UM1000
RTL8721Dx User Manual

This document describes the specification of RTL8721Dx at a system level.

Rev. 1.0

Mar., 2024

Realtek Semiconductor Corp.

No. 2, Innovation Road II, Hsinchu Science Park, Hsinchu 300, Taiwan

Tel.: +886-3-578-0211. Fax: +886-3-577-6047

www.realtek.com

COPYRIGHT

©2024 Realtek Semiconductor Corp. All rights reserved. No part of this document may be reproduced, transmitted, transcribed, stored in a retrieval system, or translated into any language in any form or by any means without the written permission of Realtek Semiconductor Corp.

DISCLAIMER

Please Read Carefully:
Realtek Semiconductor Corp., (Realtek) reserves the right to make corrections, enhancements, improvements and other changes to its products and services. Buyers should obtain the latest relevant information before placing orders and should verify that such information is current and complete.

Reproduction of significant portions in Realtek data sheets is permissible only if reproduction is without alteration and is accompanied by all associated warranties, conditions, limitations, and notices. Realtek is not responsible or liable for such reproduced documentation. Information of third parties may be subject to additional restrictions.

Buyers and others who are developing systems that incorporate Realtek products (collectively, "Customers") understand and agree that Customers remain responsible for using their independent analysis, evaluation and judgment in designing their applications and that Customers have full and exclusive responsibility to assure the safety of Customers' applications and compliance of their applications (and of all Realtek products used in or for Customers' applications) with all applicable regulations, laws and other applicable requirements. Designer represents that, with respect to their applications, Customer has all the necessary expertise to create and implement safeguards that (1) anticipate dangerous consequences of failures, (2) monitor failures and their consequences, and (3) lessen the likelihood of failures that might cause harm and take appropriate actions. Customer agrees that prior to using or distributing any applications that include Realtek products, Customer will thoroughly test such applications and the functionality of such Realtek products as used in such applications.

Realtek's provision of technical, application or other design advice, quality characterization, reliability data or other services or information, including, but not limited to, reference designs and materials relating to evaluation kits, (collectively, "Resources") are intended to assist designers who are developing applications that incorporate Realtek products; by downloading, accessing or using Realtek's Resources in any way, Customer (individually or, if Customer is acting on behalf of a company, Customer's company) agrees to use any particular Realtek Resources solely for this purpose and subject to the terms of this Notice.

Realtek's provision of Realtek Resources does not expand or otherwise alter Realtek's applicable published warranties or warranty disclaimers for Realtek's products, and no additional obligations or liabilities arise from Realtek providing such Realtek Resources. Realtek reserves the right to make corrections, enhancements, improvements and other changes to its Realtek Resources. Realtek has not conducted any testing other than that specifically described in the published documentation for a particular Realtek Resource.

Customer is authorized to use, copy and modify any individual Realtek Resource only in connection with the development of applications that include the Realtek product(s) identified in such Realtek Resource. No other license, express or implied, by estoppel or otherwise to any other Realtek intellectual property right, and no license to any technology or intellectual property right of Realtek or any third party is granted herein, including but not limited to any patent right, copyright, mask work right, or other intellectual property right relating to any combination, machine, or process in which Realtek products or services are used. Information regarding or referencing third-party products or services does not constitute a license to use such products or services, or a warranty or endorsement thereof. Use of Realtek Resources may require a license from a third party under the patents or other intellectual property of the third party, or a license from Realtek under the patents or other Realtek's intellectual property.

Realtek's Resources are provided "as is" and with all faults. Realtek disclaims all other warranties or representations, express or implied, regarding resources or use thereof, including but not limited to accuracy or completeness, title, any epidemic failure warranty and any implied warranties of merchantability, fitness for a particular purpose, and non-infringement of any third party intellectual property rights.

Realtek shall not be liable for and shall not defend or indemnify Customer against any claim, including but not limited to any infringement claim that related to or is based on any combination of products even if described in Realtek Resources or otherwise. In no event shall Realtek be liable for any actual, direct, special, collateral, indirect, punitive, incidental, consequential or exemplary damages in connection with or arising out of Realtek's Resources or use thereof, and regardless of whether Realtek has been advised of the possibility of such damages. Realtek is not responsible for any failure to meet such industry standard requirements.

Where Realtek specifically promotes products as facilitating functional safety or as compliant with industry functional safety standards, such products are intended to help enable customers to design and create their own applications that meet applicable functional safety standards and requirements. Using products in an application does not by itself establish any safety features in the application. Customers must ensure compliance with safety-related requirements and standards applicable to their applications. Designer may not use any Realtek products in life-critical medical equipment unless authorized officers of the parties have executed a special contract specifically governing such use. Life-critical medical equipment is medical equipment where failure of such equipment would cause serious bodily injury or death. Such equipment includes, without limitation, all medical devices identified by the U.S.FDA as Class III devices and equivalent classifications outside the U.S.

Customers agree that it has the necessary expertise to select the product with the appropriate qualification designation for their applications and that proper product selection is at Customers' own risk. Customers are solely responsible for compliance with all legal and regulatory requirements in connection with such selection.

Customer will fully indemnify Realtek and its representatives against any damages, costs, losses, and/or liabilities arising out of Designer's

non-compliance with the terms and provisions of this Notice.

TRADEMARKS

Realtek is a trademark of Realtek Semiconductor Corporation. Other names mentioned in this document are trademarks/registered trademarks of their respective owners.

USING THIS DOCUMENT

This document is intended for the software engineer's reference and provides detailed programming information.

Though every effort has been made to ensure that this document is current and accurate, more information may have become available subsequent to the production of this guide.

All information provided in this document is subject to legal disclaimers.

© REALTEK 2024. All rights reserved.

Contents

Contents **4**

Conventions **14**

1 Product Overview **15**

1.1 General Description 15
1.2 Block Diagram 15

2 Memory Organization **17**

2.1 Introduction 17
2.2 Memory Map and Register Boundary Address 19
2.3 ROM 20
2.4 On-chip SRAM 20
2.5 SPI Flash Memory 20
2.6 PSRAM 20

3 Memory Protection Unit (MPU) **21**

4 Cache **22**

4.1 KM0 Cache 22
4.2 KM4 Cache 22

5 Interrupt Controller **23**

5.1 Nested Vector Interrupt Controller (NVIC) 23
5.2 Interrupt List 23

6 Hardware Crypto Engine (IPsec) **25**

6.1 Application Scenario 25
6.2 Features 25
6.3 Architecture 25
6.3.1 IPsec Block Diagram 25
6.3.2 DMA Block Diagram 27
6.4 Registers 28
6.4.1 AES Registers 28
6.4.2 Hash Registers 42

7 One-Time-Programmable Controller (OTPC) **57**

7.1 Introduction 57
7.2 OTP Layout and Security Control 57
7.2.1 Logical Zone 58
7.2.2 Security Zone 69
7.2.3 User-defined Zone 78
7.2.4 ROM Code Patch Zone 78
7.2.5 Hidden Physical Zone 78

8 Chip Enable **80**

8.1 Overview 80
8.2 Debounce Function 80
8.3 Level Reset Mode 81
8.4 Interrupt Reset Mode 81
8.5 Pulse Reset Mode 83
8.6 Registers 83
8.6.1 REG_AON_CHIPEN_CTRL 83

All information provided in this document is subject to legal disclaimers.
© REALTEK 2024. All rights reserved.

8.6.2 REG_AON_CHIPEN_ISR 85

9 Reset and Clock Control (RCC) 86

9.1 Reset Control 86
9.1.1 Introduction 86
9.1.2 Reset Diagram 86
9.1.3 Reset Types 86
9.1.4 Reset Domains 87
9.2 Clock Control 87
9.2.1 Clock Tree 87
9.2.2 Clock Source 88
9.3 Registers 88
9.3.1 System Reset Registers 88
9.3.2 Boot Reason Register 89

10 Inter-Processor Communication (IPC) 90

10.1 Introduction 90
10.2 Features 90
10.3 Architecture 90
10.4 Functional Description 90
10.4.1 Message Transmit 90
10.4.2 Interrupts 91
10.4.3 General Usage 92
10.4.4 Hardware Semaphore 94
10.5 Registers 95
10.5.1 REG_IPC_DATA 96
10.5.2 REG_IPC_ISR 96
10.5.3 REG_IPC_IMR 98
10.5.4 REG_IPC_ICR 98
10.5.5 REG_IPC_SEM_0 99
10.5.6 REG_IPC_SEM_1 99
10.5.7 REG_IPC_DUMMY 99

11 I/O Control 100

11.1 Features 100
11.2 Functional Description 100
11.2.1 I/O Types 100
11.2.2 IOCTRL Register 101
11.2.3 Function ID 101
11.2.4 I/O Pull Resistor Control 102
11.2.5 I/O Driving Strength 102
11.2.6 I/O Schmitt Trigger 103
11.2.7 Slew Rate Control 103
11.2.8 I/O Shutdown & RESET 104
11.2.9 Open Drain Mode 105
11.3 pinmapcfg 105
11.3.1 Normal GPIO 106
11.3.2 LOGUART 107
11.3.3 SWD 107
11.3.4 Flash Pin 107

12 General Purpose Input/Output (GPIO) 108

12.1 Introduction 108
12.2 Features 108
12.3 Functional Description 108
12.3.1 Data Read and Write Control 108
12.3.2 Interrupts 110

All information provided in this document is subject to legal disclaimers.
© REALTEK 2024. All rights reserved.

12.4 Registers 117
12.4.1 REG_GPIO_DR 118
12.4.2 REG_GPIO_DDR 118
12.4.3 REG_GPIO_INT_EN 118
12.4.4 REG_GPIO_INT_MASK 119
12.4.5 REG_GPIO_INT_TYPE 119
12.4.6 REG_GPIO_INT_POLARITY 119
12.4.7 REG_GPIO_INT_STATUS 120
12.4.8 REG_GPIO_INT_STATUS_RAW 120
12.4.9 REG_GPIO_DEBOUNCE 120
12.4.10 REG_GPIO_INT_EOI 121
12.4.11 REG_GPIO_EXT_PORT 121
12.4.12 REG_GPIO_ITN_LS_SYNC 121
12.4.13 REG_GPIO_ID_CODE 121
12.4.14 REG_GPIO_INT_BOTHEDGE 122
12.4.15 REG_GPIO_VER_ID_CODE 122
12.5 Programming Considerations 122

13 Direct Memory Access Controller (DMAC) 123

13.1 Introduction 123
13.2 General Description 123
13.2.1 Basic Definitions 123
13.2.2 Features 124
13.3 Functional Description 125
13.3.1 Block Flow Controller and Transfer Type 125
13.3.2 Handshaking Interface 125
13.3.3 Memory Peripherals 126
13.3.4 Basic Interface Definitions 126
13.3.5 Setting up Transfers 126
13.3.6 Generating Requests for the Master Bus Interface 126
13.3.7 Secure Mode 127
13.4 Registers 127
13.4.1 Register Memory Map 128
13.4.2 Registers and Field Descriptions 132
13.5 Programming the DMAC 156
13.5.1 DMA Transfer Types 156
13.5.2 Disabling a Channel Prior to Transfer Completion 160

14 Unmanned Peripheral System (UPS) 162

14.1 Introduction 162
14.2 Features 162
14.3 Block Diagram 162
14.4 Functional Description 162
14.4.1 Timer 162
14.4.2 Deglitch Filter 163
14.4.3 Edge Detection 163
14.4.4 Signal Reverse 163
14.4.5 Dimming Signal Types 164
14.5 Registers 164
14.5.1 REG_UPS_CTRL 164

15 General Timers 166

15.1 Basic Timer 166
15.1.1 Introduction 166
15.1.2 Features 166
15.1.3 Block Diagram 166
15.1.4 Functional Description 166

All information provided in this document is subject to legal disclaimers.
© REALTEK 2024. All rights reserved.

15.2 Capture Timer 167
15.2.1 Introduction 167
15.2.2 Features 167
15.2.3 Block Diagram 168
15.2.4 Functional Description 168
15.3 PWM Timer 170
15.3.1 Introduction 170
15.3.2 Features 170
15.3.3 Block Diagram 170
15.3.4 Functional Description 171
15.4 Registers 186
15.4.1 Basic Timer Registers 187
15.4.2 Capture Timer Registers 190
15.4.3 PWM Timer Registers 195

16 Debug Timer 204

16.1 Features 204
16.2 Functional Description 204
16.3 Registers 204
16.3.1 REG_DBGT_CTRL 204
16.3.2 REG_DBGT_CRV 205
16.3.3 REG_DBGT_ATOM_INC 205
16.3.4 REG_DBGT_ATOM 205
16.3.5 REG_DBGT_SCRATCH 205
16.3.6 REG_DBGTIM_DUMMY 206

17 Real-time Clock (RTC) 207

17.1 Product Overview 207
17.1.1 Introduction 207
17.1.2 Features 207
17.2 Functional Description 207
17.2.1 Clock and Prescaler 207
17.2.2 Programmable Alarm 208
17.2.3 Write Protection 208
17.2.4 Digital Calibration 208
17.2.5 Day Threshold Program 209
17.2.6 Daylight Saving 209
17.2.7 Periodic Auto-wakeup 209
17.3 Read and Write Timing 209
17.4 Registers 210
17.4.1 REG_RTC_TR 210
17.4.2 REG_RTC_CR 211
17.4.3 REG_RTC_ISR 213
17.4.4 REG_RTC_PRER 214
17.4.5 REG_RTC_CALIBR 214
17.4.6 REG_RTC_ALMR1L 215
17.4.7 REG_RTC_ALMR1H 216
17.4.8 REG_RTC_WPR 216
17.4.9 REG_RTC_YEAR 216
17.4.10 REG_RTC_WUTR 217
17.4.11 REG_RTC_DUMMY 217
17.5 Operation Flow 217
17.5.1 Initializing the Calendar 217
17.5.2 Configuring Alarm 218
17.5.3 Configuring Calibration 218
17.5.4 Programming the Wakeup Timer 218
17.5.5 Daylight Saving Time 219

All information provided in this document is subject to legal disclaimers.
© REALTEK 2024. All rights reserved.

18 Backup Register 220

19 Watchdog Timer 221

19.1 Introduction 221
19.2 Independent Watchdog Timer (IWDG) 221
19.2.1 Features 221
19.2.2 Clock and Power 221
19.2.3 Reset Architecture 222
19.3 System Watchdog Timer (WDG) 222
19.3.1 Features 222
19.3.2 Clock and Power 222
19.4 Functional Description 222
19.4.1 Workflow 222
19.4.2 Initialization 223
19.4.3 Operation Flow 223
19.5 Registers 225
19.5.1 IWDG Registers 225
19.5.2 SWDG Registers 226

20 Serial Peripheral Interface (SPI) 229

20.1 Product Overview 229
20.1.1 Features 229
20.1.2 Block Diagram 229
20.2 Functional Description 230
20.2.1 Introduction 230
20.2.2 Transfer Modes 234
20.2.3 Operation Modes 235
20.2.4 DMA Controller Interface 237
20.3 Registers 238
20.3.1 REG_CTRLR0 239
20.3.2 REG_CTRLR1 241
20.3.3 REG_SSIENR 241
20.3.4 REG_SER 242
20.3.5 REG_BAUDR 242
20.3.6 REG_TXFTLR 242
20.3.7 REG_RXFTLR 243
20.3.8 REG_TXFLR 243
20.3.9 REG_RXFLR 244
20.3.10 REG_SR 244
20.3.11 REG_IMR 245
20.3.12 REG_ISR 246
20.3.13 REG_RISR 247
20.3.14 REG_TXOICR 248
20.3.15 REG_RXOICR 248
20.3.16 REG_RXUICR 248
20.3.17 REG_MSTICR_FAEICR 249
20.3.18 REG_ICR 249
20.3.19 REG_DMACR 249
20.3.20 REG_DMATDLR 250
20.3.21 REG_DMARDLR 250
20.3.22 REG_TXUICR 251
20.3.23 REG_SSRICR 251
20.3.24 REG_DRx 251
20.3.25 REG_RX_SAMPLE_DLY 252
20.3.26 REG_SPI_DUMMY 253

21 Inter-integrated Circuit (I2C) Interface 254

All information provided in this document is subject to legal disclaimers.
© REALTEK 2024. All rights reserved.

21.1 Introduction 254
21.2 Features 254
21.3 Functional Description 254
21.3.1 Overview 254
21.3.2 I2C Terminology 255
21.3.3 I2C Behavior 255
21.3.4 I2C Protocols 257
21.3.5 Transmit FIFO Management and START, STOP and RESTART Generation 260
21.3.6 Clock Synchronization 262
21.3.7 Multiple Master Arbitration 262
21.3.8 SDA/SCL Filter 263
21.3.9 IC_CLK Frequency Configuration 263
21.3.10 Programmable SDA Hold Time 266
21.3.11 Slave Mode Dual Own Address 266
21.4 Operation Flow 266
21.4.1 Slave Mode Operation 266
21.4.2 Master Mode Operation 269
21.5 Registers 271
21.5.1 REG_IC_CON 272
21.5.2 REG_IC_TAR 272
21.5.3 REG_IC_SAR 273
21.5.4 REG_IC_HS_MAR 273
21.5.5 REG_IC_DATA_CMD 273
21.5.6 REG_IC_SS_SCL_HCNT 274
21.5.7 REG_IC_SS_SCL_LCNT 274
21.5.8 REG_IC_FS_SCL_HCNT 275
21.5.9 REG_IC_FS_SCL_LCNT 275
21.5.10 REG_IC_HS_SCL_HCNT 275
21.5.11 REG_IC_HS_SCL_LCNT 276
21.5.12 REG_IC_INTR_STAT 276
21.5.13 REG_IC_INTR_MASK 277
21.5.14 REG_IC_RAW_INTR_STAT 278
21.5.15 REG_IC_RX_TL 279
21.5.16 REG_IC_TX_TL 279
21.5.17 REG_IC_CLR_INTR 279
21.5.18 REG_IC_CLR_RX_UNDER 280
21.5.19 REG_IC_CLR_RX_OVER 280
21.5.20 REG_IC_CLR_TX_OVER 280
21.5.21 REG_IC_CLR_RD_REQ 281
21.5.22 REG_IC_CLR_TX_ABRT 281
21.5.23 REG_IC_CLR_RX_DONE 281
21.5.24 REG_IC_CLR_ACTIVITY 281
21.5.25 REG_IC_CLR_STOP_DET 282
21.5.26 REG_IC_CLR_START_DET 282
21.5.27 REG_IC_CLR_GEN_CALL 282
21.5.28 REG_IC_ENABLE 283
21.5.29 REG_IC_STATUS 283
21.5.30 REG_IC_TXFLR 284
21.5.31 REG_IC_RXFLR 284
21.5.32 REG_IC_SDA_HOLD 284
21.5.33 REG_IC_TX_ABRT_SOURCE 284
21.5.34 REG_IC_SLV_DATA_NACK_ONLY 286
21.5.35 REG_IC_DMA_CR 286
21.5.36 REG_IC_DMA_TDLR 286
21.5.37 REG_IC_DMA_RDLR 287
21.5.38 REG_IC_SDA_SETUP 287
21.5.39 REG_IC_ACK_GENERAL_CALL 287

All information provided in this document is subject to legal disclaimers.
© REALTEK 2024. All rights reserved.

21.5.40 REG_IC_ENABLE_STATUS 287
21.5.41 REG_IC_DMA_CMD 288
21.5.42 REG_IC_DMA_DATA_LEN 289
21.5.43 REG_IC_DMA_MODE 289
21.5.44 REG_IC_DEBUG_SEL 289
21.5.45 REG_IC_OUT_SMP_DLY 290
21.5.46 REG_IC_CLR_ADDR_MATCH 290
21.5.47 REG_IC_CLR_DMA_DONE 290
21.5.48 REG_IC_FILTER 290
21.5.49 REG_IC_SAR2 291
21.5.50 REG_IC_COMP_VERSION 291
21.5.51 REG_IC_DUMMY 291

22 Universal Asynchronous Receiver/Transmitter (UART) 293

22.1 Overview 293
22.1.1 Features 293
22.1.2 Block Diagram 293
22.2 Functional Description 294
22.2.1 UART Serial Protocol 294
22.2.2 IrDA (SIR) Protocol 294
22.2.3 Interrupt 295
22.2.4 Auto Flow Control 296
22.2.5 DMA Flow Control 297
22.2.6 Rx Path Monitor Function 297
22.2.7 Clock Structure of Rx Path 298
22.2.8 UART Rx Debounce 298
22.2.9 UART2_BT Used as Normal UART 298
22.3 Registers 299
22.3.1 REG_UART_DLL 299
22.3.2 REG_UART_IER 299
22.3.3 REG_UART_IIR 300
22.3.4 REG_UART_LCR 300
22.3.5 REG_UART_MCR 301
22.3.6 REG_UART_LSR 302
22.3.7 REG_UART_MSR 303
22.3.8 REG_UART_SCR 304
22.3.9 REG_UART_STS 304
22.3.10 REG_UART_RBR_OR_UART_THR 305
22.3.11 REG_UART_MISCR 305
22.3.12 REG_UART_SIR_TX_PWC0 306
22.3.13 REG_UART_SIR_RX_PFC 306
22.3.14 REG_UART_BAUD_MON 306
22.3.15 REG_UART_DBGR 307
22.3.16 REG_UART_RX_PATH_CTRL 307
22.3.17 REG_UART_MON_BAUD_CTRL 307
22.3.18 REG_UART_MON_BAUD_STS 308
22.3.19 REG_UART_MON_CYC_NUM 308
22.3.20 REG_UART_RX_BYTE_CNT 309
22.3.21 REG_UART_FCR 309
22.3.22 REG_UART_ICR 310
22.3.23 REG_UART_RXDBCR 310
22.3.24 REG_UART_DUMMY 310

23 Infrared Radiation (IR) 311

23.1 Introduction 311
23.2 Features 312
23.3 Block diagram 312

All information provided in this document is subject to legal disclaimers.
© REALTEK 2024. All rights reserved.

23.4 Functional Description 313
23.4.1 Scaler 313
23.4.2 Data Format 313
23.4.3 Tx Compensation Mechanism 314
23.4.4 Rx Glitch Filter 314
23.5 Interrupt 315
23.6 Registers 315
23.6.1 REG_IR_CLK_DIV 315
23.6.2 REG_IR_TX_CONFIG 316
23.6.3 REG_IR_TX_SR 317
23.6.4 REG_IR_TX_COMPE_DIV 317
23.6.5 REG_IR_TX_INT_CLR 318
23.6.6 REG_IR_TX_FIFO 318
23.6.7 REG_IR_RX_CONFIG 319
23.6.8 REG_IR_RX_SR 320
23.6.9 REG_IR_RX_INT_CLR 321
23.6.10 REG_IR_RX_CNT_INT_SEL 322
23.6.11 REG_IR_RX_FIFO 322
23.6.12 REG_IR_VERSION 322
23.6.13 REG_IR_DUMMY 323

24 Light Emitting Diode Controller (LEDC) 324

24.1 Introduction 324
24.2 Features 324
24.3 Functional Description 324
24.3.1 Block Diagram 324
24.3.2 LEDC Data Transfer Format 324
24.3.3 LEDC Interrupts 327
24.4 Registers 327
24.4.1 REG_LEDC_CTRL_REG 328
24.4.2 REG_LED_T01_TIMING_CTRL_REG 329
24.4.3 REG_LEDC_DATA_FINISH_CNT_REG 329
24.4.4 REG_LED_RESET_TIMING_CTRL_REG 330
24.4.5 REG_LEDC_WAIT_TIME0_CTRL_REG 330
24.4.6 REG_LEDC_WAIT_TIME1_CTRL_REG 331
24.4.7 REG_LEDC_DATA_REG 331
24.4.8 REG_LEDC_DMA_CTRL_REG 332
24.4.9 REG_LED_INTERRUPT_CTRL_REG 332
24.4.10 REG_LED_INT_STS_REG 333
24.4.11 REG_LEDC_DEBUG_SEL_REG 334
24.4.12 REG_DUMMY_F0 334
24.4.13 REG_DUMMY_F4 334

25 SDIO Device 335

25.1 Features 335
25.2 Architecture 335
25.3 Functional Description 335
25.3.1 SDIO Device Interface Control 335
25.3.2 DMA Transfer 336
25.4 Register 339
25.4.1 SDIO Local Control Registers 339
25.4.2 SDIO Control Registers 344
25.4.3 CCCR Bit Definition 351

26 Key-Scan 355

26.1 Overview 355
26.2 Features 355

All information provided in this document is subject to legal disclaimers.
© REALTEK 2024. All rights reserved.

26.3 Functional Description 355
26.3.1 Block Diagram 355
26.3.2 Work Principle 356
26.3.3 FIFO Mechanism 359
26.3.4 Clock Configuration 360
26.3.5 Low Power Mode 360
26.3.6 Shadow Key Problem 361
26.3.7 Stuck Key Problem 362
26.4 Registers 363
26.4.1 REG_KS_CLK_DIV 363
26.4.2 REG_KS_TIM_CFG0 364
26.4.3 REG_KS_TIM_CFG1 364
26.4.4 REG_KS_CTRL 364
26.4.5 REG_KS_FIFO_CFG 365
26.4.6 REG_KS_COL_CFG 365
26.4.7 REG_KS_ROW_CFG 366
26.4.8 REG_KS_DATA_NUM 366
26.4.9 REG_KS_DATA 367
26.4.10 REG_KS_IMR 367
26.4.11 REG_KS_ICR 368
26.4.12 REG_KS_ISR 368
26.4.13 REG_KS_ISR_RAW 369
26.4.14 REG_KS_DISCHARGED_CTRL 369
26.4.15 REG_KS_STUCK_CFG0 370
26.4.16 REG_KS_STUCK_CFG1 370
26.4.17 REG_KS_STUCK_CFG2 370
26.4.18 REG_KS_STUCK_KEY_STATUS1 371
26.4.19 REG_KS_STUCK_KEY_STATUS2 371
26.4.20 REG_KS_STUCK_KEY_ROW_STATUS1 372
26.4.21 REG_KS_DUMMY 372

27 Cap-Touch Controller (CTC) 373

27.1 Introduction 373
27.2 Features 373
27.3 Architecture 374
27.3.1 CTC Block Diagram 374
27.3.2 Capacitive Sensors 374
27.3.3 Cap-Touch Controller 374
27.3.4 CTC Clock Divider 375
27.4 Functional Description 376
27.4.1 Sample Range 376
27.4.2 Debug Mode 376
27.4.3 Auto-scan 376
27.4.4 Scan Period 377
27.4.5 Sensitivity 379
27.4.6 Touch Judgement 379
27.4.7 Active Noise Immunity 380
27.4.8 Environment Tracking and Calibration (ETC) 382
27.4.9 Debounce 383
27.4.10 FIFO 384
27.4.11 Interrupt 384
27.4.12 Cap-Touch Initialization 384
27.4.13 Calibration 385
27.4.14 Power Consumption 385
27.5 Registers 385
27.5.1 REG_CTC_CTRL 386
27.5.2 REG_SCAN_PERIOD 387

All information provided in this document is subject to legal disclaimers.
© REALTEK 2024. All rights reserved.

27.5.3 REG_ETC_CTRL 387
27.5.4 REG_SNR_INF 388
27.5.5 REG_DEBUG_MODE_CTRL 388
27.5.6 REG_RAW_CODE_FIFO_STATUS 389
27.5.7 REG_RAW_CODE_FIFO_READ 389
27.5.8 REG_INTERRUPT_ENABLE 389
27.5.9 REG_INTERRUPT_STATUS 390
27.5.10 REG_RAW_INTERRUPT_STATUS 391
27.5.11 REG_INTERRUPT_ALL_CLR 391
27.5.12 REG_INTERRUPT_STATUS_CLR 392
27.5.13 REG_DEBUG_SEL 392
27.5.14 REG_DEBUG_PORT 393
27.5.15 REG_ECO_USE0 393
27.5.16 REG_ECO_USE1 393
27.5.17 REG_CTC_COMP_VERSION 393
27.5.18 REG_CHx_CTRL 394
27.5.19 REG_CHx_NOISE_TH 394
27.5.20 REG_CHx_MBIAS_ATH 395
27.5.21 REG_CHx_DATA_INF 395
27.5.22 REG_CTC_SAMPLE_CTRL 396
27.5.23 REG_CTC_HW_TESTMODE 396
27.5.24 REG_CTC_FILTER_CTRL0 397
27.5.25 REG_CTC_FILTER_CTRL1 397
27.5.26 REG_CTC_DUMMY 398

28 Analog-to-Digital Converter (ADC) 399

28.1 Introduction 399
28.2 Features 399
28.3 Functional Description 400
28.3.1 SAR ADC Principle 400
28.3.2 Clock Generation 401
28.3.3 Input Mode 401
28.3.4 Operation Mode 402
28.4 Registers 404

29 Audio 422

29.1 Overview 422
29.2 Clock Domain 422
29.3 SPORT 423
29.3.1 SPORT Data Path 423
29.3.2 SPORT Function 424
29.3.3 SPORT Register 431
29.4 Digital Microphone Interface 450
29.4.1 General Description 450
29.4.2 DMIC Interface Data Path 450
29.4.3 DMIC Interface Function 450
29.4.4 DMIC Interface Register 452

Revision History 470

All information provided in this document is subject to legal disclaimers.
© REALTEK 2024. All rights reserved.

Conventions

Abbreviations for MCUs

The following abbreviations apply to indicate the MCUs/platforms of Realtek.

RTL8721Dx	32-bit MCU family including RTL8721DAx/RTL8721DCx/RTL8721DGx series
Real-M300 (KM4)	Arm® Cortex®-M55 instruction set compatible core based on Armv8.1-M mainline architecture, running at a frequency of up to 345MHz.
Real-M200 (KM0)	Arm® Cortex®-M23 instruction set compatible core based on Armv8-M baseline architecture, running at a frequency of up to 115MHz.
AP	Application Processor, designed for user application.
NP	Network Processor, designed for network protocol, provides network and power management services to AP.

Abbreviations for Registers

The following abbreviations are used to illustrate the access for registers, and they represent the same meaning in different registers.

R/W	Read/write. Software can read and write to this bit.
R	Read-only. Software can only read this bit.
W	Write-only. Software can only write to this bit, reading this bit returns the reset value.
R/W/EC	EC: external clear. Software can read and write to this bit, hardware can also clear this bit.
R/W/ES	ES: external set. Software can read and write to this bit, hardware can also write to this bit.
R/W1C	Software can read as well as clear this bit by writing 1. Writing 0 has no effect on the bit value.
R/W1only	Software can read and write 1 to this bit only. Writing 0 has no effect on the bit value.
W1C	Software can clear this bit by writing 1.
WA0	Hardware will automatically clear this bit to 0 when software has written to this bit.
REG_NAME[n]	The bit number n within the REG_NAME register.
REG_NAME[m:n]	The register bit ranges, which are indicated by the left-most bit number m and the right-most bit number n. For example, REG_WDG_MKEYR[15:0] refers to the lower half word in the WDG_MKEYR register.
N/A	Used for the reserved bit which should not be concerned by users.

All information provided in this document is subject to legal disclaimers.
© REALTEK 2024. All rights reserved.

1 Product Overview

1.1 General Description

The RTL8721Dx (including RTL8721DAx/RTL8721DCx/RTL8721DGx series) is a low-power dual-band microcontroller integrating a high-performance MCU (Armv8.1-M, Cortex-M55 instruction set compatible) called Real-M300 and a low-power MCU (Armv8-M, Cortex-M23 instruction set compatible) called Real-M200. It is designed to achieve enhanced power and RF performance and low-power consumption, featuring all the characteristics of low-power chips, such as fine-grained clock gating, multiple power modes, and dynamic power scaling.

The Real-M300 (or KM4 thereafter), acting as application processor (AP), is a 3-staged pipelined 32-bit high-performance processor that bases on Armv8.1-M mainline architecture supporting Arm Cortex-M55 instruction set compatible, running at a frequency of up to 345MHz. It offers system enhancements such as enhanced debug features, floating-point computation, Digital Signal Processing (DSP) extension instructions, and a high level of support block integration for high-performance, deeply embedded applications. The TrustZone-M security technology provides hardware-enforced isolation between the Trusted and Non-Trusted resources on the devices, while maintaining efficient exception handling and determinism.

The Real-M200 (or KM0 thereafter), acting as network processor (NP), is a 2-staged pipelined 32-bit low-power processor that bases on Armv8-M baseline architecture supporting Cortex-M23 instruction set compatible, running at a frequency of up to 115MHz. It is an energy-efficient and easy-to-use processor with a simple instruction set and reduced code size, and is code- and tool-compatible with the KM4 processor. It is intended for operations requiring fast response and low power consumption features, such as power management and network protocol processing.

The RTL8721Dx is a dual-band (2.4GHz and 5GHz) communication controller that integrates the specifications of Wi-Fi (Wi-Fi 4) and Bluetooth (Bluetooth 5.0). It supports 802.11 a/b/g/n wireless LAN (WLAN) network with 40MHz bandwidth. It consists of WLAN MAC, a 1T1R capable WLAN baseband, RF, and Bluetooth, providing complete Wi-Fi and Bluetooth functionalities.

A variety of peripheral interfaces, including UART, SPI, QSPI/OSPI, I2C, LEDC, etc., as well as sensor controllers (such as ADC, Cap-Touch, and Key-Scan) are integrated into RTL8721Dx devices. High-speed connectivity interfaces, SDIO and USB, are also provided. Besides, the RTL8721Dx has audio features with a dedicated digital microphone (DMIC) interface and I2S. Abundant general-purpose I/O (GPIOs) can be configured to different functions according to different IoT (Internet of Things) applications flexibly. The user-friendly development kits (SDK and HDK) are provided to customers for developing applications.

The RTL8721Dx also incorporates high-speed memories with on-chip SRAM and stacked Flash or PSRAM. A dedicated SPI Flash controller provides a flexible and efficient way to access NOR Flash (e.g., byte and block access). A multilayer AXI bus interconnect supports internal and external memory access.

The RTL8721Dx family offers devices in different packages ranging from 48 pins to 100 pins. The included peripherals change with the device.

1.2 Block Diagram

The functional block diagram is summarized in *Figure 1-1*. This diagram provides a view of the chip's major functional components and core complexes.

All information provided in this document is subject to legal disclaimers.

© REALTEK 2024. All rights reserved.

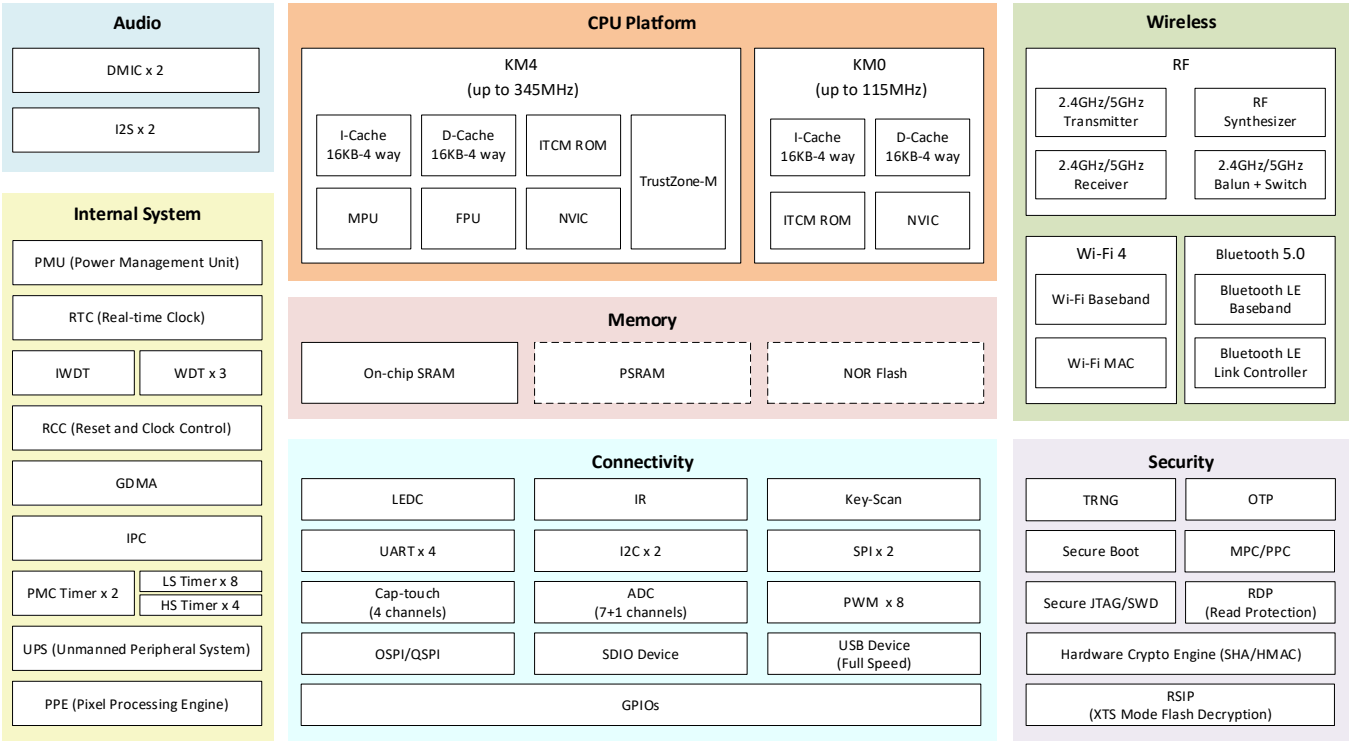


Figure 1-1 Block diagram

All information provided in this document is subject to legal disclaimers.
© REALTEK 2024. All rights reserved.

2 Memory Organization

2.1 Introduction

The RTL8721Dx incorporates several distinct address regions. Program memory, data memory, registers, and I/O ports are organized within the same linear 4G bytes address space. The bytes are coded in Little-Endian format.

The addressable space is divided into multiple main blocks, including code, SRAM, peripherals and other pre-defined regions, as shown in *Table 2-1* and *Figure 2-1*. All the areas that are not allocated to on-chip memories and peripherals are considered as “RSVD” (reserved).

Table 2-1 Address space main blocks

Base address	End address	Size (bytes)	Function		TrustZone*
0x0000_0000	0x0007_FFFF	512K	KM4 ROM (KM4 only) KM0 ROM (KM0 only)	BOOT ROM	-
0x0008_0000	0x000F_FFFF	512K	RSVD		
0x0010_0000	0x07FF_FFFF	127M	RSVD	Flash	
0x0800_0000	0x0FFF_FFFF	128M	SPI NOR Flash		
0x1000_0000	0x1FFF_FFFF	256M	RSVD		-
0x2000_0000	0x2007_FFFF	512K	SRAM	SRAM	
0x2008_0000	0x200F_FFFF	512K	Shared SRAM		
0x2010_0000	0x2FFF_FFFF	255M	RSVD		
0x3000_0000	0x3FFF_FFFF	256M	TrustZone secure address (SRAM)		Secure
0x4000_0000	0x40FF_FFFF	16M	High-Speed peripherals group	Peripherals	-
0x4100_0000	0x41FF_FFFF	16M	Low-Speed peripherals group		
0x4200_0000	0x4FFF_FFFF	224M	RSVD		
0x5000_0000	0x5FFF_FFFF	256M	TrustZone secure address (Peripherals)		Secure
0x6000_0000	0x6FFF_FFFF	256M	PSRAM	DRAM	-
0x7000_0000	0x7FFF_FFFF	256M	TrustZone secure address (PSRAM)		Secure
0x8000_0000	0xFFFF_FFFF	2048M	RSVD		-

NOTE

- *The function of TrustZone is only applicable to KM4, so the secure address spaces can only be accessed from the secure world of KM4.*
- *The security attribution of address space is determined by the bit[28] of this address.*

All information provided in this document is subject to legal disclaimers.
© REALTEK 2024. All rights reserved.

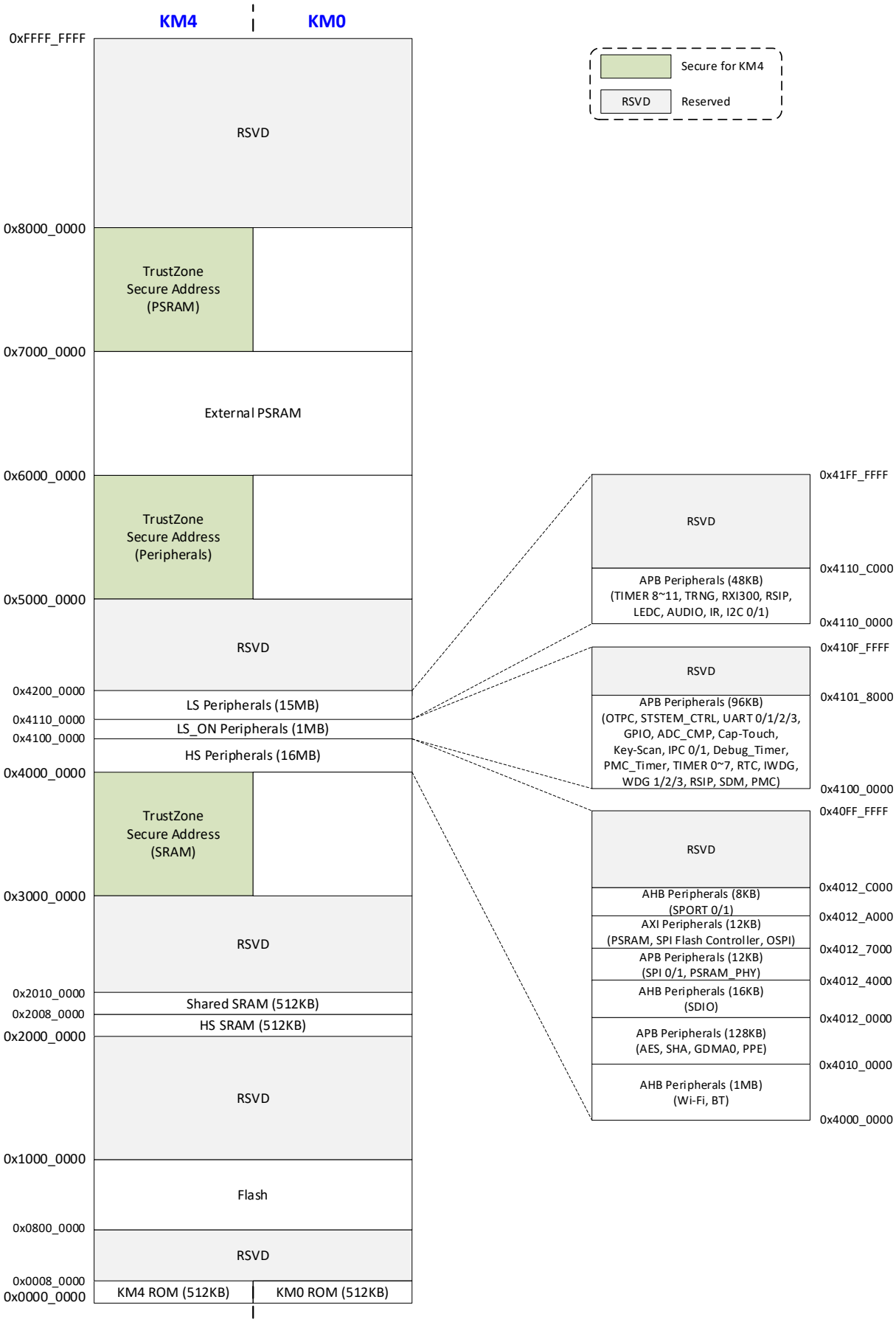


Figure 2-1 Memory map

All information provided in this document is subject to legal disclaimers.
© REALTEK 2024. All rights reserved.

2.2 Memory Map and Register Boundary Address

Table 2-2 lists the memory map and boundary addresses of registers available in the RTL8721Dx device.

Table 2-2 Register boundary addresses

Port ID	Port name	Security	Base address	End address	Size (bytes)
S0	SPIC_AUTO_MODE	Non-secure	0x0800_0000	0x0FFF_FFFF	128M
S1	HS SRAM0	MPC	0x2000_0000	0x2007_FFFF	512K
S2	HS_SHARE_SRAM	MPC	0x2008_0000	0x200F_FFFF	512K
S3	WIFI_REG	PPC	0x4000_0000	0x4007_FFFF	512K
	BT_REG	PPC	0x4008_0000	0x400F_FFFF	512K
	AES_REG	PPC	0x4010_0000	0x4010_7FFF	32K
	SHA_REG	PPC	0x4010_8000	0x4010_FFFF	32K
	GDMA0_REG	PPC	0x4011_0000	0x4011_7FFF	32K
	PPE_REG	PPC	0x4011_8000	0x4011_FFFF	32K
	SDIO_REG	PPC	0x4012_0000	0x4012_3FFF	16K
	SPI0_REG	PPC	0x4012_4000	0x4012_4FFF	4K
	SPI1_REG	PPC	0x4012_5000	0x4012_5FFF	4K
	PSRAM_PHY_REG	PPC	0x4012_6000	0x4012_6FFF	4K
	PSRAM_REG	PPC	0x4012_7000	0x4012_7FFF	4K
	SPI_FLASH_CTRL	PPC	0x4012_8000	0x4012_8FFF	4K
	QSPI_REG	PPC	0x4012_9000	0x4012_9FFF	4K
	SPORT0_REG	PPC	0x4012_A000	0x4012_AFFF	4K
	SPORT1_REG	PPC	0x4012_B000	0x4012_BFFF	4K
	USB_REG	PPC	0x4012_C000	0x4017_FFFF	256K
S4	OTPC_REG	PPC	0x4100_0000	0x4100_7FFF	32K
	SYSTEM_CTRL	PPC	0x4100_8000	0x4100_BFFF	16K
	UART0_REG	PPC	0x4100_C000	0x4100_CFFF	4K
	UART1_REG	PPC	0x4100_D000	0x4100_DFFF	4K
	UART2_REG	PPC	0x4100_E000	0x4100_EFFF	4K
	LOGUART_REG	PPC	0x4100_F000	0x4100_FFFF	4K
	GPIO_REG	PPC	0x4101_0000	0x4101_0FFF	4K
	ADC_REG	PPC	0x4101_1000	0x4101_17FF	2K
	CMP_REG	PPC	0x4101_1800	0x4101_1FFF	2K
	CTC_REG	PPC	0x4101_2000	0x4101_2FFF	4K
	KSCAN_REG	PPC	0x4101_3000	0x4101_3FFF	4K
	IPC0_REG	PPC	0x4101_4000	0x4101_47FF	2K
	IPC1_REG	PPC	0x4101_4800	0x4101_4FFF	2K
	DEBUGTIMER_REG	PPC	0x4101_5000	0x4101_5FFF	4K
	PMC_TIMER_REG	PPC	0x4101_6000	0x4101_6FFF	4K
	TIMER0_REG	PPC	0x4101_7000	0x4101_71FF	512
	TIMER1_REG	PPC	0x4101_7200	0x4101_73FF	512
	TIMER2_REG	PPC	0x4101_7400	0x4101_75FF	512
	TIMER3_REG	PPC	0x4101_7600	0x4101_77FF	512
	TIMER4_REG	PPC	0x4101_7800	0x4101_79FF	512
	TIMER5_REG	PPC	0x4101_7A00	0x4101_7BFF	512
	TIMER6_REG	PPC	0x4101_7C00	0x4101_7DFF	512
	TIMER7_REG	PPC	0x4101_7E00	0x4101_7FFF	512
S5	TIMER8_REG	PPC	0x4110_0000	0x4110_01FF	512
	TIMER9_REG	PPC	0x4110_0200	0x4110_03FF	512
	TIMER10_REG	PPC	0x4110_0400	0x4110_05FF	512
	TIMER11_REG	PPC	0x4110_0600	0x4110_07FF	512
	TRNG_REG	PPC	0x4110_1000	0x4110_1FFF	4K
	RXI300M4_REG	PPC	0x4110_2000	0x4110_3FFF	8K
	RSIP_REG	PPC	0x4110_4000	0x4110_4FFF	4K
	LEDC_REG	PPC	0x4110_5000	0x4110_5FFF	4K
	AUDIO_REG	PPC	0x4110_6000	0x4110_6FFF	4K
	IR_REG	PPC	0x4110_7000	0x4110_7FFF	4K

All information provided in this document is subject to legal disclaimers.
© REALTEK 2024. All rights reserved.

	I2C0_REG	PPC	0x4110_8000	0x4110_9FFF	8K
	I2C1_REG	PPC	0x4110_A000	0x4110_BFFF	8K
S6	PSRAM	MPC	0x6000_0000	0x6FFF_FFFF	256M

2.3 ROM

Both processors share the same address map for SRAM, Flash, PSRAM, and ROM.

The ROM address of KM4 and KM0 is the same from 0x0000_0000 to 0x0007_FFFF. However, KM4 and KM0 have physically separated ITCM ROMs, and each CPU can only access its own ROM.

2.4 On-chip SRAM

The on-chip SRAM starts from 0x2000_0000 and consists of two blocks:

- A general purposed 512KB of contiguous SRAM for system heap and application, is connected to S1. The clock are configurable but cannot exceed 345MHz.
- A dedicated 160KB of connectivity SRAM, 40KB shared with Wi-Fi and 120KB shared with BT, is connected to S2. If user's application applies Wi-Fi function or BT function, the corresponding SRAM would be occupied and cannot be used as general SRAM anymore.

All the SRAMs can be accessed as bytes (8 bits), half-words (16 bits) or full words (32 bits) by processors, DMA engine and other AXI masters.

The entire SRAM can be disabled or enabled in the Power Management Unit (PMU) to save power, and can also enter retention mode for quickly resuming from sleep mode when the system enters sleep mode.

2.5 SPI Flash Memory

The Flash memory consists of a SPI Flash controller and a Flash memory array module. The SPI Flash controller acts as an interface between the system bus and the Flash memory device. It implements the erase and program Flash memory operations, and the read/write protection mechanisms, and accelerates code execution with a system of instruction prefetch and cache lines.

The SPI Flash controller of RTL8721Dx supports SPI NOR Flash with Single/Dual/Quad I/O pins. The I/O voltage is 3.3V. It can run up to 100MHz Single Data Rate (SDR) speed.

2.6 PSRAM

The PSRAM controller of RTL8721Dx supports high-speed hyperbus PSRAM with Double Data Rate (DDR) and 1.8V I/O voltage.

- Clock rate: up to 200MHz
- 8-bit I/O
- Support half sleep-mode and deep power-down mode

All information provided in this document is subject to legal disclaimers.
© REALTEK 2024. All rights reserved.

3 Memory Protection Unit (MPU)

The hardware performs memory management every time that memory is accessed by either the OS or applications. Memory management is a way of dynamically allocating regions of memory to applications.

Bothe the KM4 processor and KM0 processor in RTL8721Dx have a memory protection unit (MPU) that provides fine-grained memory system control, enabling applications to implement security privilege levels, separating code, data and stack on a task-by-task basis.

The MPU sets up the protection by defining the memory map as a number of regions. As *Table 3-1* shows, up to eight regions can be defined in the RTL8721Dx. For More information about how to use the MPU, refer to *Arm®v8-M Architecture Reference Manual*.

Table 3-1 MPU entries

CPU	Security	Region number
KM4	Non-Secure MPU	8
	Secure MPU	4
KM0	Non-Secure MPU	4

The MPU can improve the reliability of an embedded system by:

- preventing user applications from corrupting data used by the operating system
- separating data between processing tasks by blocking tasks from accessing others' data
- allowing memory regions to be defined as read-only so that vital data can be protected
- detecting unexpected memory accesses (e.g. stack corruption)

In addition, the MPU can also be used to define memory access characteristics such as caching and buffering behaviors for different regions. The MPU sets up the protection by defining the memory map as a number of regions, and it needs to be programmed and enabled before use. All MPU regions are aligned to a multiple of 32 bytes.

For KM4, when the MPU is disabled, accesses will use memory attributes from the default system address map.

All information provided in this document is subject to legal disclaimers.
© REALTEK 2024. All rights reserved.

4 Cache

Table 4-1 lists the basic information about cache of RTL8721Dx.

Table 4-1 Cache attribution

CPU	Type	Size (bytes)	Way
KM4	I-Cache	16K	4
	D-Cache	16K	4
KM0	I-Cache	16K	4
	D-Cache	16K	4

4.1 KM0 Cache

The cache operation of KM0 is similar to that of KM4, but the cache registers of KM0 are not banked between Security States. Refer to *Arm®v8-M Architecture Reference Manual* for cache memories characteristics.

4.2 KM4 Cache

The KM4 processor can cache data and instructions that meet certain memory attribute criteria.

L1 Instruction Cache

- is virtually indexed and physically tagged.
- is 4-way set associative.
- is configurable to 16KB.
- uses a cache line length of 32 bytes.
- uses a pseudo-random replacement policy.
- Dynamic program flow prediction.

L1 Data Cache

- is physically indexed and physically tagged.
- is 4-way set associative.
- is configurable to 16KB.
- uses a cache line length of 32 bytes.
- uses a pseudo-random replacement policy.

All information provided in this document is subject to legal disclaimers.
© REALTEK 2024. All rights reserved.

5 Interrupt Controller

5.1 Nested Vector Interrupt Controller (NVIC)

The Nested Vectored Interrupt Controller (NVIC) is closely integrated with the core to achieve low-latency interrupt processing. All the interrupts including the core exceptions are managed by the NVIC.

The NVIC contains the following features:

- The NVIC is an integral part of each CPU.
- Tightly-coupled interrupt controller provides low interrupt latency.
- The NVIC controls system exceptions and peripheral interrupts.
- The NVIC of the KM4 supports:
  - 80 vectored interrupts
  - 8 programmable interrupt priority levels with hardware priority level masking
  - Vector table offset register (VTOR)
  - Secure NVIC and non-secure NVIC
- The NVIC of the KM0 supports:
  - 36 vectored interrupts
  - 4 programmable interrupt priority levels with hardware priority level masking
  - Vector table offset register (VTOR)
- Support for NMI from any interrupts

The NVIC and the processor core are closely coupled, which enables low latency interrupt processing and efficient processing of late arriving interrupts.

For more information about exceptions and NVIC programming, refer to *ARM® Cortex®-M33 Processor Technical Reference Manual*.

5.2 Interrupt List

Table 5-1 lists the interrupt sources for each peripheral function. Each peripheral device may have one or more interrupt lines to the Vectored Interrupt Controller. Each line may represent more than one interrupt source. The interrupt number does not imply any interrupt priority.

Table 5-1 Interrupt list

Function	KM0 INT#	KM4 INT#	KM0_WAKE_MASK
WIFI_FISR_FESR_IRQ	0	0	0
WIFI_FTSR_MAILBOX	1	1	1
WL_DMA	2	2	
WL_PROTOCOL	3	3	
BT_SCB	4	4	
SYS_ILLEGAL_WRITE		5	
KM4_WAKE_IRQ	5		2
BT_WAKE_HOST	6	6	3
RXI300	7	7	
IPC_KM0	8		
IPC_KM4		8	4
IWDG0	9	9	5
Timer0	10	10	
Timer1	11	11	
Timer2	12	12	
Timer3	13	13	
Timer4	14	14	6
Timer5	15	15	7
Timer6	16	16	8
Timer7	17	17	9
Timer8	18	18	
Timer9	19	19	
Timer10	20	20	

All information provided in this document is subject to legal disclaimers.
© REALTEK 2024. All rights reserved.

Timer11	21	21	
PMC_Timer0	22	22	10
PMC_Timer1	23	23	11
UART0	24	24	12
UART1	25	25	13
UART2_BT	26	26	14
UART_LOG	27		
UART_LOG		27	
UART_LOG			15
GPIOA	28	28	16
GPIOB	29	29	17
I2C0	30	30	18
I2C1	31	31	19
CAP_TOUCH	32	32	20
GDMA0_Channel0	33	33	
GDMA0_Channel1	34	34	
GDMA0_Channel2	35	35	
GDMA0_Channel3	36	36	
GDMA0_Channel4	37	37	
GDMA0_Channel5	38	38	
GDMA0_Channel6	39	39	
GDMA0_Channel7	40	40	
PPE	41	41	
SPI0	42	42	
SPI1	43	43	
SPORT0	44	44	
SPORT1	45	45	
RTC	46	46	21
ADC	47	47	22
ADC_COMP	48	48	23
BOR	49	49	24
PWR_DOWN	50	50	25
SPI_FLASH	51	51	
KEY_SCAN	52	52	26
RSIP	53	53	
AES	54	54	
SHA	55	55	
PSRAMC	56	56	
TRNG	57	57	
AES_S	58	58	
SHA_S	59	59	
AON_TIM	60	60	27
AON_WAKEPIN	61	61	28
LEDC	62	62	
IR	63	63	
SDIO	64	64	29
KM0_WDG	65		
KM4_NS_WDG		65	
KM4_S_WDG		66	
QSPI_INT		67	
USB_INT	66	68	
OCP_INT		69	

All information provided in this document is subject to legal disclaimers.
© REALTEK 2024. All rights reserved.

6 Hardware Crypto Engine (IPsec)

6.1 Application Scenario

The Hardware Crypto Engine (also called IPsec) accelerates applications that need cryptographic functions, such as authentication, encryption and decryption. Hardware crypto engine executing these functions cannot only reduce software overhead but also save CPU and memory resources, and the processing of it is more secure and faster than software.

6.2 Features

Realtek IPsec provides the basic cryptographic features as listed below:

- Authentication algorithms
  - General cryptographic hash function
    - MD5 (weak, not recommended)
    - SHA1 (weak, not recommended)
    - SHA2-224
    - SHA2-256
    - SHA2-384
    - SHA2-512
  - HMAC (Hash-based message authentication code)
    - HMAC_MD5 (weak, not recommended)
    - HMAC_SHA1 (weak, not recommended)
    - HMAC_SHA2-224
    - HMAC_SHA2-256
    - HMAC_SHA2-384
    - HMAC_SHA2-512
- Cipher (Encryption/Decryption) algorithms
  - AES-128/192/256
    - ECB (Electronic Codebook) mode (weak, not recommended)
    - CBC (Cipher Block Chaining) mode
    - OFB (Output Feedback) mode
    - CFB (Cipher Feedback) mode
    - CTR (Counter) mode (weak, not recommended)
    - GCM (Galois/Counter Mode) mode

NOTE

The IPsec supports Once Process Way or Sequential Hash Mechanism in authentication algorithms.

- *If the message length doesn't exceed 245745 bytes (15 * (214 -1)), we could use Once Process Way to verify the hash function.*
- *If the message length exceeds 245745 bytes (15 * (214 -1)) or we want to divide this message into many particular size blocks to process them, we could use Sequential Hash Mechanism to verify the hash function.*

6.3 Architecture

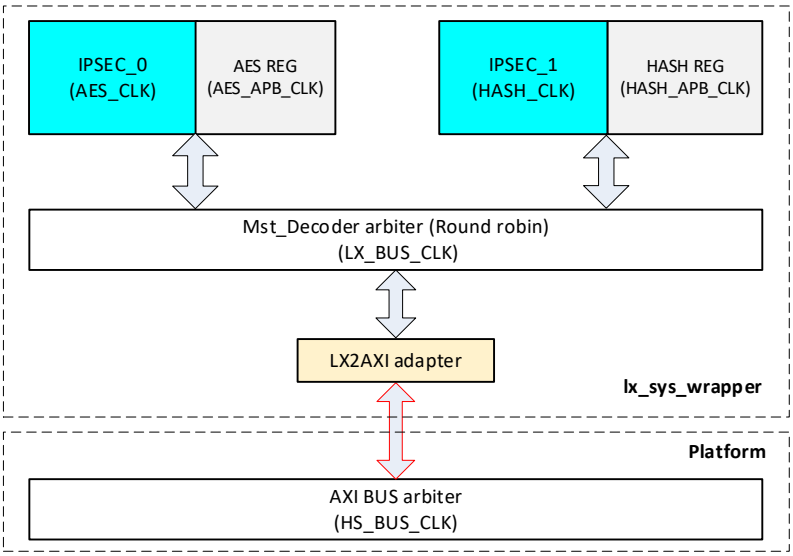
6.3.1 IPsec Block Diagram

The architecture of IPsec on system bus is shown in *Figure 6-1*. IPSEC_0 (AES), IPSEC_1 (HASH), and SDEMMC are masters on lx_bus. There is an arbiter of these masters in lx_sys_wrapper and an adapter of lexra bus and AXI bus, which is responsible for transferring between bus interfaces. The control register of each device is acted as slave on APB bus independently.

There are 4 main parts inside the IPsec architecture.

- Control registers
- Source/Destination FIFO descriptors
- DMA engine
- Crypto engine

All information provided in this document is subject to legal disclaimers.
© REALTEK 2024. All rights reserved.



Note:

- LX_BUS_CLK = HS_BUS_CLK = AES_CLK = HASH_CLK= AES_APB_CLK = HASH_APB_CLK
- All clocks above are divided from CPU PLL, and the division factor varies from 1 to 8.

Figure 6-1 Architecture of IPsec on system bus

The programmer can configure or get more detail information of IPsec via directly setting or reading Control registers. For example, setting basic cryptographic function parameters, setting/reading Crypto interrupt status, setting/reading Crypto Error interrupt, setting DMA/Crypto engine. The more setting details are illustrated in the section *Registers*. The crypto engine of AES and HASH are illustrated in *Figure 6-2* and *Figure 6-3*.

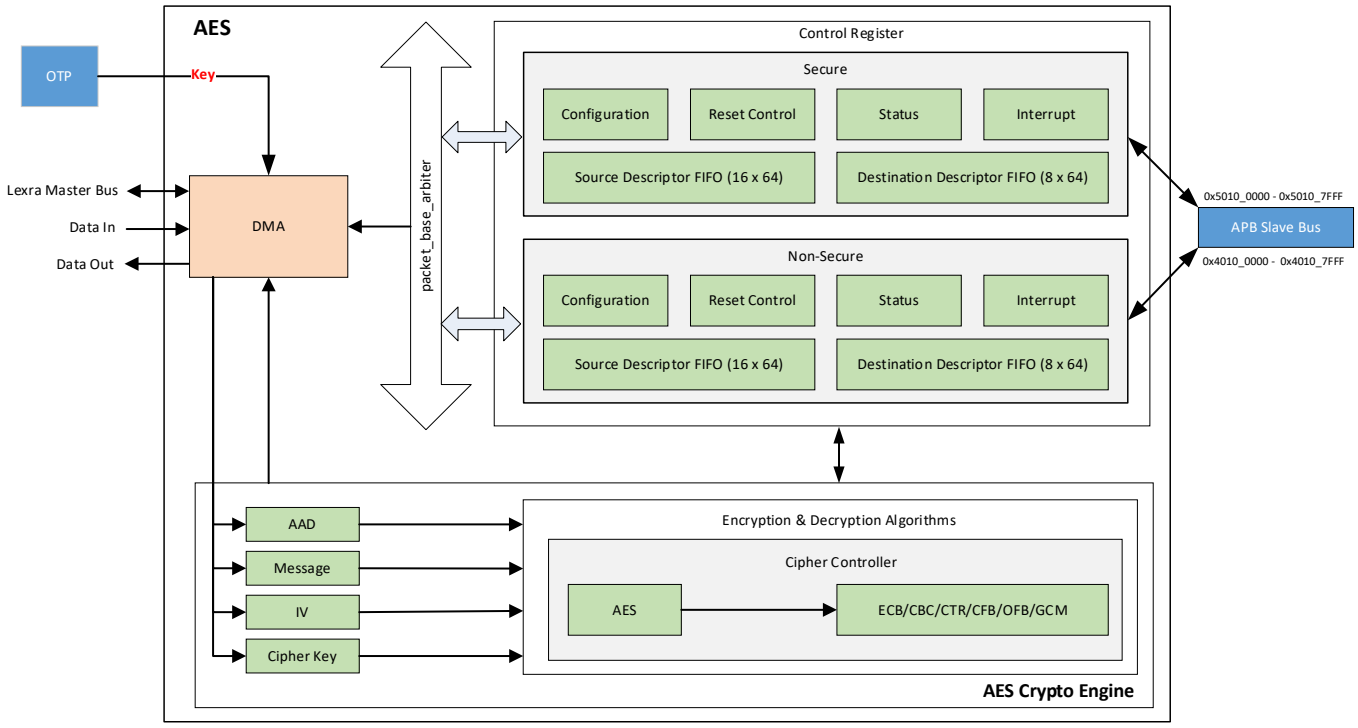


Figure 6-2 Crypto engine of AES block diagram

All information provided in this document is subject to legal disclaimers.
© REALTEK 2024. All rights reserved.

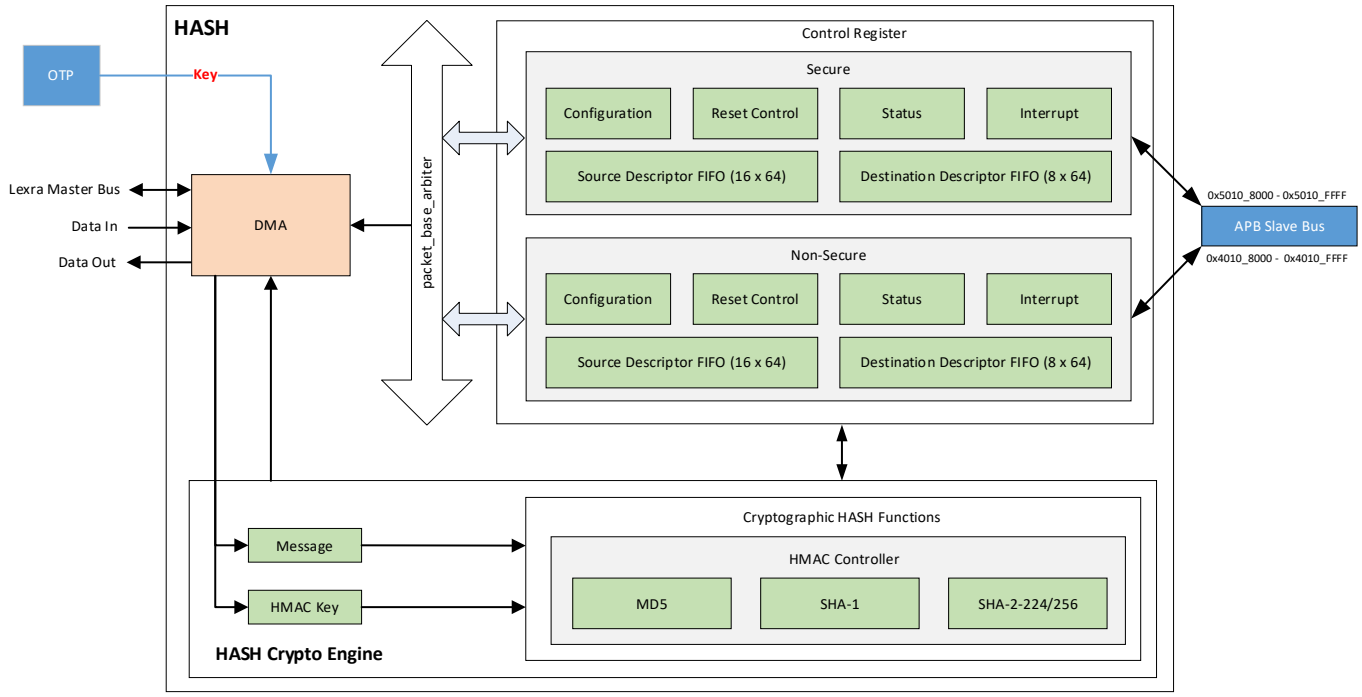


Figure 6-3 Crypto engine of HASH block diagram

IPsec implements many kinds of cryptographic algorithm in Crypto engine. For programmer, the way of setting basic cryptographic function parameters is writing data into Source/Destination descriptor first/second word register. The difference between Source descriptor and Destination descriptor is the data buffer that the Crypto engine wants to read/write.

Source descriptor first/second word register is used to set cryptographic function input parameters (HMAC Key PAD/KEY/IV/Plaintext buffer); Destination descriptor first/second word register is used to set cryptographic function output parameters (Digest/Cipher result buffer). Because there are many parameters that need to set, the programmer can disassemble a Source/Destination packet command into some hardware FIFO in Source/Destination descriptors.

6.3.2 DMA Block Diagram

DMA engine gets buffer address from the Source/Destination Descriptor FIFO node, then it moves data into Crypto engine before Crypto engine starts to calculate, and when Crypto engine finishes the calculation, it will help move the result data to the result buffer.

The DMA module acts as a data transfer controller between source and destination. Programmer can access to DMA registers to set DMA configuration information, including source address, destination address, data length, read/write mode, etc. The DMA block diagrams of AES and HASH are shown in *Figure 6-4* and *Figure 6-5* respectively.

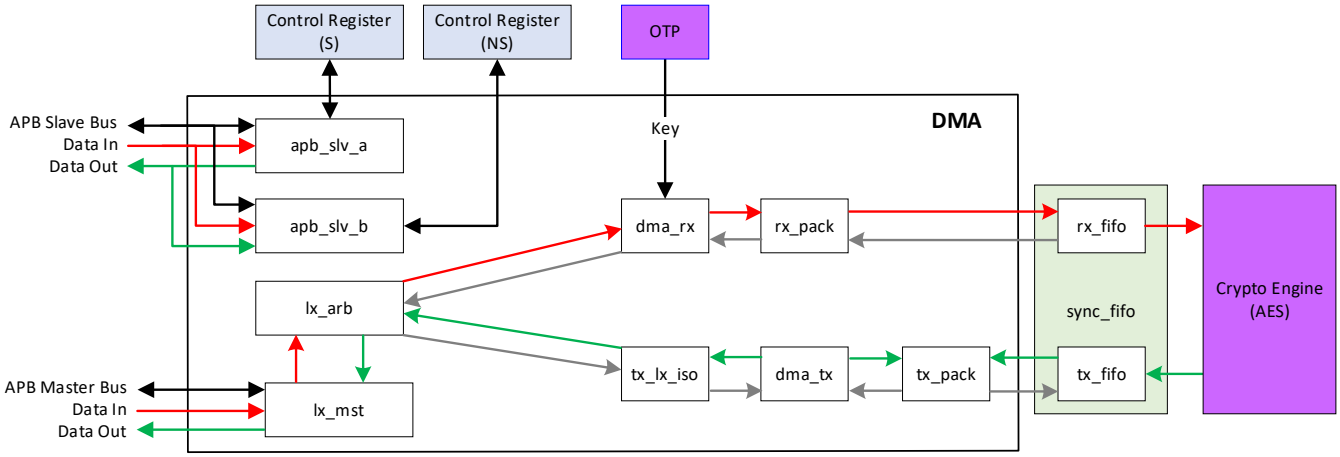


Figure 6-4 DMA block diagram of AES

All information provided in this document is subject to legal disclaimers.
© REALTEK 2024. All rights reserved.

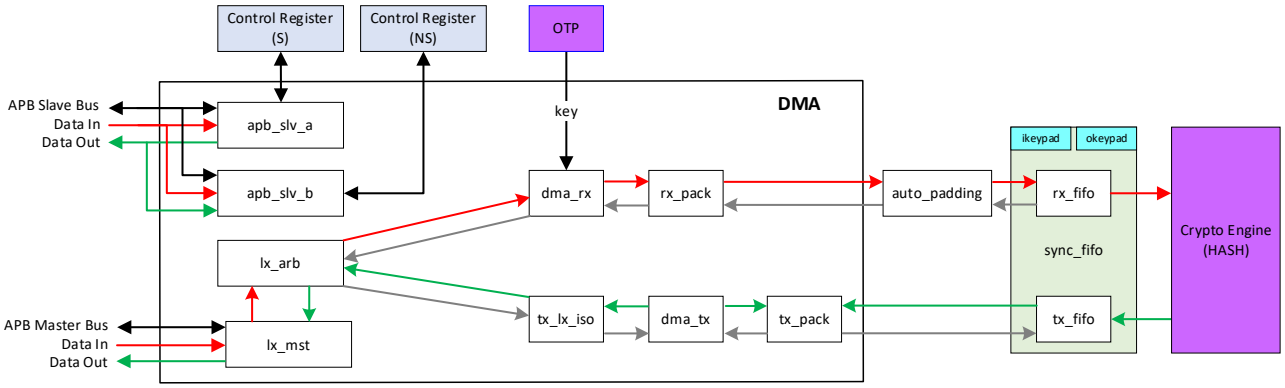


Figure 6-5 DMA block diagram of HASH

6.4 Registers

The system programmer may access any of the IPsec relative control and status registers via CPU. These registers control IPsec operations including setting/debugging descriptors, setting crypto engine and setting DMA mode.

- Physical base address of AES registers: 0x 4010_0000
- Physical base address of HASH registers: 0x 4010_8000

NOTE

Do not use the bits named as RSVD; otherwise, it may lead to unpredictable results.

6.4.1 AES Registers

Name	Address offset	Access	Description
REG_SDSR	000h	R/W	Source Descriptor Status Register
REG_SDFWR	004h	R/W	Source Descriptor First Word Register
REG_SDSWR	008h	R/W	Source Descriptor Second Word Register
REG_IPSCSR_RSTEACONFISR	010h	R/W	Reset Engine and Configure Interrupt Service Routine Register
REG_IPSCSR_INTM	014h	R/W	Interrupt Mask Register
REG_IPSCSR_DBG	018h	R/W	Debug Register
REG_IPSCSR_ERR_INT	01Ch	R/W	Error Interrupt Register
REG_IPSCSR_SADDLR	020h	R	Sum of Additional Authentication Data Length Register
REG_IPSCSR_SENLR	024h	R	Sum of Encryption Data Length Register
REG_IPSCSR_SAPLR	028h	R	Sum of Authentication Padding Data Length Register
REG_IPSCSR_SEPLR	02Ch	R	Sum of Encryption Padding Data Length Register
REG_IPSCSR_SWAPABURSTR	030h	R/W	Swap and Burst Register
REG_IPSEKCR	034h	R/W	Key Control Register
REG_ENG_DBG	040h	R	Crypto Engine Debug Register
REG_DMA_LX_DBG	044h	R	Crypto Engine DMA LX Bus Debug Register
REG_DMA_RX_DBG	048h	R	Crypto Engine DMA Rx Debug Register
REG_DMA_TX_DBG	04Ch	R	Crypto Engine DMA Tx Debug Register
REG_DDSR	1000h	R/W	Destination Descriptor Status Register
REG_DDFWR	1004h	R/W	Destination Descriptor First Word Register
REG_DDSWR	1008h	R/W	Destination Descriptor Second Word Register
REG_DES_PKTCONF	100Ch	R/W	Packet Configure Register
REG_DBGSDR	1010h	R	Debug Source Descriptor Data Register
REG_DBGDDR	1014h	R	Debug Destination Descriptor Data Register
REG_DUMMY	1018h	R/W	Dummy Register

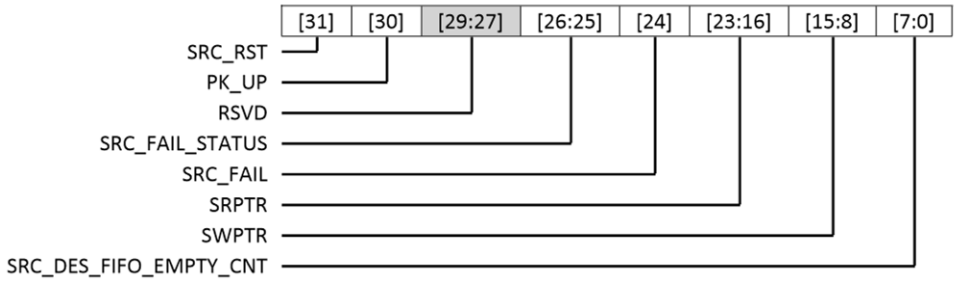
6.4.1.1 REG_SDSR

- **Name:** Source Descriptor Status Register
- **Size:** 32 bits
- **Address offset:** 000h

All information provided in this document is subject to legal disclaimers.
© REALTEK 2024. All rights reserved.

- **Read/write access:** R/W

This register is used to check Source Descriptor FIFO status.

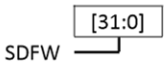


Bit	Symbol	Access	Reset	Description
31	SRC_RST	R/W	0	Source descriptor reset. (only for pk_up = 1'b1)
30	PK_UP	R/W	1	Packet base update wptr to engine. If pk_up =1, the total number of source descriptor in one packet can't be over 16, and the total number of destination descriptor in one packet can't be over 8.
29:27	RSVD	-	0	Reserved
26:25	SRC_FAIL_STATUS	R	0	Source descriptor failed status. 2'd1: Users write the first word twice consecutively 2'd2: Users write the second word directly without writing first word in the beginning 2'd3: Overflow (Detect users try to write source descriptor, but there isn't available FIFO node to use).
24	SRC_FAIL	R/W	0	Source descriptor failed interrupt. Write 1 to clear this bit.
23:16	SRPTR	R	0	Source descriptor FIFO read pointer. When engine reads a descriptor and finishes it, SRPTR = SRPTR+2.
15:8	SWPTR	R	0	Source descriptor FIFO write pointer. When write descriptor to SDSW successfully, SWPTR = SWPTR+2.
7:0	SRC_DES_FIFO_EMPTY_CNT	R	0	Source descriptor FIFO empty counter. Use this field to check how many available FIFO nodes in source descriptor FIFO could use.

6.4.1.2 REG_SDFWR

- **Name:** Source Descriptor First Word Register
- **Size:** 32 bits
- **Address offset:** 004h
- **Read/write access:** R/W

This register is used to set descriptor header. The programmer can follow Source Crypto Descriptor Type1/2 offset 0 to set header information data, and write this data to this register.



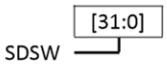
Bit	Symbol	Access	Reset	Description
31:0	SDFW	R/W	0	Source Descriptor First Word

6.4.1.3 REG_SDSWR

- **Name:** Source Descriptor Second Word Register
- **Size:** 32 bits
- **Address offset:** 008h
- **Read/write access:** R/W

All information provided in this document is subject to legal disclaimers.
© REALTEK 2024. All rights reserved.

This register is used to set descriptor data buffer address. The programmer can know this register is a hardware pointer which records the data buffer address from Source Crypto Descriptor Type1/2 offset 4. If the programmer sets a descriptor, need to write the header information data in first word register and write data buffer address in second word register.

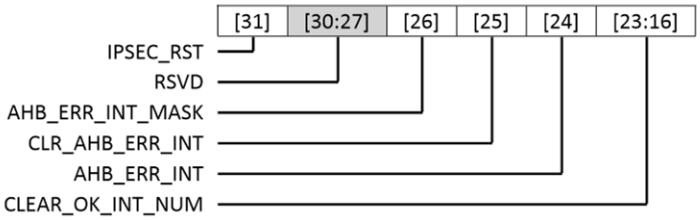


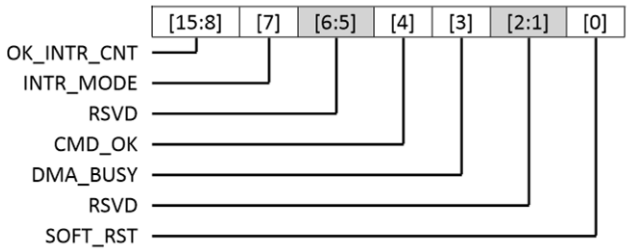
Bit	Symbol	Access	Reset	Description
31:0	SDSW	R/W	0	Source Descriptor Second Word

6.4.1.4 REG_IPSCSR_RSTEACONFISR

- **Name:** Reset Engine and Configure Interrupt Service Routine Register
- **Size:** 32 bits
- **Address offset:** 010h
- **Read/write access:** R/W

This register allows the programmer can reset crypto engine and DMA engine, check and configure crypto engine interrupt status.





Bit	Symbol	Access	Reset	Description
31	IPSEC_RST	R/W	0	IPsec engine reset. Write '1' to reset the crypto engine and DMA engine (used to clear fatal error).
30:27	RSVD	-	0	Reserved
26	AHB_ERR_INT_MASK	R/W	1	
25	CLR_AHB_ERR_INT	R/W	0	
24	AHB_ERR_INT	R	0	
23:16	CLEAR_OK_INT_NUM	R/W	1	Clear ok interrupt number. In interrupt counter mode, if the programmer wants to clear ok_intr_cnt, need to write ok_intr_cnt to this field first, then write '1' to cmd_ok.
15:8	OK_INTR_CNT	R	0	Ok interrupt counter. Reading this field can know how many interrupts are coming to notify that crypto engine has calculated cryptographic feature results. Use this field in interrupt counter mode.
7	INTR_MODE	R/W	0	Select ok interrupt mode: 1'd0: Interrupt mode is general mode. 1'd1: Interrupt mode is counter mode. · General mode means no matter how many interrupts are coming to notify that crypto engine has calculated cryptographic feature results. There is always one interrupt signal shows in cmd_ok field, and nothing in ok_intr_cnt. So if the programmer want to clear this signal, just write '1' to cmd_ok. · Counter mode means the programmer can know how many interrupts are coming to notify that crypto engine has calculated cryptographic feature results from

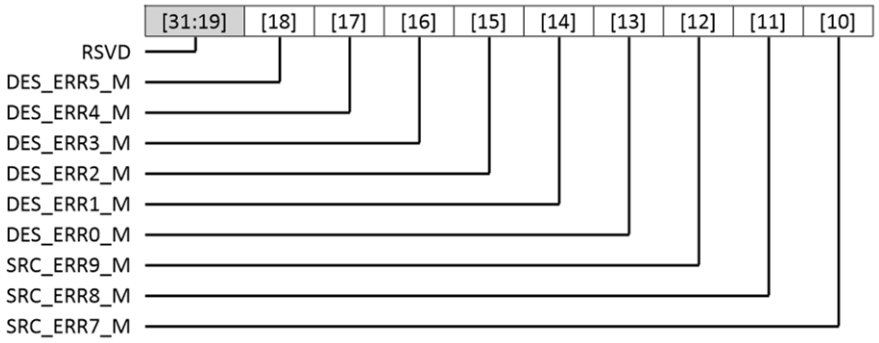
All information provided in this document is subject to legal disclaimers.
© REALTEK 2024. All rights reserved.

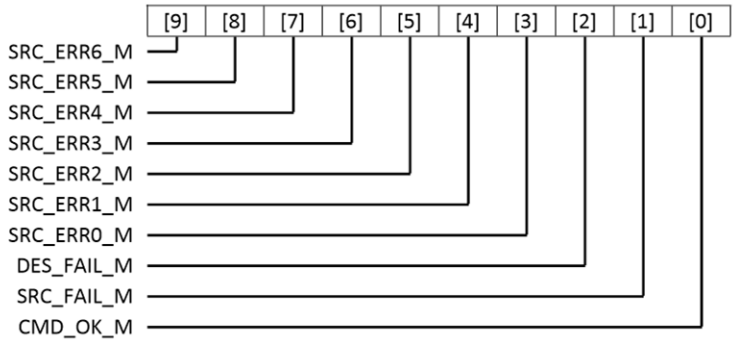
				ok_intr_cnt. The programmer also can know there are at least one interrupt from reading cmd_ok. So if the programmer want to clear ok_intr_cnt signals, need to write ok_intr_cnt into clear_ok_int_num, then write '1' to cmd_ok.
6:5	RSVD	-	0	Reserved
4	CMD_OK	R/W	0	Command ok Interrupt. Reading this field to detect whether crypto engine has calculated a cryptographic feature result. Even if interrupt mode is counter mode, still can use this field to detect whether an interrupt is coming. Write '1' to clear this interrupt signal.
3	DMA_BUSY	R	0	Detect whether IPsec DMA is busy: 1'd0: DMA is not busy 1'd1: DMA is busy Used for debug to avoid data coherence issue (when user gets crypto finish calculated interrupt, user can make sure DMA engine has already move data to result buffer.)
2:1	RSVD	-	0	Reserved
0	SOFT_RST	W	0	Software reset. Write '1' to reset the crypto engine.

6.4.1.5 REG_IPSCSR_INTM

- **Name:** Interrupt Mask Register
- **Size:** 32 bits
- **Address offset:** 014h
- **Read/write access:** R/W

This register is used to enable or disable Command ok interrupt mask, Source descriptor error interrupt mask and Destination descriptor error interrupt mask.





Bit	Symbol	Access	Reset	Description
31:19	RSVD	-	0	Reserved
18	DES_ERR5_M	R/W	1	Destination Descriptor Error 5 Interrupt Mask 1'd0: Disable mask 1'd1: Enable mask

All information provided in this document is subject to legal disclaimers.
© REALTEK 2024. All rights reserved.

17	DES_ERR4_M	R/W	1	Destination Descriptor Error 4 Interrupt Mask 1'd0: Disable mask 1'd1: Enable mask
16	DES_ERR3_M	R/W	1	Destination Descriptor Error 3 Interrupt Mask 1'd0: Disable mask 1'd1: Enable mask
15	DES_ERR2_M	R/W	1	Destination Descriptor Error 2 Interrupt Mask 1'd0: Disable mask 1'd1: Enable mask
14	DES_ERR1_M	R/W	1	Destination Descriptor Error 1 Interrupt Mask 1'd0: Disable mask 1'd1: Enable mask
13	DES_ERR0_M	R/W	1	Destination Descriptor Error 0 Interrupt Mask 1'd0: Disable mask 1'd1: Enable mask
12	SRC_ERR9_M	R/W	1	Source Descriptor Error 9 Interrupt Mask 1'd0: Disable mask 1'd1: Enable mask
11	SRC_ERR8_M	R/W	1	Source Descriptor Error 8 Interrupt Mask 1'd0: Disable mask 1'd1: Enable mask
10	SRC_ERR7_M	R/W	1	Source Descriptor Error 7 Interrupt Mask 1'd0: Disable mask 1'd1: Enable mask
9	SRC_ERR6_M	R/W	1	Source Descriptor Error 6 Interrupt Mask 1'd0: Disable mask 1'd1: Enable mask
8	SRC_ERR5_M	R/W	1	Source Descriptor Error 5 Interrupt Mask 1'd0: Disable mask 1'd1: Enable mask
7	SRC_ERR4_M	R/W	1	Source Descriptor Error 4 Interrupt Mask 1'd0: Disable mask 1'd1: Enable mask
6	SRC_ERR3_M	R/W	1	Source Descriptor Error 3 Interrupt Mask 1'd0: Disable mask 1'd1: Enable mask
5	SRC_ERR2_M	R/W	1	Source Descriptor Error 2 Interrupt Mask 1'd0: Disable mask 1'd1: Enable mask
4	SRC_ERR1_M	R/W	1	Source Descriptor Error 1 Interrupt Mask 1'd0: Disable mask 1'd1: Enable mask
3	SRC_ERR0_M	R/W	1	Source Descriptor Error 0 Interrupt Mask 1'd0: Disable mask 1'd1: Enable mask
2	DES_FAIL_M	R/W	1	Source Descriptor fail interrupt mask. 1'd0: Disable mask 1'd1: Enable mask
1	SRC_FAIL_M	R/W	1	Destination Descriptor fail interrupt mask. 1'd0: Disable mask 1'd1: Enable mask
0	CMD_OK_M	R/W	1	Command ok Interrupt Mask. 1'd0: Disable mask 1'd1: Enable mask If the programmer wants to disable cmd_ok interrupt signal, write '1' to enable interrupt mask.

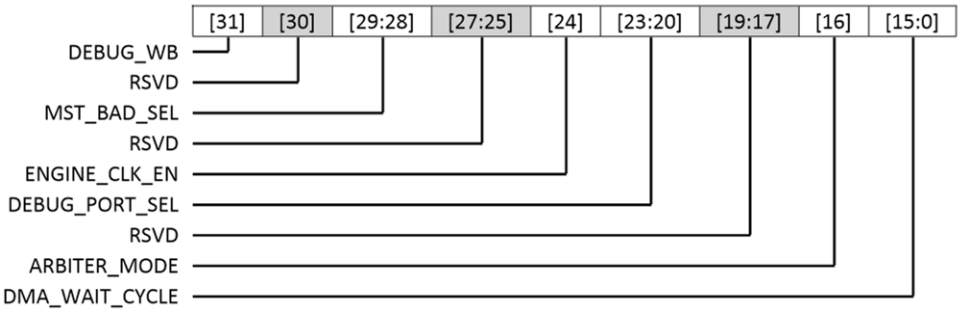
6.4.1.6 REG_IPSCSR_DBG

- **Name:** Debug Register
- **Size:** 32 bits

All information provided in this document is subject to legal disclaimers.
© REALTEK 2024. All rights reserved.

- **Address offset:** 018h
- **Read/write access:** R/W

This register is used to configure or debug IPsec DMA engine.



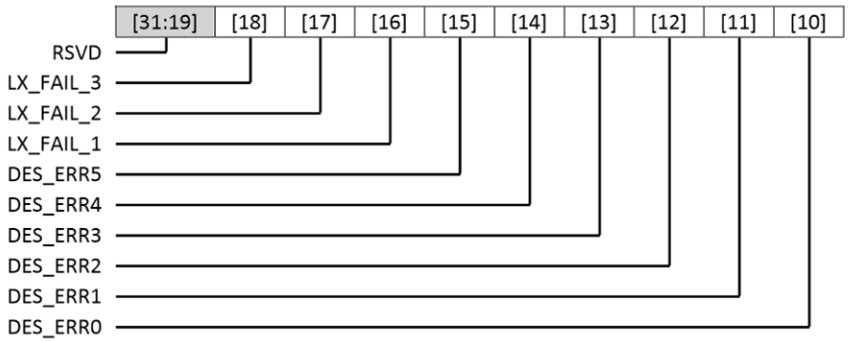
Bit	Symbol	Access	Reset	Description
31	DEBUG_WB	R/W	0	Debug: write back mode. 1'd0: Disable DMA write back mode. 1'd1: Enable DMA write back mode. Enable this field, crypto DMA will move data from source address to destination address. During this period, the data won't be processed in any calculation (encryption, decryption, hash), then the programmer can check whether the source data is the same as destination data.
30	RSVD	-	0	Reserved
29:28	MST_BAD_SEL	R/W	0	(For debug port) select Lexra master data bus (option) 2'd0: mst_ipsec_bad[07:00] 2'd1: mst_ipsec_bad[15:08] 2'd2: mst_ipsec_bad[23:16] 2'd3: mst_ipsec_bad[31:24]
27:25	RSVD	-	0	Reserved
24	ENGINE_CLK_EN	R/W	0	IPsec engine clock enable 1'd0: Disable IPsec engine clock 1'd1: Enable IPsec engine clock
23:20	DEBUG_PORT_SEL	R/W	0	Debug port selection 4'd0: engine_debug 4'd1: dma_lexra_debug 4'd2: dma_rx_debug 4'd3: dma_tx_debug
19:17	RSVD	-	0	Reserved
16	ARBITER_MODE	R/W	1	DMA arbiter mode: 1'd0: round-robin 1'd1: detect FIFO wasted level There are 2 FIFOs in DMA, which are read-FIFO and write-FIFO. Detecting FIFO wasted level means that DMA will process which FIFO first depends on which wasted level is high. Because the wasted level is high means there are many FIFO nodes waiting to be processed in this FIFO, so DMA must process them in hurry.
15:0	DMA_WAIT_CYCLE	R/W	0	Setting DMA wait cycles to assert next DMA request

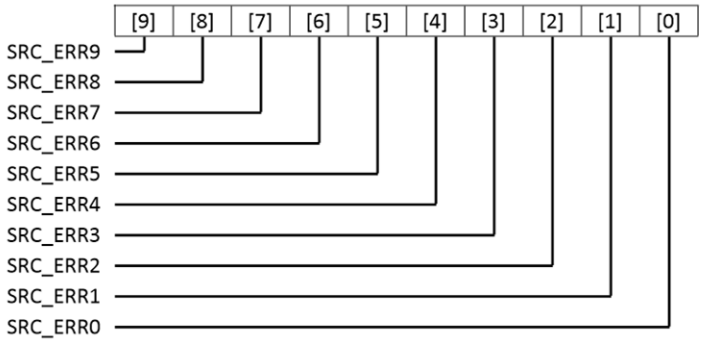
6.4.1.7 REG_IPSCSR_ERR_INT

- **Name:** Error Interrupt Register
- **Size:** 32 bits
- **Address offset:** 01Ch
- **Read/write access:** R/W

This register is used to detect Source/Destination descriptor setting error interrupts.

All information provided in this document is subject to legal disclaimers.
© REALTEK 2024. All rights reserved.





Bit	Symbol	Access	Reset	Description
31:19	RSVD	-	0	Reserved
18	LX_FAIL_3	RW1C	0	When a burst done, burst length != burst cnt. Write “1” to clear this error signal.
17	LX_FAIL_2	RW1C	0	When IPsec is working, rdata from bus != rdata of ipsec or wdata of ipsec != wdata on bus Write “1” to clear this error signal.
16	LX_FAIL_1	RW1C	0	LX bus timing issue Write “1” to clear this error signal.
15	DES_ERR5	RW1C	0	Destination Descriptor Error 5 Interrupt. The error condition is authentication digest length error in destination descriptor. MD5: adl != 16 or SHA1: adl != 20 or SHA2-224: adl != 28 or adl != 32 when set sequential hash and not sequential hash last SHA2-256: adl != 32 or SHA2-384: adl != 48 or adl != 64 when set sequential hash and not sequential hash last SHA2-512: adl != 64 Write '1' to clear this error signal.
14	DES_ERR4	RW1C	0	Destination Descriptor Error 4 Interrupt. The error condition is authentication digest length error in destination descriptor. MD5: adl > 16 or SHA1: adl > 20 or SHA2-224: adl > 28 or adl > 32when set sequential hash and not sequential hash last SHA2-256: adl > 32 or SHA2-384: adl > 48 or adl > 64when set sequential hash and not sequential hash last SHA2-512: adl > 64 Write '1' to clear this error signal.
13	DES_ERR3	RW1C	0	Destination Descriptor Error 3 Interrupt. One of the error condition is that the sum of enl is greater than Encryption Total Length. The other error condition is about destination descriptor length error ((enl(Encryption length)==0) & (adl(authentication data length)==0)). Write '1' to clear this error signal. Sum of enl reference the register REG_IPSCSR_SENLR (offset 0x24).
12	DES_ERR2	RW1C	0	Destination Descriptor Error 2 Interrupt. The error condition is the bit field ENC=1'b0 (active Hash engine to process hash data), but only active cipher engine.

All information provided in this document is subject to legal disclaimers.
© REALTEK 2024. All rights reserved.

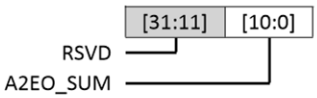
				Write '1' to clear this error signal.
11	DES_ERR1	RW1C	0	Destination Descriptor Error 1 Interrupt. The error condition is the bit field ENC=1'b1 (active cipher engine to process cipher data), but only active hash engine. Write '1' to clear this error signal.
10	DES_ERR0	RW1C	0	Destination Descriptor Error 0 Interrupt. One of the error condition is destination descriptor starting address error (DDSA[1:0]!=2'd0 [At least 4 bytes-aligned]). The other error condition is memory protection event. Write '1' to clear this error signal.
9	SRC_ERR9	RW1C	0	Source Descriptor Error 9 Interrupt. The error condition is that sum of epl is not equal to EPTL (Encryption Padding Total Length). Write '1' to clear this error signal. Sum of epl reference the register REG_IPSCSR_SEPLR (offset 0x2C).
8	SRC_ERR8	RW1C	0	Source Descriptor Error 8 Interrupt. The error condition is that sum of apl is not equal to HPTL (Hash Padding Total Length). Write '1' to clear this error signal. Sum of apl reference the register REG_IPSCSR_SAPLR (offset 0x28).
7	SRC_ERR7	RW1C	0	Source Descriptor Error 7 Interrupt. The error condition is that sum of enl is not equal to ETL (Encryption Total Length). Write '1' to clear this error signal. Sum of enl reference the register REG_IPSCSR_SENLR (offset 0x24).
6	SRC_ERR6	RW1C	0	Source Descriptor Error 6 Interrupt. The error condition is that sum of a2eo is not equal to HTL (Header Total Length). Write '1' to clear this error signal. Sum of a2eo reference the register REG_IPSCSR_SAADLR (offset 0x20).
5	SRC_ERR5	RW1C	0	Source Descriptor Error 5 Interrupt. The error condition is that there is a source descriptor, but the data length ((a2eo/epl) +(enl) + (apl)) is 0. Write '1' to clear this error signal.
4	SRC_ERR4	RW1C	0	Source Descriptor Error 4 Interrupt. The error condition is source data buffer pointer error (SDBP[1:0]!=2'd0 [At least 4 byte aligned]). Write '1' to clear this error signal.
3	SRC_ERR3	RW1C	0	Source Descriptor Error 3 Interrupt. About source descriptor length error. One of the error condition is setting the first descriptor but (CL!=3). The other error condition is that there is a source descriptor but CL, KL, IL, PL, SHIVL are all 0. Write '1' to clear this error signal.
2	SRC_ERR2	RW1C	0	Source Descriptor Error 2 Interrupt. One of the error condition is source descriptor starting address error (SDSA[1:0]!=2'd0 [At least 4 bytes-aligned]). The other error condition is memory protection event. Write '1' to clear this error signal.
1	SRC_ERR1	RW1C	0	Source Descriptor Error 1 Interrupt. The error condition is last segment descriptor is set (LS=1), when descriptor is pointing to the first segment of the packet. Write '1' to clear this error signal.
0	SRC_ERR0	RW1C	0	Source Descriptor Error 0 Interrupt. One of the error condition is that first segment descriptor is not set (FS=0), when descriptor is pointing to the first segment of the packet. The other error condition is that first segment descriptor is set (FS=1), when descriptor is not pointing to the first segment of the packet. Write '1' to clear this error signal.

6.4.1.8 REG_IPSCSR_SADDLR

- **Name:** Sum of Additional Authentication Data Length Register
- **Size:** 32 bits
- **Address offset:** 020h
- **Read/write access:** R

This register is used to get sum of additional authentication data length.

All information provided in this document is subject to legal disclaimers.
© REALTEK 2024. All rights reserved.

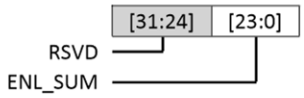


Bit	Symbol	Access	Reset	Description
31:11	RSVD	-	0	Reserved
10:0	A2EO_SUM	R	0	Sum of additional authentication data length.

6.4.1.9 REG_IPSCSR_SENLR

- **Name:** Sum of Encryption Data Length Register
- **Size:** 32 bits
- **Address offset:** 024h
- **Read/write access:** R

This register is used to get sum of encryption data length.

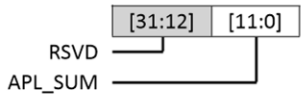


Bit	Symbol	Access	Reset	Description
31:24	RSVD	-	0	Reserved
23:0	ENL_SUM	R	0	Sum of encryption data length.

6.4.1.10 REG_IPSCSR_SAPLR

- **Name:** Sum of Authentication Padding Data Length Register
- **Size:** 32 bits
- **Address offset:** 028h
- **Read/write access:** R

This register is used to get sum of authentication padding data length.

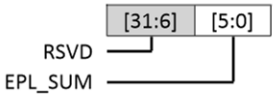


Bit	Symbol	Access	Reset	Description
31:12	RSVD	-	0	Reserved
11:0	APL_SUM	R	0	Sum of authentication padding data length.[Mix mode uses this field]

6.4.1.11 REG_IPSCSR_SEPLR

- **Name:** Sum of Encryption Padding Data Length Register
- **Size:** 32 bits
- **Address offset:** 02Ch
- **Read/write access:** R

This register is used to get sum of encryption padding data length.



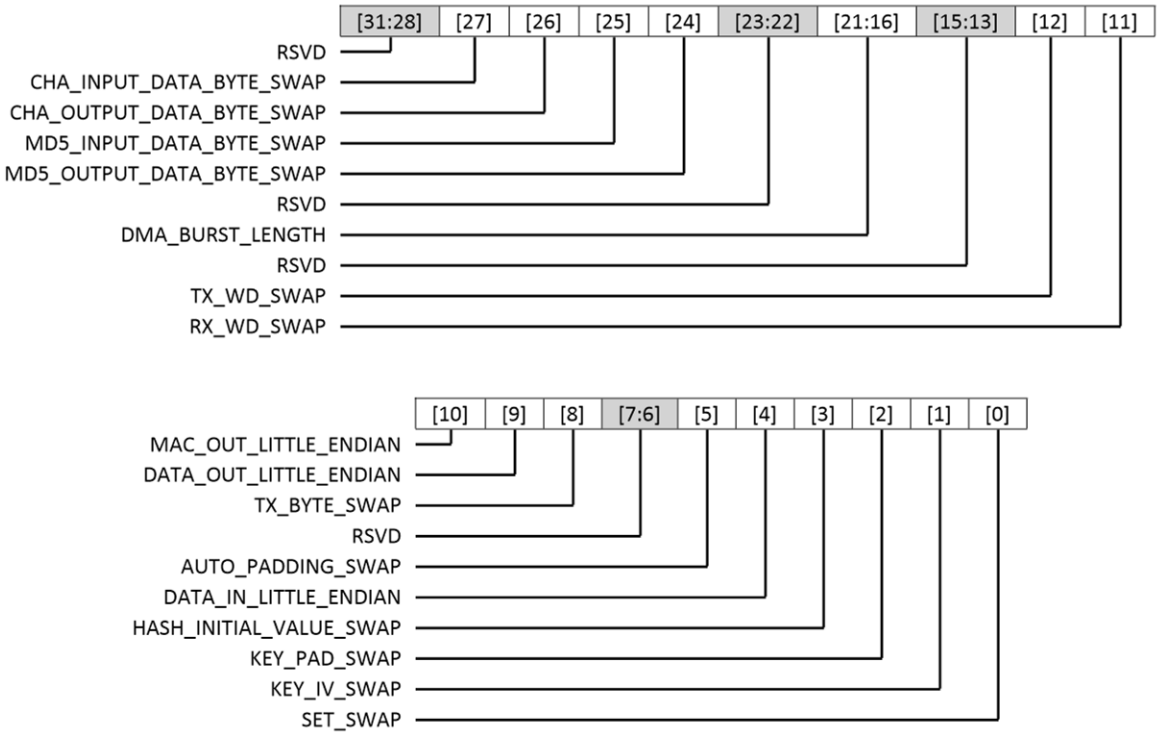
Bit	Symbol	Access	Reset	Description
31:6	RSVD	-	0	Reserved
5:0	EPL_SUM	R	0	Sum of encryption padding data length. [Mix mode uses this field]

All information provided in this document is subject to legal disclaimers.
© REALTEK 2024. All rights reserved.

6.4.1.12 REG_IPSCSR_SWAPABURSTR

- **Name:** Swap and Burst Register
- **Size:** 32 bits
- **Address offset:** 030h
- **Read/write access:** R/W

This register configures swap and endian setting in crypto engine, and DMA burst length.



Bit	Symbol	Access	Reset	Description
31:28	RSVD	-	0	Reserved
27	CHA_INPUT_DATA_BYTE_SWAP	R/W	0	Byte swap for CHA input data 1: Enable 0: Disable
26	CHA_OUTPUT_DATA_BYTE_SWAP	R/W	0	Byte swap for CHA output data 1: Enable 0: Disable
25	MD5_INPUT_DATA_BYTE_SWAP	R/W	0	Byte swap for MD5 input data 1: Enable 0: Disable
24	MD5_OUTPUT_DATA_BYTE_SWAP	R/W	0	Byte swap for MD5 output data 1: Enable 0: Disable
23:22	RSVD	-	0	Reserved
21:16	DMA_BURST_LENGTH	R/W	0x10	Set DMA burst length. The maximum setting length is 16 bursts. The minimum setting length is 1 burst. NOTE *If all bits are zero (6'd0), it means that the maximum burst size or undefined burst size.*
15:13	RSVD	-	0	Reserved
12	TX_WD_SWAP	R/W	0	Word swap for dma_tx engine input data: 1'd0: Disable 1'd1: Enable
11	RX_WD_SWAP	R/W	0	Word swap for dma_rx engine output data:

All information provided in this document is subject to legal disclaimers.
© REALTEK 2024. All rights reserved.

				1'd0: Disable 1'd1: Enable
10	MAC_OUT_LITTLE_ENDIAN	R/W	0	Output mac is little endian: 1'd0: Big endian 1'd1: Little endian
9	DATA_OUT_LITTLE_ENDIAN	R/W	0	Output data is little endian: 1'd0: Big endian 1'd1: Little endian
8	TX_BYTE_SWAP	R/W	0	Byte swap for dma_tx engine input data: 1'd0: Disable 1'd1: Enable
7:6	RSVD	-	0	Reserved
5	AUTO_PADDING_SWAP	R/W	0	Byte swap for padding_len input data 1: Enable 0: Disable
4	DATA_IN_LITTLE_ENDIAN	R/W	0	Input data is little endian: 1'd0: Big endian 1'd1: Little endian
3	HASH_INITIAL_VALUE_SWAP	R/W	0	Byte swap for sequential hash initial value: 1'd0: Disable 1'd1: Enable
2	KEY_PAD_SWAP	R/W	0	Byte swap for HMAC key pad: 1'd0: Disable 1'd1: Enable
1	KEY_IV_SWAP	R/W	0	Byte swap for Cipher key and Initial Vector: 1'd0: Disable 1'd1: Enable
0	SET_SWAP	R/W	0	Byte swap for command setting data: 1'd0: Disable 1'd1: Enable

6.4.1.13 REG_IPSEKCR

- **Name:** Key Control Register
- **Size:** 32 bits
- **Address offset:** 034h
- **Read/write access:** R/W

This register is used to configure HMAC/AES key from OTP.

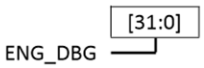


Bit	Symbol	Access	Reset	Description
31	EFUSE_KEY_REQ	R/W	0	Request of loading key from OTP, cleared by key_rdy
30:29	EFUSE_KEY_ID	R/W	0	2'b00: load key from OTP key1 2'b01: load key from OTP key2 2'b10: load key from OTP RSIP key_ecb 2'b11: load key from OTP RSIP key_ctr
28:0	RSVD	-	0	Reserved

6.4.1.14 REG_ENG_DBG

- **Name:** Crypto Engine Debug Register
- **Size:** 32 bits
- **Address offset:** 040h
- **Read/write access:** R

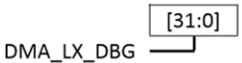
All information provided in this document is subject to legal disclaimers.
© REALTEK 2024. All rights reserved.



Bit	Symbol	Access	Reset	Description
31:0	ENG_DBG	R	0	Crypto engine debug info

6.4.1.15 REG_DMA_LX_DBG

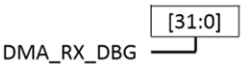
- **Name:** Crypto Engine DMA LX Bus Debug Register
- **Size:** 32 bits
- **Address offset:** 044h
- **Read/write access:** R



Bit	Symbol	Access	Reset	Description
31:0	DMA_LX_DBG	R	0	Crypto engine LX bus debug info

6.4.1.16 REG_DMA_RX_DBG

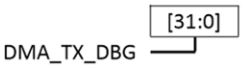
- **Name:** Crypto Engine DMA Rx Debug Register
- **Size:** 32 bits
- **Address offset:** 048h
- **Read/write access:** R



Bit	Symbol	Access	Reset	Description
31:0	DMA_RX_DBG	R	0	Crypto engine DMA Rx debug info

6.4.1.17 REG_DMA_TX_DBG

- **Name:** Crypto Engine DMA Tx Debug Register
- **Size:** 32 bits
- **Address offset:** 04Ch
- **Read/write access:** R



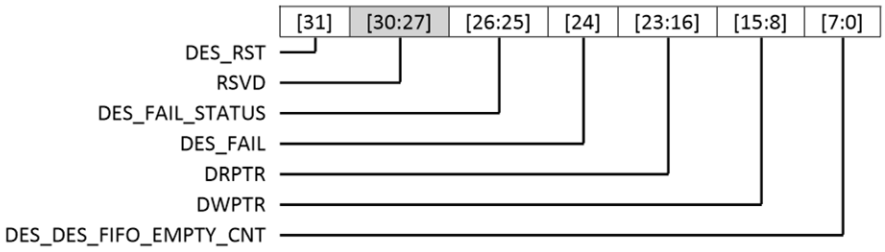
Bit	Symbol	Access	Reset	Description
31:0	DMA_TX_DBG	R	0	Crypto engine DMA Tx debug info

6.4.1.18 REG_DDSR

- **Name:** Destination Descriptor Status Register
- **Size:** 32 bits
- **Address offset:** 1000h
- **Read/write access:** R/W

This register is used to check Destination Descriptor FIFO status.

All information provided in this document is subject to legal disclaimers.
© REALTEK 2024. All rights reserved.

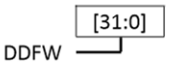


Bit	Symbol	Access	Reset	Description
31	DES_RST	R/W	0	Destination descriptor reset. (only for pk_up = 1'b1)
30:27	RSVD	-	0	Reserved
26:25	DES_FAIL_STATUS	R	0	Destination Descriptor failed status: 2'd1: Users write the first word twice consecutively 2'd2: Users write the second word directly without writing first word in the beginning 2'd3: Overflow (detect users try to write source descriptor, but there isn't available FIFO node to use).
24	DES_FAIL	R/W	0	Destination descriptor failed interrupt. Write 1 to clear this bit.
23:16	DRPTR	R	0	Destination descriptor FIFO read pointer. When engine reads a descriptor and finished it, DRPTR = DRPTR+2.
15:8	DWPTR	R	0	Destination descriptor FIFO write pointer. When writing descriptor to DDSW successfully, DWPTR = DWPTR+2.
7:0	DES_DES_FIFO_EMPTY_CNT	R	0	Destination descriptor FIFO empty counter. Use this field to check how many available FIFO nodes in Destination descriptor FIFO could use.

6.4.1.19 REG_DDFWR

- **Name:** Destination Descriptor First Word Register
- **Size:** 32 bits
- **Address offset:** 1004h
- **Read/write access:** R/W

This register is used to set descriptor header. The programmer can follow Destination Crypto Descriptor Type1/2 offset 0 to set header information data, and write this data to this register.

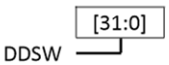


Bit	Symbol	Access	Reset	Description
31:0	DDFW	R/W	0	Destination Descriptor First Word

6.4.1.20 REG_DDSWR

- **Name:** Destination Descriptor Second Word Register
- **Size:** 32 bits
- **Address offset:** 1008h
- **Read/write access:** R/W

This register is used to set descriptor data buffer address. The programmer can know this register is a hardware pointer which records the data buffer address from Destination Crypto Descriptor Type1/2 offset 4. If the programmer sets a descriptor, need to write the header information data in first word register and write data buffer address in second word register.



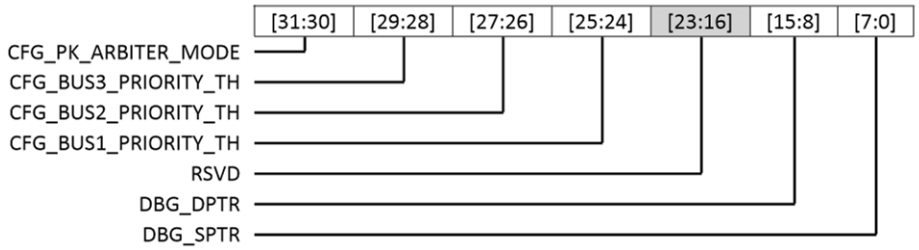
All information provided in this document is subject to legal disclaimers.
© REALTEK 2024. All rights reserved.

Bit	Symbol	Access	Reset	Description
31:0	DDSW	R/W	0	Destination Descriptor Second Word

6.4.1.21 REG_DES_PKTCONF

- **Name:** Packet Configure Register
- **Size:** 32 bits
- **Address offset:** 100Ch
- **Read/write access:** R/W

This register is used to configure packet arbiter and debug source/destination descriptor FIFO read pointer.



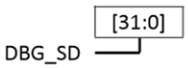
Bit	Symbol	Access	Reset	Description
31:30	CFG_PK_ARBITER_MODE	R/W	0	Arbiter rule between secure and no-secure access 0: round-robin 1/2/3: decided by the load_count of source descriptor FIFO of secure and no-secure, the more full is the winner.
29:28	CFG_BUS3_PRIORITY_TH	R/W	0	The priority of bus3 (reserved now)
27:26	CFG_BUS2_PRIORITY_TH	R/W	0	The priority of bus2-- security access. If 0x100C[31:30]=1, the load_count of source descriptor FIFO is wr_ptr - rd_ptr + 0x100C[27:26]. Otherwise, the load_count of source descriptor FIFO is wr_ptr - rd_ptr.
25:24	CFG_BUS1_PRIORITY_TH	R/W		The priority of bus2-- non-secure access. If 0x100C[31:30]=1, the load_count of source descriptor FIFO is wr_ptr - rd_ptr + 0x100C[25:24]. Otherwise, the load_count of source descriptor FIFO is wr_ptr - rd_ptr.
23:16	RSVD	-	0	Reserved
15:8	DBG_DPTR	R/W	0	Destination descriptor FIFO read pointer: Users read this pointer value can know what the ID number of FIFO node that is now pointed by read pointer is. If users want to know what the 32 bits value of a specified FIFO node, users can write an ID number to this read pointer, then get the 32 bits value from Debug Destination Descriptor Data Register.
7:0	DBG_SPTR	R/W	0	Source descriptor FIFO read pointer: Users read this pointer value can know what the ID number of FIFO node that is now pointed by read pointer is. If users want to know what the 32 bits value of a specified FIFO node, users can write an ID number to this read pointer, then get the 32 bits value from Debug Source Descriptor Data Register.

6.4.1.22 REG_DBGSDR

- **Name:** Debug Source Descriptor Data Register
- **Size:** 32 bits
- **Address offset:** 1010h
- **Read/write access:** R

This register helps to debug source descriptor FIFO read pointer data.

All information provided in this document is subject to legal disclaimers.
© REALTEK 2024. All rights reserved.

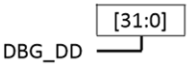


Bit	Symbol	Access	Reset	Description
31:0	DBG_SD	R	0	Debug Source Descriptor Data: This register is used with DBG_SPTR.

6.4.1.23 REG_DBGDDR

- **Name:** Debug Destination Descriptor Data Register
- **Size:** 32 bits
- **Address offset:** 1014h
- **Read/write access:** R

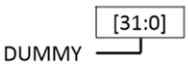
This register helps to debug destination descriptor FIFO read pointer data.



Bit	Symbol	Access	Reset	Description
31:0	DBG_DD	R	0	Debug Destination Descriptor Data: This register is used with DBG_DPTR.

6.4.1.24 REG_DUMMY

- **Name:** Dummy Register
- **Size:** 32 bits
- **Address offset:** 1018h
- **Read/write access:** R/W



Bit	Symbol	Access	Reset	Description
31:0	DUMMY	R/W	0	HW used only (reserved)

6.4.2 Hash Registers

Name	Address offset	Access	Description
REG_SDSR	000h	R/W	Source Descriptor Status Register
REG_SDFWR	004h	R/W	Source Descriptor First Word Register
REG_SDSWR	008h	R/W	Source Descriptor Second Word Register
REG_IPSCSR_RSTEACONFISR	010h	R/W	Reset Engine and Configure Interrupt Service Routine Register
REG_IPSCSR_INTM	014h	R/W	Interrupt Mask Register
REG_IPSCSR_DBG	018h	R/W	Debug Register
REG_IPSCSR_ERR_INT	01Ch	R/W	Error Interrupt Register
REG_IPSCSR_SADDLR	020h	R	Sum of Additional Authentication Data Length Register
REG_IPSCSR_SENLR	024h	R	Sum of Encryption Data Length Register
REG_IPSCSR_SAPLR	028h	R	Sum of Authentication Padding Data Length Register
REG_IPSCSR_SEPLR	02Ch	R	Sum of Encryption Padding Data Length Register
REG_IPSCSR_SWAPABURSTR	030h	R/W	Swap and Burst Register
REG_IPSEKCR	034h	R/W	Key Control Register
REG_ENG_DBG	040h	R	Crypto Engine Debug Register
REG_DMA_LX_DBG	044h	R	Crypto Engine LX Bus Debug Register
REG_DMA_RX_DBG	048h	R	Crypto Engine DMA Rx Debug Register
REG_DMA_TX_DBG	04Ch	R	Crypto Engine DMA Tx Debug Register

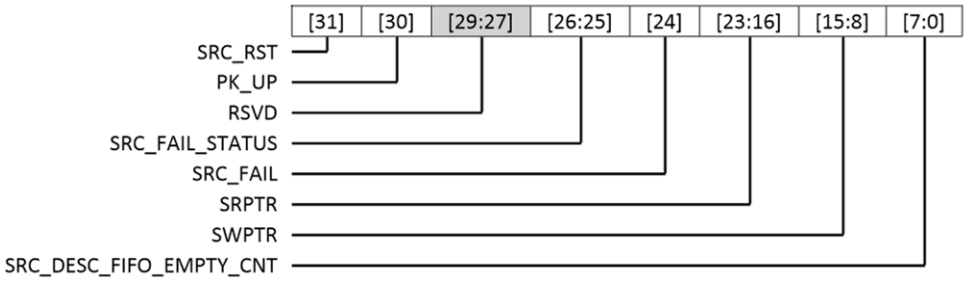
All information provided in this document is subject to legal disclaimers.
© REALTEK 2024. All rights reserved.

REG_DDSR	1000h	R/W	Destination Descriptor Status Register
REG_DDFWR	1004h	R/W	Destination Descriptor First Word Register
REG_DDSWR	1008h	R/W	Destination Descriptor Second Word Register
REG_DES_PKTCONF	100Ch	R/W	Packet Configure Register
REG_DBGSDR	1010h	R	Debug Source Descriptor Data Register
REG_DBGDDR	1014h	R	Debug Destination Descriptor Data Register
REG_DUMMY	1018h	R/W	Dummy Register

6.4.2.1 REG_SDSR

- **Name:** Source Descriptor Status Register
- **Size:** 32 bits
- **Address offset:** 000h
- **Read/write access:** R/W

This register is used to check source descriptor FIFO status.

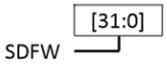


Bit	Symbol	Access	Reset	Description
31	SRC_RST	R/W	0	Source descriptor reset (only for pk_up = 1'b1)
30	PK_UP	R/W	1	Packet base update wptr to engine. If pk_up =1, the total number of souce descriptor in one packet can't be over 16, and the total number of destination descriptor in one packet can't be over 8.
29:27	RSVD	-	0	Reserved
26:25	SRC_FAIL_STATUS	R	0	Reserved
24	SRC_FAIL	R/W	0	Source descriptor failed interrupt. Write 1 to clear this bit.
23:16	SRPTR	R	0	Source descriptor FIFO read pointer. When engine reads a descriptor and finishes it, SRPTR = SRPTR+2.
15:8	SWPTR	R	0	Source descriptor FIFO write pointer. When write descriptor to SDSW successfully, SWPTR = SWPTR+2.
7:0	SRC_DESC_FIFO_EMPTY_CNT	R	0	Source descriptor FIFO empty counter. Use this field to check how many available FIFO nodes in source descriptor FIFO to use.

6.4.2.2 REG_SDFWR

- **Name:** Source Descriptor First Word Register
- **Size:** 32 bits
- **Address offset:** 004h
- **Read/write access:** R/W

This register is used to set descriptor header. The programmer can follow Source Crypto Descriptor Type1/2 offset 0 to set header information data, and write this data to this register.



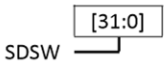
All information provided in this document is subject to legal disclaimers.
© REALTEK 2024. All rights reserved.

Bit	Symbol	Access	Reset	Description
31:0	SDFW	R/W	0	Source Descriptor First Word

6.4.2.3 REG_SDSWR

- **Name:** Source Descriptor Second Word Register
- **Size:** 32 bits
- **Address offset:** 008h
- **Read/write access:** R/W

This register is used to set descriptor data buffer address. The programmer can know this register is a hardware pointer which records the data buffer address from Source Crypto Descriptor Type1/2 offset 4. If the programmer sets a descriptor, need to write the header information data in first word register and write data buffer address in second word register.

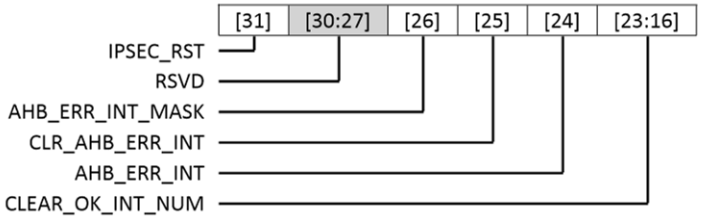


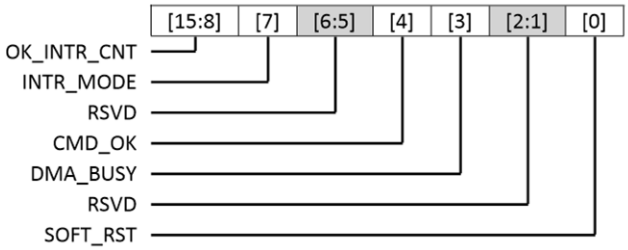
Bit	Symbol	Access	Reset	Description
31:0	SDSW	R/W	0	Source Descriptor Second Word

6.4.2.4 REG_IPSCSR_RSTEACONFISR

- **Name:** Reset Engine and Configure Interrupt Service Routine Register
- **Size:** 32 bits
- **Address offset:** 010h
- **Read/write access:** R/W

This register allows the programmer can reset crypto engine and DMA engine, check and configure crypto engine interrupt status.





Bit	Symbol	Access	Reset	Description
31	IPSEC_RST	R/W	0	IPsec engine reset. Write '1' to reset the crypto engine and DMA engine (used to clear fatal error)
30:27	RSVD	-	0	Reserved
26	AHB_ERR_INT_MASK	R/W	1	
25	CLR_AHB_ERR_INT	R/W	0	
24	AHB_ERR_INT	R	0	
23:16	CLEAR_OK_INT_NUM	R/W	1	Clear ok interrupt number. In interrupt counter mode, if the programmer wants to clear ok_intr_cnt, need to write ok_intr_cnt to this field first, then write '1' to cmd_ok.
15:8	OK_INTR_CNT	R	0	Ok interrupt counter. Reading this field can know how many interrupts are coming to notify crypto engine

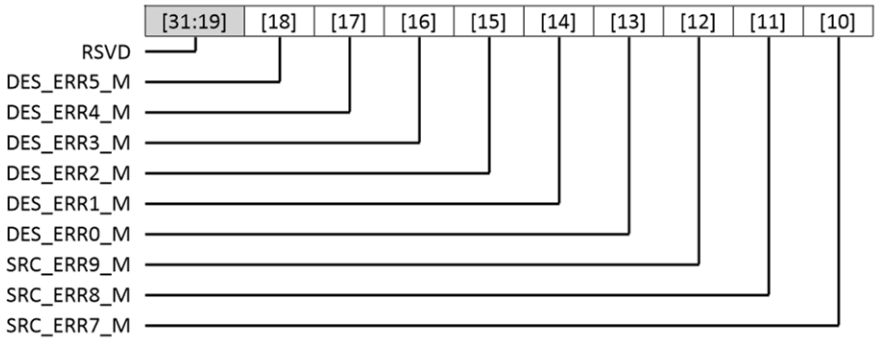
All information provided in this document is subject to legal disclaimers.
© REALTEK 2024. All rights reserved.

				has calculated cryptographic feature results. Use this field in interrupt counter mode.
7	INTR_MODE	R/W	0	Select ok interrupt mode: 1'd0: Interrupt mode is general mode. 1'd1: Interrupt mode is counter mode. General mode means no matter how many interrupts are coming to notify crypto engine has calculated cryptographic feature results, there always is one interrupt signal shows in cmd_ok field, and nothing in ok_intr_cnt. So if the programmer want to clear this signal, just write '1' to cmd_ok. Counter mode means the programmer can know how many interrupts are coming to notify crypto engine has calculated cryptographic feature results from ok_intr_cnt. The programmer also can know there are at least one interrupt from reading cmd_ok. So if the programmer want to clear ok_intr_cnt signals, need to write ok_intr_cnt into lear_ok_int_num, then write 1 to cmd_ok.
6:5	RSVD	-	0	Reserved
4	CMD_OK	R/W	0	Command ok Interrupt. Reading this field to detect whether crypto engine has calculated a cryptographic feature result. Even if interrupt mode is counter mode, still can use this field to detect whether an interrupt is coming. Write 1 to clear this interrupt signal.
3	DMA_BUSY	R	0	Detect whether IPsec DMA is busy: 1'd0: DMA is not busy 1'd1: DMA is busy Using for debug to avoid data coherence issue[when user get crypto finish calculated interrupt, user can make sure DMA engine has already move data to result buffer.
2:1	RSVD	-	0	Reserved
0	SOFT_RST	W	0	Software Reset. Write 1 to reset the crypto engine.

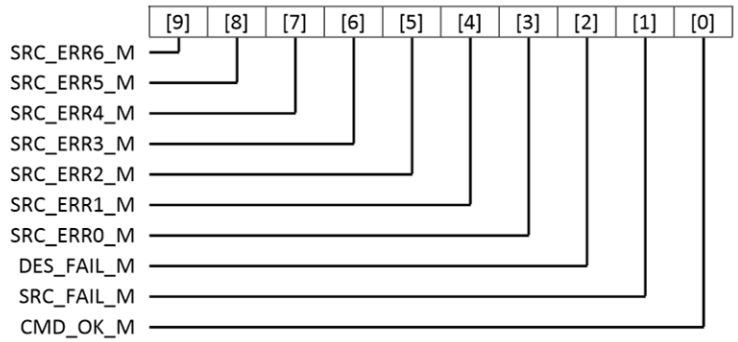
6.4.2.5 REG_IPSCSR_INTM

- **Name:** Interrupt Mask Register
- **Size:** 32 bits
- **Address offset:** 014h
- **Read/write access:** R/W

This register is used to enable or disable command ok interrupt mask, source descriptor error interrupt mask and destination descriptor error interrupt mask.



All information provided in this document is subject to legal disclaimers.
© REALTEK 2024. All rights reserved.



Bit	Symbol	Access	Reset	Description
31:19	RSVD	-	0	Reserved
18	DES_ERR5_M	R/W	1	Destination Descriptor Error 5 Interrupt Mask 1'd0: Disable mask 1'd1: Enable mask
17	DES_ERR4_M	R/W	1	Destination Descriptor Error 4 Interrupt Mask 1'd0: Disable mask 1'd1: Enable mask
16	DES_ERR3_M	R/W	1	Destination Descriptor Error 3 Interrupt Mask 1'd0: Disable mask 1'd1: Enable mask
15	DES_ERR2_M	R/W	1	Destination Descriptor Error 2 Interrupt Mask 1'd0: Disable mask 1'd1: Enable mask
14	DES_ERR1_M	R/W	1	Destination Descriptor Error 1 Interrupt Mask 1'd0: Disable mask 1'd1: Enable mask
13	DES_ERR0_M	R/W	1	Destination Descriptor Error 0 Interrupt Mask 1'd0: Disable mask 1'd1: Enable mask
12	SRC_ERR9_M	R/W	1	Source Descriptor Error 9 Interrupt Mask 1'd0: Disable mask 1'd1: Enable mask
11	SRC_ERR8_M	R/W	1	Source Descriptor Error 8 Interrupt Mask 1'd0: Disable mask 1'd1: Enable mask
10	SRC_ERR7_M	R/W	1	Source Descriptor Error 7 Interrupt Mask 1'd0: Disable mask 1'd1: Enable mask
9	SRC_ERR6_M	R/W	1	Source Descriptor Error 6 Interrupt Mask 1'd0: Disable mask 1'd1: Enable mask
8	SRC_ERR5_M	R/W	1	Source Descriptor Error 5 Interrupt Mask 1'd0: Disable mask 1'd1: Enable mask
7	SRC_ERR4_M	R/W	1	Source Descriptor Error 4 Interrupt Mask 1'd0: Disable mask 1'd1: Enable mask
6	SRC_ERR3_M	R/W	1	Source Descriptor Error 3 Interrupt Mask 1'd0: Disable mask 1'd1: Enable mask
5	SRC_ERR2_M	R/W	1	Source Descriptor Error 2 Interrupt Mask 1'd0: Disable mask 1'd1: Enable mask
4	SRC_ERR1_M	R/W	1	Source Descriptor Error 1 Interrupt Mask 1'd0: Disable mask 1'd1: Enable mask
3	SRC_ERR0_M	R/W	1	Source Descriptor Error 0 Interrupt Mask

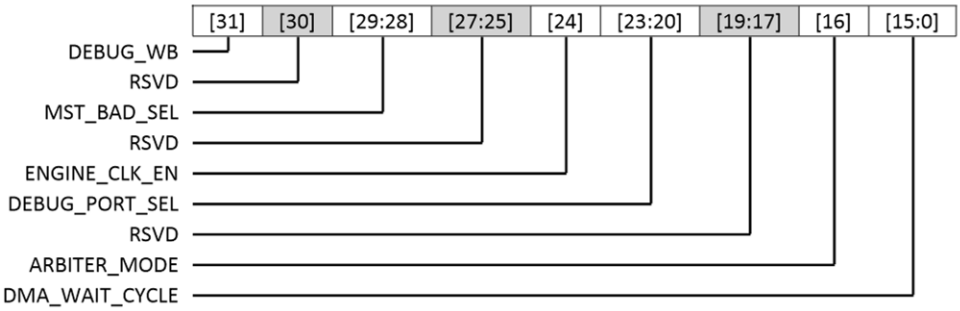
All information provided in this document is subject to legal disclaimers.
© REALTEK 2024. All rights reserved.

				1'd0: Disable mask 1'd1: Enable mask
2	DES_FAIL_M	R/W	1	Source Descriptor fail interrupt mask. 1'd0: Disable mask 1'd1: Enable mask
1	SRC_FAIL_M	R/W	1	Destination Descriptor fail interrupt mask. 1'd0: Disable mask 1'd1: Enable mask
0	CMD_OK_M	R/W	1	Command ok Interrupt Mask. 1'd0: Disable mask 1'd1: Enable mask If the programmer wants to disable cmd_ok interrupt signal, write '1' to enable interrupt mask.

6.4.2.6 REG_IPSCSR_DBG

- **Name:** Debug Register
- **Size:** 32 bits
- **Address offset:** 018h
- **Read/write access:** R/W

This register is used to configure or debug IPsec DMA engine.



Bit	Symbol	Access	Reset	Description
31	DEBUG_WB	R/W	0	Debug: write back mode. 1'd0: Disable DMA write back mode. 1'd1: Enable DMA write back mode. Enable this field, crypto DMA will move data from source address to destination address. During this period, the data won't be processed in any calculation (encryption, decryption, hash), then the programmer can check whether the source data is the same as destination data.
30	RSVD	-	0	Reserved
29:28	MST_BAD_SEL	R/W	0	(For debug port) select Lexra master data bus (option) 2'd0: mst_ipsec_bad[07:00] 2'd1: mst_ipsec_bad[15:08] 2'd2: mst_ipsec_bad[23:16] 2'd3: mst_ipsec_bad[31:24]
27:25	RSVD	-	0	Reserved
24	ENGINE_CLK_EN	R/W	0	IPsec Engine clock enable 1'd0: Disable IPsec engine clock 1'd1: Enable IPsec engine clock
23:20	DEBUG_PORT_SEL	R/W	0	Debug port selection 4'd0: engine_debug 4'd1: dma_lexra_debug 4'd2: dma_rx_debug 4'd3: dma_tx_debug
19:17	RSVD	-	0	Reserved
16	ARBITER_MODE	R/W	1	DMA arbiter mode 1'd0: round-robin

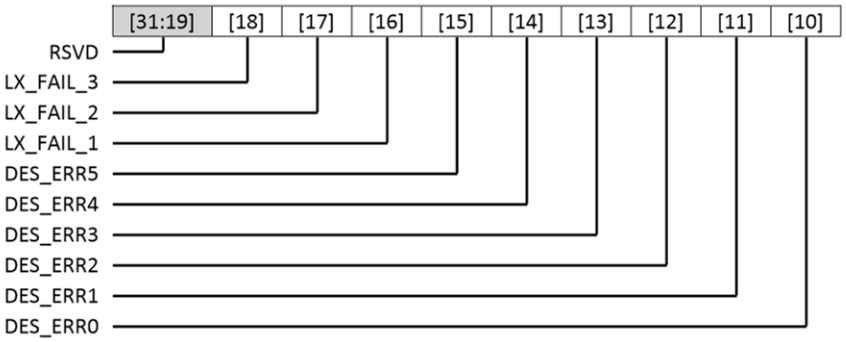
All information provided in this document is subject to legal disclaimers.
© REALTEK 2024. All rights reserved.

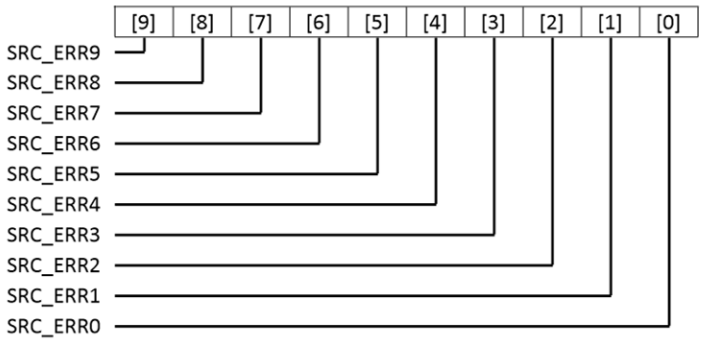
				1'd1: detect FIFO wasted level There are 2 FIFOs in DMA, which are read-FIFO and write-FIFO. Detecting FIFO wasted level means that DMA will process which FIFO first depends on which wasted level is high. Because the wasted level is high means there are many FIFO nodes waiting to be processed in this FIFO, so DMA must process them in hurry.
15:0	DMA_WAIT_CYCLE	R/W	0	Setting DMA wait cycles to assert next DMA request

6.4.2.7 REG_IPSCSR_ERR_INT

- **Name:** Error Interrupt Register
- **Size:** 32 bits
- **Address offset:** 01Ch
- **Read/write access:** R/W

This register is used to detect source/destination descriptor setting error interrupts.





Bit	Symbol	Access	Reset	Description
31:19	RSVD	-	0	Reserved
18	LX_FAIL_3	RW1C	0	When a burst done, burst length != burst cnt. Write “1” to clear this error signal.
17	LX_FAIL_2	RW1C	0	When IPsec is working, rdata from bus != rdata of ipsec or wdata of ipsec != wdata on bus. Write “1” to clear this error signal.
16	LX_FAIL_1	RW1C	0	Lx bus timing issue Write “1” to clear this error signal.
15	DES_ERR5	RW1C	0	Destination Descriptor Error 5 Interrupt. The error condition is authentication digest length error in destination descriptor. MD5: adl != 16 or SHA1: adl != 20 or SHA2-224: adl != 28 or adl != 32 when set sequential hash and not sequential hash last SHA2-256: adl != 32 or SHA2-384: adl != 48 or adl != 64 when set sequential hash and not sequential hash last SHA2-512: adl != 64 Write '1' to clear this error signal.
14	DES_ERR4	RW1C	0	Destination Descriptor Error 4 Interrupt. The error condition is authentication digest length error in destination descriptor.

All information provided in this document is subject to legal disclaimers.
© REALTEK 2024. All rights reserved.

				MD5: adl > 16 or SHA1: adl > 20 or SHA2-224: adl > 28 or adl > 32when set sequential hash and not sequential hash last SHA2-256: adl > 32 or SHA2-384: adl > 48 or adl > 64when set sequential hash and not sequential hash last SHA2-512: adl > 64 Write '1' to clear this error signal.
13	DES_ERR3	RW1C	0	Destination Descriptor Error 3 Interrupt. One of the error condition is that the sum of enl is greater than Encryption Total Length. The other error condition is about destination descriptor length error ((enl(Encryption length)==0) & (adl(authentication data length)==0)). Write '1' to clear this error signal. Sum of enl reference the register REG_IPSCSR_SENLR (offset 0x24).
12	DES_ERR2	RW1C	0	Destination Descriptor Error 2 Interrupt. The error condition is the bit field ENC=1'b0 (active Hash engine to process hash data), but only active cipher engine. Write '1' to clear this error signal.
11	DES_ERR1	RW1C	0	Destination Descriptor Error 1 Interrupt. The error condition is the bit field ENC=1'b1 (active cipher engine to process cipher data), but only active hash engine. Write '1' to clear this error signal.
10	DES_ERR0	RW1C	0	Destination Descriptor Error 0 Interrupt. One of the error condition is destination descriptor starting address error (DDSA[1:0]!=2'd0 [at least 4 bytes-aligned]). The other error condition is memory protection event. Write '1' to clear this error signal.
9	SRC_ERR9	RW1C	0	Source Descriptor Error 9 Interrupt. The error condition is that sum of epl is not equal to EPTL (Encryption Padding Total Length). Write '1' to clear this error signal. Sum of epl reference the register REG_IPSCSR_SEPLR (offset 0x2C).
8	SRC_ERR8	RW1C	0	Source Descriptor Error 8 Interrupt. The error condition is that sum of apl is not equal to HPTL (Hash Padding Total Length). Write '1' to clear this error signal. Sum of apl reference the register REG_IPSCSR_SAPLR (offset 0x28).
7	SRC_ERR7	RW1C	0	Source Descriptor Error 7 Interrupt. The error condition is that sum of enl is not equal to ETL (Encryption Total Length). Write '1' to clear this error signal. Sum of enl reference the register REG_IPSCSR_SENLR (offset 0x24).
6	SRC_ERR6	RW1C	0	Source Descriptor Error 6 Interrupt. The error condition is that sum of a2eo is not equal to HTL (Header Total Length). Write '1' to clear this error signal. Sum of a2eo reference the register REG_IPSCSR_SAADLR (offset 0x20).
5	SRC_ERR5	RW1C	0	Source Descriptor Error 5 Interrupt. The error condition is that there is a source descriptor, but the data length ((a2eo/epl) +(enl) + (apl)) is 0. Write '1' to clear this error signal.
4	SRC_ERR4	RW1C	0	Source Descriptor Error 4 Interrupt. The error condition is source data buffer pointer error (SDBP[1:0]!=2'd0 [at least 4 byte aligned]). Write '1' to clear this error signal.
3	SRC_ERR3	RW1C	0	Source Descriptor Error 3 Interrupt. About source descriptor length error. One of the error condition is setting the first descriptor but (CL!=3). The other error condition is there is a source descriptor but CL, KL, IL, PL, SHIVL are all 0. Write '1' to clear this error signal.
2	SRC_ERR2	RW1C	0	Source Descriptor Error 2 Interrupt. One of the error condition is source descriptor starting address error (SDSA[1:0]!=2'd0 [at least 4 bytes-aligned]). The other error condition is memory protection event. Write '1' to clear this error signal.
1	SRC_ERR1	RW1C	0	Source Descriptor Error 1 Interrupt. The error condition is last segment descriptor is setting (LS=1), when descriptor is pointing to the first segment of the packet. Write '1' to clear this error signal.
0	SRC_ERR0	RW1C	0	Source Descriptor Error 0 Interrupt.

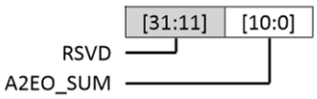
All information provided in this document is subject to legal disclaimers.
© REALTEK 2024. All rights reserved.

				One of the error condition is first segment descriptor is not set (FS=0), when descriptor is pointing to the first segment of the packet. The other error condition is first segment descriptor is set (FS=1), when descriptor is not pointing to the first segment of the packet. Write '1' to clear this error signal.

6.4.2.8 REG_IPSCSR_SADDLR

- **Name:** Sum of Additional Authentication Data Length Register
- **Size:** 32 bits
- **Address offset:** 020h
- **Read/write access:** R

This register is used to get sum of additional authentication data length.

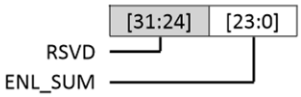


Bit	Symbol	Access	Reset	Description
31:11	RSVD	-	0	Reserved
10:0	A2EO_SUM	R	0	Sum of additional authentication data length.

6.4.2.9 REG_IPSCSR_SENLR

- **Name:** Sum of Encryption Data Length Register
- **Size:** 32 bits
- **Address offset:** 024h
- **Read/write access:** R

This register is used to get sum of encryption data length.

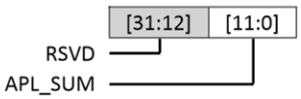


Bit	Symbol	Access	Reset	Description
31:24	RSVD	-	0	Reserved
23:0	ENL_SUM	R	0	Sum of encryption data length.

6.4.2.10 REG_IPSCSR_SAPLR

- **Name:** Sum of Authentication Padding Data Length Register
- **Size:** 32 bits
- **Address offset:** 028h
- **Read/write access:** R

This register is used to get sum of authentication padding data length.



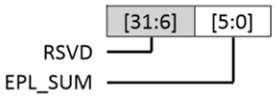
Bit	Symbol	Access	Reset	Description
31:12	RSVD	-	0	Reserved
11:0	APL_SUM	R	0	Sum of authentication padding data length. [Mix mode uses this field]

All information provided in this document is subject to legal disclaimers.
© REALTEK 2024. All rights reserved.

6.4.2.11 REG_IPSCSR_SEPLR

- **Name:** Sum of Encryption Padding Data Length Register
- **Size:** 32 bits
- **Address offset:** 02Ch
- **Read/write access:** R

This register is used to get sum of encryption padding data length.

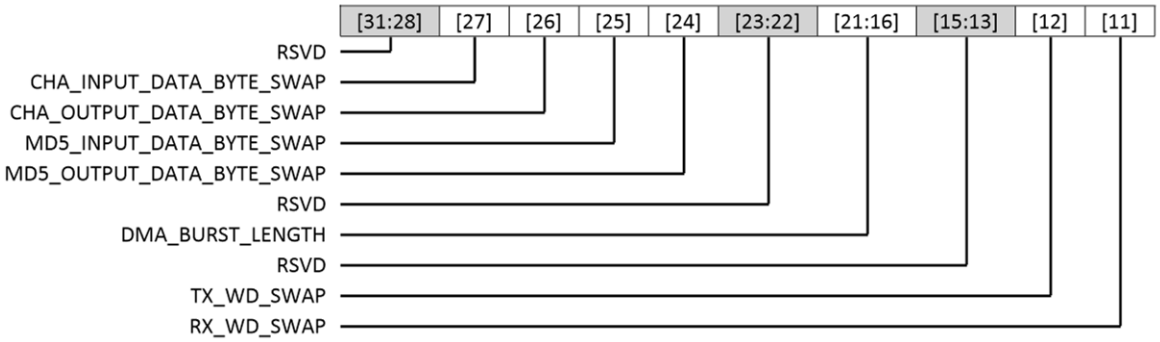


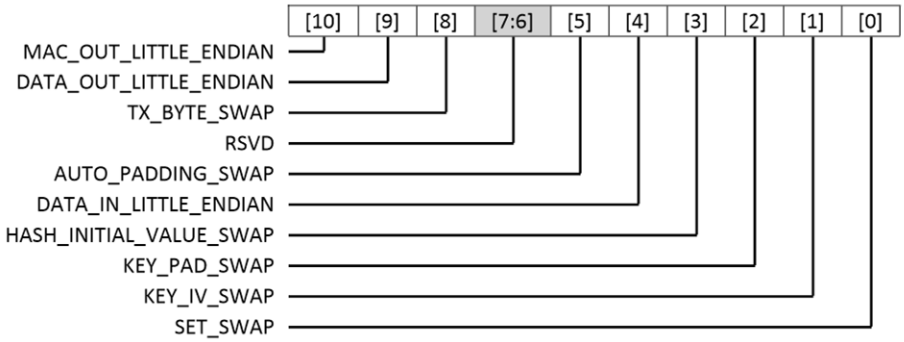
Bit	Symbol	Access	Reset	Description
31:6	RSVD	-	0	Reserved
5:0	EPL_SUM	R	0	Sum of encryption padding data length. [Mix mode uses this field]

6.4.2.12 REG_IPSCSR_SWAPABURSTR

- **Name:** Swap and Burst Register
- **Size:** 32 bits
- **Address offset:** 030h
- **Read/write access:** R/W

This register configures swap and endian setting in crypto engine, and DMA burst length.





Bit	Symbol	Access	Reset	Description
31:28	RSVD	-	0	Reserved
27	CHA_INPUT_DATA_BYTE_SWAP	R/W	0	Byte swap for CHA input data 1: Enable 0: Disable
26	CHA_OUTPUT_DATA_BYTE_SWAP	R/W	0	Byte swap for CHA output data 1: Enable 0: Disable

All information provided in this document is subject to legal disclaimers.
© REALTEK 2024. All rights reserved.

25	MD5_INPUT_DATA_BYTE_SWAP	R/W	0	Byte swap for MD5 input data 1: Enable 0: Disable
24	MD5_OUTPUT_DATA_BYTE_SWAP	R/W	0	Byte swap for MD5 output data 1: Enable 0: Disable
23:22	RSVD	-	0	Reserved
21:16	DMA_BURST_LENGTH	R/W	0x10	Set DMA burst length: The maximum setting length is 16 bursts. The minimum setting length is 1 burst. NOTE *If all bits are zero (6'd0), it means maximum burst size or undefined burst size.*
15:13	RSVD	-	0	Reserved
12	TX_WD_SWAP	R/W	0	Word swap for dma_tx engine input data: 1'd0: Disable 1'd1: Enable
11	RX_WD_SWAP	R/W	0	Word swap for dma_rx engine output data: 1'd0: Disable 1'd1: Enable
10	MAC_OUT_LITTLE_ENDIAN	R/W	0	Output mac is little endian: 1'd0: Big endian 1'd1: Little endian
9	DATA_OUT_LITTLE_ENDIAN	R/W	0	Output data is little endian: 1'd0: Big endian 1'd1: Little endian
8	TX_BYTE_SWAP	R/W	0	Byte swap for dma_tx engine input data: 1'd0: Disable 1'd1: Enable
7:6	RSVD	-	0	Reserved
5	AUTO_PADDING_SWAP	R/W	0	Byte swap for padding_len input data 1: Enable 0: Disable
4	DATA_IN_LITTLE_ENDIAN	R/W	0	Input data is little endian: 1'd0: Big endian 1'd1: Little endian
3	HASH_INITIAL_VALUE_SWAP	R/W	0	Byte swap for sequential hash initial value: 1'd0: Disable 1'd1: Enable
2	KEY_PAD_SWAP	R/W	0	Byte swap for HMAC key pad: 1'd0: Disable 1'd1: Enable
1	KEY_IV_SWAP	R/W	0	Byte swap for Cipher key and Initial Vector: 1'd0: Disable 1'd1: Enable
0	SET_SWAP	R/W	0	Byte swap for command setting data: 1'd0: Disable 1'd1: Enable

6.4.2.13 REG_IPSEKCR

- **Name:** Key Control Register
- **Size:** 32 bits
- **Address offset:** 034h
- **Read/write access:** R/W

This register is used to configure HMAC key/AES key from OTP.

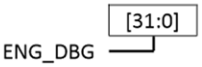
All information provided in this document is subject to legal disclaimers.
© REALTEK 2024. All rights reserved.



Bit	Symbol	Access	Reset	Description
31	EFUSE_KEY_REQ	R/W	0	Request of loading key from OTP, cleared by key_rdy
30:29	EFUSE_KEY_ID	R/W	0	2'b00/2'b01: load key from OTP key1 2'b10/2'b11: load key from OTP key2
28:0	RSVD	-	0	Reserved

6.4.2.14 REG_ENG_DBG

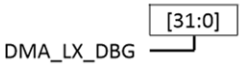
- **Name:** Crypto Engine Debug Register
- **Size:** 32 bits
- **Address offset:** 040h
- **Read/write access:** R



Bit	Symbol	Access	Reset	Description
31:0	ENG_DBG	R	0	Crypto engine debug info

6.4.2.15 REG_DMA_LX_DBG

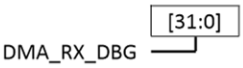
- **Name:** Crypto Engine LX Bus Debug Register
- **Size:** 32 bits
- **Address offset:** 044h
- **Read/write access:** R



Bit	Symbol	Access	Reset	Description
31:0	DMA_LX_DBG	R	0	Crypto engine LX bus debug info

6.4.2.16 REG_DMA_RX_DBG

- **Name:** Crypto Engine DMA Rx Debug Register
- **Size:** 32 bits
- **Address offset:** 048h
- **Read/write access:** R

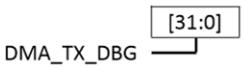


Bit	Symbol	Access	Reset	Description
31:0	DMA_RX_DBG	R	0	Crypto engine DMA Rx debug info

6.4.2.17 REG_DMA_TX_DBG

- **Name:** Crypto Engine DMA Tx Debug Register
- **Size:** 32 bits
- **Address offset:** 04Ch
- **Read/write access:** R

All information provided in this document is subject to legal disclaimers.
© REALTEK 2024. All rights reserved.

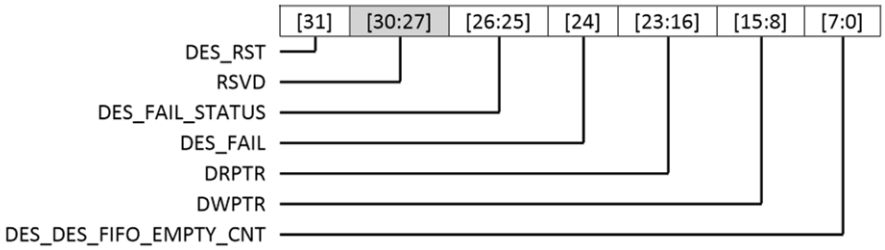


Bit	Symbol	Access	Reset	Description
31:0	DMA_TX_DBG	R	0	Crypto engine DMA Tx debug info

6.4.2.18 REG_DDSR

- **Name:** Destination Descriptor Status Register
- **Size:** 32 bits
- **Address offset:** 1000h
- **Read/write access:** R/W

This register is used to check Destination Descriptor FIFO status.

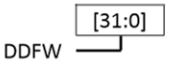


Bit	Symbol	Access	Reset	Description
31	DES_RST	R/W	0	Destination descriptor reset. (only for pk_up = 1'b1)
30:27	RSVD	-	0	Reserved
26:25	DES_FAIL_STATUS	R	0	Destination Descriptor failed status: 2'd1: Users write the first word twice consecutively 2'd2: Users write the second word directly without writing first word in the beginning 2'd3: Overflow (detect users try to write source descriptor, but there isn't available FIFO node to use).
24	DES_FAIL	R/W	0	Destination descriptor failed interrupt. Write 1 to clear this bit.
23:16	DRPTR	R	0	Destination descriptor FIFO read pointer. When engine reads a descriptor and finishes it, DRPTR = DRPTR+2.
15:8	DWPTR	R	0	Destination descriptor FIFO write pointer. When write descriptor to DDSW successfully, DWPTR = DWPTR+2.
7:0	DES_DES_FIFO_EMPTY_CNT	R	0	Destination descriptor FIFO empty counter. Use this field to check how many available FIFO nodes in destination descriptor FIFO could use.

6.4.2.19 REG_DDFWR

- **Name:** Destination Descriptor First Word Register
- **Size:** 32 bits
- **Address offset:** 1004h
- **Read/write access:** R/W

This register is used to set descriptor header. The programmer can follow Destination Crypto Descriptor Type1/2 offset 0 to set header information data, and write this data to this register.



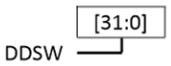
Bit	Symbol	Access	Reset	Description
31:0	DDFW	R/W	0	Destination Descriptor First Word

All information provided in this document is subject to legal disclaimers.
© REALTEK 2024. All rights reserved.

6.4.2.20 REG_DDSWR

- **Name:** Destination Descriptor Second Word Register
- **Size:** 32 bits
- **Address offset:** 1008h
- **Read/write access:** R/W

This register is used to set descriptor data buffer address. The programmer can know this register is a hardware pointer which records the data buffer address from Destination Crypto Descriptor Type1/2 offset 4. If the programmer sets a descriptor, need to write the header information data in first word register and write data buffer address in second word register.

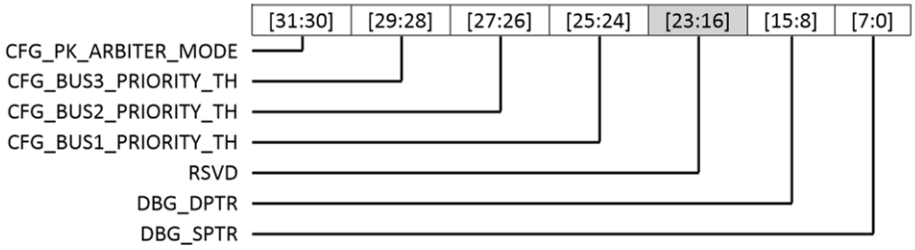


Bit	Symbol	Access	Reset	Description
31:0	DDSW	R/W	0	Destination Descriptor Second Word

6.4.2.21 REG_DES_PKTCONF

- **Name:** Packet Configure Register
- **Size:** 32 bits
- **Address offset:** 100Ch
- **Read/write access:** R/W

This register is used to configure packet arbiter and debug Source/Destination descriptor FIFO read pointer.



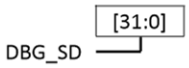
Bit	Symbol	Access	Reset	Description
31:30	CFG_PK_ARBITER_MODE	R/W	0	Arbiter rule between secure and no-secure access 0: round-robin 1/2/3: decided by the load_count of source descriptor FIFO of secure and no-secure, the more full is the winner.
29:28	CFG_BUS3_PRIORITY_TH	R/W	0	The priority of bus3 (reserved now)
27:26	CFG_BUS2_PRIORITY_TH	R/W	0	The priority of bus2-- secure access. If 0x100C[31:30]=1, the load_count of source descriptor FIFO is wr_ptr - rd_ptr + 0x100C[27:26]; otherwise, the load_count of source descriptor FIFO is wr_ptr - rd_ptr.
25:24	CFG_BUS1_PRIORITY_TH	R/W		The priority of bus2-- non-secure access. If 0x100C[31:30]=1, the load_count of source descriptor FIFO is wr_ptr - rd_ptr + 0x100C[25:24]; otherwise, the load_count of source descriptor FIFO is wr_ptr - rd_ptr.
23:16	RSVD	-	0	Reserved
15:8	DBG_DPTR	R/W	0	Destination descriptor FIFO read pointer: Users read this pointer value can know what the ID number of FIFO node that is now pointed by read pointer is. If users want to know what the 32 bits value of a specified FIFO node, users can write an ID number to this read pointer, then get the 32 bits value from Debug Destination Descriptor Data Register.
7:0	DBG_SPTR	R/W	0	Source descriptor FIFO read pointer: Users read this pointer value can know what the ID number of FIFO node that is now pointed by read pointer is. If users want to know what the 32 bits value of a specified FIFO node, users can write an ID number to this read pointer, then get the 32 bits value from Debug Source Descriptor Data Register.

All information provided in this document is subject to legal disclaimers.
© REALTEK 2024. All rights reserved.

6.4.2.22 REG_DBGSDR

- **Name:** Debug Source Descriptor Data Register
- **Size:** 32 bits
- **Address offset:** 1010h
- **Read/write access:** R

This register helps to debug source descriptor FIFO read pointer data.

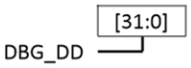


Bit	Symbol	Access	Reset	Description
31:0	DBG_SD	R	0	Debug Source Descriptor Data. This register is used with DBG_SPTR.

6.4.2.23 REG_DBGDDR

- **Name:** Debug Destination Descriptor Data Register
- **Size:** 32 bits
- **Address offset:** 1014h
- **Read/write access:** R

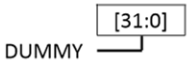
This register helps to debug destination descriptor FIFO read pointer data.



Bit	Symbol	Access	Reset	Description
31:0	DBG_DD	R	0	Debug Destination Descriptor Data. This register is used with DBG_DPTR.

6.4.2.24 REG_DUMMY

- **Name:** Dummy Register
- **Size:** 32 bits
- **Address offset:** 1018h
- **Read/write access:** R/W



Bit	Symbol	Access	Reset	Description
31:0	DUMMY	R/W	0	HW used only (reserved)

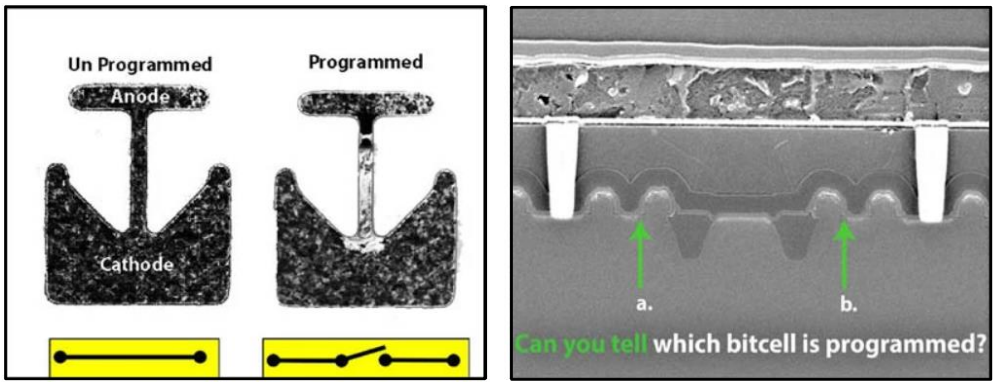
All information provided in this document is subject to legal disclaimers.
© REALTEK 2024. All rights reserved.

7 One-Time-Programmable Controller (OTPC)

7.1 Introduction

For system secure requirements, non-volatile information storage becomes more and more important nowadays. Besides non-volatile requirements, access privilege control of the storage media for different application scenarios is another significant topic for a secure system. For RTL8721Dx, an OTP device is used to achieve this goal.

Among all embedded non-volatile memory (eNVM) such as embedded flash, electrical fuses (eFuse), multi-time programmable (MTP), and antifuse one-time programmable (OTP), OTP is the most secure today. OTP cannot be hacked using passive, semi-invasive or invasive methods. It provides a strong layer of protection at the most vulnerable physical layer. As shown in *Figure 7-1*, eFuse is not a secure solution because it is easy to identify the programmed bits. Programming is accomplished by silicide electromigration of the fuse link between the anode and the cathode. While there is no visible difference between the programmed and non-programmed antifuse bits. The antifuse bit cell does not store a charge, whether antifuse bit is programmed is determined through current, not voltage.



(a) eFuse bit (b) Antifuse bit

Figure 7-1 A cross-section comparison of eFuse bit and antifuse OTP bit

The chip contains up to 2K bytes of one-time-programmable antifuse used for part configuration, key storage, and various other uses. The OTP contains factory configuration data such as on-chip SWR calibration values (Realtek factory), Wi-Fi calibration values (user factory), among other things. It may also be used by customer applications to configure some details of device operation, secure boot public key, aspects of device security, debug options, and boot options.

- OTP device adopts UMC 22nm single poly baseline ULP process.
  Neo Fuse is an antifuse-based technology.
  - Icell current >I threshold: programmed bit =0
  - Icell current < I threshold: un-programmed bit=1
- One-bit-two-cell architecture
  - Redundancy cell can reduce access failed rate

7.2 OTP Layout and Security Control

There are 16K bits physical OTP in RTL8721Dx. The OTP layout is summarized in *Figure 7-2*.

The OTP is divided into four zones:

- Logical zone: This zone contains system data for boot configuration, Wi-Fi & BT parameters, and calibration data. These data are actually programed in a Realtek-defined format in physical OTP. So even the physical data can only be programmed once, the logical data can be overwritten after format conversion.
- Security zone: The security-related keys, key access controls, and user-defined area with TrustZone attribute are all located in this zone.
- User-defined zone: Users can use this zone freely.
- Hidden physical zone: This zone contains the Return Material Authorization (RMA) control, RMA keys and Realtek calibration data.

All information provided in this document is subject to legal disclaimers.
© REALTEK 2024. All rights reserved.

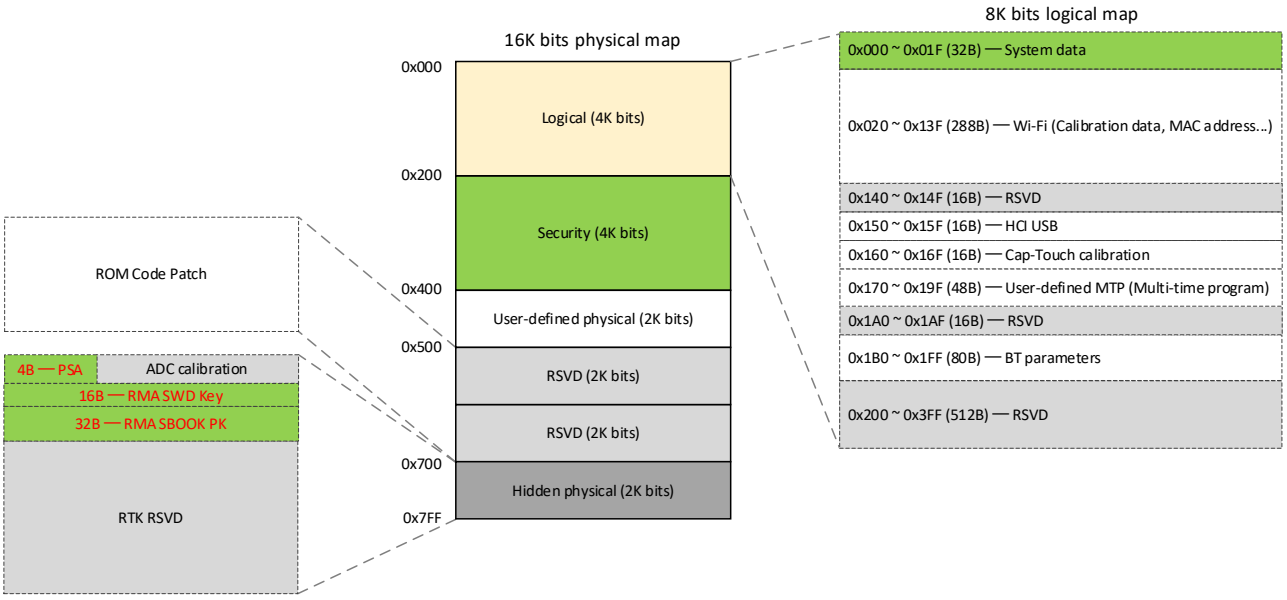


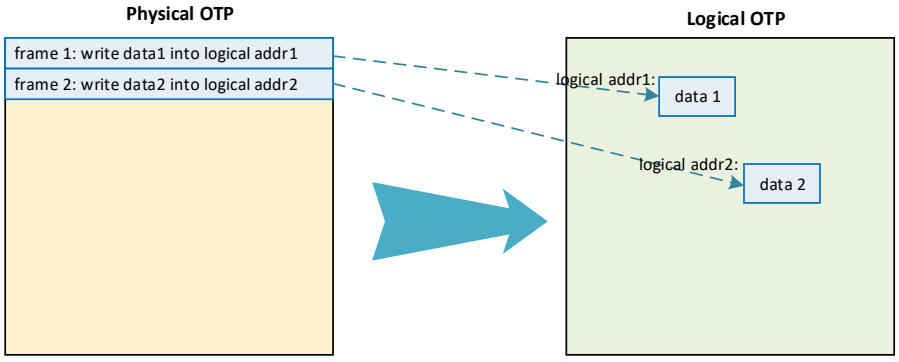
Figure 7-2 OTP layout

7.2.1 Logical Zone

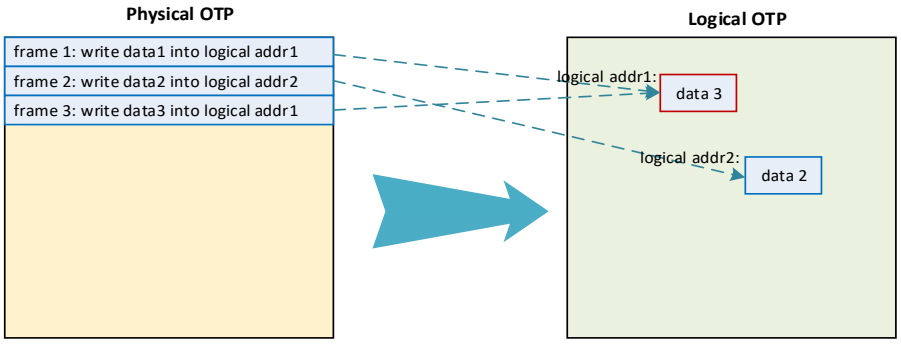
7.2.1.1 Mapping Relationship of Physical OTP and Logical OTP

The following figure is an example to illustrate mapping relationship.

(1) After programming frame1 and frame2, the value of logical OTP addr1 is data1, and value of addr2 is data2.

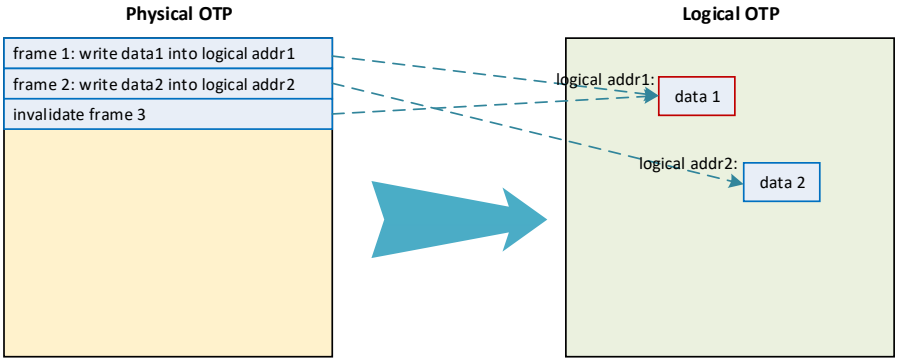


(2) After program frame3 in physical OTP, the value of logical OTP addr1 is data3 now, which has been overwritten and changed from data1 to data3.



(3) After changing frame3 in physical OTP, fram3 is invalidate. Then operation of frame3 is doing nothing. So the value of logical OTP addr1 has changed back to data1.

All information provided in this document is subject to legal disclaimers.
© REALTEK 2024. All rights reserved.



7.2.1.2 Logic Zone Layout

As *Figure 7-2* shows, the low address 0x000 ~ 0x200 of physical OTP is used for the logical area. Hardware and software will only parse this area for logical settings.

- The logical area 0x000 ~ 0x01F needs to be auto-loaded by hardware when the system boots.
- The logical area 0x020~0x3FF is used by software. Software will parse this area using a Realtek-defined OTP format.

This area needs to be programmed in word (16bits). The default value for the auto-loaded register is 0x00. e.g. If you want only to disable Boot log (0x3[0]), both 0x2 and 0x3 need to be programmed, and 0x2 should be programmed with default value 0x00, 0x3 should be programmed with the value 0x01.

Table 7-1 Logical zone layout

Start address	End address	Size (bytes)	Description
0x000	0x01F	32	System data to be auto-loaded to the system registers
0x020	0x13F	288	Wi-Fi calibration data and MAC address
0x140	0x14F	16	Reserved
0x150	0x15F	16	HCI USB
0x160	0x16F	16	Cap-Touch calibration
0x170	0x19F	48	User-defined MTP (Multi-Time Program)
0x1A0	0x1AF	16	Reserved
0x1B0	0x1FF	80	Bluetooth parameters
0x200	0x3FF	512	Reserved

7.2.1.2.1 System Data

Offset	Bit	Symbol	Description	HW address	INI
0x0	[7:0]	OTP_CHIPID0	These two bytes contain the chip ID if need	0x41000100	0
0x1	[7:0]	OTP_CHIPID1		0x41000101	0
0x2	[0]	FLASH_DEEP_SLEEP_EN	Flash deep-sleep enable	0x41000102	0
	[1]	SPIC_ADDR_4BYTE_EN	SPI Flash controller address 4-byte enable		0
	[2]	SPIC_BOOT_SPEEDUP_DIS	SPI Flash controller boot speed up disable		0
	[3]	BOOT_FLASH_BAUD_SEL	Boot Flash clock selection. Baudrate = 40/(2*(1+x)) 0: 20M 1: 10M		0
	[4]	BOOT_SELECT	Boot selection. Boot from internal Flash or external Flash. 0: Boot from internal Flash 1: Boot from external Flash		0
	[5]	BOOT_CNT_LMT_EN	Enable boot count limit 0: Disable 1: Enable		
	[7:6]	SW_RSVD	Reserved for software		0
0x3	[0]	DIS_BOOT_LOG_EN	Boot log disable	0x41000103	0
	[1]	LOW_BAUD_LOG_EN	LOGUART baud rate selection 1: 115200bps		0

All information provided in this document is subject to legal disclaimers.
© REALTEK 2024. All rights reserved.

			0: 1.5Mbps		
	[2]	LOGIC_SECURE_BOOT_EN	Secure boot enable		0
	[3]	LOGIC_RSIP_EN	RSIP enable		0
	[4]	LOGIC_RDP_EN	RDP enable		0
	[5]	BOOT_RAND_DELAY_EN	Enable random boot delay 0: Disable 1: Enable		0
	[6:7]	SW_RSVD	Reserved for software		0
0x4	[7:0]	SW_RSVD	Reserved for software	0x41000104	0
0x5	[7:0]	SW_RSVD	Reserved for software	0x41000105	0
0x6	[7:0]	SW_RSVD	Reserved for software	0x41000106	0
0x7	[7:0]	SW_RSVD	Reserved for software	0x41000107	0
0x8	[7:0]	SW_RSVD	Reserved for software	0x41000108	0
0x9	[7:0]	SW_RSVD	Reserved for software	0x41000109	0
0xA	[7:0]	SW_RSVD	Reserved for software	0x4100010A	0
0xB	[7:0]	SW_RSVD	Reserved for software	0x4100010B	0
0xC	[7:0]	SW_RSVD	Reserved for software	0x4100010C	0
0xD	[7:0]	SW_RSVD	Reserved for software	0x4100010D	0
0xE	[7:0]	SW_RSVD	Reserved for software	0x4100010E	0
0xF	[7:0]	SW_RSVD	Reserved for software	0x4100010F	0
0x10 ~ 0x1F		RSVD	Reserved		

7.2.1.2.2 Wi-Fi Calibration Data

Offset	Name	Bit	INI	Description
0x20	Path A/S0 2.4G CCK TSSI offset (Obtained from MP tool while performing power calibration)	[7:0]	00h	Bit[7]: Sign-bit for TSSI offset (0 for plus and 1 for minus) Bit[6:0]: Path A 2.4G CCK TSSI offset for channel 1/2. The calculation is 2's complement. Ex: FF=-1, 1=+1
0x21		[7:0]	00h	Bit[7]: Sign-bit for TSSI offset (0 for plus and 1 for minus) Bit[6:0]: Path A 2.4G CCK TSSI offset for channel 3/4/5. The calculation is 2's complement. Ex: FF=-1, 1=+1
0x22		[7:0]	00h	Bit[7]: Sign-bit for TSSI offset (0 for plus and 1 for minus) Bit[6:0]: Path A 2.4G CCK TSSI offset for channel 6/7/8. The calculation is 2's complement. Ex: FF=-1, 1=+1
0x23		[7:0]	00h	Bit[7]: Sign-bit for TSSI offset (0 for plus and 1 for minus) Bit[6:0]: Path A 2.4G CCK TSSI offset for channel 9/10/11. The calculation is 2's complement. Ex: FF=-1, 1=+1
0x24		[7:0]	00h	Bit[7]: Sign-bit for TSSI offset (0 for plus and 1 for minus) Bit[6:0]: Path A 2.4G CCK TSSI offset for channel 12/13. The calculation is 2's complement. Ex: FF=-1, 1=+1
0x25		[7:0]	00h	Bit[7]: Sign-bit for TSSI offset (0 for plus and 1 for minus) Bit[6:0]: Path A 2.4G CCK TSSI offset for channel 14. The calculation is 2's complement. Ex: FF=-1, 1=+1
0x26	Path A/S0 2.4G BW20-1S TSSI offset (Obtained from MP tool while performing power calibration)	[7:0]	00h	Bit[7]: Sign-bit for TSSI offset (0 for plus and 1 for minus) Bit[6:0]: Path A 2.4G BW20-1S TSSI offset for channel 1/2. The calculation is 2's complement. Ex: FF=-1, 1=+1
0x27		[7:0]	00h	Bit[7]: Sign-bit for TSSI offset (0 for plus and 1 for minus) Bit[6:0]: Path A 2.4G BW20-1S TSSI offset for channel 3/4/5. The calculation is 2's complement. Ex: FF=-1, 1=+1
0x28		[7:0]	00h	Bit[7]: Sign-bit for TSSI offset (0 for plus and 1 for minus) Bit[6:0]: Path A 2.4G BW20-1S TSSI offset for channel 6/7/8. The calculation is 2's complement. Ex: FF=-1, 1=+1
0x29		[7:0]	00h	Bit[7]: Sign-bit for TSSI offset (0 for plus and 1 for minus) Bit[6:0]: Path A 2.4G BW20-1S TSSI offset for channel 9/10/11. The calculation is 2's complement. Ex: FF=-1, 1=+1
0x2A		[7:0]	00h	Bit[7]: Sign-bit for TSSI offset (0 for plus and 1 for minus) Bit[6:0]: Path A 2.4G BW20-1S TSSI offset for channel 12/13/14. The calculation is 2's complement. Ex: FF=-1, 1=+1
0x2B	RSVD	[7:0]		Reserved

All information provided in this document is subject to legal disclaimers.
© REALTEK 2024. All rights reserved.

0x2C	RSVD	[7:0]		Reserved
0x2D	RSVD	[7:0]		Reserved
0x2E	RSVD	[7:0]		Reserved
0x2F	RSVD	[7:0]		Reserved
0x30	RSVD	[7:0]		Reserved
0x31	RSVD	[7:0]		Reserved
0x32	Path A/S0 5G BW40-1S TSSI offset (Obtained from MP tool while performing power calibration)	[7:0]	00h	Bit[7]: Sign-bit for TSSI offset (0 for plus and 1 for minus) Bit[6:0]: Path A 5G BW40-1S TSSI offset for channel 36/38/40. The calculation is 2's complement. Ex: FF=-1, 1=+1
0x33		[7:0]	00h	Bit[7]: Sign-bit for TSSI offset (0 for plus and 1 for minus) Bit[6:0]: Path A 5G BW40-1S TSSI offset for channel 44/46/48. The calculation is 2's complement. Ex: FF=-1, 1=+1
0x34		[7:0]	00h	Bit[7]: Sign-bit for TSSI offset (0 for plus and 1 for minus) Bit[6:0]: Path A 5G BW40-1S TSSI offset for channel 52/54/56. The calculation is 2's complement. Ex: FF=-1, 1=+1
0x35		[7:0]	00h	Bit[7]: Sign-bit for TSSI offset (0 for plus and 1 for minus) Bit[6:0]: Path A 5G BW40-1S TSSI offset for channel 60/62/64. The calculation is 2's complement. Ex: FF=-1, 1=+1
0x36		[7:0]	00h	Bit[7]: Sign-bit for TSSI offset (0 for plus and 1 for minus) Bit[6:0]: Path A 5G BW40-1S TSSI offset for channel 100/102/104. The calculation is 2's complement. Ex: FF=-1, 1=+1
0x37		[7:0]	00h	Bit[7]: Sign-bit for TSSI offset (0 for plus and 1 for minus) Bit[6:0]: Path A 5G BW40-1S TSSI offset for channel 108/110/112. The calculation is 2's complement. Ex: FF=-1, 1=+1
0x38		[7:0]	00h	Bit[7]: Sign-bit for TSSI offset (0 for plus and 1 for minus) Bit[6:0]: Path A 5G BW40-1S TSSI offset for channel 116/118/120. The calculation is 2's complement. Ex: FF=-1, 1=+1
0x39		[7:0]	00h	Bit[7]: Sign-bit for TSSI offset (0 for plus and 1 for minus) Bit[6:0]: Path A 5G BW40-1S TSSI offset for channel 124/126/128. The calculation is 2's complement. Ex: FF=-1, 1=+1
0x3A		[7:0]	00h	Bit[7]: Sign-bit for TSSI offset (0 for plus and 1 for minus) Bit[6:0]: Path A 5G BW40-1S TSSI offset for channel 132/134/136. The calculation is 2's complement. Ex: FF=-1, 1=+1
0x3B		[7:0]	00h	Bit[7]: Sign-bit for TSSI offset (0 for plus and 1 for minus) Bit[6:0]: Path A 5G BW40-1S TSSI offset for channel 140/142/144. The calculation is 2's complement. Ex: FF=-1, 1=+1
0x3C		[7:0]	00h	Bit[7]: Sign-bit for TSSI offset (0 for plus and 1 for minus) Bit[6:0]: Path A 5G BW40-1S TSSI offset for channel 149/151/153. The calculation is 2's complement. Ex: FF=-1, 1=+1
0x3D		[7:0]	00h	Bit[7]: Sign-bit for TSSI offset (0 for plus and 1 for minus) Bit[6:0]: Path A 5G BW40-1S TSSI offset for channel 157/159/161. The calculation is 2's complement. Ex: FF=-1, 1=+1
0x3E		[7:0]	00h	Bit[7]: Sign-bit for TSSI offset (0 for plus and 1 for minus) Bit[6:0]: Path A 5G BW40-1S TSSI offset for channel 165/167/169. The calculation is 2's complement. Ex: FF=-1, 1=+1
0x3F		[7:0]	00h	Bit[7]: Sign-bit for TSSI offset (0 for plus and 1 for minus) Bit[6:0]: Path A 5G BW40-1S TSSI offset for channel 173/175/177. The calculation is 2's complement. Ex: FF=-1, 1=+1
0x40	RSVD	[7:0]		Reserved
0x41	RSVD	[7:0]		Reserved
0x42	RSVD	[7:0]		Reserved
0x43	RSVD	[7:0]		Reserved
0x44	RSVD	[7:0]		Reserved
0x45	RSVD	[7:0]		Reserved
0x46	RSVD	[7:0]		Reserved
0x47	RSVD	[7:0]		Reserved
0x48	RSVD	[7:0]		Reserved
0x49	RSVD	[7:0]		Reserved
0x4A	Path B/S1 2.4G CCK TSSI offset (Obtained from MP tool while performing power calibration)	[7:0]		Bit[7]: Sign-bit for TSSI offset (0 for plus and 1 for minus) Bit[6:0]: Path B 2.4G CCK TSSI offset for channel 1/2. The calculation is 2's complement. Ex: FF=-1, 1=+1

All information provided in this document is subject to legal disclaimers.
© REALTEK 2024. All rights reserved.

0x4B		[7:0]		Bit[7]: Sign-bit for TSSI offset (0 for plus and 1 for minus) Bit[6:0]: Path B 2.4G CCK TSSI offset for channel 3/4/5. The calculation is 2's complement. Ex: FF=-1, 1=+1
0x4C		[7:0]		Bit[7]: Sign-bit for TSSI offset (0 for plus and 1 for minus) Bit[6:0]: Path B 2.4G CCK TSSI offset for channel 6/7/8. The calculation is 2's complement. Ex: FF=-1, 1=+1
0x4D		[7:0]		Bit[7]: Sign-bit for TSSI offset (0 for plus and 1 for minus) Bit[6:0]: Path B 2.4G CCK TSSI offset for channel 9/10/11. The calculation is 2's complement. Ex: FF=-1, 1=+1
0x4E		[7:0]		Bit[7]: Sign-bit for TSSI offset (0 for plus and 1 for minus) Bit[6:0]: Path B 2.4G CCK TSSI offset for channel 12/13. The calculation is 2's complement. Ex: FF=-1, 1=+1
0x4F		[7:0]		Bit[7]: Sign-bit for TSSI offset (0 for plus and 1 for minus) Bit[6:0]: Path B 2.4G CCK TSSI offset for channel 14. The calculation is 2's complement. Ex: FF=-1, 1=+1
0x50	Path B/S1 2.4G BW40-1S TSSI offset (Obtained from MP tool while performing power calibration)	[7:0]		Bit[7]: Sign-bit for TSSI offset (0 for plus and 1 for minus) Bit[6:0]: Path B 2.4G BW40-1S TSSI offset for channel 1/2. The calculation is 2's complement. Ex: FF=-1, 1=+1
0x51		[7:0]		Bit[7]: Sign-bit for TSSI offset (0 for plus and 1 for minus) Bit[6:0]: Path B 2.4G BW40-1S TSSI offset for channel 3/4/5. The calculation is 2's complement. Ex: FF=-1, 1=+1
0x52		[7:0]		Bit[7]: Sign-bit for TSSI offset (0 for plus and 1 for minus) Bit[6:0]: Path B 2.4G BW40-1S TSSI offset for channel 6/7/8. The calculation is 2's complement. Ex: FF=-1, 1=+1
0x53		[7:0]		Bit[7]: Sign-bit for TSSI offset (0 for plus and 1 for minus) Bit[6:0]: Path B 2.4G BW40-1S TSSI offset for channel 9/10/11. The calculation is 2's complement. Ex: FF=-1, 1=+1
0x54		[7:0]		Bit[7]: Sign-bit for TSSI offset (0 for plus and 1 for minus) Bit[6:0]: Path B 2.4G BW40-1S TSSI offset for channel 12/13/14. The calculation is 2's complement. Ex: FF=-1, 1=+1
0x55	RSVD	[7:0]		Reserved
0x56	RSVD	[7:0]		Reserved
0x57	RSVD	[7:0]		Reserved
0x58	RSVD	[7:0]		Reserved
0x59	RSVD	[7:0]		Reserved
0x5A	RSVD	[7:0]		Reserved
0x5B	RSVD	[7:0]		Reserved
0x5C	Path B 5G BW40-1S TSSI offset (Obtained from MP tool while performing power calibration)	[7:0]		Bit[7]: Sign-bit for TSSI offset (0 for plus and 1 for minus) Bit[6:0]: Path B 5G BW40-1S TSSI offset for channel 36/38/40. The calculation is 2's complement. Ex: FF=-1, 1=+1
0x5D		[7:0]		Bit[7]: Sign-bit for TSSI offset (0 for plus and 1 for minus) Bit[6:0]: Path B 5G BW40-1S TSSI offset for channel 44/46/48. The calculation is 2's complement. Ex: FF=-1, 1=+1
0x5E		[7:0]		Bit[7]: Sign-bit for TSSI offset (0 for plus and 1 for minus) Bit[6:0]: Path B 5G BW40-1S TSSI offset for channel 52/54/56. The calculation is 2's complement. Ex: FF=-1, 1=+1
0x5F		[7:0]		Bit[7]: Sign-bit for TSSI offset (0 for plus and 1 for minus) Bit[6:0]: Path B 5G BW40-1S TSSI offset for channel 60/62/64. The calculation is 2's complement. Ex: FF=-1, 1=+1
0x60		[7:0]		Bit[7]: Sign-bit for TSSI offset (0 for plus and 1 for minus) Bit[6:0]: Path B 5G BW40-1S TSSI offset for channel 100/102/104. The calculation is 2's complement. Ex: FF=-1, 1=+1
0x61		[7:0]		Bit[7]: Sign-bit for TSSI offset (0 for plus and 1 for minus) Bit[6:0]: Path B 5G BW40-1S TSSI offset for channel 108/110/112. The calculation is 2's complement. Ex: FF=-1, 1=+1
0x62		[7:0]		Bit[7]: Sign-bit for TSSI offset (0 for plus and 1 for minus) Bit[6:0]: Path B 5G BW40-1S TSSI offset for channel 116/118/120. The calculation is 2's complement. Ex: FF=-1, 1=+1
0x63		[7:0]		Bit[7]: Sign-bit for TSSI offset (0 for plus and 1 for minus) Bit[6:0]: Path B 5G BW40-1S TSSI offset for channel 124/126/128. The calculation is 2's complement. Ex: FF=-1, 1=+1

All information provided in this document is subject to legal disclaimers.
© REALTEK 2024. All rights reserved.

0x64		[7:0]		Bit[7]: Sign-bit for TSSI offset (0 for plus and 1 for minus) Bit[6:0]: Path B 5G BW40-1S TSSI offset for channel 132/134/136. The calculation is 2's complement. Ex: FF=-1, 1=+1
0x65		[7:0]		Bit[7]: Sign-bit for TSSI offset (0 for plus and 1 for minus) Bit[6:0]: Path B 5G BW40-1S TSSI offset for channel 140/142/144. The calculation is 2's complement. Ex: FF=-1, 1=+1
0x66		[7:0]		Bit[7]: Sign-bit for TSSI offset (0 for plus and 1 for minus) Bit[6:0]: Path B 5G BW40-1S TSSI offset for channel 149/151/153. The calculation is 2's complement. Ex: FF=-1, 1=+1
0x67		[7:0]		Bit[7]: Sign-bit for TSSI offset (0 for plus and 1 for minus) Bit[6:0]: Path B 5G BW40-1S TSSI offset for channel 157/159/161. The calculation is 2's complement. Ex: FF=-1, 1=+1
0x68		[7:0]		Bit[7]: Sign-bit for TSSI offset (0 for plus and 1 for minus) Bit[6:0]: Path B 5G BW40-1S TSSI offset for channel 165/167/169. The calculation is 2's complement. Ex: FF=-1, 1=+1
0x69		[7:0]		Bit[7]: Sign-bit for TSSI offset (0 for plus and 1 for minus) Bit[6:0]: Path B 5G BW40-1S TSSI offset for channel 173/175/177. The calculation is 2's complement. Ex: FF=-1, 1=+1
0x6A	RSVD	[7:0]		Reserved
0x6B	RSVD	[7:0]		Reserved
0x6C	RSVD	[7:0]		Reserved
0x6D	RSVD	[7:0]		Reserved
0x6E	RSVD	[7:0]		Reserved
0x6F	RSVD	[7:0]		Reserved
0x70	RSVD	[7:0]		Reserved
0x71	RSVD	[7:0]		Reserved
0x72	RSVD	[7:0]		Reserved
0x73	RSVD	[7:0]		Reserved
0x74	Path C 2.4G CCK TSSI offset (Obtained from MP tool while performing power calibration)	[7:0]	00h	Bit[7]: Sign-bit for TSSI offset (0 for plus and 1 for minus) Bit[6:0]: Path C 2.4G CCK TSSI offset for channel 1/2. The calculation is 2's complement. Ex: FF=-1, 1=+1
0x75		[7:0]	00h	Bit[7]: Sign-bit for TSSI offset (0 for plus and 1 for minus) Bit[6:0]: Path C 2.4G CCK TSSI offset for channel 3/4/5. The calculation is 2's complement. Ex: FF=-1, 1=+1
0x76		[7:0]	00h	Bit[7]: Sign-bit for TSSI offset (0 for plus and 1 for minus) Bit[6:0]: Path C 2.4G CCK TSSI offset for channel 6/7/8. The calculation is 2's complement. Ex: FF=-1, 1=+1
0x77		[7:0]	00h	Bit[7]: Sign-bit for TSSI offset (0 for plus and 1 for minus) Bit[6:0]: Path C 2.4G CCK TSSI offset for channel 9/10/11. The calculation is 2's complement. Ex: FF=-1, 1=+1
0x78		[7:0]	00h	Bit[7]: Sign-bit for TSSI offset (0 for plus and 1 for minus) Bit[6:0]: Path C 2.4G CCK TSSI offset for channel 12/13. The calculation is 2's complement. Ex: FF=-1, 1=+1
0x79		[7:0]	00h	Bit[7]: Sign-bit for TSSI offset (0 for plus and 1 for minus) Bit[6:0]: Path C 2.4G CCK TSSI offset for channel 14. The calculation is 2's complement. Ex: FF=-1, 1=+1
0x7A	Path C 2.4G BW40-1S TSSI offset (Obtained from MP tool while performing power calibration)	[7:0]		Bit[7]: Sign-bit for TSSI offset (0 for plus and 1 for minus) Bit[6:0]: Path C 2.4G BW40-1S TSSI offset for channel 1/2. The calculation is 2's complement. Ex: FF=-1, 1=+1
0x7B		[7:0]		Bit[7]: Sign-bit for TSSI offset (0 for plus and 1 for minus) Bit[6:0]: Path C 2.4G BW40-1S TSSI offset for channel 3/4/5. The calculation is 2's complement. Ex: FF=-1, 1=+1
0x7C		[7:0]		Bit[7]: Sign-bit for TSSI offset (0 for plus and 1 for minus) Bit[6:0]: Path C 2.4G BW40-1S TSSI offset for channel 6/7/8. The calculation is 2's complement. Ex: FF=-1, 1=+1
0x7D		[7:0]		Bit[7]: Sign-bit for TSSI offset (0 for plus and 1 for minus) Bit[6:0]: Path C 2.4G BW40-1S TSSI offset for channel 9/10/11. The calculation is 2's complement. Ex: FF=-1, 1=+1
0x7E		[7:0]		Bit[7]: Sign-bit for TSSI offset (0 for plus and 1 for minus) Bit[6:0]: Path C 2.4G BW40-1S TSSI offset for channel 12/13/14. The calculation is 2's complement. Ex: FF=-1, 1=+1

All information provided in this document is subject to legal disclaimers.
© REALTEK 2024. All rights reserved.

