

飞腾派软件编程手册

V1.0 (2023-10-16)

版权所有 © 飞腾信息技术有限公司 2023。保留所有权利。

未经本公司同意，任何单位、公司或个人不得擅自复制、翻译、摘抄本文档内容的部分或全部，不得以任何方式或途径进行传播和宣传。

商标声明

Phytium和其他飞腾商标均为飞腾信息技术有限公司的商标。
本文档提及的其他所有商标或注册商标，由各自的所有人拥有。

注意

本文档的内容视为飞腾的保密信息，您应当严格遵守保密义务；未经飞腾事先书面同意，您不得向任何第三方披露本文档内容或提供给任何第三方使用。

由于产品版本升级或其他原因，本文档内容会不定期进行更新。除非另有约定，本文档仅作为使用指导，飞腾在现有技术的基础上尽最大努力提供相应的介绍及操作指引，但飞腾在此明确声明对本文档内容的准确性、完整性、适用性、可靠性等不作任何明示或暗示的保证。本文档中的所有陈述、信息和建议不构成任何明示或暗示的担保。

本文档中所有内容，包括但不限于图片、架构设计、页面布局、文字描述，均由飞腾和/或其关联公司依法拥有其知识产权，包括但不限于商标权、专利权、著作权等。非经飞腾和/或其关联公司书面同意，任何人不得擅自使用、修改、复制上述内容。
如有技术问题，可联系 support@phytium.com.cn 获取支持。

飞腾信息技术有限公司

地址：天津市滨海新区海缘路1号信安创业广场5号楼

网址：www.phytium.com.cn

版本历史

文档版本	发布日期	更新说明
V1.0	2023-10-16	首次发布。

目 录

1 概述	1
2 地址空间分配	2
2.1 总体地址空间划分	2
2.2 中断控制器内部地址划分	2
2.3 片上设备地址空间划分	3
3 计算单元	6
3.1 浮点功能部件	6
3.2 SIMD 功能部件	6
4 存储	8
4.1 Cache	8
4.1.1 L1 Cache	9
4.1.2 L2 Cache	10
4.2 Memory	10
4.2.1 寄存器列表	10
4.2.2 寄存器说明	11
5 外设	15
5.1 PCIe 控制器	15
5.1.1 简介	15
5.1.2 树形拓扑	15
5.1.3 地址空间划分	16
5.1.4 配置空间寻址	16
5.1.5 PCIe 控制器寄存器列表	17
5.1.6 PCIe 控制器寄存器说明	18
5.1.7 HPB 寄存器列表	24
5.1.8 HPB 寄存器说明	24
5.1.9 DMA 引擎技术	32

5.1.10 DMA Engine 寄存器列表	35
5.1.11 DMA Engine 寄存器说明	36
5.1.12 描述符配置寄存器列表	42
5.1.13 描述符配置寄存器说明	42
5.2 SMMU	45
5.2.1 操作说明	45
5.2.2 寄存器列表	46
5.2.3 寄存器说明	46
5.3 以太网控制器	52
5.3.1 操作说明	52
5.3.2 寄存器列表	57
5.3.3 寄存器说明	61
5.3.4 描述符表	104
5.4 USB3.0 控制器	110
5.4.1 操作说明	110
5.4.2 寄存器列表	111
5.4.3 寄存器说明	112
5.5 USB2.0 OTG 控制器	114
5.5.1 操作说明	114
5.5.2 寄存器列表	115
5.5.3 寄存器说明	116
5.6 USB HUB 控制器	126
5.6.1 操作说明	126
5.6.2 寄存器列表	127
5.6.3 寄存器说明	128
5.7 SATA 控制器	131
5.8 DC	132
5.8.1 简介	132
5.8.2 操作说明	133
5.8.3 DC 寄存器列表	138
5.8.4 DC 寄存器说明	141
5.8.5 DP 寄存器列表	152
5.8.6 DP 寄存器说明	154
5.8.7 DP-PHY 寄存器列表	173

5.8.8 DP-PHY 寄存器说明	174
5.9 VPU 控制器	176
5.9.1 简介	176
5.9.2 寄存器列表	177
5.9.3 寄存器说明	177
5.10 GDMA 控制器	178
5.10.1 简介	178
5.10.2 操作说明	178
5.10.3 寄存器列表	181
5.10.4 寄存器说明	181
5.11 DDMA 控制器	187
5.11.1 简介	187
5.11.2 操作说明	188
5.11.3 寄存器列表	191
5.11.4 寄存器说明	192
5.12 硬件信号量 (Semaphore)	197
5.12.1 简介	197
5.12.2 操作说明	198
5.12.3 寄存器列表	198
5.12.4 寄存器说明	198
5.13 看门狗(WDT)	200
5.13.1 操作说明	201
5.13.2 寄存器列表	201
5.13.3 寄存器说明	202
5.14 QSPI 控制器	203
5.14.1 操作说明	203
5.14.2 寄存器列表	205
5.14.3 寄存器说明	206
5.15 SPI 控制器	211
5.15.1 操作说明	211
5.15.2 寄存器列表	212
5.15.3 寄存器说明	213
5.16 SD/SDIO/eMMC 控制器	220
5.16.1 操作说明	220

5.16.2 寄存器列表	228
5.16.3 寄存器说明	229
5.17 NAND Flash 控制器	239
5.17.1 操作说明	239
5.17.2 寄存器列表	241
5.17.3 寄存器说明	242
5.18 I2S 控制器	253
5.18.1 操作说明	254
5.18.2 寄存器列表	254
5.18.3 寄存器说明	255
5.19 CAN 控制器	260
5.19.1 操作说明	260
5.19.2 寄存器列表	261
5.19.3 寄存器说明	262
5.20 JTAG Master 控制器	270
5.20.1 操作说明	270
5.20.2 寄存器列表	273
5.20.3 寄存器说明	273
5.21 MIO 控制器	276
5.21.1 操作说明	276
5.21.2 寄存器列表	277
5.21.3 寄存器说明	277
5.22 UART 控制器	278
5.22.1 操作说明	278
5.22.2 寄存器列表	280
5.22.3 寄存器说明	280
5.23 I2C/SMBUS 控制器	288
5.23.1 操作说明	288
5.23.2 I2C 寄存器列表	292
5.23.3 I2C 寄存器说明	293
5.23.4 SMBUS 寄存器列表	308
5.23.5 SMBUS 寄存器说明	309
5.24 PWM 控制器	315
5.24.1 操作说明	315

