ESP32

技术参考手册

版本 4.5

版 © 2021

关于本手册

《ESP32 参 册》 标 ESP32 。本 册 ESP32 ，包 个 部 、 。

、 可 《ESP32 》 。

文档版本

https://www.espressif.com/zh-hans/support/download/documents 版本 。

修订历史

查 。

文档变更通知

可 www.espressif.com/zh-hans/subscribe 变 。

证书下载

可 www.espressif.com/zh-hans/certificates 产 。

目录

1 系统和存储器 24
1.1 24
1.2 24
1.3 26
1.3.1 26
1.3.2 26
1.3.2.1 Internal ROM 0 27
1.3.2.2 Internal ROM 1 27
1.3.2.3 Internal SRAM 0 27
1.3.2.4 Internal SRAM 1 28
1.3.2.5 Internal SRAM 2 28
1.3.2.6 DMA 28
1.3.2.7 RTC FAST Memory 29
1.3.2.8 RTC SLOW Memory 29
1.3.3 29
1.3.4 Cache 29
1.3.5 30
1.3.5.1 不 PID Controller 32
1.3.5.2 不 32
1.3.5.3 32
2 中断矩阵 (INTERRUPT) 33
2.1 33
2.2 33
2.3 33
2.3.1 部 33
2.3.2 CPU 36
2.3.3 部 CPU 部 36
2.3.4 屏蔽 CPU NMI 37
2.3.5 查 部 37
3 复位和时钟 38
3.1 System 38
3.1.1 38
3.1.2 38
3.2 39
3.2.1 39
3.2.2 39
3.2.3 CPU 40
3.2.4 41
3.2.4.1 APB_CLK 41
3.2.4.2 REF_TICK 42

5.5 99

6 DMA 控制器 (DMA) 113
6.1 113
6.2 113
6.3 113
6.3.1 DMA 113
6.3.2 表 114
6.4 UART DMA (UDMA) 114
6.5 SPI DMA 115
6.6 I2S DMA 116

7 SPI 控制器 (SPI) 117
7.1 117
7.2 SPI 117
7.3 GP-SPI 118
7.3.1 GP-SPI 118
7.3.2 GP-SPI 半 119
7.3.3 GP-SPI 三 半 119
7.3.4 GP-SPI 120
7.4 GP-SPI 时 120
7.4.1 GP-SPI 时 时 120
7.4.2 GP-SPI 时 121
7.5 并 QSPI 122
7.5.1 并 QSPI 122
7.6 GP-SPI 中 123
7.6.1 SPI 中 123
7.6.2 DMA 中 123
7.7 表 124
7.8 126

8 SDIO 从机控制器 146
8.1 146
8.2 146
8.3 146
8.3.1 SDIO Slave 146
8.3.2 SDIO 上 146
8.3.3 147
8.3.4 DMA 147
8.3.5 包 148
8.3.5.1 Slave Host 包 148
8.3.5.2 Slave Host 包 150
8.3.6 SDIO 时 151
8.3.7 中 152
8.3.7.1 Host 侧中 152
8.3.7.2 Slave 侧中 152

8.4 表 153
8.5 SLC 155
8.6 SLC Host 163
8.7 HINF 175

9 SD/MMC 主机控制器 176

9.1 176
9.2 176
9.3 SD/MMC 部 176
9.4 177
9.4.1 SD/MMC 177
9.4.1.1 BIU 178
9.4.1.2 CIU 178
9.4.2 178
9.4.3 179
9.4.3.1 179
9.4.3.2 180
9.5 CIU 操 180
9.6 RAM 181
9.6.1 RAM 181
9.6.2 RAM 181
9.7 表 181
9.8 表 182
9.9 始 184
9.9.1 DMAC 始 184
9.9.2 DMAC 始 184
9.9.3 DMAC 始 185
9.10 时 185
9.11 中 186
9.12 表 186
9.13 188

10 以太网 (MAC) 207

10.1 207
10.2 EMAC_CORE 209
10.2.1 操 209
10.2.1.1 209
10.2.1.2 期间 重 210
10.2.2 操 210
10.2.2.1 210
10.2.2.2 211
10.2.2.3 211
10.2.2.4 操 211
10.2.2.5 211
10.2.2.6 211
10.3 MAC 中 212

10.4 MAC 212
10.4.1 单播 标 212
10.4.2 播 标 212
10.4.3 播 212
10.4.4 单播 212
10.4.5 操 212
10.4.6 214
10.5 EMAC_MTL（MAC 层） 214
10.6 PHY 214
10.6.1 MII（ ） 214
10.6.1.1 MII PHY 间 215
10.6.1.2 MII 时 216
10.6.2 RMII（ ） 216
10.6.2.1 RMII 217
10.6.2.2 RMII 时 217
10.6.3 Station Management Agent (SMA) 218
10.6.4 RMII 时 218
10.7 DMA 219
10.8 表 219
10.8.1 219
10.8.2 223
10.9 表 228
10.10 229

11 I2C 控制器 (I2C) 264
11.1 264
11.2 264
11.3 I2C 264
11.3.1 I2C 264
11.3.2 I2C 265
11.3.3 I2C 时 266
11.3.4 I2C cmd 266
11.3.5 I2C 写 267
11.3.6 I2C 271
11.3.7 中 273
11.4 表 273
11.5 275

12 I2S 控制器 (I2S) 285
12.1 285
12.2 286
12.3 I2S 时 287
12.4 I2S 287
12.4.1 标 288
12.4.1.1 Philips 标 288
12.4.1.2 MSB 标 288

12.4.1.3 PCM 标 289
12.4.2 289
12.4.3 FIFO 操 289
12.4.4 289
12.4.5 291
12.4.6 I2S / 292
12.4.7 I2S PDM 293
12.5 Camera-LCD 294
12.5.1 LCD 295
12.5.2 Camera 295
12.5.3 ADC/DAC 296
12.6 I2S 中 297
12.6.1 FIFO 中 297
12.6.2 DMA 中 298
12.7 表 298
12.8 300

13 UART 控制器 (UART) 317
13.1 317
13.2 317
13.3 317
13.3.1 UART 317
13.3.2 UART 318
13.3.3 UART RAM 319
13.3.4 波 测 319
13.3.5 UART 320
13.3.6 320
13.3.6.1 321
13.3.6.2 321
13.3.7 UDMA 322
13.3.8 UART 中 322
13.3.9 UHCI 中 322
13.4 表 323
13.4.1 UART 323
13.4.2 UHCI 324
13.5 326

14 LED PWM 控制器 (LEDC) 353
14.1 353
14.2 353
14.2.1 353
14.2.2 354
14.2.3 355
14.2.4 中 356
14.3 表 356
14.4 359

15 红外遥控 (RMT) 368
15.1 368
15.2 368
15.2.1 RMT 368
15.2.2 RMT RAM 369
15.2.3 时 369
15.2.4 369
15.2.5 370
15.2.6 中 370
15.3 表 370
15.4 371

16 电机控制脉宽调制器 (PWM) 376
16.1 376
16.2 376
16.3 378
16.3.1 378
16.3.1.1 378
16.3.1.2 时 378
16.3.1.3 操 379
16.3.1.4 故 测 381
16.3.1.5 捕 381
16.3.2 PWM 时 381
16.3.2.1 PWM 时 381
16.3.2.2 PWM 时 时 382
16.3.2.3 PWM 时 386
16.3.2.4 PWM 时 步 386
16.3.3 PWM 操 386
16.3.3.1 PWM 387
16.3.3.2 死 397
16.3.3.3 PWM 波 401
16.3.3.4 故 404
16.3.4 捕 405
16.3.4.1 405
16.3.4.2 捕 时 405
16.3.4.3 捕 406
16.4 表 406
16.5 408

17 脉冲计数器 (PCNT) 452
17.1 452
17.2 452
17.2.1 452
17.2.2 452
17.2.3 察 453
17.2.4 453

17.2.5 中 454
17.3 表 454
17.4 456

18 定时器组 (TIMG) 461

18.1 461
18.2 461
18.2.1 16-bit 461
18.2.2 64-bit 时 461
18.2.3 报 产 462
18.2.4 MWDT 462
18.2.5 中 462
18.3 表 462
18.4 464

19 看门狗定时器 (WDT) 471

19.1 471
19.2 471
19.3 471
19.3.1 时 471
19.3.2 471
19.3.3 写保 472
19.3.4 Flash 启 保 472
19.3.5 472

20 eFuse 控制器 (eFuse) 473

20.1 473
20.2 473
20.3 473
20.3.1 473
20.3.1.1 参 efuse_wr_disable 474
20.3.1.2 参 efuse_rd_disable 474
20.3.1.3 参 coding_scheme 475
20.3.1.4 参 BLK3_part_reserve 476
20.3.2 写 参 476
20.3.3 参 479
20.3.4 参 480
20.3.5 中 480
20.4 表 481
20.5 483

21 双线汽车接口 (TWAI) 494

21.1 494
21.2 494
21.3 494
21.3.1 TWAI 494

21.3.2 TWAI 报 495
21.3.2.1 496
21.3.2.2 498
21.3.2.3 间 499
21.3.3 TWAI 500
21.3.3.1 500
21.3.3.2 500
21.3.3.3 500
21.3.4 TWAI 时 501
21.3.4.1 501
21.3.4.2 步 步 502
21.4 503
21.4.1 503
21.4.2 504
21.4.3 504
21.4.4 时 504
21.4.5 波 504
21.4.6 FIFO 504
21.5 504
21.5.1 504
21.5.1.1 505
21.5.1.2 操 505
21.5.2 时 505
21.5.3 中 506
21.5.3.1 中 (RXI) 506
21.5.3.2 中 (TXI) 506
21.5.3.3 报 中 (EWI) 506
21.5.3.4 中 (DOI) 507
21.5.3.5 被 中 (TXI) 507
21.5.3.6 裁 中 (ALI) 507
21.5.3.7 中 (BEI) 507
21.5.4 507
21.5.4.1 507
21.5.4.2 508
21.5.4.3 标识 508
21.5.4.4 509
21.5.5 FIFO 509
21.5.6 波 510
21.5.6.1 单 波 510
21.5.6.2 波 511
21.5.7 511
21.5.7.1 报 513
21.5.7.2 被 513
21.5.7.3 513
21.5.8 捕 513
21.5.9 裁 捕 514

21.6 表 515
21.7 516

22 AES 加速器 (AES) 528
22.1 528
22.2 528
22.3 528
22.3.1 528
22.3.2 、 、 528
22.3.3 528
22.3.4 531
22.3.5 531
22.4 表 531
22.5 533

23 SHA 加速器 (SHA) 535
23.1 535
23.2 535
23.3 535
23.3.1 535
23.3.2 535
23.3.3 535
23.3.4 536
23.4 表 536
23.5 538

24 RSA 加速器 (RSA) 543
24.1 543
24.2 543
24.3 543
24.3.1 始化 543
24.3.2 543
24.3.3 544
24.3.4 545
24.4 表 546
24.5 547

25 随机数发生器 (RNG) 549
25.1 549
25.2 549
25.3 549
25.4 编 549
25.5 表 550
25.6 550

26 片外存储器加密与解密 (FLASH) 551

26.1 551
26.2 551
26.3 551
26.3.1 Key Generator 552
26.3.2 Flash Encryption 552
26.3.3 Flash Decryption 553
26.4 表 553
26.5 553

27 存储器管理和保护单元 (MMU, MPU) 555
27.1 555
27.2 555
27.3 555
27.3.1 PID 555
27.3.2 MPU/MMU 555
27.3.2.1 555
27.3.2.2 562
27.3.2.3 设 567

28 PID 控制器 (PID) 569
28.1 569
28.2 569
28.3 569
28.3.1 中 识别 569
28.3.2 记录 570
28.3.3 572
28.4 表 573
28.5 575

29 片上传感器与模拟信号处理 579
29.1 579
29.2 579
29.2.1 579
29.2.2 579
29.2.3 可 580
29.2.4 580
29.2.5 580
29.3 SAR ADC 582
29.3.1 582
29.3.2 583
29.3.3 583
29.3.4 RTC SAR ADC 585
29.3.5 DIG SAR ADC 585
29.4 587
29.4.1 587
29.4.2 587

29.4.3 587
29.5 588
29.5.1 588
29.5.2 588
29.5.3 588
29.5.4 波 589
29.5.5 DMA 589
29.6 表 590
29.6.1 590
29.6.2 591
29.6.3 RTC I/O 591
29.7 592
29.7.1 592
29.7.2 602
29.7.3 RTC I/O 605

30 超低功耗协处理器 (ULP) 606
30.1 606
30.2 606
30.3 607
30.4 607
30.4.1 ALU - 607
30.4.1.1 608
30.4.1.2 608
30.4.1.3 609
30.4.2 ST 610
30.4.3 LD 610
30.4.4 JUMP 611
30.4.5 JUMPR (R0) 611
30.4.6 JUMPS ( ) 612
30.4.7 HALT 612
30.4.8 WAKE 612
30.4.9 SLEEP 设 时 期 613
30.4.10 WAIT 个 期 613
30.4.11 ADC ADC 测 613
30.4.12 I2C_RD / I2C_WR / 写 I²C 614
30.4.13 REG_RD 614
30.4.14 REG_WR 写 615
30.5 ULP 615
30.6 RTC_I2C 617
30.6.1 RTC_I2C 617
30.6.2 RTC_I2C 618
30.6.2.1 I2C_RD - 单 个 618
30.6.2.2 I2C_WR - 写 单 个 618
30.6.2.3 测 619
30.6.2.4 I²C 619

30.7 表 619
30.7.1 SENS_ULP 间 619
30.7.2 RTC_I2C 间 620
30.8 620
30.8.1 SENS_ULP 间 620
30.8.2 RTC_I2C 间 623

31 低功耗管理 (RTC_CNTL) 629
31.1 629
31.2 629
31.3 629
31.3.1 630
31.3.2 630
31.3.3 630
31.3.4 Flash 631
31.3.5 632
31.3.6 RTC 632
31.3.7 633
31.3.8 635
31.3.9 设 636
31.3.10 638
31.3.11 RTC 638
31.3.12 RTC Boot 638
31.4 表 640
31.5 642

词汇列表 666
设 666
666

修订历史 667

表格

1-1 26
1-2 27
1-3 DMA 29
1-4 29
1-5 Cache memory 30
1-6 31
2-1 PRO_CPU、APP_CPU 34
2-2 CPU 36
3-1 PRO_CPU APP_CPU 38
3-2 CPU_CLK 40
3-3 CPU_CLK 41
3-4 41
3-5 APB_CLK 42
3-6 REF_TICK 42
3-7 LEDC_SCLK 42
4-1 IO_MUX Light-sleep 48
4-2 GPIO 50
4-3 IO_MUX Pad 55
4-4 RTC_MUX 56
7-1 117
7-2 119
7-3 121
7-4 121
9-1 SD/MMC 177
9-2 DES0 182
9-3 DES1 183
9-4 DES2 184
9-5 DES3 184
10-1 213
10-2 213
10-3 218
10-4 219
10-5 0 (TDES0) 220
10-6 1 (TDES1) 222
10-7 2 (TDES2) 223
10-8 3 (TDES3) 223
10-9 6 (TDES6) 223
10-10 7 (TDES7) 223
10-11 0 (RDES0) 224
10-12 1 (RDES1) 225
10-13 2 (RDES2) 226
10-14 3 (RDES3) 226
10-15 4 (RDES4) 226

10-16 6 (RDES6) 227
10-17 7 (RDES7) 228
11-1 SCL 266
12-1 I2S 286
12-2 290
12-3 290
12-4 写 FIFO 291
12-5 4 292
12-6 采 293
12-7 采 294
14-1 常 354
16-1 操 参 380
16-2 PWM 中 时 388
16-3 PWM 时 时， 时 388
16-4 PWM 时 时， 时 389
16-5 死 时间 398
16-6 死 操 399
20-1 参 473
20-2 BLOCK1/2/3 编 475
20-3 写 476
20-4 时 478
20-5 479
21-1 SFF EFF 中 497
21-2 498
21-3 499
21-4 间 499
21-5 时 中包 502
21-6 TWAI_CLOCK_DIVIDER_REG bit ; TWAI 0x18 505
21-7 TWAI_BUS_TIMING_1_REG bit ; TWAI 0x1c 505
21-8 SFF EFF 布 507
21-9 TX/RX (SFF/EFF)；TWAI 0x40 508
21-10 TX/RX 标识 1 (SFF); TWAI 0x44 509
21-11 TX/RX 标识 2 (SFF); TWAI 0x48 509
21-12 TX/RX 标识 1 (EFF); TWAI 0x44 509
21-13 TX/RX 标识 2 (EFF); TWAI 0x48 509
21-14 TX/RX 标识 3 (EFF); TWAI 0x4c 509
21-15 TX/RX 标识 4 (EFF); TWAI 0x50 509
21-16 TWAI_ERR_CODE_CAP_REG bit ; TWAI 0x30 513
21-17 SEG.4 - SEG.0 514
21-18 TWAI_ARB LOST CAP_REG 中 ; TWAI 0x2c 515
22-1 528
22-2 AES 本 529
22-3 AES-128 530
22-4 AES-192 530
22-5 AES-256 530
27-1 [illegible] MPU MMU 556

27-2 RTC FAST Memory MPU 556
27-3 RTC SLOW Memory MPU 556
27-4 SRAM 0 SRAM2 128 KB MMU 557
27-5 SRAM0 MMU 边 558
27-6 SRAM2 MMU 边 559
27-7 DPORT_DMMU_TABLE*n*_REG DPORT_IMMU_TABLE*n*_REG 560
27-8 DMA MPU 设 561
27-9 562
27-10 PRO_CPU MMU 563
27-11 APP_CPU MMU 563
27-12 PRO_CPU MMU ( ) 563
27-13 APP_CPU MMU ( ) 564
27-14 SRAM 565
27-15 SRAM (常) 565
27-16 SRAM (-) 565
27-17 SRAM (- 奇) 566
27-18 RAM MMU 566
27-19 设 MPU 567
27-20 DPORT_AHBLITE_MPU_TABLE_*X*_REG 568
28-1 中 570
28-2 PIDCTRL_LEVEL_REG 570
28-3 PIDCTRL_FROM_*n*_REG 571
29-1 ESP32 580
29-2 SAR ADC 584
29-3 ESP32 SAR ADC 584
29-4 表 586
29-5 I DMA 587
29-6 II DMA 587
30-1 ALU 608
30-2 ALU 609
30-3 ALU 609
30-4 ADC 613
31-1 RTC 635
31-2 638

插图

1-1 25
1-2 25
1-3 Cache 30
2-1 中 33
3-1 38
3-2 时 39
4-1 IO_MUX、RTC IO_MUX GPIO 44
4-2 IO_MUX、GPIO 设 45
4-3 GPIO 47
4-4 ESP32 I/O Pad （QFN 6*6， 视 ） 49
4-5 ESP32 I/O Pad （QFN 5*5， 视 ） 50
6-1 DMA 113
6-2 表 114
6-3 UDMA 115
6-4 SPI DMA 115
7-1 SPI 117
7-2 SPI /半 118
7-3 SPI 120
7-4 GP-SPI 122
7-5 并 QSPI 123
7-6 并 QSPI 123
8-1 SDIO Slave 146
8-2 SDIO 上 147
8-3 CMD53 147
8-4 SDIO Slave DMA 表 147
8-5 表 148
8-6 Slave Host 包 149
8-7 Slave Host 包 150
8-8 Slave CPU buffer 151
8-9 采 时 151
8-10 时 152
9-1 SD/MMC 设 176
9-2 SD/MMC 部 177
9-3 SD/MMC 本 177
9-4 179
9-5 179
9-6 180
9-7 表 182
9-8 表 182
9-9 时 185
10-1 Ethernet MAC 207
10-2 Ethernet 209
10-3 MII 215

10-4 MII 时 216
10-5 RMII 217
10-6 RMII 时 218
10-7 时 218
10-8 时 219
10-9 219
10-10 表 223
11-1 I2C Master 本 265
11-2 I2C Slave 本 265
11-3 I2C 时 266
11-4 I2C 267
11-5 I2C Master 写 7-bit Slave 268
11-6 I2C Master 写 10-bit Slave 269
11-7 I2C Master 写 7-bit Slave M RAM 269
11-8 I2C Master 写 7-bit Slave 270
11-9 I2C Master 7-bit Slave 271
11-10 I2C Master 10-bit Slave 271
11-11 I2C Master 7-bit Slave M N 个 272
11-12 I2C Master 7-bit Slave 273
12-1 I2S 285
12-2 I2S 时 287
12-3 Philips 标 288
12-4 MSB 标 288
12-5 PCM 标 289
12-6 FIFO 290
12-7 一 291
12-8 写 FIFO 292
12-9 PDM 293
12-10 PDM 294
12-11 PDM 294
12-12 PDM 294
12-13 LCD 295
12-14 LCD 1 295
12-15 LCD 2 295
12-16 Camera 296
12-17 I2S ADC 296
12-18 I2S DAC 297
12-19 I2S DAC 297
13-1 UART 本 318
13-2 UART RAM 319
13-3 UART 320
13-4 AT_CMD 320
13-5 321
14-1 LED_PWM 353
14-2 LED_PWM 353
14-3 LED_PWM 354

14-4 LED_PWM 355
14-5 变 比 355
15-1 RMT 368
15-2 369
16-1 MCPWM 设 376
16-2 378
16-3 时 378
16-4 操 379
16-5 故 测 381
16-6 捕 381
16-7 波 382
16-8 波 383
16-9 波 , 步 383
16-10 波 , 步 383
16-11 中 UTEP UTEZ 384
16-12 中 UTEP UTEZ 385
16-13 中 UTEP UTEZ 386
16-14 PWM 操 387
16-15 波 390
16-16 , 单边不 波 , PWMxA PWMxB 门 391
16-17 , 脉 不 波 , PWMxA 门 392
16-18 , 波 , PWMxA PWMxB 上 门 393
16-19 , 波 , PWMxA PWMxB 上 门 补 394
16-20 NCI PWMxA 上 395
16-21 CNTU PWMxB 上 396
16-22 死 398
16-23 补 (AHC) 死 波 399
16-24 补 (ALC) 死 波 399
16-25 (AH) 死 波 400
16-26 (AL) 死 波 400
16-27 PWM 波操 波 402
16-28 波 一个脉 脉 403
16-29 PWM 波 中 脉 7 比设 404
17-1 PULSE_CNT单 本 452
17-2 PULSE_CNT 454
17-3 PULSE_CNT 454
21-1 中 496
21-2 中 498
21-3 中 498
21-4 间 中 499
21-5 时 502
21-6 TWAI 503
21-7 波 510
21-8 单 波 511
21-9 波 512
21-10 变化 512

21-11 裁 bit 515
25-1 声 549
26-1 Flash 551
27-1 MMU 557
28-1 中 572
29-1 579
29-2 部 580
29-3 581
29-4 FSM 部 582
29-5 SAR ADC 582
29-6 SAR ADC 583
29-7 RTC SAR ADC 585
29-8 DIG SAR ADC 586
29-9 588
29-10 DAC 589
29-11 波 590
30-1 ULP 本 606
30-2 ULP 607
30-3 - ALU 608
30-4 - ALU 608
30-5 - ALU 609
30-6 - ST 610
30-7 - LD 610
30-8 - JUMP 611
30-9 - JUMPR 611
30-10 - JUMPS 612
30-11 - HALT 612
30-12 - WAKE 612
30-13 - SLEEP 613
30-14 - WAIT 613
30-15 - ADC 613
30-16 - I²C 614
30-17 - REG_RD 614
30-18 - REG_WR 615
30-19 ULP 616
30-20 ULP 617
30-21 I2C 操 618
30-22 I²C 写操 619
31-1 ESP32 629
31-2 门 630
31-3 门 631
31-4 Flash 门 632
31-5 测 632
31-6 RTC 633
31-7 RTC 时 634
31-8 时 634

31-9 RTC 635
31-10 637
31-11 ESP32启 639

1-1 。 1-2 。

图 1-1. 系统结构

图 1-2. 地址映射结构

表 1-2. 片上寄存器地址映射

	边			标	备
	0x3FF8_0000	0x3FF8_1FFF	8 KB	RTC FAST Memory	PRO_CPU Only
	0x3FF8_2000	0x3FF8_FFFF	56 KB	保	
	0x3FF9_0000	0x3FF9_FFFF	64 KB	Internal ROM 1	
	0x3FFA_0000	0x3FFA_DFFF	56 KB	保	
	0x3FFA_E000	0x3FFD_FFFF	200 KB	Internal SRAM 2	DMA
	0x3FFE_0000	0x3FFF_FFFF	128 KB	Internal SRAM 1	DMA
	边			标	备
	0x4000_0000	0x4000_7FFF	32 KB	Internal ROM 0	Remap
	0x4000_8000	0x4005_FFFF	352 KB	Internal ROM 0	
	0x4006_0000	0x4006_FFFF	64 KB	保	
	0x4007_0000	0x4007_FFFF	64 KB	Internal SRAM 0	Cache
	0x4008_0000	0x4009_FFFF	128 KB	Internal SRAM 0	
	0x400A_0000	0x400A_FFFF	64 KB	Internal SRAM 1	
	0x400B_0000	0x400B_7FFF	32 KB	Internal SRAM 1	Remap
	0x400B_8000	0x400B_FFFF	32 KB	Internal SRAM 1	
	0x400C_0000	0x400C_1FFF	8 KB	RTC FAST Memory	PRO_CPU Only
	边			标	备
	0x5000_0000	0x5000_1FFF	8 KB	RTC SLOW Memory	

1.3.2.1 Internal ROM 0

Internal ROM 0 384 KB，可 被 个 CPU 0x4000_0000 ~ 0x4005_FFFF 。

ROM 0 32 KB （0x4000_0000 ~ 0x4000_7FFF）可 被重 Internal SRAM 1 中 一部 ， 部 本被 0x400B_0000 ~ 0x400B_7FFF 。重 时， 32 KB SRAM 不 被 0x400B_0000 ~ 0x400B_7FFF ， 可 被 (0x3FFE_8000 ~ 0x3FFE_FFFF) 。 别 个 CPU 一个 ， PRO_CPU DPORT_PRO_BOOT_REMAP_CTRL_REG bit 0 APP_CPU DPORT_APP_BOOT_REMAP_CTRL_REG bit 0。

1.3.2.2 Internal ROM 1

Internal ROM 1 64 KB， 可 被 个 CPU 0x3FF9_0000 ~ 0x3FF9_FFFF 。

1.3.2.3 Internal SRAM 0

Internal SRAM 0 192 KB， ， 64 KB 可 Cache 。不 Cache 时， 64 KB 可 被 个 CPU 0x4007_0000 ~ 0x4007_FFFF 写， 128 KB 可 被 个 CPU 0x4008_0000 ~ 0x4009_FFFF 写。

1.3.2.4 Internal SRAM 1

Internal SRAM 1 128 KB，可被个 CPU 0x3FFE_0000 ~ 0x3FFF_FFFF 写，可被个 CPU 0x400A_0000 ~ 0x400B_FFFF 写。

word 。 :

0x3FFE_0000 0x400B_FFFC word
0x3FFE_0004 0x400B_FFF8 word
0x3FFE_0008 0x400B_FFF4 word
……
0x3FFF_FFF4 0x400A_0008 word
0x3FFF_FFF8 0x400A_0004 word
0x3FFF_FFFC 0x400A_0000 word

CPU 。 间个 word 顺不 。 :

0x3FFE_0000 0x400B_FFFC word 中
0x3FFE_0001 0x400B_FFFC word 中
0x3FFE_0002 0x400B_FFFC word 中
0x3FFE_0003 0x400B_FFFC word 中
0x3FFE_0004 0x400B_FFF8 word 中
0x3FFE_0005 0x400B_FFF8 word 中
0x3FFE_0006 0x400B_FFF8 word 中
0x3FFE_0007 0x400B_FFF8 word 中
……
0x3FFF_FFF8 0x400A_0004 word 中
0x3FFF_FFF9 0x400A_0004 word 中
0x3FFF_FFFA 0x400A_0004 word 中
0x3FFF_FFFB 0x400A_0004 word 中
0x3FFF_FFFC 0x400A_0000 word 中
0x3FFF_FFFD 0x400A_0000 word 中
0x3FFF_FFFE 0x400A_0000 word 中
0x3FFF_FFFF 0x400A_0000 word 中

部 可被重 ROM 0 间。 参见 Internal Rom 0。

1.3.2.5 Internal SRAM 2

Internal SRAM 2 200 KB，可被个 CPU 0x3FFA_E000 ~ 0x3FFD_FFFF 写。

1.3.2.6 DMA

DMA CPU 写 Internal SRAM 1 Internal SRAM 2。 DMA 0x3FFE_0000 ~ 0x3FFF_FFFF 写 Internal SRAM 1，0x3FFA_E000 ~ 0x3FFD_FFFF 写 Internal SRAM 2。

中 DMA 13 个。表 1-3 DMA 。

表 1-3. 具有 DMA 功能的模块

UART0	UART1	UART2
SPI1	SPI2	SPI3
I2S0	I2S1	
SDIO Slave	SDMMC	
EMAC		
BT	WIFI	

1.3.2.7 RTC FAST Memory

RTC FAST Memory 8 KB SRAM， 被 PRO_CPU 0x3FF8_0000 ~ 0x3FF8_1FFF 写，被 PRO_CPU 0x400C_0000 ~ 0x400C_1FFF 写。 不 ，APP_CPU 不 RTC FAST Memory。

PRO_CPU RTC FAST Memory。 0x3FF8_0000 0x400C_0000 word，0x3FF8_0004 0x400C_0004 word，0x3FF8_0008 0x400C_0008 word， 。APP_CPU 不 RTC FAST Memory， 不 标。

1.3.2.8 RTC SLOW Memory

RTC SLOW Memory 8 KB SRAM， 可 被 个 CPU 0x5000_0000 ~ 0x5000_1FFF 写。

1.3.3 片外存储器

ESP32 External Flash External SRAM 。表 1-4 个 CPU 中 Cache MMU 。 个 CPU Cache MMU 时，Cache MMU 中 设 把 CPU 变 External Flash External SRAM 。 变 16 MB External Flash 8 MB External SRAM。

表 1-4. 片外存储器地址映射

	边			标	备
	0x3F40_0000	0x3F7F_FFFF	4 MB	External Flash	
	0x3F80_0000	0x3FBF_FFFF	4 MB	External SRAM	/ 写
	边			标	备
	0x400C_2000	0x40BF_FFFF	11512 KB	External Flash	

1.3.4 Cache

1-3 ，ESP32 2 个 CPU 一 32 KB cache， 部 。PRO CPU APP CPU 别 DPORT_PRO_CACHE_CTRL_REG PRO_CACHE_ENABLE DPORT_APP_CACHE_CTRL_REG APP_CACHE_ENABLE Cache 。

ESP32 Cache 采 。 PRO CPU Cache APP CPU Cache 时，可 DPORT_CACHE_MUX_MODE_REG CACHE_MUX_MODE[1:0] ， Internal

图 1-3. Cache 系统框图

SRAM0 POOL0 POOL1 cache memory。 PRO CPU APP CPU Cache 时，Internal SRAM0 POOL0 POOL1 可 cache memory。 见表 1-5。

表 1-5. Cache memory 模式

CACHE_MUX_MODE	POOL0	POOL1
0	PRO CPU	APP CPU
1	PRO CPU/APP CPU	-
2	-	PRO CPU/APP CPU
3	APP CPU	PRO CPU

表 1-5 可 ， CACHE_MUX_MODE 1 2 时，PRO CPU APP CPU 不可 时启 Cache 。启 Cache ，POOL0 POOL1 cache memory ，不 。

ESP32 Cache Flush 。 ， Flush 时，写 cache 被 ，并不 写 External SRAM 中。 flush 操 方 ： DPORT_*x*_CACHE_CTRL_REG *x*_CACHE_FLUSH_ENA 0， 1。 ， 中 *x*_CACHE_FLUSH_DONE 1 时，表 cache flush 操 ， 中 *x* 表 “PRO” “APP”。

ESP32 Cache 见下图 。

1.3.5 外设

ESP32 41 个 设 。表 1-6 个 CPU 中 个 设 。 PID Controller ， 设 可 被 个 CPU 。

表 1-6. 外设地址映射

				标	备
	0x3FF0_0000	0x3FF0_0FFF	4 KB	DPort Register	
	0x3FF0_1000	0x3FF0_1FFF	4 KB	AES Accelerator	
	0x3FF0_2000	0x3FF0_2FFF	4 KB	RSA Accelerator	
	0x3FF0_3000	0x3FF0_3FFF	4 KB	SHA Accelerator	
	0x3FF0_4000	0x3FF0_4FFF	4 KB	Secure Boot	
	0x3FF0_5000	0x3FF0_FFFF	44 KB	保	
	0x3FF1_0000	0x3FF1_3FFF	16 KB	Cache MMU Table	
	0x3FF1_4000	0x3FF1_EFFF	44 KB	保	
	0x3FF1_F000	0x3FF1_FFFF	4 KB	PID Controller	个 CPU单 设
	0x3FF2_0000	0x3FF3_FFFF	128 KB	保	
	0x3FF4_0000	0x3FF4_0FFF	4 KB	UART0	
	0x3FF4_1000	0x3FF4_1FFF	4 KB	保	
	0x3FF4_2000	0x3FF4_2FFF	4 KB	SPI1	
	0x3FF4_3000	0x3FF4_3FFF	4 KB	SPI0	
	0x3FF4_4000	0x3FF4_4FFF	4 KB	GPIO	
	0x3FF4_5000	0x3FF4_7FFF	12 KB	保	
	0x3FF4_8000	0x3FF4_8FFF	4 KB	RTC	
	0x3FF4_9000	0x3FF4_9FFF	4 KB	IO MUX	
	0x3FF4_A000	0x3FF4_AFFF	4 KB	保	
	0x3FF4_B000	0x3FF4_BFFF	4 KB	SDIO Slave	三个部 一
	0x3FF4_C000	0x3FF4_CFFF	4 KB	UDMA1	
	0x3FF4_D000	0x3FF4_EFFF	8 KB	保	
	0x3FF4_F000	0x3FF4_FFFF	4 KB	I2S0	
	0x3FF5_0000	0x3FF5_0FFF	4 KB	UART1	
	0x3FF5_1000	0x3FF5_2FFF	8 KB	保	
	0x3FF5_3000	0x3FF5_3FFF	4 KB	I2C0	
	0x3FF5_4000	0x3FF5_4FFF	4 KB	UDMA0	
	0x3FF5_5000	0x3FF5_5FFF	4 KB	SDIO Slave	三个部 一
	0x3FF5_6000	0x3FF5_6FFF	4 KB	RMT	
	0x3FF5_7000	0x3FF5_7FFF	4 KB	PCNT	
	0x3FF5_8000	0x3FF5_8FFF	4 KB	SDIO Slave	三个部 一
	0x3FF5_9000	0x3FF5_9FFF	4 KB	LED PWM	
	0x3FF5_A000	0x3FF5_AFFF	4 KB	Efuse Controller	
	0x3FF5_B000	0x3FF5_BFFF	4 KB	Flash Encryption	
	0x3FF5_C000	0x3FF5_DFFF	8 KB	保	
	0x3FF5_E000	0x3FF5_EFFF	4 KB	PWM0	
	0x3FF5_F000	0x3FF5_FFFF	4 KB	TIMG0	
	0x3FF6_0000	0x3FF6_0FFF	4 KB	TIMG1	
	0x3FF6_1000	0x3FF6_3FFF	12 KB	保	
	0x3FF6_4000	0x3FF6_4FFF	4 KB	SPI2	
	0x3FF6_5000	0x3FF6_5FFF	4 KB	SPI3	

				标	备
	0x3FF6_6000	0x3FF6_6FFF	4 KB	SYSCON	
	0x3FF6_7000	0x3FF6_7FFF	4 KB	I2C1	
	0x3FF6_8000	0x3FF6_8FFF	4 KB	SDMMC	
	0x3FF6_9000	0x3FF6_AFFF	8 KB	EMAC	
	0x3FF6_B000	0x3FF6_BFFF	4 KB	保	
	0x3FF6_C000	0x3FF6_CFFF	4 KB	PWM1	
	0x3FF6_D000	0x3FF6_DFFF	4 KB	I2S1	
	0x3FF6_E000	0x3FF6_EFFF	4 KB	UART2	
	0x3FF6_F000	0x3FF6_FFFF	4 KB	PWM2	
	0x3FF7_0000	0x3FF7_0FFF	4 KB	PWM3	
	0x3FF7_1000	0x3FF7_4FFF	16 KB	保	
	0x3FF7_5000	0x3FF7_5FFF	4 KB	RNG	
	0x3FF7_6000	0x3FF7_FFFF	40 KB	保	

注意：

- 0x3FF40000 ~ 0x3FF7FFFF 间 (DPORT) 设，CPU 可 0x60000000 ~ 0x6003FFFF 间 (AHB) 。 (0x3FF40000 + n) (0x60000000 + n) ， 中 n = 0 ~ 0x3FFFF。
- 比 AHB 设，CPU DPORT 设 。 DPORT 测 (speculative read) ，不 保 一 。 ，DPORT 上 写操 顺 升 ， 可 写操 顺 崩 。 AHB FIFO 。 时 《ESP32 表 办 》中 3.3、3.10、3.16 3.17 。

1.3.5.1 不对称 PID Controller 外设

中 个 PID Controller 别 PRO_CPU APP_CPU。**PRO_CPU 和 APP_CPU 都只能访问自己的 PID Controller，不能访问对方的 PID Controller。** 个 CPU 0x3FF1_F000 ~ 3FF1_FFFF PID Controller。

1.3.5.2 不连续外设地址范围

设 SDIO Slave 被 三部 。 个 CPU 三部 不 。 三部 别被 个 CPU 0x3FF4_B000 ~ 3FF4_BFFF、0x3FF5_5000 ~ 3FF5_5FFF、0x3FF5_8000 ~ 3FF5_8FFF 。 设一 ，SDIO Slave 被 个 CPU 。

1.3.5.3 存储器速度

ROM SRAM 时 CPU_CLK，CPU 可 单 个时 期 个 。 RTC FAST Memory 时 APB_CLOCK，RTC SLOW Memory 时 FAST_CLOCK， CPU 个 。DMA APB_CLK 时 。

SRAM 32K 一个 。 时 不 ， CPU DMA 可 时 SRAM。

2 中断矩阵 (INTERRUPT)

2.1 概述

ESP32 中 一 部中 单 个 CPU 一 部中 上。 ，
不 。

2.2 主要特性

- 71 个 部中
- 个 CPU 别 26 个 部中 （ 52 个）
- 屏蔽 CPU NMI 中
- 查 部中 中

中 2-1 。

图 2-1. 中断矩阵结构图

2.3 功能描述

2.3.1 外部中断源

ESP32 71 个 部中 。表 2-1 部中 。ESP32 中 71 个 部中 中 67 个可 个 CPU。 4 个 部中 CPU， 个 CPU 2 个。GPIO_INTERRUPT_PRO GPIO_INTERRUPT_PRO_NMI 可 PRO_CPU，GPIO_INTERRUPT_APP GPIO_INTERRUPT_APP_NMI 可 APP_CPU。 ，PRO_CPU APP_CPU 可 69 个 部中 。

表 2-1. PRO_CPU、APP_CPU 外部中断配置寄存器、外部中断源中断状态寄存器、外部中断源

PRO_CPU						APP_CPU			
Peripheral Interrupt Configuration Register	Peripheral Interrupt Source								Peripheral Interrupt Configuration Register
	Bit	Status Register Name	No.	Name		No.	Status Register Name	Bit	
PRO_MAC_INTR_MAP_REG	0	PRO_INTR_STATUS_REG_0	0	MAC_INTR		0	APP_INTR_STATUS_REG_0	0	APP_MAC_INTR_MAP_REG
PRO_MAC_NMI_MAP_REG	1		1	MAC_NMI		1		1	APP_MAC_NMI_MAP_REG
PRO_BB_INT_MAP_REG	2		2	BB_INT		2		2	APP_BB_INT_MAP_REG
PRO_BT_MAC_INT_MAP_REG	3		3	BT_MAC_INT		3		3	APP_BT_MAC_INT_MAP_REG
PRO_BT_BB_INT_MAP_REG	4		4	BT_BB_INT		4		4	APP_BT_BB_INT_MAP_REG
PRO_BT_BB_NMI_MAP_REG	5		5	BT_BB_NMI		5		5	APP_BT_BB_NMI_MAP_REG
PRO_RWBT_IRQ_MAP_REG	6		6	RWBT_IRQ		6		6	APP_RWBT_IRQ_MAP_REG
PRO_RWBLE_IRQ_MAP_REG	7		7	RWBLE_IRQ		7		7	APP_RWBLE_IRQ_MAP_REG
PRO_RWBT_NMI_MAP_REG	8		8	RWBT_NMI		8		8	APP_RWBT_NMI_MAP_REG
PRO_RWBLE_NMI_MAP_REG	9		9	RWBLE_NMI		9		9	APP_RWBLE_NMI_MAP_REG
PRO_SLC0_INTR_MAP_REG	10		10	SLC0_INTR		10		10	APP_SLC0_INTR_MAP_REG
PRO_SLC1_INTR_MAP_REG	11		11	SLC1_INTR		11		11	APP_SLC1_INTR_MAP_REG
PRO_UHCI0_INTR_MAP_REG	12		12	UHCI0_INTR		12		12	APP_UHCI0_INTR_MAP_REG
PRO_UHCI1_INTR_MAP_REG	13		13	UHCI1_INTR		13		13	APP_UHCI1_INTR_MAP_REG
PRO_TG_T0_LEVEL_INT_MAP_REG	14		14	TG_T0_LEVEL_INT		14		14	APP_TG_T0_LEVEL_INT_MAP_REG
PRO_TG_T1_LEVEL_INT_MAP_REG	15		15	TG_T1_LEVEL_INT		15		15	APP_TG_T1_LEVEL_INT_MAP_REG
PRO_TG_WDT_LEVEL_INT_MAP_REG	16		16	TG_WDT_LEVEL_INT		16		16	APP_TG_WDT_LEVEL_INT_MAP_REG
PRO_TG_LACT_LEVEL_INT_MAP_REG	17		17	TG_LACT_LEVEL_INT		17		17	APP_TG_LACT_LEVEL_INT_MAP_REG
PRO_TG1_T0_LEVEL_INT_MAP_REG	18		18	TG1_T0_LEVEL_INT		18		18	APP_TG1_T0_LEVEL_INT_MAP_REG
PRO_TG1_T1_LEVEL_INT_MAP_REG	19		19	TG1_T1_LEVEL_INT		19		19	APP_TG1_T1_LEVEL_INT_MAP_REG
PRO_TG1_WDT_LEVEL_INT_MAP_REG	20		20	TG1_WDT_LEVEL_INT		20		20	APP_TG1_WDT_LEVEL_INT_MAP_REG
PRO_TG1_LACT_LEVEL_INT_MAP_REG	21		21	TG1_LACT_LEVEL_INT		21		21	APP_TG1_LACT_LEVEL_INT_MAP_REG
PRO_GPIO_INTERRUPT_PRO_MAP_REG	22		22	**GPIO_INTERRUPT_PRO**	**GPIO_INTERRUPT_APP**	22		22	APP_GPIO_INTERRUPT_APP_MAP_REG
PRO_GPIO_INTERRUPT_PRO_NMI_MAP_REG	23		23	**GPIO_INTERRUPT_PRO_NMI**	**GPIO_INTERRUPT_APP_NMI**	23		23	APP_GPIO_INTERRUPT_APP_NMI_MAP_REG
PRO_CPU_INTR_FROM_CPU_0_MAP_REG	24		24	CPU_INTR_FROM_CPU_0		24		24	APP_CPU_INTR_FROM_CPU_0_MAP_REG
PRO_CPU_INTR_FROM_CPU_1_MAP_REG	25		25	CPU_INTR_FROM_CPU_1		25		25	APP_CPU_INTR_FROM_CPU_1_MAP_REG
PRO_CPU_INTR_FROM_CPU_2_MAP_REG	26		26	CPU_INTR_FROM_CPU_2		26		26	APP_CPU_INTR_FROM_CPU_2_MAP_REG
PRO_CPU_INTR_FROM_CPU_3_MAP_REG	27		27	CPU_INTR_FROM_CPU_3		27		27	APP_CPU_INTR_FROM_CPU_3_MAP_REG
PRO_SPI_INTR_0_MAP_REG	28		28	SPI_INTR_0		28		28	APP_SPI_INTR_0_MAP_REG
PRO_SPI_INTR_1_MAP_REG	29		29	SPI_INTR_1		29		29	APP_SPI_INTR_1_MAP_REG
PRO_SPI_INTR_2_MAP_REG	30		30	SPI_INTR_2		30		30	APP_SPI_INTR_2_MAP_REG
PRO_SPI_INTR_3_MAP_REG	31		31	SPI_INTR_3		31		31	APP_SPI_INTR_3_MAP_REG
PRO_I2S0_INT_MAP_REG	0	PRO_INTR_STATUS_REG_1	32	I2S0_INT		32	APP_INTR_STATUS_REG_1	0	APP_I2S0_INT_MAP_REG
PRO_I2S1_INT_MAP_REG	1		33	I2S1_INT		33		1	APP_I2S1_INT_MAP_REG
PRO_UART_INTR_MAP_REG	2		34	UART_INTR		34		2	APP_UART_INTR_MAP_REG
PRO_UART1_INTR_MAP_REG	3		35	UART1_INTR		35		3	APP_UART1_INTR_MAP_REG
PRO_UART2_INTR_MAP_REG	4		36	UART2_INTR		36		4	APP_UART2_INTR_MAP_REG
PRO_SDIO_HOST_INTERRUPT_MAP_REG	5		37	SDIO_HOST_INTERRUPT		37		5	APP_SDIO_HOST_INTERRUPT_MAP_REG
PRO_EMAC_INT_MAP_REG	6		38	EMAC_INT		38		6	APP_EMAC_INT_MAP_REG
PRO_PWM0_INTR_MAP_REG	7		39	PWM0_INTR		39		7	APP_PWM0_INTR_MAP_REG
PRO_PWM1_INTR_MAP_REG	8		40	PWM1_INTR		40		8	APP_PWM1_INTR_MAP_REG
PRO_PWM2_INTR_MAP_REG	9		41	PWM2_INTR		41		9	APP_PWM2_INTR_MAP_REG
PRO_PWM3_INTR_MAP_REG	10		42	PWM3_INTR		42		10	APP_PWM3_INTR_MAP_REG
PRO_LEDC_INT_MAP_REG	11		43	LEDC_INT		43		11	APP_LEDC_INT_MAP_REG
PRO_EFUSE_INT_MAP_REG	12		44	EFUSE_INT		44		12	APP_EFUSE_INT_MAP_REG
PRO_CAN_INT_MAP_REG	13		45	CAN_INT		45		13	APP_CAN_INT_MAP_REG
PRO_RTC_CORE_INTR_MAP_REG	14		46	RTC_CORE_INTR		46		14	APP_RTC_CORE_INTR_MAP_REG
PRO_RMT_INTR_MAP_REG	15		47	RMT_INTR		47		15	APP_RMT_INTR_MAP_REG
PRO_PCNT_INTR_MAP_REG	16		48	PCNT_INTR		48		16	APP_PCNT_INTR_MAP_REG
PRO_I2C_EXT0_INTR_MAP_REG	17		49	I2C_EXT0_INTR		49		17	APP_I2C_EXT0_INTR_MAP_REG
PRO_I2C_EXT1_INTR_MAP_REG	18		50	I2C_EXT1_INTR		50		18	APP_I2C_EXT1_INTR_MAP_REG
PRO_RSA_INTR_MAP_REG	19		51	RSA_INTR		51		19	APP_RSA_INTR_MAP_REG
PRO_SPI1_DMA_INT_MAP_REG	20		52	SPI1_DMA_INT		52		20	APP_SPI1_DMA_INT_MAP_REG

PRO_CPU					APP_CPU			
Peripheral Interrupt Configuration Register	Peripheral Interrupt Source							Peripheral Interrupt Configuration Register
	Status Register		No.	Name	No.	Status Register		
	Bit	Name				Name	Bit	
PRO_SPI2_DMA_INT_MAP_REG	21	PRO_INTR_STATUS_REG_1	53	SPI2_DMA_INT	53	APP_INTR_STATUS_REG_1	21	APP_SPI2_DMA_INT_MAP_REG
PRO_SPI3_DMA_INT_MAP_REG	22		54	SPI3_DMA_INT	54		22	APP_SPI3_DMA_INT_MAP_REG
PRO_WDG_INT_MAP_REG	23		55	WDG_INT	55		23	APP_WDG_INT_MAP_REG
PRO_TIMER_INT1_MAP_REG	24		56	TIMER_INT1	56		24	APP_TIMER_INT1_MAP_REG
PRO_TIMER_INT2_MAP_REG	25		57	TIMER_INT2	57		25	APP_TIMER_INT2_MAP_REG
PRO_TG_T0_EDGE_INT_MAP_REG	26		58	TG_T0_EDGE_INT	58		26	APP_TG_T0_EDGE_INT_MAP_REG
PRO_TG_T1_EDGE_INT_MAP_REG	27		59	TG_T1_EDGE_INT	59		27	APP_TG_T1_EDGE_INT_MAP_REG
PRO_TG_WDT_EDGE_INT_MAP_REG	28		60	TG_WDT_EDGE_INT	60		28	APP_TG_WDT_EDGE_INT_MAP_REG
PRO_TG_LACT_EDGE_INT_MAP_REG	29		61	TG_LACT_EDGE_INT	61		29	APP_TG_LACT_EDGE_INT_MAP_REG
PRO_TG1_T0_EDGE_INT_MAP_REG	30		62	TG1_T0_EDGE_INT	62		30	APP_TG1_T0_EDGE_INT_MAP_REG
PRO_TG1_T1_EDGE_INT_MAP_REG	31		63	TG1_T1_EDGE_INT	63		31	APP_TG1_T1_EDGE_INT_MAP_REG
PRO_TG1_WDT_EDGE_INT_MAP_REG	0	PRO_INTR_STATUS_REG_2	64	TG1_WDT_EDGE_INT	64	APP_INTR_STATUS_REG_2	0	APP_TG1_WDT_EDGE_INT_MAP_REG
PRO_TG1_LACT_EDGE_INT_MAP_REG	1		65	TG1_LACT_EDGE_INT	65		1	APP_TG1_LACT_EDGE_INT_MAP_REG
PRO_MMU_IA_INT_MAP_REG	2		66	MMU_IA_INT	66		2	APP_MMU_IA_INT_MAP_REG
PRO_MPU_IA_INT_MAP_REG	3		67	MPU_IA_INT	67		3	APP_MPU_IA_INT_MAP_REG
PRO_CACHE_IA_INT_MAP_REG	4		68	CACHE_IA_INT	68		4	APP_CACHE_IA_INT_MAP_REG

2.3.2 CPU 中断

个 CPU（PRO_CPU APP_CPU） 32 个中 ， 中 26 个 部中 。表 2-2 个 CPU 中 。

表 2-2. CPU 中断

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

2.3.3 分配外部中断源至 CPU 外部中断

本 中：

- 记 Source_X 表 个 部中 。
- 记 PRO_X_MAP_REG（ APP_X_MAP_REG）表 PRO_CPU（ APP_CPU） 个 部中

，部中 部中 Source_X 。表 2-1 中 “PRO_CPU（APP_CPU）- Peripheral Interrupt Configuration Register” 一 中 “Peripheral Interrupt Source - Name” 一 中 个 部中 一 。

- 记 Interrupt_P 表 CPU 中 Num_P 部中 ，Num_P 0 ~ 5、8 ~ 10、12 ~ 14、17 ~ 28、30 ~ 31。
- 记 Interrupt_I 表 CPU 中 Num_I 部中 ，Num_I 6、7、11、15、16、29。

上 ，可 中 操 ：

- **将外部中断源 Source_X 分配到 CPU（PRO_CPU 或 APP_CPU）**

  PRO_X_MAP_REG（APP_X_MAP_REG） Num_P。Num_P 可 CPU 部中 。CPU 中 可 被 个 设 （具 ）。

- **关闭 CPU（PRO_CPU 或 APP_CPU）外部中断源 Source_X**

  PRO_X_MAP_REG（APP_X_MAP_REG） Num_I。 被 Num_I 中 2 个 CPU 上， 部中 不 。

- **将多个外部中断源 Source_X*n* ORed 分配到 PRO_CPU（APP_CPU）的外部中断**

  个 PRO_X*n*_MAP_REG (APP_X*n*_MAP_REG) Num_P。 设中 CPU Interrupt_P。

2.3.4 屏蔽 CPU 的 NMI 类型中断

中 PRO_CPU NMI Interrupt Mask（ APP_CPU NMI Interrupt Mask） 时 屏蔽 被 PRO_CPU（ APP_CPU） 部中 NMI 中 。 PRO_CPU NMI Interrupt Mask APP_CPU NMI Interrupt Mask 别 设 PID Controller。

2.3.5 查询外部中断源当前的中断状态

PRO_INTR_STATUS_REG_*n*（APP_INTR_STATUS_REG_*n*） 中 Bit 可 部中 中 。 PRO_INTR_STATUS_REG_*n*（APP_INTR_STATUS_REG_*n*） 部中 表 2-1 。

3 复位和时钟

3.1 System 复位

3.1.1 概述

三 别 ，别 CPU ， ， 。

不 MEM 中 。 3-1 个 ：

- CPU : CPU
- : RTC， 部 ，包括 CPU IO。
- : 个 ，包括 RTC。

3.1.2 复位源

，APP_CPU PRO_CPU 被 ， _CPU PRO_CPU 不 ： PRO_CPU RTC_CNTL_RESET_CAUSE_PROCPU ，A RTC_CNTL_RESET_CAUSE_APPCPU 。

表 3-1 中可 。

表 3-1. PRO_C

PRO	APP			
0x01	0x01			

PRO	APP			
0x08	0x08	MWDT1		WDT
0x09	0x09	RWDT		WDT
0x0B	-	MWDT0 CPU	CPU	WDT
0x0C	-	CPU	CPU	RTC_CNTL_SW_APPCPU_RST
-	0x0B	MWDT1 CPU	CPU	WDT
-	0x0C	CPU	CPU	RTC_CNTL_SW_APPCPU_RST
0x0D	0x0D	RWDT CPU	CPU	WDT
-	0xE	PRO CPU	CPU	表 PRO CPU DPORT_APPCPU_RESETTING 单 APP CPU

3.2 系统时钟

3.2.1 概述

ESP32 不 时 ，可 CPU， 设， RTC ， 不 。 3-2 时 。

图 3-2. 系统时钟

3.2.2 时钟源

ESP32 时 别 部 、 部 PLL 。 ， 时 ：

- 时
  - PLL_CLK，320 MHz 480 MHz 部 PLL 时
  - XTL_CLK，2 ~ 40 MHz 部 时
- 时
  - XTL32K_CLK，32 KHz 部 时
  - RTC8M_CLK，8 MHz 部时 ， 可调。
  - RTC8M_D256_CLK RTC8M_CLK 256 ， （RTC8M_CLK/256）。RTC8M_CLK 始 8 MHz 时， 时 31.250 KHz 。
  - RTC_CLK，150 KHz 部 时 ， 可调。
- 时
  - APLL_CLK，16 ~ 128 MHz 部 Audio PLL 时

3.2.3 CPU 时钟

3-2 ，CPU_CLK CPU 时 ， ， 可 240 MHz。 时，CPU 超 ， 。

CPU_CLK RTC_CNTL_SOC_CLK_SEL 时 ， PLL_CLK，APLL_CLK，RTC8M_CLK，XTL_CLK CPU_CLK 时 。 参 表 3-2 表 3-3。

表 3-2. CPU_CLK 源

RTC_CNTL_SOC_CLK_SEL	时
0	XTL_CLK
1	PLL_CLK
2	RTC8M_CLK
3	APLL_CLK

表 3-3. CPU_CLK 源

时	*SEL_0	*SEL_1	CPU 时
XTL_CLK	0	-	CPU_CLK = XTL_CLK / (APB_CTRL_PRE_DIV_CNT+1) APB_CTL_PRE_DIV_CNT 认 0， 0 ~ 1023
PLL_CLK (320 MHz)	1	0	CPU_CLK = PLL_CLK / 4 CPU_CLK 80 MHz
PLL_CLK (320 MHz)	1	1	CPU_CLK = PLL_CLK / 2 CPU_CLK 160 MHz
PLL_CLK (480 MHz)	1	2	CPU_CLK = PLL_CLK / 2 CPU_CLK 240 MHz
RTC8M_CLK	2	-	CPU_CLK = RTC8M_CLK / (APB_CTRL_PRE_DIV_CNT+1) APB_CTL_PRE_DIV_CNT 认 0， 0 ~ 1023
APLL_CLK	3	0	CPU_CLK = APLL_CLK / 4
APLL_CLK	3	1	CPU_CLK = APLL_CLK / 2

*SEL_0： RTC_CNTL_SOC_CLK_SEL
*SEL_1： CPU_CPUPERIOD_SEL

3.2.4 外设时钟

设 时 包括 APB_CLK，REF_TICK，LEDC_SCLK，APLL_CLK PLL_D2_CLK。
表 3-4 个 设 时 。

表 3-4. 外设时钟用法

设	APB_CLK	REF_TICK	LEDC_SCLK	APLL_CLK	PLL_D2_CLK
EMAC	Y	N	N	Y	N
TIMG	Y	N	N	N	N
I2S	Y	N	N	Y	Y
UART	Y	Y	N	N	N
RMT	Y	Y	N	N	N
LED PWM	Y	Y	Y	N	N
PWM	Y	N	N	N	N
I2C	Y	N	N	N	N
SPI	Y	N	N	N	N
PCNT	Y	N	N	N	N
Efuse Controller	Y	N	N	N	N
SDIO Slave	Y	N	N	N	N
SDMMC	Y	N	N	N	N

3.2.4.1 APB_CLK 源

表 3-5 ，APB_CLK CPU_CLK 产 ， CPU_CLK ：

表 3-5. APB_CLK 源

CPU_CLK	APB_CLK
PLL_CLK	80 MHz
APLL_CLK	CPU_CLK / 2
XTAL_CLK	CPU_CLK
RTC8M_CLK	CPU_CLK

3.2.4.2 REF_TICK 源

REF_TICK APB_CLK 产 , APB_CLK CPU_CLK 。 , 可 保 REF_TICK APB_CLK 时 不变。 表 3-6 :

表 3-6. REF_TICK 源

CPU_CLK & APB_CLK	时
PLL_CLK	APB_CTRL_PLL_TICK_NUM
XTAL_CLK	APB_CTRL_XTAL_TICK_NUM
APLL_CLK	APB_CTRL_APLL_TICK_NUM
RTC8M_CLK	APB_CTRL_CK8M_TICK_NUM

3.2.4.3 LEDC_SCLK 源

LEDC_SCLK 时 LEDC_APB_CLK_SEL , 表 3-7 。

表 3-7. LEDC_SCLK 源

LEDC_APB_CLK_SEL	LEDC_SCLK
0	RTC8M_CLK
1	APB_CLK

3.2.4.4 APLL_SCLK 源

APLL_CLK 部 PLL_CLK, APLL 。

3.2.4.5 PLL_D2_CLK 源

PLL_D2_CLK PLL_CLK 时 。

3.2.4.6 时钟源注意事项

设一般 PLL_CLK 时 。 变化 设 改 才 。 REF_TICK 设 时 , 不 改 设 可 。 参 表 3-4。

LED PWM RTC8M_CLK 时 , APB_CLK 闭 时 , LED PWM 可 。 所 , 时 (参), 常 设 (APB_CLK 闭), LED PWM 可 RTC8M_CLK 常 。

3.2.5 Wi-Fi BT 时钟

Wi-Fi BT 必 APB_CLK 时 PLL_CLK 才 。 Wi-Fi BT 时 时，才 时 闭 PLL_CLK。

LOW_POWER_CLK RTC_CLK、SLOW_CLK、RTC8M_CLK XTL_CLK， Wi-Fi BT 。

3.2.6 RTC 时钟

SLOW_CLK FAST_CLK 时 时 。RTC 时 闭 。

SLOW_CLK RTC_CLK，XTL32K_CLK RTC8M_D256_CLK， Power Management 。

FAST_CLK XTL_CLK 时 RTC8M_CLK， On-chip Sensor 。

3.2.7 音频 PLL

时 可 、 并 时 。时 时 可 ，并 不 时 。

本，ESP32 I2S 设 PLL。APLL 时 I2S 时 ， 参 I2S。

Audio PLL :

$$f_{\text{out}} = \frac{f_{\text{xtal}}(\text{sdm2}+\frac{\text{sdm1}}{2^8}+\frac{\text{sdm0}}{2^{16}}+4)}{2(odiv+2)}$$

中，

- f_{xtal}: ， 常 40 MHz
- sdm0: 可 参 0 ~ 255
- sdm1: 可 参 0 ~ 255
- sdm2: 可 参 0 ~ 63
- odiv: 可 参 0 ~ 31
- 350 MHz ~ 500 MHz

  $$350MHz < f_{\text{xtal}}(\text{sdm2}+\frac{\text{sdm1}}{2^8}+\frac{\text{sdm0}}{2^{16}}+4) < 500MHz$$

: sdm1 sdm0 ESP32 revision0 版本中不 。 ESP32 版本 ，可 《ESP32 Bug 》中查 。

Audio PLL 可 RTC_CNTL_PLLA_FORCE_PU ，RTC_CNTL_PLLA_FORCE_PD 闭， 闭 。 RTC_CNTL_PLLA_FORCE_PU RTC_CNTL_PLLA_FORCE_PD 时 0 时 ，PLL ， 时 闭，被 时 。

4 IO_MUX 和 GPIO 交换矩阵 (GPIO, IO_MUX)

4.1 概述

ESP32 34 个 GPIO pad。 个 pad 可 一个 IO， 一个 部 设 。IO_MUX、RTC IO_MUX GPIO 设 GPIO pad。 IO 。

注意：这 34 个物理 GPIO pad 的序列号为：0-19, 21-23, 25-27, 32-39。其中 GPIO 34-39 仅用作输入管脚，其他的既可以作为输入又可以作为输出管脚。

本 pad（ ：FUN_SEL、IE、OE、WPU、WDU ） 162 个 设 176 个 设 （ ：SIG_IN_SEL、SIG_OUT_SEL、IE、OE ） 设 / （ ：IE、OE ） RTC IO_MUX 间 。

图 4-1. IO_MUX、RTC IO_MUX 和 GPIO 交换矩阵结构框图

1. IO_MUX 中 个 GPIO pad 一 。 个 pad 可 GPIO （ GPIO ） （ GPIO ， 、SDIO、SPI、JTAG、UART GPIO 。 IO_MUX 。）

   4.10 GPIO pad IO_MUX 。

2. GPIO 设 pad 间 。
   - 方 ：162 个 设 可 一个 GPIO pad 。
   - 方 ： 个 GPIO pad 可 176 个 设 中 一个。

   4.9 GPIO 设 。

3. RTC IO_MUX GPIO pad 。 部 GPIO pad 。

   4.11 RTC IO_MUX 。

4.2 通过 GPIO 交换矩阵的外设输入

4.2.1 概述

GPIO 设，GPIO 34 个 GPIO（0-19，21-23，25-27，32-39）中 设（0-18，23-36，39-58，61-90，95-124，140-155，164-181，190-195，198-206）。

IO_MUX GPIO pad 中。IO_MUX 必 设 pad GPIO。GPIO pad 可 GPIO GPIO 设。

4.2.2 功能描述

4-2 GPIO 设。

图 4-2. 通过 IO_MUX、GPIO 交换矩阵的外设输入

把 个 设 Y 绑 个 GPIO pad X：

1. GPIO 中 设 Y GPIO_FUNCy_IN_SEL_CFG：
   - 设 GPIO_FUNCy_IN_SEL GPIO pad X。GPIO pad。
2. GPIO 中 GPIO pad X GPIO_FUNCx_OUT_SEL_CFG、GPIO_ENABLE_DATA[x]：
   - 始 GPIO_ENABLE_DATA[x]，GPIO_FUNCx_OUT_SEL_CFG GPIO_FUNCx_OEN_SEL 1。
   - GPIO_ENABLE_DATA[x] GPIO_ENABLE_REG (GPIOs 0-31) GPIO_ENABLE1_REG (GPIOs 32-39) 中，可 闭 GPIO pad。
3. IO_MUX GPIO。GPIO pad X IO_MUX_x_REG：
   - 设 (MCU_SEL) GPIO X IO_MUX（ Function 2， 2）。
   - FUN_IE。
   - FUN_WPU FUN_WPD，闭 部上 /。

说明：

- 一个 pad 上可 时绑 个 部 input_signals。
- GPIO_FUNC*y*_IN_INV_SEL 可 把 。
- 绑 一个 pad 可 设 。 GPIO_FUNC*y*_IN_SEL 不 一个 GPIO ：
  - GPIO_FUNC*y*_IN_SEL 0x30 时，input_signal_*x* 始 0。
  - GPIO_FUNC*y*_IN_SEL 0x38 时，input_signal_*x* 始 1。

，把 RMT 设 0 RMT_SIG_IN0_IDX（ 83）绑 GPIO15，按 步操（ GPIO15 MTDO ）：

1. GPIO_FUNC83_IN_SEL_CFG GPIO_FUNC83_IN_SEL 设 15。
2. ， GPIO_FUNC15_OUT_SEL_CFG_REG 中 GPIO_FUNC15_OEN_SEL 。
3. GPIO_ENABLE_REG bit 15（GPIO_ENABLE_DATA[15] ）。
4. IO_MUX_GPIO15 MCU_SEL 2 (GPIO function)， 时 FUN_IE（ ）。

4.2.3 简单 GPIO 输入

GPIO_IN_REG/GPIO_IN1_REG 一个 GPIO pad 。

GPIO pin 可 时 一个 设 GPIO 。 pad *X* IO_MUX_*x*_REG FUN_IE ， 4.2.2 。

4.3 通过 GPIO 交换矩阵的外设输出

4.3.1 概述

GPIO 设 ， GPIO 0-18，23-37，61-121，140-215，224-228 设 28 个 GPIO (0-19, 21-23, 25-27, 32-33)。

设 GPIO ， IO_MUX。IO_MUX 必 设 pad GPIO 。 GPIO pad。

> **说明：**
> 224-228 设 ，可 一个 GPIO ， 一个 GPIO 。

4.3.2 功能描述

4-3 176 个 中 一个 GPIO IO_MUX 个 pad。

一 pad GPIO ，设 GPIO GPIO_FUNC*x*_OUT_SEL 设 256 (0x100)。

4.4 IO_MUX 的直接 I/O 功能

4.4.1 概述

、SDIO、SPI、JTAG、UART GPIO 。
IO_MUX 。

比 GPIO ， 个 GPIO pad IO_MUX ， 可
。

4.4.2 功能描述

设 I/O GPIO 必 个 ：

1. GPIO pad IO_MUX 必 设 pad ， 4.10 pad 。
2. ，必 SIG_IN_SEL ， 设。

4.5 RTC IO_MUX 的低功耗和模拟 I/O 功能

4.5.1 概述

18 个 GPIO (RTC) ， ESP32 RTC 。 不 IO_MUX GPIO ，而 RTC_MUX I/O RTC 。

被 RTC GPIO ， 时 Deep-sleep 保
时可 Deep-sleep 中 。
4.11 RTC_MUX 。

4.5.2 功能描述

个 pad RTC RTC_GPIO_PIN*x* 中 RTC_IO_TOUCH_PAD*x*_TO_GPIO 。
认 1， IO_MUX ， 。

RTC_IO_TOUCH_PAD*x*_TO_GPIO ， RTC 。 ，
RTC_GPIO_PIN*x* I/O，pad 可 。 4.11 RTC 。

表 4.11 GPIO pad RTC 。 RTC_IO_PIN*x*
RTC GPIO ，不 GPIO pad 。

4.6 Light-sleep 模式管脚功能

ESP32 Light-sleep 时 可 不 。 一 GPIO pad IO_MUX 中 SLP_SEL
1， Light-sleep 一 不 pad。

表 4-1. IO_MUX Light-sleep 管脚功能寄存器

IO_MUX	常 SLP_SEL = 0	Light-sleep 并 SLP_SEL = 1
Output Drive Strength	FUN_DRV	MCU_DRV
Pull-up Resistor	FUN_WPU	MCU_WPU
Pull-down Resistor	FUN_WPD	MCU_WPD
Output Enable	(From GPIO Matrix _OEN field)	MCU_OE

SLP_SEL 0， 常 Light-sleep ， 。

4.7 Pad Hold 特性

个 IO pad（包括 RTC pad） 单 hold ， RTC 。pad hold 被 上 ，pad 上 hold 一 被 保 ， 部 变化 已 IO_MUX GPIO ， 不 已 变 pad 。 超时 时 Deep-sleep 时 pad 不被 已 变， 把 hold 上。

说明：

- pad 新 ， 保 pad ， 把 REG_DG_PAD_FORCE_UNHOLD 设 0。 RTC pad 新 ，pad ， RTC_CNTL_HOLD_FORCE_REG 中 Hold Unhold pad 。
- 被 ， 闭 Hold ， REG_DG_PAD_FORCE_UNHOLD 设 1。 保 pad ，可把 RTC_CNTL_HOLD_FORCE_REG 中 设 1。

4.8 I/O Pad 供电

IO pad 4-4 4-5 。

图 4-4. ESP32 I/O Pad 供电源（QFN 6*6，顶视图）

图 4-5. ESP32 I/O Pad 供电源（QFN 5*5，顶视图）

- pad RTC pad， ， 可 常 IO pad ， 见 4.11。
- pad IO 。
- pad 可 VDD_SDIO 部 可 部 （ 见 ）。

4.8.1 VDD_SDIO 电源域

VDD_SDIO 可 ， VDD_SDIO 可 部 部 。 部 ，必 VDD3P3_RTC 。

部不 ， 部 VDD_SDIO 。VDD_SDIO 可 1.8V VDD3P3_RTC ， MTDI pad 时 时 1.8V， 时 VDD3P3_RTC 。eFuse bit 上 可 VDD_SDIO 认 。 ， 启 可 改变 VDD_SDIO 。

4.9 外设信号列表

表 4-2 GPIO 设 / 。

表 4-2. GPIO 交换矩阵外设信号

Signal	Input Signal	Output Signal	Direct I/O in IO_MUX
0	SPICLK_in	SPICLK_out	YES
1	SPIQ_in	SPIQ_out	YES
2	SPID_in	SPID_out	YES

Signal	Input Signal	Output Signal	Direct I/O in IO_MUX
3	SPIHD_in	SPIHD_out	YES
4	SPIWP_in	SPIWP_out	YES
5	SPICS0_in	SPICS0_out	YES
6	SPICS1_in	SPICS1_out	-
7	SPICS2_in	SPICS2_out	-
8	HSPICLK_in	HSPICLK_out	YES
9	HSPIQ_in	HSPIQ_out	YES
10	HSPID_in	HSPID_out	YES
11	HSPICS0_in	HSPICS0_out	YES
12	HSPIHD_in	HSPIHD_out	YES
13	HSPIWP_in	HSPIWP_out	YES
14	U0RXD_in	U0TXD_out	YES
15	U0CTS_in	U0RTS_out	YES
16	U0DSR_in	U0DTR_out	-
17	U1RXD_in	U1TXD_out	YES
18	U1CTS_in	U1RTS_out	YES
23	I2S0O_BCK_in	I2S0O_BCK_out	-
24	I2S1O_BCK_in	I2S1O_BCK_out	-
25	I2S0O_WS_in	I2S0O_WS_out	-
26	I2S1O_WS_in	I2S1O_WS_out	-
27	I2S0I_BCK_in	I2S0I_BCK_out	-
28	I2S0I_WS_in	I2S0I_WS_out	-
29	I2CEXT0_SCL_in	I2CEXT0_SCL_out	-
30	I2CEXT0_SDA_in	I2CEXT0_SDA_out	-
31	pwm0_sync0_in	sdio_tohost_int_out	-
32	pwm0_sync1_in	pwm0_out0a	-
33	pwm0_sync2_in	pwm0_out0b	-
34	pwm0_f0_in	pwm0_out1a	-
35	pwm0_f1_in	pwm0_out1b	-
36	pwm0_f2_in	pwm0_out2a	-
37	-	pwm0_out2b	-
39	pcnt_sig_ch0_in0	-	-
40	pcnt_sig_ch1_in0	-	-
41	pcnt_ctrl_ch0_in0	-	-
42	pcnt_ctrl_ch1_in0	-	-
43	pcnt_sig_ch0_in1	-	-
44	pcnt_sig_ch1_in1	-	-
45	pcnt_ctrl_ch0_in1	-	-
46	pcnt_ctrl_ch1_in1	-	-
47	pcnt_sig_ch0_in2	-	-
48	pcnt_sig_ch1_in2	-	-
49	pcnt_ctrl_ch0_in2	-	-
50	pcnt_ctrl_ch1_in2	-	-

Signal	Input Signal	Output Signal	Direct I/O in IO_MUX
51	pcnt_sig_ch0_in3	-	-
52	pcnt_sig_ch1_in3	-	-
53	pcnt_ctrl_ch0_in3	-	-
54	pcnt_ctrl_ch1_in3	-	-
55	pcnt_sig_ch0_in4	-	-
56	pcnt_sig_ch1_in4	-	-
57	pcnt_ctrl_ch0_in4	-	-
58	pcnt_ctrl_ch1_in4	-	-
61	HSPICS1_in	HSPICS1_out	-
62	HSPICS2_in	HSPICS2_out	-
63	VSPICLK_in	VSPICLK_out_mux	YES
64	VSPIQ_in	VSPIQ_out	YES
65	VSPID_in	VSPID_out	YES
66	VSPIHD_in	VSPIHD_out	YES
67	VSPIWP_in	VSPIWP_out	YES
68	VSPICS0_in	VSPICS0_out	YES
69	VSPICS1_in	VSPICS1_out	-
70	VSPICS2_in	VSPICS2_out	-
71	pcnt_sig_ch0_in5	ledc_hs_sig_out0	-
72	pcnt_sig_ch1_in5	ledc_hs_sig_out1	-
73	pcnt_ctrl_ch0_in5	ledc_hs_sig_out2	-
74	pcnt_ctrl_ch1_in5	ledc_hs_sig_out3	-
75	pcnt_sig_ch0_in6	ledc_hs_sig_out4	-
76	pcnt_sig_ch1_in6	ledc_hs_sig_out5	-
77	pcnt_ctrl_ch0_in6	ledc_hs_sig_out6	-
78	pcnt_ctrl_ch1_in6	ledc_hs_sig_out7	-
79	pcnt_sig_ch0_in7	ledc_ls_sig_out0	-
80	pcnt_sig_ch1_in7	ledc_ls_sig_out1	-
81	pcnt_ctrl_ch0_in7	ledc_ls_sig_out2	-
82	pcnt_ctrl_ch1_in7	ledc_ls_sig_out3	-
83	rmt_sig_in0	ledc_ls_sig_out4	-
84	rmt_sig_in1	ledc_ls_sig_out5	-
85	rmt_sig_in2	ledc_ls_sig_out6	-
86	rmt_sig_in3	ledc_ls_sig_out7	-
87	rmt_sig_in4	rmt_sig_out0	-
88	rmt_sig_in5	rmt_sig_out1	-
89	rmt_sig_in6	rmt_sig_out2	-
90	rmt_sig_in7	rmt_sig_out3	-
91	-	rmt_sig_out4	-
92	-	rmt_sig_out5	-
93	-	rmt_sig_out6	-
94	-	rmt_sig_out7	-
95	I2CEXT1_SCL_in	I2CEXT1_SCL_out	-

Signal	Input Signal	Output Signal	Direct I/O in IO_MUX
96	I2CEXT1_SDA_in	I2CEXT1_SDA_out	-
97	host_card_detect_n_1	host_ccmd_od_pullup_en_n	-
98	host_card_detect_n_2	host_rst_n_1	-
99	host_card_write_prt_1	host_rst_n_2	-
100	host_card_write_prt_2	gpio_sd0_out	-
101	host_card_int_n_1	gpio_sd1_out	-
102	host_card_int_n_2	gpio_sd2_out	-
103	pwm1_sync0_in	gpio_sd3_out	-
104	pwm1_sync1_in	gpio_sd4_out	-
105	pwm1_sync2_in	gpio_sd5_out	-
106	pwm1_f0_in	gpio_sd6_out	-
107	pwm1_f1_in	gpio_sd7_out	-
108	pwm1_f2_in	pwm1_out0a	-
109	pwm0_cap0_in	pwm1_out0b	-
110	pwm0_cap1_in	pwm1_out1a	-
111	pwm0_cap2_in	pwm1_out1b	-
112	pwm1_cap0_in	pwm1_out2a	-
113	pwm1_cap1_in	pwm1_out2b	-
114	pwm1_cap2_in	pwm2_out1h	-
115	pwm2_flta	pwm2_out1l	-
116	pwm2_fltb	pwm2_out2h	-
117	pwm2_cap1_in	pwm2_out2l	-
118	pwm2_cap2_in	pwm2_out3h	-
119	pwm2_cap3_in	pwm2_out3l	-
120	pwm3_flta	pwm2_out4h	-
121	pwm3_fltb	pwm2_out4l	-
122	pwm3_cap1_in	-	-
123	pwm3_cap2_in	-	-
124	pwm3_cap3_in	-	-
140	I2S0I_DATA_in0	I2S0O_DATA_out0	-
141	I2S0I_DATA_in1	I2S0O_DATA_out1	-
142	I2S0I_DATA_in2	I2S0O_DATA_out2	-
143	I2S0I_DATA_in3	I2S0O_DATA_out3	-
144	I2S0I_DATA_in4	I2S0O_DATA_out4	-
145	I2S0I_DATA_in5	I2S0O_DATA_out5	-
146	I2S0I_DATA_in6	I2S0O_DATA_out6	-
147	I2S0I_DATA_in7	I2S0O_DATA_out7	-
148	I2S0I_DATA_in8	I2S0O_DATA_out8	-
149	I2S0I_DATA_in9	I2S0O_DATA_out9	-
150	I2S0I_DATA_in10	I2S0O_DATA_out10	-
151	I2S0I_DATA_in11	I2S0O_DATA_out11	-
152	I2S0I_DATA_in12	I2S0O_DATA_out12	-
153	I2S0I_DATA_in13	I2S0O_DATA_out13	-

Signal	Input Signal	Output Signal	Direct I/O in IO_MUX
154	I2S0I_DATA_in14	I2S0O_DATA_out14	-
155	I2S0I_DATA_in15	I2S0O_DATA_out15	-
156	-	I2S0O_DATA_out16	-
157	-	I2S0O_DATA_out17	-
158	-	I2S0O_DATA_out18	-
159	-	I2S0O_DATA_out19	-
160	-	I2S0O_DATA_out20	-
161	-	I2S0O_DATA_out21	-
162	-	I2S0O_DATA_out22	-
163	-	I2S0O_DATA_out23	-
164	I2S1I_BCK_in	I2S1I_BCK_out	-
165	I2S1I_WS_in	I2S1I_WS_out	-
166	I2S1I_DATA_in0	I2S1O_DATA_out0	-
167	I2S1I_DATA_in1	I2S1O_DATA_out1	-
168	I2S1I_DATA_in2	I2S1O_DATA_out2	-
169	I2S1I_DATA_in3	I2S1O_DATA_out3	-
170	I2S1I_DATA_in4	I2S1O_DATA_out4	-
171	I2S1I_DATA_in5	I2S1O_DATA_out5	-
172	I2S1I_DATA_in6	I2S1O_DATA_out6	-
173	I2S1I_DATA_in7	I2S1O_DATA_out7	-
174	I2S1I_DATA_in8	I2S1O_DATA_out8	-
175	I2S1I_DATA_in9	I2S1O_DATA_out9	-
176	I2S1I_DATA_in10	I2S1O_DATA_out10	-
177	I2S1I_DATA_in11	I2S1O_DATA_out11	-
178	I2S1I_DATA_in12	I2S1O_DATA_out12	-
179	I2S1I_DATA_in13	I2S1O_DATA_out13	-
180	I2S1I_DATA_in14	I2S1O_DATA_out14	-
181	I2S1I_DATA_in15	I2S1O_DATA_out15	-
182	-	I2S1O_DATA_out16	-
183	-	I2S1O_DATA_out17	-
184	-	I2S1O_DATA_out18	-
185	-	I2S1O_DATA_out19	-
186	-	I2S1O_DATA_out20	-
187	-	I2S1O_DATA_out21	-
188	-	I2S1O_DATA_out22	-
189	-	I2S1O_DATA_out23	-
190	I2S0I_H_SYNC	pwm3_out1h	-
191	I2S0I_V_SYNC	pwm3_out1l	-
192	I2S0I_H_ENABLE	pwm3_out2h	-
193	I2S1I_H_SYNC	pwm3_out2l	-
194	I2S1I_V_SYNC	pwm3_out3h	-
195	I2S1I_H_ENABLE	pwm3_out3l	-
196	-	pwm3_out4h	-

Signal	Input Signal	Output Signal	Direct I/O in IO_MUX
197	-	pwm3_out4l	-
198	U2RXD_in	U2TXD_out	YES
199	U2CTS_in	U2RTS_out	YES
200	emac_mdc_i	emac_mdc_o	-
201	emac_mdi_i	emac_mdo_o	-
202	emac_crs_i	emac_crs_o	-
203	emac_col_i	emac_col_o	-
204	pcmfsync_in	bt_audio0_irq	-
205	pcmclk_in	bt_audio1_irq	-
206	pcmdin	bt_audio2_irq	-
207	-	ble_audio0_irq	-
208	-	ble_audio1_irq	-
209	-	ble_audio2_irq	-
210	-	pcmfsync_out	-
211	-	pcmclk_out	-
212	-	pcmdout	-
213	-	ble_audio_sync0_p	-
214	-	ble_audio_sync1_p	-
215	-	ble_audio_sync2_p	-
224	-	sig_in_func224	-
225	-	sig_in_func225	-
226	-	sig_in_func226	-
227	-	sig_in_func227	-
228	-	sig_in_func228	-

Direct I/O in IO_MUX "YES" 可 IO_MUX pad。 GPIO ， 必 SIG_IN_SEL 。

4.10 IO_MUX Pad 列表

表 4-3 个 I/O pad IO_MUX 。

表 4-3. IO_MUX Pad 列表

GPIO	Pad Name	Function 0	Function 1	Function 2	Function 3	Function 4	Function 5	Reset	Notes
0	GPIO0	GPIO0	CLK_OUT1	GPIO0	-	-	EMAC_TX_CLK	3	R
1	U0TXD	U0TXD	CLK_OUT3	GPIO1	-	-	EMAC_RXD2	3	-
2	GPIO2	GPIO2	HSPIWP	GPIO2	HS2_DATA0	SD_DATA0	-	2	R
3	U0RXD	U0RXD	CLK_OUT2	GPIO3	-	-	-	3	-
4	GPIO4	GPIO4	HSPIHD	GPIO4	HS2_DATA1	SD_DATA1	EMAC_TX_ER	2	R
5	GPIO5	GPIO5	VSPICS0	GPIO5	HS1_DATA6	-	EMAC_RX_CLK	3	-
6	SD_CLK	SD_CLK	SPICLK	GPIO6	HS1_CLK	U1CTS	-	3	-
7	SD_DATA_0	SD_DATA0	SPIQ	GPIO7	HS1_DATA0	U2RTS	-	3	-
8	SD_DATA_1	SD_DATA1	SPID	GPIO8	HS1_DATA1	U2CTS	-	3	-
9	SD_DATA_2	SD_DATA2	SPIHD	GPIO9	HS1_DATA2	U1RXD	-	3	-
10	SD_DATA_3	SD_DATA3	SPIWP	GPIO10	HS1_DATA3	U1TXD	-	3	-
11	SD_CMD	SD_CMD	SPICS0	GPIO11	HS1_CMD	U1RTS	-	3	-
12	MTDI	MTDI	HSPIQ	GPIO12	HS2_DATA2	SD_DATA2	EMAC_TXD3	2	R
13	MTCK	MTCK	HSPID	GPIO13	HS2_DATA3	SD_DATA3	EMAC_RX_ER	2	R

GPIO	Pad Name	Function 0	Function 1	Function 2	Function 3	Function 4	Function 5	Reset	Notes
14	MTMS	MTMS	HSPICLK	GPIO14	HS2_CLK	SD_CLK	EMAC_TXD2	3	R
15	MTDO	MTDO	HSPICS0	GPIO15	HS2_CMD	SD_CMD	EMAC_RXD3	3	R
16	GPIO16	GPIO16	-	GPIO16	HS1_DATA4	U2RXD	EMAC_CLK_OUT	1	-
17	GPIO17	GPIO17	-	GPIO17	HS1_DATA5	U2TXD	EMAC_CLK_180	1	-
18	GPIO18	GPIO18	VSPICLK	GPIO18	HS1_DATA7	-	-	1	-
19	GPIO19	GPIO19	VSPIQ	GPIO19	U0CTS	-	EMAC_TXD0	1	-
21	GPIO21	GPIO21	VSPIHD	GPIO21	-	-	EMAC_TX_EN	1	-
22	GPIO22	GPIO22	VSPIWP	GPIO22	U0RTS	-	EMAC_TXD1	1	-
23	GPIO23	GPIO23	VSPID	GPIO23	HS1_STROBE	-	-	1	-
25	GPIO25	GPIO25	-	GPIO25	-	-	EMAC_RXD0	0	R
26	GPIO26	GPIO26	-	GPIO26	-	-	EMAC_RXD1	0	R
27	GPIO27	GPIO27	-	GPIO27	-	-	EMAC_RX_DV	0	R
32	32K_XP	GPIO32	-	GPIO32	-	-	-	0	R
33	32K_XN	GPIO33	-	GPIO33	-	-	-	0	R
34	VDET_1	GPIO34	-	GPIO34	-	-	-	0	R, I
35	VDET_2	GPIO35	-	GPIO35	-	-	-	0	R, I
36	SENSOR_VP	GPIO36	-	GPIO36	-	-	-	0	R, I
37	SENSOR_CAPP	GPIO37	-	GPIO37	-	-	-	0	R, I
38	SENSOR_CAPN	GPIO38	-	GPIO38	-	-	-	0	R, I
39	SENSOR_VN	GPIO39	-	GPIO39	-	-	-	0	R, I

复位配置

“Reset” 一 个 pad 认 。

- **0** - IE=0（ 闭）
- **1** - IE=1（ ）
- **2** - IE=1, WPD=1（ , ）
- **3** - IE=1, WPU=1（ , ）

说明

- **R** - Pad RTC_MUX RTC / 。
- **I** - Pad GPIO。 Pad 不 备 / 。

参 《ESP32 》 查 表 。

4.11 RTC_MUX 管脚清单

表 4-4 RTC GPIO pad。

表 4-4. RTC_MUX 管脚清单

RTC GPIO Num	GPIO Num	Pad Name	Analog Function			RTC Function	
			0	1	2	Function 0 (FUN_SEL = 0)	Function 1 (FUN_SEL = 3)
0	36	SENSOR_VP	ADC_H	ADC1_CH0	-	RTC_GPIO0	-
1	37	SENSOR_CAPP	ADC_H	ADC1_CH1	-	RTC_GPIO1	-
2	38	SENSOR_CAPN	ADC_H	ADC1_CH2	-	RTC_GPIO2	-
3	39	SENSOR_VN	ADC_H	ADC1_CH3	-	RTC_GPIO3	-
4	34	VDET_1	-	ADC1_CH6	-	RTC_GPIO4	-

RTC GPIO Num	GPIO Num	Pad Name	Analog Function			RTC Function	
			0	1	2	Function 0 (FUN_SEL = 0)	Function 1 (FUN_SEL = 3)
5	35	VDET_2	-	ADC1_CH7	-	RTC_GPIO5	-
6	25	GPIO25	DAC_1	ADC2_CH8	-	RTC_GPIO6	-
7	26	GPIO26	DAC_2	ADC2_CH9	-	RTC_GPIO7	-
8	33	32K_XN	XTAL_32K_N	ADC1_CH5	TOUCH8	RTC_GPIO8	-
9	32	32K_XP	XTAL_32K_P	ADC1_CH4	TOUCH9	RTC_GPIO9	-
10	4	GPIO4	-	ADC2_CH0	TOUCH0	RTC_GPIO10	I2C_SCL*
11	0	GPIO0	-	ADC2_CH1	TOUCH1	RTC_GPIO11	I2C_SDA*
12	2	GPIO2	-	ADC2_CH2	TOUCH2	RTC_GPIO12	I2C_SCL*
13	15	MTDO	-	ADC2_CH3	TOUCH3	RTC_GPIO13	I2C_SDA*
14	13	MTCK	-	ADC2_CH4	TOUCH4	RTC_GPIO14	-
15	12	MTDI	-	ADC2_CH5	TOUCH5	RTC_GPIO15	-
16	14	MTMS	-	ADC2_CH6	TOUCH6	RTC_GPIO16	-
17	27	GPIO27	-	ADC2_CH7	TOUCH7	RTC_GPIO17	-

说明：

RTC I2C , 参 30 超低功耗协处理器 (ULP)：RTC I2C 。

4.12 寄存器列表

GPIO_OUT_REG	GPIO 0-31 output register	0x3FF44004	/写
GPIO_OUT_W1TS_REG	GPIO 0-31 output register_W1TS	0x3FF44008	写
GPIO_OUT_W1TC_REG	GPIO 0-31 output register_W1TC	0x3FF4400C	写
GPIO_OUT1_REG	GPIO 32-39 output register	0x3FF44010	/写
GPIO_OUT1_W1TS_REG	GPIO 32-39 output bit set register	0x3FF44014	写
GPIO_OUT1_W1TC_REG	GPIO 32-39 output bit clear register	0x3FF44018	写
GPIO_ENABLE_REG	GPIO 0-31 output enable register	0x3FF44020	/写
GPIO_ENABLE_W1TS_REG	GPIO 0-31 output enable register_W1TS	0x3FF44024	写
GPIO_ENABLE_W1TC_REG	GPIO 0-31 output enable register_W1TC	0x3FF44028	写
GPIO_ENABLE1_REG	GPIO 32-39 output enable register	0x3FF4402C	/写
GPIO_ENABLE1_W1TS_REG	GPIO 32-39 output enable bit set register	0x3FF44030	写
GPIO_ENABLE1_W1TC_REG	GPIO 32-39 output enable bit clear register	0x3FF44034	写
GPIO_STRAP_REG	Bootstrap pin value register	0x3FF44038	
GPIO_IN_REG	GPIO 0-31 input register	0x3FF4403C	
GPIO_IN1_REG	GPIO 32-39 input register	0x3FF44040	
GPIO_STATUS_REG	GPIO 0-31 interrupt status register	0x3FF44044	/写
GPIO_STATUS_W1TS_REG	GPIO 0-31 interrupt status register_W1TS	0x3FF44048	写
GPIO_STATUS_W1TC_REG	GPIO 0-31 interrupt status register_W1TC	0x3FF4404C	写
GPIO_STATUS1_REG	GPIO 32-39 interrupt status register1	0x3FF44050	/写
GPIO_STATUS1_W1TS_REG	GPIO 32-39 interrupt status bit set register	0x3FF44054	写
GPIO_STATUS1_W1TC_REG	GPIO 32-39 interrupt status bit clear register	0x3FF44058	写

GPIO_ACPU_INT_REG	GPIO 0-31 APP_CPU interrupt status	0x3FF44060	
GPIO_ACPU_NMI_INT_REG	GPIO 0-31 APP_CPU non-maskable interrupt status	0x3FF44064	
GPIO_PCPU_INT_REG	GPIO 0-31 PRO_CPU interrupt status	0x3FF44068	
GPIO_PCPU_NMI_INT_REG	GPIO 0-31 PRO_CPU non-maskable interrupt status	0x3FF4406C	
GPIO_ACPU_INT1_REG	GPIO 32-39 APP_CPU interrupt status	0x3FF44074	
GPIO_ACPU_NMI_INT1_REG	GPIO 32-39 APP_CPU non-maskable interrupt status	0x3FF44078	
GPIO_PCPU_INT1_REG	GPIO 32-39 PRO_CPU interrupt status	0x3FF4407C	
GPIO_PCPU_NMI_INT1_REG	GPIO 32-39 PRO_CPU non-maskable interrupt status	0x3FF44080	
GPIO_PIN0_REG	Configuration for GPIO pin 0	0x3FF44088	/写
GPIO_PIN1_REG	Configuration for GPIO pin 1	0x3FF4408C	/写
GPIO_PIN2_REG	Configuration for GPIO pin 2	0x3FF44090	/写
...	...		
GPIO_PIN38_REG	Configuration for GPIO pin 38	0x3FF44120	/写
GPIO_PIN39_REG	Configuration for GPIO pin 39	0x3FF44124	/写
GPIO_FUNC0_IN_SEL_CFG_REG	Peripheral function 0 input selection register	0x3FF44130	/写
GPIO_FUNC1_IN_SEL_CFG_REG	Peripheral function 1 input selection register	0x3FF44134	/写
...	...		
GPIO_FUNC254_IN_SEL_CFG_REG	Peripheral function 254 input selection register	0x3FF44528	/写
GPIO_FUNC255_IN_SEL_CFG_REG	Peripheral function 255 input selection register	0x3FF4452C	/写
GPIO_FUNC0_OUT_SEL_CFG_REG	Peripheral output selection for GPIO0	0x3FF44530	/写
GPIO_FUNC1_OUT_SEL_CFG_REG	Peripheral output selection for GPIO1	0x3FF44534	/写
...	...		
GPIO_FUNC38_OUT_SEL_CFG_REG	Peripheral output selection for GPIO38	0x3FF445C8	/写
GPIO_FUNC39_OUT_SEL_CFG_REG	Peripheral output selection for GPIO39	0x3FF445CC	/写

IO_MUX_PIN_CTRL	Clock output configuration register	0x3FF49000	/写
IO_MUX_GPIO36_REG	Configuration register for pad GPIO36	0x3FF49004	/写
IO_MUX_GPIO37_REG	Configuration register for pad GPIO37	0x3FF49008	/写
IO_MUX_GPIO38_REG	Configuration register for pad GPIO38	0x3FF4900C	/写
IO_MUX_GPIO39_REG	Configuration register for pad GPIO39	0x3FF49010	/写
IO_MUX_GPIO34_REG	Configuration register for pad GPIO34	0x3FF49014	/写
IO_MUX_GPIO35_REG	Configuration register for pad GPIO35	0x3FF49018	/写
IO_MUX_GPIO32_REG	Configuration register for pad GPIO32	0x3FF4901C	/写
IO_MUX_GPIO33_REG	Configuration register for pad GPIO33	0x3FF49020	/写
IO_MUX_GPIO25_REG	Configuration register for pad GPIO25	0x3FF49024	/写
IO_MUX_GPIO26_REG	Configuration register for pad GPIO26	0x3FF49028	/写
IO_MUX_GPIO27_REG	Configuration register for pad GPIO27	0x3FF4902C	/写
IO_MUX_MTMS_REG	Configuration register for pad MTMS	0x3FF49030	/写

IO_MUX_MTDI_REG	Configuration register for pad MTDI	0x3FF49034	/写
IO_MUX_MTCK_REG	Configuration register for pad MTCK	0x3FF49038	/写
IO_MUX_MTDO_REG	Configuration register for pad MTDO	0x3FF4903C	/写
IO_MUX_GPIO2_REG	Configuration register for pad GPIO2	0x3FF49040	/写
IO_MUX_GPIO0_REG	Configuration register for pad GPIO0	0x3FF49044	/写
IO_MUX_GPIO4_REG	Configuration register for pad GPIO4	0x3FF49048	/写
IO_MUX_GPIO16_REG	Configuration register for pad GPIO16	0x3FF4904C	/写
IO_MUX_GPIO17_REG	Configuration register for pad GPIO17	0x3FF49050	/写
IO_MUX_SD_DATA2_REG	Configuration register for pad SD_DATA2	0x3FF49054	/写
IO_MUX_SD_DATA3_REG	Configuration register for pad SD_DATA3	0x3FF49058	/写
IO_MUX_SD_CMD_REG	Configuration register for pad SD_CMD	0x3FF4905C	/写
IO_MUX_SD_CLK_REG	Configuration register for pad SD_CLK	0x3FF49060	/写
IO_MUX_SD_DATA0_REG	Configuration register for pad SD_DATA0	0x3FF49064	/写
IO_MUX_SD_DATA1_REG	Configuration register for pad SD_DATA1	0x3FF49068	/写
IO_MUX_GPIO5_REG	Configuration register for pad GPIO5	0x3FF4906C	/写
IO_MUX_GPIO18_REG	Configuration register for pad GPIO18	0x3FF49070	/写
IO_MUX_GPIO19_REG	Configuration register for pad GPIO19	0x3FF49074	/写
IO_MUX_GPIO20_REG	Configuration register for pad GPIO20	0x3FF49078	/写
IO_MUX_GPIO21_REG	Configuration register for pad GPIO21	0x3FF4907C	/写
IO_MUX_GPIO22_REG	Configuration register for pad GPIO22	0x3FF49080	/写
IO_MUX_U0RXD_REG	Configuration register for pad U0RXD	0x3FF49084	/写
IO_MUX_U0TXD_REG	Configuration register for pad U0TXD	0x3FF49088	/写
IO_MUX_GPIO23_REG	Configuration register for pad GPIO23	0x3FF4908C	/写
IO_MUX_GPIO24_REG	Configuration register for pad GPIO24	0x3FF49090	/写

GPIO 配置／数据寄存器			
RTCIO_RTC_GPIO_OUT_REG	RTC GPIO output register	0x3FF48400	/写
RTCIO_RTC_GPIO_OUT_W1TS_REG	RTC GPIO output bit set register	0x3FF48404	写
RTCIO_RTC_GPIO_OUT_W1TC_REG	RTC GPIO output bit clear register	0x3FF48408	写
RTCIO_RTC_GPIO_ENABLE_REG	RTC GPIO output enable register	0x3FF4840C	/写
RTCIO_RTC_GPIO_ENABLE_W1TS_REG	RTC GPIO output enable bit set register	0x3FF48410	写
RTCIO_RTC_GPIO_ENABLE_W1TC_REG	RTC GPIO output enable bit clear register	0x3FF48414	写
RTCIO_RTC_GPIO_STATUS_REG	RTC GPIO interrupt status register	0x3FF48418	/写
RTCIO_RTC_GPIO_STATUS_W1TS_REG	RTC GPIO interrupt status bit set register	0x3FF4841C	写
RTCIO_RTC_GPIO_STATUS_W1TC_REG	RTC GPIO interrupt status bit clear register	0x3FF48420	写
RTCIO_RTC_GPIO_IN_REG	RTC GPIO input register	0x3FF48424	
RTCIO_RTC_GPIO_PIN0_REG	RTC configuration for pin 0	0x3FF48428	/写
RTCIO_RTC_GPIO_PIN1_REG	RTC configuration for pin 1	0x3FF4842C	/写
RTCIO_RTC_GPIO_PIN2_REG	RTC configuration for pin 2	0x3FF48430	/写
RTCIO_RTC_GPIO_PIN3_REG	RTC configuration for pin 3	0x3FF48434	/写
RTCIO_RTC_GPIO_PIN4_REG	RTC configuration for pin 4	0x3FF48438	/写
RTCIO_RTC_GPIO_PIN5_REG	RTC configuration for pin 5	0x3FF4843C	/写

RTCIO_RTC_GPIO_PIN6_REG	RTC configuration for pin 6	0x3FF48440	/写
RTCIO_RTC_GPIO_PIN7_REG	RTC configuration for pin 7	0x3FF48444	/写
RTCIO_RTC_GPIO_PIN8_REG	RTC configuration for pin 8	0x3FF48448	/写
RTCIO_RTC_GPIO_PIN9_REG	RTC configuration for pin 9	0x3FF4844C	/写
RTCIO_RTC_GPIO_PIN10_REG	RTC configuration for pin 10	0x3FF48450	/写
RTCIO_RTC_GPIO_PIN11_REG	RTC configuration for pin 11	0x3FF48454	/写
RTCIO_RTC_GPIO_PIN12_REG	RTC configuration for pin 12	0x3FF48458	/写
RTCIO_RTC_GPIO_PIN13_REG	RTC configuration for pin 13	0x3FF4845C	/写
RTCIO_RTC_GPIO_PIN14_REG	RTC configuration for pin 14	0x3FF48460	/写
RTCIO_RTC_GPIO_PIN15_REG	RTC configuration for pin 15	0x3FF48464	/写
RTCIO_RTC_GPIO_PIN16_REG	RTC configuration for pin 16	0x3FF48468	/写
RTCIO_RTC_GPIO_PIN17_REG	RTC configuration for pin 17	0x3FF4846C	/写
RTCIO_DIG_PAD_HOLD_REG	RTC GPIO hold register	0x3FF48474	/写
GPIO RTC 功能配置寄存器			
RTCIO_HALL_SENS_REG	Hall sensor configuration	0x3FF48478	/写
RTCIO_SENSOR_PADS_REG	Sensor pads configuration register	0x3FF4847C	/写
RTCIO_ADC_PAD_REG	ADC configuration register	0x3FF48480	/写
RTCIO_PAD_DAC1_REG	DAC1 configuration register	0x3FF48484	/写
RTCIO_PAD_DAC2_REG	DAC2 configuration register	0x3FF48488	/写
RTCIO_XTAL_32K_PAD_REG	32KHz crystal pads configuration register	0x3FF4848C	/写
RTCIO_TOUCH_CFG_REG	Touch sensor configuration register	0x3FF48490	/写
RTCIO_TOUCH_PAD0_REG	Touch pad configuration register	0x3FF48494	/写
...	...		
RTCIO_TOUCH_PAD9_REG	Touch pad configuration register	0x3FF484B8	/写
RTCIO_EXT_WAKEUP0_REG	External wake up configuration register	0x3FF484BC	/写
RTCIO_XTL_EXT_CTR_REG	Crystal power down enable gpio source	0x3FF484C0	/写
RTCIO_SAR_I2C_IO_REG	RTC I2C pad selection	0x3FF484C4	/写

4.13 寄存器

Register 4.1. GPIO_OUT_REG (0x0004)

31																															0	
x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	Reset

GPIO_OUT_REG GPIO0-31 。(/写)

Register 4.2. GPIO_OUT_W1TS_REG (0x0008)

31																															0	
x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	Reset

GPIO_OUT_W1TS_REG GPIO0-31 。 一 1， GPIO_OUT_REG 中 1。(写)

Register 4.3. GPIO_OUT_W1TC_REG (0x000c)

31																															0	
x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	Reset

GPIO_OUT_W1TC_REG GPIO0-31 。 一 1， GPIO_OUT_REG 中 。(写)

Register 4.4. GPIO_OUT1_REG (0x0010)

(reserved)																								GPIO_OUT_DATA								
31																							8	7							0	
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	x	x	x	x	x	x	x	x	Reset

GPIO_OUT_DATA GPIO32-39 。( )

Register 4.5. GPIO_OUT1_W1TS_REG (0x0014)

(reserved)	GPIO_OUT_DATA	
31 … 8	7 … 0	
0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0	x x x x x x x x	Reset

GPIO_OUT_DATA GPIO32-39 。 一 1， GPIO_OUT1_DATA 中 1。( )

Register 4.6. GPIO_OUT1_W1TC_REG (0x0018)

(reserved)	GPIO_OUT_DATA	
31 … 8	7 … 0	
0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0	x x x x x x x x	Reset

GPIO_OUT_DATA GPIO32-39 。 一 1， GPIO_OUT1_DATA 中 。(写)

Register 4.7. GPIO_ENABLE_REG (0x0020)

31 … 0	
x x x x x x x x x x x x x x x x x x x x x x x x x x x x x x x x	Reset

GPIO_ENABLE_REG GPIO0-31 。(/写)

Register 4.8. GPIO_ENABLE_W1TS_REG (0x0024)

31 … 0	
x x x x x x x x x x x x x x x x x x x x x x x x x x x x x x x x	Reset

GPIO_ENABLE_W1TS_REG GPIO0-31 。 一 1， GPIO_ENABLE 中 1。(写)

Register 4.9. GPIO_ENABLE_W1TC_REG (0x0028)

31																															0	
x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	Reset

GPIO_ENABLE_W1TC_REG GPIO0-31 。 一 1， GPIO_ENABLE 中 。(写)

Register 4.10. GPIO_ENABLE1_REG (0x002c)

(reserved)																									GPIO_ENABLE_DATA								
31																								8	7							0	
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	x	x	x	x	x	x	x	x	Reset

GPIO_ENABLE_DATA GPIO32-39 。(/写)

Register 4.11. GPIO_ENABLE1_W1TS_REG (0x0030)

(reserved)																									GPIO_ENABLE_DATA								
31																								8	7							0	
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	x	x	x	x	x	x	x	x	Reset

GPIO_ENABLE_DATA GPIO32-39 。 一 1， GPIO_ENABLE1 中 1。(写)

Register 4.12. GPIO_ENABLE1_W1TC_REG (0x0034)

(reserved)																									GPIO_ENABLE_DATA								
31																								8	7							0	
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	x	x	x	x	x	x	x	x	Reset

GPIO_ENABLE_DATA GPIO32-39 。 一 1， GPIO_ENABLE1 中 。(写)

Register 4.13. GPIO_STRAP_REG (0x0038)

(reserved)	GPIO_STRAPPING	
31 … 16	15 … 0	
0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0	x x x x x x x x x x x x x x x x	Reset

GPIO_STRAPPING GPIO strapping : boot_sel_chip[5:0] bit5 bit0 别 MTDI, GPIO0, GPIO2, GPIO4, MTDO, GPIO5。

Register 4.14. GPIO_IN_REG (0x003c)

31 … 0	
x x x x x x x x x x x x x x x x x x x x x x x x x x x x x x x x	Reset

GPIO_IN_REG GPIO0-31 。 个 bit 表 pad ，比 ， bit 1， ， bit 0。( )

Register 4.15. GPIO_IN1_REG (0x0040)

(reserved)	GPIO_IN_DATA_NEXT	
31 … 8	7 … 0	
0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0	x x x x x x x x	Reset

GPIO_IN_DATA_NEXT GPIO32-39 。 个 bit 表 pad 。( )

Register 4.16. GPIO_STATUS_REG (0x0044)

31 … 0	
x x x x x x x x x x x x x x x x x x x x x x x x x x x x x x x x	Reset

GPIO_STATUS_REG GPIO0-31 中 。 个 bit 可 个 CPU 中 ， 时 把 GPIO_PIN*n*_REG 0-4 bit GPIO_STATUS_INTERRUPT 1。(/写)

Register 4.17. GPIO_STATUS_W1TS_REG (0x0048)

31																															0	
x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	Reset

GPIO_STATUS_W1TS_REG GPIO0-31 中 。 一 1，GPIO_STATUS_INTERRUPT 中 1。( )

Register 4.18. GPIO_STATUS_W1TC_REG (0x004c)

31																															0	
x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	Reset

GPIO_STATUS_W1TC_REG GPIO0-31 中 。 一 1，GPIO_STATUS_INTERRUPT 中 。(写)

Register 4.19. GPIO_STATUS1_REG (0x0050)

(reserved)																									GPIO_STATUS_INTERRUPT								
31																								8	7							0	
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	x	x	x	x	x	x	x	x	Reset

GPIO_STATUS_INTERRUPT GPIO32-39 中 。(/写)

Register 4.20. GPIO_STATUS1_W1TS_REG (0x0054)

(reserved)																									GPIO_STATUS_INTERRUPT								
31																								8	7							0	
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	x	x	x	x	x	x	x	x	Reset

GPIO_STATUS_INTERRUPT GPIO32-39 中 。 一 1，GPIO_STATUS_INTERRUPT1 中 1。(写)

Register 4.21. GPIO_STATUS1_W1TC_REG (0x0058)

(reserved)	GPIO_STATUS_INTERRUPT	
31 … 8	7 … 0	
0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0	x x x x x x x x	Reset

GPIO_STATUS_INTERRUPT GPIO32-39 中 。 一 1，GPIO_STATUS_INTERRUPT1 中 。（ 写）

Register 4.22. GPIO_ACPU_INT_REG (0x0060)

31 … 0	
x x x x x x x x x x x x x x x x x x x x x x x x x x x x x x x x	Reset

GPIO_ACPU_INT_REG GPIO0-31 APP CPU 中 。（ ）

Register 4.23. GPIO_ACPU_NMI_INT_REG (0x0064)

31 … 0	
x x x x x x x x x x x x x x x x x x x x x x x x x x x x x x x x	Reset

GPIO_ACPU_NMI_INT_REG GPIO0-31 APP CPU 屏蔽中 。（ ）

Register 4.24. GPIO_PCPU_INT_REG (0x0068)

31 … 0	
x x x x x x x x x x x x x x x x x x x x x x x x x x x x x x x x	Reset

GPIO_PCPU_INT_REG GPIO0-31 PRO CPU 中 。（ ）

Register 4.25. GPIO_PCPU_NMI_INT_REG (0x006c)

31 … 0	
x x x x x x x x x x x x x x x x x x x x x x x x x x x x x x x x	Reset

GPIO_PCPU_NMI_INT_REG GPIO0-31 PRO CPU 屏蔽中 。（ ）

Register 4.26. GPIO_ACPU_INT1_REG (0x0074)

(reserved)	GPIO_APPCPU_INT	
31 … 8	7 … 0	
0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0	x x x x x x x x	Reset

GPIO_APPCPU_INT GPIO32-39 APP CPU 中 。( )

Register 4.27. GPIO_ACPU_NMI_INT1_REG (0x0078)

(reserved)	GPIO_APPCPU_NMI_INT	
31 … 8	7 … 0	
0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0	x x x x x x x x	Reset

GPIO_APPCPU_NMI_INT GPIO32-39 APP CPU 屏蔽中 。( )

Register 4.28. GPIO_PCPU_INT1_REG (0x007c)

(reserved)	GPIO_PROCPU_INT	
31 … 8	7 … 0	
0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0	x x x x x x x x	Reset

GPIO_PROCPU_INT GPIO32-39 PRO CPU 中 。( )

Register 4.29. GPIO_PCPU_NMI_INT1_REG (0x0080)

(reserved)	GPIO_PROCPU_NMI_INT	
31 … 8	7 … 0	
0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0	x x x x x x x x	Reset

GPIO_PROCPU_NMI_INT GPIO32-39 PRO CPU 屏蔽中 。( )

Register 4.30. GPIO_PIN*n*_REG (*n*: 0-39) (0x88+0x4n*)**

(reserved)	GPIO_PIN*n*_INT_ENA
31 … 18	17 … 13
0 0 0 0 0 0 0 0 0 0 0 0 0 0	x x x x x

Register 4.32. GPIO_FUNC*n*_OUT_SEL_CFG_REG (*n*: 0-19, 21-23, 25-27, 32-33) (0x530+0x4n*)**

(reserved)	GPIO_FUNC*n*_OEN_INV_SEL	GPIO_FUNC*n*_OEN_SEL	GPIO_FUNC*n*_OUT_INV_SEL	GPIO_FUNC*n*_OUT_SEL	
31 12	11	10	9	8 0	
0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0	x	x	x	x x x x x x x x x	Reset

GPIO_FUNC*n*_OEN_INV_SEL 1: ; 0: 不 。(/写)

GPIO_FUNC*n*_OEN_SEL 1: GPIO_ENABLE_REG bit *n* 中 。0: 设 。(/写)

GPIO_FUNC*n*_OUT_INV_SEL 1: ; 0: 不 。(/写)

GPIO_FUNC*n*_OUT_SEL GPIO *n* 。 *s* (0<=*s*<256) 设 *s* GPIO *n*。 256 GPIO_OUT_REG/GPIO_OUT1_REG GPIO_ENABLE_REG/GPIO_ENABLE1_REG bit *n* 。(/写)

Register 4.33. IO_MUX_PIN_CTRL (0x3FF49000)

(reserved)	PIN_CTRL_CLK3	PIN_CTRL_CLK2	PIN_CTRL_CLK1	
31 12	11 8	7 4	3 0	
0x0	0x0	0x0	0x0	Reset

要将 I2S0 外设时钟输出到：

CLK_OUT1， PIN_CTRL[3:0] = 0x0；

CLK_OUT2， PIN_CTRL[3:0] = 0x0 and PIN_CTRL[7:4] = 0x0；

CLK_OUT3， PIN_CTRL[3:0] = 0x0 and PIN_CTRL[11:8] = 0x0。

要将 I2S1 外设时钟输出到：

CLK_OUT1， PIN_CTRL[3:0] = 0xF；

CLK_OUT2， PIN_CTRL[3:0] = 0xF and PIN_CTRL[7:4] = 0x0；

CLK_OUT3， PIN_CTRL[3:0] = 0xF and PIN_CTRL[11:8] = 0x0。(/写)

说明：

。

CLK_OUT1-3 可 IO_MUX Pad 表中查 。

Register 4.34. IO_MUX_*x*_REG (*x*: GPIO0-GPIO39) (0x10+4x*)**

(reserved)	MCU_SEL	FUN_DRV	FUN_IE	FUN_WPU	FUN_WPD	MCU_DRV	MCU_IE	MCU_WPU	MCU_WPD	SLP_SEL	MCU_OE	
31 … 15	14 … 12	11 … 10	9	8	7	6 … 5	4	3	2	1	0	
0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0	0x0	0x2	0	0	0	0x0	0	0	0	0	0	Reset

MCU_SEL IO_MUX 。0: 0; 1: 1; 。(/写)

FUN_DRV , , 。 GPIO34-39, FUN_DRV 0。
参 《ESP32 中》 “ ” 8 。(/写)

FUN_IE pad 。1: ; 0: 闭。(/写)

FUN_WPU pad 。1: 部 ; 0: 部 闭。 GPIO34-39 可 , 不 备 / , FUN_WPU 0。(/写)

FUN_WPD pad 。1: 部 ; 0: 部 闭。 GPIO34-39 可 , 不 备 / , FUN_WPD 0。(/写)

MCU_DRV pad , , 。(/写)

MCU_IE pad 。1: ; 0: 闭。(/写)

MCU_WPU pad 。1: 部 ; 0: 部 闭。(/写)

MCU_WPD pad 。1: 部 ; 0: 部 闭。(/写)

SLP_SEL pad 。 1 。(/写)

MCU_OE pad 。1: ; 2: 闭。(/写)

Register 4.35. RTCIO_RTC_GPIO_OUT_REG (0x0000)

RTCIO_RTC_GPIO_OUT_DATA	(reserved)	
31 … 14	13 … 0	
x x x x x x x x x x x x x x x x	0 0 0 0 0 0 0 0 0 0 0 0 0 0	Reset

RTCIO_RTC_GPIO_OUT_DATA GPIO0-17 。Bit14 GPIO[0], bit15 GPIO[1], 。(/写)

Register 4.36. RTCIO_RTC_GPIO_OUT_W1TS_REG (0x0004)

RTCIO_RTC_GPIO_OUT_DATA_W1TS	(reserved)	
31 … 14	13 … 0	
x x x x x x x x x x x x x x x x x x	0 0 0 0 0 0 0 0 0 0 0 0 0 0	Reset

RTCIO_RTC_GPIO_OUT_DATA_W1TS GPIO0-17 设 。 一 1， RTCIO_RTC_GPIO_OUT 中 1。(写)

Register 4.37. RTCIO_RTC_GPIO_OUT_W1TC_REG (0x0008)

RTCIO_RTC_GPIO_OUT_DATA_W1TC	(reserved)	
31 … 14	13 … 0	
x x x x x x x x x x x x x x x x x x	0 0 0 0 0 0 0 0 0 0 0 0 0 0	Reset

RTCIO_RTC_GPIO_OUT_DATA_W1TC GPIO0-17 。 一 1， RTCIO_RTC_GPIO_OUT 中 。(写)

Register 4.38. RTCIO_RTC_GPIO_ENABLE_REG (0x000C)

RTCIO_RTC_GPIO_ENABLE	(reserved)	
31 … 14	13 … 0	
x x x x x x x x x x x x x x x x x x	0 0 0 0 0 0 0 0 0 0 0 0 0 0	Reset

RTCIO_RTC_GPIO_ENABLE GPIO0-17 。Bit14 GPIO[0]，bit15 GPIO[1]， 。 1 表 GPIO 。(/写)

Register 4.39. RTCIO_RTC_GPIO_ENABLE_W1TS_REG (0x0010)

RTCIO_RTC_GPIO_ENABLE_W1TS	(reserved)	
31 … 14	13 … 0	
x x x x x x x x x x x x x x x x	0 0 0 0 0 0 0 0 0 0 0 0 0 0	Reset

RTCIO_RTC_GPIO_ENABLE_W1TS GPIO0-17 设 。 一 1， RTCIO_RTC_GPIO_ENABLE 中 1。(写)

Register 4.40. RTCIO_RTC_GPIO_ENABLE_W1TC_REG (0x0014)

RTCIO_RTC_GPIO_ENABLE_W1TC	(reserved)	
31 … 14	13 … 0	
x x x x x x x x x x x x x x x x	0 0 0 0 0 0 0 0 0 0 0 0 0 0	Reset

RTCIO_RTC_GPIO_ENABLE_W1TC GPIO0-17 。 一 1， RTCIO_RTC_GPIO_ENABLE 中 。(写)

Register 4.41. RTCIO_RTC_GPIO_STATUS_REG (0x0018)

RTCIO_RTC_GPIO_STATUS_INT	(reserved)	
31 … 14	13 … 0	
x x x x x x x x x x x x x x x x	0 0 0 0 0 0 0 0 0 0 0 0 0 0	Reset

RTCIO_RTC_GPIO_STATUS_INT GPIO0-17 中 。Bit14 GPIO[0]，bit15 GPIO[1]， 。 时 RTCIO_RTC_GPIO_PIN*n*_REG RTCIO_RTC_GPIO_PIN*n*_INT_TYPE 中 ，1 表 中 ，0 表 中 。(/写)

Register 4.42. RTCIO_RTC_GPIO_STATUS_W1TS_REG (0x001C)

RTCIO_RTC_GPIO_STATUS_INT_W1TS	(reserved)	
31 … 14	13 … 0	
x x x x x x x x x x x x x x x x x x	0 0 0 0 0 0 0 0 0 0 0 0 0 0	Reset

RTCIO_RTC_GPIO_STATUS_INT_W1TS GPIO0-17 中 设 。 一 1， RTCIO_RTC_GPIO_STATUS_INT 中 1。(写)

Register 4.43. RTCIO_RTC_GPIO_STATUS_W1TC_REG (0x0020)

RTCIO_RTC_GPIO_STATUS_INT_W1TC	(reserved)	
31 … 14	13 … 0	
x x x x x x x x x x x x x x x x x x	0 0 0 0 0 0 0 0 0 0 0 0 0 0	Reset

RTCIO_RTC_GPIO_STATUS_INT_W1TC GPIO0-17 中 。 一 1， RTCIO_RTC_GPIO_STATUS_INT 中 。(写)

Register 4.44. RTCIO_RTC_GPIO_IN_REG (0x0024)

RTCIO_RTC_GPIO_IN_NEXT	(reserved)	
31 … 14	13 … 0	
x x x x x x x x x x x x x x x x x x	0 0 0 0 0 0 0 0 0 0 0 0 0 0	Reset

RTCIO_RTC_GPIO_IN_NEXT GPIO0-17 。Bit14 GPIO[0]，bit15 GPIO[1]， 。 个 bit 表 pad ，比 ， bit 1， ， bit 0。( )

Register 4.45. RTCIO_RTC_GPIO_PIN*n*_REG (*n*: 0-17) (28+4n*)**

(reserved)	RTCIO_RTC_GPIO_PIN*n*_WAKEUP_ENABLE	RTCIO_RTC_GPIO_PIN*n*_INT_TYPE	(reserved)	RTCIO_RTC_GPIO_PIN*n*_PAD_DRIVER	(reserved)	
31 … 11	10	9 … 7	6 … 3	2	1 … 0	
0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0	x	x x x	0 0 0 0	x	0 0	Reset

RTCIO_RTC_GPIO_PIN*n*_WAKEUP_ENABLE GPIO 。 Light-sleep ESP32 。(/写)

RTCIO_RTC_GPIO_PIN*n*_INT_TYPE GPIO 中 。(/写)

0: GPIO 中 闭;
1: 上升 ;
2: ;
3: 一 ;
4: ;
5: 。

RTCIO_RTC_GPIO_PIN*n*_PAD_DRIVER Pad 。0: 常 ; 1: 漂 。(/写)

Register 4.46. RTCIO_DIG_PAD_HOLD_REG (0x0074)

31 … 0	
0	Reset

RTCIO_DIG_PAD_HOLD_REG 一个 pad hold 。0: 常操 ; 1: hold 。(/写)

Bit[0]	pad U0RTD Hold
Bit[1]	pad U0TXD Hold
Bit[2]	pad SD_CLK Hold
Bit[3]	pad SD_DATA0 Hold
Bit[4]	pad SD_DATA1 Hold
Bit[5]	pad SD_DATA2 Hold
Bit[6]	pad SD_DATA3 Hold
Bit[7]	pad SD_CMD Hold
Bit[8]	pad GPIO5 Hold
Bit[9]	pad GPIO16 Hold
Bit[10]	pad GPIO17 Hold
Bit[11]	pad GPIO18 Hold
Bit[12]	pad GPIO19 Hold
Bit[13]	pad GPIO20 Hold
Bit[14]	pad GPIO21 Hold
Bit[15]	pad GPIO22 Hold
Bit[16]	pad GPIO23 Hold

Register 4.47. RTCIO_HALL_SENS_REG (0x0078)

RTCIO_HALL_XPD_HALL	RTCIO_HALL_PHASE	(reserved)	
31	30	29 … 0	
0	0	0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0	Reset

RTCIO_HALL_XPD_HALL 上 , VP VN。(/写)

RTCIO_HALL_PHASE 。(/写)

Register 4.48. RTCIO_SENSOR_PADS_REG (0x007C)

RTCIO_SENSOR_SENSE1_HOLD	RTCIO_SENSOR_SENSE2_HOLD	RTCIO_SENSOR_SENSE3_HOLD	RTCIO_SENSOR_SENSE4_HOLD	RTCIO_SENSOR_SENSE1_MUX_SEL	RTCIO_SENSOR_SENSE2_MUX_SEL	RTCIO_SENSOR_SENSE3_MUX_SEL	RTCIO_SENSOR_SENSE4_MUX_SEL	RTCIO_SENSOR_SENSE1_FUN_SEL	RTCIO_SENSOR_SENSE1_SLP_SEL	RTCIO_SENSOR_SENSE1_SLP_IE	RTCIO_SENSOR_SENSE1_FUN_IE	RTCIO_SENSOR_SENSE2_FUN_SEL	RTCIO_SENSOR_SENSE2_SLP_SEL	RTCIO_SENSOR_SENSE2_SLP_IE	RTCIO_SENSOR_SENSE2_FUN_IE	RTCIO_SENSOR_SENSE3_FUN_SEL	RTCIO_SENSOR_SENSE3_SLP_SEL	RTCIO_SENSOR_SENSE3_SLP_IE	RTCIO_SENSOR_SENSE3_FUN_IE	RTCIO_SENSOR_SENSE4_FUN_SEL	RTCIO_SENSOR_SENSE4_SLP_SEL	RTCIO_SENSOR_SENSE4_SLP_IE	RTCIO_SENSOR_SENSE4_FUN_IE	(reserved)	
31	30	29	28	27	26	25	24	23 22	21	20	19	18 17	16	15	14	13 12	11	10	9	8 7	6	5	4	3 0	
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0 0 0 0	Reset

