Must be kept, and can't be modified.

**Bits 2 CLK\_32M\_EN\_BAT:**RF TRX 的 CLK\_32M\_EN\_BAT 控制.

* 0:Disable
* 1:Enable

**Bits 1 TCXO\_EN\_BAT:**RF TRX 的 TCXO\_EN\_BAT 控制.

* 0:Disable
* 1:Enable

**Bits 0 PWRTCXO\_EN\_BAT:**RF TRX 的 PWRTCXO\_EN\_BAT 控制.

* 0:Disable
* 1:Enable

##### 12.4.13 LORAC\_SR

Offset:0x108 Reset Value:0x00000100

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **31-9** | **8** | **7-5** | **4-2** | **1** | **0** |
| RESERVED | BUSY\_DIG\_SR | IRQ\_DIG\_SR | RESERVED | CLK\_32M\_RDY\_BAT  \_SR | RESERVED |
| r | r | r | r | r | r |

**Bits 31-9 RESERVED:** Must be kept, and can't be modified.

**Bits 8 BUSY\_DIG\_SR:**BUSY\_DIG 状态Bits,用于指示 RF TRX 是否处于命令处理状态.硬件置 1 与清 0.

* 0:未进行命令处理
* 1:正在进行命令处理**Bits 7-5 IRQ\_DIG\_SR:**IRQ\_DIG 状态Bits,用于指示 RF TRX 的interruptRequest .硬件置 1 与清 0.需注意的是,需要向 RF TRX 发送 ClearIrqStatus 命令清除interrupt,不然interruptRequest 会一直触发.
* 0:无interrupt
* 1:有interrupt**Bits 4-2 RESERVED:** Must be kept, and can't be modified.

**Bits 1 CLK\_32M\_RDY\_BAT\_SR:**CLK\_32M\_RDY\_BAT 状态Bits,用于指示 RF TRX 的 XO32M clock 是否建立.硬件置 1 与清 0.

* 0:未建立
* 1:已建立**Bits 0 RESERVED:** Must be kept, and can't be modified.

##### 12.4.14 LORAC\_NSS\_CR

Offset:0x10C Reset Value:0x00000001

|  |  |
| --- | --- |
| **31-1** | **0** |
| RESERVED | REG\_NSS |
| r | r/w |

**Bits 31-1 RESERVED:**Must be kept, and can't be modified.**Bits 0 REG\_NSS:**nss 的Register控制Bits.

* 0:nss 拉低
* 1:nss 拉高

##### 12.4.15 LORAC\_SCK\_CR

Offset:0x110 Reset Value:0x00000000

|  |  |
| --- | --- |
| **31-1** | **0** |
| RESERVED | REG\_SCK |
| r | r/w |

**Bits 31-1 RESERVED:**Must be kept, and can't be modified.**Bits 0 REG\_SCK:**sck 的Register控制Bits.

* 0:sck 拉低
* 1:sck 拉高

##### 12.4.16 LORAC\_MOSI\_CR

Offset:0x114 Reset Value:0x00000000

|  |  |
| --- | --- |
| **31-1** | **0** |
| RESERVED | REG\_MOSI |
| r | r/w |

**Bits 31-1 RESERVED:**Must be kept, and can't be modified.**Bits 0 REG\_MOSI:**mosi 的Register控制Bits.

* 0:mosi 拉低
* 1:mosi 拉高

##### 12.4.17 LORAC\_MISO\_SR

Offset:0x118 Reset Value:0x00000000

|  |  |
| --- | --- |
| **31-1** | **0** |
| RESERVED | REG\_MISO |
| r | r |

**Bits 31-1 RESERVED:** Must be kept, and can't be modified.

**Bits 0 REG\_MISO:**miso 状态Bits,指示 RF TRX 输出的 miso 状态.硬件置 1 与清 0.

* 0:低电平
* 1:高电平

### 13. 通用异步收发器 (UART)

#### 13.1 Introduction

支持 UART 和 IrDA 模式.

发送与接收的 FIFO 独立,深度 16,FIFO 水Bits可配置为 1/8、1/4、1/2、3/4 和 7/8,禁用 FIFO 相

当于 1 字符深度.16 Bits波特率除数整数部和 6 Bits波特率除数小数部.标准异步通信Bits,支持 5、 6、7和8Bits数据,支持奇偶校验,支持1个或者2个停止Bits.支持DMA,支持假开始Bits检测,支持 Line Break 产生与检测,支持硬件流控.

IrDA模式码率最高460800,Low-Power IrDA模式最高码率115200,半双工.支持 3/16和Low-

Power (1.41~2.23µs) Bits长度.Low-Power IrDA 模式,通过 UARTCLK 分频近似产生Bits长度.

可通过 ID Register唯一地识别每个 UART 端口.

**13.2 clock 复Bits**

每个 UART 都有独立的 APB 总线clock 和独立的 APB 总线复Bits.

#### 13.3 参考clock

UARTCLK 的频率必须要满足波特率产生的要求:

***FUARTCLK(min) >= 16 x baudrate(max) F UARTCLK(max) <= 16 x 65535 x baudrate(min)***

例如,要产生介于 110 到 460800 之间的波特率,UARTCLK 的频率必须要介于 7.3728MHz 与

115.34MHz 之间.

同时,UARTCLK 不能大于 5/3 倍 PCLK:***FUARTCLK <= 5/3 \* FPCLK***

例如,UART 模式下,UARTCLK 为 14.7456MHz,要产生 921600 的波特率,那么 PCLK 必须大于等于 8.85276MHz.这保证了 UART 有足够的时间将接收的数据写入接收 FIFO.

#### 13.4 波特率发生器

波特率发生器包含产生内部 16 倍clock 的自由运行counter,Baud16 和 IrLPBaud16.Baud16 为 UART 发送和接收控制提供时序信息,是一个宽度为 1 个 UARTCLK 周期的脉冲流,频率为 16 倍波特率.当运行在 Low-Power IrDA 模式时,IrLPBaud16 为 IrDA 编码发送比特流的脉冲宽度的产生提供时序信息.

##### 13.5 FIFO

发送与接收的 FIFO 独立,通过线控Register UARTx\_LCR\_H{FEN} 选择开启或者关闭.发送 16 x 8,接收 16 x 12,接收 FIFO 每个字符有 4 个Bits的状态码,FIFO 水Bits可通过 FIFO interrupt水Bits选择RegisterUARTx\_IFLS配置为 1/8、1/4、1/2、3/4和 7/8,当FIFO禁用时相当于深度1.FIFO 状态通过查询标志Register UARTx\_FR 获取.接收FIFO的[10:8]Bits为错误Bits,指示相应的错误.第11Bits为OverrunBits,指示Overrun错误.

###### Table 13-1 接收 FIFO Bits功能Description

|  |  |
| --- | --- |
| FIFO Bit | 功能 |
| 11 | Overrun 错误 |
| 10 | Break 错误 |
| 9 | 奇偶校验错误 |
| 8 | 帧错误 |
| 7:0 | 被接收数据 |

##### 13.6 UART 方式

###### 13.6.1 波特率除数

波特率除数由 16 Bits的整数部和 6 Bits的小数部组成,整数部存入Register UARTx\_IBRD,小数部存入Register UARTx\_FBRD,这allowed使用任何 > 3.6864MHz 的clock 作为 UARTCLK 以支持产生所有标准波特率,满足以下公式:

*Baud Rate Divisor = UARTCLK / (16 x BautRate) = BRDI + BRDF*

其中 BRDI 为整数部,BRDF 为小数部,以小数点隔开,如下所示:

|  |  |  |
| --- | --- | --- |
| 16-bit Integer Part | **.** | 6-bit Fractional Part |

6 Bits小数部的计算方法是,将波特率除数的小数部分乘以 64（ =2n, n 为 UARTx\_FBRD Register的有效Bits宽度 6 ）,考虑到舍入误差,再加上 0.5,公式如下:

***Fractional Part = BRDF x 2n + 0.5***

###### 13.6.2 数据发送

发送与接收的数据都保存在 16 字节的 FIFO 里面,接收 FIFO 每个字符另有 4 个状态标志Bits.

发送时,数据通过数据Register UARTx\_DR 写入发送 FIFO.通过 UARTx\_CR{UARTEN} Enable UART,通过线控Register UARTx\_LCR\_H 的配置数据Bits、停止Bits、奇偶校验等参数,数据开始发送,直到 FIFO 为空.一旦数据写入发送 FIFO,BUSY 信号变高,并且当数据在发送的时候一直保持为高.只有当发送 FIFO 为空,并且Bits移Register的最后一个字符的 stop Bits发送完,

BUSY 信号才会变低.当 UART 没有Enable的时候,BUSY 信号也为高.

###### 13.6.3 数据接收

通过 UARTx\_CR{UARTEN}Enable UART,通过线控Register UARTx\_LCR\_H 的配置数据Bits、停止Bits、奇偶校验等参数.

当接收为空闲,UARTRXD 拉低,Baud16 Enable接收counter开始计数,UART 模式在第 8 个计数周期开始采样.IrDA 模式在第 4 个计数周期开始采样,以allowed更短的逻辑 0 脉冲.

若 UARTRXD 在第 8 个计数周期仍保持为低,那么有效的 start Bits被检测到,否则判定为假 start,并且被忽略.

若startBits有效,接着每16个Baud16周期进行一次数据采样,长度由UARTx\_LCR\_H{WLEN} 决定.如果Enable了奇偶校验,会进行奇偶校验Bits的比对.

最后,当 UARTRXD 变高,有效的 stop Bits被确认到,否则发生帧错误.完整接收的字符与错误Bits一起被存入接收 FIFO.

##### 13.7 IrDA SIR 方式

IrDA SIR ENDEC 提供了在 UART 数据流和半双工串行 SIR 接口之间转换的功能,将数据从

UART 编码输出和解码输入到 UART,有两种模式:

* **IrDA模式**,逻辑0电平被转换为高电平脉冲,宽度为nSIROUT波特率比特周期的 3/16,逻辑 1 电平被转换为低电平.
* **Low-Power IrDA模式**,发送的高电平脉冲宽度为内部IrLPBaud16周期的3倍（1.63us,假定名义频率为 1.842MHz）.

IrDA SIR 物理层为半双工的通信链接,发送与接收之间切换至少要保持 10ms 的延时.这个延

时必须由软件完成,因为 UART 不支持自动延时.这是必须的,因为红外接收端可能会有偏差.

###### 13.7.1 低功耗除数

IrLPBAUD16 由 UARTCLK 分频产生,分频系数通过 UARTx\_ILPR{ ILPDVSR } 配置.

***Low-Power Divider = (FUARTCLK / FIrLPBAUD16)***

FIrLPBAUD16 名义上为 1.8432MHz,满足 **1.42MHz < FIrLPBAUD16 < 2.12MHz** 的要求.

###### 13.7.2 IrDA SIR 发送编码器

SIR 发送编码器将 UART 输出的 NRZ（Non Return-to-Zero）发送比特流进行调制.IrDA 物理层使用 RZI（Return-to-Zero Inverted）调制模式,把逻辑 0 转换为一个红外光脉冲.调制的输出脉冲流被传送到外部的输出驱动和红外发光二极管.

在 IrDA 模式,发送的脉冲宽度为 Baud16 周期的 3 倍,即 3/16 比特周期.

在Low-Power IrDA模式,被传输的脉冲宽度为115200波特率比特周期的 3/16.即为1.842MHz clock IrLPBaud16 周期的三倍.

如果使用了小数波特率除数,发送的 SIR 脉冲流会包含更多的抖动,这是因为使用了小数波特率除数后,不能产生一个规律间隔的Baud16脉冲,这些Baud16周期包含了不同的UARTCLK 周期数.最坏的情况下,导致 SIR 脉冲流的抖动可以达到 3 个 UARTCLK 周期.只要 UARTCLK > 3.6864MHz,且 IrDA 的波特率小于 115200,抖动 < 9%,这满足 SIR IrDA 抖动小于 13%的要求.

###### 13.7.3 IrDA SIR 接收解码器

SIR 接收解码器从红外接收器解调 Return-to-Zero 比特流,输出接收到的 NRZ 串行比特流到 UART 接收输入.正常情况下,在空闲状态接收的解码器的输入为高,发送的编码器的输出与解码器输入的极性相反.

当接收的解码器输入为低则 start Bits被检测到.

为防止 UART 响应接收数据输入的毛刺,在 IrDA 模式,SIRIN 上小于 Buad16 的 3/16 的脉冲会被忽略；在 Low-Power IrDA 模式,SIRIN 上小于 IrLPBuad16 的 3/16 的脉冲会被忽略.

##### 13.8 UART 字符帧结构

UART

字符帧

结构如下Figure所示

：

**Figure**

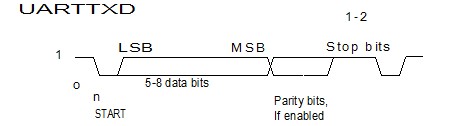
**13**

**-**

**1**

**UART**

字符帧



##### 13.9 IrDA 数据调制

下Figure展示

了

I

rDA

3

/

16

数据

调制的效果

：

**Figure**

**13**

**-**

**2**

**IrDA**

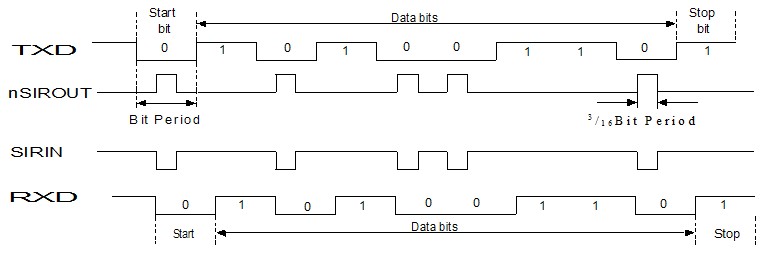
数据调制（

**3**

**/**

**16**

）



##### 13.10 硬件流控

硬件流控可以通过 UARTx\_CR{CTSEn}和 UARTx\_CR{RTSEn}配置.当 RTS 功能被Enable,且接收 FIFO 水Bits未满,则 nUARTRTS 信号一直有效.

当 CTS 功能被Enable,只有 nUARTCTS 信号有效,且发送 FIFO 不为空才会进行传输.

##### 13.11 interrupt

支持 Tx Done、Rx Done、Rx Timeout、Frame Error、Break Error、Parity Error 和 Overrun

Error interrupt,可以通过interrupt掩码设置清除Register UARTx\_IMSC 配置.所有interrupt信号可以通过原始interrupt状态Register UARTx\_RIS 查询到,包括未Enable的interruptBits.通过掩码interruptRegister UARTx\_MIS 查询当前已Enable的interrupt,通过interrupt清除Register UARTx\_ICR 写 1 清除相应interrupt.

**13.12 DMA**

支持 DMA 发送与接收,通过 UARTx\_DMACR 开启或者关闭.

##### 13.13 UART 相关RegisterDescription

UART0 基地址:0x40003000

UART1 基地址:0x40004000

UART2 基地址:0x40010000

UART3 基地址:0x40011000

**Table 13-2 UART Register Summary**

|  |  |  |
| --- | --- | --- |
| Register | Offset | Description |
| UARTx\_DR | 0x00 | 数据Register |
| UARTx\_RSR\_ECR | 0x04 | 接收状态Register/错误清除Register |
| UARTx\_RSV0[4] | 0x08 | 4 x 4 字节保留 |
| UARTx\_FR | 0x18 | 标志Register |
| UARTx\_RSV1 | 0x1C | 4 字节保留 |
| UARTx\_ILPR | 0x20 | 红外低功耗计数Register |
| UARTx\_IBRD | 0x24 | 波特率整数Register |
| UARTx\_FBRD | 0x28 | 波特率小数Register |
| UARTx\_LCR\_H | 0x2C | 线控Register |
| UARTx\_CR | 0x30 | Control Register |
| UARTx\_IFLS | 0x34 | interrupt FIFO 水Bits选择Register |
| UARTx\_IMSC | 0x38 | interrupt掩码设置/清除Register |
| UARTx\_RIS | 0x3C | 原始interrupt状态Register |
| UARTx\_MIS | 0x40 | 被掩interrupt状态Register |
| UARTx\_ICR | 0x44 | interrupt清除Register |
| UARTx\_DMACR | 0x48 | DMA Control Register |
| UARTx\_RSV2[997] | 0x4C | 4 x 997 字节保留 |

###### 13.13.1 UARTx\_DR (x=0, 1, 2, 3)

Offset:0x00

Reset Value:0x00000000

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **31-12** | **11** | **10** | **9** | **8** | **7-0** |
| RESERVED | OE | BE | PE | FE | DATA |
| r-0h | r-0h | r-0h | r-0h | r-0h | rw-0h |

**Bits 31-12 RESERVED:**保留,不可更改.

**Bits 11 OE:**溢出错误标志.

* 0:无溢出
* 1:发生溢出

**Bits 10 BE:**Break 错误标志.

* 0:未发生 Break 错误
* 1:发生 Break 错误接收数据的输入被拉低超过1个整字（=开始Bits+数据+奇偶校验Bits+停止Bits）的传输时间长度为Break错误.

FIFO 模式,该错误与 FIFO 顶部的字符相关.当 Break 错误产生时,只有一个 0 字符会被写入 FIFO.

**Bits 9 PE:**奇偶校验错误标志.

* 0:未发生奇偶校验错误
* 1:发生奇偶校验错误接收字符的奇偶校验Bits与 UARTx\_LCR\_H {EPS} 不匹配则产生奇偶校验错误.

FIFO 模式,该错误与 FIFO 顶部的字符相关.

**Bits 8 FE:**帧错误标志.

* 0:未发生帧错误
* 1:发生帧错误错误Table示收到的字符停止Bits无效.

FIFO 模式,该错误与 FIFO 顶部的字符相关.**Bits 7-0 DATA:**发送数据字符/接收数据字符.

###### 13.13.2 UARTx\_RSR\_ECR (x=0, 1, 2, 3)

Offset:0x04

Reset Value:0x00000000

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **31-4** | **3** | **2** | **1** | **0** |
| RESERVED | OE | BE | PE | FE |
| r-0h | r-0h | r-0h | r-0h | r-0h |

**Bits 31-4 RESERVED:**保留,不可更改.

**Bits 3 OE:**溢出错误标志.

* 0:无溢出
* 1:发生溢出.

**Bits 2 BE:**Break 错误标志.

* 0:未发生 Break 错误
* 1:发生 Break 错误

接收数据的输入被拉低超过 1 个整字（=开始Bits+数据+奇偶校验Bits+停止Bits）的传输时间长度为 Break 错误.

FIFO 模式,该错误与 FIFO 顶部的字符相关.当 Break 错误产生时,只有一个 0 字符会被写入 FIFO.

**Bits 1 PE:**奇偶校验错误标志.

* 0:未发生奇偶校验错误
* 1:发生奇偶校验错误

接收字符的奇偶校验Bits与 UARTx\_LCR\_H {EPS} 不匹配则产生奇偶校验错误.

FIFO 模式,该错误与 FIFO 顶部的字符相关.

**Bits 0 FE:**帧错误标志.

* 0:未发生帧错误帧
* 1:发生帧错误错误Table示收到的字符停止Bits无效.

FIFO 模式,该错误与 FIFO 顶部的字符相关.

13.13.3 UARTx\_FR (x=0, 1, 2, 3)

Offset:0x18 Reset Value:0x00000000

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| **31-8** | **7** | **6** | **5** | **4** | **3** | **2-0** |
| RESERVED | TXFE | RXFF | TXFF | RXFE | BUSY | RESERVED |
| r-0h | r-0h | r-0h | r-0h | r-0h | r-0h | r-0h |

**Bits 31-8 RESERVED:**保留,不可更改.**Bits 7 TXFE:**发送 FIFO 空.

* 0:发送 FIFO/数据Register不为空
* 1:发送 FIFO/数据Register为空与 UARTx\_LCR\_H{FEN} Bits相关,该Bits不能用于指示发送Bits移Register中是否有数据.

**Bits 6 RXFF:**接收 FIFO 满.

* 0:接收 FIFO/数据Register未满
* 1:接收 FIFO/数据Register满

与 UARTx\_LCR\_H{FEN} Bits相关.**Bits 5 TXFF:**发送 FIFO 满.

* 0:发送 FIFO/数据Register未满
* 1:发送 FIFO/数据Register满

与 UARTx\_LCR\_H{FEN} Bits相关.

**Bits 4 RXFE:**接收 FIFO 空标志.

* 0:接收 FIFO/数据Register不为空
* 1:接收 FIFO/数据Register为空与 UARTx\_LCR\_H{FEN} Bits相关.

**Bits 3 BUSY:**忙标志.

* 0:无数据发送
* 1:正在发送数据该Bits在发送 FIFO 变为非空状态马上置 1,不论 UART 是否Enable.

**Bits 2-0 RESERVED:**保留,不可更改.

13.13.4 UARTx\_ILPR (x=0, 1, 2, 3)

Offset:0x20 Reset Value:0x00000000

|  |  |
| --- | --- |
| **31-8** | **7-0** |
| RESERVED | ILPDVSR |
| r-0h | rw-0h |

**Bits 31-8 RESERVED:**保留,不可更改.

**Bits 7-0 ILPDVSR:**低功耗除数值,0 为非法值,写入 0 将导致无 IrLPBaud16 产生.

13.13.5 UARTx\_IBRD (x=0, 1, 2, 3)

Offset:0x24 Reset Value:0x00000000

|  |  |
| --- | --- |
| **31-16** | **15-0** |
| RESERVED | BAUD\_DIVINT |
| r-0h | rw-0h |

**Bits 31-16 RESERVED:**保留,不可更改.

**Bits 15-0 BAUD\_DIVINT:**波特率除数整数部.

13.13.6 UARTx\_FBRD (x=0, 1, 2, 3)

Offset:0x28 Reset Value:0x00000000

|  |  |
| --- | --- |
| **31-6** | **5-0** |
| RESERVED | BAUD\_DIVFRAC |
| r-0h | rw-0h |

**Bits 31-6 RESERVED:**保留,不可更改.**Bits 5-0 BAUD\_DIVFRAC:**波特率除数小数部.

13.13.7 UARTx\_LCR\_H (x=0, 1, 2, 3)

Offset:0x2C

Reset Value:0x00000000

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| **31-7** | **6-5** | **4** | **3** | **2** | **1** | **0** |
| RESERVED | WLEN | FEN | STP2 | EPS | PEN | BRK |
| r-0h | rw-0h | rw-0h | rw-0h | rw-0h | rw-0h | rw-0h |

**Bits 31-7 RESERVED:**保留,不可更改.

**Bits 6-5 WLEN:**数据Bits长度.

* 00:5 Bits
* 01:6 Bits
* 10:7 Bits
* 11:8 Bits**Bits 4 FEN:**FIFO Enable.
* 0:禁用 FIFO 模式
* 1:Enable FIFO 模式**Bits 3 STP2:**停止Bits选择.
* 0:1 个停止Bits
* 1:2 个停止Bits**Bits 2 EPS:**偶校验Bits选择.
* 0:奇校验
* 1:偶校验当 PEN 为 0 时该Bits无效.**Bits 1 PEN:**奇偶校验Enable.
* 0:禁用奇偶校验功能
* 1:Enable奇偶校验功能,发送产生奇偶校验Bits,接收检查奇偶校验Bits**Bits 0 BRK:**发送 Break.
* 写 0:结束 Break 命令
* 写 1:当前字符发送完成后,UART\_TXD 引脚一直拉低为确保 Break 命令的执行,软件必须至少保持该Bits被设置超过 2 个完整的帧长度.

13.13.8 UARTx\_CR (x=0, 1, 2, 3)

Offset:0x30 Reset Value:0x00000000

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **31-24** | | **23-16** | **15** | **14** | **13-10** |
| RESERVED | | RESERVED | CTSEn | RTSEn | RESERVED |
| r-0h | | r-0h | rw-0h | rw-0h | r-0h |
| **9** | **8** | **7-3** | **2** | **1** | **0** |
| RXE | TXE | RESERVED | SIRLP | SIREN | UARTEN |
| rw-0h | rw-0h | r-0h | rw-0h | rw-0h | rw-0h |

**Bits 31-16 RESERVED:**保留,不可更改.**Bits 15 CTSEn:**硬件 CTS 流控Enable.

* 0:关闭硬件 CTS 流控
* 1:开启硬件 CTS 流控**Bits 14 RTSEn:**硬件 RTS 流控Enable.
* 0:关闭硬件 RTS 流控
* 1:开启硬件 RTS 流控**Bits 13-10 RESERVED:**保留,不可更改.

**Bits 9 RXE:**接收Enable.

* 写 0:禁止接收,若当前正在接收数据,将在这一帧数据接收完成后停止
* 写 1:Enable接收**Bits 8 TXE:**发送Enable.
* 写 0:禁止发送,若当前有数据在发送,将在这一帧数据发送完成后停止
* 写 1:Enable发送**Bits 7-3 RESERVED:**保留,不可更改.**Bits 2 SIRLP:**Low-Power IrDA Enable.
* 0:低电平Bits以 3/16比特周期的脉冲宽度传输.
* 1:低电平Bits以 3 倍 IrLPBAUD16 周期的脉冲宽度来传输,有利于降低功耗,但是也会缩短传输距离.**Bits 1 SIRE:**IrDA Enable.
* 0:关闭 IrDA SIR ENDEC,SIR\_OUT 保持低,SIR\_IN 被忽略.数据在 UART\_TXD 和 UART\_RXD 上传输.
* 1:开启 IrDA SIR ENDEC,UART\_TXD 保持高,UART\_RXD 被忽略.数据在 SIR\_OUT 和 SIR\_IN 上传输.

若 UARTEN Bits为 0,该Bits无效.**Bits 0 UARTEN:**串口Enable.

* 写 0:关闭 UART 功能,若当前有数据在发送或者接收,将在这一帧数据传输完成后再关闭 UART.  写 1:Enable UART 功能

13.13.9 UARTx\_IFLS (x=0, 1, 2, 3)

Offset:0x34 Reset Value:0x00000000

|  |  |  |
| --- | --- | --- |
| **31-6** | **5-3** | **2-0** |
| RESERVED | RXIFLSEL | TXIFLSEL |
| r-0h | rw-0h | rw-0h |

**Bits 31-6 RESERVED:**保留,不可更改.**Bits 5-3 RXIFLSEL:**接收 FIFO interrupt水Bits选择.

* 000:接收 FIFO 水Bits≥1/8
* 001:接收 FIFO 水Bits≥1/4
* 010:接收 FIFO 水Bits≥1/2
* 011:接收 FIFO 水Bits≥3/4
* 100:接收 FIFO 水Bits≥7/8
* 101~111:保留**Bits 2-0 TXIFLSEL:**发送 FIFO interrupt水Bits选择.
* 000:发送 FIFO 水Bits≥1/8
* 001:发送 FIFO 水Bits≥1/4
* 010:发送 FIFO 水Bits≥1/2
* 011:发送 FIFO 水Bits≥3/4
* 100:发送 FIFO 水Bits≥7/8
* 101~111:保留

13.13.10 UARTx\_IMSC (x=0, 1, 2, 3)

Offset:0x38 Reset Value:0x00000000

|  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| **31-16** | **15-11** | **10** | **9** | **8** | **7** | **6** | **5** | **4** | **3-0** |
| RESERVED | RESERVED | OEIM | BEIM | PEIM | FEIM | RTIM | TXIM | RXIM | RESERVED |
| r-0h | r-0h | rw-0h | rw-0h | rw-0h | rw-0h | rw-0h | rw-0h | rw-0h | r-0h |

**Bits 31-11 RESERVED:**保留,不可更改.**Bits 10 OEIM:**Overrun 错误interrupt掩码.

* 0:禁用 Overrun 错误interrupt
* 1:Enable Overrun 错误interrupt**Bits 9 BEIM:**Break 错误interrupt掩码.
* 0:禁用 Break 错误interrupt
* 1:Enable Break 错误interrupt**Bits 8 PEIM:**校验错误interrupt掩码.
* 0:禁用校验错误interrupt
* 1:Enable校验错误interrupt**Bits 7 FEIM:**帧错误interrupt掩码.
* 0:禁用帧错误interrupt
* 1:Enable帧错误interrupt**Bits 6 RTIM:**接收超时interrupt掩码.
* 0:禁用接收超时interrupt
* 1:Enable接收超时interrupt**Bits 5 TXIM:**发送完成interrupt掩码.
* 0:禁用发送完成interrupt
* 1:Enable发送完成interrupt**Bits 4 RXIM:**接收完成interrupt掩码.
* 0:禁用接收完成interrupt
* 1:Enable接收完成interrupt

**Bits 3-0 RESERVED:**保留,不可更改.

13.13.11 UARTx\_RIS (x=0, 1, 2, 3)

Offset:0x3C

Reset Value:0x00000000

|  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| **31-16** | **15-11** | **10** | **9** | **8** | **7** | **6** | **5** | **4** | **3-0** |
| RESERVED | RESERVED | OERIS | BERIS | PERIS | FERIS | RTRIS | TXRIS | RXRIS | RESERVED |
| r-0h | r-0h | r-0h | r-0h | r-0h | r-0h | r-0h | r-0h | r-0h | r-0h |

**Bits 31-11 RESERVED:**保留,不可更改.

**Bits 10 OERIS:**Overrun 错误原始interrupt状态.

**Bits 9 BERIS:**Break 错误原始interrupt状态.

**Bits 8 PERIS:**校验错误原始interrupt状态.

**Bits 7 FERIS:**帧错误原始interrupt状态.

**Bits 6 RTRIS:**接收超时原始interrupt状态.**Bits 5 TXRIS:**发送完成原始interrupt状态.**Bits 4 RXRIS:**接收完成原始interrupt状态.**Bits 3-0 RESERVED:**保留,不可更改.

13.13.12 UARTx\_MIS (x=0, 1, 2, 3)

Offset:0x40 Reset Value:0x00000000

|  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| **31-16** | **15-11** | **10** | **9** | **8** | **7** | **6** | **5** | **4** | **3-0** |
| RESERVED | RESERVED | OEMIS | BEMIS | PEMIS | FEMIS | RTMIS | TXMIS | RXMIS | RESERVED |
| r-0h | r-0h | r-0h | r-0h | r-0h | r-0h | r-0h | r-0h | r-0h | r-0h |

**Bits 31-11 RESERVED:**保留,不可更改.**Bits 10 OEMIS:**Overrun 错误interrupt状态. **Bits 9 BEMIS:**Break 错误interrupt状态.**Bits 8 PEMIS:**校验错误interrupt状态.**Bits 7 FEMIS:**帧错误interrupt状态.**Bits 6 RTMIS:**接收超时interrupt状态.**Bits 5 TXMIS:**发送完成interrupt状态.**Bits 4 RXMIS:**接收完成interrupt状态.

**Bits 3-0 RESERVED:**保留,不可更改.

13.13.13 UARTx\_ICR (x=0, 1, 2, 3)

Offset:0x44 Reset Value:0x00000000

|  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| **31-16** | **15-11** | **10** | **9** | **8** | **7** | **6** | **5** | **4** | **3-0** |
| RESERVED | RESERVED | OEIC | BEIC | PEIC | FEIC | RTIC | TXIC | RXIC | RESERVED |
| r-0h | r-0h | w-0h | w-0h | w-0h | w-0h | w-0h | w-0h | w-0h | r-0h |

**Bits 31-11 RESERVED:**保留,不可更改.**Bits 10 OEIC:**Overrun 错误interrupt清除.

* 写 0:无效
* 写 1:清除 overrun interrupt**Bits 9 BEIC:**Break 错误interrupt清除.
* 写 0:无效
* 写 1:清除 Break 错误interrupt**Bits 8 PEIC:**校验错误interrupt清除.
* 写 0:无效
* 写 1:清除校验错误interrupt**Bits 7 FEIC:**帧错误interrupt清除.
* 写 0:无效
* 写 1:清除帧错误interrupt

**Bits 6 RTIC:**接收超时interrupt清除.

* 写 0:无效
* 写 1:清除接收超时interrupt**Bits 5 TXIC:**发送完成interrupt清除.
* 写 0:无效
* 写 1:清除发送完成interrupt**Bits 4 RXIC:**接收完成interrupt清除.
* 写 0:无效
* 写 1:清除接收完成interrupt**Bits 3-0 RESERVED:**保留,不可更改.

###### 13.13.14 UARTx\_DMACR (x=0, 1, 2, 3)

Offset:0x48 Reset Value:0x00000000

|  |  |  |  |
| --- | --- | --- | --- |
| **31-3** | **2** | **1** | **0** |
| RESERVED | DMAONERR | TXDMAE | RXDMAE |
| r-0h | rw-0h | rw-0h | rw-0h |

**Bits 31-3 RESERVED:**保留,不可更改.**Bits 2 DMAONERR:**DMA 错误.**Bits 1 TXDMAE:**发送 DMA Enable.

* 0:关闭发送 DMA 功能
* 1:开启发送 DMA 功能**Bits 0 RXDMAE:**接收 DMA Enable.
* 0:关闭接收 DMA 功能
* 1:开启接收 DMA 功能

13.13.15 UARTx\_ID[8] (x=0, 1, 2, 3)

13.13.15.1 PeriphID0

Offset:0x0FE0 Reset Value:0x00000000

|  |  |
| --- | --- |
| **31-8** | **7-0** |
| RESERVED | PARTNUMBER0 |
| r-0h | r-11h |

**Bits 31-8 RESERVED:**保留,不可更改.**Bits 7-0 PARTNUMBER0:**=0x11

13.13.15.2 PeriphID1

Offset:0x0FE4 Reset Value:0x00000000

|  |  |  |
| --- | --- | --- |
| **31-8** | **7-4** | **3-0** |
| RESERVED | DESIGNER0 | PARTNUMBER1 |
| r-0h | r-1h | r-0h |

**Bits 31-8 RESERVED:**保留,不可更改.

**Bits 7-4 DESIGNER0:**=0x1 **Bits 3-0 PARTNUMBER1:**=0x0

13.13.15.3 PeriphID2

Offset:0x0FE8 Reset Value:0x00000000

|  |  |  |
| --- | --- | --- |
| **31-8** | **7-4** | **3-0** |
| RESERVED | REVISION0 | DESIGNER1 |
| r-0h | r-xh | r-0h |

**Bits 31-8 RESERVED:**保留,不可更改**Bits 7-4 REVISION0:**

* 0x0:r1p0
* 0x1:r1p1
* 0x2:r1p3/r1p4
* 0x3:r1p5 **Bits 3-0 DESIGNER1:**=0x0

13.13.15.4 PeriphID3

Offset:0x0FEC Reset Value:0x00000000

|  |  |
| --- | --- |
| **31-8** | **7-0** |
| RESERVED | CONFIGURATION |
| r-0h | r-0h |

**Bits 31-8** **RESERVED:**保留,不可更改.

**Bits 7-0** **CONFIGURATION:**=0x00

13.13.15.5 PCellID0

Offset:0x0FD0 Reset Value:0x00000000

|  |  |
| --- | --- |
| **31-8** | **7-0** |
| RESERVED | CellID0 |
| r-0h | r-dh |

**Bits 31-8** **RESERVED:**保留,不可更改.

**Bits 7-0 CellID0:**=0x0d

13.13.15.6 PCellID1

Offset:0x0FD4 Reset Value:0x00000000

|  |  |
| --- | --- |
| **31-8** | **7-0** |
| RESERVED | CellID1 |
| r-0h | r-f0h |

**Bits 31-8** **RESERVED:**保留,不可更改.

**Bits 7-0** **CellID1:**=0xf0

13.13.15.7 PCellID2

Offset:0x0FD8 Reset Value:0x00000000

|  |  |
| --- | --- |
| **31-8** | **7-0** |
| RESERVED | CellID2 |
| r-0h | r-5h |

**Bits 31-8 RESERVED:**保留,不可更改.

**Bits 7-0** **CellID2:**=0x05

13.13.15.8 PCellID3

Offset:0x0FDC Reset Value:0x00000000

|  |  |
| --- | --- |
| **31-8** | **7-0** |
| RESERVED | CellID3 |
| r-0h | r-b1h |

**Bits 31-8** **RESERVED:**保留,不可更改.

**Bits 7-0** **CellID3:**=0xb1

### 14. 同步串行端口 (SSP)

#### 14.1 Introduction

SSP（Synchronous serial port）是一种同步串行接口,支持 MASTER 和 SLAVE 模式.

SSP 支持多种帧格式,并且可以根据需要配置数据宽度和输出速率.

#### 14.2 Main features

* 支持 MASTER 和 SLAVE 模式的配置
* 最大支持 16MHz 输出
* 支持 16-bit 宽,深度为 8 的 TX/RX FIFO
* 支持多种帧格式
* 支持 4-16 bit 数据宽度
* 支持 DMA Request
* 支持interruptRequest

#### 14.3 功能Description

##### 14.3.1 基础说明

SSP 主要有 4 个 pin:SSP\_NSS,SSP\_CLK,SSP\_TX 和 SSP\_RX.

1. **SSP\_NSS**

SSP 片选信号,低有效.

1. **SSP\_CLK**

SSP clock 信号,对 MASTER 模式来说是clock 输出,对 SLAVE 模式来说是clock 输入.

1. **SSP\_TX**

SSP 发送信号,无论 MASTER 模式还是 SLAVE 模式,均为发送 pin.

###### 4. SSP\_RX

SSP 接收信号,无论 MASTER 模式还是 SLAVE 模式,均为接收 pin.

SSP 与 SPI 设备的连接如下Figure,需注意 SSP\_TX/SSP\_RX 与 SPI\_MOSI/SPI\_MISO 的不同.

SSP Master

SPI Slave

SSP

\_

TX

SPI

\_

MISO

SPI

\_

MOSI

SSP

\_

RX

SSP

\_

CLK

SPI

\_

CLK

SSP

\_

NSS

SPI

\_

NSS

**Figure**

**14**

**-**

**1**

**SSP master**

与

**SPI slave**

之间的连接

SPI Master

SSP Slave

SPI

\_

MOSI

SSP

\_

TX

SSP

\_

RX

SPI

\_

MISO

SSP

\_

CLK

SPI

\_

CLK

SSP

\_

NSS

SPI

\_

NSS

**Figure**

**14**

**-**

**2**

**SPI Master**

与

**SSP Slave**

之间的连接

##### 14.3.2 clock 分频

SSP clock 约束条件:

1. 最大支持的输出clock 为 16MHz
2. MASTER 模式下clock 最大为 PCLK 的 1/2
3. SLAVE 模式下clock 最大为 PCLK 的 1/12

MASTER 模式下clock 输出的公式如下:



###### Figure 14-3 MASTER 模式下clock 输出的公式

SSPCLK 为 SSP 的接口clock ,SSPCLKOUT 为 SSP 的输出clock .以默认 24MHz 为例,如果要输出 1MHz 的clock ,设置 CPSDVR 为 2,设置 SCR 为 11.

##### 14.3.3 数据格式

SSP 支持 3 种帧格式:

* Motorola SPI
* Texas Instruments SPI
* National Semiconductor Microwire

##### 14.3.4 DMA 传输

###### SSP DMA 发送过程:

1. 将Register SSP\_DMACR 中的 TXDMAE Bits配置为Enable；
2. 将Register SSP\_DR 地址配置为 DMA 的目的地址；
3. 将发送数据的内存地址配置为 DMA 的源地址；
4. 配置 DMA 的 SRC\_TR\_WIDTH 和 DES\_TR\_WIDTH 为 0（数据Bits宽为 8bit）；
5. 配置 DMA 的 SRC\_MSIZE 和 DEST\_MSIZE 为 1（burst length 为 4）；
6. 配置 DMA 的数据传输总长度；
7. 配置DMA的handshake类型为Correspond to SSP的TX类型（如SSP0为DMA\_HANDSHAKE\_ SSP\_0\_TX）；
8. 激活 DMA 通道.当 DMA 传输完成后,会将 DMA\_CHENREG Register的 CH\_EN\_x Bits清 0.

###### SSP DMA 接收过程:

1. 将Register SSP\_DMACR 中的 RXDMAE Bits配置为Enable；
2. 将Register SSP\_DR 地址配置为 DMA 的源地址；
3. 将数据接收的内存地址配置为 DMA 的目的地址；
4. 配置 DMA 的 SRC\_TR\_WIDTH 和 DES\_TR\_WIDTH 为 0（数据Bits宽为 8bit）；
5. 配置 DMA 的 SRC\_MSIZE 和 DEST\_MSIZE 为 1（burst length 为 4）；
6. 配置 DMA 的数据传输总长度；
7. 配置DMA的handshake类型为Correspond to SSP的RX类型（如SSP0为DMA\_HANDSHAKE\_ SSP\_0\_RX）；
8. 激活 DMA 通道.

当 DMA 传输完成后,会将 DMA\_CHENREG Register的 CH\_EN\_x Bits清 0.

##### 14.3.5 interrupt信号

SSP 主要有四个interrupt:SSP RX interrupt,SSP TX interrupt,SSP RX OVERRUN interrupt和 SSP RX TIMEOUT.

1. **SSP RX interrupt**当 SSP RX FIFO 中有大于等于 4 个数据时触发.
2. **SSP TX interrupt**当 SSP TX FIFO 中有小于等于 4 个数据时触发.
3. **SSP RX Overrun interrupt**

当 SSP RX FIFO 已满,继续收到数据时触发.

1. **SSP RX Timeout interrupt**

当 SSP RX FIFO 不为空,但是 SSP 在 32bit 传输周期中未继续收到数据时触发.