5. 24. 2 寄存器列表	318
5. 24. 3 寄存器说明	319
5. 25 Timer	322
5. 25. 1 简介	322
5. 25. 2 操作说明	323
5. 25. 3 寄存器列表	325
5. 25. 4 寄存器说明	326
5. 26 矩阵键盘 (Keypad)	328
5. 26. 1 操作说明	329
5. 26. 2 寄存器列表	330
5. 26. 3 寄存器说明	330
5. 27 GPIO 控制器	331
5. 27. 1 简介	331
5. 27. 2 操作说明	332
5. 27. 3 寄存器列表	332
5. 27. 4 寄存器说明	333
5. 28 PAD 复用	335
5. 28. 1 简介	335
5. 28. 2 操作说明	335
5. 28. 3 寄存器列表	336
5. 28. 4 寄存器说明	348
5. 29 RAS	349
5. 29. 1 错误的分类与上报	349
5. 29. 2 错误记录	350
5. 29. 3 寄存器说明	354
5. 30 MHU	360
5. 30. 1 操作说明	360
5. 30. 2 寄存器列表	361
5. 30. 3 寄存器说明	362
6 中断管理	365
6. 1 PPI 中断	365
6. 2 SPI 中断	365
7 功耗管理	371
7. 1 动态调频	371

7.2 温度控制	371
8 术语和缩略语	372

图 目 录

图 4-1	包含 2 个 FTC310 小核的 Cluster 架构图	8
图 4-2	包含 1 个 FTC664 大核的 Cluster 架构图	9
图 5-1	PCIe 树	16
图 5-2	DMA 直接传输模式	32
图 5-3	SG-DMA 整体结构图	33
图 5-4	SG_TYPE 00 类型传输模式	33
图 5-5	SG_TYPE 01 类型传输模式	34
图 5-6	SG_TYPE 10 类型传输模式	34
图 5-7	SG_TYPE 11 类型传输模式	35
图 5-8	以太网控制器初始化流程图	54
图 5-9	USB HUB 工作模式示意图	126
图 5-10	硬件信号量状态转换示意图	198
图 5-11	控制器两级时钟调频结构	222
图 5-12	时钟参数设置参考 1	223
图 5-13	时钟参数设置参考 2	223
图 5-14	时钟扩展	291
图 5-15	物理通道结构框图	360

表 目 录

表 2-1	飞腾派总体地址空间划分	2
表 2-2	飞腾派中断控制器内部地址划分	2
表 2-3	片内 SOC 设备地址空间划分	3
表 4-1	Memory 控制器基地址	10
表 4-2	Memory 控制器直接寄存器列表	11
表 5-1	PCIe 控制器与 PCIe 信号的对应关系	16
表 5-2	PCIe 地址空间划分	16
表 5-3	PCIe 配置空间寻址格式	17
表 5-4	PCIe 控制器寄存器基地址	17
表 5-5	PCIe 控制器寄存器列表	17
表 5-6	HPB 寄存器基地址	24
表 5-7	HPB 寄存器列表	24
表 5-8	DMA Engine 寄存器基地址	35
表 5-9	DMA Engine 寄存器列表	35
表 5-10	描述符配置寄存器列表	42
表 5-11	SSMU 中断分配	45
表 5-12	SMMU 寄存器基地址	46
表 5-13	SMMU 寄存器列表	46
表 5-14	以太网控制器寄存器基地址	57
表 5-15	以太网控制器寄存器列表	57
表 5-16	USB3.0 寄存器基地址	111
表 5-17	USB3 寄存器列表	111
表 5-18	USB2 寄存器基地址	115
表 5-19	USB2 寄存器列表	115
表 5-20	USB HUB 寄存器基地址	127
表 5-21	VHUB_CFG 寄存器列表	127

表 5-22	DEV (1~3)_CFG 寄存器列表	127
表 5-23	SATA 寄存器基地址	132
表 5-24	DC 寄存器基地址	138
表 5-25	DC 寄存器列表	138
表 5-26	分辨率与像素时钟频率对应关系表	142
表 5-27	DP 寄存器基地址	152
表 5-28	DP 寄存器列表	152
表 5-29	DP-PHY 寄存器基地址	173
表 5-30	DP-PHY 寄存器列表	173
表 5-31	VPU 寄存器基地址	177
表 5-32	VPU 寄存器列表	177
表 5-33	BDL 基本格式	179
表 5-34	GDMA 寄存器基地址	181
表 5-35	GDMA 寄存器列表	181
表 5-36	DDMA 通道 slave ID 分配表	190
表 5-37	DDMA 寄存器基地址	191
表 5-38	DDMA 寄存器列表	191
表 5-39	Semaphore 寄存器基地址	198
表 5-40	Semaphore 寄存器列表	198
表 5-41	WDT 寄存器基地址	201
表 5-42	WDT 寄存器列表	201
表 5-43	QSPI 错误信号	205
表 5-44	QSPI 寄存器基地址	205
表 5-45	QSPI 寄存器列表	205
表 5-46	SPI 寄存器基地址	212
表 5-47	SPI 寄存器列表	212
表 5-48	分频参数表	223
表 5-49	SD/SDIO/eMMC 寄存器基地址	228
表 5-50	SD/SDIO/eMMC 寄存器列表	228
表 5-51	ECC 编解码数据长度表	240
表 5-52	NAND Flash 寄存器基地址	241
表 5-53	NAND Flash 寄存器列表	241
表 5-54	I2S 寄存器基地址	254
表 5-55	I2S 寄存器列表	255