RTCIO_SENSOR_SENSE*n*_HOLD 1 保 *n* 。0: 常操 。(/写)

RTCIO_SENSOR_SENSE*n*_MUX_SEL 1: *n* RTC ; 0: *n* IO_MUX。(/写)

RTCIO_SENSOR_SENSE*n*_FUN_SEL RTC IO_MUX 。0: Function 0。(/写)

RTCIO_SENSOR_SENSE*n*_SLP_SEL pad 。pad 时 bit 1 。(/写)

RTCIO_SENSOR_SENSE*n*_SLP_IE pad 。1: ; 0: 闭。(/写)

RTCIO_SENSOR_SENSE*n*_FUN_IE pad 。1: ; 0: 闭。(/写)

Register 4.49. RTCIO_ADC_PAD_REG (0x0080)

RTCIO_ADC_ADC1_HOLD	RTCIO_ADC_ADC2_HOLD	RTCIO_ADC_ADC1_MUX_SEL	RTCIO_ADC_ADC2_MUX_SEL	RTCIO_ADC_ADC1_FUN_SEL	RTCIO_ADC_ADC1_SLP_SEL	RTCIO_ADC_ADC1_SLP_IE	RTCIO_ADC_ADC1_FUN_IE	RTCIO_ADC_ADC2_FUN_SEL	RTCIO_ADC_ADC2_SLP_SEL	RTCIO_ADC_ADC2_SLP_IE	RTCIO_ADC_ADC2_FUN_IE	(reserved)	
31	30	29	28	27 26	25	24	23	22 21	20	19	18	17 0	
0	0	0	0	0	0	0	0	0	0	0	0	0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0	Reset

RTCIO_ADC_ADC*n*_HOLD 1 保 pad , 0: 常操 。(/写)

RTCIO_ADC_ADC*n*_MUX_SEL 0: pad IO_MUX。1: pad RTC 。(/写)

RTCIO_ADC_ADC*n*_FUN_SEL RTC 。0: Function 0; 3: Function 1。(/写)

RTCIO_ADC_ADC*n*_SLP_SEL pad 。 1 pad 。(/写)

RTCIO_ADC_ADC*n*_SLP_IE pad 。1: ; 0: 闭。(/写)

RTCIO_ADC_ADC*n*_FUN_IE pad 。1: ; 0: 闭。(/写)

Register 4.50. RTCIO_PAD_DAC1_REG (0x0084)

31 30	29	28	27	26 19	18	17	16 15	14	13	12	11	10	9 0	
(reserved)	RTCIO_PAD_PDAC1_DRV	RTCIO_PAD_PDAC1_HOLD	RTCIO_PAD_PDAC1_RDE	RTCIO_PAD_PDAC1_RUE	RTCIO_PAD_PDAC1_DAC	RTCIO_PAD_PDAC1_XPD_DAC	RTCIO_PAD_PDAC1_MUX_SEL	RTCIO_PAD_PDAC1_FUN_SEL	RTCIO_PAD_PDAC1_SLP_SEL	RTCIO_PAD_PDAC1_SLP_IE	RTCIO_PAD_PDAC1_SLP_OE	RTCIO_PAD_PDAC1_FUN_IE	RTCIO_PAD_PDAC1_DAC_XPD_FORCE	
2	0	0	0	0	0	0	0	0	0	0	0	0	0 0 0 0 0 0 0 0 0 0	Reset

RTCIO_PAD_PDAC1_DRV pad 。(/写)

RTCIO_PAD_PDAC1_HOLD 1 保 pad ，0: 常操 。(/写)

RTCIO_PAD_PDAC1_RDE 1: ；0: 闭。(/写)

RTCIO_PAD_PDAC1_RUE 1: ；0: 闭。(/写)

RTCIO_PAD_PDAC1_DAC PAD DAC1 。(/写)

RTCIO_PAD_PDAC1_XPD_DAC DAC1 。一般 ，DAC 时，PDAC1 ，IE=0、OE=0、RDE=0、RUE=0。(/写)

RTCIO_PAD_PDAC1_MUX_SEL 0: pad IO_MUX。1: RTC 。(/写)

RTCIO_PAD_PDAC1_FUN_SEL pad 。(/写)

RTCIO_PAD_PDAC1_SLP_SEL pad 。 1 pad 。(/写)

RTCIO_PAD_PDAC1_SLP_IE pad 。1: ；0: 闭。(/写)

RTCIO_PAD_PDAC1_SLP_OE pad 。1: ；0: 闭。(/写)

RTCIO_PAD_PDAC1_FUN_IE pad 。1: ；0: 闭。(/写)

RTCIO_PAD_PDAC1_DAC_XPD_FORCE DAC1 。一般 ，DAC 时，PDAC1 ，IE=0、OE=0、RDE=0、RUE=0。(/写)

Register 4.51. RTCIO_PAD_DAC2_REG (0x0088)

31 30	29	28	27	26 19	18	17	16 15	14	13	12	11	10	9 0	
RTCIO_PAD_PDAC2_DRV	RTCIO_PAD_PDAC2_HOLD	RTCIO_PAD_PDAC2_RDE	RTCIO_PAD_PDAC2_RUE	RTCIO_PAD_PDAC2_DAC	RTCIO_PAD_PDAC2_XPD_DAC	RTCIO_PAD_PDAC2_MUX_SEL	RTCIO_PAD_PDAC2_FUN_SEL	RTCIO_PAD_PDAC2_SLP_SEL	RTCIO_PAD_PDAC2_SLP_IE	RTCIO_PAD_PDAC2_SLP_OE	RTCIO_PAD_PDAC2_FUN_IE	RTCIO_PAD_PDAC2_DAC_XPD_FORCE	(reserved)	
2	0	0	0	0	0	0	0	0	0	0	0	0	0 0 0 0 0 0 0 0 0 0	Reset

RTCIO_PAD_PDAC2_DRV pad 。(/写)

RTCIO_PAD_PDAC2_HOLD 1 保 pad ，0: 常操 。(/写)

RTCIO_PAD_PDAC2_RDE 1: ；0: 闭。(/写)

RTCIO_PAD_PDAC2_RUE 1: ；0: 闭。(/写)

RTCIO_PAD_PDAC2_DAC PAD DAC2 。(/写)

RTCIO_PAD_PDAC2_XPD_DAC DAC2 。一般 ，DAC 时，PDAC2 ，IE=0, OE=0, RDE=0, RUE=0。(/写)

RTCIO_PAD_PDAC2_MUX_SEL 0: pad IO_MUX。1: RTC 。(/写)

RTCIO_PAD_PDAC2_FUN_SEL pad RTC ，0: Function 0。(/写)

RTCIO_PAD_PDAC2_SLP_SEL pad 。 1 pad 。(/写)

RTCIO_PAD_PDAC2_SLP_IE pad 。1: ；0: 闭。(/写)

RTCIO_PAD_PDAC2_SLP_OE pad 。1: ；0: 闭。(/写)

RTCIO_PAD_PDAC2_FUN_IE pad 。1: ；0: 闭。(/写)

RTCIO_PAD_PDAC2_DAC_XPD_FORCE DAC2 。一般 ，DAC 时，PDAC2 ，IE=0, OE=0, RDE=0, RUE=0。(/写)

Register 4.52. RTCIO_XTAL_32K_PAD_REG (0x008C)

31 30	29	28	27	26 25	24	23	22	21 20	19	18	17	16 15	14	13	12	11	10 9	8	7	6	5	4 3	2 1	0	
RTCIO_XTAL_X32N_DRV	RTCIO_XTAL_X32N_HOLD	RTCIO_XTAL_X32N_RDE	RTCIO_XTAL_X32N_RUE	RTCIO_XTAL_X32P_DRV	RTCIO_XTAL_X32P_HOLD	RTCIO_XTAL_X32P_RDE	RTCIO_XTAL_X32P_RUE	RTCIO_XTAL_DAC_XTAL_32K	RTCIO_XTAL_XPD_XTAL_32K	RTCIO_XTAL_X32N_MUX_SEL	RTCIO_XTAL_X32P_MUX_SEL	RTCIO_XTAL_X32N_FUN_SEL	RTCIO_XTAL_X32N_SLP_SEL	RTCIO_XTAL_X32N_SLP_IE	RTCIO_XTAL_X32N_SLP_OE	RTCIO_XTAL_X32N_FUN_IE	RTCIO_XTAL_X32P_FUN_SEL	RTCIO_XTAL_X32P_SLP_SEL	RTCIO_XTAL_X32P_SLP_IE	RTCIO_XTAL_X32P_SLP_OE	RTCIO_XTAL_X32P_FUN_IE	RTCIO_XTAL_DRES_XTAL_32K	RTCIO_XTAL_DBIAS_XTAL_32K	(reserved)	
2	0	0	0	2	0	0	0	0 1	0	0	0	0	0	0	0	0	0	0	0	0	0	1 0	0 0	0	Reset

RTCIO_XTAL_X32N_DRV pad 。(/写)

RTCIO_XTAL_X32N_HOLD 1 保 pad , 0: 常操 。(/写)

RTCIO_XTAL_X32N_RDE 1: ; 0: 闭。(/写)

RTCIO_XTAL_X32N_RUE 1: ; 0: 闭。(/写)

RTCIO_XTAL_X32P_DRV pad 。(/写)

RTCIO_XTAL_X32P_HOLD 1 保 pad , 0: 常操 。(/写)

RTCIO_XTAL_X32P_RDE 1: ; 0: 闭。(/写)

RTCIO_XTAL_X32P_RUE 1: ; 0: 闭。(/写)

RTCIO_XTAL_DAC_XTAL_32K 32K XTAL DAC 。(/写)

RTCIO_XTAL_XPD_XTAL_32K 32 kHz 。(/写)

RTCIO_XTAL_X32N_MUX_SEL 0: X32N IO_MUX; 1: RTC 。(/写)

RTCIO_XTAL_X32P_MUX_SEL 0: X32P pad IO_MUX; 1: RTC 。(/写)

RTCIO_XTAL_X32N_FUN_SEL pad RTC , 0: Function 0。(/写)

RTCIO_XTAL_X32N_SLP_SEL pad 。 1 pad 。(/写)

RTCIO_XTAL_X32N_SLP_IE pad 。1: ; 0: 闭。(/写)

RTCIO_XTAL_X32N_SLP_OE pad 。1: ; 0: 闭。(/写)

RTCIO_XTAL_X32N_FUN_IE pad 。1: ; 0: 闭。(/写)

RTCIO_XTAL_X32P_FUN_SEL pad RTC , 0: Function 0; 1: Function 1。(/写)

RTCIO_XTAL_X32P_SLP_SEL , 1 pad 。(/写)

RTCIO_XTAL_X32P_SLP_IE pad 。1: ; 0: 闭。(/写)

RTCIO_XTAL_X32P_SLP_OE pad 。1: ; 0: 闭。(/写)

RTCIO_XTAL_X32P_FUN_IE pad 。1: ; 0: 闭。(/写)

RTCIO_XTAL_DRES_XTAL_32K 32K XTAL 。(/写)

RTCIO_XTAL_DBIAS_XTAL_32K 32K XTAL 参 。(/写)

Register 4.53. RTCIO_TOUCH_CFG_REG (0x0090)

RTCIO_TOUCH_XPD_BIAS	RTCIO_TOUCH_DREFH	RTCIO_TOUCH_DREFL	RTCIO_TOUCH_DRANGE	RTCIO_TOUCH_DCUR	(reserved)	
31	30 29	28 27	26 25	24 23	22 0	
0	1 1	0 0	1 1	0 0	0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0	Reset

RTCIO_TOUCH_XPD_BIAS 上 。1：上 ；0： 。(/写)

RTCIO_TOUCH_DREFH 波 。(/写)

RTCIO_TOUCH_DREFL 波 。(/写)

RTCIO_TOUCH_DRANGE 变 。(/写)

RTCIO_TOUCH_DCUR 。 BIAS_SLEEP 时，可 设 。(/写)

Register 4.54. RTCIO_TOUCH_PADn_REG (n: 0-9) (94+4*n)

(reserved)	RTCIO_TOUCH_PADn_DAC	RTCIO_TOUCH_PADn_START	RTCIO_TOUCH_PADn_TIE_OPT	RTCIO_TOUCH_PADn_XPD	RTCIO_TOUCH_PADn_TO_GPIO	RTCIO_TOUCH_PADn_FUN_SEL	(reserved)	
31 26	25 23	22	21	20	19	18 17	16 0	
0 0 0 0 0 0	0x4	0	0	0	0	0 0	0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0	Reset

RTCIO_TOUCH_PADn_DAC 。 个 pad 3-bit， 认 100。(/写)

RTCIO_TOUCH_PADn_START 启 。(/写)

RTCIO_TOUCH_PADn_TIE_OPT 认 始 。0：0V；1：VDD_RTC 。(/写)

RTCIO_TOUCH_PADn_XPD 上 。(/写)

RTCIO_TOUCH_PADn_TO_GPIO RTC pad pad ，可 0。(/写)

RTCIO_TOUCH_PADn_FUN_SEL RTC pad 。0： RTC Function 0。(/写)

Register 4.55. RTCIO_EXT_WAKEUP0_REG (0x00BC)

RTCIO_EXT_WAKEUP0_SEL	(reserved)	
31 27	26 0	
0	0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0	Reset

RTCIO_EXT_WAKEUP0_SEL GPIO[0-17] 可 中 。 pad Deep/Light-sleep 中 。0： GPIO0；1： GPIO2， 。(/写)

Register 4.56. RTCIO_XTL_EXT_CTR_REG (0x00C0)

RTCIO_XTL_EXT_CTR_SEL	(reserved)	
31 27	26 0	
0	0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0	Reset

RTCIO_XTL_EXT_CTR_SEL 部 。0： GPIO0；1： GPIO2， 。被 RTC_CNTL_XTL_EXT_CTR_LV 上 。(/写)

Register 4.57. RTCIO_SAR_I2C_IO_REG (0x00C4)

RTCIO_SAR_I2C_SDA_SEL	RTCIO_SAR_I2C_SCL_SEL	(reserved)	
31 30	29 28	27 0	
0	0	0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0	Reset

RTCIO_SAR_I2C_SDA_SEL 一个 pad RTC I2C SDA 。0： TOUCH_PAD[1]；1： TOUCH_PAD[3]。 认 0。(/写)

RTCIO_SAR_I2C_SCL_SEL 一个 pad RTC I2C SCL signal。0： TOUCH_PAD[0]；1： TOUCH_PAD[2]。 认 0。(/写)

5 DPort 寄存器

5.1 概述

ESP32 设， DPort 时 ， ， 设 ，可 保 时 。 DPort 中 个 。

5.2 主要特性

DPort 包 个 设 :

-
- 时
- 中
- DMA
- PID/MPU/MMU
- APP_CPU
- 设时

5.3 功能描述

5.3.1 系统和存储器寄存器

， cache 。 。

- DPORT_PRO_BOOT_REMAP_CTRL_REG
- DPORT_APP_BOOT_REMAP_CTRL_REG
- DPORT_CACHE_MUX_MODE_REG

5.3.2 复位和时钟寄存器

时 ， 时 。

- DPORT_CPU_PER_CONF_REG

5.3.3 中断矩阵寄存器

中 中 ， 中 。

- DPORT_CPU_INTR_FROM_CPU_0_REG
- DPORT_CPU_INTR_FROM_CPU_1_REG
- DPORT_CPU_INTR_FROM_CPU_2_REG
- DPORT_CPU_INTR_FROM_CPU_3_REG
- DPORT_PRO_INTR_STATUS_0_REG
- DPORT_PRO_INTR_STATUS_1_REG

- DPORT_PRO_INTR_STATUS_2_REG
- DPORT_APP_INTR_STATUS_0_REG
- DPORT_APP_INTR_STATUS_1_REG
- DPORT_APP_INTR_STATUS_2_REG
- DPORT_PRO_MAC_INTR_MAP_REG
- DPORT_PRO_MAC_NMI_MAP_REG
- DPORT_PRO_BB_INT_MAP_REG
- DPORT_PRO_BT_MAC_INT_MAP_REG
- DPORT_PRO_BT_BB_INT_MAP_REG
- DPORT_PRO_BT_BB_NMI_MAP_REG
- DPORT_PRO_RWBT_IRQ_MAP_REG
- DPORT_PRO_RWBLE_IRQ_MAP_REG
- DPORT_PRO_RWBT_NMI_MAP_REG
- DPORT_PRO_RWBLE_NMI_MAP_REG
- DPORT_PRO_SLC0_INTR_MAP_REG
- DPORT_PRO_SLC1_INTR_MAP_REG
- DPORT_PRO_UHCI0_INTR_MAP_REG
- DPORT_PRO_UHCI1_INTR_MAP_REG
- DPORT_PRO_TG_T0_LEVEL_INT_MAP_REG
- DPORT_PRO_TG_T1_LEVEL_INT_MAP_REG
- DPORT_PRO_TG_WDT_LEVEL_INT_MAP_REG
- DPORT_PRO_TG_LACT_LEVEL_INT_MAP_REG
- DPORT_PRO_TG1_T0_LEVEL_INT_MAP_REG
- DPORT_PRO_TG1_T1_LEVEL_INT_MAP_REG
- DPORT_PRO_TG1_WDT_LEVEL_INT_MAP_REG
- DPORT_PRO_TG1_LACT_LEVEL_INT_MAP_REG
- DPORT_PRO_GPIO_INTERRUPT_MAP_REG
- DPORT_PRO_GPIO_INTERRUPT_NMI_MAP_REG
- DPORT_PRO_CPU_INTR_FROM_CPU_0_MAP_REG
- DPORT_PRO_CPU_INTR_FROM_CPU_1_MAP_REG
- DPORT_PRO_CPU_INTR_FROM_CPU_2_MAP_REG

- DPORT_PRO_CPU_INTR_FROM_CPU_3_MAP_REG
- DPORT_PRO_SPI_INTR_0_MAP_REG
- DPORT_PRO_SPI_INTR_1_MAP_REG
- DPORT_PRO_SPI_INTR_2_MAP_REG
- DPORT_PRO_SPI_INTR_3_MAP_REG
- DPORT_PRO_I2S0_INT_MAP_REG
- DPORT_PRO_I2S1_INT_MAP_REG
- DPORT_PRO_UART_INTR_MAP_REG
- DPORT_PRO_UART1_INTR_MAP_REG
- DPORT_PRO_UART2_INTR_MAP_REG
- DPORT_PRO_SDIO_HOST_INTERRUPT_MAP_REG
- DPORT_PRO_EMAC_INT_MAP_REG
- DPORT_PRO_PWM0_INTR_MAP_REG
- DPORT_PRO_PWM1_INTR_MAP_REG
- DPORT_PRO_PWM2_INTR_MAP_REG
- DPORT_PRO_PWM3_INTR_MAP_REG
- DPORT_PRO_LEDC_INT_MAP_REG
- DPORT_PRO_EFUSE_INT_MAP_REG
- DPORT_PRO_CAN_INT_MAP_REG
- DPORT_PRO_RTC_CORE_INTR_MAP_REG
- DPORT_PRO_RMT_INTR_MAP_REG
- DPORT_PRO_PCNT_INTR_MAP_REG
- DPORT_PRO_I2C_EXT0_INTR_MAP_REG
- DPORT_PRO_I2C_EXT1_INTR_MAP_REG
- DPORT_PRO_RSA_INTR_MAP_REG
- DPORT_PRO_SPI1_DMA_INT_MAP_REG
- DPORT_PRO_SPI2_DMA_INT_MAP_REG
- DPORT_PRO_SPI3_DMA_INT_MAP_REG
- DPORT_PRO_WDG_INT_MAP_REG
- DPORT_PRO_TIMER_INT1_MAP_REG
- DPORT_PRO_TIMER_INT2_MAP_REG

- DPORT_PRO_TG_T0_EDGE_INT_MAP_REG
- DPORT_PRO_TG_T1_EDGE_INT_MAP_REG
- DPORT_PRO_TG_WDT_EDGE_INT_MAP_REG
- DPORT_PRO_TG_LACT_EDGE_INT_MAP_REG
- DPORT_PRO_TG1_T0_EDGE_INT_MAP_REG
- DPORT_PRO_TG1_T1_EDGE_INT_MAP_REG
- DPORT_PRO_TG1_WDT_EDGE_INT_MAP_REG
- DPORT_PRO_TG1_LACT_EDGE_INT_MAP_REG
- DPORT_PRO_MMU_IA_INT_MAP_REG
- DPORT_PRO_MPU_IA_INT_MAP_REG
- DPORT_PRO_CACHE_IA_INT_MAP_REG
- DPORT_APP_MAC_INTR_MAP_REG
- DPORT_APP_MAC_NMI_MAP_REG
- DPORT_APP_BB_INT_MAP_REG
- DPORT_APP_BT_MAC_INT_MAP_REG
- DPORT_APP_BT_BB_INT_MAP_REG
- DPORT_APP_BT_BB_NMI_MAP_REG
- DPORT_APP_RWBT_IRQ_MAP_REG
- DPORT_APP_RWBLE_IRQ_MAP_REG
- DPORT_APP_RWBT_NMI_MAP_REG
- DPORT_APP_RWBLE_NMI_MAP_REG
- DPORT_APP_SLC0_INTR_MAP_REG
- DPORT_APP_SLC1_INTR_MAP_REG
- DPORT_APP_UHCI0_INTR_MAP_REG
- DPORT_APP_UHCI1_INTR_MAP_REG
- DPORT_APP_TG_T0_LEVEL_INT_MAP_REG
- DPORT_APP_TG_T1_LEVEL_INT_MAP_REG
- DPORT_APP_TG_WDT_LEVEL_INT_MAP_REG
- DPORT_APP_TG_LACT_LEVEL_INT_MAP_REG
- DPORT_APP_TG1_T0_LEVEL_INT_MAP_REG
- DPORT_APP_TG1_T1_LEVEL_INT_MAP_REG

- DPORT_APP_TG1_WDT_LEVEL_INT_MAP_REG
- DPORT_APP_TG1_LACT_LEVEL_INT_MAP_REG
- DPORT_APP_GPIO_INTERRUPT_MAP_REG
- DPORT_APP_GPIO_INTERRUPT_NMI_MAP_REG
- DPORT_APP_CPU_INTR_FROM_CPU_0_MAP_REG
- DPORT_APP_CPU_INTR_FROM_CPU_1_MAP_REG
- DPORT_APP_CPU_INTR_FROM_CPU_2_MAP_REG
- DPORT_APP_CPU_INTR_FROM_CPU_3_MAP_REG
- DPORT_APP_SPI_INTR_0_MAP_REG
- DPORT_APP_SPI_INTR_1_MAP_REG
- DPORT_APP_SPI_INTR_2_MAP_REG
- DPORT_APP_SPI_INTR_3_MAP_REG
- DPORT_APP_I2S0_INT_MAP_REG
- DPORT_APP_I2S1_INT_MAP_REG
- DPORT_APP_UART_INTR_MAP_REG
- DPORT_APP_UART1_INTR_MAP_REG
- DPORT_APP_UART2_INTR_MAP_REG
- DPORT_APP_SDIO_HOST_INTERRUPT_MAP_REG
- DPORT_APP_EMAC_INT_MAP_REG
- DPORT_APP_PWM0_INTR_MAP_REG
- DPORT_APP_PWM1_INTR_MAP_REG
- DPORT_APP_PWM2_INTR_MAP_REG
- DPORT_APP_PWM3_INTR_MAP_REG
- DPORT_APP_LEDC_INT_MAP_REG
- DPORT_APP_EFUSE_INT_MAP_REG
- DPORT_APP_CAN_INT_MAP_REG
- DPORT_APP_RTC_CORE_INTR_MAP_REG
- DPORT_APP_RMT_INTR_MAP_REG
- DPORT_APP_PCNT_INTR_MAP_REG
- DPORT_APP_I2C_EXT0_INTR_MAP_REG
- DPORT_APP_I2C_EXT1_INTR_MAP_REG

- DPORT_APP_RSA_INTR_MAP_REG
- DPORT_APP_SPI1_DMA_INT_MAP_REG
- DPORT_APP_SPI2_DMA_INT_MAP_REG
- DPORT_APP_SPI3_DMA_INT_MAP_REG
- DPORT_APP_WDG_INT_MAP_REG
- DPORT_APP_TIMER_INT1_MAP_REG
- DPORT_APP_TIMER_INT2_MAP_REG
- DPORT_APP_TG_T0_EDGE_INT_MAP_REG
- DPORT_APP_TG_T1_EDGE_INT_MAP_REG
- DPORT_APP_TG_WDT_EDGE_INT_MAP_REG
- DPORT_APP_TG_LACT_EDGE_INT_MAP_REG
- DPORT_APP_TG1_T0_EDGE_INT_MAP_REG
- DPORT_APP_TG1_T1_EDGE_INT_MAP_REG
- DPORT_APP_TG1_WDT_EDGE_INT_MAP_REG
- DPORT_APP_TG1_LACT_EDGE_INT_MAP_REG
- DPORT_APP_MMU_IA_INT_MAP_REG
- DPORT_APP_MPU_IA_INT_MAP_REG
- DPORT_APP_CACHE_IA_INT_MAP_REG

5.3.4 DMA 寄存器

SPI DMA , DMA。

- DPORT_SPI_DMA_CHAN_SEL_REG

5.3.5 PID/MPU/MMU 寄存器

PID/MPU/MMU 操 , PID/MPU/MMU。

- DPORT_PRO_CACHE_CTRL_REG
- DPORT_APP_CACHE_CTRL_REG
- DPORT_IMMU_PAGE_MODE_REG
- DPORT_DMMU_PAGE_MODE_REG
- DPORT_AHB_MPU_TABLE_0_REG
- DPORT_AHB_MPU_TABLE_1_REG
- DPORT_AHBLITE_MPU_TABLE_UART_REG
- DPORT_AHBLITE_MPU_TABLE_SPI1_REG

- DPORT_AHBLITE_MPU_TABLE_SPI0_REG
- DPORT_AHBLITE_MPU_TABLE_GPIO_REG
- DPORT_AHBLITE_MPU_TABLE_FE2_REG
- DPORT_AHBLITE_MPU_TABLE_FE_REG
- DPORT_AHBLITE_MPU_TABLE_TIMER_REG
- DPORT_AHBLITE_MPU_TABLE_RTC_REG
- DPORT_AHBLITE_MPU_TABLE_IO_MUX_REG
- DPORT_AHBLITE_MPU_TABLE_WDG_REG
- DPORT_AHBLITE_MPU_TABLE_HINF_REG
- DPORT_AHBLITE_MPU_TABLE_UHCI1_REG
- DPORT_AHBLITE_MPU_TABLE_I2S0_REG
- DPORT_AHBLITE_MPU_TABLE_UART1_REG
- DPORT_AHBLITE_MPU_TABLE_I2C_EXT0_REG
- DPORT_AHBLITE_MPU_TABLE_UHCI0_REG
- DPORT_AHBLITE_MPU_TABLE_SLCHOST_REG
- DPORT_AHBLITE_MPU_TABLE_RMT_REG
- DPORT_AHBLITE_MPU_TABLE_PCNT_REG
- DPORT_AHBLITE_MPU_TABLE_SLC_REG
- DPORT_AHBLITE_MPU_TABLE_LEDC_REG
- DPORT_AHBLITE_MPU_TABLE_EFUSE_REG
- DPORT_AHBLITE_MPU_TABLE_SPI_ENCRYPT_REG
- DPORT_AHBLITE_MPU_TABLE_PWM0_REG
- DPORT_AHBLITE_MPU_TABLE_TIMERGROUP_REG
- DPORT_AHBLITE_MPU_TABLE_TIMERGROUP1_REG
- DPORT_AHBLITE_MPU_TABLE_SPI2_REG
- DPORT_AHBLITE_MPU_TABLE_SPI3_REG
- DPORT_AHBLITE_MPU_TABLE_APB_CTRL_REG
- DPORT_AHBLITE_MPU_TABLE_I2C_EXT1_REG
- DPORT_AHBLITE_MPU_TABLE_SDIO_HOST_REG
- DPORT_AHBLITE_MPU_TABLE_EMAC_REG
- DPORT_AHBLITE_MPU_TABLE_PWM1_REG

- DPORT_AHBLITE_MPU_TABLE_I2S1_REG
- DPORT_AHBLITE_MPU_TABLE_UART2_REG
- DPORT_AHBLITE_MPU_TABLE_PWM2_REG
- DPORT_AHBLITE_MPU_TABLE_PWM3_REG
- DPORT_AHBLITE_MPU_TABLE_PWR_REG
- DPORT_IMMU_TABLE0_REG
- DPORT_IMMU_TABLE1_REG
- DPORT_IMMU_TABLE2_REG
- DPORT_IMMU_TABLE3_REG
- DPORT_IMMU_TABLE4_REG
- DPORT_IMMU_TABLE5_REG
- DPORT_IMMU_TABLE6_REG
- DPORT_IMMU_TABLE7_REG
- DPORT_IMMU_TABLE8_REG
- DPORT_IMMU_TABLE9_REG
- DPORT_IMMU_TABLE10_REG
- DPORT_IMMU_TABLE11_REG
- DPORT_IMMU_TABLE12_REG
- DPORT_IMMU_TABLE13_REG
- DPORT_IMMU_TABLE14_REG
- DPORT_IMMU_TABLE15_REG
- DPORT_DMMU_TABLE0_REG
- DPORT_DMMU_TABLE1_REG
- DPORT_DMMU_TABLE2_REG
- DPORT_DMMU_TABLE3_REG
- DPORT_DMMU_TABLE4_REG
- DPORT_DMMU_TABLE5_REG
- DPORT_DMMU_TABLE6_REG
- DPORT_DMMU_TABLE7_REG
- DPORT_DMMU_TABLE8_REG
- DPORT_DMMU_TABLE9_REG

- DPORT_DMMU_TABLE10_REG
- DPORT_DMMU_TABLE11_REG
- DPORT_DMMU_TABLE12_REG
- DPORT_DMMU_TABLE13_REG
- DPORT_DMMU_TABLE14_REG
- DPORT_DMMU_TABLE15_REG

5.3.6 APP_CPU 控制器寄存器

DPort 可 APP_CPU 本 , , 设 ROM code 启 。

- DPORT_APPCPU_RESETTING=1 时，APP_CPU 被 ; DPORT_APPCPU_RESETTING=0 时，APP_CPU 被 。
- DPORT_APPCPU_CLKGATE_EN=0 时，可 闭 APP_CPU 时 , 。
- DPORT_APPCPU_RUNSTALL=1 时，可 APP_CPU （stall） 。
- APP_CPU ROM code 启 , DPORT_APPCPU_BOOT_ADDR 中 。

5.3.7 外设时钟门控和复位

本 中 时 。 时 1 时， 被 , 0 闭。 1 时， 设 , 0 时 闭 , 设 常 。 , 。

- DPORT_PERI_CLK_EN_REG; 时 。
  - BIT4，Digital Signature
  - BIT3，Secure boot
  - BIT2，RSA
  - BIT1，SHA
  - BIT0，AES
- DPORT_PERI_RST_EN_REG; 。
  - BIT4，Digital Signature
    AES RSA 时 被 。
  - BIT3, Secure boot
    AES SHA 时 被 。
  - BIT2，RSA
  - BIT1，SHA
  - BIT0，AES
- DPORT_PERIP_CLK_EN_REG=1; 设时 。

  - BIT26, PWM3
  - BIT25, PWM2
  - BIT24, UART MEM
    UART , 一个 UART , UART 不 。
  - BIT23, UART2
  - BIT22, SPI_DMA
  - BIT21, I2S1
  - BIT20, PWM1
  - BIT19, CAN
  - BIT18, I2C1
  - BIT17, PWM0
  - BIT16, SPI3
  - BIT15, Timer Group1
  - BIT14, eFuse
  - BIT13, Timer Group0
  - BIT12, UHCI1
  - BIT11, LED_PWM
  - BIT10, PULSE_CNT
  - BIT9, Remote Controller
  - BIT8, UHCI0
  - BIT7, I2C0
  - BIT6, SPI2
  - BIT5, UART1
  - BIT4, I2S0
  - BIT3, WDG
  - BIT2, UART
  - BIT1, SPI
  - BIT0, Timers
- DPORT_PERIP_RST_EN_REG; 设 。
  - BIT26, PWM3
  - BIT25, PWM2

  - BIT24, UART MEM
  - BIT23, UART2
  - BIT22, SPI_DMA
  - BIT21, I2S1
  - BIT20, PWM1
  - BIT19, CAN
  - BIT18, I2C1
  - BIT17, PWM0
  - BIT16, SPI3
  - BIT15, Timer Group1
  - BIT14, eFuse
  - BIT13, Timer Group0
  - BIT12, UHCI1
  - BIT11, LED_PWM
  - BIT10, PULSE_CNT
  - BIT9, Remote Controller
  - BIT8, UHCI0
  - BIT7, I2C0
  - BIT6, SPI2
  - BIT5, UART1
  - BIT4, I2S0
  - BIT3, WDG
  - BIT2, UART
  - BIT1, SPI
  - BIT0, Timers
- DPORT_WIFI_CLK_EN_REG, Wi-Fi BT 时 。
- DPORT_WIFI_RST_EN_REG, Wi-Fi BT 。

5.4 寄存器列表

PRO_BOOT_REMAP_CTRL_REG	PRO_CPU 重	0x3FF00000	/写
APP_BOOT_REMAP_CTRL_REG	APP_CPU 重	0x3FF00004	/写
PERI_CLK_EN_REG	设时	0x3FF0001C	/写
PERI_RST_EN_REG	设	0x3FF00020	/写
APPCPU_CTRL_REG_A_REG	APP_CPU	0x3FF0002C	/写
APPCPU_CTRL_REG_B_REG	APP_CPU 时	0x3FF00030	/写
APPCPU_CTRL_REG_C_REG	APP_CPU (stall)	0x3FF00034	/写
APPCPU_CTRL_REG_D_REG	APP_CPU 启	0x3FF00038	/写
PRO_CACHE_CTRL_REG	External SRAM	0x3FF00040	/写
APP_CACHE_CTRL_REG	External SRAM	0x3FF00058	/写
CACHE_MUX_MODE_REG	个 cache 时	0x3FF0007C	/写
IMMU_PAGE_MODE_REG	Internal SRAM 0 MMU	0x3FF00080	/写
DMMU_PAGE_MODE_REG	Internal SRAM 2 MMU	0x3FF00084	/写
SRAM_PD_CTRL_REG_0_REG	闭 Internal SRAM_REG	0x3FF00098	/写
SRAM_PD_CTRL_REG_1_REG	闭 Internal SRAM_REG	0x3FF0009C	/写
AHB_MPU_TABLE_0_REG	DMA MPU	0x3FF000B4	/写
AHB_MPU_TABLE_1_REG	DMA MPU	0x3FF000B8	/写
PERIP_CLK_EN_REG	设时	0x3FF000C0	/写
PERIP_RST_EN_REG	设	0x3FF000C4	/写
SLAVE_SPI_CONFIG_REG	Flash	0x3FF000C8	/写
WIFI_CLK_EN_REG	Wi-Fi 时	0x3FF000CC	/写
WIFI_RST_EN_REG	Wi-Fi	0x3FF000D0	/写
CPU_INTR_FROM_CPU_0_REG	个 CPU Interrupt 0	0x3FF000DC	/写
CPU_INTR_FROM_CPU_1_REG	个 CPU Interrupt 1	0x3FF000E0	/写
CPU_INTR_FROM_CPU_2_REG	个 CPU Interrupt 2	0x3FF000E4	/写
CPU_INTR_FROM_CPU_3_REG	个 CPU Interrupt 3	0x3FF000E8	/写
PRO_INTR_STATUS_REG_0_REG	PRO_CPU 中 0	0x3FF000EC	
PRO_INTR_STATUS_REG_1_REG	PRO_CPU 中 1	0x3FF000F0	
PRO_INTR_STATUS_REG_2_REG	PRO_CPU 中 2	0x3FF000F4	
APP_INTR_STATUS_REG_0_REG	APP_CPU 中 0	0x3FF000F8	
APP_INTR_STATUS_REG_1_REG	APP_CPU 中 1	0x3FF000FC	
APP_INTR_STATUS_REG_2_REG	APP_CPU 中 2	0x3FF00100	
PRO_MAC_INTR_MAP_REG	中	0x3FF00104	/写
PRO_MAC_NMI_MAP_REG	中	0x3FF00108	/写
PRO_BB_INT_MAP_REG	中	0x3FF0010C	/写
PRO_BT_MAC_INT_MAP_REG	中	0x3FF00110	/写
PRO_BT_BB_INT_MAP_REG	中	0x3FF00114	/写
PRO_BT_BB_NMI_MAP_REG	中	0x3FF00118	/写
PRO_RWBT_IRQ_MAP_REG	中	0x3FF0011C	/写
PRO_RWBLE_IRQ_MAP_REG	中	0x3FF00120	/写
PRO_RWBT_NMI_MAP_REG	中	0x3FF00124	/写
PRO_RWBLE_NMI_MAP_REG	中	0x3FF00128	/写
PRO_SLC0_INTR_MAP_REG	中	0x3FF0012C	/写

PRO_SLC1_INTR_MAP_REG	中	0x3FF00130	/写
PRO_UHCI0_INTR_MAP_REG	中	0x3FF00134	/写
PRO_UHCI1_INTR_MAP_REG	中	0x3FF00138	/写
PRO_TG_T0_LEVEL_INT_MAP_REG	中	0x3FF0013C	/写
PRO_TG_T1_LEVEL_INT_MAP_REG	中	0x3FF00140	/写
PRO_TG_WDT_LEVEL_INT_MAP_REG	中	0x3FF00144	/写
PRO_TG_LACT_LEVEL_INT_MAP_REG	中	0x3FF00148	/写
PRO_TG1_T0_LEVEL_INT_MAP_REG	中	0x3FF0014C	/写
PRO_TG1_T1_LEVEL_INT_MAP_REG	中	0x3FF00150	/写
PRO_TG1_WDT_LEVEL_INT_MAP_REG	中	0x3FF00154	/写
PRO_TG1_LACT_LEVEL_INT_MAP_REG	中	0x3FF00158	/写
PRO_GPIO_INTERRUPT_MAP_REG	中	0x3FF0015C	/写
PRO_GPIO_INTERRUPT_NMI_MAP_REG	中	0x3FF00160	/写
PRO_CPU_INTR_FROM_CPU_0_MAP_REG	中	0x3FF00164	/写
PRO_CPU_INTR_FROM_CPU_1_MAP_REG	中	0x3FF00168	/写
PRO_CPU_INTR_FROM_CPU_2_MAP_REG	中	0x3FF0016C	/写
PRO_CPU_INTR_FROM_CPU_3_MAP_REG	中	0x3FF00170	/写
PRO_SPI_INTR_0_MAP_REG	中	0x3FF00174	/写
PRO_SPI_INTR_1_MAP_REG	中	0x3FF00178	/写
PRO_SPI_INTR_2_MAP_REG	中	0x3FF0017C	/写
PRO_SPI_INTR_3_MAP_REG	中	0x3FF00180	/写
PRO_I2S0_INT_MAP_REG	中	0x3FF00184	/写
PRO_I2S1_INT_MAP_REG	中	0x3FF00188	/写
PRO_UART_INTR_MAP_REG	中	0x3FF0018C	/写
PRO_UART1_INTR_MAP_REG	中	0x3FF00190	/写
PRO_UART2_INTR_MAP_REG	中	0x3FF00194	/写
PRO_SDIO_HOST_INTERRUPT_MAP_REG	中	0x3FF00198	/写
PRO_EMAC_INT_MAP_REG	中	0x3FF0019C	/写
PRO_PWM0_INTR_MAP_REG	中	0x3FF001A0	/写
PRO_PWM1_INTR_MAP_REG	中	0x3FF001A4	/写
PRO_PWM2_INTR_MAP_REG	中	0x3FF001A8	/写
PRO_PWM3_INTR_MAP_REG	中	0x3FF001AC	/写
PRO_LEDC_INT_MAP_REG	中	0x3FF001B0	/写
PRO_EFUSE_INT_MAP_REG	中	0x3FF001B4	/写
PRO_CAN_INT_MAP_REG	中	0x3FF001B8	/写
PRO_RTC_CORE_INTR_MAP_REG	中	0x3FF001BC	/写
PRO_RMT_INTR_MAP_REG	中	0x3FF001C0	/写
PRO_PCNT_INTR_MAP_REG	中	0x3FF001C4	/写
PRO_I2C_EXT0_INTR_MAP_REG	中	0x3FF001C8	/写
PRO_I2C_EXT1_INTR_MAP_REG	中	0x3FF001CC	/写
PRO_RSA_INTR_MAP_REG	中	0x3FF001D0	/写
PRO_SPI1_DMA_INT_MAP_REG	中	0x3FF001D4	/写
PRO_SPI2_DMA_INT_MAP_REG	中	0x3FF001D8	/写

PRO_SPI3_DMA_INT_MAP_REG	未	0x3FF001DC	/写
PRO_WDG_INT_MAP_REG	未	0x3FF001E0	/写
PRO_TIMER_INT1_MAP_REG	未	0x3FF001E4	/写
PRO_TIMER_INT2_MAP_REG	未	0x3FF001E8	/写
PRO_TG_T0_EDGE_INT_MAP_REG	未	0x3FF001EC	/写
PRO_TG_T1_EDGE_INT_MAP_REG	未	0x3FF001F0	/写
PRO_TG_WDT_EDGE_INT_MAP_REG	未	0x3FF001F4	/写
PRO_TG_LACT_EDGE_INT_MAP_REG	未	0x3FF001F8	/写
PRO_TG1_T0_EDGE_INT_MAP_REG	未	0x3FF001FC	/写
PRO_TG1_T1_EDGE_INT_MAP_REG	未	0x3FF00200	/写
PRO_TG1_WDT_EDGE_INT_MAP_REG	未	0x3FF00204	/写
PRO_TG1_LACT_EDGE_INT_MAP_REG	未	0x3FF00208	/写
PRO_MMU_IA_INT_MAP_REG	未	0x3FF0020C	/写
PRO_MPU_IA_INT_MAP_REG	未	0x3FF00210	/写
PRO_CACHE_IA_INT_MAP_REG	未	0x3FF00214	/写
APP_MAC_INTR_MAP_REG	未	0x3FF00218	/写
APP_MAC_NMI_MAP_REG	未	0x3FF0021C	/写
APP_BB_INT_MAP_REG	未	0x3FF00220	/写
APP_BT_MAC_INT_MAP_REG	未	0x3FF00224	/写
APP_BT_BB_INT_MAP_REG	未	0x3FF00228	/写
APP_BT_BB_NMI_MAP_REG	未	0x3FF0022C	/写
APP_RWBT_IRQ_MAP_REG	未	0x3FF00230	/写
APP_RWBLE_IRQ_MAP_REG	未	0x3FF00234	/写
APP_RWBT_NMI_MAP_REG	未	0x3FF00238	/写
APP_RWBLE_NMI_MAP_REG	未	0x3FF0023C	/写
APP_SLC0_INTR_MAP_REG	未	0x3FF00240	/写
APP_SLC1_INTR_MAP_REG	未	0x3FF00244	/写
APP_UHCI0_INTR_MAP_REG	未	0x3FF00248	/写
APP_UHCI1_INTR_MAP_REG	未	0x3FF0024C	/写
APP_TG_T0_LEVEL_INT_MAP_REG	未	0x3FF00250	/写
APP_TG_T1_LEVEL_INT_MAP_REG	未	0x3FF00254	/写
APP_TG_WDT_LEVEL_INT_MAP_REG	未	0x3FF00258	/写
APP_TG_LACT_LEVEL_INT_MAP_REG	未	0x3FF0025C	/写
APP_TG1_T0_LEVEL_INT_MAP_REG	未	0x3FF00260	/写
APP_TG1_T1_LEVEL_INT_MAP_REG	未	0x3FF00264	/写
APP_TG1_WDT_LEVEL_INT_MAP_REG	未	0x3FF00268	/写
APP_TG1_LACT_LEVEL_INT_MAP_REG	未	0x3FF0026C	/写
APP_GPIO_INTERRUPT_MAP_REG	未	0x3FF00270	/写
APP_GPIO_INTERRUPT_NMI_MAP_REG	未	0x3FF00274	/写
APP_CPU_INTR_FROM_CPU_0_MAP_REG	未	0x3FF00278	/写
APP_CPU_INTR_FROM_CPU_1_MAP_REG	未	0x3FF0027C	/写
APP_CPU_INTR_FROM_CPU_2_MAP_REG	未	0x3FF00280	/写
APP_CPU_INTR_FROM_CPU_3_MAP_REG	未	0x3FF00284	/写

APP_SPI_INTR_0_MAP_REG	中	0x3FF00288	/写
APP_SPI_INTR_1_MAP_REG	中	0x3FF0028C	/写
APP_SPI_INTR_2_MAP_REG	中	0x3FF00290	/写
APP_SPI_INTR_3_MAP_REG	中	0x3FF00294	/写
APP_I2S0_INT_MAP_REG	中	0x3FF00298	/写
APP_I2S1_INT_MAP_REG	中	0x3FF0029C	/写
APP_UART_INTR_MAP_REG	中	0x3FF002A0	/写
APP_UART1_INTR_MAP_REG	中	0x3FF002A4	/写
APP_UART2_INTR_MAP_REG	中	0x3FF002A8	/写
APP_SDIO_HOST_INTERRUPT_MAP_REG	中	0x3FF002AC	/写
APP_EMAC_INT_MAP_REG	中	0x3FF002B0	/写
APP_PWM0_INTR_MAP_REG	中	0x3FF002B4	/写
APP_PWM1_INTR_MAP_REG	中	0x3FF002B8	/写
APP_PWM2_INTR_MAP_REG	中	0x3FF002BC	/写
APP_PWM3_INTR_MAP_REG	中	0x3FF002C0	/写
APP_LEDC_INT_MAP_REG	中	0x3FF002C4	/写
APP_EFUSE_INT_MAP_REG	中	0x3FF002C8	/写
APP_CAN_INT_MAP_REG	中	0x3FF002CC	/写
APP_RTC_CORE_INTR_MAP_REG	中	0x3FF002D0	/写
APP_RMT_INTR_MAP_REG	中	0x3FF002D4	/写
APP_PCNT_INTR_MAP_REG	中	0x3FF002D8	/写
APP_I2C_EXT0_INTR_MAP_REG	中	0x3FF002DC	/写
APP_I2C_EXT1_INTR_MAP_REG	中	0x3FF002E0	/写
APP_RSA_INTR_MAP_REG	中	0x3FF002E4	/写
APP_SPI1_DMA_INT_MAP_REG	中	0x3FF002E8	/写
APP_SPI2_DMA_INT_MAP_REG	中	0x3FF002EC	/写
APP_SPI3_DMA_INT_MAP_REG	中	0x3FF002F0	/写
APP_WDG_INT_MAP_REG	中	0x3FF002F4	/写
APP_TIMER_INT1_MAP_REG	中	0x3FF002F8	/写
APP_TIMER_INT2_MAP_REG	中	0x3FF002FC	/写
APP_TG_T0_EDGE_INT_MAP_REG	中	0x3FF00300	/写
APP_TG_T1_EDGE_INT_MAP_REG	中	0x3FF00304	/写
APP_TG_WDT_EDGE_INT_MAP_REG	中	0x3FF00308	/写
APP_TG_LACT_EDGE_INT_MAP_REG	中	0x3FF0030C	/写
APP_TG1_T0_EDGE_INT_MAP_REG	中	0x3FF00310	/写
APP_TG1_T1_EDGE_INT_MAP_REG	中	0x3FF00314	/写
APP_TG1_WDT_EDGE_INT_MAP_REG	中	0x3FF00318	/写
APP_TG1_LACT_EDGE_INT_MAP_REG	中	0x3FF0031C	/写
APP_MMU_IA_INT_MAP_REG	中	0x3FF00320	/写
APP_MPU_IA_INT_MAP_REG	中	0x3FF00324	/写
APP_CACHE_IA_INT_MAP_REG	中	0x3FF00328	/写
AHBLITE_MPU_TABLE_UART_REG	设 MPU	0x3FF0032C	/写
AHBLITE_MPU_TABLE_SPI1_REG	设 MPU	0x3FF00330	/写

AHBLITE_MPU_TABLE_SPI0_REG	设 MPU	0x3FF00334	/写
AHBLITE_MPU_TABLE_GPIO_REG	设 MPU	0x3FF00338	/写
AHBLITE_MPU_TABLE_RTC_REG	设 MPU	0x3FF00348	/写
AHBLITE_MPU_TABLE_IO_MUX_REG	设 MPU	0x3FF0034C	/写
AHBLITE_MPU_TABLE_HINF_REG	设 MPU	0x3FF00354	/写
AHBLITE_MPU_TABLE_UHCI1_REG	设 MPU	0x3FF00358	/写
AHBLITE_MPU_TABLE_I2S0_REG	设 MPU	0x3FF00364	/写
AHBLITE_MPU_TABLE_UART1_REG	设 MPU	0x3FF00368	/写
AHBLITE_MPU_TABLE_I2C_EXT0_REG	设 MPU	0x3FF00374	/写
AHBLITE_MPU_TABLE_UHCI0_REG	设 MPU	0x3FF00378	/写
AHBLITE_MPU_TABLE_SLCHOST_REG	设 MPU	0x3FF0037C	/写
AHBLITE_MPU_TABLE_RMT_REG	设 MPU	0x3FF00380	/写
AHBLITE_MPU_TABLE_PCNT_REG	设 MPU	0x3FF00384	/写
AHBLITE_MPU_TABLE_SLC_REG	设 MPU	0x3FF00388	/写
AHBLITE_MPU_TABLE_LEDC_REG	设 MPU	0x3FF0038C	/写
AHBLITE_MPU_TABLE_EFUSE_REG	设 MPU	0x3FF00390	/写
AHBLITE_MPU_TABLE_SPI_ENCRYPT_REG	设 MPU	0x3FF00394	/写
AHBLITE_MPU_TABLE_PWM0_REG	设 MPU	0x3FF0039C	/写
AHBLITE_MPU_TABLE_TIMERGROUP_REG	设 MPU	0x3FF003A0	/写
AHBLITE_MPU_TABLE_TIMERGROUP1_REG	设 MPU	0x3FF003A4	/写
AHBLITE_MPU_TABLE_SPI2_REG	设 MPU	0x3FF003A8	/写
AHBLITE_MPU_TABLE_SPI3_REG	设 MPU	0x3FF003AC	/写
AHBLITE_MPU_TABLE_APB_CTRL_REG	设 MPU	0x3FF003B0	/写
AHBLITE_MPU_TABLE_I2C_EXT1_REG	设 MPU	0x3FF003B4	/写
AHBLITE_MPU_TABLE_SDIO_HOST_REG	设 MPU	0x3FF003B8	/写
AHBLITE_MPU_TABLE_EMAC_REG	设 MPU	0x3FF003BC	/写
AHBLITE_MPU_TABLE_PWM1_REG	设 MPU	0x3FF003C4	/写
AHBLITE_MPU_TABLE_I2S1_REG	设 MPU	0x3FF003C8	/写
AHBLITE_MPU_TABLE_UART2_REG	设 MPU	0x3FF003CC	/写
AHBLITE_MPU_TABLE_PWM2_REG	设 MPU	0x3FF003D0	/写
AHBLITE_MPU_TABLE_PWM3_REG	设 MPU	0x3FF003D4	/写
AHBLITE_MPU_TABLE_PWR_REG	设 MPU	0x3FF003E4	/写
IMMU_TABLE0_REG	Internal SRAM 0 MMU	0x3FF00504	/写
IMMU_TABLE1_REG	Internal SRAM 0 MMU	0x3FF00508	/写
IMMU_TABLE2_REG	Internal SRAM 0 MMU	0x3FF0050C	/写
IMMU_TABLE3_REG	Internal SRAM 0 MMU	0x3FF00510	/写
IMMU_TABLE4_REG	Internal SRAM 0 MMU	0x3FF00514	/写
IMMU_TABLE5_REG	Internal SRAM 0 MMU	0x3FF00518	/写
IMMU_TABLE6_REG	Internal SRAM 0 MMU	0x3FF0051C	/写
IMMU_TABLE7_REG	Internal SRAM 0 MMU	0x3FF00520	/写
IMMU_TABLE8_REG	Internal SRAM 0 MMU	0x3FF00524	/写
IMMU_TABLE9_REG	Internal SRAM 0 MMU	0x3FF00528	/写
IMMU_TABLE10_REG	Internal SRAM 0 MMU	0x3FF0052C	/写

IMMU_TABLE11_REG	Internal SRAM 0 MMU	0x3FF00530	/写
IMMU_TABLE12_REG	Internal SRAM 0 MMU	0x3FF00534	/写
IMMU_TABLE13_REG	Internal SRAM 0 MMU	0x3FF00538	/写
IMMU_TABLE14_REG	Internal SRAM 0 MMU	0x3FF0053C	/写
IMMU_TABLE15_REG	Internal SRAM 0 MMU	0x3FF00540	/写
DMMU_TABLE0_REG	Internal SRAM 2 MMU	0x3FF00544	/写
DMMU_TABLE1_REG	Internal SRAM 2 MMU	0x3FF00548	/写
DMMU_TABLE2_REG	Internal SRAM 2 MMU	0x3FF0054C	/写
DMMU_TABLE3_REG	Internal SRAM 2 MMU	0x3FF00550	/写
DMMU_TABLE4_REG	Internal SRAM 2 MMU	0x3FF00554	/写
DMMU_TABLE5_REG	Internal SRAM 2 MMU	0x3FF00558	/写
DMMU_TABLE6_REG	Internal SRAM 2 MMU	0x3FF0055C	/写
DMMU_TABLE7_REG	Internal SRAM 2 MMU	0x3FF00560	/写
DMMU_TABLE8_REG	Internal SRAM 2 MMU	0x3FF00564	/写
DMMU_TABLE9_REG	Internal SRAM 2 MMU	0x3FF00568	/写
DMMU_TABLE10_REG	Internal SRAM 2 MMU	0x3FF0056C	/写
DMMU_TABLE11_REG	Internal SRAM 2 MMU	0x3FF00570	/写
DMMU_TABLE12_REG	Internal SRAM 2 MMU	0x3FF00574	/写
DMMU_TABLE13_REG	Internal SRAM 2 MMU	0x3FF00578	/写
DMMU_TABLE14_REG	Internal SRAM 2 MMU	0x3FF0057C	/写
DMMU_TABLE15_REG	Internal SRAM 2 MMU	0x3FF00580	/写
SECURE_BOOT_CTRL_REG	secure_boot	0x3FF005A4	/写
SPI_DMA_CHAN_SEL_REG	SPI1, SPI2, SPI3 DMA	0x3FF005A8	/写

5.5 寄存器

Register 5.1. PRO_BOOT_REMAP_CTRL_REG (0x000)

31																															1	0	
(reserved)																																PRO_BOOT_REMAP	
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	Reset

PRO_BOOT_REMAP PRO_CPU 重 。(/ 写)

Register 5.2. APP_BOOT_REMAP_CTRL_REG (0x004)

31																															1	0	
(reserved)																																APP_BOOT_REMAP	
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	Reset

APP_BOOT_REMAP APP_CPU 重 。(/ 写)

Register 5.3. PERI_CLK_EN_REG (0x01C)

31	0	
0x000000000		Reset

PERI_CLK_EN_REG 设时 。(/ 写)

Register 5.4. PERI_RST_EN_REG (0x020)

31	0	
0x000000000		Reset

PERI_RST_EN_REG 设 。(/ 写)

Register 5.5. APPCPU_CTRL_REG_A_REG (0x02C)

(reserved)	APPCPU_RESETTING	
31 … 1	0	
0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0	1	Reset

APPCPU_RESETTING APP_CPU 。(/ 写)

Register 5.6. APPCPU_CTRL_REG_B_REG (0x030)

(reserved)	APPCPU_CLKGATE_EN	
31 … 1	0	
0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0	0	Reset

APPCPU_CLKGATE_EN APP_CPU 时 。(/ 写)

Register 5.7. APPCPU_CTRL_REG_C_REG (0x034)

(reserved)	APPCPU_RUNSTALL	
31 … 1	0	
0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0	0	Reset

APPCPU_RUNSTALL APP_CPU (stall)。(/ 写)

Register 5.8. APPCPU_CTRL_REG_D_REG (0x038)

31 … 0	
0x000000000	Reset

APPCPU_CTRL_REG_D_REG APP_CPU 启 。(/ 写)

Register 5.9. CPU_PER_CONF_REG (0x03C)

(reserved)	CPU_CPUPERIOD_SEL	
31 … 2	1 … 0	
0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0	0 0	Reset

CPU_CPUPERIOD_SEL CPU 时 。 参 表 3-3。(/ 写)

Register 5.10. PRO_CACHE_CTRL_REG (0x040)

(reserved)	PRO_DRAM_HL	(reserved)	PRO_DRAM_SPLIT	PRO_SINGLE_IRAM_ENA	(reserved)	PRO_CACHE_FLUSH_DONE	PRO_CACHE_FLUSH_ENA	PRO_CACHE_ENABLE	(reserved)	
31 … 17	16	15 … 12	11	10	9 … 6	5	4	3	2 … 0	
0 0 0 0 0 0 0 0 0 0 0 0 0 0 0	0	0 0 0 0	0	0	0 0 0 0	0	1	0	0 0 0	Reset

PRO_DRAM_HL External SRAM 。(/ 写)

PRO_DRAM_SPLIT External SRAM 。(/ 写)

PRO_SINGLE_IRAM_ENA PRO_CPU external flash 。(/ 写)

PRO_CACHE_FLUSH_DONE PRO_CPU cache 标 。( )

PRO_CACHE_FLUSH_ENA PRO_CPU cache。(/ 写)

PRO_CACHE_ENABLE PRO_CPU cache。(/ 写)

Register 5.11. APP_CACHE_CTRL_REG (0x058)

(reserved)	APP_DRAM_HL	(reserved)	APP_DRAM_SPLIT	APP_SINGLE_IRAM_ENA	(reserved)	APP_CACHE_FLUSH_DONE	APP_CACHE_FLUSH_ENA	APP_CACHE_ENABLE	(reserved)	
31 15	14	13 12	11	10	9 6	5	4	3	2 0	
0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0	0	0 0	0	0	0 0 0 0	0	1	0	0 0 0	Reset

APP_DRAM_HL External SRAM 。(/ 写)

APP_DRAM_SPLIT External SRAM 。(/ 写)

APP_SINGLE_IRAM_ENA APP_CPU external flash 。(/ 写)

APP_CACHE_FLUSH_DONE APP_CPU cache 标 。( )

APP_CACHE_FLUSH_ENA APP_CPU cache。(/ 写)

APP_CACHE_ENABLE APP_CPU cache。(/ 写)

Register 5.12. CACHE_MUX_MODE_REG (0x07C)

(reserved)	CACHE_MUX_MODE	
31 2	1 0	
0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0	0 0	Reset

CACHE_MUX_MODE 个 cache 。(/ 写)

Register 5.13. IMMU_PAGE_MODE_REG (0x080)

(reserved)	IMMU_PAGE_MODE	(reserved)	
31 3	2 1	0	
0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0	0 0	0	Reset

IMMU_PAGE_MODE Internal SRAM 0 MMU 。(/ 写)

Register 5.14. DMMU_PAGE_MODE_REG (0x084)

(reserved)	DMMU_PAGE_MODE	(reserved)
31 … 3	2 … 1	0
0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0	0 0	0
		Reset

DMMU_PAGE_MODE Internal SRAM 2 MMU 。(/ 写)

Register 5.15. SRAM_PD_CTRL_REG_0_REG (0x098)

31 … 0
0x000000000
Reset

SRAM_PD_CTRL_REG_0_REG 闭 internal SRAM。(/ 写)

Register 5.16. SRAM_PD_CTRL_REG_1_REG (0x09C)

(reserved)	SRAM_PD_1
31 … 1	0
0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0	0
	Reset

SRAM_PD_1 闭 internal SRAM。(/ 写)

Register 5.17. AHB_MPU_TABLE_0_REG (0x0B4)

31 … 0
0x0FFFFFFFF
Reset

AHB_MPU_TABLE_0_REG DMA MPU。(/ 写)

Register 5.22. WIFI_CLK_EN_REG (0x0CC)

31	0	
0x0FFFCE030		Reset

WIFI_CLK_EN_REG Wi-Fi 时 。(/ 写)

Register 5.23. WIFI_RST_EN_REG (0x0D0)

31	0	
0x000000000		Reset

WIFI_RST_EN_REG Wi-Fi 。(/ 写)

Register 5.24. CPU_INTR_FROM_CPU_*n*_REG (*n*: 0-3) (0xDC+4**n*)

(reserved)	CPU_INTR_FROM_CPU_*n*	
31 … 1	0	
0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0	0	Reset

CPU_INTR_FROM_CPU_*n* 1 CPU 中 。(/ 写)

Register 5.25. PRO_INTR_STATUS_REG_*n*_REG (*n*: 0-2) (0xEC+4**n*)

31	0	
0x000000000		Reset

PRO_INTR_STATUS_REG_*n*_REG PRO_CPU 中 。( )

Register 5.26. APP_INTR_STATUS_REG_*n*_REG (*n*: 0-2) (0xF8+4**n*)

31	0	
0x000000000		Reset

APP_INTR_STATUS_REG_*n*_REG APP_CPU 中 。( )

Register 5.27. PRO_MAC_INTR_MAP_REG (0x104)

Register 5.28. PRO_MAC_NMI_MAP_REG (0x108)
Register 5.29. PRO_BB_INT_MAP_REG (0x10C)
Register 5.30. PRO_BT_MAC_INT_MAP_REG (0x110)
Register 5.31. PRO_BT_BB_INT_MAP_REG (0x114)
Register 5.32. PRO_BT_BB_NMI_MAP_REG (0x118)
Register 5.33. PRO_RWBT_IRQ_MAP_REG (0x11C)
Register 5.34. PRO_RWBLE_IRQ_MAP_REG (0x120)
Register 5.35. PRO_RWBT_NMI_MAP_REG (0x124)
Register 5.36. PRO_RWBLE_NMI_MAP_REG (0x128)
Register 5.37. PRO_SLC0_INTR_MAP_REG (0x12C)
Register 5.38. PRO_SLC1_INTR_MAP_REG (0x130)
Register 5.39. PRO_UHCI0_INTR_MAP_REG (0x134)
Register 5.40. PRO_UHCI1_INTR_MAP_REG (0x138)
Register 5.41. PRO_TG_T0_LEVEL_INT_MAP_REG (0x13C)
Register 5.42. PRO_TG_T1_LEVEL_INT_MAP_REG (0x140)
Register 5.43. PRO_TG_WDT_LEVEL_INT_MAP_REG (0x144)
Register 5.44. PRO_TG_LACT_LEVEL_INT_MAP_REG (0x148)
Register 5.45. PRO_TG1_T0_LEVEL_INT_MAP_REG (0x14C)
Register 5.46. PRO_TG1_T1_LEVEL_INT_MAP_REG (0x150)
Register 5.47. PRO_TG1_WDT_LEVEL_INT_MAP_REG (0x154)
Register 5.48. PRO_TG1_LACT_LEVEL_INT_MAP_REG (0x158)
Register 5.49. PRO_GPIO_INTERRUPT_MAP_REG (0x15C)
Register 5.50. PRO_GPIO_INTERRUPT_NMI_MAP_REG (0x160)
Register 5.51. PRO_CPU_INTR_FROM_CPU_0_MAP_REG (0x164)
Register 5.52. PRO_CPU_INTR_FROM_CPU_1_MAP_REG (0x168)
Register 5.53. PRO_CPU_INTR_FROM_CPU_2_MAP_REG (0x16C)
Register 5.54. PRO_CPU_INTR_FROM_CPU_3_MAP_REG (0x170)
Register 5.55. PRO_SPI_INTR_0_MAP_REG (0x174)
Register 5.56. PRO_SPI_INTR_1_MAP_REG (0x178)
Register 5.57. PRO_SPI_INTR_2_MAP_REG (0x17C)
Register 5.58. PRO_SPI_INTR_3_MAP_REG (0x180)
Register 5.59. PRO_I2S0_INT_MAP_REG (0x184)
Register 5.60. PRO_I2S1_INT_MAP_REG (0x188)
Register 5.61. PRO_UART_INTR_MAP_REG (0x18C)
Register 5.62. PRO_UART1_INTR_MAP_REG (0x190)

Register 5.63. PRO_UART2_INTR_MAP_REG (0x194)
Register 5.64. PRO_SDIO_HOST_INTERRUPT_MAP_REG (0x198)
Register 5.65. PRO_EMAC_INT_MAP_REG (0x19C)
Register 5.66. PRO_PWM0_INTR_MAP_REG (0x1A0)
Register 5.67. PRO_PWM1_INTR_MAP_REG (0x1A4)
Register 5.68. PRO_PWM2_INTR_MAP_REG (0x1A8)
Register 5.69. PRO_PWM3_INTR_MAP_REG (0x1AC)
Register 5.70. PRO_LEDC_INT_MAP_REG (0x1B0)
Register 5.71. PRO_EFUSE_INT_MAP_REG (0x1B4)
Register 5.72. PRO_CAN_INT_MAP_REG (0x1B8)
Register 5.73. PRO_RTC_CORE_INTR_MAP_REG (0x1BC)
Register 5.74. PRO_RMT_INTR_MAP_REG (0x1C0)
Register 5.75. PRO_PCNT_INTR_MAP_REG (0x1C4)
Register 5.76. PRO_I2C_EXT0_INTR_MAP_REG (0x1C8)
Register 5.77. PRO_I2C_EXT1_INTR_MAP_REG (0x1CC)
Register 5.78. PRO_RSA_INTR_MAP_REG (0x1D0)
Register 5.79. PRO_SPI1_DMA_INT_MAP_REG (0x1D4)
Register 5.80. PRO_SPI2_DMA_INT_MAP_REG (0x1D8)
Register 5.81. PRO_SPI3_DMA_INT_MAP_REG (0x1DC)
Register 5.82. PRO_WDG_INT_MAP_REG (0x1E0)
Register 5.83. PRO_TIMER_INT1_MAP_REG (0x1E4)
Register 5.84. PRO_TIMER_INT2_MAP_REG (0x1E8)
Register 5.85. PRO_TG_T0_EDGE_INT_MAP_REG (0x1EC)
Register 5.86. PRO_TG_T1_EDGE_INT_MAP_REG (0x1F0)
Register 5.87. PRO_TG_WDT_EDGE_INT_MAP_REG (0x1F4)
Register 5.88. PRO_TG_LACT_EDGE_INT_MAP_REG (0x1F8)
Register 5.89. PRO_TG1_T0_EDGE_INT_MAP_REG (0x1FC)
Register 5.90. PRO_TG1_T1_EDGE_INT_MAP_REG (0x200)
Register 5.91. PRO_TG1_WDT_EDGE_INT_MAP_REG (0x204)
Register 5.92. PRO_TG1_LACT_EDGE_INT_MAP_REG (0x208)
Register 5.93. PRO_MMU_IA_INT_MAP_REG (0x20C)
Register 5.94. PRO_MPU_IA_INT_MAP_REG (0x210)

Register 5.95. PRO_CACHE_IA_INT_MAP_REG (0x214)

(reserved)	PRO_*_MAP
31 … 5	4 … 0
0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0	16

Reset

PRO_*_MAP 中 。(/ 写)

Register 5.96. APP_MAC_INTR_MAP_REG (0x218)

Register 5.97. APP_MAC_NMI_MAP_REG (0x21C)

Register 5.98. APP_BB_INT_MAP_REG (0x220)

Register 5.99. APP_BT_MAC_INT_MAP_REG (0x224)

Register 5.100. APP_BT_BB_INT_MAP_REG (0x228)

Register 5.101. APP_BT_BB_NMI_MAP_REG (0x22C)

Register 5.102. APP_RWBT_IRQ_MAP_REG (0x230)

Register 5.103. APP_RWBLE_IRQ_MAP_REG (0x234)

Register 5.104. APP_RWBT_NMI_MAP_REG (0x238)

Register 5.105. APP_RWBLE_NMI_MAP_REG (0x23C)

Register 5.106. APP_SLC0_INTR_MAP_REG (0x240)

Register 5.107. APP_SLC1_INTR_MAP_REG (0x244)

Register 5.108. APP_UHCI0_INTR_MAP_REG (0x248)

Register 5.109. APP_UHCI1_INTR_MAP_REG (0x24C)

Register 5.110. APP_TG_T0_LEVEL_INT_MAP_REG (0x250)

Register 5.111. APP_TG_T1_LEVEL_INT_MAP_REG (0x254)

Register 5.112. APP_TG_WDT_LEVEL_INT_MAP_REG (0x258)

Register 5.113. APP_TG_LACT_LEVEL_INT_MAP_REG (0x25C)

Register 5.114. APP_TG1_T0_LEVEL_INT_MAP_REG (0x260)

Register 5.115. APP_TG1_T1_LEVEL_INT_MAP_REG (0x264)

Register 5.116. APP_TG1_WDT_LEVEL_INT_MAP_REG (0x268)

Register 5.117. APP_TG1_LACT_LEVEL_INT_MAP_REG (0x26C)

Register 5.118. APP_GPIO_INTERRUPT_MAP_REG (0x270)

Register 5.119. APP_GPIO_INTERRUPT_NMI_MAP_REG (0x274)

Register 5.120. APP_CPU_INTR_FROM_CPU_0_MAP_REG (0x278)

Register 5.121. APP_CPU_INTR_FROM_CPU_1_MAP_REG (0x27C)

Register 5.122. APP_CPU_INTR_FROM_CPU_2_MAP_REG (0x280)

Register 5.123. APP_CPU_INTR_FROM_CPU_3_MAP_REG (0x284)
Register 5.124. APP_SPI_INTR_0_MAP_REG (0x288)
Register 5.125. APP_SPI_INTR_1_MAP_REG (0x28C)
Register 5.126. APP_SPI_INTR_2_MAP_REG (0x290)
Register 5.127. APP_SPI_INTR_3_MAP_REG (0x294)
Register 5.128. APP_I2S0_INT_MAP_REG (0x298)
Register 5.129. APP_I2S1_INT_MAP_REG (0x29C)
Register 5.130. APP_UART_INTR_MAP_REG (0x2A0)
Register 5.131. APP_UART1_INTR_MAP_REG (0x2A4)
Register 5.132. APP_UART2_INTR_MAP_REG (0x2A8)
Register 5.133. APP_SDIO_HOST_INTERRUPT_MAP_REG (0x2AC)
Register 5.134. APP_EMAC_INT_MAP_REG (0x2B0)
Register 5.135. APP_PWM0_INTR_MAP_REG (0x2B4)
Register 5.136. APP_PWM1_INTR_MAP_REG (0x2B8)
Register 5.137. APP_PWM2_INTR_MAP_REG (0x2BC)
Register 5.138. APP_PWM3_INTR_MAP_REG (0x2C0)
Register 5.139. APP_LEDC_INT_MAP_REG (0x2C4)
Register 5.140. APP_EFUSE_INT_MAP_REG (0x2C8)
Register 5.141. APP_CAN_INT_MAP_REG (0x2CC)
Register 5.142. APP_RTC_CORE_INTR_MAP_REG (0x2D0)
Register 5.143. APP_RMT_INTR_MAP_REG (0x2D4)
Register 5.144. APP_PCNT_INTR_MAP_REG (0x2D8)
Register 5.145. APP_I2C_EXT0_INTR_MAP_REG (0x2DC)
Register 5.146. APP_I2C_EXT1_INTR_MAP_REG (0x2E0)
Register 5.147. APP_RSA_INTR_MAP_REG (0x2E4)
Register 5.148. APP_SPI1_DMA_INT_MAP_REG (0x2E8)
Register 5.149. APP_SPI2_DMA_INT_MAP_REG (0x2EC)
Register 5.150. APP_SPI3_DMA_INT_MAP_REG (0x2F0)
Register 5.151. APP_WDG_INT_MAP_REG (0x2F4)
Register 5.152. APP_TIMER_INT1_MAP_REG (0x2F8)
Register 5.153. APP_TIMER_INT2_MAP_REG (0x2FC)
Register 5.154. APP_TG_T0_EDGE_INT_MAP_REG (0x300)
Register 5.155. APP_TG_T1_EDGE_INT_MAP_REG (0x304)
Register 5.156. APP_TG_WDT_EDGE_INT_MAP_REG (0x308)
Register 5.157. APP_TG_LACT_EDGE_INT_MAP_REG (0x30C)

Register 5.158. APP_TG1_T0_EDGE_INT_MAP_REG (0x310)

Register 5.159. APP_TG1_T1_EDGE_INT_MAP_REG (0x314)

Register 5.160. APP_TG1_WDT_EDGE_INT_MAP_REG (0x318)

Register 5.161. APP_TG1_LACT_EDGE_INT_MAP_REG (0x31C)

Register 5.162. APP_MMU_IA_INT_MAP_REG (0x320)

Register 5.163. APP_MPU_IA_INT_MAP_REG (0x324)

Register 5.164. APP_CACHE_IA_INT_MAP_REG (0x328)

(reserved)	APP_*_MAP	
31 … 5	4 … 0	
0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0	16	Reset

APP_*_MAP 中 。(/ 写)

Register 5.165. AHBLITE_MPU_TABLE_UART_REG (0x32C)

Register 5.166. AHBLITE_MPU_TABLE_SPI1_REG (0x330)

Register 5.167. AHBLITE_MPU_TABLE_SPI0_REG (0x334)

Register 5.168. AHBLITE_MPU_TABLE_GPIO_REG (0x338)

Register 5.169. AHBLITE_MPU_TABLE_RTC_REG (0x348)

Register 5.170. AHBLITE_MPU_TABLE_IO_MUX_REG (0x34C)

Register 5.171. AHBLITE_MPU_TABLE_HINF_REG (0x354)

Register 5.172. AHBLITE_MPU_TABLE_UHCI1_REG (0x358)

Register 5.173. AHBLITE_MPU_TABLE_I2S0_REG (0x364)

Register 5.174. AHBLITE_MPU_TABLE_UART1_REG (0x368)

Register 5.175. AHBLITE_MPU_TABLE_I2C_EXT0_REG (0x374)

Register 5.176. AHBLITE_MPU_TABLE_UHCI0_REG (0x378)

Register 5.177. AHBLITE_MPU_TABLE_SLCHOST_REG (0x37C)

Register 5.178. AHBLITE_MPU_TABLE_RMT_REG (0x380)

Register 5.179. AHBLITE_MPU_TABLE_PCNT_REG (0x384)

Register 5.180. AHBLITE_MPU_TABLE_SLC_REG (0x388)

Register 5.181. AHBLITE_MPU_TABLE_LEDC_REG (0x38C)

Register 5.182. AHBLITE_MPU_TABLE_EFUSE_REG (0x390)

Register 5.183. AHBLITE_MPU_TABLE_SPI_ENCRYPT_REG (0x394)

Register 5.184. AHBLITE_MPU_TABLE_PWM0_REG (0x39C)

Register 5.185. AHBLITE_MPU_TABLE_TIMERGROUP_REG (0x3A0)

Register 5.186. AHBLITE_MPU_TABLE_TIMERGROUP1_REG (0x3A4)

Register 5.187. AHBLITE_MPU_TABLE_SPI2_REG (0x3A8)

Register 5.188. AHBLITE_MPU_TABLE_SPI3_REG (0x3AC)

Register 5.189. AHBLITE_MPU_TABLE_APB_CTRL_REG (0x3B0)

Register 5.190. AHBLITE_MPU_TABLE_I2C_EXT1_REG (0x3B4)

Register 5.191. AHBLITE_MPU_TABLE_SDIO_HOST_REG (0x3B8)

Register 5.192. AHBLITE_MPU_TABLE_EMAC_REG (0x3BC)

Register 5.193. AHBLITE_MPU_TABLE_PWM1_REG (0x3C4)

Register 5.194. AHBLITE_MPU_TABLE_I2S1_REG (0x3C8)

Register 5.195. AHBLITE_MPU_TABLE_UART2_REG (0x3CC)

Register 5.196. AHBLITE_MPU_TABLE_PWM2_REG (0x3D0)

Register 5.197. AHBLITE_MPU_TABLE_PWM3_REG (0x3D4)

Register 5.198. AHBLITE_MPU_TABLE_PWR_REG (0x3E4)

(reserved)	AHBLITE_*_ACCESS_GRANT_CONFIG	
31 … 6	5 … 0	
0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0	0 0 0 0 0 0	Reset

AHBLITE_*_ACCESS_GRANT_CONFIG 设 MPU。(/ 写)

Register 5.199. IMMU_TABLE*n*_REG (*n*: 0-15) (0x504+4***n*)

(reserved)	IMMU_TABLE*n*	
31 … 7	6 … 0	
0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0	15	Reset

IMMU_TABLE*n* Internal SRAM MMU。(/ 写)

Register 5.200. DMMU_TABLE*n*_REG (*n*: 0-15) (0x544+4***n*)

(reserved)	DMMU_TABLE*n*	
31 … 7	6 … 0	
0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0	15	Reset

DMMU_TABLE*n* Internal SRAM MMU。(/ 寄)

Register 5.201. SECURE_BOOT_CTRL_REG (0x5A4)

(reserved)	SECURE_SW_BOOTLOADER_SEL	
31 … 1	0	
0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0	0	Reset

SECURE_SW_BOOTLOADER_SEL secure_boot 。(/ 寄)

Register 5.202. SPI_DMA_CHAN_SEL_REG (0x5A8)

(reserved)	SPI_SPI3_DMA_CHAN_SEL	SPI_SPI2_DMA_CHAN_SEL	SPI_SPI1_DMA_CHAN_SEL	
31 … 6	5 … 4	3 … 2	1 … 0	
0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0	0 0	0 0	0 0	Reset

SPI_SPI3_DMA_CHAN_SEL SPI3 DMA 。(/ 寄)

SPI_SPI2_DMA_CHAN_SEL SPI2 DMA 。(/ 寄)

SPI_SPI1_DMA_CHAN_SEL SPI1 DMA 。(/ 寄)

6 DMA 控制器 (DMA)

6.1 概述

(Direct Memory Access, DMA) 设 间 间 。可 CPU 操 DMA ，而 CPU 。

ESP32 中 13 个 设 DMA ， 13 个 设 ：UART0、UART1、UART2、SPI1、SPI2、SPI3、I2S0、I2S1、SDIO slave、SD/MMC host、EMAC、BT Wi-Fi。

6.2 特性

DMA 个 ：

- AHB
- 半
- 单 ， 可 编
- 4-beat burst
- 328 KB DMA 间
- DMA

6.3 功能描述

ESP32 中 DMA 。DMA CPU 间 中 部 RAM。

， 个 DMA 差别， DMA (DMA_ENGINE) 。

6.3.1 DMA 引擎的架构

图 6-1. DMA 引擎的架构

DMA AHB_BUS 部 RAM RAM 。 6-1 DMA 本 。 中 RAM ESP32 部 SRAM，SRAM 。 可 表

DMA 。DMA_ENGING out_link 中 RAM 中 ， 可 in_link 中 RAM 间。

6.3.2 链表

图 6-2. 链表结构图

out_link in_link ， 6-2 表 ，一个 表 3 个 。 一 ：

- owner (DW0) [31]：表 表 buffer 操 。
  1'b0： 操 CPU；
  1'b1： 操 DMA 。
- eof (DW0) [30]：表 标 。
  1'b0： 表不 一个 表；
  1'b1： 表 包 一个 表。
- reserved (DW0) [29:24]：reserved。
  不 写 1。
- length (DW0) [23:12]：表 表 buffer 中 。 buffer 中 时，表 ； buffer 中 时，表 。
- size (DW0) [11:0]：表 表 buffer 。
  注意： 必 。
- buffer address pointer (DW1)：buffer 。
  注意： 必 。
- next descriptor address (DW2)： 一个 表 。 表 一个 表时，(eof=1)， 0。

DMA 时， 一 长 buffer 长 ， DMA 不 个 buffer 间。 DMA_ENGING 可 。

6.4 UART DMA (UDMA) 控制器

ESP32 中 3 个 UART ， 2 个 UDMA 。UHCI_UART*x*_CE（*x* 0、1、2） 一个 UART UDMA。

图 6-3. UDMA 模式数据传输

6-3 UDMA 。 , 表 备 。UHCI_INLINK_ADDR 一个 in_link 表。 必 表 20 。 UHCI_INLINK_START , (UHCI) UART Decoder。 Decoder DMA 表 RAM 间。

, 表 备 , UHCI_OUTLINK_ADDR 一个 out_link 表。 必 表 20 。 UHCI_OUTLINK_START , DMA 表中 RAM , 并 Encoder 包 , UART 。

UART DMA 包 (+ +)。Encoder 上 , 并 中 一 。Decoder 包 , 并 中 。 可 个。 可 UHCI_SEPER_CHAR , 认 0xC0。 中 一 可 UHCI_ESC_SEQ0_CHAR0 (认 0xDB) UHCI_ESC_SEQ0_CHAR1 (认 0xDD) 。 部 , 产 UHCI_OUT_TOTAL_EOF_INT 中 。 , 产 UHCI_IN_SUC_EOF_INT 中 。

6.5 SPI DMA 控制器

图 6-4. SPI DMA

ESP32 SPI CPU 设 , 可 DMA 。 6-4 , 个 DMA 可 SPI1、SPI2 SPI3 , 个 DMA 可 一个 SPI , 可 个 SPI 时 DMA。

ESP32 SPI DMA 表 / , burst 操 , 一 / 长 1 个 。SPI DMA 。

DPORT_SPI_DMA_CHAN_SEL_REG SPI1_DMA_CHAN_SEL[1:0]、SPI2_DMA_CHAN_SEL[1:0] SPI3_DMA_CHAN_SEL[1:0] 三个 SPI DMA 。 个 SPI 一个 , 个 2 比 , 0、1 2，所不可 3 。

SPI1 ,
SPI1_DMA_CHAN_SEL[1:0] = 0， SPI1 不 DMA ;
SPI1_DMA_CHAN_SEL[1:0] = 1， SPI1 DMA 1;
SPI1_DMA_CHAN_SEL[1:0] = 2， SPI1 DMA 2。

SPI_DMA_OUT_LINK_REG SPI_OUTLINK_START 比 SPI_DMA_IN_LINK_REG SPI_INLINK_START 比 DMA , 个比 。 SPI_OUTLINK_START 比 被 1 时，DMA 始 表，并 备 ; SPI_INLINK_START 比 被 1 时，DMA 始 表，并 备 。

SPI_DMA :

1. 字 DMA FIFO ;
2. DMA ;
3. SPI ;
4. 一 DMA 操 。

6.6 I2S DMA 控制器

ESP32 个 I2S , I2S0 I2S1。I2S0 I2S1 一个 DMA 。 I2S_FIFO_CONF_REG REG_I2S_DSCR_EN 比 I2S DMA 操 。ESP32 I2S DMA 表 / , burst 操 , 一 / 长 1 个 （4 个 ）。 I2S_RXEOF_NUM_REG REG_I2S_RX_EOF_NUM[31:0] 比 DMA 一 长 , 单 。

I2S_OUT_LINK_REG I2S_OUTLINK_START 比 I2S_IN_LINK_REG I2S_INLINK_START 比 DMA , 个比 。 I2S_OUTLINK_START 比 被 1 时，DMA 始 表，并 备 , I2S_INLINK_START 比 被 1 时，DMA 始 表，并 备 。

I2S DMA :

1. 字 I2S ;
2. DMA FIFO ;
3. DMA ;
4. I2S ，设 I2S_TX_START 比 I2S_RX_START 比 , 一 I2S 操 ;
   I2S ，设 I2S_TX_START 比 I2S_RX_START 比 。

I2S DMA 中 见 I2S，DMA 中 。

7 SPI 控制器 (SPI)

7.1 概述

图 7-1. SPI 系统框图

7-1 ，ESP32 4 个 SPI SPI0、SPI1、SPI2、SPI3， SPI 设备。SPI0 cache 部 单 ，SPI1 ，SPI2 SPI3 可 可 。 时， 个 SPI 可 个 (CS0 ~ CS2) 个 SPI 设备。SPI1 ~ SPI3 个 DMA 。

SPI0 SPI1 一个 裁 一 ， SPI D、Q、CS0 ~ CS2、CLK、WP HD ， 表 7-1 。 ， SPI2 SPI3 别 HSPI VSPI 。 包 可 GPIO IO_MUX 可 (IO_MUX)。

SPI GP-SPI ， 标 /半 (MOSI、MISO、CS、CLK) 三 半 (DATA、CS、CLK)。SPI QSPI D、Q、CS0 ~ CS2、CLK、WP HD 4-bit 并 SPI 部 flash SRAM。不 表 7-1 。

表 7-1. 管脚功能信号与总线信号映射关系

GP-SPI /半	GP-SPI 三 半	QSPI			
			SPI	HSPI	VSPI
MOSI	DATA	D	SPID	HSPID	VSPID
MISO	-	Q	SPIQ	HSPIQ	VSPIQ
CS	CS	CS	SPICS0	HSPICS0	VSPICS0
CLK	CLK	CLK	SPICLK	HSPICLK	VSPICLK
-	-	WP	SPIWP	HSPIWP	VSPIWP
-	-	HD	SPIHD	HSPIHD	VSPIHD

7.2 SPI 特征

GP-SPI（通用 SPI）接口

- 长 byte 单 可
- /半 三 半

-
- 时 (CPOL) 时 (CPHA) 可
- 时 可

并行 QSPI 接口

- flash 设备
- 可
- 6 flash 操
- flash SRAM
- flash

SPI DMA 接口

- 表 /

SPI 中断接口

- SPI 中
- SPI DMA 中

7.3 GP-SPI 接口

ESP32 SPI /半 三 半 。 /半 7-2 。

图 7-2. SPI 四线全双工/半双工通信

ESP32 SPI1 ~ SPI3 可 SPI ，SPI2 SPI3 可 。 个 ESP32 SPI 认 可 3 个 。 DMA ，一 可 / 64 byte ， 长 单 。

7.3.1 GP-SPI 四线全双工模式

时，ESP32 SPI 可 可 。 时，长 ： ， SPI_MISO_DLEN_REG SPI_MOSI_DLEN_REG ； ， SPI_SLV_RDBUF_DLEN_REG SPI_SLV_WRBUF_DLEN_REG 。 SPI_USER_REG SPI_DOUTDIN SPI_USR_MOSI 。 SPI_CMD_REG SPI_USR 启 一 。

7.3.2 GP-SPI 四线半双工模式

半 时，ESP32 SPI 可 可 。 时 ESP32 SPI
： + + + / 。 ：

1. ： 长 0 ~ 16 bit； 。
2. ： 长 0 ~ 32 bit/64 bit； 。
3. ： 长 0 ~ 256 个 SPI 时 。
4. / ： 长 0 ~ 512 bit (64 byte)； 。

中 GP-SPI 长 32 bit，QSPI 长 64 bit。 、
、 / 别 SPI_USER_REG SPI_USR_COMMAND、
SPI_USR_ADDR、SPI_USR_DUMMY SPI_USR_MISO/SPI_USR_MOSI 。 比 被 1 时，一
SPI 操 才 ， 。 时， 可
包 、 、 。

时， 时必 包 、 ，并 表 7-2。 一
/ 中， CS 必 保 。 中 CS 被 ， 部 。

表 7-2. 从机命令描述

0x1	， MOSI 写
0x2	， MOSI 写
0x3	， 中 MISO
0x4	， 中 MISO
0x6	MOSI 上 写 ， 中 MISO

可 写 SPI SPI_SLV_WR_STATUS_REG，并可 SPI_SLAVE1_REG 中
SPI_SLV_STATUS_READBACK ， SPI_SLV_WR_STATUS_REG SPI_RD_STATUS_REG
。SPI 可 写 ， 保 ，
。

SPI_MISO_DLEN_REG SPI_MOSI_DLEN_REG 、
SPI_SLV_RDBUF_DLEN_REG、SPI_SLV_WRBUF_DLEN_REG SPI 长 。
SPI_USER_REG SPI_USR_MOSI SPI_USR_MISO SPI 。
SPI_CMD_REG SPI_USR 启 一 。

7.3.3 GP-SPI 三线半双工模式

ESP32 SPI 三 半 半 别 一 ， 必 包 、
。 SPI_USER_REG SPI_SIO 三 半
。

说明：

- 半 ， 、 顺 不可变；
- 半 ，“ + + + ” “ + ” 不 DMA；

- ESP32 SPI 时， CS 一个 SPI 时 长 ； /写 ， CS 一个 SPI 时 长 。

7.3.4 GP-SPI 数据缓存

图 7-3. SPI 数据缓存

ESP32 SPI 16 × 32 bit ， / 。 7-3 。 时， 认 SPI_W0_REG 部 始 ，一 SPI_W15_REG 。 长 64 byte， 部 SPI_W0_REG 始 。

SPI_W0_REG ~ SPI_W7_REG SPI_W8_REG ~ SPI_W15_REG 别 个部 ， 可 。 SPI_USER_REG SPI_USR_MOSI_HIGHPART SPI_USR_MISO_HIGHPART 个比 。 ， SPI ， SPI_USR_MOSI_HIGHPART = 1 时， SPI_W8_REG ~ SPI_W15_REG ， SPI_USR_MISO_HIGHPART = 1 时， SPI_W8_REG-SPI_W15_REG 。而 SPI 时， SPI_USR_MOSI_HIGHPART = 1， SPI_W8_REG ~ SPI_W15_REG ， SPI_USR_MISO_HIGHPART = 1， SPI_W8_REG-SPI_W15_REG 。

7.4 GP-SPI 时钟控制

ESP32 GP-SPI 时 $f_{apb}/2$， 时 $f_{apb}/8$。 可 时 。

$$f_{spi} = \frac{f_{apb}}{(SPI_CLKCNT_N+1)(SPI_CLKDIV_PRE+1)}$$

中 SPI_CLKCNT_N SPI_CLKDIV_PRE SPI_CLOCK_REG 个 （ 见 7.7 ）， $SPI_CLKCNT_H = \lfloor \frac{SPI_CLKCNT_N+1}{2} - 1 \rfloor$， SPI_CLKCNT_N=SPI_CLKCNT_L。 SPI_CLOCK_REG

SPI_CLK_EQU_SYSCLK 1， 0 时， SPI 时 f_{apb}； ， SPI_CLK_EQU_SYSCLK 0。 ， SPI_CLKCNT_N、SPI_CLKCNT_L、SPI_CLKCNT_H SPI_CLKDIV_PRE 0。

7.4.1 GP-SPI 时钟极性和时钟相位

ESP32 SPI 时 ， SPI_PIN_REG SPI_CK_IDLE_EDGE 、 SPI_USER_REG SPI_CK_OUT_EDGE SPI_CK_I_EDGE ， SPI_CTRL2_REG SPI_MISO_DELAY_MODE[1:0] 、SPI_MISO_DELAY_NUM[2:0] 、SPI_MOSI_DELAY_MODE[1:0] SPI_MOSI_DELAY_MUM[2:0] 。表 7-3 表 7-4 别 ESP32 SPI 时

。表 7-4 中 mode0 mode2 DMA DMA 。
DMA mode0 mode2 。

表 7-3. 主机模式时钟极性和相位控制寄存器值

	mode0	mode1	mode2	mode3
SPI_CK_IDLE_EDGE	0	0	1	1
SPI_CK_OUT_EDGE	0	1	1	0
SPI_MISO_DELAY_MODE	2(0)	1(0)	1(0)	2(0)
SPI_MISO_DELAY_NUM	0	0	0	0
SPI_MOSI_DELAY_MODE	0	0	0	0
SPI_MOSI_DELAY_NUM	0	0	0	0

表 7-4. 从机模式时钟极性和相位控制寄存器值

	mode0		mode1	mode2		mode3
	DMA	DMA		DMA	DMA	
SPI_CK_IDLE_EDGE	1	0	1	0	1	0
SPI_CK_I_EDGE	0	1	1	1	0	0
SPI_MISO_DELAY_MODE	0	0	2	0	0	1
SPI_MISO_DELAY_NUM	0	2	0	0	2	0
SPI_MOSI_DELAY_MODE	2	0	0	1	0	0
SPI_MOSI_DELAY_NUM	2	3	0	2	3	0

1. mode0 表 CPOL=0, CPHA=0，SPI 时，时 ，SPI 变化 上升 采 ；
2. mode1 表 CPOL=0, CPHA=1，SPI 时，时 ，SPI 上升 变化 采 ；
3. mode2 表 CPOL=1, CPHA=0，SPI 时，时 ，SPI 上升 变化 采 ；
4. mode3 表 CPOL=1, CPHA=1，SPI 时，时 ，SPI 变化 上升 采 。

7.4.2 GP-SPI 时序

ESP32 GP-SPI 可 IO_MUX ，可 IO_MUX GPIO 。 时，被 个 clk_{apb} 时 期，不 被 。

GP-SPI ，并 不 GPIO SPI 时，GP-SPI 时 $clk_{apb}/2$，时 时，SPI_MISO_DELAY_MODE 0。GP-SPI 时 不 $clk_{apb}/4$，时 时，SPI_MISO_DELAY_MODE 可 表 7-3 中 。

GP-SPI 时，并 GPIO SPI ：

1. GP-SPI 时 $clk_{apb}/2$，时 时，SPI_MISO_DELAY_MODE 0，时 (SPI_USR_DUMMY = 1)，长 1 个 clk_{spi} 时 期 (SPI_USR_DUMMY_CYCLELEN = 0)；

2. GP-SPI 时 $clk_{apb}/4$, 时 时 SPI_MISO_DELAY_MODE 0;
3. GP-SPI 时 不 $clk_{apb}/8$, 时 时, SPI_MISO_DELAY_MODE 可 表 7-3 中 。

GP-SPI 时, 时 SPI , 时 不 GPIO SPI 。 才 SPI , 时间不 。

mode0 , 7-4 中 t_{spi}、t_{pre} t_v 别 SPI 时 期、 时间 时间, 时 设 SPI 时 期 t_{apb}。 DMA , , 时间 t_v:

- CLK 不 , $t_v < 3.5 * t_{apb}$;
- CLK , $t_v < 5.5 * t_{apb}$;

DMA , mode1 mode3, , 时间 t_v DMA 。 mode0 mode2, , 时间 t_{pre}:

- clk 不 , $t_{pre} < (t_{spi}/2 - 5.5 * t_{apb})$;
- clk , $t_{pre} < (t_{spi}/2 - 7.5 * t_{apb})$;

图 7-4. GP-SPI 从机数据输出

, 不 , 时 $f_{apb}/8$, , $f_{apb}/12$。 时 , mode0 mode2 时, $(t_{spi}/2 - t_{pre})$ 保 时间时 。

7.5 并行 QSPI 接口

ESP32 SPI SPI (flash, SRAM) 。SPI 7-5 。

SPI1、SPI2 SPI3 可 QSPI 部 。SPI 时 f_{apb}, 时 GP-SPI 时 。

7.5.1 并行 QSPI 接口通信格式

ESP32 QSPI 间 , 单 设 。ESP32 QSPI GP-SPI 半 。不 , 可 2 4 , 7-6 , 4 。

ESP32 QSPI 一 、 flash 操 。 GP-SPI , 时, QSPI 可 、 、 。

注意： GPI-SPI 不可 。

图 7-5. 并行 QSPI 接口

图 7-6. 并行 QSPI 接口的通信模式

7.6 GP-SPI 中断硬件

ESP32 SPI 中 ，一 SPI 中 ， 一 SPI DMA 中 。

ESP32 SPI / 操 时 中 一 一个， 认 一 操 ，所不 。ESP32 SPI 时， 操 不 ， 产 /写 /写 中 。

7.6.1 SPI 中断

SPI_SLAVE_REG SPI_*_INTEN 1，可 SPI 中 。 中 时，SPI_*_DONE 中 标 被 1。中 标 可写， 中 ， bit 0。

- SPI_TRANS_DONE_INT：SPI 操 被 。
- SPI_SLV_WR_STA_INT：SPI 写 时被 。
- SPI_SLV_RD_STA_INT：SPI 时被 。
- SPI_SLV_WR_BUF_INT：SPI 写 时被 。
- SPI_SLV_RD_BUD_INT：SPI 时被 。

7.6.2 DMA 中断

- SPI_OUT_TOTAL_EOF_INT： 表 时被 。
- SPI_OUT_EOF_INT：一个 表 时被 。

- SPI_OUT_DONE_INT: 一个 表长 0 时被 。
- SPI_IN_SUC_EOF_INT: 表被 时被 。
- SPI_IN_ERR_EOF_INT: 表 时被 。
- SPI_IN_DONE_INT: 一个 表长 0 时被 。
- SPI_INLINK_DSCR_ERROR_INT: 表 单 时被 。
- SPI_OUTLINK_DSCR_ERROR_INT: 表 时被 。
- SPI_INLINK_DSCR_EMPTY_INT: 表时被 。

7.7 寄存器列表

		SPI0	SPI1	SPI2	SPI3	
控制和配置寄存器						
SPI_CTRL_REG	Bit 顺 QIO/DIO/QOUT/DOUT 设	3FF43008	3FF42008	3FF64008	3FF65008	/写
SPI_CTRL2_REG	时	3FF43014	3FF42014	3FF64014	3FF65014	/写
SPI_CLOCK_REG	时	3FF43018	3FF42018	3FF64018	3FF65018	/写
SPI_PIN_REG	CS	3FF43034	3FF42034	3FF64034	3FF65034	/写
从机模式配置寄存器						
SPI_SLAVE_REG	中	3FF43038	3FF42038	3FF64038	3FF65038	/写
SPI_SLAVE1_REG	bit 长	3FF4303C	3FF4203C	3FF6403C	3FF6503C	/写
SPI_SLAVE2_REG	期长	3FF43040	3FF42040	3FF64040	3FF65040	/写
SPI_SLV_WR_STATUS_REG	/	3FF43030	3FF42030	3FF64030	3FF65030	/写
SPI_SLV_WRBUF_DLEN_REG	写 操 长	3FF43048	3FF42048	3FF64048	3FF65048	/写
SPI_SLV_RDBUF_DLEN_REG	操 长	3FF4304C	3FF4204C	3FF6404C	3FF6504C	/写
SPI_SLV_RD_BIT_REG	操 长	3FF43064	3FF42064	3FF64064	3FF65064	/写
用户自定义命令模式寄存器						
SPI_CMD_REG	始	3FF43000	3FF42000	3FF64000	3FF65000	/写
SPI_ADDR_REG		3FF43004	3FF42004	3FF64004	3FF65004	/写
SPI_USER_REG		3FF4301C	3FF4201C	3FF6401C	3FF6501C	/写
SPI_USER1_REG	期	3FF43020	3FF42020	3FF64020	3FF65020	/写
SPI_USER2_REG	长	3FF43024	3FF42024	3FF64024	3FF65024	/写
SPI_MOSI_DLEN_REG	MOSI 长	3FF43028	3FF42028	3FF64028	3FF65028	/写
SPI_W0_REG	SPI 0	3FF43080	3FF42080	3FF64080	3FF65080	/写
SPI_W1_REG	SPI 1	3FF43084	3FF42084	3FF64084	3FF65084	/写
SPI_W2_REG	SPI 2	3FF43088	3FF42088	3FF64088	3FF65088	/写
SPI_W3_REG	SPI 3	3FF4308C	3FF4208C	3FF6408C	3FF6508C	/写
SPI_W4_REG	SPI 4	3FF43090	3FF42090	3FF64090	3FF65090	/写
SPI_W5_REG	SPI 5	3FF43094	3FF42094	3FF64094	3FF65094	/写
SPI_W6_REG	SPI 6	3FF43098	3FF42098	3FF64098	3FF65098	/写

		SPI0	SPI1	SPI2	SPI3	
SPI_W7_REG	SPI 7	3FF4309C	3FF4209C	3FF6409C	3FF6509C	/写
SPI_W8_REG	SPI 8	3FF430A0	3FF420A0	3FF640A0	3FF650A0	/写
SPI_W9_REG	SPI 9	3FF430A4	3FF420A4	3FF640A4	3FF650A4	/写
SPI_W10_REG	SPI 10	3FF430A8	3FF420A8	3FF640A8	3FF650A8	/写
SPI_W11_REG	SPI 11	3FF430AC	3FF420AC	3FF640AC	3FF650AC	/写
SPI_W12_REG	SPI 12	3FF430B0	3FF420B0	3FF640B0	3FF650B0	/写
SPI_W13_REG	SPI 13	3FF430B4	3FF420B4	3FF640B4	3FF650B4	/写
SPI_W14_REG	SPI 14	3FF430B8	3FF420B8	3FF640B8	3FF650B8	/写
SPI_W15_REG	SPI 15	3FF430BC	3FF420BC	3FF640BC	3FF650BC	/写
DMA 配置寄存器						
SPI_DMA_CONF_REG	DMA	3FF43100	3FF42100	3FF64100	3FF65100	/写
SPI_DMA_OUT_LINK_REG	DMA 表	3FF43104	3FF42104	3FF64104	3FF65104	/写
SPI_DMA_IN_LINK_REG	DMA 表	3FF43108	3FF42108	3FF64108	3FF65108	/写
SPI_DMA_STATUS_REG	DMA	3FF4310C	3FF4210C	3FF6410C	3FF6510C	
SPI_IN_ERR_EOF_DES_ADDR_REG		3FF43120	3FF42120	3FF64120	3FF65120	
SPI_IN_SUC_EOF_DES_ADDR_REG		3FF43124	3FF42124	3FF64124	3FF65124	
SPI_INLINK_DSCR_REG		3FF43128	3FF42128	3FF64128	3FF65128	
SPI_INLINK_DSCR_BF0_REG	一个	3FF4312C	3FF4212C	3FF6412C	3FF6512C	
SPI_INLINK_DSCR_BF1_REG		3FF43130	3FF42130	3FF64130	3FF65130	
SPI_OUT_EOF_BFR_DES_ADDR_REG	EOF	3FF43134	3FF42134	3FF64134	3FF65134	
SPI_OUT_EOF_DES_ADDR_REG	EOF	3FF43138	3FF42138	3FF64138	3FF65138	
SPI_OUTLINK_DSCR_REG		3FF4313C	3FF4213C	3FF6413C	3FF6513C	
SPI_OUTLINK_DSCR_BF0_REG	一个	3FF43140	3FF42140	3FF64140	3FF65140	
SPI_OUTLINK_DSCR_BF1_REG		3FF43144	3FF42144	3FF64144	3FF65144	
SPI_DMA_RSTATUS_REG	DMA	3FF43148	3FF42148	3FF64148	3FF65148	
SPI_DMA_TSTATUS_REG	DMA 写	3FF4314C	3FF4214C	3FF6414C	3FF6514C	
DMA 中断寄存器						
SPI_DMA_INT_RAW_REG	始中	3FF43114	3FF42114	3FF64114	3FF65114	
SPI_DMA_INT_ST_REG	屏蔽中	3FF43118	3FF42118	3FF64118	3FF65118	
SPI_DMA_INT_ENA_REG	中	3FF43110	3FF42110	3FF64110	3FF65110	/写
SPI_DMA_INT_CLR_REG	中	3FF4311C	3FF4211C	3FF6411C	3FF6511C	/写

7.8 寄存器

Register 7.1. SPI_CMD_REG (0x0)

(reserved)	SPI_USR	(reserved)	
31 … 19	18	17 … 0	
0 0 0 0 0 0 0 0 0 0 0 0 0	0	0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0	Reset

SPI_USR 1 时，一 SPI 操 ，操 被 。(/写)

Register 7.2. SPI_ADDR_REG (0x4)

31 … 0	
0x000000000	Reset

SPI_ADDR_REG 半 QSPI 。 长 32 bit， 包 ，SPI_SLV_WR_STATUS_REG 包 32 bit。SPI_USR_ADDR 被 1 时 。(/写)

Register 7.3. SPI_CTRL_REG (0x8)

(reserved)	SPI_WR_BIT_ORDER	SPI_RD_BIT_ORDER	SPI_FREAD_QIO	SPI_FREAD_DIO	(reserved)	SPI_WP	SPI_FREAD_QUAD	(reserved)	SPI_FREAD_DUAL	SPI_FASTRD_MODE	(reserved)	
31 … 27	26	25	24	23	22	21	20	19 … 15	14	13	12 … 0	
0 0 0 0 0	0	0	0	0	0	1	0	0 0 0 0 0	0	1	0 0 0 0 0 0 0 0 0 0 0 0 0	Reset

SPI_WR_BIT_ORDER 比 顺 ， 、 。1: ；0: 。(/写)

SPI_RD_BIT_ORDER 比 顺 ， 。1: ；0: 。(/写)

SPI_FREAD_QIO 4 。 QSPI 。(/写)

SPI_FREAD_DIO 2 。 QSPI 。(/写)

SPI_WP 时 WP 。1: ；2: 。 QSPI 。(/写)

SPI_FREAD_QUAD 4 。 QSPI 。(/写)

SPI_FREAD_DUAL 2 。 QSPI 。(/写)

SPI_FASTRD_MODE 保 。

Register 7.4. SPI_CTRL1_REG (0xC)

Register 7.6. SPI_CTRL2_REG (0x14)

SPI_CS_DELAY_NUM	SPI_CS_DELAY_MODE	SPI_MOSI_DELAY_NUM	SPI_MOSI_DELAY_MODE	SPI_MISO_DELAY_NUM	SPI_MISO_DELAY_MODE	SPI_CK_OUT_HIGH_MODE	reserved	SPI_HOLD_TIME	SPI_SETUP_TIME	
31 28	27 26	25 23	22 21	20 18	17 16	15 12	11 8	7 4	3 0	
0x00	0x0	0x0	0x0	0x0	0x0	0x00	0x00	0x01	0x01	Reset

SPI_CS_DELAY_NUM 保 。

SPI_CS_DELAY_MODE 保 。

SPI_MOSI_DELAY_NUM MOSI 时 。(/写)

SPI_MOSI_DELAY_MODE MOSI SPI 时 方 。(/写)

MOSI SPI_MOSI_DELAY_NUM 个 时 , :

0: 。

1: SPI_CK_OUT_EDGE SPI_CK_I_EDGE 被 1，MOSI 被 半个 期， 被 一个 期。

2: SPI_CK_OUT_EDGE SPI_CK_I_EDGE 被 1，MOSI 被 一个 期， 被 半个 期。

3: 一个 期。

SPI_MISO_DELAY_NUM MISO 时 。(/写)

SPI_MISO_DELAY_MODE MISO SPI 时 方 。(/写)

MISO SPI_MISO_DELAY_NUM 个 时 , :

0: 。

1: SPI_CK_OUT_EDGE SPI_CK_I_EDGE 被 1，MISO 被 半个 期， 被 一个 期。

2: SPI_CK_OUT_EDGE SPI_CK_I_EDGE 被 1，MISO 被 一个 期， 被 半个 期。

3: 一个 期。

SPI_HOLD_TIME CS 被 SPI 时 期 。 SPI_CS_HOLD 1 时 。(/写)

SPI_SETUP_TIME 时 CS 一个 SPI 时 时间。 SPI_CS_SETUP 1 时 。 半 QSPI 。(/写)

Register 7.7. SPI_CLOCK_REG (0x18)

SPI_CLK_EQU_SYSCLK	SPI_CLKDIV_PRE	SPI_CLKCNT_N	SPI_CLKCNT_H	SPI_CLKCNT_L	
31	30 18	17 12	11 6	5 0	
1	0 0 0 0 0 0 0 0 0 0 0 0 0	0x03	0x01	0x03	Reset

SPI_CLK_EQU_SYSCLK , 1: SPI 时 时 ; 0: SPI 时 时 可 。 , 0。(/写)

SPI_CLKDIV_PRE , SPI 时 。 SPI_CLK_EQU_SYSCLK 0 时 。 , 0。(/写)

SPI_CLKCNT_N , SPI 时 。 SPI_CLK_EQU_SYSCLK 0 时 。 , 0。(/写)

SPI_CLKCNT_H , SPI_CLKCNT_H $= \lfloor \frac{\text{SPI_CLKCNT_N}+1}{2} - 1 \rfloor$。 SPI_CLK_EQU_SYSCLK 0 时 。 , 0。(/写)

SPI_CLKCNT_L , SPI_CLKCNT_N。 SPI_CLK_EQU_SYSCLK 0 时 。 , 0。(/写)

Register 7.8. SPI_USER_REG (0x1C)

31	30	29	28	27	26	25	24	23	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	1	0	
SPI_USR_COMMAND	SPI_USR_ADDR	SPI_USR_DUMMY	SPI_USR_MISO	SPI_USR_MOSI	SPI_USR_DUMMY_IDLE	SPI_USR_MOSI_HIGHPART	SPI_USR_MISO_HIGHPART	(reserved)		SPI_SIO	SPI_FWRITE_QIO	SPI_FWRITE_DIO	SPI_FWRITE_QUAD	SPI_FWRITE_DUAL	SPI_WR_BYTE_ORDER	SPI_RD_BYTE_ORDER	(reserved)		SPI_CK_OUT_EDGE	SPI_CK_I_EDGE	SPI_CS_SETUP	SPI_CS_HOLD	(reserved)		SPI_DOUTDIN	
1	0	0	0	0	0	0	0	0 0 0 0 0 0 0		0	0	0	0	0	0	0	0	0	0	1	0	0	0 0 0		0	Reset

SPI_USR_COMMAND SPI 半 QSPI , SPI 操 。(/写)

SPI_USR_ADDR SPI 半 QSPI , SPI 操 。(/写)

SPI_USR_DUMMY SPI 半 QSPI , SPI 操 。(/写)

SPI_USR_MISO SPI 半 QSPI , SPI 操 。(/写)

SPI_USR_MOSI SPI 半 QSPI , SPI 操 写 。(/写)

SPI_USR_DUMMY_IDLE SPI 半 QSPI , 时 。(/写)

SPI_USR_MOSI_HIGHPART / SPI_W8_REG ~ SPI_W15_REG。(/写)

SPI_USR_MISO_HIGHPART / SPI_W8_REG ~ SPI_W15_REG。(/写)

SPI_SIO 三 半 。(/写)

SPI_FWRITE_QIO 保 。

SPI_FWRITE_DIO 保 。

SPI_FWRITE_QUAD 保 。

SPI_FWRITE_DUAL 保 。

SPI_WR_BYTE_ORDER 顺 , , 。1: ; 0: 。(/写)

SPI_RD_BYTE_ORDER 顺 , 。1: ; 0: 。(/写)

SPI_CK_OUT_EDGE SPI_MOSI_DELAY_MODE 设 MOSI 。 。(/写)

SPI_CK_I_EDGE SPI_CK_OUT_EDGE 。 SPI_MISO_DELAY_MODE 。 。(/写)

SPI_CS_SETUP CS 一个时 间 , SPI 时 期 单 。 1 时, 半 QSPI , 时 间 (SPI_SETUP_TIME + 1.5); , mode0 mode2 时 间 1.5, mode1 mode3 时 间 1。(/写)

SPI_CS_HOLD CS 间 一 , 时 间 SPI_HOLD_TIME。(/写)

SPI_DOUTDIN 。(/写)

Register 7.9. SPI_USER1_REG (0x20)

SPI_USR_ADDR_BITLEN	(reserved)	SPI_USR_DUMMY_CYCLELEN	
31 26	25 8	7 0	
23	0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0	7	Reset

SPI_USR_ADDR_BITLEN SPI 半 QSPI ，表 SPI 1，单 bit。 SPI_USR_ADDR 被 1 时 。( )

SPI_USR_DUMMY_CYCLELEN SPI 半 QSPI ，表 SPI spi_clk 时 期 1 。 SPI_USR_DUMMY 被 1 时 。(/写)

Register 7.10. SPI_USER2_REG (0x24)

SPI_USR_COMMAND_BITLEN	(reserved)	SPI_USR_COMMAND_VALUE	
31 28	27 16	15 0	
7	0 0 0 0 0 0 0 0 0 0 0 0	0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0	Reset

SPI_USR_COMMAND_BITLEN SPI 半 QSPI ，表 SPI 长 1，单 bit。 SPI_USR_COMMAND 被 1 时 。(/写)

SPI_USR_COMMAND_VALUE SPI 半 QSPI ，表 SPI 。 SPI_USR_COMMAND 被 1 时 。(/写)

Register 7.11. SPI_MOSI_DLEN_REG (0x28)

(reserved)	SPI_USR_MOSI_DBITLEN	
31 24	23 0	
0 0 0 0 0 0 0 0	0x0000000	Reset

SPI_USR_MOSI_DBITLEN 长 ， / 长 1，单 bit。 SPI_USR_MOSI 被 1 时 。 。(/写)

Register 7.12. SPI_MISO_DLEN_REG (0x2C)

(reserved)	SPI_USR_MISO_DBITLEN	
31 24	23 0	
0 0 0 0 0 0 0 0	0x0000000	Reset

SPI_USR_MISO_DBITLEN 长 , / 长 1, 单 bit。 SPI_USR_MISO 被 1 时 。 。(/写)

Register 7.13. SPI_SLV_WR_STATUS_REG (0x30)

31 0	
0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0	Reset

SPI_SLV_WR_STATUS_REG , 表 写 。 , 长 32 bit, 32 bit。(/写)

Register 7.14. SPI_PIN_REG (0x34)

(reserved)	SPI_CS_KEEP_ACTIVE	SPI_CK_IDLE_EDGE	(reserved)	SPI_MASTER_CK_SEL	(reserved)	SPI_MASTER_CS_POL	SPI_CK_DIS	(reserved)	SPI_CS2_DIS	SPI_CS1_DIS	SPI_CS0_DIS	
31	30	29	28 14	13 11	10 9	8 6	5	4 3	2	1	0	
0	0	0	0 0 0 0 0 0 0 0 0 0 0 0 0 0	0 0 0	0 0	0 0 0 0 0	0	0 0	1	1	0	Reset

SPI_CS_KEEP_ACTIVE 1 时, CS 保 。 。(/写)

SPI_CK_IDLE_EDGE SPI 时 。 。(/写)

1: 时, spi_clk 保 ;

0: 时, spi_clk 保 。

SPI_MASTER_CK_SEL 保 。

SPI_MASTER_CS_POL 保 。

SPI_CK_DIS 保 。

SPI_CS2_DIS SPI CS2 。1: 闭 CS2; 0: 启 CS2。(/写)

SPI_CS1_DIS SPI CS1 。1: 闭 CS1; 0: 启 CS1。(/写)

SPI_CS0_DIS SPI CS0 。1: 闭 CS0; 0: 启 CS0。(/写)

Register 7.15. SPI_SLAVE_REG (0x38)

31	30	29	28	27	26 23	22 20	19 17	16 12	11 10	9	8	7	6	5	4	3	2	1	0	
SPI_SYNC_RESET	SPI_SLAVE_MODE	SPI_SLV_WR_RD_BUF_EN	SPI_SLV_WR_RD_STA_EN	SPI_SLV_CMD_DEFINE	SPI_TRANS_CNT	SPI_SLV_LAST_STATE	SPI_SLV_LAST_COMMAND	(reserved)	SPI_CS_I_MODE	SPI_TRANS_INTEN	SPI_SLV_WR_STA_INTEN	SPI_SLV_RD_STA_INTEN	SPI_SLV_WR_BUF_INTEN	SPI_SLV_RD_BUF_INTEN	SPI_TRANS_DONE	SPI_SLV_WR_STA_DONE	SPI_SLV_RD_STA_DONE	SPI_SLV_WR_BUF_DONE	SPI_SLV_RD_BUF_DONE	
0	0	0	0	0	0 0 0 0	0 0 0	0 0 0	0 0 0 0 0	0 0	0	0	0	0	0	0	0	0	0	0	Reset

SPI_SYNC_RESET SPI 时 , CS 。(/写)

SPI_SLAVE_MODE 设 SPI 设备 。(/写)

1: ;

0: 。

SPI_SLV_WR_RD_BUF_EN 1, 写 操 。 半 。(/写)

SPI_SLV_WR_RD_STA_EN 1, 写 操 。 半 。(/写)

SPI_SLV_CMD_DEFINE 保 。

SPI_TRANS_CNT 操 。( )

SPI_SLV_LAST_STATE , 包 SPI 。( )

SPI_SLV_LAST_COMMAND 保 。

SPI_CS_I_MODE 保 。

SPI_TRANS_INTEN SPI_TRANS_DONE_INT 中 。(/写)

SPI_SLV_WR_STA_INTEN SPI_SLV_WR_STA_INT 中 。(/写)

SPI_SLV_RD_STA_INTEN SPI_SLV_RD_STA_INT 中 。(/写)

SPI_SLV_WR_BUF_INTEN SPI_SLV_WR_BUF_INT 中 。(/写)

SPI_SLV_RD_BUF_INTEN SPI_SLV_RD_BUF_INT 中 。(/写)

SPI_TRANS_DONE SPI_TRANS_DONE_INT 中 。 1, 。(/写)

SPI_SLV_WR_STA_DONE SPI_SLV_WR_STA_INT 中 。 标 1, 。 半 。(/写)

SPI_SLV_RD_STA_DONE SPI_SLV_RD_STA_INT 中 。 标 1, 。 半 。(/写)

SPI_SLV_WR_BUF_DONE SPI_SLV_WR_BUF_INT 中 。 标 1, 。 半 。(/写)

SPI_SLV_RD_BUF_DONE SPI_SLV_RD_BUF_INT 中 。 标 1, 。 半 。(/写)

Register 7.16. SPI_SLAVE1_REG (0x3C)

SPI_SLV_STATUS_BITLEN	SPI_SLV_STATUS_FAST_EN	SPI_SLV_STATUS_READBACK	(reserved)	SPI_SLV_RD_ADDR_BITLEN	SPI_SLV_WR_ADDR_BITLEN	SPI_SLV_WRSTA_DUMMY_EN	SPI_SLV_RDSTA_DUMMY_EN	SPI_SLV_WRBUF_DUMMY_EN	SPI_SLV_RDBUF_DUMMY_EN	
31 27	26	25	24 16	15 10	9 4	3	2	1	0	
0 0 0 0 0	0	1	0 0 0 0 0 0 0 0 0	0x00	0x00	0	0	0	0	Reset

SPI_SLV_STATUS_BITLEN 写 长 。 半 。(/写)

SPI_SLV_STATUS_FAST_EN 保 。

SPI_SLV_STATUS_READBACK 保 。

SPI_SLV_RD_ADDR_BITLEN 表 长 1, 单 bit。 半 。(/写)

SPI_SLV_WR_ADDR_BITLEN 表 写 长 1, 单 bit。 半 。(/写)

SPI_SLV_WRSTA_DUMMY_EN 写 。 半 。(/写)

SPI_SLV_RDSTA_DUMMY_EN 。 半 。(/写)

SPI_SLV_WRBUF_DUMMY_EN 写 操 。 半 。(/写)

SPI_SLV_RDBUF_DUMMY_EN 操 。 半 。(/写)

Register 7.17. SPI_SLAVE2_REG (0x40)

SPI_SLV_WRBUF_DUMMY_CYCLELEN (31–24)	SPI_SLV_RDBUF_DUMMY_CYCLELEN (23–16)	SPI_SLV_WRSTA_DUMMY_CYCLELEN (15–8)	SPI_SLV_RDSTA_DUMMY_CYCLELEN (7–0)	
0 0 0 0 0 0 0 0	0x000	0x000	0x000	Reset

SPI_SLV_WRBUF_DUMMY_CYCLELEN 写 操 SPI 时 期 1。 SPI_SLV_WRBUF_DUMMY_EN 被 1 时 。 半 。(/写)

SPI_SLV_RDBUF_DUMMY_CYCLELEN 操 SPI 时 期 1。 SPI_SLV_RDBUF_DUMMY_EN 被 1 时 。 半 。(/写)

SPI_SLV_WRSTA_DUMMY_CYCLELEN 写 操 SPI 时 期 1。 SPI_SLV_WRSTA_DUMMY_EN 被 1 时 。 半 。(/写)

SPI_SLV_RDSTA_DUMMY_CYCLELEN 操 SPI 时 期 1。 SPI_SLV_RDSTA_DUMMY_EN 被 1 时 。 半 。(/写)

Register 7.18. SPI_SLAVE3_REG (0x44)

SPI_SLV_WRSTA_CMD_VALUE (31–24)	SPI_SLV_RDSTA_CMD_VALUE (23–16)	SPI_SLV_WRBUF_CMD_VALUE (15–8)	SPI_SLV_RDBUF_CMD_VALUE (7–0)	
0 0 0 0 0 0 0 0	0 0 0 0 0 0 0 0	0 0 0 0 0 0 0 0	0 0 0 0 0 0 0 0	Reset

SPI_SLV_WRSTA_CMD_VALUE 保 。

SPI_SLV_RDSTA_CMD_VALUE 保 。

SPI_SLV_WRBUF_CMD_VALUE 保 。

SPI_SLV_RDBUF_CMD_VALUE 保 。

Register 7.19. SPI_SLV_WRBUF_DLEN_REG (0x48)

(reserved)	SPI_SLV_WRBUF_DBITLEN	
31 … 24	23 … 0	
0 0 0 0 0 0 0 0	0x0000000	Reset

SPI_SLV_WRBUF_DBITLEN 表 写 长 1，单 bit。 半 。

Register 7.20. SPI_SLV_RDBUF_DLEN_REG (0x4C)

(reserved)	SPI_SLV_RDBUF_DBITLEN	
31 … 24	23 … 0	
0 0 0 0 0 0 0 0	0x0000000	Reset

SPI_SLV_RDBUF_DBITLEN 表 长 1，单 bit。 半 。(/写)

Register 7.21. SPI_SLV_RD_BIT_REG (0x64)

(reserved)	SPI_SLV_RDATA_BIT	
31 … 24	23 … 0	
0 0 0 0 0 0 0 0	0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0	Reset

SPI_SLV_RDATA_BIT 长 1。 半 。(/写)

Register 7.22. SPI_W*n*_REG (*n*：0-15) (0x80+4***n*)

31 … 0	
0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0	Reset

SPI_W*n*_REG 。(/写)

Register 7.23. SPI_TX_CRC_REG (0xC0)

31																															0	
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	Reset

SPI_TX_CRC_REG 保 。

Register 7.24. SPI_EXT2_REG (0xF8)

(reserved)																															SPI_ST			
31																														3	2			0	
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	Reset

SPI_ST SPI 。( )

0: ;

1: 备 ;

2: ;

3: ;

4: ;

5: 写 ;

6: ;