0x7F	RSVD	[7:0]		Reserved
0x80	RSVD	[7:0]		Reserved
0x81	RSVD	[7:0]		Reserved
0x82	RSVD	[7:0]		Reserved
0x83	RSVD	[7:0]		Reserved
0x84	RSVD	[7:0]		Reserved
0x85	RSVD	[7:0]		Reserved
0x86	Path C 5G BW40-1S TSSI offset (Obtained from MP tool while performing power calibration)	[7:0]		Bit[7]: Sign-bit for TSSI offset (0 for plus and 1 for minus) Bit[6:0]: Path C 5G BW40-1S TSSI offset for channel 36/38/40. The calculation is 2's complement. Ex: FF=-1, 1=+1
0x87		[7:0]		Bit[7]: Sign-bit for TSSI offset (0 for plus and 1 for minus) Bit[6:0]: Path C 5G BW40-1S TSSI offset for channel 44/46/48. The calculation is 2's complement. Ex: FF=-1, 1=+1
0x88		[7:0]		Bit[7]: Sign-bit for TSSI offset (0 for plus and 1 for minus) Bit[6:0]: Path C 5G BW40-1S TSSI offset for channel 52/54/56. The calculation is 2's complement. Ex: FF=-1, 1=+1
0x89		[7:0]		Bit[7]: Sign-bit for TSSI offset (0 for plus and 1 for minus) Bit[6:0]: Path C 5G BW40-1S TSSI offset for channel 60/62/64. The calculation is 2's complement. Ex: FF=-1, 1=+1
0x8A		[7:0]		Bit[7]: Sign-bit for TSSI offset (0 for plus and 1 for minus) Bit[6:0]: Path C 5G BW40-1S TSSI offset for channel 100/102/104. The calculation is 2's complement. Ex: FF=-1, 1=+1
0x8B		[7:0]		Bit[7]: Sign-bit for TSSI offset (0 for plus and 1 for minus) Bit[6:0]: Path C 5G BW40-1S TSSI offset for channel 108/110/112. The calculation is 2's complement. Ex: FF=-1, 1=+1
0x8C		[7:0]		Bit[7]: Sign-bit for TSSI offset (0 for plus and 1 for minus) Bit[6:0]: Path C 5G BW40-1S TSSI offset for channel 116/118/120. The calculation is 2's complement. Ex: FF=-1, 1=+1
0x8D		[7:0]		Bit[7]: Sign-bit for TSSI offset (0 for plus and 1 for minus) Bit[6:0]: Path C 5G BW40-1S TSSI offset for channel 124/126/128. The calculation is 2's complement. Ex: FF=-1, 1=+1
0x8E		[7:0]		Bit[7]: Sign-bit for TSSI offset (0 for plus and 1 for minus) Bit[6:0]: Path C 5G BW40-1S TSSI offset for channel 132/134/136. The calculation is 2's complement. Ex: FF=-1, 1=+1
0x8F		[7:0]		Bit[7]: Sign-bit for TSSI offset (0 for plus and 1 for minus) Bit[6:0]: Path C 5G BW40-1S TSSI offset for channel 140/142/144. The calculation is 2's complement. Ex: FF=-1, 1=+1
0x90		[7:0]		Bit[7]: Sign-bit for TSSI offset (0 for plus and 1 for minus) Bit[6:0]: Path C 5G BW40-1S TSSI offset for channel 149/151/153. The calculation is 2's complement. Ex: FF=-1, 1=+1
0x91		[7:0]		Bit[7]: Sign-bit for TSSI offset (0 for plus and 1 for minus) Bit[6:0]: Path C 5G BW40-1S TSSI offset for channel 157/159/161. The calculation is 2's complement. Ex: FF=-1, 1=+1
0x92		[7:0]		Bit[7]: Sign-bit for TSSI offset (0 for plus and 1 for minus) Bit[6:0]: Path C 5G BW40-1S TSSI offset for channel 165/167/169. The calculation is 2's complement. Ex: FF=-1, 1=+1
0x93		[7:0]		Bit[7]: Sign-bit for TSSI offset (0 for plus and 1 for minus) Bit[6:0]: Path C 5G BW40-1S TSSI offset for channel 173/175/177. The calculation is 2's complement. Ex: FF=-1, 1=+1
0x94	RSVD	[7:0]		Reserved
0x95	RSVD	[7:0]		Reserved
0x96	RSVD	[7:0]		Reserved
0x97	RSVD	[7:0]		Reserved
0x98	RSVD	[7:0]		Reserved
0x99	RSVD	[7:0]		Reserved
0x9A	RSVD	[7:0]		Reserved
0x9B	RSVD	[7:0]		Reserved
0x9C	RSVD	[7:0]		Reserved
0x9D	RSVD	[7:0]		Reserved
0x9E	Path D 2.4G CCK TSSI offset (Obtained from MP tool while performing	[7:0]		Bit[7]: Sign-bit for TSSI offset (0 for plus and 1 for minus) Bit[6:0]: Path D 2.4G CCK TSSI offset for channel 1/2.

All information provided in this document is subject to legal disclaimers.
© REALTEK 2024. All rights reserved.

	power calibration)			The calculation is 2's complement. Ex: FF=-1, 1=+1
0x9F		[7:0]		Bit[7]: Sign-bit for TSSI offset (0 for plus and 1 for minus) Bit[6:0]: Path D 2.4G CCK TSSI offset for channel 3/4/5. The calculation is 2's complement. Ex: FF=-1, 1=+1
0xA0		[7:0]		Bit[7]: Sign-bit for TSSI offset (0 for plus and 1 for minus) Bit[6:0]: Path D 2.4G CCK TSSI offset for channel 6/7/8. The calculation is 2's complement. Ex: FF=-1, 1=+1
0xA1		[7:0]		Bit[7]: Sign-bit for TSSI offset (0 for plus and 1 for minus) Bit[6:0]: Path D 2.4G CCK TSSI offset for channel 9/10/11. The calculation is 2's complement. Ex: FF=-1, 1=+1
0xA2		[7:0]		Bit[7]: Sign-bit for TSSI offset (0 for plus and 1 for minus) Bit[6:0]: Path D 2.4G CCK TSSI offset for channel 12/13. The calculation is 2's complement. Ex: FF=-1, 1=+1
0xA3		[7:0]		Bit[7]: Sign-bit for TSSI offset (0 for plus and 1 for minus) Bit[6:0]: Path D 2.4G CCK TSSI offset for channel 14. The calculation is 2's complement. Ex: FF=-1, 1=+1
0xA4	Path D 2.4G BW40-1S TSSI offset (Obtained from MP tool while performing power calibration)	[7:0]		Bit[7]: Sign-bit for TSSI offset (0 for plus and 1 for minus) Bit[6:0]: Path D 2.4G BW40-1S TSSI offset for channel 1/2. The calculation is 2's complement. Ex: FF=-1, 1=+1
0xA5		[7:0]		Bit[7]: Sign-bit for TSSI offset (0 for plus and 1 for minus) Bit[6:0]: Path D 2.4G BW40-1S TSSI offset for channel 3/4/5. The calculation is 2's complement. Ex: FF=-1, 1=+1
0xA6		[7:0]		Bit[7]: Sign-bit for TSSI offset (0 for plus and 1 for minus) Bit[6:0]: Path D 2.4G BW40-1S TSSI offset for channel 6/7/8. The calculation is 2's complement. Ex: FF=-1, 1=+1
0xA7		[7:0]		Bit[7]: Sign-bit for TSSI offset (0 for plus and 1 for minus) Bit[6:0]: Path D 2.4G BW40-1S TSSI offset for channel 9/10/11. The calculation is 2's complement. Ex: FF=-1, 1=+1
0xA8		[7:0]		Bit[7]: Sign-bit for TSSI offset (0 for plus and 1 for minus) Bit[6:0]: Path D 2.4G BW40-1S TSSI offset for channel 12/13/14. The calculation is 2's complement. Ex: FF=-1, 1=+1
0xA9	RSVD	[7:0]		Reserved
0xAA	RSVD	[7:0]		Reserved
0xAB	RSVD	[7:0]		Reserved
0xAC	RSVD	[7:0]		Reserved
0xAD	RSVD	[7:0]		Reserved
0xAE	RSVD	[7:0]		Reserved
0xAF	RSVD	[7:0]		Reserved
0xB0	Path D 5G BW40-1S TSSI offset (Obtained from MP tool while performing power calibration)	[7:0]		Bit[7]: Sign-bit for TSSI offset (0 for plus and 1 for minus) Bit[6:0]: Path D 5G BW40-1S TSSI offset for channel 36/38/40. The calculation is 2's complement. Ex: FF=-1, 1=+1
0xB1		[7:0]		Bit[7]: Sign-bit for TSSI offset (0 for plus and 1 for minus) Bit[6:0]: Path D 5G BW40-1S TSSI offset for channel 44/46/48. The calculation is 2's complement. Ex: FF=-1, 1=+1
0xB2		[7:0]		Bit[7]: Sign-bit for TSSI offset (0 for plus and 1 for minus) Bit[6:0]: Path D 5G BW40-1S TSSI offset for channel 52/54/56. The calculation is 2's complement. Ex: FF=-1, 1=+1
0xB3		[7:0]		Bit[7]: Sign-bit for TSSI offset (0 for plus and 1 for minus) Bit[6:0]: Path D 5G BW40-1S TSSI offset for channel 60/62/64. The calculation is 2's complement. Ex: FF=-1, 1=+1
0xB4		[7:0]		Bit[7]: Sign-bit for TSSI offset (0 for plus and 1 for minus) Bit[6:0]: Path D 5G BW40-1S TSSI offset for channel 100/102/104. The calculation is 2's complement. Ex: FF=-1, 1=+1
0xB5		[7:0]		Bit[7]: Sign-bit for TSSI offset (0 for plus and 1 for minus) Bit[6:0]: Path D 5G BW40-1S TSSI offset for channel 108/110/112. The calculation is 2's complement. Ex: FF=-1, 1=+1
0xB6		[7:0]		Bit[7]: Sign-bit for TSSI offset (0 for plus and 1 for minus) Bit[6:0]: Path D 5G BW40-1S TSSI offset for channel 116/118/120. The calculation is 2's complement. Ex: FF=-1, 1=+1
0xB7		[7:0]		Bit[7]: Sign-bit for TSSI offset (0 for plus and 1 for minus) Bit[6:0]: Path D 5G BW40-1S TSSI offset for channel 124/126/128.

All information provided in this document is subject to legal disclaimers.
© REALTEK 2024. All rights reserved.

				The calculation is 2's complement. Ex: FF=-1, 1=+1
0xB8		[7:0]		Bit[7]: Sign-bit for TSSI offset (0 for plus and 1 for minus) Bit[6:0]: Path D 5G BW40-1S TSSI offset for channel 132/134/136. The calculation is 2's complement. Ex: FF=-1, 1=+1
0xB9		[7:0]		Bit[7]: Sign-bit for TSSI offset (0 for plus and 1 for minus) Bit[6:0]: Path D 5G BW40-1S TSSI offset for channel 140/142/144. The calculation is 2's complement. Ex: FF=-1, 1=+1
0xBA		[7:0]		Bit[7]: Sign-bit for TSSI offset (0 for plus and 1 for minus) Bit[6:0]: Path D 5G BW40-1S TSSI offset for channel 149/151/153. The calculation is 2's complement. Ex: FF=-1, 1=+1
0xBB		[7:0]		Bit[7]: Sign-bit for TSSI offset (0 for plus and 1 for minus) Bit[6:0]: Path D 5G BW40-1S TSSI offset for channel 157/159/161. The calculation is 2's complement. Ex: FF=-1, 1=+1
0xBC		[7:0]		Bit[7]: Sign-bit for TSSI offset (0 for plus and 1 for minus) Bit[6:0]: Path D 5G BW40-1S TSSI offset for channel 165/167/169. The calculation is 2's complement. Ex: FF=-1, 1=+1
0xBD		[7:0]		Bit[7]: Sign-bit for TSSI offset (0 for plus and 1 for minus) Bit[6:0]: Path D 5G BW40-1S TSSI offset for channel 173/175/177. The calculation is 2's complement. Ex: FF=-1, 1=+1
0xBE	RSVD	[7:0]		Reserved
0xBF	RSVD	[7:0]		Reserved
0xC0	RSVD	[7:0]		Reserved
0xC1	RSVD	[7:0]		Reserved
0xC2	RSVD	[7:0]		Reserved
0xC3	Channel Plan Page	[7:6]		Defined Channel Plan Page 0/1/2/3. 11: Page 0 10: Page 1 01: Page 2 00: Page 3
		[5:0]		Reserved
0xC4	RSVD	[7:0]		Reserved
0xC5	RSVD	[7:0]		Reserved
0xC6	RSVD	[7:0]		Reserved
0xC7	RSVD	[7:0]		Reserved
0xC8	Channel Plan	[7:0]	7Fh	Refer to the document with the topic “Channel Plan Domain Code”
0xC9	Crystal Calibration	[6:0]	3Fh	XTAL_K value Xi=Xo, Range: 0~7Fh. NOTE *If in the case without assigned value, 0x3F will be used by the driver.*
		[7]	0h	Reserved
0xCA	Thermal Meter	[7:0]	20h	Thermal Meter Value System maker will calibrate a value and save it in EEPROM. 0xFF: Disable Tx power tracking function
0xCB	IQ Calibration and LC Calibration	[1:0]	0h	Do IQK by Thermal Meter Δvalue. 0h: Mode 1 1h: Mode 2 2h: Mode 3 3h: Don’t IQK
		[3:2]	0h	Do LCK by Thermal Meter Δvalue. 0h: Mode 1 1h: Mode 2 2h: Mode 3 3h: Don’t LCK
		[7:4]	0h	Reserved
0xCC	2.4G and 5G PA Type	[7:0]		
0xCD	2.4G LNA Type and Gain Selection	[7:0]		
0xCE		[7:0]		
0xCF	5G LNA Type and Gain Selection	[7:0]		
0x130		[7:0]		

All information provided in this document is subject to legal disclaimers.
© REALTEK 2024. All rights reserved.

0x131	Board Options	[1:0]	1h	Regulatory selection 0h: Tx output power varies according to the power-by-rate table. (Power-limit table takes no effect on output power, non-regular use) 1h: Tx output power varies according to both the power-by-rate table and the power-limit table. (Power-by-rate table is generally provided by Realtek according to optimized through-put test, power-limit table is based on certification-pass criterion) (Default setting to satisfy both of the two purposed) 2h: Disable power-by-rate function.
		[2]	0h	Tx diversity 0: Disable or multi-Tx-path 1: Enable (de-featured to 1 Tx/stream)
		[3]	0h	Rx diversity 0: Disable or multi-Rx-path (Rx MRC) 1: Enable (de-featured to 1 Rx/stream)
		[4]	0h	Option for 11ac mode 0: Enable 11ac mode 1: Disable 11ac mode
		[7:5]	1h	Module type 0h: Wi-Fi solo module 1h: Wi-Fi/BT combo module 2~7h: Reserved
0x132	Feature Options	[1:0]	0h	Function configuration of pin_LED0 and pin_LED1
		[3:2]	0h	Link Speed shown in OS 0h: Current Tx PHY Rate 1h: Current Rx PHY Rate 2h: Maximum RX PHY Rate 3h: Reserved
		[4]	0h	WLAN disable mode selection 0: Radio off 1: Power down
		[7:5]	0h	Reserved
x133	BT Setting (From BT's point of view, not concerned with Wi-Fi)	[0]	0h	Total antenna number 0: BT/Wi-Fi use dedicated antenna 1: BT/Wi-Fi use shared antenna
		[3:1]	0h	Co-existence type 0h: Internal Mailbox 1h: Realtek BT I2C Mailbox (contact FAE for more information)
		[4]	1h	Antenna isolation 0: High isolation (>35dB) 1: Low
		[5]	0h	Radio on/off type 0: Combine with Wi-Fi 1: Individual
		[6]	0h	BT AFE 0: BT SEL WLAFE 1: BT SEL BTAFE
		[7]	0h	Reserved
0x134	Version	[7:0]	00h	EEPROM content version
0x135	Customer ID	[7:0]	00h	Customer ID NOTE *0x00 and 0xFF are reserved for Realtek.*
0x136	2.4G Tx BB Swing Setting	[1:0]	0h	2.4G Path A OFDM 0h: 0dB 1h: -3dB 2h: -6dB 3h: -9dB
		[3:2]	0h	2.4G Path B OFDM 0h: 0dB 1h: -3dB

All information provided in this document is subject to legal disclaimers.
© REALTEK 2024. All rights reserved.

				2h: -6dB 3h: -9dB
		[5:4]	0h	2.4G Path C OFDM 0h: 0dB 1h: -3dB 2h: -6dB 3h: -9dB
		[7:6]	0h	2.4G Path D OFDM 0h: 0dB 1h: -3dB 2h: -6dB 3h: -9dB
0x137	5G Tx BB Swing Setting	[1:0]	0h	5G Path A OFDM 0h: 0dB 1h: -3dB 2h: -6dB 3h: -9dB
		[3:2]	0h	5G Path B OFDM 0h: 0dB 1h: -3dB 2h: -6dB 3h: -9dB
		[5:4]	0h	5G Path C OFDM 0h: 0dB 1h: -3dB 2h: -6dB 3h: -9dB
		[7:6]	0h	5G Path D OFDM 0h: 0dB 1h: -3dB 2h: -6dB 3h: -9dB
0x138	Tx Power Calibration Rate & Tx Power Tracking Mode	[0]	0h	Power Calibration: 2.4G 40M Tx Power Calibrator Rate 0h: HT40, MCS7 64QAM 1h: VHT40, MCS9 256QAM
		[1]	0h	Power Calibration: 5G 40M Tx Power Calibrator Rate 0h: HT40, MCS7 64QAM 1h: VHT40, MCS9 256QAM
		[3:2]	0h	Reserved
		[7:4]	1h	Power Tracking 0x0h ~ 0x3h: Power tracking by thermal 0x4h ~ 0x7h: Power tracking by TSSI
0x139	Tx/Rx Path Selection	[3:0]	1h	Rx path assignment Bit[3]: Path S3 Bit[2]: Path S2 Bit[1]: Path S1 Bit[0]: Path S0 ● 1: On ● 0: Off NOTE *The ability for maximum Tx/Rx paths would be constrained by real physical limitation.*
		[7:4]	1h	Tx path assignment Bit[7]: Path S3 Bit[6]: Path S2 Bit[5]: Path S1 Bit[4]: Path S0 ● 1: On ● 0: Off
0x13A	RFE Type (RF Front-end Type) (Concerned with hardware parameter)	[6:0]	1h	RFE Type Contact FAE to get the right RFE type for individual hardware

All information provided in this document is subject to legal disclaimers.
© REALTEK 2024. All rights reserved.

				design.
		[7]	0h	Determined RFE type selection by driver or eFuse 1h: RFE type defined by the driver (discard the below setting of 0x13A[6:0]) 0h: RFE type defined by 0x13A[6:0]
0x13B	Country code	[7:0]	FFh	Driver will set the ISO 3166-1 country code according to eFuse 0xC8 (channel plan) if the value is 0xFFFF.
0x13C	Country code	[7:0]	FFh	
0x13D	RSVD	[7:0]		Reserved
0x13E	RSVD	[7:0]		Reserved
0x13F	RSVD	[7:0]		Reserved

7.2.1.2.3 Cap-Touch Calibration

The logical zone 0x160 ~ 0x16F is reserved for Cap-Touch calibration.

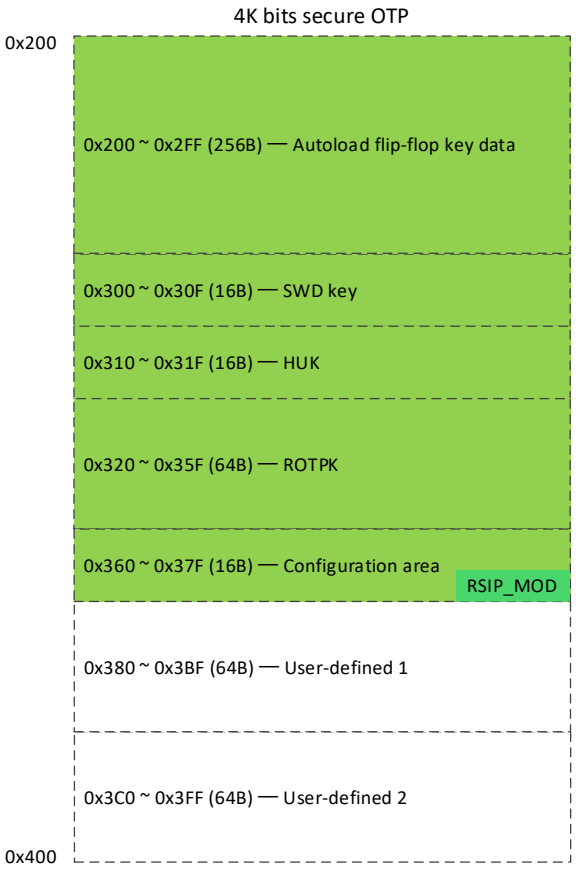
7.2.1.2.4 User-defined MTP

The logical zone 0x170 ~ 0x19F can be allocated freely by users according to the application.

7.2.2 Security Zone

7.2.2.1 Security Zone Layout

Physical area 0x200~0x400 is used for security function, as illustrated in *Figure 7-3*, and this security zone will be visited using a physical address directly. However, access to this zone is limited, and the whole zone will be protected in RMA mode.



Note:

- *The zone filled with green color will be autoloaded to internal memory by hardware.*
- *The RW access to each smaller zone is controlled by configuration area.*

Figure 7-3 Security zone layout

All information provided in this document is subject to legal disclaimers.
© REALTEK 2024. All rights reserved.

7.2.2.1.1 0x200 ~ 0x35F

Table 7-2 Logical area auto-load condition

Function	Name	Size (bits)	Start offset	End offset	Load order	Specific IP can read?
IPSEC	S_IPSEC_Key1 (RDP)	256	0x200	0x021F	Negative	Y (IPSEC_S)
IPSEC	S_IPSEC_Key2 (Secure boot HMAC)	256	0x220	0x023F	Negative	Y (IPSEC_S)
IPSEC	NS_IPSEC_Key1	256	0x240	0x025F	Negative	Y (IPSEC_NS)
IPSEC	NS_IPSEC_Key2	256	0x260	0x027F	Negative	Y (IPSEC_NS)
USER PRI	USER_PRI_KEY1	256	0x280	0x029F	Positive	
USER PRI	USER_PRI_KEY2	256	0x2A0	0x02BF	Positive	
RSIP	RSIP_ECB_KEY	256	0x2C0	0x02DF	Negative	Y (RSIP)
RSIP	RSIP_CTR_KEY	256	0x2E0	0x02FF	Negative	Y (RSIP)
SWD	SWD_PASSWORD	128	0x300	0x030F	Positive	Y (SWD_PWD)
PSA	HUK	128	0x310	0x031F	Negative	N (0x41000310)
Secure Boot	PK1 (ROTPK hash)	256	0x320	0x033F	Positive	N (0x41000320)
Secure Boot	PK2 (ROTPK hash)	256	0x340	0x035F	Positive	N (0x41000340)

7.2.2.1.2 0x360 ~ 0x37F

Table 7-3 Logical area auto-load condition

Offset	Bit	Symbol	Description	HW address	INI
0x360	[31:0]	SWD_ID	SWDID use to mapping the Real SWD Key	Core-Sight internal REG	8'hff
0x364	[0]	SWD_PWD_EN	SWD Password Enable	Core-Sight internal REG	1'h1
	[1]	SWD_DBGEN	SWD external debug authentication	No-readable REG	1'b1
	[2]	SWD_NIDEN		No-readable REG	1'b1
	[3]	SWD_SPIDEN		No-readable REG	1'b1
	[4]	SWD_SPNIDEN		No-readable REG	1'b1
	[5]	SWD_PWD_R_Protection_EN	Key write protection and read protections	No-readable REG	1'b1
	[6]	SWD_PWD_W_Forbidden_EN		No-readable REG	1'b1
	[7]	HUK_W_Forbidden_EN		No-readable REG	1'b1
0x365	[0]	RSVD		No-readable REG	1'b1
	[1]	PK1_W_Forbidden_EN		No-readable REG	1'b1
	[2]	PK2_W_Forbidden_EN		No-readable REG	1'b1
	[3]	S_IPSEC_Key1_R_Protection_EN		No-readable REG	1'b1
	[4]	S_IPSEC_Key1_W_Forbidden_EN		No-readable REG	1'b1
	[5]	S_IPSEC_Key2_R_Protection_EN		No-readable REG	1'b1
	[6]	S_IPSEC_Key2_W_Forbidden_EN		No-readable REG	1'b1
	[7]	NS_IPSEC_Key1_R_Protection_EN		No-readable REG	1'b1
0x366	[0]	NS_IPSEC_Key1_W_Forbidden_EN		No-readable REG	1'b1
	[1]	NS_IPSEC_Key2_R_Protection_EN		No-readable REG	1'b1
	[2]	NS_IPSEC_Key2_W_Forbidden_EN		No-readable REG	1'b1
	[3]	USER_PRI_KEY1_R_Protection_EN		No-readable REG	1'b1
	[4]	USER_PRI_KEY1_W_Forbidden_EN		No-readable REG	1'b1
	[5]	USER_PRI_KEY2_R_Protection_EN		No-readable REG	1'b1
	[6]	USER_PRI_KEY2_W_Forbidden_EN		No-readable REG	1'b1
	[7]	RSIP_KEY1_R_Protection_EN		No-readable REG	1'b1
0x367	[0]	RSIP_KEY1_W_Forbidden_EN		No-readable REG	1'b1
	[1]	RSIP_KEY2_R_Protection_EN		No-readable REG	1'b1
	[2]	RSIP_KEY2_W_Forbidden_EN		No-readable REG	1'b1
	[3]	RSIP_MODE_W_Forbidden_EN	RSIP_MODE	No-readable REG	1'b1
	[4]	SIC_SECURE_EN		No-readable REG	1'b1
	[5]	CPU_PC_DBG_EN		No-readable REG	1'b1
	[6]	UDF1_TRUSTZONE_EN		No-readable REG	1'b1
	[7]	UDF2_TRUSTZONE_EN		No-readable REG	1'b1
0x368	[0]	HWTRIG_UART_DOWNLOAD_DISABLE		0x41000368[0]	1'b1

All information provided in this document is subject to legal disclaimers.
© REALTEK 2024. All rights reserved.

	[1]	RSVD		0x41000368[1]	1'b1
	[2]	RSIP_EN		0x41000368[2]	1'b1
	[3]	SECURE_BOOT_EN		0x41000368[3]	1'b1
	[4]	SECURE_BOOT_HW_DIS		0x41000368[4]	1'b1
	[5]	RDP_EN		0x41000368[5]	1'b1
	[6]	Anti_Roback_EN		0x41000368[6]	1'b1
	[7]	FAULT_LOG_PRINT_DIS		0x41000368[7]	1'b1
0x369	[1:0]	RSIP_MODE		0x41000368[9:8]	2'b11
	[2]	HUK_DERIV_EN		0x41000368[10]	1'b1
	[3]	User_physical_Trustzone1_EN		0x41000368[11]	1'b1
	[4]	User_physical_Trustzone2_EN		0x41000368[12]	1'b1
	[5]	SW_RSVD0		0x41000368[13]	1'b1
	[6]	SWTRIG_UART_DOWNLOAD_DISABLE		0x41000368[14]	1'b1
	[7]	SPIC_PINMUX_IN_TESTMODE_DISABLE		0x41000368[15]	5'h1f
0x36A	[7:0]	RSVD		0x41000368[23:16]	8'hff
0x36B	[3:0]	SECURE_BOOT_AUTH_ALG		0x41000368[27:24]	4'hf
0x36B	[7:4]	SECURE_BOOT_HASH_ALG		0x41000368[31:28]	4'hf
0x36C	[15:0]	OTA_ADDR	OTA address, 4K aligned	0x4100036C[15:0]	16'hffff
0x36E	[15:0]	BOOTLOADER_VER	Bootloader version	0x4100036C[31:16]	16'hffff
0x370	[31:0]	CRC0	CRC check	0x41000370[31:0]	32'hffffffff
0x374	[31:0]	CRC1		0x41000374[31:0]	32'hffffffff
0x378	[31:0]	CRC2		0x4100037A[31:0]	32'hffffffff
0x37C	[31:0]	CRC3		0x4100037C[31:0]	32'hffffffff

7.2.2.2 Security Area Auto-load

7.2.2.2.1 Auto-load Control

This security area can be divided into key area (0x200~0x35F), configuration area (0x360~0x37F) and user-defined area (0x380~0x3FF). The key area can also be divided into two groups: software inaccessible area (0x200~0x30F) and software accessible area (0x310~0x35F).

The auto-load software inaccessible data in a flip-flop can only be accessed by the special HW engine; there is no other way to access the flip-flop area. e.g. S_IPSEC_KEY can only be accessed by secure IPsec, and other HW engines even non-secure IPsec cannot access this key.

The software accessible area will be loaded to the read-only area and can be accessed by CPU. This area contains two kinds of keys – HUK and Public key. The HUK is protected by the sticky bit. The ROM boot code will set the sticky bit after reading the HUK. After that, CPU cannot access the HUK any longer.

The configuration area will be auto-loaded to the internal read-only registers.

All information provided in this document is subject to legal disclaimers.
© REALTEK 2024. All rights reserved.

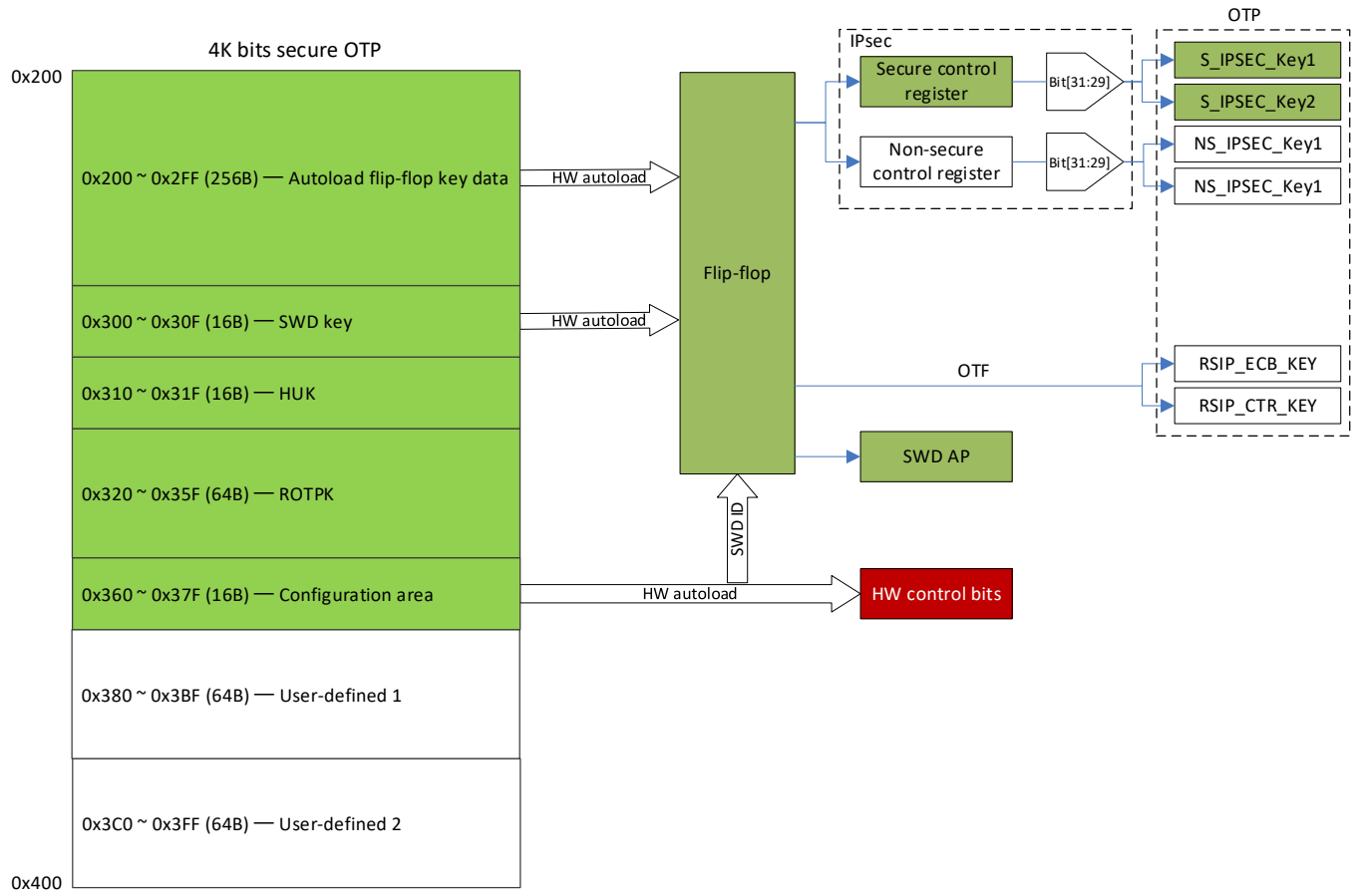


Figure 7-4 Security auto-load control

NOTE

- *The IPsec key located in 0x200~0x23F can only be accessed by secure IPsec with the corresponding Key ID.*
- *The IPsec key located in 0x240~0x27F can only be accessed by non-secure IPsec with the corresponding Key ID.*
- *The ECDSA key can only be accessed by ECDSA master with the corresponding Key ID.*
- *The RSIP key can only be accessed by OTF master and IPsec with the corresponding Key ID.*
- *The SWD key and SWD ID can only be accessed by SWD access point.*
- *Configuration data located in 0x368~0x369 will be auto-loaded to the internal read-only register.*

7.2.2.2.2 Auto-load Data Access Flow

In this section, IPsec will be taken as an example to illustrate how auto-load keys work.

There are two types of IPsec keys, which are used for secure IPsec and non-secure IPsec respectively. Software should configure these Key IDs to the corresponding IPsec, and then enable IPsec when using these keys.

- S_IPSEC_Key: the IPsec key located in 0x200~0x23F for secure IPsec. After configured and enabled by software, the secure IPsec will use this key to encrypt or decrypt the data.
- NS_IPSEC_Key: the IPsec key located in 0x240~0x27F for non-secure IPsec. After configured and enabled by software, the non-secure IPsec will use this key to encrypt or decrypt the data.
- RSIP key can also be loaded to IPSEC to achieve image encryption. Both Secure IPsec and non-secure IPsec can access the keys by correct Key ID.

7.2.2.3 Configuration Area

7.2.2.3.1 Security Area Read Access Control

In the configuration area, most of the bits are used to control the write enable or read enable of the key area. Hardware will autoloader these bits and use them as access permission control.

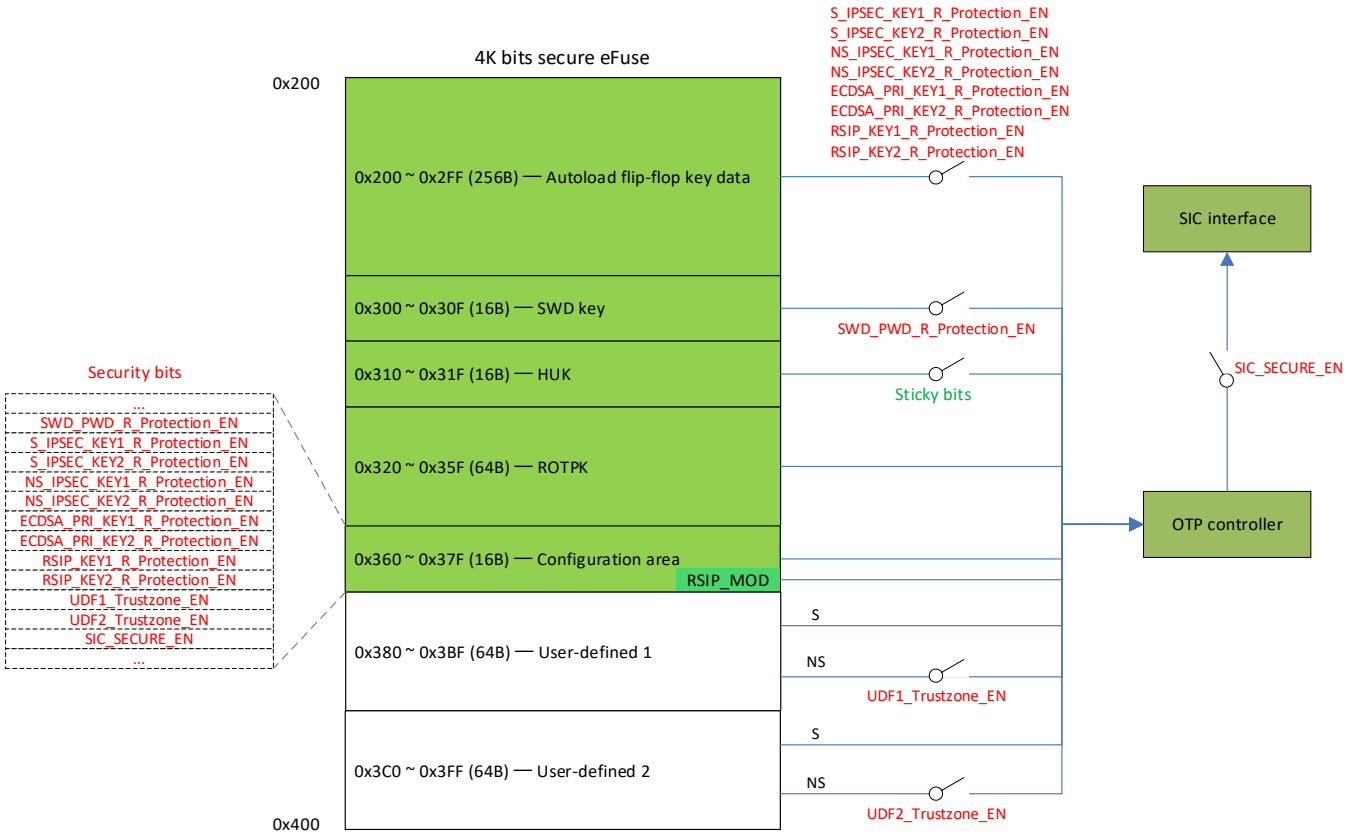
For the key zone, each key has a read forbidden control bit in the configuration area. If this bit is programmed to '0', the key cannot be read

All information provided in this document is subject to legal disclaimers.
© REALTEK 2024. All rights reserved.

anymore.

The HUK can only be read before a sticky bit written. In ROM boot code, software will read HUK and write the sticky bits. So the HUK can only be accessed in ROM boot code.

Each user-defined zone has a control bit. If this bit is programmed to '0', this zone can only be accessed from secure word.



Note: The SIC read forbidden bits only control the access of security zone. The SIC can always visit other zones.

Figure 7-5 Security area read access control

The SIC_EN bit is used to cut off the path of SIC interface to all internal registers, meaning that if SIC_EN is programmed, the SIC is totally disabled. The SIC_EN can't read In RMA mode, so the SIC can access internal registers, however, the OTP area from 0x200~0x400 cannot be accessed in RMA mode, the system registers and other OTP regions can be accessed from SIC.

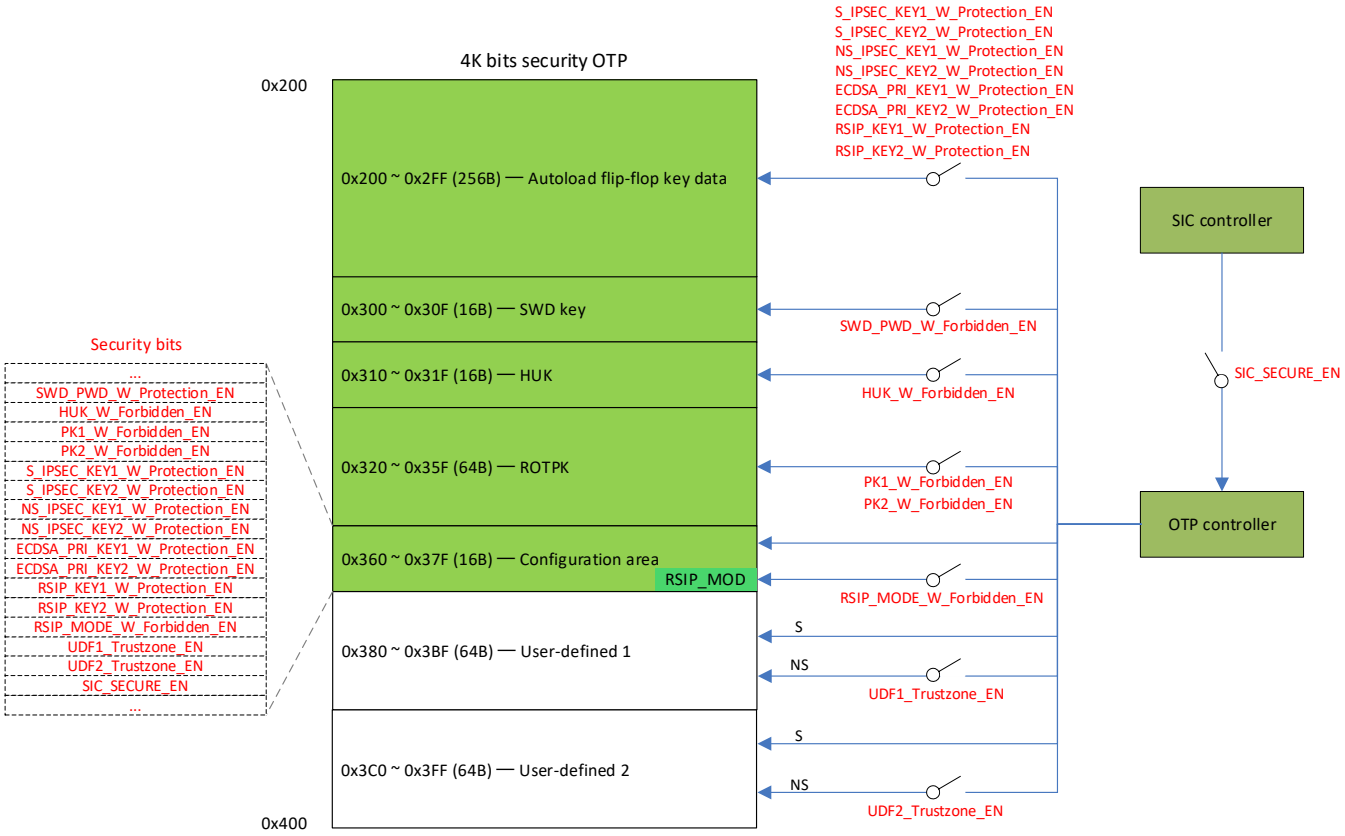
7.2.2.3.2 Security Area Write Access Control

For the key zone, each key has a write forbidden control bit in the configuration area. If this bit is programmed to '0', the key cannot be programmed anymore.

The RSIP mode in the configuration area also has a write forbidden control bit. If this bit is programmed to '0', RSIP cannot be programmed again.

Each user-defined zone has a control bit. If this bit is programmed to '0', this zone can only be accessed from secure word.

All information provided in this document is subject to legal disclaimers.
© REALTEK 2024. All rights reserved.



Note: The SIC read forbidden bits only control the access of security zone. The SIC can always visit other zones.

Figure 7-6 Security area write access control

7.2.2.4 CRC Check

To avoid the fault injection, an internal LDO is added as the power supply of OTP. The LDO can filter most of the glitches. From published information, a hacker may have the ability to influence one OTP bit, to prevent this problem from being our weakness, CRC16 check is added.

The CRC is comprised of four CRC entries, each entry is 32 bits, which includes a 16-bit magic number and 16-bit CRC. CRC check passes only when both the magic number and CRC value are matched.

	CRC disabled		Entry 0 enabled		Entry 1 enabled	
Entry 0	Magic Number	0xFFFF	Magic Number	0x8730	Magic Number	0x0000
	Valid CRC16	0xFFFF	Valid CRC16	XXXX	Valid CRC16	0x0000
Entry 1	Magic Number	0xFFFF	Magic Number	0xFFFF	Magic Number	0x8730
	Valid CRC16	0xFFFF	Valid CRC16	0xFFFF	Valid CRC16	XXXX
Entry 2	Magic Number	0xFFFF	Magic Number	0xFFFF	Magic Number	0xFFFF
	Valid CRC16	0xFFFF	Valid CRC16	0xFFFF	Valid CRC16	0xFFFF
Entry 3	Magic Number	0xFFFF	Magic Number	0xFFFF	Magic Number	0xFFFF
	Valid CRC16	0xFFFF	Valid CRC16	0xFFFF	Valid CRC16	0xFFFF

Figure 7-7 CRC check

If the CRC check fails, hardware will bypass this information to software. The information is stored in two 32-bit registers, which includes the compare result flag, expected CRC, valid CRC also with the magic number. SW will just print out the expected and real CRC value in ROM and go to an endless loop instead of normal boot.

By default, the value of CRC entry 0 is all 0xFF, indicating that the CRC check is disabled; otherwise, CRC check is enabled.

If entry 0 is neither all 0xFF nor all 0x00, entry 0 is enabled. If entry 0 is all 0x0, it indicates that entry 0 is disabled and the next entry is enabled; and so on until entry 3. If the last entry check fails, it means that CRC check fails.

All information provided in this document is subject to legal disclaimers.
© REALTEK 2024. All rights reserved.

NOTE

- *CRC check cannot be disabled once enabled. If entry 0 is all 0x00 and entry 1 is all 0xFF, the CRC check will fail with at least wrong magic number.*
- *CRC check fails if entry 0~entry 3 are all zero.*

HW will auto-load the entry 0 twice with a random delay between them, and do OR operation to calculate the final load result. If the final result equals 0xFF, CRC check is disabled; otherwise, CRC check is enabled.

Whether CRC check needs to be done is related to the auto-load of security zone. Each time the system auto-loads the security zone when boot, CRC will be checked. All the reset types including CHIP_EN, POR, WDG, SYS_RET, BOR, Thermal_RST, and waking up from deep-sleep will trigger auto-load, thus CRC check will be performed by HW subsequently and automatically.

If customers want to enable CRC check, we suggest do it in the last step at factory. Hardware of Realtek already provides the ability to calculate the CRC even if the related keys are read-forbidden, for customers, software only needs to trigger hardware to do a calculation for all auto-load in security zone (0x200~0x36B), then read out the CRC value and write it into the corresponding entry.

7.2.2.5 Random Load

Some hackers may want to crack the OTP content by observing the current. Even the OTP principle is based on the difference of current, but it reads out 32 bits at the same time, also the OTP itself did the balance. From the technology we know now, it's very hard to distinguish the difference. However, to achieve a supper secure level, a random seed is added to OTP auto-load flow. When HW does secure zone auto-load, the random seed comes from two different clocks, and the auto-load scheme is not published to the public.

7.2.2.6 Multi-time Load

The PSA area and CRC entries can't be protected by the above strategy, here we use a multi-time load scheme to prevent it from being cracked. Both PSA and CRC will be loaded more than one time, and the final result is the "or operation" of multi-read.

7.2.2.7 Protected in RMA Mode

RMA means Return Material Authorization. When customers have already programmed their keys or security materials into the security zone, and then suspect that there are some defects with the chip, they need Realtek's help to analyze, but do not want Realtek to touch any of their security materials. That is why this security area needs to be protected in RMA mode.

RMA mode can be achieved by programming the first byte of PSA (Life Status) in hidden OTP.

- If the number of 1 in this byte is even, it means that the chip is in normal mode and the access permission to each zone is determined by configuration area.
- If the number of 1 in this byte is odd, it means that the chip is in RMA mode, writing to security zone is forbidden and reading from this zone always returns 1.

All information provided in this document is subject to legal disclaimers.
© REALTEK 2024. All rights reserved.

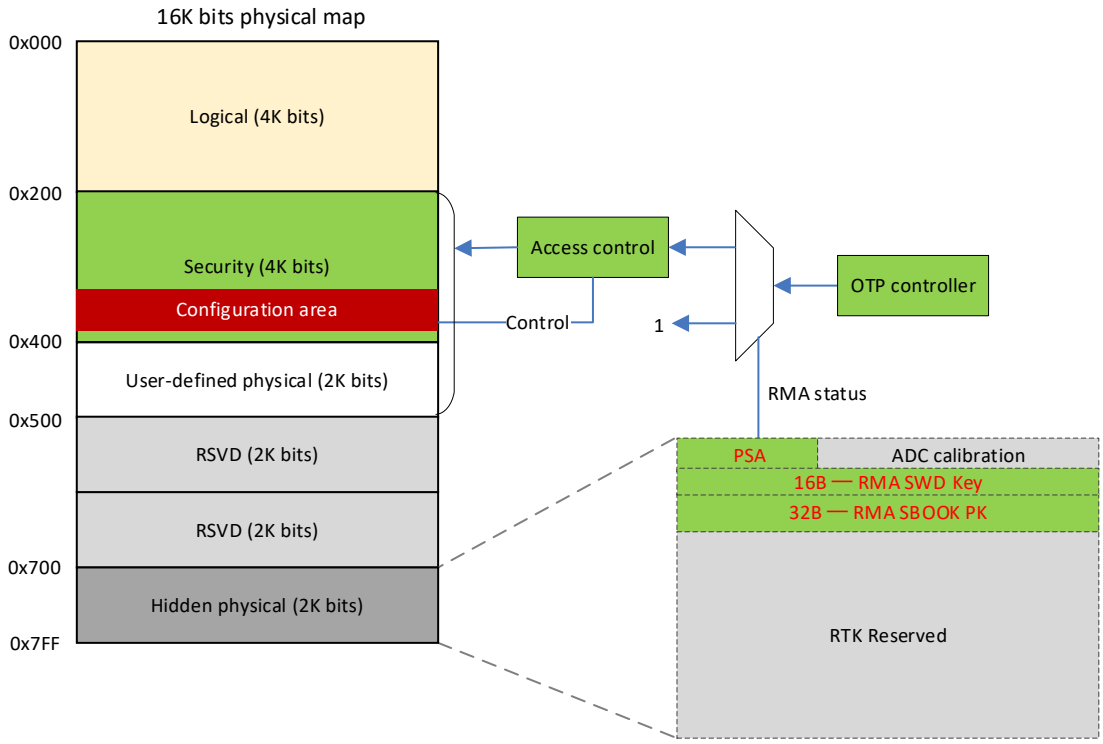


Figure 7-8 Security zone access control

In RMA mode, HW cannot access the whole security zone and user-defined physical zone, because the security zone is all 0xFF, just like a new chip. However, we don't want anyone to have the ability to look inside into the chip. So in this case, HW will always load SWD Key into core-sight, customers need to offer the SWD key to Realtek. Moreover, if the RMA Key is programmed (not all 0xFF), the ROM boot code will enable secure boot, customers also need to offer the secure boot key pair to Realtek or help Realtek do signature.

The PSA has 8 bits to control the life cycle of the device, so one device can be programed to RMA mode 4 times at most. And customers need to program the chip to RMA mode before shipping the chip to Realtek, because Realtek cannot program OTP without the customer's help.

The SWD key and SWD ID are designed to prevent from unexpected debug. You may write different keys to each device, and need to get the SWD ID by J-Link first, then find the SWD key in your database. Finally, use the SWD key to enable the debug function.

NOTE

It is the most convenient way to get SWD ID by J-Link.

7.2.2.8 Load Order

When the Keys are auto-loaded to HW, some of them are using positive order while others using the negative order. The Keys that may be used by IPSEC all use negative order, so that it can get the same results with mbedtls. The definition of load order is illustrated in *Figure 7-9* and *Figure 7-10*.

All information provided in this document is subject to legal disclaimers.
© REALTEK 2024. All rights reserved.

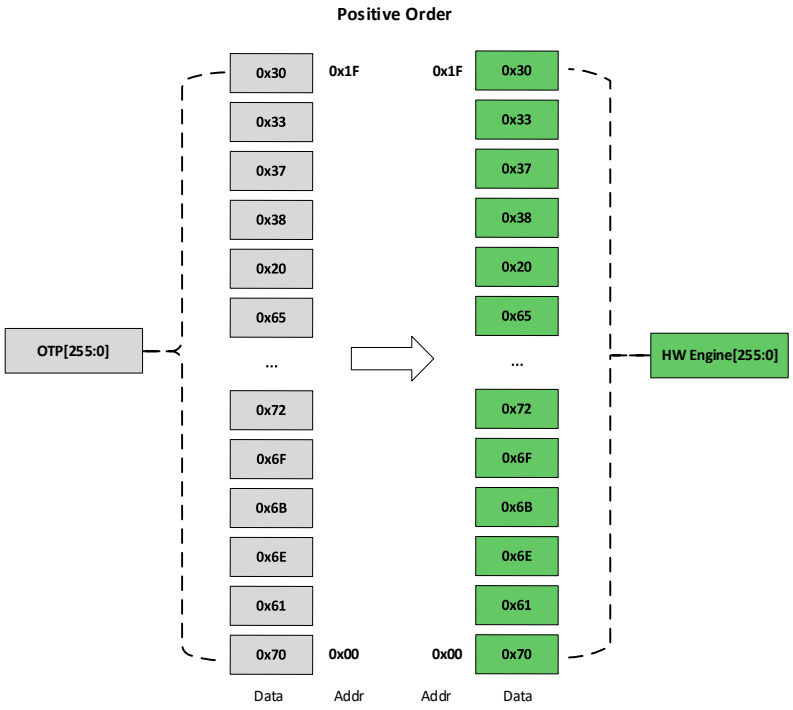


Figure 7-9 Positive load order

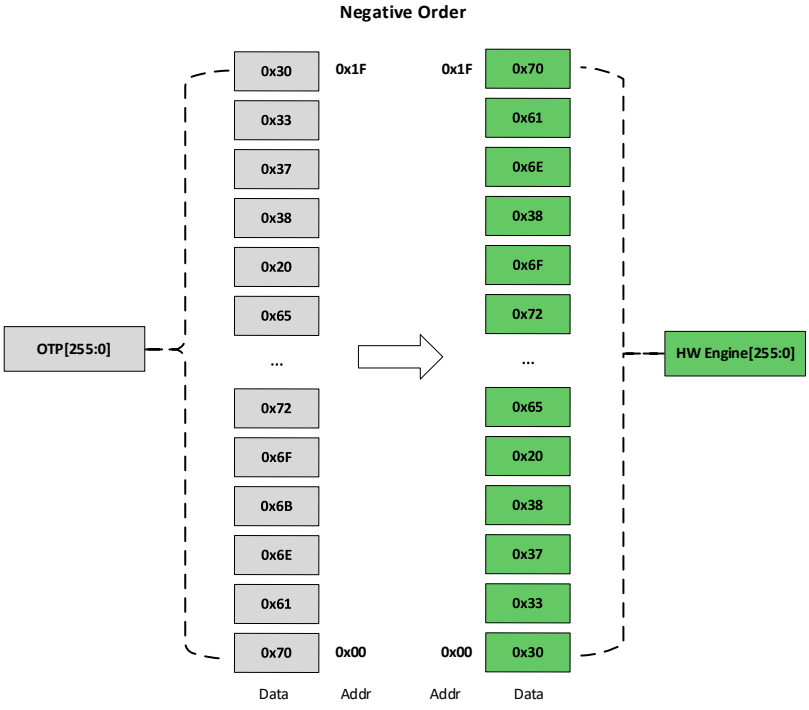


Figure 7-10 Negative load order

7.2.2.9 User-defined Area

In the security zone, there are two user-defined areas: 0x380~0x3BF and 0x3C0~0x3FF. These areas can be defined to secure-only OTP areas by the configuration bits. The secure-only area is an area that can only be accessed from secure world, customers can put the keys with higher security level in this area.

By default, these two areas are non-secure. If the bit UDF1_Trustzone_EN (0x367[6]) is programmed, the 0x380~0x3BF area can only be operated in secure word. Thus, the write from non-secure word is invalid and read from non-secure word returns all 0xFF. The configuration bit UDF2_Trustzone_EN has the same function.

All information provided in this document is subject to legal disclaimers.
© REALTEK 2024. All rights reserved.

7.2.3 User-defined Zone

The user-defined zone can be used as a rollback counter private key or any other purposes freely.

The size of this area is 256 bytes, and can be divided into two areas: 0x400~0x47F and 0x480~0x4FF. These areas can be defined to secure-only OTP areas by the configuration bits. The secure-only area is an area that can only be accessed from secure world, customers can put the keys with higher security level in this area.

By default, these two areas are non-secure. If the bit User_physical_Trustzone1_EN (0x369[3]) is programmed, the 0x400~0x47F area can only be operated in secure word. Thus, the write from non-secure word is invalid and read from non-secure word returns all 0xFF. The configuration bit User_physical_Trustzone2_EN has the same function.

7.2.4 ROM Code Patch Zone

The ROM code patch zone stores the ROM code in a Realtek-defined format, and the format won't be published to the public. This area is 512 bytes and divided into two parts. In case the ROM code patch only needs less than 256 bytes, another part can be used for other purposes.

ROM code patch can always be read. But in PSA area, the two bits ROM_PATCH_LWE and ROM_PATCH_HWE are used to control write permission to this zone.

- If ROM_PATCH_LWE is programmed, the 0x500~0x5FF cannot be programmed anymore.
- If ROM_PATCH_HWE is programmed, the 0x600~0x6FF cannot be programmed anymore.

For secure concerns, the ROM_PATCH_LWE is always suggested to be programmed in the factory. The low 256 bytes of ROM code patch zone (0x500~0x5FF) is not allowed for other usages. If you want to use the high 256 bytes (0x600~0x6FF), you should guarantee that the bit ROM_PATCH_LWE is programmed and the last 4 bytes of the low 256 bytes is empty (0x5FC = 0xFFFFFFFF).

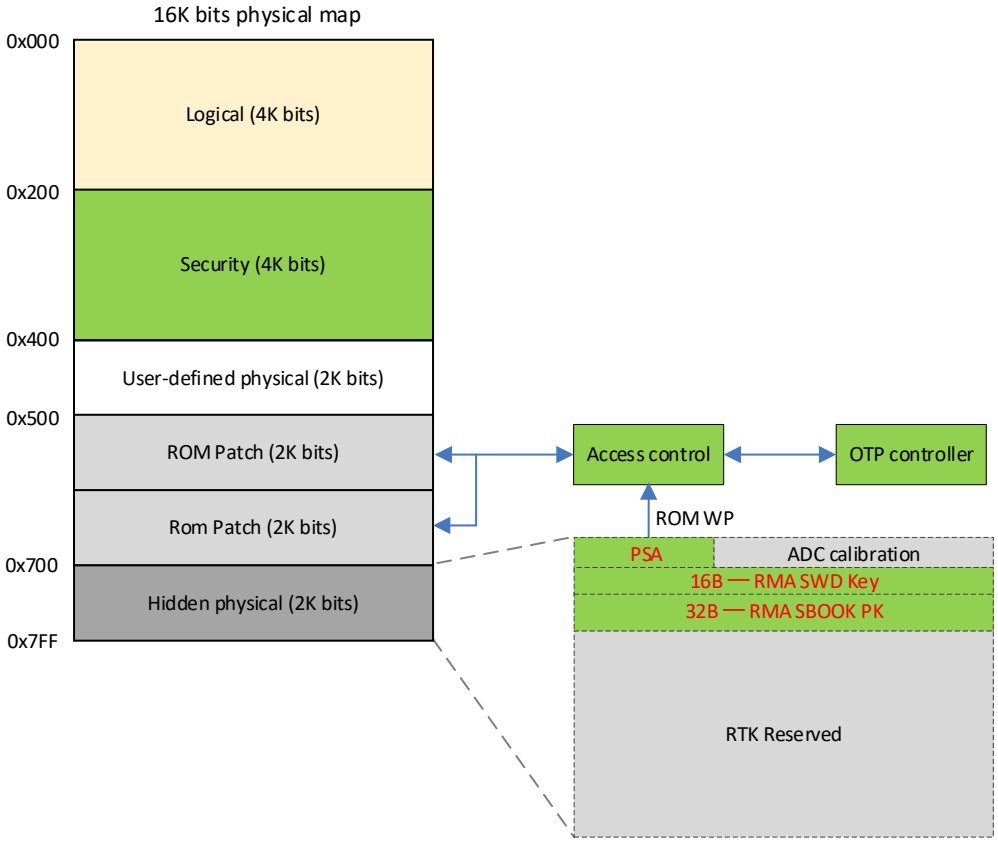


Figure 7-11 ROM code patch area

7.2.5 Hidden Physical Zone

The hidden physical zone contains some RMA keys and the Realtek's calibration data. Users can only program the RMA-related area.

All information provided in this document is subject to legal disclaimers.
© REALTEK 2024. All rights reserved.

In this area, two keys have their own read protection and write protection. These two keys will be auto-loaded to the HW.

- For SWD Key, a non-programmed value means that the key is all 0xFF.
- For secure boot public key hash, a non-programmed value means that the secure boot is disabled in RMA mode.

Offset	Bit	Symbol	Description	HW address	HW INI
0x700	[7:0]	RMA	Life State HW should auto-load life state first when boot. The default value is 0xFF and it's in normal mode. When this field is programmed and the most significant 0 bit is even (bit 0, 2, 4, 6), the device is in RMA mode; otherwise, it is in normal mode. In RMA mode, the secure 4K bits should be protected and return all "1" when read.	System register	8'hff
0x701	[1:0]	ROM_PATCH_EN	ROM path enable This bit will be programmed by FT if ROM patch does not needed.	System register	1'h1
	[2]	ROM_PATCH_LWE1	ROM patch write protection (protecting low 256 bits): 500~51f	No-readable register	1'b1
	[3]	ROM_PATCH_LWE2	ROM patch write protection (protecting low 256bits): 520~53f	No-readable register	1'b1
	[4]	ROM_PATCH_LWE3	ROM patch write protection (protecting low 256bits): 540~55f	No-readable register	1'b1
	[5]	ROM_PATCH_LWE4	ROM patch write protection (protecting low 256bits): 560~57f	No-readable register	1'b1
	[6]	ROM_PATCH_LWE5	ROM patch write protection (protecting low 256bits): 580~5ff	No-readable register	1'b1
	[7]	ROM_PATCH_HWE	ROM patch write protection (protecting high 2K bits)	No-readable register	4'b1f
0x702	[0]	RMA_SWD_PWD_R _Protection_EN	SWD Key read protection	No-readable register	1'b1
	[1]	SWD_PWD_W_Forb idden_EN	SWD Key write protection	No-readable register	1'b1
	[2]	RMA_PK_W_Forbid den_EN	RMA PK write protection	No-readable register	1'b1
	[7:3]	RSVD	Reserved		4'h3f
0x703	[7:0]	RSVD	Reserved for PSA		
0x704~ 0x70F		ADC calibration 0	ADC calibration data 0		
0x710~ 0x71F		RMA SWD Key	SWD password in RMA mode, positive order	Core-Sight internal register	128'hfffff
0x720~ 0x73F		RMA Secure Boot Key	Secure boot public key in RMA mode	System register	256'hff

All information provided in this document is subject to legal disclaimers.
© REALTEK 2024. All rights reserved.

8 Chip Enable

8.1 Overview

The CHIP_EN (chip enable) is an external pin that can be used to control the reset status of the whole SoC. This pin can work in level reset mode, interrupt reset mode or pulse reset mode. By default, it works in level reset mode. This pin always has the function of resetting system no matter in which mode.

- Level reset mode: reset the chip once detects low on CHIP_EN pin, and exit reset after CHIP_EN goes high.
- Interrupt reset mode: fire an interrupt when low detect, and trigger reset if the interrupt isn't cleared with the response time.
- Pulse reset mode: reset the chip once detecting low on CHIP_EN pin, and exit reset immediately.

When the system works in interrupt reset mode, software needs to clear the interrupt within the pre-defined time once the CHIP_EN is pressed. If the software crashes and cannot clear the interrupt in the pre-defined time, hardware will reset the system.

The software interrupt mode is mainly used in the power save scenario. The CHIP_EN pin just likes the power button of the smartphone: a short press will trigger the system to enter low power mode or wake up the system; a long press will trigger the system reboot.

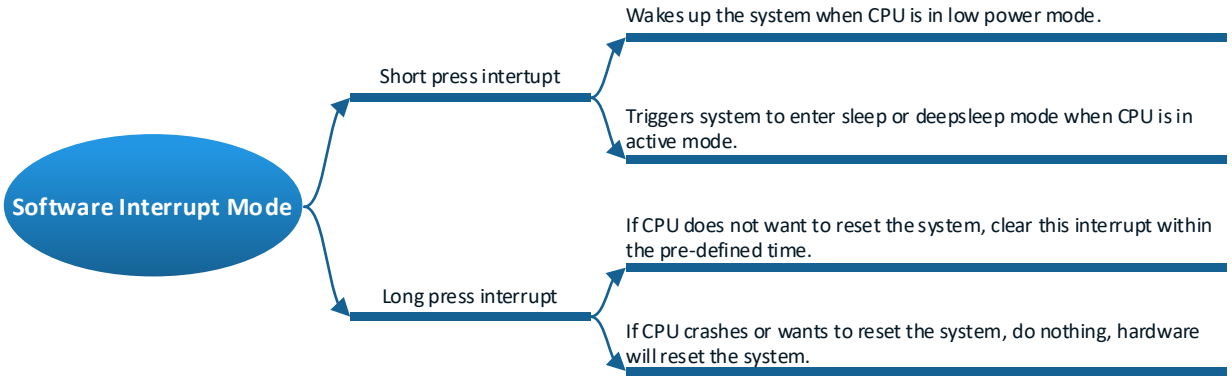


Figure 8-1 CHIP_EN software interrupt mode

One user scenario for pulse reset mode is to reset the device to factory settings. The CHIP_EN pin of the SoC just likes the reset button in the router. From the user's view, a short press will trigger the device reboot, and a long press will reset the device to factory settings. Actually, resetting the device to factory settings is a software behavior; the device will reboot once the CHIP_EN is pressed, and in its initial power flow, it will check the CHIP_EN status. If this pin keeps in low for more than pre-defined time, software will reset the device to factory settings.

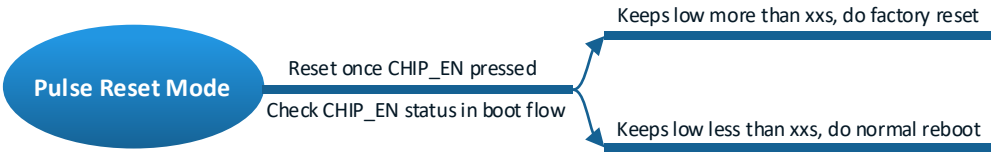


Figure 8-2 CHIP_EN pulse reset mode

By default, the CHIP_EN pin works in level reset mode, software can configure it to software mode in the bootloader or the application. Before configuration, the CHIP_EN can only work in level reset mode.

NOTE

- *The CHIP_EN can reset the whole system in any conditions without the help of software.*
- *The CHIP_EN can control the SWR or other power sources in the right order, even the out-chip circuit uses the internal power source. The CHIP_EN reset has the ability to reset the external chips and to guarantee them work properly.*

8.2 Debounce Function

Debounce function can be enabled by configuring debounce settings; this function is enabled by default with a setting of 100us for T_{DB}, the debounce time can be configured from 100us to 16ms. Only when a continual low or high lasts longer than T_{DB} will the hardware recognize it as a low or high. The hardware will ignore the glitch that is shorter than T_{DB}.

All information provided in this document is subject to legal disclaimers.
© REALTEK 2024. All rights reserved.

Debounce function can work well in level reset mode, interrupt reset mode and pulse reset mode, the function of debounce is shown in *Figure 8-3*.

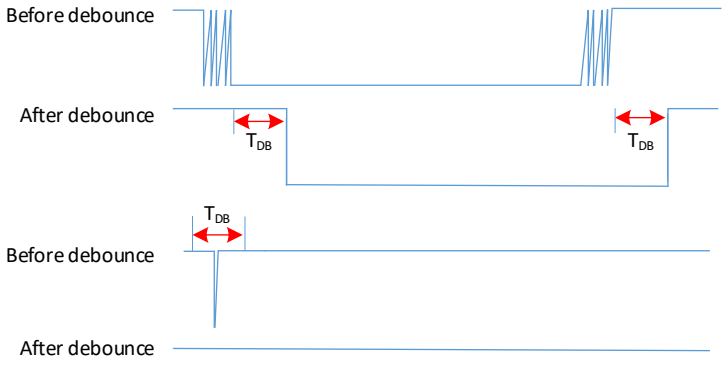


Figure 8-3 Debounce function

8.3 Level Reset Mode

The hardware works in level reset mode by default. When initial power-on, once the chip enable pad rises to high value, the system reset will be released and hardware PMC will power on the system.

The timing sequence of the initial power-on is illustrated in *Figure 8-4*.

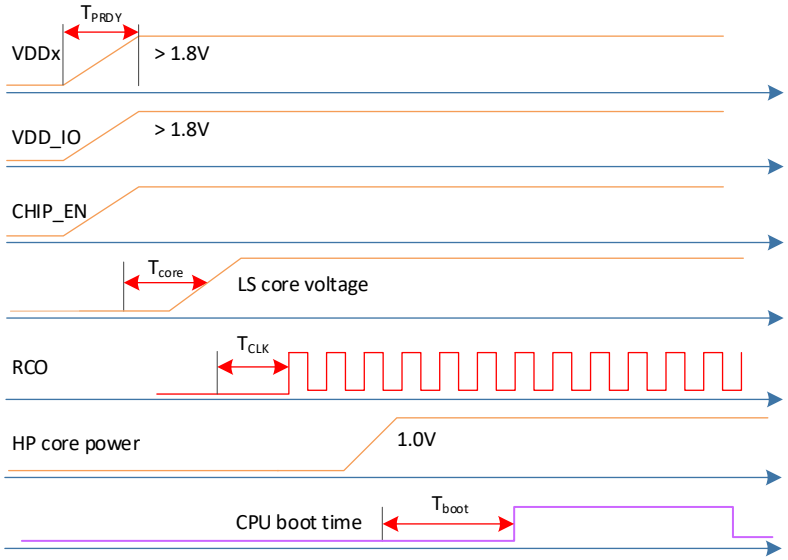


Figure 8-4 Timing sequence of initial power-on

The CHIP_EN will continue to work in reset mode if no software configuration. In level reset mode, the whole system will be reset once a low level is detected on CHIP_EN. Moreover, it keeps in power-down mode until a high level is detected.

8.4 Interrupt Reset Mode

The CHIP_EN can be configured to work in interrupt reset mode. In interrupt reset mode, a level change in CHIP_EN will trigger an interrupt instead of resetting the system.

If the time of CHIP_EN keeps low longer than a short press threshold, a short press interrupt will be triggered. If it keeps low longer than a long press threshold, a long press interrupt will be triggered.

The long-press event should be acknowledged within T_{ack}; otherwise, the system will reset. Behaviors will be illustrated in the following sections; the signal of CHIP_EN means the signal after debounce.

All information provided in this document is subject to legal disclaimers.
© REALTEK 2024. All rights reserved.

Case 1: button press time shorter than T_{sp}

In this case, no interrupt is triggered, just like nothing happens, as shown in *Figure 8-5*.

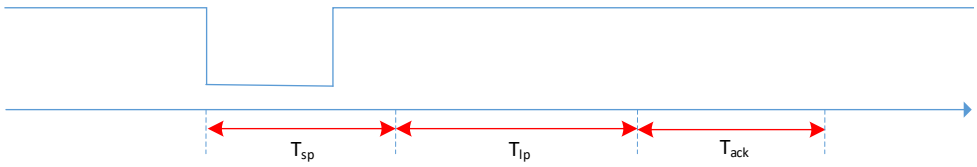


Figure 8-5 Short press without triggering interrupt

Case 2: short press time longer than T_{sp} and shorter than (T_{sp}+T_{lp})

In this case, an interrupt will be triggered with short-press event, as shown in *Figure 8-6*. If the short press interrupt happens in the sleep mode, this interrupt will also wake up the system with a wake-up reason indicating the wake-up event.

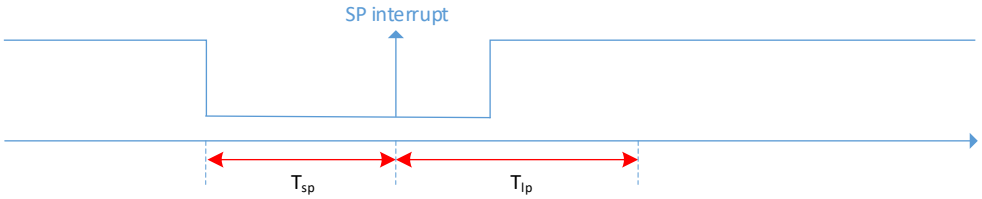


Figure 8-6 Short press triggering short-press interrupt

Case 3: long press and system acknowledge to the interrupt

In this case, the press time is longer than (T_{sp}+T_{lp}), an interrupt will be triggered with a short-press event first, and then a long-press event, as shown in *Figure 8-7*.

If software works properly, it can clear the long press event; otherwise, the hardware will reset the whole system if no acknowledgment after T_{ack}. After software clears the interrupt, no matter how long the button is pressed, there will not be other interrupts.

If the long press interrupt happens in the sleep mode, this interrupt will also wake up system with a wake-up reason indicating the wake-up event.

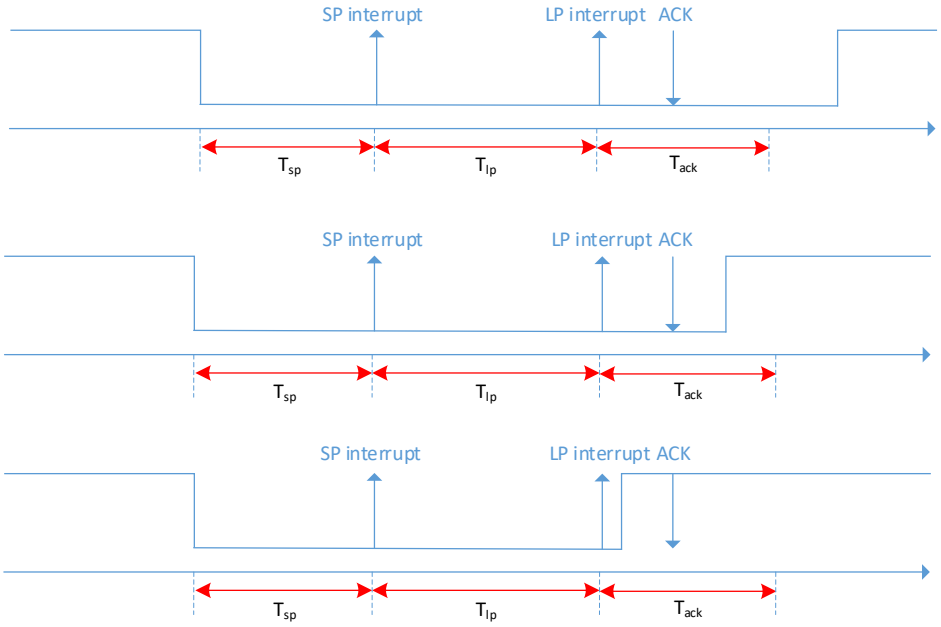


Figure 8-7 Long press without system reset

Case 4: long press and no acknowledge

In this case, the system will reset the system after (T_{sp} + T_{lp} + T_{ack}) if the CHIP_EN stays low and software does not clear the interrupt, and keeps in reset mode until CHIP_EN goes to high, as shown in *Figure 8-8*.

All information provided in this document is subject to legal disclaimers.
© REALTEK 2024. All rights reserved.

When the system boots up after CHIP_EN reset, the settings of CHIP_EN are also reset The CHIP_EN will work in level reset mode until software configures it again.

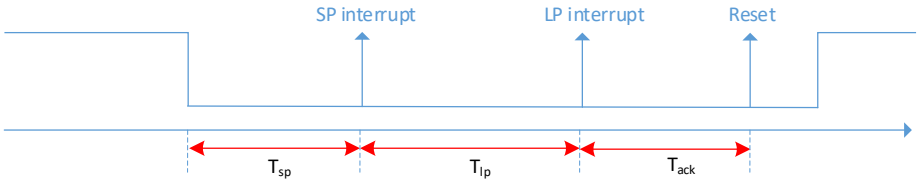


Figure 8-8 Long press with system reset

NOTE

If CHIP_EN stays low when a reset is triggered in software reset mode, the chip will stay in reset mode until CHIP_EN goes high from low.

Only when the CHIP_EN stays low longer than (T_{sp} + T_{lp} + T_{ack}) can CHIP_EN reset the system. Otherwise, the system cannot be reset. The case shown in *Figure 8-9* cannot trigger system reset.

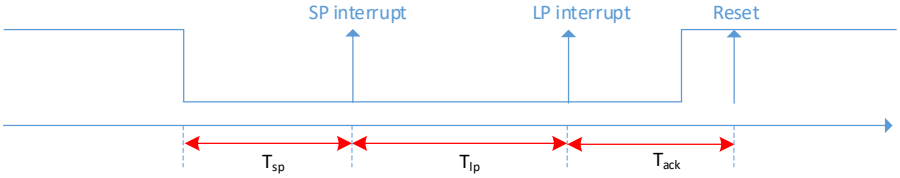


Figure 8-9 Long press without system reset

8.5 Pulse Reset Mode

In order to realize the function of long press to restore factory settings and short press to reset the system directly, the pulse reset mode is added.

The difference between pulse reset mode and level reset mode is that in pulse reset mode the system reboot immediately (in level reset mode, the system reboots after CHIP_EN goes to high). This feature gives software the ability to count the key press period to distinguish short and long press.

8.6 Registers

The memory map of CHIP_EN registers is listed in *Table 8-1*. The base address of CHIP_EN registers is 0x4100_8050.

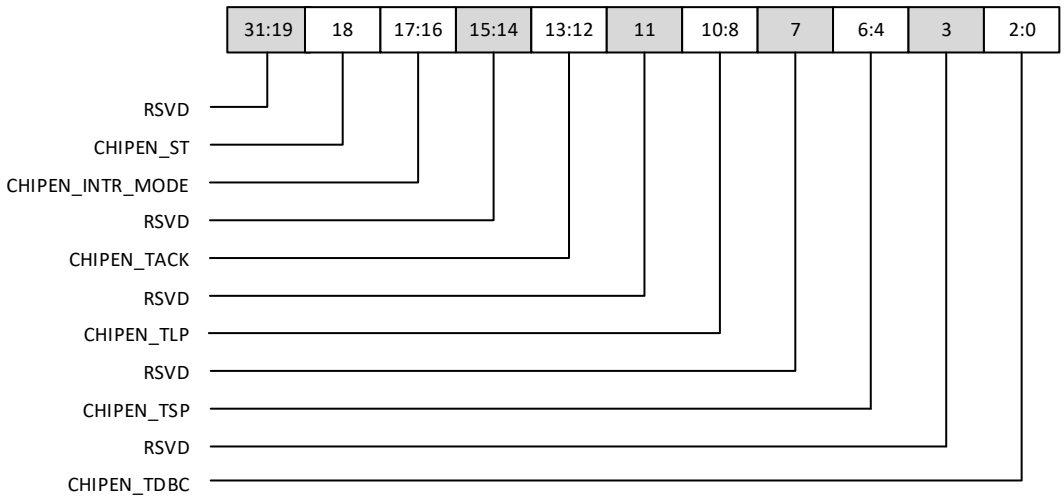
Table 8-1 Memory map of CHIP_EN registers

Name	Offset	Access	Description
REG_AON_CHIPEN_CTRL	0x00	R/W	CHIP_EN Function Control Register
REG_AON_CHIPEN_ISR	0x04	R/W1C	CHIP_EN Interrupt Clear Register

8.6.1 REG_AON_CHIPEN_CTRL

- **Name:** CHIP_EN Function Control Register
- **Size:** 32 bits
- **Address offset:** 0x00
- **Read/write access:** read/write

All information provided in this document is subject to legal disclaimers.
© REALTEK 2024. All rights reserved.

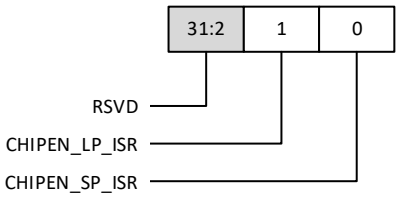


Bit	Symbol	Access	Reset	Description
31:19	RSVD	-	-	Reserved
18	CHIPEN_ST	R	0	CHIP_EN pin status ● 0: Logic '0' on the CHIP_EN pin ● 1: Logic '1' on the CHIP_EN pin
17:16	CHIPEN_INTR_MODE	R/W/ES	00	● 00: reset mode. When CHIP_EN pad is low, HW will reset the whole chip. ● 01: interrupt mode. It will enable short/long press function. ● 1x: pulse reset mode. The transaction from other value to 2'd2 will generate a positive edge to set RST_PULSR_EN in HV domain. Once in pulse reset mode, this field cannot be modified again, and always read out 2'b11.
15:14	RSVD	-	0	Reserved
13:12	CHIPEN_TACK	R/W	0	● 00: 50ms ● 01: 100ms ● 10: 200ms ● 11: 400ms
11	RSVD	-	-	Reserved
10:8	CHIPEN_TLP	R/W	0	● 000: 1000ms ● 001: 1500ms ● 010: 2000ms ● 011: 2500ms ● 100: 3000ms ● 101: 3500ms ● 110: 4000ms ● 111: 4500ms
7	RSVD	-	-	Reserved
6:4	CHIPEN_TSP	R/W	0	● 000: 0ms ● 001: 50ms ● 010: 100ms ● 011: 150ms ● 100: 200ms ● 101: 250ms ● 110: 300ms ● 111: 350ms
3	RSVD	-	-	Reserved
2:0	CHIPEN_TDBC	R/W	1	● 000: 0us ● 001: 100us ● 010: 500us ● 011: 1000us ● 100: 2000us ● 101: 4000us ● 110: 8000us ● 111: 16000us

All information provided in this document is subject to legal disclaimers.
© REALTEK 2024. All rights reserved.

8.6.2 REG_AON_CHIPEN_ISR

- **Name:** CHIP_EN Interrupt Clear Register
- **Size:** 32 bits
- **Address offset:** 0x04
- **Read/write access:** read/write to clear



Bit	Symbol	Access	Description
31:2	RSVD	-	Reserved
1	CHIPEN_LP_ISR	R/W1C	CHIP_EN long-press interrupt status 0: no interrupt 1: interrupt occurs
0	CHIPEN_SP_ISR	R/W1C	CHIP_EN short-press interrupt status 0: no interrupt 1: interrupt occurs

All information provided in this document is subject to legal disclaimers.
© REALTEK 2024. All rights reserved.

9 Reset and Clock Control (RCC)

The RCC module manages the generation of all the clocks, as well as the clock gating and the control of the system and peripheral resets. It provides high flexibility in the choice of clock sources and allows the application of clock ratios to improve power consumption.

9.1 Reset Control

9.1.1 Introduction

In RTL8721Dx, there are serval types of resets: POR, CHIP_EN, BOR, WDG, SYS, and DBG reset.

In principle, each module has only one external asynchronous reset input, but may have multiple clock inputs. The synchronized reset signal is generated according to the input clock inside the module.

9.1.2 Reset Diagram

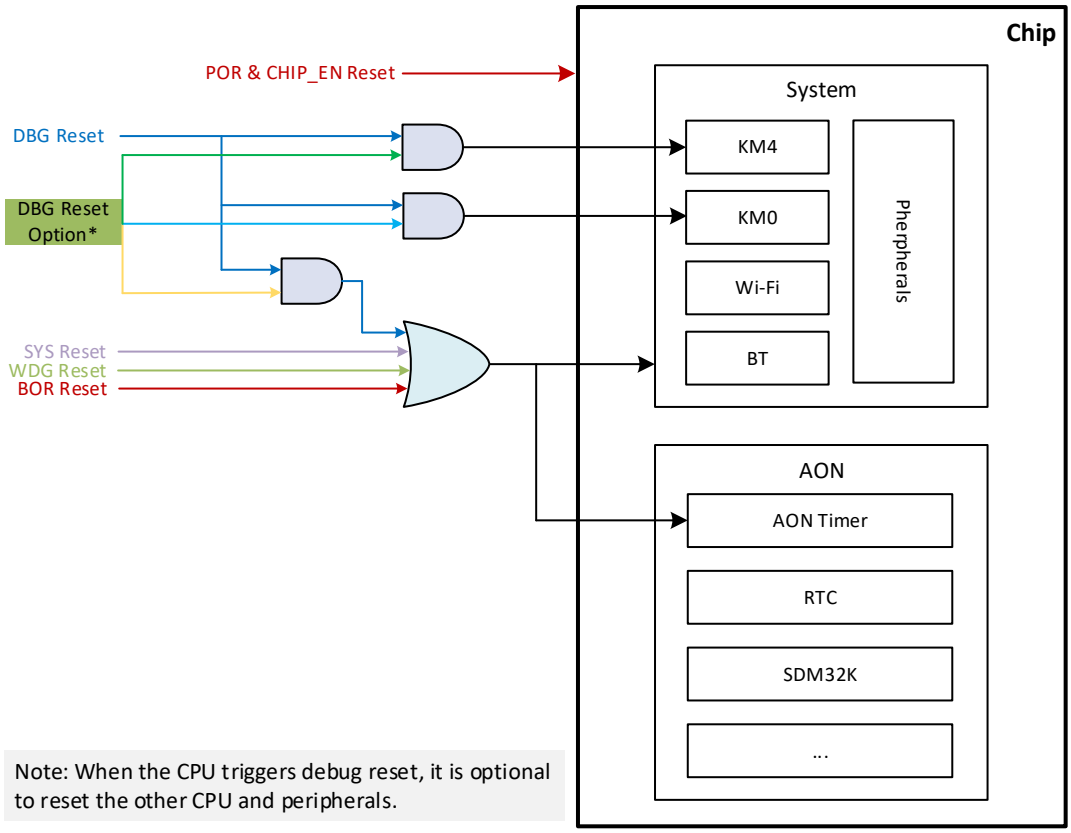


Figure 9-1 Reset diagram

9.1.3 Reset Types

The following reset sources or events, listed in *Table 9-1*, are able to generate a reset.

Table 9-1 Reset types

Reset type	Description
POR	A power-on reset is generated when power on
BOR	A brownout reset is generated when BOR reset detected
CHIP_EN	Generated by external CHIP_EN pin
WDG	A watchdog reset is generated when watchdog timeout

All information provided in this document is subject to legal disclaimers.
© REALTEK 2024. All rights reserved.

SYS	A system reset is triggered by software
DBG	A debug reset is triggered by SWD debug

9.1.4 Reset Domains

Different reset types reset different domains, as shown in *Figure 9-1*.

- The power-on reset (POR) and external CHIP_EN reset can reset the whole chip.
- The BOR reset, SYS reset, and WDG reset can reset the system and AON timer.
- A debug reset is triggered by SWD debug, which will reset the CPU core definitely, and has the flexibility to reset the other CPU core and peripherals.

9.2 Clock Control

This section describes the generation of the various clocks used by RTL8721Dx and options of clock source selection.

9.2.1 Clock Tree

The clock tree of RTL8721Dx is shown in *Figure 9-2*. Different clock sources can be used to drive different functions. Each clock source can be switched on or off independently when it is not used, to optimize the power consumption.

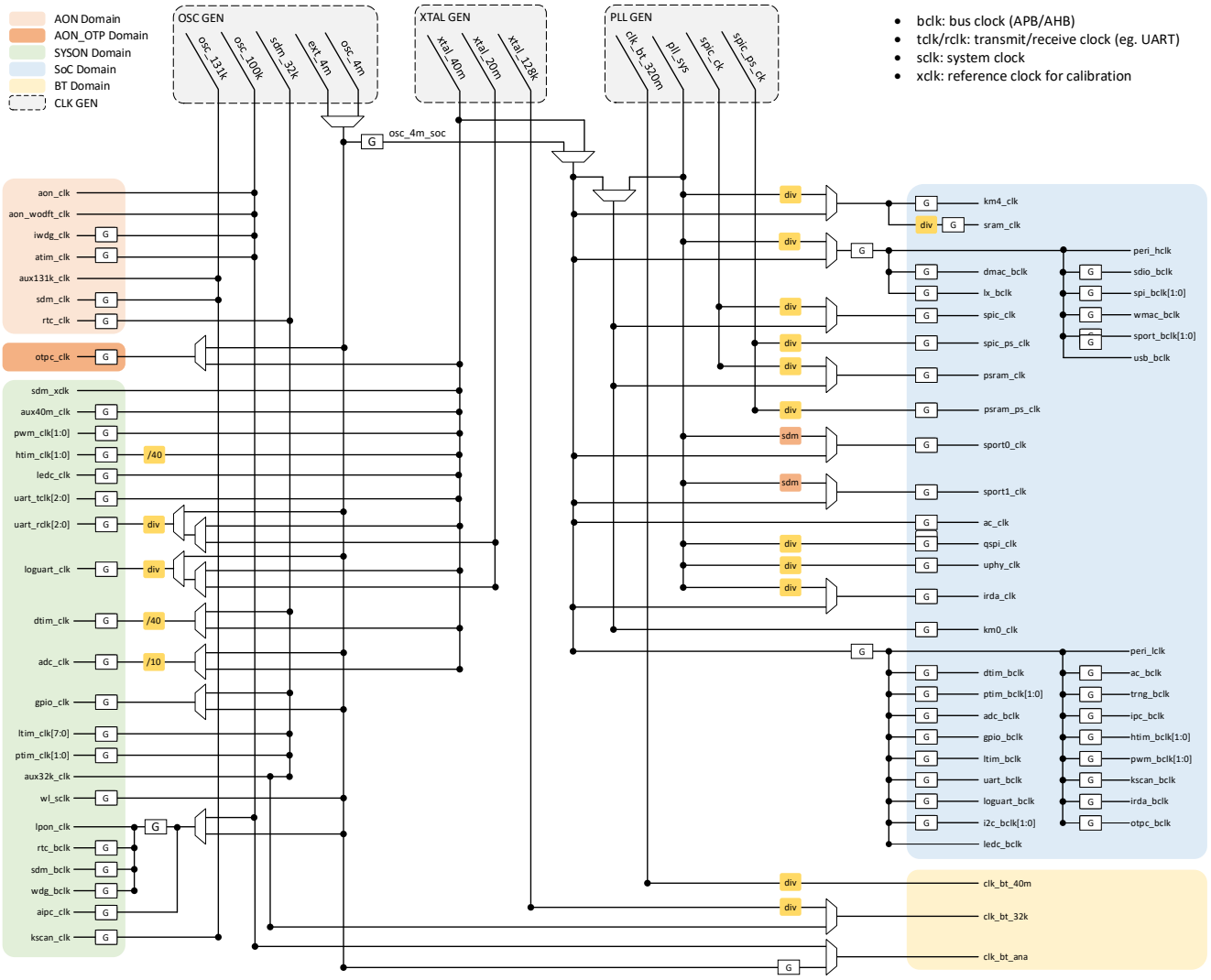


Figure 9-2 Clock tree

All information provided in this document is subject to legal disclaimers.
© REALTEK 2024. All rights reserved.

9.2.2 Clock Source

The clock sources of RTL8721Dx are listed below. Different clock sources can drive different functions.

- 40MHz clock based on external oscillator:
  - XTAL40M: used for peripherals directly or after frequency division.
- Internal oscillators:
  - OSC4M: provides 4MHz clock for KM0 or 2MHz clock for peripherals after frequency division.
  - OSC131K: used for input of SDM and clock for Cap-Touch and Key-Scan.
- Separate PLL:
  - PLL_SYS: 300MHz ~ 600MHz, provides clock for KM4, KM0, and high-performance peripherals after frequency division.

Each clock source can be switched on or off independently when it is not used, to optimize the power consumption.

9.3 Registers

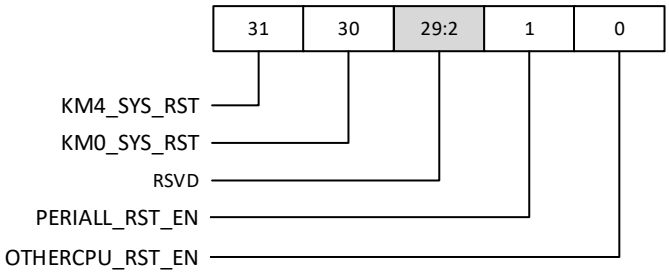
9.3.1 System Reset Registers

The software needs a way to trigger system reset. A healthy way to implement this function is very important. Two registers are used to implement this function. The base address of system reset registers is 0x4100_8200.

9.3.1.1 REG_SW_RST_CTRL

- **Name:** Reset Control Register
- **Size:** 32 bits
- **Address offset:** 0x60
- **Read/write access:** read/write

This register can be written when the REG_SW_RST_TRIG register is 0x96969696. The reserved bits in this register may be used to control the reset domain later.



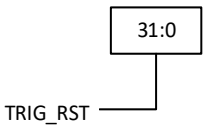
Bit	Symbol	Access	Reset	Description
31	KM4_SYS_RST	R/WE/EC	0	KM4 resets the system
30	KM0_SYS_RST	R/WE/EC	0	KM0 resets the system
29:2	RSVD	-	-	Reserved
1	PERIALL_RST_EN	R/W	0	1: Warm reset request by one CPU will reset all peripherals and local CPU 0: Warm reset request by one CPU will reset local CPU Note: Peripheral reset will also use sysrst mst.
0	OTHERCPU_RST_EN	R/W	0	1: Warm reset request by one CPU will reset other CPU's core and local CPU 0: Warm reset request by one CPU will only reset local CPU

9.3.1.2 REG_SW_RST_TRIG

- **Name:** Reset Trigger Register
- **Size:** 32 bits
- **Address offset:** 0x64
- **Read/write access:** read/write

The CPU needs to enable the write Reset control permission first by writing 0x96969696 to the REG_SW_RST_TRIG register, and then write the REG_SW_RST_CTRL register, finally write 0x69696969 to trigger the system reset.

All information provided in this document is subject to legal disclaimers.
© REALTEK 2024. All rights reserved.



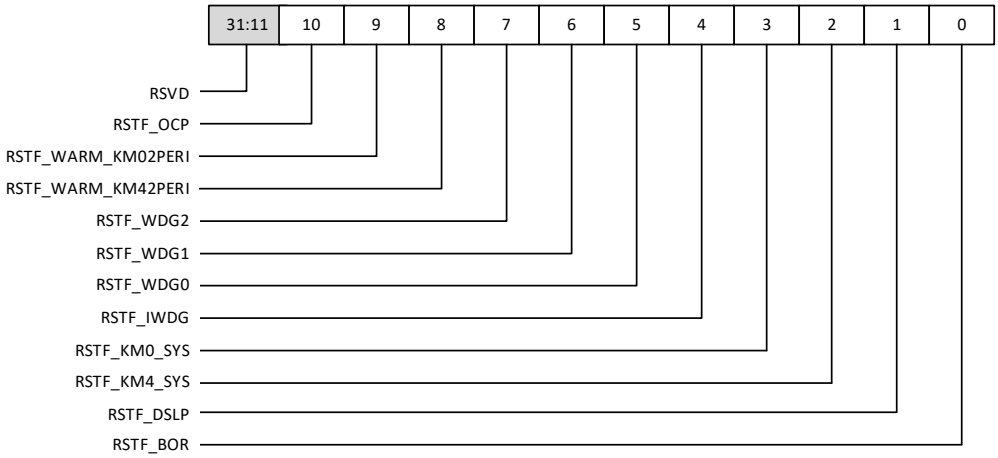
Bit	Symbol	Access	Description
31:0	TRIG_RST	R/WPD	System reset trigger register ● Writing 0x96969696 to enable the write of the REG_SW_RST_CTRL register. ● Writing 0x69696969 to trigger the system reset if the REG_SW_RST_CTRL register has been set.

9.3.2 Boot Reason Register

Each time the system boots up, software needs to know if any unexpected event triggered this boot. The boot reason only reflects the latest boot. The base address of this register is 0x4100_8000.

9.3.2.1 REG_BOOT_REASON_HW

- **Name:** Boot Reason Register
- **Size:** 32 bits
- **Address offset:** 0x20
- **Read/write access:** read



Bit	Symbol	Access	Reset	Description
31:11	RSVD	-	-	Reserved
10	RSTF_OCP	R/W1CB	0	This bit will be set to 1 when OCP happens, which will let system enter deep-sleep mode. Write 1 to clear.
9	RSTF_WARM_KM02PERI	R/W1CB	0	This bit will be set to 1 when KM0 warm resets the system. Write 1 to clear.
8	RSTF_WARM_KM42PERI	R/W1CB	0	This bit will be set to 1 when KM4 warm resets the system. Write 1 to clear.
7	RSTF_WDG2	R/W1CB	0	This bit will be set to 1 when WDG resets the system. Write 1 to clear.
6	RSTF_WDG1	R/W1CB	0	This bit will be set to 1 when WDG resets the system. Write 1 to clear.
5	RSTF_WDG0	R/W1CB	0	This bit will be set to 1 when WDG resets the system. Write 1 to clear.
4	RSTF_IWDG	R/W1CB	0	This bit will be set to 1 when WDG resets the system. Write 1 to clear.
3	RSTF_KM0_SYS	R/W1C	0	This bit will be set to 1 when KM0 resets the system. Write 1 to clear.
2	RSTF_KM4_SYS	R/W1C	0	This bit will be set to 1 when KM4 resets the system. Write 1 to clear.
1	RSTF_DSLP	R/W1C	0	This is boot reason for deep-sleep exit, cleared by SW writing to 1.
0	RSTF_BOR	R/W1C	0	This bit will be set to 1 when BOR resets the system. Write 1 to clear.

All information provided in this document is subject to legal disclaimers.
© REALTEK 2024. All rights reserved.

10 Inter-Processor Communication (IPC)

10.1 Introduction

There are two CPUs named KM4 and KM0 integrated into RTL8721Dx. The inter-processor communication (IPC) hardware is designed to make CPUs communicate with each other. The IPC provides a set of registers for each processor that facilitates inter-processor communication via interrupts. Interrupts may be independently masked by each processor to allow polled-mode operation.

The IPC communication data must be located in a common memory, which is not part of the IPC block.

10.2 Features

- Status signaling for the 32 channels (16 channels for Tx and 16 channels for Rx)
  - Channel empty/full flag, also used as a lock
- Two sets interrupt lines per processor
  - Two sets for Rx channel full (communication data posted by sending processors)
  - Two sets for Tx channel empty (communication data retrieved by receiving processors)
- Interrupt masking per channel
  - Channel Tx empty mask
  - Channel Rx full mask
- 64 hardware semaphores for atomic operation of shared resources

10.3 Architecture

The IPC provides a means for simple communication among CPUs. It can use shared SRAM to transmit information to each other. The system block diagram of IPC is depicted in *Figure 10-1*.

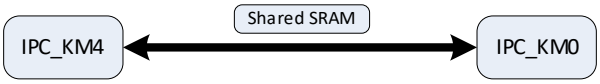


Figure 10-1 IPC block diagram

The IPC submodule's structure of each processor is the same. *Figure 10-2* summarizes the submodule's structure of IPC_KM0. Each processor has two sets of transmission channels, one for transmitting data (Tx) to another CPU, and one for receiving data (Rx) from another CPU. An Interrupt Status Register (ISR) is used to present current channel Tx empty or Rx full status change, an Interrupt Mask Register (IMR) is used to enable interrupts for Tx empty or Rx full, and an Interrupt Clear Register (ICR) is used to clear the corresponding interrupts. The submodule's structure of IPC_KM4 is similar.

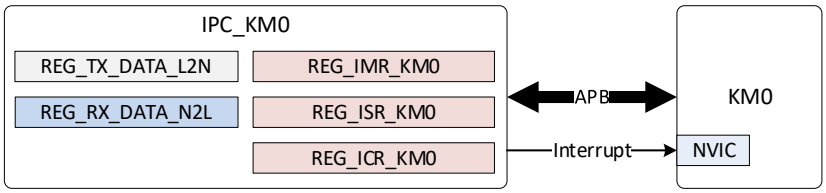


Figure 10-2 IPC submodule of KM0

10.4 Functional Description

10.4.1 Message Transmit

As there are two CPUs in this chip, each CPU has a set of Tx channels to send data to another CPU, and a set of Rx channels to receive requests from another CPU. Each set contains 16 channels. Channel index of Tx channels and the corresponding Rx channels are one-to-one mapping.

Figure 10-3 shows the IPC transmission mapping from KM0 to KM4, and *Figure 10-4* shows the IPC receive mapping of KM0 from KM4.

- REG_TX_DATA_L2N is used for KM0 to transmit data to KM4, once bit '1' is written to the channel x (x = 0, 1, …, 7), the corresponding bit in target REG_RX_DATA_L2Nx will be set. The Rx full bit of channel x in REG_ISR_KM4 will be set at the same time.

All information provided in this document is subject to legal disclaimers.
© REALTEK 2024. All rights reserved.

- Writing bit '1' to channel x in REG_ISR_KM4 clears the Rx full interrupt of channel x, the corresponding channel x Tx empty interrupt bit in REG_ISR_KM0 will be set automatically.

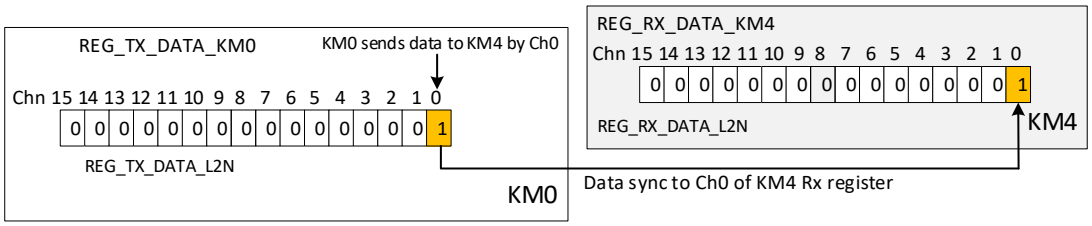


Figure 10-3 IPC message transmission mapping (Tx)

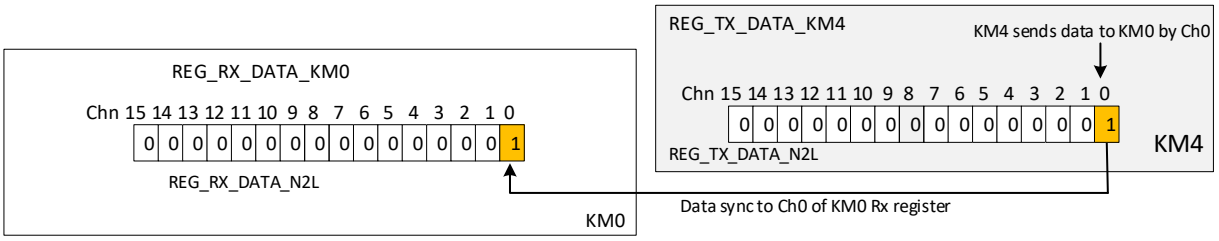


Figure 10-4 IPC message receive mapping (Rx)

10.4.2 Interrupts

10.4.2.1 Interrupt for Normal Usage

There are totally 32 receive and transmit interrupts with maskable interrupts for each processor. IPC NVIC of AP will be connected to NP for a quick power-resume issue.

If you want to transmit a message from CPUn to CPUm (n ≠ m), after preparing data in a common memory, write bit '1' to the channel x of REG_TX_DATA_n2m.

(1) The data will be mapped to the REG_RX_DATA_n2mx of CPUm.
(2) If REG_ISR_TX_EMPTY_INTR_STATUS_n2mx in REG_ISR_n is set, it will be cleaned automatically.
(3) The REG_ISR_RX_FULL_INTR_STATUS_n2mx in REG_ISR_m will be set automatically.
(4) If the REG_ISR_RX_FULL_INTR_MASK_n2mx is enabled in REG_IMR_m and IPC interrupt is enabled in NVIC, thus announce the IPC_IRQ interrupt to the CPUm.

After handling the message from CPUn, CPUm can write '1' to REG_ISR_RX_FULL_INTR_STATUS_n2mx in REG_ISR_m.

(1) The REG_ISR_RX_FULL_INTR_STATUS_n2mx in REG_ISR_m will be cleaned.
(2) The REG_ISR_TX_EMPTY_INTR_STATUS_n2mx in REG_ISR_n will be set automatically.
(3) If the REG_ISR_TX_EMPTY_INTR_MASK_n2mx is enabled in REG_IMR_n and IPC interrupt is enabled in NVIC, thus announce the IPC_IRQ interrupt to the CPUn.
(4) The channel x bit in REG_RX_DATA_n2m and REG_TX_DATA_n2m will be cleaned automatically.

10.4.2.2 Interrupt for Wakeup

In RTL8721Dx, the power of AP is controlled by NP, and NP can be KM4 or KM0 according to user's configuration. Only NP knows the status of AP. If an wakeup interrupt is triggered in AP and AP is in sleep mode at this time, it is necessary to notify NP to power on AP. The fastest way is to connect the interrupt of IPC_AP and other AP wakeup INT to the AP wake mask of NP and enable it. When IPC_AP receives the data to trigger the interrupt, NP will immediately enter the IRQ of AP_WAKE to power on AP, and then AP can process the received message. The same for KM4. *Figure 10-5* illustrates the IPC interrupt requests.

All information provided in this document is subject to legal disclaimers.
© REALTEK 2024. All rights reserved.

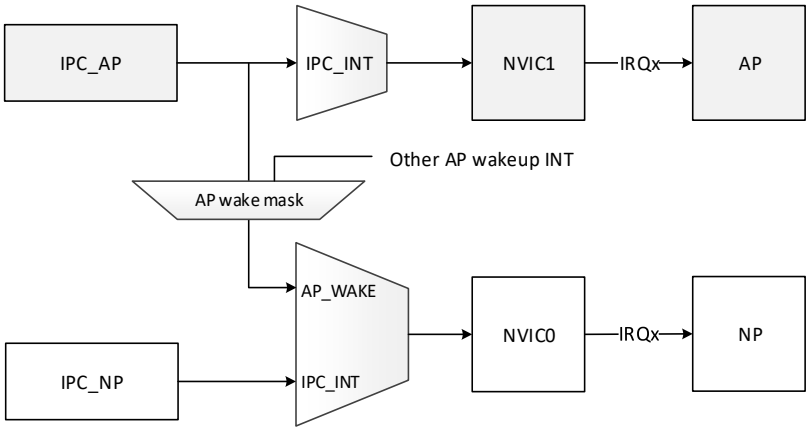


Figure 10-5 IPC interrupt request

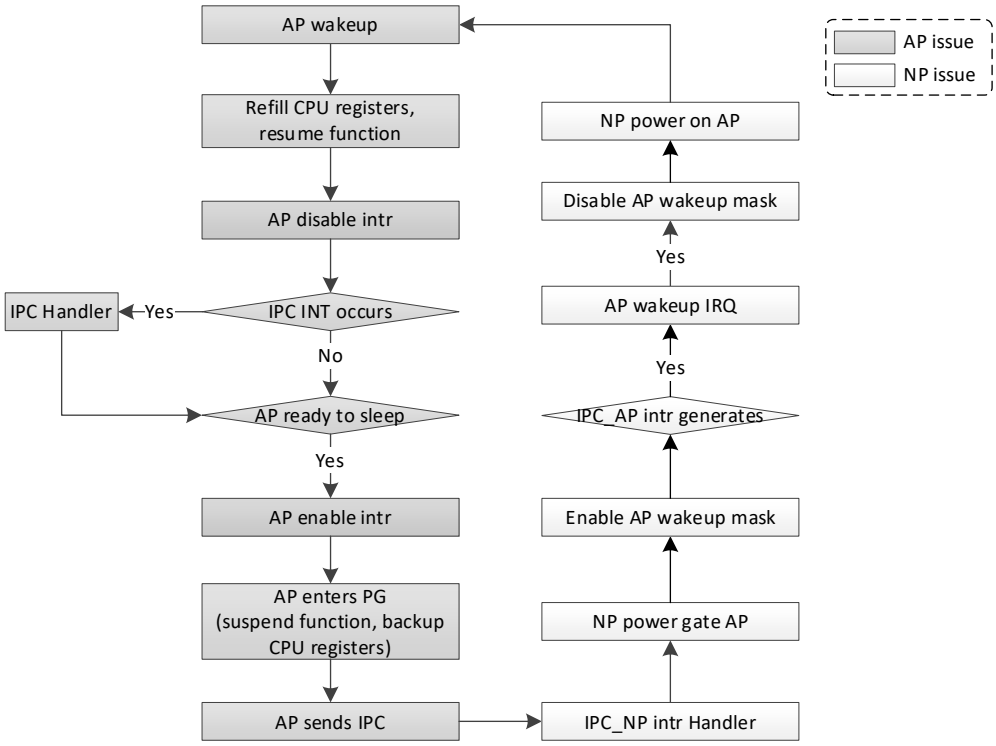


Figure 10-6 IPC interrupt software handler flow

10.4.3 General Usage

10.4.3.1 IPC without Handshake

As there are Rx full and Tx empty interrupts for each processor, these functions can be used to realize message transmission and handshake. *Figure 10-7* shows the interrupt messaging protocol of IPC without waiting for a handshake from receiver. It takes the message transmission from KM0 to KM4 for example.

Table 10-1 IPC message transmission steps without handshake

Step	Description
①	Check the Tx data status of the channel 0 in KM0 IPC Tx Data Register
②	KM0 writes data to the Common Memory
③	Write '1' to idle channel 0 in IPC Tx Data Register
④	The KM0 IPC Tx Data Register write operation ● Map to KM4 IPC Rx Data Register ● Clear the Tx empty interrupt bit in the KM0 Interrupt Status Register

All information provided in this document is subject to legal disclaimers.
© REALTEK 2024. All rights reserved.

	● Set the Rx full bit in the KM4 Interrupt Status Register
⑤	If the Rx full interrupt mask is enabled in KM4 Interrupt Mask Register, a Rx full interrupt request will be generated to KM4.
⑥	After checking Rx full channel, KM4 performs a data read from the corresponding Common Memory
⑦	After finishing the interrupt handler, KM4 writes '1' to KM4 Interrupt Status Register to clear the Rx full interrupt
⑧	The KM4 Interrupt Status Register write operation ● Clear the Rx full interrupt status bit in the KM4 Status Register ● Set Tx empty interrupt status bit in the KM0 Status Register
⑨	Clear the channel 0 bit in KM4 IPC Rx Data Register and KM0 IPC Tx Data Register automatically

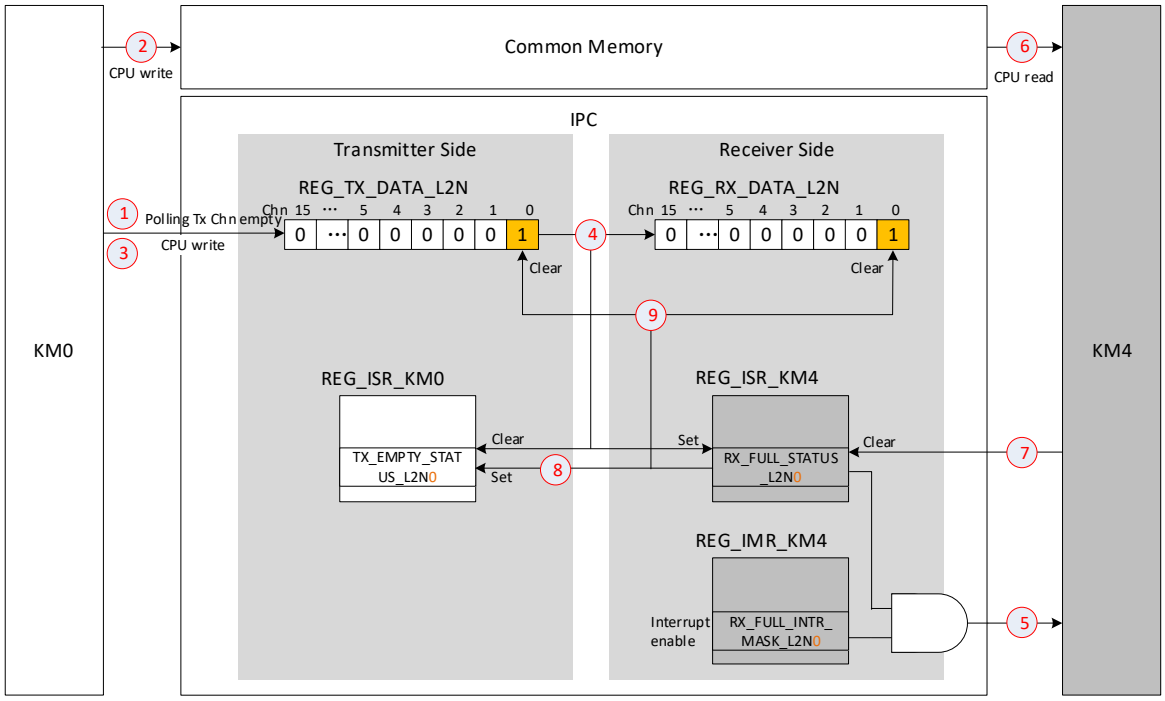


Figure 10-7 IPC message transmission flow without handshake

10.4.3.2 IPC with Handshake

Figure 10-8 shows the interrupt messaging protocol of IPC with waiting for a handshake from receiver. It takes the message transmission from KM0 to KM4 for example.

Table 10-2 IPC message transmission steps with handshake

Step	Description
①	Check the Tx data status of the channel 0 in KM0 IPC Tx Data Register
②	KM0 writes data to the Common Memory
③	Write '1' to idle channel 0 in IPC Tx Data Register.
④	The KM0 IPC Tx Data Register write operation ● Map to KM4 IPC Rx Data Register ● Set the Rx full bit in the KM4 Interrupt Status Register
⑤	If the Rx full interrupt mask is enabled in KM4 Interrupt Mask Register, a Rx full interrupt request will be generated to KM4.
⑥	After checking Rx full channel, KM4 performs a data read from the corresponding Common Memory
⑦	After finishing the interrupt handler, KM4 write '1' to KM4 Interrupt Status Register to clear the Rx full interrupt
⑧	The IPC Interrupt Status Register write operation ● Clear the Rx full interrupt status bit in the KM4 Status Register ● Set Tx empty interrupt status bit in the KM0 Status Register
⑨	Clear the channel 0 bit in KM4 IPC Rx Data Register and KM0 IPC Tx Data Register automatically
⑩	If Tx empty mask is enabled in KM0 Interrupt Mask Register, the setting of the Tx empty bit in the KM0 Status Register generates a Tx Empty interrupt to KM0
⑪	KM0 writes '1' to KM0 Interrupt Status Register to clear the Tx full interrupt

All information provided in this document is subject to legal disclaimers.
© REALTEK 2024. All rights reserved.

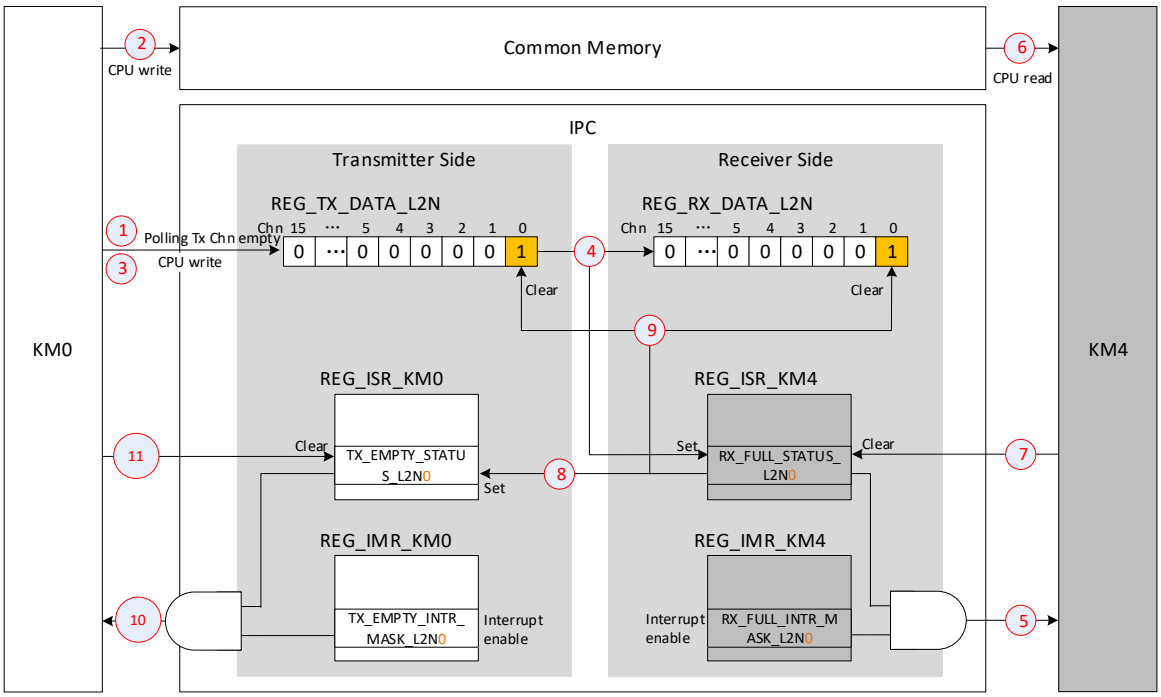


Figure 10-8 IPC message transmission flow with handshake

10.4.4 Hardware Semaphore

10.4.4.1 Overview

The hardware semaphore is used for a mutual exclusion mechanism. It can be accessed by the cores in an atomic operation.

Traditionally, the software takes three steps to use the semaphore: read → write → read.

(1) The first read operation of the CPU is to poll the current semaphore to be competed whether it is occupied.
(2) If this semaphore is free, the CPU will compete it by writing.
(3) Another read operation is to check that whether the CPU has already occupied this semaphore because maybe serval CPUs or threads write to compete the same semaphore.

In the RTL8721Dx, the above flow executed by software will be offloaded to hardware. *Figure 10-9* illustrates the hardware semaphore mechanism.

- The address mapping is responsible for mapping the 64 bit semaphore status to 64*4 address space.
- The read set logic is responsible for determining the status of the semaphore to be read after receiving the read request. If the corresponding semaphore register is 0, the read set logic will set the semaphore register first, then return 0 to the read request.
- The write clear logic is responsible for clearing the status of the semaphore to 0 after receiving write request.
- The semaphore register has 64 bits, and each bit in the register represents a semaphore. This register can be read directly, or read/written through the address mapped register.
  - If the semaphore register is read through the address mapped register, the status of only one bit can be read at a time, and this bit will be set synchronously when it is 0.
  - If the semaphore register is read directly, the read does not change the status of the semaphore, and all the 32 bits can be read at a time.

All information provided in this document is subject to legal disclaimers.
© REALTEK 2024. All rights reserved.

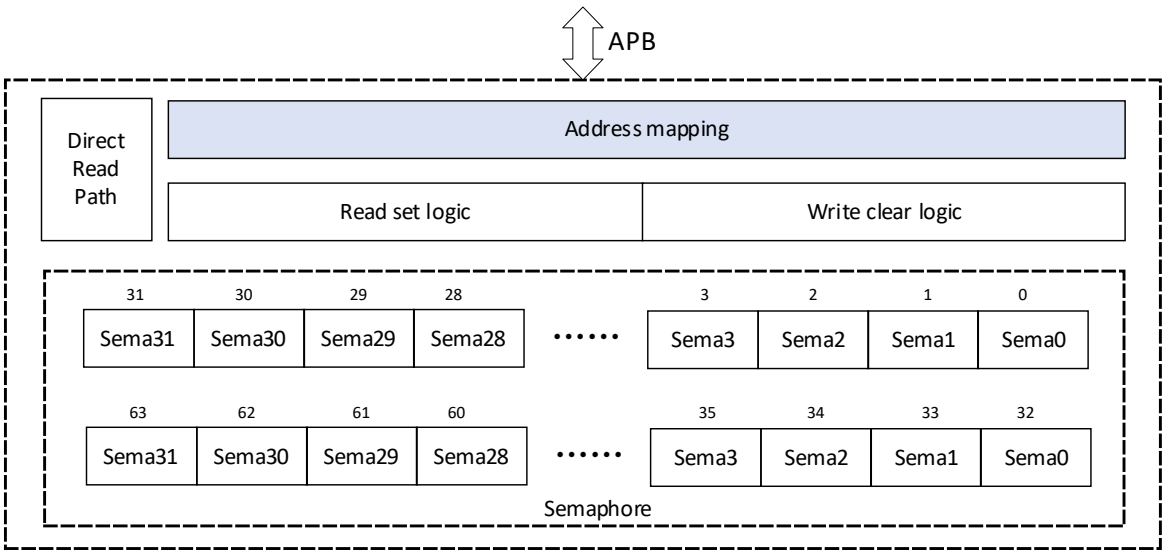


Figure 10-9 IPC hardware semaphore mechanism

10.4.4.2 Address Mapping

If the bit[0] of the semaphore register can be acquired and released through the base address of address mapping, the bit[1] can be acquired and released through (base address +4) accordingly. Thus the address of the entire access window ranges from 0 to (64*4 - 1).

10.4.4.3 Acquire Flow

If the software wants to require the semaphore 0, it can directly read the base address of the access window register.

- If the bit is 0 when the hardware receives the read request, the hardware will set this bit and return 0 synchronously. At this time, the acquire flow will be regarded as successful by software.
- If the bit is 1 when the hardware receives the read request, the hardware will directly return 1. At this time, the acquire flow will be regarded as failed by software.

10.4.4.4 Release Flow

If the software wants to release the semaphore 0, it can directly write the base address of the access window register. The hardware will clear this bit after receiving the write command.

10.4.4.5 Inquire Flow

If the software wants to inquire the semaphore status, it can directly read the semaphore registers (without address mapping). This operation only returns the semaphore status, but does not change it.

10.5 Registers

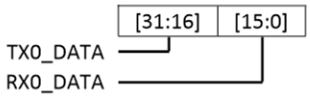
Name	Address offset	Access	Description
REG_IPC_DATA	000h	R/W	Tx_Rx Register
REG_IPC_ISR	004h	R/W	Interrupt Empty Full Status Register
REG_IPC_IMR	008h	R/W	Interrupt Empty Full Mask Register
REG_IPC_ICR	00Ch	R/W	Clear Tx Register
REG_IPC_SEM_0	010h	R	Semaphore Register
REG_IPC_SEM_1	014h	R	Semaphore Register
REG_IPC_DUMMY	018h	R/W	Dummy Register

All information provided in this document is subject to legal disclaimers.
© REALTEK 2024. All rights reserved.

10.5.1 REG_IPC_DATA

- **Name:** Tx_Rx Register
- **Size:** 32 bits
- **Address offset:** 000h
- **Read/write access:** R/W

Take NP's IPC as an example to describe the function of IPC's internal registers. The REG_TX_DATA is used to transmit message from NP to AP. (The logic cannot be generated automatically)

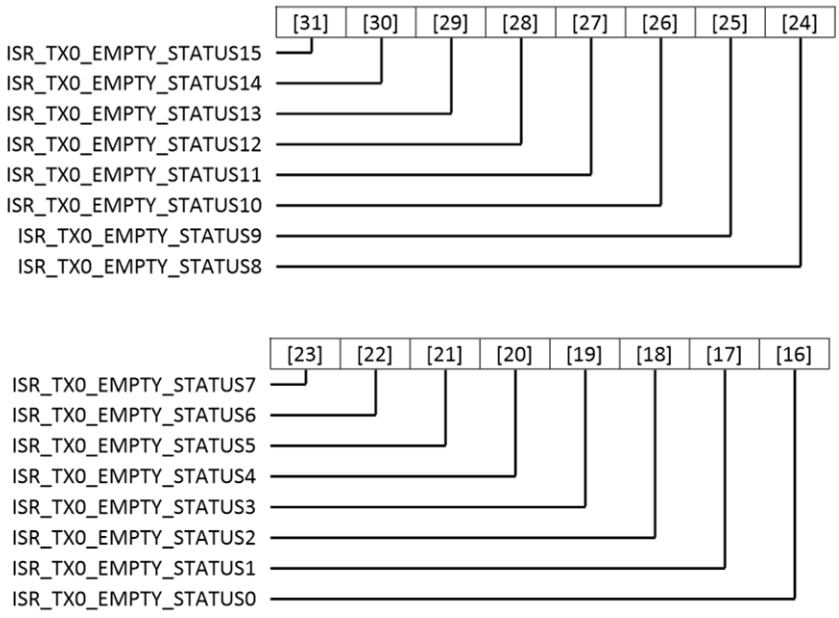


Bit	Symbol	Access	Reset	Description
31:16	TX0_DATA	R/WH/ES	0x0	After preparing descriptor, data and corresponding memory. 1: Writing 1 to the Tx data register (tx0_data) channel x bit, and the data will be mapped to the CPU0's RX data register (rx0_data) channel x bit. And the corresponding Rx full status bit in REG_ISR_CPU0 will be set. 0: Writing 0 has no effect. If the corresponding Rx full status bit in REG_ISR_CPU0 (e.g. isr_rx0_full_status6) is cleared, the corresponding bit in tx0_data will be cleaned automatically.
15:0	RX0_DATA	R	0x0	Rx data automatically maps data from tx0_data of CPU0

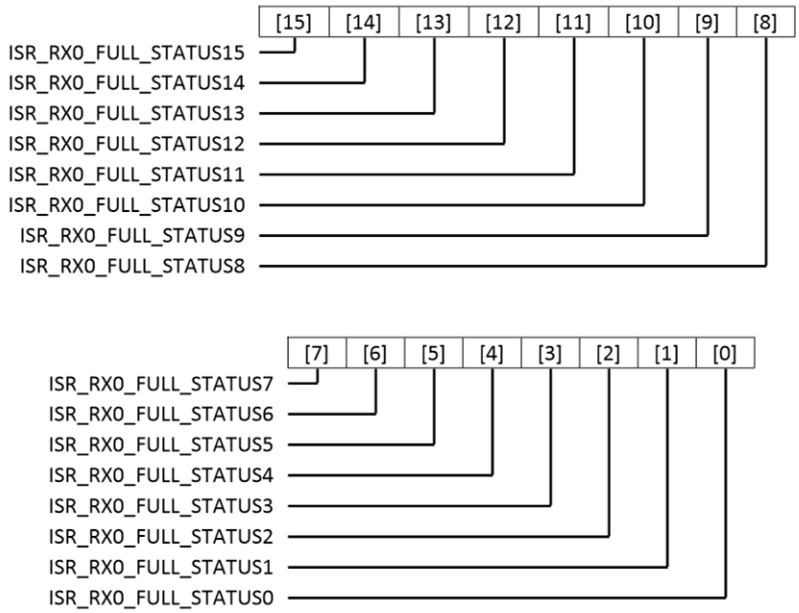
10.5.2 REG_IPC_ISR

- **Name:** Interrupt Empty Full Status Register
- **Size:** 32 bits
- **Address offset:** 004h
- **Read/write access:** R/W

This register is used to record Tx empty.



All information provided in this document is subject to legal disclaimers.
© REALTEK 2024. All rights reserved.

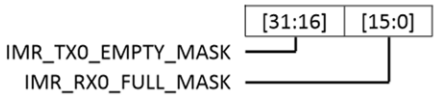


Bit	Symbol	Access	Reset	Description
31	ISR_TX0_EMPTY_STATUS15	RW1CB	0x0	Refer to the description of isr_tx0_empty_status0
30	ISR_TX0_EMPTY_STATUS14	RW1CB	0x0	Refer to the description of isr_tx0_empty_status0
29	ISR_TX0_EMPTY_STATUS13	RW1CB	0x0	Refer to the description of isr_tx0_empty_status0
28	ISR_TX0_EMPTY_STATUS12	RW1CB	0x0	Refer to the description of isr_tx0_empty_status0
27	ISR_TX0_EMPTY_STATUS11	RW1CB	0x0	Refer to the description of isr_tx0_empty_status0
26	ISR_TX0_EMPTY_STATUS10	RW1CB	0x0	Refer to the description of isr_tx0_empty_status0
25	ISR_TX0_EMPTY_STATUS9	RW1CB	0x0	Refer to the description of isr_tx0_empty_status0
24	ISR_TX0_EMPTY_STATUS8	RW1CB	0x0	Refer to the description of isr_tx0_empty_status0
23	ISR_TX0_EMPTY_STATUS7	RW1CB	0x0	Refer to the description of isr_tx0_empty_status0
22	ISR_TX0_EMPTY_STATUS6	RW1CB	0x0	Refer to the description of isr_tx0_empty_status0
21	ISR_TX0_EMPTY_STATUS5	RW1CB	0x0	Refer to the description of isr_tx0_empty_status0
20	ISR_TX0_EMPTY_STATUS4	RW1CB	0x0	Refer to the description of isr_tx0_empty_status0
19	ISR_TX0_EMPTY_STATUS3	RW1CB	0x0	Refer to the description of isr_tx0_empty_status0
18	ISR_TX0_EMPTY_STATUS2	RW1CB	0x0	Refer to the description of isr_tx0_empty_status0
17	ISR_TX0_EMPTY_STATUS1	RW1CB	0x0	Refer to the description of isr_tx0_empty_status0
16	ISR_TX0_EMPTY_STATUS0	RW1CB	0x0	Tx channel 0 empty interrupt status of CPU1 transmit to CPU0. If the corresponding Rx full status bit in REG_ISR_CPU0 (e.g. isr_rx0_full_status0) is cleared, the corresponding Tx empty status bit will be set automatically. It will be cleared by software writing 1.
15	ISR_RX0_FULL_STATUS15	RW1CB	0x0	Refer to the description of isr_rx0_full_status0
14	ISR_RX0_FULL_STATUS14	RW1CB	0x0	Refer to the description of isr_rx0_full_status0
13	ISR_RX0_FULL_STATUS13	RW1CB	0x0	Refer to the description of isr_rx0_full_status0
12	ISR_RX0_FULL_STATUS12	RW1CB	0x0	Refer to the description of isr_rx0_full_status0
11	ISR_RX0_FULL_STATUS11	RW1CB	0x0	Refer to the description of isr_rx0_full_status0
10	ISR_RX0_FULL_STATUS10	RW1CB	0x0	Refer to the description of isr_rx0_full_status0
9	ISR_RX0_FULL_STATUS9	RW1CB	0x0	Refer to the description of isr_rx0_full_status0
8	ISR_RX0_FULL_STATUS8	RW1CB	0x0	Refer to the description of isr_rx0_full_status0
7	ISR_RX0_FULL_STATUS7	RW1CB	0x0	Refer to the description of isr_rx0_full_status0
6	ISR_RX0_FULL_STATUS6	RW1CB	0x0	Refer to the description of isr_rx0_full_status0
5	ISR_RX0_FULL_STATUS5	RW1CB	0x0	Refer to the description of isr_rx0_full_status0
4	ISR_RX0_FULL_STATUS4	RW1CB	0x0	Refer to the description of isr_rx0_full_status0
3	ISR_RX0_FULL_STATUS3	RW1CB	0x0	Refer to the description of isr_rx0_full_status0
2	ISR_RX0_FULL_STATUS2	RW1CB	0x0	Refer to the description of isr_rx0_full_status0
1	ISR_RX0_FULL_STATUS1	RW1CB	0x0	Refer to the description of isr_rx0_full_status0
0	ISR_RX0_FULL_STATUS0	RW1CB	0x0	Rx channel 0 full interrupt status of CPU1. The corresponding Rx full status bit will be set by CPU0's Tx data register (tx0_data) channel 0. It will be cleared by software writing 1.

All information provided in this document is subject to legal disclaimers.
© REALTEK 2024. All rights reserved.

10.5.3 REG_IPC_IMR

- **Name:** Interrupt Empty Full Mask Register
- **Size:** 32 bits
- **Address offset:** 008h
- **Read/write access:** R/W

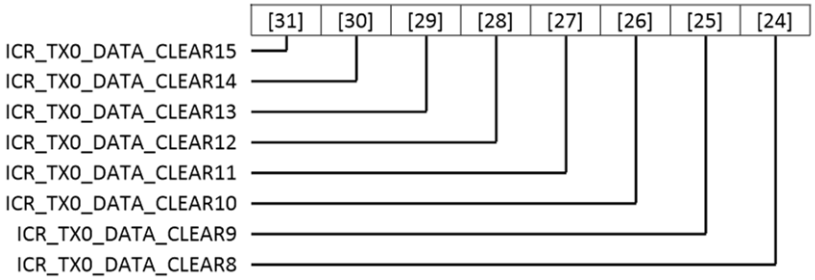


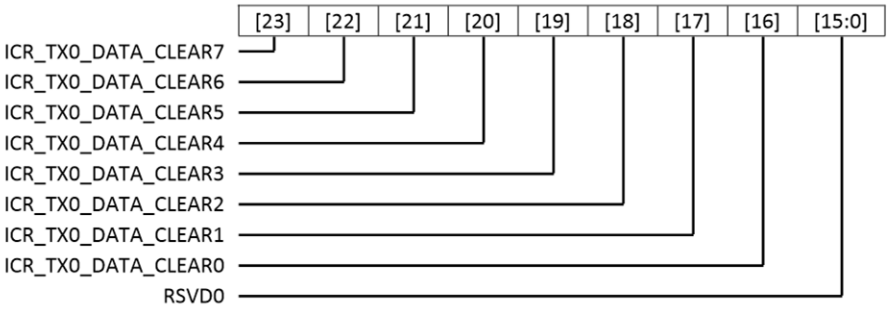
Bit	Symbol	Access	Reset	Description
31:16	IMR_TX0_EMPTY_MASK	R/W	0x0	0: Mask Tx Channel x empty interrupt of CPU1 transmit to CPU0 1: Unmask Tx Channel x empty interrupt of CPU1 transmit to CPU0
15:0	IMR_RX0_FULL_MASK	R/W	0x0	0: Mask Rx Channel x full interrupt of CPU1 received from CPU0 1: Unmask Rx Channel x full interrupt of CPU1 received from CPU0

10.5.4 REG_IPC_ICR

- **Name:** Clear Tx Register
- **Size:** 32 bits
- **Address offset:** 00Ch
- **Read/write access:** R/W

This register can ensure that the software can manually clear the Tx register in an emergency.





Bit	Symbol	Access	Reset	Description
31	ICR_TX0_DATA_CLEAR15	RW1CB	0x0	Refer to the description of icr_tx0_data_clear0
30	ICR_TX0_DATA_CLEAR14	RW1CB	0x0	Refer to the description of icr_tx0_data_clear0
29	ICR_TX0_DATA_CLEAR13	RW1CB	0x0	Refer to the description of icr_tx0_data_clear0
28	ICR_TX0_DATA_CLEAR12	RW1CB	0x0	Refer to the description of icr_tx0_data_clear0
27	ICR_TX0_DATA_CLEAR11	RW1CB	0x0	Refer to the description of icr_tx0_data_clear0
26	ICR_TX0_DATA_CLEAR10	RW1CB	0x0	Refer to the description of icr_tx0_data_clear0
25	ICR_TX0_DATA_CLEAR9	RW1CB	0x0	Refer to the description of icr_tx0_data_clear0
24	ICR_TX0_DATA_CLEAR8	RW1CB	0x0	Refer to the description of icr_tx0_data_clear0
23	ICR_TX0_DATA_CLEAR7	RW1CB	0x0	Refer to the description of icr_tx0_data_clear0

All information provided in this document is subject to legal disclaimers.
© REALTEK 2024. All rights reserved.

22	ICR_TX0_DATA_CLEAR6	RW1CB	0x0	Refer to the description of icr_tx0_data_clear0
21	ICR_TX0_DATA_CLEAR5	RW1CB	0x0	Refer to the description of icr_tx0_data_clear0
20	ICR_TX0_DATA_CLEAR4	RW1CB	0x0	Refer to the description of icr_tx0_data_clear0
19	ICR_TX0_DATA_CLEAR3	RW1CB	0x0	Refer to the description of icr_tx0_data_clear0
18	ICR_TX0_DATA_CLEAR2	RW1CB	0x0	Refer to the description of icr_tx0_data_clear0
17	ICR_TX0_DATA_CLEAR1	RW1CB	0x0	Refer to the description of icr_tx0_data_clear0
16	ICR_TX0_DATA_CLEAR0	RW1CB	0x0	0: Writing 0 has no effect. 1: Clear the Tx data register (tx0_data) channel 0 bit.
15:0	RSVD0	R/W	0x0	

10.5.5 REG_IPC_SEM_0

- **Name:** Semaphore Register
- **Size:** 32 bits
- **Address offset:** 010h
- **Read/write access:** R



Bit	Symbol	Access	Reset	Description
31:0	SEM_0_DATA	R	0x0	Used to indicate who owns this semaphore.

10.5.6 REG_IPC_SEM_1

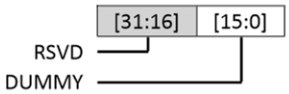
- **Name:** Semaphore Register
- **Size:** 32 bits
- **Address offset:** 014h
- **Read/write access:** R



Bit	Symbol	Access	Reset	Description
31:0	SEM_1_DATA	R	0x0	Used to indicate who owns this semaphore.

10.5.7 REG_IPC_DUMMY

- **Name:** Dummy Register
- **Size:** 32 bits
- **Address offset:** 018h
- **Read/write access:** R/W



Bit	Symbol	Access	Reset	Description
31:16	RSVD	-	0	Reserved
15:0	DUMMY	R/W	0x0	Dummy register

All information provided in this document is subject to legal disclaimers.
© REALTEK 2024. All rights reserved.

11 I/O Control

11.1 Features

The following electrical properties are configurable for standard I/O pins:

- Function ID
- Pull-up/Pull-down Resistor
- Driving strength
- Slew rate control
- Schmitt trigger
- Shutdown & RESET
- Open drain mode

11.2 Functional Description

The I/O diagram is given in *Figure 11-1*. There are many kinds of I/Os in the RTL8721Dx, different I/Os have different configurations.

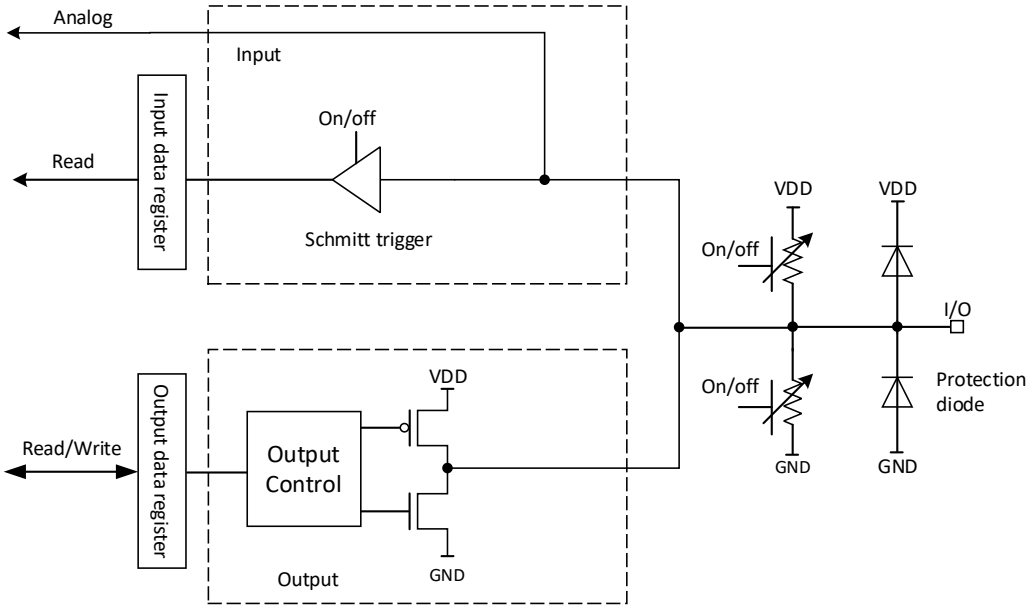


Figure 11-1 I/O diagram

11.2.1 I/O Types

All I/Os are listed in *Table 11-1*. The Human Body Model (HBM) of all IOs are above 3.5KV.

Table 11-1 I/O Types

Pin name	I/O power pin	Driving (mA)[1]		Internal pull resistor (ohm)[2]			Resistor available in deep sleep mode?
		1.8V (±10%)	3.3V (±10%)	Min.	Typ.	Max.	
PA0~PA14	VAH_LDOM	4/8	8/16	40K	80K	120K	Yes
PA12	VAH_LDOM	2/4	4/8	40K	80K	120K	Yes
PA13~PA18	VDH_IO1	4/8	8/16	40K	80K	120K	Yes
PA19~PA25	VDH_IO1	4/8	8/16	2.3K/25K	4.7K/50K	7.1K/75K	Yes
PA26~PA27, PB30~PB3	VDH_IO2	4/8	8/16	2.3K/25K	4.7K/50K	7.1K/75K	Yes
PA28~PA29	VDH_IO2	2/4	4/8	PU:1.1K/25K PD:10.75K	PU:2.2K/50K PD:21.5K	PU:3.3K/75K PD:32.25K	Yes
PB4~PB12	VDH_IO3[3]	4/8	8/16	2.3K/25K	4.7K/50K	7.1K/75K	Yes
PB13~PB19	VDH_IO3	2/4	4/8	40K	80K	120K	Yes
PB20~PB28	VDH_IO3	2/4	4/8	40K	80K	120K	Yes
PB30~PB31	VDH_IO3	4/8	8/16	40K	80K	120K	Yes

All information provided in this document is subject to legal disclaimers.
© REALTEK 2024. All rights reserved.

NOTE

1. *The I/O power supported by different IOs is different, and the driving capability is related to the I/O power.*
2. *The pull up and pull down values of other IOs in the table are the typical values at 3.3V. The values at 1.8V are twice the typical values at 3.3V. The range is ±50%.*
3. *All groups support both 3.3V and 1.8V I/O power.*
4. *The typical voltage of VDH_IO3 can be 1.8V or 3.3V, and is related to the operating voltage of the chip. It is required to be no lower than the operating voltage of the chip. For example, if the operating voltage of the chip is required to be 1.8V, the typical voltage of VDH_IO3 can be 1.8V or 3.3V; if the operating voltage of the chip is required to be 3.3V, the typical voltage of VDH_IO3 cannot be 1.8V, but only be 3.3V.*

11.2.2 IOCTRL Register

Each I/O pin has one IOCTRL register assigned to control the pin's electrical characteristics. I/O Register base address 0x41008800. For example, the register address of PA0 is 0x41008800, the register address of PA1 is 0x41008804. The bit[4:0] field in the I/O register can be set to GPIO (typically value 00000) or a special function.

- For pins set to GPIO, the GPIO IP registers determine whether the pin is configured as input or output.
- For any special function, the pin direction is controlled automatically depending on the function.
- For specific information about PINMUX function, please refer to the PINMUX documentation.

Table 11-2 IOCTRL register

Offset	Bit	Access	INI	Symbol	Description
0000h	[15]	R/W	0h	gpio@_dis	GPIO shutdown, only when the gpio@_dis signals of this group of GPIOs are all 1, the group of GPIOs will be shutdown. 1. disable 0. enable
	[13]	R/W	0h	gpio@_sr[1]	GPIO slew rate control. 0: fast slew rate 1: low slew rate
	[12]	R/W	1h	gpio@_smt	GPIO Schmitt control
	[11]	R/W	1h	gpio@_e2	GPIO driving strength control. 0: low 1: high
	[10]	R/W	0h	gpio@_pupdc	Some GPIO may have two types of PU/PD resistors, this bit can select it. 1: small resistor 0: big resistor
	[8]	R/W	1h	gpio@_ie[2]	GPIO input enable control.
	[4:0]	R/W/ES	0h	gpio@_sel	GPIO PINMUX function id select

NOTE

1. *This bit is valid except for PA0-PA14.*
2. *This bit is only valid for PA12&PB13~PB19.*

IO internal pull up and pull down are controlled by separate registers. The 32 bits of each register control the status of 32 I/Os respectively. Refer to *Table 11-3* for details.

Table 11-3 I/O pull up & pull down register

Offset	Bit	Access	INI	Symbol	Description
100h	[31:0]	R/W	40022h	PA_PU	PA0~PA31 pull up enable when system is in active.
104h	[31:0]	R/W	C0000020h	PB_PU	PB0~PB31 pull up enable when system is in active.
108h	[31:0]	R/W	0h	PA_PD	PA0~PA31 pull down enable when system is in active.
10Ch	[31:0]	R/W	0h	PB_PD	PB0~PB31 pull down enable when system is in active.
110h	[31:0]	R/W	0h	PA_PU_SLP	PA0~PA31 pull up enable when system is in sleep.
114h	[31:0]	R/W	0h	PB_PU_SLP	PB0~PB31 pull up enable when system is in sleep.
118h	[31:0]	R/W	0h	PA_PD_SLP	PA0~PA31 pull down enable when system is in sleep.
11Ch	[31:0]	R/W	0h	PB_PD_SLP	PB0~PB31 pull down enable when system is in sleep.

11.2.3 Function ID

Change the I/O function ID through gpio@_sel in *Table 11-2*, refer to the PINMUX documentation for details.

All information provided in this document is subject to legal disclaimers. © REALTEK 2024. All rights reserved.

11.2.4 I/O Pull Resistor Control

The PU/PD of the I/O in Active & Sleep mode is controlled by the PU&PD register. Allows selection of on-chip pull-up or pull-down resistors for each pin. For Specific information, please refer to the PU&PD register table.

Pull-up and Pull-down resistor's value is different between different I/Os. The resistor value is changed by controlling the gpio@_pupdc of register table (*Table 11-2*). For I/Os with only one resistance value, gpio@_pupdc is not valid.

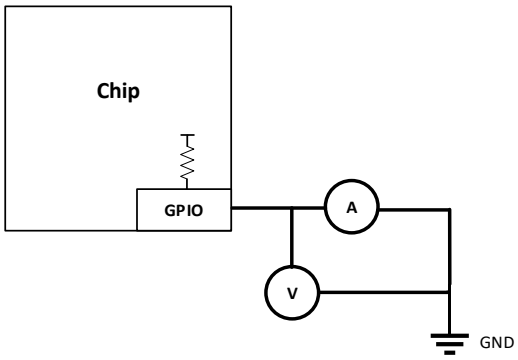


Figure 11-2 I/O pull-up resistor test

I/O pull-up resistor test procedure:
(1) Configure the GPIO to be tested as input mode.
(2) Configure the GPIO registers to be tested as 1 for gpio@_pu and 0 for gpio@_pd.
(3) For GPIOs with multiple pull-up resistance values, it is necessary to change the configuration of the register gpio@_pupdc and test them separately.
(4) When the ammeter is not connected, measure the voltage of the GPIO pin to be tested, and the voltage value is V_1.
(5) According to *Figure 11-2*, only connect the ammeter to test. At this time, the current value is I_1, and the pull-up resistor is $R_{pu}=V_1/I_1$.

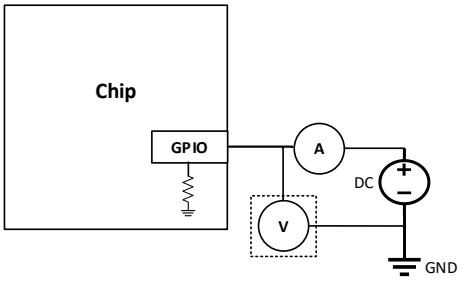


Figure 11-3 I/O pull-down resistor test

I/O pull-down resistor test procedure:
(1) Configure the GPIO to be tested as input mode.
(2) Configure the GPIO registers to be tested as 0 for gpio@_pu and 1 for gpio@_pd.
(3) For GPIOs with multiple pull-down resistance values, it is necessary to change the configuration of the register gpio@_pupdc and test them separately.
(4) When the ammeter is not connected, measure the voltage of the GPIO pin to be tested, and the voltage value is 0V.
(5) According to *Figure 11-3*, use an external power supply to provide the same voltage (V_2) as the GPIO power, and measure the current (I_2) at this time, and the pull-up resistor is $R_{pd}=V_2/I_2$.

11.2.5 I/O Driving Strength

The I/O driving strength can be configured through gpio@_e2 in the IOCTRL register. I/O driving strength is different between different I/O types.

The GPIO output high driving strength test procedure:
(1) Configure the GPIO to be tested as Output mode.
(2) Connect the power supply that can be poured back to the pin of the GPIO to be tested, and set the power supply voltage to 0.85*VDDIO.
(3) Set the driving Strength to high or low through gpio@_e2.
(4) I_{HIGH} is the driving strength of GPIO output high.

All information provided in this document is subject to legal disclaimers.
© REALTEK 2024. All rights reserved.

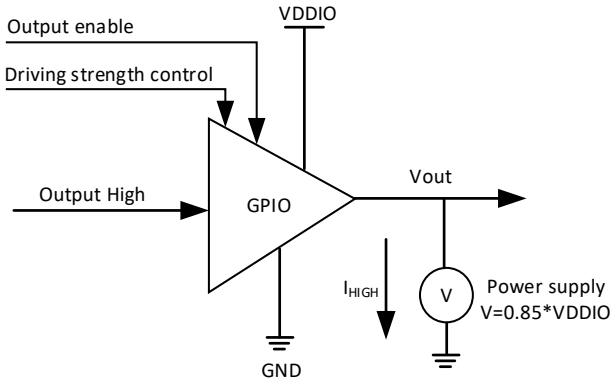


Figure 11-4 GPIO output high driving strength test

GPIO output low driving strength test procedure:

(1) Configure the GPIO to be tested as Output mode.
(2) Connect the power supply that can be poured back to the pin of the GPIO to be tested, and set the power supply voltage to 0.15*VDDIO.
(3) Set the driving Strength to high or low through gpio@_e2.
(4) I_{LOW} is the driving strength of GPIO output Low.

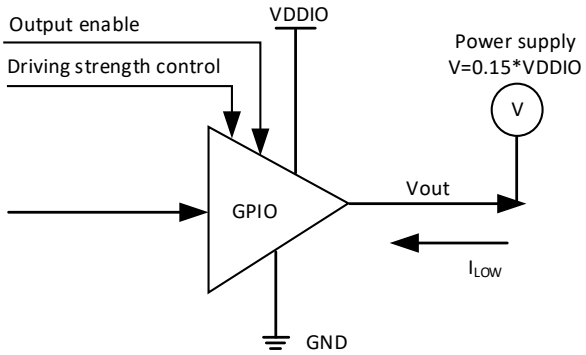


Figure 11-5 GPIO output low driving strength test

11.2.6 I/O Schmitt Trigger

The I/O pin contains a schmitt trigger as a digital function, which can be selectively disabled by setting gpio@_smt in the IOCTRL register. The specifications of the schmitt trigger are shown in *Table 11-4*.

Table 11-4 Digital I/O pin DC characteristics

Symbol	Parameter	Conditions	Min.	Typ.	Max.	Unit
V_{IL}	I/O input low level voltage	V_{IO}=1.8V±10%	-0.3	-	0.35*V_{IO}	V
		V_{IO} =3.3V±10%	-0.3	-	0.8	
V_{IH}	I/O input high level voltage	V_{IO}=1.8V±10%	0.65*V_{IO}	-	-	
		V_{IO}=3.3V±10%	2	-	-	
V_{OL}	I/O output Low level voltage	V_{IO}=1.8V±10%, I_{OL} Max.	-	-	0.15*V_{IO}	
		V_{IO}=3.3V±10%, I_{OL} Max.	-	-	0.15*V_{IO}	
V_{OH}	I/O output high level voltage	V_{IO}=1.8V±10%, I_{OH} Max.	0.85*V_{IO}	-	-	
		V_{IO}=3.3V±10%, I_{OH} Max.	0.85*V_{IO}	-	-	

NOTE

- *V_{IO} is the power supply for I/O pin.*
- *If the schmitt trigger is disabled in IOCTRL register, V_{IH}=V_{IL}=0.5*V_{IO}.*
- *Refer to Table 11-1 for I/O driving strength.*

11.2.7 Slew Rate Control

Each IO port has independent slew rate control function except for PA0~PA14. Refer to *Table 11-2* for specific control settings.

All information provided in this document is subject to legal disclaimers.
© REALTEK 2024. All rights reserved.

Different slew rate control settings will mainly affect the rise/fall time corresponding to IO port output high/low. When the user's application scenario is sensitive to EMI and there is a problem with the default setting (faster rise/fall time), users can adjust the slew rate corresponding to the IO port to a low gear (slower rise/fall time).

Refer to for the impact of different slew rate settings on rise/fall time under the same external loading conditions.

Table 11-5 Effect of different slew rate settings on rise/fall time

Pin name	Conditions	Rise/fall time typical value (ns)			
		SR =0 I/O power = 3.3V	SR =0 I/O power = 1.8V	SR =1 I/O power = 3.3V	SR =1 I/O power = 1.8V
PA0~PA14	Loading of I/O port = 15pF	0.7/0.7	1.2/1.2	-	-
PA12		1.4/1.4	2.6/2.5	4.1/4.2	8.3/9.0
PA13~PA18		0.7/0.7	1.3/1.3	5.0/4.8	10.9/10.9
PA19~PA25		0.8/0.9	1.5/1.4	4.7/4.8	10.2/10.9
PA26~PA27		0.8/0.9	1.5/1.4	4.7/4.8	10.3/11.0
PA28~PA29		1.2/1.3	2.2/2.3	4.3/4.5	8.8/9.6
PB30~PB3		0.8/0.9	1.5/1.4	4.7/4.8	10.2/10.9
PB4~PB12		0.8/0.9	1.5/1.4	4.7/4.8	10.2/10.9
PB13~PB19		1.4/1.4	2.6/2.5	4.1/4.2	8.3/9.0
PB20~PB28		1.4/1.4	2.6/2.5	4.1/4.2	8.3/8.9
PB30~PB31		0.8/0.9	1.5/1.4	4.7/4.7	10.2/10.8

NOTE

When setting the SR of the I/O port to 1 (slower level), the timing of all functions will not be guaranteed.

11.2.8 I/O Shutdown & RESET

The power of I/O can be shut down through the GROUP_X_SHDN bit in the PADCTRL register. You can use this function to conserve power.

- When GROUP_X_SHDN is configured, the state of I/O pull resistor will maintain the state before configuration. At this time, the change of the I/O pull-up/pull-down register will not change the state of the I/O pull resistor.
- When GROUP_X_RSTB is set to 0, the state of this group of I/O pull resistors will change to no pull. At this time, the change of the I/O pull-up/pull-down register will not change the state of the I/O pull resistor.

Table 11-6 I/O Shutdown & RESET register

Address	Bit	Access	INI	Symbol	Description
0x41008000	[25]	R/W	0	LPGPIO_RSTB	1: Release LP I/O RSTB33 (except Group 5) 0: Global Assert LP I/O RSTB33
	[24]	R/W	0	LPGPIO_SHDN	1: Global Enable LP I/O (except Group 5) 0: Shutdown LP I/O
	[21]	R/W	1	GROUP_5_RSTB	1: Enable group 5 I/O 0: Reset group 5 I/O
	[20]	R/W	1	GROUP_4_RSTB	1: Enable group 4 I/O 0: Reset group 4 I/O
	[19]	R/W	1	GROUP_3_RSTB	1: Enable group 3 I/O 0: Reset group 3 I/O
	[18]	R/W	1	GROUP_2_RSTB	1: Enable group 2 I/O 0: Reset group 2 I/O
	[17]	R/W	1	GROUP_1_RSTB	1: Enable group 1 I/O 0: Reset group 1 I/O
	[16]	R/W	1	GROUP_0_RSTB	1: Enable group 0 I/O 0: Reset group 0 I/O
	[13]	R/W	0	GROUP_5_SHDN	1: Enable group 5 I/O 0: Shutdown I/O Note: AON PMC will auto release this bit in power on init seq.
	[12]	R/W	1	GROUP_4_SHDN	1: Enable group 4 I/O 0: Shutdown I/O
	[11]	R/W	1	GROUP_3_SHDN	1: Enable group 3 I/O 0: Shutdown I/O
	[10]	R/W	0	GROUP_2_SHDN	1: Enable group 2 I/O 0: Shutdown I/O

All information provided in this document is subject to legal disclaimers.
© REALTEK 2024. All rights reserved.

	[9]	R/W	1	GROUP_1_SHDN	1: Enable group 1 I/O 0: Shutdown I/O
	[8]	R/W	1	GROUP_0_SHDN	1: Enable group 0 I/O 0: Shutdown I/O

NOTE

- *GROUP_0: PA0~PA12*
- *GROUP_1: PA13~PA25*
- *GROUP_2:PA26~PA31 & PB0~PB3*
- *GROUP_3:PB4~PB12*
- *GROUP_4:PB13~PB28*
- *GROUP_5:PB30~PB31*

11.2.9 Open Drain Mode

The I/O is default to push pull mode, and can be configured to implement open drain mode. For more details, refer to the software APIs.

11.3 pinmapcfg

During the process of boot, sleep and deep-sleep, I/O pull control is needed, and the chip will load the I/O status of each GPIO from the "pinmapcfg.c" file. The correct configuration of pinmap can achieve low power consumption; otherwise, the unsuitable configuration may lead to leakage. For example, UART voltage level is high. If the UART pin is pulled down or not pulled, power leakage happens. So it is necessary to make sure that each pin has proper pull control.

In SDK, users should set GPIO function pull control and sleep/deep-sleep pull control based on users' PCB board, and SDK will set pull control based on users' setting between suspend and resume.

In the "pinmapcfg.c" file, PMAP_TypeDef pmap_func[] should be configured. In which,

- Pin Name: indicates the GPIO.
- Func PU/PD: is used to configure the I/O status when power-on.
- Slp PU/PD: is used to configure the I/O status at sleep mode.

The following configuration is only applicable to Realtek EVB, and customers need to configure it according to the external circuit.

```
const PMAP_TypeDef pmap_func[] = {
    //Pin Name      Func PU/PD              Slp PU/PD
    {_PA_0,         GPIO_PuPd_UP,           GPIO_PuPd_UP},          //
    {_PA_1,         GPIO_PuPd_UP,           GPIO_PuPd_UP},          //Flash cs pin
    {_PA_2,         GPIO_PuPd_DOWN,         GPIO_PuPd_DOWN},        //
    {_PA_3,         GPIO_PuPd_DOWN,         GPIO_PuPd_DOWN},        //
    {_PA_4,         GPIO_PuPd_DOWN,         GPIO_PuPd_DOWN},        //
    {_PA_5,         GPIO_PuPd_KEEP,         GPIO_PuPd_KEEP},        //
    {_PA_6,         GPIO_PuPd_DOWN,         GPIO_PuPd_DOWN},        //
    {_PA_7,         GPIO_PuPd_DOWN,         GPIO_PuPd_DOWN},        //
    {_PA_8,         GPIO_PuPd_KEEP,         GPIO_PuPd_DOWN},        //
    {_PA_9,         GPIO_PuPd_DOWN,         GPIO_PuPd_DOWN},        //
    {_PA_10,        GPIO_PuPd_DOWN,         GPIO_PuPd_DOWN},        //
    {_PA_11,        GPIO_PuPd_DOWN,         GPIO_PuPd_DOWN},        //
    {_PA_12,        GPIO_PuPd_DOWN,         GPIO_PuPd_DOWN},        //
    {_PA_13,        GPIO_PuPd_DOWN,         GPIO_PuPd_DOWN},        //
    {_PA_14,        GPIO_PuPd_DOWN,         GPIO_PuPd_DOWN},        //
    {_PA_15,        GPIO_PuPd_DOWN,         GPIO_PuPd_DOWN},        //
    {_PA_16,        GPIO_PuPd_DOWN,         GPIO_PuPd_DOWN},        //
    {_PA_17,        GPIO_PuPd_DOWN,         GPIO_PuPd_DOWN},        //
    {_PA_18,        GPIO_PuPd_UP,           GPIO_PuPd_UP},          //Flash cs pin
    {_PA_19,        GPIO_PuPd_DOWN,         GPIO_PuPd_DOWN},        //
    {_PA_20,        GPIO_PuPd_DOWN,         GPIO_PuPd_DOWN},        //
    {_PA_21,        GPIO_PuPd_DOWN,         GPIO_PuPd_DOWN},        //
    {_PA_22,        GPIO_PuPd_DOWN,         GPIO_PuPd_DOWN},        //
    {_PA_23,        GPIO_PuPd_DOWN,         GPIO_PuPd_DOWN},        //
    {_PA_24,        GPIO_PuPd_DOWN,         GPIO_PuPd_DOWN},        //
    {_PA_25,        GPIO_PuPd_DOWN,         GPIO_PuPd_DOWN},        //
    {_PA_26,        GPIO_PuPd_DOWN,         GPIO_PuPd_DOWN},        //
    {_PA_27,        GPIO_PuPd_KEEP,         GPIO_PuPd_DOWN},        //
```


All information provided in this document is subject to legal disclaimers.
© REALTEK 2024. All rights reserved.

```
    {_PA_28,        GPIO_PuPd_DOWN,         GPIO_PuPd_DOWN},        //
    {_PA_29,        GPIO_PuPd_DOWN,         GPIO_PuPd_DOWN},        //
    {_PA_30,        GPIO_PuPd_DOWN,         GPIO_PuPd_DOWN},        //
    {_PA_31,        GPIO_PuPd_DOWN,         GPIO_PuPd_DOWN},        //
    {_PB_0,         GPIO_PuPd_DOWN,         GPIO_PuPd_DOWN},        //
    {_PB_1,         GPIO_PuPd_DOWN,         GPIO_PuPd_DOWN},        //
    {_PB_2,         GPIO_PuPd_DOWN,         GPIO_PuPd_DOWN},        //
    {_PB_3,         GPIO_PuPd_DOWN,         GPIO_PuPd_DOWN},        //
    {_PB_4,         GPIO_PuPd_UP,           GPIO_PuPd_UP},          //log_RX  sleep  need
pull up
    {_PB_5,         GPIO_PuPd_UP,           GPIO_PuPd_UP},          //log_TX  sleep  need
pull up
    {_PB_6,         GPIO_PuPd_DOWN,         GPIO_PuPd_DOWN},        //
    {_PB_7,         GPIO_PuPd_DOWN,         GPIO_PuPd_DOWN},        //
    {_PB_8,         GPIO_PuPd_DOWN,         GPIO_PuPd_DOWN},        //
    {_PB_9,         GPIO_PuPd_DOWN,         GPIO_PuPd_DOWN},        //
    {_PB_10,        GPIO_PuPd_DOWN,         GPIO_PuPd_DOWN},        //
    {_PB_11,        GPIO_PuPd_DOWN,         GPIO_PuPd_DOWN},        //
    {_PB_12,        GPIO_PuPd_NOPULL,       GPIO_PuPd_NOPULL},      //
    {_PB_13,        GPIO_PuPd_DOWN,         GPIO_PuPd_DOWN},        //
    {_PB_14,        GPIO_PuPd_DOWN,         GPIO_PuPd_DOWN},        //
    {_PB_15,        GPIO_PuPd_DOWN,         GPIO_PuPd_DOWN},        //
    {_PB_16,        GPIO_PuPd_DOWN,         GPIO_PuPd_DOWN},        //
    {_PB_17,        GPIO_PuPd_DOWN,         GPIO_PuPd_DOWN},        //
    {_PB_18,        GPIO_PuPd_DOWN,         GPIO_PuPd_DOWN},        //
    {_PB_19,        GPIO_PuPd_DOWN,         GPIO_PuPd_DOWN},        //
    {_PB_20,        GPIO_PuPd_DOWN,         GPIO_PuPd_DOWN},        //
    {_PB_21,        GPIO_PuPd_DOWN,         GPIO_PuPd_DOWN},        //
    {_PB_22,        GPIO_PuPd_DOWN,         GPIO_PuPd_DOWN},        //
    {_PB_23,        GPIO_PuPd_DOWN,         GPIO_PuPd_DOWN},        //
    {_PB_24,        GPIO_PuPd_DOWN,         GPIO_PuPd_DOWN},        //
    {_PB_25,        GPIO_PuPd_DOWN,         GPIO_PuPd_DOWN},        //
    {_PB_26,        GPIO_PuPd_DOWN,         GPIO_PuPd_DOWN},        //
    {_PB_27,        GPIO_PuPd_DOWN,         GPIO_PuPd_DOWN},        //
    {_PB_28,        GPIO_PuPd_DOWN,         GPIO_PuPd_DOWN},        //
    {_PB_29,        GPIO_PuPd_DOWN,         GPIO_PuPd_DOWN},        //
    {_PB_30,        GPIO_PuPd_UP,           GPIO_PuPd_UP},          //
    {_PB_31,        GPIO_PuPd_UP,           GPIO_PuPd_UP},          //
    {_PNC,          GPIO_PuPd_KEEP,         GPIO_PuPd_KEEP},        //
```

There are five states of the GPIO.

- GPIO_PuPd_UP: indicates that the GPIO is through the resistor pulled up to VDDIO.
- GPIO_PuPd_DOWN: indicates that the GPIO is through the resistor pulled down to GND.
- GPIO_PuPd_NOPULL: indicates that the GPIO is in High-Z.
- GPIO_PuPd_KEEP: is used for sleep and deep-sleep mode, and indicates that the GPIO will maintain the last status before the chip enters sleep mode or deep-sleep mode.
- GPIO_PuPd_SHUTDOWN: indicates that the GPIO is shut down.

The following section illustrates the recommendation of IO status configurations according to the function of different pins.

11.3.1 Normal GPIO

When a pin is used as a normal GPIO connecting with external circuit, the GPIO PU/PD status depends on the state of the external circuit. If the GPIO is used to driver LED, and pin status is 'External pull up', the GPIO status need to be 'PULL UP' in sleep and DSLP mode.

Configure the state of the internal PU/PD according to the GPIO external circuits.

Table 11-7 Normal GPIO status

I/O type	Pin status	Active PU/PD	Slp PU/PD
Input	External Pull UP	UP/NOPULL	UP/NOPULL
Input	External Pull Down	DOWN/NOPULL	DOWN/NOPULL
Input	Floating	DOWN	DOWN
Output	Output High	UP/DOWN/NOPULL	DOWN
Output	Output Low	UP/DOWN/NOPULL	DOWN

All information provided in this document is subject to legal disclaimers.
© REALTEK 2024. All rights reserved.

11.3.2 LOGUART

The pins PB4 and PB5 are LOGUART function by default. The I/O status are listed in *Table 11-8*.

Table 11-8 LOGUART I/O status

Pin function	Pin name	Func PU/PD	Slp PU/PD
UART_LOG_RXD	PB4	UP	KEEP
UART_LOG_TXD	PB5	UP	KEEP

11.3.3 SWD

The pins PA30 and PA31 are SWD function by default.The I/O status are listed in *Table 11-9*.

Table 11-9 SWD I/O status

Pin function	Pin name	Func PU/PD	Slp PU/PD
SWD_DATA	PA31	UP	KEEP
SWD_CLK	PA30	UP	KEEP

NOTE

PA30&PA31 is the SWD function by default. If they need to be used as GPIOs or other functions, besides switching the PINMUX function ID, users also need to set 0x410089F4[0] = 0.

11.3.4 Flash Pin

When a pin is configured as SPI_FLASH function, the I/O status is listed in *Table 11-10*.

Table 11-10 Flash I/O status

Pin function	Func PU/PD	Slp PU/PD
SPI_DATA_X	UP	KEEP
SPI_CS	UP	KEEP
SPI_CLK	DOWN	KEEP

All information provided in this document is subject to legal disclaimers.
© REALTEK 2024. All rights reserved.

12 General Purpose Input/Output (GPIO)

12.1 Introduction

GPIO is a programmable general purpose programming I/O peripheral.

Figure 12-1 illustrates the following functional groupings of the main interfaces to the GPIO block.

- APB interface to or from APB bridge
- External data interface to or from I/O pads
- Interrupt interface to or from interrupt controller

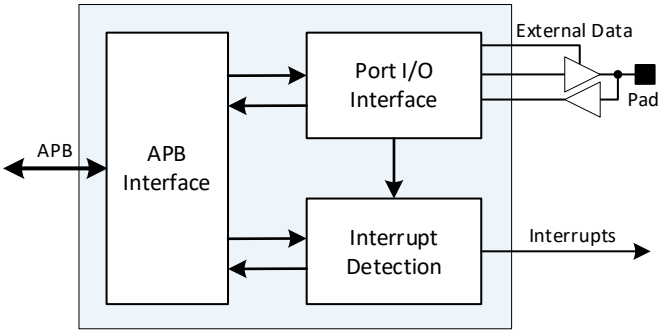


Figure 12-1 Block diagram

RTL8721Dx supports two independent GPIO IP: Port A (0~31), Port B (0~31), which controls the output data and direction of external I/O pads. Also, it can read back the data on external pads using memory-mapped registers.

Port A/B can be programmed to accept external signals as interrupt sources on any bits of the signal. The type of interrupt is programmable with one of the following settings:

- Each general purpose I/O port has four 32-bit data registers (gpio_swportx_dr, gpio_swportX_ddr, gpio_swportX_ctrl, gpio_ext_portx);
- Each port has eleven 32-bit interrupt related registers (gpio_int_en, gpio_int_mask, gpio_int_type, gpio_int_polarity, gpio_int_status, gpio_int_status_raw, gpio_debounce, gpio_int_eoi, gpio_int_ls_sync, gpio_int_bothedge, gpio_db_div_cnt).

12.2 Features

GPIO supports the following features:

- Up to 96 independently configurable signals and 32 signals for each port
- Separate data register and data direction register for each signal
- Configurable software control for each signal, or for each bit of each signal
- Independently controllable signal bits
- Interrupt mode
- Option to generate single or multiple interrupts
- Configurable debounce logic with an external slow clock to debounce interrupts

12.3 Functional Description

This chapter describes the basic functional operation of the GPIO.

12.3.1 Data Read and Write Control

GPIO controls the output data and direction of external I/O pads. It also can read back the data on external pads using memory-mapped registers. Software can control each signal over the APB bus interface. The data and control flow for a signal are shown in *Figure 12-2*.

All information provided in this document is subject to legal disclaimers.
© REALTEK 2024. All rights reserved.

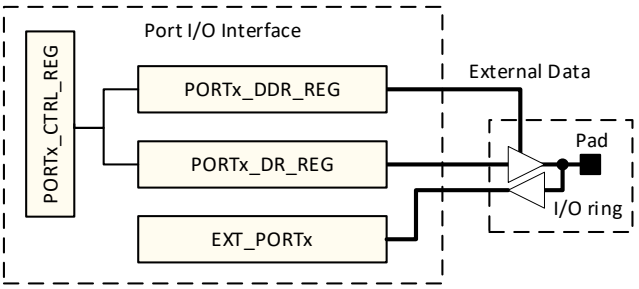


Figure 12-2 Control RTL block diagram

NOTE

- *Each bit in each signal is individually controllable. Therefore, each register can be regarded as N individual registers, where N is the signal width with the range from 0 to 31.*
- *PORTx in this chapter means Port A (0~31), Port B (0~31), where x is A or B.*

12.3.1.1 GPIO Direction

The direction of the external I/O pad is controlled by a write to the PORTx data direction register (**gpio_swportx_ddr**). This output signal controls the direction of an external I/O pad.

12.3.1.2 GPIO Write

When configured as output, the data written to the PORTx data register (**gpio_swportx_dr**) drives the output buffer of the I/O pad.

12.3.1.3 GPIO Read

The data on the external GPIO signal is read by an APB read of the memory-mapped register, **gpio_ext_portx**. An APB read to the **gpio_ext_portx** register provides either the data on the **gpio_ext_portx** control lines or the contents of the **gpio_swportx_dr**, depending on whether **gpio_ext_portx** is configured as an input or output, the following occurs:

- Input: read the values on the signal
- Output: read the data register for PORTx

The **gpio_ext_portx** register is read-only, meaning that it cannot be written from the APB software interface.

Figure 12-3 shows a timing diagram of a read to the **gpio_ext_portx** memory map registers when the direction is set to input and the metastability registers are included, which brings a data delay of the port line.

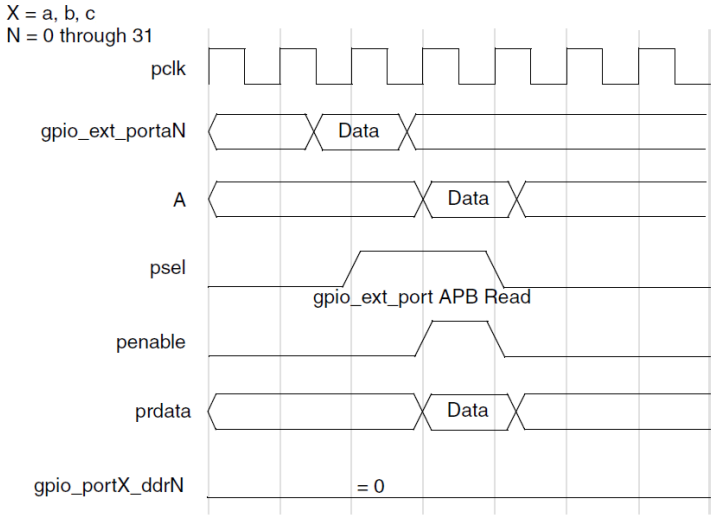


Figure 12-3 Read back of external I/O data timing

NOTE

All information provided in this document is subject to legal disclaimers.
© REALTEK 2024. All rights reserved.

The maximum data rate that can be read back on consecutive reads from gpio_ext_portx must be less than pclk/2. Two pclk cycles per read access is required due to the non-pipelined nature of the APB. The assumption is that the APB bridge does not lose ownership of the AHB during consecutive accesses when PCLK=HCLK.

12.3.2 Interrupts

External signals can be configured as interrupt sources on any bits of the signal. The type of interrupt is programmable with one of the following settings:

- Level-sensitive interrupt:
  - Active-high level
  - Active-low level
- Edge-trigger interrupt:
  - Rising edge
  - Falling edge
  - Both edge

The interrupts can be masked by programming the **gpio_intmask** register. The interrupt status can be read before masking (called raw status from **gpio_int_status_raw**) and after masking (**gpio_int_status**).

To configure PORTx for interrupts, the data direction must be set to Input. If the data direction register is reprogrammed to Output, then any pending interrupts will not be lost. However, no new interrupts are generated.

Figure 12-4 illustrates how the interrupts are generated and how the data flows. The signal names in the diagram correspond to either I/O signals or memory-mapped registers.

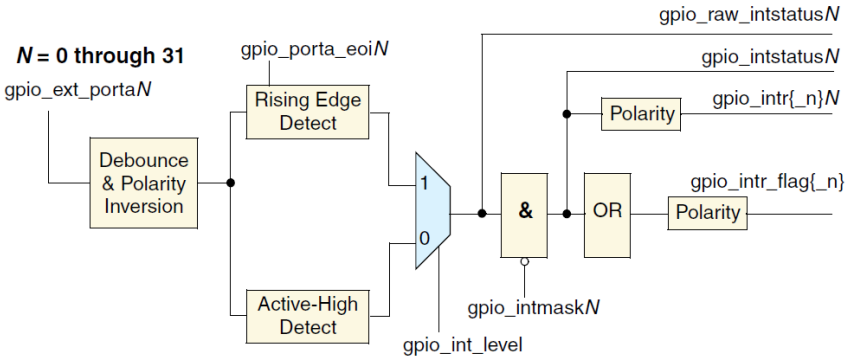


Figure 12-4 Interrupt RTL block diagram

The interrupts are also combined into a single interrupt output signal, which has the same polarity as the individual interrupts. Either individual interrupts (gpio_intr or gpio_intr_n) or a single combined interrupt (gpio_intr_flag or gpio_intr_flag_n) can be generated. In order to mask the combined interrupt, all individual interrupts have to be masked. The single combined interrupt does not have its own mask bit.

The simplest connection scheme is where the combined interrupt gpio_intr_flag is generated by ORing together the bits of the gpio_intr bus.

- When only the combined interrupt request is used, then the gpio_status register must be read in the interrupt service routine (ISR) to find the source of the interrupt. Then it is up to the processor to prioritize these pending interrupt requests.
- When the individual interrupts lines are connected directly to the interrupt controller, then the **gpio_status** register does not have to be read by the ISR.

12.3.2.1 Debounce Operation (optional)

If the user has configured the interrupt feature, GPIO can be configured to either include or exclude a debounce capability using the GPIO_DEBOUNCE parameter. The external signal can be debounced to remove any spurious glitches that are less than one period of the external debouncing clock.

Figure 12-5 shows an RTL diagram of the debounce circuitry. The timing diagram shows an active-high input signal on gpio_ext_portxN. The polarity of the input signal detection is controlled by the memory-mapped signal, gpio_int_polarity. For a falling-edge or active-low-sensitive input, the input is then inverted and the same debounce logic is used as for rising-edge or active-high level-sensitive interrupts.

All information provided in this document is subject to legal disclaimers.
© REALTEK 2024. All rights reserved.

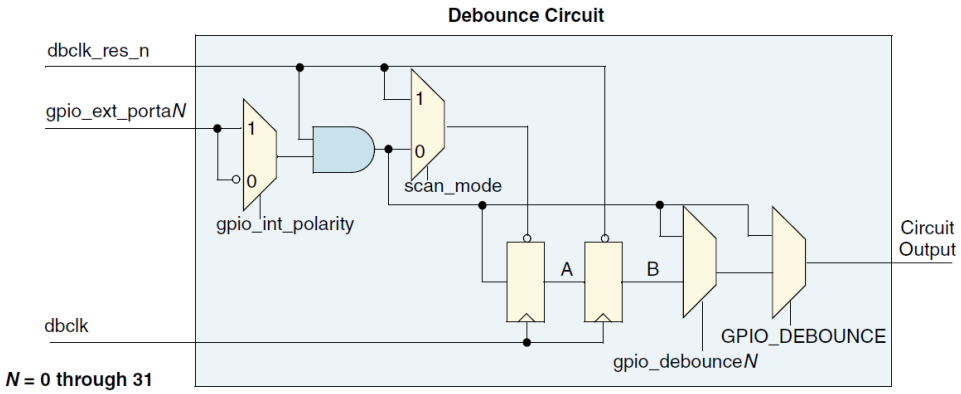


Figure 12-5 Debounce RTL diagram

- dbclk_res_n signal is asynchronously asserted and synchronously de-asserted to the debounce clock, dbclk.
- presetn, the system reset signal, is asynchronously asserted and synchronously de-asserted to pclk; synchronization must be external to the component.
- The pclk and dbclk signals are assumed to be asynchronous to each other.

When input interrupt signals are debounced using a debounce clock, the signals must be active for a minimum of two cycles of the debounce clock to guarantee that they are registered. Any input pulse widths less than a debounce clock period are bounced. A pulse width between one and two debounce clock widths may or may not propagate, depending on its phase relationship to the debounce clock. If the input pulse spans two rising edges of the debounce clock, it is registered. If it spans only one rising edge, it is not registered.

The timing diagram in *Figure 12-6* shows both cases: the input signal being bounced, and later, a propagated input signal. If GPIO supports debounce, then debouncing input signals on Port A can be enabled or disabled under software control.

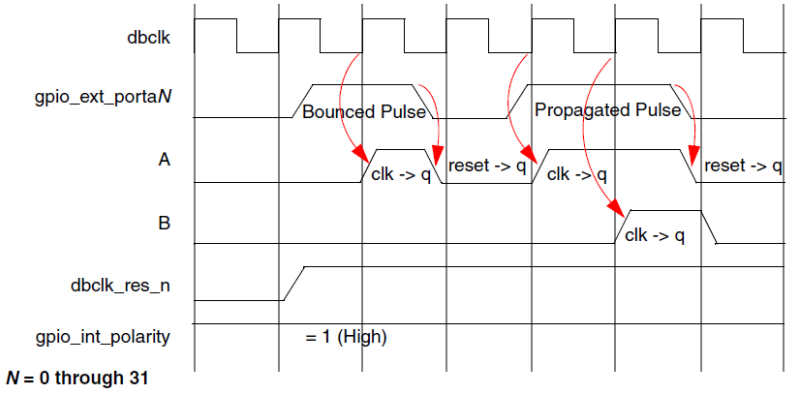


Figure 12-6 Debounce timing with asynchronous reset flip-flops

NOTE

- *The use of the debounce circuitry increases interrupt latency by two clock cycles of the debounce clock.*
- *The debounce circuitry works with only asynchronous reset flip-flops.*

12.3.2.2 Synchronization of Interrupt Signals to System Clock

Interrupt signals can be internally synchronized to a system clock, pclk_intr. Synchronization to pclk_intr must occur for edge-detect signals. Edge-detected interrupts to the processor are always synchronous to the system bus clock. For level-sensitive interrupts, synchronization is optional and under software control.

The pclk_intr signal is needed for systems that may have the GPIO pclk bus clock gated off, but the system still wants to detect interrupts. It is assumed that this clock is synchronous to pclk. If interrupt detection is required only when pclk is running, then pclk_intr and pclk can be connected to the same clock source. If the system employs a gated pclk to the GPIO, pclk_intr needs to be running for interrupt detection to occur.

The gpio_intrclk_en output signal is asserted when either edge-sensitive interrupts or level-sensitive interrupts requiring synchronization are enabled in the GPIO block. Both cases require a clock for detection. Therefore, this signal can cause the external clock generator block to generate pclk_intr.

All information provided in this document is subject to legal disclaimers.
© REALTEK 2024. All rights reserved.

12.3.2.3 Interrupt Edge Detection

For edge-detected interrupts, the ISR can clear the interrupt by writing a 1 to the **gpio_porta_eoi** register for the corresponding bit to disable the interrupt. This write also clears the interrupt status and raw status registers. It is recommended that the interrupt source be cleared prior to writing to the **gpio_porta_eoi** register. Writing to the **gpio_porta_eoi** register has no effect on level-sensitive interrupts. For combined interrupt request connection, there are no restrictions on the number of edge-detected interrupts that can be cleared simultaneously by writing multiple 1's to the **gpio_porta_eoi** register.

12.3.2.3.1 Single Edge

Figure 12-7 shows an RTL diagram of the synchronization and edge detection of interrupt sources on gpio_ext_portaN signals, when GPIO_INT_BOTH_EDGE=0.

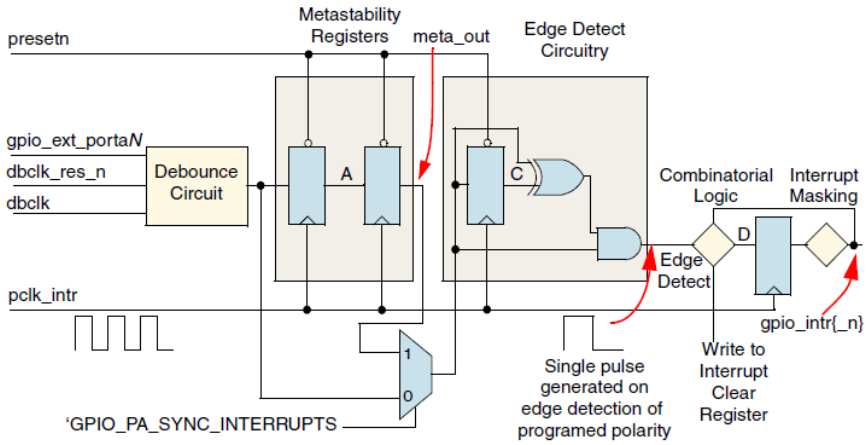


Figure 12-7 Synchronization and edge detect interrupt generation (GPIO_INT_BOTH_EDGE=0)

NOTE

Metastability part is default included.

Figure 12-8 shows a timing diagram in which an interrupt is generated on the rising edge of input on Port A; this is where the debounce logic is disabled and metastability registers are included. It also shows how an interrupt is cleared by a write to the interrupt clear register.

A case may arise where the Interrupt Service Routine (ISR) writes to the interrupt clear register to clear an existing interrupt during the same clock cycle in which a new interrupt is detected. In such a case, writing to the interrupt clear register clears only the first interrupt. The second interrupt is not lost, since setting an interrupt has a higher priority than clearing it.

Figure 12-9 shows a timing diagram similar to *Figure 12-8* except that metastability registers are removed. It also shows how an interrupt is cleared by a write to the interrupt clear register.

NOTE

Since the metastability registers are removed from the path, A and B waveforms, shown in Figure 12-8, no longer exist, and all subsequent signal delays are reduced by two clock cycles.

All information provided in this document is subject to legal disclaimers.
© REALTEK 2024. All rights reserved.

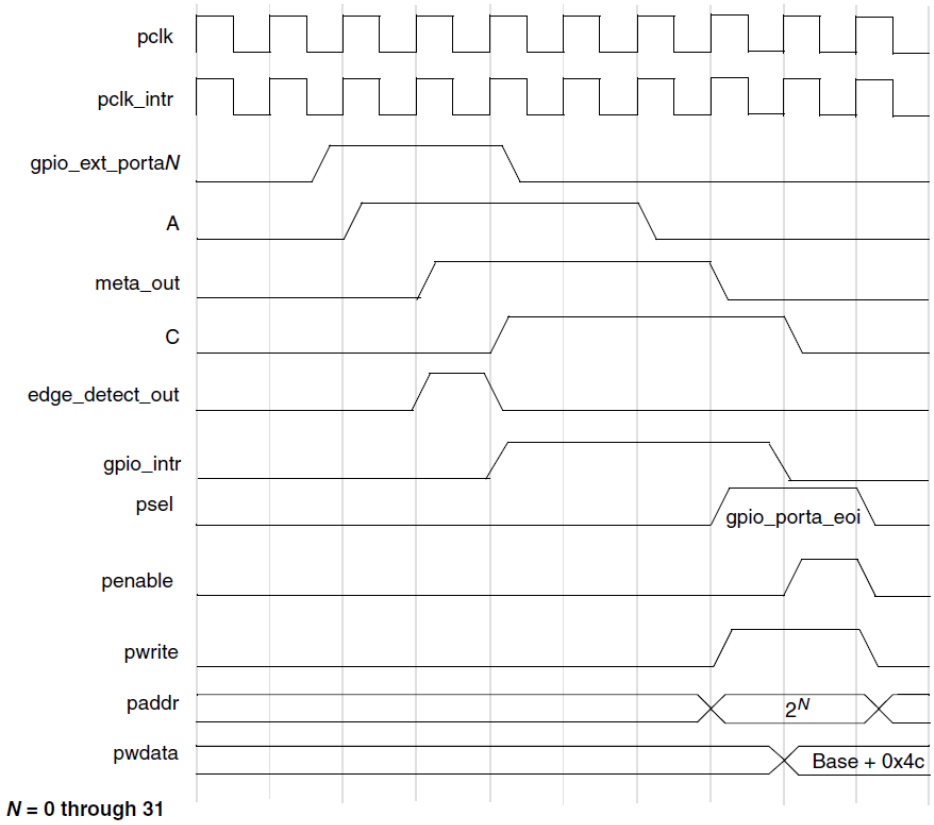


Figure 12-8 Interrupt edge detection and interrupt clear timing (GPIO_SYNC_PA_INTERRPUTS = 1, metastability included)

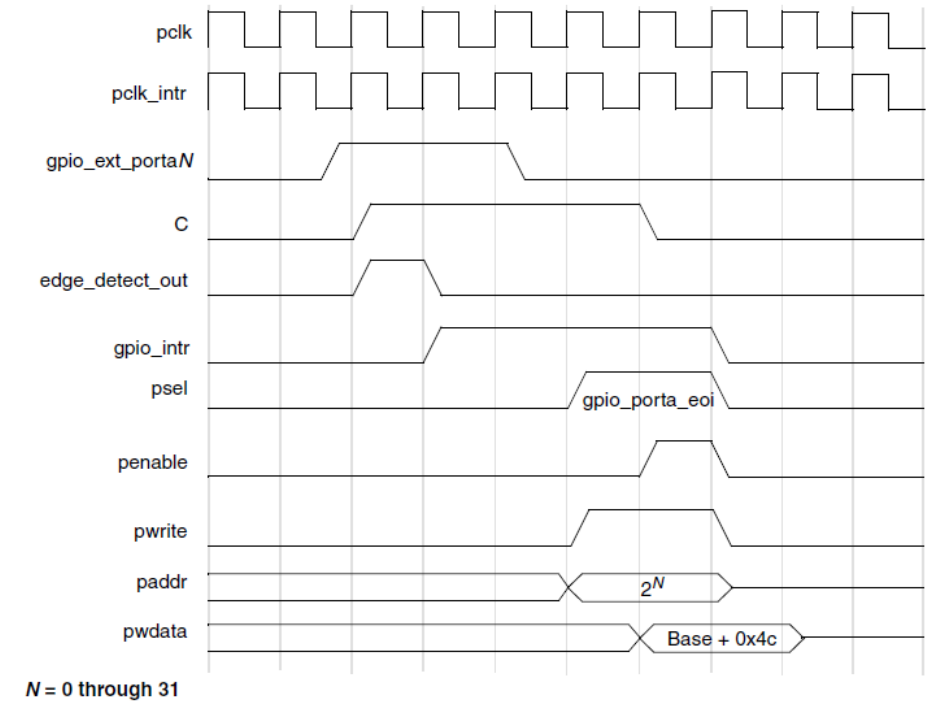


Figure 12-9 Interrupt edge detection and interrupt clear timing (GPIO_SYNC_PA_INTERRPUTS = 0, metastability removed)

Figure 12-10 shows such a case where the debounce logic is unused. In this timing diagram, meta_out and edge_detect_out are the outputs of the second metastability register and the edge detect logic, respectively. The second edge detection occurs on the same cycle as the write to the interrupt clear register. In this example, the write to the interrupt clear register does not clear the second interrupt, and the gpio_intr{_n} signal is not de-asserted.