表 5-56	CAN 寄存器基地址	261
表 5-57	CAN 寄存器列表	261
表 5-58	JTAG Master 寄存器基地址	273
表 5-59	JTAG Master 寄存器列表	273
表 5-60	MIO 寄存器基地址	277
表 5-61	MIO 全局控制寄存器列表	277
表 5-62	UART 寄存器基地址	280
表 5-63	UART 寄存器列表	280
表 5-64	I2C 寄存器列表	292
表 5-65	SMBUS 寄存器基地址	308
表 5-66	SMBUS 寄存器列表	308
表 5-67	PWM 寄存器基地址	318
表 5-68	PWM 寄存器列表	319
表 5-69	Timer 寄存器基地址	325
表 5-70	Timer 寄存器列表	325
表 5-71	Keypad 寄存器基地址	330
表 5-72	Keypad 寄存器列表	330
表 5-73	GPIO 寄存器基地址	332
表 5-74	GPIO 寄存器列表	333
表 5-75	PAD 寄存器基地址	336
表 5-76	PAD 寄存器列表	336
表 5-77	ras_soc 错误信号统计表	350
表 5-78	ras_peu_psu 错误信号统计表	352
表 5-79	ras_peu 错误信号统计表	353
表 5-80	RAS 错误记录寄存器基地址	354
表 5-81	RAS 错误记录寄存器列表	354
表 5-82	错误类型编码	359
表 5-83	MHU 寄存器基地址	361
表 5-84	MHU 寄存器列表	361
表 6-1	PPI 中断 ID 分配表	365
表 6-2	SPI 中断 ID 分配表	366
表 8-1	术语和缩略语	372

1

概述

飞腾派是一款面向行业工程师、学生和爱好者的开源硬件，主板处理器采用飞腾嵌入式四核处理器。飞腾派详细技术指标与功能特性请参考《飞腾派数据手册》。

2

地址空间分配

飞腾派的物理地址空间宽度为 44bit，地址空间共计 16TB。

2.1 总体地址空间划分

表 2-1 飞腾派总体地址空间划分

地址空间	长度	描述
0x000_0000_0000 ~ 0x000_0FFF_FFFF	256MB	QSPI
0x000_1000_0000 ~ 0x000_1FFF_FFFF	256MB	LocalBus
0x000_2800_0000 ~ 0x000_2FFF_FFFF	128MB	低速设备
0x000_3000_0000 ~ 0x000_37FF_FFFF	128MB	其他部件空间，包含各子网络配置空间
0x000_3800_0000 ~ 0x000_3FFF_FFFF	128MB	IACC：指令控制空间
0x000_4000_0000 ~ 0x000_7FFF_FFFF	1GB	PCIe 配置、IO 和 MEM32 空间
0x000_8000_0000 ~ 0x000_FFFF_FFFF	2GB	Memory 空间
0x001_0000_0000 ~ 0x001_7FFF_FFFF	2GB	QSPI 高位地址空间
0x001_8000_0000 ~ 0x001_FFFF_FFFF	2GB	LocalBus 高位地址空间
0x002_4000_0000 ~ 0x00F_FFFF_FFFF	56GB	保留空间
0x010_0000_0000 ~ 0x01F_FFFF_FFFF	64GB	PCIe MEM64 空间
0x020_0000_0000 ~ 0xFFF_FFFF_FFFF	15TB768GB	扩展 Memory 空间

2.2 中断控制器内部地址划分

中断控制器的寄存器基地址为 0x3080_0000，各部分寄存器的偏移见下表。

表 2-2 飞腾派中断控制器内部地址划分

地址空间	长度	描述
0x00_0000 ~ 0x00_FFFF	64KB	Distributor 寄存器
0x01_0000 ~ 0x01_FFFF	64KB	基于消息的 SPI 中断 Distributor 寄存器
0x02_0000 ~ 0x02_FFFF	64KB	ITS 控制器寄存器
0x03_0000 ~ 0x03_FFFF	64KB	ITS 地址转换寄存器

地址空间	长度	描述
0x04_0000 ~ 0x07_FFFF	256KB	保留
0x08_0000 ~ 0x08_FFFF	64KB	RD0 控制与物理 LPI 寄存器
0x09_0000 ~ 0x09_FFFF	64KB	RD0 SGI 与 PPI 寄存器
0x0A_0000 ~ 0x0A_FFFF	64KB	RD1 控制与物理 LPI 寄存器
0x0B_0000 ~ 0x0B_FFFF	64KB	RD1 SGI 与 PPI 寄存器
0x0C_0000 ~ 0x0C_FFFF	64KB	RD2 控制与物理 LPI 寄存器
0x0D_0000 ~ 0x0D_FFFF	64KB	RD2 SGI 与 PPI 寄存器
0x0E_0000 ~ 0x0E_FFFF	64KB	RD3 控制与物理 LPI 寄存器
0x0F_0000 ~ 0x0F_FFFF	64KB	RD3 SGI 与 PPI 寄存器

2.3 片上设备地址空间划分

表 2-3 片内 SOC 设备地址空间划分

地址空间	长度	描述
0x000_2800_0000 ~ 0x000_2800_0FFF	4KB	MMCS0
0x000_2800_1000 ~ 0x000_2800_1FFF	4KB	MMCS1
0x000_2800_2000 ~ 0x000_2800_2FFF	4KB	NANDFlash
0x000_2800_3000 ~ 0x000_2800_3FFF	4KB	DMA0
0x000_2800_4000 ~ 0x000_2800_4FFF	4KB	DMA1
0x000_2800_5000 ~ 0x000_2800_5FFF	4KB	DDMA1
0x000_2800_8000 ~ 0x000_2800_8FFF	4KB	QSPI
0x000_2800_9000 ~ 0x000_2800_9FFF	4KB	I2S
0x000_2800_A000 ~ 0x000_2800_AFFF	4KB	CAN0
0x000_2800_B000 ~ 0x000_2800_BFFF	4KB	CAN1
0x000_2800_C000 ~ 0x000_2800_CFFF	4KB	UART0
0x000_2800_D000 ~ 0x000_2800_DFFF	4KB	UART1
0x000_2800_E000 ~ 0x000_2800_EFFF	4KB	UART2
0x000_2800_F000 ~ 0x000_2800_FFFF	4KB	UART3
0x000_2801_3000 ~ 0x000_2801_3FFF	4KB	SMBUS
0x000_2801_4000 ~ 0x000_2801_5FFF	8KB	MI00
0x000_2801_6000 ~ 0x000_2801_7FFF	8KB	MI01
0x000_2801_8000 ~ 0x000_2801_9FFF	8KB	MI02
0x000_2801_A000 ~ 0x000_2801_BFFF	8KB	MI03
0x000_2801_C000 ~ 0x000_2801_DFFF	8KB	MI04
0x000_2801_E000 ~ 0x000_2801_FFFF	8KB	MI05
0x000_2802_0000 ~ 0x000_2802_1FFF	8KB	MI06
0x000_2802_2000 ~ 0x000_2802_3FFF	8KB	MI07
0x000_2802_4000 ~ 0x000_2802_5FFF	8KB	MI08
0x000_2802_6000 ~ 0x000_2802_7FFF	8KB	MI09
0x000_2802_8000 ~ 0x000_2802_9FFF	8KB	MI010
0x000_2802_A000 ~ 0x000_2802_BFFF	8KB	MI011
0x000_2802_C000 ~ 0x000_2802_DFFF	8KB	MI012
0x000_2802_E000 ~ 0x000_2802_FFFF	8KB	MI013