7: 。

Register 7.25. SPI_DMA_CONF_REG (0x100)

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0	
(reserved)															SPI_DMA_CONTINUE	SPI_DMA_TX_STOP	SPI_DMA_RX_STOP	(reserved)	SPI_OUT_DATA_BURST_EN	SPI_INDSCR_BURST_EN	SPI_OUTDSCR_BURST_EN	SPI_OUT_EOF_MODE	(reserved)			SPI_AHBM_RST	SPI_AHBM_FIFO_RST	SPI_OUT_RST	SPI_IN_RST	(reserved)		
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	0	0	0	0	0	0	0	0	Reset

SPI_DMA_CONTINUE SPI DMA TX/RX 。(/写)

SPI_DMA_TX_STOP TX/RX , 1 。(/写)

SPI_DMA_RX_STOP TX/RX , 1 。(/写)

SPI_OUT_DATA_BURST_EN DMA burst 。(/写)

SPI_INDSCR_BURST_EN DMA burst 表 。(/写)

SPI_OUTDSCR_BURST_EN DMA burst 表 。(/写)

SPI_OUT_EOF_MODE DMA EOF 标 。(/写)

1: DMA FIFO 中 时，产 EOF 标 ;

0: DMA 写 FIFO 时，产 EOF 标 。

SPI_AHBM_RST SPI DMA AHB 。(/写)

SPI_AHBM_FIFO_RST SPI DMA AHB FIFO 。(/写)

SPI_OUT_RST DMA FIFO 。(/写)

SPI_IN_RST DMA FIFO 。(/写)

Register 7.26. SPI_DMA_OUT_LINK_REG (0x104)

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0	
(reserved)	SPI_OUTLINK_RESTART	SPI_OUTLINK_START	SPI_OUTLINK_STOP	(reserved)								SPI_OUTLINK_ADDR																				
0	0	0	0	0	0	0	0	0	0	0	0	0x000000																				Reset

SPI_OUTLINK_RESTART 1 表 。(/写)

SPI_OUTLINK_START 1 始 表 。(/写)

SPI_OUTLINK_STOP 1 表 。(/写)

SPI_OUTLINK_ADDR 一个 表 。(/写)

Register 7.27. SPI_DMA_IN_LINK_REG (0x108)

(reserved)	SPI_INLINK_RESTART	SPI_INLINK_START	SPI_INLINK_STOP	(reserved)	SPI_INLINK_AUTO_RET	SPI_INLINK_ADDR	
31	30	29	28	27 … 21	20	19 … 0	
0	0	0	0	0 0 0 0 0 0 0	0	0x000000	Reset

SPI_INLINK_RESTART 1 表 。(/写)

SPI_INLINK_START 1 始 表 。(/写)

SPI_INLINK_STOP 1 表 。(/写)

SPI_INLINK_AUTO_RET 1 时， 包 ， 表 一个 。(/写)

SPI_INLINK_ADDR 一个 表 。(/写)

Register 7.28. SPI_DMA_STATUS_REG (0x10C)

(reserved)	SPI_DMA_TX_EN	SPI_DMA_RX_EN	
31 … 2	1	0	
0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0	0	0	Reset

SPI_DMA_TX_EN SPI DMA 。( )

SPI_DMA_RX_EN SPI DMA 。( )

Register 7.29. SPI_DMA_INT_ENA_REG (0x110)

31																					9	8	7	6	5	4	3	2	1	0	
(reserved)																						SPI_OUT_TOTAL_EOF_INT_ENA	SPI_OUT_EOF_INT_ENA	SPI_OUT_DONE_INT_ENA	SPI_IN_SUC_EOF_INT_ENA	SPI_IN_ERR_EOF_INT_ENA	SPI_IN_DONE_INT_ENA	SPI_INLINK_DSCR_ERROR_INT_ENA	SPI_OUTLINK_DSCR_ERROR_INT_ENA	SPI_INLINK_DSCR_EMPTY_INT_ENA	
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	Reset

SPI_OUT_TOTAL_EOF_INT_ENA SPI_OUT_TOTAL_EOF_INT 中 。(/写)

SPI_OUT_EOF_INT_ENA SPI_OUT_EOF_INT 中 。(/写)

SPI_OUT_DONE_INT_ENA SPI_OUT_DONE_INT 中 。(/写)

SPI_IN_SUC_EOF_INT_ENA SPI_IN_SUC_EOF_INT 中 。(/写)

SPI_IN_ERR_EOF_INT_ENA SPI_IN_ERR_EOF_INT 中 。(/写)

SPI_IN_DONE_INT_ENA SPI_IN_DONE_INT 中 。(/写)

SPI_INLINK_DSCR_ERROR_INT_ENA SPI_INLINK_DSCR_ERROR_INT 中 。(/写)

SPI_OUTLINK_DSCR_ERROR_INT_ENA SPI_OUTLINK_DSCR_ERROR_INT 中 。(/写)

SPI_INLINK_DSCR_EMPTY_INT_ENA SPI_INLINK_DSCR_EMPTY_INT 中 。(/写)

Register 7.30. SPI_DMA_INT_RAW_REG (0x114)

31																						9	8	7	6	5	4	3	2	1	0	
																						(reserved)	SPI_OUT_TOTAL_EOF_INT_RAW	SPI_OUT_EOF_INT_RAW	SPI_OUT_DONE_INT_RAW	SPI_IN_SUC_EOF_INT_RAW	SPI_IN_ERR_EOF_INT_RAW	SPI_IN_DONE_INT_RAW	SPI_INLINK_DSCR_ERROR_INT_RAW	SPI_OUTLINK_DSCR_ERROR_INT_RAW	SPI_INLINK_DSCR_EMPTY_INT_RAW	
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	Reset

SPI_OUT_TOTAL_EOF_INT_RAW SPI_OUT_TOTAL_EOF_INT 始中 。( )

SPI_OUT_EOF_INT_RAW SPI_OUT_EOF_INT 始中 。( )

SPI_OUT_DONE_INT_RAW SPI_OUT_DONE_INT 始中 。( )

SPI_IN_SUC_EOF_INT_RAW SPI_IN_SUC_EOF_INT 始中 。( )

SPI_IN_ERR_EOF_INT_RAW SPI_IN_ERR_EOF_INT 始中 。( )

SPI_IN_DONE_INT_RAW SPI_IN_DONE_INT 始中 。( )

SPI_INLINK_DSCR_ERROR_INT_RAW SPI_INLINK_DSCR_ERROR_INT 始中 。( )

SPI_OUTLINK_DSCR_ERROR_INT_RAW SPI_OUTLINK_DSCR_ERROR_INT 始中 。( )

SPI_INLINK_DSCR_EMPTY_INT_RAW SPI_INLINK_DSCR_EMPTY_INT 始中 。( )

Register 7.31. SPI_DMA_INT_ST_REG (0x118)

(reserved)	SPI_OUT_TOTAL_EOF_INT_ST	SPI_OUT_EOF_INT_ST	SPI_OUT_DONE_INT_ST	SPI_IN_SUC_EOF_INT_ST	SPI_IN_ERR_EOF_INT_ST	SPI_IN_DONE_INT_ST	SPI_INLINK_DSCR_ERROR_INT_ST	SPI_OUTLINK_DSCR_ERROR_INT_ST	SPI_INLINK_DSCR_EMPTY_INT_ST	
31 … 9	8	7	6	5	4	3	2	1	0	
0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0	0	0	0	0	0	0	0	0	0	Reset

SPI_OUT_TOTAL_EOF_INT_ST SPI_OUT_TOTAL_EOF_INT 屏蔽中 。( )

SPI_OUT_EOF_INT_ST SPI_OUT_EOF_INT 屏蔽中 。( )

SPI_OUT_DONE_INT_ST SPI_OUT_DONE_INT 屏蔽中 。( )

SPI_IN_SUC_EOF_INT_ST SPI_IN_SUC_EOF_INT 屏蔽中 。( )

SPI_IN_ERR_EOF_INT_ST SPI_IN_ERR_EOF_INT 屏蔽中 。( )

SPI_IN_DONE_INT_ST SPI_IN_DONE_INT 屏蔽中 。( )

SPI_INLINK_DSCR_ERROR_INT_ST SPI_INLINK_DSCR_ERROR_INT 屏蔽中 。( )

SPI_OUTLINK_DSCR_ERROR_INT_ST SPI_OUTLINK_DSCR_ERROR_INT 屏蔽中 。( )

SPI_INLINK_DSCR_EMPTY_INT_ST SPI_INLINK_DSCR_EMPTY_INT 屏蔽中 。( )

Register 7.32. SPI_DMA_INT_CLR_REG (0x11C)

31																						9	8	7	6	5	4	3	2	1	0	
																						(reserved)	SPI_OUT_TOTAL_EOF_INT_CLR	SPI_OUT_EOF_INT_CLR	SPI_OUT_DONE_INT_CLR	SPI_IN_SUC_EOF_INT_CLR	SPI_IN_ERR_EOF_INT_CLR	SPI_IN_DONE_INT_CLR	SPI_INLINK_DSCR_ERROR_INT_CLR	SPI_OUTLINK_DSCR_ERROR_INT_CLR	SPI_INLINK_DSCR_EMPTY_INT_CLR	
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	Reset

SPI_OUT_TOTAL_EOF_INT_CLR 1 SPI_OUT_TOTAL_EOF_INT 中 。(/写)

SPI_OUT_EOF_INT_CLR 1 SPI_OUT_EOF_INT 中 。(/写)

SPI_OUT_DONE_INT_CLR 1 SPI_OUT_DONE_INT 中 。(/写)

SPI_IN_SUC_EOF_INT_CLR 1 SPI_IN_SUC_EOF_INT 中 。(/写)

SPI_IN_ERR_EOF_INT_CLR 1 SPI_IN_ERR_EOF_INT 中 。(/写)

SPI_IN_DONE_INT_CLR 1 SPI_IN_DONE_INT 中 。(/写)

SPI_INLINK_DSCR_ERROR_INT_CLR 1 SPI_INLINK_DSCR_ERROR_INT 中 。(/写)

SPI_OUTLINK_DSCR_ERROR_INT_CLR 1 SPI_OUTLINK_DSCR_ERROR_INT 中 。(/写)

SPI_INLINK_DSCR_EMPTY_INT_CLR 1 SPI_INLINK_DSCR_EMPTY_INT 中 。(/写)

Register 7.33. SPI_IN_ERR_EOF_DES_ADDR_REG (0x120)

31																															0	
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	Reset

SPI_IN_ERR_EOF_DES_ADDR_REG SPI DMA 时 表 。( )

Register 7.34. SPI_IN_SUC_EOF_DES_ADDR_REG (0x124)

31																															0	
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	Reset

SPI_IN_SUC_EOF_DES_ADDR_REG SPI DMA 时 一个 表 。( )

Register 7.35. SPI_INLINK_DSCR_REG (0x128)

31																															0	
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	Reset

SPI_INLINK_DSCR_REG 表 。( )

Register 7.36. SPI_INLINK_DSCR_BF0_REG (0x12C)

31																															0	
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	Reset

SPI_INLINK_DSCR_BF0_REG 一个 表 。( )

Register 7.37. SPI_INLINK_DSCR_BF1_REG (0x130)

31																															0	
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	Reset

SPI_INLINK_DSCR_BF1_REG 一个 表 。( )

Register 7.38. SPI_OUT_EOF_BFR_DES_ADDR_REG (0x134)

31																															0	
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	Reset

SPI_OUT_EOF_BFR_DES_ADDR_REG EOF 表 。( )

Register 7.39. SPI_OUT_EOF_DES_ADDR_REG (0x138)

31																															0	
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	Reset

SPI_OUT_EOF_DES_ADDR_REG SPI DMA 时 一个 表 。( )

Register 7.40. SPI_OUTLINK_DSCR_REG (0x13C)

31																															0	
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	Reset

SPI_OUTLINK_DSCR_REG 表 。( )

Register 7.41. SPI_OUTLINK_DSCR_BF0_REG (0x140)

31																															0	
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	Reset

SPI_OUTLINK_DSCR_BF0_REG 一个 表 。( )

Register 7.42. SPI_OUTLINK_DSCR_BF1_REG (0x144)

31																															0	
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	Reset

SPI_OUTLINK_DSCR_BF1_REG 一个 表 。( )

Register 7.43. SPI_DMA_RSTATUS_REG (0x148)

TX_FIFO_EMPTY TX_FIFO_FULL		(reserved)	TX_DES_ADDRESS	
31	30	29 … 20	19 … 0	
0	0	0 0 0 0 0 0 0 0 0 0	0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0	Reset

TX_FIFO_EMPTY SPI DMA FIFO 。( )

TX_FIFO_FULL SPI DMA FIFO 。( )

TX_DES_ADDRESS SPI DMA 。( )

Register 7.44. SPI_DMA_TSTATUS_REG (0x14C)

RX_FIFO_EMPTY RX_FIFO_FULL		(reserved)	RX_DES_ADDRESS	
31	30	29 … 20	19 … 0	
0	0	0 0 0 0 0 0 0 0 0 0	0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0	Reset

RX_FIFO_EMPTY SPI DMA FIFO 。( )

RX_FIFO_FULL SPI DMA FIFO 。( )

RX_DES_ADDRESS SPI DMA 。( )

8 SDIO 从机控制器

8.1 概述

ESP32 (SDIO) 设备 , SDIO V2.0 。 可 SDIO ESP32。

(Host) 可 SDIO , DMA 设备上 , 保 时 浪 。

8.2 主要特性

- SDIO V2.0
- SDIO SPI，1-bit 4-bit
- 0 ~ 50 MHz 时
- 可 采 时 时
- 设
- SDIO 上 , SDIO 上
- 512
- Host Slave 间 中 可 中 扩
- DMA

8.3 功能描述

8.3.1 SDIO Slave 功能块图

SDIO Slave 8-1 。

图 8-1. SDIO Slave 功能块图

(Host System) 表 一 SDIO V2.0 Host 设备。Host 标 SDIO SDIO Slave ESP32 。SDIO 设备 SDIO 并 DMA 操 Host , (AHB) 上 ，并 不 CPU 参 。

8.3.2 SDIO 总线上的数据发送和接收

Host Slave 间 SDIO bus I/O Function1 。 Host 按 SDIO Slave I/O Function1 , 可 。

包 单 ， 一个 包。 ， 单 （ 见 SDIO ） 包 。 单 ，Host Slave SDIO 上 个 。Slave 包时 ， 包时 。

Slave SDIO 上 ， 0x1F800 ， 始 0x1F800 Packet_length（Packet_length 单 ）。 SDIO 上 8-2 ：

图 8-2. SDIO 总线上数据传输

IO_RW_EXTENDED (CMD53) ， 8-3， 部 查 SDIO 。

图 8-3. CMD53 内容

8.3.3 寄存器访问

方便 Host Slave 间 ，Host 可 SDIO bus I/O Function1 Slave 中 。 SLC0HOST_TOKEN_RDATA SLCHOST_INF_ST 。Host 时 CMD52 CMD53 中 设 10 。Host 可 CMD53 时 个 ， 。

SLCHOST_CONF_W0_REG SLCHOST_CONF_W15_REG 54 个 ，Host Slave 可 改 方便 Host Slave 间 。

8.3.4 DMA

SDIO Slave 一个 DMA RAM 。 8-1 ，DMA AHB RAM。 DMA 表 RAM。 个 表 3 个 (word) ， 8-4 。

Owner	Eof	Reserved	Len

图 8-4. SDIO Slave DMA 链表结构

- Owner： 表 buffer 操 。

  0： 操 CPU

1： 操 DMA

- Eof： 标 ，表 表 包 一个 表。
- Length：Buffer 中 ， buffer 中 被 。
- Size：Buffer 。
- Buffer Address Pointer：Buffer 。
- Next Descriptor Address： 一个 表 。 表 一个 表时，Eof 1， 0。

Slave 表 8-5 ：

图 8-5. 链表串

8.3.5 包的发送和接收流程

Host Slave 间 包 按 。

8.3.5.1 Slave 向 Host 发送包

包 Slave ， 中 Host（中 参 SDIO ）。Host Slave SDIO 。 个 8-6 ：

图 8-6. Slave 向 Host 发送包的流程

Host Slave SLC0HOST_INT SLCHOST_PKT_LEN 个 。

- SLC0HOST_INT：中 ， 中 bit SLC0_RX_NEW_PACKET_INT_ST 1 表 中 Slave 包。
- SLCHOST_PKT_LEN：Slave 包长 ，Host 本 上 可 本 包 长 。

CPU 启 DMA 表 一个 表 20 写 SLC0RX_LINK SLC0_RXLINK_ADDR 中， SLC0RX_LINK SLC0_RXLINK_START 启 DMA。 DMA 。 DMA CPU 中 ， 时 CPU 可 buffer。

8.3.5.2 Slave 从 Host 接收包

包 Host ，Slave DMA 并 RAM 中， 中 CPU 。 个 8-7 ：

图 8-7. Slave 从 Host 接收包的流程

Host SLC0HOST_TOKEN_RDATA Slave 侧可 包 buffer 。Slave 侧 CPU 备 包 DMA 表 个 。

SLC0HOST_TOKEN_RDATA 中 HOSTREG_SLC0_TOKEN1 可 buffer 。

Host HOSTREG_SLC0_TOKEN1 buffer 可 buffer 。

buffer 不 时，Host 不 个 buffer 。

保 buffer 包，Slave CPU 必 不 表上 buffer。 8-8 。

图 8-8. Slave CPU 挂载 buffer 的流程

CPU 空 buffer 表 并 DMA 表 。

CPU DMA 表 ，可 SLC0TX_LINK SLC0_TXLINK_RESTART 1 。 ，CPU 空 启 DMA 包时 SLC0TX_LINK SLC0_TXLINK_START 1。

，CPU SLC0TOKEN1 可 buffer。

8.3.6 SDIO 总线时序

Host Slave 间 PCB ， 。 保 上时 ，Slave 时 SDIO 采 时 时 。

Host 变化 时 时 上升 时，Slave 时 采 。采 时 8-9 ：

图 8-9. 采样时序图

Slave SLCHOST_CONF 中 FRC_POS_SAMP FRC_NEG_SAMP 上升 采 。 个 5 bit，5 个 bit 别 CMD 4 个 DATA (0-3)。 FRC_POS_SAMP Slave 时 上升 上 采 。 FRC_NEG_SAMP Slave 时 上 采 。

Slave 可 时 上升 ， 不 。时 8-10 ：

图 8-10. 输出时序图

Slave SLCHOST_CONF 中 FRC_SDIO20 FRC_SDIO11 上升 。个 5 bit，5 个 bit 别 CMD 4 个 DATA (0-3)。 FRC_SDIO20 Slave 时 上升 上 。 FRC_NEG_SAMP Slave 时 。

8.3.7 中断

Host Slave 间可 中 中 。Host Slave 8 个中 可 中 。 中 中 (中)。中 ， 一 产 一 中 ，不 操 。

8.3.7.1 Host 侧中断

- *SLC0HOST_SLC0_RX_NEW_PACKET_INT* Slave 包中
- *SLC0HOST_SLC0_TX_OVF_INT* Slave buffer 中
- *SLC0HOST_SLC0_RX_UDF_INT* Slave buffer 中
- *SLC0HOST_SLC0_TOHOST_BITn_INT* (*n*: 0 ~ 7) Slave 中 Host

8.3.7.2 Slave 侧中断

- *SLC0INT_SLC0_RX_DSCR_ERR_INT* Slave 中
- *SLC0INT_SLC0_TX_DSCR_ERR_INT* Slave 中
- *SLC0INT_SLC0_RX_EOF_INT* Slave 操 中
- *SLC0INT_SLC0_RX_DONE_INT* 单 个 buffer Slave 中
- *SLC0INT_SLC0_TX_SUC_EOF_INT* Slave 操 中
- *SLC0INT_SLC0_TX_DONE_INT* A 单 个 buffer Slave 操 时 中
- *SLC0INT_SLC0_TX_OVF_INT* Slave buffer 中
- *SLC0INT_SLC0_RX_UDF_INT* Slave buffer 中
- *SLC0INT_SLC0_TX_START_INT* Slave 操 始中
- *SLC0INT_SLC0_RX_START_INT* Slave 操 始中
- *SLC0INT_SLC_FRHOST_BITn_INT* (*n*: 0 ~ 7) Host 中 Slave

8.4 寄存器列表

SDIO DMA (SLC) 配置寄存器			
SLCCONF0_REG	SLCCONF0_SLC configuration	0x3FF58000	/写
SLC0RX_LINK_REG	Transmitting linked list configuration	0x3FF5803C	/写
SLC0TX_LINK_REG	Receiving linked list configuration	0x3FF58040	/写
SLCINTVEC_TOHOST_REG	Interrupt sector for Slave to interrupt Host	0x3FF5804C	写
SLC0TOKEN1_REG	Number of receiving buffer	0x3FF58054	写
SLCCONF1_REG	Control register	0x3FF58060	/写
SLC_RX_DSCR_CONF_REG	DMA transmission configuration	0x3FF58098	/写
SLC0_LEN_CONF_REG	Length control of the transmitting packets	0x3FF580E4	/写
SLC0_LENGTH_REG	Length of the transmitting packets	0x3FF580E8	/写
中断寄存器			
SLC0INT_RAW_REG	Raw interrupt status	0x3FF58004	
SLC0INT_ST_REG	Interrupt status	0x3FF58008	
SLC0INT_ENA_REG	Interrupt enable	0x3FF5800C	/写
SLC0INT_CLR_REG	Interrupt clear	0x3FF58010	写

SDIO SLC Host 寄存器			
SLC0HOST_TOKEN_RDATA	The accumulated number of Slave's receiving buffers	0x3FF55044	
SLCHOST_PKT_LEN_REG	Length of the transmitting packets	0x3FF55060	
SLCHOST_CONF_W0_REG	Host and Slave communication register0	0x3FF5506C	/写
SLCHOST_CONF_W1_REG	Host and Slave communication register1	0x3FF55070	/写
SLCHOST_CONF_W2_REG	Host and Slave communication register2	0x3FF55074	/写
SLCHOST_CONF_W3_REG	Host and Slave communication register3	0x3FF55078	/写
SLCHOST_CONF_W4_REG	Host and Slave communication register4	0x3FF5507C	/写
SLCHOST_CONF_W6_REG	Host and Slave communication register6	0x3FF55088	/写
SLCHOST_CONF_W7_REG	Interrupt vector for Host to interrupt Slave	0x3FF5508C	写
SLCHOST_CONF_W8_REG	Host and Slave communication register8	0x3FF5509C	/写
SLCHOST_CONF_W9_REG	Host and Slave communication register9	0x3FF550A0	/写
SLCHOST_CONF_W10_REG	Host and Slave communication register10	0x3FF550A4	/写
SLCHOST_CONF_W11_REG	Host and Slave communication register11	0x3FF550A8	/写
SLCHOST_CONF_W12_REG	Host and Slave communication register12	0x3FF550AC	/写
SLCHOST_CONF_W13_REG	Host and Slave communication register13	0x3FF550B0	/写
SLCHOST_CONF_W14_REG	Host and Slave communication register14	0x3FF550B4	/写
SLCHOST_CONF_W15_REG	Host and Slave communication register15	0x3FF550B8	/写
SLCHOST_CONF_REG	Edge configuration	0x3FF551F0	/写
中断寄存器			
SLC0HOST_INT_RAW_REG	Raw interrupt	0x3FF55000	
SLC0HOST_INT_ST_REG	Masked interrupt status	0x3FF55058	
SLC0HOST_INT_CLR_REG	Interrupt clear	0x3FF550D4	写
SLC0HOST_FUNC1_INT_ENA_REG	Interrupt enable	0x3FF550DC	/写

SDIO HINF 寄存器			
HINF_CFG_DATA1_REG	SDIO specification configuration	0x3FF4B004	/写

8.5 SLC 寄存器

SDIO 一个 始 0x3FF5_8000。

Register 8.1. SLCCONF0_REG (0x0)

(reserved)	SLCCONF0_SLC0_TOKEN_AUTO_CLR	(reserved)	SLCCONF0_SLC0_RX_AUTO_WRBACK	SLCCONF0_SLC0_RX_LOOP_TEST	SLCCONF0_SLC0_TX_LOOP_TEST	(reserved)	SLCCONF0_SLC0_RX_RST	SLCCONF0_SLC0_TX_RST	
31 … 15	14	13 … 7	6	5	4	3 … 2	1	0	
0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0	1	0 0 0 0 0 0 0	0	1	1	0 0	0	0	Reset

SLCCONF0_SLC0_TOKEN_AUTO_CLR 始化 0。 已 (/写)

SLCCONF0_SLC0_RX_AUTO_WRBACK 时 buffer 表上已写 owner bit。(/写)

SLCCONF0_SLC0_RX_LOOP_TEST Slave buffer 包 。 , 不 已 表中 owner bit。(/写)

SLCCONF0_SLC0_TX_LOOP_TEST Slave buffer 包 。 , 不 已 表中 owner bit。(/写)

SLCCONF0_SLC0_RX_RST 1 FSM。(/写)

SLCCONF0_SLC0_TX_RST 1 FSM。(/写)

Register 8.2. SLC0INT_RAW_REG (0x4)

31–27	26–21	20	19	18	17	16	15	14	13–12	11	10	9	8	7	6	5	4	3	2	1	0	
(reserved)	(reserved)	SLC0INT_SLC0_RX_DSCR_ERR_INT_RAW	SLC0INT_SLC0_TX_DSCR_ERR_INT_RAW	(reserved)	SLC0INT_SLC0_RX_EOF_INT_RAW	SLC0INT_SLC0_RX_DONE_INT_RAW	SLC0INT_SLC0_TX_SUC_EOF_INT_RAW	SLC0INT_SLC0_TX_DONE_INT_RAW	(reserved)	SLC0INT_SLC0_TX_OVF_INT_RAW	SLC0INT_SLC0_RX_UDF_INT_RAW	SLC0INT_SLC0_TX_START_INT_RAW	SLC0INT_SLC0_RX_START_INT_RAW	SLC0INT_SLC_FRHOST_BIT7_INT_RAW	SLC0INT_SLC_FRHOST_BIT6_INT_RAW	SLC0INT_SLC_FRHOST_BIT5_INT_RAW	SLC0INT_SLC_FRHOST_BIT4_INT_RAW	SLC0INT_SLC_FRHOST_BIT3_INT_RAW	SLC0INT_SLC_FRHOST_BIT2_INT_RAW	SLC0INT_SLC_FRHOST_BIT1_INT_RAW	SLC0INT_SLC_FRHOST_BIT0_INT_RAW	
0x00	0 0 0 0 0 0	0	0	0	0	0	0	0	0 0	0	0	0	0	0	0	0	0	0	0	0	0	Reset

SLC0INT_SLC0_RX_DSCR_ERR_INT_RAW Slave 始中 。( )

SLC0INT_SLC0_TX_DSCR_ERR_INT_RAW Slave 始中 。( )

SLC0INT_SLC0_RX_EOF_INT_RAW Slave 操 中 标 。( )

SLC0INT_SLC0_RX_DONE_INT_RAW 单个 buffer Slave 始中 。( )

SLC0INT_SLC0_TX_SUC_EOF_INT_RAW Slave 操 始中 。( )

SLC0INT_SLC0_TX_DONE_INT_RAW 单个 buffer Slave 操 时 始中 。( )

SLC0INT_SLC0_TX_OVF_INT_RAW Slave buffer 始中 。( )

SLC0INT_SLC0_RX_UDF_INT_RAW Slave buffer 始中 。( )

SLC0INT_SLC0_TX_START_INT_RAW Slave 始中 始中 。( )

SLC0INT_SLC0_RX_START_INT_RAW Slave 始中 始中 。( )

SLC0INT_SLC_FRHOST_BIT7_INT_RAW Host 中 Slave 中 标 7。( )

SLC0INT_SLC_FRHOST_BIT6_INT_RAW Host 中 Slave 中 标 6。( )

SLC0INT_SLC_FRHOST_BIT5_INT_RAW Host 中 Slave 中 标 5。( )

SLC0INT_SLC_FRHOST_BIT4_INT_RAW Host 中 Slave 中 标 4。( )

SLC0INT_SLC_FRHOST_BIT3_INT_RAW Host 中 Slave 中 标 3。( )

SLC0INT_SLC_FRHOST_BIT2_INT_RAW Host 中 Slave 中 标 2。( )

SLC0INT_SLC_FRHOST_BIT1_INT_RAW Host 中 Slave 中 标 1。( )

SLC0INT_SLC_FRHOST_BIT0_INT_RAW Host 中 Slave 中 标 0。( )

Register 8.4. SLC0INT_ENA_REG (0xC)

(reserved)	(reserved)	SLC0INT_SLC0_RX_DSCR_ERR_INT_ENA	SLC0INT_SLC0_TX_DSCR_ERR_INT_ENA	(reserved)	SLC0INT_SLC0_RX_EOF_INT_ENA	SLC0INT_SLC0_RX_DONE_INT_ENA	SLC0INT_SLC0_TX_SUC_EOF_INT_ENA	SLC0INT_SLC0_TX_DONE_INT_ENA	(reserved)	SLC0INT_SLC0_TX_OVF_INT_ENA	SLC0INT_SLC0_RX_UDF_INT_ENA	SLC0INT_SLC0_TX_START_INT_ENA	SLC0INT_SLC0_RX_START_INT_ENA	SLC_FRHOST_BIT7_INT_ENA	SLC_FRHOST_BIT6_INT_ENA	SLC_FRHOST_BIT5_INT_ENA	SLC_FRHOST_BIT4_INT_ENA	SLC_FRHOST_BIT3_INT_ENA	SLC_FRHOST_BIT2_INT_ENA	SLC_FRHOST_BIT1_INT_ENA	SLC_FRHOST_BIT0_INT_ENA	
31 27	26 21	20	19	18	17	16	15	14	13 12	11	10	9	8	7	6	5	4	3	2	1	0	
0x00	0 0 0 0 0 0	0	0	0	0	0	0	0	0 0	0	0	0	0	0	0	0	0	0	0	0	0	Reset

SLC0INT_SLC0_RX_DSCR_ERR_INT_ENA Slave 表 中 。(/写)

SLC0INT_SLC0_TX_DSCR_ERR_INT_ENA Slave 表 中 。(/写)

SLC0INT_SLC0_RX_EOF_INT_ENA Slave 操 中 。(/写)

SLC0INT_SLC0_RX_DONE_INT_ENA Slave 单 个 buffer 中 。(/写)

SLC0INT_SLC0_TX_SUC_EOF_INT_ENA Slave 操 中 。(/写)

SLC0INT_SLC0_TX_DONE_INT_ENA Slave 单 个 buffer 中 。( )

SLC0INT_SLC0_TX_OVF_INT_ENA Slave buffer 中 。(/写)

SLC0INT_SLC0_RX_UDF_INT_ENA Slave buffer 中 。(/写)

SLC0INT_SLC0_TX_START_INT_ENA Slave 操 始 中 。(/写)

SLC0INT_SLC0_RX_START_INT_ENA Slave 操 始 中 。(/写)

SLC0INT_SLC_FRHOST_BIT7_INT_ENA Host 中 Slave 中 7。(/写)

SLC0INT_SLC_FRHOST_BIT6_INT_ENA Host 中 Slave 中 6。(/写)

SLC0INT_SLC_FRHOST_BIT5_INT_ENA Host 中 Slave 中 5。(/写)

SLC0INT_SLC_FRHOST_BIT4_INT_ENA Host 中 Slave 中 4。(/写)

SLC0INT_SLC_FRHOST_BIT3_INT_ENA Host 中 Slave 中 3。(/写)

SLC0INT_SLC_FRHOST_BIT2_INT_ENA Host 中 Slave 中 2。(/写)

SLC0INT_SLC_FRHOST_BIT1_INT_ENA Host 中 Slave 中 1。(/写)

SLC0INT_SLC_FRHOST_BIT0_INT_ENA Host 中 Slave 中 0。(/写)

Register 8.5. SLC0INT_CLR_REG (0x10)

31–27	26–21	20	19	18	17	16	15	14	13–12	11	10	9	8	7	6	5	4	3	2	1	0	
(reserved)	(reserved)	SLC0INT_SLC0_RX_DSCR_ERR_INT_CLR	SLC0INT_SLC0_TX_DSCR_ERR_INT_CLR	(reserved)	SLC0INT_SLC0_RX_EOF_INT_CLR	SLC0INT_SLC0_RX_DONE_INT_CLR	SLC0INT_SLC0_TX_SUC_EOF_INT_CLR	SLC0INT_SLC0_TX_DONE_INT_CLR	(reserved)	SLC0INT_SLC0_TX_OVF_INT_CLR	SLC0INT_SLC0_RX_UDF_INT_CLR	SLC0INT_SLC0_TX_START_INT_CLR	SLC0INT_SLC0_RX_START_INT_CLR	SLC0INT_SLC_FRHOST_BIT7_INT_CLR	SLC0INT_SLC_FRHOST_BIT6_INT_CLR	SLC0INT_SLC_FRHOST_BIT5_INT_CLR	SLC0INT_SLC_FRHOST_BIT4_INT_CLR	SLC0INT_SLC_FRHOST_BIT3_INT_CLR	SLC0INT_SLC_FRHOST_BIT2_INT_CLR	SLC0INT_SLC_FRHOST_BIT1_INT_CLR	SLC0INT_SLC_FRHOST_BIT0_INT_CLR	
0x00	0 0 0 0 0 0	0	0	0	0	0	0	0	0 0	0	0	0	0	0	0	0	0	0	0	0	0	Reset

SLC0INT_SLC0_RX_DSCR_ERR_INT_CLR 1， 表 中 。(写)

SLC0INT_SLC0_TX_DSCR_ERR_INT_CLR 1， 表 中 。(写)

SLC0INT_SLC0_RX_EOF_INT_CLR 1， Slave 操 中 。(写)

SLC0INT_SLC0_RX_DONE_INT_CLR 1， Slave 单个 buffer 中 。(写)

SLC0INT_SLC0_TX_SUC_EOF_INT_CLR 1， Slave 操 中 。(写)

SLC0INT_SLC0_TX_DONE_INT_CLR 1， Slave 单个 buffer 中 。(写)

SLC0INT_SLC0_TX_OVF_INT_CLR 1， Slave 中 。(写)

SLC0INT_SLC0_RX_UDF_INT_CLR 1， Slave 中 。(写)

SLC0INT_SLC0_TX_START_INT_CLR 1， Slave 操 始信中 。(写)

SLC0INT_SLC0_RX_START_INT_CLR 1， Slave 操 始信中 。(写)

SLC0INT_SLC_FRHOST_BIT7_INT_CLR SLC0INT_SLC_FRHOST_BIT7_INT 中 。(写)

SLC0INT_SLC_FRHOST_BIT6_INT_CLR SLC0INT_SLC_FRHOST_BIT6_INT 中 。(写)

SLC0INT_SLC_FRHOST_BIT5_INT_CLR SLC0INT_SLC_FRHOST_BIT5_INT 中 。(写)

SLC0INT_SLC_FRHOST_BIT4_INT_CLR SLC0INT_SLC_FRHOST_BIT4_INT 中 。(写)

SLC0INT_SLC_FRHOST_BIT3_INT_CLR SLC0INT_SLC_FRHOST_BIT3_INT 中 。(写)

SLC0INT_SLC_FRHOST_BIT2_INT_CLR SLC0INT_SLC_FRHOST_BIT2_INT 中 。(写)

寄存器描述下一页继续。

Register 8.5. SLC0INT_CLR_REG (0x10)

继上一页寄存器描述。

SLC0INT_SLC_FRHOST_BIT1_INT_CLR SLC0INT_SLC_FRHOST_BIT1_INT 中 。(写)

SLC0INT_SLC_FRHOST_BIT0_INT_CLR SLC0INT_SLC_FRHOST_BIT0_INT 中 。(写)

Register 8.6. SLC0RX_LINK_REG (0x3C)

31	30	29	28	27	20	19	0	
(reserved)	SLC0RX_SLC0_RXLINK_RESTART	SLC0RX_SLC0_RXLINK_START	SLC0RX_SLC0_RXLINK_STOP	(reserved)		SLC0RX_SLC0_RXLINK_ADDR		
0	0	0	0	0 0 0 0 0 0 0 0		0x000000		Reset

SLC0RX_SLC0_RXLINK_RESTART 1，重启并表操包。(/写)

SLC0RX_SLC0_RXLINK_START 1，启始表操包。SLC0_RXLINK_ADDR 。(/写)

SLC0RX_SLC0_RXLINK_STOP 1，表操 。(/写)

SLC0RX_SLC0_RXLINK_ADDR Slave 包表始 20 。(/写)

Register 8.7. SLC0TX_LINK_REG (0x40)

(reserved)	SLC0TX_SLC0_TXLINK_RESTART	SLC0TX_SLC0_TXLINK_START	SLC0TX_SLC0_TXLINK_STOP	(reserved)	SLC0TX_SLC0_TXLINK_ADDR	
31	30	29	28	27 … 20	19 … 0	
0	0	0	0	0 0 0 0 0 0 0 0	0x000000	Reset

SLC0TX_SLC0_TXLINK_RESTART 1，重启 并 表操 包。(/写)

SLC0TX_SLC0_TXLINK_START 1，启 表操 包。 始 SLC0_TXLINK_ADDR 。(/写)

SLC0TX_SLC0_TXLINK_STOP 1， 表操 。(/写)

SLC0TX_SLC0_TXLINK_ADDR Slave 包 表 始 20 。(/写)

Register 8.8. SLCINTVEC_TOHOST_REG (0x4C)

(reserved)	(reserved)	(reserved)	SLCINTVEC_SLC0_TOHOST_INTVEC	
31 … 24	23 … 16	15 … 8	7 … 0	
0x000	0 0 0 0 0 0 0 0	0x000	0x000	Reset

SLCINTVEC_SLC0_TOHOST_INTVEC Slave 中 Host 中 。(写)

Register 8.9. SLC0TOKEN1_REG (0x54)

(reserved)	SLC0TOKEN1_SLC0_TOKEN1	(reserved)	SLC0TOKEN1_SLC0_TOKEN1_INC_MORE	(reserved)	SLC0TOKEN1_SLC0_TOKEN1_WDATA	
31 28	27 16	15	14	13 12	11 0	
0x00	0x0000	0	0	0 0	0x0000	Reset

SLC0TOKEN1_SLC0_TOKEN1 包 buffer 。( )

SLC0TOKEN1_SLC0_TOKEN1_INC_MORE 可 包 buffer 。(写)

SLC0TOKEN1_SLC0_TOKEN1_WDATA 可 包 buffer 。(写)

Register 8.10. SLCCONF1_REG (0x60)

(reserved)	(reserved)	(reserved)	SLCCONF1_SLC0_RX_STITCH_EN	SLCCONF1_SLC0_TX_STITCH_EN	SLCCONF1_SLC0_LEN_AUTO_CLR	
31 23	22 16	15 7	6	5	4	
0x000	0 0 0 0 0 0 0	0 0 0 0 0 0 0 0 0	1	1	1	Reset

SLCCONF1_SLC0_RX_STITCH_EN 始 0。 (/写)

SLCCONF1_SLC0_TX_STITCH_EN 始 0。 (/写)

SLCCONF1_SLC0_LEN_AUTO_CLR 始 0。 (/写)

Register 8.11. SLC_RX_DSCR_CONF_REG (0x98)

(reserved)	SLC_SLC0_TOKEN_NO_REPLACE	
31 … 1	0	
0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0	0	Reset

SLC_SLC0_TOKEN_NO_REPLACE 始化 1。 已 (/写)

Register 8.12. SLC0_LEN_CONF_REG (0xE4)

(reserved)	(reserved)	SLC0_LEN_INC_MORE	(reserved)	SLC0_LEN_WDATA	
31 … 29	28 … 23	22	21 … 20	19 … 0	
0x0	0 0 0 0 0 0	0	0 0	0x000000	Reset

SLC0_LEN_INC_MORE 包长 。(写)

SLC0_LEN_WDATA 包长。(写)

Register 8.13. SLC0_LENGTH_REG (0xE8)

(reserved)	SLC0_LEN	
31 … 20	19 … 0	
0x0000	0x000000	Reset

SLC0_LEN Slave 包 长 。( )

8.6 SLC Host 寄存器

SDIO 个 始 0x3FF5_5000.

Register 8.14. SLC0HOST_TOKEN_RDATA (0x44)

(reserved)	HOSTREG_SLC0_TOKEN1	(reserved)	
31 28	27 16	15 0	
0x000	0x000	0x000	Reset

HOSTREG_SLC0_TOKEN1 Slave 侧可 Host buffer 。( )

Register 8.15. SLC0HOST_INT_RAW_REG (0x50)

(reserved)	(reserved)	SLC0HOST_SLC0_RX_NEW_PACKET_INT_RAW	(reserved)	SLC0HOST_SLC0_TX_OVF_INT_RAW	SLC0HOST_SLC0_RX_UDF_INT_RAW	(reserved)	SLC0HOST_SLC0_TOHOST_BIT7_INT_RAW	SLC0HOST_SLC0_TOHOST_BIT6_INT_RAW	SLC0HOST_SLC0_TOHOST_BIT5_INT_RAW	SLC0HOST_SLC0_TOHOST_BIT4_INT_RAW	SLC0HOST_SLC0_TOHOST_BIT3_INT_RAW	SLC0HOST_SLC0_TOHOST_BIT2_INT_RAW	SLC0HOST_SLC0_TOHOST_BIT1_INT_RAW	SLC0HOST_SLC0_TOHOST_BIT0_INT_RAW	
31 26	25 24	23	22 18	17	16	15 8	7	6	5	4	3	2	1	0	
0x00	0 0	0	0 0 0 0 0	0	0	0 0 0 0 0 0 0 0	0	0	0	0	0	0	0	0	Reset

SLC0HOST_SLC0_RX_NEW_PACKET_INT_RAW SLC0HOST_SLC0_RX_NEW_PACKET_INT 始中 。( )

SLC0HOST_SLC0_TX_OVF_INT_RAW SLC0HOST_SLC0_TX_OVF_INT 始中 。( )

SLC0HOST_SLC0_RX_UDF_INT_RAW SLC0HOST_SLC0_RX_UDF_INT 始中 。( )

SLC0HOST_SLC0_TOHOST_BIT7_INT_RAW SLC0HOST_SLC0_TOHOST_BIT7_INT 始中 。( )

SLC0HOST_SLC0_TOHOST_BIT6_INT_RAW SLC0HOST_SLC0_TOHOST_BIT6_INT 始中 。( )

SLC0HOST_SLC0_TOHOST_BIT5_INT_RAW SLC0HOST_SLC0_TOHOST_BIT5_INT 始中 。( )

SLC0HOST_SLC0_TOHOST_BIT4_INT_RAW SLC0HOST_SLC0_TOHOST_BIT4_INT 始中 。( )

SLC0HOST_SLC0_TOHOST_BIT3_INT_RAW SLC0HOST_SLC0_TOHOST_BIT3_INT 始中 。( )

SLC0HOST_SLC0_TOHOST_BIT2_INT_RAW SLC0HOST_SLC0_TOHOST_BIT2_INT 始中 。( )

SLC0HOST_SLC0_TOHOST_BIT1_INT_RAW SLC0HOST_SLC0_TOHOST_BIT1_INT 始中 。( )

SLC0HOST_SLC0_TOHOST_BIT0_INT_RAW SLC0HOST_SLC0_TOHOST_BIT0_INT 始中 。( )

Register 8.16. SLC0HOST_INT_ST_REG (0x58)

(reserved)	(reserved)	SLC0HOST_SLC0_RX_NEW_PACKET_INT_ST	(reserved)	SLC0HOST_SLC0_TX_OVF_INT_ST	SLC0HOST_SLC0_RX_UDF_INT_ST	(reserved)	SLC0HOST_SLC0_TOHOST_BIT7_INT_ST	SLC0HOST_SLC0_TOHOST_BIT6_INT_ST	SLC0HOST_SLC0_TOHOST_BIT5_INT_ST	SLC0HOST_SLC0_TOHOST_BIT4_INT_ST	SLC0HOST_SLC0_TOHOST_BIT3_INT_ST	SLC0HOST_SLC0_TOHOST_BIT2_INT_ST	SLC0HOST_SLC0_TOHOST_BIT1_INT_ST	SLC0HOST_SLC0_TOHOST_BIT0_INT_ST	
31 26	25 24	23	22 18	17	16	15 8	7	6	5	4	3	2	1	0	
0x00	0 0	0	0 0 0 0 0	0	0	0 0 0 0 0 0 0 0	0	0	0	0	0	0	0	0	Reset

SLC0HOST_SLC0_RX_NEW_PACKET_INT_ST SLC0HOST_SLC0_RX_NEW_PACKET_INT 中断屏蔽中 。( )

SLC0HOST_SLC0_TX_OVF_INT_ST SLC0HOST_SLC0_TX_OVF_INT 中断屏蔽中 。( )

SLC0HOST_SLC0_RX_UDF_INT_ST SLC0HOST_SLC0_RX_UDF_INT 中断屏蔽中 。( )

SLC0HOST_SLC0_TOHOST_BIT7_INT_ST SLC0HOST_SLC0_TOHOST_BIT7_INT 中断屏蔽中 。( )

SLC0HOST_SLC0_TOHOST_BIT6_INT_ST SLC0HOST_SLC0_TOHOST_BIT6_INT 中断屏蔽中 。( )

SLC0HOST_SLC0_TOHOST_BIT5_INT_ST SLC0HOST_SLC0_TOHOST_BIT5_INT 中断屏蔽中 。( )

SLC0HOST_SLC0_TOHOST_BIT4_INT_ST SLC0HOST_SLC0_TOHOST_BIT4_INT 中断屏蔽中 。( )

SLC0HOST_SLC0_TOHOST_BIT3_INT_ST SLC0HOST_SLC0_TOHOST_BIT3_INT 中断屏蔽中 。( )

SLC0HOST_SLC0_TOHOST_BIT2_INT_ST SLC0HOST_SLC0_TOHOST_BIT2_INT 中断屏蔽中 。( )

SLC0HOST_SLC0_TOHOST_BIT1_INT_ST SLC0HOST_SLC0_TOHOST_BIT1_INT 中断屏蔽中 。( )

SLC0HOST_SLC0_TOHOST_BIT0_INT_ST SLC0HOST_SLC0_TOHOST_BIT0_INT 中断屏蔽中 。( )

Register 8.17. SLCHOST_PKT_LEN_REG (0x60)

SLCHOST_HOSTREG_SLC0_LEN_CHECK	SLCHOST_HOSTREG_SLC0_LEN	
31 … 20	19 … 0	
0x000	0x000	Reset

SLCHOST_HOSTREG_SLC0_LEN_CHECK HOSTREG_SLC0_LEN[9:0] HOSTREG_SLC0_LEN[19:10]。( )

SLCHOST_HOSTREG_SLC0_LEN Slave 包长 。 Host 时才 。

Register 8.18. SLCHOST_CONF_W0_REG (0x6C)

SLCHOST_CONF3	SLCHOST_CONF2	SLCHOST_CONF1	SLCHOST_CONF0	
31 … 24	23 … 16	15 … 8	7 … 0	
0x000	0x000	0x000	0x000	Reset

SLCHOST_CONF3 Host Slave 。Host Slave 可 写。(/写)

SLCHOST_CONF2 Host Slave 。Host Slave 可 写。(/写)

SLCHOST_CONF1 Host Slave 。Host Slave 可 写。(/写)

SLCHOST_CONF0 Host Slave 。Host Slave 可 写。(/写)

Register 8.19. SLCHOST_CONF_W1_REG (0x70)

SLCHOST_CONF7	SLCHOST_CONF6	SLCHOST_CONF5	SLCHOST_CONF4	
31 24	23 16	15 8	7 0	
0x000	0x000	0x000	0x000	Reset

SLCHOST_CONF7 Host Slave 。Host Slave 可 写。(/写)

SLCHOST_CONF6 Host Slave 。Host Slave 可 写。(/写)

SLCHOST_CONF5 Host Slave 。Host Slave 可 写。(/写)

SLCHOST_CONF4 Host Slave 。Host Slave 可 写。(/写)

Register 8.20. SLCHOST_CONF_W2_REG (0x74)

SLCHOST_CONF11	SLCHOST_CONF10	SLCHOST_CONF9	SLCHOST_CONF8	
31 24	23 16	15 8	7 0	
0x000	0x000	0x000	0x000	Reset

SLCHOST_CONF11 Host Slave 。Host Slave 可 写。(/写)

SLCHOST_CONF10 Host Slave 。Host Slave 可 写。(/写)

SLCHOST_CONF9 Host Slave 。Host Slave 可 写。(/写)

SLCHOST_CONF8 Host Slave 。Host Slave 可 写。(/写)

Register 8.21. SLCHOST_CONF_W3_REG (0x78)

SLCHOST_CONF15	SLCHOST_CONF14	
31 24	23 16	
0x000	0x000	Reset

SLCHOST_CONF15 Host Slave 。Host Slave 可 写。(/写)

SLCHOST_CONF14 Host Slave 。Host Slave 可 写。(/写)

Register 8.22. SLCHOST_CONF_W4_REG (0x7C)

SLCHOST_CONF19 (31–24)	SLCHOST_CONF18 (23–16)	
0x000	0x000	Reset

SLCHOST_CONF19 Host Slave 。Host Slave 可 写。(/写)

SLCHOST_CONF18 Host Slave 。Host Slave 可 写。(/写)

Register 8.23. SLCHOST_CONF_W6_REG (0x88)

SLCHOST_CONF27 (31–24)	SLCHOST_CONF26 (23–16)	SLCHOST_CONF25 (15–8)	SLCHOST_CONF24 (7–0)	
0x000	0x000	0x000	0x000	Reset

SLCHOST_CONF27 Host Slave 。Host Slave 可 写。(/写)

SLCHOST_CONF26 Host Slave 。Host Slave 可 写。(/写)

SLCHOST_CONF25 Host Slave 。Host Slave 可 写。(/写)

SLCHOST_CONF24 Host Slave 。Host Slave 可 写。(/写)

Register 8.24. SLCHOST_CONF_W7_REG (0x8C)

SLCHOST_CONF31 (31–24)	(reserved) (23–16)	SLCHOST_CONF29 (15–8)	(reserved) (7–0)	
0 0 0 0 0 0 0 0	0x000	0 0 0 0 0 0 0 0	0x000	Reset

SLCHOST_CONF31 Host 中 Slave 中 。(写)

SLCHOST_CONF29 Host 中 Slave 中 。(写)

Register 8.25. SLCHOST_CONF_W8_REG (0x9C)

SLCHOST_CONF35	SLCHOST_CONF34	SLCHOST_CONF33	SLCHOST_CONF32	
31 24	23 16	15 8	7 0	
0x000	0x000	0x000	0x000	Reset

SLCHOST_CONF35 Host Slave 。Host Slave 可 写。(/写)

SLCHOST_CONF34 Host Slave 。Host Slave 可 写。(/写)

SLCHOST_CONF33 Host Slave 。Host Slave 可 写。(/写)

SLCHOST_CONF32 Host Slave 。Host Slave 可 写。(/写)

Register 8.26. SLCHOST_CONF_W9_REG (0xA0)

SLCHOST_CONF39	SLCHOST_CONF38	SLCHOST_CONF37	SLCHOST_CONF36	
31 24	23 16	15 8	7 0	
0x000	0x000	0x000	0x000	Reset

SLCHOST_CONF39 Host Slave 。Host Slave 可 写。(/写)

SLCHOST_CONF38 Host Slave 。Host Slave 可 写。(/写)

SLCHOST_CONF37 Host Slave 。Host Slave 可 写。(/写)

SLCHOST_CONF36 Host Slave 。Host Slave 可 写。(/写)

Register 8.27. SLCHOST_CONF_W10_REG (0xA4)

SLCHOST_CONF43	SLCHOST_CONF42	SLCHOST_CONF41	SLCHOST_CONF40	
31 24	23 16	15 8	7 0	
0x000	0x000	0x000	0x000	Reset

SLCHOST_CONF43 Host Slave 。Host Slave 可 写。(/写)

SLCHOST_CONF42 Host Slave 。Host Slave 可 写。(/写)

SLCHOST_CONF41 Host Slave 。Host Slave 可 写。(/写)

SLCHOST_CONF40 Host Slave 。Host Slave 可 写。(/写)

Register 8.28. SLCHOST_CONF_W11_REG (0xA8)

SLCHOST_CONF47	SLCHOST_CONF46	SLCHOST_CONF45	SLCHOST_CONF44	
31 24	23 16	15 8	7 0	
0x000	0x000	0x000	0x000	Reset

SLCHOST_CONF47 Host Slave 。Host Slave 可 写。(/写)

SLCHOST_CONF46 Host Slave 。Host Slave 可 写。(/写)

SLCHOST_CONF45 Host Slave 。Host Slave 可 写。(/写)

SLCHOST_CONF44 Host Slave 。Host Slave 可 写。(/写)

Register 8.29. SLCHOST_CONF_W12_REG (0xAC)

SLCHOST_CONF51	SLCHOST_CONF50	SLCHOST_CONF49	SLCHOST_CONF48	
31 24	23 16	15 8	7 0	
0x000	0x000	0x000	0x000	Reset

SLCHOST_CONF51 Host Slave 。Host Slave 可 写。(/写)

SLCHOST_CONF50 Host Slave 。Host Slave 可 写。(/写)

SLCHOST_CONF49 Host Slave 。Host Slave 可 写。(/写)

SLCHOST_CONF48 Host Slave 。Host Slave 可 写。(/写)

Register 8.30. SLCHOST_CONF_W13_REG (0xB0)

SLCHOST_CONF55	SLCHOST_CONF54	SLCHOST_CONF53	SLCHOST_CONF52	
31 24	23 16	15 8	7 0	
0x000	0x000	0x000	0x000	Reset

SLCHOST_CONF55 Host Slave 。Host Slave 可 写。(/写)

SLCHOST_CONF54 Host Slave 。Host Slave 可 写。(/写)

SLCHOST_CONF53 Host Slave 。Host Slave 可 写。(/写)

SLCHOST_CONF52 Host Slave 。Host Slave 可 写。(/写)

Register 8.31. SLCHOST_CONF_W14_REG (0xB4)

SLCHOST_CONF59	SLCHOST_CONF58	SLCHOST_CONF57	SLCHOST_CONF56	
31 24	23 16	15 8	7 0	
0x000	0x000	0x000	0x000	Reset

SLCHOST_CONF59 Host Slave 。Host Slave 可 写。(/写)

SLCHOST_CONF58 Host Slave 。Host Slave 可 写。(/写)

SLCHOST_CONF57 Host Slave 。Host Slave 可 写。(/写)

SLCHOST_CONF56 Host Slave 。Host Slave 可 写。(/写)

Register 8.32. SLCHOST_CONF_W15_REG (0xB8)

SLCHOST_CONF63	SLCHOST_CONF62	SLCHOST_CONF61	SLCHOST_CONF60	
31 24	23 16	15 8	7 0	
0x000	0x000	0x000	0x000	Reset

SLCHOST_CONF63 Host Slave 。Host Slave 可 写。(/写)

SLCHOST_CONF62 Host Slave 。Host Slave 可 写。(/写)

SLCHOST_CONF61 Host Slave 。Host Slave 可 写。(/写)

SLCHOST_CONF60 Host Slave 。Host Slave 可 写。(/写)

Register 8.33. SLC0HOST_INT_CLR_REG (0xD4)

(reserved)	(reserved)	SLC0HOST_SLC0_RX_NEW_PACKET_INT_CLR	(reserved)	SLC0HOST_SLC0_TX_OVF_INT_CLR	SLC0HOST_SLC0_RX_UDF_INT_CLR	(reserved)	SLC0HOST_SLC0_TOHOST_BIT7_INT_CLR	SLC0HOST_SLC0_TOHOST_BIT6_INT_CLR	SLC0HOST_SLC0_TOHOST_BIT5_INT_CLR	SLC0HOST_SLC0_TOHOST_BIT4_INT_CLR	SLC0HOST_SLC0_TOHOST_BIT3_INT_CLR	SLC0HOST_SLC0_TOHOST_BIT2_INT_CLR	SLC0HOST_SLC0_TOHOST_BIT1_INT_CLR	SLC0HOST_SLC0_TOHOST_BIT0_INT_CLR	
31 26	25 24	23	22 18	17	16	15 8	7	6	5	4	3	2	1	0	
0x00	0 0	0	0 0 0 0 0	0	0	0 0 0 0 0 0 0 0	0	0	0	0	0	0	0	0	Reset

SLC0HOST_SLC0_RX_NEW_PACKET_INT_CLR SLC0HOST_SLC0_RX_NEW_PACKET_INT 中 。(写)

SLC0HOST_SLC0_TX_OVF_INT_CLR SLC0HOST_SLC0_TX_OVF_INT 中 。(写)

SLC0HOST_SLC0_RX_UDF_INT_CLR SLC0HOST_SLC0_RX_UDF_INT 中 。(写)

SLC0HOST_SLC0_TOHOST_BIT7_INT_CLR SLC0HOST_SLC0_TOHOST_BIT7_INT 中 。(写)

SLC0HOST_SLC0_TOHOST_BIT6_INT_CLR SLC0HOST_SLC0_TOHOST_BIT6_INT 中 。(写)

SLC0HOST_SLC0_TOHOST_BIT5_INT_CLR SLC0HOST_SLC0_TOHOST_BIT5_INT 中 。(写)

SLC0HOST_SLC0_TOHOST_BIT4_INT_CLR SLC0HOST_SLC0_TOHOST_BIT4_INT 中 。(写)

SLC0HOST_SLC0_TOHOST_BIT3_INT_CLR SLC0HOST_SLC0_TOHOST_BIT3_INT 中 。(写)

SLC0HOST_SLC0_TOHOST_BIT2_INT_CLR SLC0HOST_SLC0_TOHOST_BIT2_INT 中 。(写)

SLC0HOST_SLC0_TOHOST_BIT1_INT_CLR SLC0HOST_SLC0_TOHOST_BIT1_INT 中 。(写)

SLC0HOST_SLC0_TOHOST_BIT0_INT_CLR SLC0HOST_SLC0_TOHOST_BIT0_INT 中 。(写)

Register 8.34. SLC0HOST_FUNC1_INT_ENA_REG (0xDC)

(reserved)	(reserved)	SLC0HOST_FN1_SLC0_RX_NEW_PACKET_INT_ENA	(reserved)	SLC0HOST_FN1_SLC0_TX_OVF_INT_ENA	SLC0HOST_FN1_SLC0_RX_UDF_INT_ENA	(reserved)	SLC0HOST_FN1_SLC0_TOHOST_BIT7_INT_ENA	SLC0HOST_FN1_SLC0_TOHOST_BIT6_INT_ENA	SLC0HOST_FN1_SLC0_TOHOST_BIT5_INT_ENA	SLC0HOST_FN1_SLC0_TOHOST_BIT4_INT_ENA	SLC0HOST_FN1_SLC0_TOHOST_BIT3_INT_ENA	SLC0HOST_FN1_SLC0_TOHOST_BIT2_INT_ENA	SLC0HOST_FN1_SLC0_TOHOST_BIT1_INT_ENA	SLC0HOST_FN1_SLC0_TOHOST_BIT0_INT_ENA	
31 26	25 24	23	22 18	17	16	15 8	7	6	5	4	3	2	1	0	
0x00	0 0	0	0 0 0 0 0	0	0	0 0 0 0 0 0 0 0	0	0	0	0	0	0	0	0	Reset

SLC0HOST_FN1_SLC0_RX_NEW_PACKET_INT_ENA SLC0HOST_FN1_SLC0_RX_NEW_PACKET_INT 中 中 。(/写)

SLC0HOST_FN1_SLC0_TX_OVF_INT_ENA SLC0HOST_FN1_SLC0_TX_OVF_INT 中 中 。(/写)

SLC0HOST_FN1_SLC0_RX_UDF_INT_ENA SLC0HOST_FN1_SLC0_RX_UDF_INT 中 中 。(/写)

SLC0HOST_FN1_SLC0_TOHOST_BIT7_INT_ENA SLC0HOST_FN1_SLC0_TOHOST_BIT7_INT 中 中 。(/写)

SLC0HOST_FN1_SLC0_TOHOST_BIT6_INT_ENA SLC0HOST_FN1_SLC0_TOHOST_BIT6_INT 中 中 。(/写)

SLC0HOST_FN1_SLC0_TOHOST_BIT5_INT_ENA SLC0HOST_FN1_SLC0_TOHOST_BIT5_INT 中 中 。(/写)

SLC0HOST_FN1_SLC0_TOHOST_BIT4_INT_ENA SLC0HOST_FN1_SLC0_TOHOST_BIT4_INT 中 中 。(/写)

SLC0HOST_FN1_SLC0_TOHOST_BIT3_INT_ENA SLC0HOST_FN1_SLC0_TOHOST_BIT3_INT 中 中 。(/写)

SLC0HOST_FN1_SLC0_TOHOST_BIT2_INT_ENA SLC0HOST_FN1_SLC0_TOHOST_BIT2_INT 中 中 。(/写)

SLC0HOST_FN1_SLC0_TOHOST_BIT1_INT_ENA SLC0HOST_FN1_SLC0_TOHOST_BIT1_INT 中 中 。(/写)

SLC0HOST_FN1_SLC0_TOHOST_BIT0_INT_ENA SLC0HOST_FN1_SLC0_TOHOST_BIT0_INT 中 中 。(/写)

Register 8.35. SLCHOST_CONF_REG (0x1F0)

(reserved)	(reserved)	SLCHOST_FRC_POS_SAMP	SLCHOST_FRC_NEG_SAMP	SLCHOST_FRC_SDIO20	SLCHOST_FRC_SDIO11	
31 28	27 20	19 15	14 10	9 5	4 0	
0 0 0 0	0 0 0 0 0 0 0 0	0 0 0 0 0	0 0 0 0 0	0 0 0 0 0	0 0 0 0 0	Reset

SLCHOST_FRC_POS_SAMP 时 上升 采 。(/写)

SLCHOST_FRC_NEG_SAMP 时 采 。(/写)

SLCHOST_FRC_SDIO20 时 上升 。(/写)

SLCHOST_FRC_SDIO11 时 。(/写)

8.7 HINF 寄存器

SDIO 三个 始 0x3FF4_B000。

Register 8.36. HINF_CFG_DATA1_REG (0x4)

(reserved)	HINF_HIGHSPEED_ENABLE	HINF_SDIO_IOREADY1	
31 3	2	1	
0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0	0	0	Reset

HINF_HIGHSPEED_ENABLE 始 1。 (/ 写)

HINF_SDIO_IOREADY1 始 1。 (/ 写)

9 SD/MMC 主机控制器

9.1 概述

ESP32 一个 安 (SDIO)、MMC CE-ATA 设备 ，
设备 (APB) 部 设备。ESP32 个 部 （ 0 1）。

9.2 主要特性

ESP32 ：

- 个 部
- SD 3.0 3.01 标
- MMC 版本 4.41、4.5、4.51
- CE-ATA 版本 1.1
- 1-bit、4-bit 8-bit （ 0 ）

SD/MMC 设 9-1 。 设 ， 不 时 。

图 9-1. SD/MMC 外设连接的拓扑结构

9.3 SD/MMC 外部接口信号

SD/MMC 部 clk、cmd、data ， 包括 中 、 测 写保 。 个
9-2 。 个 表 9-1 。

图 9-2. SD/MMC 外部接口信号

表 9-1. SD/MMC 管脚描述

cclk_out		时
ccmd		/
cdata		写
card_detect_n		测 上
card_write_prt		写保

9.4 功能描述

9.4.1 SD/MMC 架构

SD/MMC 单 (BIU) 单 (CIU) 部 , 9-3 。 中 :

BIU : APB 、FIFO 写 , 写操 DMA 。

CIU : 部 , 时 。

图 9-3. SD/MMC 基本架构

9.4.1.1 BIU 模块

(HIU) APB FIFO。 , DMA 。 9-3 BIU 。

BIU :

-
- DMA
-
-
- FIFO
- /

9.4.1.2 CIU 模块

。 CIU , (Cmd Path) (Data Path) SD/MMC/CE-ATA 。CIU 。 9-3 CIU 。

CIU 包 :

-
-
- SDIO
-
- Mux/Demux

9.4.2 命令通路

:

- 设 参
- 设 参
- （ccmd_out ）
- （ccmd_in ）
- BIU
- P-bit

9-4 。

图 9-4. 命令通路状态机

9.4.3 数据通路

写 时 FIFO 中 并 cdata_out 上 ; 时 cdata_in 上 并 FIFO。 不 时, 才 参 , 包括 expected data、 /写 、 / 、 、 、 、超时 。

中设 data_expected , , 始 操 :

- /写 1 时,
- /写 0 时,

9.4.3.1 数据发送

9-5 。 写 个时 期 始 ; 测 冗 查 (CRC) , 。 超时所 , 不 。 中 transfer_mode , 上。

图 9-5. 数据传输状态机

9.4.3.2 数据接收

9-6 。 个时 期 始 ; 测 CRC , 。 超时而 , BIU 不 。 CIU 操 , , BIU 不 。

超时 , BIU 超时 并 。 中 transfer_mode , 。

图 9-6. 数据接收状态机

9.5 CIU 操作的软件限制

- 一 一个 。 , 时, 不 一 。 可 一 , 。
- 一 一个 。
- 写操 期间, 时 FIFO 而 , 必 FIFO 中并启 时 , 才可 一个 /中 。
- SDIO/COMBO 期间, , 并 , 必 重 FIFO 并启 , 启 一个 。
- 时 (CMD0、CMD15 CMD52_reset), 必 上设 stop_abort_cmd , 保 CIU 可 。
- RINTSTS 中设 时, CIU 不 保 SDIO 中 。 SDIO 中 , 并 /中 , 。
- 一个 中 FIFO 而 时 , 个 FIFO 重 时 。
- 一 / 中 一个 CE-ATA 设备。 , 一个 CE-ATA 设备 , CE-ATA 设备不可 。
- CE-ATA 设备 中 (nIEN = 0), 时 RW_BLK , 不 RW_BLK 一设备 (中 RW_BLK CE-ATA 可 RW_MULTIPLE_BLOCK MMC)。 CCS 时可 CCSD。

- 一设备，中 CE-ATA 设备上 (nIEN = 1)，一个 。
- CE-ATA 设备不 。
- CE-ATA 不 send_auto_stop （不设 send_auto_stop ）。

始被上，被 不改变：

- CMD —
- CMDARG — 参
- BYTCNT —
- BLKSIZ —
- CLKDIV — 时
- CKLENA — 时
- CLKSRC — 时
- TMOUT — 超时
- CTYPE —

9.6 收发数据 RAM

RAM 一个 ，个单 。时可 CPU APB DMA 写操 ，DMA 9.8 中 。

9.6.1 发送 RAM 模块

写 ：DMA CPU 写。

SDIO ，可 APB DMA 写 RAM 。中，APB CPU 写 EMAC_FIFO。

9.6.2 接收 RAM 模块

：DMA CPU 写。

单时，写 RAM 。可 APB DMA RAM 中 。中，APB CPU EMAC_FIFO 中 。

9.7 链表环结构

一 表 部 ：表本 buffer。一个 表 一个 一 buffer 一个 表。表 9-7 。

图 9-7. 链表环结构

9.8 链表结构

个 表 ， 个 表 4 个 word ， 9-8 。表 9-2、表 9-3、表 9-4、表 9-5 表 。

图 9-8. 链表结构

DES0单 包 。

表 9-2. DES0 链表描述

31	OWN	设 时， 表 表 DMAC 。 被 重 ， 表 表 。DMAC 。

30	CES (Card Error Summary)	表 /写 。 RINTSTS 中，CES ： • EBE： • RTO： 超时 • RCRC： CRC • SBE： 始 • DRTO： 超时 • DCRC： 冗 • RE：
29:6	Reserved	Reserved
5	ER (End of Ring)	时， 表 表 一个 表。 DMAC 表 ， 一个 表 。
4	CH (Second Address Chained)	时， 表 表 个 一个 表 。 时，BS2 (DES1[25:13]) 。
3	FD (First Descriptor)	时， 表 表包 一 。 一个 0， 一个 表包 始。
2	LD (Last Descriptor)	DMA 一个 。 时， 表 表 一 个 。 表 ， 0。 ， 被 LD 表 ， 0。
1	DIC (Disable Interrupt on Completion)	时， 保 表 中 ， DMAC (IDSTS) 上 TI/RI 设 。
0	Reserved	Reserved

DES1 包 。

表 9-3. DES1

31:26	Reserved	Reserved
25:13	Reserved	Reserved
12:0	BS1（Buffer 1 ）	表 。 必 4 倍 。 不 4 倍 ， 。 不 0。

DES2 包 。

表 9-4. DES2

31:0	Buffer Address Pointer 1 ( )	表 。

表不 表 中 一个 表， DES3 包 一个 表 。

表 9-5. DES3

31:0	Next Descriptor Address (一个 表)	个 (DES0[4])， 包 包 一个 表 。 不 一个 表， 一个 表 必 DES3[1:0] = 0。

9.9 初始化

9.9.1 DMAC 初始化

DMAC 始化 :

1. DMAC Bus Mode Register (BMOD_REG) 设 参 。
2. DMAC Interrupt Enable Register (IDINTEN) 写 屏蔽不必 中 。
3. 表 表 表 表。 写 DMAC 表 表 (BDADDR)，DMAC 表 始 。
4. DMAC 尝试 表 表 表。

9.9.2 DMAC 数据发送初始化

DMAC :

1. 设 (DES0-DES3)，设 OWN (DES0[31])，并 备 。
2. BIU 中 CMD 中 写 。
3. 设 (FIFOTH 中 TX_WMARK)。
4. DMAC 表并 查 OWN 。 OWN ， 表 表。 ，DMAC ，并 IDSTS 中产 Descriptor Unable 中 。 时， 写 PLDMND_REG DMAC。
5. Command Done ， BIU 报 ， 表 可 。
6. DMAC BIU DMA (dw_dma_req)。 。 所不 一个 ， AHB 上 单 。
7. DMAC ，并 FIFO 中。
8. 个 表时，DMAC 一个 表，并 一个 表 操 。 一 表 表 个 表。

9. 时，设中 IDSTS 中 。，OWN DMAC DES0 写操 。

9.9.3 DMAC 数据接收初始化

DMAC :

1. (DES0 - DES3)，设 OWN (DES0[31])。
2. BIU 中 。
3. 设 (FIFOTH 中 RX_WMARK)。
4. DMAC 表并查 OWN 。 OWN 设，表 表。，DMAC，并 IDSTS 中产 Descriptor Unable 中 。时， 写 PLDMND_REG DMAC。
5. Command Done ， BIU 报 ，表 可 。
6. DMAC BIU DMA (dw_dma_req)。 。burst 而不 一个 ， AHB 上 单 。
7. DMAC FIFO ，并 。
8. 个 表时，DMAC 一个 表，并 一个 表 操 。 一 表 表 个 表。
9. 时，设中 IDSTS 中 。，OWN DMAC DES0 写操 。

9.10 时钟相位选择

采 时 ，可参 改变时 。

图 9-9. 时钟相位选择

时 CLK_EDGE_SEL ， 。

9.11 中断

中 可 产 。IDSTS 包 可 中 。IDINTEN 包 一个 中 。

IDSTS 中 中 一 常中 bit8 NIS 常中 bit9 AIS。 1 写 ，可 中 。 中 中 被 ， 被 。 被 ，中 dmac_intr_o 。

中 不 ， 中 ， 不 产 中 。 ， 中 IDSTS [1] 表 一个 个 。

个 时 产 一个中 。 必 IDSTS 查 中 。

9.12 寄存器列表

CTRL_REG	Control register	0x0000	R/W
CLKDIV_REG	Clock divider configuration register	0x0008	R/W
CLKSRC_REG	Clock source selection register	0x000C	R/W
CLKENA_REG	Clock enable register	0x0010	R/W
TMOUT_REG	Data and response timeout configuration register	0x0014	R/W
CTYPE_REG	Card bus width configuration register	0x0018	R/W
BLKSIZ_REG	Card data block size configuration register	0x001C	R/W
BYTCNT_REG	Data transfer length configuration register	0x0020	R/W
INTMASK_REG	SDIO interrupt mask register	0x0024	R/W
CMDARG_REG	Command argument data register	0x0028	R/W
CMD_REG	Command and boot configuration register	0x002C	R/W
RESP0_REG	Response data register	0x0030	RO
RESP1_REG	Long response data register	0x0034	RO
RESP2_REG	Long response data register	0x0038	RO
RESP3_REG	Long response data register	0x003C	RO
MINTSTS_REG	Masked interrupt status register	0x0040	RO
RINTSTS_REG	Raw interrupt status register	0x0044	R/W
STATUS_REG	SD/MMC status register	0x0048	RO
FIFOTH_REG	FIFO configuration register	0x004C	R/W
CDETECT_REG	Card detect register	0x0050	RO
WRTPRT_REG	Card write protection (WP) status register	0x0054	RO
TCBCNT_REG	Transferred byte count register	0x005C	RO
TBBCNT_REG	Transferred byte count register	0x0060	RO
DEBNCE_REG	Debounce filter time configuration register	0x0064	R/W
USRID_REG	User ID (scratchpad) register	0x0068	R/W
RST_N_REG	Card reset register	0x0078	R/W
BMOD_REG	Burst mode transfer configuration register	0x0080	R/W
PLDMND_REG	Poll demand configuration register	0x0084	WO
DBADDR_REG	Descriptor base address register	0x0088	R/W

IDSTS_REG	IDMAC status register	0x008C	R/W
IDINTEN_REG	IDMAC interrupt enable register	0x0090	R/W
DSCADDR_REG	Host descriptor address pointer	0x0094	RO
BUFADDR_REG	Host buffer address pointer register	0x0098	RO
CLK_EDGE_SEL	Clock phase selection register	0x0800	R/W

9.13 寄存器

SD/MMC CPU APB 。

Register 9.1. CTRL_REG (0x0000)

31 25	24	23 12	11	10	9	8	7	6	5	4	3	2	1	0	
(reserved)	(reserved)	(reserved)	CEATA_DEVICE_INTERRUPT_STATUS	SEND_AUTO_STOP_CCSD	SEND_CCSD	ABORT_READ_DATA	SEND_IRQ_RESPONSE	READ_WAIT	(reserved)	INT_ENABLE	(reserved)	DMA_RESET	FIFO_RESET	CONTROLLER_RESET	
0x00	1	0x00	0	0	0	0	0	0	0	0	0	0	0	0	Reset

CEATA_DEVICE_INTERRUPT_STATUS CE-ATA 设备上 写 。CE-ATA 设备 中 常被 (nIEN = 1)。 中 ， 1。(/写)

SEND_AUTO_STOP_CCSD 始 send_auto_stop_ccsd send_ccsd 一 ，不 单 。 时，SD/MMC 部 STOP (CMD12) CE-ATA 设备。 STOP ，RINTSTS Auto Command Done (ACD) 被 1， ACD 中 屏蔽， 中 。 Command Completion Signal Disable (CCSD) ，SD/MMC send_auto_stop_ccsd 。(/写)

SEND_CCSD 时，SD/MMC CCSD CE-ATA 设备。 CCS（RW_BLK）并 CE-ATA 设备 中 时 才 。一 CCSD 设备，SD/MMC send_ccsd 。 ACD 中 屏蔽， RINTSTS Command Done (CD) 被 1， 中 。 ：一 SEND_CCSD 被 ，个 时 期 CMD 上 CCSD。 ， 设备 CCS ， 边 CCSD 可 CE-ATA 设备。(/写)

ABORT_READ_DATA 操 期间，suspend ， 并 suspend 时 始 。一 suspend 始， ， 时 一个 。 时， 。(/写)

SEND_IRQ_RESPONSE 。 MMC 中 ， CMD40， MMC 中 。 时， SD/MMC 中 ， 1， 时 SD/MMC CMD40 并 。(/写)

READ_WAIT 操 SDIO 。(/写

INT_ENABLE 中 / 。0： ；1： 。(/写)

DMA_RESET DMA ， 1。 个 AHB 时 。(/写)

寄存器描述下一页继续。

Register 9.1. CTRL_REG (0x0000)

继上一页寄存器描述。

FIFO_RESET FIFO，1。操。：FIFO_RESET，个时期步（个时期），FIFO。（/写）

CONTROLLER_RESET，1。个AHB时期个cclk_in时期。（/写）

Register 9.2. CLKDIV_REG (0x0008)

CLK_DIVIDER3	CLK_DIVIDER2	CLK_DIVIDER1	CLK_DIVIDER0	
31 24	23 16	15 8	7 0	
0x000	0x000	0x000	0x000	Reset

CLK_DIVIDER3 时divider-3。时2*n*，*n*=0（1）。，1表2*1 = 2，0xFF表2*255 = 510，。MMC-Ver3.3-only，一个时，比不。（/写）

CLK_DIVIDER2 时divider-2。。（/写）

CLK_DIVIDER1 时divider-1。。（/写）

CLK_DIVIDER0 时divider-0。。（/写）

Register 9.3. CLKSRC_REG (0x000C)

(reserved)	CLKSRC_REG	
31 4	3 0	
0x000000	0x0	Reset

CLKSRC_REG 时可2个SD。个个比。，bit[1:0]0，bit[3:2]1。0比时[0:3]cclk_out[1:0]。

00：时0；
01：时1；
10：时2；
11：时3。

MMC-Ver3.3-only，一个时。cclk_out时时0，并不。（/写）

Register 9.4. CLKENA_REG (0x0010)

(reserved)	CCLK_ENABEL	
31 … 2	1 … 0	
0x00000	0x00000	Reset

CCLK_ENABEL 时 可 2 个 SD 时 一个 MMC 时 。

0: 时 ;

1: 时 。

MMC-Ver3.3-only , 一个 cclk_out, cclk_enable[0]。(/ 写)

Register 9.5. TMOUT_REG (0x0014)

DATA_TIMEOUT	RESPONSE_TIMEOUT	
31 … 8	7 … 0	
0x0FFFFFF	0x040	Reset

DATA_TIMEOUT 设 超时 , 可 设 超时 时 。 时 超时 才 始启 。 时 表 , 被 cclk_out。

: 超时 100 ms , 时 。 , 超时中 被 。(/ 写)

RESPONSE_TIMEOUT 设 超时 , 时 表 , cclk_out。(/ 写)

Register 9.6. CTYPE_REG (0x0018)

(reserved)	CARD_WIDTH8	(reserved)	CARD_WIDTH4	
31 18	17 16	15 2	1 0	
0x00000	0x00000	0x00000	0x00000	Reset

CARD_WIDTH8 个 一个比 ，表 8-bit 。

0： 8-bit ；

1：8-bit 。

Bit[17:16] 别 [1:0]。(/ 写)

CARD_WIDTH4 个 一个比 ，表 1-bit 4-bit 。

0：1-bit ；

1：4-bit 。

Bit[1:0] 别 [1:0]。 NUM_CARDS*2 个比 被 。(/ 写)

Register 9.7. BLKSIZ_REG (0x001C)

(reserved)	BLOCK_SIZE	
31 16	15 0	
0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0	0x00200	Reset

BLOCK_SIZE 。(/ 写)

Register 9.8. BYTCNT_REG (0x0020)

31 0	
0x000000200	Reset

BYTCNT_REG 表 。 ， 倍 。

长 ， 设 0。 0 时， /

。(/ 写)

Register 9.9. INTMASK_REG (0x0024)

(reserved)	SDIO_INT_MASK	INT_MASK	
31 … 18	17 … 16	15 … 0	
0x00000	0x00000	0x00000	Reset

SDIO_INT_MASK SDIO 中 屏蔽 , 个 一个比 。Bit[17:16] 别 [1:0]。 被屏蔽时, SDIO 中 测被 。“0” 屏蔽中 ,“1” 中 。 MMC-Ver3.3-only , 比 始 0。(/ 写)

INT_MASK 比 屏蔽不 中 。“0” 屏蔽中 ,“1” 中 。(/ 写)

Bit 15 (EBE): , / 写 (CRC);
Bit 14 (ACD): ;
Bit 13 (SBE/BCI): 启 Bit Error/Busy Clear 中 ;
Bit 12 (HLE): 写
Bit 11 (FRUN): FIFO underrun/overrun ;
Bit 10 (HTO): 超时 / Volt_switch_int;
Bit 9 (DRTO): 超时;
Bit 8 (RTO): 超时;
Bit 7 (DCRC): CRC ;
Bit 6 (RCRC): CRC ;
Bit 5 (RXDR): FIFO ;
Bit 4 (TXDR): FIFO ;
Bit 3 (DTO): ;
Bit 2 (CD): 毕;
Bit 1 (RE): ;
Bit 0 (CD): 测。

Register 9.10. CMDARG_REG (0x0028)

31 … 0	
0x000000000	Reset

CMDARG_REG 参 。(/ 写)

Register 9.11. CMD_REG (0x002C)

31	30	29	28	27	26	25	24	23	22	21	20 … 16	15	14	13	12	11	10	9	8	7	6	5 … 0	
START_CMD	(reserved)	USE_HOLE	(reserved)	(reserved)	(reserved)	(reserved)	CCS_EXPECTED	READ_CEATA_DEVICE	UPDATE_CLOCK_REGISTERS_ONLY		CARD_NUMBER	SEND_INITIALIZATION	STOP_ABORT_CMD	WAIT_PRVDATA_COMPLETE	SEND_AUTO_STOP	TRANSFER_MODE	READ/WRITE	DATA_EXPECTED	CHECK_RESPONSE_CRC	RESPONSE_LENGTH	RESPONSE_EXPECT	CMD_INDEX	
0	0	1	0	0	0	0	0	0	0	0	0x00	0	0	0	0	0	0	0	0	0	0	0x00	Reset

START_CMD 始 。一 CIU , 。 1 时, 不 尝试 写 。 尝试写 , 始中 被 1。一 被 并 SD_MMC_CEATA , 始中 Command Done 比 被 1。(/ 写)

USE_HOLE Use Hold 。(/ 写)

0: CMD DATA Hold ;

1: CMD DATA Hold 。

CCS_EXPECTED 期 (CCS) 。(/ 写)

0: CE-ATA 设备 中 不 (ATA 中 nIEN = 1); 不 设备 CCS 。

1: CE-ATA 设备 中 (nIEN = 0), 并 RW_BLK CE-ATA 设备 CCS 。 CE-ATA 设备 CCS , 1。SD/MMC RINTSTS Data Transfer Over (DTO) 并 DTO 中 被屏蔽, 产 中 。

READ_CEATA_DEVICE 操 标 。(/ 写)

0: 不 CE-ATA 设备 操 (RW_REG RW_BLK);

1: CE-ATA 设备 操 (RW_REG RW_BLK)。

1 表 CE-ATA 设备 被 。 CE-ATA 时 超时 。I/O 10 。SD/MMC CE-ATA 设备 时不 超时。(/ 写)

Register 9.12. CMD_REG (continued) (0x002C)

START_CMD	(reserved)	USE_HOLE	(reserved)	(reserved)	(reserved)	(reserved)	CCS_EXPECTED	READ_CEATA_DEVICE	UPDATE_CLOCK_REGISTERS_ONLY	CARD_NUMBER	SEND_INITIALIZATION	STOP_ABORT_CMD	WAIT_PRVDATA_COMPLETE	SEND_AUTO_STOP	TRANSFER_MODE	READ/WRITE	DATA_EXPECTED	CHECK_RESPONSE_CRC	RESPONSE_LENGTH	RESPONSE_EXPECT	CMD_INDEX	
31	30	29	28	27	26	25	24	23	22	21	20 16	15	14	13	12	11	10	9	8	7	6	5 0	
0	0	1	0	0	0	0	0	0	0	0	0x00	0	0	0	0	0	0	0	0	0	0	0x00	Reset

UPDATE_CLOCK_REGISTERS_ONLY (R/W)

0: 常 ;

1: 不 , 时 时 。 被 时 : CLKDIV、CLRSRC CLKENA (、时 时)。 已变时 时 , 所不必 。

常 , update_clock_registers_only = 0, BIU CIU: CMD、CMDARG、TMOUT、CTYPE、BLKSIZ BYTCNT。CIU 。 1 时, 被 SD_MMC_CEATA , Command Done 中 。(/ 写)

CARD_NUMBER 中 , 表 插槽编 。 MMC-Ver3.3-only , 2 。 SD , 2 。(/ 写)

SEND_INITIALIZATION (/ 写)

0: 不 始化 (80 个 期);

1: 始化 。

上 , , 必 80 个时 始化 一个 时 1, 便 始化时 。

STOP_ABORT_CMD (/ 写)

0: 中 , 中 不 。 中 不 , 设 0;

1: 中 , 。 时, 并 中 时, 1, CIU / 可 。

Register 9.13. CMD_REG (continued) (0x002C)

31	30	29	28	27	26	25	24	23	22	21	20 16	15	14	13	12	11	10	9	8	7	6	5 0	
START_CMD	(reserved)	USE_HOLE	(reserved)	(reserved)	(reserved)	(reserved)	(reserved)	CCS_EXPECTED	READ_CEATA_DEVICE	UPDATE_CLOCK_REGISTERS_ONLY	CARD_NUMBER	SEND_INITIALIZATION	STOP_ABORT_CMD	WAIT_PRVDATA_COMPLETE	SEND_AUTO_STOP	TRANSFER_MODE	READ/WRITE	DATA_EXPECTED	CHECK_RESPONSE_CRC	RESPONSE_LENGTH	RESPONSE_EXPECT	CMD_INDEX	
0	0	1	0	0	0	0	0	0	0	0	0x00	0	0	0	0	0	0	0	0	0	0	0x00	Reset

WAIT_PRVDATA_COMPLETE (/ 写)

0: , ;

1: 。

wait_prvdata_complete = 0 常 时 。

card_number 上一个 。

SEND_AUTO_STOP (/ 写)

0: 时不 ;

1: 时 。

TRANSFER_MODE (/ 写)

0: ;

1: 。 不 。

READ/WRITE (/ 写)

0: ;

1: 写 。

不 。

DATA_EXPECTED (/ 写)

0: 不 ;

1: 。

CHECK_RESPONSE_CRC (/ 写)

0: 不 查;

1: 查 CRC。

不 CRC 。 CRC 查 便

CRC 查。

RESPONSE_LENGTH (/ 写)

0: ;

1: 长 。

RESPONSE_EXPECT (/ 写)

0: 不 ;

1: 。

CMD_INDEX 。(/ 写)

Register 9.18. MINTSTS_REG (0x0040)

(reserved)	SDIO_INTERRUPT_MSK	INT_STATUS_MSK	
31 18	17 16	15 0	
0	0x0	0x00000	Reset

SDIO_INTERRUPT_MSK SDIO 中 屏蔽 , 个 一个比 。Bit[17:16] 别 1 0。
sdio_int_mask 被 1 时，SDIO 中 才 (屏蔽 中)。( )

INT_STATUS_MSK 中 屏蔽 中 被 1 时，中 才 。( )

Bit 15 (EBE): , / 写 (CRC)；
Bit 14 (ACD): ；
Bit 13 (SBE/BCI): 启 Bit Error/Busy Clear 中 ；
Bit 12 (HLE): 写
Bit 11 (FRUN): FIFO underrun/overrun ；
Bit 10 (HTO): 超时；
Bit 9 (DRTO): 超时；
Bit 8 (RTO): 超时；
Bit 7 (DCRC): CRC ；
Bit 6 (RCRC): CRC ；
Bit 5 (RXDR): FIFO ；
Bit 4 (TXDR): FIFO ；
Bit 3 (DTO): ；
Bit 2 (CD): 毕；
Bit 1 (RE): ；
Bit 0 (CD): 测。

Register 9.19. RINTSTS_REG (0x0044)

(reserved)	SDIO_INTERRUPT_RAW	INT_STATUS_RAW
31 16	17 16	15 0
0x00000	0x0	0x00000

Reset

SDIO_INTERRUPT_RAW SDIO 中，一个 一个中 。Bit[17:16] 别 1 0。
比 把 中 ，写0 。(/ 写)
0: SDIO 中 ;
1: SDIO 中 。
MMC-Ver3.3-only ， 比 始 0。 中 始 中 ，而不 中 屏蔽
。(/ 写)

INT_STATUS_RAW 比 把 中 ，写0 。 中 屏蔽 ，
中 被记录。(R/W)
Bit 15 (EBE): ， / 写 (CRC);
Bit 14 (ACD): ;
Bit 13 (SBE/BCI): 启 Bit Error/Busy Clear 中 ;
Bit 12 (HLE): 写
Bit 11 (FRUN): FIFO underrun/overrun ;
Bit 10 (HTO): 超时;
Bit 9 (DRTO): 超时;
Bit 8 (RTO): 超时;
Bit 7 (DCRC): CRC ;
Bit 6 (RCRC): CRC ;
Bit 5 (RXDR): FIFO ;
Bit 4 (TXDR): FIFO ;
Bit 3 (DTO): ;
Bit 2 (CD): 毕;
Bit 1 (RE): ;
Bit 0 (CD): 测。

Register 9.20. STATUS_REG (0x0048)

(reserved)	(reserved)	FIFO_COUNT	RESPONSE_INDEX	DATA_STATE_MC_BUSY	DATA_BUSY	DATA_3_STATUS	COMMAND_FSM_STATES	FIFO_FULL	FIFO_EMPTY	FIFO_TX_WATERMARK	FIFO_RX_WATERMARK	
31	30	29 17	16 11	10	9	8	7 4	3	2	1	0	
0	0	0x000	0x00	1	1	1	0x01	0	1	1	0	Reset

FIFO_COUNT FIFO ，FIFO 中被 。( )

RESPONSE_INDEX 一个 ，包括 。( )

DATA_STATE_MC_BUSY 。( )

DATA_BUSY 始 card_data[0] 版。( )

0: 不 ;

1: 。

DATA_3_STATUS 始 card_data[3]， 查 。( )

0: 不 ;

1: 。

COMMAND_FSM_STATES FSM 。( )

0: ;

1: 始 ;

2: 始 ;

3: ;

4: 参 ;

5: CRC7;

6: ;

7: 始 ;

8: IRQ ;

9: ;

10: ;

11: ;

12: CRC7;

13: ;

14: 完 NCC;

15: ， - 。

FIFO_FULL FIFO 。( )

FIFO_EMPTY FIFO 。( )

FIFO_TX_WATERMARK FIFO ，不 必 。( )

FIFO_RX_WATERMARK FIFO ，不 必 。( )

Register 9.21. FIFOTH_REG (0x004C)

(reserved)	DMA_MULTIPLE_TRANSACTION_SIZE	(reserved)	RX_WMARK	(reserved)	TX_WMARK	
31	30 28	27	26 16	15 12	11 0	
0	0x0	0	x x x x x x x x x x x	0 0 0 0	0x0000	Reset

DMA_MULTIPLE_TRANSACTION_SIZE , DMA 个 SRC/DEST_MSIZE 。000: 1 ; 001: 4 ; 010: 8 ; 011: 16 ; 100: 32 ; 101: 64 ; 110: 128 ; 111: 256 。(/ 写)

RX_WMARK 时 FIFO 。 FIFO (FIFO_RX_WATERMARK) 时, DMA/FIFO 被 。 包 期间, , , 。 DMA , FIFO (RXDR) 中 时, 产 中 , 所不 DMA 。 , 不产 中 。 时 中 时, 。 DMA , 包 时, , DMA 中 设 单 。(/ 写)

TX_WMARK 时 FIFO 。 FIFO (FIFO_TX_WATERMARK) 时, DMA/FIFO 被 。 中 , 中 。 包 期间, , 。 DMA , FIFO (TXDR) 中 时, 产 中 , 所不 DMA 。 包 期间, 一个中 时, FIFO (不 FIFO CIU , FIFO 可 不)。 DMA , 包 时, 一 比 , DMA 单个 期, 。(/ 写)

Register 9.22. CDETECT_REG (0x0050)

(reserved)	CARD_DETECT_N	
31 2	1 0	
0x0	0x0	Reset

CARD_DETECT_N card_detect_n (个 一个比) 。0 表 。 NUM_CARDS 被 。( )

Register 9.23. WRTPRT_REG (0x0054)

(reserved)	WRITE_PROTECT
31 … 2	1 … 0
0x0	0x0

Reset

WRITE_PROTECT card_write_prt （ 个 一个比 ） 。1 表 写保 。 NUM_CARDS 被 。( )

Register 9.24. TCBCNT_REG (0x005C)

31 … 0
0x000000000

Reset

TCBCNT_REG CIU 。( )

Register 9.25. TBBCNT_REG (0x0060)

31 … 0
0x000000000

Reset

TBBCNT_REG /DMA BIU FIFO 间 。( )

Register 9.26. DEBNCE_REG (0x0064)

(reserved)	DEBOUNCE_COUNT
31 … 24	23 … 0
0 0 0 0 0 0 0 0	0x0000000

Reset

DEBOUNCE_COUNT 波 时 (clk) 。 时间 5 ~ 25 ms， 插 时 不 。(/ 写)

Register 9.27. USRID_REG (0x0068)

31 … 0	
0x000000000	Reset

USRID_REG 识别 , 设 。 可 。(/ 写)

Register 9.28. RST_N_REG (0x0078)

(reserved) 31 … 2	RST_CARD_RESET 1 … 0	
0	0x1	Reset

RST_CARD_RESET 。1: ; 0: 。 比 , 被重 始化 CARD_RESET[0] 被 1'b0 0。CARD_RESET[1] 被 1'b0 1。被 比 NUM_CARDS 。(/ 写)

Register 9.29. BMOD_REG (0x0080)

(reserved) 31 … 11	BMOD_PBL 10 … 8	BMOD_DE 7	(reserved) 6 … 2	BMOD_FB 1	BMOD_SWR 0	
0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0	0x0	0	0x00	0	0	Reset

BMOD_PBL 可编 长 。 一个 IDMAC 中 。IDMAC 上 始 时 尝试 PBL 中 。 1、4、8、16、32、64、128 256。 FIFOTH MSIZE 。 已 , 写 FIFOTH 。 一个编 , : 000: 1 ; 001: 4 ; 010: 8 ; 011: 16 ; 100: 32 ; 101: 64 ; 110: 128 ; 111: 256 。

PBL , 并 , 不 表 。(/ 写)

BMOD_DE IDMAC 。 IDMAC 。(/ 写)

BMOD_FB 。 AHB 。 时, AHB 常 始期间 SINGLE、INCR4、INCR8 INCR16。 时, AHB SINGLE INCR 操 。(/ 写)

BMOD_SWR 。 时, DMA 部 。一个时 期 。(/ 写)

Register 9.30. PLDMND_REG (0x0080)

31	0	
0x000000000		Reset

PLDMND_REG 。 设 表 OWN , FSM 。 个 写 , IDMAC FSM 常 表 操 。 一个 写 , PD 写 。(写)

Register 9.31. DBADDR_REG (0x0088)

31	0	
0x000000000		Reset

DBADDR_REG 表 表 始。包 一个 表 。 (LSB bit) [1:0] 被忽 , 并 IDMAC 部 部 , LSB 可被视 。(/ 写)

Register 9.33. IDINTEN_REG (0x0090)

(reserved)	IDINTEN_AI	IDINTEN_NI	(reserved)	IDINTEN_CES	IDINTEN_DU	(reserved)	IDINTEN_FBE	IDINTEN_RI	IDINTEN_TI	
31 … 10	9	8	7 … 6	5	4	3	2	1	0	
0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0	0	0	0 0	0	0	0	0	0	0	Reset

IDINTEN_AI 常中 。(/ 写)
1 时， 常中 。 : IDINTEN [2]: 中 ; IDINTEN [4]: DU 中 。

IDINTEN_NI 中 。(/ 写)
1 时， 常中 。重 时， 常中 。 :
IDINTEN[0]: 中 ;
IDINTEN[1]: 中 。

IDINTEN_CES 中 。 1 时 中 。(/ 写)

IDINTEN_DU 表不可 中 。 常中 一 设 时， DU 中 。(/ 写)

IDINTEN_FBE 。 常中 一 设 时，
中 。 时， 中 被 。(/ 写)

IDINTEN_RI 中 。 常中 一 设 时， 中 。 时，
中 被 。(/ 写)

IDINTEN_TI 中 。 常中 一 设 时， 中 。 时，
中 被 。(/写)

Register 9.34. DSCADDR_REG (0x0094)

31 … 0	
0x000000000	Reset

DSCADDR_REG 表 , 操 期间 IDMAC , 并 时 。
IDMAC 表 始 。( )

Register 9.35. BUFADDR_REG (0x0098)

31 … 0	
0x000000000	Reset

BUFADDR_REG , 操 期间 IDMAC , 并 时 。
IDMAC 。( )

Register 9.36. CLK_EDGE_SEL (0x0800)

(reserved)	CCLKIN_EDGE_N	CCLKIN_EDGE_L	CCLKIN_EDGE_H	CCLKIN_EDGE_SLF_SEL	CCLKIN_EDGE_SAM_SEL	CCLKIN_EDGE_DRV_SEL	
31 21	20 17	16 13	12 9	8 6	5 3	2 0	
0x000	0x1	0x0	0x1	0x0	0x0	0x0	Reset

CCLKIN_EDGE_N CCLKIN_EDGE_L 。(/写)

CCLKIN_EDGE_L 时 , 比 CCLKIN_EDGE_H 。(/写)

CCLKIN_EDGE_H 时 , 比 CCLKIN_EDGE_L 。(/写)

CCLKIN_EDGE_SLF_SEL 部时 , 90 、180 270 。(/写)

CCLKIN_EDGE_SAM_SEL 时 , 90 、180 270 。(/写)

CCLKIN_EDGE_DRV_SEL 时 , 90 、180 270 。(/写)

10 以太网 (MAC)

10.1 概述

Ethernet 主要特性

部 层 (Ethernet PHY)，ESP32 可 (Ethernet MAC) 按 IEEE 802.3 标 。 见 10-1。 遍 (LAN) (WAN) 播 。

图 10-1. Ethernet MAC 功能概述

ESP32 MAC 标 ：

- IEEE 802.3-2002， MAC。
- IEEE 1588-2008 标 ， 时 步 。
- IEEE 802.3 标 ： (MII) 信 (RMII)。

MAC 层特性

- 部 PHY 10/100 Mbit/s
- 可 IEEE802.3 MII RMII 部 PHY
- 半
  - 半 CSMA/CD
  - IEEE 802.3x
  - 时可
  - 半 时 背
  - 操 中 ， 时间
- 报 始 (SFD) 插 、
- 可 CRC pad
- 长 ， pad
- 可编 长 ， 16 KB

- 可编 间 (IFG)（40-96 时间， 8 步长）
- :
  - 8 个 48 ， 个 操
  - 8 个 48 SA 比 查， 个 操
  - 可 播
  - ， 可 ， 视
  - 包时（ 时） 一 报
- 包 别 32
- 单 、
- MDIO PHY 设备
- 中 IPv4 TCP 包
- 中 查 IPv4 IPv4/IPv6 包中 TCP、UDP ICMP
- 时间 （ 参 IEEE 1588-2008）。 个 时 64 时间 。
- FIFO：一个 可编 2 KB FIFO 一个 可 （ 认 64 个 ） 2 KB FIFO
- FIFO 时， EOF ， FIFO 插 ， 而 FIFO
- ，可 时 ， 不
- 可
- FIFO 中 扩 脉 ，
- MAC 时
- 时 重 （ 一 ， 见 10.2.1.2）
- 、 、
- Tx FIFO
- IPv4 TCP、UDP ICMP 并 插 中

Ethernet 结构框图

Ethernet 10-2 。

图 10-2. Ethernet 功能框图

Ethernet MAC 层 包括 EMAC_CORE，EMAC_MTL (MAC Transition Layer)，EMAC_DMA (Direct Memory Access) 三层 MAC 层 ， 别 Rx Tx 个方 ， 方 部 AHB APB ， 方 部 MII RMII 部 PHY ， 。

10.2 EMAC_CORE

MAC PHY 方 。 PHY 一 。MAC MAC (MTI)，MAC (MRI) MAC (MCI) 侧（DMA 侧） 。

10.2.1 传输操作

MTL 并 SOF（ 始） 时启 操 。 测 SOF 时，MAC 并 始 RMII MII。 RMII MII 时间 ， IFG ， 报 SFD（ 始 ） 时间 半 。 ， ，MAC 不 MTL 。

EOF（ ） MAC ，MAC 常 ，并 (Transmit Status) MTL。 中（ 半 ） 常 ， MAC MTL 。 并 ， 一个 SOF。 中 测 MAC 重试 时，MTL SOF 始重 。

MTL 不 期间 ， MAC 。 MTL 常 中， MAC 一个 SOF 而 一 EOF， 方心 SOF，并 视 一 。

10.2.1.1 发送流量控制

， （Flow Control Register 中 TFE/Transmit Flow Control ） 1 时， MAC 并 。 CRC 一 并 。可 方 启 。

Flow Control Register 中 FCB (Flow Control Busy) 1 FIFO 时， 。

- Flow Control Register 中 FCB 1 ，MAC 并 单个 。 中 时间 Flow Control Register 中编 时间。 中 时间 长 时间， 必 编 时间 （Flow Control Register 中 PT/Pause Time），一 。
- FIFO 时 ，MAC 并 。 中 时间 Flow Control Register 中编 时间 。 时间 ，FIFO 可 时 （Flow Control Register 中 PLT (Pause Low Threshold) ）时间保 ， 个 。 FIFO 保 ，一 重 。 采 时间 不 ，MAC 一个 时间 ， 表 备 。

10.2.1.2 冲突期间的重新发送

半 ，MAC 时，可 MAC 上 。MAC 重试。 间， 重 并 FIFO 中 。 超 96 个 MAC ， FIFO DMA 可 。 超 MAC 时， 重 。

，Tx FIFO ， 波 ，jabber 超时， 波， ，MAC 可 中 。 而中 时，MAC 重 。

10.2.2 接收操作

MAC RMII MII 上 测 SFD 时 操 。 ，MAC 剥 SFD。 查报 CRC FCS 。 中， 。 ， MAC 中。

MAC Rx FIFO。 FIFO 一 超 （Operation Mode 中 Receive Threshold Control/RTC ），并 DMA， DMA 可 AHB 。

认 ， FIFO 64 个 （ Operation Mode 中 Receive Threshold Control/RTC ） 包时， ， 可 DMA。DMA AHB ， FIFO ， 个 包。 EOF ， 并 DMA 。

Rx FIFO （ Operation Mode 中 Receive Store and Forward/RSF ），并 。 ， 不 被 ， 时才 ，而 时 FIFO 。

10.2.2.1 接收协议

包 ， 报 SFD。 测 SFD ，MAC 始 FIFO ， SFD 一个 （ 标 ） 始 。 IEEE 1588 时间 ， MII 上 测 SFD 时， 时间 。 MAC 并 ， 时间 。

长 / 0x600 并 MAC CRC/pad ， MAC FIFO （ 不超 长 / 中 ），始 （包括 FCS ）。 长 / 0x600， 不 CRC ，MAC Rx FIFO 。 认 ， MAC 时 ， ，超 2048 个 （DA+SA+LT+ +pad+FCS） 被 。可 MAC 中 (WD/Watchdog Disable) 编 。 ， 时 ， 16 KB 并 超时 。

10.2.2.2 接收帧控制器

MAC 中 RA (Receive All) , MAC 标/ 。
不 不 , 矮 、CRC , 一 。 测 败时, 被
不 。 参 已变时, (DA-SA) 败, 被 并
(长 、CRC 矮 1), 败。

10.2.2.3 接收流量控制

MAC 测 并 , 时间 ( )。可
Flow Control Register 中 (Receive Flow Control Enable/RFCE) 测 。
, 始 视 标 播 (0x0180 C200 0001) 。 测
(标 保 标), MAC Frame Filter Register 中 (Pass Control
Frames/PCF) 。

MAC 、操 时 。 64 个 , 并
不 CRC , MAC , 时间 时间 时
(10/100 Mbit/s , 64)。 时, 测 一个 时间 , MAC
时间并 。

(0x8808)、操 (0x00001) 长 (64) 不 , CRC
, MAC 不 。

播 标 , MAC 。

单播 标 , MAC DA MAC 0 Flow Control
Register 中 UPFD (Unicast Pause Frame Detect) 1 (测 单播 标) 。
PCF (Frame Filter Register 中 Pass Control Frames [7:6]) 可
。

10.2.2.4 接收多帧的操作处理

可 , FIFO , 可 中 。

10.2.2.5 错误处理

MAC EOF Rx FIFO , 产 上 并 个 。 上 ,
(RDESO[11]) 一个部 。 Operation Mode Register 中 FTF (Flush Transmit FIFO)
FUGF (Forward Undersized Good Frames) , Rx FIFO 可 。 FIFO
, 可 并 。

, Rx FIFO SOF 时, 长 可 , 可 个 。DMA 可
FIFO 。 , 部
并 。 FIFO 可 , 可 启 一 。

10.2.2.6 接收状态字

时, MAC (DMA 侧) 。 RDES0 中 [31:0]
。

10.3 MAC 中断控制器

MAC 可 不 中 。

Interrupt Status 可 MAC 中 。可 中 屏蔽 中 屏蔽 1 中 。

中 中 。必 才 中 。 , 中 3 设 , 包 。必 PMT Control and Status 才 中 。

10.4 MAC 地址的过滤

查 标 并 报 。 查 不 参 ( )。 可 识别 : 播 播 。 (MAC) 查。

10.4.1 单播目标地址过滤

MAC 8 个 单播 MAC 。 (中 bit[1]), MAC 单播 48 编 MAC 比 。 认 , 始 EMACADDR0, EMACADDR0 ~ EMACADDR7 单 。 (EMACADDR0 ~ EMACADDR7) 个 DA 比 时, 可 中 屏蔽 1 屏蔽 。 DA 。

10.4.2 多播目标地址过滤

可 中 PAM (Pass All Multicast) 1, MAC 编 播 。 PAM (Pass All Multicast) , MAC 中 bit[2] (必 保) 播 。

, 播 编 MAC 标 EMACADDR0 ~ EMACADDR7 比 。

。

10.4.3 广播地址过滤

认 , MAC 不 播 。 , 中 DBF (Disable Broadcast Frames) 1 MAC 编 播 , 播 。

10.4.4 单播源地址过滤

MAC 可 。 认 , AFM SA SA 中 编 。可 中 bit[30] 1, MAC [1:3] 包 SA 不 DA 比 。 SA 。 中 SAF (Source Address Filter Enable) 1, MAC SA 。 , SA 中 (表 10-11)。

SAF (Source Address Filter Enable) 1 时, SA DA ,

。 一个 。 个 必 , 才 。

10.4.5 反向过滤操作

标 , 可 时 。 别 中 DAIF SAIF 。DAIF 时 单播 播 DA 。 操 中, DAIF 1 时, 单播/ 播 标 。 , SAIF 1 时, 单播 SA 。

个表按 标 。

表 10-1. 目标地址过滤

	PM	PF	DAIF	PAM	DB	DA
播	1	X	X	X	X	
	0	X	X	X	0	
	0	X	X	X	1	不
单播	1	X	X	X	X	
	0	X	0	X	X	/ 时
	0	X	1	X	X	/ 时不
	0	1	0	X	X	/ 时
	0	1	1	X	X	/ 时不
播	1	X	X	X	X	
	X	X	X	1	X	
	0	X	0	0	X	/ 时 ，并 PCF = 0x 时
	0	1	0	0	X	/ 时 ，并 PCF = 0x 时
	0	X	1	0	X	/ 时不 ，并 PCF = 0x 时
	0	1	1	0	X	/ 时不 ，并 PCF = 0x 时

表

10.4.6 好的发送帧与接收帧

，视“ ”。 ，如中 ，认 ：

- Jabber 超时
- 波 / 波
-
-
-
-

不 ，认 “ ”：

- CRC
- 矮 （ 64 ）
- （ 10/100 Mbit/s）
- 长 （ ）
- 超 包 （ ，超 ）
- MII_RXER

，：

- 标记 = 1518
- VLAN = 1522

10.5 EMAC_MTL（MAC 传输层）

MAC 层 FIFO 时 间 。MTL 层 MAC 间 。 可 时 MAC 时 间 32，采 单 FIFO 操 。

10.6 PHY 接口

DMA 个 ：

- (CSR)
- 表

见 表 表 。

10.6.1 MII（介质独立接口）

(MII) 10 Mbit/s 100 Mbit/s MAC 层 PHY 间 。

10.6.1.1 MII 与 PHY 间的接口信号

MII 10-3 。

图 10-3. MII 接口

MII :

- MII_TX_CLK：Tx 。 Tx 参 。 : 10 Mbit/s 2.5 MHz; 100 Mbit/s 25 MHz。
- MII_TXD[3:0]: 。 4 个一 , MAC 层 步 , MII_TX_EN 才 ( )。MII_TXD[0] , MII_TXD[3] 。MII_TX_EN , 不 PHY 产 。
- MII_TX_EN: 。 表 MAC MII 半 (4 bits)。 必 报 半 步 (MII_TX_CLK)，并 半 MII 必 保 步。
- MII_RX_CLK：RX 。 RX 参 。 : 10 Mbit/s 2.5 MHz; 100 Mbit/s 25 MHz。
- MII_RXD[3:0]: 。 4 个一 , PHY 步 , MII_RX_DV 才 ( )。MII_RXD[0] , MII_RXD[3] 。 MII_RX_DV 、MII_RX_ER , MII_RXD[3:0] 表 PHY 。
- MII_RX_DV: 。 表 PHY MII 并 半 。 必 半 步 MII_RX_CLK，并 一 保 步 半 。 必 半 一个 期 。 , MII_RX_DV 必 间 上 , 始 间 不 SFD 间 。
- MII_CRS: 波 。 , PHY 。 , PHY 。PHY 必 保 MII_CS 保 。 Tx Rx 保 步。 , 。
- MII_COL: 测 。 测 上 , PHY 必 测 , 并 , 测 必 保 。 Tx Rx 保 步。 , 。

- MII_RX_ER：。必保一个个期(MII_RX_CLK)，后MAC层测。
- MDIO MDC：时。个IEEE 802.3标，PHY。见Station Management Agent (SMA) Interface。

10.6.1.2 MII 时钟

MII时，MII PHY Tx Rx个方时，MII_TX_CLK步Tx，MII_RX_CLK步Rx。中MII_RX_CLK时PHY，MII_TX_CLK扩部PLL扩部。10-4参表中时。

图 10-4. MII 时钟

10.6.2 RMII（精简介质独立接口）

RMII 10-5 。

图 10-5. RMII 接口

10.6.2.1 RMII 接口信号描述

(RMII) 10 Mbit/s 100 Mbit/s 设 部 PHY 间 。
IEEE 802.3u 标 ，MII 包括 16 个包 。RMII 7 个（62.5%）。

RMII :

- 10 Mbit/s 100 Mbit/s
- 参 时 必 50 MHz
- 参 时 必 部 MAC 部 PHY。PHY 2 设

10.6.2.2 RMII 时钟

RMII 时 10-6 。

图 10-6. RMII 时钟

10.6.3 Station Management Agent (SMA) 接口

10-4 ，MAC MDC MDIO PHY。 时 2.5 MHz。时 时 时 产 。 MDIO 写/ 操 期间，PHY 。 MDC 时 步 。

EMII EMII 见 表。

10.6.4 RMII 接口时序要求

本 RMII 时 。

图 10-7. 接收数据时序图

表 10-3. 接收数据时序要求

时序参数	描述	最小值	典型值	最大值	单位
t_{CYC}	时 期 (Clock cycle)	20	20	20	ns
t_{SU}	时 间 (Setup time)	4			ns
t_H	保 时 间 (Hold time)	1			ns
t_{ID}	(Input delay)	3	5	8	ns

图 10-8. 发送数据时序图

表 10-4. 发送数据时序要求

时序参数	描述	最小值	典型值	最大值	单位
t_{CYC}	时 期 (Clock cycle)	20	20	20	ns
t_{SU}	时 间 (Setup time)	4			ns
t_H	保 时 间 (Hold time)	1			ns
t_{OD}	(Output delay)	6	9	12	ns

10.7 以太网 DMA 特性

DMA 。 设备
(MTL)，新 设备 。 ， CPU
。DMA 包 ， 中 。 可 常
中 CPU， ， 时。

10.8 链表描述符

本 表 。 个 表 8 个 word 。

10.8.1 发送描述符

表 10-9 。表 10-5 表 10-10 表 。

	31							0
TDES0	OWN	Ctrl[30:26]	TTSE	Ctrl[24:18]	TTSS	Status[16:7]	Ctrl/status [6:3]	Status [2:0]
TDES1	Ctrl [31:29]	Reserved				Transmit Buffer Size[12:0]		
TDES2	Buffer Address [31:0]							
TDES3	Next Descriptor Address[31:0]							
TDES4	Reserved							
TDES5	Reserved							
TDES6	Transmit Frame Timestamp Low[31:0]							
TDES7	Transmit Frame Timestamp High[31:0]							

图 10-9. 发送描述符

表 10-5. 发送描述符 0 (TDES0)

[31]	OWN: Own Bit	1 时，表 DMA 。 时，表 。 DMA 中 时，DMA 。 设 一 ，设 一个 。 避 间 设 可 。
[30]	IC: Interrupt on Completion	1 ， 设 中 (Operation Mode Register[0])。 TDES0 [29] 1 时 。
[29]	LS: Last Segment	1 时， 表 中包 一 。 1 时，TDES1 中 TBS1 TBS2 不 。
[28]	FS: First Segment	1 时， 表 中包 一 。
[27]	DC: Disable CRC	1 时，MAC 不 冗 (CRC)。 TDES0 [28] 1 时 。
[26]	DP: Disable Pad	1 时，MAC 不 64 padding。 时，DMA padding CRC 64 个 中，并 DC (TDES0 [27]) ，CRC 。 TDES0 [28] 1 时才 。
[25]	TTSE: Transmit Timestamp Enable	1 时， IEEE1588 时间 。 TDES0 [28] 1 时 。
[24]	CRCR: CRC Replacement Control	1 时，MAC 重 CRC 包 个 。 保 CRC 中。 (TDES0 [28]) 1 时， 。 ， TDES0 [27] 1 时才 CRC 。
[23:22]	CIC: Checksum Insertion Control	插 。 表 编 ： • 2’b00： 闭 插 。 • 2’b01： IP 报 插 。 • 2’b10： IP 报 插 ， 报 不 中 。 • 2’b11： IP 报 插 ， 报 中 。 TDES0 [28] 1 时， 。
[21]	TER: Transmit End of Ring	1 时， 表 表 一个 表 。DMA 表 ， 一个 。
[20]	TCH: Second Address Chained	1 时， 中 个 一个 。 TDES0 [20] 1 时，TBS2 (TDES1 [28:16]) 一个“ ”。TDES0 [21] TDES0 [20]。 必 1。

[19:18]	VLIC: VLAN Insertion Control	1 时，MAC 上 VLAN 标记 标记。被 上 VLAN 标记，MAC 重 并 CRC 。 : • 2'b00：不 上 VLAN 标记。 • 2'b01： VLAN 标记， VLAN 。 • 2'b10：插 VLAN 标记，标记 VLAN 标记插 中 。 • 2'b11： VLAN 标记插 中 标记 中 VLAN 标记。 VLAN 。
[17]	TTSS: Transmit Timestamp Status	捕 一个时间 。 1 时，TDES2 TDES3 捕 时间 。 TDES0 [29] 1 时 。
[16]	IHE: IP Header Error	1 时， MAC IP 报 中 测 。 查 IPv4 包中 报 长 报 ， 不 ， 。 IPv6 ， 报 长 不 40 个 ， 报 。 ，IPv4 IPv6 " 长 / " 必 IP 报 版本 。 IPv4 ， "报 长 " 0x5， 。
[15]	ES: Error Summary	表 bit : • TDES0[14]: Jabber 超时 • TDES0[13]: • TDES0[11]: 波 • TDES0[10]: 波 • TDES0[9]: • TDES0[8]: • TDES0[2]: • TDES0[1]: • TDES0[16]: IP 报 • TDES0[12]: IP
[14]	JT: Jabber Timeout	1 时， 表 MAC jabber 超时。 EMAC-CONFIG_REG EMACJABBER （ 闭 Jabber） 1 时， 1。
[13]	FF: Frame Flushed	1 时， 表 DMA MTL 按 CPU 。
[12]	IPE: IP Payload Error	1 时， 表 MAC TCP，UDP ICMP IP 报 中 测 。 查 IPv4 IPv6 报 中 长 TCP，UDP ICMP 包 ，并 不 时 。
[11]	LOC: Loss of Carrier	1 时， 期间 波 （ ， 期间，一个 个 时 期 ，MII_CRS ）。 MAC 半 时， 中 。
[10]	NC: No Carrier	1 时， 中 PHY 波 被 。

表 10-7. 发送描述符 2 (TDES2)

[31:0]	Buffer Address Pointer	。

表 10-8. 发送描述符 3 (TDES3)

[31:0]	Next Descriptor Address	包 一个 。

表 10-9. 发送描述符 6 (TDES6)

[31:0]	TTSL: Transmit Frame Timestamp Low	DMA , 捕 时间 32 。 中 (LS) 被 1 时 时间 (TTSS) 1 时, 才 时间 。

表 10-10. 发送描述符 7 (TDES7)

[31:0]	TTSH: Transmit Frame Timestamp High	DMA , 捕 时间 32 。 中 LS 被 1 时 TTSS 1 时, 才 时间 。

10.8.2 接收描述符

表 10-10 。表 10-11 表 10-17 表 。

	31				0
RDES0	OWN	Status[30:0]			
RDES1	Ctrl	Reserved[30:16]	Ctrl [15:14]	Res	Receive Buffer Size[12:0]
RDES2	Buffer Address [31:0]				
RDES3	Next Descriptor Address[31:0]				
RDES4	Extended Status[31:0]				
RDES5	Reserved				
RDES6	Receive Frame Timestamp Low[31:0]				
RDES7	Receive Frame Timestamp High[31:0]				

图 10-10. 接收链表结构

表 10-11. 接收描述符 0 (RDES0)

[31]	OWN: Own Bit	1 时，表 DMA 。 时，表 。DMA 表 时 。
[30]	AFM: Destination Address Filter Fail	1 时，表 MAC 中 DA 波 败 。
[29:16]	FL: Frame Length	表 CPU 长 。 RDES0 [8] 被 1 RDES0 [14] 被 时， 。 IP （ 1）并 不 MAC 时， 长 包 个 。
[15]	ES: Error Summary	表 ： • RDES0[1]：CRC • RDES0[3]： • RDES0[4]： 超时 • RDES0[6]： • RDES0[7]： • RDES4[4:3]：IP 报 • RDES0[11]：上 • RDES0[14]： RDES0 [8] 1 时， 。
[14]	DE: Descriptor Error	1 时，表 超 被 ，并 DMA 不 一个 。 被 。 RDES0 [8] 1 时， 才 。
[13]	SAF: Source Address Filter Fail	1 时，表 SA MAC 中 SA 。
[12]	LE: Length Error	1 时，表 长 长 / 不 。 (RDES0 [5]) 时 。
[11]	OE: Overflow Error	1 时，表 MTL 中 而 被 丢。
[10]	VLAN: VLAN Tag	1 时，表 MAC 标记 VLAN 。VLAN 标记 VLAN 标记 设 查 VLAN 。
[9]	FS: First Descriptor	1 时，表 包 一个 。 一个 0， 个 始。 个 0， 一个 包 。
[8]	LS: Last Descriptor	1 时，表 一个 。

[7]	Timestamp Available, IP Checksum Error (Type1), or Giant Frame	时间 时, 1 时, 表 时间 写 6 (RDES6) 7 (RDES7)。 RDES0 [8] 被 1 时才 。 IP (1) 时, 1 表 一: • MAC 16 IPv4 报 不 。 • IPv4 报 查。 上 不 , 1 表 。 , 1,518 (VLAN , 1,522 ; GMAC_CONFIGREG bit[27] 1 时, 2,000)。 时, 9,018 (VLAN , 9,022)。
[6]	LC: Late Collision	1 时, 表 半 时 。
[5]	FT: Frame Type	1 时, 表 (LT 1,536)。 时, 表 IEEE 802.3 。 14 个 矮 。
[4]	RWT: Receive Watchdog Timeout	1 表 时 时 超时, 超时 被 。
[3]	RE: Receive Error	1 时, 表 期间 MII_RXDV 被 1, MII_RXER。
[2]	DE: Dribble Bit Error	1 时, 表 倍 (奇 个半)。 MII 。
[1]	CE: CRC Error	1 时, 表 上 冗 (CRC) 。 RDES0 [8] 1 时, 才 。
[0]	Extended Status Available/ Rx MAC Address	时间 IP (2) 时, 1 时表 扩 4 (RDES4) 中可 。 RDES0 [8] 1 时才 。 30 1 时 。 IP (2) 时, IP , 被 1。 可 IP TCP/UDP/ICMP IP 。 时间 IPC 部 被 时, 表 Rx MAC 。 1 时, 表 Rx MAC (1 15) DA 。 时, 表 Rx MAC 0 DA 。

表 10-12. 接收描述符 1 (RDES1)

[31]	Ctrl	1 时, RI (CSR5 [6]) 被 1, 。 而 RI 中 。
[30:29]	Reserved	保
[28:16]	Reserved	保

[15]	RER: Receive End of Ring	1 时，表 表 一个 。DMA 表 ， 一个 。
[14]	RCH: Second Address Chained	1 时，表 中 个 一个 ， 必 1。 1 时，RBS2 (RDES1 [28:16]) 一个 “ ” 。RDES1 [15] RDES1 [14]。
[13]	Reserved	保
[12:0]	RBS1: Receive Buffer 1 Size	表 一个 （ ）。 RDES2（buffer1 ） ， 必 4 倍 。 不 4 倍 时， 不 。 0， DMA 忽略 ，并 RCH (bit[14]) 2 一个 。

表 10-13. 接收描述符 2 (RDES2)

[31:0]	Buffer Address Pointer	表 。

表 10-14. 接收描述符 3 (RDES3)

[31:0]	Next Descriptor Address	包 一个 。

表 10-15. 接收描述符 4 (RDES4)

[31:28]	Reserved	保
[27:26]	Reserved	保
[25]	Reserved	保
[24]	Reserved	保
[23:21]	Reserved	保
[20:18]	Reserved	保
[17]	Reserved	保
[16]	Reserved	保
[15]	Reserved	保
[14]	Timestamp Dropped	1 时，表 时间 被捕 ， 时间 丢弃 MTL Rx FIFO 中 。
[13]	PTP Version	1 时，表 PTP IEEE 1588 版本 2 。 时， 版本 1 ， 0 时 。
[12]	PTP Frame Type	1 时，表 PTP 。 不 时，表 UDP-IPv4 UDP-IPv6 。 IPv4 IPv6 可 bit[6] bit[7] 中 。