All information provided in this document is subject to legal disclaimers.
© REALTEK 2024. All rights reserved.

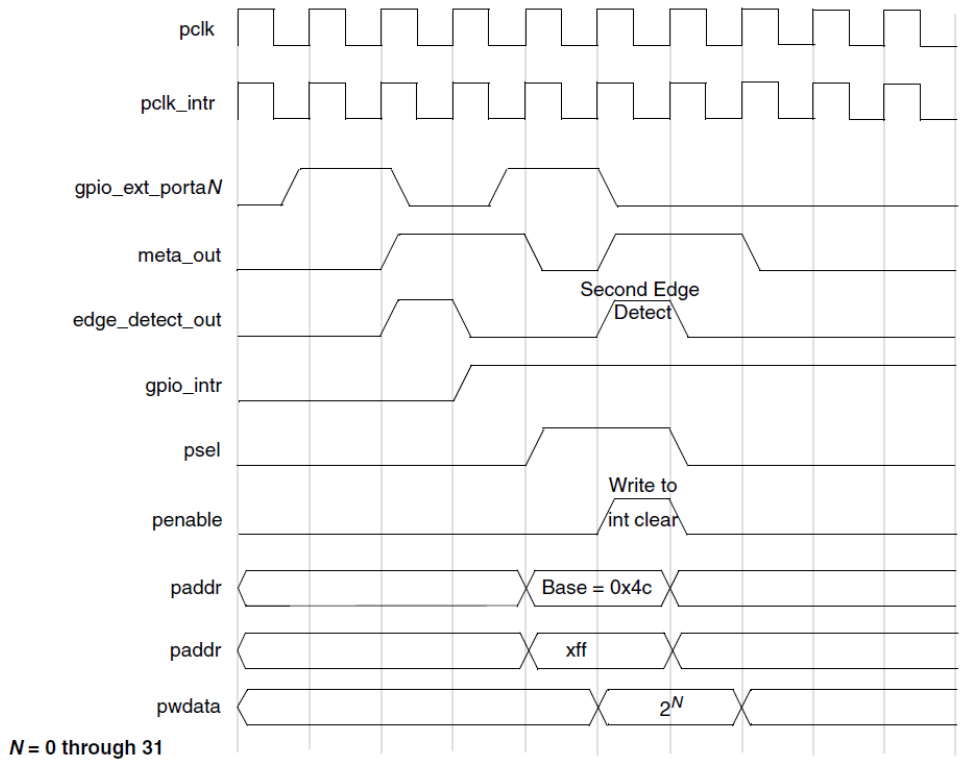


Figure 12-10 Write to interrupt clear register, coincident with detection of new interrupt

12.3.2.3.2 Both Edges

Interrupt detection logic for both the rising edge and the falling edge is available only when GPIO_INT_BOTH_EDGE = 1 and this logic detects the interrupt on both the rising edge and the falling edge when the **gpio_int_bothedge** register is programmed to 1.

Figure 12-11 shows the synchronization and edge detect interrupt generation of the interrupt sources on gpio_ext_portaN signals when GPIO_INT_BOTH_EDGE = 1.

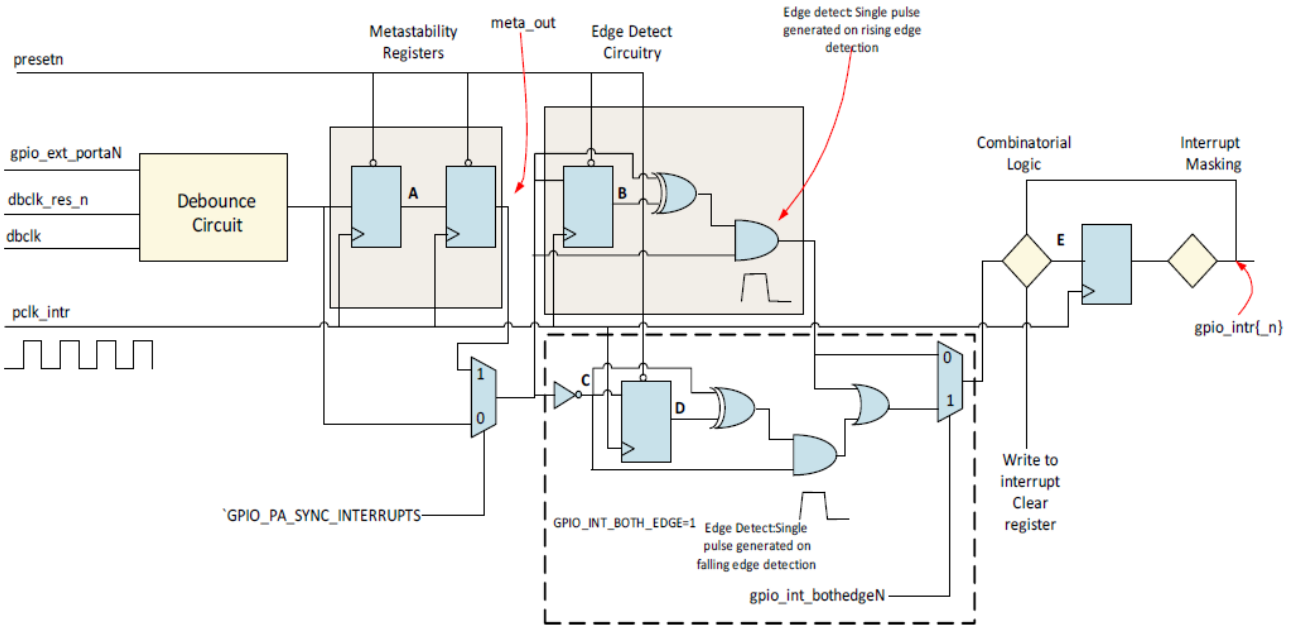


Figure 12-11 Synchronization and edge detect interrupt generation (GPIO_INT_BOTH_EDGE=0)

Figure 12-12 shows a timing diagram where an interrupt is generated on both the rising edge and the falling edge of input on Port A, that is, with GPIO_INT_BOTH_EDGE = 1 and **gpio_int_bothedge** programmed to detect both edges. In this scenario, debounce logic is disabled and

All information provided in this document is subject to legal disclaimers.
© REALTEK 2024. All rights reserved.

metastability registers are included. This figure also shows how an interrupt is cleared by a write to the interrupt clear register.

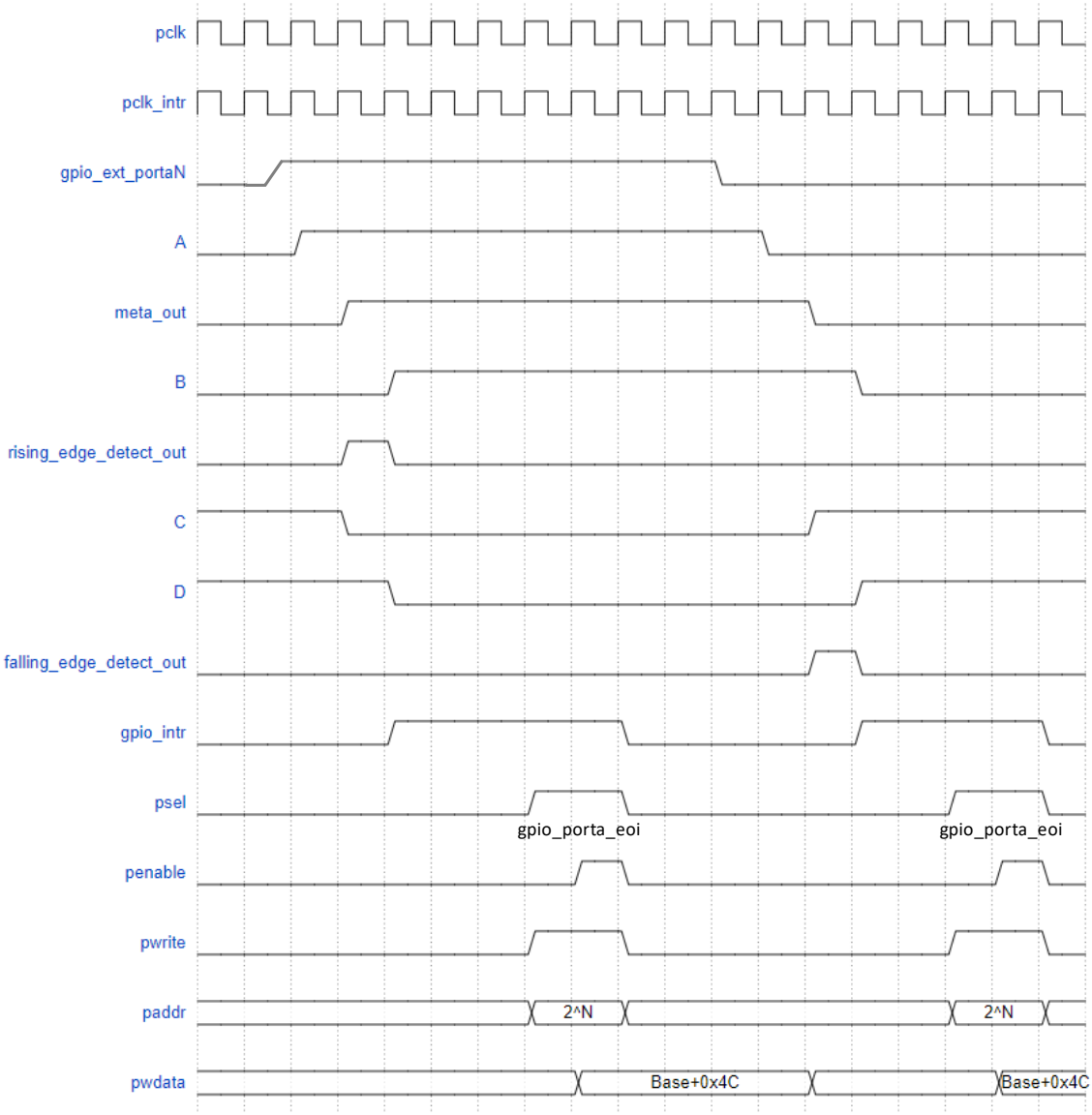


Figure 12-12 Interrupt edge detection and interrupt clear timing (GPIO_SYNC_PA_INTERRPUTS = 1 and GPIO_INT_BOTH_EDGE=1, metastability included)

Figure 12-13 shows a timing diagram similar to *Figure 12-12*, except that in this scenario, metastability registers are removed.

All information provided in this document is subject to legal disclaimers.
© REALTEK 2024. All rights reserved.

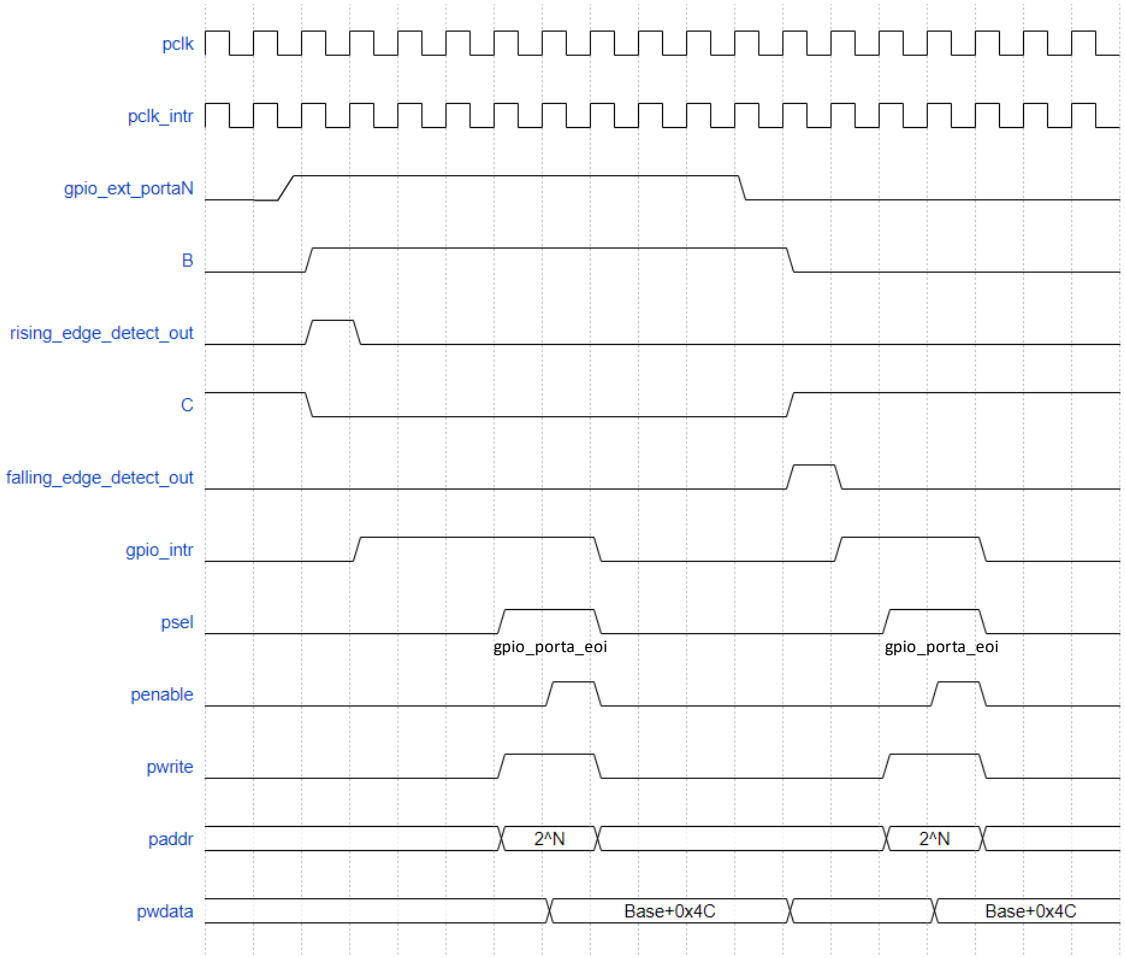


Figure 12-13 Interrupt edge detection and interrupt clear timing (GPIO_SYNC_PA_INTERRPUTS = 0 and GPIO_INT_BOTH_EDGE=1, metastability removed)

12.3.2.4 Level-Sensitive Interrupts

If level-sensitive interrupts cause the processor to interrupt, then the ISR can poll the gpio_rawint status register until the interrupt source disappears, or it can write to the gpio_intmask register to mask the interrupt before exiting the ISR. If the ISR exits without masking or disabling the interrupt prior to exiting, then the level-sensitive interrupt repeatedly requests an interrupt until the interrupt is cleared at the source.

Figure 12-14 shows the generation of level-sensitive interrupts. As for edge-detect interrupts, the user can configure GPIO with or without external debounce logic. The debounce circuit below is an example of debounce or remove spurious glitch less than 2 cycles of dbclk.

All information provided in this document is subject to legal disclaimers.
© REALTEK 2024. All rights reserved.

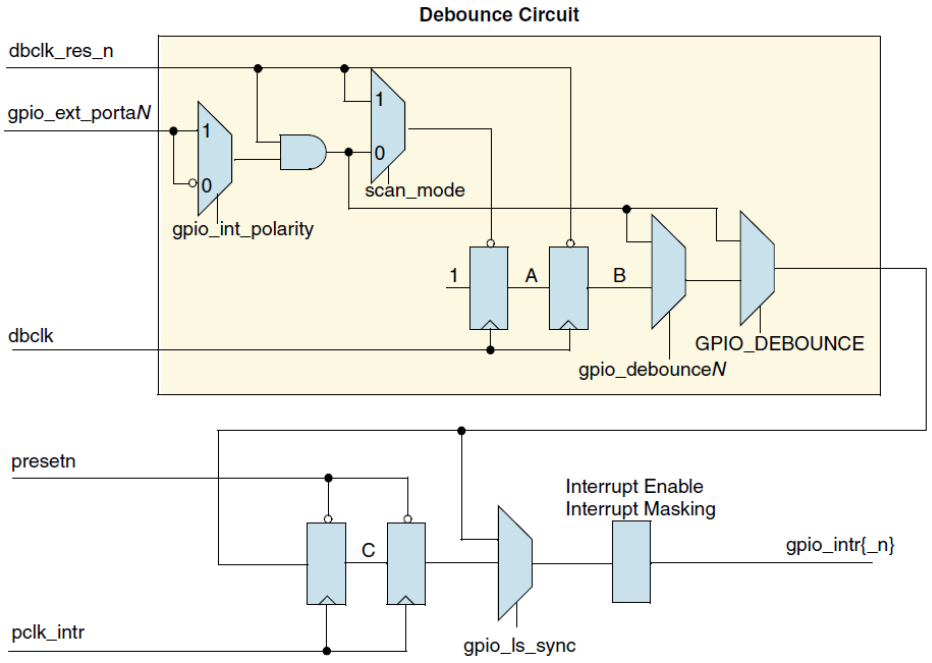


Figure 12-14 Level-sensitive interrupt RTL diagram

With level-sensitive interrupts, there is a choice of whether they are synchronized to the interrupt clock pclk_intr or are entirely combinational (aside from the debounce circuit). The selection is done by programming the **gpio_ls_sync** (GPIO Level Sensitive Synchronous) register.

This is a memory-mapped bit that inserts two metastability registers clocked off of pclk_intr to synchronize the level-sensitive interrupts to pclk_intr. When **gpio_ls_sync** is not asserted, then there is no guarantee that the interrupt lines are synchronous to pclk_intr. A processor status register may need to be set to indicate asynchronous interrupts. When **gpio_ls_sync** is asserted, then the pclk_intr clock must be present to pass the interrupt to the interrupt controller block. The gpio_intrclk_en output signal is asserted when level-sensitive interrupts to be synchronized to pclk_intr are selected. The gpio_intrclk_en signal can be used in the clock generation block to turn on pclk_intr.

NOTE

- *In active mode, pclk_intr synchronize to pclk is recommended to prevent the production of burrs in the circuit*
- *In sleep mode, if you need to wake up quickly, you can be out of this synchronization to pclk_intr.*

Figure 12-15 shows the generation of an active-low level-sensitive interrupt where the debounce circuitry is disabled.

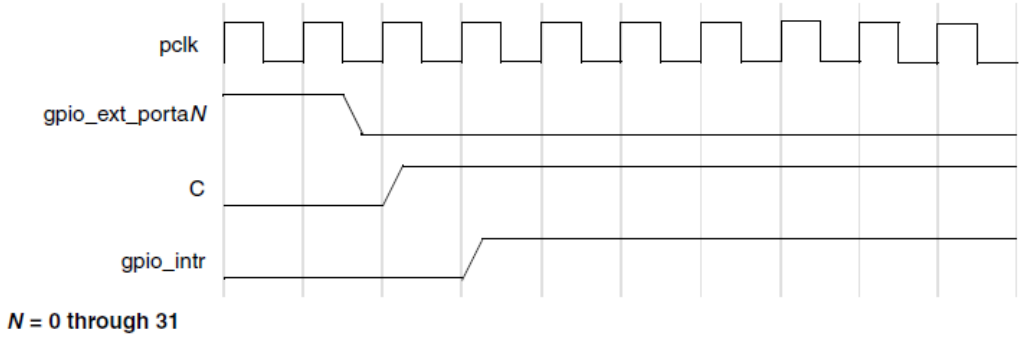


Figure 12-15 Active-low level-sensitive interrupt generation timing

12.4 Registers

The following table lists the memory map of the GPIO register. The base address are:

- GPIO_A: 0x4101_0000
- GPIO_B: 0x4101_0400

Name	Address offset	Access	Description
REG_GPIO_DR	000h	R/W	PortA/B Data Register

All information provided in this document is subject to legal disclaimers.
© REALTEK 2024. All rights reserved.

REG_GPIO_DDR	004h	R/W	PortA/B Data Direction Register
REG_GPIO_INT_EN	030h	R/W	PortA/B Interrupt Enable Register
REG_GPIO_INT_MASK	034h	R/W	PortA/B Interrupt Mask Register
REG_GPIO_INT_TYPE	038h	R/W	PortA/B Interrupt Trigger Type Register
REG_GPIO_INT_POLARITY	03Ch	R/W	PortA/B Interrupt Polarity Register
REG_GPIO_INT_STATUS	040h	R	PortA/B Interrupt Status Register
REG_GPIO_INT_STATUS_RAW	044h	R	PortA/B Raw Interrupt Status Register
REG_GPIO_DEBOUNCE	048h	R/W	PortA/B Interrupt Debounce Register
REG_GPIO_INT_EOI	04Ch	R/W	PortA/B Edge Type Interrupt Clear Register
REG_GPIO_EXT_PORT	050h	R	External PortA/B Data Register
REG_GPIO_ITN_LS_SYNC	060h	R/W	PortA/B Level Interrupt Synchronization
REG_GPIO_ID_CODE	064h	R	PortA/B GPIO ID Code Register
REG_GPIO_INT_BOTHEDGE	068h	R/W	PortA/B Both Edge Interrupt Register
REG_GPIO_VER_ID_CODE	06Ch	R	PortA/B GPIO Component Version Register

12.4.1 REG_GPIO_DR

- **Name:** PortA/B Data Register
- **Size:** 32 bits
- **Address offset:** 000h
- **Read/write access:** R/W



Bit	Symbol	Access	Reset	Description
31:0	PORT_DATA	R/W	0	Values written to this register are output on the I/O signals for Portx if the corresponding data direction bits for Portx are set to output mode and the corresponding control bit for Portx is set to software mode. The value read back equals to the last value written to this register.

12.4.2 REG_GPIO_DDR

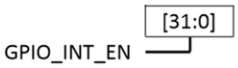
- **Name:** PortA/B Data Direction Register
- **Size:** 32 bits
- **Address offset:** 004h
- **Read/write access:** R/W



Bit	Symbol	Access	Reset	Description
31:0	PORT_DATA_DIRECTION	R/W	0	Values written to this register independently control the direction of the corresponding data bit in Portx. The default direction can be configured as input or output after system reset through the GPIO_DFLT_DIR_x parameter. 0: Input (default) 1: Output

12.4.3 REG_GPIO_INT_EN

- **Name:** PortA/B Interrupt Enable Register
- **Size:** 32 bits
- **Address offset:** 030h
- **Read/write access:** R/W



All information provided in this document is subject to legal disclaimers.
© REALTEK 2024. All rights reserved.

Bit	Symbol	Access	Reset	Description
31:0	GPIO_INT_EN	R/W	0	Allows each bit of Portx to be configured for interrupts. By default, the generation of interrupts is disabled. Whenever a 1 is written to a bit of this register, it configures the corresponding bit on Portx to become an interrupt; otherwise, Portx operates as a normal GPIO signal. Interrupts are disabled on the corresponding bits of Portx if the corresponding data direction register is set to output or if Portx mode is set to hardware. 0: Configure Portx bit as normal GPIO signal (default) 1: Configure Portx bit as interrupt

12.4.4 REG_GPIO_INT_MASK

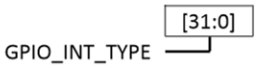
- **Name:** PortA/B Interrupt Mask Register
- **Size:** 32 bits
- **Address offset:** 034h
- **Read/write access:** R/W



Bit	Symbol	Access	Reset	Description
31:0	GPIO_INT_MASK	R/W	0	Controls whether an interrupt on Portx can create an interrupt for the interrupt controller by not masking it. By default, all interrupts bits are unmasked. Whenever a 1 is written to a bit in this register, it masks the interrupt generation capability for this signal; otherwise interrupts are allowed through. The unmasked status can be read as well as the resultant status after masking. 0: Interrupt bits are unmasked (default) 1: Mask interrupt

12.4.5 REG_GPIO_INT_TYPE

- **Name:** PortA/B Interrupt Trigger Type Register
- **Size:** 32 bits
- **Address offset:** 038h
- **Read/write access:** R/W



Bit	Symbol	Access	Reset	Description
31:0	GPIO_INT_TYPE	R/W	0	Controls the type of interrupt that can occur on Portx. Whenever a 0 is written to a bit of this register, it configures the interrupt type to be level-sensitive; otherwise, it is edge-sensitive. 0: Level-sensitive (default) 1: Edge-sensitive

12.4.6 REG_GPIO_INT_POLARITY

- **Name:** PortA/B Interrupt Polarity Register
- **Size:** 32 bits
- **Address offset:** 03Ch
- **Read/write access:** R/W



All information provided in this document is subject to legal disclaimers.
© REALTEK 2024. All rights reserved.

Bit	Symbol	Access	Reset	Description
31:0	GPIO_INT_POLARITY	R/W	0	Controls the polarity of edge or level sensitivity that can occur on input of Portx. Whenever a 0 is written to a bit of this register, it configures the interrupt type to falling-edge or active-low sensitive; otherwise, it is rising-edge or active-high sensitive. 0: Active-low (default) 1: Active-high

12.4.7 REG_GPIO_INT_STATUS

- **Name:** PortA/B Interrupt Status Register
- **Size:** 32 bits
- **Address offset:** 040h
- **Read/write access:** R



Bit	Symbol	Access	Reset	Description
31:0	GPIO_INT_STATUS	R	0	Interrupt status of Portx.

12.4.8 REG_GPIO_INT_STATUS_RAW

- **Name:** PortA/B Raw Interrupt Status Register
- **Size:** 32 bits
- **Address offset:** 044h
- **Read/write access:** R



Bit	Symbol	Access	Reset	Description
31:0	GPIO_RAW_INT_STATUS	R	0	Raw interrupt status of Portx (premasking bits).

12.4.9 REG_GPIO_DEBOUNCE

- **Name:** PortA/B Interrupt Debounce Register
- **Size:** 32 bits
- **Address offset:** 048h
- **Read/write access:** R/W

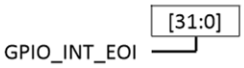


Bit	Symbol	Access	Reset	Description
31:0	GPIO_DEBOUNCE_EN	R/W	0	Controls whether an external signal that is the source of an interrupt needs to be debounced to remove any spurious glitches. Writing a 1 to a bit in this register enables the debouncing circuitry. A signal must be valid for two periods of an external clock before it is internally processed. 0: No debounce (default) 1: Enable debounce This register is available only if Portx is configured to generate interrupts (GPIO_PORTA_INTR = Include(1)) and when the debounce logic is included (GPIO_DEBOUNCE = Include(1)).

All information provided in this document is subject to legal disclaimers.
© REALTEK 2024. All rights reserved.

12.4.10 REG_GPIO_INT_EOI

- **Name:** PortA/B Edge Type Interrupt Clear Register
- **Size:** 32 bits
- **Address offset:** 04Ch
- **Read/write access:** R/W



Bit	Symbol	Access	Reset	Description
31:0	GPIO_INT_EOI	WA0	0	Controls the clearing of edge type interrupts from Portx. When a 1 is written into a corresponding bit of this register, the interrupt is cleared. All interrupts are cleared when Portx is not configured for interrupts. 0: No interrupt clear (default) 1: Clear interrupt

12.4.11 REG_GPIO_EXT_PORT

- **Name:** External PortA/B Data Register
- **Size:** 32 bits
- **Address offset:** 050h
- **Read/write access:** R



Bit	Symbol	Access	Reset	Description
31:0	GPIO_EXT_PORTA	R	0	When Portx is configured as Input, then reading this location reads the values on the signal. When the data direction of Portx is set as output, reading this location reads the data register for Portx.

12.4.12 REG_GPIO_ITN_LS_SYNC

- **Name:** PortA/B Level Interrupt Synchronization
- **Size:** 32 bits
- **Address offset:** 060h
- **Read/write access:** R/W



Bit	Symbol	Access	Reset	Description
31:1	RSVD	-	0	Reserved
0	GPIO_LEVEL_SYNC	R/W	0	Writing a 1 to this register results in all level-sensitive interrupts being synchronized to pclk_intr. 0: No synchronization to pclk_intr (default) 1: Synchronize to pclk_intr

12.4.13 REG_GPIO_ID_CODE

- **Name:** PortA/B GPIO ID Code Register
- **Size:** 32 bits
- **Address offset:** 064h
- **Read/write access:** R

All information provided in this document is subject to legal disclaimers.
© REALTEK 2024. All rights reserved.



Bit	Symbol	Access	Reset	Description
31:0	GPIO_ID_CODE	R	32'h1	This register is available only if Portx is configured to generate interrupts (GPIO_PORTx_INTR = Include(1)) and interrupt detection is configured to generate on both rising and falling edges of external input signal (GPIO_INT_BOTH_EDGE=Include(1)).

12.4.14 REG_GPIO_INT_BOTHEDGE

- **Name:** PortA/B Both Edge Interrupt Register
- **Size:** 32 bits
- **Address offset:** 068h
- **Read/write access:** R/W



Bit	Symbol	Access	Reset	Description
31:0	GPIO_INT_BOTHEDGE	R/W	0	Controls the edge types of interrupt that can occur on Portx. Whenever a particular bit is programmed to 1, it enables the generation of interrupt on both the rising edge and falling edge of an external input signal corresponding to that bit on Portx. The values programmed in the registers gpio_intype_level and gpio_int_polarity for this particular bit are not considered when the corresponding bit of this register is set to 1. Whenever a particular bit is programmed to 0, the interrupt type depends on the value of the corresponding bits in the gpio_inttype_level and gpio_int_polarity registers. 0: Disabled, single edge sensitive 1: Enabled, both edge sensitive

12.4.15 REG_GPIO_VER_ID_CODE

- **Name:** PortA/B GPIO Component Version Register
- **Size:** 32 bits
- **Address offset:** 06Ch
- **Read/write access:** R



Bit	Symbol	Access	Reset	Description
31:0	GPIO_VER_ID_CODE	R	32'h3231302A	ASCII value for each number in the version, followed by *. For example, 32_30_31_2A represents the version 2.01*.

12.5 Programming Considerations

- Reading from an unused location or unused bit sin a particular register always returns zeros. There is no error mechanism in the APB.
- Programming the GPIO registers for interrupt capability, edge-sensitive or level-sensitive interrupts, and interrupt polarity should be completed prior to enabling the interrupts on Port A/B to prevent spurious glitches on the interrupt lines to the interrupt controller.
  - If using external debounce function to remove spurious glitches, not expected interrupt may bring into the interrupt controller when enabling the interrupts.
  - To avoid unexpected interrupt with debounce circuit while re-configure the interrupt trigger mode, you had better wait the debounce time before enabling interrupt again.
- Writing to the interrupt clear register clears an edge-detected interrupt and has no effect on a level-sensitive interrupt.
- When reading back registers that are no longer present due to configuration parameters settings, 0 is read back. For example, if APB_DATA_WITDH = 32 bits and GPIO_PWIDTH_A = 8, then the top 24 bits read back 0.

All information provided in this document is subject to legal disclaimers.
© REALTEK 2024. All rights reserved.

13 Direct Memory Access Controller (DMAC)

13.1 Introduction

This chapter provides a basic overview of DMAC, which is an OCP/AXI-Central DMA Controller core that transfers data from a source peripheral to a destination peripheral over OCP/AXI bus.

13.2 General Description

Figure 13-1 shows the following functional groupings of the main interfaces to the DMAC block.

- DMA hardware request interface
- FIFO per channel for source and destination
- Arbiter
- OCP HP master interface or AXI master interface
- OCP PP slave interface or APB slave interface

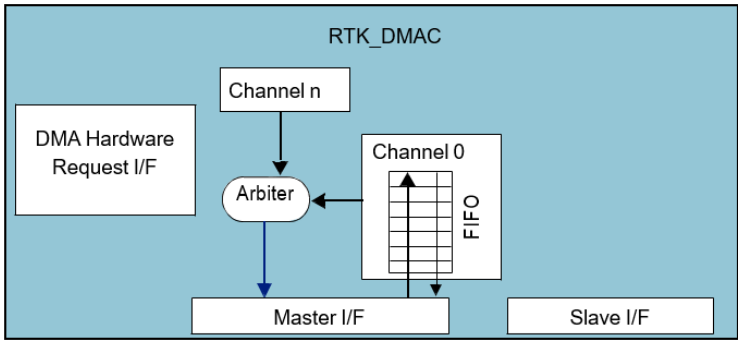


Figure 13-1 Block diagram of DMAC

One channel of the DMAC is required for each source/destination pair. In the most basic configurations, the DMAC has two master interfaces and one channel. The master interface reads the data from a source peripheral (A) and writes it to a destination peripheral (B). Two OCP/AXI transfers are required for each DMA data transfer; this is also known as a dual-access transfer.

13.2.1 Basic Definitions

The following terms are concise definitions of the DMA concepts used throughout this chapter.

- Handshaking interface – A set of signals that conform to a protocol and handshake between the DMAC and source or destination peripheral in order to control transferring a single or burst transaction between them. This interface is used to request, acknowledge, and control a DMAC transaction. A channel can receive a request through one of two types of handshaking interfaces: hardware or peripheral interrupt.
  - Hardware handshaking interface – Uses hardware signals to control transferring a single or burst transaction between the DMAC and the source or destination peripheral.
  - Peripheral interrupt handshaking interface – Simple use of the hardware handshaking interface. In this mode, the interrupt line from the peripheral is tied to the dma_req input of the hardware handshaking interface; other interface signals are ignored.
- Flow controller – Device (either the DMAC, or source/destination peripheral) that determines the length of a DMA block transfer and terminates it.
  - If you know the length of a block before enabling the channel, then you should program the DMAC as the flow controller.
  - If the length of a block is not known prior to enabling the channel, the source or destination peripheral needs to terminate a block transfer. In this mode, the peripheral is the flow controller.
- Flow control mode (CFGx.FCMODE) – Special mode that only applies when the destination peripheral is the flow controller. It controls the data pre-fetching from the source peripheral.
- FIFO mode – Special mode to improve bandwidth. When enabled, the channel waits until the FIFO is less than half full to fetch the data from the source peripheral, and waits until the FIFO is greater than or equal to half full in order to send data to the destination peripheral. Because of this, the channel can transfer the data using bursts, which eliminates the need to arbitrate for the AHB master interface in each single AHB transfer. When this mode is not enabled, the channel waits only until the FIFO can transmit or accept a single AHB transfer before it requests the master bus interface.

All information provided in this document is subject to legal disclaimers.
© REALTEK 2024. All rights reserved.

13.2.2 Features

The DMAC component includes the following features.

13.2.2.1 General

- Channels
  - Up to eight channels
  - Unidirectional channels – data transfers in one direction only
  - Programmable channel priority
- Secure mode support: channels can be configured to secure mode to enhance the security
- Power save: support DMAC low power mode (internal clock gating)
- FIFO per channel for source and destination, the FIFO depth for channel 0/1 is 128 bytes, while 32 bytes for other channels.
- Transaction
  - Size: the max. transfer size for one block is 65535.
  - Mode: support single block mode and multi-block mode.
  - Type: Support memory-to-memory, memory-to-peripheral, peripheral-to-memory, and peripheral-to-peripheral.

13.2.2.2 Address Generation

- Programmable source and destination addresses (on OCP/AXI bus)
- Address increment, decrement, or no change
- Multi-block transfers achieved through:
  - Linked Lists (block chaining)
  - Auto-reloading of channel registers
  - Contiguous address between blocks

13.2.2.3 Channel Control

- Programmable source and destination for each channel
- Programmable transfer type for each channel (memory-to-memory, memory-to-peripheral, peripheral-to-memory, and peripheral-to-peripheral)
- Each channel can be enabled/disabled, or set to secure/non-secure mode independently
- Support for disabling channel without data loss
- Support for suspension of DMA operation

13.2.2.4 Transfer Initiation

- Hardware handshaking interfaces for source and destination peripherals
  - UART0 Tx/Rx
  - UART1 Tx/Rx
  - UART2 Tx/Rx
  - SPI0 Tx/Rx
  - SPI1 Tx/Rx
  - SPIC Tx/Rx
  - SPORT0F0 Tx/Rx
  - SPORT0F1 Tx/Rx
  - SPORT1F0 Tx/Rx
  - SPORT1F1 Tx/Rx
  - LEDC Tx
  - I2C0 Tx/Rx
  - I2C1 Tx/Rx
- Handshaking interface supporting single or burst DMA transactions
- Polarity control for hardware handshaking interface
- Enabling and disabling of individual DMA handshaking interfaces

13.2.2.5 Flow Control

- Programmable flow control at block transfer level
- Software control of source data pre-fetch when destination is flow controller

All information provided in this document is subject to legal disclaimers.
© REALTEK 2024. All rights reserved.

13.2.2.6 Interrupts

- Combined and separate interrupt requests
- Interrupt generation on:
  - DMA transfer (multi-block) completion
  - Block transfer completion
  - Error condition
- Support of interrupt enabling and masking

13.3 Functional Description

This section describes the functional details of the DMAC component.

13.3.1 Block Flow Controller and Transfer Type

The device that controls the length of a block is known as the flow controller. Either the DMAC, the source peripheral, or the destination peripheral must be assigned as the flow controller.

- If the block size is known prior to when the channel is enabled, then the DMAC should be programmed as the flow controller. The block size should be programmed into the CTLx.BLOCK_TS field.
- If the block size is unknown when the DMAC channel is enabled, either the source or destination peripheral must be the flow controller.

The CTLx.TT_FC field indicates the transfer type and flow controller for that channel. *Table 13-1* lists valid transfer types and flow controller combinations.

Table 13-1 Transfer types and flow control combinations

Transfer type	Flow controller
Memory to Memory	DMAC
Memory to Peripheral	DMAC
Memory to Peripheral	Peripheral
Peripheral to Memory	DMAC
Peripheral to Memory	Peripheral
Peripheral to Peripheral	DMAC
Peripheral to Peripheral	Source Peripheral
Peripheral to Peripheral	Destination Peripheral

As an example, the DMAC can be programmed as the flow controller when a DMA block must be transferred from a receive peripheral to memory. Software then programs the CTLx.BLOCK_TS register with the same value and programs the DMAC as the flow controller.

13.3.2 Handshaking Interface

Handshaking interfaces are used at the transaction level to control the flow of single or burst transactions. The operation of the handshaking interface is different and depends on whether the peripheral or the DMAC is the flow controller.

The peripheral uses the handshaking interface to indicate to the DMAC that it is ready to transfer or accept data over the bus.

A non-memory peripheral can request a DMA transfer through the DMAC using hardware handshaking interfaces.

NOTE

Throughout the remainder of this document, references to both source and destination hardware handshaking interfaces assume an active-high interface (refer to CFGx.SRC(DST)_HS_POL bits in the Channel Configuration register). When active-low handshaking interfaces are used, then the active level and edge are reversed from that of an active-high interface.

The type of handshaking interface depends on whether the peripheral is a flow controller or not.

NOTE

Source and destination peripherals can independently select the handshaking interface type. For more information, refer to the CFGx.HS_SEL_SRC and CFGx.HS_SEL_DST parameters in the CFGx register.

All information provided in this document is subject to legal disclaimers.
© REALTEK 2024. All rights reserved.

13.3.3 Memory Peripherals

For memory, there is no handshaking interface with the DMAC, and therefore the memory peripheral can never be a flow controller. Once the channel is enabled, the transfer proceeds immediately without waiting for a transaction request.

13.3.4 Basic Interface Definitions

The following definitions are used in a data transaction.

- Source single transaction size in bytes
  ```
  src_single_size_bytes = CTLx.SRC_TR_WIDTH/8
  ```
- Source burst transaction size in bytes
  ```
  src_burst_size_bytes = CTLx.SRC_MSIZE * src_single_size_bytes
  ```
- Destination single transaction size in bytes
  ```
  dst_single_size_bytes = CTLx.DST_TR_WIDTH/8
  ```
- Destination burst transaction size in bytes
  ```
  dst_burst_size_bytes = CTLx.DST_MSIZE * dst_single_size_bytes
  ```
- Block size in bytes
  - DMAC is flow controller–With the DMAC as the flow controller, the processor programs the DMAC with the number of data items (block size) of source transfer width (CTLx.SRC_TR_WIDTH) to be transferred by the DMAC in a block transfer; this is programmed into the CTLx.BLOCK_TS field. Therefore, the total number of bytes to be transferred in a block is:
    ```
    blk_size_bytes_dma = CTLx.BLOCK_TS * src_single_size_bytes
    ```
  - Source peripheral is block flow controller
    ```
    blk_size_bytes_src = (Number of source burst transactions * src_burst_size_bytes) +
    (Number of source single transactions * src_single_size_bytes)
    ```
  - Destination peripheral is block flow controller
    ```
    blk_size_bytes_dst = (Number of destination burst transactions * dst_burst_size_bytes)
    + (Number of destination single transactions * dst_single_size_bytes)
    ```

13.3.5 Setting up Transfers

Transfers are set up by programming fields of the CTLx and CFGx registers for that channel. A single block is made up of numerous transactions – single and burst – which are in turn composed of OCP transfers. A peripheral requests a transaction through the handshaking interface to the DMAC. The operation of the handshaking interface is different and depends on what is acting as the flow controller.

Table 13-2 lists the parameters that are investigated in a transaction.

Table 13-2 Parameters used in a transaction

Parameter	Description
DMAC_CHx_FIFO_DEPTH	Channel x FIFO depth in bytes, 128 bytes for channel 0/1, while 32 bytes for others
CTLx.TT_FC	Transfer type and flow control
CTLx.BLOCK_TS	Block transfer size
CTLx.SRC_TR_WIDTH	Source transfer width
CTLx.DST_TR_WIDTH	Destination transfer width
CTLx.SRC_MSIZE	Source burst transaction length
CTLx.DEST_MSIZE	Destination burst transaction length
CFGx.FIFO_MODE	FIFO mode select
CFGx.FCMODE	Flow-control mode

13.3.6 Generating Requests for the Master Bus Interface

Each channel has a source state machine and destination state machine running in parallel. These state machines generate the request inputs to the arbiter, which arbitrates for the master bus interface (one arbiter per master bus interface).

When the source/destination state machine is granted control of the master bus interface, and when the master bus interface is granted control of the external bus, then transfers between the peripheral and the DMAC (on behalf of the granted state machine) can take place.

Transfers from the source peripheral or to the destination peripheral cannot proceed until the channel FIFO is ready. For burst transaction requests and for transfers involving memory peripherals, the criterion for “FIFO readiness” is controlled by the FIFO_MODE field of the CFGx register.

All information provided in this document is subject to legal disclaimers.
© REALTEK 2024. All rights reserved.

The definition of FIFO readiness is the same for:

- Single transactions
- Burst transactions, where CFGx.FIFO_MODE = 0
- Transfers involving memory peripherals, where CFGx.FIFO_MODE = 0

The channel FIFO is deemed ready when the space/data available is sufficient to complete a single transfer of the specified transfer width. FIFO readiness for source transfers occurs when the channel FIFO contains enough room to accept at least a single transfer of CTLx.SRC_TR_WIDTH width. FIFO readiness for destination transfers occurs when the channel FIFO contains data to form at least a single transfer of CTLx.DST_TR_WIDTH width.

NOTE

An exception to FIFO readiness for destination transfers occurs in "FIFO flush mode." In this mode, FIFO readiness for destination transfers occurs when the channel FIFO contains data to form at least a single transfer of CTLx.SRC_TR_WIDTH width (and not CTLx.DST_TR_WIDTH width, as is the normal case).

When CFGx.FIFO_MODE = 1, then the criteria for FIFO readiness for burst transaction requests and transfers involving memory peripherals are as follows:

- A FIFO is ready for a source burst transfer when the FIFO has enough space for a source burst.
- A FIFO is ready for a destination burst transfer when the FIFO is greater than or equal to a destination burst full.

Exceptions to this "readiness" occur. During these exceptions, a value of CTLx.FIFO_MODE = 0 is assumed. The following are the exceptions:

- Near the end of a burst transaction or block transfer – The channel source state machine does not wait for the channel FIFO to be less than half empty if the number of source data items left to complete the source burst transaction or source block transfer is less than DMAC_CHx_FIFO_DEPTH/2. Similarly, the channel destination state machine does not wait for the channel FIFO to be greater than or equal to half full, if the number of destination data items left to complete the destination burst transaction or destination block transfer is less than DMAC_CHx_FIFO_DEPTH/2.
- In FIFO flush mode
- When a channel is suspended – The destination state machine does not wait for the FIFO to become half empty to flush the FIFO, regardless of the value of the FIFO_MODE field.

When the source/destination peripheral is not memory, the source/destination state machine waits for a single/burst transaction request. Upon receipt of a transaction request and only if the channel FIFO is "ready" for source/destination transfers, a request for the master bus interface is made by the source/destination state machine.

NOTE

There is one exception to this, which occurs when the destination peripheral is the flow controller and CFGx.FCMODE = 1 (data pre-fetching is disabled). Then the source state machine does not generate a request for the master bus interface (even if the FIFO is "ready" for source transfers and has received a source transaction request) until the destination requests new data.

When the source/destination peripheral is memory, the source/destination state machine must wait until the channel FIFO is "ready." A request is then made for the master bus interface. There is no handshaking mechanism employed between a memory peripheral and the DMAC.

13.3.7 Secure Mode

DMAC supports secure mode, which can prevent non-secure access to secure range. When CPU configures the DMAC transfer without secure mode register access, the secure access bit will not be set. For AXI DMAC, master interface will output AxProt[1] =0 for secure access to source or destination address.

Generally, secure slave will response error data (for source) or reject the write (for destination) when non-secure access to it.

The work flow for secure DMAC transfer should be as follows:

(1) Set the secure access bit to 0 in configure register for some channel with secure mode write.
(2) Enable the DMAC transfer.
(3) Wait for DMAC transfer over or interrupt, then DMAC will clear channel enable bit automatically.
(4) Secure access bit should be cleared when the channel doesn't need secure access anymore.

Refer to the registers for more details.

13.4 Registers

This section describes the programmable registers of the DMAC.

All information provided in this document is subject to legal disclaimers.
© REALTEK 2024. All rights reserved.

NOTE

There are references to both software and hardware parameters throughout this chapter. The software parameters are the field names in each register description table and are prefixed by the register name; for example, the Block Transfer Size field in the Control register for Channel x is designated as "CTLx.BLOCK_TS".

Shipped with the DMAC component is an address definition (memory map) C header file. This can be used when the DMAC is programmed in a C environment.

13.4.1 Register Memory Map

Table 13-3 shows the memory map for the DMAC.

NOTE

The address offsets of the registers are always fixed, regardless of the number of configured channels.

Table 13-3 Memory map of DMAC

Name	Address offset	Access	Reset	Description
Channel Registers If DMAC_NUM_CHANNELS = "dnc", "Reg Exist" field shows "dnc" value for register to exist.				
SAR0	0x000	R/W	0x0	Channel 0 Source Address Register Reg Exist: Yes
DAR0	0x008	R/W	0x0	Channel 0 Destination Address Register Reg Exist: Yes
LLP0	0x010	R/W	0x0	Channel 0 Linked List Pointer Register Reg Exist: llp0_hc = False llp0_hc = DMAC_CH0_HC_LLP
CTL0	0x018	R/W	Configuration dependent	Channel 0 Control Register Reg Exist: Yes
SSTAT0	0x020	R/W	0x0	Channel 0 Source Status Register Reg Exist: sstat0 = True sstat0 = DMAC_CH0_STAT_SRC
DSTAT0	0x028	R/W	0x0	Channel 0 Destination Status Register Reg Exist: dstat0 = True dstat0 = DMAC_CH0_STAT_DST
SSTATAR0	0x030	R/W	0x0	Channel 0 Source Status Address Register Reg Exist: stat0 = True stat0 = DMAC_CH0_STAT_SRC
DSTATAR0	0x038	R/W	0x0	Channel 0 Destination Status Address Register Reg Exist: dstat0 = True dstat0 = DMAC_CH0_STAT_DST
CFG0	0x040	R/W	0x0000000400000e00	Channel 0 Configuration Register Reg Exist: Yes
SGR0	0x048	R/W	0x0	Channel 0 Source Gather Register Reg Exist: sgr0 = True sgr0 = DMAC_CH0_SRC_GAT_EN
DSR0	0x050	R/W	0x0	Channel 0 Destination Scatter Register Reg Exist: dsr0 = True dsr0 = DMAC_CH0_DST_SCA_EN
SAR1	0x058	R/W	0x0	Channel 1 Source Address Register Reg Exist: dnc $\geqslant$ 2
DAR1	0x060	R/W	0x0	Channel 1 Destination Address Register Reg Exist: dnc $\geqslant$ 2
LLP1	0x068	R/W	0x0	Channel 1 Linked List Pointer Register Reg Exist: dnc $\geqslant$ 2 and llp1_hc = False
CTL1	0x070	R/W	Configuration dependent	Channel 1 Control Register Reg Exist: dnc $\geqslant$ 2
SSTAT1	0x078	R/W	0x0	Channel 1 Source Status Register Reg Exist: dnc $\geqslant$ 2 and sstat1 = True
DSTAT1	0x080	R/W	0x0	Channel 1 Destination Status Register Reg Exist: dnc $\geqslant$ 2 and dstat1 = True
SSTATAR1	0x088	R/W	0x0	Channel 1 Source Status Address Register Reg Exist: dnc $\geqslant$ 2 and sstat1 = True
DSTATAR1	0x090	R/W	0x0	Channel 1 Destination Status Address Register Reg Exist: dnc $\geqslant$ 2 and dstat1 = True
CFG1	0x098	R/W	0x0000000800000e00	Channel 1 Configuration Register Reg Exist: dnc $\geqslant$ 2

All information provided in this document is subject to legal disclaimers.
© REALTEK 2024. All rights reserved.

SGR1	0x0a0	R/W	0x0	Channel 1 Source Gather Register Reg Exist: dnc ≥ 2 and sgr1 = True
DSR1	0x0a8	R/W	0x0	Channel 1 Destination Scatter Register Reg Exist: dnc ≥ 2 and dsr1 = True
SAR2	0x0b0	R/W	0x0	Channel 2 Source Address Register Reg Exist: dnc ≥ 3
DAR2	0x0b8	R/W	0x0	Channel 2 Destination Address Register Reg Exist: dnc ≥ 3
LLP2	0x0c0	R/W	0x0	Channel 2 Linked List Pointer Register Reg Exist: dnc ≥ 3 and llp2_hc = False
CTL2	0x0c8	R/W	Configuration dependent	Channel 2 Control Register Reg Exist: dnc ≥ 3
SSTAT2	0x0d0	R/W	0x0	Channel 2 Source Status Register Reg Exist: dnc ≥ 3 and sstat2 = True
DSTAT2	0x0d8	R/W	0x0	Channel 2 Destination Status Register Reg Exist: dnc ≥ 3 and dstat2 = True
SSTATAR2	0x0e0	R/W	0x0	Channel 2 Source Status Address Register Reg Exist: dnc ≥ 3 and sstat2 = True
DSTATAR2	0x0e8	R/W	0x0	Channel 2 Destination Status Address Register Reg Exist: dnc ≥ 3 and dstat2 = True
CFG2	0x0f0	R/W	0x0000000800000e40	Channel 2 Configuration Register Reg Exist: dnc ≥ 3
SGR2	0x0f8	R/W	0x0	Channel 2 Source Gather Register Reg Exist: dnc ≥ 3 and sgr2 = True
DSR2	0x100	R/W	0x0	Channel 2 Destination Scatter Register Reg Exist: dnc ≥ 3 and dsr2 = True
SAR3	0x108	R/W	0x0	Channel 3 Source Address Register Reg Exist: dnc ≥ 4
DAR3	0x110	R/W	0x0	Channel 3 Destination Address Register Reg Exist: dnc ≥ 4
LLP3	0x118	R/W	0x0	Channel 3 Linked List Pointer Register Reg Exist: dnc ≥ 4 and llp3_hc = False
CTL3	0x120	R/W	Configuration dependent	Channel 3 Control Register Reg Exist: dnc ≥ 4
SSTAT3	0x128	R/W	0x0	Channel 3 Source Status Register Reg Exist: dnc ≥ 4 and sstat3 = True
DSTAT3	0x130	R/W	0x0	Channel 3 Destination Status Register Reg Exist: dnc ≥ 4 and dstat3 = True
SSTATAR3	0x138	R/W	0x0	Channel 3 Source Status Address Register Reg Exist: dnc ≥ 4 and sstat3 = True
DSTATAR3	0x140	R/W	0x0	Channel 3 Destination Status Address Register Reg Exist: dnc ≥ 4 and dstat3 = True
CFG3	0x148	R/W	0x0000000800000e60	Channel 3 Configuration Register Reg Exist: dnc ≥ 4
SGR3	0x150	R/W	0x0	Channel 3 Source Gather Register Reg Exist: dnc ≥ 4 and sgr3 = True
DSR3	0x158	R/W	0x0	Channel 3 Destination Scatter Register Reg Exist: dnc ≥ 4 and dsr3 = True
SAR4	0x160	R/W	0x0	Channel 4 Source Address Register Reg Exist: dnc ≥ 5
DAR4	0x168	R/W	0x0	Channel 4 Destination Address Register Reg Exist: dnc ≥ 5
LLP4	0x170	R/W	0x0	Channel 4 Linked List Pointer Register Reg Exist: dnc ≥ 5 and llp4_hc = False
CTL4	0x178	R/W	Configuration dependent	Channel 4 Control Register Reg Exist: dnc ≥ 5
SSTAT4	0x180	R/W	0x0	Channel 4 Source Status Register Reg Exist: dnc ≥ 5 and sstat4 = True
DSTAT4	0x188	R/W	0x0	Channel 4 Destination Status Register Reg Exist: dnc ≥ 5 and dstat4 = True
SSTATAR4	0x190	R/W	0x0	Channel 4 Source Status Address Register Reg Exist: dnc ≥ 5 and sstat4 = True
DSTATAR4	0x198	R/W	0x0	Channel 4 Destination Status Address Register Reg Exist: dnc ≥ 5 and dstat4 = True
CFG4	0x1a0	R/W	0x0000000800000e00	Channel 4 Configuration Register Reg Exist: dnc ≥ 5

All information provided in this document is subject to legal disclaimers.
© REALTEK 2024. All rights reserved.

SGR4	0x1a8	R/W	0x0	Channel 4 Source Gather Register Reg Exist: dnc ≥ 5 and sgr4 = True
DSR4	0x1b0	R/W	0x0	Channel 4 Destination Scatter Register Reg Exist: dnc ≥ 5 and dsr4= True
SAR5	0x1b8	R/W	0x0	Channel 5 Source Address Register Reg Exist: dnc ≥ 6
DAR5	0x1c0	R/W	0x0	Channel 5 Destination Address Register Reg Exist: dnc ≥ 6
LLP5	0x1c8	R/W	0x0	Channel 5 Linked List Pointer Register Reg Exist: dnc ≥ 6 and llp5_hc = False
CTL5	0x1d0	R/W	Configuration dependent	Channel 5 Control Register Reg Exist: dnc ≥ 6
SSTAT5	0x1d8	R/W	0x0	Channel 5 Source Status Register Reg Exist: dnc ≥ 6 and sstat5 = True
DSTAT5	0x1e0	R/W	0x0	Channel 5 Destination Status Register Reg Exist: dnc ≥ 6 and dstat5 = True
SSTATAR5	0x1e8	R/W	0x0	Channel 5 Source Status Address Register Reg Exist: dnc ≥ 6 and sstat5 = True
DSTATAR5	0x1f0	R/W	0x0	Channel 5 Destination Status Address Register Reg Exist: dnc ≥ 6 and dstat5 = True
CFG5	0x1f8	R/W	0x0000000800000e00	Channel 5 Configuration Register Reg Exist: dnc ≥ 6
SGR5	0x200	R/W	0x0	Channel 5 Source Gather Register Reg Exist: dnc ≥ 6 and sgr5 = True
DSR5	0x208	R/W	0x0	Channel 5 Destination Scatter Register Reg Exist: dnc ≥ 6 and dsr5= True
SAR6	0x210	R/W	0x0	Channel 6 Source Address Register Reg Exist: dnc ≥ 7
DAR6	0x218	R/W	0x0	Channel 6 Destination Address Register Reg Exist: dnc ≥ 7
LLP6	0x220	R/W	0x0	Channel 6 Linked List Pointer Register Reg Exist: dnc ≥ 7 and llp6_hc = False
CTL6	0x228	R/W	Configuration dependent	Channel 6 Control Register Reg Exist: dnc ≥ 7
SSTAT6	0x230	R/W	0x0	Channel 6 Source Status Register Reg Exist: dnc ≥ 7 and sstat6 = True
DSTAT6	0x238	R/W	0x0	Channel 6 Destination Status Register Reg Exist: dnc ≥ 7 and dstat6 = True
SSTATAR6	0x240	R/W	0x0	Channel 6 Source Status Address Register Reg Exist: dnc ≥ 7 and sstat6 = True
DSTATAR6	0x248	R/W	0x0	Channel 6 Destination Status Address Register Reg Exist: dnc ≥ 7 and dstat6 = True
CFG6	0x250	R/W	0x0000000800000e00	Channel 6 Configuration Register Reg Exist: dnc ≥ 7
SGR6	0x258	R/W	0x0	Channel 6 Source Gather Register Reg Exist: dnc ≥ 7 and sgr6 = True
DSR6	0x260	R/W	0x0	Channel 6 Destination Scatter Register Reg Exist: dnc ≥ 7 and dsr6= True
SAR7	0x268	R/W	0x0	Channel 7 Source Address Register Reg Exist: dnc = 8
DAR7	0x270	R/W	0x0	Channel 7 Destination Address Register Reg Exist: dnc = 8
LLP7	0x278	R/W	0x0	Channel 7 Linked List Pointer Register Reg Exist: dnc = 8 and llp7_hc = False
CTL7	0x280	R/W	Configuration dependent	Channel 7 Control Register Reg Exist: dnc = 8
SSTAT7	0x288	R/W	0x0	Channel 7 Source Status Register Reg Exist: dnc = 8 and sstat7 = True
DSTAT7	0x290	R/W	0x0	Channel 7 Destination Status Register Reg Exist: dnc = 8 and dstat7 = True
SSTATAR7	0x298	R/W	0x0	Channel 7 Source Status Address Register Reg Exist: dnc = 8 and sstat7 = True
DSTATAR7	0x2a0	R/W	0x0	Channel 7 Destination Status Address Register Reg Exist: dnc = 8 and dstat7 = True
CFG7	0x2a8	R/W	0x0000000800000e00	Channel 7 Configuration Register Reg Exist: dnc = 8
SGR7	0x2b0	R/W	0x0	Channel 7 Source Gather Register

All information provided in this document is subject to legal disclaimers.
© REALTEK 2024. All rights reserved.

				Reg Exist: dnc = 8 and sgr7 = True
DSR7	0x2b8	R/W	0x0	Channel 7 Destination Scatter Register Reg Exist: dnc = 8 and dsr7= True
Interrupt Registers RawBlock, RawDstTran, RawErr, RawSrcTran, RawTfr				
RawTfr	0x2c0	R	0x0	Raw Status for IntTfr Interrupt Reg Exist: Yes
RawBlock	0x2c8	R	0x0	Raw Status for IntBlock Interrupt Reg Exist: Yes
RawSrcTran	0x2d0	R	0x0	Raw Status for IntSrcTran Interrupt Reg Exist: Yes
RawDstTran	0x2d8	R	0x0	Raw Status for IntDstTran Interrupt Reg Exist: Yes
RawErr	0x2e0	R	0x0	Raw Status for IntErr Interrupt Reg Exist: Yes
StatusBlock, StatusDstTran, StatusErr, StatusSrcTran, StatusTfr				
StatusTfr	0x2e8	R	0x0	Status for IntTfr Interrupt Reg Exist: Yes
StatusBlock	0x2f0	R	0x0	Status for IntBlock Interrupt Reg Exist: Yes
StatusSrcTran	0x2f8	R	0x0	Status for IntSrcTran Interrupt Reg Exist: Yes
StatusDstTran	0x300	R	0x0	Status for IntDstTran Interrupt Reg Exist: Yes
StatusErr	0x308	R	0x0	Status for IntErr Interrupt Reg Exist: Yes
MaskBlock, MaskDstTran, MaskErr, MaskSrcTran, MaskTfr				
MaskTfr	0x310	R/W	0x0	Mask for IntTfr Interrupt Reg Exist: Yes
MaskBlock	0x318	R/W	0x0	Mask for IntBlock Interrupt Reg Exist: Yes
MaskSrcTran	0x320	R/W	0x0	Mask for IntSrcTran Interrupt Reg Exist: Yes
MaskDstTran	0x328	R/W	0x0	Mask for IntDstTran Interrupt Reg Exist: Yes
MaskErr	0x330	R/W	0x0	Mask for IntErr Interrupt Reg Exist: Yes
ClearBlock, ClearDstTran, ClearErr, ClearSrcTran, ClearTfr				
ClearTfr	0x338	W	0x0	Clear for IntTfr Interrupt Reg Exist: Yes
ClearBlock	0x340	W	0x0	Clear for IntBlock Interrupt Reg Exist: Yes
ClearSrcTran	0x348	W	0x0	Clear for IntSrcTran Interrupt Reg Exist: Yes
ClearDstTran	0x350	W	0x0	Clear for IntDstTran Interrupt Reg Exist: Yes
ClearErr	0x358	W	0x0	Clear for IntErr Interrupt Reg Exist: Yes
StatusInt	0x360	W	0x0	Status for each interrupt type Reg Exist: Yes
Miscellaneous Registers				
DmaCfgReg	0x398	R/W	0x0	DMA Configuration Register Reg Exist: Yes
ChEnReg	0x3a0	R/W	0x0	DMA Channel Enable Register Reg Exist: Yes
DmaIdReg	0x3a8	R	0x0	DMA ID Register Reg Exist: Yes
DmaTestReg	0x3b0	R/W	0x0	DMA Test Register Reg Exist: Yes
Reserved	0x3b8	N/A	-	Reserved
Reserved	0x3c0	N/A	-	Reserved
DMA_COMP_PARAMS_6	0x3c8	R	Depending on user configuration	Refer to the bit table in the description for DMA_COMP_PARAMS_6. Reg Exist: Depending on the setting of DMAC_ADD_ENCODED_PARAMS configuration parameter. The register exists when this parameter is set to True.
DMA_COMP_PARAMS_5	0x3d0	R	Depending on user configuration	Refer to the bit table in the description for DMA_COMP_PARAMS_5. Reg Exist: Depending on the setting of DMAC_ADD_ENCODED_PARAMS configuration parameter. The register exists when this parameter is set to

All information provided in this document is subject to legal disclaimers.
© REALTEK 2024. All rights reserved.

				True.
DMA_COMP_PARAMS_4	0x3d8	R	Depending on user configuration	Refer to the bit table in the description for DMA_COMP_PARAMS_4. Reg Exist: Depending on the setting of DMAC_ADD_ENCODED_PARAMS configuration parameter. The register exists when this parameter is set to True.
DMA_COMP_PARAMS_3	0x3e0	R	Depending on user configuration	Refer to the bit table in the description for DMA_COMP_PARAMS_3. Reg Exist: Depending on the setting of DMAC_ADD_ENCODED_PARAMS configuration parameter. The register exists when this parameter is set to True.
DMA_COMP_PARAMS_2	0x3e8	R	Depending on user configuration	Refer to the bit table in the description for DMA_COMP_PARAMS_2. Reg Exist: Depending on the setting of DMAC_ADD_ENCODED_PARAMS configuration parameter. The register exists when this parameter is set to True.
DMA_COMP_PARAMS_1	0x3f0	R	Depending on user configuration	Refer to the bit table in the description for DMA_COMP_PARAMS_1 Reg Exist: Depending on the setting of DMAC_ADD_ENCODED_PARAMS configuration parameter. The register exists when this parameter is set to True.
DMA Component ID Register	0x3f8	R	See description	Component version register. Please refer to the bit table in the description for DMA Component ID Register. Reg Exist: Yes

13.4.2 Registers and Field Descriptions

The following sections contain the memory diagrams and field descriptions for the individual registers.

13.4.2.1 Configuration and Channel Enable Registers

The channel registers consist of the following, where x = 0 to 7:

- DmaCfgReg – Configuration Register
- ChEnReg – Channel Enable Register

13.4.2.1.1 DmaCfgReg

- Name: DMA Configuration Register
- Size: 64 bits
- Address offset: 0x398
- Read/write access: read/write

This register is used to enable the DMAC, which must be done before any channel activity can begin.

Bit	Name	Access	Reset	Description
63:1	RSVD	N/A	0x0	Reserved
0	DMA_EN	R/W	0x0	DMAC Enable bit. 0 = DMAC disabled 1 = DMAC enabled

This register can only be set or cleared by secure CPU.

If this register is cleared while any channel is still active, then DmaCfgReg.DMA_EN still returns 1 to indicate that there are channels still active.

13.4.2.1.2 ChEnReg

- Name: DMA Channel Enable Register
- Size: 64 bits
- Address offset: 0x3A0
- Read/write access: read/write

This is the DMAC Channel Enable Register. If software needs to set up a new channel, then it can read this register in order to find out which channels are currently inactive; it can then enable an inactive channel with the required priority.

Bit	Name	Access	Reset	Description

All information provided in this document is subject to legal disclaimers.
© REALTEK 2024. All rights reserved.

63:16	RSVD	N/A	0x0	Reserved
15:8	CH_EN_WE	W	0x0	Channel Enable Write Enable.
7:0	CH_EN	R/W	0x0	Enables/Disables the channel. Setting this bit enables a channel; clearing this bit disables the channel. 0 = Disable the Channel 1 = Enable the Channel The ChEnReg.CH_EN bit is automatically cleared by hardware to disable the channel after the last AMBA transfer of the DMA transfer to the destination has completed. Software can therefore poll this bit to determine when this channel is free for a new DMA transfer.

- When the DmaCfgReg.DMA_EN bit is 0, a write to the ChEnReg register is ignored.
- When any channel is working, a write to DmaCfgReg register is ignored.
- When a channel is set to a secure channel (CFGx.PROTCTL is 0), the value read by non-secure CPU of this channel's CH_EN bit is always 1.

The channel enable bit, ChEnReg.CH_EN, is written only if the corresponding channel write enable bit, ChEnReg.CH_EN_WE, is asserted on the same write transfer. For example, writing hex 00xx01x1 writes a 1 into ChEnReg[0], while ChEnReg[7:1] and ChEnReg[23:16] remains unchanged. Writing hex 00xx00xx leaves ChEnReg[7:0] and ChEnReg[23:16] unchanged. Note that a read-modified write is not required.

13.4.2.2 Channel Registers

The channel registers consist of the following, where x = 0 to 7:
- CFGx – Configuration register for channel x
- CTLx – Control register for channel x
- DARx – Destination address register for channel x
- DSRx – Destination scatter register for channel x
- DSTATx – Destination status register for channel x
- DSTATARx – Destination status address register for channel x
- LLPx – Linked list pointer register for channel x
- SARx – Source address register for channel x
- SGRx – Source gather register for channel x
- SSTATx – Source status register for channel x
- SSTATARx – Source status address register for channel x

The SARx, DARx, LLPx, CTLx, and CFGx channel registers should be programmed prior to enabling the channel. It is illegal access when a write to the SARx, DARx, LLPx, CTLx, SSTATx, DSTATx, SSTATARx or DSTATARx registers occurs when the channel is enabled.

13.4.2.2.1 SARx

- Name: Source Address Register for Channel x
- Size: 64 bits (upper 32 bits are reserved)
- Address offset: for x = 0 to 7:
  - SAR0 – 0x000
  - SAR1 – 0x058
  - SAR2 – 0x0B0
  - SAR3 – 0x108
  - SAR4 – 0x160
  - SAR5 – 0x1B8
  - SAR6 – 0x210
  - SAR7 – 0x268
- Read/write access: read/write

The starting source address is programmed by software before the DMA channel is enabled, or by an LLI update before the start of the DMA transfer. While the DMA transfer is in progress, this register is updated to reflect the source address of the current OCP or AXI transfer.

NOTE

You must program the SAR address to be aligned to CTLx.SRC_TR_WIDTH.

For information on how the SARx is updated at the start of each DMA block for multi-block transfers, refer to *Table 13-7*.

Bit	Name	Access	Reset	Description

All information provided in this document is subject to legal disclaimers.
© REALTEK 2024. All rights reserved.

63:32	RSVD	N/A	0x0	Reserved
31:0	SAR	R/W	0x0	Source Address of DMA transfer. Updated after each source transfer. The SINC field in the CTLx register determines whether the address increments, decrements, or is left unchanged on every source transfer throughout the block transfer.

13.4.2.2.2 DARx

- Name: Destination Address Register for Channel x
- Size: 64 bits (upper 32 bits are reserved)
- Address offset: for x = 0 to 7:
  - DAR0 – 0x008
  - DAR1 – 0x060
  - DAR2 – 0x0B8
  - DAR3 – 0x110
  - DAR4 – 0x168
  - DAR5 – 0x1C0
  - DAR6 – 0x218
  - DAR7 – 0x270
- Read/write access: read/write

The starting destination address is programmed by software before the DMA channel is enabled, or by an LLI update before the start of the DMA transfer. While the DMA transfer is in progress, this register is updated to reflect the destination address of the current OCP or AXI transfer.

NOTE

You must program the DAR address to be aligned to CTLx.DST_TR_WIDTH.

For information on how the DARx is updated at the start of each DMA block for multi-block transfers, refer to *Table 13-7*.

Bit	Name	Access	Reset	Description
63:32	RSVD	N/A	0x0	Reserved
31:0	DAR	R/W	0x0	Destination Address of DMA transfer. Updated after each destination transfer. The DINC field in the CTLx register determines whether the address increments, decrements, or is left unchanged on every destination transfer throughout the block transfer.

The destination address is automatically realigned by the DMAC in the following DMA transfer setup scenario:
- Contiguous multi-block transfers on destination side, AND
- DST_TR_WIDTH > SRC_TR_WIDTH, AND
- (BLOCK_TS * SRC_TR_WIDTH)/DST_TR_WIDTH != integer (where SRC_TR_WIDTH, DST_TR_WIDTH is byte width of transfer)

13.4.2.2.3 LLPx

- Name: Linked List Pointer Register for Channel x
- Size: 64 bits (upper 32 bits are reserved)
- Address offset: for x = 0 to 7:
  - LLP0 – 0x010
  - LLP1 – 0x068
  - LLP2 – 0x0C0
  - LLP3 – 0x118
  - LLP4 – 0x170
  - LLP5 – 0x1C8
  - LLP6 – 0x220
  - LLP7 – 0x278
- Read/write access: read/write

This register does not exist if the DMAC_CHx_HC_LLP configuration parameter is set to True.

NOTE

You need to program this register to point to the first Linked List Item (LLI) in memory prior to enabling the channel if block chaining is enabled – rows 6 to 10 of Table 13-7.

Bit	Name	Access	Reset	Description

All information provided in this document is subject to legal disclaimers.
© REALTEK 2024. All rights reserved.

63:32	RSVD	N/A	0x0	Reserved
31:2	LOC	R/W	0x0	Starting Address in Memory of next LLI if block chaining is enabled. NOTE *The two LSBs of the starting address are not stored because the address is assumed to be aligned to a 32-bit boundary. LLI accesses are always 32-bit accesses (Hsize = 2) aligned to 32-bit boundaries and cannot be changed or programmed to anything other than 32- bit.*
1:0	LMS	R/W	0x0	List Master Select. Identifies the OCP/AXI layer/interface where the memory device that stores the next linked list item resides.

The LLP register has two functions:

- The logical result of the equation LLP.LOC != 0 is used to set up the type of DMA transfer—single or multi-block. *Table 13-7* shows how the method of updating the channel registers is a function of LLP.LOC != 0. If LLP.LOC is set to 0x0, then transfers using linked lists are not enabled. This register must be programmed prior to enabling the channel in order to set up the transfer type.
- LLP.LOC != 0 contains the pointer to the next LLI for block chaining using linked lists. The LLPx register can also point to the address where write-back of the control and source/destination status information occur after block completion.

13.4.2.2.4 CTLx

- Name: Control Register for Channel x
- Size: 64 bits
- Address offset: for x = 0 to 7:
  - CTL0 – 0x018
  - CTL1 – 0x070
  - CTL2 – 0x0C8
  - CTL3 – 0x120
  - CTL4 – 0x178
  - CTL5 – 0x1D0
  - CTL6 – 0x228
  - CTL7 – 0x280
- Read/write access: read/write

This register contains fields that control the DMA transfer.

NOTE

You need to program this register prior to enabling the channel.

Bit	Name	Access	Reset	Description
63:32	TRANS_DATA_CNT	R		Data Transferred Count. Once the transfer starts, the read-back value is the total number of data bytes already write to destination memory or peripheral.
b:32 (See description)	BLOCK_TS	R/W	0x2	Block Transfer Size. When the DMAC is the flow controller, the user writes this field before the channel is enabled in order to indicate the block size. The number programmed into BLOCK_TS indicates the total number of single transactions to perform for every block transfer; a single transaction is mapped to a single OCP beat. This field is write-only, number of block size programmed into CTLx.BLOCK_TS can be read from SGRx.BLOCK_TS. Width: The width of the single transaction is determined by CTLx.SRC_TR_WIDTH. b = log2(DMAC_CHx_MAX_BLK_SIZE + 1) + 31
31:29	RSVD	N/A	0x0	Reserved
28	LLP_SRC_EN	R/W	0x0	Block chaining is enabled on the source side only if the LLP_SRC_EN field is high and LLPx.LOC is non-zero.
27	LLP_DST_EN	R/W	0x0	Block chaining is enabled on the destination side only if the LLP_DST_EN field is high and LLPx.LOC is non-zero.
26:23	RSVD	N/A	0x0	Reserved
22:20	TT_FC	R/W	See description	Transfer Type and Flow Control. The following transfer types are supported. ● Memory to Memory ● Memory to Peripheral

All information provided in this document is subject to legal disclaimers.
© REALTEK 2024. All rights reserved.

				● Peripheral to Memory ● Peripheral to Peripheral Flow Control can be assigned to the DMAC, the source peripheral, or the destination peripheral. *Table 13-6* lists the decoding for this field. Reset Value: Configuration dependent: TT_FC[0] = 1'b1 TT_FC[1] = DMAC_CHx_FC[1] || (!DMAC_CHx_FC[0]) TT_FC[2] = DMAC_CHx_FC[1] ^ DMAC_CHx_FC[0] Dependencies: If hardware handshake is not supported, TT_FC does not work no matter what value in it.
19	WRNP_EN	R/W	0x0	OCP non-posted write enable bit: 0 = non-posted write disabled 1 = non-posted write enabled If set to 1, and DMAC master interface is OCP protocol, every write command of the specified channel's block transfer will be non-posted write. DMAC does not start a new write transfer until accepting the previous write's response. If destination is peripheral, and data transferred by hardware handshake, DMAC will not assert dma_ack signal until accepting the previous write's response. If the destination memory or peripheral does not support non-posted write, this field cannot be set to 1. Dependencies: This field does not exist if AXI master interface is selected.
18	DST_SCATTER_EN	R/W	0x0	Destination scatter enable bit: 0 = Scatter disabled 1 = Scatter enabled Scatter on the destination side is applicable only when the CTLx.DINC bit indicates an incrementing or decrementing address control. Dependencies: This field does not exist if DMAC_CHx_DST_SCA_EN is not selected.
17	SRC_GATHER_EN	R/W	0x0	Source gather enable bit: 0 = Gather disabled 1 = Gather enabled Gather on the source side is applicable only when the CTLx.SINC bit indicates an incrementing or decrementing address control. Dependencies: This field does not exist if DMAC_CHx_SRC_GAT_EN is not selected.
16:14	SRC_MSIZE	R/W	0x1	Source Burst Transaction Length. Number of data items, each of width CTLx.SRC_TR_WIDTH, to be read from the source every time a source burst transaction request is made from the corresponding hardware handshaking interface. *Table 13-4* lists the decoding for this field. NOTE *This value is not related to the OCP/AXI bus width.* Dependencies: DMAC_CHx_MAX_MULT_SIZE determines the bit width of this field. All remaining bits in this field do not exist and read back as 0.
13:11	DEST_MSIZE	R/W	0x1	Destination Burst Transaction Length. Number of data items, each of width CTLx.DST_TR_WIDTH, to be written to the destination every time a destination burst transaction request is made from the corresponding hardware handshaking interface. *Table 13-4* lists the decoding for this field. NOTE *This value is not related to the OCP bus width.* Dependencies: DMAC_CHx_MAX_MULT_SIZE determines the bit width of this field. All surplus bits in this field do not exist and read back as 0.
10:9	SINC	R/W	0x0	Source Address Increment. Indicates whether to increment or decrement the source address on every source transfer. If the device is fetching data from a source peripheral FIFO with a fixed address, then set this field to "No change." 00 = Increment 01 = Decrement

All information provided in this document is subject to legal disclaimers.
© REALTEK 2024. All rights reserved.

				1x = No change NOTE *Incrementing or decrementing is done for alignment to the next CTLx.SRC_TR_WIDTH boundary.*
8:7	DINC	R/W	0x0	Destination Address Increment. Indicates whether to increment or decrement the destination address on every destination transfer. If your device is writing data to a destination peripheral FIFO with a fixed address, then set this field to "No change." 00 = Increment 01 = Decrement 1x = No change NOTE *Incrementing or decrementing is done for alignment to the next CTLx.DST_TR_WIDTH boundary.*
6:4	SRC_TR_WIDTH	R/W	See description	Source Transfer Width. *Table 13-5* lists the decoding for this field. It can't be mapped to OCP bus data width directly. Since OCP bus data width is fixed (such as 32bit/64bit), OCP uses byte enable signal for byte/half-word transfer (still burst); so SRC_TR_WIDTH is only used for total block data size calculation and streaming mode byte/half-word transfer (for FIFO transfer). If SRC_TR_WIDTH is bigger than OCP bus width, DMA will always use OCP bus width for transfer. Reset Value: Encoded value; refer to *Table 13-5*. Dependencies: This field does not exist if DMAC_CHx_STW is hardcoded.
3:1	DST_TR_WIDTH	R/W	See description	Destination Transfer Width. *Table 13-5* lists the decoding for this field. It can't be mapped to OCP bus data width directly. For a non-memory peripheral, typically the peripheral (destination) FIFO width. Reset Value: Encoded value; refer to *Table 13-5*. Dependencies: This field does not exist if DMAC_CHx_DTW is hardcoded.
0	INT_EN	R/W	0x1	Interrupt Enable Bit. If set, then all interrupt-generating sources are enabled. Functions as a global mask bit for all interrupts for the channel; raw* interrupt registers still assert if CTLx.INT_EN = 0.

Table 13-4 CTLx.SRC_MSIZE and DEST_MSIZE decoding

CTLx.SRC_MSIZE/CTLx.DEST_MSIZE	Number of data items to be transferred (of width CTLx.SRC_TR_WIDTH or CTLx.DST_TR_WIDTH)
000	1
001	4
010	8
011	16
100	32
101	64
110	128
111	256

Table 13-5 CTLx.SRC_TR_WIDTH and CTLx.DST_TR_WIDTH decoding

CTLx.SRC_TR_WIDTH/TLx.DST_TR_WIDTH	Size (bits)
000	8
001	16
010	32
011	64
100	128
101	256
11x	256

Table 13-6 CTLx.TT_FC field decoding

CTLx.TT_FC field	Transfer type	Flow controller
000	Memory to Memory	DMAC

All information provided in this document is subject to legal disclaimers.
© REALTEK 2024. All rights reserved.

001	Memory to Peripheral	DMAC
010	Peripheral to Memory	DMAC
011	Peripheral to Peripheral	DMAC
100	Peripheral to Memory	Peripheral
101	Peripheral to Peripheral	Source Peripheral
110	Memory to Peripheral	Peripheral
111	Peripheral to Peripheral	Destination Peripheral

NOTE

If transfer type is Peripheral to Peripheral, and Destination is flow Controller (CTLx.TT_FC=111), CTLx.SRC_MSIZE and CTLx.DST_MSIZE must be 0, and CTLx.SRC_TR_WIDTH must be equal to CTLx.DST_TR_WIDTH.

13.4.2.2.5 CFGx

- Name: Configuration Register for Channel x
- Size: 64 bits (upper 32 bits are reserved)
- Address offset: for x = 0 to 7:
  - CFG0 – 0x040
  - CFG1 – 0x098
  - CFG2 – 0x0F0
  - CFG3 – 0x148
  - CFG4 – 0x1A0
  - CFG5 – 0x1F8
  - CFG6 – 0x250
  - CFG7 – 0x2A8
- Read/write access: read/write

This register contains fields that configure the DMA transfer. The channel configuration register remains fixed for all blocks of a multi-block transfer.

NOTE

You need to program this register prior to enabling the channel.

Bit	Name	Access	Reset	Description
63:53	RSVD	N/A	0x0	Reserved
52	Extended DEST_PER3	R/W	0x0	The extended 6th bit of DEST_PER when hardware handshake interfaces number is configured to 128.
51	Extended SRC_PER3	R/W	0x0	The extended 6th bit of SRC_PER when hardware handshake interfaces number is configured to 128.
50	Extended DEST_PER2	R/W	0x0	The extended 5th bit of DEST_PER when hardware handshake interfaces number is configured to 64.
49	Extended SRC_PER2	R/W	0x0	The extended 5th bit of SRC_PER when hardware handshake interfaces number is configured to 64.
48	Extended DEST_PER1	R/W	0x0	The extended 4th bit of DEST_PER when hardware handshake interfaces number is configured to 32.
47	Extended SRC_PER1	R/W	0x0	The extended 4th bit of SRC_PER when hardware handshake interfaces number is configured to 32.
b:43 (See notes)	DEST_PER	R/W	0x0	Assigns a hardware handshaking interface (0 - DMAC_NUM_HS_INT-1) to the destination of channel x if the CFGx.HS_SEL_DST field is 0; otherwise, this field is ignored. The channel can then communicate with the destination peripheral connected to that interface through the assigned hardware handshaking interface. NOTE ■ *For correct DMA operation, only one peripheral (source or destination) should be assigned to the same handshaking interface.* ■ *This field does not exist if the configuration parameter DMAC_NUM_HS_INT is set to 0.* ■ *b = 43 if DMAC_NUM_HS_INT is 1; b = ceil(log2(DMAC_NUM_HS_INT)) + 42 if DMAC_NUM_HS_INT is greater than 1.* Bits 46:(b+1) do not exist and return 0 on a read.
b:39	SRC_PER	R/W	0x0	Assigns a hardware handshaking interface

All information provided in this document is subject to legal disclaimers.
© REALTEK 2024. All rights reserved.

(See notes)				(0 - DMAC_NUM_HS_INT-1) to the source of channel x if the CFGx.HS_SEL_SRC field is 0; otherwise, this field is ignored. The channel can then communicate with the source peripheral connected to that interface through the assigned hardware handshaking interface. NOTE ■ *For correct DMA operation, only one peripheral (source or destination) should be assigned to the same handshaking interface.* ■ *This field does not exist if the configuration parameter DMAC_NUM_HS_INT is set to 0.* ■ *b = 39 if DMAC_NUM_HS_INT is 1; b = ceil(log2(DMAC_NUM_HS_INT)) + 38 if DMAC_NUM_HS_INT is greater than 1.* Bits 42:(b+1) do not exist and return 0 on a read.
38:36	RSVD	N/A	0x0	Reserved
35	PROTCTL	R/W	0x1	Protection Control Set this bit to 0 will enable the channel secure DMA transfer. Set this bit to 1 will enable the channel non-secure DMAC transfer.
34	RSVD	N/A	0x0	Reserved
33	FIFO_MODE	R/W	0x0	FIFO Mode Select. Determines how much space or data needs to be available in the FIFO before a burst transaction request is serviced. 0 = Space/data available for single AHB transfer of the specified transfer width. 1 = Data available is greater than or equal to half the FIFO depth for destination transfers, and space available is greater than FIFO depth or half FIFO depth for source transfers. The exceptions are at the end of a burst transaction request or at the end of block transfer. The space or data needs to be available in the FIFO before a burst transaction request is serviced is half FIFO depth on the following conditions: ● Transfer address is un-aligned ● Scatter or gather is enabled Enable this field will improve OCP/AXI master interface burst transaction length, but the burst transaction length also controlled by CTL.SRC_MSIZE and CTL.DST_MSIZE. If FIFO_MODE is enabled, the max. burst transaction length is the minimum value of FIFO depth (half FIFO depth) and MSIZE. If FIFO_MODE is disabled, the burst transaction length will be fixed to 1. The longer burst transaction will use less command to transmit the same amount of data, it will get more transfer efficiency. It is recommended to enable this field to improve transfer performance as long as the memory or peripheral accessed by DMAC support burst command.
32	FCMODE	R/W	0x0	Flow Control Mode. Determines whether source transaction requests are serviced when the Destination Peripheral is the flow controller. 0 = Source transaction requests are serviced when they occur. Data pre-fetching is enabled. 1 = Source transaction requests are not serviced until a destination transaction request occurs. Data pre-fetching is disabled. NOTE *Keep this bit 0 when the destination peripheral is not the flow controller.*
31	RELOAD_DST	N/A	0x0	Automatic Destination Reload. The DARx register can be automatically reloaded from its initial value at the end of every block for multi-block transfers. A new block transfer is then initiated. This register does not exist if the configuration parameter DMAC_CHx_MULTI_BLK_EN is not selected; in this case, the read-back value is always 0.
30	RELOAD_SRC	R/W	0x0	Automatic Source Reload. The SARx register can be automatically reloaded from its initial value at the end of every block for multi-block transfers. A new block transfer is then initiated. This field does not exist if the configuration parameter DMAC_CHx_MULTI_BLK_EN is not selected; in this case, the read-back value is always 0.
29:20	MAX_ABRST	R/W	0x0	Maximum AXI Burst Length. Maximum AXI burst length that is used for DMA transfers on this channel. A value of 0 indicates that software is not limiting the maximum AXI burst length for DMA transfers on this channel.

All information provided in this document is subject to legal disclaimers.
© REALTEK 2024. All rights reserved.

				This field does not exist if the DMA is OCP type, or if the configuration parameter DMAC_MABRST is not select, in this case, the read-back value is always 0. The legal value is 0 1 4 8 16 32 64 128 256.
19	SRC_HS_POL	R/W	0x0	Source Handshaking Interface Polarity. 0 = Active high 1 = Active low
18	DST_HS_POL	R/W	0x0	Destination Handshaking Interface Polarity. 0 = Active high 1 = Active low
17:12	RSVD	N/A	0x0	Reserved
11	HS_SEL_SRC	N/A	0x1	Source Software or Hardware Handshaking Select. This register selects which of the handshaking interfaces – hardware or software – is active for source requests on this channel. 0 = Hardware handshaking interface. Software-initiated transaction requests are ignored. 1 = Software handshaking interface. Hardware-initiated transaction requests are ignored. If the source peripheral is memory, then this bit is ignored.
10	HS_SEL_DST	R/W	0x1	Destination Software or Hardware Handshaking Select. This register selects which of the handshaking interfaces – hardware or software – is active for destination requests on this channel. 0 = Hardware handshaking interface. Software-initiated transaction requests are ignored. 1 = Software handshaking interface. Hardware-initiated transaction requests are ignored. If the destination peripheral is memory, then this bit is ignored.
9	FIFO_EMPTY	R/W	0x1	Indicates if there is data left in the channel FIFO. 1 = Channel FIFO empty 0 = Channel FIFO not empty
8	CH_SUSP	R/W	0x0	Channel Suspend. Suspends all DMA data transfers from the source until this bit is cleared. There is no guarantee that the current transaction will complete. Can also be used in conjunction with CFGx. INACTIVE to cleanly disable a channel without losing any data. 0 = Not suspended. 1 = Suspend DMA transfer from the source.
7:4	CH_PRIOR	R/W	0x0	Channel priority. DMAC supports three kinds of channel priority, fixed priority, round robin priority and programmable priority. The default way is fixed priority to reduce area, and channel0 has the highest priority, channel15 has the lowest priority. Changing the definition of FIX_PRIORITY or Programmable_PRIORITY can change priority way. If DMAC is configured to support programmable priority, this field will be used to set priority between channels. CH_PRIOR's legal value is 0~(DMAC_NUM_CHANNELS-1), and 0 is the highest priority value, (DMAC_NUM_CHANNELS-1) is the lowest priority value. If more than one channel are set with the same priority value, the channel with small channel number index has higher priority. If DMAC is configured to support fix priority or round-robin priority, this field is not exist.
3	RSVD	N/A	0x0	Reserved
2	DST_PCTL_OVER	R	0x1	DMA destination bus protocol finish, indicating the current write command has finished.
1	SRC_PCTL_OVER	R	0x1	DMA Source bus protocol finish, indicating the current read command has finished.
0	INACTIVE	R	0x0	Indicates if the channel is inactive. Can be used in conjunction with CFGx.CH_SUSP to cleanly disable a channel when destination is peripheral. 1 = Channel Inactive 0 = Channel not Inactive

13.4.2.3 Interrupt Registers

The following sections describe the registers pertaining to interrupts, their status, and how to clear them. For each channel, there are five types of interrupt sources:

All information provided in this document is subject to legal disclaimers.
© REALTEK 2024. All rights reserved.

- IntBlock – Block Transfer Complete Interrupt
  This interrupt is generated on DMA block transfer completion to the destination peripheral.
- IntDstTran – Destination Transaction Complete Interrupt
  This interrupt is generated after completion of the last OCP transfer of the requested single/burst transaction from the handshaking interface on the destination side.
- IntErr – Error Interrupt
  This interrupt is generated when an ERROR response is received from an OCP slave on the HRESP bus during a DMA transfer.
- IntSrcTran – Source Transaction Complete Interrupt
  This interrupt is generated after completion of the last OCP transfer of the requested single/burst transaction from the handshaking interface on the source side.
- IntTfr – DMA Transfer Complete Interrupt
  This interrupt is generated on DMA transfer completion to the destination peripheral.

There are several groups of interrupt-related registers:
- RawBlock, RawDstTran, RawErr, RawSrcTran, RawTfr
- StatusBlock, StatusDstTran, StatusErr, StatusSrcTran, StatusTfr
- MaskBlock, MaskDstTran, MaskErr, MaskSrcTran, MaskTfr
- ClearBlock, ClearDstTran, ClearErr, ClearSrcTran, ClearTfr
- StatusInt

When a channel has been enabled to generate interrupts, the following is true:
- Interrupt events are stored in the Raw Status registers.
- The contents of the Raw Status registers are masked with the contents of the Mask registers.
- The masked interrupts are stored in the Status registers.
- The contents of the Status registers are used to drive the int_* port signals.
- Writing to the appropriate bit in the Clear registers clears an interrupt in the Raw Status registers and the Status registers on the same clock cycle.

The contents of each of the five Status registers is ORed to produce a single bit for each interrupt type in the Combined Status register; that is, StatusInt.

NOTE

For interrupts to propagate past the raw interrupt register stage, CTLx.INT_EN must be set to 1'b1, and the relevant interrupt must be unmasked in the mask* interrupt register.*

13.4.2.3.1 RawBlock, RawDstTran, RawErr, RawSrcTran, RawTfr

- Name: Interrupt Raw Status Registers
- Size: 64 bits
- Address offset:
  - RawTfr – 0x2C0
  - RawBlock – 0x2C8
  - RawSrcTran – 0x2D0
  - RawDstTran – 0x2D8
  - RawErr – 0x2E0
- Read/write access: read/write

Interrupt events are stored in these Raw Interrupt Status registers before masking: RawBlock, RawDstTran, RawErr, RawSrcTran, and RawTfr. Each Raw Interrupt Status register has a bit allocated per channel; for example, RawTfr[2] is the Channel 2 raw transfer complete interrupt.

Each bit in these registers is cleared by writing a 1 to the corresponding location in the ClearTfr, ClearBlock, ClearSrcTran, ClearDstTran, ClearErr registers.

NOTE

Write access is available to these registers for software testing purposes only. Under normal operation, writes to these registers are not recommended.

Bit	Name	Access	Reset	Description
63:32	RSVD	N/A	0x0	Raw interrupt status. (Non-secure access secure register error)
31:0	RAW	R/W	0x0	Raw interrupt status. (Protocol error)

RawErr register contents of protocol error about OCP/AXI transaction, and RawErr register read value also contents of “Non-secure access secure registers Error”.

All information provided in this document is subject to legal disclaimers.
© REALTEK 2024. All rights reserved.

If a channel is set to secure channel (CFGx.PROTCTL is 0), the value read by non-secure CPU of this channel's bit in Raw* is always 0. Except RawErr[63:32]: these bits' values read by non-secure CPU are the actual values.

13.4.2.3.2 StatusBlock, StatusDstTran, StatusErr, StatusSrcTran, StatusTfr

- Name: Interrupt Status Registers
- Size: 64 bits
- Address offset:
  - StatusTfr – 0x2E8
  - StatusBlock – 0x2F0
  - StatusSrcTran – 0x2F8
  - StatusDstTran – 0x300
  - StatusErr – 0x308
- Read/write access: read

All interrupt events from all channels are stored in these Interrupt Status registers after masking: StatusBlock, StatusDstTran, StatusErr, StatusSrcTran, and StatusTfr. Each Interrupt Status register has a bit allocated per channel; for example, StatusTfr[2] is the Channel 2 status transfer complete interrupt. The contents of these registers are used to generate the interrupt signals (int or int_n bus, depending on interrupt polarity) leaving the DMAC.

If a channel is set to secure channel (CFGx.PROTCTL is 0), the value read by non-secure CPU of this channel's bit in Status* is always 0. Except StatusErr[63:32]: these bits' values read by non-secure CPU are the actual values.

Bit	Name	Access	Reset	Description
63:32	RSVD	N/A	0x0	Interrupt status. (Non-secure access secure register error)
31:0	STATUS	R	0x0	Interrupt status. (Protocol error)

13.4.2.3.3 MaskBlock, MaskDstTran, MaskErr, MaskSrcTran, MaskTfr

- Name: Interrupt Mask Registers
- Size: 64 bits
- Address offset:
  - MaskTfr – 0x310
  - MaskBlock – 0x318
  - MaskSrcTran – 0x320
  - MaskDstTran – 0x328
  - MaskErr – 0x330
- Read/write access: read/write

The contents of the Raw Status registers are masked with the contents of the Mask registers: MaskBlock, MaskDstTran, MaskErr, MaskSrcTran, and MaskTfr. Each Interrupt Mask register has a bit allocated per channel; for example, MaskTfr[2] is the mask bit for the Channel 2 transfer complete interrupt.

When the source peripheral of DMA channel i is memory, then the source transaction complete interrupt, MaskSrcTran[i], must be masked to prevent an erroneous triggering of an interrupt on the int_combined signal. Similarly, when the destination peripheral of DMA channel i is memory, then the destination transaction complete interrupt, MaskDstTran[i], must be masked to prevent an erroneous triggering of an interrupt on the int_combined(_n) signal.

A channel INT_MASK bit will be written only if the corresponding mask write enable bit in the INT_MASK_WE field is asserted on the same AHB write transfer. This allows software to set a mask bit without performing a read-modified write operation. For example, writing hex 01x1 to the MaskTfr register writes a 1 into MaskTfr[0], while MaskTfr[7:1] remains unchanged. Writing hex 00xx leaves MaskTfr[7:0] unchanged.

Writing a 1 to any bit in these registers unmasks the corresponding interrupt, thus allowing the DMAC to set the appropriate bit in the Status registers and int_* port signals.

MaskErr register decides protocol error about OCP/AXI transaction to be masked or unmasked, and MaskErr register also decides "Non-secure access secure registers Error" to be masked or unmasked.

If a channel is set to secure channel (CFGx.PROTCTL is 0), the value read by non-secure CPU of this channel's bit in Mask* is always 0, and the write operation from non-secure CPU is ignored. Except RawErr[63:32]: these bits' values read by non-secure CPU are the actual values and non-secure CPU can write 1 or 0 to these bits

All information provided in this document is subject to legal disclaimers.
© REALTEK 2024. All rights reserved.

Bit	Name	Access	Reset	Description
63:48	RSVD	N/A	0x0	Reserved
47:40	INT_MASK_WE	W	0x0	Interrupt Mask Write Enable (Non-secure access secure register error) 0 = write disabled 1 = write enabled
39:32	INT_MASK	R/W	0x0	Interrupt Mask (Non-secure access secure register error) 0 = masked 1 = unmasked
31:16	RSVD	N/A	0x0	Reserved
15:8	INT_MASK_WE	W	0x0	Interrupt Mask Write Enable (Protocol error) 0 = write disabled 1 = write enabled
7:0	INT_MASK	R/W	0x0	Interrupt Mask (Protocol error) 0 = masked 1 = unmasked

13.4.2.3.4 ClearBlock, ClearDstTran, ClearErr, ClearSrcTran, ClearTfr

- Name: Interrupt Clear Registers
- Size: 64 bits
- Address offset:
  - ClearTfr – 0x338
  - ClearBlock – 0x340
  - ClearSrcTran – 0x348
  - ClearDstTran – 0x350
  - ClearErr – 0x358
- Read/write access: write

Each bit in the Raw Status and Status registers is cleared on the same cycle by writing a 1 to the corresponding location in the Clear registers: ClearBlock, ClearDstTran, ClearErr, ClearSrcTran, and ClearTfr. Each Interrupt Clear register has a bit allocated per channel; for example, ClearTfr[2] is the clear bit for the Channel 2 transfer complete interrupt. Writing a 0 has no effect. These registers are not readable.

ClearErr register used to clear interrupt of protocol error about OCP/AXI transaction, and ClearErr register also used to clear “Non-secure access secure registers Error”.

If a channel is set to secure channel (CFGx.PROTCTL is 0), the write operation from non-secure CPU to this channel’s bit in Clear* is ignored. Except ClearErr[63:32]: Non-secure CPU can clear RawErr[63:32] bits by writing a 1 to these bits.

Bit	Name	Access	Reset	Description
63:40	Undefined	N/A	0x0	Reserved
39:32	CLEAR	W	N/A	Interrupt clear.(Non-secure access secure register error) 0 = no effect 1 = clear interrupt
31:8	Undefined	N/A	0x0	Reserved
7:0	CLEAR	W	N/A	Interrupt clear.(Protocol error) 0 = no effect 1 = clear interrupt

13.4.2.3.5 StatusInt

- Name: Combined Interrupt Status Register
- Size: 64 bits
- Address offset: 0x360
- Read/write access: read

The contents of each of the five Status registers – StatusTfr, StatusBlock, StatusSrcTran, StatusDstTran, StatusErr – is ORed to produce a single bit for each interrupt type in the Combined Status register (StatusInt). This register is read-only.

Bit	Name	Access	Reset	Description
63:5	RSVD	N/A	0x0	Reserved

All information provided in this document is subject to legal disclaimers.
© REALTEK 2024. All rights reserved.

4	ERR	R	0x0	OR of the contents of StatusErr register.
3	DSTT	R	0x0	OR of the contents of StatusDst register.
2	SRCT	R	0x0	OR of the contents of StatusSrcTran register.
1	BLOCK	R	0x0	OR of the contents of StatusBlock register.
0	TFR	R	0x0	OR of the contents of StatusTfr register.

It makes no distinction between secure interrupt and non-secure interrupt, the reading value of StatusInt is the same in secure mode or non-secure mode.

13.4.2.4 Miscellaneous DMAC Registers

This section describes the following registers of the DMAC:
- DmaIdReg – DMAC ID
- DmaOsNum – DMAC Outstanding Number
- DMA Component ID Register – DMAC Version ID

13.4.2.4.1 DmaIdReg

- Name: DMAC ID Register
- Size: 64 bits
- Address offset: 0x3A8
- Read/write access: read

This is a read-only register that reads back the configured hardcoded ID number, DMAC_ID_NUM.

Bit	Name	Access	Reset	Description
63:32	RSVD	N/A	0x0	Reserved
31:0	DMA_ID	R	DMA_ID_NUM	Hardcoded DMAC Peripheral ID

13.4.2.4.2 DmaOsNum

- Name: DMAC Outstanding Number Register
- Size: 64 bits
- Address offset: 0x3B8
- Read/write access: read/write

Bit	Name	Access	Reset	Description
63:16	RSVD	N/A	0x0	Reserved
15:8	OSW	R/W	OSW_CNT	If the OSW number of DMAC is programmable, this register is writable. And DMAC will not issue new write command until OSW of DMAC is less than this value. If the OSW number is not programmable, this value is read as “4”. And DMAC's OSW is fixed to 4. NOTE *If writing a '0' to this register, DMAC can not issue any new write commands.*
7:0	OSR	R/W	OSR_CNT	If the OSR number of DMAC is programmable, this register is writable. And DMAC will not issue new read command until OSR of DMAC is less than this value. If the OSR number is not programmable, this value is read as “channel number”. And DMAC does not limit the OSR number. NOTE *If writing a '0' to this register, DMAC can not issue any new read commands.*

13.4.2.4.3 DMA_COMP_PARAMS_6

- Name: DMAC Component Parameters Register 6
- Size: 64 bits

All information provided in this document is subject to legal disclaimers.
© REALTEK 2024. All rights reserved.

- Address offset: 0x3C8
- Read/write access: read

This is a constant read-only register that contains encoded information about the component parameter settings for Channel 7. The reset value depends on parameter(s).

Bit	Name	Access	Description
63	RSVD	N/A	Reserved
62:60	CH7_FIFO_DEPTH	R	The value is derived from the DMAC_CH7_FIFO_DEPTH parameter. 0x0 = 8 bytes 0x1 = 16 0x2 = 32 0x3 = 64 0x4 = 128
59:51	RSVD	R	Reserved
50:48	CH7_MAX_MULT_SIZE	R	The value is derived from the DMAC_CH7_MULT_SIZE parameter. Maximum burst transaction size. 0x0 = 4 0x1 = 8 0x2 = 16 0x3 = 32 0x4 = 64 0x5 = 128 0x6 = 256 0x7 = reserved
47:46	CH7_FC	R	The value is derived from the DMAC_CH7_FC parameter. 0x0 = DMA 0x1 = SRC 0x2 = DST 0x3 = ANY
45	CH7_HC_LLP	R	The value is derived from the DMAC_CH7_HC_LLP parameter. 0x0 = FALSE 0x1 = TRUE
44	CH7_CTL_WB_EN	R	The value is derived from the DMAC_CH7_CTL_WB_EN parameter. 0x0 = FALSE 0x1 = TRUE
43	CH7_MULTI_BLK_EN	R	The value is derived from the DMAC_CH7_MULT_BLK_EN parameter. 0x0 = FALSE 0x1 = TRUE
42	CH7_LOCK_EN	R	The value is derived from the DMAC_CH7_LOCK_EN parameter. 0x0 = FALSE 0x1 = TRUE
41	CH7_SRC_GAT_EN	R	The value is derived from the DMAC_CH7_SRC_GAT_EN parameter. 0x0 = FALSE 0x1 = TRUE
40	CH7_DST_SCA_EN	R	The value is derived from the DMAC_CH7_DST_SCA_EN parameter. 0x0 = FALSE 0x1 = TRUE
39	CH7_STAT_SRC	R	The value is derived from the DMAC_CH7_STAT_SRC parameter. 0x0 = FALSE 0x1 = TRUE
38	CH7_STAT_DST	R	The value is derived from the DMAC_CH7_STAT_DST parameter. 0x0 = FALSE 0x1 = TRUE
37:35	CH7_STW	R	The value is derived from the DMAC_CH7_STW parameter. 0x0 = NO_HARDCODE 0x1 = 8 0x2 = 16 0x3 = 32 0x4 = 64 0x5 = 128 0x6 = 256

All information provided in this document is subject to legal disclaimers.
© REALTEK 2024. All rights reserved.

			0x7 = reserved
34:32	CH7_DTW	R	The value is derived from the DMAC_CH7_DTW parameter. 0x0 = NO_HARDCODE 0x1 = 8 0x2 = 16 0x3 = 32 0x4 = 64 0x5 = 128 0x6 = 256 0x7 = reserved
31:0	RSVD	N/A	Reserved

13.4.2.4.4 DMA_COMP_PARAMS_5

- Name: DMAC Component Parameters Register 5
- Size: 64 bits
- Address offset: 0x3D0
- Read/write access: read

This is a constant read-only register that contains encoded information about the component parameter settings for Channel 5 and Channel 6. The reset value depends on parameter(s).

Bits 63:32 include the configuration parameters for Channel 5. Bits 31:0 include configuration parameters for Channel 6.

Bit	Name	Access	Description
63	RSVD	N/A	Reserved
62:60	CH5_FIFO_DEPTH	R	The value is derived from the DMAC_CH5_FIFO_DEPTH parameter. 0x0 = 8 0x1 = 16 0x2 = 32 0x3 = 64 0x4 = 128
59:51	RSVD	R	Reserved
50:48	CH5_MAX_MULT_SIZE	R	The value is derived from the DMAC_CH5_MULT_SIZE parameter. 0x0 = 4 0x1 = 8 0x2 = 16 0x3 = 32 0x4 = 64 0x5 = 128 0x6 = 256 0x7 = reserved
47:46	CH5_FC	R	The value is derived from the DMAC_CH5_FC parameter. 0x0 = DMA 0x1 = SRC 0x2 = DST 0x3 = ANY
45	CH5_HC_LLP	R	The value is derived from the DMAC_CH5_HC_LLP parameter. 0x0 = FALSE 0x1 = TRUE
44	CH5_CTL_WB_EN	R	The value is derived from the DMAC_CH5_CTL_WB_EN parameter. 0x0 = FALSE 0x1 = TRUE
43	CH5_MULTI_BLK_EN	R	The value is derived from the DMAC_CH5_MULT_BLK_EN parameter. 0x0 = FALSE 0x1 = TRUE
42	CH5_LOCK_EN	R	The value is derived from the DMAC_CH5_LOCK_EN parameter. 0x0 = FALSE 0x1 = TRUE
41	CH5_SRC_GAT_EN	R	The value is derived from the DMAC_CH5_SRC_GAT_EN parameter. 0x0 = FALSE

All information provided in this document is subject to legal disclaimers.
© REALTEK 2024. All rights reserved.

			0x1 = TRUE
40	CH5_DST_SCA_EN	R	The value is derived from the DMAC_CH5_DST_SCA_EN parameter. 0x0 = FALSE 0x1 = TRUE
39	CH5_STAT_SRC	R	The value is derived from the DMAC_CH5_STAT_SRC parameter. 0x0 = FALSE 0x1 = TRUE
38	CH5_STAT_DST	R	The value is derived from the DMAC_CH5_STAT_DST parameter. 0x0 = FALSE 0x1 = TRUE
37:35	CH5_STW	R	The value is derived from the DMAC_CH5_STW parameter. 0x0 = NO_HARDCODE 0x1 = 8 0x2 = 16 0x3 = 32 0x4 = 64 0x5 = 128 0x6 = 256 0x7 = reserved
34:32	CH5_DTW	R	The value is derived from the DMAC_CH5_DTW parameter. 0x0 = NO_HARDCODE 0x1 = 8 0x2 = 16 0x3 = 32 0x4 = 64 0x5 = 128 0x6 = 256 0x7 = reserved
31	RSVD	N/A	Reserved
30:28	CH5_FIFO_DEPTH	R	The value is derived from the DMAC_CH6_FIFO_DEPTH parameter. 0x0 = 8 0x1 = 16 0x2 = 32 0x3 = 64 0x4 = 128 0x5 = 256
27:19	RSVD	N/A	Reserved
18:16	CH5_MAX_MULT_SIZE	R	The value is derived from the DMAC_CH6_MULT_SIZE parameter. 0x0 = 4 0x1 = 8 0x2 = 16 0x3 = 32 0x4 = 64 0x5 = 128 0x6 = 256 0x7 = reserved
15:14	CH5_FC	R	The value is derived from the DMAC_CH6_FC parameter. 0x0 = DMA 0x1 = SRC 0x2 = DST 0x3 = ANY
13	CH5_HC_LLP	R	The value is derived from the DMAC_CH6_HC_LLP parameter. 0x0 = FALSE 0x1 = TRUE
12	CH5_CTL_WB_EN	R	The value is derived from the DMAC_CH6_CTL_WB_EN parameter. 0x0 = FALSE 0x1 = TRUE
11	CH5_MULTI_BLK_EN	R	The value is derived from the DMAC_CH6_MULT_BLK_EN parameter. 0x0 = FALSE 0x1 = TRUE
10	CH5_LOCK_EN	R	The value is derived from the DMAC_CH6_LOCK_EN parameter. 0x0 = FALSE

All information provided in this document is subject to legal disclaimers.
© REALTEK 2024. All rights reserved.

			0x1 = TRUE
9	CH5_SRC_GAT_EN	R	The value is derived from the DMAC_CH6_SRC_GAT_EN parameter. 0x0 = FALSE 0x1 = TRUE
8	CH5_DST_SCA_EN	R	The value is derived from the DMAC_CH6_DST_SCA_EN parameter. 0x0 = FALSE 0x1 = TRUE
7	CH5_STAT_SRC	R	The value is derived from the DMAC_CH6_STAT_SRC parameter. 0x0 = FALSE 0x1 = TRUE
6	CH5_STAT_DST	R	The value is derived from the DMAC_CH6_STAT_DST parameter. 0x0 = FALSE 0x1 = TRUE
5:3	CH5_STW	R	The value is derived from the DMAC_CH6_STW parameter. 0x0 = NO_HARDCODE 0x1 = 8 0x2 = 16 0x3 = 32 0x4 = 64 0x5 = 128 0x6 = 256 0x7 = reserved
2:0	CH5_DTW	R	The value is derived from the DMAC_CH6_DTW parameter. 0x0 = NO_HARDCODE 0x1 = 8 0x2 = 16 0x3 = 32 0x4 = 64 0x5 = 128 0x6 = 256 0x7 = reserved

13.4.2.4.5 DMA_COMP_PARAMS_4

- Name: DMAC Component Parameters Register 4
- Size: 64 bits
- Address offset: 0x3d8
- Read/write access: read

This is a constant read-only register that contains encoded information about the component parameter settings for Channel 3 and Channel 4. The reset value depends on parameter(s).

Bits 63:32 include the configuration parameters for Channel 3. Bits 31:0 include configuration parameters for Channel 4.

Bit	Name	Access	Description
63	RSVD	N/A	Reserved
62:60	CH3_FIFO_DEPTH	R	The value is derived from the DMAC_CH3_FIFO_DEPTH parameter. 0x0 = 8 0x1 = 16 0x2 = 32 0x3 = 64 0x4 = 128
59:51	RSVD	N/A	Reserved
50:48	CH3_MAX_MULT_SIZE	R	The value is derived from the DMAC_CH3_MULT_SIZE parameter. 0x0 = 4 0x1 = 8 0x2 = 16 0x3 = 32 0x4 = 64 0x5 = 128 0x6 = 256

All information provided in this document is subject to legal disclaimers.
© REALTEK 2024. All rights reserved.

			0x7 = reserved
47:46	CH3_FC	R	The value is derived from the DMAC_CH3_FC parameter. 0x0 = DMA 0x1 = SRC 0x2 = DST 0x3 = ANY
45	CH3_HC_LLP	R	The value is derived from the DMAC_CH3_HC_LLP parameter. 0x0 = FALSE 0x1 = TRUE
44	CH3_CTL_WB_EN	R	The value is derived from the DMAC_CH3_CTL_WB_EN parameter. 0x0 = FALSE 0x1 = TRUE
43	CH3_MULTI_BLK_EN	R	The value is derived from the DMAC_CH3_MULT_BLK_EN parameter. 0x0 = FALSE 0x1 = TRUE
42	CH3_LOCK_EN	R	The value is derived from the DMAC_CH3_LOCK_EN parameter. 0x0 = FALSE 0x1 = TRUE
41	CH3_SRC_GAT_EN	R	The value is derived from the DMAC_CH3_SRC_GAT_EN parameter. 0x0 = FALSE 0x1 = TRUE
40	CH3_DST_SCA_EN	R	The value is derived from the DMAC_CH3_DST_SCA_EN parameter. 0x0 = FALSE 0x1 = TRUE
39	CH3_STAT_SRC	R	The value is derived from the DMAC_CH3_STAT_SRC parameter. 0x0 = FALSE 0x1 = TRUE
38	CH3_STAT_DST	R	The value is derived from the DMAC_CH3_STAT_DST parameter. 0x0 = FALSE 0x1 = TRUE
37:35	CH3_STW	R	The value is derived from the DMAC_CH3_STW parameter. 0x0 = NO_HARDCODE 0x1 = 8 0x2 = 16 0x3 = 32 0x4 = 64 0x5 = 128 0x6 = 256 0x7 = reserved
34:32	CH3_DTW	R	The value is derived from the DMAC_CH3_DTW parameter. 0x0 = NO_HARDCODE 0x1 = 8 0x2 = 16 0x3 = 32 0x4 = 64 0x5 = 128 0x6 = 256 0x7 = reserved
31	RSVD	N/A	Reserved
30:28	CH4_FIFO_DEPTH	R	The value is derived from the DMAC_CH4_FIFO_DEPTH parameter. 0x0 = 8 0x1 = 16 0x2 = 32 0x3 = 64 0x4 = 128 0x5 = 256
27:19	RSVD	N/A	Reserved
18:16	CH4_MAX_MULT_SIZE	R	The value is derived from the DMAC_CH4_MULT_SIZE parameter. 0x0 = 4 0x1 = 8 0x2 = 16 0x3 = 32

All information provided in this document is subject to legal disclaimers.
© REALTEK 2024. All rights reserved.

			0x4 = 64 0x5 = 128 0x6 = 256 0x7 = reserved
15:14	CH4_FC	R	The value is derived from the DMAC_CH4_FC parameter. 0x0 = DMA 0x1 = SRC 0x2 = DST 0x3 = ANY
13	CH4_HC_LLP	R	The value is derived from the DMAC_CH4_HC_LLP parameter. 0x0 = FALSE 0x1 = TRUE
12	CH4_CTL_WB_EN	R	The value is derived from the DMAC_CH4_CTL_WB_EN parameter. 0x0 = FALSE 0x1 = TRUE
11	CH4_MULTI_BLK_EN	R	The value is derived from the DMAC_CH4_MULT_BLK_EN parameter. 0x0 = FALSE 0x1 = TRUE
10	CH4_LOCK_EN	R	The value is derived from the DMAC_CH4_LOCK_EN parameter. 0x0 = FALSE 0x1 = TRUE
9	CH4_SRC_GAT_EN	R	The value is derived from the DMAC_CH4_SRC_GAT_EN parameter. 0x0 = FALSE 0x1 = TRUE
8	CH4_DST_SCA_EN	R	The value is derived from the DMAC_CH4_DST_SCA_EN parameter. 0x0 = FALSE 0x1 = TRUE
7	CH4_STAT_SRC	R	The value is derived from the DMAC_CH4_STAT_SRC parameter. 0x0 = FALSE 0x1 = TRUE
6	CH4_STAT_DST	R	The value is derived from the DMAC_CH4_STAT_DST parameter. 0x0 = FALSE 0x1 = TRUE
5:3	CH4_STW	R	The value is derived from the DMAC_CH4_STW parameter. 0x0 = NO_HARDCODE 0x1 = 8 0x2 = 16 0x3 = 32 0x4 = 64 0x5 = 128 0x6 = 256 0x7 = reserved
2:0	CH4_DTW	R	The value is derived from the DMAC_CH4_DTW parameter. 0x0 = NO_HARDCODE 0x1 = 8 0x2 = 16 0x3 = 32 0x4 = 64 0x5 = 128 0x6 = 256 0x7 = reserved

13.4.2.4.6 DMA_COMP_PARAMS_3

- Name: DMAC Component Parameters Register 3
- Size: 64 bits
- Address offset: 0x3E0
- Read/write access: read

This is a constant read-only register that contains encoded information about the component parameter settings for Channel 1 and Channel 2. The reset value depends on parameter(s).

All information provided in this document is subject to legal disclaimers.
© REALTEK 2024. All rights reserved.

Bits 63:32 include the configuration parameters for Channel 1. Bits 31:0 include configuration parameters for Channel 2.

Bit	Name	Access	Description
63	RSVD	N/A	Reserved
62:60	CH1_FIFO_DEPTH	R	The value is derived from the DMAC_CH1_FIFO_DEPTH parameter. 0x0 = 8 0x1 = 16 0x2 = 32 0x3 = 64 0x4 = 128 0x5 = 256
59:51	RSVD	N/A	Reserved
50:48	CH1_MAX_MULT_SIZE	R	The value is derived from the DMAC_CH1_MULT_SIZE parameter. 0x0 = 4 0x1 = 8 0x2 = 16 0x3 = 32 0x4 = 64 0x5 = 128 0x6 = 256 0x7 = reserved
47:46	CH1_FC	R	The value is derived from the DMAC_CH1_FC parameter. 0x0 = DMA 0x1 = SRC 0x2 = DST 0x3 = ANY
45	CH1_HC_LLP	R	The value is derived from the DMAC_CH1_HC_LLP parameter. 0x0 = FALSE 0x1 = TRUE
44	CH1_CTL_WB_EN	R	The value is derived from the DMAC_CH1_CTL_WB_EN parameter. 0x0 = FALSE 0x1 = TRUE
43	CH1_MULTI_BLK_EN	R	The value is derived from the DMAC_CH1_MULT_BLK_EN parameter. 0x0 = FALSE 0x1 = TRUE
42	CH1_LOCK_EN	R	The value is derived from the DMAC_CH1_LOCK_EN parameter. 0x0 = FALSE 0x1 = TRUE
41	CH1_SRC_GAT_EN	R	The value is derived from the DMAC_CH1_SRC_GAT_EN parameter. 0x0 = FALSE 0x1 = TRUE
40	CH1_DST_SCA_EN	R	The value is derived from the DMAC_CH1_DST_SCA_EN parameter. 0x0 = FALSE 0x1 = TRUE
39	CH1_STAT_SRC	R	The value is derived from the DMAC_CH1_STAT_SRC parameter. 0x0 = FALSE 0x1 = TRUE
38	CH1_STAT_DST	R	The value is derived from the DMAC_CH1_STAT_DST parameter. 0x0 = FALSE 0x1 = TRUE
37:35	CH1_STW	R	The value is derived from the DMAC_CH1_STW parameter. 0x0 = NO_HARDCODE 0x1 = 8 0x2 = 16 0x3 = 32 0x4 = 64 0x5 = 128 0x6 = 256 0x7 = reserved
34:32	CH1_DTW	R	The value is derived from the DMAC_CH1_DTW parameter. 0x0 = NO_HARDCODE 0x1 = 8

All information provided in this document is subject to legal disclaimers.
© REALTEK 2024. All rights reserved.

			0x2 = 16 0x3 = 32 0x4 = 64 0x5 = 128 0x6 = 256 0x7 = reserved
31	RSVD	N/A	Reserved
30:28	CH2_FIFO_DEPTH	R	The value is derived from the DMAC_CH2_FIFO_DEPTH parameter. 0x0 = 8 0x1 = 16 0x2 = 32 0x3 = 64 0x4 = 128
27:19	RSVD	N/A	Reserved
18:16	CH2_MAX_MULT_SIZE	R	The value is derived from the DMAC_CH2_MULT_SIZE parameter. 0x0 = 4 0x1 = 8 0x2 = 16 0x3 = 32 0x4 = 64 0x5 = 128 0x6 = 256 0x7 = reserved
15:14	CH2_FC	R	The value is derived from the DMAC_CH2_FC parameter. 0x0 = DMA 0x1 = SRC 0x2 = DST 0x3 = ANY
13	CH2_HC_LLP	R	The value is derived from the DMAC_CH2_HC_LLP parameter. 0x0 = FALSE 0x1 = TRUE
12	CH2_CTL_WB_EN	R	The value is derived from the DMAC_CH2_CTL_WB_EN parameter. 0x0 = FALSE 0x1 = TRUE
11	CH2_MULTI_BLK_EN	R	The value is derived from the DMAC_CH2_MULT_BLK_EN parameter. 0x0 = FALSE 0x1 = TRUE
10	CH2_LOCK_EN	R	The value is derived from the DMAC_CH2_LOCK_EN parameter. 0x0 = FALSE 0x1 = TRUE
9	CH2_SRC_GAT_EN	R	The value is derived from the DMAC_CH2_SRC_GAT_EN parameter. 0x0 = FALSE 0x1 = TRUE
8	CH2_DST_SCA_EN	R	The value is derived from the DMAC_CH2_DST_SCA_EN parameter. 0x0 = FALSE 0x1 = TRUE
7	CH2_STAT_SRC	R	The value is derived from the DMAC_CH2_STAT_SRC parameter. 0x0 = FALSE 0x1 = TRUE
6	CH2_STAT_DST	R	The value is derived from the DMAC_CH2_STAT_DST parameter. 0x0 = FALSE 0x1 = TRUE
5:3	CH2_STW	R	The value is derived from the DMAC_CH2_STW parameter. 0x0 = NO_HARDCODE 0x1 = 8 0x2 = 16 0x3 = 32 0x4 = 64 0x5 = 128 0x6 = 256 0x7 = reserved
2:0	CH2_DTW	R	The value is derived from the DMAC_CH2_DTW parameter.

All information provided in this document is subject to legal disclaimers.
© REALTEK 2024. All rights reserved.

			0x0 = NO_HARDCODE 0x1 = 8 0x2 = 16 0x3 = 32 0x4 = 64 0x5 = 128 0x6 = 256 0x7 = reserved

13.4.2.4.7 DMA_COMP_PARAMS_2

- Name: DMAC Component Parameters Register 2
- Size: 64 bits
- Address offset: 0x3E8
- Read/write access: read

This is a constant read-only register that contains encoded information about the component parameter settings. The reset value depends on parameter(s).

Bits 63:32 include the Multi-Block type parameter for Channel 0 to Channel 7. Bits 31:0 include the configuration parameters for Channel 0.

Bit	Name	Access	Description
63:60	CH7_MULTI_BLK_TYPE	R	The values of these fields are derived from the DMAC_CHx_MULTI_BLK_TYPE parameter. 0x0 = NO_HARDCODE 0x1 = CONT_RELOAD 0x2 = RELOAD_CONT 0x3 = RELOAD_RELOAD 0x4 = CONT_LLP 0x5 = RELOAD_LLP 0x6 = LLP_CONT 0x7 = LLP_RELOAD 0x8 = LLP_LLP
59:56	CH7_MULTI_BLK_TYPE	R	
55:52	CH7_MULTI_BLK_TYPE	R	
51:48	CH7_MULTI_BLK_TYPE	R	
47:44	CH7_MULTI_BLK_TYPE	R	
43:40	CH7_MULTI_BLK_TYPE	R	
39:36	CH7_MULTI_BLK_TYPE	R	
35:32	CH7_MULTI_BLK_TYPE	R	
31	RSVD	N/A	Reserved
30:28	CH0_FIFO_DEPTH	R	The value is derived from the DMAC_CH0_FIFO_DEPTH parameter. 0x0 = 8 0x1 = 16 0x2 = 32 0x3 = 64 0x4 = 128 0x5 = 256
27:19	RSVD	N/A	Reserved
18:16	CH0_MAX_MULT_SIZE	R	The value is derived from the DMAC_CH0_MULT_SIZE parameter. 0x0 = 4 0x1 = 8 0x2 = 16 0x3 = 32 0x4 = 64 0x5 = 128 0x6 = 256 0x7 = reserved
15:14	CH0_FC	R	The value is derived from the DMAC_CH0_FC parameter. 0x0 = DMA 0x1 = SRC 0x2 = DST 0x3 = ANY
13	CH0_HC_LLP	R	The value is derived from the DMAC_CH0_HC_LLP parameter. 0x0 = FALSE 0x1 = TRUE
12	CH0_CTL_WB_EN	R	The value is derived from the DMAC_CH0_CTL_WB_EN parameter. 0x0 = FALSE

All information provided in this document is subject to legal disclaimers.
© REALTEK 2024. All rights reserved.

			0x1 = TRUE
11	CH0_MULTI_BLK_EN	R	The value is derived from the DMAC_CH0_MULT_BLK_EN parameter. 0x0 = FALSE 0x1 = TRUE
10	CH0_LOCK_EN	R	The value is derived from the DMAC_CH0_LOCK_EN parameter. 0x0 = FALSE 0x1 = TRUE
9	CH0_SRC_GAT_EN	R	The value is derived from the DMAC_CH0_SRC_GAT_EN parameter. 0x0 = FALSE 0x1 = TRUE
8	CH0_DST_SCA_EN	R	The value is derived from the DMAC_CH0_DST_SCA_EN parameter. 0x0 = FALSE 0x1 = TRUE
7	CH0_STAT_SRC	R	The value is derived from the DMAC_CH0_STAT_SRC parameter. 0x0 = FALSE 0x1 = TRUE
6	CH0_STAT_DST	R	The value is derived from the DMAC_CH0_STAT_DST parameter. 0x0 = FALSE 0x1 = TRUE
5:3	CH0_STW	R	The value is derived from the DMAC_CH0_STW parameter. 0x0 = NO_HARDCODE 0x1 = 8 0x2 = 16 0x3 = 32 0x4 = 64 0x5 = 128 0x6 = 256 0x7 = reserved
2:0	CH0_DTW	R	The value is derived from the DMAC_CH0_DTW parameter. 0x0 = NO_HARDCODE 0x1 = 8 0x2 = 16 0x3 = 32 0x4 = 64 0x5 = 128 0x6 = 256 0x7 = reserved

13.4.2.4.8 DMA_COMP_PARAMS_1

- Name: DMAC Component Parameters Register 1
- Size: 64 bits
- Address offset: 0x3F0
- Read/write access: read

This is a constant read-only register that contains encoded information about the component parameter settings. The reset value depends on parameter(s).

Bits 63:32 include the top-level parameter for DMAC. Bits 31:0 include the maximum block size parameters for Channel 0 to Channel 7.

Bit	Name	Access	Description
63	HIGH_BIT_NUM_HS_INT2	R	Number of hardware handshake interfaces. If it is greater than 31 and less than 64, this bit read value is 1.
62	HIGH_BIT_NUM_HS_INT1	R	Number of hardware handshake interfaces. If it is 64, this bit read value is 1.
61	STATIC_ENDIAN_SELECT	R	The value is derived from the DMAC_STATIC_ENDIAN_SELECT parameter. 0 = FALSE (dynamic endian fixed) 0 = TRUE (not supported)
60	ADD_ENCODED_PARAMS	R	The value is derived from the DMAC_ADD_ENCODED_PARAMS parameter. 0 = FALSE 0 = TRUE
59:55	NUM_HS_INT	R	The value is derived from the DMAC_NUM_HS_INT parameter.

All information provided in this document is subject to legal disclaimers.
© REALTEK 2024. All rights reserved.

			0x00 = 0 … 0x10 = 16
54:51	RSVD	N/A	Reserved
50:49	M2_HDATA_WIDTH	R	OCP Master 2 data width. The value is derived from the DMAC_M2_HDATA_WIDTH parameter. Master 2 data width should be equal to that of Master 1. 0x0 = 32 bits 0x1 = 64 bits 0x2 = 128 bits 0x3 = 256 bits
48:47	M1_HDATA_WIDTH	R	OCP Master 1 data width. The value is derived from the DMAC_M1_HDATA_WIDTH parameter. Master 1 data width should be equal to that of Master 2. 0x0 = 32 bits 0x1 = 64 bits 0x2 = 128 bits 0x3 = 256 bits
46:45	S_HDATA_WIDTH	R	OCP PP slave interface data width. The value is derived from the DMAC_S_HDATA_WIDTH parameter. The width is fixed to 32 bits. 0x0 = 32 bits
44:43	NUM_MASTER_INT	R	Number of master interfaces. The value is derived from the DMAC_NUM_MASTER_INT parameter. This value is fixed to 1 (only 2 OCP masters supported) 0x1 = 2
42:39	NUM_CHANNELS	R	Number of channels. The value is derived from the DMAC_NUM_CHANNELS parameter. 0x00 = 1 to 0xf = 16
38:36	RSVD	N/A	Reserved
35	MABRST	R	Maximum burst length fixed or not. The value is derived from the DMAC_MABRST parameter. Actually, for Realtek OCP profile, the max. burst length is related to data width. 0 = FALSE 1 = TRUE
34:33	INTR_IO	R	Interrupt output format. The value is derived from the DMAC_INTR_IO parameter. 0x0 = ALL 0x1 = TYPE 0x2 = COMBINED 0x3 = reserved
32	BIG_ENDIAN	R	The value is derived from the DMAC_BIG_ENDIAN parameter. 0 = FALSE 1 = TRUE
31:28	CH7_MAX_BLK_SIZE	R	The values of these fields are derived from DMAC_CHx_MAX_BLK_SIZE parameter. 0x0 = 3 0x1 = 7 0x2 =15 0x3 = 31 0x4 = 63 0x5 = 127 0x6 = 255 0x7 = 511 0x8 = 1023 0x9 = 2047 0xa = 4095
27:24	CH6_MAX_BLK_SIZE	R	
23:20	CH5_MAX_BLK_SIZE	R	
19:16	CH4_MAX_BLK_SIZE	R	
15:12	CH3_MAX_BLK_SIZE	R	
11:8	CH2_MAX_BLK_SIZE	R	
7:4	CH1_MAX_BLK_SIZE	R	
3:0	CH0_MAX_BLK_SIZE	R	

13.4.2.4.9 DMA Component ID Register

- Name: DMA Component ID Register

All information provided in this document is subject to legal disclaimers.
© REALTEK 2024. All rights reserved.

- Size: 64 bits
- Address Offset: 0x3F8
- Read/Write Access: Read

This is the DMAC Component Version register, which is a read-only register that specifies the version of the packaged component in the upper 32 bits and the component type in the lower 32 bits.

Bit	Name	Access	Reset	Description
63:32	DMA_COMP_VERSION	R	Version/YEAR/MON/DAY	Version of the component. Example: 0x10141108 For version 1.0, 2014/11/08
31:0	DMA_ID	R	DMAC_ID_NUM	DMAC ID. Configure it in config form

13.5 Programming the DMAC

The DMAC can be programmed through software registers or the DMAC low-level software driver.

13.5.1 DMA Transfer Types

A DMA transfer may consist of single or multi-block transfers. On successive blocks of a multi-block transfer, the SARx/DARx register in the DMAC is reprogrammed using either of the following methods:

- Block chaining using linked lists
- Auto-reloading
- Contiguous address between blocks

On successive blocks of a multi-block transfer, the CTLx register in the DMAC is reprogrammed using either of the following methods:

- Block chaining using linked lists
- Auto-reloading

When block chaining, using Linked Lists is the multi-block method of choice. On successive blocks, the LLPx register in the DMAC is reprogrammed using block chaining with linked lists.

A block descriptor consists of six registers: SARx, DARx, LLPx, CTLx, SSTATx, and DSTATx. The first four registers, along with the CFGx register, are used by the DMAC to set up and describe the block transfer.

13.5.1.1 Multi-Block Transfers

NOTE

Multi-block transfers—in which the source and destination are swapped during the transfer—are not supported. In a multi-block transfer, the direction must not change for the duration of the transfer.

13.5.1.2 Block Chaining Using Linked List

In this case, the DMAC reprograms the channel registers prior to the start of each block by fetching the block descriptor for that block from system memory. This is known as an LLI update.

DMAC block chaining uses a Linked List Pointer register (LLPx) that stores the address in memory of the next linked list item. Each LLI contains the corresponding block descriptors:

- SARx
- DARx
- LLPx
- CTLx
- SSTATx
- DSTATx

To set up block chaining, you program a sequence of Linked Lists in memory.

LLI accesses are always 32-bit accesses (Hsize = 2) aligned to 32-bit boundaries and cannot be changed or programmed to anything other than 32-bit, even if the OCP master interface of the LLI supports more than a 32-bit data width.

All information provided in this document is subject to legal disclaimers.
© REALTEK 2024. All rights reserved.

The SARx, DARx, LLPx, and CTLx registers are fetched from system memory on an LLI update. If configuration parameter DMAC_CHx_CTL_WB_EN = True, then the updated contents of the CTLx, SSTATx, and DSTATx registers are written back to memory on block completion. *Figure 13-2* and *Figure 13-3* show how you use chained linked lists in memory to define multi-block transfers using block chaining.

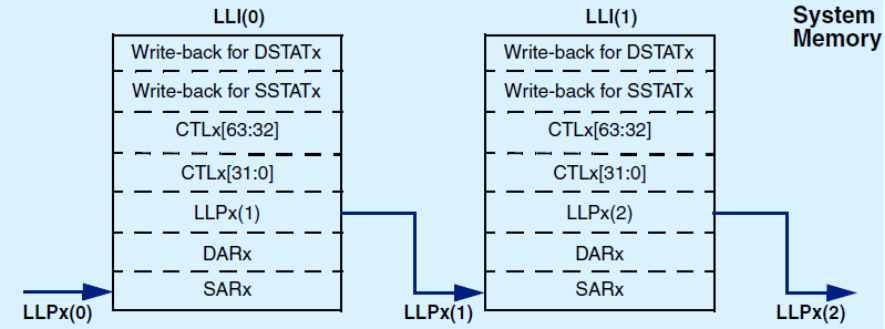


Figure 13-2 Multi-block transfer using linked lists when DMAC_CHx_STAT_SRC set to true

It is assumed that no allocation is made in system memory for the source status when the configuration parameter DMAC_CHx_STAT_SRC is set to False. If this parameter is False, then the order of a Linked List item is as follows:

- SARx
- DARx
- LLPx
- CTLx
- DSTATx

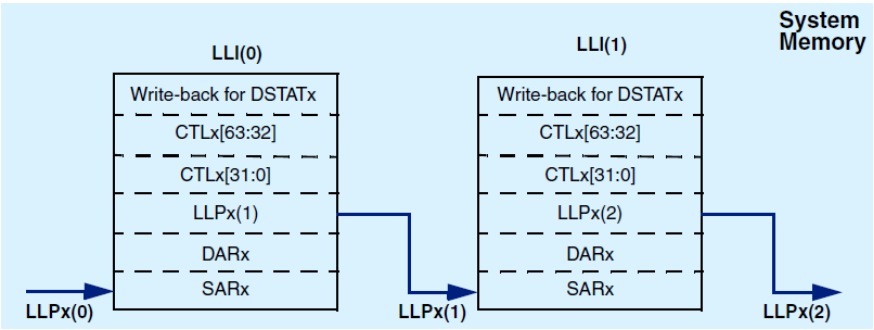


Figure 13-3 Multi-block transfer using linked lists when DMAC_CHx_STAT_SRC set to false

NOTE

In order to not confuse the SARx, DARx, LLPx, CTLx, STATx, and DSTATx register locations of the LLI with the corresponding DMAC memory mapped register locations, the LLI register locations are prefixed with LLI; that is, LLI.SARx, LLI.DARx, LLI.LLPx, LLI.CTLx, LLI.SSTATx, and LLI.DSTATx.

Figure 13-4 and *Figure 13-5* show the mapping of a Linked List Item stored in memory to the channel registers block descriptor.

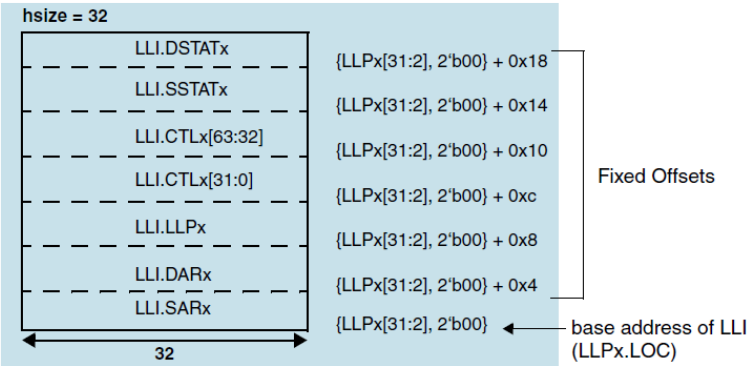


Figure 13-4 Mapping of block descriptor (LLI) in memory to channel registers when DMAC_CHx_STAT_SRC set to true

All information provided in this document is subject to legal disclaimers.
© REALTEK 2024. All rights reserved.

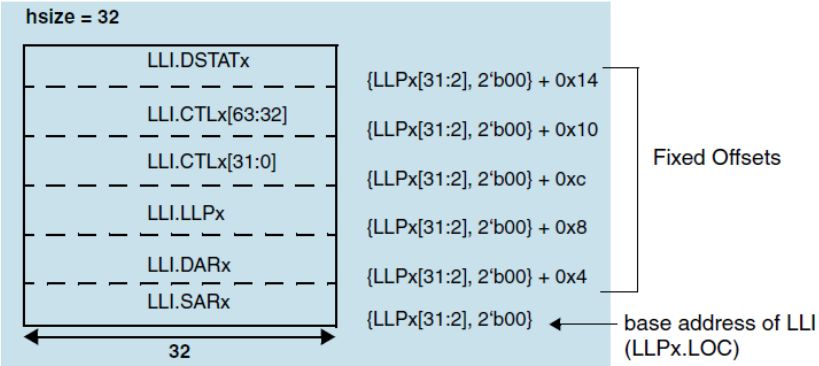


Figure 13-5 Mapping of block descriptor (LLI) in memory to channel registers when DMAC_CHx_STAT_SRC set to false

Rows 6 through 10 of *Table 13-7* show the required values of LLPx, CTLx, and CFGx for multi-block DMA transfers using block chaining.

NOTE

For rows 6 through 10 of Table 13-7, the LLI.CTLx, LLI.LLPx, LLI.SARx, and LLI.DARx register locations of the LLI are always affected at the start of every block transfer. The LLI.LLPx and LLI.CTLx locations are always used to reprogram the DMAC LLPx and CTLx registers. However, depending on the Table 13-7 row number, the LLI.SARx/LLI.DARx address may or may not be used to reprogram the DMAC SARx/DARx registers.

Table 13-7 Programming of transfer types and channel register update method

No.	Transfer type	LLP.LOC=0	LLP_SRC_EN (CTLx)	RELOAD_SRC (CRGx)	LLP_DST_EN (CTLx)	RELOAD_DST (CRGx)	CTLx, LLPx update method	SARx update method	DARx update method	Write back[1]
1	Single-block or last transfer of multi-block	Yes	0	0	0	0	None, user reprograms	None (Single)	None (Single)	No
2	Auto-reload multi-block transfer with contiguous SAR	Yes	0	0	0	1	CTLx, LLPx are reloaded from initial values	Contiguous	Auto-Reload	No
3	Auto-reload multi-block transfer with contiguous DAR	Yes	0	1	0	0	CTLx, LLPx are reloaded from initial values	Auto-Reload	Contiguous	No
4	Auto-reload multi-block transfer	Yes	0	1	0	1	CTLx, LLPx are reloaded from initial values	Auto-Reload	Auto-Reload	No
5	Single-block or last transfer of multi-block	No	0	0	0	0	None, user reprograms	None (Single)	None (Single)	Yes
6	Linked list multi-block transfer with contiguous SAR	No	0	0	1	0	CTLx, LLPx loaded from next Linked List item	Contiguous	Linked List	Yes
7	Linked list multi-block transfer with auto-reload SAR	No	0	1	1	0	CTLx, LLPx loaded from next Linked List item	Auto-Reload	Linked List	Yes
8	Linked list multi-block transfer with contiguous DAR	No	1	0	0	0	CTLx, LLPx loaded from next Linked List item	Linked List	Contiguous	Yes
9	Linked list multi-block transfer with auto-reload DAR	No	1	0	0	1	CTLx, LLPx loaded from next Linked List item	Linked List	Auto-Reload	Yes

[1] This column assumes that the configuration parameter DMAC_CHx_CTL_WB_EN = True. If DMAC_CHx_CTL_WB_EN = False, then there is never writeback of the control and status registers regardless of transfer type, and all rows of this column are "No".

All information provided in this document is subject to legal disclaimers.
© REALTEK 2024. All rights reserved.

10	Linked list multi-block transfer	No	1	0	1	0	CTLx, LLPx loaded from next Linked List item	Linked List	Linked List	Yes

NOTE

- *Throughout this table, there are descriptions about fetching the LLI.CTLx register from the location pointed to by the LLPx register. This exact location is the LLI base address (stored in LLPx register) plus the fixed offset. For example, in Figure 13-4, the location of the LLI.CTLx register is LLPx.LOC + 0xc.*
- *Referring to Table 13-7, if the Write Back column entry is "Yes" and the configuration parameter DMAC_CHx_CTL_WB_EN = True, then the CTLx[63:32] register is always written to system memory (to LLI.CTLx[63:32]) at the end of every block transfer. The source status is fetched and written to system memory at the end of every block transfer if the Write Back column entry is "Yes," DMAC_CHx_CTL_WB_EN = True, DMAC_CHx_STAT_SRC = True, and CFGx.SS_UPD_EN is enabled. The destination status is fetched and written to system memory at the end of every block transfer if the Write Back column entry is "Yes," DMAC_CHx_CTL_WB_EN = True, DMAC_CHx_STAT_DST = True, and CFGx.DS_UPD_EN is enabled.*

13.5.1.3 Auto-Reloading of Channel Registers

During auto-reloading, the channel registers are reloaded with their initial values at the completion of each block and the new values used for the new block. Depending on the row number in *Table 13-7*, some or all of the SARx, DARx, and CTLx channel registers are reloaded from their initial value at the start of a block transfer.

13.5.1.4 Contiguous Address between Blocks

In this case, the address between successive blocks is selected as a continuation from the end of the previous block.
Enabling the source or destination address to be contiguous between blocks is a function of the CTLx.LLP_SRC_EN, CFGx.RELOAD_SRC, CTLx.LLP_DST_EN, and CTLx.RELOAD_DST registers (see *Table 13-7*).

NOTE

You cannot select both SARx and DARx updates to be contiguous. If you want this functionality, you should increase the size of the Block Transfer (CTLx.BLOCK_TS), or if this is at the maximum value, use Row 10 of Table 13-7 and set up the LLI.SARx address of the block descriptor to be equal to the end SARx address of the previous block. Similarly, set up the LLI.DARx address of the block descriptor to be equal to the end DARx address of the previous block. For more information, refer to "Multi-Block Transfer with Linked List for Source and Linked List for Destination (Row 10)"

13.5.1.5 Suspension of Transfers between Blocks

At the end of every block transfer, an end-of-block interrupt is asserted if:
(1) Interrupts are enabled, CTLx.INT_EN = 1, and
(2) The channel block interrupt is unmasked, MaskBlock[n] = 1, where n is the channel number.

NOTE

The block-complete interrupt is generated at the completion of the block transfer to the destination.

For rows 6, 8, and 10 of *Table 13-7*, the DMA transfer does not stall between block transfers. For example, at the end-of-block N, the DMAC automatically proceeds to block (N + 1).

For rows 2, 3, 4, 7, and 9 of *Table 13-7* (SARx and/or DARx auto-reloaded between block transfers), the DMA transfer automatically stalls after the end-of-block interrupt is asserted, if the end-of-block interrupt is enabled and unmasked.

The DMAC does not proceed to the next block transfer until a write to the ClearBlock[n] block interrupt clear register, done by software to clear the channel block-complete interrupt, is detected by hardware.

For rows 2, 3, 4, 7, and 9 of *Table 13-7* (SARx and/or DARx auto-reloaded between block transfers), the DMA transfer does not stall if either:
- Interrupts are disabled, CTLx.INT_EN = 0, or
- The channel block interrupt is masked, MaskBlock[n] = 0, where n is the channel number.

Channel suspension between blocks is used to ensure that the end-of-block ISR (interrupt service routine) of the next-to-last block is serviced before the start of the final block commences. This ensures that the ISR has cleared the CFGx.RELOAD_SRC and/or CFGx.RELOAD_DST bits before completion of the final block. The reload bits CFGx.RELOAD_SRC and/or CFGx.RELOAD_DST should be cleared in the end-of-block ISR for the next-to-last block transfer.

All information provided in this document is subject to legal disclaimers.
© REALTEK 2024. All rights reserved.

13.5.1.6 Ending Multi-Block Transfers

All multi-block transfers must end as shown in either Row 1 or Row 5 of *Table 13-7*. At the end of every block transfer, the DMAC samples the row number, and if the DMAC is in the Row 1 or Row 5 state, then the previous block transferred was the last block and the DMA transfer is terminated.

NOTE

Row 1 and Row 5 are used for single-block transfers or terminating multi-block transfers. Transfers initiated in rows 2, 3 or 4 can only end in row 1; similarly, transfers initiated in rows 6 through 10 can only end in row 5. Ending in the Row 5 state enables status fetch and write-back for the last block. Ending in the Row 1 state disables status fetch and write-back for the last block.

For rows 2, 3, and 4 of *Table 13-7*, (LLPx.LOC = 0 and CFGx.RELOAD_SRC and/or CFGx.RELOAD_DST is set), multi-block DMA transfers continue until both the CFGx.RELOAD_SRC and CFGx.RELOAD_DST registers are cleared by software. They should be programmed to 0 in the end-of-block interrupt service routine that services the next-to-last block transfer; this puts the DMAC into the Row 1 state.

For rows 6, 8, and 10 of *Table 13-7* (both CFGx.RELOAD_SRC and CFGx.RELOAD_DST cleared), the user must set up the last block descriptor in memory so that both LLI.CTLx.LLP_SRC_EN and LLI.CTLx.LLP_DST_EN are 0.

The sampling of the LLPx.LOC bit takes place exclusively at the beginning of the transfer when the channel is enabled. This determines whether writeback is enabled throughout the complete transfer, and changing the value of this bit in subsequent blocks on the same transfer does not have any effect.

NOTE

The only allowed transitions between the rows of Table 13-7 are from any row into Row 1 or Row 5. As already stated, a transition into row 1 or row 5 is used to terminate the DMA transfer; all other transitions between rows are not allowed. Software must ensure that illegal transitions between rows do not occur between blocks of a multi-block transfer. For example, if block N is in row 10, then the only allowed rows for block (N +1) are rows 10 or 5.

13.5.2 Disabling a Channel Prior to Transfer Completion

Under normal operation, software enables a channel by writing a 1 to the channel enable register, ChEnReg.CH_EN, and hardware disables a channel on transfer completion by clearing the ChEnReg.CH_EN register bit.

The recommended way for software to disable a channel without losing data is to use the CH_SUSP bit in conjunction with the FIFO_EMPTY bit in the Channel Configuration Register (CFGx).

(1) If software wishes to disable a channel prior to the DMA transfer completion, then it can set the CFGx.CH_SUSP bit to tell the DMAC to halt all transfers from the source peripheral. Therefore, the channel FIFO receives no new data.
(2) Software can now poll the CFGx.FIFO_EMPTY bit until it indicates that the channel FIFO is empty.
(3) The ChEnReg.CH_EN bit can then be cleared by software once the channel FIFO is empty.

When CTLx.SRC_TR_WIDTH < CTLx.DST_TR_WIDTH and the CFGx.CH_SUSP bit is high, the CFGx.FIFO_EMPTY is asserted once the contents of the FIFO do not permit a single word of CTLx.DST_TR_WIDTH to be formed. However, there may still be data in the channel FIFO, but not enough to form a single transfer of CTLx.DST_TR_WIDTH. In this scenario, once the channel is disabled, the remaining data in the channel FIFO is not transferred to the destination peripheral.

It is permissible to remove the channel from the suspension state by writing a 0 to the CFGx.CH_SUSP register. The DMA transfer completes in the normal manner.

NOTE

- *If a channel is disabled by software, an active single or burst transaction is not guaranteed to receive an acknowledgement.*
- *If the DMAC is configured to use defined length bursts (DMAC_INCR_BURSTS = 0), disabling the channel via software prior to completing a transfer is not supported.*

13.5.2.1 Abnormal Transfer Termination

A DMAC transfer may be terminated abruptly by software by clearing the channel enable bit, ChEnReg.CH_EN. You must not assume that the channel is disabled immediately after the ChEnReg. The CH_EN bit is cleared over the AHB slave interface. Consider this as a request to disable the channel. You must poll ChEnReg.CH_EN and confirm that the channel is disabled by reading back 0. A case where the channel is not disabled after a channel disable request is where either the source or destination has received a split or retry response. The DMAC must keep re-attempting the transfer to the system HADDR that originally received the split or retry response until an OKAY response is returned; to do otherwise is an AMBA protocol violation.

Software may terminate all channels abruptly by clearing the global enable bit in the DMAC Configuration Register (DmaCfgReg[0]). Again,

All information provided in this document is subject to legal disclaimers.
© REALTEK 2024. All rights reserved.

you must not assume that all channels are disabled immediately after the DmaCfgReg[0] is cleared over the AHB slave interface. Consider this as a request to disable all channels. You must poll ChEnReg and confirm that all channels are disabled by reading back 0.

NOTE

- *If the channel enable bit is cleared while there is data in the channel FIFO, this data is not sent to the destination peripheral and is not present when the channel is re-enabled. For read-sensitive source peripherals, such as a source FIFO, this data is therefore lost. When the source is not a read-sensitive device (such as memory), disabling a channel without waiting for the channel FIFO to empty may be acceptable, since the data is available from the source peripheral upon request and is not lost.*
- *If a channel is disabled by software, an active single or burst transaction is not guaranteed to receive an acknowledgement.*
- *If the DMAC is configured to use defined length bursts (DMAC_INCR_BURSTS = 0), disabling the channel via software prior to completing a transfer is not supported.*

All information provided in this document is subject to legal disclaimers.
© REALTEK 2024. All rights reserved.

14 Unmanned Peripheral System (UPS)

14.1 Introduction

The Unmanned Peripheral System (UPS) can be regarded as a network that lets the different peripheral modules communicate directly with each other without the participation of MCU. With UPS, the participation of software can be minimized to avoid the time error caused by software operation while realizing the control of light dimming. The two sides communicating through UPS are called producer and consumer, among which the producer is the peripheral module sending out signal, and the consumer is another peripheral module that applies corresponding actions according to the received signal. For the scenario of multiple producers transmitting signals to the same consumer, the consumer required signal source can be selected by configuring the UPS_SRC_SEL field in REG_UPS_CTRL.

14.2 Features

- Producer and Consumer
- Configurable signal source
- Configurable input reverse
- Positive edge detection
- Various dimming signal types

14.3 Block Diagram

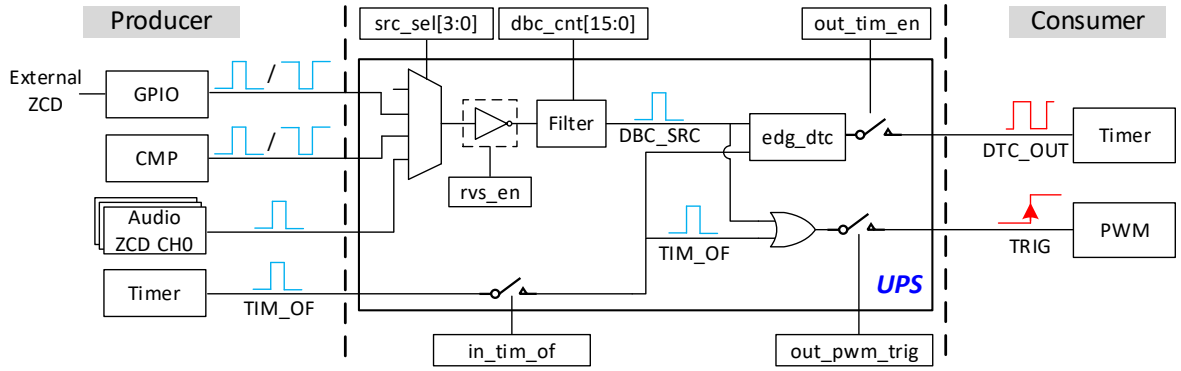


Figure 14-1 UPS block diagram

14.4 Functional Description

14.4.1 Timer

Timer can be used for time counting. *Figure 14-2* is the block diagram of timer, and *Figure 14-3* is the timing diagram of timer.

Timer works as below:

- If timer detects a high level in a terminal, the counter inside timer counts up from 0.
- If the counter value CNT reaches the max. value, the counter overflows and the of_out signal changes from low level to high level. Meanwhile, the counter continues to count from 0.
- After counting for a clock cycle, hardware will pull down the of_out signal.
- If timer detects a low level in a terminal, counter stops working and CNT retains. After enabled, the timer continues counting.

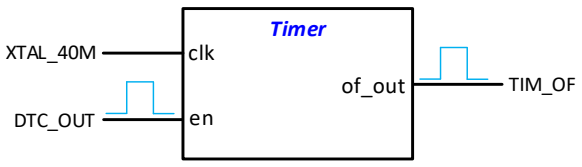


Figure 14-2 Timer block diagram

All information provided in this document is subject to legal disclaimers.
© REALTEK 2024. All rights reserved.

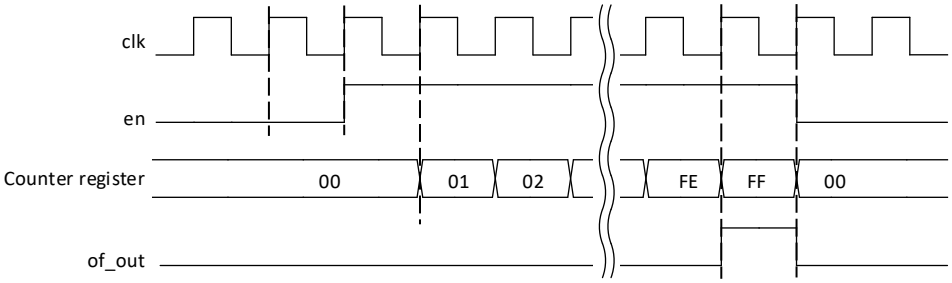


Figure 14-3 Timer timing diagram

NOTE

In the practical application of UPS, TIM10 is used as the fixed timer.

14.4.2 Deglitch Filter

In order to reduce the impact of signal glitches on the digital circuit, the deglitch filter is used to debounce signal with XTAL_40MHz sampling clock. The larger the dbc_cnt is, the wider the deglitch can be filtered, and the longer the delay will be. For different UPS signal sources, adjusting the value of dbc_cnt appropriately can filter glitches and minimize the delay caused by digital filtering at the same time.

14.4.3 Edge Detection

Figure 14-4 is the block diagram of edge detection. The input signals of edg_dtc module include clk, start_in and stop_in, and the output is dtc_out.

edg_dtc module works as below:

- edg_dtc detects the rising edges of start_in and stop_in.
- When a rising edge is detected in start_in, edg_dtc pulls up the dtc_out signal after a clock cycle.
- When a rising edge is detected in stop_in, edg_dtc pulls down the dtc_out signal after a clock cycle; otherwise, the dtc_out signal stays high.

The timing diagram of edge detection is shown in *Figure 14-5*.

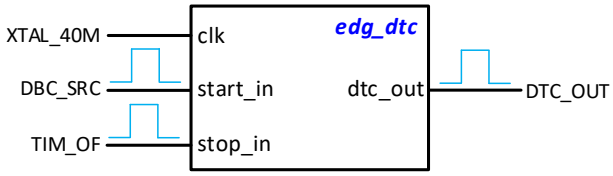


Figure 14-4 Edge detection block diagram

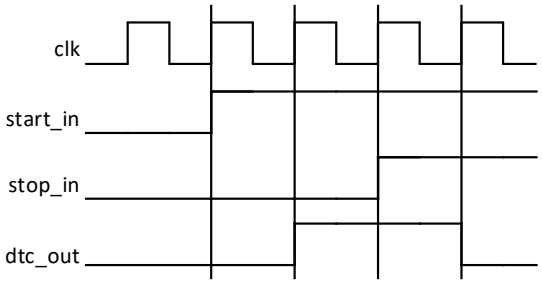


Figure 14-5 Edge detection timing diagram

14.4.4 Signal Reverse

External ZCD signals and CMP outputs can both be positive pulse or negative pulse. When it is a positive pulse signal, the system works normally; when it is a negative pulse signal, rvs_en is configured to 1 to reverse the negative pulse signal to ensure the normal operation of the system.

All information provided in this document is subject to legal disclaimers.
© REALTEK 2024. All rights reserved.

14.4.5 Dimming Signal Types

In order to meet various demands, three kinds of dimming control signals can be realized by adjusting the parameters of the PWM timer.

- OPM_DefaultLevelx = low, CCxPolarity = active_low: Pull down the control signal before the second zero-crossing point.
- OPM_DefaultLevelx = high, CCxPolarity = active_low: Pull down the control signal at the second zero-crossing point.
- OPM_DefaultLevelx = high, CCxPolarity = active_high: Pull down the control signal after the second zero-crossing point.

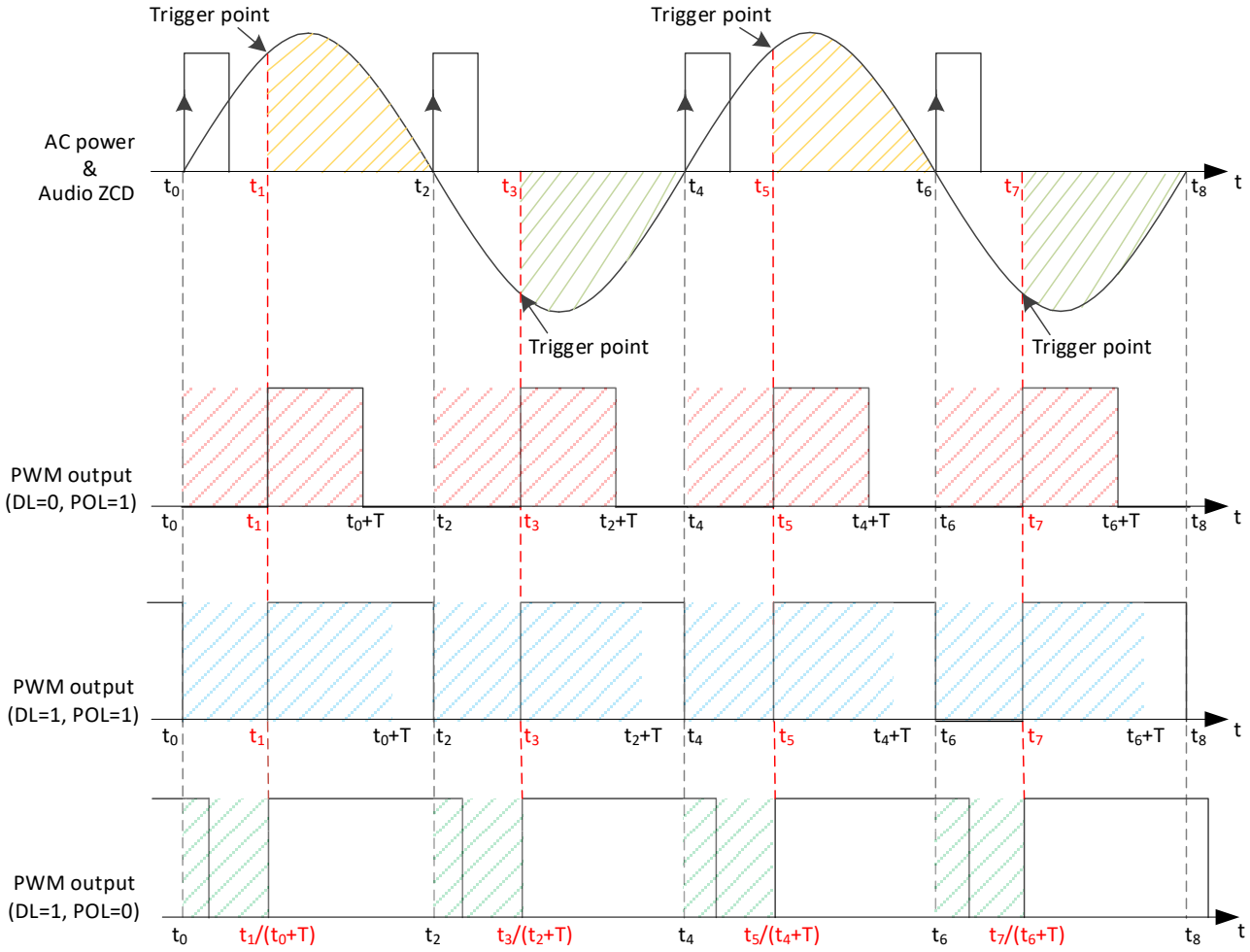


Figure 14-6 Various PWM output types

14.5 Registers

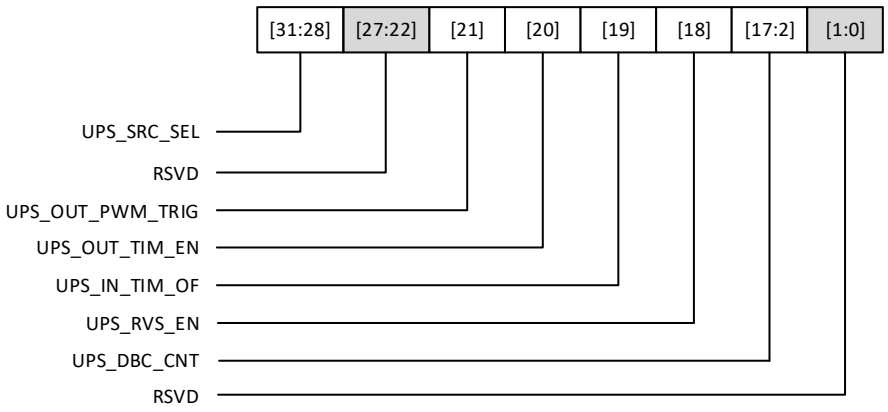
The base address of UPS registers is 0x4100_C200.

Name	Address offset	Access	Description
REG_UPS_CTRL	0C4h	R/W	UPS Control Register

14.5.1 REG_UPS_CTRL

- Name: UPS Control Register
- Size: 32 bits
- Address offset: 0C4h
- Read/write access: R/W

All information provided in this document is subject to legal disclaimers.
© REALTEK 2024. All rights reserved.



Bit	Symbol	Access	Reset	Description
31:28	UPS_SRC_SEL	R/W	0	UPS trigger source selection ● 000: None ● 001: GPIO trigger ● 010: Comparator output ● 011: Audio ZCD_0 ● 100: Audio ZCD_1 ● 101: Audio ZCD_2
27:22	RSVD	-	0	Reserved
21	UPS_OUT_PWM_TRIG	R/W	0	Enable for output trigger to PWM
20	UPS_OUT_TIM_EN	R/W	0	Enable for output a signal to basic timer
19	UPS_IN_TIM_OF	R/W	0	Enable for basic timer overflow input
18	UPS_RVS_EN	R/W	0	Invert polarity for UPS input 0: Positive edge polarity 1: Negative edge polarity
17:2	UPS_DBC_CNT	R/W	0	Debounce count for UPS function
1:0	RSVD	-	0	Reserved

All information provided in this document is subject to legal disclaimers.
© REALTEK 2024. All rights reserved.

15 General Timers

The RTL8721Dx includes ten basic timers, one capture timers, and one PWM timer.

15.1 Basic Timer

15.1.1 Introduction

The basic timers (TIM0 ~ TIM7, TIM10, TIM11) consist of a 32-bit auto-reload counter without prescaler. They may be used as generic timers for time-base generation.

15.1.2 Features

The features of basic timer are listed in *Table 15-1*.

Table 15-1 Basic timer features

Name	TIM0 ~ TIM7	TIM10/TIM11
Clock source	SDM32kHz	XTAL40MHz
Resolution	32-bit	32-bit
Prescaler	-	-
Counter mode	Up	Up
One pulse mode	-	-
PWM mode with polarity selection	-	-
Statistic pulse width	-	-
Statistic pulse number	-	-
Interrupt generation	✓	✓

15.1.3 Block Diagram

The block diagram of the basic timer is shown in *Figure 15-1*.

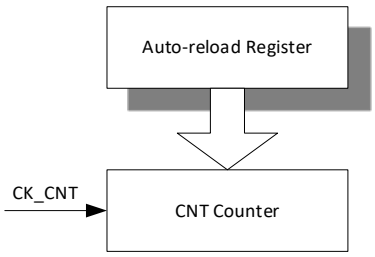


Figure 15-1 Basic timer block diagram

15.1.4 Functional Description

15.1.4.1 Upcounting Mode

This timer is a 32-bit counter with its related auto-reload register. The counter can count up.

The counter, as well as the auto-reload register, can be written or read by software. This is true even when the counter is running.

The time-base unit includes:

- Counter register (TIMx_CNT)
- Auto-reload register (TIMx_ARR)

In upcounting mode, the counter counts from 0 to the auto-reload value (the content of the TIMx_ARR register), then restarts from 0 and generates a counter overflow event.

All information provided in this document is subject to legal disclaimers.
© REALTEK 2024. All rights reserved.

15.1.4.2 Preloaded Auto-reload Register

The auto-reload register is preloaded. Writing to or reading from the auto-reload register accesses the preload register. The contents of the preload register are transferred into the shadow register permanently or at each update event (UEV) depending on the auto-reload preload enable bit (ARPE) in the TIMx_CR register.

The update event is sent when the counter reaches the overflow in count-up mode or reaches the underflow in count-down mode; or if the UDIS bit equals 0 in the TIMx_CR register, it can also be generated by software.

Setting the UG bit in the TIMx_EGR register by software also generates an update event.

The update event can be disabled by software by setting the UDIS bit in the TIMx_CR register. This is to avoid updating the shadow registers while writing new values into the preload register. Then no update event occurs until the UDIS bit has been written to 0. However, the counter restarts from 0 in up-counting mode or from auto-reload value in down-counting mode. In addition, if the URS bit in the TIMx_CR register is set, setting the UG bit generates an update event but without setting the update interrupt flag (UIF), no interrupt is sent.

When an update event occurs, all the registers are updated and the update flag (the UIF bit in TIMx_SR) is set depending on the URS bit. The auto-reload shadow register is updated with the preload value (TIMx_ARR).

15.1.4.3 Interrupt Event

The interrupt event is configured by the TIMx_DIER register.

When the UIF bit in the TIMx_SR register is asserted, the according interrupt event is asserted, which is determined by the configuration of the TIMx_DIER register. And when the counter counts overflow, the UIF is set automatically.

15.1.4.4 Security Attribution

Basic timers 0~7 are connected to the same port, and can be accessed by both secure code and non-secure code without the limitation of PPC. Therefore, it is necessary to restrict access to the secure/non-secure register inside the timer.

Every basic timer consists of a secure control register, which contains a secure enable bit. This register is used for setting the security attribution of this basic timer. The secure control register can only be accessed by secure regions, and non-secure regions cannot access this register.

- When the secure enable bit of one basic timer is set, this basic timer is secure. All registers of this timer can be accessed by a secure address, and the non-secure address cannot access them.
- When the secure enable bit of one basic timer is cleared, this basic timer is non-secure and all registers of this timer except the secure control register can be accessed by both secure address and non-secure address.

15.2 Capture Timer

15.2.1 Introduction

The capture timer (TIM9) consists of a 16-bit auto-reload counter driven by a 16-bit programmable prescaler. It can be used for a variety of purposes, including measuring the pulse lengths or numbers of input signals.

15.2.2 Features

The features of the capture timer are listed in *Table 15-2*.

Table 15-2 Capture timer features

Name	TIM9
Clock source	XTAL40MHz
Resolution	16-bit
Prescaler	16-bit
Counter mode	Up
One pulse mode	-
PWM mode with polarity selection	-

All information provided in this document is subject to legal disclaimers.
© REALTEK 2024. All rights reserved.

Statistic pulse width	✓
Statistic pulse number	✓
Interrupt generation	✓

15.2.3 Block Diagram

The block diagram of the capture timer is shown in *Figure 15-2*.

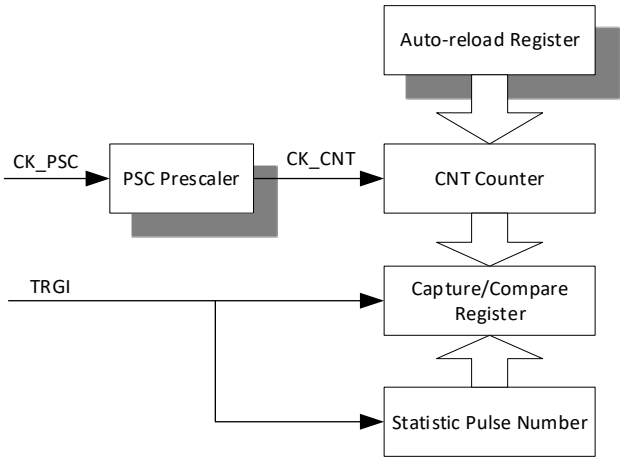


Figure 15-2 Capture timer block diagram

15.2.4 Functional Description

15.2.4.1 Up-counting Mode

This timer is a 16-bit counter with its related auto-reload register. The counter can count up. The counter clock can be divided by a 16-bit prescaler.

The counter, also the auto-reload register and the prescaler register can be written or read by software. This is true even when the counter is running.

The time-base unit includes:

- Counter register (TIMx_CNT)
- Prescaler register (TIMx_PSC)
- Auto-reload register (TIMx_ARR)

The auto-reload register is preloaded. Writing to or reading from the auto-reload register accesses the preload register. The contents of the preload register are transferred into the shadow register permanently or at each update event depending on the ARPE bit in the TIMx_CR register. The update event is sent when the counter reaches the overflow; or if the UDIS bit equals 0 in the TIMx_CR register, it can also be generated by software.

The prescaler can divide the counter clock frequency by any factor between 1 and 65536. It is based on a 16-bit counter controlled through a 16-bit register (in the TIMx_PSC register). It can be changed on the fly as this control register is buffered. The new prescaler ratio is taken into account at the next update event.

In up-counting mode, the counter counts from 0 to the auto-reload value (the content of the TIMx_ARR register), then restarts from 0 and generates a counter overflow event.

Setting the UG bit in the TIMx_EGR register by software also generates an update event.

The update event can be disabled by software by setting the UDIS bit in the TIMx_CR register. This is to avoid updating the shadow registers while writing new values to the preload registers. Then no update event occurs until the UDIS bit has been written to 0. However, the counter restarts from 0, as well as the counter of the prescaler, but the prescaler rate doesn't change. In addition, if the URS bit in the TIMx_CR register is set, setting the UG bit generates an update event but without setting the UIF flag, no interrupt is sent. This is to avoid generating both update and capture interrupts when clearing the counter on the capture event.

All information provided in this document is subject to legal disclaimers.
© REALTEK 2024. All rights reserved.

When an update event occurs, all the registers are updated and the update flag (the UIF bit in TIMx_SR) is set depending on the URS bit:

- The auto-reload shadow register is updated with the preload value (TIMx_ARR).
- The buffer of the prescaler is reloaded with the preload value (the content of the TIMx_PSC register).

15.2.4.2 Statistic Pulse Width

In pulse mode 0, setting '0' in the CC0M bit of the TIMx_CCR0 register, the capture timer can count the width of the active level of TRGI. When the TRGI is transferred to active level from inactive level, the counter is enabled automatically. When the TRGI is transferred to inactive level from active level, the counter is disabled automatically, the CC0IF is set and the current counter value is copied to the CCR0 field of the TIMx_CCR0 register.

Figure 15-3 gives an example of statistic pulse width mode when prescaler division is 1 and the positive edge of TRGI is active for capture.

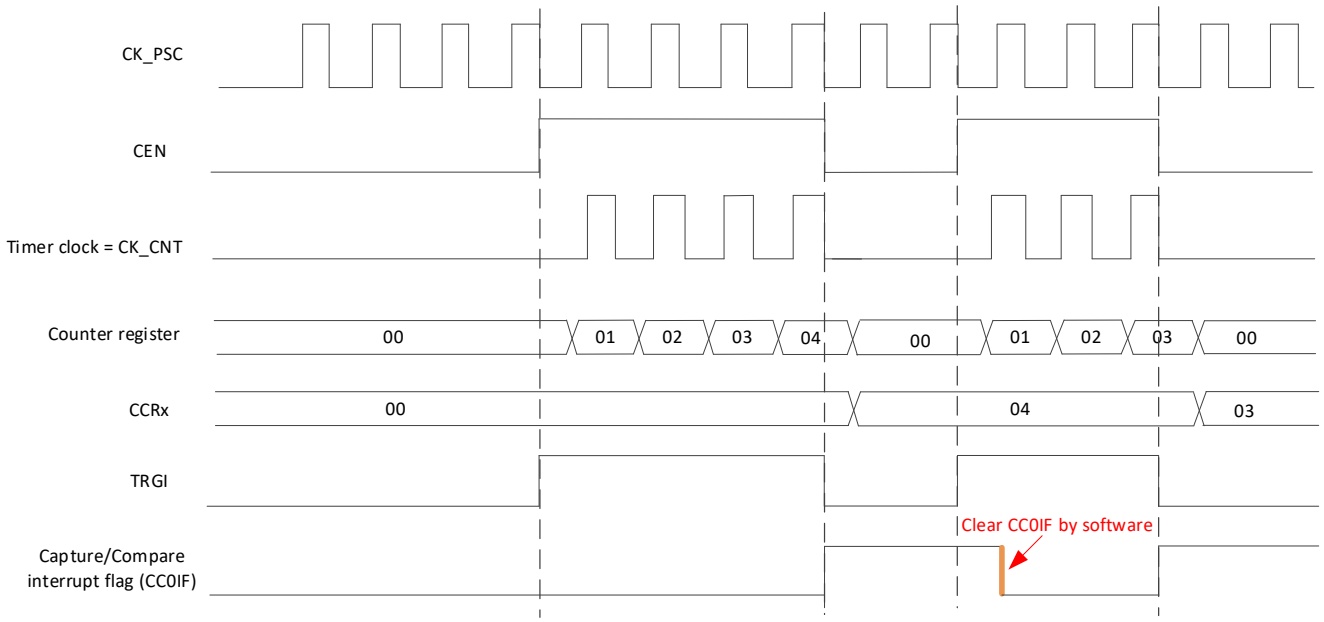


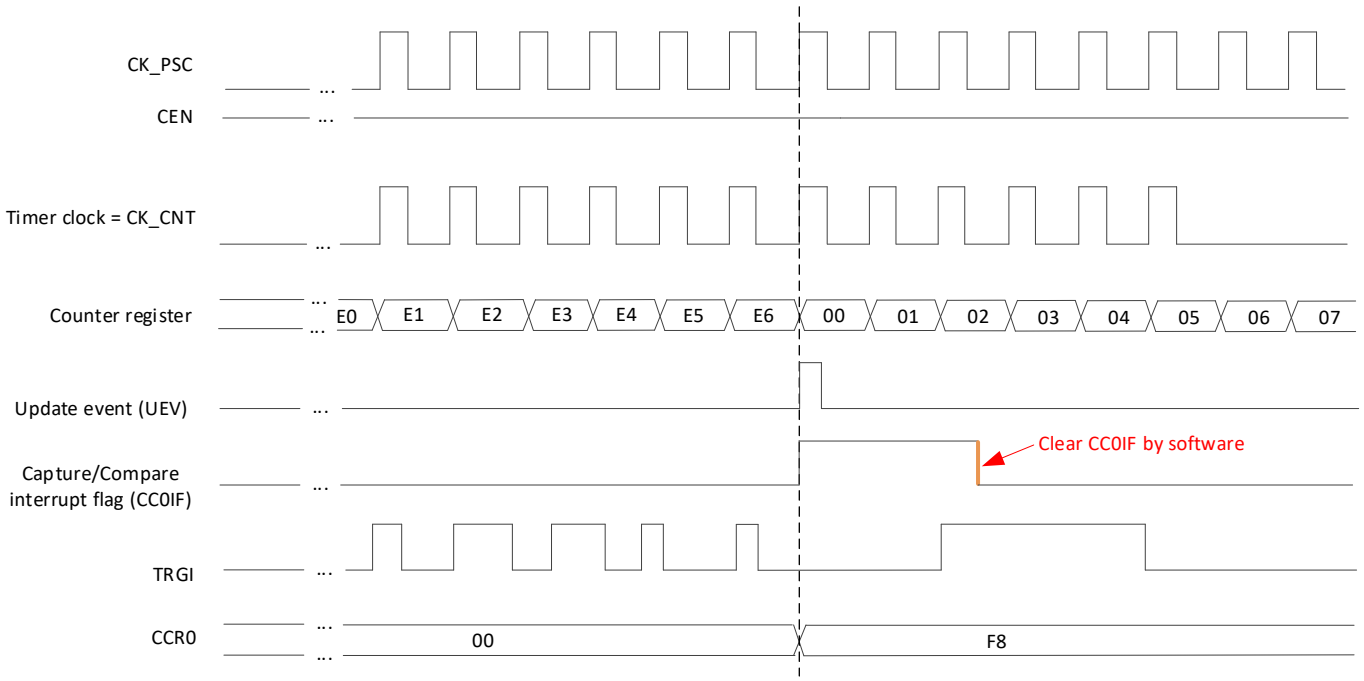
Figure 15-3 Statistic pulse width mode diagram (positive edge of TRGI is active for capture)

15.2.4.3 Statistic Pulse Number

In pulse mode 1, setting '1' in the CC0M bit of the TIMx_CCR0 register, the capture timer can count the number of active edges of TRGI in the given period. When the counter overflows, the CC0IF is set and the number is copied to the CCR0 field of the TIMx_CCR0 register.

Figure 15-4 gives an example of statistic pulse number mode when the prescaler division is 1, the positive edge of TRGI is active for capture, and the ARR field equals E6.

All information provided in this document is subject to legal disclaimers.
© REALTEK 2024. All rights reserved.



Note: The TIMx_CCR0 register captures the pulse number F8 when the counter is overflowed (UEV).

Figure 15-4 Statistic pulse number mode diagram (positive edge of TRGI is active for capture, ARR=E6)

15.3 PWM Timer

15.3.1 Introduction

The PWM timer (TIM8) consists of a 16-bit auto-reload counter driven by a 16-bit programmable prescaler. It can be used for a variety of purposes, including measuring the pulse frequency of input signals (input capture) or generating output waveforms (PWM).

Pulse lengths and waveform periods can be modulated from a few microseconds to several seconds using the timer prescaler.

15.3.2 Features

The features of the PWM timer are listed in *Table 15-3*.

Table 15-3 PWM timer features

Name	TIM8
Channels	8
Clock source	XTAL40M
Resolution	16-bit
Prescaler	16-bit
Counter mode	Up
One-pulse mode	✓
PWM mode with polarity selection	✓
Statistic pulse width	-
Statistic pulse number	-
Interrupt generation	✓
Sleep mode	✓
Duty cycle 0%/100%	✓
Phase shift	✓

15.3.3 Block Diagram

The block diagram of the PWM timer is shown in *Figure 15-5*.

All information provided in this document is subject to legal disclaimers.
© REALTEK 2024. All rights reserved.

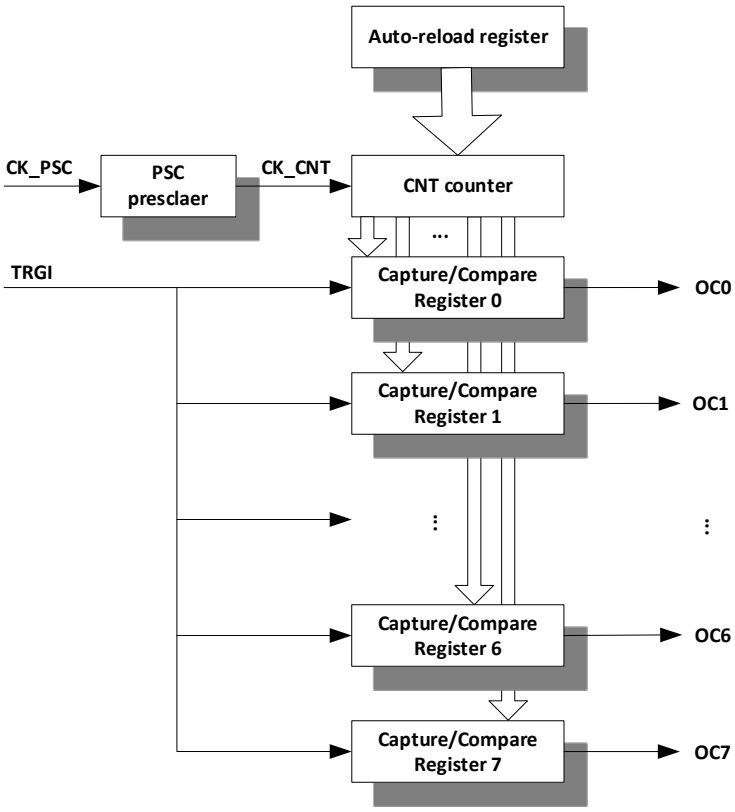


Figure 15-5 PWM timer block diagram

15.3.4 Functional Description

15.3.4.1 Up-counting Mode

This timer is a 16-bit counter with its related auto-reload register. The counter can count up. The counter clock can be divided by a 16-bit prescaler.

The counter, also the auto-reload register and the prescaler register can be written or read by software. This is true even when the counter is running.

The time-base unit includes:

- Counter register (TIMx_CNT)
- Prescaler register (TIMx_PSC)
- Auto-reload register (TIMx_ARR)

The auto-reload register is preloaded. Writing to or reading from the auto-reload register accesses the preload register. The contents of the preload register are transferred into the shadow register permanently or at each update event depending on the ARPE bit in the TIMx_CR register. The update event is sent when the counter reaches the overflow; or if the UDIS bit equals 0 in the TIMx_CR register, it can also be generated by software.

The prescaler can divide the counter clock frequency by any factor between 1 and 65536. It is based on a 16-bit counter controlled through a 16-bit register (in the TIMx_PSC register). It can be changed on the fly as this control register is buffered. The new prescaler ratio is taken into account at the next update event.

Figure 15-6 and *Figure 15-7* give some examples of the counter behavior when the prescaler ratio is changed on the fly.

All information provided in this document is subject to legal disclaimers.
© REALTEK 2024. All rights reserved.

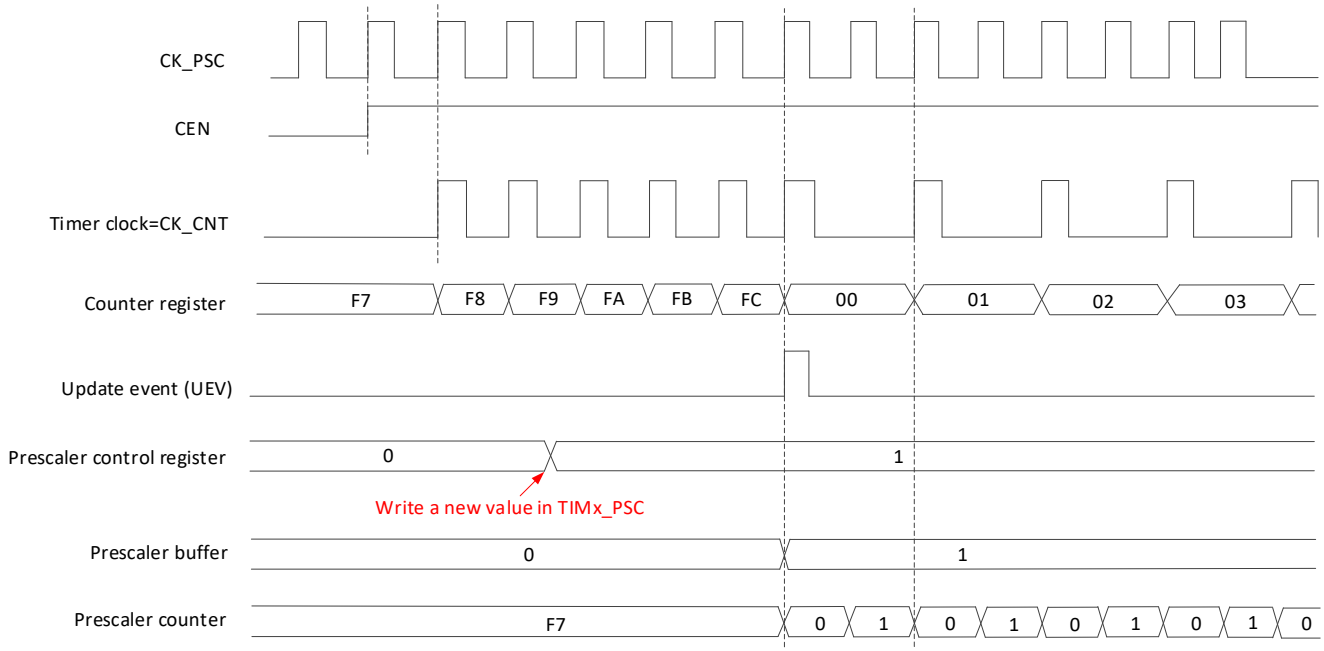


Figure 15-6 Counter timing diagram with prescaler division change from 1 to 2

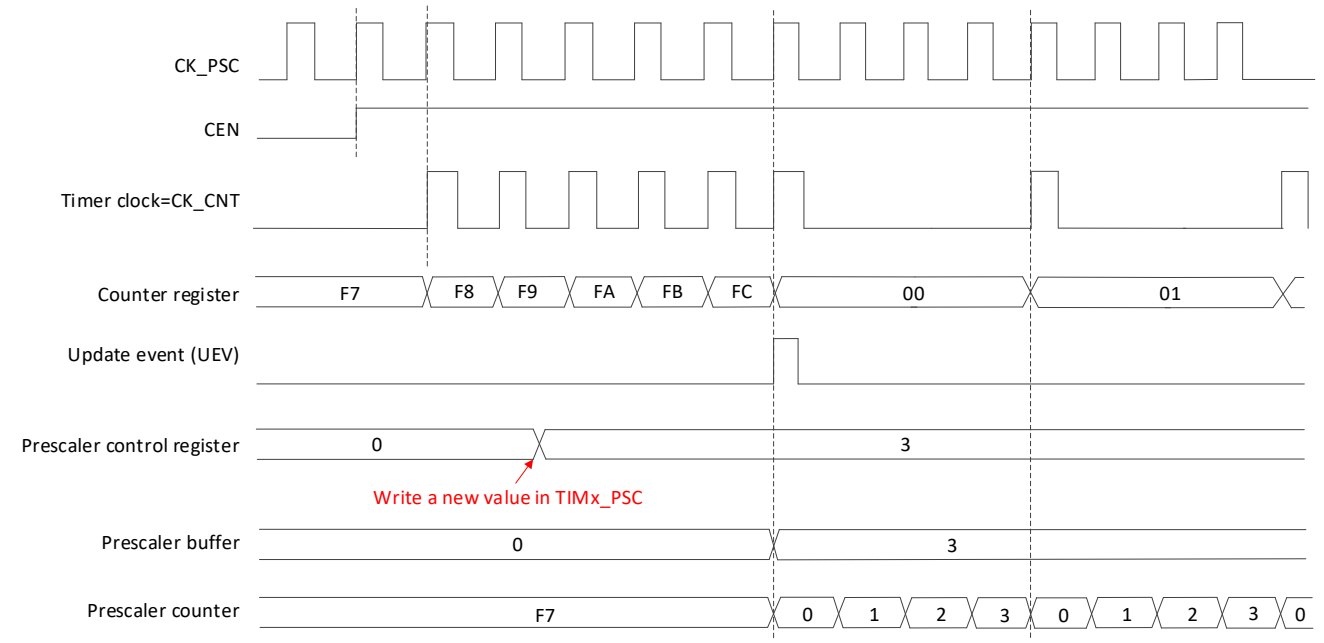


Figure 15-7 Counter timing diagram with prescaler division change from 1 to 4

In up-counting mode, the counter counts from 0 to the auto-reload value (the content of the TIMx_ARR register), then restarts from 0 and generates a counter overflow event.

Setting the UG bit in the TIMx_EGR register by software also generates an update event.

The update event can be disabled by software by setting the UDIS bit in the TIMx_CR register. This is to avoid updating the shadow registers while writing new values to the preload registers. No update event occurs until the UDIS bit has been written to 0. However, the counter restarts from 0, as well as the counter of the prescaler, but the prescale rate doesn't change. In addition, if the URS bit in the TIMx_CR register is set, setting the UG bit generates an update event but without setting the UIF flag, no interrupt is sent. This is to avoid generating both update and capture interrupts when clearing the counter on the capture event.

When an update event occurs, all the registers are updated and the update flag (the UIF bit in TIMx_SR) is set depending on the URS bit:

- The auto-reload shadow register is updated with the preload value (TIMx_ARR).
- The buffer of the prescaler is reloaded with the preload value (the content of the TIMx_PSC register)

All information provided in this document is subject to legal disclaimers.
© REALTEK 2024. All rights reserved.

Figure 15-8 to *Figure 15-13* show some examples of the counter behavior for different clock frequencies when the ARR field equals 0x36.