#### 14.4 SSP 相关RegisterDescription

SSP0 Base Address:0x40006000

SSP1 Base Address:0x40012000

SSP2 Base Address:0x40013000

**Table 14-1 SSP Register Summary**

|  |  |  |
| --- | --- | --- |
| Register | Offset | Description |
| SSP\_CR0 | 0x00 | Control Register 0 |
| SSP\_CR1 | 0x04 | Control Register 1 |
| SSP\_DR | 0x08 | 数据Register |
| SSP\_SR | 0x0C | 状态Register |
| SSP\_CPSR | 0x10 | clock 分频Register |
| SSP\_IMSC | 0x14 | interrupt设置Register |
| SSP\_RIS | 0x18 | 原始interrupt状态Register |
| SSP\_MIS | 0x1C | 屏蔽interrupt状态Register |
| SSP\_ICR | 0x20 | interrupt清除Register |
| SSP\_DMACR | 0x24 | DMA Control Register |

##### 14.4.1 SSP\_CR0

Offset:0x00

Reset Value:0x00000000

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **31-16** | **15-8** | **7** | **6** | **5-4** | **3-0** |
| RESERVED | SCR | SPH | SPO | FRF | DSS |
| r | r/w | r/w | r/w | r/w | r/w |

**Bits 31-16 RESERVED:**Must be kept, and can't be modified.**Bits 15-8 SCR:**串行clock 速率,用于设置 SSP 传输的数据速率.



SSP 的数据速率计算公式如上,其中 CPSDVR 是取值 2 到 254 的偶数.**Bits 7 SPH:**SSP 相Bits设置,仅应用于 Motorola SPI 格式.**Bits 6 SPO:**SSP 极性设置,仅应用于 Motorola SPI 格式.

**Bits 5-4 FRF:**SSP 帧格式设置.

* 0:Motorola SPI 格式
* 1:Texas Instruments SPI 格式
* 2:National Semiconductor Microwire 格式
* 3:保留**Bits 3-0 DSS:**数据Bits宽设置.
* 0:保留
* 1:保留
* 2:保留
* 3:4 bit  4:5 bit  5:6 bit  6:7 bit  7:8 bit  8:9 bit
* 9:10 bit
* 10:11 bit  11:12 bit
* 12:13 bit  13:14 bit  14:15 bit
* 15:16 bit

##### 14.4.2 SSP\_CR1

Offset:0x04

Reset Value:0x00000000

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **31-4** | **3** | **2** | **1** | **0** |
| RESERVED | SOD | MS | SSE | LBM |
| r | r/w | r/w | r/w | r/w |

**Bits 31-4 RESERVED:**Must be kept, and can't be modified.**Bits 3 SOD:**从模式输出禁止.

* 0:从模式下,SSP 可以输出
* 1:从模式下,SSP 不可输出**Bits 2 MS:**主从模式选择.
* 0:主模式
* 1:从模式**Bits 1 SSE:**SSP Enable.
* 0:Disable
* 1:Enable**Bits 0 LBM:**回环模式.
* 0:正常模式
* 1:回环模式

##### 14.4.3 SSP\_DR

Offset:0x08 Reset Value:0x00000000

|  |  |
| --- | --- |
| **31-16** | **15-0** |
| RESERVED | DATA |
| r | r/w |

**Bits 31-16 RESERVED:**Must be kept, and can't be modified.**Bits 15-0 DATA:**SSP TX/RX 数据.

##### 14.4.4 SSP\_SR

Offset:0x0C

Reset Value:0x00000003

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **31-5** | **4** | **3** | **2** | **1** | **0** |
| RESERVED | BSY | RFF | RNE | TNF | TFE |
| r | r | r | r | r | r |

**Bits 31-5 RESERVED:** Must be kept, and can't be modified.

**Bits 4 BSY:**SSP 忙标识.

* 0:SSP 空闲
* 1:SSP 正在传输中**Bits 3 RFF:**RX FIFO 满标识.
* 0:RX FIFO 未满
* 1:RX FIFO 满**Bits 2 RNE:**RX FIFO 非空标识.
* 0:RX FIFO 为空
* 1:RX FIFO 不为空**Bits 1 TNF:**TX FIFO 非满标识.
* 0:TX FIFO 满
* 1:TX FIFO 未满**Bits 0 TFE:**TX FIFO 空标识.
* 0:TX FIFO 不为空
* 1:TX FIFO 为空

##### 14.4.5 SSP\_CPSR

Offset:0x0C

Reset Value:0x00000000

|  |  |
| --- | --- |
| **31-8** | **7-0** |
| RESERVED | CPSDVSR |
| r | r/w |

**Bits 31-8 RESERVED:** Must be kept, and can't be modified.

**Bits 7-0 CPSDVSR:**clock 分频因子,必须为 2-254 之间的偶数.

###### 14.4.6 SSP\_IMSC

Offset:0x00

Reset Value:0x00000000

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **31-4** | **3** | **2** | **1** | **0** |
| RESERVED | TXIM | RXIM | RTIM | RORIM |
| r | r/w | r/w | r/w | r/w |

**Bits 31-4 RESERVED:**Must be kept, and can't be modified.**Bits 3 TXIM:**TX interrupt屏蔽Bits.

* 0:not allowed产生 TX interrupt
* 1:allowed产生 TX interrupt**Bits 2 RXIM:**RX interrupt屏蔽Bits.
* 0:not allowed产生 RX interrupt
* 1:allowed产生 RX interrupt**Bits 1 RTIM:**RX TIMEOUT interrupt屏蔽Bits.
* 0:not allowed产生 RX TIMEOUT interrupt
* 1:allowed产生 RX TIMEOUT interrupt**Bits 0 RORIM:**RX OVERRUN interrupt屏蔽Bits.
* 0:not allowed产生 RX OVERRUN interrupt
* 1:allowed产生 RX OVERRUN interrupt

##### 14.4.7 SSP\_RIS

Offset:0x00 Reset Value:0x00000008

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **31-4** | **3** | **2** | **1** | **0** |
| RESERVED | TXRIS | RXRIS | RTRIS | RORRIS |
| r | r | r | r | r |

**Bits 31-4 RESERVED:**Must be kept, and can't be modified.**Bits 3 TXRIS:**TX 原始interrupt状态.**Bits 2 RXRIS:**RX 原始interrupt状态.**Bits 1 RTRIS:**RX TIMEOUT 原始interrupt状态.**Bits 0 RORRIS:**RX OVERRUN 原始interrupt状态.

##### 14.4.8 SSP\_MIS

Offset:0x00

Reset Value:0x00000000

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **31-4** | **3** | **2** | **1** | **0** |
| RESERVED | TXMIS | RXMIS | RTMIS | RORMIS |
| r | r | r | r | r |

**Bits 31-4 RESERVED:**Must be kept, and can't be modified.**Bits 3 TXMIS:**TX 屏蔽interrupt状态.**Bits 2 RXMIS:**RX 屏蔽interrupt状态.**Bits 1 RTMIS:**RX TIMEOUT 屏蔽interrupt状态.**Bits 0 RORMIS:**RX OVERRUN 屏蔽interrupt状态.

##### 14.4.9 SSP\_ICR

Offset:0x00 Reset Value:0x00000000

|  |  |  |
| --- | --- | --- |
| **31-2** | **1** | **0** |
| RESERVED | RTIC | RORIC |
| r | w | w |

**Bits 31-2 RESERVED:** Must be kept, and can't be modified.

**Bits 1 RTIC:**RX TIMEOUT interrupt清除,写 1 清除,写 0 无效.

**Bits 0 RORIC:**RX OVERRUN interrupt清除,写 1 清除,写 0 无效.

##### 14.4.10 SSP\_DMACR

Offset:0x00 Reset Value:0x00000000

|  |  |  |
| --- | --- | --- |
| **31-2** | **1** | **0** |
| RESERVED | TXDMAE | RXDMAE |
| r | r/w | r/w |

**Bits 31-2 RESERVED:**Must be kept, and can't be modified.**Bits 1 TXDMAE:**DMA TX Enable.

* 0:关闭 DMA TX
* 1:Enable DMA TX **Bits 0 RXDMAE:**DMA RX Enable.
* 0:关闭 DMA RX
* 1:Enable DMA RX

### 15. 内部集成电路 (I2C) 接口

#### 15.1 Introduction

I2C 总线接口单元支持主机模式和从机模式.SDA 为数据传输线,SCL 为参考clock 线.支持多主机和总线仲裁功能.支持 100Kbps 标准速率模式,和 400Kbps 快速模式.支持 FIFO 模式,发送 FIFO 深度 8,接收 FIFO 深度 16,FIFO 的读写指针可配.

 **Figure 15-1 I2C 框Figure**

#### 15.2 Start 与 Stop 条件

Start 条件:当 SCL 为高时,SDA 从高跳变为低则产生 Start 条件.

Stop 条件:当 SCL 为高时,SDA 从低跳变为高则产生 Stop 条件.



**Figure 15-2 Start 与 Stop 条件的 SDA 与 SCL 信号**

通过配置 I2Cx\_CR{START} 和 I2Cx\_CR{STOP} 来开始一个字节的传输,或者产生 Start、 Repeated Start 和 Stop 条件.

**Table 15-1 Start 和 Stop 条件定义**

|  |  |  |  |
| --- | --- | --- | --- |
| StartBits | StopBits | 条件 | Description |
| 0 | 0 | 无 Start 和 Stop | 当有多个数据字节将要被传输的时候,I2C 不会发送  Start 或者 Stop 条件 |
| 0 | 1 | Start 或  Repeated Start | I2C 发送一个 Start 条件然后再发送 I2Cx\_DBR 内的 8 Bits数据.Start 发送前,I2Cx\_DBR 必须要包含 7 Bits的从地址和 1 Bits的 R/nW.  Repeated Start 条件,I2Cx\_DBR 包含目标从设备地址和 R/nW Bits,这allowed主机在不释放总线的情况下进行多次传输.  接口停留在主机发送模式用于写,切换到主机接收模式用于读. |
| 1 | x | Stop 条件 | 在主机发送模式,I2Cx\_DBR 内的 8 Bits数据发送完成之后在总线上发送一个 Stop 条件.  在主机接收模式,I2Cx\_CR{ACKNAK}必需置 1 用来发送一个 NAK 脉冲,接收的数据被存入 I2Cx\_DBR Register,然后在总线上发送一个 Stop 条件. |

##### 1. Start 条件

Start 条件和 I2Cx\_DBR 内的数据在 I2Cx\_CR{TB}被置 1 后开始发送.写Request ,I2C 总线停留在主机发送模式,读Request 将停留在主机接收模式.Repeated Start 条件,改变读写或者目标从设备地址,I2Cx\_DBR 将包含更新的从设备地址和 R/nW Bits.

I2C 不会清除 Start 条件.如果在开始发送 Start 条件的时候丢失总线仲裁,I2C 会在总线空闲的时候重新尝试发送一个 Start 条件.

##### 2. 无 Start 或者 Stop 条件

当 I2C 在发送多个数据字节的时候,I2Cx\_CR{START}=0,I2Cx\_CR{STOP}=0,此时无 Start 或者 Stop 条件.软件写数据字节,I2C 将 I2Cx\_SR{ITE}置 1 并且清除 I2Cx\_CR{TB}.软件继续写一个新的字节到 I2Cx\_DBR Register,并且把 I2Cx\_CR{TB}置 1,开始一个新的字节发送.这个过程一直继续,直到软件把I2Cx\_CR{START}或者I2Cx\_CR{STOP}置1, I2C 在 发送 完一 个 Start、 Stop 或 Repeated Start 条 件 后, I2Cx\_CR{START} 和 I2Cx\_CR{STOP}不会被自动清 0.

在每个字节与 ACK/NAK 被发送完成后,I2C 一直将 SCL 拉低等待,直到 I2Cx\_CR{TB}被置 1.

##### 3. 停止条件

停止条件结束一次数据传输.在主机发送模式,I2Cx\_CR{STOP}和 I2Cx\_CR{TB}必须置 1 来开始最后一个字节的传输.在主机接收模式,I2Cx\_CR{ACKNAK}、I2Cx\_CR{STOP}和

I2Cx\_CR{TB}必须置 1 来开始最后一个字节的接收.Stop 条件发送完成之后,软件必须把

I2Cx\_CR{STOP}清 0.

#### 15.3 数据传输顺序

I2C 以 1 字节递增的方式传输数据,遵循以下顺序:

1. Start
2. 7 Bits从机地址
3. R/nW Bits
4. Acknowledge
5. 8 Bits数据
6. Acknowledge
7. 重复步骤 5 和步骤 6
8. Repeated Start（重复步骤 1）或 Stop

#### 15.4 数据与寻址

I2C 数据 Buffer Register I2Cx\_DBR 和 I2C 从地址Register I2Cx\_SAR 管理数据和从机寻址. I2Cx\_DBR 包含 1 字节数据或者 7 Bits目标从机地址和 1 Bits R/nW.I2Cx\_SAR 包含 I2C 单元自身的从机地址.I2C 接收完一个完整字节数据和 ACK 后将数据存入 I2Cx\_DBR.发送时,CPU 将数据写入 I2Cx\_DBR,当 I2Cx\_CR{TB}置 1 后把数据发送到总线上.

##### 1. 主机或者从机发送模式:

1. 将数据写入 I2Cx\_DBR Register开始一次主机事务,或者在 I2Cx\_SR{ITE} 置 1 后发送下一个字节.
2. 当 I2Cx\_CR{TB} 置 1 后发送 I2Cx\_DBR 中的数据.
3. 如果Enable了 I2Cx\_CR{ITEIE},在发送完一个字节和 ACK 后会触发 I2Cx\_DBR 空interrupt.
4. 在CPU写 I2Cx\_DBR Register前,当I2C已准备好发送下一个字节,且无Stop条件,

I2C 处于等待状态,直到 CPU 写 I2Cx\_DBR Register并且把 I2Cx\_CR{TB} 置 1.

**Notice:** 在*FIFO*模式,以*TX FIFO*替代*I2Cx\_DBR*.

##### 2. 主机或者从机接收模式:

1. 当一个完整字节数据被接收后（Enable I2Cx\_CR{DRFIE},触发 I2Cx\_DBR 接收满interrupt,

I2Cx\_SR{IRF} 被置 1）,CPU 读 I2Cx\_DBR Register取回数据.

1. 当 ACK 完成后,I2C 将数据从Bits移Register传输到 I2Cx\_DBR Register.
2. I2C 处于等待模式,直到 I2Cx\_DBR Register被 CPU 读取.
3. CPU 读取 I2Cx\_DBR Register之后,I2C 更新 I2Cx\_CR{ACKNAK}Bits和 I2Cx\_CR{TB} Bits,allowed下一字节的传输.

**Notice:** 在*FIFO*模式,以*RX FIFO*替代*I2Cx\_DBR*.

##### 3. 从机寻址:

作为主机设备,I2C 必须要构建和发送一次事务的第一个字节.这个字节由 7 Bits的从机地址和 1 Bits的 R/nW 组成.第一个字节的发送必须要得到从设备的 ACK 响应.如果是写事务,I2C 保持在主机发送模式,同时从机保持在接收模式.如果是读事务,I2C 在收到 ACK 后马上切换到主机接收模式,同时从机切换到发送模式.如果收到 NAK,I2C 自动发送 Stop 条件并且把 I2Cx\_SR{BED}置 1 来中止当前事务.

(I2C)

##### 15.5 应答（ACK）

每一个字节的传输必须伴随 ACK,由接收的主机或者从机产生.发送方必须释放 SDA 线给接收方传输 ACK 脉冲.

在主机发送模式,如果目标接收从机未产生 ACK,SDA 线保持高电平指示一个 NAK.缺少

ACK 导致 I2C 将 I2Cx\_SR{BED}置 1 并产生interrupt,I2C 自动产生 Stop 条件并且中止传输.

在主机接收模式,I2C 发送 NAK 给发送从机通知从机停止发送数据,I2Cx\_CR{ACKNAK}控制总线上 ACK/NAK 的产生.按照 I2C 协议的规定,主机接收模式 NAK 不会将 I2Cx\_SR{BED}置

1.I2C 从总线上每接收一个字节会自动发送 ACK,在接收到最后一个字节之前软件必须将 I2Cx\_CR{ACKNAK}置 1 来发送 NAK.NAK 在最后一个字节被传输后发送,告知最后一个字节被发送完成.

在从机接收模式,I2C 自动对自身从机地址进行 ACK 响应,不论 I2Cx\_CR{ACKNAK}是否被置 1.在从机模式,I2C 自动在接收到的每一个字节数据后进行 ACK 响应,不论 I2Cx\_CR Register的 ACKNAK Bits是否被置 1.

在从机发送模式,接收NAK意味着当前这次传输的最后一个字节被发送完成.主机接着发送一个 Stop 条件或者 Repeated Start 条件.I2Cx\_SR{UB}保持为 1 直到一个 Stop 条件或者

Repeated Start 条件被接收到.

##### 15.6 仲裁

为兼容多主机,需要总线仲裁功能.总线仲裁用于在最小 I2C Start 条件时间内有 2 个或者更多主机同时产生 Start 条件的情形.

仲裁可以持续一段长的时间.如果从机地址和 R/nW Bits一致的话,仲裁移到数据阶段.由于 I2C 总线的 “线与” 属性,如果 2个或者所有主机输出同样的总线状态,不会丢失数据.如果地址,或者 R/nW Bits,或者数据不同,转变到高状态的主机（主机数据与 SDA 线不同）丢失仲裁,并且结束数据传输,将 I2Cx\_SR{ALD}置 1,返回空闲状态.

在 FIFO 模式,丢失仲裁的时候软件必须清空 FIFO.这可以通过清空发送和接收 FIFO 的读写指针Register来实现.

#### 15.7 主机模式

当软件开始执行读或者写操作,I2C 从默认的从机接收模式切换到主机发送模式.Start 条件之后跟随着 7 Bits的从机地址和 1 Bits的 R/nW.

当接收到 ACK 后,I2C 进入以下两种模式之一:

* 主机发送模式-写数据
* 主机接收模式-读数据

CPU 写 I2Cx\_CR Register来开始一次主机事务.

##### Table 15-2 主机事务

|  |  |  |
| --- | --- | --- |
| 主机动作 | 主机操作 | 定义 |
| 产生clock 输出 | 主机发送主机接收 | * 主机驱动 SCL 线 * I2Cx\_CR{SCLE} 和 I2Cx\_CR{UE} 必须置 1 |
| 写目标从机地址到  I2Cx\_DBR | 主机发送主机接收 | * CPU 在Enable Start 条件前写 I2Cx\_DBR[7:1] * 前 7 个Bits在 Start 条件之后发送 |
| 写 R/nW Bits到 I2Cx\_DBR | 主机发送主机接收 | * CPU 把 R/nW 控制Bits写入 I2Cx\_DBR 的最低Bits * 若 R/nW 为低则主机保持发送模式,若 R/nW 为高则主机切换到接收模式 |
| 发送 Start 条件 | 主机发送主机接收 | 在 7 Bits目标从机地址和 1 Bits R/nW 写入 I2Cx\_DBR Register之后,   * 软件把 I2Cx\_CR{START} 置 1 * 软件把 I2Cx\_CR{TB}置 1 开始发送 Start 条件 |
| 开始第一个字节传输 | 主机发送主机接收 | * CPU 写一个字节到 I2Cx\_DBR Register * 软件把 I2Cx\_CR{TB} 置 1,开始这个字节的发送 * 发送完成之后,I2Cx\_CR{TB} 被清 0,I2Cx\_SR{ITE} 被置 1 |
| 总线仲裁 | 主机发送主机接收 | 如果在同一个clock 周期内有多个主机在总线上发送了 Start 条件,那么总线仲裁必须产生,   * 只要有需要,I2C 仲裁就会产生.总线仲裁发生在目标从机地址和 R/nW Bits,以及数据传输阶段,直到除 1 个主机之外的主机都丢失总线.数据不会丢失. * 如果丢失仲裁,I2Cx\_SR{ALD} 会被置 1,I2C 切换到从机接收模式. * 如果在发送目标从机地址的时候丢失仲裁,I2C 会在总线空闲的时候再次尝试重发. |
| 写一个字节到 I2Cx\_DBR | 仅主机发送 | * 如果 I2Cx\_SR{ITE} 被置 1 并且 I2Cx\_CR{TB} 被清 0,当 I2Cx\_DBR 空interrupt被Enable,那么interrupt产生. * CPU 写一个字节到 I2Cx\_DBR Register,并且根据需要设   置合适的 Start/Stop 条件组合,然后把 I2Cx\_CR{TB} 置 |
|  |  | 1 发送数据.数据的 8 个Bits被从Bits移Register搬到串行总线  上.若发送前 I2Cx\_CR{STOP} 置 1,那么在数据的 8 个Bits传输完成之后会跟随一个 Stop 条件. |
| 等待接收从机 ACK | 仅主机发送 | 作为发送方,主机产生 ACK 的clock ,并且将 SDA 线释放给接收的从机发送 ACK. |
| 从 I2Cx\_DBR 读取一个字节 | 仅主机接收 | 在 I2Cx\_CR{ACKNAK} 被读取之后,Bits移Register内的 8 Bits数据被搬到 I2Cx\_DBR Register,   * 当 I2Cx\_SR{IRF} 被置 1 且 I2Cx\_CR{TB} 被清 0 时,   CPU 读取 I2Cx\_DBR Register.可以Enable I2Cx\_DBR Register接收满interrupt通知 CPU.   * 当 I2Cx\_DBR 被读取完,如果 I2Cx\_SR{ACKNAK} 被清 0（代Table ACK）,软件必须把 I2Cx\_CR{ACKNAK} 清 0 并且把 I2Cx\_CR{TB} 置 1 来开始下一个字节的读取.  如果 I2Cx\_SR{ACKNAK} 被置 1（代Table NAK）,   I2Cx\_CR{TB} 被清除,I2Cx\_CR{STOP} 被置 1,且  I2Cx\_SR{UB} 被置 1,最后一个字节已经被读取到 I2Cx\_DBR Register,I2C 正在发送 Stop 条件.   * 如果 I2Cx\_SR{ACKNAK} 被置 1（代Table NAK）,并且   I2Cx\_CR{TB} 被清 0,但是 I2Cx\_CR{STOP} 被清 0,软  件有两个选择:   * 1. 把 I2Cx\_CR{START} 置 1,将新的目标从机地址写入 I2Cx\_DBR,把 I2Cx\_CR{TB} 置 1,发送一个   Repeated Start 条件.   * 1. 把 I2Cx\_CR{MA} 置 1,并且保持 I2Cx\_CR{TB}为 0,仅发送一个 Stop 条件. |
| 发送 ACK 到发送从机 | 仅主机接收 | * 作为接收主机一方,在 ACK 期间,产生 ACK clock ,并且驱动 SDA 线 * 如果下一个字节为最后一个事务,软件需要把   I2Cx\_CR{ACKNAK} 置 1 来产生 NAK. |
| 产生 Repeated Start 条件 | 主机发送主机接收 | 使用 Repeated Start 代替 Stop 条件可以在不释放总线的情  况下继续新的传输   Repeated Start 条件在最后一个字节数据被传输后产生  软件必须把 7 Bits的目标从机地址和 1 Bits的 R/nW Bits写入  I2Cx\_DBR Register,然后把 I2Cx\_CR{START} 置 1,再把 I2Cx\_CR{TB} 置 1 |
| 产生 Stop 条件 | 主机发送主机接收 | * Stop 条件在最后一个字节数据被传输后产生 * I2Cx\_CR{STOP} 要在最后一个字节被传输前置 1 |

##### 15.8 FIFO 模式

FIFO 模式只能在主机模式被使用.



**Figure 15-3 FIFO 模式示意Figure**

FIFO 模式可以用于发送和接收,以帮助减少 I2Cx\_DBR Register空interrupt和满interrupt,FIFO allowed读取和写入多个字节而不需要在每个字节操作之后interrupt CPU.

DMA 被用于改善传输长度超过 8 个字节数据的 I2C 事务,整个事务可用 DMA 的方式完成,而不用产生多次 FIFO interrupt.

FIFO 模式向下兼容普通模式,通过把 I2Cx\_CR{FIFO\_EN}清 0 来禁用 FIFO 模式.

发送 FIFO的宽度为12 Bits,4 个控制Bits,8 个数据Bits,深度为 8.4个控制Bits为 I2Cx\_CR[3:0],这是发送每个字节所必须的控制Bits.当一个字节被传输之后,新的字节从 TX FIFO 拷贝到Bits移

Register,控制Bits被拷贝到 I2Cx\_CR[3:0].这个字节现在被传输,持续循环直到 Stop 条件产生.

接收 FIFO 为宽度为 8 Bits,用于保存接收到的数据,深度为 16.每个字节的控制Bits与一个空数据被保存到 TX FIFO 的相应Bits置.当接收 FIFO 半满,会产生 FIFO 半满interrupt或者 DMA Request ,将 FIFO 中的数据读出.

为支持 FIFO 功能,同时完整利用 FIFO 的容量,需要配置以下状态和控制Bits:

1. I2Cx\_CR{FIFO\_EN} 置 1 Enable FIFO 模式
2. I2Cx\_CR{TXBEGIN} 置 1 开始事务
3. Enable I2Cx\_CR[31:27] 的 FIFO 相关interruptBits,可通过 I2Cx\_SR[31:27] 查询相应interrupt状态
4. 每次事务完成之后（Stop 条件发送完）触发 TXDONE interrupt
5. I2Cx\_CR{DMA\_EN} 用来Enable/禁用 DMA 模式

在DMA模式,I2Cx\_CR[31:28] 相关的FIFOinterrupt必须禁用,同时 I2Cx\_CR{DMA\_EN} 置1.这样,所有 DMA Request 被发送到 DMA,而不是 CPU.I2Cx\_CR{TXDONE\_IE} 在 FIFO 模式和

DMA 模式都需要置 1,用来通知 CPU 事务的结束.

#### 15.9 从机模式

##### Table 15-3 从机事务

|  |  |  |
| --- | --- | --- |
| 从机动作 | 从机操作 | 定义 |
| 从机接收（默认模式） | 仅从机接收 | * I2C 监视所有从机地址事务 * I2Cx\_CR{UE} 必须置 1 * I2C 监视总线上的Start条件.若检测到 Start 条件,接口读取前 8 Bits数据,并把前 7 Bits与自身从机地址做比较,若匹配则响应 ACK * 若首字节的第 8 Bits（R/nW）为低,那么 I2C 保持在从机接收模式,并把 I2Cx\_SR{SAD} 清 0.若 R/nW 为高,I2C 切换到从机发送模式,并把   I2Cx\_SR{SAD} 置 1 |
| 设置从机地址检测Bits | 从机接收从机发送 | * 用来指示接口检测到匹配的 I2C 寻址 * 若Enable I2Cx\_CR{SADIE},在匹配的从机地址被接收和 ACK 响应之后,interrupt产生,I2Cx\_SR{SAD} 置   1 |
| 从 I2Cx\_DBR 读取 1 个字  节 | 仅从机接收 | * 8Bits数据从总线上读取到Bits移Register,在整个字节被接收完成和ACK/NAK完成之后,Bits移Register内的数据被搬到 I2Cx\_DBR Register * 当 I2Cx\_SR{IRF} 置1,且 I2Cx\_CR{TB} 清0,若Enable I2Cx\_CR{DRFIE},I2Cx\_DBR 接收满interrupt产生 * 软件从 I2Cx\_DBR 读取数据,并根据需要配置   I2Cx\_CR{ACKNAK},把 I2Cx\_CR{TB} 置 1,这个  操作使从机退出等待模式,接续接收主机的数据 |
| 响应 ACK 到发送主机 | 仅从机接收 | * 作为接收从机,I2C 在 SCL 为高的时候将 SDA 线拉低产生 ACK * ACK/NAK 由 I2Cx\_CR{ACKNAK} 控制 |
| 写 1 个字节到 I2Cx\_DBR | 仅从机发送 | * I2Cx\_SR{ITE} 置 1,I2Cx\_CR{TB} 清 0,若Enable I2Cx\_CR{ITEIE} interrupt,I2Cx\_DBR 发送空interrupt产生 * 软 件 把 数 据 写 入 I2Cx\_DBR 寄 存 器 , 然 后 把   I2Cx\_CR{TB} 置 1 开始数据的发送 |
| 等待接收主机的 ACK | 仅从机发送 | 作为发送从机,I2C 释放 SDA 线等待接收主机拉低响应  ACK |

##### 15.10 clock 复Bits

每个 I2C 接口都有独立的 APB 总线clock 和独立的 APB 总线复Bits.

复Bits前软件必须保证I2Cx\_CR{UE}为0,并且复Bits后保证总线在空闲状态（I2Cx\_SR{IBB}为0）. Reset 时,除 I2Cx\_SAR Register之外的所有Register均恢复到默认的复Bits状态,I2Cx\_SAR 不受复Bits影响.

复Bits操作顺序:

1. 将 I2Cx\_CR{UR} 置 1,并把 I2Cx\_CR Register的其余Bits清 0
2. 将 I2Cx\_SR Register清 0
3. 将 I2Cx\_CR{UR} 清 0

* 1. **interruptRequest**

通过 I2Cx\_CR 配置interruptEnable,查询 I2Cx\_SR 相应Bits可以获取interrupt状态.

* 1. **DMA Request**

通过 I2Cx\_CR{DMA\_EN} Enable DMA,支持发送和接收.

##### 15.13 I2C 相关RegisterDescription

I2C0 基地址:0x40007000 I2C1 基地址:0x40014000

I2C2 基地址:0x40015000

**Table 15-4 I2C Register Summary**

|  |  |  |
| --- | --- | --- |
| Register | Offset | Description |
| I2Cx\_CR | 0x00 | Control Register |
| I2Cx\_SR | 0x04 | 状态Register |
| I2Cx\_SAR | 0x08 | 从地址Register |
| I2Cx\_DBR | 0x0C | 数据 Buffer Register |
| I2Cx\_LCR | 0x10 | 加载计数Register |
| I2Cx\_WCR | 0x14 | 等待计数Register |
| I2Cx\_RST\_CYCL | 0x18 | 复Bits周期Register |
| I2Cx\_BMR | 0x1C | 总线监视Register |
| I2Cx\_WFIF0 | 0x20 | 发送 FIFO Register |
| I2Cx\_WFIFO\_WPTR | 0x24 | 发送 FIFO 写指针Register |
| I2Cx\_WFIFO\_RPTR | 0x28 | 发送 FIFO 读指针Register |
| I2Cx\_RFIFO | 0x2C | 接收 FIFO Register |
| I2Cx\_RFIFO\_WPTR | 0x30 | 接收 FIFO 写指针Register |
| I2Cx\_RFIFO\_RPTR | 0x34 | 接收 FIFO 读指针Register |
| I2Cx\_RESV[2] | 0x38 | 4 x 2 字节保留 |
| I2Cx\_WFIFO\_STATUS | 0x40 | 写 FIFO 状态Register |
| I2Cx\_RFIFO\_STATUS | 0x44 | 读 FIFO 状态Register |

###### 15.13.1 I2Cx\_CR (x=0, 1, 2)

Offset:0x00 Reset Value:0x00000200

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| **31** | **30** | **29** | **28** | **27** | **26** | **25** | **24** |
| RXOV\_IE | RXF\_IE | RXHF\_IE | TXE\_IE | TXDONE\_IE | MSDE | MSDIE | SSDIE |
| rw-0h | rw-0h | rw-0h | rw-0h | rw-0h | rw-0h | rw-0h | rw-0h |
| **23** | **22** | **21** | **20** | **19** | **18** | **17-16** | |
| SADIE | BEIE | RESERVED | DRFIE | ITEIE | ALDIE | RESERVED | |
| rw-0h | rw-0h | r-0h | rw-0h | rw-0h | rw-0h | r-0h | |
| **15** | **14** | **13** | **12** | **11** | **10** | **9-8** | |
| RESERVED | UE | SCLE | MA | IBRR | UR | MODE | |
| r-0h | rw-0h | rw-0h | rw-0h | rw-0h | rw-0h | rw-2h | |
| **7** | **6** | **5** | **4** | **3** | **2** | **1** | **0** |
| DMA\_EN | RESERVED | FIFOEN | TXBEGIN | TB | ACKNAK | STOP | START |
| rw-0h | r-0h | rw-0h | rw-0h | rw-0h | rw-0h | rw-0h | rw-0h |

**Bits 31 RXOV\_IE:**接收 FIFO overrun interruptEnable.

* 0:禁用接收 FIFO overrun interrupt
* 1:Enable接收 FIFO overrun interrupt**Bits 30** **RXF\_IE:**接收 FIFO 满interruptEnable.
* 0:禁用接收 FIFO 满interrupt
* 1:Enable接收 FIFO 满interrupt**Bits 29** **RXHF\_IE:**接收 FIFO 半满interruptEnable.
* 0:禁用接收 FIFO 半满interrupt
* 1:Enable接收 FIFO 半满interrupt**Bits 28** **TXE\_IE:**发送 FIFO 空interruptEnable.
* 0:禁用发送 FIFO 空interrupt
* 1:Enable发送 FIFO 空interrupt**Bits 27** **TXDONE\_IE:**事务完成interruptEnable.
* 0:禁用事务完成interrupt
* 1:Enable事务完成interrupt**Bits 26** **MSDE:**主机停止检测Enable.
* 0:禁用主机停止检测功能
* 1:Enable主机停止检测功能**Bits 25** **MSDIE:**主机停止检测interruptEnable.
* 0:禁用主机停止检测interrupt
* 1:Enable主机停止检测interrupt

**Bits 24** **SSDIE:**从机停止检测interruptEnable.

* 0:禁用从机停止检测interrupt
* 1:Enable从机停止检测interrupt**Bits 23** **SADIE:**从机地址检测interruptEnable.
* 0:禁用从机地址检测interrupt
* 1:Enable从机地址检测interrupt

**Bits 22** **BEIE:**总线错误interruptEnable.

* 0:禁用总线错误interrupt
* 1:Enable总线错误interrupt

**Bits 21** **RESERVED:**保留,不可更改.

**Bits 20** **DRFIE:**I2Cx\_DBR 接收满interruptEnable.

* 0:禁用 I2Cx\_DBR 接收满interrupt
* 1:Enable I2Cx\_DBR 接收满interrupt**Bits 19 ITEIE:**I2Cx\_DBR 发送空interruptEnable.
* 0:禁用 I2Cx\_DBR 发送空interrupt
* 1:Enable I2Cx\_DBR 发送空interrupt**Bits 18** **ALDIE:**仲裁丢失检测interruptEnable.
* 0:禁用仲裁丢失检测interrupt
* 1:Enable仲裁丢失检测interrupt

**Bits 17-15** **RESERVED:**保留,不可更改.

**Bits 14 UE:**I2C 接口单元Enable.

* 0:禁用 I2C 接口单元
* 1:Enable I2C 接口单元,默认为从机接收模式

软件必须保证在Enable I2C 接口单元前总线为空闲状态,并且在置 1 或者清 0 该Bits前Enable I2C 内部clock .

**Bits 13** **SCLE:**SCL Enable.

* 0:禁用 SCL 线
* 1:Enable主模式 I2C clock 输出**Bits 12 MA:**主机中止.

用于 Master 模式产生 Stop 条件.

* 0:STOP 为 1 产生 Stop 条件
* 1:产生 Stop 条件无需发送数据

在主机发送模式,当一个数据字节发送完成,TB 被清除,I2Cx\_SR{ITE} 被置 1,若没有更多的数据需要被发送,可将 MA 置 1 产生 Stop 条件释放总线.在主机接收模式,当 STOP 为 0,发送 NAK 后,且没有发送

Repeated Start 条件,可将 MA 置 1 产生 Stop 条件释放总线.TB 必须保持为 0.

**Bits 11 IBRR:**总线复BitsRequest .

* 0:无效
* 1:总线复Bits,该Bits自动清 0

**Bits 10 UR:**单元复Bits.

* 0:无效
* 1:I2C 单元复Bits

**Bits 9-8** **MODE:**主机总线clock 模式.

* 00:标准模式,100Kbps
* 01:快速模式,400Kbps **Bits 7** **DMA\_EN:**DMA Enable.
* 0:禁用 DMA Request
* 1:Enable DMA Request

**Bits 6** **RESERVED:**保留,不可更改.

**Bits 5** **FIFOEN:**FIFO 模式Enable.

* 0:禁用 FIFO 模式
* 1:Enable FIFO 模式

**Bits 4** **TXBEGIN:**事务开始.

* 0:无事务开始
* 1:新的事务开始

该Bits在产生 Stop 条件后被硬件清 0,软件需要在开始新的事务时置 1.

**Bits 3** **TB:**传输字节,用来在总线上发送或者接收一个字节.

* 0:一个字节收发完成后被清 0
* 1:发送或者接收一个字节

I2C单元会监视这个Bits来确定该字节是否收发完成.在主机或从机模式,在一个字节包括ACK收发完成后,

I2C 会一直将 SCL 拉低直到 TB 被置 1.

**Bits 2** **ACKNAK:**主机接收模式 ACK/NAK 控制Bits.

* 0:接收完成一个字节后发送 ACK
* 1:接收完成一个字节后发送 NAK

从机模式,从地址匹配或者接收完成时,I2C 单元自动发送一个 ACK,不论 ACKNAK 是否置 1.

**Bits 1** **STOP:**产生 Stop 条件.

* 0:不产生 Stop 条件
* 1:产生 Stop 条件

用来在主机模式传输完下一个字节后在总线上产生 Stop 条件.在主机接收模式,ACKNAK 必须与 STOP Bits同时置 1.

**Bits 0** **START:**产生 Start 条件.

* 0:不产生 Start 条件
* 1:产生 Start 条件

用来在主机模式在总线上产生 Start 条件.

###### 15.13.2 I2Cx\_SR (x=0, 1, 2)

Offset:0x04 Reset Value:0x00000000

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| **31** | **30** | **29** | **28** | **27** | **26** | **25** |
| RXOV | RXF | RXHF | TXE | TXDONE | MSD | RESERVED |
| rw1c-0h | rw1c-0h | rw1c-0h | rw1c-0h | rw1c-h | r1ch | r-0h |
| **24** | **23** | **22** | **21** | **20** | **19** | **18** |
| SSD | SAD | BED | RESERVED | IRF | ITE | ALD |
| rw1c-0h | rw1c-0h | rw1c-0h | r-0h | rw1c-0h | rw1c-0h | rw1c-0h |
| **17** | **16** | **15** | **14** | **13-8** | **7-0** | |
| RESERVED | IBB | UB | ACKNAK | RESERVED | RESERVED | |
| r-0h | r-0h | r-0h | r-0h | r-0h | r-0h | |

**Bits 31 RXOV:**接收 FIFO overrun 标志.

* 0:接收 FIFO 未发生 overrun interrupt
* 1:接收 FIFO 发生 overrun,写 1 清 0 **Bits 30** **RXF:**接收 FIFO 满标志.
* 0:接收 FIFO 未满
* 1:接收 FIFO 满,写 1 清 0 **Bits 29** **RXHF:**接收 FIFO 半满标志.
* 0:接收 FIFO 未半满
* 1:接收 FIFO 半满,写 1 清 0 **Bits 28** **TXE:**发送 FIFO 空标志.
* 0:发送 FIFO 不为空
* 1:发送 FIFO 空,写 1 清 0 **Bits 27** **TXDONE:**事务完成标志,FIFO 模式使用.
* 0:事务未完成
* 1:事务完成,写 1 清 0 **Bits 26** **MSD:**主机停止检测标志（主模式有效）.
* 0:未检测到主机停止
* 1:检测到主机停止,写 1 清 0 **Bits 25** **RESERVED:**保留,不可更改.**Bits 24** **SSDIE:**从机停止检测标志.
* 0:未检测到从机停止
* 1:检测到从机停止,写 1 清 0 **Bits 23** **SAD:**从机地址检测标志.
* 0:无匹配的从机地址被检测到
* 1:检测到匹配的从机地址,写 1 清 0

**Bits 22** **BED:**总线错误标志.

* 0:未检测到总线错误
* 1:检测到总线错误,写 1 清 0

两种情况下回产生该标志,主机发送一个字节后未收到 ACK,或者从机接收产生一个 NAK 脉冲.

**Bits 21** **RESERVED:**保留,不可更改.

**Bits 20 IRF:**I2Cx\_DBR 接收满标志.

* 0:I2Cx\_DBR 没有收到新的数据字节或者 I2C 总线处于空闲状态
* 1:I2Cx\_DBR 收到一个新的数据字节,写 1 清 0 **Bits 19 ITE:**I2Cx\_DBR 发送空.
* 0:数据仍在发送中
* 1:总线发送完成一个数据字节,写 1 清 0

**Bits 18 ALD:**仲裁丢失标志,多主机场景使用.

* 0:获得仲裁,或者未发生仲裁
* 1:丢失仲裁,写 1 清 0

**Bits 17** **RESERVED:**保留,不可更改.

**Bits 16 IBB:**总线忙标志.

* 0:总线空闲或者总线正在被 I2C 接口使用
* 1:总线忙但是未被 I2C 接口使用**Bits 15 UB:**I2C 接口单元忙标志.
* 0:I2C 接口单元空闲
* 1:I2C 接口单元忙

**Bits 14 ACKNAK:**ACK/NAK 状态标志.

* 0:收到或发送完一个 ACK
* 1:收到或发送完一个 NAK

在从机发送模式,该Bits用于确定被发送的字节是否是最后一个.该Bits在每个字节的 ACK/NAK 信息被收到后都会更新.