[11:8]	Message Type	被编 。 • 3’b0000： PTP message received • 3’b0001：SYNC (时) • 3’b0010：Follow_Up (时) • 3’b0011：Delay_Req (时) • 3’b0100：Delay_Resp (时) • 3’b0101：Pdelay_Req (时) • 3’b0110：Pdelay_Resp (时) • 3’b0111：Pdelay_Resp_Follow_Up (时) • 3’b1000：Announce • 3’b1001：Management • 3’b1010：Signaling • 3’b1011-3’b1110：保 • 3’b1111： 保 PTP 包
[7]	IPv6 Packet Received	1 时， 表 包 一个 IPv6 包。 （MAC ） bit[10] (IPC) 1 时 。
[6]	IPv4 Packet Received	1 时，表 包 IPv4 包。 （MAC ） bit[10] (IPC) 1 时 。
[5]	IP Checksum Bypassed	1 时， 表 被 。
[4]	IP Payload Error	1 时， 表 MAC 16 IP （TCP，UDP ICMP ） 中 不 。 TCP，UDP ICMP 长 IP Header 中 长 不 时， 1。 bit[7] bit[6] 1 时， 。
[3]	IP Header Error	1 时， 表 MAC 16 IPv4 报 不 ， IP 报版本 不一 。 bit[7] bit[6] 被 1 时， 。
[2:0]	IP Payload Type	表 (COE) IP 报 中 。 COE IP 报 IP 而不 IP 报 ，COE 2’b00。 • 3’b000： IP • 3’b001：UDP • 3’b010：TCP • 3’b011：ICMP • 3’b1xx：保 7 6 1 时， 。

表 10-16. 接收描述符 6 (RDES6)

[31:0]	RTSH: Receive Frame Timestamp Low	捕 时间 32 。 一个 RDES0 [8] 一个 表 DMA 。

表 10-17. 接收描述符 7 (RDES7)

[31:0]	RTSH: Receive Frame Timestamp High	DMA , 捕 时间 32 。 一个 RDES0 [8] 一个 表 DMA 。

10.9 寄存器列表

不 。 表 中 写。

- Read Only (RO)
- Write Only (WO) 写
- Read and Write (R/W) /写
- Read, Write, and Self Clear (R/W/SC) /写/
- Read, Self Set, and Write Clear (R/SS/WC) / 设 /写
- Read, Write Set, and Self Clear (R/WS/SC) /写设 /
- Read, Self Set, and Self Clear or Write Clear (R/SS/SC/WC) / 设 / /写
- Read Only and Write Trigger (RO/WT) /写
- Read, Self Set, and Read Clear (R/SS/RC) / 设 /
- Read, Write, and Self Update (R/W/SU) /写/
- Latched-low (LL)
- Latched-high (LH)

DMA 配置和控制寄存器			
DMABUSMODE_REG		0x3FF69000	R/WS/SC
DMATXPOLLDEMAND_REG		0x3FF69004	RO/WT
DMARXPOLLDEMAND_REG		0x3FF69008	RO/WT
DMARXBASEADDR_REG	一个	0x3FF6900C	R/W
DMATXBASEADDR_REG	一个	0x3FF69010	R/W
DMASTATUS_REG	中 ,	0x3FF69014	R/SS/WC
DMAOPERATION_MODE_REG	操	0x3FF69018	R/SS/WC
DMAIN_EN_REG	中 闭/	0x3FF6901C	R/W
DMAMISSEDFR_REG		0x3FF69020	R/W
DMARINTWDTIMER_REG		0x3FF69024	R/W
DMATXCURRDESC_REG		0x3FF69048	RO
DMARXCURRDESC_REG		0x3FF6904C	RO
DMATXCURRADDR_BUF_REG		0x3FF69050	RO
DMARXCURRADDR_BUF_REG		0x3FF69054	RO
MAC 配置和控制寄存器			

EMACCONFIG_REG	MAC	0x3FF6A000	R/W
EMACFF_REG	设	0x3FF6A004	R/W
EMACMIIADDR_REG	PHY	0x3FF6A010	R/WS/SC
EMACMIIDATA_REG	PHY 写	0x3FF6A014	R/W
EMACFC_REG		0x3FF6A018	R/WS/SC(FCB) R/W(BPA)
EMACDEBUG_REG	调试	0x3FF6A024	RO
PMT_RWUFFR_REG	波	0x3FF6A028	RO
PMT_CSR_REG		0x3FF6A02C	RO
EMACLPI_CSR_REG	LPI	0x3FF6A030	RO
EMACLPITIMERSCONTROL_REG	LPI 时	0x3FF6A034	RO
EMACINTS_REG	中	0x3FF6A038	RO
EMACINTMASK_REG	中 屏蔽	0x3FF6A03C	R/W
EMACADDR0HIGH_REG	一个 6 MAC 16	0x3FF6A040	R/W
EMACADDR0LOW_REG	一个 6 MAC 32	0x3FF6A044	R/W
EMACADDR1HIGH_REG	个 6 MAC 16	0x3FF6A048	R/W
EMACADDR1LOW_REG	个 6 MAC 32	0x3FF6A04C	R/W
EMACADDR2HIGH_REG	三个 6 MAC 16	0x3FF6A050	R/W
EMACADDR2LOW_REG	三个 6 MAC 32	0x3FF6A054	R/W
EMACADDR3HIGH_REG	个 6 MAC 16	0x3FF6A058	R/W
EMACADDR3LOW_REG	个 6 MAC 32	0x3FF6A05C	R/W
EMACADDR4HIGH_REG	个 6 MAC 16	0x3FF6A060	R/W
EMACADDR4LOW_REG	个 6 MAC 32	0x3FF6A064	R/W
EMACADDR5HIGH_REG	个 6 MAC 16	0x3FF6A068	R/W
EMACADDR5LOW_REG	个 6 MAC 32	0x3FF6A06C	R/W
EMACADDR6HIGH_REG	七个 6 MAC 16	0x3FF6A070	R/W
EMACADDR6LOW_REG	七个 6 MAC 32	0x3FF6A074	R/W
EMACADDR7HIGH_REG	八个 6 MAC 16	0x3FF6A078	R/W
EMACADDR7LOW_REG	八个 6 MAC 32	0x3FF6A07C	R/W
EMACCSTATUS_REG		0x3FF6A0D8	RO
EMACWDOGTO_REG	超时	0x3FF6A0DC	R/W
EMAC_EX_CLKOUT_CONF_REG	RMII 时 设	0x3FF69800	R/W
EMAC_EX_OSCCLK_CONF_REG	RMII 时 半 设	0x3FF69804	R/W
EMAC_EX_CLK_CTRL_REG	时 部/ 部时	0x3FF69808	R/W
PHY 类型和 SRAM 配置寄存器			
EMAC_EX_PHYINF_CONF_REG	MII/RMII PHY	0x3FF6980C	R/W
EMAC_PD_SEL_REG	闭 Ethernet RAM， 不	0x3FF69810	R/W

10.10 寄存器

说明：所有 reserved 寄存器的值必须保持复位值。

Register 10.1. DMABUSMODE_REG (0x0000)

(reserved)	DMAMIXEDBURST	DMAADDRALIBEA	PBLX8_MODE	USE_SEP_PBL	RX_DMA_PBL	FIXED_BURST	PRI_RATIO	PROG_BURST_LEN	ALT_DESC_SIZE	DESC_SKIP_LEN	DMA_ARB_SCH	SW_RST	
31 27	26	25	24	23	22 17	16	15 14	13 8	7	6 2	1	0	
0 0 0 0 0	0	0	0	0	0x01	0	0x0	0x01	0	0x00	0	1	Reset

DMAMIXEDBURST FB(FIXED_BURST) 时，AHB INCR（ burst）始 长 超 16 ，而 长 16 16 AHB （INCRx SINGLE）。（ /写）

DMAADDRALIBEA FB 1 时，AHB 产 始 LS 。 FB 0， 一个 （ 始 ） ， 。（ /写）

PBLX8_MODE 时， 编 PBL(PROG_BURST_LEN) （Bit [22:17] Bit [13：8]） 8。 ，DMA PBL 8，16，32，64，128 256 。（ /写）

USE_SEP_PBL 时，Rx DMA Bit [22:17] 中 PBL。Bit [13：8] 中 PBL Tx DMA 操 。 时，Bit [13：8] 中 PBL 个 DMA 。（ /写）

RX_DMA_PBL 表 一个 Rx DMA 中 。 单 写 。 上 始 时，Rx DMA 尝试按 RPBL 。 RPBL(RX_DMA_PBL) 可 1，2，4，8，16 32。 。 USP(USE_SEP_PBL) 时 。（ /写）

FIXED_BURST AHB 。 1 时， 常 始 时， AHB SINGLE，INCR4，INCR8 INCR16 。 时，AHB SINGLE INCR 操 。（ /写）

PRI_RATIO Rx DMA Tx DMA 间 时， 中 比 。 1（DA） 时， 才 。Rx Tx 比 ：（ /写）

- 2’b00 —1：1
- 2’b01 —2：0
- 2’b10 —3：1
- 2’b11 —4：1

PROG_BURST_LEN 表 一 DMA 中 。 32， 步 ：1. 设 PBLx8 ；2. PBL(PROG_BURST_LEN) 。（ /写）

寄存器描述下一页继续。

Register 10.1. DMABUSMODE_REG (0x0000)

继上一页寄存器描述。

ALT_DESC_SIZE 1 时， 32 个 。(/写)

DESC_SKIP_LEN 个 间 。 一个 始。 DSL(DESC_SKIP_LEN) 时， 表 被 DMA 视 。(/写)

DMA_ARB_SCH 之 之 间 裁决。1'b0: RX：TX TX：RX 时， PR（Bit [15:14]） ，1'b1 （Rx Tx）。(/写)

SW_RST 1 时，MAC DMA 重 MAC MAC 部 。 ETH_MAC 时 中 操 ， 。 ETH_MAC 重 编 ， 一个 。(/写设 /)

Register 10.2. DMATXPOLLDEMAND_REG (0x0004)

31	0
0x000000000	Reset

TRANS_POLL_DEMAND 被写时，DMA ( ) 。 不可 ( )， ，并 ( ) Bit[2]（TU）被 1。 可 ， 。(/写)

Register 10.3. DMARXPOLLDEMAND_REG (0x0008)

31	0
0x000000000	Reset

RECV_POLL_DEMAND 被写 时，DMA ( ) 。 不可 ( )， ，并 Bit[7]（RU）被 1。 可 ， Rx DMA 。(/写)

Register 10.4. DMARXBASEADDR_REG (0x000C)

31	0
0x000000000	Reset

START_RECV_LIST 包 表中 一个 。DMA LSB Bit[1:0] 被 忽略，并被 部视 。 ， LSB 。(/写)

Register 10.5. DMATXBASEADDR_REG (0x0010)

31	0	
0x000000000		Reset

START_TRANS_LIST 包 表中 一个 。LSB Bit[1:0] 被忽略，
部被 DMA 视 。 ， LSB 。(/写)

Register 10.6. DMASTATUS_REG (0x0014)

31 30	29	28	27 26	25 23	22 20	19 17	16	15	14	13	12 11	10	9	8	7	6	5	4	3	2	1	0	
(reserved)	TS_TRI_INT	EMAC_PMT_INT	(reserved)	ERROR_BITS	TRANS_PROC_STATE	RECV_PROC_STATE	NORM_INT_SUMM	ABN_INT_SUMM	EARLY_RECV_INT	FATAL_BUS_ERR_INT	(reserved)	EARLY_TRANS_INT	RECV_WDT_TO	RECV_PROC_STOP	RECV_BUF_UNAVAIL	RECV_INT	TRANS_UNDFLOW	RECV_OVFLOW	TRANS_JABBER_TO	TRANS_BUF_UNAVAIL	TRANS_PROC_STOP	TRANS_INT	
0 0	0	0	0 0	0x0	0x0	0x0	0	0	0	0	0 0	0	0	0	0	0	0	0	0	0	0	0	Reset

TS_TRI_INT 表 ETH_MAC 时间 中 中 。 必 ETH_MAC 中 中 并 , 1’b0。( )

EMAC_PMT_INT 表 ETH_MAC PMT 中 中 。 必 MAC 中 PMT 才 中 并 , 1’b0。( )

ERROR_BITS , AHB 上 。 Bit[13]（FBI） 1 时 。 不 中 。( )

- 3’b000：Rx DMA 写 期间 。
- 3’b011：Tx DMA 期间 。
- 3’b100： Rx DMA 期间 。
- 3’b101：写 Tx DMA 期间 。
- 3’b110： Rx DMA 期间 。
- 3’b111： Tx DMA 期间 。

TRANS_PROC_STATE 表 DMA FSM 。 不 中 。

- 3’b000： : 。
- 3’b001： : 。
- 3’b010：保 。
- 3’b011： : 包。
- 3’b100： : 不可 。
- 3’b101： : 闭 。
- 3’b110：TIME_STAMP 写 。
- 3’b111： : 包 。

寄存器描述下一页继续。

Register 10.6. DMASTATUS_REG (0x0014)

继上一页寄存器描述。

RECV_PROC_STATE 表 DMA FSM 。 不 中 。( )

- 3’b000: : 。
- 3’b001: : 。
- 3’b010: 保 。
- 3’b011: : 包。
- 3’b100: : 不可 。
- 3’b101: : 闭 。
- 3’b110: TIME_STAMP 写 。
- 3’b111: : 包 。

NORM_INT_SUMM 中 中 中 时， 常中 :

- Bit[0]: 中 。
- Bit[2]: 不可 。
- Bit[6]: 中 。
- Bit[14]: 中 。

  被屏蔽 常中 。 一个 ，必 NIS (写 1）被 。(/ 设 /写)

ABN_INT_SUMM 中 中 中 时， 常中 :

- Bit[1]: 。
- Bit[3]: Jabber 超时。
- Bit[4]: FIFO 。
- Bit[5]: 。
- Bit[7]: 不可 。
- Bit[8]: 。
- Bit[9]: 超时。
- Bit[10]: 中 。
- Bit[13]: 重 。

  被屏蔽 常中 。 一个 ，必 AIS 时（ 写 1） 。(/ 设 /写)

寄存器描述下一页继续。

3FHJTUFS %."45"564@3&(Y

? B:7i í•b

&"3-:@3&$7@¾ÊVU %."A " ¥»Bñ" Ï ub'Èq_¾Ê•Æ ¾
7i ¥ #JU<3*Â H ¾Êb,b 1î!Â •b"

'"5"-@#64@&33@¾ÊVU?3 #JU< î•¥9Lp b'¾ÊÂ H M‹¥ %."„
c1> îµ¥9L`Ùb 1î!Â •b"

&"3-:@53"/4@¾ÊVU1?Â¥ $›†.Âž .5-?Â '*'0b 1î!Â •b"

3&$7@8%5@Â50VU¤lAÊ çH ¤lž'- HXÜVù '- AÊ ÑHª
$© b 1î!Â •b"

3&$7@130$@45lÉñÉÆT¸ ÿH ¾ÊÂ b 1î!Â •b"

3&$7@#6'@6/"7¾ÊVUö šµ¤l VÏ¥/Bñí•˜ iO %."íE |¾í•
˜b¤lVñ$ Tb1ë¯)Ø¤lí•˜ ö ‹÷¿í•˜¥îµƒi? o¤l}Ø³
pp 7bÂTÀµlž}Øhp 5¤lÉñ lž/BñMY¥.Æ Hë¯bºµ' %."
šµ-Bñ¤lí•˜H ¾ÊŒ$Â b 1î!Â •b"

3&$7@*/5ÊVU ¤lX›îb'¤l›îH 3%&4¥ #JU<Kª¥í•˜Ï$¯Êi
O+ç¥ ÿ• í•˜Ï$÷•b¤l)¿ › ÿb 1î!Â •b"

53"/4@6/%'-08¾ÊVU?ÂÏi .{ùW µ/nb?Â T i|/np Ê
5%&4 Â >b 1î!Â •b"

3&$7@07'-08¾ÊVU¤lÏi ¤lVñÏµn bÂT†s $.Âž‹¨ñ½ 5n
ÿ| 3%&4 <ÏÂ> b 1î!Â •b"

53"/4@+"##&3@¾ÊVU?Â +BCÇÑHS Ñ H ' vlÑV 3« ' +VNCP
$P?H¹ 3« H?3b'?3 +BCÇÑHS H .{Vñ$Ï¸iÉÆT¸ ÿbƒ
ö•Á?Â +BCÇÑHS5%&4 <S½>Â b 1î!Â •b"

53"/4@#6'@6/"7¾ÊVUö ?Â VÏšµ/Bñí•˜ iO %."íE |ñb.
{$ Tb #JU< VU?ÂÉñ ÿ Ðb1ë¯)Ø.{í•˜ ö ‹YV!Â 5%&4
< ¥÷¿í•˜¥îµƒ –ª? .{}Ø³p 7b 1î!Â •b"

53"/4@130$@45{01¸H¾Ê$Â b 1î!Â •b"

53"/4@*/5¾ÊVU .{X›îb.{›îª 5%&4¥ #JU<08/ ¯Êi í•˜Ï
÷•+ç¥ ÿ• b 1î!Â •b"

Register 10.7. DMAOPERATION_MODE_REG (0x0018)

31 27	26	25	24	23	22	21	20	19 17	16 14	13	12 11	10 9	8	7	6	5	4 3	2	1	0	
(reserved)	DIS_DROP_TCPIP_ERR_FRAM	RX_STORE_FORWARD	DIS_FLUSH_RECV_FRAMES	(reserved)	(reserved)	TX_STR_FWD	FLUSH_TX_FIFO	(reserved)	TX_THRESH_CTRL	START_STOP_TRANSMISSION_COMMAND	(reserved)	(reserved)	(reserved)	FWD_ERR_FRAME	FWD_UNDER_GF	DROP_GFRM	RX_THRESH_CTRL	OPT_SECOND_FRAME	START_STOP_RX	(reserved)	
0 0 0 0 0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	Reset

DIS_DROP_TCPIP_ERR_FRAM 1 时，ETH_MAC 不 checksum 包。 1 时，并 FWD_ERR_FRAME 0 时， 包 被 。(/写)

RX_STORE_FORWARD 1 时，MTL 层 把写 RX FIFO 包 。(/写)

DIS_FLUSH_RECV_FRAMES 1 时，RX DAM 不 表 不可 包。(/写)

TX_STR_FWD 1 时，并 个 TX FIFO 中时， 始 包。 时， TX_THRESH_CTRL 中 被忽略。(/写)

FLUSH_TX_FIFO 1 时，TX FIFO 中 被 认 。 ， 。(/写设 /)

TX_THRESH_CTRL TX FIFO 中 中 ， 始 。 个 时， 被 。 TX_STR_FWD 0 时， 才 。 ：3'b000：64；3'b001：128；3'b010：192；3'b011：256；3'b100：40；3'b101：32；3'b110：24；3'b111：1。(/写)

START_STOP_TRANSMISSION_COMMAND TX FIFO 中 中 ， 始 。 个 时， 被 。 TX_STR_FWD 0 时， 才 。 ：000：64, 001：128, 010：192, 011：256, 100：40, 101：32, 110：24, 111：16。(/写)

FWD_ERR_FRAME 0 时，RX FIFO （CRC ， ，超 ， 超时 ， ），。(/写)

FWD_UNDER_GF 1 时，RX FIFO 包括 PAD CRC （长 64 ），不 。

DROP_GFRM 1 时，ETH_MAC RX FIFO 中 超 。(/写)

寄存器描述下一页继续。

Register 10.7. DMAOPERATION_MODE_REG (0x0018)

继上一页寄存器描述。

RX_THRESH_CTRL 2’b00：64，2’b01：32，2’b10：96，2’b11：128. RX FIFO 中 个 时， DMA 始 中。2’b00：64；2’b01：32；2’b10：96；2’b11：128。(/写)

OPT_SECOND_FRAME 1 时， 一 被 TX DMA 始 一 。(/写)

START_STOP_RX 1 时，RX DMA 始 ， 0 时，RX DMA 。(/写)

Register 10.8. DMAIN_EN_REG (0x001C)

31														17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0	
(reserved)															DMAIN_NISE	DMAIN_AISE	DMAIN_ERIE	DMAIN_FBEE	(reserved)		DMAIN_ETIE	DMAIN_RWTE	DMAIN_RSE	DMAIN_RBUE	DMAIN_RIE	DMAIN_UIE	DMAIN_OIE	DMAIN_TJTE	DMAIN_TBUE	DMAIN_TSE	DMAIN_TIE	
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	Reset

DMAIN_NISE 1 时， 常中 。 时， 闭 常中 。 中 中 : Bit[0]: 中 。 Bit[2]: 不可 。 Bit[6]: 中 。 Bit[14]: 期 中 。(/写)

DMAIN_AISE 1 时， 常中 。 时， 闭 常中 。 中 中 :

- Bit[1]: 。
- Bit[3]: Jabber 超时。
- Bit[4]: 。
- Bit[5]: 。
- Bit[7]: 不可 。
- Bit[8]: 。
- Bit[9]: 超时。
- Bit[10]: 中 。
- Bit[13]: 重 。(/写)

DMAIN_ERIE 常中 (Bit[16]) 1 时， 中 。 时， 闭 中 。(/写)

DMAIN_FBEE 常中 (Bit[15]) 1 时， 重 中 。 时， 闭 中 。(/写)

DMAIN_ETIE 常中 (Bit[15]) 1 时， 中 。 时， 闭 中 。(/写)

DMAIN_RWTE 常中 (Bit 15) 1 时， 超时中 。 时， 闭 超时中 。(/写)

DMAIN_RSE 常中 (Bit[15]) 1 时， 中 。 时， 闭 中 。(/写)

DMAIN_RBUE 常中 (Bit[15]) 1 时， 不可 中 。 时， 闭 不可 中 。(/写)

DMAIN_RIE 常中 (Bit[16]) 1 时， 中 。 时， 闭 中 。(/写)

寄存器描述下一页继续。

Register 10.8. DMAIN_EN_REG (0x001C)

继上一页寄存器描述。

DMAIN_UIE 常中 (Bit[15]) 1 时， 中 。 时， 闭 中 。(/写)

DMAIN_OIE 常中 (Bit[15]) 1 时， 中 。 时， 闭 中 。(/写)

DMAIN_TJTE 常中 (Bit[15]) 1 时， Jabber 超时中 。 时， 闭 Jabber 超时中 。(/写)

DMAIN_TBUE 常中 (Bit[16]) 1 时， 不可 中 。 时， 闭 不可 中 。(/写)

DMAIN_TSE 常中 (Bit[15]) 1 时， 中 。 时， 闭 中 。(/写)

DMAIN_TIE 常中 (Bit[16]) 1 时， 中 。 时， 闭 中 。(/写)

Register 10.9. DMAMISSEDFR_REG (0x0020)

(reserved)	Overflow_BFOC	Overflow_FC	Overflow_BMFC	Missed_FC	
30 29	28	27 17	16	10 0	
0 0	0x0	0x0	0x0	0x0	Reset

Overflow_BFOC Overflow_FC (Bits[27:17]) 时， 1。 RX FIFO ， 时 。 ， 0， 表 。(/ 设 /)

Overflow_FC 表 个 而 上 上层 ，MTL FIFO 一 ， 个 1， ， 。(/ 设 /)

Overflow_BMFC Missed_FC (Bits[15:0]) 时， 1。 时 不可 ， 时 。 ， 0， 表 。(/ 设 /)

Missed_FC 表 不可 ， 个 而 上 上层 ，DMA 一 ， 个 1， ， 。(/ 设 /)

Register 10.10. DMARINTWDTIMER_REG (0x0024)

(reserved)	RIWTC
31 … 8	7 … 0
0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0	0x000

Reset

RIWTC 表 时 期 256。 Rx DMA RDES1 [31] 中 设 而 设 RI , 时 编 。 时 , RI 1, 时 。 RDES1[31] RI 设 , RI (RECV_INT) 被 时, 时 。(/写)

Register 10.11. DMATXCURRDESC_REG (0x0048)

31 … 0
0x000000000

Reset

TRANS_DECR_ADDR_PTR 时 。操 期间 DMA 。 表 。( )

Register 10.12. DMARXCURRDESC_REG (0x004C)

31 … 0
0x000000000

Reset

RECV_DECR_ADDR_PTR 时 。操 期间 DMA 。 表 。( )

Register 10.13. DMATXCURRADDR_BUF_REG (0x0050)

31 … 0
0x000000000

Reset

TRANS_BUFF_ADDR_PTR 时 。操 期间 DMA 。 表 。( )

Register 10.14. DMARXCURRADDR_BUF_REG (0x0054)

31 … 0
0x000000000

Reset

RECV_BUFF_ADDR_PTR 时 。操 期间 DMA 。 表 。( )

Register 10.15. EMACCONFIG_REG (0x1000)

31	30 28	27	26 24	23	22	21	20	19 17	16	15	14	13	12	11	10	9	8	7	6 5	4	3	2	1 0	
(reserved)	SAIRC	ASS2KP	(reserved)	EMACWATCHDOG	EMACJABBER	(reserved)	EMACJUMBOFRAME	EMACINTERFRAMEGAP	EMACDISABLECRS	EMACMII	EMACFESPEED	EMACRXOWN	EMACLOOPBACK	EMACDUPLEX	EMACRXIPCOFFLOAD	EMACRETRY	(reserved)	EMACPADCRCSTRIP	EMACBACKOFFLIMIT	EMACDEFERRALCHECK	EMACTX	EMACRX	PLTF	
0	0x0	0	0 0 0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0x0	0	0	0	0x0	Reset

SAIRC 插 。Bit[30] Bit[29:28] 个 MAC (0 1) 插 。(/写)

- 2'b0x: mti_sa_ctrl_i ati_sa_ctrl_i SA 。
- 2'b10: Bit[30] 0, MAC MAC 0 插 SA 中。 Bit[30] 1, MAC MAC 1 插 SA 。
- 2'b11: Bit[30] 0, MAC SA 中 MAC 0 。 Bit[30] 1, MAC SA 中 MAC 1 。

ASS2KP 1 时, MAC 长 不超 2,000 个 视 包。 Bit[20] (JE) 1 时, MAC 2K 视 。 Bit[20] (JE) 1 时, MAC 超 1,518 (标记 1,522) 视 。 Bit[20] 1 时, 。(/写)

EMACWATCHDOG 1 时, MAC 闭 上 时 。MAC 可 16,383 。 时, MAC 不 超 2048 (JE , 10,240) 超时 中设 。MAC 超 。(/写)

EMACJABBER 1 时, MAC 闭 上 Jabber 时 。MAC 可 长 16,383 。 时, 中 超 2048 (JE 设 , 10240) , MAC 。(/写)

EMACJUMBOFRAME 1 时, MAC 9,018 (9,022 VLAN 标记), 不 中报 。(/写)

EMACINTERFRAMEGAP 时间 IFG。(/写)

- 3'b000: 96 比 时间。
- 3'b001: 88 比 时间。
- 3'b010: 80 比 时间。
- 3'b111: 40 比 时间。 半 , IFG 64 比 时间 (IFG = 100)。

寄存器描述下一页继续。

Register 10.15. EMACCONFIG_REG (0x1000)

继上一页寄存器描述。

EMACDISABLECRS 时，MAC 半 期间 忽略 MII CRS 。 不 中 波 波 而产 。 时，MAC 波 而产 ， 可 中 。(/写)

EMACMII 。 10 100 Mbps 操 ， 1。 10 100 Mbps 操 中， FES(EMACFESPEED) 一 。(/写)

EMACFESPEED MII，RMII 。0：10 Mbps；1：100 Mbps。(/写)

EMACRXOWN 1 时， 半 TX_EN 中 时，MAC 闭 。 时， MAC 时 PHY 包。MAC 时， 不 。(/写)

EMACLOOPBACK 1 时，MAC 环回 MII。MII 时 (CLK_RX) 常 必 ， 时 部 。(/写)

EMACDUPLEX 1 时，MAC ，可 时 。 RO， 认 1’b1。(/写)

EMACRXIPCOFFLOAD 1 时，MAC 16 补 ， 补。 查 IPv4 报 (25/26 29/30（VLAN 标记） ，并 中 。MAC IP 报 报 （IPv4 报 ）16 中 (2 COE 时)。 时， 被 。(/写)

EMACRETRY 1 时，MAC 尝试一 。 MII 上 时，MAC 忽略 ，并 中报 。 时，MAC BL （Bit [6:5]） 设 尝试重试。 半 。(/写)

EMACPADCRCSTRIP 1 时， 长 1,536 ，MAC 才 上 Pad FCS 。 长 1,536 被 ，而不 Pad FCS 。 时，MAC ，而不 改 。(/写)

EMACBACKOFFLIMIT 避 ， 重试时重 安排下 ，MAC 时 时间 （10/100 Mbps 512 比 时间） (r)。 半 。00：k = min（n，10）。01：k = min（n，8）。10：k = min（n，4）。11：k = min（n，1），中 n = 重 尝试 。 r 0 <r <2k。(/写)

EMACDEFERRALCHECK 期 查 。(/写)

寄存器描述下一页继续。

Register 10.15. EMACCONFIG_REG (0x1000)

继上一页寄存器描述。

EMACTX 1 时，MAC 被 MII 上 。 被 时，MAC 被 闭，并 不 。(/写)

EMACRX 1 时， MAC MII 。 时，MAC 被 闭，并 不 MII 。(/写)

PLTF 个 始 。 MAC 时，才 报 。(/写)

- 2’b00：7 报 。
- 2’b01：5 报 。
- 2’b10：3 报 。

Register 10.16. EMACFF_REG (0x1004)

RECEIVE_ALL	(reserved)	SAFE	SAIF	PCF	DBF	PAM	DAIF	(reserved)	PMODE	
31	30 … 10	9	8	7 … 6	5	4	3	2 … 1	0	
0	0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0	0	0	0x0	0	0	0	0 0	0	Reset

RECEIVE_ALL 1 时，MAC ，丢不 。SA DA 中 （ 败）。 时， SA DA 。（ /写）

SAFE 1 时，MAC SA SA 中编 比 。 比 败， MAC 。 时，MAC SA 比 Rx SAF 。（ /写）

SAIF 1 时， 查 波 SA 比 。SA SA 被标记 SA 。 时，SA SA 不 被标记 SA 。（ /写）

PCF （包括单播 播 ）。（ /写）

- 2’b00：MAC 。
- 2’b01：MAC ， 。
- 2’b10： MAC 败，MAC 。
- 2’b11：MAC 。

：

- 1：MAC ， 2（RFE） 1 。
- 2： Bit 3（UP） 1 时， （DA） 播 EMACADDR0 。
- 3： Type 0x8808，OPCODE 0x0001。

DBF 1 ，AFM (Address Filtering Module) 播 ，并 设 。 时，AFM 播 。（ /写）

PAM 1 时， 表 播 （ 中 一 “1”） 。 时， 播 HMC 。（ /写）

DAIF 1 时， 单播 播 DA 比 。 时， 常 。（ /写）

PMODE 1 时， ，丢不 。PR (PRI_RATIO) 1 时， SA DA 波 败 。（ /写）

Register 10.17. EMACMIIADDR_REG (0x1010)

(reserved)	MIIDEV	MIIREG	MIICSRCLK	MIIWRITE	MIIBUSY	
31 … 16	15 … 11	10 … 6	5 … 2	1	0	
0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0	0x00	0x00	0x00	0	0	Reset

MIIDEV 表 32 个 PHY 设备中 被 一个。(/写)

MIIREG PHY 设备中 。(/写)

[MIICSRCLK]CSR 时 , 1.0 MHz ~ 2.5 MHz。(/写)

- 4’b0000: APB 时 80 MHz 时, MDC 时 APB_CLK/42。
- 4’b0011: APB 时 40 MHz 时, MDC 时 APB_CLK/26。

MIIWRITE 1 时, 表 MII 写操 。 1, 表 一个 操 , MII 。(/写)

MIIBUSY 写 PHYADDR PHY , 0。 PHY 期间, 1’b1, 表 写 。 被 MAC , 。 , PHY 写操 期间, PHY (MII) 保 , MAC 。 操 , 时, 5 才 。 写操 一个操 才 。 写 操 PHY 不 认 MAC, PHY 不 , 变化 (/写设 /)

Register 10.18. EMACMIIDATA_REG (0x1014)

(reserved)	MII_DATA	
31 … 16	15 … 0	
0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0	0x00000	Reset

MII_DATA 包 操 PHY 16 , 写操 写 PHY 16 。(/写)

Register 10.19. EMACFC_REG (0x1018)

PAUSE_TIME	(reserved)	PLT	UPFD	RFCE	TFCE	FCBBA	
31 … 16	15 … 6	5 … 4	3	2	1	0	
0x00000	0 0 0 0 0 0 0 0 0 0	0x0	0	0	0	0	Reset

PAUSE_TIME 保 时间 中 。 时间 MII 时 重 步， 标时 中 个时 期 ，才 写 。(/写)

PLT 时 重 。 始 Bit [31:16] 中 时间。， PT = 100H（256 个时 ）并 PLT = 01， 一个 228 个 时 时间（256-28） 个 。 表 不 ：(/写)

- 2’b00： 时间 4 个时 （PT-4 时 时间）。
- 2’b01： 时间 28 个时 （PT-28 时 ）。
- 2’b10： 时间 144 时 时间（PT-144 时 时间）。
- 2’b11： 时间 256 个时 （PT-256 个时 ）。时 时间被 MII 上 512 （64 个 ） 时间。

UPFD 一个 IEEE Std 802.3 中 一 播 时， 。 1 时，MAC 可 测 单播 。 个单播 EMACADDR0 EMACADDR0 中 。 时，MAC 测 一 播 。

RFCE 1 时，MAC ，并 闭 （ ）时间。 时， 闭 。(/写)

TFCE ， 1 时，MAC 操 。 时，MAC 中 操 被 ，MAC 不 。 半 ， 1 时，MAC 启 背 操 。 时， 闭背 。(/写)

FCBBA 启 ， TFCE 1， 半 背 。 ， 写 ， 1’b0。 启 ， 必 1’b1。 中， 被设 表 。 ，MAC 1’b0。 不 写 ， 被 。 半 ， 1（并 TFCE 1）时，背 MAC 。 背 期间， MAC 一个 时， 始 一个 案， 。 MAC 被 时，BPA 被 闭。(/写设 / ）(FCB)/（ /写）(BPA/Backpressure Activate)

Register 10.20. EMACDEBUG_REG (0x1024)

(reserved)	MTLTSFFS	MTLTFNES	(reserved)	MTLTFWCS	MTLTFRCS	MACTP	MACTFCS	MACTPES	(reserved)	MTLRFFLS	(reserved)	MTLRFRCS	MTLRFWCAS	(reserved)	MACRFFCS	MACRPES	
31 26	25	24	23	22	21 20	19	18 17	16	15 10	9 8	7	6 5	4	3	2 1	0	
0 0 0 0 0 0	0	0	0	0	0x0	0	0x0	0	0 0 0 0 0 0	0x0	0	0x0	0	0	0x0	0	Reset

MTLTSFFS 时， 表 MTL TxStatus FIFO 。 ，MTL 不 。( )

MTLTFNES 时，表 MTL Tx FIFO 不 ， 一 。( )

MTLTFWCS 时， 表 MTL Tx FIFO 写 ，并 Tx FIFO。( )

MTLTFRCS 表 Tx FIFO ：( )

- 2’b00： 。
- 2’b01：READ (MAC)。
- 2’b10： MAC TxStatus。
- 2’b11：写 TxStatus Tx FIFO。

MACTP 时，表 MAC ( )， 不 。( )

MACTFCS 表 MAC ：( )

- 2’b00： 。
- 2’b01： 一 IFG 避 期 。
- 2’b10： 并 ( )。
- 2’b11： 。

MACTPES 时，表 MAC MII ，不 。( )

MTLRFFLS 表 Rx FIFO 别 ：( )

- 2’b00：Rx FIFO 。
- 2’b01：Rx FIFO 闭 。
- 2’b10：Rx FIFO 。
- 2’b11：Rx FIFO 。

寄存器描述下一页继续。

PMT_RWU

31																															0	
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	Reset

WKUPPKTFILTER（ /写）

- RWKPTR 0 时，表 0 屏蔽。bit[31] 必 0，bit[30:0] 屏蔽。1，0 offset+j (j=0–30) 屏蔽，不屏蔽。
- RWKPTR 1 时，表 1 屏蔽。bit[31] 必 0，bit[30:0] 屏蔽。1，1 offset+j (j=0–30) 屏蔽，不屏蔽。
- RWKPTR 2 时，表 2 屏蔽。bit[31] 必 0，bit[30:0] 屏蔽。1，2 offset+j (j=0–30) 屏蔽，不屏蔽。
- RWKPTR 3 时，表 3 屏蔽。bit[31] 必 0，bit[30:0] 屏蔽。1，3 offset+j (j=0–30) 屏蔽，不屏蔽。

Register 10.22. PMT_CSR_REG (0x102C)

RWKFILTRST	(reserved)	RWKPTR	(reserved)	GLBLUCAST	(reserved)	RWKPRCVD	MGKPRCVD	(reserved)	RWKPKTEN	MGKPKTEN	PWRDWN	
31	30 29	28 24	23 10	9	8 7	6	5	4 3	2	1	0	
0	00	0 0 0 0 0	0 0 0 0 0 0 0 0 0 0 0 0 0 0	0	00	0	0	00	0	0	0	Reset

RWKFILTRST 1 时，把 RWKPTR 0。(/写设 /)

RWKPTR 7，个 表 PMT_RWUFFR 不 ， 从PMT_RWUFFR 。( )

GLBLUCAST 1 单播包 。(/写)

RWKPRCVD 1 表 一 。(/ 设 /)

MGKPRCVD 1 表 magic 一 。(/ 设 /)

RWKPKTEN 1 表 。(/写)

MGKPKTEN 1 表 magic 。(/写)

PWRDWN 1 表 EMAC , 期 magic 。 MGKPKTEN, GLBLUCAST RWKPKTEN 1 时才 。(/写设 /)

Register 10.23. EMACLPI_CSR_REG (0x1030)

(reserved)	LPITXA	(reserved)	PLS	LPIEN	(reserved)	RLPIST	TLPIST	(reserved)	RLPIEX	RLPIEN	TLPIEX	TLPIEN	
31 20	19	18	17	16	15 10	9	8	7 4	3	2	1	0	
0 0 0 0 0 0 0 0 0 0 0	0	0	0	0	0 0 0 0 0 0	0	0	0	0	0	0	0	Reset

LPITXA EMAC LPI , LPITXA LPIEN 1, 并 常 部 , EMAC LPI , 层 时 LPI 。 0 时, LPIEN EMAC LPI 。(/写)

PLS 表 PHY , 1 表 , 0 表 败。(/写)

LPIEN 1 表 EMAC LPI , 0 表 EMAC LPI 常 。 LPITXA 1 并 EMAC 而 LPI 时, 被 0。(/写/)

RLPIST 1 表 EMAC LPI 。(/写)

TLPIST 1 表 EMAC LPI 。(/写)

RLPIEX 1 表 EMAC LPI , LPI , 常 , 被 时 0。(/ 设 /)

RLPIEN 1 表 EMAC LPI 并 LPI , 被 时 0。(/ 设 /)

TLPIEX 1 表 EMAC LPIEN 被 0, 并 LPI_TW_TIMER 期 LPI , 被 时 0。(/ 设 /)

TLPIEN 1 表 EMAC LPIEN 1 LPI , 被 时 0。(/ 设 /)

Register 10.24. EMACLPITIMERSCONTROL_REG (0x1034)

(reserved)	LPI_LS_TIMER	LPI_TW_TIMER	
31 26	25 16	15 0	
0 0 0 0 0 0	0 x 3 E 8	0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0	Reset

LPI_LS_TIMER 表 LPI 层 层 时间 ( )。 IEEE 标 认 1 。(/写)

LPI_TW_TIMER 表 EMAC 常 LPI 时间 ( )。 个 时 期 , TLPIEX 1。(/写)

Register 10.25. EMACINTS_REG (0x1038)

(reserved)	LPIINTS	TINTS	(reserved)	PMTINTS	(reserved)	
31 … 11	10	9	8 … 5	3	2 … 0	
0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0	0	0	0 0 0 0	0	0 0 0	Reset

LPIINTS 启 时， 设 MAC LPI 。（LPI ） Bit[0] 时 。（ ）

TINTS 一个 时， 1：（ / / 设 /写 ）

- 时间 超 标时间 中 。
- 中 。 被 1。 时间 Bit[0] 时， 被 。 认时间 ， 1 时， 表 时间 超 标时间 。 ， 。

PMTINTS 一个 包 时， 1（见PMT Bit[5] Bit[6]）。 PMT 操 ，Bit[6:5] 被 时， 被 。 期间 可 PMT 时， 才 。（ ）

Register 10.26. EMACINTMASK_REG (0x103C)

(reserved)	LPIINTMASK	TINTMASK	(reserved)	PMTINTMASK	(reserved)	
31 … 11	10	9	8 … 4	3	2 … 0	
0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0	0	0	0 0 0 0 0	0	0 0 0	Reset

LPIINTMASK 1 时， 中 中 LPI 中 设 ， 中 。（ /写）

TINTMASK 1 时， 中 中 时间 中 设 ， 中 1。 启 IEEE1588 时间 时 。 ， 保 。（ /写）

PMTINTMASK 1 时， 中 中 PMT 中 设 ， 中 。（ /写）

Register 10.27. EMACADDR0HIGH_REG (0x1040)

ADDRESS_ENABLE0	(reserved)	MAC_ADDRESS0_HI	
31	30 … 16	15 … 0	
0	0 0 0 0 0 0 0 0 0 0 0 0 0 0 0	0x0FFFF	Reset

ADDRESS_ENABLE0 始 1。( )

MAC_ADDRESS0_HI 包 一个 6 MAC 16 [47:32]。MAC ，并 MAC 插 ( ) 中。(/写)

Register 10.28. EMACADDR0LOW_REG (0x1044)

31 … 0	
0x0FFFFFFFF	Reset

EMACADDR0LOW_REG 包 一个 6 MAC 32 。MAC 被 ，并 MAC 插 ( ) 中。(/写)

Register 10.29. EMACADDR1HIGH_REG (0x1048)

ADDRESS_ENABLE1	SOURCE_ADDRESS	MASK_BYTE_CONTROL	(reserved)	MAC_ADDRESS1_HI	
31	30	29 24	23 16	15 0	
0	0	0x00	0 0 0 0 0 0 0 0	0x0FFFF	Reset

ADDRESS_ENABLE1 1 时， 个 MAC 。 时， 忽 。(/写)

SOURCE_ADDRESS 1 时， EMACADDR1 [47:0] SA 比 。 时， EMACADDR1 [47:0] DA 比 。(/写)

MASK_BYTE_CONTROL 屏蔽 , 比 个 MAC 。 时， MAC 不 DA SA EMACADDR1 比 。 一 :

- Bit[9]: EMACADDR1 [15:8]。
- Bit[28]: EMACADDR1 [7:0]。
- Bit[27]: EMACADDR1 [31:24]。
- Bit[24]: EMACADDR1 [7:0]。

可 屏蔽 一个 个 一 ( )。(/写)

MAC_ADDRESS1_HI 包 个 6 MAC 16 [47:32]。(/写)

Register 10.30. EMACADDR1LOW_REG (0x104C)

31 0	
0x0FFFFFFFF	Reset

EMACADDR1LOW_REG 包 个 6 MAC 32 。 始 , 被 , 不 。(/写)

Register 10.31. EMACADDR2HIGH_REG (0x1050)

ADDRESS_ENABLE2	SOURCE_ADDRESS2	MASK_BYTE_CONTROL2	(reserved)	MAC_ADDRESS2_HI	
31	30	29 24	23 16	15 0	
0	0	0x00	0 0 0 0 0 0 0 0	0x0FFFF	Reset

ADDRESS_ENABLE2 1 时， 三个 MAC 。 时， 。(/写)

SOURCE_ADDRESS2 1 时，EMACADDR2 [47:0] SA 比 。 时， EMACADDR2 [47:0] DA 比 。(/写)

MASK_BYTE_CONTROL2 屏蔽 ， 比 个 MAC 。 时，MAC 不 DA SA EMACADDR2 比 。 一 :

- Bit[9]：EMACADDR2 [15:8]。
- Bit[28]：EMACADDR2 [7:0]。
- Bit[27]：EMACADDR2 [31:24]。
- Bit[24]：EMACADDR2 [7:0]。

可 屏蔽 一个 个 一 ( )。(/写)

MAC_ADDRESS2_HI 包 三个 6 MAC 16 [47:32]。(/写)

Register 10.32. EMACADDR2LOW_REG (0x1054)

31 0	
0x0FFFFFFFF	Reset

EMACADDR2LOW_REG 包 三个 6 MAC 32 。 始化 ， 被 ， 不 。(/写)

Register 10.33. EMACADDR3HIGH_REG (0x1058)

ADDRESS_ENABLE3	SOURCE_ADDRESS3	MASK_BYTE_CONTROL3	(reserved)	MAC_ADDRESS3_HI	
31	30	29 24	23 16	15 0	
0	0	0x00	0 0 0 0 0 0 0 0	0x0FFFF	Reset

ADDRESS_ENABLE3 1 时， 个 MAC 。 时， 。(/写)

SOURCE_ADDRESS3 1 时，EMACADDR3 [47:0] SA 比 。 时， EMACADDR3 [47:0] DA 比 。(/写)

MASK_BYTE_CONTROL3 屏蔽 ， 比 个 MAC 。 时，MAC 不 DA SA EMACADDR3 比 。 一 :

- Bit[9]：EMACADDR3 [15:8]。
- Bit[28]：EMACADDR3 [7:0]。
- Bit[27]：EMACADDR3 [31:24]。
- Bit[24]：EMACADDR3 [7:0]。

可 屏蔽 一个 个 一 ( )。(/写)

MAC_ADDRESS3_HI 包 个 6 MAC 16 [47:32]。(/写)

Register 10.34. EMACADDR3LOW_REG (0x1060)

31 0	
0x0FFFFFFFF	Reset

EMACADDR3LOW_REG 包 个 6 MAC 32 。 始化 ， 被 ， 不 。(/写)

Register 10.35. EMACADDR4HIGH_REG (0x1064)

ADDRESS_ENABLE4	SOURCE_ADDRESS4	MASK_BYTE_CONTROL4	(reserved)	MAC_ADDRESS4_HI	
31	30	29 24	23 16	15 0	
0	0	0x00	0 0 0 0 0 0 0 0	0x0FFFF	Reset

ADDRESS_ENABLE4 1 时， 个 MAC 。 时， 。(/写)

SOURCE_ADDRESS4 1 时，EMACADDR4 [47:0] SA 比 。 时， EMACADDR4 [47:0] DA 比 。(/写)

MASK_BYTE_CONTROL4 屏蔽 ， 比 个 MAC 。 时，MAC 不 DA SA EMACADDR4 比 。 一 :

- Bit[9]：EMACADDR4 [15:8]。
- Bit[28]：EMACADDR4 [7:0]。
- Bit[27]：EMACADDR4 [31:24]。
- Bit[24]：EMACADDR4 [7:0]。

可 屏蔽 一个 个 一 ( )。(/写)

MAC_ADDRESS4_HI 包 个 6 MAC 16 [47:32]。(/写)

Register 10.36. EMACADDR4LOW_REG (0x1068)

31 0	
0x0FFFFFFFF	Reset

EMACADDR4LOW_REG 包 个 6 MAC 32 。 始化 ， 被 ， 不 。(/写)

Register 10.37. EMACADDR5HIGH_REG (0x106C)

ADDRESS_ENABLE5	SOURCE_ADDRESS5	MASK_BYTE_CONTROL5	(reserved)	MAC_ADDRESS5_HI	
31	30	29 24	23 16	15 0	
0	0	0x00	0 0 0 0 0 0 0 0	0x0FFFF	Reset

ADDRESS_ENABLE5 1 时， 个 MAC 。 时， 。(/写)

SOURCE_ADDRESS5 1 时，EMACADDR5 [47:0] SA 比 。 时， EMACADDR5 [47:0] DA 比 。(/写)

MASK_BYTE_CONTROL5 屏蔽 ， 比 个 MAC 。 时，MAC 不 DA SA EMACADDR5 比 。 一 :

- Bit[9]：EMACADDR5 [15:8]。
- Bit[28]：EMACADDR5 [7:0]。
- Bit[27]：EMACADDR5 [31:24]。
- Bit[24]：EMACADDR5 [7:0]。

可 屏蔽 一个 个 一 ( )。(/写)

MAC_ADDRESS5_HI 包 个 6 MAC 16 [47:32]。(/写)

Register 10.38. EMACADDR5LOW_REG (0x1070)

31 0	
0x0FFFFFFFF	Reset

EMACADDR5LOW_REG 包 个 6 MAC 32 。 始化 ， 被 ， 不 。(/写)

Register 10.39. EMACADDR6HIGH_REG (0x1074)

ADDRESS_ENABLE6	SOURCE_ADDRESS6	MASK_BYTE_CONTROL6	(reserved)	MAC_ADDRESS6_HI	
31	30	29 24	23 16	15 0	
0	0	0x00	0 0 0 0 0 0 0 0	0x0FFFF	Reset

ADDRESS_ENABLE6 1 时， 个 MAC 。 时， 。(/写)

SOURCE_ADDRESS6 1 时，EMACADDR6 [47:0] SA 比 。 时， EMACADDR6 [47:0] DA 比 。(/写)

MASK_BYTE_CONTROL6 屏蔽 ， 比 个 MAC 。 时，MAC 不 DA SA EMACADDR6 比 。 一 :

- Bit[9]：EMACADDR6 [15:8]。
- Bit[28]：EMACADDR6 [7:0]。
- Bit[27]：EMACADDR6 [31:24]。
- Bit[24]：EMACADDR6 [7:0]。

可 屏蔽 一个 个 一 ( )。(/写)

MAC_ADDRESS6_HI 包 个 6 MAC 16 [47:32]。(/写)

Register 10.40. EMACADDR6LOW_REG (0x1078)

31 0	
0x0FFFFFFFF	Reset

EMACADDR6LOW_REG 包 个 6 MAC 32 。 始化 ， 被 ， 不 。(/写)

Register 10.41. EMACADDR7HIGH_REG (0x107C)

ADDRESS_ENABLE7	SOURCE_ADDRESS7	MASK_BYTE_CONTROL7	(reserved)	MAC_ADDRESS7_HI	
31	30	29 24	23 16	15 0	
0	0	0x00	0 0 0 0 0 0 0 0	0x0FFFF	Reset

ADDRESS_ENABLE7 1 时， 八个 MAC 。
时， 。(/写)

SOURCE_ADDRESS7 1 时，EMACADDR7 [47:0] SA 比 。
时， EMACADDR7 [47:0] DA 比 。(/写)

MASK_BYTE_CONTROL7 屏蔽 , 比 个 MAC 。 时，MAC
不 DA SA EMACADDR7 比 。 一
:

- Bit[9]：EMACADDR7 [15:8]。
- Bit[28]：EMACADDR7 [7:0]。
- Bit[27]：EMACADDR7 [31:24]。
- Bit[24]：EMACADDR7 [7:0]。

可 屏蔽 一个 个 一 ( )。(/写)

MAC_ADDRESS7_HI 包 八个 6 MAC 16 [47:32]。(/写)

Register 10.42. EMACADDR7LOW_REG (0x1070)

31 0	
0x0FFFFFFFF	Reset

EMACADDR7LOW_REG 包 八个 6 MAC 32 。 始化 ,
被 , 不 。(/写)

Register 10.43. EMACCSTATUS_REG (0x10D8)

(reserved)	SMIDRXS	(reserved)	JABBER_TIMEOUT	(reserved)	LINK_MODE
31 … 17	16	15 … 5	4	3 … 1	0
0 0 0 0 0 0 0 0 0 0 0 0 0 0 0	0	0 0 0 0 0 0 0 0 0 0 0	0	0 0 0	0

Reset

JABBER_TIMEOUT 表 中 超时 (1'b1)。( )

LINK_MODE 表 操 ( ):

- 1'b0: 半 。
- 1'b1: 。

Register 10.44. EMACWDOGTO_REG (0x10DC)

(reserved)	PWDOGEN	(reserved)	WDOGTO
31 … 17	16	15 … 14	13 … 0
0 0 0 0 0 0 0 0 0 0 0 0 0 0 0	0	0 0	0x0000

Reset

PWDOGEN 1 EMACCONFIG_REG Bit[23] (WD) 时, WTO (Bit [13:0]) 超时。 时, EMACCONFIG_REG Bit[23] (WD) Bit[20] (JE) 设 。(/写)

WDOGTO Bit[16] (PWE) 1 MAC Bit[23] (WD) 时, 超时。 长 超 个 , 个 被 并被声 。(/写)

Register 10.45. EMAC_EX_CLKOUT_CONF_REG (0x0000)

(reserved)	EMAC_CLK_OUT_H_DIV_NUM	EMAC_CLK_OUT_DIV_NUM
31 … 8	7 … 4	3 … 0
0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0	0x02	0x04

Reset

EMAC_CLK_OUT_H_DIV_NUM RMII PHY 时, RMII CLK 部 APLL CLK, 半 。(/写)

EMAC_CLK_OUT_DIV_NUM RMII PHY 时, RMII CLK 部 APLL CLK, 。(/写)

Register 10.46. EMAC_EX_OSCCLK_CONF_REG (0x0004)

(reserved)	EMAC_OSC_CLK_SEL	EMAC_OSC_H_DIV_NUM_100M	EMAC_OSC_DIV_NUM_100M	EMAC_OSC_H_DIV_NUM_10M	EMAC_OSC_DIV_NUM_10M	
31 25	24	23 18	17 12	11 6	5 0	
0 0 0 0 0 0 0	0	0	1	9	19	Reset

EMAC_OSC_CLK_SEL RMII PHY 时，部 PHY 时 RMII CLK。1 时，部 CLK；0 时，部 APLL CLK。(/写)

EMAC_OSC_H_DIV_NUM_100M RMII/MII 半 ，EMAC_EX_CLKOUT_CONF 100 MHz 时。(/写)

EMAC_OSC_DIV_NUM_100M RMII/MII ，EMAC_EX_CLKOUT_CONF 100 MHz 时。(/写)

EMAC_OSC_H_DIV_NUM_10M RMII/MII 半 ，EMAC_EX_CLKOUT_CONF 10 MHz 时。(/写)

EMAC_OSC_DIV_NUM_10M RMII/MII ，EMAC_EX_CLKOUT_CONF 10 MHz 时。(/写)

Register 10.47. EMAC_EX_CLK_CTRL_REG (0x0008)

(reserved)	EMAC_CLK_EN	EMAC_MII_CLK_RX_EN	EMAC_MII_CLK_TX_EN	(reserved)	EMAC_INT_OSC_EN	EMAC_EXT_OSC_EN	
31 6	5	4	3	2	1	0	
0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0	0	0	0	0	0	0	Reset

EMAC_MII_CLK_RX_EN Ethernet RX CLK 。(/写)

EMAC_MII_CLK_TX_EN Ethernet TX CLK 。(/写)

EMAC_INT_OSC_EN RMII PHY 部 APLL CLK。(/写)

EMAC_EXT_OSC_EN RMII PHY 部 XTAL CLK。(/写)

Register 10.48. EMAC_EX_PHYINF_CONF_REG (0x000c)

(reserved)	EMAC_PHY_INTF_SEL	(reserved)	
31 … 16	15 … 13	12 … 0	
0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0	0 0 0	0 0 0 0 0 0 0 0 0 0 0 0 0	Reset

EMAC_PHY_INTF_SEL PHY 。0x0 表 PHY MII，0x4 表 PHY RMII。(/写)

Register 10.49. EMAC_PD_SEL_REG (0x0010)

(reserved)	EMAC_RAM_PD_EN	
31 … 2	1 … 0	
0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0	0	Reset

EMAC_RAM_PD_EN RAM ，Bit 0 TX SRAM，Bit 1 RX SRAM。
1 表 。(/写)

11.3.2 I2C 架构

图 11-1. I2C Master 基本架构

图 11-2. I2C Slave 基本架构

I2C 可 Master Slave ，I2C_MS_MODE 。 11-1 I2C Master 本 ， 11-2 I2C Slave 本 。 中可 ，I2C 部 个单 ：

- RAM： 32 x 8 bit， CPU 上，寄 (REG_I2C_BASE+0x100)，I2C 一个 一个 word （ ，寄 +0x100， +0x104， 三 +0x108， ）。 I2C_NONFIFO_EN 。
- 16 个 (cmd0 ~ cmd15) 一个 CMD_Controller： I2C Master 。I2C 一个 。
- SCL_FSM： SCL 时 ，I2C_SCL_HIGH_PERIOD_REG I2C_SCL_LOW_PERIOD_REG SCL 比。
- SDA_FSM： SDA 。
- DATA_Shifter： 位 比 比 位 。I2C_RX_LSB_FIRST

I2C_TX_LSB_FIRST 。

- SCL_Filter SDA_Filter: I2C_Slave 声。 I2C_SCL_FILTER_EN I2C_SDA_FILTER_EN 可 启 闭 波 。 波 可 脉 I2C_SCL_FILTER_THRES I2C_SDA_FILTER_THRES 。

11.3.3 I2C 总线时序

图 11-3. I2C 时序图

11-3 I2C 时 。 I2C Master , SCL 。 I2C Slave , SCL 。 I2C_SDA_HOLD_REG I2C_SDA_SAMPLE_REG Slave 。 I2C Master I2C_SDA_HOLD _TIME I2C_SDA_SAMPLE_TIME , 可 I2C Slave 不 。 SCL 时, SCL 期 表 11-1 , SCL f_{scl}。

表 11-1. SCL 频率配置

I2C_SCL_FILTER_EN	I2C_SCL_FILTER_THRES	SCL_Low_Level_Cycles	SCL_High_Level_Cycles
0	Don’t care	I2C_SCL_LOW_PERIOD+1	I2C_SCL_HIGH_PERIOD+7
1	[0,2]		I2C_SCL_HIGH_PERIOD+8
	[3,7]		I2C_SCL_HIGH_PERIOD+6+I2C_SCL_FILTER_THRES

$$f_{scl} = \frac{80\ \text{MHz}}{\text{SCL_Low_Level_Cycles} + \text{SCL_High_Level_Cycles}}$$

I2C , 一 始 START STOP 。 按 一个 , 个 一个 ACK 。 ACK 。 可 不 ACK 。

I2C START 、STOP 、 保 时间、 采 时间 可 11-3 中 。

注意： SCL pad 漏 时, SCL 时间 变长, 个 变时间 SCL 上 上 。 漏 , SCL 。

11.3.4 I2C cmd 结构

I2C Master 中 , 部 11-4 。

图 11-4. I2C 命令寄存器结构

CMD_DONE：可 CMD_DONE 一 毕。

op_code：op_code 编 ，I2C 4 。

- RSTART：op_code 0 时 RSTART ， I2C 中 START RESTART 。
- WRITE：op_code 1 时 WRITE ， 表 Master 。
- READ：op_code 2 时 READ ， 表 Master 。
- STOP：op_code 3 时 STOP ， 中 STOP 。
- END：op_code 4 时 END ， master 。 闭 SCL 时 ， 备 毕， 上 。

一 始 RSTART ， STOP 。

ack_value： 时， 被 ， 表 一个 ACK 。

ack_exp： 设 期 ACK 。

ack_check_en： ACK 测。1： 测 ACK ；0：不 测 ACK 。

byte_num： 写 长 (单)， 255， 1。RSTART、STOP、END 中 byte_num 。

11.3.5 I2C 主机写入从机

便 ， 中 I2C Master Slave ESP32 I2C 设 。

11-5 I2C Master 写 N 个 I2C Slave 。 I2C ， 一个 I2C Slave ， 中 RAM ， 一个 (Slave 7-bit +1-bit 写标)， 中 写标 0 时表 写查 上 。

，

- 中，SDA 被 I2C Slave ，时 I2C Master Slave 个 SCL 时 可 SDA 。

，上 Master Slave 操 可 ESP32 I2C 设 不 ，参 个 I2C 设备 寻。

ESP32 I2C 认 7-bit ，可 10-bit 。中，一个 个 I2C 可 。中，可 I2C_ADDR_10BIT_EN 启 10-bit 。I2C_SLAVE_ADDR I2C Slave 。11-6 I2C Master 写 N 个 10-bit I2C Slave ，10-bit Slave 比 7-bit 一个 ，WRITE byte_num RAM 中 1。

图 11-6. I2C Master 写 10-bit 地址 Slave

不 END ，I2C Master 一 (14*255-2) 个 10-bit I2C Slave。

I2C Slave 。I2C Master 1 个 可 中 。ESP32 I2C Slave 上 操 。

I2C_FIFO_ADDR_CFG_EN 启 I2C Slave 。11-7 ，I2C Slave byte0 ~ byte(N-1) Slave RAM 中 addrM 始 。，I2C Slave 一 32 ，I2C Master 超 32 时，。

图 11-7. I2C Master 写 7-bit 地址 Slave 的 M 地址 RAM

一 ，超 14 个 /写 cmd 时，可 END 。

11-8 I2C Master 三 写Slave。当 I2C Master 一 ，并 Master RAM 中 备 ， I2C_TRANS_START，I2C Master 始 。 END ，I2C Master 闭 SCL 时 ，并 SCL 设备 I2C 。 时 产 I2C_END_DETECT_INT 中 。

图 11-8. I2C Master 分段写 7-bit 地址 Slave

测 I2C_END_DETECT_INT 中 ， 可 cmd RAM 中 ，并 I2C_END_DETECT_INT 中 。 I2C_TRANS_START ，I2C Master 可 。 不 时， 测 I2C Master I2C_END_DETECT_INT 中 可 cmd 三 ， I2C_TRANS_START ，I2C Master 产 STOP ， 而 。

注意：

- 个 间，I2C 上 Master 设备不 时间 。 STOP 才 被 。
- 超 3 时， 一 END cmd 不 被 一 。

11.3.6 I2C 主机读取从机

图 11-9. I2C Master 读 7-bit 地址 Slave

11-9 I2C Master 7-bit I2C Slave N 个 。 I2C Master I2C Slave , cmd1 WRITE。 一个 I2C Slave 写标 , 中，1 表 一个 操 。I2C Slave 始 I2C Master。I2C Master READ 中 ack_value 个 ACK。 11-9 中 READ , I2C Master cmd2 中 N-1 个 ACK， cmd3 中 一个 不 ACK， 时可 。 时，I2C Master RAM 始 , 中 byte0 (Slave +1-bit 写)。

不 END , I2C Master 一 I2C Slave (13*255) 个 , cmd RSTART + 1 个 WRITE + 13 个 READ + 1 个 STOP。

11-10 I2C Master 10-bit I2C Slave 中 。 I2C Slave 中 I2C_SLAVE_ADDR_10BIT_EN I2C Slave RAM 中 备 , I2C Master RAM 中 备 I2C Slave 2 个 10-bit 。 I2C Master I2C_TRANS_START， 可 启 一 。

图 11-10. I2C Master 读 10-bit 地址 Slave

11-11 I2C Master I2C Slave 中 。常 I2C Slave 中 I2C_FIFO_ADDR_CFG_EN 并 RAM 中 备 。 I2C Master 中 备 I2C Slave M 。 I2C Master I2C_TRANS_START，I2C Slave RAM 中 M 始 N 个 I2C Master。

图 11-11. I2C Master 从 7-bit 地址 Slave 的 M 地址读取 N 个数据

11-12 I2C Master END 三 I2C Slave N+M 个 。常 cmd 一 并 Slave RAM 中 备 ， I2C_TRANS_START，I2C 始 。 END 时， I2C Master 可 cmd ，并 I2C_END_DETECT_INT 中 。 I2C_TRANS_START，I2C Master 。 不 时， 测 I2C Master I2C_END_DETECT_INT 中 ， cmd 三 。 I2C_TRANS_START，I2C Master STOP 。

图 11-12. I2C Master 分段读 7-bit 地址 Slave

11.3.7 中断

- I2C_TX_SEND_EMPTY_INT: I2C nonfifo_tx_thres 个 ， 中 。
- I2C_RX_REC_FULL_INT: I2C nonfifo_rx_thres 个 ， 中 。
- I2C_ACK_ERR_INT: I2C Master 时， ACK 中期 ACK 不一 时， 中 ； I2C Slave 时， ACK 1 时 中 。
- I2C_TRANS_START_INT: I2C 一个 START 时， 中 。
- I2C_TIME_OUT_INT: 中， I2C SCL 保 时间超 I2C_TIME_OUT 个时 ， 中 。
- I2C_TRANS_COMPLETE_INT: I2C 测 STOP 时， 中 。
- I2C_MASTER_TRAN_COMP_INT: I2C Master 一个 ， 中 。
- I2C_ARBITRATION_LOST_INT: I2C Master SCL ，SDA 不 时， 中 。
- I2C_END_DETECT_INT: I2C END 时， 中 。

11.4 寄存器列表

		I2C0	I2C1	
配置寄存器				
I2C_SLAVE_ADDR_REG	I2C slave	0x3FF53010	0x3FF67010	/写
I2C_RXFIFO_ST_REG	FIFO	0x3FF53014	0x3FF67014	
I2C_FIFO_CONF_REG	FIFO	0x3FF53018	0x3FF67018	/写
时序寄存器				
I2C_SDA_HOLD_REG	SCL 保 时间	0x3FF53030	0x3FF67030	/写
I2C_SDA_SAMPLE_REG	SCL 上升 采 时间	0x3FF53034	0x3FF67034	/写
I2C_SCL_LOW_PERIOD_REG	SCL 时	0x3FF53000	0x3FF67000	/写
I2C_SCL_HIGH_PERIOD_REG	SCL 时	0x3FF53038	0x3FF67038	/写
I2C_SCL_START_HOLD_REG	SDA SCL 间 ，启	0x3FF53040	0x3FF67040	/写
I2C_SCL_RSTART_SETUP_REG	SCL 上升 SDA 间	0x3FF53044	0x3FF67044	/写
I2C_SCL_STOP_HOLD_REG	SCL 时 边 ，	0x3FF53048	0x3FF67048	/写
I2C_SCL_STOP_SETUP_REG	SDA SCL 上升 间 ，	0x3FF5304C	0x3FF6704C	/写
滤波寄存器				
I2C_SCL_FILTER_CFG_REG	SCL 波	0x3FF53050	0x3FF67050	/写
I2C_SDA_FILTER_CFG_REG	SDA 波	0x3FF53054	0x3FF67054	/写
中断寄存器				
I2C_INT_RAW_REG	始中	0x3FF53020	0x3FF67020	
I2C_INT_CLR_REG	中	0x3FF53024	0x3FF67024	写
I2C_INT_ENA_REG	中	0x3FF53028	0x3FF67028	/写
I2C_INT_STATUS_REG	中	0x3FF5302C	0x3FF6702C	
命令寄存器				
I2C_COMD0_REG	I2C 0	0x3FF53058	0x3FF67058	/写
I2C_COMD1_REG	I2C 1	0x3FF5305C	0x3FF6705C	/写
I2C_COMD2_REG	I2C 2	0x3FF53060	0x3FF67060	/写
I2C_COMD3_REG	I2C 3	0x3FF53064	0x3FF67064	/写
I2C_COMD4_REG	I2C 4	0x3FF53068	0x3FF67068	/写
I2C_COMD5_REG	I2C 5	0x3FF5306C	0x3FF6706C	/写
I2C_COMD6_REG	I2C 6	0x3FF53070	0x3FF67070	/写
I2C_COMD7_REG	I2C 7	0x3FF53074	0x3FF67074	/写
I2C_COMD8_REG	I2C 8	0x3FF53078	0x3FF67078	/写
I2C_COMD9_REG	I2C 9	0x3FF5307C	0x3FF6707C	/写
I2C_COMD10_REG	I2C 10	0x3FF53080	0x3FF67080	/写
I2C_COMD11_REG	I2C 11	0x3FF53084	0x3FF67084	/写
I2C_COMD12_REG	I2C 12	0x3FF53088	0x3FF67088	/写
I2C_COMD13_REG	I2C 13	0x3FF5308C	0x3FF6708C	/写
I2C_COMD14_REG	I2C 14	0x3FF53090	0x3FF67090	/写
I2C_COMD15_REG	I2C 15	0x3FF53094	0x3FF67094	/写

11.5 寄存器

Register 11.1. I2C_SCL_LOW_PERIOD_REG (0x0000)

(reserved)	I2C_SCL_LOW_PERIOD	
31 … 14	13 … 0	
0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0	0 0 0 0 0 0 0 0 0 0 0 0 0 0	Reset

I2C_SCL_LOW_PERIOD I2C Master 时，SCL 时 APB 时 期 。(/写)

Register 11.2. I2C_CTR_REG (0x0004)

(reserved)	I2C_RX_LSB_FIRST	I2C_TX_LSB_FIRST	I2C_TRANS_START	I2C_MS_MODE	(reserved)	I2C_SAMPLE_SCL_LEVEL	I2C_SCL_FORCE_OUT	I2C_SDA_FORCE_OUT	
31 … 8	7	6	5	4	3	2	1	0	
0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0	0	0	0	0	0	0	1	1	Reset

I2C_RX_LSB_FIRST 。(/写)

1: 始 ;

0: 始 。

I2C_TX_LSB_FIRST 。(/写)

1: 始 ;

0: 始 。

I2C_TRANS_START , 始 txfifo 中 。(/写)

I2C_MS_MODE , I2C Master。 , I2C Slave。(/写)

I2C_SAMPLE_SCL_LEVEL 1: SCL 时，采 SDA ; 0: SCL 时，采 SDA 。(/写)

I2C_SCL_FORCE_OUT 0: ; 1: 漂 漂 。(/写)

I2C_SDA_FORCE_OUT 0: ; 1: 漂 漂 。(/写)

Register 11.3. I2C_SR_REG (0x0008)

(reserved)	I2C_SCL_STATE_LAST	(reserved)	I2C_SCL_MAIN_STATE_LAST	I2C_TXFIFO_CNT	(reserved)	I2C_RXFIFO_CNT	(reserved)	I2C_BYTE_TRANS	I2C_SLAVE_ADDRESSED	I2C_BUS_BUSY	I2C_ARB_LOST	I2C_TIME_OUT	I2C_SLAVE_RW	I2C_ACK_REC	
31	30 28	27	26 24	23 18	17 14	13 8	7	6	5	4	3	2	1	0	
0	0 0 0	0	0 0 0	0 0 0 0 0 0	0 0 0 0	0 0 0 0 0 0	0	0	0	0	0	0	0	0	Reset

I2C_SCL_STATE_LAST 产 SCL 。( )
0: ; 1: 始; 2: ; 3: ; 4: 上升 ; 5: ; 6:

I2C_SCL_MAIN_STATE_LAST I2C 。( )
0: ; 1: ; 2: ACK ; 3: ; 4: ; 5: ACK; 6: ACK

I2C_TXFIFO_CNT RAM 中 。( )

I2C_RXFIFO_CNT 表 。( )

I2C_BYTE_TRANS 一个 时, 变 1。( )

I2C_SLAVE_ADDRESSED I2C Slave, 时,
。( )

I2C_BUS_BUSY 1: I2C ; 0: I2C 。( )

I2C_ARB_LOST I2C 不 SCL , 变 1 。( )

I2C_TIME_OUT I2C 一个 bit 时间超时, 变 1。( )

I2C_SLAVE_RW 中, 1: 中 ; 0: 写 。( )

I2C_ACK_REC ACK 。( )

Register 11.4. I2C_TO_REG (0x000c)

(reserved)	I2C_TIME_OUT_REG	
31 20	19 0	
0 0 0 0 0 0 0 0 0 0 0 0	0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0	Reset

I2C_TIME_OUT_REG 一个 bit 超 时, 产 超时中 。(/写)

Register 11.5. I2C_SLAVE_ADDR_REG (0x0010)

I2C_SLAVE_ADDR_10BIT_EN	(reserved)	I2C_SLAVE_ADDR	
31	30 15	14 0	
0	0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0	0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0	Reset

I2C_SLAVE_ADDR_10BIT_EN I2C Slave 时， 启 10-bit 。(/写)

I2C_SLAVE_ADDR I2C Slave 时， Slave 。(/写)

Register 11.6. I2C_RXFIFO_ST_REG (0x0014)

(reserved)	I2C_RXFIFO_TXFIFO_END_ADDR	I2C_RXFIFO_TXFIFO_START_ADDR	I2C_RXFIFO_END_ADDR	I2C_RXFIFO_START_ADDR	
31 20	19 15	14 10	9 5	4 0	
0 0 0 0 0 0 0 0 0 0 0 0	0 0 0 0 0	0 0 0 0 0	0 0 0 0 0	0 0 0 0 0	Reset

I2C_TXFIFO_END_ADDR nonfifo_tx_thres , 。 I2C_TX_SEND_EMPTY_INT 中 I2C_TRANS_COMPLETE_INT 中 产 时 。( )

I2C_TXFIFO_START_ADDR nonfifo_tx_thres , 。( )

I2C_RXFIFO_END_ADDR nonfifo_rx_thres , 。 I2C_RX_REC_FULL_INT 中 I2C_TRANS_COMPLETE_INT 中 产 时 。( )

I2C_RXFIFO_START_ADDR nonfifo_rx_thres , 。( )

Register 11.7. I2C_FIFO_CONF_REG (0x0018)

(reserved)	I2C_NONFIFO_TX_THRES	I2C_NONFIFO_RX_THRES	(reserved)	I2C_FIFO_ADDR_CFG_EN	I2C_NONFIFO_EN	
31 26	25 20	19 14	13 12	11	10	
0 0 0 0 0 0	0x15	0x15	0 0	0	0	Reset

I2C_NONFIFO_TX_THRES I2C nonfifo_tx_thres , 产 tx_send_empty_int_raw 中 ，并 。(/写)

I2C_NONFIFO_RX_THRES I2C nonfifo_tx_thres , 产 rx_send_full_int_raw 中 ，并 。(/写)

I2C_FIFO_ADDR_CFG_EN 1 时，可 按 Slave。(/写)

I2C_NONFIFO_EN APB nonfifo 。(/写)

Register 11.8. I2C_INT_RAW_REG (0x0020)

(reserved)	I2C_TX_SEND_EMPTY_INT_RAW	I2C_RX_REC_FULL_INT_RAW	I2C_ACK_ERR_INT_RAW	I2C_TRANS_START_INT_RAW	I2C_TIME_OUT_INT_RAW	I2C_TRANS_COMPLETE_INT_RAW	I2C_MASTER_TRAN_COMP_INT_RAW	I2C_ARBITRATION_LOST_INT_RAW	I2C_END_DETECT_INT_RAW	
31 13	12	11	10	9	8	7	6	5	3	
0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0	0	0	0	0	0	0	0	0	0	Reset

I2C_TX_SEND_EMPTY_INT_RAW I2C_TX_SEND_EMPTY_INT 中 始中 。( )

I2C_RX_REC_FULL_INT_RAW I2C_RX_REC_FULL_INT 中 始中 。( )

I2C_ACK_ERR_INT_RAW I2C_ACK_ERR_INT 中 始中 。( )

I2C_TRANS_START_INT_RAW I2C_TRANS_START_INT 中 始中 。( )

I2C_TIME_OUT_INT_RAW I2C_TIME_OUT_INT 中 始中 。( )

I2C_TRANS_COMPLETE_INT_RAW I2C_TRANS_COMPLETE_INT 中 始中 。( )

I2C_MASTER_TRAN_COMP_INT_RAW I2C_MASTER_TRAN_COMP_INT 中 始中 。( )

I2C_ARBITRATION_LOST_INT_RAW I2C_ARBITRATION_LOST_INT 中 始中 。( )

I2C_END_DETECT_INT_RAW I2C_END_DETECT_INT 中 始中 。( )

Register 11.9. I2C_INT_CLR_REG (0x0024)

(reserved)	I2C_TX_SEND_EMPTY_INT_CLR	I2C_RX_REC_FULL_INT_CLR	I2C_ACK_ERR_INT_CLR	I2C_TRANS_START_INT_CLR	I2C_TIME_OUT_INT_CLR	I2C_TRANS_COMPLETE_INT_CLR	I2C_MASTER_TRAN_COMP_INT_CLR	I2C_ARBITRATION_LOST_INT_CLR	I2C_END_DETECT_INT_CLR	
31 … 13	12	11	10	9	8	7	6	5	3	
0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0	0	0	0	0	0	0	0	0	0	Reset

I2C_TX_SEND_EMPTY_INT_CLR I2C_TX_SEND_EMPTY_INT 中 。(写)

I2C_RX_REC_FULL_INT_CLR I2C_RX_REC_FULL_INT 中 。(写)

I2C_ACK_ERR_INT_CLR I2C_ACK_ERR_INT 中 。(写)

I2C_TRANS_START_INT_CLR I2C_TRANS_START_INT 中 。(写)

I2C_TIME_OUT_INT_CLR I2C_TIME_OUT_INT 中 。(写)

I2C_TRANS_COMPLETE_INT_CLR I2C_TRANS_COMPLETE_INT 中 。(写)

I2C_MASTER_TRAN_COMP_INT_CLR I2C_MASTER_TRAN_COMP_INT 中 。(写)

I2C_ARBITRATION_LOST_INT_CLR I2C_ARBITRATION_LOST_INT 中 。(写) I2C_SLAVE_TRAN_COMP_INT 中 。(写)

I2C_END_DETECT_INT_CLR I2C_END_DETECT_INT 中 。(写)

Register 11.10. I2C_INT_ENA_REG (0x0028)

(reserved)	I2C_TX_SEND_EMPTY_INT_ENA	I2C_RX_REC_FULL_INT_ENA	I2C_ACK_ERR_INT_ENA	I2C_TRANS_START_INT_ENA	I2C_TIME_OUT_INT_ENA	I2C_TRANS_COMPLETE_INT_ENA	I2C_MASTER_TRAN_COMP_INT_ENA	I2C_ARBITRATION_LOST_INT_ENA	I2C_END_DETECT_INT_ENA	
31 … 13	12	11	10	9	8	7	6	5	3	
0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0	0	0	0	0	0	0	0	0	0	Reset

I2C_TX_SEND_EMPTY_INT_ENA 中 , I2C_TX_SEND_EMPTY_INT 中 。(/写)

I2C_RX_REC_FULL_INT_ENA 中 , I2C_RX_REC_FULL_INT 中 。(/写)

I2C_ACK_ERR_INT_ENA 中 , I2C_ACK_ERR_INT 中 。(/写)

I2C_TRANS_START_INT_ENA 中 , I2C_TRANS_START_INT 中 。(/写)

I2C_TIME_OUT_INT_ENA 中 , I2C_TIME_OUT_INT 中 。(/写)

I2C_TRANS_COMPLETE_INT_ENA 中 , I2C_TRANS_COMPLETE_INT 中 。(/写)

I2C_MASTER_TRAN_COMP_INT_ENA 中 , I2C_MASTER_TRAN_COMP_INT 中 。(/写)

I2C_ARBITRATION_LOST_INT_ENA 中 , I2C_ARBITRATION_LOST_INT 中 。(/写)

I2C_END_DETECT_INT_ENA 中 , I2C_END_DETECT_INT 中 。(/写)

Register 11.11. I2C_INT_STATUS_REG (0x002c)

(reserved)	I2C_TX_SEND_EMPTY_INT_ST	I2C_RX_REC_FULL_INT_ST	I2C_ACK_ERR_INT_ST	I2C_TRANS_START_INT_ST	I2C_TIME_OUT_INT_ST	I2C_TRANS_COMPLETE_INT_ST	I2C_MASTER_TRAN_COMP_INT_ST	I2C_ARBITRATION_LOST_INT_ST	I2C_END_DETECT_INT_ST	
31 … 13	12	11	10	9	8	7	6	5	3	
0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0	0	0	0	0	0	0	0	0	0	Reset

I2C_TX_SEND_EMPTY_INT_ST 蔽中 , I2C_TX_SEND_EMPTY_INT 中 。( )

I2C_RX_REC_FULL_INT_ST 蔽中 , I2C_RX_REC_FULL_INT 中 。( )

I2C_ACK_ERR_INT_ST 蔽中 , I2C_ACK_ERR_INT 中 。( )

I2C_TRANS_START_INT_ST 蔽中 , I2C_TRANS_START_INT 中 。( )

I2C_TIME_OUT_INT_ST 蔽中 , I2C_TIME_OUT_INT 中 。( )

I2C_TRANS_COMPLETE_INT_ST 蔽中 , I2C_TRANS_COMPLETE_INT 中 。( )

I2C_MASTER_TRAN_COMP_INT_ST 蔽中 , I2C_MASTER_TRAN_COMP_INT 中 。( )

I2C_ARBITRATION_LOST_INT_ST 蔽中 , I2C_ARBITRATION_LOST_INT 中 。( )

I2C_END_DETECT_INT_ST 蔽中 , I2C_END_DETECT_INT 中 。( )

Register 11.12. I2C_SDA_HOLD_REG (0x0030)

(reserved)	I2C_SDA_HOLD_TIME	
31 … 10	9 … 0	
0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0	0 0 0 0 0 0 0 0 0 0	Reset

I2C_SDA_HOLD_TIME SCL SDA 保 不变 APB 时 期 。(/写)

Register 11.13. I2C_SDA_SAMPLE_REG (0x0034)

(reserved)	I2C_SDA_SAMPLE_TIME	
31 … 10	9 … 0	
0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0	0 0 0 0 0 0 0 0 0 0	Reset

I2C_SDA_SAMPLE_TIME I2C 采 SDA 时间, APB 时 期 。(/写)

Register 11.14. I2C_SCL_HIGH_PERIOD_REG (0x0038)

(reserved)	I2C_SCL_HIGH_PERIOD	
31 … 14	13 … 0	
0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0	0 0 0 0 0 0 0 0 0 0 0 0 0 0	Reset

I2C_SCL_HIGH_PERIOD I2C Master 时, SCL APB 时 期 。(/写)

Register 11.15. I2C_SCL_START_HOLD_REG (0x0040)

(reserved)	I2C_SCL_START_HOLD_TIME	
31 … 10	9 … 0	
0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0	0 0 0 0 0 0 1 0 0 0	Reset

I2C_SCL_START_HOLD_TIME SDA SCL 间 时间, START , APB 时 期 。(/写)

Register 11.16. I2C_SCL_RSTART_SETUP_REG (0x0044)

(reserved)	I2C_SCL_RSTART_SETUP_TIME	
31 … 10	9 … 0	
0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0	0 0 0 0 0 0 1 0 0 0	Reset

I2C_SCL_RSTART_SETUP_TIME SCL 上升 SDA 间 时间，RESTART ，APB 时 期 。(/写)

Register 11.17. I2C_SCL_STOP_HOLD_REG (0x0048)

(reserved)	I2C_SCL_STOP_HOLD_TIME	
31 … 14	13 … 0	
0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0	0 0 0 0 0 0 0 0 0 0 0 0 0 0	Reset

I2C_SCL_STOP_HOLD_TIME STOP ，APB 时 期 。(/写)

Register 11.18. I2C_SCL_STOP_SETUP_REG (0x004C)

(reserved)	I2C_SCL_STOP_SETUP_TIME	
31 … 10	9 … 0	
0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0	0 0 0 0 0 0 0 0 0 0	Reset

I2C_SCL_STOP_SETUP_TIME SCL 上升 SDA 上升 间 时间，APB 时 期 。(/写)

Register 11.19. I2C_SCL_FILTER_CFG_REG (0x0050)

(reserved)	I2C_SCL_FILTER_EN	I2C_SCL_FILTER_THRES	
31 … 4	3	2 … 0	
0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0	1	0 0 0	Reset

I2C_SCL_FILTER_EN SCL 波 。(/写)

I2C_SCL_FILTER_THRES APB 时 期 , SCL 上 脉 时, I2C 脉 。(/写)

Register 11.20. I2C_SDA_FILTER_CFG_REG (0x0054)

(reserved)	I2C_SDA_FILTER_EN	I2C_SDA_FILTER_THRES	
31 … 4	3	2 … 0	
0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0	1	0 0 0	Reset

I2C_SDA_FILTER_EN SDA 波 。(/写)

I2C_SDA_FILTER_THRES APB 时 期 , SDA 上 脉 时, I2C 脉 。(/写)

Register 11.21. I2C_COMD*n*_REG (*n*: 0-15) (0x58+4n*)**

I2C_COMMAND*n*_DONE	(reserved)	I2C_COMMAND*n*	
31	30 … 14	13 … 0	
0	0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0	0 0 0 0 0 0 0 0 0 0 0 0 0 0	Reset

I2C_COMMAND*n*_DONE I2C command *n* 时, 。(/写)

I2C_COMMAND*n* command *n* 。 包括三个部 :(/写)

op_code , 0: START; 1: WRITE; 2: READ; 3: STOP; 4: END。

Byte_num 表 被 。

ack_check_en, ack_exp ack ACK 。 参 I2C cmd 。

12 I2S 控制器 (I2S)

12.1 概述

I2S , 。ESP32 个 I2S , I2S0 I2S1。

I2S 标 三 : 时 BCK、声 WS SD。一个 本 I2S 一个 一个 。 中保 不变。ESP32 I2S 包 声 , 保 。

图 12-1. I2S 系统框图

12-1 ESP32 I2S , 中 "*n*" 0 1, I2S0 I2S1。 个 I2S 包 一个 单 (Tx) 一个 单 (Rx)。 单 一 三 , 别 时 BCK, 声 WS SD。 中, 单 , 单 。 单 单 时 声 可 。 LCD , 扩 并 。I2S 单 一 32 bit、 64 bit FIFO。 , I2S0 / PDM 并 上 DAC/ADC 。

12-1 侧 I2S 。Rx Tx : I2S*n*A_B_C。 中 "*n*" , 表 I2S0 I2S1; "A" 表 I2S 广 , "I" 表 , "O" 表 ; "B" 表 ; "C" 表 广 , "in" 表 I2S , "out" 表 I2S 。 见表 12-1。 I2S*n*_CLK , GPIO IO_MUX 广 。 I2S*n*_CLK IO_MUX 广 。 参 IO_MUX GPIO 。

3 F H J T U F S 6) $ * @ $ 0 / ' @ 3 & (Y $

	SFTFSWFE	6) $ * @ 5 9 @ " $, @ / 6 . @ 3 & 6) $ * @ 5 9 @ $) & $, @ 4 6 . @ 3 &	SFTFSWFE	6) $ * @ $) & $, @ 4 & 2 @ & / 6) $ * @ $) & $, @ 4 6 . @ & /		
j R	e	8	9	j k	R	y
						_ 2 b 2 i

6) $ * @ 5 9 @ " $, @ / 6 . @ 3 & SÄ ¹ b •

6) $ * @ 5 9 @ $) & $, @ 4 6 . @ 3 & SÄ ¹ b •

6) $ * @ $) & $, @ 4 & 2 @ & / SÄ ¹ b •

6) $ * @ $) & $, @ 4 6 . @ & / SÄ ¹ b •

3 F H J T U F S 6) $ * @ % . " @ 0 6 5 @ & 0 ' @ % & 4 @ " % % 3 @ 3 & (Y

j R	y
Y	_ 2 b 2 i

6) $ * @ % . " @ 0 6 5 @ & 0 ' @ % & 4 @ & 0 'Ê ¹ H ¥ ¹ ¶ b º

3 F H J T U F S 6) $ * @ % . " @ * / @ 4 6 $ @ & 0 ' @ % & 4 @ " % % 3 @ 3 & (Y $

j R	y
Y	_ 2 b 2 i

6) $ * @ % . " @ * / @ 4 6 $ @ & 0 ' @ % & 4 @ & 0 'Ê ¹ H ¥ ¹ ¶ b º

3 F H J T U F S 6) $ * @ % . " @ * / @ & 3 3 @ & 0 ' @ % & 4 @ " % % 3 @ 3 & (Y

j R	y
Y	_ 2 b 2 i

6) $ * @ % . " @ * / @ & 3 3 @ & 0 ' @ % & 4 H ¥ ¹ ¶ b º

3 F H J T U F S 6) $ * @ % . " @ 0 6 5 @ & 0 ' @ # ' 3 @ % & 4 @ " % % 3 @ 3 & (Y

j R y

Y _ 2 b 2 i

6) $ * @ % . " @ 0 6 5 @ & 0 ' @ # ' 3 @ % & 4 @ % % ?Â ÿ @ í 3 ¯ Ï(µ p H ¥ ¹ ¶ b º

3 F H J T U F S 6) $ * @ % . " @ * / @ % 4 $ 3 @ 3 & (Y $

j R y

_ 2 b 2 i

6) $ * @ % . " @ * / @ % 4 $ 3 @ Bý&V(í • ˜ ¥ ¹ ¶ b º

3 F H J T U F S 6) $ * @ % . " @ * / @ % 4 $ 3 @ # ' @ 3 & (Y

j R y

_ 2 b 2 i

6) $ * @ % . " @ * / @ % 4 $ 3 @ # ' ¤ @ ý 3 V & í (• ˜ ¥ - ë » ñ ¥ ¹ ¶ b º

3 F H J T U F S 6) $ * @ % . " @ * / @ % 4 $ 3 @ # ' @ 3 & (Y

j R y

_ 2 b 2 i

6) $ * @ % . " @ * / @ % 4 $ 3 @ # ' ¤ @ ý 3 V & í (• ˜ ¥ - ë » ñ ¥ ¹ ¶ b º

3 F H J T U F S 6) $ * @ % . " @ 0 6 5 @ % 4 $ 3 @ 3 & (Y

j R y

_ 2 b 2 i

6) $ * @ % . " @ 0 6 5 @ % 4 $ 3 @ Â ý & V(í • ˜ ¥ ¹ ¶ b º

3 F H J T U F S 6) $ * @ % . " @ 0 6 5 @ % 4 $ 3 @ # ' @ 3 & (Y $

j R y

_ 2 b 2 i

6) $ * @ % . " @ 0 6 5 @ % 4 $ 3 @ # ' ? @ Â ý 3 V & í (• ˜ ¥ - ë » ñ ¥ ¹ ¶ b º

3 F H J T U F S 6) $ * @ % . " @ 0 6 5 @ % 4 $ 3 @ # ' @ 3 & (Y

j R	y
	_ 2 b 2 i

6) $ * @ % . " @ 0 6 5 @ % 4 $ 3 @ # ¹ ? @ Â ? ¥ & (• ˜ ¥ - ë » ñ ¥ ¹ ¶ b º

3 F H J T U F S 6) $ * @ & 4 $ " 1 & @ $ 0 / ' @ 3 & (Y

S F T F S W F E

6) $ * @ 3 9 @ @ & 4 $ @ & / 6) $ * @ 3 9 @ @ & 4 $ @ & / 6) $ * @ 3 9 @ % # @ & 4 $ @ & / 6) $ * @ 3 9 @ $ @ & 4 $ @ & / 6) $ * @ 5 9 @ @ & 4 $ @ & / 6) $ * @ 5 9 @ @ & 4 $ @ & / 6) $ * @ 5 9 @ % # @ & 4 $ @ & / 6) $ * @ 5 9 @ $ @ & 4 $ @ & /

j R	3	d	e	8	9	j	k	R	y
									_ 2 b 2 i

6) $ * @ 3 9 @ @ & 4 Â $ Ê @ P & / % . " ? Â " H @ e 3 ˜ Y ¥ 9 Ð b •

6) $ * @ 3 9 @ @ & 4 Â $ Ê @ P & / % . " ? Â " H @ e 3 ˜ Y ¥ 9 Ð b •

6) $ * @ 3 9 @ % # @ & Â 4 $ Ê @ P & / % . " ? Â " H @ e 3 ˜ Y E ¥ 9 Ð b •

6) $ * @ 3 9 @ $ @ & 4 Â $ Ê @ P & / % . " ? Â " H @ e 3 ˜ Y D ¥ 9 Ð b •

6) $ * @ 5 9 @ @ & 4 Â $ Ê @ P & / % . " ¤ l " H @ e 3 ˜ Y ¥ ³ ' b •

6) $ * @ 5 9 @ @ & 4 Â $ Ê @ P & / % . " ¤ l " H @ e 3 ˜ Y ¥ ³ ' b •

6) $ * @ 5 9 @ % # @ & Â 4 $ Ê @ P & / % . " ¤ l " H @ e 3 ˜ Y E ¥ ³ ' b •

6) $ * @ 5 9 @ $ @ & 4 Â $ Ê @ P & / % . " ¤ l " H @ e 3 ˜ Y D ¥ ³ ' b •

3FHJTUFS 6)$*@)6/(@$0/'@3&(/" Y

SFTFSWFE	6)$*@39'*'0@5*.&065@&/"	6)$*@39'*'0@5*.&065@4)*'5	6)$*@39'*'0@5*.&065	6)$*@59'*'0@5*.&065@&/"	6)$*@59'*'0@5*.&065@4)*'5	6)$*@59'*'0@5*.&065
jR k9	kj	kk ky	RN Rk	RR Ry 3	d y	
			Y			Y

_2b2i

6)$*@39'*'0@5*.&065%@.&?"Â" ÑH¥P?Êb •

6)$*@39'*'0@5*.&065@94)*'5 E xSFH@SYGJGPI@9U"JNbFPVU@TIJGU
,b •

6)$*@39'*'0@5*.&065%ÑH´b' %."V3".Ï |" ¥HWÑV¥Â´H öÁ3
6)$*@39@)6/(Ï@b*/5 •

6)$*@59'*'0@5*.&065?@A&/"0¤l" ÑH¥P?Êb •

6)$*@59'*'0@5*.&065@94)*'5 E xSFH@UYGJGPI@9U"JNbFPVU@TIJGU
,b •

6)$*@59'*'0@5*.&065%ÑH´b' %."V3".Ï |" ¥HWÑV¥Â´H öÁ3
6)$*@59@)6/(Ï@b*/5 •

3FHJTUFS 6)$*@&4$@$0/'O@3&?(O Y# O

SFTFSWFE	6)$*@&4$@4&2 @$)	6)$*@&4$@4&2 @$)	6)$*@&4$@4&2
jR k9	kj Re	R8 3	d y
	Y %'	Y %#	Y

_2b2i

6)$*@&4$@4&2 @$%"3¿9Ð" Ï SFH@FTD¥@TFñR3˜b •

6)$*@&4$@4&2 @$%"3¿9Ð" Ï SFH@FTD¥@TFñR3˜b •

6)$*@&4$@4&2 GMPX@D3P"Q1U>SPMGMPX@bDPO•USPM

14 LED PWM 控制器 (LEDC)

14.1 概述

LED_PWM LED ，可产 PWM 。LED_PWM 16 ，8 8 。16 产 波 RGB LED 设备。

可 个 时 一 个 时 一 。PWM 比， 变。LED_PWM 。

本 中，hsch*n* ，lsch*n* 。 时 时 别 h_timer*x* l_timer*x*。

14.2 功能描述

14.2.1 架构

14-1 LED_PWM 本 。 中可 ，LED_PWM 部 8 个 8 个 。4 个 时 ，可 中 一个 h_timer*x*。 4 个 时 ，可 中 一个 l_timer*x*。

图 14-1. LED_PWM 架构

14-2 表 一个 PWM ； ，一个 一个 。

图 14-2. LED_PWM 高速通道框图

14.2.2 分频器

图 14-3. LED_PWM 分频器

一个 时 ， 可 2 个时 中 一个时 ：REF_TICK APB_CLK（ 时 参 时 ）。 时 ， LEDC_CLK_DIV_NUM_HSTIMERx， 18 ： 中 10 部 A， 8 部 B。 ：

$$LEDC_CLK_DIV_NUM_HSTIMER x = A\frac{B}{256}$$

部 不 0 时， 时 14-3 。

2. 。 LEDC_LSTIMERx_PARA_UP ， 参 。

14.2.3 通道

个 个比 ， 14-2 high_level_comparator low_level_comparator。
high_level_comparator 比 hpoint， LEDC_HPOINT_HSCHn 。 hpoint 时，
。
low_level_comparator 比 lpoint， LEDC_DUTY_HSCHn，LEDC_DUTY_START_HSCHn，
LEDC_DUTY_INC_HSCHn，LEDC_DUTY_NUM_HSCHn LEDC_DUTY_SCALE_HSCHn 。
lpoint 时， 。 14-4 LED_PWM 。

图 14-4. LED_PWM 输出信号图

LEDC_DUTY_HSCHn 一个 4 ， 20 部 ， 4 部 。
4 0 时， 脉 LEDC_DUTY_HSCHn [3：0]/16 一个 期。 4
比 。 LEDC_DUTY_START_HSCHn， LEDC_DUTY_HSCHn
才 ， 可 一 比 一 比 。
LEDC_DUTY_INC_HSCHn 比 变化方 。 变 比 ：
LEDC_DUTY_CYCLE_HSCHn 个脉 ，LEDC_DUTY_HSCHn 20
LEDC_DUTY_SCALE_HSCHn， 变长 LEDC_DUTY_NUM_HSCHn ， 变 ， 产 变
中 ， 一 脉 。 14-5 LEDC_DUTY_INC_HSCHn 1 时 变 ，
LEDC_DUTY_CYCLE_HSCHn 个脉 ， 脉 LEDC_DUTY_SCALE_HSCHn。

图 14-5. 渐变占空比输出信号图

注意

- LEDC 变 时，必 LEDC_DUTY_CHNG_END_HSCHn LEDC_DUTY_CHNG_END_LSCHn 中 才可 变 。
- LEDC 变 LEDC_DUTY_HSCHn $2^{LEDC_HSTIMERx_DUTY_RES}$ 时，不 LEDC_DUTY_SCALE_HSCHn 1； LEDC 变 LEDC_DUTY_LSCHn $2^{LEDC_LSTIMERx_DUTY_RES}$ 时，不 LEDC_DUTY_SCALE_LSCHn 1。

14.2.4 中断

- LEDC_DUTY_CHNG_END_LSCH*n*_INT 上 比 变 中 。
- LEDC_DUTY_CHNG_END_HSCH*n*_INT 上 比 变 中 。
- LEDC_HS_TIMER*n*_OVF_INT 时 中 。
- LEDC_LS_TIMER*n*_OVF_INT 时 中 。

14.3 寄存器列表

配置寄存器			
LEDC_CONF_REG	Global ledc configuration register	0x3FF59190	/ 写
LEDC_HSCH0_CONF0_REG	Configuration register 0 for high-speed channel 0	0x3FF59000	/ 写
LEDC_HSCH1_CONF0_REG	Configuration register 0 for high-speed channel 1	0x3FF59014	/ 写
LEDC_HSCH2_CONF0_REG	Configuration register 0 for high-speed channel 2	0x3FF59028	/ 写
LEDC_HSCH3_CONF0_REG	Configuration register 0 for high-speed channel 3	0x3FF5903C	/ 写
LEDC_HSCH4_CONF0_REG	Configuration register 0 for high-speed channel 4	0x3FF59050	/ 写
LEDC_HSCH5_CONF0_REG	Configuration register 0 for high-speed channel 5	0x3FF59064	/ 写
LEDC_HSCH6_CONF0_REG	Configuration register 0 for high-speed channel 6	0x3FF59078	/ 写
LEDC_HSCH7_CONF0_REG	Configuration register 0 for high-speed channel 7	0x3FF5908C	/ 写
LEDC_HSCH0_CONF1_REG	Configuration register 1 for high-speed channel 0	0x3FF5900C	/ 写
LEDC_HSCH1_CONF1_REG	Configuration register 1 for high-speed channel 1	0x3FF59020	/ 写
LEDC_HSCH2_CONF1_REG	Configuration register 1 for high-speed channel 2	0x3FF59034	/ 写
LEDC_HSCH3_CONF1_REG	Configuration register 1 for high-speed channel 3	0x3FF59048	/ 写
LEDC_HSCH4_CONF1_REG	Configuration register 1 for high-speed channel 4	0x3FF5905C	/ 写
LEDC_HSCH5_CONF1_REG	Configuration register 1 for high-speed channel 5	0x3FF59070	/ 写
LEDC_HSCH6_CONF1_REG	Configuration register 1 for high-speed channel 6	0x3FF59084	/ 写
LEDC_HSCH7_CONF1_REG	Configuration register 1 for high-speed channel 7	0x3FF59098	/ 写
LEDC_LSCH0_CONF0_REG	Configuration register 0 for low-speed channel 0	0x3FF590A0	/ 写
LEDC_LSCH1_CONF0_REG	Configuration register 0 for low-speed channel 1	0x3FF590B4	/ 写
LEDC_LSCH2_CONF0_REG	Configuration register 0 for low-speed channel 2	0x3FF590C8	/ 写
LEDC_LSCH3_CONF0_REG	Configuration register 0 for low-speed channel 3	0x3FF590DC	/ 写
LEDC_LSCH4_CONF0_REG	Configuration register 0 for low-speed channel 4	0x3FF590F0	/ 写
LEDC_LSCH5_CONF0_REG	Configuration register 0 for low-speed channel 5	0x3FF59104	/ 写
LEDC_LSCH6_CONF0_REG	Configuration register 0 for low-speed channel 6	0x3FF59118	/ 写
LEDC_LSCH7_CONF0_REG	Configuration register 0 for low-speed channel 7	0x3FF5912C	/ 写
LEDC_LSCH0_CONF1_REG	Configuration register 1 for low-speed channel 0	0x3FF590AC	/ 写
LEDC_LSCH1_CONF1_REG	Configuration register 1 for low-speed channel 1	0x3FF590C0	/ 写
LEDC_LSCH2_CONF1_REG	Configuration register 1 for low-speed channel 2	0x3FF590D4	/ 写
LEDC_LSCH3_CONF1_REG	Configuration register 1 for low-speed channel 3	0x3FF590E8	/ 写
LEDC_LSCH4_CONF1_REG	Configuration register 1 for low-speed channel 4	0x3FF590FC	/ 写
LEDC_LSCH5_CONF1_REG	Configuration register 1 for low-speed channel 5	0x3FF59110	/ 写
LEDC_LSCH6_CONF1_REG	Configuration register 1 for low-speed channel 6	0x3FF59124	/ 写
LEDC_LSCH7_CONF1_REG	Configuration register 1 for low-speed channel 7	0x3FF59138	/ 写

占空比寄存器			
LEDC_HSCH0_DUTY_REG	Initial duty cycle for high-speed channel 0	0x3FF59008	/ 写
LEDC_HSCH1_DUTY_REG	Initial duty cycle for high-speed channel 1	0x3FF5901C	/ 写
LEDC_HSCH2_DUTY_REG	Initial duty cycle for high-speed channel 2	0x3FF59030	/ 写
LEDC_HSCH3_DUTY_REG	Initial duty cycle for high-speed channel 3	0x3FF59044	/ 写
LEDC_HSCH4_DUTY_REG	Initial duty cycle for high-speed channel 4	0x3FF59058	/ 写
LEDC_HSCH5_DUTY_REG	Initial duty cycle for high-speed channel 5	0x3FF5906C	/ 写
LEDC_HSCH6_DUTY_REG	Initial duty cycle for high-speed channel 6	0x3FF59080	/ 写
LEDC_HSCH7_DUTY_REG	Initial duty cycle for high-speed channel 7	0x3FF59094	/ 写
LEDC_HSCH0_DUTY_R_REG	Current duty cycle for high-speed channel 0	0x3FF59010	
LEDC_HSCH1_DUTY_R_REG	Current duty cycle for high-speed channel 1	0x3FF59024	
LEDC_HSCH2_DUTY_R_REG	Current duty cycle for high-speed channel 2	0x3FF59038	
LEDC_HSCH3_DUTY_R_REG	Current duty cycle for high-speed channel 3	0x3FF5904C	
LEDC_HSCH4_DUTY_R_REG	Current duty cycle for high-speed channel 4	0x3FF59060	
LEDC_HSCH5_DUTY_R_REG	Current duty cycle for high-speed channel 5	0x3FF59074	
LEDC_HSCH6_DUTY_R_REG	Current duty cycle for high-speed channel 6	0x3FF59088	
LEDC_HSCH7_DUTY_R_REG	Current duty cycle for high-speed channel 7	0x3FF5909C	
LEDC_LSCH0_DUTY_REG	Initial duty cycle for low-speed channel 0	0x3FF590A8	/ 写
LEDC_LSCH1_DUTY_REG	Initial duty cycle for low-speed channel 1	0x3FF590BC	/ 写
LEDC_LSCH2_DUTY_REG	Initial duty cycle for low-speed channel 2	0x3FF590D0	/ 写
LEDC_LSCH3_DUTY_REG	Initial duty cycle for low-speed channel 3	0x3FF590E4	/ 写
LEDC_LSCH4_DUTY_REG	Initial duty cycle for low-speed channel 4	0x3FF590F8	/ 写
LEDC_LSCH5_DUTY_REG	Initial duty cycle for low-speed channel 5	0x3FF5910C	/ 写
LEDC_LSCH6_DUTY_REG	Initial duty cycle for low-speed channel 6	0x3FF59120	/ 写
LEDC_LSCH7_DUTY_REG	Initial duty cycle for low-speed channel 7	0x3FF59134	/ 写
LEDC_LSCH0_DUTY_R_REG	Current duty cycle for low-speed channel 0	0x3FF590B0	
LEDC_LSCH1_DUTY_R_REG	Current duty cycle for low-speed channel 1	0x3FF590C4	
LEDC_LSCH2_DUTY_R_REG	Current duty cycle for low-speed channel 2	0x3FF590D8	
LEDC_LSCH3_DUTY_R_REG	Current duty cycle for low-speed channel 3	0x3FF590EC	
LEDC_LSCH4_DUTY_R_REG	Current duty cycle for low-speed channel 4	0x3FF59100	
LEDC_LSCH5_DUTY_R_REG	Current duty cycle for low-speed channel 5	0x3FF59114	
LEDC_LSCH6_DUTY_R_REG	Current duty cycle for low-speed channel 6	0x3FF59128	
LEDC_LSCH7_DUTY_R_REG	Current duty cycle for low-speed channel 7	0x3FF5913C	
分频器寄存器			
LEDC_HSTIMER0_CONF_REG	High-speed timer 0 configuration	0x3FF59140	/ 写
LEDC_HSTIMER1_CONF_REG	High-speed timer 1 configuration	0x3FF59148	/ 写
LEDC_HSTIMER2_CONF_REG	High-speed timer 2 configuration	0x3FF59150	/ 写
LEDC_HSTIMER3_CONF_REG	High-speed timer 3 configuration	0x3FF59158	/ 写
LEDC_HSTIMER0_VALUE_REG	High-speed timer 0 current counter value	0x3FF59144	
LEDC_HSTIMER1_VALUE_REG	High-speed timer 1 current counter value	0x3FF5914C	
LEDC_HSTIMER2_VALUE_REG	High-speed timer 2 current counter value	0x3FF59154	
LEDC_HSTIMER3_VALUE_REG	High-speed timer 3 current counter value	0x3FF5915C	
LEDC_LSTIMER0_CONF_REG	Low-speed timer 0 configuration	0x3FF59160	/ 写

LEDC_LSTIMER1_CONF_REG	Low-speed timer 1 configuration	0x3FF59168	/ 写
LEDC_LSTIMER2_CONF_REG	Low-speed timer 2 configuration	0x3FF59170	/ 写
LEDC_LSTIMER3_CONF_REG	Low-speed timer 3 configuration	0x3FF59178	/ 写
LEDC_LSTIMER0_VALUE_REG	Low-speed timer 0 current counter value	0x3FF59164	
LEDC_LSTIMER1_VALUE_REG	Low-speed timer 1 current counter value	0x3FF5916C	
LEDC_LSTIMER2_VALUE_REG	Low-speed timer 2 current counter value	0x3FF59174	
LEDC_LSTIMER3_VALUE_REG	Low-speed timer 3 current counter value	0x3FF5917C	
中断寄存器			
LEDC_INT_RAW_REG	Raw interrupt status	0x3FF59180	
LEDC_INT_ST_REG	Masked interrupt status	0x3FF59184	
LEDC_INT_ENA_REG	Interrupt enable bits	0x3FF59188	/ 写
LEDC_INT_CLR_REG	Interrupt clear bits	0x3FF5918C	写

14.4 寄存器

Register 14.1. LEDC_HSCH*n*_CONF0_REG (*n*: 0-7) (0x1C+0x10n*)**

(reserved)	LEDC_IDLE_LV_HSCH*n*	LEDC_SIG_OUT_EN_HSCH*n*	LEDC_TIMER_SEL_HSCH*n*	
31 … 4	3	2	1 … 0	
0x00000000	0	0	0	Reset

LEDC_IDLE_LV_HSCH*n* *n* 不 时， 。(/ 写)

LEDC_SIG_OUT_EN_HSCH*n* *n* 。(/ 写)

LEDC_TIMER_SEL_HSCH*n* 4 时 中 *n* 一个时 。(/ 写)

0: hstimer0；

1: hstimer1；

2: hstimer2；

3: hstimer3。

Register 14.2. LEDC_HSCH*n*_HPOINT_REG (*n*: 0-7) (0x20+0x10n*)**

(reserved)	LEDC_HPOINT_HSCH*n*	
31 … 20	19 … 0	
0x0000	0x000000	Reset

LEDC_HPOINT_HSCH*n* *n* htimerx(x=[0,3]) LEDC_HPOINT_HSCH*n*[19:0]， 。(/ 写)

Register 14.3. LEDC_HSCH*n*_DUTY_REG (*n*: 0-7) (0x24+0x10n*)**

(reserved)	LEDC_DUTY_HSCH*n*	
31 … 25	24 … 0	
0x00	0x0000000	Reset

LEDC_DUTY_HSCH*n* 比。 *n* hstimerx(x=[0,3])

LEDC_LPOINT_HSCH*n* 时， 。

LEDC_LPOINT_HSCH*n*=LEDC_LPOINT_HSCH*n*[19:0]+LEDC_DUTY_HSCH*n*[24:4]) (1)

LEDC_LPOINT_HSCH*n*=LEDC_LPOINT_HSCH*n*[19:0]+LEDC_DUTY_HSCH*n*[24:4] +1) (2)

1，2 时， 中 。

Register 14.4. LEDC_HSCH*n*_CONF1_REG (*n*: 0-7) (0x28+0x10n*)**

LEDC_DUTY_START_HSCH*n*	LEDC_DUTY_INC_HSCH*n*	LEDC_DUTY_NUM_HSCH*n*	LEDC_DUTY_CYCLE_HSCH*n*	LEDC_DUTY_SCALE_HSCH*n*	
31	30	29 … 20	19 … 10	9 … 0	
0	1	0x000	0x000	0x000	Reset

LEDC_DUTY_START_HSCH*n* LEDC_DUTY_NUM_HSCH*n*,

LEDC_DUTY_CYCLE_HSCH*n* LEDC_DUTY_SCALE_HSCH*n* 时，

LEDC_DUTY_START_HSCH*n*， 才 。 。(/ 写)

LEDC_DUTY_INC_HSCH*n* *n* 比。(/ 写)

LEDC_DUTY_NUM_HSCH*n* *n* 比变化 。(/ 写)

LEDC_DUTY_CYCLE_HSCH*n* LEDC_DUTY_CYCLE_HSCH*n* 个时 期 *n* 比。

LEDC_DUTY_SCALE_HSCH*n* *n* 步长。(/ 写)

Register 14.5. LEDC_HSCH*n*_DUTY_R_REG (*n*: 0-7) (0x2C+0x10**n*)

(reserved)	LEDC_DUTY_LSCH*n*_R	
31 25	24 0	
0x00	0x0000000	Reset

LEDC_DUTY_HSCH*n*_R *n* 比。( )

Register 14.6. LEDC_LSCH*n*_CONF0_REG (*n*: 0-7) (0xBC+0x10**n*)

(reserved)	LEDC_PARA_UP_LSCH*n*	LEDC_IDLE_LV_LSCH*n*	LEDC_SIG_OUT_EN_LSCH*n*	LEDC_TIMER_SEL_LSCH*n*	
31 5	4	3	2	1 0	
0x0000000	0	0	0	0	Reset

LEDC_PARA_UP_LSCH*n* *n* LEDC_LSCH*n*_HPOINT LEDC_LSCH*n*_DUTY 。(/ 写)

LEDC_IDLE_LV_LSCH*n* *n* 不 时， 。(/ 写)

LEDC_SIG_OUT_EN_LSCH*n* *n* 。(/ 写)

LEDC_TIMER_SEL_LSCH*n* 4 中 *n* 一个时 。(/ 写)

0: lstimer0；
1: lstimer1；
2: lstimer2；
3: lstimer3。

Register 14.7. LEDC_LSCH*n*_HPOINT_REG (*n*: 0-7) (0xC0+0x10**n*)

(reserved)	LEDC_HPOINT_LSCH*n*	
31 20	19 0	
0x0000	0x000000	Reset

LEDC_HPOINT_LSCH*n* *n* lstimerx(x=[0,3]) LEDC_HPOINT_LSCH

Register 14.8. LEDC_LSCH*n*_DUTY_REG (*n*: 0-7) (0xC4+0x10n*)**

(reserved)	LEDC_DUTY_LSCH*n*	
31 … 25	24 … 0	
0x00	0x0000000	Reset

LEDC_DUTY_LSCH*n* 比。 *n* lstimerx(x=[0,3])

LEDC_LPOINT_LSCH*n*时， 。(/ 写)

LEDC_LPOINT_LSCH*n*=(LEDC_HPOINT_LSCH*n*[19:0]+LEDC_DUTY_LSCH*n*[24:4]) (1)

LEDC_LPOINT_LSCH*n*=(LEDC_HPOINT_LSCH*n*[19:0]+LEDC_DUTY_LSCH*n*[24:4] +1) (2)

1，2时， 中 。

Register 14.9. LEDC_LSCH*n*_CONF1_REG (*n*: 0-7) (0xC8+0x10n*)**

LEDC_DUTY_START_LSCH*n*	LEDC_DUTY_INC_LSCH*n*	LEDC_DUTY_NUM_LSCH*n*	LEDC_DUTY_CYCLE_LSCH*n*	LEDC_DUTY_SCALE_LSCH*n*	
31	30	29 … 20	19 … 10	9 … 0	
0	1	0x000	0x000	0x000	Reset

LEDC_DUTY_START_LSCH*n* LEDC_DUTY_NUM_HSCH*n*、LEDC_DUTY_CYCLE_HSCH*n* LEDC_DUTY_SCALE_HSCH*n* 时，LEDC_DUTY_START_HSCH*n*， 才 。 。(/ 写)

LEDC_DUTY_INC_LSCH*n* *n* 比。(/ 写)

LEDC_DUTY_NUM_LSCH*n* *n* 比变化 。(/ 写)

LEDC_DUTY_CYCLE_LSCH*n* LEDC_DUTY_CYCLE_LSCH*n* 个时 期 比。(/ 写)

LEDC_DUTY_SCALE_LSCH*n* *n* 步长。(/ 写)

Register 14.10. LEDC_LSCHn_DUTY_R_REG (n: 0-7) (0xCC+0x10*n)

(reserved)	LEDC_DUTY_LSCHn_R	
31 25	24 0	
0x00	0x0000000	Reset

LEDC_DUTY_R_LSCHn_R n 比。( )

Register 14.11. LEDC_HSTIMERx_CONF_REG (x: 0-3) (0x140+8*x)

(reserved)	LEDC_TICK_SEL_HSTIMERx	LEDC_HSTIMERx_RST	LEDC_HSTIMERx_PAUSE	LEDC_CLK_DIV_NUM_HSTIMERx	LEDC_HSTIMERx_DUTY_RES	
31 26	25	24	23	22 5	4 0	
0x00	0	1	0	0x00000	0x00	Reset

LEDC_TICK_SEL_HSTIMERx APB_CLK REF_TICK 时 。(/ 写)

1：APB_CLK；

0：REF_TICK。

LEDC_HSTIMERx_RST 时 x， 0。(/ 写)

LEDC_HSTIMERx_PAUSE 时 x。(/ 写)

LEDC_CLK_DIV_NUM_HSTIMERx 时 x， 8 部 。(/ 写)

LEDC_HSTIMERx_DUTY_RES 时 x 。

[0,2**LEDC_HSTIMERx_DUTY_RES]， 20 。(/ 写)

Register 14.12. LEDC_HSTIMERx_VALUE_REG (x: 0-3) (0x144+8*x)

(reserved)	LEDC_HSTIMERx_CNT	
31 20	19 0	
0x0000	0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0	Reset

LEDC_HSTIMERx_CNT 可 时 x 。( )

Register 14.13. LEDC_LSTIMERx_CONF_REG (x: 0-3) (0x160+8*x)

(reserved)	LEDC_LSTIMERx_PARA_UP	LEDC_TICK_SEL_LSTIMERx	LEDC_LSTIMERx_RST	LEDC_LSTIMERx_PAUSE	LEDC_CLK_DIV_NUM_LSTIMERx	LEDC_LSTIMERx_DUTY_RES	
31 27	26	25	24	23	22 5	4 0	
0x00	0	0	1	0	0x00000	0x00	Reset

LEDC_LSTIMERx_PARA_UP LEDC_CLK_DIV_NUM_LSTIMERx LEDC_LSTIMERx_DUTY_RES。

LEDC_TICK_SEL_LSTIMERx 时 x SLOW_CLK REF_TICK 时 。(/ 写)

1：SLOW_CLK；

0：REF_TICK。

LEDC_LSTIMERx_RST 时 x, 0。(/ 写)

LEDC_LSTIMERx_PAUSE 时 x。(/ 写)

LEDC_CLK_DIV_NUM_LSTIMERx 时 x, 8 部 。(/ 写)

LEDC_LSTIMERx_DUTY_RES 时 x 。[0,2**LEDC_LSTIMERx_DUTY_RES], 20 。(/ 写)

Register 14.14. LEDC_LSTIMERx_VALUE_REG (x: 0-3) (0x164+8*x)

(reserved)	LEDC_LSTIMERx_CNT	
31 20	19 0	
0x0000	0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0	Reset

LEDC_LSTIMERx_CNT 时 x 。( )

Register 14.15. LEDC_INT_RAW_REG (0x0180)

31–24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0	
(reserved)	LEDC_DUTY_CHNG_END_LSCH7_INT_RAW	LEDC_DUTY_CHNG_END_LSCH6_INT_RAW	LEDC_DUTY_CHNG_END_LSCH5_INT_RAW	LEDC_DUTY_CHNG_END_LSCH4_INT_RAW	LEDC_DUTY_CHNG_END_LSCH3_INT_RAW	LEDC_DUTY_CHNG_END_LSCH2_INT_RAW	LEDC_DUTY_CHNG_END_LSCH1_INT_RAW	LEDC_DUTY_CHNG_END_LSCH0_INT_RAW	LEDC_DUTY_CHNG_END_HSCH7_INT_RAW	LEDC_DUTY_CHNG_END_HSCH6_INT_RAW	LEDC_DUTY_CHNG_END_HSCH5_INT_RAW	LEDC_DUTY_CHNG_END_HSCH4_INT_RAW	LEDC_DUTY_CHNG_END_HSCH3_INT_RAW	LEDC_DUTY_CHNG_END_HSCH2_INT_RAW	LEDC_DUTY_CHNG_END_HSCH1_INT_RAW	LEDC_DUTY_CHNG_END_HSCH0_INT_RAW	LEDC_LSTIMER3_OVF_INT_RAW	LEDC_LSTIMER2_OVF_INT_RAW	LEDC_LSTIMER1_OVF_INT_RAW	LEDC_LSTIMER0_OVF_INT_RAW	LEDC_HSTIMER3_OVF_INT_RAW	LEDC_HSTIMER2_OVF_INT_RAW	LEDC_HSTIMER1_OVF_INT_RAW	LEDC_HSTIMER0_OVF_INT_RAW	
0 0 0 0 0 0 0 0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	Reset

LEDC_DUTY_CHNG_END_LSCH*n*_INT_RAW LEDC_DUTY_CHNG_END_LSCH*n*_INT 中 始 中 。( )

LEDC_DUTY_CHNG_END_HSCH*n*_INT_RAW LEDC_DUTY_CHNG_END_HSCH*n*_INT 中 始 中 。( )

LEDC_LSTIMER*x*_OVF_INT_RAW LEDC_LSTIMER*x*_OVF_INT 中 始 中 。( )

LEDC_HSTIMER*x*_OVF_INT_RAW LEDC_HSTIMER*x*_OVF_INT 中 始 中 。( )

Register 14.16. LEDC_INT_ST_REG (0x0184)

31–24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0	
(reserved)	LEDC_DUTY_CHNG_END_LSCH7_INT_ST	LEDC_DUTY_CHNG_END_LSCH6_INT_ST	LEDC_DUTY_CHNG_END_LSCH5_INT_ST	LEDC_DUTY_CHNG_END_LSCH4_INT_ST	LEDC_DUTY_CHNG_END_LSCH3_INT_ST	LEDC_DUTY_CHNG_END_LSCH2_INT_ST	LEDC_DUTY_CHNG_END_LSCH1_INT_ST	LEDC_DUTY_CHNG_END_LSCH0_INT_ST	LEDC_DUTY_CHNG_END_HSCH7_INT_ST	LEDC_DUTY_CHNG_END_HSCH6_INT_ST	LEDC_DUTY_CHNG_END_HSCH5_INT_ST	LEDC_DUTY_CHNG_END_HSCH4_INT_ST	LEDC_DUTY_CHNG_END_HSCH3_INT_ST	LEDC_DUTY_CHNG_END_HSCH2_INT_ST	LEDC_DUTY_CHNG_END_HSCH1_INT_ST	LEDC_DUTY_CHNG_END_HSCH0_INT_ST	LEDC_LSTIMER3_OVF_INT_ST	LEDC_LSTIMER2_OVF_INT_ST	LEDC_LSTIMER1_OVF_INT_ST	LEDC_LSTIMER0_OVF_INT_ST	LEDC_HSTIMER3_OVF_INT_ST	LEDC_HSTIMER2_OVF_INT_ST	LEDC_HSTIMER1_OVF_INT_ST	LEDC_HSTIMER0_OVF_INT_ST	
0 0 0 0 0 0 0 0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	Reset

LEDC_DUTY_CHNG_END_LSCH*n*_INT_ST LEDC_DUTY_CHNG_END_LSCH*n*_INT 中 蔽 中 。( )

LEDC_DUTY_CHNG_END_HSCH*n*_INT_ST LEDC_DUTY_CHNG_END_HSCH*n*_INT 中 蔽 中 。( )

LEDC_LSTIMER*x*_OVF_INT_ST LEDC_LSTIMER*x*_OVF_INT 中 蔽 中 。( )

LEDC_HSTIMER*x*_OVF_INT_ST LEDC_HSTIMER*x*_OVF_INT 中 蔽 中 。( )

Register 14.19. LEDC_CONF_REG (0x0190)

(reserved)	LEDC_APB_CLK_SEL	
31 … 1	0	
0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0	0	Reset

LEDC_APB_CLK_SEL 设 SLOW_CLK 。(/ 写)

0：8 MHz；

1：80 MHz。

15 红外遥控 (RMT)

15.1 概述

RMT（ ）一个 / ， 设 。
RAM（ ）中 脉 ，并 波形。 测 ，并
波， 长 RAM 中。

RMT 8 个 ，编 0 ~ 7， 个 一 。 方便 ， *n* 表 个
。

15.2 功能描述

15.2.1 RMT 架构

图 15-1. RMT 架构

RMT 8 个 ， 个 部 一个 一个 ， 不可 时 。8 个 一
512x32 bit RAM。RAM 可 APB 写。 可 波形。 可
RMT_REF_ALWAYS_ON_CH*n* 个 时 ： 80 MHz APB（ ）时
REF_TICK。