地址空间	长度	描述
0x000_2803_0000 ~ 0x000_2803_1FFF	8KB	MI014
0x000_2803_2000 ~ 0x000_2803_3FFF	8KB	MI015
0x000_2803_4000 ~ 0x000_2803_4FFF	4KB	GP100
0x000_2803_5000 ~ 0x000_2803_5FFF	4KB	GP101
0x000_2803_6000 ~ 0x000_2803_6FFF	4KB	GP102
0x000_2803_7000 ~ 0x000_2803_7FFF	4KB	GP103
0x000_2803_8000 ~ 0x000_2803_8FFF	4KB	GP104
0x000_2803_9000 ~ 0x000_2803_9FFF	4KB	GP105
0x000_2803_A000 ~ 0x000_2803_AFFF	4KB	SPIM0
0x000_2803_B000 ~ 0x000_2803_BFFF	4KB	SPIM1
0x000_2803_C000 ~ 0x000_2803_CFFF	4KB	SPIM2
0x000_2803_D000 ~ 0x000_2803_DFFF	4KB	SPIM3
0x000_2804_0000 ~ 0x000_2804_1FFF	8KB	WDT0
0x000_2804_2000 ~ 0x000_2804_3FFF	8KB	WDT1
0x000_2804_4000 ~ 0x000_2804_4FFF	4KB	JTAG master
0x000_2804_A000 ~ 0x000_2804_AFFF	4KB	PWM0
0x000_2804_B000 ~ 0x000_2804_BFFF	4KB	PWM1
0x000_2804_C000 ~ 0x000_2804_CFFF	4KB	PWM2
0x000_2804_D000 ~ 0x000_2804_DFFF	4KB	PWM3
0x000_2804_E000 ~ 0x000_2804_EFFF	4KB	PWM4
0x000_2804_F000 ~ 0x000_2804_FFFF	4KB	PWM5
0x000_2805_0000 ~ 0x000_2805_0FFF	4KB	PWM6
0x000_2805_1000 ~ 0x000_2805_1FFF	4KB	PWM7
0x000_2805_4000 ~ 0x000_2805_4FFF	4KB	Tacho0
0x000_2805_5000 ~ 0x000_2805_5FFF	4KB	Tacho1
0x000_2805_6000 ~ 0x000_2805_6FFF	4KB	Tacho2
0x000_2805_7000 ~ 0x000_2805_7FFF	4KB	Tacho3
0x000_2805_8000 ~ 0x000_2805_8FFF	4KB	Tacho4
0x000_2805_9000 ~ 0x000_2805_9FFF	4KB	Tacho5
0x000_2805_A000 ~ 0x000_2805_AFFF	4KB	Tacho6
0x000_2805_B000 ~ 0x000_2805_BFFF	4KB	Tacho7
0x000_2805_C000 ~ 0x000_2805_CFFF	4KB	Tacho8
0x000_2805_D000 ~ 0x000_2805_DFFF	4KB	Tacho9
0x000_2805_E000 ~ 0x000_2805_EFFF	4KB	Tacho10
0x000_2805_F000 ~ 0x000_2805_FFFF	4KB	Tacho11
0x000_2806_0000 ~ 0x000_2806_0FFF	4KB	Tacho12
0x000_2806_1000 ~ 0x000_2806_1FFF	4KB	Tacho13
0x000_2806_2000 ~ 0x000_2806_2FFF	4KB	Tacho14
0x000_2806_3000 ~ 0x000_2806_3FFF	4KB	Tacho15
0x000_2806_4000 ~ 0x000_2806_4FFF	4KB	Tacho16
0x000_2806_5000 ~ 0x000_2806_5FFF	4KB	Tacho17
0x000_2806_6000 ~ 0x000_2806_6FFF	4KB	Tacho18
0x000_2806_7000 ~ 0x000_2806_7FFF	4KB	Tacho19