**Bits 13-0** **RESERVED:**保留,不可更改.

###### 15.13.3 I2Cx\_SAR (x=0, 1, 2)

Offset:0x08

Reset Value:0x00000000

|  |  |
| --- | --- |
| **31-7** | **6-0** |
| RESERVED | SLAVE\_ADDRESS |
| r-0h | rw-0h |

**Bits 31-7** **RESERVED:**保留,不可更改.

**Bits 6-0 SLAVE\_ADDRESS:**从地址,从机模式使用.

###### 15.13.4 I2Cx\_DBR (x=0, 1, 2)

Offset:0x0C

Reset Value:0x00000000

|  |  |
| --- | --- |
| **31-8** | **7-0** |
| RESERVED | DATA\_BUFFER |
| r-0h | rw-0h |

**Bits 31-8 RESERVED:**保留,不可更改.

**Bits 7-0** **DATA\_BUFFER:**收发数据 Buffer.

###### 15.13.5 I2Cx\_LCR (x=0, 1, 2)

Offset:0x10 Reset Value:0x18183a7e

|  |  |  |
| --- | --- | --- |
| **31-18** | **17-9** | **8-0** |
| RESERVED | FLV | SLV |
| r-1818h | rw-1dh | rw-7eh |

**Bits 31-18** **RESERVED:**保留,不可更改.

**Bits 17-9 FLV:**主机快速clock 模式相Bits减幅器装载值.

**Bits 8-0** **SLV:**主机标准clock 模式相Bits减幅器装载值.

###### 15.13.6 I2Cx\_WCR (x=0, 1, 2)

Offset:0x14 Reset Value:0x0000143a

|  |  |
| --- | --- |
| **31-5** | **4-0** |
| RESERVED | COUNT |
| r-a1h | rw-1ah |

**Bits 31-5 RESERVED:**保留,不可更改.**Bits 4-0 COUNT:**快速与标准clock 模式 setup 与 hold 次数计数值.

###### 15.13.7 I2Cx\_RST\_CYCL (x=0, 1, 2)

Offset:0x18 Reset Value:0x00000000

|  |  |
| --- | --- |
| **31-4** | **3-0** |
| RESERVED | RST\_CYC |
| r-0h | rw-0h |

**Bits 31-4** **RESERVED:**保留,不可更改.

**Bits 3-0 RST\_CYC:**总线复Bits SCL 周期计数.

15.13.8 I2Cx\_BMR (x=0, 1, 2)

Offset:0x1C

Reset Value:0x00000003

|  |  |  |
| --- | --- | --- |
| **31-2** | **1** | **0** |
| RESERVED | SCL | SDA |
| r-0h | r-1h | r-1h |

**Bits 31-2** **RESERVED:**保留,不可更改.**Bits 1 SCL:**SCL 引脚状态.

**Bits 0 SDA:**SDA 引脚状态.

15.13.9 I2Cx\_WFIF0 (x=0, 1, 2)

Offset:0x20 Reset Value:0x00000000

|  |  |  |
| --- | --- | --- |
| **31-12** | **11-8** | **7-0** |
| RESERVED | CONTROL | DATA |
| r-0h | w-0h | w-0h |

**Bits 31-12** **RESERVED:**保留,不可更改.**Bits 11-8 CONTROL:**收发数据控制Bits.**Bits 7-0 DATA:**写事务发送数据和读事务空数据.

###### 15.13.10 I2Cx\_WFIFO\_WPTR (x=0, 1, 2)

Offset:0x24 Reset Value:0x00000000

|  |  |
| --- | --- |
| **31-4** | **3-0** |
| RESERVED | DATA |
| r-0h | rw-0h |

**Bits 31-4** **RESERVED:**保留,不可更改.

**Bits 3-0 DATA:**发送 FIFO 软件写入Bits置指针.

###### 15.13.11 I2Cx\_WFIFO\_RPTR (x=0, 1, 2)

Offset:0x28 Reset Value:0x00000000

|  |  |
| --- | --- |
| **31-4** | **3-0** |
| RESERVED | DATA |
| r-0h | rw-0h |

**Bits 31-4** **RESERVED:**保留,不可更改.

**Bits 3-0 DATA:**发送 FIFO 硬件读取Bits置指针.

15.13.12 I2Cx\_RFIFO (x=0, 1, 2)

Offset:0x2C

Reset Value:0x00000000

|  |  |
| --- | --- |
| **31-8** | **7-0** |
| RESERVED | DATA |
| r-0h | r-0h |

**Bits 31-8 RESERVED:**保留,不可更改.**Bits 7-0 DATA:**读事务接收数据.

###### 15.13.13 I2Cx\_RFIFO\_WPTR (x=0, 1, 2)

Offset:0x30 Reset Value:0x00000000

|  |  |
| --- | --- |
| **31-4** | **3-0** |
| RESERVED | DATA |
| r-0h | r-0h |

**Bits 31-4 RESERVED:**保留,不可更改.

**Bits 3-0 DATA:**接收 FIFO 硬件写入Bits置指针.

###### 15.13.14 I2Cx\_RFIFO\_RPTR (x=0, 1, 2)

Offset:0x34 Reset Value:0x00000000

|  |  |
| --- | --- |
| **31-4** | **3-0** |
| RESERVED | DATA |
| r-0h | r-0h |

**Bits 31-4** **RESERVED:**保留,不可更改.

**Bits 3-0 DATA:**接收 FIFO 软件读取Bits置指针.

###### 15.13.15 I2Cx\_WFIFO\_STATUS (x=0, 1, 2)

Offset:0x40 Reset Value:0x00000000

|  |  |  |  |
| --- | --- | --- | --- |
| **31-16** | **15-9** | **8-1** | **0** |
| RESERVED | WFIFO\_SIZE | WFIFO\_EMPTY | WFIFO\_FULL |
| r-0h | r-0h | r-0h | r-0h |

**Bits 31-6** **RESERVED:**保留,不可更改.**Bits 5-2 WFIFO\_SIZE:**发送 FIFO 空间.**Bits 1** **WFIFO\_EMPTY:**发送 FIFO 空.**Bits 0 WFIFO\_FULL:**发送 FIFO 满.

###### 15.13.16 I2Cx\_RFIFO\_STATUS (x=0, 1, 2)

Offset:0x44 Reset Value:0x00000000

|  |  |  |  |
| --- | --- | --- | --- |
| **31-24** | **23-16** | **15-8** | **7-4** |
| RESERVED | RESERVED | RESERVED | RFIFO\_SIZE |
| r-0h | r-0h | r-0h | r-0h |
| **3** | **2** | **1** | **0** |
| RFIFO\_EMPTY | RFIFO\_FULL | RFIFO\_HALFFULL | RFIFO\_OVERRUN |
| r-0h | r-0h | r-0h | r-0h |

**Bits 31-8** **RESERVED:**保留,不可更改.**Bits 7-4 RFIFO\_SIZE:**接收 FIFO 空间.**Bits 3** **RFIFO\_EMPTY:**接收 FIFO 空.**Bits 2 RFIFO\_FULL:**接收 FIFO 满.**Bits 1 RFIFO\_HALFFULL:**接收 FIFO 半满.**Bits 0 RFIFO\_OVERRUN:**接收 FIFO overrun.

### 16. 模数转换器 (ADC)

#### 16.1 Introduction

12 Bits模数转换器（Analog to Digital Converter）,支持 8 个外部通道,7 个内部通道,内部通道可采集 VBAT/3,最高支持 1M 采样率.支持单端和差分两种模式,单端量程 0.1V~1.1V,差分量程 -1.0~1.0V.可配置 16 个采样序列,支持连续、单次、非连续采样方式.支持软件触发和硬件触发,触发源可配.支持 DMA Request 和interruptRequest .



**Figure 16-1 ADC 框Figure**

#### 16.2 输入模式

支持配置为单端与差分模式.外部通道支持单端与差分模式,内部通道只支持单端模式.差分为固定组合,不支持随意配对,其中 0/1 通道为一组,2/3 通道为一组,4/5 通道为一组,6/7 通道为一组.单端和差分仅在采样阶段控制不同,保持阶段没有区别,最后的数据中差分输入最高Bits为符号Bits（11bit 数据Bits,1 个符号Bits）,单端输入为 12bit 数据Bits,没有符号Bits.通过采样通道差分/单端选择Register ADC\_DIFFSEL 配置输入模式.

#### 16.3 采样通道

* **外部通道:**8 个,单端模式各通道独立,差分模式则每两个通道为一组,不可拆分.
* **内部通道:**7 个,包括 DAC 输出、内部 VRef、VDD/3（电池电量）、Vts（内部温度传感器）、内部测试专用.内部通道不支持差分模式.

**Table 16-1 ADC 采样通道**

|  |  |  |
| --- | --- | --- |
| 采样通道号 | 采样内容 | 备注 |
| 1 | ADC\_PAD\_IN<0> | gpio11 |
| 2 | ADC\_PAD\_IN<1> | gpio08 |
| 3 | ADC\_PAD\_IN<2> | gpio05 |
| 4 | ADC\_PAD\_IN<3> | gpio04 |
| 5 | ADC\_PAD\_IN<4> | gpio50 |
| 6 | ADC\_PAD\_IN<5> | gpio49 |
| 7 | ADC\_PAD\_IN<6> | gpio48 |
| 8 | ADC\_PAD\_IN<7> | gpio47 |
| 9 | OPA0\_ADC\_OUT |  |
| 10 | OPA1\_ADC\_OUT |  |
| 11 | OPA2\_ADC\_OUT |  |
| 12 | DCTEST\_OUT |  |
| 13 | TD\_OUT\_TEST |  |
| 14 | DAC\_CORE\_AOUT |  |
| 15 | VBAT31 |  |

VBAT31 需要通过模拟部分 RESV1 Register的 D\_VBAT\_DIV3\_EN 置BitsEnable VBAT/3 分压.这个通道名义为 VBAT 的 1/3 分压,精确值为 1/3.06.

#### 16.4 触发方式

* **软件方式:**ADC\_START 上升沿决定转换立即开始.
* **硬件方式:**支持 Timer 与 IO 触发,10 个触发源供选择,可配置触发电平.

通过配置Register ADC\_CFGR 的 TRIG\_SEL Bits选择触发方式,通过配置 EXT\_TRIG\_SEL Bits选择触发源.

#### 16.5 低功耗运行

ADC\_DR 数据被读走或者 EOC 标志被清除后才能接收新的触发Request ,可以防止 overrun,但可能 bypass 触发Request .

**16.6 溢出控制**

控制 overrun 发生时,数据Register采样新数据或者保持.

#### 16.7 采样模式

通过 ADC\_CFGR{CONV\_MODE} 配置采样模式:

支持采样序列配置,采样序列最多 16 个通道,单端和差分通道都可以配置.差分模式,采样序列仅配置 P 端即可.采样通道可以重复配置相同通道以决定每次序列多次采样该通道.通过通道采样序列Control Register ADC\_SEQR0 和 ADC\_SEQR1 配置采样序列,每 4 Bits配置 1 个采样通道,两个 32 BitsRegister共 64 Bits,最多可以配置 16 个采样通道.

* **连续采样:**一旦触发有效,则开始连续地转化选定的输入序列,每轮循环完成后自动开始新一轮循环,直到软件配置 stop.
* **单次采样:**每次触发执行一次采样序列循环,采样完成自动结束.
* **非连续采样:**序列中的每一次 ADC 转化都需要硬件或软件触发,如果一个序列完后,再次触发又从该序列的开头开始；而连续和单次模式,每次触发都会完成一个完整序列.

#### 16.8 参考电压

模拟部分 RST Register的 D\_ADC\_SEL\_VREF Bits配置参考电压,清 0 配置为外部参考电压,置

1 配置为内部参考电压,默认为 1.

* **内部参考电压:**VRef,1.2V
* **外部参考电压:**VREFP/3,VREFP 不可超过 3.6V,48PIN 内部 VREFP 与 VDDA 有连接

##### 16.9 数据 Buffer

1 个 12bit 的数据 buffer,差分模式下最高Bits为符号Bits.

|  |  |  |
| --- | --- | --- |
| ADC编码 | 差分模式意义 | 单端模式意义 |
| 1111\_1111\_1111 | +Vref**(1)** | +Vref**(1)** |
| … | … | … |
| … | … | … |
| … | … | … |
| … | … | … |
| … | … | … |
| 1000\_0000\_0001 | +Vref/2048**(1)** | +Vref/2+Vref/4096**(1)** |
| 1000\_0000\_0000 | 0 | +Vref/2**(1)** |
| 0111\_1111\_1111 | -Vref/2048**(1)** | +Vref/2-Vref/4096**(1)** |
| … | … | … |
| … | … | … |
| … | … | … |
| … | … | … |
| … | … | … |
| 0000\_0000\_0000 | -Vref**(1)** | 0 |

***(1)*** 为校准前从数据*buffer*读到的值.

差分模式的量程为 -1.0~1.0V,单端模式的量程为 0.1~1.1V.为了纠正 ADC 模拟电路上的误差,在出厂前会对 ASR6601 进行校准,校准数据 Offset 和 Gain 存储在 Flash 中,用户需要将从 ADC\_DR 读到的数据做一个转换才能得到最终的 AD 值,公式如下:

***V = (Vout – Offset) / Gain***

其中 Vout 为从数据 buffer 中读到的值.

##### 16.10 DMA Request

采用 Request 与 Clear 方式,数据 buf 为 1 个 12bit,因此 Buffer 数据满则产生Request .通过 ADC\_CFGR{DMA\_EN} 配置.

##### 16.11 interruptRequest

ADC interrupt包括单次转换完成 EOC,序列转换完成 EOS,溢出 OVERRUN.通过interruptEnableRegister ADC\_IER Enableinterrupt,通过interrupt状态Register ADC\_ISR 查询interrupt.

**16.12 低功耗工作与唤醒**

支持 Sleep 模式interrupt唤醒与事件唤醒.

#### 16.13 clock 和复Bits

总线复Bits和工作clock 复Bits独立,支持 APB 总线clock ,可配置内部分频,接口clock 来源,包括 sys\_clk、apb\_x\_pclk、pll\_clk 和 rco48m\_clk.

##### 16.14 ADC 相关RegisterDescription

基地址:0x40017000

**Table 16-2 ADC Register Summary**

|  |  |  |
| --- | --- | --- |
| Register | Offset | Description |
| ADC\_CR | 0x00 | Control Register |
| ADC\_CFGR | 0x04 | 配置Register |
| ADC\_SEQR0 | 0x08 | 通道采样序列Control Register 0 |
| ADC\_SEQR1 | 0x0C | 通道采样序列Control Register 1 |
| ADC\_DIFFSEL | 0x10 | 采样通道差分/单端选择Register |
| ADC\_ISR | 0x14 | interrupt和状态Register |
| ADC\_IER | 0x18 | interruptEnableRegister |
| ADC\_DR | 0x1C | 数据Register |

###### 16.14.1 ADC\_CR

Offset:0x00

Reset Value:0x00000000

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **31-4** | **3** | **2** | **1** | **0** |
| RESERVED | STOP | START | DIS | EN |
| r-0h | rw-0h | rw-0h | w-0h | rw-0h |

**Bits 31-4** RESERVED**:**保留,不可更改.

**Bits 3 STOP:**ADC 转换停止控制.

* 写 0:无效
* 写 1:控制 ADC 转换强制停止,读该Bits为 1 Table示 STOP 的关闭动作正在执行**Notice:**

1. 软件对该Bits写*1*来强制停止当前转换,当前转换的数据会被丢弃,扫描序列也会回到初始状态；该Bits硬件自动清零.
2. 软件查询到 *START* 关闭后等待 *3* 拍 *ADCCLK* 后才能再次配置 *START*；或等待 *1* 拍

*ADC\_CFGR{CLK\_DIV}* 后才能配置*DIS*关闭*ADC*功能.

1. 仅在 *START=1*且*STOP=0* 时对该Bits的写*1*动作才有效.
2. 配置*STOP*关闭*ADC*转换前,建议先将触发源关闭,或触发电平处于无效状态.

**Bits 2 START:**ADC 转换开启控制.

* 写 0:无效
* 写 1:开启 ADC 转换,读该Bits为 1 Table示 ADC 正在转换

该Bits软件置 1 开启 ADC 转换功能,并根据 ADC\_CFGR{TRIG\_SEL} 的配置决定 ADC 转换马上开始（软件触发模式）,或者等待硬件触发事件才开始；仅在 EN=1 且 DIS=0 时才能够配置 ADC 转换开启.

该Bits由硬件自动清零,分为以下几种情况:

1. 单次转换模式下,且选择软件触发模式时（ADC\_CFGR{TRIG\_SEL}=00）,当 ADC\_ISR{EOS} 标志置高时清除 START Bits.
2. 非连续模式下,且选择软件触发模式时,当 ADC\_ISR{EOC} 标志置高时清除 START.
3. 在任何情况下,执行 STOP 命令清除 START（START 和 STOP 同时清除）.

**Bits 1 DIS:**ADC 功能除能控制.

* 写 0:无效
* 写 1:控制 ADC 除能

该Bits仅在 EN=1 且 START=0（没有转换正在执行）时配置写 1 才有效.

**Bits 0 EN:**ADC 功能Enable控制.

* 写 0:无效
* 写 1:控制 ADC Enable,读该Bits为 1 Table示 ADC Enable

软件配置该Enable后,Table示 ADC 可以开始触发转换；该Bits仅在 ADC\_CR Register全为 0 时才能写 1；读该Bits可以反映 ADC 功能Enable状态.软件应当在初始化 ADC 模拟电路后延时至少 100us 以等待电路稳定,然后再Enable ADC 功能.

###### 16.14.2 ADC\_CFGR

Offset:0x04

Reset Value:0x00000002

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| **31-24** | **23** | | **22** | **21-20** | **19** |  | **18-17** |
| RESERVED | RESERVED | | WAIT\_MODE | CONV\_MODE | OVERRUN\_MO | DE | TRIG\_SEL |
| r-0h | r-0h | | r-0h | r-0h | r-0h |  | rw-0h |
| **16** | | **15-13** | | **12** | **11-8** | **7-0** | |
| EXT\_TRIG\_SEL[3] | | EXT\_TRIG\_SEL[2:0] | | DMA\_EN | CLK\_DIV[11:8] | CLK\_DIV[7:0] | |
| rw-0h | | rw-0h | | rw-0h | rw-0h | rw-2h | |

**Bits 31-23 RESERVED:**保留,不可更改.**Bits 22 WAIT\_MODE:**等待转换模式控制.

* 0:等待转换Disable
* 1:等待转换Enable

等待转换模式,即 ADC\_DR 数据被读走或者 ADC\_ISR{EOC} 标志被清除后才能接收新的触发Request ,可以防止 overrun,但可能 bypass 触发Request .

仅在 ADC\_CR{START} 为 0 时该Bits才能够配置.**Bits 21-20 CONV\_MODE:**ADC 转换模式选择.

* 00:单次转换模式
* 01:连续转换模式
* 1x:非连续转换模式

仅在 ADC\_CR{START} 为 0 时该Bits才能够配置.

**说明:**

1. 单次转换模式,每次触发完成 *ADC\_SEQR0/1* 整个采样序列后即停止,等待下一次触发.
2. 连续转换模式,触发开始转换后,一直按照 *ADC\_SEQR0/1* 的采样序列循环采样,直到配置

*ADC\_CR{STOP}*.

1. 非连续转换模式,每次触发完成一次*ADC*采样（按照*ADC\_SEQR0/1* 的采样序列）即停止,等待下一次触发.

**Bits 19 OVERRUN\_MODE:**控制 overrun 时的数据操作.

* 0:当 overrun 发生时,原 ADC\_DR 中的数据被保留
* 1:当 overrun 发生时,ADC\_DR 被新的转换数据覆盖仅在 ADC\_CR{START} 为 0 时该Bits才能够配置.

**Bits 18-17 TRIG\_SEL:**触发源模式和触发极性选择.

* 00:软件触发,ADC\_CR{START}上升沿决定转换立即开始
* 01:硬件触发,上升沿触发
* 10:硬件触发,下降沿触发
* 11:硬件触发,上升沿和下降沿均可触发仅在 ADC\_CR{START}为 0 时该Bits才能够配置.

使用硬件触发时,配置 ADC\_CR{START} 后,需要等待 3 拍 ADCCLK 后才可以接收触发信号.

**Bits 16-13 EXT\_TRIG\_SEL:**ADC 转换开始外部触发源选择.

* 0000~0100:Reserved
* 0101:GPIO47
* 0110:GPIO31
* 0111:GPIO19
* 1000:GPIO10
* 1001:GPTIM1\_TRGO
* 1010:GPTIM0\_CH2\_OUT
* 1011:GPTIM3\_TRGO
* 1100:GPTIM0\_CH3\_OUT
* 1101:GPTIM0\_TRGO
* 1110:GPTIM2\_CH1\_OUT
* 1111:Reserved

Notice:

1. 仅在 *ADC\_CR{START}* 为*0*时该Bits才能够配置.
2. 若使用*GPTIMx* 的*TRGO* 信号作为触发,则 *GPTIMx\_CR2{MMS}* 仅可以配置为*0x100(OC0REF), 0x101(OC1REF), 0x110(OC2REF), 0x111(OC3REF)*.对于 *GPTIM2* 和 *GPTIM3*,仅可以配置为 *0x100(OC0REF), 0x101(OC1REF)*.
3. 若要实现定时触发或周期触发,需配置所选通道为输出模式,选择相应的输出模式,并根据所需时间配置相应的 *GPTIMx\_ARR* 和 *GPTIMx\_CCRx*.

**Bits 12 DMA\_EN:**DMA 功能Enable.

* 0:DMA Disable
* 1:DMA Enable**Bits 11-0 CLK\_DIV:**ADCCLK 的clock 预分频选择.
* 000:不分频
* 001:不分频
* n:ADC\_IP\_CLK=ADCCLK/n,占空比 50%.

Notice:

1. 仅在*ADC\_CR*均为*0*时,该Bits配置才有效；*ADCCLK*的clock 源选择在*RCC\_CR2*中配置.
2. clock 分频和clock 源选择需要考虑数据的读出速度,*ADC* 每*16* 拍完成一次采样,若该*ADC* clock 配置过快,软件或*DMA*不能及时读走,则可能造成溢出.

###### 16.14.3 ADC\_SEQR0

Offset:0x08

Reset Value:0x00000000

**Notice:** 仅在 *ADC\_CR{START}* 和 *ADC\_CR{EN}* 为*0*时,才能够配置该Register.

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| **31-28** | **27-24** | **23-20** | **19-16** | **15-12** | **11-8** | **7-4** | **3-0** |
| SEL7 | SEL6 | SEL5 | SEL4 | SEL3 | SEL2 | SEL1 | SEL0 |
| rw-0h | rw-0h | rw-0h | rw-0h | rw-0h | rw-0h | rw-0h | rw-0h |

**Bits 31-28 SEL7:**ADC 采样序列第 7 个通道配置.

配置值为采样通道号 1~15,若发现配置为 0,则当前序列结束；若出现 SELx 的通道号相同,则执行多次重复采样.

差分输入只配置正端的通道号即可,负端通道号硬件根据 ADC\_DIFFSEL 自动选择.

**Bits 27-24 SEL6:**ADC 采样序列第 6 个通道配置.

配置值为采样通道号 1~15,若发现配置为 0,则当前序列结束；若出现 SELx 的通道号相同,则执行多次重复采样.

差分输入只配置正端的通道号即可,负端通道号硬件根据 ADC\_DIFFSEL 自动选择.

**Bits 23-20 SEL5:**ADC 采样序列第 5 个通道配置.

配置值为采样通道号 1~15,若发现配置为 0,则当前序列结束；若出现 SELx 的通道号相同,则执行多次重复采样.

差分输入只配置正端的通道号即可,负端通道号硬件根据 ADC\_DIFFSEL 自动选择.

**Bits 19-16 SEL4:**ADC 采样序列第 4 个通道配置.

配置值为采样通道号 1~15,若发现配置为 0,则当前序列结束；若出现 SELx 的通道号相同,则执行多次重复采样.

差分输入只配置正端的通道号即可,负端通道号硬件根据 ADC\_DIFFSEL 自动选择.

**Bits 15-12 SEL3:**ADC 采样序列第 3 个通道配置.

配置值为采样通道号 1~15,若发现配置为 0,则当前序列结束；若出现 SELx 的通道号相同,则执行多次重复采样.

差分输入只配置正端的通道号即可,负端通道号硬件根据 ADC\_DIFFSEL 自动选择.

**Bits 11-8 SEL2:**ADC 采样序列第 2 个通道配置.

配置值为采样通道号 1~15,若发现配置为 0,则当前序列结束；若出现 SELx 的通道号相同,则执行多次重复采样.

差分输入只配置正端的通道号即可,负端通道号硬件根据 ADC\_DIFFSEL 自动选择.

**Bits 7-4 SEL1:**ADC 采样序列第 1 个通道配置.

配置值为采样通道号 1~15,若发现配置为 0,则当前序列结束；若出现 SELx 的通道号相同,则执行多次重复采样.

差分输入只配置正端的通道号即可,负端通道号硬件根据 ADC\_DIFFSEL 自动选择.

**Bits 3-0 SEL0:**ADC 采样序列第 0 个通道配置.

配置值为采样通道号 1~15,若发现配置为 0,则当前序列结束；若出现 SELx 的通道号相同,则执行多次重复采样.

差分输入只配置正端的通道号即可,负端通道号硬件根据 ADC\_DIFFSEL 自动选择.

###### 16.14.4 ADC\_SEQR1

Offset:0x0C

Reset Value:0x00000000

**Notice:** 仅在 *ADC\_CR{START}* 和 *ADC\_CR{EN}* 为*0*时,才能够配置该Register.

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| **31-28** | **27-24** | **23-20** | **19-16** | **15-12** | **11-8** | **7-4** | **3-0** |
| SEL15 | SEL14 | SEL13 | SEL12 | SEL11 | SEL10 | SEL9 | SEL8 |
| rw-0h | rw-0h | rw-0h | rw-0h | rw-0h | rw-0h | rw-0h | rw-0h |

**Bits 31-28 SEL15:**ADC 采样序列第 15 个通道配置.

配置值为采样通道号 1~15,若发现配置为 0,则当前序列结束；若出现 SELx 的通道号相同,则执行多次重复采样.

差分输入只配置正端的通道号即可,负端通道号硬件根据 ADC\_DIFFSEL 自动选择.

**Bits 27-24 SEL14:**ADC 采样序列第 14 个通道配置.

配置值为采样通道号 1~15,若发现配置为 0,则当前序列结束；若出现 SELx 的通道号相同,则执行多次重复采样.

差分输入只配置正端的通道号即可,负端通道号硬件根据 ADC\_DIFFSEL 自动选择.

**Bits 23-20 SEL13:**ADC 采样序列第 13 个通道配置.

配置值为采样通道号 1~15,若发现配置为 0,则当前序列结束；若出现 SELx 的通道号相同,则执行多次重复采样.

差分输入只配置正端的通道号即可,负端通道号硬件根据 ADC\_DIFFSEL 自动选择.

**Bits 19-16 SEL12:**ADC 采样序列第 12 个通道配置.

配置值为采样通道号 1~15,若发现配置为 0,则当前序列结束；若出现 SELx 的通道号相同,则执行多次重复采样.

差分输入只配置正端的通道号即可,负端通道号硬件根据 ADC\_DIFFSEL 自动选择.

**Bits 15-12 SEL11:**ADC 采样序列第 11 个通道配置.

配置值为采样通道号 1~15,若发现配置为 0,则当前序列结束；若出现 SELx 的通道号相同,则执行多次重复采样.

差分输入只配置正端的通道号即可,负端通道号硬件根据 ADC\_DIFFSEL 自动选择.

**Bits 11-8 SEL10:**ADC 采样序列第 10 个通道配置.

配置值为采样通道号 1~15,若发现配置为 0,则当前序列结束；若出现 SELx 的通道号相同,则执行多次重复采样.

差分输入只配置正端的通道号即可,负端通道号硬件根据 ADC\_DIFFSEL 自动选择.

**Bits 7-4 SEL9:**ADC 采样序列第 9 个通道配置.

配置值为采样通道号 1~15,若发现配置为 0,则当前序列结束；若出现 SELx 的通道号相同,则执行多次重复采样.

差分输入只配置正端的通道号即可,负端通道号硬件根据 ADC\_DIFFSEL 自动选择.

**Bits 3-0 SEL8:**ADC 采样序列第 8 个通道配置.

配置值为采样通道号 1~15,若发现配置为 0,则当前序列结束；若出现 SELx 的通道号相同,则执行多次重复采样.

差分输入只配置正端的通道号即可,负端通道号硬件根据 ADC\_DIFFSEL 自动选择.

###### 16.14.5 ADC\_DIFFSEL

Offset:0x10

Reset Value:0x00000000 Notice: 仅在ADC\_CR{START}和ADC\_CR{EN}为0时,该Register才能够配置.

|  |  |  |  |
| --- | --- | --- | --- |
| **31-16** | **15-9** | **8-1** | **0** |
| RESERVED | SEL1 | SEL0 | RESERVED |
| r-0h | r-0h | rw-0h | r-0h |

**Bits 31-16 RESERVED:**保留,不可更改.

**Bits 15-9 SEL1:**ADC 通道 9~15 为内部通道.

只支持单端模式,不支持差分模式,这些Bits只读.

**Bits 8-1 SEL0:**ADC 通道 1~8 差分/单端模式选择.

每一Bits控制一个通道:

* 0:通道 x 为单端模式
* 1:通道 x 为差分模式

差分通道仅支持两个外部相邻通道之间,如通道 2 和通道 3,则该Register相应的两个控制Bits要配置为 1.

**Bits 0 RESERVED:**保留,不可更改.

###### 16.14.6 ADC\_ISR

Offset:0x14

Reset Value:0x00000000

**Notice:** 软件Enable *ADC\_CR{START}* 前建议先清除该Register.

|  |  |  |  |
| --- | --- | --- | --- |
| **31-3** | **2** | **1** | **0** |
| RESERVED | OVERRUN | EOS | EOC |
| r-0h | rw1c-0h | rw1c-0h | rw1c-0h |

**Bits 31-3 RESERVED:**保留,不可更改.

**Bits 2 OVERRUN:**ADC 转换 overrun 标志.

* 0:没有 overrun 发生
* 1:发生 overrun

当 EOC 标志为高时（ADC\_DR 数据未取走或未配置软件写 1 清零）,新的转换完成,该Bits由硬件置 1.

软件写 1 清零.

**Bits 1 EOS:**ADC 通道序列采样完成标志.

* 0:通道序列转换未完成
* 1:通道序列转换完成当 ADC\_SEQR0/1 中的整个通道序列完成一轮转换后,该Bits由硬件置 1.

软件写 1 清零.

**Bits 0 EOC:**ADC 转换完成标志.

* 0:通道转换未完成
* 1:通道转换完成某个通道 ADC 转换结束,新的转换数据写入到 ADC\_DR 中后该标志由硬件置高.

软件写 1 清零或者读 ADC\_DR 后清零.

###### 16.14.7 ADC\_IER

Offset:0x18 Reset Value:0x00000000

|  |  |  |  |
| --- | --- | --- | --- |
| **31-3** | **2** | **1** | **0** |
| RESERVED | OVERRUN\_INT\_EN | EOS\_INT\_EN | EOC\_INT\_EN |
| r-0h | rw-0h | rw-0h | rw-0h |

**Bits 31-3 RESERVED:**保留,不可更改.

**Bits 2 OVERRUN\_INT\_EN:**ADC 转换 overrun interruptEnable.

* 0:Disable overrun interrupt
* 1:Enable overrun interrupt**Bits 1 EOS\_INT\_EN:**ADC 通道序列采样完成interruptEnable.
* 0:Disable通道序列采样完成interrupt
* 1:Enable通道序列采样完成interrupt**Bits 0 EOC\_INT\_EN:**ADC 转换完成interruptEnable.
* 0:Disable转换完成interrupt
* 1:Enable转换完成interrupt

###### 16.14.8 ADC\_DR

Offset:0x1C

Reset Value:0x00000000

|  |  |
| --- | --- |
| **31-12** | **11-0** |
| RESERVED | DATA |
| r-0h | r-0h |

**Bits 31-12 RESERVED:**保留,不可更改.

**Bits 11-0 DATA:**ADC 转换数据.差分模式下,第 11 Bits为符号Bits.

## 17. 基本定时器 (BSTIM)

### 17.1 Introduction

BSTIMER（Basic Timer）包含 16bits counter,支持自动重装载功能,且支持最多 16bits 可编程的分频counter.有两个 BSTIMER,分别为 BSTIMER0 和 BSTIMER1.

### 17.2 Main features

BSTIMER 包括如下功能:

* 16bits counter,加法计数,支持自动重加载
* 分频counter
* DMA 控制
* 支持单脉冲
* 支持主模式功能
* 更新事件管理
* Debug 模式控制
* interrupt信号产生 BSTIMER 的框Figure如下:



**u**

**\_**

**reg**

**\_**

**model**



**Master**

**C**

**ontrol**



**u**

**\_**

**psc**

**\_**

**counter**



**I**

**nterrupt**

**Control**



**DMA**

**C**

**ontrol**

**dma**

**\_**

**ack**

**dma**

**\_**

**req**

**apb**

**\_**

**write**

**apb**

**\_**

**read**

**interrupt**

**trigger**

**\_**

**output**

**rcc**

**\_**

**tim**

**\_**

**clk**



**u**

**\_**

**counter**

**reset**

**,**

**enable**

**,**

**update**



**ARR**

**\_**

**shadow**

**Figure 17-1 BSTIMER 框Figure**

####  rcc\_tim\_clk:BSTIMER 的接口clock  apb\_read:APB 总线的读

* **dma\_ack:**DMA 回复的 ACK  **trigger\_output:**BSTIMER 的 TRGO 输出
* **dma\_req:**BSTIMER 对 DMA 的Request  **interrupt:**BSTIMER 的interrupt
* **apb\_write:**APB 总线的写

### 17.3 接口clock

BSTIMER 接口clock 源为 PCLK,不能设置为其它clock 源.clock Enable和复Bits配置可以参考 RCC 章节.

### 17.4 counter

counter仅支持向上计数,计数到 ARR,这样counter的值会从 ARR 变为 0,然后继续计数,同时状态标记Bits UIF 置Bits,如果更新事件interruptRequest Enable即 UIE 置Bits,则也会产生interrupt,此时Table示一个计数周期完成.下个计数周期counter继续从 0 开始计数,如此循环往复.

#### 17.5 自动重加载

可软件配置Register BSTIM\_CR1 的 ARPE Bits来设置是否启用 ARR 影子Register,如果 ARPE=0,则禁用影子Register,软件写入的值直接同步更新到 ARR 供counter使用,如果 ARPE=1,则软件写入的值不会立即生效,直到更新事件到来,才会将该值更新到 ARR 供counter使用.

#### 17.6 分频counter

BSTIMER 支持 16-bit（1~65535）可编程分频,此功能通过分频counter BSTIM\_PSC 实现.接口clock 作为分频counter的clock ,Register BSTIM\_CR1 的 CEN 作为分频counter的计数Enable,当分频counter计数到预先加载的分频值后,输出一个脉冲,作为下一级counter的计数Enable,然后分频counter归零重新计数,依次类推.

分频counter的分频值默认启用影子Register,即软件的写操作不会立即生效,而是直到更新事件（UG 事件置Bits、计数溢出）到来,才会将新的分频值写入影子Register,此时该分频值才正式生效.软件读操作读取的是写入的Register值,而不是影子Register,如果在更新事件到来前有多次写操作,则会覆盖之前写入的值.计数和分频波形如下:



**Figure 17-2 计数和分频波形**

### 17.7 DMA 控制

BSTIMER 支持 DMA 功能,Enable DMA 功能后,其所有Register除 BSTIM\_SR、BSTIM\_EGR 外与 memory 之间可以相互传递数据,BSTIM\_SR 只能被读取数据,BSTIM\_EGR 只能被写入数据.通过Register BSTIM\_DIER 的 UDE BitsEnable DMA,当有更新事件时则会产生 DMA Request ,

DMA 返回的 ACK 信号会清除模块的 DMA Request 信号.

#### 17.8 支持单脉冲

BSTIMER 支持单脉冲计数模式,通过置BitsRegister BSTIM\_CR1 的 OPM BitsEnable该模式,在该模式下,当counter计数到 ARR 值后会归零并停止计数（CEN 硬件自动清零）,除非再次初始化才会重新计数,如下Figure所示:

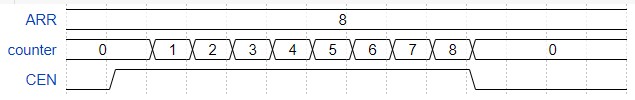
**Figure**

**17**

**-**

**3**

单脉冲波形



#### 17.9 支持主模式选择

BSTIMER 可以与其他内部模块级联,并作为主机使用,通过产生触发输出信号（TRGO）来控制 DAC.可通过软件配置 BSTIM\_CR2 Register的 MMS Bits来选择 TRGO 信号的来源,具体如下:

* MMS=3’b000:复Bits模式,此时 UG 标志Bits将作为 TRGO 信号输出给外部从机.
* MMS=3’b001:Enable模式,此时counter的计数Enable CEN 将作为 TRGO 信号输出给外部从机.
* MMS=3’b010:更新模式,此时将更新事件作为 TRGO 信号输出给外部从机.
* MMS 的其它值为保留值.

#### 17.10 更新事件管理

更新事件主要有以下事件源:

1. counter的溢出事件（overflow）,即counter的值从 ARR 变为 0.
2. UG 置Bits（软件置Bits）,即配置Register BSTIM\_EGR 的 UG Bits.

与更新事件管理相关的控制信号主要是Register BSTIM\_CR1 的 URS 和 UDIS,具体控制如下:

* 若 UDIS=0,URS=0,则 overflow、UG 置Bits会初始化counter和分频counter,如果启用影子Register,更新事件则会把写入的值更新到影子Register中（ARR 取决于 ARPE）,

UIF 会置Bits,如果Enable了interrupt或 DMA,则会产生interrupt或 DMA Request .

* 若 UDIS=0,URS=1,则 overflow、UG 置Bits会初始化counter和分频counter,如果启用影子Register,更新事件将会把写入的值更新到影子Register中（ARR 取决于 ARPE）,

UIF 只会在 overflow 情况下置Bits,如果Enable了interrupt或 DMA,则会产生interrupt或 DMA Request .

* 若 UDIS=1（忽略 URS）,则只有 UG 置Bits仍会初始化counter和分频counter,但是影子Register不会被更新,且 UIF 不会置Bits,因此也不会产生相应interrupt或 DMA Request .

### 17.11 Debug 模式控制

BSTIMER 可由软件配置 debug 下是否停止计数,通过 SYSCFG 的 CR2 Register来实现 BSTIMER0 和 BSTIMER1 的 DEBUG 模式计数控制,如果Enable该功能,则进入系统 debug 模式时,BSTIMER 停止计数（counter不会被初始化）.

### 17.12 interrupt信号

BSTIMER 的interrupt信号如下:

**Table 17-1 BSTIMER interrupt信号**

|  |  |
| --- | --- |
| interrupt名称 | Description |
| 更新事件interrupt | counter溢出、UG 置Bits均可以产生更新事件interrupt |

上述interrupt的Enable通过配置Register BSTIM\_DIER 的 UIE Bits实现,interrupt状态可以通过Register

BSTIM\_SR 获得.

### 17.13 BSTIMER 相关RegisterDescription

BSTIMER0 基地址:0x4000C000

BSTIMER1 基地址:0x4001C000

**Table 17-2 BSTIMER Register Summary**

|  |  |  |
| --- | --- | --- |
| Register | Offset | Description |
| BSTIM\_CR1 | 0x00 | Control Register 1 |
| BSTIM\_CR2 | 0x04 | Control Register 2 |
| BSTIM\_DIER | 0x0c | DMA/interruptEnableRegister |
| BSTIM\_SR | 0x10 | 状态Register |
| BSTIM\_EGR | 0x14 | 事件Register |
| BSTIM\_CNT | 0x24 | counterRegister |
| BSTIM\_PSC | 0x28 | counter分频值 |
| BSTIM\_ARR | 0x2c | counter重装载值 |

#### 17.13.1 BSTIM\_CR1

Offset:0x00

Reset Value:0x00000000

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| **31-8** | **7** | **6-4** | **3** | **2** | **1** | **0** |
| RESERVED | ARPE | RESERVED | OPM | URS | UDIS | CEN |
| rw-0h | rw-0h | rw-0h | rw-0h | rw-0h | rw-0h | rw-0h |

**Bits 31-8 RESERVED:**保留.

**Bits 7 ARPE:**重装载影子RegisterEnable.

* 0:BSTIM\_ARR 影子Register除能
* 1:BSTIM\_ARR 影子RegisterEnable**Bits 6-4 RESERVED:**保留.

**Bits 3 OPM:**单脉冲模式Enable.

* 0:单脉冲模式除能
* 1:单脉冲模式Enable,counter在下一次更新事件停止计数

**Bits 2 URS:**更新事件源选择,该Bits仅影响interrupt（UIF）和 DMA 标志Bits,不影响内部逻辑.

* 0:counter溢出、UG Bits置Bits,均可以置Bits UIF
* 1:只有counter溢出事件可以置Bits UIF **Bits 1 UDIS:**更新事件除能.
* 0:更新事件Enable,可以产生更新事件.
* 1:更新事件除能,影子Register和 UIF 均不会被更新,但是此时counter和分频counter仍可以被 UG 置Bits事件初始化.

**Bits 0 CEN:**counterEnable,单脉冲模式下 CEN 由硬件清零.

* 0:counter除能
* 1:counterEnable

#### 17.13.2 BSTIM\_CR2

Offset:0x00 Reset Value:0x00000000

|  |  |  |
| --- | --- | --- |
| **31-7** | **6-4** | **3-0** |
| RESERVED | MMS | RESERVED |
| rw-0h | rw-0h | rw-0h |

**Bits 31-7 RESERVED:**保留.

**Bits 6-4 MMS:**主模式选择,可以配置 TRGO 输出.

* 000:复Bits模式,UG 将作为 TRGO 信号输出
* 001:Enable模式,CEN 将作为 TRGO 信号输出
* 010:更新模式,更新事件（内部信号）将作为 TRGO 信号输出
* 其它值:保留**Bits 3-0 RESERVED:**保留.