15.2.2 RMT RAM

图 15-2. 数据结构

RAM 中 15-2 ， 16 16 。 个 中 level 表 ， period 表 level 时 （ 15-1 中 clk_div） 期 。period 0 表 标 ， ， 测 变 与 。

RAM 可 APB ， 始 0x3FF56800。 个 可 RAM 按 64x32 bit 8 个 block。 认 个 可 block 1 （ 0 block0， 1 block1， ）。 *n* 可 block 时， 可 RMT_MEM_SIZE_CH*n* 扩 block 可 操 。 RMT_MEM_SIZE_CH*n* >1， 一个 block。 *n* RAM ：

start_addr_ch*n* $= 0x3FF56800 + 64 * 4 * n$, and
end_addr_ch*n* $= 0x3FF56800 + (64 * 4 * n + 64 * 4 *$ RMT_MEM_SIZE_CH*n*$)\mathrm{mod}(512 * 4) - 4$

， 可 RMT_MEM_OWNER_CH*n* 表 *n* RAM 。 不 时， 产 RMT_CH*n*_ERR 中 。

注意： 启 （ RMT_REG_TX_CONTI_MODE） 时， 中 （ 一个 间 一个 ， 一个 一个， ）。 ， N N+1 间 一个 clk_div 期 IDLE 。

15.2.3 时钟

可 RMT_REF_ALWAYS_ON_CH*n* 个 时 ： 80 MHz APB 时 REF_TICK， 参 时 。 8 时 波 。 可 RMT_REF_CNT_RST_CH*n* 。 RMT_DIV_CNT_CH*n* 。

15.2.4 发射器

RMT_TX_START_CH*n* 1 时， *n* 始 RAM 中 。 一 RAM， 可 32 ， 16 ， 16 。

可 block 时， 可 操 。 操 RMT_TX_LIM_CH*n*， RMT_TX_LIM_CH*n* 时， 产 RMT_CH*n*_TX_THR_EVENT_INT 中 ， 可 测 中 。 block 上 时， start_addr_ch*n* 。

中 period 0 时， 并产 RMT_CH*n*_TX_END_INT 中 ， 。 认 标 0 ， 时 可 RMT_IDLE_OUT_EN_CH*n* RMT_IDLE_OUT_LV_CH*n* 。

可 RMT_CARRIER_EN_CH*n* 波 。 波 可

RMT_CARRIER_HIGH_CH*n* RMT_CARRIER_LOW_CH*n* 设 。

15.2.5 接收器

RMT_RX_EN_CH*n* 1 时， 始测 长 并 一 变化时 上 长 RAM。 长时间 测不 变化 RMT_IDLE_THRES_CH*n* 时， ，产 RMT_CH*n*_RX_END_INT 中 并 。

可 RMT_RX_FILTER_EN_CH*n* 波。 波 可 RMT_RX_FILTER_THRES_CH*n* 个 APB 时 期 脉 。

RMT 不 时，可 RMT_MEM_PD RAM 。

15.2.6 中断

- RMT_CH*n*_TX_THR_EVENT_INT： RMT_CH*n*_TX_LIM_REG ， 一 中 。
- RMT_CH*n*_TX_END_INT： 时， 中 。
- RMT_CH*n*_RX_END_INT： 时， 中 。

15.3 寄存器列表

配置寄存器			
RMT_CH0CONF0_REG	Channel 0 config register 0	0x3FF56020	/写
RMT_CH0CONF1_REG	Channel 0 config register 1	0x3FF56024	/写
RMT_CH1CONF0_REG	Channel 1 config register 0	0x3FF56028	/写
RMT_CH1CONF1_REG	Channel 1 config register 1	0x3FF5602C	/写
RMT_CH2CONF0_REG	Channel 2 config register 0	0x3FF56030	/写
RMT_CH2CONF1_REG	Channel 2 config register 1	0x3FF56034	/写
RMT_CH3CONF0_REG	Channel 3 config register 0	0x3FF56038	/写
RMT_CH3CONF1_REG	Channel 3 config register 1	0x3FF5603C	/写
RMT_CH4CONF0_REG	Channel 4 config register 0	0x3FF56040	/写
RMT_CH4CONF1_REG	Channel 4 config register 1	0x3FF56044	/写
RMT_CH5CONF0_REG	Channel 5 config register 0	0x3FF56048	/写
RMT_CH5CONF1_REG	Channel 5 config register 1	0x3FF5604C	/写
RMT_CH6CONF0_REG	Channel 6 config register 0	0x3FF56050	/写
RMT_CH6CONF1_REG	Channel 6 config register 1	0x3FF56054	/写
RMT_CH7CONF0_REG	Channel 7 config register 0	0x3FF56058	/写
RMT_CH7CONF1_REG	Channel 7 config register 1	0x3FF5605C	/写
中断寄存器			
RMT_INT_RAW_REG	始中	0x3FF560A0	
RMT_INT_ST_REG	蔽中	0x3FF560A4	
RMT_INT_ENA_REG	中	0x3FF560A8	/写
RMT_INT_CLR_REG	中	0x3FF560AC	写
载波占空比寄存器			
RMT_CH0CARRIER_DUTY_REG	Channel 0 duty cycle configuration register	0x3FF560B0	/写
RMT_CH1CARRIER_DUTY_REG	Channel 1 duty cycle configuration register	0x3FF560B4	/写

RMT_CH2CARRIER_DUTY_REG	Channel 2 duty cycle configuration register	0x3FF560B8	/写
RMT_CH3CARRIER_DUTY_REG	Channel 3 duty cycle configuration register	0x3FF560BC	/写
RMT_CH4CARRIER_DUTY_REG	Channel 4 duty cycle configuration register	0x3FF560C0	/写
RMT_CH5CARRIER_DUTY_REG	Channel 5 duty cycle configuration register	0x3FF560C4	/写
RMT_CH6CARRIER_DUTY_REG	Channel 6 duty cycle configuration register	0x3FF560C8	/写
RMT_CH7CARRIER_DUTY_REG	Channel 7 duty cycle configuration register	0x3FF560CC	/写
发送事件配置寄存器			
RMT_CH0_TX_LIM_REG	Channel 0 Tx event configuration register	0x3FF560D0	/写
RMT_CH1_TX_LIM_REG	Channel 1 Tx event configuration register	0x3FF560D4	/写
RMT_CH2_TX_LIM_REG	Channel 2 Tx event configuration register	0x3FF560D8	/写
RMT_CH3_TX_LIM_REG	Channel 3 Tx event configuration register	0x3FF560DC	/写
RMT_CH4_TX_LIM_REG	Channel 4 Tx event configuration register	0x3FF560E0	/写
RMT_CH5_TX_LIM_REG	Channel 5 Tx event configuration register	0x3FF560E4	/写
RMT_CH6_TX_LIM_REG	Channel 6 Tx event configuration register	0x3FF560E8	/写
RMT_CH7_TX_LIM_REG	Channel 7 Tx event configuration register	0x3FF560EC	/写
其他寄存器			
RMT_APB_CONF_REG	RMT-wide configuration register	0x3FF560F0	/写

15.4 寄存器

Register 15.1. RMT_CH*n*CONF0_REG (*n*: 0-7) (0x0020+8n*)**

RMT_MEM_PD RMT RAM (RMT_CH0CONF0)。1：RAM ；0：RAM 常 。(/写)

RMT_CARRIER_OUT_LV_CH*n* 波 。1: 波 ；0: 波 。(/写)

RMT_CARRIER_EN_CH*n* *n* 波 。1: 波 ；0: 闭 波 。(/写)

RMT_MEM_SIZE_CH*n* *n* block 。(/写)

RMT_IDLE_THRES_CH*n* 长 间 测 不 变 RMT_IDLE_THRES_CH*n* 时， 。(/写)

RMT_DIV_CNT_CH*n* 设 *n* 。(/写)

Register 15.2. RMT_CH*n*CONF1_REG (*n*: 0-7) (0x0024+8n*)**

(reserved)	RMT_IDLE_OUT_EN_CH*n*	RMT_IDLE_OUT_LV_CH*n*	RMT_REF_ALWAYS_ON_CH*n*	RMT_REF_CNT_RST_CH*n*	RMT_RX_FILTER_THRES_CH*n*	RMT_RX_FILTER_EN_CH*n*	RMT_TX_CONTI_MODE_CH*n*	RMT_MEM_OWNER_CH*n*	(reserved)	RMT_MEM_RD_RST_CH*n*	RMT_MEM_WR_RST_CH*n*	RMT_RX_EN_CH*n*	RMT_TX_START_CH*n*	
31 … 20	19	18	17	16	15 … 8	7	6	5	4	3	2	1	0	
0x0000	0	0	0	0	0x00F	0	0	1	0	0	0	0	0	Reset

RMT_IDLE_OUT_EN_CH*n* *n* 。(/写)

RMT_IDLE_OUT_LV_CH*n* *n* 。(/写)

RMT_REF_ALWAYS_ON_CH*n* 时 。1：clk_apb；0：clk_ref。(/写)

RMT_REF_CNT_RST_CH*n* *n* 时 。(/写)

RMT_RX_FILTER_THRES_CH*n* ， *n* 脉 RMT_RX_FILTER_THRES_CH*n* 个 APB 时 脉 。(/写)

RMT_RX_FILTER_EN_CH*n* *n* 波 。(/写)

RMT_TX_CONTI_MODE_CH*n* 时， 重 ，不 ， 重 。(/写)

RMT_MEM_OWNER_CH*n* 标 *n* RAM 。1: ；0: 。(/写)

RMT_MEM_RD_RST_CH*n* *n* RAM 。(/写)

RMT_MEM_WR_RST_CH*n* *n* RAM 写 。(/写)

RMT_RX_EN_CH*n* *n* 。(/写)

RMT_TX_START_CH*n* *n* 。(/写)

Register 15.3. RMT_INT_RAW_REG (0x00A0)

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0	
RMT_CH7_TX_THR_EVENT_INT_RAW	RMT_CH6_TX_THR_EVENT_INT_RAW	RMT_CH5_TX_THR_EVENT_INT_RAW	RMT_CH4_TX_THR_EVENT_INT_RAW	RMT_CH3_TX_THR_EVENT_INT_RAW	RMT_CH2_TX_THR_EVENT_INT_RAW	RMT_CH1_TX_THR_EVENT_INT_RAW	RMT_CH0_TX_THR_EVENT_INT_RAW	RMT_CH7_ERR_INT_RAW	RMT_CH7_RX_END_INT_RAW	RMT_CH7_TX_END_INT_RAW	RMT_CH6_ERR_INT_RAW	RMT_CH6_RX_END_INT_RAW	RMT_CH6_TX_END_INT_RAW	RMT_CH5_ERR_INT_RAW	RMT_CH5_RX_END_INT_RAW	RMT_CH5_TX_END_INT_RAW	RMT_CH4_ERR_INT_RAW	RMT_CH4_RX_END_INT_RAW	RMT_CH4_TX_END_INT_RAW	RMT_CH3_ERR_INT_RAW	RMT_CH3_RX_END_INT_RAW	RMT_CH3_TX_END_INT_RAW	RMT_CH2_ERR_INT_RAW	RMT_CH2_RX_END_INT_RAW	RMT_CH2_TX_END_INT_RAW	RMT_CH1_ERR_INT_RAW	RMT_CH1_RX_END_INT_RAW	RMT_CH1_TX_END_INT_RAW	RMT_CH0_ERR_INT_RAW	RMT_CH0_RX_END_INT_RAW	RMT_CH0_TX_END_INT_RAW	
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	Reset

RMT_CH*n*_TX_THR_EVENT_INT_RAW RMT_CH*n*_TX_THR_EVENT_INT 中 始中 。( )

RMT_CH*n*_ERR_INT_RAW RMT_CH*n*_ERR_INT 中 始中 。( )

RMT_CH*n*_RX_END_INT_RAW RMT_CH*n*_RX_END_INT 中 始中 。( )

RMT_CH*n*_TX_END_INT_RAW RMT_CH*n*_TX_END_INT 中 始中 。( )

Register 15.4. RMT_INT_ST_REG (0x00A4)

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0	
RMT_CH7_TX_THR_EVENT_INT_ST	RMT_CH6_TX_THR_EVENT_INT_ST	RMT_CH5_TX_THR_EVENT_INT_ST	RMT_CH4_TX_THR_EVENT_INT_ST	RMT_CH3_TX_THR_EVENT_INT_ST	RMT_CH2_TX_THR_EVENT_INT_ST	RMT_CH1_TX_THR_EVENT_INT_ST	RMT_CH0_TX_THR_EVENT_INT_ST	RMT_CH7_ERR_INT_ST	RMT_CH7_RX_END_INT_ST	RMT_CH7_TX_END_INT_ST	RMT_CH6_ERR_INT_ST	RMT_CH6_RX_END_INT_ST	RMT_CH6_TX_END_INT_ST	RMT_CH5_ERR_INT_ST	RMT_CH5_RX_END_INT_ST	RMT_CH5_TX_END_INT_ST	RMT_CH4_ERR_INT_ST	RMT_CH4_RX_END_INT_ST	RMT_CH4_TX_END_INT_ST	RMT_CH3_ERR_INT_ST	RMT_CH3_RX_END_INT_ST	RMT_CH3_TX_END_INT_ST	RMT_CH2_ERR_INT_ST	RMT_CH2_RX_END_INT_ST	RMT_CH2_TX_END_INT_ST	RMT_CH1_ERR_INT_ST	RMT_CH1_RX_END_INT_ST	RMT_CH1_TX_END_INT_ST	RMT_CH0_ERR_INT_ST	RMT_CH0_RX_END_INT_ST	RMT_CH0_TX_END_INT_ST	
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	Reset

RMT_CH*n*_TX_THR_EVENT_INT_ST RMT_CH*n*_TX_THR_EVENT_INT 中 蔽中 。( )

RMT_CH*n*_ERR_INT_ST RMT_CH*n*_ERR_INT 中 蔽中 。( )

RMT_CH*n*_RX_END_INT_ST RMT_CH*n*_RX_END_INT 中 蔽中 。( )

RMT_CH*n*_TX_END_INT_ST RMT_CH*n*_TX_END_INT 中 蔽中 。( )

Register 15.5. RMT_INT_ENA_REG (0x00A8)

RMT_CH7_TX_THR_EVENT_INT_ENA	RMT_CH6_TX_THR_EVENT_INT_ENA	RMT_CH5_TX_THR_EVENT_INT_ENA	RMT_CH4_TX_THR_EVENT_INT_ENA	RMT_CH3_TX_THR_EVENT_INT_ENA	RMT_CH2_TX_THR_EVENT_INT_ENA	RMT_CH1_TX_THR_EVENT_INT_ENA	RMT_CH0_TX_THR_EVENT_INT_ENA	RMT_CH7_ERR_INT_ENA	RMT_CH7_RX_END_INT_ENA	RMT_CH7_TX_END_INT_ENA	RMT_CH6_ERR_INT_ENA	RMT_CH6_RX_END_INT_ENA	RMT_CH6_TX_END_INT_ENA	RMT_CH5_ERR_INT_ENA	RMT_CH5_RX_END_INT_ENA	RMT_CH5_TX_END_INT_ENA	RMT_CH4_ERR_INT_ENA	RMT_CH4_RX_END_INT_ENA	RMT_CH4_TX_END_INT_ENA	RMT_CH3_ERR_INT_ENA	RMT_CH3_RX_END_INT_ENA	RMT_CH3_TX_END_INT_ENA	RMT_CH2_ERR_INT_ENA	RMT_CH2_RX_END_INT_ENA	RMT_CH2_TX_END_INT_ENA	RMT_CH1_ERR_INT_ENA	RMT_CH1_RX_END_INT_ENA	RMT_CH1_TX_END_INT_ENA	RMT_CH0_ERR_INT_ENA	RMT_CH0_RX_END_INT_ENA	RMT_CH0_TX_END_INT_ENA	
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0	
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	Reset

RMT_CH*n*_TX_THR_EVENT_INT_ENA RMT_CH*n*_TX_THR_EVENT_INT 中 。(/写)

RMT_CH*n*_ERR_INT_ENA RMT_CH*n*_ERROR_INT 中 。(/写)

RMT_CH*n*_RX_END_INT_ENA RMT_CH*n*_RX_END_INT 中 。(/写)

RMT_CH*n*_TX_END_INT_ENA RMT_CH*n*_TX_END_INT 中 。(/写)

Register 15.6. RMT_INT_CLR_REG (0x00AC)

RMT_CH7_TX_THR_EVENT_INT_CLR	RMT_CH6_TX_THR_EVENT_INT_CLR	RMT_CH5_TX_THR_EVENT_INT_CLR	RMT_CH4_TX_THR_EVENT_INT_CLR	RMT_CH3_TX_THR_EVENT_INT_CLR	RMT_CH2_TX_THR_EVENT_INT_CLR	RMT_CH1_TX_THR_EVENT_INT_CLR	RMT_CH0_TX_THR_EVENT_INT_CLR	RMT_CH7_ERR_INT_CLR	RMT_CH7_RX_END_INT_CLR	RMT_CH7_TX_END_INT_CLR	RMT_CH6_ERR_INT_CLR	RMT_CH6_RX_END_INT_CLR	RMT_CH6_TX_END_INT_CLR	RMT_CH5_ERR_INT_CLR	RMT_CH5_RX_END_INT_CLR	RMT_CH5_TX_END_INT_CLR	RMT_CH4_ERR_INT_CLR	RMT_CH4_RX_END_INT_CLR	RMT_CH4_TX_END_INT_CLR	RMT_CH3_ERR_INT_CLR	RMT_CH3_RX_END_INT_CLR	RMT_CH3_TX_END_INT_CLR	RMT_CH2_ERR_INT_CLR	RMT_CH2_RX_END_INT_CLR	RMT_CH2_TX_END_INT_CLR	RMT_CH1_ERR_INT_CLR	RMT_CH1_RX_END_INT_CLR	RMT_CH1_TX_END_INT_CLR	RMT_CH0_ERR_INT_CLR	RMT_CH0_RX_END_INT_CLR	RMT_CH0_TX_END_INT_CLR	
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0	
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	Reset

RMT_CH*n*_TX_THR_EVENT_INT_CLR RMT_CH*n*_TX_THR_EVENT_INT 中 。(写)

RMT_CH*n*_ERR_INT_CLR RMT_CH*n*_ERRINT 中 。(写)

RMT_CH*n*_RX_END_INT_CLR RMT_CH*n*_RX_END_INT 中 。(写)

RMT_CH*n*_TX_END_INT_CLR RMT_CH*n*_TX_END_INT 中 。(写)

Register 15.7. RMT_CH*n*CARRIER_DUTY_REG (*n*: 0-7) (0x00B0+4*n)

RMT_CARRIER_HIGH_CH*n* (31:16)	RMT_CARRIER_LOW_CH*n* (15:0)	
0x00040	0x00040	Reset

RMT_CARRIER_HIGH_CH*n* *n* 波 时 期。时 可 REF_TICK APB_CLK。(/写)

RMT_CARRIER_LOW_CH*n* *n* 波 时 期。时 可 REF_TICK APB_CLK。(/写)

Register 15.8. RMT_CH*n*_TX_LIM_REG (*n*: 0-7) (0x00D0+4*n)

(reserved) (31:9)	RMT_TX_LIM_CH*n* (8:0)	
0x000000	0x080	Reset

RMT_TX_LIM_CH*n* block , 产 TX_THR_EVENT 中 。(/写)

Register 15.9. RMT_APB_CONF_REG (0x00F0)

(reserved) (31:2)	RMT_MEM_TX_WRAP_EN (1)	RMT_MEM_ACCESS_EN (0)	
0x00000000	0	0	Reset

RMT_MEM_TX_WRAP_EN 操 。 block , 一 始 。(/写)

RMT_MEM_ACCESS_EN 必 1, RMT 。

16 电机控制脉宽调制器 (PWM)

16.1 概述

脉 时器 (MCPWM) 设 。 个 PWM ，可 中 。常见 一 一 PWM H 方 。

MCPWM 时 ：PWM 时 PWM 操 。 个 PWM 时 时参 ，可 ， 步 时 部 。 个 PWM 操 一个 PWM 波 。MCPWM 设 包 捕 ， 时 部 。

ESP32 个 MCPWM 设， 别 MCPWM0 MCPWM1。 别 0x3FF5E000 0x3FF6C000 始 4 KB 中。 见 16.4 表。

16.2 主要特性

个 MCPWM 设 一个时 （ ），三个 PWM 时 ，三个 PWM 操 一个捕 。 16-1 MCPWM 上 。PWM 时 时参 。PWM 操 时参 波 。 ， 一 PWM 操 可 一 PWM 时 时参 。不 PWM 操 可 PWM 时 时参 产 PWM 。 ，不 PWM 操 可 不 PWM 时 单 PWM 。不 PWM 时 可 步。

图 16-1. MCPWM 外设概览

16-1 中 ：

- PWM 时 0，1 2
  - 个 PWM 时 一个 8 时 。
  - PWM 时 中 16 包括： ， ，

。

- 步可 PWM 时 重 ，重 中； 时 重 ， 而 步 时 时 。 步 可 GPIO PWM 时 sync_out。

- PWM 操 0，1 2
  - 个 PWM 操 个 PWM （PWMxA PWMxB），可 中 。
  - 步 PWM 。
  - 死 上升 可 ，并可 别设 。
  - 可 CPU 中 。
  - 波 时 PWM ， 变 时 常 。
  - 期，时间 ， 方 。
- 故 测
  - 故 时，可 期 一 。
  - 故 可 PWM 。
- 捕
  - 测 （ ， 测 ）。
  - 脉 间 间 时间测 。
  - 脉 期 比测 。
  - 比编 / 。
  - 3 个 捕 ， 备一个 32 时间 。
  - 捕 可 ，边 可 。
  - 捕 时 可 PWM 时 部 步。
  - 3 个捕 上 可 产 中 。

16.3 模块

16.3.1 模块概述

MCPWM 参 。 参 ， PWM 步 ， 参 16.3.2 。

16.3.1.1 预分频器模块

图 16-2. 预分频器模块

参 ：

- CLK_160M PWM 。

16.3.1.2 定时器模块

图 16-3. 定时器模块

参 ：

- 设 PWM 。
- ：
  - ： PWM
  - ： PWM
  - ： PWM
- 步 ， 。

- 步 PWM 时 彼 步。
- 设 PWM 时 步 ，7 1：
  - 3 个 PWM 时 步
  - GPIO 3 个 步 （SYNC0，SYNC1，SYNC2）
  - 步
- PWM 时 步 时 ，4 1：
  - 步
  - PWM 时 0 时
  - PMW 时 时 期
  - 步
- 期 方 。

16.3.1.3 操作器模块

图 16-4. 操作器模块

表 16-1 操 参 。

表 16-1. 操作器模块的配置参数

	参 /
PWM	• 设 PWMxA / PWMxB PWM 比 • 设 时 时间 • 设 时 时采 : – 已变 PWMxA / PWMxB – PWMxA / PWMxB – 不 操 • S/W PWM • PWM 上升 / 边 上 死 • 矿
死	• 侧 侧 间 补死 • 上升 死 • 死 • 死 ，PWM 波 不插 死 • 可 PWMxA PWMxB • 矿
PWM 波	• 波，设 波 • 设 波波 中 一个脉 时间 • 设 个 脉 比 • PWM 波 ，PWM 波 变
故	• PWM 故 • 故 时采 操 : – PWMxA / PWMxB – PWMxA / PWMxB – PWMxA / PWMxB 刃心 故 • PWM 故 : – 一 – 期 • 中 • 故 • 设 期操 矿 • 时 采 时 时 时，可采 不 操

16.3.1.4 故障检测模块

图 16-5. 故障检测模块

参 ：

- 启 故 ，并 个故 故
- 故 中

16.3.1.5 捕获模块

图 16-6. 捕获模块

参 ：

- 捕 边
- 设 捕
- 捕 时 步 步
- 步捕 时

16.3.2 PWM 定时器模块

个 MCPWM 设 三个 PWM 时 。 中 一个 可 三个 PWM 操 中 一个 必 时 。 GPIO 步 ， 步 一个 个 MCPWM 设中 个 PWM 时 一个 。

16.3.2.1 PWM 定时器模块的配置

可 PWM 时 ：

- PWM 时 期 。
- PWM 时 PWM 时 步。

- PWM 时 PWM 时 。
- 设 时 : , , 。
- 已 PWM 时 时 (PT_clk) 。 个 时 , PWM_TIMER0_CFG0_REG PWM_TIMERx_PRESCALE 。PWM 时 设 。

16.3.2.2 PWM 定时器工作模式和定时事件生成

PWM 时 三 , PWMx 时 :

- :
  时 期 中 。一 期 , PWM 时 , 并 始 。PWM 期 期 中 期 + 1。
  : 期 PWM_TIMERx_PERIOD, x = 0, 1, 2, PWM_TIMER0_PERIOD, PWM_TIMER1_PERIOD, PWM_TIMER2_PERIOD。
- :
  PWM 时 期 中 始 。 , 期 , 始 。 , PWM 期 期 中 期 + 1。
- - :
  上 。PWM 时 始 , 期 , 。PWM 时 按 。PWM 期 期 期 × 2 + 1。

16-7 16-10 不 PWM 时 波 , 包 步 期间 时 。

图 16-7. 递增计数模式波形

图 16-8. 递减计数模式波形

图 16-9. 递增递减循环模式波形，同步事件后递减

图 16-10. 递增递减循环模式波形，同步事件后递增

PWM 时 时， 期 时 ：

- UTEP
  PWM 时 期 （PWM_TIMERx_PEROD） PWM 时 时 时 。
- UTEZ
  PWM 时 PWM 时 时 时 。
- DTEP
  PWM 时 期 （PWM_TIMERx_PEROD） PWM 时 时 时 。
- DTEZ
  PWM 时 PWM 时 时 时 。

16-11 16-13 U/DTEP U/DTEZ 时 时 波 。

图 16-11. 递增模式中生成的 UTEP 和 UTEZ

图 16-12. 递减模式中生成的 UTEP 和 UTEZ

16.3.3 PWM 定时囚同怖怖窍边

图 16-13. 递增递减模式中生成的 UTEP 和 UTEZ

16.3.2.3 PWM 定时器影子寄存器

PWM 时 期 PWM 时 时 。 保 个 ， 步 时 写 。 ：

•

- PWM 时 时参 PWM 。
- PWM 个 可 设 死 。
- 可 波 PWM 上。
- 故 。

16-14 PWM 操 。

图 16-14. PWM 操作器的子模块

16.3.3.1 PWM 生成器模块

PWM 生成器模块的作用

中 重 时 ，并 操 ， PWMxA PWMxB 波 。

PWM 操 ：

- A B 时间 时 。 时 时 ：
  - UTEA：PWM 时 并 A。
  - UTEB：PWM 时 并 B。
  - DTEA：PWM 时 并 A。
  - DTEB：PWM 时 并 B。
- 故 步 U/DT1，U/DT2 时 。
- 时 时 时 。
- 时 产 1， 0 操 。
- PWM PWM 比。

- 时间 ， PWM 波 中 脉 。

PWM 操作器影子寄存器

时间 A B， 操 PWM_GENx_A_REG PWM_GENx_B_REG 。
一 步 。 查 16.3.2.3 。

定时事件

表 16-2 时 。

表 16-2. PWM 生成器中的所有定时事件

		PWM 时 操
DTEP	PWM 时 期	PWM 时
DTEZ	PWM 时 0	
DTEA	PWM 时 A	
DTEB	PWM 时 B	
DT0	故 步	
DT1	故 步	
UTEP	PWM 时 期	PWM 时
UTEZ	时 0	
UTEA	PWM 时 A	
UTEB	PWM 时 B	
UT0	故 步	
UT1	故 步	
	步	-

PWMxA PWMxB 上 。 改 步 。
PWM_PWM_GENx_FORCE_REG 。

PWM 中 T0/T1 故 中 故 。故 可 不被
故 中 ， 可 PWM T0/T1 PWM 波
。

， PWM 时 时， TEP ， TEZ 。
， PWM 时 时， DTEP UTEZ， UTEP DTEZ 不 。

PWM 可 时 个 。 ， 见表 16-3 表 16-4。 1（ ）
7（ ）并 。 ，TEP TEZ PWM 时 。

A B 设 期， U/DTEA U/DTEB 不 。

表 16-3. PWM 定时器递增计数时，定时事件的优先级

1（ ）	
2	UTEP
3	UT0

4	UT1
5	UTEB
6	UTEA
7（ ）	UTEZ

表 16-4. PWM 定时器递减计数时，定时事件的优先级

PWMxB 上。 0，1 2 ， 参 16.4 。

常见配置的波形

16-15 PWM 时 时 PWM 波 。 0%-100% 时 可 ：

$$Duty = (Period - A) \div Period$$

A PWM 时 ，并 PWM 时 ， PWM 被上 。 A PWM 时 时 PWM 时 ， PWM 被 。

图 16-15. 递增递减模式下的对称波形

16-16 16-19 PWM 波 常见 PWM 操 。 中 :

- Period，A B 别表 写 期 , A B 。
- PWMxA PWMxB PWM 操 x 。

图 16-16. 递增计数模式，单边不对称波形，PWMxA 和 PWMxB 独立调制–高电平

PWMxA 比时 B 设 , , B 比。
PWMxB 比时 A 设 , , A 比。

$$Period = (PWM_TIMERx_PERIOD + 1) \times T_{PT_clk}$$

图 16-17. 递增计数模式，脉冲位置不对称波形，PWMxA 独立调制

脉 PWM 波 (期 间) 。
可
PWMxA 比 (B A) 比。

$$Period = (PWM_TIMERx_PERIOD + 1) \times T_{PT_clk}$$

图 16-18. 递增递减循环计数模式，双沿对称波形，在 PWMxA 和 PWMxB 上独立调制–高电平有效

PWMxA 比时 A 设 ， ， A 比。
PWMxB 比时 B 设 ， ， B 比。
PWMxA PWMxB 可 不 。

$$Period = (2 \times PWM_TIMERx_PERIOD + 1) \times T_{PT_clk}$$

图 16-19. 递增递减循环计数模式，双沿对称波形，在 PWMxA 和 PWMxB 上独立调制-互补

PWMxA 比时 A 设 ， ， A 比。
PWMxB 比时 B 设 ， ， B 比。
PWMxA/B 可 上/ （ 补） 。
死 = B A，边 可 。必 时，可 死 设 边 。

$$Period = (2 \times PWM_TIMERx_PERIOD + 1) \times T_{PT_clk}$$

软件强制事件

PWM 2 :

- 时 (NCI)
  时， PWM 上 。并 不 ， 一个
  时 已变 PWM 。
- (CNTU)
  一 。 ， PWM 。 可 。 一 可
  时 时 。

16-20 NCI 一 波 。NCI 单 PWMxA ，PWMxB 不
。

图 16-20. NCI 在 PWMxA 输出上软件强制事件示例

16-21 CNTU 波 。UTEZ 被 CNTU 。CNTU 第 PWMxB , PWMxA 不 。

图 16-21. CNTU 在 PWMxB 输出上软件强制事件示例

16.3.3.2 死区生成器模块

死区生成器模块作用

PWM PWMxA PWMxB 上 边 个 。 改变 间 边 设 比，可 死 。 一 个 死 死 。

死 :

- 单个 PWMxA 死 (PWMxA PWMxB)
- 边 死 :
  - 上升 (RED)
  - (FED)
- :
  - 补 (AHC)
  - 补 (ALC)
  - (AH)
  - (AL)
- 死 中 ， 死 不 。

死区模块生成器影子寄存器

PWM_DT RED FED PWM_DT

死区生成器模块的操作要点

16-22 死 。

图 16-22. 死区模块的开关拓扑

上 中 S0 - S8 表 16-5 中 PWM_DTx_CFG_REG 。

表 16-5. 控制死区时间生成器开关的寄存器

S0	PWM_DTx_B_OUTBYPASS
S1	PWM_DTx_A_OUTBYPASS
S2	PWM_DTx_RED_OUTINVERT
S3	PWM_DTx_FED_OUTINVERT
S4	PWM_DTx_RED_INSEL

表 16-6. 死区生成器的典型操作模式

		S0	S1	S2	S3
1	PWMxA PWMxB 波 变	1	1	X	X
2	补 (AHC)，参 16-23	0	0	0	1
3	补 (ALC)，参 16-24	0	0	1	0
4	(AH)，参 16-25	0	0	0	0
5	(AL)，参 16-26	0	0	1	1
6	PWMxA = PWMxA ( ) PWMxB = PWMxA ,	0	1	0 1	0 1
7	PWMxA = PWMxA ，上升 PWMxB = PWMxB ( )	1	0	0 1	0 1
: 上 中，S4 - S8 0。					

图 16-23. 高电平有效互补 (AHC) 死区波形

图 16-24. 低电平有效互补 (ALC) 死区波形

图 16-25. 高电平有效 (AH) 死区波形

图 16-26. 低电平有效 (AL) 死区波形

上升 (RED) (FED) 可 别设 。 16 PWM_DTx_RED PWM_DTx_FED 。 表 一个 边 可 DT_clk 时 期 。DT_clk 可 PWM_DTx_CLK_SEL PWM_clk PT_clk 中 。

(FED) 上升 (RED) :

$$FED = PWM_DTx_FED \times T_{DT_clk}$$

$$RED = PWM_DTx_RED \times T_{DT_clk}$$

16.3.3.3 PWM 载波模块

PWM 可 变 。变 ，新 PWM 比可 0% 100% 间变化 PWM 波 可 波 调 ， 变 。

功能概述

可 :

- 波
- 一个脉 脉
- 个 脉 比
- 启 / 闭 波

操作要点

PWM 波时 (PC_clk) PWM_clk。 PWM_CARRIERx_CFG_REG PWM_CARRIERx_PRESCALE PWM_CARRIERx_DUTY 比。一 脉 脉 。 脉 保 上 。一 脉 可 PWM_CARRIERx_OSHTWTH 。 PWM_CARRIERx_EN / 波 。

载波示例

16-27 波 始 PWM 脉 上 波 。 不 一个脉 比 ， 中 。

图 16-27. PWM 载波操作的波形示例

第一个脉冲

一个脉 可 , 16 可 ，可 :

$$T_{1stpulse} = T_{PWM_clk} \times 8 \times (PWM_CARRIERx_PRESCALE + 1) \times (PWM_CARRIERx_OSHTWTH + 1)$$

中 :

- T_{PMW_clk} PWM 时 期 (PWM_clk)
- $(PWM_CARRIERx_OSHTWTH + 1)$ 一 脉 (: 1-16)
- $(PWM_CARRIERx_PRESCALE + 1)$ PWM 波时 (PC_clk)

16-28 一个脉 脉 。

图 16-28. 载波模块的第一个脉冲和之后持续的脉冲示例

占空比控制

一个一 脉 ， 波 宽调制 PWM 。 可 比。 一 ，宽调制 比可 变 可 启 闭 ，改变 。

比 PWM_CARRIER*x*_CFG_REG PWM_CARRIER*x*_DUTY 5 7 设 ， 7 可 。

比 可 ：

$$Duty = PWM_CARRIER\mathit{x}_DUTY \div 8$$

16-29 7 比设 。

图 16-29. PWM 载波模块中持续脉冲的 7 种占空比设置

16.3.3.4 故障处理器模块

个 MCPWM 设 GPIO 3 个故 (FAULT0，FAULT1 FAULT2)。部故 ，并 可 故 测 故 。故 ， 故 MCPWM 。

故障处理模块功能

故 ：

- 测 故 时 PWMxA PWMxB 一：
  -
  -
  -
  -
- 测 / 时 一 (OST)。
- 期 (CBC) 操 。
- 个故 单 一 期操 。
- 个故 中 。
- 。
- 启 闭 。

操作与配置要点

本 故 操 。

故 GPIO 中采 步。 保 故 脉 采 ， 个脉 时间必 2 个 APB 时 期。故 测 PWM_clk 故 采 ， GPIO 故 脉 时间必 1 个 PWM_clk 期。 ， APB 时 期 PWM_clk 期 ， 上 故 脉 必 个 APB 时 期 一个 PWM_clk 期 。

故 可 故 FAULT0 FAULT2 中 故 fault_event0 fault_event2。 个故 可 单 CBC 操 ，OST 操 操 。

- **逐周期 (CBC) 操作：**

  CBC 操 被 ，PWM*x*A PWM*x*B PWM_FH*x*_A_CBC_U/D PWM_FH*x*_B_CBC_U/D 设 已 变。PWM 时 时，可 不 操 。不 故 可 不 期操 中 。 PWM_FH*x*_CBC_ON 启 闭 CBC 操 。 故 时， 时 间 ， D/UTEP D/UTEZ 时 PWM*x*A/B 上 CBC 操 。 PWM_FH*x*_CBCPULSE PWM*x*A PWM*x*B 常 。 ， ，CBC 操 个 PWM 。

- **一次性（OST）操作：**

  OST 操 被 时，PWM*x*A PWM*x*B PWM_FH*x*_A_OST_U/D PWM_FH*x*_B_OST_U/D 已 变。PWM 时 时，可 不 操 。不 故 可 不 OST 操 中 。 PWM_FH*x*_OST_ON 启 闭 OST 操 。PWM*x*A/B 上 OST 操 故 时不 。一 操 必 PWM_FH*x*_CLR_OST 。

16.3.4 捕获模块

16.3.4.1 介绍

捕 包 3 个 捕 。 CAP0，CAP1 CAP2 GPIO 。 GPIO ，CAP0，CAP1 CAP2 可 一 。 个捕 可 一 ，所 个 可 别设 。 ， 个捕 可 不 。 ，可 为 捕 ，所不 CPU 。

个 ：

- 一个 32 时 （ ），可 PWM 时 ， 一个 步。
- 3 个捕 ， 个 一个 32 时间 一个捕 。
- 捕 边 （上升/ ）可 。
- 捕 （ ：1 256）。
- 三个捕 中 。

16.3.4.2 捕获定时器

捕 时 一个 32 ， 时不 。 APB 时 常 80 MHz。 步 时， ， PWM_CAP_TIMER_PHASE_REG 中。 步 可 PWM 时 步 ，PWM 步 ， 。 捕 时 3 个捕 时参 。

16.3.4.3 捕获通道

必 时， 捕 捕 可 被 ， 。 ， 捕 边 捕 。 捕 时，捕 时 时间 PWM_CAP_CHx_REG 中。捕 中 不 捕 可 不 中 。 捕 边 PWM_CAPx_EDGE 中。捕 可 。

16.4 寄存器列表

表		PWM0	PWM1	
预分频器配置				
PWM_CLK_CFG_REG		0x3FF5E000	0x3FF6C000	/写
PWM 定时器 0 配置与状态				
PWM_TIMER0_CFG0_REG	时 期 广	0x3FF5E004	0x3FF6C004	/写
PWM_TIMER0_CFG1_REG	始 /	0x3FF5E008	0x3FF6C008	/写
PWM_TIMER0_SYNC_REG	步设	0x3FF5E00C	0x3FF6C00C	/写
PWM_TIMER0_STATUS_REG	时	0x3FF5E010	0x3FF6C010	
PWM 定时器 1 配置与状态				
PWM_TIMER1_CFG0_REG	时 广 期	0x3FF5E014	0x3FF6C014	/写
PWM_TIMER1_CFG1_REG	始 /	0x3FF5E018	0x3FF6C018	/写
PWM_TIMER1_SYNC_REG	步设	0x3FF5E01C	0x3FF6C01C	/写
PWM_TIMER1_STATUS_REG	时	0x3FF5E020	0x3FF6C020	
PWM 定时器 2 配置与状态				
PWM_TIMER2_CFG0_REG	时	0x3FF5E024	0x3FF6C024	/写
PWM_TIMER2_CFG1_REG	始 /	0x3FF5E028	0x3FF6C028	/写
PWM_TIMER2_SYNC_REG	步设	0x3FF5E02C	0x3FF6C02C	/写
PWM_TIMER2_STATUS_REG	时	0x3FF5E030	0x3FF6C030	
PWM 定时器常见配置				
PWM_TIMER_SYNCI_CFG_REG	时 步	0x3FF5E034	0x3FF6C034	/写
PWM_OPERATOR_TIMERSEL_REG	PWM 操 时	0x3FF5E038	0x3FF6C038	/写
PWM 操作器 0 配置与状态				
PWM_GEN0_STMP_CFG_REG	时间 A B 广	0x3FF5E03C	0x3FF6C03C	/写
PWM_GEN0_TSTMP_A_REG	A	0x3FF5E040	0x3FF6C040	/写
PWM_GEN0_TSTMP_B_REG	B	0x3FF5E044	0x3FF6C044	/写
PWM_GEN0_CFG0_REG	故 时间 T0 T1	0x3FF5E048	0x3FF6C048	/写
PWM_GEN0_FORCE_REG	PWM0A PWM0B	0x3FF5E04C	0x3FF6C04C	/写
PWM_GEN0_A_REG	PWM0A 上 操	0x3FF5E050	0x3FF6C050	/写
PWM_GEN0_B_REG	PWM0B 上 操	0x3FF5E054	0x3FF6C054	/写
PWM_DT0_CFG_REG	死	0x3FF5E058	0x3FF6C058	/写
PWM_DT0_FED_CFG_REG	FED	0x3FF5E05C	0x3FF6C05C	/写
PWM_DT0_RED_CFG_REG	RED	0x3FF5E060	0x3FF6C060	/写
PWM_CARRIER0_CFG_REG	波	0x3FF5E064	0x3FF6C064	/写
PWM_FH0_CFG0_REG	故 中 PWM0A PWM0B 上 操	0x3FF5E068	0x3FF6C068	/写

表		PWM0	PWM1	
PWM_FH0_CFG1_REG	故	0x3FF5E06C	0x3FF6C06C	/写
PWM_FH0_STATUS_REG	故	0x3FF5E070	0x3FF6C070	
PWM 操作器 1 配置与状态				
PWM_GEN1_STMP_CFG_REG	时间 A B 分	0x3FF5E074	0x3FF6C074	/写
PWM_GEN1_TSTMP_A_REG	A	0x3FF5E078	0x3FF6C078	/写
PWM_GEN1_TSTMP_B_REG	B	0x3FF5E07C	0x3FF6C07C	/写
PWM_GEN1_CFG0_REG	故 T0 T1	0x3FF5E080	0x3FF6C080	/写
PWM_GEN1_FORCE_REG	PWM1A PWM1B	0x3FF5E084	0x3FF6C084	/写
PWM_GEN1_A_REG	PWM1A 上 操	0x3FF5E088	0x3FF6C088	/写
PWM_GEN1_B_REG	PWM1B 上 操	0x3FF5E08C	0x3FF6C08C	/写
PWM_DT1_CFG_REG	死	0x3FF5E090	0x3FF6C090	/写
PWM_DT1_FED_CFG_REG	FED	0x3FF5E094	0x3FF6C094	/写
PWM_DT1_RED_CFG_REG	RED	0x3FF5E098	0x3FF6C098	/写
PWM_CARRIER1_CFG_REG	波	0x3FF5E09C	0x3FF6C09C	/写
PWM_FH1_CFG0_REG	故 中 PWM1A PWM1B 上 操	0x3FF5E0A0	0x3FF6C0A0	/写
PWM_FH1_CFG1_REG	故	0x3FF5E0A4	0x3FF6C0A4	/写
PWM_FH1_STATUS_REG	故	0x3FF5E0A8	0x3FF6C0A8	
PWM 操作器 2 的配置与状态				
PWM_GEN2_STMP_CFG_REG	时间 A B 分	0x3FF5E0AC	0x3FF6C0AC	/写
PWM_GEN2_TSTMP_A_REG	A	0x3FF5E0B0	0x3FF6C0B0	/写
PWM_GEN2_TSTMP_B_REG	B	0x3FF5E0B4	0x3FF6C0B4	/写
PWM_GEN2_CFG0_REG	故 T0 T1	0x3FF5E080	0x3FF6C080	/写
PWM_GEN2_FORCE_REG	PWM2A PWM2B	0x3FF5E0BC	0x3FF6C0BC	/写
PWM_GEN2_A_REG	PWM2A 上 操	0x3FF5E0C0	0x3FF6C0C0	/写
PWM_GEN2_B_REG	PWM2B 上 操	0x3FF5E0C4	0x3FF6C0C4	/写
PWM_DT2_CFG_REG	死	0x3FF5E0C8	0x3FF6C0C8	/写
PWM_DT2_FED_CFG_REG	FED	0x3FF5E0CC	0x3FF6C0CC	/写
PWM_DT2_RED_CFG_REG	RED	0x3FF5E0D0	0x3FF6C0D0	/写
PWM_CARRIER2_CFG_REG	波	0x3FF5E0D4	0x3FF6C0D4	/写
PWM_FH2_CFG0_REG	故 中 PWM2A PWM2B 上 操	0x3FF5E0D8	0x3FF6C0D8	/写
PWM_FH2_CFG1_REG	故	0x3FF5E0DC	0x3FF6C0DC	/写
PWM_FH2_STATUS_REG	故	0x3FF5E0E0	0x3FF6C0E0	
故障检测与配置				
PWM_FAULT_DETECT_REG	故 测	0x3FF5E0E4	0x3FF6C0E4	/写
捕获配置与状态				
PWM_CAP_TIMER_CFG_REG	捕 时	0x3FF5E0E8	0x3FF6C0E8	/写
PWM_CAP_TIMER_PHASE_REG	捕 时 步	0x3FF5E0EC	0x3FF6C0EC	/写

表		PWM0	PWM1	
PWM_CAP_CH0_CFG_REG	捕 0	0x3FF5E0F0	0x3FF6C0F0	/写
PWM_CAP_CH1_CFG_REG	捕 1	0x3FF5E0F4	0x3FF6C0F4	/写
PWM_CAP_CH2_CFG_REG	捕 2	0x3FF5E0F8	0x3FF6C0F8	/写
PWM_CAP_CH0_REG	捕 0 上一 捕	0x3FF5E0FC	0x3FF6C0FC	
PWM_CAP_CH1_REG	捕 1 上一 捕	0x3FF5E100	0x3FF6C100	
PWM_CAP_CH2_REG	捕 2 上一 捕	0x3FF5E104	0x3FF6C104	
PWM_CAP_STATUS_REG	上一 捕 边	0x3FF5E108	0x3FF6C108	
使能有效寄存器的更新				
PWM_UPDATE_CFG_REG		0x3FF5E10C	0x3FF6C10C	/写
管理中断				
INT_ENA_PWM_REG	中	0x3FF5E110	0x3FF6C110	/写
INT_RAW_PWM_REG	始中	0x3FF5E114	0x3FF6C114	
INT_ST_PWM_REG	屏蔽中	0x3FF5E118	0x3FF6C118	
INT_CLR_PWM_REG	中	0x3FF5E11C	0x3FF6C11C	WO

16.5 寄存器

Register 16.1. PWM_CLK_CFG_REG (0x0000)

(reserved)	PWM_CLK_PRESCALE	
31 … 8	7 … 0	
0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0	0x000	Reset

PWM_CLK_PRESCALE PWM_clk 期 = 6.25 ns * (PWM_CLK_PRESCALE + 1)。(/ 写)

Register 16.2. PWM_TIMER0_CFG0_REG (0x0004)

(reserved)	PWM_TIMER0_PERIOD_UPMETHOD	PWM_TIMER0_PERIOD	PWM_TIMER0_PRESCALE	
31 26	25 24	23 8	7 0	
0 0 0 0 0 0	0	0x000FF	0x000	Reset

PWM_TIMER0_PERIOD_UPMETHOD PWM 时 0 期 方 。0: ; 1: TEZ 时 ; 2: 步时 ; 3: TEZ 步时 。本 中，TEZ 时 0时 。(/ 写)

PWM_TIMER0_PERIOD 时 0 期 。(/ 写)

PWM_TIMER0_PRESCALE PT0_clk 期 = PWM_clk 期 * (PWM_TIMER0_PRESCALE + 1)。(/ 写)

Register 16.3. PWM_TIMER0_CFG1_REG (0x0008)

(reserved)	PWM_TIMER0_MOD	PWM_TIMER0_START	
31 5	4 3	2 0	
0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0	0x0	0x0	Reset

PWM_TIMER0_MOD PWM 时 0 。0: ; 1: ; 2: ; 3: 。(/ 写)

PWM_TIMER0_START PWM 时 启 闭。0: 启 ， TEZ 时 ; 1: 启 ， TEP 时 ; 2: 启 ; 3: 启 ，并 一个 TEZ 时 ; 4: 启 ，并 一个 TEP 时 。本 中，TEP 时 期 时 。(/ 写)

Register 16.4. PWM_TIMER0_SYNC_REG (0x000c)

(reserved)	PWM_TIMER0_PHASE_DIRECTION	PWM_TIMER0_PHASE	PWM_TIMER0_SYNCO_SEL	PWM_TIMER0_SYNC_SW	PWM_TIMER0_SYNCI_EN	
31 … 21	20	19 … 4	3 … 2	1	0	
0 0 0 0 0 0 0 0 0 0 0	0	0	0	0	0	Reset

PWM_TIMER0_PHASE_DIRECTION 0: ; 1: 。(/写)

PWM_TIMER0_PHASE 步 中 时 重 。(/ 写)

PWM_TIMER0_SYNCO_SEL PWM 时 0 步 。0: 步; 1: TEZ; 2: TEP; : 步 — 0。(/ 写)

PWM_TIMER0_SYNC_SW , 步。(/ 写)

PWM_TIMER0_SYNCI_EN 1 时, 步 时 时 重 。(/ 写)

Register 16.5. PWM_TIMER0_STATUS_REG (0x0010)

(reserved)	PWM_TIMER0_DIRECTION	PWM_TIMER0_VALUE	
31 … 17	16	15 … 0	
0 0 0 0 0 0 0 0 0 0 0 0 0 0 0	0	0	Reset

PWM_TIMER0_DIRECTION PWM 时 0 。0: ; 1: 。( )

PWM_TIMER0_VALUE PWM 时 0 。( )

Register 16.6. PWM_TIMER1_CFG0_REG (0x0014)

(reserved)	PWM_TIMER1_PERIOD_UPMETHOD	PWM_TIMER1_PERIOD	PWM_TIMER1_PRESCALE	
31 26	25 24	23 8	7 0	
0 0 0 0 0 0	0	0x000FF	0x000	Reset

PWM_TIMER1_PERIOD_UPMETHOD PWM 时 1 期 方 。0: ; 1: TEZ 时 ; 2: 步时 ; 3: TEZ 步时 。(/ 写)

PWM_TIMER1_PERIOD 时 1 期 。(/ 写)

PWM_TIMER1_PRESCALE PT1_clk 期 = PWM_clk 期 * (PWM_TIMER1_PRESCALE + 1)。(/ 写)

Register 16.7. PWM_TIMER1_CFG1_REG (0x0018)

(reserved)	PWM_TIMER1_MOD	PWM_TIMER1_START	
31 5	4 3	2 0	
0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0	0x0	0x0	Reset

PWM_TIMER1_MOD PWM 时 1 。0: ; 1: ; 2: ; 3: 。(/ 写)

PWM_TIMER1_START PWM 时 1 启 。0: 启 , TEZ 时 ; 1: 启 , TEP 时 ; 2: 启 ; 3: 启 并 一 TEZ 中 ; 4: 启 并 一 TEP 中 。(/ 写)

Register 16.8. PWM_TIMER1_SYNC_REG (0x001c)

(reserved)	PWM_TIMER1_PHASE_DIRECTION	PWM_TIMER1_PHASE	PWM_TIMER1_SYNCO_SEL	PWM_TIMER1_SYNC_SW	PWM_TIMER1_SYNCI_EN	
31 … 21	20	19 … 4	3 … 2	1	0	
0 0 0 0 0 0 0 0 0 0 0	0	0	0	0	0	Reset

PWM_TIMER1_PHASE_DIRECTION 0: ; 1: 。(/写)

PWM_TIMER1_PHASE 步时间中 时 重 。(/ 写)

PWM_TIMER1_SYNCO_SEL PWM 时 1 步 。0: 步 ; 1: TEZ; 2: TEP; : 步 — 0。(/ 写)

PWM_TIMER1_SYNC_SW , 步 。(/ 写)

PWM_TIMER1_SYNCI_EN 1 时, 步 时 时 重 。(/ 写)

Register 16.9. PWM_TIMER1_STATUS_REG (0x0020)

(reserved)	PWM_TIMER1_DIRECTION	PWM_TIMER1_VALUE	
31 … 17	16	15 … 0	
0 0 0 0 0 0 0 0 0 0 0 0 0 0 0	0	0	Reset

PWM_TIMER1_DIRECTION PWM 时 1 。0: ; 1: 。( )

PWM_TIMER1_VALUE PWM 时 1 。( )

Register 16.10. PWM_TIMER2_CFG0_REG (0x0024)

(reserved)	PWM_TIMER2_PERIOD_UPMETHOD	PWM_TIMER2_PERIOD	PWM_TIMER2_PRESCALE	
31 26	25 24	23 8	7 0	
0 0 0 0 0 0	0	0x000FF	0x000	Reset

PWM_TIMER2_PERIOD_UPMETHOD PWM 时 2 期 方 。0: ; 1: TEZ 时 ; 2: 步时间时 ; 3: TEZ 步 时 。(/ 写)

PWM_TIMER2_PERIOD PWM 时 2 期 。(/ 写)

PWM_TIMER2_PRESCALE PT2_clk 期 = PWM_clk 期 * (PWM_TIMER2_PRESCALE + 1)。(/ 写)

Register 16.11. PWM_TIMER2_CFG1_REG (0x0028)

(reserved)	PWM_TIMER2_MOD	PWM_TIMER2_START	
31 5	4 3	2 0	
0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0	0x0	0x0	Reset

PWM_TIMER2_MOD PWM 时 2 。0: ; 1: ; 2: ; 3: 。(/ 写)

PWM_TIMER2_START PWM 时 2 启 。0: 启 , TEZ 时 ; 1: 启 , TEP 时 ; 2: 启 ; 3: 启 并 一个 TEZ 时 ; 4: 启 并 一个 TEZ 时 。(/ 写)

Register 16.12. PWM_TIMER2_SYNC_REG (0x002c)

(reserved)										
31										21
0	0	0	0	0	0	0	0	0	0	0

Register 16.14. PWM_TIMER_SYNCI_CFG_REG (0x0034)

(reserved)	PWM_EXTERNAL_SYNCI2_INVERT	PWM_EXTERNAL_SYNCI1_INVERT	PWM_EXTERNAL_SYNCI0_INVERT	PWM_TIMER2_SYNCISEL	PWM_TIMER1_SYNCISEL	PWM_TIMER0_SYNCISEL	
31 … 12	11	10	9	8 … 6	5 … 3	2 … 0	
0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0	0	0	0	0	0	0	Reset

PWM_EXTERNAL_SYNCI2_INVERT GPIO SYNC2 。(/ 写)

PWM_EXTERNAL_SYNCI1_INVERT GPIO SYNC1 。(/ 写)

PWM_EXTERNAL_SYNCI0_INVERT GPIO SYNC0 。(/ 写)

PWM_TIMER2_SYNCISEL PWM 时 2 步 。1: PWM 时 0 步 ; 2: PWM 时 1 步 ; 3: PWM 时 2 步 ; 4: GPIO SYNC0; 5: GPIO SYNC1; 6: GPIO SYNC2; : 步 。(/ 写)

PWM_TIMER1_SYNCISEL PWM 时 1 步 。1: PWM 时 0 步 ; 2: PWM 时 1 步 ; 3: PWM 时 2 步 ; 4: GPIO SYNC0; 5: GPIO SYNC1; 6: GPIO SYNC2; : 步 。(/ 写)

PWM_TIMER0_SYNCISEL PWM 时 0 步 。1: PWM 时 0 步 ; 2: PWM 时 1 步 ; 3: PWM 时 2 步 ; 4: GPIO SYNC0; 5: GPIO SYNC1; 6: GPIO SYNC2; : 步 。(/ 写)

Register 16.15. PWM_OPERATOR_TIMERSEL_REG (0x0038)

(reserved)	PWM_OPERATOR2_TIMERSEL	PWM_OPERATOR1_TIMERSEL	PWM_OPERATOR0_TIMERSEL	
31 … 6	5 … 4	3 … 2	1 … 0	
0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0	0	0	0	Reset

PWM_OPERATOR2_TIMERSEL PWM 操 2 时参 。0: 时 0; 1: 时 1; 2: 时 2。(/ 写)

PWM_OPERATOR1_TIMERSEL PWM 操 1 时参 。0: 时 0; 1: 时 1; 2: 时 2。(/ 写)

PWM_OPERATOR0_TIMERSEL PWM 操 0 时参 。0: 时 0; 1: 时 1; 2: 时 2。(/ 写)

Register 16.16. PWM_GEN0_STMP_CFG_REG (0x003c)

(reserved)	PWM_GEN0_B_SHDW_FULL	PWM_GEN0_A_SHDW_FULL	PWM_GEN0_B_UPMETHOD	PWM_GEN0_A_UPMETHOD	
31 … 10	9	8	7 … 4	3 … 0	
0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0	0	0	0	0	Reset

PWM_GEN0_B_SHDW_FULL 1 。 1 时， PWM 0 时间 B 被写 ，写 B。 时， B 中写 。( )

PWM_GEN0_A_SHDW_FULL 1 。 1 时， PWM 0 时间 A 被写 ，写 A。 时， A 中写 。( )

PWM_GEN0_B_UPMETHOD PWM 0 时间 B 力 。 bit 0: ; bit0 1: TEZ 时 ; bit1 1: TEP 时 ; bit2 1: 步时间 时 ; bit3 1: 闭 。(/ 写)

PWM_GEN0_A_UPMETHOD PWM 0 时间 A 力 。 bit 0: ; bit0 1: TEZ 时 ; bit1 1: TEP 时 ; bit2 1: 步时间 时 ; bit3 1: 闭 。(/ 写)

Register 16.17. PWM_GEN0_TSTMP_A_REG (0x0040)

(reserved)	PWM_GEN0_A	
31 … 16	15 … 0	
0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0	0	Reset

PWM_GEN0_A PWM 0 时间 A 。(/ 写)

Register 16.18. PWM_GEN0_TSTMP_B_REG (0x0044)

(reserved)	PWM_GEN0_B	
31 … 16	15 … 0	
0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0	0	Reset

PWM_GEN0_B PWM 0 时间 B 。(/ 写)

Register 16.19. PWM_GEN0_CFG0_REG (0x0048)

(reserved)	PWM_GEN0_T1_SEL	PWM_GEN0_T0_SEL	PWM_GEN0_CFG_UPMETHOD	
31 … 10	9 … 7	6 … 4	3 … 0	
0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0	0	0	0	Reset

PWM_GEN0_T1_SEL PWM 0 event_t1 , 。0：fault_event0；1：fault_event1；2：fault_event2；3：sync_taken；4： 。(/ 写)

PWM_GEN0_T0_SEL PWM 0 event_t0 , 。0：fault_event0；1：fault_event1；2：fault_event2；3：sync_taken；4： 。(/ 写)

PWM_GEN0_CFG_UPMETHOD PWM 0 。 bit 0： ；bit0 1： TEZ 时 ；bit1 1： TEP 时 ；bit2 1： 步时间时 ；bit3 1： 闭 。(/ 写)

Register 16.20. PWM_GEN0_FORCE_REG (0x004c)

(reserved)	PWM_GEN0_B_NCIFORCE_MODE	PWM_GEN0_B_NCIFORCE	PWM_GEN0_A_NCIFORCE_MODE	PWM_GEN0_A_NCIFORCE	PWM_GEN0_B_CNTUFORCE_MODE	PWM_GEN0_A_CNTUFORCE_MODE	PWM_GEN0_CNTUFORCE_UPMETHOD	
31 … 16	15 … 14	13	12 … 11	10	9 … 8	7 … 6	5 … 0	
0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0	0	0	0	0	0	0	0x20	Reset

PWM_GEN0_B_NCIFORCE_MODE 设 PWM0B 上 时 。0: 闭; 1: ; 2: ; 3: 闭。(/ 写)

PWM_GEN0_B_NCIFORCE 时 PWM0B 上 时 。(/ 写)

PWM_GEN0_A_NCIFORCE_MODE 设 PWM0A 时 。0: 闭; 1: ; 2: ; 3: 闭。(/ 写)

PWM_GEN0_A_NCIFORCE 时 PWM0A 上 时 。(/ 写)

PWM_GEN0_B_CNTUFORCE_MODE PWM0B 。0: 闭; 1: ; 2: ; 3: 闭。(/ 写)

PWM_GEN0_A_CNTUFORCE_MODE 设 PWM0A 。0: 闭; 1: ; 2: ; 3: 闭。(/ 写)

PWM_GEN0_CNTUFORCE_UPMETHOD 0 力 。 bit 0 时: ; bit0 1: TEZ 时 ; bit1 1: TEP 时 ; bit2 1: TEA 时 ; bit3 1: TEB 时 ; bit4: 步 时 ; bit5 1: 闭 。(本 中, TEA/B 时 A/B 时 。) (/ 写)

Register 16.21. PWM_GEN0_A_REG (0x0050)

(reserved)	PWM_GEN0_A_DT1	PWM_GEN0_A_DT0	PWM_GEN0_A_DTEB	PWM_GEN0_A_DTEA	PWM_GEN0_A_DTEP	PWM_GEN0_A_DTEZ	PWM_GEN0_A_UT1	PWM_GEN0_A_UT0	PWM_GEN0_A_UTEB	PWM_GEN0_A_UTEA	PWM_GEN0_A_UTEP	PWM_GEN0_A_UTEZ	
31 24	23 22	21 20	19 18	17 16	15 14	13 12	11 10	9 8	7 6	5 4	3 2	1 0	
0 0 0 0 0 0 0 0	0	0	0	0	0	0	0	0	0	0	0	0	Reset

PWM_GEN0_A_DT1 时 时，event_t1 PWM0A 上 操 。0：波 已 变；1： ；2： ；3： 。(/ 写)

PWM_GEN0_A_DT0 时 时，event_t0 PWM0A 上 操 。(/ 写)

PWM_GEN0_A_DTEB 时 时，TEB PWM0A 上 操 。(/ 写)

PWM_GEN0_A_DTEA 时 时，TEA PWM0A 上 操 。(/ 写)

PWM_GEN0_A_DTEP 时 时，TEP PWM0A 上 操 。(/ 写)

PWM_GEN0_A_DTEZ 时 时，TEZ PWM0A 上 操 。(/ 写)

PWM_GEN0_A_UT1 时 时，event_t1 PWM0A 上 操 。(/ 写)

PWM_GEN0_A_UT0 时 时，event_t0 PWM0A 上 操 。(/ 写)

PWM_GEN0_A_UTEB 时 时，TEB PWM0A 上 操 。(/ 写)

PWM_GEN0_A_UTEA 时 时，TEA PWM0A 上 操 。(/ 写)

PWM_GEN0_A_UTEP 时 时，TEP PWM0A 上 操 。(/ 写)

PWM_GEN0_A_UTEZ 时 时，TEZ PWM0A 上 操 。(/ 写)

Register 16.22. PWM_GEN0_B_REG (0x0054)

(reserved)	PWM_GEN0_B_DT1	PWM_GEN0_B_DT0	PWM_GEN0_B_DTEB	PWM_GEN0_B_DTEA	PWM_GEN0_B_DTEP	PWM_GEN0_B_DTEZ	PWM_GEN0_B_UT1	PWM_GEN0_B_UT0	PWM_GEN0_B_UTEB	PWM_GEN0_B_UTEA	PWM_GEN0_B_UTEP	PWM_GEN0_B_UTEZ	
31 24	23 22	21 20	19 18	17 16	15 14	13 12	11 10	9 8	7 6	5 4	3 2	1 0	
0 0 0 0 0 0 0 0	0	0	0	0	0	0	0	0	0	0	0	0	Reset

PWM_GEN0_B_DT1 时 时，event_t1 PWM0B 上 操 。0：波 改变；1： ；2： ；3： 。(/ 写)

PWM_GEN0_B_DT0 时 时，event_t0 PWM0B 上 操 。(/ 写)

PWM_GEN0_B_DTEB 时 时，TEB PWM0B 上 操 。(/ 写)

PWM_GEN0_B_DTEA 时 时，TEA PWM0B 上 操 。(/ 写)

PWM_GEN0_B_DTEP 时 时，TEP PWM0B 上 操 。(/ 写)

PWM_GEN0_B_DTEZ 时 时，TEZ PWM0B 上 操 。(/ 写)

PWM_GEN0_B_UT1 时 时，event_t1 PWM0B 上 操 。(/ 写)

PWM_GEN0_B_UT0 时 时，event_t0 PWM0B 上 操 。(/ 写)

PWM_GEN0_B_UTEB 时 时，TEB PWM0B 上 操 。(/ 写)

PWM_GEN0_B_UTEA 时 时，TEA PWM0B 上 操 。(/ 写)

PWM_GEN0_B_UTEP 时 时，TEP PWM0B 上 操 。(/ 写)

PWM_GEN0_B_UTEZ 时 时，TEZ PWM0B 上 操 。(/ 写)

Register 16.23. PWM_DT0_CFG_REG (0x0058)

(reserved)	PWM_DT0_CLK_SEL	PWM_DT0_B_OUTBYPASS	PWM_DT0_A_OUTBYPASS	PWM_DT0_FED_OUTINVERT	PWM_DT0_RED_OUTINVERT	PWM_DT0_FED_INSEL	PWM_DT0_RED_INSEL	PWM_DT0_B_OUTSWAP	PWM_DT0_A_OUTSWAP	PWM_DT0_DEB_MODE	PWM_DT0_RED_UPMETHOD	PWM_DT0_FED_UPMETHOD	
31 … 18	17	16	15	14	13	12	11	10	9	8	7 … 4	3 … 0	
0 0 0 0 0 0 0 0 0 0 0 0 0 0	0	1	1	0	0	0	0	0	0	0	0	0	Reset

PWM_DT0_CLK_SEL 死 时间 0 时 。0：PWM_clk；1：PT_clk。(/ 写)

PWM_DT0_B_OUTBYPASS 表 16-5 中 S0。(/ 写)

PWM_DT0_A_OUTBYPASS 表 16-5 中 S1。(/ 写)

PWM_DT0_FED_OUTINVERT 表 16-5 中 S3。(/ 写)

PWM_DT0_RED_OUTINVERT 表 16-5 中 S2。(/ 写)

PWM_DT0_FED_INSEL 表 16-5 中 S5。(/ 写)

PWM_DT0_RED_INSEL 表 16-5 中 S4。(/ 写)

PWM_DT0_B_OUTSWAP 表 16-5 中 S7。(/ 写)

PWM_DT0_A_OUTSWAP S6 表 16-5 中 S6。(/ 写)

PWM_DT0_DEB_MODE S8 表 16-5 中 S8, B 边 。0： / 别 不 在 中 ；1： / 在 上 ；PWMxA 常 。(/ 写)

PWM_DT0_RED_UPMETHOD 上升 方 。 bit 0： ；bit0 1： TEZ 时 ；bit1 1： TEP 时 ；bit2 1： 步 时 间 时 ；bit3 1： 闭 。(/ 写)

PWM_DT0_FED_UPMETHOD 方 。 bit 0： ；bit0 1： TEZ 时 ；bit1 1： TEP 时 ；bit2 1： 步 时 间 时 ；bit3 1： 闭 。(/ 写)

Register 16.24. PWM_DT0_FED_CFG_REG (0x005c)

(reserved)	PWM_DT0_FED	
31 … 16	15 … 0	
0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0	0	Reset

PWM_DT0_FED FED 。(/ 写)

Register 16.25. PWM_DT0_RED_CFG_REG (0x0060)

(reserved)	PWM_DT0_RED	
31 — 16	15 — 0	
0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0	0	Reset

PWM_DT0_RED RED 。(/ 写)

Register 16.26. PWM_CARRIER0_CFG_REG (0x0064)

(reserved)	PWM_CARRIER0_IN_INVERT	PWM_CARRIER0_OUT_INVERT	PWM_CARRIER0_OSHWTH	PWM_CARRIER0_DUTY	PWM_CARRIER0_PRESCALE	PWM_CARRIER0_EN	
31 — 14	13	12	11 — 8	7 — 5	4 — 1	0	
0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0	0	0	0	0	0	0	Reset

PWM_CARRIER0_IN_INVERT 1 时， PWM0A PWM0B 。(/ 写)

PWM_CARRIER0_OUT_INVERT 1 时， PWM0A PWM0B 。(/ 写)

PWM_CARRIER0_OSHWTH 波 一个脉 ，单 波 期。(/ 写)

PWM_CARRIER0_DUTY 波 比。 比 = PWM_CARRIER0_DUTY / 8。(/ 写)

PWM_CARRIER0_PRESCALE PWM 波 0 时 (PC_clk) 。PC_clk 期 = PWM_clk 期 * (PWM_CARRIER0_PRESCALE + 1)。(/ 写)

PWM_CARRIER0_EN 1 时， 波 0 。 时， 波 0 被 。(/ 写)

Register 16.27. PWM_FH0_CFG0_REG (0x0068)

(reserved)	PWM_FH0_B_OST_U	PWM_FH0_B_OST_D	PWM_FH0_B_CBC_U	PWM_FH0_B_CBC_D	PWM_FH0_A_OST_U	PWM_FH0_A_OST_D	PWM_FH0_A_CBC_U	PWM_FH0_A_CBC_D	PWM_FH0_F0_OST	PWM_FH0_F1_OST	PWM_FH0_F2_OST	PWM_FH0_SW_OST	PWM_FH0_F0_CBC	PWM_FH0_F1_CBC	PWM_FH0_F2_CBC	PWM_FH0_SW_CBC	
31 24	23 22	21 20	19 18	17 16	15 14	13 12	11 10	9 8	7	6	5	4	3	2	1	0	
0 0 0 0 0 0 0 0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	Reset

PWM_FH0_B_OST_U 时 并 故 时，PWM0B 上 一 操 。0： ；1： ；2： ；3： 。(/ 写)

PWM_FH0_B_OST_D 时 并 故 时，PWM0B 上 一 操 。0： ；1： ；2： ；3： 。(/ 写)

PWM_FH0_B_CBC_U 时 并 故 时，PWM0B 上 期 操 。0： ；1： ；2： ；3： 。(/ 写)

PWM_FH0_B_CBC_D 时 并 故 时，PWM0B 上 期 操 。0： ；1： ；2： ；3： 。(/ 写)

PWM_FH0_A_OST_U 时 并 故 时，PWM0A 上 一 操 。0： ；1： ；2： ；3： 。(/ 写)

PWM_FH0_A_OST_D 时 并 故 时，PWM0A 上 一 操 。0： ；1： ；2： ；3： 。(/ 写)

PWM_FH0_A_CBC_U 时 并 故 时，PWM0A 上 期 操 。0： ；1： ；2： ；3： 。(/ 写)

PWM_FH0_A_CBC_D 时 并 故 时，PWM0A 上 期 操 。0： ；1： ；2： ；3： 。(/ 写)

PWM_FH0_F0_OST 设 event_f0 一 操 。0： 闭；1： 。(/ 写)

PWM_FH0_F1_OST 设 event_f1 一 操 。0： 闭；1： 。(/ 写)

PWM_FH0_F2_OST 设 event_f2 一 操 。0： 闭；1： 。(/ 写)

PWM_FH0_SW_OST 一 操 。0： 闭；1： 。(/ 写)

PWM_FH0_F0_CBC 设 event_f0 期 操 。0： 闭；1： 。(/ 写)

PWM_FH0_F1_CBC 设 event_f1 期 操 。0： 闭；1： 。(/ 写)

PWM_FH0_F2_CBC 设 event_f2 期 操 。0： 闭；1： 。(/ 写)

PWM_FH0_SW_CBC 期 操。0： 闭；1： 。(/ 写)

Register 16.28. PWM_FH0_CFG1_REG (0x006c)

(reserved)	PWM_FH0_FORCE_OST	PWM_FH0_FORCE_CBC	PWM_FH0_CBCPULSE	PWM_FH0_CLR_OST	
31 … 5	4	3	2 … 1	0	
0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0	0	0	0	0	Reset

PWM_FH0_FORCE_OST ，可 一 操 。(/ 写)

PWM_FH0_FORCE_CBC ，可 期 操 。(/ 写)

PWM_FH0_CBCPULSE 设 期 操 时间 。bit0 1: TEZ 时; bit1 1: TEP 时。(/ 写)

PWM_FH0_CLR_OST , 一 操 。(/ 写)

Register 16.29. PWM_FH0_STATUS_REG (0x0070)

(reserved)	PWM_FH0_OST_ON	PWM_FH0_CBC_ON	
31 … 2	1	0	
0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0	0	0	Reset

PWM_FH0_OST_ON 1 。 1 时, 一 操 。( )

PWM_FH0_CBC_ON 1 。 1 时, 期 操 。( )

Register 16.30. PWM_GEN1_STMP_CFG_REG (0x0074)

(reserved)	PWM_GEN1_B_SHDW_FULL	PWM_GEN1_A_SHDW_FULL	PWM_GEN1_B_UPMETHOD	PWM_GEN1_A_UPMETHOD	
31 … 10	9	8	7 … 4	3 … 0	
0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0	0	0	0	0	Reset

PWM_GEN1_B_SHDW_FULL 1 。 1 时， PWM 1 时间 B 被写 ，写 B。 时， B 中写 。( )

PWM_GEN1_A_SHDW_FULL 1 。 1 时， PWM 1 时间 A 被写 ，写 A。 时， A 中写 。( )

PWM_GEN1_B_UPMETHOD PWM 1 时间 B 方 。 bit 0: ; bit0 1: TEZ 时 ; bit1 1: TEP 时 ; bit2 1: 步时 ; bit3 1: 闭 。(/ 写)

PWM_GEN1_A_UPMETHOD PWM 1 时间 A 方 。 bit 0: ; bit0 1: TEZ 时 ; bit1 1: TEP 时 ; bit2 1: 步时间时 ; bit3 1: 闭 。(/ 写)

Register 16.31. PWM_GEN1_TSTMP_A_REG (0x0078)

(reserved)	PWM_GEN1_A	
31 … 16	15 … 0	
0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0	0	Reset

PWM_GEN1_A PWM 1 时间 A 。(/ 写)

Register 16.32. PWM_GEN1_TSTMP_B_REG (0x007c)

(reserved)	PWM_GEN1_B	
31 … 16	15 … 0	
0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0	0	Reset

PWM_GEN1_B PWM 1 时间 B 。(/ 写)

Register 16.33. PWM_GEN1_CFG0_REG (0x0080)

(reserved)	PWM_GEN1_T1_SEL	PWM_GEN1_T0_SEL	PWM_GEN1_CFG_UPMETHOD	
31 … 10	9 … 7	6 … 4	3 … 0	
0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0	0	0	0	Reset

PWM_GEN1_T1_SEL PWM 1 event_t1 , 。0：fault_event0；1：fault_event1；2：fault_event2；3：sync_taken；4： 。(/ 写)

PWM_GEN1_T0_SEL PWM 1 event_t0 , 。0：fault_event0；1：fault_event1；2：fault_event2；3：sync_taken；4： 。(/ 写)

PWM_GEN1_CFG_UPMETHOD PWM 1 方 。 bit 0： ；bit0 1： TEZ 时 ；bit1 1： TEP 时 ；bit2 1： 步时 间时 ；bit3 1： 闭 。(/ 写)

Register 16.35. PWM_GEN1_A_REG (0x0088)

(reserved)	PWM_GEN1_A_DT1	PWM_GEN1_A_DT0	PWM_GEN1_A_DTEB	PWM_GEN1_A_DTEA	PWM_GEN1_A_DTEP	PWM_GEN1_A_DTEZ	PWM_GEN1_A_UT1	PWM_GEN1_A_UT0	PWM_GEN1_A_UTEB	PWM_GEN1_A_UTEA	PWM_GEN1_A_UTEP	PWM_GEN1_A_UTEZ	
31 24	23 22	21 20	19 18	17 16	15 14	13 12	11 10	9 8	7 6	5 4	3 2	1 0	
0 0 0 0 0 0 0 0	0	0	0	0	0	0	0	0	0	0	0	0	Reset

PWM_GEN1_A_DT1 时 时，event_t1 PWM1A 操 。0： ；1： ；2： ；3： 。(/ 写)

PWM_GEN1_A_DT0 时 时，event_t0 PWM1A 操 。(/ 写)

PWM_GEN1_A_DTEB 时 时，TEB PWM1A 操 。(/ 写)

PWM_GEN1_A_DTEA 时 时，TEA PWM1A 操 。(/ 写)

PWM_GEN1_A_DTEP 时 时，TEP PWM1A 操 。(/ 写)

PWM_GEN1_A_DTEZ 时 时，TEZ PWM1A 操 。(/ 写)

PWM_GEN1_A_UT1 时 时，event_t1 PWM1A 操 。(/ 写)

PWM_GEN1_A_UT0 时 时，event_t0 PWM1A 操 。(/ 写)

PWM_GEN1_A_UTEB 时 时，TEB PWM1A 操 。(/ 写)

PWM_GEN1_A_UTEA 时 时，TEA PWM1A 操 。(/ 写)

PWM_GEN1_A_UTEP 时 时，TEP PWM1A 操 。(/ 写)

PWM_GEN1_A_UTEZ 时 时，TEZ PWM1A 操 。(/ 写)

Register 16.36. PWM_GEN1_B_REG (0x008c)

(reserved)	PWM_GEN1_B_DT1	PWM_GEN1_B_DT0	PWM_GEN1_B_DTEB	PWM_GEN1_B_DTEA	PWM_GEN1_B_DTEP	PWM_GEN1_B_DTEZ	PWM_GEN1_B_UT1	PWM_GEN1_B_UT0	PWM_GEN1_B_UTEB	PWM_GEN1_B_UTEA	PWM_GEN1_B_UTEP	PWM_GEN1_B_UTEZ	
31 24	23 22	21 20	19 18	17 16	15 14	13 12	11 10	9 8	7 6	5 4	3 2	1 0	
0 0 0 0 0 0 0 0	0	0	0	0	0	0	0	0	0	0	0	0	Reset

PWM_GEN1_B_DT1 时 时，event_t1 PWM1B 上 操 。0： ；1： ；2： ；3： 。(/ 写)

PWM_GEN1_B_DT0 时 时，event_t0 PWM1B 上 操 。(/ 写)

PWM_GEN1_B_DTEB 时 时，TEB PWM1B 上 操 。(/ 写)

PWM_GEN1_B_DTEA 时 时，TEA PWM1B 上 操 。(/ 写)

PWM_GEN1_B_DTEP 时 时，TEP PWM1B 上 操 。(/ 写)

PWM_GEN1_B_DTEZ 时 时，TEZ PWM1B 上 操 。(/ 写)

PWM_GEN1_B_UT1 时 时，event_t1 PWM1B 上 操 。(/ 写)

PWM_GEN1_B_UT0 时 时，event_t0 PWM1B 上 操 。(/ 写)

PWM_GEN1_B_UTEB 时 时，TEB PWM1B 上 操 。(/ 写)

PWM_GEN1_B_UTEA 时 时，TEA PWM1B 上 操 。(/ 写)

PWM_GEN1_B_UTEP 时 时，TEP PWM1B 上 操 。(/ 写)

PWM_GEN1_B_UTEZ 时 时，TEZ PWM1B 上 操 。(/ 写)

Register 16.37. PWM_DT1_CFG_REG (0x0090)

(reserved)	PWM_DT1_CLK_SEL	PWM_DT1_B_OUTBYPASS	PWM_DT1_A_OUTBYPASS	PWM_DT1_FED_OUTINVERT	PWM_DT1_RED_OUTINVERT	PWM_DT1_FED_INSEL	PWM_DT1_RED_INSEL	PWM_DT1_B_OUTSWAP	PWM_DT1_A_OUTSWAP	PWM_DT1_DEB_MODE	PWM_DT1_RED_UPMETHOD	PWM_DT1_FED_UPMETHOD	
31 … 18	17	16	15	14	13	12	11	10	9	8	7 … 4	3 … 0	
0 0 0 0 0 0 0 0 0 0 0 0 0 0	0	1	1	0	0	0	0	0	0	0	0	0	Reset

PWM_DT1_CLK_SEL 设 死 时间 时 。0：PWM_clk；1：PT_clk。(/ 写)

PWM_DT1_B_OUTBYPASS 表 16-5 中 S0。(/ 写)

PWM_DT1_A_OUTBYPASS 表 16-5 中 S1。(/ 写)

PWM_DT1_FED_OUTINVERT 表 16-5 中 S3。(/ 写)

PWM_DT1_RED_OUTINVERT 表 16-5 中 S2。(/ 写)

PWM_DT1_FED_INSEL 表 16-5 中 S5。(/ 写)

PWM_DT1_RED_INSEL 表 16-5 中 S4。(/ 写)

PWM_DT1_B_OUTSWAP 表 16-5 中 S7。(/ 写)

PWM_DT1_A_OUTSWAP 表 16-5 中 S6。(/ 写)

PWM_DT1_DEB_MODE 表 16-5 中 S8, B 。0： / 别 不 又 ；1： / 至 上 ；PWMxA 常 。(/ 写)

PWM_DT1_RED_UPMETHOD RED (上升) 方 。0： ；bit0 1： TEZ 时 ；bit1 1： TEP 时 ；bit2 1： 步 时 ；bit3 1： 闭 。(/ 写)

PWM_DT1_FED_UPMETHOD FED ( ) 方 。0： ；bit0 1： TEZ 时 ；bit1 1： TEP 时 ；bit2 1： 步 时 ；bit3 1： 闭 。(/ 写)

Register 16.38. PWM_DT1_FED_CFG_REG (0x0094)

(reserved)	PWM_DT1_FED	
31 … 16	15 … 0	
0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0	0	Reset

PWM_DT1_FED FED 。(/ 写)

Register 16.39. PWM_DT1_RED_CFG_REG (0x0098)

(reserved)	PWM_DT1_RED	
31 — 16	15 — 0	
0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0	0	Reset

PWM_DT1_RED RED 。(/ 写)

Register 16.40. PWM_CARRIER1_CFG_REG (0x009c)

(reserved)	PWM_CARRIER1_IN_INVERT	PWM_CARRIER1_OUT_INVERT	PWM_CARRIER1_OSHWTH	PWM_CARRIER1_DUTY	PWM_CARRIER1_PRESCALE	PWM_CARRIER1_EN	
31 — 14	13	12	11 — 8	7 — 5	4 — 1	0	
0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0	0	0	0	0	0	0	Reset

PWM_CARRIER1_IN_INVERT 1 时， PWM1A PWM1B 。(/ 写)

PWM_CARRIER1_OUT_INVERT 1 时， PWM1A PWM1B 。(/ 写)

PWM_CARRIER1_OSHWTH 波 一个脉 ，单 波 期。(/ 写)

PWM_CARRIER1_DUTY 设 波 比。 比 = PWM_CARRIER1_DUTY/8。(/ 写)

PWM_CARRIER1_PRESCALE PWM 波 1 时 (PC_clk) 。PC_clk 期 = PWM_clk 期 * (PWM_CARRIER1_PRESCALE + 1)。(/ 写)

PWM_CARRIER1_EN 1 时， 波 1 。 时， 波 1。(/ 写)

Register 16.41. PWM_FH1_CFG0_REG (0x00a0)

(reserved)	PWM_FH1_B_OST_U	PWM_FH1_B_OST_D	PWM_FH1_B_CBC_U	PWM_FH1_B_CBC_D	PWM_FH1_A_OST_U	PWM_FH1_A_OST_D	PWM_FH1_A_CBC_U	PWM_FH1_A_CBC_D	PWM_FH1_F0_OST	PWM_FH1_F1_OST	PWM_FH1_F2_OST	PWM_FH1_SW_OST	PWM_FH1_F0_CBC	PWM_FH1_F1_CBC	PWM_FH1_F2_CBC	PWM_FH1_SW_CBC	
31 24	23 22	21 20	19 18	17 16	15 14	13 12	11 10	9 8	7	6	5	4	3	2	1	0	
0 0 0 0 0 0 0 0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	Reset

PWM_FH1_B_OST_U 时 并 故 时，PWM1B 上 一 操 。0: ；1: ；2: ；3: 。(/ 写)

PWM_FH1_B_OST_D 时 并 故 时，PWM1B 上 一 操 。0: ；1: ；2: ；3: 。(/ 写)

PWM_FH1_B_CBC_U 时 并 故 时，PWM1B 上 期 操 。0: ；1: ；2: ；3: 。(/ 写)

PWM_FH1_B_CBC_D 时 并 故 时，PWM1B 上 期 操 。0: ；1: ；2: ；3: 。(/ 写)

PWM_FH1_A_OST_U 时 并 故 时，PWM1A 上 一 操 。0: ；1: ；2: ；3: 。(/ 写)

PWM_FH1_A_OST_D 时 并 故 时，PWM1A 上 期 操 。0: ；1: ；2: ；3: 。(/ 写)

PWM_FH1_A_CBC_U 时 并 故 时，PWM1A 上 期 操 。0: ；1: ；2: ；3: 。(/ 写)

PWM_FH1_A_CBC_D 时 并 故 时，PWM1A 上 期 操 。0: ；1: ；2: ；3: 。(/ 写)

PWM_FH1_F0_OST 设 event_f0 一 操 。0: 闭；1: 。(/ 写)

PWM_FH1_F1_OST 设 event_f1 一 操 。0: 闭；1: 。(/ 写)

PWM_FH1_F2_OST 设 event_f2 一 操 。0: 闭；1: 。(/ 写)

PWM_FH1_SW_OST 一 操 。0: 闭；1: 。(/ 写)

PWM_FH1_F0_CBC 设 event_f0 期 操 。0: 闭；1: 。(/ 写)

PWM_FH1_F1_CBC 设 event_f1 期 操 。0: 闭；1: 。(/ 写)

PWM_FH1_F2_CBC 设 event_f2 期 操 。0: 闭；1: 。(/ 写)

PWM_FH1_SW_CBC 期 操 。0: 闭；1: 。(/ 写)

Register 16.44. PWM_GEN2_STMP_CFG_REG (0x00ac)

(reserved)	PWM_GEN2_B_SHDW_FULL	PWM_GEN2_A_SHDW_FULL	PWM_GEN2_B_UPMETHOD	PWM_GEN2_A_UPMETHOD	
31 … 10	9	8	7 … 4	3 … 0	
0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0	0	0	0	0	Reset

PWM_GEN2_B_SHDW_FULL 1 。 1 时，PWM 2 时间 B 被写 ，写 B。 时， B 中写 。( )

PWM_GEN2_A_SHDW_FULL 1 。 1 时，PWM 2 时间 A 被写 ，写 A。 时， A 中写 。( )

PWM_GEN2_B_UPMETHOD 2 时间 B 方 。 bit 0: ；bit0 1: TEZ 时 ；bit1 1: TEP 时 ；bit2 1: 步 时 ；bit3 1: 闭 。(/ 写)

PWM_GEN2_A_UPMETHOD 2 时间 A 方 。 bit 0: ；bit0 1: TEZ 时 ；bit1 1: TEP 时 ；bit2 1: 步 时 ；bit3 1: 闭 。(/ 写)

Register 16.45. PWM_GEN2_TSTMP_A_REG (0x00b0)

(reserved)	PWM_GEN2_A	
31 … 16	15 … 0	
0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0	0	Reset

PWM_GEN2_A PWM 2 时间 A 。(/ 写)

Register 16.46. PWM_GEN2_TSTMP_B_REG (0x00b4)

(reserved)	PWM_GEN2_B	
31 … 16	15 … 0	
0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0	0	Reset

PWM_GEN2_B PWM 2 时间 B 。(/ 写)

Register 16.47. PWM_GEN2_CFG0_REG (0x00b8)

(reserved)	PWM_GEN2_T1_SEL	PWM_GEN2_T0_SEL	PWM_GEN2_CFG_UPMETHOD	
31 … 10	9 … 7	6 … 4	3 … 0	
0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0	0	0	0	Reset

PWM_GEN2_T1_SEL 设 PWM 操 2 event_t1 , 。0：fault_event0；1：fault_event1；2：fault_event2；3：sync_taken, 4： 。(/ 写)

PWM_GEN2_T0_SEL 设 PWM 操 2 event_t0 , 。0：fault_event0；1：fault_event1；2：fault_event2；3：sync_taken, 4： 。(/ 写)

PWM_GEN2_CFG_UPMETHOD PWM 2 方 。 bit 0： ；bit0 1： TEZ 时 ；bit1 1： TEP 时 ；bit2 1： 步时间时 ；bit3 1： 闭 。(/ 写)

Register 16.48. PWM_GEN2_FORCE_REG (0x00bc)

(reserved)	PWM_GEN2_B_NCIFORCE_MODE	PWM_GEN2_B_NCIFORCE	PWM_GEN2_A_NCIFORCE_MODE	PWM_GEN2_A_NCIFORCE	PWM_GEN2_B_CNTUFORCE_MODE	PWM_GEN2_A_CNTUFORCE_MODE	PWM_GEN2_CNTUFORCE_UPMETHOD	
31 … 16	15 … 14	13	12 … 11	10	9 … 8	7 … 6	5 … 0	
0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0	0	0	0	0	0	0	0x20	Reset

PWM_GEN2_B_NCIFORCE_MODE 设 PWM2B 时 。0: 闭; 1: ; 2: ; 3: 闭。(/ 写)

PWM_GEN2_B_NCIFORCE 时 PWM2B 上 时 。(/ 写)

PWM_GEN2_A_NCIFORCE_MODE PWM2A 时 。0: 闭; 1: ; 2: ; 3: 闭。(/ 写)

PWM_GEN2_A_NCIFORCE 时 PWM2A 上 时 。(/ 写)

PWM_GEN2_B_CNTUFORCE_MODE PWM2B 时 。0: 闭; 1: ; 2: ; 3: 闭。(/ 写)

PWM_GEN2_A_CNTUFORCE_MODE PWM2A 时 。0: 闭; 1: ; 2: ; 3: 闭。(/ 写)

PWM_GEN2_CNTUFORCE_UPMETHOD PWM 2 方 。0: ; bit0 1: TEZ 时 ; bit1 1: TEP 时 ; bit2 1: TEA 时 ; bit3 1: TEB 时 ; bit4 1: 步 时 ; bit5 1: 闭 。(TEA/B 表 时 A/B 时 。) (/ 写)

Register 16.49. PWM_GEN2_A_REG (0x00c0)

(reserved)	PWM_GEN2_A_DT1	PWM_GEN2_A_DT0	PWM_GEN2_A_DTEB	PWM_GEN2_A_DTEA	PWM_GEN2_A_DTEP	PWM_GEN2_A_DTEZ	PWM_GEN2_A_UT1	PWM_GEN2_A_UT0	PWM_GEN2_A_UTEB	PWM_GEN2_A_UTEA	PWM_GEN2_A_UTEP	PWM_GEN2_A_UTEZ	
31 24	23 22	21 20	19 18	17 16	15 14	13 12	11 10	9 8	7 6	5 4	3 2	1 0	
0 0 0 0 0 0 0 0	0	0	0	0	0	0	0	0	0	0	0	0	Reset

PWM_GEN2_A_DT1 时 时，event_t1 PWM2A 上 操 。0：波 改变；1： ；2： ；3： 。(/ 写)

PWM_GEN2_A_DT0 时 时，event_t0 PWM2A 上 操 。(/ 写)

PWM_GEN2_A_DTEB 时 时，TEB PWM2A 上 操 。(/ 写)

PWM_GEN2_A_DTEA 时 时，TEA PWM2A 上 操 。(/ 写)

PWM_GEN2_A_DTEP 时 时，TEP PWM2A 上 操 。(/ 写)

PWM_GEN2_A_DTEZ 时 时，TEZ PWM2A 上 操 。(/ 写)

PWM_GEN2_A_UT1 时 时，event_t1 PWM2A 上 操 。(/ 写)

PWM_GEN2_A_UT0 时 时，event_t0 PWM2A 上 操 。(/ 写)

PWM_GEN2_A_UTEB 时 时，TEB PWM2A 上 操 。(/ 写)

PWM_GEN2_A_UTEA 时 时，TEA PWM2A 上 操 。(/ 写)

PWM_GEN2_A_UTEP 时 时，TEP PWM2A 上 操 。(/ 写)

PWM_GEN2_A_UTEZ 时 时，TEZ PWM2A 上 操 。(/ 写)

Register 16.50. PWM_GEN2_B_REG (0x00c4)

(reserved)	PWM_GEN2_B_DT1	PWM_GEN2_B_DT0	PWM_GEN2_B_DTEB	PWM_GEN2_B_DTEA	PWM_GEN2_B_DTEP	PWM_GEN2_B_DTEZ	PWM_GEN2_B_UT1	PWM_GEN2_B_UT0	PWM_GEN2_B_UTEB	PWM_GEN2_B_UTEA	PWM_GEN2_B_UTEP	PWM_GEN2_B_UTEZ	
31 24	23 22	21 20	19 18	17 16	15 14	13 12	11 10	9 8	7 6	5 4	3 2	1 0	
0 0 0 0 0 0 0 0	0	0	0	0	0	0	0	0	0	0	0	0	Reset

PWM_GEN2_B_DT1 时 时，event_t1 PWM2B 上 操 。0：波 改 变；1： ；2： ；3： 。(/ 写)

PWM_GEN2_B_DT0 时 时，event_t0 PWM2B 上 操 。(/ 写)

PWM_GEN2_B_DTEB 时 时，TEB PWM2B 上 操 。(/ 写)

PWM_GEN2_B_DTEA 时 时，TEA PWM2B 上 操 。(/ 写)

PWM_GEN2_B_DTEP 时 时，TEP PWM2B 上 操 。(/ 写)

PWM_GEN2_B_DTEZ 时 时，TEZ PWM2B 上 操 。(/ 写)

PWM_GEN2_B_UT1 时 时，event_t1 PWM2B 上 操 。(/ 写)

PWM_GEN2_B_UT0 时 时，event_t0 PWM2B 上 操 。(/ 写)

PWM_GEN2_B_UTEB 时 时，TEB PWM2B 上 操 。(/ 写)

PWM_GEN2_B_UTEA 时 时，TEA PWM2B 上 操 。(/ 写)

PWM_GEN2_B_UTEP 时 时，TEP PWM2B 上 操 。(/ 写)

PWM_GEN2_B_UTEZ 时 时，TEZ PWM2B 上 操 。(/ 写)

Register 16.51. PWM_DT2_CFG_REG (0x00c8)

(reserved)	PWM_DT2_CLK_SEL	PWM_DT2_B_OUTBYPASS	PWM_DT2_A_OUTBYPASS	PWM_DT2_FED_OUTINVERT	PWM_DT2_RED_OUTINVERT	PWM_DT2_FED_INSEL	PWM_DT2_RED_INSEL	PWM_DT2_B_OUTSWAP	PWM_DT2_A_OUTSWAP	PWM_DT2_DEB_MODE	PWM_DT2_RED_UPMETHOD	PWM_DT2_FED_UPMETHOD	
31 … 18	17	16	15	14	13	12	11	10	9	8	7 … 4	3 … 0	
0 0 0 0 0 0 0 0 0 0 0 0 0 0	0	1	1	0	0	0	0	0	0	0	0	0	Reset

PWM_DT2_CLK_SEL 设 死 时间 1 时 。0：PWM_clk；1：PT_clk。(/ 写)

PWM_DT2_B_OUTBYPASS 表 16-5 中 S0。(/ 写)

PWM_DT2_A_OUTBYPASS 表 16-5 中 S1。(/ 写)

PWM_DT2_FED_OUTINVERT 表 16-5 中 S3。(/ 写)

PWM_DT2_RED_OUTINVERT 表 16-5 中 S2。(/ 写)

PWM_DT2_FED_INSEL 表 16-5 中 S5。(/ 写)

PWM_DT2_RED_INSEL 表 16-5 中 S4。(/ 写)

PWM_DT2_B_OUTSWAP 表 16-5 中 S7。(/ 写)

PWM_DT2_A_OUTSWAP 表 16-5 中 S6。(/ 写)

PWM_DT2_DEB_MODE 表 16-5 中 S8, B 。0： / 别 不 又 ；1： / 至 上 ；PWMxA 常 。(/ 写)

PWM_DT2_RED_UPMETHOD RED (上升) 方 。0： ；bit0 1： TEZ 时 ；bit1 1： TEP 时 ；bit2 1： 步 时 ；bit3 1： 闭 。(/ 写)

PWM_DT2_FED_UPMETHOD FED ( ) 方 。0： ；bit0 1： TEZ 时 ；bit1 1： TEP 时 ；bit2 1： 步 时 ；bit3 1： 闭 。(/ 写)

Register 16.52. PWM_DT2_FED_CFG_REG (0x00cc)

(reserved)	PWM_DT2_FED	
31 … 16	15 … 0	
0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0	0	Reset

PWM_DT2_FED FED 。(/ 写)

Register 16.53. PWM_DT2_RED_CFG_REG (0x00d0)

(reserved)	PWM_DT2_RED	
31 — 16	15 — 0	
0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0	0	Reset

PWM_DT2_RED RED 。(/ 写)

Register 16.54. PWM_CARRIER2_CFG_REG (0x00d4)

(reserved)	PWM_CARRIER2_IN_INVERT	PWM_CARRIER2_OUT_INVERT	PWM_CARRIER2_OSHWTH	PWM_CARRIER2_DUTY	PWM_CARRIER2_PRESCALE	PWM_CARRIER2_EN	
31 — 14	13	12	11 — 8	7 — 5	4 — 1	0	
0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0	0	0	0	0	0	0	Reset

PWM_CARRIER2_IN_INVERT 1 时, PWM2A PWM2B 。(/ 写)

PWM_CARRIER2_OUT_INVERT 1 时, PWM2A PWM2B 。(/ 写)

PWM_CARRIER2_OSHWTH 波 一个脉 , 单 波 期。(/ 写)

PWM_CARRIER2_DUTY 设 波 比。 比 = PWM_CARRIER2_DUTY / 8。(/ 写)

PWM_CARRIER2_PRESCALE PWM 波 2 时 (PC_clk) 。PC_clk 期 = PWM_clk 期 * (PWM_CARRIER0_PRESCALE + 1)。(/ 写)

PWM_CARRIER2_EN 1 时, 波 2 。 时, 波 2 被 。(/ 写)

Register 16.55. PWM_FH2_CFG0_REG (0x00d8)

(reserved)	PWM_FH2_B_OST_U	PWM_FH2_B_OST_D	PWM_FH2_B_CBC_U	PWM_FH2_B_CBC_D	PWM_FH2_A_OST_U	PWM_FH2_A_OST_D	PWM_FH2_A_CBC_U	PWM_FH2_A_CBC_D	PWM_FH2_F0_OST	PWM_FH2_F1_OST	PWM_FH2_F2_OST	PWM_FH2_SW_OST	PWM_FH2_F0_CBC	PWM_FH2_F1_CBC	PWM_FH2_F2_CBC	PWM_FH2_SW_CBC	
31 … 24	23 22	21 20	19 18	17 16	15 14	13 12	11 10	9 8	7	6	5	4	3	2	1	0	
0 0 0 0 0 0 0 0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	Reset

PWM_FH2_B_OST_U 故 并 时 时，PWM2B 上 — 操 。0: ；1: ；2: ；3: 。(/ 写)

PWM_FH2_B_OST_D 故 并 时 时，PWM2B 上 — 操 。0: ；1: ；2: ；3: 。(/ 写)

PWM_FH2_B_CBC_U 故 并 时 时，PWM2B 上 期 操 。0: ；1: ；2: ；3: 。(/ 写)

PWM_FH2_B_CBC_D 故 并 时 时，PWM2B 上 期 操 。0: ；1: ；2: ；3: 。(/ 写)

PWM_FH2_A_OST_U 故 并 时 时，PWM2A 上 — 操 。0: ；1: ；2: ；3: 。(/ 写)

PWM_FH2_A_OST_D 故 并 时 时，PWM2A 上 — 操 。0: ；1: ；2: ；3: 。(/ 写)

PWM_FH2_A_CBC_U 故 并 时 时，PWM2A 上 期 操 。0: ；1: ；2: ；3: 。(/ 写)

PWM_FH2_A_CBC_D 故 并 时 时，PWM2A 上 期 操 。0: ；1: ；2: ；3: 。(/ 写)

PWM_FH2_F0_OST 设 event_f0 — 操 。0: 闭；1: 。(/ 写)

PWM_FH2_F1_OST 设 event_f1 — 操 。0: 闭；1: 。(/ 写)

PWM_FH2_F2_OST 设 event_f2 — 操 。0: 闭；1: 。(/ 写)

PWM_FH2_SW_OST — 操 。0: 闭；1: 。(/ 写)

PWM_FH2_F0_CBC 设 event_f0 期 操 。0: 闭；1: 。(/ 写)

PWM_FH2_F1_CBC 设 event_f1 期 操 。0: 闭；1: 。(/ 写)

PWM_FH2_F2_CBC 设 event_f2 期 操 。0: 闭；1: 。(/ 写)

PWM_FH2_SW_CBC 期 操 。0: 闭；1: 。(/ 写)

Register 16.56. PWM_FH2_CFG1_REG (0x00dc)

31			(reserved)	PWM_FH2_FORCE_OST	PWM_FH2_FORCE_CBC	PWM_FH2_CBCPULSE
31			5	4	3	2 1
0	0	0	0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0	0	0	0

Register 16.58. PWM_FAULT_DETECT_REG (0x00e4)

31																							9	8	7	6	5	4	3	2	1	0	
(reserved)																								PWM_EVENT_F2	PWM_EVENT_F1	PWM_EVENT_F0	PWM_F2_POLE	PWM_F1_POLE	PWM_F0_POLE	PWM_F2_EN	PWM_F1_EN	PWM_F0_EN	
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	Reset

PWM_EVENT_F2 1 。 1 时， event_f2 。( )

PWM_EVENT_F1 1 。 1 时， event_f1 。( )

PWM_EVENT_F0 1 。 1 时， event_f0 。( )

PWM_F2_POLE 设 GPIO FAULT2 event_f2 时 。0: ; 1: 。(/ 写)

PWM_F1_POLE 设 GPIO FAULT2 event_f1 时 。0: ; 1: 。(/ 写)

PWM_F0_POLE 设 GPIO FAULT2 event_f0 时 。0: ; 1: 。(/ 写)

PWM_F2_EN 1 event_f2 。(/ 写)

PWM_F1_EN 1 event_f1 。(/ 写)

PWM_F0_EN 1 event_f0 。(/ 写)

Register 16.59. PWM_CAP_TIMER_CFG_REG (0x00e8)

31																										6	5	4	2	1	0	
(reserved)																											PWM_CAP_SYNC_SW	PWM_CAP_SYNCI_SEL		PWM_CAP_SYNCI_EN	PWM_CAP_TIMER_EN	
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0		0	0	Reset

PWM_CAP_SYNC_SW 1 时， 步捕 时 ，捕 时 中写 。(写)

PWM_CAP_SYNCI_SEL 捕 步 。0: ; 1: 时 0 步 ; 2: 时 1 步 ; 3: 时 2 步 ; 4: GPIO SYNC0 ; 5: GPIO SYNC1; 6: GPIO SYNC2。(/ 写)

PWM_CAP_SYNCI_EN 1 时， 捕 时 步。(/ 写)

PWM_CAP_TIMER_EN 1 时， 捕 时 APB_clk 。(/ 写)

Register 16.60. PWM_CAP_TIMER_PHASE_REG (0x00ec)

31 … 0	
0	Reset

PWM_CAP_TIMER_PHASE_REG 捕 时 步操 。(/ 号)

Register 16.61. PWM_CAP_CH0_CFG_REG (0x00f0)

(reserved)	PWM_CAP0_SW	PWM_CAP0_IN_INVERT	PWM_CAP0_PRESCALE	PWM_CAP0_MODE
31 … 13	12	11	10 … 3	2 … 1
0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0	0	0	0	0

Register 16.62. PWM_CAP_CH1_CFG_REG (0x00f4)

(reserved)	PWM_CAP1_SW	PWM_CAP1_IN_INVERT	PWM_CAP1_PRESCALE	PWM_CAP1_MODE	PWM_CAP1_EN	
31 … 13	12	11	10 … 3	2 … 1	0	
0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0	0	0	0	0	0	Reset

PWM_CAP1_SW 1 时，1 上 捕 。(写)

PWM_CAP1_IN_INVERT 1 时，GPIO CAP1 被 。(/写)

PWM_CAP1_PRESCALE CAP1 上升 。 = PWM_CAP1_PRESCALE + 1。(/写)

PWM_CAP1_MODE 1 上 捕 。bit0 1: 捕 , bit1 1: 上升 捕 。(/写)

PWM_CAP1_EN 1 时，1 上 捕 。(/写)

Register 16.63. PWM_CAP_CH2_CFG_REG (0x00f8)

(reserved)	PWM_CAP2_SW	PWM_CAP2_IN_INVERT	PWM_CAP2_PRESCALE	PWM_CAP2_MODE	PWM_CAP2_EN	
31 … 13	12	11	10 … 3	2 … 1	0	
0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0	0	0	0	0	0	Reset

PWM_CAP2_SW 1 2 上 捕 。(写)

PWM_CAP2_IN_INVERT 1 时，GPIO CAP2 被 。(/写)

PWM_CAP2_PRESCALE CAP2 上升 。 = PWM_CAP2_PRESCALE + 1。(/写)

PWM_CAP2_MODE 2 上 捕 。bit0: 捕 ; bit1 1: 上升 捕 。(/写)

PWM_CAP2_EN 1 时，2 上 捕 。(/写)

Register 16.64. PWM_CAP_CH0_REG (0x00fc)

31 … 0	
0	Reset

PWM_CAP_CH0_REG 0 上 捕 。( )

Register 16.65. PWM_CAP_CH1_REG (0x0100)

31 … 0	
0	Reset

PWM_CAP_CH1_REG 1 上 捕 。( )

Register 16.66. PWM_CAP_CH2_REG (0x0104)

31 … 0	
0	Reset

PWM_CAP_CH2_REG 2 上 捕 。( )

Register 16.67. PWM_CAP_STATUS_REG (0x0108)

31	(reserved)	3	2	1	0	
	0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0		PWM_CAP2_EDGE	PWM_CAP1_EDGE	PWM_CAP0_EDGE	Reset

PWM_CAP2_EDGE 2 上 捕 边 。0: 上升 ; 1: 。( )

PWM_CAP1_EDGE 1 上 捕 边 。0: 上升 ; 1: 。( )

PWM_CAP0_EDGE 0 上 捕 边 。0: 上升 ; 1: 。( )

Register 16.68. PWM_UPDATE_CFG_REG (0x010c)

(reserved)																								PWM_OP2_FORCE_UP	PWM_OP2_UP_EN	PWM_OP1_FORCE_UP	PWM_OP1_UP_EN	PWM_OP0_FORCE_UP	PWM_OP0_UP_EN	PWM_GLOBAL_FORCE_UP	PWM_GLOBAL_UP_EN	
31																							8	7	6	5	4	3	2	1	0	
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	1	0	1	0	1	Reset

PWM_OP2_FORCE_UP PWM 操 2 。(/ 写)

PWM_OP2_UP_EN PWM_GLOBAL_UP_EN 1 时, PWM 操 2 。(/ 写)

PWM_OP1_FORCE_UP PWM 操 1 。(/ 写)

PWM_OP1_UP_EN PWM_GLOBAL_UP_EN 1 时, PWM 操 1 。(/ 写)

PWM_OP0_FORCE_UP PWM 操 0 。(/ 写)

PWM_OP0_UP_EN PWM_GLOBAL_UP_EN 1 时, PWM 操 0 。(/ 写)

PWM_GLOBAL_FORCE_UP MCPWM 。(/ 写)

PWM_GLOBAL_UP_EN MCPWM 。(/ 写)

Register 16.69. INT_ENA_PWM_REG (0x0110)

31 30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0	
(reserved)	INT_CAP2_INT_ENA	INT_CAP1_INT_ENA	INT_CAP0_INT_ENA	INT_FH2_OST_INT_ENA	INT_FH1_OST_INT_ENA	INT_FH0_OST_INT_ENA	INT_FH2_CBC_INT_ENA	INT_FH1_CBC_INT_ENA	INT_FH0_CBC_INT_ENA	INT_OP2_TEB_INT_ENA	INT_OP1_TEB_INT_ENA	INT_OP0_TEB_INT_ENA	INT_OP2_TEA_INT_ENA	INT_OP1_TEA_INT_ENA	INT_OP0_TEA_INT_ENA	INT_FAULT2_CLR_INT_ENA	INT_FAULT1_CLR_INT_ENA	INT_FAULT0_CLR_INT_ENA	INT_FAULT2_INT_ENA	INT_FAULT1_INT_ENA	INT_FAULT0_INT_ENA	INT_TIMER2_TEP_INT_ENA	INT_TIMER1_TEP_INT_ENA	INT_TIMER0_TEP_INT_ENA	INT_TIMER2_TEZ_INT_ENA	INT_TIMER1_TEZ_INT_ENA	INT_TIMER0_TEZ_INT_ENA	INT_TIMER2_STOP_INT_ENA	INT_TIMER1_STOP_INT_ENA	INT_TIMER0_STOP_INT_ENA	
0 0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	Reset

INT_CAP2_INT_ENA 2 捕 中。(/写)

INT_CAP1_INT_ENA 1 捕 中。(/写)

INT_CAP0_INT_ENA 0 捕 中。(/写)

INT_FH2_OST_INT_ENA PWM2 一 操 中。(/写)

INT_FH1_OST_INT_ENA PWM1 一 操 中。(/写)

INT_FH0_OST_INT_ENA PWM0 一 操 中。(/写)

INT_FH2_CBC_INT_ENA PWM2 期 操 中。(/写)

INT_FH1_CBC_INT_ENA PWM1 期 操 中。(/写)

INT_FH0_CBC_INT_ENA PWM0 期 操 中。(/写)

INT_OP2_TEB_INT_ENA PWM 操 2 TEB 中。(/写)

INT_OP1_TEB_INT_ENA PWM 操 2 TEB 中。(/写)

INT_OP0_TEB_INT_ENA PWM 操 0 TEB 中。(/写)

INT_OP2_TEA_INT_ENA PWM 操 2 TEA 中。(/写)

INT_OP1_TEA_INT_ENA PWM 操 1 TEA 中。(/写)

INT_OP0_TEA_INT_ENA PWM 操 0 TEA 中。(/写)

INT_FAULT2_CLR_INT_ENA event_f2 中。(/写)

INT_FAULT1_CLR_INT_ENA event_f1 中。(/写)

INT_FAULT0_CLR_INT_ENA event_f0 中。(/写)

INT_FAULT2_INT_ENA event_f2 始 时 中。(/写)

INT_FAULT1_INT_ENA event_f1 始 时 中。(/写)

INT_FAULT0_INT_ENA event_f0 始 时 中。(/写)

INT_TIMER2_TEP_INT_ENA PWM 时 2 TEP 中。(/写)

INT_TIMER1_TEP_INT_ENA PWM 时 1 TEP 中。(/写)

INT_TIMER0_TEP_INT_ENA PWM 时 0 TEP 中。(/写)

INT_TIMER2_TEZ_INT_ENA PWM 时 2 TEZ 中。(/写)

INT_TIMER1_TEZ_INT_ENA PWM 时 1 TEZ 中。(/写)

INT_TIMER0_TEZ_INT_ENA PWM 时 0 TEZ 中。(/写)

INT_TIMER2_STOP_INT_ENA 时 2 时 中。(/写)

INT_TIMER1_STOP_INT_ENA 时 1 时 中。(/写)

INT_TIMER0_STOP_INT_ENA 时 0 时 中。(/写)

Register 16.70. INT_RAW_PWM_REG (0x0114)

31 30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0	
(reserved)	INT_CAP2_INT_RAW	INT_CAP1_INT_RAW	INT_CAP0_INT_RAW	INT_FH2_OST_INT_RAW	INT_FH1_OST_INT_RAW	INT_FH0_OST_INT_RAW	INT_FH2_CBC_INT_RAW	INT_FH1_CBC_INT_RAW	INT_FH0_CBC_INT_RAW	INT_OP2_TEB_INT_RAW	INT_OP1_TEB_INT_RAW	INT_OP0_TEB_INT_RAW	INT_OP2_TEA_INT_RAW	INT_OP1_TEA_INT_RAW	INT_OP0_TEA_INT_RAW	INT_FAULT2_CLR_INT_RAW	INT_FAULT1_CLR_INT_RAW	INT_FAULT0_CLR_INT_RAW	INT_FAULT2_INT_RAW	INT_FAULT1_INT_RAW	INT_FAULT0_INT_RAW	INT_TIMER2_TEP_INT_RAW	INT_TIMER1_TEP_INT_RAW	INT_TIMER0_TEP_INT_RAW	INT_TIMER2_TEZ_INT_RAW	INT_TIMER1_TEZ_INT_RAW	INT_TIMER0_TEZ_INT_RAW	INT_TIMER2_STOP_INT_RAW	INT_TIMER1_STOP_INT_RAW	INT_TIMER0_STOP_INT_RAW	
0 0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	Reset

INT_CAP2_INT_RAW 2 捕 中 始 。( )

INT_CAP1_INT_RAW 1 捕 中 始 。( )

INT_CAP0_INT_RAW 0 捕 中 始 。( )

INT_FH2_OST_INT_RAW PWM2 操 中 始 。( )

INT_FH1_OST_INT_RAW PWM1 操 中 始 。( )

INT_FH0_OST_INT_RAW PWM0 操 中 始 。( )

INT_FH2_CBC_INT_RAW PWM2 期操 中 始 。( )

INT_FH1_CBC_INT_RAW PWM1 期操 中 始 。( )

INT_FH0_CBC_INT_RAW PWM0 期操 中 始 。( )

INT_OP2_TEB_INT_RAW PWM 操 2 TEB 中 始 。( )

INT_OP1_TEB_INT_RAW PWM 操 1 TEB 中 始 。( )

INT_OP0_TEB_INT_RAW PWM 操 0 TEB 中 始 。( )

INT_OP2_TEA_INT_RAW PWM 操 2 TEA 中 始 。( )

INT_OP1_TEA_INT_RAW PWM 操 1 TEA 中 始 。( )

INT_OP0_TEA_INT_RAW PWM 操 0 TEA 中 始 。( )

INT_FAULT2_CLR_INT_RAW event_f2 时 中 始 。( )

INT_FAULT1_CLR_INT_RAW event_f1 时 中 始 。( )

INT_FAULT0_CLR_INT_RAW event_f0 时 中 始 。( )

INT_FAULT2_INT_RAW event_f2 始 中 始 。( )

INT_FAULT1_INT_RAW event_f1 始 中 始 。( )

INT_FAULT0_INT_RAW event_f0 始 中 始 。( )

INT_TIMER2_TEP_INT_RAW PWM 时 2 TEP 中 始 。( )

INT_TIMER1_TEP_INT_RAW PWM 时 1 TEP 中 始 。( )

INT_TIMER0_TEP_INT_RAW PWM 时 0 TEP 中 始 。( )

INT_TIMER2_TEZ_INT_RAW PWM 时 2 TEZ 中 始 。( )

INT_TIMER1_TEZ_INT_RAW PWM 时 1 TEP 中 始 。( )

INT_TIMER0_TEZ_INT_RAW PWM 时 0 TEP 中 始 。( )

INT_TIMER2_STOP_INT_RAW 时 2 中 始 。( )

INT_TIMER1_STOP_INT_RAW 时 1 中 始 。( )

INT_TIMER0_STOP_INT_RAW 时 0 中 始 。( )

Register 16.71. INT_ST_PWM_REG (0x0118)

31 30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0	
(reserved)	INT_CAP2_INT_ST	INT_CAP1_INT_ST	INT_CAP0_INT_ST	INT_FH2_OST_INT_ST	INT_FH1_OST_INT_ST	INT_FH0_OST_INT_ST	INT_FH2_CBC_INT_ST	INT_FH1_CBC_INT_ST	INT_FH0_CBC_INT_ST	INT_OP2_TEB_INT_ST	INT_OP1_TEB_INT_ST	INT_OP0_TEB_INT_ST	INT_OP2_TEA_INT_ST	INT_OP1_TEA_INT_ST	INT_OP0_TEA_INT_ST	INT_FAULT2_CLR_INT_ST	INT_FAULT1_CLR_INT_ST	INT_FAULT0_CLR_INT_ST	INT_FAULT2_INT_ST	INT_FAULT1_INT_ST	INT_FAULT0_INT_ST	INT_TIMER2_TEP_INT_ST	INT_TIMER1_TEP_INT_ST	INT_TIMER0_TEP_INT_ST	INT_TIMER2_TEZ_INT_ST	INT_TIMER1_TEZ_INT_ST	INT_TIMER0_TEZ_INT_ST	INT_TIMER2_STOP_INT_ST	INT_TIMER1_STOP_INT_ST	INT_TIMER0_STOP_INT_ST	
0 0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	Reset

INT_CAP2_INT_ST 2 上捕 中 屏蔽 。( )

INT_CAP1_INT_ST 1 上捕 中 屏蔽 。( )

INT_CAP0_INT_ST 0 上捕 中 屏蔽 。( )

INT_FH2_OST_INT_ST PWM2 一 操 中 屏蔽 。( )

INT_FH1_OST_INT_ST PWM1 一 操 中 屏蔽 。( )

INT_FH0_OST_INT_ST PWM0 一 操 中 屏蔽 。( )

INT_FH2_CBC_INT_ST PWM2 期操 中 屏蔽 。( )

INT_FH1_CBC_INT_ST PWM1 期操 中 屏蔽 。( )

INT_FH0_CBC_INT_ST PWM0 期操 中 屏蔽 。( )

INT_OP2_TEB_INT_ST PWM 操 2 TEB 中 屏蔽 。( )

INT_OP1_TEB_INT_ST PWM 操 1 TEB 中 屏蔽 。( )

INT_OP0_TEB_INT_ST PWM 操 0 TEB 中 屏蔽 。( )

INT_OP2_TEA_INT_ST PWM 操 2 TEA 中 屏蔽 。( )

INT_OP1_TEA_INT_ST PWM 操 1 TEA 中 屏蔽 。( )

INT_OP0_TEA_INT_ST PWM 操 0 TEA 中 屏蔽 。( )

INT_FAULT2_CLR_INT_ST event_f2 时 中 屏蔽 。( )

INT_FAULT1_CLR_INT_ST event_f1 时 中 屏蔽 。( )

INT_FAULT0_CLR_INT_ST event_f0 时 中 屏蔽 。( )

INT_FAULT2_INT_ST event_f2 始 时 中 屏蔽 。( )

INT_FAULT1_INT_ST event_f1 始 时 中 屏蔽 。( )

INT_FAULT0_INT_ST event_f0 始 时 中 屏蔽 。( )

INT_TIMER2_TEP_INT_ST PWM 时 2 TEP 中 屏蔽 。( )

INT_TIMER1_TEP_INT_ST PWM 时 1 TEP 中 屏蔽 。( )

INT_TIMER0_TEP_INT_ST PWM 时 0 TEP 中 屏蔽 。( )

INT_TIMER2_TEZ_INT_ST PWM 时 2 TEZ 中 屏蔽 。( )

INT_TIMER1_TEZ_INT_ST PWM 时 1 TEZ 中 屏蔽 。( )

INT_TIMER0_TEZ_INT_ST PWM 时 0 TEZ 中 屏蔽 。( )

INT_TIMER2_STOP_INT_ST 时 2 中 屏蔽 。( )

INT_TIMER1_STOP_INT_ST 时 1 中 屏蔽 。( )

INT_TIMER0_STOP_INT_ST 时 0 中 屏蔽 。( )

Register 16.72. INT_CLR_PWM_REG (0x011c)

31 30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0	
(reserved)	INT_CAP2_INT_CLR	INT_CAP1_INT_CLR	INT_CAP0_INT_CLR	INT_FH2_OST_INT_CLR	INT_FH1_OST_INT_CLR	INT_FH0_OST_INT_CLR	INT_FH2_CBC_INT_CLR	INT_FH1_CBC_INT_CLR	INT_FH0_CBC_INT_CLR	INT_OP2_TEB_INT_CLR	INT_OP1_TEB_INT_CLR	INT_OP0_TEB_INT_CLR	INT_OP2_TEA_INT_CLR	INT_OP1_TEA_INT_CLR	INT_OP0_TEA_INT_CLR	INT_FAULT2_CLR_INT_CLR	INT_FAULT1_CLR_INT_CLR	INT_FAULT0_CLR_INT_CLR	INT_FAULT2_INT_CLR	INT_FAULT1_INT_CLR	INT_FAULT0_INT_CLR	INT_TIMER2_TEP_INT_CLR	INT_TIMER1_TEP_INT_CLR	INT_TIMER0_TEP_INT_CLR	INT_TIMER2_TEZ_INT_CLR	INT_TIMER1_TEZ_INT_CLR	INT_TIMER0_TEZ_INT_CLR	INT_TIMER2_STOP_INT_CLR	INT_TIMER1_STOP_INT_CLR	INT_TIMER0_STOP_INT_CLR	
0 0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	Reset

INT_CAP2_INT_CLR 1 时 2 捕 中 。(写)

INT_CAP1_INT_CLR 1 时 1 捕 中 。(写)

INT_CAP0_INT_CLR 1 时 0 捕 中 。(写)

INT_FH2_OST_INT_CLR 1 时 PWM2 一 操 中 。(写)

INT_FH1_OST_INT_CLR 1 时 PWM1 一 操 中 。(写)

INT_FH0_OST_INT_CLR 1 时 PWM0 一 操 中 。(写)

INT_FH2_CBC_INT_CLR 1 时 PWM2 期操 中 。(写)

INT_FH1_CBC_INT_CLR 1 时 PWM1 期操 中 。(写)

INT_FH0_CBC_INT_CLR 1 时 PWM0 期操 中 。(写)

INT_OP2_TEB_INT_CLR 1 时 PWM 操 2 TEB 中 。(写)

INT_OP1_TEB_INT_CLR 1 时 PWM 操 1 TEB 中 。(写)

INT_OP0_TEB_INT_CLR 1 时 PWM 操 0 TEB 中 。(写)

INT_OP2_TEA_INT_CLR 1 时 PWM 操 2 TEA 中 。(写)

INT_OP1_TEA_INT_CLR 1 时 PWM 操 1 TEA 中 。(写)

INT_OP0_TEA_INT_CLR 1 时 PWM 操 0 TEA 中 。(写)

INT_FAULT2_CLR_INT_CLR 1 时 event_f2 时 中 。(写)

INT_FAULT1_CLR_INT_CLR 1 时 event_f1 时 中 。(写)

INT_FAULT0_CLR_INT_CLR 1 时 event_f0 时 中 。(写)

INT_FAULT2_INT_CLR 1 时 event_f2 始 时 中 。(写)

INT_FAULT1_INT_CLR 1 时 event_f1 始 时 中 。(写)