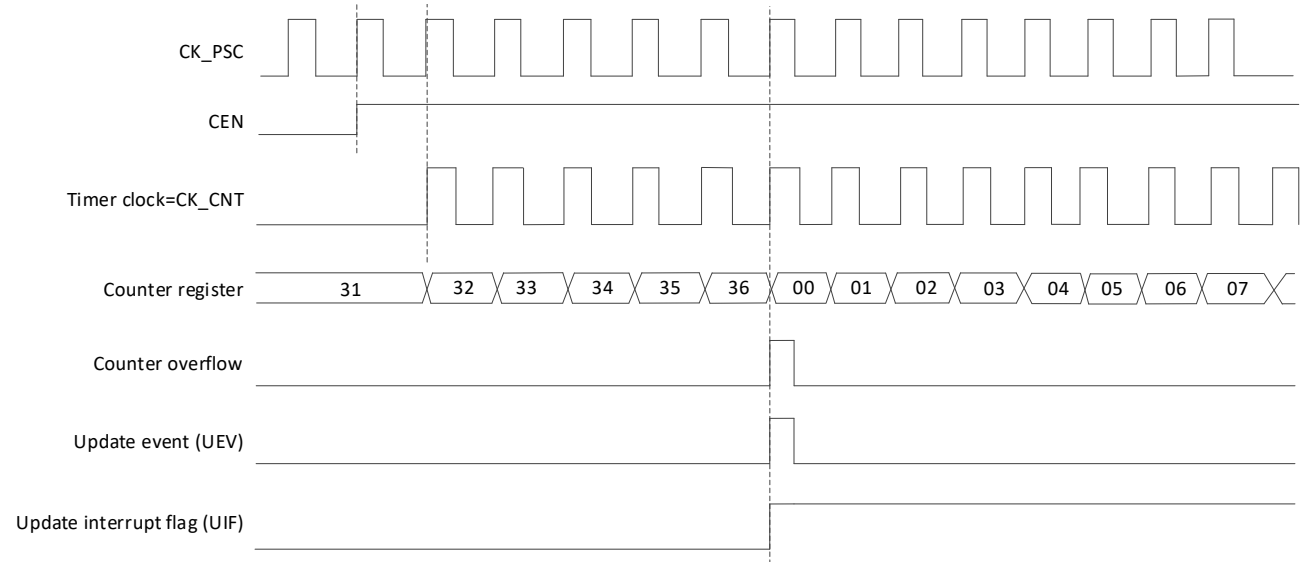


Figure 15-8 Counter timing diagram (internal clock divided by 1)

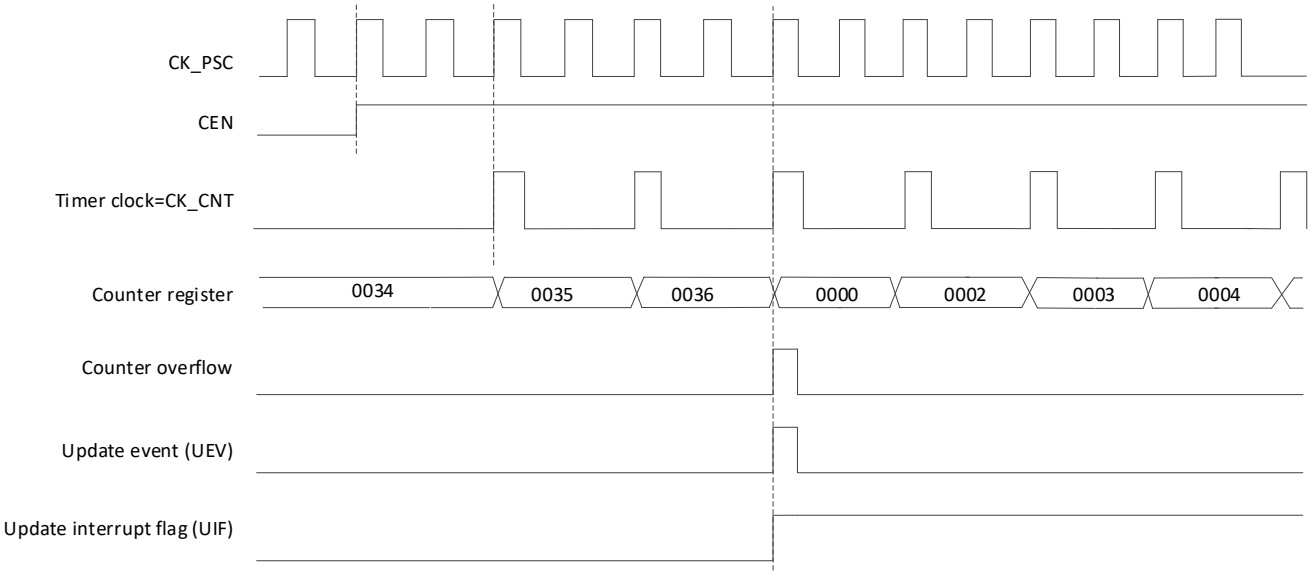


Figure 15-9 Counter timing diagram (internal clock divided by 2)

All information provided in this document is subject to legal disclaimers.
© REALTEK 2024. All rights reserved.

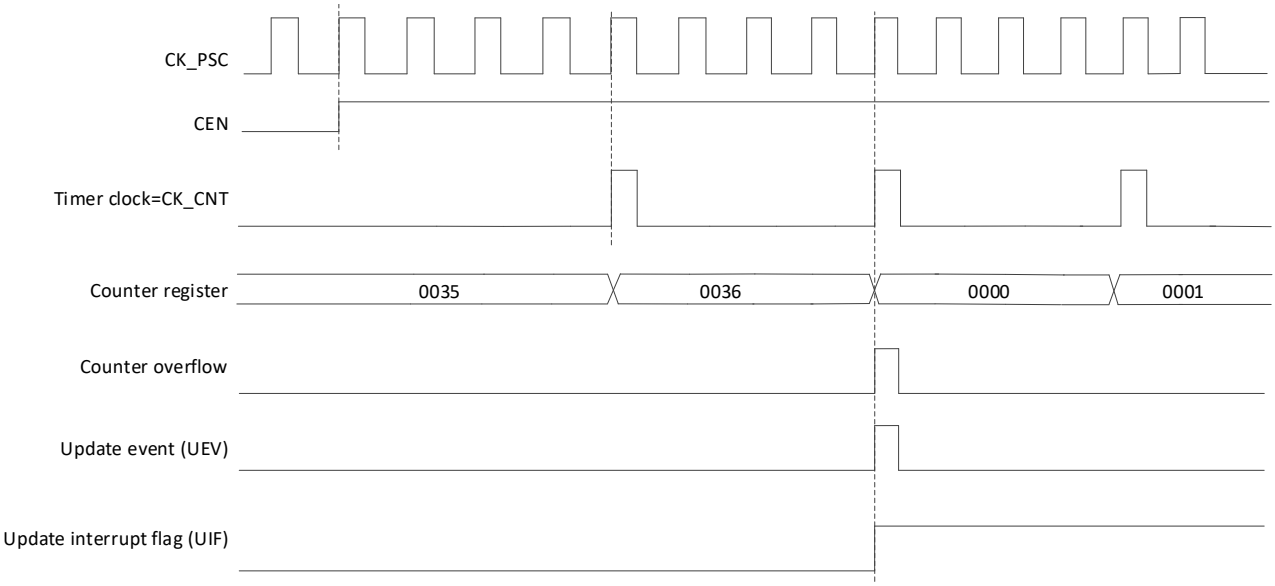


Figure 15-10 Counter timing diagram (internal clock divided by 4)

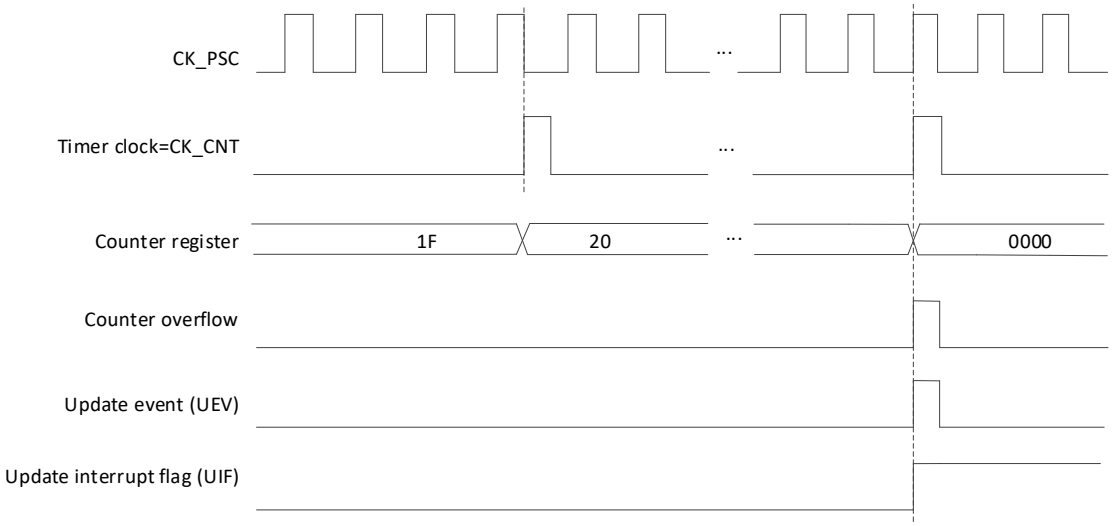


Figure 15-11 Counter timing diagram (internal clock divided by *N*)

All information provided in this document is subject to legal disclaimers.
© REALTEK 2024. All rights reserved.

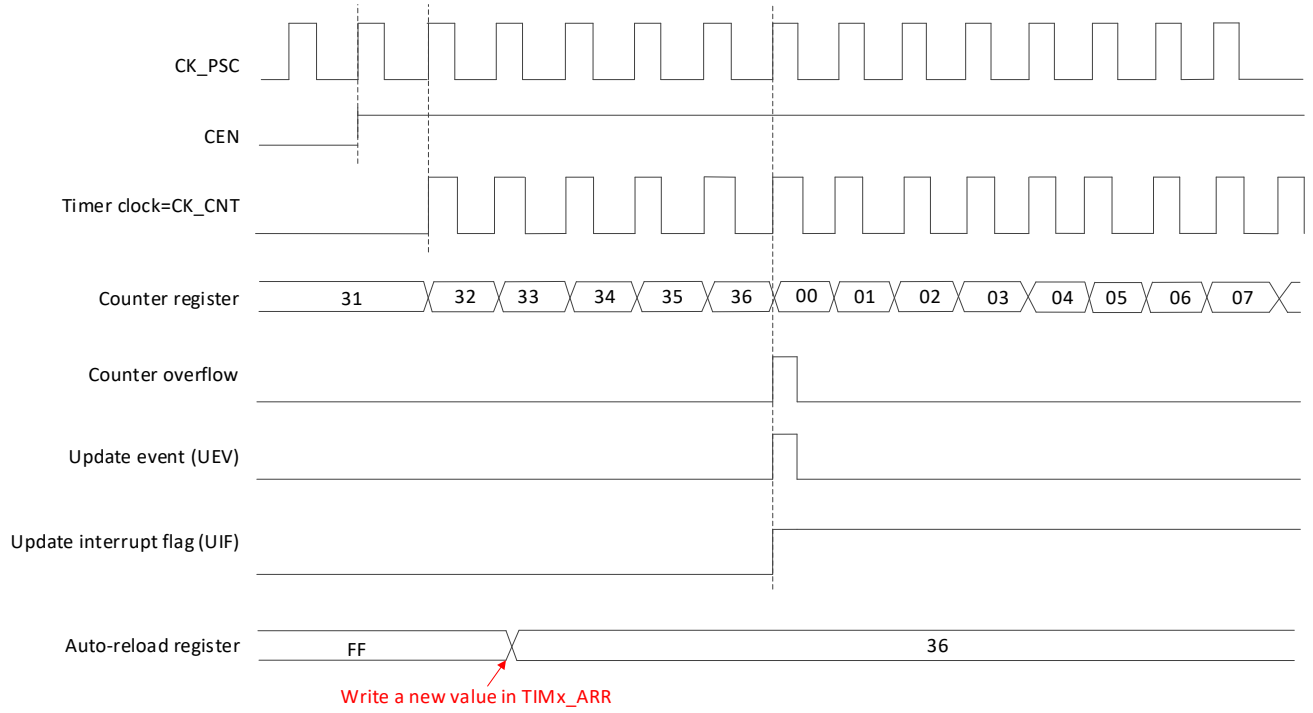


Figure 15-12 Counter timing diagram, update event when ARPE=0 (TIMx_ARR not preloaded)

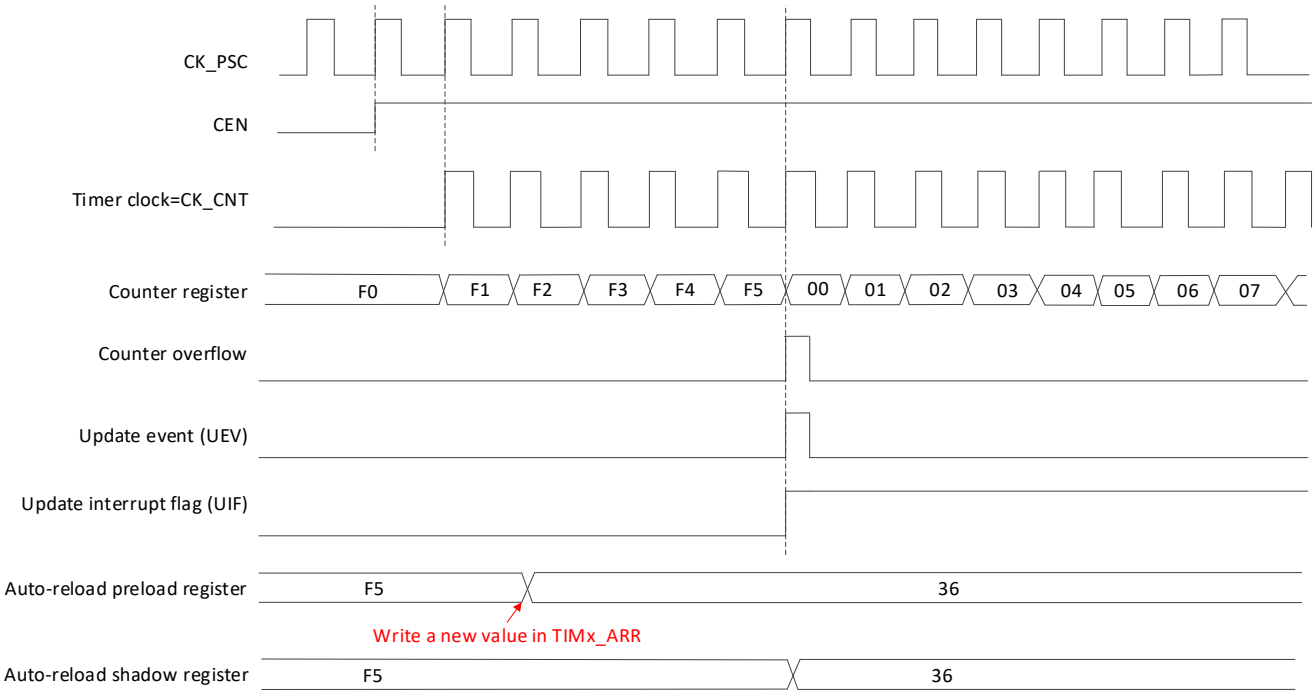


Figure 15-13 Counter timing diagram, update event when ARPE=1 (TIMx_ARR preloaded)

15.3.4.2 Input Capture Mode

In input capture mode, setting '1' in the CCxM bit of the TIMx_CCRx register, the CCRx field of capture/compare registers (TIMx_CCRx) is used to latch the value of the counter after a transition detected by the TRGI signal. When a capture occurs, the corresponding CCxIF flag in the TIMx_SR register is set and an interrupt can be sent if they are enabled. CCxIF can be cleared by software by writing it to '1'.

15.3.4.3 PWM Mode

Pulse Width Modulation mode allows you to generate a signal with a frequency determined by the value of the TIMx_ARR register and a duty cycle determined by the value of the CCRx field of the TIMx_CCRx register.

All information provided in this document is subject to legal disclaimers.
© REALTEK 2024. All rights reserved.

- Period: $\text{Period} = (\text{ARR} + 1) * \ T_{CNT}$
- Duty cycle: $D_{PWM} = \frac{CCRx \ * \ T_{CNT}}{T_{PWM}}$
- Where $T_{CNT} = T_{\text{XTAL}} * (\text{PSC} + 1)$

The PWM mode can be selected independently on each channel (one PWM per OCx output) by setting '0' in the OCxM bits in the TIMx_CCRx register. You must enable the corresponding preload register by setting the OCxPE bit in the TIMx_CCRx register, and eventually the auto-reload preload register by setting the ARPE bit in the TIMx_CR register.

As the preload registers are transferred to the shadow registers only when an update event occurs, before starting the counter, you have to initialize all the registers by setting the UG bit in the TIMx_EGR register.

OCx polarity is software programmable, using the CCxP bit in the TIMx_CCRx register. It can be programmed as active high or active low.

In PWM mode, TIMx_CNT and CCRx (in TIMx_CCRx) are always compared to determine whether TIMx_CNT< CCRx. The PWM signal OCx is active as long as TIMx_CNT < CCRx, otherwise it becomes inactive. If the compare value in TIMx_CCRx is greater than the auto_reload value in TIMx_ARR, then PWM signal OCx output is active all the period. If the compare value is 0, then the PWM signal OCx output is inactive all the periods.

The timer is only able to generate PWM in edge-aligned mode. *Figure 15-14* shows some edge-aligned PWM waveforms in an example where TIMx_ARR=8.

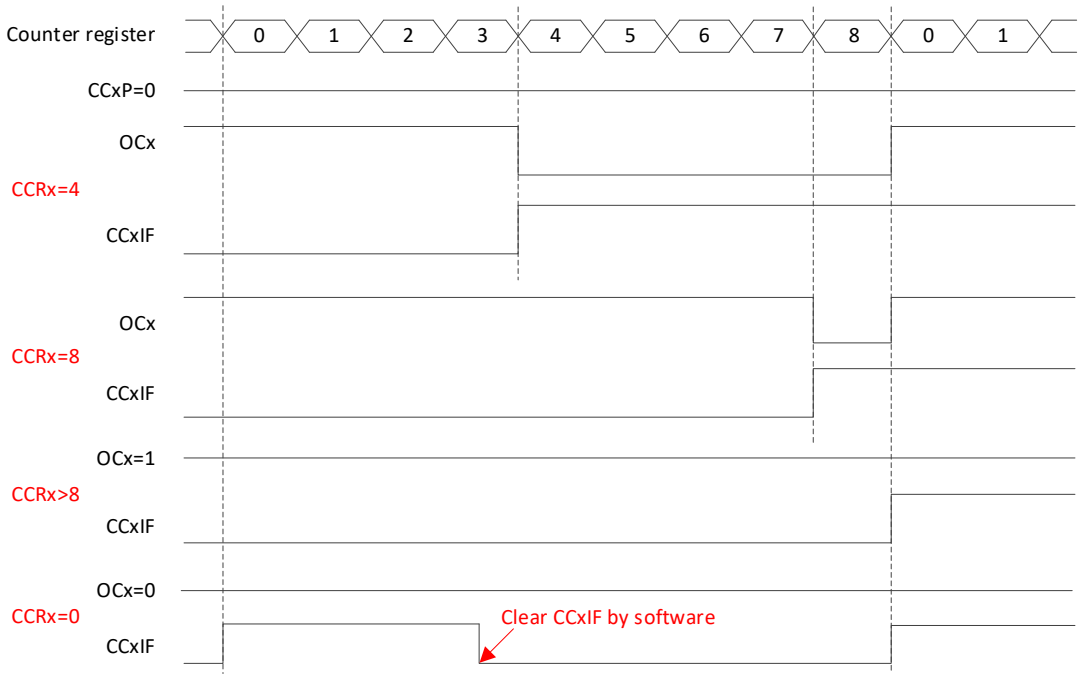


Figure 15-14 Edge-aligned PWM waveforms (ARR=8, CCxP=0)

15.3.4.4 One-pulse Mode

One-pulse mode (OPM) is a particular case of the previous modes. It allows the counter to be started in response to a stimulus and to generate a pulse with a programmable length after a programmable delay.

Starting the counter can be controlled through the active edge of TRGI, which can be the positive edge or negative edge of both edges. Generating the waveform can be done in PWM mode. You select One-pulse mode by setting the OPM bit in the TIMx_CR register. This makes the counter stop automatically at the next update.

The one-pulse mode timing is shown in *Figure 15-15*.

All information provided in this document is subject to legal disclaimers.
© REALTEK 2024. All rights reserved.

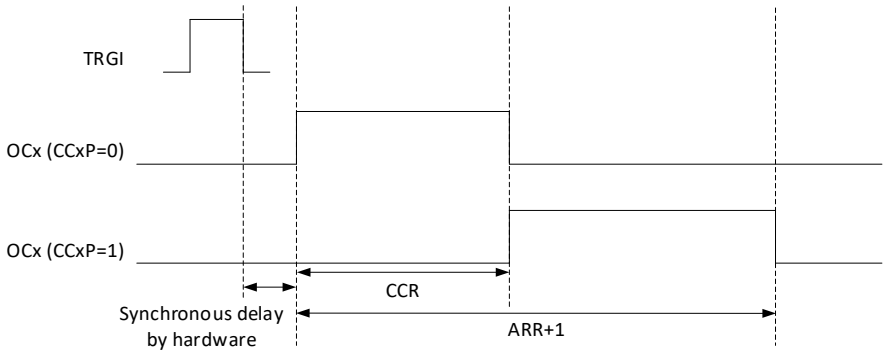


Figure 15-15 One-pulse mode timing

NOTE

Since the 6 channels are independent of each other, when TRGI is used to trigger one or more channels to output a PWM signal, it can also be used as a capture source in the input capture mode of other channels.

15.3.4.5 Phase Synchronization Function

This function is for users to configure a particular phase shift between channels. It can only be applied in PWM mode and one-pulse mode, input capture mode is not affected.

There are 7 counters in the PWM timer, including one CNT in the TIM_CNT register and six-channel counters for six channels. Each channel has one channel counter, the value of which can be from reading TIM_PHASECNTx(x=0, 1, 2, 3, 4, 5) registers. The value of each channel counter is calculated from the CNT value in the TIM_CNT register. When the counter value in TIM_CNT changes, the channel value always updates immediately. The waveform of each channel uses its own channel counter as a benchmark.

The phase shift between channels is achieved by setting a phase shift of the channel counter compared to the CNT value in the TIM_CNT register. The phase shift function can be configured by TIM_PSYNCx (x=0, 1, 2, 3, 4, 5) registers. When phase shift function is enabled by setting SYNCENx bit in TIM_PSYNCx register, users can configure a phase shift between this channel counter and CNT value in TIM_CNT, and the phase shift value is set by SYNCDIR and SYNCPHASE in TIM_PSYNCx. When the phase shift function is disabled, the channel counter value of this channel is exactly the same as CNT. The channel counter value is:

```
channel counter value = CNT - (-1)SYNCDIR *SYNCPHASE
```

Hardware can always ensure channel value in the range 0-ARR.

- When the channel counter value is above ARR, it changes to the old value - ARR immediately to keep in range 0-ARR.
- When the channel counter value is below zero, it changes to the old value + ARR immediately.

Figure 15-16 shows change curves of channel counter value compared to the counter, and the corresponding waveforms generated from channel counter of channels 0, 1, 2. In the following configuration, the phase shift between every two channels is 1/3.

In the above example,

- For channel 0, SYNCEN0 of channel 0 is 0, so that channel counter 0 is always equal to CNT.
  - When channel counter 0 is below CCR0, which is 30, channel 0 outputs a high level.
  - When channel counter 0 is above CCR0, channel 0 outputs a low level.
- For channel 1, SYNCEN1 is set to 1, so that channel counter 1 has a phase shift with TIM_CNT. The value of channel counter 1 is CNT + 66, in which CNT is the value of TIM_CNT.
  - When channel counter 1 is below CCR1, channel 1 outputs a high level, and vice versa.
- For channel 2, channel counter 2 is CNT + 33.

The software ensures that SYNCPHASEx is always smaller than the ARR value. Otherwise, a low level will be generated and output.

After the PWM timer is enabled and waveforms are generated. Users may want to update some parameters such as period, duty cycle, and phase shift value. Because ARR, PSC, CCRx, and PSYNCx registers can be configured to be preloaded or not, there are three ways for users to change configurations on-the-fly.

(1) Set the registers to be not preloaded and then set a new value, the new value will take effect immediately after the register value is changed.
(2) Set the registers to be preloaded and then set a new value, the new value will update after the next UEV of CNT.
(3) Set the registers to be preloaded, set a new value, and then write UG bit. After that CNT will clear and recount from 0, and the new value will take effect immediately. This can help when multiple parameters need to be updated immediately.

All information provided in this document is subject to legal disclaimers.
© REALTEK 2024. All rights reserved.

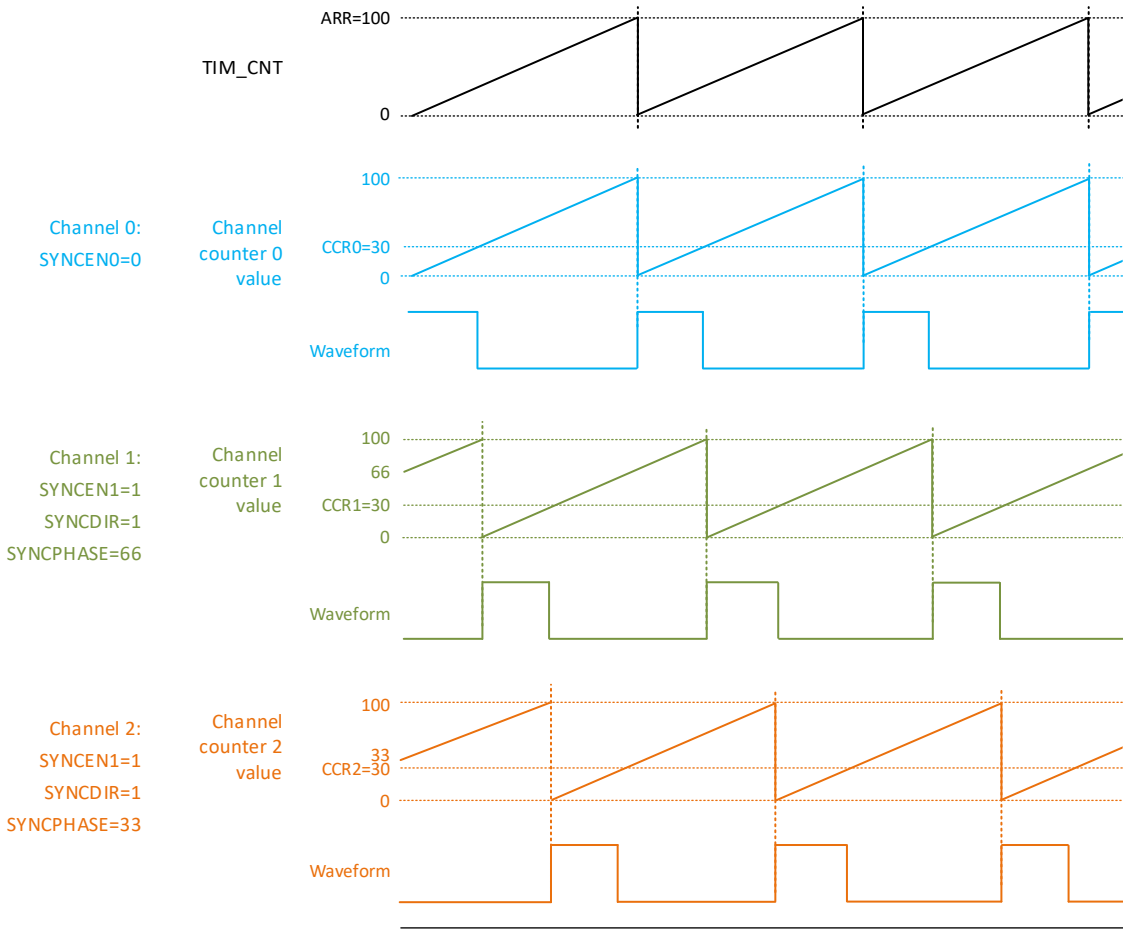


Figure 15-16 Phase shift function

The following sections illustrate some cases.

15.3.4.5.1 Update Immediately

(1) Change the period

a) Increase ARR value

Figure 15-17 illustrates when the ARR value is changed from 100 to 150. Because ARR is increased, the waveforms in the red cycle show that all three channels are in low-level states.

b) Decrease ARR value

Figure 15-18 shows when the ARR value is changed from 100 to 80. Before ARR is decreased, there is only one channel in high-level state at the same time, After ARR is set to 80, the waveforms in red cycle show that channel 0 and 2 are both in high-level states in this case.

All information provided in this document is subject to legal disclaimers.
© REALTEK 2024. All rights reserved.

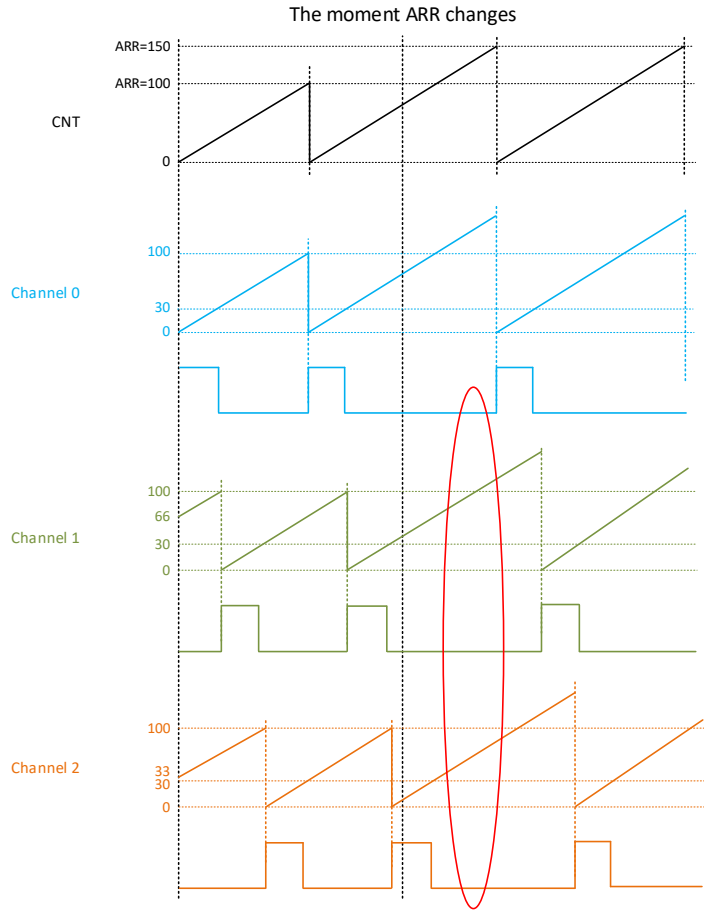


Figure 15-17 Waveform when ARR changes from 100 to 150

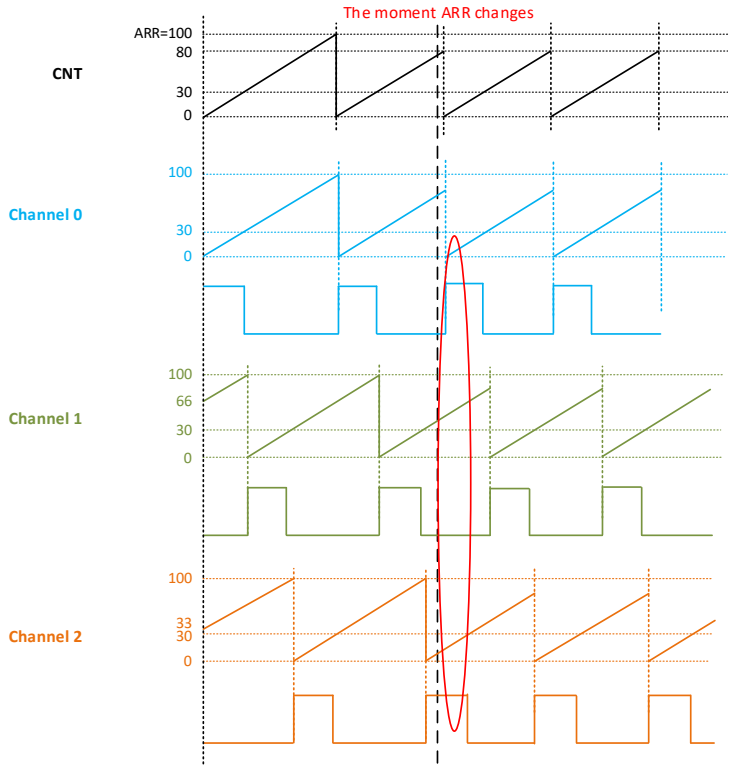


Figure 15-18 Waveform when ARR changes from 100 to 80

(2) Change the duty cycle

The duty cycle of each channel depends on its CCRx value which is configured individually in its own TIM_CCRx registers. They can't be

All information provided in this document is subject to legal disclaimers.
© REALTEK 2024. All rights reserved.

updated exactly at the same time due to the time gap that CPU executes instructions. The length of this time gap is unpredictable. The following examples ignore this time gap.

a) Increase CCRx value

Figure 15-19 displays when CCRx changes from 30 to 40. For channel 2, CCR2 is changed when the channel counter 2 value is 35 and also 30 < 35 < 40, so that the waveform of channel 2 falls into high level simultaneously when CCRs is updated. This causes a glitch. Also, channel 0 and 1 are both in high-level state because of increasing its CCRx in this case.

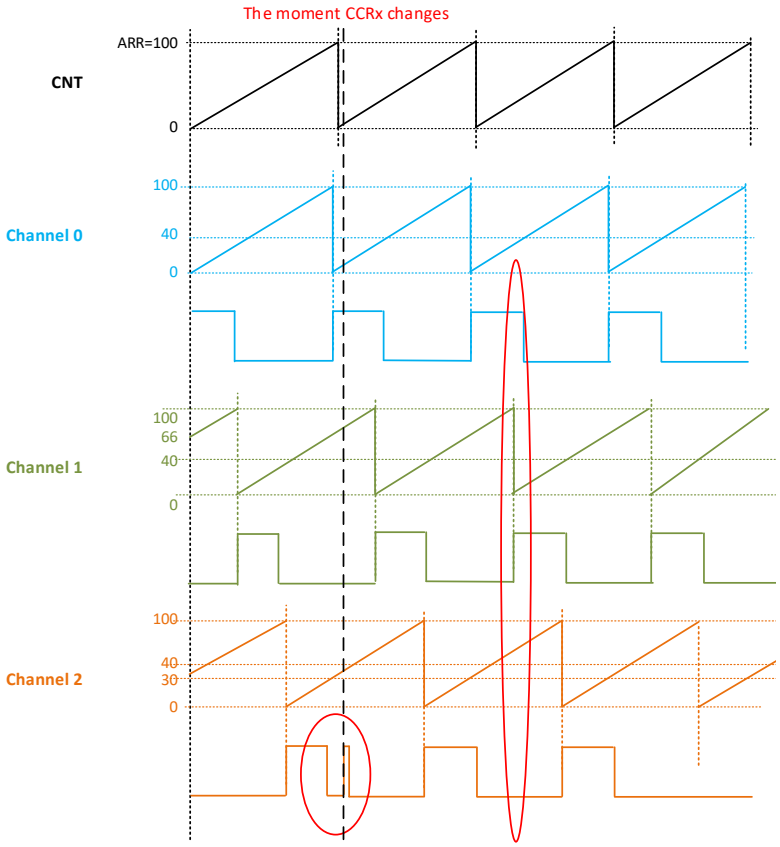


Figure 15-19 Waveform when CCRx changes from 30 to 40

b) Decrease CCRx value

Figure 15-20 shows when CCRx changes from 30 to 20. For channel 2, CCR2 is changed when the channel counter 2 value is 25 and also 20 < 25 < 30, so that the waveform of channel 2 falls into low level simultaneously when CCRs is updated. The duty cycle of channel 2 in this period is neither 20% nor 30%, but 25% instead.

All information provided in this document is subject to legal disclaimers.
© REALTEK 2024. All rights reserved.

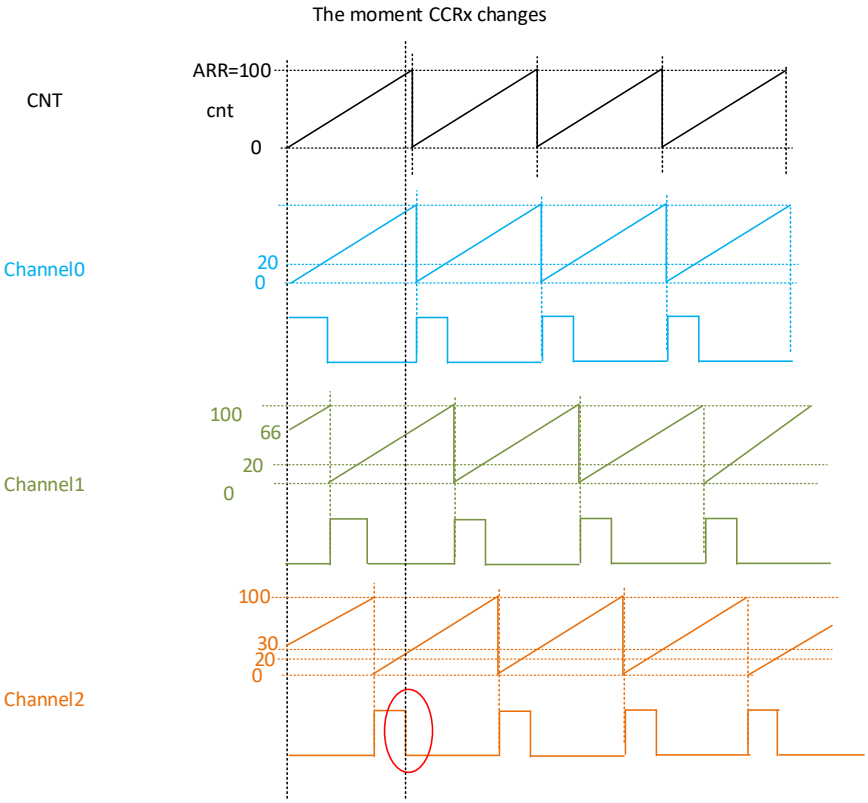


Figure 15-20 Waveform when CCRx changes from 30 to 20

(3) Change phase shift value

The phase shift of each channel is configured individually in TIM_PSYNCx registers. They can't be updated exactly at the same time and the following example ignores this time gap. Also, the change of phase shift always results in discrete channel counters and waveforms. *Figure 15-21* is an example that the phase shift of channel 1 changes from 66 to 36 and channel 2 from 33 to 63. For channel 1, channel counter1 is above and below CCR1 before and after changing. For channel 2, the channel counter 2 is below and above CCR2 before and after changing.

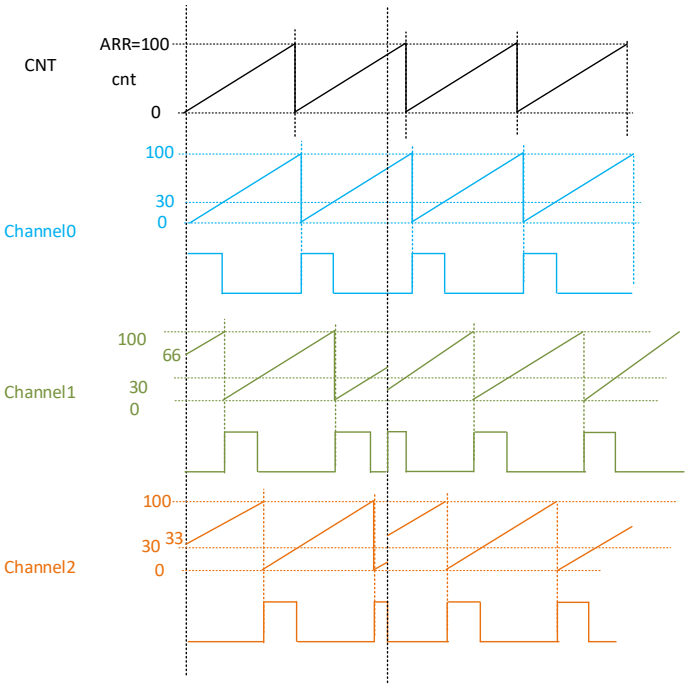


Figure 15-21 Waveforms when phase shifts change

All information provided in this document is subject to legal disclaimers.
© REALTEK 2024. All rights reserved.

15.3.4.5.2 Update after Next UEV of CNT

(1) Change the period

a) Increase ARR value

Figure 15-22 illustrates when the ARR value is changed from 100 to 150. Because ARR is increased, the waveforms in the red cycle show that all three channels are in low-level states in this case.

b) Decrease ARR value

Figure 15-23 shows when the ARR value is changed from 100 to 80. Before ARR is decreased, there is only one channel in high-level state at the same time, After ARR is set to 80, the waveforms in the red cycle show that channel 0 and 2 are both in high-level states in this case.

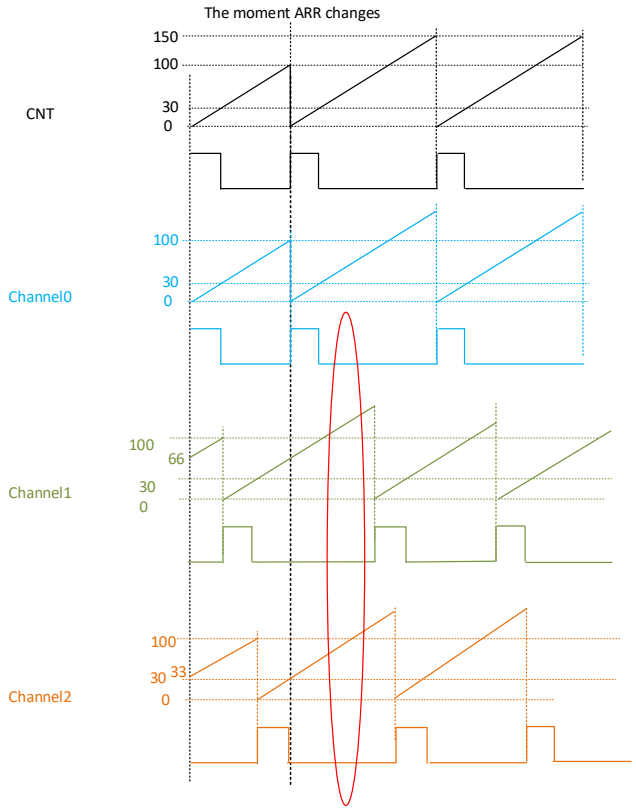


Figure 15-22 Waveform when ARR changes from 100 to 150

All information provided in this document is subject to legal disclaimers.
© REALTEK 2024. All rights reserved.

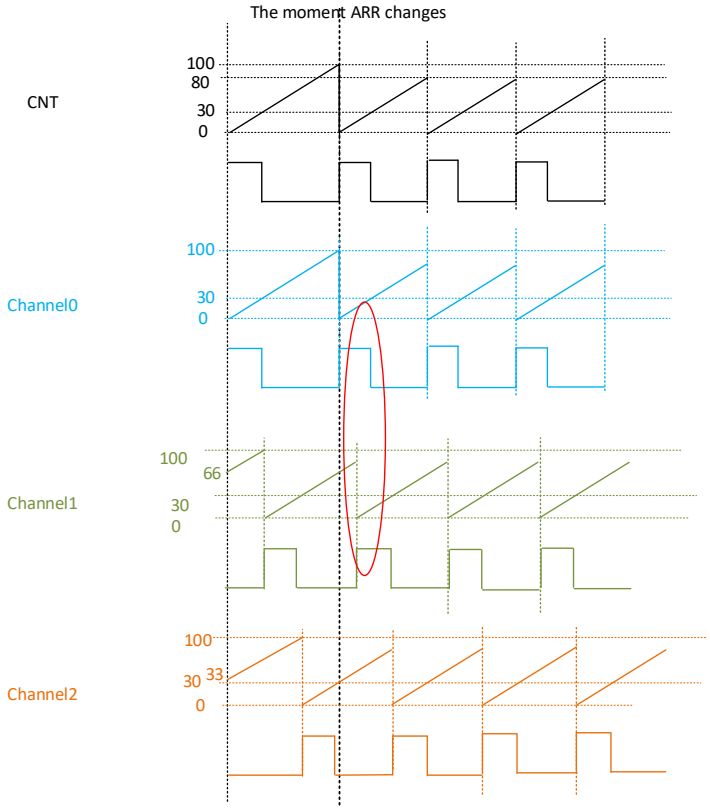


Figure 15-23 Waveform when ARR changes from 100 to 80

(2) Change the duty cycle

The following examples ignore the time gap of configuration. The waveform is similar to cases that CCRx updates immediately.

a) Increase CCRx value

Figure 15-24 displays when CCRx changes from 30 to 40.

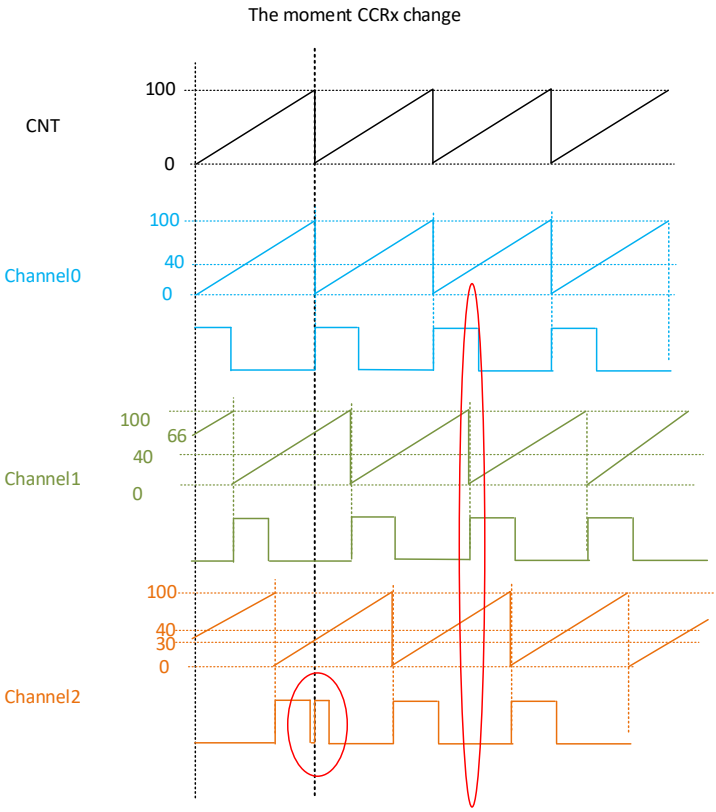


Figure 15-24 Waveform when CCRx changes from 30 to 40

All information provided in this document is subject to legal disclaimers.
© REALTEK 2024. All rights reserved.

b) Decrease CCRx value

Figure 15-25 shows when CCRx changes from 30 to 20.

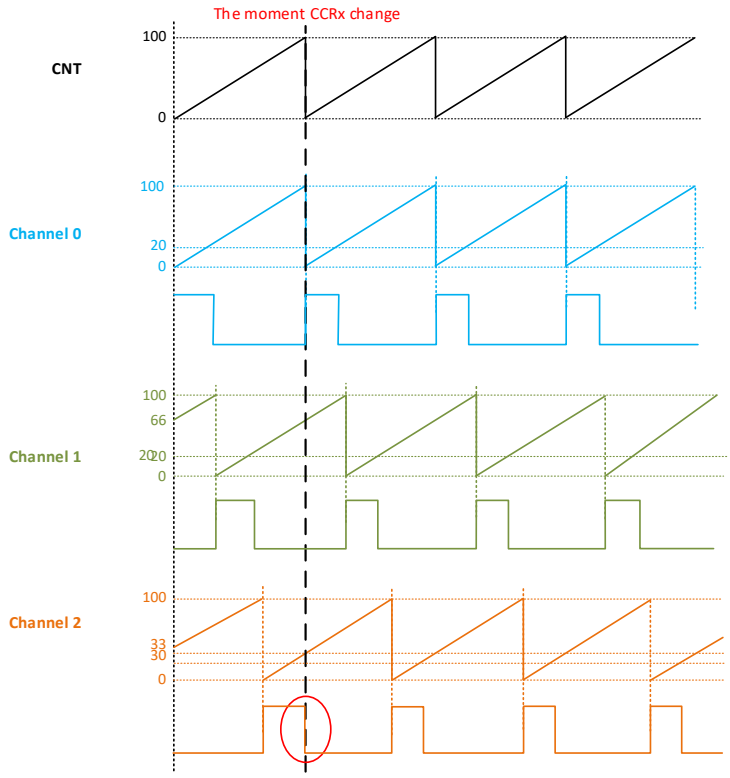


Figure 15-25 Waveform when CCRx changes from 30 to 20

(3) Change phase shift value

The following example ignores the time gap of configuration; the waveform is similar to the cases that phase shifts update immediately.

All information provided in this document is subject to legal disclaimers.
© REALTEK 2024. All rights reserved.

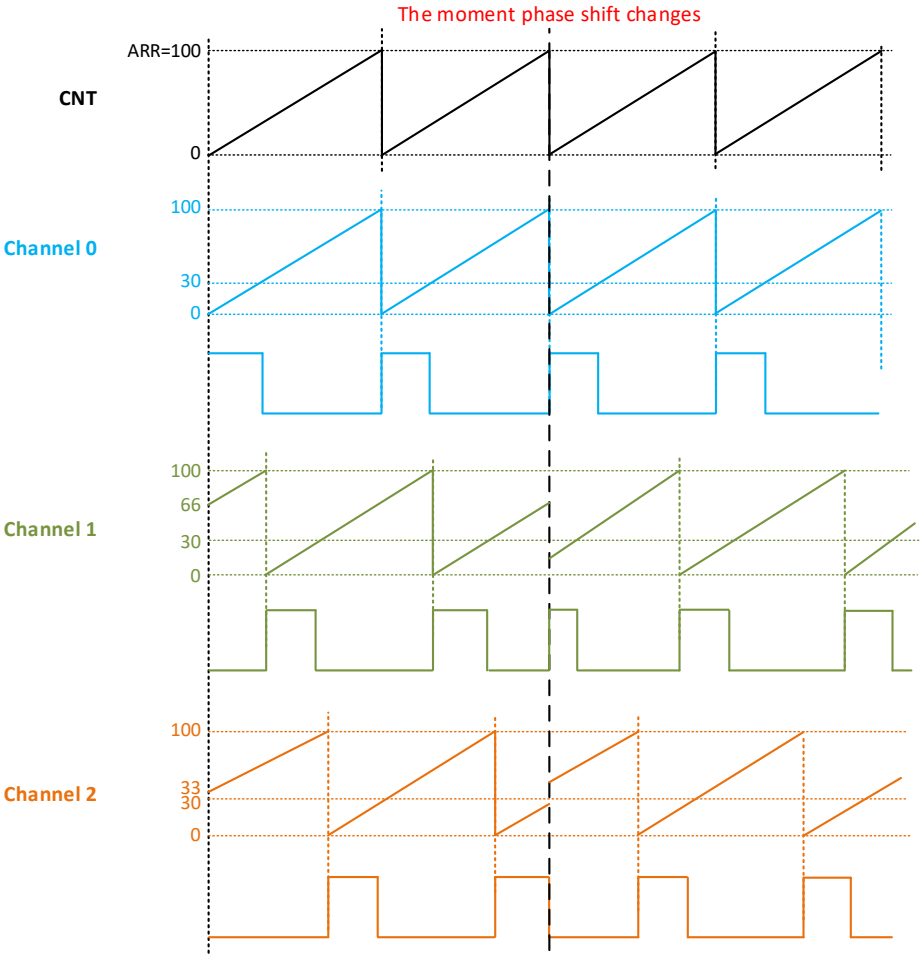


Figure 15-26 Waveform when phase shift changes

15.3.4.5.3 Update after Writing UG Bit

Before writing the UG bit, the waveform was generated according to the old configuration. The new configuration comes into effect after writing the UG bit, and CNT and channel counter are cleared to 0. In the following example, ARR changes from 100 to 150, CCRx is not changed, the phase shift function of channel 0 is disabled, the phase shift of channel 1 changes from 66 to 33, and the phase shift of channel 2 changes from 33 to 66.

All information provided in this document is subject to legal disclaimers.
© REALTEK 2024. All rights reserved.

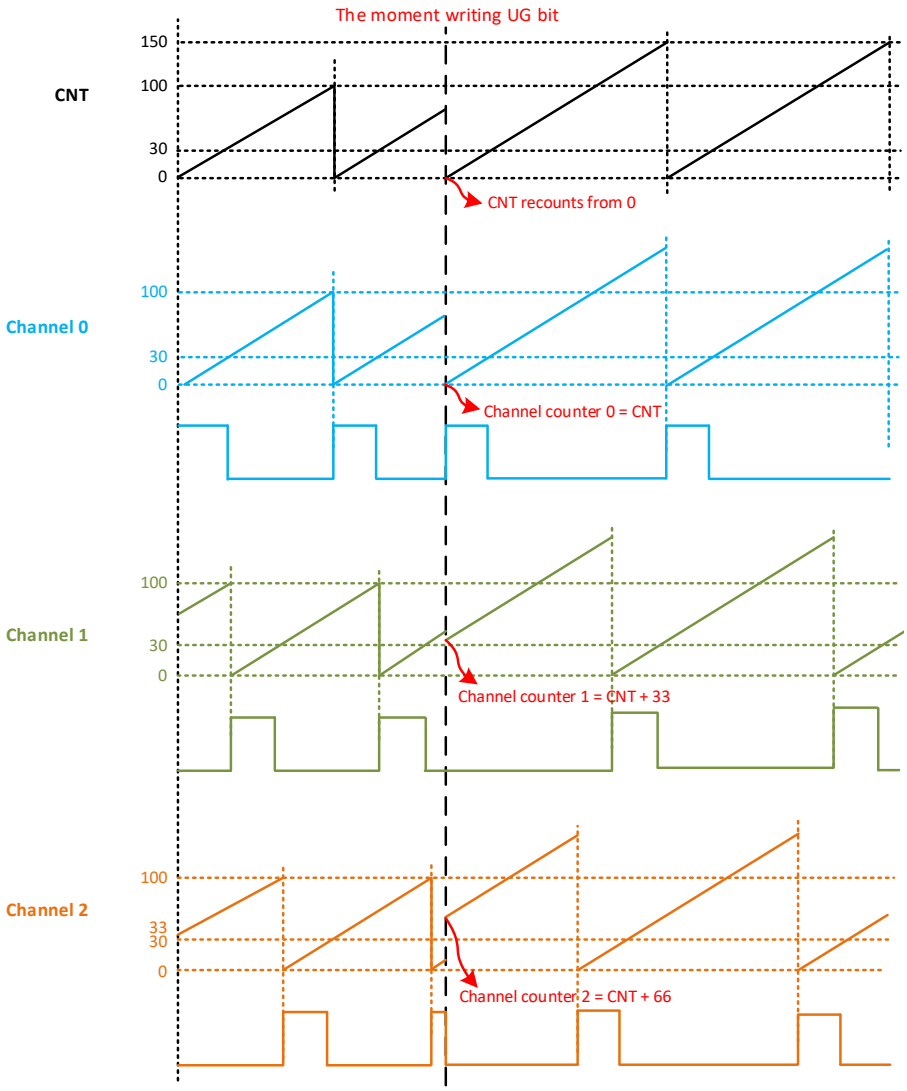


Figure 15-27 Waveform when writing UG bit

15.4 Registers

Table 15-4 Memory map of timers

Type	Name	Base address	Address offset
Basic Timer	TIM0	0x4200_B000	0x000 ~ 0x1FF
	TIM1		0x200 ~ 0x3FF
	TIM2		0x400 ~ 0x5FF
	TIM3		0x600 ~ 0x7FF
	TIM4		0x800 ~ 0x9FF
	TIM5		0xA00 ~ 0xBFF
	TIM6		0xC00 ~ 0xDFF
	TIM7		0xE00 ~ 0xFFF
	TIM10	0x4100_A000	0x400 ~ 0x5FF
	TIM11		0x600 ~ 0x7FF
PWM Timer	TIM8		0x000 ~ 0x1FF
Capture Timer	TIM9		0x200 ~ 0x3FF

All information provided in this document is subject to legal disclaimers.
© REALTEK 2024. All rights reserved.

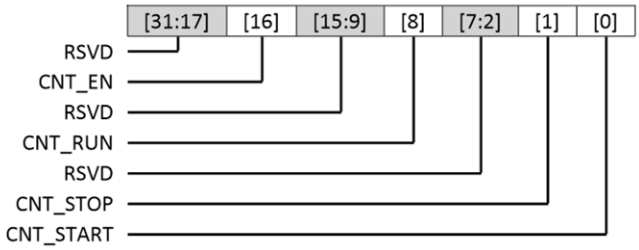
15.4.1 Basic Timer Registers

Name	Address offset	Access	Description
REG_TIM_EN	000h	R/W	TIM Enable Register
REG_TIM_CR	004h	R/W	TIM Control Register
REG_TIM_DIER	008h	R/W	TIM Interrupt Enable Register
REG_TIM_SR	00Ch	R/W	TIM Status Register
REG_TIM_EGR	010h	R/W	TIM Event Generation Register
REG_TIM_CNT	014h	R	TIM Counter Register
REG_TIM_ARR	01Ch	R/W	TIM Auto-Reload Register
REG_TIM_SEC	020h	R/W	TIM Security Control Register
REG_TIM_BASIC_DUMMY	024h	R/W	Timer Basic Dummy Register

15.4.1.1 REG_TIM_EN

- **Name:** TIM Enable Register
- **Size:** 32 bits
- **Address offset:** 000h
- **Read/write access:** R/W

Users can configure and observe the counter operation.



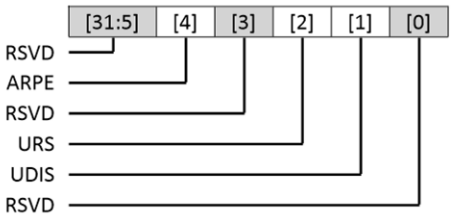
Bit	Symbol	Access	Reset	Description
31:17	RSVD	-	0	Reserved
16	CNT_EN	R	0x0	Counter working status **0**: Counter is stopped **1**: Counter is working
15:9	RSVD	-	0	Reserved
8	CNT_RUN	R	0x0	Counter run status **0**: Counter is disabled **1**: Counter is enabled
7:2	RSVD	-	0	Reserved
1	CNT_STOP	WA0	0x0	Disable the counter Poll CNT_RUN to see the counter status. If CNT_RUN is 0, it means that the counter has been disabled internally.
0	CNT_START	WA0	0x0	Enable the counter Poll CNT_RUN to see the counter status. If CNT_RUN is 1, it means that the counter has been enabled internally.

15.4.1.2 REG_TIM_CR

- **Name:** TIM Control Register
- **Size:** 32 bits
- **Address offset:** 004h
- **Read/write access:** R/W

This register is about the update operation.

All information provided in this document is subject to legal disclaimers.
© REALTEK 2024. All rights reserved.

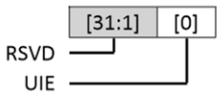


Bit	Symbol	Access	Reset	Description
31:5	RSVD	-	0	Reserved
4	ARPE	R/W	0x0	Enable auto-reload preloading **0**: TIM_ARR register is not buffered. **1**: TIM_ARR register is buffered.
3	RSVD	-	0	Reserved
2	URS	R/W	0x0	Update request source **0**: Counter overflow and setting the UG bit **1**: Counter overflow
1	UDIS	R/W	0x0	Disable UEV Shadow registers keep their values. Otherwise, buffered registers are then loaded with their preload values when UEV happens.
0	RSVD	-	0	Reserved

15.4.1.3 REG_TIM_DIER

- **Name:** TIM Interrupt Enable Register
- **Size:** 32 bits
- **Address offset:** 008h
- **Read/write access:** R/W

This is the interrupt enable register.

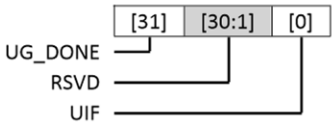


Bit	Symbol	Access	Reset	Description
31:1	RSVD	-	0	Reserved
0	UIE	R/W	0x0	Enable update interrupt

15.4.1.4 REG_TIM_SR

- **Name:** TIM Status Register
- **Size:** 32 bits
- **Address offset:** 00Ch
- **Read/write access:** R/W

This register is about timer interrupt and update status.



Bit	Symbol	Access	Reset	Description
31	UG_DONE	R	0x1	UG operation status This bit is cleared by hardware when the UG bit in the TIMx_EGR register is set. When the UG operation is done, hardware set this bit to '1'. So, software can poll this bit to see the UG operation status.

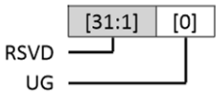
All information provided in this document is subject to legal disclaimers.
© REALTEK 2024. All rights reserved.

30:1	RSVD	-	0	Reserved
0	UIF	RW1C	0x0	Update interrupt flag

15.4.1.5 REG_TIM_EGR

- **Name:** TIM Event Generation Register
- **Size:** 32 bits
- **Address offset:** 010h
- **Read/write access:** R/W

This register is about update generation, and is user defined.

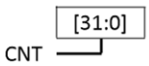


Bit	Symbol	Access	Reset	Description
31:1	RSVD	-	0	Reserved
0	UG	WA0	0x0	Update generation Re-initialize the counter and generate an update of registers. Note that the prescaler counter is cleared too. (anyway the prescaler ratio is not affected)

15.4.1.6 REG_TIM_CNT

- **Name:** TIM Counter Register
- **Size:** 32 bits
- **Address offset:** 014h
- **Read/write access:** R

This is the timer count register.

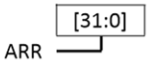


Bit	Symbol	Access	Reset	Description
31:0	CNT	R	0x0	Counter value

15.4.1.7 REG_TIM_ARR

- **Name:** TIM Auto-Reload Register
- **Size:** 32 bits
- **Address offset:** 01Ch
- **Read/write access:** R/W

This is the auto-reload value register.



Bit	Symbol	Access	Reset	Description
31:0	ARR	R/W	0xFFFFFFFF	The value to be loaded in the actual auto-reload register. It can be preloaded by setting ARPE bit in TIM_CR.

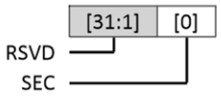
15.4.1.8 REG_TIM_SEC

- **Name:** TIM Security Control Register
- **Size:** 32 bits

All information provided in this document is subject to legal disclaimers.
© REALTEK 2024. All rights reserved.

- **Address offset:** 020h
- **Read/write access:** R/W

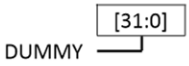
This register is about security status.



Bit	Symbol	Access	Reset	Description
31:1	RSVD	-	0	Reserved
0	SEC	R/W	0x0	Secure timer enable 0: This is a non-secure timer. All its registers except TIMx_SEC can be accessed from both secure and non-secure address. As a result, both secure and non-secure code can access it. 1: This is a secure timer. All its registers can only be accessed from secure address. As a result, only secure code can access it. Non-secure code can't access it through non-secure address.

15.4.1.9 REG_TIM_BASIC_DUMMY

- **Name:** Timer Basic Dummy Register
- **Size:** 32 bits
- **Address offset:** 024h
- **Read/write access:** R/W



Bit	Symbol	Access	Reset	Description
31:0	DUMMY	R/W	ffffh	Dummy register

15.4.2 Capture Timer Registers

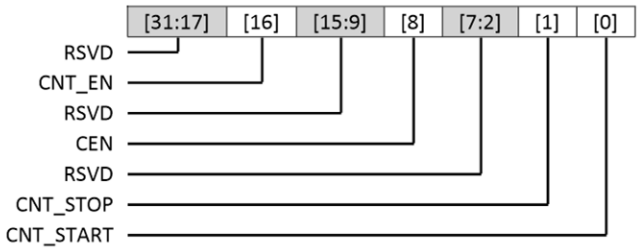
Name	Address offset	Access	Description
REG_TIM_EN	000h	R/W	TIM Enable Register
REG_TIM_CR	004h	R/W	TIM Control Register
REG_TIM_DIER	008h	R/W	TIM Interrupt Enable Register
REG_TIM_SR	00Ch	R/W	TIM Status Register
REG_TIM_EGR	010h	R/W	TIM Event Generation Register
REG_TIM_CNT	014h	R	TIM Counter
REG_TIM_PSC	018h	R/W	TIM Prescaler Register
REG_TIM_ARR	01Ch	R/W	TIM Auto-Reload Register
REG_TIM_CCR0	024h	R/W	TIM Capture Register
REG_TIM_PULSE_DBC	028h	R/W	Timer Pulse DBC Register
REG_TIM_PULSE_DUMMY	02Ch	R/W	Timer Pulse Dummy Register

15.4.2.1 REG_TIM_EN

- **Name:** TIM Enable Register
- **Size:** 32 bits
- **Address offset:** 000h
- **Read/write access:** R/W

Users can configure and observe the counter operation.

All information provided in this document is subject to legal disclaimers.
© REALTEK 2024. All rights reserved.

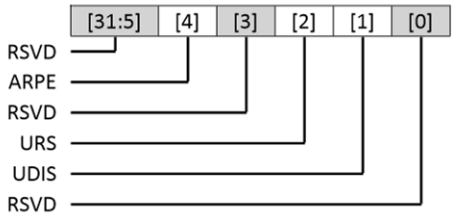


Bit	Symbol	Access	Reset	Description
31:17	RSVD	-	0	Reserved
16	CNT_EN	R	0x0	Counter working status **0**: Counter is stopped **1**: Counter is working
15:9	RSVD	-	0	Reserved
8	CEN	R	0x0	Counter run status **0**: Counter is disabled **1**: Counter is enabled
7:2	RSVD	-	0	Reserved
1	CNT_STOP	WA0	0x0	Disable the counter Poll CNT_RUN to see the counter status. If CNT_RUN is 0, it means that the counter has been disabled internally.
0	CNT_START	WA0	0x0	Enable the counter Poll CNT_RUN to see the counter status. If CNT_RUN is 1, it means that the counter has been enabled internally.

15.4.2.2 REG_TIM_CR

- **Name:** TIM Control Register
- **Size:** 32 bits
- **Address offset:** 004h
- **Read/write access:** R/W

This register is about the update operation.

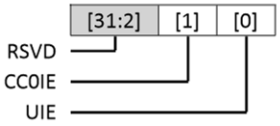


Bit	Symbol	Access	Reset	Description
31:5	RSVD	-	0	Reserved
4	ARPE	R/W	0x0	Enable auto-reload preloading **0**: TIM_ARR register is not buffered **1**: TIM_ARR register is buffered
3	RSVD	-	0	Reserved
2	URS	R/W	0x0	update request source **0**: Counter overflow and setting the UG bit **1**: Counter overflow
1	UDIS	R/W	0x0	Disable UEV Shadow registers keep their values. Otherwise, buffered registers are then loaded with their preload values when UEV happens.
0	RSVD	-	0	Reserved

All information provided in this document is subject to legal disclaimers.
© REALTEK 2024. All rights reserved.

15.4.2.3 REG_TIM_DIER

- **Name:** TIM Interrupt Enable Register
- **Size:** 32 bits
- **Address offset:** 008h
- **Read/write access:** R/W



Bit	Symbol	Access	Reset	Description
31:2	RSVD	-	0	Reserved
1	CC0IE	R/W	0x0	Enable Capture/Compare 0 interrupt
0	UIE	R/W	0x0	Enable update interrupt

15.4.2.4 REG_TIM_SR

- **Name:** TIM Status Register
- **Size:** 32 bits
- **Address offset:** 00Ch
- **Read/write access:** R/W

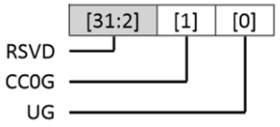


Bit	Symbol	Access	Reset	Description
31	UG_DONE	R	0x1	UG operation status This bit is cleared by hardware when the UG bit in the TIMx_EGR register is set. When the UG operation is done, hardware set this bit to '1'. So, software can poll this bit to see the UG operation status.
30:2	RSVD	-	0	Reserved
1	CC0IF	RW1C	0x0	Capture/Compare 0 interrupt flag CC0 channel configured as pulse mode 0: This bit is set when TRGI is transferred to inactive level from active level. CC0 channel configured as pulse mode 1: This bit is set by hardware when the counter overflows. It is cleared by software.
0	UIF	RW1C	0x0	Update interrupt flag

15.4.2.5 REG_TIM_EGR

- **Name:** TIM Event Generation Register
- **Size:** 32 bits
- **Address offset:** 010h
- **Read/write access:** R/W

This register is about the update operation, and is user-defined.



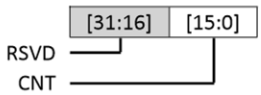
All information provided in this document is subject to legal disclaimers.
© REALTEK 2024. All rights reserved.

Bit	Symbol	Access	Reset	Description
31:2	RSVD	-	0	Reserved
1	CC0G	WA0	0x0	Capture/Compare 0 generation This bit is set by software and is automatically cleared by hardware. A Capture/Compare event is generated on channel 0: the current value of the counter is captured in TIM_CCR0 register. The CC0IF flag is set, the corresponding interrupt or request is sent if enabled.
0	UG	WA0	0x0	Update generation Re-initialize the counter and generate an update of registers. Note that the prescaler counter is cleared too. (anyway the prescaler ratio is not affected)

15.4.2.6 REG_TIM_CNT

- **Name:** TIM Counter
- **Size:** 32 bits
- **Address offset:** 014h
- **Read/write access:** R

This is timer count register.

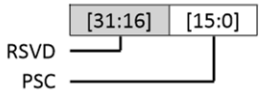


Bit	Symbol	Access	Reset	Description
31:16	RSVD	-	0	Reserved
15:0	CNT	R	0x0	Counter value

15.4.2.7 REG_TIM_PSC

- **Name:** TIM Prescaler Register
- **Size:** 32 bits
- **Address offset:** 018h
- **Read/write access:** R/W

This is prescaler value register.

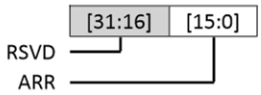


Bit	Symbol	Access	Reset	Description
31:16	RSVD	-	0	Reserved
15:0	PSC	R/W	0x0	Prescaler value PSC contains the value to be loaded in the actual prescaler register at each UEV (including when the counter is cleared through UG bit of TIM_EGR register).

15.4.2.8 REG_TIM_ARR

- **Name:** TIM Auto-Reload Register
- **Size:** 32 bits
- **Address offset:** 01Ch
- **Read/write access:** R/W

This is auto-reload value register.



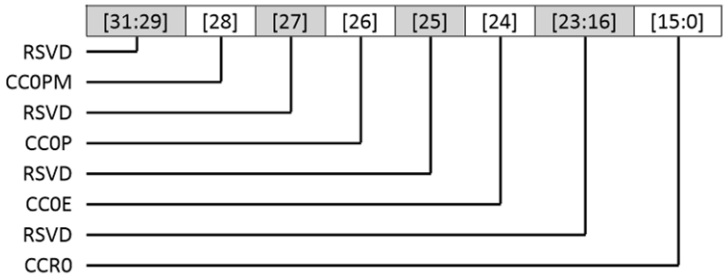
All information provided in this document is subject to legal disclaimers.
© REALTEK 2024. All rights reserved.

Bit	Symbol	Access	Reset	Description
31:16	RSVD	-	0	Reserved
15:0	ARR	R/W	0xFFFF	The value to be loaded in the actual auto-reload register. It can be preloaded by setting ARPE bit in TIM_CR.

15.4.2.9 REG_TIM_CCR0

- **Name:** TIM Capture Register
- **Size:** 32 bits
- **Address offset:** 024h
- **Read/write access:** R/W

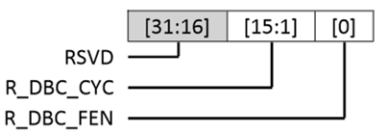
This is the register about input capture configuration and the data of capture.



Bit	Symbol	Access	Reset	Description
31:29	RSVD	-	0	Reserved
28	CC0PM	R/W	0x0	CC0 pulse mode **0**: Pulse mode 0 **1**: Pulse mode 1
27	RSVD	-	0	Reserved
26	CC0P	R/W	0x0	CC0 channel configured as output: **0**: OC0 active high **1**: OC0 active low CC0 channel configured as input: **0**: Positive edge of TRGI is active for capture **1**: Negative edge of TRGI is active for capture
25	RSVD	-	0	Reserved
24	CC0E	R/W	0x0	Enable CC0
23:16	RSVD	-	0	Reserved
15:0	CCR0	R/W/ES	0x0	Capture/Compare 0 value CC0 channel configured as pulse mode 0: the pulse width of TRGI CC0 channel configured as pulse mode 1: the pulse number of TRGI when the counter counts from 0 to ARR

15.4.2.10 REG_TIM_PULSE_DBC

- **Name:** Timer Pulse DBC Register
- **Size:** 32 bits
- **Address offset:** 028h
- **Read/write access:** R/W



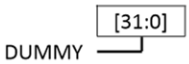
Bit	Symbol	Access	Reset	Description
31:16	RSVD	-	0	Reserved
15:1	R_DBC_CYC	R/W	0x0	DBC cycle

All information provided in this document is subject to legal disclaimers.
© REALTEK 2024. All rights reserved.

0	R_DBC_FEN	R/W	0x0	Enable debounce for pulse 0: Disable 1: Enable

15.4.2.11 REG_TIM_PULSE_DUMMY

- **Name:** Timer Pulse Dummy Register
- **Size:** 32 bits
- **Address offset:** 02Ch
- **Read/write access:** R/W



Bit	Symbol	Access	Reset	Description
31:0	DUMMY	R/W	ffffh	Dummy register

15.4.3 PWM Timer Registers

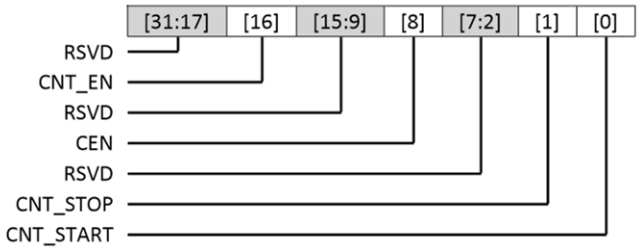
Name	Address offset	Access	Description
REG_TIM_EN	000h	R/W	TIM Enable Register
REG_TIM_CR	004h	R/W	TIM Control Register
REG_TIM_DIER	008h	R/W	TIM Interrupt Enable Register
REG_TIM_SR	00Ch	R/W	TIM Status Register
REG_TIM_EGR	010h	R/W	TIM Event Generation Register
REG_TIM_CNT	014h	R	TIM Counter
REG_TIM_PSC	018h	R/W	TIM Prescaler Register
REG_TIM_ARR	01Ch	R/W	TIM Auto-Reload Register
REG_TIM_CCR0	024h	R/W	TIM Capture/Compare Register 0
REG_TIM_CCR1	028h	R/W	TIM Capture/Compare Register 1
REG_TIM_CCR2	02Ch	R/W	TIM Capture/Compare Register 2
REG_TIM_CCR3	030h	R/W	TIM Capture/Compare Register 3
REG_TIM_CCR4	034h	R/W	TIM Capture/Compare Register 4
REG_TIM_CCR5	038h	R/W	TIM Capture/Compare Register 5
REG_TIM_CCR6	03Ch	R/W	TIM Capture/Compare Register 6
REG_TIM_CCR7	040h	R/W	TIM Capture/Compare Register 7
REG_TIM_PSYNC0	044h	R/W	TIM Phase Synchronization Register 0
REG_TIM_PSYNC1	048h	R/W	TIM Phase Synchronization Register 1
REG_TIM_PSYNC2	04Ch	R/W	TIM Phase Synchronization Register 2
REG_TIM_PSYNC3	050h	R/W	TIM Phase Synchronization Register 3
REG_TIM_PSYNC4	054h	R/W	TIM Phase Synchronization Register 4
REG_TIM_PSYNC5	058h	R/W	TIM Phase Synchronization Register 5
REG_TIM_PSYNC6	05Ch	R/W	TIM Phase Synchronization Register 6
REG_TIM_PSYNC7	060h	R/W	TIM Phase Synchronization Register 7
REG_TIM_PHASECNT0	064h	R	TIM Phase Counter Register 0
REG_TIM_PHASECNT1	068h	R	TIM Phase Counter Register 1
REG_TIM_PHASECNT2	06Ch	R	TIM Phase Counter Register 2
REG_TIM_PHASECNT3	070h	R	TIM Phase Counter Register 3
REG_TIM_PHASECNT4	074h	R	TIM Phase Counter Register 4
REG_TIM_PHASECNT5	078h	R	TIM Phase Counter Register 5
REG_TIM_PHASECNT6	07Ch	R	TIM Phase Counter Register 6
REG_TIM_PHASECNT7	080h	R	TIM Phase Counter Register 7
REG_TIM_PWM_DUMMY	084h	R/W	Timer PWM Dummy Register

All information provided in this document is subject to legal disclaimers.
© REALTEK 2024. All rights reserved.

15.4.3.1 REG_TIM_EN

- **Name:** TIM Enable Register
- **Size:** 32 bits
- **Address offset:** 000h
- **Read/write access:** R/W

Users can configure and observe the counter operation.

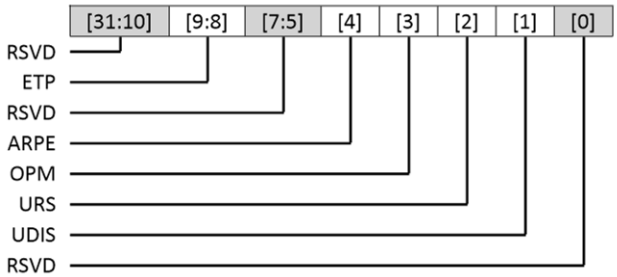


Bit	Symbol	Access	Reset	Description
31:17	RSVD	-	0	Reserved
16	CNT_EN	R	0x0	Counter working status **0**: Counter is stopped **1**: Counter is working
15:9	RSVD	-	0	Reserved
8	CEN	R	0x0	Counter run status **0**: Counter is disabled **1**: Counter is enabled
7:2	RSVD	-	0	Reserved
1	CNT_STOP	WA0	0x0	Disable the counter. Poll CNT_RUN to see the counter status. If CNT_RUN is 0, it means that the counter has been disabled internally.
0	CNT_START	WA0	0x0	Enable the counter. Poll CNT_RUN to see the counter status. If CNT_RUN is 1, it means that the counter has been enabled internally.

15.4.3.2 REG_TIM_CR

- **Name:** TIM Control Register
- **Size:** 32 bits
- **Address offset:** 004h
- **Read/write access:** R/W

This is the register about the update operation and the work mode configuration.



Bit	Symbol	Access	Reset	Description
31:10	RSVD	-	0	Reserved
9:8	ETP	R/W	0x0	External trigger polarity (TRGI) **0**: Positive edge is active **1**: Negative edge is active

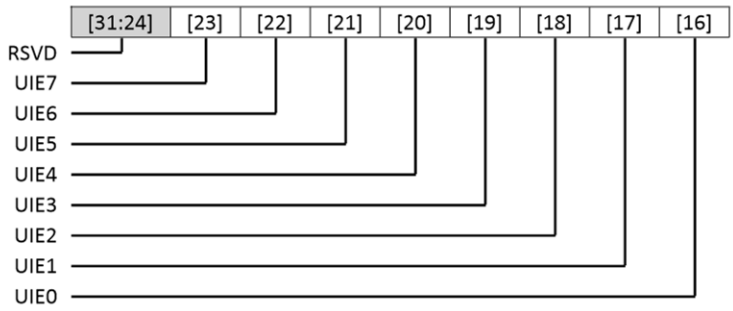
All information provided in this document is subject to legal disclaimers.
© REALTEK 2024. All rights reserved.

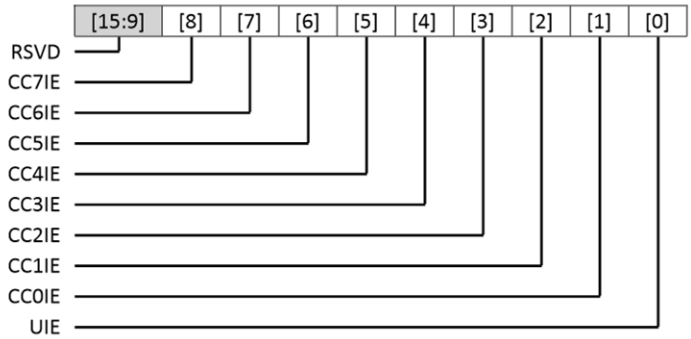
				2 or 3: Both edge is active NOTE *These bits are only valid in one-pulse mode.*
7:5	RSVD	-	0	Reserved
4	ARPE	R/W	0x0	Enable auto-reload preloading **0**: TIM_ARR register is not buffered **1**: TIM_ARR register is buffered
3	OPM	R/W	0x0	One-pulse mode **0**: Counter is not stopped at UEV **1**: Counter stops counting at next UEV
2	URS	R/W	0x0	Update request source **0**: Counter overflow and setting the UG bit **1**: Counter overflow
1	UDIS	R/W	0x0	Disable UEV Shadow registers keep their values. Otherwise, buffered registers are then loaded with their preload values when UEV happens.
0	RSVD	-	0	Reserved

15.4.3.3 REG_TIM_DIER

- **Name:** TIM Interrupt Enable Register
- **Size:** 32 bits
- **Address offset:** 008h
- **Read/write access:** R/W

This is the interrupt enable register.





Bit	Symbol	Access	Reset	Description
31:24	RSVD	-	0	Reserved
23	UIE7	R/W	0x0	Enable channel 7 update interrupt
22	UIE6	R/W	0x0	Enable channel 6 update interrupt
21	UIE5	R/W	0x0	Enable channel 5 update interrupt
20	UIE4	R/W	0x0	Enable channel 4 update interrupt
19	UIE3	R/W	0x0	Enable channel 3 update interrupt
18	UIE2	R/W	0x0	Enable channel 2 update interrupt

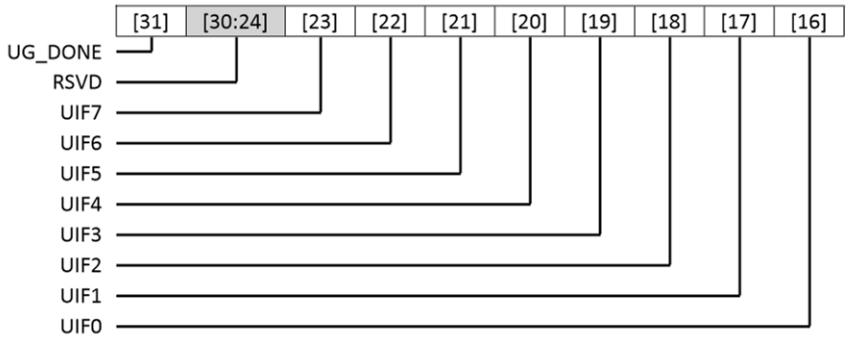
All information provided in this document is subject to legal disclaimers.
© REALTEK 2024. All rights reserved.

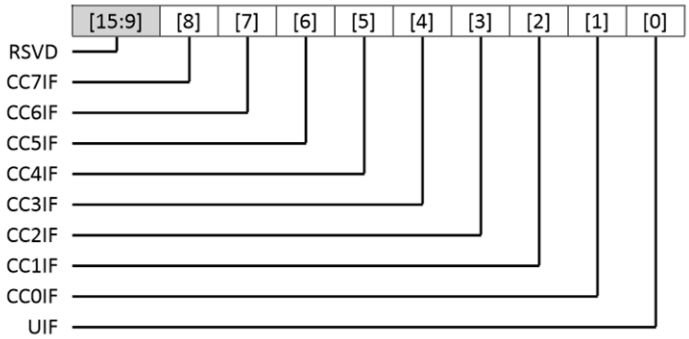
17	UIE1	R/W	0x0	Enable channel 1 update interrupt
16	UIE0	R/W	0x0	Enable channel 0 update interrupt
15:9	RSVD	-	0	Reserved
8	CC7IE	R/W	0x0	Enable Capture/Compare 7 interrupt
7	CC6IE	R/W	0x0	Enable Capture/Compare 6 interrupt
6	CC5IE	R/W	0x0	Enable Capture/Compare 5 interrupt
5	CC4IE	R/W	0x0	Enable Capture/Compare 4 interrupt
4	CC3IE	R/W	0x0	Enable Capture/Compare 3 interrupt
3	CC2IE	R/W	0x0	Enable Capture/Compare 2 interrupt
2	CC1IE	R/W	0x0	Enable Capture/Compare 1 interrupt
1	CC0IE	R/W	0x0	Enable Capture/Compare 0 interrupt
0	UIE	R/W	0x0	Enable counter update interrupt

15.4.3.4 REG_TIM_SR

- **Name:** TIM Status Register
- **Size:** 32 bits
- **Address offset:** 00Ch
- **Read/write access:** R/W

This is the register about timer interrupt and update status.





Bit	Symbol	Access	Reset	Description
31	UG_DONE	R	0x1	UG operation status This bit is cleared by hardware when the UG bit in the TIMx_EGR register is set. When the UG operation is done, hardware sets this bit to '1'. So, software can poll this bit to see the UG operation status.
30:24	RSVD	-	0	Reserved
23	UIF7	RW1C	0x0	Refer to UIF0 description
22	UIF6	RW1C	0x0	Refer to UIF0 description
21	UIF5	RW1C	0x0	Refer to UIF0 description
20	UIF4	RW1C	0x0	Refer to UIF0 description
19	UIF3	RW1C	0x0	Refer to UIF0 description
18	UIF2	RW1C	0x0	Refer to UIF0 description

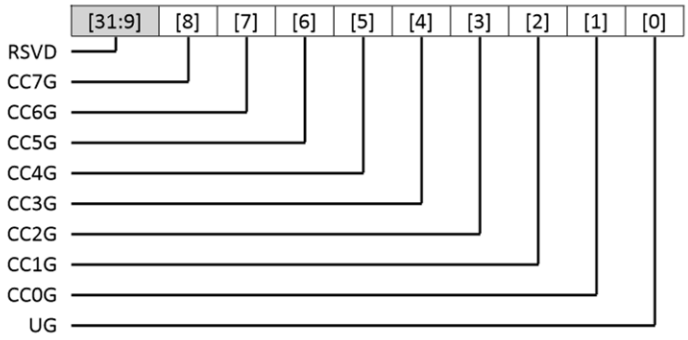
All information provided in this document is subject to legal disclaimers.
© REALTEK 2024. All rights reserved.

17	UIF1	RW1C	0x0	Refer to UIF0 description
16	UIF0	RW1C	0x0	Update interrupt flag of channel 0 This flag is set by hardware when the UIE0 bit in the TIMx_EGR register is set, and update interrupt of channel 0 happens.
15:9	RSVD	-	0	Reserved
8	CC7IF	RW1C	0x0	Refer to CC0IF description
7	CC6IF	RW1C	0x0	Refer to CC0IF description
6	CC5IF	RW1C	0x0	Refer to CC0IF description
5	CC4IF	RW1C	0x0	Refer to CC0IF description
4	CC3IF	RW1C	0x0	Refer to CC0IF description
3	CC2IF	RW1C	0x0	Refer to CC0IF description
2	CC1IF	RW1C	0x0	Refer to CC0IF description
1	CC0IF	RW1C	0x0	Capture/Compare 0 interrupt flag Channel CC0 configured as output: This flag is set by hardware when the counter TIM_CNT value matches the value of TIM_CCR0 register. It is cleared by software. Channel CC0 configured as input: This flag is set by hardware and the counter value has been captured in CCR0 field of TIM_CCR0 register when an active edge has been detected. It is cleared by software.
0	UIF	RW1C	0x0	Update interrupt flag

15.4.3.5 REG_TIM_EGR

- **Name:** TIM Event Generation Register
- **Size:** 32 bits
- **Address offset:** 010h
- **Read/write access:** R/W

This is the register about update generation, and is user-defined.



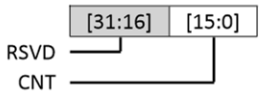
Bit	Symbol	Access	Reset	Description
31:9	RSVD	-	0	Reserved
8	CC7G	WA0	0x0	Refer to CC0G description
7	CC6G	WA0	0x0	Refer to CC0G description
6	CC5G	WA0	0x0	Refer to CC0G description
5	CC4G	WA0	0x0	Refer to CC0G description
4	CC3G	WA0	0x0	Refer to CC0G description
3	CC2G	WA0	0x0	Refer to CC0G description
2	CC1G	WA0	0x0	Refer to CC0G description
1	CC0G	WA0	0x0	Capture/Compare 0 generation This bit is set by software and is automatically cleared by hardware. Channel CC0 configured as output: The CC0IF flag is set, the corresponding interrupt or request is sent if enabled. Channel CC0 configured as input: The current value of counter is capture in CCR0 field of TIM_CCR0 register. The CC0IF flag is set, the corresponding interrupt or request is sent if enabled.
0	UG	WA0	0x0	Update generation Re-initialize the counter and generate an update of registers. Note that the prescaler counter is cleared too. (anyway the prescaler ratio is not affected)

All information provided in this document is subject to legal disclaimers.
© REALTEK 2024. All rights reserved.

15.4.3.6 REG_TIM_CNT

- **Name:** TIM Counter
- **Size:** 32 bits
- **Address offset:** 014h
- **Read/write access:** R

This is the register about timer count.

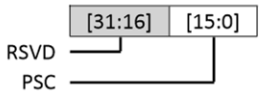


Bit	Symbol	Access	Reset	Description
31:16	RSVD	-	0	Reserved
15:0	CNT	R	0x0	Counter value

15.4.3.7 REG_TIM_PSC

- **Name:** TIM Prescaler Register
- **Size:** 32 bits
- **Address offset:** 018h
- **Read/write access:** R/W

This is the register about prescaler value.

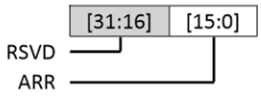


Bit	Symbol	Access	Reset	Description
31:16	RSVD	-	0	Reserved
15:0	PSC	R/W	0x0	Prescaler value PSC contains the value to be loaded in the actual prescaler register at each UEV (including when the counter is cleared through UG bit of TIM_EGR register).

15.4.3.8 REG_TIM_ARR

- **Name:** TIM Auto-Reload Register
- **Size:** 32 bits
- **Address offset:** 01Ch
- **Read/write access:** R/W

This is the register about auto-reload value.



Bit	Symbol	Access	Reset	Description
31:16	RSVD	-	0	Reserved
15:0	ARR	R/W	0xFFFF	The value to be loaded in the actual auto-reload register. It can be preloaded by setting ARPE bit in the TIM_CR register.

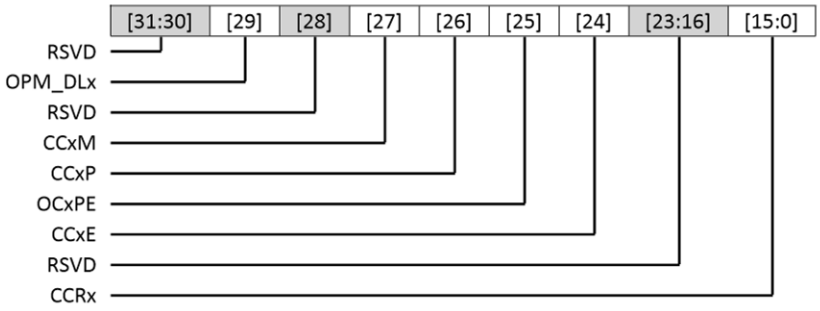
15.4.3.9 REG_TIM_CCRx

- **Name:** TIM Capture/Compare Register x (x={0, 1, 2, 3, 4, 5, 6, 7})
- **Size:** 32 bits

All information provided in this document is subject to legal disclaimers.
© REALTEK 2024. All rights reserved.

- **Address offset:**
  - **REG_TIM_CCR0:** 024h
  - **REG_TIM_CCR1:** 028h
  - **REG_TIM_CCR2:** 02Ch
  - **REG_TIM_CCR3:** 030h
  - **REG_TIM_CCR4:** 034h
  - **REG_TIM_CCR5:** 038h
  - **REG_TIM_CCR6:** 03Ch
  - **REG_TIM_CCR7:** 040h
- **Read/write access:** R/W

This is the register about the configuration of input capture and output compare work mode.



Bit	Symbol	Access	Reset	Description
31:30	RSVD	-	0	Reserved
29	OPM_DLx	R/W	0x0	One-pulse mode default level 0: Default level is 0 1: Default level is 1 This bit takes effect only in PWM output one-pulse mode.
28	RSVD	-	0	Reserved
27	CCxM	R/W	0x0	CCx working mode **0**: PWM mode **1**: Input capture mode
26	CCxP	R/W	0x0	CCx channel configured as output: **0**: OCx active high **1**: OCx active low CCx channel configured as input: **0**: Positive edge of TRGI is active for capture **1**: Negative edge of TRGI is active for capture
25	OCxPE	R/W	0x0	Enable output capture x preload **0**: Disable preload register on CCRx. CCRx can be written at any time, the new value is taken into account immediately. **1**: Enable preload register on CCRx. Read/Write operations access the preload register. CCRx preload value is loaded in the active register at each UEV.
24	CCxE	R/W	0x0	Enable CCx
23:16	RSVD	-	0	Reserved
15:0	CCRx	R/W/ES	0x0	Capture/Compare x value CCx channel configured as output: CCRx is the value to be loaded in the actual capture/compare x register (preload value). It is loaded permanently if the preload feature is not selected in the OCxPE bit. Else the preload value is copied in the active capture/compare x register when an UEV occurs. The active capture/compare x register contains the value to be compared to the counter TIM_CNT and signalled on OCx output. CCx channel configured as pulse mode 1: CCRx is the counter value transferred by the last input capture event (TRGI)

15.4.3.10 REG_TIM_PSYNCx

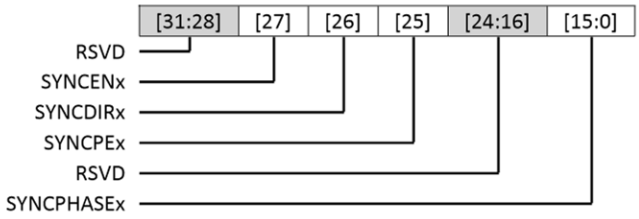
- **Name:** TIM Phase Synchronization Register x (x={0, 1, 2, 3, 4, 5, 6, 7})
- **Size:** 32 bits
- **Address offset:**

All information provided in this document is subject to legal disclaimers.
© REALTEK 2024. All rights reserved.

- **REG_TIM_PSYNC0:** 044h
- **REG_TIM_PSYNC1:** 048h
- **REG_TIM_PSYNC2:** 04Ch
- **REG_TIM_PSYNC3:** 050h
- **REG_TIM_PSYNC4:** 054h
- **REG_TIM_PSYNC5:** 058h
- **REG_TIM_PSYNC6:** 05Ch
- **REG_TIM_PSYNC7:** 060h

- **Read/write access:** R/W

This is the register about the configuration of phase deviation for PWM output.

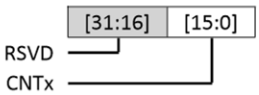


Bit	Symbol	Access	Reset	Description
31:28	RSVD	-	0	Reserved
27	SYNCENx	R/W	0x0	Enable phase sync function in channel x. Phase sync takes effect only in output PWM mode.
26	SYNCDIRx	R/W	0x0	Phase sync direction to the reference counter 0: Delay to the reference counter 1: Ahead to the reference counter
25	SYNCPEx	R/W	0x0	Enable sync phase parameter preload **0**: Disable preload register on PSYNCx. PSYNCx can be written at any time, the new value is taken into account immediately. **1**: Enable preload register on PSYNCx. Read/Write operations access the preload register. PSYNCx preload value is loaded in the active register at each UEV.
24:16	RSVD	-	0	Reserved
15:0	SYNCPHASEx	R/W	0x0	The deviation value of channel x to the reference counter

15.4.3.11 REG_TIM_PHASECNTx

- **Name:** TIM Phase Counter Register x (x={0, 1, 2, 3, 4, 5, 6, 7})
- **Size:** 32 bits
- **Address offset:**
  - **REG_TIM_PHASECNT0:** 064h
  - **REG_TIM_PHASECNT1:** 068h
  - **REG_TIM_PHASECNT2:** 06Ch
  - **REG_TIM_PHASECNT3:** 070h
  - **REG_TIM_PHASECNT4:** 074h
  - **REG_TIM_PHASECNT5:** 078h
  - **REG_TIM_PHASECNT6:** 07Ch
  - **REG_TIM_PHASECNT7:** 080h
- **Read/write access:** R

This is the register about phase counter.

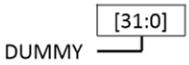


Bit	Symbol	Access	Reset	Description
31:16	RSVD	-	0	Reserved
15:0	CNTx	R	0x0	Phase counter value of channel x

All information provided in this document is subject to legal disclaimers.
© REALTEK 2024. All rights reserved.

15.4.3.12 REG_TIM_PWM_DUMMY

- **Name:** Timer PWM Dummy Register
- **Size:** 32 bits
- **Address offset:** 084h
- **Read/write access:** R/W



Bit	Symbol	Access	Reset	Description
31:0	DUMMY	R/W	ffffh	Dummy register

All information provided in this document is subject to legal disclaimers.
© REALTEK 2024. All rights reserved.

16 Debug Timer

Debug timer is a common timestamp for all debug messages originating from all on-die processors and processor execution domains (application, kernel, and firmware). It also includes a lock-free increment counter.

16.1 Features

- Simple 32-bit wrap timer
- A lock-free counter

16.2 Functional Description

The debug timer has two types of clock input: XTAL and internal 32K.

- When selecting XTAL with an internal divider, XTAL can generate a fixed 1us time interval between each time increments.
- When selecting 32K clock, the counter increases 31us every 30.5us.

The counter can be enabled and disabled, by default it is enabled. The counter wraps around to zero and continues to count once it reaches 32'hFFFFFFFF. A write to timestamp will set the current value of the timestamp, however, it must continue to increment at the base of the new setting value if the writing happens when the counter is active.

The XTAL clock may be gated in sleep mode. If you select XTAL as the clock source in sleep mode, the debug timer will stop counting, and all the registers will be maintained. The counter will resume the increment immediately after XTAL resumes. You can select 32K as the clock source in sleep mode; however, the counter itself needs 220us to do switch clock before continuing counting. In this period, writing to this IP is not allowed. All the registers will be reset to the initial values after wakeup from deep-sleep mode.

The lock-free counter is a read trigger increase counter, can be used to debug timing issues in multicore SoC. Arbitration is implemented at the bus level. The counter has two read windows: read will trigger the counter increases one if reading from DBGT _ATOM_INC, and will not increase if reading from DBGT_ATOM.

The timestamp and atom should be reset to zero on the rising edge of the ENABLE bit (DBGT_CTRL[0]).

16.3 Registers

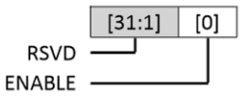
The following table lists the memory map of the debug timer registers, and the base address is 0x4200_9000.

Name	Address offset	Access	Description
REG_DBGT_CTRL	000h	R/W	Debug Timer Control Register
REG_DBGT_CRV	004h	R/W	Debug Timer Count Register
REG_DBGT_ATOM_INC	008h	R	Debug Timer Atom Read Increase Register
REG_DBGT_ATOM	00Ch	R/W	Debug Timer Atom Register
REG_DBGT_SCRATCH	010h	R/W	Debug Timer Scratch Register
REG_DBGTIM_DUMMY	014h	R/W	Debug Timer Dummy Register

16.3.1 REG_DBGT_CTRL

- **Name:** Debug Timer Control Register
- **Size:** 32 bits
- **Address offset:** 000h
- **Read/write access:** R/W

This register contains the configuration of debug timer's clock and enable signal.



All information provided in this document is subject to legal disclaimers.
© REALTEK 2024. All rights reserved.

Bit	Symbol	Access	Reset	Description
31:1	RSVD	-	0	Reserved
0	ENABLE	R/W	1h	Debug timer enable. 1: The counter is enabled. 0: The counter is disabled.

16.3.2 REG_DBGT_CRV

- **Name:** Debug Timer Count Register
- **Size:** 32 bits
- **Address offset:** 004h
- **Read/write access:** R/W



Bit	Symbol	Access	Reset	Description
31:0	DBGT_CRV	R/WOP	0h	Continue to count from 0 once it's reached 32'hFFFFFFFF.

16.3.3 REG_DBGT_ATOM_INC

- **Name:** Debug Timer Atom Read Increase Register
- **Size:** 32 bits
- **Address offset:** 008h
- **Read/write access:** R

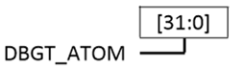
Atom count adds 1.



Bit	Symbol	Access	Reset	Description
31:0	DBGT_ATOM_INC	RP	0h	Each read will increase 1 and return to the increased value to bus.

16.3.4 REG_DBGT_ATOM

- **Name:** Debug Timer Atom Register
- **Size:** 32 bits
- **Address offset:** 00Ch
- **Read/write access:** R/W



Bit	Symbol	Access	Reset	Description
31:0	DBGT_ATOM	R/WOP	0h	Return the last increased value of atom counter.

16.3.5 REG_DBGT_SCRATCH

- **Name:** Debug Timer Scratch Register
- **Size:** 32 bits
- **Address offset:** 010h
- **Read/write access:** R/W

This register is user-defined.

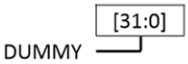
All information provided in this document is subject to legal disclaimers.
© REALTEK 2024. All rights reserved.



Bit	Symbol	Access	Reset	Description
31:0	DBGT_SCRATCH	R/W	0h	Reserved

16.3.6 REG_DBGTIM_DUMMY

- **Name:** Debug Timer Dummy Register
- **Size:** 32 bits
- **Address offset:** 014h
- **Read/write access:** R/W



Bit	Symbol	Access	Reset	Description
31:0	DUMMY	R/W	ffffh	Dummy register

All information provided in this document is subject to legal disclaimers.
© REALTEK 2024. All rights reserved.

17 Real-time Clock (RTC)

17.1 Product Overview

17.1.1 Introduction

The real-time clock (RTC) is an independent binary coded decimal (BCD) timer/counter. One 32-bit register contains the seconds, minutes, hours (12- or 24-hour format) expressed in BCD format. One 32-bit register contains the days expressed in binary format. One 8-bit register contains the years expressed in binary format.

Daylight saving time compensation can be performed.

Additional two 32-bit registers contain the programmable alarm seconds, minutes, hours and days. Year information is saved in another register.

A digital calibration feature is available to compensate for some deviation.

Periodic auto-wakeup function can wakeup systems in periodically with programmable period.

After backup domain reset, all RTC registers are protected against possible parasitic write accesses.

As long as the supply voltage remains in the operating range, the RTC never stops, regardless of the device status which is running mode, low power mode or under reset.

When the system boots up, the clock source of RTC timer is from SDM32K. And RTC module's deviation is less than 2s in 24-hour.

17.1.2 Features

- Time with seconds, minutes, hours (12- or 24-hour format) days and years
- Daylight saving compensation programmable by software
- One programmable alarm with interrupt function. The alarm can be triggered by any combination of the time fields.
- Maskable interrupt/event:
  - Alarm
  - Day threshold
  - Wakeup timer
- Digital calibration circuit
- Register write protection
- Periodic auto-wakeup

17.2 Functional Description

17.2.1 Clock and Prescaler

A programmable prescaler stage generates a 1Hz clock which is used to update the calendar. To minimize power consumption, the prescaler is split into 2 programmable prescalers.

- A 9-bit asynchronous prescaler configured through the PREDIV_A bits of the RTC_PRER register.
- A 9-bit synchronous prescaler configured through the PREDIV_S bits of the RTC_PRER register.

NOTE

It is recommended to configure the asynchronous prescaler to a high value to minimize consumption.

Default, the asynchronous prescaler division factor is set to 128, and the synchronous division factor is set to 256, to obtain an internal clock frequency of 1Hz (clk_spre) with 32.768kHz as RTCCLK.

fclk_apre is given by the following formula:

$$f_{clk_apre} = \frac{f_{RTCCLK}}{\text{PREDIV_A} + 1}$$

fclk_spre is given by the following formula:

All information provided in this document is subject to legal disclaimers.
© REALTEK 2024. All rights reserved.

$$f_{clk_spre} = \frac{f_{clk_apre}}{\text{PREDIV_S} + 1}$$

17.2.2 Programmable Alarm

The RTC unit provides one programmable alarm.

The programmable alarm function is enabled through the ALME bit in the RTC_CR register. The ALMF is set to '1' if the calendar seconds, minutes, hours or days match the values programmed in the alarm registers RTC_ALMR1L and RTL_ALMR1H. Each calendar field can be independently selected through the MSKx bits. The alarm interrupt is enabled through the ALMIE bit in the RTC_CR register.

Alarm (if enabled by the OSEL[1:0] bits in RTC_CR) can be routed to the RTC_OUT output except when system is in deepsleep mode. The alarm output is a pulse which width is 1/RTCCLK.

17.2.3 Write Protection

After RTC domain reset, all the RTC registers are write-protected. Writing to the RTC registers is enabled by writing a key into the RTC_WPR register.

The following steps are required to unlock the write protection on all the RTC registers except for ALMF in RTC_ISR.
(1) Write ‘0xCA’ into the RTC_WPR register.
(2) Write ‘0x53’ into the RTC_WPR register.

Writing a wrong key reactivates the write protection. The protection mechanism is not affected by system reset.

17.2.4 Digital Calibration

The digital calibration can be used to compensate RTCCLK inaccuracy by adding (positive calibration) or masking (negative calibration) clock cycles at the output of the asynchronous prescaler (clk_apre).

Positive and negative calibrations are selected by setting the DCS bit to ‘0’ and ‘1’ in RTC_CALIBR register, respectively.
- When positive calibration is enabled (DCS = ‘0’), DC clk_apre cycle is added every (CALP+1) minutes, which causes the calendar updated sooner, thereby adjusting the effective RTC frequency to be a bit higher.
- When negative calibration is enabled (DCS = ‘1’), DC clk_apre cycle is removed every (CALP +1) minutes, which causes the calendar updated later, thereby adjusting the effective RTC frequency to be a bit lower.

DC and CALP can be configured through the RTC_CALIBR register. DC must be less than PREDIV_S in the RTC_PRER register.

The calibration parameter can be configured on-the-fly. Calibrating resolution is determined by the frequency of clk_apre and the calibration period. Formula is as follows:

$$\text{resolution} = \frac{period_clk_apre}{calp}$$

The example is illustrated in *Table 17-1*.

Table 17-1 Example of calibrating resolution

CALP	clk_apre (128Hz)	clk_apre (256Hz)	clk_apre (512Hz)
1 min.	130.2ppm	65.1ppm	32.55ppm
2 min.	65.1ppm	32.55ppm	16.27ppm
4 min.	32.55ppm	16.27ppm	8.14ppm
8 min.	16.27ppm	8.14ppm	4.07ppm

Re-calibration on-the-fly:

The calibration register (RTC_CALIBR) can be updated on the fly while INITF=0, by following the following steps:
- Poll the RECALPF (re-calibration pending flag).
- If RECALPF is set to 0, write a new value to RTC_CALIBR if necessary, RECALPF is then automatically set to 1.
- Within three clk_apre cycles after the write operation to RTC_CALIBR, the new calibration settings take effect.

All information provided in this document is subject to legal disclaimers.
© REALTEK 2024. All rights reserved.

17.2.5 Day Threshold Program

The RTC provides day interrupt for users. When wants to use day interrupt, DAY_THRES[8:0] must be programmed first in RTC_CR and then set the DOVTHIE bit in RTC_CR to enable day over threshold interrupt.

17.2.6 Daylight Saving

The daylight saving time management is performed through bits SUBIH, ADD1H, and BKP of the RTC_CR register.

Using SUB1H or ADD1H, the software can subtract or add one hour to the calendar in one single operation without going through the initialization procedure. In addition, the software can use the BKP bit to memorize this operation. It is recommended not to change the hour during the calendar hour increment as it could mask the incrimination of the calendar hour. (HW can handle this condition, if this condition occurs, HW will delay the SUB1H or ADD1H operation for 1 second. When calendar hour is 0 of 24-hour system, SUB1H is not supported.

17.2.7 Periodic Auto-wakeup

The wakeup function is enabled by setting the WUTE bit in the RTC_CR register.

The clock source of wakeup timer is ck_spre (usually 1Hz internal clock). When ck_spre frequency is 1Hz, this allows to achieve a wakeup time from 1s to around 36hours with one-second resolution. 3 to 4 wakeup timer cycles after the initialization sequence is complete (refer to *17.5.4*), the timer starts counting down. The down-counting remains active in low-power modes. In addition, when it reaches 0, the WUTF flag is set in the RTC_ISR register, and the wakeup counter is automatically reloaded with its reload value (RTC_WUTR register value).

The WUTF flag must then be cleared by software.

When the periodic wakeup interrupt is enabled by setting the WUTIE bit in the RTC_CR register, it can exit the device from low-power modes.

System resets, as well as low-power modes have no influence on the wakeup timer.

Once wakeup timer is initialized and enabled, wakeup auto-reload value can be changed according to steps in *17.5.4*.

Each time wakeup timer is enabled, 3 to 4 wakeup timer cycles (usually 3 to 4 seconds) are needed before wakeup timer works and counting actually begins. Because the enable command is generated in APB domain, it costs 2 RTC clock to sync to RTC domain, another 2 wakeup timer cycles are needed to sync to wakeup timer domain and one more wakeup cycle is needed before starting counting. After sync process of WUTE bit is done, WUTRSF flag bit is set for users to check.

17.3 Read and Write Timing

The write timing of RTC is shown in *Figure 17-1*. Take the example of writing register RTC_TR. First write INIT bit=1 in RTC_ISR register, enter initialize mode, steady_n_o_f (low active) and steady_o_f (high active) are init mode flag. In the initialization phase, RTC_TR register is written, and then INIT BIT is configured to exit the initialization phase. exit_initmode_ps_rtc is the exit initialization flag signal that the APB domain synchronizes to the RTC domain. When it is raised, the APB value is loaded to the RTC, as shown at moment 1. Init_mode_n_rtc is the signal that steady_n_o_f synchronizes to the RTC domain, and then combines with rtc_rst_n to form init_rst_n_rtc.When counter_init is low, the value on the RTC is loaded into the clk_spre field, as shown at moment 2, when the value in the RTC field is loaded into the clk_spre field.

All information provided in this document is subject to legal disclaimers.
© REALTEK 2024. All rights reserved.

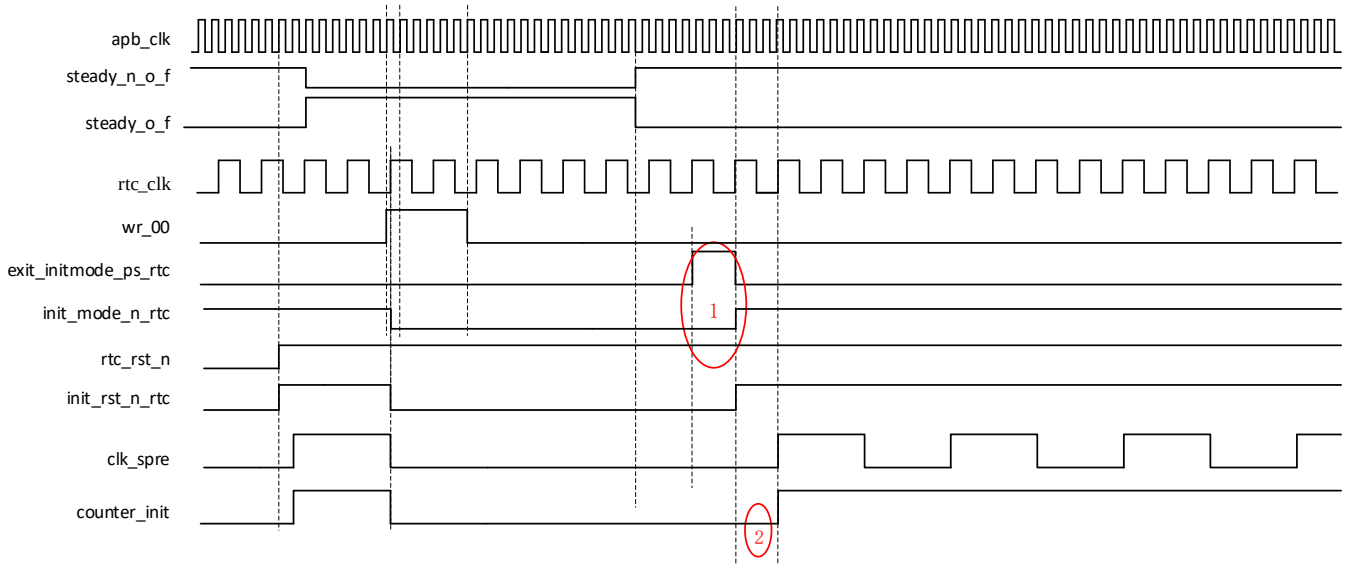


Figure 17-1 RTC write timing

The read timing of RTC is shown in *Figure 17-2*. The RTC domain clock collects the rising edge of clk_spre clock and synchronizes the acquired rising edge to the APB domain. When rd_time_ps_rtc is raised in the RTC domain, the values in the clk_spre domain are loaded into the RTC domain. When rd_time_ps_apb is raised in the APB domain, the values in the RTC domain are loaded into the APB domain. The read value is determined by the BYPSHAD bit in RTC_CR register. When BYPSHAD =0, it is read from the APB field, and when BYPSHAD =1, it is read directly from the RTC field.

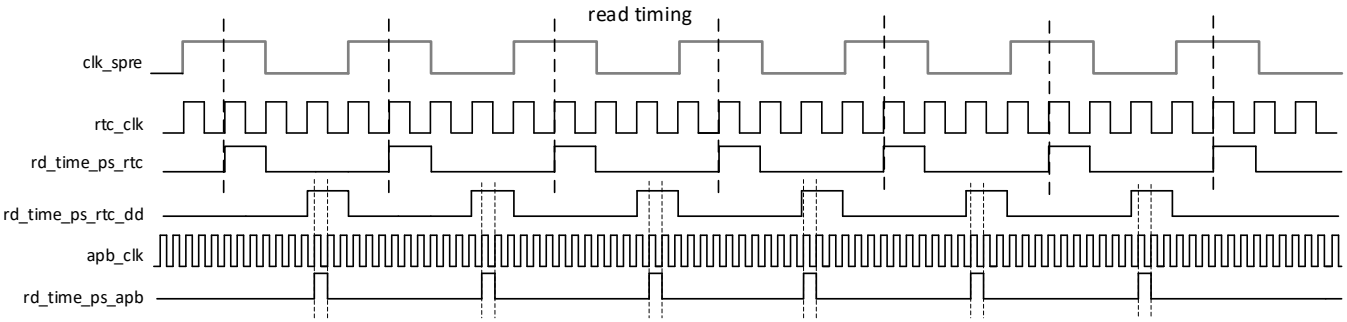


Figure 17-2 RTC read timing

17.4 Registers

Name	Address offset	Access	Description
REG_RTC_TR	000h	R/W	RTC Time Register
REG_RTC_CR	004h	R/W	RTC Control Register
REG_RTC_ISR	008h	R/W	RTC Initialization Mode and Status Register
REG_RTC_PRER	00Ch	R/W	RTC Prescaler Register
REG_RTC_CALIBR	010h	R/W	RTC Calibration Register
REG_RTC_ALMR1L	014h	R/W	RTC Alarm 1 Register Low
REG_RTC_ALMR1H	018h	R/W	RTC Alarm 1 Register High
REG_RTC_WPR	01Ch	R/W	RTC Write Protection Register
REG_RTC_YEAR	020h	R/W	RTC Year Register
REG_RTC_WUTR	024h	R/W	RTC Wakeup Timer Register
REG_RTC_DUMMY	028h	R/W	RTC Dummy Register

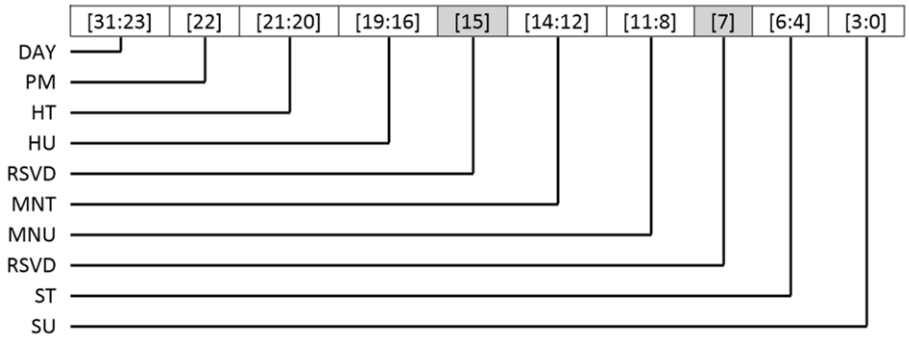
17.4.1 REG_RTC_TR

- **Name:** RTC Time Register
- **Size:** 32 bits

All information provided in this document is subject to legal disclaimers.
© REALTEK 2024. All rights reserved.

- **Address offset:** 000h
- **Read/write access:** R/W

This register is the calendar time shadow register. This register is write protected, and must be written in initialization mode only

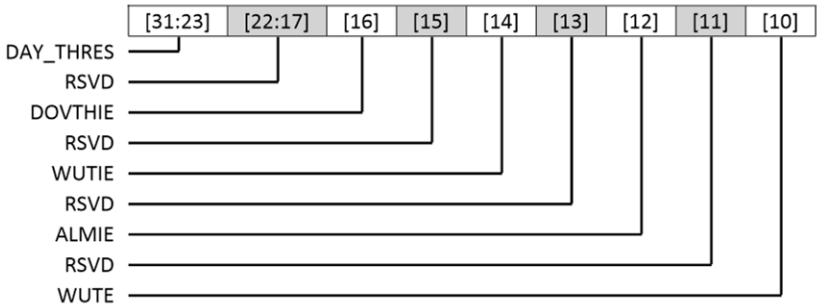


Bit	Symbol	Access	Reset	Description
31:23	DAY	R/W	0	DAY in binary format
22	PM	R/W	0	PM:AM/PM notation **0**: AM or 24-hour format **1**: PM
21:20	HT	R/W	0	Hour tens in BCD format
19:16	HU	R/W	0	Hour units in BCD format
15	RSVD	-	0	Reserved
14:12	MNT	R/W	0	Minute tens in BCD format
11:8	MNU	R/W	0	Minute units in BCD format
7	RSVD	-	0	Reserved
6:4	ST	R/W	0	Second tens in BCD format
3:0	SU	R/W	0	Second units in BCD format

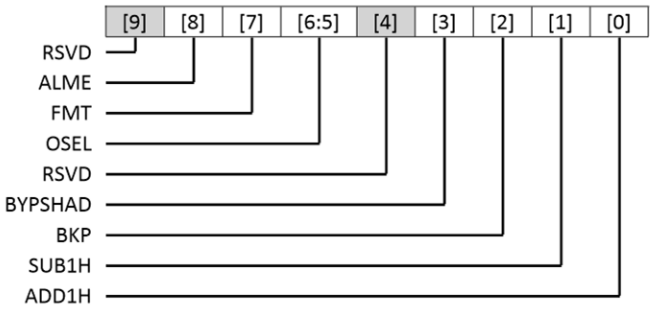
17.4.2 REG_RTC_CR

- **Name:** RTC Control Register
- **Size:** 32 bits
- **Address offset:** 004h
- **Read/write access:** R/W

This register is write protected. Bit[7] (FMT) of this register can be written in initialization mode only when INITF = 1. ADD1H and SUB1H changes are effective in 2~3 seconds. Don't write this register continuously without any delay when RTC is in free run mode. Software can use the RSF bit in RTC_ISR register to handle the delay.



All information provided in this document is subject to legal disclaimers.
© REALTEK 2024. All rights reserved.



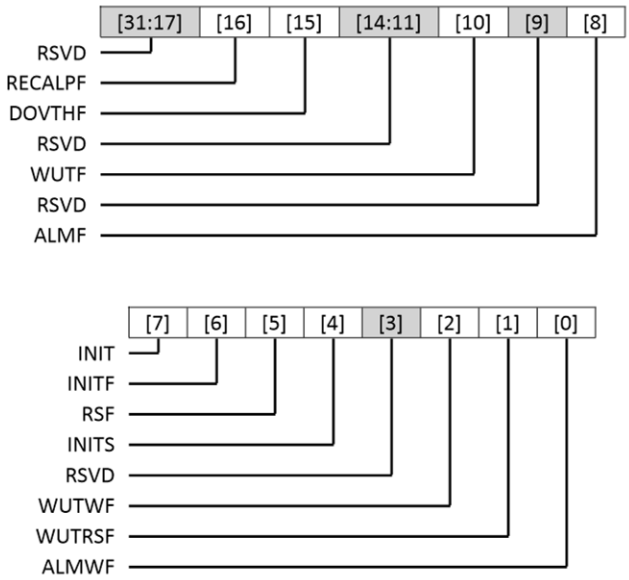
Bit	Symbol	Access	Reset	Description
31:23	DAY_THRES	R	1FFh	Day threshold in binary format
22:17	RSVD	-	0	Reserved
16	DOVTHIE	R/W	0	Day over threshold interrupt enable **0**: Day over threshold interrupt disabled **1**: Day over threshold interrupt enabled
15	RSVD	-	0	Reserved
14	WUTIE	R/W	0	Wakeup timer interrupt enable **0**: Wakeup timer interrupt disabled **1**: Wakeup timer interrupt enabled
13	RSVD	-	0	Reserved
12	ALMIE	R/W	0	Alarm interrupt enable **0**: Alarm interrupt disabled **1**: Alarm interrupt enabled
11	RSVD	-	0	Reserved
10	WUTE	R/W	0	Wakeup timer enable **0**: Wakeup timer disabled **1**: Wakeup timer enabled
9	RSVD	-	0	Reserved
8	ALME	R/W	0	Alarm enable **0**: Alarm disabled **1**: Alarm enabled
7	FMT	R/W	0	Hour format **0**: 24 hour/day format **1**: AM/PM hour format
6:5	OSEL	R/W	0	Output selection. These bits are used to select the flag to be routed to RTC_OUT output. **00**: Output disabled **01**: Alarm output enabled **10**: Clock output is clk_spre (default: 1Hz) **11**: Clock output is clk_apre (default: 256Hz)
4	RSVD	-	0	Reserved
3	BYPSHAD	R/W	0	Bypass the shadow registers. **0**: Calendar values (when reading from RTC_TR) are taken from the shadow register, which are updated once every two RTCCLK cycles. **1**: Calendar values (when reading from RTC_TR) are taken directly from the calendar counters.
2	BKP	R/W	0	Backup This bit can be written by the user to memorize whether the daylight saving time change has been performed or not.
1	SUB1H	W	0	Subtract one hour (winter time changes) When this bit is set outside initialization mode, 1 hour is subtracted to the calendar time if the current hour is not 0, and this bit is always read as 0. **0**: No effect **1**: Subtract one hour to the current time
0	ADD1H	W	0	Add one hour (summer time changes) When this bit is set outside initialization mode, 1 hour is added to the calendar time, and this bit is always read as 0. **0**: No effect **1**: Add one hour to the current time

All information provided in this document is subject to legal disclaimers.
© REALTEK 2024. All rights reserved.

17.4.3 REG_RTC_ISR

- **Name:** RTC Initialization Mode and Status Register
- **Size:** 32 bits
- **Address offset:** 008h
- **Read/write access:** R/W

The ALMF/WUTF/DOVTHF bit can be written without unlocking the write protection. Two APB clock cycles after programming it to 1, this bit is cleaned.



Bit	Symbol	Access	Reset	Description
31:17	RSVD	-	0	Reserved
16	RECALPF	R	0	Recalibration pending flag The RECALPF status flag is automatically set to '1' when software writes to the RTC_CALIBR register, indicating that the RTCCALIBR register is blocked. When the new calibration settings are taken into account, this bit returns to '0'. Refer to Re-calibration on-the-fly.
15	DOVTHF	RW1C	0	Day over threshold flag This flag is set by hardware when the Day in RTC_TR over the DAY_THRES set in RTC_CR register.
14:11	RSVD	-	0	Reserved
10	WUTF	RW1C	0	Wakeup timer flag This flag is set by hardware when the wakeup auto-reload counter reaches to. This flag must be cleared by software at least 2 RTCCLK periods before WUTF is set to '1' again.
9	RSVD	-	0	Reserved
8	ALMF	RW1C	0	Alarm flag This flag is set by hardware when the time register (RTC_TR) matches the Alarm registers (RTC_ALMR1L and RTC_ALMR1H).
7	INIT	R/W	0	Initialization mode **0**: Free running mode. **1**: Initialization mode used to program time and date register (RTC_TR) and prescaler register (RTC_PRER). Counters are stopped and start counting from the new value when INIT is set.
6	INITF	R	0	Initialization flag. When this bit is set to '1', the RTC is in initialization state, and the time, date, and prescaler registers can be updated. **0**: Calendar registers update is not allowed. **1**: Calendar registers update is allowed.
5	RSF	RW1C	0	Registers synchronization flag. This bit is set by hardware each time the calendar registers are copied into the shadow registers (RTC_TR). This bit is cleared by hardware in initialization mode or when in bypass shadow register mode (BYPSHAD=1). This bit can also be cleared by software. It is cleared either by software or by hardware in initialization mode.

All information provided in this document is subject to legal disclaimers.
© REALTEK 2024. All rights reserved.

				0: Calendar shadow registers are not yet synchronized. **1**: Calendar shadow registers are synchronized.
4	INITS	R	0	This bit is set by hardware when the calendar day field is different from 0 (RTC domain reset state). **0**: Calendar has not been initialized. **1**: Calendar has been initialized.
3	RSVD	-	0	Reserved
2	WUTWF	R	0	Wakeup timer write flag This flag is set by hardware when WUT value can be changed, after the WUTE bit has been set to '0' in RTC_CR. **0**: Wakeup timer configuration update not allowed except in initialization mode **1**: Wakeup timer configuration update allowed
1	WUTRSF	RW1C	0	This bit is set by hardware each time the WUTE bit is copied into the shadow register. This bit is cleared by hardware in initialization mode. This bit can also be cleared by software. It is cleared either by software or by hardware in initialization mode. **0**: WUTE is not yet synchronized. **1**: WUTE is synchronized.
0	ALMWF	R	0	Alarm write flag This bit is set by hardware when Alarm values can be changed, after the ALME bit has been set to '0' in RTC_CR. It is cleared by hardware when ALME bit has been set to '1' in RTC_CR. **0**: Alarm update not allowed **1**: Alarm update allowed

17.4.4 REG_RTC_PRER

- **Name:** RTC Prescaler Register
- **Size:** 32 bits
- **Address offset:** 00Ch
- **Read/write access:** R/W

This register is write protected, and must be written in initialization mode only.



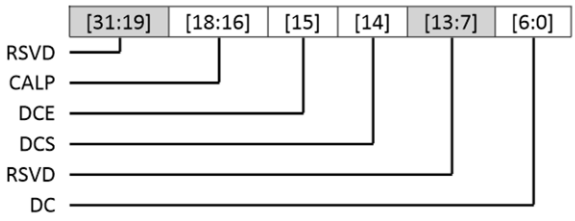
Bit	Symbol	Access	Reset	Description
31:25	RSVD	-	0	Reserved
24:16	PREDIV_A	R/W	7Fh	Asynchronous prescaler factor This is the asynchronous division factor: clk_apre freq = RTCCLK freq/(PREDIV_A + 1)
15:9	RSVD	-	0	Reserved
8:0	PREDIV_S	R/W	FFh	Synchronous prescaler factor This is the synchronous division factor: clk_spre freq = clk_apre freq/(PREDIV_S + 1)

17.4.5 REG_RTC_CALIBR

- **Name:** RTC Calibration Register
- **Size:** 32 bits
- **Address offset:** 010h
- **Read/write access:** R/W

This register is write protected, and can be dynamically configured when RTC is running.

All information provided in this document is subject to legal disclaimers.
© REALTEK 2024. All rights reserved.

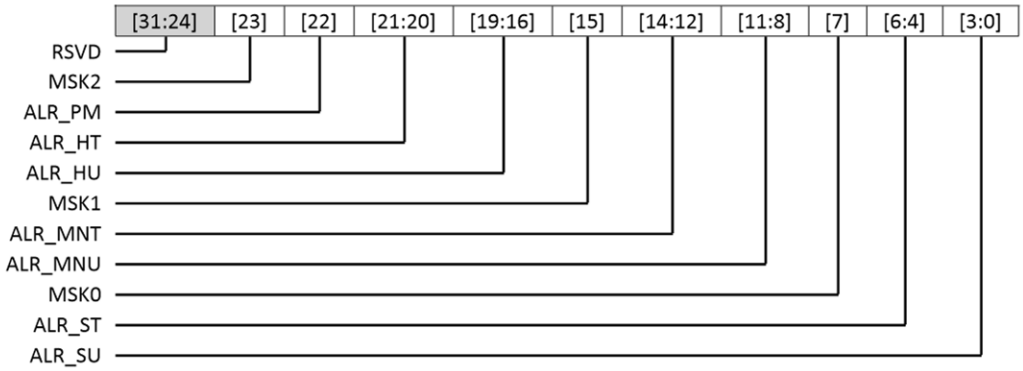


Bit	Symbol	Access	Reset	Description
31:19	RSVD	-	0	Reserved
18:16	CALP	R/W	0	Calibration period
15	DCE	R/W	0	Digital calibration enable **0**: Digital calibration disabled **1**: Digital calibration enabled
14	DCS	R/W	0	Digital calibration signal **0**: Positive calibration, time update frequency is increased. **1**: Negative calibration, time update frequency is decreased.
13:7	RSVD	-	0	Reserved
6:0	DC	R/W	0	Digital calibration

17.4.6 REG_RTC_ALMR1L

- **Name:** RTC Alarm 1 Register Low
- **Size:** 32 bits
- **Address offset:** 014h
- **Read/write access:** R/W

This register is write protected, and can be written only when ALMWF is set to 1 in RTC_ISR register, or in initialization mode.



Bit	Symbol	Access	Reset	Description
31:24	RSVD	-	0	Reserved
23	MSK2	R/W	0	Alarm hour mask **0**: Alarm set if the hours match **1**: Hours don't care in Alarm comparison
22	ALR_PM	R/W	0	AM/PM notation. **0**: AM or 24-hour format **1**: PM
21:20	ALR_HT	R/W	0	Hour tens in BCD format
19:16	ALR_HU	R/W	0	Hour units in BCD format
15	MSK1	R/W	0	Alarm minutes mask. **0**: Alarm set if the minutes match **1**: Minutes don't care in Alarm comparison
14:12	ALR_MNT	R/W	0	Minute tens in BCD format
11:8	ALR_MNU	R/W	0	Minute units in BCD format
7	MSK0	R/W	0	Alarm seconds mask

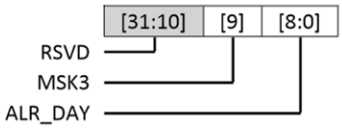
All information provided in this document is subject to legal disclaimers.
© REALTEK 2024. All rights reserved.

				0: Alarm set if the seconds match **1**: Seconds don't care in Alarm comparison
6:4	ALR_ST	R/W	0	Second tens in BCD format
3:0	ALR_SU	R/W	0	Second units in BCD format

17.4.7 REG_RTC_ALMR1H

- **Name:** RTC Alarm 1 Register High
- **Size:** 32 bits
- **Address offset:** 018h
- **Read/write access:** R/W

This register is write protected, and can be written only when ALMWF is set to 1 in RTC_ISR register, or in initialization mode

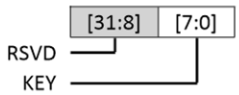


Bit	Symbol	Access	Reset	Description
31:10	RSVD	-	0	Reserved
9	MSK3	R/W	0	Alarm day mask **0**: Alarm set if the days match **1**: Days don't care in Alarm comparison
8:0	ALR_DAY	R/W	0	DAY in binary format

17.4.8 REG_RTC_WPR

- **Name:** RTC Write Protection Register
- **Size:** 32 bits
- **Address offset:** 01Ch
- **Read/write access:** R/W

Write "0xCA" then write "0x53" to disable write protection, and write "0xFF" to enable write protection.



Bit	Symbol	Access	Reset	Description
31:8	RSVD	-	0	Reserved
7:0	KEY	R/W	0	Write protection key This byte is written by software. Refer to RTC register write protection for a description of how to unlock RTC register write protection.

17.4.9 REG_RTC_YEAR

- **Name:** RTC Year Register
- **Size:** 32 bits
- **Address offset:** 020h
- **Read/write access:** R/W

This register is controlled by software.

All information provided in this document is subject to legal disclaimers.
© REALTEK 2024. All rights reserved.

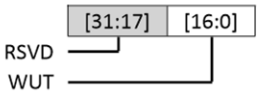


Bit	Symbol	Access	Reset	Description
31	RESTORE	R/W	0	Reset Flag. Indicates whether any reset conditions have occurred (except POR/PDR/BOD), so that the bit is set to '1' before reset, and then the global variable that determines whether the recovery time information is needed depending on whether the bit is '1'.
30:8	RSVD	-	0	Reserved
7:0	YEAR	R/W	0	Year in binary format

17.4.10 REG_RTC_WUTR

- **Name:** RTC Wakeup Timer Register
- **Size:** 32 bits
- **Address offset:** 024h
- **Read/write access:** R/W

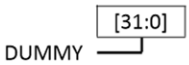
This register is used to set wakeup timer value.



Bit	Symbol	Access	Reset	Description
31:17	RSVD	-	0	Reserved
16:0	WUT	R/W	0	Wakeup auto-reload value bits When the wakeup timer is enabled (WUTE is set to '1'), the WUTF flag is set every (WUT[16:0]+1) cycles. The first assertion of WUTF occurs between WUT and (WUT+1) cycles after WUTE is set.

17.4.11 REG_RTC_DUMMY

- **Name:** RTC Dummy Register
- **Size:** 32 bits
- **Address offset:** 028h
- **Read/write access:** R/W



Bit	Symbol	Access	Reset	Description
31:0	DUMMY	R/W	ffffh	Dummy register

17.5 Operation Flow

17.5.1 Initializing the Calendar

The initialization steps of the calendar are illustrated in *Table 17-2*.

Table 17-2 Initializing the calendar

Step	What to do	How to do	Comments
1	Disable the RTC registers write protection	Write "0xCA" and then "0x53" into the RTC_WPR register	RTC registers can be modified
2	Enter initialization mode	Set INIT bit to '1' in RTC_ISR register	The calendar counter is stopped to allow update

All information provided in this document is subject to legal disclaimers.
© REALTEK 2024. All rights reserved.

3	Wait for the confirmation of initialization mode (clock synchronization)	Poll INITF bit in RTC_ISR until it is set	It takes approximately 2 RTCCLK clock cycles for medium density devices
4	Program the prescaler register if needed	Configure RTC_PRER register	By default, the RTC_PRER prescaler register is initialized to provide 1Hz to the calendar unit when RTCCLK=32768Hz
5	Load time value in the shadow registers	Set RTC_TR	
6	Configure the time format (12-hour or 24-hour)	Set FMT bit in RTC_CR register	FMT=0: 24 hour/day format FMT=1: AM/PM hour format
7	Exit initialization mode	Clear INIT bit in RTC_ISR register	The current calendar counter is automatically loaded and the counting restarts after 4 RTCCLK clock cycles.
8	Enable the RTC registers write protection	Write "0xFF" into the RTC WPR register	RTC registers can no longer be modified

17.5.2 Configuring Alarm

The configuration steps of alarm are illustrated in *Table 17-3*.

Table 17-3 Configuring alarm

Step	What to do	How to do	Comments
1	Disable the RTC registers write protection	Write "0xCA" and then "0x53" into the RTC_WPR register	RTC registers can be modified
2	Disable alarm	Clear ALME bit in RTC_CR register	Read RTC_CR first and set ALME to 1'b0
3	Check that the RTC_ALRMA register can be accessed	Poll ALMWF bit until it is set in RTC_ISR	It takes approximately two RTCCLK clock cycles(clock synchronization)
4	Configure the alarm	Configure RTC_ALMR1 and RTC_ALMR2 register	The alarm hour format must be the same as the RTC calendar
5	Re-enable alarm	Set ALME bit in RTC_CR register	
6	Enable the RTC registers write protection	Write "0xFF" into the RTC_WPR register	RTC registers can no longer be modified

NOTE

- *The alarm behavior can be configured using the MSKx bits (x=0, 1, 2, 3) of the RTC_ALMR1 and RTC_ALMR2 registers.*
- *Alarm can also be configured in the initialization mode.*

17.5.3 Configuring Calibration

The configuration steps of calibration are illustrated in *Table 17-4*.

Table 17-4 Configuring calibration

Step	What to do	How to do	Comments
1	Disable the RTC registers write protection	Write "0xCA" and then "0x53" into the RTC_WPR register	RTC registers can be modified
2	Check that the RTC_CALIBR register can be updated	Poll the RTC_ISR/RECALPF until it is 0 in RTC_ISR	
3	Configure the calibration parameter	Configure RTC_CALIBR register	
4	Enable the RTC registers write protection	Write "0xFF" into the RTC_WPR register	RTC registers can no longer be modified

NOTE

- *Calibration can also be configured in the initialization mode.*
- *The calibration parameter will be not used immediately. HW will automatically reload the new calibration parameter at the end of calibration period.*

17.5.4 Programming the Wakeup Timer

The following sequence is required to configure or change the wakeup timer auto-reload value (WUT[16:0] in RTC_WUTR).

All information provided in this document is subject to legal disclaimers.
© REALTEK 2024. All rights reserved.

Table 17-5 Programming wakeup timer

Step	What to do	How to do	Comments
1	Disable the RTC registers write protection	Write "0xCA" and then "0x53" into the RTC_WPR register	RTC registers can be modified
2	Disable the wakeup timer	Clear WUTE in RTC_CR register	Read RTC_CR first and set WUTE to 1'b0
3	Check that wakeup auto-reload counter can be updated	Poll WUTWF in RTC_ISR register until it is set	
4	Program the wakeup auto-reload value	Configure WUT[16:0] in RTC_WUTR register	
5	Re-enable the wakeup timer	Set WUTE in RTC_CR register	The WUTWF bit is cleared up to 2 RTCCLK clocks cycles after WUTE is set, due to clock synchronization.
6	Wait for the confirmation of configuration (clock synchronization)	Poll WUTRSF bit in RTC_ISR until it is set	It takes approximately 3 to 4 RTCCLK clock cycles before the wakeup timer restarts down-counting
7	Enable the RTC registers write protection	Write "0xFF" into the RTC_WPR register	RTC registers can no longer be modified

17.5.5 Daylight Saving Time

In the U.S., 2:00 a.m. was originally chosen as the changeover time because it was practical and minimized disruption. Most people were at home and this was the time when fewest trains were running. It is late enough to minimally affect bars and restaurants, and it prevents the day from switching to yesterday, which would be confusing. It is early enough that the entire continental U.S. switches by daybreak, and the changeover occurs before earliest shift workers and early churchgoers are affected.

All information provided in this document is subject to legal disclaimers.
© REALTEK 2024. All rights reserved.

18 Backup Register

The backup register contains four 32-bit registers, which are used for Realtek and users to save some data before reset happens.

Table 18-1 illustrates two reset types which can reset RTC module and backup register, which are both in AON domain. Other reset types can't reset RTC and backup register.

Table 18-1 Reset domain of RTC and backup register

Reset signal	Reset type	Description
por_rstb	POR	A power on reset is generated when power on
pdr_rstb	PDR	A power down reset is generated when a low level on the CHIP_EN pin (external reset) occurs

The power domain of RTC and backup register is in *Table 18-2*.

Table 18-2 Power domain of RTC and backup register

Item	Sleep (WoWLAN)	Deep-sleep
RTC	√	√
Backup register	√	√

NOTE

✓ *means this module has power.*

All information provided in this document is subject to legal disclaimers.
© REALTEK 2024. All rights reserved.

19 Watchdog Timer

19.1 Introduction

This chip integrates four watchdog timers: one independent watchdog timer (IWDG) and three system watchdog timers (WDG).

- IWDG: a shared watchdog for KM0 and KM4.
- WDG0: a watchdog timer for KM0
- WDG1: a secure watchdog timer for KM4
- WDG2: a non-secure watchdog timer for KM4

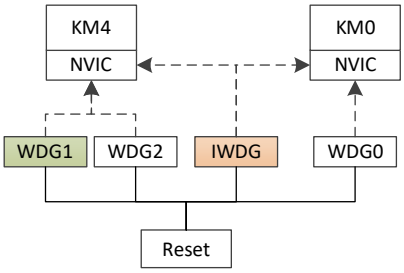


Figure 19-1 Watchdog architecture

All watchdog timers can trigger the reset of the corresponding CPU or the whole system. The reset logic is illustrated in the following sections.

19.2 Independent Watchdog Timer (IWDG)

The independent watchdog timer (IWDG) is powered by always-on power and always-on clock source, which means that it can stay active even when the core power or crystal fails. Also, IWDG can be configured to start by hardware at the very beginning of reset.

19.2.1 Features

- Powered by always-on power and always-on clock source
- Once enabled, cannot be disabled
- Can wake up the system from sleep mode with an early interrupt
- Can reset the whole system even in sleep mode
- Optional early interrupt can be generated at a programmable time prior to watchdog timeout
- Gated automatically when the KM0 or KM4 processor is in debug mode
- Can be configured to run or gate in sleep mode
- A separate wake reason for each watchdog timer

19.2.2 Clock and Power

The IWDG is powered by always-on power and clocked by an always-on 100kHz. The processer cannot shut off the power and clock after the system boots up, so there is no way to disable the IWDG after enabled.

The always-on 100kHz clock is a no-calibration clock, so it may have a margin of error of 50%. When feeding the watchdog, software should take the margin into consideration. Moreover, we do not suggest enabling the window function of IWDG.

The IWDG will be reset in deep-sleep mode, so the IWDG should be enabled again by software or hardware after wakeup again. IWDG can be configured to retain or keep counting in sleep mode by setting the LPEN bit in the control register. The watchdog timeout event in sleep mode can reset the whole system, and the early interrupt in sleep mode can wake up the system.

The early interrupt of IWDG is connected to both KM4 and KM0, however only the CPU working as KM0 is suggested to handle the interrupt.

OTP can auto-load an AON register to trigger IWDG enable. The register can only be set from 0 to 1, meaning that once the IWDG is enabled by hardware, it will automatically run no matter the initial boot or waking from deep-sleep.

The IWDG keeps counting when one of the CPUs is active (both clock and function enabled) and non-CPU is in debug mode.

All information provided in this document is subject to legal disclaimers.
© REALTEK 2024. All rights reserved.

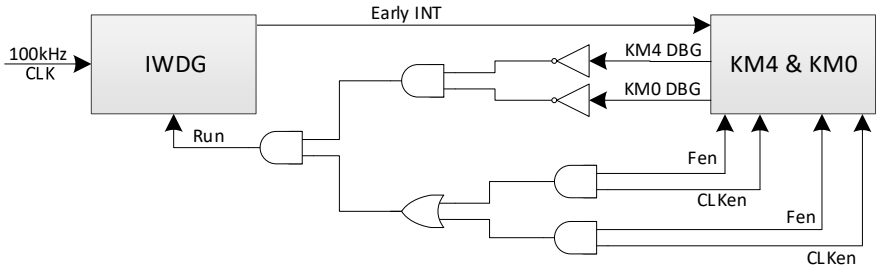


Figure 19-2 IWDG clock and function enable diagram

19.2.3 Reset Architecture

The IWDG has the highest privilege. The IWDG timeout will trigger GLB_RST, and this reset signal will reset the whole system, including all the processors and almost all the peripherals. Only very few peripherals won't be reset by GLB_RST, e.g. RTC, AON PAD.

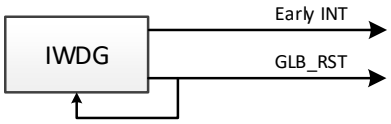


Figure 19-3 IWDG reset control

19.3 System Watchdog Timer (WDG)

19.3.1 Features

- Optional early interrupt can be generated at a programmable time prior to watchdog timeout
- Gate automatically when the processor is in debug mode
- Gate and maintains settings in sleep mode.
- Window protection function and timeout cannot be changed anymore once WDG is enabled.
- A separate boot reason for each watchdog timer

19.3.2 Clock and Power

The system watchdog timers are powered by SYSON and clocked by SDM32K.

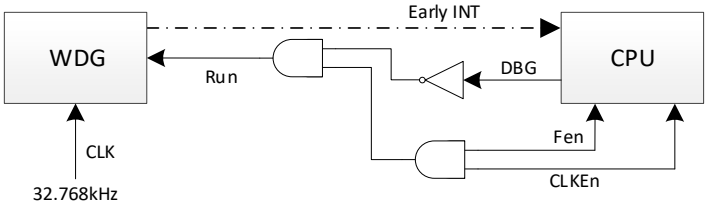


Figure 19-4 SWDG clock and function enable diagram

19.4 Functional Description

19.4.1 Workflow

The workflow of system watchdog timer is illustrated in *Figure 19-5*.

All information provided in this document is subject to legal disclaimers.
© REALTEK 2024. All rights reserved.

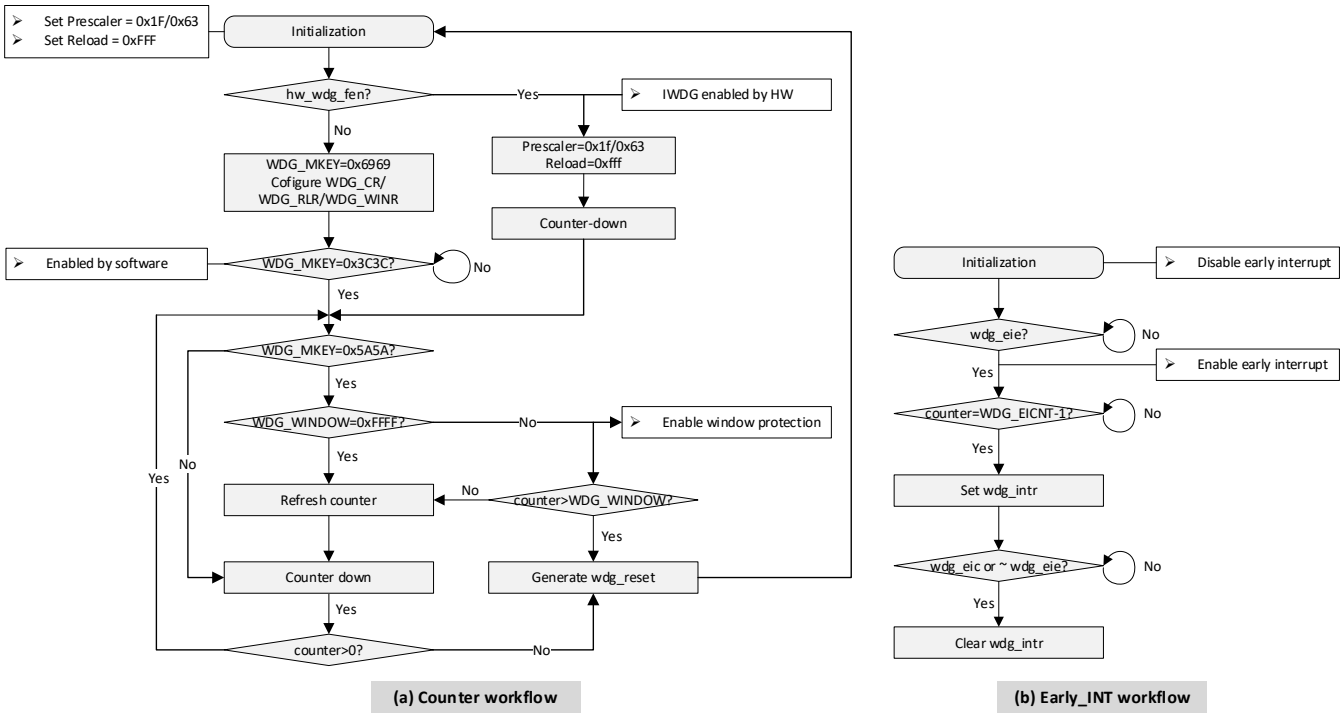


Figure 19-5 Watchdog workflow

19.4.2 Initialization

Table 19-1 Watchdog initialization

Step	What to do	How to do	Comment
1	Check HW IWDG enable	IWDG can start counting when hw_wdg_fen is asserted. ● For the IWDG prescaler, using default value 0x63 ● For the WDG prescaler, using default value 0x1F For the reload counter, using default value 0xFFF	IWDG can be enabled by HW and SW, but WDG can only be enabled by SW.
2	Enable register access	Write "0x6969" into the WDG_MKEYR register	Enable register access to the WDG_CR/ WDG_RLR/WDG_WINR registers
3	Program Reload register	Configure the RELOAD and PRER bits of the WDG_RLR register	
4	Program Control register if needed	Configure the EICNT bit of the WDG_CR register	
5	Program Window register if needed	Configure the WDG_WINR register	Enable the window protection
6	Enable SW WDG	Write "0x3C3C" into the WDG_MKEYR register	WDG is enabled by SW.
7	Enable reload command if needed	Write "0x5A5A" into the WDG_MKEYR register	Reload value for WDG counter

19.4.3 Operation Flow

19.4.3.1 Hardware Enable

In the OTP area, one bit is used to control the IWDG automatically to enable the function. There is no difference between hardware WDG (enabled by hardware) and software WDG (enabled by software). When the HW WDG is enabled, the prescaler will use default value 0x63 and the reload counter will use default value 0xFFF.

An application can adjust the barking interval of the watchdog. Once the IWDG is enabled, it cannot be disabled anymore. No matter which way is used to enable the watchdog timer, the software is responsible for feeding the dog to prevent barking.

The system watchdog timers can only be enabled by software.

19.4.3.2 Low-power Mode

For IWDG, whether the IWDG can keep running or gating in sleep mode depends on the LPEN bit.

- If LPEN is cleared, the clock for IWDG will be gated internally, and IWDG will be frozen with all status maintained.

All information provided in this document is subject to legal disclaimers.
© REALTEK 2024. All rights reserved.

- If LPEN is enabled, IWDG will keep counting down, wake up the system if the early interrupt event fires, and reset system if the dog barks.

For system watchdog timers, each of them can maintain its registers but keep freezing in sleep mode. Counting down will resume after leaving from sleep.

However, IWDG will be disabled by HW and the system watchdog timers will be powered off in deep-sleep mode.

19.4.3.3 Register Protection

After the watchdog timer is enabled, WDG_MKEYR is the only register that can be accessed at any time. Other registers can be accessed only when WDG_MKEYR equals 0x6969.

19.4.3.4 Early Interrupt

The watchdog timer can trigger an interrupt to wake up the system or inform the system that a watchdog reset is coming. This interrupt can be enabled by setting a proper value to EARLY_INT_CNT and EIE. When the down-counter (WDGCNT) reaches the value of (EARLY_INT_CNT -1), the interrupt fires.

It is possible to change the early interrupt threshold after EIE is enabled, by performing the following sequences:
(1) Enable register access by writing 0x6969 into the WDG_MKEYR register
(2) Disable the EIE bit of the WDG_CR register
(3) Wait for EVU = '0', EVU is also located in the WDG_CR register
(4) Program the early interrupt threshold and enable the early interrupt in the WDG_CR register
(5) Wait for EVU = '0'
(6) Refresh the counter with RL[15:0], and write-protect registers by writing 0x5A5A into the WDG_MKEYR register

19.4.3.5 Window Protection

The watchdog timer can also work as a window watchdog by setting the KM4 appropriate window in the WDG_WIN. If the reload operation is performed while the counter is greater than WDG_WIN[15:0], a reset is generated. The default value of the WDG_WIN is 0x0000FFFF, if not updated, the window option is disabled.

In order to prevent malicious attacks from changing the value of window, the window value can only be set before watchdog enable. Once the watchdog is enabled, the value cannot be changed. Due to the inaccurate of the source clock, we do no suggest enabling the window protection function of IWDG.

19.4.3.6 Prescaler

The watchdog timer itself works in a low clock (IWDG works at 100kHz and system watchdog timers work at 32kHz), while the access to the registers works in KM4B domain whose frequency is much higher than the clock of watchdog timer. If the prescaler dynamic change is involved, the cross-clock synchronization must be considered. In order to simplify the programming and design flow, prescaler dynamic change is not allowed. The prescaler can only be configured before watchdog enable.

19.4.3.7 Update Indication

Because the watchdog and its registers work in different clock domains, and frequencies are varied, cross-clock synchronization must be considered. In order to prevent the subsequent write operation from affecting the previous operation, two status flags are introduced in *Table 19-2*.

Table 19-2 Status flags

Bit	Flag	Description
EVU	Watchdog early interrupt function update	This bit is set by hardware to indicate that an update of the interrupt threshold (EICNT[15:0]) is ongoing or an update of the EIE is ongoing. It is reset by hardware when the update operation is completed. The EICNT[15:0] and EIE fields can be updated only when the EVU bit is reset.
RVU	Watchdog counter value update	This bit is set by hardware to indicate that the reload operation is ongoing. It is reset by hardware when the counter update operation is completed. The reload value can be updated only when the RVU bit is reset.

All information provided in this document is subject to legal disclaimers.
© REALTEK 2024. All rights reserved.

Before writing to the watchdog register, software needs to check these flags to ensure no conflict and disorder issues.

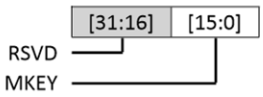
19.5 Registers

19.5.1 IWDG Registers

Name	Address offset	Access	Description
REG_WDG_MKEYR	000h	R/W	WDG Magic Key Register
REG_WDG_CR	004h	R/W	WDG Control Regsietr
REG_WDG_RLR	008h	R/W	WDG Relaod Register
REG_WDG_WINR	00Ch	R/W	WDG Window Register
REG_IWDG_DUMMY	010h	R/W	WDG Dummy Register

19.5.1.1 REG_WDG_MKEYR

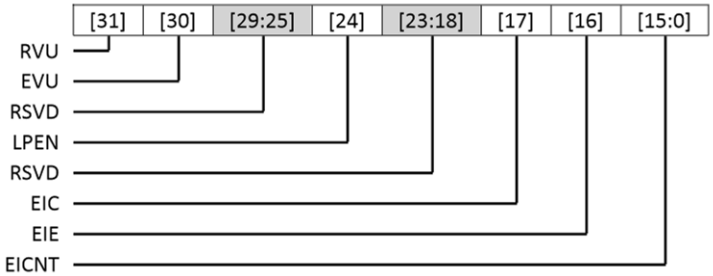
- **Name:** WDG Magic Key Register
- **Size:** 32 bits
- **Address offset:** 000h
- **Read/write access:** R/W



Bit	Symbol	Access	Reset	Description
31:16	RSVD	-	0	Reserved
15:0	MKEY	R/WPD	0h	0x6969: enable access to register WDG_CR/WDG_RLR/WDG_WINR 0x5A5A: reload WDG counter 0x3C3C: enable WDG function

19.5.1.2 REG_WDG_CR

- **Name:** WDG Control Regsietr
- **Size:** 32 bits
- **Address offset:** 004h
- **Read/write access:** R/W



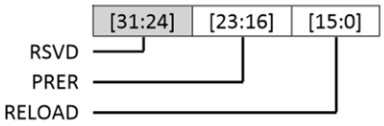
Bit	Symbol	Access	Reset	Description
31	RVU	R	0h	Watchdog counter update by reload command
30	EVU	R	0h	Watchdog early interrupt function update
29:25	RSVD	-	0	Reserved
24	LPEN	R/WE	1h	Low power enable 0: WDG will gating when system goes into sleep mode 1: WDG keep running when system goes into sleep mode
23:18	RSVD	-	0	Reserved
17	EIC	WA0	0h	Write '1' clear the early interrupt

All information provided in this document is subject to legal disclaimers.
© REALTEK 2024. All rights reserved.

16	EIE	R/WE	0h	Watchdog early interrupt enable
15:0	EICNT	R/WE	0h	Early interrupt trigger threshold

19.5.1.3 REG_WDG_RLR

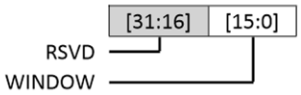
- **Name:** WDG Relaod Register
- **Size:** 32 bits
- **Address offset:** 008h
- **Read/write access:** R/W



Bit	Symbol	Access	Reset	Description
31:24	RSVD	-	0	Reserved
23:16	PRER	R/WE	63h	Prescaler counter, configuration only allowed before WDG enable IWDG: 0x63 System WDG: 0x1F
15:0	RELOAD	R/WE	FFFh	Reload value for watchdog counter

19.5.1.4 REG_WDG_WINR

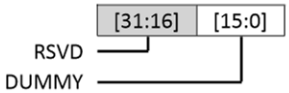
- **Name:** WDG Window Register
- **Size:** 32 bits
- **Address offset:** 00Ch
- **Read/write access:** R/W



Bit	Symbol	Access	Reset	Description
31:16	RSVD	-	0	Reserved
15:0	WINDOW	R/WE	FFFFh	Watchdog feed protect window register

19.5.1.5 REG_IWDG_DUMMY

- **Name:** WDG Dummy Register
- **Size:** 32 bits
- **Address offset:** 010h
- **Read/write access:** R/W



Bit	Symbol	Access	Reset	Description
31:16	RSVD	-	0	Reserved
15:0	DUMMY	R/W	0xFF	Reserved for HW

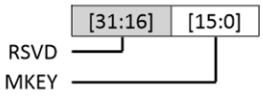
19.5.2 SWDG Registers

All information provided in this document is subject to legal disclaimers.
© REALTEK 2024. All rights reserved.

Name	Address offset	Access	Description
REG_SWDG_MKEYR	000h	R/W	System WDG Magic Key Register
REG_SWDG_CR	004h	R/W	System WDG Control Regsietr
REG_SWDG_RLR	008h	R/W	System WDG Relaod Register
REG_SWDG_WINR	00Ch	R/W	System WDG Window Register
REG_WDG_DUMMY	010h	R	System WDG Dummy Register

19.5.2.1 REG_SWDG_MKEYR

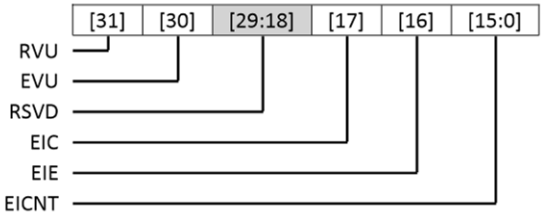
- **Name:** System WDG Magic Key Register
- **Size:** 32 bits
- **Address offset:** 000h
- **Read/write access:** R/W



Bit	Symbol	Access	Reset	Description
31:16	RSVD	-	0	Reserved
15:0	MKEY	R/WPD	0h	0x6969: enable access to register WDG_CR/WDG_RLR/WDG_WINR 0x5A5A: reload WDG counter 0x3C3C: enable WDG function

19.5.2.2 REG_SWDG_CR

- **Name:** System WDG Control Regsietr
- **Size:** 32 bits
- **Address offset:** 004h
- **Read/write access:** R/W

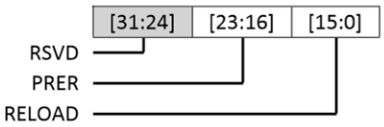


Bit	Symbol	Access	Reset	Description
31	RVU	R	0h	Watchdog counter update by reload command
30	EVU	R	0h	Watchdog early interrupt function update
29:18	RSVD	-	0	Reserved
17	EIC	WA0	0h	Write '1' clear the early interrupt
16	EIE	R/WE	0h	Watchdog early interrupt enable
15:0	EICNT	R/WE	0h	Early interrupt trigger threshold

19.5.2.3 REG_SWDG_RLR

- **Name:** System WDG Relaod Register
- **Size:** 32 bits
- **Address offset:** 008h
- **Read/write access:** R/W

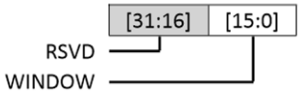
All information provided in this document is subject to legal disclaimers.
© REALTEK 2024. All rights reserved.



Bit	Symbol	Access	Reset	Description
31:24	RSVD	-	0	Reserved
23:16	PRER	R/WE	1Fh	Prescaler counter, configuration only allowed before WDG enable IWDG: 0x63 System WDG: 0x1F
15:0	RELOAD	R/WE	FFFh	Reload value for watchdog counter

19.5.2.4 REG_SWDG_WINR

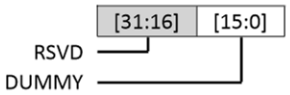
- **Name:** System WDG Window Register
- **Size:** 32 bits
- **Address offset:** 00Ch
- **Read/write access:** R/W



Bit	Symbol	Access	Reset	Description
31:16	RSVD	-	0	Reserved
15:0	WINDOW	R/WE	FFFFh	Watchdog feed protect window register, configuration only allowed before WDG enable

19.5.2.5 REG_WDG_DUMMY

- **Name:** System WDG Dummy Register
- **Size:** 32 bits
- **Address offset:** 010h
- **Read/write access:** R



Bit	Symbol	Access	Reset	Description
31:16	RSVD	-	0	Reserved
15:0	DUMMY	R	0x0	Reserved for HW

All information provided in this document is subject to legal disclaimers.
© REALTEK 2024. All rights reserved.

20 Serial Peripheral Interface (SPI)

20.1 Product Overview

The RTL8721Dx supports Motorola Serial Peripheral Interface (SPI) – A four-wire, full-duplex serial protocol.

20.1.1 Features

- Supports Motorola SPI Serial interface operation
- Master or slave operation mode
- Provides two high-speed SPI ports: configured as master or slave with max. baud rate 50Mbps
- DMA interface for DMA transfer
- Independent masking of interrupts
- The Transmit and Receive FIFO buffers are 64 words depth. The FIFO width is fixed at 16 bits.
- Hardware/Software slave-select
  - Dedicated hardware slave-select lines
  - Software control to select target the serial-slave device
- Programmable features
  - Clock bit-rate – Dynamic control of the serial bit rate of the data transfer; only when configured in master mode
  - Data frame size (4 to 16 bits) – Frame size of each data transfer under the control of the programmer.
  - Configurable clock polarity and phase
  - Programmable delay on the sample time of the received serial data bit (rxd), when configured in master mode
- Transfer mode
  - Transmit and receive
  - Transmit only
  - Receive only
- Operation mode
  - Polling mode
  - Interrupt mode
  - DMA mode

20.1.2 Block Diagram

Figure 20-1 shows the following functional groupings of the main interfaces to the SPI block.

- APB interface and DMA Controller Interface
- Transmit and receive FIFO controllers and an FSM controller
- Register block
- Shift control and interrupt logic

All information provided in this document is subject to legal disclaimers.
© REALTEK 2024. All rights reserved.

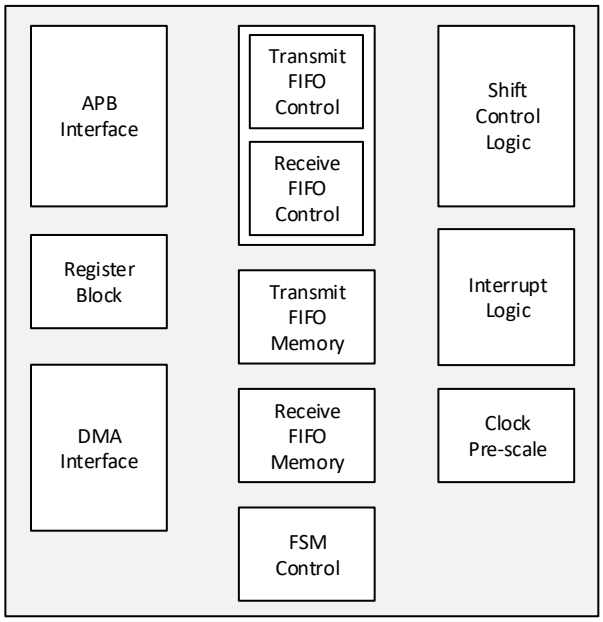


Figure 20-1 SPI block diagram

20.2 Functional Description

This section describes the functional operation of the SPI.

The SPI is a configurable, synthesizable, and programmable component that is a full-duplex master or slave-synchronous serial interface. It can be configured in one of two modes of operations: as a serial master or a serial slave. The SPI can connect to any serial-master or serial-slave peripheral device using Motorola Serial Peripheral Interface (SPI).

20.2.1 Introduction

In order to connect to a serial-master or serial-slave peripheral device for the SPI, the peripheral must have Motorola Serial Peripheral Interface (SPI) – A four-wire, full-duplex serial protocol from Motorola.

The serial protocols supported by the SPI allow for serial slaves to be selected or addressed.

- Single slave: connecting master's Slave Select pin to slave's Slave Select pin.

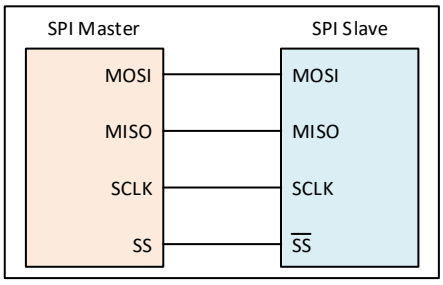


Figure 20-2 Master with single slave device connection

- Multiple slaves: As the SPI master has only a single slave select output (determined by SPI hardware design), users should use GPIOs to control multiple SPI salves. Only one slave should be selected at any time.

All information provided in this document is subject to legal disclaimers.
© REALTEK 2024. All rights reserved.

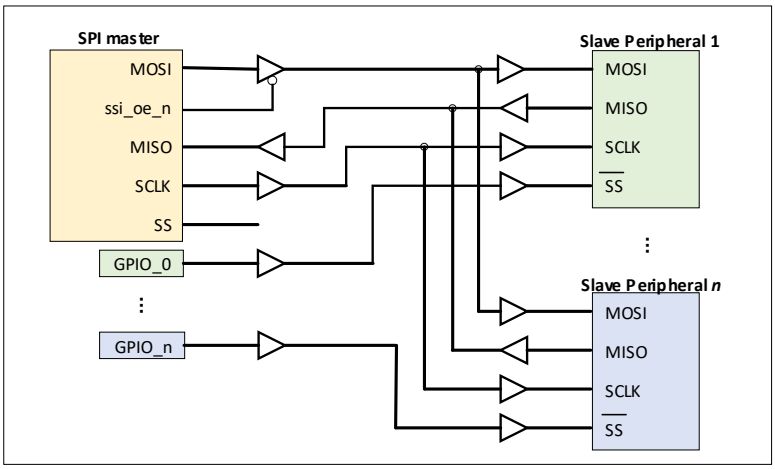


Figure 20-3 Configured master device with multiple slaves

20.2.1.1 Motorola Serial Peripheral Interface (SPI)

There are four possible combinations for the serial clock phase and clock polarity. The clock phase (SCPH) determines whether the serial transfer begins with the falling edge of the slave select signal or the first edge of the serial clock. The clock polarity (SCPOL) configuration parameter determines whether the inactive state of the serial clock is high or low. To transmit data, both SPI peripherals must have identical serial clock phase (SCPH) and clock polarity (SCPOL) values.

When the configuration parameter SCPH = 0, data transmission begins on the falling edge of the slave select signal. The first data bit is captured by the master and slave peripherals on the first edge of the serial clock; therefore, valid data must be present on the MOSI and MISO lines prior to the first serial clock edge. *Figure 20-4* shows a timing diagram for a single SPI data transfer with SCPH = 0. The serial clock is shown for configuration parameters SCPOL = 0 and SCPOL = 1.

The signals are illustrated in the timing diagrams.

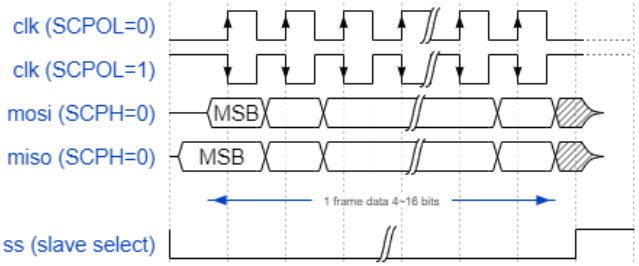


Figure 20-4 SPI serial format (SCPH = 0)

As data transmission starts on the falling edge of the slave select signal when SCPH = 0, whether the slave select signal of SPI master should toggle or not before the next data frame during continuous data transfer can be configured through SS_T in CTRLR0 register. The SPI slave can communicate with serial-master no matter that the slave select signal from master is toggling or not during continuous data transfer. The timing diagram of SS toggling is illustrated in *Figure 20-5*. The timing diagram of SS not-toggling is illustrated in *Figure 20-6*. SS toggling function is used to compatible with different SPI devices.

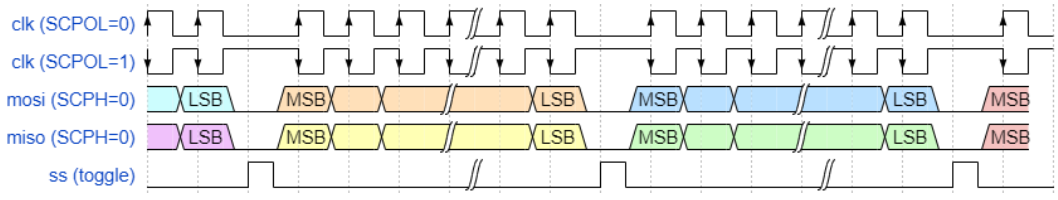


Figure 20-5 SPI serial format continuous transfers (SCPH = 0 and SS toggling)

All information provided in this document is subject to legal disclaimers.
© REALTEK 2024. All rights reserved.

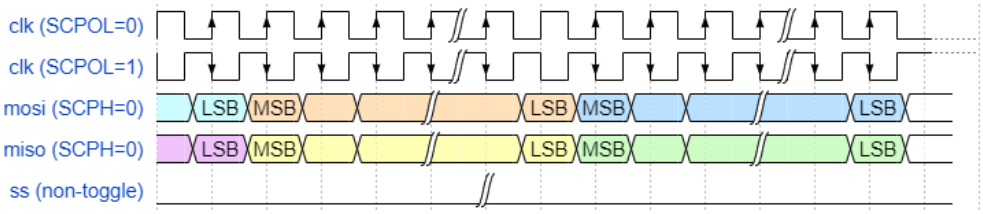


Figure 20-6 SPI serial format continuous transfers (SCPH = 0 and SS not-toggling)

When the configuration parameter SCPH = 1, both master and slave peripherals begin transmitting data on the first serial clock edge after the slave select line is activated. The first data bit is captured on the second (trailing) serial clock edge. Data are propagated by the master and slave peripherals on the leading edge of the serial clock. During continuous data frame transfers, the slave select line may be held active-low until the last bit of the last frame has been captured. *Figure 20-7* shows the timing diagram for the SPI format when the configuration parameter SCPH = 1.

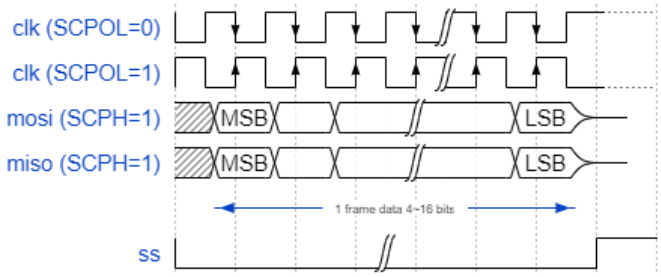


Figure 20-7 SPI serial format (SCPH = 1)

Continuous data frames are transferred in the same way as single frames, with the MSB of the next frame following directly after the LSB of the current frame. The slave select signal is held active for the duration of the transfer. *Figure 20-8* shows the timing diagram for continuous SPI transfers when the configuration parameter SCPH = 1.

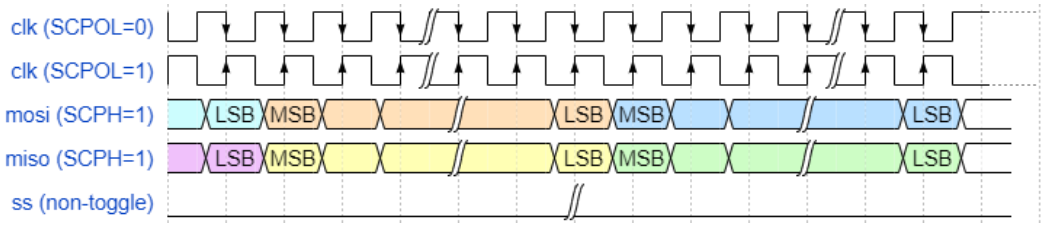


Figure 20-8 SPI serial format continuous transfers (SCPH = 1 and SS not toggling)

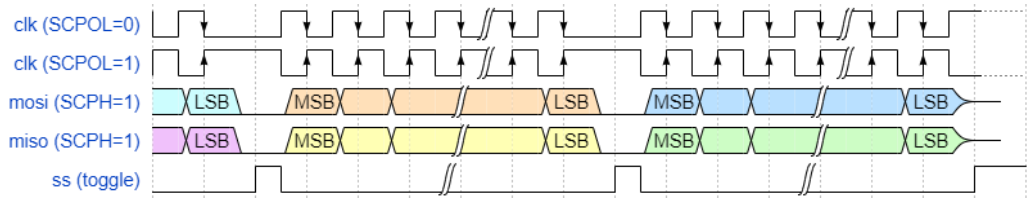


Figure 20-9 SPI serial format continuous transfers (SCPH = 1 and SS toggling)

20.2.1.2 Clock Ratios

When SPI is configured as a master device, the maximum frequency of the bit-rate clock (sclk_out) is one-half the frequency of ssi_clk. This allows the shift control logic to capture data on one clock edge of sclk_out, and propagate data on the opposite edge; as illustrated in *Figure 20-10*. The sclk_out line toggles only when an active transfer is in progress. At all other times, it is held in an inactive state, as defined by the serial protocol under which it operates.

The frequency of sclk_out can be derived from the following equation:

$$F_{sclk_out} = \frac{F_{ssi_clk}}{\mathrm{SCKDV}}$$

SCKDV is a bit field in the programmable register BAUDR, holding any even value in the range 0 to 65,534. If SCKDV is 0, then sclk_out is disabled.

All information provided in this document is subject to legal disclaimers.
© REALTEK 2024. All rights reserved.

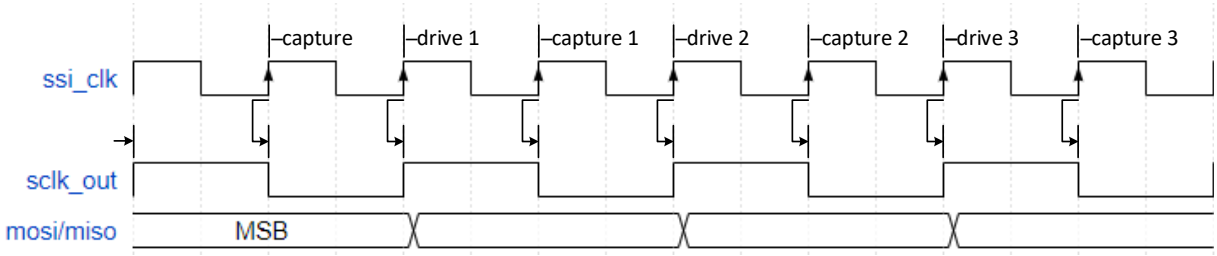


Figure 20-10 Maximum sclk_out/ssi_clk Ratio

The RTL8721Dx supports high-speed SPI slave. When SPI is configured as a slave device, the maximum frequency of the bit-rate clock (sclk_in) provided by serial master can reach up to one-half the frequency of ssi_clk.

A summary of the frequency ratio restrictions between bit-rate clock (sclk_out/sclk_in) and SPI peripheral clock (ssi_clk) are described as:

- Master: Fssi_clk >= 2 × (maximum Fsclk_out)
- Slave: Fssi_clk >= 2 × (maximum Fsclk_in)

The Fssi_clk of SPI0 is 100MHz, the same as the Fssi_clk of SPI1.

20.2.1.3 Transmit and Receive FIFO Buffers

The depth of both transmit and receive FIFO buffers is 64. The width is fixed at 16 bits due to the serial specifications, which state that a serial transfer (data frame) can be 4 to 16 bits in length. Each data entry in the FIFO buffers contains a single data frame.

NOTE

The transmit and receive FIFO buffers are cleared when the SPI is disabled (SSI_EN = 0) or when it is reset.

The transmit FIFO is loaded by APB write commands to the SPI data register (DR). Data are popped (removed) from the transmit FIFO by the shift control logic into the transmit shift register. The transmit FIFO generates a FIFO empty interrupt request (ssi_txe_intr) when the number of entries in the FIFO is less than or equal to the FIFO threshold value. The threshold value, set through the programmable register TXFTLR, determines the level of FIFO entries at which an interrupt is generated. The threshold value allows you to provide early indication to the processor that the transmit FIFO is nearly empty. A transmit FIFO overflow interrupt (ssi_txo_intr) is generated if you attempt to write data into an already full transmit FIFO.

Data are popped from the receive FIFO by APB read commands to the SPI data register (DR). The receive FIFO is loaded from the receive shift register by the shift control logic. The receive FIFO generates a FIFO-full interrupt request (ssi_rxf_intr) when the number of entries in the FIFO is greater than or equal to the FIFO threshold value plus 1. The threshold value, set through programmable register RXFTLR, determines the level of FIFO entries at which an interrupt is generated.

The threshold value allows you to provide early indication to the processor that the receive FIFO is nearly full. A receive FIFO overrun interrupt (ssi_rxo_intr) is generated when the receive shift logic attempts to load data into a completely full receive FIFO. However, this newly received data are lost. A receive FIFO underflow interrupt (ssi_rxu_intr) is generated if you attempt to read from an empty receive FIFO. This alerts the processor that the read data are invalid.

20.2.1.4 SPI Interrupts

The SPI supports combined and individual interrupt requests, each of which can be masked. The combined interrupt request is the ored result of all other SPI interrupts after masking. The system designer has the choice of routing individual interrupt requests or only the combined interrupt request to the Interrupt Controller. The SPI interrupts are described as follows:

- Transmit FIFO Empty Interrupt (ssi_txe_intr) – Set when the transmit FIFO is equal to or below its threshold value and requires service to prevent an under-run. The threshold value, set through a software-programmable register, determines the level of transmit FIFO entries at which an interrupt is generated. This interrupt is cleared by hardware when data are written into the transmit FIFO buffer, bringing it over the threshold level.
- Transmit FIFO Overflow Interrupt (ssi_txo_intr) – Set when an APB access attempts to write into the transmit FIFO after it has been completely filled. When set, data written from the APB is discarded. This interrupt remains set until you read the transmit FIFO overflow interrupt clear register (TXOICR).
- Receive FIFO Underflow Interrupt (ssi_rxu_intr) – Set when an APB access attempts to read from the receive FIFO when it is empty. When set, zeros are read back from the receive FIFO. This interrupt remains set until you read the receive FIFO underflow interrupt clear register (RXUICR).
- Receive FIFO Overflow Interrupt (ssi_rxo_intr) – Set when the receive logic attempts to place data into the receive FIFO after it has been completely filled. When set, the newly received data are discarded. This interrupt remains set until you read the receive FIFO overflow interrupt clear register (RXOICR).
- Receive FIFO Full Interrupt (ssi_rxf_intr) – Set when the receive FIFO is equal to or above its threshold value plus 1 and requires service

All information provided in this document is subject to legal disclaimers.
© REALTEK 2024. All rights reserved.

to prevent an overflow. The threshold value, set through a software-programmable register, determines the level of receive FIFO entries at which an interrupt is generated. This interrupt is cleared by hardware when data are read from the receive FIFO buffer, bringing it below the threshold level.

- Frame Alignment Error Interrupt (ssi_fae_intr) – Present only if the SPI is configured as a serial-slave device. The interrupt is set when the received data frame length is not matching the DFS (Data Frame Size) setting in CTRLR0. This interrupt remains set until you read the Frame Alignment Error Interrupt Clear Register (MSTICR_FAEICR).
- Multi-Master Contention Interrupt (ssi_mst_intr) – Unable to use on this chip.
- Transmit FIFO under Flow Interrupt (ssi_txu_intr) – Present only if the SPI is configured as a serial-slave device. The interrupt is set when transmit logic attempts to read from transmit FIFO when it is empty. When set, data sent from slave device is invalid. This interrupt remains set until you read the transmit FIFO underflow interrupt clear register (TXUICR).
- SS_N Rising Edge Detect Interrupt (ssi_ssr_intr) – Present only if the SPI is configured as a serial-slave device. The interrupt is set when SS_N Rising Edge is detected. This interrupt remains set until you read the SS_N Rising Edge Detect Interrupt Clear Register (SSRICR).

20.2.1.5 SPI Master/Slave Role Set

There are two different sets of circuits when the RTL8721Dx works as SPI Master or SPI Slave, so users need to set the SPI role before initializing the device, the relevant register is described below.

- Name: REG_LSYS_PLAT_CTRL
- Size: 32 bits
- Base Address: 0x4100C000
- Address offset: 0x230
- Read/write access: read/write

This register is valid only when the SPI is configured as serial-master.

Bit	Name	Access	Reset	Description
27	SPI1_MST	R/W	0x0	● 1: SPI1 used as master ● 0: SPI1 used as slaver
26	SPI0_MST	R/W	0x0	● 1: SPI0 used as master ● 0: SPI0 used as slaver

20.2.2 Transfer Modes

When transferring data on the serial bus, the SPI master can operate in transmit and receive mode, transmit only mode or receive only mode as following. The transfer mode (TMOD) is set by writing to control register 0 (CTRLR0).

When transferring data on the serial bus, the SPI slave can only operate in transmit and receive mode. That is, TMOD field in CTRLR0 register is invalid for SPI slave. If you do not want this device to respond with data, slave output can be disabled through SLV_OE bit in CTRLR0.

20.2.2.1 Transmit and Receive

When TMOD = 2'b00, both transmit and receive logic are valid. Transmit data are popped from the transmit FIFO and sent through the MOSI line to the target device, which replies with data on the MISO line. The received data from the target device is moved from the receive shift register into the receive FIFO at the end of each data frame.

20.2.2.2 Transmit Only

When TMOD = 2'b01, the receive data are invalid and should not be stored in the receive FIFO. Transmit data are popped from the transmit FIFO and sent through the MOSI line to the target device, which replies with data on the MISO line. At the end of the data frame, the receive shift register does not load its newly received data into the receive FIFO. The data in the receive shift register is overwritten by the next transfer. You should mask interrupts originating from the receive logic when this mode is entered.

20.2.2.3 Receive Only

When TMOD = 2'b10, the transmit data are invalid. The MOSI output remains at a constant logic level during the transmission. The received data from the target device is moved from the receive shift register into the receive FIFO at the end of each data frame. You should mask interrupts originating from the transmit logic when this mode is entered.

All information provided in this document is subject to legal disclaimers.
© REALTEK 2024. All rights reserved.

20.2.3 Operation Modes

The SPI can be configured in the fundamental modes of operation discussed in this section.

20.2.3.1 Serial-Master Mode

This mode enables serial communication with serial-slave peripheral devices by software. When configured as a serial-master device, the SPI initiates and controls all serial transfers. *Figure 20-3* shows an example of the SPI configured as a serial master with all other devices on the serial bus controlled by GPIO as slave select signal.

The serial bit-rate clock, generated and controlled by the SPI master, is driven out on the sclk_out line. When the SPI is disabled (SSI_EN = 0), no serial transfers can occur and sclk_out is held in an inactive state.

20.2.3.1.1 RXD Sample Delay

The slave uses the clk signal from the master as a strobe in order to drive MOSI/MISO signal data onto the serial bus. As there are round trip routing delays on clk signal from master, slave drive data output may delay at the same time as seen by the master, which has moved away from the normal sampling time. When the SPI is configured as a master, additional logic can be included in the design in order to delay the default sample time of the MISO signal. This additional logic can help to increase the maximum achievable frequency on the serial bus.

Figure 20-11 shows an example of how a routing delay on the MOSI signal can result in an incorrect MISO value at the default time when the master samples the port. Where, blue arrows indicate routing delay between master and slave devices; red line indicates sampling delay within slave from receiving clk to driing data out on MISO line.

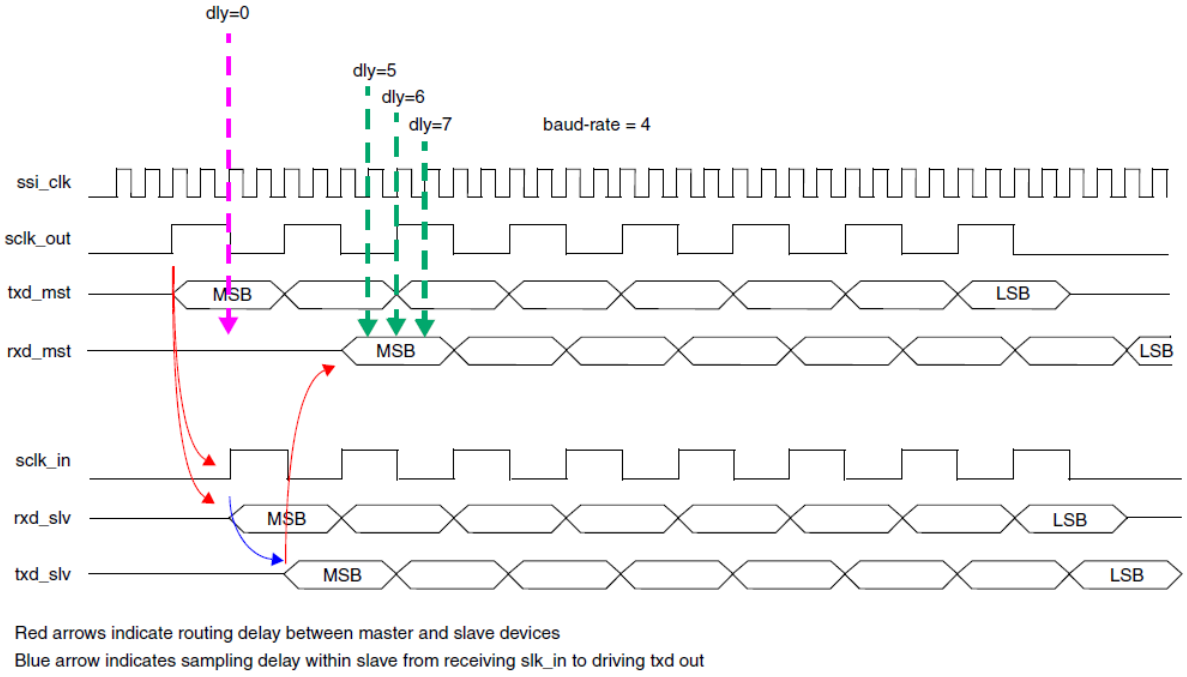


Figure 20-11 Effects of round trip routing delays on clk signal

When the RXD Sample Delay logic is included, user can dynamically program a delay value in order to move the sampling time of the rxd signal equal to a number of pclk cycles from the default.

The sample delay logic has a resolution of one pclk cycle. Software can "train" the serial bus by coding a loop that continually reads from the slave and increments the master's RXD Sample Delay value until the correct data is received by the master.

NOTE

The maximum number of ssi_clk cycles that can be used to delay the sampling of the rxd input is 4. If programmed delay value exceeds the maximum number, a zero (0) delay will be applied to the rxd sample.

20.2.3.1.2 Data Transfers

Data transfers are started by the serial-master device. When the SPI is enabled (SSI_EN=1), at least one valid data entry is present in the

All information provided in this document is subject to legal disclaimers.
© REALTEK 2024. All rights reserved.

transmit FIFO and a serial-slave device is selected. When actively transferring data, the busy flag (BUSY) in the status register (SR) is set. You must wait until the busy flag is cleared before attempting a new serial transfer.

20.2.3.1.3 Master SPI Serial Transfers

When the transfer mode is "transmit and receive" or "transmit only" (TMOD = 2'b00 or TMOD = 2'b01, respectively), transfers are terminated by the shift control logic when the transmit FIFO is empty. For continuous data transfers, you must ensure that the transmit FIFO buffer does not become empty before all the data have been transmitted. The transmit FIFO threshold level (TXFTLR) can be used to early interrupt (ssi_txe_intr) the processor indicating that the transmit FIFO buffer is nearly empty.

When a DMA is used for APB accesses, the transmit data level (DMATDLR) can be used to early request (dma_tx_req) the DMA Controller, indicating that the transmit FIFO is nearly empty. The FIFO can then be refilled with data to continue the serial transfer. The user may also write a block of data (at least two FIFO entries) into the transmit FIFO before enabling a serial slave. This ensures that serial transmission does not begin until the number of data-frames that make up the continuous transfer are present in the transmit FIFO.

When the transfer mode is "receive only" (TMOD = 2'b10), a serial transfer is started by writing one "dummy" data word into the transmit FIFO when a serial slave is selected. The MOSI output from the SPI is held at a constant logic level for the duration of the serial transfer. The transmit FIFO is popped only once at the beginning and may remain empty for the duration of the serial transfer. The end of the serial transfer is controlled by the 'number of data frames' (NDF) field in control register 1 (CTRLR1).

If, for example, you want to receive 24 data frames from a serial-slave peripheral, you should program the NDF field with the value 23; the receive logic terminates the serial transfer when the number of frames received is equal to the NDF value + 1. This transfer mode increases the bandwidth of the APB bus as the transmit FIFO never needs to be serviced during the transfer. The receive FIFO buffer should be read each time the receive FIFO generates a FIFO full interrupt request to prevent an overflow.

The receive FIFO threshold level (RXFTLR) can be used to give early indication that the receive FIFO is nearly full. When a DMA is used for APB accesses, the received data level (DMARDLR) can be used to early request (dma_rx_req) the DMA Controller, indicating that the receive FIFO is nearly full.

A typical software flow for completing an SPI serial transfer from the SPI serial master is outlined as follows:

(1) If the SPI is enabled, disable it by writing 0 to the SSI Enable register (SSIENR).