#### 17.13.3 BSTIM\_DIER

Offset:0x0c Reset Value:0x00000000

|  |  |  |  |
| --- | --- | --- | --- |
| **31-9** | **8** | **7-1** | **0** |
| RESERVED | UDE | RESERVED | UIE |
| rw-0h | rw-0h | rw-0h | rw-0h |

**Bits 31-9 RESERVED:**保留.**Bits 8 UDE:**更新事件 DMA Request Enable.

* 0:禁用更新事件 DMA Request
* 1:Enable更新事件 DMA Request **Bits 7-1 RESERVED:**保留.**Bits 0 UIE:**更新事件interruptRequest Enable.
* 0:禁用更新事件interruptRequest
* 1:Enable更新事件interruptRequest

#### 17.13.4 BSTIM\_SR

Offset:0x10 Reset Value:0x00000000

|  |  |
| --- | --- |
| **31-1** | **0** |
| RESERVED | UIF |
| r-0h | r-0h |

**Bits 31-1 RESERVED:**保留.**Bits 0 UIF:**更新事件标志.

* 0:无事件
* 1:更新事件发生

#### 17.13.5 BSTIM\_EGR

Offset:0x14 Reset Value:0x00000000

|  |  |
| --- | --- |
| **31-1** | **0** |
| RESERVED | UG |
| w-0h | w-0h |

**Bits 31-1 RESERVED:**保留.**Bits 0 UG:**更新事件产生.

* 0:无动作
* 1:产生一次更新事件

#### 17.13.6 BSTIM\_CNT

Offset:0x24 Reset Value:0x00000000

|  |  |
| --- | --- |
| **31-16** | **15-0** |
| RESERVED | CNT |
| rw-0h | rw-0h |

**Bits 31-16 RESERVED:**保留.**Bits 15-0 CNT:**counter计数值.

#### 17.13.7 BSTIM\_PSC

Offset:0x28 Reset Value:0x00000000

|  |  |
| --- | --- |
| **31-16** | **15-0** |
| RESERVED | PSC |
| rw-0h | rw-0h |

**Bits 31-16 RESERVED:**保留.

**Bits 15-0 PSC:**clock 分频值为 PSC+1.

#### 17.13.8 BSTIM\_ARR

Offset:0x2c

Reset Value:0x0000ffff

|  |  |
| --- | --- |
| **31-16** | **15-0** |
| RESERVED | ARR |
| rw-0h | rw-ffffh |

**Bits 31-16 RESERVED:**保留.

**Bits 15-0 ARR:**counter重装载值.

## 18. 实时clock (RTC)

### 18.1 Introduction

RTC（Real Time Clock）是一个独立的BCD计时器/counter,有两个32BitsRegister包含秒、分、小时（12 或 24 小时格式）、星期、日期（月的日期）、月和年,用二进制编码的十进制格式

Table示（BCD）,还有一个 32 BitsRegisterTable示亚秒值.RTC 支持在低功耗模式运行.

### 18.2 Main features

RTC 包括如下功能:

* 日历计数功能,采用 BCD 格式,支持秒、分、小时、日、星期、月、年、
* 支持 ppm 调整,调整步长 0.5ppm,支持+/-1024 ppm 调整
* 支持低功耗唤醒
* tamper/wakeup IO 检测功能,支持有效电平选择,滤波拍数可配置
* 周期计数功能,32 Bitscounter
* 闹钟功能,支持两个闹钟,支持 Mask 选择与日历匹配
* tamper/wakeup 报警清除 retention sram 功能
* 内部信号 IO 输出,包括 alarm0 匹配脉冲,alarm1 匹配脉冲,周期计数匹配脉冲,秒信号输出
* 支持日历计数值读取
* 支持 sub-second 计数值读取
* 支持周期计数的计数值读取
* Interrupt signal generation

### 18.3 接口clock

RTC 接口clock 源有 XO32K 和 RCO32K,XO32K 的clock 精度一般比 RCO32K 要高.

clock 配置和选择可以参考 *RCC* 章节.

### 18.4 日历

RTC 日历时间和日期可以通过两种类型的Register访问获取,一种是异步的,一种是同步的,两种Register分别如下:

* **异步Register** RTC\_SYNDATA 和 RTC\_SYNDATA\_H,RTC\_SYNDATA Table示时分秒, RTC\_SYNDATA\_H Table示年月日星期.
* **同步Register** RTC\_CALENDAR\_R 和 RTC\_CALENDAR\_R\_H,RTC\_CALENDAR\_R Table示时分秒,RTC\_CALENDAR\_R\_H Table示年月日星期.

#### 18.4.1 读取日历

只介绍通过同步Register读取日历,由于采用打拍同步,需要多次读到同一值,才可以使用,不能仅读取一次.读取顺序如下:

1. 先读取 RTC\_SUB\_SECOND Register的值,获取 SUBSECOND\_COUNT 拍数.
2. 连续两次读取 RTC\_CALENDAR\_R 的值,如果两次的值不等,则继续读取,直到连续两次的值相等.
3. 连续两次读取 RTC\_CALENDAR\_R\_H 的值,如果两次的值不等,则继续读取,直到连续两次的值相等.
4. 最后再读一次 RTC\_SUB\_SECOND Register的值,如果与步骤(1) 的值不一致,则重新从步骤(1) 开始读取Register的值.
5. 由于 SUBSECOND\_COUNT 从最大值变为 0 时,Register RTC\_CALENDAR\_R 或 RTC\_CALENDAR\_R\_H 的值可能没有产生变化,因此如果 SUBSECOND\_COUNT 为 0 时,则继续重新从步骤(1) 开始读取Register的值,如果 SUBSECOND\_COUNT 不为 0,则完整的日历时间读取完成.

SUBSECOND\_COUNT 拍数转换为微秒 sub-second,先要通过 RCC 获知 RTC 的接口clock fRTCCLK,则计算 sub-second 的公式如下:

***sub-second=(1000000\*SUBSECOND\_COUNT)/fRTCCLK***

#### 18.4.2 设置日历

设置日历需要配置两个 RTC Register:RTC\_CALENDAR\_H 和 RTC\_CALENDAR.RTC\_ CALENDAR\_H 为年月日星期信息,RTC\_CALENDAR 为时分秒信息,RTC\_SUB\_SECOND Register不能配置,所以 sub-second 不能配置.日历配置顺序如下:

1. 读 RTC\_SR1 Register,等待Register中全部的 WRITE\_XXX\_DONE Bits为 1 后,即 RTC\_SR1 Register的 bit1 至 bit11 全为 1,才allowed对Register进行写操作.
2. 配置 RTC\_CALENDAR\_H Register,即配置年月日星期.
3. 读 RTC\_SR1 Register,等待Register中全部的 WRITE\_XXX\_DONE Bits为 1 后,即 RTC\_SR1 Register的 bit1 至 bit11 全为 1,才allowed对Register进行写操作.
4. 配置 RTC\_CALENDAR Register,即配置时分秒.

### 18.5 RTC PPM 调整

RTC 频率的校准分辨率约为 0.5ppm 范围从 -1024ppm 到 +1024ppm,为 0 时Table示不调整,通过配置RegisterRTC\_PPMADJUST 的值来设置 PPM 的调整值,0 ppm Correspond to RTC\_PPMADJUST Register的值应该为 0x7FFF.设置 PPM 调整的流程如下:

1. 读 RTC\_SR1 Register,等待Register中全部的 WRITE\_XXX\_DONE Bits为 1 后,即 RTC\_SR1 Register的 bit1 至 bit11 全为 1,才allowed对Register进行写操作.
2. 配置 RTC\_PPMADJUST Register的值.

#### 18.6 低功耗唤醒

RTC 可以通过interrupt或唤醒信号将 MCU 从 Sleep、Stop、Standby 中唤醒.

**Table 18-1 RTC Wakeup Source**

|  |  |
| --- | --- |
| 休眠模式 | 唤醒Description |
| Sleep | RTC 所有interrupt可以使设备从 sleep 模式中唤醒 |
| Stop0、Stop1、Stop2、  Stop3 | Wakeup/tamper IO、alarm、cyc 的唤醒信号可以使设备从 stop 模式中唤醒 |
| Standby | Wakeup/tamper IO、alarm、cyc 的唤醒信号可以使设备从 standby 模式中唤醒 |

设置 wakeup/tamper IO、alarm、cyc 的唤醒信号是通过Register RTC\_CR,Correspond to Register中的 bit 名称分别如下Table:

##### Table 18-2 配置唤醒信号Enable的 bit 信息

|  |  |
| --- | --- |
| 功能 | RTC\_CRRegisterbit信息 |
| WAKEUP\_IO0 | WAKEUP0\_WKEN1 |
| WAKEUP\_IO1 | WAKEUP1\_WKEN1 |
| WAKEUP\_IO2 | WAKEUP2\_WKEN1 |
| TAMPER | TAMPER\_WKEN1 |
| ALARM0 | RTC\_ALARM0\_WKEN |
| ALARM1 | RTC\_ALARM1\_WKEN |
| CYC | CYC\_WKEN |

### 18.7 tamper/wakeup IO 检测

tamper/wakeup IO 的输入事件可以配置为边沿检测,也可以配置为电平检测,电平检测时可以配置滤波.边沿检测Table示检测 GPIO 的上升沿或下降沿,电平检测Table示检测 GPIO 的高电平或低电平,如果高电平有效,GPIO 输入高电平则会检测到输入事件,如果低电平有效,GPIO输入低电平则会检测到输入事件.当检测到输入事件可以配置产生如下操作:

* 清除保留 SRAM 的内容
* 产生一个interrupt,并且能够从 sleep 模式中唤醒
* 产生唤醒信号,能够从 stop、standby 模式中唤醒

#### 18.7.1 初始化和配置流程

在 tamper/wakeup 初始化之前需要配置Correspond to 的 GPIO 为 tamper/wakeup 功能.如果是电平检测,需要配置 GPIO 的上下拉,高电平有效则配置下拉,低电平有效则配置上拉.以 tamper为例,其初始化和配置流程如下:

1. 如果是电平检测,则通过RTC\_CRRegister的TAMPER\_FILTER\_CFGBits配置滤波拍数,并且配置有效电平即低电平有效还是高电平有效,此 bit Bits为 RTC\_CR Register的 TAMPER\_LEVEL\_SEL,最后Enable电平唤醒,即配置 RTC\_CR Register的 TAMPER\_

WKEN0.**如果是边沿检测,则忽略此步骤.**

1. 如果需要从 stop、standby 模式中唤醒,则配置唤醒Enable,即配置 RTC\_CR Register的

TAMPER\_WKEN1,**否则忽略此步骤**.

1. 置Bits RTC\_CR Register的 TAMPER\_EN 来Enable tamper 功能.

##### 18.7.2 清除保留 SRAM

当检测到 tamper/wakeup IO 的输入事件时可以配置清除保留 SRAM 的内容,置Bits RTC\_CR2

Register的RTC\_RET\_SRAM\_ERASE\_EN的Correspond to 的bitBits可以配置该功能.bit0Correspond to wakeupio0, bit1 Correspond to wakeupio1,bit2 Correspond to wakeupio2,bit3 Correspond to tamper.

#### 18.8 周期计数功能

周期计数功能可以定时产生interrupt或唤醒事件.定时时间通过配置Register RTC\_CYC\_MAX\_ VALUE 的值进行设置,计算定时时间 time 先要通过 RCC 获知 RTC 的接口clock fRTCCLK,Register RTC\_CYC\_MAX\_VALUE 的值为 CYC\_MAX\_VALUE,则计算 time 的公式如下,单Bits为微秒:

##### *Regular interval= (1000000 \* CYC\_MAX\_VALUE) / fRTCCLK*

周期计数过程中可以读取已经经过的拍数,就可以得到本轮计数开始到当前时刻的时间间隔 interval,已经经过的拍数通过读取 RTC\_CYC\_CNT\_VALUE 的值 CYC\_CNT\_VALUE 获得,计算 interval 的公式如下,单Bits为微秒:

##### *Interval= (1000000 \* CYC\_CNT\_VALUE) / fRTCCLK*

周期计数功能的配置流程如下:

1. 已知 time,根据上述公式计算得到 CYC\_MAX\_VALUE 的值,把此值配置到Register

RTC\_CYC\_MAX\_VALUE 中.

1. 如果需要从 stop、standby 模式中唤醒,则配置唤醒Enable,即配置 RTC\_CR Register的

CYC\_WKEN,否则无需配置.

1. Enable周期计数功能,即置Bits RTC\_CR Register的 CYC\_START\_COUNTER.

#### 18.9 闹钟功能

RTC 提供两个闹钟为闹钟 0 和闹钟 1,两个闹钟都支持 Mask 选择与日历匹配,Mask 配置可以选择匹配 sub-second、秒、分钟、小时、日期、星期,日期和星期匹配只能二选一.

如果RegisterRTC\_ALARMx中的BitBitsALARMx\_WEEK\_SEL为0,则只能选择是否匹配日期,如果 ALARMx\_WEEK\_SEL 为 1,则只能选择是否匹配星期,x Table示 0 或 1.当没有配置秒和 sub-second 匹配而配置了分钟匹配,则闹钟定时时间到了后,在当前一分钟内会以一秒为间隔产生 60 次interrupt或/和唤醒事件,如果Enable了闹钟interrupt则会产生interrupt,如果Enable了闹钟唤醒功能则会产生唤醒事件,当没有配置秒、sub-second 和分钟的匹配而配置了小时匹配,则闹钟定时时间到了后,在当前一小时内会以一秒为间隔产生 3600 次interrupt或/和唤醒事件.

闹 钟 的 秒 、 分 钟 、 小 时 、 日 期 、 星 期 Mask 通 过 配 置 RTC\_ALARMx 寄 存 器 中 的 ALARMx\_MASK 字段实现,sub-second 的 Mask 设置通过配置 RTC\_ALARMx\_SUB Register的 RTC\_ALARMx\_SUB\_MASK 实现,sub-second 的定时值设置通过配置 RTC\_ALARMx\_SUB Register的 RTC\_ALARMx\_SUB\_VALUE 实现,RTC\_ALARMx\_SUB\_VALUE 的值Table示拍数,拍数转换成时间的公式与周期计数的拍数转换成时间的一样,x Table示 0 或 1,分别Correspond to 闹钟 0 和闹钟 1.以闹钟 0 为例Description闹钟配置流程如下:

1. 设置日历.
2. 配置闹钟 0 定时值,配置Register RTC\_ALARM0 的 ALARM0\_VALUE 字段,即配置定时时间的小时、分钟、秒、日期或星期.
3. 配 置 闹 钟 0 的 sub-second 定 时 值 , 即 配 置 寄 存 器 RTC\_ALARM0\_SUB 的

RTC\_ALARM0\_SUB\_VALUE.

1. 配置闹钟 0 的小时、分钟、秒、日期或星期的 Mask.
2. 配置闹钟 0 的 sub-second 的 Mask.
3. 如果需要闹钟的interrupt,则Enableinterrupt,如果需要唤醒事件,则Enable闹钟 0 的唤醒功能,即配置Register RTC\_CR 的 bit Bits RTC\_ALARM0\_WKEN.
4. Enable闹钟 0 功能,通过配置Register RTC\_ALARM0 的 ALARM0\_EN.
5. Enable日历功能,即置BitsRegister RTC\_CR 的 bit Bits RTC\_START\_RTC.

### 18.10 内部信号 IO 输出

内部的能通过 IO 输出的信号包括闹钟 0 脉冲,闹钟 1 脉冲,周期计数脉冲,秒信号输出.闹钟、周期计数脉冲是宽度为一拍的脉冲,alarm 脉冲是在定时时间到的时候输出,周期计数脉冲是在每次计数满一个周期后输出,是周期输出的,秒信号是占空比为 50%的方波,周期为 1 秒.IO 输出的电平可以被取反,配置Register RTC\_CR2 的 RTC\_OUT\_POL Bits为 0 时Table示原电平,为 1 是Table示电平取反.配置Register RTC\_CR2 的 RTC\_OUT\_SEL 进行输出信号的选择.

### 18.11 interrupt

RTC 的interrupt信号如下:

**Table 18-3 RTC interrupt信号**

|  |  |
| --- | --- |
| interrupt名称 | Description |
| 闹钟 0 interrupt | 闹钟 0 定时时间到产生的interrupt |
| 闹钟 1 interrupt | 闹钟 1 定时时间到产生的interrupt |
| 周期计数interrupt | 计数满一个周期时产生的interrupt |
| Tamper interrupt | Tamper 检测到输入事件时产生的interrupt |
| Wakeupio0 interrupt | Wakeupio0 检测到输入事件时产生的interrupt |
| Wakeupio1 interrupt | Wakeupio1 检测到输入事件时产生的interrupt |
| Wakeupio2 interrupt | Wakeupio2 检测到输入事件时产生的interrupt |
| 秒interrupt | 秒信号在每一秒产生的interrupt |

上述interrupt的Enable通过配置Register RTC\_CR1 实现,除秒interrupt的interrupt状态外,其它所有interrupt的interrupt状态都可以通过Register RTC\_SR 获得,秒interrupt的interrupt状态通过Register RTC\_SR1 的

SECOND\_SR Bits获得.

### 18.12 RTC 相关RegisterDescription

基地址:0x4000E000

**Table 18-4 RTC Register Summary**

|  |  |  |
| --- | --- | --- |
| Register | Offset | Description |
| RTC\_CR | 0x00 | Control Register |
| RTC\_ALARM0 | 0x04 | 闹钟 0 Register |
| RTC\_ALARM1 | 0x08 | 闹钟 1 Register |
| RTC\_PPMADJUST | 0x0c | PPM 调整Register |
| RTC\_CALENDAR | 0x10 | 日历配置时分秒Register |
| RTC\_CALENDAR\_H | 0x14 | 日历配置年月日星期Register |
| RTC\_CYC\_MAX\_VALUE | 0x18 | 周期计数值配置Register |
| RTC\_SR | 0x1c | interrupt状态Register |
| RTC\_ASYNDATA | 0x20 | 日历异步读取时分秒Register |
| RTC\_ASYNDATA\_H | 0x24 | 日历异步读取年月日星期Register |
| RTC\_CR1 | 0x28 | interruptEnableRegister |
| RTC\_SR1 | 0x2c | 操作状态Register |
| RTC\_CR2 | 0x30 | Control Register 2 |
| RTC\_SUB\_SECOND | 0x34 | 读取 sub-second Register |
| RTC\_CYC\_CNT\_VALUE | 0x38 | 只读周期计数值Register |
| RTC\_ALARM0\_SUB | 0x3c | 闹钟 0 的 sub-second Register |
| RTC\_ALARM1\_SUB | 0x40 | 闹钟 1 的 sub-second Register |
| RTC\_CALENDAR\_R | 0x44 | 日历同步读取时分秒Register |
| RTC\_CALENDAR\_R\_H | 0x48 | 日历同步读取年月日星期Register |

#### 18.12.1 RTC\_CR

Offset:0x00 Reset Value:0x00000000

|  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| **31-29** |  | **28** | | **27** | | **26** | | **25** | |
| RESERVED |  | RTC\_START\_RT  C | | RTC\_ALARM0\_W  KEN | | RTC\_ALARM1\_W  KEN | | CYC\_WKEN | |
| r-0h |  | rw-0h | | rw-0h | | rw-0h | | rw-0h | |
| **24** |  | **23** | | **22** | | **21** | | **20** | |
| CYC\_START\_C  UNTER | O | TAMPER\_EN | | TAMPER\_LEVEL\_  SEL | | TAMPER\_WKEN0 | | TAMPER\_WKEN1 | |
| rw-0h |  | rw-0h | | rw-0h | | rw-0h | | rw-0h | |
| **19-18** |  | **17** | | **16** | | **15** | | **14** | |
| TAMPER\_FILT  \_CFG | ER | WAKEUP0\_EN | | WAKEUP0\_LEVE  L\_SEL | | WAKEUP0\_WKE  N0 | | WAKEUP0\_WKE  N1 | |
| rw-0h |  | rw-0h | | rw-0h | | rw-0h | | rw-0h | |
| **13-12** |  | **11** | | **10** | | **9** | | **8** | |
| WAKEUP0\_FIL  R\_CFG | TE | WAKEUP1\_EN | | WAKEUP1\_LEVE  L\_SEL | | WAKEUP1\_WKE  N0 | | WAKEUP1\_WKE  N1 | |
| rw-0h |  | rw-0h | | rw-0h | | rw-0h | | rw-0h | |
| **7-6** | **5** | | **4** | | **3** | | **2** | | **1-0** |
| WAKEUP1\_FI  LTER\_CFG | WAKEUP2\_E  N | | WAKEUP2\_LE  VEL\_SEL | | WAKEUP2\_W  KEN0 | | WAKEUP2\_W  KEN1 | | WAKEUP2\_FI  LTER\_CFG |
| rw-0h | rw-0h | | rw-0h | | rw-0h | | rw-0h | | rw-0h |

**Bits 31-29 RESERVED:**保留.

**Bits 28 RTC\_START\_RTC:**RTC 日历Enable控制.

* 0:Disable
* 1:Enable**Bits 27 RTC\_ALARM0\_WKEN:**ALARM0\_SR 唤醒Enable.
* 0:Disable
* 1:Enable**Bits 26 RTC\_ALARM1\_WKEN:**ALARM1\_SR 唤醒Enable.
* 0:Disable
* 1:Enable**Bits 25 CYC\_WKEN:**CYC\_SR 唤醒Enable.
* 0:Disable
* 1:Enable**Bits 24 CYC\_START\_COUNTER:**定时功能Enable控制.
* 0:Disable
* 1:Enable

**Bits 23 TAMPER\_EN:**TAMPER 功能Enable.

* 0:Disable
* 1:Enable**Bits 22 TAMPER\_LEVEL\_SEL:**TAMPER 有效电平选择.
* 0:低电平有效
* 1:高电平有效**Bits 21 TAMPER\_WKEN0:**TAMPER 电平唤醒Enable.
* 0:Disable
* 1:Enable

TAMPER\_EN 为 0 时,仍可用于唤醒.

**Bits 20 TAMPER\_WKEN1:**TAMPER\_SR 唤醒Enable.

* 0:Disable
* 1:Enable**Bits 19-18 TAMPER\_FILTER\_CFG:**TAMPER 的滤波控制.
* 0:不滤波
* 1:滤波 1 拍 rtc 接口clock
* 2:滤波 3 拍 rtc 接口clock
* 3:滤波 7 拍 rtc 接口clock **Bits 17 WAKEUP0\_EN:**WAKEUP0 功能Enable.
* 0:Disable
* 1:Enable**Bits 16 WAKEUP0\_LEVEL\_SEL:**WAKEUP0 有效电平选择.
* 0:低电平有效  1:高电平有效**Bits 15 WAKEUP0\_WKEN0:**WAKEUP0 电平唤醒Enable.
* 0:Disable
* 1:Enable

WAKEUP0\_EN 为 0 时,仍可用于唤醒.

**Bits 14 WAKEUP0\_WKEN1:**WAKEUP0\_SR 唤醒Enable.

* 0:Disable
* 1:Enable**Bits 13-12 WAKEUP0\_FILTER\_CFG:**WAKEUP0 的滤波控制.
* 0:不滤波
* 1:滤波 1 拍 rtc 接口clock
* 2:滤波 3 拍 rtc 接口clock
* 3:滤波 7 拍 rtc 接口clock **Bits 11 WAKEUP1\_EN:**WAKEUP1 功能Enable.
* 0:Disable
* 1:Enable

**Bits 10 WAKEUP1\_LEVEL\_SEL:**WAKEUP1 有效电平选择.

* 0:低电平有效
* 1:高电平有效**Bits 9 WAKEUP1\_WKEN0:**WAKEUP1 电平唤醒Enable.
* 0:Disable
* 1:Enable

WAKEUP1\_EN 为 0 时,仍可用于唤醒.

**Bits 8 WAKEUP1\_WKEN1:**WAKEUP1\_SR 唤醒Enable.

* 0:Disable
* 1:Enable**Bits 7-6 WAKEUP1\_FILTER\_CFG:**WAKEUP1 的滤波控制.
* 0:不滤波
* 1:滤波 1 拍 rtc 接口clock
* 2:滤波 3 拍 rtc 接口clock
* 3:滤波 7 拍 rtc 接口clock **Bits 5 WAKEUP2\_EN:**WAKEUP2 功能Enable.
* 0:Disable
* 1:Enable**Bits 4 WAKEUP2\_LEVEL\_SEL:**WAKEUP2 有效电平选择.
* 0:低电平有效
* 1:高电平有效**Bits 3 WAKEUP2\_WKEN0:**WAKEUP2 电平唤醒Enable.
* 0:Disable
* 1:Enable

WAKEUP2\_EN 为 0 时,仍可用于唤醒.

**Bits 2 WAKEUP2\_WKEN1:**WAKEUP2\_SR 唤醒Enable.

* 0:Disable
* 1:Enable**Bits 1-0 WAKEUP2\_FILTER\_CFG:**WAKEUP2 的滤波控制.
* 0:不滤波
* 1:滤波 1 拍 rtc 接口clock
* 2:滤波 3 拍 rtc 接口clock
* 3:滤波 7 拍 rtc 接口clock

#### 18.12.2 RTC\_ALARM0

Offset:0x04 Reset Value:0x00000000

|  |  |  |  |
| --- | --- | --- | --- |
| **31** | **30** | **29-26** | **25-0** |
| ALARM0\_EN | ALARM0\_WEEK\_SEL | ALARM0\_MASK | ALARM0\_VALUE |
| rw-0h | rw-0h | rw-0h | rw-0h |

**Bits 31 ALARM0\_EN:**闹钟 0 Enable控制.

* 0:Disable
* 1:Enable**Bits 30 ALARM0\_WEEK\_SEL:**闹钟 0 星期与日期选择.
* 0:匹配日期
* 1:匹配星期,星期为日期个Bits的低 3bit **Bits 29-26 ALARM0\_MASK:**闹钟 0 的 Mask 配置.

1. 控制是否匹配秒.
   * 0:匹配
   * 1:不匹配
2. 控制是否匹配分钟.
   * 0:匹配
   * 1:不匹配
3. 控制是否匹配小时.
   * 0:匹配
   * 1:不匹配
4. 控制是否匹配日期或星期.
   * 0:匹配
   * 1:不匹配**Bits 25-0 ALARM0\_VALUE:**闹钟 0 定时值配置.当 rtc 日历与闹钟 0 配置匹配时,可以产生

ALARM0\_SR.

[3:0]:秒个Bits

[6:4]:秒十Bits

[10:7]:分钟个Bits

[13:11]:分钟十Bits

[17:14]:小时个Bits

[19:18]:小时十Bits

[23:20]:日期个Bits,其中 [22:20] 也可代Table星期 [25:24]:日期十Bits

#### 18.12.3 RTC\_ALARM1

Offset:0x08 Reset Value:0x00000000

|  |  |  |  |
| --- | --- | --- | --- |
| **31** | **30** | **29-26** | **25-0** |
| ALARM1\_EN | ALARM1\_WEEK\_SEL | ALARM1\_MASK | ALARM1\_VALUE |
| rw-0h | rw-0h | rw-0h | rw-0h |

**Bits 31 ALARM1\_EN:**闹钟 1 Enable控制.

* 0:Disable
* 1:Enable**Bits 30 ALARM1\_WEEK\_SEL:**闹钟 1 星期与日期选择.
* 0:匹配日期
* 1:匹配星期,星期为日期个Bits的低 3bit **Bits 29-26 ALARM1\_MASK:**闹钟 1 的 Mask 配置.

1. 控制是否匹配秒.
   * 0:匹配
   * 1:不匹配
2. 控制是否匹配分钟.
   * 0:匹配
   * 1:不匹配
3. 控制是否匹配小时.
   * 0:匹配
   * 1:不匹配
4. 控制是否匹配日期或星期.
   * 0:匹配
   * 1:不匹配**Bits 25:0 ALARM1\_VALUE:**闹钟 1 定时值配置.当 rtc 日历与闹钟 1 配置匹配时,可以产生

ALARM1\_SR.

[3:0]:秒个Bits

[6:4]:秒十Bits

[10:7]:分钟个Bits

[13:11]:分钟十Bits

[17:14]:小时个Bits

[19:18]:小时十Bits

[23:20]:日期个Bits,其中 [22:20] 也可代Table星期 [25:24]:日期十Bits

#### 18.12.4 RTC\_PPMADJUST

Offset:0x0c

Reset Value:0x00007fff

|  |  |
| --- | --- |
| **31-16** | **15-0** |
| RESERVED | PPMADJUST\_VALUE |
| r-0h | rw-7fffh |

**Bits 31-16 RESERVED:**保留.

**Bits 15-0 PPMADJUST\_VALUE:**32k clock 频率 ppm 调整配置.1 代Table 0.5ppm.可配置范围为 77ff 到 87ff.

* 77ff:调整+1024 ppm  7800:调整+1023.5 ppm
* ...
* 7ffd:调整+1 ppm
* 7ffe:调整+0.5 ppm
* 7fff:调整 0 ppm,即不调整
* 8000:调整-0.5 ppm
* 8001:调整-1 ppm
* …
* 87fe:调整-1023.5ppm
* 87ff:调整-1024ppm

#### 18.12.5 RTC\_CALENDAR

Offset:0x10 Reset Value:0x00000000

|  |  |
| --- | --- |
| **31-20** | **19-0** |
| RESERVED | CALENDAR\_VALUE |
| r-0h | w-0h |

**Bits 31-20 RESERVED:**保留.

**Bits 19-0 CALENDAR\_VALUE:**RTC 日历数据低Bits.

[3:0]:秒个Bits

[6:4]:秒十Bits

[10:7]:分钟个Bits

[13:11]:分钟十Bits

[17:14]:小时个Bits

[19:18]:小时十Bits

#### 18.12.6 RTC\_CALENDAR\_H

Offset:0x14 Reset Value:0x00000841

|  |  |
| --- | --- |
| **31-22** | **21-0** |
| RESERVED | CALENDAR\_H\_VALUE |
| r-0h | w-841h |

**Bits 31-22 RESERVED:**保留.

**Bits 21-0 CALENDAR\_H\_VALUE:**RTC 日历数据高Bits.

[3:0]:日期个Bits

[5:4]:日期十Bits

[9:6]:月份个Bits

[10]:月份十Bits

[13:11]:星期

[17:14]:年个Bits

[21:18]:年十Bits

#### 18.12.7 RTC\_CYC\_MAX\_VALUE

Offset:0x18 Reset Value:0x00008000

|  |
| --- |
| **31-0** |
| CYC\_MAX\_VALUE |
| rw-8000h |

**Bits 31-0 CYC\_MAX\_VALUE:**周期计数值配置,当周期counter的计数值等于 CYC\_MAX\_VALUE 时,触发周期唤醒状态Bits,周期计数使用 RTC 接口clock 进行计数.

#### 18.12.8 RTC\_SR

Offset:0x1c Reset Value:0x00000000

|  |  |  |  |
| --- | --- | --- | --- |
| **31-7** | **6** | **5** | **4** |
| RESERVED | ALARM0\_SR | ALARM1\_SR | CYC\_SR |
| r-0h | rw-0h | rw-0h | rw-0h |
| **3** | **2** | **1** | **0** |
| TAMPER\_SR | WAKEUP0\_SR | WAKEUP1\_SR | WAKEUP2\_SR |
| rw-0h | rw-0h | rw-0h | rw-0h |

**Bits 31-7 RESERVED:**保留.

**Bits 6 ALARM0\_SR:**闹钟 0 触发状态Bits.硬件置 1,软件写 1 清 0.

* 0:未发生
* 1:发生**Bits 5 ALARM1\_SR:**闹钟 1 触发状态Bits.硬件置 1,软件写 1 清 0.
* 0:未发生
* 1:发生**Bits 4 CYC\_SR:**周期唤醒触发状态Bits.硬件置 1,软件写 1 清 0.
* 0:未发生
* 1:发生**Bits 3 TAMPER\_SR:**tamper 触发状态Bits.硬件置 1,软件写 1 清 0.
* 0:未发生
* 1:发生**Bits 2 WAKEUP0\_SR:**wakeup0 触发状态Bits.硬件置 1,软件写 1 清 0.
* 0:未发生
* 1:发生**Bits 1 WAKEUP1\_SR:**wakeup1 触发状态Bits.硬件置 1,软件写 1 清 0.
* 0:未发生
* 1:发生**Bits 0 WAKEUP2\_SR:**wakeup2 触发状态Bits.硬件置 1,软件写 1 清 0.
* 0:未发生
* 1:发生

#### 18.12.9 RTC\_ASYNDATA

Offset:0x20 Reset Value:0x00000000

|  |  |
| --- | --- |
| **31-20** | **19-0** |
| RESERVED | SYN\_DATA |
| r-0h | r-0h |

**Bits 31-20 RESERVED:**保留.

**Bits 19-0 SYN\_DATA:**RTC 日历数据低Bits.软件只读.

[3:0]:秒个Bits

[6:4]:秒十Bits

[10:7]:分钟个Bits

[13:11]:分钟十Bits

[17:14]:小时个Bits

[19:18]:小时十Bits

#### 18.12.10 RTC\_ASYNDATA\_H

Offset:0x24 Reset Value:0x00000000

|  |  |
| --- | --- |
| **31-22** | **21-0** |
| RESERVED | SYN\_DATA\_H |
| r-0h | r-0h |

**Bits 31-22 RESERVED:**保留.

**Bits 21-0 SYN\_DATA\_H:**RTC 日历数据高Bits.软件只读.

[3:0]:日期个Bits

[5:4]:日期十Bits

[9:6]:月份个Bits

[10]:月份十Bits

[13:11]:星期

[17:14]:年个Bits

[21:18]:年十Bits

#### 18.12.11 RTC\_CR1

Offset:0x28 Reset Value:0x00000000

|  |  |  |
| --- | --- | --- |
| **31-8** | **7** | **6** |
| RESERVED | SECOND\_SR\_INT\_EN | ALARM0\_SR\_INT\_EN |
| r-0h | rw-0h | rw-0h |
| **5** | **4** | **3** |
| ALARM1\_SR\_INT\_EN | CYC\_SR\_INT\_EN | TAMPER\_SR\_INT\_EN |
| rw-0h | rw-0h | rw-0h |
| **2** | **1** | **0** |
| WAKEUP0\_SR\_INT\_EN | WAKEUP1\_SR\_INT\_EN | WAKEUP2\_SR\_INT\_EN |
| rw-0h | rw-0h | rw-0h |

**Bits 31-8 RESERVED:**保留.

**Bits 7 SECOND\_SR\_INT\_EN:**秒interruptEnable.

* 0:Disable
* 1:Enable**Bits 6 ALARM0\_SR\_INT\_EN:**ALARM0\_SR interruptEnable.
* 0:Disable
* 1:Enable**Bits 5 ALARM1\_SR\_INT\_EN:**ALARM1\_SR interruptEnable.
* 0:Disable
* 1:Enable**Bits 4 CYC\_SR\_INT\_EN:**CYC\_SR interruptEnable.
* 0:Disable
* 1:Enable**Bits 3 TAMPER\_SR\_INT\_EN:**TAMPER\_SR interruptEnable.
* 0:Disable
* 1:Enable**Bits 2 WAKEUP0\_SR\_INT\_EN:**WAKEUP0\_SR interruptEnable.
* 0:Disable
* 1:Enable**Bits 1 WAKEUP1\_SR\_INT\_EN:**WAKEUP1\_SR interruptEnable.
* 0:Disable
* 1:Enable**Bits 0 WAKEUP2\_SR\_INT\_EN:**WAKEUP2\_SR interruptEnable.
* 0:Disable
* 1:Enable

#### 18.12.12 RTC\_SR1

Offset:0x2c

Reset Value:0x00000dff

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **31-12** | **11** | | **10** | | **9** |
| RESERVED | WRITE\_ALARM0\_SUB\_DONE | | WRITE\_ALARM1\_SUB\_DONE | | SECOND\_SR |
| r-0h | r-1h | | r-1h | | rw-0h |
| **8** | | **7** | | **6** | |
| WRITE\_RTCCR2\_DONE | | WRITE\_RTCCR\_DONE | | WRITE\_ALARM0\_DONE | |
| r-1h | | r-1h | | r-1h | |
| **5** | | **4** | | **3** | |
| WRITE\_ALARM1\_DONE | | WRITE\_PPMADJUST\_DONE | | WRITE\_CALENDAR\_DONE | |
| r-1h | | r-1h | | r-1h | |
| **2** | | **1** | | **0** | |
| WRITE\_CYC\_MAX\_VALUE\_DONE | | WRITE\_RTCSR\_DONE | | READ\_CALENDAR\_DONE | |
| r-1h | | r-1h | | r-1h | |

**Bits 31-12 RESERVED:**保留.

**Bits 11 WRITE\_ALARM0\_SUB\_DONE:**对 *RTC\_ALARM0\_SUB* Register的写操作的完成状态.硬件控制置 1 与清 0.

* 0:操作正在进行中
* 1:操作已完成**Bits 10 WRITE\_ALARM1\_SUB\_DONE:**对 *RTC\_ALARM1\_SUB* Register的写操作的完成状态.硬件控制置 1 与清 0.
* 0:操作正在进行中
* 1:操作已完成**Bits 9 SECOND\_SR:**秒interrupt状态.硬件置 1,软件写 1 清 0.
* 0:无interrupt发生
* 1:有interrupt发生**Bits 8 WRITE\_RTCCR2\_DONE:**对 *RTC\_CR2* Register的写操作的完成状态.硬件控制置 1 与清 0.
* 0:操作正在进行中
* 1:操作已完成**Bits 7 WRITE\_RTCCR\_DONE:**对 *RTC\_CR* Register的写操作的完成状态.硬件控制置 1 与清 0.
* 0:操作正在进行中
* 1:操作已完成**Bits 6 WRITE\_ALARM0\_DONE:**对 *RTC\_ALARM0* Register的写操作的完成状态.硬件控制置 1 与清 0.
* 0:操作正在进行中
* 1:操作已完成**Bits 5 WRITE\_ALARM1\_DONE:**对 *RTC\_ALARM1* Register的写操作的完成状态.硬件控制置 1 与清 0.
* 0:操作正在进行中
* 1:操作已完成**Bits 4 WRITE\_PPMADJUST\_DONE:**对 *RTC\_PPMADJUST* Register的写操作的完成状态.硬件控制置 1 与清 0.
* 0:操作正在进行中
* 1:操作已完成**Bits 3 WRITE\_CALENDAR\_DONE:**对 *RTC\_CALENDAR* 和 *RTC\_CALENDAR\_H* Register的写操作的完成状态.硬件控制置 1 与清 0.
* 0:操作正在进行中
* 1:操作已完成**Bits 2 WRITE\_CYC\_MAX\_VALUE\_DONE:**对 *RTC\_CYC\_MAX\_VALUE* 的写操作的完成状态.硬件控制置 1 与清 0.
* 0:操作正在进行中
* 1:操作已完成**Bits 1 WRITE\_RTCSR\_DONE:**对 *RTC\_SR* Register的写操作的完成状态.硬件控制置 1 与清 0.
* 0:操作正在进行中
* 1:操作已完成**Bits 0 READ\_CALENDAR\_DONE:**对 *RTC\_CALENDAR\_R* 和 *RTC\_CALENDAR\_R\_H* Register的读操作的完成状态.硬件控制置 1 与清 0.
* 0:操作正在进行中
* 1:操作已完成

#### 18.12.13 RTC\_CR2

Offset:0x30 Reset Value:0x00000000

|  |  |  |  |
| --- | --- | --- | --- |
| **31-8** | **7** | **6-4** | **3-0** |
| RESERVED | RTC\_OUT\_POL | RTC\_OUT\_SEL | RTC\_RET\_SRAM\_ERASE\_EN |
| r-0h | rw-0h | rw-0h | rw-0h |

**Bits 31-8 RESERVED:**保留.

**Bits 7 RTC\_OUT\_POL:**RTC IO 输出电平选择.

* 0:原电平
* 1:取反电平**Bits 6-4 RTC\_OUT\_SEL:**RTC IO 输出选择.
* 0-3:输出 0
* 4:alarm0 脉冲
* 5:alarm1 脉冲
* 6:cyc 脉冲
* 7:秒信号,占空比 50%

**Bits 3-0 RTC\_RET\_SRAM\_ERASE\_EN:**tamper 与 wakeup 报警触发 Retention SRAM 清除的功能Enable.[0] Correspond to wakeup0,[1] Correspond to wakeup1,[2] Correspond to wakeup2,[3] Correspond to tamper.

* 0:Disable
* 1:Enable

#### 18.12.14 RTC\_SUB\_SECOND

Offset:0x34 Reset Value:0x00000000

|  |  |
| --- | --- |
| **31-15** | **14-0** |
| RESERVED | RTC\_SUB\_SECOND\_VALUE |
| r-0h | r-0h |

**Bits 31-15 RESERVED:**保留.

**Bits 14-0 RTC\_SUB\_SECOND\_VALUE:**rtc 日历计数中的秒以下部分的counter值,由于采用打拍同步,需要多次读到同一值,才可以使用,不能仅读取一次（可能由于同步而读错）.

#### 18.12.15 RTC\_CYC\_CNT\_VALUE

Offset:0x38 Reset Value:0x00000000

|  |
| --- |
| **31-0** |
| CYC\_CNT\_VALUE |
| r-0h |

**Bits 31-0 CYC\_CNT\_VALUE:**周期counter值,由于采用打拍同步,需要多次读到同一值,才可以使用,不能仅读取一次（可能由于同步而读错）.

#### 18.12.16 RTC\_ALARM0\_SUB

Offset:0x3c Reset Value:0x00000000

|  |  |  |  |
| --- | --- | --- | --- |
| **31-20** | **19-16** | **15** | **14-0** |
| RESERVED | RTC\_ALARM0\_SUB\_MASK | RESERVED | RTC\_ALARM0\_SUB\_VALUE |
| r-0h | rw-0h | r-0h | rw-0h |

**Bits 31-20 RESERVED:**保留.

**Bits 19-16 RTC\_ALARM0\_SUB\_MASK:**闹钟 0 的 sub-second Mask 配置,使用 alarm0 的 subsecond 功能时,ppm 调整功能建议不使用.

* 0:sub-second 定时值配置不生效,仅在每秒考虑 rtc\_alarm0 是否匹配.
* 1:匹配 sub-second 计数的[0],其它 sub-second 计数Bits不生效.
* 2:匹配 sub-second 计数的[1:0],其它 sub-second 计数Bits不生效.
* N:匹配 sub-second 计数的[N-1:0],其它 sub-second 计数Bits不生效.