INT_FAULT0_INT_CLR 1 时 event_f0 始 时 中 。(写)

INT_TIMER2_TEP_INT_CLR 1 时 PWM 时 2 TEP 中 。(写)

INT_TIMER1_TEP_INT_CLR 1 时 PWM 时 1 TEP 中 。(写)

INT_TIMER0_TEP_INT_CLR 1 时 PWM 时 0 TEP 中 。(写)

INT_TIMER2_TEZ_INT_CLR 1 时 PWM 时 2 TEZ 中 。(写)

INT_TIMER1_TEZ_INT_CLR 1 时 PWM 时 1 TEZ 中 。(写)

INT_TIMER0_TEZ_INT_CLR 1 时 PWM 时 0 TEZ 中 。(写)

INT_TIMER2_STOP_INT_CLR 1 时 时 2 中 。(写)

INT_TIMER1_STOP_INT_CLR 1 时 时 1 中 。(写)

INT_TIMER0_STOP_INT_CLR 1 时 时 0 中 。(写)

17 脉冲计数器 (PCNT)

17.1 概述

脉 脉 上升 。 个脉 单 一个 16-bit 个 ， 可 。 个 一个脉 一个 。 可 闭 波 。

脉 8 单 ， ， PULSE_CNT_U*n*。

17.2 功能描述

17.2.1 架构图

图 17-1. PULSE_CNT 单元基本架构图

17-1 脉 本 。 个单 个 ： ch0 ch1。 个 。 个 一个 一个 ， 。上升 中 可 别 、不 不 。 如 ， 可 已上升 ，包括： 、 保 。 本 一个 16-bit 。 可 ， 一 比 比 ，可 产 中 。

17.2.2 计数器通道输入信号

上 ，一个 脉 ： LCTRL_MODE HCTRL_MODE 别 ；POS_MODE NEG_MODE 别 上升 。POS_MODE NEG_MODE 1， ； 2 时， ； 表 保 始 ， 不 ， 不 。 LCTRL_MODE HCTRL_MODE 0，表 不 已 NEG_MODE POS_MODE ； 1 表 （ ， POS_MODE NEG_MODE 1 ， ， ）； 。

表 一 上升 ，包括 / 。 可

表 一 ，x 表 “ ”。

POS_ MODE	LCTRL_ MODE	HCTRL_ MODE	sig l→h when ctrl=0	sig l→h when ctrl=1
1 (inc)	0 (-)	0 (-)	Inc ctr	Inc ctr
2 (dec)	0 (-)	0 (-)	Dec ctr	Dec ctr
0 (-)	x	x	No action	No action
1 (inc)	0 (-)	1 (inv)	Inc ctr	Dec ctr
1 (inc)	1 (inv)	0 (-)	Dec ctr	Inc ctr
2 (dec)	0 (-)	1 (inv)	Dec ctr	Inc ctr
1 (inc)	0 (-)	2 (dis)	Inc ctr	No action
1 (inc)	2 (dis)	0 (-)	No action	Inc ctr

表 (sig h→l) ， NEG_MODE POS_MODE。

个脉 单 4 个 中 一个 波 ，可 声。单 4 个 可 PCNT_FILTER_EN_U*n* 波 。一 波 被 ， 比 REG_FILTER_THRES_U*n* 个时 期 脉 被 ， 被 脉 不 启 。

， 一部 。比 PCNT_CNT_PAUSE_U*n*，可 。 PCNT_PLUS_CNT_RST_U*n* 。

17.2.3 观察点

PULSE_CNT 可 设 5 个 察 ，5 个 察 一个中 ，可 中 启 屏蔽中 。 察 别 ：

- 。 PULSE_CNT PCNT_CNT_H_LIM_U*n* 时， PULSE_CNT。 中 PCNT_CNT_H_LIM_U*n* 设 。
- 。 PULSE_CNT PCNT_CNT_L_LIM_U*n* 时， PULSE_CNT。 中 PCNT_CNT_L_LIM_U*n* 设 。
- 个中间 。 PULSE_CNT PCNT_THR_THRES0_U*n* PCNT_THR_THRES1_U*n* 时，产 thr_event 。
- 。 PULSE_CNT 0 时，产 thr_event 。

17.2.4 举例

17-2 0 ， 0 。

- CNT_CH0_POS_MODE_U*n*=1， sig_ch0_u*n* 上升 。
- PCNT_CH0_NEG_MODE_U*n*=0， sig_ch0_u*n* 升 不 。
- PCNT_CH0_LCTRL_MODE_U*n*=0， ctrl_ch0_u*n* 时， sig_ch0_u*n* 。
- PCNT_CH0_HCTRL_MODE_U*n*=2， ctrl_ch0_u*n* 时， sig_ch0_u*n* 不 。
- PCNT_CNT_H_LIM_U*n*=5， PULSE_CNT PCNT_CNT_H_LIM_U*n*，PULSE_CNT 0。

图 17-2. PULSE_CNT 递增计数图

图 17-3. PULSE_CNT 递减计数图

17-3 0 , 17-3 中 0 17-2 中 别 :

- PCNT_CH0_LCTRL_MODE_U*n*=1, ctrl_ch0_u*n* 时, sig_ch0_u*n* 。
- PCNT_CNT_H_LIM_U*n*= 5, PULSE_CNT PCNT_CNT_H_LIM_U*n*, PULSE_CNT 0。

17.2.5 溢出中断

PCNT_CNT_THR_EVENT_U*n*_IN: 中 5 个中 , 一个 中 , 一个 中 , 个中间 中 一个 中 , 可 中 启 屏蔽中 。

17.3 寄存器列表

配置寄存器			
PCNT_U0_CONF0_REG	Configuration register 0 for unit 0	0x3FF57000	/写
PCNT_U1_CONF0_REG	Configuration register 0 for unit 1	0x3FF5700C	/写
PCNT_U2_CONF0_REG	Configuration register 0 for unit 2	0x3FF57018	/写
PCNT_U3_CONF0_REG	Configuration register 0 for unit 3	0x3FF57024	/写
PCNT_U4_CONF0_REG	Configuration register 0 for unit 4	0x3FF57030	/写
PCNT_U5_CONF0_REG	Configuration register 0 for unit 5	0x3FF5703C	/写
PCNT_U6_CONF0_REG	Configuration register 0 for unit 6	0x3FF57048	/写
PCNT_U7_CONF0_REG	Configuration register 0 for unit 7	0x3FF57054	/写
PCNT_U0_CONF1_REG	Configuration register 1 for unit 0	0x3FF57004	/写
PCNT_U1_CONF1_REG	Configuration register 1 for unit 1	0x3FF57010	/写
PCNT_U2_CONF1_REG	Configuration register 1 for unit 2	0x3FF5701C	/写
PCNT_U3_CONF1_REG	Configuration register 1 for unit 3	0x3FF57028	/写
PCNT_U4_CONF1_REG	Configuration register 1 for unit 4	0x3FF57034	/写

PCNT_U5_CONF1_REG	Configuration register 1 for unit 5	0x3FF57040	/写
PCNT_U6_CONF1_REG	Configuration register 1 for unit 6	0x3FF5704C	/写
PCNT_U7_CONF1_REG	Configuration register 1 for unit 7	0x3FF57058	/写
PCNT_U0_CONF2_REG	Configuration register 2 for unit 0	0x3FF57008	/写
PCNT_U1_CONF2_REG	Configuration register 2 for unit 1	0x3FF57014	/写
PCNT_U2_CONF2_REG	Configuration register 2 for unit 2	0x3FF57020	/写
PCNT_U3_CONF2_REG	Configuration register 2 for unit 3	0x3FF5702C	/写
PCNT_U4_CONF2_REG	Configuration register 2 for unit 4	0x3FF57038	/写
PCNT_U5_CONF2_REG	Configuration register 2 for unit 5	0x3FF57044	/写
PCNT_U6_CONF2_REG	Configuration register 2 for unit 6	0x3FF57050	/写
PCNT_U7_CONF2_REG	Configuration register 2 for unit 7	0x3FF5705C	/写
计数器值			
PCNT_U0_CNT_REG	Counter value for unit 0	0x3FF57060	
PCNT_U1_CNT_REG	Counter value for unit 1	0x3FF57064	
PCNT_U2_CNT_REG	Counter value for unit 2	0x3FF57068	
PCNT_U3_CNT_REG	Counter value for unit 3	0x3FF5706C	
PCNT_U4_CNT_REG	Counter value for unit 4	0x3FF57070	
PCNT_U5_CNT_REG	Counter value for unit 5	0x3FF57074	
PCNT_U6_CNT_REG	Counter value for unit 6	0x3FF57078	
PCNT_U7_CNT_REG	Counter value for unit 7	0x3FF5707C	
控制寄存器			
PCNT_CTRL_REG	Control register for all counters	0x3FF570B0	/写
中断寄存器			
PCNT_INT_RAW_REG	Raw interrupt status	0x3FF57080	
PCNT_INT_ST_REG	Masked interrupt status	0x3FF57084	
PCNT_INT_ENA_REG	Interrupt enable bits	0x3FF57088	/写
PCNT_INT_CLR_REG	Interrupt clear bits	0x3FF5708C	写
状态寄存器			
PCNT_U*n*_STATUS_REG		0x3FF57090	

17.4 寄存器

Register 17.1. PCNT_U*n*_CONF0_REG (*n*: 0-7) (0x0+0x0Cn*)**

PCNT_CH1_LCTRL_MODE_U*n*	PCNT_CH1_HCTRL_MODE_U*n*	PCNT_CH1_POS_MODE_U*n*	PCNT_CH1_NEG_MODE_U*n*	PCNT_CH0_LCTRL_MODE_U*n*	PCNT_CH0_HCTRL_MODE_U*n*	PCNT_CH0_POS_MODE_U*n*	PCNT_CH0_NEG_MODE_U*n*	PCNT_THR_THRES1_EN_U*n*	PCNT_THR_THRES0_EN_U*n*	PCNT_THR_L_LIM_EN_U*n*	PCNT_THR_H_LIM_EN_U*n*	PCNT_THR_ZERO_EN_U*n*	PCNT_FILTER_EN_U*n*	PCNT_FILTER_THRES_U*n*	
31 30	29 28	27 26	25 24	23 22	21 20	19 18	17 16	15	14	13	12	11	10	9 0	
0	0	0	0	0	0	0	0	0	0	1	1	1	1	0x010	Reset

PCNT_CH1_LCTRL_MODE_U*n* 时， 改 变
CH1_POS_MODE/CH1_NEG_MODE 设 。(/写)
0：不 改； 1： (,)； 2, 3： 。

PCNT_CH1_HCTRL_MODE_U*n* 时， 改 变
CH1_POS_MODE/CH1_NEG_MODE 设 。(/写)
0：不 改； 1： (,)； 2, 3： 。

PCNT_CH1_POS_MODE_U*n* 设 1 测 上升 。(/写)
1： ； 2： ； 0, 3: 。

PCNT_CH1_NEG_MODE_U*n* 设 1 测 。(/写)
1： ； 2： ； 0, 3: 。

PCNT_CH0_LCTRL_MODE_U*n* 时，
CH0_POS_MODE/CH0_NEG_MODE 设 改 。(/写)
0：不 改； 1： (,)； 2, 3： 改

PCNT_CH0_HCTRL_MODE_U*n* 时，
CH0_POS_MODE/CH0_NEG_MODE 设 改 。(/写)
0：不 改； 1： (,)； 2, 3： 改

PCNT_CH0_POS_MODE_U*n* 设 1 测 上升 。(/写)
1： ； 2： ； 0, 3: 。

PCNT_CH0_NEG_MODE_U*n* T 0 测 一个 时， 设
。(/写) 1： ； 2： ； 0, 3: 。

PCNT_THR_THRES1_EN_U*n* 单 *n* 1 比 。(/写)

PCNT_THR_THRES0_EN_U*n* 单 *n* 0 比 。(/写)

PCNT_THR_L_LIM_EN_U*n* 单 *n* 1 比 。(/写)

PCNT_THR_H_LIM_EN_U*n* 单 *n* thr_h_lim 比 。(/写)

PCNT_THR_ZERO_EN_U*n* 单 *n* 比 。(/写)

PCNT_FILTER_EN_U*n* 单 *n* 波 。(/写)

PCNT_FILTER_THRES_U*n* APB_CLK 个时 期 波 设 。 波 启 时，
比 APB_CLK 个时 期 脉 被 。(/写)

Register 17.2. PCNT_U*n*_CONF1_REG (*n*: 0-7) (0x4+0x0C***n*)

PCNT_CNT_THRES1_U*n* (31:16)	PCNT_CNT_THRES0_U*n* (15:0)	
0x000	0x000	Reset

PCNT_CNT_THRES1_U*n* 单 *n* 1 。(/写)

PCNT_CNT_THRES0_U*n* 单 *n* 0 。(/写)

Register 17.3. PCNT_U*n*_CONF2_REG (*n*: 0-7) (0x8+0x0C***n*)

PCNT_CNT_L_LIM_U*n* (31:16)	PCNT_CNT_H_LIM_U*n* (15:0)	
0x000	0x000	Reset

PCNT_CNT_L_LIM_U*n* 单 *n* 。(/写)

PCNT_CNT_H_LIM_U*n* 单 *n* 。(/写)

Register 17.4. PCNT_U*n*_CNT_REG (*n*: 0-7) (0x28+0x0C***n*)

(reserved) (31:16)	PCNT_PLUS_CNT_U*n* (15:0)	
0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0	0x00000	Reset

PCNT_PLUS_CNT_U*n* 单 *n* ，可 。( )

Register 17.5. PCNT_INT_RAW_REG (0x0080)

(reserved)	PCNT_CNT_THR_EVENT_U7_INT_RAW	PCNT_CNT_THR_EVENT_U6_INT_RAW	PCNT_CNT_THR_EVENT_U5_INT_RAW	PCNT_CNT_THR_EVENT_U4_INT_RAW	PCNT_CNT_THR_EVENT_U3_INT_RAW	PCNT_CNT_THR_EVENT_U2_INT_RAW	PCNT_CNT_THR_EVENT_U1_INT_RAW	PCNT_CNT_THR_EVENT_U0_INT_RAW	
31 … 8	7	6	5	4	3	2	1	0	
0x0000000	0	0	0	0	0	0	0	0	Reset

PCNT_CNT_THR_EVENT_U*n*_INT_RAW PCNT_CNT_THR_EVENT_U*n*_INT 中 始 中 。( )

Register 17.6. PCNT_INT_ST_REG (0x0084)

(reserved)	PCNT_CNT_THR_EVENT_U7_INT_ST	PCNT_CNT_THR_EVENT_U6_INT_ST	PCNT_CNT_THR_EVENT_U5_INT_ST	PCNT_CNT_THR_EVENT_U4_INT_ST	PCNT_CNT_THR_EVENT_U3_INT_ST	PCNT_CNT_THR_EVENT_U2_INT_ST	PCNT_CNT_THR_EVENT_U1_INT_ST	PCNT_CNT_THR_EVENT_U0_INT_ST	
31 … 8	7	6	5	4	3	2	1	0	
0x0000000	0	0	0	0	0	0	0	0	Reset

PCNT_CNT_THR_EVENT_U*n*_INT_ST PCNT_CNT_THR_EVENT_U*n*_INT 中 屏蔽 中 。( )

Register 17.7. PCNT_INT_ENA_REG (0x0088)

(reserved)	PCNT_CNT_THR_EVENT_U7_INT_ENA	PCNT_CNT_THR_EVENT_U6_INT_ENA	PCNT_CNT_THR_EVENT_U5_INT_ENA	PCNT_CNT_THR_EVENT_U4_INT_ENA	PCNT_CNT_THR_EVENT_U3_INT_ENA	PCNT_CNT_THR_EVENT_U2_INT_ENA	PCNT_CNT_THR_EVENT_U1_INT_ENA	PCNT_CNT_THR_EVENT_U0_INT_ENA	
31 … 8	7	6	5	4	3	2	1	0	
0x0000000	0	0	0	0	0	0	0	0	Reset

PCNT_CNT_THR_EVENT_U*n*_INT_ENA PCNT_CNT_THR_EVENT_U*n*_INT 中 。(/写)

Register 17.8. PCNT_INT_CLR_REG (0x008c)

(reserved)	PCNT_CNT_THR_EVENT_U7_INT_CLR	PCNT_CNT_THR_EVENT_U6_INT_CLR	PCNT_CNT_THR_EVENT_U5_INT_CLR	PCNT_CNT_THR_EVENT_U4_INT_CLR	PCNT_CNT_THR_EVENT_U3_INT_CLR	PCNT_CNT_THR_EVENT_U2_INT_CLR	PCNT_CNT_THR_EVENT_U1_INT_CLR	PCNT_CNT_THR_EVENT_U0_INT_CLR	
31 … 8	7	6	5	4	3	2	1	0	
0x0000000	0	0	0	0	0	0	0	0	Reset

PCNT_CNT_THR_EVENT_U*n*_INT_CLR PCNT_CNT_THR_EVENT_U*n*_INT 中 。(写)

Register 17.9. PCNT_CTRL_REG (0x00b0)

(reserved)	(reserved)	PCNT_CNT_PAUSE_U7	PCNT_PLUS_CNT_RST_U7	PCNT_CNT_PAUSE_U6	PCNT_PLUS_CNT_RST_U6	PCNT_CNT_PAUSE_U5	PCNT_PLUS_CNT_RST_U5	PCNT_CNT_PAUSE_U4	PCNT_PLUS_CNT_RST_U4	PCNT_CNT_PAUSE_U3	PCNT_PLUS_CNT_RST_U3	PCNT_CNT_PAUSE_U2	PCNT_PLUS_CNT_RST_U2	PCNT_CNT_PAUSE_U1	PCNT_PLUS_CNT_RST_U1	PCNT_CNT_PAUSE_U0	PCNT_PLUS_CNT_RST_U0	
31 … 17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0	
0x0000	0	0	1	0	1	0	1	0	1	0	1	0	1	0	1	0	1	Reset

PCNT_CNT_PAUSE_U*n* 单 *n* 。(/写)

PCNT_PLUS_CNT_RST_U*n* 单 *n* 。(/写)

Register 17.10. PCNT_U*n*_STATUS_REG (*n*: 0-7) (0x90+0x0Cn*)**

31																									7	6	5	4	3	2	1	0	
(reserved)																										PCNT_THR_ZERO_LAT_U*n*	PCNT_THR_H_LIM_LAT_U*n*	PCNT_THR_L_LIM_LAT_U*n*	PCNT_THR_THRES0_LAT_U*n*	PCNT_THR_THRES1_LAT_U*n*	PCNT_THR_ZERO_MODE_U*n*		
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	00		Reset

PCNT_THR_ZERO_LAT_U*n* 上一 中 时， 0。( )

PCNT_THR_H_LIM_LAT_U*n* 上一 中 时， 。( )

PCNT_THR_L_LIM_LAT_U*n* 上一 中 时， 。( )

PCNT_THR_THRES0_LAT_U*n* 上一 中 时， PCNT_THR_THRES0_U*n*。( )

PCNT_THR_THRES1_LAT_U*n* 上一 中 时， PCNT_THR_THRES1_U*n*。( )

PCNT_THR_ZERO_MODE_U*n* 记录 。0: +0 ( ); 1: -0; 2: ; 3: 。( )

18 定时器组 (TIMG)

18.1 概述

ESP32 4 个 64-bit 时 。 个 时 包 一个 16-bit 一个 64-bit 可 重 上 / 。

ESP32 时 2 , 2 个。TIMG*n*_T*x* *n* 表 别, *x* 表 时 编 。

时 :

- 16-bit 时 , 2-65536
- 64-bit 时
- 可 上 / 时 :
- 时
- 报 时 重
- 报 / 保 时报
- 时重
- 中 边 中

18.2 功能描述

18.2.1 16-bit 预分频器

个 时 APB 时 (写 APB_CLK, 常 80 MHz) 时 。16-bit APB 时 , 产 时 时 (TB_clk)。TB_clk 一个 期, 时 上 一 一。 TIMG*n*_T*x*_DIVIDER , 必 闭 时 (TIMG*n*_T*x*_DIVIDER)。 时 时 不可 。 可 APB 时 2 65536 。 , TIMG*n*_T*x*_DIVIDER 1 2 时, 时 2; TIMG*n*_T*x*_DIVIDER 0 时, 时 65536。 TIMG*n*_T*x*_DIVIDER , 时 被 。

18.2.2 64-bit 时基计数器

TIMG*n*_T*x*_INCREASE 1 可 64-bit 时 别 上 。 时, 64-bit 时 重 时重 , 设 时 报 。

TIMG*n*_T*x*_EN 1 可 闭 。 , 并 TIMG*n*_T*x*_ EN 重 1 保 不变。 TIMG*n*_T*x* _EN 重 并 变 , 设 TIMG*n*_T*x*_EN 不 。

可 TIMG*n*_T*x*_LOAD_LO TIMG*n*_T*x*_LOAD_HI 重 。重 时,

TIMG*n*_T*x*_LOAD_LO TIMG*n*_T*x*_LOAD_HI 才 被 64-bit 时 。报 (报 时 重) (时重) 重 。 TIMG*n*_T*x*_AUTORELOAD 1 可 报 时 重 。 报 时 重 被 , 时 报 上 。 TIMG*n*_T*x*_LOAD_REG 上 写 可 时重 , 写 时 变。 变 TIMG*n*_T*x*_INCREASE 变 时 方 。

时 。 64-bit，CPU 个 32-bit 。 常
被 TIMG*n*_T*x*LO_REG TIMG*n*_T*x*HI_REG。 TIMG*n*_T*x*UPDATE_REG 上写 可 上 64-bit
时 个 。 ， 可 时间 。 可 时 时
。

18.2.3 报警产生

时 可 报 ，报 重 / 中 。 报 TIMG*n*_T*x*_ALARMLO_REG
TIMG*n*_T*x*_ALARMHI_REG 时 ， 报 。 设 ， 超 报
， 时 （ 上 时 ） （ 时 ） 报 报 。
， 报 上 报 。报 ，报 。

18.2.4 MWDT

个 时 包 一个 时 （ 写 MWDT） 。本 ，
参阅 时 。

18.2.5 中断

- TIMG*n*_T*x*_INT_WDT_INT 中 时 中 超时 产 。
- TIMG*n*_T*x*_INT_T1_INT 中 时 1 上 报 产 。
- TIMG*n*_T*x*_INT_T0_INT 中 时 0 上 报 产 。

18.3 寄存器列表

		TIMG0	TIMG1	
定时器 0 配置和控制寄存器				
TIMG*n*_T0CONFIG_REG	Timer 0 configuration register	0x3FF5F000	0x3FF60000	/写
TIMG*n*_T0LO_REG	Timer 0 current value, low 32 bits	0x3FF5F004	0x3FF60004	
TIMG*n*_T0HI_REG	Timer 0 current value, high 32 bits	0x3FF5F008	0x3FF60008	
TIMG*n*_T0UPDATE_REG	Write to copy current timer value to TIMG*n*_T0_(LO/HI)_REG	0x3FF5F00C	0x3FF6000C	写
TIMG*n*_T0ALARMLO_REG	Timer 0 alarm value, low 32 bits	0x3FF5F010	0x3FF60010	/写
TIMG*n*_T0ALARMHI_REG	Timer 0 alarm value, high 32 bits	0x3FF5F014	0x3FF60014	/写
TIMG*n*_T0LOADLO_REG	Timer 0 reload value, low 32 bits	0x3FF5F018	0x3FF60018	/写
TIMG*n*_T0LOAD_REG	Write to reload timer from TIMG*n*_T0_(LOADLO/LOADHI)_REG	0x3FF5F020	0x3FF60020	写
定时器 1 配置和控制寄存器				
TIMG*n*_T1CONFIG_REG	Timer 1 configuration register	0x3FF5F024	0x3FF60024	/写
TIMG*n*_T1LO_REG	Timer 1 current value, low 32 bits	0x3FF5F028	0x3FF60028	
TIMG*n*_T1HI_REG	Timer 1 current value, high 32 bits	0x3FF5F02C	0x3FF6002C	
TIMG*n*_T1UPDATE_REG	Write to copy current timer value to TIMG*n*_T1_(LO/HI)_REG	0x3FF5F030	0x3FF60030	写
TIMG*n*_T1ALARMLO_REG	Timer 1 alarm value, low 32 bits	0x3FF5F034	0x3FF60034	/写
TIMG*n*_T1ALARMHI_REG	Timer 1 alarm value, high 32 bits	0x3FF5F038	0x3FF60038	/写
TIMG*n*_T1LOADLO_REG	Timer 1 reload value, low 32 bits	0x3FF5F03C	0x3FF6003C	/写
TIMG*n*_T1LOAD_REG	Write to reload timer from TIMG*n*_T1_(LOADLO/LOADHI)_REG	0x3FF5F044	0x3FF60044	写

		TIMG0	TIMG1	
系统看门狗定时器配置和控制寄存器				
TIMG*n*_T*x*_WDTCONFIG0_REG	Watchdog timer configuration register	0x3FF5F048	0x3FF60048	/写
TIMG*n*_T*x*_WDTCONFIG1_REG	Watchdog timer prescaler register	0x3FF5F04C	0x3FF6004C	/写
TIMG*n*_T*x*_WDTCONFIG2_REG	Watchdog timer stage 0 timeout value	0x3FF5F050	0x3FF60050	/写
TIMG*n*_T*x*_WDTCONFIG3_REG	Watchdog timer stage 1 timeout value	0x3FF5F054	0x3FF60054	/写
TIMG*n*_T*x*_WDTCONFIG4_REG	Watchdog timer stage 2 timeout value	0x3FF5F058	0x3FF60058	/写
TIMG*n*_T*x*_WDTCONFIG5_REG	Watchdog timer stage 3 timeout value	0x3FF5F05C	0x3FF6005C	/写
TIMG*n*_T*x*_WDTFEED_REG	Write to feed the watchdog timer	0x3FF5F060	0x3FF60060	写
TIMG*n*_T*x*_WDTWPROTECT_REG	Watchdog write protect register	0x3FF5F064	0x3FF60064	/写
RTC 校准配置寄存器				
TIMG*n*_RTCCALICFG_REG	RTC calibration configuration register	0x3FF5F068	0x3FF60068	不
TIMG*n*_RTCCALICFG1_REG	RTC calibration configuration register 1	0x3FF5F06C	0x3FF6006C	
中断寄存器				
TIMG*n*_T*x*_INT_ENA_REG	Interrupt enable bits	0x3FF5F098	0x3FF60098	/写
TIMG*n*_T*x*_INT_RAW_REG	Raw interrupt status	0x3FF5F09C	0x3FF6009C	
TIMG*n*_T*x*_INT_ST_REG	Masked interrupt status	0x3FF5F0A0	0x3FF600A0	
TIMG*n*_T*x*_INT_CLR_REG	Interrupt clear bits	0x3FF5F0A4	0x3FF600A4	写

18.4 寄存器

Register 18.1. TIMG*n*_T*x*CONFIG_REG (*x*: 0-1) (0x0+0x24x*)**

TIMG*n*_T*x*_EN	TIMG*n*_T*x*_INCREASE	TIMG*n*_T*x*_AUTORELOAD	TIMG*n*_T*x*_DIVIDER	TIMG*n*_T*x*_EDGE_INT_EN	TIMG*n*_T*x*_LEVEL_INT_EN	TIMG*n*_T*x*_ALARM_EN	
31	30	29	28 … 13	12	11	10	
0	1	1	0x00001	0	0	0	Reset

TIMG*n*_T*x*_EN 1 ， 时 *x* 时 。(/写)

TIMG*n*_T*x*_INCREASE 1 ， 时 *x* 时 个时 期 。 ， 时 *x* 时 个时 期 。(/写)

TIMG*n*_T*x*_AUTORELOAD 1 ， 时 *x* 报 时 重 。(/写)

TIMG*n*_T*x*_DIVIDER 时 *x* 时 (T*x*_clk) 。(/写)

TIMG*n*_T*x*_EDGE_INT_EN 1 ，报 产 一个边 中 。(/写)

TIMG*n*_T*x*_LEVEL_INT_EN 1 ，报 产 一个 中 。(/写)

TIMG*n*_T*x*_ALARM_EN 1 ，报 。报 ， 。(/写)

Register 18.2. TIMG*n*_T*x*LO_REG (*x*: 0-1) (0x4+0x24x*)**

31 … 0	
0x000000000	Reset

TIMG*n*_T*x*LO_REG TIMG*n*_T*x*UPDATE_REG 上写 ， 时 *x* 时 32 可 被 。( )

Register 18.3. TIMG*n*_T*x*HI_REG (*x*: 0-1) (0x8+0x24x*)**

31 … 0	
0x000000000	Reset

TIMG*n*_T*x*HI_REG TIMG*n*_T*x*UPDATE_REG 上写 ， 时 *x* 时 32 可 被 。( )

Register 18.4. TIMG*n*_T*x*UPDATE_REG (*x*: 0-1) (0xC+0x24***x*)

31	0	
0x000000000		Reset

TIMG*n*_T*x*UPDATE_REG 写 时 *x* 时 (时 *x* 被 上)。(写)

Register 18.5. TIMG*n*_T*x*ALARMLO_REG (*x*: 0-1） (0x10+0x24***x*)

31	0	
0x000000000		Reset

TIMG*n*_T*x*ALARMLO_REG 时 *x* 时 报 32 。(/写)

Register 18.6. TIMG*n*_T*x*ALARMHI_REG (*x*: 0-1) (0x14+0x24***x*)

31	0	
0x000000000		Reset

TIMG*n*_T*x*ALARMHI_REG 时 *x* 时 报 32 。(/写)

Register 18.7. TIMG*n*_T*x*LOADLO_REG (*x*: 0-1) (0x18+0x24***x*)

31	0	
0x000000000		Reset

TIMG*n*_T*x*LOADLO_REG 时 *x* 时 重 32-bit 。(/写)

Register 18.8. TIMG*n*_T*x*LOADHI_REG (*x*: 0-1) (0x1C+0x24***x*)

31	0	
0x000000000		Reset

TIMG*n*_T*x*LOADHI_REG 时 *x* 时 重 32-bit 。(/写)

Register 18.9. TIMG*n*_T*x*LOAD_REG (*x*: 0-1) (0x20+0x24***x*)

31	0	
0x000000000		Reset

TIMG*n*_T*x*LOAD_REG 写 时 *x* 时 重 。(写)

Register 18.10. TIMGn_Tx_WDTCONFIG0_REG (0x0048)

TIMGn_Tx_WDT_EN	TIMGn_Tx_WDT_STG0	TIMGn_Tx_WDT_STG1	TIMGn_Tx_WDT_STG2	TIMGn_Tx_WDT_STG3	TIMGn_Tx_WDT_EDGE_INT_EN	TIMGn_Tx_WDT_LEVEL_INT_EN	TIMGn_Tx_WDT_CPU_RESET_LENGTH	TIMGn_Tx_WDT_SYS_RESET_LENGTH	TIMGn_Tx_WDT_FLASHBOOT_MOD_EN	
31	30 29	28 27	26 25	24 23	22	21	20 18	17 15	14	
0	0	0	0	0	0	0	0x1	0x1	1	Reset

TIMGn_Tx_WDT_EN 1 , SWDT 。(/写)

TIMGn_TxS_WDT_STG0 0 。0: 闭，1: 中 , 2: CPU, 3: 。(/写)

TIMGn_Tx_WDT_STG1 1 。0: 闭，1: 中 , 2: CPU, 3: 。(/写)

TIMGn_Tx_WDT_STG2 2 。0: 闭，1: 中 , 2: CPU, 3: 。(/写)

TIMGn_Tx_WDT_STG3 3 。0: 闭，1: 中 , 2: CPU, 3: 。(/写)

TIMGn_Tx_WDT_EDGE_INT_EN 1 ， 超 时 x 中 产 时间， 产 边 中 。(/写)

TIMGn_Tx_WDT_LEVEL_INT_EN 1 ， 超 设 中 产 时间， 产 中 。(/写)

TIMGn_Tx_WDT_CPU_RESET_LENGTH CPU 长 。0: 100 ns, 1: 200 ns, 2: 300 ns, 3: 400 ns, 4: 500 ns, 5: 800 ns, 6: 1.6 μs, 7: 3.2 μs。(/写)

TIMGn_Tx_WDT_SYS_RESET_LENGTH 长 。0: 100 ns, 1: 200 ns, 2: 300 ns, 3: 400 ns, 4: 500 ns, 5: 800 ns, 6: 1.6 μs, 7: 3.2 μs。(/写)

TIMGn_Tx_WDT_FLASHBOOT_MOD_EN 1 , Flash 启 保 。(/写)

Register 18.11. TIMGn_Tx_WDTCONFIG1_REG (0x004c)

TIMGn_Tx_WDT_CLK_PRESCALE	
31 16	
0x00001	Reset

TIMGn_Tx_WDT_CLK_PRESCALE SWDT 时 , 辨 = 12.5 ns * TIMGn_Tx_WDT_CLK_PRESCALE。(/写)

Register 18.12. TIMG*n*_T*x*_WDTCONFIG2_REG (0x0050)

31	0	
26000000		Reset

TIMG*n*_T*x*_WDTCONFIG2_REG SWDT 时 期中 0 超时时间。(/写)

Register 18.13. TIMG*n*_T*x*_WDTCONFIG3_REG (0x0054)

31	0	
0x007FFFFFF		Reset

TIMG*n*_T*x*_WDTCONFIG3_REG SWDT 时 期中 1 超时时间。(/写)

Register 18.14. TIMG*n*_T*x*_WDTCONFIG4_REG (0x0058)

31	0	
0x0000FFFFF		Reset

TIMG*n*_T*x*_WDTCONFIG4_REG SWDT 时 期中 2 超时时间。(/写)

Register 18.15. TIMG*n*_T*x*_WDTCONFIG5_REG (0x005c)

31	0	
0x0000FFFFF		Reset

TIMG*n*_T*x*_WDTCONFIG5_REG SWDT 时 期中 3 超时时间。(/写)

Register 18.16. TIMG*n*_T*x*_WDTFEED_REG (0x0060)

31	0	
0x000000000		Reset

TIMG*n*_T*x*_WDTFEED_REG 写 SWDT。(写)

Register 18.17. TIMG*n*_T*x*_WDTWPROTECT_REG (0x0064)

31	0	
0x050D83AA1		Reset

TIMG*n*_T*x*_WDTWPROTECT_REG 中 不 ，写保 。(/写)

Register 18.18. TIMG*n*_RTCCALICFG_REG (0x0068)

31	30 … 16	15	14 … 13	12	11 … 0	
TIMG*n*_RTC_CALI_START	TIMG*n*_RTC_CALI_MAX	TIMG*n*_RTC_CALI_RDY	TIMG*n*_RTC_CALI_CLK_SEL	TIMG*n*_RTC_CALI_START_CYCLING	(reserved)	
0	0x01	0	0x1	1	0 0 0 0 0 0 0 0 0 0 0 0	Reset

TIMG*n*_RTC_CALI_START_CYCLING 保 。(/写)

TIMG*n*_RTC_CALI_CLK_SEL 时 。0：RTC_CLK。1：RTC20M_D256_CLK。2：XTAL32K_CLK。(/写)

TIMG*n*_RTC_CALI_RDY 标记本 。( )

TIMG*n*_RTC_CALI_MAX 时间， 单 时 期。(/写)

TIMG*n*_RTC_CALI_START 启 。(/写)

Register 18.19. TIMG*n*_RTCCALICFG1_REG (0x006C)

31 … 7	5 … 0	
TIMG*n*_RTC_CALI_VALUE	(reserved)	
0x00000	0 0 0 0 0 0	Reset

TIMG*n*_RTC_CALI_VALUE 时 期 TIMG*n*_RTC_CALI_MAX 时 ，单 XTAL_CLK 期。( )

Register 18.20. TIMG*n*_T*x*_INT_ENA_REG (0x0098)

31																																3	2	1	0	
(reserved)																																	TIMG*n*_T*x*_INT_WDT_INT_ENA	TIMG*n*_T*x*_INT_T1_INT_ENA	TIMG*n*_T*x*_INT_T0_INT_ENA	
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	Reset

TIMG*n*_T*x*_INT_WDT_INT_ENA TIMG*n*_T*x*_INT_WDT_INT 中 中 。(/写)

TIMG*n*_T*x*_INT_T1_INT_ENA TIMG*n*_T*x*_INT_T1_INT 中 中 。(/写)

TIMG*n*_T*x*_INT_T0_INT_ENA TIMG*n*_T*x*_INT_T0_INT 中 中 。(/写)

Register 18.21. TIMG*n*_T*x*_INT_RAW_REG (0x009c)

31																																3	2	1	0	
(reserved)																																	TIMG*n*_T*x*_INT_WDT_INT_RAW	TIMG*n*_T*x*_INT_T1_INT_RAW	TIMG*n*_T*x*_INT_T0_INT_RAW	
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	Reset

TIMG*n*_T*x*_INT_WDT_INT_RAW TIMG*n*_T*x*_INT_WDT_INT 中 始 中 。( )

TIMG*n*_T*x*_INT_T1_INT_RAW TIMG*n*_T*x*_INT_T1_INT 中 始 中 。( )

TIMG*n*_T*x*_INT_T0_INT_RAW TIMG*n*_T*x*_INT_T0_INT 中 始 中 。( )

Register 18.22. TIMG*n*_T*x*_INT_ST_REG (0x00a0)

31																																3	2	1	0	
(reserved)																																	TIMG*n*_T*x*_INT_WDT_INT_ST	TIMG*n*_T*x*_INT_T1_INT_ST	TIMG*n*_T*x*_INT_T0_INT_ST	
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	Reset

TIMG*n*_T*x*_INT_WDT_INT_ST TIMG*n*_T*x*_INT_WDT_INT 中 屏蔽 中 。( )

TIMG*n*_T*x*_INT_T1_INT_ST TIMG*n*_T*x*_INT_T1_INT 中 屏蔽 中 。( )

TIMG*n*_T*x*_INT_T0_INT_ST TIMG*n*_T*x*_INT_T0_INT 中 屏蔽 中 。( )

Register 18.23. TIMG*n*_T*x*_INT_CLR_REG (0x00a4)

31																															3	2	1	0	
(reserved)																																TIMG*n*_T*x*_INT_WDT_INT_CLR	TIMG*n*_T*x*_INT_T1_INT_CLR	TIMG*n*_T*x*_INT_T0_INT_CLR	
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	Reset

TIMG*n*_T*x*_INT_WDT_INT_CLR 1 TIMG*n*_T*x*_INT_WDT_INT 中 。(写)

TIMG*n*_T*x*_INT_T1_INT_CLR 1 TIMG*n*_T*x*_INT_T1_INT 中 。(写)

TIMG*n*_T*x*_INT_T0_INT_CLR 1 TIMG*n*_T*x*_INT_T0_INT 中 。(写)

19 看门狗定时器 (WDT)

19.1 概述

ESP32 中 三个 时 ： 个 时 中 一个（ 时 ， 写 MWDT），RTC 中一个（ RTC 时 ， 写 RWDT）。不可 常，时 可 帮 中 。 三时 个 。 超 时间，闭 时 ， 个 可 被 三 中 一 。 ：中 、CPU 、 。 中， RWDT ， 包括 RTC 个 。 个超时时间 可单 设 。

Flash 启 期间，RWDT 一个 MWDT 启 测 启 。

19.2 主要特性

- 4 个 ， 可被单 闭
- 超时时间可
- 超时， 采 三 可 中 一 （中 、CPU 、 ）
- 32-bit 超时
- 写保 ， RWDT MWDT 被不 改变
- Flash 启 保 时间 ，SPI Flash 启 ， 重启 个

19.3 功能描述

19.3.1 时钟

RWDT 时 RTC 时 ， 常 32 KHz。MWDT 时 APB 时 ， 可 16-bit MWDT。RWDT MWDT 时 32-bit 超时 。 超时时间， ，超时 ， 一 启 。

19.3.2 运行过程

RWDT MWDT 时 ， 0 3， 0 重 始。 超时 超时时间 可 被单 。

超时 时 超时时间， 可 被 个 ：

- 中
  超时，中 被 。
- CPU
  超时， CPU 。 MWDT0 CPU PRO CPU， MWDT1 CPU APP CPU。 不 ， RWDT CPU 可 个，一个 不 CPU 。
-
  超时，包括 MWDT 被 。 本 中， CPU 设。 RTC 一个 ，不 。

- RTC
  超时， RTC 时 。 可 RWDT 中 。
- 闭
  不产 。

时， 时 重 0，超时 0 重 始。

19.3.3 写保护

个 MWDT RWDT 可被保 不 写 。 ， 个 写 保 。 MWDT TIMERS_WDT_WKEY, RWDT RTC_CNTL_WDT_WKEY。 时， 始 0x50D83AA1。 被 变，写保 。 时， 写 保 ， 包括 WDT 上写 操 被 。 按 步 WDT：

1. 闭写保
2.
3. 重 写保

19.3.4 Flash 启动保护

Flash 启 中， 时 0（TIMG0）中 MWDT RWDT 。 个 时 0 认 超时 。启 ， TIMERS_WDT_FLASHBOOT_MOD_EN 闭 MWDT Flash 启 保 。 RWDT， RTC_CNTL_WDT_FLASHBOOT_MOD_EN 。 ， 可 MWDT RWDT。

19.3.5 寄存器

MWDT 时 一部 ， 时 中 。RWDT RTC 一部 ， RTC 中 。

20 eFuse 控制器 (eFuse)

20.1 概述

ESP32 中 个 eFuse， 中 参 。 一 单 ，eFuse bit 一 被 写 1，不 0。eFuse 按 操 eFuse 中 个 参 中 个 bit 写。

参 可 eFuse 被 ， 。

20.2 主要特性

- 33 个 参
- 写保 （可 ）
- 保 （可 ）

20.3 功能描述

20.3.1 结构

eFuse 中 33 个 参 ， 个 参 不 ， 表 20-1 。 中，efuse_wr_disable、efuse_rd_disable、coding_scheme、BLK3_part_reserve 4 个 参 eFuse 。

表 20-1. 系统参数

参		efuse_wr_disable 写保	efuse_rd_disable 保	
efuse_wr_disable	16	1	-	eFuse
efuse_rd_disable	4	0	-	eFuse
flash_crypt_cnt	7	2	-	flash /
WIFI_MAC_Address	56	3	-	Wi-Fi MAC CRC
SPI_pad_config_hd	5	3	-	SPI I/O 个
XPD_SDIO_REG	1	5	-	flash 时 上
SDIO_TIEH	1	5	-	flash 时 : 1 时 3.3 V; 0 时 1.8 V
sdio_force	1	5	-	XPD_SDIO_REG SDIO_TIEH flash 时
BLK3_part_reserve	1	5	-	BLOCK3
SPI_pad_config_clk	5	6	-	SPI I/O 个
SPI_pad_config_q	5	6	-	SPI I/O 个
SPI_pad_config_d	5	6	-	SPI I/O 个
SPI_pad_config_cs0	5	6	-	SPI I/O 个
flash_crypt_config	4	10	3	flash /
coding_scheme*	2	10	3	eFuse
console_debug_disable	1	15	-	1 时， ROM BASIC 时 fallback 。

中 部 时 。

6 个 参 保 时， 可 被 可 被 。
保 时， 被 。

表 20-1 中 “efuse_rd_disable 保 ” 一 6 个 参 保
efuse_rd_disable 个 bit 。 参 efuse_rd_disable 中 个 bit 0，表 bit 参
保 。 参 efuse_rd_disable 中 个 bit 1，表 bit 参
保 。 个 参 保 ， 。

20.3.1.3 系统参数 coding_scheme

表 20-1 ， 参 BLOCK1、BLOCK2、BLOCK3 变化 。 一个 参

coding_scheme 。 BLOCK1、BLOCK2、BLOCK3 变化 ， 3 个 参 eFuse 中
bit 始 不变。 3 个 参 eFuse 中 间 编 。 参见表
20-2。

表 20-2. BLOCK1/2/3 编码

coding_scheme[1:0]	BLOCK1/2/3	编 方	eFuse 中 bit
00/11	256	编	256
01	192	3/4 编	256
10	128	重 编	256

三 编 方 ， 中，

- $BLOCKN$ 表 参 BLOCK1、BLOCK2 BLOCK3。
- $BLOCKN[255:0]$、$BLOCKN[191:0]$、$BLOCKN[127:0]$ 别表 三 编 方 个 参 个 bit。
- ${}^{e}BLOCKN[255:0]$ 表 个 参 编 个 bit， eFuse 中 个 bit。

非编码方式

$$ {}^{e}BLOCKN[255:0] = BLOCKN[255:0] $$

3/4 编码方式

$$ BLOCKN_i^j[7:0] = BLOCKN[48i+8j+7:48i+8j] \qquad i \in \{0,1,2,3\} \qquad j \in \{0,1,2,3,4,5\} $$
$$ {}^{e}BLOCKN_i^j[7:0] = {}^{e}BLOCKN[64i+8j+7:64i+8j] \qquad i \in \{0,1,2,3\} \qquad j \in \{0,1,2,3,4,5,6,7\} $$

$$
{}^{e}BLOCKN_i^j[7:0] = \begin{cases} BLOCKN_i^j[7:0] & j \in \{0,1,2,3,4,5\} \\ BLOCKN_i^0[7:0] \oplus BLOCKN_i^1[7:0] \\ \quad \oplus BLOCKN_i^2[7:0] \oplus BLOCKN_i^3[7:0] & j \in \{6\} \\ \quad \oplus BLOCKN_i^4[7:0] \oplus BLOCKN_i^5[7:0] \\ \sum_{l=0}^{5}(l+1)\sum_{k=0}^{7} BLOCKN_i^l[k] & j \in \{7\} \end{cases} \qquad i \in \{0,1,2,3\}
$$

$\oplus$ 表 按

$\sum$ + 表

重复编码方式

$$
{}^{e}BLOCKN[255:128] = {}^{e}BLOCKN[127:0] = BLOCKN[127:0]
$$

20.3.1.4 系统参数 BLK3_part_reserve

参 coding_scheme、BLOCK1、BLOCK2、BLOCK3 参 BLK3_part_reserve 。

BLK3_part_reserve 0 时，coding_scheme、BLOCK1、BLOCK2、BLOCK3 可 。

BLK3_part_reserve 1 时，coding_scheme、BLOCK1、BLOCK2、BLOCK3 必 被 3/4 编 。时 $BLOCK3[143:96]$ ${}^{e}BLOCK3[191:128]$ 不 。

20.3.2 烧写系统参数

写变长 参 BLOCK1、BLOCK2、BLOCK3 不 写 长 参 。**我们并不直接烧写系统参数 BLOCK1、BLOCK2、BLOCK3 本身，而是烧写其编码之后的值 ${}^{e}BLOCKN[255:0]$，这个值的位宽始终是 256。**所 写 长 参 写 参 本 。

30 个 长 参 一个 bit 3 个变长 参 编 一个 bit 别 一个 写 bit，表 20-3 。 写 参 时 bit。

表 20-3. 烧写寄存器

参				
		Bit		Bit
efuse_wr_disable	16	[15:0]	EFUSE_BLK0_WDATA0_REG	[15:0]
efuse_rd_disable	4	[3:0]		[19:16]
flash_crypt_cnt	7	[6:0]		[26:20]
uart_download_dis	1	[0]		[27]
WIFI_MAC_Address	56	[31:0]	EFUSE_BLK0_WDATA1_REG	[31:0]
		[55:32]	EFUSE_BLK0_WDATA2_REG	[23:0]
disable_app_cpu	1	[0]	EFUSE_BLK0_WDATA3_REG	[0]
disable_bt	1	[0]		[1]
pkg_version	4	[3:0]		[2], [11:9]
disable_cache	1	[0]		[3]
SPI_pad_config_hd	5	[4:0]		[8:4]
BLK3_part_reserve	1	[0]		[14]
CK8M Frequency	8	[7:0]	EFUSE_BLK0_WDATA4_REG	[7:0]
XPD_SDIO_REG	1	[0]		[14]

写 参 :

1. EFUSE_CLK EFUSE_CLK_SEL0 、EFUSE_CLK_SEL1 EFUSE_DAC_CONF EFUSE_DAC_CLK_DIV 。
2. 写 bit bit 1。
3. EFUSE_CONF 写 0x5A5A。
4. EFUSE_CMD 写 0x2。
5. EFUSE_CMD 0x0, 写 中 产 。
6. EFUSE_CONF 写 0x5AA5。
7. EFUSE_CMD 写 0x1。
8. EFUSE_CMD 0x0, 中 产 。
9. 写 bit bit 0。

表 20-4 , EFUSE_CLK EFUSE_CLK_SEL0 、EFUSE_CLK_SEL1 EFUSE_DAC_CONF EFUSE_DAC_CLK_DIV APB_CLK 。

表 20-4. 时序配置

		APB_CLK		
		26 MHz	40 MHz	80 MHz
EFUSE_CLK	EFUSE_CLK_SEL0[7:0]	250	160	80
	EFUSE_CLK_SEL1[7:0]	255	255	128
EFUSE_DAC_CONF	EFUSE_DAC_CLK_DIV[7:0]	52	80	100

可 识别 写/ 中 产 :

一:

1. EFUSE_INT_RAW bit 1/0, bit 1/0 1, 表 写/ 中 产 。
2. EFUSE_INT_CLR bit 1/0 1 写/ 中 。

:

1. EFUSE_INT_ENA bit 1/0 1, eFuse 写/ 中 。
2. Interrupt Matrix CPU EFUSE_INT 中 。
3. 写/ 中 产 。
4. 查 EFUSE_INT_ST bit 1/0 写/ 中 产 。
5. EFUSE_INT_CLR bit 1/0 1 写/ 中 。

写不 参 , 一个 参 中 不 bit 可 写中 别 。
写 , 个 参 中 写 bit 一 写中 。并 efuse_wr_disable 个 bit 参 写 , 写efuse_wr_disable bit。 可 一 写中 写

efuse_wr_disable 个 bit 参，时 写 efuse_wr_disable bit。**另外，严禁对已经烧写了的 bit 重复烧写。**

20.3.3 软件读取系统参数

30 个 长 参 一个 bit 3 个变长 参 本 一个 bit 别 一个 bit，表 20-5 。 参 。 参 BLOCK1、BLOCK2、BLOCK3 不 。 表 20-5 ，3 个参 256 个 bit，3/4 编 重 编 ，256 个 bit 中 部 。 编 ，$BLOCKN[255:0]$ 一个 bit bit 。 3/4 编 ，$BLOCKN[191:0]$ bit 。 重 编 ，$BLOCKN[127:0]$ bit 。 不 编 广 ， bit 中 。**软件从有用的寄存器 bit 读出的值是系统参数 BLOCK1、BLOCK2、BLOCK3 本身，不是这几个系统参数编码之后的值。**

表 20-5. 软件读取寄存器

参				
		Bit		Bit
efuse_wr_disable	16	[15:0]	EFUSE_BLK0_RDATA0_REG	[15:0]
efuse_rd_disable	4	[3:0]		[19:16]
flash_crypt_cnt	7	[6:0]		[26:20]
uart_download_dis	1	[0]		[27]
WIFI_MAC_Address	56	[31:0]	EFUSE_BLK0_RDATA1_REG	[31:0]
		[55:32]	EFUSE_BLK0_RDATA2_REG	[23:0]
disable_app_cpu	1	[0]	EFUSE_BLK0_RDATA3_REG	[0]
disable_bt	1	[0]		[1]
pkg_version	4	[3:0]		[2], [11:9]
disable_cache	1	[0]		[3]
SPI_pad_config_hd	5	[4:0]		[8:4]
BLK3_part_reserve	1	[0]		[14]
CK8M Frequency	8	[7:0]	EFUSE_BLK0_RDATA4_REG	[7:0]
XPD_SDIO_REG	1	[0]		[14]
SDIO_TIEH	1	[0]		[15]
sdio_force	1	[0]		[16]
SPI_pad_config_clk	5	[4:0]	EFUSE_BLK0_RDATA5_REG	[4:0]
SPI_pad_config_q	5	[4:0]		[9:5]
SPI_pad_config_d	5	[4:0]		[14:10]
SPI_pad_config_cs0	5	[4:0]		[19:15]
vol_level_hp_inv	2	[1:0]		[23:22]
dig_vol_l6	4	[3:0]		[27:24]
flash_crypt_config	4	[3:0]		[31:28]
coding_scheme	2	[1:0]	EFUSE_BLK0_RDATA6_REG	[1:0]
console_debug_disable	1	[0]		[2]
abstract_done_0	1	[0]		[4]
abstract_done_1	1	[0]		[5]
JTAG_disable	1	[0]		[6]
download_dis_encrypt	1	[0]		[7]
download_dis_decrypt	1	[0]		[8]

参				
		Bit		Bit
download_dis_cache	1	[0]		[9]
key_status	1	[0]		[10]
BLOCK1	256/192/128	[31:0]	EFUSE_BLK1_RDATA0_REG	[31:0]
		[63:32]	EFUSE_BLK1_RDATA1_REG	[31:0]
		[95:64]	EFUSE_BLK1_RDATA2_REG	[31:0]
		[127:96]	EFUSE_BLK1_RDATA3_REG	[31:0]
		[159:128]	EFUSE_BLK1_RDATA4_REG	[31:0]
		[191:160]	EFUSE_BLK1_RDATA5_REG	[31:0]
		[223:192]	EFUSE_BLK1_RDATA6_REG	[31:0]
		[255:224]	EFUSE_BLK1_RDATA7_REG	[31:0]
BLOCK2	256/192/128	[31:0]	EFUSE_BLK2_RDATA0_REG	[31:0]
		[63:32]	EFUSE_BLK2_RDATA1_REG	[31:0]
		[95:64]	EFUSE_BLK2_RDATA2_REG	[31:0]
		[127:96]	EFUSE_BLK2_RDATA3_REG	[31:0]
		[159:128]	EFUSE_BLK2_RDATA4_REG	[31:0]
		[191:160]	EFUSE_BLK2_RDATA5_REG	[31:0]
		[223:192]	EFUSE_BLK2_RDATA6_REG	[31:0]
		[255:224]	EFUSE_BLK2_RDATA7_REG	[31:0]
BLOCK3	256/192/128	[31:0]	EFUSE_BLK3_RDATA0_REG	[31:0]
		[63:32]	EFUSE_BLK3_RDATA1_REG	[31:0]
		[95:64]	EFUSE_BLK3_RDATA2_REG	[31:0]
		[127:96]	EFUSE_BLK3_RDATA3_REG	[31:0]
		[159:128]	EFUSE_BLK3_RDATA4_REG	[31:0]
		[191:160]	EFUSE_BLK3_RDATA5_REG	[31:0]
		[223:192]	EFUSE_BLK3_RDATA6_REG	[31:0]
		[255:224]	EFUSE_BLK3_RDATA7_REG	[31:0]

20.3.4 硬件模块使用系统参数

参 , 个 。 参 本 。

对于 BLOCK1、BLOCK2、BLOCK3 而言，硬件模块使用的是解码之后的值，不是它们编码之后的值。

20.3.5 中断

- EFUSE_PGM_DONE_INT： eFuse 写 , 中 被 。
- EFUSE_READ_DONE_INT： eFuse , 中 被 。

20.4 寄存器列表

eFuse 读取寄存器			
EFUSE_BLK0_RDATA0_REG	eFuse BLOCK 0 中 word 0	0x3FF5A000	
EFUSE_BLK0_RDATA1_REG	eFuse BLOCK 0 中 word 1	0x3FF5A004	
EFUSE_BLK0_RDATA2_REG	eFuse BLOCK 0 中 word 2	0x3FF5A008	
EFUSE_BLK0_RDATA3_REG	eFuse BLOCK 0 中 word 3	0x3FF5A00C	
EFUSE_BLK0_RDATA4_REG	eFuse BLOCK 0 中 word 4	0x3FF5A010	
EFUSE_BLK0_RDATA5_REG	eFuse BLOCK 0 中 word 5	0x3FF5A014	
EFUSE_BLK0_RDATA6_REG	eFuse BLOCK 0 中 word 6	0x3FF5A018	
EFUSE_BLK1_RDATA0_REG	eFuse BLOCK 1 中 word 0	0x3FF5A038	
EFUSE_BLK1_RDATA1_REG	eFuse BLOCK 1 中 word 1	0x3FF5A03C	
EFUSE_BLK1_RDATA2_REG	eFuse BLOCK 1 中 word 2	0x3FF5A040	
EFUSE_BLK1_RDATA3_REG	eFuse BLOCK 1 中 word 3	0x3FF5A044	
EFUSE_BLK1_RDATA4_REG	eFuse BLOCK 1 中 word 4	0x3FF5A048	
EFUSE_BLK1_RDATA5_REG	eFuse BLOCK 1 中 word 5	0x3FF5A04C	
EFUSE_BLK1_RDATA6_REG	eFuse BLOCK 1 中 word 6	0x3FF5A050	
EFUSE_BLK1_RDATA7_REG	eFuse BLOCK 1 中 word 7	0x3FF5A054	
EFUSE_BLK2_RDATA0_REG	eFuse BLOCK 2 中 word 0	0x3FF5A058	
EFUSE_BLK2_RDATA1_REG	eFuse BLOCK 2 中 word 1	0x3FF5A05C	
EFUSE_BLK2_RDATA2_REG	eFuse BLOCK 2 中 word 2	0x3FF5A060	
EFUSE_BLK2_RDATA3_REG	eFuse BLOCK 2 中 word 3	0x3FF5A064	
EFUSE_BLK2_RDATA4_REG	eFuse BLOCK 2 中 word 4	0x3FF5A068	
EFUSE_BLK2_RDATA5_REG	eFuse BLOCK 2 中 word 5	0x3FF5A06C	
EFUSE_BLK2_RDATA6_REG	eFuse BLOCK 2 中 word 6	0x3FF5A070	
EFUSE_BLK2_RDATA7_REG	eFuse BLOCK 2 中 word 7	0x3FF5A074	
EFUSE_BLK3_RDATA0_REG	eFuse BLOCK 3 中 word 0	0x3FF5A078	
EFUSE_BLK3_RDATA1_REG	eFuse BLOCK 3 中 word 1	0x3FF5A07C	
EFUSE_BLK3_RDATA2_REG	eFuse BLOCK 3 中 word 2	0x3FF5A080	
EFUSE_BLK3_RDATA3_REG	eFuse BLOCK 3 中 word 3	0x3FF5A084	
EFUSE_BLK3_RDATA4_REG	eFuse BLOCK 3 中 word 4	0x3FF5A088	
EFUSE_BLK3_RDATA5_REG	eFuse BLOCK 3 中 word 5	0x3FF5A08C	
EFUSE_BLK3_RDATA6_REG	eFuse BLOCK 3 中 word 6	0x3FF5A090	
EFUSE_BLK3_RDATA7_REG	eFuse BLOCK 3 中 word 7	0x3FF5A094	
eFuse 烧写寄存器			
EFUSE_BLK0_WDATA0_REG	写 eFuse BLOCK 0 中 word 0	0x3FF5A01c	/写
EFUSE_BLK0_WDATA1_REG	写 eFuse BLOCK 0 中 word 1	0x3FF5A020	/写
EFUSE_BLK0_WDATA2_REG	写 eFuse BLOCK 0 中 word 2	0x3FF5A024	/写
EFUSE_BLK0_WDATA3_REG	写 eFuse BLOCK 0 中 word 3	0x3FF5A028	/写
EFUSE_BLK0_WDATA4_REG	写 eFuse BLOCK 0 中 word 4	0x3FF5A02c	/写
EFUSE_BLK0_WDATA5_REG	写 eFuse BLOCK 0 中 word 5	0x3FF5A030	/写
EFUSE_BLK0_WDATA6_REG	写 eFuse BLOCK 0 中 word 6	0x3FF5A034	/写
EFUSE_BLK1_WDATA0_REG	写 eFuse BLOCK 1 中 word 0	0x3FF5A098	/写
EFUSE_BLK1_WDATA1_REG	写 eFuse BLOCK 1 中 word 1	0x3FF5A09c	/写

EFUSE_BLK1_WDATA2_REG	写 eFuse BLOCK 1 中 word 2	0x3FF5A0a0	/写
EFUSE_BLK1_WDATA3_REG	写 eFuse BLOCK 1 中 word 3	0x3FF5A0a4	/写
EFUSE_BLK1_WDATA4_REG	写 eFuse BLOCK 1 中 word 4	0x3FF5A0a8	/写
EFUSE_BLK1_WDATA5_REG	写 eFuse BLOCK 1 中 word 5	0x3FF5A0ac	/写
EFUSE_BLK1_WDATA6_REG	写 eFuse BLOCK 1 中 word 6	0x3FF5A0b0	/写
EFUSE_BLK1_WDATA7_REG	写 eFuse BLOCK 1 中 word 7	0x3FF5A0b4	/写
EFUSE_BLK2_WDATA0_REG	写 eFuse BLOCK 2 中 word 0	0x3FF5A0b8	/写
EFUSE_BLK2_WDATA1_REG	写 eFuse BLOCK 2 中 word 1	0x3FF5A0bc	/写
EFUSE_BLK2_WDATA2_REG	写 eFuse BLOCK 2 中 word 2	0x3FF5A0c0	/写
EFUSE_BLK2_WDATA3_REG	写 eFuse BLOCK 2 中 word 3	0x3FF5A0c4	/写
EFUSE_BLK2_WDATA4_REG	写 eFuse BLOCK 2 中 word 4	0x3FF5A0c8	/写
EFUSE_BLK2_WDATA5_REG	写 eFuse BLOCK 2 中 word 5	0x3FF5A0cc	/写
EFUSE_BLK2_WDATA6_REG	写 eFuse BLOCK 2 中 word 6	0x3FF5A0d0	/写
EFUSE_BLK2_WDATA7_REG	写 eFuse BLOCK 2 中 word 7	0x3FF5A0d4	/写
EFUSE_BLK3_WDATA0_REG	写 eFuse BLOCK 3 中 word 0	0x3FF5A0d8	/写
EFUSE_BLK3_WDATA1_REG	写 eFuse BLOCK 3 中 word 1	0x3FF5A0dc	/写
EFUSE_BLK3_WDATA2_REG	写 eFuse BLOCK 3 中 word 2	0x3FF5A0e0	/写
EFUSE_BLK3_WDATA3_REG	写 eFuse BLOCK 3 中 word 3	0x3FF5A0e4	/写
EFUSE_BLK3_WDATA4_REG	写 eFuse BLOCK 3 中 word 4	0x3FF5A0e8	/写
EFUSE_BLK3_WDATA5_REG	写 eFuse BLOCK 3 中 word 5	0x3FF5A0ec	/写
EFUSE_BLK3_WDATA6_REG	写 eFuse BLOCK 3 中 word 6	0x3FF5A0f0	/写
EFUSE_BLK3_WDATA7_REG	写 eFuse BLOCK 3 中 word 7	0x3FF5A0f4	/写
控制寄存器			
EFUSE_CLK_REG	时	0x3FF5A0F8	/写
EFUSE_CONF_REG	操	0x3FF5A0FC	/写
EFUSE_CMD_REG	/写	0x3FF5A104	/写
中断寄存器			
EFUSE_INT_RAW_REG	始中	0x3FF5A108	
EFUSE_INT_ST_REG	屏蔽中	0x3FF5A10C	
EFUSE_INT_ENA_REG	中	0x3FF5A110	/写
EFUSE_INT_CLR_REG	中	0x3FF5A114	写
其它寄存器			
EFUSE_DAC_CONF_REG	eFuse 时	0x3FF5A118	/写
EFUSE_DEC_STATUS_REG	3/4 编	0x3FF5A11C	

20.5 寄存器

Register 20.1. EFUSE_BLK0_RDATA0_REG (0x000)

(reserved)	EFUSE_RD_UART_DOWNLOAD_DIS	EFUSE_RD_FLASH_CRYPT_CNT	EFUSE_RD_EFUSE_RD_DIS	EFUSE_RD_EFUSE_WR_DIS	
31 28	27	26 20	19 16	15 0	
0 0 0 0	0	0 0 0 0 0 0 0 0	0 0 0 0	0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0	Reset

EFUSE_RD_UART_DOWNLOAD_DIS uart_download_dis , ESP32 ECO V3 。( )

EFUSE_RD_FLASH_CRYPT_CNT flash_crypt_cnt 。( )

EFUSE_RD_EFUSE_RD_DIS efuse_rd_disable 。( )

EFUSE_RD_EFUSE_WR_DIS efuse_wr_disable 。( )

Register 20.2. EFUSE_BLK0_RDATA1_REG (0x004)

31 0	
0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0	Reset

EFUSE_BLK0_RDATA1_REG WIFI_MAC_Address 32 。( )

Register 20.3. EFUSE_BLK0_RDATA2_REG (0x008)

(reserved)	EFUSE_RD_WIFI_MAC_CRC_HIGH	
31 24	23 0	
0 0 0 0 0 0 0 0	0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0	Reset

EFUSE_RD_WIFI_MAC_CRC_HIGH WIFI_MAC_Address 24 。( )

Register 20.4. EFUSE_BLK0_RDATA3_REG (0x00c)

31	12	11	9	8	4	3	2	1	0	
(reserved)		EFUSE_RD_CHIP_VER_PKG		EFUSE_RD_SPI_PAD_CONFIG_HD		EFUSE_RD_CHIP_VER_DIS_CACHE	EFUSE_RD_CHIP_VER_PKG	EFUSE_RD_CHIP_VER_DIS_BT	EFUSE_RD_CHIP_VER_DIS_APP_CPU	
0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0		0 0 0		0 0 0 0 0		0	0	0	0	Reset

EFUSE_RD_CHIP_VER_PKG 个 bit 标记 版本， 。( )

EFUSE_RD_SPI_PAD_CONFIG_HD SPI_pad_config_hd 。( )

EFUSE_RD_CHIP_VER_DIS_CACHE cache 。( )

EFUSE_RD_CHIP_VER_PKG 个 bit 标记 版本， 。( )

EFUSE_RD_CHIP_VER_DIS_BT 。( )

EFUSE_RD_CHIP_VER_DIS_APP_CPU APP CPU。( )

Register 20.5. EFUSE_BLK0_RDATA4_REG (0x010)

31	17	16	15	14	13	8	7	0	
(reserved)		EFUSE_RD_SDIO_FORCE	EFUSE_RD_SDIO_TIEH	EFUSE_RD_XPD_SDIO	(reserved)		ESFUSE_RD_CK8M_FREQ		
0 0 0 0 0 0 0 0 0 0 0 0 0 0 0		0	0	0	0 0 0 0 0 0		0 0 0 0 0 0 0 0		Reset

EFUSE_RD_SDIO_FORCE sdio_force 。( )

EFUSE_RD_SDIO_TIEH SDIO_TIEH 。( )

EFUSE_RD_XPD_SDIO XPD_SDIO_REG 。( )

ESFUSE_RD_CK8M_FREQ RTC8M_CLK 。( )

Register 20.6. EFUSE_BLK0_RDATA5_REG (0x014)

EFUSE_RD_FLASH_CRYPT_CONFIG	EFUSE_RD_DIG_VOL_L6	EFUSE_RD_VOL_LEVEL_HP_INV	(reserved)	EFUSE_RD_SPI_PAD_CONFIG_CS0	EFUSE_RD_SPI_PAD_CONFIG_D	EFUSE_RD_SPI_PAD_CONFIG_Q	EFUSE_RD_SPI_PAD_CONFIG_CLK	
31 28	27 24	23 22	21 20	19 15	14 10	9 5	4 0	
0 0 0 0	0 0 0 0	0 0	0 0	0 0 0 0 0	0 0 0 0 0	0 0 0 0 0	0 0 0 0 0	Reset

EFUSE_RD_FLASH_CRYPT_CONFIG flash_crypt_config 。( )

EFUSE_RD_DIG_VOL_L6 6 dig regulator 1.2 V 差 。( )

EFUSE_RD_VOL_LEVEL_HP_INV 标识 CPU 240 MHz flash/PSRAM 80 MHz 。0x0：采 7；0x1：采 6；0x2：采 5；0x3：采 4。( )

EFUSE_RD_SPI_PAD_CONFIG_CS0 SPI_pad_config_cs0 。( )

EFUSE_RD_SPI_PAD_CONFIG_D SPI_pad_config_d 。( )

EFUSE_RD_SPI_PAD_CONFIG_Q SPI_pad_config_q 。( )

EFUSE_RD_SPI_PAD_CONFIG_CLK SPI_pad_config_clk 。( )

3FHJTUFS &'64&@#-, @3%"5" @3&(Y

SFTFSWFE

&'64&@3%@,&:@45"564 &'64&@3%@%*4"#-&@%E-@XOM)R&BE@E¥JT@DBDIF
&'64&@3%@%*4"#-&@%-@P%X&C$M:P15E@E¥Th@EFDSZQU
&'64&@3%@%*4"#-&@%-@R&X$C3M1P5BE@E¥Th@FODSZQU
&'64&@3%@%*4"#-&+@"+(@'E-EJT#BC@MF
&'64&@3%@"#4@%B0C7&@SBDU¥@E@POF@
&'64&@3%@"#4@%B0C7&@SBDU¥@E@POF@
&'64&@3%@$0/40-&@%&#6(@P%O4T'P#M&@EFO¥Vh@EJTBCMF
&'64&@3%@$0%*/(@4D$P&E.J&OH@¥DhFNF

3FHJTUFS &'64&@#-, @8%"5" @3&(Y D

SFTFSWFE &'64&@6"35@%08/-0"%@%*4 &'64&@'-"4)@$3:15@$/5 &'64&@3%@%*4 &'64&@83@%*4

&'64&@6"35@%08/-0"%@%B4SU@EPXO¥MÇ@BE@4E1JT&$0µ7rb •
&'64&@'-"4)@$3:15@$/5GMBTI@D¥ZbQU@DOU
&'64&@3%@%*4GVTF@SE¥@EJTB•CMF
&'64&@83@%*4FGVTF@XS¥@EJTB•CMF

Register 20.9. EFUSE_BLK0_WDATA1_REG (0x020)

31																															0	
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	Reset

EFUSE_BLK0_WDATA1_REG 写WIFI_MAC_Address 32 。(/写)

Register 20.10. EFUSE_BLK0_WDATA2_REG (0x024)

(reserved)								EFUSE_WIFI_MAC_CRC_HIGH																								
31							24	23																							0	
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	Reset

EFUSE_WIFI_MAC_CRC_HIGH 写WIFI_MAC_Address 24 。(/写)

Register 20.11. EFUSE_BLK0_WDATA3_REG (0x028)

(reserved)																								EFUSE_CHIP_VER_PKG			EFUSE_SPI_PAD_CONFIG_HD					EFUSE_CHIP_VER_DIS_CACHE	EFUSE_CHIP_VER_PKG	EFUSE_CHIP_VER_DIS_BT	EFUSE_CHIP_VER_DIS_APP_CPU	
31																							12	11		9	8				4	3	2	1	0	
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	Reset

EFUSE_CHIP_VER_PKG 个bit 写 版本， 。(/写)

EFUSE_SPI_PAD_CONFIG_HD 写SPI_pad_config_hd 。(/写)

EFUSE_CHIP_VER_DIS_CACHE 写 cache 。(/写)

EFUSE_CHIP_VER_PKG 个bit 写 版本， 。 /写)

EFUSE_CHIP_VER_DIS_BT 写 。(/写)

EFUSE_CHIP_VER_DIS_APP_CPU 写 APP CPU。(/写)

Register 20.12. EFUSE_BLK0_WDATA4_REG (0x02c)

(reserved)	EFUSE_SDIO_FORCE	EFUSE_SDIO_TIEH	EFUSE_XPD_SDIO	(reserved)	ESFUSE_CK8M_FREQ	
31 17	16	15	14	13 8	7 0	
0 0 0 0 0 0 0 0 0 0 0 0 0 0 0	0	0	0	0 0 0 0 0 0	0 0 0 0 0 0 0 0	Reset

EFUSE_SDIO_FORCE 写SDIO_TIEH 。(/写)

EFUSE_SDIO_TIEH 写SDIO_TIEH 。(/写)

EFUSE_XPD_SDIO 写XPD_SDIO_REG 。(/写)

ESFUSE_CK8M_FREQ 写RTC8M_CLK 。(/写)

Register 20.13. EFUSE_BLK0_WDATA5_REG (0x030)

EFUSE_FLASH_CRYPT_CONFIG	EFUSE_DIG_VOL_L6	EFUSE_VOL_LEVEL_HP_INV	(reserved)	EFUSE_SPI_PAD_CONFIG_CS0	EFUSE_SPI_PAD_CONFIG_D	EFUSE_SPI_PAD_CONFIG_Q	EFUSE_SPI_PAD_CONFIG_CLK	
31 28	27 24	23 22	21 20	19 15	14 10	9 5	4 0	
0 0 0 0	0 0 0 0	0 0	0 0	0 0 0 0 0	0 0 0 0 0	0 0 0 0 0	0 0 0 0 0	Reset

EFUSE_FLASH_CRYPT_CONFIG 写flash_crypt_config 。(/写)

EFUSE_DIG_VOL_L6 写 6 dig regulator 1.2 V 差 。(/写)

EFUSE_VOL_LEVEL_HP_INV 写CPU 240 MHz flash/PSRAM 80 MHz 。
0x0：采 7；0x1：采 6；0x2：采 5；0x3：采 4。(/写)

EFUSE_SPI_PAD_CONFIG_CS0 写SPI_pad_config_cs0 。(/写)

EFUSE_SPI_PAD_CONFIG_D 写SPI_pad_config_d 。(/写)

EFUSE_SPI_PAD_CONFIG_Q 写SPI_pad_config_q 。(/写)

EFUSE_SPI_PAD_CONFIG_CLK 写SPI_pad_config_clk 。(/写)

Register 20.14. EFUSE_BLK0_WDATA6_REG (0x034)

(reserved)	EFUSE_KEY_STATUS	EFUSE_DISABLE_DL_CACHE	EFUSE_DISABLE_DL_DECRYPT	EFUSE_DISABLE_DL_ENCRYPT	EFUSE_DISABLE_JTAG	EFUSE_ABS_DONE_1	EFUSE_ABS_DONE_0	(reserved)	EFUSE_CONSOLE_DEBUG_DISABLE	EFUSE_CODING_SCHEME	
31 … 11	10	9	8	7	6	5	4	3	2	1 … 0	
0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0	0	0	0	0	0	0	0	0	0	0 0	Reset

EFUSE_KEY_STATUS 写key_status 。(/写)

EFUSE_DISABLE_DL_CACHE 写download_dis_cache 。(/写)

EFUSE_DISABLE_DL_DECRYPT 写download_dis_decrypt 。(/写)

EFUSE_DISABLE_DL_ENCRYPT 写download_dis_encrypt 。(/写)

EFUSE_DISABLE_JTAG 写JTAG_disable 。(/写)

EFUSE_ABS_DONE_1 写abstract_done_1 。(/写)

EFUSE_ABS_DONE_0 写abstract_done_0 。(/写)

EFUSE_CONSOLE_DEBUG_DISABLE 写console_debug_disable 。(/写)

EFUSE_CODING_SCHEME 写coding_scheme 。(/写)

Register 20.15. EFUSE_BLK1_RDATA*n*_REG (*n*: 0-7) (0x38+4**n*)

31 … 0	
0x000000000	Reset

EFUSE_BLK1_RDATA*n*_REG BLOCK1 中 word *n* 。( )

Register 20.16. EFUSE_BLK2_RDATA*n*_REG (*n*: 0-7) (0x58+4**n*)

31 … 0	
0x000000000	Reset

EFUSE_BLK2_RDATA*n*_REG BLOCK2 中 word *n* 。( )

Register 20.17. EFUSE_BLK3_RDATA*n*_REG (*n*: 0-7) (0x78+4**n*)

31 … 0
0x000000000

Reset

EFUSE_BLK3_RDATA*n*_REG BLOCK3 中 word *n* 。( )

Register 20.18. EFUSE_BLK1_WDATA*n*_REG (*n*: 0-7) (0x98+4**n*)

31 … 0
0x000000000

Reset

EFUSE_BLK1_WDATA*n*_REG BLOCK1 中 word *n* 。(/写)

Register 20.19. EFUSE_BLK2_WDATA*n*_REG (*n*: 0-7) (0xB8+4**n*)

31 … 0
0x000000000

Reset

EFUSE_BLK2_WDATA*n*_REG BLOCK2 中 word *n* 。(/写)

Register 20.20. EFUSE_BLK3_WDATA*n*_REG (*n*: 0-7) (0xD8+4**n*)

31 … 0
0x000000000

Reset

EFUSE_BLK3_WDATA*n*_REG BLOCK3 中 word *n* 。(/写)

Register 20.21. EFUSE_CLK_REG (0x0f8)

(reserved)	EFUSE_CLK_SEL1	EFUSE_CLK_SEL0
31 … 16	15 … 8	7 … 0
0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0	0x040	0x052

Reset

EFUSE_CLK_SEL1 eFuse 时 。(/写)

EFUSE_CLK_SEL0 eFuse 时 。(/写)

Register 20.22. EFUSE_CONF_REG (0x0fc)

(reserved)	EFUSE_OP_CODE	
31 … 16	15 … 0	
0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0	0x00000	Reset

EFUSE_OP_CODE eFuse 操 。(/写)

Register 20.23. EFUSE_CMD_REG (0x104)

(reserved)	EFUSE_PGM_CMD	EFUSE_READ_CMD	
31 … 2	1	0	
0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0	0	0	Reset

EFUSE_PGM_CMD 1 始 写操 ; 写 0。(/写)

EFUSE_READ_CMD 1 始 操 ; 0。(/写)

Register 20.24. EFUSE_INT_RAW_REG (0x108)

(reserved)	EFUSE_PGM_DONE_INT_RAW	EFUSE_READ_DONE_INT_RAW	
31 … 2	1	0	
0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0	0	0	Reset

EFUSE_PGM_DONE_INT_RAW EFUSE_PGM_DONE_INT 中 始中 。( )

EFUSE_READ_DONE_INT_RAW EFUSE_READ_DONE_INT 中 始中 。( )

Register 20.25. EFUSE_INT_ST_REG (0x10c)

31																															2	1	0	
(reserved)																																EFUSE_PGM_DONE_INT_ST	EFUSE_READ_DONE_INT_ST	
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	Reset

Register 20.28. EFUSE_DAC_CONF_REG (0x118)

(reserved)	EFUSE_DAC_CLK_DIV	
31 … 8	7 … 0	
0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0	40	Reset

EFUSE_DAC_CLK_DIV eFuse 时 。(/号)

Register 20.29. EFUSE_DEC_STATUS_REG (0x11c)

(reserved)	EFUSE_DEC_WARNINGS	
31 … 12	11 … 0	
0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0	0 0 0 0 0 0 0 0 0 0 0 0	Reset

EFUSE_DEC_WARNINGS 1, 3/4 编 时 。( )

21 双线汽车接口 (TWAI)

21.1 概述

(Two-wire Automotive Interface, TWAI ®) 一 、 播 , 测 、 报 裁 。TWAI (可参见 TWAI)。

ESP32 包 一个 TWAI ，可 部 TWAI 。TWAI 包 一 ，可 产 、 信 、 信 。

21.2 主要特性

ESP32 TWAI :

- ISO 11898-1 (CAN2.0)
- 标 （11-bit 标识） 扩 （29-bit 标识）
- 25 Kbit/s ~ 1 Mbit/s
- 操
  - 常
  - （不 ）
  - 测 （ 时不 ）
- 64-byte FIFO
-
  - 单 （ 时不 重 ）
  - （TWAI 时 报 ）
- 波 （ 单 波 波 ）
- 测
  -
  - 报 可
  - 捕
  - 裁 捕

21.3 功能性协议

21.3.1 TWAI 性能

TWAI 中 个 个 ，并 报 。TWAI :

单通道通信与不归零编码： TWAI 单 ， 半 。 步 单 中 ， 不 （ 时 ）。TWAI 上报 采 不 编 (NRZ) 方 。

位值：单 可 , 0, 1。
比 。 上 (, 差)
。

位填充： TWAI 报 。 个 ( ) 个 ,
插 一个 补 。 , 5 个 一个 视 。 :
SOF、 裁 、 、 CRC (可参见 21.3.2)。

多播： 个 上时, 。 上 保 一 ,
(可参见 21.3.3)。

多主机： 可 。 ,
。

报文优先级与仲裁： 个 个 时 , TWAI 保 中一个
裁 。 报 裁 个 可 裁。

错误检测与通报： 测 上 , 并 报 测 。

故障限制： 一 / 时, 。 超 一 时,
闭 。

可配置位速率：单 个 TWAI 可 。 , 个 中
。

发送器与接收器：不 时, TWAI 可 。

- 产 报 。 一 , 裁。 ,
裁 个 可 。
- 。

21.3.2 TWAI 报文

TWAI 报 , 并 测 上 时 。报 不
, 不 。

TWAI :

-
-
-
-
- 间

TWAI :

- 标 (SFF) 11-bit 标识
- 扩 (EFF) 29-bit 标识

21.3.2.1 数据帧和远程帧

，可 0 ~ 8 。 标识
，中不包 。 ，中包 。 21-1
不 不 中包 。

图 21-1. 数据帧和远程帧中的位域

仲裁域

个 个 时 时， 裁 上 裁 。
裁 时， 一个 时 测 一个 ， 表 个
。 ， 个 裁， 。

裁 标识 。 表 0， 表 1， ：

- ID 裁。
- ID ， RTR ， 。
- ID 11 ， 扩 SRR ， 标 扩 。

控制域

长 (DLC) ，DLC 表 一个 中 ， 一个
。DLC 。

数据域

中包 一个 。 中不包 。

CRC 域

CRC CRC 。CRC 一个 15-bit 冗 编 ， 中 （ SOF ）中 。

确认域

认 (ACK) 认槽 认 ， ： 报 报 。

表 21-1. SFF 和 EFF 中的数据帧和远程帧

/	
SOF	始 (SOF) 一个 步 上 单个 。
Base ID	标识 (ID.28 ~ ID.18) SFF 11-bit 标识 ， EFF 中 29-bit 标识 11-bit。
RTR	(RTR) 报 （ ） （ ）。 ， 个 一个 标识 时， 始 裁。
SRR	EFF 中 (SRR)， SFF 中 RTR 。
IDE	标识 扩 (IED) 报 SFF （ ） EFF （ ）。 ， SFF EFF 标识 时，SFF 始 EFF 裁。
Extd ID	扩 标识 (ID.17 ~ ID.0) EFF 中 29-bit 标识 18-bit。
r1	r1（保 1）始 。
r0	r0（保 0）始 。
DLC	长 (DLC) 4-bits， 包 0 ~ 8 中 一 。 DLC 表 包 。 DLC 表 。
	表 。 DLC 。 0， 。
CRC	CRC 一个 15-bit 冗 编 。
CRC	CRC CRC 单一 。
认槽	认槽 ，表 。 认槽中 一个 ， ， 一个 认槽。
认	认 一个单一 。
EOF	(EOF) 标 ， 个 。

21.3.2.2 错误帧和过载帧

错误帧

测 时， 一个 。 一个 标 ， 标 个 ， 而 。 ， 测 并 时， 测 一个 并 。 ， 时， 上 可 报 上 。

测 时， 一个 。 : CRC ， 认 一个 始（可参见 21.3.3 ）。 21-2 一个 包 不 :

图 21-2. 错误帧中的位域

表 21-2. 错误帧

标	标 包 : 标 被 标 ， 标 6 个 ，被 标 6 个 （被 裁 时 ）。 标 ，被 被 标 。
标	标 上 标 。 0 ~ 6 ， 测 一个 时 （ 测 上 一个 时 可 ）。
	标 / ， 8 个 。

过载帧

包 标 。 别 。 21-3 中包 :

图 21-3. 过载帧中的位域

表 21-3. 过载帧

标	6 个 。 标 。
标	标 , 标 。
	8 个 。 。

:

1. 部 一个 。
2. 间 个 个 测 。
3. 八个（ 一个） 测 。 , TEC REC 不 （可参见 21.3.3 ）。

时， :

- 1 间 一个 始。
- 2、3 测 一个 始。
- 一个 , 可 个 。

21.3.2.3 帧间距

间 间 。 必 一 一个 间 ，不（ 、 、 、 ）。 , 一个 。

21-4 间 中包 :

图 21-4. 帧间距中的域

表 21-4. 帧间距

间	
间	间 3 个 。
	被 报 , 中 包 一个 , 8 个 。 中不 个 。
	长 。 SOF 时， 。 中 , SOF 间 一 。

21.3.3 TWAI 错误

21.3.3.1 错误类型

TWAI 中 包括 :

位错误

一个 ( ) 测 时 (, 时 测), 。 , , 裁 认槽 被 标 中, 时 测 不 认 。

填充错误

测 6 个 时 (编), 。

CRC 错误

CRC 。 中 CRC 不 时, CRC 。

格式错误

个报 中 中包 时, 可 测 。比 , r1 r0 必 。

确认错误

认槽中 测 时, 认 。

21.3.3.2 错误状态

TWAI 个 个 故 , 。 个 别 : (TEC) (REC)。TWAI 包 。

主动错误

可参 中, 测 时可 标 。

被动错误

被 可参 中, 测 时 一 被 标 。被 , 间 中设 。

离线

(, 不)。

21.3.3.3 错误计数

TEC REC / 。**请注意，一条报文传输中可应用多个规则。**

1. 测 时, REC 1。 测 标 标 时间 。
2. 标 , 一个 测 时, REC 8。
3. 标 时, TEC 8。 , 不 :
   - 被 , 槽 测 产 , 被 标 时 测 时, TEC 不 。

- 裁期间 而 标 ， 本 测 ， TEC 不 。

4. 标 标 时 测 ， TEC 8。
5. 标 标 时 测 ， REC 8。
6. /被 标 标 ， 7 个 。 （ 标 标 时） 测 14 个 ， 被 标 测 8 个 ， TEC 8，而 REC 8。 8 个 时，（ ）TEC （ ）REC 8。
7. 报 （ ACK， EOF ），TEC 1， TEC 0。
8. 报 （ 认槽 测 ， ACK）， REC 。
   - REC 1 ~ 127 间， 1。
   - REC 127， 127。
   - REC 0， 保 0。
9. 一个 TEC / REC 128 时， 变 被 。 上 ， 标 。 ，一 REC 128， ， REC 128 。
10. TEC 256 时， 变 。
11. 被 TEC REC 127， 变 。
12. 上 测 128 11 个 ， 可变 （TEC REC 重设 0）。

21.3.4 TWAI 位时序

21.3.4.1 名义位

TWAI TWAI 。 ， 必 一 。

- **名义位速率** 比 。
- **名义位时间 1/名义位速率**。

个 时间中 个 ， 个时间 (Time Quanta) 。**时间定额** 时间单 ， 一 时 个 中。 21-5 一个 时间 包 。

TWAI 一个时间 时间步长中 操 ， 个时间 中 TWAI 。 个 时间 中 不 （ - ， ），边 产 。PBS1 PBS2 被视 采 ， 采 个 。

图 21-5. 位时序构成

表 21-5. 名义位时序中包含的段

步 (SS)	SS（ 步 ） 长 1 个时间 。 步 常， 边 。
时期 1 (PBS1)	PBS1 长 可 1 ~ 16 个时间 ， 补偿 中 时间。可 PBS1 长 ， 所 步。
时期 2 (PBS2)	PBS2 长 可 1 ~ 8 个时间 ， 补偿 中 时间。可 PBS2 长 ， 所 步。

21.3.4.2 硬同步与再同步

时 ， 一 上 时 可 。 所， 边 可 步 。 上 边 TWAI 步方 。这 边 TQ（时间 ） **相位错误 “ e ”**， SS 。

- (e > 0)： 边 步 采 （ ，边 ）。
- 被 (e < 0)： 边 个 采 步 （ ，边 ）。

，可 步方 ， **硬同步 再同步**。**硬同步 再同步** 守 ：

- 单个 时 中 可 一 步。
- 步 可 边 上。

硬同步

期间， 步 变化边 上（ SOF 上）。 时， 重 部 时 ， 所 变化边 重 时 步 。

再同步

期间， 步 变化边 上。 边 上 (e > 0)， PBS1 长 。 边 上 被 (e < 0)， PBS2 长 。

PBS1/PBS2 时间 ， 时 可 步 (SJW) 。

- SJW 时，PBS1/PBS2 / **e** 个时间 。 步 。
- SJW 时，PBS1/PBS2 / SJW 时间 。 ， ，可 个 步 。

21.4 结构概述

图 21-6. TWAI 概略图

ESP32 中包 一个 TWAI 。 21-6 TWAI 。

21.4.1 寄存器模块

ESP32 CPU 32-bit 设。 ，TWAI 中 部 (bits [7:0]) 上 。 中，bits [31:8] 写 时被忽 ， 时 0。

配置寄存器

TWAI ， 、操 、 波 。 TWAI 时，才可 改 （可参见 21.5.1 ）。

指令寄存器

CPU TWAI ， 报 。 TWAI 操 时，才可 改 （可参见 21.5.1 ）。

中断 & 状态寄存器

中 TWAI 中 （ 个 一个单 表 ）。 TWAI 。

错误管理寄存器

包括 捕 。 表 TEC REC 。捕 记录 ， TWAI 测 ， 时 裁。

发送缓冲寄存器

13 ， TWAI 报 。

接收缓冲寄存器

13 , 单个报 。 FIFO , FIFO 中 一个报 被 中。

, 、 波 (包 0x0040 ~ 0x0070)。 :

- TWAI 时, 被 波 中。
- TWAI 操 时:
  - 中。
  - 写 中。

21.4.2 位流处理器

(BSP) (, CRC) 并 时 (BTL) 。 时, BSP BTL 中 (, CRC), 并 报 FIFO。BSP 测 TWAI 上 并 报 (EML)。

21.4.3 错误管理逻辑

(EML) TEC REC , 记录 (,), , 保 BSP 标 。 , 记录 TWAI 裁时 bit 。

21.4.4 位时序逻辑

时 (BTL) 报 。BTL 步 时 , 保 。 个可编 , 可设 个 TQ (时间) 长 , 时间 报 。

21.4.5 接收滤波器

波 一个可编 报 单 , TWAI 报 标识 报 。 波 报 才 被 FIFO 中。 可 波 : 单 波 、 波 。

21.4.6 接收 FIFO

FIFO 64-byte (TWAI 部), 波 报 。 FIFO 中 报 可 不 (3 ~ 13 byte 间)。 FIFO 时 (间不 一个 报), 中 , 报 , FIFO 中 间。 FIFO 中 一 报 被 13-byte 中, 报 被 ( )。 , FIFO 中 一 报 , FIFO 中上一 报 间 被 。

21.5 功能描述

21.5.1 模式

ESP32 TWAI : 操 。 TWAI_RESET_MODE 1, ; 0, 操 。

21.5.1.1 复位模式

TWAI 。 ，TWAI 。 时，TWAI TWAI 。 ，TWAI 报 （包括 ）。 报 被 。 ，TWAI 报 。

21.5.1.2 操作模式

操 ，TWAI ，并 写保 ， 保 期间保 一 。操 ，TWAI 可 报 （包括 ），TWAI 。TWAI 三 ：

- **正常模式：** TWAI 可 包 报 （ ， ）。
- **自测模式：** 常 ， ，TWAI 报 时， CRC ， 不 产 。 常 TWAI 测时 。
- **只听模式：** TWAI 可 报 ， TWAI 上保 被 。 ，TWAI 报 、 。 保 。 TWAI 。

， （ ， 操 时），TWAI 11 个 ，才 上 TWAI （ ，可 报 ）。

21.5.2 位时序

TWAI 必 时 。 TWAI_BUS_TIMING_0_REG TWAI_BUS_TIMING_1_REG 中 ， 个 包 ：

表 21-6 TWAI_BUS_TIMING_0_REG 包 。

表 21-6. TWAI_CLOCK_DIVIDER_REG 的 bit 信息; TWAI 地址 0x18

Bit 31-8	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
Reserved	SJW.1	SJW.0	BRP.5	BRP.4	BRP.3	BRP.2	BRP.1	BRP.0

说明：

- (BRP)：TWAI 时间 时 APB 时 ，APB 时 常 80 MHz。可 ， 中 t_{Tq} 时间 时 时 期，t_{CLK} APB 时 期：
  $t_{Tq} = 2 \times t_{CLK} \times (2^5 \times BRP.5 + 2^4 \times BRP.4 + 2^3 \times BRP.3 + 2^2 \times BRP.2 + 2^1 \times BRP.1 + 2^0 \times BRP.0 + 1)$
- 步 (SJW)：SJW SJW.0 SJW.1 中 ， ：SJW = (2 x SJW.1 + SJW.0 + 1)。

表 21-7 TWAI_BUS_TIMING_1_REG 包 。

表 21-7. TWAI_BUS_TIMING_1_REG 的 bit 信息; TWAI 地址 0x1c

Bit 31-8	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
保	SAM	PBS2.2	PBS2.1	PBS2.0	PBS1.3	PBS1.2	PBS1.1	PBS1.0

说明：

- PBS1: 时期 1 中 时间 ：(8 x PBS1.3 + 4 x PBS1.2 + 2 x PBS1.1 + PBS1.0 + 1)。

- PBS2: 时期 2 中 时间 : (4 x PBS2.2 + 2 x PBS2.1 + PBS2.0 + 1)。
- SAM: 1 启 三 采 。 / 中 , 上 。

21.5.3 中断管理

ESP32 TWAI 中 , 中 TWAI_INT_RAW_REG 中 一个 表 。 个 中 , 设 TWAI_INT ENA_REG 中 。

TWAI 中 :

- 中
- 中
- 报 中
- 中
- 被 中
- 裁 中
- 中

TWAI_INT_RAW_REG 一个 个中 1，TWAI 中 中 ，TWAI_INT_RAW_REG 中 被 时，TWAI 中 中 。TWAI_INT_RAW_REG 被 ，中 中 。 ，中 不包括 ，TWAI_RELEASE_BUF 报 ，中 才 被 。

21.5.3.1 接收中断 (RXI)

TWAI FIFO 中 报 时 (TWAI_RX_MESSAGE_CNT_REG > 0)，RXI。TWAI_RX_MESSAGE_CNT_REG 中记录 报 包括 FIFO 中 报 报 。TWAI_RELEASE_BUF 报 ，RXI 才 。

21.5.3.2 发送中断 (TXI)

，报 中 时，TXI。 ，变 ，时 TXI :

- 报 (,)。 败 重 。
- 单 (TWAI_TX_COMPLETE)。
- TWAI_ABORT_TX 报 。

21.5.3.3 错误报警中断 (EWI)

TWAI_STATUS_REG 中 TWAI_ERR_ST TWAI_BUS_OFF_ST 已变时 (, 0 变 1)，EWI。EWI 时 TWAI_ERR_ST TWAI_BUS_OFF_ST :

- TWAI_ERR_ST = 0 TWAI_BUS_OFF_ST = 0:

  - TWAI , 表 TEC REC TWAI_ERR_WARNING_LIMIT_REG 设 。
  - TWAI , 表 时 。
- TWAI_ERR_ST = 1 TWAI_BUS_OFF_ST = 0：表 TEC REC 超 TWAI_ERR_WARNING_LIMIT_REG 设 。
- TWAI_ERR_ST = 1 TWAI_BUS_OFF_ST = 1：表 TWAI BUS_OFF (TEC >= 256)。
- TWAI_ERR_ST = 0 TWAI_BUS_OFF_ST = 1：表 BUS_OFF 期间，TWAI TEC TWAI_ERR_WARNING_LIMIT_REG 设 。

21.5.3.4 数据溢出中断 (DOI)

中 报 时， 报 报 ， DOI。

21.5.3.5 被动错误中断 (TXI)

TWAI 变 被 ， 时， EPI。

21.5.3.6 仲裁丢失中断 (ALI)

TWAI 尝试 报 裁时， ALI。TWAI 裁 bit 记录 裁 捕 (TWAI_ARB LOST CAP_REG) 中。 裁 捕 被 (CPU) ， 不 记录 裁 败时 bit 。

21.5.3.7 总线错误中断 (BEI)

TWAI TWAI 上 测 时， BEI。 时， 时 bit 记录 捕 (TWAI_ERR_CODE_CAP_REG) 中。 捕 被 (CPU) ， 不 记录 。

21.5.4 发送缓冲器与接收缓冲器

21.5.4.1 缓冲器概述

表 21-8. SFF 与 EFF 的缓冲器布局

标 (SFF)		扩 (EFF)	
TWAI		TWAI	
0x40	TX/RX	0x40	TX/RX
0x44	TX/RX identifier 1	0x44	TX/RX identifier 1
0x48	TX/RX identifier 2	0x48	TX/RX identifier 2
0x4c	TX/RX data byte 1	0x4c	TX/RX identifier 3
0x50	TX/RX data byte 2	0x50	TX/RX identifier 4
0x54	TX/RX data byte 3	0x54	TX/RX data byte 1
0x58	TX/RX data byte 4	0x58	TX/RX data byte 2
0x5c	TX/RX data byte 5	0x5c	TX/RX data byte 3

标 (SFF)		扩 (EFF)	
TWAI		TWAI	
0x60	TX/RX data byte 6	0x60	TX/RX data byte 4
0x64	TX/RX data byte 7	0x64	TX/RX data byte 5
0x68	TX/RX data byte 8	0x68	TX/RX data byte 6
0x6c	保	0x6c	TX/RX data byte 7
0x70	保	0x70	TX/RX data byte 8

表 21-8 布 。 ，
TWAI 操 时才可 。CPU 写 操 ，CPU 操
。

TWAI 报 。CPU 写 操 ， 报 、
、 ID ( )。一 ，CPU TWAI_CMD_REG 中
TWAI_TX_REQ 1， 始报 。

- ，变 TWAI_SELF_RX_REQ 1。
- 单 ， 时 TWAI_TX_REQ TWAI_ABORT_TX 1。

FIFO 中 一 报 。CPU 中 操 ， 一 报
、 、 ID ( )。 中 报 ，CPU
TWAI_CMD_REG 中 TWAI_RELEASE_BUF 1 ， FIFO 中 报
，按 报 报 。

21.5.4.2 帧信息

长 1-byte， 报 、 长 。 表 21-9
。

表 21-9. TX/RX 帧信息 (SFF/EFF)；TWAI 地址 0x40

Bit 31-8	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
保	FF	RTR	X	X	XDLC.3	DLC.2	DLC.1	DLC.0

说明：

- FF: 报 EFF SFF。 FF 1 时， 报 EFF， FF 0 时， 报 SFF。
- RTR: 报 。 RTR 1 时， 报 ， RTR 0 时，
  报 。
- DLC: 中 ， 中 。TWAI
  8 个 ， DLC 0 ~ 8。
- X: bit， 可 。

21.5.4.3 帧标识符

报 SFF， 标识 2-bytes (11-bits)； 报 EFF， 标识 4-bytes
(29-bits)。

表 Table 21-10-21-11 SFF (11-bits) 报 标识 。

表 21-10. TX/RX 标识符 1 (SFF); TWAI 地址 0x44

Bit 31-8	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
保	ID.28	ID.27	ID.26	ID.25	ID.24	ID.23	ID.22	ID.21

表 21-11. TX/RX 标识符 2 (SFF); TWAI 地址 0x48

Bit 31-8	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
保	ID.20	ID.19	ID.18	X[1]	X[2]	X[2]	X[2]	X[2]

表 21-12-21-15 EFF (29-bits) 报 标识 。

表 21-12. TX/RX 标识符 1 (EFF); TWAI 地址 0x44

Bit 31-8	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
保	ID.28	ID.27	ID.26	ID.25	ID.24	ID.23	ID.22	ID.21

表 21-13. TX/RX 标识符 2 (EFF); TWAI 地址 0x48

Bit 31-8	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
保	ID.20	ID.19	ID.18	ID.17	ID.16	ID.15	ID.14	ID.13

表 21-14. TX/RX 标识符 3 (EFF); TWAI 地址 0x4c

Bit 31-8	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
保	ID.12	ID.11	ID.10	ID.9	ID.8	ID.7	ID.6	ID.5

表 21-15. TX/RX 标识符 4 (EFF); TWAI 地址 0x50

Bit 31-8	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
保	ID.4	ID.3	ID.2	ID.1	ID.0	X[1]	X[2]	X[2]

21.5.4.4 帧数据

包 , 0 ~ 8 bytes。 中 DLC 。 , DLC 8, 8。 中不包 , 不 。

比 , 5 个 时, CPU DLC 中写 5, 并 写 1 ~ 5 。 , DLC 5 时, 1 ~ 5 中包 CPU 可 。

21.5.5 接收 FIFO 和数据溢出

FIFO 一个 64-bytes 部 , 报 。一 报 可 FIFO 中 3 ~ 13 bytes 间, 中 顺 。 被

FIFO 中 一 报 。

TWAI 一 报 时，TWAI_RX_MESSAGE_COUNTER 1， 64。FIFO 中 间，报 被写 FIFO 中。 中 ，TWAI_RELEASE_BUF 1， FIFO 一 报 间，TWAI_RX_MESSAGE_COUNTER 1。 ， FIFO 中 一 报 。

TWAI 一 报 ， FIFO 间 报 时（不 报 FIFO 中 间， FIFO ），FIFO 部 报 标记 。 报 TWAI_RX_MESSAGE_COUNTER 64。

FIFO 中 报 ， 重 时，TWAI_RELEASE_BUF， TWAI_RX_MESSAGE_COUNTER 0。 可 FIFO 中 报 ，并 报 。

21.5.6 接收滤波器

波 TWAI 报 ID 报（ 时可 报 一个 ）。 报 才 FIFO 中。 波 可 一 TWAI （ ，可 FIFO 中 ），TWAI 操 一 部 报 。

TWAI 时，才可 波 ， / 间 。

波 32-bit Code 32-bit Mask 。Code 一 ，报 中 必 ，才 报 。Mask 可屏蔽 Code 中 （ 屏蔽 设 “不 ” ）。 21-7 ， 报 ， 报 ID 必 Code 设 被 Mask 屏蔽。

图 21-7. 接收滤波器

TWAI 波 32-bit Code Mask 单个 波 (单 波)， 个 波 （ 波 ）。 波 32-bit code mask ， 波 报 （ ，SFF EFF）。

21.5.6.1 单滤波模式

TWAI_RX_FILTER_MODE 1，可 单 波 。 ，32-bit code/mask 单个 波 。

单个 波 可 中 ：

- SFF
  - 11-bit ID
  - RTR bit

- 1 2

- EFF
  - 29-bit ID
  - RTR bit

21-8 单 波 32-bit code/mask 。

图 21-8. 单滤波模式

21.5.6.2 双滤波模式

TWAI_RX_FILTER_MODE 0， 可 波 。 ，32-bit code/mask 个 波 一， 波 1 波 2。 波 ， 报 个 波 中 一个， 表 报 。

个 波 可 中 ：

- SFF
  - 11-bit ID
  - RTR bit
  - 1 (波 1)
- EFF
  - 29-bit ID 16-bit

21-9 波 32-bit code/mask 。

21.5.7 错误管理

TWAI 个 TWAI 中 包 (TEC) (REC)。 个 TWAI (， 、被 、)。TWAI TEC REC 别 TWAI_TX_ERR_CNT_REG TWAI_RX_ERR_CNT_REG 中，CPU 可 时 。 ，

图 21-9. 双滤波模式

TWAI 报 (EWL) ，个 可 TWAI 被 ，
重 。

TWAI ， ：TEC、REC、TWAI_ERR_ST
TWAI_BUS_OFF_ST。 变化 中 ， 而 变化 (可参见
21.5.3)。 21-10 、 中 间 。

图 21-10. 错误状态变化

21.5.7.1 错误报警限制

报 (EWL) TEC REC 可 , 超 , EWI 中 。EWL 一个报 重 TWAI , TWAI 被 被 。EWL TWAI_ERR_WARNING_LIMIT_REG 中 , 时 TWAI 必 。 TWAI_ERR_WARNING_LIMIT_REG 认 96。

TEC / REC EWL 时，TWAI_ERR_ST 被 1。 , TEC REC EWL 时，TWAI_ERR_ST 0。 TWAI_ERR_ST (TWAI_BUS_OFF_ST) 变化 便 报 中 。

21.5.7.2 被动错误

TEC REC 127 时，TWAI 被 。 , TEC REC 127 时，TWAI 。 TWAI 变 被 , 时， 被 中 。

21.5.7.3 离线状态与离线恢复

TEC 255 时，TWAI 。 ，TWAI :

- REC 0
- TEC 127
- TWAI_BUS_OFF_ST 1
- TWAI_BUS_OFF_ST (TWAI_ERR_ST) 变化时， 报 中 。

，TWAI 必 。启 ， 操 。 TWAI 上 测 128 11 个 。

一 TWAI 测 11 个 时，TEC , 。 (TEC 127 0)，TWAI_BUS_OFF_ST 0， 而 报 中 。

21.5.8 错误捕捉

捕 (ECC) TWAI 记录 TWAI bit 。 测 一个 TWAI 时， 中 被 , 记录 TWAI_ERR_CODE_CAP_REG 中。 TWAI_ERR_CODE_CAP_REG 中 被 , 中 时， 不 记录 。

表 21-16 TWAI_ERR_CODE_CAP_REG 中 :

表 21-16. TWAI_ERR_CODE_CAP_REG 的 bit 信息; TWAI 地址 0x30

Bit 31-8	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
保	ERRC.1	ERRC.0	DIR	SEG.4	SEG.3	SEG.2	SEG.1	SEG.0

说明：

- (ERRC)：表 。00 表 ，01 表 ，10 表 ，11 表 。
- (DIR)：表 时，TWAI 。0 表 ，1 表 。
- (SEG)：表 TWAI 报 个 。

表 21-17 SEG.0 ~ SEG.4 。

表 21-17. SEG.4 - SEG.0 的位信息

Bit SEG.4	Bit SEG.3	Bit SEG.2	Bit SEG.1	Bit SEG.0	
0	0	0	1	1	始
0	0	0	1	0	ID.28 ~ ID.21
0	0	1	1	0	ID.20 ~ ID.18
0	0	1	0	0	bit SRTR
0	0	1	0	1	bit IDE
0	0	1	1	1	ID.17 ~ ID.13
0	1	1	1	1	ID.12 ~ ID.5
0	1	1	1	0	ID.4 ~ ID.0
0	1	1	0	0	bit RTR
0	1	1	0	1	保 1
0	1	0	0	1	保 0
0	1	0	1	1	长
0	1	0	1	0	
0	1	0	0	0	CRC
1	1	0	0	0	CRC
1	1	0	0	1	认 槽
1	1	0	1	1	认
1	1	0	1	0	
1	0	0	1	0	间
1	0	0	0	1	标
1	0	1	1	0	被 标
1	0	0	1	1	
1	0	1	1	1	
1	1	1	0	0	标

说明：

- Bit SRTR: 标 RTR bit。
- Bit IDE: 标识 扩 。0 表 标 。

21.5.9 仲裁丢失捕捉

裁 捕 (ALC) TWAI 记录 裁 bit 。TWAI 裁时，bit 被记录 TWAI_ARB_LOST_CAP_REG 中，时 裁 中 。

裁 中 时，bit 不 被记录 TWAI_ARB_LOST_CAP_REG 中，

TWAI_ERR_CODE_CAP_REG 中 裁 捕 被 。

表 21-18 TWAI_ERR_CODE_CAP_REG 中 ; 21-11 一 TWAI 报 bit 。

表 21-18. TWAI_ARB LOST CAP_REG 中的位信息; TWAI 地址 0x2c

Bit 31-5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
保	BITNO.4	BITNO.3	BITNO.2	BITNO.1	BITNO.0

说明：

- (BITNO)：表 裁 TWAI 报 n 个 。

图 21-11. 丢失仲裁的 bit 位置

21.6 寄存器列表

配置寄存器			
TWAI_MODE_REG		0x3FF6B000	/写
TWAI_BUS_TIMING_0_REG	时 0	0x3FF6B018	| /写
TWAI_BUS_TIMING_1_REG	时 1	0x3FF6B01C	| /写
TWAI_ERR_WARNING_LIMIT_REG		0x3FF6B034	| /写
TWAI_DATA_0_REG	0	0x3FF6B040	写| /写
TWAI_DATA_1_REG	1	0x3FF6B044	写| /写
TWAI_DATA_2_REG	2	0x3FF6B048	写| /写
TWAI_DATA_3_REG	3	0x3FF6B04C	写| /写
TWAI_DATA_4_REG	4	0x3FF6B050	写| /写
TWAI_DATA_5_REG	5	0x3FF6B054	写| /写
TWAI_DATA_6_REG	6	0x3FF6B058	写| /写
TWAI_DATA_7_REG	7	0x3FF6B05C	写| /写
TWAI_DATA_8_REG	8	0x3FF6B060	写|
TWAI_DATA_9_REG	9	0x3FF6B064	写|
TWAI_DATA_10_REG	10	0x3FF6B068	写|
TWAI_DATA_11_REG	11	0x3FF6B06C	写|
TWAI_DATA_12_REG	12	0x3FF6B070	写|
TWAI_CLOCK_DIVIDER_REG	时	0x3FF6B07C	不
控制寄存器			
TWAI_CMD_REG		0x3FF6B004	写
状态寄存器			
TWAI_STATUS_REG		0x3FF6B008	

TWAI_ARB_LOST_CAP_REG	裁	0x3FF6B02C	
TWAI_ERR_CODE_CAP_REG	捕	0x3FF6B030	
TWAI_RX_ERR_CNT_REG		0x3FF6B038	| /写
TWAI_TX_ERR_CNT_REG		0x3FF6B03C	| /写
TWAI_RX_MESSAGE_CNT_REG		0x3FF6B074	
中断寄存器			
TWAI_INT_RAW_REG	中	0x3FF6B00C	
TWAI_INT_ENA_REG	中	0x3FF6B010	/写

寄存器

3 F H J T U F S 5 8 " * @ . 0 % & @ 3 & (Y

5 8 " * @ 3 & 4 & 5 @ . 0 % & TWAI 操 。1: ; 0: 操 (/写)

5 8 " * @ - * 4 5 & / @ 0 / - : @ . 0 1 % & , 上 , 不产 , 不 。(/写)

5 8 " * @ 4 & - ' @ 5 & 4 5 @ . 0 1 % & 测 , 启 。 常 测试 个 。(/写)

5 8 " * @ 3 9 @ ' * - 5 & 3 @ . 0 % & 波 。0: 波 ; 1: 单 波 (/写)

3 F H J T U F S 5 8 " * @ # 6 4 @ 5 * . * / (@ @ 3 & (Y

5 8 " * @ # " 6 % @ 1 3 & 4 $, 比 。(| /写)

5 8 " * @ 4 : / $ @ + 6 . 1 @ 8 * 步 5) (SJW), 1 ~ 4 个时间 。(| /写)

3FHJTUFS 58"*@#64@5*.*/(@ @3&(Y $

(reserved)	TWAI_TIME_SAMP	TWAI_TIME_SEG2	TWAI_TIME_SEG1
jR … 3	d	e … 9	j … y
0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0	0	0x0	0x0

_2b2i

58"*@5*.&@4&(时期 1 。(| /写)

58"*@5*.&@4&(时期 2 。(| /写)

58"*@5*.&@4采1 。0: 采 1 ; 1: 采 三 (| /写)

3FHJTUFS 58"*@&33@8"3/*/(@-*.*5@3&(Y

(reserved)	TWAI_ERR_WARNING_LIMIT
jR … 3	d … y
0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0	0x60

_2b2i

58"*@&33@8"3/*/(@-*.报5 , 一 超
时, 报 中 ( )。(| /写)

3FHJTUFS 58"*@%"5"@ @3&(Y

(reserved)	TWAI_TX_BYTE_0 | TWAI_ACCEPTANCE_CODE_0
jR … 3	d … y
0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0	0x0

_2b2i

58"*@59@#:5&@ , 0个 。(写)

58"*@"$$&15"/$&@$0%&@ , 波编 0个 。(/写)

3 F H J T U F S 5 8 " * @ % " 5 " @ @ 3 & (Y

(reserved)	TWAI_TX_BYTE_1 | TWAI_ACCEPTANCE_CODE_1
j R … 3	d … y
0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0	0x0

_ 2 b 2 i

5 8 " * @ 5 9 @ # : 5 & @ , 1 个 。(写)

5 8 " * @ " $ $ & 1 5 " / $ & @ $ 0 % & @ , 波编 1 个 。(/写)

3 F H J T U F S 5 8 " * @ % " 5 " @ @ 3 & (Y

(reserved)	TWAI_TX_BYTE_2 | TWAI_ACCEPTANCE_CODE_2
j R … 3	d … y
0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0	0x0

_ 2 b 2 i

5 8 " * @ 5 9 @ # : 5 & @ , 2 个 。(写)

5 8 " * @ " $ $ & 1 5 " / $ & @ $ 0 % & @ , 波编 2 个 。(/写)

3 F H J T U F S 5 8 " * @ % " 5 " @ @ 3 & (Y $

(reserved)	TWAI_TX_BYTE_3 | TWAI_ACCEPTANCE_CODE_3
j R … 3	d … y
0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0	0x0

_ 2 b 2 i

5 8 " * @ 5 9 @ # : 5 & @ , 3 个 。(写)

5 8 " * @ " $ $ & 1 5 " / $ & @ $ 0 % & @ , 波编 3 个 。(/写)

3 F H J T U F S 5 8 " * @ % " 5 " @ @ 3 & (Y

(reserved)	TWAI_TX_BYTE_4 | TWAI_ACCEPTANCE_MASK_0
j R … 3	d … y
0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0	0x0

_ 2 b 2 i

5 8 " * @ 5 9 @ # : 5 & @ , 4 个 。(写)

5 8 " * @ " $ $ & 1 5 " / $ & @ . " 4 , @ , 波编 0 个 。(/写)

3 F H J T U F S 5 8 " * @ % " 5 " @ @ 3 & (Y

5 8 " * @ 5 9 @ # : 5 & @ , 5 个 。(写)

5 8 " * @ " $ $ & 1 5 " / $ & @ . " 4 , @ , 波编 1 个 。(/写)

3 F H J T U F S 5 8 " * @ % " 5 " @ @ 3 & (Y

5 8 " * @ 5 9 @ # : 5 & @ , 6 个 。(写)

5 8 " * @ " $ $ & 1 5 " / $ & @ . " 4 , @ , 波编 2 个 。(/写)

3 F H J T U F S 5 8 " * @ % " 5 " @ @ 3 & (Y $

TWAI_TX_BYTE_7 | TWAI_ACCEPTANCE_MASK_3

(reserved)

j R	3	d	y
0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0		0x0	

_ 2 b 2 i

5 8 " * @ 5 9 @ # : 5 & @ , 7 个 。(写)

5 8 " * @ " $ $ & 1 5 " / $ & @ . " 4 , @ , 波编 3 个 。(/写)

3 F H J T U F S 5 8 " * @ % " 5 " @ @ 3 & (Y

TWAI_TX_BYTE_8

(reserved)

j R	3	d	y
0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0		0x0	

_ 2 b 2 i

5 8 " * @ 5 9 @ # : 5 & @ , 8 个 。(写)

3 F H J T U F S 5 8 " * @ % " 5 " @ @ 3 & (Y

TWAI_TX_BYTE_9

(reserved)

j R	3	d	y
0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0		0x0	

_ 2 b 2 i

5 8 " * @ 5 9 @ # : 5 & @ , 9 个 。(写)

3 F H J T U F S 5 8 " * @ % " 5 " @ @ 3 & (Y

(reserved)	TWAI_TX_BYTE_10
j R … 3	d … y
0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0	0x0

_ 2 b 2 i

5 8 " * @ 5 9 @ # : 5 操@ , 10 个 。(写)

3 F H J T U F S 5 8 " * @ % " 5 " @ @ 3 & (Y $

(reserved)	TWAI_TX_BYTE_11
j R … 3	d … y
0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0	0x0

_ 2 b 2 i

5 8 " * @ 5 9 @ # : 5 操@ , 11 个 。(写)

3 F H J T U F S 5 8 " * @ % " 5 " @ @ 3 & (Y

(reserved)	TWAI_TX_BYTE_12
j R … 3	d … y
0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0	0x0

_ 2 b 2 i

5 8 " * @ 5 9 @ # : 5 操@ , 12 个 。(写)

3 F H J T U F S 5 8 " * @ $ - 0 $, @ % * 7 * % & 3 @ 3 & (Y $

(reserved)	TWAI_EXT_MODE	(reserved)	TWAI_CLOCK_OFF	TWAI_CD
j R 3	d	e 9	j	k y
0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0	1	0	0	0x0

_ 2 b 2 i

5 8 " * @ $ % 时 CLKOUT 。(/写)

5 8 " * @ $ - 0 $, @ 0 ' ' 可 。1: 闭 CLKOUT 时 ; 0: CLKOUT 时 (| /写)

5 8 " * @ & 9 5 @ . 0 % & 可 , 认 1。0: ; 1: 扩 , CAN 2.0B (| /写)

3 F H J T U F S 5 8 " * @ $. % @ 3 & (Y

(reserved)	TWAI_SELF_RX_REQ	TWAI_CLR_OVERRUN	TWAI_RELEASE_BUF	TWAI_ABORT_TX	TWAI_TX_REQ
j R 8	9	j	k	R	y
0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0	0	0	0	0	0

_ 2 b 2 i

5 8 " * @ 5 9 @ 3 &12 始 。(写)

5 8 " * @ " # 0 3 5 @ 519 始 。(写)

5 8 " * @ 3 & - & " 4 & @ #16 ' 。(写)

5 8 " * @ $ - 3 @ 0 7 & 3 3 6 / 。(写)

5 8 " * @ 4 & - ' @ 3 9 @ 3 & 2 。 1 时 。(写)

3FHJTUFS 58"*@45"564@3&(Y

(reserved)	TWAI_BUS_OFF_ST	TWAI_ERR_ST	TWAI_TX_ST	TWAI_RX_ST	TWAI_TX_COMPLETE	TWAI_TX_BUF_ST	TWAI_OVERRUN_ST	TWAI_RX_BUF_ST	
jR 3	d	e	8	9	j	k	R	y	
0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0	0	0	0	0	1	1	0	0	_2b2i

58"*@39@#6'@451, 表 中 不 , 一个 包。( )

58"*@07&336/@45 1, 表 FIFO 中 , 产 。( )

58"*@59@#6'@451, 表 , 号 。( )

58"*@59@$0.1-&5& 1, 表 上 一个 包。( )

58"*@39@45 1, 表 上 。( )

58"*@59@45 1, 表 上 。( )

58"*@&33@45 1, 表 中 一个 TWAI_ERR_WARNING_LIMIT_REG 中 ( )

58"*@#64@0''@45 1, 表 , 不 上 。( )

3FHJTUFS 58"*@"3#-045$"1@3&(Y $

(reserved)	TWAI_ARB_LOST_CAP	
jR 8	9 y	
0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0	0x0	_2b2i

58"*@"3#@-045@"1 裁 bit 。( )

3 F H J T U F S 5 8 " * @ & 3 3 @ $ 0 % & @ $ " 1 @ 3 & (Y

(reserved)	TWAI_ECC_TYPE	TWAI_ECC_DIRECTION	TWAI_ECC_SEGMENT
j R … 3	d … e	8	9 … y
0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0	0x0	0	0x0

_ 2 b 2 i

5 8 " * @ & $ $ @ 4 & (记录 & 5 , 见表 21-16。( )

5 8 " * @ & $ $ @ % * 3 & 记录 $ * 0 / 时 。1: 时 ; 0: 时 ( )

5 8 " * @ & $ $ @ 5 记录 & 别: 00: ; 01: ; 10: ; 11: - ( )

3 F H J T U F S 5 8 " * @ 3 9 @ & 3 3 @ $ / 5 @ 3 & (Y

(reserved)	TWAI_RX_ERR_CNT
j R … 3	d … y
0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0	0x0

_ 2 b 2 i

5 8 " * @ 3 9 @ & 3 3 @ $ / 5 , 变化 。(| /写)

3 F H J T U F S 5 8 " * @ 5 9 @ & 3 3 @ $ / 5 @ 3 & (Y $

(reserved)	TWAI_TX_ERR_CNT
j R … 3	d … y
0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0	0x0

_ 2 b 2 i

5 8 " * @ 5 9 @ & 3 3 @ $ / 5 , 变化 。(| /写)

3FHJTUFS 58"*@39@.&44"(&@$/5@3&(Y

(reserved)	TWAI_RX_MESSAGE_COUNTER
jR d	e y
0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0	0x0

_2b2i

58"*@39@.&44"(&@$06/5&3 FIFO 中 包 个 。( )

3FHJTUFS 58"*@*/5@3"8@3&(Y $

(reserved)	TWAI_BUS_ERR_INT_ST	TWAI_ARB_LOST_INT_ST	TWAI_ERR_PASSIVE_INT_ST	(reserved)	TWAI_OVERRUN_INT_ST	TWAI_ERR_WARN_INT_ST	TWAI_TX_INT_ST	TWAI_RX_INT_ST
jR 3	d	e	8	9	j	k	R	y
0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0	0	0	0	0	0	0	0	0

_2b2i

58"*@39@*/5@45 中 。 1，表 FIFO 不 ， 。( )

58"*@59@*/5@45 中 。 1，表 ，可 。( )

58"*@&33@8"3/@*/5@45 中 。 1，表 中 变化(0 变 1 1 变 0)。( )

58"*@07&336/@*/5@45 中 。 1，表 中 报 报 。( )

58"*@&33@1"44*7&@*/5@45 中。 1，表 变化 被 间 。( )

58"*@"3#@-045@*/5@45 中 。 1，表 裁。( )

58"*@#64@&33@*/5@45 中。 1，表 测 上 。( )

3 F H J T U F S 5 8 " * @ * / 5 & / " @ 3 & (Y

| (reserved) | | |

22 AES 加速器 (AES)

22.1 概述

&41 "&4(标) , 比 "&4 , "&4
。"&4 '*14 16# 标 , "&4 , "&4 , "&4 。

22.2 主要特性

t "&4

t "&4

t "&4

t 本

22.3 功能描述

22.3.1 运算模式

"&4 "&4 。 "&4@.0%&@3&(不 。
"&4@.0%&@3&(表 。

表 22-1. 运算模式

"&4@.0%&@3&(< >	
	"&4
	"&4
	"&4
	"&4
	"&4
	"&4

22.3.2 密钥、明文、密文

"&4@,&@@3&(。 个 , 个 。 "&4
, "&4@,&:@ @3&4(@,&:@ @3&("&4 ,
"&4@,&:@ @3&4(@,&:@ @3&("&4 ,
"&4@,&:@ @3&4(@,&:@ @3&(

"&4@5&N@@@&(。 个 , 个 。
"&4 , 始 始 "&4@5&N@@@&(, "&4
把 "&4@5&N@@@&("&4 , 始
始 "&4@5&N@@@&(, "&4 把
"&4@5&N@@@&(