地址空间	长度	描述
0x000_2806_8000 ~ 0x000_2806_8FFF	4KB	Tacho20
0x000_2806_9000 ~ 0x000_2806_9FFF	4KB	Tacho21
0x000_2806_A000 ~ 0x000_2806_AFFF	4KB	Tacho22
0x000_2806_B000 ~ 0x000_2806_BFFF	4KB	Tacho23
0x000_2806_C000 ~ 0x000_2806_CFFF	4KB	Tacho24
0x000_2806_D000 ~ 0x000_2806_DFFF	4KB	Tacho25
0x000_2806_E000 ~ 0x000_2806_EFFF	4KB	Tacho26
0x000_2806_F000 ~ 0x000_2806_FFFF	4KB	Tacho27
0x000_2807_0000 ~ 0x000_2807_0FFF	4KB	Tacho28
0x000_2807_1000 ~ 0x000_2807_1FFF	4KB	Tacho29
0x000_2807_2000 ~ 0x000_2807_2FFF	4KB	Tacho30
0x000_2807_3000 ~ 0x000_2807_3FFF	4KB	Tacho31
0x000_2807_4000 ~ 0x000_2807_4FFF	4KB	Tacho32
0x000_2807_5000 ~ 0x000_2807_5FFF	4KB	Tacho33
0x000_2807_6000 ~ 0x000_2807_6FFF	4KB	Tacho34
0x000_2807_7000 ~ 0x000_2807_7FFF	4KB	Tacho35
0x000_2807_8000 ~ 0x000_2807_8FFF	4KB	Tacho36
0x000_2807_9000 ~ 0x000_2807_9FFF	4KB	Tacho37
0x000_2807_A000 ~ 0x000_2807_AFFF	4KB	Keypad
0x000_3180_0000 ~ 0x000_3187_FFFF	512KB	usb_vhub0
0x000_3188_0000 ~ 0x000_318F_FFFF	512KB	usb_vhub1
0x000_3190_0000 ~ 0x000_3197_FFFF	512KB	usb_vhub2
0x000_31A0_0000 ~ 0x000_31A1_FFFF	128KB	USB3_0
0x000_31A2_0000 ~ 0x000_31A3_FFFF	128KB	USB3_1
0x000_31A4_0000 ~ 0x000_31A4_1FFF	8KB	SATA0
0x000_3200_0000 ~ 0x000_3200_3FFF	16KB	DC
0x000_3200_4000 ~ 0x000_3200_4FFF	4KB	DP0
0x000_3200_5000 ~ 0x000_3200_5FFF	4KB	DP1
0x000_3200_8000 ~ 0x000_3200_8FFF	4KB	I2S0_DMA
0x000_3200_9000 ~ 0x000_3200_9FFF	4KB	I2S0
0x000_3200_A000 ~ 0x000_3200_AFFF	4KB	I2S1_DMA
0x000_3200_B000 ~ 0x000_3200_BFFF	4KB	I2S1
0x000_3200_C000 ~ 0x000_3200_DFFF	8KB	GMU0
0x000_3200_E000 ~ 0x000_3200_FFFF	8KB	GMU1
0x000_3201_0000 ~ 0x000_3201_1FFF	8KB	GMU2
0x000_3201_2000 ~ 0x000_3201_3FFF	8KB	GMU3
0x000_3201_4000 ~ 0x000_3201_5FFF	8KB	SATA1
0x000_3280_0000 ~ 0x000_3283_FFFF	256KB	USB2_0
0x000_3284_0000 ~ 0x000_3287_FFFF	256KB	USB2_1
0x000_32B3_4000 ~ 0x000_32B3_4FFF	4KB	DMAC0
0x000_32B3_6000 ~ 0x000_32B3_6FFF	4KB	Semaphore

3

计算单元

飞腾派集成了 2 种类型的自主研发 64 位通用处理核心，可以运行在 64 位与 32 位工作模式。FTC664 处理器核与 FTC310 处理器核均兼容 ARMv8.0 处理器架构。其中，FTC664 是高性能处理器核，采用乱序多发射超标量体系结构，FTC310 是低功耗处理器核，采用顺序架构。

飞腾派兼容 ARMv8-A 处理器架构，提供如下支持：

- 支持 AArch32 和 AArch64 两种运行模式；
- 在 AArch32 和 AArch64 两种运行模式下，均支持 ARMv8-A 体系结构定义的所有异常级，即 EL0、EL1、EL2、EL3；
- 在每个异常级都支持 Little-Endian 和 Big-Endian 两种字节序；
- 支持 A32 指令集；
- 支持 T32 指令集；
- 支持 A64 指令集；
- 提供 FPU 浮点计算单元；
- 提供 SIMD 向量处理单元；
- 支持非对齐地址访问。

3.1 浮点功能部件

浮点功能部件兼容 ARMv8-A 体系结构定义的寄存器和指令接口。请参见《ARM Architecture Reference Manual ARMv8 for ARMv8-A architecture profile》。

3.2 SIMD 功能部件

SIMD 功能部件兼容 ARMv8-A 体系结构定义的寄存器和指令接口。请参见《ARM

Architecture Reference Manual ARMv8 for ARMv8-A architecture profile》。

4 存储

4.1 Cache

飞腾派主板处理器系统中包含 3 个 Cluster，一个 Cluster 里包含 2 个小核 FTC310，另外两个 Cluster 各包含 1 个大核 FTC664。

对于包含 2 个 FTC310 小核的 Cluster，一级 Cache 为各个内核私有，每个内核分别有 32KB 的一级指令缓存（L1 Instruction Cache, L1I）和 32KB 的一级数据缓存（L1 Data Cache, L1D）；二级 Cache 为每个 Cluster 内 2 个 FTC310 内核共享，每个 Cluster 内共 256KB。包含 2 个 FTC310 小核的 Cluster 架构图如图 4-1 所示。

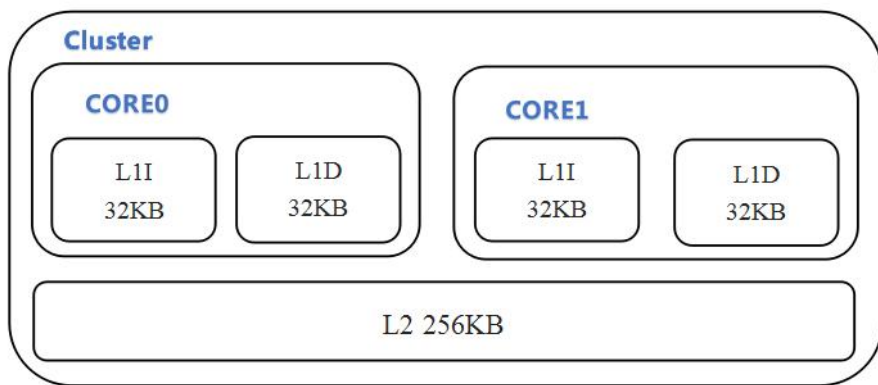


图 4-1 包含 2 个 FTC310 小核的 Cluster 架构图

对于包含 1 个 FTC664 大核的 Cluster，一级 Cache 与二级 Cache 为内核私有，分别有 48KB 的 L1 指令 Cache 和 32KB 的 L1 数据 Cache 以及 1MB 的 L2 混合 Cache。包含 1 个 FTC664 大核的 Cluster 架构图如图 4-2 所示。