**Bits 15 RESERVED:**保留.

**Bits 14-0 RTC\_ALARM0\_SUB\_VALUE:**闹钟 0 sub-second 定时值配置.当 rtc 日历计数值与闹钟

0 配置匹配时,产生 ALARM0\_SR.

#### 18.12.17 RTC\_ALARM1\_SUB

Offset:0x40 Reset Value:0x00000000

|  |  |  |  |
| --- | --- | --- | --- |
| **31-20** | **19-16** | **15** | **14-0** |
| RESERVED | RTC\_ALARM1\_SUB\_MASK | RESERVED | RTC\_ALARM1\_SUB\_VALUE |
| r-0h | rw-0h | r-0h | rw-0h |

**Bits 31-20 RESERVED:**保留.

**Bits 19-16 RTC\_ALARM1\_SUB\_MASK:**闹钟 1 的 sub-second Mask 配置,使用 alarm1 的 subsecond 功能时,ppm 调整功能建议不使用.

* 0:sub-second 定时值配置不生效,仅在每秒考虑 rtc\_alarm1 是否匹配.
* 1:匹配 sub-second 计数的[0],其它 sub-second 计数Bits不生效.
* 2:匹配 sub-second 计数的[1:0],其它 sub-second 计数Bits不生效.
* N:匹配 sub-second 计数的[N-1:0],其它 sub-second 计数Bits不生效.

**Bits 15 RESERVED:**保留.

**Bits 14-0 RTC\_ALARM1\_SUB\_VALUE:**闹钟 1 sub-second 定时值配置.当 rtc 日历计数值与闹钟 1 配置匹配时,产生 ALARM1\_SR.

#### 18.12.18 RTC\_CALENDAR\_R

Offset:0x44 Reset Value:0x00000000

|  |  |
| --- | --- |
| **31-20** | **19-0** |
| RESERVED | CALENDAR\_SYNC |
| r-0h | r-0h |

**Bits 31-20 RESERVED:**保留.

**Bits 19-0 CALENDAR\_SYNC:**RTC 日历计数中的时分秒的counter值,由于采用打拍同步,需要多次读到同一值,才可以使用,不能仅读取一次（可能由于同步而读错）.

#### 18.12.19 RTC\_CALENDAR\_R\_H

Offset:0x48 Reset Value:0x00000841

|  |  |
| --- | --- |
| **31-22** | **21-0** |
| RESERVED | CALENDAR\_H\_SYNC |
| r-0h | r-841h |

**Bits 31-22 RESERVED:**保留.

**Bits 21-0 CALENDAR\_H\_SYNC:**RTC 日历计数中的年月日星期的counter值,由于采用打拍同步,需要多次读到同一值,才可以使用,不能仅读取一次（可能由于同步而读错）.

### 19. 低功耗通用异步接收器 (LPUART)

#### 19.1 Introduction

LPUART（Low Power Universal Asynchronous Receiver/Transmitter）是一种低功耗的串口外设,32K clock 下波特率最高支持 9600.在极低功耗模式下,LPUART 也可以被接收到的数据唤醒.

LPUART 支持 CTS/RTS 流量控制.

LPUART 支持 DMA Request .

#### 19.2 Main features

* 波特率可配置
* 数据格式可配置（5-8 Bits数据,1-2 Bits停止Bits,0-1 Bits奇偶校验Bits）
* 支持 DMA Request
* TX/RX FIFO 深度为 1
* 支持 CTS/RTS 流控
* 支持interruptRequest
* 支持低功耗唤醒

#### 19.3 功能Description

##### 19.3.1 数据格式

0

/

1

0

/

1

0

/

1

0

/

1

0

/

1

|  |  |  |
| --- | --- | --- |
| DATA | PARITY | STOP |

START

**Figure 19-1 LPUART 的数据传输格式**

在空闲时,LPUART 的数据线应该保持在高电平.

在数据传输时,依次传输起始Bits（START）, 数据Bits（DATA）,奇偶校验Bits（PARITY）和停止Bits（STOP）.各个Bits的含义如下:

1. **起始Bits:**先发送一个 0 信号,Table示数据传输开始.
2. **数据Bits:**根据配置,依次传输 5-8 个 bit.
3. **奇偶校验Bits:**数据Bits后,传输一个 bit 的奇偶校验Bits,也可以配置为无奇偶校验Bits.
4. **停止Bits:**数据传输结束的标志,可以是 1 或者 2 个 bit.

##### 19.3.2 波特率产生

LPUART 波特率的配置支持小数分频,其主要通过 LPUART\_BAUD\_RATE\_INT 和 LPUART\_

BAUD\_RATE\_FRA 两个Register来配置.

以LPUART接口clock 频率为32.768KHz,波特率为9600为例,分频系数为32768/9600=3.413,

则Register LPUART\_BAUD\_RATE\_INT 配置为 3,Register LPUART\_BAUD\_RATE\_FRA 配置为

7（0.413\*16=6.608,四舍五入为 7）.

##### 19.3.3 CTS/RTS 流控

两个 LPUART 之间的连接如下Figure:

LPUART

1

LPUART

2

TX

TX

RX

RX

RTS

CTS

CTS

RTS

**Figure**

**19**

**-**

**2**

两个

**LPUART**

设备之间的连接

**RTS（Require To Send,发送Request ）**为输出信号,用于指示本设备准备好可接收数据,低电平有效,低电平说明本设备可以接收数据.

**CTS（Clear To Send,发送allowed）**为输入信号,用于判断是否可以向对方发送数据,低电平有效,低电平说明本设备可以向对方发送数据.

##### 19.3.4 DMA Request

###### LPUART DMA 发送过程:

1. 将Register LPUART\_CR1 中的 DMA\_TX\_EN Bits配置为Enable；
2. 将Register LPUART\_DATA 地址配置为 DMA 的目的地址；
3. 将发送数据的内存地址配置为 DMA 的源地址；
4. 配置 DMA 的 SRC\_TR\_WIDTH 和 DES\_TR\_WIDTH 为 0（数据Bits宽为 8bit）；
5. 配置 DMA 的 SRC\_MSIZE 和 DEST\_MSIZE 为 0（burst length 为 1）；
6. 配置 DMA 的数据传输总长度；
7. 配置 DMA 的 handshake 类型为 DMA\_HANDSHAKE\_LPUART\_TX；
8. 激活 DMA 通道.

当 DMA 传输完成后,会将 DMA\_CHENREG Register的 CH\_EN\_x Bits清 0.

###### LPUART DMA 接收过程:

1. 将Register LPUART\_CR1 中的 DMA\_RX\_EN Bits配置为Enable；
2. 将Register LPUART\_DATA 地址配置为 DMA 的源地址；
3. 将数据接收的内存地址配置为 DMA 的目的地址；
4. 配置 DMA 的 SRC\_TR\_WIDTH 和 DES\_TR\_WIDTH 为 0（数据Bits宽为 8bit）；
5. 配置 DMA 的 SRC\_MSIZE 和 DEST\_MSIZE 为 0（burst length 为 1）；
6. 配置 DMA 的数据传输总长度；
7. 配置 DMA 的 handshake 类型为 DMA\_HANDSHAKE\_LPUART\_RX；
8. 激活 DMA 通道.当 DMA 传输完成后,会将 DMA\_CHENREG Register的 CH\_EN\_x Bits清 0.

##### 19.3.5 interrupt信号

LPUART 的interrupt信号主要有:

* TX\_DONE interrupt
* TXFIFO\_EMPTY interrupt
* RXFIFO\_NOT\_EMPTY interrupt
* RX\_OVERFLOW interrupt
* STOP\_ERR interrupt
* PARITY\_ERR interrupt
* START\_INVALID interrupt
* RX\_DONE interrupt
* START\_VALID interrupt

##### 19.3.6 低功耗唤醒

LPUART 的低功耗唤醒包括 RX 低电平唤醒,有效 START 唤醒,RX\_DONE 唤醒.

通过配置 LPUART\_CR0 Register的 LPUART\_WAKEUP\_EN Bits来Enable唤醒方式.

##### 19.4 LPUART 相关RegisterDescription

Base Address:0x40005000

**Table 19-1 LPUART Register Summary**

|  |  |  |
| --- | --- | --- |
| Register | Offset | Description |
| LPUART\_CR0 | 0x00 | Control Register 0 |
| LPUART\_CR1 | 0x04 | Control Register 1 |
| LPUART\_SR0 | 0x08 | 状态Register 0 |
| LPUART\_SR1 | 0x0C | 状态Register 1 |
| LPUART\_DATA | 0x10 | 数据Register |

###### 19.4.1 LPUART\_CR0

Offset:0x00 Reset Value:0x00000E13

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **31-27** | **26** | | **25** | **24-22** | **21-10** |
| RESERVED | LPUART\_RTS\_EN | | LPUART\_RX\_EN | LPUART\_WAKEUP\_EN | LPUART\_BAUD\_RA  TE\_INT |
| r | r/w | | r/w | r/w | r/w |
| **9-6** | | **5** | | **4-2** | **1-0** |
| LPUART\_BAUD\_RATE\_FRA | | LPUART\_STOP\_LEN | | LPUART\_PARITY\_CFG | LPUART\_DATA\_LEN |
| r/w | | r/w | | r/w | r/w |

**Bits 31-27 RESERVED:** Must be kept, and can't be modified.

**Bits 26 LPUART\_RTS\_EN:**lpuart rts 流控Enable.

* 0:Disable
* 1:Enable**Bits 25 LPUART\_RX\_EN:**lpuart 接收Enable.
* 0:Disable
* 1:Enable**Bits 24-22 LPUART\_WAKEUP\_EN:**lpuart 唤醒Enable.

1. 为 rx 低电平唤醒Enable.
   * 0:Disable
   * 1:Enable
2. 为有效 start 唤醒Enable.
   * 0:Disable
   * 1:Enable
3. 为接收数据完成唤醒Enable.
   * 0:Disable
   * 1:Enable**Bits 21-10 LPUART\_BAUD\_RATE\_INT:**波特率分频系数的整数部分.

分频系数等于 UART 接口clock 频率/波特率；

以UART接口clock 频率为32.768KHz,波特率为9600为例,分频系数为32768/9600=3.413,lpuart\_baud\_ rate\_int 配置为 3,lpuart\_baud\_rate\_fra 配置为 0.413\*16=6 或 7.

**Bits 9-6 LPUART\_BAUD\_RATE\_FRA:**波特率分频系数的小数部分,支持 4 Bits小数分频.

**Bits 5 LPUART\_STOP\_LEN:**LPUART stop Bits长度.

* + 0:1 Bits stop
  + 1:2 Bits stop

**Bits 4-2 LPUART\_PARITY\_CFG:**LPUART 奇偶校验配置.

* + 0:偶校验
  + 1:奇校验
  + 2:校验Bits固定为 0
  + 3:校验Bits固定为 1
  + >3:无奇偶校验Bits**Bits 1-0 LPUART\_DATA\_LEN:**LPUART 数据宽度.数据宽度等于 LPUART\_DATA\_LEN+5.

###### 19.4.2 LPUART\_CR1

Offset:0x04 Reset Value:0x00000000

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| **31-13** | **12** | | **11** | | **10** | | **9** |
| RESERVED | LPUART\_CTS\_EN | | DMA\_TX\_EN | | DMA\_RX\_EN | | LPUART\_TX\_EN |
| r | r/w | | r/w | | r/w | | r/w |
| **8** | **7** | | **6** | | **5** | | **4** |
| TX\_DONE\_INT  \_EN | TXFIFO\_EMPTY\_INT  \_EN | | RXFIFO\_NOT\_EMPTY  \_INT\_EN | | RX\_OVERFLOW  \_INT\_EN | | STOP\_ERR\_INT  \_EN |
| r/w | r/w | | r/w | | r/w | | r/w |
| **3** | | **2** | | **1** | | **0** | |
| PARITY\_ERR\_INT\_EN | | START\_INVALID\_INT\_EN | | RX\_DONE\_INT\_EN | | START\_VALID\_INT\_EN | |
| r/w | | r/w | | r/w | | r/w | |

**Bits 31:13 RESERVED:**Must be kept, and can't be modified.**Bits 12 LPUART\_CTS\_EN:**lpuart cts 流控Enable.

* 0:Disable
* 1:Enable**Bits 11 DMA\_TX\_EN:**dma tx Request Enable.
* 0:Disable
* 1:Enable**Bits 10 DMA\_RX\_EN:**dma rx Request Enable.
* 0:Disable
* 1:Enable**Bits 9 LPUART\_TX\_EN:**lpuart 发送Enable.
* 0:Disable
* 1:Enable**Bits 8 TX\_DONE\_INT\_EN:**tx\_done interruptEnable.
* 0:Disable
* 1:Enable

**Bits 7 TXFIFO\_EMPTY\_INT\_EN:**txfifo\_empty interruptEnable.

* 0:Disable
* 1:Enable

**Bits 6 RXFIFO\_NOT\_EMPTY\_INT\_EN:**rxfifo\_not\_empty interruptEnable.

* 0:Disable
* 1:Enable

**Bits 5 RX\_OVERFLOW\_INT\_EN:**rx\_overflow interruptEnable.

* 0:Disable
* 1:Enable

**Bits 4 STOP\_ERR\_INT\_EN:**stop\_err interruptEnable.

* 0:Disable
* 1:Enable

**Bits 3 PARITY\_ERR\_INT\_EN:**parity\_err interruptEnable.

* 0:Disable
* 1:Enable

**Bits 2 START\_INVALID\_INT\_EN:**start\_invalid interruptEnable.

* 0:Disable
* 1:Enable

**Bits 1 RX\_DONE\_INT\_EN:**rx\_done interruptEnable.

* 0:Disable
* 1:Enable

**Bits 0 START\_VALID\_INT\_EN:**start\_valid interruptEnable.

* 0:Disable
* 1:Enable

###### 19.4.3 LPUART\_SR0

Offset:0x08 Reset Value:0x00000000

|  |  |  |  |
| --- | --- | --- | --- |
| **31-6** | | **5** | **4** |
| RESERVED | | RX\_OVERFLOW\_SR | STOP\_ERR\_SR |
| r | | r/w | r/w |
| **3** | **2** | **1** | **0** |
| PARITY\_ERR\_SR | START\_INVALID\_SR | RX\_DONE\_SR | START\_VALID\_SR |
| r/w | r/w | r/w | r/w |

**Bits 31-6 RESERVED:** Must be kept, and can't be modified.

**Bits 5 RX\_OVERFLOW\_SR:**rx\_overflow 状态Bits,用于指示是否发生接收数据 Buffer 溢出.硬件置 1,软件写 1 清 0.

* 0:未发生
* 1:发生**Bits 4 STOP\_ERR\_SR:**stop\_err 状态Bits,用于指示是否发生 stop 电平错误.硬件置 1,软件写 1 清

0.

* 0:未发生
* 1:发生**Bits 3 PARITY\_ERR\_SR:**parity\_err 状态Bits,用于指示是否发生 parity 校验错误.硬件置 1,软件写 1 清 0.
* 0:未发生
* 1:发生**Bits 2 START\_INVALID\_SR:**start\_invalid 状态Bits,用于指示是否收到错误的 start Bits.硬件置 1,软件写 1 清 0.
* 0:未发生
* 1:发生**Bits 1 RX\_DONE\_SR:**rx\_done 状态Bits,用于指示是否接收完数据.硬件置 1,软件写 1 清 0.
* 0:未发生
* 1:发生**Bits 0 START\_VALID\_SR:**start\_valid 状态Bits,用于指示是否收到有效的 start Bits.硬件置 1,软件写 1 清 0.
* 0:未发生
* 1:发生

###### 19.4.4 LPUART\_SR1

Offset:0x0C

Reset Value:0x00000016

|  |  |  |  |
| --- | --- | --- | --- |
| **31-6** | | **5** | **4** |
| RESERVED | | TX\_DONE | TXFIFO\_EMPTY |
| r | | r/w | r |
| **3** | **2** | **1** | **0** |
| RXFIFO\_NOT\_EMPTY | WRITE\_CR0\_DONE | WRITE\_SR0\_DONE | RESERVED |
| r | r | r | r |

**Bits 31-6 RESERVED:** Must be kept, and can't be modified.

**Bits 5 TX\_DONE:**tx\_done 状态Bits.硬件置 1,软件写 1 清 0.

* 0:发送未完成
* 1:发送完成**Bits 4 TXFIFO\_EMPTY:**txfifo\_empty 状态Bits.硬件置 1,软件写 LPUART\_DATA 来清 0 此Bits.
* 0:非空
* 1:空

**Bits 3 RXFIFO\_NOT\_EMPTY:**rxfifo\_not\_empty 状态Bits.硬件置 1,软件读 LPUART\_DATA 来清 0 此Bits.

* 0:空
* 1:非空**Bits 2 WRITE\_CR0\_DONE:**write cr0 操作完成状态.硬件控制置 1 与清 0.
* 0:操作正在进行中
* 1:操作已完成**Bits 1 WRITE\_SR0\_DONE:**write sr0 操作完成状态.硬件控制置 1 与清 0.
* 0:操作正在进行中
* 1:操作已完成**Bits 0 RESERVED:** Must be kept, and can't be modified.

###### 19.4.5 LPUART\_DATA

Offset:0x10 Reset Value:0x00000000

|  |  |
| --- | --- |
| **31-8** | **7-0** |
| RESERVED | LPUART\_DATA |
| r | r/w |

**Bits 31-8 RESERVED:** Must be kept, and can't be modified.

**Bits 7-0 LPUART\_DATA:**fifo 接口,读操作返回 rx data,写操作写入 tx data.

Notice:

1. 如果数据Bits宽小于*8*时,*LPUART\_DATA*的低Bits有效.
2. 读之前要查询*RXFIFO\_NOT\_EMPTY* 状态Bits,保证*rxfifo* 中有数据；写之前要查询*txfifo\_ empty* 状态Bits,保证*txfifo*中可写入.

## 20. 低功耗定时器 (LPTIM)

### 20.1 Introduction

LPTIMER（Low Power Timer）是一个 16 Bits的计时器,由于其有多种clock 源,LPTIMER 能够在除 standby 模式外的所有工作模式下运行,并且支持从所有低功耗工作模式中唤醒.有两个

LPTIMER,分别为 LPTIMER0 和 LPTIMER1.

### 20.2 Main features

LPTIMER 包括如下功能:

* 支持选择内部clock 与外部clock 作为计数clock
* 16bits counter,加法计数,支持自动加载
* 支持两种计数模式,单次计数和连续计数
* 支持软件触发和外部触发源触发计数
* 分频counter
* 支持产生 PWM
* 支持单脉冲、Set-once、Timeout 模式输出
* 支持 DEBUG 模式控制
* 支持产生通道输出事件、匹配事件、溢出事件、触发事件、DOWN 事件、UP 事件作为唤醒信号输出
* 正交解码
* interrupt信号产生

LPTIMER 的框Figure如下:



**Figure 20-1 LPTIMER 框Figure**

####  lptim\_trig0~7:LPTIMER 的外部触发源

* **lptim\_in2:**LPTIMER 的 IN2 pin 脚  **lptim\_in1:**LPTIMER 的 IN1 pin 脚
* **lptim\_wkup:**LPTIMER 的唤醒信号
* **lptim\_out:**LPTIMER 的 OUT pin 脚

### 20.3 接口clock

LPTIMER 接口clock 源分内部clock 和外部clock ,内部clock 源有 PCLK0、RCO3.6M、XO32K、 RCO32K,外部clock 源通过 IN1 的 GPIO 输入.clock 配置和选择可以参考 RCC 章节.

### 20.4 计数clock 选择

LPTIMER 除了接口clock 有内部和外部之分外,计数clock 也有内部和外部之分,其内部与外部的clock 源与接口clock 的一致.控制计数clock 选择的Register bit Bits为Register LPTIM\_CFGR 的 COUNTMODE,值为 0 Table示counter由内部clock 控制,值为 1 Table示counter由外部clock 控制,如果 RCC 模块的Register RCC\_CR1 的 LPTIM1\_EXT\_CLK\_SEL Bits或 LPTIM\_EXT\_CLK\_SEL Bits为 0,即Table示 LPTIMER0 或 LPTIMER1 的接口clock 为内部clock ,则 COUNTMODE 的值可以为 0 或 1,即计数clock 既可以为内部clock 也可以为外部clock ；如果 LPTIM1\_EXT\_CLK\_SEL Bits或 LPTIM\_EXT\_CLK\_SEL Bits为 1,则 LPTIMER0 或 LPTIMER1 的Register LPTIM\_CFGR 的 COUNTMODE Bits只能设置为 0,这时 0 不是Table示计数clock 为内部clock ,是Table示 COUNTMODE 值需要清零,计数clock 只能为外部clock .

### 20.5 counter

除编码模式外,counter仅支持向上计数,计数到 ARR 时产生 ARRM interrupt,counter回到 0 重新计数.若Enable timeout 模式,则除了counter值增加到 ARR 时清零外,触发信号也可以清零counter重新计数.若Enable编码模式,则counter的计数方向由硬件控制,向上计数到 ARR 时产生

ARRM 事件并清零counter,向下计数到 0 时则重新加载 ARR 到counter.

#### 20.6 计数模式

LPTIMER 支持两种计数模式,单次计数和连续计数.单次计数模式下,counter停止阶段第一个到来的触发信号（硬件或软件）会触发counter开始计数,计数过程中的触发信号将会被忽略,计数到 ARR 时counter会停止计数,直到下一次触发信号到来才会再次开始计数,依次类推.连续计数模式下,一旦触发（硬件或软件）,counter会一直计数下去,从 0 到 ARR,然后回到

0 再次计数,如此循环往复.

两种计数模式可以在任意时刻切换（前提是 enable 置Bits）,例如,配置 LPTIMER 为单次计数模式,若置BitsRegister LPTIM\_CR 的 CNTSTRT,则counter计数到 ARR 值将不会停止计数；配置 LPTIMER 为连续计数模式,若置BitsRegister LPTIM\_CR 的 SNGSTRT,则counter计数到 ARR 时将会停止计数,直到下一次触发信号到来.因此状态Figure如下:



IDLE



SNG



WAIT

\_

CONT



WAIT

\_

SNG



CONT

SNGSTRT



SNG

：

单次计数模式



CONT

：

连续计数模式



WAIT

\_

SNG

：

等待进入单次计数模式



WAIT

\_

CONT

：

等待进入连续计数模式

**Figure 20-2 计数模式转换Figure**

#### 20.7 软件触发和外部触发

触发 LPTIMER 计数有两种方式,一种是软件触发,另一种是外部触发源触发.通过Register LPTIM\_CFGR 的 TRIGEN Bits段进行控制,当值为 0 时为软件触发,非零时为外部触发,当为外部触发时,可以设置外部触发信号上升沿有效、下降沿有效或双沿有效.LPTIMER 有 8 种触发输入源可以选择使用,**LPTIMER0** 的外部触发源如下Table:

**Table 20-1 LPTIMER0 的外部触发源**

|  |  |  |
| --- | --- | --- |
| TRIGSEL | External Trigger | Comment |
| lptim\_ext\_trig0 | lptim\_etr | Lptimer etr pin input |
| lptim\_ext\_trig1 | comp0 | Comp0 output |
| lptim\_ext\_trig2 | comp1 | Comp1 output |
| lptim\_ext\_trig3 | rtc\_cyc\_counter | RTC cyc counter output pulse |
| lptim\_ext\_trig4 | rtc\_alarm0 | RTC alarm0 output pulse |
| lptim\_ext\_trig5 | rtc\_alarm1 | RTC alarm1 output pulse |
| lptim\_ext\_trig6 | gpio | GPIO58 |
| lptim\_ext\_trig7 | gpio | GPIO59 |

**LPTIMER1** 的外部触发源如下Table:

**Table 20-2 LPTIMER1 的外部触发源**

|  |  |  |
| --- | --- | --- |
| TRIGSEL | External Trigger | Comment |
| lptim\_ext\_trig0 | lptim\_etr | Lptimer etr pin input |
| lptim\_ext\_trig1 | comp0 | Comp0 output |
| lptim\_ext\_trig2 | comp1 | Comp1 output |
| lptim\_ext\_trig3 | rtc\_cyc\_counter | RTC cyc counter output pulse |
| lptim\_ext\_trig4 | rtc\_alarm0 | RTC alarm0 output pulse |
| lptim\_ext\_trig5 | rtc\_alarm1 | RTC alarm1 output pulse |
| lptim\_ext\_trig6 | gpio | GPIO60 |
| lptim\_ext\_trig7 | gpio | GPIO61 |

#### 20.8 分频counter

计数Enable信号可以被软件配置分频,支持 1、2、4、8、16、32、64、128 分频,通过配置Register LPTIM\_CFGR 的 PRESC Bits段进行分频配置.该分频通过counter实现,即上一级电路产生的计数Enable信号将作为该分频counter的计数Enable,当分频counter计数到预先加载的分频值后,输出一个脉冲,作为下一级counter的计数Enable,然后分频counter归零重新计数,依次类推.

### 20.9 PWM

LPTIMER 可以产生 PWM 波形,波形的极性可以通过Register LPTIM\_CFGR 的 WAVPOL 比特控制,占空比可以通过Register LPTIM\_CMP 和 LPTIM\_ARR 的值进行控制.以软件触发和内部clock 计数为例,配置 PWM 的流程如下:

1. 配置Register LPTIM\_CFGR 的 COUNTMODE 为 0,即设置内部clock 计数.
2. Register LPTIM\_CFGR 的 PRESC 为默认值,即不设置counter分频.
3. 配置Register LPTIM\_CFGR 的 PRELOAD 的值为 0,即DisableRegister LPTIM\_CMP 和

LPTIM\_ARR 的缓存功能.如果需要也可以Enable.

1. 配置Register LPTIM\_CFGR 的 WAVPOL 为 0,即波形输出不反相.
2. 配置Register LPTIM\_CFGR 的 WAVE 为 0.
3. Register LPTIM\_CFGR 的 TRIGEN Bits段的值为 0,即软件触发.
4. Enable LPTIMER,就是置BitsRegister LPTIM\_CR 的 ENABLE.
5. 设置Register LPTIM\_ARR 和 LPTIM\_CMP 的值.
6. Enable连续计数功能,通过置BitsRegister LPTIM\_CR 的 CNTSTRT 实现.

### 20.10 支持单脉冲、Set-once、Timeout 模式输出

单脉冲模式下,counter未计数时,检测到第一个触发信号,则计数Enable置Bits,若计数到ARR或 enable 清零或模块复Bits,则计数Enable清零,计数过程中的触发信号将会被忽略,如下Figure:

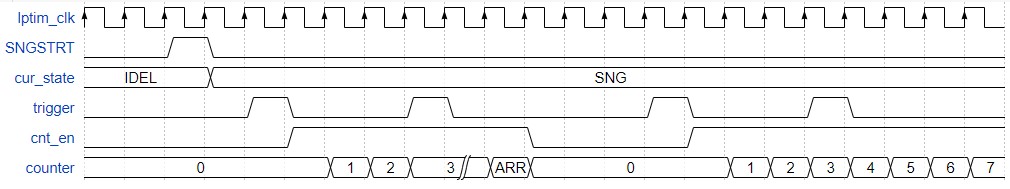
**Figure**

**20**

**-**

**3**

单脉冲计数



单脉冲模式通过配置Register LPTIM\_CFGR 的WAVE 为 0 以及Register LPTIM\_CR 的 SNGSTRT 为 1 实现.

Set-once 模式下,检测到第一个触发信号后,计数Enable置Bits,若计数到 ARR,则计数Enable清零,计数过程中的触发信号将会被屏蔽,屏蔽信号通过 mask 实现,即检测到第一个触发信号后,mask 有效,屏蔽之后的所有触发信号,如下Figure:

**Figure**

**20**

**-**

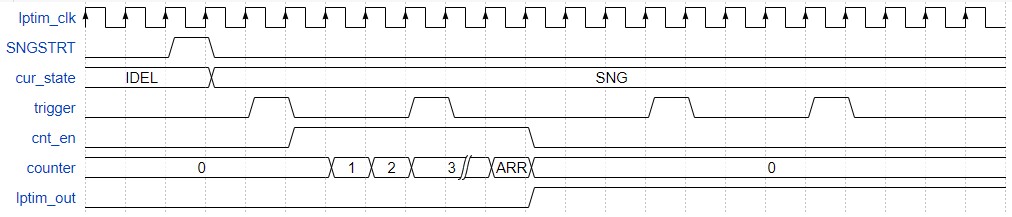
**4**

**Set**

**-**

**once**

计数



Set-once 模式通过配置Register LPTIM\_CFGR 的 WAVE 为 1 以及Register LPTIM\_CR 的

SNGSTRT 为 1 实现.

Timeout 模式与连续计数模式类似,一旦被触发,计数Enable一直有效,区别是,计数过程中的触发信号会让counter从 0 开始重新计数,且输出波形也会被清除,如下Figure:

**Figure**

**20**

**-**

**5**

**Timeout**

计数



Timeout模式通过配置RegisterLPTIM\_CFGR的WAVE为0以及RegisterLPTIM\_CR的CNTSTRT 为 1 实现.

### 20.11 正交编码

LPTIMER 支持正交编码计数功能,可以通过 IN1和 IN2输入正交信号,进行计数和方向检测.编码模式共有三种,仅在上升沿计数、下降沿计数以及双边沿计数,编码模式的Enable通过Register LPTIM\_CFGR 的 ENC 控制,编码模式的边沿控制通过Register LPTIM\_CFGR 的 CKPOL 来实现.在此功能下,两个通道输入可以配置数字滤波功能,滤波Enable通过Register LPTIM\_CFGR 的 CKFLT\_ENABLE 控制,滤波值通过Register LPTIM\_CFGR 的 CKFLT 进行配置.通过两个通道信号的组合,可以产生计数Enable和方向控制信号,控制counter加减.具体的组合方式见下Table:

#### Table 20-3 正交编码通道信号

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| 编码模式 | IN1/IN2电平 | IN1 | | IN2 | |
| 上升沿 | 下降沿 | 上升沿 | 下降沿 |
| 上升沿计数 | 高电平 | 向下计数 | - | 向上计数 | - |
| 低电平 | 向上计数 | - | 向下计数 | - |
| 下降沿计数 | 高电平 | - | 向上计数 | - | 向下计数 |
| 低电平 | - | 向下计数 | - | 向上计数 |
| 双沿计数 | 高电平 | 向下计数 | 向上计数 | 向上计数 | 向下计数 |
| 低电平 | 向上计数 | 向下计数 | 向下计数 | 向上计数 |

IN1 和 IN2 输入信号频率必须小于 LPTIMER clock 频率的 1/4.

### 20.12 支持 DEBUG 模式控制

LPTIMER 可由软件配置 debug 下是否停止计数,通过 SYSCFG 的 CR2 Register来实现 LPTIMER0 和 LPTIMER1 的 DEBUG 模式计数控制,如果Enable该功能,则进入系统 debug 模式时,LPTIMER 停止计数（counter不会被初始化）.

### 20.13 唤醒信号

LPTIMER 有 6 种唤醒信号输出,分别是,

* **通道输出信号**,此时通道输出将作为唤醒信号输出.
* **匹配事件（CMPM）**,此时counter与Register LPTIM\_CMP 的匹配事件将作为唤醒信号输出.
* **溢出事件（ARRM）**,此时 overflow 事件将作为唤醒信号输出.
* **触发事件（EXTTRIG）**,此时有效的触发事件将作为唤醒信号输出.
* **DOWN 事件**,若计数方向由向上计数变为向下计数,DOWN 事件会置Bits,此时 DOWN 事件会作为唤醒信号输出.
* **UP 事件**,若计数方向由向下计数变为向上计数,UP 事件会置Bits,此时 UP 事件会作为唤醒信号输出.

以上唤醒信号除了通道输出信号,均为 LPTIM\_ISR Register的标志Bits,且有独立的EnableBits,EnableBits分别为Register LPTIM\_CFGR 的 *OUT\_WKUP\_EN*、*CMPM\_WKUP\_EN*、*ARRM\_WKUP\_ EN*、*EXTTRIG\_WKUP\_EN*、*DOWN\_WKUP\_EN*、*UP\_WKUP\_EN* 比特Bits,唤醒信号与相应EnableBits是 AND 的关系,各Wakeup Source之间是 OR 的关系.

### 20.14 interrupt信号

LPTIMER 的interrupt信号如下:

**Table 20-4 LPTIMER interrupt信号**

|  |  |
| --- | --- |
| interrupt名称 | Description |
| DOWN interrupt | 编码模式下,Table示计数方向由向上变为向下 |
| UP interrupt | 编码模式下,Table示计数方向由向下变为向上 |
| ARROK interrupt | Table示 ARR 值加载完成 |
| CMPOK interrupt | Table示 CMP 值加载完成 |
| EXTTRIG interrupt | Table示检测到有效触发边沿 |
| ARRM interrupt | Table示counter值到达 ARR |
| CMPM interrupt | Table示counter值与 CMP 匹配 |

上述interrupt的Enable通过配置Register LPTIM\_IER 实现,所有interrupt的interrupt状态都可以通过Register

LPTIM\_SR1 获得.

### 20.15 LPTIMER 相关RegisterDescription

LPTIMER0 基地址:0x4000D000

LPTIMER1 基地址:0x4000D800

**Table 20-5 LPTIMER Register Summary**

|  |  |  |
| --- | --- | --- |
| Register | Offset | Description |
| LPTIM\_ISR | 0x00 | 状态Register |
| LPTIM\_ICR | 0x04 | 状态清除Register |
| LPTIM\_IER | 0x08 | interruptEnableRegister |
| LPTIM\_CFGR | 0x0c | 配置Register,该Register需在 LPTIM\_CR Register的 ENABLE 清零时修改 |
| LPTIM\_CR | 0x10 | Control Register |
| LPTIM\_CMP | 0x14 | 比较Register |
| LPTIM\_ARR | 0x18 | 重装载Register |
| LPTIM\_CNT | 0x1c | counterRegister |
| LPTIM\_CSR | 0x20 | 清除状态标志Register,Table示使用Register LPTIM\_ICR 清除 LPTIM\_ISR 某些状态Bits时的是否清除完成标志 |
| LPTIM\_SR1 | 0x24 | interrupt标志Register,interrupt标志Bits会被Register LPTIM\_ICR 立即清零 |

#### 20.15.1 LPTIM\_ISR

Offset:0x00

Reset Value:0x00000180

|  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| **31-9** | **8** | **7** | **6** | **5** | **4** | **3** | **2** | **1** | **0** |
| RESERVED | CROK | CFGROK | DOWN | UP | ARROK | CMPOK | EXTTRIG | ARRM | CMPM |
| r-0h | r-1h | r-1h | r-0h | r-0h | r-0h | r-0h | r-0h | r-0h | r-0h |

**Bits 31-9 RESERVED:**保留.

**Bits 8 CROK:**上一次对Register LPTIM\_CR 的写操作状态.该Bits由硬件控制,写操作前需要检查上一次写操作是否完成.

* 0:正在进行写操作
* 1:上一次对 LPTIM\_CR 的写操作已经完成

**Bits 7 CFGROK:**上一次对 LPTIM\_CFGR 的写操作状态.该Bits由硬件控制,写操作前需要检查上一次写操作是否完成.

* 0:正在进行写操作
* 1:上一次对 LPTIM\_CFGR 的写操作已经完成

**Bits 6 DOWN:**编码模式下计数方向由向上变为向下.

* 0:计数方向未发生由上向下的变化
* 1:计数方向由向上变为向下

可以通过写 LPTIM\_ICR Register清零,但是需要时间同步清零脉冲,因此无法立即清除.

**Bits 5 UP:**编码模式下计数方向由向下变为向上.

* 0:计数方向未发生由下向上的变化
* 1:计数方向由向下变为向上

可以通过写 LPTIM\_ICR Register清零,但是需要时间同步清零脉冲,因此无法立即清除.

**Bits 4 ARROK:**ARR 值加载状态.

* 0:未加载完成
* 1:加载完成

可以通过写 LPTIM\_ICR Register清零,但是需要时间同步清零脉冲,因此无法立即清除.

**Bits 3 CMPOK:**CMP 值加载状态.

* 0:未加载完成
* 1:加载完成

可以通过写 LPTIM\_ICR Register清零,但是需要时间同步清零脉冲,因此无法立即清除.

**Bits 2 EXTTRIG:**是否检测到有效触发边沿.

* 0:未检测到有效触发边沿
* 1:检测到有效触发边沿

可以通过写 LPTIM\_ICR Register清零,但是需要时间同步清零脉冲,因此无法立即清除.

**Bits 1 ARRM:**counter值是否到达 ARR 值.

* 0:counter值未到达 ARR
* 1:counter值到达 ARR

可以通过写 LPTIM\_ICR Register清零,但是需要时间同步清零脉冲,因此无法立即清除.

**Bits 0 CMPM:**counter值与 CMP 值匹配状态.

* 0:counter值与 CMP 值未匹配
* 1:counter值与 CMP 值匹配

可以通过写 LPTIM\_ICR Register清零,但是需要时间同步清零脉冲,因此无法立即清除.

#### 20.15.2 LPTIM\_ICR

Offset:0x04

Reset Value:0x00000000

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| **31-7** | **6** | **5** | **4** | **3** | **2** | **1** | **0** |
| RESERVED | DOWNCF | UPCF | ARROKCF | CMPOKCF | EXTTRIGCF | ARRMCF | CMPMCF |
| w-0h | w-0h | w-0h | w-0h | w-0h | w-0h | w-0h | w-0h |

**Bits 31-7 RESERVED:**保留.

**Bits 6 DOWNCF:**清除 DOWN 标志Bits.软件写 1 清除标记Bits,该Bits由硬件清零.

* 0:无操作
* 1:清除操作

**Bits 5 UPCF:**清除 UP 标志Bits.软件写 1 清除标记Bits,该Bits由硬件清零.

* 0:无操作
* 1:清除操作

**Bits 4 ARROKCF:**清除 ARROK 标志Bits.软件写 1 清除标记Bits,该Bits由硬件清零.

* 0:无操作
* 1:清除操作

**Bits 3 CMPOKCF:**清除 CMPOK 标志Bits.软件写 1 清除标记Bits,该Bits由硬件清零.

* 0:无操作
* 1:清除操作

**Bits 2 EXTTRIGCF:**清除 EXTTRIG 标志Bits.软件写 1 清除标记Bits,该Bits由硬件清零.

* 0:无操作
* 1:清除操作

**Bits 1 ARRMCF:**清除 ARRM 标志Bits.软件写 1 清除标记Bits,该Bits由硬件清零.

* 0:无操作
* 1:清除操作

**Bits 0 CMPMCF:**清除 CMPM 标志Bits.软件写 1 清除标记Bits,该Bits由硬件清零.

* 0:无操作
* 1:清除操作

##### 20.15.3 LPTIM\_IER

Offset:0x08 Reset Value:0x00000000

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| **31-7** | **6** | **5** | **4** | **3** | **2** | **1** | **0** |
| RESERVED | DOWNIE | UPIE | ARROKIE | CMPOKIE | EXTTRIGIE | ARRMIE | CMPMIE |
| rw-0h | rw-0h | rw-0h | rw-0h | rw-0h | rw-0h | rw-0h | rw-0h |

**Bits 31-7 RESERVED:**保留.**Bits 6 DOWNIE:**DOWN interruptEnable.

* 0:禁用interrupt
* 1:Enableinterrupt**Bits 5 UPIE:**UP interruptEnable.
* 0:禁用interrupt
* 1:Enableinterrupt**Bits 4 ARROKIE:**ARROK interruptEnable.
* 0:禁用interrupt
* 1:Enableinterrupt**Bits 3 CMPOKIE:**CMPOK interruptEnable.
* 0:禁用interrupt
* 1:Enableinterrupt**Bits 2 EXTTRIGIE:**EXTTRIG interruptEnable.
* 0:禁用interrupt
* 1:Enableinterrupt**Bits 1 ARRMIE:**ARRM interruptEnable.
* 0:禁用interrupt
* 1:Enableinterrupt**Bits 0 CMPMIE:**CMPM interruptEnable.
* 0:禁用interrupt
* 1:Enableinterrupt

#### 20.15.4 LPTIM\_CFGR

Offset:0x0c Reset Value:0x00000000

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **31** | **30** | **29** | **28** | **27** | **26** |
| RESERVED | OUT\_WKUP\_  EN | DOWN\_WKUP  \_EN | UP\_WKUP\_E  N | EXTTRIG\_WK  UP\_EN | ARRM\_WKUP  \_EN |
| rw-0h | rw-0h | rw-0h | rw-0h | rw-0h | rw-0h |
| **25** | **24** | **23** | **22** | **21** | **20** |
| CMPM\_WKUP  \_EN | ENC | COUNTMODE | PRELOAD | WAVPOL | WAVE |
| rw-0h | rw-0h | rw-0h | rw-0h | rw-0h | rw-0h |
| **19** | **18-17** | **16** | **15-13** | **12** | **11-9** |
| TIMEOUT | TRIGEN | RESERVED | TRIGSEL | RESERVED | PRESC |
| rw-0h | rw-0h | rw-0h | rw-0h | rw-0h | rw-0h |
| **8** | **7-6** | **5** | **4-3** | **2-1** | **0** |
| TRGLT\_ENAB  LE | TRGFLT | CKFLT\_ENAB  LE | CKFLT | CKPOL | RESERVED |
| rw-0h | rw-0h | rw-0h | rw-0h | rw-0h | rw-0h |

**Bits 31 RESERVED:**保留.

**Bits 30 OUT\_WKUP\_EN:**LPTIM\_OUT 唤醒Enable.

* 0:LPTIM\_OUT 不能触发唤醒信号
* 1:LPTIM\_OUT 可以触发唤醒信号**Bits 29 DOWN\_WKUP\_EN:**DOWN 事件唤醒Enable.
* 0:DOWN 事件不能触发唤醒信号
* 1:DOWN 事件可以触发唤醒信号**Bits 28 UP\_WKUP\_EN:**UP 事件唤醒Enable.
* 0:UP 事件不能触发唤醒信号
* 1:UP 事件可以触发唤醒信号**Bits 27 EXTTRIG\_WKUP\_EN:**外部触发事件唤醒Enable.
* 0:外部触发事件不能触发唤醒信号
* 1:外部触发事件可以触发唤醒信号**Bits 26 ARRM\_WKUP\_EN:**计数溢出事件唤醒Enable（ENC 模式除外）.
* 0:计数溢出不能触发唤醒信号
* 1:计数溢出事件触发唤醒信号**Bits 25 CMPM\_WKUP\_EN:**计数匹配事件唤醒Enable.
* 0:计数匹配不能触发唤醒信号
* 1:计数匹配事件触发唤醒信号**Bits 24 ENC:**编码模式Enable.
* 0:禁用编码模式
* 1:Enable编码模式**Bits 23 COUNTMODE:**计数模式选择.
* 0:counter由内部clock 控制
* 1:counter由外部clock 控制**Bits 22 PRELOAD:**Register缓存Enable.
* 0:ARR 和 CMP 直接由软件操作
* 1:ARR 和 CMP 由更新事件更新**Bits 21 WAVPOL:**输出波形极性.
* 0:输出不反相
* 1:输出反相**Bits 20 WAVE:**波形形状.
* 0:禁用 Set-once,选择 PWM 或单脉冲模式
* 1:Enable Set-once 模式**Bits 19 TIMEOUT:**Timeout 模式Enable.
* 0:禁用 Timeout 模式
* 1:Enable Timeout 模式**Bits 18-17 TRIGEN:**外部触发Enable及极性选择.
* 00:软件触发
* 01:外部触发上升沿有效
* 10:外部触发下降沿有效
* 11:外部触发双沿有效**Bits 16 RESERVED:**保留.