22.3.3 字节序

密钥字节序

"&4@&/%*"/@3#&J(U #JU , 表 、表 、表 。表
X<~>X<、表 X<~>X<、表 X<~>X< > 标 '*14 16# “ ,FZ

&YQBOTJPOUIF GJSTU /L XPSET PG UIF FLY QBOXE<FLE XL<FZ 个 XPSE 个 。
三 表表 不 , "&4@,&@@3&(个 “UIF GJSTU /L XPSET PG
FYQBOEF.E LFZ

文本字节序

"&4@&/%*"/@3#&J(U #JU 本 , #JU #JU 本 。
本 "&4 时 , "&4 时 。 本
"&4 时 , "&4 时 。 表 。表
4UBUF '*14 16# “ 5IF 4UBUF“UIF "&4 BMHPSJUIN T PQFSBUJPOT
PO B UPX EJNFOTJPOBM BSSBZ”PG 表表UFT DBMMFE UIF 4UBUF
"&4@5&95@ @3&(个 4UBUF

表 22-2. AES 文本字节序

"&4@&/%*"/@3&(<	"&4@&/%*"/@3&(<	1MBJOUFYU $JQIFSUFYU					
		4UBUF		D			
		S		"&4@5&95@ @3&(	<"&4@5&95@ @3&(	<"&4@5&95@ @3&(	<"&4@5&95@ @3&(< >
				"&4@5&95@ @3&(	<"&4@5&95@ @3&(	<"&4@5&95@ @3&(	<"&4@5&95@ @3&(< >
				"&4@5&95@ @3&(	<"&4@5&95@ @3&(	<"&4@5&95@ @3&(	<"&4@5&95@ @3&(< >
				"&4@5&95@ @3&(	<"&4@5&95@ @3&(	<"&4@5&95@ @3&(	<"&4@5&95@ @3&(< >
		4UBUF		D			
		S		"&4@5&95@ @3&(	<"&4@5&95@ @3&(	<"&4@5&95@ @3&(	<"&4@5&95@ @3&(< >
				"&4@5&95@ @3&(	<"&4@5&95@ @3&(	<"&4@5&95@ @3&(	<"&4@5&95@ @3&(< >
				"&4@5&95@ @3&(	<"&4@5&95@ @3&(	<"&4@5&95@ @3&(	<"&4@5&95@ @3&(< >
				"&4@5&95@ @3&(	<"&4@5&95@ @3&(	<"&4@5&95@ @3&(	<"&4@5&95@ @3&(< >
		4UBUF		D			
		S		"&4@5&95@ @3&(	<"&4@5&95@ @3&(	<"&4@5&95@ @3&(	<"&4@5&95@ @3&(< >
				"&4@5&95@ @3&(	<"&4@5&95@ @3&(	<"&4@5&95@ @3&(	<"&4@5&95@ @3&(< >
				"&4@5&95@ @3&(	<"&4@5&95@ @3&(	<"&4@5&95@ @3&(	<"&4@5&95@ @3&(< >
				"&4@5&95@ @3&(	<"&4@5&95@ @3&(	<"&4@5&95@ @3&(	<"&4@5&95@ @3&(< >
		4UBUF		D			
		S		"&4@5&95@ @3&(	<"&4@5&95@ @3&(	<"&4@5&95@ @3&(	<"&4@5&95@ @3&(< >
				"&4@5&95@ @3&(	<"&4@5&95@ @3&(	<"&4@5&95@ @3&(	<"&4@5&95@ @3&(< >
				"&4@5&95@ @3&(	<"&4@5&95@ @3&(	<"&4@5&95@ @3&(	<"&4@5&95@ @3&(< >
				"&4@5&95@ @3&(	<"&4@5&95@ @3&(	<"&4@5&95@ @3&(	<"&4@5&95@ @3&(< >

表 22-3. AES-128 密钥字节序

AES_ENDIAN_REG[1]	AES_ENDIAN_REG[0]	Bit	w[0]	w[1]	w[2]	w[3]
0	0	[31:24]	AES_KEY_3_REG[31:24]	AES_KEY_2_REG[31:24]	AES_KEY_1_REG[31:24]	AES_KEY_0_REG[31:24]
		[23:16]	AES_KEY_3_REG[23:16]	AES_KEY_2_REG[23:16]	AES_KEY_1_REG[23:16]	AES_KEY_0_REG[23:16]
		[15:8]	AES_KEY_3_REG[15:8]	AES_KEY_2_REG[15:8]	AES_KEY_1_REG[15:8]	AES_KEY_0_REG[15:8]
		[7:0]	AES_KEY_3_REG[7:0]	AES_KEY_2_REG[7:0]	AES_KEY_1_REG[7:0]	AES_KEY_0_REG[7:0]
0	1	[31:24]	AES_KEY_3_REG[7:0]	AES_KEY_2_REG[7:0]	AES_KEY_1_REG[7:0]	AES_KEY_0_REG[7:0]
		[23:16]	AES_KEY_3_REG[15:8]	AES_KEY_2_REG[15:8]	AES_KEY_1_REG[15:8]	AES_KEY_0_REG[15:8]
		[15:8]	AES_KEY_3_REG[23:16]	AES_KEY_2_REG[23:16]	AES_KEY_1_REG[23:16]	AES_KEY_0_REG[23:16]
		[7:0]	AES_KEY_3_REG[31:24]	AES_KEY_2_REG[31:24]	AES_KEY_1_REG[31:24]	AES_KEY_0_REG[31:24]
1	0	[31:24]	AES_KEY_0_REG[31:24]	AES_KEY_1_REG[31:24]	AES_KEY_2_REG[31:24]	AES_KEY_3_REG[31:24]
		[23:16]	AES_KEY_0_REG[23:16]	AES_KEY_1_REG[23:16]	AES_KEY_2_REG[23:16]	AES_KEY_3_REG[23:16]
		[15:8]	AES_KEY_0_REG[15:8]	AES_KEY_1_REG[15:8]	AES_KEY_2_REG[15:8]	AES_KEY_3_REG[15:8]
		[7:0]	AES_KEY_0_REG[7:0]	AES_KEY_1_REG[7:0]	AES_KEY_2_REG[7:0]	AES_KEY_3_REG[7:0]
1	1	[31:24]	AES_KEY_0_REG[7:0]	AES_KEY_1_REG[7:0]	AES_KEY_2_REG[7:0]	AES_KEY_3_REG[7:0]
		[23:16]	AES_KEY_0_REG[15:8]	AES_KEY_1_REG[15:8]	AES_KEY_2_REG[15:8]	AES_KEY_3_REG[15:8]
		[15:8]	AES_KEY_0_REG[23:16]	AES_KEY_1_REG[23:16]	AES_KEY_2_REG[23:16]	AES_KEY_3_REG[23:16]
		[7:0]	AES_KEY_0_REG[31:24]	AES_KEY_1_REG[31:24]	AES_KEY_2_REG[31:24]	AES_KEY_3_REG[31:24]

表 22-4. AES-192 密钥字节序

AES_ENDIAN_REG[1]	AES_ENDIAN_REG[0]	Bit	w[0]	w[1]	w[2]	w[3]	w[4]	w[5] 3 F 8
0	0	[31:24]	AES_KEY_5_REG[31:24]	AES_KEY_4_REG[31:24]	AES_KEY_3_REG[31:24]	AES_KEY_2_REG[31:24]	AES_KEY_1_REG[31:24]	AES_KEY_0_REG[31:24]
		[23:16]	AES_KEY_5_REG[23:16]	AES_KEY_4_REG[23:16]	AES_KEY_3_REG[23:16]	AES_KEY_2_REG[23:16]	AES_KEY_1_REG[23:16]	AES_KEY_0_REG[23:16]
		[15:8]	AES_KEY_5_REG[15:8]	AES_KEY_4_REG[15:8]	AES_KEY_3_REG[15:8]	AES_KEY_2_REG[15:8]	AES_KEY_1_REG[15:8]	AES_KEY_0_REG[15:8]
		[7:0]	AES_KEY_5_REG[7:0]	AES_KEY_4_REG[7:0]	AES_KEY_3_REG[7:0]	AES_KEY_2_REG[7:0]	AES_KEY_1_REG[7:0]	AES_KEY_0_REG[7:0]
0	1	[31:24]	AES_KEY_5_REG[7:0]	AES_KEY_4_REG[7:0]	AES_KEY_3_REG[7:0]	AES_KEY_2_REG[7:0]	AES_KEY_1_REG[7:0]	AES_KEY_0_REG[7:0]
		[23:16]	AES_KEY_5_REG[15:8]	AES_KEY_4_REG[15:8]	AES_KEY_3_REG[15:8]	AES_KEY_2_REG[15:8]	AES_KEY_1_REG[15:8]	AES_KEY_0_REG[15:8]
		[15:8]	AES_KEY_5_REG[23:16]	AES_KEY_4_REG[23:16]	AES_KEY_3_REG[23:16]	AES_KEY_2_REG[23:16]	AES_KEY_1_REG[23:16]	AES_KEY_0_REG[23:16]
		[7:0]	AES_KEY_5_REG[31:24]	AES_KEY_4_REG[31:24]	AES_KEY_3_REG[31:24]	AES_KEY_2_REG[31:24]	AES_KEY_1_REG[31:24]	AES_KEY_0_REG[31:24]
1	0	[31:24]	AES_KEY_0_REG[31:24]	AES_KEY_1_REG[31:24]	AES_KEY_2_REG[31:24]	AES_KEY_3_REG[31:24]	AES_KEY_4_REG[31:24]	AES_KEY_5_REG[31:24]

22.3.4 加密与解密运算

单次运算

始 "&4@.0%&@3&(4@,&@@@3&("&4@5&N9@5@@&("&4@&/%*"/@3&(

"&4@45"35@3&(

"&4@*%-&@3&("&4@*%-&@3&(

"&4@5&N9@5@@&(。

连续运算

, "&4@5&N9@5@@&(被 "&4 , "&4@.0%&@3&("&4@,&@@@3&("&4@&/%*"/@3&(不 变 时可 始操 。

"&4@.0%&@3&(4@,&@@@3&("&4@&/%*"/@3&(, 不 变 。

"&4@5&N9@5@@&(

"&4@45"35@3&(

"&4@*%-&@3&("&4@*%-&@3&(

"&4@5&N9@5@@&(。

22.3.5 运行效率

"&4 一个 ~ 个时 期, 一个 个时 期。

22.4 寄存器列表

配置寄存器			
"&4@.0%&@3&(	0QFSBUJPO NPEF PG UIF "&4 "DDFTT	MFSBUPS	/写
"&4@&/%*"/@3&(	&OEJBO OFTT DPOGJHVSBUJPO SF'HJTUFS		/写
密钥寄存器			
"&4@,&:@ @3&(	"&4 LFZ NBUFSJBM SFHJTUFS	Y ''	/写
"&4@,&:@ @3&(	"&4 LFZ NBUFSJBM SFHJTUFS	Y ''	/写
"&4@,&:@ @3&(	"&4 LFZ NBUFSJBM SFHJTUFS	Y ''	/写
"&4@,&:@ @3&(	"&4 LFZ NBUFSJBM SFHJTUFS	Y '' $	/写
"&4@,&:@ @3&(	"&4 LFZ NBUFSJBM SFHJTUFS	Y ''	/写
"&4@,&:@ @3&(	"&4 LFZ NBUFSJBM SFHJTUFS	Y ''	/写
"&4@,&:@ @3&(	"&4 LFZ NBUFSJBM SFHJTUFS	Y ''	/写
"&4@,&:@ @3&(	"&4 LFZ NBUFSJBM SFHJTUFS	Y '' $	/写
加密／解密数据寄存器			
"&4@5&95@ @3&(	("&4 FODSZQUFE EFDSZQUFE EBUB	'SFHJTUFS	/写
"&4@5&95@ @3&(	("&4 FODSZQUFE EFDSZQUFE EBUB	'SFHJTUFS	/写
"&4@5&95@ @3&(	("&4 FODSZQUFE EFDSZQUFE EBUB	'SFHJTUFS	/写
"&4@5&95@ @3&(	("&4 FODSZQUFE EFDSZQUFE EBUB	'SFH$STUFS	/写
控制／状态寄存器			
"&4@45"35@3&(	"&4 PQFSBUJPO TUBSU DPOUSPM	'SFHJTUFS	/写

ë	í •	¹ ¶	` Ù
" & 4 @ * % - & @ 3 & (	" & 4 J E M F T U B U V T S F H J T U F S	Y ' '	º

寄存器

3FHJTUFS "&4@45"35@3&(Y

SFTFSWFE "&4@45"35

jR R	y
Y	Y

_2b2i

"&4@45"35 "&4 。(写)

3FHJTUFS "&4@*%-&@3&(Y

SFTFSWFE "&4@*%-&

jR R	y
Y	

_2b2i

"&4@*%"-&4 。"&4 时 , 时 。( )

3FHJTUFS "&4@.0%&@3&(Y

SFTFSWFE "&4@.0%&

jR j	k y
Y	

_2b2i

"&4@.0%& "&4 。 表 。(/ 写)

3FHJTUFS "&4@3&&(@ Y O

jR y
Y

_2b2i

"&4@,&@3&("&4 。(/ 写)

3FHJTUFS "&4@5&(5@ Y N

jR y
Y

_2b2i

"&4@5&5@3&(。(/ 写)

23　SHA 加速器 (SHA)

23.1　概述

比　　中单　　4)"（安　　），4)"　　升 4)"　　。4)"
标　'*14 16#　　：4)"、4)"　、4)"　　4)"　。

23.2　主要特性

t　　4)"

t　　4)"

t　　4)"

t　　4)"

23.3　功能描述

23.3.1　填充解析信息

4)"　　一个　。　　个　按　标　“'*14 16#　”中“　1BSTJOH UIF
.FTTBHF　　一个一个　，　　一个　与　　4)"@5&95@&(
4)"、4)"　，　　CJU 与　　4)"@5&95@ @3&(4)"@5&95@ 。@3&(
　　4)"　、4)"　，　　CJU 与
4)"@5&95@ @3&(4)"@5&95@ 。@3&(

4)"　　不　　标　“'*14 16#　”中“　1BEEJOH UIF.FTTBHF 操　；
　　，　　操　。

标　“'*14 16#　”中“　1BSBNFUFS $M_0^{(i)}$ JT UIF MFGU NPTU XPSE”PG NPTT BHF C
　　4)"@5&95@ 中@3&(　　4)"@5&95@ 中@3&(　　中　个
　个 XPSE。

23.3.2　信息摘要

　　，　　被 4)"　　4)"@5&95@&(　　4)"　，
CJU　　4)"@5&95@ @3&(4)"@5&95@。@3&(4)"　，　CJU
　　4)"@5&95@ @3&(4)"@5&95@。@3&(4)"　，　CJU
　　4)"@5&95@ @3&(4)"@5&95@ 。@3&(4)"　，　CJU
　4)"@5&95@ @3&(4)"@5&95@ 。@3&(

标　“'*14 16#　”中“　1BSBNFUFS $H_0^{(N)}$ JT UIF GJOBM IBTI WBMVF BOE JT VTFE
NFTTBHF EJHFTU JUT UIF MFGU NPTU XPSE PG IBTI WBMVF $H_0^{(N)}$
4)"@5&95@ 中@3&(　　中　　个 XPSE $H_1^{(N)}$　　4)"@5&95@ 中@3&(
……

23.3.3　哈希运算

　4)"、4)"　、4)"　　4)"　　一　　；不　　不　　。
　4)"　　4)"@4)" @45"35@&(4)"@4)" @$0/5*/6&@&(4)"@4)" @-0"%@3&(
4)"@4)" @#64:@3&(4)"　　4)"@4)" @45"35@3&(
4)"@4)" @$0/5*/6&@&(4)"@4)" @-0"%@3&(4)"@4)" @#64:@3&(

4)"@4)"　@45"35,@43)&@4)"　@$0/5*/6&@43)&@4)"　@-0"%@3&(
4)"@4)"　@#64:@3&(4)"　4)"@4)"　@45"35,@3&(
4)"@4)"　@$0/5*/6&@43)&@4)"　@-0"%@3&(
4)"@4)"　@#64:@3&操　。

操　一个

B　一个　始　4)"@5&95@3&(

C　4)"@@45"35@3&(。

D　4)"@@#64:@3&(　4)"@@#64:@3&(　。

操　一个

B　始　4)"@5&95@3&(

C　4)"@@$0/5*/6&@3&(。

D　4)"@@#64:@3&(　4)"@@#64:@3&(　。

B　4)"@@-0"%@3&(　。

C　4)"@@#64:@3&(　4)"@@#64:@3&(　。

D　4)"@5&95@3&(　。

23.3.4　运行效率

4)"　个时　期　一个　个时　期
。

23.4　寄存器列表

加密／解密数据寄存器			
4)"@5&95@　@3&(	4)"FODSZQUFE　EFDSZQUFE	EBYU'B　SFHJTUFS	
4)"@5&95@　@3&(	4)"FODSZQUFE　EFDSZQUFE	EBYU'B　SFHJTUFS	
4)"@5&95@　@3&(	4)"FODSZQUFE　EFDSZQUFE	EBYU'B　SFHJTUFS	
4)"@5&95@　@3&(	4)"FODSZQUFE　EFDSZQUFE	EBYU'B　SFHJTUFS	
4)"@5&95@　@3&(	4)"FODSZQUFE　EFDSZQUFE	EBYU'B　SFHJTUFS	
4)"@5&95@　@3&(	4)"FODSZQUFE　EFDSZQUFE	EBYU'B　SFHJTUFS	
4)"@5&95@　@3&(	4)"FODSZQUFE　EFDSZQUFE	EBYU'B　SFHJTUFS	
4)"@5&95@　@3&(	4)"FODSZQUFE　EFDSZQUFE	EBYU'B　SFHJTUFS	
4)"@5&95@　@3&(	4)"FODSZQUFE　EFDSZQUFE	EBYU'B　SFHJTUFS	
4)"@5&95@　@3&(	4)"FODSZQUFE　EFDSZQUFE	EBYU'B　SFHJTUFS	
4)"@5&95@　@3&(	4)"FODSZQUFE　EFDSZQUFE	EBYU'B　SFHJTUFS	
4)"@5&95@　@3&(	4)"FODSZQUFE　EFDSZQUFE	EBYU'B　SFHJTUFS	
4)"@5&95@　@3&(	4)"FODSZQUFE　EFDSZQUFE	EBYU'B　SFHJTUFS	
4)"@5&95@　@3&(	4)"FODSZQUFE　EFDSZQUFE	EBYU'B　SFHJTUFS	
4)"@5&95@　@3&(	4)"FODSZQUFE　EFDSZQUFE	EBYU'B　SFHJTUFS	
4)"@5&95@　@3&(	4)"FODSZQUFE　EFDSZQUFE	EBYU'B　SFHJTUFS	

ë	í•	¹¶	`Ù
4)"@5&95@ @3&(	4)"FODSZQUFE EFDSZQUFE	EBYU'B SFB	BJTUFS
4)"@5&95@ @3&(	4)"FODSZQUFE EFDSZQUFE	EBYU'B SFH	JTUFS
4)"@5&95@ @3&(	4)"FODSZQUFE EFDSZQUFE	EBYU'B SFH	JTUFS
4)"@5&95@ @3&(	4)"FODSZQUFE EFDSZQUFE	EBYU'B SFH	JTUFS
4)"@5&95@ @3&(	4)"FODSZQUFE EFDSZQUFE	EBYU'B SFB	BJTUFS
4)"@5&95@ @3&(	4)"FODSZQUFE EFDSZQUFE	EBYU'B SFH	JTUFS
4)"@5&95@ @3&(	4)"FODSZQUFE EFDSZQUFE	EBYU'B SFH	JTUFS
4)"@5&95@ @3&(	4)"FODSZQUFE EFDSZQUFE	EBYU'B SFH	JTUFS
4)"@5&95@ @3&(	4)"FODSZQUFE EFDSZQUFE	EBYU'B SFB	BJTUFS
4)"@5&95@ @3&(	4)"FODSZQUFE EFDSZQUFE	EBYU'B SFH	JTUFS
4)"@5&95@ @3&(	4)"FODSZQUFE EFDSZQUFE	EBYU'B SFH	JTUFS
4)"@5&95@ @3&(	4)"FODSZQUFE EFDSZQUFE	EBYU'B SFH	JTUFS
4)"@5&95@ @3&(	4)"FODSZQUFE EFDSZQUFE	EBYU'B SFB	BJTUFS
4)"@5&95@ @3&(	4)"FODSZQUFE EFDSZQUFE	EBYU'B SFH	JTUFS
4)"@5&95@ @3&(	4)"FODSZQUFE EFDSZQUFE	EBYU'B SFH	JTUFS
4)"@5&95@ @3&(	4)"FODSZQUFE EFDSZQUFE	EBYU'B SFH	JTUFS
4)"@5&95@ @3&(	4)"FODSZQUFE EFDSZQUFE	EBYU'B SFB	BJTUFS
eÅ ÿ7i			
4)"@4)" @45"35@3&(	$POUSPM SFHJTUFS UP JOJUJBYUF 4)"	PQFSBUJPO	
4)"@4)" @$0/5*/6&@3&(	$POUSPM SFHJTUFS UP DPOUJOVF 4)"	PQFSBUJPO	
4)"@4)" @-0"%@3&(	$POUSPM SFHJTUFS UP DBMDVMBUF UIF GJOBM 4)"	IB	
4)"@4)" @#64:@3&(	4UBUVT SFHJTUFS GPS 4)" PQFSBUJPO		
4)"@4)" @45"35@3&(	$POUSPM SFHJTUFS UP JOJUJBYUF 4)"	PQFSBUJPO	
4)"@4)" @$0/5*/6&@3&(	$POUSPM SFHJTUFS UP DPOUJOVF 4)"	PQFSBUJPO	
4)"@4)" @-0"%@3&(	$POUSPM SFHJTUFS UP DBMDVMBUF UIF GJOBM 4)" IBTI	Y''	
4)"@4)" @#64:@3&(	4UBUVT SFHJTUFS GPS 4)"	PQFSBUJPO	
4)"@4)" @45"35@3&(	$POUSPM SFHJTUFS UP JOJUJBYUF 4)"	PQFSBUJPO	
4)"@4)" @$0/5*/6&@3&(	$POUSPM SFHJTUFS UP DPOUJOVF 4)"	PQFSBUJPO	
4)"@4)" @-0"%@3&(	$POUSPM SFHJTUFS UP DBMDVMBUF UIF GJOBM 4)" IBTI	Y'' "	
4)"@4)" @#64:@3&(	4UBUVT SFHJTUFS GPS 4)"	PQFSBUJPO	
4)"@4)" @45"35@3&(	$POUSPM SFHJTUFS UP JOJUJBYUF 4)#	PQFSBUJPO	
4)"@4)" @$0/5*/6&@3&(	$POUSPM SFHJTUFS UP DPOUJOVF 4)#	PQFSBUJPO	
4)"@4)" @-0"%@3&(	$POUSPM SFHJTUFS UP DBMDVMBUF UIF GJOBM 4)" IBTI	Y'' #	
4)"@4)" @#64:@3&(	4UBUVT SFHJTUFS GPS 4)"	PQFSBUJPO	

7i

3FHJTUFS 4)"@5&95@ Y O

jR y

Y _2b2i

4)"@5&95@3&(4)"• v„W Ø²T7i b •

3FHJTUFS 4)"@4)" @45"35@3&(Y

SFTFSWFE 4)"@4)" @45"35

jR R y

Y _2b2i

4)"@4)" @45"35Æ »Bñ• vÉ› 4)" Øbº•

3FHJTUFS 4)"@4)" @$0/5*/6&@3&(Y

SFTFSWFE 4)"@4)" @$0/5*/6&

jR R y

Y _2b2i

4)"@4)" @$0/5*/6&Æ ªÃ¥• v?ÃÉ› 4)" Øbº•

3FHJTUFS 4)"@4)" @-0"%@3&(Y

SFTFSWFE 4)"@4)" @-0"%

jR R y

Y _2b2i

4)"@4)" @-0"%Æ ²• 4)" Ø 9ØKÔ¥ Ø²Tbº•

3FHJTUFS 4)"@4)" @#64:@3&(Y $

SFTFSWFE

4)"@4)" @#64:

jR R y

Y _2b2i

4)"@4)" @#64:)" Ø ÿ7i 4)"FÎ ž)Ø" b=bº

3FHJTUFS 4)"@4)" @45"35@3&(Y

SFTFSWFE

4)"@4)" @45"35

jR R y

Y _2b2i

4)"@4)" @45"35Æ »Bñ• vÉ› 4)" Øbº•

3FHJTUFS 4)"@4)" @$0/5*/6&@3&(Y

SFTFSWFE

4)"@4)" @$0/5*/6&

jR R y

Y _2b2i

4)"@4)" @$0/5*/6Æ ªÃ¥• v?ÃÉ› 4)" Øbº•

3FHJTUFS 4)"@4)" @-0"%@3&(Y

SFTFSWFE

4)"@4)" @-0"%

jR R y

Y _2b2i

4)"@4)" @-0"%Æ ²• 4)" Ø 9ØKÔ¥ Ø²Tbº•

Register 23.9. SHA_SHA256_BUSY_REG (0x09C)

(reserved)	SHA_SHA256_BUSY
31 … 1	0
0x00000000	0 Reset

SHA_SHA256_BUSY SHA-256 : 1: SHA ; 0: 。( )

Register 23.10. SHA_SHA384_START_REG (0x0A0)

(reserved)	SHA_SHA512_START
31 … 1	0
0x00000000	0 Reset

SHA_SHA384_START 写 1 一个 SHA-384 。(写)

Register 23.11. SHA_SHA384_CONTINUE_REG (0x0A4)

(reserved)	SHA_SHA384_CONTINUE
31 … 1	0
0x00000000	0 Reset

SHA_SHA384_CONTINUE 写 1 SHA-384 。(写)

Register 23.12. SHA_SHA384_LOAD_REG (0x0A8)

(reserved)	SHA_SHA384_LOAD
31 … 1	0
0x00000000	0 Reset

SHA_SHA384_LOAD 写 1 SHA-384 , 。(写)

Register 23.13. SHA_SHA384_BUSY_REG (0x0AC)

(reserved)	SHA_SHA384_BUSY
31 … 1	0
0x00000000	0 Reset

SHA_SHA384_BUSY SHA-384 : 1: SHA ; 0: 。( )

Register 23.14. SHA_SHA512_START_REG (0x0B0)

(reserved)	SHA_SHA512_START
31 … 1	0
0x00000000	0 Reset

SHA_SHA512_START 写 1 一个 SHA-512 。(写)

Register 23.15. SHA_SHA512_CONTINUE_REG (0x0B4)

(reserved)	SHA_SHA512_CONTINUE
31 … 1	0
0x00000000	0 Reset

SHA_SHA512_CONTINUE 写 1 SHA-512 。(写)

Register 23.16. SHA_SHA512_LOAD_REG (0x0B8)

(reserved)	SHA_SHA512_LOAD
31 … 1	0
0x00000000	0 Reset

SHA_SHA512_LOAD 写 1 SHA-512 , 。(写)

Register 23.17. SHA_SHA512_BUSY_REG (0x0BC)

(reserved)	SHA_SHA512_BUSY	
31 … 1	0	
0x00000000	0	Reset

SHA_SHA512_BUSY SHA-512 : 1: SHA ; 0: 。( )

24 RSA 加速器 (RSA)

24.1 概述

RSA RSA 。 包括

、 。

RSA 三 ，并 长 ， 不被浪 。

24.2 主要特性

-
-
-
- 长

24.3 功能描述

24.3.1 初始化

设时 并 DPORT_RSA_PD_CTRL_REG 中 DPORT_RSA_PD 可 并 RSA 。

RSA 被 ， RSA_CLEAN_REG 0， 始化 始。 4 始化 0。 始化 ， RSA_CLEAN_REG 1。 ，RSA 被 一 时， 查 RSA_CLEAN_REG， 保 RSA 可 。

24.3.2 大数模幂运算

$Z = X^Y \bmod M$， Montgomery Multiplication （ ）。 ， X、Y、M ， 个 ， 参 $\overline{r}$ M^0。 个参 。

RSA $N \in \{512, 1024, 1536, 2048, 2560, 3072, 3584, 4096\}$ 8 长 。 Z、X、Y、M $\overline{r}$ 8 中 一 ， 必 ，而 M^0 始 32。

设

$$b = 2^{32}$$

可 个 b 表 ：

$$n = \frac{N}{32}$$

$$Z = (Z_{n-1}Z_{n-2}\cdots Z_0)_b$$

$$X = (X_{n-1}X_{n-2}\cdots X_0)_b$$

$$Y = (Y_{n-1}Y_{n-2}\cdots Y_0)_b$$

$$M = (M_{n-1}M_{n-2}\cdots M_0)_b$$

$$\overline{r} = (\overline{r}_{n-1}\overline{r}_{n-2}\cdots \overline{r}_0)_b$$

中 $Z_{n-1} \sim Z_0$、$X_{n-1} \sim X_0$、$Y_{n-1} \sim Y_0$、$M_{n-1} \sim M_0$、$\bar{r}_{n-1} \sim \bar{r}_0$ 别表 一个 b ， 32。Z_{n-1}、X_{n-1}、Y_{n-1}、M_{n-1}、$\bar{r}_{n-1}$ 别 Z、X、Y、M、$\bar{r}$ b ，而 Z_0、X_0、Y_0、M_0、$\bar{r}_0$ 别 Z、X、Y、M、$\bar{r}$ b 。

设

$$R = b^n$$

参 ：

$$\bar{r} = R^2 \bmod M \tag{1}$$

$$\begin{cases} M^{00} \times M + 1 = R \times R^{-1} \\ M^0 = M^{00} \bmod b \end{cases} \tag{2}$$

((2) 扩 GCD 。)

：

1. RSA_MODEXP_MODE_REG 写 $(\frac{N}{512} - 1)$。
2. X_i、Y_i、M_i、$\bar{r}_i$ $(i \in [0, n) \cap \mathbb{N})$ 别写 RSA_X_MEM、RSA_Y_MEM、RSA_M_MEM、RSA_Z_MEM。 128 word。 一个 word 一个 b 。 ， 。 长 ， 个 中 写 ， 可 。
3. M^0 写 RSA_M_PRIME_REG。
4. RSA_MODEXP_START_REG 写 1。
5. 。 RSA_INTERRUPT_REG 1， RSA_INTR 中 产 。
6. RSA_Z_MEM Z_i $(i \in [0, n) \cap \mathbb{N})$。
7. RSA_INTERRUPT_REG 写 1 中 。

， RSA_MODEXP_MODE_REG 中 长 RSA_Y_MEM 中 Y_i、RSA_M_MEM 中 M_i、 RSA_M_PRIME_REG 中 M^0 不 变化 ， RSA_X_MEM 中 X_i RSA_Z_MEM 中 $\bar{r}_i$ 被 。 时， 必 可。

24.3.3 大数模乘运算

Montgomery Multiplication $Z = X \times Y \bmod M$， 1、 2 $\bar{r}$ M^0。

RSA 8 长 。 采 方 ， 期间 一 。

：

1. RSA_MULT_MODE_REG 写 $(\frac{N}{512} - 1)$。
2. X_i、M_i、$\bar{r}_i$ $(i \in [0, n) \cap \mathbb{N})$ 别写 RSA_X_MEM、RSA_M_MEM、RSA_Z_MEM。 长 ， 个 中 写 ， 可 。

3. M^0 写 RSA_M_PRIME_REG。
4. RSA_MULT_START_REG 写 1。
5. 一 。 RSA_INTERRUPT_REG 1， RSA_INTR 中 产 。
6. RSA_INTERRUPT_REG 写 1 中 。
7. Y_i $(i \in [0, n) \cap \mathbb{N})$ 写 RSA_X_MEM。 长 ， 中 写 ， 不 改。
8. RSA_MULT_START_REG 写 1。
9. 。 RSA_INTERRUPT_REG 1， RSA_INTR 中 产 。
10. RSA_Z_MEM Z_i $(i \in [0, n) \cap \mathbb{N})$。
11. RSA_INTERRUPT_REG 写 1 中 。

， RSA_MULT_MODE_REG 中 长 RSA_M_MEM 中 M_i、 RSA_M_PRIME_REG 中 M^0 被 。 时，可 不 重 写 。

24.3.4 大数乘法运算

$Z = X \times Y$。 中 Z 长 X、Y 长 倍。 RSA 4 $N \in \{512, 1024, 1536, 2048\}$ 长 。 Z 长 $\hat{N}$ $2 \times N$。

常 $\hat{X}$、$\hat{Y}$， Z 长 一 $(\hat{N})$。

$$n = \frac{N}{32}$$

$$\hat{N} = 2 \times N$$

$$\hat{n} = \frac{\hat{N}}{32} = 2n$$

$$\hat{X} = (\hat{X}_{\hat{n}-1}\hat{X}_{\hat{n}-2}\cdots\hat{X}_0)_b = (\underbrace{00\cdots0}_{n}X)_b = (\underbrace{00\cdots0}_{n}X_{n-1}X_{n-2}\cdots X_0)_b$$

$$\hat{Y} = (\hat{Y}_{\hat{n}-1}\hat{Y}_{\hat{n}-2}\cdots\hat{Y}_0)_b = (Y\underbrace{00\cdots0}_{n})_b = (Y_{n-1}Y_{n-2}\cdots Y_0\underbrace{00\cdots0}_{n})_b$$

：

1. RSA_MULT_MODE_REG 写 $(\frac{\hat{N}}{512} - 1 + 8)$。
2. $\hat{X}_i$、$\hat{Y}_i$ $(i \in [0, \hat{n}) \cap \mathbb{N})$ 别写 RSA_X_MEM、RSA_Z_MEM。

   长 ， 个 中 写 ， 可 。 写 b 中 一半 0， 0 不可 。
3. RSA_MULT_START_REG 写 1。
4. 。 RSA_INTERRUPT_REG 1， RSA_INTR 中 产 。
5. RSA_Z_MEM Z_i $(i \in [0, \hat{n}) \cap \mathbb{N})$。
6. RSA_INTERRUPT_REG 写 1 中 。

， RSA_MULT_MODE_REG 中 长 被 改。

24.4 寄存器列表

配置寄存器			
RSA_M_PRIME_REG	Register to store M’	0x3FF02800	/写
模幂运算寄存器			
RSA_MODEXP_MODE_REG	Modular exponentiation mode	0x3FF02804	/写
RSA_MODEXP_START_REG	Start bit	0x3FF02808	写
模乘运算寄存器			
RSA_MULT_MODE_REG	Modular multiplication mode	0x3FF0280C	/写
RSA_MULT_START_REG	Start bit	0x3FF02810	写
其他寄存器			
RSA_INTERRUPT_REG	RSA interrupt register	0x3FF02814	/写
RSA_CLEAN_REG	RSA clean register	0x3FF02818	

24.5 寄存器

Register 24.1. RSA_M_PRIME_REG (0x800)

31 … 0
0x000000000

Reset

RSA_M_PRIME_REG 包 M'。(/写)

Register 24.2. RSA_MODEXP_MODE_REG (0x804)

(reserved) 31 … 3	RSA_MODEXP_MODE 2 … 0
0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0	0 0 0

Reset

RSA_MODEXP_MODE 包 。(/写)

Register 24.3. RSA_MODEXP_START_REG (0x808)

(reserved) 31 … 1	RSA_MODEXP_START 0
0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0	0

Reset

RSA_MODEXP_START 写 1 始 。(写)

Register 24.4. RSA_MULT_MODE_REG (0x80C)

(reserved) 31 … 4	RSA_MULT_MODE 3 … 0
0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0	0 0 0 0

Reset

RSA_MULT_MODE 包 。(写)

Register 24.5. RSA_MULT_START_REG (0x810)

(reserved)	RSA_MULT_START
31 … 1	0
0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0	0
	Reset

RSA_MULT_START 写 1 始 。(写)

Register 24.6. RSA_INTERRUPT_REG (0x814)

(reserved)	RSA_INTERRUPT
31 … 1	0
0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0	0
	Reset

RSA_INTERRUPT RSA 中 。一 1。(/写)

Register 24.7. RSA_CLEAN_REG (0x818)

(reserved)	RSA_CLEAN
31 … 1	0
0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0	0
	Reset

RSA_CLEAN 一 始 , 1。( )

25 随机数发生器 (RNG)

25.1 概述

ESP32 一个 , 32 可 操 。

25.2 主要特性

ESP32 可 如 ,
一 。

25.3 功能描述

可 RNG_DATA_REG 中 , 个 32 ,
声 中 声 步时 。

, 声可 SAR ADC ADC 。 SAR ADC ADC 时,
产 比 , 并 (XOR) 。

图 25-1. 噪声源

SAR ADC 时, 个 RTC8M_CLK (8 MHz) 时 期 (部 RC , 见 3 复位和时钟), 2 。 , , RNG_DATA_REG 时 不超 500 kHz。

ADC 时, 个 APB 时 期 (常 80 MHz) , 2 。 ,
, RNG_DATA_REG 时 不超 5 MHz。

ADC , 5 MHz RNG_DATA_REG 2 GB 本，并 Dieharder 测试 (版本 3.31.1) 本 测试。 , 本 测试。

25.4 编程指南

ESP32 时, SAR ADC ADC, 可 产 ,
避 。 中,

- SAR ADC 可 DIG ADC , 见 29 片上传感器与模拟信号处理 。
- ADC Wi-Fi 启 时 。

说明：

，Wi-Fi 启时，ADC 饱可，。，Wi-Fi 启时，时 DIG ADC1 SAR ADC 产。

时，RNG_DATA_REG，。时，不超25.3。

25.5 寄存器列表

RNG_DATA_REG		0x3FF75144	

25.6 寄存器

Register 25.1. RNG_DATA_REG (0x144)

31	0	
0x000000000		Reset

RNG_DATA_REG 。( )

26 片外存储器加密与解密 (FLASH)

26.1 概述

ESP32 flash。 flash 可 、（比 ）。ESP32 flash ，与 flash。 CPU cache flash 时，flash 。ESP32 flash 安 保 。

26.2 主要特性

-
- 参
- ， 参
- 、 参 启 (boot) flash

26.3 功能描述

图 26-1. Flash 加解密模块架构

Flash 包 三个部 ， 别 (Key Generator)、flash (Flash Encryption) flash (Flash Decryption) ， 26-1 。Key Generator 被 Flash Encryption Flash Decryption 。Flash Encryption Flash Decryption 可 时 。

设 DPort 中 flash DPORT_SLAVE_SPI_CONFIG_REG 中 DPORT_SPI_ENCRYPT_ENABLE DPORT_SPI_DECRYPT_ENABLE 。Flash 设 eFuse 中 6 个 参 ， coding_scheme、BLOCK1、flash_crypt_config、download_dis_encrypt、flash_crypt_cnt download_dis_decrypt。

26.3.1 Key Generator 模块

Key Generator 参 coding_scheme、BLOCK1
$Key_o = f(coding_scheme, BLOCK1)$。

Key Generator 参 flash_crypt_config flash flash $Addr_e$、$Addr_d$ 别
$Key_e = g(Key_o, flash_crypt_config, Addr_e)$、
$Key_d = g(Key_o, flash_crypt_config, Addr_d)$。

参 flash_crypt_config 0 时，Key_e、Key_d flash 。 参 flash_crypt_config 0 时，flash 8 个 word 一个 Key_e Key_d。

26.3.2 Flash Encryption 模块

Flash Encryption 一个 设 ， ， 可 被 CPU 。 、 设 DPort 、 参 、boot 并 一 。

时 参 ， ：

1. DPORT_SLAVE_SPI_CONFIG_REG DPORT_SPI_ENCRYPT_ENABLE 1。
2. 备写 flash FLASH_ENCRYPT_ADDRESS_REG， 必 8-word 。
3. 8-word 中 word FLASH_ENCRYPT_BUFFER_0_REG， word FLASH_ENCRYPT_BUFFER_1_REG， FLASH_ENCRYPT_BUFFER_7_REG。
4. FLASH_ENCRYPT_START_REG 1。
5. FLASH_ENCRYPT_DONE_REG， 1。
6. 时 ， 设 SPI0 flash 8-word 写 8-word 。

步 1 5 操 Flash Encryption 8-word 。 Key_e。 8 个 word。步 6 操 设 SPI0 Flash Encryption 写 flash。步 6 中时 一个参 写 flash ， 个 必 8-word ， 必 步 2 中 FLASH_ENCRYPT_ADDRESS_REG 一 。 步 6 中时 一个参 8-word ， 步 1 5 ， 个参 ， 设 SPI0 时 。 Flash Encryption 不 步 1 5， 步 6 不 而 中 参 。

Flash Encryption ：

- SPI Flash Boot

  DPORT_SLAVE_SPI_CONFIG_REG DPORT_SPI_ENCRYPT_ENABLE 1 时，Flash Encryption ， 。

- Download Boot

  DPORT_SLAVE_SPI_CONFIG_REG DPORT_SPI_ENCRYPT_ENABLE 1 参 download_dis_encrypt 0 时，Flash Encryption ， 。

个 参 ， 被 。 被 写 flash 中。 CPU 可 不 cache 而 flash 而 ，

Key_e。

26.3.3 Flash Decryption 模块

Flash Decryption 并 设 , 不 , 不 被 CPU 。 设 DPort 、 参 、 boot 并 一 。

时, CPU cache flash 中 。Flash Decryption 把 。 个 参 并 cache 。 Key_d, 被 。

Flash Encryption 时, 不 flash 中 产 , , CPU cache flash 中 始 。

Flash Decryption :

- SPI Flash Boot

  参 flash_crypt_cnt（7 bit ）中 有 个 bit 1 时, Flash Decryption ,
  。

- Download Boot

  DPORT_SLAVE_SPI_CONFIG_REG DPORT_SPI_DECRYPT_ENABLE 1 参 download_dis_decrypt 0 时, Flash Decryption , 。

26.4 寄存器列表

FLASH_ENCRYPTION_BUFFER_0_REG	Flash encryption buffer register 0	0x3FF5B000	写
FLASH_ENCRYPTION_BUFFER_1_REG	Flash encryption buffer register 1	0x3FF5B004	写
FLASH_ENCRYPTION_BUFFER_2_REG	Flash encryption buffer register 2	0x3FF5B008	写
FLASH_ENCRYPTION_BUFFER_3_REG	Flash encryption buffer register 3	0x3FF5B00C	写
FLASH_ENCRYPTION_BUFFER_4_REG	Flash encryption buffer register 4	0x3FF5B010	写
FLASH_ENCRYPTION_BUFFER_5_REG	Flash encryption buffer register 5	0x3FF5B014	写
FLASH_ENCRYPTION_BUFFER_6_REG	Flash encryption buffer register 6	0x3FF5B018	写
FLASH_ENCRYPTION_BUFFER_7_REG	Flash encryption buffer register 7	0x3FF5B01C	写
FLASH_ENCRYPTION_START_REG	Encrypt operation control register	0x3FF5B020	写
FLASH_ENCRYPTION_ADDRESS_REG	External flash address register	0x3FF5B024	写
FLASH_ENCRYPTION_DONE_REG	Encrypt operation status register	0x3FF5B028	

26.5 寄存器

Register 26.1. FLASH_ENCRYPTION_BUFFER_*n*_REG (*n*: 0-7) (0x0+4n*)**

31	0	
0x000000000		Reset

FLASH_ENCRYPTION_BUFFER_*n*_REG buffer 。(写)

Register 26.2. FLASH_ENCRYPTION_START_REG (0x020)

(reserved)	FLASH_START
31 … 1	0
0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0	0

Reset

FLASH_START 1,启 操 。(写)

Register 26.3. FLASH_ENCRYPTION_ADDRESS_REG (0x024)

31 … 0
0x000000000

Reset

FLASH_ENCRYPTION_ADDRESS_REG flash , 个 必 8-word 。(写)

Register 26.4. FLASH_ENCRYPTION_DONE_REG (0x028)

(reserved)	FLASH_DONE
31 … 1	0
0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0	0

Reset

FLASH_DONE 操 , 1。( )

27 存储器管理和保护单元 (MMU, MPU)

27.1 概述

ESP32 中 个 设 MMU（ 单 ） MPU（ 保 单 ）被 。 OS ，MPU MMU 可 部 设。MMU 可 ， 上 个 。 可 ， 个 可 。OS 时， 别 一个 （ PID），彼 。 8 个。 个 OS 。

27.2 主要特性

- PRO_CPU APP_CPU 8 个
- MPU/MMU PID 上 、 、 设
- 上 MPU/MMU
- MMU
- 设 MPU

27.3 功能描述

27.3.1 PID 控制器

ESP32 中，PID ，MMU/MPU PID。OS 上 一 时， PID 中 PID。 ，PID 可 测中 并 PID OS PID。

中 个 设 PID ， 别 ESP32 中 个 CPU。 个 CPU 一个 PID ，时 不 CPU 上 不 。

27.3.2 MPU/MMU

MPU MMU 设 上 ， 设。 试 MMU/MPU 保 设时，MMU MPU CPU PID PID。

MMU MPU PID 上 ，并不 个 CPU; ，部 设 MMU MPU 8 个不 PID 。 比 ， MMU 不 识别 PID，而 识别 CPU。 不 APP_CPU PRO_CPU 上，MMU 个 PID 。 中 一个 ， 个 CPU ， 并不 。

PID ，MPU 可 设。MMU ，并 ， 而 可 不 。 ，MMU 可 别 。

27.3.2.1 嵌入式存储器

上 MPU，可 MPU MMU ：

表 27-1. 片上存储器的 MPU 和 MMU 结构

		始		MPU
ROM0	384 KB	0x4000_0000	0x4005_FFFF	Static MPU
ROM1	64 KB	0x3FF9_0000	0x3FF9_FFFF	Static MPU
SRAM0	64 KB 128 KB	0x4007_0000 0x4008_0000	0x4007_FFFF 0x4009_FFFF	Static MPU SRAM0 MMU
SRAM1 ( )	128 KB 128 KB 32 KB	0x3FFE_0000 0x400A_0000 0x4000_0000	0x3FFF_FFFF 0x400B_FFFF 0x4000_7FFF	Static MPU Static MPU Static MPU
SRAM2	72 KB 128 KB	0x3FFA_E000 0x3FFC_0000	0x3FFB_FFFF 0x3FFD_FFFF	Static MPU SRAM2 MMU
RTC FAST ( )	8 KB 8 KB	0x3FF8_0000 0x400C_0000	0x3FF8_1FFF 0x400C_1FFF	RTC FAST MPU RTC FAST MPU
RTC SLOW	8 KB	0x5000_0000	0x5000_1FFF	RTC SLOW MPU

Static MPU

ROM0，ROM1，SRAM0 64 KB，SRAM1 SRAM2 72 KB Static MPU 。 MPU ，不 。 PID 。 PID 0 1 时，可 表 27-1 中 (RAM 时 写) 。 PID 2 7，不可被 。

RTC FAST & RTC SLOW MPU

8 KB RTC FAST Memory 8 KB RTC SLOW Memory 个可 MPU 。 RTC_CNTL_ RTC_PID_CONFIG_REG DPORT_AHBLITE_MPU_TABLE_RTC_REG ，MPU 可 个单 PID 。 设 中 个 ，可 PID 写 ， PID 写 。PID 0 1 始 可 RTC SLOW ， 已 表 27-2 27-3 PID 间 。

表 27-2. 管理 RTC FAST Memory 的 MPU

	边		PID RTC_CNTL_RTC_PID_CONFIG
8 KB	0x3FF8_0000	0x3FF8_1FFF	0 1 2 3 4 5 6 7
8 KB	0x400C_0000	0x400C_1FFF	0 1 2 3 4 5 6 7

表 27-3. 管理 RTC SLOW Memory 的 MPU

	边		PID = 0/1	PID DPORT_AHBLITE_MPU_TABLE_RTC_REG
8 KB	0x5000_0000	0x5000_1FFF	/写	2 3 4 5 6 7 0 1 2 3 4 5

RTC_CNTL_RTC_PID_CONFIG_REG RTC 设 一部 , PID 0 改
DPORT _AHBLITE_MPU_TABLE_RTC_REG 一个 Dport , 可 PID 0 1 改

管理 SRAM0 和 SRAM2 高 128 KB 的 MMU

SRAM0 128 KB SRAM 128 KB MMU 。 MMU 不 可 MPU 一
, 而 CPU 写 ( ) 中可 不
( )。

一 , 片上 RAM MMU 16 。 可 8 KB，4 KB 2 KB。
8 KB 时，16 包 个 128 KB 间; 4 KB 2 KB 时， 间
别 64 KB 96 KB 不 MPU 。 , 可
: MMU 可 。
PID 0 1 , 一 一 。 个 写 被
写 。 PID 0 / 1 操 个
。

PID 2 7 , MMU 可 个 PID，重 个 , 不
。 , 写 可 被 不 写 。
27-1 , CPU (PID 2 7) 尝试 0x3FFC_2345。 1
, 0x0345。MMU 被 PID 1 2
。 , 被重 2 中 , 0x3FFC_4345。
中 8 KB。

图 27-1. MMU 访问示例

表 27-4. 管理片上 SRAM 0 和 SRAM2 剩余 128 KB 的 MMU 页模式

DPORT_IMMU_PAGE_MODE	DPORT_DMMU_PAGE_MODE	
0	0	8 KB
1	1	4 KB
2	2	2 KB

非 MMU 管理的存储器

MMU SRAM0 SRAM2 , 可 8 KB，4 KB 2 KB。 表 27-4 , DPORT_IMMU_PAGE_MODE_REG DPORT_DMMU_PAGE_MODE_REG 中 DPORT_IMMU_PAGE_MODE (SRAM0) DPORT_DMMU_PAGE_MODE (SRAM2) 设 。 个 16， 8 KB 时， 16 间 128 KB; 4 KB 时， 间 64 KB; 2 KB 时， 间 32 KB。

, 8 KB , 个 MMU ; , 128 KB 一部 不 MMU 。 , 4 KB ，不 MMU 0x4009_0000 0x4009_FFFF 0x3FFD_0000 0x3FFD_FFFF; 2 KB ，不 MMU 0x4008_8000 0x4009_FFFF 0x3FFC_8000 0x3FFD_FFFF。 可 PID 0 1 亏; PID 2 7 。

间 中， 布 , ，MMU SRAM0 n $0x40080000 + (pagesize * n)$ $0x40080000 + (pagesize * (n+1) - 1)$; ，MMU SRAM2 n $0x3FFC0000 + (pagesize * n)$ $0x3FFC0000 + (pagesize * (n+1) - 1)$。表 27-5 27-6 不 , MMU SRAM0 SRAM2 。

表 27-5. SRAM0 MMU 页边界地址

	8 KB		4 KB		2 KB	
0	40080000	40081FFF	40080000	40080FFF	40080000	400807FF
1	40082000	40083FFF	40081000	40081FFF	40080800	40080FFF
2	40084000	40085FFF	40082000	40082FFF	40081000	400817FF
3	40086000	40087FFF	40083000	40083FFF	40081800	40081FFF
4	40088000	40089FFF	40084000	40084FFF	40082000	400827FF
5	4008A000	4008BFFF	40085000	40085FFF	40082800	40082FFF
6	4008C000	4008DFFF	40086000	40086FFF	40083000	400837FF
7	4008E000	4008FFFF	40087000	40087FFF	40083800	40083FFF
8	40090000	40091FFF	40088000	40088FFF	40084000	400847FF
9	40092000	40093FFF	40089000	40089FFF	40084800	40084FFF
10	40094000	40095FFF	4008A000	4008AFFF	40085000	400857FF
11	40096000	40097FFF	4008B000	4008BFFF	40085800	40085FFF
12	40098000	40099FFF	4008C000	4008CFFF	40086000	400867FF
13	4009A000	4009BFFF	4008D000	4008DFFF	40086800	40086FFF
14	4009C000	4009DFFF	4008E000	4008EFFF	40087000	400877FF
15	4009E000	4009FFFF	4008F000	4008FFFF	40087800	40087FFF
间	-	-	40090000	4009FFFF	40088000	4009FFFF

表 27-6. SRAM2 MMU 页边界地址

	8 KB		4 KB		2 KB	
0	3FFC0000	3FFC1FFF	3FFC0000	3FFC0FFF	3FFC0000	3FFC07FF
1	3FFC2000	3FFC3FFF	3FFC1000	3FFC1FFF	3FFC0800	3FFC0FFF
2	3FFC4000	3FFC5FFF	3FFC2000	3FFC2FFF	3FFC1000	3FFC17FF
3	3FFC6000	3FFC7FFF	3FFC3000	3FFC3FFF	3FFC1800	3FFC1FFF
4	3FFC8000	3FFC9FFF	3FFC4000	3FFC4FFF	3FFC2000	3FFC27FF
5	3FFCA000	3FFCBFFF	3FFC5000	3FFC5FFF	3FFC2800	3FFC2FFF
6	3FFCC000	3FFCDFFF	3FFC6000	3FFC6FFF	3FFC3000	3FFC37FF
7	3FFCE000	3FFCFFFF	3FFC7000	3FFC7FFF	3FFC3800	3FFC3FFF
8	3FFD0000	3FFD1FFF	3FFC8000	3FFC8FFF	3FFC4000	3FFC47FF
9	3FFD2000	3FFD3FFF	3FFC9000	3FFC9FFF	3FFC4800	3FFC4FFF
10	3FFD4000	3FFD5FFF	3FFCA000	3FFCAFFF	3FFC5000	3FFC57FF
11	3FFD6000	3FFD7FFF	3FFCB000	3FFCBFFF	3FFC5800	3FFC5FFF
12	3FFD8000	3FFD9FFF	3FFCC000	3FFCCFFF	3FFC6000	3FFC67FF
13	3FFDA000	3FFDBFFF	3FFCD000	3FFCDFFF	3FFC6800	3FFC6FFF
14	3FFDC000	3FFDDFFF	3FFCE000	3FFCEFFF	3FFC7000	3FFC77FF
15	3FFDE000	3FFDFFFF	3FFCF000	3FFCFFFF	3FFC7800	3FFC7FFF
间	-	-	3FFD0000	3FFDFFFF	3FFC8000	3FFDFFFF

MMU 映射

SRAM0 MMU SRAM2 MMU, 一 16 个 。
MMU 不 , 个 一个**实地址** , 而不 一个 。 PID
可 个 。表 27-7 。
, PID 2 7 ; PID 0 1 可 写 ,
并 。 设 , PID 0 1 时,
x。 16 个 DPORT_IMMU_PAGE_MODE_REG
DPORT_DMMU_PAGE_MODE_REG PID 0 1 写 。

表 27-7. DPORT_DMMU_TABLE*n*_REG 和 DPORT_IMMU_TABLE*n*_REG

[6:4]	PID 2~ 7
0	PID 2 ~ 7 。
1	PID 2 ~ 7 。
2	PID 2 。
3	PID 3 。
4	PID 4 。
5	PID 5 。
6	PID 6 。
7	PID 7 。

[3:0]	
0x00	0 可 。
0x01	1 可 。
0x02	2 可 。
0x03	3 可 。
0x04	4 可 。
0x05	5 可 。
0x06	6 可 。
0x07	7 可 。
0x08	8 可 。
0x09	9 可 。
0x10	10 可 。
0x11	11 可 。
0x12	12 可 。
0x13	13 可 。
0x14	14 可 。
0x15	15 可 。

SRAM0 和 SRAM2 MMU 的差异

SRAM0 MMU 被 ，而 SRAM2 MMU 。 ， 常 SRAM0 MMU 中， SRAM2 MMU 中。常 ， PID ，PID 2 7 SRAM0 MMU 。而， 必 改 部 ， 写 SRAM2 中 MMU 。 上 ，PID 0 1 始 写 个 SRAM0 SRAM2。

DMA MPU

可 DMA 可 设， 设中 。 DMA， 可 常 ， 常 。 ，DMA MPU 可 DMA 。

上 SRAM1 SRAM2 中 个 8 KB ，MPU DPORT_AHB_MPU_TABLE_*n*_REG 中 个 DMA 。DMA MPU 可 始 DMA ；并不 PID。 OS 分 ， 上 时，OS 重 。

表 27-8 SRAM1 SRAM2 个 8 KB 。 被 1 时， DAM 可 /写 8 KB 间。 被 时，DMA 间 被 。

表 27-8. 针对 DMA 的 MPU 设置

	边			
SRAM 2				
8 KB	0x3FFA_E000	0x3FFA_FFFF	DPORT_AHB_MPU_TABLE_0_REG	0
8 KB	0x3FFB_0000	0x3FFB_1FFF	DPORT_AHB_MPU_TABLE_0_REG	1
8 KB	0x3FFB_2000	0x3FFB_3FFF	DPORT_AHB_MPU_TABLE_0_REG	2
8 KB	0x3FFB_4000	0x3FFB_5FFF	DPORT_AHB_MPU_TABLE_0_REG	3
8 KB	0x3FFB_6000	0x3FFB_7FFF	DPORT_AHB_MPU_TABLE_0_REG	4
8 KB	0x3FFB_8000	0x3FFB_9FFF	DPORT_AHB_MPU_TABLE_0_REG	5
8 KB	0x3FFB_A000	0x3FFB_BFFF	DPORT_AHB_MPU_TABLE_0_REG	6
8 KB	0x3FFB_C000	0x3FFB_DFFF	DPORT_AHB_MPU_TABLE_0_REG	7
8 KB	0x3FFB_E000	0x3FFB_FFFF	DPORT_AHB_MPU_TABLE_0_REG	8
8 KB	0x3FFC_0000	0x3FFC_1FFF	DPORT_AHB_MPU_TABLE_0_REG	9
8 KB	0x3FFC_2000	0x3FFC_3FFF	DPORT_AHB_MPU_TABLE_0_REG	10
8 KB	0x3FFC_4000	0x3FFC_5FFF	DPORT_AHB_MPU_TABLE_0_REG	11
8 KB	0x3FFC_6000	0x3FFC_7FFF	DPORT_AHB_MPU_TABLE_0_REG	12
8 KB	0x3FFC_8000	0x3FFC_9FFF	DPORT_AHB_MPU_TABLE_0_REG	13
8 KB	0x3FFC_A000	0x3FFC_BFFF	DPORT_AHB_MPU_TABLE_0_REG	14
8 KB	0x3FFC_C000	0x3FFC_DFFF	DPORT_AHB_MPU_TABLE_0_REG	15
8 KB	0x3FFC_E000	0x3FFC_FFFF	DPORT_AHB_MPU_TABLE_0_REG	16
8 KB	0x3FFD_0000	0x3FFD_1FFF	DPORT_AHB_MPU_TABLE_0_REG	17
8 KB	0x3FFD_2000	0x3FFD_3FFF	DPORT_AHB_MPU_TABLE_0_REG	18
8 KB	0x3FFD_4000	0x3FFD_5FFF	DPORT_AHB_MPU_TABLE_0_REG	19
8 KB	0x3FFD_6000	0x3FFD_7FFF	DPORT_AHB_MPU_TABLE_0_REG	20
8 KB	0x3FFD_8000	0x3FFD_9FFF	DPORT_AHB_MPU_TABLE_0_REG	21
8 KB	0x3FFD_A000	0x3FFD_BFFF	DPORT_AHB_MPU_TABLE_0_REG	22
8 KB	0x3FFD_C000	0x3FFD_DFFF	DPORT_AHB_MPU_TABLE_0_REG	23
8 KB	0x3FFD_E000	0x3FFD_FFFF	DPORT_AHB_MPU_TABLE_0_REG	24
SRAM 1				
8 KB	0x3FFE_0000	0x3FFE_1FFF	DPORT_AHB_MPU_TABLE_0_REG	25
8 KB	0x3FFE_2000	0x3FFE_3FFF	DPORT_AHB_MPU_TABLE_0_REG	26
8 KB	0x3FFE_4000	0x3FFE_5FFF	DPORT_AHB_MPU_TABLE_0_REG	27
8 KB	0x3FFE_6000	0x3FFE_7FFF	DPORT_AHB_MPU_TABLE_0_REG	28
8 KB	0x3FFE_8000	0x3FFE_9FFF	DPORT_AHB_MPU_TABLE_0_REG	29
8 KB	0x3FFE_A000	0x3FFE_BFFF	DPORT_AHB_MPU_TABLE_0_REG	30
8 KB	0x3FFE_C000	0x3FFE_DFFF	DPORT_AHB_MPU_TABLE_0_REG	31
8 KB	0x3FFE_E000	0x3FFE_FFFF	DPORT_AHB_MPU_TABLE_1_REG	0
8 KB	0x3FFF_0000	0x3FFF_1FFF	DPORT_AHB_MPU_TABLE_1_REG	1
8 KB	0x3FFF_2000	0x3FFF_3FFF	DPORT_AHB_MPU_TABLE_1_REG	2
8 KB	0x3FFF_4000	0x3FFF_5FFF	DPORT_AHB_MPU_TABLE_1_REG	3
8 KB	0x3FFF_6000	0x3FFF_7FFF	DPORT_AHB_MPU_TABLE_1_REG	4
8 KB	0x3FFF_8000	0x3FFF_9FFF	DPORT_AHB_MPU_TABLE_1_REG	5
8 KB	0x3FFF_A000	0x3FFF_BFFF	DPORT_AHB_MPU_TABLE_1_REG	6

	边			
8 KB	0x3FFF_C000	0x3FFF_DFFF	DPORT_AHB_MPU_TABLE_1_REG	7
8 KB	0x3FFF_E000	0x3FFF_FFFF	DPORT_AHB_MPU_TABLE_1_REG	8

DPROT_AHB_MPU_TABLE_0_REG DPROT_AHB_MPU_TABLE_1_REG DPort 间。PID 0 1 可 个 。

说明：

，3 别 $VAddr_1$、$VAddr_2$ $VAddr_3$，3 可 时 / ，一个 。 被屏蔽 超 一个， MMU 表 bit8 0。 ， MMU 表 被 个 时， ，cache 被 。

27.3.2.2 片外存储器

SPI RAM Cache ，并 MMU 。 PID CPU，Cache MMU 可 不 ， MMU。 ， 个 ，一个 。 MMU Cache MMU 间 差 。 ，Cache MMU (64 KB，RAM 32 KB)；，Cache MMU 个 PID 表，不 MMU 。，MMU 被 “ ”。 PID 0 1 ；PID 2 7 必 PID 0 1 变 MMU 设 。

，MMU 。MMU 间 个 ， 表 27-9。 $VAddr_1$ $VAddr_4$ ，$VAddr_{RAM}$ RAM。 $VAddr_4$ $VAddr_0$ 。

表 27-9. 片外存储器的虚地址

		边		
$VAddr_0$	4 MB	0x3F40_0000	0x3F7F_FFFF	64
$VAddr_1$	4 MB	0x4000_0000	0x403F_FFFF	64*
$VAddr_2$	4 MB	0x4040_0000	0x407F_FFFF	64
$VAddr_3$	4 MB	0x4080_0000	0x40BF_FFFF	64
$VAddr_4$	1 MB	0x3F40_0000	0x3F4F_FFFF	16
$VAddr_{RAM}$	4 MB	0x3F80_0000	0x3FBF_FFFF	128

* 表 便 写 0x4000_0000 ~ 0x403F_FFFF 一个 4 MB 间。部 。 0x4000_0000 ~ 0x400C_1FFF 。 $VAddr_1$ 不 被 。

片外闪存

表 27-10 27-11 ， PID 。 个表 个 PID 一个 MMU 。表 一个 MMU ；“ ”一 表 ， 。

个表本 上 ， 别 APP_CPU 比 PRO_CPU 2048。 ，
$VAddr_0$ $VAddr_1$ PID 0 1 ，$VAddr_4$ PID 2 7 。

表 27-10. PRO_CPU 的 MMU 配置项号

		PID 一个 MMU						
		0/1	2	3	4	5	6	7
$VAddr_0$	64	0	-	-	-	-	-	-
$VAddr_1$	64	64	-	-	-	-	-	-
$VAddr_2$	64	128	256	384	512	640	768	896
$VAddr_3$	64	192	320	448	576	704	832	960
$VAddr_4$	16	-	1056	1072	1088	1104	1120	1136

表 27-11. APP_CPU 的 MMU 配置项号

		PID 一个 MMU						
		0/1	2	3	4	5	6	7
$VAddr_0$	64	2048	-	-	-	-	-	-
$VAddr_1$	64	2112	-	-	-	-	-	-
$VAddr_2$	64	2176	2304	2432	2560	2688	2816	2944
$VAddr_3$	64	2240	2368	2496	2624	2752	2880	3008
$VAddr_4$	16	-	3104	3120	3136	3152	3168	3184

上表 ， $VAddr_1$ PID 0 1 。 一 PID 2 7
$VAddr_1$ 。 DPORT_PRO_CACHE_CTRL_REG 中 DPORT_PRO_SINGLE_IRAM_ENA 1 时，MMU PRO_CPU 。 ，DPORT_APP_ CACHE_CTRL_REG 中 DPORT_APP_SINGLE_IRAM_ENA 1 时，APP_CPU 。 ，MMU 个 不 。 参 表 27-12 表 27-13。 表 ， ，不 $VAddr_2$ $VAddr_3$ 。

表 27-12. PRO_CPU 的 MMU 配置项号（特殊模式）

		PID 一个 MMU						
		0/1	2	3	4	5	6	7
$VAddr_0$	64	0	-	-	-	-	-	-
$VAddr_1$	64	64	256	384	512	640	768	896
$VAddr_2$	64	-	-	-	-	-	-	-
$VAddr_3$	64	-	-	-	-	-	-	-
$VAddr_4$	16	-	1056	1072	1088	1104	1120	1136

表 27-13. APP_CPU 的 MMU 配置项号（特殊模式）

		PID 一个 MMU						
		0/1	2	3	4	5	6	7
$VAddr_0$	64	2048	-	-	-	-	-	-
$VAddr_1$	64	2112	2304	2432	2560	2688	2816	2944
$VAddr_2$	64	-	-	-	-	-	-	-
$VAddr_3$	64	-	-	-	-	-	-	-
$VAddr_4$	16	-	3104	3120	3136	3152	3168	3184

MMU 个 CPU 。 个 32。 中，0 7 表 。MMU 时必 8 。8 1 不 。10 32 不 ， 写 。 MMU 中 8 个 ，扩 64 KB， 扩 256 * 64 KB = 16 MB。

示例

1：PID 1 PRO_CPU 0x3F70_2375 扩 0x07_2375。MMU 不 。

- 表 27-9， 0x3F70_2375 $VAddr_0$ 0x30 。
- 表 27-10，PID 0/1 时， PRO_CPU，$VAddr_0$ MMU 0 始。
- 被 改 MMU 0 + 0x30 = 0x30。
- 0x07_2375 7 64 KB 上。
- MMU 0x30 设 7，并 8 0 标记 ， ， 0x007 写 MMU 0x30。

2：PID 4 APP_CPU 0x4044_048C 扩 0x44_048C。MMU 不 。

- 表 27-9， 0x4044_048C $VAddr_2$ 0x4 。
- 表 27-11，PID 4 时， APP_CPU，$VAddr_2$ MMU 2560 始。
- 被 改 MMU 2560 + 0x4 = 2564。
- 0x44_048C 0x44 64 KB 上。
- MMU 2564 设 0x44 并 8 0 标记 ， ， 0x044 写 MMU 2564。

片外 RAM

PRO_CPU APP_CPU 上 可 写扩 SRAM， 写 $VAddr_{RAM}$，0x3F80_0000 ~ 0x3FBF_FFFF。 MMU 一 ， 间 被 。 扩 RAM MMU， 32 KB，并 MMU 256 个 间， 32 KB * 256 = 8 MB 扩 RAM 。

MMU : - ， - 奇 常 。 ， DPORT_PRO_CACHE_CTRL_REG 中 DPORT_PRO_DRAM_HL DPORT_PRO_DRAM_SPLIT ，

DPORT _APP_CACHE_CTRL_REG 中 DPORT_APP_DRAM_HL DPORT_APP_DRAM_SPLIT
SRAM , 参 表 27-14。 PRO_CPU APP_CPU 不 , 常 ,
一可 。 PRO_CPU APP_CPU , - -奇
可 个 CPU 时 。
APP_CPU Cache 闭时，0x4007_8000 0x4007_FFFF 常 RAM， Cache 可
变化 常 ，可 PRO_CPU RAM 时保 Cache 常 , APP_CPU
RAM。 - , 个 CPU 可 RAM， 0x3F80_0000 0x3F9F_FFFF
2 MB 。不 APP_CPU Cache 闭 奇- 。

表 27-14. 片外 SRAM 的虚拟地址模式

	DPORT_PRO_DRAM_HL DPORT_APP_DRAM_HL	DPORT_PRO_DRAM_SPLIT DPORT_APP_DRAM_SPLIT
-	1	0
-奇	0	1
常	0	0

常 , 个 CPU 可 不 。 ${}^{L}VAddr_{RAM}$ MMU 设
PRO_CPU ; ${}^{R}VAddr_{RAM}$ MMU 设 APP_CPU 。 一 ,
${}^{L}VAddr$ and ${}^{R}VAddr$ 128 被 , 新 8 MB 。 中 4 MB PRO_CPU
, 4 MB APP_CPU , 表 27-15 。

表 27-15. 片外 SRAM 的虚地址（正常模式）

		PRO_CPU	
${}^{L}VAddr_{RAM}$	4 MB	0x3F80_0000	0x3FBF_FFFF
		APP_CPU	
${}^{R}VAddr_{RAM}$	4 MB	0x3F80_0000	0x3FBF_FFFF

- , PRO_CPU APP_CPU 。 , ${}^{L}VAddr_{RAM}$
间 2 MB，新 ${}^{R}VAddr_{RAM}$ 间 2 MB。 ${}^{L}VAddr_{RAM}$ 64 个 MMU
${}^{R}VAddr_{RAM}$ 64 个 。表 27-16 。

表 27-16. 片外 SRAM 的虚地址（低-高模式）

		PRO_CPU/APP_CPU	
${}^{L}VAddr_{RAM}$	2 MB	0x3F80_0000	0x3F9F_FFFF
${}^{R}VAddr_{RAM}$	2 MB	0x3FA0_0000	0x3FBF_FFFF

-奇 中，VRAM 被拆 32 。 MMU ${}^{L}VAddr_{RAM}$，奇
MMU ${}^{R}VAddr_{RAM}$。一般 , ${}^{L}VAddr_{RAM}$ ${}^{R}VAddr_{RAM}$ MMU 被设
, 可 一个 。表 27-17 。

表 27-17. 片外 SRAM 的虚地址（偶-奇模式）

		PRO_CPU/APP_CPU	
${}^{L}VAddr_{RAM}$	32	0x3F80_0000	0x3F80_001F
${}^{R}VAddr_{RAM}$	32	0x3F80_0020	0x3F80_003F
${}^{L}VAddr_{RAM}$	32	0x3F80_0040	0x3F80_005F
${}^{R}VAddr_{RAM}$	32	0x3F80_0060	0x3F80_007F
...			
${}^{L}VAddr_{RAM}$	32	0x3FBF_FFC0	0x3FBF_FFDF
${}^{R}VAddr_{RAM}$	32	0x3FBF_FFE0	0x3FBF_FFFF

RAM MMU : 32 , 9 。0 7 包 。 , 8 被 ; , 8 1。表 27-18 ${}^{L}VAddr_{RAM}$ ${}^{R}VAddr_{RAM}$ PID 一个 MMU 。

表 27-18. 片外 RAM 的 MMU 配置项号

		PID 一个 MMU						
		0/1	2	3	4	5	6	7
${}^{L}VAddr_{RAM}$	128	1152	1280	1408	1536	1664	1792	1920
${}^{R}VAddr_{RAM}$	128	3200	3328	3456	3584	3712	3840	3968

示例

1：PRO_CPU PID 7 0x3FA7_2375 RAM 0x7F_A375。MMU - 。

- 表 27-9， 0x3FA7_2375 $VAddr_{RAM}$ 0x4E 32 KB 。
- 表 27-16， 0x3FA7_2375 ${}^{R}VAddr_{RAM}$ 。
- 表 27-18 PRO_CPU PID 7 ，${}^{R}VAddr_{RAM}$ MMU 始 3968。
- 被 MMU 3968 + 0x4E = 4046。
- 0x7F_A375 255 32 KB 。
- MMU 4046 被 255，并 8 标记 , , 0x0FF MMU 4046。

2：APP_CPU PID 5 0x3F85_5805 始 RAM 0x55_5805 0x55_5823。MMU 奇- 。

- 表 27-9， 0x3F85_5805 $VAddr_{RAM}$ 0x0A 32 KB 。
- 表 27-17， 包 ${}^{R}VAddr_{RAM}$ ${}^{L}VAddr_{RAM}$ 中 32 。
- 表 27-18， PID 5，${}^{L}VAddr_{RAM}$ MMU 始 1664。
- 表 27-18， PID 5，${}^{R}VAddr_{RAM}$ MMU 始 3712。
- 被 MMU 1664 + 0x0A = 1674 3712 + 0x0A = 3722。
- 0x55_5805 0x55_5823 0xAA 32 KB 。

- MMU 1674 3722 被设 0xAA，并 8 0 标记 ， ，0x0AA 写 MMU 1674 3722。 PRO_CPU APP_CPU。

3：PRO_CPU 上 PID 1 APP_CPU 上 PID 1 0x3F80_0876 写 RAM。PRO_CPU 0x10_0876，新 APP_CPU 0x20_0876。MMU 常 。

- 表 27-9， 0x3F80_0876 $VAddr_{RAM}$ 0 32 KB 上。
- 表 27-18， PRO_CPU 上 PID 1 ，MMU 始 1152。
- 表 27-18， APP_CPU 上 PID 1 ，MMU 始 3200。
- PRO_CPU，被 记 MMU 1152 + 0 = 1152， APP_CPU，被 记 MMU 3200 + 0 = 3200。
- 0x10_0876 0x20 32 KB 上。
- 0x20_0876 0x40 32 KB 上。
- PRO_CPU, MMU 1152 被 0x20 并 8 标记 ， 0x020 写 MMU 1152。
- APP_CPU，MMU 3200 被 0x40 并 8 标记 ， 0x040 写 MMU 3200。
- PRO_CPU APP_CPU 可 不 。

27.3.2.3 外设

设 MPU 41 个 设 。 个 设 ，可 MMU， PID 设。MMU 参 表 27-19。

表 27-19. 管理外设的 MPU

设	PID = 0/1	PID = 2 ~ 7
DPort Register		
AES Accelerator		
RSA Accelerator		
SHA Accelerator		
Secure Boot		
Cache MMU Table		
PID Controller		
UART0		DPORT_AHBLITE_MPU_TABLE_UART_REG
SPI1		DPORT_AHBLITE_MPU_TABLE_SPI1_REG
SPI0		DPORT_AHBLITE_MPU_TABLE_SPI0_REG
GPIO		DPORT_AHBLITE_MPU_TABLE_GPIO_REG
RTC		DPORT_AHBLITE_MPU_TABLE_RTC_REG
IO MUX		DPORT_AHBLITE_MPU_TABLE_IO_MUX_REG
SDIO Slave		DPORT_AHBLITE_MPU_TABLE_HINF_REG
UDMA1		DPORT_AHBLITE_MPU_TABLE_UHCI1_REG
I2S0		DPORT_AHBLITE_MPU_TABLE_I2S0_REG
UART1		DPORT_AHBLITE_MPU_TABLE_UART1_REG

设		
	PID = 0/1	PID = 2 ~ 7
I2C0		DPORT_AHBLITE_MPU_TABLE_I2C_EXT0_REG
UDMA0		DPORT_AHBLITE_MPU_TABLE_UHCI0_REG
SDIO Slave		DPORT_AHBLITE_MPU_TABLE_SLCHOST_REG
RMT		DPORT_AHBLITE_MPU_TABLE_RMT_REG
PCNT		DPORT_AHBLITE_MPU_TABLE_PCNT_REG
SDIO Slave		DPORT_AHBLITE_MPU_TABLE_SLC_REG
LED PWM		DPORT_AHBLITE_MPU_TABLE_LEDC_REG
Efuse Controller		DPORT_AHBLITE_MPU_TABLE_EFUSE_REG
Flash Encryption		DPORT_AHBLITE_MPU_TABLE_SPI_ENCRYPT_REG
PWM0		DPORT_AHBLITE_MPU_TABLE_PWM0_REG
TIMG0		DPORT_AHBLITE_MPU_TABLE_TIMERGROUP_REG
TIMG1		DPORT_AHBLITE_MPU_TABLE_TIMERGROUP1_REG
SPI2		DPORT_AHBLITE_MPU_TABLE_SPI2_REG
SPI3		DPORT_AHBLITE_MPU_TABLE_SPI3_REG
SYSCON		DPORT_AHBLITE_MPU_TABLE_APB_CTRL_REG
I2C1		DPORT_AHBLITE_MPU_TABLE_I2C_EXT1_REG
SDMMC		DPORT_AHBLITE_MPU_TABLE_SDIO_HOST_REG
EMAC		DPORT_AHBLITE_MPU_TABLE_EMAC_REG
PWM1		DPORT_AHBLITE_MPU_TABLE_PWM1_REG
I2S1		DPORT_AHBLITE_MPU_TABLE_I2S1_REG
UART2		DPORT_AHBLITE_MPU_TABLE_UART2_REG
PWM2		DPORT_AHBLITE_MPU_TABLE_PWM2_REG
PWM3		DPORT_AHBLITE_MPU_TABLE_PWM3_REG
RNG		DPORT_AHBLITE_MPU_TABLE_PWR_REG

DPORT_AHBLITE_MPU_TABLE_X_REG 个 个 可 设。参 表 27-20。 DPORT_AHBLITE_MPU_TABLE_X_REG 个 1 时，PID 可 设。 ， 设。

表 27-20. DPORT_AHBLITE_MPU_TABLE_X_REG

PID	2 3 4 5 6 7
DPORT_AHBLITE_MPU_TABLE_X_REG	0 1 2 3 4 5

DPORT_AHBLITE_MPU_TABLE_X_REG 设 DPort 中。 PID 0/1 可 改 。

28 PID 控制器 (PID)

28.1 概述

ESP32 可 时 个 。PID 中， PID 。 ， PID 可 记录 CPU 中 中 。 ，PID 可 中 中 。

28.2 主要特性

PID :

-
- PID
- 记录中
- 中

28.3 功能描述

CPU 8 个 ， PID 别 0 ~ 7。 8 个 中，PID 0/1 比 PID 2 ~ 7 。

CPU 。

- 中 ，CPU 中 时 。 中 个 ， 时 中 视 PID 0 。
- 时。 一 PID 0/1 。

28.3.1 中断识别

CPU 一 Level 1、Level 2、Level 3、Level 4、Level 5、Level 6（Debug）、NMI 个 中 。 一 中 一个中 。PID 识别 CPU 中 时 PID 0。 CPU 中 ， PID 不采 。

PID 识别 中 PIDCTRL_INTERRUPT_ENABLE_REG。 PIDCTRL_INTERRUPT_ENABLE_REG 中 1 时， CPU 中 时，PID ； PIDCTRL_INTERRUPT_ENABLE_REG 中 0 时， PID 不采 。 中 中 PIDCTRL_INTERRUPT_ADDR_1_REG ~ PIDCTRL_INTERRUPT_ADDR_7_REG ， 表 28-1。

表 28-1. 中断向量入口地址

中	中 识别 PIDCTRL_INTERRUPT_ENABLE_REG	中
Level 1	1	PIDCTRL_INTERRUPT_ADDR_1_REG
Level 2	2	PIDCTRL_INTERRUPT_ADDR_2_REG
Level 3	3	PIDCTRL_INTERRUPT_ADDR_3_REG
Level 4	4	PIDCTRL_INTERRUPT_ADDR_4_REG
Level 5	5	PIDCTRL_INTERRUPT_ADDR_5_REG
Level 6 (Debug)	6	PIDCTRL_INTERRUPT_ADDR_6_REG
NMI	7	PIDCTRL_INTERRUPT_ADDR_7_REG

28.3.2 信息记录

PID 识别 中 时， PID 0， 记录 3 :

1. 中
2. 上一 中
3. CPU 上一个

PID 中 记录 PIDCTRL_LEVEL_REG， 见表 28-2。

表 28-2. PIDCTRL_LEVEL_REG

	中
0	不 中 中
1	Level 1 中
2	Level 2 中
3	Level 3 中
4	Level 4 中
5	Level 5 中
6	Level 6 中
7	NMI 中

PID 中 记录 PIDCTRL_FROM_*n*_REG。
PIDCTRL_FROM_*n*_REG 7。 4 表 中 中 。 3 表 中 个 。 见表 28-3。

表 28-3. PIDCTRL_FROM_*n*_REG

[6:3]		[2:0]	
0	不	0	PID 0
1	Level 1	1	PID 1
2	Level 2	2	PID 2
3	Level 3	3	PID 3
4	Level 4	4	PID 4
5	Level 5	5	PID 5
6	Level 6	6	PID 6
7	Level 7	7	PID 7

PID PIDCTRL_FROM_1_REG ~ PIDCTRL_FROM_7_REG， 别 Level 1、Level 2、Level 3、Level 4、Level 5、Level 6（Debug）、NMI 。 ， 28-1 。

图 28-1. 中断嵌套

中，PIDCTRL_INTERRUPT_ENABLE_REG，而 PID 识别，PID 不记录，不 PIDCTRL_LEVEL_REG、PIDCTRL_FROM_*n*_REG。

28.3.3 进程主动切换进程

PID 0/1 可。可 PID 0 ~ 7 中一个。，一一时，PID 0/1 变 PID。

1. 屏蔽 NMI 中 中
2. PIDCTRL_NMI_MASK_ENABLE_REG 1， CPU NMI 中 屏蔽
3. PIDCTRL_PID_DELAY_REG、PIDCTRL_NMI_DELAY_REG
4. PIDCTRL_PID_NEW_REG
5. PIDCTRL_LEVEL_REG、PIDCTRL_FROM_*n*_REG
6. PIDCTRL_PID_CONFIRM_REG、PIDCTRL_NMI_MASK_DISABLE_REG 1
7. NMI 中 中 屏蔽
8.

可 中 ， PID 0/1 不 时被 中 ， 步 1、步 2 中屏蔽 中 。

步 3 中 PIDCTRL_PID_DELAY_REG、PIDCTRL_NMI_DELAY_REG 步 6。

步 4 中 PIDCTRL_PID_NEW_REG 步 6 PID。

中 中 上一个中 ， 步 5 中 *n* 中记录 PIDCTRL_LEVEL_REG。

步 6 中， PIDCTRL_PID_CONFIRM_REG、PIDCTRL_NMI_MASK_DISABLE_REG 1 ，PID 并不 PID PIDCTRL_PID_NEW_REG 中 ， 不 闭 CPU NMI 中 屏蔽 ，所 别 一 时 期 才 个 。 时 期 PIDCTRL_PID_DELAY_REG、PIDCTRL_NMI_DELAY_REG 中 。

步 7 中 可 ， 步 3 中 PIDCTRL_PID_DELAY_REG、PIDCTRL_NMI_DELAY_REG 时包 时间 可。

28.4 寄存器列表

ë	í •	¹ ¶	` Ù
1 * % $ 5 3 - @ * / 5 & 3 3 6 1 5 @ & / " #	1 * % M Y & P (? Ê	Y ' ' '	•
1 * % $ 5 3 - @ * / 5 & 3 3 6 1 5 @ " % %	F W FÏM_ 3 & E g ¹ ¶	Y ' ' '	•
1 * % $ 5 3 - @ * / 5 & 3 3 6 1 5 @ " % %	F W FÏM_ 3 & E g ¹ ¶	Y ' ' '	•
1 * % $ 5 3 - @ * / 5 & 3 3 6 1 5 @ " % %	F W FÏM_ 3 & E g ¹ ¶	Y ' ' ' $	•
1 * % $ 5 3 - @ * / 5 & 3 3 6 1 5 @ " % %	F W FÏM_ 3 & E g ¹ ¶	Y ' ' '	•
1 * % $ 5 3 - @ * / 5 & 3 3 6 1 5 @ " % %	F W FÏM_ 3 & E g ¹ ¶	Y ' ' '	•
1 * % $ 5 3 - @ * / 5 & 3 3 6 1 5 @ " % %	F W FÏM_ 3 & E g ¹ ¶	Y ' ' '	•
1 * % $ 5 3 - @ * / 5 & 3 3 6 1 5 @ " % %	/ . * Ï _ @ & E g ¹ ¶	Y ' ' ' $	•
1 * % $ 5 3 - @ 1 * % @ % & - " : @ 3 &	(¥ 1 * % 3 r - ¥ ü þ	Y ' ' '	•
1 * % $ 5 3 - @ / . * @ % & - " : @ 3 & (	1 > / . * à 7 • | - ¥ ü þ	Y ' ' '	•
1 * % $ 5 3 - @ - & 7 & - @ 3 & (	' - Ï ª 5)	Y ' ' '	•
1 * % $ 5 3 - @ ' 3 0 . @ @ 3 & (	- F W FÏM? 3 - ¥ " d ÿ	Y ' ' ' $	•
1 * % $ 5 3 - @ ' 3 0 . @ @ 3 & (	- F W FÏM? 3 - ¥ " d ÿ	Y ' ' '	•
1 * % $ 5 3 - @ ' 3 0 . @ @ 3 & (	- F W FÏM? 3 - ¥ " d ÿ	Y ' ' '	•
1 * % $ 5 3 - @ ' 3 0 . @ @ 3 & (	- F W FÏM? 3 - ¥ " d ÿ	Y ' ' '	•
1 * % $ 5 3 - @ ' 3 0 . @ @ 3 & (	- F W FÏM? 3 - ¥ " d ÿ	Y ' ' ' $	•
1 * % $ 5 3 - @ ' 3 0 . @ @ 3 & (	- F W FÏM? 3 - ¥ " d ÿ	Y ' ' '	•
1 * % $ 5 3 - @ ' 3 0 . @ @ 3 & (	/ . * Ï ? 3 - ¥ " d ÿ	Y ' ' '	•
1 * % $ 5 3 - @ 1 * % @ / & 8 @ 3 & (	¥ Â • ¥ 1 * %	Y ' ' '	•
1 * % $ 5 3 - @ 1 * % @ $ 0 / ' * 3 . @	3 ' ª (¥ 1 * %	Y ' ' ' $	º •
1 * % $ 5 3 - @ / . * @ . " 4 , @ & / " #	- & . @ Ì 3 & (ª à 7 P ? 7 i	Y ' ' '	º •
1 * % $ 5 3 - @ / . * @ . " 4 , @ % * 4 " #	/ - & . @ à 7 & (1 > 7 i	Y ' ' '	º •

7 i

3 F H J T U F S 1 * % $ 5 3 - @ * / 5 & 3 3 6 1 5 @ & / " # - & @ 3 & (Y

S F T F S W F E 1 * % $ 5 3 - @ * / 5 & 3 3 6 1 5 @ & / " # - & S F T F S W F E

j R 3	d R	y	_ 2 b 2 i

1 * % $ 5 3 - @ * / 5 & 3 3 6 1 5 @ & ¼ Ê # ¿ ® ? Ï ¥ M Y „) Ø b •

3 F H J T U F S 1 * % $ 5 3 - @ * / 5 & 3 3 6 1 5 @ " % % 3 @ @ 3 & (Y

j R	y	
Y		_ 2 b 2 i

1 * % $ 5 3 - @ * / 5 & 3 3 6 1 5 @ " % % - 3 F @ W F @ M 3 & Æ E g ¹ ¶ b •

3 F H J T U F S 1 * % $ 5 3 - @ * / 5 & 3 3 6 1 5 @ " % % 3 @ @ 3 & (Y

j R	y	
Y		_ 2 b 2 i

1 * % $ 5 3 - @ * / 5 & 3 3 6 1 5 @ " % % - 3 F @ W F @ M 3 & Æ E g ¹ ¶ b •

3 F H J T U F S 1 * % $ 5 3 - @ * / 5 & 3 3 6 1 5 @ " % % 3 @ @ 3 & (Y $

j R	y	
Y $		_ 2 b 2 i

1 * % $ 5 3 - @ * / 5 & 3 3 6 1 5 @ " % % - 3 F @ W F @ M 3 & Æ E g ¹ ¶ b •

3 F H J T U F S 1 * % $ 5 3 - @ * / 5 & 3 3 6 1 5 @ " % % 3 @ @ 3 & (Y

j R	y	
Y		_ 2 b 2 i

1 * % $ 5 3 - @ * / 5 & 3 3 6 1 5 @ " % % - 3 F @ W F @ M 3 & Æ E g ¹ ¶ b •

3 F H J T U F S 1 * % $ 5 3 - @ * / 5 & 3 3 6 1 5 @ " % % 3 @ @ 3 & (Y

j R	y
Y	_ 2 b 2 i

1 * % $ 5 3 - @ * / 5 & 3 3 6 1 5 @ " % % - 3 F @ W F @ M 3 & Æ E g ¹ ¶ b •

3 F H J T U F S 1 * % $ 5 3 - @ * / 5 & 3 3 6 1 5 @ " % % 3 @ @ 3 & (Y

j R	y
Y	_ 2 b 2 i

1 * % $ 5 3 - @ * / 5 & 3 3 6 1 5 @ " % % - 3 F @ W F @ M 3 & Æ E g ¹ ¶ b •

3 F H J T U F S 1 * % $ 5 3 - @ * / 5 & 3 3 6 1 5 @ " % % 3 @ @ 3 & (Y $

j R	y
Y $	_ 2 b 2 i

1 * % $ 5 3 - @ * / 5 & 3 3 6 1 5 @ " % % / 3 * @ Ï @ 3 E g ¹ ¶ b •

3 F H J T U F S 1 * % $ 5 3 - @ 1 * % @ % & - " : @ 3 & (Y

S F T F S W F E

1 * % $ 5 3 - @ 1 * % @ % & -

j R R k	R R y
	_ 2 b 2 i

1 * % $ 5 3 - @ 1 * % @ % & ¥ " : 1 * % 3 r - ¥ ü þ b •

3 F H J T U F S 1 * % $ 5 3 - @ / . * @ % & - " : @ 3 & (Y

S F T F S W F E

1 * % $ 5 3 - @ / . * @ % & -

j R R k	R R y
	_ 2 b 2 i

1 * % $ 5 3 - @ / . * @ % & > " $ 1 6 / . * Ï à 7 • | - ¥ ü þ b •

3 F H J T U F S 1 * % $ 5 3 - @ - & 7 & - @ 3 & (Y

SFTFSWFE	1 * % $ 5 3 - @ $ 6 3 3 & / 5 @ 4 " 5 6 4
jR 9	j y

_ 2 b 2 i

1 * % $ 5 3 - @ $ 6 3 3 & / 5 @ 4 " 5 6 ÿ b •

3 F H J T U F S 1 * % $ 5 3 @ @ & 3 0 . @ Y Y

SFTFSWFE	1 * % $ 5 3 - @ 1 3 & 7 * 0 6 4 @ 4 5 " 5 6 4 @
jR d	e y

_ 2 b 2 i

1 * % $ 5 3 - @ 1 3 & 7 * 0 6 4 @ 4 5 " 5 6 4 @ F À M - F W F M / . * Ï ? 3 - ¥ " d ÿ b

•

3 F H J T U F S 1 * % $ 5 3 - @ 1 * % @ / & 8 @ 3 & (Y

SFTFSWFE	1 * % $ 5 3 - @ 1 * % @ / & 8
jR j	k y

_ 2 b 2 i

1 * % $ 5 3 - @ 1 * % @ ¥ / & 1 3 * % •

3 F H J T U F S 1 * % $ 5 3 - @ 1 * % @ $ 0 / ' * 3 . @ 3 & (Y $

S F T F S W F E

1 * % $ 5 3 - @ 1 * % @ $ 0 / ' * 3 .

j R	R	y
		_ 2 b 2 i

1 * % $ 5 3 - @ 1 * % @ $ 0¾'Ê 3¿ P • 1 * %3 r b º •

3 F H J T U F S 1 * % $ 5 3 - @ / . * @ . " 4 , @ & / " # - & @ 3 & (Y

S F T F S W F E

1 * % $ 5 3 - @ / . * @ . " 4 , @ & / " #

j R	R	y
		_ 2 b 2 i

1 * % $ 5 3 - @ / . * @ . " 4 , @ & /P ? - &1 6 / . *Ï à 7 • | b º •

3 F H J T U F S 1 * % $ 5 3 - @ / . * @ . " 4 , @ % * 4 " # - & @ 3 & (Y

S F T F S W F E

1 * % $ 5 3 - @ / . * @ . " 4 , @ % * 4 "

j R	R	y
		_ 2 b 2 i

1 * % $ 5 3 - @ / . * @ . " 4 , @ % *14> " #$ 1&6 / . *Ï à 7 • | b º •

Ë .Ë Ð E•|)Ø

À•

¹ ¦û Õ‹¨Æß &41 ö1•¨ ÕË˜¥.Ë È¸T(.Ë KÚ¦û ^{Æ
„ :r‹.Ë b

&41 ¥ E•|)Ø ö1® ñïQ/Í E”3 Ð 4VDDFTTJWF "QQSPYJNBUJPO "%$ 4"
"d Ê=Â ñ "%$¨eÅ V Ð E{Æ•|H¦ûÚŸ?Ð®ÿ{ Õ T)Ø ¥7
gK®b

N“ &41 ÎVP¨ ñÿë”3 E Ð %"$ ñÅ?o™?3 3î E•|b

È¸T(.Ë

e°

(.Ë "dö1® ñ†sFî V“ž=GQ¹Üë ¹ªaÈ Ð Ë nm b'¨¼(· ¹
ªH .Ë "d¥È¸ ö?3¿M ?73î ñV[Q˜'Q(·^Ž(?¥=ÉÅ•|b

m (.Ë

ö1+Ÿ

t K ¦û ^È¸(5– Y¨{Æ{ ¤g (FOFSBM 1VSQPTF *OQVU BOE 0VUQVU (1

t (5–V[F†P¨ V-Â÷v(Ëu× ÷ (ËÄ

t (5–¥.Ë®µK ÿ '4. —qeÅ ®Èq ¨—q9H ?

t (5–^Žsž(·¥• V®[/ZT ¤

o ®Èq°¤_°(.Ë ¥7i

o ®(S© v? ¥Ï •|‘

o ®(S© v ¥ $16^ŽV %FFQ TÏÒFœF‘Q

t ¦û[/Æß/¥®ÿ{ýT

o $16)¿ %FFQ TÏMFFQ sž(·ªï„Òœ

o (S©®Ñ®ÿ{x)Ø 6-1 DPQSPD5ØTTPS
6-1¨¼ñ½VYV•ÆÐ_°+ç7i ‘ ^Žrž(·•´

V¨Y¨{Æ{ ¤g

†† ñV¨Y¨{Æ{ ¤g¥• hnV b

V &41È¸T(.Ë ¥5–

(.Ë•|	5–
5	(1*0
5	(1*0
5	(1*0
5	.5%0
5	.5$,
5	.5%*
5	.5.4
5	(1*0
5	,@9/
5	,@91

ÿ?í•

(.Ë ¥=†² hnm ýT@ñhnm b

m (.Ë ¥=†²

(5–¥È¸öÉ›ÛùŸ bÈb w(5–¥=†Èâ w}V bÈÈâ •IÚ´ ESFÐ•)I®´
ESF-Ø-¥MÄb ÄQMÄÏ (.Ë |3îBñ{ • 065®¿(5–sž(· ÚÈ¸
ÐÃsž(· ®È¸ H¥ÈâMÄÎq,] áÌV[YVd9]BHWWï= C¥{ • ” ‘
(5–^Žsž(·bV[YV 5*&@0ÂÀ7S bÈ¥ SÈâÈÜb

(?.Ë ¥ ÿ

µK ÿ 'JOJUF 4UBUF .B|Ð»IJOF '4c.«í•¥½ _©bÈqVYV ¨7i T '4.b
'4. ¥=†² Vnm b

'4. ¥ÿ? “

m (.Ë ¥ýT@ñ

t ¤lÈq 9H ? ¥7S•|

- o' 4&/4@4"3@506$)@45"35@H0V3¥X!Â 4&/4@4"3@506$)@45?"35@&/ BQŸ_©
- o' 4&/4@4"3@506$)@45"35@H0V3ãe&9H LCÛùŸ_©b

(.Ë ¤ä T/9?ýTb÷ • hnÿ{5Øc«bVYV7i
4&/4@4"3@506$)@4-&&1ç_$©Ûù&4Ë ® '"45@$e,Å ÈnHÒÔq¹
.)[b÷ • hn ¯Ê„HÒ c«b

t ô VØ«H½ 3î 91%@506$)@#*"4 506$)@91% 506$)@45"35
Ê4P?(5–H 506$)@91% 506$)@Ï¥45¸35 Ê7i
4&/4@4"3@506$)@1"%@q8b03,&/

t 9" 506$) @06a5506$) @06¥5• "
9"²TVn 4&/4@4"3@506$)@.&Cb4¹@065ñ(5–V¦û]HýTb

t 3îÒœÏ
ÂTBñ5–¥• 9"²T®¿•´ 5 '4. j¾5–$o(·pb Ê7i
4&/4@506$)@1"%@04&&4@506$)@1"%@V5|5µ5–çl¹ F ' 4&5
4&5b ª ÿ/ ÂT 4&5Ï¥©i5–$o(·p'V3îÒœÏ 9V[¥Â¹ 4&5„
4&5Ï(µ5–$o(·pHµrb

m '4 . ¥ = † ²

4 " 3 " % $

e º

& 4 1 = Â ñ Ê ¥ 4 " 3 " % $® ñ ¨ Ð e Å 5 Ø V © Ÿ 1 ñ 5 – ¥ E • | Î V © W E E © = † • | b

m 4 " 3 " % ¥ $ À ƒ

4 " 3 " % $ Þ ¨ ¥ ñ e Å (¹ ¨ e Å Ï ñ ¦ û Ú Ÿ ? Y ¡ á í a ñ Ü V ª Ä V ¦ û

Á ˆ • S / & 4 1 / Œ • I m ¨ ñ '

%FFQ TMTF¥FQÿ{ › 6“ ñ Ê¨¿ 18%&5 1,%ÿ&Ф_©„,´S©b 4"3"%$¥
'Àƒnm b

ªü
18%&5 1,%eÅ5Ç 8J 'Ð†P¨bÂT 8J 'ÐP¨ 4"3"%$5¨¼íEP¨ 4"3"%$© 5–
¥ E•|b 8J 'dÐb 4"3"%$-ª ¨¼VžÈP¨ 4"3"%$b

ö1+Ÿ

t •¨ ñ 4"3"%$V¦û]H•"Ð Ð

t •¨ ñ ¨ "%$eÅ V¦û,]‹¨ÆÈß 1Â ÚŸ?a®ÿ{ ÿq_©„,´_©

t ¦û ñ E{Æ5–

t ñ=†Èâ WEEY¡a ñ QB@QYEFHUeÅ ¦û

t V¥Â Êa Êa Êa Ê ÕsOq

t ¦û %."ñeÅ ¦û

t ¦û Y¡áí T ñeÅ ¦û

t ¦û %FFQ TMTF›FQ ñeÅ ¦û

t ¦û 6-1x)Ø eÅ ñeÅ ¦û

ÿ?Àƒ

4"3"%$¥ö1íqÐõ¤ffnm b

m 4"3"%$¥ÿ?Àƒ

îµV?Ð 4"3 "%$“ "%$„ "%$ µ1¥5–• hnV b

V 4"3 "%¥$•|{Æ

•| ë	5–	"%$Ê4
7%&5@		4"3 "%$
7%&5@		
,@9/		
,@91		
4&/403@7/		
4&/403@$"1/		
4&/403@$"11		
4&/403@71		
)BMM TFOTPS	O B	
(1*0		4"3 "%$
(1*0		
(1*0		
.5.4		
.5%*		
.5$,		
.5%0		
(1*0		
(1*0		
(1*0		
QB@QLEFU	O B	
QB@QLEFU	O B	
WEE	O B	

ªü
4"3 "%$¥_©5–Ï “ (1*0a(1*0„ (1*0 ¹‰Ë 4USBQ5Q-JO¼HP¨H³+Yÿib

&41 =Â ñ ¨ "%$eÅ 35$ "%$ $5a33 5$ "%$ $5a3%*("%$ $5a3%*("%$
$53-# 18%&5 $5b3ò-eÅ ¥Æß¦ûfƒnV b

V &41¥ 4"3 "%$eÅ

	35$ "%$	35$ "%$	%*("%$	%*("%$	18%&5
%"$	:				
%FFQ TMFF.Q		:			
6-1x)Ø	:	:			
WEE		:		:	
18%&5 1,%&5					:
:.Ë	:				
%."			:		

35$ 4"3 "%$Å

35$È÷×Ï¥ 4"3 "%$Å 35$ "%$ $53-35$ "%$ $53V- ®Ô ÿ/4 Klÿ{ "%$© b

òeÅ ¥ 8ÿ?Àƒnm b ¿ÄñeÅ Ÿª Ð^®7i 4&/4@4"3@ "45"35@4"3 (? © ²TVn7i 4&/4@4"3@ "%"5"@b4"3

m 35$ 4"3 "%$ÿ?Àƒ

6-1x)Ø ÐeÅ -W¥1"dÈÅá XÜ=Â ·7ŸP¨ "%$“ fƒ/ eÅ (³1Ð 6-1 x)Ø x]ýT 1Â

- t V %FFQ TMFFQÉ›ÛùŸ_©b %FFQ TMFFQ6-1x)Ø ^·B¥(? b
- t V¿Bç¨½ Y¡É›õÃŸáíbÐ5eÅ íE¦ûõÃŸáí %." Œ 6-1x)Ø Vxù LCƒ†sÿ?b

%*(4"3 "%$Å

Ð 35$ 4"3 "%$Å M1 %*(4"3 "%$Å ¥Ÿ?„}s(LC BçªÄ ![/+Ä

- t ÚŸ?bHÒ÷y yN•"ÎqLC v–46b
- t ¦û Y¡áí TbÄñ 4"3 "%$© ?5Vn"TVbáí TV¥Â¹†Y¡ Ta Y¡ T Œ9 Tb
- t áíV®Èq * 49L? b
- t ¦û %."báí›î'?3Ï b

ªü
®¿íEYV°¤`Ù? BQŸ 4"3 "%$Ð yNáÌ| 'c«Ï•¨o7Sáíp¥ªE }VáÌ|æ¨
%*(4"3 "%$Å áíB" Y¡b

m ZU %*(4"3 "%$Å ¥ðØmb

m %*(4"3 "%$Å ¥À*f*

"TVV[í• %*(4"3 "%$Å ³1No¥ò[© ?5 ÄñV[šµ ñ[Vi%Y¡Ê4as Oq„šh• ©=¸b'áí7SH eÅ |ïH |"TVÏ¥© ?5b ¿ÄñeÅ 7ý Äñ áí½ K šµ H,]?5b

"TV7i ¥É ¹ Ê " ñ3 sYi% Y¡asOq„šh• ¥=¸ 8nV b

V "TV7i ¥3 •

"TV7i < >		
DI@TFM< >	CJU@XJEUI< >	BUUFO< >
áíY¡	sOq	šh

áí TV¥Â¹ †Y¡ Ta Y¡ T Œ9 Tb

t †Y¡ T Ç 4"3 "%$ 4"3 "%$¥Y¡|$áíb

t Y¡ T 4"3 "%$„ 4"3 "%$¥Y¡]|$áíb

t Œ9 T 4"3 "%$„ 4"3 "%$¥Y¡|$Œ9áíb

&41 KÚ¦û Ê¥ 4"3 "%$Oq KÔ_ %.".¾¥ Ê" " "%$ Ð²T #Btyáí T,]7µîµY¥M1• 8¹

t †Y¡ T Ç9F ÊY¡Ê4• b

t Y¡ T Œ9 T 9F ÊY¡Ê4• # Ê 4"3 "%$Ê4• b

Á^• S/ &41 /Œ•Im¨ ñ'

ÄÕáí T(µ ‹¥" ìT ' *˜„ **bµ1ƒ Õ" ìT¥ 8í• hnV „V
b

V * %." " ìT

*˜ %." " ìT < >	
DI@TFM< >	EBUB< >
Y¡	4"3 "%$

V *˜* %." " ìT

** %." " ìT < >		
TBS@TFM	DI@TFM< >	4"3 "%$ EBUB< >
4"3 "%$	Y¡	4"3 "%$

*˜" ìT¥ 4"3 "%$OqKÚV¦û Ê **" ìT¥ 4"3 "%$OqKÚV¦û Êb

%*(4"3 "%$ ¸YV * 49LLC°¤=i`Ùb * 49L¥ 84•|V¨T© (?•|bVYV
%"5"•| ¤© ²T^Ž›î¥• bVYVÈq¥Â "1#@4"3"%$@%"5"@ 50@%$ 4À * 4
9Lb

:.Ë

e º

ô :r‹ 'È@<°¿HÆYV /˜ö•8H ö <°¿È@„HÆ¥Z_Á3¸FÈÆ V7 ö
•8 ™îÈ]µ 8Ú®ÐÈHÆ¥< „È@vlµ1b'šçÈ@,VHÆ È@i ¿šçHÆ
H :r‹.Ë V¨¿© HÆ< b :.Ë ¥‹¨Æ†dÈ<W "¤Í ©açÊa©ÎÐÈ
@_©©b

&41 Ï¥ :.Ë ÜV Ê!9 V_ 4"3 "%$ È â•| LCHÆ.Ëÿ?b ³1®Èâ¥ý
T T/ ¾.Ë V® 6-1x)Ø eÅb NËÿ?¥¦û/ &41 !¥)Ø?ï„2 Ÿ(P
ÊÂ.Ëa¤Í_©#©Î©‹¨ÆÃ/î¹BÕ „ï¥³%ZÃb

ö1+Ÿ

- t =Â :íq
- t V¥† "%$ýT
- t V¦û{ }VHÆ< ¥ EÈâÐ"3•|
- t ÿ?<vO^¿LC •¨ =ÂT 6-1x)Ø a (1*a $16# 8J '@ v

ÿ?í•

:.Ë V|HÆ ¹Èâ ÂÆbv = 7ªYV5– 4&/403@„751– 4&/403@{7/b &41
=Â "%$V|•| Ä¹"3´ Œ® $16 "3×=›î[/ Tb

:.Ë ¥² nm b

m :.Ë ¥²

VYV7i 4&/4@4"3@506$)@$53î @Ë&¥(|¥Â 35$*0@)"--@4&/4@3&(
Ë È bªÃ TVŒ® 4"3 "%$›îbKÔ²TVYV 35$ "%$eÅ ¤b÷ • hn
c«b

”3 E Ð

eº

”3 E Ð %"${µ ñ ÊY¡ V|”3´ Ð¹KÚ ^ E{ •| “”îÈE1„Ï
ubƒÕ Y¡ %"$¦û|È÷'S{ÆÈâ•I O¦û Y¡¥ÿë]H Ðb

ö1+Ÿ

%"$¥ö1+Ÿ “

- t ñ Ê %"$Y¡
- t ¦û Y¡¥ÿë]H Ð
- t VV 7%% 1 @3-5$¤È â•I
- t cµÅ?o˜?3
- t ¦û %."ÿ?
- t VYVÈq 4"3 "%$ '47S Ðb÷ • hn 4"3 "%$«b
- t V® 6-1x)Ø YVeÅ7i ŸLC›†eÅbhn 6-1x)Ø c«b

†Y¡ %"$¥ÿ?Àƒhnm b 8º hn'c«/Z=¸b

2

Y¡ %"$¥ ñ ÊY¡VLCÿë¥Â ÄñY¡¥{ EÈâ9ØZTn/

%"$D@ 06-5 7%% 1 @ 315%$ "$D@ %/"$

- t 7%% 1 @ 3V5$7%% 1 @ 3-5¥$È â YÈ¹ 7b

m %"$¥ÿ?À*f*

t 1%"$@%"$µ ñŸ÷ Å?o™3î a7i 35$*0@1"%@%"3&$# %."b

VYV7i 35$*0@1"%@%"1$%@%"ç"$Ð^Ž7S È q 4"3"%$'4eÅ Ð@ñ'&
8hnm b

Å?o™3î

Å?o™3î V¨¿3îÅ?o™ ž?o™ 8ýT@ñVnm b

Å?o™3î ¥+Ä "

t ÔqVØ«
Å?o¥ÔqVYV7i 4&/4@4"3@48@'45Ø1< >

GS=FRJH@DML@4&U4D@4G$S@F48@'45&1

YÈ EJH@D¥Ô1q@SU.D)[b

t '–VØ«
VYV7i 4&/4@4"3@%"$@<4$'!Å8o™'– Ø œ¹ a a b

t °@ÊM
7i 4&/4@4"3@%"$@%"3„ÆBt°@ÊM •Á²T „b

t MÊMî
VYV7i 4&/4@4"3@%"$@*9F7 MÊÊMb

¦Û %."

Y¡ %"$¥°¤=ii| %."eÅ V ñ %"$Y¡¥{ É›!ÂbYV¥Â
4&/4@4"3@%"$@%*(@'4@$D&M¤LÀ %"$ DM*L4@%"5"@V6¤5À %"$@%'L5C°¤=
i`Ùb

÷ • hn %."c«b

m Å?o™3î ¥ýT@ñ

7i V

ªü /Z7i ¥sËö1[ÿ?¹ö i,Q˜`Ù=i¥ 8¨½b

.Ë

ë	í•	¹¶	`ÙË˜
(5–!ÂÐeÅ7i			
4&/4@4"3@506$)@$53- @3&(	3&(eÅ	Y ''	•
4&/4@4"3@506$)@$53- @3&(	3&(eÅÐ ÿ	Y ''	º
4&/4@4"3@506$)@&/"#-&@3&(	3&(eÅÐýT 4&5	Y '' $	•
4&/4@4"3@506$)@5)3&4	5-3&(„5– ¥•´!Â	Y '' $	•
4&/4@4"3@506$)@5)3&4	5-3&(„5– ¥•´!Â	Y ''	•
4&/4@4"3@506$)@5)3&4	5-3&(„5– ¥•´!Â	Y ''	•
4&/4@4"3@506$)@5)3&4	5-3&(„5– ¥•´!Â	Y ''	•
4&/4@4"3@506$)@5)3&4	5-3&(„5– ¥•´!Â	Y '' $	•
4&/4@4"3@506$)@065 @35&(	„5– ¥9”	Y ''	º
4&/4@4"3@506$)@065 @35&(	„5– ¥9”	Y ''	º
4&/4@4"3@506$)@065 @35&(	„5– ¥9”	Y ''	º
4&/4@4"3@506$)@065 @35&(	„5– ¥9”	Y '' $	º
4&/4@4"3@506$)@065 @35&(	„5– ¥9”	Y ''	º
4"3 "%$eÅ7i			
4&/4@4"3@45"35@'03$&@4"3(	"%$„ "%$eÅ	Y '' $	•
4"3 "%$`ÙeÅ			
4&/4@4"3@3&"%@$53-@3&(	4"3 "%$” Ð•"eÅ	Y ''	•
4&/4@4"3@.&"4@45"35 @3&(	4"3 "%$ ÐeÅÐ ÿ	Y ''	º
4"3 "%$eÅ7i			
4&/4@4"3@3&"%@$53- @3&(	4"3 "%$” Ð•"eÅ	Y ''	•
4&/4@4"3@.&"4@45"35 @3&(	4"3 "%$ ÐeÅÐ ÿ	Y ''	º
6-1x)Ø ¥Â7i			
4&/4@6-1@$1@4-&&1@$	6-1x)Ø ¥¤äÛù	Y ''	•
5–šh¥Â7i			

4&/4@4"3@"55&/ @3&(	Äñ5–¥ Êšh	Y ''	•
4&/4@4"3@"55&/ @3&(	Äñ5–¥ Êšh	Y ''	•
%"$eÅ7i			
4&/4@4"3@%"$@$53- @3	&%"$eÅ	Y ''	•
4&/4@4"3@%"$@$53- @3	&%"$[eÅ	Y '' $	•

“¶9L

ë	í•	¹¶	`ÙZT
4"3 "%$„ "%$ Y¨¥Â7i			
"1#@4"3"%$@$53-@3&(	4"3 "%$Y¨¥Â	Y	•
"1#@4"3"%$@$53- @3&(	4"3 "%$Y¨¥Â	Y	•
"1#@4"3"%$@'4.@3&(	4"3 "%$ '4•"Ûù¥Â	Y	•
4"3 "%$"TV7i			
"1#@4"3"%$@4"3 @1"55@	5TÑV @3&(	Y $	•
"1#@4"3"%$@4"3 @1"55@	5TÑV @3&(	Y	•
"1#@4"3"%$@4"3 @1"55@	5TÑV @3&(	Y	•
"1#@4"3"%$@4"3 @1"55@	5TÑV @3&(	Y	•
4"3 "%$"TV7i			
"1#@4"3"%$@4"3 @1"55@	5TÑV @3&(	Y $	•
"1#@4"3"%$@4"3 @1"55@	5TÑV @3&(	Y	•
"1#@4"3"%$@4"3 @1"55@	5TÑV @3&(	Y	•
"1#@4"3"%$@4"3 @1"55@	5TÑV @3&(	Y	•

35$ * 0

µ1 35$ * ¥0M17i V h•n *0@.6„9(1*0ŒÐ ” c«Ï 7i V ¥=¸b

7 i

.Ë

3 F H J T U F S 4 & / 4 @ 4 " 3 @ 3 & " % @ $ 5 3 @ 3 & (Y

SFTFSWFE	4&/4@4"3 @%"5"@*/7 4&/4@4"3 @%*(@'03$&	SFTFSWFE	4&/4@4"3 @4".1-&@#*3	4&/4@4"3 @4".1-&@$:$	4&/4@4"3 @$-,@%*7
jR kN k3 kd ke R3 Rd Re R8 3 d y					

_ 2 b 2 i

4 & / 4 @ 4 " 3 @ % " 5 " @ * / 7 " 3 " % $ " b •

4 & / 4 @ 4 " 3 @ % * (@ ' 0 3 $ & " 3 " % $® % * (" % $ $ 5 3 4 " 3 " % $® 3 5 $ " % $ $ 5 3 -e Å b •

4 & / 4 @ 4 " 3 @ 4 " . 1 - & @ # * ' 5 % $ ¥ Ê z Ê Ê Ê Ê b •

4 & / 4 @ 4 " 3 @ 4 " . 1 - & @ $: $ 3 & " % $ ¥ • " Û ù b •

4 & / 4 @ 4 " 3 @ $ - , @ % * 7 b •

3 F H J T U F S 4 & / 4 @ 6 - 1 @ $ 1 @ 4 - & & 1 @ $: $ @ 3 & (Y

jR y

_ 2 b 2 i

4 & / 4 @ 6 - 1 @ $ 1 @ 4 - & & 1 @ $ 6 $ 1 @ ¥ ¤ ä Û ù b •

3FHJTUFS 4&/4@4"3@45"35@'03$&@3&(D

SFTFSWFE	4&/4@4"3 @4501 4&/4@4"3 @4501	4&/4@1$@*/*5	SFTFSWFE 4&/4@6-1@$1@45"35@501 4&/4@6-1@$1@'03$&@45"35@501 4&/4@4"3 @18%&5@$$5 4&/4@4"3 @&/@5&45 4&/4@4"3 @#*5@8*%5) 4&/4@4"3 @#*5@8*%5)
jR k9	kj kk	kR RR Ry	N 3 d 8 9 j k R y

_2b2i

4&/4@4"3 @45TQ14"3"%$¥ Ðb •

4&/4@4"3 @45TQ14"3"%$¥ Ðb •

4&/4@1$@*6/*1x)Ø ¥ SÄ 1$b •

4&/4@6-1@$1@45"35@'506-1x)Ø Ç SFH@VMQ@DQ@GPSHDF@TUBSU@ rb •

4&/4@6-1@$1@'03$&@45"35@'506-1x)Ø ®Èq î 6-1x)Ø ®9H îb •

4&/4@4"3 @18%&5@'03$&@18%&5@1"$5q S© ¥È¸Ø~b •

4&/4@4"3 @&/@4"34&5@&/@5&45SFH@TBS @EJHHµ@GPSDF

4&/4@4"3 @#*5@8*%5%$¥Êz Ê Ê Ê Êb •

4&/4@4"3 @#*5@8*%5%$¥Êz Ê Ê Ê Êb •

3FHJTUFS 4&/4@4"3@"55&/ @3&(Y

jR	y
Y ''''''''	

_2b2i

4&/4@4"3@"55&/ @Ä3&(¥šh E# E# E# E#b < ¨>
¿í• "%$ @$a) < ¨¿í• "%$ @$)[NËwb •

3FHJTUFS 4&/4@4"3@"55&/ @3&(Y

jR	y
Y ''''''''	

_2b2i

4&/4@4"3@"55&/ @Ä3&(¥šh E# E# E# E#b < ¨>
¿í• "%$ @$a) < ¨¿í• "%$ @$)[NËwb •

3FHJTUFS 4&/4@4"3@&"4@45"35 @3&(Y

| | | |

3FHJTUFS 4&/4@4"3@506$)@5)3&4 @3&(Y D

4&/4@506$)@065@5) 4&/4@506$)@065@5)

jR Re	R8 y
Y	Y

_2b2i

4&/4@506$)@065@5) ¥•´b •

4&/4@506$)@065@5) ¥•´b •

3FHJTUFS 4&/4@4"3@506$)@5)3&4 @3&(Y

4&/4@506$)@065@5) 4&/4@506$)@065@5)

jR Re	R8 y
Y	Y

_2b2i

4&/4@506$)@065@5) ¥•´b •

4&/4@506$)@065@5) ¥•´b •

3FHJTUFS 4&/4@4"3@506$)@5)3&4 @3&(Y

4&/4@506$)@065@5) 4&/4@506$)@065@5)

jR Re	R8 y
Y	Y

_2b2i

4&/4@506$)@065@5) ¥•´b •

4&/4@506$)@065@5) ¥•´b •

3FHJTUFS 4&/4@4"3@506$)@5)3&4 @3&(Y

4&/4@506$)@065@5) 4&/4@506$)@065@5)

jR Re	R8 y
Y	Y

_2b2i

4&/4@506$)@065@5) ¥•´b •

4&/4@506$)@065@5) ¥•´b •

3FHJTUFS 4&/4@4"3@506$)@5)3&4 @3&(Y D

4&/4@506$)@065@5) 4&/4@506$)@065@5)

jR Re	R8 y
Y	Y

_2b2i

4&/4@506$)@065@5) ¥•´b •

4&/4@506$)@065@5) ¥•´b •

3FHJTUFS 4&/4@4"3@506$)@065 @3&(Y

4&/4@506$)@.&"4@065 4&/4@506$)@.&"4@065

jR Re	R8 y
Y	Y

_2b2i

4&/4@506$)@.&"4@065 ¥9" bº

4&/4@506$)@.&"4@065 ¥9" bº

3FHJTUFS 4&/4@4"3@506$)@065 @3&(065 Y

4&/4@506$)@.&"4@065 4&/4@506$)@.&"4@065

jR	Re R8	y
Y	Y	_2b2i

4&/4@506$)@.&"4@065 ¥9” bº

4&/4@506$)@.&"4@065 ¥9” bº

3FHJTUFS 4&/4@4"3@506$)@065 @3&(065 Y

4&/4@506$)@.&"4@065 4&/4@506$)@.&"4@065

jR	Re R8	y
Y	Y	_2b2i

4&/4@506$)@.&"4@065 ¥9” bº

4&/4@506$)@.&"4@065 ¥9” bº

3FHJTUFS 4&/4@4"3@506$)@065 @3&(065 Y D

4&/4@506$)@.&"4@065 4&/4@506$)@.&"4@065

jR	Re R8	y
Y	Y	_2b2i

4&/4@506$)@.&"4@065 ¥9” bº

4&/4@506$)@.&"4@065 ¥9” bº

3FHJTUFS 4&/4@4"3@506$)@065 @3&(065 Y

4&/4@506$)@.&"4@065

4&/4@506$)@.&"4@065

jR	Re R8	y
Y	Y	_2b2i

4&/4@506$)@.&"4@065 ¥9” bº

4&/4@506$)@.&"4@065 ¥9” bº

3FHJTUFS 4&/4@4"3@506$)@$53-@@3&(Y

SFTFSWFE

4&/4@506$)@.&"4@&/@$53

4&/4@506$)@4-&&1@$:$&4

4&/4@506$)@45"35@'0$&/

4&/4@506$)@45"35@'4.&/

4&/4@506$)@45"35@'%0/&/

4&/4@506$)@.&"4@&/

jR	jy	kN	R9	Rj	Rk	RR	Ry	N	y
		Y						Y	_2b2i

4&/4@506$)@.&"4@&/b@$SFH@UPVDI@N•FBT@FO

4&/4@506$)@4-&&1@$:$&4äÛùb •

4&/4@506$)@45"35@'0$&'4.®Èq î ®9H îb •

4&/4@506$)@45"35@î&/ '4.' SFH@UPVDI@!ÂUHµSb @GP•SDF

4&/4@506$)@45"35@'4.@&/ 506$)@45"35 506$)@(91%4.eÅ ®7 i eÅb •

4&/4@506$)@.&"4@%04&!Â }V(© X›îbº

4&/4@506$)@.&"4@&/i }V 8sž(·¥5–bº

3FHJTUFS 4&/4@4"3@506$)@&/"#-&@3&(Y

SFTFSWFE	4&/4@506$)@1"%@065&/	4&/4@506$)@1"%@065&/	4&/4@506$)@1"%@803,&/
jR jy kN ky	RN Ry	N y	
	Y ''	Y ''	Y ''

_2b2i

4&/4@506$)@1"%@06ç5&/4&5Òœï 3î¥Êm Ç' 4&5ïÀ µ ñ5–s
ž(· Œj 4&5sž(·b •

4&/4@506$)@1"%@06ç5&/4&5Òœï 3î¥Êm Ç' 4&5ïÀ µ ñ5–s
ž(· Œj 4&5sž(·b •

4&/4@506$)@1"%@80ç3I©&/µr 4&5¥Êmb •

3FHJTUFS 4&/4@4"3@3&"%@$53-&@3&(Y

SFTFSWFE	4&/4@4"3 @%"5"@*/7	4&/4@4"3 @%*(@'03$&@	SFTFSWFE	4&/4@4"3 @4".1-&@#*3	4&/4@4"3 @4".1-&@$.$-	4&/4@4"3 @$-,@%*7
jR jy kN	k3	kd R3	Rd Re	R8 3	d y	

_2b2i

4&/4@4"3 @%"5"@*/47"3 "%$" b •

4&/4@4"3 @%*(@'034$"&3 "%$® %*("%$ $53-18%&5 $53eÅ ® 35$
"%$ $53e-Åb •

4&/4@4"3 @4".1-&@#*'5%$¥Êz Ê Ê Ê Êb •

4&/4@4"3 @4".1-&@$4$3&'%$¥•"Ûùb •

4&/4@4"3 @$-,@H%ò*7Ô b •

3FHJTUFS 4&/4@4"3@&"4@45"35 @3&(Y

4&/4@4"3 @&/@1"%@'03$&	4&/4@4"3 @&/@1"%	4&/4@.&"4 @45"35@'03$& 4&/4@.&"4 @45"35@4"3 4&/4@.&"4 @%0/&@4"3	4&/4@.&"4 @%"5"@4"3	
jR jy RN R3 Rd Re R8 y				_2b2i

4&/4@4"3 @&/@1"%@'04"3&"%$5–P?Êm®ÈqeÅ ® 6-1x)Ø e Åb •

4&/4@4"3 @&/@41"3%$5–P?Êm Ç' SFH@TBS @FO@HJBE@GPSDF

4&/4@.&"4 @45"35@'034"3 "%$eÅ 35$®Èq î ® 6-1xeÅ îb •

4&/4@.&"4 @45"35@"43"3%$eÅ 35$7S Ð Ç' SFH@NFBT @TUBSU@GPSD Hµrb •

4&/4@.&"4 @%0/&@V4"3 "%$ ÐX›îbº

4&/4@.&"4 @%"5"@4"3%$" bº

3FHJTUFS 4&/4@4"3@%"$@$53- @3&(Y

SFTFSWFE	4&/4@%"$@$-,@*/7 4&/4@%"$@$-,@'03$&@)*() 4&/4@%"$@$-,@'03$&@-08 4&/4@%"$@%*(@'03$&	SFTFSWFE	4&/4@48@50/&@&/	4&/4@48@'45&1
jR ke k8 k9 kj kk kR Rd Re R8 y				_2b2i

4&/4@%"$@$-,@*Q/7 1%"$@$-, ,É›Q b •

4&/4@%"$@$-,@'03$&1@)*()@$<,Å| b •

4&/4@%"$@$-,@'03$&1@%-"08@$<,Å| b •

4&/4@%"$@%*(@'03$&8 %"$P¨ %." %"$ %"$,P¨ %."b •

4&/4@48@50/&@R&7 $8?3 Ì¨ $8?3 b •

4&/4@48@'45&81?3 ¥Ôq¨ V¨¿Ø«o™Ôqb •

3 F H J T U \ F S 4 & / 4 @ 4 " 3 @ % " $ @ $ 5 3 - @ 3 & (Y D

SFTFSWFE	4&/4@%"$@$8@&Ê	4&/4@%"$@$8@&Ê	4&/4@%"$@*/7	4&/4@%"$@*/7	4&/4@%"$@4$"-&	4&/4@%"$@4$"-&	4&/4@%"$@%$	4&/4@%"$@%$
jR ke	k8	k9	kj kk	kR ky	RN R3	Rd Re	R8 3	d y

_2b2i

4 & / 4 @ % " $ @ $ 8 @ &Ê/ 4 $ 8 ? 3 ¹ 1 % " $ @ % " $ ¥ " Ÿ ÷ Ê 4 7 i
S F H @ Q E B D @ E 1 B % D " $ @ % " $ ¥ " Ÿ ÷ b •

4 & / 4 @ % " $ @ $ 8 @ &Ê/ 4 $ 8 ? 3 ¹ 1 % " $ @ % " $ ¥ " Ÿ ÷ Ê 4 7 i
S F H @ Q E B D @ E 1 B % D " $ @ % " $ ¥ " Ÿ ÷ b •

4 & / 4 @ % " $ @ *%7 " $, Q © … Ê Q î µ Ê Q . 4 # Q " . 4 #
“ ¥ î µ Ê b •

4 & / 4 @ % " $ @ *%7 " $, Q © … Ê Q î µ Ê Q . 4 # Q " . 4 #
“ ¥ î µ Ê b •

4 & / 4 @ % " $ @ 4 $ " % & $ b •

4 & / 4 @ % " $ @ 4 $ " % & $ b •

4 & / 4 @ % " $ @ % $ $ $ 8 ? 3 ¥ ° @ Ê M b •

4 & / 4 @ % " $ @ % $ $ $ 8 ? 3 ¥ ° @ Ê M b •

Ú)“¶9L

3FHJTUFS "1#@4"3"%$@$53-@3&(Y

SFTFSWFE	"1#@4"3"%$@%"5"@50@*4	"1#@4"3"%$@%"5"@4"3@4&-	"1#@4"3"%$@4"3@1"55@1@$-&"3	"1#@4"3"%$@4"3@1"55@1@$-&"3	"1#@4"3"%$@4"3@1"55@-&"3	"1#@4"3"%$@4"3@1"55@-&"3	"1#@4"3"%$@4"3@$-,@%*(	"1#@4"3"%$@4"3@$-,@("5&%	"1#@4"3"%$@4"3@4&-	"1#@4"3"%$@803,@.0%&	"1#@4"3"%$@4"3@.69	"1#@4"3"%$@45"35	"1#@4"3"%$@45"35@'03
jR kd	ke	k8	k9	kj	kk RN	R3 R8	R9 d	e	8	9 j	k	R	y

_2b2i

"1#@4"3"%$@%"5"@50*@4 4%.{Æ” Ÿ1 4"3"%$ * 4{Æ” Ÿ1 (1*0 ”b •

"1#@4"3"%$@%"5"@4"3@ ®&@®TFMÊ{ ” ¥ .4#I’ NHsOq,‹v ¿ Ê NH 4"3"%$Oq‹¹ Êb •

"1#@4"3"%$@4"3 @1"55@ %@($-%"$3 $5¥3"-TV·Œb,b •

"1#@4"3"%$@4"3 @1"55@ %@($-%"$3 $5¥3"-TV·Œb,b •

"1#@4"3"%$@4"3 @1"545"@-"&%/$ }V"TV¥ Êb •

"1#@4"3"%$@4"3 @1"545"@-"&%/$ }V"TV¥ Êb •

"1#@4"3"%$@4"3@$-4,"@3%ÒÔ7sÔ b •

"1#@4"3"%$@4"3@$-,@(-"Ê5&h%SÄ¹ C b •

"1#@4"3"%$@4"3@44&"-3 4"3 ¾!Âa¨†Y¡ 4"3 Tb •

"1#@4"3"%$@803,@.0%Y&¡ T Y¡ T Œ9 Tb •

"1#@4"3"%$@4"3 @.6419'3 "%$® %*("%$ $5e3Å 4"3 "%$® 18%&5 $53-eÅb •

"1#@4"3"%$@45"-3Ê5 h SÄ¹ b •

"1#@4"3"%$@45"35@'0-Ê$k& SÄ¹ b •

3FHJTUFS "1#@4"3"%$@$53- @3&(Y

SFTFSWFE

"1#@4"3"%$@4"3 @*/7

"1#@4"3"%$@4"3 @*/7

"1#@4"3"%$@."9@.&"4@/6.