(2) Set up the SPI control registers for the transfer; these registers can be set in any order.
- Write Control Register 0 (CTRLR0). For SPI transfers, the serial clock polarity and serial clock phase parameters must be set identical to target slave device.
- If the transfer mode is receive-only, write Control Register 1 (CTRLR1) with the number of frames in the transfer minus 1; for example, if you want to receive four data frames, write this register with 3.
- Write the Baud Rate Select Register (BAUDR) to set the baud rate for the transfer.
- Write the Transmit and Receive FIFO Threshold Level registers (TXFTLR and RXFTLR) respectively to set FIFO threshold levels.
- Write the IMR register to set up interrupt masks.
- The Slave Enable Register (SER) can be written here to enable the target slave for selection. If a slave is enabled here, the transfer begins as soon as one valid data entry is present in the transmit FIFO. If no slaves are enabled prior to writing to the Data Register (DR), the transfer does not begin until a slave is enabled.

(3) Enable the SPI by writing 1 to the SSIENR register.

(4) Write data for transmission to the target slave into the transmit FIFO (write DR).
If no slaves were enabled in the SER register at this point, enable it now to begin the transfer.

(5) Poll the BUSY status to wait for the completion of the transfer. The BUSY status cannot be polled immediately.
If a transmit FIFO empty interrupt request is made, write the transmit FIFO (write DR). If a receive FIFO full interrupt request is made, read the receive FIFO (read DR).

(6) The transfer is stopped by the shift control logic when the transmit FIFO is empty. If the transfer mode is receive only (TMOD = 2'b10), the transfer is stopped by the shift control logic when the specified number of frames have been received. When the transfer is done, the BUSY status is reset to 0.

(7) If the transfer mode is not transmitted only (TMOD != 01), read the receive FIFO until it is empty.

(8) Disable the SPI by writing 0 to SSIENR.

20.2.3.2 Serial-Slave Mode

SPI slave enables serial communication with master peripheral devices. When the SPI is configured as a slave device, all serial transfers are initiated and controlled by the serial bus master.

The SPI slave can only operate in transmit and receive mode. That is, TMOD field in CTRLR0 register is invalid for SPI slave. If you do not want this device to respond with data, slave output can be disabled through SLV_OE bit in CTRLR0.

All information provided in this document is subject to legal disclaimers.
© REALTEK 2024. All rights reserved.

When the SPI serial slave is selected during configuration, it enables its MISO data onto the serial bus. All data transfers to and from the serial slave are regulated on the serial clock line (sclk_in), driven from the serial-master device. Data are propagated from the serial slave on one edge of the serial clock line and sampled on the opposite edge.

When the SPI serial slave is not selected, it must not interfere with data transfers between the serial-master and other serial-slave devices. When the serial slave is not selected, its MISO output is buffered, resulting in a high impedance drive onto the serial master MISO line.

The serial clock that regulates the data transfer is generated by the serial-master device and input to the SPI slave on sclk_in. The slave remains in an idle state until selected by the bus master. When not actively transmitting data, the slave must hold its MISO line in a high impedance state to avoid interference with serial transfers to other slave devices. The slave continues to transfer data to and from the master device as long as it is selected.

20.2.3.2.1 Slave SPI Serial Transfers

If the SPI slave transmits data to the master, you must ensure that data exists in the transmit FIFO before a transfer is initiated by the serial-master device. If the master initiates a transfer to the SPI slave when no data exists in the transmit FIFO, an error flag (TXE) is set in the SPI status register, and the previously transmitted data frame is resent on MISO. For continuous data transfers, you must ensure that the transmit FIFO buffer does not become empty before all the data have been transmitted. The transmit FIFO threshold level register (TXFTLR) can be used to early interrupt (ssi_txe_intr) the processor, indicating that the transmit FIFO buffer is nearly empty. When a DMA Controller is used for APB accesses, the DMA transmit data level register (DMATDLR) can be used to early request (dma_tx_req) the DMA Controller, indicating that the transmit FIFO is nearly empty. The FIFO can then be refilled with data to continue the serial transfer.

The receive FIFO buffer should be read each time the receive FIFO generates a FIFO full interrupt request to prevent an overflow. The receive FIFO threshold level register (RXFTLR) can be used to give early indication that the receive FIFO is nearly full. When a DMA Controller is used for APB accesses, the DMA receive data level register (DMARDLR) can be used to early request (dma_rx_req) the DMA controller, indicating that the receive FIFO is nearly full.

A typical software flow for completing a continuous serial transfer from a serial master to the SPI slave is described as follows:

(1) If the SPI slave is enabled, disable it by writing 0 to the SSI Enable register (SSIENR).
   a) Set up the SPI control registers for the transfer. These registers can be set in any order.
   b) Write CTRLR0 (for SPI transfers SCPH and SCPOL must be set identical to the master device).
   c) Write TXFTLR and RXFTLR to set FIFO threshold levels.
   d) Write the IMR register to set up interrupt masks.
(2) Enable the SPI slave by writing 1 to SSIENR.
(3) Write data to the data register (DR)
(4) The SPI slave is now ready for the serial transfer. The transfer begins when the SPI slave is selected by a serial-master device.
(5) When the transfer is underway, the BUSY status can be polled to return the transfer status. If a transmit FIFO empty interrupt request is made, write the transmit FIFO (write DR). If a receive FIFO full interrupt request is made, read the receive FIFO (read DR).
(6) The transfer ends when the serial master removes the select input to the SPI slave. When the transfer is completed, the BUSY status is reset to 0.
(7) Disable the SPI slave by writing 0 to SSIENR.

20.2.4 DMA Controller Interface

The SPI has optional built-in DMA capability which can be selected at configuration time; it has a handshaking interface to a DMA Controller to request and control transfers. The APB bus is used to perform the data transfer to or from the DMA.

The SPI uses two DMA channels, one for the transmit data and one for the received data. The SPI has these DMA registers:

- DMACR: Control register to enable DMA operation.
- DMATDLR: Register to set the transmit FIFO level at which a DMA request is made.
- DMARDLR: Register to set the receive FIFO level at which a DMA request is made.

To enable the DMA Controller interface on the SPI, you must write the DMA Control Register (DMACR). Writing a '1' into the TDMAE bit field of DMACR enables the SPI transmit handshaking interface. Writing a '1' into the RDMAE bit field of DMACR enables the SPI receive handshaking interface.

20.2.4.1 Transmit Watermark Level and Transmit FIFO Underflow

During SPI serial transfers, transmit FIFO requests are made to the DMA Controller whenever the number of entries in the transmit FIFO is less than or equal to the DMA Transmit Data Level Register (DMATDLR) value; this is known as the watermark level. The DMA Controller responds by writing a burst of data to the transmit FIFO buffer.

All information provided in this document is subject to legal disclaimers.
© REALTEK 2024. All rights reserved.

Data should be fetched from the DMA often enough for the transmit FIFO to perform serial transfers continuously; that is, when the FIFO begins to empty another DMA request should be triggered. Otherwise, the FIFO will run out of data (underflow). To prevent this condition, the user must set the watermark level correctly.

20.2.4.2 Receive Watermark Level and Receive FIFO Overflow

During SPI serial transfers, receive FIFO requests are made to the DMA Controller whenever the number of entries in the receive FIFO is at or above the DMA Receive Data Level Register; that is, DMARDLR+1. This is known as the watermark level. The DMA Controller responds by fetching a burst of data from the receive FIFO buffer.

Data should be fetched by the DMA often enough for the receive FIFO to accept serial transfers continuously; that is, when the FIFO begins to fill, another DMA transfer is requested. Otherwise, the FIFO will fill with data (overflow). To prevent this condition, the user must correctly set the watermark level.

20.3 Registers

This section describes the programmable registers of the SPI. The following table provides the details of the SPI memory map. All registers in the SPI are addressed at 32-bit boundaries to remain consistent with the AHB bus. Where the physical size of any register is less than 32-bit wide, the upper unused bits of the 32-bit boundary are reserved. Writing to these bits has no effect; reading from these bits returns 0.

The base addresses of SPI are:

- SPI0: 0x40118000
- SPI1: 0x40119000

Name	Address offset	Access	Description
REG_CTRLR0	000h	R/W	Control Register 0
REG_CTRLR1	004h	R/W	Control Register 1
REG_SSIENR	008h	R/W	SSI Enable Register
REG_SER	010h	R/W	Slave Enable Register
REG_BAUDR	014h	R/W	Baud Rate Select Register
REG_TXFTLR	018h	R/W	Transmit FIFO Threshold Level Register
REG_RXFTLR	01Ch	R/W	Receive FIFO Threshold Level Register
REG_TXFLR	020h	R	Transmit FIFO Level Register
REG_RXFLR	024h	R	Receive FIFO Level Register
REG_SR	028h	R	Status Register
REG_IMR	02Ch	R/W	Interrupt Mask Register
REG_ISR	030h	R	Interrupt Status Register
REG_RISR	034h	R	Raw Interrupt Status Register
REG_TXOICR	038h	R	Transmit FIFO Overflow Interrupt Clear Register
REG_RXOICR	03Ch	R	Receive FIFO Overflow Interrupt Clear Register
REG_RXUICR	040h	R	Receive FIFO Underflow Interrupt Clear Register
REG_MSTICR_FAEICR	044h	R	Multi-Master & Frame Alignment Error Interrupt Clear Register
REG_ICR	048h	R	Interrupt Clear Register
REG_DMACR	04Ch	R/W	DMA Control Register
REG_DMATDLR	050h	R/W	DMA Transmit Data Level
REG_DMARDLR	054h	R/W	DMA Receive Data Level
REG_TXUICR	058h	R	Transmit FIFO Underflow Interrupt Clear Register
REG_SSRICR	05Ch	R	SS_N Rising Edge Detect Interrupt Clear Register
REG_DR0	060h	R/W	Data Register 0
REG_DR1	064h	R/W	Data Register 1
REG_DR2	068h	R/W	Data Register 2
REG_DR3	06Ch	R/W	Data Register 3
REG_DR4	070h	R/W	Data Register 4
REG_DR5	074h	R/W	Data Register 5
REG_DR6	078h	R/W	Data Register 6
REG_DR7	07Ch	R/W	Data Register 7
REG_DR8	080h	R/W	Data Register 8
REG_DR9	084h	R/W	Data Register 9

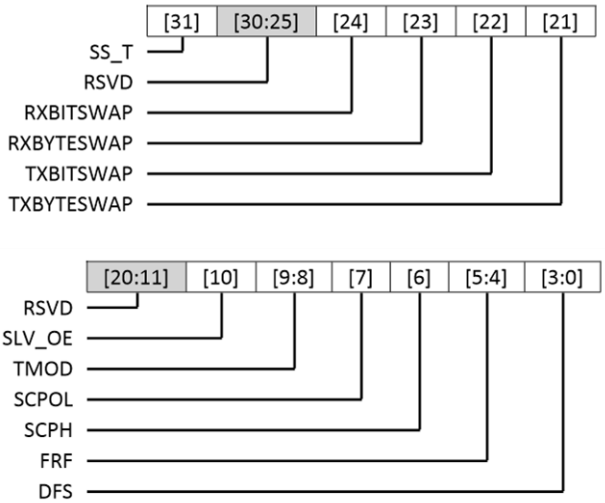
All information provided in this document is subject to legal disclaimers.
© REALTEK 2024. All rights reserved.

REG_DR10	088h	R/W	Data Register 10
REG_DR11	08Ch	R/W	Data Register 11
REG_DR12	090h	R/W	Data Register 12
REG_DR13	094h	R/W	Data Register 13
REG_DR14	098h	R/W	Data Register 14
REG_DR15	09Ch	R/W	Data Register 15
REG_DR16	0A0h	R/W	Data Register 16
REG_DR17	0A4h	R/W	Data Register 17
REG_DR18	0A8h	R/W	Data Register 18
REG_DR19	0ACh	R/W	Data Register 19
REG_DR20	0B0h	R/W	Data Register 20
REG_DR21	0B4h	R/W	Data Register 21
REG_DR22	B08h	R/W	Data Register 22
REG_DR23	0BCh	R/W	Data Register 23
REG_DR24	0C0h	R/W	Data Register 24
REG_DR25	0C4h	R/W	Data Register 25
REG_DR26	0C8h	R/W	Data Register 26
REG_DR27	0CCh	R/W	Data Register 27
REG_DR28	0D0h	R/W	Data Register 28
REG_DR29	0D4h	R/W	Data Register 29
REG_DR30	0D8h	R/W	Data Register 30
REG_DR31	0DCh	R/W	Data Register 31
REG_DR32	0E0h	R/W	Data Register 32
REG_DR33	0E4h	R/W	Data Register 33
REG_DR34	0E8h	R/W	Data Register 34
REG_DR35	0ECh	R/W	Data Register 35
REG_RX_SAMPLE_DLY	0F0h	R/W	Rxd Sample Delay Register
REG_SPI_DUMMY	100h	R	SPI Dummy Register

20.3.1 REG_CTRLR0

- **Name:** Control Register 0
- **Size:** 32 bits
- **Address offset:** 000h
- **Read/write access:** R/W

This register controls the serial data transfer. It is impossible to write to this register when the SPI is enabled. The SPI is enabled and disabled by writing to the SSIENR register.



Bit	Symbol	Access	Reset	Description
31	SS_T	R/W	0	When SCPH is 0 (Relevant only when the SPI is configured as a serial-master device) 0: ss_n_out doesn't toggle between successive frames

All information provided in this document is subject to legal disclaimers.
© REALTEK 2024. All rights reserved.

				1: ss_n_out does toggle between successive frames
30:25	RSVD	-	0	Reserved
24	RXBITSWAP	R/W	0	0: Order of receive bit doesn't swap 1: Order of receive bit does swap
23	RXBYTESWAP	R/W	0	0: Order of receive byte doesn't swap 1: Order of receive byte does swap
22	TXBITSWAP	R/W	0	0: Order of transmit bit doesn't swap 1: Order of transmit bit does swap
21	TXBYTESWAP	R/W	0	0: Order of transmit byte doesn't swap 1: Order of transmit byte does swap
20:11	RSVD	-	0	Reserved
10	SLV_OE	R/W	0	Slave Output Enable. Relevant only when the SPI is configured as a serial-slave device. When configured as a serial master, this bit field has no functionality. This bit enables or disables the setting of the ssi_oe_n output from the SPI serial slave. When SLV_OE = 1, the ssi_oe_n output can never be active. When the ssi_oe_n output controls the tri-state buffer on the txd output from the slave, a high impedance state is always present on the slave txd output when SLV_OE = 1. This is useful when the master transmits in broadcast mode (master transmits data to all slave devices). Only one slave may respond with data on the master rxd line. This bit is enabled after reset and must be disabled by software (when broadcast mode is used), if you do not want this device to respond with data. 0: Slave txd is enabled 1: Slave txd is disabled
9:8	TMOD	R/W	00	Transfer Mode. Relevant only when the SPI is configured as a serial-master device. Selects the mode of transfer for serial communication. This field does not affect the transfer duplicity. Only indicates whether the receive or transmit data are valid. In transmit-only mode, data received from the external device is not valid and is not stored in the receive FIFO memory; it is overwritten on the next transfer. In receive-only mode, transmitted data are not valid. After the first write to the transmit FIFO, the same word is retransmitted for the duration of the transfer. In transmit-and-receive mode, both transmit and receive data are valid. The transfer continues until the transmit FIFO is empty. Data received from the external device are stored into the receive FIFO memory, where it can be accessed by the host processor. 00: Transmit & Receive 01: Transmit Only 10: Receive Only
7	SCPOL	R/W	0	Serial Clock Polarity. Valid when the frame format (FRF) is set to Motorola SPI. Used to select the polarity of the inactive serial clock, which is held inactive when the SPI master is not actively transferring data on the serial bus. 0: Inactive state of serial clock is low 1: Inactive state of serial clock is high
6	SCPH	R/W	0	Serial Clock Phase. Valid when the frame format (FRF) is set to Motorola SPI. The serial clock phase selects the relationship of the serial clock with the slave select signal. When SCPH = 0, data are captured on the first edge of the serial clock. When SCPH = 1, the serial clock starts toggling one cycle after the slave select line is activated, and data are captured on the second edge of the serial clock. 0: Serial clock toggles in middle of the first data bit 1: Serial clock toggles at start of the first data bit
5:4	FRF	R/W	00	Frame Format. Selects which serial protocol transfers the data. 00: Motorola SPI Other: Reserved
3:0	DFS	R/W	0111	Data Frame Size. Selects the data frame length. When the data frame size is programmed to be less than 16 bits, the receive data are automatically right-justified by the receive logic, with the upper bits of the receive FIFO zero-padded. You must right-justify transmit data before writing into the transmit FIFO. The transmit logic ignores the upper unused bits when transmitting the data. For the DFS decode, refer to the following description: 0000/0001/0010: Reserved (undefined operation) 0011: 4-bit serial data transfer 0100: 5-bit serial data transfer 0101: 6-bit serial data transfer 0110: 7-bit serial data transfer

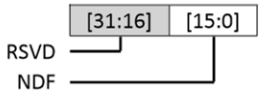
All information provided in this document is subject to legal disclaimers.
© REALTEK 2024. All rights reserved.

				0111: 8-bit serial data transfer 1000: 9-bit serial data transfer 1001: 10-bit serial data transfer 1010: 11-bit serial data transfer 1011: 12-bit serial data transfer 1100: 13-bit serial data transfer 1101: 14-bit serial data transfer 1110: 15-bit serial data transfer 1111: 16-bit serial data transfer

20.3.2 REG_CTRLR1

- **Name:** Control Register 1
- **Size:** 32 bits
- **Address offset:** 004h
- **Read/write access:** R/W

This register exists only when the SPI is configured as a master device. When the SPI is configured as a serial slave, writing to this location has no effect; reading from this location returns 0. Control register 1 controls the end of serial transfers when in receive-only mode. It is impossible to write to this register when the SPI is enabled. The SPI is enabled and disabled by writing to the SSIENR register.

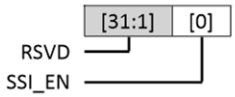


Bit	Symbol	Access	Reset	Description
31:16	RSVD	-	0	Reserved
15:0	NDF	R/W	0x0	Number of Data Frames. When TMOD = 10 or TMOD = 11, this register field sets the number of data frames to be continuously received by the SPI. The SPI continues to receive serial data until the number of data frames received is equal to this register value plus 1, which enables you to receive up to 64 KB of data in a continuous transfer. When the SPI is configured as a serial slave, the transfer continues for as long as the slave is selected. Therefore, this register serves no purpose and is not present when the SPI is configured as a serial slave.

20.3.3 REG_SSIENR

- **Name:** SSI Enable Register
- **Size:** 32 bits
- **Address offset:** 008h
- **Read/write access:** R/W

This register enables and disables the SPI.



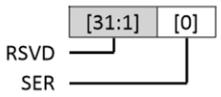
Bit	Symbol	Access	Reset	Description
31:1	RSVD	-	0	Reserved
0	SSI_EN	R/W	0	SPI Enable. Enables and disables all SPI operations. When disabled, all serial transfers are halted immediately. Transmit and receive FIFO buffers are cleared when the device is disabled. It is impossible to program some of the SPI control registers when enabled. When disabled, the ssi_sleep output is set (after delay) to inform the system that it is safe to remove the ssi_clk, thus saving power consumption in the system.

All information provided in this document is subject to legal disclaimers.
© REALTEK 2024. All rights reserved.

20.3.4 REG_SER

- **Name:** Slave Enable Register
- **Size:** 32 bits
- **Address offset:** 010h
- **Read/write access:** R/W

This register is valid only when the SPI is configured as a master device. When the SPI is configured as a serial slave, writing to this location has no effect; reading from this location returns 0. You cannot write to this register when SPI is busy.

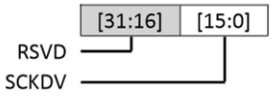


Bit	Symbol	Access	Reset	Description
31:1	RSVD	-	0	Reserved
0	SER	R/W	0	Slave Select Enable Flag. When this bit is set 1, the corresponding slave select line from the master is activated when a serial transfer begins. It should be noted that setting or clearing bits in this register have no effect on the corresponding slave select outputs until a transfer is started. Before beginning a transfer, you should enable the bit in this register that corresponds to the slave device with which the master wants to communicate. 1: Selected 0: Not selected

20.3.5 REG_BAUDR

- **Name:** Baud Rate Select Register
- **Size:** 32 bits
- **Address offset:** 014h
- **Read/write access:** R/W

This register is valid only when the SPI is configured as a master device. When the SPI is configured as a serial slave, writing to this location has no effect; reading from this location returns 0. The register derives the frequency of the serial clock that regulates the data transfer. The 16-bit field in this register defines the ssi_clk divider value. It is impossible to write to this register when the SPI is enabled. The SPI is enabled and disabled by writing to the SSIENR register.



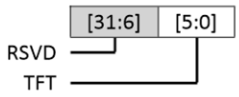
Bit	Symbol	Access	Reset	Description
31:16	RSVD	-	0	Reserved
15:0	SCKDV	R/W	0x0	SPI Clock Divider. The LSB for this field is always set to 0 and is unaffected by a write operation, which ensures an even value is held in this register. If the value is 0, the serial output clock (sclk_out) is disabled. The frequency of the sclk_out is derived from the equation: Fsclk_out =Fssi_clk/SCKDV, where SCKDV is any even value between 2 and 65534.

20.3.6 REG_TXFTLR

- **Name:** Transmit FIFO Threshold Level Register
- **Size:** 32 bits
- **Address offset:** 018h
- **Read/write access:** R/W

This register controls the threshold value for the transmit FIFO memory. It is impossible to write to this register when the SPI is enabled. The SPI is enabled and disabled by writing to the SSIENR register.

All information provided in this document is subject to legal disclaimers.
© REALTEK 2024. All rights reserved.

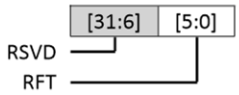


Bit	Symbol	Access	Reset	Description
31:6	RSVD	-	0	Reserved
5:0	TFT	R/W	0x0	Transmit FIFO Threshold. Controls the level of entries (or below) at which the transmit FIFO controller triggers an interrupt. The FIFO depth is 64; this register is sized to the number of address bits needed to access the FIFO. If you attempt to set this value greater than or equal to the depth of the FIFO, this field is not written and retains its current value. When the number of transmit FIFO entries is less than or equal to this value, the transmit FIFO empty interrupt is triggered. For TFT decode, refer to the following description: 000000: ssi_txe_intr is asserted when 0 data entry is present in transmit FIFO. 000001: ssi_txe_intr is asserted when 1 data entry is present in transmit FIFO. 000010: ssi_txe_intr is asserted when 2 data entries are present in transmit FIFO. 000011: ssi_txe_intr is asserted when 3 data entries are present in transmit FIFO. ... 111100: ssi_txe_intr is asserted when 60 data entries are present in transmit FIFO. 111101: ssi_txe_intr is asserted when 61 data entries are present in transmit FIFO. 111110: ssi_txe_intr is asserted when 62 data entries are present in transmit FIFO. 111111: ssi_txe_intr is asserted when 63 data entries are present in transmit FIFO.

20.3.7 REG_RXFTLR

- **Name:** Receive FIFO Threshold Level Register
- **Size:** 32 bits
- **Address offset:** 01Ch
- **Read/write access:** R/W

This register controls the threshold value for the receive FIFO memory. It is impossible to write to this register when the SPI is enabled. The SPI is enabled and disabled by writing to the SSIENR register.



Bit	Symbol	Access	Reset	Description
31:6	RSVD	-	0	Reserved
5:0	RFT	R/W	0x0	Receive FIFO Threshold. Controls the level of entries (or above) at which the receive FIFO controller triggers an interrupt. The FIFO depth is configurable in the range 2~64. This register is sized to the number of address bits needed to access the FIFO. If you attempt to set this value greater than the depth of the FIFO, this field is not written and retains its current value. When the number of receive FIFO entries is greater than or equal to this value + 1, the receive FIFO full interrupt is triggered. For RFT decode, refer to the following description: 000000: ssi_rxf_intr is asserted when 1 or more data entries are present in receive FIFO. 000001: ssi_rxf_intr is asserted when 2 or more data entries are present in receive FIFO. 000010: ssi_rxf_intr is asserted when 3 or more data entries are present in receive FIFO. 000011: ssi_rxf_intr is asserted when 4 or more data entries are present in receive FIFO. ... 111110: ssi_rxf_intr is asserted when 61 or more data entries are present in receive FIFO. 111101: ssi_rxf_intr is asserted when 62 or more data entries are present in receive FIFO. 111110: ssi_rxf_intr is asserted when 63 or more data entries are present in receive FIFO. 111111: ssi_rxf_intr is asserted when 64 or more data entries are present in receive FIFO.

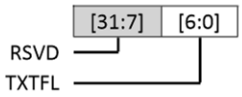
20.3.8 REG_TXFLR

- **Name:** Transmit FIFO Level Register
- **Size:** 32 bits

All information provided in this document is subject to legal disclaimers.
© REALTEK 2024. All rights reserved.

- **Address offset:** 020h
- **Read/write access:** R

This register contains the number of valid data entries in the transmit FIFO memory.

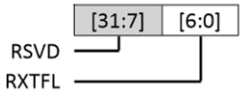


Bit	Symbol	Access	Reset	Description
31:7	RSVD	-	0	Reserved
6:0	TXTFL	R	0x0	Transmit FIFO Level. Contains the number of valid data entries in the transmit FIFO.

20.3.9 REG_RXFLR

- **Name:** Receive FIFO Level Register
- **Size:** 32 bits
- **Address offset:** 024h
- **Read/write access:** R

This register contains the number of valid data entries in the receive FIFO memory. This register can be read at any time.

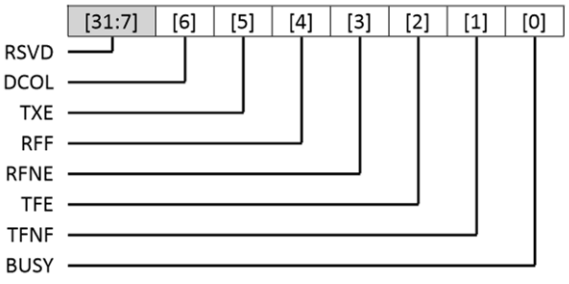


Bit	Symbol	Access	Reset	Description
31:7	RSVD	-	0	Reserved
6:0	RXTFL	R	0x0	Receive FIFO Level. Contains the number of valid data entries in the receive FIFO.

20.3.10 REG_SR

- **Name:** Status Register
- **Size:** 32 bits
- **Address offset:** 028h
- **Read/write access:** R

This is a read-only register used to indicate the current transfer status, FIFO status, and any transmission/reception errors that may have occurred. The status register may be read at any time. None of the bits in this register request an interrupt.



Bit	Symbol	Access	Reset	Description
31:7	RSVD	-	0	Reserved
6	DCOL	R	0	Data Collision Error. Relevant only when the SPI is configured as a master device. This bit is set if the SPI master is actively transmitting when another master selects this device as a slave. This informs the processor that the last data transfer was halted before completion. This bit is cleared when read. 0: No error 1: Transmit data collision error
5	TXE	R	0	Transmission Error. Set if the transmit FIFO is empty when a transfer is started. This bit can be set

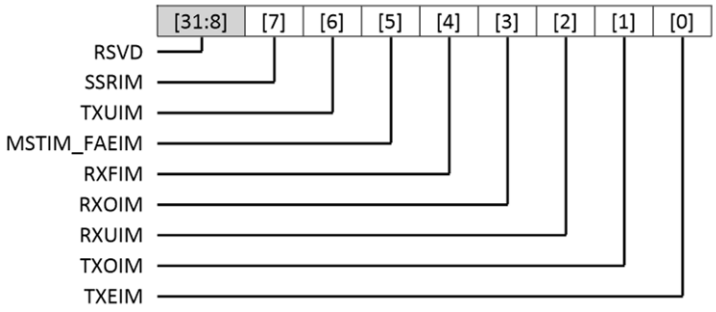
All information provided in this document is subject to legal disclaimers.
© REALTEK 2024. All rights reserved.

				only when the SPI is configured as a slave device. Data from the previous transmission is resent on the txd line. This bit is cleared when read. 0: No error 1: Transmission error
4	RFF	R	0	Receive FIFO Full. When the receive FIFO is completely full, this bit is set. When the receive FIFO contains one or more empty location, this bit is cleared. 0: Receive FIFO is not full 1: Receive FIFO is full
3	RFNE	R	0	Receive FIFO Not Empty. Set when the receive FIFO contains one or more entries and is cleared when the receive FIFO is empty. This bit can be polled by software to completely empty the receive FIFO. 0: Receive FIFO is empty 1: Receive FIFO is not empty
2	TFE	R	1	Transmit FIFO Empty. When the transmit FIFO is completely empty, this bit is set. When the transmit FIFO contains one or more valid entries, this bit is cleared. This bit field does not request an interrupt. 0: Transmit FIFO is not empty 1: Transmit FIFO is empty
1	TFNF	R	1	Transmit FIFO Not Full. Set when the transmit FIFO contains one or more empty locations, and is cleared when the FIFO is full. 0: Transmit FIFO is full 1: Transmit FIFO is not full
0	BUSY	R	0	SSI Busy Flag. When set, indicates that a serial transfer is in progress; when cleared indicates that the SPI is idle or disabled. 0: SPI is idle or disabled 1: SPI is actively transferring data

20.3.11 REG_IMR

- **Name:** Interrupt Mask Register
- **Size:** 32 bits
- **Address offset:** 02Ch
- **Read/write access:** R/W

This read/write register masks or enables all interrupts generated by the SPI.



Bit	Symbol	Access	Reset	Description
31:8	RSVD	-	0	Reserved
7	SSRIM	R/W	1	SS_N Rising Edge Detect Interrupt Mask. This bit field is present only if the SPI is configured as a serial-slave device. 0: ssi_ssr_intr is masked 1: ssi_ssr_intr is not masked
6	TXUIM	R/W	1	Transmit FIFO Underflow Interrupt Mask. This bit field is present only if the SPI is configured as a serial-slave device. 0: ssi_txu_intr is masked 1: ssi_txu_intr is not masked
5	MSTIM_FAEIM	R/W	1	When SPI is configured as serial-master, this bit field is present as Multi-Master Contention Interrupt Mask. 0: ssi_mst_intr interrupt is masked 1: ssi_mst_intr interrupt is not masked When SPI is configured as serial-slave, this bit field is present as Frame Alignment Interrupt

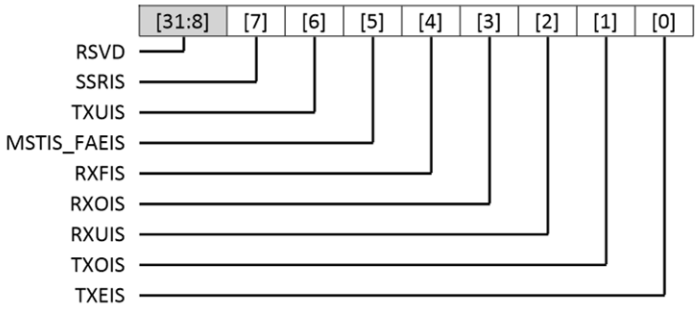
All information provided in this document is subject to legal disclaimers.
© REALTEK 2024. All rights reserved.

				Mask. 0: ssi_fae_intr interrupt is masked 1: ssi_fae_intr interrupt is not masked
4	RXFIM	R/W	1	Receive FIFO Full Interrupt Mask. 0: ssi_rxf_intr interrupt is masked 1: ssi_rxf_intr interrupt is not masked
3	RXOIM	R/W	1	Receive FIFO Overflow Interrupt Mask. 0: ssi_rxo_intr interrupt is masked 1: ssi_rxo_intr interrupt is not masked
2	RXUIM	R/W	1	Receive FIFO Underflow Interrupt Mask. 0: ssi_rxu_intr interrupt is masked 1: ssi_rxu_intr interrupt is not masked
1	TXOIM	R/W	1	Transmit FIFO Overflow Interrupt Mask. 0: ssi_txo_intr interrupt is masked 1: ssi_txo_intr interrupt is not masked
0	TXEIM	R/W	1	Transmit FIFO Empty Interrupt Mask. 0: ssi_txe_intr interrupt is masked 1: ssi_txe_intr interrupt is not masked

20.3.12 REG_ISR

- **Name:** Interrupt Status Register
- **Size:** 32 bits
- **Address offset:** 030h
- **Read/write access:** R

This register reports the status of the SPI interrupts after they have been masked.



Bit	Symbol	Access	Reset	Description
31:8	RSVD	-	0	Reserved
7	SSRIS	R	0	SS_N Rising Edge Detect Interrupt Status. This bit field is present only if the SPI is configured as a serial-slave device. 0: ssi_ssr_intr interrupt is not active after masking 1: ssi_ssr_intr interrupt is active after masking
6	TXUIS	R	0	Transmit FIFO Under Flow Interrupt Status. This bit field is present only if the SPI is configured as a serial-slave device. 0: ssi_txu_intr interrupt is not active after masking 1: ssi_txu_intr interrupt is active after masking
5	MSTIS_FAEIS	R	0	When SPI is configured as serial-master, this bit field is present as Multi-Master Contention Interrupt Status. 0: ssi_mst_intr interrupt is not active after masking 1: ssi_mst_intr interrupt is active after masking When SPI is configured as serial-slave, this bit field is present as Frame Alignment Interrupt Status. 0: ssi_fae_intr interrupt not active after masking 1: ssi_fae_intr interrupt is active after masking
4	RXFIS	R	0	Receive FIFO Full Interrupt Status. 0: ssi_rxf_intr interrupt is not active after masking 1: ssi_rxf_intr interrupt is full after masking

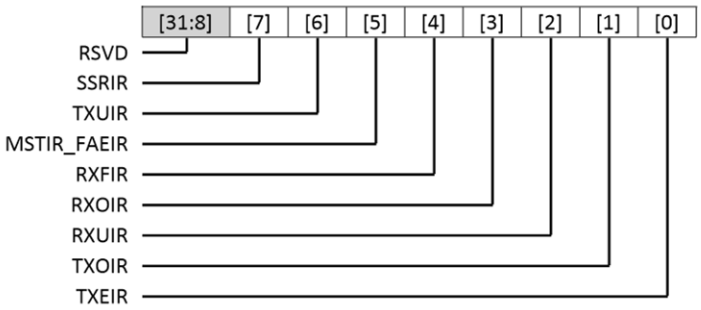
All information provided in this document is subject to legal disclaimers.
© REALTEK 2024. All rights reserved.

3	RXOIS	R	0	Receive FIFO Overflow Interrupt Status. 0: ssi_rxo_intr interrupt is not active after masking 1: ssi_rxo_intr interrupt is active after masking
2	RXUIS	R	0	Receive FIFO Underflow Interrupt Status. 0: ssi_rxu_intr interrupt is not active after masking 1: ssi_rxu_intr interrupt is active after masking
1	TXOIS	R	0	Transmit FIFO Overflow Interrupt Status. 0: ssi_txo_intr interrupt is not active after masking 1: ssi_txo_intr interrupt is active after masking
0	TXEIS	R	0	Transmit FIFO Empty Interrupt Status. 0: ssi_txe_intr interrupt is not active after masking 1: ssi_txe_intr interrupt is active after masking

20.3.13 REG_RISR

- **Name:** Raw Interrupt Status Register
- **Size:** 32 bits
- **Address offset:** 034h
- **Read/write access:** R

This read-only register reports the status of the SPI interrupts prior to masking.



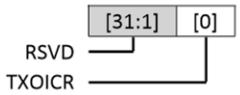
Bit	Symbol	Access	Reset	Description
31:8	RSVD	-	0	Reserved
7	SSRIR	R	0	SS_N Rising Edge Detect Raw Interrupt Status. This bit field is present only if the SPI is configured as a serial-slave device. 0: ssi_ssr_intr interrupt is not active prior to masking 1: ssi_ssr_intr interrupt is active prior to masking
6	TXUIR	R	0	Transmit FIFO Under Flow Raw Interrupt Status. This bit field is present only if the SPI is configured as a serial-slave device. 0: ssi_txu_intr interrupt is not active prior to masking 1: ssi_txu_intr interrupt is active prior to masking
5	MSTIR_FAEIR	R	0	When SPI is configured as serial-master, this bit field is present as Multi-Master Contention Raw Interrupt Status. 0: ssi_mst_intr interrupt is not active prior to masking 1: ssi_mst_intr interrupt is active prior to masking When SPI is configured as serial-slave, this bit field is present as Frame Alignment Error Raw Interrupt Status. 0: ssi_fae_intr interrupt not active prior to masking 1: ssi_fae_intr interrupt is active prior to masking
4	RXFIR	R	0	Receive FIFO Full Raw Interrupt Status. 0: ssi_rxf_intr interrupt is not active prior to masking 1: ssi_rxf_intr interrupt is active prior to masking
3	RXOIR	R	0	Receive FIFO Overflow Raw Interrupt Status. 0: ssi_rxo_intr interrupt is not active prior to masking 1: ssi_rxo_intr interrupt is active prior to masking
2	RXUIR	R	0	Receive FIFO Underflow Raw Interrupt Status. 0: ssi_rxu_intr interrupt is not active prior to masking 1: ssi_rxu_intr interrupt is active prior to masking

All information provided in this document is subject to legal disclaimers.
© REALTEK 2024. All rights reserved.

1	TXOIR	R	0	Transmit FIFO Overflow Raw Interrupt Status. 0: ssi_txo_intr interrupt is not active prior to masking 1: ssi_txo_intr interrupt is active prior to masking
0	TXEIR	R	0	Transmit FIFO Empty Raw Interrupt Status. 0: ssi_txe_intr interrupt is not active prior to masking 1: ssi_txe_intr interrupt is active prior to masking

20.3.14 REG_TXOICR

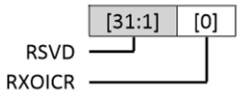
- **Name:** Transmit FIFO Overflow Interrupt Clear Register
- **Size:** 32 bits
- **Address offset:** 038h
- **Read/write access:** R



Bit	Symbol	Access	Reset	Description
31:1	RSVD	-	0	Reserved
0	TXOICR	R	0	Clear Transmit FIFO Overflow Interrupt. This register reflects the status of the interrupt. A read from this register clears the ssi_txo_intr interrupt; writing has no effect.

20.3.15 REG_RXOICR

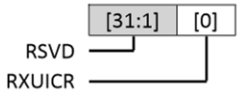
- **Name:** Receive FIFO Overflow Interrupt Clear Register
- **Size:** 32 bits
- **Address offset:** 03Ch
- **Read/write access:** R



Bit	Symbol	Access	Reset	Description
31:1	RSVD	-	0	Reserved
0	RXOICR	R	0	Clear Receive FIFO Overflow Interrupt. This register reflects the status of the interrupt. A read from this register clears the ssi_rxo_intr interrupt; writing has no effect.

20.3.16 REG_RXUICR

- **Name:** Receive FIFO Underflow Interrupt Clear Register
- **Size:** 32 bits
- **Address offset:** 040h
- **Read/write access:** R



Bit	Symbol	Access	Reset	Description
31:1	RSVD	-	0	Reserved
0	RXUICR	R	0	Clear Receive FIFO Underflow Interrupt. This register reflects the status of the interrupt. A read from this register clears the ssi_rxu_intr interrupt; writing has no effect.

All information provided in this document is subject to legal disclaimers.
© REALTEK 2024. All rights reserved.

20.3.17 REG_MSTICR_FAEICR

- **Name:** Multi-Master & Frame Alignment Error Interrupt Clear Register
- **Size:** 32 bits
- **Address offset:** 044h
- **Read/write access:** R

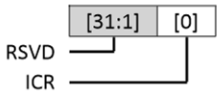
Serial-master: MSTICR is not used because ss_in_n is tie 1 for Motorola-SPI, ssi_mst_intr is not used.



Bit	Symbol	Access	Reset	Description
31:1	RSVD	-	0	Reserved
0	MSTICR_FAEICR	R	0	Multi-Master Interrupt Clear Register/Frame Alignment Error Interrupt Clear Register. When SPI is configured as serial-master, this bit field is used to Clear Multi-Master Contention Interrupt. A read from this register clears the ssi_mst_intr interrupt; writing has no effect. When SPI is configured as serial-slave, this bit field is used to Clear Frame Alignment Interrupt. A read from this register clears the ssi_fae_intr interrupt; writing has no effect.

20.3.18 REG_ICR

- **Name:** Interrupt Clear Register
- **Size:** 32 bits
- **Address offset:** 048h
- **Read/write access:** R

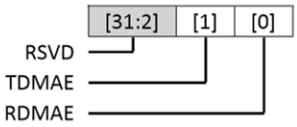


Bit	Symbol	Access	Reset	Description
31:1	RSVD	-	0	Reserved
0	ICR	R	0	Clear Interrupt. This register is set if any of the interrupts below are active. A read clears the ssi_txo_intr, ssi_rxu_intr, ssi_rxo_intr, and the ssi_mst_intr/ssi_fae_intr interrupts. Writing to this register has no effect.

20.3.19 REG_DMACR

- **Name:** DMA Control Register
- **Size:** 32 bits
- **Address offset:** 04Ch
- **Read/write access:** R/W

This register is only valid when SPI is configured with a set of DMA Controller interface signals.



Bit	Symbol	Access	Reset	Description
31:2	RSVD	-	0	Reserved
1	TDMAE	R/W	0	Transmit DMA Enable. This bit enables/disables the transmit FIFO DMA channel. 0: Transmit DMA disabled

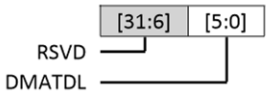
All information provided in this document is subject to legal disclaimers.
© REALTEK 2024. All rights reserved.

				1: Transmit DMA enabled
0	RDMAE	R/W	0	Receive DMA Enable. This bit enables/disables the receive FIFO DMA channel 0: Receive DMA disabled 1: Receive DMA enabled

20.3.20 REG_DMATDLR

- **Name:** DMA Transmit Data Level
- **Size:** 32 bits
- **Address offset:** 050h
- **Read/write access:** R/W

This register is only valid when the SPI is configured with a set of DMA interface signals.

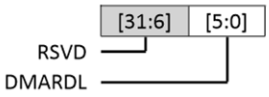


Bit	Symbol	Access	Reset	Description
31:6	RSVD	-	0	Reserved
5:0	DMATDL	R/W	0x0	Transmit Data Level. This bit field controls the level at which a DMA request is made by the transmit logic. It is equal to the watermark level; that is, the dma_tx_req signal is generated when the number of valid data entries in the transmit FIFO is equal to or below this field value, and TDMAE = 1. For DMATDL decode, refer to the following description: 000000: dma_tx_req is asserted when 0 data entry is present in transmit FIFO. 000001: dma_tx_req is asserted when 1 data entry is present in transmit FIFO. 000010: dma_tx_req is asserted when 2 data entries are present in transmit FIFO. 000011: dma_tx_req is asserted when 3 data entries are present in transmit FIFO. ... 111100: dma_tx_req is asserted when 60 data entries are present in transmit FIFO. 111101: dma_tx_req is asserted when 61 data entries are present in transmit FIFO. 111110: dma_tx_req is asserted when 62 data entries are present in transmit FIFO. 111111: dma_tx_req is asserted when 63 data entries are present in transmit FIFO.

20.3.21 REG_DMARDLR

- **Name:** DMA Receive Data Level
- **Size:** 32 bits
- **Address offset:** 054h
- **Read/write access:** R/W

This register is only valid when SPI is configured with a set of DMA interface signals.



Bit	Symbol	Access	Reset	Description
31:6	RSVD	-	0	Reserved
5:0	DMARDL	R/W	0x0	Receive Data Level. This bit field controls the level at which a DMA request is made by the receive logic. The watermark level = DMARDL+1; that is, dma_rx_req is generated when the number of valid data entries in the receive FIFO is equal to or above this field value + 1, and RDMAE=1. For DMARDL decode, refer to the following description: 000000: dma_rx_req is asserted when 1 or more data entries are present in transmit FIFO. 000001: dma_rx_req is asserted when 2 or more data entries are present in transmit FIFO. 000010: dma_rx_req is asserted when 3 or more data entries are present in transmit FIFO. 000011: dma_rx_req is asserted when 4 or more data entries are present in transmit FIFO. ... 111100: dma_rx_req is asserted when 61 or more data entries are present in transmit FIFO.

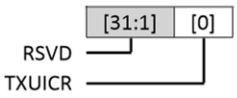
All information provided in this document is subject to legal disclaimers.
© REALTEK 2024. All rights reserved.

				111101: dma_rx_req is asserted when 62 or more data entries are present in transmit FIFO. 111110: dma_rx_req is asserted when 63 or more data entries are present in transmit FIFO. 111111: dma_rx_req is asserted when 64 data entries are present in transmit FIFO.

20.3.22 REG_TXUICR

- **Name:** Transmit FIFO Underflow Interrupt Clear Register
- **Size:** 32 bits
- **Address offset:** 058h
- **Read/write access:** R

This register is present only if SPI is configured as serial-slave.

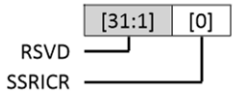


Bit	Symbol	Access	Reset	Description
31:1	RSVD	-	0	Reserved
0	TXUICR	R	0	When SPI is configured as serial-slave, this register is used to Clear Transmit FIFO Underflow Interrupt. This register reflects the status of the interrupt. A read from this register clears the ssi_txu_intr interrupt; writing has no effect.

20.3.23 REG_SSRICR

- **Name:** SS_N Rising Edge Detect Interrupt Clear Register
- **Size:** 32 bits
- **Address offset:** 05Ch
- **Read/write access:** R

This register is present only if SPI is configured as serial-slave.



Bit	Symbol	Access	Reset	Description
31:1	RSVD	-	0	Reserved
0	SSRICR	R	0	When SPI is configured as serial-slave, this register is used to Clear SS_N Rinsing Edge Detect Interrupt. This register reflects the status of the interrupt. A read from this register clears the ssi_ssr_intr interrupt; writing has no effect.

20.3.24 REG_DRx

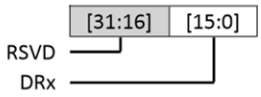
- **Name:** Data Register x (x={0, 1, 2, 3, 4, 5, 6, 7, 8, 9, 10, 11, 12, 13, 14, 15, 16, 17, 18, 19, 20, 21, 22, 23, 24, 25, 26, 27, 28, 29, 30, 31, 32, 33, 34, 35})
- **Size:** 32 bits
- **Address offset:**
  - **REG_DR0:** 060h
  - **REG_DR1:** 064h
  - **REG_DR2:** 068h
  - **REG_DR3:** 06Ch
  - **REG_DR4:** 070h
  - **REG_DR5:** 074h
  - **REG_DR6:** 078h
  - **REG_DR7:** 07Ch
  - **REG_DR8:** 080h
  - **REG_DR9:** 084h

All information provided in this document is subject to legal disclaimers.
© REALTEK 2024. All rights reserved.

- **REG_DR10:** 088h
- **REG_DR11:** 08Ch
- **REG_DR12:** 090h
- **REG_DR13:** 094h
- **REG_DR14:** 098h
- **REG_DR15:** 09Ch
- **REG_DR16:** 0A0h
- **REG_DR17:** 0A4h
- **REG_DR18:** 0A8h
- **REG_DR19:** 0ACh
- **REG_DR20:** 0B0h
- **REG_DR21:** 0B4h
- **REG_DR22:** B08h
- **REG_DR23:** 0BCh
- **REG_DR24:** 0C0h
- **REG_DR25:** 0C4h
- **REG_DR26:** 0C8h
- **REG_DR27:** 0CCh
- **REG_DR28:** 0D0h
- **REG_DR29:** 0D4h
- **REG_DR30:** 0D8h
- **REG_DR31:** 0DCh
- **REG_DR32:** 0E0h
- **REG_DR33:** 0E4h
- **REG_DR34:** 0E8h
- **REG_DR35:** 0ECh

- **Read/write access:** R/W

The SPI data register is a 16-bit read/write buffer for the transmit/receive FIFOs. When the register is read, data in the receive FIFO buffer is accessed. When it is written to, data are moved into the transmit FIFO buffer; a write can occur only when SSI_EN = 1. FIFOs are reset when SSI_EN = 0. Note1: If the Data Register (DR) is accessed from an AHB master (such as a DMA controller or a processor), the AHB transfer type may be a burst. During AHB burst transfers, the address increments after each beat of the burst. To facilitate an AHB burst, read, or write operation to the transmit or receive FIFO, the Date Register occupies thirty-six 32-bit address locations of memory map. Each of the 16-bit address locations are aliased to the DR; single accesses to the DR may use any of the 16-bit address locations. Note2: The DR register in the SPI occupies thirty-six 32-bit address locations of the memory map to facilitate AHB burst transfers. Writing to any of these address locations has the same effect as pushing the data from the pwdata bus into the transmit FIFO. Reading from any of these locations has the same effect as popping data from the receive FIFO onto the prdata bus. The FIFO buffers on the SPI are not addressable.



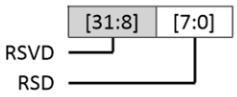
Bit	Symbol	Access	Reset	Description
31:16	RSVD	-	0	Reserved
15:0	DRx	R/W	0x0	Data Register. When writing to this register, you must right-justify the data. Read data are automatically right-justified. Read: Receive FIFO buffer Write: Transmit FIFO buffer

20.3.25 REG_RX_SAMPLE_DLY

- **Name:** Rxd Sample Delay Register
- **Size:** 32 bits
- **Address offset:** 0F0h
- **Read/write access:** R/W

This register is valid only when the SPI is configured as serial-master. This register controls the number of ssi_clk cycles that are delayed—from the default sample time—before the actual sample of the rxd input signal occurs. It is impossible to write to this register when the SPI is enabled; the SPI is enabled and disabled by writing to the SSIENR register. Note: If this register is programmed with a value that exceeds the depth of the internal shift registers (SSI_RX_DLY_SR_DEPTH), a zero (0) delay will be applied to the rxd sample.

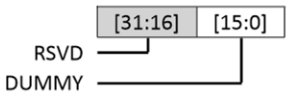
All information provided in this document is subject to legal disclaimers.
© REALTEK 2024. All rights reserved.



Bit	Symbol	Access	Reset	Description
31:8	RSVD	-	0	Reserved
7:0	RSD	R/W	0x0	Receive Data (rxd) Sample Delay. This register is used to delay the sample of the rxd input signal. Each value represents a single ssi_clk delay on the sample of the rxd signal.

20.3.26 REG_SPI_DUMMY

- **Name:** SPI Dummy Register
- **Size:** 32 bits
- **Address offset:** 100h
- **Read/write access:** R



Bit	Symbol	Access	Reset	Description
31:16	RSVD	-	0	Reserved
15:0	DUMMY	R	0x0	Reserved for HW

All information provided in this document is subject to legal disclaimers.
© REALTEK 2024. All rights reserved.

21 Inter-integrated Circuit (I2C) Interface

21.1 Introduction

This chapter describes the I2C interface peripherals, referred to as I2C.

The I2C bus interface handles communications between the RTL8721Dx and the serial I2C bus. It controls all I2C bus-specific sequencing, protocol, arbitration and timing. The design of I2C aims on sensor hub application in low-power or battery-powered productions. Essential features of I2C bus protocol should be provided for acquiring or controlling external sensor data.

21.2 Features

The I2C has the following features:

- Two-wire I2C serial interface – a serial data line (SDA) and a serial clock (SCL)
- Three speed modes:
  - Standard Speed (SS), up to 100Kbps
  - Fast Speed (FS), up to 400Kbps
  - High Speed (HS), up to 3.4Mbps
- I2C interface: x3
  - I2C0 is on LS platform (LS_APB_CLK, 10MHz), only supports Standard Speed and Fast Speed modes.
  - I2C1 and I2C2 are on HP platform (HP_AHB_CLK, 100MHz), support all speed modes.
- Master or Slave I2C operation
- Transmitter or Receiver
- Transmit and receive FIFOs with depth of 16 and width of 12-bit
- Multi-master ability including bus arbitration scheme
- Clock stretch in master/slave mode
- 7-bit or 10-bit addressing mode, 7-bit or 10-bit combined format transfer
- Manual START/RESTART/STOP bit control
- Supports General Call, NULL DATA, START BYTE transfer protocol
- Component parameters for configurable software driver support (programmable SDA hold time, slave address, SCL duty cycle, etc.)
- Filter to eliminate the glitches on the signal of SDA and SCL, programmable digital noise Filter
- Status flags (Bus busy flag, activity flag, FIFO status flag, etc.) and Error flags (arbitration lost, acknowledge failure, etc.)
- Slave Mode Dual Own Address
  - Slave 1 supports 7-bit or 10-bit address mode
  - Slave 2 only supports 7-bit address mode
- Operation mode:
  - Polling
  - Interrupt

21.3 Functional Description

This section describes the functional behavior of the I2C in more details.

21.3.1 Overview

The I2C bus is a two-wire serial interface, consisting of a serial data line (SDA) and a serial clock (SCL). These wires carry information between the devices connected to the bus. Each device is recognized by a unique address and can operate as either a “transmitter” or “receiver,” depending on the function of the device. Devices can also be considered as masters or slaves when performing data transfers. A master is a device that initiates a data transfer on the bus and generates the clock signals to permit that transfer. At that time, any device addressed is considered a slave.

NOTE

The I2C must only be programmed to operate in either master or slave mode only. Operating as master and slave simultaneously is not supported.

Any I2C device can be attached to an I²C-bus and every device can talk with any master, passing information back and forth. There needs to be at least one master (such as a microcontroller or DSP) on the bus but there can be multiple masters, which require them to arbitrate for ownership. Multiple masters and arbitration are explained later in this section.

All information provided in this document is subject to legal disclaimers.
© REALTEK 2024. All rights reserved.

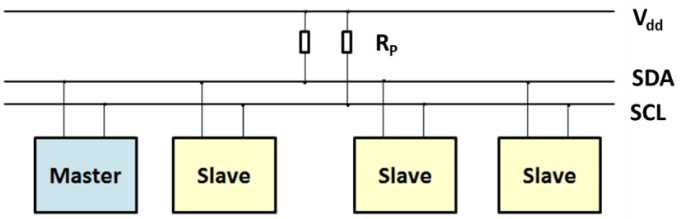


Figure 21-1 An example schematic with one master and three slave nodes

21.3.2 I2C Terminology

The following terms are used throughout this chapter and are defined as below.

21.3.2.1 I2C Bus Terms

The following terms relate to what is the role of the I2C device and how it interacts with other I2C devices on the bus.

- Transmitter – the device that sends data to the bus. A transmitter can either be a device that initiates the data transmission to the bus (a master-transmitter) or responds to a request from the master to send data to the bus (a slave-transmitter).
- Receiver – the device that receives data from the bus. A receiver can either be a device that receives data on its own request (a master-receiver) or in response to a request from the master (a slave-receiver).
- Master -– the component that initializes a transfer (START command), generates the clock (SCL) signal and terminates the transfer (STOP command). A master can be either a transmitter or a receiver.
- Slave – the device addressed by the master. A slave can be either receiver or transmitter. These concepts are illustrated in *Figure 21-2*.
- Multi-master – the ability for more than one master to co-exist on the bus at the same time without collision or data loss.
- Arbitration – the predefined procedure that authorizes only one master at a time to take control of the bus.
- Synchronization – the predefined procedure that synchronizes the clock signals provided by two or more masters.
- SDA – data signal line (Serial Data)
- SCL – clock signal line (Serial Clock)

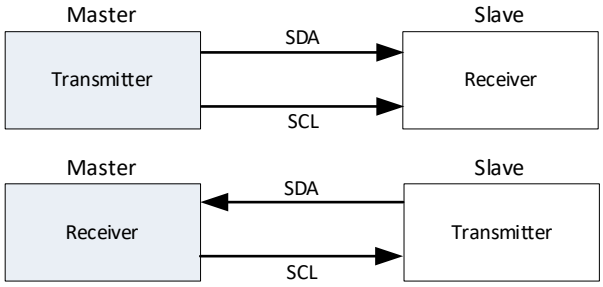


Figure 21-2 Master/Slave and Transmitter/Receiver relationships

21.3.2.2 Bus Transfer Terms

The following terms are specific to data transfers that occur to/from the I2C bus.

- START (RESTART) – data transfer begins with a START or RESTART condition. The level of the SDA data line changes from high to low, while the SCL clock line remains high. When this occurs, the bus becomes busy.
- STOP – data transfer is terminated by a STOP condition. This occurs when the level on the SDA data line passes from the low state to the high state, while the SCL clock line remains high. When the data transfer has been terminated, the bus is free or idle once again. The bus stays busy if a RESTART is generated instead of a STOP condition.

NOTE

START and RESTART conditions are functionally identical.

21.3.3 I2C Behavior

The I2C can be controlled via software to be either:

- An I2C master only, communicating with other I2C slaves; OR
- An I2C slave only, communicating with one more I2C masters.

All information provided in this document is subject to legal disclaimers.
© REALTEK 2024. All rights reserved.

The master is responsible for generating the clock and controlling the transfer of data. The slave is responsible for either transmitting or receiving data to/from the master. The acknowledgement of data is sent by the device that is receiving data, which can be either a master or a slave. As mentioned previously, the I2C protocol also allows multiple masters to reside on the I2C bus and uses an arbitration procedure to determine bus ownership.

Each slave has a unique address that is determined by the system designer. When a master wants to communicate with a slave, the master transmits a START/RESTART condition that is then followed by the slave's address and a control bit (R/W) to determine if the master wants to transmit data or receive data from the slave. The slave then sends an acknowledge (ACK) pulse after the address.

If the master (master-transmitter) is writing to the slave (slave-receiver), the receiver gets one byte of data. This transaction continues until the master terminates the transmission with a STOP condition. If the master is reading from a slave (master-receiver), the slave transmits (slave-transmitter) a byte of data to the master, and the master then acknowledges the transaction with the ACK pulse. This transaction continues until the master terminates the transmission by not acknowledging (NACK) the transaction after the last byte is received, and then the master issues a STOP condition or addresses another slave after issuing a RESTART condition. This behavior is illustrated in *Figure 21-3*.

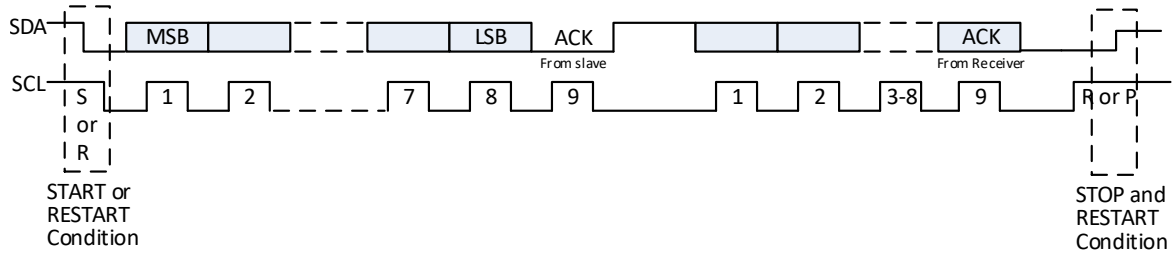


Figure 21-3 Data transfer on the I2C bus

The I2C is a synchronous serial interface. The SDA line is a bidirectional signal and changes only while the SCL line is low, except for STOP, START, and RESTART conditions. The output drivers are open-drain or open-collector to perform wire-AND functions on the bus. The maximum number of devices on the bus is limited by only the maximum capacitance specification of 400pF. Data is transmitted in byte packages.

21.3.3.1 START and STOP Generation

Refer to *Figure 21-13* for IC_DATA_CMD structure. When operating as an I2C master, putting data into the transmit FIFO causes the I2C to generate a START condition on the I2C bus. Writing a 1 to IC_DATA_CMD[9] causes the hardware to generate a STOP condition on the I2C bus; a STOP condition is not issued if this bit is not set, even if the transmit FIFO is empty. Writing a 1 to IC_DATA_CMD[10] causes the hardware to hold bus after the current data is transmitted and generate a RESTART condition when the next data in FIFO is ready to be transmitted on bus.

When operating as a slave, the I2C does not generate START and STOP conditions, as per the protocol. However, if a read request is made to the I2C, it holds the SCL line low until read data has been supplied to it. This stalls the I2C bus until read data is provided to the slave I2C, or the I2C slave is disabled by writing a 0 to bit0 of the REG_IC_ENABLE.

21.3.3.2 Combined Formats

Combined format transactions: the data send direction may change during a transfer. For example, a master may send data and then receive data or vice versa without release the I2C bus.

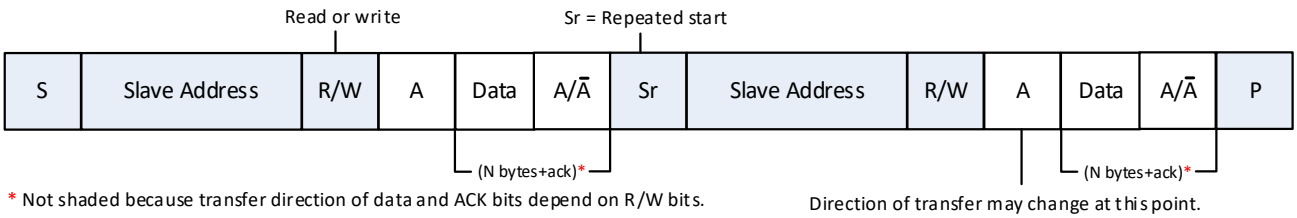


Figure 21-4 Combined format

The I2C supports combined format transactions for both 7- or 10-bit addressing modes. The I2C does not support mixed address—that is, a 7-bit address transaction followed by a 10-bit address transaction or vice versa—in combined format transaction.

To initiate combined format transfers, IC_CON.IC_RESTART_EN should be set to 1. With this value set and operating as a master, when the I2C completes an I2C transfer, it checks the transmit FIFO and executes the next transfer. If the direction of this transfer differs from the previous transfer, the combined format is used to issue the transfer. If the transmit FIFO is empty when the current I2C transfer completes,

All information provided in this document is subject to legal disclaimers.
© REALTEK 2024. All rights reserved.

IC_DATA_CMD[9] is checked.

- If set to 1, a STOP bit is issued.
- If set to 0, the SCL is hold low until the next command is written to the transmit FIFO.

21.3.4 I2C Protocols

21.3.4.1 START and STOP Conditions

When the bus is idle, both the SCL and SDA signals are pulled high through external pull-up resistors on the bus. When the master wants to start a transmission on the bus, the master issues a START condition. This is defined to be a high-to-low transition of the SDA signal while SCL is 1. When the master wants to terminate the transmission, the master issues a STOP condition. This is defined to be a low-to-high transition of the SDA line while SCL is 1. *Figure 21-5* shows the timing of the START and STOP conditions. When data is being transmitted on the bus, the SDA line must be stable when SCL is 1.

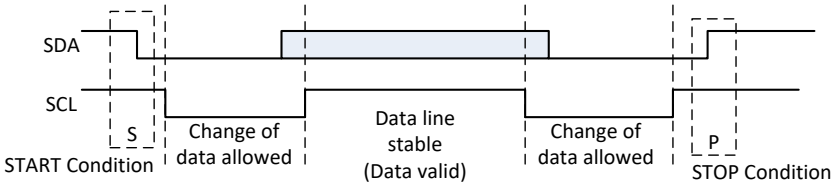


Figure 21-5 START and STOP conditions

21.3.4.2 Addressing Slave Protocol

There are two address formats:

- 7-bit address format
- 10-bit address format

21.3.4.2.1 7-bit Address Format

During the 7-bit address format, the first seven bits (bit[7:1]) of the first byte set the slave address and the LSB bit (bit[0]) is the R/W bit as shown in *Figure 21-6*. When bit[0] is set to 0, the master write data to the slave. When bit[0] is set to 1, the master read data from the slave.

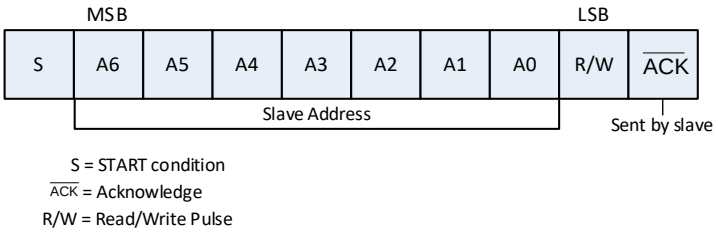


Figure 21-6 7-bit address format

21.3.4.2.2 10-bit Address Format

During 10-bit addressing, two bytes are transferred to set the 10-bit address. The transfer of the first byte contains the following bit definition. The first five bits (bit[7:3]) notify the slaves that this is a 10-bit transfer followed by the next two bits (bit[2:1]), which set the slaves address bits 9:8, and the LSB bit (bit[0]) is the R/W bit. The second byte transferred sets bit[7:0] of the slave address. *Figure 21-7* shows the 10-bit address format, and *Table 21-1* defines the special purpose and reserved first byte addresses.

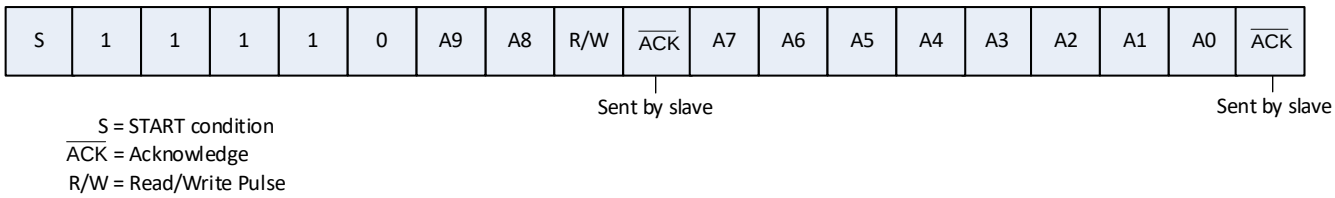


Figure 21-7 10-bit address format

All information provided in this document is subject to legal disclaimers.
© REALTEK 2024. All rights reserved.

Table 21-1 I2C definition of bits in the first byte

Slave address	R/W bit	Description
0000 000	0	General Call Address. I2C places the data in the receive buffer and issues a General Call interrupt.
0000 000	1	START byte
0000 001	x	CBUS address. I2C ignores these accesses.
0000 010	x	Reserved
0000 011	x	Reserved
0000 1XX	x	High speed master code
1111 1XX	x	Reserved
1111 0XX	x	10-bit slave addressing

The I2C does not restrict you from using these reserved addresses. However, if the reserved addresses are used, you may run into incompatibilities with other I2C components.

21.3.4.3 Transmitting and Receiving Protocol

The master can initiate data transmission and reception to/from the bus, acting as either a master-transmitter or master-receiver. A slave responds to requests from the master to either transmit data or receive data to/from the bus, acting as either a slave-transmitter or slave-receiver, respectively.

21.3.4.3.1 Master-Transmitter and Slave-Receiver

All data is transmitted in byte format, with no limit on the number of bytes transferred per data transfer. After the master sends the address and R/W bit or the master transmits a byte of data to the slave, the slave-receiver must respond with the acknowledge signal (ACK). When a slave-receiver does not respond with an ACK pulse, the master aborts the transfer by issuing a STOP condition. The slave must leave the SDA line high so that the master can abort the transfer.

If the master-transmitter is transmitting data as shown in *Figure 21-8*, then the slave-receiver responds to the master-transmitter with an acknowledge pulse after every bytes of data is received.

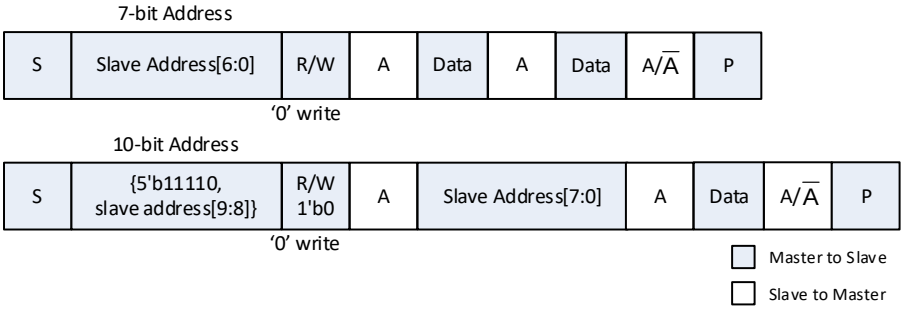


Figure 21-8 Master-Transmitter protocol

21.3.4.3.2 Master-Receiver and Slave-Transmitter

If the master is receiving data as shown in *Figure 21-9*, then the master responds to the slave-transmitter with an acknowledge pulse after a byte of data has been received, except for the last byte. This is the way the master-receiver notifies the slave-transmitter that this is the last byte. The slave-transmitter relinquishes the SDA line after detecting the No Acknowledge (NACK) so that the master can issue a STOP condition.

For 10-bit address, the transfer progress is shown in *Figure 21-9*.

- Address phase 1: master send {5'b11110, slave address [9:8], 1'b0}; R/W=1'b0 means master will send data in data phase.
- Data phase 1: master send slave address [7:0];
- A restart is set to indicate the direction toggle of data transfer.
- Address phase 2: master send {5'b11110, slave address [9:8], 1'b1}; R/W=1'b1 means master will receive data in data phase.
- Data phase 2: master receives data from slave.

When a master does not want to relinquish the bus with a STOP condition, the master can issue a RESTART condition. This is identical to a START condition except it occurs after the ACK pulse. Operating in master mode, the I2C can then communicate with the same slave using a transfer of a different direction.

All information provided in this document is subject to legal disclaimers.
© REALTEK 2024. All rights reserved.

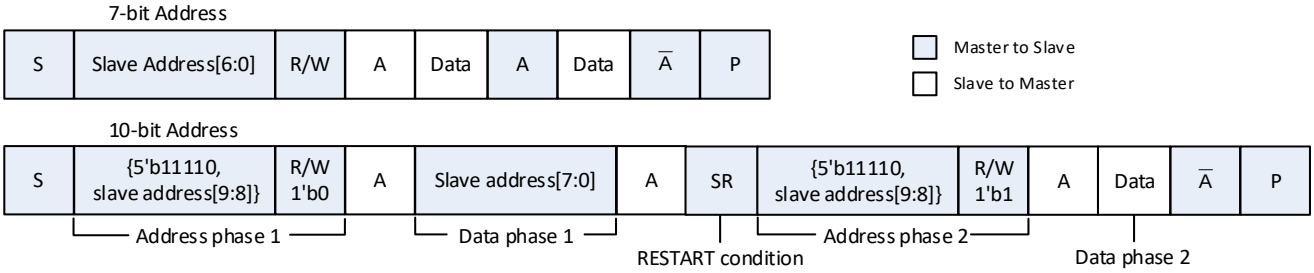


Figure 21-9 Master-Receiver protocol

21.3.4.4 START BYTE Transfer Protocol

Microcontroller can be connected to I2C-bus in two ways:

- Interrupt by START BYTE request: A microcontroller with an on-chip hardware I2C-bus interface can be programed to be only interrupted by requests from the bus. Or
- Polling the bus: When the device does not have such an interface, it must constantly monitor the bus. Obviously, the more times the microcontroller monitors, or polls the bus, the less time carrying out its intended function. There is therefore a speed difference between fast hardware device and a relatively slow microcontroller, which relies on software polling.

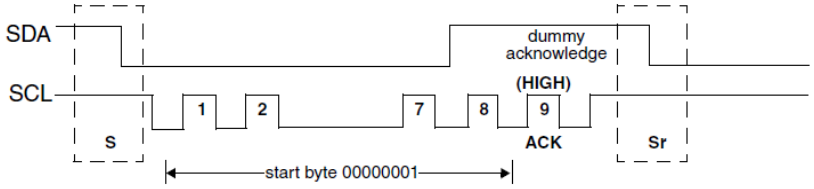


Figure 21-10 START BYTE transfer

Refer to *Figure 21-10*, the START BYTE procedure is as follows:

(1) Master generates a START condition.
(2) Master transmits the START byte (0000 0001).
(3) Master transmits the ACK clock pulse. (Present only to conform with the byte handling format used on the bus)
(4) No slave sets the ACK signal to 0.
(5) Master generates a RESTART (R) condition.

After the START condition S has been transmitted by a master which requires bus access, the START byte (0000 0001) is transmitted. Another microcontroller can therefore sample the SDA line at a low sampling rate until one of the seven zeros in the START byte is detected. After detection of this LOW level on the SDA line, the microcontroller can switch to a higher sampling rate to find the repeated START condition Sr which is then used for synchronization.

A hardware receiver will reset on receipt of the repeated START condition Sr and will therefore ignore the START byte.

An acknowledge-related clock pulse is generated after the START byte. This is present only to conform with the byte handling format used on the bus. No device is allowed to acknowledge the START byte.

A hardware receiver does not respond to the START BYTE because it is a reserved address and resets after the RESTART condition is generated.

21.3.4.5 General Call Transfer Protocol

The general call is for addressing every device connected to the I2C bus. However, if a device doesn't need any of the data supplied within the general call structure, it can ignore this address by not issuing an acknowledgement. If a device does require this address and behave as a slave-receiver. The second and following bytes will be acknowledged by every slave-receiver capable of handling this data. *Figure 21-11* gives the General Call address format. The second byte can be:

- 'h06: reset and write programmable part of slave address by hardware.
- 'h04: write programmable part of slave address by hardware.
- 7-bit mater address of the hardware master.

All information provided in this document is subject to legal disclaimers.
© REALTEK 2024. All rights reserved.

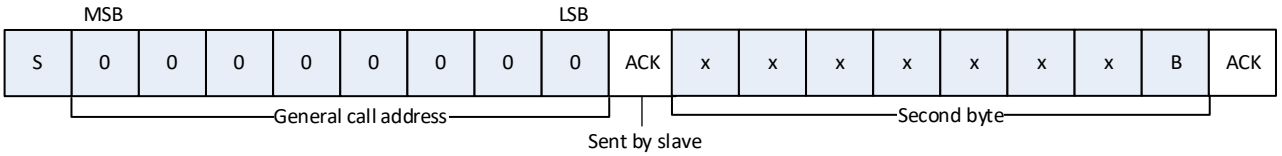


Figure 21-11 General call address format

When I2C acts as a slave, set IC_ACK_GENERAL_CALL (0x98) bit[0] to 1 to respond with a ACK when it receives a General Call, and a General Call interrupt will be issued. When set this bit to 0, the I2C does not generate General Call interrupts.

21.3.4.6 NULL DATA Transfer Protocol

NULL DATA transfer is used for some sensors. When IC_DATA_CMD[11] and IC_DATA_CMD[9] is set to 1, I2C would ignore REG_IC_TAR but take the TXFIFO data as slave address. It would only send TXFIFO data in the address phase without any further transmission. *Figure 21-12* gives the NULL DATA transfer format.

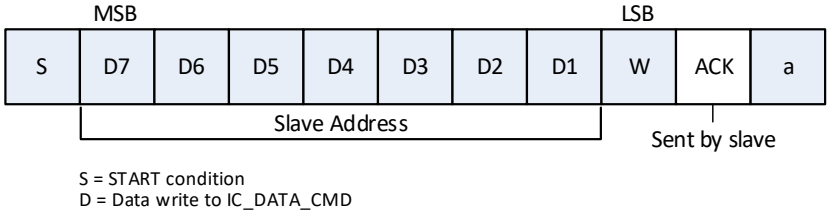


Figure 21-12 NULL DATA transfer format

21.3.5 Transmit FIFO Management and START, STOP and RESTART Generation

The I2C does not generate a STOP if the Tx FIFO becomes empty; in this situation, the component holds the SCL line low, stalling the bus until a new entry is available in the Tx FIFO. A STOP condition is generated only when the user specifically requests it by setting bit[9] (STOP bit) of the command written to IC_DATA_CMD register. *Figure 21-13* shows the fields in IC_DATA_CMD. Refer to the register description for more detailed information.

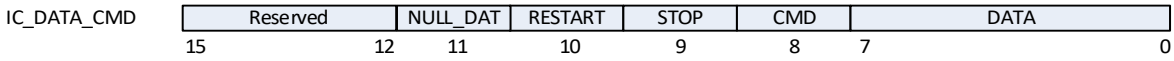


Figure 21-13 IC_DATA_CMD register content

Figure 21-14 and *Figure 21-15* illustrate the behavior of the I2C when the Tx FIFO becomes empty while operating as a master transmitter and a master receiver respectively, as well as showing the generation of a STOP condition.

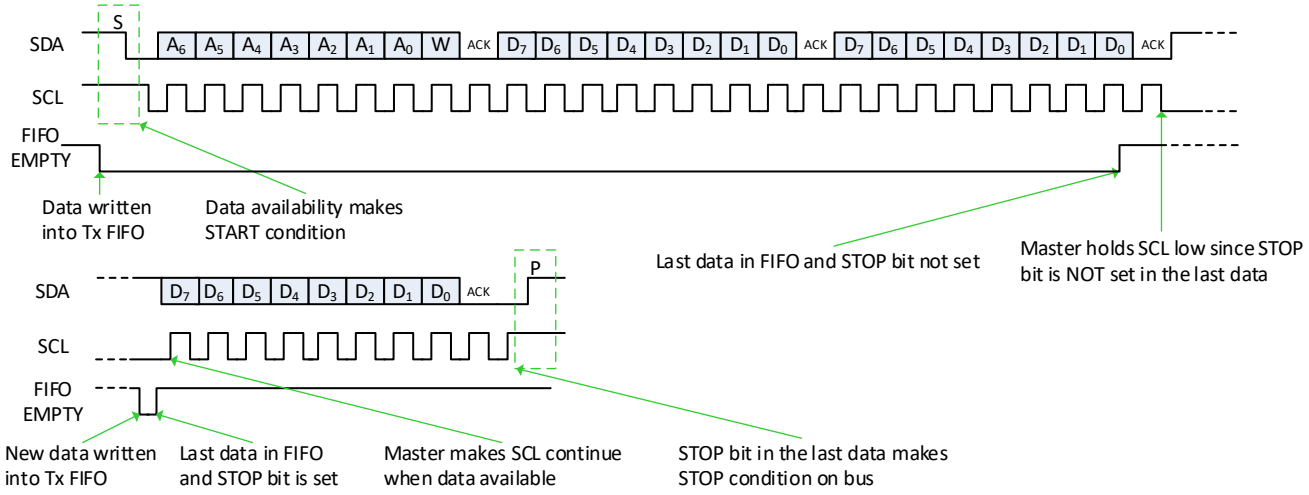


Figure 21-14 Master transmitter — Tx FIFO empties/STOP generation

All information provided in this document is subject to legal disclaimers.
© REALTEK 2024. All rights reserved.

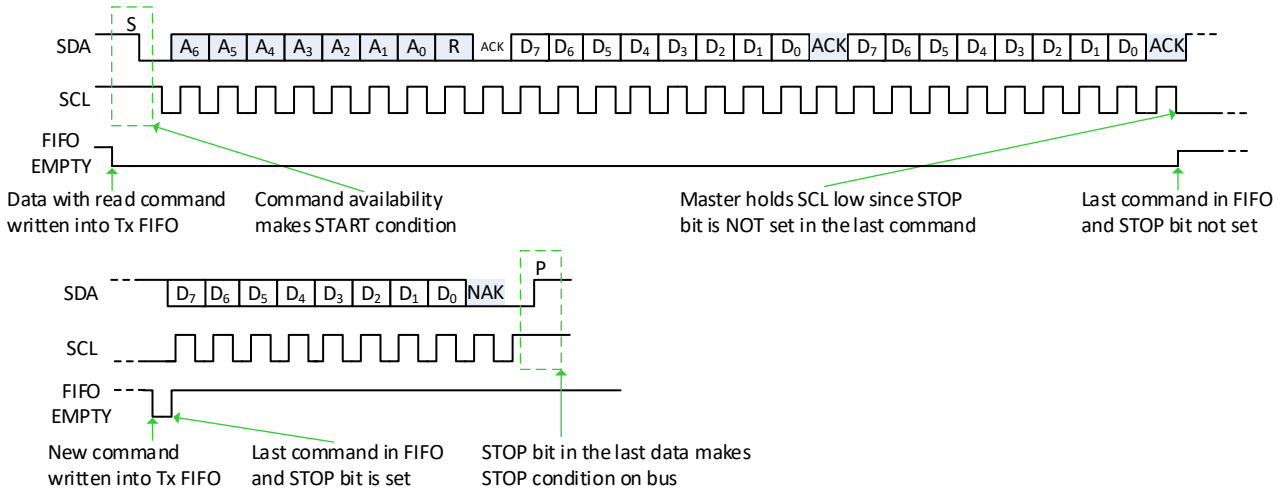


Figure 21-15 Master receiver — Tx FIFO empties/STOP generation

Figure 21-16 and *Figure 21-17* illustrate configurations where the user can control the generation of RESTART conditions on the I2C bus. If bit[10] (RESTART) of the IC_DATA_CMD register is set and the restart capability is enabled (IC_RESTART_EN=1), a RESTART is generated before the data byte is written to or read from the slave. If the restart capability is not enabled a STOP followed by a START is generated in place of the RESTART. *Figure 21-16* illustrates this situation during operation as a master transmitter. *Figure 21-17* illustrates the same situation, but during operation as a master receiver.

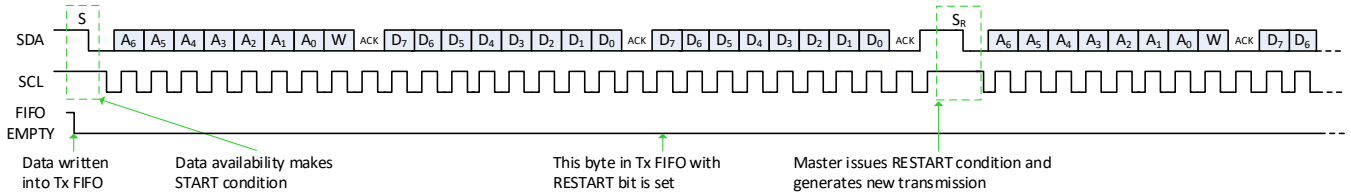


Figure 21-16 Master transmitter — RESTART bit of IC_DATA_CMD set

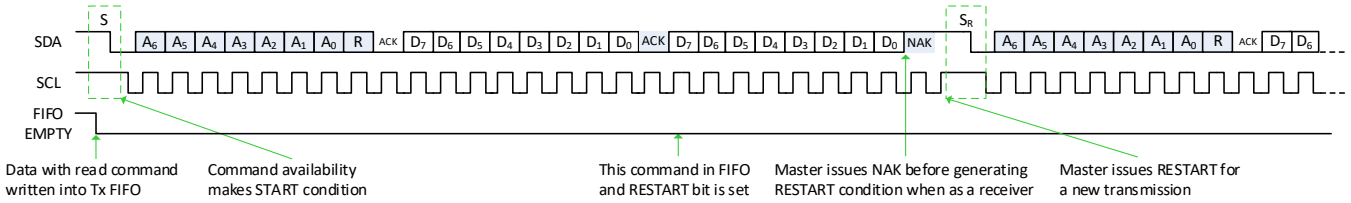


Figure 21-17 Master receiver — RESTART bit of IC_DATA_CMD set

Figure 21-18 and *Figure 21-19* illustrate operation as a master transmitter and a master receiver respectively where the STOP bit of the IC_DATA_CMD register is set and the Tx FIFO is not empty.

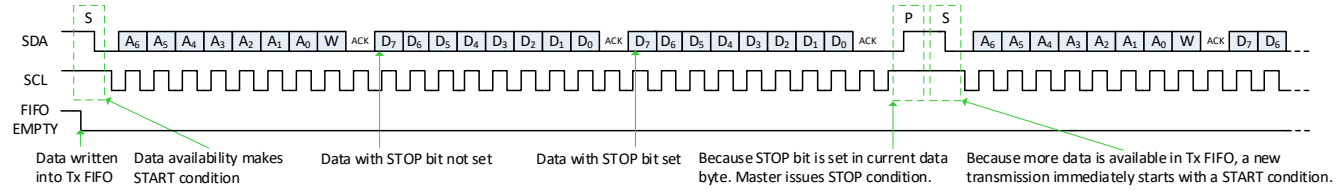


Figure 21-18 Master transmitter — STOP bit of IC_DATA_CMD set/Tx FIFO not empty

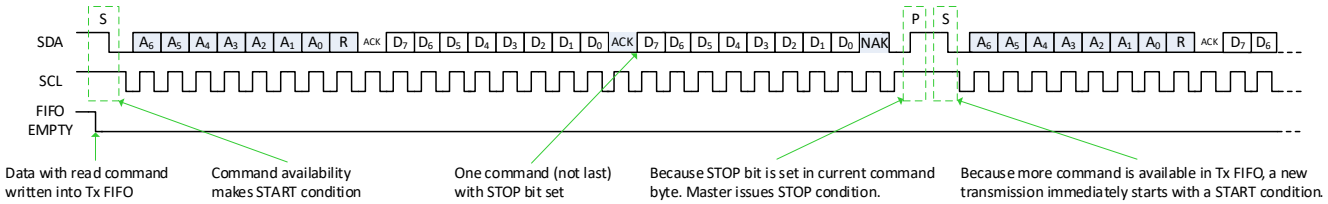


Figure 21-19 Master receiver — STOP bit of IC_DATA_CMD set/Tx FIFO not empty

All information provided in this document is subject to legal disclaimers.
© REALTEK 2024. All rights reserved.

21.3.6 Clock Synchronization

When two or more masters try to transfer information on the bus at the same time, they must arbitrate and synchronize the SCL clock. All masters generate their own clock to transfer messages. Data is valid only during the high period of SCL clock. Clock synchronization is performed using the wired-AND connection to the SCL signal. When the master transitions the SCL clock to 0, the master starts counting the low time of the SCL clock and transitions the SCL clock signal to 1 at the beginning of the next clock period. However, if another master is holding the SCL line to 0, then the master goes into a HIGH wait state until the SCL clock line transitions to 1.

All masters then count off their high time and the master with the shortest high time transitions the SCL line to 0. The masters then count out their low time and the one with the longest low time forces the other master into a HIGH wait state. Therefore, a synchronized SCL clock is generated, which is illustrated in *Figure 21-20*. Optionally, slaves may hold the SCL line low to slow down the timing on the I2C bus.

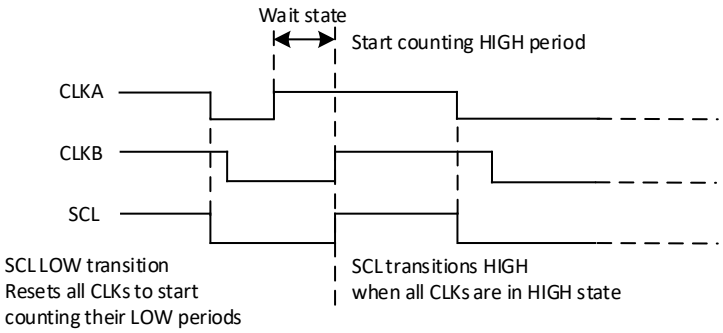


Figure 21-20 Multi-Master clock synchronization

21.3.7 Multiple Master Arbitration

The I2C bus protocol allows multiple masters to reside on the same bus. I2C master may start a transfer only if the bus is free. Two or more I2C master may start a transfer in a very short time interval by generating a START condition. CLKA is I2C bus clock generated by master 1, and CLKB is I2C bus clock generated by master 2. SCL in *Figure 21-21* and *Figure 21-20* is wire-AND of CLKA and CLKB.

If two master send a START on the bus within the minimum hold time ($t_{HD, STA}$), arbitration takes place on the SDA line, while the SCL line is 1. The master, which transmits a 1 while the other master transmits 0, loses arbitration and turns off its data and clock output stage.

Figure 21-21 illustrates the timing of when two masters are arbitrating on the bus.

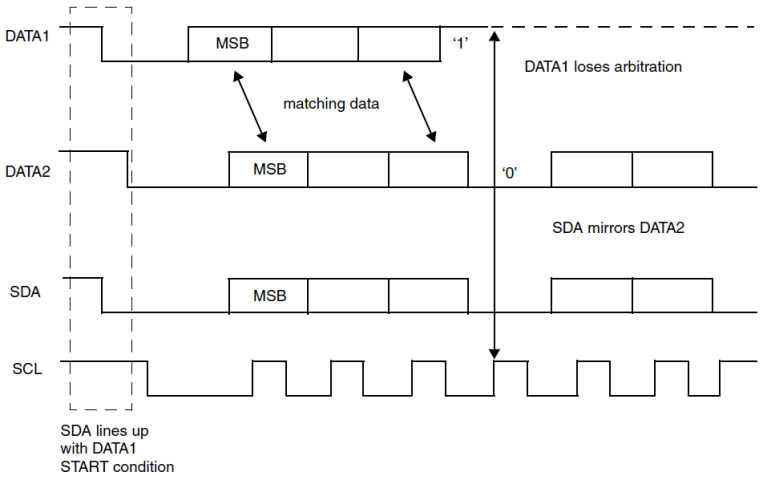


Figure 21-21 Multiple master arbitration

For SS-mode or FS-mode master, arbitration may continue through address, address ACK, data, and data ACK bits. Special attention must be paid, during a serial transfer, if the arbitration procedure is still in progress at the moment when a repeated START condition or a STOP condition is transmitted to the I2C bus. If it's possible for such a situation to occur, the masters involved must send this repeated START condition or STOP condition at the same position in the format frame. In other words, arbitration isn't allowed between:

- A repeated START condition and a data bit
- A STOP condition and a data bit
- A repeated START condition and a STOP condition.

All information provided in this document is subject to legal disclaimers.
© REALTEK 2024. All rights reserved.

As an HS-mode master has a unique 8-bit master code, it will always finish the arbitration during the first byte (sending of HS master code). This 8-bit code is defined by the system designer and is set by writing to the IC_HS_MADDR (High Speed Master Mode Code Address) register. Because the codes are unique, only one master can win arbitration, which occurs by the end of the transmission of the high-speed master code. Control of the bus is determined by address or master code and data sent by competing masters, so there is no central master nor any order of priority on the bus.

If a master also incorporates a slave function and it loses arbitration during the addressing stage, the winning master may be trying to address it. The losing master must therefore switch over immediately to its slave mode.

Slaves are not involved in the arbitration process.

21.3.8 SDA/SCL Filter

The I2C provides filtering on the SDA (ic_data_in_a) and SCL (ic_scl_in_a) inputs, suppressing noise and signal spikes. Set IC_DIG_FLTR_SEL (0xEC[8]) to open the filter. Set IC_DIG_FLTR_DEG (0xEC[3:0]) to adjust the duration of noise or spikes that can be suppressed. The larger IC_DIG_FLTR_DEG is, the longer duration can be suppressed. However, the delay of signals after filter is also longer, refer to *Table 21-2*.

- M – Delay between ic_data_in_a and dig_sda or ic_scl_in_a and dig_scl is M ic_clk cycles, including 2 meta-stability removal cycles.
- N – Noise or signal spikes with duration less than N ic_clk cycles could be suppressed.

Table 21-2 SDA/SCL digital filter

IC_DIG_FLTR_DEG[3:1]	N (max. filtered cycles)	M (delay cycles)
0	1	3
1	2	4
2	3	5
3	4	6
4	5	7
5	6	8
6	7	9
7	8	10

21.3.9 IC_CLK Frequency Configuration

When the I2C is configured as a master, the *CNT registers must be set before any I2C bus transaction can take place in order to ensure proper I/O timing. The *CNT registers are:

- IC_SS_SCL_HCNT
- IC_SS_SCL_LCNT
- IC_FS_SCL_HCNT
- IC_FS_SCL_LCNT
- IC_HS_SCL_HCNT
- IC_HS_SCL_LCNT

NOTE

*It is not necessary to program any of the *CNT registers if the I2C is enabled to operate only as an I2C slave, since these registers are used only to determine the SCL timing requirements for operation as an I2C master.*

21.3.9.1 Minimum High and Low Counts

When the I2C operates as an I2C master, in both transmit and receive transfers:

- Min. value that can be programmed in the *_LCNT registers is 8.
- Min. value allowed for the *_HCNT registers is 6.

The min. value of 8 for the *_LCNT registers is due to the time required for the I2C to drive SDA after a negative edge of SCL.
The min. value of 6 for the *_HCNT register is due to the time required for the I2C to sample SDA during the high period of SCL.

21.3.9.2 Programmable Duty Cycle

This section describes the minimum ic_clk frequencies that the I2C supports for each speed mode, and the associated high and low count values. Note that these limits apply to the I2C in both master and slave modes. The limits for slave mode are required so that the I2C does not break the $t_{HD, DAT}$ max. I2C protocol timing requirement.

All information provided in this document is subject to legal disclaimers.
© REALTEK 2024. All rights reserved.

21.3.9.2.1 High and Low Counts Calculation

The calculations below show how to calculate SCL high and low counts for each speed mode in the I2C.

The following conditions should be given:

- Ic_clk frequency: f_{ic_clk} → T_{ic_clk}=1/f_{ic_clk}
- HIGH_TIME: HIGH period of SCL
- LOW_TIME: LOW period of SCL
- SCL frequency: f_{SCL} → SCL_{PERIOD}=1/f_{SCL}

While we want to get HIGH_COUNT & LOW_COUNT, we can derive the equations:

$$\begin{cases} \dfrac{SCL_{PERIOD}}{HIGH_{COUNT} + LOW_{COUNT}} = T_{ic_{clk}} \\ \dfrac{HIGH_{COUNT}}{HIGH_{TIME}} = \dfrac{LOW_{COUNT}}{LOW_{TIME}} \\ HIGH_{TIME} + LOW_{TIME} = SCL_{PERIOD} \end{cases}$$

Then, we can get

$$\begin{cases} HIGH_{COUNT} = \dfrac{HIGH_{TIME}}{T_{ic_clk}} \\ LOW_{COUNT} = \dfrac{LOW_{TIME}}{T_{ic_clk}} \end{cases}$$

Full Speed Mode and Standard Speed Mode Example

For I2C0, f_{ic_clk}=10MHz; for I2C1/I2C2, f_{ic_clk}=100MHz

- ic_clk frequency: f_{ic_clk}=10MHz → T_{ic_clk}=100ns
- According to table 5 of I2C-Bus specification version2.1, FS Mode and SS Mode timing parameters for SCL are specified as:
  - Standard mode: min. HIGH period of SCL is 4.0us (min. IC_SS_SCL_HCNT = 4.0us/100ns = 40)
  - Standard mode: min. LOW period of SCL is 4.7us (min. IC_SS_SCL_LCNT = 4.7us/100ns = 47)
  - Fast mode: min. HIGH period of SCL is 0.6us (min. IC_FS_SCL_HCNT = 0.6us/100ns = 6)
  - Fast mode: min. LOW period of SCL is 1.3us (min. IC_FS_SCL_LCNT = 1.3us/100ns = 13)
- SCL frequency:
  - 100kHz → SCL_{PERIOD}=10000ns
  - 400kHz → SCL_{PERIOD}=2500ns
- SS Mode (100kHz) :
  - IC_SS_SCL_HCNT = (10000ns/100ns)*(40/(40+47)) = 45.97 = 46 (round up)
  - IC_SS_SCL_LCNT = (10000ns/100ns)*(47/(40+47)) = 54.02 = 54 (round down)
- FS Mode (400kHz):
  - IC_FS_SCL_HCNT = (2500ns/100ns)*(6/(6+13)) = 7.89 = 8 (round up)
  - IC_FS_SCL_LCNT = (2500ns/100ns)*(13/(6+13)) = 17.11 = 17 (round down)

High Speed Mode Example

I2C0 does not support HS mode; for I2C1/2, f_{ic_clk}=100MHz

- Ic_clk frequency: f_{ic_clk}=100MHz → T_{ic_clk}=10ns
- According to table 7 of I2C-Bus specification version2.1, HS mode timing parameters for SCL are specified as:
  - 3.4M, bus loading=100pf: min. HIGH period of SCL is 60ns (min. IC_HS_SCL_HCNT = 60ns/10ns= 6)
  - 3.4M, bus loading=100pf: min. LOW period of SCL is 120ns (min. IC_HS_SCL_LCNT = 120ns/10ns= 12)
  - 1.7M, bus loading=400pf: min. HIGH period of SCL is 160ns (min. IC_HS_SCL_HCNT = 160ns/10ns= 16)
  - 1.7M, bus loading=400pf: min. LOW period of SCL is 320ns (min. IC_HS_SCL_LCNT = 320ns/10ns= 32)
- SCL frequency:
  - 3.4M → SCL_{PERIOD}=294ns
  - 1.7M → SCL_{PERIOD}=588ns
- HS Mode (3.4M, bus loading=100pf):
  - IC_HS_SCL_HCNT = (294ns/10ns)*(6/(6+12)) = 9.8 = 10 (round up)
  - IC_HS_SCL_LCNT = (294ns/10ns)*(12/(6+12)) = 19.6 = 19 (round down)
- HS Mode (1.7M, bus loading=400pf):
  - IC_HS_SCL_HCNT = (588ns/10ns)*(16/(16+32)) = 19.6 = 20 (round up)
  - IC_HS_SCL_LCNT = (588ns/10ns)*(32/(16+32)) = 39.2 = 39 (round down)

NOTE

Since the slave may stretch SCL line, the actual speed is a little bit smaller then 100K/400K/3.4M if we set the above calculated values. In order to reach the accurate speed, calculated values should be reduced a little according to test results.

All information provided in this document is subject to legal disclaimers. © REALTEK 2024. All rights reserved.

21.3.9.2.2 Minimum Core Clock Frequency for HS Mode

The core clock of I2C is ic_clk. SCL stretch occur when the master is waiting for data but the slave's data buffer is empty. The slave drives the SCL low to hold the bus until software put data in slave's data buffer. The master waiting for data by stretching the low period of first data bit.

An example of SCL stretch in master receive mode is shown in *Figure 21-22*. There are two I2C reside on the I2C bus. Set one of our I2C as master, and another as slave. filter_delay =M indicates slave digital filter delay, refer to section *21.3.8*, M≤10.

- Slave SCL low delay = (2+ filter_delay + 4 + 1) cycles = (7+ filter_delay) cycles = (7+M) cycles≤17 cycles (1)
- Master stretch SCL delay = (Slave SCL low delay + 3+1) cycles = (11+M) cycles≤21 cycles (2)

After slave detect ACK, slv_fctrl set 1 if data buffer is empty. Then slave generated SCL (slv_SCL) is pull low. To make sure SCL stays low before slave drives SCL low, positive edge of mst_SCL (point 1) must be later than negative edge of slv_SCL (point 2).

- Slave SCL low delay ≤ SCL_HCNT+SCL_LCNT (3)

To make sure stretch occurs in the first bit of data, point 3 must arrived before the end of data bit[0].

- Master stretch SCL delay < 2*SCL_HCNT+SCL_LCNT (4)

Refer to *21.3.9.2.1* , HS-Mode (3.4M, bus loading=100pf):

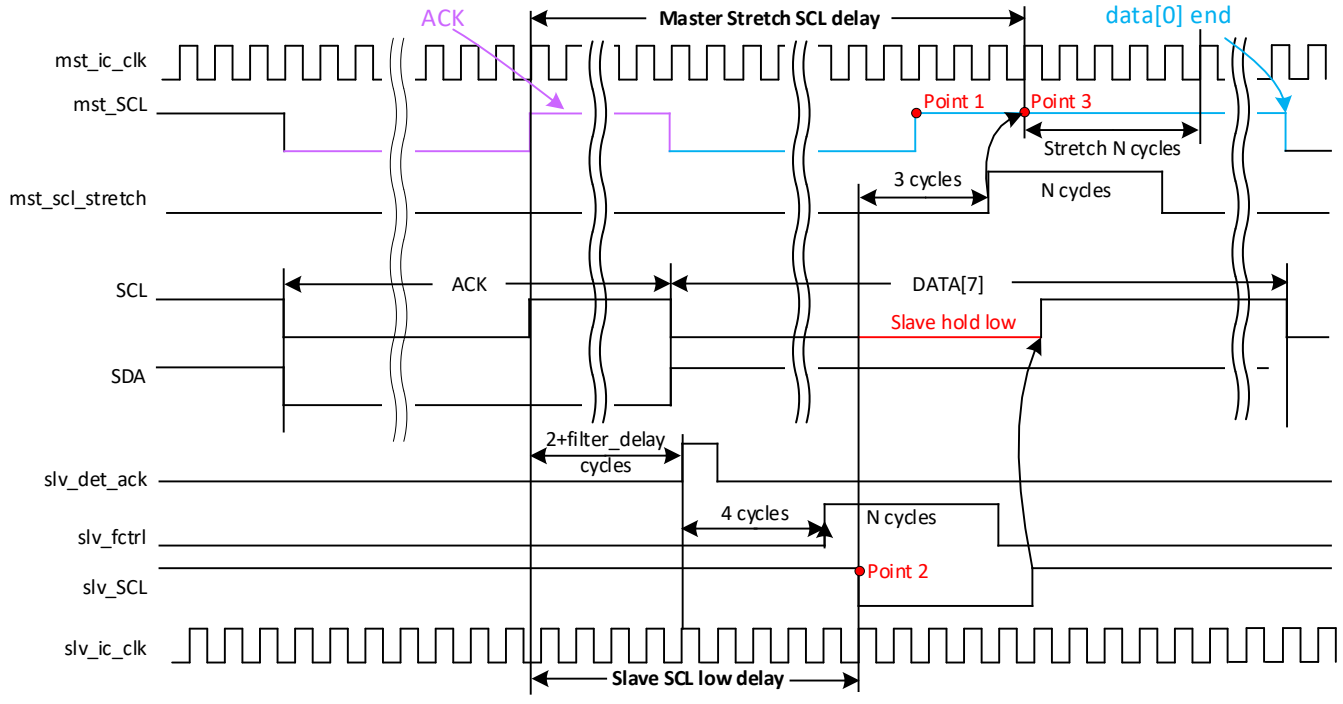
- Min. IC_HS_SCL_HCNT = 6 (5)
- Min. IC_HS_SCL_LCNT = 12 (6)

Min. value in (5) (6) is satisfied equivalent in (1) (3).
Min. value in (5) (6) is satisfied equivalent in (2) (4).

So that:

- T_{ic_clk}= SCL_{PERIOD}/(IC_HS_SCL_HCNT+ IC_HS_SCL_LCNT) <= 294ns/18 = 16.34
- f_{ic_clk}= 1/T_{ic_clk} > 61.2MHz

Because the value of slave_SCL_low_delay depends on the slaves connected to I2C bus, which is unknown, ic_clk frequency larger than 61.2MHz is suggested to compatible with different I2C device. f_{ic_clk}<61.2MHz may cause transfer error in High Speed Mode.



Point 1: high period start of master generated SCL;
Point 2: slave starts stretch SCL low period;
Point 3: master gets the slave's SCL stretch, and start stretch data[0] cycle.
If master's low period is long enough, point 3 may be in the left side of point 1.

Figure 21-22 Master receive SCL stretch

All information provided in this document is subject to legal disclaimers.
© REALTEK 2024. All rights reserved.

21.3.10 Programmable SDA Hold Time

As each application will encounter differing board delays, the I2C contains a software programmable register IC_SDA_HOLD (0x7C) to enable dynamic adjustment of the SDA hold time.

- When I2C acts as a master transmitter:
  - Master ADDRESS/MSCODE phase: When the master sends addresses or master_code, the SCL from I2C bus may be inconsistent with the mst_scl from master because of the arbitration mechanism. To ensure the SDA toggles after the negative edge of SCL, shift out register is triggered by neg_scl, which is synchronization of SCL from I2C bus. The SCL to neg_scl path includes a digital filter and a two-stage synchronizer. Refer to *Table 21-2*, the digital filter delay of I2C bus interface is (IC_DIG_FLTR_DEG[3:1]+3) cycles.
    The hold time is maximum of (5+ IC_DIG_FLTR_DEG[3:1]) and IC_SDA_HOLD.
  - Master WDATA phase: mst_scl is consistent with the SCL of I2C bus, use the negative edge of mst_scl to trigger shift out register. The hold time is IC_SDA_HOLD ic_clk periods.
- When I2C acts as a slave transmitter:
  - Slave WDATA phase: Shift out register is triggered by neg_scl, which is synchronization of SCL from I2C bus. The SCL to neg_scl path includes a digital filter and a two-stage synchronizer. Refer to *Table 21-2*, the digital filter delay of I2C bus interface is (IC_DIG_FLTR_DEG[3:1]+3) cycles.
    The hold time is maximum of (5+ IC_DIG_FLTR_DEG[3:1]) and IC_SDA_HOLD.

NOTE

The 8th bit of data is not under control of IC_SDA_HOLD, because after the 8th SCL clock negative edge, receiver controls the SDA line to respond ACK or NACK.

The IC_SDA_HOLD register can be programmed only when I2C is disabled (IC_ENABLE=0).

21.3.11 Slave Mode Dual Own Address

The I2C has dual own address:

- Slave 1 supports 7- or 10-bit address modes.
- Slave 2 only supports 7-bit address mode.
- Set IC_CON[7:6] (0x10) to enable slave 1 or slave 2.
- Slave 1 address is set in IC_SAR (0x08), which supports 7 bits or 10 bits.
- Slave 2 address is set in IC_SAR2 (0xF4), which only supports 7 bits.
- IC_CON[3] (0x10) IC_10BITADDR_SLAVE controls whether the I2C responds to 7- or 10-bit address.
- If slave 1 and slave 2 are both enabled, the one who matches address would response ACK.

21.4 Operation Flow

This section provides information on operation modes.

NOTE

The I2C should only be set to operate as an I2C master, or I2C slave, but not both simultaneously. This is achieved by ensuring that bit[6] (IC_SLAVE_DISABLE) and bit[0] (MASTER_MODE) of the IC_CON register are never set to 0 and 1 respectively.

21.4.1 Slave Mode Operation

This section discusses slave mode procedures.

21.4.1.1 Initial Configuration

To use the I2C as a slave, perform the following steps:

(1) Disable the I2C by writing a '0' to bit[0] of the IC_ENABLE register.
(2) Write the IC_SAR register (bit[9:0]) to set the slave address. This is the address to which the I2C responds.
(3) Write the IC_CON register to specify which type of addressing is supported (7- or 10-bit by setting bit[3]). Enable the I2C in slave-only mode by writing a '0' into bit[6] (IC_SLAVE_DISABLE) and a '0' to bit[0] (MASTER_MODE).
(4) Enable the I2C by writing a '1' in bit[0] of the IC_ENABLE register.

NOTE

The slaves and masters do not have to be programmed with the same type of addressing 7- or 10-bit address. For instance, a slave can be programmed with 7-bit addressing and a master with 10-bit addressing, and vice versa.

All information provided in this document is subject to legal disclaimers.
© REALTEK 2024. All rights reserved.

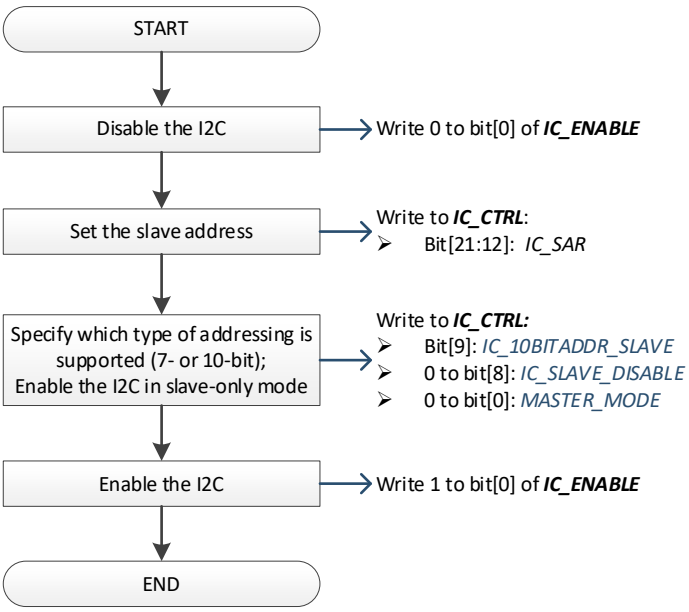


Figure 21-23 I2C Slave mode initial configuration

21.4.1.2 Slave-Transmitter Operation for a Single Byte

When another I2C master device on the bus addresses the I2C and requests data, the I2C acts as a slave-transmitter and the following steps occur:

(1) The other I2C master device initiates an I2C transfer with an address that matches the slave address in the IC_SAR register of the I2C.
(2) The I2C acknowledges the sent address and recognizes the direction of the transfer to indicate that it is acting as a slave-transmitter.
(3) The I2C asserts the RD_REQ interrupt (bit[5] of the IC_RAW_INTR_STAT register) and holds the SCL line low. It is in a wait state until software responds.
   a) Reads that indicate IC_RAW_INTR_STAT[5] (R_RD_REQ bit) being set to 1 must be treated as the equivalent of the RD_REQ interrupt being asserted.
   b) Software must then act to satisfy the I2C transfer.
(4) If there is any data remaining in the Tx FIFO before receiving the read request, then the I2C asserts a TX_ABRT interrupt (bit[6] of the IC_RAW_INTR_STAT register) to flush the old data from the Tx FIFO.

NOTE

Because the I2C's Tx FIFO is forced into a flushed/reset state whenever a TX_ABRT Interrupt clear event occurs, it is necessary for software to release the I2C from this state by reading the IC_CLR_TX_ABRT register before attempting to write into the Tx FIFO. See register IC_RAW_INTR_STAT for more details.

   a) Reads that indicate bit[6] (R_TX_ABRT) being set to 1 must be treated as the equivalent of the TX_ABRT interrupt being asserted.
   b) There is no further action required from software.

(5) Software writes to the IC_DATA_CMD register with the data to be written (by writing a '0' in bit[8]).
(6) Software must clear the RD_REQ and TX_ABRT interrupts (bit[5] and bit[6] respectively) of the IC_RAW_INTR_STAT register before proceeding.
(7) The I2C releases the SCL and transmits the byte.
(8) The master may hold the I2C bus by issuing a RESTART condition or release the bus by issuing a STOP condition.

All information provided in this document is subject to legal disclaimers.
© REALTEK 2024. All rights reserved.

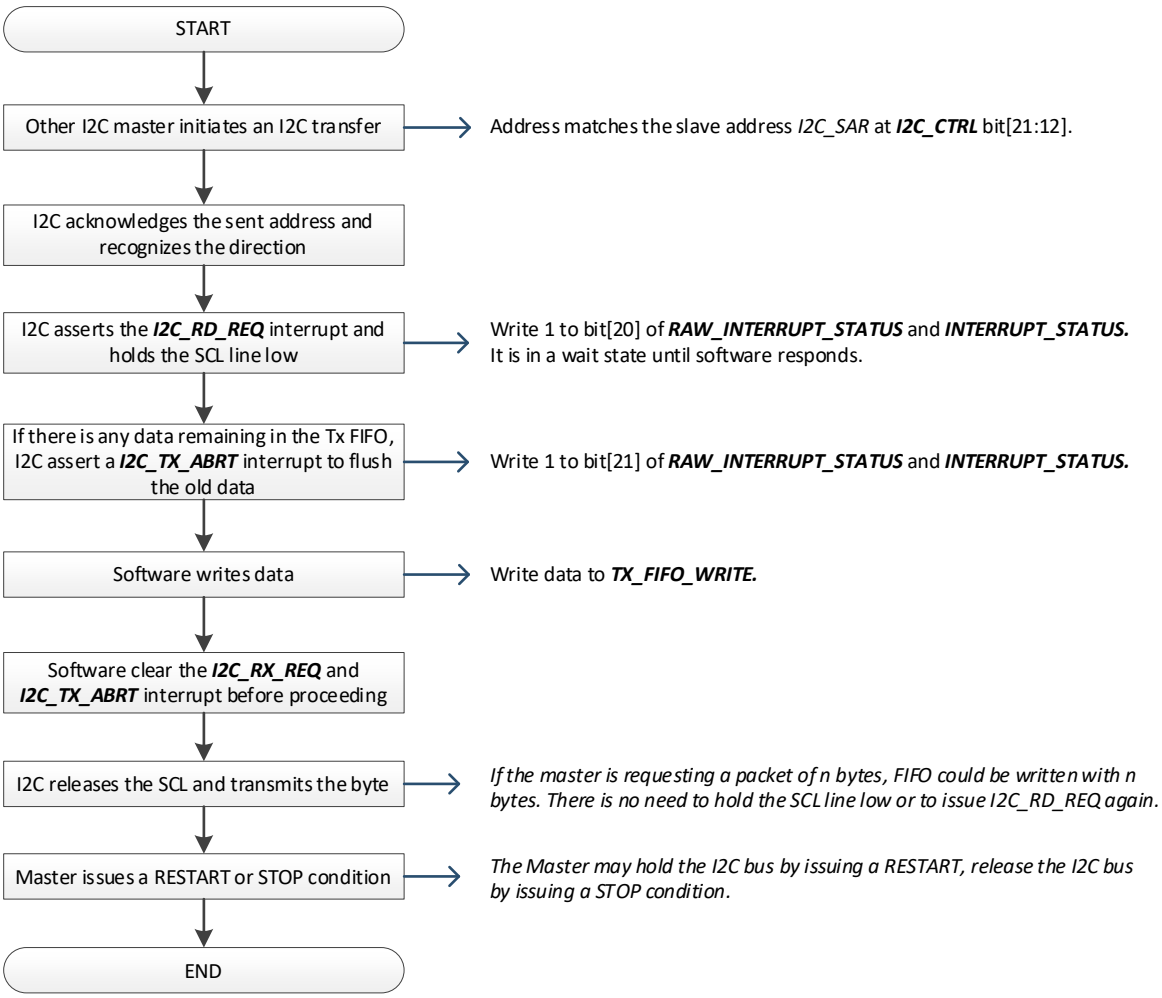


Figure 21-24 I2C Slave mode data transmit

21.4.1.3 Slave-Receiver Operation for a Single Byte

When another I2C master device on the bus addresses the I2C and is sending data, the I2C acts as a slave-receiver and the following steps occur:

(1) The other I2C master device initiates an I2C transfer with an address that matches the I2C's slave address in the IC_SAR register.
(2) The I2C acknowledges the sent address and recognizes the direction of the transfer to indicate that the I2C is acting as a slave-receiver.
(3) I2C receives the transmitted byte and places it in the receive buffer.

NOTE

If the Rx FIFO is completely filled with data when a byte is pushed, then an overflow occurs and the I2C continues with subsequent I2C transfers. Because a NACK is not generated, software must recognize the overflow when indicated by the I2C (by the R_RX_OVER bit in the IC_INTR_STAT register) and take appropriate actions to recover from lost data. Hence, there is a real time constraint on software to service the Rx FIFO before the latter overflow as there is no way to reapply pressure to the remote transmitting master. You must select a deep enough Rx FIFO depth to satisfy the interrupt service interval of their system.

(4) I2C asserts the RX_FULL interrupt (IC_RAW_INTR_STAT[2] register).
If the RX_FULL interrupt has been masked, due to setting IC_INTR_MASK[2] register to 0 or setting IC_TX_TL to a value larger than 0, then it is recommended that a timing routine be implemented for periodic reads of the IC_STATUS register. Reads of the IC_STATUS register, with bit[3] (RFNE) set at 1, must then be treated by software as the equivalent of the RX_FULL interrupt being asserted.
(5) Software may read the byte from the IC_DATA_CMD register (bit[7:0]).
(6) The other master device may hold the I2C bus by issuing a RESTART condition or release the bus by issuing a STOP condition.

All information provided in this document is subject to legal disclaimers.
© REALTEK 2024. All rights reserved.

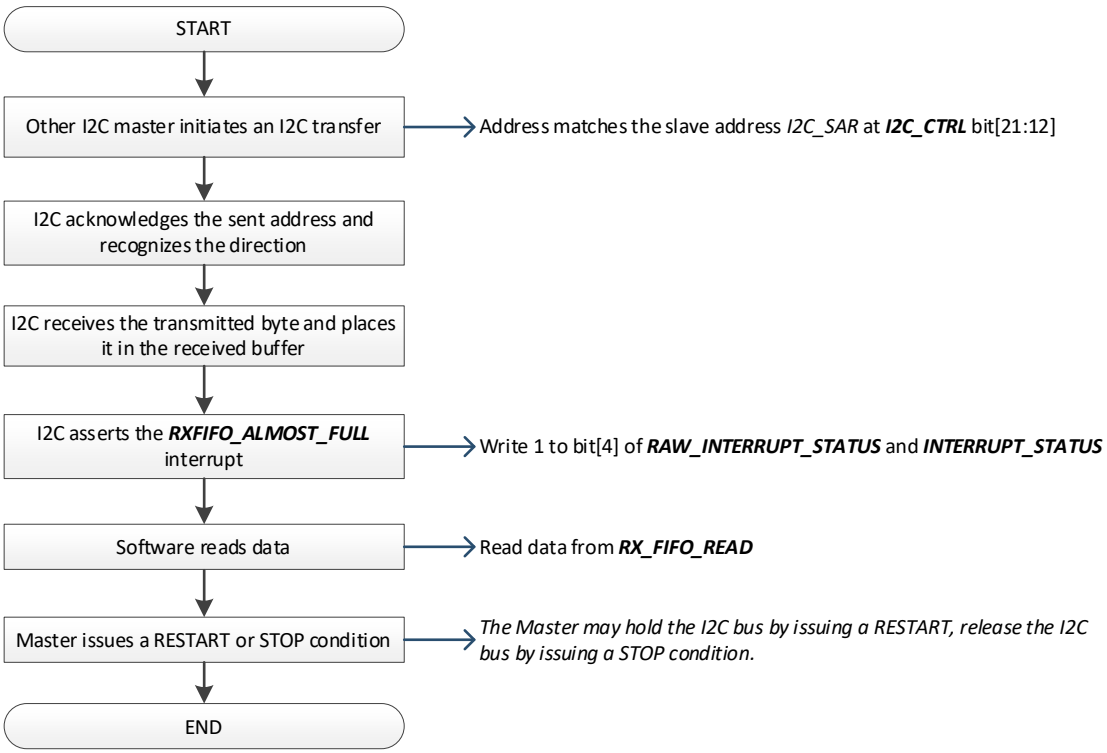


Figure 21-25 I2C Slave mode receive data

21.4.1.4 Slave-Transfer Operation for Bulk Transfers

In the standard I2C protocol, all transactions are single byte transactions and the programmer responds to a remote master read request by writing one byte into the slave's Tx FIFO. When a slave (slave-transmitter) is issued with a read request (RD_REQ) from the remote master (master-receiver), at a minimum there should be at least one entry placed into the slave-transmitter's Tx FIFO. I2C is designed to handle more data in the Tx FIFO so that subsequent read requests can take that data without raising an interrupt to get more data. Ultimately, this eliminates the possibility of significant latencies being incurred between raising the interrupt for data each time had there been a restriction of having only one entry placed in the Tx FIFO.

This mode only occurs when I2C is acting as a slave-transmitter. If the remote master acknowledges the data sent by the slave-transmitter and there is no data in the slave's Tx FIFO, the I2C holds the I2C SCL line low while it raises the read request interrupt (RD_REQ) and waits for data to be written into the Tx FIFO before it can be sent to the remote master.

If the RD_REQ interrupt is masked, due to bit[5] (M_RD_REQ) of the IC_INTR_STAT register being set to 0, then it is recommended that a timing routine be used to activate periodic reads of the IC_RAW_INTR_STAT register. Reads of IC_RAW_INTR_STAT that return bit[5] (R_RD_REQ) set to 1 must be treated as the equivalent of the RD_REQ interrupt referred to in this section.

The RD_REQ interrupt is raised upon a read request, and like interrupts, must be cleared when exiting the interrupt service handling routine (ISR). The ISR allows you to either write 1 byte or more than 1 byte into the Tx FIFO. During the transmission of these bytes to the master, if the master acknowledges the last byte, then the slave must raise the RD_REQ again because the master is requesting for more data.

If the programmer knows in advance that the remote master is requesting a packet of n bytes, then when another master addresses I2C and requests data, the Tx FIFO could be written with n number bytes and the remote master receives it as a continuous stream of data. For example, the I2C slave continues to send data to the remote master as long as the remote master is acknowledging the data sent and there is data available in the Tx FIFO. There is no need to hold the SCL line low or to issue RD_REQ again.

If the remote master is to receive n bytes from the I2C but the programmer wrote a number of bytes larger than n to the Tx FIFO, then when the slave finishes sending the requested n bytes, it clears the Tx FIFO and ignores any excess bytes.

21.4.2 Master Mode Operation

This section discusses master mode procedures.

All information provided in this document is subject to legal disclaimers.
© REALTEK 2024. All rights reserved.

21.4.2.1 Initial Configuration

The target address and address format can be changed dynamically without having to disable I2C. This parameter only applies to when I2C is acting as a master because the slave requires the component to be disabled before any changes can be made to the address.

The procedures are very similar and are only different with regard to where the IC_10BITADDR_MASTER bit is set (bit[12] of IC_TAR register).

(1) Disable the I2C by writing 0 to the IC_ENABLE register.
(2) Write to the IC_CON register to set the maximum speed mode supported for slave operation (bit[2:1]). Writing 1 to bit[0] and bit[6] to enable Master Module and disable Slave Module.
(3) Write to the IC_TAR register the address of the I2C device to be addressed. It also indicates whether a General Call or a START BYTE command is going to be performed by I2C. The desired speed of the I2C master-initiated transfers, either 7-bit or 10-bit addressing, is controlled by the IC_10BITADDR_MASTER bit (bit[12]).
(4) Enable the I2C by writing a '1' in the IC_ENABLE register.
(5) Write transfer direction and data to be sent to the IC_DATA_CMD register. If the IC_DATA_CMD register is written before the I2C is enabled, the data and commands are lost as the buffers are kept cleared when I2C is disabled.

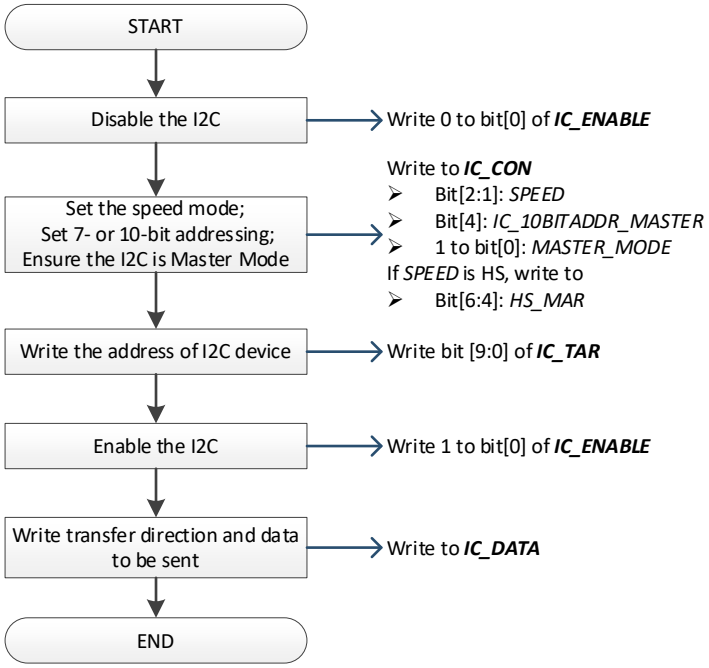


Figure 21-26 I2C Mater mode initial configuration

21.4.2.2 Dynamic IC_TAR or IC_10BITADDR_MASTER Update

The I2C supports dynamic updating of the IC_TAR (bit[9:0]) and IC_10BITADDR_MASTER (bit[12]) bits of the IC_TAR register. You can dynamically write to the IC_TAR register provided the following conditions are met:

(1) I2C is not enabled (IC_ENABLE=0); OR
(2) I2C is enabled (IC_ENABLE=1); AND
I2C is NOT engaged in any Master (Tx, Rx) operation (IC_STATUS[5]=0); AND
I2C is enabled to operate in Master mode (IC_CON[0]=1); AND
There are NO entries in the Tx FIFO (IC_STATUS[2]=1)

21.4.2.3 Master Transmit and Master Receive

To transmit data, write the data to be written to the lower byte of the I2C Rx/Tx Data Buffer and Command Register (IC_DATA_CMD). The bit[8] (CMD) should be written to 0 for I2C write operations. Subsequently, a read command may be issued by writing "don't cares" to the lower byte of the IC_DATA_CMD register, and a 1 should be written to the CMD bit. The I2C master continues to initiate transfers as long as there are commands present in the transmit FIFO. If the STOP bit set 1, the I2C inserts a STOP condition after completing the current transfer.

The I2C supports switching back and forth between reading and writing dynamically. For combine transfer, a 1 should be write to the RESTART bit of the last command ahead of reverse direction.

All information provided in this document is subject to legal disclaimers.
© REALTEK 2024. All rights reserved.

21.5 Registers

This section describes the programmable registers of the I2C.

Name	Address offset	Access	Description
REG_IC_CON	000h	R/W	I2C Control Register
REG_IC_TAR	004h	R/W	I2C Target Address Register
REG_IC_SAR	008h	R/W	I2C Slave Address Register
REG_IC_HS_MAR	00Ch	R/W	I2C High Speed Master Mode Code Address Register
REG_IC_DATA_CMD	010h	R/W	I2C Rx/Tx Data Buffer and Command Register
REG_IC_SS_SCL_HCNT	014h	R/W	Standard Speed I2C Clock SCL High Count Register
REG_IC_SS_SCL_LCNT	018h	R/W	Standard Speed I2C Clock SCL Low Count Register
REG_IC_FS_SCL_HCNT	01Ch	R/W	Fast Speed I2C Clock SCL High Count Register
REG_IC_FS_SCL_LCNT	020h	R/W	Fast Speed I2C Clock SCL Low Count Register
REG_IC_HS_SCL_HCNT	024h	R/W	High Speed I2C Clock SCL High Count Register
REG_IC_HS_SCL_LCNT	028h	R/W	High Speed I2C Clock SCL Low Count Register
REG_IC_INTR_STAT	02Ch	R	I2C Interrupt Status Register
REG_IC_INTR_MASK	030h	R/W	I2C Interrupt Mask Register
REG_IC_RAW_INTR_STAT	034h	R	I2C Raw Interrupt Status Register
REG_IC_RX_TL	038h	R/W	I2C Receive FIFO Threshold Register
REG_IC_TX_TL	03Ch	R/W	I2C Transmit FIFO Threshold Register
REG_IC_CLR_INTR	040h	R	Clear Combined and Individual Interrupt Register
REG_IC_CLR_RX_UNDER	044h	R	Clear RX_UNDER Interrupt Register
REG_IC_CLR_RX_OVER	048h	R	Clear RX_OVER Interrupt Register
REG_IC_CLR_TX_OVER	04Ch	R	Clear TX_OVER Interrupt Register
REG_IC_CLR_RD_REQ	050h	R	Clear RD_REQ Interrupt Register
REG_IC_CLR_TX_ABRT	054h	R	Clear TX_ABRT Interrupt Register
REG_IC_CLR_RX_DONE	058h	R	Clear RX_DONE Interrupt Register
REG_IC_CLR_ACTIVITY	05Ch	R	Clear ACTIVITY Interrupt Register
REG_IC_CLR_STOP_DET	060h	R	Clear STOP_DET Interrupt Register
REG_IC_CLR_START_DET	064h	R	Clear START_DET Interrupt Register
REG_IC_CLR_GEN_CALL	068h	R	Clear GEN_ALL Interrupt Register
REG_IC_ENABLE	06Ch	R/W	I2C Enable Register
REG_IC_STATUS	070h	R	I2C Status Register
REG_IC_TXFLR	074h	R	I2C Transmit FIFO Level Register
REG_IC_RXFLR	078h	R	I2C Receive FIFO Level Register
REG_IC_SDA_HOLD	07Ch	R/W	I2C SDA Hold Time Length Register
REG_IC_TX_ABRT_SOURCE	080h	R	I2C Transmit Abort Source Register
REG_IC_SLV_DATA_NACK_ONLY	084h	R/W	Generate Slave Data NACK Register
REG_IC_DMA_CR	088h	R/W	DMA Control Register
REG_IC_DMA_TDLR	08Ch	R/W	DMA Transmit Data Level Register
REG_IC_DMA_RDLR	090h	R/W	I2C Receive Data Level Register
REG_IC_SDA_SETUP	094h	R/W	I2C SDA Setup Register
REG_IC_ACK_GENERAL_CALL	098h	R/W	I2C ACK General Call Register
REG_IC_ENABLE_STATUS	09Ch	R	I2C Enable Status Register
REG_IC_DMA_CMD	0A0h	R/W	I2C DMA Command Register
REG_IC_DMA_DATA_LEN	0A4h	R/W	I2C DMA Mode Transfer Data Length Register
REG_IC_DMA_MODE	0A8h	R/W	I2C DMA Mode Register
REG_IC_DEBUG_SEL	0B0h	R/W	I2C Debug SEL Register
REG_IC_OUT_SMP_DLY	0B4h	R/W	Sample Delay Register
REG_IC_CLR_ADDR_MATCH	0E4h	R	Clear Slave Mode Address Match Interrupt Register
REG_IC_CLR_DMA_DONE	0E8h	R	Clear DMA DONE Interrupt Register
REG_IC_FILTER	0ECh	R/W	I2C Filter Register
REG_IC_SAR2	0F4h	R/W	I2C Slave2 Address Register
REG_IC_COMP_VERSION	0FCh	R	I2C Component Version Register
REG_IC_DUMMY	100h	R	IC Dummy Register

All information provided in this document is subject to legal disclaimers.
© REALTEK 2024. All rights reserved.

21.5.1 REG_IC_CON

- **Name:** I2C Control Register
- **Size:** 32 bits
- **Address offset:** 000h
- **Read/write access:** R/W



Bit	Symbol	Access	Reset	Description
31:8	RSVD	-	0	Reserved
7	IC_SLAVE_DISABLE_1	R/W	0x0	This bit controls whether I2C has its slave2 (7-bit address) disabled. **0**: slave2 is enabled **1**: slave2 is disabled
6	IC_SLAVE_DISABLE_0	R/W	0x0	This bit controls whether I2C has its slave1 (7-bit or 10-bit address) disabled. **0**: slave1 is enabled **1**: slave1 is disabled
5	IC_RESTATRT_EN	R/W	0x0	Determine whether RESTART conditions may be sent when acting as a master. **0**: disable **1**: enable
4	RSVD	-	0	Reserved
3	IC_10BITADDR_SLAVE	R/W	0x0	When acting as a slave, this bit controls whether the I2C responds to 7- or 10-bit addresses. **0**: 7-bit addressing **1**: 10-bit addressing
2:1	SPEED	R/W	0x0	These bits control at which speed the I2C operates; its setting is relevant only if one is operating the I2C in master mode. **1**: standard mode (0 to 100kbit/s) **2**: fast mode (<=400kbit/s) **3**: high speed mode (<=3.4Mbit/s)
0	MASTER_MODE	R/W	0x0	This bit controls whether the I2C master is enabled. **0**: master disabled **1**: master enabled

21.5.2 REG_IC_TAR

- **Name:** I2C Target Address Register
- **Size:** 32 bits
- **Address offset:** 004h
- **Read/write access:** R/W



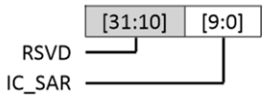
Bit	Symbol	Access	Reset	Description
31:13	RSVD	-	0	Reserved
12	IC_10BITADDR_MASTER	R/W	0x0	Control whether I2C starts its transfers in 7- or 10-bit addressing mode when acting as a master.

All information provided in this document is subject to legal disclaimers.
© REALTEK 2024. All rights reserved.

				0: 7-bit addressing **1**: 10-bit addressing
11	SPECIAL	R/W	0x0	This bit indicates whether software performs a General Call or START BYTE command. **0**: ignore bit 10 GC_OR_START and use IC_TAR normally **1**: perform special I2C command as specified in GC_OR_START bit
10	GC_OR_START	R/W	0x0	If SPECIAL is set to 1, then this bit indicates whether a General Call or START BYTE command is to be performed by I2C. (ic_clk domain) **0**: General Call - after issuing a General Call, only write may be performed **1**: START BYTE
9:0	IC_TAR	R/W	0x10	This is the target address for any master transaction. When transmitting a General Call, these bits are ignored. To generate a START BYTE, the CPU needs to write only once into these bits.

21.5.3 REG_IC_SAR

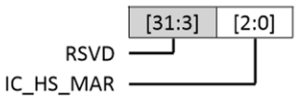
- **Name:** I2C Slave Address Register
- **Size:** 32 bits
- **Address offset:** 008h
- **Read/write access:** R/W



Bit	Symbol	Access	Reset	Description
31:10	RSVD	-	0	Reserved
9:0	IC_SAR	R/W	0x11	The IC_SAR holds the slave address when the I2C is operating as slave1. For 7-bit addressing, only IC_SAR[6:0] is used. The register can be written only when the I2C interface is disabled, which corresponds to the IC_ENABLE register being set to 0. Writes at other times have no effect.

21.5.4 REG_IC_HS_MAR

- **Name:** I2C High Speed Master Mode Code Address Register
- **Size:** 32 bits
- **Address offset:** 00Ch
- **Read/write access:** R/W

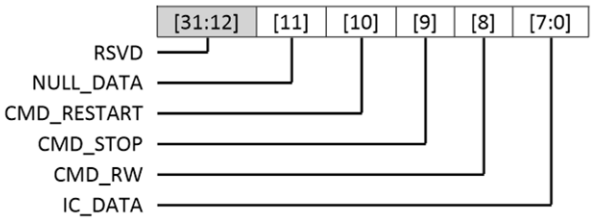


Bit	Symbol	Access	Reset	Description
31:3	RSVD	-	0	Reserved
2:0	IC_HS_MAR	R/W	0x0	This bit field holds the value of the I2C HS mode master code. HS-mode master codes are reserved 8-bit codes (00001xxx) that are not used for slave addressing or other purposes. This register can be written only when the I2C interface is disabled, which corresponds to the IC_ENABLE register being set to 0. Write at other times have no effect.

21.5.5 REG_IC_DATA_CMD

- **Name:** I2C Rx/Tx Data Buffer and Command Register
- **Size:** 32 bits
- **Address offset:** 010h
- **Read/write access:** R/W

All information provided in this document is subject to legal disclaimers.
© REALTEK 2024. All rights reserved.



Bit	Symbol	Access	Reset	Description
31:12	RSVD	-	0	Reserved
11	NULL_DATA	W	0x0	**1**: null data transfer
10	CMD_RESTART	W	0x0	This bit controls whether a RESTART is issued after the byte is sent or received. **1**: a RESTART is issued after the data is sent/received (according to the value of CMD_RW), regardless of whether or not the transfer direction is changing from the previous command.
9	CMD_STOP	W	0x0	This bit controls whether a STOP is issued after the byte is sent or received. **1**: STOP is issued after this byte, regardless of whether or not the Tx FIFO is empty. If the Tx FIFO is not empty, the master immediately tries to start a new transfer by issuing a START and arbitrating for the bus. **0**: STOP is not issued after this byte.
8	CMD_RW	W	0x0	This bit controls whether a read or write is performed. This bit does not control the direction when the I2C acts as a slave. It controls only the direction when it acts as a master. **1**: Read **0**: Write
7:0	IC_DATA	R/W	0x0	This register contains the data to be transmitted or received on the I2C bus. If you are writing to this register and want to perform a read, these bits are ignored by the I2C. However, when you read this register, these bits return the value of data received on the I2C interface.

21.5.6 REG_IC_SS_SCL_HCNT

- **Name:** Standard Speed I2C Clock SCL High Count Register
- **Size:** 32 bits
- **Address offset:** 014h
- **Read/write access:** R/W



Bit	Symbol	Access	Reset	Description
31:16	RSVD	-	0	Reserved
15:0	IC_SS_SCL_HCNT	R/W	0x0190	This register sets the SCL clock high-period count for standard speed. This register must be set before any I2C bus transaction can take place to ensure proper I/O timing. This register can be written only when the I2C interface is disabled which corresponds to the IC_ENABLE register being set to 0. Writes at other times have no effect.

21.5.7 REG_IC_SS_SCL_LCNT

- **Name:** Standard Speed I2C Clock SCL Low Count Register
- **Size:** 32 bits
- **Address offset:** 018h
- **Read/write access:** R/W



All information provided in this document is subject to legal disclaimers.
© REALTEK 2024. All rights reserved.

Bit	Symbol	Access	Reset	Description
31:16	RSVD	-	0	Reserved
15:0	IC_SS_SCL_LCNT	R/W	0x01D6	This register sets the SCL clock low-period count for standard speed. This register must be set before any I2C bus transaction can take place to ensure proper I/O timing. This register can be written only when the I2C interface is disabled which corresponds to the IC_ENABLE register being set to 0. Writes at other times have no effect.

21.5.8 REG_IC_FS_SCL_HCNT

- **Name:** Fast Speed I2C Clock SCL High Count Register
- **Size:** 32 bits
- **Address offset:** 01Ch
- **Read/write access:** R/W



Bit	Symbol	Access	Reset	Description
31:16	RSVD	-	0	Reserved
15:0	IC_FS_SCL_HCNT	R/W	0x003C	This register sets the SCL clock high-period count for fast speed. This register must be set before any I2C bus transaction can take place to ensure proper I/O timing. This register can be written only when the I2C interface is disabled which corresponds to the IC_ENABLE register being set to 0. Writes at other times have no effect.

21.5.9 REG_IC_FS_SCL_LCNT

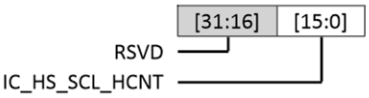
- **Name:** Fast Speed I2C Clock SCL Low Count Register
- **Size:** 32 bits
- **Address offset:** 020h
- **Read/write access:** R/W



Bit	Symbol	Access	Reset	Description
31:16	RSVD	-	0	Reserved
15:0	IC_FS_SCL_LCNT	R/W	0x0082	This register sets the SCL clock low-period count for fast speed. This register must be set before any I2C bus transaction can take place to ensure proper I/O timing. This register can be written only when the I2C interface is disabled which corresponds to the IC_ENABLE register being set to 0. Writes at other times have no effect.

21.5.10 REG_IC_HS_SCL_HCNT

- **Name:** High Speed I2C Clock SCL High Count Register
- **Size:** 32 bits
- **Address offset:** 024h
- **Read/write access:** R/W



All information provided in this document is subject to legal disclaimers.
© REALTEK 2024. All rights reserved.

Bit	Symbol	Access	Reset	Description
31:16	RSVD	-	0	Reserved
15:0	IC_HS_SCL_HCNT	R/W	0x0006	This register sets the SCL clock high-period count for high speed. This register must be set before any I2C bus transaction can take place to ensure proper I/O timing. This register can be written only when the I2C interface is disabled which corresponds to the IC_ENABLE register being set to 0. Writes at other times have no effect. The SCL High time depends on the loading of the bus. For 100pF loading, the SCL High time is 60ns; for 400pF loading, the SCL High time is 120ns.

21.5.11 REG_IC_HS_SCL_LCNT

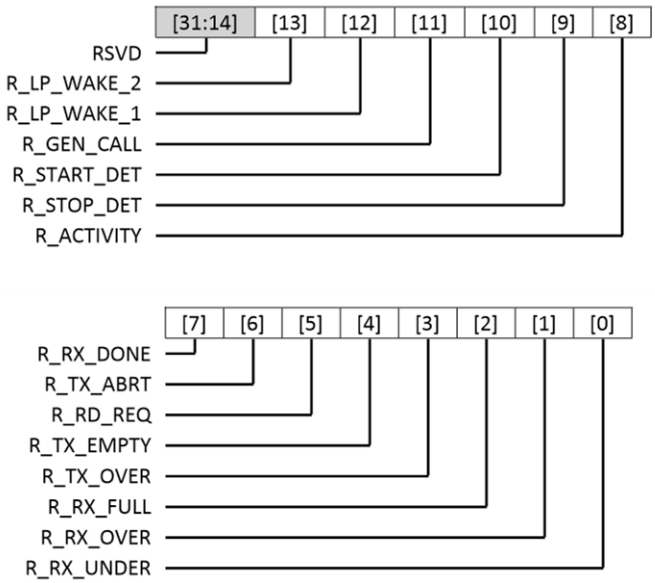
- **Name:** High Speed I2C Clock SCL Low Count Register
- **Size:** 32 bits
- **Address offset:** 028h
- **Read/write access:** R/W



Bit	Symbol	Access	Reset	Description
31:16	RSVD	-	0	Reserved
15:0	IC_HS_SCL_LCNT	R/W	0x0010	This register sets the SCL clock low-period count for high speed. This register must be set before any I2C bus transaction can take place to ensure proper I/O timing. This register can be written only when the I2C interface is disabled which corresponds to the IC_ENABLE register being set to 0. Writes at other times have no effect. The SCL Low time depends on the loading of the bus. For 100pF loading, the SCL High time is 160ns; for 400pF loading, the SCL High time is 320ns.

21.5.12 REG_IC_INTR_STAT

- **Name:** I2C Interrupt Status Register
- **Size:** 32 bits
- **Address offset:** 02Ch
- **Read/write access:** R

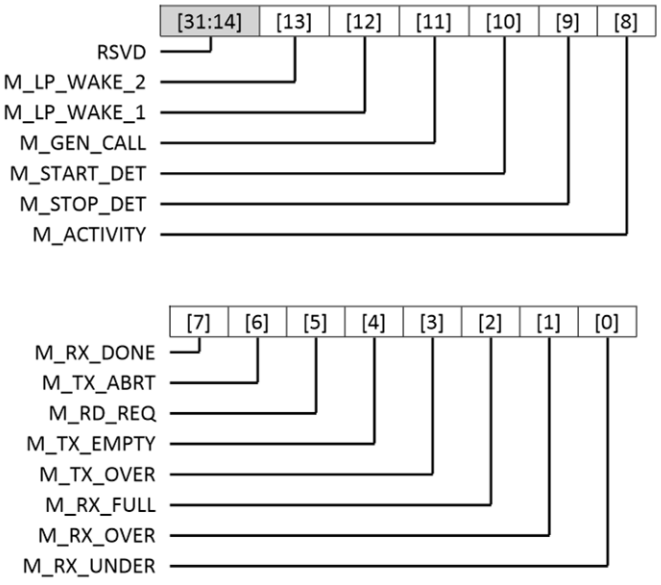


All information provided in this document is subject to legal disclaimers.
© REALTEK 2024. All rights reserved.

Bit	Symbol	Access	Reset	Description
31:14	RSVD	-	0	Reserved
13	R_LP_WAKE_2	R	0x0	Refer to I2C Raw Interrupt Status Register (IC_RAW_INTR_STAT) for a detailed description of this bit.
12	R_LP_WAKE_1	R	0x0	Refer to I2C Raw Interrupt Status Register (IC_RAW_INTR_STAT) for a detailed description of this bit.
11	R_GEN_CALL	R	0x0	Refer to I2C Raw Interrupt Status Register (IC_RAW_INTR_STAT) for a detailed description of this bit.
10	R_START_DET	R	0x0	Refer to I2C Raw Interrupt Status Register (IC_RAW_INTR_STAT) for a detailed description of this bit.
9	R_STOP_DET	R	0x0	Refer to I2C Raw Interrupt Status Register (IC_RAW_INTR_STAT) for a detailed description of this bit.
8	R_ACTIVITY	R	0x0	Refer to I2C Raw Interrupt Status Register (IC_RAW_INTR_STAT) for a detailed description of this bit.
7	R_RX_DONE	R	0x0	Refer to I2C Raw Interrupt Status Register (IC_RAW_INTR_STAT) for a detailed description of this bit.
6	R_TX_ABRT	R	0x0	Refer to I2C Raw Interrupt Status Register (IC_RAW_INTR_STAT) for a detailed description of this bit.
5	R_RD_REQ	R	0x0	Refer to I2C Raw Interrupt Status Register (IC_RAW_INTR_STAT) for a detailed description of this bit.
4	R_TX_EMPTY	R	0x0	Refer to I2C Raw Interrupt Status Register (IC_RAW_INTR_STAT) for a detailed description of this bit.
3	R_TX_OVER	R	0x0	Refer to I2C Raw Interrupt Status Register (IC_RAW_INTR_STAT) for a detailed description of this bit.
2	R_RX_FULL	R	0x0	Refer to I2C Raw Interrupt Status Register (IC_RAW_INTR_STAT) for a detailed description of this bit.
1	R_RX_OVER	R	0x0	Refer to I2C Raw Interrupt Status Register (IC_RAW_INTR_STAT) for a detailed description of this bit.
0	R_RX_UNDER	R	0x0	Refer to I2C Raw Interrupt Status Register (IC_RAW_INTR_STAT) for a detailed description of this bit.

21.5.13 REG_IC_INTR_MASK

- **Name:** I2C Interrupt Mask Register
- **Size:** 32 bits
- **Address offset:** 030h
- **Read/write access:** R/W

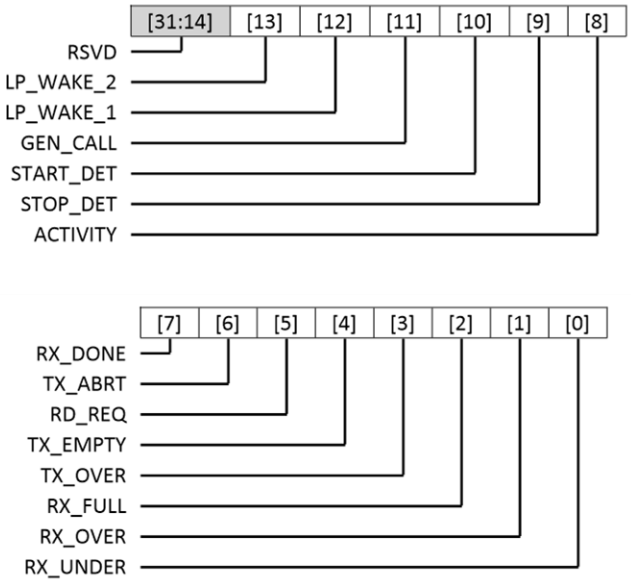


All information provided in this document is subject to legal disclaimers.
© REALTEK 2024. All rights reserved.

Bit	Symbol	Access	Reset	Description
31:14	RSVD	-	0	Reserved
13	M_LP_WAKE_2	R/W	0x0	This bit masks the corresponding interrupt status bit in the IC_INTR_STAT register.
12	M_LP_WAKE_1	R/W	0x0	This bit masks the corresponding interrupt status bit in the IC_INTR_STAT register.
11	M_GEN_CALL	R/W	0x0	This bit masks the corresponding interrupt status bit in the IC_INTR_STAT register.
10	M_START_DET	R/W	0x0	This bit masks the corresponding interrupt status bit in the IC_INTR_STAT register.
9	M_STOP_DET	R/W	0x0	This bit masks the corresponding interrupt status bit in the IC_INTR_STAT register.
8	M_ACTIVITY	R/W	0x0	This bit masks the corresponding interrupt status bit in the IC_INTR_STAT register.
7	M_RX_DONE	R/W	0x0	This bit masks the corresponding interrupt status bit in the IC_INTR_STAT register.
6	M_TX_ABRT	R/W	0x0	This bit masks the corresponding interrupt status bit in the IC_INTR_STAT register.
5	M_RD_REQ	R/W	0x0	This bit masks the corresponding interrupt status bit in the IC_INTR_STAT register.
4	M_TX_EMPTY	R/W	0x0	This bit masks the corresponding interrupt status bit in the IC_INTR_STAT register.
3	M_TX_OVER	R/W	0x0	This bit masks the corresponding interrupt status bit in the IC_INTR_STAT register.
2	M_RX_FULL	R/W	0x0	This bit masks the corresponding interrupt status bit in the IC_INTR_STAT register.
1	M_RX_OVER	R/W	0x0	This bit masks the corresponding interrupt status bit in the IC_INTR_STAT register.
0	M_RX_UNDER	R/W	0x0	This bit masks the corresponding interrupt status bit in the IC_INTR_STAT register.

21.5.14 REG_IC_RAW_INTR_STAT

- **Name:** I2C Raw Interrupt Status Register
- **Size:** 32 bits
- **Address offset:** 034h
- **Read/write access:** R



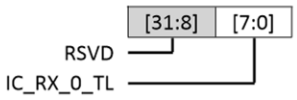
Bit	Symbol	Access	Reset	Description
31:14	RSVD	-	0	Reserved
13	LP_WAKE_2	R	0x0	Set when address SAR_2 matches with address sending on I2C BUS.
12	LP_WAKE_1	R	0x0	Set when address SAR matches with address sending on I2C BUS.
11	GEN_CALL	R	0x0	Set only when a General Call address is received and it is acknowledged.
10	START_DET	R	0x0	Indicate whether a START or RESTART condition has occurred on the I2C interface regardless of whether I2C is operating in slave or master mode.
9	STOP_DET	R	0x0	Indicate whether a STOP condition has occurred on the I2C interface regardless of whether I2C is operating in slave or master mode.
8	ACTIVITY	R	0x0	This bit captures I2C activity.
7	RX_DONE	R	0x0	When the I2C is acting as a slave-transmitter, this bit is set to 1 if the master does not acknowledge a transmitted byte. This occurs on the last byte of the transmission, indicating that the transmission is done.
6	TX_ABRT	R	0x0	This bit indicates if I2C as a transmitter, is unable to complete the intended actions on the contents of the transmit FIFO. This situation can occur both as an I2C master or an I2C slave,

All information provided in this document is subject to legal disclaimers.
© REALTEK 2024. All rights reserved.

				and is referred to as a 'transmit abort'. **1**: the IC_TX_ABRT_SOURCE register indicates the reason why the transmit abort take places.
5	RD_REQ	R	0x0	This bit is set to 1 when I2C is acting as a slave and another I2C master is attempting to read data from I2C. The I2C holds the I2C bus in a wait state (SCL=0) until this interrupt is serviced, which means that the slave has been addressed by a remote master that is asking for data to be transferred. The processor must respond to this interrupt and then write the requested data to the IC_DATA_CMD register.
4	TX_EMPTY	R	0x0	This bit is set to 1 when the transmit buffer is at or below the threshold value set in the IC_TX_TL register. It is automatically cleared by hardware when the buffer level goes above the threshold.
3	TX_OVER	R	0x0	Set during transmit if the transmit buffer is filled to IC_TX_BUFFER_DEPTH and the processor attempts to issue another I2C command by writing to the IC_DATA_CMD register.
2	RX_FULL	R	0x0	Set when the receive buffer goes above the RX_TL threshold in the IC_RX_TL register. It is automatically cleared by hardware when buffer level is equal to or less than the threshold.
1	RX_OVER	R	0x0	Not used. Set if the receive buffer is completely filled to IC_RX_BUFFER_DEPTH and an additional byte is received from an external I2C device.
0	RX_UNDER	R	0x0	Set if the processor attempts to read the receive buffer when it is empty by reading from the IC_DATA_CMD register.

21.5.15 REG_IC_RX_TL

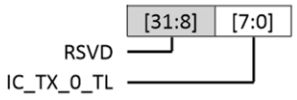
- **Name:** I2C Receive FIFO Threshold Register
- **Size:** 32 bits
- **Address offset:** 038h
- **Read/write access:** R/W



Bit	Symbol	Access	Reset	Description
31:8	RSVD	-	0	Reserved
7:0	IC_RX_0_TL	R/W	0x0	Receive FIFO Threshold Level. Control the level of entries (or above) that triggers the RX_FULL interrupt (bit[2] in IC_RAW_INTR_STAT register).

21.5.16 REG_IC_TX_TL

- **Name:** I2C Transmit FIFO Threshold Register
- **Size:** 32 bits
- **Address offset:** 03Ch
- **Read/write access:** R/W



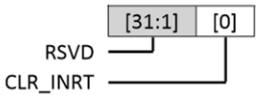
Bit	Symbol	Access	Reset	Description
31:8	RSVD	-	0	Reserved
7:0	IC_TX_0_TL	R/W	0x0	Transmit FIFO Threshold Level. Control the level of entries (or below) that triggers the TX_EMPTY interrupt (bit[4] in IC_RAW_INTR_STAT register).

21.5.17 REG_IC_CLR_INTR

- **Name:** Clear Combined and Individual Interrupt Register
- **Size:** 32 bits

All information provided in this document is subject to legal disclaimers.
© REALTEK 2024. All rights reserved.

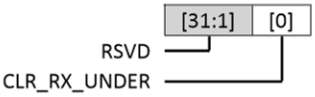
- **Address offset:** 040h
- **Read/write access:** R



Bit	Symbol	Access	Reset	Description
31:1	RSVD	-	0	Reserved
0	CLR_INRT	R	0x0	Read this register to clear the combined interrupt, all individual interrupts, and the IC_TX_ABRT_SOURCE register. This bit does not clear hardware clearable interrupts but software clearable interrupts.

21.5.18 REG_IC_CLR_RX_UNDER

- **Name:** Clear RX_UNDER Interrupt Register
- **Size:** 32 bits
- **Address offset:** 044h
- **Read/write access:** R



Bit	Symbol	Access	Reset	Description
31:1	RSVD	-	0	Reserved
0	CLR_RX_UNDER	R	0x0	Read this register to clear the RX_UNDER interrupt (bit[0]) of IC_RAW_INTR_STAT register.

21.5.19 REG_IC_CLR_RX_OVER

- **Name:** Clear RX_OVER Interrupt Register
- **Size:** 32 bits
- **Address offset:** 048h
- **Read/write access:** R



Bit	Symbol	Access	Reset	Description
31:1	RSVD	-	0	Reserved
0	CLR_RX_OVER	R	0x0	Read this register to clear the RX_OVER interrupt (bit[1]) of IC_RAW_INTR_STAT register.

21.5.20 REG_IC_CLR_TX_OVER

- **Name:** Clear TX_OVER Interrupt Register
- **Size:** 32 bits
- **Address offset:** 04Ch
- **Read/write access:** R



All information provided in this document is subject to legal disclaimers.
© REALTEK 2024. All rights reserved.

Bit	Symbol	Access	Reset	Description
31:1	RSVD	-	0	Reserved
0	CLR_TX_OVER	R	0x0	Read this register to clear the TX_OVER interrupt (bit[3]) of IC_RAW_INTR_STAT register.

21.5.21 REG_IC_CLR_RD_REQ

- **Name:** Clear RD_REQ Interrupt Register
- **Size:** 32 bits
- **Address offset:** 050h
- **Read/write access:** R



Bit	Symbol	Access	Reset	Description
31:1	RSVD	-	0	Reserved
0	CLR_RD_REQ	R	0x0	Read this register to clear the RD_REQ interrupt (bit[5]) of IC_RAW_INTR_STAT register.

21.5.22 REG_IC_CLR_TX_ABRT

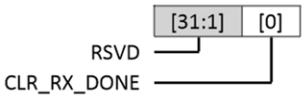
- **Name:** Clear TX_ABRT Interrupt Register
- **Size:** 32 bits
- **Address offset:** 054h
- **Read/write access:** R



Bit	Symbol	Access	Reset	Description
31:1	RSVD	-	0	Reserved
0	CLR_TX_ABRT	R	0x0	Read this register to clear the TX_ABRT interrupt (bit[6]) of IC_RAW_INTR_STAT register and the IC_TX_ABRT_SOURCE register. This also releases the TxFIFO from the flushed/reset state, allowing more writes to the Tx FIFO.

21.5.23 REG_IC_CLR_RX_DONE

- **Name:** Clear RX_DONE Interrupt Register
- **Size:** 32 bits
- **Address offset:** 058h
- **Read/write access:** R



Bit	Symbol	Access	Reset	Description
31:1	RSVD	-	0	Reserved
0	CLR_RX_DONE	R	0x0	Read this register to clear the RX_DONE interrupt (bit[7]) of IC_RAW_INTR_STAT register.

21.5.24 REG_IC_CLR_ACTIVITY

- **Name:** Clear ACTIVITY Interrupt Register
- **Size:** 32 bits
- **Address offset:** 05Ch
- **Read/write access:** R

All information provided in this document is subject to legal disclaimers.
© REALTEK 2024. All rights reserved.



Bit	Symbol	Access	Reset	Description
31:1	RSVD	-	0	Reserved
0	CLR_ACTIVITY	R	0x0	Read this register to clear the ACTIVITY interrupt if the I2C is not active anymore. If the I2C module is still active on the bus, the ACTIVITY interrupt bit continues to be set. It is automatically cleared by hardware if the module is disabled and if there is no further activity on the bus. The value read from this register to get status of the ACTIVITY interrupt (bit[8]) of the IC_RAW_INTR_STAT register.

21.5.25 REG_IC_CLR_STOP_DET

- **Name:** Clear STOP_DET Interrupt Register
- **Size:** 32 bits
- **Address offset:** 060h
- **Read/write access:** R



Bit	Symbol	Access	Reset	Description
31:1	RSVD	-	0	Reserved
0	CLR_STOP_DET	R	0x0	Read this register to clear the STOP_DET interrupt (bit[9]) of IC_RAW_INTR_STAT register.

21.5.26 REG_IC_CLR_START_DET

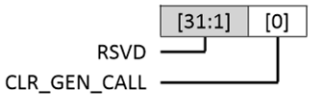
- **Name:** Clear START_DET Interrupt Register
- **Size:** 32 bits
- **Address offset:** 064h
- **Read/write access:** R



Bit	Symbol	Access	Reset	Description
31:1	RSVD	-	0	Reserved
0	CLR_START_DET	R	0x0	Read this register to clear the START_DET interrupt (bit[10]) of IC_RAW_INTR_STAT register.

21.5.27 REG_IC_CLR_GEN_CALL

- **Name:** Clear GEN_ALL Interrupt Register
- **Size:** 32 bits
- **Address offset:** 068h
- **Read/write access:** R

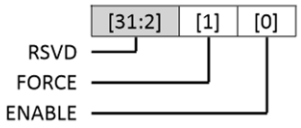


Bit	Symbol	Access	Reset	Description
31:1	RSVD	-	0	Reserved
0	CLR_GEN_CALL	R	0x0	Read this register to clear the GEN_CALL interrupt (bit[11]) of IC_RAW_INTR_STAT register.

All information provided in this document is subject to legal disclaimers.
© REALTEK 2024. All rights reserved.

21.5.28 REG_IC_ENABLE

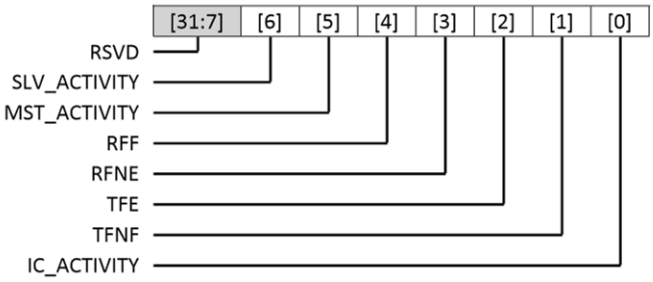
- **Name:** I2C Enable Register
- **Size:** 32 bits
- **Address offset:** 06Ch
- **Read/write access:** R/W



Bit	Symbol	Access	Reset	Description
31:2	RSVD	-	0	Reserved
1	FORCE	R/W	0x0	Force master I2C stop transfer immediately after IC_ENABLE set 0. **1**: when IC_ENABLE is set 0 during transfer, force master FSM go to IDLE immediately. **0**: when IC_ENABLE is set 0 during transfer, master FSM while finish current transition (include RESTART) before go to IDLE.
0	ENABLE	R/W	0x0	Control whether the I2C is enabled. **0**: Disable I2C (Tx and Rx FIFOs are held in an erased state) **1**: Enable I2C

21.5.29 REG_IC_STATUS

- **Name:** I2C Status Register
- **Size:** 32 bits
- **Address offset:** 070h
- **Read/write access:** R



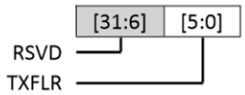
Bit	Symbol	Access	Reset	Description
31:7	RSVD	-	0	Reserved
6	SLV_ACTIVITY	R	0x0	Slave FSM Activity Status. When the Slave FSM is not in the IDLE state, this bit is set. **0**: Slave FSM is in the IDLE state so the slave part of I2C is not active **1**: Slave FSM is not he IDLE state so the slave part of I2C is active
5	MST_ACTIVITY	R	0x0	Master FSM Activity Status. When the Master FSM is not in the IDLE state, this bit is set. **0**: Master FSM is in the IDLE state so the master part of I2C is not active **1**: Master FSM is not he IDLE state so the master part of I2C is active
4	RFF	R	0x0	Receive FIFO Completely Full. When the receive FIFO is completely full, this bit is set. When the receive FIFO contains one or more empty location, this bit is cleared. **0**: Receive FIFO is not full **1**: Receive FIFO is full
3	RFNE	R	0x0	Receive FIFO Not Empty. This bit is set when the receive FIFO contains one or more entries; it is cleared when the receive FIFO is empty. **0**: Receive FIFO is empty **1**: Receive FIFO is not empty
2	TFE	R	0x1	Transmit FIFO Completely Empty. When the transmit FIFO is completely empty, this bit is set. When it contains one or more valid entries, this bit is cleared. This bit field does not request an interrupt. **0**: Transmit FIFO is not empty

All information provided in this document is subject to legal disclaimers.
© REALTEK 2024. All rights reserved.

				1: Transmit FIFO is empty
1	TFNF	R	0x1	Transmit FIFO Not Full. Set when the transmit FIFO contains one or more empty locations, and is cleared when the FIFO is full. **0**: Transmit FIFO is full **1**: Transmit FIFO is not full
0	IC_ACTIVITY	R	0x0	I2C Activity Status.

21.5.30 REG_IC_TXFLR

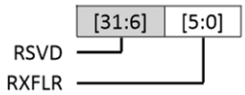
- **Name:** I2C Transmit FIFO Level Register
- **Size:** 32 bits
- **Address offset:** 074h
- **Read/write access:** R



Bit	Symbol	Access	Reset	Description
31:6	RSVD	-	0	Reserved
5:0	TXFLR	R	0x0	Transmit FIFO Level. Contains the number of valid data entries in the transmit FIFO.

21.5.31 REG_IC_RXFLR

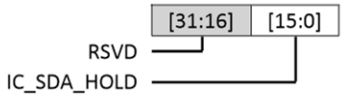
- **Name:** I2C Receive FIFO Level Register
- **Size:** 32 bits
- **Address offset:** 078h
- **Read/write access:** R



Bit	Symbol	Access	Reset	Description
31:6	RSVD	-	0	Reserved
5:0	RXFLR	R	0x0	Receive FIFO Level. Contains the number of valid data entries in the receive FIFO.

21.5.32 REG_IC_SDA_HOLD

- **Name:** I2C SDA Hold Time Length Register
- **Size:** 32 bits
- **Address offset:** 07Ch
- **Read/write access:** R/W



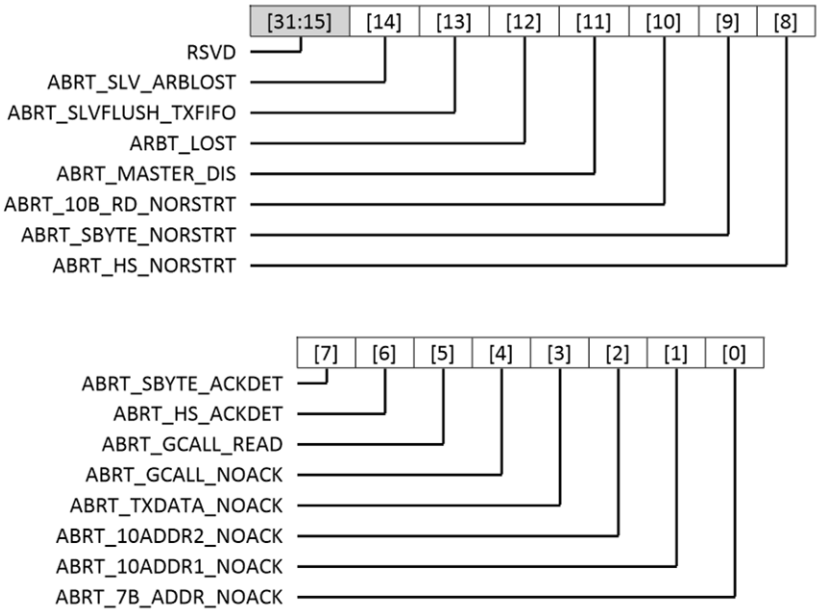
Bit	Symbol	Access	Reset	Description
31:16	RSVD	-	0	Reserved
15:0	IC_SDA_HOLD	R/W	0x1	Set the required SDA hold time in units of ic_clk period.

21.5.33 REG_IC_TX_ABRT_SOURCE

- **Name:** I2C Transmit Abort Source Register
- **Size:** 32 bits

All information provided in this document is subject to legal disclaimers.
© REALTEK 2024. All rights reserved.

- **Address offset:** 080h
- **Read/write access:** R

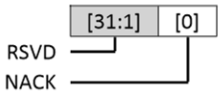


Bit	Symbol	Access	Reset	Description
31:15	RSVD	-	0	Reserved
14	ABRT_SLV_ARBLOST	R	0x0	**1**: Slave losts the bus while transmitting data to a remote master. IC_TX_ABRT_SOURCE[12] is set at the same time.
13	ABRT_SLVFLUSH_TXFIFO	R	0x0	**1**: Slave has received a read command and some data exists in the TXFIFO so the slave issues a TX_ABRT interrupt to flush old data in Tx FIFO.
12	ARBT_LOST	R	0x0	**1**: Master has lost arbitration, or if IC_TX_ABRT_SOURCE[14] is also set, then the slave transmitter has lost arbitration.
11	ABRT_MASTER_DIS	R	0x0	**1**: User tries to initiate a Master operation with the master mode disabled.
10	ABRT_10B_RD_NORSTRT	R	0x0	**1**: The restart is disabled and the master sends a read command in 10-bit addressing mode.
9	ABRT_SBYTE_NORSTRT	R	0x0	**1**: The restart is disabled and the user is trying to send a START Byte.
8	ABRT_HS_NORSTRT	R	0x0	**1**: The restart is disabled and the user is trying to use the master to transfer data in High Speed mode.
7	ABRT_SBYTE_ACKDET	R	0x0	**1**: Master has send a START Byte and the START Byte was acknowledged (wrong behavior).
6	ABRT_HS_ACKDET	R	0x0	**1**: Master is in High Speed mode and the High Speed Master code was acknowledged (wrong behavior).
5	ABRT_GCALL_READ	R	0x0	**1**: I2C in master mode sents a General Call but the user programmed the byte following the General Call to be a read from the bus.
4	ABRT_GCALL_NOACK	R	0x0	**1**: I2C in master mode sents a General Call and no slave on the bus acknowledged the General Call.
3	ABRT_TXDATA_NOACK	R	0x0	**1**: This is a master-mode only bit. Master has received an acknowledgement for the address, but when it sent data byte(s) following the address, it did not receive an acknowledge from the remote slave(s).
2	ABRT_10ADDR2_NOACK	R	0x0	**1**: Master is in 10-bit address mode and the second byte of the 10-bit address was not acknowledged by any slave.
1	ABRT_10ADDR1_NOACK	R	0x0	**1**: Master is in 10-bit address mode and the first 10-bit address byte was not acknowledged by any slave.
0	ABRT_7B_ADDR_NOACK	R	0x0	**1**: Master is in 7-bit addressing mode and the address sent was not acknowledged by any slave

All information provided in this document is subject to legal disclaimers.
© REALTEK 2024. All rights reserved.

21.5.34 REG_IC_SLV_DATA_NACK_ONLY

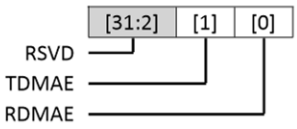
- **Name:** Generate Slave Data NACK Register
- **Size:** 32 bits
- **Address offset:** 084h
- **Read/write access:** R/W



Bit	Symbol	Access	Reset	Description
31:1	RSVD	-	0	Reserved
0	NACK	R/W	0x0	Generate NACK. This NACK generation only occurs when I2C is a slave-receiver. If this register is set to a value of 1, it can only generate a NACK after a data byte is received; hence, the data transfer is aborted and the data received is not pushed to the receive buffer. When the register is set to a value of 0, it generates NACK/ACK, depending on normal criteria. **1**: generate NACK after data byte received **0**: generate NACK/ACK normally

21.5.35 REG_IC_DMA_CR

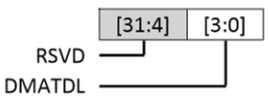
- **Name:** DMA Control Register
- **Size:** 32 bits
- **Address offset:** 088h
- **Read/write access:** R/W



Bit	Symbol	Access	Reset	Description
31:2	RSVD	-	0	Reserved
1	TDMAE	R/W	0x0	Transmit DMA Enable. This bit enables/disables the transmit FIFO DMA channel. **0**: Transmit DMA disabled **1**: Transmit DMA enabled
0	RDMAE	R/W	0x0	Receive DMA Enable. This bit enables/disables the receive FIFO DMA channel. **0**: Receive DMA disabled **1**: Receive DMA enabled

21.5.36 REG_IC_DMA_TDLR

- **Name:** DMA Transmit Data Level Register
- **Size:** 32 bits
- **Address offset:** 08Ch
- **Read/write access:** R/W

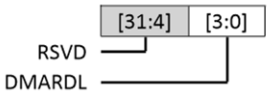


Bit	Symbol	Access	Reset	Description
31:4	RSVD	-	0	Reserved
3:0	DMATDL	R/W	0x0	Transmit Data Level. This bit field controls the level at which a DMA request is made by the transmit logic. It is equal to the watermark level; that is, the dma_tx_req signal is generated when the number of valid data entries in the transmit FIFO is equal to or below this field value, and TDMAE=1.

All information provided in this document is subject to legal disclaimers.
© REALTEK 2024. All rights reserved.

21.5.37 REG_IC_DMA_RDLR

- **Name:** I2C Receive Data Level Register
- **Size:** 32 bits
- **Address offset:** 090h
- **Read/write access:** R/W



Bit	Symbol	Access	Reset	Description
31:4	RSVD	-	0	Reserved
3:0	DMARDL	R/W	0x0	Receive Data Level. This bit field controls the level at which a DMA request is made by the receive logic. The watermark level=DMARDL+1; that is, the dma_tx_req signal is generated when the number of valid data entries in the receive FIFO is equal to or more than this field value+1, and RDMAE=1. For instance, when DMARDL is 0, the dma_rx_req is asserted when 1 or more data entries are present in the receive FIFO.

21.5.38 REG_IC_SDA_SETUP

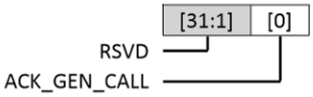
- **Name:** I2C SDA Setup Register
- **Size:** 32 bits
- **Address offset:** 094h
- **Read/write access:** R/W



Bit	Symbol	Access	Reset	Description
31:8	RSVD	-	0	Reserved
7:0	IC_SDA_SETUP	R/W	0x5	Set the required SDA setup time in units of ic_clk period.

21.5.39 REG_IC_ACK_GENERAL_CALL

- **Name:** I2C ACK General Call Register
- **Size:** 32 bits
- **Address offset:** 098h
- **Read/write access:** R/W

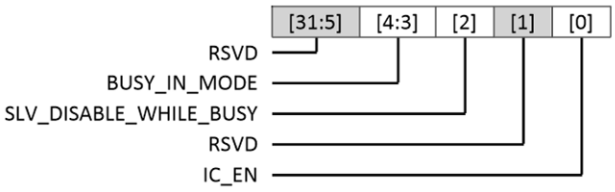


Bit	Symbol	Access	Reset	Description
31:1	RSVD	-	0	Reserved
0	ACK_GEN_CALL	R/W	0x1	**1**: I2C responds with an ACK when it receives a General Call. **0**: I2C does not generate General Call interrupts.

21.5.40 REG_IC_ENABLE_STATUS

- **Name:** I2C Enable Status Register
- **Size:** 32 bits
- **Address offset:** 09Ch
- **Read/write access:** R

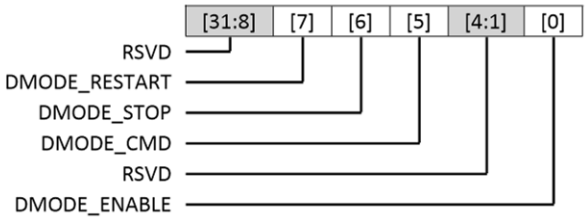
All information provided in this document is subject to legal disclaimers.
© REALTEK 2024. All rights reserved.



Bit	Symbol	Access	Reset	Description
31:5	RSVD	-	0	Reserved
4:3	BUSY_IN_MODE	R	0x0	**00**: I2C is disable while busy in legacy mode. **01**: I2C is disable while busy in DMA mode. **10**: I2C is disable while busy in Descriptor mode.
2	SLV_DISABLE_WHILE_BUSY	R	0x0	Slave Disabled While Busy (Transmit, Receive). This bit indicates if a potential or active Slave operation has been aborted due to the setting of the IC_SLAVE_DISBALE register from 1 to 0.
1	RSVD	-	0	Reserved
0	IC_EN	R	0x0	IC_EN Status. This bit always reflects the value driven on the output port IC_EN. **1**: I2C is deemed to be in an enabled state. **0**: I2C is deemed completely inactive.

21.5.41 REG_IC_DMA_CMD

- **Name:** I2C DMA Command Register
- **Size:** 32 bits
- **Address offset:** 0A0h
- **Read/write access:** R/W

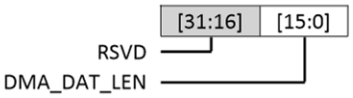


Bit	Symbol	Access	Reset	Description
31:8	RSVD	-	0	Reserved
7	DMODE_RESTART	R/W	0x0	This bit controls whether a RESTART is issued after the byte is sent or received in DMA mode. **1**: a RESTART is issued after the data is sent/received (according to the value of CMD_RW), regardless of whether or not the transfer direction is changing from the previous command.
6	DMODE_STOP	R/W	0x0	This bit controls whether a STOP is issued after the byte is sent or received in DMA mode. **1**: STOP is issued after this byte, regardless of whether or not the Tx FIFO is empty. If the Tx FIFO is not empty, the master immediately tries to start a new transfer by issuing a START and arbitrating for the bus. **0**: STOP is not issued after this byte.
5	DMODE_CMD	R/W	0x0	This bit controls whether a read or write is performed in DMA mode. This bit does not control the direction when the I2C acts as a slave. It controls only the direction when it acts as a master. **1**: Read **0**: Write
4:1	RSVD	-	0	Reserved
0	DMODE_ENABLE	WA0	0x0	**1**: Set to enable DMA mode, cleared when transfer is done

All information provided in this document is subject to legal disclaimers.
© REALTEK 2024. All rights reserved.

21.5.42 REG_IC_DMA_DATA_LEN

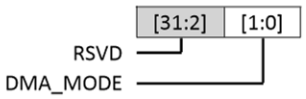
- **Name:** I2C DMA Mode Transfer Data Length Register
- **Size:** 32 bits
- **Address offset:** 0A4h
- **Read/write access:** R/W



Bit	Symbol	Access	Reset	Description
31:16	RSVD	-	0	Reserved
15:0	DMA_DAT_LEN	R/W	0x0	DMA transfer data length

21.5.43 REG_IC_DMA_MODE

- **Name:** I2C DMA Mode Register
- **Size:** 32 bits
- **Address offset:** 0A8h
- **Read/write access:** R/W



Bit	Symbol	Access	Reset	Description
31:2	RSVD	-	0	Reserved
1:0	DMA_MODE	R/W	0x0	**0**: DWC DMA legacy mode **1**: DMA with control register **2**: DMA with transfer descriptor

21.5.44 REG_IC_DEBUG_SEL

- **Name:** I2C Debug SEL Register
- **Size:** 32 bits
- **Address offset:** 0B0h
- **Read/write access:** R/W



Bit	Symbol	Access	Reset	Description
31:4	RSVD	-	0	Reserved
3:0	IC_DEBUG_SEL	R/W	0x0	**0**: Debug clock **1**: Debug APB **2**: RSVD **3**: Debug FIFO **4**: Debug Timing **5**: Debug Slave Mode Address Match **6**: Debug Interrupt

All information provided in this document is subject to legal disclaimers.
© REALTEK 2024. All rights reserved.

21.5.45 REG_IC_OUT_SMP_DLY

- **Name:** Sample Delay Register
- **Size:** 32 bits
- **Address offset:** 0B4h
- **Read/write access:** R/W



Bit	Symbol	Access	Reset	Description
31:3	RSVD	-	0	Reserved
2:0	IC_OUT_SMP_DLY	R/W	0x2	Delay cycles for sample of master derived SCL value. The sample of master derived SCL is used for slave stretch. Slave stretch is valid when master drive SCL High, but slave drive SCL low. Valid value of IC_OUT_SMP_DLY is 0-7 indicates a delay of 1 to 8 cycles.

21.5.46 REG_IC_CLR_ADDR_MATCH

- **Name:** Clear Slave Mode Address Match Interrupt Register
- **Size:** 32 bits
- **Address offset:** 0E4h
- **Read/write access:** R



Bit	Symbol	Access	Reset	Description
31:1	RSVD	-	0	Reserved
0	IC_CLR_ADDR_MATCH	R	0x0	Read this register to clear the slave mode address match interrupt (bit[13:12]) of IC_RAW_INTR_STAT register.

21.5.47 REG_IC_CLR_DMA_DONE

- **Name:** Clear DMA DONE Interrupt Register
- **Size:** 32 bits
- **Address offset:** 0E8h
- **Read/write access:** R

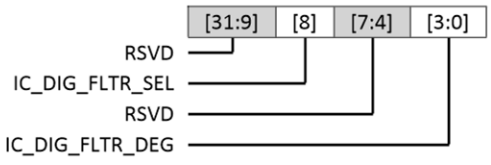


Bit	Symbol	Access	Reset	Description
31:1	RSVD	-	0	Reserved
0	IC_CLR_RAW_INTR_STAT	R	0x0	Read this register to clear the DMA_DONE interrupt (bit[15]) of IC_RAW_INTR_STAT register.

21.5.48 REG_IC_FILTER

- **Name:** I2C Filter Register
- **Size:** 32 bits
- **Address offset:** 0ECh
- **Read/write access:** R/W

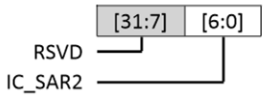
All information provided in this document is subject to legal disclaimers.
© REALTEK 2024. All rights reserved.



Bit	Symbol	Access	Reset	Description
31:9	RSVD	-	0	Reserved
8	IC_DIG_FLTR_SEL	R/W	0x0	1: Enable filter
7:4	RSVD	-	0	Reserved
3:0	IC_DIG_FLTR_DEG	R/W	0x0	DIG_FLTR_DEG is to define frequency range of filter. A greater value of DIG_FLTR_DEG results in a slower transfer speed and hardware would be able to filter a lower frequency.

21.5.49 REG_IC_SAR2

- **Name:** I2C Slave2 Address Register
- **Size:** 32 bits
- **Address offset:** 0F4h
- **Read/write access:** R/W



Bit	Symbol	Access	Reset	Description
31:7	RSVD	-	0	Reserved
6:0	IC_SAR2	R/W	0x12	The IC_SAR2 holds the slave address when the I2C is operating as slave2. Slave2 only supports 7-bit address. The register can be written only when the I2C interface is disabled, which corresponds to the IC_ENABLE register being set to 0. Writes at other times have no effect.

21.5.50 REG_IC_COMP_VERSION

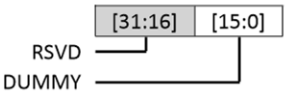
- **Name:** I2C Component Version Register
- **Size:** 32 bits
- **Address offset:** 0FCh
- **Read/write access:** R



Bit	Symbol	Access	Reset	Description
31:0	IC_COMP_VERSION	R	0x20200702	I2C version number

21.5.51 REG_IC_DUMMY

- **Name:** IC Dummy Register
- **Size:** 32 bits
- **Address offset:** 100h
- **Read/write access:** R



All information provided in this document is subject to legal disclaimers.
© REALTEK 2024. All rights reserved.

Bit	Symbol	Access	Reset	Description
31:16	RSVD	-	0	Reserved
15:0	DUMMY	R	0x0	Reserved for HW

All information provided in this document is subject to legal disclaimers.
© REALTEK 2024. All rights reserved.

22 Universal Asynchronous Receiver/Transmitter (UART)

22.1 Overview

The Universal Asynchronous Receiver/Transmitter (UART) offers a flexible means of full-duplex data exchange with external equipment, requiring an industry-standard NRZ asynchronous serial data format. It offers a very wide range of baud rates using a fractional baud rate generator. Low power Rx mode is implemented by monitoring Rx baud rate error and own frequency drift.

22.1.1 Features

- UART x 4
  - Two general UART interfaces: UART0 and UART1
  - One internal BT UART interface: UART2_BT
  - One specific UART as LOGUART
- Various UART format: 1 start bit, 7/8 data bits, 0/1 parity bit and 1/2 stop bits
- Separated clocks for Tx path and Rx path
  - Tx path: XTAL 40MHz
  - Rx path: XTAL 40MHz, OSC 2MHz
- Fractional baud rate
  - Up to 8Mbps within high-speed mode (XTAL 40MHz)
  - Up to 115.2Kbps within low-power mode (OSC 2MHz)
- Asynchronous Transmit/Receive FIFO
- Configurable Auto Flow Control
- Interrupt control and error detection
- IrDA (SIR mode) encoder and decoder module
- Loop-back mode for test
- Low power mode for Rx path
- Monitor and elimination of Rx baud rate error and own frequency drift automatically for Rx path
- UART Rx timeout mechanism
- DMA interface for DMA transfer
  - DMA as DMA TRx flow controller
  - UART as DMA Rx flow controller
- Operation mode
  - Polling mode
  - Interrupt mode
  - DMA mode

22.1.2 Block Diagram

The block diagram of UART is illustrated in *Figure 22-1*.

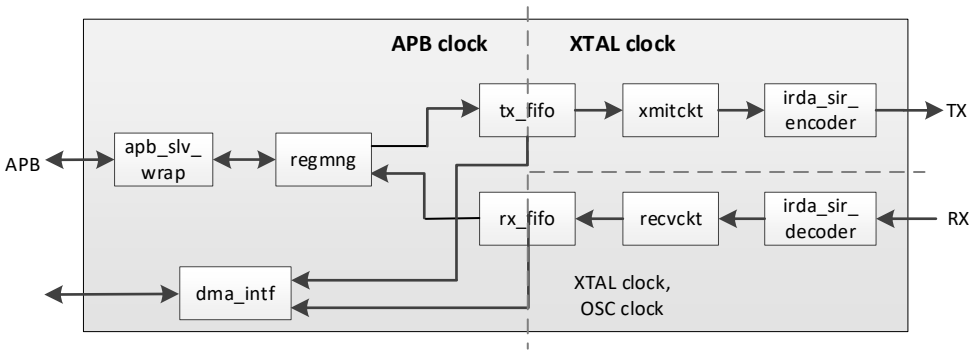


Figure 22-1 UART block diagram

All information provided in this document is subject to legal disclaimers.
© REALTEK 2024. All rights reserved.

The UART consists of several sub-modules, which provide the following functions:

- dma_intf
  - DMA control unit
  - Interface with GDMA module
- tx_fifo
  - Asynchronous FIFO
  - 16 * 11bits for UART0/1 and 32 * 11bits for UART2_BT
- xmitckt
  - Shift register
  - Generate UART format data
  - Support fractional baud rate
- irda_sir_encoder
  - SIR encoder of UART Tx data
- rx_fifo
  - Asynchronous FIFO
  - 64 * 11bits for UART0/1 and 32 * 11bits for UART2_BT
- recvckt
  - Receive UART format data
  - Error detection
  - Interrupt control
  - Support fractional baud rate
  - Oscillator (OSC) clock for low power
  - Monitor and eliminate Rx baud rate error and own frequency drift
- irda_sir_decoder
  - SIR decoder of UART Rx data
- apb_slv_wrap
  - APB3 bus interface
- regmng
  - Register control unit

22.2 Functional Description

This section describes the functional operation of the UART.

22.2.1 UART Serial Protocol

Because the serial communication between the UART and the selected device is asynchronous, additional bits (start bit and stop bit) are added to the serial data to indicate the beginning and end. Utilizing these bits allows two devices to be synchronized. This structure of serial data accompanied by start and stop bits is referred to as a character, as shown in *Figure 22-2*.

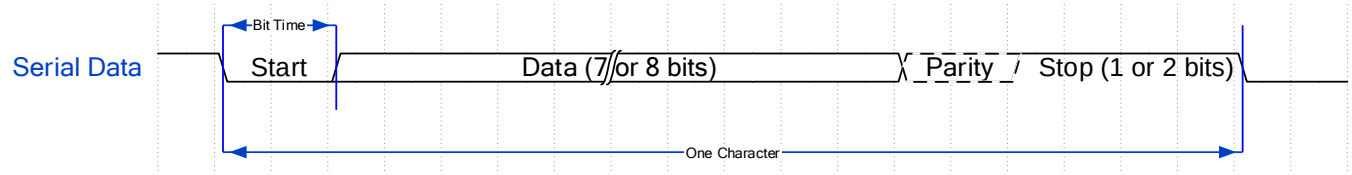


Figure 22-2 Serial data format

An additional parity bit may be added to the serial character. This bit appears after the last data bit and before the stop bit(s) in the character structure to provide the UART with the ability to perform simple error checking on the received data. The UART Line Control Register (LCR) is used to control the serial character characteristics. The individual bits of the data word (7 or 8 bits) are sent after the start bit, starting with the least-significant bit (LSB). These are followed by the optional parity bit (odd parity, even parity or stick parity), followed by the stop bit(s), which can be 1 or 2.

22.2.2 IrDA (SIR) Protocol

The Infrared Data Association (IrDA) Serial Infrared (SIR) mode supports bi-directional data communication with remote devices using infrared radiation as the transmission medium. IrDA 1.0 SIR mode specifies a maximum baud rate of 115.2K.

The data format is similar to the standard UART data format. Each data character is sent serially, beginning with a start bit, followed by 7 or

All information provided in this document is subject to legal disclaimers.
© REALTEK 2024. All rights reserved.

8 data bits and ending with 1 or 2 stop bits. The parity bit is invalid in IrDA mode, try to enable parity with the Line Control Register (LCR) will add one wrong parity bit in the serial data. When UART is configured to support IrDA, it can be enabled with MISC Control Register (MISCR) bit 0.

Transmitting a single infrared pulse signals a logic zero, while a logic one is represented by not sending a pulse. Thus, each new character begins with an infrared pulse for the start bit. However, received data is inverted from transmitted data due to the infrared pulses energizing the photo transistor base of the IrDA receiver, pulling its output low. This inverted transistor output is then fed to the UART Rx port, which then has correct UART polarity. The IrDA output and input logic can be reversed by setting MISCR bit19 and bit20. The width of each pulse is 3/16ths of a normal serial bit time. The IrDA Tx width can also be adjusted by configuring register SIR_TX_PWC0 with software. *Figure 22-3* shows the timing diagram of the IrDA SIR Data format in comparison with the standard UART format.