**Bits 15-13 TRIGSEL:**外部触发源选择.

* 000:lptim\_ext\_trig0
* 001:lptim\_ext\_trig1
* 010:lptim\_ext\_trig2
* 011:lptim\_ext\_trig3
* 100:lptim\_ext\_trig4
* 101:lptim\_ext\_trig5
* 110:lptim\_ext\_trig6
* 111:lptim\_ext\_trig7 **Bits 12 RESERVED:**保留.**Bits 11-9 PRESC:**clock 分频.
* 000:/1
* 001:/2
* 010:/4
* 011:/8
* 100:/16
* 101:/32
* 110:/64
* 111:/128

**Bits 8 TRGLT\_ENABLE:**触发输入滤波器Enable,必须先配置滤波器长度,再Enable.

* 0:禁用触发输入滤波器
* 1:Enable触发输入滤波器

**Bits 7-6 TRGFLT:**触发输入滤波器配置.

* 00:无操作
* 01:Enable滤波器,滤波器长度 N=2
* 10:Enable滤波器,滤波器长度 N=4
* 11:Enable滤波器,滤波器长度 N=8

**Bits 5 CKFLT\_ENABLE:**外部clock 滤波器Enable,必须先配置滤波器长度,再Enable.

* 0:禁用外部clock 滤波器
* 1:Enable外部clock 滤波器

**Bits 4-3 CKFLT:**外部clock 滤波器配置.

* 00:无操作
* 01:Enable滤波器,滤波器长度 N=2
* 10:Enable滤波器,滤波器长度 N=4
* 11:Enable滤波器,滤波器长度 N=8 **Bits 2-1 CKPOL:**Encoder 模式控制.
* 00:选择 Encoder 模式 1,上升沿计数
* 01:选择 Encoder 模式 2,下降沿计数
* 10:选择 Encoder 模式 3,双沿计数
* 11:保留

**Bits 0 RESERVED:**保留.

#### 20.15.5 LPTIM\_CR

Offset:0x10

Reset Value:0x00000000

|  |  |  |  |
| --- | --- | --- | --- |
| **31-3** | **2** | **1** | **0** |
| RESERVED | CNTSTRT | SNGSTRT | ENABLE |
| rw-0h | rw-0h | rw-0h | rw-0h |

**Bits 31-7 RESERVED:**保留.

**Bits 2 CNTSTRT:**连续计数模式Enable.

* 0:Disable
* 1:Enable连续计数模式,写 1 开始连续计数模式,若在连续计数模式过程中置Bits SNGSTRT,则在下一次计数到 ARR 时停止计数（切换到单次计数模式）.该比特Bits需在 ENABLE 置Bits后修改.

**Bits 1 SNGSTRT:**单次计数模式Enable.

* 0:Disable
* 1:Enable单次计数模式,写 1 开始单次计数模式,若在单次计数模式过程中置Bits CNTSTRT,则在下一次计数到 ARR 时继续计数（切换到连续计数模式）.该比特Bits需在 ENABLE 置Bits后修改.

**Bits 0 ENABLE:**LPTIMER Enable.

* 0:禁用 LPTIMER
* 1:Enable LPTIMER

#### 20.15.6 LPTIM\_CMP

Offset:0x14 Reset Value:0x00000000

|  |  |
| --- | --- |
| **31-16** | **15-0** |
| RESERVED | CMP |
| rw-0h | rw-0h |

**Bits 31-16 RESERVED:**保留.

**Bits 15-0 CMP:**比较值,需在Register LPTIM\_CR 的 ENABLE 置Bits后才能修改.

#### 20.15.7 LPTIM\_ARR

Offset:0x18 Reset Value:0x00000001

|  |  |
| --- | --- |
| **31-16** | **15-0** |
| RESERVED | ARR |
| rw-0h | rw-0h |

**Bits 31-16 RESERVED:**保留.

**Bits 15-0 ARR:**重装载值,需在Register LPTIM\_CR 的 ENABLE 置Bits后才能修改.

#### 20.15.8 LPTIM\_CNT

Offset:0x1c Reset Value:0x00000000

|  |  |
| --- | --- |
| **31-16** | **15-0** |
| RESERVED | CNT |
| r-0h | r-0h |

**Bits 31-16 RESERVED:**保留.**Bits 15-0 CNT:**计数结果,读该值时,连续两次读到的结果一致才算有效.

#### 20.15.9 LPTIM\_CSR

Offset:0x20 Reset Value:0x0000001f

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **31-5** | **4** | **3** | **2** | **1** | **0** |
| RESERVED | DOWN\_CLR\_DONE | UP\_CLR\_DONE | EXTTRIG\_CLR  \_DONE | ARRM\_CLR  \_DONE | CMPM\_CLR  \_DONE |
| r-0h | r-0h | r-0h | r-0h | r-0h | r-0h |

**Bits 31-5 RESERVED:**保留.

**Bits 4 DOWN\_CLR\_DONE:**DOWN 清除完成.

* 0:正在清除 DOWN 标志Bits
* 1:清除成功**Bits 3 UP\_CLR\_DONE:**UP 清除完成.
* 0:正在清除 UP 标志Bits
* 1:清除成功**Bits 2 EXTTRIG\_CLR\_DONE:**EXTTRIG 清除完成.
* 0:正在清除 EXTTRIG 标志Bits
* 1:清除成功**Bits 1 ARRM\_CLR\_DONE:**ARRM 清除完成.
* 0:正在清除 ARRM 标志Bits
* 1:清除成功**Bits 0 CMPM \_CLR\_DONE:**CMPM 清除完成.
* 0:正在清除 CMPM 标志Bits
* 1:清除成功

#### 20.15.10 LPTIM\_SR1

Offset:0x24

Reset Value:0x00000000

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| **31-7** | **6** | **5** | **4** | **3** | **2** | **1** | **0** |
| RESERVED | DOWN | UP | ARROK | CMPOK | EXTTRIG | ARRM | CMPM |
| r-0h | r-0h | r-0h | r-0h | r-0h | r-0h | r-0h | r-0h |

**Bits 31-7 RESERVED:**保留.

**Bits 6 DOWN:**编码模式下计数方向由向上变为向下.

* 0:计数方向未发生由上向下的变化
* 1:计数方向由向上变为向下

**Bits 5 UP:**编码模式下计数方向由向下变为向上.

* 0:计数方向未发生由下向上的变化
* 1:计数方向由向下变为向上

**Bits 4 ARROK:**ARR 值加载状态.

* 0:未加载完成
* 1:加载完成

**Bits 3 CMPOK:**CMP 值加载状态.

* 0:未加载完成
* 1:加载完成

**Bits 2 EXTTRIG:**是否检测到有效触发边沿.

* 0:未检测到有效触发边沿
* 1:检测到有效触发边沿**Bits 1 ARRM:**counter值是否到达 ARR 值.
* 0:counter值未到达 ARR
* 1:counter值到达 ARR

**Bits 0 CMPM:**counter值与 CMP 值匹配状态.

* 0:counter值与 CMP 值未匹配
* 1:counter值与 CMP 值匹配

## 21. 直接存储器访问控制器 (DMA)

### 21.1 Introduction

DMA 支持外设到外设,外设到 memory,memory 到外设,memory 到 memory 这四种数据搬移方式,支持数据Bits宽为 8 Bits、16 Bits或 32 Bits,并支持数据的 Auto-reloading 以及数据的链Table（LLI）.共有两个 DMA,分别为 DMA0 和 DMA1,每个 DMA 有 4 个 channel.两个 DMA 相互独立,可以同时工作,每个 DMA 中的 4 个 channel 也是相互独立的,可以同时运行.

### 21.2 Main features

* 传输数据长度的配置
* 支持数据搬移方式的配置
* 支持 Auto-reloading
* 支持 LLI

### 21.3 配置传输数据长度

DMA 可以传输多个 block 的数据,传输每个 block 的数据时先以 burst 方式传送,后面有不够 burst 的数据长度的数据时再以 single 方式发送.外设的数据传输如下Figure:

 **Figure 21-1 数据传输**

DMA 的源和目的数据Bits宽通过 *DMA\_CTLx* Register的 *SRC\_TR\_WIDTH* 和 *DST\_TR\_WIDTH* Bits段进行配置（x 为 0、1、2 或 3）,此Bits段值为 000 时Table示 8bit,为 001 Table示 16bit,为 002 Table示 32bit.

DMA 的源和目的 burst 数据长度通过 *DMA\_CTLx* Register的 *SRC\_MSIZE* 和 *DEST\_MSIZE*

Bits段进行配置,此Bits段值为 000 时Table示 1,为 001 Table示 4,为 002 Table示 8,那么转化为 Bytes 就是 *SRC\_MSIZE (DEST\_MSIZE) \* (*数据Bits宽的*bit*数 */ 8)*.DMA 的 burst 数据长度 Bytes 需要与外设的输入或输出 FIFO 长度一致,否则可能导致数据丢失.

DMA 的 block size 通过 *DMA\_CTLx* Register的 *BLOCK\_TS* Bits进行配置,最多为 12 个 bit,那么 block size 最大为 4095,转换为 Bytes 时为 *BLOCK\_TS \* (*数据Bits宽的*bit*数 */ 8)*.

### 21.4 数据搬移方式

DMA 支持外设到外设,外设到 memory,memory 到外设,memory 到 memory 四种数据搬移方式.外设到外设指数据的源和目的都为外设；外设到 memory 指源为外设,目的为 memory； memory 到外设指源为 memory,目的为外设；memory 到 memory 指源和目的都为 memory.数据搬移方式通过 *DMA\_CTLx* Register的 *TT\_FC* Bits段进行配置.除了 memory 到 memory 的搬移方式,其他几种方式都要配置外设与 DMA 之间的握手信号即 handshake.外设的 handshake 的值如下Table所示:

**Table 21-1 Handshake 值**

|  |  |  |
| --- | --- | --- |
| Handshake值 | 外设信号 | 外设信号Description |
| 4 | lorac\_tx | LORA 的 tx |
| 5 | lorac\_rx | LORA 的 rx |
| 6 | dacctrl | DAC |
| 7 | adcctrl | ADC |
| 10 | i2c2\_tx | I2C2 的 tx |
| 11 | i2c2\_rx | I2C2 的 rx |
| 12 | i2c1\_tx | I2C1 的 tx |
| 13 | i2c1\_rx | I2C1 的 rx |
| 14 | i2c0\_tx | I2C0 的 tx |
| 15 | i2c0\_rx | I2C0 的 rx |
| 16 | ssp2\_tx | SSP2 的 tx |
| 17 | ssp2\_rx | SSP2 的 rx |
| 18 | ssp1\_tx | SSP1 的 tx |
| 19 | ssp1\_rx | SSP1 的 rx |
| 20 | ssp0\_tx | SSP0 的 tx |
| 21 | ssp0\_rx | SSP0 的 rx |
| 22 | lpuart\_tx | LPUAR 的 tx |
| 23 | lpuart\_rx | LPUAR 的 rx |
| Handshake值 | 外设信号 | 外设信号Description |
| 24 | uart3\_tx | UART3 的 tx |
| 25 | uart3\_rx | UART3 的 rx |
| 26 | uart2\_tx | UART2 的 tx |
| 27 | uart2\_rx | UART2 的 rx |
| 28 | uart1\_tx | UART1 的 tx |
| 29 | uart1\_rx | UART1 的 rx |
| 30 | uart0\_tx | UART0 的 tx |
| 31 | uart0\_rx | UART0 的 rx |
| 32 | gptim0\_ch3 | GPTIMER0 的 channel3 |
| 33 | gptim0\_ch2 | GPTIMER0 的 channel2 |
| 34 | gptim0\_ch1 | GPTIMER0 的 channel1 |
| 35 | gptim0\_ch0 | GPTIMER0 的 channel0 |
| 36 | gptim0\_trg | GPTIMER0 的 trigger |
| 37 | gptim0\_up | GPTIMER0 的 update |
| 38 | Gptim1\_ch3 | GPTIMER1 的 channel3 |
| 39 | Gptim1\_ch2 | GPTIMER1 的 channel2 |
| 40 | Gptim1\_ch1 | GPTIMER1 的 channel1 |
| 41 | Gptim1\_ch0 | GPTIMER1 的 channel0 |
| 42 | Gptim1\_trg | GPTIMER1 的 trigger |
| 43 | Gptim1\_up | GPTIMER1 的 update |
| 44 | gptim2\_ch1 | GPTIMER2 的 channel1 |
| 45 | gptim2\_ch0 | GPTIMER2 的 channel0 |
| 46 | gptim2\_trg | GPTIMER2 的 trigger |
| 47 | gptim2\_up | GPTIMER2 的 update |
| 48 | Gptim3\_ch1 | GPTIMER3 的 channel1 |
| 49 | Gptim3\_ch0 | GPTIMER3 的 channel0 |
| 50 | Gptim3\_trg | GPTIMER3 的 trigger |
| 51 | Gptim3\_up | GPTIMER3 的 update |
| 52 | basictim1\_up | BSTIMER1 的 update |
| 53 | basictim0\_up | BSTIMER0 的 update |

### 21.5 LLI

当有多块不连续的memory的数据需要搬移到外设或memory时,可以使用LLI（即链Table方式）,如下Figure所示:



**Figure 21-2 LLI 链Table**

LLI(0)、LLI(1)Table示配置 block0、block1 的信息,包括源目的地址、数据Bits宽、burst 长度和 block 长度.LLPx Table示当前 block 指向下个 block 的地址,第一个 block 的 LLP 指向第二个 block 的地址即 LLI(1)的首地址,依次类推,最后一个 block 的 LLP 为 0.每个 block 长度是可以不一样的,并且 memory 的首地址也是不一样的.

#### 21.6 Auto-reloading

Auto-reloading 指 block 中的 memory 的数据被搬完或 memory 都被写入完成,然后重新从此 memory 的起始地址开始搬送或写入数据,如此循环往复,直至把所用的 DMA 的 channel 去Enable后才会停止.DMA 的源和目的都可以使用 Auto-reloading 功能,只要其为 memory 就可以.

### 21.7 interrupt

DMA 的interrupt信号如下:

**Table 21-2 DMA interrupt信号**

|  |  |
| --- | --- |
| interrupt名称 | Description |
| DMA 块传输完成interrupt | DMA 块传输完成后产生的interrupt |
| DMA 目的端处理完成interrupt | DMA 目的端处理完成后产生的interrupt |
| DMA 源端处理完成interrupt | DMA 源端处理完成后产生的interrupt |
| DMA 传输出错interrupt | DMA 传输过程中产生错误时发生的interrupt |
| DMA 完全传输完成interrupt | DMA 完全传输完成后产生的interrupt |

通过配置 *DMA\_MaskBlock*,*DMA\_MaskDstTran*,*DMA\_MaskSrcTran*,*DMA\_MaskErr* 和 *DMA\_MaskTfr* Register来Enable上述interrupt.

通过 *DMA\_StatusBlock*,*DMA\_StatusDstTran*,*DMA\_StatusSrcTran*,*DMA\_StatusErr* 和 *DMA\_StatusTfr* Register可以获得所有interrupt的状态.

通过配置 *DMA\_ClearBlock*、*DMA\_ClearDstTran*、*DMA\_ClearSrcTran*、*DMA\_ClearErr* 和 *DMA\_ClearTfr* Register来清除interrupt状态.

#### 21.8 DMA 相关RegisterDescription

DMA0 基地址:0x40023000

DMA1 基地址:0x40024000

**Table 21-3 DMA Register Summary**

|  |  |  |
| --- | --- | --- |
| Register | Offset | Description |
| DMA\_SARx | 0x00 | 源地址Register,x Table示 channel 0、1、2、3,Correspond to 的Offset分  别为 0x00、0x58、0xb0、0x108 |
| DMA\_DARx | 0x08 | 目的地址Register,x Table示 channel 0、1、2、3,Correspond to 的Offset分别为 0x08、0x60、0xb8、0x110 |
| DMA\_LLPx | 0x10 | 链Table指针Register,x Table示 channel 0、1、2、3,Correspond to 的Offset分别为 0x10、0x68、0xc0、0x118 |
| DMA\_CTLx | 0x18 | 通道Control Register,x Table示 channel 0、1、2、3,Correspond to 的Offset分别为 0x18、0x70、0xc8、0x120 |
| DMA\_CFGx | 0x40 | 通道配置Register,x Table示 channel 0、1、2、3,Correspond to 的Offset  分别为 0x40、0x98、0xf0、0x148 |
| DMA\_StatusTfr | 0x2e8 | DMA 完全传输完成interrupt状态Register |
| DMA\_StatusBlock | 0x2f0 | DMA 块传输完成interrupt状态Register |
| DMA\_StatusSrcTran | 0x2f8 | DMA 源端处理完成interrupt状态Register |
| DMA\_StatusDstTran | 0x300 | DMA 目的端处理完成interrupt状态Register |
| DMA\_StatusErr | 0x308 | DMA 传输出错interrupt状态Register |
| DMA\_MaskTfr | 0x310 | DMA 完全传输完成interruptEnableRegister |
| DMA\_MaskBlock | 0x318 | DMA 块传输完成interruptEnableRegister |
| DMA\_MaskSrcTran | 0x320 | DMA 源端处理完成interruptEnableRegister |
| DMA\_MaskDstTran | 0x328 | DMA 目的端处理完成interruptEnableRegister |
| DMA\_MaskErr | 0x330 | DMA 传输出错interruptEnableRegister |
| DMA\_ClearTfr | 0x338 | DMA 完全传输完成interrupt状态清除Register |
| DMA\_ClearBlock | 0x340 | DMA 块传输完成interrupt状态清除Register |
| DMA\_ClearSrcTran | 0x348 | DMA 源端处理完成interrupt状态清除Register |
| DMA\_ClearDstTran | 0x350 | DMA 目的端处理完成interrupt状态清除Register |
| DMA\_ClearErr | 0x358 | DMA 传输出错interrupt状态清除Register |
| DMA\_DmaCfgReg | 0x398 | DMA EnableRegister |
| DMA\_ChEnReg | 0x3a0 | DMA channel EnableRegister |

##### 21.8.1 DMA\_SARx

Offset:0x00、0x58、0xb0、0x108 Reset Value:0x0000000000000000

|  |  |
| --- | --- |
| **63-32** | **31-0** |
| RESERVED | SAR |
| r-0h | rw-0h |

**Bits 63-32 RESERVED:**保留.

**Bits 31-0 SAR:**DMA 源地址Register.

##### 21.8.2 DMA\_DARx

Offset:0x08、0x60、0xb8、0x110 Reset Value:0x0000000000000000

|  |  |
| --- | --- |
| **63-32** | **31-0** |
| RESERVED | DAR |
| r-0h | rw-0h |

**Bits 63-32 RESERVED:**保留.

**Bits 31-0 DAR:**DMA 目的地址Register.

##### 21.8.3 DMA\_LLPx

Offset:0x10、0x68、0xc0、0x118 Reset Value:0x0000000000000000

|  |  |
| --- | --- |
| **63-32** | **31-0** |
| RESERVED | LOC |
| r-0h | rw-0h |

**Bits 63-32 RESERVED:**保留.

**Bits 31-0 LOC:**下一个 LLI 链Table的首地址.

##### 21.8.4 DMA\_CTLx

Offset:0x18、0x70、0xc8、0x120 Reset Value:0x0000000200308801

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| **63-45** |  | **44** | **43-32** | | **31-29** | **28** |
| RESERVED |  | DONE | BLOCK\_TS | | RESERVED | LLP\_SRC\_EN |
| r-0h |  | rw-0h | rw-2h | | rw-0h | rw-0h |
| **27** |  | **26-25** | **24-23** | | **22-20** | **19** |
| LLP\_DST\_EN |  | SMS | DMS | | TT\_FC | RESERVED |
| rw-0h |  | rw-0h | rw-0h | | rw-3h | rw-0h |
| **18** |  | **17** | **16-14** | | **13-11** | **10-9** |
| DST\_SCATTER\_E | N | SRC\_GATHER\_EN | SRC\_MSIZE | | DEST\_MSIZE | SINC |
| rw-0h |  | rw-0h | rw-1h | | rw-1h | rw-0h |
| **8-7** | **6-4** | |  | **3-1** | | **0** |
| DINC | SRC\_TR\_WIDTH | |  | DST\_TR\_WIDTH | | INT\_EN |
| rw-0h | rw-0h | |  | rw-0h | | rw-1h |

**Bits 63-45 RESERVED:**保留.

**Bits 44 DONE:**LLI 链Table中一个 block 是否传输完成.

* 0:完成
* 1:未完成**Bits 43-32 BLOCK\_TS:**block 的长度.**Bits 31-29 RESERVED:**保留.

**Bits 28 LLP\_SRC\_EN:**DMA 源Enable LLI 链Table.

* 0:去Enable
* 1:Enable**Bits 27 LLP\_DST\_EN:**DMA 目的Enable LLI 链Table.
* 0:去Enable
* 1:Enable**Bits 26-25 SMS:**DMA 源的 AHB master 选择.
* 00:AHB master 1
* 01:AHB master 2
* 10:AHB master 3
* 11:AHB master 4

**Bits 24-23 SMS:**DMA 目的的 AHB master 选择.

* 00:AHB master 1
* 01:AHB master 2
* 10:AHB master 3
* 11:AHB master 4

**22-20 TT\_FC:**DMA 数据搬移方式选择.

* 000:DMA 流控的 Memory 到 Memory 方式
* 001:DMA 流控的 Memory 到外设方式
* 010:DMA 流控的外设到 Memory 方式
* 011:DMA 流控的外设到外设方式
* 其它值:无效**Bits 19 RESERVED:**保留.

**Bits 18 DST\_SCATTER\_EN:**DMA 目的Enable Scatter.

* 0:去Enable
* 1:Enable**Bits 17 SRC\_GATHER\_EN:**DMA 源Enable Gather.
* 0:去Enable
* 1:Enable**Bits 16-14 SRC\_MSIZE:**DMA 源的 Burst 长度配置.
* 000:1
* 001:4
* 010:8
* 其它值:无效**Bits 13-11 DEST\_MSIZE:**DMA 目的的 Burst 长度配置.
* 000:1
* 001:4
* 010:8
* 其它值:无效**Bits 10-9 SINC:**DMA 源地址控制.
* 00:递增
* 01:递减
* 10:不变化
* 11:不变化**Bits 8-7 DINC:**DMA 目的地址控制.
* 00:递增
* 01:递减
* 10:不变化
* 11:不变化**Bits 6-4 SRC\_TR\_WIDTH:**DMA 源数据Bits宽配置.
* 000:8bit
* 001:16bit
* 010:32bit
* 其它值:无效**Bits 3-1 DST\_TR\_WIDTH:**DMA 目的数据Bits宽配置.
* 000:8bit
* 001:16bit
* 010:32bit
* 其它值:无效**Bits 0 INT\_EN:**DMA interruptEnable.
* 0:去Enable
* 1:Enable

##### 21.8.5 DMA\_CFGx

Offset:0x40、0x98、0xf0、0x148 Reset Value:0x0000000400020e00

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **63-47** | **46-43** | **42-39** | **38** | **37** | **36-34** |
| RESERVED | DEST\_PER | SRC\_PER | SS\_UPD\_EN | DS\_UPD\_EN | PROTCTL |
| r-0h | rw-0h | rw-0h | rw-0h | rw-0h | rw-1h |
| **33** | **32** | **31** | **30** | **29-20** | **19** |
| FIFO\_MODE | FCMODE | RELOAD\_DST | RELOAD\_SRC | RESERVED | SRC\_HS\_POL |
| rw-0h | rw-0h | rw-0h | rw-0h | r-0h | rw-0h |
| **18** | **17** | **16** | **15-14** | **13-12** | **11** |
| DST\_HS\_POL | LOCK\_B | LOCK\_CH | LOCK\_B\_L | LOCK\_CH\_L | HS\_SEL\_SRC |
| rw-0h | rw-1h | rw-0h | rw-0h | rw-0h | rw-1h |
| **10** | **9** | **8** | **7-5** | **4-0** | |
| HS\_SEL\_DST | FIFO\_EMPTY | CH\_SUSP | CH\_PRIOR | RESERVED | |
| rw-1h | r-1h | rw-0h | rw-0h | r-0h | |

**Bits 63-47 RESERVED:**保留.**Bits 46-43 DEST\_PER:**DMA 目的握手接口,有效值为 0 至 3. **Bits 42-39 SRC\_PER:**DMA 源握手接口,有效值为 0 至 3.

**Bits 38 SS\_UPD\_EN:**DMA 源状态更新Enable.

* 0:去Enable
* 1:Enable**Bits 37 DS\_UPD\_EN:**DMA 目的状态更新Enable.
* 0:去Enable
* 1:Enable**Bits 36-34 PROTCTL:**保护控制.**Bits 33 FIFO\_MODE:**FIFO 模式选择.
* 0:可以获取全部 FIFO
* 1:只能获得一半 FIFO **Bits 32 FCMODE:**源端流控模式选择.
* 0:源端的Request 发出就处理
* 1:直到目的端有Request 发生才会处理源端的Request

**31 RELOAD\_DST:**DMA 目的Enable Auto-reloading.

* 0:去Enable
* 1:Enable**Bits 30 RELOAD\_SRC:**DMA 源Enable Auto-reloading.
* 0:去Enable
* 1:Enable**Bits 29-20 RESERVED:**保留.

**Bits 19 SRC\_HS\_POL:**DMA 源握手接口信息极性.

* 0:高有效  1:低有效**Bits 18 SRC\_HS\_POL:**DMA 目的握手接口信息极性.
* 0:高有效  1:低有效**Bits 17 LOCK\_B:**总线锁定控制.
* 0:不锁定
* 1:锁定**Bits 16 LOCK\_CH:**DMA channel 锁定控制.
* 0:不锁定
* 1:锁定**Bits 15-14 LOCK\_B\_L:**总线锁定延时.
* 00:等到 DMA 传输完成
* 01:等到 block 传输完成
* 10:等到 DMA 处理完成**Bits 13-12 LOCK\_CH\_L:**DMA channel 锁定延时.
* 00:等到 DMA 传输完成
* 01:等到 block 传输完成
* 10:等到 DMA 处理完成**Bits 11 HS\_SEL\_SRC:**DMA 源握手信号选择.
* 0:硬件握手  1:软件握手**Bits 10 HS\_SEL\_DST:**DMA 目的握手信号选择.
* 0:硬件握手  1:软件握手**Bits 9 FIFO\_EMPTY:**DMA channel FIFO 是否为空.
* 0:非空
* 1:空**Bits 8 CH\_SUSP:**DMA channel FIFO 是否暂停.
* 0:非暂停
* 1:暂停

**7-5 CH\_PRIOR:**DMA channel 优先级配置,有效值为 0 至 3,0 为最低优先级,3 为最高优先级.

**Bits 4-0 RESERVED:**保留.

###### 21.8.6 DMA\_StatusTfr

Offset:0x2e8 Reset Value:0x0000000000000000

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **63-4** | **3** | **2** | **1** | **0** |
| RESERVED | CHAN3\_STATUS | CHAN2\_STATUS | CHAN1\_STATUS | CHAN0\_STATUS |
| r-0h | r-0h | r-0h | r-0h | r-0h |

**Bits 63-4 RESERVED:**保留.

**Bits 3 CHAN3\_STATUS:**DMA channel3 的传输完成状态.

* 0:未完成
* 1:完成 **Bits 2 CHAN2\_STATUS:**DMA channel2 的传输完成状态.
* 0:未完成
* 1:完成**Bits 1 CHAN1\_STATUS:**DMA channel1 的传输完成状态.
* 0:未完成
* 1:完成**Bits 0 CHAN0\_STATUS:**DMA channel0 的传输完成状态.
* 0:未完成
* 1:完成

###### 21.8.7 DMA\_StatusBlock

Offset:0x2f0 Reset Value:0x0000000000000000

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **63-4** | **3** | **2** | **1** | **0** |
| RESERVED | CHAN3\_STATUS | CHAN2\_STATUS | CHAN1\_STATUS | CHAN0\_STATUS |
| r-0h | r-0h | r-0h | r-0h | r-0h |

**Bits 63-4 RESERVED:**保留.

**Bits 3 CHAN3\_STATUS:**DMA channel3 的块传输完成状态.

* 0:未完成
* 1:完成

**2 CHAN2\_STATUS:**DMA channel2 的块传输完成状态.

* 0:未完成
* 1:完成**Bits 1 CHAN1\_STATUS:**DMA channel1 的块传输完成状态.
* 0:未完成
* 1:完成**Bits 0 CHAN0\_STATUS:**DMA channel0 的块传输完成状态.
* 0:未完成
* 1:完成

###### 21.8.8 DMA\_StatusSrcTran

Offset:0x2f8 Reset Value:0x0000000000000000

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **63-4** | **3** | **2** | **1** | **0** |
| RESERVED | CHAN3\_STATUS | CHAN2\_STATUS | CHAN1\_STATUS | CHAN0\_STATUS |
| r-0h | r-0h | r-0h | r-0h | r-0h |

**Bits 63-4 RESERVED:**保留.

**Bits 3 CHAN3\_STATUS:**DMA channel3 的源端传输完成状态.

* 0:未完成
* 1:完成 **Bits 2 CHAN2\_STATUS:**DMA channel2 的源端传输完成状态.
* 0:未完成
* 1:完成**Bits 1 CHAN1\_STATUS:**DMA channel1 的源端传输完成状态.
* 0:未完成
* 1:完成**Bits 0 CHAN0\_STATUS:**DMA channel0 的源端传输完成状态.
* 0:未完成
* 1:完成

###### 21.8.9 DMA\_StatusDstTran

Offset:0x300 Reset Value:0x0000000000000000

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **63-4** | **3** | **2** | **1** | **0** |
| RESERVED | CHAN3\_STATUS | CHAN2\_STATUS | CHAN1\_STATUS | CHAN0\_STATUS |
| r-0h | r-0h | r-0h | r-0h | r-0h |

**63-4 RESERVED:**保留.

**Bits 3 CHAN3\_STATUS:**DMA channel3 的目的端传输完成状态.

* 0:未完成
* 1:完成 **Bits 2 CHAN2\_STATUS:**DMA channel2 的目的端传输完成状态.
* 0:未完成
* 1:完成**Bits 1 CHAN1\_STATUS:**DMA channel1 的目的端传输完成状态.
* 0:未完成
* 1:完成**Bits 0 CHAN0\_STATUS:**DMA channel0 的目的端传输完成状态.
* 0:未完成
* 1:完成

###### 21.8.10 DMA\_StatusErr

Offset:0x308 Reset Value:0x0000000000000000

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **63-4** | **3** | **2** | **1** | **0** |
| RESERVED | CHAN3\_STATUS | CHAN2\_STATUS | CHAN1\_STATUS | CHAN0\_STATUS |
| r-0h | r-0h | r-0h | r-0h | r-0h |

**Bits 63-4 RESERVED:**保留.

**Bits 3 CHAN3\_STATUS:**DMA channel3 的传输错误状态.

* 0:未出错
* 1:出错 **Bits 2 CHAN2\_STATUS:**DMA channel2 的传输错误状态.
* 0:未出错
* 1:出错**Bits 1 CHAN1\_STATUS:**DMA channel1 的传输错误状态.
* 0:未出错
* 1:出错**Bits 0 CHAN0\_STATUS:**DMA channel0 的传输错误状态.
* 0:未出错
* 1:出错

###### 21.8.11 DMA\_MaskTfr

Offset:0x310



**Bits 63-12 RESERVED:**保留.

**Bits 11 INT\_MASK\_WE\_3:**DMA channel3 的传输完成interrupt掩码写Enable.

* 0:去Enable
* 1:Enable **Bits 10 INT\_MASK\_WE\_2:**DMA channel2 的传输完成interrupt掩码写Enable.
* 0:去Enable
* 1:Enable**Bits 9 INT\_MASK\_WE\_1:**DMA channel1 的传输完成interrupt掩码写Enable.
* 0:去Enable
* 1:Enable**Bits 8 INT\_MASK\_WE\_0:**DMA channel0 的传输完成interrupt掩码写Enable.
* 0:去Enable
* 1:Enable**Bits 7-4 RESERVED:**保留.

**Bits 3 INT\_MASK\_3:**DMA channel3 的传输完成interruptEnable.

* 0:去Enable
* 1:Enable **Bits 2 INT\_MASK\_2:**DMA channel2 的传输完成interruptEnable.
* 0:去Enable
* 1:Enable**Bits 1 INT\_MASK\_1:**DMA channel1 的传输完成interruptEnable.
* 0:去Enable
* 1:Enable**Bits 0 INT\_MASK\_0:**DMA channel0 的传输完成interruptEnable.
* 0:去Enable

###### 21.8.12 DMA\_MaskBlock

Offset:0x318



**Bits 63-12 RESERVED:**保留.

**Bits 11 INT\_MASK\_WE\_3:**DMA channel3 的块传输完成interrupt掩码写Enable.

* 0:去Enable
* 1:Enable **Bits 10 INT\_MASK\_WE\_2:**DMA channel2 的块传输完成interrupt掩码写Enable.
* 0:去Enable
* 1:Enable**Bits 9 INT\_MASK\_WE\_1:**DMA channel1 的块传输完成interrupt掩码写Enable.
* 0:去Enable
* 1:Enable**Bits 8 INT\_MASK\_WE\_0:**DMA channel0 的块传输完成interrupt掩码写Enable.
* 0:去Enable
* 1:Enable**Bits 7-4 RESERVED:**保留.

**Bits 3 INT\_MASK\_3:**DMA channel3 的块传输完成interruptEnable.

* 0:去Enable
* 1:Enable **Bits 2 INT\_MASK\_2:**DMA channel2 的块传输完成interruptEnable.
* 0:去Enable
* 1:Enable**Bits 1 INT\_MASK\_1:**DMA channel1 的块传输完成interruptEnable.
* 0:去Enable
* 1:Enable**Bits 0 INT\_MASK\_0:**DMA channel0 的块传输完成interruptEnable.
* 0:去Enable

###### 21.8.13 DMA\_MaskSrcTran

Offset:0x320



**Bits 63-12 RESERVED:**保留.

**Bits 11 INT\_MASK\_WE\_3:**DMA channel3 的源端传输完成interrupt掩码写Enable.

* 0:去Enable
* 1:Enable **Bits 10 INT\_MASK\_WE\_2:**DMA channel2 的源端传输完成interrupt掩码写Enable.
* 0:去Enable
* 1:Enable**Bits 9 INT\_MASK\_WE\_1:**DMA channel1 的源端传输完成interrupt掩码写Enable.
* 0:去Enable
* 1:Enable**Bits 8 INT\_MASK\_WE\_0:**DMA channel0 的源端传输完成interrupt掩码写Enable.
* 0:去Enable
* 1:Enable**Bits 7-4 RESERVED:**保留.

**Bits 3 INT\_MASK\_3:**DMA channel3 的源端传输完成interruptEnable.

* 0:去Enable
* 1:Enable **Bits 2 INT\_MASK\_2:**DMA channel2 的源端传输完成interruptEnable.
* 0:去Enable
* 1:Enable**Bits 1 INT\_MASK\_1:**DMA channel1 的源端传输完成interruptEnable.
* 0:去Enable
* 1:Enable**Bits 0 INT\_MASK\_0:**DMA channel0 的源端传输完成interruptEnable.
* 0:去Enable

###### 21.8.14 DMA\_MaskDstTran

Offset:0x328



**Bits 63-12 RESERVED:**保留.

**Bits 11 INT\_MASK\_WE\_3:**DMA channel3 的目的端传输完成interrupt掩码写Enable.

* 0:去Enable
* 1:Enable **Bits 10 INT\_MASK\_WE\_2:**DMA channel2 的目的端传输完成interrupt掩码写Enable.
* 0:去Enable
* 1:Enable**Bits 9 INT\_MASK\_WE\_1:**DMA channel1 的目的端传输完成interrupt掩码写Enable.
* 0:去Enable
* 1:Enable**Bits 8 INT\_MASK\_WE\_0:**DMA channel0 的目的端传输完成interrupt掩码写Enable.
* 0:去Enable
* 1:Enable**Bits 7-4 RESERVED:**保留.

**Bits 3 INT\_MASK\_3:**DMA channel3 的目的端传输完成interruptEnable.

* 0:去Enable
* 1:Enable **Bits 2 INT\_MASK\_2:**DMA channel2 的目的端传输完成interruptEnable.
* 0:去Enable
* 1:Enable**Bits 1 INT\_MASK\_1:**DMA channel1 的目的端传输完成interruptEnable.
* 0:去Enable
* 1:Enable**Bits 0 INT\_MASK\_0:**DMA channel0 的目的端传输完成interruptEnable.
* 0:去Enable

###### 21.8.15 DMA\_MaskErr

Offset:0x330



**Bits 63-12 RESERVED:**保留.

**Bits 11 INT\_MASK\_WE\_3:**DMA channel3 的传输出错interrupt掩码写Enable.

* 0:去Enable
* 1:Enable **Bits 10 INT\_MASK\_WE\_2:**DMA channel2 的传输出错interrupt掩码写Enable.
* 0:去Enable
* 1:Enable**Bits 9 INT\_MASK\_WE\_1:**DMA channel1 的传输出错interrupt掩码写Enable.
* 0:去Enable
* 1:Enable**Bits 8 INT\_MASK\_WE\_0:**DMA channel0 的传输出错interrupt掩码写Enable.
* 0:去Enable
* 1:Enable**Bits 7-4 RESERVED:**保留.