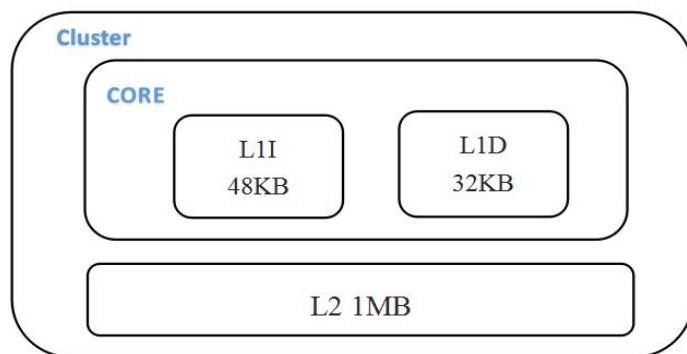


图 4-2 包含 1 个 FT0664 大核的 Cluster 架构图

4.1.1 L1 Cache

L1 Cache 分为指令 Cache 和数据 Cache。

小核 L1 指令 Cache 有如下特性：

- 容量为 32KB
- 64 字节定长 Cache line
- 物理地址寻址
- 非阻塞
- LRU 替换算法
- 支持奇偶校验
- 支持硬件预取

小核 L1 数据 Cache 有如下特性：

- 容量为 32KB
- 64 字节定长 Cache line
- 物理地址寻址
- 非阻塞
- LRU 替换算法
- 全局同步 monitor
- 奇偶校验，支持重新取数

大核 L1 指令 Cache 有如下特性：

- 容量为 48KB
- 64 字节定长 Cache line
- 物理地址寻址
- 非阻塞
- LRU 替换算法
- 支持奇偶校验

- 支持硬件预取
- 大核 L1 数据 Cache 有如下特性：
- 容量为 32KB
- 64 字节定长 Cache line
- 物理地址寻址
- 非阻塞
- LRU 替换算法
- 全局同步 monitor
- 奇偶校验，支持重新取数

4.1.2 L2 Cache

L2 存储系统的特征包括：

- 64 字节定长 Cache line 物理寻址及标识缓存
- 16 路组相连的 Cache 结构
- 支持多条流水线并行处理访存请求
- 支持 ECC 纠错
- 支持可选的硬件预取
- 支持软件可编程的 RAM 可变响应时间
- 支持奇偶校验

4.2 Memory

飞腾派集成的 DDR 控制器主要功能如下：

- 支持 DDR4、LPDDR4 协议
- 支持 BIST
- 支持多种低功耗模式
- 支持 R-DIMM、U-DIMM 和 SODIMM
- 支持纠一检二的 ECC 校验

4.2.1 寄存器列表

表 4-1 Memory 控制器基地址

名称	基地址
Memory 控制器	0x000_32B3_3000

写寄存器：

1. write(0x32B33000 + DDRC_PADDR, addr)
2. write(0x32B33000 + DDRC_PDATA, wdata)

读寄存器：

1. write(0x32B33000 + DDRC_PADDR, addr)
2. read(0x32B33000 + DDRC_PDATA)

其中, DDRC_PADDR 与 DDRC_PDATA 分别为 Memory 控制器地址寄存器与数据寄存器的偏移地址。addr 为需要写或读的寄存器的偏移地址。

表 4-2 Memory 控制器直接寄存器列表

寄存器名称	偏移	描述
DDRC_PADDR	0x0080	Memory 控制器地址寄存器
DDRC_PDATA	0x0084	Memory 控制器数据寄存器
ECC_ENABLE	0x07A0	ECC 使能寄存器
ECC_U_ADDR_L	0x07B8	ECC 不可纠地址低位寄存器
ECC_U_ADDR_H	0x07BC	ECC 不可纠地址高位寄存器
ECC_U_DATA_L	0x07C0	ECC 不可纠数据低位寄存器
ECC_U_DATA_H	0x07C4	ECC 不可纠数据高位寄存器
ECC_C_ADDR_L	0x07C8	ECC 可纠地址低位寄存器
ECC_C_ADDR_H	0x07CC	ECC 可纠地址高位寄存器
ECC_C_DATA_L	0x07D0	ECC 可纠数据低位寄存器
ECC_C_DATA_H	0x07D4	ECC 可纠数据高位寄存器
INT_STATUS_ECC	0x0944	ECC 状态寄存器
INT_ACK_ECC	0x0964	ECC 状态清除寄存器
INT_MASK_ECC	0x0984	ECC 状态掩码寄存器

4.2.2 寄存器说明

4.2.2.1 DDRC_PADDR (0x0080)

域	位	读写	复位值	描述
DDRC_PADDR	31:0	RW	0x0	Memory 控制器地址寄存器

4.2.2.2 DDRC_PDATA (0x0084)

域	位	读写	复位值	描述
DDRC_PDATA	31:0	RW	0x0	Memory 控制器数据寄存器

4.2.2.3 ECC_ENABLE (0x07A0)

域	位	读写	复位值	描述
Reserved	31:26	RO	0x0	保留位

域	位	读写	复位值	描述
ECC_ENABLE	25:24	RW	0x0	00: 不使能 ECC 01: 使能 ECC, 不检错, 不纠错 10: 使能 ECC, 检错, 不纠错 11: 使能 ECC, 检错, 纠错
Reserved	23:0	R0	0x0	保留位

4.2.2.4 ECC_U_ADDR_L (0x07B8)

域	位	读写	复位值	描述
ECC_U_ADDR_L	31:0	R0	0x0	ECC 不可纠错地址 [31:0] 部分。

4.2.2.5 ECC_U_ADDR_H (0x07BC)

域	位	读写	复位值	描述
Reserved	31:10	R0	0x0	保留位
ECC_U_ADDR_H	9:0	R0	0x0	ECC 不可纠错地址 [41:32] 部分。