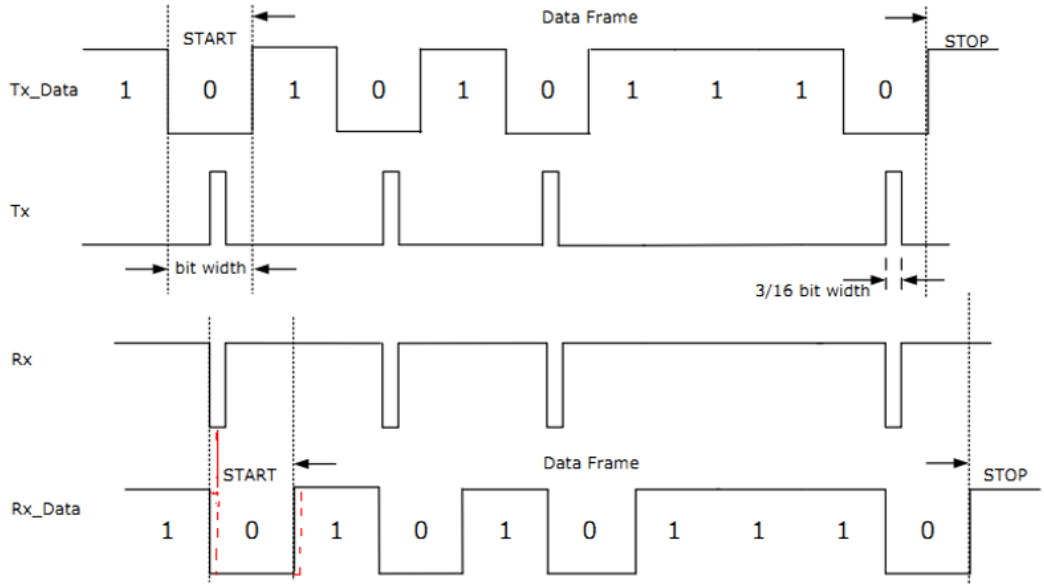


Figure 22-3 Relationship between IrDA signal and UART signal

22.2.3 Interrupt

The UART supports both individual and combined interrupt requests, which can both be masked as well. The combined interrupt request is the ORed result of all other UART interrupts after masking. The system designer has the choice of routing individual interrupt requests or only the combined interrupt request to the Interrupt Controller. The UART interrupts are described as follows:

- Received Data Available Interrupt (ERBFI) – Set when Rx FIFO Trigger Level is reached or Rx FIFO is full. Rx FIFO Trigger Level can be configured by software in FIFO Control Register (FCR, bit[7:6]), defines the level of receiver FIFO entries at which an interrupt is generated. This interrupt is cleared by hardware when Rx FIFO entries decrease under the trigger level. Bit RXFIFO_INT in Line Status Register (LSR, bit8) will be set when this interrupt happens.
- Transmitter FIFO Empty Interrupt (ETBEI) – Set when the transmit FIFO is empty. This interrupt is cleared by hardware when data is written into the Tx FIFO buffer. Bit TX_EMPTY in Line Status Register (LSR, bit5) will be set when this interrupt happens.
- Receiver Line Status Interrupt (ELSI) – Set when overrun, parity error, framing error or break condition is detected on the Rx line. This interrupt remains set until the Receiver Line Status Interrupt Clear Flag (RLSICF) in Interrupt Clear Register (ICR) is set by the user. Bit OVR_ERR (LSR, bit1), bit PAR_ERR (LSR, bit2), bit FRM_ERR (LSR, bit3) and bit BREAK_INT (LSR, bit4) will be set accordingly when detecting overrun, parity error, framing error and break condition on the Rx line. Bit RXFIFO_ERR (LSR, bit7) will be set when any of the events mentioned before happens. Writing 1 to RLSICF will also clear RXFIFO_ERR status.
- Modem Status Interrupt (EDSSI) – Set when a falling edge is detected on CTS line with hardware auto-flow enabled. This interrupt remains set until the Modem Interrupt Clear Flag (MICF) in ICR is set by the user. Bit MODEM_INT (LSR, bit11) will be set when this interrupt happens.
- RX Path Monitor Done Interrupt (EMDI) – Set when monitor process is over with Rx path monitor function enabled. This interrupt remains set until the Monitor Done Interrupt Clear Flag (MDICF) in ICR is set by the user. Bit MONITOR_DONE_INT (LSR, bit10) will be set when this interrupt happens.
- RX Timeout Interrupt (ETOI) – Set when no character has been input to the Rx FIFO or read from it before time expires when there's at least one character in the Rx FIFO. Time can be set in Rx Path Control Register (RX_PATH_CTRL, bit[31:16]). This interrupt remains set until the Timeout Interrupt Clear Flag (TOICF) in ICR is set by the user. Bit TIMEOUT_INT (LSR, bit9) will be set when this interrupt happens.
- RX Noise Detected Interrupt (ERXNDI) – Set when Rx debouncer has detected noise. This interrupt remains set until the Rx Noise Detected Interrupt Clear Flag (RXNDICF) in ICR is set by the user. Bit RXND_INT (LSR, bit12) will be set when this interrupt happens.

All information provided in this document is subject to legal disclaimers.
© REALTEK 2024. All rights reserved.

22.2.4 Auto Flow Control

The UART can be configured to work under hardware data flow control, which is comprised of Auto RTS and Auto CTS. Enable Auto Flow Control by asserting AFE bit in the Modem Control Register (MCR, bit5). Auto RTS and Auto CTS are described as follows:

22.2.4.1 Auto RTS

Auto RTS becomes active when the following occurs:
- Auto Flow Control is enabled (MCR[5] is set)
- RTS output is enabled (MCR[1] is set)
- FIFOs are enabled (FCR[0] is set)

When Auto RTS is enabled (active), RTS signal is forced inactive (high) automatically once the receiver FIFO reaches Rx FIFO Trigger Level. When RTS output is connected to the CTS input of the sending device (e.g. another UART), the sending device stops sending serial data until the receiver FIFO has available space (until it is completely empty).

The optional receiver FIFO threshold values are 1, ¼, ½, and "2 less than full". Since one additional character may be transmitted to the UART after RTS has become inactive as shown in *Figure 22-4* (due to data in the other UART has already entered the transmitter block before it recognize the deassertion of RTS), setting the threshold to "2 less than full" makes the best use of the FIFO with a safety zone of one character.

Once the receiver FIFO becomes completely empty by reading the Receiver Buffer Register (RBR_OR_THR), RTS becomes active (low) again, signaling the other UART to continue sending data. It is important to note that even if everything else is selected and the correct MCR bits are set, if the FIFOs are disabled through FCR bit0, Auto Flow Control is also disabled. When Auto RTS is not implemented or disabled, RTS is controlled solely by MCR bit1. *Figure 22-4* shows a timing diagram of Auto RTS operation.

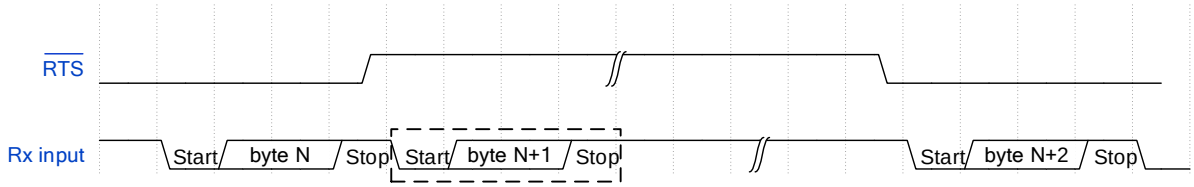


Figure 22-4 Auto RTS timing

NOTE

UART0/UART1/UART2_BT all support Auto RTS while only UART0 and UART2_BT are designed with RTS pin.

22.2.4.2 Auto CTS

Auto CTS becomes active when the following occurs:
- Auto Flow Control is enabled (MCR[5] is set)
- RTS output is enabled (MCR[1] is set)
- FIFOs are enabled (FCR[0] is set)
- SIR mode is disabled (MISCR[0] bit is cleared)

When Auto CTS is enabled (active), the UART transmitter is disabled automatically once the CTS input becomes inactive (high). This prevents overflowing the FIFO of the receiving device (e.g. another UART). If the CTS input is not inactivated before the middle of the last stop bit, another character is transmitted before the transmitter is disabled. While the transmitter is disabled, the transmitter FIFO can still be written to, and even overflowed. Therefore, when using this mode, the following happens:
- The UART status register can be read to check if the transmit FIFO is full (LSR bit6 = 0) or
- The Programmable ERBFI Interrupt can be enabled to access the "FIFO full" status via the Line Status Register (LSR).

When using the "FIFO full" status, the software can poll this before each write to the transmitter FIFO. When the CTS input becomes active (low) again, transmission resumes. It is important to note that even if everything else is selected, if the FIFOs are disabled via FCR[0], Auto Flow Control is also disabled. When Auto CTS is not implemented or disabled, the transmitter is unaffected by CTS. A Timing Diagram showing Auto CTS operation can be seen in *Figure 22-5*.

All information provided in this document is subject to legal disclaimers.
© REALTEK 2024. All rights reserved.

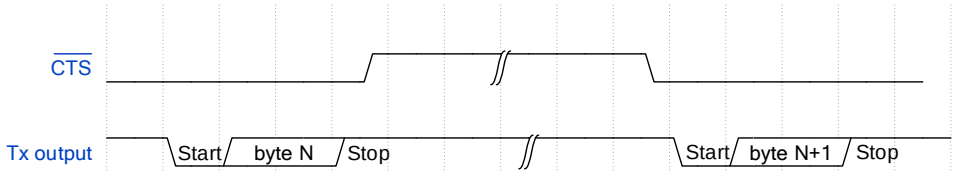


Figure 22-5 Auto CTS timing

NOTE

UART0/UART1/UART2_BT all support Auto CTS while only UART0 and UART2_BT are designed with CTS pin.

22.2.5 DMA Flow Control

GDMA is always DMA flow controller for UART Tx, while either GDMA or UART can both be DMA flow controller for UART Rx. Only Rx DMA mode is illustrated here.

When GDMA is DMA flow controller as shown in *Figure 22-6*, GDMA sets expected block length before blocks transfer, and terminates block transfer when Rx length is equal to block length. rx_dma_req is asserted when Rx size in Rx FIFO is not less than burst size, and rx_dma_single is asserted when Rx FIFO is not empty. rx_dma_ack signal is asserted after the data phase of the last AHB transfer (single or burst) in the current transaction has been completed. When GDMA is the flow controller, rx_dma_last is not sampled and is ignored.

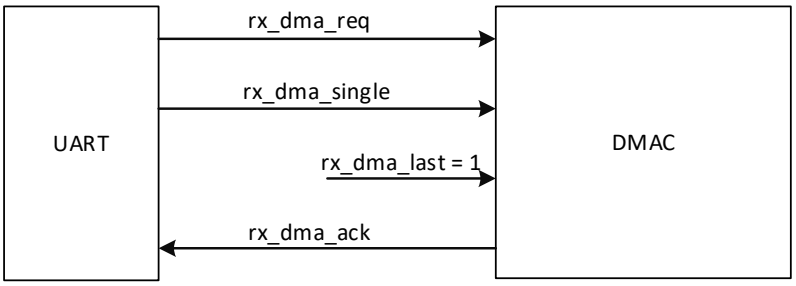


Figure 22-6 GDMA is DMA flow controller

When UART is DMA flow controller as shown in *Figure 22-7*, UART asserts rx_dma_last to terminate block transfer when UART has received the whole packet. The finish flag is defined that no new Rx character comes in for R_RXTO_THRS time in register UART_RX_PATH_CTRL after the last Rx character.

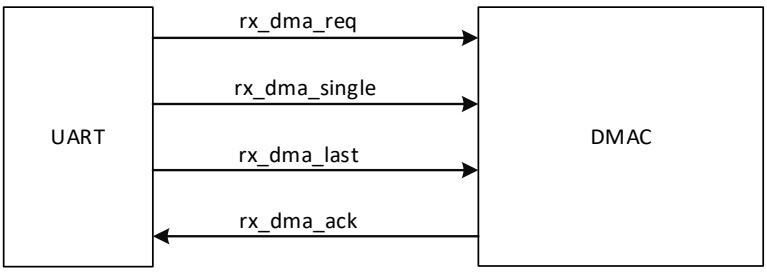


Figure 22-7 UART is DMA flow controller

22.2.6 Rx Path Monitor Function

UART Rx path supports fractional baud rate generator and it can monitor and eliminate Rx baud rate error and own frequency drift. Fractional baud rate and baud rate monitor cannot be used simultaneously.

When the monitor function is enabled (UART_MON_BAUD_CTRL, bit[0]), software sets the original cycle number according to default baud rate, then Rx circuit counts total cycle number M of N bit duration, so average cycle number per bit duration L=M/N, the maximum value of N is 127. Hardware will replace the original cycle number per bit with the newly calculated value L, thus baud rate error from peer device and own frequency deviation can be almost eliminated.

NOTE

When Rx path monitor function is enabled, baud rate should not be greater than 1/17 of current LP clock frequency.

All information provided in this document is subject to legal disclaimers.
© REALTEK 2024. All rights reserved.

22.2.7 Clock Structure of Rx Path

The clock structure of Rx path for UART0, UART1, UART2_BT and LOGUART is similar. *Figure 22-8* is the clock structure of UARTx Rx clock, where x can be from 0 to 2. In *Figure 22-8*, cksl_uart signal in REG_CKSL_GRP0 register decides clock for UARTx Rx path.

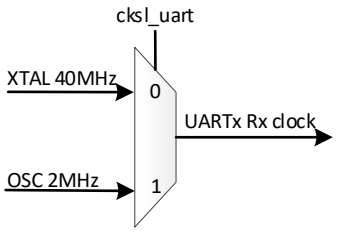


Figure 22-8 Clock structure of UARTx Rx path

22.2.8 UART Rx Debounce

UART supports Rx debounce to filter glitches in Rx signals. The function could be enabled by asserting DBNC_FEN bit in UART_RXDBCR register. Rx debounce clock source is the same with Rx clock source, which could be XTAL 40MHz, XTAL 20MHz or OSC 4MHz. DBNC_CYC field in UART_RXDBCR register is set to debounce Rx signals with different glitches.

When DBNC_FEN bit in UART_RXDBCR register is asserted, glitches shorter than DBNC_CYC debounce clock cycles will be filtered as shown in *Figure 22-9*. Meanwhile, delay will be introduced to Rx signal for (DBNC_CYC + 2) debounce clock cycles.

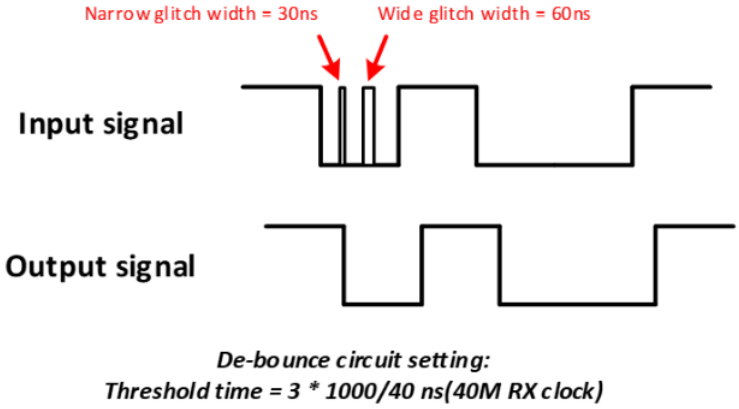
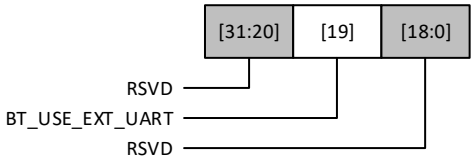


Figure 22-9 UART Rx debounce

22.2.9 UART2_BT Used as Normal UART

UART2_BT is used for internal BT UART by default. For normal UART usage, set the relevant register described below.

- **Name**: REG_LSYS_BT_CTRL0
- **Size**: 32 bits
- **Address**: 0x4100_8280
- **Read/write access**: R/W



Bit	Symbol	Access	Reset	Description
19	BT_USE_EXT_UART	R/W	0x0	● 1: BT uses external UART ● 0: BT uses internal UART

All information provided in this document is subject to legal disclaimers.
© REALTEK 2024. All rights reserved.

22.3 Registers

The following table lists the memory map of the UART registers. The base addresses are:

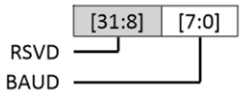
- UART0: 0x4100_C000
- UART1: 0x4100_D000
- UART2_BT: 0x4100_E000
- LOGUART: 0x4100_F000, refer to LOGUART specification for more information

Name	Address offset	Access	Description
REG_UART_DLL	000h	R/W	UART Divisor Length Register
REG_UART_IER	004h	R/W	UART Interrupt Enable Register
REG_UART_IIR	008h	R	UART Interrupt Identification Register
REG_UART_LCR	00Ch	R/W	UART Line Control Register
REG_UART_MCR	010h	R/W	UART Modem Control Register
REG_UART_LSR	014h	R	UART Line Status Register
REG_UART_MSR	018h	R	UART Modem Status Register
REG_UART_SCR	01Ch	R/W	UART Scratch Pad Register
REG_UART_STS	020h	R/W	UART STS Register
REG_UART_RBR_OR_UART_THR	024h	R/W	UART Receiver Buffer Register/UART Transmitter Holding Register
REG_UART_MISCR	028h	R/W	UART MISC Control Register
REG_UART_SIR_TX_PWC0	02Ch	R/W	UART IRDA SIR TX Pulse Width Control 0 Register
REG_UART_SIR_RX_PFC	030h	R/W	UART IRDA SIR Rx Pulse Filter Control Register
REG_UART_BAUD_MON	034h	R/W	UART Baud Monitor Register
REG_UART_DBGR	03Ch	R	UART Debug Register
REG_UART_RX_PATH_CTRL	040h	R/W	UART Rx Path Control Register
REG_UART_MON_BAUD_CTRL	044h	R/W	UART Monitor Baud Rate Control Register
REG_UART_MON_BAUD_STS	048h	R/W	UART Monitor Baud Rate Status Register
REG_UART_MON_CYC_NUM	04Ch	R	UART Monitored Cycle Number Register
REG_UART_RX_BYTE_CNT	050h	R/W	UART Rx Data Byte Count Register
REG_UART_FCR	054h	R/W	UART FIFO Control Register
REG_UART_ICR	058h	R/W	UART Interrupt Clear Register
REG_UART_RXDBCR	05Ch	R/W	UART RX Debounce Control Register
REG_UART_DUMMY	060h	R	UART Dummy Register

22.3.1 REG_UART_DLL

- **Name:** UART Divisor Length Register
- **Size:** 32 bits
- **Address offset:** 000h
- **Read/write access:** R/W

The low 8 bits of divisor coefficient only valid when the DLAB bit of LCR bit[7] = 1.



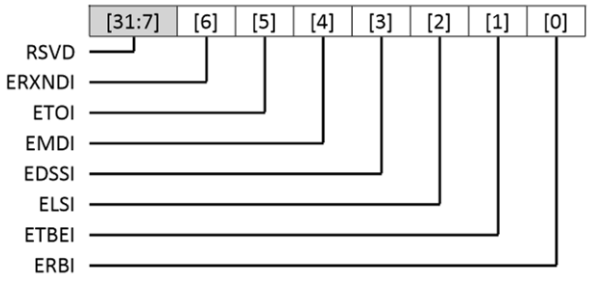
Bit	Symbol	Access	Reset	Description
31:8	RSVD	-	0	Reserved
7:0	BAUD	R/W	0x0	The low 8 bits of divisor coefficient (not used)

22.3.2 REG_UART_IER

- **Name:** UART Interrupt Enable Register
- **Size:** 32 bits
- **Address offset:** 004h
- **Read/write access:** R/W

All information provided in this document is subject to legal disclaimers.
© REALTEK 2024. All rights reserved.

Generate enable and disable interrupt by the UART, only valid when the DLAB bit of LCR bit[7] = 0.

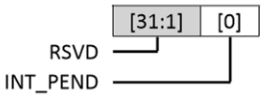


Bit	Symbol	Access	Reset	Description
31:7	RSVD	-	0	Reserved
6	ERXNDI	R/W	0x0	Rx noise detected interrupt enable
5	ETOI	R/W	0x0	Rx timeout interrupt enable
4	EMDI	R/W	0x0	Rx path monitor done interrupt enable
3	EDSSI	R/W	0x0	Modem Status Interrupt (EDSSI) (modem status transition) enable
2	ELSI	R/W	0x0	Receiver Line Status Interrupt (ELSI) (receiver line status) enable
1	ETBEI	R/W	0x0	Transmitter FIFO Empty interrupt (ETBEI) (Tx FIFO empty) enable
0	ERBI	R/W	0x0	Received Data Available Interrupt (ERBFI) (Rx trigger) enable

22.3.3 REG_UART_IIR

- **Name:** UART Interrupt Identification Register
- **Size:** 32 bits
- **Address offset:** 008h
- **Read/write access:** R

Interrupt source indication



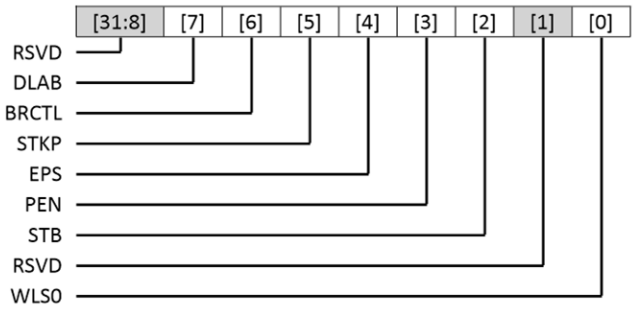
Bit	Symbol	Access	Reset	Description
31:1	RSVD	-	0	Reserved
0	INT_PEND	R	0x1	**0**: pending, and the IR contents may be used as a pointer to the appropriate interrupt service routine. **1**: no interrupt is pending.

22.3.4 REG_UART_LCR

- **Name:** UART Line Control Register
- **Size:** 32 bits
- **Address offset:** 00Ch
- **Read/write access:** R/W

Set the general communication parameters

All information provided in this document is subject to legal disclaimers.
© REALTEK 2024. All rights reserved.



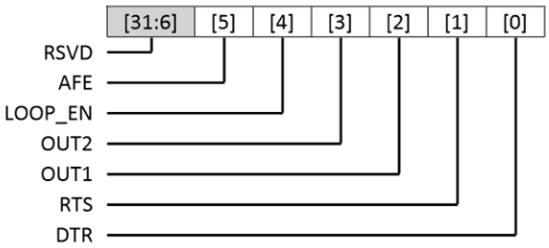
Bit	Symbol	Access	Reset	Description
31:8	RSVD	-	0	Reserved
7	DLAB	R/W	0x0	Divisor Latch Access bit, and control the accessibility of the DLL, DLM and IER registers **0**: The divisor latches not accessible; IER accessible **1**: The divisor latches accessible; DLL and DLM accessible NOTE *THR/RBR don't care the value of this bit.*
6	BRCTL	R/W	0x0	Break Control bit **0**: break signal disable **1**: break signal enable (logic '0')
5	STKP	R/W	0x0	Stick Parity bit **0**: Not stick parity, the parity type is ODD or EVEN (depending on EPS) **1**: Stick parity EPS=1: low parity (1'b0) EPS=0: high parity (1'b1)
4	EPS	R/W	0x0	Even Parity selection **0**: Odd parity, the number of logic '1' is odd. **1**: Even parity, the number of logic '1' is even.
3	PEN	R/W	0x0	Parity Enable **0**: No parity bit **1**: Parity bit is generated on each outgoing character and is checked on each incoming one
2	STB	R/W	0x0	Stop bits length selection **0**: 1 stop bit **1**: 2 stop bits NOTE *The receiver always checks the first stop bit only.*
1	RSVD	-	0	Reserved
0	WLS0	R/W	0x1	Word length selection **0**: 7 bits data **1**: 8 bits data

22.3.5 REG_UART_MCR

- **Name:** UART Modem Control Register
- **Size:** 32 bits
- **Address offset:** 010h
- **Read/write access:** R/W

It is used to perform handshaking actions with the attached device.

All information provided in this document is subject to legal disclaimers.
© REALTEK 2024. All rights reserved.

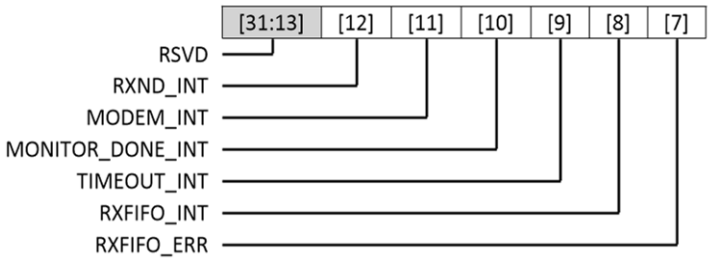


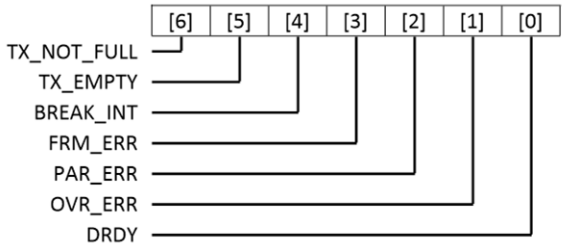
Bit	Symbol	Access	Reset	Description
31:6	RSVD	-	0	Reserved
5	AFE	R/W	0x0	Auto flow control enable
4	LOOP_EN	R/W	0x0	LoopBack mode **0**: Normal operation mode **1**: Loopback mode
3	OUT2	R/W	0x0	Auxiliary output 2 Bit[3] affects the OUT2_output in a manner identical to that described above set to 0. In loopback mode, connected to Data Carrier Detect (DCD).
2	OUT1	R/W	0x0	Auxiliary output 1 Bit3 affects the OUT1_output in a manner identical to that described above set to 0. In loopback mode, connected to Ring Indicator (RI).
1	RTS	R/W/ES	0x0	Request to Send (RTS) signal control **0**: RTS is logic 1. **1**: RTS is logic 0. This bit controls the RTS_output. Bit[1] affects the RTS_ output in a manner identical to that described above set to 0.
0	DTR	R/W	0x0	**0**: DTR is logic 1. **1**: DTR is logic 0. This bit controls the DTR output.

22.3.6 REG_UART_LSR

- **Name:** UART Line Status Register
- **Size:** 32 bits
- **Address offset:** 014h
- **Read/write access:** R

Indicate the current state of communication and errors





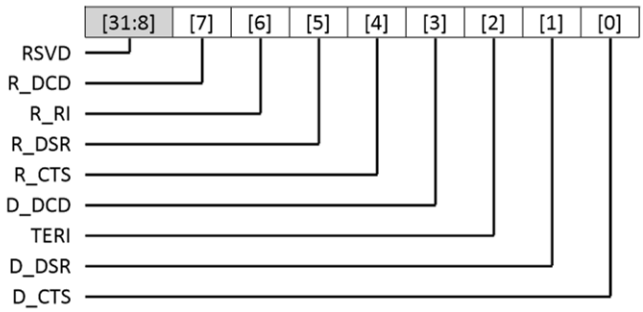
All information provided in this document is subject to legal disclaimers.
© REALTEK 2024. All rights reserved.

Bit	Symbol	Access	Reset	Description
31:13	RSVD	-	0	Reserved
12	RXND_INT	R	0	This bit is only set when Rx debouncer has detected noise. 1: Rx path noise detected interrupt
11	MODEM_INT	R	0	This bit is only set when Modem status is pending. 1: Modem interrupt
10	MONITOR_DONE_INT	R	0	This bit is only set when Rx Path Monitor Done interrupt is enabled (EMDI is set). 1: Rx path monitor done
9	TIMEOUT_INT	R	0	This bit is only set when Rx Timeout interrupt is enabled (ETOI is set). 1: at least one character in the FIFO but no character has been input to the FIFO or reading from it for the time duration, which depends on the value in register REG_RX_PATH_CTRL[31:16].
8	RXFIFO_INT	R	0	This bit is only set when Received Data Available Interrupt interrupt is enabled (ERBI is set). 1: FIFO Trigger level reached or Rx FIFO full
7	RXFIFO_ERR	R	0	Uart_rx_error, depending on the FIFO control register (FCR) bit[0] **0**: cleared when the CPU reads the LSR, if there are no subsequent errors in the FIFO. **1**: parity error, overrun error, framing error or break indication in the FIFO.
6	TX_NOT_FULL	R	1	TXFIFO not full indicator
5	TX_EMPTY	R	1	TXFIFO empty indicator
4	BREAK_INT	R		Break Interrupt (BI) indicator **0**: No break condition in the current character **1**: The received data input held in the Spacing (logic 0) state is longer than a full word transmission time
3	FRM_ERR	R		Framing Error (FE) indicator **0**: No framing error in the current character **1**: The received character does not have a valid stop bit
2	PAR_ERR	R		Parity Error (PE) indicator **0**: No parity error in current character **1**: The received character does not have the correct parity bit
1	OVR_ERR	R		Overrun Error (OE) indicator **0**: No overrun state **1**: The transmission is continue although the Rx FIFO is full
0	DRDY	R		**0**: No characters in the Receiver FIFO **1**: At least one character has been received and transferred into the FIFO

22.3.7 REG_UART_MSR

- **Name:** UART Modem Status Register
- **Size:** 32 bits
- **Address offset:** 018h
- **Read/write access:** R

It contains information about the four incoming modem control lines on the device.



Bit	Symbol	Access	Reset	Description
31:8	RSVD	-	0	Reserved
7	R_DCD	R		Complement of the DCD input or equals to Out2 in loopback mode.

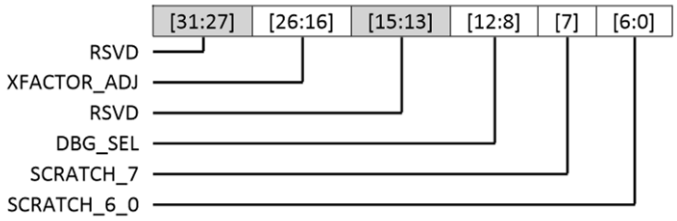
All information provided in this document is subject to legal disclaimers.
© REALTEK 2024. All rights reserved.

6	R_RI	R		Complement of the RI input or equals to Out1 in loopback mode.
5	R_DSR	R		Complement of the DSR input or equals to DTR in loopback mode.
4	R_CTS	R		Complement of the CTS input or equals to RTS in loopback mode.
3	D_DCD	R		**0**: Otherwise **1**: The DCD line has changed its state
2	TERI	R		**0**: Otherwise **1**: The RI line has changed its state from low to high state
1	D_DSR	R		**0**: Otherwise **1**: The DSR line has changed its state
0	D_CTS	R		**0**: Otherwise **1**: The CTS line has changed its state

22.3.8 REG_UART_SCR

- **Name:** UART Scratch Pad Register
- **Size:** 32 bits
- **Address offset:** 01Ch
- **Read/write access:** R/W

It is used to store Tx information.

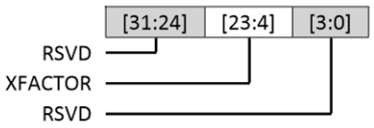


Bit	Symbol	Access	Reset	Description
31:27	RSVD	-	0	Reserved
26:16	XFACTOR_ADJ	R/W	0x0	One factor of baud rate calculation for Tx path
15:13	RSVD	-	0	Reserved
12:8	DBG_SEL	R/W	0x0	Debug port selection
7	SCRATCH_7	R/W/ES	0x0	Clear Rx break signal interrupt status
6:0	SCRATCH_6_0	R/W	0x30	Bit[6]: Rx break signal interrupt enable Bit[5]: fl_set_bi_err Bit[4]: fl_frame_err Bit[3]: pin_lb_test Bit[2:0]: reserved

22.3.9 REG_UART_STS

- **Name:** UART STS Register
- **Size:** 32 bits
- **Address offset:** 020h
- **Read/write access:** R/W

It is used to indicate the information of Tx baud rate.



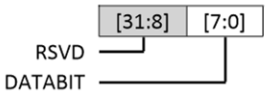
Bit	Symbol	Access	Reset	Description
31:24	RSVD	-	0	Reserved
23:4	XFACTOR	R/W	0x1047	Factor of baud rate calculation for Tx path
3:0	RSVD	-	0	Reserved

All information provided in this document is subject to legal disclaimers.
© REALTEK 2024. All rights reserved.

22.3.10 REG_UART_RBR_OR_UART_THR

- **Name:** UART Receiver Buffer Register/UART Transmitter Holding Register
- **Size:** 32 bits
- **Address offset:** 024h
- **Read/write access:** R/W

The buffers of received data/transmitted data

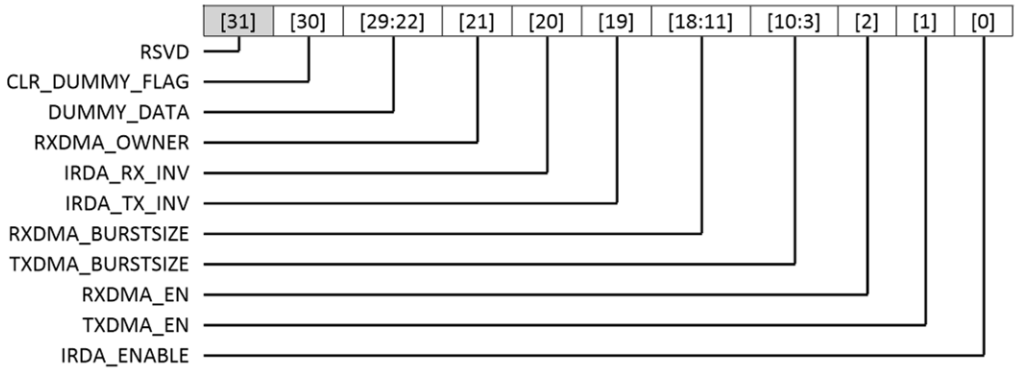


Bit	Symbol	Access	Reset	Description
31:8	RSVD	-	0	Reserved
7:0	DATABIT	R/W	0x0	UART_RBR's Rx data bits or UART_THR's Tx data bit[7:0] NOTE *Bit[0] is the least significant bit. It is the first bit serially received/transmitted.*

22.3.11 REG_UART_MISCR

- **Name:** UART MISC Control Register
- **Size:** 32 bits
- **Address offset:** 028h
- **Read/write access:** R/W

It is used to provide DMA and IrDA related information.



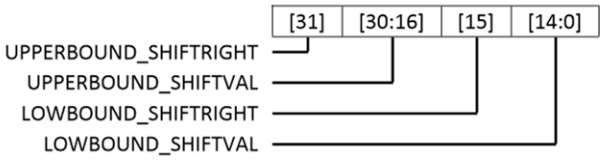
Bit	Symbol	Access	Reset	Description
31	RSVD	-	0	Reserved
30	CLR_DUMMY_FLAG	W	0x0	Indicate the clearing of dummy data
29:22	DUMMY_DATA	R/W	0x0	Store dummy data, when UART is DMA flow controller, dummy data is padded for last RXDMA burst request.
21	RXDMA_OWNER	R/W	0x0	**0**: GDMA **1**: UART, interrupt enable signal other than erpd should be masked
20	IRDA_RX_INV	R/W	0x0	irda_rx_i reverse input enable
19	IRDA_TX_INV	R/W	0x0	irda_tx_o reverse output enable
18:11	RXDMA_BURSTSIZE	R/W	0x4	RXDMA burstsize, maximum value is 16.
10:3	TXDMA_BURSTSIZE	R/W	0x4	TXDMA burstsize, maximum value is 16.
2	RXDMA_EN	R/W	0x0	RXDMA enable, valid when dma_mode in FCR is 1'b1
1	TXDMA_EN	R/W	0x0	TXDMA enable, valid when dma_mode in FCR is 1'b1
0	IRDA_ENABLE	R/W	0x0	**0**: UART mode only **1**: UART is co-work with IRDA SIR mode

All information provided in this document is subject to legal disclaimers.
© REALTEK 2024. All rights reserved.

22.3.12 REG_UART_SIR_TX_PWC0

- **Name:** UART IRDA SIR TX Pulse Width Control 0 Register
- **Size:** 32 bits
- **Address offset:** 02Ch
- **Read/write access:** R/W

It is used to indicate the shift value of SIR Tx pulse.

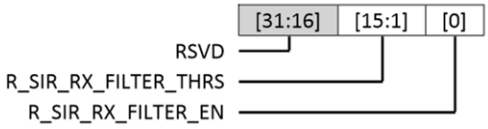


Bit	Symbol	Access	Reset	Description
31	UPPERBOUND_SHIFTRIGHT	R/W	0x0	**0**: Shift left, minus offset value of upperbound_shiftval **1**: Shift right, plus offset value of upperbound_shiftval
30:16	UPPERBOUND_SHIFTVAL	R/W	0x0	The shift value of SIR TX pulse's right edge
15	LOWBOUND_SHIFTRIGHT	R/W	0x0	**0**: Shift left, minus offset value of lowbound_shiftval **1**: Shift right, plus offset value of lowbound_shiftval
14:0	LOWBOUND_SHIFTVAL	R/W	0x0	The shift value of SIR TX pulse's left edge

22.3.13 REG_UART_SIR_RX_PFC

- **Name:** UART IRDA SIR Rx Pulse Filter Control Register
- **Size:** 32 bits
- **Address offset:** 030h
- **Read/write access:** R/W

It is used to indicate the threshold of SIR Rx pulse.



Bit	Symbol	Access	Reset	Description
31:16	RSVD	-	0	Reserved
15:1	R_SIR_RX_FILTER_THRS	R/W	0x0	The threshold of SIR Rx filter, Rx pulse is valid only when Rx pulse width is larger than the threshold.
0	R_SIR_RX_FILTER_EN	R/W	0x0	IrDA SIR Rx filter enable

22.3.14 REG_UART_BAUD_MON

- **Name:** UART Baud Monitor Register
- **Size:** 32 bits
- **Address offset:** 034h
- **Read/write access:** R/W

It is used to monitor the baud rate of Rx pulse.

All information provided in this document is subject to legal disclaimers.
© REALTEK 2024. All rights reserved.



Bit	Symbol	Access	Reset	Description
31	TOGGLE_MON_EN	R/W	0x0	Toggle monitor enable
30	MON_DATA_VLD	R		Indicate valid monitoring data
29:24	FALLING_THRESH	R/W	0x7	Indicate the threshold of Rx data rising edge
23:12	MIN_LOW_PERIOD	R		Minimum space of consecutive logic '0' of Rx data
11:0	MIN_FALL_SPACE	R		Minimum space of adjacent falling edge of Rx data

22.3.15 REG_UART_DBGR

- **Name:** UART Debug Register
- **Size:** 32 bits
- **Address offset:** 03Ch
- **Read/write access:** R

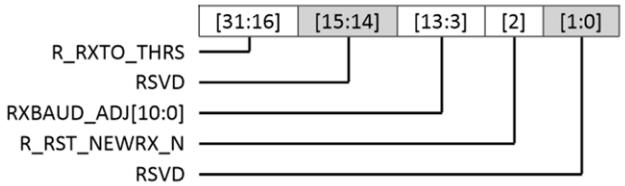


Bit	Symbol	Access	Reset	Description
31:0	DBG_UART	R		The output value of debug port, depending on dbg_sel value in SCR[2:0]

22.3.16 REG_UART_RX_PATH_CTRL

- **Name:** UART Rx Path Control Register
- **Size:** 32 bits
- **Address offset:** 040h
- **Read/write access:** R/W

It is used to monitor the Rx baud rate error and Rx baudrate factor setting.



Bit	Symbol	Access	Reset	Description
31:16	R_RXTO_THRS	R/W	0x40	Indicate the timeout threshold of Rx path, unit in one-bit time
15:14	RSVD	-	0	Reserved
13:3	RXBAUD_ADJ[10:0]	R/W	0x0	One factor of Baud rate calculation for Rx path, similar with xfactor_adj
2	R_RST_NEWRX_N	R/W	0x0	Reset new Rx path, low active
1:0	RSVD	-	0	Reserved

22.3.17 REG_UART_MON_BAUD_CTRL

- **Name:** UART Monitor Baud Rate Control Register
- **Size:** 32 bits
- **Address offset:** 044h
- **Read/write access:** R/W

All information provided in this document is subject to legal disclaimers.
© REALTEK 2024. All rights reserved.

It is used to monitor the baud rate of Rx path.

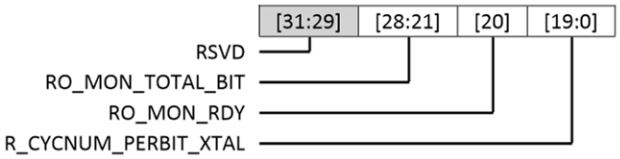


Bit	Symbol	Access	Reset	Description
31:30	RSVD	-	0	Reserved
29	R_UPD_OSC_IN_XTAL	R/W	0x0	Update parameter r_cycnum_perbit_osc (during the update of parameter r_cycnum_perbit_xtal) enable
28:9	R_CYCNUM_PERBIT_OSC	R/W/ES	0x0	Indicate the average OSC clock cycle number of one bit, for Rx path OSC clock NOTE *SW sets the initial value, and HW updates it depending on the monitor result.*
8:1	R_BIT_NUM_THRES	R/W	0x0	Indicate the bit number threshold of one monitor period, max value is 127
0	R_MON_BAUD_EN	R/W	0x0	Monitor Rx baud enable

22.3.18 REG_UART_MON_BAUD_STS

- **Name:** UART Monitor Baud Rate Status Register
- **Size:** 32 bits
- **Address offset:** 048h
- **Read/write access:** R/W

It is used to monitor the baud rate of Rx path.



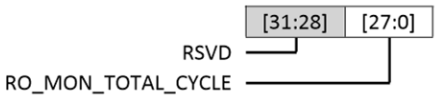
Bit	Symbol	Access	Reset	Description
31:29	RSVD	-	0	Reserved
28:21	RO_MON_TOTAL_BIT	R	0x0	Indicate the number of bits actually monitored
20	RO_MON_RDY	R	0x0	Indicate the finish of calculation of actual cycle number per bit, it's cleared when r_mon_baud_en is 0.
19:0	R_CYCNUM_PERBIT_XTAL	R/W/ES	0x0	Indicate the fractional xtal clock cycle number of one bit, for Rx path XTAL clock NOTE *SW sets the initial value, and HW updates it depending on the monitor result.*

22.3.19 REG_UART_MON_CYC_NUM

- **Name:** UART Monitored Cycle Number Register
- **Size:** 32 bits
- **Address offset:** 04Ch
- **Read/write access:** R

It is used to monitor the clock cycle number of Rx path.

All information provided in this document is subject to legal disclaimers.
© REALTEK 2024. All rights reserved.

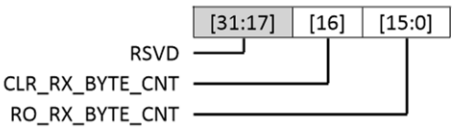


Bit	Symbol	Access	Reset	Description
31:28	RSVD	-	0	Reserved
27:0	RO_MON_TOTAL_CYCLE	R		Indicate the number of clock cycle actually monitored

22.3.20 REG_UART_RX_BYTE_CNT

- **Name:** UART Rx Data Byte Count Register
- **Size:** 32 bits
- **Address offset:** 050h
- **Read/write access:** R/W

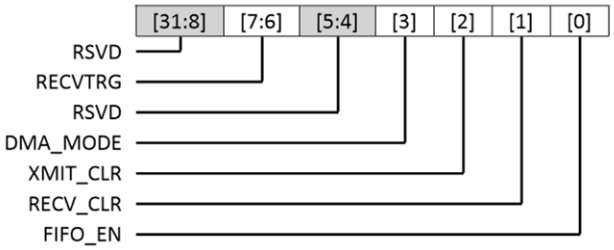
It is used to count the number of data read from RXFIFO (byte).



Bit	Symbol	Access	Reset	Description
31:17	RSVD	-	0	Reserved
16	CLR_RX_BYTE_CNT	W		Write 1 to clear ro_rx_byte_cnt
15:0	RO_RX_BYTE_CNT	R		Count the byte number of data read from RXFIFO

22.3.21 REG_UART_FCR

- **Name:** UART FIFO Control Register
- **Size:** 32 bits
- **Address offset:** 054h
- **Read/write access:** R/W

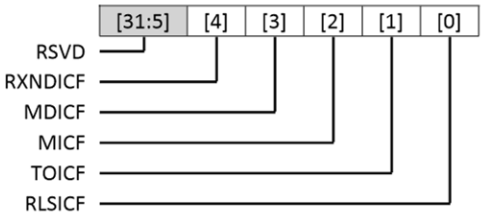


Bit	Symbol	Access	Reset	Description
31:8	RSVD	-	0	Reserved
7:6	RECVTRG	R/W	0x3	Define the 16-entries Receiver FIFO Interrupt trigger level (0~15 bytes) **00**: 1 byte **01**: 16 bytes (FIFO_DEPTH/4) **10**: 32 bytes (FIFO_DEPTH/2) **11**: 62 bytes (FIFO_DEPTH-2)
5:4	RSVD	-	0	Reserved
3	DMA_MODE	R/W	0x0	Support DMA mode (co-operate with DW DDMA in the data path)
2	XMIT_CLR	W	0x0	Clear xmitckt
1	RECV_CLR	W	0x0	Clear receiving
0	FIFO_EN	R/W	0x1	Error report of RCVR FIFO field enable, only valid when the LSR bit[7] is set to 1.

All information provided in this document is subject to legal disclaimers.
© REALTEK 2024. All rights reserved.

22.3.22 REG_UART_ICR

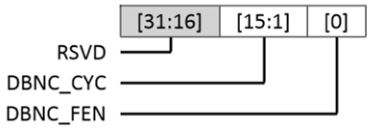
- **Name:** UART Interrupt Clear Register
- **Size:** 32 bits
- **Address offset:** 058h
- **Read/write access:** R/W



Bit	Symbol	Access	Reset	Description
31:5	RSVD	-	0	Reserved
4	RXNDICF	W	0x0	Rx noise Detected Interrupt Flag
3	MDICF	W	0x0	Monitor Done Interrupt Flag
2	MICF	W	0x0	Modem Interrupt Clear Flag
1	TOICF	W	0x0	Timeout Interrupt Clear Flag
0	RLSICF	W	0x0	Receiver Line Statue Interrupt Clear Flag Writing 1 to this bit clears PARITY_ERR, OVERRUN_ERR, FRAMING_ERR, BREAK_ERR_INT and RXFIFO_ERR in the USART_LSR register.

22.3.23 REG_UART_RXDBCR

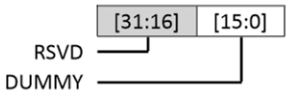
- **Name:** UART RX Debounce Control Register
- **Size:** 32 bits
- **Address offset:** 05Ch
- **Read/write access:** R/W



Bit	Symbol	Access	Reset	Description
31:16	RSVD	-	0	Reserved
15:1	DBNC_CYC	R/W	0xF	Debounce cycles in unit of sclk
0	DBNC_FEN	R/W	0x0	UART Rx debounce enable

22.3.24 REG_UART_DUMMY

- **Name:** UART Dummy Register
- **Size:** 32 bits
- **Address offset:** 060h
- **Read/write access:** R



Bit	Symbol	Access	Reset	Description
31:16	RSVD	-	0	Reserved
15:0	DUMMY	R	0x0	Reserved for HW

All information provided in this document is subject to legal disclaimers.
© REALTEK 2024. All rights reserved.

23 Infrared Radiation (IR)

23.1 Introduction

The infrared radiation (IR) is mainly designed to process IR signal with carrier frequency under 500kHz. The hardware IP supports hardware modulation which can be used on the IR Tx transmission. It also can detect the period of a continuous high/low level signal, and record in Rx FIFO, and then the software can recognize a received IR signal serial and process it.

The IR IP supports 2 types of IR Rx front end, IR diode or IR receiver module. With the IR diode front end, the IR Rx input signal still has carrier clock on it. So the software needs to do the de-modulation, which increases the CPU load. If the IR Rx front end is an IR receiver module, the carrier is filtered by the IR receiver module. So there isn't a carrier on the IR Rx input signal.

To simplify the IR signal model, a serial of IR signals is divided into several symbols. A symbol is defined as several carrier clock cycles in a certain period which is called carrier symbol, or a continuous of high/low level signal for a certain period which is called space symbol. Carrier symbol is a real signal that spreads in the air; space symbol is the signal before modulation or after de-modulation.

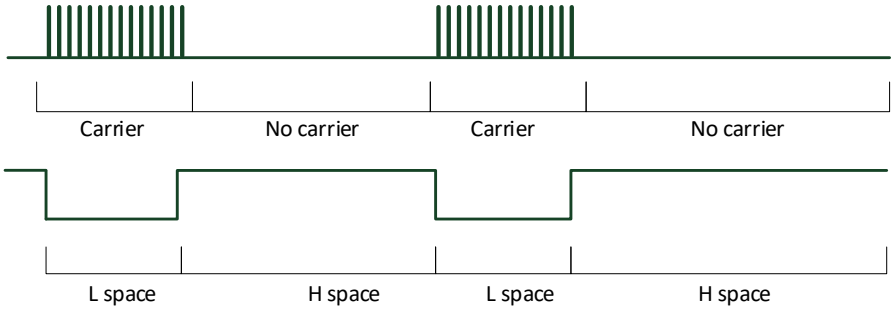


Figure 23-1 IR signal model

In Tx mode, firstly, software programs modulation parameter to generate a fixed frequency carrier, and then programs Tx FIFO to generate space symbols. Tx module modulates and transmits those IR signals. The IR Tx flow is illustrated in *Figure 23-2*.

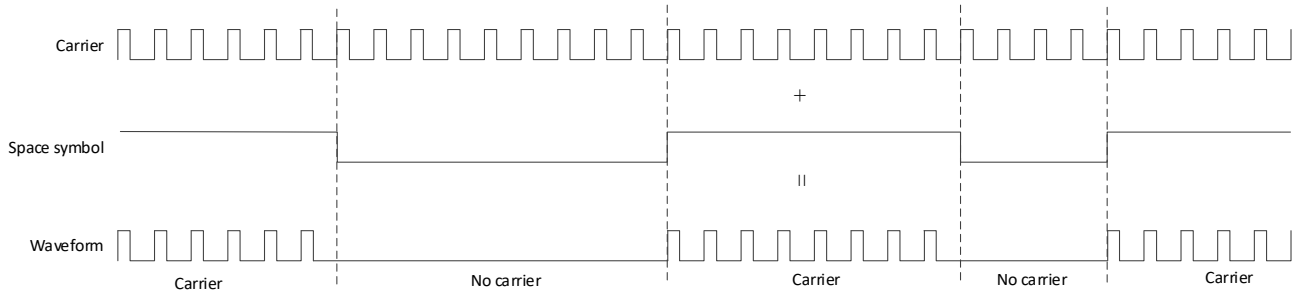


Figure 23-2 IR Tx flow

In Rx mode, if Rx front end is IR receiver module, the signals captured by IR IP core are space symbol; if Rx front end is IR diode, the signals captured by IR IP core are carrier symbols. The software process flow is divided into de-modulation and decoding. If IR diode is used, received IR signals must be de-modulated firstly. The IR Rx flow is illustrated in *Figure 23-3*.

All information provided in this document is subject to legal disclaimers.
© REALTEK 2024. All rights reserved.

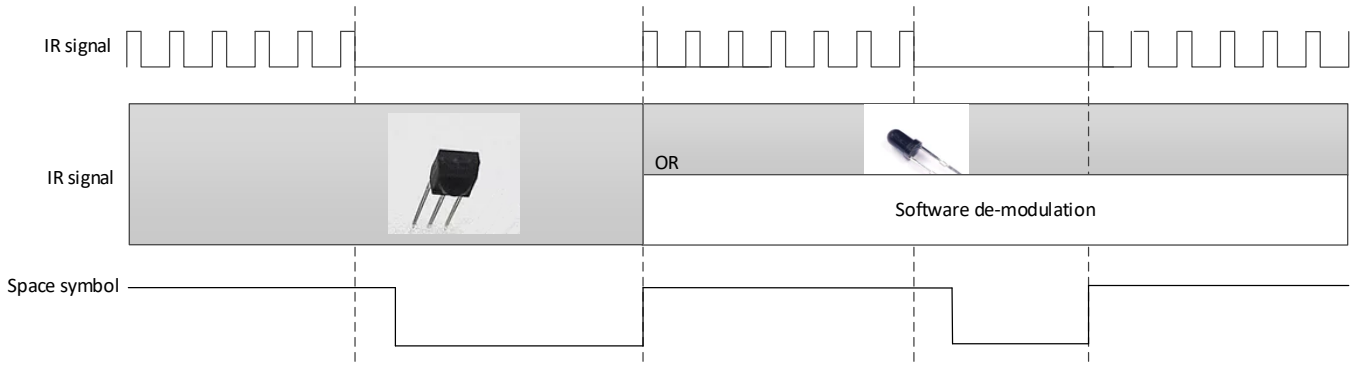


Figure 23-3 IR Rx flow with receive module and diode

23.2 Features

- Wide range of carrier frequency: from 25kHz to 500kHz
- Customizable duty by users
- Optional IR diode input
- Optional IR receiver module input
- 32*4 bytes Tx FIFO
- 32*4 bytes Rx FIFO
- Configurable Tx carrier frequency
- Configurable Tx carrier duty cycle
- Half-duplex mode

23.3 Block diagram

The block diagram of IR is shown in *Figure 23-4*.

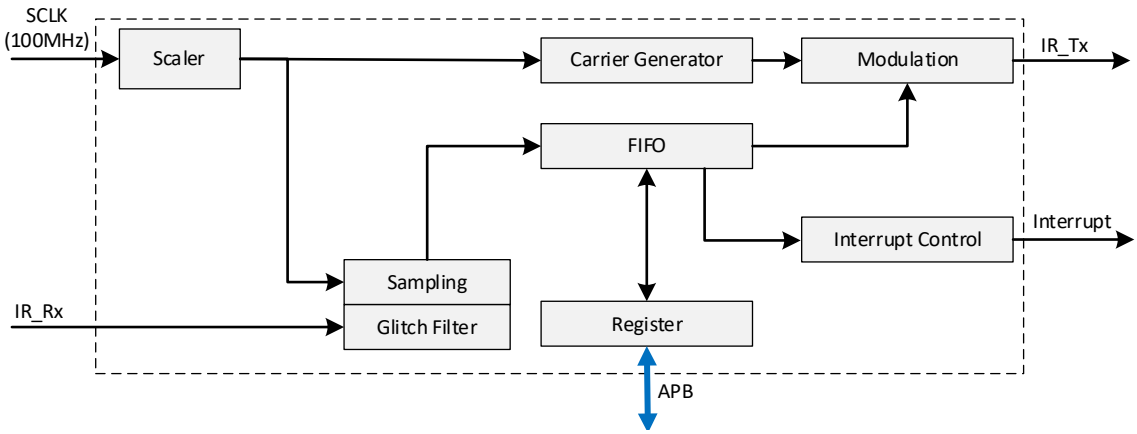


Figure 23-4 IR block diagram

- Scaler: for Tx carrier frequency or Rx sample frequency generated from SCLK (100MHz).
- Carrier Generator: Tx carrier symbol with fixed carrier duty number.
- Modulation: modulates space symbol to carrier symbol.
- Glitch Filer: IR rx filter to avoid glitches when receiving data from bus.
- FIFO: IR Tx and Rx FIFO. Tx FIFO is used to generate Tx carrier symbol count and state of Tx data; Rx FIFO is used to save the data received from bus, for software programs.
- Interrupt Control: generates interrupt for CPU handler.
- Register: contains the configuration of IR module, it's the interface with software.

All information provided in this document is subject to legal disclaimers.
© REALTEK 2024. All rights reserved.

23.4 Functional Description

23.4.1 Scaler

- Tx mode
  The division clock after scaler is carrier frequency.
  For example, SCLK=100MHz, carrier frequency = 455kHz, IR_DIV_NUM = (100M/455k) – 1.
- Rx mode
  The division clock after scaler is sampling clock. Combining Rx counter with sampling clock, software can analyze high or low level signal duration.
  For example, Rx sampling clock is 10MHz, Rx count is 0x800000FF, indicating that high level duration is 255us (where bit[31] indicates input level: 1 = high; 0 = low).

23.4.2 Data Format

This section introduces the data format of Tx FIFO and Rx FIFO.

23.4.2.1 IR Tx

To send IR data, you should write the data into IR Tx FIFO register, so it's important to understand the data format to be sent to Tx FIFO register. Before sending data, you should convert the data into the appropriate format that Tx FIFO register can recognize. The size of Tx FIFO register is 32 bits, where:

- Bit[31] indicates data level, where 1 is high level and 0 is low level.
- Bit[30] indicates the data end flag, where 1 is the last package, and 0 is the normal package.
- Bit[29:28] indicates Tx compensation mode, where 00 using Tsample and 11 using Tcomp_clk.
- Bit[27:0] indicates the number of carrier cycles.

Let $f_{carrier}$ represents the carrier frequency (the unit of $f_{carrier}$ is Hz), $T_{duration}$ represents the duration of carrier or no carrier symbol (the unit of $T_{duration}$ is us). Then,

$$\text{bit}[27:0] = f_{carrier} * T_{duration}$$

Taking NEC protocol for example, the carrier frequency is 38kHz. The format of NEC is shown in *Figure 23-5*. The NEC format consists of 2 start symbols, 64 data symbols and 1 stop symbol.

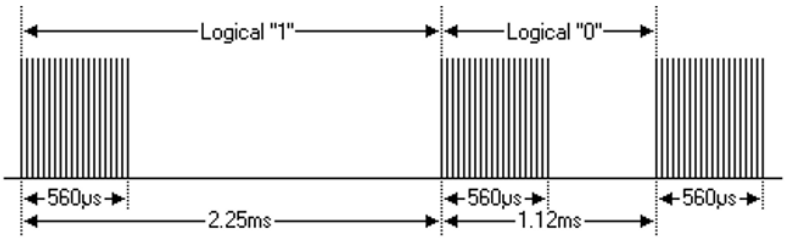


Figure 23-5 NEC modulation

- To send logical '1', you should write two data into Tx FIFO register.
  - For the first data, bit[31] = 1, bit[30] = 0, bit[27:0] = 38*560/1000 = 21.
  - For the second data, bit[31] = 0, bit[30] = 0, bit[27:0] = 38*(2250-560)/1000 = 64.
- To send logical '0', you should write two data into Tx FIFO register.
  - For the first data, bit[31] = 1, bit[30] = 0, bit[27:0] = 38*560/1000 = 21.
  - For the second data, bit[31] = 0, bit[30] = 0, bit[27:0] = 38*(1120-560)/1000 =21.

At last, you need to send the stop symbol. In this symbol, set bit[30] = 1 because the stop symbol indicates the data end of the current transmission.

23.4.2.2 IR Rx

To receive IR data, you should read the data in Rx FIFO register, so it's important to understand the data format to be received in Rx FIFO register. The size of Rx FIFO register is 32 bits, where:

- Bit[31] indicates data level, where 1 is high level and 0 is low level.
- Bit[30:0] stores the data to be received and the data format is cycle duration. Let $f_{sampling}$ represents the sampling frequency (the

All information provided in this document is subject to legal disclaimers.
© REALTEK 2024. All rights reserved.

unit of $f_{sampling}$ is kHz), T_{level} represents the duration of each high/low level (the unit of $T_{level}$is us). Then,

$$T_{level} = \frac{bit[30:0]}{f_{sampling}} * 1000$$

With the use of T_{level}, you can do further processing to get the information you need.

23.4.3 Tx Compensation Mechanism

Tx waveforms are composed of some carrier symbols and no carrier symbols. Software calculates the duration of a certain symbol by application specification (such as NEC). But no carrier symbols cannot be divisible by carrier frequency accurately. If there is no effective compensation mechanism, the error will increase. So Tx compensation is proposed and used in IR Tx to decrease the error.

In most scenarios, it is not necessary to adopt a compensation mechanism. Therefore, if you are not interested in the compensation mechanism, just skip this section. Next, we would introduce the application of compensation mechanism.

Take NEC application for example, the Tx NEC waveform shown in *Figure 23-6* is with $f_{carrier}$= 38kHz, and duty = 1/3. The system clock is 100MHz.

For 38kHz carrier frequency:

$$T_{carrier} = \frac{1}{f_{carrier}} = \frac{1}{38K} = 26.31us$$
$$T_{carrier_duty} = T_{carrier} * duty = 8.77us$$

Compensation frequency f_{comp} is a dependent clock, you can set $f_{comp}$to any value you want. In this example, we set f_{comp} =1MHz, so $T_{comp} = \frac{1}{f_{comp}} = 1us$.

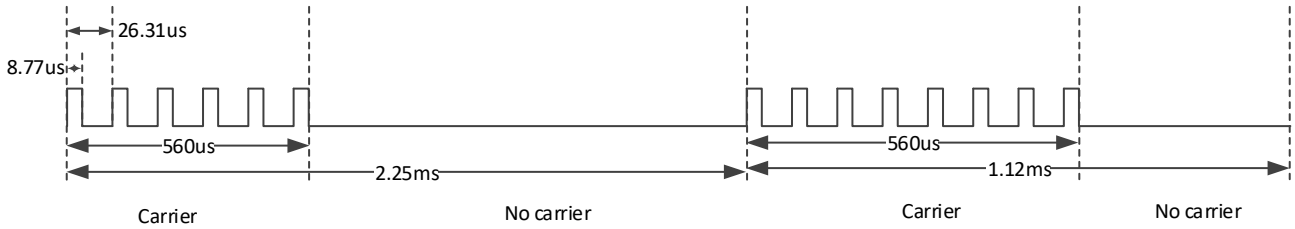


Figure 23-6 Tx NEC waveform

We divide this waveform into four symbols: 560µs carrier symbol, 1690µs (2250µs ~ 560µs) no carrier symbol, 560µs carrier symbol, and 560µs (1120µs ~ 560µs) no carrier symbol. Compare the above two calculating methods, we can find that by using a compensation mechanism, the accuracy can be improved.

Item	Normal Tx (IR_TX_COMPENSATION = 0)	Tx compensation (IR_TX_COMPENSATION = 3)
1st cycle	560µs ≈ 21*26.31 + 8.77 = 561.26µs	560µs ≈ 21*26.31 + 8.77 = 561.26µs
2nd cycle	1690µs ≈ 64*26.31+(26.31-8.77)=1701.38µs	1690µs ≈ (26.31-8.77) + 1673*1 = 1690.54µs
3rd cycle	560µs ≈ 21*26.31+ 8.77 = 561.28µs	560µs ≈ 21*26.31 + 8.77 = 561.26µs
4th cycle	560µs ≈ 21*26.31+(26.31-8.77)=570.05µs	560µs ≈ (26.31-8.77) + 543 * 1 = 560.54µs
TX FIFO data value	0x80000015, 0x00000040 0x80000015, 0x00000015	0x80000015, 0x30000689 0x80000015, 0x3000021F

NOTE

- *The compensation mechanism can only be used for no carrier symbol.*
- *Compensation method 1 (IR_TX_COMPENSATION=1) and method 2 (IR_TX_COMPENSATION=2) are not recommended. If you want to use a compensation mechanism, refer to method 3 (IR_TX_COMPENSATION=3).*
- *If IR_TX_DE_INVERSE enabled, then make sure TX compensation divider (0xC [11:0]) no less than Tx duty number (0X4[27:16]).*

23.4.4 Rx Glitch Filter

There is a glitch filter before IR input sampling, and the glitch filter can be set to 20ns, 30ns, 40ns, 50ns, 60ns, 70ns, 80ns or 90ns.

All information provided in this document is subject to legal disclaimers.
© REALTEK 2024. All rights reserved.

23.5 Interrupt

Each mode has some interrupt flags, as listed below, and these interrupt sources can be enabled, masked, or cleared individually.

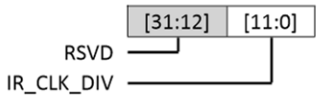
Mode	Interrupt flag	Description
Tx mode	IR_TX_FIFO_OVER_INT	When Tx FIFO is full, writing data to Tx FIFO triggers this interrupt.
	IR_TX_FIFO_LEVEL_INT	When Tx FIFO offset decreases from threshold value to threshold value − 1, this interrupt is triggered.
	IR_TX_FIFO_EMPTY_INT	When Tx FIFO offset decreases from 1 to 0, this interrupt is triggered.
Rx mode	IR_RX_FIFO_ERROR_INT	When Rx FIFO is empty, reading data from Rx FIFO triggers this interrupt.
	IR_RX_CNT_THR_INT	When given input level duration exceeds Rx counter threshold, this interrupt is triggered.
	IR_RX_CNT_OF_INT	When Rx counter overflow without IR_RX_CNT_THR_INT happens.
	IR_RX_FIFO_OF_INT	When Rx FIFO is full, Rx more data from IR input pin triggers this interrupt.
	IR_RX_FIFO_LEVEL_INT	When Rx FIFO offset increases from threshold value to threshold value + 1, this interrupt is triggered.
	IR_RX_FIFO_FULL_INT	When Rx FIFO is full, this interrupt is triggered.

23.6 Registers

Name	Address offset	Access	Description
REG_IR_CLK_DIV	000h	R/W	IR Clock Division Register
REG_IR_TX_CONFIG	004h	R/W	IR Tx Configuration Register
REG_IR_TX_SR	008h	R	IR Tx FIFO and Interrupt Status Register
REG_IR_TX_COMPE_DIV	00Ch	R/W	IR Tx Compensation Division Register
REG_IR_TX_INT_CLR	010h	R/W	IR Tx FIFO and Interrupt Clear Register
REG_IR_TX_FIFO	014h	R/W	IR Tx FIFO Register
REG_IR_RX_CONFIG	018h	R/W	IR Rx Configuration Register
REG_IR_RX_SR	01Ch	R	IR Rx FIFO and Interrupt Status Register
REG_IR_RX_INT_CLR	020h	R/W	IR Rx FIFO and Interrupt Clear Register
REG_IR_RX_CNT_INT_SEL	024h	R/W	IR Rx Count Threshold Configure Register
REG_IR_RX_FIFO	028h	R	IR Rx FIFO Register
REG_IR_VERSION	02Ch	R	IR Version Register
REG_IR_DUMMY	030h	R	IR Dummy Register

23.6.1 REG_IR_CLK_DIV

- **Name:** IR Clock Division Register
- **Size:** 32 bits
- **Address offset:** 000h
- **Read/write access:** R/W



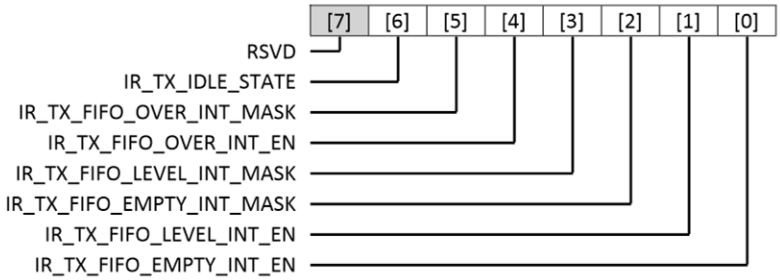
Bit	Symbol	Access	Reset	Description
31:12	RSVD	-	0	Reserved
11:0	IR_CLK_DIV	R/W	0x0	IR_CLK = IO_CLK/(1 + IR_CLK_DIV) Tx mode: divider number to generate IrDA modulation frequency. For example: sys_clk = 100MHz, modulation_freq = 455kHz, IR_DIV_NUM = (sys_clk/modulation_freq) - 1 Rx mode: waveform sample clock. IR_DIV_NUM = (sys_clk/sample clock) - 1 For example: sample clock = 100MHz, IR_DIV_NUM = 0; sample clock = 50MHz, IR_DIV_NUM = 1

All information provided in this document is subject to legal disclaimers.
© REALTEK 2024. All rights reserved.

23.6.2 REG_IR_TX_CONFIG

- **Name:** IR Tx Configuration Register
- **Size:** 32 bits
- **Address offset:** 004h
- **Read/write access:** R/W





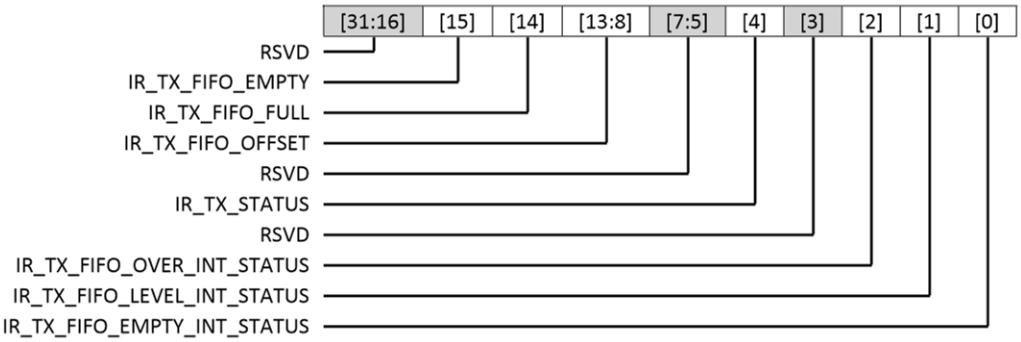
Bit	Symbol	Access	Reset	Description
31	IR_MODE_SEL	R/W	0x0	**0**: Tx mode **1**: Rx mode
30	IR_TX_START	R/W	0x0	**0**: FSM stops at idle state. **1**: FSM runs.
29:28	RSVD	-	0	Reserved
27:16	IR_TX_DUTY_NUM	R/W	0x0	Duty cycle setting for modulation frequency For example: for 1/3 duty cycle, IR_DUTY_NUM = (IR_DIV_NUM+1)/3 NOTE *Set this value equal to IR_DIV_NUM to generate 100% duty waveform.*
15	RSVD	-	0	Reserved
14	IR_TX_OUTPUT_INVERSE	R/W	0x0	**0**: Not inverse active output **1**: Inverse active output
13	IR_TX_DE_INVERSE	R/W	0x0	**0**: Not inverse FIFO define **1**: Inverse FIFO define
12:8	IR_TX_FIFO_LEVEL_TH	R/W	0x0	Tx FIFO interrupt threshold. When Tx FIFO depth = < threshold value, interrupt is triggered.
7	RSVD	-	0	Reserved
6	IR_TX_IDLE_STATE	R/W	0x0	Tx output state in idle **0**: Low **1**: High
5	IR_TX_FIFO_OVER_INT_MASK	R/W	0x0	Tx FIFO overflow interrupt **0**: Unmask **1**: Mask
4	IR_TX_FIFO_OVER_INT_EN	R/W	0x0	Tx FIFO overflow interrupt **0**: Disable **1**: Enable
3	IR_TX_FIFO_LEVEL_INT_MASK	R/W	0x0	Tx FIFO level interrupt **0**: Unmask **1**: Mask
2	IR_TX_FIFO_EMPTY_INT_MASK	R/W	0x0	Tx FIFO empty interrupt **0**: Unmask

All information provided in this document is subject to legal disclaimers.
© REALTEK 2024. All rights reserved.

				1: Mask
1	IR_TX_FIFO_LEVEL_INT_EN	R/W	0x0	Tx FIFO level interrupt When Tx FIFO offset <= threshold value, interrupt is triggered. **0**: Disable **1**: Enable
0	IR_TX_FIFO_EMPTY_INT_EN	R/W	0x0	Tx FIFO empty interrupt **0**: Disable **1**: Enable

23.6.3 REG_IR_TX_SR

- **Name:** IR Tx FIFO and Interrupt Status Register
- **Size:** 32 bits
- **Address offset:** 008h
- **Read/write access:** R



Bit	Symbol	Access	Reset	Description
31:16	RSVD	-	0	Reserved
15	IR_TX_FIFO_EMPTY	R	0x0	**0**: Not empty **1**: Empty
14	IR_TX_FIFO_FULL	R	0x0	**0**: Not full **1**: Full
13:8	IR_TX_FIFO_OFFSET	R	0x0	Tx FIFO offset is from 0 to 32. NOTE *After Tx last packet, hardware can't clear Tx FIFO offset.*
7:5	RSVD	-	0	Reserved
4	IR_TX_STATUS	R	0x0	**0**: Idle **1**: Run
3	RSVD	-	0	Reserved
2	IR_TX_FIFO_OVER_INT_STATUS	R	0x0	Tx FIFO overflow interrupt **0**: Interrupt inactive **1**: Interrupt active
1	IR_TX_FIFO_LEVEL_INT_STATUS	R	0x0	When Tx FIFO offset <= threshold value, interrupt is triggered. **0**: Interrupt inactive **1**: Interrupt active
0	IR_TX_FIFO_EMPTY_INT_STATUS	R	0x0	Tx FIFO empty interrupt **0**: Interrupt inactive **1**: Interrupt active

23.6.4 REG_IR_TX_COMPE_DIV

- **Name:** IR Tx Compensation Division Register
- **Size:** 32 bits
- **Address offset:** 00Ch
- **Read/write access:** R/W

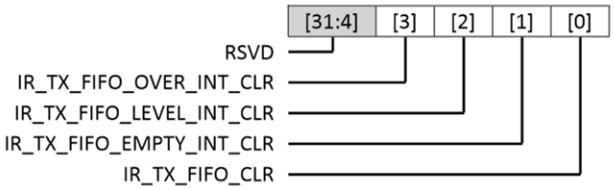
All information provided in this document is subject to legal disclaimers.
© REALTEK 2024. All rights reserved.



Bit	Symbol	Access	Reset	Description
31:12	RSVD	-	0	Reserved
11:0	TX_COMPE_DIV	R/W	0x0	IR_TX_CLK_Period = SCLK/(TX_COMPE_DIV + 1)

23.6.5 REG_IR_TX_INT_CLR

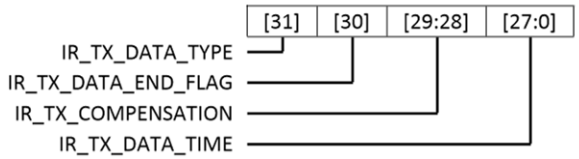
- **Name:** IR Tx FIFO and Interrupt Clear Register
- **Size:** 32 bits
- **Address offset:** 010h
- **Read/write access:** R/W



Bit	Symbol	Access	Reset	Description
31:4	RSVD	-	0	Reserved
3	IR_TX_FIFO_OVER_INT_CLR	WA0		Tx FIFO overflow interrupt Write 1 to clear
2	IR_TX_FIFO_LEVEL_INT_CLR	WA0		When Tx FIFO offset <= threshold value, interrupt is triggered. Write 1 to clear
1	IR_TX_FIFO_EMPTY_INT_CLR	WA0		Tx FIFO empty interrupt Write 1 to clear
0	IR_TX_FIFO_CLR	WA0		Write 1 to clear Tx FIFO

23.6.6 REG_IR_TX_FIFO

- **Name:** IR Tx FIFO Register
- **Size:** 32 bits
- **Address offset:** 014h
- **Read/write access:** R/W

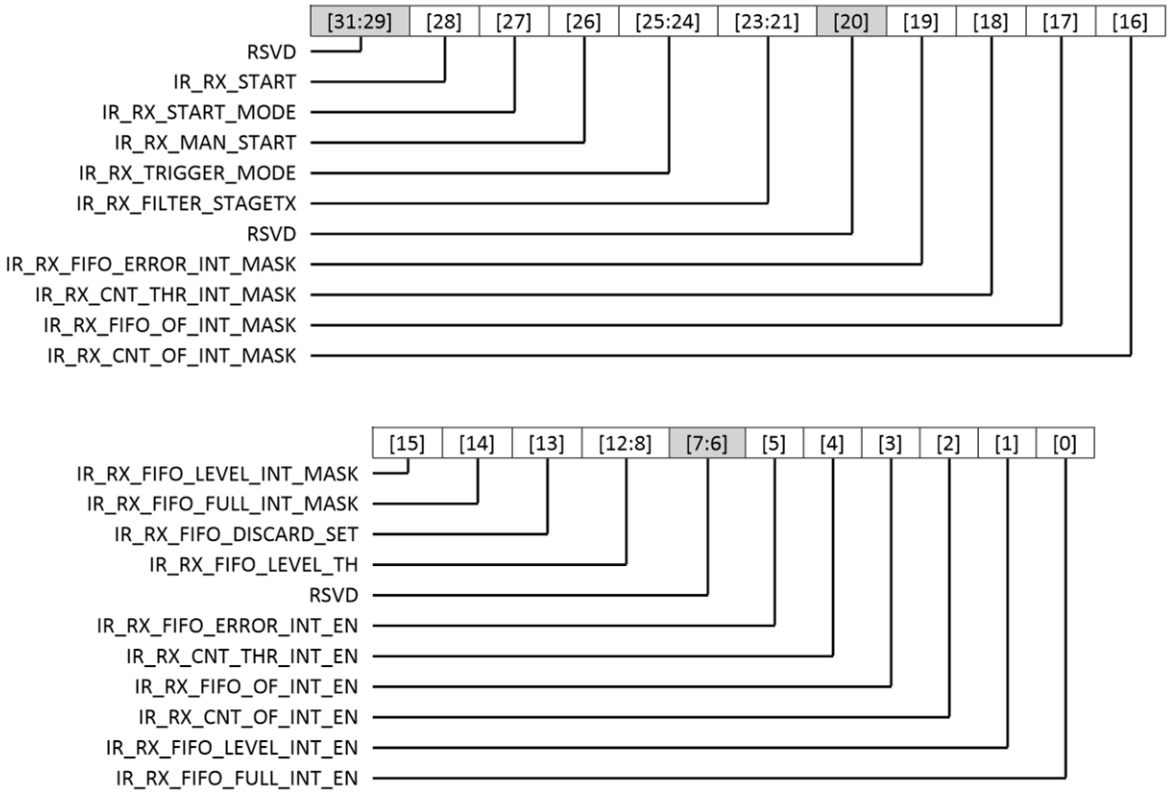


Bit	Symbol	Access	Reset	Description
31	IR_TX_DATA_TYPE	W	0	Data type **0**: Inactive carrier (no carrier) **1**: Active carrier (carrier)
30	IR_TX_DATA_END_FLAG	W	0	**0**: Normal packet **1**: Last packet
29:28	IR_TX_COMPENSATION	W	0	0x0: IR_TX_CLK_Period = Tsys_clk * IR_CLK_DIV 0x1: IR_TX_CLK_Period = (1 + 1/2) Tsys_clk * IR_CLK_DIV 0x2: IR_TX_CLK_Period = (1 + 1/4) Tsys_clk * IR_CLK_DIV 0x3: IR_TX_CLK_Period = Tsys_clk * (IR_TX_COMPE_DIV + 1)
27:0	IR_TX_DATA_TIME	W	0	Real active time = (IR_TX_DATA_TIME + 1) * IR_TX_CLK_Period

All information provided in this document is subject to legal disclaimers.
© REALTEK 2024. All rights reserved.

23.6.7 REG_IR_RX_CONFIG

- **Name:** IR Rx Configuration Register
- **Size:** 32 bits
- **Address offset:** 018h
- **Read/write access:** R/W



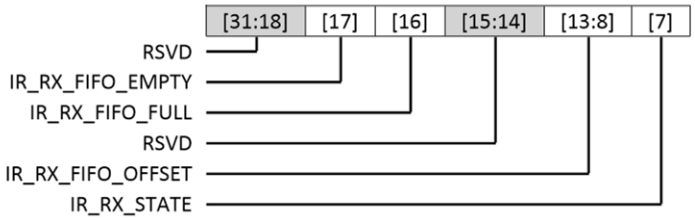
Bit	Symbol	Access	Reset	Description
31:29	RSVD	-	0	Reserved
28	IR_RX_START	R/W	0x0	**0**: FSM stops at idle state. **1**: FSM runs.
27	IR_RX_START_MODE	R/W	0x0	**0**: Manual mode, IR_RX_MAN_START control **1**: Auto-mode, Trigger-mode control
26	IR_RX_MAN_START	R/W	0x0	If IR_RX_TRIGGER_MODE =0, writing 1 means starting to check the waveform.
25:24	IR_RX_TRIGGER_MODE	R/W	0x0	**00**: High --> low trigger **01**: Low -> high trigger **10**: High -> low or low ->high trigger
23:21	IR_RX_FILTER_STAGETX	R/W	0x0	0x0: Filter <= 20ns glitch 0x1: Filter <= 30ns glitch 0x2: Filter <= 40ns glitch 0x3: Filter <= 50ns glitch …
20	RSVD	-	0	Reserved
19	IR_RX_FIFO_ERROR_INT_MASK	R/W	0x0	Rx FIFO error read interrupt When Rx FIFO is empty, read Rx FIFO and trigger interrupt. **0**: Unmask **1**: Mask
18	IR_RX_CNT_THR_INT_MASK	R/W	0x0	Rx count threshold interrupt **0**: Unmask **1**: Mask
17	IR_RX_FIFO_OF_INT_MASK	R/W	0x0	Rx FIFO overflow **0**: Unmask

All information provided in this document is subject to legal disclaimers.
© REALTEK 2024. All rights reserved.

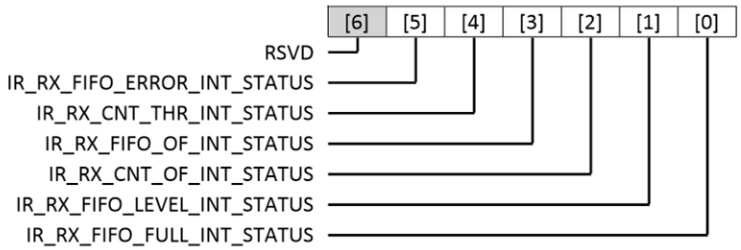
				1: Mask
16	IR_RX_CNT_OF_INT_MASK	R/W	0x0	RX counter overflow **0**: Unmask **1**: Mask
15	IR_RX_FIFO_LEVEL_INT_MASK	R/W	0x0	Rx FIFO level interrupt **0**: Unmask **1**: Mask When Rx FIFO offset >= threshold value, this interrupt is triggered.
14	IR_RX_FIFO_FULL_INT_MASK	R/W	0x0	Rx FIFO full interrupt **0**: Unmask **1**: Mask
13	IR_RX_FIFO_DISCARD_SET	R/W	0x0	When FIFO is full, new data is send to FIFO. **0**: Discard oldest data in FIFO. **1**: Reject new data sending to FIFO.
12:8	IR_RX_FIFO_LEVEL_TH	R/W	0x0	Rx FIFO interrupt threshold When Rx FIFO depth > threshold value, this interrupt is triggered.
7:6	RSVD	-	0	Reserved
5	IR_RX_FIFO_ERROR_INT_EN	R/W	0x0	Rx FIFO error read interrupt When Rx FIFO is empty, reading the Rx FIFO triggers this interrupt. **0**: Disable **1**: Enable
4	IR_RX_CNT_THR_INT_EN	R/W	0x0	Rx count threshold interrupt **0**: Disable **1**: Enable
3	IR_RX_FIFO_OF_INT_EN	R/W	0x0	RX FIFO overflow **0**: Disable **1**: Enable
2	IR_RX_CNT_OF_INT_EN	R/W	0x0	RX counter overflow **0**: Disable **1**: Enable
1	IR_RX_FIFO_LEVEL_INT_EN	R/W	0x0	Rx FIFO level interrupt When Rx FIFO offset >= threshold value, this interrupt is triggered. **0**: Disable **1**: Enable
0	IR_RX_FIFO_FULL_INT_EN	R/W	0x0	Rx FIFO full interrupt **0**: Disable **1**: Enable

23.6.8 REG_IR_RX_SR

- **Name:** IR Rx FIFO and Interrupt Status Register
- **Size:** 32 bits
- **Address offset:** 01Ch
- **Read/write access:** R



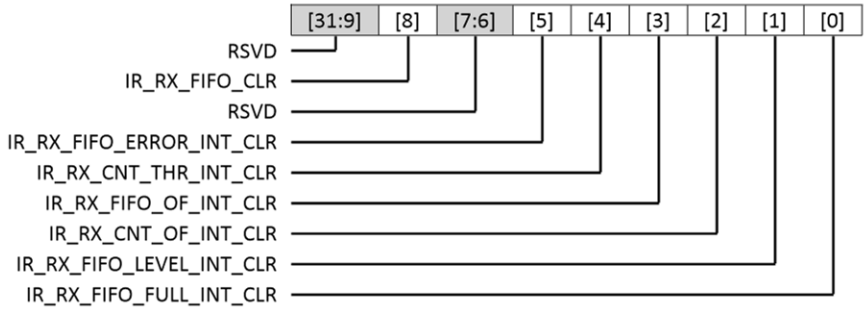
All information provided in this document is subject to legal disclaimers.
© REALTEK 2024. All rights reserved.



Bit	Symbol	Access	Reset	Description
31:18	RSVD	-	0	Reserved
17	IR_RX_FIFO_EMPTY	R	0x0	**0**: Not empty **1**: Empty
16	IR_RX_FIFO_FULL	R	0x0	**0**: Not full **1**: Full
15:14	RSVD	-	0	Reserved
13:8	IR_RX_FIFO_OFFSET	R	0x0	Rx FIFO offset
7	IR_RX_STATE	R	0x0	**0**: Idle **1**: Run
6	RSVD	-	0	Reserved
5	IR_RX_FIFO_ERROR_INT_STATUS	R	0x0	Rx FIFO error read interrupt status When Rx FIFO is empty, reading the Rx FIFO triggers this interrupt. **0**: Interrupt is inactive **1**: Interrupt is active
4	IR_RX_CNT_THR_INT_STATUS	R	0x0	Rx count threshold interrupt status **0**: Interrupt is inactive **1**: Interrupt is active
3	IR_RX_FIFO_OF_INT_STATUS	R	0x0	Rx FIFO overflow interrupt status **0**: Interrupt is inactive **1**: Interrupt is active
2	IR_RX_CNT_OF_INT_STATUS	R	0x0	Rx counter overflow interrupt status **0**: Interrupt is inactive **1**: Interrupt is active
1	IR_RX_FIFO_LEVEL_INT_STATUS	R	0x0	Rx FIFO level interrupt status **0**: Interrupt is inactive **1**: Interrupt is active
0	IR_RX_FIFO_FULL_INT_STATUS	R	0x0	Rx FIFO full interrupt status **0**: Interrupt is inactive **1**: Interrupt is active

23.6.9 REG_IR_RX_INT_CLR

- **Name:** IR Rx FIFO and Interrupt Clear Register
- **Size:** 32 bits
- **Address offset:** 020h
- **Read/write access:** R/W

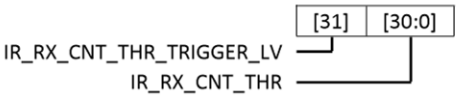


All information provided in this document is subject to legal disclaimers.
© REALTEK 2024. All rights reserved.

Bit	Symbol	Access	Reset	Description
31:9	RSVD	-	0	Reserved
8	IR_RX_FIFO_CLR	WA0		Write 1 to clear Rx FIFO
7:6	RSVD	-	0	Reserved
5	IR_RX_FIFO_ERROR_INT_CLR	WA0		Rx FIFO error read interrupt Write 1 to clear
4	IR_RX_CNT_THR_INT_CLR	WA0		Rx count threshold interrupt Write 1 to clear
3	IR_RX_FIFO_OF_INT_CLR	WA0		Rx FIFO overflow interrupt Write 1 to clear
2	IR_RX_CNT_OF_INT_CLR	WA0		Rx counter overflow interrupt Write 1 to clear
1	IR_RX_FIFO_LEVEL_INT_CLR	WA0		Rx FIFO level interrupt Write 1 to clear
0	IR_RX_FIFO_FULL_INT_CLR	WA0		Rx FIFO full interrupt Write 1 to clear

23.6.10 REG_IR_RX_CNT_INT_SEL

- **Name:** IR Rx Count Threshold Configure Register
- **Size:** 32 bits
- **Address offset:** 024h
- **Read/write access:** R/W



Bit	Symbol	Access	Reset	Description
31	IR_RX_CNT_THR_TRIGGER_LV	R/W	0x0	Trigger level **0**: When low level counter >= threshold, trigger interrupt **1**: When high level counter >= threshold, trigger interrupt
30:0	IR_RX_CNT_THR	R/W	0x0	31-bits threshold

23.6.11 REG_IR_RX_FIFO

- **Name:** IR Rx FIFO Register
- **Size:** 32 bits
- **Address offset:** 028h
- **Read/write access:** R



Bit	Symbol	Access	Reset	Description
31	IR_RX_LEVEL	RO	0x0	Rx Level **1**: High level **0**: Low level
30:0	IR_RX_CNT	RO	0x0	31 bits cycle duration

23.6.12 REG_IR_VERSION

- **Name:** IR Version Register
- **Size:** 32 bits
- **Address offset:** 02Ch
- **Read/write access:** R

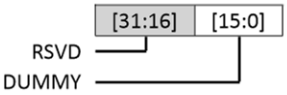
All information provided in this document is subject to legal disclaimers.
© REALTEK 2024. All rights reserved.



Bit	Symbol	Access	Reset	Description
31:0	IR_VERSION	R	0x1410150A	IR IP version

23.6.13 REG_IR_DUMMY

- **Name:** IR Dummy Register
- **Size:** 32 bits
- **Address offset:** 030h
- **Read/write access:** R



Bit	Symbol	Access	Reset	Description
31:16	RSVD	-	0	Reserved
15:0	DUMMY	R	0x0	Reserved for HW

All information provided in this document is subject to legal disclaimers.
© REALTEK 2024. All rights reserved.

24 Light Emitting Diode Controller (LEDC)

24.1 Introduction

Light Emitting Diode (LED) is a common light-emitting device. It can efficiently convert electrical energy into light energy, which has a wide range of applications in modern society.

The LEDC is used to control external Intelligent LED lamps, like WS2812B. LEDC data supports DMA mode and CPU mode, with 24-bit*32 LEDC FIFO. LEDC supports configurable refresh time.

24.2 Features

- Configurable transfer data pulse time and non-data period
- Configurable IDLE state output level
- Configurable RGB888 display mode and LED refresh time
- Up to 8095 data sequence configuration
- 32*24-bit transmit data FIFO
- DMA interface with LEDC as DMA flow controller
- Operation mode
  - DMA mode
  - Interrupt mode

24.3 Functional Description

24.3.1 Block Diagram

The block diagram of LEDC is shown in *Figure 24-1*.

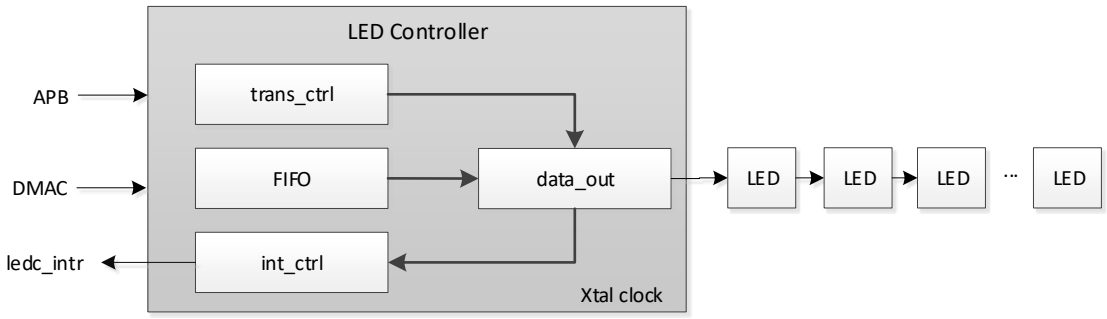


Figure 24-1 LEDC block diagram

- trans_ctrl: configures LED timing, data order, data transfer mode, data length, and LED number
- FIFO: 32*24 bits FIFO for transmitting LED data
- Int_ctrl: indicates FIFO status of LED controller, including wait data timeout or data overflow exception
- data_out: converts input data to a logical character '0' and character '1' according to trans_ctrl configuration

24.3.2 LEDC Data Transfer Format

24.3.2.1 LEDC Output Timing

The sequence of logic "1" code, logic "0" code and refresh period differs in intelligent LEDs. The basic sequence decode chart is shown in *Figure 24-2*.

All information provided in this document is subject to legal disclaimers.
© REALTEK 2024. All rights reserved.

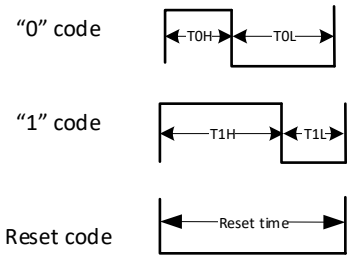


Figure 24-2 LED decode data sequence chart

In order to generate configurable LED sequence signals, LEDC counts several XTAL40MHz clock periods according to TxH_CNT and TxL_CNT where x represents 0 and 1 in the LEDC_T01_Timing_CTRL_REG register. LEDC output signals are shown in *Figure 24-3*.

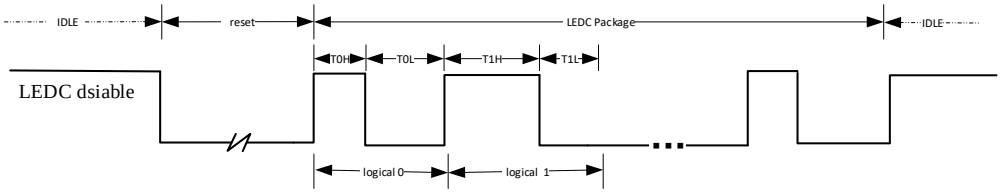


Figure 24-3 LEDC output data format when idle state is High

24.3.2.2 LEDC Transfer Timing

Figure 24-4 shows the LEDC frame output. Before sending a frame of data, a RESET is sent to refresh the LED lamp. 'wait_time1' is used to slow down the refresh rate, and can be disabled by setting the WAIT_TIME1_EN bit to 0 in the LEDC_WAIT_TIME1_CTRL_REG register. Only the last frame of a scan may be less than LED numbers.

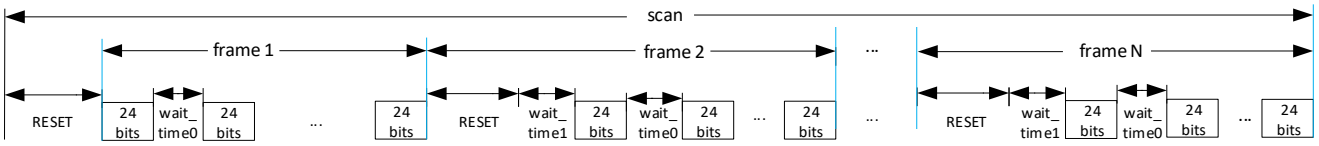
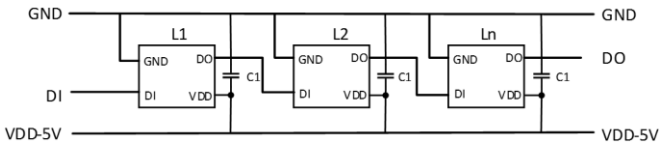


Figure 24-4 LEDC output timing diagram of a scan

- RESET: a period of time keeps LOW to refresh the LED lamp.
- Package: 24-bit logical '0' or '1' data decoded to waveforms composed of a package or a pixel. LED lamps bypass low level and reshape high level by the internal signal reshaping amplification circuit.
- wait_time0: a period of Low-Level time between two pixels' data. It is optional.
- wait_time1: a period of Low-Level time after RESET except for the 1th frame. It is optional.
- Frame: total LED lamp number except for the last frame. A frame starts after LEDC sends a RESET to the LED lamp.
- Scan: including total data packages. A scan may include more than one frame or small than a frame. A scan starts when software configures LEDC and sets the LEDC_EN bit to 1 in the LEDC_CTRL_REG register. The scan stops when LEDC is disabled.

24.3.2.3 LED Cascade Method

The data mode of LEDC output is based on intelligence LED lamps. The typical application circuit of LEDC lamps is shown in *Figure 24-5*



Note: C1, filter capacitor of LED, usually 100nF.

Figure 24-5 LED typical application circuit

The supported RGB LED lamp uses auto-reshaping transmission technology. LEDC sends data in sequence as *Figure 24-6*, where N stands for the number of LEDs driven by LEDC, and Dn stands for the data the Nth LED received.

All information provided in this document is subject to legal disclaimers.
© REALTEK 2024. All rights reserved.

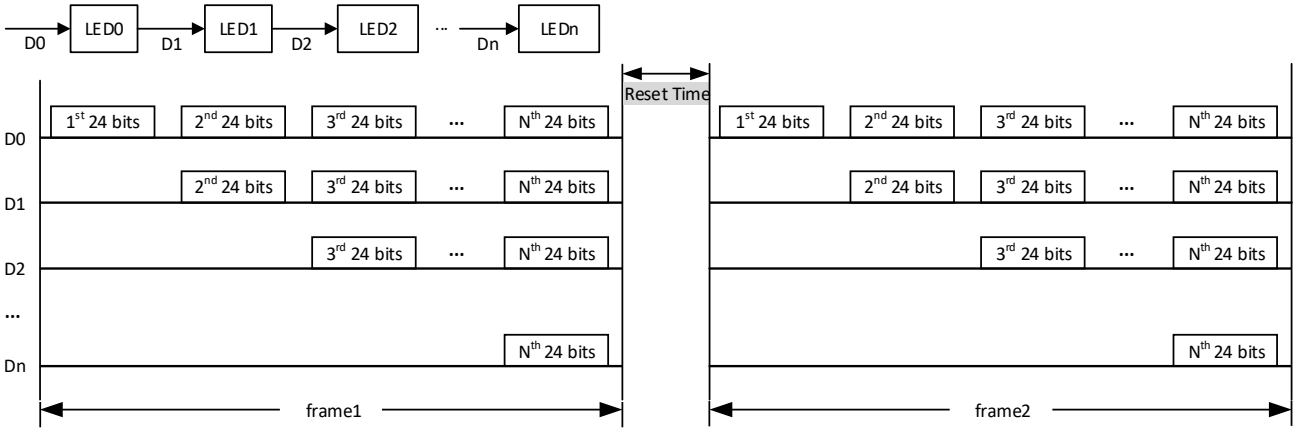


Figure 24-6 LEDC data transfer sequence

The intelligence LED lamp will catch the first 24-bit data package from Din, reshape the followed data and output to Dout. To refresh a LED lamp's buffered data, a RESET should be sent to this LED lamp before DI inputs new data packages.

24.3.2.4 RGB LED Data Structure

For LED lamps, the control circuit and RGB chip are integrated into a package, to form a complete addressable pixel. Each pixel of the three primary colors can achieve 256 brightness display, complete 16777216 color full-color display.

Each data input is RGB888. The LEDC is compatible with different data modes in both input and output ports. By default, the 24-bit data is sent in GRB mode, as shown in *Figure 24-7*. The higher bit is transmitted first.



Figure 24-7 LEDC data MSB order of GRB

Once knowing the LEDC input and output data mode, software sets the LED_RGB_MODE field (bit[8:6]) in the LEDC_CTRL_REG register according to *Table 24-1*.

Table 24-1 RGB mode configuration

RGB out mode	Input GRB	Input GBR	Input RGB	Input RBG	Input BGR	Input BRG
000	GRB	GBR	RGB	RBG	BGR	BRG
001	GBR	GRB	RBG	RGB	BRG	BGR
010	RGB	BGR	GRB	BRG	GBR	RBG
011	BGR	RGB	BRG	GRB	RBG	GBR
100	RBG	BRG	GBR	BGR	GRB	RGB
101	BRG	RBG	BGR	GBR	RGB	GRB

24.3.2.5 LEDC Mode

LEDC data transfer mode supports DMA mode and CPU interrupt mode, it is selected by setting the LEDC_DMA_EN bit (bit[5]) in the LEDC_DMA_CTRL_REG register. The internal FIFO depth is 32, data width is 24-bit.

DMA Mode

- Register LEDC_DMA_CTRL_REG.bit[5] is set to 1.
- LEDC is the flow controller of DMA transfer.
- The recommended burst length is 8 or 16, and the max. burst length is 32.
- When the remaining data to be transferred is more than or equal to the burst length, and the valid space of internal FIFO is larger than the FIFO free space threshold, LEDC requires a DMA burst transaction. For each APB write data, only the lower 24 bits are valid.
- When the remaining data to be transferred is less than the burst length, and internal FIFO is not full, LEDC requires a DMA single transaction.

CPU Mode

- Register LEDC_DMA_CTRL_REG.bit[5] is set to 0.
- The burst length can be any value less than 32.

All information provided in this document is subject to legal disclaimers.
© REALTEK 2024. All rights reserved.

- When the remaining data to be transferred is more than 0 and internal FIFO free space is larger than the FIFO free space threshold, LEDC generates a FIFO_CPUREQ_INT in register LED_INTERRUPT_CTRL_REG. CPU gets this interrupt and CPU can move data from memory to LEDC FIFO through the APB bus.

24.3.3 LEDC Interrupts

There are four LEDC interrupts: WAITDATA_TIMEOUT_INT, FIFO_OVERFLOW_INT, LEDC_TRANS_FINISH_INT, and FIFO_CPUREQ_INT.
When generating WAITDATA_TIMEOUT_INT, FIFO_OVERFLOW_INT, and LEDC_TRANS_FINISH_INT, LEDC state machine goes to TRAP state. Software should set the LEDC_SOFT_RESET bit in the LEDC_CTRL_REG register to reset.

FIFO_OVERFLOW_INT — FIFO overflow interrupt

When the data written by external is over the maximum storage space of LEDC FIFO, LEDC will be in a data loss state, thus LEDC generates FIFO_OVERFLOW_INT and enters into a TRAP state. In DMA mode, this will not happen because the FIFO threshold is set as desired 1/4 or 1/2 of FIFO depth. LEDC as a flow controller will send dma_req as FIFO threshold that matches DMA source data width and destination data width. While in CPU mode, be careful that the data to be sent is no less than the FIFO left.

WAITDATA_TIMEOUT_INT – Wait data timeout interrupt

WAITDATA_TIMIEOUT_INT indicates that a data request is sent but no data are pushed to LEDC FIFO in time. Wait data stage includes wait_time0/1 and wait_data0/1, as shown in *Figure 24-8*. If the internal FIFO of LEDC cannot get data because of some abnormal situation, LEDC turns to wait data state. This may happen between two 24-bit data packages where (Twait_time0 + Twait_data0) >= min(LED_WAIT_DATA_TIME, RESET_TIME), or between two frames where Twait_data1 >= LED_WAIT_DATA_TIME. When this happens, restart LEDC and send data on time.

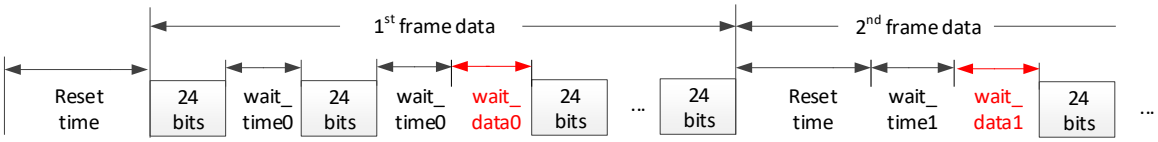


Figure 24-8 wait_data0 and wait_data1 state position in transfer

FIFO_CPUREQ_INT – FIFO request CPU data interrupt

This interrupt is only used in CPU mode. CPU gets this interrupt and can move data from memory to LEDC FIFO through the APB bus.

LEDC_TRANS_FINISH_INT – LEDC transfer finish interrupt

When LEDC has sent all data packages to LED lamps, this interrupt is generated. LEDC state machine goes to TRAP state and then switches to IDLE state.

24.4 Registers

Name	Address offset	Access	Description
REG_LEDC_CTRL_REG	000h	R/W	LEDC Control Register
REG_LED_T01_TIMING_CTRL_REG	004h	R/W	LED Timing Control Register
REG_LEDC_DATA_FINISH_CNT_REG	008h	R/W	LEDC Data Finish Register
REG_LED_RESET_TIMING_CTRL_REG	00Ch	R/W	LED Reset Timing Control Register
REG_LEDC_WAIT_TIME0_CTRL_REG	010h	R/W	LED Wait Time 0 Control Register
REG_LEDC_WAIT_TIME1_CTRL_REG	014h	R/W	LED Wait Time 1 Control Register
REG_LEDC_DATA_REG	018h	R/W	LED Data Register
REG_LEDC_DMA_CTRL_REG	01Ch	R/W	LED DMA Control Register
REG_LED_INTERRUPT_CTRL_REG	020h	R/W	LED Interrupt Control Register
REG_LED_INT_STS_REG	024h	R/W	LED FIFO Status Register
REG_LEDC_DEBUG_SEL_REG	030h	R/W	LED Debug Select Register
REG_DUMMY_F0	0F0h	R/W	Dummy Register
REG_DUMMY_F4	0F4h	R/W	Dummy Register

All information provided in this document is subject to legal disclaimers.
© REALTEK 2024. All rights reserved.

24.4.1 REG_LEDC_CTRL_REG

- **Name:** LEDC Control Register
- **Size:** 32 bits
- **Address offset:** 000h
- **Read/write access:** R/W



Bit	Symbol	Access	Reset	Description
31:29	RSVD	-	0	Reserved
28:16	TOTAL_DATA_LENGTH	R/W	0x0	Total length of transfer data, range from 1 to 8K (unit: 32 bits, only low 24-bit is valid). Actual transfer package is TOTAL_DATA_LENGTH+1. The field is recommended to an integer multiple of (LED_NUM+1). NOTE *If TOTAL_DATA_LENGTH is greater than (LED_NUM+1) but not integer multiple, the last frame of data will be transferred less than (LED_NUM+1).*
15:11	RSVD	-	0	Reserved
10	RESET_LED_EN	R/W/ES	0x0	Write operation: When software writes 1 to this bit, LEDC FSM turns to CPU_RESET_LED state, and CPU triggers LEDC to transfer a reset to LED. NOTE *Only when LEDC is in IDLE state, FSM can turn to CPU_RESET_LED state.* When LEDC finishs to send RESET to LED, LEDC FSM returns to the IDLE state, and hardware clears RESET_LED_EN. Or when software sets SOFT_RESET, LEDC FSM returns to IDLE state, and hardware clears RESET_LED_EN. Read operation: 0: LEDC completes the transmission of LED reset operation. 1: LEDC does not complete the transmission of LED reset operation.
9	LED_POLARITY	R/W	0x0	LED DI level when free. 0: Low level 1: High level
8:6	LED_RGB_MODE	R/W	0x0	LEDC inputs 24 bits data package in order of {byte2, byte1, byte0}. 000: Output 24 bits data package in order of {byte2, byte1, byte0} 001: Output 24 bits data package in order of {byte2, byte0, byte1} 010: Output 24 bits data package in order of {byte1, byte2, byte0} 011: Output 24 bits data package in order of {byte0, byte2, byte1} 100: Output 24 bits data package in order of {byte1, byte0, byte2} 101: Output 24 bits data package in order of {byte0, byte1, byte2}
5	LED_MSB_TOP	R/W	0x1	Source RGB data format. 0: LSB 1: MSB
4	LED_MSB_BYTE2	R/W	0x1	LEDC outputs 24 bits data package in order of {byte2, byte1, byte0}. 1: Output data byte2 MSB first 0: Output data byte2 LSB first
3	LED_MSB_BYTE1	R/W	0x1	LEDC outputs 24 bits data package in order of {byte2, byte1, byte0}.

All information provided in this document is subject to legal disclaimers.
© REALTEK 2024. All rights reserved.

				1: Output data byte1 MSB first 0: Output data byte1 LSB first
2	LED_MSB_BYTE0	R/W	0x1	LEDC outputs 24 bits data package in order of {byte2, byte1, byte0}. 1: Output data byte0 MSB first 0: Output data byte0 LSB first
1	LEDC_SOFT_RESET	R/WA0	0x0	LEDC software reset. If software writes 1 to this bit, the next cycle hardware will clear LEDC_SOFT_RESET to 0, which will generate a pulse. LEDC soft reset includes all interrupt status register, the control state machine returns to the IDLE state, LEDC FIFO read and write point is cleared to 0, LEDC interrupt is cleared. The related registers are as below: LEDC_CTRL_REG (LEDC_EN is cleared to 0) LEDC_DATA_FINISH_CNT_REG (LEDC_DATA_FINISH_CNT is cleared to 0) LEDC_INT_STS_REG (all interrupt are cleared) Other registers remain unchanged.
0	LEDC_EN	R/W/ES	0x0	LEDC enable bit. 0: Disable 1: Enable This bit enable indicates that LEDC can be started when LEDC data has finished transmission, or this bit is cleared to 0 by hardware when software sets LEDC_SOFT_RESET. Software clearing LEDC_EN when LEDC FSM is not IDLE doesn't affect data transfer.

24.4.2 REG_LED_T01_TIMING_CTRL_REG

- **Name:** LED Timing Control Register
- **Size:** 32 bits
- **Address offset:** 004h
- **Read/write access:** R/W



Bit	Symbol	Access	Reset	Description
31:24	T1H_CNT	R/W	0x20	LED modulate logical "1" high level time (T1H_TIME). Unit: cycle (40MHz), T1H_TIME = 25ns * T1H_CNT, where T1H_CNT>0 T1H_TIME default value is 800ns.
23:16	T1L_CNT	R/W	0xc	LED modulate logical "1" low level time. Unit: cycle (40MHz), T1L_TIME = 25ns *T1L_CNT, where T1L_CNT>0 T1L_TIME default value is 300ns.
15:8	T0H_CNT	R/W	0xc	LED modulate logical "0" high level time. Unit: cycle (40MHz), T0H_TIME = 25ns * T0H_CNT, where T0H_CNT>0 T0H_TIME default value is 300ns.
7:0	T0L_CNT	R/W	0x20	LED modulate logical "0" low level time. Unit: cycle (40MHz), T0L_TIME = 25ns *T0L_CNT, where T0L_CNT>0 T0L_TIME default value is 800ns.

24.4.3 REG_LEDC_DATA_FINISH_CNT_REG

- **Name:** LEDC Data Finish Register
- **Size:** 32 bits
- **Address offset:** 008h
- **Read/write access:** R/W

All information provided in this document is subject to legal disclaimers.
© REALTEK 2024. All rights reserved.



Bit	Symbol	Access	Reset	Description
31	RSVD	-	0	Reserved
30:16	LED_WAIT_DATA_TIME	R/W	0x2edf	The time that internal FIFO of LEDC is waiting for data. The default value is 300us by default. To avoid to insert LED RESET between two-pixel data, hardware will send WAITDATA_TIMEOUT_INT when wait_time>= min(RESET_TIME, LED_WAIT_DATA_TIME). The adjust range is from 25ns ~ 819us, LED_WAIT_DATA_TIME = 25ns *(N+1) where N is from 0 to 0x7FFF. When the setting time is exceeded, LEDC will send WAITDATA_TIMEOUT_INT interrupt. Under this abnormal situation, software should reset LEDC.
15:13	RSVD	-	0	Reserved
12:0	LED_DATA_FINISH_CNT	R	0x0	The total LED data have been sent (range: 0 ~ 8K). When LEDC transfer finishes normally, this register is cleared by hardware when generating trans_finish_int. Software knows LED_DATA_FINISH_CNT equal to TOTAL_DATA_LENGTH. When LEDC transfer interrupt abnormally, this register will be cleared to 0 by software by setting LEDC_SOFT_RESET.

24.4.4 REG_LED_RESET_TIMING_CTRL_REG

- **Name:** LED Reset Timing Control Register
- **Size:** 32 bits
- **Address offset:** 00Ch
- **Read/write access:** R/W



Bit	Symbol	Access	Reset	Description
31:30	RSVD	-	0	Reserved
29:16	RESET_TIME	R/W	0x2edf	Reset time control of LED lamp. The default reset time is 300us. The adjust range is from 25ns to 409.6us. RESET_TIME = 25ns *(N+1) where N is from 0 to 0x3FFF.
15:10	RSVD	-	0	Reserved
9:0	LED_NUM	R/W	0x0	The number of external LED lamp. Maximum up to 1024. The default value is 0, which indicate that 1 LED lamp is connected. The range is from 0 to 1023.

24.4.5 REG_LEDC_WAIT_TIME0_CTRL_REG

- **Name:** LED Wait Time 0 Control Register
- **Size:** 32 bits
- **Address offset:** 010h
- **Read/write access:** R/W

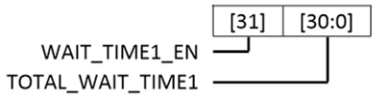
All information provided in this document is subject to legal disclaimers.
© REALTEK 2024. All rights reserved.



Bit	Symbol	Access	Reset	Description
31:10	RSVD	-	0	Reserved
9	WAIT_TIME0_EN	R/W	0	Wait time0 enable. When setting this bit, the controller automatically inserts waiting time before next LED package data. 0: Disable 1: Enable
8:0	TOTAL_WAIT_TIME0	R/W	0x1FF	Waiting time between 2 LED data, and LEDC outputs Low level during waiting time0. The adjust range is from 25ns to 12.5us, WAIT_TIME0 = 25ns*(N+1), where N is from 0 to 0x1FF.

24.4.6 REG_LEDC_WAIT_TIME1_CTRL_REG

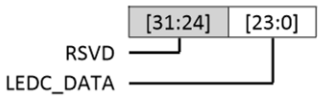
- **Name:** LED Wait Time 1 Control Register
- **Size:** 32 bits
- **Address offset:** 014h
- **Read/write access:** R/W



Bit	Symbol	Access	Reset	Description
31	WAIT_TIME1_EN	R/W	0	Wait time1 enable. When setting this bit, the controller automatically inserts waiting time before next LED frame data. 0: Disable 1: Enable
30:0	TOTAL_WAIT_TIME1	R/W	0x1FFFFFF	Waiting time between 2 frame data, and LEDC outputs Low level during waiting time1. The adjust range is from 25ns to 53s, WAIT_TIME0 = 25ns*(N+1), where N is from 0 to 0x7FFFFFFF.

24.4.7 REG_LEDC_DATA_REG

- **Name:** LED Data Register
- **Size:** 32 bits
- **Address offset:** 018h
- **Read/write access:** R/W

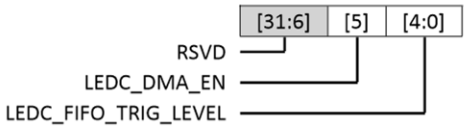


Bit	Symbol	Access	Reset	Description
31:24	RSVD	-	0	Reserved
23:0	LEDC_DATA	WPD	0	LEDC display data, the lower 24-bit is valid. Writ this register means push data to LEDC FIFO. Register LEDC_EN toggling from 0 to 1 will clear LEDC FIFO. Thus, only data pushed when LEDC_EN=1 will be send to LED.

All information provided in this document is subject to legal disclaimers.
© REALTEK 2024. All rights reserved.

24.4.8 REG_LEDC_DMA_CTRL_REG

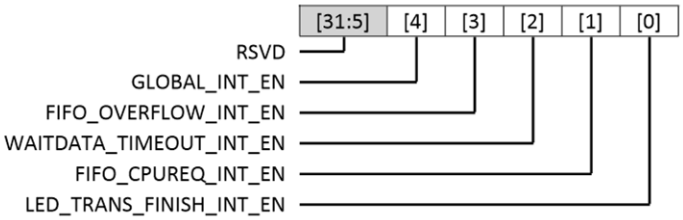
- **Name:** LED DMA Control Register
- **Size:** 32 bits
- **Address offset:** 01Ch
- **Read/write access:** R/W



Bit	Symbol	Access	Reset	Description
31:6	RSVD	-	0	Reserved
5	LEDC_DMA_EN	R/W	0x1	LEDC DMA request enable. 0: Disable request of DMA transfer data 1: Enable request of DMA transfer data
4:0	LEDC_FIFO_TRIG_LEVEL	R/W	0xF	The remaining space of internal FIFO in LEDC. When the remain space of internal LEDC FIFO is larger than LEDC_FIFO_TRIG_LEVEL, a DMA request or cpu_req_int will generate. NOTE *The default value is 15. The adjust range is from 0 to 31, and the recommended configuration is 7 or 15.*

24.4.9 REG_LED_INTERRUPT_CTRL_REG

- **Name:** LED Interrupt Control Register
- **Size:** 32 bits
- **Address offset:** 020h
- **Read/write access:** R/W



Bit	Symbol	Access	Reset	Description
31:5	RSVD	-	0	Reserved
4	GLOBAL_INT_EN	R/W	0	Global interrupt enable. 0: Disable 1: Enable
3	FIFO_OVERFLOW_INT_EN	R/W	0	FIFO overflow interrupt enable. When the data written by software is more than internal FIFO level of LEDC, LEDC is in data loss state. 0: Disable 1: Enable
2	WAITDATA_TIMEOUT_INT_EN	R/W	0	The internal FIFO in LEDC can't get data after the LED_WAIT_DATA_TIME, the interrupt will be enable. 0: Disable 1: Enable
1	FIFO_CPUREQ_INT_EN	R/W	0	FIFO request CPU data interrupt enable. 0: Disable 1: Enable
0	LED_TRANS_FINISH_INT_EN	R/W	0	Data transmission complete interrupt enable. 0: Disable 1: Enable

All information provided in this document is subject to legal disclaimers.
© REALTEK 2024. All rights reserved.

24.4.10 REG_LED_INT_STS_REG

- **Name:** LED FIFO Status Register
- **Size:** 32 bits
- **Address offset:** 024h
- **Read/write access:** R/W



Bit	Symbol	Access	Reset	Description
31:18	RSVD	-	0	Reserved
17	FIFO_EMPTY	R	0x1	FIFO empty status flag. When LEDC_EN is 0, FIFO_EMPTY equal 1. This flush residual data when LED_NUM is not integer multiples of burst length.
16	FIFO_FULL	R	0	FIFO full status flag.
15:4	RSVD	-	0	Reserved
3	FIFO_OVERFLOW_INT	RW1C	0	FIFO overflow interrupt Clear 0 when software set LEDC_SOFT_RESET. The data written by external is more than the maximum storage space of LEDC FIFO. At this time, software needs to deal with the abnormal situation. LEDC performs soft_reset operation to refresh all data. 0: FIFO does not overflow. 1: FIFO overflows.
2	WAITDATA_TIMEOUT_INT	RW1C	0	Before transfer a frame: When the internal FIFO in LEDC can't get data after the LED_WAIT_DATA_TIME, this bit is set. During transfer of a frame (between two pixel of data): to avoid insert LED RESET between two pixel data, hardware will send WAITDATA_TIMEOUT_INT when wait_time>= min(RESET_TIME,LED_WAIT_DATA_TIME). Clear 0 when software sets LEDC_SOFT_RESET. When new data arrives at WAIT_DATA state, LEDC will continue sending data. NOTE *Software should note that if the waiting time exceeds the operation of reset time, LEDC may enter refresh state, new data has not been sent.* 0: LEDC is not timeout. 1: LEDC is timeout.
1	FIFO_CPUREQ_INT	RW1C	0	When FIFO data available is less than the threshold left, the interrupt is reported to CPU. Clear 0 when software sets LEDC_SOFT_RESET. 0: FIFO doesn't request CPU for data. 1: FIFO request CPU to transfer data. NOTE *When getting FIFO_CPUREQ_INT, CPU writes 1 data to LEDC Buffer. It is inefficiently compared to DMA mode.*
0	LED_TRANS_FINISH_INT	RW1C	0	When the data configured as TOTAL_DATA_LENGTH has been transferred complete, this bit is set. Clear 0 when software sets LEDC_SOFT_RESET. 0: Data transfer has not completed. 1: Data transfer has completed.

All information provided in this document is subject to legal disclaimers.
© REALTEK 2024. All rights reserved.

24.4.11 REG_LEDC_DEBUG_SEL_REG

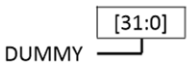
- **Name:** LED Debug Select Register
- **Size:** 32 bits
- **Address offset:** 030h
- **Read/write access:** R/W



Bit	Symbol	Access	Reset	Description
31:9	RSVD	-	0	Reserved
8:7	LEDC_DEBUG_SEL_DRIVE	R/W	0	Drive module debug out select.
6:3	LEDC_DEBUG_SEL_FIFO	R/W	0	FIFO module debug out select. [6:5]: select debug_data of ledc_txfifo, valid range is 0 or 1. [4:3]: select output data byte when bit[6:5] is 0. 2'b00: byte0 2'b01: byte1 2'b1x: byte2
2:0	LEDC_DEBUG_SEL_TOP	R/W	0	LEDC top module debug out select.

24.4.12 REG_DUMMY_F0

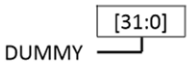
- **Name:** Dummy Register
- **Size:** 32 bits
- **Address offset:** 0F0h
- **Read/write access:** R/W



Bit	Symbol	Access	Reset	Description
31:0	DUMMY	R/W	32'hffffffff	Dummy register

24.4.13 REG_DUMMY_F4

- **Name:** Dummy Register
- **Size:** 32 bits
- **Address offset:** 0F4h
- **Read/write access:** R/W



Bit	Symbol	Access	Reset	Description
31:0	DUMMY	R/W	32'h0	Dummy register

All information provided in this document is subject to legal disclaimers.
© REALTEK 2024. All rights reserved.

25 SDIO Device

25.1 Features

The SDIO device supports the following features:

- Full compliance with SDIO card specification version 2.0:
  - 1-bit and 4-bit mode
  - Default speed mode (25MHz) and high-speed mode (50MHz)
- Partial CCCR registers are configurable
- SDIO INIC mode (SDIO to Wi-Fi transformation)
- Internal DMA supported
- Interrupt control
- 3.3V/1.8V operating voltage

25.2 Architecture

Figure 25-1 shows the hardware architecture of SDIO device interface. The SDIO ON domain implements the interface bus protocol and is controlled by the host side. The SDIO OFF domain mainly is controlled by CPU, which implement the data packets DMA transfer.

The host side can access the SDIO control registers (also called SDIO local register) via SDIO command CMD52 or CMD53. The data packets are transferred via SDIO command CMD53.

For a SDIO packet transmit, the SDIO host need to read the available receive buffer count register to make sure SDIO device has memory space to receive this packet. Then the SDIO host sends a CMD53 with this packet to SDIO device. This packet data will be written to the Data FIFO first and then the Tx DMA hardware will move these data from the Data FIFO to the given memory buffer. The DMA destination buffer is configured by the SDIO device driver.

For a SDIO packet receiving, the SDIO host will start a CMD53 transfer. Rx DMA will continuously move data from the given memory buffer to the Data FIFO. The Data State Machine will read data from the Data FIFO and shift them to the SDIO bus.

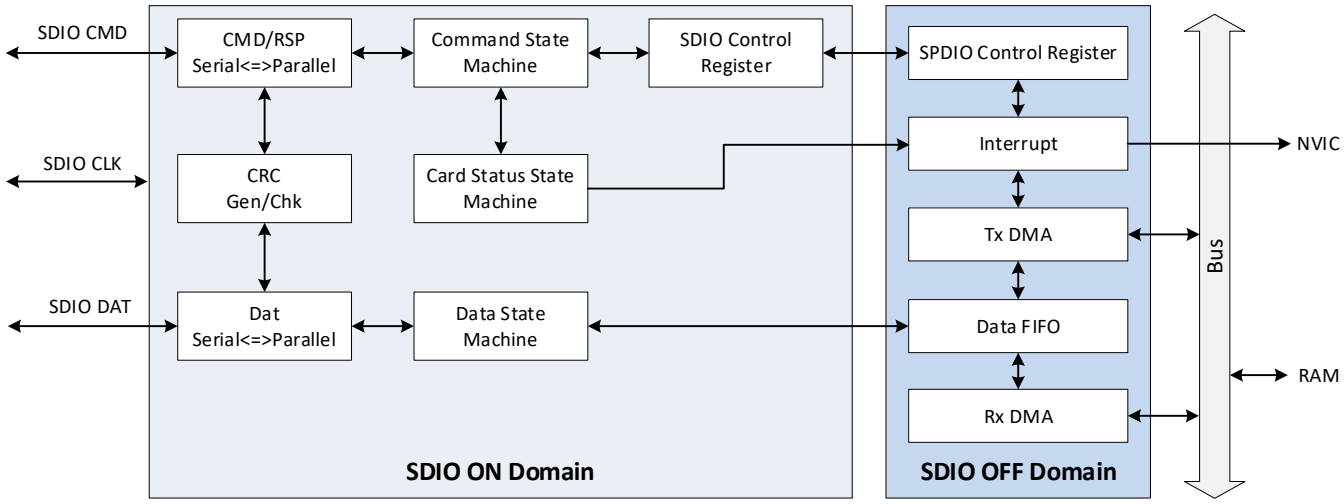


Figure 25-1 SDIO device architecture

25.3 Functional Description

25.3.1 SDIO Device Interface Control

The SDIO host side driver can control the SDIO Device hardware via SDIO control registers reading/writing. SPDIO device status also can be got by the SPDIO control registers reading. For packets transmission, the host can use CMD53 with a specific register address to read or write the SDPIO data FIFO.

All information provided in this document is subject to legal disclaimers.
© REALTEK 2024. All rights reserved.

Here is the “Register Address” coding of SDIO command CMD52 or CMD53 for control register read/write or data FIFO read/write.

Description	Device ID (Bit[16])	Domain ID (Bit[15:13])	Address/Length (Bit[12:0])
Control Register	0	000b	SDIO control register address or SPDIO control register address.
Tx FIFO	0	100b	Tx packet length. The SDIO device hardware will check this length to see if the SPDIO device has enough of memory for this packet. If not, this command will be considered as an error command. The packet length includes the Tx descriptor length, offset length and the packet payload length. The unit is 4 bytes.
Rx FIFO	0	111b	Stuff bits (not used, shall be set to 0)

NOTE

The SDIO device can response card recognition process command (such as CMD5, CMD3, CMD7) automatically. If host driver is compliant with standard protocol, it can be recognized to a high speed SDIO card. Then host can use CMD52/CMD53 to access SDIO/SGPIO control register and Tx/Rx FIFO by address as described above.

25.3.2 DMA Transfer

The SDIO device interface provides a high data rate interface for mass of data transmission. It has a dedicated Tx/Rx DMA to transfer data between data FIFO and RAM.

25.3.2.1 Tx Flow

Tx means the data transmission path is from the host side to SDIO device. The data from the host will be written into the SPDIO’s data FIFO first. Then the Tx DMA hardware moves those data from data FIFO to the given buffer.

25.3.2.1.1 Tx DMA Configuration

The memory addresses of buffers for storing Tx packets data are described by TXBDs (Tx Buffer Descriptor). A TXBD actually is 4-bytes memory which is located at RAM and containing the destination address of a Tx DMA transfer for a Tx packet data copying. The SDIO device driver is responsible for the Tx DMA configuration and the TXBDs initialization.

Figure 25-2 is used to illustrate the Tx DMA configuration. The SPDIO Tx DMA configuration includes:

- Register SPDIO_TXBD_ADDR is used to give the start address of TXBDs. The Tx DMA will fetch TXBDs with an offset from this address. This address must be an 8 bytes aligned address.
- Register SPDIO_TXBD_NUM is used to assign the number of total TXBDs.
- Register TX_BUFF_UNIT_SIZE describes the size of the buffer for Tx DMA. The unit is 64bytes. For example, TX_BUFF_UNIT_SIZE register value is 24 means that the size of the Tx DMA buffer is 24 x 64 = 1536 bytes.
- Each TXBD point to a Tx DMA buffer. The start address of a Tx DMA buffer must be an 8 bytes aligned address.
- Register SPDIO_TXBD_WPTR is the index of the TXBD which point to the Tx buffer for the next Tx packet DMA transfer. This register is maintained by the SPDIO Tx DMA hardware state machine. It will be updated as next available TXBD index whenever a Tx packet DMA transfers is done. The difference of SPDIO_TXBD_WPTR between SPDIO_TXBD_RPTR is the number of free Tx buffer. Once the number of free Tx buffer is smaller than 1, the Tx DMA hardware will stop working. Until the free Tx buffer number is bigger or equal to 1.
- Register SPDIO_TXBD_RPTR is maintained by the SPDIO Device driver. When processing a Tx buffer for a Tx DMA transfer is complete, the SPDIO device driver can either modify the corresponding TXBD to make it point to a new Tx buffer or reuse the same Tx buffer. The SPDIO Device driver need to update this register once the Tx packets buffer processing is done. The SPDIO device driver should never let the SPDIO_TXBD_RPTR value run over the SPDIO_TXBD_WPTR value, otherwise some of Tx packets will be dropped due to this error.

All information provided in this document is subject to legal disclaimers.
© REALTEK 2024. All rights reserved.

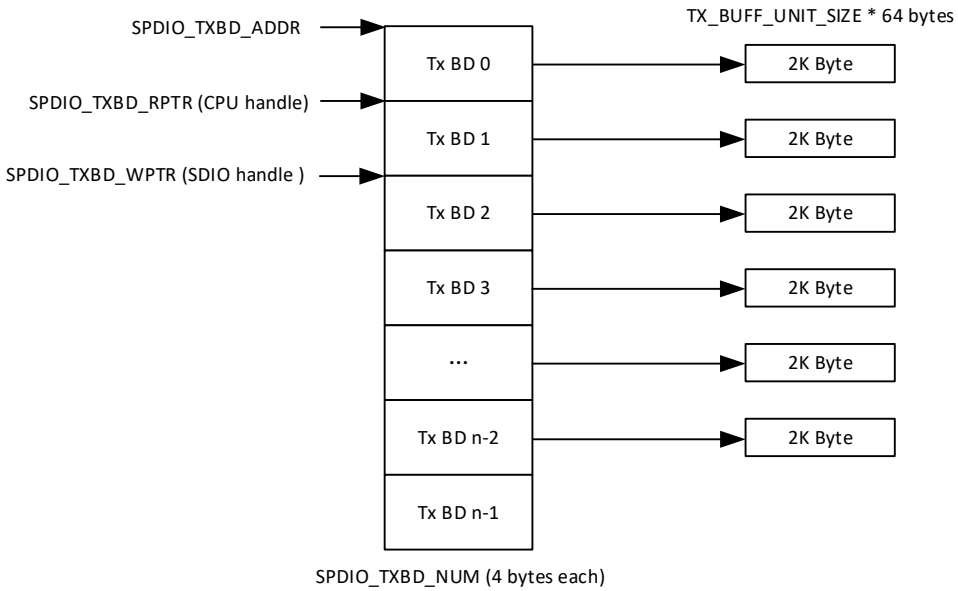


Figure 25-2 Tx DMA configuration

25.3.2.1.2 Tx DMA Behavior

On the event of SPDIO receiving packet from host:

(1) If the available TXBD Number (FREE_TXBD_NUM, which is calculated from SPDIO_TXBD_RPTR and SPDIO_TXBD_WPTR) is bigger than or equal to the packet number received from SDIO host, go to step *(2)*, else trigger TXBD_OVF interrupt.
(2) If the Packet Size is larger than the Tx Buffer Size assigned in register TX_BUFF_UNIT_SIZE, trigger TXPKT_SIZE_OVER_BUFF, else go to next step.
(3) Put the packet to the Tx Buffer which is described by the TXBD at index SPDIO_TXBD_WPTR.
(4) Update SPDIO_TXBD_WPTR and update FREE_TXBD_NUM, which should be updated as long as SPDIO_TXBD_RPTR or SPDIO_TXBD_WPTR changes.
(5) Trigger H2C_DMA_OK_INT interrupt to Ameba CPU to let the SPDIO driver to process received Tx packets.

On the event of CPU receiving H2C_DMA_OK_INT:

(1) If SPDIO_TXBD_RPTR equals to SPDIO_TXBD_WPTR (All Packets have been processed by the SPDIO driver), clear H2C_DMA_OK_INT and return; else go to next step.
(2) SPDIO driver processes the packet and informs application to move the packet.
(3) SPDIO driver updates SPDIO_TXBD_RPTR after the packet processing is finished, go to step *(1)*.

25.3.2.2 Rx Flow

Rx means the data transmission path is from Ameba to the host side. The Rx DMA state machine hardware will copy data packets from the buffer on Ameba platform to the SPDIO's data FIFO sequentially first, then the SPDIO hardware moves those data from data FIFO to either SDIO Device IP shifter or the SPI Slave IP Rx state machine shifter.

25.3.2.2.1 Rx DMA Configuration

A RXBD (Rx Buffer Descriptor) is used to describe the start memory address of an Rx packet data and its size. All of RXBDs should be located in a continuous memory block. The SPDIO device driver is responsible for the Rx DMA configuration and the RXBDs initialization. An Rx packet can be described by multiple RXBDs. The RXBD format is shown in *Figure 25-3*, and each field is described in *Table 25-1*.

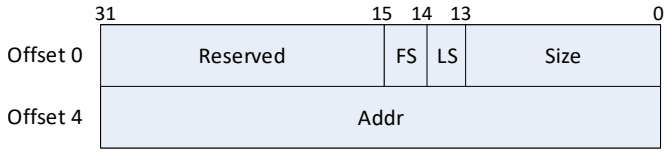


Figure 25-3 RXBD format

All information provided in this document is subject to legal disclaimers.
© REALTEK 2024. All rights reserved.

Table 25-1 RXBD field description

Offset (bit)	Field	Description
0x0[13:0]	Size	The data size of the Rx packet, in byte. This value is valid only when FS bit is 1.
0x0[14]	LS	If the value of this field is 1, it means that this RXBD is the last RXBD of an Rx packet.
0x0[15]	FS	If the value of this field is 1, it means that this RXBD is the first RXBD of an Rx packet. The size field is valid only when this bit value is 1.
0x4[31:0]	Addr	The start address of the Rx packet data, it must be a 4-byte aligned address.

Figure 25-4 shows the illustration of the Rx DMA configuration. The SPDIO Rx DMA configuration includes:

- Register SPDIO_RXBD_ADDR is used to give the start address of the memory for all RXBDs. The Rx DMA will fetch RX_BDs with an offset from this address. This address must be an 8-bytes aligned address to make the Rx DMA can work correctly.
- Register SPDIO_RXBD_NUM is used to assign the number of total RXBDs.
- Register SPDIO_RXBD_WPTR is maintained by the SPDIO Device driver. When the SPDIO driver sending an Rx packet, the SPDIO driver fill one or more of RXBD to describe this Rx packet. And then update the SPDIO_RXBD_WPTR by increase its value or wrapped as 0 to make its value is the index of the last RXBD of this sending Rx packet. The SPDIO device driver should never let the SPDIO_RXBD_WPTR value run over the SPDIO_RXBD_RPTR value. Otherwise some of Rx packets will be dropped due to this error.
- Register SPDIO_RXBD_RPTR is the index of the RXBD which describing the Rx packet for the Rx DMA is going to transfer it to the host side. It is maintained by the SPDIO Rx DMA hardware. Once an Rx DMA transfer of a Rx packet is finished, the Rx DMA hardware will update this register to make its value is the index of the last fetched RXBD.

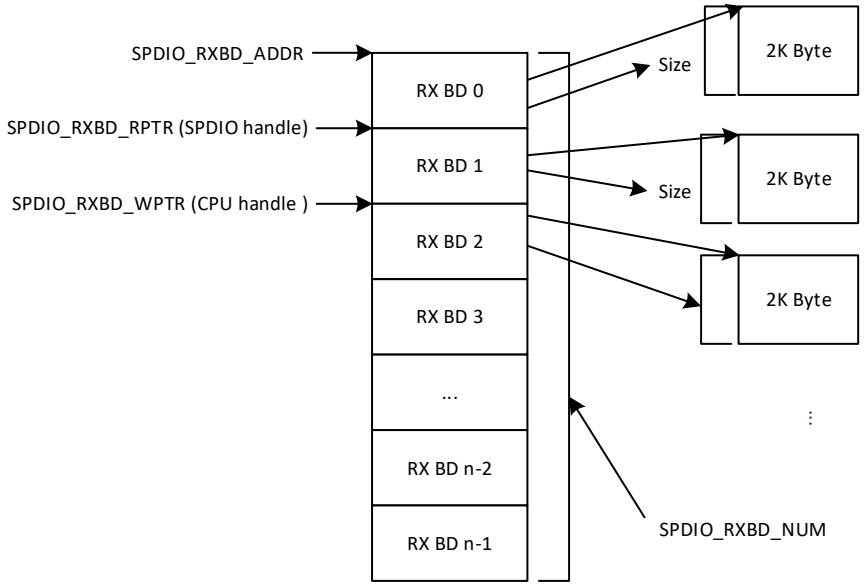


Figure 25-4 Rx DMA configuration

25.3.2.2.2 Rx DMA behavior

On the event of SPDIO Driver requested to send a packet to the host:
(1) SPDIO Driver processes this packet and update SPDIO_RXBD_WPTR.
(2) Set HCI_RX_REQ to trigger SPDIO Rx DMA start the fetching of RXBD.

On the event of SPDIO Rx DMA receiving HCI_RX_REQ:
(1) Update RX_REQ_LEN and RX_REQ_ADDR as described in the RXBD at index SPDIO_RXBD_RPTR.
(2) Trigger an interrupt to host to make it start the Rx packet transfer.
(3) Get packet data from RX_FIFO and sends them to host.
(4) SPDIO Rx DMA updates the SPDIO_RXBD_RPTR when a packet transfer is finished.
(5) If SPDIO_RXBD_RPTR < SPDIO_RXBD_WPTR, go to step *(1)*; else go to next step.
(6) Clear HCI_RX_REQ.
(7) Trigger C2H_DMA_OK_INT interrupt to inform the SPDIO driver that Rx packets has been transferred.

On the event of the CPU receiving C2H_DMA_OK_INT interrupt:
(1) Inform the SPDIO driver that Rx packets have been transferred and one or more RXBD is free.

All information provided in this document is subject to legal disclaimers.
© REALTEK 2024. All rights reserved.

25.4 Register

There are three register areas. The CPU and driver (host driver) have different access permissions.

Register	Base address for CPU	Base address for driver	Description
SDIO Control register	0x400121000	0	This area is mainly accessed by driver. The driver accesses this area for SDIO function implementation. The CPU can only read this area.
SPDIO Control Register	0x400120000	0x100	This area is mainly accessed by CPU. The CPU uses this area to interact with driver. The driver can only read this area.
CCCR Bit Definition	0x400122000	-	This area can only be accessed by CPU. The CPU accesses this area to control the feature of SDIO device.

25.4.1 SDIO Local Control Registers

25.4.1.1 SDIO_STATIS_RECOVERY_TIMEOUT

- Name: SDIO_STATIS_RECOVERY_TIMEOUT Register
- Width: 32 bit
- Initial Value: 0x0C00
- Offset: 0002h

Bit	Access	INI	Symbol	Description
31:16	-		RSVD	Reserved
15:0	R/W		SDIO_STATIS_RECOVERY_TIMEOUT	It is SDIO status timeout. The unit is 32 usec. The default value is 0xc00, meaning that timeout is 98304 usec (32*0xc00)

25.4.1.2 SDIO_HIMR

- Name: SDIO_HIMR Register
- Width: 32 bit
- Initial Value: 0xF
- Offset: 0014h

Bit	Access	INI	Symbol	Description
31:23	-		RSVD	Reserved
22	R/W		CPU_NOT_RDY_MSK	This bit is clear, and CPU_NOT_RDY is masked.
21	-		RSVD	Reserved
20	R/W		H2C_BUS_FAIL_MSK	This bit is clear, and H2C_BUS_FAIL_INT is masked.
19	R/W		CPWM2_MSK	This bit is clear, and CPWM2_INT is masked.
18	R/W		CPWM1_ MSK	This bit is clear, and CPWM1_INT is masked.
17	R/W		C2H_MSG_MSK	This bit is clear, and C2H_MSG_INT is masked.
16:5	-		RSVD	Reserved
4	R/W		TXBD_OVERFLOW_MSK	This bit is clear, and TXBD_OVERFLOW is masked.
3	R/W		TXAGG_SIZE_MISMATCH_MSK	This bit is clear, and TXAGG_SIZE_MISMATCH is masked.
2	R/W		TXPKT_SIZE_OVER_BUFF_MSK	This bit is clear, and TXPKT_SIZE_OVER_BUFF is masked.
1	R/W		TXFIFO_AVAL_MSK	This bit is clear, and TXFIFO_AVAL _INT INT is masked.
0	R/W		RX_REQUEST_MSK	This bit is clear, and RX_REQUEST_INT is masked.

25.4.1.3 SDIO_HISR

- Name: SDIO_HISR Register
- Width: 32 bit
- Initial Value: 0x0
- Offset: 0018h

All information provided in this document is subject to legal disclaimers.
© REALTEK 2024. All rights reserved.

Bit	Access	INI	Symbol	Description
31:23	-		RSVD	Reserved
22	R/W1C		CPU_NOT_RDY	When host send CMD53 for to access SDIO OFF, and CPU_RDY_IND=0, trigger this interrupt
21	-		RSVD	Reserved
20	R/W1C		H2C_BUS_FAIL	H2C BUS Fail Interrupt: To notify Host DMA transfer error by AHB BUS resource conflict. Host get this error INT, and know the previous packet fail, and need to re-send or IO_RESET device!
19	R/W1C		CPWM2_INT	CPWM2 Write Interrupt: This interrupt will be raised when FW writes CPWM2_TOGGLING of CPWM2 register. Write 1 to clear
18	R/W1C		CPWM1 _INT	CPWM1 Write Interrupt: This interrupt will be raised when FW writes CPWM1_TOGGLING of CPWM1 register. Write 1 to clear
17	R/W1C		C2H_MSG_INIT	CPU to Host Message INT Status: This interrupt will be raised when FW sends a C2H message. Write 1 to clear
16:5	-		RSVD	Reserved
4	R/W1C		TXBD_OVF	When host send Tx packet and there is no enough TXBD, trigger this interrupt.
3	R/W1C		TX_AGG_SIZE_MISMATCH	If the total size of the Tx aggregation packets is LARGER the CMD53 Data Size, this interrupt trigger.
2	R/W1C		TXPKT_SIZE_OVER_BUFF	If the Packet size (aggregated or not) is larger than the buffer size assigned in the TX_BD, this interrupt trigger
1	R		TXAVAL_INT	When the free page is lower than SDIO_AVAI_BD_NUM_TH_L and higher than SDIO_AVAI_BD_NUM_TH_H consequently, this interrupt occurs. When Host reads FREE_TXBD_NUM, the bit becomes zero. When using CMD52 to read FREE_TXBD_NUM, we suggest to read offset 0x23->0x22->0x21->0x20 consequently.
0	R		RX_REQUEST	This bit is set when one complete Rx aggregation packet is received into RXPKTBUF and RX_REQUEST_MSK is set to 1. Only when Rx FIFO is empty, the bit becomes zero.

25.4.1.4 RX0_REQ_LEN

- Name: RX0_REQ_LEN Register
- Width: 32 bit
- Initial Value: 0x0
- Offset: 001Ch

Bit	Access	INI	Symbol	Description
31	R		RX_REQ_LEN_RDY	Indicate RX0_REQ_LEN is ready
30:24	-		RSVD	Reserved
23:0	R		RX0_REQ_LEN	The requested Rx Packet length; Unit: Byte For the Rx Aggregation case, the length is total length of the packet. Each packet is 4-byte alignment. This value is legal only when RX_REQ_LEN_RDY is 1.

25.4.1.5 FREE_TXBD_NUM

- Name: FREE_TXBD_NUM Register
- Width: 16 bit
- Initial Value: 0x0
- Offset: 0020h

Bit	Access	INI	Symbol	Description
15:0	R		FREE_TXBD_NUM	The free TXBD number.

All information provided in this document is subject to legal disclaimers.
© REALTEK 2024. All rights reserved.

25.4.1.6 HCPWM

- Name: HCPWM Register
- Width: 8 bit
- Initial Value: 0x0
- Offset: 0038h

Bit	Access	INI	Symbol	Description
7	R		TOGGLING	Toggling Bit: this is the one bit sequence number field. Interrupt is issued when this bit is changed from “1” to “0” or from “0” to “1”.
6:0	-		RSVD	Reserved

25.4.1.7 HCPWM2

- Name: HCPWM2 Register
- Width: 16 bit
- Initial Value: 0x0
- Offset: 003Ah

Bit	Access	INI	Symbol	Description
15	R		TOGGLING	Toggling Bit: this is the one bit sequence number field. Interrupt is issued when this bit is changed from “1” to “0” or from “0” to “1”.
14:0	-		RSVD	Reserved

25.4.1.8 SDIO_H2C_MSG

- Name: SDIO_H2C_MSG Register
- Width: 32 bit
- Initial Value: 0x0
- Offset: 0060h

Bit	Access	INI	Symbol	Description
31	R/W		TOGGLING	Toggling Bit: this is the one bit sequence number field. Interrupt is issued when this bit is changed from “1” to “0” or from “0” to “1”.
30:0	R/W		SDIO_H2C_MSG	Driver to FW Message

25.4.1.9 SDIO_C2H_MSG

- Name: SDIO_C2H_MSG Register
- Width: 32 bit
- Initial Value: 0x0
- Offset: 0064h

Bit	Access	INI	Symbol	Description
31	R		TOGGLING	Toggling Bit: this is the one bit sequence number field. Interrupt is issued when this bit is changed from “1” to “0” or from “0” to “1”.
30:0	R		SDIO_C2H_MSG	FW to Driver Message

25.4.1.10 SDIO_H2C_MSG_EXT

- Name: SDIO_H2C_MSG_EXT Register
- Width: 32 bit
- Initial Value: 0x0
- Offset: 0068h

Bit	Access	INI	Symbol	Description
31:0	R/W		SDIO_H2C_MSG_EXT	Driver to FW EXT Message, trigger by 0x0060[31]

All information provided in this document is subject to legal disclaimers.
© REALTEK 2024. All rights reserved.

25.4.1.11 SDIO_C2H_MSG_EXT

- Name: SDIO_C2H_MSG_EXT Register
- Width: 32 bit
- Initial Value: 0x0
- Offset: 006Ch

Bit	Access	INI	Symbol	Description
31:0	R		SDIO_C2H_MSG_EXT	FW to Driver EXT Message

25.4.1.12 HRPWM

- Name: HRPWM Register
- Width: 8 bit
- Initial Value: 0x7
- Offset: 0080h

Bit	Access	INI	Symbol	Description
7	R/W		TOGGLING	Toggling Bit: this is the one bit sequence number field. Interrupt is issued when this bit is changed from “1” to “0” or from “0” to “1”.
6:0	-		RSVD	Reserved

25.4.1.13 HRPWM2

- Name: HRPWM2 Register
- Width: 16 bit
- Initial Value: 0x0
- Offset: 0082h

Bit	Access	INI	Symbol	Description
15	R/W		TOGGLING	Toggling Bit: this is the one bit sequence number field. Interrupt is issued when this bit is changed from “1” to “0” or from “0” to “1”.
14:0	R/W		RSVD	RSVD

25.4.1.14 CPU_INDICATION

- Name: CPU_INDICATION Register
- Width: 8 bit
- Initial Value:
- Offset: 0087h

Bit	Access	INI	Symbol	Description
7:4	R/W		SDIO_TEST_BUSY_CNT	For Device test BUSY control count, BUSY time ~= BUSY_CNT * 16-SD_CLK
3:1	-		RSVD	Reserved
0	R		CPU_RDY_IND	Used to indicate that CPU is ready for TRX. Default: 0 This bit is synchronized from SYTEM_CPU_RDY_IND(Offset 0x1C5)

25.4.1.15 ERR_FLAG

- Name: ERR_FLAG Register
- Width: 8 bit
- Initial Value: 0x0
- Offset: 00C0h

Bit	Access	INI	Symbol	Description
7:0	R/W1C		ERR_FLAG	Error flag for SDIO debug

All information provided in this document is subject to legal disclaimers.
© REALTEK 2024. All rights reserved.

25.4.1.16 SDIO_CMD_ERRCNT

- Name: SDIO_CMD_ERRCNT Register
- Width: 8 bit
- Initial Value: 0x0
- Offset: 00C2h

Bit	Access	INI	Symbol	Description
7:0	R/W		CMD_CRC_ERR_CNT	The number of CRC error of SDIO command

25.4.1.17 SDIO_DATA_ERRCNT

- Name: SDIO_DATA_ERRCNT Register
- Width: 8 bit
- Initial Value: 0x0
- Offset: 00C3h

Bit	Access	INI	Symbol	Description
7:0	R/W		DATA_CRC_ERR_CNT	The number of CRC error of SDIO data block

25.4.1.18 SDIO_CRC_ERR_INDEX

- Name: SDIO_CRC_ERR_INDEX Register
- Width: 8 bit
- Initial Value: 0x0
- Offset: 00C4h

Bit	Access	INI	Symbol	Description
7:5	-		RSVD	Reserved
4	R/W1C		D3_CRC_ERR	Indicate there is CRC error on SD_D3 It is suggested to clear it by CMD52
3	R/W1C		D2_CRC_ERR	Indicate there is CRC error on SD_D2 It is suggested to clear it by CMD52
2	R/W1C		D1_CRC_ERR	Indicate there is CRC error on SD_D1 It is suggested to clear it by CMD52
1	R/W1C		D0_CRC_ERR	Indicate there is CRC error on SD_D0 It is suggested to clear it by CMD52
0	-		RSVD	Reserved

25.4.1.19 SDIO_AVAI_BD_NUM_TH_L

- Name: SDIO_AVAI_BD_NUM_TH_L Register
- Width: 32 bit
- Initial Value: 0x0
- Offset: 00D0h

Bit	Access	INI	Symbol	Description
31:16	-		RSVD	Reserved
15:0	R/W		SDIO_AVAI_BD_NUM_TH_L	Low threshold for TXBD number

25.4.1.20 SDIO_AVAI_BD_NUM_TH_H

- Name: SDIO_AVAI_BD_NUM_TH_H Register
- Width: 32 bit
- Initial Value: 0x0
- Offset: 00D4h

All information provided in this document is subject to legal disclaimers.
© REALTEK 2024. All rights reserved.

Bit	Access	INI	Symbol	Description
31:16	-		RSVD	Reserved
15:0	R/W		SDIO_AVAI_BD_NUM_TH_H	High threshold for TXBD number

25.4.1.21 SDIO_RX_AGG_CFG

- Name: SDIO_RX_AGG_CFG Register
- Width: 16 bit
- Initial Value: 0x0
- Offset: 00D8h

Bit	Access	INI	Symbol	Description
15	R/W		SDIO_RX_AGG_EN	Enable SDIO Bus Rx aggregation
14:8	R/W		RX_AGG_TO	Timeout threshold for Rx aggregation; Unit: 32us
7:0	R/W		RX_AGG_BD_COUNT_TH	RX_BD count threshold for Rx aggregation

25.4.1.22 SDIO_CMD_TIMING_CTRL

- Name: SDIO_CMD_TIMING_CTRL Register
- Width: 8 bit
- Initial Value: 0x0
- Offset: 00DCh

Bit	Access	INI	Symbol	Description
7:4	R/W		SD_CMD_DLY_SEL	Used to control the delay of SD_CMD
3:0	-		RSVD	Reserved

25.4.1.23 SD_DATA01_TIMING_CTRL

- Name: SD_DATA01_TIMING_CTRL Register
- Width: 8 bit
- Initial Value: 0x0
- Offset: 00DDh

Bit	Access	INI	Symbol	Description
7:4	R/W		SD_DATA1_DLY_SEL	Used to control the delay of SD_DATA1
3:0	R/W		SD_DATA0_DLY_SEL	Used to control the delay of SD_DATA0

25.4.1.24 SD_DATA23_TIMING_CTRL

- Name: SD_DATA23_TIMING_CTRL Register
- Width: 8 bit
- Initial Value: 0x0
- Offset: 00DEh

Bit	Access	INI	Symbol	Description
7:4	R/W		SD_DATA3_DLY_SEL	Used to control the delay of SD_DATA3
3:0	R/W		SD_DATA2_DLY_SEL	Used to control the delay of SD_DATA2

25.4.2 SDIO Control Registers

25.4.2.1 SPDIO_TXBD_ADDR

- Name: SPDIO_TXBD_ADDR Register
- Width: 32 bit
- Initial Value: 0x0
- Offset: 01A0h

All information provided in this document is subject to legal disclaimers.
© REALTEK 2024. All rights reserved.

Bit	Access	INI	Symbol	Description
31:0	CPU: R/W DRV: R		SPDIO_TXFIFO_ADDR	The base address of TXBD The value of this register should be configure by CPU It should be 4-byte alignment. (Because TXBD size is 8 bytes.)

25.4.2.2 SPDIO_TXBD_NUM

- Name: SPDIO_TXBD_NUM Register
- Width: 16 bit
- Initial Value: 0x0
- Offset: 01A4h

Bit	Access	INI	Symbol	Description
15:0	CPU: R/W DRV: R		SPDIO_TXBD_NUM	The Number of TXBD. Unit: Number The value of this register should be configured by CPU.

25.4.2.3 SPDIO_TXBD_WPTR

- Name: SPDIO_TXBD_WPTR Register
- Width: 32 bit
- Initial Value: 0x0
- Offset: 01A8h

Bit	Access	INI	Symbol	Description
31	R		SPDIO_TXBD_H2C_WPTR_WRAP	Toggle once wptr reaches txbd_size
30:16	-		RSVD	Reserved
15:0	R		SPDIO_TXBD_H2C_WPTR	When the packet has been sent from host to Tx Buffer, this index should be updated by SDIO.

25.4.2.4 SPDIO_TXBD_RPTR

- Name: SPDIO_TXBD_RPTR Register
- Width: 32 bit
- Initial Value: 0x0
- Offset: 01ACh

Bit	Access	INI	Symbol	Description
31	CPU: R/W DRV: R		SPDIO_TXBD_H2C_RPTR_WRAP	Software should write(toggle) this bit once rptr reaches txbd_size
30:16	-		RSVD	Reserved
15:0	CPU: R/W DRV: R		SPDIO_TXBD_H2C_RPTR	When the packet has been processed by CPU and moved to MAC Tx FIFO, this index should be updated by CPU.

25.4.2.5 SPDIO_RXBD_ADDR

- Name: SPDIO_RXBD_ADDR Register
- Width: 32 bit
- Initial Value: 0x0
- Offset: 01B0h

Bit	Access	INI	Symbol	Description
31:0	CPU: R/W DRV: R		SPDIO_RXBD_ADDR	The base address of RXBD. The value of this register should be configured by CPU. It should be 8-Byte alignment

All information provided in this document is subject to legal disclaimers.
© REALTEK 2024. All rights reserved.

25.4.2.6 SPDIO_RXBD_NUM

- Name: SPDIO_RXBD_NUM Register
- Width: 16 bit
- Initial Value: 0x0100
- Offset: 01B4h

Bit	Access	INI	Symbol	Description
15:0	CPU: R/W DRV: R		SPDIO_RXBD_NUM	The total count of RXBD The value of this register should be configured by CPU. The unit is BD count.

25.4.2.7 SPDIO_RXBD_C2H_WPTR

- Name: SPDIO_RXBD_C2H_WPTR Register
- Width: 16 bit
- Initial Value: 0x0
- Offset: 01B6h

Bit	Access	INI	Symbol	Description
15:0	CPU: R/W DRV: R		SPDIO_RXBD_C2H_WPTR	When the packet has been processed by CPU and to be moved to host, this index should be updated by CPU.

25.4.2.8 SPDIO_RXBD_C2H_RPTR

- Name: SPDIO_RXBD_C2H_RPTR Register
- Width: 16 bit
- Initial Value: 0x0
- Offset: 01B8h

Bit	Access	INI	Symbol	Description
15:0	CPU: R DRV: R		SPDIO_RXBD_C2H_RPTR	When the packet has been moved to host, this index should be updated by SDIO.

25.4.2.9 HCI_RX_REQ

- Name: HCI_RX_REQ Register
- Width: 8 bit
- Initial Value: 0x0
- Offset: 01BAh

Bit	Access	INI	Symbol	Description
7:1	-		RSVD	Reserved
0	CPU: R/W DRV: R		HCI_RX_REQ	CPU triggers this bit to enable SDIO Rx transfer by fetch BD info. SDIO fetches BD to get the Rx length and address and transfer Rx packet to host, and clear this bit when all BD transfer done.

25.4.2.10 CPU_RST_SDIO_DMA

- Name: CPU_RST_SDIO_DMA Register
- Width: 8 bit
- Initial Value: 0x0
- Offset: 01BBh

Bit	Access	INI	Symbol	Description
7	CPU: R/W DRV: R		CPU_RST_SDIO_DMA	CPU sets this bit to reset SDIO DMA. This bit is auto-cleared.
6:1	-		RSVD	Reserved

All information provided in this document is subject to legal disclaimers.
© REALTEK 2024. All rights reserved.

0	CPU: R/W DRV: R		SDIO_DAT_EDGE_INV	Invert SDIO latch input data edge.

25.4.2.11 SPDIO_RX_REQ_ADDR

- Name: SPDIO_RX_REQ_ADDR Register
- Width: 32 bit
- Initial Value: 0x0
- Offset: 01BCh

Bit	Access	INI	Symbol	Description
31:0	CPU: R/W DRV: R		SPDIO_RX_REQ_ADDR	The address of the requested Rx packet. This is unnecessary for driver.

25.4.2.12 SPDIO_CPU_INT_MASK

- Name: SPDIO_CPU_INT_MASK Register
- Width: 16 bit
- Initial Value: 0x0
- Offset: 01C0h

Bit	Access	INI	Symbol	Description
15:12	-		RSVD	Reserved
11	CPU: R/W DRV: R		HOST_CMD11_INT_MSK	This bit is clear, and HOST_CMD11_INT is masked.
10	CPU: R/W DRV: R		HOST_WAKE_CPU_INT_MSK	This bit is clear, and HOST_WAKE_CPU_INT is masked.
9	CPU: R/W DRV: R		RX_BD_AVAI_INT_MSK	This bit is clear, and RX_BD_AVAI_INT is masked.
8	CPU: R/W DRV: R		RX_BD_FLAG_ERR_INT_MSK	This bit is clear, and RX_BD_FLAG_ERR_INT is masked.
7	CPU: R/W DRV: R		SDIO_RST_CMD_INT_MSK	This bit is clear, and SDIO_RST_CMD_INT is masked.
6	CPU: R/W DRV: R		RPWM2_INT_MSK	This bit is clear, and RPWM2_INT is masked.
5	CPU: R/W DRV: R		RPWM_INT_MSK	This bit is clear, and RPWM_INT is masked.
4	CPU: R/W DRV: R		H2C_MSG_INT_MSK	This bit is clear, and H2C_MSG_INT is masked.
3	CPU: R/W DRV: R		C2H_DMA_OK_MSK	This bit is clear, and C2H_DMA_OK _INT is masked.
2	CPU: R/W DRV: R		H2C_DMA_OK_MSK	This bit is clear, and H2C_DMA_OK _INT is masked.
1	CPU: R/W DRV: R		H2C_BUS_RES_FAIL_MSK	This bit is clear, and H2C_BUS_RES_FAIL_INT is masked.
0	CPU: R/W DRV: R		TXBD_H2C_OVF _MSK	This bit is clear, and TXBD_H2C_OVF is masked.

25.4.2.13 SPDIO_CPU_INT_REG

- Name: SPDIO_CPU_INT_REG Register
- Width: 16 bit
- Initial Value: 0x0
- Offset: 01C2h

Bit	Access	INI	Symbol	Description
15:12	-		RSVD	Reserved
11	CPU: R/W DRV: R		HOST_CMD11_INT	When Host sends CMD11 to notify that the signal voltage level is going to switch to 1.8v, the HW should issue this interrupt to notify the local CPU to do the LDO

All information provided in this document is subject to legal disclaimers.
© REALTEK 2024. All rights reserved.

				power voltage level switch.
10	CPU: R/W DRV: R		HOST_WAKE_CPU_INT	When Host Send TRX CMD53 while CPU is not ready (SYSTEM_CPU_RDY_IND=0, or CPU_RDY_IND=0), trigger this interrupt to wake CPU, and then indicate BUSY status to host. CPU should be able to receive this interrupt even CPU clock is gated; therefore, this interrupt should be connected to System On circuit.
9	CPU: R/W DRV: R		RX_BD_AVAI_INT	If the free RXBD number becomes larger than FREE_RXBD_COUNT (0x1D8), trigger this interrupt. This interrupt triggers only once when free RXBD number crosses FREE_RXBD_COUNT
8	CPU: R/W DRV: R		RX_BD_FLAG_ERR_INT	Trigger by SDIO to CPU when detect RX_BD error
7	CPU: R/W DRV: R		SDIO_RST_CMD_INT	Trigger by SDIO to CPU when SDIO is reset. CPU should be able to receive this interrupt even CPU clock is gated; therefore, this interrupt should be connected to System On circuit.
6	CPU: R/W DRV: R		RPWM2_INT	Trigger by SDIO to CPU that RPWM2 occurs (HRPWM is toggled) CPU should be able to receive this interrupt even CPU clock is gated; therefore, this interrupt should be connected to System On circuit.
5	CPU: R/W DRV: R		RPWM1_INT	Trigger by SDIO to CPU that RPWM1 occurs(HRPWM2 is written) CPU should be able to receive this interrupt even CPU clock is gated; therefore, this interrupt should be connected to System On circuit.
4	CPU: R/W DRV: R		H2C_MSG_INT	Trigger by SDIO to CPU that H2C_MSG_INT occurs (SDIO_H2C_MSG is written)
3	CPU: R/W DRV: R		C2H_DMA_OK	Trigger by SDIO to CPU that packet is sent from RXFIFO to Host
2	CPU: R/W DRV: R		H2C_DMA_OK_	Trigger by SDIO to CPU that packet is sent from Host to TXFIFO
1	CPU: R/W DRV: R		H2C_BUS_RES_FAIL	If SDIO is going to move packet to Tx FIFO and fails to get bus resource, this interrupt raise. Driver writes 1 to clear.
0	CPU: R/W DRV: R		TXBD_H2C_OVF	If there is not enough TX_BD for Tx Packet, this bit raise. Driver writes 1 to clear.

25.4.2.14 CCPWM

- Name: CCPWM Register
- Width: 8 bit
- Initial Value: 0x0
- Offset: 01C4h

Bit	Access	INI	Symbol	Description
7	CPU: R/W DRV: R		TOGGLING	Toggling Bit: this is the one bit sequence number field. Interrupt is issued when this bit is changed from “1” to “0” or from “0” to “1”.
6:4	-		RSVD	Reserved

25.4.2.15 SYS_CPU_INDICATION

- Name: SYS_CPU_INDICATION Register
- Width: 8 bit
- Initial Value: 0x0
- Offset: 01C5h

Bit	Access	INI	Symbol	Description
7:2	-		RSVD	Reserved
1	R/W		READEE_DONE	After CCCR register is configured, write this bit to 1
0	R/W	0	SYSTEM_CPU_RDY_IND	Used to indicate that CPU is ready for TRX. This bit is synchronized to CPU_RDY_IND (offset: 0x87)

All information provided in this document is subject to legal disclaimers.
© REALTEK 2024. All rights reserved.

25.4.2.16 CCPWM2

- Name: CCPWM2 Register
- Width: 16 bit
- Initial Value: 0x0
- Offset: 01C6h

Bit	Access	INI	Symbol	Description
15	CPU: R/W DRV: R		TOGGLING	Toggling Bit: this is the one bit sequence number field. Interrupt is issued when this bit is changed from “1” to “0” or from “0” to “1”.
14:0	-		RSVD	Reserved

25.4.2.17 CPU_H2C_MSG

- Name: CPU_H2C_MSG Register
- Width: 32 bit
- Initial Value: 0x0
- Offset: 01C8h

Bit	Access	INI	Symbol	Description
31	CPU: R DRV: R		TOGGLING	Toggling Bit: this is the one bit sequence number field. Interrupt is issued when this bit is changed from “1” to “0” or from “0” to “1”.
30:0	CPU: R DRV: R		CPU_H2C_MSG	Driver to FW Message

25.4.2.18 CPU_C2H_MSG

- Name: CPU_C2H_MSG Register
- Width: 32 bit
- Initial Value: 0x0
- Offset: 01CCh

Bit	Access	INI	Symbol	Description
31	CPU: R/W DRV: R		TOGGLING	Toggling Bit: this is the one bit sequence number field. Interrupt is issued when this bit is changed from “1” to “0” or from “0” to “1”.
30:0	CPU: R/W DRV: R		CPU_C2H_MSG	FW to Driver Message

25.4.2.19 CRPWM

- Name: CRPWM Register
- Width: 8 bit
- Initial Value: 0x7
- Offset: 01D0h

Bit	Access	INI	Symbol	Description
7	CPU: R DRV: R		TOGGLING	Toggling Bit: this is the one bit sequence number field. Interrupt is issued when this bit is changed from “1” to “0” or from “0” to “1”.
6:4	-		RSVD	Reserved

25.4.2.20 CRPWM2

- Name: CRPWM2 Register
- Width: 16 bit
- Initial Value: 0x0
- Offset: 01D2h

All information provided in this document is subject to legal disclaimers.
© REALTEK 2024. All rights reserved.

Bit	Access	INI	Symbol	Description
15	CPU: R DRV: R		TOGGLING	Toggling Bit: this is the one bit sequence number field. Interrupt is issued when this bit is changed from “1” to “0” or from “0” to “1”.
14:0	-		RSVD	Reserved

25.4.2.21 AHB_DMA_CTRL

- Name: AHB_DMA_CTRL Register
- Width: 32 bit
- Initial Value: 0x74004040
- Offset: 01D4h

Bit	Access	INI	Symbol	Description
31	CPU: R/W DRV: R		DISPATCH_TXAGG_PKT	Enable SPDIO to dispatch the Aggregated Tx packet.
30:28	CPU: R/W DRV: R	3’b111	AHB_BURST_TYPE	This field is used to control to SPDIO AHB support Burst type. 3’b100: Support Burst 16 DW 3’b010: Support Burst 8 DW 3’b001: Support Burst 4 DW Default is 3’b111, all support.
27:24	CPU: R/W DRV: R		RX_AHB_BUSY_WAIT_CNT	When SPDIO Rx transfer, AHB controller will wait BUSY counter for AHB access target not response READY signal. If timeout, AHB controller will issue AHB_BUS_RES_FAIL INT to CPU.
23:22	CPU: R DRV: R		AHBM_SPDIO_TRANS	SPDIO AHB Master HTRANS signal
21	CPU: R DRV: R		AHBM_SPDIO_READY	SPDIO AHB Master HREADY signal
20	-		RSVD	Reserved
19:16	CPU: R DRV: R		AHB_DMA_CS	AHB DMA state
15	-		RSVD	Reserved
14:8	CPU: R/W DRV: R		SPDIO_RXFF_WLEVEL	SPDIO FIFO Rx water level Range 1~126, it can be modified only when AHB_DMA_CS = 000
7	-		RSVD	Reserved
6:0	CPU: R/W DRV: R		SPDIO_TXFF_WLEVEL	SPDIO FIFO Tx water level Range 1~126, it can be modified only when AHB_DMA_CS = 000

25.4.2.22 FREE_RXBD_COUNT

- Name: FREE_RXBD_COUNT Register
- Width: 32 bit
- Initial Value: 0x10100
- Offset: 01D8h

Bit	Access	INI	Symbol	Description
31:16	CPU: R/W DRV: R		FREE_RXBD_COUNT	If SPDIO_RXBD_C2H_RPTR is updated and the free RXBD Number is larger than FREE_RXBD_COUNT, trigger RX_BD_AVAI_INT interrupt.
15:8	CPU: R/W DRV: R		TX_BUFF_UNIT_SIZE	The Size of each single Tx Buffer which is addressed by TX_BD Unit: 64Byte Ex: 0x01: 64Byte 0x10: 1024Byte
7	-		RSVD	Reserved
6:0	CPU: R DRV: R		FIFO_CNT	The SPDIO Local FIFO counter, for debug usage

25.4.2.23 CPU_H2C_MSG_EXT

- Name: CPU_H2C_MSG_EXT Register
- Width: 32 bit

All information provided in this document is subject to legal disclaimers.
© REALTEK 2024. All rights reserved.

- Initial Value: 0x0
- Offset: 01DCh

Bit	Access	INI	Symbol	Description
31	R		TOGGLING	Toggling Bit: this is the one bit sequence number field. Interrupt is issued when this bit is changed from “1” to “0” or from “0” to “1”.
30:0	R		CPU_H2C_MSG_EXT	Driver to FW Message

25.4.2.24 CPU_C2H_MSG_EXT

- Name: CPU_C2H_MSG_EXT Register
- Width: 32 bit
- Initial Value: 0x0
- Offset: 01E0h

Bit	Access	INI	Symbol	Description
31	R/W		TOGGLING	Toggling Bit: this is the one bit sequence number field. Interrupt is issued when this bit is changed from “1” to “0” or from “0” to “1”.
30:0	R/W		CPU_C2H_MSG_EXT	FW to driver Message

25.4.3 CCCR Bit Definition

Some CCCR bit are configurable. The base address of CCCR bit definition is 0x4002D000. Only CPU can access this area.

NOTE

In general, users do not need to configure these control bits. If needed, users must configure it before SDIO initialization, otherwise it will not work.

Offset	Name	Bit	INI	Description
Byte 0	SCSI	[0]	0	CCCR: 0x07 bit[6]
	SDC	[1]	1	CCCR: 0x08 bit[0]
	SMB	[2]	1	CCCR: 0x08 bit[1]
	S4MI	[3]	1	CCCR: 0x08 bit[4]
	SMPC	[4]	1	CCCR: 0x12 bit[0]
	SHS	[5]	1	CCCR: 0x13 bit[0]
	SSDR50	[6]	0	CCCR: 0x14 bit[0]
	SSDR104	[7]	0	CCCR: 0x14 bit[1]
Byte 1	SDDR50	[0]	0	CCCR: 0x14 bit[2]
	SDTA	[1]	0	CCCR: 0x15 bit[0]
	SDTC	[2]	0	CCCR: 0x15 bit[1]
	SDTD	[3]	0	CCCR: 0x15 bit[2]
	SAI	[4]	0	CCCR: 0x16 bit[0]
	INIT_SKIP	[5]	0	0: need to use CMD 0 5 5 3 7 to enable Wi-Fi 1: skip
	SDIO_VOLTAGE	[6]	0	Auto-load to SDIO local 0x86[3] 0: 3.3V 1: 1.8V
	SDIO_INTR_CTRL	[7]	1	Auto-load to SDIO local 0x86[4] 0: disable 1: enable SDIO interrupt function
Byte 2	SPS	[0]	1	FBR: 0x102 bit[0]
	RSVD	[3:1]	000	
	PS3	[7:4]	0000	FBR: 0x102 bit[7:4]
Byte 3	CCCR	[3:0]	0010	CCCR: 0x0 bit[3:0]
	SDIO_CMD_FORCE_VLD	[4]	0	Auto-load to SDIO local 0x00[5], force all SDIO CMD legacy
	SDIO_S18A	[5]	0	Auto-load to SDIO local 0x00[6] 1: switching to 1.8V accepted
	RESV	[7:6]	00	
Byte 4	SDIOX	[3:0]	0011	CCCR:0 bit[3:0]

All information provided in this document is subject to legal disclaimers.
© REALTEK 2024. All rights reserved.

	SDX	[7:4]	0010	CCCR:1 bit[7:4]
Byte 5	OCR	[7:0]	0	OCR[7:0]
Byte 6	OCR	[7:0]	0	OCR[15:8]
Byte 7	OCR	[7:0]	FC	OCR[23:16]
Byte 8	COMMON CIS DATA	[7:0]	20	MANFID
Byte 9		[7:0]	04	Data length
Byte A		[7:0]	4c	MANF SDIO card manufacturer code
Byte B		[7:0]	02	MANF SDIO card manufacturer code
Byte C		[7:0]		Manufacturer information (Part Number and/or Revision) 8195AM: 95 8711: 11
Byte D		[7:0]		manufacturer information (Part Number and/or Revision) 8195AM: 81 8711: 87
Byte E		[7:0]	21	FUNCID
Byte F		[7:0]	02	Data length
Byte 10		[7:0]	0c	Card function code 0Ch: PCMCIA
Byte 11		[7:0]	00	System initialization bit mask. (Not used, set to 00h)
Byte 12		[7:0]	22	FUNCE
Byte 13		[7:0]	04	Data length
Byte 14		[7:0]	00	Type of extended data
Byte 15		[7:0]	08	Maximum block size and byte count that function 0 can support
Byte 16		[7:0]	00	
Byte 17		[7:0]	32	The maximum transfer rate per one data line during data transfer. Bit[2:0] contains the transfer rate unit coded as follows: 0=100kbit/s 1=1Mbit/s 2=10Mbit/s 3=100Mbit/s 4...7=reserved Bit[6:3] contains the time value codes as follows: 0=reserved 1=1.0 2=1.2 3=1.3 4=1.5 5=2.0 6=2.5 7=3.0 8=3.5 9=4.0 A=4.5 B=5.0 C=5.5 D=6.0 E=7.0 F=8.0
Byte 18		[7:0]	FF	End
Byte 19	Function 1 CIS Data	[7:0]	21	FUNCID
Byte 1A		[7:0]	02	Data length
Byte 1B		[7:0]	0c	Card function code 0Ch: PCMCIA
Byte 1C		[7:0]	00	System initialization bit mask. (Not used, set to 00h)
Byte 1D		[7:0]	22	Function1 FUNCE
Byte 1E		[7:0]	2a	Data length
Byte 1F		[7:0]	01	Type of extended data (01h)
Byte 10		[7:0]	01	Bit[0]: wake up support, may be placed into a low power state using some function standard or vendor defined method and the SDCLK can be stopped with the power remaining applied to the card.
Byte 21		[7:0]	00	STD_IO_REV: contains the version level of the Application

All information provided in this document is subject to legal disclaimers.
© REALTEK 2024. All rights reserved.

				Specification for Standard SDIO Functions that this function supports. The format is x.y, where x is the major version (4-bit) and y is the minor version level. For example, if the version is 2.4 the value would be 24h. If this function does not support an SDIO standard function, the value shall be 00h.
Byte 22		[7:0]	00	Product Serial Number: Support of a serial number is optional
Byte 23		[7:0]	00	
Byte 24		[7:0]	00	
Byte 25		[7:0]	00	
Byte 26		[7:0]	00	Size of the CSA space available for this function in bytes
Byte 27		[7:0]	00	
Byte 28		[7:0]	00	
Byte 29		[7:0]	00	
Byte 2A		[7:0]	00	Contains flags identifying properties of this function's CSA.
Byte 2B		[7:0]	00	Maximum block size and byte count that this function can support.
Byte 2C		[7:0]	02	
Byte 2D		[7:0]	00	The OCR value for this function. The format is identical to the 32-bit OCR format used by SD memory devices.
Byte 2E		[7:0]	FF	
Byte 2F		[7:0]	FF	
Byte 30		[7:0]	00	
Byte 31		[7:0]	00	The minimum current at 2.7V operating voltage, in mA required by this function when operating. This value is valid for all voltages supported by this function. If the required current exceeds 200mA, this value shall be zero.
Byte 32		[7:0]	00	The average current at 3.3V operating voltage, in mA, required by this function when operating. This value is valid for all voltages supported by this function. If the required current exceeds 200mA, this value shall be zero.
Byte 33		[7:0]	00	The maximum (peak) current at 3.6V operating voltage, in mA, required by this function when operating. This value is valid for all voltages supported by this function. If the required current exceeds 200mA, this value shall be zero.
Byte 34		[7:0]	00	The minimum current, in mA, required by this function when in the standby condition. If this function does not support standby, this value shall be 00h.
Byte 35		[7:0]	00	The average current, in mA, required by this function when in the standby condition. If this function does not support standby, this value shall be 00h.
Byte 36		[7:0]	00	The maximum current, in mA, required by this function when in the standby condition. If this function does not support standby, this value shall be 00h.
Byte 37		[7:0]	00	The minimum data transfer bandwidth, in KB/sec, needed by this function to successfully operate. If this function has no minimum necessary bandwidth, these bytes shall be 0000h.
Byte 38		[7:0]	00	
Byte 39		[7:0]	00	The data transfer bandwidth, in KB/sec, needed by this function to function at an optimum level. If this function has no optimum bandwidth, these bytes shall be 0000h.
Byte 3A		[7:0]	00	
Byte 3B		[7:0]	00	Indicates the function's required time-out value for coming ready after being enabled. If the cards required no time-out, this field shall be set to 0000h.
Byte 3C		[7:0]	00	
Byte 3D		[7:0]	00	This is the average current at 3.3V operating voltage, in mA, required by this function when operating. This value is valid for all voltages supported by this function. If the required current exceeds 200mA, this value shall be zero.
Byte 3E		[7:0]	00	
Byte 3F		[7:0]	00	The maximum (peak) current at 3.6V operating voltage, in mA, required by this function when operating. This value is valid for all voltages supported by this function. If the required current exceeds 200mA, this value shall be zero.
Byte 110		[7:0]	00	
Byte 41		[7:0]	eb	This 16-bit value indicates the average current, in mA, required by this function when operating in the Higher Current Mode (EMPC=1 & EPS=0). This value indicates a current range from 1 to 65535 mA.
Byte 42		[7:0]	00	
Byte 43		[7:0]	6e	This 16-bit value indicates the peak current, in mA, required by this function when operating in the Higher Current Mode. This value indicates a current range from 1 to 65535 mA.
Byte 44		[7:0]	01	
Byte 45		[7:0]	00	This 16-bit value indicates the average current, in mA, required by this function when operating in the Lower Current Mode (EMPC=1 & EPS=1). This value indicates a current range from 1 to 65535 mA.
Byte 46		[7:0]	00	
Byte 47		[7:0]	00	This 16-bit value indicated the maximum (peak) current, in mA, required by this function when operating in the lower current mode (EMPC=1 & EPS=1). This value indicates a current range from 1 to 65535 mA.
Byte 48		[7:0]	00	

All information provided in this document is subject to legal disclaimers.
© REALTEK 2024. All rights reserved.

Byte 49		[7:0]	FF	End
Byte 4A		[7:0]	FF	MAC address: [7:0]
Byte 4B		[7:0]	FF	MAC address: [15:8]
Byte 4C		[7:0]	FF	MAC address: [23:16]
Byte 4D		[7:0]	FF	MAC address: [31:24]
Byte 4E		[7:0]	FF	MAC address: [39:32]
Byte 4F		[7:0]	FF	MAC address: [47:40]
Byte 50		[7:0]	00h	Fn1_cis[447:440]
Byte 62	CMD11_EN	[0]	1	Auto-load to SDIO local 0x04[0]
	CMD11_SAMPLE_NUM	[3:1]	001	Auto-load to SDIO local 0x04[3:1]
	CMD11_END_DLY	[7:4]	0001	Auto-load to SDIO local 0x04[7:4]

All information provided in this document is subject to legal disclaimers.
© REALTEK 2024. All rights reserved.

26 Key-Scan

26.1 Overview

As a keypad scan device, Key-Scan can be applied to simple key, remote control or even game pad. It needs to scan the operations of key press and release accurately and timely.

The major benefit of this device is to free up the CPU from scanning the keypad all the time. It triggers the corresponding interrupts to inform CPU in time. In addition, chip can enter low-power state in most of time, and take little time to wake up and handle the key events.

26.2 Features

- Up to 8 *8 (64) keypad array with use of 16 GPIOs
- Configurable rows and columns of keypad array
- Hardware debounce with configurable time at each scan
- Configurable Scan Clock, Scan Interval, and Release Time
- Interrupts, interrupts mask, interrupts clear, interrupts status
- Multi-key detect
- FIFO with width of 12 bits and depth of 16 to store Key Press and Release Events
- Two work modes: Event (Press and Release) Trigger Mode and Regular Scan Mode (high priority)
- Low power mode: Key press event can wakeup CPU from sleep
- Support stuck key handling

26.3 Functional Description

26.3.1 Block Diagram

The block diagram of Key-Scan is illustrated in *Figure 26-1*. The Key-Scan contains six main parts:

- Interrupt Control: control and manage interrupts.
- Control Register and FIFO: configure Key-Scan parameters and FIFO.
- Clock: Key-Scan clock, 131K clock is used for stuck row detection.
- Keypad Control: contains Key-Scan control, wake up control and key input/output control.
- Keypad: up to 8*8 keypad array with use of 16 GPIOs.

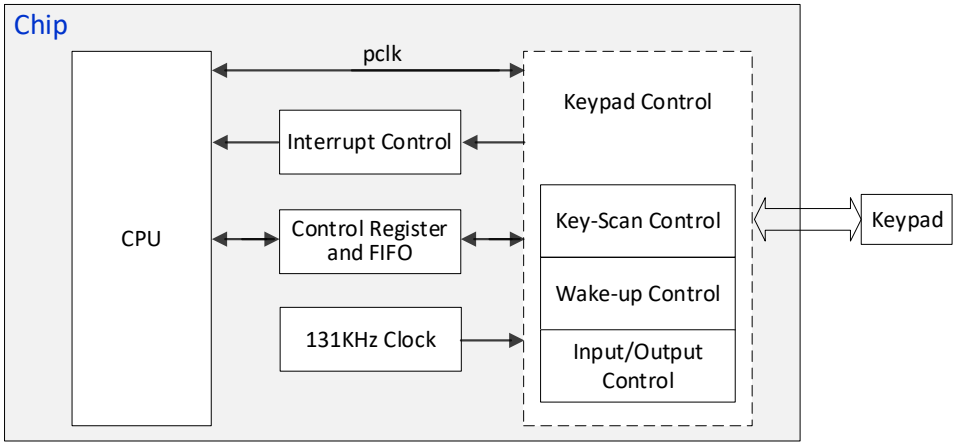


Figure 26-1 Key-Scan block diagram

Taking 8*8 keypad array for example, the typical application setup is depicted in *Figure 26-2*, and external keypad is needed.

All information provided in this document is subject to legal disclaimers.
© REALTEK 2024. All rights reserved.

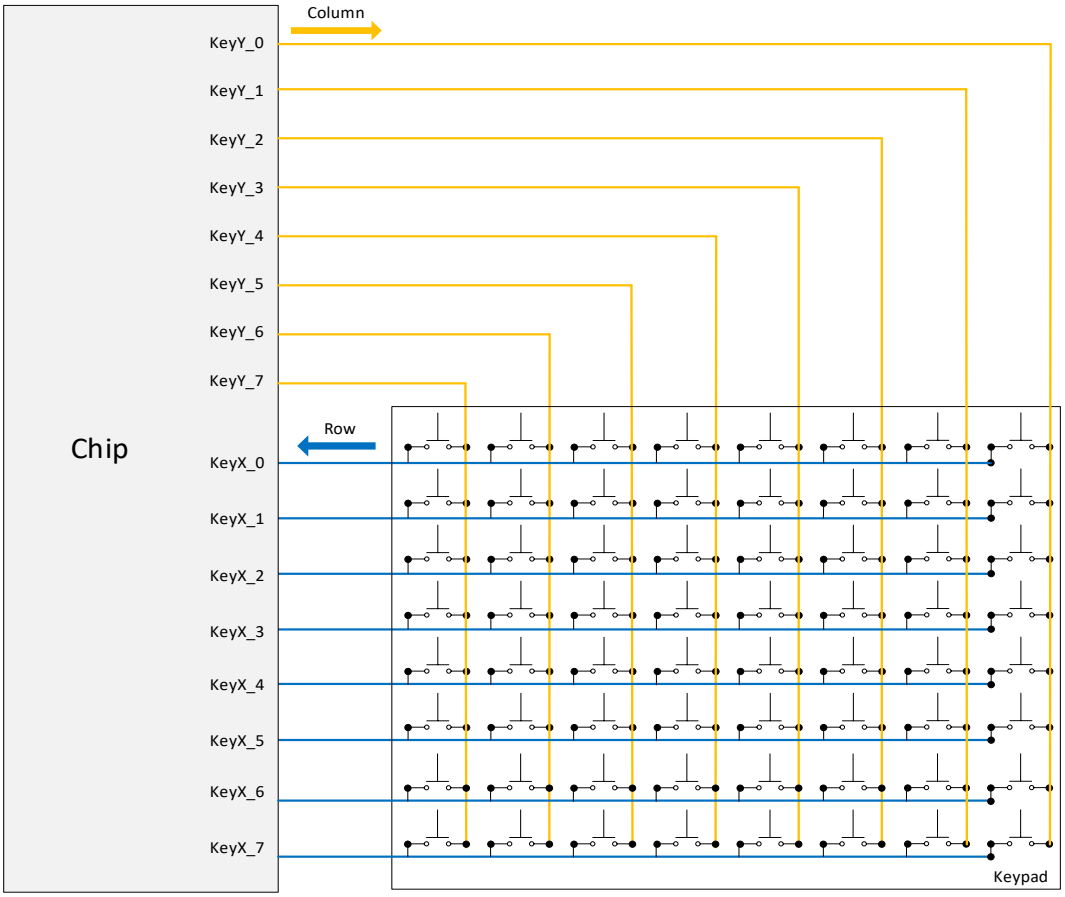


Figure 26-2 Typical application setup with external keypad

26.3.2 Work Principle

At the beginning, all columns output low, and all rows are set as input with internal pull up.

When there is a key or multiple keys pressed (short between the Column and the Row), it triggers an internal state machine to start a Key-Scan cycle to determine the column and row of the key that is pressed. The state machine sets first column as an output low and all other columns are in Hi-Z state. The state machine then scans all rows to determine which keys are being pressed, and then second column is output low until the last column. Firstly, scan circuit waits for a first debounce time and does first scan. Then, the circuit waits for a debounce time and scans again. After the two scan cycles, the circuit decides the current key status, and compares with internal key status table to find which key is pressed or released. After a full scan, the key press or release events are stored in the key event FIFO. Debounce time for the keypad is for each full scan and is configurable, as shown in *Figure 26-3*.

Interval time between each full scan is configurable. Once all keys are detected release, there is a release timer to confirm that, then an all release interrupt is triggered and state machine enters idle state. The Key-Scan timing is shown in *Figure 26-4*.

All information provided in this document is subject to legal disclaimers.
© REALTEK 2024. All rights reserved.

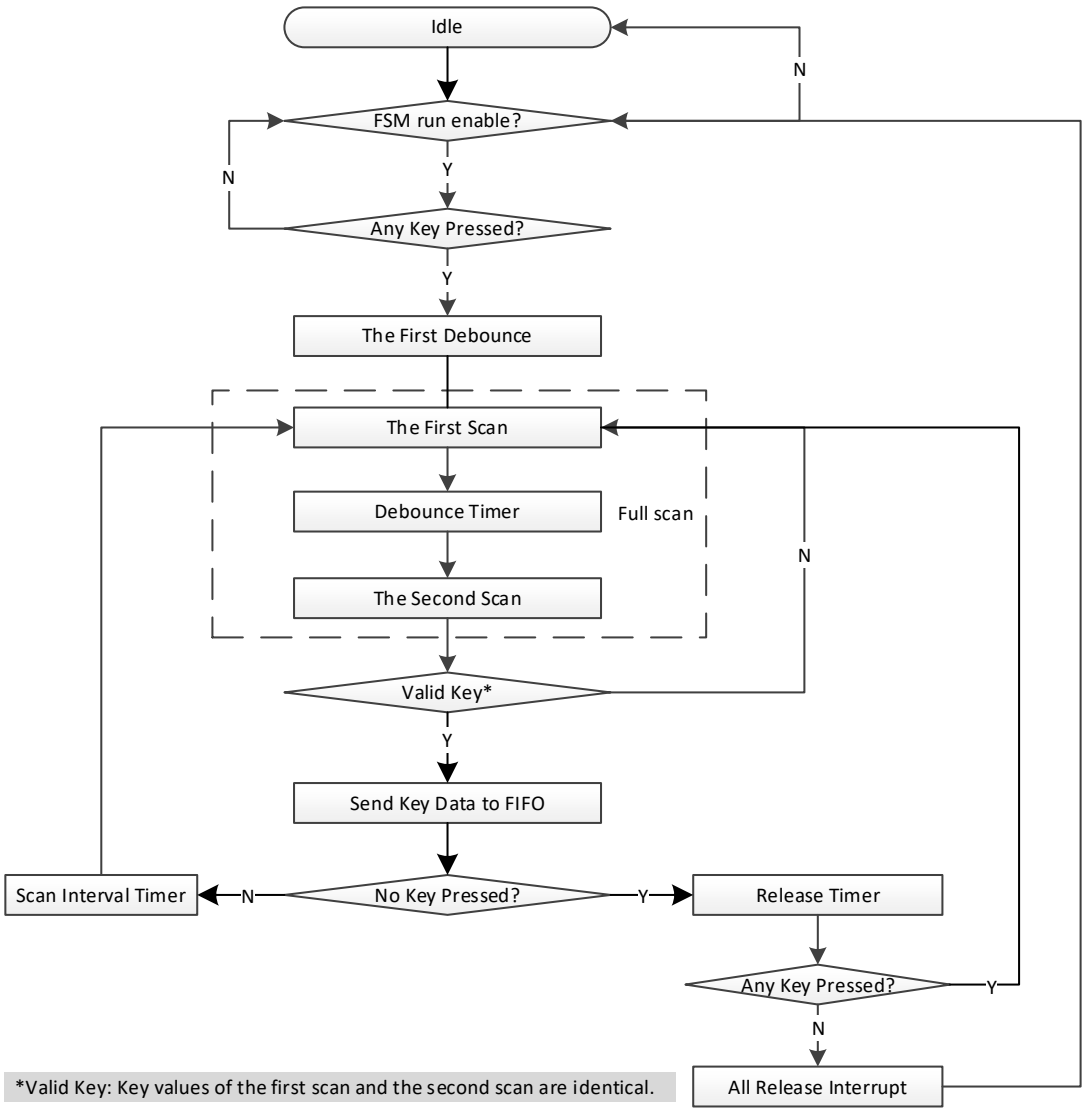


Figure 26-3 Key-Scan flow

All information provided in this document is subject to legal disclaimers.
© REALTEK 2024. All rights reserved.

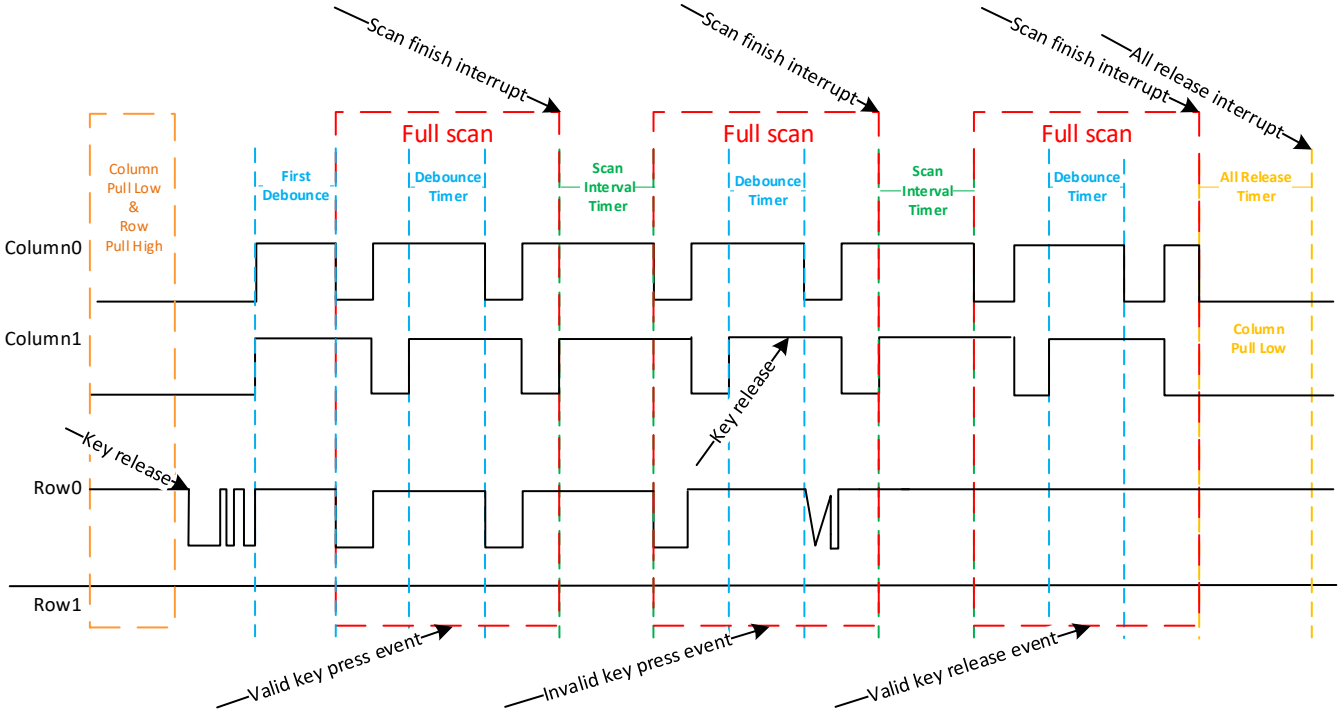


Figure 26-4 Key-Scan timing

There are two work modes: Event Trigger Mode and Regular Scan Mode, which are described in the following sections.