**Bits 3 INT\_MASK\_3:**DMA channel3 的传输出错interruptEnable.

* 0:去Enable
* 1:Enable **Bits 2 INT\_MASK\_2:**DMA channel2 的传输出错interruptEnable.
* 0:去Enable
* 1:Enable**Bits 1 INT\_MASK\_1:**DMA channel1 的传输出错interruptEnable.
* 0:去Enable
* 1:Enable**Bits 0 INT\_MASK\_0:**DMA channel0 的传输出错interruptEnable.
* 0:去Enable

###### 21.8.16 DMA\_ClearTfr

Offset:0x338 Reset Value:0x0000000000000000

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **63-4** | **3** | **2** | **1** | **0** |
| RESERVED | CHAN3\_CLEAR | CHAN2\_CLEAR | CHAN1\_CLEAR | CHAN0\_CLEAR |
| r-0h | w-0h | w-0h | w-0h | w-0h |

**Bits 63-4 RESERVED:**保留.

**Bits 3 CHAN3\_CLEAR:**DMA channel3 的传输完成状态清除.

* 0:不操作
* 1:清除 **Bits 2 CHAN2\_CLEAR:**DMA channel2 的传输完成状态清除.
* 0:不操作
* 1:清除**Bits 1 CHAN1\_CLEAR:**DMA channel1 的传输完成状态清除.
* 0:不操作
* 1:清除**Bits 0 CHAN0\_CLEAR:**DMA channel0 的传输完成状态清除.
* 0:不操作
* 1:清除

###### 21.8.17 DMA\_ClearBlock

Offset:0x340 Reset Value:0x0000000000000000

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **63-4** | **3** | **2** | **1** | **0** |
| RESERVED | CHAN3\_CLEAR | CHAN2\_CLEAR | CHAN1\_CLEAR | CHAN0\_CLEAR |
| r-0h | w-0h | w-0h | w-0h | w-0h |

**Bits 63-4 RESERVED:**保留.

**Bits 3 CHAN3\_CLEAR:**DMA channel3 的块传输完成状态清除.

* 0:不操作
* 1:清除 **Bits 2 CHAN2\_CLEAR:**DMA channel2 的块传输完成状态清除.
* 0:不操作
* 1:清除**Bits 1 CHAN1\_CLEAR:**DMA channel1 的块传输完成状态清除.
* 0:不操作
* 1:清除

**Bits 0 CHAN0\_CLEAR:**DMA channel0 的块传输完成状态清除.

* 0:不操作
* 1:清除

###### 21.8.18 DMA\_ClearSrcTran

Offset:0x348 Reset Value:0x0000000000000000

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **63-4** | **3** | **2** | **1** | **0** |
| RESERVED | CHAN3\_CLEAR | CHAN2\_CLEAR | CHAN1\_CLEAR | CHAN0\_CLEAR |
| r-0h | w-0h | w-0h | w-0h | w-0h |

**Bits 63-4 RESERVED:**保留.

**Bits 3 CHAN3\_CLEAR:**DMA channel3 的源端传输完成状态清除.

* 0:不操作
* 1:清除 **Bits 2 CHAN2\_CLEAR:**DMA channel2 的源端传输完成状态清除.
* 0:不操作
* 1:清除**Bits 1 CHAN1\_CLEAR:**DMA channel1 的源端传输完成状态清除.
* 0:不操作
* 1:清除**Bits 0 CHAN0\_CLEAR:**DMA channel0 的源端传输完成状态清除.
* 0:不操作
* 1:清除

###### 21.8.19 DMA\_ClearDstTran

Offset:0x350 Reset Value:0x0000000000000000

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **63-4** | **3** | **2** | **1** | **0** |
| RESERVED | CHAN3\_CLEAR | CHAN2\_CLEAR | CHAN1\_CLEAR | CHAN0\_CLEAR |
| r-0h | w-0h | w-0h | w-0h | w-0h |

**Bits 63-4 RESERVED:**保留.

**Bits 3 CHAN3\_CLEAR:**DMA channel3 的目的端传输完成状态清除.

* 0:不操作
* 1:清除 **Bits 2 CHAN2\_CLEAR:**DMA channel2 的目的端传输完成状态清除.
* 0:不操作
* 1:清除**Bits 1 CHAN1\_CLEAR:**DMA channel1 的目的端传输完成状态清除.
* 0:不操作
* 1:清除**Bits 0 CHAN0\_CLEAR:**DMA channel0 的目的端传输完成状态清除.
* 0:不操作
* 1:清除

###### 21.8.20 DMA\_ClearErr

Offset:0x358 Reset Value:0x0000000000000000

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **63-4** | **3** | **2** | **1** | **0** |
| RESERVED | CHAN3\_CLEAR | CHAN2\_CLEAR | CHAN1\_CLEAR | CHAN0\_CLEAR |
| r-0h | w-0h | w-0h | w-0h | w-0h |

**Bits 63-4 RESERVED:**保留.

**Bits 3 CHAN3\_CLEAR:**DMA channel3 的传输出错状态清除.

* 0:不操作
* 1:清除 **Bits 2 CHAN2\_CLEAR:**DMA channel2 的传输出错状态清除.
* 0:不操作
* 1:清除**Bits 1 CHAN1\_CLEAR:**DMA channel1 的传输出错状态清除.
* 0:不操作
* 1:清除**Bits 0 CHAN0\_CLEAR:**DMA channel0 的传输出错状态清除.
* 0:不操作
* 1:清除

###### 21.8.21 DMA\_DmaCfgReg

Offset:0x398 Reset Value:0x0000000000000000

|  |  |
| --- | --- |
| **63-1** | **0** |
| RESERVED | DMA\_EN |
| r-0h | rw-0h |

**Bits 63-1 RESERVED:**保留.**Bits 0 DMA\_EN:**DMA Enable控制.

* 0:去Enable
* 1:Enable

###### 21.8.22 DMA\_ChEnReg

Offset:0x3a0 Reset Value:0x0000000000000000

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **63-12** | **11** | **10** | **9** | **8** |
| RESERVED | CH\_EN\_WE\_3 | CH\_EN\_WE\_2 | CH\_EN\_WE\_1 | CH\_EN\_WE\_0 |
| r-0h | w-0h | w-0h | w-0h | w-0h |
| **7-4** | **3** | **2** | **1** | **0** |
| RESERVED | CH\_EN\_3 | CH\_EN\_2 | CH\_EN\_1 | CH\_EN\_0 |
| r-0h | rw-0h | rw-0h | rw-0h | rw-0h |

**Bits 63-12 RESERVED:**保留.

**Bits 11 CH\_EN\_WE\_3:**DMA channel3 的Enable控制信息的写Enable.

* 0:去Enable
* 1:Enable **Bits 10 CH\_EN\_WE\_2:**DMA channel2 的Enable控制信息的写Enable.
* 0:去Enable
* 1:Enable**Bits 9 CH\_EN\_WE\_1:**DMA channel1 的Enable控制信息的写Enable.
* 0:去Enable
* 1:Enable**Bits 8 CH\_EN\_WE\_0:**DMA channel0 的Enable控制信息的写Enable.
* 0:去Enable
* 1:Enable**Bits 7-4 RESERVED:**保留.

**Bits3 CH\_EN\_3:**DMA channel3的Enable控制,当DMA传输完成后,硬件自动将此channel去Enable.

* 0:去Enable
* 1:Enable **Bits2 CH\_EN\_2:**DMA channel2的Enable控制,当DMA传输完成后,硬件自动将此channel去Enable.
* 0:去Enable
* 1:Enable**Bits1 CH\_EN\_1:**DMA channel1的Enable控制,当DMA传输完成后,硬件自动将此channel去Enable.
* 0:去Enable
* 1:Enable**Bits0 CH\_EN\_0:**DMA channel0的Enable控制,当DMA传输完成后,硬件自动将此channel去Enable.
* 0:去Enable
* 1:Enable

### 22. 通用定时器 (GPTIMER)

#### 22.1 Introduction

ASR6601 共有 4 个通用定时器（GPTIMER）,其中 GPTIMER0 和 GPTIMER1 有 4 路通道,

GPTIMER2 和 GPTIMER3 有 2 路通道,即 GPTIMER2 和 GPTIMER3 没有通道 2 和 3.

GPTIMER包含16-bitcounter,支持自动重装载功能,且支持最多16-bit可编程的分频counter, 4 路通道可独立配置为输入或输出,支持输入捕获、输出比较等功能,计数clock 和计数模式可软件配置,支持连接霍尔器件即支持编码模式（仅适用于 GPTIMER0 和 GPTIMER1）,支持 DMA 配置,有独立interrupt输出,支持编码功能等.基于丰富的通道配置和功能,该 GPTIMER 可用于定时计数、测量输入脉冲宽度（us-ms 级）、产生 PWM 波形等应用.

##### 22.2 Main features

* 16-bit counter,支持自动重装载,可配置边沿对齐（向上、向下）计数和中间对齐（向上/ 向下）计数
* 16-bit 可编程分频counter（分频系数 1-65535）,可在计数过程中配置
* 最多 4 路独立通道,可完成输入捕获、输出比较、PWM 波形输出、单脉冲波形输出
* 支持通道输出极性选择,和输入边沿配置
* 支持与外部输入或其他模块（GPTIMER、ADC、DAC）同步
* 独立的 DMA 通道,最多 6 组 DMA Request ,包括更新事件、触发事件以及 4 组通道事件（捕获、比较）
* 支持正交编码功能
* 支持外部触发通道输入clock 用于计数并且支持外部触发通道输入触发信号,支持通道输入clock 用于计数
* 支持通道重映射,即把其它模块的 GPIO 信号或内部信号映射到通道或外部通道

GPTIMER 的架构框Figure如下:



**Figure 22-1 GPTIMER 框Figure**

**Table 22-1 GPTIMER 模块介绍**

|  |  |
| --- | --- |
| Module name | Description |
| slave control | 从模式控制器 |
| master control | 主模式控制器 |
| u\_etr\_ctrl | ETR 通道控制,包括极性、分频、滤波等配置 |
| u\_channel\_in\_x | 输入通道 x 控制,包括极性、滤波及边沿配置 |
| u\_icx\_div | 输入通道 x 事件分频器 |
| u\_sampling\_clock | 产生滤波器的采样clock |
| u\_capture\_x | 输入通道 x 捕获功能 |
| u\_compare\_x | 输出通道 x 比较功能 |
| u\_psc\_counter | 16-bit 分频counter |
| u\_counter | 16-bit counter |
| u\_reg\_model | Register相关配置 |
| output stage | 输出控制 |
| interrupt control | interrupt控制 |
| dma control | DMA 功能控制 |
| itr\_input | 其它 GPTIMER 的内部输入 |
| etr\_input | 外部触发通道的输入 |
| channel\_input | 通道输入 |
| dma\_ack | DMA 回复的 ACK |
| dma\_req | 向 DMA 发送的Request |
| apb\_write | apb 总线写 |
| Module name | Description |
| apb\_read | apb 总线读 |
| trigger\_output | 主模式下的信号输出,为内部信号,不会输出到外部 |
| channel\_output | 通道输出 |

#### 22.3 counter

GPTIMER的counter共16-bit,支持向上、向下、中间对齐计数,计数clock 可选,可软件配置计数Enable与关闭,软件可随时读写（建议不要在计数过程中写入,以免发生未知错误）.

##### 22.3.1 计数clock 选择

GPTimer 共有四种计数clock 源,分别是内部clock 、外部clock 模式 1、外部clock 模式 2 以及内部触发信号控制计数.其中,内部clock 为默认方式（SMS==3’b000）,clock 来自 RCC,只要 CEN 置Bits,则分频counter和该counter便开始计数,其他三种情况,均使用相应信号作为计数Enable,并不是作为真正的clock .

外部clock 模式 1（SMS==3’b111,TS==3’b100/101/110）,该模式下,counter由所选择的通道输入的上升沿或下降沿或双沿作为counter的计数Enable控制计数,例如选择通道 0 的上升沿控制计数,则每个上升沿都会让counter加 1（向上计数、不分频）,波形如下Figure:

 **Figure 22-2 外部clock 模式 1 计数**

外部clock 模式2（ECE==1）,该模式下,counter由ETR 的上升沿或下降沿作为counter的计数Enable控制计数,例如配置 ETR 的上升沿有效,则波形如下Figure所示.

 **Figure 22-3 外部clock 模式 2 计数**

GPTIMER 还可以选择内部触发信号控制计数（SMS==3’b111,TS==3’b001/010/011）,即可以由上一级 GPTIMER 的触发输出信号作为该 GPTIMER 的计数clock ,从而实现 GPTIMER 的级联,该情况下,上一级 GPTIMER 相当于一个分频counter,波形如下Figure所示.



###### Figure 22-4 内部触发信号做clock 计数

当ETR做为计数clock 输入时可以有两种方式实现,一种是外部clock 模式1,配置SMS==3’b111,

TS==3’b111,另一种是外部clock 模式 2,配置 ECE==1

###### 22.3.2 自动重装载

GPTIMER 支持自动重装载功能,向上计数时,计数到重装载值（ARR）后,将会归零重新计数,向下计数时,会从 ARR 开始计数,计数到 0 后回到 ARR 重新计数,中间对齐计数时,counter从 0 开始计数到 ARR-1,接着从 ARR 计数到 0.

ARR 可软件配置（ARPE）是否使用启用影子Register,如果 ARPE=0,则禁用影子Register,软件写入的值同步更新到 ARR 供counter使用,如果 ARPE=1,则软件写入的值不会立即生效,直到更新事件到来,才会将该值更新到影子Register中供counter使用.

###### 22.3.3 向上计数

若配置为向上计数模式,则counterEnable且有计数clock 后,会从 0 开始递加到 ARR,产生向上溢出事件（overflow）,然后归零重新开始计数.计数过程中如果 UG 置Bits（软件或硬件）,则counter包括分频counter会被初始化（归零）.时序上,overflow标志将在最后一个计数值期间产生,如果启用影子Register,则 ARR、PSC、CCRx 等Register将会在下一轮计数开始时更新到相应的影子Register,波形如下Figure所示.



**Figure 22-5 向上计数**

###### 22.3.4 向下计数

若配置为向下计数模式,则counterEnable且有计数clock 后,会从 ARR 开始递减到 0,产生向下溢

出事件（underflow）,然后回到 ARR 重新开始计数.计数过程中如果 UG 置Bits（软件或硬件）,则counter包括分频counter会被初始化（counter回到 ARR,分频counter归零）.时序上,

underflow 标志将在最后一个计数值（CNT=0）期间产生,但是请注意,如果启用影子Register,则 ARR Register将会在下一轮计数开始之前（CNT=0）更新到相应的影子Register,以保证下一轮计数过程可以使用最新的装载值和分频值,PSC 和 CCRx 则与之前相同,将在 underflow 下一clock 更新到影子Register,波形如下Figure所示.

 **Figure 22-6 向下计数**

##### 22.3.5 中间对齐计数

若配置为中间对齐计数模式,则counterEnable且有计数clock 后,会从 0 开始递增到 ARR-1,产生 overflow 事件,然后从 ARR 递减到 1,产生 underflow 事件,再从 0 开始重新计数.计数过程中如果 UG 置Bits（软件或硬件）,则counter包括分频counter会被初始化（归零）.请注意,如果启用影子Register,则 ARR 和 PSC Register将会在向上计数到老的 ARR-1 时更新到相应的影子Register,以保证在向下计数时可以使用新的 ARR 和新的 PSC,CCRx 的更新与之前情况相同.向下计数时,将会在产生 underflow 后更新 ARR、PSC 和 CCRx 的影子Register.在该模式下,计数方向由硬件控制,软件配置无效.波形如Figure,



**Figure 22-7 中间对齐计数**

##### 22.4 分频counter

GPTIMER 支持 16-bit（1~65535）可编程分频,这一功能通过该分频counter实现.上一级电路产生的计数Enable信号将作为该分频counter的Enable控制计数,当分频counter计数到预先加载的分频值后,输出一个脉冲,作为下一级counter的计数Enable,然后分频counter归零重新计数,如此循环.

分频counter的分频值默认启用影子Register,即软件的写操作不会立即生效,而是直到更新事件

（UG 置Bits、计数溢出）到来,才会将新的分频值写入影子Register,此时该分频值才正式生效.

软件读操作读取的是写入的Register值,而不是影子Register,如果在更新事件到来前有多次写操作,则会覆盖之前写入的值.

举一个例子说明分频counter,如配置为 4 分频,则输入 4 个高电平,才会输出一个有效的脉冲,波形如下Figure（通道 0,无滤波,选择通道 0 上升沿作为有效脉冲,配置 ic0 为 4 分频）.



**Figure 22-8 分频counter**

##### 22.5 采样clock

各输入通道和外部触发通道均可以选择数字滤波功能,该数字滤波功能通过使用高频的采样clock （频率至少是输入信号的 4 倍）对输入信号采样.GPTIMER 内部所有 Flip-Flop 的clock 均由 pclk 提供.软件可以配置采样clock 的频率（CKD 分别为 pclk、pclk/2,pclk/4）,通过采用counter实现分频,如配置采样频率为 pclk 的 4 分频,则counter由 pclk 控制计数,每 4 个 pclk 周期产生一个脉冲（宽度为 pclk 的一个周期）,用于后级counter的Enable信号.在各通道内,用户还可以再次配置数字滤波器的采样clock 分频,即配置 ETF 的值,滤波原理上述相同.

#### 22.6 通道

每个 GPTIMER 的各个通道有多路来源,这些信号来源与 GPTIMER 均为异步关系,因此在模块内部需要做同步处理.同步后的通道输入信号,可以根据软件配置进行滤波处理,滤波的采样频率和窗口长度均可以软件配置（ICxF）,滤波后的信号由一个边沿检测器产生边沿信号,可以由软件配置有效电平（或有效边沿）.处理后的通道信号可以作为从模式控制器的控制信号,编码模式输入信号,也可以作为输入捕获Enable信号（可配置分频）.每个输入通道可映射到当前通道、相邻通道或内部触发信号 TRC（CCxS[1:0]配置）,具体方案见Table格（以通道 0 为例）,其中 ti0fp0 为映射到通道 0 的输入信号,ti1fp0 为映射到通道 1 的输入信号.

##### Table 22-2 输入通道有效极性配置

|  |  |  |  |
| --- | --- | --- | --- |
| {CC0NP, CC0P} | 有效脉冲（应用于输入捕获、复Bits模式、触发模式、外部clock 模式） | | 有效电平（应用于Gate 模式、编码模式） |
| ti0fp0 | ti1fp0 | ti0fp |
| 2’b00 | 通道 0 上升沿 | 通道 1 上升沿 | 通道 0 高电平 |
| 2’b01 | 通道 0 下降沿 | 通道 1 下降沿 | 通道 0 低电平 |
| 2’b10 | 保留 | 保留 | 保留 |
| 2’b11 | 通道 0 双沿 | 通道 1 双沿 | 通道 0 高电平 |

##### Table 22-3 输入通道映射

|  |  |
| --- | --- |
| CCxS | icx映射 |
| 2’b01 | tixfpx（x 代Table当前通道） |
| 2’b10 | tiyfpx（y 代Table相邻通道） |
| 2’b11 | trc（仅适用于 TS=3’b000、3’b001、3’b010、3’b100） |

此外,通道 0 与其他通道不同,可以软件配置（TI0S 置Bits）通道 0 连接到通道 0、通道 1 和通

道2的异或输出,此时该通道的其他功能依然有效,该功能仅适用于GPTIMER0和GPTIMER1.

##### 22.6.1 输入捕获

输入捕获仅在通道被配置为输入模式且 CCxE 置Bits时被激活,可以由软件（CCxG）或硬件

（当前通道、相邻通道或内部互联信号）触发捕获行为.当有效的捕获触发信号产生时, GPTIMER 会把当前 counter 的值锁存到相应的 CCRx Register中,并且置Bits CCxIF 标志Bits,如果Enable了相应interrupt或 DMA 屏蔽Bits,则会产生interrupt信号或 DMA Request .如果 CCxIF 置Bits时（未被软件清除）又发生了不止一次捕获行为,则 CCxOF 置Bits,指示发生了捕获溢出事件,读取

CCxR Register（或 SR Register相应Bits写 0）可以清除 CCxIF 和 CCxOF.波形如下Figure所示.

 **Figure 22-9 输入捕获**

##### 22.6.2 输出比较

输出比较功能仅在通道被配置为输出模式且 CCxE 置Bits时被激活,该功能通过比较 counter 值与 CCRx 的值,控制通道输出高低翻转,进而输出特定的波形.

###### 22.6.2.1 CCRx 预装载功能

CCRx Register的写入有两种方式,若 CCxPE 置Bits,则软件写入的 CCRx 值不会直接被使用,真正起作用的是影子Register,作为缓冲,直到更新事件发生后,才会将 CCRx 的值更新到影子Register中；若 OCxPE 复Bits,则软件写入的 CCRx 值会直接被使用,影子Register禁用.

###### 22.6.2.2 输出比较模式

当匹配（CNT==CCR）发生时,通道输出会根据配置的模式进行翻转,且 CCxIF 标志Bits会置Bits,若Enable了相应的interrupt或 DMA 屏蔽Bits,则会产生interrupt或 DMA Request ,具体的模式控制如下Table格所示.

Table 22-4 输出比较各种模式下的输出波形Description

|  |  |  |
| --- | --- | --- |
| 比较模式 | 计数模式 | 输出波形 |
| 冻结模式 | Any | 无论 CNT 如何变化,输出维持不变 |
| SET 模式 | Any | 在 CNT==CCR 后,输出高电平 |
| RESET 模式 | Any | 在 CNT==CCR 后,输出低电平 |
| TOGGLE 模式 | Any | 在 CNT==CCR 时,翻转当前电平 |
| 强制 RESET 模式 | Any | 选择该模式后,直接输出低电平,忽略比较结果 |
| 强制 SET 模式 | Any | 选择该模式后,直接输出高电平,忽略比较结果 |
| PWM1 模式 | 向上计数（边  沿对齐 pwm） | CNT<CCR 时,输出高电平,CNT>=CCR 时,输出低电平.如果 CCR>ARR,则输出一直为高电平（100%PWM）,如果  CCR==0,则输出一直为低电平（0%PWM）. |
| 向下计数（边  沿对齐 pwm） | CNT<=CCR 时,输出高电平,CNT>CCR 时,输出低电平.  如果 CCR>ARR,则输出一直为高电平（100%PWM）.Notice: 0%PWM 模式在该情况下不支持. |
| 中间计数（中  间对齐 pwm） | 相当于向上计数与向下计数相结合.如果 CCR>=ARR,则输出一直为高电平（100%PWM）,如果 CCR==0,则输出一直为低电平（0%PWM）. |
| PWM2 模式 | 向上计数（边  沿对齐 pwm） | CNT<CCR 时,输出低电平,CNT>=CCR 时,输出高电平.  如果 CCR>ARR,则输出一直为低电平（0%PWM）,如果  CCR==0,则输出一直为高电平（100%PWM）. |
| 向下计数（边  沿对齐 pwm） | CNT<=CCR 时,输出低电平,CNT>CCR 时,输出高电平.如果 CCR>ARR,则输出一直为低电平（0%PWM）.Notice:  100%PWM 模式在该情况下不支持. |
| 中间计数（中  间对齐 pwm） | 相当于向上计数与向下计数相结合.如果 CCR>=ARR,则输出一直为低电平（0%PWM）,如果 CCR==0,则输出一直为高电平（100%PWM）. |

**各模式下输出波形**如下所示（以向上计数为例）:



Figure 22-10 各种输出比较模式下的波形

其中,PWM 模式下还支持通过配置 ARR 和 CCR 控制输出 0%和 100%波形,边沿对齐计数的

PWM2 波形如下Figure所示.



**Figure 22-11 边沿对齐计数 PWM2** 中间对齐计数的 PWM2 波形如下Figure所示.

 **Figure 22-12 中间对齐计数 PWM2**

22.6.2.3 单脉冲快速输出功能

单脉冲模式下（OPM 置Bits）,两个 PWM 模式可以配置为快速输出模式（置Bits OCxFE）,Enable快速模式后,输出波形将忽略 CNT 和 CCR 的比较结果,改为由触发信号（根据 TS 选择）上升沿控制电平翻转,输出信号电平等同于匹配事件发生后的电平,例如,配置 GPTIMER 通道 0 为输出模式,选择 PWM1 模式,触发信号选择 ETR 输入,则当 ETR 输入高电平后,通道 0 立刻输出高电平（OCxP=0 情况下）,该功能可以有效减少从触发信号边沿到波形输出之间的延迟.Enable快速模式时的单脉冲输出波形如下Figure:

 **Figure 22-13 Enable快速模式时的单脉冲输出波形**

22.6.2.4 外部触发信号清除通道输出功能

输出波形除了受计数值的影响,还可以通过外部触发信号（ETR）硬件清零,若要使用该功能,需提前Enable OCxCE Bits,同时保证 ETR 禁用分频（ETP=2’b00）,且 ETR 不得作为计数clock .Enable该功能后（OCxCE=1）,ETR 的电平有效（默认高电平）时,通道输出将被清除,更改 ETR的有效电平时通过配置ETP实现.关闭该功能后（OCxCE=0）,通道输出不会立刻恢复,而是等到下一次计数周期开始才会恢复正常输出.开启和关闭外部触发信号清除通道输出功能的对比波形如下Figure:

 **Figure 22-14 外部触发信号清除通道输出**

##### 22.7 触发输入通道

每个 GPTIMER 的 ETR 有多路来源,通过 MUX 选择一路输入到模块内,这些信号来源与 GPTIMER 均为异步关系,因此在模块内部需要做同步处理.同步后的 ETR 信号,可以根据软件配置选择有效电平（或有效边沿）、配置分频（1、2、4、8）以及滤波处理,滤波的采样频率和窗长度均可以软件配置（ETF）.

##### 22.8 更新事件管理

更新事件主要有以下事件源:

1. counter的溢出事件（overflow 和 underflow）
2. UG 置Bits

与更新事件管理相关的控制信号主要是 URS 和 UDIS,具体控制如下:

* 若 UDIS=0,URS=0,则 underflow、overflow、UG 置Bits会初始化 counter 和 pre-scale counter（center-aligned 模式下 counter 不会被 overflow 清零,也不会被 underflow 加载 ARR）,如果启用影子Register,更新事件将会把写入的值更新到影子Register中（ARR 取决于 ARPE,CCRx 取决于 OCxPE）,UIF 会置Bits,如果Enable了interrupt或 DMA 屏蔽Bits,则会产生interrupt或 DMA Request .
* 若 UDIS=0,URS=1,则 underflow、overflow、UG 置Bits会初始化 counter 和 pre-scale counter（center-aligned 模式下 counter 不会被 overflow 清零,也不会被 underflow 加载 ARR）,如果启用影子Register,更新事件将会把写入的值更新到影子Register中（ARR 取决于ARPE,CCRx取决于 OCxPE）,UIF 只会在 overflow或underflow情况下置Bits,如果Enable了interrupt或 DMA 屏蔽Bits,则会产生interrupt或 DMA Request ,该配置可以有效避免输入捕获模式下 UG 置Bits初始化counter时,同时产生捕获interrupt和更新interrupt的情况.
* 若 UDIS=1（忽略 URS）,则 underflow、overflow、UG 置Bits会初始化 counter 和 prescale counter（center-aligned模式下counter不会被overflow清零,也不会被underflow 加载 ARR）,但是影子Register不会被更新,且 UIF 不会置Bits,因此不会产生相应interrupt或

DMA Request .

##### 22.9 编码模式控制

该 GPTIMER 支持正交编码计数功能,可以通过通道 0 和通道 1 输入正交信号,进行计数和方向检测.编码模式共有三种,仅在通道 0 边沿计数、仅在通道 1 边沿计数以及在通道 1 和通道 2 边沿计数.在此功能下,两个通道输入可以配置数字滤波功能,极性配置和分频配置无效.通过两个通道信号的组合,可以产生计数Enable和方向控制信号,控制counter加减（如果CENEnable）,因此在该模式下,软件配置计数方向无效.具体的组合方式见下Table,

###### Table 22-5 编码模式

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| 编码模式 | 通道0/1电平 | 通道0边沿 | | 通道1电平 | |
| 上升沿 | 下降沿 | 上升沿 | 下降沿 |
| 编码模式 1  （在通道 1 边沿计数） | 高电平 | - | - | 向上计数 | 向下计数 |
| 低电平 | - | - | 向下计数 | 向上计数 |
| 编码模式 2  （在通道 0 边沿计数） | 高电平 | 向下计数 | 向上计数 | - | - |
| 低电平 | 向上计数 | 向下计数 | - | - |
| 编码模式 3  （在所有通道边沿计数） | 高电平 | 向下计数 | 向上计数 | 向上计数 | 向下计数 |
| 低电平 | 向上计数 | 向下计数 | 向下计数 | 向上计数 |

在编码模式下,counter同样是在 0-ARR 之间计数,向上计数到 ARR 时会产生 overflow,然后回到 0 重新计数,向下计数到 0 时会产生 underflow,然后回到 ARR 重新计数.此外,在该模式下,输入捕获（通道 2 和通道 3）、输出比较、分频、触发输出功能依然适用.编码模式 1 的计数波形Figure如下:



**Figure 22-15 编码模式 1 的计数波形**

##### 22.10 从模式控制

GPTIMER 支持级联操作,作为外部或内部模块的从机.从模式的触发输入信号 TRGI 有多路来源,通过 TS[2:0] 进行选择,结构如上Figure,其中 ITRx 来自于内部其他 GPTIMER 的触发输出信号（TRGO）,具体映射关系见下Table.

###### Table 22-6 各 GPTIMER 的内部触发输入映射

|  |  |  |  |
| --- | --- | --- | --- |
| 从机GPTIMER | ITR0 | ITR1 | ITR2 |
| GPTIMER0 | GPTIMER2 | GPTIMER3 | GPTIMER1 |
| GPTIMER1 | GPTIMER0 | GPTIMER3 | GPTIMER2 |
| GPTIMER2 | GPTIMER3 | GPTIMER0 | - |
| GPTIMER3 | GPTIMER1 | GPTIMER2 | - |

从模式控制主要有以下四种方式:

1. **复Bits模式**:TRGI 的上升沿将会初始化counter和分频counter,并且可以更新影子Register

（UDIS=0 时）,从模式下复Bits模式波形Figure如下:

 **Figure 22-16 从模式下的复Bits模式波形**

1. **门控模式**:TRGI 电平可以控制counter的运行和停止,默认有效电平下,高电平时counter计数,低电平时counter停止计数（不是复Bits）,在该模式下,CEN 需要软件置Bits.从模式下门控模式波形Figure如下:

 **Figure 22-17 从模式下的门控模式波形**

1. **触发模式**:TRGI 的上升沿可以控制counter开始计数,但是无法控制counter是否停止,在该模式下,CEN 不需要软件置Bits.从模式下触发模式波形Figure如下:

 **Figure 22-18 从模式下的触发模式波形**

1. **clock 模式（即外部clock 模式 1）**:TRGI 的上升沿作为counter的计数Enable控制计数,此时分频电路依然有效.

在从模式下,TRGI 的上升沿会置Bits TIF 标志Bits,如果Enable了相应interrupt或 DMA 屏蔽Bits,则会产生interrupt或 DMA Request .但是门控模式有一些特殊,在该模式下,除上升沿外,下降沿也可以置Bits TIF.

另外使用 GPTIMER 级联时,需要保证主从的clock 同频同相,否则会发生未知错误.

##### 22.11 主模式控制

GPTIMER也可以作为主模式使用,通过产生触发输出信号（TRGO）来控制其他GPTIMER或

ADC 和 DAC.TRGO 信号的来源可以由软件配置,具体如下:

* MMS=3’b000:复Bits模式,此时 UG 标志Bits将作为 TRGO 信号输出给外部从机.
* MMS=3’b001:Enable模式,此时counter的计数Enable将作为 TRGO 信号输出给外部从机.如果当前GPTIMER同时处于从机门控模式,则该信号为门控信号,否则直接将CEN作为 TRGO 信号输出.
* MMS=3’b010:更新模式,此时将更新事件作为 TRGO 信号输出.
* MMS=3’b011:通道 0 比较脉冲模式,此时如果 CC0IF 将置Bits,则输出一个脉冲作为

TRGO 信号,无论此时 CC0IF 是否已经置Bits.

* MMS=3’b100:比较模式 1,此时将 OC0REF 作为 TRGO 信号输出.  MMS=3’b101:比较模式 2,此时将 OC1REF 作为 TRGO 信号输出.
* MMS=3’b110:比较模式 3,此时将 OC2REF 作为 TRGO 信号输出.
* MMS=3’b111:比较模式 4,此时将 OC3REF 作为 TRGO 信号输出.

**Notice:** 后*4*种模式输出的信号*OCxREF*,并不是最终的通道输出,而是内部信号.

GPTIMER 配置为主机Enable模式时,有一种特殊应用,即同步启动主机和从机的counter.但是因为主机的 CEN 作为 TRGO 输出到从机并Enable从机counter需要两个clock 的延迟（假定主从clock 同频同相）,因此在使用这一功能时,内部会把主机的 CEN 信号用两级Register延迟 2 个clock 周期,以保证同步,该功能可以软件配置是否Enable（MSM）.

#### 22.12 输出控制

GPTIMER0 和 GPTIMER1 共 4 路通道输出,GPTIMER2 和 GPTIMER3 共 2 路通道输出,同时有相应的输出Enable信号,通道输出仅在 CCxE 置Bits时有效,此时可以通过 CCxP 控制输出极性,输出极性指输出有效电平为高电平还是低电平.输出Enable信号为高有效,即在 CCxE 置Bits时有效,同时需保证通道被正确配置为输出模式,输出模式通过 CCxS 配置.

##### 22.13 通道重映射

通道重映射就是把 GPTIMER 的通道或外部触发通道 ETR 的输入信号从其他外部或内部信号映射过来.GPTIMER0 的 ETR 通道、通道 0 和通道 3 支持重映射,GPTIMER0 的通道 2 支持重映射,GPTIMER2 的 ETR 通道、通道 0 和通道 1 支持重映射,GPTIMER3 的 ETR 通道、通道 0 支持重映射.

#### 22.14 Debug 模式控制

GPTIMER 可由软件配置 debug 下是否停止计数,如果Enable该功能,则进入系统 debug 模式时,

GPTIMER 停止计数（counter不会初始化）.

#### 22.15 DMA 控制

GPTIMER 共有 6 个 DMA Request 源,分别是 update 事件（UIF）、4 路通道事件（捕获事件、比较匹配）（CCxIF）以及触发事件（TIF）,可以由独立的屏蔽Bits配置是否Enable相应的 DMA Request .

对于通道事件 DMA,可以软件（CCDS Bits）配置通道的 DMA Request 源,若 CCDS=0,各通道 DMA Request 来自于各通道的事件,如捕获、比较匹配事件；若 CCDS=1,则各通道的 DMA Request 均来自于更新事件,通道事件将被屏蔽.

各 DMA Request 仅在无相应应答信号、DMA Enable开启且 DMA 事件发生时置Bits,在 DMA Request 置Bits时,应答信号可以清除 DMA Request ,否则 DMA Request 将一直保持置Bits状态.

除常规的 DMA 操作外,GPTIMER 还支持 burst 功能,即一个 DMA Request 可以连续读写多个内部Register.DBL Bits可以选择 burst 长度,最多 18 个,DBA 可以选择 burst 的起始地址,DMAR Register的地址可以作为 DMA 的目标地址或源地址（DMA 内部不需要设置每次递增）.当某一个 DMA Request 置Bits时,GPTIMER 根据 DBL 和 DBA 的值,计算出每一次读写操作的实际地址,实际地址计算方法为:*CR1 + (DBA + index) x 4*,其中 index 的值为 0 至 DBL.

**Notice:** Register组中间有保留Register地址,该地址也将包含在*DMA*的*burst*操作中,实际使用时需注意配置的长度.例如,起始地址选择*ARR* Register（*0x2C*）,*DBL* 配置为*5’b00010*（*3* 个

*burst*）,则*DMA* 实际操作的三个Register分别是*0x2C*、 *0x30*、 *0x34*,其中*0x30* 为保留Register,因此无法写入且读出永远为*0*.

#### 22.16 interrupt

GPTIMER 共有 6 个interrupt源,分别是 update 事件（UIF）、4 路通道事件（捕获事件、比较匹配）（CCxIF）以及触发事件（TIF）,各interrupt可以由独立的interrupt屏蔽Bits选择是否Enable,interrupt标志Bits与相应屏蔽Bits是 AND 的关系,interrupt之间是 OR 的关系.GPTIMER 的interrupt信号如下Table:

**Table 22-7 GPTIMER interrupt信号**

|  |  |
| --- | --- |
| interrupt名称 | Description |
| 触发事件interrupt | 触发源产生事件时的interrupt |
| 通道 3 事件interrupt | 通道 3 产生捕获或比较事件时的interrupt |
| 通道 2 事件interrupt | 通道 2 产生捕获或比较事件时的interrupt |
| 通道 1 事件interrupt | 通道 1 产生捕获或比较事件时的interrupt |
| 通道 0 事件interrupt | 通道 0 产生捕获或比较事件时的interrupt |
| 更新事件interrupt | 产生更新事件时的interrupt |

上述interrupt的Enable分别通过配置Register DIER 的 TIE、CC3IE、CC2IE、CC1IE、CC0IE、UIE Bits实现.

#### 22.17 GPTIMER 相关RegisterDescription

GPTIMER0 基地址:0x4000A000

GPTIMER1 基地址:0x4001A000

GPTIMER2 基地址:0x4000B000

GPTIMER3 基地址:0x4001B000

**Table 22-8 GPTIMER Register Summary**

|  |  |  |
| --- | --- | --- |
| Register | Offset | Description |
| GPTIM\_CR1 | 0x00 | Control Register 1 |
| GPTIM\_CR2 | 0x04 | Control Register 2 |
| GPTIM\_SMCR | 0x08 | 从模式Control Register |
| GPTIM\_DIER | 0x0C | DMA/interruptEnableRegister |
| GPTIM\_SR | 0x10 | 状态Register |
| GPTIM\_EGR | 0x14 | 事件Register |
| GPTIM\_CCMR1 | 0x18 | 捕获比较模式Register 1 |
| GPTIM\_CCMR2 | 0x1C | 捕获比较模式Register 2 |
| GPTIM\_CCER | 0x20 | 捕获比较EnableRegister |
| GPTIM\_CNT | 0x24 | 计数Register |
| GPTIM\_PSC | 0x28 | counter分频值Register |
| GPTIM\_ARR | 0x2C | counter重装载值Register |
| GPTIM\_CCR0 | 0x34 | 通道 0 捕获比较Register |
| GPTIM\_CCR1 | 0x38 | 通道 1 捕获比较Register |
| GPTIM\_CCR2 | 0x3C | 通道 2 捕获比较Register |
| GPTIM\_CCR3 | 0x40 | 通道 3 捕获比较Register |
| GPTIM\_DCR | 0x48 | DMA Control Register |
| GPTIM\_DMAR | 0x4C | DMA 地址Register |
| GPTIM\_OR | 0x50 | 通道重映射Register |

##### 22.17.1 GPTIM\_CR1

Offset:0x00

Reset Value:0x0000

|  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- |
| **15-10** | **9-8** | **7** | **6-5** | **4** | **3** | **2** | **1** | **0** |
| RESERVED | CKD | ARPE | CMS | DIR | OPM | URS | UDIS | CEN |
| r-0h | rw-0h | rw-0h | rw-0h | rw-0h | rw-0h | rw-0h | rw-0h | rw-0h |

**Bits 15-10 RESERVED:**保留.**Bits 9-8 CKD:**采样clock 分频.

* 00:fDTS = fpclk
* 01:fDTS = fpclk
* 10:fDTS = fpclk
* 11:fDTS = reserved

**Bits 7 ARPE:**重装载影子RegisterEnable.

* 0:ARR 影子Register除能
* 1:ARR 影子RegisterEnable

**Bits 6-5 CMS:**中间计数模式选择.

* 00:边沿对齐计数模式,DIR 控制向上或向下计数
* 01:中间对齐模式 1.输出比较interrupt标志Bits仅在向下计数过程中置Bits
* 10:中间对齐模式 2.输出比较interrupt标志Bits仅在向上计数过程中置Bits
* 11:中间对齐模式 3.输出比较interrupt标志Bits在向上和向下计数过程中均置Bits**Bits 4 DIR:**计数方向选择.中间对齐模式和编码模式,该Bits由硬件控制.
* 0:向上计数
* 1:向下计数

**Bits 3 OPM:**单脉冲模式Enable.

* 0:单脉冲模式除能
* 1:单脉冲模式Enable,counter在下一次更新事件停止计数

**Bits 2 URS:**更新事件源选择,该Bits仅影响interrupt和 DMA 标志Bits（UIF）,不影响内部逻辑.

* 0:counter溢出、UG Bits置Bits、从模式 reset 模式下的触发,均可以置Bits UIF
* 1:只有counter溢出事件可以置Bits UIF **Bits 1 UDIS:**更新事件除能.
* 0:更新事件Enable,中间对齐模式 1.输出比较interrupt标志Bits仅在向下计数过程中置Bits均可以产生更新事件
* 1:更新事件除能,影子Register和 UIF 均不会被更新,但是此时counter和分频counter仍可以被 UG 置Bits事件初始化**Bits 0 CEN:**counterEnable,触发模式下 CEN 由硬件置Bits,单脉冲模式下 CEN 由硬件清零.
* 0:counter除能
* 1:counterEnable

##### 22.17.2 GPTIM\_CR2

Offset:0x04

Reset Value:0x0000

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **15-8** | **7** | **6-4** | **3** | **2-0** |
| RESERVED | TI0S | MMS | CCDS | RESERVED |
| r-0h | rw-0h | rw-0h | rw-0h | r-0h |

**Bits 15-8 RESERVED:**保留.

**Bits 7 TI0S:**通道 1 源异或选择（该功能仅 timer0 和 timer1 支持）.

* 0:通道 0 映射到通道 0 输入
* 1:通道 0 为通道 0、1、2 的异或输出

**Bits 6-4 MMS:**主模式选择,可以配置 TRGO 输出.

* 000:复Bits模式,UG 将作为 TRGO 信号输出
* 001:Enable模式,CNT\_EN（不是 CEN）将作为 TRGO 信号输出
* 010:更新模式,更新事件（内部信号）将作为 TRGO 信号输出
* 011:比较脉冲模式,每次 CC0IF 将要置Bits时 TRGO 会输出一个脉冲,即使 CC0IF 已经置Bits
* 100:比较模式,OC0REF（内部信号）作为 TRGO 信号输出
* 101:比较模式,OC1REF（内部信号）作为 TRGO 信号输出
* 110:比较模式,OC2REF（内部信号）作为 TRGO 信号输出
* 111:比较模式,OC3REF（内部信号）作为 TRGO 信号输出

**Bits 3 CCDS:**通道 DMA Request 源选择（该功能仅 gptimer0 和 gptimer1 支持）.

* 0:各通道的 DMA Request （不包含更新事件Request 和触发事件Request ）由通道事件（捕获、比较）产生
* 1:各通道的 DMA Request （不包含更新事件Request 和触发事件Request ）由更新事件产生

**Bits 2-0 RESERVED:**保留.

##### 22.17.3 GPTIM\_SMCR

Offset:0x08 Reset Value:0x0000

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| **15** | **14** | **13-12** | **11-8** | **7** | **6-4** | **3** | **2-0** |
| ETP | ECE | ETPS | ETF | MSM | TS | RESERVED | SMS |
| rw-0h | rw-0h | rw-0h | rw-0h | rw-0h | rw-0h | r-0h | rw-0h |

**Bits 15 ETP:**外部触发极性选择（配置极性时最好先不要选择模式（SMS）,以防内部信号翻转触发未知错误）.

* 0:外部触发输入不反相
* 1:外部触发输入反相**Bits 14 ECE:**外部clock 模式 2 Enable.
* 0:禁用外部clock 模式 2
* 1:Enable外部clock 模式 2 **Bits 13-12 ETPS:**外部触发输入分频（该分频主要用于 50%占空比降频,如 24M 信号 2 分频为 12M,电平延展一倍）择.
* 00:不分频
* 01:2 分频
* 10:4 分频
* 11:8 分频**Bits 11-8 ETF:**外部触发输入滤波器配置.
* 0000:禁用滤波器
* 0001:滤波器采样频率 fsampling=fpclk, 滤波器长度 N=2
* 0010:滤波器采样频率 fsampling=fpclk, 滤波器长度 N=4
* 0011:滤波器采样频率 fsampling=fpclk, 滤波器长度 N=8
* 0100:滤波器采样频率 fsampling=fDTS/2, 滤波器长度 N=6
* 0101:滤波器采样频率 fsampling=fDTS/2, 滤波器长度 N=8
* 0110:滤波器采样频率 fsampling=fDTS/4, 滤波器长度 N=6
* 0111:滤波器采样频率 fsampling=fDTS/4, 滤波器长度 N=8
* 1000:滤波器采样频率 fsampling=fDTS/8, 滤波器长度 N=6
* 1001:滤波器采样频率 fsampling=fDTS/8, 滤波器长度 N=8
* 1010:滤波器采样频率 fsampling=fDTS/16, 滤波器长度 N=5
* 1011:滤波器采样频率 fsampling=fDTS/16, 滤波器长度 N=6
* 1100:滤波器采样频率 fsampling=fDTS/16, 滤波器长度 N=8
* 1101:滤波器采样频率 fsampling=fDTS/32, 滤波器长度 N=5
* 1110:滤波器采样频率 fsampling=fDTS/32, 滤波器长度 N=6
* 1111:滤波器采样频率 fsampling=fDTS/32, 滤波器长度 N=8

**Bits 7 MSM:**主从模式同步（使用该功能时,需保证两个 timer 的clock 同频同相）.

* 0:无动作
* 1:TRGI 触发输入将延迟产生作用,以便与从counter同时开始计数**Bits 6-4 TS:**触发源选择,选择 TRGI 的来源（配置该Bits时 SMS 必须处于清零状态）.
* 000:ITR0
* 001:ITR1
* 010:ITR2（timer2 和 timer3 无此通道）
* 011:保留
* 100:通道 0 边沿检测输出
* 101:通道 0 滤波器输出
* 110:通道 1 滤波器输出
* 111:外部触发输入**Bits 3 RESERVED:**保留.