"1#@4"3"%$@.&"4@/6.@

jR	RR Ry	N	3	R	y

_2b2i

"1#@4"3"%$@4"3 @{*ÆÆ7 %*("%$ $5¥3"- $Q {Æ %*("%$ $5¥3-
" Ã$Q b •

"1#@4"3"%$@4"3 @{*ÆÆ7 %*("%$ $5¥3"- $Q {Æ %*("%$ $5¥3-
" Ã$Q b •

"1#@4"3"%$@."9@.&"4@Ø/6Ð" b •

"1#@4"3"%$@.&"4@/6.@=*Ê*I5 SÄ¹ C b •

3FHJTUFS "1#@4"3"%$@'4.@3&(Y

"1#@4"3"%$@4".1-&@$.&

SFTFSWFE

jR k9	9d k9

_2b2i

"1#@4"3"%$@4".1-&@$."$Ĵ-ù&b •

3FHJTUFS "1#@4"3"%$@4"3 @1"55@5"# @3&(Y $

jR y
Y ' ' ' '

_2b2i

"1#@4"3"%$@4"3 @1"55@5}'¥ @4'3&("%$¥"TV Ä[]¨ 3«
< > QBUUFSOa<@DIBOOFMa<@CJUa<@QXBJUEUUFSO a<@BUUFSOQVBUUJPO
UFSO @DIBOOFM

3FHJTUFS "1#@4"3"%$@4"3 @1"55@5"# @3&(Y

jR y

Y '''' _2b2i

"1#@4"3"%$@4"3 @1"55@5}'¥ @4'33&("%$¥"TV Ä[]¨ 3«

< > QBUUFSOa<@DIBOOFM QBUUFSOa<@CJUQBUUFSO @BUUFOVBUJPO

UFSO @DIBOOFM

3FHJTUFS "1#@4"3"%$@4"3 @1"55@5"# @3&(Y

jR y

Y '''' _2b2i

"1#@4"3"%$@4"3 @1"55@5}'¥ @4'33&("%$¥"TV Ä[]¨ 3«

< > QBUUFSOa<@DIBOOFM QBUUFSOa<@CJUQBUUFSO @BUUFOVBUJPO

UFSO @DIBOOFM

3FHJTUFS "1#@4"3"%$@4"3 @1"55@5"# @3&(Y

jR y

Y '''' _2b2i

"1#@4"3"%$@4"3 @1"55@5}'¥ @4'33&("%$¥"TV Ä[]¨ 3«

< > QBUUFSOa<@DIBOOFM QBUUFSOa<@CJUQBUUFSO @BUUFOVBUJPO

QBUUFSO @DIBOOFM

3FHJTUFS "1#@4"3"%$@4"3 @1"55@5"# @3&(Y $

jR y

Y '''' _2b2i

"1#@4"3"%$@4"3 @1"55@5}'¥ @4'33&("%$¥"TV Ä[]¨ 3«

< > QBUUFSOa<@DIBOOFM QBUUFSOa<@CJUQBUUFSO @BUUFOVBUJPO

UFSO @DIBOOFM

3FHJTUFS "1#@4"3"%$@4"3 @1"55@5"# @3&(Y

jR y

Y ' ' ' ' _2b2i

"1#@4"3"%$@4"3 @1"55@5}'¥ @4'33&("%$¥"TV Ä[]¨ 3«

< > QBUUFSOa<@DIBOOFM QOBFUWUFSOa<@CJU>@QXBJUEUUFSO a@<BUUFSOQVBUUJPO

UFSO @DIBOOFM

3FHJTUFS "1#@4"3"%$@4"3 @1"55@5"# @3&(Y

jR y

Y ' ' ' ' _2b2i

"1#@4"3"%$@4"3 @1"55@5}'¥ @4'33&("%$¥"TV Ä[]¨ 3«

< > QBUUFSOa<@DIBOOFM QOBFUWUFSOa<@CJU>@QXBJUEUUFSO a@<BUUFSOQVBUUJPO

UFSO @DIBOOFM

3FHJTUFS "1#@4"3"%$@4"3 @1"55@5"# @3&(Y

jR y

Y ' ' ' ' _2b2i

"1#@4"3"%$@4"3 @1"55@5}'¥ @4'33&("%$¥"TV Ä[]¨ 3«

< > QBUUFSOa<@DIBOOFM QOBFUWUFSOa<@CJU>@QXBJUEUUFSO a@<BUUFSOVBUJPO

QBUUFSO @DIBOOFM

35$ * 0

µ1 35$ * ¥0M17i h•n *0@.6„9(1*0ŒÐ ” c«Ï 7i ¥=¸b

Ñ®ÿ{x)Ø 6-1

À•

Ñ®ÿ{x)Ø 6-1 $PQSPD^FBTÕÏÿP{S®¥x)Ø !! V ö"d)‰Ë 4P$"dÉÆ
%FFQ TMÿFFÛQ È ¸7?€YVi% 35$Ï¥ ¨ñ½ `Ù"¶!!a=†.Ë # 35$
7i b 6-1x)Ø ¥ö1‹¨ÆÈß "Bt³1 £K®ÿ{¥ƒƒ/ YV"† î 9H
€Yµ Òœ $16¥‹¨b

ö1+Ÿ

- t V`ÙK ,#43".35$¤Î=i %i·7„"
- t •¨ .)[35$@'"45@$ÒÔq
- t ¦ûžÈ T„ %FFQ TMFFQ
- t VÒœ$16_ $16?ÂÏ
- t V`Ù"¶!!a=†.Ë # 35$7i
- t •¨ ñ ÊY¨7i 3 3 É›" T„=i`Ù
- t •¨ ñ Ê¨ 9" 7i 4UBHF@ÙÙÙU"-6·7É› Ti¨¿ +6.1·7

m 6-1x)Ø 'O

ÿ?í•

6-1x)Ø ^BÕVIñµK ÿ '4. V $16ÉÆ%FFQ TMÿHÿÿQbx)Ø ¦û†sY
¨ $16·7 VxùÉ›Bt¯ † ØbN“ Î¦ûBt+y¥ 35$eÅÐ“¶!!eÅ·7bÐ
$16B" 6-1x)Ø 9V`Ù ,# 43". 35$¤Î=ib9žyÂN ƒv=iÜÈ$¨¿i%Btx
)Ø „ $16¥Y¨·7b

6-1x)Ø V®Èqñ½ —qçH ÛùŸ î iYV³›)"-5·7T¸bx)Ø ¥ÿ?dÈ<v
V[YV=Â·7„ 35$7i `Ù 35$×Ï¥+¬îµ vb 6-1x)Ø V“ ‹¨ÆßÏî¹
$16¥µï€ .À|} $16+Y^ Bt ÿ{“ùË¥‹¨Ïb 6-1x)Ø ¥ 'O Vnm
b

·7"

6-1x)Ø V¦û/ ·7

- t Ø”Ð† "-6
- t F Ð” i% -%a 45a 3&(@3#%3&(@83
- t K À ¹¶ +6.1
- t 5Øñ½³› 8"*5„)"-5
- t eÅx)Ø ¥¤äÛù 4-&&1
- t Òœ$16#Ð 4P$Y• 8",&
- t © "%$
- t *¨$ÐLY• * $@3„% $@83

6-1x)Ø ·7¥ìTVnm b

0Q$PEF	0QFSBOET

m 6-1x)Ø ¥·7ìT

ô 0QFSB¥OÊ]]]Bñ 0Q$PÊVF‹ Õ,] Tb1Â "-6? ³› Õ,]¥Ø”„†
Ø +6.19V³›µHqK aíHqK a' K #M K © Õ™T¥K b

6-1x)Ø ¥îµ·7(%ç¹ ÊbYVƒB" ·7 x)Ø ñ½'V¤ž³›bñ½=†¥³›
(•¨ ÊÙ¶b¾ñ½ 8i% v ¨¥¤Î=iu 35$@4-08@.¹&¶.S¶¹ Y @
ž Y @ ''' ,#ö $16Vnb

'ÓÏ 0Q$PÊ¥FT¹ ÊY Ï }V Êz E}VEÉÅ Y}V 0Q$PÊ¥F Y a b

"-6 Ø”Ð† Ø

Ø”† †í "-6V[É›Ø”„† Ø `¹x)Ø 7i Ïi%¥”´ ·7Ïi%¥ë'
´b
8V[¦û¥ ØË˜Â/

- t Ø” F "%%h 46#

t † Ð "/%„ 03

t MÊ PM -4)„·M 34)

t 7i ®´ Mî .07&

t 9" 7i T 45"(&@3a445"(&@*„/$45"(&@%&$

Ð5 0Q$PBMF] ŒVYV!Âx)Ø ·7 < Ê> Ê4+ç¥Ø"„† Øb

7i "´¥ Ø

E	C	"-6@TFM	3TSD3TSD3ETU

m ·7Ë˜ 7i "´¥ "-6 Ø

Âm îU 'x)Ø ·7 < Ê>!Â¹ CH "-6| x)Ø 7i 3< Ïi>%¥=¸É › Ø ØË˜5|%¿·7¥ "-6@TFM <Ê 8!>ÂZTn/V b

0QFSBOÓE nm
"-6@TFM-6 ØË˜
3ETU 7i 3< '>S¹¶7i
3TSD 7i 3< ›¹¶7i
3TSD 7i 3< ›¹¶7i

"-6@TFM7	ØË˜	í•
"%%	3ETU3TSD3TSD	F
46#	3ETU3TSD3TSD	h
"/%	3ETU3TSD3TSD	† Ð
03	3ETU3TSD3TSD	†
.07&	3ETU3TSD	7i ®´
-4)	3ETU3TSD3TSD	† PM
34)	3ETU3TSD3TSD	† ·M

V 7i "´¥ "-6 Ø

ÿi

t "%% 46# ØV¨¿!Â b" "-6n S½Êb

t îµ "-6 Ø(V¨¿!Â b" "-6,S½Êb

·7ë'´¥ Ø

E	C	"-6@TFM	*NN	3TSD3ETU

m ·7Ë˜ ·7ë'´¥ "-6 Ø

Âm îU 'x)Ø ·7 < Ê>!Â¹ CH "-6| x)Ø 7i 3< „>7 < Ê>
i%¥ë'´É› Ø ØË˜|%¿·7¥ "-6@TFM <Ê 8!>ÂZTn/V b

0QFSBOíE nm
"-6@TFM-6 ØË˜
3ETU 7i 3< '>S¹¶7i
3TSD 7i 3< ÷¹¶7i
*NN ·7ë'´ Êµ˜|"

"-6@TFM	Ø	í•
"%%	3ETU3TSD*NN	F
46#	3ETU3TSD*NN	h
"/%	3ETU3TSD*NN	† Ð
03	3ETU3TSD*NN	†
.07&	3ETU*NN	7i ®´
-4)	3ETU3TSD*NN	† PM
34)	3ETU3TSD*NN	† ·M

V ·7ë'´¥ "-6 Ø

ÿi

t "%% 46# ØV¨¿!Â b" "-6n S½Êb

t îµ "-6 Ø(V¨¿!Â b" "-6,S½Êb

¨ 9" 7i "´¥ Ø

E	E	"-6@TFM	*NN	

m ·7Ë˜ ¨ 9" 7i ¥ "-6 Ø

Âm îU 'x)Ø ·7 < Ê>!Â¹ CH "-6| Ê7i 4UBHF@DOUÊ¨ 9" 7i ¾h
×Â T ØË˜|%¿·7¥ "-6@TFM <Ê 8!>ÂZTn/V b

0QFSBOíE nm
"-6@TFM-6 ØË˜
4UBHF@DOUÊ¨ 9" 7i Vi%ÖÌ/S©M
*NN ·7ë'´ Ê"

"-6@TFM	Ø	í•
45"(&@*/$4	UBHF@4DJCUHF@*NNU	9" 7i ¾9
45"(&@%&4$	UBHF@4DJCUHF@*NNU	9" 7i ¾h
45"(&@34	UBHF@DOU	¨ 9" 7i ¯Ê

V ¨ 9" 7i ¥ "-6 Ø

4 5&¶ i %" À = i

E	C	C	0GGTFU	C	3ETU	3TSD

m ·7Ë˜ 45

0QFSBÓE nm
0GGTFU Êµ˜|" †Ê¹ Ê3
3TSD 7i 3< >=³i%¥ Ê"
3ETU 7i 3< '>S7i ¹¶ †Ê¹ Ê3

í•

¾·7V| 3TSÏD=¥ Ê"i%À=i¹¶¹ 3ETU0GGÏ¥®ÚÏ3Ïb¾=iÏ¥Úö3¹ñ½9"
1$Ï¥=¸ [3¹†Ê PM Ê † 3ETU

.FN < 3ETU 0GGTFU >\ ^ \1$< > C 3ETU 3TSD< >

‹¨ñ½VYVÚö3' 6-1ñ½Ï¥'H·7 8_=i•Æ I¹=¸b

ÿi

t ¾·7Ç?[Ê3¹†ÊÉ›`Ùb

t ©...ƒƒ/ 3TSÏD=¥ Ê"¥ùÇ?i%À=i¥®ö3Ï '¥ù,V?| 3TSÏD=À=i
¥Úö3Ïb

t .FN•Æ¥^ 35$@4-08@¤&Ê=i 6-1x)Ø ¥¹¶ 'M'¿ö $16¥¹¶
Y b

-%&¶ V = i F "

E		0GGTFU		3TSD	3ETU

m ·7Ë˜ -%

0QFSBÓE nm
0GGTFU Êµ˜|" †Ê¹ Ê3
3TSD 7i 3< >="S7i ¥¹¶ †Ê¹ Ê3
3ETU 7i 3< '>S7i

í•

¾·7V|=i¹¶ 3TSDPGGÏ¥®ÚÏ3F À"S7i 3ETU

3ETU< > .FN< 3TSD 0GGTFU >< >

ÿi

t ¾·7Ç?[Ê3¹†ÊÉ›`Ùb

t ©...ƒƒ/ 3TSÏÇÚVF =i¥®ö3 '¥ù,V?|=i¥Úö3F À 3TSÏÚ

t F ¥ .FN|i%À 35$@4-08@¤Ô=iÏ 6-1x)Ø ¥¹¶ 'M'¿ö $16¥¹¶
Y b

+6.1&¶K À' ¹¶

E	C	5ZQF	4FM		*NN"EES	3ETU

m ·7Ë˜ +6.1

0QFSBÓÍE nm
3ETU 7i 3< %i³K À¥"S¹¶
*NN"EES Ê¹¶ †Ê¹ Ê3
4FM K "S¹¶Ÿ÷
*NN"EiÊ%S¥¹¶
3ETiÊ%¥¹¶
5ZQF K Ë˜
íHqK
µHqK Ç'KªBQ "-6 Ø!Â ,S½ÊHK
µHqK Ç'KªBQ "-6 Ø!Â n S½ÊHK

ÿi
îµK ¹¶([Ê3¹†Êb

í•
¾·7V[žx)Ø K À+ç¹¶ K V[¹íHqK µHqK b

+6.13&¶K ÀM ¹¶ ¿ 3 7i '

E	C	4UFQ	$POE	5ISFTIPME

m ·7Ë˜ +6.13

0QFSBÓÍE nm
4UFQ M MÊ †Ê¹ Ê3
ÂT 4UFQ< 5 1$ 1$ 4UFQ< >
ÂT 4UFQ< 5 1$ 1$ 4UFQ< >
5ISFTIPKMEHq•´ n/Z $POE
$POE K Hq
ÂT 3 5ISFTIPKME
ÂT 3 5ISFTIPKME

ÿi
îµK ¹¶([Ê3¹†Êb

í•
ÂTK Hq '1 3 7i ¥´Ð 5ISFTIPKME ¾·7V[žx)Ø K À ñM ¹
¶b

+6.14&¶K ÀM ¹¶ ¿¨ 9” 7i ‘

E	E	4UFQ	$POE	5ISFTIPME

m ·7Ë˜ +6.14

0QFSBÓÍE nm
4UFQ M ÊM †Ê¹ Ê3
ÂT 4UFQ< 5 1$ 1$ 4UFQ< >
ÂT 4UFQ< 5 1$ 1$ 4UFQ< >
5ISFTIPKMEq•´ n/Z $POE
$POE K Hq
9 ÂT 4UBHF@ÐKOSUFTIPKME
ÂT 4UBHF@5DDSFUTIPKME
ÂT 4UBHF@ÐKOSUFTIPKME

ÿi

t µ1¨ 9” ¥M1!Â hn "-6¨ 9” c«b

t îµK ¹¶([Ê3¹†Êb

í•
ÂTK Hq '1 4UBHF¨@9D”O7U ¥´Ð 5ISFTI•PMEE ¾·7V[žx)Ø K
À ñM ¹¶b

)"-5&¶²•ñ½

E	

m ·7Ë˜)"-5

í•
¾·7V[žx)Ø ÉÆ È Tb

ÿi
³›¾·7ª 6-1x)Ø ¥—q9H |7S9Hb

8",&&¶Òœ‰Ë

E	C	

m ·7Ë˜ 8",&

í•
¾·7V[ž 6-1x)Ø _ 35$eÅ ?ÂÏ b

t ’ 4P$)¿ %FFQ TMTFF¾Q·7VÒœ 4P$b

t ’ 4P$)¿ %FFQ T-M¥FTQH ÂT 35$@$/5-@*/5@&7'l@Â&(
35$@$/5-@6-1@$1@*Ê5¾@·&7/'(? 35$Ï b

4-&&&¶!Â—q9H ¥ÒœÛù

E	C		TMFFQ@SFH

m ·7Ë˜ 4-&&1

0QFSBÓE nm
TMFFQ@SFñH% ,]ÒœÛù¥ 4&/4@6-1@$1@4-&&1@$1@$:$ 7i ÏÉ›Ê4b

í•
¾·7VÊ4 6-1x)Ø 9H ¥ÒœÛùŸ÷ ' 4&/4@6-1@$1@4-&&1@$1@$:$ 7i Ï
¥ ñb ªff/ 6-1x)Ø |•¨ 4&/4@6-1@$1@4-&&1@ï¥:$ÒœÛù(b

8"*5&¶©€ÏÄñÛù

E		$ZDMFT

m ·7Ë˜ 8"*5

0QFSBÓE nm
$ZDMFTQ¤ä-W¥©€Ûù”

í•
¾·7V[!çx)Ø Q¤ä-W¥©€Ûùb

"%$&¶ "%$É›©

E		4FM	4BS.VY	3ETU

m ·7Ë˜ "%$

0QFSBÓE nm
3ETU "S¹¶7i 3< |›i%© ²T
4FM Ê4 "%$ }VÊ4 4"3"%$ }VÊ4 4"3"%$ 8VnV b
4BS.VYP? 4"3"%$– <4BS@.VY›V > b

V "%$7¥{Æ•|

5– •	(1*0	4BS@.VY	"%$Ê4 4FM
4&/403@71 (1*0		4FM Ê4 4"3"%$
4&/403@$"11 (1*0		
4&/403@$"1/ (1*0		
4&/403@7/ (1*0		
,@91 (1*0		
,@9/ (1*0		
7%&5@ (1*0		
7%&5@ (1*0		
)BMM QIBTF		
)BMM QIBTF		

5– •| (1*0	4BS@.VY	"%$Ê4 4FM
(1*0		4FM Ê4 4"3 "%$
(1*0		
(1*0		
.5%0 (1*0		
.5$, (1*0		
.5%* (1*0		
.5.4 (1*0		
(1*0		
(1*0		
(1*0		

í•
¾·7V "%$¥•|É›© b "%$·7© ¥5–Ð•|nV b

* $@3% * $&@83 *¤$

E	38		* $4F	M)JHI	-PX	%BUB	4VC BEES

m ·7Ë˜ *¤$

0QFSBÓE nm
4VC BEÈS¥7i ¹¶
%BUB * $@83Ø Ï³•Æ¥" Œ,ö¨¿ * $@3Ø
-PX Ê '¥ÚÊ
)JHI Ê '¥®Ê
* $4FM ñi% *¨$V ¹¶¥7i 4&/4@* $@4-"7&@@"%%ïÉ›Ê4
3 8 *¨$Y•Ë˜
*¨$
*¨$

í•
¾·7V[¦ûÐ"† *¨$V É›Y• • µ1 35$*¨"$¶!!¥P¨ Vn b

ÿi
ö T/ 35$@*V$ 4$-HÒ¥/† 4%"{Æ•|É›•"b

3&(@3&%V"¶7i |

E	)JHI	-PX		"EES

m ·7Ë˜ 3&(@3%

0QFSBÓE nm
"EES "¶!!7i ¹¶ †Ê¹ Ê3
)JHI 7i ²•Ê
-PX 7i 7SÊ

í•

¾·7V[V“¶V 7i Ï |KÚ Ê¥=¸ iiÆY¨7i 3 b

3 3&(<"EES><)JHI -PX>

Â³ |¥=¸ÑV Ê ')JHI-PX 5¾·7|Rí <-PX -P¥X¸>b

ÿi

t ¾·7V`Ù 35$@$/5 35$@*4&/4# 35$@*“¶$!!Ï¥7i b 6-1x)Ø VYVM]
7i %1039L ¥¹¶ 9Ø“¶7i ¥¹¶ 8ZTn/

BEES@VMQ BEES@EQPSU %3@3&(@35$$/5-@#"4&

t BEES@VMÊQ 7d3« ¹†Ê Vg À %3@3&(@35$$/5-@#"4&¥˜ b
yN Ê 6-1x)Ø ¥¹¶V-Â“¶7i bW¥ 3« “ %3@3&(@35$$/5-@#"4&
%3@3&(@35$*@%3"@4&&(@4&/4@#%"4@3&(@35$@*u$×@#"4&

3&(@83¶•Æ“¶7i

E	)JHI	-PX	%BUB	"EES

m ·7Ë˜ 3&(@83

0QFSBOE nm
"EES "S7i ¹¶ †Ê¹ Ê3
)JHI 7i ²•Ê
-PX 7i 7SÊ
%BUB ³•Æ¥´ Ê”

í•

¾·7V[_“!7i •ÆBñ Êë'´ %BUB

3&(<"EES><)JHI -PX> %BUB

Â³•Æ¥=¸ÑV Ê ')JHI-PX 5¾·7öó Ê[¥=¸A b

ÿi

µ1 BEES@=V¸MhQn b

6-1x)Ø ñ½¥³›

6-1x)Ø ÜV Ê!9 Vÿë¿ö $16› í,ª€^Ž)¿ %FFQ TMFQ

Å˜ÆßÏ ¹ †®ÿ{ "dVPö $16ÉÆ%FFQ TMFQ 6-1x)Ø É›A1 Tb
¹ ÉB„†®ÿ{ 6-1x)Ø 1&9V[ÉÆ¤ä Tb ƒÕƒƒ/ ®¿íE ›©…Èqñ½
yN"d¥! ñ+y¥—q9H V¨¿Òœ 6-1x)Ø b¾—q9H A¶4-É›¥Â i
ñi% ,]ÒœÛù¥ 4&/4@6-1@$1@4-&@13@7i:$É›Ê4bö $16„ 6-1x)Ø ñ

½(VYV 3&(@8„3 4-&&17É›M1!Âb¤" "dVYV!Â 35$@$/5-@45"5&7¡@3&(
Ï¥ 35$@$/5-@6-1@$1@4-1@5ï.&3@1&)Ø 9H b

6-1x)Ø VYV³›)"-5·7ÉÆ¤ä ÿ ƒ9|]H(? 6-1x)Ø ¥—q9H
35$@4-08@7$Sd9HÒ" ª ÿ/ Ÿ1=†¥ L)[3$'• b9H BŠžù 6-1x
)Ø | ÈiYVi% 4&/4@1$@ï¥/*15$ îb ••|Ð7i -W¥1"Vnm b

m 6-1x)Ø ñ½•m

¯Ê„ È§W 6-1x)Ø ñ½Çö 6-19H ¥ ªÛù 4&/4@6-1@$1@4-&&1@V$ù$ @3&(
ª îb

6-1x)Ø ñ½¥ ›¨½UèVnm Ï 8„ç "

ÈqYV 35$@$/5-@6-1@$1@4-1@5*.&31@8/b

6-19H Vù 6-1x)Ø 7SV 1$ 4&/4@1$@*›/ñ½b

³›)"-5·7 6-1ñ½T¸› 6-19H Q îb

³› 4-&&17 ©¿¤ä9H Ûù7i b

6-1x)Ø ñ½ ÈqYV 35$@$/5-@6-1@$1@4-1@51*>.&31@8/b

m 6-1x)Ø ñ½@em

6-1x)Ø Òœa³›ñ½„¤ä¥ 8H½® 6-1 '4.eÅ 8¹

6-19H VùH '4. Òœ6-1x)Ø ƒñVñûÃ ñHÒÛùb

6-1x)Ø ñ½³›-- '4. ©€ 35$@$/5-@5*.&37@3&(
35$@$/5-@6-1$1@506$)@4!Â¥5Ä@8"ÒÛù ƒñVñ^¹ ©€ .)[HÒ
ÿ×çb

6-1x)Ø ñ½³›b

Ø¨)"-5·7ª 6-1x)Ø ñ½³›T¸b '4. Î³1 ñHÒÛùŸP 6-1x)Ø ÉÆ¤ä
ÿb

35$@*e$Å

6-1x)Ø VP¨ {Ê¿ 35$×¥ÿë *¨$e Å Ð"† *¨$V É›Y•bÐ * $ * $¶!!
M1 35$@*¥$?"M µKb

¥Â 35$@* $

6-1x)Ø žÈP¨ *¨$7--A¶¥Â 35$@*Ï¥$+ç•" VYV ö $16 6-1x)Ø '
&›ñ½›î 8'_ 35$@*7$•Æ+ç9H•"b

YV 35$@* $@4$-@-08@1&„3*0%@3&(@4$-@)*()@1&!Â*03%@&"(45@$-,
ÛùÏ 4$-HÒ¥Ú®ÈÜz „Ûù è Ôq¹ L)[H !Â 35$@* $@4$-@-08@1&3*0%
a 35$@* $@4$-@)*()@1&3*0%

YV 35$@'"45@ï$¥-,35$@* $@4%"@%5Â:@%3"MÐ-©€¥Ûù" è
35$@* $@4%"@%65:

YV 35$@* $@4$-@45"35@1!Â3*0|%¥©€&H(W è
35$@* $@4$-@45"35@1b&3*0%

YV 35$@* $@4$-@4501@1&Â3*0•%-@©&H W è
35$@* $@4$-@4501@1b&3*0%

YV 35$@* $@5*.&06!Â@3Ñ&H•" è 35$@* $@5*.&06b5

YV 35$@* $@$53Ï¥@ 3&5($@* $@.4@Ê0%ö&Tb

|"†V ¥¹¶•Æ 4&/4@* $@4-"7&@@"%%K3 VYVƒÕZTçIñ ñV ¹¶b
Nª V¹ÄQY•Ê4 ñ •¹¶]Fîx)Ø * $·7b

›î • 35$@*¥ÅÊª 'VP¨ * $@3% * $@83* ¨$7b

P¨ 35$@* $

6-1x)Ø ¥ ñ·7" •¨] ñ 0Q$PE(FVP¨ 35$@*eÅÊ * $@3%„ * $@83
• Wn * $@3% * $@83* ¨$c«b

* $@3%|†ñ3«

* $@3%¥³›„çÂ/ nm

ö ?Â îº|b

ö ?Â 73« "V ¹¶„ •eÅÊ NH •eÅÊÂ¹ }Vo•pbV ¹¶V
V 4&/4@* $@4-"7&@@"|%%Ï73 ¥ 8Ê4VYV * $@3%Âb

V ?Â‹s•|b

ö ?ÂV 7i ¹¶ Ï7i ¥ 8Ê4VYV * $@3%Âb

V ?Â‹s•|b

ö ?Â×¯ îº|b

ö ?ÂV ¹¶ Ï •eÅÊÂ¹ }Vo pb

V ?Â ñ3«¥" b

ö ?Âd‹s•|b

ö ?ÂT¸•| ²• |b

ö ?Â 73« "V ¹¶„ •eÅÊ NH •eÅÊÂ¹ }Vo•pbV ¹¶V
V 4&/4@* $@4-"7&@ '|%Ï73 ¥ 8Ê4VYV * $@8!3Âb

V ?Â‹s•|b

ö ?ÂV 7i ¹¶ Ï7i ¥ 8Ê4VYV * $@8!3Âb

V ?Â‹s•|b

ö ?Â×¯ î•|b

ö ?ÂV ¹¶ Ï •ÊÂ¹ }Vo•pb

ö ?Â ñ3«¥" b

V ?Â‹s•|b

ö ?ÂT¸•| ²••Æb

.BTUFS	4MBWF "EESFTT	4MBWF "EESFTT	4MBWF "EESFTT	%BUB	
4MBWF					

m *¤•$T

_©p Hq

6-1x)Ø ·7 * $@3,%* $@8,3YV7i áV /"$, ©p HqbMQ ‹¨ñ½V[YV
°Ø 35$@* $@*/5@7i ¥3¤ç‹ ‘ ·7^Žîÿ³›b¹ _°+ç¥Y• î
35$@* $@*/5@7i ¥3M‹(Ê‹É›!Âbÿi "dÊm|M bÂT_©ž+çY• î O
!Â 35$@* $@*/5@7i 5V&(V 35$@* $@*/5@$7B0,b&(

õ¤ *¤$|

4%", 4$-HÒ•|VYV¥Â 35$*0@4"3@* $@7i0@8À&(ñ (1*05– ñ W%
çlhnc« *0@.6,9(1*0ŒÐ " Ï¥V 35$@.659–b† b

7i V

4&/4@6¹¶bW

ë	í•	¹¶	`ÙË˜
6-19H ÛùÊ4			
4&/4@6-1@$1@4-&&1@$	$9H@3Ù&(!Â	Y ''	•
4&/4@6-1@$1@4-&&1@$	$9H@3Ù&(!Â	Y '' $	•
4&/4@6-1@$1@4-&&1@$	$9H@3Ù&(!Â	Y ''	•
4&/4@6-1@$1@4-&&1@$	$9H@3Ù&(!Â	Y ''	•
4&/4@6-1@$1@4-&&1@$	$9H@3Ù&(!Â	Y ''	•
35$*¤$!!¹¶Ê4			
4&/4@4"3@4-"7&@"%%3	@*3&¶ „	Y '' $	•
4&/4@4"3@4-"7&@"%%3	@*3&¶ „	Y ''	•

4&/4@4"3@4-"7&@"%%3	@*3&¶ „	Y ''	•
4&/4@4"3@4-"7&@"%%3	@*3&¶ „ *¨$e Å	Y ''	•
35$ *¤e$Å			
4&/4@4"3@* $@$53-@3&(	*¨$e Å7i	Y ''	•

35$@*¹$bW

ë	í•	¹¶	`ÙË˜
35$ *¤e$Å7i			
35$@* $@$53-@3&(	.{!Â	Y '' $	•
35$@* $@%&#6(@45"564@	Ø3k&ÿ	Y '' $	•
35$@* $@5*.&065@3&(	ÑH!Â	Y '' $ $	•
35$@* $@4-"7&@"%%3@3&¹	(V!!¹¶!Â	Y '' $	•
35$ *¤•$!Â7i			
35$@* $@4%"@%65:@3&(	¥Â 4$-/† ª¥ 4%"ûHW	Y '' $	•
35$@* $@4$-@-08@1&3*0	¥Â@43$&-(HÒ¥®ÈÜz	Y '' $	•
35$@* $@4$-@)*()@1&3*0	¥Â@43$&(HÒ¥ÚÈÜz	Y '' $	•
35$@* $@4$-@45"35@1&3*0%@3&(	¥Â7SHq/ 4%"Ð 4$-/† -W¥üþ	Y '' $	•
35$@* $@4$-@4501@1&3*0%@3&(	¥ÂT Hq/ 4%"Ð 4$-/† -W¥üþ	Y '' $	•
35$ *¤Ï$7i Ç¨¿Ø k"¥			
35$@* $@*/5@$-3@3&(	b" *¨$Y• î ÿ	Y '' $	•
35$@* $@*/5@&/@3&(	7S} *¨$Y• ÿ î	Y '' $	•
35$@* $@*/5@45@3&(	} *¨$Y• î¥ ÿ	Y '' $ $	º

ÿi

Ÿ1 35$@*e$Å ¥Ï H Ãõ¤ [Ï 7i Ç¨¿ Øk"¥b

7i

4&/4@6¹¶bW

3FHJTUFS 4&/4@6-1@$1@4-&&(&1@$:$Y Y

jR	y
	_2b2i

4&/4@6-1@$1@4-&&(&1@$19H Ûù!Â 6-1x)Ø VYV·7 4-&&1

•7i ÏÉ›Ê4b •

3FHJTUFS 4&/4@4"3@45"35@'03$&@3&(D

SFTFSWFE	4&/4@1$@*/*5	SFTFSWFE 4&/4@6-1@$1@45"35@501 4&/4@6-1@$1@'03$&@45"35@501	SFTFSWFE
jR kk	kR RR Ry	N 3 d y	
			_2b2i

4&/4@1$@*6/*15 1$Æg¹¶b •

4&/4@6-1@$1@45"35@501x)Ø Ç' 4&/4@6-1@$1@'03$&@45"35@501 Hµrb •

4&/4@6-1@$1@'03$&@45"35@501x)Ø ® 4&/4@6-1@$1@45"35@501 6-1x)Ø ®—q9H îb •

3FHJTUFS 4&/4@4"3@4%%3"7&@"%%3 @3&(Y D

SFTFSWFE	4&/4@* $@4-"7&@"%%3	4&/4@* $@4-"7&@"%%3
jR kk	kR RR	Ry y
	Y	Y
		_2b2i

4&/4@* $@4-"7&@*%%3¶ b •

4&/4@* $@4-"7&@*%%3¶ b •

3FHJTUFS 4&/4@4"3@4%%3"7&@"%%3 @3&(Y

SFTFSWFE	4&/4@* $@4-"7&@"%%3	4&/4@* $@4-"7&@"%%3
jR kk	kR RR	Ry y
	Y	Y
		_2b2i

4&/4@* $@4-"7&@*%%3¶ b •

4&/4@* $@4-"7&@*%%3¶ b •

3FHJTUFS 4&/4@4"3@4%%"7&@"%%3 @3&(Y

SFTFSWFE | 4&/4@* $@4-"7&@"%%3 | 4&/4@* $@4-"7&@"%%3

jR kk	kR RR	Ry y
	Y	Y

_2b2i

4&/4@* $@4-"7&@"%%3¶ b •

4&/4@* $@4-"7&@"%%3¶ b •

3FHJTUFS 4&/4@4"3@4%%"7&@"%%3 @3&(Y

SFTFSWFE | 4&/4@* $@%0/& | 4&/4@* $@3%"5" | 4&/4@* $@4-"7&@"%%3 | 4&/4@* $@4-"7&@"%%3

jR	jy	kN kk	kR RR	Ry y
		Y	Y	Y

_2b2i

4&/4@* $@%0/&¨$X›îbº

4&/4@* $@3%"5"|"bº

4&/4@* $@4-"7&@"%%3¶ b •

4&/4@* $@4-"7&@"%%3¶ b •

3FHJTUFS 4&/4@4"3@* $@$53-@3&(Y

SFTFSWFE | 4&/4@4"3@* $@45"35@ | 4&/4@4"3@* $@45"35 | 4&/4@4"3@* $@$53-

jR	jy	kN	k3	kd y

_2b2i

4&/4@4"3@* $@45"35@'0"3$&q î *¨$®'4. îb •

4&/4@4"3@* $@45"35$Ç' 4&/4@4"3@* $@45"35@H0"3$& •

4&/4@4"3@* $@*$¨$&3Å" Ç' 4&/4@4"3@* $@45"35@H0"3$& •

35$@*¹$bW

3FHJTUFS 35$@* $@4$-@-08@1&3*0%-@3&(Y

SFTFSWFE

35$@* $@4$-@-08@1&3*0%

jR	RNR3	y
		_2b2i

35$@* $@4$-@-08@1&3*0%|)¿®ÈÜH¥Ûù”b •

3FHJTUFS 35$@* $@$53-@3&(Y

SFTFSWFE

35$@* $@39@-4#@'*345
35$@* $@59@-4#@'*345
35$@* $@53"/4@45"35
35$@* $@.4@.0%&
SFTFSWFE
35$@* $@4$-@'03$&@065
35$@* $@4%"@'03$&@0

jR 3	d	e	8	9	j k	R	y
							_2b2i

35$@* $@39@-4#@5'*345 4#b •

35$@* $@59@-4#@5'*345 4#b •

35$@* $@53"/4@45"35 7SHqb •

35$@* $@.4@.0%&T }Vö T }VV Tb •

35$@* $@4$-@'03$&@065 }Vw•{ }V7P{ b •

35$@* $@4%"@'03$&@065 }Vw•{ }V7P{ b •

3FHJTUFS 35$@* $@%&#6(@45"564@3&(Y

SFTFSWFE 35$@* $@4$-@45"5& 35$@* $@."*/@45"5& SFTFSWFE 35$@* $@#:5&@53"/4 35$@* $@4-"7&@"%%3@."5$) 35$@* $@#64@#64: 35$@* $@"3#@-045 35$@* $@5*.&%@065 35$@* $@4-"7&@38 35$@* $@"$,@7"-

jR	jy k3	kd k8	k9 d	e	8	9	j	k	R	y

_2b2i

35$@* $@4$-@45"5ÿ& ¥ ÿb •

35$@* $@."*/@45"5ÿ& ¥ ÿb •

35$@* $@#:5&@53"/4Ê.{X›îb •

35$@* $@4-"7&@"%%3@.T5$)¹¶^ŽM˜b •

35$@* $@#64@#64:ž É›Ïb •

35$@* $@"3#@-045 >€ *¨$9LeÅb •

35$@* $@5*.&%@065 ÑHb •

35$@* $@4-"7&@38/ ¤lž¥ •Ê¥´b •

35$@* $@"$,@7"‹s•|¥´b •

3FHJTUFS 35$@* $@5*.&065@3&(Y D

SFTFSWFE 35$@* $@5*.&065@3&

jR ky	RN y

_2b2i

35$@* $@5*.&065¤l¥ '"45@$-,vÛù"b •

3FHJTUFS 35$@* $@4-"7&@"%%3@3&(Y

35$@* $@4-"7&@"%%3@	SFTFSWFE	35$@* $@4-"7&@"%%3
jR jy	R8 R9	y
		_2b2i

35$@* $@4-"7&@"%%'3@Y ¹ÑÑ*5 ÊH!Âb •

35$@* $@4-"7&@"%%¶3b •

3FHJTUFS 35$@* $@*/5@$-3@3&(Y

SFTFSWFE	35$@* $@5*.&@065@*/5@$-3	35$@* $@53"/4@$0.1-&5&@*/5@$-3	35$@* $@53"/4@$0.1-&5&@*/5@$-3	35$@* $@."45&3@53"/4@$0.1-&5&@*/5@$-3	35$@* $@"3#*53"5*0/@-045@*/5@$-3	35$@* $@4-"7&@53"/4@$0.1-&5&@*/5	SFTFSWFE
jR N	3	d	e	8	9	j y	
							_2b2i

35$@* $@5*.&@065@b*"/Ñ@Ïb3 •

35$@* $@53"/4@$0.1-&5&@b*"/5¸@TSS@Ïb •

35$@* $@."45&3@53"/4@$0.1-&5&ö@*T//5@"$T3îÏb •

35$@* $@"3#*53"5*0/@-045@*/5@$>3E9LeÅÏb •

35$@* $@4-"7&@53"/4@$0.1-&5&V@*T//5@"$T3îÏb •

3FHJTUFS 35$@* $@4%"@%65:@3&(Y

SFTFSWFE

35$@* $@4%"@%

jR ky	RN y

_2b2i

35$@* $@4%"@%65:Đ 4%'MĐ-W¥ '"45@$-Û¡ù"b •

3FHJTUFS 35$@* $@4$-@)*()@1&3*0%@3&(Y

SFTFSWFE

35$@* $@4$-@)*()@1&3*0%

jR ky	RN y

_2b2i

35$@* $@4$-@)*()@1&3*0% '"45@$-Û¡ù"b •

3FHJTUFS 35$@* $@4$-@45"35@1&3*0%@3&(Y

SFTFSWFE

35$@* $@4$-@45"35@1&3*0%

jR ky	RN y

_2b2i

35$@* $@4$-@45"35@1&3*0% ³1©€¥ '"45@$-Û¡ù"b •

3FHJTUFS 35$@* $@4$-@4501@1&3*0% @3&(Y

SFTFSWFE 35$@* $@4$-@4501@1&3*0%

jR ky	RN y

_2b2i

35$@* $@4$-@4501@Á13&T3ÿHOq% ³1©€¥ '"45@$-Û¡ù”b •

®ÿ{5Ø 35$@$/5-

À•

&41 •¨ Úra2 ¥ÿ{5Ø/Œ V[ÿ{eÅaÒœüþ„,]Òœ÷-WLCKDÜ™b‰Ë
¥ö)Ø ¦û Õÿ{ T V[¡@¨¼¥,]Æß³pbN“ &41 Î¦ûÑ®ÿ{x)Ø 6-1
$P QSPDÆÅTP¸ß)Ø ÉÆ %FFQ TMFVFQKvñ †®ÿ{ ¦û¨¼¥®ÿ{‹
¨b

ö1+Ÿ

t ¦û Õç!ÿ{ T a¨¿ Õ‹¨Æß

t ¦ûÚr ,# ==i 3FUFOUJPO .FNPSZ

t Y CJ=U7i 3FUFOUJPO 3FHJTUFS

t îµ®ÿ{ T(¦û 6-1x)Ø eÅ

t ¦û 35$CPÿP?UVê Òœüþ

m &41ÿ{eÅUim

ÿ?í•

eº

&41 ¥®ÿ{5Ø†í "Øâ aÿ{eÅ aÈ÷71†íaÈ÷×ïÖ†í *TPMBUCFTO $FMM
8² UimVn Zm b

"3=,Øâ

&41 ¥=Â"3=,Øâ V[|"†È÷Èâ YÈ¹ 7 Ð¹ 7¦û"3=,¥žÈýTb
¾Øâ V[¤s¥"†{ÆÈâS¶¹ 7ž 7{ ÈâS¶¹ 7ž 7b 8² UimV
n/Zm b

' 91%@%*(@3&H(¾Øâ ¥{ Èâ¹ 7"3=,V[žÈýT ' 91%@%*(@3&(
H Øâ „"3=,(íEýTb

%*(@3&(@%#*V'[4Ø«"3>=,¥ ÈÈâ

7%%@=%*(+ %#*"4: 7

@Æ"3=,¥È@Ÿ15– 7%% 1 @$17%% 1 @$35$

m "3=,Øâ

®ÿ{Øâ

&41 ¥=Â®ÿ{Øâ V[|"†È÷Èâ YÈ¹ 7 Ð¹ 7¦û=† 35$"3=,¥žÈ
ýTb¹ †®ÿ{ ¾Øâ V[¤s¥"†{ÆÈâS¶¹ 7ž 7{ ÈâS¶VØ žÈýT
T/¥Èâ{ S¶¹ 7ž 7 %FFQ T„M¤EFTQ/¥Èâ{ %ç¹ 7¨ä T/¥
Èâ{ ÷®b 8² UimVn/Zm b

'5– $)*1@1ÓÈÜH ®ÿ{Øâ íE1> Ç? žÈ„ %FFQ TMFWÓ›MÐb

žÈ T/ 35$@%#*"4V<[Ø«{ Èâ

7%%@35$ + %#*"4: 7

%FFQ TMFFØQâ ¥{ È â%ç¹ 7b

@_35$”3=,¥È@Ÿ15– 7%% 1 @b35$

m ®ÿ{Ø â

'MBTØIâ

&41 ¥=Â GMBØTâIV[_"dÏ¥ ð!!{ 7 7È â 1Â GMB3TØ â ¥KvÈ@ { ¹ N"b 8² UimVn/Zm b

’ 91%@4%*0@73H&(Ø â ¥{ È â¹ 7 7’ 91%@4%*0@73H&(Ø â ¥{ È â¹ÚEF È â®“†È÷ ‹b

’ 4%*0@5*&)H Ø â |5– 7%%@4„%5*0 7%% 1 @3”5$È â{ ¹ 7'5– 7%% 1 @3È5$ ’ 4%*0@5*&)H Ø â ¥È â{ ¹•IÈ â 73&'YÈ¹ 7b

÷¿ %3&')@4%*0 3&'.@4%*0 3&'-@4%*0 3&'Œƒ Õ TV?ö• Y"d=Ì¥×çŸ yN,wiÉ›÷¿b

’Ø â { ¹ 7 7H { È@Ÿ15– 7%% 1 @b35$

m 'MBØâ

4â_©

&41 ¥4â_© V[_°5– 7%% 1 @¥È5ab'È âyÎ/‹ÀBç£Ü 4â_© |? (
?•| 1>†s{È v 1Â -/" „ 1"© V7¹"3 vš|÷ HW ¨[ia M×1" b
4â_© ¥ÿ{dÈ® | ‰Ë7 H¥ù û7 8¥•|(?•´V[Ø« YÈ¹ 7b 8
² UimVn/Zm b

35$@$/5-@#308/@064@%&¥{ ÈÜ | 5– 7%% 1 @È5a®¿•´HKÀ
ÚÈÜb

35$@$/5-@%#308/@065@5)¿Ø«4â_© ¥•|(?•´ YÈ¹ 7b

m 4â_©

35$ v

&41 ¥ 35$ vÜV Ê!9 V¨¿5Ø®ÿ{ T¥ÉÆ„| eÅHÒ÷a 1--aÈ÷71„ïÖ
†í[Á3È÷ÊeaHÒÊe„¯Ê•|b 35$ v¥² mVnm ö1†s "

- t 35$ö ÿ &¶&¶:cÈ÷ ÿb
- t "3 EÈ÷eÅ &¶&¶V¨¿¹ 35$¥"3 v„ E v3îÈ÷Êe HÒÊe•|b
- t ¤ä ÒœeÅ &¶&¶V)Ø®ÿ{ T¥ÉÆ„| b
- t 9H &¶&¶ " 35$ö9H a 6-1x)Ø 9H „(9H b

t ®ÿ{)Ø „.Ë eÅ &¶&¶ 6-1x)Ø a(eÅ a 4"3"%$ÅÅ ©b

t ==i

o 35$¤Î=i&¶&¶¦û ,#43". 'v†s¨T ==i i% 6-1x)Ø ¥·7 " =ib
$16VYV "1#9L`Ù¤Î=i S¹¶¹ Y b

o 35$yÎ=i&¶&¶¦û ,#43". 'v†s¨T ==ib $16VYV *3". %3".`ÙyÎ
=ib 35$yÎ=i¥Î ÿ¹ 35$¤Î=i¥ b

t =7i &¶&¶ Y Êb¾7i ¥ù7 V¨¿" i%b

t 35$ *θ–&¶&¶ ño BMXBZpT5-PΟΥÈT¹Òœ÷b

m 35$ m

®ÿ{HÒ

®ÿ{ T/ &41 ¥ .)[Ö'„ 1--YÈ| È[†®ÿ{ ŒHÒ¯|7 [' ‰Ë ®ÿ{ T/¥žÈýTb

35$=,V[P¨ ñHÒ÷

t “†®ÎÖ’HÒ $,@95"-@ , L)[

t “†ÚÎÖ’HÒ $,@ .@%*(a)[.)[

t =† 3$’• 4-08@$ÔqVØ YÈ¹ L)[

t =† .)[’• $, .@065

t =† L)[HÒ $, .@% @0ÿ65 =† .)[’• sÔ

ƒtHÒV[Á3 GBTU@ŞUTM@X@SUT@/DMLGBTU@Ê4UT@DM165TMPX@ŞU4D@DML
4-08@$,8Vnm b

m 35$®ÿ{HÒ

¿”3=, MPX@QPXVFS@ÑDHÒL÷-WMÐ 8Vnm b

m ”3®ÿ{HÒ

È÷Êe¥LC

m 35$ÿ

È÷Êe Äm 8Vnm bL=È÷eÅ•|VYVÈq!Â¹<Å7 '16 <Å1> '1%bÈ ÷×¦ûÿëÈ÷Êe VŒ Õ,]‹¨ÆßÉ› 8F†bV ZU &41 È÷×¥eÅZ Tb

V 35$È÷×

È÷×		35$ö1 ÿ			È qÊ[		ÿi
		%*(	35$	35$¤ä	u7	1>	
35$	”3=,	0/	0/	0/	/	/	
	35$“¶	0/	0/	0''	:	:	
	35$¤Î=i	0/	0''	0''	:	:	
	35$yÎ=i	0/	0''	0''	:	:	
”3	”3=,	0/	0''	0''	:	:	
	8J 'J	0/	0''	0''	:	:	
	30.	0/	0''	0''	:	:	
	=† 43".	0/	0''	0''	:	:	
E	.)[Ö’	0/	0''	0''	:	:	
	1--	0/	0''	0''	:	:	
	.)[’•	0/	0''	0''	:	:	
	Ô				:	:	

ÿ i

È ÷ × 3 5 $" 3 = , ¹ o B M X B Z p ¬ ×P O

o 35$=i„yÎ 35$=iV[ûb

o È@h{ "b

o Òœüþ ®¿ NTb

o wi¨¿BtdÔNõ¤ 8J 'J¡é¥Ñ®ÿ{‹¨ÆßB

t ¨ä T

o =† .)['• a .)[ÚÎÖ'a 1--# Ô v(Ì¨b

o ”3=, È $16=¸é>b

o 35$“!× Èb

o 35$=,¥ ÈÈâ†À 7b

o Y Ê” i Y¨ =7i Ïb

o 35$=i„yÎ 35$=i Èb

o È@h{ "b

o Òœ÷ Ç¦û 35$9H b

o Òœüþ ®¿ NTb

o wi¨¿BtdÔNõ¤ 8J 'J¡é¥Ñ®ÿ{‹¨ÆßB

m ÿ{ T

ªff/ &41 "d¯Êª|ÉÆ "DUJWbF' $16,³1B°ýTH 1Â'©€“† îÒœH "dV[ÉÆ Õ®ÿ{ Tb¨¼Vô 8ÿ{aÒœüþ„V¨Òœ÷³p òÕÿ{ TÏÉ›Ê 4bWnm b

一 ，ESP32 设 可 一步 化 不 场 。

31.3.10 唤醒源

ESP32 可 ，可 CPU 不 中 。 RTC_CNTL_WAKEUP_ENA ，见表 31-2。

表 31-2. 唤醒源

WAKEUP_ENA		Light-sleep	Deep-sleep		*
0x1	EXT0	Y	Y	-	1
0x2	EXT1	Y	Y	Y	2
0x4	GPIO	Y	Y	-	3
0x8	RTC 时	Y	Y	Y	-
0x10	SDIO	Y	-	-	4
0x20	Wi-Fi	Y	-	-	5
0x40	UART0	Y	-	-	6
0x80	UART1	Y	-	-	6
0x100	TOUCH	Y	Y	-	-
0x200	ULP	Y	Y	-	-
0x400	BT	Y	-	-	5

*:

1. EXT0 可 Light-sleep/Deep-sleep 中 。 RTC_CNTL_EXT_WAKEUP0_LV 1 时 ， 。 可 设 RTCIO_EXT_WAKEUP0_SEL[4:0]， RTC 。
2. EXT1 设 ，可 中 ，所 个 。常 ， 按 ， RTC_CNTL_EXT_WAKEUP1_SEL[17:0]。 ， RTC_CNTL_EXT_WAKEUP1_LV 1 时， ， ； RTC_CNTL_EXT_WAKEUP1_LV 0 时， 必 部 才 。
3. Deep-sleep ， RTC GPIO 可 ，所 GPIO。
4. SDIO 。
5. Wi-Fi BT ， Active、Modem-sleep Light-sleep 间 ，CPU、Wi-Fi、Bluetooth 设间 中 ，保 Wi-Fi/ 常 。
6. RxD 变化 (UART_ACTIVE_THRESHOLD+2) 时， 。**注意：**RxD 不 GPIO ， IO_MUX 。

31.3.11 RTC 计时器

RTC 时 一个可 48-bit ，时 RTC_SLOW_CLK。 上 / 不 RTC 时 。

RTC 时 可 时间 CPU，并 期 ULP 。

31.3.12 RTC Boot

CPU ROM RAM Deep-sleep ，所 ROM 包 SPI (flash 中) 时间， 比 Light-sleep Modem-sleep 启长 。RTC

中保 2 SRAM ，别 RTC RTC 。2 SRAM Deep-sleep 启。一 不 （ 8 KB）可 ，避 ROM 包 SPI 启 ，
。

一 ： RTC

1. 设 PRO_CPU RTC_CNTL_PROCPU_STAT_VECTOR_SEL（ 设 APP_CPU RTC_CNTL_APPCPU_STAT_VECTOR_SEL） 0。
2. 。
3. CPU 启 时， 0x50000000，0x40000400 始 ， 个 并不 ROM 包 SPI boot。RTC 中 C 中 部 操 可。

： RTC

1. 设 PRO_CPU RTC_CNTL_PROCPU_STAT_VECTOR_SEL（ 设 APP_CPU RTC_CNTL_APPCPU_STAT_VECTOR_SEL） 1。
2. RTC CRC ，并 保 RTC_CNTL_RTC_STORE6_REG[31:0] 中。
3. RTC RTC_CNTL_RTC_STORE7_REG[31:0]。
4. 。
5. CPU 启 ， ROM 包 部 ， CRC 。 RTC_CNTL_RTC_STORE6_REG[31:0] 一 ， CPU 。

ESP32 启 31-11。

图 31-11. ESP32 启动流程图

31.4 寄存器列表

：

- 按 ， 表中 顺 并不 中 顺 。
- AHB ， 0x60008000； DPORT ， 0x3FF48000。

RTC 选项寄存器			
RTC_CNTL_OPTIONS0_REG	RTC	0x3FF48000	/ 写
RTC 计时器寄存器的控制与配置			
RTC_CNTL_SLP_TIMER0_REG	RTC 时	0x3FF48004	/ 写
RTC_CNTL_SLP_TIMER1_REG	RTC 时 、 报	0x3FF48008	/ 写
RTC_CNTL_TIME_UPDATE_REG	RTC 时	0x3FF4800C	
RTC_CNTL_TIME0_REG	RTC 时 32	0x3FF48010	
RTC_CNTL_TIME1_REG	RTC 时 16	0x3FF48014	
RTC_CNTL_STATE0_REG	RTC 、SDIO ULP	0x3FF48018	/ 写
RTC_CNTL_TIMER1_REG	CPU	0x3FF4801C	/ 写
RTC_CNTL_TIMER2_REG	时	0x3FF48020	/ 写
RTC_CNTL_TIMER5_REG	时 期	0x3FF4802C	/ 写
复位状态与唤醒控制寄存器			
RTC_CNTL_RESET_STATE_REG	CPU	0x3FF48034	
RTC_CNTL_WAKEUP_STATE_REG	、	0x3FF48038	
RTC_CNTL_EXT_WAKEUP_CONF_REG	/	0x3FF48060	/ 写
RTC_CNTL_EXT_WAKEUP1_REG	部	0x3FF480CC	/写
RTC_CNTL_EXT_WAKEUP1_STATUS_REG	部	0x3FF480D0	
RTC 中断控制与状态寄存器			
RTC_CNTL_INT_ENA_REG	中	0x3FF4803C	/ 写
RTC_CNTL_INT_RAW_REG	始中	0x3FF48040	
RTC_CNTL_INT_ST_REG	屏蔽中	0x3FF48044	
RTC_CNTL_INT_CLR_REG	中	0x3FF48048	写
RTC 通用保留寄存器			
RTC_CNTL_STORE0_REG	保 0	0x3FF4804C	/ 写
RTC_CNTL_STORE1_REG	保 1	0x3FF48050	/ 写
RTC_CNTL_STORE2_REG	保 2	0x3FF48054	/ 写
RTC_CNTL_STORE3_REG	保 3	0x3FF48058	/ 写
RTC_CNTL_STORE4_REG	保 4	0x3FF480B0	/ 写
RTC_CNTL_STORE5_REG	保 5	0x3FF480B4	/ 写
RTC_CNTL_STORE6_REG	保 6	0x3FF480B8	/ 写
RTC_CNTL_STORE7_REG	保 7	0x3FF480BC	/ 写
内部功耗管理寄存器			
RTC_CNTL_ANA_CONF_REG	上 /	0x3FF48030	/ 写
RTC_CNTL_VREG_REG	部	0x3FF4807C	/ 写
RTC_CNTL_PWC_REG	RTC	0x3FF48080	/ 写

RTC_CNTL_DIG_PWC_REG		0x3FF48084	/ 写
RTC_CNTL_DIG_ISO_REG		0x3FF48088	
RTC 看门狗配置域控制寄存器			
RTC_CNTL_WDTCONFIG0_REG	WDT 0	0x3FF4808C	/ 写
RTC_CNTL_WDTCONFIG1_REG	WDT 1	0x3FF48090	/ 写
RTC_CNTL_WDTCONFIG2_REG	WDT 2	0x3FF48094	/ 写
RTC_CNTL_WDTCONFIG3_REG	WDT 3	0x3FF48098	/ 写
RTC_CNTL_WDTCONFIG4_REG	WDT 4	0x3FF4809C	/ 写
RTC_CNTL_WDTFEED_REG		0x3FF480A0	写
RTC_CNTL_WDTWPROTECT_REG	写保	0x3FF480A4	/ 写
其他 RTC 配置寄存器			
RTC_CNTL_EXT_XTL_CONF_REG	部 XTAL	0x3FF4805C	/ 写
RTC_CNTL_SLP_REJECT_CONF_REG		0x3FF48064	/ 写
RTC_CNTL_CPU_PERIOD_CONF_REG	CPU 期	0x3FF48068	/ 写
RTC_CNTL_CLK_CONF_REG	RTC 时	0x3FF48070	/ 写
RTC_CNTL_SDIO_CONF_REG	SDIO	0x3FF48074	/ 写
RTC_CNTL_SW_CPU_STALL_REG	CPU	0x3FF480AC	/ 写
RTC_CNTL_HOLD_FORCE_REG	RTC 保	0x3FF480C8	/ 写
RTC_CNTL_BROWN_OUT_REG		0x3FF480D4	/ 写

31.5 寄存器

Register 31.1. RTC_CNTL_OPTIONS0_REG (0x3FF48000)

31	30	29	28 23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3 2	1 0	
RTC_CNTL_SW_SYS_RST	RTC_CNTL_DG_WRAP_FORCE_NORST	RTC_CNTL_DG_WRAP_FORCE_RST	(reserved)	RTC_CNTL_BIAS_CORE_FORCE_PU	RTC_CNTL_BIAS_CORE_FORCE_PD	RTC_CNTL_BIAS_CORE_FOLW_8M	RTC_CNTL_BIAS_I2C_FORCE_PU	RTC_CNTL_BIAS_I2C_FORCE_PD	RTC_CNTL_BIAS_I2C_FOLW_8M	RTC_CNTL_BIAS_FORCE_NOSLEEP	RTC_CNTL_BIAS_FORCE_SLEEP	RTC_CNTL_BIAS_SLEEP_FOLW_8M	RTC_CNTL_XTL_FORCE_PU	RTC_CNTL_XTL_FORCE_PD	RTC_CNTL_BBPLL_FORCE_PU	RTC_CNTL_BBPLL_FORCE_PD	RTC_CNTL_BBPLL_I2C_FORCE_PU	RTC_CNTL_BBPLL_I2C_FORCE_PD	RTC_CNTL_BB_I2C_FORCE_PU	RTC_CNTL_BB_I2C_FORCE_PD	RTC_CNTL_SW_PROCPU_RST	RTC_CNTL_SW_APPCPU_RST	RTC_CNTL_SW_STALL_PROCPU_C0	RTC_CNTL_SW_STALL_APPCPU_C0	
0	0	0	0 0 0 0 0 0	1	0	0	1	0	0	1	0	0	1	0	0	0	0	0	0	0	0	0	0 0	0 0	Reset

RTC_CNTL_SW_SYS_RST 。(写)

RTC_CNTL_DG_WRAP_FORCE_NORST Deep-sleep , 不 。(/ 写)

RTC_CNTL_DG_WRAP_FORCE_RST Deep-sleep , 。(/ 写)

RTC_CNTL_BIAS_CORE_FORCE_PU BIAS_CORE 。(/ 写)

RTC_CNTL_BIAS_CORE_FORCE_PD BIAS_CORE 闭。(/ 写)

RTC_CNTL_BIAS_CORE_FOLW_8M BIAS_CORE CK8M 变化 (/ 写)

RTC_CNTL_BIAS_I2C_FORCE_PU BIAS_I2C 。(/ 写)

RTC_CNTL_BIAS_I2C_FORCE_PD BIAS_I2C 闭。(/ 写)

RTC_CNTL_BIAS_I2C_FOLW_8M BIAS_I2C CK8M 变化 (/ 写)

RTC_CNTL_BIAS_FORCE_NOSLEEP BIAS_SLEEP 不 。(/ 写)

RTC_CNTL_BIAS_FORCE_SLEEP BIAS_SLEEP 。(/ 写)

RTC_CNTL_BIAS_SLEEP_FOLW_8M BIAS_SLEEP CK8M 变化 (/ 写)

RTC_CNTL_XTL_FORCE_PU Crystal 。(/ 写)

RTC_CNTL_XTL_FORCE_PD Crystal 闭。(/ 写)

RTC_CNTL_BBPLL_FORCE_PU BB_PLL 。(/ 写)

RTC_CNTL_BBPLL_FORCE_PD BB_PLL 闭。(/ 写)

RTC_CNTL_BBPLL_I2C_FORCE_PU BB_PLL_I2C 。(/ 写)

RTC_CNTL_BBPLL_I2C_FORCE_PD BB_PLL _I2C 闭。(/ 写)

RTC_CNTL_BB_I2C_FORCE_PU BB_I2C 。(/ 写)

RTC_CNTL_BB_I2C_FORCE_PD BB_I2C 闭。(/ 写)

RTC_CNTL_SW_PROCPU_RST PRO_CPU 。(写)

寄存器描述下一页继续。

Register 31.1. RTC_CNTL_OPTIONS0_REG (0x3FF48000)

继上一页寄存器描述。

RTC_CNTL_SW_APPCPU_RST APP_CPU 。(写)

RTC_CNTL_SW_STALL_PROCPU_C0 参见 RTC_CNTL_SW_CPU_STALL_REG。(/ 写)

RTC_CNTL_SW_STALL_APPCPU_C0 参见 RTC_CNTL_SW_CPU_STALL_REG。(/ 写)

Register 31.2. RTC_CNTL_SLP_TIMER0_REG (0x3FF48004)

31 … 0	
0x000000000	Reset

RTC_CNTL_SLP_TIMER0_REG RTC 时 32 。(/ 写)

Register 31.3. RTC_CNTL_SLP_TIMER1_REG (0x3FF48008)

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15 … 0	
(reserved)															RTC_CNTL_MAIN_TIMER_ALARM_EN	RTC_CNTL_SLP_VAL_HI	
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0x00000	Reset

RTC_CNTL_MAIN_TIMER_ALARM_EN 时 报 。(/ 写)

RTC_CNTL_SLP_VAL_HI RTC 时 16 。(/ 写)

Register 31.4. RTC_CNTL_TIME_UPDATE_REG (0x3FF4800C)

31	30	29 … 0	
RTC_CNTL_TIME_UPDATE	RTC_CNTL_TIME_VALID	(reserved)	
0	0	0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0	Reset

RTC_CNTL_TIME_UPDATE 1: RTC 时 。(写)

RTC_CNTL_TIME_VALID 表 。( )

Register 31.5. RTC_CNTL_TIME0_REG (0x3FF48010)

31 … 0	
0x000000000	Reset

RTC_CNTL_TIME0_REG RTC 时 32 。( )

Register 31.6. RTC_CNTL_TIME1_REG (0x3FF48014)

(reserved) 31 … 16	RTC_CNTL_TIME_HI 15 … 0	
0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0	0x00000	Reset

RTC_CNTL_TIME_HI RTC 时 16 。( )

Register 31.7. RTC_CNTL_STATE0_REG (0x3FF48018)

RTC_CNTL_SLEEP_EN 31	RTC_CNTL_SLP_REJECT 30	RTC_CNTL_SLP_WAKEUP 29	RTC_CNTL_SDIO_ACTIVE_IND 28	(reserved) 27 … 25	RTC_CNTL_ULP_CP_SLP_TIMER_EN 24	RTC_CNTL_TOUCH_SLP_TIMER_EN 23	(reserved) 22 … 0	
0	0	0	0	0 0 0	0	0	0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0	Reset

RTC_CNTL_SLEEP_EN 。(/ 写)

RTC_CNTL_SLP_REJECT 。(/ 写)

RTC_CNTL_SLP_WAKEUP 。(/ 写)

RTC_CNTL_SDIO_ACTIVE_IND 表 SDIO 。( )

RTC_CNTL_ULP_CP_SLP_TIMER_EN ULP 时 。(/ 写)

RTC_CNTL_TOUCH_SLP_TIMER_EN 。(/ 写)

Register 31.8. RTC_CNTL_TIMER1_REG (0x3FF4801C)

(reserved)	RTC_CNTL_CPU_STALL_EN	
31 … 1	0	
0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0	1	Reset

RTC_CNTL_CPU_STALL_EN CPU 。(/)

Register 31.9. RTC_CNTL_TIMER2_REG (0x3FF48020)

RTC_CNTL_MIN_TIME_CK8M_OFF	RTC_CNTL_ULPCP_TOUCH_START_WAIT
31 … 24	
0x001	0x010

Register 31.11. RTC_CNTL_ANA_CONF_REG (0x3FF48030)

31	30	29	28	27	26	25	24	23	22 … 0	
RTC_CNTL_PLL_I2C_PU	RTC_CNTL_CKGEN_I2C_PU	(reserved)	RTC_CNTL_RFRX_PBUS_PU	RTC_CNTL_TXRF_I2C_PU	RTC_CNTL_PVTMON_PU	(reserved)	RTC_CNTL_PLLA_FORCE_PU	RTC_CNTL_PLLA_FORCE_PD	(reserved)	
0	0	0	0	0	0	0	0	1	0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0	Reset

RTC_CNTL_PLL_I2C_PU 1：PLL_I2C ；0： 闭。(/ 写)

RTC_CNTL_CKGEN_I2C_PU 1：CKGEN_I2C ；0： 闭。(/ 写)

RTC_CNTL_RFRX_PBUS_PU 1：RFRX_PBUS ；0： 闭。(/ 写)

RTC_CNTL_TXRF_I2C_PU 1：TXRF_I2C ；0： 闭。(/ 写)

RTC_CNTL_PVTMON_PU 1：PVTMON ；0： 闭。(/ 写)

RTC_CNTL_PLLA_FORCE_PU PLLA 。(/ 写)

RTC_CNTL_PLLA_FORCE_PD PLLA 闭。(/ 写)

Register 31.12. RTC_CNTL_RESET_STATE_REG (0x3FF48034)

31 … 14	13	12	11 … 6	5 … 0	
(reserved)	RTC_CNTL_PROCPU_STAT_VECTOR_SEL	RTC_CNTL_APPCPU_STAT_VECTOR_SEL	RTC_CNTL_RESET_CAUSE_APPCPU	RTC_CNTL_RESET_CAUSE_PROCPU	
0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0	1	1	x x x x x x	x x x x x x	Reset

RTC_CNTL_PROCPU_STAT_VECTOR_SEL PRO_CPU 。(/ 写)

RTC_CNTL_APPCPU_STAT_VECTOR_SEL APP_CPU 。(/ 写)

RTC_CNTL_RESET_CAUSE_APPCPU APP_CPU 。( )

RTC_CNTL_RESET_CAUSE_PROCPU PRO_CPU 。( )

Register 31.13. RTC_CNTL_WAKEUP_STATE_REG (0x3FF48038)

(reserved)	RTC_CNTL_GPIO_WAKEUP_FILTER	RTC_CNTL_WAKEUP_ENA	RTC_CNTL_WAKEUP_CAUSE	
31 23	22	21 11	10 0	
0 0 0 0 0 0 0 0 0	0	0 0 0 0 0 0 0 1 1 0 0	0x000	Reset

RTC_CNTL_GPIO_WAKEUP_FILTER GPIO 。(/ 写)

RTC_CNTL_WAKEUP_ENA 。(/ 写)

RTC_CNTL_WAKEUP_CAUSE 。( )

Register 31.14. RTC_CNTL_INT_ENA_REG (0x3FF4803C)

(reserved)	RTC_CNTL_MAIN_TIMER_INT_ENA	RTC_CNTL_BROWN_OUT_INT_ENA	RTC_CNTL_TOUCH_INT_ENA	RTC_CNTL_ULP_CP_INT_ENA	RTC_CNTL_TIME_VALID_INT_ENA	RTC_CNTL_WDT_INT_ENA	RTC_CNTL_SDIO_IDLE_INT_ENA	RTC_CNTL_SLP_REJECT_INT_ENA	RTC_CNTL_SLP_WAKEUP_INT_ENA	
31 9	8	7	6	5	4	3	2	1	0	
0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0	0	0	0	0	0	0	0	0	0	Reset

RTC_CNTL_MAIN_TIMER_INT_ENA RTC_CNTL_MAIN_TIMER_INT 中 中 。(/ 写)

RTC_CNTL_BROWN_OUT_INT_ENA RTC_CNTL_BROWN_OUT_INT 中 中 。(/ 写)

RTC_CNTL_TOUCH_INT_ENA RTC_CNTL_TOUCH_INT 中 中 。(/ 写)

RTC_CNTL_ULP_CP_INT_ENA RTC_CNTL_ULP_CP_INT 中 中 。(/ 写)

RTC_CNTL_TIME_VALID_INT_ENA RTC_CNTL_TIME_VALID_INT 中 中 。(/ 写)

RTC_CNTL_WDT_INT_ENA RTC_CNTL_WDT_INT 中 中 。(/ 写)

RTC_CNTL_SDIO_IDLE_INT_ENA RTC_CNTL_SDIO_IDLE_INT 中 中 。(/ 写)

RTC_CNTL_SLP_REJECT_INT_ENA RTC_CNTL_SLP_REJECT_INT 中 中 。(/ 写)

RTC_CNTL_SLP_WAKEUP_INT_ENA RTC_CNTL_SLP_WAKEUP_INT 中 中 。(/ 写)

Register 31.15. RTC_CNTL_INT_RAW_REG (0x3FF48040)

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0	
(reserved)																							RTC_CNTL_MAIN_TIMER_INT_RAW	RTC_CNTL_BROWN_OUT_INT_RAW	RTC_CNTL_TOUCH_INT_RAW	RTC_CNTL_ULP_CP_INT_RAW	RTC_CNTL_TIME_VALID_INT_RAW	RTC_CNTL_WDT_INT_RAW	RTC_CNTL_SDIO_IDLE_INT_RAW	RTC_CNTL_SLP_REJECT_INT_RAW	RTC_CNTL_SLP_WAKEUP_INT_RAW	
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	Reset

RTC_CNTL_MAIN_TIMER_INT_RAW RTC_CNTL_MAIN_TIMER_INT 中 始中 。( )

RTC_CNTL_BROWN_OUT_INT_RAW RTC_CNTL_BROWN_OUT_INT 中 始中 。( )

RTC_CNTL_TOUCH_INT_RAW RTC_CNTL_TOUCH_INT 中 始中 。( )

RTC_CNTL_ULP_CP_INT_RAW RTC_CNTL_ULP_CP_INT 中 始中 。( )

RTC_CNTL_TIME_VALID_INT_RAW RTC_CNTL_TIME_VALID_INT 中 始中 。( )

RTC_CNTL_WDT_INT_RAW RTC_CNTL_WDT_INT 中 始中 。( )

RTC_CNTL_SDIO_IDLE_INT_RAW RTC_CNTL_SDIO_IDLE_INT 中 始中 。( )

RTC_CNTL_SLP_REJECT_INT_RAW RTC_CNTL_SLP_REJECT_INT 中 始中 。( )

RTC_CNTL_SLP_WAKEUP_INT_RAW RTC_CNTL_SLP_WAKEUP_INT 中 始中 。( )

Register 31.16. RTC_CNTL_INT_ST_REG (0x3FF48044)

31																							9	8	7	6	5	4	3	2	1	0	
																							(reserved)	RTC_CNTL_MAIN_TIMER_INT_ST	RTC_CNTL_BROWN_OUT_INT_ST	RTC_CNTL_TOUCH_INT_ST	RTC_CNTL_SAR_INT_ST	RTC_CNTL_TIME_VALID_INT_ST	RTC_CNTL_WDT_INT_ST	RTC_CNTL_SDIO_IDLE_INT_ST	RTC_CNTL_SLP_REJECT_INT_ST	RTC_CNTL_SLP_WAKEUP_INT_ST	
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	Reset

RTC_CNTL_MAIN_TIMER_INT_ST RTC_CNTL_MAIN_TIMER_INT 中断屏蔽中断。(R/W/SS/SC)

RTC_CNTL_BROWN_OUT_INT_ST RTC_CNTL_BROWN_OUT_INT 中断屏蔽中断。(R/W/SS/SC)

RTC_CNTL_TOUCH_INT_ST RTC_CNTL_TOUCH_INT 中断屏蔽中断。(R/W/SS/SC)

RTC_CNTL_SAR_INT_ST RTC_CNTL_SAR_INT 中断屏蔽中断。(R/W/SS/SC)

RTC_CNTL_TIME_VALID_INT_ST RTC_CNTL_TIME_VALID_INT 中断屏蔽中断。(R/W/SS/SC)

RTC_CNTL_WDT_INT_ST RTC_CNTL_WDT_INT 中断屏蔽中断。(R/W/SS/SC)

RTC_CNTL_SDIO_IDLE_INT_ST RTC_CNTL_SDIO_IDLE_INT 中断屏蔽中断。(R/W/SS/SC)

RTC_CNTL_SLP_REJECT_INT_ST RTC_CNTL_SLP_REJECT_INT 中断屏蔽中断。(R/W/SS/SC)

RTC_CNTL_SLP_WAKEUP_INT_ST RTC_CNTL_SLP_WAKEUP_INT 中断屏蔽中断。(R/W/SS/SC)

Register 31.17. RTC_CNTL_INT_CLR_REG (0x3FF48048)

31																							9	8	7	6	5	4	3	2	1	0	
																							(reserved)	RTC_CNTL_MAIN_TIMER_INT_CLR	RTC_CNTL_BROWN_OUT_INT_CLR	RTC_CNTL_TOUCH_INT_CLR	RTC_CNTL_SAR_INT_CLR	RTC_CNTL_TIME_VALID_INT_CLR	RTC_CNTL_WDT_INT_CLR	RTC_CNTL_SDIO_IDLE_INT_CLR	RTC_CNTL_SLP_REJECT_INT_CLR	RTC_CNTL_SLP_WAKEUP_INT_CLR	
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	Reset

RTC_CNTL_MAIN_TIMER_INT_CLR 1 RTC_CNTL_MAIN_TIMER_INT 中 。(写)

RTC_CNTL_BROWN_OUT_INT_CLR 1 RTC_CNTL_BROWN_OUT_INT 中 。(写)

RTC_CNTL_TOUCH_INT_CLR 1 RTC_CNTL_TOUCH_INT 中 。(写)

RTC_CNTL_SAR_INT_CLR 1 RTC_CNTL_SAR_INT 中 。(写)

RTC_CNTL_TIME_VALID_INT_CLR 1 RTC_CNTL_TIME_VALID_INT 中 。(写)

RTC_CNTL_WDT_INT_CLR 1 RTC_CNTL_WDT_INT 中 。(写)

RTC_CNTL_SDIO_IDLE_INT_CLR 1 RTC_CNTL_SDIO_IDLE_INT 中 。(写)

RTC_CNTL_SLP_REJECT_INT_CLR 1 RTC_CNTL_SLP_REJECT_INT 中 。(写)

RTC_CNTL_SLP_WAKEUP_INT_CLR 1 RTC_CNTL_SLP_WAKEUP_INT 中 。(写)

Register 31.18. RTC_CNTL_STORE*n*_REG (*n*: 0-3) (0x3FF4804C+4**n*)

31																																0	
x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	Reset

RTC_CNTL_STORE*n*_REG 32-bit 保 。(/ 写)

Register 31.19. RTC_CNTL_EXT_XTL_CONF_REG (0x3FF4805C)

RTC_CNTL_XTL_EXT_CTR_EN	RTC_CNTL_XTL_EXT_CTR_LV	(reserved)	
31	30	29 … 0	
0	0	0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0	Reset

RTC_CNTL_XTL_EXT_CTR_EN 部 XTAL 。(/ 写)

RTC_CNTL_XTL_EXT_CTR_LV 0: 时, XTAL 闭; 1: 时, XTAL 闭。(/ 写)

Register 31.20. RTC_CNTL_EXT_WAKEUP_CONF_REG (0x3FF48060)

RTC_CNTL_EXT_WAKEUP1_LV	RTC_CNTL_EXT_WAKEUP0_LV	(reserved)	
31	30	29 … 0	
0	0	0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0	Reset

RTC_CNTL_EXT_WAKEUP1_LV 0: 时, 部 ; 1: 时, 部 。(/ 写)

RTC_CNTL_EXT_WAKEUP0_LV 0: 时, 部 ; 1: 时, 部 。(/ 写)

Register 31.21. RTC_CNTL_SLP_REJECT_CONF_REG (0x3FF48064)

RTC_CNTL_REJECT_CAUSE	RTC_CNTL_DEEP_SLP_REJECT_EN	RTC_CNTL_LIGHT_SLP_REJECT_EN	RTC_CNTL_SDIO_REJECT_EN	RTC_CNTL_GPIO_REJECT_EN	(reserved)	
31 28	27	26	25	24	23 0	
0 0 0 0	0	0	0	0	0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0	Reset

RTC_CNTL_REJECT_CAUSE 。( )

RTC_CNTL_DEEP_SLP_REJECT_EN Deep-sleep。(/ 写)

RTC_CNTL_LIGHT_SLP_REJECT_EN Light-sleep。(/ 写)

RTC_CNTL_SDIO_REJECT_EN SDIO 。(/ 写)

RTC_CNTL_GPIO_REJECT_EN GPIO 。(/ 写)

Register 31.22. RTC_CNTL_CPU_PERIOD_CONF_REG (0x3FF48068)

RTC_CNTL_RTC_CPUPERIOD_SEL	RTC_CNTL_CPUSEL_CONF	(reserved)	
31 30	29	28 0	
0 0	0	0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0	Reset

RTC_CNTL_RTC_CPUPERIOD_SEL CPU 期 。(/ 写)

RTC_CNTL_CPUSEL_CONF CPU 。(/ 写)

Register 31.23. RTC_CNTL_CLK_CONF_REG (0x3FF48070)

RTC_CNTL_ANA_CLK_RTC_SEL	RTC_CNTL_FAST_CLK_RTC_SEL	RTC_CNTL_SOC_CLK_SEL	RTC_CNTL_CK8M_FORCE_PU	RTC_CNTL_CK8M_FORCE_PD	RTC_CNTL_CK8M_DFREQ	(reserved)	RTC_CNTL_CK8M_DIV_SEL	(reserved)	RTC_CNTL_DIG_CLK8M_EN	RTC_CNTL_DIG_CLK8M_D256_EN	RTC_CNTL_DIG_XTAL32K_EN	RTC_CNTL_ENB_CK8M_DIV	RTC_CNTL_ENB_CK8M	RTC_CNTL_CK8M_DIV	(reserved)	
31 30	29	28 27	26	25	24 17	16 15	14 12	11	10	9	8	7	6	5 4	3 0	
0	0	0	0	0	0	0 0	2	0	0	1	0	0	0	0 1	0 0 0 0	Reset

RTC_CNTL_ANA_CLK_RTC_SEL slow_clk_rtc。0：SLOW_CK；1：CK_XTAL_32K；2：CK8M_D256_OUT。(/ 写)

RTC_CNTL_FAST_CLK_RTC_SEL fast_clk_rtc。0：XTAL div 4；1：CK8M 。(/ 写)

RTC_CNTL_SOC_CLK_SEL SOC 时 。0：XTAL；1：PLL；2：CK8M；3: APLL。(/ 写)

RTC_CNTL_CK8M_FORCE_PU CK8M 。(/ 写)

RTC_CNTL_CK8M_FORCE_PD CK8M 闭。(/ 写)

RTC_CNTL_CK8M_DFREQ CK8M_DFREQ。(/ 写)

RTC_CNTL_CK8M_DIV_SEL Divider = reg_rtc_cntl_ck8m_div_sel + 1。(/ 写)

RTC_CNTL_DIG_CLK8M_EN CK8M (: RTC)。(/ 写)

RTC_CNTL_DIG_CLK8M_D256_EN CK8M_D256_OUT (: RTC)。(/ 写)

RTC_CNTL_DIG_XTAL32K_EN CK_XTAL_32K (: RTC)。(/ 写)

RTC_CNTL_ENB_CK8M_DIV 1：CK8M_D256_OUT CK8M；0：CK8M_D256_OUT CK8M 256 。(/ 写)

RTC_CNTL_ENB_CK8M CK8M CK8M_D256_OUT。(/ 写)

RTC_CNTL_CK8M_DIV CK8M_D256_OUT 。00：128 ；01：256 ；10：512 ；11：1024 。(/ 写)

Register 31.24. RTC_CNTL_SDIO_CONF_REG (0x3FF48074)

RTC_CNTL_XPD_SDIO_VREG	RTC_CNTL_DREFH_SDIO	RTC_CNTL_DREFM_SDIO	RTC_CNTL_DREFL_SDIO	RTC_CNTL_REG1P8_READY	RTC_CNTL_SDIO_TIEH	RTC_CNTL_SDIO_FORCE	RTC_CNTL_SDIO_VREG_PD_EN	(reserved)	
31	30 29	28 27	26 25	24	23	22	21	20 0	
0	0 0	0 0	0 1	0	1	0	1	0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0	Reset

RTC_CNTL_XPD_SDIO_VREG XPD_SDIO_VREG , reg_rtc_cntl_sdio_force == 1 时 。(/ 写)

RTC_CNTL_DREFH_SDIO DREFH_SDIO , reg_rtc_cntl_sdio_force == 1 时 。(/ 写)

RTC_CNTL_DREFM_SDIO DREFM_SDIO , reg_rtc_cntl_sdio_force == 1 时 。(/ 写)

RTC_CNTL_DREFL_SDIO DREFL_SDIO , reg_rtc_cntl_sdio_force == 1 时 。(/ 写)

RTC_CNTL_REG1P8_READY REG1P8_READY 。( )

RTC_CNTL_SDIO_TIEH SDIO_TIEH , reg_rtc_cntl_sdio_force == 1 时 。(/ 写)

RTC_CNTL_SDIO_FORCE 1: SDIO_VREG;0: SDIO_VREG。(/ 写)

RTC_CNTL_SDIO_VREG_PD_EN , SDIO_VREG 闭, reg_rtc_cntl_sdio_force == 0 时 。(/ 写)

Register 31.25. RTC_CNTL_VREG_REG (0x3FF4807C)

RTC_CNTL_PREG_FORCE_PU	RTC_CNTL_PREG_FORCE_PD	RTC_CNTL_DBOOST_FORCE_PU	RTC_CNTL_DBOOST_FORCE_PD	RTC_CNTL_DBIAS_WAK	RTC_CNTL_DBIAS_SLP	RTC_CNTL_SCK_DCAP	RTC_CNTL_DIG_VREG_DBIAS_WAK	RTC_CNTL_DIG_VREG_DBIAS_SLP	(reserved)	
31	30	29	28	27 25	24 22	21 14	13 11	10 8	7 0	
1	0	1	0	4	4	0	4	4	0 0 0 0 0 0 0 0	Reset

RTC_CNTL_VREG_FORCE_PU RTC 门 - 。(/ 写)

RTC_CNTL_VREG_FORCE_PD RTC 门 - 闭 (,“ 闭” 0.8V)。(/ 写)

RTC_CNTL_DBOOST_FORCE_PU RTC_DBOOST 。(/ 写)

RTC_CNTL_DBOOST_FORCE_PD RTC_DBOOST 闭。(/ 写)

RTC_CNTL_DBIAS_WAK RTC_DBIAS。(/ 写)

RTC_CNTL_DBIAS_SLP RTC_DBIAS。(/ 写)

RTC_CNTL_SCK_DCAP 门 RTC 时 。(/ 写)

RTC_CNTL_DIG_VREG_DBIAS_WAK 门 DBIAS。(/ 写)

RTC_CNTL_DIG_VREG_DBIAS_SLP 门 DBIAS。(/ 写)

Register 31.26. RTC_CNTL_PWC_REG (0x3FF48080)

(reserved)	RTC_CNTL_PD_EN	RTC_CNTL_FORCE_PU	RTC_CNTL_FORCE_PD	RTC_CNTL_SLOWMEM_PD_EN	RTC_CNTL_SLOWMEM_FORCE_PU	RTC_CNTL_SLOWMEM_FORCE_PD	RTC_CNTL_FASTMEM_PD_EN	RTC_CNTL_FASTMEM_FORCE_PU	RTC_CNTL_FASTMEM_FORCE_PD	RTC_CNTL_SLOWMEM_FORCE_LPU	RTC_CNTL_SLOWMEM_FORCE_LPD	RTC_CNTL_SLOWMEM_FOLW_CPU	RTC_CNTL_FASTMEM_FORCE_LPU	RTC_CNTL_FASTMEM_FORCE_LPD	RTC_CNTL_FASTMEM_FOLW_CPU	RTC_CNTL_FORCE_NOISO	RTC_CNTL_FORCE_ISO	RTC_CNTL_SLOWMEM_FORCE_ISO	RTC_CNTL_SLOWMEM_FORCE_NOISO	RTC_CNTL_FASTMEM_FORCE_ISO	RTC_CNTL_FASTMEM_FORCE_NOISO										
31																																21
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	0	1	0	1	0	0	1	0	0	1	0	0	1	0	1	Reset

RTC_CNTL_PD_EN , 闭 rtc_peri。(/ 写)

RTC_CNTL_FORCE_PU rtc_peri 。(/ 写)

RTC_CNTL_FORCE_PD rtc_peri 闭。(/ 写)

RTC_CNTL_SLOWMEM_PD_EN , 闭 RTC 。(/ 写)

RTC_CNTL_SLOWMEM_FORCE_PU RTC 。(/ 写)

RTC_CNTL_SLOWMEM_FORCE_PD RTC 闭。(/ 写)

RTC_CNTL_FASTMEM_PD_EN , 闭 RTC 。(/ 写)

RTC_CNTL_FASTMEM_FORCE_PU RTC 。(/ 写)

RTC_CNTL_FASTMEM_FORCE_PD RTC 闭。(/ 写)

RTC_CNTL_SLOWMEM_FORCE_LPU RTC 。(/ 写)

RTC_CNTL_SLOWMEM_FORCE_LPD RTC 闭 。(/ 写)

RTC_CNTL_SLOWMEM_FOLW_CPU 1：RTC 闭， CPU 变化 0：RTC 闭 PD， RTC 变化 (/ 写)

RTC_CNTL_FASTMEM_FORCE_LPU RTC 。(/ 写)

RTC_CNTL_FASTMEM_FORCE_LPD RTC 闭 。(/ 写)

RTC_CNTL_FASTMEM_FOLW_CPU 1： RTC 闭 , CPU 变化 0： RTC 闭 , RTC 变化 (/ 写)

RTC_CNTL_FORCE_NOISO rtc_peri 不 。(/ 写)

RTC_CNTL_FORCE_ISO rtc_peri 。(/ 写)

RTC_CNTL_SLOWMEM_FORCE_ISO RTC 。(/ 写)

RTC_CNTL_SLOWMEM_FORCE_NOISO RTC 不 。(/ 写)

RTC_CNTL_FASTMEM_FORCE_ISO RTC 。(/ 写)

RTC_CNTL_FASTMEM_FORCE_NOISO RTC 不 。(/ 写)

Register 31.27. RTC_CNTL_DIG_PWC_REG (0x3FF48084)

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0	
RTC_CNTL_DG_WRAP_PD_EN	RTC_CNTL_WIFI_PD_EN	RTC_CNTL_INTER_RAM4_PD_EN	RTC_CNTL_INTER_RAM3_PD_EN	RTC_CNTL_INTER_RAM2_PD_EN	RTC_CNTL_INTER_RAM1_PD_EN	RTC_CNTL_INTER_RAM0_PD_EN	RTC_CNTL_ROM0_PD_EN	(reserved)			RTC_CNTL_DG_WRAP_FORCE_PU	RTC_CNTL_DG_WRAP_FORCE_PD	RTC_CNTL_WIFI_FORCE_PU	RTC_CNTL_WIFI_FORCE_PD	RTC_CNTL_INTER_RAM4_FORCE_PU	RTC_CNTL_INTER_RAM4_FORCE_PD	RTC_CNTL_INTER_RAM3_FORCE_PU	RTC_CNTL_INTER_RAM3_FORCE_PD	RTC_CNTL_INTER_RAM2_FORCE_PU	RTC_CNTL_INTER_RAM2_FORCE_PD	RTC_CNTL_INTER_RAM1_FORCE_PU	RTC_CNTL_INTER_RAM1_FORCE_PD	RTC_CNTL_INTER_RAM0_FORCE_PU	RTC_CNTL_INTER_RAM0_FORCE_PD	RTC_CNTL_ROM0_FORCE_PU	RTC_CNTL_ROM0_FORCE_PD	RTC_CNTL_LSLP_MEM_FORCE_PU	RTC_CNTL_LSLP_MEM_FORCE_PD	(reserved)		
x	x	x	x	x	x	x	x	0	0	0	1	0	1	0	1	0	1	0	1	0	1	0	1	0	1	0	1	0	0	0	0	Reset

RTC_CNTL_DG_WRAP_PD_EN , 闭 。(/ 写)

RTC_CNTL_WIFI_PD_EN , 闭 Wi-Fi 。(/ 写)

RTC_CNTL_INTER_RAM4_PD_EN , 闭 部 SRAM 4。(/ 写)

RTC_CNTL_INTER_RAM3_PD_EN , 闭 部 SRAM 3。(/ 写)

RTC_CNTL_INTER_RAM2_PD_EN , 闭 部 SRAM 2。(/ 写)

RTC_CNTL_INTER_RAM1_PD_EN , 闭 部 SRAM 1。(/ 写)

RTC_CNTL_INTER_RAM0_PD_EN , 闭 部 SRAM 0。(/ 写)

RTC_CNTL_ROM0_PD_EN , 闭 ROM。(/ 写)

RTC_CNTL_DG_WRAP_FORCE_PU 。(/ 写)

RTC_CNTL_DG_WRAP_FORCE_PD 闭。(/ 写)

RTC_CNTL_WIFI_FORCE_PU Wi-Fi 。(/ 写)

RTC_CNTL_WIFI_FORCE_PD Wi-Fi 闭。(/ 写)

RTC_CNTL_INTER_RAM4_FORCE_PU 部 SRAM 4 。(/ 写)

RTC_CNTL_INTER_RAM4_FORCE_PD 部 SRAM 4 闭。(/ 写)

RTC_CNTL_INTER_RAM3_FORCE_PU 部 SRAM 3 。(/ 写)

RTC_CNTL_INTER_RAM3_FORCE_PD 部 SRAM 3 闭。(/ 写)

RTC_CNTL_INTER_RAM2_FORCE_PU 部 SRAM 2 。(/ 写)

RTC_CNTL_INTER_RAM2_FORCE_PD 部 SRAM 2 闭。(/ 写)

RTC_CNTL_INTER_RAM1_FORCE_PU 部 SRAM 1 。(/ 写)

RTC_CNTL_INTER_RAM1_FORCE_PD 部 SRAM 1 闭。(/ 写)

RTC_CNTL_INTER_RAM0_FORCE_PU 部 SRAM 0 。(/ 写)

RTC_CNTL_INTER_RAM0_FORCE_PD 部 SRAM 0 闭。(/ 写)

寄存器描述下一页继续。

Register 31.27. RTC_CNTL_DIG_PWC_REG (0x3FF48084)

继上一页寄存器描述。

RTC_CNTL_ROM0_FORCE_PU ROM 。(/ 写)

RTC_CNTL_ROM0_FORCE_PD ROM 闭。(/ 写)

RTC_CNTL_LSLP_MEM_FORCE_PU , 。(/ 写)

RTC_CNTL_LSLP_MEM_FORCE_PD , 闭。(/ 写)

Register 31.28. RTC_CNTL_DIG_ISO_REG (0x3FF48088)

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8 … 0	
RTC_CNTL_DG_WRAP_FORCE_NOISO	RTC_CNTL_DG_WRAP_FORCE_ISO	RTC_CNTL_WIFI_FORCE_NOISO	RTC_CNTL_WIFI_FORCE_ISO	RTC_CNTL_INTER_RAM4_FORCE_NOISO	RTC_CNTL_INTER_RAM4_FORCE_ISO	RTC_CNTL_INTER_RAM3_FORCE_NOISO	RTC_CNTL_INTER_RAM3_FORCE_ISO	RTC_CNTL_INTER_RAM2_FORCE_NOISO	RTC_CNTL_INTER_RAM2_FORCE_ISO	RTC_CNTL_INTER_RAM1_FORCE_NOISO	RTC_CNTL_INTER_RAM1_FORCE_ISO	RTC_CNTL_INTER_RAM0_FORCE_NOISO	RTC_CNTL_INTER_RAM0_FORCE_ISO	RTC_CNTL_ROM0_FORCE_NOISO	RTC_CNTL_ROM0_FORCE_ISO	RTC_CNTL_DG_PAD_FORCE_HOLD	RTC_CNTL_DG_PAD_FORCE_UNHOLD	RTC_CNTL_DG_PAD_FORCE_ISO	RTC_CNTL_DG_PAD_FORCE_NOISO	RTC_CNTL_REG_RTC_CNTL_DG_PAD_AUTOHOLD_EN	RTC_CNTL_CLR_REG_RTC_CNTL_DG_PAD_AUTOHOLD	RTC_CNTL_DG_PAD_AUTOHOLD	(reserved)	
1	0	1	0	1	0	1	0	1	0	1	0	1	0	1	0	0	1	0	1	0	0	0	0 0 0 0 0 0 0 0 0	Reset

RTC_CNTL_DG_WRAP_FORCE_NOISO 不 。(/ 写)

RTC_CNTL_DG_WRAP_FORCE_ISO 。(/ 写)

RTC_CNTL_WIFI_FORCE_NOISO Wi-Fi 不 。(/ 写)

RTC_CNTL_WIFI_FORCE_ISO Wi-Fi 。(/ 写)

RTC_CNTL_INTER_RAM4_FORCE_NOISO 部 SRAM 4 不 。(/ 写)

RTC_CNTL_INTER_RAM4_FORCE_ISO 部 SRAM 4 。(/ 写)

RTC_CNTL_INTER_RAM3_FORCE_NOISO 部 SRAM 3 不 。(/ 写)

RTC_CNTL_INTER_RAM3_FORCE_ISO 部 SRAM 3 。(/ 写)

RTC_CNTL_INTER_RAM2_FORCE_NOISO 部 SRAM 2 不 。(/ 写)

RTC_CNTL_INTER_RAM2_FORCE_ISO 部 SRAM 2 。(/ 写)

RTC_CNTL_INTER_RAM1_FORCE_NOISO 部 SRAM 1 不 。(/ 写)

RTC_CNTL_INTER_RAM1_FORCE_ISO 部 SRAM 1 。(/ 写)

RTC_CNTL_INTER_RAM0_FORCE_NOISO 部 SRAM 0 不 。(/ 写)

RTC_CNTL_INTER_RAM0_FORCE_ISO 部 SRAM 0 。(/ 写)

RTC_CNTL_ROM0_FORCE_NOISO ROM 不 。(/ 写)

RTC_CNTL_ROM0_FORCE_ISO ROM 。(/ 写)

RTC_CNTL_DG_PAD_FORCE_HOLD 保 。(/ 写)

RTC_CNTL_DG_PAD_FORCE_UNHOLD 保 。(/ 写)

RTC_CNTL_DG_PAD_FORCE_ISO 。(/ 写)

RTC_CNTL_DG_PAD_FORCE_NOISO 不 。(/ 写)

寄存器描述下一页继续。

Register 31.28. RTC_CNTL_DIG_ISO_REG (0x3FF48088)

继上一页寄存器描述。

RTC_CNTL_REG_RTC_CNTL_DG_PAD_AUTOHOLD_EN 保 。(/ 写)

RTC_CNTL_CLR_REG_RTC_CNTL_DG_PAD_AUTOHOLD 写 , 保 。(写)

RTC_CNTL_DG_PAD_AUTOHOLD , 表 保 。( )

Register 31.29. RTC_CNTL_WDTCONFIG0_REG (0x3FF4808C)

RTC_CNTL_WDT_EN	RTC_CNTL_WDT_STG0	RTC_CNTL_WDT_STG1	RTC_CNTL_WDT_STG2	RTC_CNTL_WDT_STG3	reserved	reserved	RTC_CNTL_WDT_CPU_RESET_LENGTH	RTC_CNTL_WDT_SYS_RESET_LENGTH	RTC_CNTL_WDT_FLASHBOOT_MOD_EN	RTC_CNTL_WDT_PROCPU_RESET_EN	RTC_CNTL_WDT_APPCPU_RESET_EN	RTC_CNTL_WDT_PAUSE_IN_SLP	reserved	
31	30 28	27 25	24 22	21 19	18	17	16 14	13 11	10	9	8	7	6 0	
0	0	0	0	0	0	0	1	1	1	0	0	1	0	Reset

RTC_CNTL_WDT_PAUSE_IN_SLP 1 设 “ 中 RTC WDT ”。(/写)

RTC_CNTL_WDT_APPCPU_RESET_EN 1 RTC WDT APP_CPU。(/写)

RTC_CNTL_WDT_PROCPU_RESET_EN 1 RTC WDT PRO_CPU。(/写)

RTC_CNTL_WDT_FLASHBOOT_MOD_EN 1 flash 启 时 RTC WDT。(/写)

RTC_CNTL_WDT_SYS_RESET_LENGTH 设 时长，单 : RTC_SLOW_CLK, : 0 ~ 7。(/写)

RTC_CNTL_WDT_CPU_RESET_LENGTH 设 CPU 时长单 : RTC_SLOW_CLK, : 0 ~ 7。(/写)

RTC_CNTL_WDT_STG3 1: 中 ; 2: CPU ; 3: ; 4: RTC 。(/写)

RTC_CNTL_WDT_STG2 1: 中 ; 2: CPU ; 3: ; 4: RTC 。(/写)

RTC_CNTL_WDT_STG1 1: 中 ; 2: CPU ; 3: ; 4: RTC 。(/写)

RTC_CNTL_WDT_STG0 1: 中 ; 2: CPU ; 3: ; 4: RTC 。(/写)

RTC_CNTL_WDT_EN RTC WDT。(/写)

Register 31.30. RTC_CNTL_WDTCONFIG*n*_REG (*n*: 1-4) (0x3FF4808C+4*n*)

31	0	
0x000000FFF		Reset

RTC_CNTL_WDTCONFIG*n*_REG 时 *n* 保 期。(/ 写)

Register 31.31. RTC_CNTL_WDTFEED_REG (0x3FF480A0)

RTC_CNTL_WDT_FEED	(reserved)	
31	30 … 0	
0	0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0	Reset

RTC_CNTL_WDT_FEED 。(写)

Register 31.32. RTC_CNTL_WDTWPROTECT_REG (0x3FF480A4)

31	0	
0x050D83AA1		Reset

RTC_CNTL_WDTWPROTECT_REG RTC_CNTL_WDTWPROTECT 不 0x50d83aa1 时，RTC 时 写保 ， 时 RTC_CNTL_WDTCONFIG*n*_REG 改。(/ 写)

Register 31.33. RTC_CNTL_SW_CPU_STALL_REG (0x3FF480AC)

RTC_CNTL_SW_STALL_PROCPU_C1	RTC_CNTL_SW_STALL_APPCPU_C1	(reserved)	
31 26	25 20	19 0	
0 0 0 0 0 0	0 0 0 0 0 0	0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0	Reset

RTC_CNTL_SW_STALL_PROCPU_C1 reg_rtc_cntl_sw_stall_procpu_c1[5:0], reg_rtc_cntl_sw_stall_procpu_c0[1:0] == 0x86 (100001 10) 时，PRO_CPU ，参见 RTC_CNTL_OPTIONS0_REG。(R/W)

RTC_CNTL_SW_STALL_APPCPU_C1 reg_rtc_cntl_sw_stall_appcpu_c1[5:0], reg_rtc_cntl_sw_stall_appcpu_c0[1:0] == 0x86 (100001 10) 时，APP_CPU ，参见 RTC_CNTL_OPTIONS0_REG。(R/W)

Register 31.34. RTC_CNTL_STORE*n*_REG (*n*: 4-7) (0x3FF480B0+4*n)

31 0	
x x x x x x x x x x x x x x x x x x x x x x x x x x x x x x x x	Reset

RTC_CNTL_STORE*n*_REG 32 保 。(R/W)

Register 31.35. RTC_CNTL_HOLD_FORCE_REG (0x3FF480C8)

(reserved)	RTC_CNTL_X32N_HOLD_FORCE	RTC_CNTL_X32P_HOLD_FORCE	RTC_CNTL_TOUCH_PAD7_HOLD_FORCE	RTC_CNTL_TOUCH_PAD6_HOLD_FORCE	RTC_CNTL_TOUCH_PAD5_HOLD_FORCE	RTC_CNTL_TOUCH_PAD4_HOLD_FORCE	RTC_CNTL_TOUCH_PAD3_HOLD_FORCE	RTC_CNTL_TOUCH_PAD2_HOLD_FORCE	RTC_CNTL_TOUCH_PAD1_HOLD_FORCE	RTC_CNTL_TOUCH_PAD0_HOLD_FORCE	RTC_CNTL_SENSE4_HOLD_FORCE	RTC_CNTL_SENSE3_HOLD_FORCE	RTC_CNTL_SENSE2_HOLD_FORCE	RTC_CNTL_SENSE1_HOLD_FORCE	RTC_CNTL_PDAC2_HOLD_FORCE	RTC_CNTL_PDAC1_HOLD_FORCE	RTC_CNTL_ADC2_HOLD_FORCE	RTC_CNTL_ADC1_HOLD_FORCE	
31 … 18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0	
0 0 0 0 0 0 0 0 0 0 0 0 0 0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	Reset

RTC_CNTL_X32N_HOLD_FORCE , 1 保 。(/ 写)

RTC_CNTL_X32P_HOLD_FORCE , 1 保 。(/ 写)

RTC_CNTL_TOUCH_PAD7_HOLD_FORCE , 1 保 。(/ 写)

RTC_CNTL_TOUCH_PAD6_HOLD_FORCE , 1 保 。(/ 写)

RTC_CNTL_TOUCH_PAD5_HOLD_FORCE , 1 保 。(/ 写)

RTC_CNTL_TOUCH_PAD4_HOLD_FORCE , 1 保 。(/ 写)

RTC_CNTL_TOUCH_PAD3_HOLD_FORCE , 1 保 。(/ 写)

RTC_CNTL_TOUCH_PAD2_HOLD_FORCE , 1 保 。(/ 写)

RTC_CNTL_TOUCH_PAD1_HOLD_FORCE , 1 保 。(/ 写)

RTC_CNTL_TOUCH_PAD0_HOLD_FORCE , 1 保 。(/ 写)

RTC_CNTL_SENSE4_HOLD_FORCE , 1 保 。(/ 写)

RTC_CNTL_SENSE3_HOLD_FORCE , 1 保 。(/ 写)

RTC_CNTL_SENSE2_HOLD_FORCE , 1 保 。(/ 写)

RTC_CNTL_SENSE1_HOLD_FORCE , 1 保 。(/ 写)

RTC_CNTL_PDAC2_HOLD_FORCE , 1 保 。(/ 写)

RTC_CNTL_PDAC1_HOLD_FORCE , 1 保 。(/ 写)

RTC_CNTL_ADC2_HOLD_FORCE , 1 保 。(/ 写)

RTC_CNTL_ADC1_HOLD_FORCE , 1 保 。(/ 写)

Register 31.36. RTC_CNTL_EXT_WAKEUP1_REG (0x3FF480CC)

(reserved)	RTC_CNTL_EXT_WAKEUP1_STATUS_CLR	RTC_CNTL_EXT_WAKEUP1_SEL	
31 19	18	17 0	
0 0 0 0 0 0 0 0 0 0 0 0 0	0	0	Reset

RTC_CNTL_EXT_WAKEUP1_STATUS_CLR 部 1 。(写)

RTC_CNTL_EXT_WAKEUP1_SEL 部 RTC 。(/ 写)

Register 31.37. RTC_CNTL_EXT_WAKEUP1_STATUS_REG (0x3FF480D0)

(reserved)	RTC_CNTL_EXT_WAKEUP1_STATUS	
31 18	17 0	
0 0 0 0 0 0 0 0 0 0 0 0 0 0	0	Reset

RTC_CNTL_EXT_WAKEUP1_STATUS 部 1 。( )

Register 31.38. RTC_CNTL_BROWN_OUT_REG (0x3FF480D4)

RTC_CNTL_BROWN_OUT_DET	RTC_CNTL_BROWN_OUT_ENA	RTC_CNTL_DBROWN_OUT_THRES	RTC_CNTL_BROWN_OUT_RST_ENA	RTC_CNTL_BROWN_OUT_RST_WAIT	RTC_CNTL_BROWN_OUT_PD_RF_ENA	RTC_CNTL_BROWN_OUT_CLOSE_FLASH_ENA	(reserved)	
31	30	29 27	26	25 16	15	14	13 0	
0	0	0x2	0	0x3FF	0	0	0 0 0 0 0 0 0 0 0 0 0 0 0 0	Reset

RTC_CNTL_BROWN_OUT_DET 测。( )

RTC_CNTL_BROWN_OUT_ENA 测。(/ 写)

RTC_CNTL_DBROWN_OUT_THRES 。 时， 测 。 ， 个 ESP32 间 可 差 。0：2.43 V ± 0.05；1：2.48 V ± 0.05；2：2.58 V ± 0.05；3：2.62 V ± 0.05；4：2.67 V ± 0.05；5：2.70 V ± 0.05；6：2.77 V ± 0.05；7：2.80 V ± 0.05。(/ 写)

RTC_CNTL_BROWN_OUT_RST_ENA 。(/ 写)

RTC_CNTL_BROWN_OUT_RST_WAIT 期。(/ 写)

RTC_CNTL_BROWN_OUT_PD_RF_ENA 时， 闭 RF。(/ 写)

RTC_CNTL_BROWN_OUT_CLOSE_FLASH_ENA 时， 闭 flash(flash sus-pend)。(/ 写)

词汇列表

外设相关词汇

AES	AES
BOOTCTRL	启 Boot
DS	
DMA	DMA
eFuse	eFuse
HMAC	HMAC
I2C	I2C
I2S	I2S
LEDC	LED PWM
MCPWM	PWM
PCNT	脉
RMT	
RNG	
RSA	RSA
SDHOST	SD/MMC
SHA	SHA
SPI	SPI
SYSTIMER	时
TIMG	时
TWAI	
UART	UART
ULP	超
USB OTG	USB On-The-Go
WDT	时

寄存器相关词汇

ISO	。 时， （ 中间 ）。“ISO” 一个 ， 而不 。
NMI	不可屏蔽中 。
REG	。
R/W	/写， 可 写 。
RO	， 。
SYSREG	。
WO	写， 写 。

修订历史

期	版本	布
2021.07	v4.5	10.6.4 表 4-4 广 一 14 *LED PWM 控制器 (LEDC)* 表 14-1 3.2.7 10.9 中 Ethernet MAC 0x3FF6_9000 13 *UART 控制器 (UART)* UART_SW_RTS、UART_RX_FLOW_THRHD 21 *双线汽车接口 (TWAI)* 25 *随机数发生器 (RNG)* 30.4.2 5 *DPort 寄存器* 2 *中断矩阵 (INTERRUPT)* 中 中 4 *IO_MUX 和 GPIO 交换矩阵 (GPIO, IO_MUX)* 中 编 ，编 Function0 始 13.3.4 中重 表 2-1 6.4 中 笔
2021.03	V4.4	16：*电机控制脉宽调制器 (PWM)* 中 PWM_TIMER0_SYNC_REG ~ PWM_TIMER2_SYNC_REG 18：*定时器组 (TIMG)* 中 TIMG*n*_RTCCALICFG_REG TIMG*n*_RTCCALICFG1_REG 31：*低功耗管理 (RTC_CNTL)* 中 RTC_CNTL_WDTCONFIG0_REG 10：*以太网 (MAC)* 中 EMACMIIADDR_REG 4：*IO_MUX 和 GPIO 交换矩阵 (GPIO, IO_MUX)* 中 IO_MUX_*x*_REG (*x*: GPIO0-GPIO39) IO_MUX_*x*_REG (*x*: GPIO0-GPIO39) 21：*双线汽车接口 (TWAI)* 中 标 31：*低功耗管理 (RTC_CNTL)* 中 RTC_CNTL_ULP_CP_TIMER_REG 写 重 表，并 表 4-3：*IO_MUX Pad 列表* 广
2020.09	V4.3	21 *双线汽车接口 (TWAI)* *eFuse 控制器* 中 uart_download_dis 参 。 25 *随机数发生器 (RNG)* 中 25.4 *编程指南* ，并 部 。 SPI_ADDR_REG SPI_SLV_WR_STATUS_REG

见下页

续上页

期	版本	布
2020.06	V4.2	写 系统和存储器： • 表 1-6 扩 DPORT AHB 间 复位和时钟： • 表 3-3： PLL_CLK 480 MHz，CPU_CLK 240 MHz • 表 3-5： CPU_CLK PLL_CLK 时 APB_CLK 改 80 MHz *IO_MUX 和 GPIO 交换矩阵*： • 4.4.2 中 一 笔 ： ，必 SIG_IN_SEL ， 设。 • 改 表 4-3 中 MTCK、MTMS、GPIO27 reset • FUN_DRV *I²S*： • 12.4.1.1 UART ： • 13.3.3 • UART_FIFO_REG UART_RX_TOUT_THRHD *LED_PWM*： • 表 14-1 电机控制脉宽调制器 *(MCPWM)*： • 、 、 - PWM 期 *PULSE_CNT*： • PCNT_U*n*_STATUS_REG *eFuse* 控制器： • 个 参 “32pad” “chip_version” 并 一个：pkg_version • EFUSE_RD_CHIP_VER_PKG EFUSE_CHIP_VER_PKG 片上传感器与模拟信号处理： • 笔 超低功耗协处理器： • 30.4.13 30.4.14 中 • 笔 ： • RTC_CNTL_WDTCONFIG0_REG • 改 RTC_CNTL_WDTCONFIG*n*_REG

见下页

续上页

期	版本	布
2019.11	V4.1	*IO_MUX 和 GPIO 交换矩阵*： • 表 4-4； • RTCIO_TOUCH_PADn_REG 中 RTCIO_TOUCH_PADn_FUN_SEL ； *SPI*： • 表 7.7 中 SPI2、SPI3 ； *I²C 控制器*： • I2C_SLAVE_TRAN_COMP_INT 中 ； *I²S*： • 表 12-1 — ； *UART 控制器*： • UART_FORCE_XOFF UART_FORCE_XON ； • ART_SWFC_CONF_REG ； *红外遥控*： • 15-1； *PULSE_CNT*： • 17-1； • PCNT_Un_CONF0_REG 笔 ； *eFuse 控制器*： • 并 已 8 个 参 ； ； • 表 20-4 中 ； • 参 flash_crypt_cnt 已 7 ； *PID/MPU/MMU*： • 表 27-8 — ； *片上传感器与模拟信号处理*： • SENS_SAR2_BIT_WIDTH SENS_SAR1_BIT_WIDTH ； *超低功耗协处理器*： • TSENS ； • REG_WR OpCode； • 30.6.2.4； • RTC_I2C_RX_LSB_FIRST RTC_I2C_TX_LSB_FIRST ； • RTC_I2C_SLAVE_TRAN_COMP_INT_ENA RTC_I2C_SLAVE_TRAN_COMP_INT_ST ； *低功耗管理*： • RTC_CNTL_DBROWN_OUT_THRES 认 ； • RTC_CNTL_BROWN_OUT_CLOSE_FLASH_ENA ； 。

见下页

续上页

期	版本	布
2018.12	V4.0	IO_MUX GPIO 中 ， 保 一 ； 7 SPI： • 7.3； • 7.5.1； • 7.8； 13 UART ： • UART 1/1.5/2/3 个； • 13.3.2 一 ； • UART_DL0_EN ； 一 中 一 。
2018.10	V3.9	11：I²C 中 11-3：I²C 时 。
2018.09	V3.8	TIMG*n*_T*x*_ALARM_EN ； 30.5 中 ULP 时间 。
2018.08	V3.7	UART_RX_GAP_TOUT 。
2018.08	V3.6	30.4.6 中 ； UART_ACTIVE_THRESHOLD 。
2018.07	V3.5	15 RMT： • 15.2.2：RMT RAM 中 RAM 始 ； • RMT ； • RMT_APB_CONF_REG 。 UART_RX_TOUT_THRHD、UART_RXFIFO_FULL_INT_CLR、UART_RXFIFO_FULL_INT_CLR 。
2018.06	V3.4	4.8 ESP32 I/O Pad 中 ； 11.3.3 I²C 时 ； 14.2.3 LEDC 中 ； 17.2.3 PULSE_CNT 察 中 “ ”； 超 声 。
2018.05	V3.3	中 表 中 。
2018.04	V3.2	8-3 CMD53 。 Ethernet MAC 中 6 个 ： • DMAOPERATION_MODE_REG; • DMAIN_EN_REG, • DMAMISSEDFR_REG, • PMT_RWUFFR_REG, • PMT_CSR_REG, • EMACLPI_CSR_REG, and • EMACLPITIMERSCONTROL_REG.
2018.04	V3.1	15-1 RMT ； 4.7 中 ； 4.46 中 ；
2018.03	V3.0	30.4.2 中 ST ；

见下页

续上页

期	版本	布
		10.9 10.10 中 EMACADDR2HIGH_REG EMACADDR7LOW_REG 。
2018.02	V2.9	4.2.2、4.2.3、4.3.2; I2S 中 I2S_FIFO_WR_REG I2S_FIFO_RD_REG 。
2018.01	V2.8	MAC。 eFuse 中 参 BLK3_part_reserve 。
2017.12	V2.7	中 Cache; LED_PWM 中 个 ; eFuse 中 console_debug_disable 。
2017.11	V2.6	: • 15-1 RMT ; • RMT RAM; • ; • 中 RMT_CHn_TX_THR_EVENT_INT 。 UART RAM UART_CONF0_REG 中 。
2017.11	V2.5	SPI 表中 SPI_CTRL_REG ; SD/MMC 中 时 , CLK_EDGE_SEL ; I2C 重 。
2017.09	V2.4	SDIO 中 SLC0HOST_TOKEN_RDATA ; I2S 时 中 ; GP-SPI 中 ; DPort ; DMA 。
2017.08	V2.3	Flash 。
2017.07	V2.2	。
2017.07	V2.1	IO_MUX GPIO 中 GPIO / GPIO RTC ; PID 。
2017.07	V2.0	SDIO 。
2017.06	V1.9	IO_MUX GPIO ; 脉 冲 (MCPWM) 。
2017.06	V1.8	I2S 中 I2S_STATE_REG ; IO_MUX GPIO ; 超 。
2017.05	V1.7	方 ; PLL; eFuse 表; I2S PDM LCD ; : GP-SPI 。
2017.03	V1.6	SD/MMC ; IO_MUX GPIO 中 IO_MUX_PIN_CTRL 。

见下页

续上页

期	版本	布
2017.03	V1.5	I2S。
2017.01	V1.4	SPI； UART 。
2016.12	V1.3	eFuse ； RSA ； ； I2C 中 I2C 。
2016.11	V1.2	PID/MPU/MMU； IO_MUX GPIO 表； LED_PWM 表。
2016.09	V1.1	I2C 。
2016.08	V1.0	发 布。

www.espressif.com

免责声明和版权公告

本 中 ，包括 参 URL ， 变 ， 不 。
本 可 三方 ， “按 ” ， 不
、 保 。
不 本 保 ，包括 、 ， 不
案、 方 保 。
不 本 三方 保 ， 不 本
识产 。本 方 识产
可，不 可 暗 可。
Wi-Fi 标 Wi-Fi 。 标 Bluetooth SIG 册 标。
中 标 、 标 册 标 财产， 产 。

版权归 © 2021 乐鑫信息科技（上海）股份有限公司。保留所有权利。