4.2.2.6 ECC_U_DATA_L (0x07C0)

域	位	读写	复位值	描述
ECC_U_DATA_L	31:0	R0	0x0	ECC 不可纠错数据 [31:0] 部分。

4.2.2.7 ECC_U_DATA_H (0x07C4)

域	位	读写	复位值	描述
ECC_U_DATA_H	31:0	R0	0x0	ECC 不可纠错数据 [63:32] 部分。

4.2.2.8 ECC_C_ADDR_L (0x07C8)

域	位	读写	复位值	描述
ECC_C_ADDR_L	31:0	R0	0x0	ECC 可纠错地址 [31:0] 部分。

4.2.2.9 ECC_C_ADDR_H (0x07CC)

域	位	读写	复位值	描述
Reserved	31:10	R0	0x0	保留位
ECC_C_ADDR_H	9:0	R0	0x0	ECC 可纠错地址 [41:32] 部分。

4.2.2.10 ECC_C_DATA_L (0x07D0)

域	位	读写	复位值	描述
ECC_C_DATA_L	31:0	R0	0x0	ECC 可纠错数据 [31:0] 部分。

4.2.2.11 ECC_C_DATA_H(0x07D4)

域	位	读写	复位值	描述
ECC_C_DATA_H	31:0	RO	0x0	ECC 可纠错数据 [63:32] 部分。

4.2.2.12 INT_STATUS_ECC(0x0944)

域	位	读写	复位值	描述
INT_STATUS_ECC	15:0	RO	0x0	<p>ECC 相关中断状态。</p> <p>Bits [15:9]: 保留</p> <p>Bit [8]: 指示在 scrub 操作中有一个可纠正的 ECC 错误。</p> <p>Bit [7]: 指示 scrub 操作完成。</p> <p>Bit [6]: 指示一个或多个 ECC 写回命令不能执行。</p> <p>Bit [5]: 保留</p> <p>Bit [4]: 保留</p> <p>Bit [3]: 指示有多个不可纠的 ECC 错误发生。</p> <p>Bit [2]: 指示有一个不可纠的 ECC 错误发生。</p> <p>Bit [1]: 指示有多个可纠的 ECC 错误发生。</p> <p>Bit [0]: 指示有一个可纠的 ECC 错误发生。</p>

4.2.2.13 INT_ACK_ECC(0x0964)

域	位	读写	复位值	描述
INT_ACK_ECC	15:0	WO	0x0	<p>清除 INT_STATUS_ECC 参数中状态。</p> <p>Bits [15:9]: 保留</p> <p>Bit [8]: 在 scrub 操作中有一个可纠正的 ECC 错误清除位。写 1 清除。</p> <p>Bit [7]: scrub 操作完成清除位。写 1 清除。</p> <p>Bit [6]: 一个或多个 ECC 写回命令不能执行清除位。写 1 清除。</p> <p>Bit [5]: 保留</p> <p>Bit [4]: 保留</p> <p>Bit [3]: 有多个不可纠的 ECC 错误发生清除位。写 1 清除。</p> <p>Bit [2]: 有一个不可纠的 ECC 错误发生清除位。写 1 清除。</p> <p>Bit [1]: 有多个可纠的 ECC 错误发生清除位。写 1 清除。</p> <p>Bit [0]: 有一个可纠的 ECC 错误发生清除位。写 1 清除。</p>

4.2.2.14 INT_MASK_ECC (0x0984)

域	位	读写	复位值	描述
INT_MASK_ECC	15:0	RW	0x0	<p>屏蔽 ECC 相关的中断。如果以下的 bit 位置 1，则相关的中断将不会在 INT_STATUS_ECC 参数中置 1。</p> <p>Bits [15:9]：保留</p> <p>Bit [8]：在 scrub 操作中有一个可纠正的 ECC 错误屏蔽位。1 表示屏蔽。</p> <p>Bit [7]：scrub 操作完成屏蔽位。1 表示屏蔽。</p> <p>Bit [6]：一个或多个 ECC 写回命令不能执行屏蔽位。1 表示屏蔽。</p> <p>Bit [5]：保留</p> <p>Bit [4]：保留</p> <p>Bit [3]：有多个不可纠的 ECC 错误发生屏蔽位。1 表示屏蔽。</p> <p>Bit [2]：有一个不可纠的 ECC 错误发生屏蔽位。1 表示屏蔽。</p> <p>Bit [1]：有多个可纠的 ECC 错误发生屏蔽位。1 表示屏蔽。</p> <p>Bit [0]：有一个可纠的 ECC 错误发生屏蔽位。1 表示屏蔽。</p>

5

外设

5.1 PCIe 控制器

5.1.1 简介

飞腾派集成的 PCIe 控制器包含 PSU 模块的 2 个 PCIe 控制器与 PEU 模块的 4 个 PCIe 控制器，均支持 PCIe3.0 规范，兼容 PCIe2.0 及 PCIe1.0，支持如下特性：

- PSU 模块的 c0、c1 控制器在 x1 模式工作，和 SATA、USB 控制器进行 PHY 复用；
- PEU 模块的 PCIe 控制器实现了 x4 模式、x2 模式、x1 模式，支持：
 - c0 控制器在 x4 模式工作。lane 映射关系为：c0 对应 lane0-lane3；
 - c0 控制器在 x2 模式、c2 和 c3 控制器在 x1 模式下工作。lane 映射关系为：c0 对应 lane0-lane1，c2 对应 lane2，c3 对应 lane3，c1 不使用；
 - c0、c1、c2、c3 控制器在 x1 模式下工作。lane 映射关系为：c0 对应 lane0，c1 对应 lane1，c2 对应 lane2，c3 对应 lane3。
- 支持 EP 和 RC 模式切换；
- PSU 与 PEU 模块下的 c0 控制器实现了 2 个 DMA engine；
- 支持 SRIOV 功能；
- 支持单个控制器的关断和掉电，PSU 下的 c1 控制器不支持掉电。

5.1.2 树形拓扑

飞腾派的 PCIe 是树形结构，所有控制器都是 0 号总线下的设备。当访问的总线号为 0 时，表示是对控制器的访问。当总线号对应的控制器确实存在时，那么就转发为对控制器内配置寄存器的访问；不存在时，忽略写请求，读请求返回全 F。飞腾派 PCIe 树形结构视图见图 5-1。