**Bits 2-0 SMS:**从模式选择（选择模式前最好先配置好通道参数,以防内部信号翻转触发未知错误）.

* 000:禁用从模式
* 001:编码模式 1,counter仅在通道 1 边沿计数
* 010:编码模式 2,counter仅在通道 0 边沿计数
* 011:编码模式 3,counter在通道 0 和 1 的边沿计数
* 100:复Bits模式,TRGI 的上升沿将复Bitscounter
* 101:门控模式,counter仅在 TRGI 高电平期间计数
* 110:触发模式,counter在 TRGI 上升沿将开始计数,该模式仅控制计数的开始
* 111:外部clock 模式 1,TRGI 的上升沿作为counter计数clock

##### 22.17.4 GPTIM\_DIER

Offset:0x0C

Reset Value:0x0000

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| **15** | **14** | **13** | **12** | **11** | **10** | **9** | **8** |
| RESERVED | TDE | RESERVED | CC3DE | CC2DE | CC1DE | CC0DE | UDE |
| r-0h | rw-0h | r-0h | rw-0h | rw-0h | rw-0h | rw-0h | rw-0h |
| **7** | **6** | **5** | **4** | **3** | **2** | **1** | **0** |
| RESERVED | TIE | RESERVED | CC3IE | CC2IE | CC1IE | CC0IE | UIE |
| r-0h | rw-0h | r-0h | rw-0h | rw-0h | rw-0h | rw-0h | rw-0h |

**Bits 15 RESERVED:**保留.

**Bits 14 TDE:**触发事件 DMA Request Enable.

* 0:禁用触发事件 DMA Request
* 1:Enable触发事件 DMA Request **Bits 13 RESERVED:**保留.

**Bits 12 CC3DE:**通道 3 事件 DMA Request Enable.

* 0:禁用通道 3 事件 DMA Request
* 1:Enable通道 3 事件 DMA Request **Bits 11 CC3DE:**通道 2 事件 DMA Request Enable.
* 0:禁用通道 2 事件 DMA Request
* 1:Enable通道 2 事件 DMA Request

**Bits 10 CC3DE:**通道 1 事件 DMA Request Enable.

* 0:禁用通道 1 事件 DMA Request
* 1:Enable通道 1 事件 DMA Request **Bits 9 CC3DE:**通道 0 事件 DMA Request Enable.
* 0:禁用通道 0 事件 DMA Request
* 1:Enable通道 0 事件 DMA Request

**Bits 8 UDE:**更新事件 DMA Request Enable.

* 0:禁用更新事件 DMA Request
* 1:Enable更新事件 DMA Request **Bits 7 RESERVED:**保留.

**Bits 6 TIE:**触发事件interruptRequest Enable.

* 0:禁用触发事件interruptRequest
* 1:Enable触发事件interruptRequest **Bits 5 RESERVED:**保留.

**Bits 4 CC3IE:**通道 3 事件interruptRequest Enable.

* 0:禁用通道 3 事件interruptRequest
* 1:Enable通道 3 事件interruptRequest **Bits 3 CC2IE:**通道 2 事件interruptRequest Enable.
* 0:禁用通道 2 事件interruptRequest
* 1:Enable通道 2 事件interruptRequest **Bits 2 CC1IE:**通道 1 事件interruptRequest Enable.
* 0:禁用通道 1 事件interruptRequest
* 1:Enable通道 1 事件interruptRequest **Bits 1 CC0IE:**通道 0 事件interruptRequest Enable.
* 0:禁用通道 0 事件interruptRequest
* 1:Enable通道 0 事件interruptRequest **Bits 0 UIE:**通道 0 事件interruptRequest Enable.
* 0:禁用通道 0 事件interruptRequest
* 1:Enable通道 0 事件interruptRequest

##### 22.17.5 GPTIM\_SR

Offset:0x10 Reset Value:0x0000

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| **15-13** | | **12** | **11** | **10** | **9** | **8-7** |
| RESERVED | | CC3OF | CC2OF | CC1OF | CC0OF | RESERVED |
| r-0h | | rw-0h | rw-0h | rw-0h | rw-0h | r-0h |
| **6** | **5** | **4** | **3** | **2** | **1** | **0** |
| TIF | RESERVED | CC3IF | CC2IF | CC1IF | CC0IF | UIF |
| rw-0h | r-0h | rw-0h | rw-0h | rw-0h | rw-0h | rw-0h |

**Bits 15-13 RESERVED:**保留.

**Bits 12 CC3OF:**通道 3 overcapture 标志（写 0 清零）.

* 0:无 overcapture
* 1:发生了至少 1 次 overcapture **Bits 11 CC2OF:**通道 2 overcapture 标志（写 0 清零）.
* 0:无 overcapture
* 1:发生了至少 1 次 overcapture **Bits 10 CC1OF:**通道 1 overcapture 标志（写 0 清零）.
* 0:无 overcapture
* 1:发生了至少 1 次 overcapture **Bits 9 CC0OF:**通道 0 overcapture 标志（写 0 清零）.
* 0:无 overcapture
* 1:发生了至少 1 次 overcapture **Bits 8-7 RESERVED:**保留.

**Bits 6 TIF:**触发事件interrupt标志（写 0 清零）.

* 0:无触发事件
* 1:触发事件发生**Bits 5 RESERVED:**保留.

**Bits 4 CC3IF:**通道 3 捕获/比较事件标志（比较模式:写 0 清零；捕获模式:读 ccrx Register或写 0 均可清零）.

* 0:无事件
* 1:捕获或比较事件发生**Bits 3 CC3IF:**通道 2 捕获/比较事件标志（比较模式:写 0 清零；捕获模式:读 ccrx Register或写 0 均可清零）.
* 0:无事件
* 1:捕获或比较事件发生**Bits 2 CC3IF:**通道 1 捕获/比较事件标志（比较模式:写 0 清零；捕获模式:读 ccrx Register或写 0 均可清零）.
* 0:无事件
* 1:捕获或比较事件发生

**Bits 1 CC3IF:**通道 0 捕获/比较事件标志（比较模式:写 0 清零；捕获模式:读 ccrx Register或写 0 均可清零）.

* 0:无事件
* 1:捕获或比较事件发生

**Bits 0 UIF:**更新事件标志（读 SR 或写 0 可清零该Bits）.

* 0:无事件
* 1:更新事件发生

##### 22.17.6 GPTIM\_EGR

Offset:0x14

Reset Value:0x0000

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| **15-7** | **6** | **5** | **4** | **3** | **2** | **1** | **0** |
| RESERVED | TG | RESERVED | CC3G | CC2G | CC1G | CC0G | UG |
| r-0h | w-0h | r-0h | w-0h | w-0h | w-0h | w-0h | w-0h |

**Bits 15-7 RESERVED:**保留.

**Bits 6 TG:**触发产生.

* 0:无动作
* 1:产生一次触发事件,TIF 置Bits**Bits 5 RESERVED:**保留.

**Bits 4 CC3G:**通道 3 事件产生.

* 0:无动作
* 1:输入模式时产生捕获动作,输出模式时产生比较动作,两种模式下 CC3IF 置Bits**Bits 3 CC2G:**通道 2 事件产生.
* 0:无动作
* 1:输入模式时产生捕获动作,输出模式时产生比较动作,两种模式下 CC2IF 置Bits**Bits 2 CC1G:**通道 1 事件产生.
* 0:无动作
* 1:输入模式时产生捕获动作,输出模式时产生比较动作,两种模式下 CC1IF 置Bits**Bits 1 CC0G:**通道 0 事件产生.
* 0:无动作
* 1:输入模式时产生捕获动作,输出模式时产生比较动作,两种模式下 CC0IF 置Bits**Bits 0 UG:**更新事件产生.
* 0:无动作
* 1:产生一次更新事件

##### 22.17.7 GPTIM\_CCMR1

Offset:0x18

Reset Value:0x0000

**输出模式时结构如下:**

|  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| **15** | **14-12** | **11** | **10** | **9-8** | **7** | **6-4** | **3** | **2** | **1-0** |
| OC1CE | OC1M | OC1PE | OC1FE | CC1S | OC0CE | OC0M | OC0PE | OC0FE | CC0S |
| rw-0h | rw-0h | rw-0h | rw-0h | rw-0h | rw-0h | rw-0h | rw-0h | rw-0h | rw-0h |

**Bits 15 OC1CE:**通道 1 输出比较清除Enable.

* 0:禁用清除功能
* 1:Enable清除功能,ETRF 高电平可以清除通道输出**Bits 14-12 OC1M:**通道 1 输出比较模式选择.
* 000:冻结模式,通道输出不随比较结果变化
* 001:有效模式,匹配后通道输出有效电平
* 010:失效模式,匹配后通道输出失效电平
* 011:翻转模式,匹配后将翻转通道输出
* 100:强制有效模式,选择该模式后,直接输出有效电平
* 101:强制失效模式,选择该模式后,直接输出失效电平
* 110:PWM1 模式,该模式下,向上计数时,CNT<CCR 时通道输出有效电平,否则输出失效电平；向下计数时,CNT>CCR 时通道输出失效电平,否则输出有效电平（向上计数时,若 CCRx>ARR,

OCxREF 一直输出高电平,若 CCRx==0,OCxREF 一直输出低电平；向下计数时,若 CCRx>ARR,

OCxREF 一直输出高电平,此时 0% PWM 不支持）

* 111:PWM2 模式,该模式下,向上计数时,CNT<CCR 时通道输出失效电平,否则输出有效电平；

向下计数时,CNT>CCR 时通道输出有效电平,否则输出失效电平（0%与 100%波形与 PWM1 同理）**Bits 11 OC1PE:**通道 1 输出比较影子RegisterEnable.

* 0:禁用影子Register
* 1:Enable影子Register

**Bits 10 OC1FE:**通道 1 快速输出Enable.

* 0:禁用快速模式,输出仅在匹配时变化
* 1:Enable快速模式,触发输入相当于匹配事件,直接影响通道输出,不受counter与 CCR 的比较影响**Bits 9-8 CC1S:**捕获比较选择.
* 00:通道配置为 输出模式
* 01:通道配置为输入模式,捕获通道输入映射至通道 1
* 10:通道配置为输入模式,捕获通道输入映射至通道 0
* 11:通道配置为输入模式,捕获通道输入映射至触发输入 TRC **Bits 7 OC0CE:**通道 0 输出比较清除Enable.
* 0:禁用清除功能
* 1:Enable清除功能,ETRF 高电平可以清除通道输出

**Bits 6-4 OC0M:**通道 0 输出比较模式选择.

* 000:冻结模式,通道输出不随比较结果变化
* 001:有效模式,匹配后通道输出有效电平
* 010:失效模式,匹配后通道输出失效电平
* 011:翻转模式,匹配后将翻转通道输出
* 100:强制有效模式,选择该模式后,直接输出有效电平
* 101:强制失效模式,选择该模式后,直接输出失效电平
* 110:PWM1 模式,该模式下,向上计数时,CNT<CCR 时通道输出有效电平,否则输出失效电平；向下计数时,CNT>CCR 时通道输出失效电平,否则输出有效电平（向上计数时,若 CCRx>ARR,

OCxREF 一直输出高电平,若 CCRx==0,OCxREF 一直输出低电平；向下计数时,若 CCRx>ARR,

OCxREF 一直输出高电平,此时 0% PWM 不支持）

* 111:PWM2 模式,该模式下,向上计数时,CNT<CCR 时通道输出失效电平,否则输出有效电平；

向下计数时,CNT>CCR 时通道输出有效电平,否则输出失效电平（0%与 100%波形与 PWM1 同理）**Bits 3 OC0PE:**通道 0 输出比较影子RegisterEnable.

* 0:禁用影子Register
* 1:Enable影子Register

**Bits 2 OC0FE:**通道 0 快速输出Enable.

* 0:禁用快速模式,输出仅在匹配时变化
* 1:Enable快速模式,触发输入相当于匹配事件,直接影响通道输出,不受counter与 CCR 的比较影响**Bits 1-0 CC0S:**捕获比较选择.
* 00:通道配置为 输出模式
* 01:通道配置为输入模式,捕获通道输入映射至通道 0
* 10:通道配置为输入模式,捕获通道输入映射至通道 1
* 11:通道配置为输入模式,捕获通道输入映射至触发输入 TRC

**输入模式时结构如下:**

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **15-12** | **11-10** | **9-8** | **7-4** | **3-2** | **1-0** |
| IC1F | IC1PSC | CC1S | IC0F | IC0PSC | CC0S |
| rw-0h | rw-0h | rw-0h | rw-0h | rw-0h | rw-0h |

**Bits 15-12 IC1F:**通道 1 输入滤波器配置（需配置 CCxS！=0x0 该功能才能生效）.

* 0000:禁用滤波器
* 0001:滤波器采样频率 fsampling=fpclk, 滤波器长度 N=2
* 0010:滤波器采样频率 fsampling=fpclk, 滤波器长度 N=4
* 0011:滤波器采样频率 fsampling=fpclk, 滤波器长度 N=8
* 0100:滤波器采样频率 fsampling=fDTS/2, 滤波器长度 N=6
* 0101:滤波器采样频率 fsampling=fDTS/2, 滤波器长度 N=8
* 0110:滤波器采样频率 fsampling=fDTS/4, 滤波器长度 N=6
* 0111:滤波器采样频率 fsampling=fDTS/4, 滤波器长度 N=8
* 1000:滤波器采样频率 fsampling=fDTS/8, 滤波器长度 N=6
* 1001:滤波器采样频率 fsampling=fDTS/8, 滤波器长度 N=8
* 1010:滤波器采样频率 fsampling=fDTS/16, 滤波器长度 N=5**（未完,接下页）**
* 1011:滤波器采样频率 fsampling=fDTS/16, 滤波器长度 N=6
* 1100:滤波器采样频率 fsampling=fDTS/16, 滤波器长度 N=8
* 1101:滤波器采样频率 fsampling=fDTS/32, 滤波器长度 N=5
* 1110:滤波器采样频率 fsampling=fDTS/32, 滤波器长度 N=6
* 1111:滤波器采样频率 fsampling=fDTS/32, 滤波器长度 N=8 **Bits 11-10 IC1PSC:**通道 1 分频（需配置 CCxS！=0x0 该功能才能生效）.
* 00:不分频
* 01:2 分频
* 10:4 分频
* 11:8 分频**Bits 9-8 CC1S:**捕获比较选择.
* 00:通道配置为 输出模式
* 01:通道配置为输入模式,捕获通道输入映射至通道 1
* 10:通道配置为输入模式,捕获通道输入映射至通道 0
* 11:通道配置为输入模式,捕获通道输入映射至触发输入 TRC **Bits 7-4 IC0F:**通道 0 输入滤波器配置（需配置 CCxS！=0x0 该功能才能生效）.
* 0000:禁用滤波器
* 0001:滤波器采样频率 fsampling=fpclk, 滤波器长度 N=2
* 0010:滤波器采样频率 fsampling=fpclk, 滤波器长度 N=4
* 0011:滤波器采样频率 fsampling=fpclk, 滤波器长度 N=8
* 0100:滤波器采样频率 fsampling=fDTS/2, 滤波器长度 N=6
* 0101:滤波器采样频率 fsampling=fDTS/2, 滤波器长度 N=8
* 0110:滤波器采样频率 fsampling=fDTS/4, 滤波器长度 N=6
* 0111:滤波器采样频率 fsampling=fDTS/4, 滤波器长度 N=8
* 1000:滤波器采样频率 fsampling=fDTS/8, 滤波器长度 N=6
* 1001:滤波器采样频率 fsampling=fDTS/8, 滤波器长度 N=8
* 1010:滤波器采样频率 fsampling=fDTS/16, 滤波器长度 N=5
* 1011:滤波器采样频率 fsampling=fDTS/16, 滤波器长度 N=6
* 1100:滤波器采样频率 fsampling=fDTS/16, 滤波器长度 N=8
* 1101:滤波器采样频率 fsampling=fDTS/32, 滤波器长度 N=5
* 1110:滤波器采样频率 fsampling=fDTS/32, 滤波器长度 N=6
* 1111:滤波器采样频率 fsampling=fDTS/32, 滤波器长度 N=8 **Bits 3-2 IC0PSC:**通道 0 分频（需配置 CCxS！=0x0 该功能才能生效）.
* 00:不分频
* 01:2 分频
* 10:4 分频
* 11:8 分频**Bits 1-0 CC0S:**捕获比较选择.
* 00:通道配置为 输出模式
* 01:通道配置为输入模式,捕获通道输入映射至通道 0
* 10:通道配置为输入模式,捕获通道输入映射至通道 1
* 11:通道配置为输入模式,捕获通道输入映射至触发输入 TRC

##### 22.17.8 GPTIM\_CCMR2

Offset:0x1C

Reset Value:0x0000

**输出模式时结构如下:**

|  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| **15** | **14-12** | **11** | **10** | **9-8** | **7** | **6-4** | **3** | **2** | **1-0** |
| OC3CE | OC3M | OC3PE | OC3FE | CC3S | OC2CE | OC2M | OC2PE | OC2FE | CC2S |
| rw-0h | rw-0h | rw-0h | rw-0h | rw-0h | rw-0h | rw-0h | rw-0h | rw-0h | rw-0h |

**Bits 15 OC3CE:**通道 3 输出比较清除Enable.

* 0:禁用清除功能
* 1:Enable清除功能,ETRF 高电平可以清除通道输出**Bits 14-12 OC3M:**通道 3 输出比较模式选择.
* 000:冻结模式,通道输出不随比较结果变化
* 001:有效模式,匹配后通道输出有效电平
* 010:失效模式,匹配后通道输出失效电平
* 011:翻转模式,匹配后将翻转通道输出
* 100:强制有效模式,选择该模式后,直接输出有效电平
* 101:强制失效模式,选择该模式后,直接输出失效电平
* 110:PWM1 模式,该模式下,向上计数时,CNT<CCR 时通道输出有效电平,否则输出失效电平；向下计数时,CNT>CCR 时通道输出失效电平,否则输出有效电平（向上计数时,若 CCRx>ARR,

OCxREF 一直输出高电平,若 CCRx==0,OCxREF 一直输出低电平；向下计数时,若 CCRx>ARR,

OCxREF 一直输出高电平,此时 0% PWM 不支持）

* 111:PWM2 模式,该模式下,向上计数时,CNT<CCR 时通道输出失效电平,否则输出有效电平；

向下计数时,CNT>CCR 时通道输出有效电平,否则输出失效电平（0%与 100%波形与 PWM1 同理）**Bits 11 OC3PE:**通道 3 输出比较影子RegisterEnable.

* 0:禁用影子Register
* 1:Enable影子Register

**Bits 10 OC3FE:**通道 3 快速输出Enable.

* 0:禁用快速模式,输出仅在匹配时变化
* 1:Enable快速模式,触发输入相当于匹配事件,直接影响通道输出,不受counter与 CCR 的比较影响**Bits 9-8 CC3S:**捕获比较选择.
* 00:通道配置为 输出模式
* 01:通道配置为输入模式,捕获通道输入映射至通道 3
* 10:通道配置为输入模式,捕获通道输入映射至通道 2
* 11:通道配置为输入模式,捕获通道输入映射至触发输入 TRC **Bits 7 OC2CE:**通道 2 输出比较清除Enable.
* 0:禁用清除功能
* 1:Enable清除功能,ETRF 高电平可以清除通道输出

**Bits 6-4 OC2M:**通道 2 输出比较模式选择.

* 000:冻结模式,通道输出不随比较结果变化
* 001:有效模式,匹配后通道输出有效电平
* 010:失效模式,匹配后通道输出失效电平
* 011:翻转模式,匹配后将翻转通道输出
* 100:强制有效模式,选择该模式后,直接输出有效电平
* 101:强制失效模式,选择该模式后,直接输出失效电平
* 110:PWM1 模式,该模式下,向上计数时,CNT<CCR 时通道输出有效电平,否则输出失效电平；向下计数时,CNT>CCR 时通道输出失效电平,否则输出有效电平（向上计数时,若 CCRx>ARR,

OCxREF 一直输出高电平,若 CCRx==0,OCxREF 一直输出低电平；向下计数时,若 CCRx>ARR,

OCxREF 一直输出高电平,此时 0% PWM 不支持）

* 111:PWM2 模式,该模式下,向上计数时,CNT<CCR 时通道输出失效电平,否则输出有效电平；

向下计数时,CNT>CCR 时通道输出有效电平,否则输出失效电平（0%与 100%波形与 PWM1 同理）  **Bits 3 OC2PE:**通道 2 输出比较影子RegisterEnable.

* 0:禁用影子Register
* 1:Enable影子Register

**Bits 2 OC2FE:**通道 2 快速输出Enable.

* 0:禁用快速模式,输出仅在匹配时变化
* 1:Enable快速模式,触发输入相当于匹配事件,直接影响通道输出,不受counter与 CCR 的比较影响  **Bits 1-0 CC2S:**捕获比较选择.
* 00:通道配置为输出模式
* 01:通道配置为输入模式,捕获通道输入映射至通道 2
* 10:通道配置为输入模式,捕获通道输入映射至通道 3
* 11:通道配置为输入模式,捕获通道输入映射至触发输入 TRC

**输入模式时结构如下:**

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **15-12** | **11-10** | **9-8** | **7-4** | **3-2** | **1-0** |
| IC3F | IC3PSC | CC3S | IC2F | IC2PSC | CC2S |
| rw-0h | rw-0h | rw-0h | rw-0h | rw-0h | rw-0h |

**Bits 15-12 IC3F:**通道 3 输入滤波器配置（需配置 CCxS！=0x0 该功能才能生效）.

* 0000:禁用滤波器
* 0001:滤波器采样频率 fsampling=fpclk, 滤波器长度 N=2
* 0010:滤波器采样频率 fsampling=fpclk, 滤波器长度 N=4
* 0011:滤波器采样频率 fsampling=fpclk, 滤波器长度 N=8
* 0100:滤波器采样频率 fsampling=fDTS/2, 滤波器长度 N=6
* 0101:滤波器采样频率 fsampling=fDTS/2, 滤波器长度 N=8
* 0110:滤波器采样频率 fsampling=fDTS/4, 滤波器长度 N=6
* 0111:滤波器采样频率 fsampling=fDTS/4, 滤波器长度 N=8
* 1000:滤波器采样频率 fsampling=fDTS/8, 滤波器长度 N=6
* 1001:滤波器采样频率 fsampling=fDTS/8, 滤波器长度 N=8
* 1010:滤波器采样频率 fsampling=fDTS/16, 滤波器长度 N=5**（未完,接下页）**
* 1011:滤波器采样频率 fsampling=fDTS/16, 滤波器长度 N=6
* 1100:滤波器采样频率 fsampling=fDTS/16, 滤波器长度 N=8
* 1101:滤波器采样频率 fsampling=fDTS/32, 滤波器长度 N=5
* 1110:滤波器采样频率 fsampling=fDTS/32, 滤波器长度 N=6
* 1111:滤波器采样频率 fsampling=fDTS/32, 滤波器长度 N=8 **Bits 11-10 IC3PSC:**通道 3 分频（需配置 CCxS！=0x0 该功能才能生效）.
* 00:不分频
* 01:2 分频
* 10:4 分频
* 11:8 分频**Bits 9-8 CC3S:**捕获比较选择.
* 00:通道配置为 输出模式
* 01:通道配置为输入模式,捕获通道输入映射至通道 3
* 10:通道配置为输入模式,捕获通道输入映射至通道 2
* 11:通道配置为输入模式,捕获通道输入映射至触发输入 TRC **Bits 7-4 IC2F:**通道 2 输入滤波器配置（需配置 CCxS！=0x0 该功能才能生效）.
* 0000:禁用滤波器
* 0001:滤波器采样频率 fsampling=fpclk, 滤波器长度 N=2
* 0010:滤波器采样频率 fsampling=fpclk, 滤波器长度 N=4
* 0011:滤波器采样频率 fsampling=fpclk, 滤波器长度 N=8
* 0100:滤波器采样频率 fsampling=fDTS/2, 滤波器长度 N=6
* 0101:滤波器采样频率 fsampling=fDTS/2, 滤波器长度 N=8
* 0110:滤波器采样频率 fsampling=fDTS/4, 滤波器长度 N=6
* 0111:滤波器采样频率 fsampling=fDTS/4, 滤波器长度 N=8
* 1000:滤波器采样频率 fsampling=fDTS/8, 滤波器长度 N=6
* 1001:滤波器采样频率 fsampling=fDTS/8, 滤波器长度 N=8
* 1010:滤波器采样频率 fsampling=fDTS/16, 滤波器长度 N=5
* 1011:滤波器采样频率 fsampling=fDTS/16, 滤波器长度 N=6
* 1100:滤波器采样频率 fsampling=fDTS/16, 滤波器长度 N=8
* 1101:滤波器采样频率 fsampling=fDTS/32, 滤波器长度 N=5
* 1110:滤波器采样频率 fsampling=fDTS/32, 滤波器长度 N=6
* 1111:滤波器采样频率 fsampling=fDTS/32, 滤波器长度 N=8 **Bits 3-2 IC2PSC:**通道 2 分频（需配置 CCxS！=0x0 该功能才能生效）.
* 00:不分频
* 01:2 分频
* 10:4 分频
* 11:8 分频**Bits 1-0 CC2S:**捕获比较选择.
* 00:通道配置为 输出模式
* 01:通道配置为输入模式,捕获通道输入映射至通道 2
* 10:通道配置为输入模式,捕获通道输入映射至通道 3
* 11:通道配置为输入模式,捕获通道输入映射至触发输入 TRC

##### 22.17.9 GPTIM\_CCER

Offset:0x20

Reset Value:0x0000

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| **15** | **14** | **13** | **12** | **11** | **10** | **9** | **8** |
| CC3NP | RESERVED | CC3P | CC3E | CC2NP | RESERVED | CC2P | CC2E |
| rw-0h | r-0h | rw-0h | rw-0h | rw-0h | r-0h | rw-0h | rw-0h |
| **7** | **6** | **5** | **4** | **3** | **2** | **1** | **0** |
| CC1NP | RESERVED | CC1P | CC1E | CC0NP | RESERVED | CC0P | CC0E |
| rw-0h | r-0h | rw-0h | rw-0h | rw-0h | r-0h | rw-0h | rw-0h |

**Bits 15 CC3NP:**输出反相极性,输出模式时该Bits必须Bits 0,输入模式时参看 CC3P.

**Bits 14 RESERVED:**保留.

**Bits 13 CC3P:**输出极性,须与 CC3NP 共同作用（配置极性最好在模式选择之前,防止内部信号翻转触发未知错误）.

**输出模式:**

* 0:输出有效极性Bits高电平
* 1:输出有效极性Bits低电平

**输入模式,{CC3NP,CC3P}:**

* 00:通道输入上升沿有效（捕获、触发、复Bits、clock 模式）,高电平有效（门控、编码模式）
* 01:通道输入下降沿有效（捕获、触发、复Bits、clock 模式）,低电平有效（门控、编码模式）
* 10:保留
* 11:通道输入上升沿和下降沿均有效（捕获、触发、复Bits、clock 模式）,高电平有效（门控、编码模式）**Bits 12 CC3E:**通道Enable.

**输入模式:**

* 0:禁用捕获
* 1:Enable捕获**输出模式:**
* 0:禁用输出
* 1:Enable输出**Bits 11 CC2NP:**输出反相极性,输出模式时该Bits必须Bits 0,输入模式时参看 CC2P.

**Bits 10 RESERVED:**保留.

**Bits 9 CC2P:**输出极性,须与 CC2NP 共同作用（配置极性最好在模式选择之前,防止内部信号翻转触发未知错误）.

**输出模式:**

* 0:输出有效极性Bits高电平
* 1:输出有效极性Bits低电平

**输入模式,{CC2NP,CC2P}:**

* 00:通道输入上升沿有效（捕获、触发、复Bits、clock 模式）,高电平有效（门控、编码模式）
* 01:通道输入下降沿有效（捕获、触发、复Bits、clock 模式）,低电平有效（门控、编码模式）
* 10:保留
* 11:通道输入上升沿和下降沿均有效（捕获、触发、复Bits、clock 模式）,高电平有效（门控、编码模式）**Bits 8 CC2E:**通道Enable.

**输入模式:**

* 0:禁用捕获
* 1:Enable捕获**输出模式:**
* 0:禁用输出
* 1:Enable输出**Bits 7 CC1NP:**输出反相极性,输出模式时该Bits必须Bits 0,输入模式时参看 CC1P.

**Bits 6 RESERVED:**保留.

**Bits 5 CC1P:**输出极性,须与 CC3NP 共同作用（配置极性最好在模式选择之前,防止内部信号翻转触发未知错误）.

**输出模式:**

* 0:输出有效极性Bits高电平
* 1:输出有效极性Bits低电平

**输入模式,{CC1NP,CC1P}:**

* 00:通道输入上升沿有效（捕获、触发、复Bits、clock 模式）,高电平有效（门控、编码模式）
* 01:通道输入下降沿有效（捕获、触发、复Bits、clock 模式）,低电平有效（门控、编码模式）
* 10:保留
* 11:通道输入上升沿和下降沿均有效（捕获、触发、复Bits、clock 模式）,高电平有效（门控、编码模式）**Bits 4 CC1E:**通道Enable.

**输入模式:**

* 0:禁用捕获
* 1:Enable捕获**输出模式:**
* 0:禁用输出
* 1:Enable输出

**Bits 3 CC0NP:**输出反相极性,输出模式时该Bits必须Bits 0,输入模式时参看 CC0P.

**Bits 2 RESERVED:**保留.

**Bits 1 CC0P:**输出极性,须与 CC0NP 共同作用（配置极性最好在模式选择之前,防止内部信号翻转触发未知错误）.

**输出模式:**

* 0:输出有效极性Bits高电平
* 1:输出有效极性Bits低电平

**输入模式,{CC0NP,CC0P}:**

* 00:通道输入上升沿有效（捕获、触发、复Bits、clock 模式）,高电平有效（门控、编码模式）
* 01:通道输入下降沿有效（捕获、触发、复Bits、clock 模式）,低电平有效（门控、编码模式）
* 10:保留
* 11:通道输入上升沿和下降沿均有效（捕获、触发、复Bits、clock 模式）,高电平有效（门控、编码模式）**Bits 0 CC0E:**通道Enable.

**输入模式:**

* 0:禁用捕获
* 1:Enable捕获**输出模式:**
* 0:禁用输出  1:Enable输出

##### 22.17.10 GPTIM\_CNT

Offset:0x24

Reset Value:0x0000

|  |
| --- |
| **15-0** |
| CNT |
| rw-0h |

**Bits 15-0 CNT:**counter计数值.

##### 22.17.11 GPTIM\_PSC

Offset:0x28

Reset Value:0x0000

|  |
| --- |
| **15-0** |
| PSC |
| rw-0h |

**Bits 15-0 PSC:**clock 分频值为 PSC+1.

##### 22.17.12 GPTIM\_ARR

Offset:0x2C

Reset Value:0xFFFF

|  |
| --- |
| **15-0** |
| PSC |
| rw-FFFFh |

**Bits 15-0 ARR:**counter重装载值.

##### 22.17.13 GPTIM\_CCR0

Offset:0x34

Reset Value:0x0000

|  |
| --- |
| **15-0** |
| CCR0 |
| rw-0h |

**Bits 15-0 CCR0:**输出模式时,该Register保存用户写入的比较值,用于与 CNT 进行比较；输入模式时,该Register保存捕获的值,只读.

##### 22.17.14 GPTIM\_CCR1

Offset:0x38

Reset Value:0x0000

|  |
| --- |
| **15-0** |
| CCR1 |
| rw-0h |

**Bits 15-0 CCR1:**输出模式时,该Register保存用户写入的比较值,用于与 CNT 进行比较；输入模式时,该Register保存捕获的值,只读.

##### 22.17.15 GPTIM\_CCR2

Offset:0x3C Reset Value:0x0000

|  |
| --- |
| **15-0** |
| CCR2 |
| rw-0h |

**Bits 15-0 CCR2:**输出模式时,该Register保存用户写入的比较值,用于与 CNT 进行比较；输入模式时,该Register保存捕获的值,只读.

##### 22.17.16 GPTIM\_CCR3

Offset:0x40 Reset Value:0x0000

|  |
| --- |
| **15-0** |
| CCR3 |
| rw-0h |

**Bits 15-0 CCR3:**输出模式时,该Register保存用户写入的比较值,用于与 CNT 进行比较；输入模式时,该Register保存捕获的值,只读.

##### 22.17.17 GPTIM\_DCR

Offset:0x48 Reset Value:0x0000

|  |  |  |  |
| --- | --- | --- | --- |
| **15-13** | **12-8** | **7-5** | **4-0** |
| RESERVED | DBL | RESERVED | DBA |
| r-0h | rw-0h | r-0h | rw-0h |

**Bits 15-13 RESERVED:**保留.

**Bits 12-8 DBL:**DMA 连续读写长度.

* 00000:1 个传输
* 00001:2 个传输
* 00010:3 个传输
* 00011:4 个传输
* 00100:5 个传输
* 00101:6 个传输
* 00110:7 个传输
* 00111:8 个传输
* 01000:9 个传输**（未完,接下页）**
* 01001:10 个传输
* 01010:11 个传输
* 01011:12 个传输
* 01100:13 个传输
* 01101:14 个传输
* 01110:15 个传输
* 01111:16 个传输
* 10000:17 个传输
* 10001:18 个传输**Bits 7-5 RESERVED:**保留.**Bits 4-0 DBA:**DMA 连续读写基地址.
* 00000:CR1 Register
* 00001:CR2 Register
* 00010:SMCR Register
* 00011:DIER Register
* 00100:SR Register
* 00101:EGR Register
* 00110:CCMR1 Register
* 00111:CCMR2 Register
* 01000:CCER Register
* 01001:CNT Register  01010:PSC Register  01011:ARR Register
* 01100:偏移地址为 0X30 的保留Register
* 01101:CCR0 Register
* 01110:CCR1 Register
* 01111:CCR2 Register
* 10000:CCR3 Register
* 10001:偏移地址为 0X44 的保留Register
* 10010:DCR Register
* 10011:DMAR Register
* 10100:OR Register
* 10101:reserved
* 10110:reserved
* 10111:reserved
* 11000:reserved
* 11001:reserved
* 11010:reserved
* 11011:reserved
* 11100:reserved
* 11101:reserved
* 11110:reserved  11111:reserved

##### 22.17.18 GPTIM\_DMAR

Offset:0x4C Reset Value:0x0000

|  |
| --- |
| **15-0** |
| DMAR |
| rw-0h |

**Bits 15-0 DMAR:**该Register保存当前 DMA 操作的Register的值,例如当前 DMA 需要操作 TIM\_CR2 Register,那么直接操作该地址,便相当于操作 TIM\_CR2 Register.具体代Table那个Register需要参考

DSTEP、DBL 和 DBA 的值.

22.17.19 GPTIM\_OR Offset:0x50 Reset Value:0x0000

**GPTIMER0 时此Register的结构如下:**

|  |  |  |  |
| --- | --- | --- | --- |
| **15-11** | **10-7** | **6-4** | **3-0** |
| RESERVED | ETR\_RMP | TI3\_RMP | TI0\_RMP |
| r-0h | rw-0h | rw-0h | rw-0h |

**Bits 15-11 RESERVED:**保留.

**Bits 10-7 ETR\_RMP:**ETR 重映射.

* 0000:iom
* 0001:comp0
* 0010:comp1
* 0011:xo32k
* 0100:rco48m
* 0101:adcctrl\_awd0
* 0110:adcctrl\_awd1
* 0111:adcctrl\_awd2
* 1000:uart\_rx[0]
* 1001:uart\_rx[1]
* 1010:uart\_rx[2]
* 1011:uart\_rx[3]
* 1100:uart\_rx[4]
* 1101:reserved
* 1110:reserved
* 1111:reserved

**Bits 6-4 TI3\_RMP:**通道 3 重映射.

* 000:iom
* 001:comp0
* 010:comp1
* 011:reserved
* 100:reserved
* 101:reserved
* 110:reserved
* 111:reserved

**Bits 3-0 TI0\_RMP:**通道 0 重映射.

* 0000:iom
* 0001:uart\_rx[0]
* 0010:uart\_rx[1]
* 0011:uart\_rx[2]
* 0100:uart\_rx[3]
* 0101:uart\_rx[4]
* 0110:reserved
* 0111:reserved
* 1000:reserved
* 1001:reserved
* 1010:reserved
* 1011:reserved
* 1100:reserved
* 1101:reserved
* 1110:reserved
* 1111:reserved

###### GPTIMER1 时此Register的结构如下:

|  |  |
| --- | --- |
| **15-2** | **1-0** |
| RESERVED | TI2\_RMP |
| r-0h | rw-0h |

**Bits 15-2 RESERVED:**保留.**Bits 1-0 TI2\_RMP:**通道 2 重映射.

* 00:iom
* 01:TIM3\_CH1
* 10:reserved
* 11:reserved

**GPTIMER2 时此Register的结构如下:**

|  |  |  |  |
| --- | --- | --- | --- |
| **15-10** | **9-7** | **6-5** | **4-0** |
| RESERVED | ETR\_RMP | TI1\_RMP | TI0\_RMP |
| r-0h | rw-0h | rw-0h | rw-0h |

**Bits 15-10 RESERVED:**保留.**Bits 9-7 ETR\_RMP:**ETR 重映射.

* 0000:iom
* 0001:comp0
* 0010:comp1
* 0011:xo32k
* 0100:reserved
* 0101:reserved
* 0110:reserved
* 0111:reserved
* 1000:reserved
* 1001:reserved
* 1010:reserved
* 1011:reserved
* 1100:reserved
* 1101:reserved
* 1110:reserved
* 1111:reserved

**Bits 6-5 TI1\_RMP:**通道 1 重映射.

* 00:iom
* 01:comp1
* 10:reserved
* 11:reserved

**Bits 4-0 TI0\_RMP:**通道 0 重映射.

* 00000:iom
* 00001:xo24m
* 00010:xo32m
* 00011:rco48m
* 00100:xo32k
* 00101:rco32k
* 00110:mco
* 00111:comp0
* 01000:rco3.6m
* 01001:rtc\_alarm1\_happen\_pulse
* 01010:rtc\_alarm0\_happen\_pulse
* 01011:rtc\_cyc\_counter\_pulse
* 01100:reserved**（未完,接下页）**
* 01101:reserved
* 01110:reserved
* 01111:reserved
* 10000:reserved
* 10001:reserved
* 10010:reserved
* 10011:reserved
* 10100:reserved
* 10101:reserved
* 10110:reserved
* 10111:reserved
* 11000:reserved
* 11001:reserved
* 11010:reserved
* 11011:reserved
* 11100:reserved
* 11101:reserved
* 11110:reserved
* 11111:reserved

###### GPTIMER3 时此Register的结构如下:

|  |  |  |
| --- | --- | --- |
| **15-7** | **6-3** | **2-0** |
| RESERVED | ETR\_RMP | TI0\_RMP |
| r-0h | rw-0h | rw-0h |

**Bits 15-7 RESERVED:**保留.**Bits 6-3 ETR\_RMP:**ETR 重映射.

* 0000:iom
* 0001:comp0
* 0010:comp1
* 0011:xo32k
* 0100:uart\_rx[0]
* 0101:uart\_rx[1]
* 0110:uart\_rx[2]
* 0111:uart\_rx[3]
* 1000:uart\_rx[4]
* 1001:reserved
* 1010:reserved
* 1011:reserved
* 1100:reserved
* 1101:reserved
* 1110:reserved
* 1111:reserved

**Bits 2-0 TI0\_RMP:**通道 0 重映射.

* 000:iom
* 001:comp0
* 010:comp1
* 011:uart\_rx[0]
* 100:uart\_rx[1]
* 101:uart\_rx[2]
* 110:uart\_rx[3]
* 111:uart\_rx[4]