26.3.2.1 Event Trigger Mode

In Event Trigger Mode, key press or release event is stored in the key event FIFO only once in each key press and release operation.

26.3.2.2 Regular Scan Mode

In Regular Scan Mode, at each full scan, any key press event is stored in the key event FIFO until it is released (in this condition, only key press event is stored in the key event FIFO).

Figure 26-5 shows the difference of FIFO items between Event Trigger Mode and Regular Scan Mode during a key press and release.

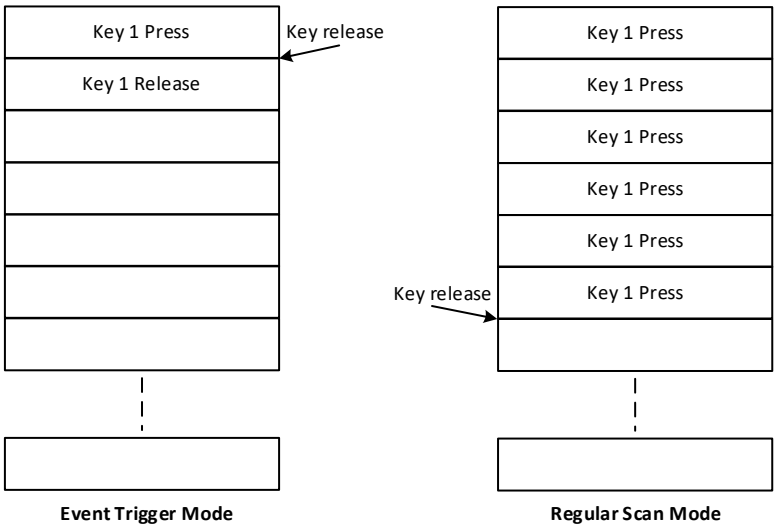


Figure 26-5 Difference of FIFO items between two work modes

All information provided in this document is subject to legal disclaimers.
© REALTEK 2024. All rights reserved.

26.3.3 FIFO Mechanism

The 12-bit FIFO with depth of 16 is used to store key press and release events, as shown in *Figure 26-6*.

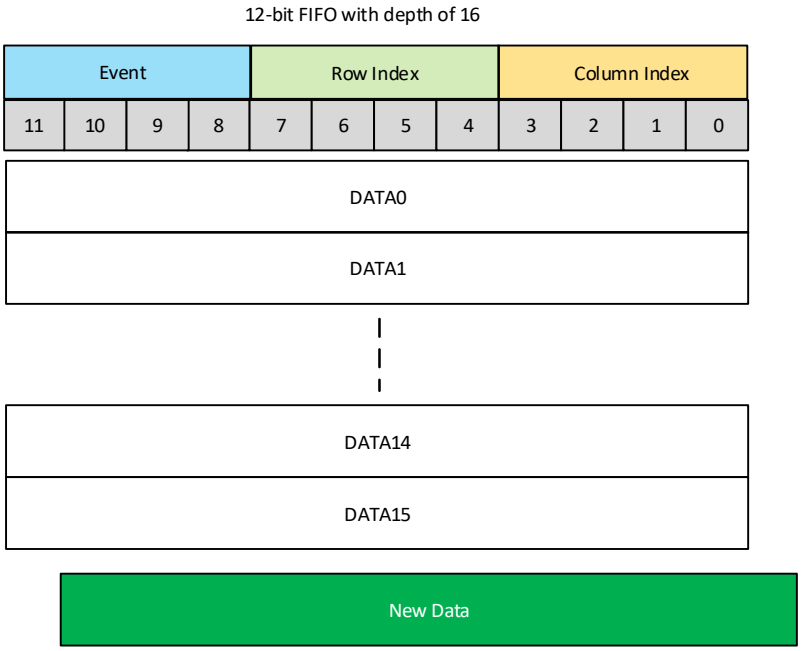


Figure 26-6 FIFO structure

Bit	Symbol	Description
8	Event	Indicates a key press or release event ● 1: a key press event, which can be cleared when read. ● 0: a key release event.
7:4	Row Index	Indicates the row of key # pressed or released in a keypad array ● 1: row 0 ● 2: row 1 ● 3: row 2 ● … ● 8: row 7
3:0	Column Index	Indicates the column of key # pressed or released in a keypad array ● 1: column 0 ● 2: column 1 ● 3: column 2 ● … ● 8: column 7

When the FIFO is full and another item is pushed into the FIFO, an overflow interrupt is triggered. When the FIFO is empty and is read, an overread interrupt is triggered.

The key value assignment is listed in *Table 26-1*.

Table 26-1 Key value assignment

Row \ Column	0	1	2	3	4	5	6	7
0	11	12	13	14	15	16	17	18
1	21	22	23	24	25	26	27	28
2	31	32	33	34	35	36	37	38
3	41	42	43	44	45	46	47	48
4	51	52	53	54	55	56	57	58
5	61	62	63	64	65	66	67	68
6	71	72	73	74	75	76	77	78
7	81	82	83	84	85	86	87	88

All information provided in this document is subject to legal disclaimers.
© REALTEK 2024. All rights reserved.

26.3.4 Clock Configuration

There are two clock domains in the Key-Scan design, as shown in *Figure 26-7*.

- The green part works in low-power clock domain, whose clock will not be gated in low power modes.
- The blue part works under bus clock domain (40MHz), whose clock will be gated in low power modes. In bus clock domain, some of the logic works in scan_clk frequency that is divided from bus clock.

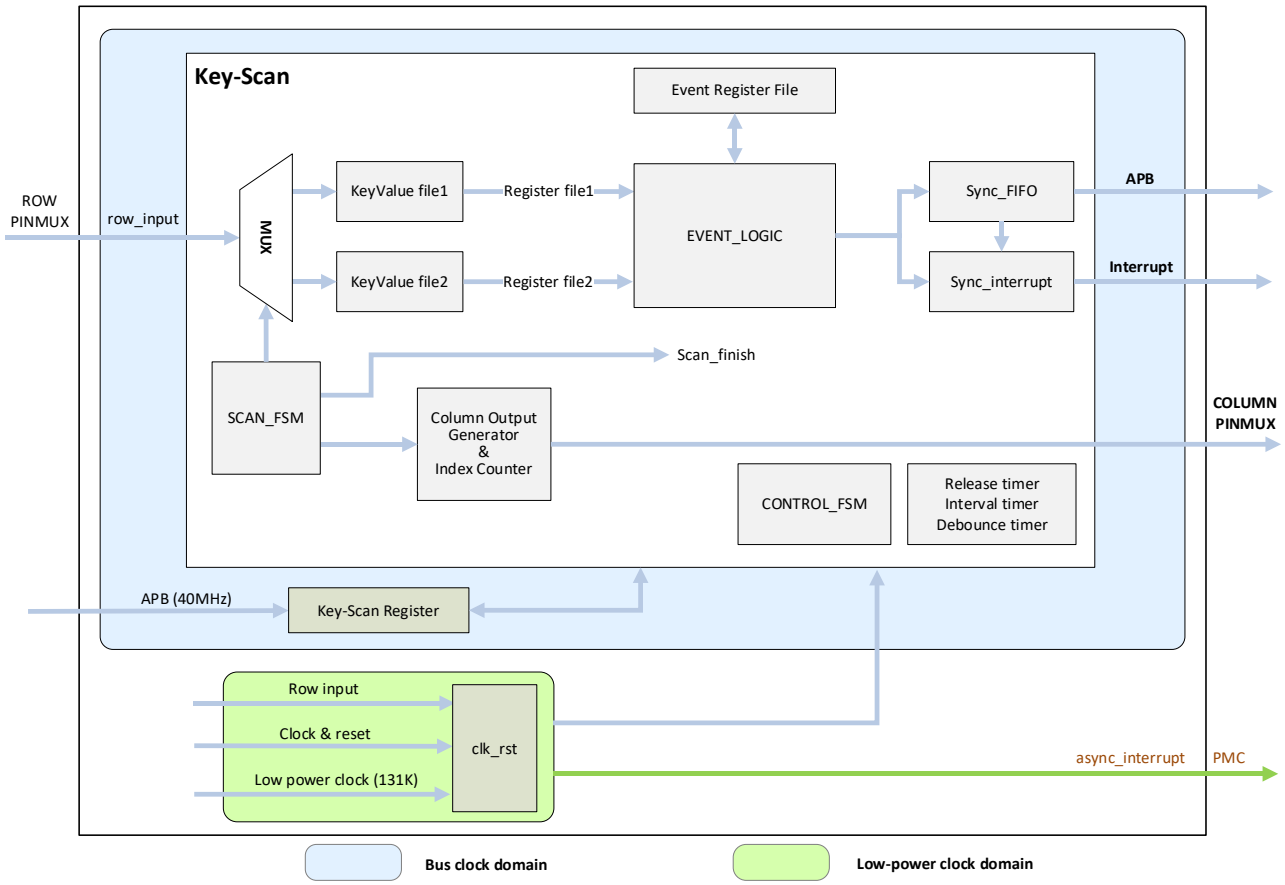


Figure 26-7 Clock domain diagram

The Key-Scan trigger signal is generated by lower frequency (131K). When the low power clock detects a key press, hardware starts scan by scan clock to confirm the key values.

26.3.5 Low Power Mode

26.3.5.1 Active

In active mode, the Key-Scan keeps power on all the time. When the Key-Scan scans column x (x is any column), it sets column x to output low, and all other columns are in Hi-Z state. Then, it scans all rows to determine which keys are being pressed or released, after this, one column scans end. The Key-Scan scans columns in sequence, and will be cyclic scanning column by column.

26.3.5.2 Sleep

In practical application scenarios, the Key-Scan keeping active will leads high power consumption. To save power consumption and optimize application, the Key-Scan often works in low power mode.

In sleep mode, CPU enters low power state. When a key or keys are pressed, it will wake up CPU and the Key-Scan will start to scan to determine which keys are being pressed. The Key-Scan keeps scanning when keys are pressed, after keys are released and there is no other event happens, CPU sleeps again.

All information provided in this document is subject to legal disclaimers.
© REALTEK 2024. All rights reserved.

26.3.6 Shadow Key Problem

The RTL8721Dx supports multiple key presses detection accurately. The applications requiring three-key combinations (such as <Ctrl><Alt><Del>, or any other combinations) must ensure that the three keys are wired in appropriate key positions to avoid shadow key (or appearing like that a 4th key has been pressed).

To avoid shadow key, it is best to keep 3-button combinations that are pressed on separate rows and columns. Considering the situation with the keypad is described in *Figure 26-8*.

In *Figure 26-8*, there is a 4*3 keypad matrix, connected to Row0 ~ Row3, and Column0 ~ Column2. All of the rows are configured as inputs with pull-up resistors. The columns are configured as outputs, driving low. When a key press is made, one of the row inputs is pulled low, letting the RTL8721Dx know that a key has been pressed, then the RTL8721Dx starts the key scanning algorithm. This algorithm sweeps the output low across the columns, such that only 1 column is driven low at a time. While this is done to each column, the RTL8721Dx reads the row inputs, to determine which keys on a column are being pressed.

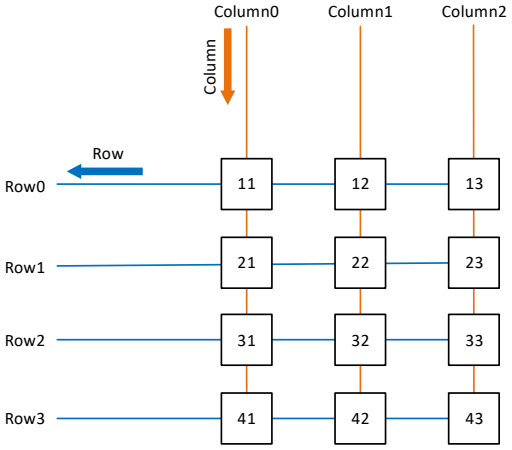


Figure 26-8 4*3 keypad example

The shadow key can occur when multiple keys are pressed, which makes it appear that additional keys (actually not being pressed) are being pressed. In *Figure 26-9*, key 11, key 12, and key 21 are pressed, which causes a shadow key issue. Since Row1 becomes pulled to ground through key 11 (which is pulled through key 12 when Column1 is transmitting a low), when Column1 is driving low, the RTL8721Dx sees a low signal at both Row0 and Row1. This falsely triggers key 22 as being pressed (the key highlighted as yellow).

The reason is that keypad matrix connects the columns and the rows together. When Column1 is driving low, the low gets transmitted onto Row0 via key 12. When key 11 is being pressed, this shorts Column0 to ground, and then shorts Row1 to Column0. In this process, Row1 is shorted to Column1, which is the reason shadow Key occurs.

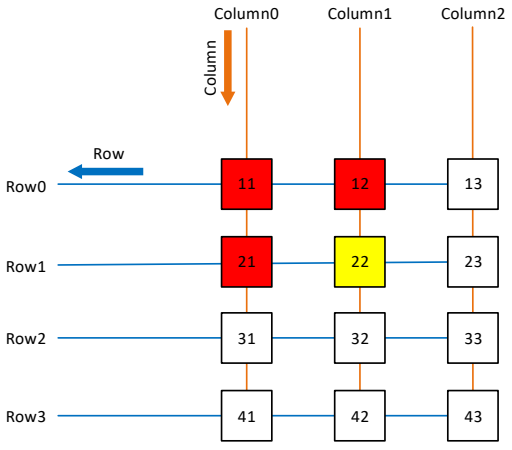


Figure 26-9 Shadow key condition

Keypad matrix can support multiple key presses properly when choosing the layout carefully. *Figure 26-10* shows a 3-button combination which works as expected.

All information provided in this document is subject to legal disclaimers.
© REALTEK 2024. All rights reserved.

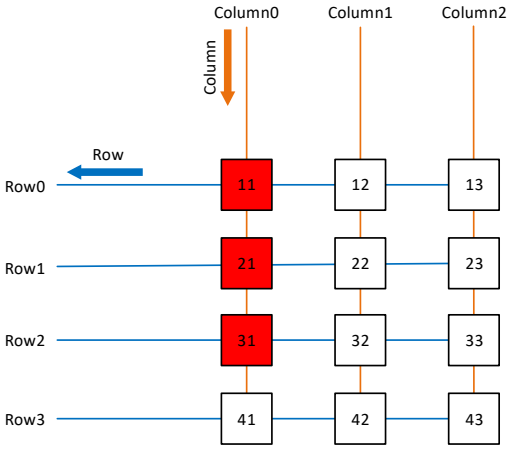


Figure 26-10 Correct 3-key condition

26.3.7 Stuck Key Problem

In actual application scenarios, the situation of key stuck (not released after being pressed) may occur. In the case of key stuck, to make sure that the system can still enter low power mode and can be woken up through Key-Scan, users can enable auto_check_key_stuck function.

26.3.7.1 Mechanism of auto_check_key_stuck Function

When the auto_check_key_stuck function is disabled, the circuit behavior is consistent with RTL8721Dx. That is to say, the state machine of Key-Scan will keep scanning as long as the key stuck, and cannot enter IDLE state unless the keys are released. Thus, the system cannot enter low power mode.

When the auto_check_key_stuck function is enabled,

(1) When the key stuck time exceeds the configured stuck_time_threshold, the stuck_event_raw_int_status bit in the KS_ISR_RAW register will be set. If the stuck event interrupt is enabled at the same time, the stuck_event_int_status bit in the KS_ISR register will be set, and the software will jump to the interrupt handling function of Key-Scan.

(2) In the interrupt handling function, the software can locate the stuck key or the row of stuck key by reading the KS_STUCK_KEY_STATUS and the KS_STUCK_KEY_ROW_STATUS1 registers, and sets the default state of the stuck row through the KS_STUCK_CFG1 register. At the same time, the software clears the current stuck interrupt by writing the KS_ICR register.

(3) When all default interrupt is enabled, the software will jump to the all default interrupt handling function.

(4) The next time when the state machine of Key-Scan scans, it will detect that the current state of the stuck row is the same as the configured default state, and consider the state is normal, then enter IDLE state. Thus, the system can smoothly enter the low power mode.

(5) In the IDLE state, if the stuck key is released, the Key-Scan will wake up the system and trigger scanning since it detects that the current state of the stuck row is different from the configured default state, and report a stuck interrupt at the same time. If all keys are released and all release interrupt is enabled, then an all release interrupt will be generated.

(6) The software gets the state of the stuck row by reading the KS_STUCK_KEY_ROW_STATUS1 register in stuck interrupt handling function., and sets the default state of the stuck row again.

(7) If all default interrupt is enabled, there will be an all default interrupt.

26.3.7.2 Key Stuck Corner Case

Since auto_check_key_stuck function configures the key stuck status in rows, the following corner cases need to be noted.

- After key stuck occurs, other keys of the same stuck row are pressed or released.
  - In the low power mode, the Key-Scan cannot wake up the system. In this case, the row and the column of the stuck key are connected, so the row of the stuck key will be pulled low. When other keys of the same row are pressed or released, it will not affect the level of the row, so other keys cannot be detected.
  - After Key-Scan wakes up the system, if any other keys in the same stuck row are pressed or released during scanning, they can be detected normally. If the release timer has run out after wakeup and the state machine enters the idle state, the pressed or released key(s) cannot be detected.
- After the key stuck, other keys that are not in the same stuck row are pressed or released.
  - When pressing or releasing a key or keys that are not in the same row, the level of the row with the pressed or released keys will be pulled low/high, so the pressed or released key(s) can be detected.
  - This operation can also wake up the system if in the low power mode.

All information provided in this document is subject to legal disclaimers.
© REALTEK 2024. All rights reserved.

26.3.7.3 Periodic Detection Function of Stuck row

When the key stuck occurs, if the Key-Scan enters the idle state, the output level of the stuck row is low since the row of the stuck key is directly connected to the column. Thus, the I/O port leakage happens from the pull-up resistor.

When the periodic detection function of the stuck row is disabled, the Key-Scan keeps scanning each row in turn, including the stuck row. The power consumption increases because of the leakage.

In order to reduce the leakage time and power consumption, it is necessary to enable the periodic detection function of the stuck row. The software can configure two times: stuckrow_detect_time and stuckrow_detect_time_interval. After the periodic detection function of the stuck row is enabled, the Key-Scan will detect to determine the status of the stuck row every stuckrow_detect_time_interval, with each detection time being stuckrow_detect_time. At this time, the corresponding I/O port of the stuck row is set to no-pull during stuckrow_detect_time_interval, while during the detection period it is set to pull-up. If the system is in low power mode at this moment, the stuck row detection behavior will not wake up the system. In addition, for other rows that are not stuck, it is continuously to detect the I/O port status to ensure response speed.

NOTE

This periodic detection function of the stuck row is only valid when auto_check_key_stuck function is enabled, otherwise it will not take effect.

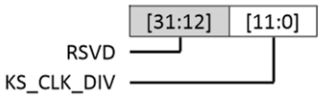
26.4 Registers

Name	Address offset	Access	Description
REG_KS_CLK_DIV	000h	R/W	Clock Division Register
REG_KS_TIM_CFG0	004h	R/W	Key-Scan Configure Register 0
REG_KS_TIM_CFG1	008h	R/W	Key-Scan Configure Register 1
REG_KS_CTRL	00Ch	R/W	Key-Scan Control Register
REG_KS_FIFO_CFG	010h	R/W	FIFO Configuration Register
REG_KS_COL_CFG	014h	R/W	Column Selection Register
REG_KS_ROW_CFG	018h	R/W	Row Selection Register
REG_KS_DATA_NUM	01Ch	R	FIFO Status
REG_KS_DATA	020h	R	Key-Scan Data Register
REG_KS_IMR	024h	R/W	Interrupt Mask Register
REG_KS_ICR	028h	R/W	Interrupt Clear Register
REG_KS_ISR	02Ch	R	Interrupt Status Register
REG_KS_ISR_RAW	030h	R	Raw Interrupt Flag Register
REG_KS_DISCHARGED_CTRL	034h	R/W	Keyboard Discharged Register
REG_KS_STUCK_CFG0	03Ch	R/W	Key-Scan Stuck Configuration Register
REG_KS_STUCK_CFG1	040h	R/W	Key-Scan Stuck Configuration Register 1
REG_KS_STUCK_CFG2	044h	R/W	Key-Scan Stuck Configuration Register 2
REG_KS_STUCK_KEY_STATUS1	048h	R	Key-Scan Status Register 1
REG_KS_STUCK_KEY_STATUS2	04Ch	R	Key-Scan Status Register 2
REG_KS_STUCK_KEY_ROW_STATUS1	050h	R	Key-Scan Row Status Register 1
REG_KS_DUMMY	058h	R/W	Key-Scan Dummy Register

26.4.1 REG_KS_CLK_DIV

- **Name:** Clock Division Register
- **Size:** 32 bits
- **Address offset:** 000h
- **Read/write access:** R/W

This register is for Key-Scan scan clock division.



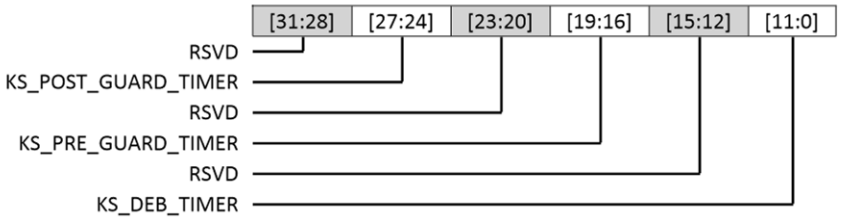
All information provided in this document is subject to legal disclaimers.
© REALTEK 2024. All rights reserved.

Bit	Symbol	Access	Reset	Description
31:12	RSVD	-	0	Reserved
11:0	KS_CLK_DIV	R/W	0x0	scan clock = bus clock/(keyscan_clk_div+1)

26.4.2 REG_KS_TIM_CFG0

- **Name:** Key-Scan Configure Register 0
- **Size:** 32 bits
- **Address offset:** 004h
- **Read/write access:** R/W

This is configuration register for timers.

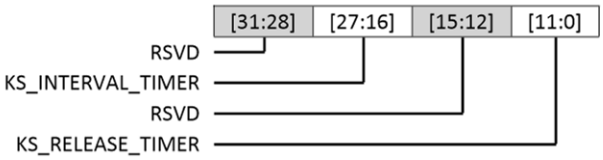


Bit	Symbol	Access	Reset	Description
31:28	RSVD	-	0	Reserved
27:24	KS_POST_GUARD_TIMER	R/W	0x3	Post guard time for key column, post guard time = scan_clk * reg_col_gt_post_sel
23:20	RSVD	-	0	Reserved
19:16	KS_PRE_GUARD_TIMER	R/W	0x3	Pre guard time for key column, pre guard time = scan_clk * reg_col_gt_pre_sel
15:12	RSVD	-	0	Reserved
11:0	KS_DEB_TIMER	R/W	0x0	Debounce counter n*scan_clk

26.4.3 REG_KS_TIM_CFG1

- **Name:** Key-Scan Configure Register 1
- **Size:** 32 bits
- **Address offset:** 008h
- **Read/write access:** R/W

This is configuration register for timers.



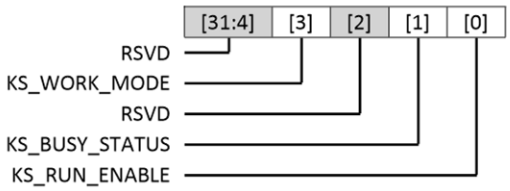
Bit	Symbol	Access	Reset	Description
31:28	RSVD	-	0	Reserved
27:16	KS_INTERVAL_TIMER	R/W	0x0	Scan interval counter n*scan_clk
15:12	RSVD	-	0	Reserved
11:0	KS_RELEASE_TIMER	R/W	0x0	Release detect counter n*scan_clk

26.4.4 REG_KS_CTRL

- **Name:** Key-Scan Control Register
- **Size:** 32 bits
- **Address offset:** 00Ch
- **Read/write access:** R/W

All information provided in this document is subject to legal disclaimers.
© REALTEK 2024. All rights reserved.

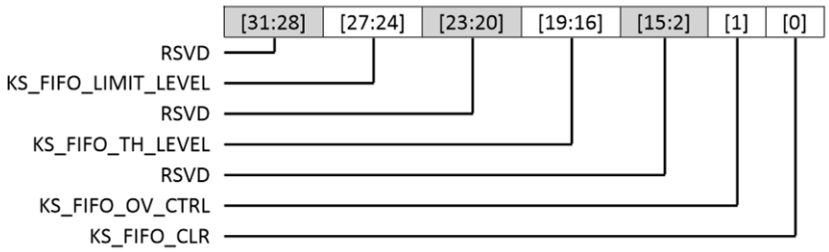
This register controls Key-Scan working mode and enable.



Bit	Symbol	Access	Reset	Description
31:4	RSVD	-	0	Reserved
3	KS_WORK_MODE	R/W	0x0	Work mode 0x1: Event Trigger Mode 0x0: Regular Scan Mode
2	RSVD	-	0	Reserved
1	KS_BUSY_STATUS	R	0x0	FSM busy status
0	KS_RUN_ENABLE	R/W	0x0	Enable internal Key-Scan scan clock. The Key-Scan clock enable must be after Key-Scan configuration is done. 0x1: Key-Scan clock enabled (HW FSM starts to run). 0x0: Key-Scan clock disabled (HW FSM stops).

26.4.5 REG_KS_FIFO_CFG

- **Name:** FIFO Configuration Register
- **Size:** 32 bits
- **Address offset:** 010h
- **Read/write access:** R/W



Bit	Symbol	Access	Reset	Description
31:28	RSVD	-	0	Reserved
27:24	KS_FIFO_LIMIT_LEVEL	R/W	0x0	Limit the max allowable key number be pressed at a time 0x0: no limit 0x1: only one scan data is allowable in each key scan … 0x6: Max 6 scan data is allowable in each key scan 0x7 ~ 0xf: DO NOT USE
23:20	RSVD	-	0	Reserved
19:16	KS_FIFO_TH_LEVEL	R/W	0x0	FIFO threshold setup
15:2	RSVD	-	0	Reserved
1	KS_FIFO_OV_CTRL	R/W	0x0	FIFO overflow control 0x0: rejects the new scan data when FIFO is full 0x1: discard the oldest scan data when FIFO is full
0	KS_FIFO_CLR	WA0	0x0	Write 1to clear FIFO data

26.4.6 REG_KS_COL_CFG

- **Name:** Column Selection Register
- **Size:** 32 bits
- **Address offset:** 014h
- **Read/write access:** R/W

All information provided in this document is subject to legal disclaimers.
© REALTEK 2024. All rights reserved.

This register selects which column is used.



Bit	Symbol	Access	Reset	Description
31:8	RSVD	-	0	Reserved
7:0	KS_COL_SEL	R/W	0x0	The control register to define which column is used. 0x0: No key column selected 0x1: Key column 0 selected 0x2: Key column 1 selected 0x3: Key column 0 and column 1 selected 0x4: Key column 2 selected ...

26.4.7 REG_KS_ROW_CFG

- **Name:** Row Selection Register
- **Size:** 32 bits
- **Address offset:** 018h
- **Read/write access:** R/W

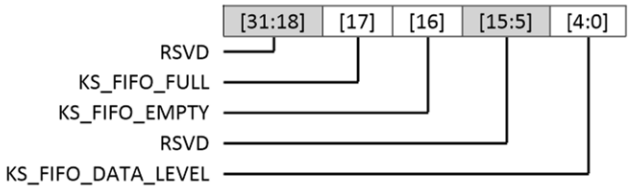
This register selects which row is used.



Bit	Symbol	Access	Reset	Description
31:8	RSVD	-	0	Reserved
7:0	KS_ROW_SEL	R/W	0x0	The control register to define which row is used. 0x0: No key row selected 0x1: Key row 0 selected 0x2: Key row 1 selected 0x3: Key row 0 and row 1 selected 0x4: Key row 2 selected ...

26.4.8 REG_KS_DATA_NUM

- **Name:** FIFO Status
- **Size:** 32 bits
- **Address offset:** 01Ch
- **Read/write access:** R



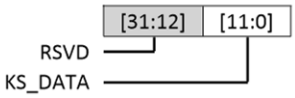
Bit	Symbol	Access	Reset	Description
31:18	RSVD	-	0	Reserved
17	KS_FIFO_FULL	R	0x0	FIFO full status
16	KS_FIFO_EMPTY	R	0x1	FIFO empty status
15:5	RSVD	-	0	Reserved
4:0	KS_FIFO_DATA_LEVEL	R	0x0	Number of entry in FIFO

All information provided in this document is subject to legal disclaimers.
© REALTEK 2024. All rights reserved.

26.4.9 REG_KS_DATA

- **Name:** Key-Scan Data Register
- **Size:** 32 bits
- **Address offset:** 020h
- **Read/write access:** R

User can read scan result from this register, meaning pop one scan result from FIFO.

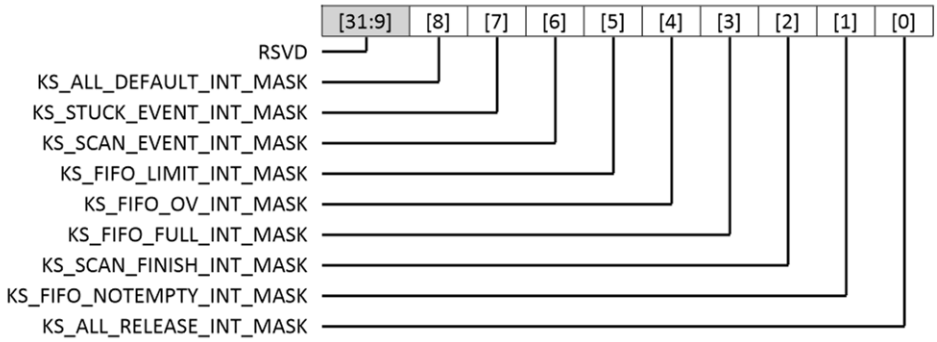


Bit	Symbol	Access	Reset	Description
31:12	RSVD	-	0	Reserved
11:0	KS_DATA	RP	0x0	When reading this register, these bits return the value in the Event FIFO. Bit[11:8]: press or release event 0x0: release event 0x1: press event Bit[7:4]: row index Bit[3:0]: column index

26.4.10 REG_KS_IMR

- **Name:** Interrupt Mask Register
- **Size:** 32 bits
- **Address offset:** 024h
- **Read/write access:** R/W

This register is used to mask unused interrupts.



Bit	Symbol	Access	Reset	Description
31:9	RSVD	-	0	Reserved
8	KS_ALL_DEFAULT_INT_MASK	R/W	0x0	0x0: Mask all default interrupt
7	KS_STUCK_EVENT_INT_MASK	R/W	0x0	0x0: Mask stuck event interrupt
6	KS_SCAN_EVENT_INT_MASK	R/W	0x0	0x0: Mask scan event interrupt
5	KS_FIFO_LIMIT_INT_MASK	R/W	0x0	0x0: Mask FIFO limit interrupt
4	KS_FIFO_OV_INT_MASK	R/W	0x0	0x0: Mask FIFO overflow interrupt
3	KS_FIFO_FULL_INT_MASK	R/W	0x0	0x0: Mask FIFO full interrupt
2	KS_SCAN_FINISH_INT_MASK	R/W	0x0	0x0: Mask scan finish interrupt
1	KS_FIFO_NOTEMPTY_INT_MASK	R/W	0x0	0x0: Mask FIFO nonempty interrupt
0	KS_ALL_RELEASE_INT_MASK	R/W	0x0	0x0: Mask All Release interrupt

All information provided in this document is subject to legal disclaimers.
© REALTEK 2024. All rights reserved.

26.4.11 REG_KS_ICR

- **Name:** Interrupt Clear Register
- **Size:** 32 bits
- **Address offset:** 028h
- **Read/write access:** R/W

This register is used to clear interrupts.

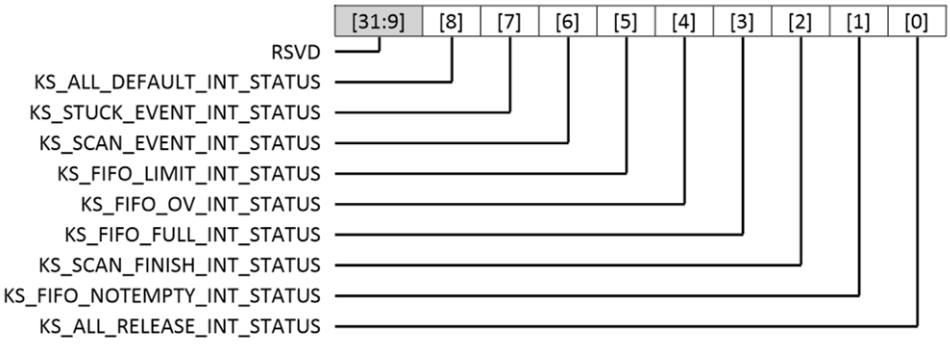


Bit	Symbol	Access	Reset	Description
31:9	RSVD	-	0	Reserved
8	KS_ALL_DEFAULT_INT_CLR	WA0	0x0	Clear All Default interrupt flag Write '1' to clear
7	KS_STUCK_INT_CLR	WA0	0x0	Clear Stuck interrupt flag Write '1' to clear
6	RSVD	-	0	Reserved
5	KS_FIFO_LIMIT_INT_CLR	WA0	0x0	Clear FIFO limit interrupt flag Write '1' to clear
4	KS_FIFO_OV_INT_CLR	WA0	0x0	Clear FIFO overflow interrupt flag Write '1' to clear
3	RSVD	-	0	Reserved
2	KS_SCAN_FINISH_INT_CLR	WA0	0x0	Clear scan finish interrupt flag Write '1' to clear
1	RSVD	-	0	Reserved
0	KS_ALL_RELEASE_INT_CLR	WA0	0x0	Clear all release interrupt flag Write '1' to clear

26.4.12 REG_KS_ISR

- **Name:** Interrupt Status Register
- **Size:** 32 bits
- **Address offset:** 02Ch
- **Read/write access:** R

This register is about interrupt flags.



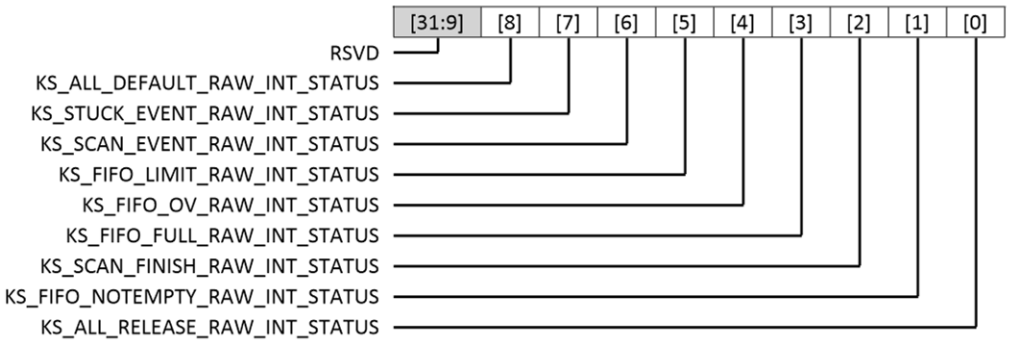
All information provided in this document is subject to legal disclaimers.
© REALTEK 2024. All rights reserved.

Bit	Symbol	Access	Reset	Description
31:9	RSVD	-	0	Reserved
8	KS_ALL_DEFAULT_INT_STATUS	R	0x0	Masked all default interrupt status, means row status equal to ks_row_default_status
7	KS_STUCK_EVENT_INT_STATUS	R	0x0	Masked Stuck event interrupt status
6	KS_SCAN_EVENT_INT_STATUS	R	0x0	Masked Scan event interrupt status
5	KS_FIFO_LIMIT_INT_STATUS	R	0x0	Masked FIFO limit interrupt status
4	KS_FIFO_OV_INT_STATUS	R	0x0	Masked FIFO overflow interrupt status
3	KS_FIFO_FULL_INT_STATUS	R	0x0	Masked FIFO full interrupt status
2	KS_SCAN_FINISH_INT_STATUS	R	0x0	Masked Scan finish interrupt status
1	KS_FIFO_NOTEMPTY_INT_STATUS	R	0x0	Masked FIFO nonempty interrupt status
0	KS_ALL_RELEASE_INT_STATUS	R	0x0	Masked All release interrupt status

26.4.13 REG_KS_ISR_RAW

- **Name:** Raw Interrupt Flag Register
- **Size:** 32 bits
- **Address offset:** 030h
- **Read/write access:** R

The raw interupt flags do not effected by interrupt mask register.

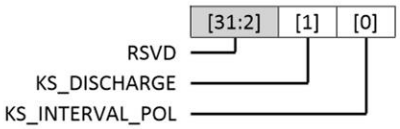


Bit	Symbol	Access	Reset	Description
31:9	RSVD	-	0	Reserved
8	KS_ALL_DEFAULT_RAW_INT_STATUS	R	0x0	All default interrupt, means row status equal to ks_row_default_status
7	KS_STUCK_EVENT_RAW_INT_STATUS	R	0x0	Key stuck event raw interrupt
6	KS_SCAN_EVENT_RAW_INT_STATUS	R	0x0	Scan event raw interrupt, auto clear when the item is read
5	KS_FIFO_LIMIT_RAW_INT_STATUS	R	0x0	FIFO limit raw interrupt
4	KS_FIFO_OV_RAW_INT_STATUS	R	0x0	FIFO overflow raw interrupt
3	KS_FIFO_FULL_RAW_INT_STATUS	R	0x0	FIFO full raw interrupt
2	KS_SCAN_FINISH_RAW_INT_STATUS	R	0x0	Scan finish raw interrupt
1	KS_FIFO_NOTEMPTY_RAW_INT_STATUS	R	0x0	FIFO nonempty raw interrupt Auto clear when FIFO is empty
0	KS_ALL_RELEASE_RAW_INT_STATUS	R	0x0	All Release raw interrupt

26.4.14 REG_KS_DISCHARGED_CTRL

- **Name:** Keyboard Discharged Register
- **Size:** 32 bits
- **Address offset:** 034h
- **Read/write access:** R/W

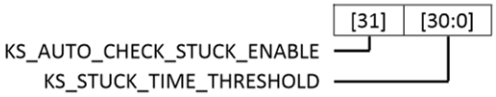
All information provided in this document is subject to legal disclaimers.
© REALTEK 2024. All rights reserved.



Bit	Symbol	Access	Reset	Description
31:2	RSVD	-	0	Reserved
1	KS_DISCHARGE	R/W	0x0	1: The keyboard is discharged before scan.
0	KS_INTERVAL_POL	R/W	0x0	1: The keyboard is discharged at interval (FSM!=scan idle allrelease).

26.4.15 REG_KS_STUCK_CFG0

- **Name:** Key-Scan Stuck Configuration Register
- **Size:** 32 bits
- **Address offset:** 03Ch
- **Read/write access:** R/W



Bit	Symbol	Access	Reset	Description
31	KS_AUTO_CHECK_STUCK_ENABLE	R/W	0x0	Enable auto check stuck or not 0x1: Key-Scan auto chek stuck enabled. 0x0: Key-Scan auto check stuck disabled.
30:0	KS_STUCK_TIME_THRESHOLD	R/W	0x7FFFFFFF	When the threshold is exceeded, it is considered to be stuck event or release stuck key, this counter is in scan clk.

26.4.16 REG_KS_STUCK_CFG1

- **Name:** Key-Scan Stuck Configuration Register 1
- **Size:** 32 bits
- **Address offset:** 040h
- **Read/write access:** R/W

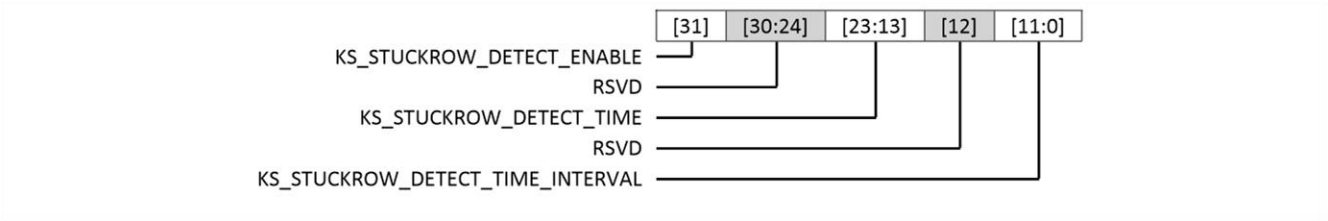


Bit	Symbol	Access	Reset	Description
31:8	RSVD	-	0	Reserved
7:0	KS_ROW_DEFAULT_STATUS	R/W	0xFF	Keyboard row default status: 0b11111111: No key row is stucked 0b11111110: Key row 0 is stucked 0b11111101: Key row 1 is stucked 0b11111100: Key row 0 and column 1 are stucked 0b11111011: Key row 2 is stucked …

26.4.17 REG_KS_STUCK_CFG2

- **Name:** Key-Scan Stuck Configuration Register 2
- **Size:** 32 bits
- **Address offset:** 044h
- **Read/write access:** R/W

All information provided in this document is subject to legal disclaimers.
© REALTEK 2024. All rights reserved.



Bit	Symbol	Access	Reset	Description
31	KS_STUCKROW_DETECT_ENABLE	R/W	0x0	Enable stuckrow detect function, and row gpio will be "no pull" state in ks_stuckrow_detect_time_interval
30:24	RSVD	-	0	Reserved
23:13	KS_STUCKROW_DETECT_TIME	R/W	0x1	Key-Scan stuckrow detect timer, this counter is in clk_131k
12	RSVD	-	0	Reserved
11:0	KS_STUCKROW_DETECT_TIME_INTERVAL	R/W	0x0	Key-Scan stuckrow detect interval timer, this counter is in clk_131k

26.4.18 REG_KS_STUCK_KEY_STATUS1

- **Name:** Key-Scan Status Register 1
- **Size:** 32 bits
- **Address offset:** 048h
- **Read/write access:** R



Bit	Symbol	Access	Reset	Description
31:0	KS_KEY_STATUS1	R	0x0	Each bit means the corresponding key's stuck state 0x1: key stucked 0x0: key unstucked Corresponding: key_status1[7:0] - row0[7:0] key_status1[15:8] - row1[7:0] key_status1[13:16] - row2[7:0] key_status1[31:24] - row3[7:0]

26.4.19 REG_KS_STUCK_KEY_STATUS2

- **Name:** Key-Scan Status Register 2
- **Size:** 32 bits
- **Address offset:** 04Ch
- **Read/write access:** R

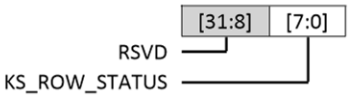


Bit	Symbol	Access	Reset	Description
31:0	KS_KEY_STATUS2	R	0x0	Each bit means the corresponding key's stuck state 0x1: key stucked 0x0: key unstucked Corresponding: key_status2[7:0] - row4[7:0] key_status2[15:8] - row5[7:0] key_status2[13:16] - row6[7:0] key_status2[31:24] - row7[7:0]

All information provided in this document is subject to legal disclaimers.
© REALTEK 2024. All rights reserved.

26.4.20 REG_KS_STUCK_KEY_ROW_STATUS1

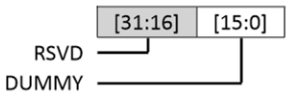
- **Name:** Key-Scan Row Status Register 1
- **Size:** 32 bits
- **Address offset:** 050h
- **Read/write access:** R



Bit	Symbol	Access	Reset	Description
31:8	RSVD	-	0	Reserved
7:0	KS_ROW_STATUS	R	0x0	Each bit means the corresponding key row's stuck state 0x1: row stucked 0x0: row unstucked Corresponding: row_status[7:0] - row[7:0]

26.4.21 REG_KS_DUMMY

- **Name:** Key-Scan Dummy Register
- **Size:** 32 bits
- **Address offset:** 058h
- **Read/write access:** R/W



Bit	Symbol	Access	Reset	Description
31:16	RSVD	-	0	Reserved
15:0	DUMMY	R/W	0x0	Reserved for HW

All information provided in this document is subject to legal disclaimers.
© REALTEK 2024. All rights reserved.

27 Cap-Touch Controller (CTC)

27.1 Introduction

Self-capacitance touch controller measures the capacitance between the capacitive sensor pin and ground. The capacitive touch controller detects the presence of a finger on or near a touch surface through capacitance changes.

The sensor parasitic capacitance is called C_P. When fingers touch or approach the sensor's surface, it forms a simple parallel plate capacitor with the sensor pad through the overlay, which is called the finger capacitance C_F. It increases the measured capacitance C_S:

$$C_S = C_P + C_F$$

In the initial state, an amount of charge is built-up on an untouched sensor to set the sensor's reference level, C_S=C_P. As the finger makes contact with the sensor, it couples with the electric field of the sensor. This results in a drop of the sensor's charge, because a portion of the charge is drained off through the finger to ground. Further charge-transfer results in a reduced signal level compared to the untouched state.

As *Figure 27-1* shows, Cap-Touch controller (CTC) is implemented by ADC oversampling in each channel. The high sensitive ADC captures the reference voltage (baseline) and the signal voltage, and transforms them into digital raw data. If the difference between the reference level and the signal level is greater than the user-defined finger touch threshold, a touch is detected. At the same time, a touch interrupt is sent to the MCU to wake up system.

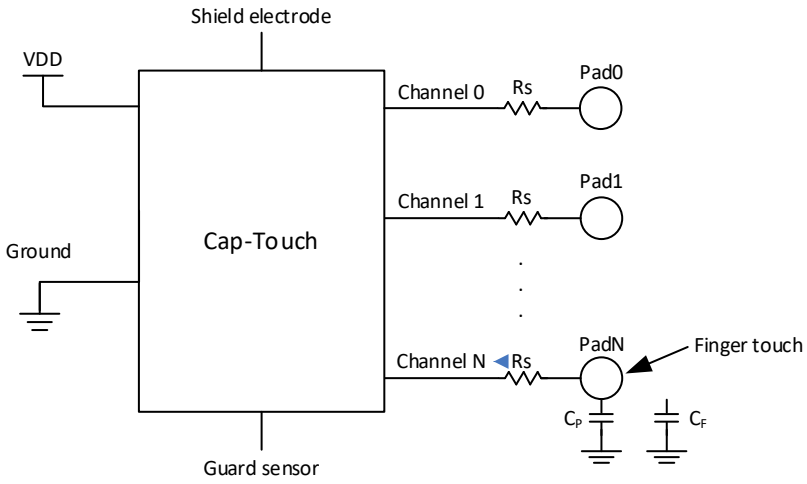


Figure 27-1 Cap-Touch system

The CTC provides up to 4 channels for capacitive sensing, which offers a wide range of capacitance detection. It can adjust the sensitivity and threshold of each channel according to different noise environment. It also can calibrate the baseline and absolute finger touch threshold automatically when environment changes. At the same time, the CTC offers an adjustable scan period to achieve lower power consumption.

27.2 Features

- 4 capacitive sensor channels:
  - Detection of finger touch
  - Programmable enable/disable for each channel
  - Adjustable sensitivity for each channel: 0.25 ~ 63.75uA
- Adjustable baseline and touch threshold for each channel
- Automatic channel scan: hardware scans each enabled channel automatically in sequence
- Programmable scan period: sample number and scan interval:
- Configurable sample clock, adapting to filter function (max. scan clock = 1MHz)
- Active noise immunity:
  - Support SNR information monitor
  - Adjustable environmental noise threshold for each channel
- Automatic environment tracking and calibration (ETC)
  - Automatic hardware baseline initialization
  - Automatic baseline and threshold update for different noise environment
  - Programmable ETC update step and factor
- Programmable button debounce function

All information provided in this document is subject to legal disclaimers.
© REALTEK 2024. All rights reserved.

- Interrupt control:
  - Programmable interrupt enabled for each interrupt source
  - Software readable interrupt status and raw status register
- Low power consumption

27.3 Architecture

27.3.1 CTC Block Diagram

A typical capacitive interface system consists of the following modules:
- Capacitive sensors
- A capacitive sensing controller connecting to the sensors
- A host processor/MCU

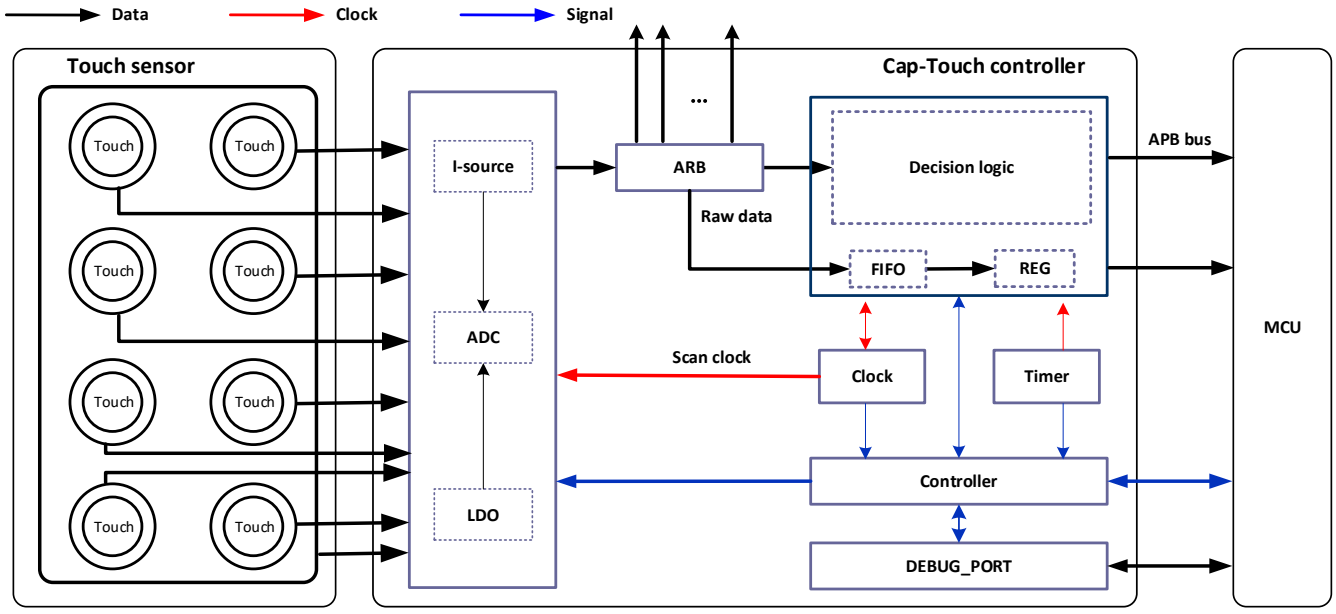


Figure 27-2 CTC block diagram

The CTC connects the sensors and the host processor through interface. The sensing algorithm, built in the controller's decision logic module, determines the presence of touch or proximity. It drives the outputs or sends interrupt signals to the MCU. This algorithm can distinguish between the effective signals and the noises caused by environmental or electrical conditions.

27.3.2 Capacitive Sensors

Capacitive sensors block consists of the following sensors:
- Capacitance/environmental element change detection sensor
- Finger on or approach detection sensor

27.3.3 Cap-Touch Controller

The CTC is the main block of the Cap-Touch system. It consists of several modules, as *Table 27-1* shows.

Table 27-1 CTC modules

Module	Description
ADC	● Captures the change of sensor's capacitance according to the environmental variation ● Captures the change of sensor's impedance according to finger on or approach ● Data process: transfer sensor's capacitance into digital raw data ● Error detection and automatic reset
ARB (Arbitration)	● ADC enables FSM ● Arbitrates CTC and other external functions using ADC

All information provided in this document is subject to legal disclaimers.
© REALTEK 2024. All rights reserved.

FIFO	Size is 16-bit*64
REG	● Contains configuration registers, sensor status, data information ● The interface to software ● Generates and manages the interrupt and interrupt flags
Decision logic	● Register module of noise threshold, finger touch threshold, baseline and signal's raw data, difference value between baseline and signal, and other control register ● Receives and analyses raw data from ADC ● Judgement of single-end mode based on control register， and select different judge arithmetic ● Determines whether there is a touch or not based on raw data analysis and touch arithmetic ● Determines whether there is a proximity or not based on raw data analysis and proximity arithmetic ● Automatically records and updates the baseline based on ETC algorithm ● Automatically records and updates the absolute finger touch threshold ● Sends touch interrupt to MCU
Controller	● Main control FSM of CTC ● Channel control: enable/disable for each channel ● Sensitivity setting for each channel ● Automatic channel switch, channel setting ● Manual setting of baseline, noise threshold and absolute touch threshold for each channel ● ETC function enable/disable and ETC configurable scan interval ● Configurable Channel mode: single-end mode ● Configurable scan period: sample number, scan interval setting ● Scan control: start/stop ● Sets up timer ● Receives and executes command from MCU
Clock	Scan source clock of capturing ADC
Timer	● Counter for scan interval ● Counter for ETC, debounce
LDO	Independent power supply for ADC

27.3.4 CTC Clock Divider

ADC/CTC/Comp share the same clock source: adcc_clk - XTAL4M (better jitter) or OSC4M

Cap-Touch sample clock (fs): max = 1MHz, min = 31.25KHz, default = 125KHz

Figure 27-3 illustrates the clock or timer used in the system.

- 32.768kHz
  - Clock source
  - Scan interval timer: 1.024kHz (32.768kHz/32)
- OSC or XTAL2M divider
  - Scan clock: scan clock for ADC
  - Core clock: the main clock for decision logic, controller, FIFO

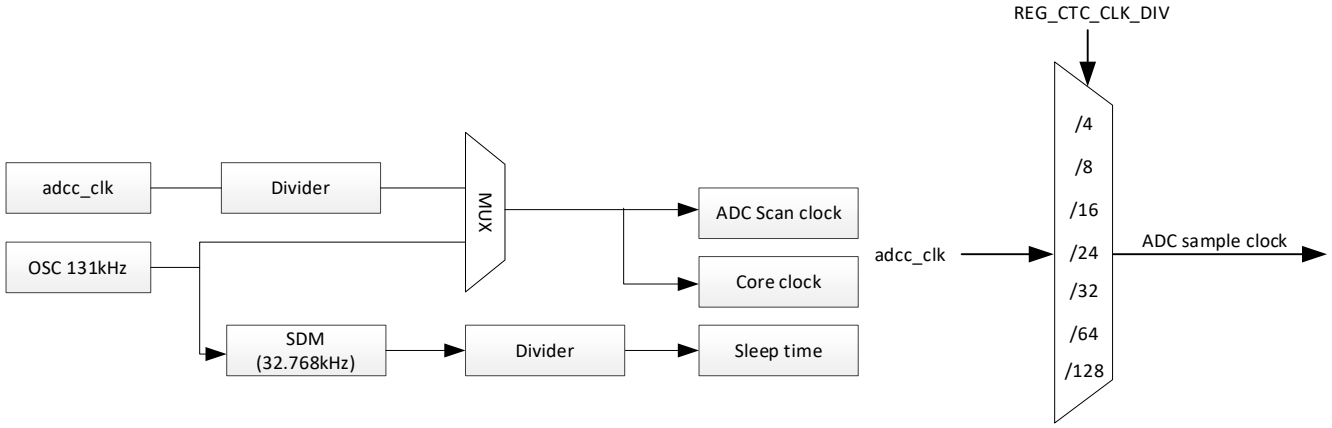


Figure 27-3 Clock and timer diagram

All information provided in this document is subject to legal disclaimers.
© REALTEK 2024. All rights reserved.

27.4 Functional Description

27.4.1 Sample Range

The CTC sample range is related to channel sensitivity. The more sensitive the channel is, the smaller the sample range is.

The sensitivity parameter is used to determine the strength of the sensor signal, which is related to difference count for the same capacitance change. A higher sensitivity (bias current) leads to a stronger signal from the sensors (more difference count for the same capacitance change), and also decreases the sample range. The more details of sensitivity can be found in *Sensitivity*. The typical sample range of CTC is 5 ~ 45pF when channel sensitivity is 5uA and sample clock is 131kHz.

27.4.2 Debug Mode

When the CTC is configured in debug mode, hardware only scans the channel specified by software once at one scan period, ignoring the channel x enable register. For example, if channel 2 is specified by software, then hardware only scans channel 2 once at every scan period. This channel scan mode is suggested to use only in debug mode.

Debug mode usually works during debug process or development. Controller switches channels manually by configuring the channel control registers. After ADC scanning and data transforming finished, all of the sample data during one scan period in FIFO is transformed into the AFIFO_RD_DATA bit for debug requirement. When packet is new in register space, AFIFO_RD_DATA_VLD = 1, after software taking the packet, it is cleared. Other registers such as CTC_CTRL, CTC_CHANNELx_DATA_INF also can be dumped for debug purpose.

Debug mode provides another way for software to realize Cap-Touch function (touch/release detection, dynamic baseline update…). In this mode, software get each channel raw data and do filtering to adapt to various applications. For example, mean filter for periodic noise (50Hz or 60Hz), IIR filter for high frequency white noise, and event-based filter to avoid false trigger caused by interference.

To better software development, more features are proposed:

- Single channel continuous sample: set bit continue_sample_en as 1 to enable continuous sample mode, if multi channels are enabled, it will choose the channel with smallest channel number to do continuous sampling.
- FIFO depth: 64 x 16 bits.
- FIFO full level judge: set FIFO full level (afifo_full_lvl) and enable FIFO full interrupt (afifo_overlvl_intr_en), when the number of data in FIFO exceeds the FIFO level, FIFO full level interrupt will be triggered.
- Add channel number flag to distinguish different channels' sample data.
- FIFO clear bit: SW write 1 (afifo_clear) to clear the data in FIFO, after that digital circuit automatically clear this bit to 0.

27.4.3 Auto-scan

The system support 4 Cap-Touch sensor channels, all of the 4 channels can be enabled or disabled by register CHx_CTRL.

Hardware scans each enabled channel in sequence during one scan period. For example, if channel 2 is disabled, hardware just scans channel 0 → channel 1 → channel 3 once in each scan period.

All information provided in this document is subject to legal disclaimers.
© REALTEK 2024. All rights reserved.

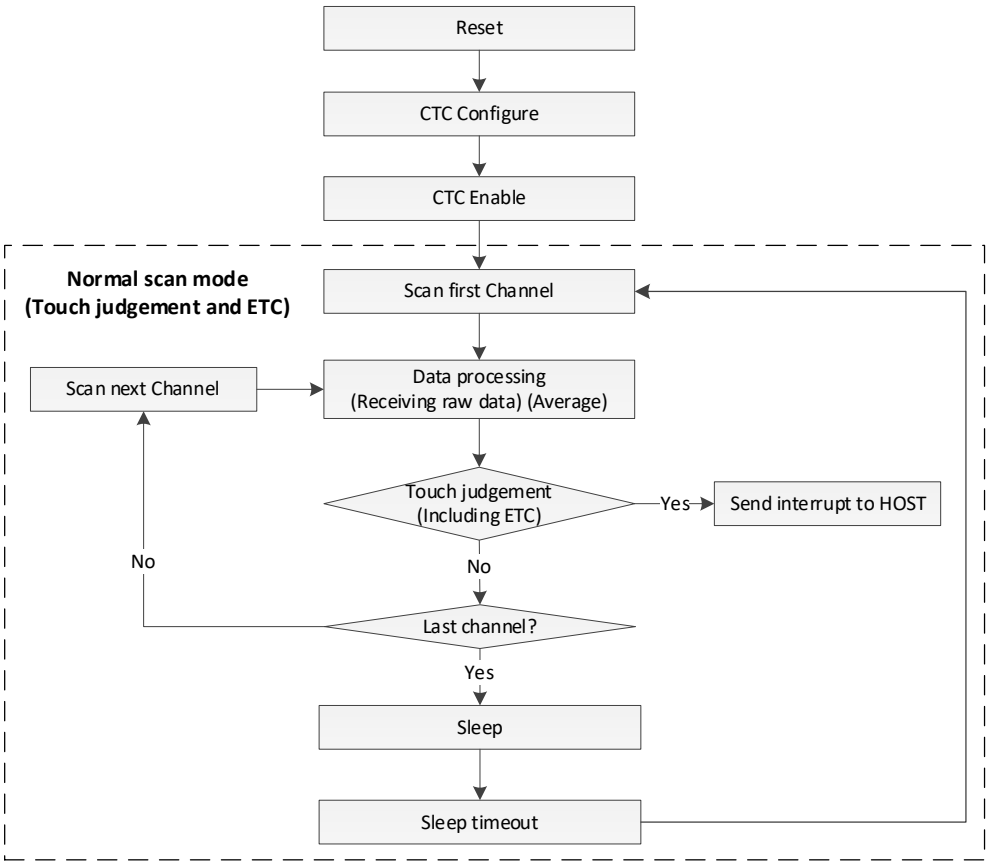


Figure 27-4 Auto-scan mode flow

27.4.4 Scan Period

The CTC scan period consists of active time and sleep time is shown in *Figure 27-5*. The active time and sleep time can be configured individually for different applications.

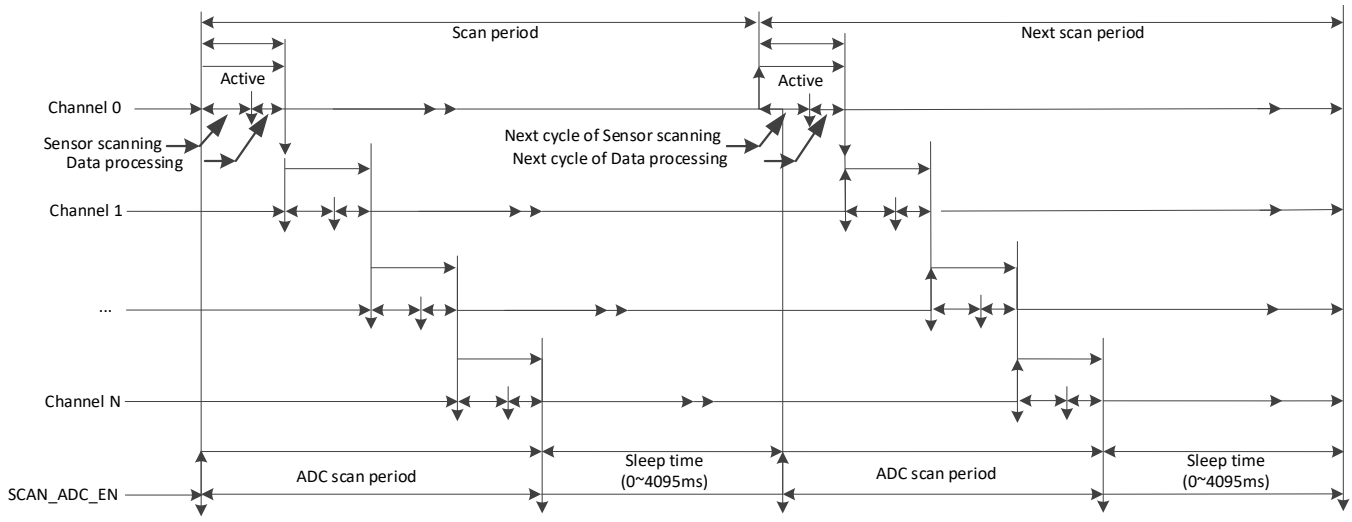


Figure 27-5 Scan period diagram

NOTE

- *The analog circuit is sampled continuously in auto-scan mode; however, when the digital circuit actually fetches data, there will be a period of 5 scan clocks (sample clocks) between every two channels due to state machine switching. If in debug mode at this time, 5 data packets pushed to the FIFO will be lost.*
- *The sleep time is configurable in the RTL8721Dx. Code: 0~4095 (0~0xFFF), unit is 1.024kHz cycle (1/32 32.768kHz). When the sleep time equals 0, the circuit will scan continuously without sleep time.*

All information provided in this document is subject to legal disclaimers.
© REALTEK 2024. All rights reserved.

27.4.4.1 Active Time

The active time is defined by ADC sample number and sample frequency. The sample frequency's clock source can be from OSC4MHz or XTAL4M, sample frequency can be flexibly configured, and the max sample frequency is up to 1MHz. The sample number is 2^N which ranges from 4 to 512.

The first raw data sampled by ADC after channel switch is unpredictable. As a result, ADC is required by CTC to capture one more raw data and discard the first one.

During the active period, the Cap-Touch system does the following operations.

- Capturing capacitance change and convert it to digital raw data by ADC. The raw data number is determined by sample number.
- Debug mode:
  - Sending the digital raw data into async_fifo for software read
  - Switching to next channel and scanning when operations finished
- Normal mode (non-debug mode):
  - Average raw data for better accuracy and noise immunity
  - Making judgements of finger touch (Touch criteria is different when ETC function is enabled or disabled)
  - Making judgements of baseline and touch threshold update (when ETC is enabled)
  - Switching to next channel and scanning

27.4.4.2 Sleep Time

After the sensor scanning and data processing, the channel enters sleep and another channel begins to capture. After all channels scanning and data processing finished, the ADC is power-off and the Cap-Touch system enters power save state until next scan period.

The sleep time can be configured through the SLEEP_TIME_CNT bit for different noise conditions and power consumption requirements. The value ranges from 0 to 4095ms at 1.024kHz clock. The recommended sleep time is 60~480ms for better response time.

When scan interval is configured to 0 ms, hardware scans continuously, but there are still about 400 us circuit power on time.

27.4.4.3 Baseline Auto-initialization

The initial value of baseline is 0x0 when CTC function is enabled. The CTC initializes baseline value automatically with ETC, and baseline initial functions are both enabled. Hardware sets the baseline with the average of ADC raw code. After that, hardware enters normal scan mode (ETC scan and judge operation). Baseline auto-initialize function is enabled by BASELINE_INI in the CTC_CTRL register.

All information provided in this document is subject to legal disclaimers.
© REALTEK 2024. All rights reserved.

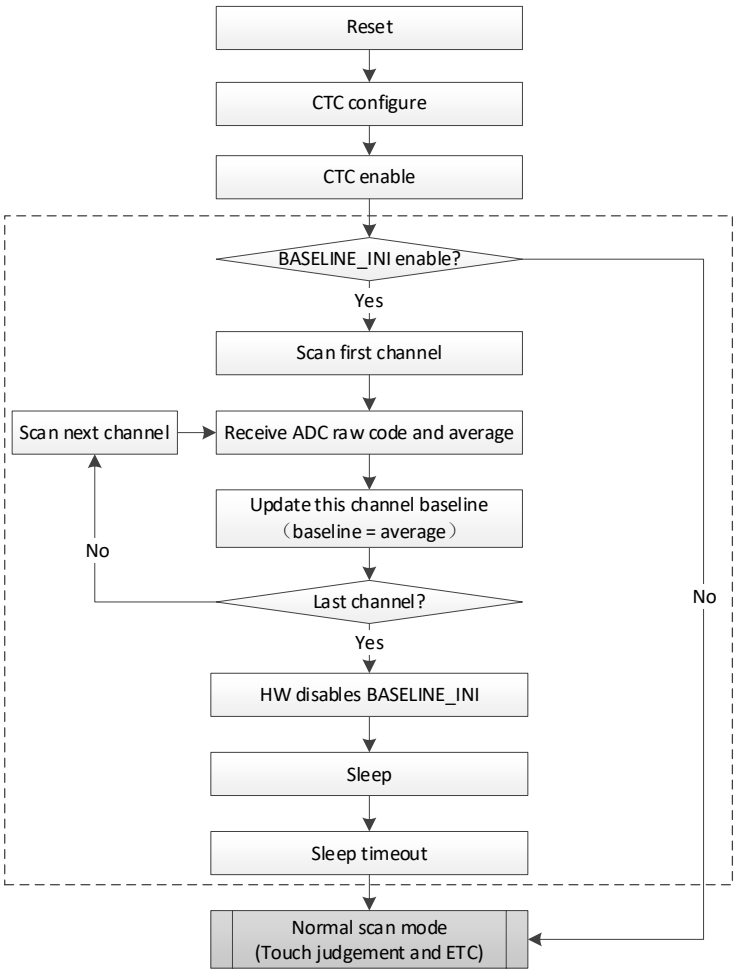


Figure 27-6 Baseline auto-initialization

27.4.5 Sensitivity

The CTC supports configurable sensitivity for each channel. The sensitivity parameter is used to increase or decrease the strength of the sensor signal (difference count). A higher value of sensitivity (bias current) setting leads to a stronger signal from the sensors (more difference count for the same capacitance change), but also increases the response time and average power consumption.

The sensitivity parameter can be set by the CTC_CHANNELx_MBIAS register, and the sensitivity available values for bit[7:0] are 32uA/16uA/8uA/4uA/2uA/1uA/0.5uA/0.25uA. The sensitivity parameter of each sensor can be configured individually for different noise environment, different sensor materials or different applications, such as:

- Different overlay thickness: Thicker overlay requires high value of sensitivity setting.
- Different sensor type: A relatively large sensor size requires lower value of sensitivity. Sensors with smaller size require higher value of sensitivity.
- Different applications: Proximity application requires higher value of sensitivity. Touch button detection application requires lower value of sensitivity.
- System noise: As system noise increases, sensitivity needs to be lower to avoid false trigger of sensors.
- Power consumption: Power consumption increases for lower values of sensitivity setting. For low power consumption, the sensitivity parameter value is low. Power consumption decreases in the order: 32uA > 16uA > 8uA > 4uA > 2uA > 1uA > 0.5uA > 0.25uA.

27.4.6 Touch Judgement

The CTC supports two touch judgement modes: difference threshold judgement and absolute threshold judgement. The flow of the two touch judgements is shown in *Figure 27-7*.

All information provided in this document is subject to legal disclaimers.
© REALTEK 2024. All rights reserved.

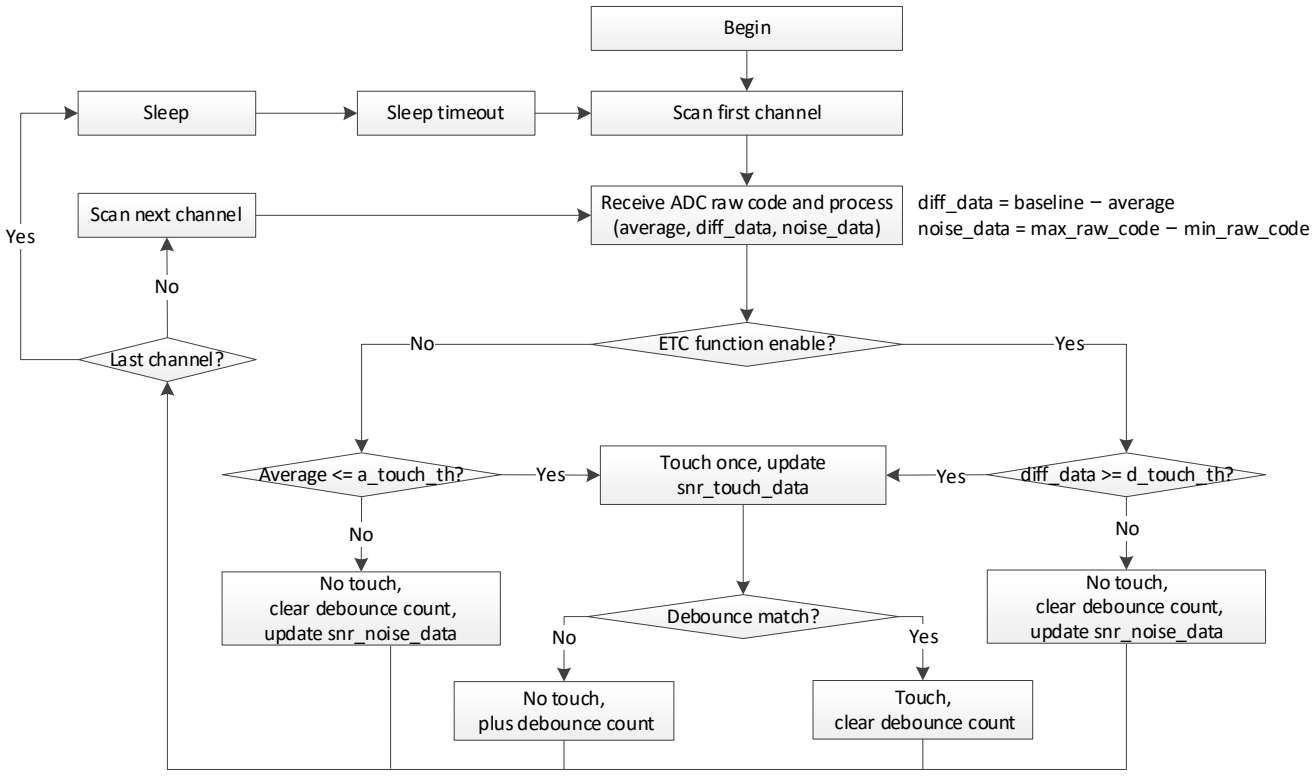


Figure 27-7 Touch judgement flow

27.4.6.1 Difference Threshold Judgement

The mode of difference threshold judgement needs to enable ETC function. After ADC data processing, decision logic does the following operations:

- Averages digital raw data (CHx_DATA_AVE)
- Gets the difference value between the average and baseline (CHx_DIF_DATA)
- Compares the difference data with the difference threshold (D_TOUCH_THRESHOLDx)

If the difference data is larger than difference threshold, touch or proximity is detected.

27.4.6.2 Absolute Threshold Judgement

Absolute threshold judgement mode operates when ETC function is disabled. After ADC data processing, decision logic do the following operations:

- Average digital raw data (CHx_DATA_AVE)
- Compare the average sample data with the absolute threshold (A_TOUCH_THRESHOLDx)

If the average data is less than the absolute threshold, touch or proximity is detected.

27.4.7 Active Noise Immunity

Noise immunity simply refers system's immunity to unwanted noisy voltages and currents, such as slowly and continuous environmental change, temperature change, moisture or liquid etc. The sources of unwanted noise are as follows:

- Environmental slowly and continuous variation
- Temperature variation
- Moisture or liquid
- RF transmitters
- Switch mode power supplies
- Supply voltage (VDD) fluctuations
- Load switching
- EMC:

All information provided in this document is subject to legal disclaimers.
© REALTEK 2024. All rights reserved.

- Electrostatic discharge (ESD)
- Electrical fast transients (EFT)
- Surge voltage (KV)
- Conducted disturbances

To eliminate the effects of unwanted noise disturbances, it is important to know the environment to which the CTC is applied. A well-tuned Cap-Touch system reliably discriminates between touch and no-touch states. To achieve a robust level of performance, the Cap-Touch signal must be significantly larger than the noise. The Cap-Touch signal is compared to the noise by signal-to-noise ratio (SNR). It is required to have a minimum SNR of 5:1 for all sensors. The SNR of sensor is given as follows:

$$\mathrm{SNR} = \frac{Cap - Touch\ \ signal}{peak - to - peak\ \ Noise}$$

Where:

- Cap-Touch signal = measured impedance raw data – calibrated impedance raw data
- Peak-to-peak noise = peak-to-peak noise of raw data measured in the last 5 scan periods

Therefore, the key to achieve a high SNR is to increase the signal and decrease the noise. The noise immunity of a system can be increased significantly by complying with the following methods.

27.4.7.1 SNR Monitor

SNR monitor operates effectively during project development or MP. To use SNR monitor function, you should firstly define a raw data of touch (80% * (signal − baseline)) and save it into SNR_TOUCH_DATA.

$$\mathrm{Cap} - \mathrm{Touch}\ \ \mathrm{signal} = \frac{finger\ \ touch\ \ threshold\ \ (difference\ \ value)}{80\%}$$

Once a real finger touch is detected, the Cap-Touch signal (SNR_TOUCH_DATA) is refreshed by the real finger touch signal average data of the scan period.

Controller captures the difference raw data of peak-to-peak noise signal of each channel during one scan period, and saves the data into SNR_NOISE_DATA.

$$\mathrm{Peak} - \mathrm{to} - \mathrm{peak}\ \ \mathrm{noise} = \max\ \ (noise\ \ signal) - \min\ \ (noise\ \ signal)$$

Then, Cap-Touch controller calculates the SNR during one scan period of each channel. Once the SNR is less than 5, users have to find the fault channel and make some operation to increase the SNR or other operation during debug or development.

SNR information has 2 update modes:

- Updates both of SNR_NOISE_DATA and SNR_TOUCH_DATA no matter detecting a touch or not
- Updates SNR_NOISE_DATA and SNR_TOUCH_DATA based on touch operation

All the register of SNR calculation is located in CTC_SNR_INF register.

During each scan period, the CTC updates the SNR_TOUCH_DATA when touch is detected, and captures the raw data difference of peak-to-peak noise of each channel to calculate SNR value.

27.4.7.2 Average

The CTC averages the sample raw data of capture ADC for best accuracy and noise immunity. Because of more cost and slower calculation speed for division module, the average module uses shifting method for sample data average. The sample number for average is 2^{N+2} (N = 0 ~ 7), and the max average sample number of raw data is 512 (2^9).

You can decide how many sample numbers they need by modify the SAMPLE_AVE_CTRL bit. But the channel current is unstable and un-settled. To avoid the confusion of the first sample data of ADC after channel switching, for better accuracy, when sample number is set to 2^{N+2}, CTC must ask ADC to capture one more raw data and discard the first raw code. Thus the real sample number for average is 2^{N+2}+1, for the first sample data is dropped by hardware automatically.

Based on the shift theory, both the capture ADC and judgement logic are driven by the same source clock. The delta time of ADC capture and data process of judgement logic is less than 1 clock cycle, therefore, there is no need of 512*12 FIFO, the depth of FIFO is 64, and it is enough for average operation.

All information provided in this document is subject to legal disclaimers.
© REALTEK 2024. All rights reserved.

27.4.7.3 Independent Internal Voltage Regulator

ADC is a sensitive module to voltage variation, and Cap-Touch system uses an independent voltage regulator for ADC module. It uses LDO as voltage regulator to get better noise immunity of supply voltage (VDD) fluctuations and load switching.

27.4.8 Environment Tracking and Calibration (ETC)

Sensor capacitance changes as the surrounding environment changes, such as temperature, moisture, overlay, or other slight environmental noise. ETC module is designed to track environmental changes and calibrate with environmental variation.

The ETC system tracks the capacitance change and updates the baseline and touch threshold based on the ETC algorithm periodically. The ETC scan interval can be configured by ETC_SCAN_INTERVAL. Users can decide how many intervals of scan periods they need to update baseline in order to save power consumption.

Manual tuning of ETC parameter is also supported to meet different noise environment or special applications.

27.4.8.1 Noise Threshold

The noise threshold indicates the raw data of the maximum capacitance change caused by the environmental change. The CTC system provides configurable positive noise threshold and negative noise threshold for optimal calibration. Both of the two noise thresholds are individually adjustable for each channel.

Environmental noise usually changes slightly and slowly, and most of the noise average data ranges from 0 ~ 50, less than 40% * touch threshold even when Wi-Fi, BT, GSM or other RF transmission occurs beside the CTC device. So the recommended value of noise threshold is 40% * touch threshold.

The details of Cap-Touch noise threshold are illustrated in *Figure 27-8* and *Table 27-2*. Users need tune the two noise thresholds for different applications and noise environment.

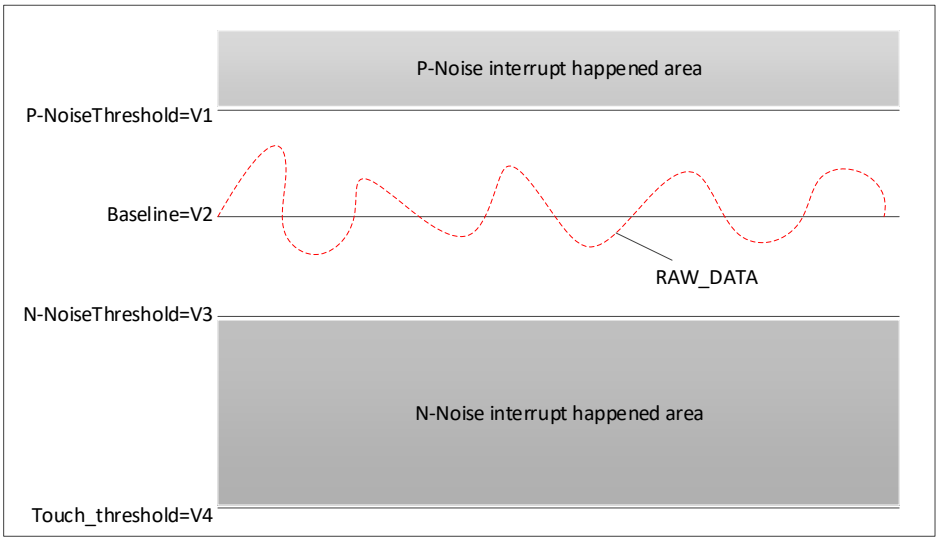


Figure 27-8 Cap-Touch noise threshold

Table 27-2 Cap-Touch noise threshold

Threshold Type	Description
P-Noise	● CTC sends OVER_P_NOISE_THRESHOLD_INT interrupt to host while the difference (RAW_DATA – baseline) is larger than P-Noise Threshold. ● ETC module ignores ETC update operation automatically when OVER_P_NOISE_THRESHOLD_INT happens ● In this case, ETC cannot tune baseline to a correct value, so drivers have to re-initialize baseline through the CTC_CTRL.BASELINE_INI bit.
N-Noise	● CTC sends OVER_N_NOISE_THRESHOLD_INT interrupt to host while the difference (baseline – RAW_DATA) is larger than N-Noise Threshold. ● OVER_N_NOISE_THRESHOLD_INT is ignored when touch press is detected.

All information provided in this document is subject to legal disclaimers.
© REALTEK 2024. All rights reserved.

27.4.8.2 ETC Update Principle

The CTC captures the average data of first scan period as channel baseline when power on no matter ETC function is enabled or disabled.

The ETC update principle applies to the difference which is in the range of noise threshold. The judgement logic module uses weighting factor ($\text{Factor} = 2^{BASELINE_WT_FACTOR}$, defined in CTC_ ETC_CTRL) as baseline update principle. For each ETC scan period, if the noise difference (baseline – signal) is larger than Factor, the baseline increases. If the noise difference (baseline – signal) is less than Factor, the baseline decreases. The baseline update step can be configured in BASELINE_UPD_STEP. If the difference is between the range of Factor and —Factor, the baseline isn't updated at this period. The current difference is accumulated until the difference is larger than Factor or less than Factor.

In some abnormal conditions, such as glass or plastic removed from sensor surface or finger release after CTC initialization, the difference data (signal − baseline) is larger than the positive noise threshold. In these cases, CTC sends OVER_P_NOISE_THRESHOLD_INT interrupt to host and instructs ETC module to ignore ETC updating automatically. And the MCU needs to reset CTC to initialize the baseline when the OVER_P_NOISE_THRESHOLD_INT is triggered.

If the difference data (baseline – signal) is larger than negative noise threshold, ETC update is ignored.

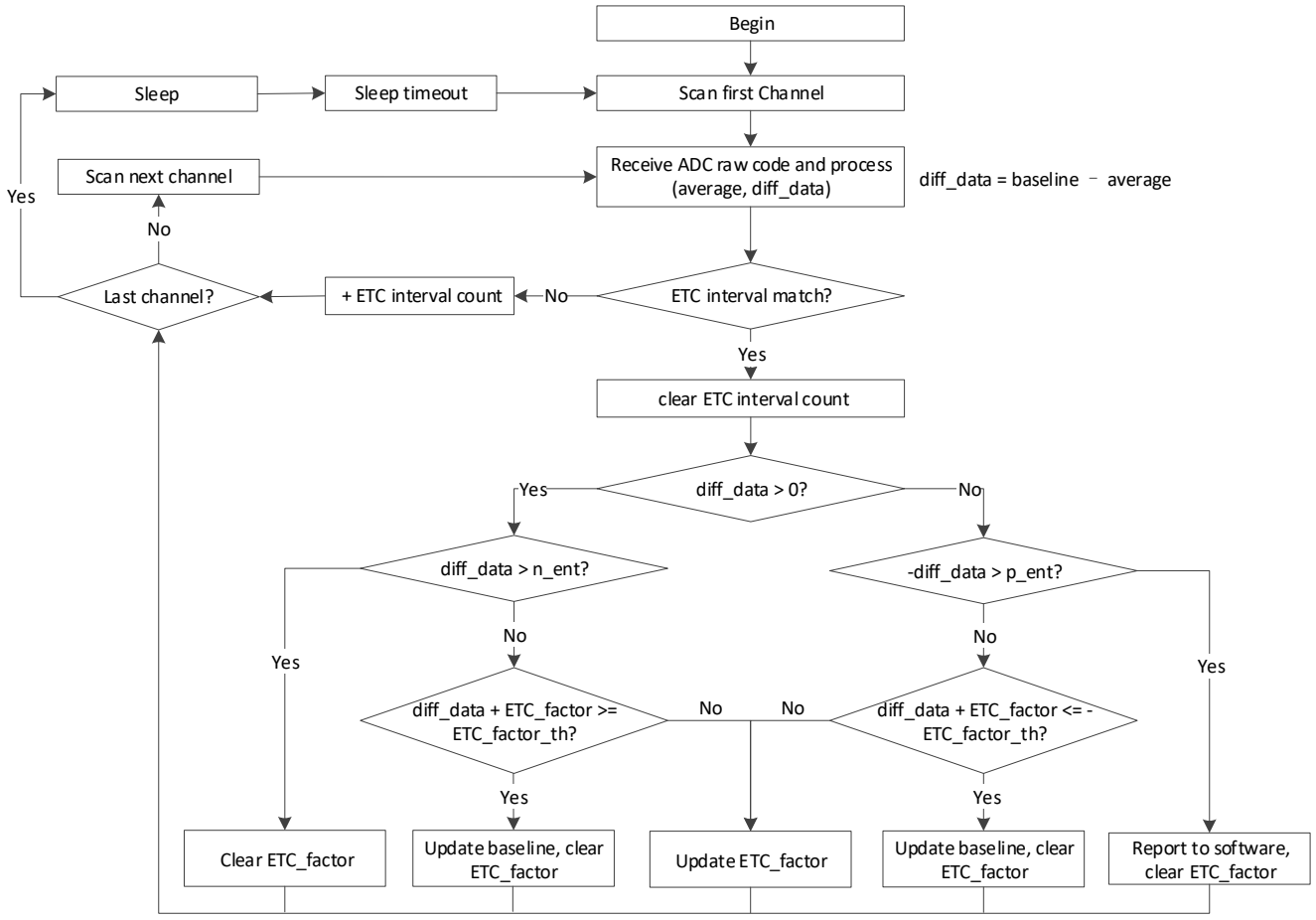


Figure 27-9 ETC scan and update operation flow

27.4.9 Debounce

The debounce feature avoids false triggering of sensors caused by noise spikes or system glitches by specifying the minimum time, for which a sensor has to be touched for a valid touch status.

The debounce value can vary depending on the sensor function. For example, the power button has a long debounce time to avoid inadvertently switching the system ON or OFF. Shorter debounce times speed up the device's response to a sensor touch events. The debounce parameter can be configured in the CTC_CTRL register. For example, when debounce is enabled and configured as 00 (2 times scan), finger touch interrupt is not sent to the host until 2 times continual finger touch event is detected. Sensor debounce function is suitable for both button application and proximity detection.

All information provided in this document is subject to legal disclaimers.
© REALTEK 2024. All rights reserved.

27.4.10 FIFO

When ADC finishes converting the sensor capacitance into digital raw data, it is sent into the Cap-Touch FIFO for average settle. The Cap-Touch FIFO supports 64*16 bits touch data of sensors' measured capacitance, and the maximum storage number is 64.

- FIFO depth: 64
- FIFO width: 16 bits

NOTE

When not in debug mode, the circuit fills the FIFO with filtered data sequentially to facilitate data acquisition. When the debug mode is enabled, the circuit is switched to fill the FIFO with raw data sampled by the ADC.

27.4.11 Interrupt

The CTC system consists of the following interrupt sources, all of the interrupts can be cleared by software.

Table 27-3 interrupt sources

Interrupt	Comment
TOUCH_PRESS_INTx	Cap-Touch single-ended mode press interrupt When the capture data of channel x is larger than touch threshold, a single-ended mode press touch is detected, and the CTC sends a touch interrupt signal to the MCU.
TOUCH_RELEASE_INTx	Cap-Touch single-ended mode release interrupt The TOUCH_RELEASE_INTx is sent to the MCU when the capture data of channel x is less than touch threshold after a TOUCH_PRESS_INTx has been sent to the host.
OVER_P_NOISE_THRESHOLD_INT	The OVER_P_NOISE THRESHOLD_INT is sent to the MCU when the difference data (signal – baseline) is larger than the positive noise threshold to indicate abnormal conditions.
AFIFO_OVERFLOW_INT	The AFIFO_OVERFLOW_INT is sent to the MCU when the FIFO overflows.
OVER_N_NOISE_THRESHOLD_INT	The OVER_N_NOISE THRESHOLD_INT is sent to the MCU when the difference data (baseline – signal) is larger than the negative noise threshold to indicate abnormal conditions.
Ctc_brk_by_adc_INT	Cap-Touch breaked by ADC enable
AFIFO_OVERLVL_INT	Raw code FIFO over level enable

27.4.12 Cap-Touch Initialization

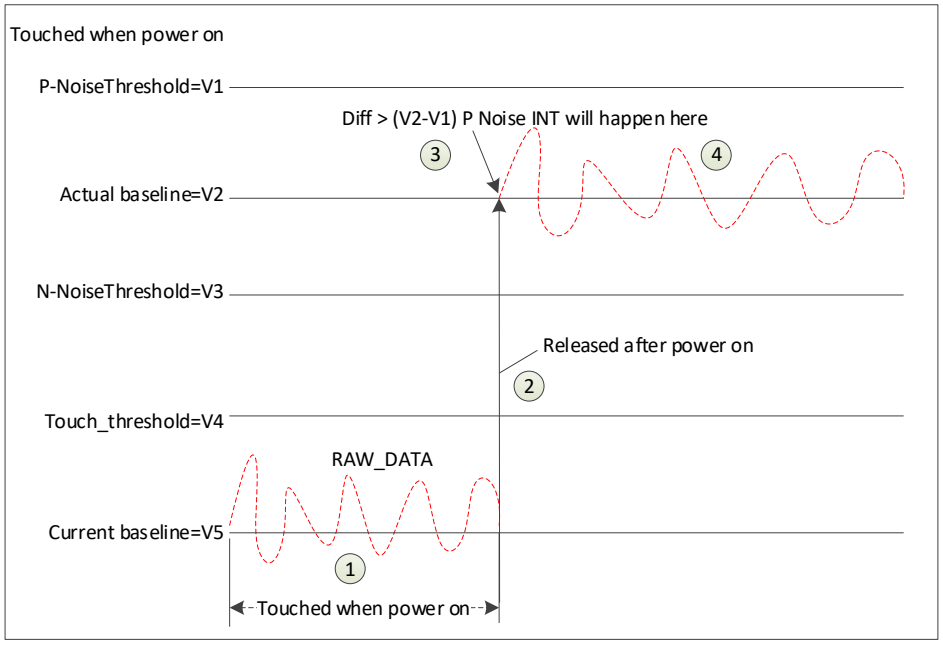


Figure 27-10 P-Noise interrupt for touch initialization with key pressed

All information provided in this document is subject to legal disclaimers.
© REALTEK 2024. All rights reserved.

Table 27-4 Touch initialization flow

Item	Step
Initialization with key pressed (RCU battery installation)	(1) Driver initialization ■ P-Noise threshold=V1 ■ N-Noise threshold=V3 ■ Touch diff threshold=600 or other value based on your PCB ■ ETC updates baseline (default=0, it costs 1 average cycle to get a baseline) and touch threshold based on RAW_DATA (of course these values aren't correct) (2) When touch released: ■ P-Noise interrupt happens (when P-Noise happens, ETC can't tune baseline to a correct value) ■ Driver disables channel (CHANNELx_ENABLE) ■ Driver enables channel (CHANNELx_ENABLE) and channel baseline (CTC_CTRL.BASELINE_INI) ■ ETC updates baseline and touch threshold based on RAW_DATA
Initialization without key pressed	Driver initialization (1) P-Noise threshold=V1 (2) N-Noise threshold=V3 (3) Touch diff threshold=600 or other value based on your PCB (4) ETC updates baseline and touch threshold based on RAW_DATA

27.4.13 Calibration

The CTC operates in auto scan mode effectively with default parameters, such as scan period, ETC parameter, sensitivity, threshold and debounce times. Most of CTC parameters calibrate automatically all the time, such as baseline and absolute threshold.

To achieve the best performance (sensitivity, reliability or response time), users only need to tune touch threshold, noise threshold and mbias current during development. Of course, users can tune other parameters manually to fit special requirements. There is no need to tune parameters during mass production (MP).

SNR information is helpful for difference threshold or noise threshold calibration. The recommended difference threshold is 80% of the difference threshold (baseline − touch signal), and 40% for noise threshold. Users can increase or decrease the threshold value to meet the SNR requirement (SNR>5).

After manual tuning, CTC calibrates the parameters automatically when ETC function is enabled.

NOTE

Realtek provides a tool named as Cap Test Tool to facilitate the debug and calibration of Cap-Touch in the development stage. Refer to AN1000 (Chapter: Cap Test Tool) for the detailed usage of Cap Test Tool.

27.4.14 Power Consumption

The active time can be set to 1 ~ 4ms, while the sleep time is 0 ~ 4095ms. There are some common power states and power consumption based on different recommend period settings. The power consumption of touch system includes ADC (100uA@131.072KHz), IMbias (50uA), and digital logic (can be ignored).

$$\text{Power consumption} = \begin{cases} Active\ period\ power + Sleep\ period\ power \\ Analog\ power + Digital\ power \end{cases}$$

27.5 Registers

Name	Address offset	Access	Description
REG_CTC_CTRL	000h	R/W	Cap-Touch Control Register
REG_SCAN_PERIOD	004h	R/W	Scan Parameters Register
REG_ETC_CTRL	008h	R/W	Environment Tracking Control Register
REG_SNR_INF	00Ch	R/W	Signal Noise Ratio Register
REG_DEBUG_MODE_CTRL	010h	R/W	Debug and Channel Swith Mode Register
REG_RAW_CODE_FIFO_STATUS	014h	R/W	FIFO Status Register
REG_RAW_CODE_FIFO_READ	018h	R	FIFO Read Register

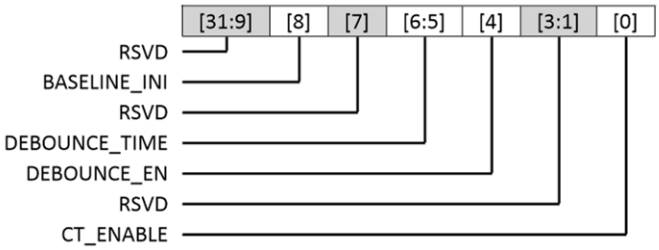
All information provided in this document is subject to legal disclaimers.
© REALTEK 2024. All rights reserved.

REG_INTERRUPT_ENABLE	020h	R/W	Interrupt Enable Register
REG_INTERRUPT_STATUS	024h	R	Interrupt Status Register
REG_RAW_INTERRUPT_STATUS	028h	R	Raw Interrupt Status Register
REG_INTERRUPT_ALL_CLR	030h	R/W	Interrupt All Clear Register
REG_INTERRUPT_STATUS_CLR	034h	R/W	Interrupt Clear Register
REG_DEBUG_SEL	040h	R/W	Debug Select Register
REG_DEBUG_PORT	044h	R	Debug Register
REG_ECO_USE0	048h	R/W	ECO USE0 Register
REG_ECO_USE1	04Ch	R/W	ECO USE1 Register
REG_CTC_COMP_VERSION	050h	R/W	Version Register
REG_CH0_CTRL	100h	R/W	Channel 0 Control Register
REG_CH0_NOISE_TH	104h	R/W	Channel 0 Noise Threshold Register
REG_CH0_MBIAS_ATH	108h	R/W	Channel 0 Mbias Register
REG_CH0_DATA_INF	10Ch	R/W	Channel 0 Dtouch Register
REG_CH1_CTRL	110h	R/W	Channel 1 Control Register
REG_CH1_NOISE_TH	114h	R/W	Channel 1 Noise Threshold Register
REG_CH1_MBIAS_ATH	118h	R/W	Channel 1 Mbias Register
REG_CH1_DATA_INF	11Ch	R/W	Channel 1 Dtouch Register
REG_CH2_CTRL	120h	R/W	Channel 2 Control Register
REG_CH2_NOISE_TH	124h	R/W	Channel 2 Noise Threshold Register
REG_CH2_MBIAS_ATH	128h	R/W	Channel 2 Mbias Register
REG_CH2_DATA_INF	12Ch	R/W	Channel 2 Dtouch Register
REG_CH3_CTRL	130h	R/W	Channel 3 Control Register
REG_CH3_NOISE_TH	134h	R/W	Channel 3 Noise Threshold Register
REG_CH3_MBIAS_ATH	138h	R/W	Channel 3 Mbias Register
REG_CH3_DATA_INF	13Ch	R/W	Channel 3 Dtouch Register
REG_CTC_SAMPLE_CTRL	200h	R/W	Cap-Touch Sample Control Register
REG_CTC_HW_TESTMODE	204h	R/W	Cap-Touch Hardware Test Mode Register
REG_CTC_FILTER_CTRL0	208h	R/W	Cap-Touch Filter Control0 Register
REG_CTC_FILTER_CTRL1	20Ch	R/W	Cap-Touch Filter Control1 Register
REG_CTC_DUMMY	210h	R/W	Cap-Touch Dummy Register

27.5.1 REG_CTC_CTRL

- **Name:** Cap-Touch Control Register
- **Size:** 32 bits
- **Address offset:** 000h
- **Read/write access:** R/W

This register includes Enable/Disable CTC Baseline initial Debounce function configuration.



Bit	Symbol	Access	Reset	Description
31:9	RSVD	-	0	Reserved
8	BASELINE_INI	R/W/EC	1h	1: Baseline initial function enable, HW will clear this bit to "0" after baseline initial
7	RSVD	-	0	Reserved
6:5	DEBOUNCE_TIME	R/W	0h	2'b00: 2 times 2'b01: 3 times 2'b10: 4 times 2'b11: 5 times

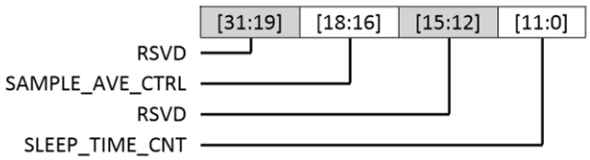
All information provided in this document is subject to legal disclaimers.
© REALTEK 2024. All rights reserved.

4	DEBOUNCE_EN	R/W	0h	0: Disable 1: Enable
3:1	RSVD	-	0	Reserved
0	CT_ENABLE	R/W	0h	0: CTC disable 1: CTC enable

27.5.2 REG_SCAN_PERIOD

- **Name:** Scan Parameters Register
- **Size:** 32 bits
- **Address offset:** 004h
- **Read/write access:** R/W

This register is about ADC sample number and scan interval.

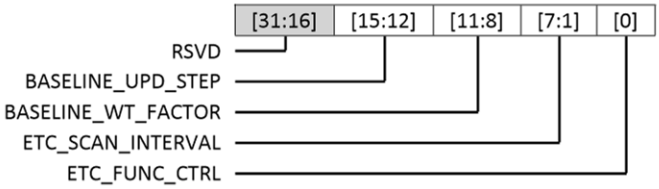


Bit	Symbol	Access	Reset	Description
31:19	RSVD	-	0	Reserved
18:16	SAMPLE_AVE_CTRL	R/W	6h	ADC sampled number for average function average number=2^(sample_ave_ctrl+2)
15:12	RSVD	-	0	Reserved
11:0	SLEEP_TIME_CNT	R/W	3Ch	Code: 0~4095 (0~0xFFF), unit is 1.024kHz cycle (1/32 32.768kHz). When this register is set to 0, HW will scan continuously and have no sleep time. Recommend value: 60~480ms

27.5.3 REG_ETC_CTRL

- **Name:** Environment Tracking Control Register
- **Size:** 32 bits
- **Address offset:** 008h
- **Read/write access:** R/W

This register is about ETC parameters and control signal.



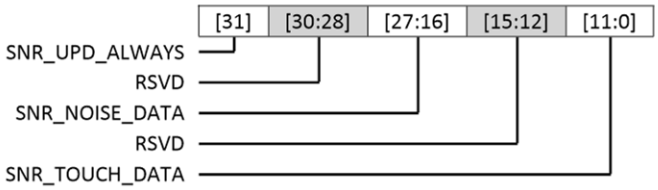
Bit	Symbol	Access	Reset	Description
31:16	RSVD	-	0	Reserved
15:12	BASELINE_UPD_STEP	R/W	1h	Baseline update step for all channels
11:8	BASELINE_WT_FACTOR	R/W	2h	Baseline update weight factor (ETC_factor_th) for all channels ETC_factor_th=2^baseline_wt_factor
7:1	ETC_SCAN_INTERVAL	R/W	2h	ETC update interval between scan period (sleep time) for all channels Interval=(etc_scan_interval+1)*scan_period
0	ETC_FUNC_CTRL	R/W	0h	Environmental cap tracking calibration function 0: Disable 1: Enable

All information provided in this document is subject to legal disclaimers.
© REALTEK 2024. All rights reserved.

27.5.4 REG_SNR_INF

- **Name:** Signal Noise Ratio Register
- **Size:** 32 bits
- **Address offset:** 00Ch
- **Read/write access:** R/W

This register is used to control SNR function.

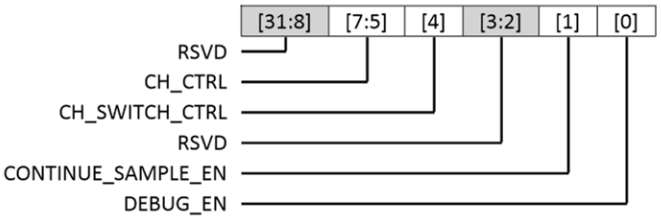


Bit	Symbol	Access	Reset	Description
31	SNR_UPD_ALWAYS	R/W	0	1: Update SNR info no matter touch or not (used only in debug function)
30:28	RSVD	-	0	Reserved
27:16	SNR_NOISE_DATA	R/W/ES	0	Noise peak to peak signal raw data for SNR monitor
15:12	RSVD	-	0	Reserved
11:0	SNR_TOUCH_DATA	R/W/ES	0h	Raw data of touch(signal-baseline)for SNR monitor

27.5.5 REG_DEBUG_MODE_CTRL

- **Name:** Debug and Channel Swith Mode Register
- **Size:** 32 bits
- **Address offset:** 010h
- **Read/write access:** R/W

This register is used to control CTC operation mode and channel.



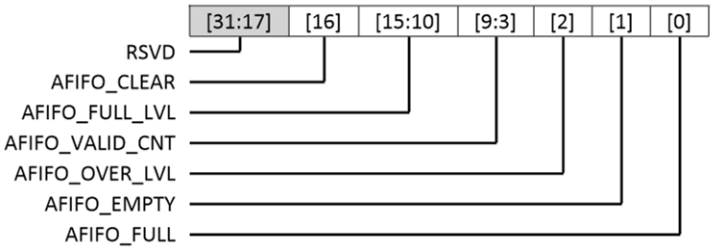
Bit	Symbol	Access	Reset	Description
31:8	RSVD	-	0	Reserved
7:5	CH_CTRL	R/W	0h	Scan channel control 000: channel 0 001: channel 1 010: channel 2 011: channel 3
4	CH_SWITCH_CTRL	R/W	1h	Scan channel switch control 0: Auto switch function disable (manual switch by ch_ctrl) 1: Auto switch to next channel (channel 0 --> channel 3 --> channel 0)
3:2	RSVD	-	0	Reserved
1	CONTINUE_SAMPLE_EN	R/W	0	continue sample one channel mode enable, only take effect in debug mode 1: enable 0: disable
0	DEBUG_EN	R/W	0	Debug mode enable 0: Disable 1: Enable

All information provided in this document is subject to legal disclaimers.
© REALTEK 2024. All rights reserved.

27.5.6 REG_RAW_CODE_FIFO_STATUS

- **Name:** FIFO Status Register
- **Size:** 32 bits
- **Address offset:** 014h
- **Read/write access:** R/W

This register is used to indicate the FIFO status.



Bit	Symbol	Access	Reset	Description
31:17	RSVD	-	0	Reserved
16	AFIFO_CLEAR	WA0	0	clear the FIFO data
15:10	AFIFO_FULL_LVL	R/W	0	user define the full level for afifo, the valid value is from 0 to 63
9:3	AFIFO_VALID_CNT	R	0h	Raw code FIFO valid count (push unit number which can be read)
2	AFIFO_OVER_LVL	R	0	0: Raw code FIFO not over level 1: Raw code FIFO over level
1	AFIFO_EMPTY	R	1	0: Raw code FIFO not empty 1: Raw code FIFO empty
0	AFIFO_FULL	R	0	0: Raw code FIFO not oveflow (not full) 1: Raw code FIFO overflow (full)

27.5.7 REG_RAW_CODE_FIFO_READ

- **Name:** FIFO Read Register
- **Size:** 32 bits
- **Address offset:** 018h
- **Read/write access:** R

This register is used to read the raw data.



Bit	Symbol	Access	Reset	Description
31	AFIFO_RD_DATA_VLD	R	0h	Read data from raw code FIFO valid
30:16	RSVD	-	0	Reserved
15:12	AFIFO_RD_CH_NUM	RP	0h	Read data from raw code FIFO [15:12]: channel number
11:0	AFIFO_RD_DATA	RP	0h	Read data from raw code FIFO [11:0]: data

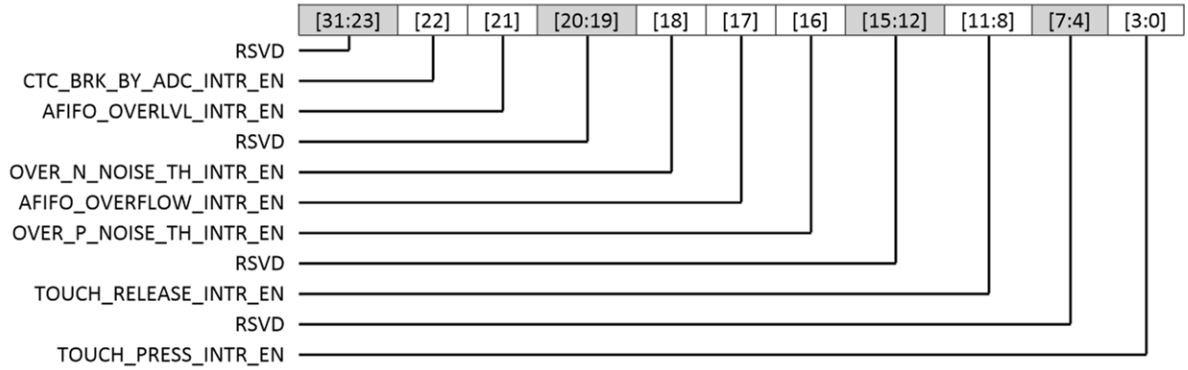
27.5.8 REG_INTERRUPT_ENABLE

- **Name:** Interrupt Enable Register
- **Size:** 32 bits

All information provided in this document is subject to legal disclaimers.
© REALTEK 2024. All rights reserved.

- **Address offset:** 020h
- **Read/write access:** R/W

This register is used to enable the specific interrupt.
- 0: disable interrupt
- 1: enable interrupt

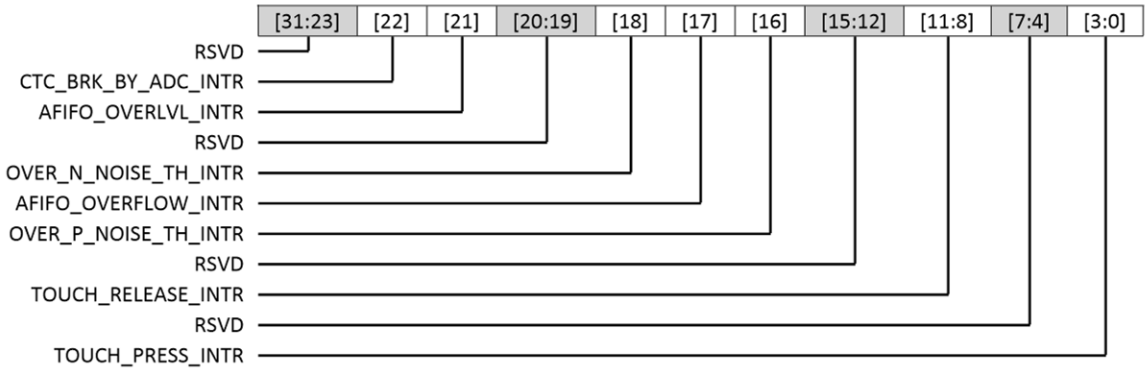


Bit	Symbol	Access	Reset	Description
31:23	RSVD	-	0	Reserved
22	CTC_BRK_BY_ADC_INTR_EN	R/W	0h	Cap-Touch breaked by adc enable
21	AFIFO_OVERLVL_INTR_EN	R/W	0h	Raw code FIFO over level enable
20:19	RSVD	-	0	Reserved
18	OVER_N_NOISE_TH_INTR_EN	R/W	0h	Negative noise threshold overflow enable
17	AFIFO_OVERFLOW_INTR_EN	R/W	0h	Raw code FIFO overflow enable
16	OVER_P_NOISE_TH_INTR_EN	R/W	0h	Positive noise threshold overflow enable
15:12	RSVD	-	0	Reserved
11:8	TOUCH_RELEASE_INTR_EN	R/W	0h	Channelx single touch release enable, x is 0~3
7:4	RSVD	-	0	Reserved
3:0	TOUCH_PRESS_INTR_EN	R/W	0h	Channelx single touch press enable, x is 0~3

27.5.9 REG_INTERRUPT_STATUS

- **Name:** Interrupt Status Register
- **Size:** 32 bits
- **Address offset:** 024h
- **Read/write access:** R

Register to indicate the interrupt status.
- 0: no interrupt
- 1: interrupt



Bit	Symbol	Access	Reset	Description
31:23	RSVD	-	0	Reserved

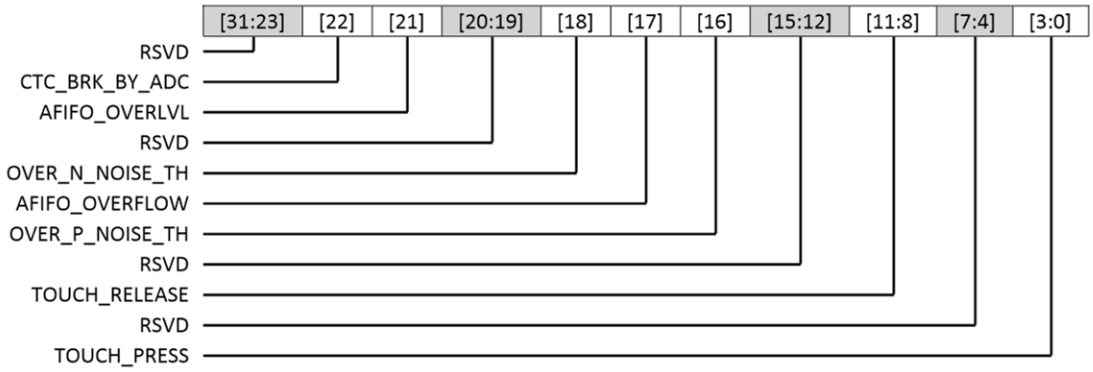
All information provided in this document is subject to legal disclaimers.
© REALTEK 2024. All rights reserved.

22	CTC_BRK_BY_ADC_INTR	R	0h	Cap-Touch breaked by adc interrupt
21	AFIFO_OVERLVL_INTR	R	0h	Raw code FIFO over level interrupt
20:19	RSVD	-	0	Reserved
18	OVER_N_NOISE_TH_INTR	R	0h	Negative noise threshold overflow interrupt
17	AFIFO_OVERFLOW_INTR	R	0h	Raw code FIFO overflow interrupt
16	OVER_P_NOISE_TH_INTR	R	0h	Positive noise threshold overflow interrupt
15:12	RSVD	-	0	Reserved
11:8	TOUCH_RELEASE_INTR	R	0h	Channelx single touch release interrupt, x is 0~3
7:4	RSVD	-	0	Reserved
3:0	TOUCH_PRESS_INTR	R	0h	Channelx single touch press interrupt, x is 0~3

27.5.10 REG_RAW_INTERRUPT_STATUS

- **Name:** Raw Interrupt Status Register
- **Size:** 32 bits
- **Address offset:** 028h
- **Read/write access:** R

This register is used to indicate the raw interrupt status.
- 0: no interrupt
- 1: interrupt



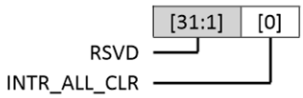
Bit	Symbol	Access	Reset	Description
31:23	RSVD	-	0	Reserved
22	CTC_BRK_BY_ADC	R	0h	Cap-Touch breaked by adc raw interrupt
21	AFIFO_OVERLVL	R	0h	Raw code FIFO over level raw interrupt
20:19	RSVD	-	0	Reserved
18	OVER_N_NOISE_TH	R	0h	Negative noise threshold overflow raw interrupt
17	AFIFO_OVERFLOW	R	0h	Raw code FIFO overflow raw interrupt
16	OVER_P_NOISE_TH	R	0h	Positive noise threshold overflow raw interrupt
15:12	RSVD	-	0	Reserved
11:8	TOUCH_RELEASE	R	0h	Channelx single touch release raw interrupt, x is 0~3
7:4	RSVD	-	0	Reserved
3:0	TOUCH_PRESS	R	0h	Channelx single touch press raw interrupt, x is 0~3

27.5.11 REG_INTERRUPT_ALL_CLR

- **Name:** Interrupt All Clear Register
- **Size:** 32 bits
- **Address offset:** 030h
- **Read/write access:** R/W

This register is used to to clear all interrupts.

All information provided in this document is subject to legal disclaimers.
© REALTEK 2024. All rights reserved.

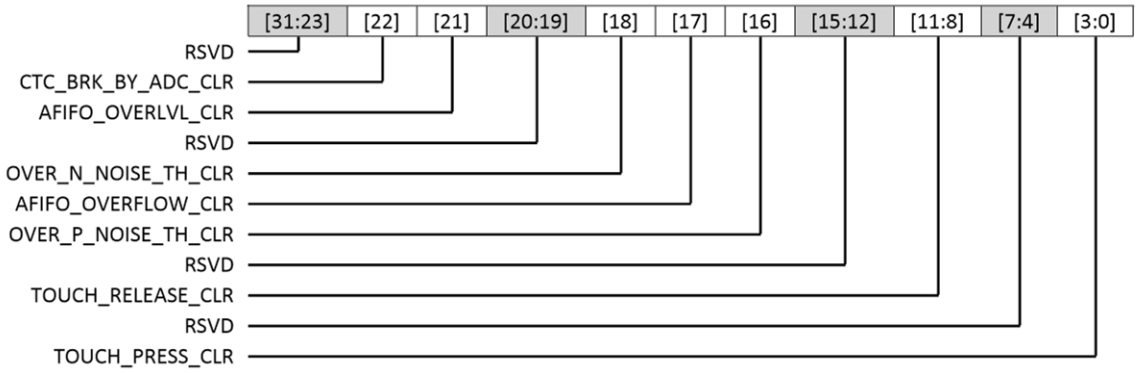


Bit	Symbol	Access	Reset	Description
31:1	RSVD	-	0	Reserved
0	INTR_ALL_CLR	R/WA0	0h	Write "1" to this register to clear the combined interrupts, all individual interrupts, interrupt status register and raw interrupt status register.

27.5.12 REG_INTERRUPT_STATUS_CLR

- **Name:** Interrupt Clear Register
- **Size:** 32 bits
- **Address offset:** 034h
- **Read/write access:** R/W

This register is used to clear the specific interrupt.

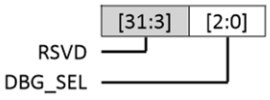


Bit	Symbol	Access	Reset	Description
31:23	RSVD	-	0	Reserved
22	CTC_BRK_BY_ADC_CLR	R/WA0	0h	Clear Cap-Touch breaked by adc raw interrupt and related register
21	AFIFO_OVERLVL_CLR	R/WA0	0h	Clear raw code FIFO over level raw interrupt and related register
20:19	RSVD	-	0	Reserved
18	OVER_N_NOISE_TH_CLR	R/WA0	0h	Clear negative noise threshold overflow interrupt and related register
17	AFIFO_OVERFLOW_CLR	R/WA0	0h	Clear raw code FIFO overflow interrupt and related register
16	OVER_P_NOISE_TH_CLR	R/WA0	0h	Clear positive noise threshold overflow interrupt and related register
15:12	RSVD	-	0	Reserved
11:8	TOUCH_RELEASE_CLR	R/WA0	0h	Clear channelx single touch release interrupt and related register
7:4	RSVD	-	0	Reserved
3:0	TOUCH_PRESS_CLR	R/WA0	0h	Clear channelx single touch press interrupt and related register

27.5.13 REG_DEBUG_SEL

- **Name:** Debug Select Register
- **Size:** 32 bits
- **Address offset:** 040h
- **Read/write access:** R/W

This register is used to select the debug information.



All information provided in this document is subject to legal disclaimers.
© REALTEK 2024. All rights reserved.

Bit	Symbol	Access	Reset	Description
31:3	RSVD	-	0	Reserved
2:0	DBG_SEL	R/W	0h	Debug information selection 0: dbg_clkrst 1: dbg_apbslv 2: dbg_ctrl 3: dbg_decision 4: dbg_adc_arb Others: dbg_clkrst

27.5.14 REG_DEBUG_PORT

- **Name:** Debug Register
- **Size:** 32 bits
- **Address offset:** 044h
- **Read/write access:** R

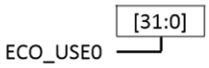
This register is for debug info.



Bit	Symbol	Access	Reset	Description
31:0	DBG_PORT	R	0h	Debug port output value,depend on dbg_sel value

27.5.15 REG_ECO_USE0

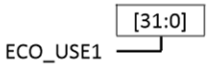
- **Name:** ECO USE0 Register
- **Size:** 32 bits
- **Address offset:** 048h
- **Read/write access:** R/W



Bit	Symbol	Access	Reset	Description
31:0	ECO_USE0	R/W	0h	For ECO use

27.5.16 REG_ECO_USE1

- **Name:** ECO USE1 Register
- **Size:** 32 bits
- **Address offset:** 04Ch
- **Read/write access:** R/W



Bit	Symbol	Access	Reset	Description
31:0	ECO_USE1	R/W	0h	For ECO use

27.5.17 REG_CTC_COMP_VERSION

- **Name:** Version Register
- **Size:** 32 bits
- **Address offset:** 050h
- **Read/write access:** R/W

All information provided in this document is subject to legal disclaimers.
© REALTEK 2024. All rights reserved.

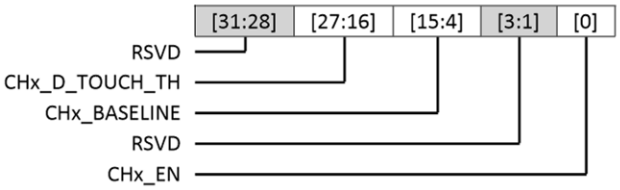


Bit	Symbol	Access	Reset	Description
31:0	COMP_VERSION	R/W	20180123h	CTC version number

27.5.18 REG_CHx_CTRL

- **Name:** Channel x Control Register (x={0, 1, 2, 3})
- **Size:** 32 bits
- **Address offset:**
  - **REG_CH0_CTRL:** 100h
  - **REG_CH1_CTRL:** 110h
  - **REG_CH2_CTRL:** 120h
  - **REG_CH3_CTRL:** 130h
- **Read/write access:** R/W

This register is used to configure channel parameters.



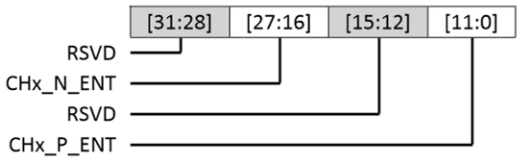
Bit	Symbol	Access	Reset	Description
31:28	RSVD	-	0	Reserved
27:16	CHx_D_TOUCH_TH	R/W	64h	Difference threshold data of touch judgement for channel x. It needs to be configured during development (0x0~0xFFF) (0~4095). Init number=0x1E Recommend data=80%*(signal-baseline)
15:4	CHx_BASELINE	R/W/ES	0h	Digital baseline data of channel x. Init number=0x0 It can be initialed by HW when baseline initial function enabled, and can be updated automatically by HW when ETC function enabled. It could be configured by manual tuning.
3:1	RSVD	-	0	Reserved
0	CHx_EN	R/W	0h	Cap sensor activity control of channel x 0: Disable 1: Enable

27.5.19 REG_CHx_NOISE_TH

- **Name:** Channel x Noise Threshold Register (x={0, 1, 2, 3})
- **Size:** 32 bits
- **Address offset:**
  - **REG_CH0_NOISE_TH:** 104h
  - **REG_CH1_NOISE_TH:** 114h
  - **REG_CH2_NOISE_TH:** 124h
  - **REG_CH3_NOISE_TH:** 134h
- **Read/write access:** R/W

This register is used to configure noise threshold.

All information provided in this document is subject to legal disclaimers.
© REALTEK 2024. All rights reserved.

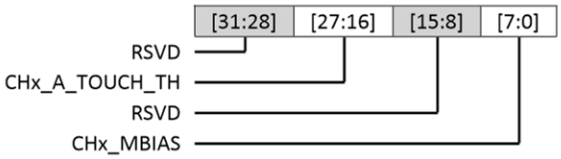


Bit	Symbol	Access	Reset	Description
31:28	RSVD	-	0	Reserved
27:16	CHx_N_ENT	R/W	28h	The environmental negative noise threshold: the negative maximum capacitance change of raw data that is still considered an environment change.
15:12	RSVD	-	0	Reserved
11:0	CHx_P_ENT	R/W	28h	The environmental positive noise threshold: the positive maximum capacitance change of raw data that is still considered an environment change.

27.5.20 REG_CHx_MBIAS_ATH

- **Name:** Channel x Mbias Register (x={0, 1, 2, 3})
- **Size:** 32 bits
- **Address offset:**
  - **REG_CH0_MBIAS_ATH:** 108h
  - **REG_CH1_MBIAS_ATH:** 118h
  - **REG_CH2_MBIAS_ATH:** 128h
  - **REG_CH3_MBIAS_ATH:** 138h
- **Read/write access:** R/W

This register is about mbias and a touch threshold.



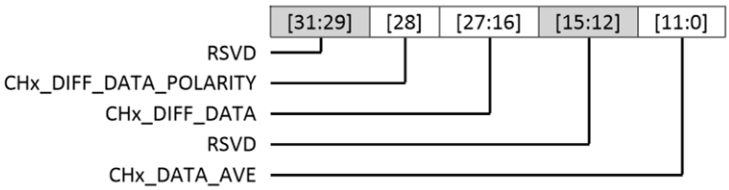
Bit	Symbol	Access	Reset	Description
31:28	RSVD	-	0	Reserved
27:16	CHx_A_TOUCH_TH	R/W/ES	0h	Absolute threshold data of touch judgement for channel x. It can be updated by HW when baseline initial function or ETC function enabled, a_touch_th=baseline-d_touch_th, manual tuning when ETC disabled (0x0~0xFFF). Recommended data=80%*signal
15:8	RSVD	-	0	Reserved
7:0	CHx_MBIAS	R/W	20h	Touch mbias current [7:0] 32u/16u/8u/4u/2u/1u/0.5u/0.25u for channel x

27.5.21 REG_CHx_DATA_INF

- **Name:** Channel x Dtouch Register (x={0, 1, 2, 3})
- **Size:** 32 bits
- **Address offset:**
  - **REG_CH0_DATA_INF:** 10Ch
  - **REG_CH1_DATA_INF:** 11Ch
  - **REG_CH2_DATA_INF:** 12Ch
  - **REG_CH3_DATA_INF:** 13Ch
- **Read/write access:** R/W

This register is for difference touch mode.

All information provided in this document is subject to legal disclaimers.
© REALTEK 2024. All rights reserved.

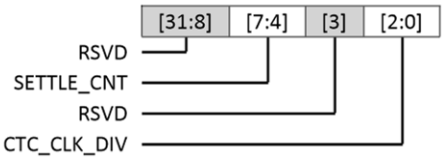


Bit	Symbol	Access	Reset	Description
31:29	RSVD	-	0	Reserved
28	CHx_DIFF_DATA_POLARITY	R/W/ES	0h	Polarity of chx_diff_data 0: chx_data_ave<baseline 1: chx_data_ave>=baseline
27:16	CHx_DIFF_DATA	R/W/ES	0h	Difference digital data of channelx between chx_data_ave and baseline
15:12	RSVD	-	0	Reserved
11:0	CHx_DATA_AVE	R/W/ES	0h	Average of channel x raw code

27.5.22 REG_CTC_SAMPLE_CTRL

- **Name:** Cap-Touch Sample Control Register
- **Size:** 32 bits
- **Address offset:** 200h
- **Read/write access:** R/W

This register is about Cap-Touch sample control



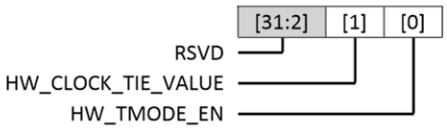
Bit	Symbol	Access	Reset	Description
31:8	RSVD	-	0	Reserved
7:4	SETTLE_CNT	R/W	0h	Settle count for sample data from ADC to captouch, the unit is sample clock. 0 means no need to settle
3	RSVD	-	0	Reserved
2:0	CTC_CLK_DIV	R/W	4h	This field defines clock driver level of ADC module. the source clock is osc4M A value of 0 is for clock divided by 4. A value of 1 is for clock divided by 8. A value of 2 is for clock divided by 16. A value of 3 is for clock divided by 24. A value of 4 is for clock divided by 32. A value of 5 is for clock divided by 64. A value of 6 is for clock divided by 128.

27.5.23 REG_CTC_HW_TESTMODE

- **Name:** Cap-Touch Hardware Test Mode Register
- **Size:** 32 bits
- **Address offset:** 204h
- **Read/write access:** R/W

This register is only used for internal test.

All information provided in this document is subject to legal disclaimers.
© REALTEK 2024. All rights reserved.

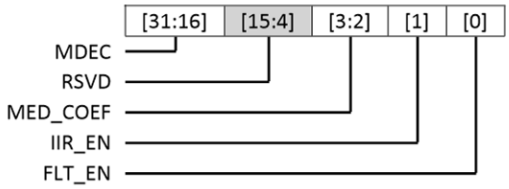


Bit	Symbol	Access	Reset	Description
31:2	RSVD	-	0	Reserved
1	HW_CLOCK_TIE_VALUE	R/W	0h	This value is the clock tied value in Cap-Touch hardware test mode
0	HW_TMODE_EN	R/W	0h	0: hardware test mode disabled 1: hardware test mode enabled, the clock sent to AUX Analog from Cap-Touch is tie to fixed value, which depends on hw_clock_tie_value (410h[1]).

27.5.24 REG_CTC_FILTER_CTRL0

- **Name:** Cap-Touch Filter Control0 Register
- **Size:** 32 bits
- **Address offset:** 208h
- **Read/write access:** R/W

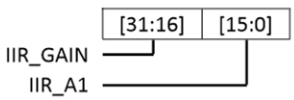
This register is about Cap-Touch filter control.



Bit	Symbol	Access	Reset	Description
31:16	MDEC	R/W	Ah	Decimation of sample point
15:4	RSVD	-	0	Reserved
3:2	MED_COEF	R/W	1	No use Median filter coefficient 00:disable 01:coef is 3 10:coef is 5 11:coef is 7
1	IIR_EN	R/W	0	Enable CTC IIR function
0	FLT_EN	R/W	0	Enable CTC filter function (global)

27.5.25 REG_CTC_FILTER_CTRL1

- **Name:** Cap-Touch Filter Control1 Register
- **Size:** 32 bits
- **Address offset:** 20Ch
- **Read/write access:** R/W

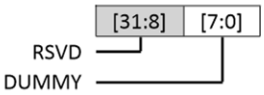


Bit	Symbol	Access	Reset	Description
31:16	IIR_GAIN	R/W	25Dh	Gain coef, Q1.15
15:0	IIR_A1	R/W	C4BBh	Denominator coef, Q1.15

All information provided in this document is subject to legal disclaimers.
© REALTEK 2024. All rights reserved.

27.5.26 REG_CTC_DUMMY

- **Name:** Cap-Touch Dummy Register
- **Size:** 32 bits
- **Address offset:** 210h
- **Read/write access:** R/W



Bit	Symbol	Access	Reset	Description
31:8	RSVD	-	0	Reserved
7:0	DUMMY	R/W	f0h	Dummy register

All information provided in this document is subject to legal disclaimers.
© REALTEK 2024. All rights reserved.

28 Analog-to-Digital Converter (ADC)

This chapter describes the architecture and function of the 12-bit analog-to-digital converter (ADC) of RTL8721Dx. The characteristics and principles of the ADC are explained, and the different operation modes are given in details.

28.1 Introduction

The ADC of the RTL8721Dx is a successive-approximation register (SAR) ADC, which includes up to 8 external channels and 3 internal channels. Among the external channels, 1 VBAT channel and 7 normal channels (V33 channel) all support divided mode.

Table 28-1 shows the description of the external channels.

Table 28-1 Description of the external channels

Table Function	ADC channel ID	Pin name	Input voltage
V33 channel	CH0~CH6	PB13~PB19	0~3.3V (Single-ended divided mode)
VBAT channel	CH7	BAT_MEAS	0 ~ 5V

Figure 28-1 shows the block diagram of the 12-bit ADC.

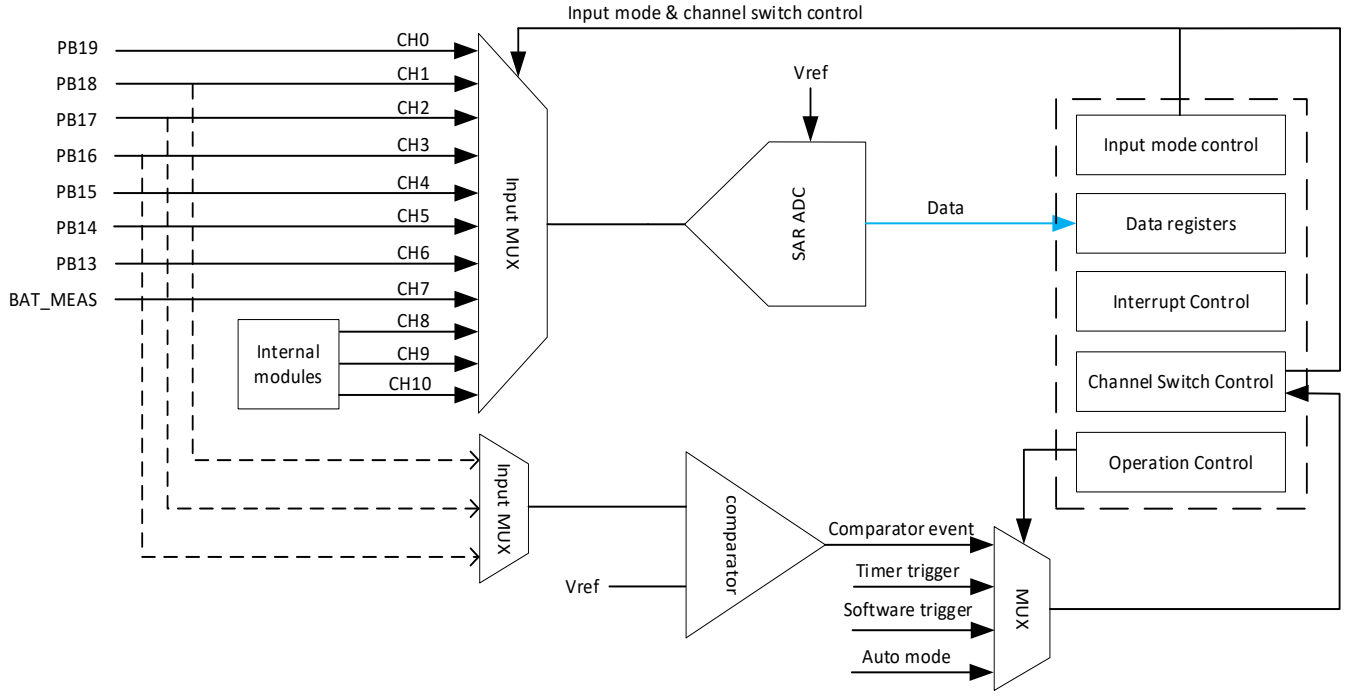


Figure 28-1 Block diagram of SAR ADC

28.2 Features

The features of ADC are listed below and the following sections will describe them in details.

- Resolution: 12-bit SAR
- ENOB: 10-bit
- Available channel number
  - 7 external channels and 1 VBAT channel
  - 3 internal channels
- Configurable input
  - Single-ended
  - Differential with predefined channel pair
- Input voltage range
  - Battery voltage channel: 0V ~ 5V
  - Single-ended divided mode

All information provided in this document is subject to legal disclaimers.
© REALTEK 2024. All rights reserved.

    - 0 ~ min(VAH_ADC, 3.3V)
  - Differential divided mode(|P-N|)
    - 0 ~ min(VAH_ADC/2, 1.65V)
- Configurable ADC clock source
- Configurable channel switch order and channel number
- Contain 64 FIFO entry which is 20-bit width
- Individual channel compare mode
- Individual channel register
- Multi-sampling trigger sources
  - Software
  - Timer
- Manual and auto mode conversion
  - Manual mode for software controllable conversion
  - Auto mode for hardware continuous conversion
- Work with comparator
- Hardware oversample

28.3 Functional Description

28.3.1 SAR ADC Principle

Figure 28-2 summaries the typical architecture of SAR ADC.

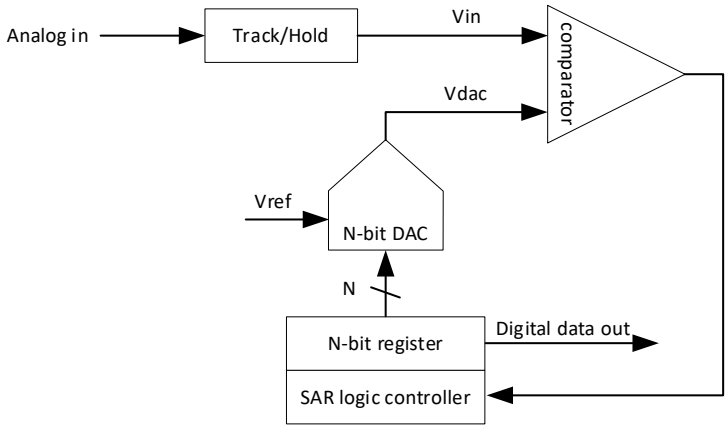


Figure 28-2 Typical architecture of SAR ADC

28.3.1.1 V33 Channel in Different Mode

When V33 channel is configured as single-ended divided mode, the simplified diagram is shown in *Figure 28-3*. A specific divider ratio with S3 open and S4 closed is used to increase the input voltage range proportionally.

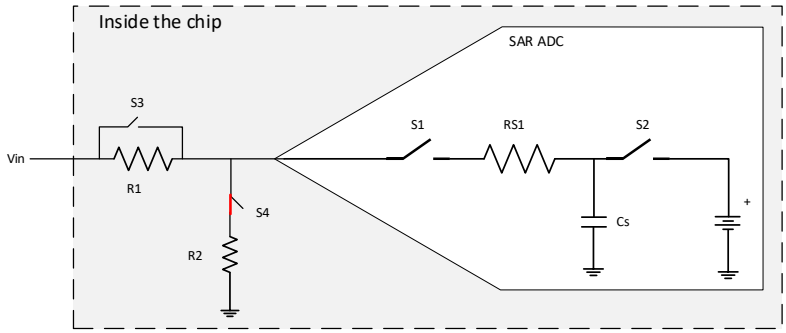


Figure 28-3 Simplified input diagram in single-ended divided mode

In single-ended divided mode, the equivalent resistance R is composed of R1 and R2. Due to the chip process and the environment temperature, the R in different ICs has different value, and variation is up to 10%. Application note introduces the influence and calibration of the internal R in detail.

All information provided in this document is subject to legal disclaimers.
© REALTEK 2024. All rights reserved.

The configuration of divided mode refers to register ANA_ADC_REG0X_LPAD [21:16], each channel can be configured individually.

Table 28-2 shows the input impedance (R) of different channels.

Table 28-2 Input impedance (R) of different channels

Function	Input mode	Input impedance
V33 channel	Single-ended divided mode	491kΩ (typical)
	Single-ended bypass mode	7MΩ (typical)

28.3.2 Clock Generation

Figure 28-4 shows the clock generation tree. The register REG_ADC_CLK_DIV[2:0] is the frequency divider, of which the default setting is 4/24MHz.

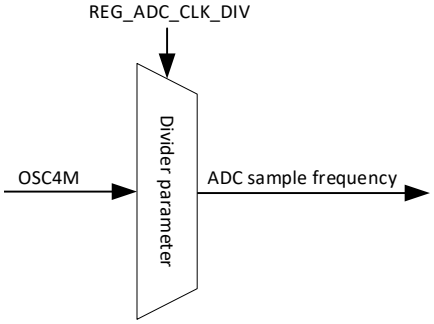


Figure 28-4 Clock generation tree

28.3.3 Input Mode

The external 6 channels supports two input mode: single-ended mode and differential mode. In the single-ended mode, it only needs one pin which has multi applications, such as thermal sensor, voltage monitoring and etc. Differential mode has better common mode rejection ratio (CMRR), but needs two pins. Only three pairs of pins support differential mode, they are CH0&CH1, CH2&CH3, and CH4&CH5.

The configuration of input mode refers to register REG_ADC_IN_TYPE[5:0], each channel can be configured individually.

28.3.3.1 Single-ended Mode

In the single-ended divided mode, the typical input voltage range is 0~3.3V.

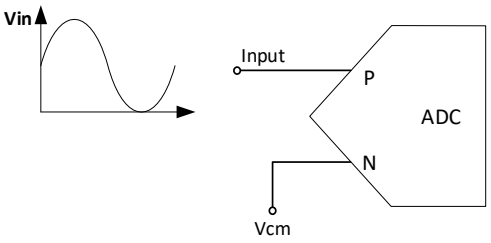


Figure 28-5 Single-ended mode

28.3.3.2 Differential Mode

In the differential divided mode, the typical value of Vcm (Vcm = (Vp + Vn)/2) is 1.65V, and the typical input range of |P-N| is 0~1.65V. For only one pin, the typical input range to GND is 0.825V~2.475V.

All information provided in this document is subject to legal disclaimers.
© REALTEK 2024. All rights reserved.

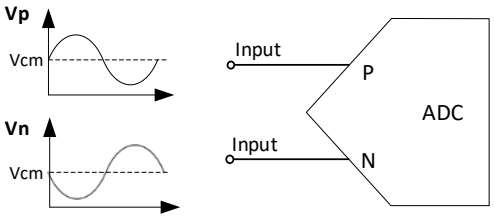


Figure 28-6 Differential mode

28.3.4 Operation Mode

The RTL8721Dx supports four different operation modes: auto mode, timer-trigger mode, software-trigger mode, and comparator-assist mode. Different modes can meet different requirements, but only one operation mode can be used at the same time. All modes can execute conversions according to channel switch list. The major difference between the operation modes is their trigger source, which drives ADC module to do conversions.

28.3.4.1 ADC Channel Switch List

ADC provides a mechanism for software to decide a whole conversion order – ADC channel switch list. Two registers (REG_ADC_CHSW_LIST_0, REG_ADC_CHSW_LIST_1) allow software to fill channel number to be converted. If software does not require all channels, a list length field in register REG_ADC_CONF[7:4] can be used to reduce channel number of the list.

For example, if the value in REG_ADC_CHSW_LIST_0 and REG_ADC_CHSW_LIST_1 are as follows, when REG_ADC_CONF[7:4] = 9, only the first 10 (the value of REG_ADC_CONF[7:4] + 1) channels in the list can be converted by ADC. If ADC operates in auto mode, it starts to do conversions from the first channel to the next in this list one by one. When channel length count (CNT) adds up to the value of REG_ADC_CONF[7:4], ADC will do conversions from the first channel again (see *Figure 28-7*).

Register	Bit & Value							
REG_ADC_CHSW_LIST_0	[3:0]	[7:4]	[11:8]	[15:12]	[19:16]	[23:20]	[27:24]	[31:28]
	0	1	2	3	4	5	8	0
REG_ADC_CHSW_LIST_1	[3:0]	[7:4]	[11:8]	[15:12]	[19:16]	[23:20]	[27:24]	[31:28]
	1	8	5	4	3	2	1	0

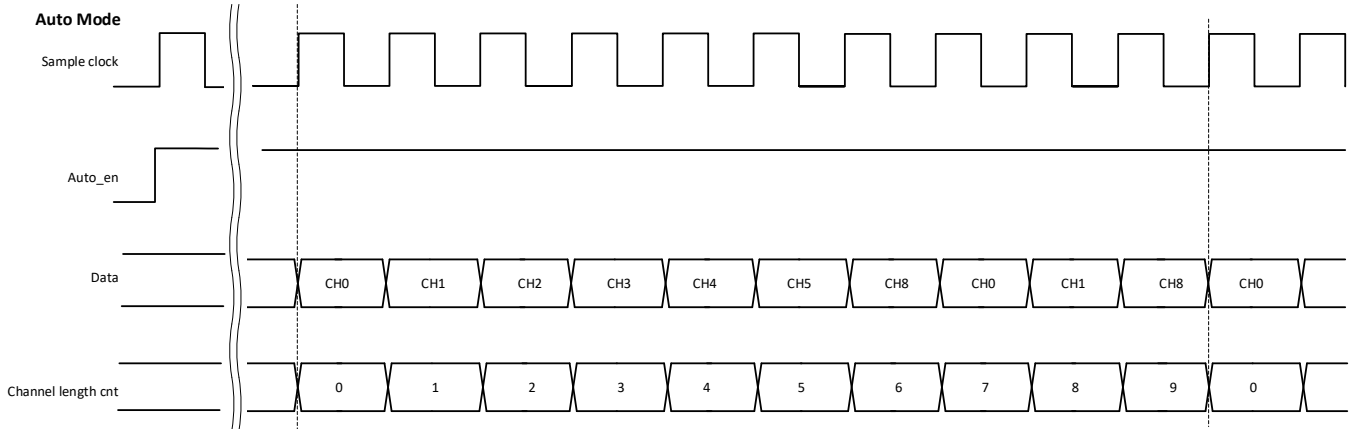


Figure 28-7 Auto mode waveform

28.3.4.2 Auto Mode

In auto mode, ADC continuously executes conversions without any trigger sources after auto channel switch register is set. Auto mode would not stop conversions until auto channel switch register is clear (register REG_ADC_AUTO_CSW_CTRL[0]) or the auto channel switch is disabled (register REG_ADC_CONF[3:1]).

In auto mode, ADC samples automatically by hardware without assistance of core or other modules, it can be applied to the situation where core resources are limited and real-time detection is required. Refer to *Figure 28-7* fort the auto mode waveform.

All information provided in this document is subject to legal disclaimers.
© REALTEK 2024. All rights reserved.

28.3.4.3 Timer-trigger Mode

In timer-trigger mode, ADC can cooperate with timer modules to do conversions. System software sets timer trigger control register to enable timer module and set an appropriate timer event period. After that, ADC do one conversions when one timer event occurs based on system design.

Same as auto mode, the time- trigger mode can work independently without system software, which means it can reduce the core resources partly. What's more, ADC can automatically go to shut down mode between two timer trigger events that can largely reduce the ADC power consumption itself. Refer to *Figure 28-8* for the timer-trigger mode waveform.

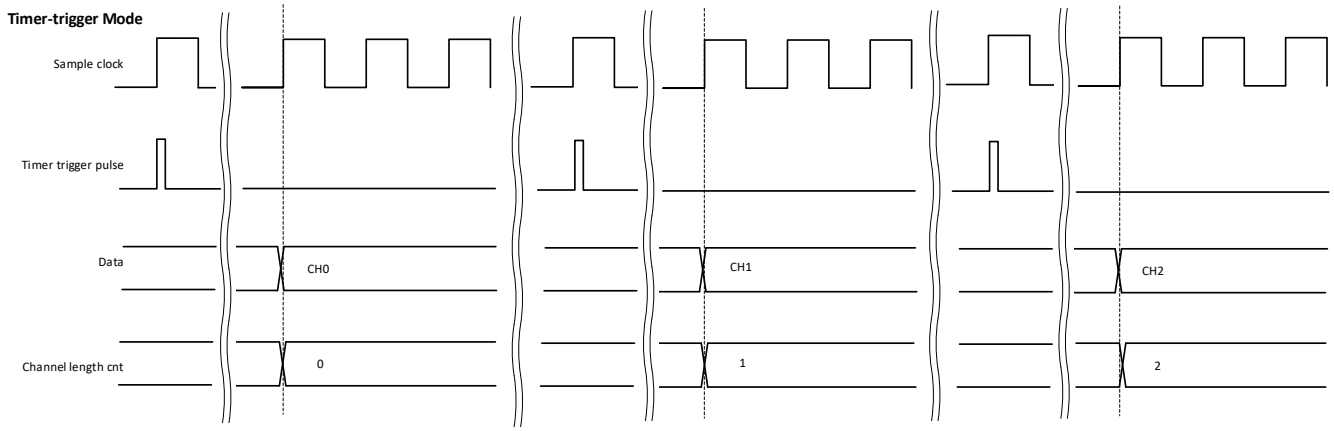


Figure 28-8 Timer-trigger mode waveform

NOTE

The RTL8721Dx supports up to 8 timer sources (timer 0~7).

28.3.4.4 Software-trigger Mode

The software-trigger mode is similar with timer-trigger mode, and the difference between them is the trigger source. For general usage, system software tends to handle ADC conversion timing. Every time software trigger register (REG_ADC_SW_TRIG[0]) is set, ADC module does channel switch to the next channel in conversion list. System software checks ready/busy register (REG_ADC_BUSY_STS[0]) before reading conversion data register (REG_ADC_DATA_GLOBAL[15:0]).

Compare with timer-trigger mode, this mode is more flexible. Software can manually trigger ADC sampling according to the actual situation, and the trigger time is not fixed. Refer to *Figure 28-9* for the software-trigger mode waveform.

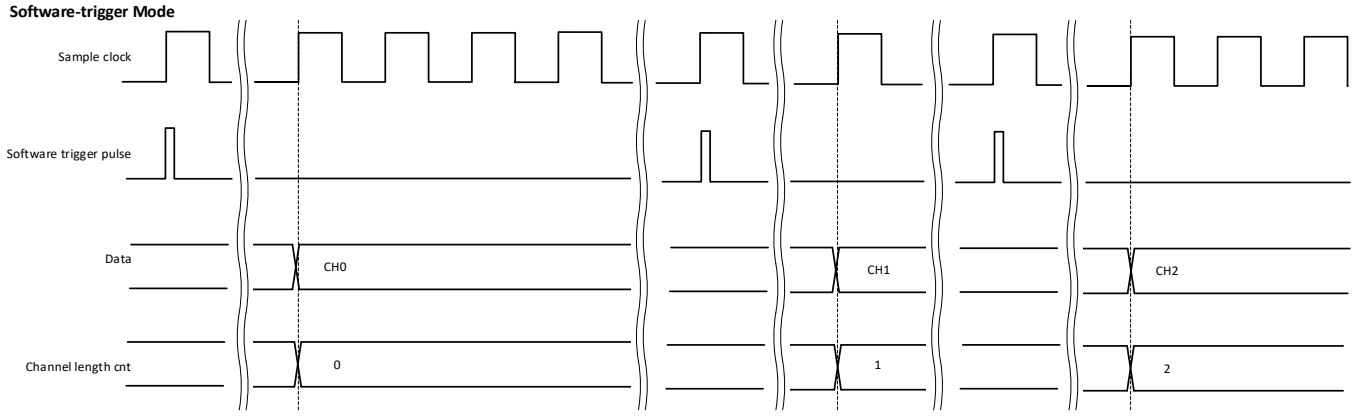


Figure 28-9 Software-trigger mode waveform

28.3.4.5 Comparator-assist Mode

To reduce ADC power consumption of itself, an additional comparator can cooperate with ADC as a first level check of input voltage. Once comparison result matches a predefined criterion, a wakeup signal can be delivered to ADC module by comparator. ADC module executes a single conversion to get a more precise voltage value.

All information provided in this document is subject to legal disclaimers.
© REALTEK 2024. All rights reserved.

In RTL8721Dx, three different comparator-match events can be detected to wakeup ADC with two analog comparators (see *Figure 28-10*). Each channel can set individual low threshold (Vref_l)/high threshold (Vref_h) and individual match criterion by register REG_COMP_REF_CH1/2/3 and register REG_COMP_INTR_CTRL.

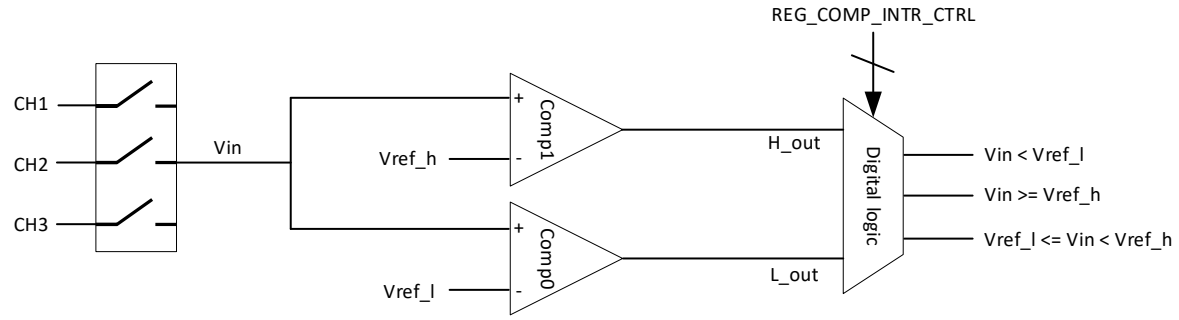


Figure 28-10 Comparator diagram

NOTE

Comparator can only detect CH1, CH2 and CH3.

The comparator-assist mode is usually used in the case of low power consumption with real-time monitoring. For example, when system is in sleep mode, most modules of chip are in power-down mode except comparator. At this time, comparator can monitor the external voltage in a low-power state. If the external voltage is lower or higher than predefined threshold, comparator match event occurs which will wakeup ADC for sampling and acquire the current voltage value. By the way, a comparator match event also can wakeup system from sleep mode.

Figure 28-11 shows the simplified comparator-assist mode waveform.

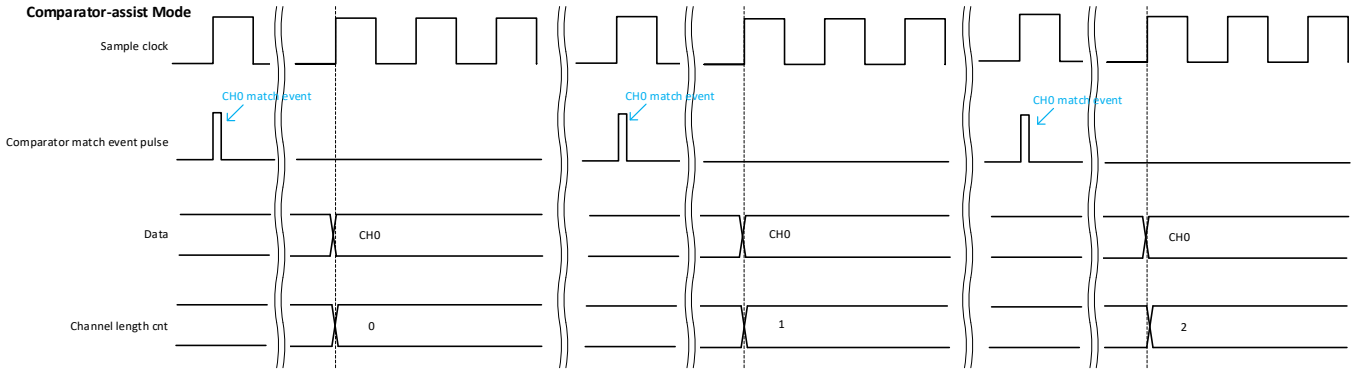


Figure 28-11 Simplified comparator-assist mode waveform

Table 28-3 shows the differences and applications of operation modes.

Table 28-3 Differences and applications of operation modes

Operation mode	Configuration	Difference	Applications
Auto mode	REG_ADC_CONF[3:1] = 1	● Work without system software, core and other modules ● Real-time detection with more consumption	The case that needs real-time detection and doesn't care about consumption.
Timer-trigger mode	REG_ADC_CONF[3:1] = 2	● Work with timer module without system software ● Less consumption	The case that needs periodic monitoring and less consumption.
Software-trigger mode	REG_ADC_SW_TRIG[0] = 1	Need software but be more flexible	The case that flexibly triggers ADC according to the current environment.
Comparator-assist mode	REG_ADC_CONF[3:1] = 3	● Work with comparator and less consumption ● Real-time detection	The case that needs real-time detection and less consumption.

28.4 Registers

Name	Address offset	Access	Description
REG_ADC_CONF	000h	R/W	ADC Configuration Register

All information provided in this document is subject to legal disclaimers.
© REALTEK 2024. All rights reserved.

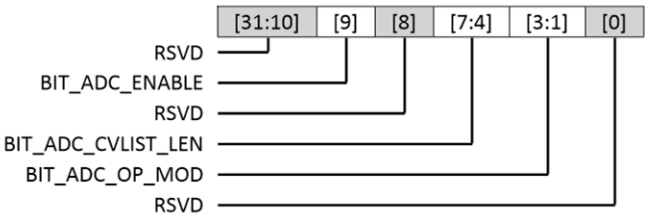
REG_ADC_IN_TYPE	004h	R/W	ADC Input Type Register
REG_ADC_COMP_TH_CH0	008h	R/W	ADC Channel 0 Comparison Threshold Register
REG_ADC_COMP_TH_CH1	00Ch	R/W	ADC Channel 1 Comparison Threshold Register
REG_ADC_COMP_TH_CH2	010h	R/W	ADC Channel 2 Comparison Threshold Register
REG_ADC_COMP_TH_CH3	014h	R/W	ADC Channel 3 Comparison Threshold Register
REG_ADC_COMP_TH_CH4	018h	R/W	ADC Channel 4 Comparison Threshold Register
REG_ADC_COMP_TH_CH5	01Ch	R/W	ADC Channel 5 Comparison Threshold Register
REG_ADC_COMP_TH_CH6	020h	R/W	ADC Channel 6 Comparison Threshold Register
REG_ADC_COMP_TH_CH7	024h	R/W	ADC Channel 7 Comparison Threshold Register
REG_ADC_COMP_TH_CH8	028h	R/W	ADC Channel 8 Comparison Threshold Register
REG_ADC_COMP_TH_CH9	02Ch	R/W	ADC Channel 9 Comparison Threshold Register
REG_ADC_COMP_TH_CH10	030h	R/W	ADC Channel 10 Comparison Threshold Register
REG_ADC_COMP_CTRL	034h	R/W	ADC Comparison Control Register
REG_ADC_COMP_STS	038h	R	ADC Comparison Status Register
REG_ADC_CHSW_LIST_0	03Ch	R/W	ADC Channel Switch List 0 Register
REG_ADC_CHSW_LIST_1	040h	R/W	ADC Channel Switch List 1 Register
REG_ADC_AUTO_CSW_CTRL	048h	R/W	ADC Automatic Channel Switch Control Register
REG_ADC_SW_TRIG	04Ch	R/W	ADC Software Trigger Register
REG_ADC_LAST_CH	050h	R	ADC Last Channel Register
REG_ADC_BUSY_STS	054h	R	ADC Busy Status Register
REG_ADC_INTR_CTRL	058h	R/W	ADC Interrupt Control Register
REG_ADC_INTR_RAW_STS	05Ch	R	ADC Interrupt Raw Status Register
REG_ADC_INTR_STS	060h	R/W	ADC Interrupt Status Register
REG_ADC_IT_CHNO_CON	064h	R/W	ADC Interrupt Channel Number Configuration Register
REG_ADC_FULL_LVL	068h	R/W	ADC FIFO Full Threshold Register
REG_ADC_EXT_TRIG_TIMER_SEL	06Ch	R/W	ADC External Trigger Timer Select Register
REG_ADC_DATA_CH0	070h	R	ADC Channel 0 Data Register
REG_ADC_DATA_CH1	074h	R	ADC Channel 1 Data Register
REG_ADC_DATA_CH2	078h	R	ADC Channel 2 Data Register
REG_ADC_DATA_CH3	07Ch	R	ADC Channel 3 Data Register
REG_ADC_DATA_CH4	080h	R	ADC Channel 4 Data Register
REG_ADC_DATA_CH5	084h	R	ADC Channel 5 Data Register
REG_ADC_DATA_CH6	088h	R	ADC Channel 6 Data Register
REG_ADC_DATA_CH7	08Ch	R	ADC Channel 7 Data Register
REG_ADC_DATA_CH8	090h	R	ADC Channel 8 Data Register
REG_ADC_DATA_CH9	094h	R	ADC Channel 9 Data Register
REG_ADC_DATA_CH10	098h	R	ADC Channel 10 Data Register
REG_ADC_DATA_GLOBAL	09Ch	R	ADC Global Data Register
REG_ADC_FLR	0A0h	R	ADC FIFO Level Register
REG_ADC_CLR_FIFO	0A4h	R/W	ADC Clear FIFO Register
REG_ADC_CLK_DIV	0A8h	R/W	ADC Clock Divider Register
REG_ADC_OVERSAMPLE	0ACh	R/W	ADC Oversample Register
REG_ADC_PWR_CTRL	0B0h	R/W	ADC Power Control Register
REG_ADC_SW_TMODE	0B4h	R/W	ADC Software Test Mode
REG_ADC_DUMMY	0B8h	R/W	ADC Dummy Register

28.4.1.1 REG_ADC_CONF

- **Name:** ADC Configuration Register
- **Size:** 32 bits
- **Address offset:** 000h
- **Read/write access:** R/W

The basic configuration of ADC includes enable and operation mode.

All information provided in this document is subject to legal disclaimers.
© REALTEK 2024. All rights reserved.

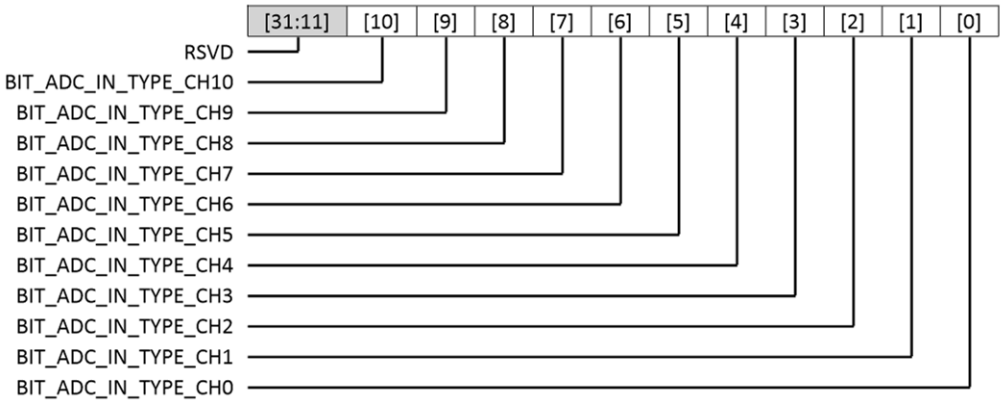


Bit	Symbol	Access	Reset	Description
31:10	RSVD	-	0	Reserved
9	BIT_ADC_ENABLE	R/W	0h	This bit is for ADC enable control
8	RSVD	-	0	Reserved
7:4	BIT_ADC_CVLIST_LEN	R/W	0h	This field defines the number of items in the ADC conversion channel list.
3:1	BIT_ADC_OP_MOD	R/W	0h	These bits selects ADC operation mode. 0: software trigger mode. 1: automatic mode 2: timer-trigger mode. 3: comparator-assist mode
0	RSVD	-	0	Reserved

28.4.1.2 REG_ADC_IN_TYPE

- **Name:** ADC Input Type Register
- **Size:** 32 bits
- **Address offset:** 004h
- **Read/write access:** R/W

This register is used to configure analog ADC input type.



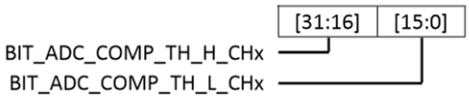
Bit	Symbol	Access	Reset	Description
31:11	RSVD	-	0	Reserved
10	BIT_ADC_IN_TYPE_CH10	R	0	This bit reflects which kind of input type channel 10 is. Since channel 10 is a fixed internal channel.
9	BIT_ADC_IN_TYPE_CH9	R	0	This bit reflects which kind of input type channel 9 is. Since channel 9 is a fixed internal channel.
8	BIT_ADC_IN_TYPE_CH8	R	0	This bit reflects which kind of input type channel 8 is. Since channel 8 is a fixed internal channel.
7	BIT_ADC_IN_TYPE_CH7	R	0	This bit reflects which kind of input type channel 7 is. Since channel 7 is a fixed internal channel.
6	BIT_ADC_IN_TYPE_CH6	R	0	This bit reflects which kind of input type channel 6 is. Since channel 6 is a fixed single-end input channel.
5	BIT_ADC_IN_TYPE_CH5	R/W	0	This bit controls channel 5 input type. If it's set to differential types (value of this bit is 1), channel 4 should also be set to differential type since they are hardware-fixed differential pair. 0: ADC channel 5 is single-end input.

All information provided in this document is subject to legal disclaimers.
© REALTEK 2024. All rights reserved.

				1: ADC channel 5 is differential input which is paired with channel 4.
4	BIT_ADC_IN_TYPE_CH4	R/W	0	This bit controls channel 4 input type. If it's set to differential types (value of this bit is 1), channel 5 should also be set to differential type since they are hardware-fixed differential pair. 0: ADC channel 4 is single-end input. 1: ADC channel 4 is differential input which is paired with channel 5.
3	BIT_ADC_IN_TYPE_CH3	R/W	0	This bit controls channel 3 input type. If it's set to differential types (value of this bit is 1), channel 2 should also be set to differential type since they are hardware-fixed differential pair. 0: ADC channel 3 is single-end input. 1: ADC channel 3 is differential input which is paired with channel 2.
2	BIT_ADC_IN_TYPE_CH2	R/W	0	This bit controls channel 2 input type. If it's set to differential types (value of this bit is 1), channel 3 should also be set to differential type since they are hardware-fixed differential pair. 0: ADC channel 2 is single-end input. 1: ADC channel 2 is differential input which is paired with channel 3.
1	BIT_ADC_IN_TYPE_CH1	R/W	0	This bit controls channel 1 input type. If it's set to differential types (value of this bit is 1), channel 0 should also be set to differential type since they are hardware-fixed differential pair. 0: ADC channel 1 is single-end input. 1: ADC channel 1 is differential input which is paired with channel 0.
0	BIT_ADC_IN_TYPE_CH0	R/W	0	This bit controls channel 0 input type. If it's set to differential types (value of this bit is 1), channel 1 should also be set to differential type since they are hardware-fixed differential pair. 0: ADC channel 0 is single-end input. 1: ADC channel 0 is differential input which is paired with channel 1.

28.4.1.3 REG_ADC_COMP_TH_CHx

- **Name:** ADC Channel x Comparison Threshold Register (x={0, 1, 2, 3, 4, 5, 6, 7, 8, 9, 10})
- **Size:** 32 bits
- **Address offset:**
  - **REG_ADC_COMP_TH_CH0:** 008h
  - **REG_ADC_COMP_TH_CH1:** 00Ch
  - **REG_ADC_COMP_TH_CH2:** 010h
  - **REG_ADC_COMP_TH_CH3:** 014h
  - **REG_ADC_COMP_TH_CH4:** 018h
  - **REG_ADC_COMP_TH_CH5:** 01Ch
  - **REG_ADC_COMP_TH_CH6:** 020h
  - **REG_ADC_COMP_TH_CH7:** 024h
  - **REG_ADC_COMP_TH_CH8:** 028h
  - **REG_ADC_COMP_TH_CH9:** 02Ch
  - **REG_ADC_COMP_TH_CH10:** 030h
- **Read/write access:** R/W



Bit	Symbol	Access	Reset	Description
31:16	BIT_ADC_COMP_TH_H_CHx	R/W	0h	This field defines the higher threshold of channel x for ADC automatic comparison.
15:0	BIT_ADC_COMP_TH_L_CHx	R/W	0h	This field defines the lower threshold of channel x for ADC automatic comparison.

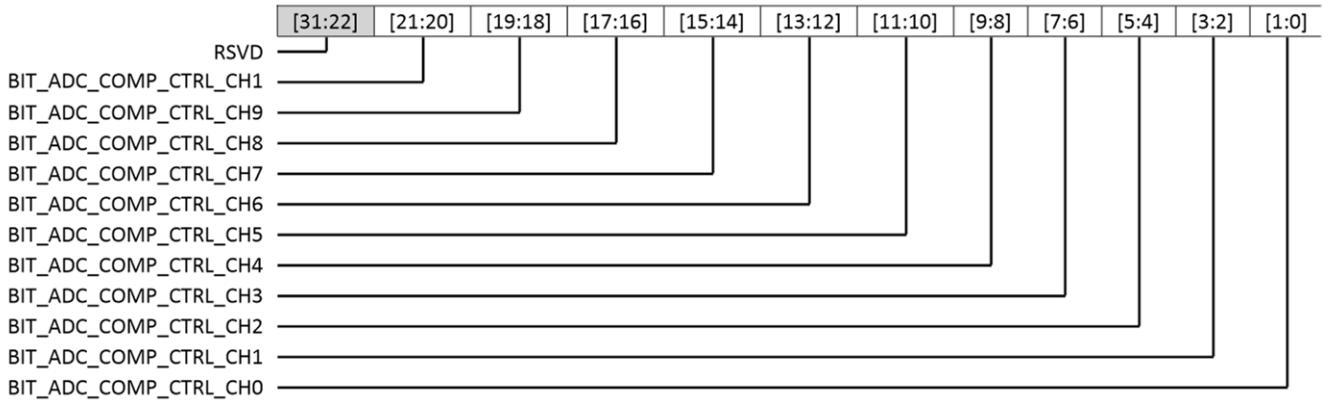
28.4.1.4 REG_ADC_COMP_CTRL

- **Name:** ADC Comparison Control Register
- **Size:** 32 bits
- **Address offset:** 034h

All information provided in this document is subject to legal disclaimers.
© REALTEK 2024. All rights reserved.

- **Read/write access:** R/W

This is channel comparison criterion control register, which can be used to control every channel's comparison criterion. There are 3 kinds of criterions.



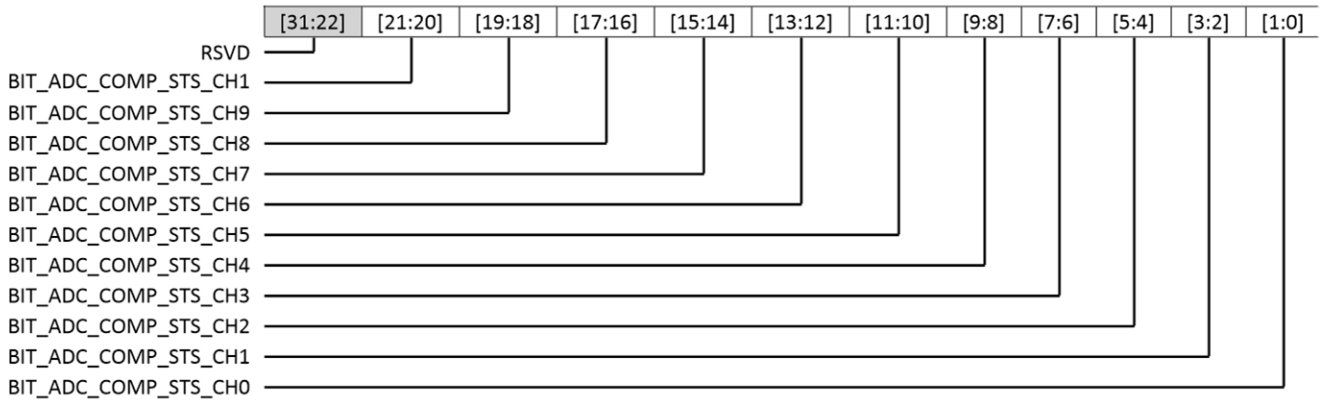
Bit	Symbol	Access	Reset	Description
31:22	RSVD	-	0	Reserved
21:20	BIT_ADC_COMP_CTRL_CH10	R/W	0h	This field defines ADC channel comparison criteria which would notify system by interrupt when the criterion matches. Once a criterion matches, ADC should send an interrupt signal to system if the related interrupt mask is unmask. ADC also updates comparison results in reg_adc_comp_sts. 0: When Vin < bit_adc_comp_th_l_chx, comparison criterion matches. 1: When Vin > bit_adc_comp_th_h_chx, comparison criterion matches. 2: When Vin >= bit_adc_comp_th_l_chx and Vin <= bit_adc_comp_th_h_chx, comparison criterion matches. 3: When Vin < bit_adc_comp_th_l_chx or Vin > bit_adc_comp_th_h_chx, comparison criterion matches.
19:18	BIT_ADC_COMP_CTRL_CH9	R/W	0h	The function is the same as BIT_ADC_COMP_CTRL_CH10.
17:16	BIT_ADC_COMP_CTRL_CH8	R/W	0h	The function is the same as BIT_ADC_COMP_CTRL_CH10.
15:14	BIT_ADC_COMP_CTRL_CH7	R/W	0h	The function is the same as BIT_ADC_COMP_CTRL_CH10.
13:12	BIT_ADC_COMP_CTRL_CH6	R/W	0h	The function is the same as BIT_ADC_COMP_CTRL_CH10.
11:10	BIT_ADC_COMP_CTRL_CH5	R/W	0h	The function is the same as BIT_ADC_COMP_CTRL_CH10.
9:8	BIT_ADC_COMP_CTRL_CH4	R/W	0h	The function is the same as BIT_ADC_COMP_CTRL_CH10.
7:6	BIT_ADC_COMP_CTRL_CH3	R/W	0h	The function is the same as BIT_ADC_COMP_CTRL_CH10.
5:4	BIT_ADC_COMP_CTRL_CH2	R/W	0h	The function is the same as BIT_ADC_COMP_CTRL_CH10.
3:2	BIT_ADC_COMP_CTRL_CH1	R/W	0h	The function is the same as BIT_ADC_COMP_CTRL_CH10.
1:0	BIT_ADC_COMP_CTRL_CH0	R/W	0h	The function is the same as BIT_ADC_COMP_CTRL_CH10.

28.4.1.5 REG_ADC_COMP_STS

- **Name:** ADC Comparison Status Register
- **Size:** 32 bits
- **Address offset:** 038h
- **Read/write access:** R

This is the register about comparison status of every channel.

All information provided in this document is subject to legal disclaimers.
© REALTEK 2024. All rights reserved.

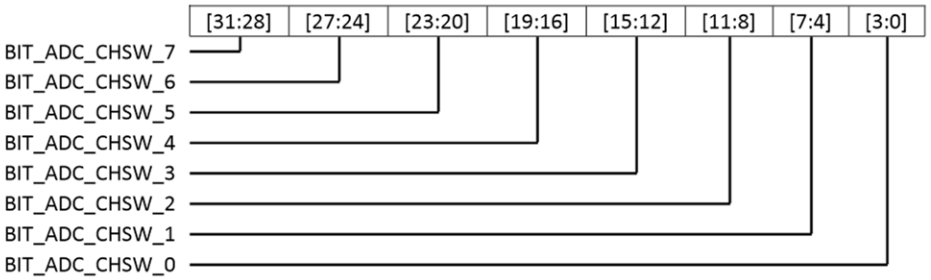


Bit	Symbol	Access	Reset	Description
31:22	RSVD	-	0	Reserved
21:20	BIT_ADC_COMP_STS_CH10	R	0h	This field reflects the comparison result of channel 10 immediately
19:18	BIT_ADC_COMP_STS_CH9	R	0h	This field reflects the comparison result of channel 9 immediately
17:16	BIT_ADC_COMP_STS_CH8	R	0h	This field reflects the comparison result of channel 8 immediately
15:14	BIT_ADC_COMP_STS_CH7	R	0h	This field reflects the comparison result of channel 7 immediately
13:12	BIT_ADC_COMP_STS_CH6	R	0h	This field reflects the comparison result of channel 6 immediately
11:10	BIT_ADC_COMP_STS_CH5	R	0h	This field reflects the comparison result of channel 5 immediately
9:8	BIT_ADC_COMP_STS_CH4	R	0h	This field reflects the comparison result of channel 4 immediately
7:6	BIT_ADC_COMP_STS_CH3	R	0h	This field reflects the comparison result of channel 3 immediately
5:4	BIT_ADC_COMP_STS_CH2	R	0h	This field reflects the comparison result of channel 2 immediately
3:2	BIT_ADC_COMP_STS_CH1	R	0h	This field reflects the comparison result of channel 1 immediately
1:0	BIT_ADC_COMP_STS_CH0	R	0h	This field reflects the comparison result of channel 0 immediately

28.4.1.6 REG_ADC_CHSW_LIST_0

- **Name:** ADC Channel Switch List 0 Register
- **Size:** 32 bits
- **Address offset:** 03Ch
- **Read/write access:** R/W

Software channel list 0, used to define the analog channels to own channel list.



Bit	Symbol	Access	Reset	Description
31:28	BIT_ADC_CHSW_7	R/W	0h	This field defines the number 7 channel to do a conversion when a conversion event takes place.
27:24	BIT_ADC_CHSW_6	R/W	0h	This field defines the number 6 channel to do a conversion when a conversion event takes place.
23:20	BIT_ADC_CHSW_5	R/W	0h	This field defines the number 5 channel to do a conversion when a conversion event takes place.
19:16	BIT_ADC_CHSW_4	R/W	0h	This field defines the number 4 channel to do a conversion when a conversion event takes place.
15:12	BIT_ADC_CHSW_3	R/W	0h	This field defines the number 3 channel to do a conversion when a conversion event takes place.

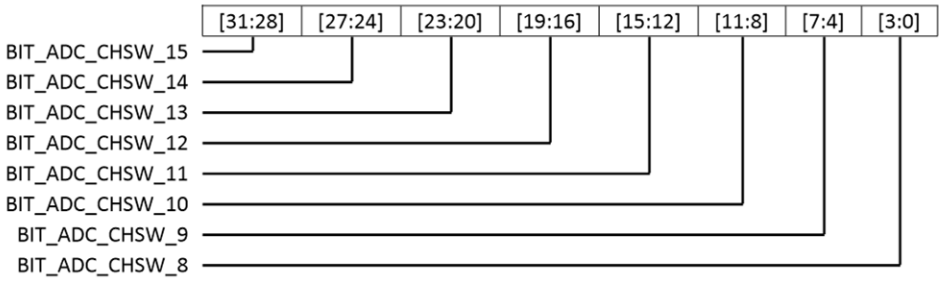
All information provided in this document is subject to legal disclaimers.
© REALTEK 2024. All rights reserved.

11:8	BIT_ADC_CHSW_2	R/W	0h	This field defines the number 2 channel to do a conversion when a conversion event takes place.
7:4	BIT_ADC_CHSW_1	R/W	0h	This field defines the number 1 channel to do a conversion when a conversion event takes place.
3:0	BIT_ADC_CHSW_0	R/W	0h	This field defines the number 0 channel to do a conversion when a conversion event takes place.

28.4.1.7 REG_ADC_CHSW_LIST_1

- **Name:** ADC Channel Switch List 1 Register
- **Size:** 32 bits
- **Address offset:** 040h
- **Read/write access:** R/W

Software channel list 1, refer to REG_ADC_CHSW_LIST_0.



Bit	Symbol	Access	Reset	Description
31:28	BIT_ADC_CHSW_15	R/W	0h	This field defines the number 15 channel to do a conversion when a conversion event takes place.
27:24	BIT_ADC_CHSW_14	R/W	0h	This field defines the number 14 channel to do a conversion when a conversion event takes place.
23:20	BIT_ADC_CHSW_13	R/W	0h	This field defines the number 13 channel to do a conversion when a conversion event takes place.
19:16	BIT_ADC_CHSW_12	R/W	0h	This field defines the number 12 channel to do a conversion when a conversion event takes place.
15:12	BIT_ADC_CHSW_11	R/W	0h	This field defines the number 11 channel to do a conversion when a conversion event takes place.
11:8	BIT_ADC_CHSW_10	R/W	0h	This field defines the number 10 channel to do a conversion when a conversion event takes place.
7:4	BIT_ADC_CHSW_9	R/W	0h	This field defines the number 9 channel to do a conversion when a conversion event takes place.
3:0	BIT_ADC_CHSW_8	R/W	0h	This field defines the number 8 channel to do a conversion when a conversion event takes place.

28.4.1.8 REG_ADC_AUTO_CSW_CTRL

- **Name:** ADC Automatic Channel Switch Control Register
- **Size:** 32 bits
- **Address offset:** 048h
- **Read/write access:** R/W

This is the register about automatic switch channel function.



Bit	Symbol	Access	Reset	Description
31:1	RSVD	-	0	Reserved

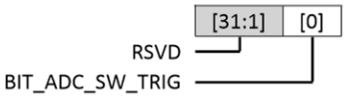
All information provided in this document is subject to legal disclaimers.
© REALTEK 2024. All rights reserved.

0	BIT_ADC_AUTOSW_EN	R/W	0h	This bit controls the automatic channel switch enabled or disabled. 0: Disable the automatic channel switch. If an automatic channel switch is in process, writing 0 will terminate the automatic channel switch. 1: Enable the automatic channel switch. When setting this bit, an automatic channel switch starts from the first channel in the channel switch list.

28.4.1.9 REG_ADC_SW_TRIG

- **Name:** ADC Software Trigger Register
- **Size:** 32 bits
- **Address offset:** 04Ch
- **Read/write access:** R/W

Software controls to trigger the channel switch.



Bit	Symbol	Access	Reset	Description
31:1	RSVD	-	0	Reserved
0	BIT_ADC_SW_TRIG	R/W	0h	This bit controls the ADC module to do a conversion. Every time this bit is set to 1, ADC module would switch to a new channel and do one conversion. Therefore, this bit could be used as a start-convert event which is controlled by software. Every time a conversion is done, software MUST clear this bit manually. The interval between clearing and restart this bit must exceed one sample clock period. 0: Disable the analog module and analog mux. 1: Enable the analog module and analog mux, and then start a new channel conversion.

28.4.1.10 REG_ADC_LAST_CH

- **Name:** ADC Last Channel Register
- **Size:** 32 bits
- **Address offset:** 050h
- **Read/write access:** R

This register records the last channel.



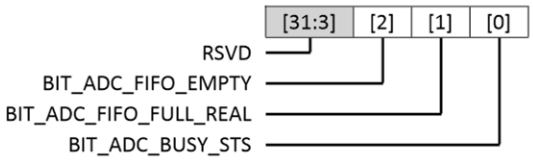
Bit	Symbol	Access	Reset	Description
31:4	RSVD	-	0	Reserved
3:0	BIT_ADC_LAST_CH	R	0h	This field reflects the last used channel.

28.4.1.11 REG_ADC_BUSY_STS

- **Name:** ADC Busy Status Register
- **Size:** 32 bits
- **Address offset:** 054h
- **Read/write access:** R

This register reflects the ADC FSM and FIFO status.

All information provided in this document is subject to legal disclaimers.
© REALTEK 2024. All rights reserved.

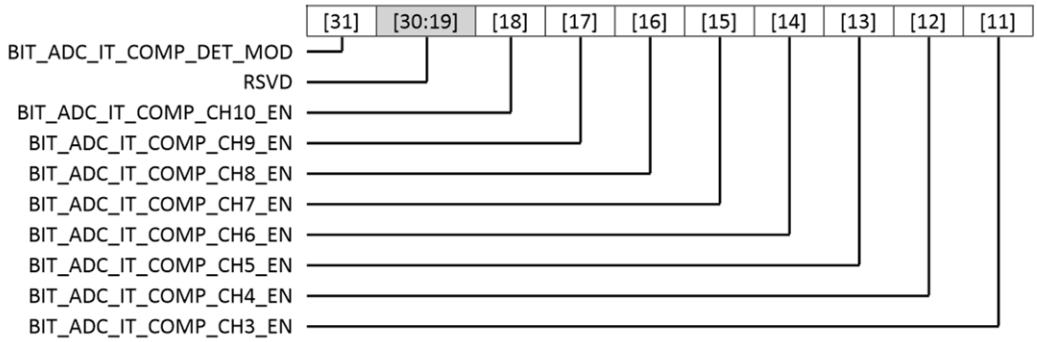


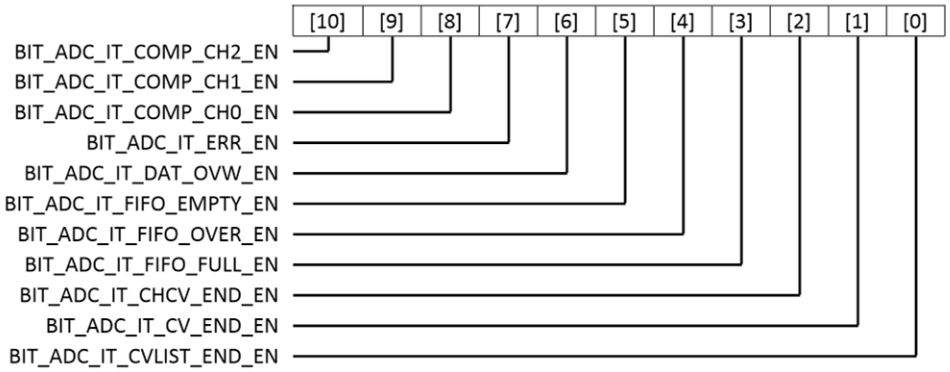
Bit	Symbol	Access	Reset	Description
31:3	RSVD	-	0	Reserved
2	BIT_ADC_FIFO_EMPTY	R	1	0: FIFO in ADC is not empty. 1: FIFO in ADC is empty.
1	BIT_ADC_FIFO_FULL_REAL	R	0	0: FIFO in ADC is not real full. 1: FIFO in ADC is real full.
0	BIT_ADC_BUSY_STS	R	0	This bit reflects the ADC is busy or not. If the ADC is processing a conversion of a channel, this bit remains 1 which indicates it's busy. Once a conversion is done, this bit becomes 0 which indicates it's ready to do another conversion. 0: The ADC is ready. 1: The ADC is busy.

28.4.1.12 REG_ADC_INTR_CTRL

- **Name:** ADC Interrupt Control Register
- **Size:** 32 bits
- **Address offset:** 058h
- **Read/write access:** R/W

This register is used to enable or disable the different kinds of interrupts.





Bit	Symbol	Access	Reset	Description
31	BIT_ADC_IT_COMP_DET_MOD	R/W	0h	This bit controls the interrupt BIT_ADC_IT_COMP_CHx operation. Comparison criterion matching is rising edge. 0: The detection mode is level detection. When the comparison criterion is match, BIT_ADC_IT_COMP_CHx will send interrupt continuously. 1: The detection mode is edge detection. When the comparison criterion is match, BIT_ADC_IT_COMP_CHx will send interrupt once at the time of

All information provided in this document is subject to legal disclaimers.
© REALTEK 2024. All rights reserved.

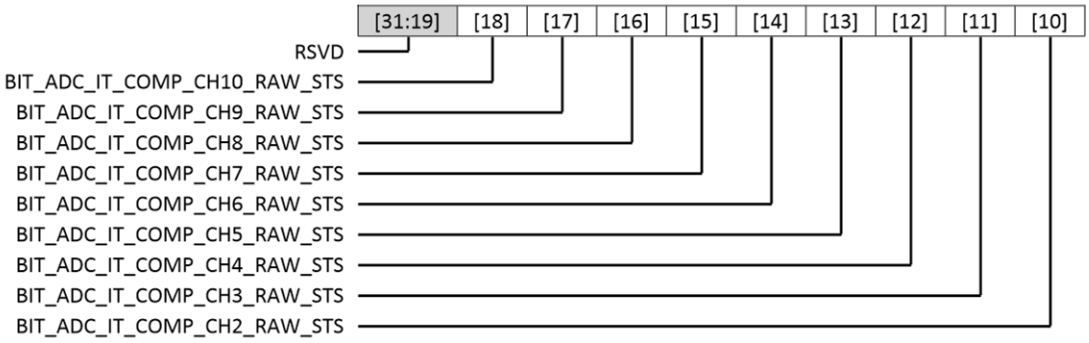
				matching.
30:19	RSVD	-	0	Reserved
18	BIT_ADC_IT_COMP_CH10_EN	R/W	0h	The function is the same as BIT_ADC_IT_COMP_CH0_EN
17	BIT_ADC_IT_COMP_CH9_EN	R/W	0h	The function is the same as BIT_ADC_IT_COMP_CH0_EN
16	BIT_ADC_IT_COMP_CH8_EN	R/W	0h	The function is the same as BIT_ADC_IT_COMP_CH0_EN
15	BIT_ADC_IT_COMP_CH7_EN	R/W	0h	The function is the same as BIT_ADC_IT_COMP_CH0_EN
14	BIT_ADC_IT_COMP_CH6_EN	R/W	0h	The function is the same as BIT_ADC_IT_COMP_CH0_EN
13	BIT_ADC_IT_COMP_CH5_EN	R/W	0h	The function is the same as BIT_ADC_IT_COMP_CH0_EN
12	BIT_ADC_IT_COMP_CH4_EN	R/W	0h	The function is the same as BIT_ADC_IT_COMP_CH0_EN
11	BIT_ADC_IT_COMP_CH3_EN	R/W	0h	The function is the same as BIT_ADC_IT_COMP_CH0_EN
10	BIT_ADC_IT_COMP_CH2_EN	R/W	0h	The function is the same as BIT_ADC_IT_COMP_CH0_EN
9	BIT_ADC_IT_COMP_CH1_EN	R/W	0h	The function is the same as BIT_ADC_IT_COMP_CH0_EN
8	BIT_ADC_IT_COMP_CH0_EN	R/W	0h	This bit controls the interrupt is enabled or not when channel 0 comparison criterion matches. 0: This interrupt is disabled. 1: This interrupt is enabled.
7	BIT_ADC_IT_ERR_EN	R/W	0h	This bit controls the interrupt is enabled or not when an error state takes place. 0: This interrupt is disabled. 1: This interrupt is enabled.
6	BIT_ADC_IT_DAT_OVW_EN	R/W	0h	This bit controls the interrupt is enabled or not when a data overwritten situation takes place. 0: This interrupt is disabled. 1: This interrupt is enabled.
5	BIT_ADC_IT_FIFO_EMPTY_EN	R/W	0h	This bit controls the interrupt is enabled or not when a FIFO empty state takes place. 0: This interrupt is disabled. 1: This interrupt is enabled.
4	BIT_ADC_IT_FIFO_OVER_EN	R/W	0h	This bit controls the interrupt is enabled or not when a FIFO overflow state takes place. 0: This interrupt is disabled. 1: This interrupt is enabled.
3	BIT_ADC_IT_FIFO_FULL_EN	R/W	0h	This bit controls the interrupt is enabled or not when a FIFO full state takes place. 0: This interrupt is disabled. 1: This interrupt is enabled.
2	BIT_ADC_IT_CHCV_END_EN	R/W	0h	This bit controls the interrupt is enabled or not when a particular channel conversion is done. refer to reg_adc_it_chno_con. 0: This interrupt is disabled. 1: This interrupt is enabled.
1	BIT_ADC_IT_CV_END_EN	R/W	0h	This bit controls the interrupt is enabled or not every time a conversion is done. No matter ADC module is in what kind of operation mode. Every time a conversion is executed, ADC module would notify system if This bit is set. 0: This interrupt is disabled. 1: This interrupt is enabled.
0	BIT_ADC_IT_CVLIST_END_EN	R/W	0h	This bit controls the interrupt is enabled or not when a conversion of the last channel in the list is done. For example, in automatic mode conversions would be executed continuously. Every time the last channel conversion is done, which means all channel conversions in the list is done, ADC could notify system if This bit is set. 0: This interrupt is disabled. 1: This interrupt is enabled.

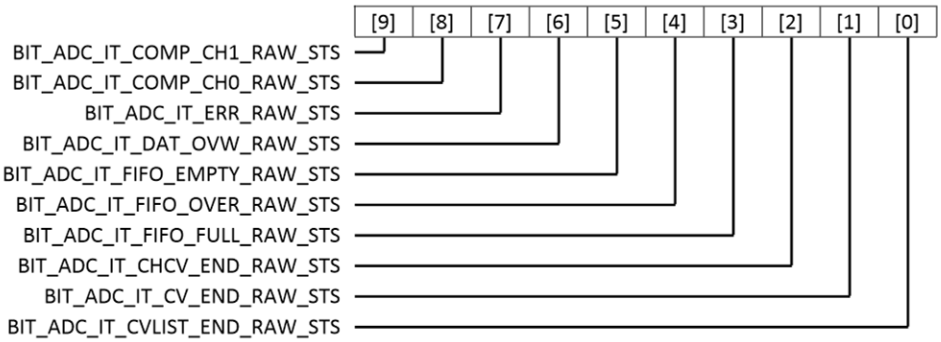
28.4.1.13 REG_ADC_INTR_RAW_STS

- **Name:** ADC Interrupt Raw Status Register
- **Size:** 32 bits
- **Address offset:** 05Ch
- **Read/write access:** R

All information provided in this document is subject to legal disclaimers.
© REALTEK 2024. All rights reserved.

This register is used to record the different kinds of interrupt raw status (bypass the enable register's control).





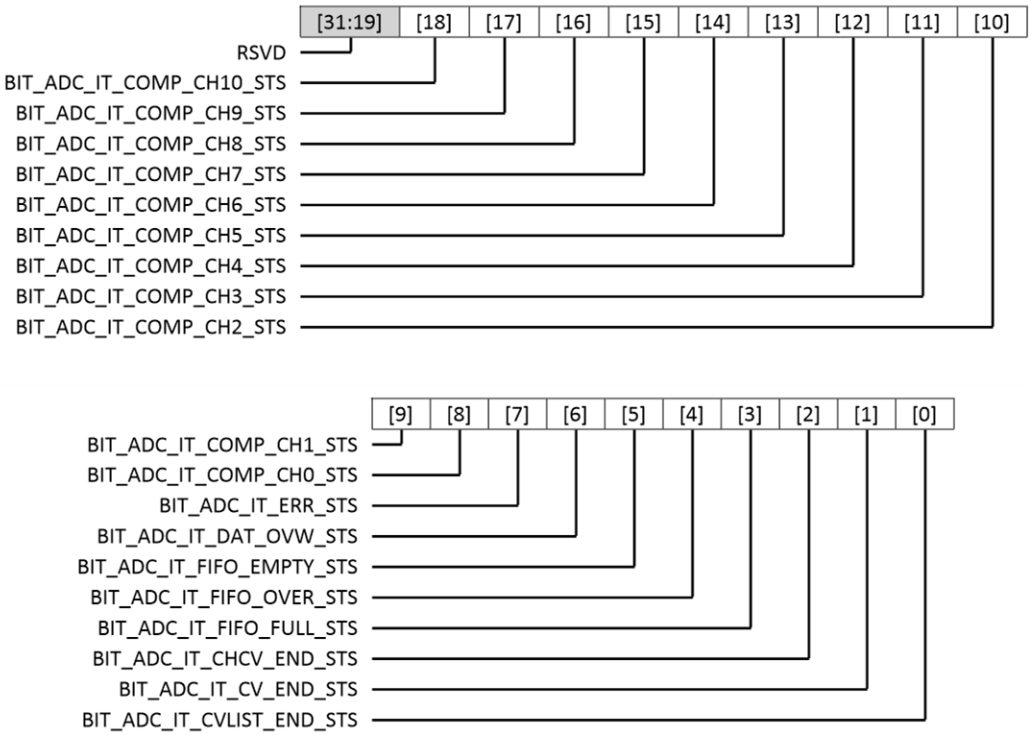
Bit	Symbol	Access	Reset	Description
31:19	RSVD	-	0	Reserved
18	BIT_ADC_IT_COMP_CH10_RAW_STS	R	0h	
17	BIT_ADC_IT_COMP_CH9_RAW_STS	R	0h	
16	BIT_ADC_IT_COMP_CH8_RAW_STS	R	0h	
15	BIT_ADC_IT_COMP_CH7_RAW_STS	R	0h	
14	BIT_ADC_IT_COMP_CH6_RAW_STS	R	0h	
13	BIT_ADC_IT_COMP_CH5_RAW_STS	R	0h	
12	BIT_ADC_IT_COMP_CH4_RAW_STS	R	0h	
11	BIT_ADC_IT_COMP_CH3_RAW_STS	R	0h	
10	BIT_ADC_IT_COMP_CH2_RAW_STS	R	0h	
9	BIT_ADC_IT_COMP_CH1_RAW_STS	R	0h	
8	BIT_ADC_IT_COMP_CH0_RAW_STS	R	0h	
7	BIT_ADC_IT_ERR_RAW_STS	R	0h	
6	BIT_ADC_IT_DAT_OVW_RAW_STS	R	0h	
5	BIT_ADC_IT_FIFO_EMPTY_RAW_STS	R	1h	
4	BIT_ADC_IT_FIFO_OVER_RAW_STS	R	0h	
3	BIT_ADC_IT_FIFO_FULL_RAW_STS	R	0h	
2	BIT_ADC_IT_CHCV_END_RAW_STS	R	0h	
1	BIT_ADC_IT_CV_END_RAW_STS	R	0h	
0	BIT_ADC_IT_CVLIST_END_RAW_STS	R	0h	

28.4.1.14 REG_ADC_INTR_STS

- **Name:** ADC Interrupt Status Register
- **Size:** 32 bits
- **Address offset:** 060h
- **Read/write access:** R/W

This register is used to record the different kinds of interrupt status.

All information provided in this document is subject to legal disclaimers.
© REALTEK 2024. All rights reserved.



Bit	Symbol	Access	Reset	Description
31:19	RSVD	-	0	Reserved
18	BIT_ADC_IT_COMP_CH10_STS	RW1C	0h	
17	BIT_ADC_IT_COMP_CH9_STS	RW1C	0h	
16	BIT_ADC_IT_COMP_CH8_STS	RW1C	0h	
15	BIT_ADC_IT_COMP_CH7_STS	RW1C	0h	
14	BIT_ADC_IT_COMP_CH6_STS	RW1C	0h	
13	BIT_ADC_IT_COMP_CH5_STS	RW1C	0h	
12	BIT_ADC_IT_COMP_CH4_STS	RW1C	0h	
11	BIT_ADC_IT_COMP_CH3_STS	RW1C	0h	
10	BIT_ADC_IT_COMP_CH2_STS	RW1C	0h	
9	BIT_ADC_IT_COMP_CH1_STS	RW1C	0h	
8	BIT_ADC_IT_COMP_CH0_STS	RW1C	0h	
7	BIT_ADC_IT_ERR_STS	RW1C	0h	
6	BIT_ADC_IT_DAT_OVW_STS	RW1C	0h	
5	BIT_ADC_IT_FIFO_EMPTY_STS	RW1C	0h	
4	BIT_ADC_IT_FIFO_OVER_STS	RW1C	0h	
3	BIT_ADC_IT_FIFO_FULL_STS	RW1C	0h	
2	BIT_ADC_IT_CHCV_END_STS	RW1C	0h	
1	BIT_ADC_IT_CV_END_STS	RW1C	0h	
0	BIT_ADC_IT_CVLIST_END_STS	RW1C	0h	

28.4.1.15 REG_ADC_IT_CHNO_CON

- **Name:** ADC Interrupt Channel Number Configuration Register
- **Size:** 32 bits
- **Address offset:** 064h
- **Read/write access:** R/W

This register is about the specific channel to trigger interrupts.

All information provided in this document is subject to legal disclaimers.
© REALTEK 2024. All rights reserved.



Bit	Symbol	Access	Reset	Description
31:4	RSVD	-	0	Reserved
3:0	BIT_ADC_IT_CHNO_CON	R/W	0h	Channel number This field defines that ADC module should send interrupt signal to system when a conversion which of channel number is the same as this field. This register is only valid when bit_adc_it_chcv_end_en is set.

28.4.1.16 REG_ADC_FULL_LVL

- **Name:** ADC FIFO Full Threshold Register
- **Size:** 32 bits
- **Address offset:** 068h
- **Read/write access:** R/W



Bit	Symbol	Access	Reset	Description
31:6	RSVD	-	0	Reserved
5:0	BIT_ADC_FULL_LVL	R/W	0h	This field defines the FIFO full entry threshold level and it's used for FIFO full interrupt. It should be set according to ADC FIFO depth. A value of 0 sets the threshold for 1 entry, and a value of 15 sets the threshold for 16 entries.

28.4.1.17 REG_ADC_EXT_TRIG_TIMER_SEL

- **Name:** ADC External Trigger Timer Select Register
- **Size:** 32 bits
- **Address offset:** 06Ch
- **Read/write access:** R/W

This register is used to select the specific external timer source.



Bit	Symbol	Access	Reset	Description
31:3	RSVD	-	0	Reserved
2:0	BIT_ADC_EXT_TRIG_TIMER_SEL	R/W	0h	These three bits defines which timer channel would be used to make ADC module do a conversion. 0: Timer module 0 is used as the comparator external trigger source. 1: Timer module 1 is used as the comparator external trigger source. .. 7: Timer module 7 is used as the comparator external trigger source.

28.4.1.18 REG_ADC_DATA_CHx

- **Name:** ADC Channel x Data Register (x={0, 1, 2, 3, 4, 5, 6, 7, 8, 9, 10})
- **Size:** 32 bits
- **Address offset:**
  - **REG_ADC_DATA_CH0:** 070h
  - **REG_ADC_DATA_CH1:** 074h

All information provided in this document is subject to legal disclaimers.
© REALTEK 2024. All rights reserved.

- **REG_ADC_DATA_CH2:** 078h
- **REG_ADC_DATA_CH3:** 07Ch
- **REG_ADC_DATA_CH4:** 080h
- **REG_ADC_DATA_CH5:** 084h
- **REG_ADC_DATA_CH6:** 088h
- **REG_ADC_DATA_CH7:** 08Ch
- **REG_ADC_DATA_CH8:** 090h
- **REG_ADC_DATA_CH9:** 094h
- **REG_ADC_DATA_CH10:** 098h

- **Read/write access:** R

This register is used to store the channel data, ready and overwritten information.

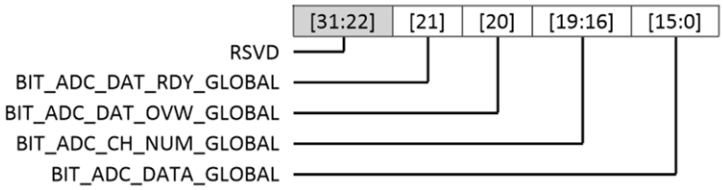


Bit	Symbol	Access	Reset	Description
31:22	RSVD	-	0	Reserved
21	BIT_ADC_DAT_RDY_CHx	R	0	This bit indicates that a conversion is done. Every time a conversion is done, this bit should be set to 1 and it would be cleared to 0 when a read operation of reg_adc_data_chx.
20	BIT_ADC_DAT_OVW_CHx	R	0	This bit indicates that there is a data overwritten situation in bit_adc_data_global taking place. A data overwritten situation is that a former conversion data is NOT read before a new conversion is written to data field. It would be cleared to 0 when a read operation of reg_adc_data_chx. 0: There is no data overwritten case. 1: There is a data overwritten case.
19:16	BIT_ADC_NUM_CHx	RP	0	This field contains the newest conversion data of channel x [19:16]: which channel in channel list the data belongs to, only valid when bit_adc_ch_unmask=1
15:0	BIT_ADC_DATA_CHx	RP	0	This field contains the newest conversion data of channel x [15:0]: newest data

28.4.1.19 REG_ADC_DATA_GLOBAL

- **Name:** ADC Global Data Register
- **Size:** 32 bits
- **Address offset:** 09Ch
- **Read/write access:** R

This register is used to store global data, ready, channel and overwritten information.



Bit	Symbol	Access	Reset	Description
31:22	RSVD	-	0	Reserved
21	BIT_ADC_DAT_RDY_GLOBAL	R	0	This bit indicates that a conversion is done. Every time a conversion is done, this bit should be set to 1 and it would be cleared to 0 when a read operation of reg_adc_data_global
20	BIT_ADC_DAT_OVW_GLOBAL	R	0	This bit indicates that there is a data overwritten situation in bit_adc_data_global taking place. A data overwritten situation is that a

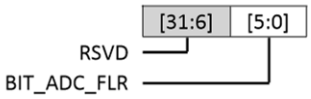
All information provided in this document is subject to legal disclaimers.
© REALTEK 2024. All rights reserved.

				former conversion data is NOT read before a new conversion is written to data field. 0: there is no data overwritten case. 1: there is a data overwritten case.
19:16	BIT_ADC_CH_NUM_GLOBAL	RP	0	This field contains the newest conversion data of channel in the list. [19:16]: which channel in channel list the data belongs to, only valid when bit_adc_ch_unmask=1
15:0	BIT_ADC_DATA_GLOBAL	RP	0	This field contains the newest conversion data of channel in the list. [15:0]: newest data

28.4.1.20 REG_ADC_FLR

- **Name:** ADC FIFO Level Register
- **Size:** 32 bits
- **Address offset:** 0A0h
- **Read/write access:** R

This register is used to store FIFO entry number.

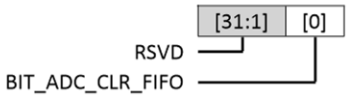


Bit	Symbol	Access	Reset	Description
31:6	RSVD	-	0	Reserved
5:0	BIT_ADC_FLR	R	0	This field records the current ADC FIFO entry number.

28.4.1.21 REG_ADC_CLR_FIFO

- **Name:** ADC Clear FIFO Register
- **Size:** 32 bits
- **Address offset:** 0A4h
- **Read/write access:** R/W

This register is used to clear FIFO.



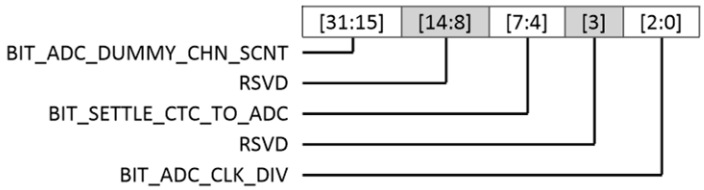
Bit	Symbol	Access	Reset	Description
31:1	RSVD	-	0	Reserved
0	BIT_ADC_CLR_FIFO	R/W	0	This bit is used to clear FIFO contents and makes it to the default status. Every time this bit is set to 1. FIFO would be cleared to the default status. Software should set this bit to 0 after FIFO is cleared. 0: No effect 1: Clear FIFO

28.4.1.22 REG_ADC_CLK_DIV

- **Name:** ADC Clock Divider Register
- **Size:** 32 bits
- **Address offset:** 0A8h
- **Read/write access:** R/W

This register is used to control clock dividend and channel unmask.

All information provided in this document is subject to legal disclaimers.
© REALTEK 2024. All rights reserved.

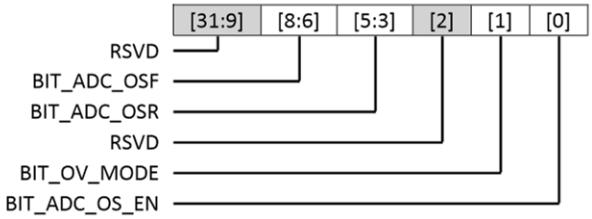


Bit	Symbol	Access	Reset	Description
31:15	BIT_ADC_DUMMY_CHN_SCNT	R/W	0h	Dummy channel time counter. In auto mode, ADC shares the dummy channel for Cap-Touch. This register controls the time of dummy channel, the unit is 4MHz.
14:8	RSVD	-	0	Reserved
7:4	BIT_SETTLE_CTC_TO_ADC	R/W	0h	Settle count for sample data from Cap-Touch to ADC, the unit is ADC sample clock. 0: means no need to settle
3	RSVD	-	0	Reserved
2:0	BIT_ADC_CLK_DIV	R/W	3h	This field defines clock driver level of ADC module. The source clock is OSC4M. A value of 0 is for clock divided by 4. A value of 1 is for clock divided by 8. A value of 2 is for clock divided by 16. A value of 3 is for clock divided by 24. A value of 4 is for clock divided by 32. A value of 5 is for clock divided by 64. A value of 6 is for clock divided by 128.

28.4.1.23 REG_ADC_OVERSAMPLE

- **Name:** ADC Oversample Register
- **Size:** 32 bits
- **Address offset:** 0ACh
- **Read/write access:** R/W

This register is used to configure ADC over sample.



Bit	Symbol	Access	Reset	Description
31:9	RSVD	-	0	Reserved
8:6	BIT_ADC_OSF	R/W	0	Oversampling right shift 000: no shift 001: shift 1-bit 010: shift 2-bit 011: shift 3-bit 100: shift 4-bit
5:3	BIT_ADC_OSR	R/W	0	Oversample ratio 000: 2x 001: 4x 010: 8x 011: 16x 100: 32x 101: 64x 110: 128x

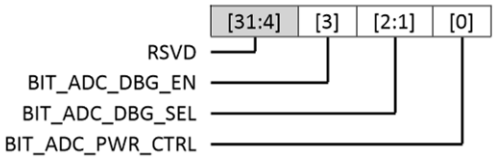
All information provided in this document is subject to legal disclaimers.
© REALTEK 2024. All rights reserved.

				111: 256x
2	RSVD	-	0	Reserved
1	BIT_OV_MODE	R/W	0	Oversample mode 1: All oversampling conversions done in regular sequence 0: All oversampling conversions done in staggered sequence
0	BIT_ADC_OS_EN	R/W	0	Oversample enable 1: enable 0: disable

28.4.1.24 REG_ADC_PWR_CTRL

- **Name:** ADC Power Control Register
- **Size:** 32 bits
- **Address offset:** 0B0h
- **Read/write access:** R/W

This register is used to control the ADC power and debug configuration.

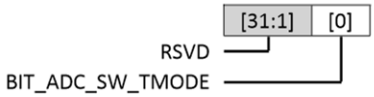


Bit	Symbol	Access	Reset	Description
31:4	RSVD	-	0	Reserved
3	BIT_ADC_DBG_EN	R/W	0	Debug port enable
2:1	BIT_ADC_DBG_SEL	R/W	0	Debug port select
0	BIT_ADC_PWR_CTRL	R/W	0	This field is used to power control.

28.4.1.25 REG_ADC_SW_TMODE

- **Name:** ADC Software Test Mode
- **Size:** 32 bits
- **Address offset:** 0B4h
- **Read/write access:** R/W

This register is used to set the software test mode.

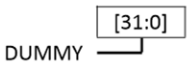


Bit	Symbol	Access	Reset	Description
31:1	RSVD	-	0	Reserved
0	BIT_ADC_SW_TMODE	R/W	0	This bit is used to change the high 4 bits in FIFO. 0: normal mode, the high 4 bits will be null or the channel number. 1: software test mode, the high 4 bits will be the counter number of sample points (loop from 0 to 15).

28.4.1.26 REG_ADC_DUMMY

- **Name:** ADC Dummy Register
- **Size:** 32 bits
- **Address offset:** 0B8h
- **Read/write access:** R/W

All information provided in this document is subject to legal disclaimers.
© REALTEK 2024. All rights reserved.



Bit	Symbol	Access	Reset	Description
31:0	DUMMY	R/W	ffffh	Dummy register

All information provided in this document is subject to legal disclaimers.
© REALTEK 2024. All rights reserved.

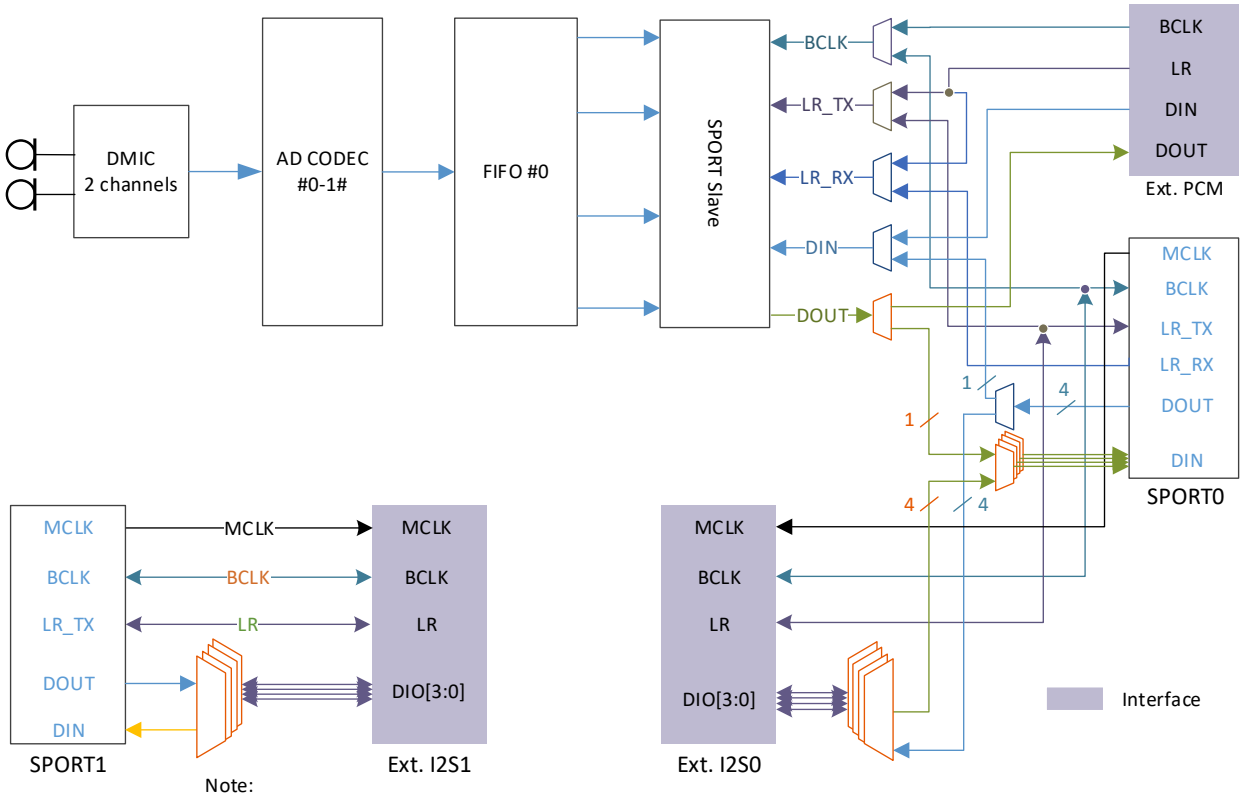
29 Audio

29.1 Overview

The Audio block consists of two parts:

- SPORT
- Digital microphone (DMIC) interface

The SPORT is an I2S interface. SPORT0 is for internal digital microphone interface and external I2S interface, and SPORT1 is only for external I2S interface. The whole block diagram is illustrated in *Figure 29-1*.



Note:

- SPORT DIN[3:0] and DOUT[3:0] are multiplexed to connect to DIO[3:0] correspondingly.
- When connected to external I2S devices, SPORT0/1 can act as master, also as slave.

Figure 29-1 Audio block diagram

29.2 Clock Domain

The clock domains of audio are also divided into two modules:

- SPORT
- Digital microphone interface

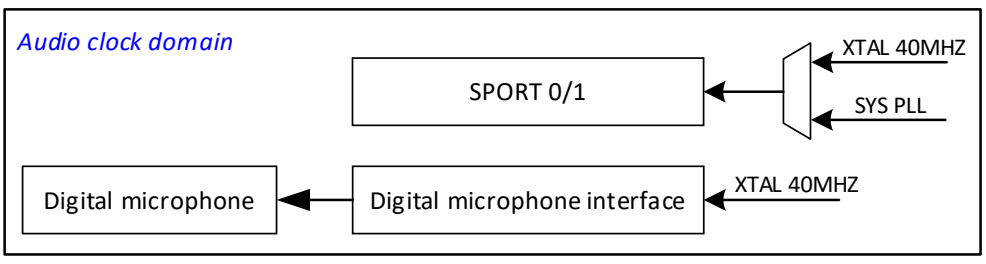


Figure 29-2 Audio clock domain

All information provided in this document is subject to legal disclaimers.
© REALTEK 2024. All rights reserved.

29.3 SPORT

In RTL8721Dx, the SPORT is designed for internal digital microphone interface and external I2S interface.

29.3.1 SPORT Data Path

The data paths of SPORT0 and SPORT1 are shown in *Figure 29-3* and *Figure 29-4*.

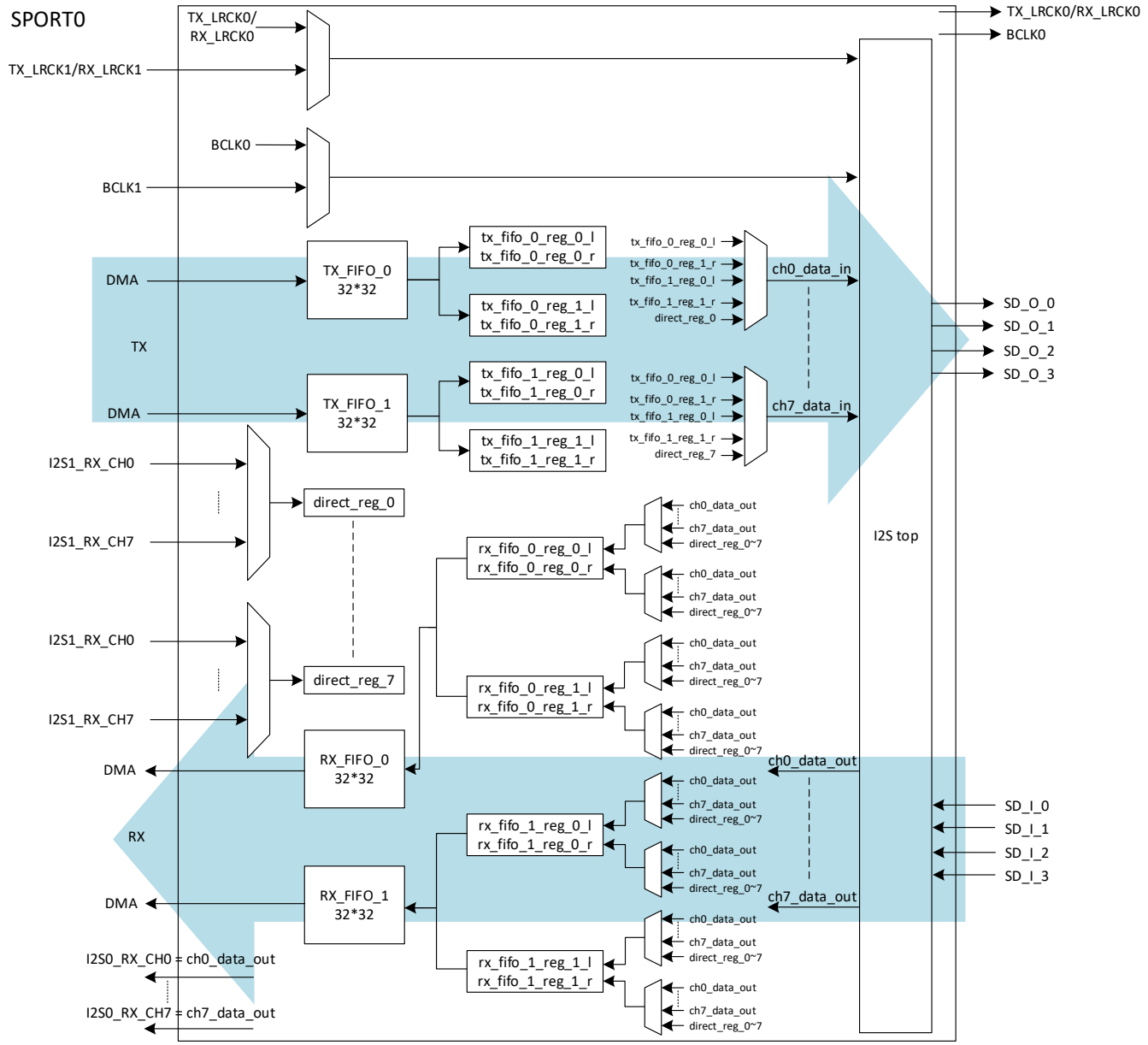


Figure 29-3 SPORT0 data path

All information provided in this document is subject to legal disclaimers.
© REALTEK 2024. All rights reserved.

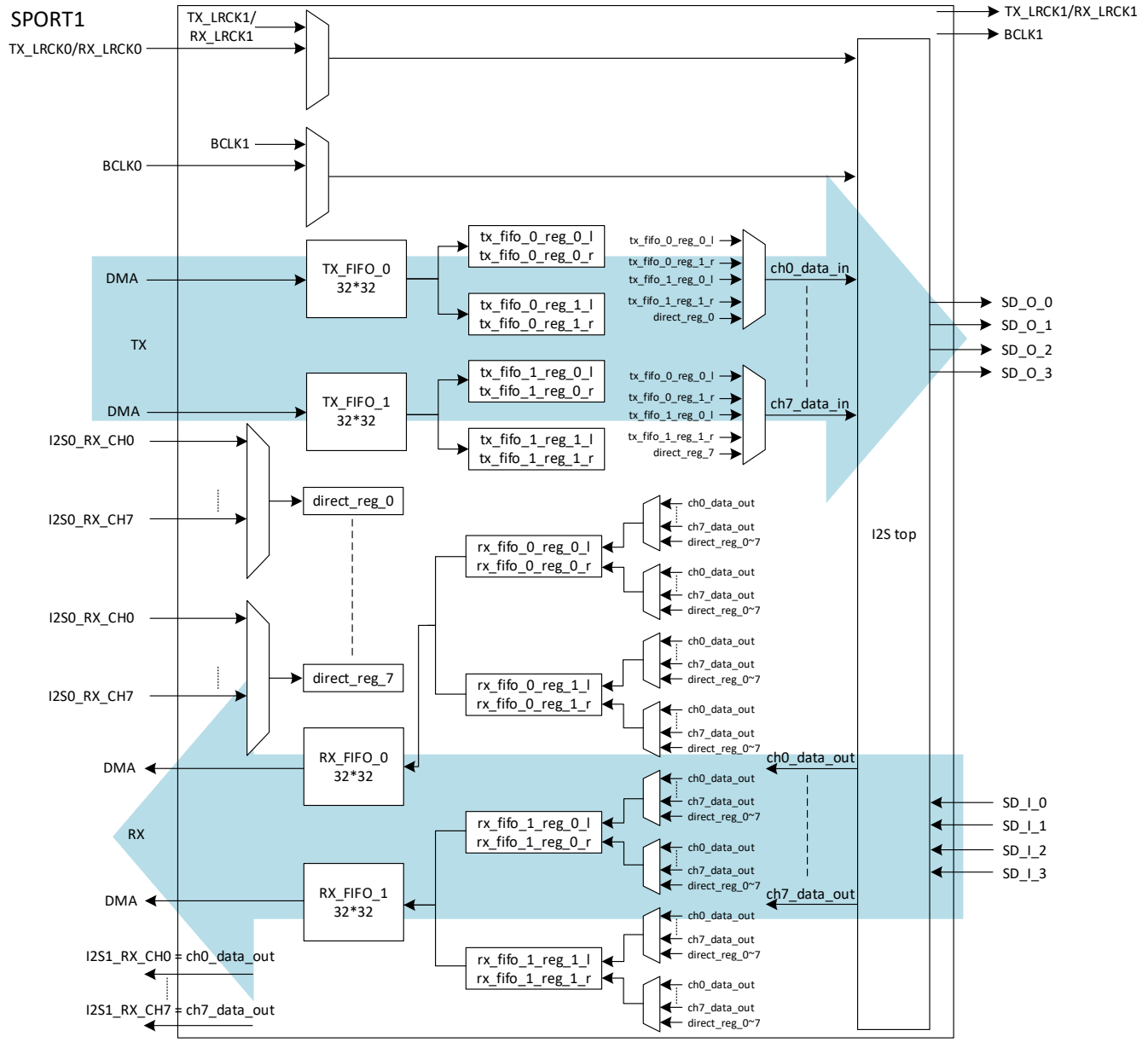


Figure 29-4 SPORT1 data path

29.3.2 SPORT Function

The SPORT0 is designed to transfer audio-data streaming between audio codec and SoC memory, and also can work with external audio codec in master or slave mode. SPORT1 only can work with external audio codec in master or slave mode.

The SPORT 0/1 include the following features:

- Supports up to 8-channel I2S transmitter by TDM or PCM mode
- Up to 4 serial data outputs/inputs are transmitted within a sample period (multi io)
- Audio data word length: 16/20/24/32 bits
- Channel length: 16/20/24/32 bits
- Works in master and slave mode
- In 2 channels I2S mode, fs supports up to 384kHz
- In multi-IO mode, fs supports up to 384kHz

SPORT fs counter and phase counter:

- fs counter is used to count the number of LRCLK, and on every falling edge of LRCLK.
- Phase counter will accumulate once in two BCLK cycles. The maximum accumulated value of phase counter is 31.

All information provided in this document is subject to legal disclaimers.
© REALTEK 2024. All rights reserved.

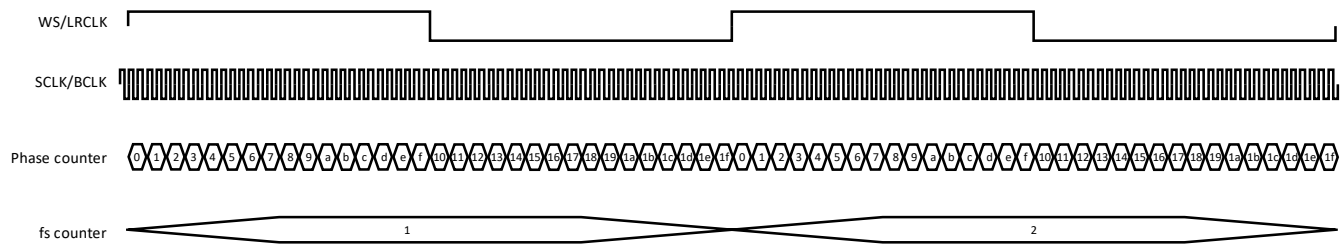


Figure 29-5 SPORT fs counter and phase counter

NOTE

When using phase counter, rx/tx_bclk_div_ratio should be configured to 63.

The SPORT direct mode features:

- Used for data transmission between different SPORTs without CPU and DMA involved in data transfer.
- When two SPORTs work in direct mode, the clock needs to be at the same frequency.

SPORT FIFO:

- TX_FIFO_0 and TX_FIFO_1 are two asynchronous ping-pong FIFO.
  Each FIFO is depth=32 and width=32, so 2*32*4 bytes=64 words.
- RX_FIFO_0 and RX_FIFO_1 are two asynchronous ping-pong FIFO.
  Each FIFO is depth=32 and width=32, so 2*32*4 bytes=64 words.

6/8 channels for data transmission, FIFO0 and FIFO1 would be used. Two FIFOs will request at the same time, and two SPORTs will produce 8 requests at the same time.

I2S PINMUX:

- The data PIN DIO[3:0] of I2S can be used as output or input. It can be set as input or output in the corresponding part of I2S in the PAD register.

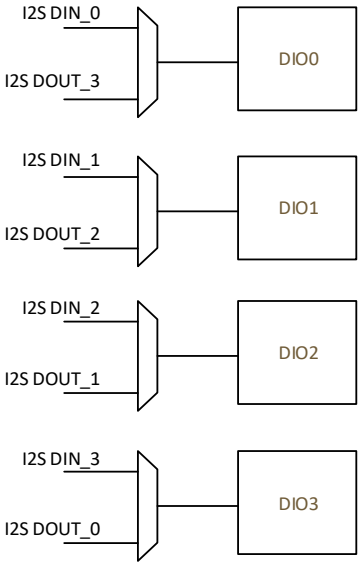


Figure 29-6 I2S data pinmux

29.3.2.1 I2S Signal Introduction

The I2S bus has three lines:

- Continuous serial clock (SCLK/BCK)
  - One SCK pulse generates a data bit
  - Master generates SCK
- Word select (WS/LRCLK)
  The word select line indicates the channel being transmitted:
  - WS = 0; channel 1 (left)
  - WS = 1; channel 2 (right)

  Changes one clock period before the MSB is transmitted

All information provided in this document is subject to legal disclaimers.
© REALTEK 2024. All rights reserved.

- Serial data (SD)
  SD is transmitted in two complements with the MSB first. The MSB has a fixed position, whereas the position of the LSB depends on the word length.
  When the system word length is greater than the transmitter word length, the word is truncated (the least significant data bits are set to '0').
  - If the receiver is sent more bits than its word length, the bits after the LSB are ignored.
  - If the receiver is sent fewer bits than its word length, the missing bits are set to 0 internally.

29.3.2.2 I2S Data Format

The I2S interface supports I2S (Philips) format, Left-justified (MSB) format, Right-justified (LSB) format, PCM, and TDM mode. Software can select any mode by setting the I2S control register. *Figure 29-7* to *Figure 29-22* show the I2S data format.

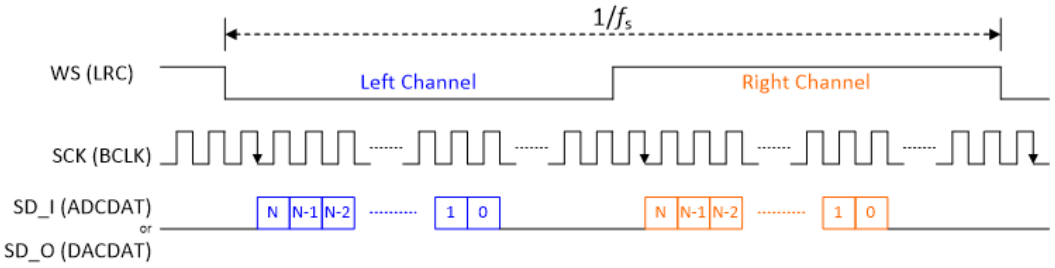


Figure 29-7 I2S format

NOTE

- *Typically, f_s = 8/16/32/44.1/48/88.2/96/192/384kHz*
- *Channel length: 16/20/24/32 bits (N+1)*
- *SCK = Arbitrarily cycles within 1/fs, but >= 2*(N+1) * fs, <= 256 * fs*

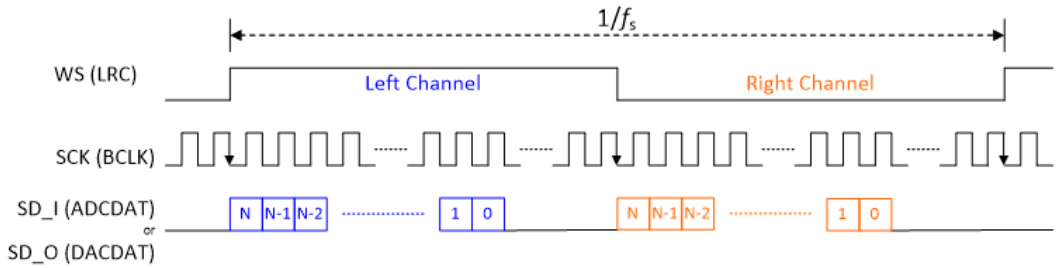


Figure 29-8 Left-justified format

NOTE

- *Typically, f_s = 8/16/32/44.1/48/88.2/96/192/384kHz*
- *Channel length: 16/20/24/32 bits (N+1)*
- *SCK >= 2*(N+1)*fs, <= 256 *fs*

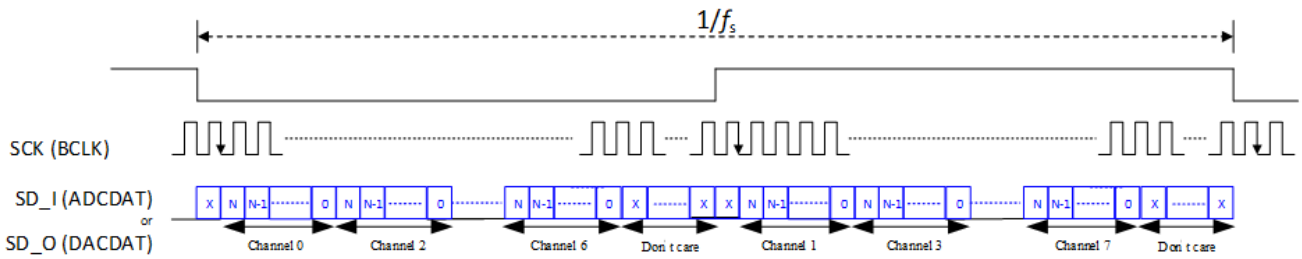


Figure 29-9 I2S TDM 8 mode

NOTE

- *Typically, f_s = 8/16/32/44.1/48kHz*
- *Channel length: 16/20/24/32 bits (N+1)*
- *SCK >= 8*(N+1)*f_s, <= 256 * f_s*

All information provided in this document is subject to legal disclaimers.
© REALTEK 2024. All rights reserved.

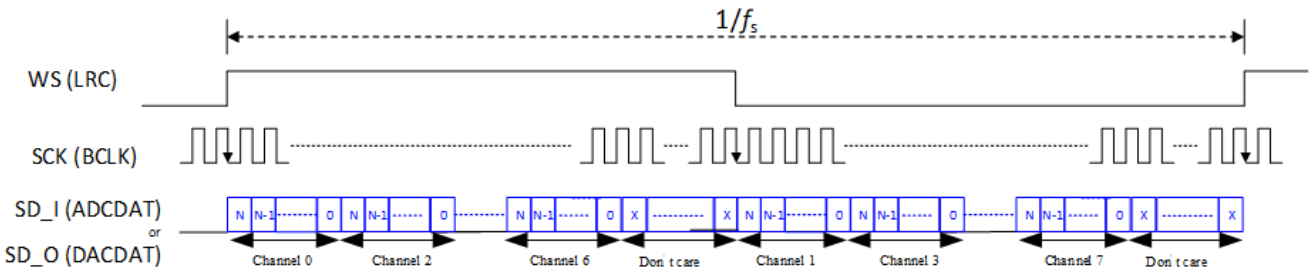


Figure 29-10 Left-justified TDM 8 mode

NOTE

- *Typically, f_s = 8/16/32/44.1/48kHz*
- *Channel length: 16/20/24/32 bits (N+1)*
- *SCK >= 8*(N+1)*f_s, <= 256 * f_s*

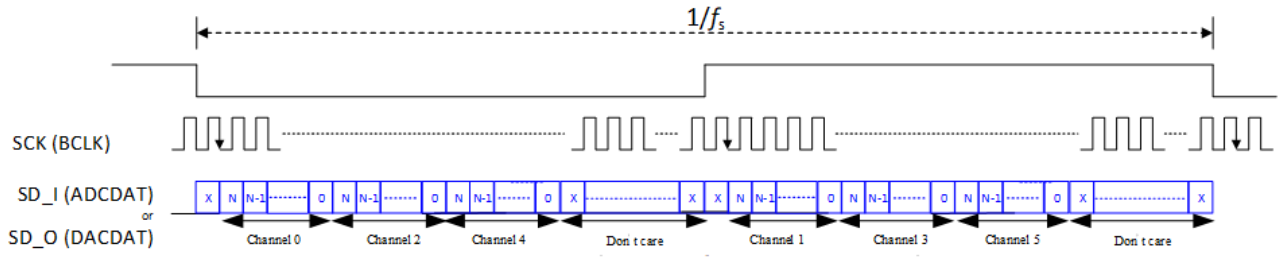


Figure 29-11 I2S TDM 6 mode

NOTE

- *Typically, f_s = 8/16/32/44.1/48kHz*
- *Channel length: 16/20/24/32 bits (N+1)*
- *SCK >= 6*(N+1)*f_s, <= 256 * f_s*

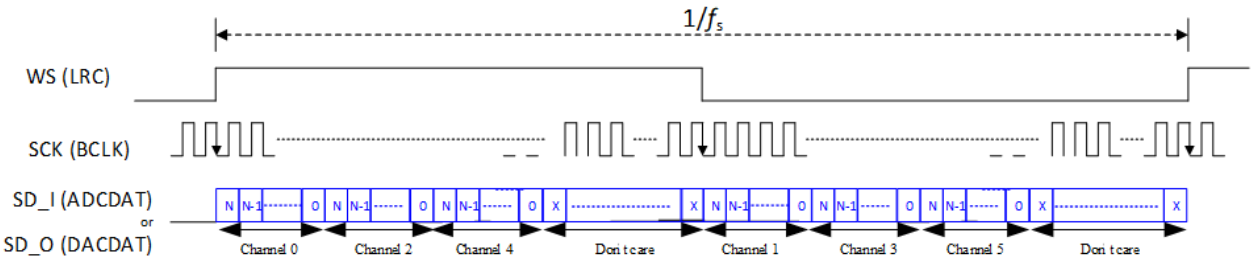


Figure 29-12 Left-justified TDM 6 mode

NOTE

- *Typically, f_s = 8/16/32/44.1/48kHz*
- *Channel length: 16/20/24/32 bits (N+1)*
- *SCK >= 6*(N+1)*f_s, <= 256 * f_s*

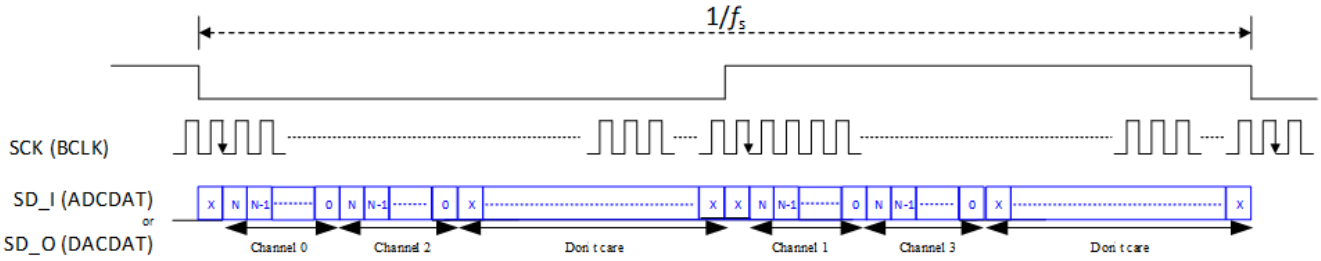


Figure 29-13 I2S TDM 4 mode

All information provided in this document is subject to legal disclaimers.
© REALTEK 2024. All rights reserved.

NOTE

- *Typically, f_s = 8/16/32/44.1/48/88.2/96kHz*
- *Channel length: 16/20/24/32 bits (N+1)*
- *SCK >= 4*(N+1)*f_s, <= 256 * f_s*

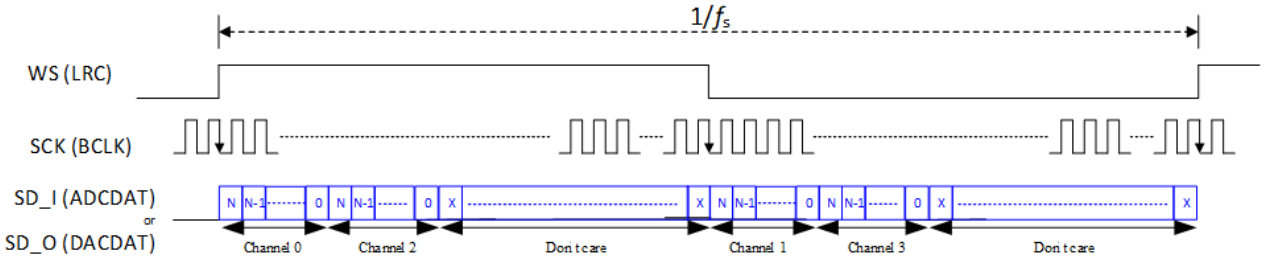


Figure 29-14 Left-justified TDM 4 mode

NOTE

- *Typically, f_s = 8/16/32/44.1/48/88.2/96kHz*
- *Channel length: 16/20/24/32 bits (N+1)*
- *SCK >= 4*(N+1)*f_s, <= 256 * f_s*

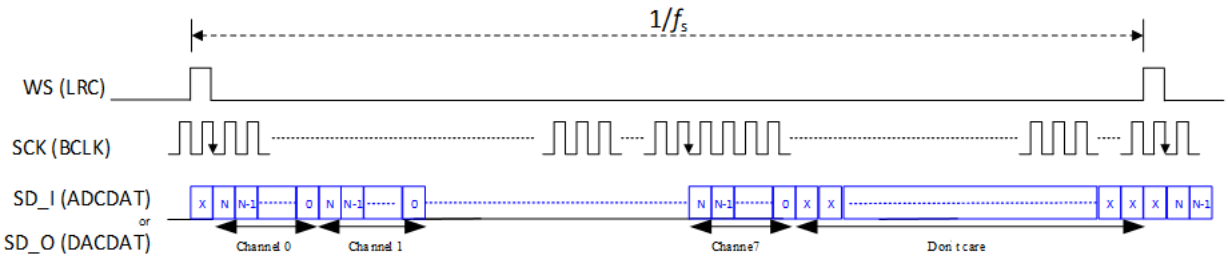


Figure 29-15 PCM mode A in TDM 8 mode

NOTE

- *Typically, f_s = 8/16/32/44.1/48kHz*
- *Channel length: 16/20/24/32 bits (N+1)*
- *SCK >= 8*(N+1)*f_s, <= 256 * f_s*

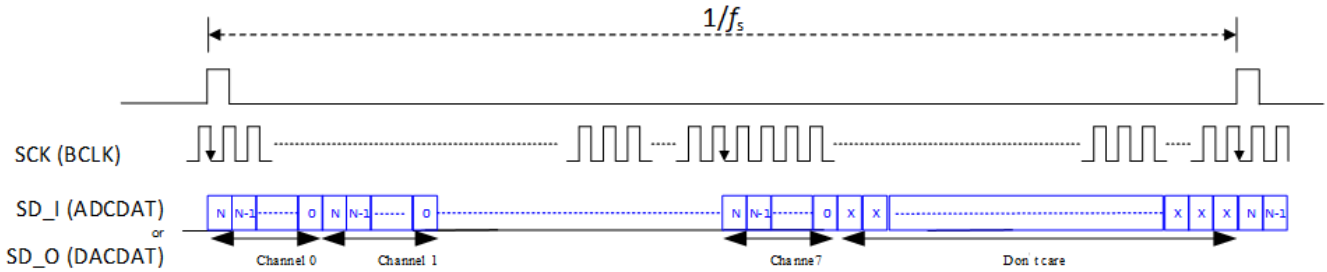


Figure 29-16 PCM mode B in TDM 8 mode

NOTE

- *Typically, f_s = 8/16/32/44.1/48kHz*
- *Channel length: 16/20/24/32 bits (N+1)*
- *SCK >= 8*(N+1)*f_s, <= 256 *f_s*

All information provided in this document is subject to legal disclaimers.
© REALTEK 2024. All rights reserved.

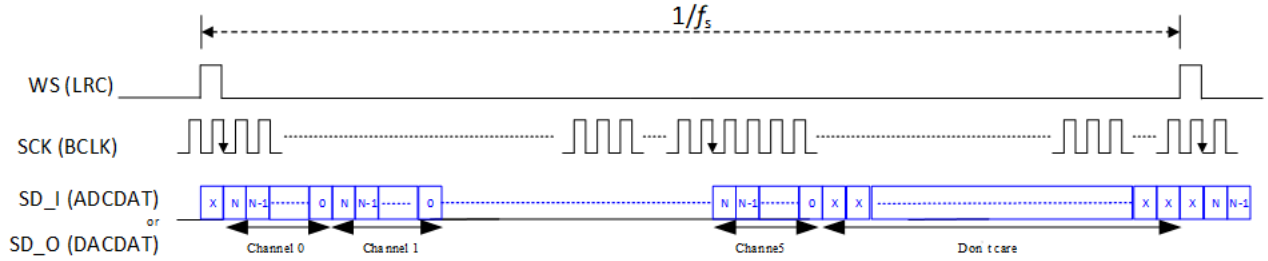


Figure 29-17 PCM mode A in TDM 6 mode

NOTE

- *Typically, f_s = 8/16/32/44.1/48kHz*
- *Channel length: 16/20/24/32 bits (N+1)*
- *SCK >= 6*(N+1)*f_s, <= 256 * f_s*

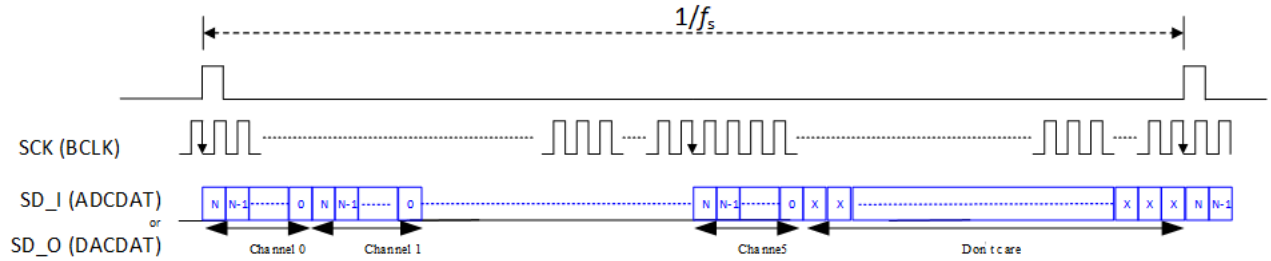


Figure 29-18 PCM mode B in TDM 6 mode

NOTE

- *Typically, f_s = 8/16/32/44.1/48kHz*
- *Channel length: 16/20/24/32 bits (N+1)*
- *SCK >= 6*(N+1)*f_s, <= 256 * f_s*

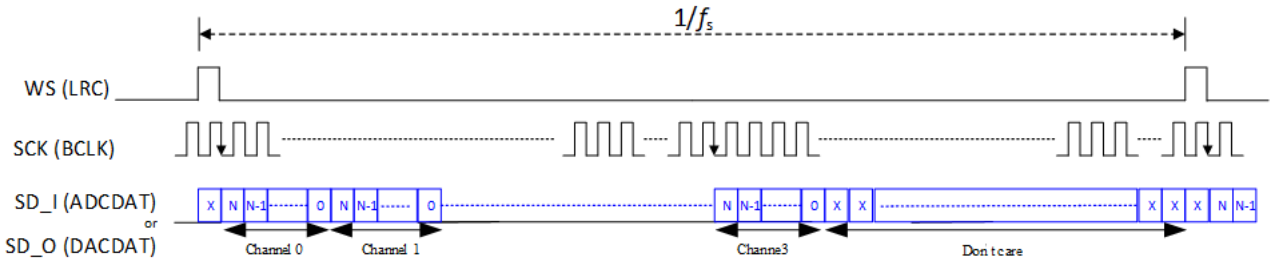


Figure 29-19 PCM mode A in TDM 4 mode

NOTE

- *Typically, f_s = 8/16/32/44.1/48/88.2/96kHz*
- *Channel length: 16/20/24/32 bits (N+1)*
- *SCK >= 4*(N+1)*f_s, <= 256 * f_s*

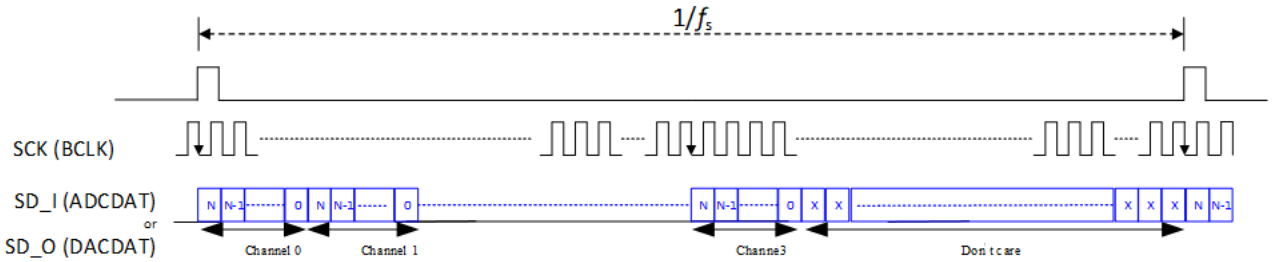


Figure 29-20 PCM mode B in TDM 4 mode

All information provided in this document is subject to legal disclaimers.
© REALTEK 2024. All rights reserved.

NOTE

- *Typically, f_s = 8/16/32/44.1/48/88.2/96kHz*
- *Channel length: 16/20/24/32 bits (N+1)*
- *SCK >= 4*(N+1)*f_s, <= 256 * f_s*

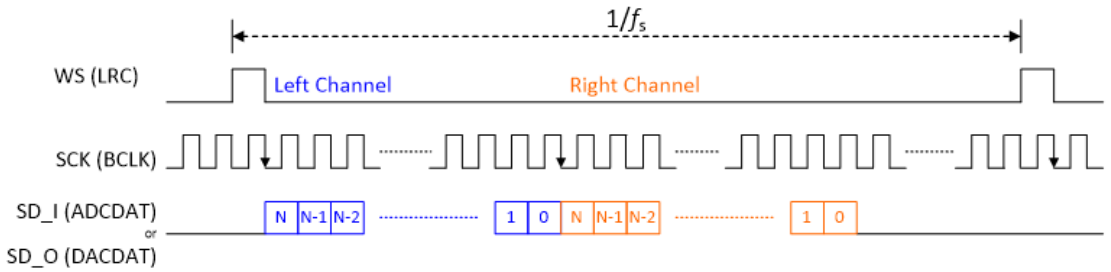


Figure 29-21 PCM mode A

NOTE

- *Typically, f_s = 8/16/32/44.148/88.2/96/192kHz*
- *Channel length: 16/20/24/32 bits (N+1)*
- *SCK >= 2*(N+1)*f_s, <= 256 * f_s*

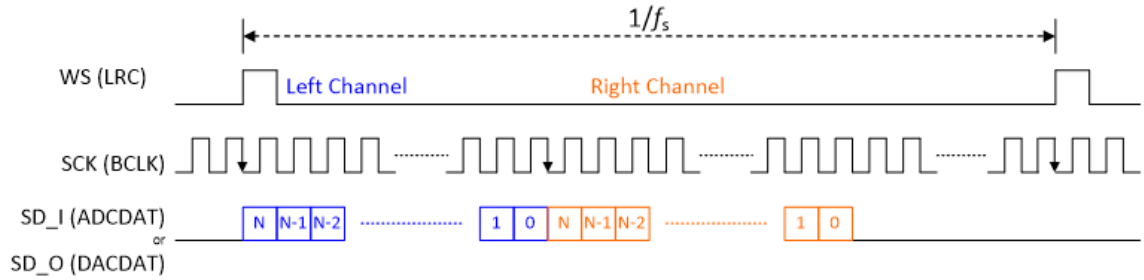


Figure 29-22 PCM mode B

NOTE

- *Typically, f_s = 8/16/32/44.148/88.2/96/192kHz*
- *Channel length: 16/20/24/32 bits (N+1)*
- *SCK >= 2*(N+1)*f_s, <= 256 * f_s*

I2S supports 16/20/24/32 bits channel length, the relationship between audio data length and channel length is illustrated in *Figure 29-23*.

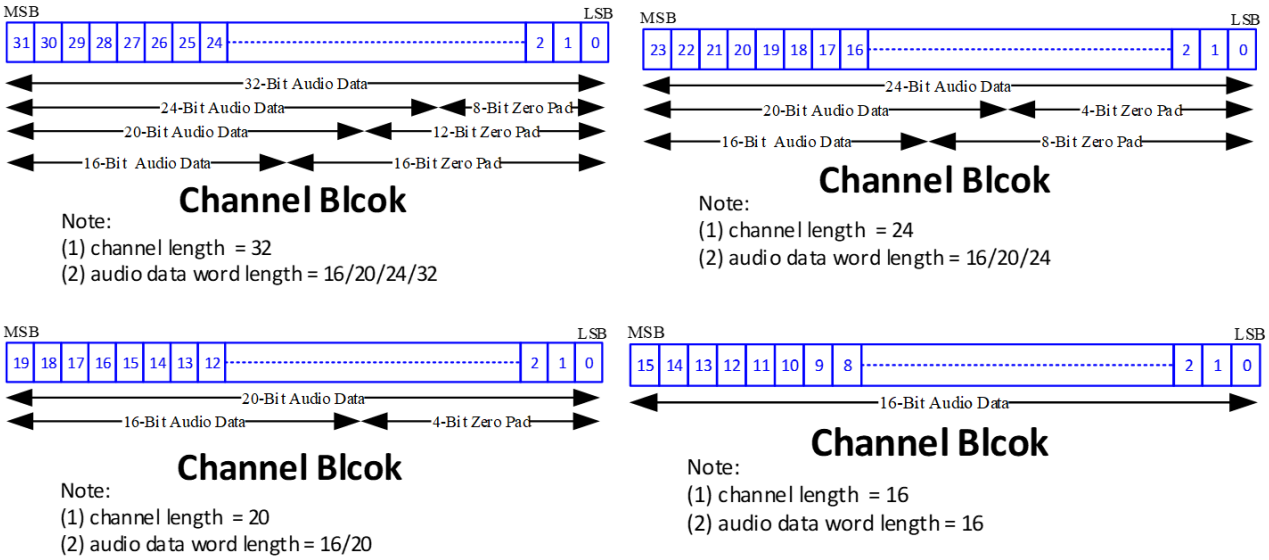


Figure 29-23 I2S data length and channel length

All information provided in this document is subject to legal disclaimers.
© REALTEK 2024. All rights reserved.

29.3.2.3 SPORT Parameters

Table 29-1 SPORT 0/1 parameters

Interface/Format	Sampling rate	Audio bits	Channel	Channel length	BCLK polarity	Serial data	Multi-IO	Mode
I2S	384kHz	16 bits	Stereo	16 bits	BCLK	MSB first	4 SD_IO	Master
Left Justified	192kHz	20 bits	Mono	20 bits	BCLK inverse	LSB first		Slave
PCM Mode A (Short Frame Sync)	96kHz	24 bits		24 bits				
	88.2kHz	32 bits		32 bits				
PCM Mode B (Short Frame Sync)	48kHz							
	44.1kHz							
	32kHz							
	16kHz							
	8kHz							

The key features include:

- Up to 4 serial data outputs are transmitted within a sample period
- Audio data word length: 16/20/24/32 bits
- Channel length: 16/20/24/32 bits
- In 2 channels mode, fs supports up to 384kHz

29.3.2.4 SPORT Clock

Two SPORTs can work in master and slave mode, and the clock of SPORT is shown in *Figure 29-24*.

- Clock source: XTAL, and CPU PLL.

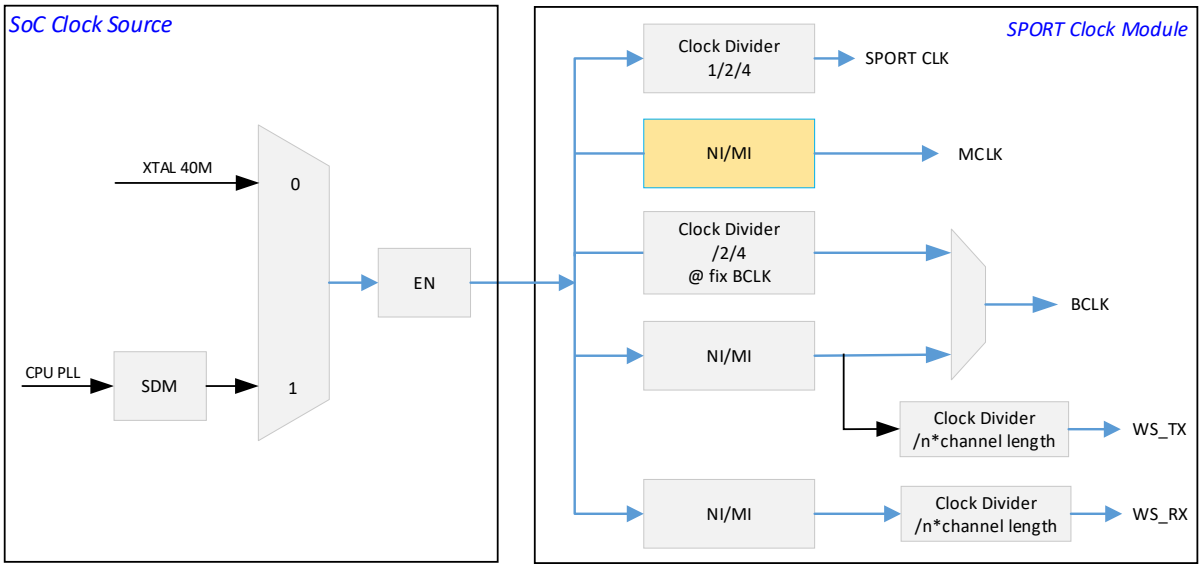


Figure 29-24 SPORT interface clock

29.3.3 SPORT Register

The base address of SPORT registers is 0x4012A000.

NOTE

Remember to add the base address when accessing each SPORT register.

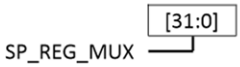
Name	Address offset	Access	Description
REG_SP_REG_MUX	000h	R/W	SPORT MUX Register
REG_SP_CTRL0	004h	R/W	SPORT Control Register 0
REG_SP_CTRL1	008h	R/W	SPORT Control Register 1
REG_SP_INT_CTRL	00Ch	R/W	SPORT Interrupt Control Register
REG_SP_TRX_COUNTER_STATUS	014h	R	SPORT TRX Counter Status Register
REG_SP_ERR	018h	R	SPORT Error Register

All information provided in this document is subject to legal disclaimers.
© REALTEK 2024. All rights reserved.

REG_SR_TX_BCLK	01Ch	R/W	SPORT Tx BCLK Register
REG_SP_TX_LRCLK	020h	R/W	SPORT Tx LRCLK Register
REG_SP_FIFO_CTRL	024h	R	SPORT FIFO Control Register
REG_SP_FORMAT	028h	R/W	SPORT Format Register
REG_SP_RX_BCLK	02Ch	R/W	SPORT Rx BCLK Register
REG_SP_RX_LRCLK	030h	R/W	SPORT Rx LRCLK Register
REG_SP_DSP_COUNTER	034h	R	SPORT DSP Counter Register
REG_SP_DIRECT_CTRL0	03Ch	R/W	SPORT Direct Control Register 0
REG_SP_FIFO_IRQ	044h	R/W	SPORT FIFO IRQ Register
REG_SP_DIRECT_CTRL1	048h	R/W	SPORT Direct Control Register 1
REG_SP_DIRECT_CTRL2	04Ch	R/W	SPORT Direct Control Register 2
REG_SP_DIRECT_CTRL3	054h	R/W	SPORT Direct Control Register 3
REG_SP_DIRECT_CTRL4	058h	R/W	SPORT Direct Control Register 4
REG_SP_RX_COUNTER1	05Ch	R/W	SPORT Rx Counter Register 1
REG_SP_RX_COUNTER2	060h	R	SPORT Rx Counter Register 2
REG_SP_MCLK	064h	R/W	SPORT MCLK Register
REG_TX_FIFO_0_WR_ADDR	800h	R/W	TX FIFO 0 Write Address Register
REG_RX_FIFO_0_RD_ADDR	880h	R	RX FIFO 0 Read Address Register
REG_TX_FIFO_1_WR_ADDR	900h	R/W	TX FIFO 1 Write Address Register
REG_RX_FIFO_1_RD_ADDR	980h	R	RX FIFO 1 Read Address Register

29.3.3.1 REG_SP_REG_MUX

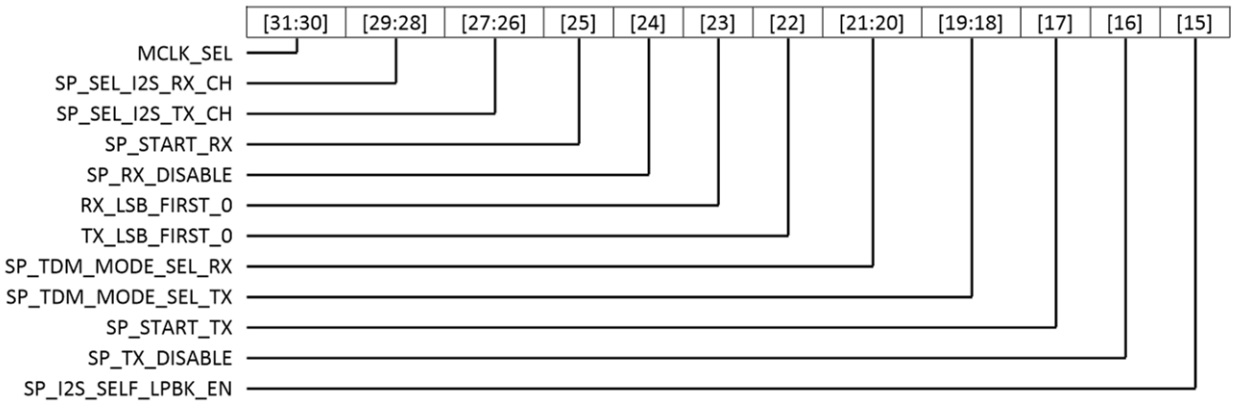
- **Name:** SPORT MUX Register
- **Size:** 32 bits
- **Address offset:** 000h
- **Read/write access:** R/W



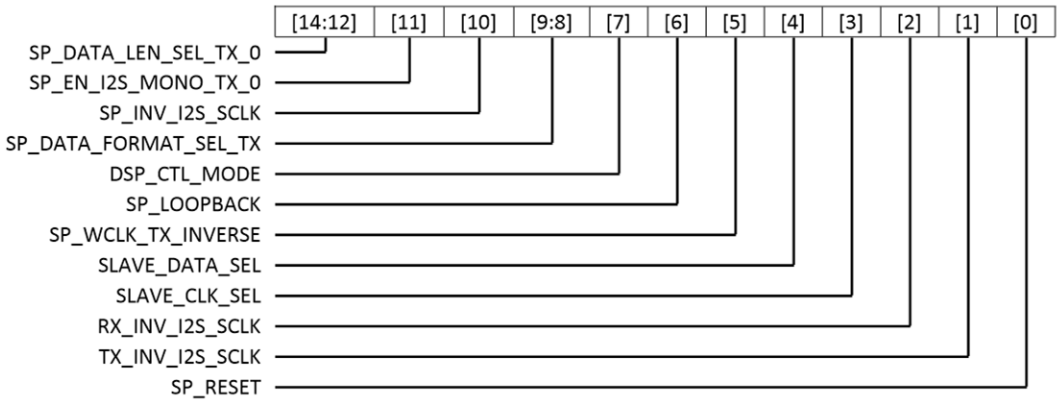
Bit	Symbol	Access	Reset	Description
31:0	SP_REG_MUX	R/W	0xFFFFFFFF	Mux of register write with different base address of the same SPORT. This field can be set as different value with four different base address in one SPORT, but other registers share the same value with four different base address in one SPORT.

29.3.3.2 REG_SP_CTRL0

- **Name:** SPORT Control Register 0
- **Size:** 32 bits
- **Address offset:** 004h
- **Read/write access:** R/W



All information provided in this document is subject to legal disclaimers.
© REALTEK 2024. All rights reserved.



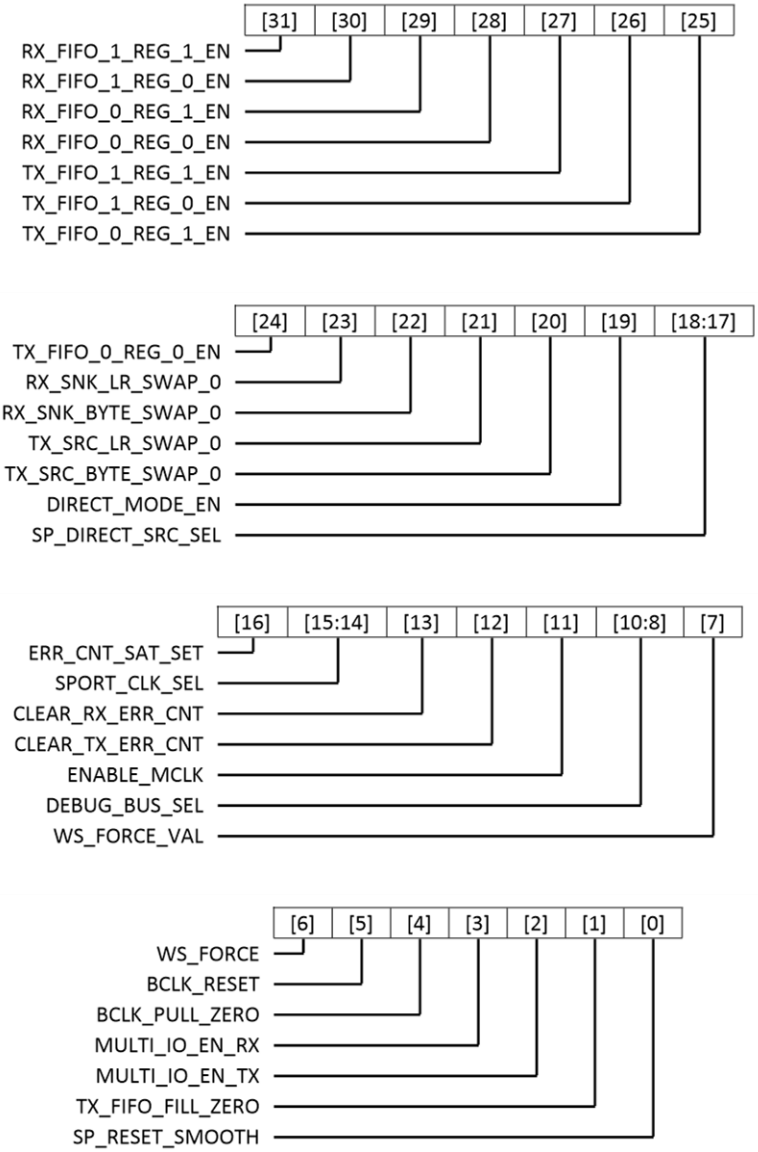
Bit	Symbol	Access	Reset	Description
31:30	MCLK_SEL	R/W	0x0	Not used, refer to 0x64 2'b00: MCLK output=dsp_src_clk/4 2'b01: MCLK output=dsp_src_clk/2 2'b10/2'b11: MCLK output=dsp_src_clk
29:28	SP_SEL_I2S_RX_CH	R/W	0x0	2'b00: L/R 2'b01: R/L 2'b10: L/L 2'b11: R/R @ ADC path
27:26	SP_SEL_I2S_TX_CH	R/W	0x0	2'b00: L/R 2'b01: R/L 2'b10: L/L 2'b11: R/R @ DAC path
25	SP_START_RX	R/W	0	1'b0: Rx is disabled 1'b1: Rx is started
24	SP_RX_DISABLE	R/W	1	1'b1: SPORT Rx is disabled. 1'b0: SPORT Rx is enabled.
23	RX_LSB_FIRST_0	R/W	0	1'b0: MSB first when Rx 1'b1: LSB first
22	TX_LSB_FIRST_0	R/W	0	1'b0: MSB first when Tx 1'b1: LSB first
21:20	SP_TDM_MODE_SEL_RX	R/W	0x0	2'b00: Without TDM 2'b01: TDM4 2'b10: TDM6 2'b11: TDM8
19:18	SP_TDM_MODE_SEL_TX	R/W	0x0	2'b00: Without TDM 2'b01: TDM4 2'b10: TDM6 2'b11: TDM8
17	SP_START_TX	R/W	0	1'b0: Tx is disabled. 1'b1: Tx is started.
16	SP_TX_DISABLE	R/W	1	1'b1: SPORT Tx is disabled. 1'b0: SPORT Tx is enabled.
15	SP_I2S_SELF_LPBK_EN	R/W	0	1'b1: internal loopback mode is enabled
14:12	SP_DATA_LEN_SEL_TX_0	R/W	0x0	3'b000: 16 bits 3'b001: 20 bits 3'b010: 24 bits 3'b100: 32 bits
11	SP_EN_I2S_MONO_TX_0	R/W	0	1'b1: mono 1'b0: stereo
10	SP_INV_I2S_SCLK	R/W	0	1'b1: I2S/PCM bit clock is inverted
9:8	SP_DATA_FORMAT_SEL_TX	R/W	0x0	2'b00: I2S 2'b01: Left-justified 2'b10: PCM mode A 2'b11: PCM mode B
7	DSP_CTL_MODE	R/W	0	1'b1: DSP and SPORT1 handshaking is enabled.

All information provided in this document is subject to legal disclaimers.
© REALTEK 2024. All rights reserved.

				1’b0: GDMA and SPORT1 handshaking is enabled.
6	SP_LOOPBACK	R/W	0	1’b1: self-loopback mode
5	SP_WCLK_TX_INVERSE	R/W	0	1’b1: I2S/PCM word clock is inverted for Tx (SPK path)
4	SLAVE_DATA_SEL	R/W	0	1’b1: To be an I2S or PCM slave (data path)
3	SLAVE_CLK_SEL	R/W	0	1’b1: To be an I2S or PCM slave (CLK path)
2	RX_INV_I2S_SCLK	R/W	0	1'b1: sclk to Rx path (ADC path) is inverted
1	TX_INV_I2S_SCLK	R/W	0	1'b1: sclk to Tx path (DAC path) is inverted
0	SP_RESET	R/W	0	1’b1: reset SPORT1 module, and remember to write “1” to reset and then write “0” to release from reset.

29.3.3.3 REG_SP_CTRL1

- **Name:** SPORT Control Register 1
- **Size:** 32 bits
- **Address offset:** 008h
- **Read/write access:** R/W



Bit	Symbol	Access	Reset	Description
31	RX_FIFO_1_REG_1_EN	R/W	0	1’b1: Enable last two channels of RX_FIFO_1. Only enable when RX_FIFO_1_REG_1_EN = 1.
30	RX_FIFO_1_REG_0_EN	R/W	0	1’b1: Enable first two channels of RX_FIFO_1. Only enable when

All information provided in this document is subject to legal disclaimers.
© REALTEK 2024. All rights reserved.

				RX_FIFO_0_REG_0_EN = 1.
29	RX_FIFO_0_REG_1_EN	R/W	0	1’b1: Enable last two channels of RX_FIFO_0. Only enable when RX_FIFO_0_REG_0_EN = 1.
28	RX_FIFO_0_REG_0_EN	R/W	1	1’b1: Enable first two channels of RX_FIFO_0. Disable 0x0008[28] ~ Disable 0x0008[31] at the same time to reset Rx FIFO.
27	TX_FIFO_1_REG_1_EN	R/W	0	1’b1: Enable last two channels of TX_FIFO_1. Only enable when TX_FIFO_1_REG_0_EN = 1.
26	TX_FIFO_1_REG_0_EN	R/W	0	1’b1: Enable first two channels of TX_FIFO_1. Only enable when TX_FIFO_0_REG_0_EN = 1.
25	TX_FIFO_0_REG_1_EN	R/W	0	1’b1: Enable last two channels of TX_FIFO_0. Only enable when TX_FIFO_0_REG_0_EN = 1.
24	TX_FIFO_0_REG_0_EN	R/W	1	1’b1: Enable first two channels of TX_FIFO_0. Disable 0x0008[24] ~ Disable 0x0008[27] at the same time to reset Tx FIFO.
23	RX_SNK_LR_SWAP_0	R/W	0	1’b1: swap L/R audio samples written to the sink memory of RX_FIFO_0
22	RX_SNK_BYTE_SWAP_0	R/W	0	1’b1: swap H/L bytes written to the sink memory of RX_FIFO_0
21	TX_SRC_LR_SWAP_0	R/W	0	1’b1: swap L/R audio samples read from the source memory of TX_FIFO_0
20	TX_SRC_BYTE_SWAP_0	R/W	0	1’b1: swap H/L bytes read from the source memory of TX_FIFO_0
19	DIRECT_MODE_EN	R/W	0	1’b1: WS (LRCK) and SCK (BCLK) are from other SPORT
18:17	SP_DIRECT_SRC_SEL	R/W	0x0	2’b00: WS and SCK are from SPORT0. 2’b01: WS and SCK are from SPORT1. 2’b10: WS and SCK are from SPORT2. 2’b11: WS and SCK are from SPORT3.
16	ERR_CNT_SAT_SET	R/W	0	1’b1: saturation count (65534 -> 65535 -> 65535 ...) 1’b0: wrap count (65534 -> 65535 -> 0 -> 1 -> 2 ...)
15:14	SPORT_CLK_SEL	R/W	0x0	2’b0x00: dsp_src_clk (BCLK*2) 2’b10: dsp_src_clk (BCLK*4)/2 2’b11: dsp_src_clk (BCLK*8)/4
13	CLEAR_RX_ERR_CNT	R/W	0	Write 1'b1 and then write 0 to clear Rx error counter
12	CLEAR_TX_ERR_CNT	R/W	0	Write 1'b1 and then write 0 to clear Tx error counter
11	ENABLE_MCLK	R/W	0	Enable mclk.
10:8	DEBUG_BUS_SEL	R/W	0x0	3’b000: debug_bus_a 3’b001: debug_bus_b … 3’b111: debug_bus_h
7	WS_FORCE_VAL	R/W	1	When WS_FORCE = 1, ws_out_tx and ws_out_rx = "ws_force_val"
6	WS_FORCE	R/W	0	1'b1: Make ws_out_tx and ws_out_rx = "ws_force_val"
5	BCLK_RESET	R/W	0	1’b0: Enable bclk 1’b1: Disable and reset bclk
4	BCLK_PULL_ZERO	R/W	0	Write 1’b1 to pull bclk to 0 smoothly Write 1’b0 to reopen bclk
3	MULTI_IO_EN_RX	R/W	1	1’b1: Enable multi-IO of Rx 1’b0: Disable multi-IO of Rx
2	MULTI_IO_EN_TX	R/W	1	1’b1: Enable multi-IO of Tx 1’b0: Disable multi-IO of Tx
1	TX_FIFO_FILL_ZERO	WA0	0	X is the burst size of TX_FIFO_0. Y is the burst size of TX_FIFO_1. Fill TX_FIFO_0 with X zero data and fill TX_FIFO_1 with Y zero data. This control bit is "write 1 clear" type
0	SP_RESET_SMOOTH	R/W	0	1’b1: reset SPORT1 module with complete LRCK cycle.

29.3.3.4 REG_SP_INT_CTRL

- **Name:** SPORT Interrupt Control Register
- **Size:** 32 bits
- **Address offset:** 00Ch
- **Read/write access:** R/W

All information provided in this document is subject to legal disclaimers.
© REALTEK 2024. All rights reserved.

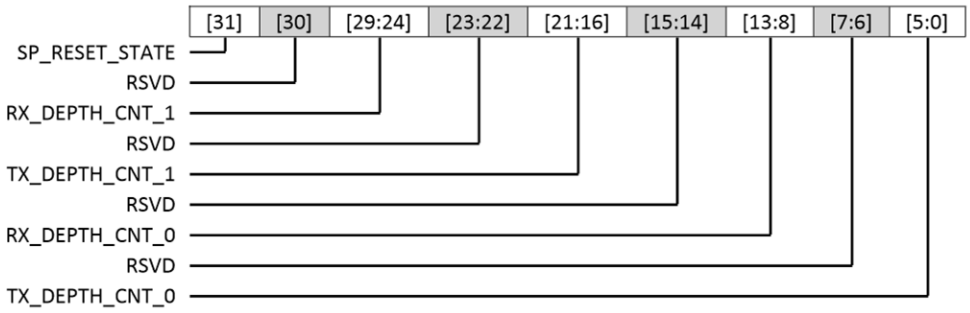


Bit	Symbol	Access	Reset	Description
31:24	INT_ENABLE_DSP_1	R/W	0x0	Bit[24]: for the interrupt of “sp_ready_to_tx_1” Bit[25]: for the interrupt of “sp_ready_to_rx_1” Bit[26]: for the interrupt of “tx_fifo_full_intr_1” Bit[27]: for the interrupt of “rx_fifo_full_intr_1” Bit[28]: for the interrupt of “tx_fifo_empty_intr_1” Bit[29]: for the interrupt of “rx_fifo_empty_intr_1” Bit[30]: for the interrupt of “tx_i2s_idle_1” Bit[31]: Reserved
23:16	INT_ENABLE_DSP_0	R/W	0x0	Bit[16]: for the interrupt of “sp_ready_to_tx” Bit[17]: for the interrupt of “sp_ready_to_rx” Bit[18]: for the interrupt of “tx_fifo_full_intr” Bit[19]: for the interrupt of “rx_fifo_full_intr” Bit[20]: for the interrupt of “tx_fifo_empty_intr” Bit[21]: for the interrupt of “rx_fifo_empty_intr” Bit[22]: for the interrupt of “tx_i2s_idle” Bit[23]: Reserved
15:14	DUMMY	R/W	0x0	-
13:9	INTR_CLR_1	R/W	0x0	Bit[9]: for the interrupt of “tx_fifo_full_intr_1” Bit[10]: for the interrupt of “rx_fifo_full_intr_1” Bit[11]: for the interrupt of “tx_fifo_empty_intr_1” Bit[12]: for the interrupt of “rx_fifo_empty_intr_1” Bit[13]: Reserved
8	RX_DSP_CLEAR_INT_1	R/W	0	For DSP mode (bypass GDMA), F/W writes 1’b1 and then 1’b0 to clear Rx interrupt. NOTE *Rx interrupt is to indicate that DSP can get audio data from RX_FIFO_1.*
7	TX_DSP_CLEAR_INT_1	R/W	0	For DSP mode (bypass GDMA), F/W writes 1’b1 and then 1’b0 to clear TX interrupt. NOTE *Tx interrupt is to indicate that DSP can write audio data to TX_FIFO_1.*
6:2	INTR_CLR_0	R/W	0x0	Bit[2]: for the interrupt of “tx_fifo_full_intr” Bit[3]: for the interrupt of “rx_fifo_full_intr” Bit[4]: for the interrupt of “tx_fifo_empty_intr” Bit[5]: for the interrupt of “rx_fifo_empty_intr” Bit[6]: Reserved
1	RX_DSP_CLEAR_INT_0	R/W	0	For DSP mode (bypass GDMA), F/W writes 1’b1 and then 1’b0 to clear Rx interrupt. NOTE *Rx interrupt is to indicate that DSP can get audio data from RX_FIFO_0.*
0	TX_DSP_CLEAR_INT_0	R/W	0	For DSP mode (bypass GDMA), F/W writes 1’b1 and then 1’b0 to clear Tx interrupt. NOTE *Tx interrupt is to indicate that DSP can write audio data to TX_FIFO_0.*

29.3.3.5 REG_SP_TRX_COUNTER_STATUS

- **Name:** SPORT TRX Counter Status Register
- **Size:** 32 bits
- **Address offset:** 014h
- **Read/write access:** R

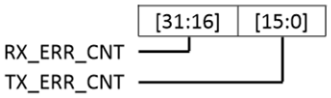
All information provided in this document is subject to legal disclaimers.
© REALTEK 2024. All rights reserved.



Bit	Symbol	Access	Reset	Description
31	SP_RESET_STATE	R	0	1'b1: sp_reset is enabled. 1'b0: sp_reset is disabled.
30	RSVD	-	0	Reserved
29:24	RX_DEPTH_CNT_1	R	0x0	RX_FIFO_1 depth counter status (MIC path)
23:22	RSVD	-	0	Reserved
21:16	TX_DEPTH_CNT_1	R	0x0	TX_FIFO_1 depth counter status (SPK path)
15:14	RSVD	-	0	Reserved
13:8	RX_DEPTH_CNT_0	R	0x0	RX_FIFO_0 depth counter status (MIC path)
7:6	RSVD	-	0	Reserved
5:0	TX_DEPTH_CNT_0	R	0x0	TX_FIFO_0 depth counter status (SPK path)

29.3.3.6 REG_SP_ERR

- **Name:** SPORT Error Register
- **Size:** 32 bits
- **Address offset:** 018h
- **Read/write access:** R



Bit	Symbol	Access	Reset	Description
31:16	RX_ERR_CNT	R	0x0	Rx error counter (MIC path) NOTE *This counter should always be zero if everything works well.*
15:0	TX_ERR_CNT	R	0x0	Tx error counter (SPK path) NOTE *This counter should always be zero if everything works well.*

29.3.3.7 REG_SR_TX_BCLK

- **Name:** SPORT Tx BCLK Register
- **Size:** 32 bits
- **Address offset:** 01Ch
- **Read/write access:** R/W



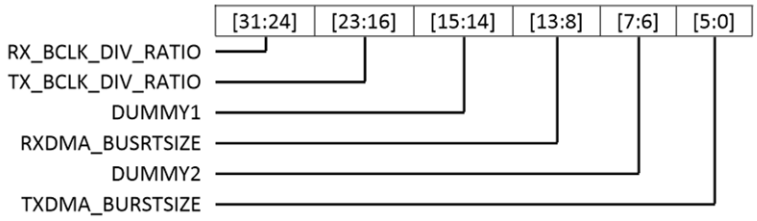
Bit	Symbol	Access	Reset	Description
31	TX_MI_NI_UPDATE	R/W	0	1'b1: to update "mi" and "ni" to get the new clock rate. This bit will be reset automatically when the update is done

All information provided in this document is subject to legal disclaimers.
© REALTEK 2024. All rights reserved.

30:16	TX_NI	R/W	0x30	BCLK = 40MHz*(ni/mi) For example: BCLK=3.072MHz=40MHz*(48/625)
15:0	TX_MI	R/W	0x271	

29.3.3.8 REG_SP_TX_LRCLK

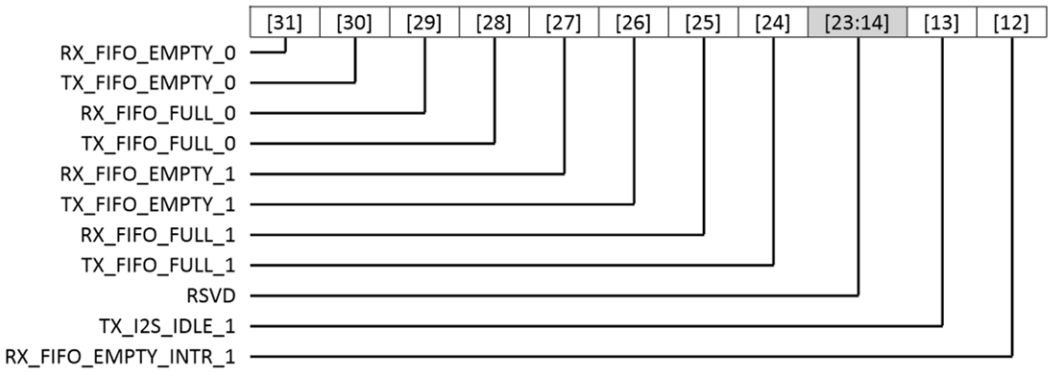
- **Name:** SPORT Tx LRCLK Register
- **Size:** 32 bits
- **Address offset:** 020h
- **Read/write access:** R/W



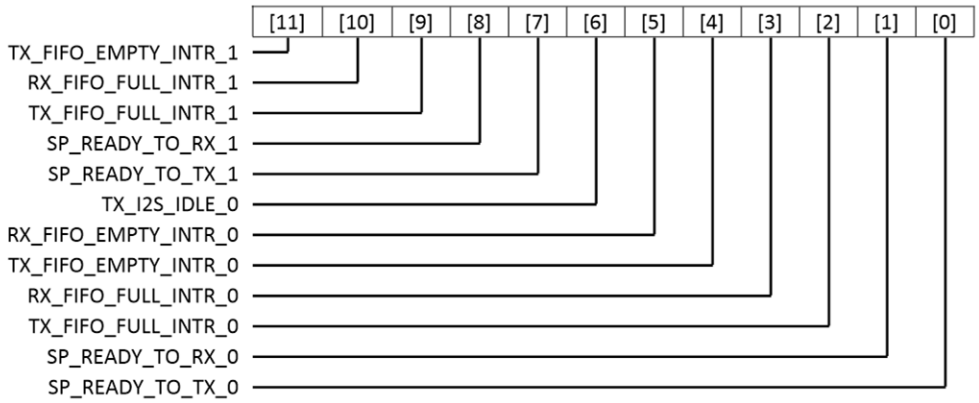
Bit	Symbol	Access	Reset	Description
31:24	RX_BCLK_DIV_RATIO	R/W	0x3F	Rx bclk even-bit integer divider. Used in “mode_40mhz” set as 1’b1. (rx_bclk_div_ratio + 1) is the number of “sck_out” cycles within a “ws_out_rx” cycle (1/fs). Default of (rx_bclk_div_ratio + 1) is 64. Set as 64 – 1 = 63. Only odd number supported. Maximum is 255.
23:16	TX_BCLK_DIV_RATIO	R/W	0x3F	Tx bclk even-bit integer divider. Used in “mode_40mhz” set as 1’b1. (tx_bclk_div_ratio + 1) is the number of “sck_out” cycles within a “ws_out_tx” cycle (1/fs). Default of (tx_bclk_div_ratio + 1) is 64. Set as 64 – 1 = 63. Only odd number supported. Maximum is 255.
15:14	DUMMY1	R/W	0x0	-
13:8	RXDMA_BUSRTSIZE	R/W	0x10	Rx DMA burst size
7:6	DUMMY2	R/W	0x0	-
5:0	TXDMA_BURSTSIZE	R/W	0x10	Tx DMA burst size

29.3.3.9 REG_SP_FIFO_CTRL

- **Name:** SPORT FIFO Control Register
- **Size:** 32 bits
- **Address offset:** 024h
- **Read/write access:** R



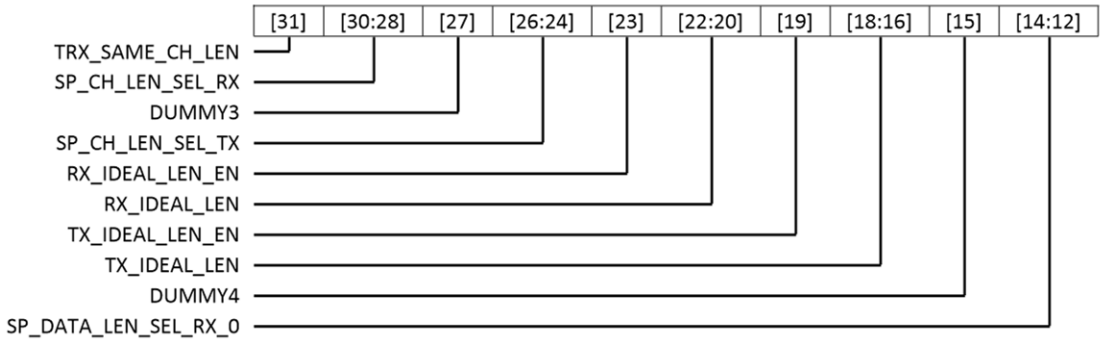
All information provided in this document is subject to legal disclaimers.
© REALTEK 2024. All rights reserved.



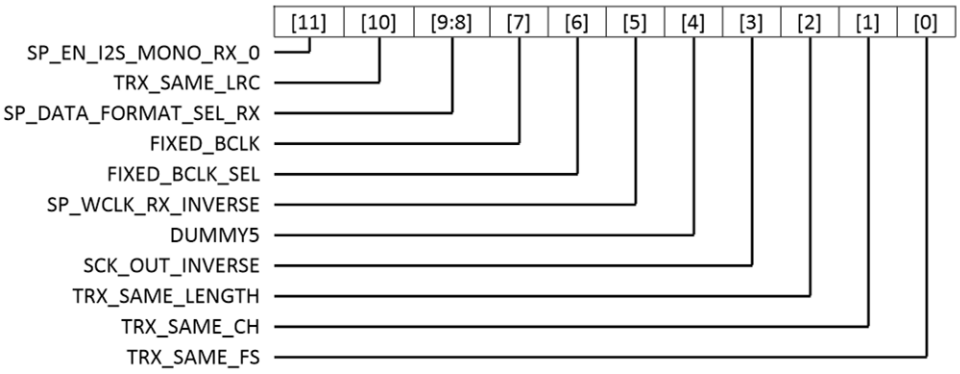
Bit	Symbol	Access	Reset	Description
31	RX_FIFO_EMPTY_0	R	1	1: RX_FIFO_0 is empty
30	TX_FIFO_EMPTY_0	R	1	1: TX_FIFO_0 is empty
29	RX_FIFO_FULL_0	R	0	1: RX_FIFO_0 is full
28	TX_FIFO_FULL_0	R	0	1: TX_FIFO_0 is full
27	RX_FIFO_EMPTY_1	R	1	1: RX_FIFO_1 is empty
26	TX_FIFO_EMPTY_1	R	1	1: TX_FIFO_1 is empty
25	RX_FIFO_FULL_1	R	0	1: RX_FIFO_1 is full
24	TX_FIFO_FULL_1	R	0	1: TX_FIFO_1 is full
23:14	RSVD	-	0	Reserved
13	TX_I2S_IDLE_1	R	0	1: Tx is working but FIFO_1 is empty.
12	RX_FIFO_EMPTY_INTR_1	R	0	1: RX_FIFO_1 is empty (MIC path)
11	TX_FIFO_EMPTY_INTR_1	R	0	1: TX_FIFO_1 is empty (SPK path)
10	RX_FIFO_FULL_INTR_1	R	0	1: RX_FIFO_1 is full (MIC path)
9	TX_FIFO_FULL_INTR_1	R	0	1: TX_FIFO_1 is full (SPK path)
8	SP_READY_TO_RX_1	R	0	1: It is ready to receive data (MIC path)
7	SP_READY_TO_TX_1	R	0	1: It is ready to send data out (SPK path)
6	TX_I2S_IDLE_0	R	0	1: Tx is working but FIFO_0 is empty.
5	RX_FIFO_EMPTY_INTR_0	R	0	1: RX_FIFO_0 is empty (MIC path)
4	TX_FIFO_EMPTY_INTR_0	R	0	1: TX_FIFO_0 is empty (SPK path)
3	RX_FIFO_FULL_INTR_0	R	0	1: RX_FIFO_0 is full (MIC path)
2	TX_FIFO_FULL_INTR_0	R	0	1: TX_FIFO_0 is full (SPK path)
1	SP_READY_TO_RX_0	R	0	1: It is ready to receive data (MIC path)
0	SP_READY_TO_TX_0	R	0	1: It is ready to send data out (SPK path)

29.3.3.10 REG_SP_FORMAT

- **Name:** SPORT Format Register
- **Size:** 32 bits
- **Address offset:** 028h
- **Read/write access:** R/W



All information provided in this document is subject to legal disclaimers.
© REALTEK 2024. All rights reserved.



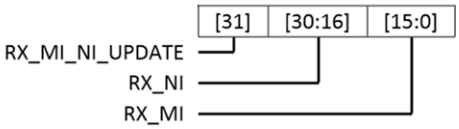
Bit	Symbol	Access	Reset	Description
31	TRX_SAME_CH_LEN	R/W	0	1: Tx (SPK path) and Rx (MIC path) have the same channel length.
30:28	SP_CH_LEN_SEL_RX	R/W	0x4	3'b000: 16 bits 3'b001: 20 bits 3'b010: 24 bits 3'b100: 32 bits
27	DUMMY3	R/W	0	-
26:24	SP_CH_LEN_SEL_TX	R/W	0x4	3'b000: 16 bits 3'b001: 20 bits 3'b010: 24 bits 3'b100: 32 bits
23	RX_IDEAL_LEN_EN	R/W	0	Function enable of rx_ideal_len.
22:20	RX_IDEAL_LEN	R/W	0x0	sd_in can be received 1 ~ 8 (= rx_ideal_len + 1) BCLK cycle latter.
19	TX_IDEAL_LEN_EN	R/W	0	Function enable of tx_ideal_len. PCMA SDO will be delayed 1 LRCK.
18:16	TX_IDEAL_LEN	R/W	0x0	sd_out can be sent 1 ~ 8 (= tx_ideal_len + 1) BCLK cycle earlier.
15	DUMMY4	R/W	0	-
14:12	SP_DATA_LEN_SEL_RX_0	R/W	0x0	Data length of MIC path and it is valid if "trx_same_length" == 1'b0. 3'b000: 16 bits 3'b001: 20 bits 3'b010: 24 bits 3'b100: 32 bits
11	SP_EN_I2S_MONO_RX_0	R/W	0	Channel format of MIC path and it is valid if "trx_same_ch" == 1'b0. 1: mono 0: stereo
10	TRX_SAME_LRC	R/W	0	1: "ws_out_rx" is as same as "ws_out_tx"
9:8	SP_DATA_FORMAT_SEL_RX	R/W	0x0	Data format of MIC path and it is valid if "trx_same_fs" == 1'b0. 2'b00: I2S 2'b01: Left-justified 2'b10: PCM mode A 2'b11: PCM mode B
7	FIXED_BCLK	R/W	0	1: Refer to the description of "fixed_bclk_sel" 0: BCLK = dsp_clk/2 when "mode_40mhz = 0"
6	FIXED_BCLK_SEL	R/W	0	0: BCLK is fixed at dsp_src_clk/4 1: BCLK is fixed at dsp_src_clk/2
5	SP_WCLK_RX_INVERSE	R/W	0	1: invert the phase of "ws_out_rx" which is also called as "ADCLRC"
4	DUMMY5	R/W	0	-
3	SCK_OUT_INVERSE	R/W	0	1: invert the phase of "sck_out"
2	TRX_SAME_LENGTH	R/W	1	1: Tx (SPK path) and Rx (MIC path) have the same data length. Both are either 16 or 24 bits
1	TRX_SAME_CH	R/W	1	1: Tx (SPK path) and Rx (MIC path) have the same channel setting. Both are either stereo or mono
0	TRX_SAME_FS	R/W	1	1: Tx (SPK path) and Rx (MIC path) have the same sampling rate

29.3.3.11 REG_SP_RX_BCLK

- **Name:** SPORT Rx BCLK Register

All information provided in this document is subject to legal disclaimers.
© REALTEK 2024. All rights reserved.

- **Size:** 32 bits
- **Address offset:** 02Ch
- **Read/write access:** R/W



Bit	Symbol	Access	Reset	Description
31	RX_MI_NI_UPDATE	R/W	0	1'b1: to update "mi" and "ni" to get the new clock rate. This bit will be reset automatically when the update is done
30:16	RX_NI	R/W	0x30	BCLK = 40MHz*(ni/mi) For example: BCLK=3.072MHz=40MHz*(48/625)
15:0	RX_MI	R/W	0x271	

29.3.3.12 REG_SP_RX_LRCLK

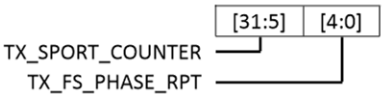
- **Name:** SPORT Rx LRCLK Register
- **Size:** 32 bits
- **Address offset:** 030h
- **Read/write access:** R/W



Bit	Symbol	Access	Reset	Description
31	CLR_TX_SPORT_RDY	WA0	0	0x001: clear tx_sport_interrupt signal This control bit is "write 1 clear" type For read, the read data is from clr_tx_sport_rdy
30	EN_TX_SPORT_INTERRUPT	R/W	0	Enable tx_sport_interrupt
29	EN_FS_PHASE_LATCH	WA0	0	0x001: Latch the value of tx_fs_phase_rpt, rx_fs_phase_rpt, tx_sport_sounter, rx_sport_sounter at the same time. This control bit is "write 1 clear" type
28:27	DUMMY6	R/W	0x0	
26:0	TX_SPORT_COMPARE_VAL	R/W	0x40	X = (tx_sport_compare_val). When counter equal X. SPORT will send tx_sport_interrupt to DSP. FW should take care X={32~134217727}

29.3.3.13 REG_SP_DSP_COUNTER

- **Name:** SPORT DSP Counter Register
- **Size:** 32 bits
- **Address offset:** 034h
- **Read/write access:** R

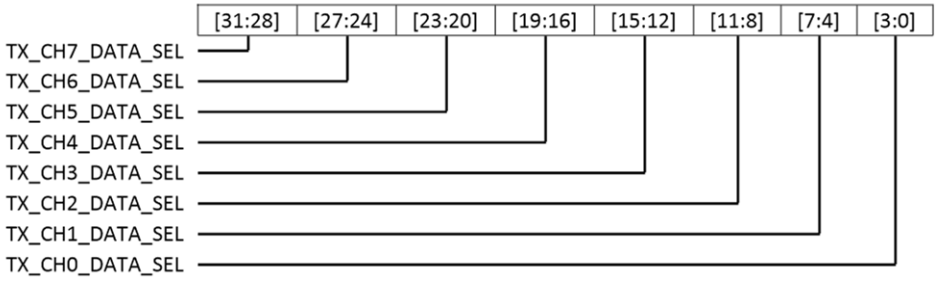


Bit	Symbol	Access	Reset	Description
31:5	TX_SPORT_COUNTER	R	0x0	For DSP read instant Tx SPORT counter value, counter down
4:0	TX_FS_PHASE_RPT	R	0x0	Report Tx phase

All information provided in this document is subject to legal disclaimers.
© REALTEK 2024. All rights reserved.

29.3.3.14 REG_SP_DIRECT_CTRL0

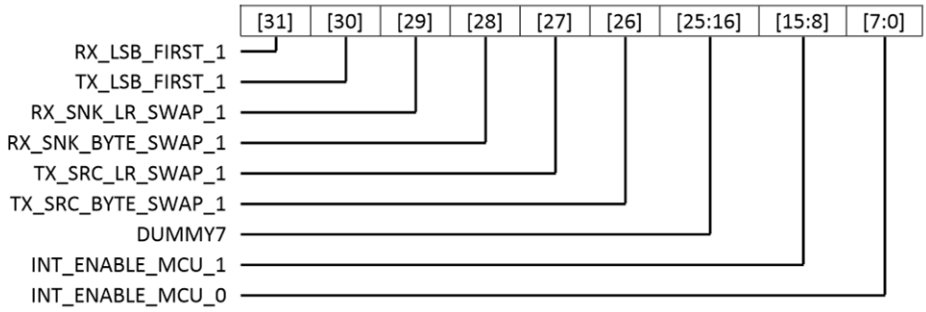
- **Name:** SPORT Direct Control Register 0
- **Size:** 32 bits
- **Address offset:** 03Ch
- **Read/write access:** R/W



Bit	Symbol	Access	Reset	Description
31:28	TX_CH7_DATA_SEL	R/W	0x7	4'h0: tx_fifo_0_reg_0_l 4'h1: tx_fifo_0_reg_0_r 4'h2: tx_fifo_0_reg_1_l 4'h3: tx_fifo_0_reg_1_r 4'h4: tx_fifo_1_reg_0_l 4'h5: tx_fifo_1_reg_0_r 4'h6: tx_fifo_1_reg_1_l 4'h7: tx_fifo_1_reg_1_r 4'h8: direct_reg_7
27:24	TX_CH6_DATA_SEL	R/W	0x6	4'h8: direct_reg_6
23:20	TX_CH5_DATA_SEL	R/W	0x5	4'h8: direct_reg_5
19:16	TX_CH4_DATA_SEL	R/W	0x4	4'h8: direct_reg_4
15:12	TX_CH3_DATA_SEL	R/W	0x3	4'h8: direct_reg_3
11:8	TX_CH2_DATA_SEL	R/W	0x2	4'h8: direct_reg_2
7:4	TX_CH1_DATA_SEL	R/W	0x1	4'h8: direct_reg_1
3:0	TX_CH0_DATA_SEL	R/W	0x0	4'h8: direct_reg_0

29.3.3.15 REG_SP_FIFO_IRQ

- **Name:** SPORT FIFO IRQ Register
- **Size:** 32 bits
- **Address offset:** 044h
- **Read/write access:** R/W



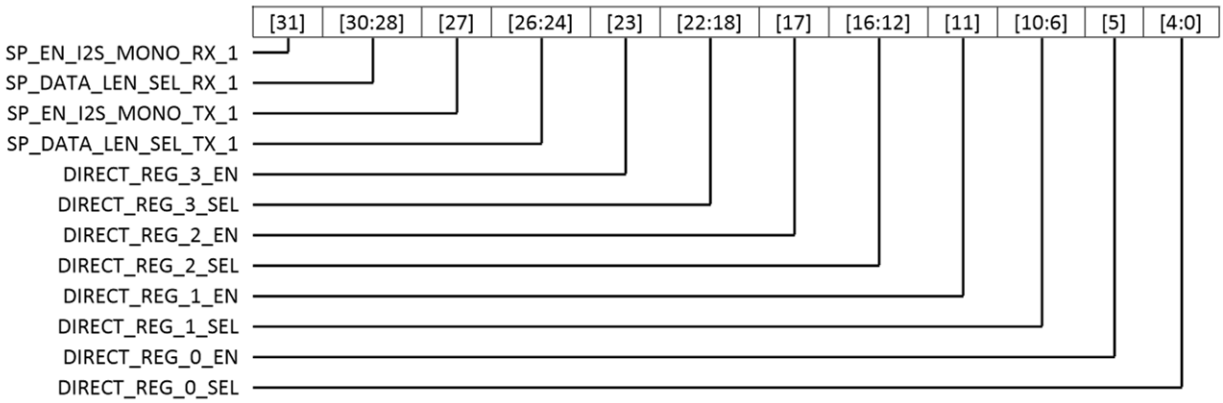
Bit	Symbol	Access	Reset	Description
31	RX_LSB_FIRST_1	R/W	0	1'b0: MSB first when Tx 1'b1: LSB first
30	TX_LSB_FIRST_1	R/W	0	1'b0: MSB first when Tx 1'b1: LSB first
29	RX_SNK_LR_SWAP_1	R/W	0	1'b1: swap L/R audio samples written to the sink memory of RX_FIFO_1

All information provided in this document is subject to legal disclaimers.
© REALTEK 2024. All rights reserved.

28	RX_SNK_BYTE_SWAP_1	R/W	0	1'b1: swap H/L bytes written to the sink memory of RX_FIFO_1
27	TX_SRC_LR_SWAP_1	R/W	0	1'b1: swap L/R audio samples read from the source memory of TX_FIFO_1
26	TX_SRC_BYTE_SWAP_1	R/W	0	1'b1: swap H/L bytes read from the source memory of TX_FIFO_1
25:16	DUMMY7	R/W	0x0	-
15:8	INT_ENABLE_MCU_1	R/W	0x0	Bit8]: for the interrupt of "sp_ready_to_tx" Bit9]: for the interrupt of "sp_ready_to_rx" Bit10]: for the interrupt of "tx_fifo_full_intr" Bit11]: for the interrupt of "rx_fifo_full_intr" Bit12]: for the interrupt of "tx_fifo_empty_intr" Bit13]: for the interrupt of "rx_fifo_empty_intr" Bit14]: for the interrupt of "tx_i2s_idle" Bit15]: reserved
7:0	INT_ENABLE_MCU_0	R/W	0x0	Bit0]: for the interrupt of "sp_ready_to_tx" Bit1]: for the interrupt of "sp_ready_to_rx" Bit2]: for the interrupt of "tx_fifo_full_intr" Bit3]: for the interrupt of "rx_fifo_full_intr" Bit4]: for the interrupt of "tx_fifo_empty_intr" Bit5]: for the interrupt of "rx_fifo_empty_intr" Bit6]: for the interrupt of "tx_i2s_idle" Bit7]: reserved

29.3.3.16 REG_SP_DIRECT_CTRL1

- **Name:** SPORT Direct Control Register 1
- **Size:** 32 bits
- **Address offset:** 048h
- **Read/write access:** R/W



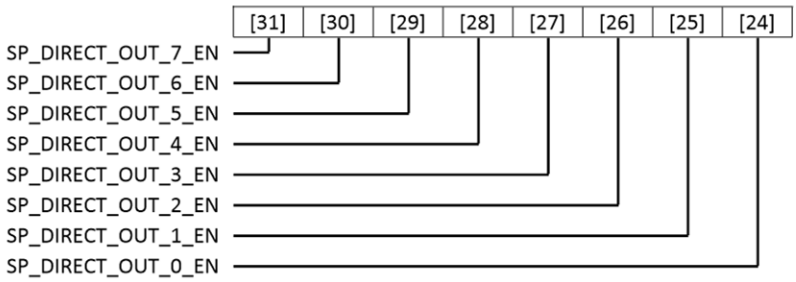
Bit	Symbol	Access	Reset	Description
31	SP_EN_I2S_MONO_RX_1	R/W	0	Channel format of MIC path and it is valid if "trx_same_ch" == 1'b0. 1'b1: mono 1'b0: stereo
30:28	SP_DATA_LEN_SEL_RX_1	R/W	0x0	Data length of MIC path and it is valid if "trx_same_length" == 1'b0. 3'b000: 16 bits 3'b001: 20 bits 3'b010: 24 bits 3'b100: 32 bits
27	SP_EN_I2S_MONO_TX_1	R/W	0	1'b1: mono 1'b0: stereo
26:24	SP_DATA_LEN_SEL_TX_1	R/W	0x0	3'b000: 16 bits 3'b001: 20 bits 3'b010: 24 bits 3'b100: 32 bits
23	DIRECT_REG_3_EN	R/W	0	1'b1: Enable direct_reg_3.
22:18	DIRECT_REG_3_SEL	R/W	0x0	5'h0: spa_direct_in_0 5'h1: spa_direct_in_1

All information provided in this document is subject to legal disclaimers.
© REALTEK 2024. All rights reserved.

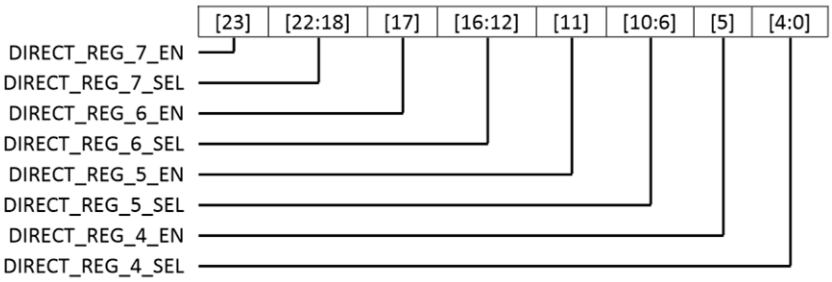
				5'h2: spa_direct_in_2 5'h3: spa_direct_in_3 5'h4: spa_direct_in_4 5'h5: spa_direct_in_5 5'h6: spa_direct_in_6 5'h7: spa_direct_in_7 5'h8: spb_direct_in_0 5'h9: spb_direct_in_1 5'ha: spb_direct_in_2 5'hb: spb_direct_in_3 5'hc: spb_direct_in_4 5'hd: spb_direct_in_5 5'he: spb_direct_in_6 5'hf: spb_direct_in_7 5'h10: spc_direct_in_0 5'h11: spc_direct_in_1 5'h12: spc_direct_in_2 5'h13: spc_direct_in_3 5'h14: spc_direct_in_4 5'h15: spc_direct_in_5 5'h16: spc_direct_in_6 5'h17: spc_direct_in_7 5'h18: sp0_direct_in_tx_fifo_0_reg_0_l 5'h19: sp0_direct_in_tx_fifo_0_reg_0_r 5'h1a: sp0_direct_in_tx_fifo_0_reg_1_l 5'h1b: sp0_direct_in_tx_fifo_0_reg_1_r 5'h1c: TDM_RX_CH3 SPORT0: a = 1, b = 2, c = 3 SPORT1: a = 0, b = 2, c = 3 SPORT2: a = 0, b = 1, c = 3 SPORT3: a = 0, b = 1, c = 2
17	DIRECT_REG_2_EN	R/W	0	1'b1: Enable direct_reg_2.
16:12	DIRECT_REG_2_SEL	R/W	0x0	5'h1c: TDM_RX_CH2
11	DIRECT_REG_1_EN	R/W	0	1'b1: Enable direct_reg_1.
10:6	DIRECT_REG_1_SEL	R/W	0x0	5'h1c: TDM_RX_CH1
5	DIRECT_REG_0_EN	R/W	0	1'b1: Enable direct_reg_0.
4:0	DIRECT_REG_0_SEL	R/W	0x0	5'h1c: TDM_RX_CH0

29.3.3.17 REG_SP_DIRECT_CTRL2

- **Name:** SPORT Direct Control Register 2
- **Size:** 32 bits
- **Address offset:** 04Ch
- **Read/write access:** R/W



All information provided in this document is subject to legal disclaimers.
© REALTEK 2024. All rights reserved.

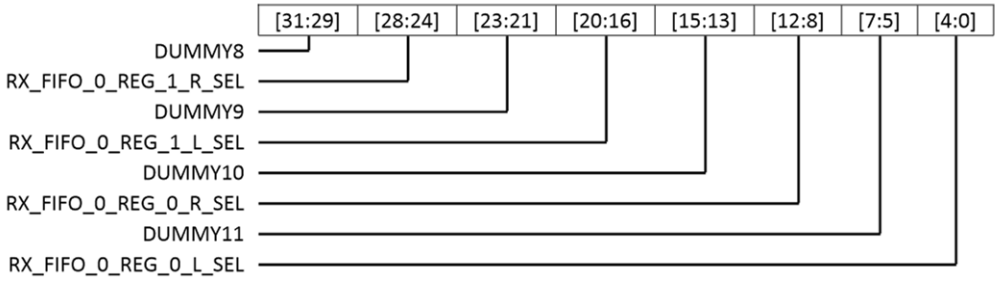


Bit	Symbol	Access	Reset	Description
31	SP_DIRECT_OUT_7_EN	R/W	0	Enable sp_direct_out_7
30	SP_DIRECT_OUT_6_EN	R/W	0	Enable sp_direct_out_6
29	SP_DIRECT_OUT_5_EN	R/W	0	Enable sp_direct_out_5
28	SP_DIRECT_OUT_4_EN	R/W	0	Enable sp_direct_out_4
27	SP_DIRECT_OUT_3_EN	R/W	0	Enable sp_direct_out_3
26	SP_DIRECT_OUT_2_EN	R/W	0	Enable sp_direct_out_2
25	SP_DIRECT_OUT_1_EN	R/W	0	Enable sp_direct_out_1
24	SP_DIRECT_OUT_0_EN	R/W	0	Enable sp_direct_out_0
23	DIRECT_REG_7_EN	R/W	0	1'b1: Enable direct_reg_7
22:18	DIRECT_REG_7_SEL	R/W	0x0	5'h0: spa_direct_in_0 5'h1: spa_direct_in_1 5'h2: spa_direct_in_2 5'h3: spa_direct_in_3 5'h4: spa_direct_in_4 5'h5: spa_direct_in_5 5'h6: spa_direct_in_6 5'h7: spa_direct_in_7 5'h8: spb_direct_in_0 5'h9: spb_direct_in_1 5'ha: spb_direct_in_2 5'hb: spb_direct_in_3 5'hc: spb_direct_in_4 5'hd: spb_direct_in_5 5'he: spb_direct_in_6 5'hf: spb_direct_in_7 5'h10: spc_direct_in_0 5'h11: spc_direct_in_1 5'h12: spc_direct_in_2 5'h13: spc_direct_in_3 5'h14: spc_direct_in_4 5'h15: spc_direct_in_5 5'h16: spc_direct_in_6 5'h17: spc_direct_in_7 5'h18: sp0_direct_in_tx_fifo_0_reg_0_l 5'h19: sp0_direct_in_tx_fifo_0_reg_0_r 5'h1a: sp0_direct_in_tx_fifo_0_reg_1_l 5'h1b: sp0_direct_in_tx_fifo_0_reg_1_r 5'h1c: TDM_RX_CH7 SPORT0: a = 1, b = 2, c = 3 SPORT1: a = 0, b = 2, c = 3 SPORT2: a = 0, b = 1, c = 3 SPORT3: a = 0, b = 1, c = 2
17	DIRECT_REG_6_EN	R/W	0	1'b1: Enable direct_reg_6.
16:12	DIRECT_REG_6_SEL	R/W	0x0	5'h1c: TDM_RX_CH6
11	DIRECT_REG_5_EN	R/W	0	1'b1: Enable direct_reg_5.
10:6	DIRECT_REG_5_SEL	R/W	0x0	5'h1c: TDM_RX_CH5
5	DIRECT_REG_4_EN	R/W	0	1'b1: Enable direct_reg_4.
4:0	DIRECT_REG_4_SEL	R/W	0x0	5'h1c: TDM_RX_CH4

All information provided in this document is subject to legal disclaimers.
© REALTEK 2024. All rights reserved.

29.3.3.18 REG_SP_DIRECT_CTRL3

- **Name:** SPORT Direct Control Register 3
- **Size:** 32 bits
- **Address offset:** 054h
- **Read/write access:** R/W

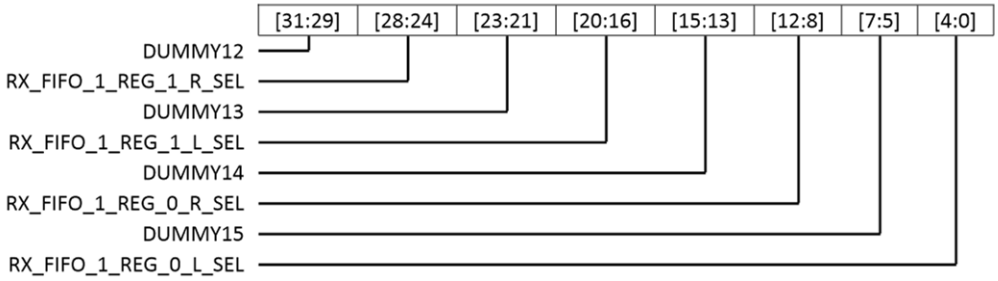


Bit	Symbol	Access	Reset	Description
31:29	DUMMY8	R/W	0x0	-
28:24	RX_FIFO_0_REG_1_R_SEL	R/W	0x3	5'd0: RX_CH0_data_out (MIC path) 5'd1: RX_CH1_data_out (MIC path) 5'd2: RX_CH2_data_out (MIC path) 5'd3: RX_CH3_data_out (MIC path) 5'd4: RX_CH4_data_out (MIC path) 5'd5: RX_CH5_data_out (MIC path) 5'd6: RX_CH6_data_out (MIC path) 5'd7: RX_CH7_data_out (MIC path) 5'd8: spa_direct_in_0 5'd9: spa_direct_in_1 5'd10: spa_direct_in_2 5'd11: spa_direct_in_3 5'd12: spa_direct_in_4 5'd13: spa_direct_in_5 5'd14: spa_direct_in_6 5'd15: spa_direct_in_7 5'd16: spb_direct_in_0 5'd17: spb_direct_in_1 5'd18: spb_direct_in_2 5'd19: spb_direct_in_3 5'd20: spb_direct_in_4 5'd21: spb_direct_in_5 5'd22: spb_direct_in_6 5'd23: spb_direct_in_7 5'd24: spc_direct_in_0 5'd25: spc_direct_in_1 5'd26: spc_direct_in_2 5'd27: spc_direct_in_3 5'd28: spc_direct_in_4 5'd29: spc_direct_in_5 5'd30: spc_direct_in_6 5'd31: spc_direct_in_7 SPORT0: a = 1, b = 2, c = 3 SPORT1: a = 0, b = 2, c = 3 SPORT2: a = 0, b = 1, c = 3 SPORT3: a = 0, b = 1, c = 2
23:21	DUMMY9	R/W	0x0	
20:16	RX_FIFO_0_REG_1_L_SEL	R/W	0x2	
15:13	DUMMY10	R/W	0x0	
12:8	RX_FIFO_0_REG_0_R_SEL	R/W	0x1	
7:5	DUMMY11	R/W	0x0	
4:0	RX_FIFO_0_REG_0_L_SEL	R/W	0x0	

All information provided in this document is subject to legal disclaimers.
© REALTEK 2024. All rights reserved.

29.3.3.19 REG_SP_DIRECT_CTRL4

- **Name:** SPORT Direct Control Register 4
- **Size:** 32 bits
- **Address offset:** 058h
- **Read/write access:** R/W

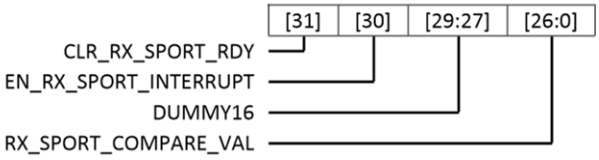


Bit	Symbol	Access	Reset	Description
31:29	DUMMY12	R/W	0x0	
28:24	RX_FIFO_1_REG_1_R_SEL	R/W	0x7	5'd0: RX_CH0_data_out (MIC path) 5'd1: RX_CH1_data_out (MIC path) 5'd2: RX_CH2_data_out (MIC path) 5'd3: RX_CH3_data_out (MIC path) 5'd4: RX_CH4_data_out (MIC path) 5'd5: RX_CH5_data_out (MIC path) 5'd6: RX_CH6_data_out (MIC path) 5'd7: RX_CH7_data_out (MIC path) 5'd8: spa_direct_in_0 5'd9: spa_direct_in_1 5'd10: spa_direct_in_2 5'd11: spa_direct_in_3 5'd12: spa_direct_in_4 5'd13: spa_direct_in_5 5'd14: spa_direct_in_6 5'd15: spa_direct_in_7 5'd16: spb_direct_in_0 5'd17: spb_direct_in_1 5'd18: spb_direct_in_2 5'd19: spb_direct_in_3 5'd20: spb_direct_in_4 5'd21: spb_direct_in_5 5'd22: spb_direct_in_6 5'd23: spb_direct_in_7 5'd24: spc_direct_in_0 5'd25: spc_direct_in_1 5'd26: spc_direct_in_2 5'd27: spc_direct_in_3 5'd28: spc_direct_in_4 5'd29: spc_direct_in_5 5'd30: spc_direct_in_6 5'd31: spc_direct_in_7 SPORT0: a = 1, b = 2, c = 3 SPORT1: a = 0, b = 2, c = 3 SPORT2: a = 0, b = 1, c = 3 SPORT3: a = 0, b = 1, c = 2
23:21	DUMMY13	R/W	0x0	
20:16	RX_FIFO_1_REG_1_L_SEL	R/W	0x6	
15:13	DUMMY14	R/W	0x0	
12:8	RX_FIFO_1_REG_0_R_SEL	R/W	0x5	
7:5	DUMMY15	R/W	0x0	
4:0	RX_FIFO_1_REG_0_L_SEL	R/W	0x4	

All information provided in this document is subject to legal disclaimers.
© REALTEK 2024. All rights reserved.

29.3.3.20 REG_SP_RX_COUNTER1

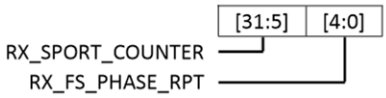
- **Name:** SPORT Rx Counter Register 1
- **Size:** 32 bits
- **Address offset:** 05Ch
- **Read/write access:** R/W



Bit	Symbol	Access	Reset	Description
31	CLR_RX_SPORT_RDY	WA0	0	X = (tx_sport_compare_val). When counter equal X. SPORT will send tx_sport_interrupt to DSP. FW should take care X={32~8191}
30	EN_RX_SPORT_INTERRUPT	R/W	0	Enable Rx sport interrupt.
29:27	DUMMY16	R/W	0x0	
26:0	RX_SPORT_COMPARE_VAL	R/W	0x40	X = (rx_sport_compare_val). When counter equal X. SPORT will send rx_sport_interrupt to DSP. FW should take care X={32~134217727}

29.3.3.21 REG_SP_RX_COUNTER2

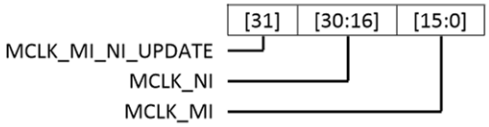
- **Name:** SPORT Rx Counter Register 2
- **Size:** 32 bits
- **Address offset:** 060h
- **Read/write access:** R



Bit	Symbol	Access	Reset	Description
31:5	RX_SPORT_COUNTER	R	0x0	For DSP read instant Rx SPORT counter value, counter down
4:0	RX_FS_PHASE_RPT	R	0x0	Report Rx phase

29.3.3.22 REG_SP_MCLK

- **Name:** SPORT MCLK Register
- **Size:** 32 bits
- **Address offset:** 064h
- **Read/write access:** R/W

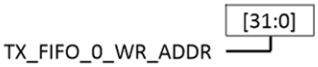


Bit	Symbol	Access	Reset	Description
31	MCLK_MI_NI_UPDATE	R/W	0	1'b1: to update "mi" and "ni" to get the new clock rate. This bit will be reset automatically when the update is done
30:16	MCLK_NI	R/W	0xc0	mclk_out = 40MHz*(ni/mi) For example: mclk_out=3.072MHz=40MHz*(48/625)
15:0	MCLK_MI	R/W	0x271	

All information provided in this document is subject to legal disclaimers.
© REALTEK 2024. All rights reserved.

29.3.3.23 REG_TX_FIFO_0_WR_ADDR

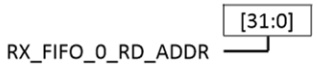
- **Name:** TX FIFO 0 Write Address Register
- **Size:** 32 bits
- **Address offset:** 800h
- **Read/write access:** R/W



Bit	Symbol	Access	Reset	Description
31:0	TX_FIFO_0_WR_ADDR	W	0x0	TX_FIFO_0 write address

29.3.3.24 REG_RX_FIFO_0_RD_ADDR

- **Name:** RX FIFO 0 Read Address Register
- **Size:** 32 bits
- **Address offset:** 880h
- **Read/write access:** R



Bit	Symbol	Access	Reset	Description
31:0	RX_FIFO_0_RD_ADDR	R	0x0	RX_FIFO_0 read address

29.3.3.25 REG_TX_FIFO_1_WR_ADDR

- **Name:** TX FIFO 1 Write Address Register
- **Size:** 32 bits
- **Address offset:** 900h
- **Read/write access:** R/W



Bit	Symbol	Access	Reset	Description
31:0	TX_FIFO_1_WR_ADDR	W	0x0	TX_FIFO_1 write address

29.3.3.26 REG_RX_FIFO_1_RD_ADDR

- **Name:** RX FIFO 1 Read Address Register
- **Size:** 32 bits
- **Address offset:** 980h
- **Read/write access:** R



Bit	Symbol	Access	Reset	Description
31:0	RX_FIFO_1_RD_ADDR	R	0x0	RX_FIFO_1 read address

All information provided in this document is subject to legal disclaimers.
© REALTEK 2024. All rights reserved.

29.4 Digital Microphone Interface

29.4.1 General Description

The digital microphone (DMIC) interface is for digital microphone, and supports 2-channel digital microphone recording. *Figure 29-25* shows the details block of digital microphone interface.

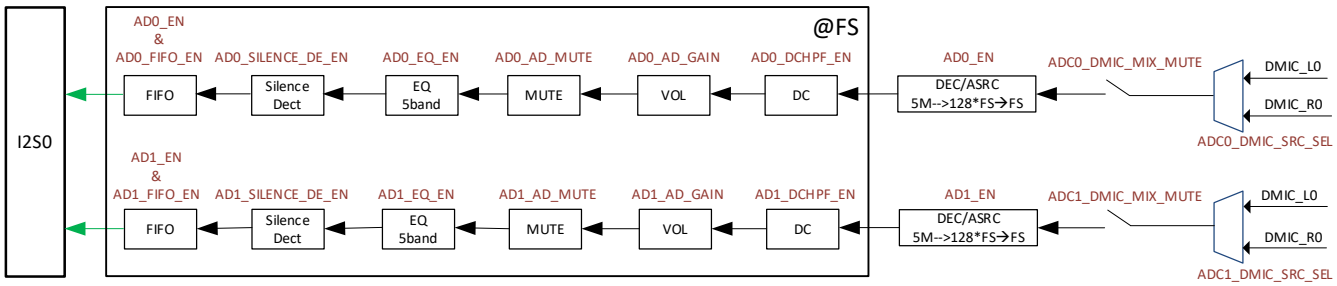


Figure 29-25 Audio recording path configuration

The DMIC interface has the following features:

- 8kHz/11.025kHz/12kHz/16kHz/22.5kHz/24kHz/32kHz/44.1kHz/48kHz/88.2kHz/96kHz for digital microphone interface
- Asynchronous sample rate converter (ASRC) for each interface
- Configurable 0-5 band EQ
- Adjustable digital volume control
- For digital volume control, supports zero-crossing detection to minimize audible artifacts
- DC remove function

29.4.2 DMIC Interface Data Path

Figure 29-26 shows the recording data path of digital microphone interface. In the recording path, the input source is 2-channel DMIC.

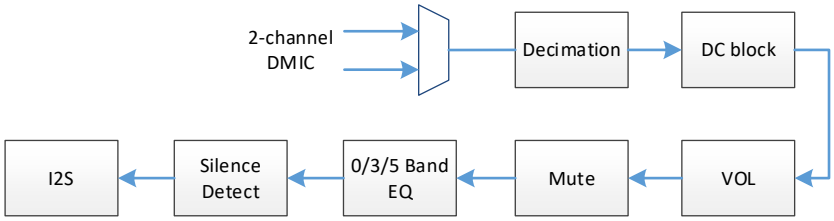


Figure 29-26 Digital microphone interface recording data path

29.4.3 DMIC Interface Function

- Recording DMIC path for sampling rate 8kHz/11.025kHz/12kHz/16kHz/32kHz/22.05kHz/44.1kHz/48kHz/88.2kHz/96kHz
- ASRC (asynchronous sample rate converter)
- When enabling ASRC function, the clock sources from ad_fs and BCLK0 (or BCLK1) are allowed to be asynchronous. The ASRC technology can ensure data accuracy and keep audio performance under clock source asynchronous.
- The ADC digital part supports digital volume control, and the gain is between 48dB and -17.625dB in 0.375dB/step.
- There is a high pass filter for DC offset
- Zero-crossing function

If the volume is adjusted while the signal is a non-zero value, an audible click can occur, as shown in *Figure 29-27*.

All information provided in this document is subject to legal disclaimers.
© REALTEK 2024. All rights reserved.

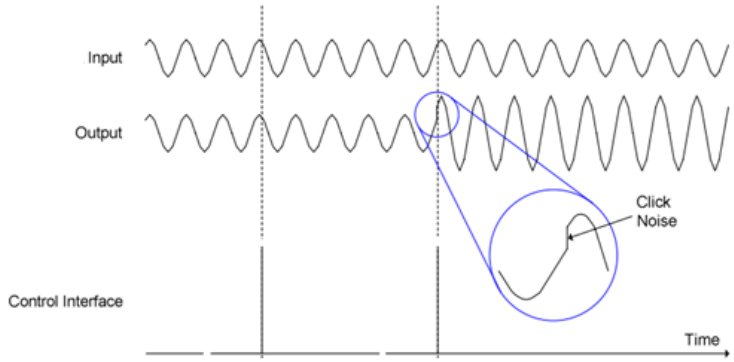


Figure 29-27 Click noise without zero-crossing

In order to prevent this click noise, a zero-crossing function is provided. When enabled, this will cause the volume to update only when a zero-crossing occurs, minimizing click noise, as shown in *Figure 29-28*.

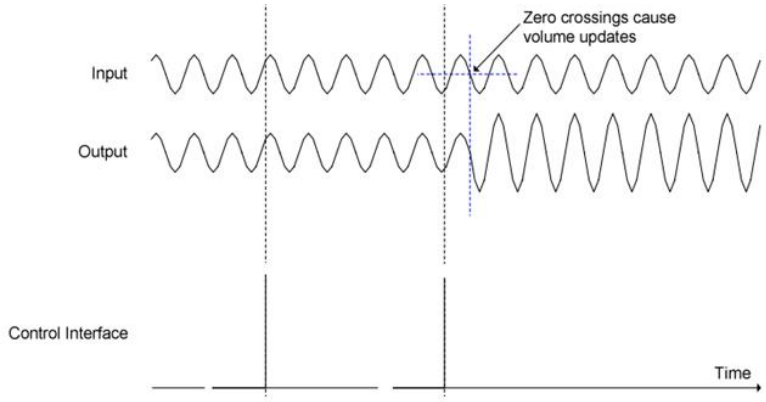


Figure 29-28 Minimizing click noise with zero-crossing

When the signal is very quiet and consists of mainly of noise, zero-crossing cannot be met, now the gain will change with steps, as shown in *Figure 29-29*.

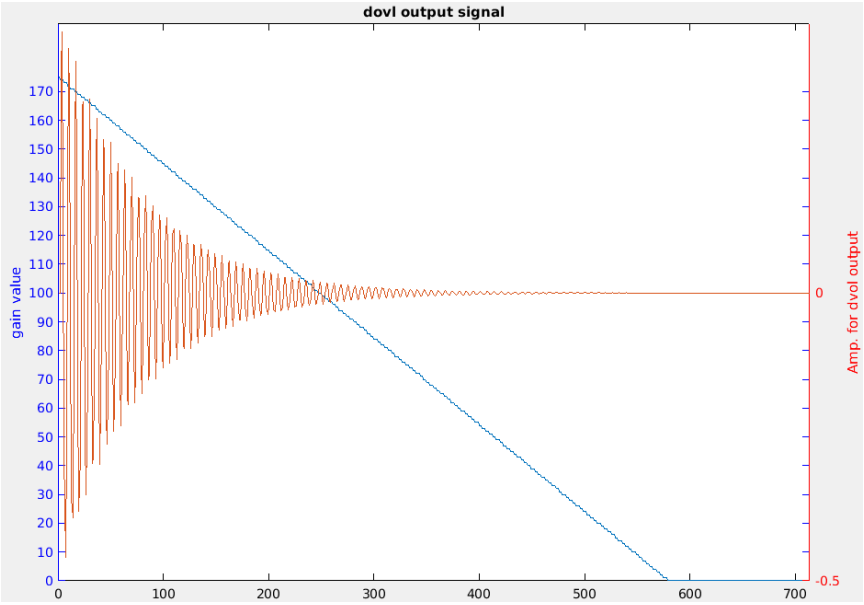


Figure 29-29 Gain update with steps as zero-crossing

- Equalizer block
  The equalizer block cascades 0-5 bands of equalizer to tailor the frequency characteristics of the recording system according to user preferences and to emulate environment sound.
- DC remove function block

All information provided in this document is subject to legal disclaimers.
© REALTEK 2024. All rights reserved.

A high pass filter is implemented for dc offset. The high pass filter is mainly for ADC recording used. The cut-off frequency of filter is programmable and is varied according to different sample rates. The filter is used to remove DC offset at normal conditions.

- Silence detector block
  The Silence detector is used to reduce the noise floor for DAC path or ADC path. When the input signal is below silence level, the input signal will be reduced to suppress the background noise. The reducing level can be set by registers.

29.4.4 DMIC Interface Register

The base address of DMIC interface registers is 0x41106000.

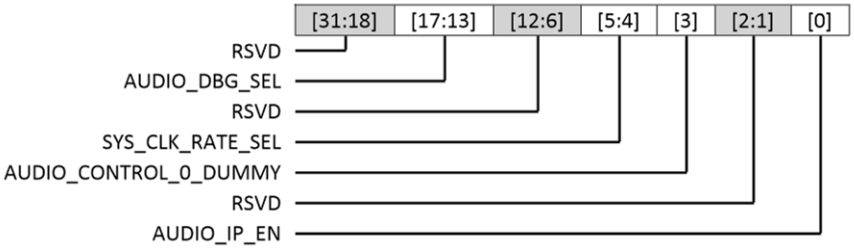
Name	Address offset	Access	Description
REG_AUDIO_CONTROL_0	000h	R/W	Audio Control Register 0
REG_AUDIO_CONTROL_1	004h	R/W	Audio Control Register 1
REG_CLOCK_CONTROL_1	008h	R/W	Clock Control Register 1
REG_CLOCK_CONTROL_2	00Ch	R/W	Clock Control Register 2
REG_CLOCK_CONTROL_3	010h	R/W	Clock Control Register 3
REG_CLOCK_CONTROL_4	014h	R/W	Clock Control Register 4
REG_CLOCK_CONTROL_5	018h	R/W	Clock Control Register 5
REG_CLOCK_CONTROL_6	01Ch	R/W	Clock Control Register 6
REG_CLOCK_CONTROL_7	020h	R/W	Clock Control Register 7
REG_ASRC_CONTROL_2	02Ch	R/W	ASRC Control Register 2
REG_ASRC_CONTROL_3	030h	R/W	ASRC Control Register 3
REG_I2S_0_CONTROL	03Ch	R/W	I2S 0 Control Register
REG_I2S_0_CONTROL_1	040h	R/W	I2S 0 Control Register 1
REG_ADC_0_CONTROL_0	050h	R/W	ADC Channel 0 Control Register 0
REG_ADC_0_CONTROL_1	054h	R/W	ADC Channel 0 Control Register 1
REG_ADC_1_CONTROL_0	058h	R/W	ADC Channel 1 Control Register 0
REG_ADC_1_CONTROL_1	05Ch	R/W	ADC Channel 1 Control Register 1
REG_ADC_ALIGN_CONTROL_REG_0	0ACh	R/W	ADC Align Control Register 0
REG_ADC_ALIGN_CONTROL_REG	0B0h	R/W	ADC Align Control Register
REG_ADC_0_SILENCE_CONTROL	100h	R/W	ADC Channel 0 Silence Control Register
REG_ADC_1_SILENCE_CONTROL	104h	R/W	ADC Channel 1 Silence Control Register
REG_ADC_0_EQ_CTRL	400h	R/W	ADC Channel 0 EQ Control Register
REG_ADC_1_EQ_CTRL	468h	R/W	ADC Channel 1 EQ Control Register
REG_ADC_1_BIQUAD_H0_0	46Ch	R/W	ADC Channel 1 EQ 0 Band Biquad H0 Register
REG_ADC_1_BIQUAD_B1_0	470h	R/W	ADC Channel 1 EQ 0 Band Biquad B1 Register
REG_ADC_1_BIQUAD_B2_0	474h	R/W	ADC Channel 1 EQ 0 Band Biquad B2 Register
REG_ADC_1_BIQUAD_A1_0	478h	R/W	ADC Channel 1 EQ 0 Band Biquad A1 Register
REG_ADC_1_BIQUAD_A2_0	47Ch	R/W	ADC Channel 1 EQ 0 Band Biquad A2 Register
REG_ADC_1_BIQUAD_H0_1	480h	R/W	ADC Channel 1 EQ 1 Band Biquad H0 Register
REG_ADC_1_BIQUAD_B1_1	484h	R/W	ADC Channel 1 EQ 1 Band Biquad B1 Register
REG_ADC_1_BIQUAD_B2_1	488h	R/W	ADC Channel 1 EQ 1 Band Biquad B2 Register
REG_ADC_1_BIQUAD_A1_1	48Ch	R/W	ADC Channel 1 EQ 1 Band Biquad A1 Register
REG_ADC_1_BIQUAD_A2_1	490h	R/W	ADC Channel 1 EQ 1 Band Biquad A2 Register
REG_ADC_1_BIQUAD_H0_2	494h	R/W	ADC Channel 1 EQ 2 Band Biquad H0 Register
REG_ADC_1_BIQUAD_B1_2	498h	R/W	ADC Channel 1 EQ 2 Band Biquad B1 Register
REG_ADC_1_BIQUAD_B2_2	49Ch	R/W	ADC Channel 1 EQ 2 Band Biquad B2 Register
REG_ADC_1_BIQUAD_A1_2	4A0h	R/W	ADC Channel 1 EQ 2 Band Biquad A1 Register
REG_ADC_1_BIQUAD_A2_2	4A4h	R/W	ADC Channel 1 EQ 2 Band Biquad A2 Register
REG_ADC_1_BIQUAD_H0_3	4A8h	R/W	ADC Channel 1 EQ 3 Band Biquad H0 Register
REG_ADC_1_BIQUAD_B1_3	4ACh	R/W	ADC Channel 1 EQ 3 Band Biquad B1 Register
REG_ADC_1_BIQUAD_B2_3	4B0h	R/W	ADC Channel 1 EQ 3 Band Biquad B2 Register
REG_ADC_1_BIQUAD_A1_3	4B4h	R/W	ADC Channel 1 EQ 3 Band Biquad A1 Register
REG_ADC_1_BIQUAD_A2_3	4B8h	R/W	ADC Channel 1 EQ 3 Band Biquad A2 Register
REG_ADC_1_BIQUAD_H0_4	4BCh	R/W	ADC Channel 1 EQ 4 Band Biquad H0 Register
REG_ADC_1_BIQUAD_B1_4	4C0h	R/W	ADC Channel 1 EQ 4 Band Biquad B1 Register
REG_ADC_1_BIQUAD_B2_4	4C4h	R/W	ADC Channel 1 EQ 4 Band Biquad B2 Register
REG_ADC_1_BIQUAD_A1_4	4C8h	R/W	ADC Channel 1 EQ 4 Band Biquad A1 Register

All information provided in this document is subject to legal disclaimers.
© REALTEK 2024. All rights reserved.

REG_ADC_1_BIQUAD_A2_4	4CCh	R/W	ADC Channel 1 EQ 4 Band Biquad A2 Register
REG_ANA_READ	800h	R	ANA Read Register
REG_ADC_0_LPF_RD	804h	R	ADC Channel 0 LPF RD Register
REG_ADC_1_LPF_RD	808h	R	ADC Channel 1 LPF RD Register
REG_SILENCE_INFORM	824h	R	Silence Inform Register
REG_AUDIO_RO_DUMMY1	828h	R	Audio RO Dummy1 Register

29.4.4.1 REG_AUDIO_CONTROL_0

- **Name:** Audio Control Register 0
- **Size:** 32 bits
- **Address offset:** 000h
- **Read/write access:** R/W



Bit	Symbol	Access	Reset	Description
31:18	RSVD	-	0	Reserved
17:13	AUDIO_DBG_SEL	R/W	0	Debug probe selection
12:6	RSVD	-	0	Reserved
5:4	SYS_CLK_RATE_SEL	R/W	0x2	Audio sys_clk selection
3	AUDIO_CONTROL_0_DUMMY	R/W	0	DUMMY
2:1	RSVD	-	0	Reserved
0	AUDIO_IP_EN	R/W	0	

29.4.4.2 REG_AUDIO_CONTROL_1

- **Name:** Audio Control Register 1
- **Size:** 32 bits
- **Address offset:** 004h
- **Read/write access:** R/W

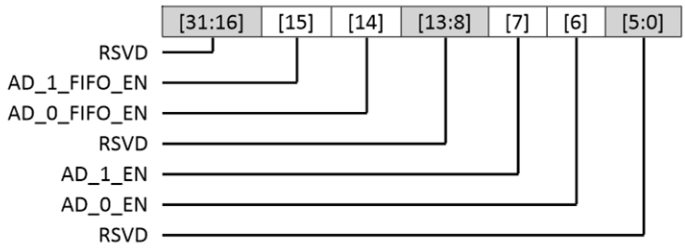


Bit	Symbol	Access	Reset	Description
31:17	RSVD	-	0	Reserved
16:13	AUDIO_CONTROL_1_DUMMY	R/W	0	Dummy
12	I2S_DATA_RND_EN	R/W	1	i2s_data_rnd_en
11:0	RSVD	-	0	Reserved

29.4.4.3 REG_CLOCK_CONTROL_1

- **Name:** Clock Control Register 1
- **Size:** 32 bits
- **Address offset:** 008h
- **Read/write access:** R/W

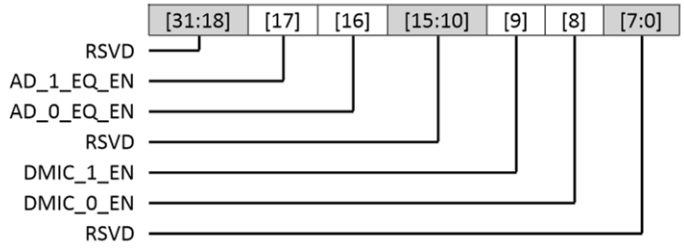
All information provided in this document is subject to legal disclaimers.
© REALTEK 2024. All rights reserved.



Bit	Symbol	Access	Reset	Description
31:16	RSVD	-	0	Reserved
15	AD_1_FIFO_EN	R/W	0	ADC channel 1 FIFO clock enable
14	AD_0_FIFO_EN	R/W	0	ADC channel 0 FIFO clock enable
13:8	RSVD	-	0	Reserved
7	AD_1_EN	R/W	0	ADC channel 1 clock enable
6	AD_0_EN	R/W	0	ADC channel 0 clock enable
5:0	RSVD	-	0	Reserved

29.4.4.4 REG_CLOCK_CONTROL_2

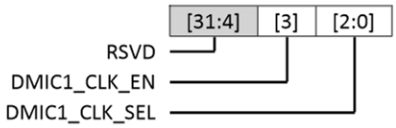
- **Name:** Clock Control Register 2
- **Size:** 32 bits
- **Address offset:** 00Ch
- **Read/write access:** R/W



Bit	Symbol	Access	Reset	Description
31:18	RSVD	-	0	Reserved
17	AD_1_EQ_EN	R/W	0	ADC channel 1 EQ clock enable
16	AD_0_EQ_EN	R/W	0	ADC channel 0 EQ clock enable
15:10	RSVD	-	0	Reserved
9	DMIC_1_EN	R/W	0	ADC filter channel 1 clock enable: DMIC path
8	DMIC_0_EN	R/W	0	ADC filter channel 0 clock enable: DMIC path
7:0	RSVD	-	0	Reserved

29.4.4.5 REG_CLOCK_CONTROL_3

- **Name:** Clock Control Register 3
- **Size:** 32 bits
- **Address offset:** 010h
- **Read/write access:** R/W



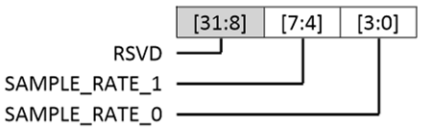
Bit	Symbol	Access	Reset	Description
31:4	RSVD	-	0	Reserved

All information provided in this document is subject to legal disclaimers.
© REALTEK 2024. All rights reserved.

3	DMIC1_CLK_EN	R/W	0	Digital microphone clock enable
2:0	DMIC1_CLK_SEL	R/W	1	Set clock of digital microphone 3'b000: 5MHz 3'b001: 2.5MHz 3'b010: 1.25MHz 3'b011: 625kHz 3'b100: 312.5kHz 3'b101: Reserved 3'b110: Reserved 3'b111: 769.2kHz

29.4.4.6 REG_CLOCK_CONTROL_4

- **Name:** Clock Control Register 4
- **Size:** 32 bits
- **Address offset:** 014h
- **Read/write access:** R/W



Bit	Symbol	Access	Reset	Description
31:8	RSVD	-	0	Reserved
7:4	SAMPLE_RATE_1	R/W	0	Set sample rate source 1 4'h0: 48K 4'h1: 96K 4'h2: 192K 4'h3: 32K 4'h4: 176.4K 4'h5: 16K 4'h6: Reserved 4'h7: 8K 4'h8: 44.1K 4'h9: 88.2K 4'ha:24K 4'hb: 12K 4'hc: 22.05K 4'hd: 11.025K 4'he~4'hf: Reserved
3:0	SAMPLE_RATE_0	R/W	0	Set sample rate source 0 4'h0: 48K 4'h1: 96K 4'h2: 192K 4'h3: 32K 4'h4: 176.4K 4'h5: 16K 4'h6: Reserved 4'h7: 8K 4'h8: 44.1K 4'h9: 88.2K 4'ha:24K 4'hb: 12K 4'hc: 22.05K 4'hd: 11.025K 4'he~4'hf: Reserved

All information provided in this document is subject to legal disclaimers.
© REALTEK 2024. All rights reserved.

29.4.4.7 REG_CLOCK_CONTROL_5

- **Name:** Clock Control Register 5
- **Size:** 32 bits
- **Address offset:** 018h
- **Read/write access:** R/W



Bit	Symbol	Access	Reset	Description
31:4	RSVD	-	0	Reserved
3:2	ADC_1_FS_SRC_SEL	R/W	0	Channel 1 ADC path sample rate source selection 2'b00: Source 0 2'b01: Source 1 2'b10: Source 2
1:0	ADC_0_FS_SRC_SEL	R/W	0	Channel 0 ADC path sample rate source selection 2'b00: Source 0 2'b01: Source 1 2'b10: Source 2

29.4.4.8 REG_CLOCK_CONTROL_6

- **Name:** Clock Control Register 6
- **Size:** 32 bits
- **Address offset:** 01Ch
- **Read/write access:** R/W



Bit	Symbol	Access	Reset	Description
31:2	RSVD	-	0	Reserved
1	ADC_1_ASRC_EN	R/W	0	Channel 1 ADC path ASRC enable 0: Disable 1: Enable If ASRC is enabled, sample_rate becomes useless.
0	ADC_0_ASRC_EN	R/W	0	Channel 0 ADC path ASRC enable 0: Disable 1: Enable If ASRC is enabled, sample_rate becomes useless.

29.4.4.9 REG_CLOCK_CONTROL_7

- **Name:** Clock Control Register 7
- **Size:** 32 bits
- **Address offset:** 020h
- **Read/write access:** R/W

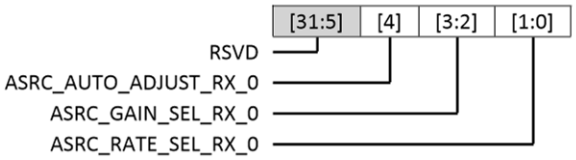


All information provided in this document is subject to legal disclaimers.
© REALTEK 2024. All rights reserved.

Bit	Symbol	Access	Reset	Description
31:11	RSVD	-	0	Reserved
10:9	ADC_1_DMIC_LPF_CLK_SEL	R/W	0	Channel 1 ADC path DMIC LPF clock 2'b00: 10M 2'b01: 5M 2'b10: 2.5M 2'b11: 769.2*2K
8:7	ADC_0_DMIC_LPF_CLK_SEL	R/W	0	Channel 0 ADC path DMIC LPF clock 2'b00: 10M 2'b01: 5M 2'b10: 2.5M 2'b11: 769.2*2K
6:0	RSVD	-	0	Reserved

29.4.4.10 REG_ASRC_CONTROL_2

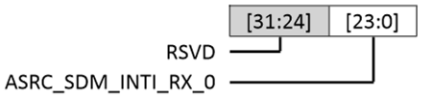
- **Name:** ASRC Control Register 2
- **Size:** 32 bits
- **Address offset:** 02Ch
- **Read/write access:** R/W



Bit	Symbol	Access	Reset	Description
31:5	RSVD	-	0	Reserved
4	ASRC_AUTO_ADJUST_RX_0	R/W	1	HW auto adjust convergence rate 0: Disable 1: Enable
3:2	ASRC_GAIN_SEL_RX_0	R/W	0x3	ASRC convergence rate: larger is faster but more noisy
1:0	ASRC_RATE_SEL_RX_0	R/W	0	0: fs supports 4kHz ~ 60kHz 1: fs supports 60kHz ~ 120kHz 2/3: Reserved

29.4.4.11 REG_ASRC_CONTROL_3

- **Name:** ASRC Control Register 3
- **Size:** 32 bits
- **Address offset:** 030h
- **Read/write access:** R/W

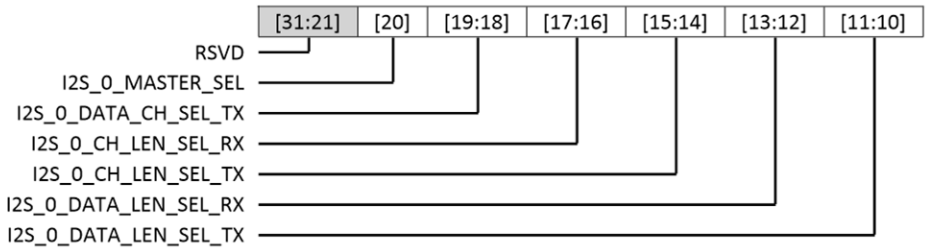


Bit	Symbol	Access	Reset	Description
31:24	RSVD	-	0	Reserved
23:0	ASRC_SDM_INTI_RX_0	R/W	0	Set initial value of tracked frequency

29.4.4.12 REG_I2S_0_CONTROL

- **Name:** I2S 0 Control Register
- **Size:** 32 bits
- **Address offset:** 03Ch
- **Read/write access:** R/W

All information provided in this document is subject to legal disclaimers.
© REALTEK 2024. All rights reserved.





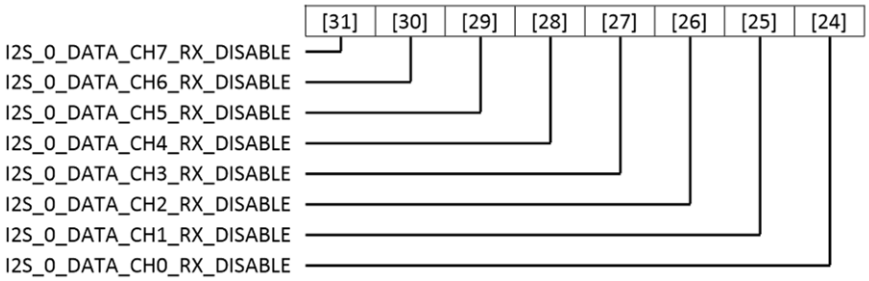
Bit	Symbol	Access	Reset	Description
31:21	RSVD	-	0	Reserved
20	I2S_0_MASTER_SEL	R/W	0	I2S 0 master source selection 1'b0: internal SPORT 1'b1: external I2S
19:18	I2S_0_DATA_CH_SEL_TX	R/W	0	I2S 0 Tx channel data 2'b00: L/R 2'b01: R/L 2'b10: L/L 2'b11: R/R
17:16	I2S_0_CH_LEN_SEL_RX	R/W	0	I2S 0 Rx channel length 2'b00: 16 bits 2'b01: 32 bits 2'b10: 24 bits 2'b11: 8 bits
15:14	I2S_0_CH_LEN_SEL_TX	R/W	0	I2S 0 Tx channel length 2'b00: 16 bits 2'b01: 32 bits 2'b10: 24 bits 2'b11: 8 bits
13:12	I2S_0_DATA_LEN_SEL_RX	R/W	0	I2S 0 Rx channel data length 2'b00: 16 bits 2'b10: 24 bits 2'b11: 8 bits
11:10	I2S_0_DATA_LEN_SEL_TX	R/W	0	I2S 0 Tx channel data length 2'b00: 16 bits 2'b10: 24 bits 2'b11: 8 bits
9:8	I2S_0_DATA_FORMAT_SEL_RX	R/W	0	I2S 0 Rx channel data format 2'b00: I2S 2'b01: Left-justified 2'b10: PCM mode A 2'b11: PCM mode B
7:6	I2S_0_DATA_FORMAT_SEL_TX	R/W	0	I2S 0 Tx channel data format 2'b00: I2S 2'b01: Left-justified 2'b10: PCM mode A 2'b11: PCM mode B
5:4	I2S_0_TDM_MODE_RX	R/W	0	I2S 0 Rx channel TDM mode 0: 2 channels 1: 4 channels

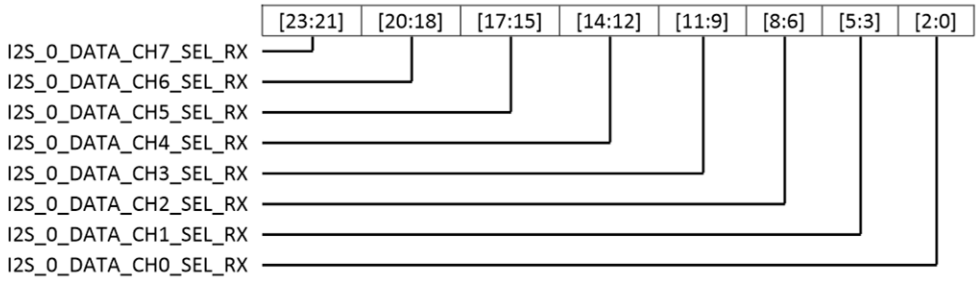
All information provided in this document is subject to legal disclaimers.
© REALTEK 2024. All rights reserved.

				2/3: Reserved
3	I2S_0_SAME_LRC_EN	R/W	0	1'b1: LRC_TX share to LRC_RX
2	I2S_0_SELF_LPBK_EN	R/W	0	1'b1: internal loopback mode is enabled
1	I2S_0_INV_SCLK	R/W	0	1'b1: I2S/PCM bit clock is inverted
0	I2S_0_RST_N_REG	R/W	0	1'b1: rst_n to audio digital IP is de-asserted 1'b0: rst_n to audio digital IP is asserted

29.4.4.13 REG_I2S_0_CONTROL_1

- **Name:** I2S 0 Control Register 1
- **Size:** 32 bits
- **Address offset:** 040h
- **Read/write access:** R/W





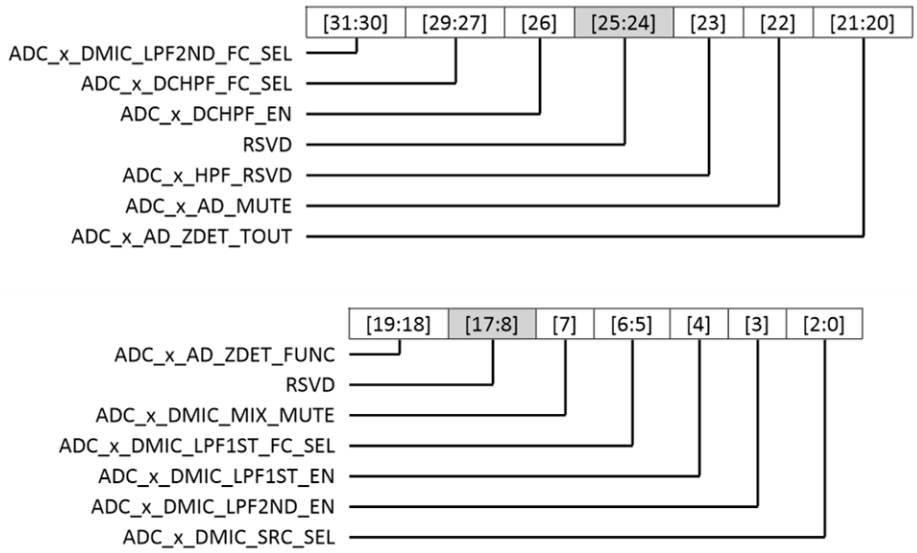
Bit	Symbol	Access	Reset	Description
31	I2S_0_DATA_CH7_RX_DISABLE	R/W	0	I2S 0 Rx channel data channel 7 1: disable 0: enable
30	I2S_0_DATA_CH6_RX_DISABLE	R/W	0	I2S 0 Rx channel data channel 6 1: disable 0: enable
29	I2S_0_DATA_CH5_RX_DISABLE	R/W	0	I2S 0 Rx channel data channel 5 1: disable 0: enable
28	I2S_0_DATA_CH4_RX_DISABLE	R/W	0	I2S 0 Rx channel data channel 4 1: disable 0: enable
27	I2S_0_DATA_CH3_RX_DISABLE	R/W	0	I2S 0 Rx channel data channel 3 1: disable 0: enable
26	I2S_0_DATA_CH2_RX_DISABLE	R/W	0	I2S 0 Rx channel data channel 2 1: disable 0: enable
25	I2S_0_DATA_CH1_RX_DISABLE	R/W	0	I2S 0 Rx channel data channel 1 1: disable 0: enable
24	I2S_0_DATA_CH0_RX_DISABLE	R/W	0	I2S 0 Rx channel data channel 0 1: disable 0: enable

All information provided in this document is subject to legal disclaimers.
© REALTEK 2024. All rights reserved.

23:21	I2S_0_DATA_CH7_SEL_RX	R/W	0	I2S 0 Rx channel data channel 7 0: CH0 1: CH1 2/3/4/5/6/7: reserved
20:18	I2S_0_DATA_CH6_SEL_RX	R/W	0	I2S 0 Rx channel data channel 6 0: CH0 1: CH1 2/3/4/5/6/7: reserved
17:15	I2S_0_DATA_CH5_SEL_RX	R/W	5	I2S 0 Rx channel data channel 5 0: CH0 1: CH1 2/3/4/5/6/7: reserved
14:12	I2S_0_DATA_CH4_SEL_RX	R/W	4	I2S 0 Rx channel data channel 4 0: CH0 1: CH1 2/3/4/5/6/7: reserved
11:9	I2S_0_DATA_CH3_SEL_RX	R/W	3	I2S 0 Rx channel data channel 3 0: CH0 1: CH1 2/3/4/5/6/7: reserved
8:6	I2S_0_DATA_CH2_SEL_RX	R/W	2	I2S 0 Rx channel data channel 2 0: CH0 1: CH1 2/3/4/5/6/7: reserved
5:3	I2S_0_DATA_CH1_SEL_RX	R/W	1	I2S 0 Rx channel data channel 1 0: CH0 1: CH1 2/3/4/5/6/7: reserved
2:0	I2S_0_DATA_CH0_SEL_RX	R/W	0	I2S 0 Rx channel data channel 0 0: CH0 1: CH1 2/3/4/5/6/7: reserved

29.4.4.14 REG_ADC_x_CONTROL_0

- **Name:** ADC Channel x Control Register 0 (x={0, 1})
- **Size:** 32 bits
- **Address offset:**
  - **REG_ADC_0_CONTROL_0:** 050h
  - **REG_ADC_1_CONTROL_0:** 058h
- **Read/write access:** R/W

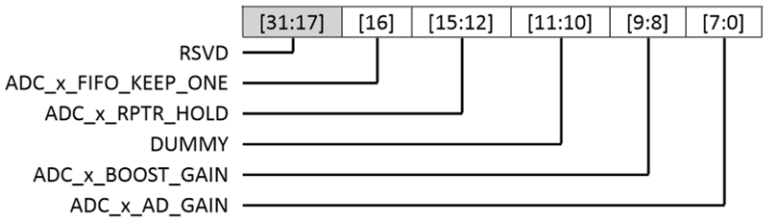


All information provided in this document is subject to legal disclaimers.
© REALTEK 2024. All rights reserved.

Bit	Symbol	Access	Reset	Description
31:30	ADC_x_DMIC_LPF2ND_FC_SEL	R/W	0	Uplink Channel x DMIC path SRC 2nd LPF FC
29:27	ADC_x_DCHPF_FC_SEL	R/W	0x4	Channel x ADC path high pass filter Fc
26	ADC_x_DCHPF_EN	R/W	0	Channel x ADC path high pass filter enable control (filter DC) 0: Disable 1: Enable
25:24	RSVD	-	0	Reserved
23	ADC_x_HPF_RSVD	R/W	0	Channel x ADC path reserved
22	ADC_x_AD_MUTE	R/W	0	Channel x ADC path mute 0: Un-Mute 1: Mute
21:20	ADC_x_AD_ZDET_TOUT	R/W	0	Channel x ADC path zero detection time out selection 2'b00: 1024*16 samples 2'b01: 1024*32 samples 2'b10: 1024*64 samples 2'b11: 64 samples
19:18	ADC_x_AD_ZDET_FUNC	R/W	0x2	Channel x ADC path zero detection function selection 2'b00: immediate change 2'b01: zero detection & immediate change 2'b10: zero detection & step 2'b11: zero detection & timeout step
17:8	RSVD	-	0	Reserved
7	ADC_x_DMIC_MIX_MUTE	R/W	1	Channel x DMIC input path mute 0: Un-Mute 1: Mute
6:5	ADC_x_DMIC_LPF1ST_FC_SEL	R/W	0	Channel x DMIC path SRC 1st LPF FC
4	ADC_x_DMIC_LPF1ST_EN	R/W	1	Channel x DMIC path SRC 1st LPF control 0: Disable 1: Enable
3	ADC_x_DMIC_LPF2ND_EN	R/W	1	Channel x DMIC path SRC 2nd LPF control 0: Disable 1: Enable
2:0	ADC_x_DMIC_SRC_SEL	R/W	0	Channel x DMIC source selection 3'b000: MIC 1 rising 3'b001: MIC 1 falling 3'b010: MIC 2 rising 3'b011: MIC 2 falling other: reserved

29.4.4.15 REG_ADC_x_CONTROL_1

- **Name:** ADC Channel x Control Register 1 (x={0, 1})
- **Size:** 32 bits
- **Address offset:**
  - **REG_ADC_0_CONTROL_1:** 054h
  - **REG_ADC_1_CONTROL_1:** 05Ch
- **Read/write access:** R/W



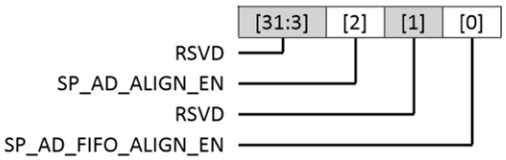
Bit	Symbol	Access	Reset	Description
31:17	RSVD	-	0	Reserved
16	ADC_x_FIFO_KEEP_ONE	R/W	0	Channel x I2S sample buffering 0: 4 samples

All information provided in this document is subject to legal disclaimers.
© REALTEK 2024. All rights reserved.

				1: 1 sample
15:12	ADC_x_RPTR_HOLD	R/W	0	Channel x I2S read point hold number rptr_hold = (I2S_fs/ad_fs) - 1
11:10	DUMMY	R/W	0	Dummy
9:8	ADC_x_BOOST_GAIN	R/W	0	Channel x ADC path boost gain control 00: 0dB 01: 12dB 10: 24dB 11: 36dB
7:0	ADC_x_AD_GAIN	R/W	0x2f	Channel x ADC digital volume -17.625dB ~ 48dB in 0.375dB step 8'h00: -17.625dB ... 8'h2f: 0dB 8'h30: 0.375dB ... 8'haf: 48dB

29.4.4.16 REG_ADC_ALIGN_CONTROL_REG_0

- **Name:** ADC Align Control Register 0
- **Size:** 32 bits
- **Address offset:** 0ACh
- **Read/write access:** R/W



Bit	Symbol	Access	Reset	Description
31:3	RSVD	-	0	Reserved
2	SP_AD_ALIGN_EN	R/W	0	1: Uplink channel will align with I2S 0: Disable
1	RSVD	-	0	Reserved
0	SP_AD_FIFO_ALIGN_EN	R/W	0	1: Uplink FIFO will align with I2S 0: Disable

29.4.4.17 REG_ADC_ALIGN_CONTROL_REG

- **Name:** ADC Align Control Register
- **Size:** 32 bits
- **Address offset:** 0B0h
- **Read/write access:** R/W



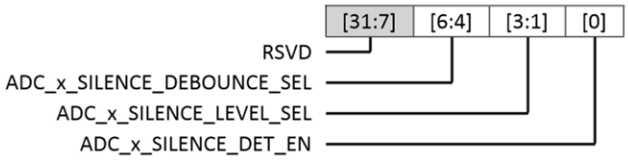
Bit	Symbol	Access	Reset	Description
31:8	RSVD	-	0	Reserved
7	ADC_1_ALIGN_EN	R/W	0	Uplink CH1 align enable
6:4	ADC_1_ALIGN_CH_SEL	R/W	0	Uplink CH1 align channel selection 0: CH0 1: CH1 Others: reserved

All information provided in this document is subject to legal disclaimers.
© REALTEK 2024. All rights reserved.

3	ADC_0_ALIGN_EN	R/W	0	Uplink CH0 align enable
2:0	ADC_0_ALIGN_CH_SEL	R/W	0	Uplink CH0 align channel selection 0: CH0 1: CH1 Others: reserved

29.4.4.18 REG_ADC_x_SILENCE_CONTROL

- **Name:** ADC Channel x Silence Control Register (x={0, 1})
- **Size:** 32 bits
- **Address offset:**
  - **REG_ADC_0_SILENCE_CONTROL:** 100h
  - **REG_ADC_1_SILENCE_CONTROL:** 104h
- **Read/write access:** R/W



Bit	Symbol	Access	Reset	Description
31:7	RSVD	-	0	Reserved
6:4	ADC_x_SILENCE_DEBOUNCE_SEL	R/W	0x3	Channel x ADC path silence detection debounce (48K) 3'b000: 80ms 3'b001: 160ms 3'b010: 320ms 3'b011: 640ms 3'b100: 1.28s 3'b101: 2.56s 3'b110: 5.12s 3'b111: 0.16ms
3:1	ADC_x_SILENCE_LEVEL_SEL	R/W	1	Channel x ADC path silence detection threshold 3'b000: -54db 3'b001: -60db 3'b010: -66db 3'b011: -72db 3'b100: -78db 3'b101: -84db 3'b110: -90db 3'b111: -96db
0	ADC_x_SILENCE_DET_EN	R/W	0	Channel x ADC path silence detection enable 0: Disable 1: Enable

29.4.4.19 REG_ADC_0_EQ_CTRL

- **Name:** ADC Channel 0 EQ Control Register
- **Size:** 32 bits
- **Address offset:** 400h
- **Read/write access:** R/W

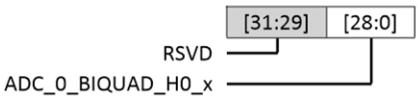


All information provided in this document is subject to legal disclaimers.
© REALTEK 2024. All rights reserved.

Bit	Symbol	Access	Reset	Description
31:5	RSVD	-	0	Reserved
4	ADC_0_BIQUAD_EN_4	R/W	0	ADC channel 0 EQ 4-band biquad enable 0: disable 1: enable
3	ADC_0_BIQUAD_EN_3	R/W	0	ADC channel 0 EQ 3-band biquad enable 0: disable 1: enable
2	ADC_0_BIQUAD_EN_2	R/W	0	ADC channel 0 EQ 2-band biquad enable 0: disable 1: enable
1	ADC_0_BIQUAD_EN_1	R/W	0	ADC channel 0 EQ 1-band biquad enable 0: disable 1: enable
0	ADC_0_BIQUAD_EN_0	R/W	0	ADC channel 0 EQ 0-band biquad enable 0: disable 1: enable

29.4.4.20 REG_ADC_0_BIQUAD_H0_x

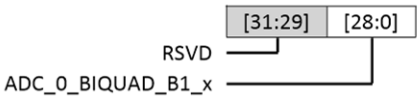
- **Name:** ADC Channel 0 EQ x Band Biquad H0 Register (x={0, 1, 2, 3, 4})
- **Size:** 32 bits
- **Address offset:**
  - **REG_ADC_0_BIQUAD_H0_0:** 404h
  - **REG_ADC_0_BIQUAD_H0_1:** 418h
  - **REG_ADC_0_BIQUAD_H0_2:** 42Ch
  - **REG_ADC_0_BIQUAD_H0_3:** 440h
  - **REG_ADC_0_BIQUAD_H0_4:** 454h
- **Read/write access:** R/W



Bit	Symbol	Access	Reset	Description
31:29	RSVD	-	0	Reserved
28:0	ADC_0_BIQUAD_H0_x	R/W	0x2000000	ADC channel 0 EQ x-band coef. H0 2's complement in 4.25 format, i.e. the range is from -8~7.99.

29.4.4.21 REG_ADC_0_BIQUAD_B1_x

- **Name:** ADC Channel 0 EQ x Band Biquad B1 Register (x={0, 1, 2, 3, 4})
- **Size:** 32 bits
- **Address offset:**
  - **REG_ADC_0_BIQUAD_B1_0:** 408h
  - **REG_ADC_0_BIQUAD_B1_1:** 41Ch
  - **REG_ADC_0_BIQUAD_B1_2:** 430h
  - **REG_ADC_0_BIQUAD_B1_3:** 444h
  - **REG_ADC_0_BIQUAD_B1_4:** 458h
- **Read/write access:** R/W

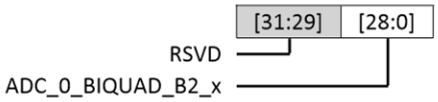


Bit	Symbol	Access	Reset	Description
31:29	RSVD	-	0	Reserved
28:0	ADC_0_BIQUAD_B1_x	R/W	0	ADC channel 0 EQ x-band coef. B1 2's complement in 4.25 format, i.e. the range is from -8~7.99.

All information provided in this document is subject to legal disclaimers.
© REALTEK 2024. All rights reserved.

29.4.4.22 REG_ADC_0_BIQUAD_B2_x

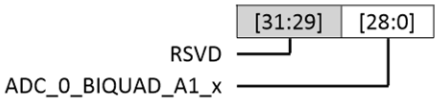
- **Name:** ADC Channel 0 EQ x Band Biquad B2 Register (x={0, 1, 2, 3, 4})
- **Size:** 32 bits
- **Address offset:**
  - **REG_ADC_0_BIQUAD_B2_0:** 40Ch
  - **REG_ADC_0_BIQUAD_B2_1:** 420h
  - **REG_ADC_0_BIQUAD_B2_2:** 434h
  - **REG_ADC_0_BIQUAD_B2_3:** 448h
  - **REG_ADC_0_BIQUAD_B2_4:** 45Ch
- **Read/write access:** R/W



Bit	Symbol	Access	Reset	Description
31:29	RSVD	-	0	Reserved
28:0	ADC_0_BIQUAD_B2_x	R/W	0	ADC channel 0 EQ x-band coef. B2 2's complement in 4.25 format, i.e. the range is from -8~7.99.

29.4.4.23 REG_ADC_0_BIQUAD_A1_x

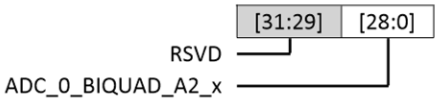
- **Name:** ADC Channel 0 EQ x Band Biquad A1 Register (x={0, 1, 2, 3, 4})
- **Size:** 32 bits
- **Address offset:**
  - **REG_ADC_0_BIQUAD_A1_0:** 410h
  - **REG_ADC_0_BIQUAD_A1_1:** 424h
  - **REG_ADC_0_BIQUAD_A1_2:** 438h
  - **REG_ADC_0_BIQUAD_A1_3:** 44Ch
  - **REG_ADC_0_BIQUAD_A1_4:** 460h
- **Read/write access:** R/W



Bit	Symbol	Access	Reset	Description
31:29	RSVD	-	0	Reserved
28:0	ADC_0_BIQUAD_A1_x	R/W	0	ADC channel 0 EQ x-band coef. A1 2's complement in 4.25 format, i.e. the range is from -8~7.99.

29.4.4.24 REG_ADC_0_BIQUAD_A2_x

- **Name:** ADC Channel 0 EQ x Band Biquad A2 Register (x={0, 1, 2, 3, 4})
- **Size:** 32 bits
- **Address offset:**
  - **REG_ADC_0_BIQUAD_A2_0:** 414h
  - **REG_ADC_0_BIQUAD_A2_1:** 428h
  - **REG_ADC_0_BIQUAD_A2_2:** 43Ch
  - **REG_ADC_0_BIQUAD_A2_3:** 450h
  - **REG_ADC_0_BIQUAD_A2_4:** 464h
- **Read/write access:** R/W



All information provided in this document is subject to legal disclaimers.
© REALTEK 2024. All rights reserved.

Bit	Symbol	Access	Reset	Description
31:29	RSVD	-	0	Reserved
28:0	ADC_0_BIQUAD_A2_x	R/W	0	ADC channel 0 EQ x-band coef. A2 2's complement in 4.25 format, i.e. the range is from -8~7.99.

29.4.4.25 REG_ADC_1_EQ_CTRL

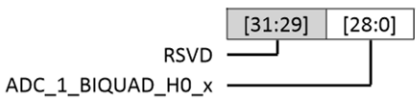
- **Name:** ADC Channel 1 EQ Control Register
- **Size:** 32 bits
- **Address offset:** 468h
- **Read/write access:** R/W



Bit	Symbol	Access	Reset	Description
31:5	RSVD	-	0	Reserved
4	ADC_1_BIQUAD_EN_4	R/W	0	ADC channel 1 EQ 4-band biquad enable 0: disable 1: enable
3	ADC_1_BIQUAD_EN_3	R/W	0	ADC channel 1 EQ 3-band biquad enable 0: disable 1: enable
2	ADC_1_BIQUAD_EN_2	R/W	0	ADC channel 1 EQ 2-band biquad enable 0: disable 1: enable
1	ADC_1_BIQUAD_EN_1	R/W	0	ADC channel 1 EQ 1-band biquad enable 0: disable 1: enable
0	ADC_1_BIQUAD_EN_0	R/W	0	ADC channel 1 EQ 0-band biquad enable 0: disable 1: enable

29.4.4.26 REG_ADC_1_BIQUAD_H0_x

- **Name:** ADC Channel 1 EQ x Band Biquad H0 Register (x={0, 1, 2, 3, 4})
- **Size:** 32 bits
- **Address offset:**
  - **REG_ADC_1_BIQUAD_H0_0:** 46Ch
  - **REG_ADC_1_BIQUAD_H0_1:** 480h
  - **REG_ADC_1_BIQUAD_H0_2:** 494h
  - **REG_ADC_1_BIQUAD_H0_3:** 4A8h
  - **REG_ADC_1_BIQUAD_H0_4:** 4BCh
- **Read/write access:** R/W

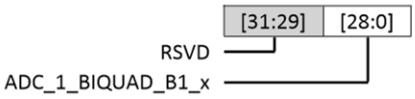


Bit	Symbol	Access	Reset	Description
31:29	RSVD	-	0	Reserved
28:0	ADC_1_BIQUAD_H0_x	R/W	0x2000000	ADC channel 1 EQ x-band coef. h0 2's complement in 4.25 format, i.e. the range is from -8~7.99.

All information provided in this document is subject to legal disclaimers.
© REALTEK 2024. All rights reserved.

29.4.4.27 REG_ADC_1_BIQUAD_B1_x

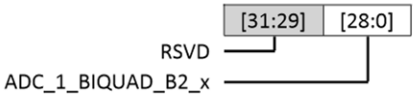
- **Name:** ADC Channel 1 EQ x Band Biquad B1 Register (x={0, 1, 2, 3, 4})
- **Size:** 32 bits
- **Address offset:**
  - **REG_ADC_1_BIQUAD_B1_0:** 470h
  - **REG_ADC_1_BIQUAD_B1_1:** 484h
  - **REG_ADC_1_BIQUAD_B1_2:** 498h
  - **REG_ADC_1_BIQUAD_B1_3:** 4ACh
  - **REG_ADC_1_BIQUAD_B1_4:** 4C0h
- **Read/write access:** R/W



Bit	Symbol	Access	Reset	Description
31:29	RSVD	-	0	Reserved
28:0	ADC_1_BIQUAD_B1_x	R/W	0	ADC channel 1 EQ x-band coef. b1 2's complement in 4.25 format, i.e. the range is from -8~7.99.

29.4.4.28 REG_ADC_1_BIQUAD_B2_x

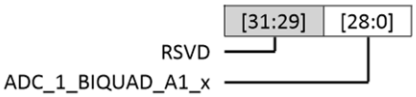
- **Name:** ADC Channel 1 EQ x Band Biquad B2 Register (x={0, 1, 2, 3, 4})
- **Size:** 32 bits
- **Address offset:**
  - **REG_ADC_1_BIQUAD_B2_0:** 474h
  - **REG_ADC_1_BIQUAD_B2_1:** 488h
  - **REG_ADC_1_BIQUAD_B2_2:** 49Ch
  - **REG_ADC_1_BIQUAD_B2_3:** 4B0h
  - **REG_ADC_1_BIQUAD_B2_4:** 4C4h
- **Read/write access:** R/W



Bit	Symbol	Access	Reset	Description
31:29	RSVD	-	0	Reserved
28:0	ADC_1_BIQUAD_B2_x	R/W	0	ADC channel 1 EQ x-band coef. b2 2's complement in 4.25 format, i.e. the range is from -8~7.99.

29.4.4.29 REG_ADC_1_BIQUAD_A1_x

- **Name:** ADC Channel 1 EQ x Band Biquad A1 Register (x={0, 1, 2, 3, 4})
- **Size:** 32 bits
- **Address offset:**
  - **REG_ADC_1_BIQUAD_A1_0:** 478h
  - **REG_ADC_1_BIQUAD_A1_1:** 48Ch
  - **REG_ADC_1_BIQUAD_A1_2:** 4A0h
  - **REG_ADC_1_BIQUAD_A1_3:** 4B4h
  - **REG_ADC_1_BIQUAD_A1_4:** 4C8h
- **Read/write access:** R/W

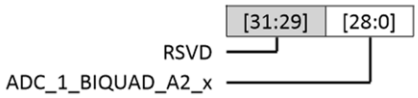


All information provided in this document is subject to legal disclaimers.
© REALTEK 2024. All rights reserved.

Bit	Symbol	Access	Reset	Description
31:29	RSVD	-	0	Reserved
28:0	ADC_1_BIQUAD_A1_x	R/W	0	ADC channel 1 EQ x-band coef. a1 2's complement in 4.25 format, i.e. the range is from -8~7.99.

29.4.4.30 REG_ADC_1_BIQUAD_A2_x

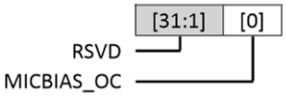
- **Name:** ADC Channel 1 EQ x Band Biquad A2 Register (x={0, 1, 2, 3, 4})
- **Size:** 32 bits
- **Address offset:**
  - **REG_ADC_1_BIQUAD_A2_0:** 47Ch
  - **REG_ADC_1_BIQUAD_A2_1:** 490h
  - **REG_ADC_1_BIQUAD_A2_2:** 4A4h
  - **REG_ADC_1_BIQUAD_A2_3:** 4B8h
  - **REG_ADC_1_BIQUAD_A2_4:** 4CCh
- **Read/write access:** R/W



Bit	Symbol	Access	Reset	Description
31:29	RSVD	-	0	Reserved
28:0	ADC_1_BIQUAD_A2_x	R/W	0	ADC channel 1 EQ x-band coef. a2 2's complement in 4.25 format, i.e. the range is from -8~7.99.

29.4.4.31 REG_ANA_READ

- **Name:** ANA Read Register
- **Size:** 32 bits
- **Address offset:** 800h
- **Read/write access:** R



Bit	Symbol	Access	Reset	Description
31:1	RSVD	-	0	Reserved
0	MICBIAS_OC	R		The status flag of MICBIAS over-current protection

29.4.4.32 REG_ADC_0_LPF_RD

- **Name:** ADC Channel 0 LPF RD Register
- **Size:** 32 bits
- **Address offset:** 804h
- **Read/write access:** R

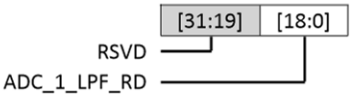


Bit	Symbol	Access	Reset	Description
31:19	RSVD	-	0	Reserved
18:0	ADC_0_LPF_RD	R		CH 0 ADC LPF out values

All information provided in this document is subject to legal disclaimers.
© REALTEK 2024. All rights reserved.

29.4.4.33 REG_ADC_1_LPF_RD

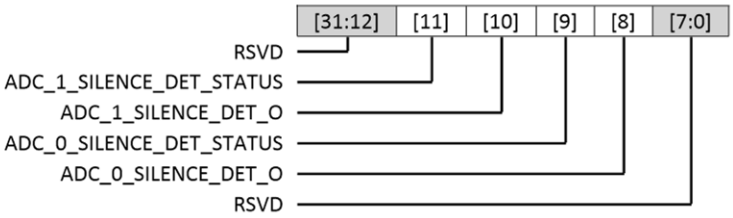
- **Name:** ADC Channel 1 LPF RD Register
- **Size:** 32 bits
- **Address offset:** 808h
- **Read/write access:** R



Bit	Symbol	Access	Reset	Description
31:19	RSVD	-	0	Reserved
18:0	ADC_1_LPF_RD	R		CH 1 ADC LPF out values

29.4.4.34 REG_SILENCE_INFORM

- **Name:** Silence Inform Register
- **Size:** 32 bits
- **Address offset:** 824h
- **Read/write access:** R



Bit	Symbol	Access	Reset	Description
31:12	RSVD	-	0	Reserved
11	ADC_1_SILENCE_DET_STATUS	R		Ongoing status of adc_1_silencedetection 1'b0: adc_1_silencedetection is resting (clock is gating) 1'b1: adc_1_silencedetection is working
10	ADC_1_SILENCE_DET_O	R		adc_1_silencedata status (result of silence detection) 1'b0: not adc_1_silencedata 1'b1: adc_1_silenceis detected
9	ADC_0_SILENCE_DET_STATUS	R		Ongoing status of adc_0_silencedetection 1'b0: adc_0_silencedetection is resting (clock is gating) 1'b1: adc_0_silencedetection is working
8	ADC_0_SILENCE_DET_O	R		adc_0_silencedata status (result of silence detection) 1'b0: not adc_0_silencedata 1'b1: adc_0_silenceis detected
7:0	RSVD	-	0	Reserved

29.4.4.35 REG_AUDIO_RO_DUMMY1

- **Name:** Audio RO Dummy1 Register
- **Size:** 32 bits
- **Address offset:** 828h
- **Read/write access:** R



Bit	Symbol	Access	Reset	Description
31:0	AUDIO_RO_DUMMY1	R		Dummy register

All information provided in this document is subject to legal disclaimers.
© REALTEK 2024. All rights reserved.

Revision History

Date	Revision	Release Notes
2024-03-18	1.0	Initial release

All information provided in this document is subject to legal disclaimers.
© REALTEK 2024. All rights reserved.