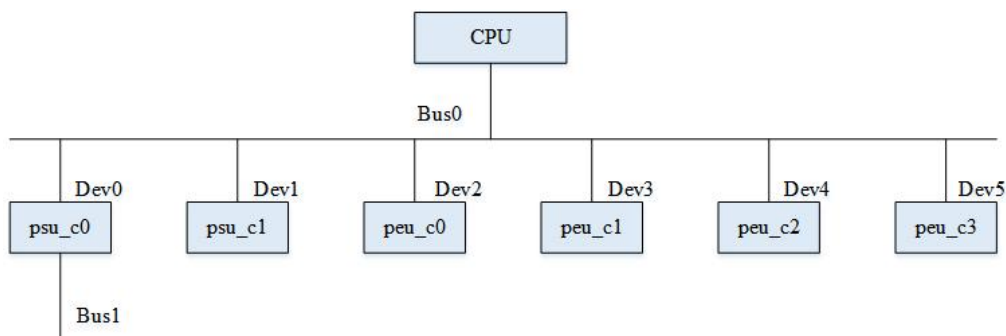


图 5-1 PCIe 树

PCIe 控制器与 PCIe 信号的对应关系如下表所示。

表 5-1 PCIe 控制器与 PCIe 信号的对应关系

PCIe 控制器名称	PCIe 信号名称
psu. c0	PCI E0
psu. c1	PCI E1
peu. c0	PCI E2 [0]
peu. c1	PCI E2 [1]
peu. c2	PCI E2 [2]
peu. c3	PCI E2 [3]

注意：PCIe 信号介绍详见飞腾派数据手册。

5.1.3 地址空间划分

表 5-2 PCIe 地址空间划分

地址范围	说明
0x000_3100_0000~0x000_316F_FFFF	7MB 片上寄存器空间
0x000_3100_0000~0x000_310F_FFFF	所有控制器内部寄存器，1MB
0x000_3110_0000~0x000_3110_0FFF	PSU 内 HPB 寄存器，4KB
0x000_3110_1000~0x000_3110_1FFF	PEU 内 HPB 寄存器，4KB
0x000_3130_0000~0x000_313F_FFFF	PEU 内 PHY 寄存器，1MB
0x000_3140_0000~0x000_3140_0FFF	PSU 内 RAS 寄存器，4KB
0x000_3140_1000~0x000_3140_1FFF	PEU 内 RAS 寄存器，4KB
0x000_3150_0000~0x000_315F_FFFF	网络，1MB
0x000_31B0_0000~0x000_31BF_FFFF	PSU 内 PHY0 寄存器，1MB
0x000_31C0_0000~0x000_31CF_FFFF	PSU 内 PHY1 寄存器，1MB
0x000_4000_0000~0x000_7FFF_FFFF	1GB 的 PCIe 配置、IO 和 MEM32 空间
0x010_0000_0000~0x01F_FFFF_FFFF	64GB 的 PCIe MEM64 空间

5.1.4 配置空间寻址

当地址位于 PCIe 配置空间时，将按照如下表格式进行解析。

表 5-3 PCIe 配置空间寻址格式

地址范围	含义
27:20	总线号, 0~255
19:15	设备号, 0~31
14:12	功能号, 0~7
11:0	4KB 配置空间内偏移

5.1.5 PCIe 控制器寄存器列表

表 5-4 PCIe 控制器寄存器基地址

名称	基地址
psu.c0 控制器寄存器	0x000_3100_0000
psu.c1 控制器寄存器	0x000_3102_0000
peu.c0 控制器寄存器	0x000_3104_0000
peu.c1 控制器寄存器	0x000_3106_0000
peu.c2 控制器寄存器	0x000_3108_0000
peu.c3 控制器寄存器	0x000_310A_0000

表 5-5 PCIe 控制器寄存器列表

寄存器	偏移	描述
LTSSM_ENABLE	0x084	ltssm 使能寄存器
PCIE_VC_CRED	0x090	PCIe 流控信用值设置寄存器
PCIE_BAR_01	0x0E4	当控制器为 EP 模式时, 用于设置 bar 空间的大小
PCIE_BAR_23	0x0EC	
PCIE_BAR_45	0x0F4	
PCIE_WINROM	0x0FC	PCIe IO/Memory 空间设置寄存器
PCIE_EQ_PRESET	0x100	PCIe eq preset 寄存器
IMASK_LOCAL	0x180	用于 RC 模式
ISTATUS_LOCAL	0x184	用于 RC 模式。当有相应的中断事件产生时, 此寄存器的相应位置 1; 各个中断源之间是独立的, 可以同时有多个中断位置 1; 写 1 清中断; 写 0 没影响。当有中断事件产生, 且 IMASK_LOCAL 没有屏蔽对应中断时, 将会通过内部信号(Axi 接口域)产生相应中断。
IMASK_HOST	0x188	用于 EP 模式
ISTATUS_HOST	0x18C	用于 EP 模式。当有相应中断事件产生时, 此寄存器相应位置 1; 各个中断源之间是独立的, 可以同时有多个中断位置 1; 写 1 清中断; 写 0 没影响。当有中断事件产生, 且 IMASK_HOST 没有屏蔽对应中断时, 将会通过 pcie 域产生相应的中断(当 msi 使能的时候, 发送 msi 中断; 否则发送 int 中断)。
PHYMAC_CFG	0x33C	链路均衡控制寄存器
PCIE_EQ_TUNING	0x35C	链路均衡参数设置寄存器
Capabilities Register	0x1080	(31:16) PCIe 能力寄存器

寄存器	偏移	描述
LTSSM_ENABLE	0x084	ltssm 使能寄存器
Next Cap PTR		(15:8) 下级 PCIe 能力指针
Capability ID		(7:0) PCIe 能力 ID
device capabilities	0x1084	PCIe 设备能力寄存器
device status	0x1088	PCIe 设备状态寄存器
device control		PCIe 设备控制寄存器
link capabilities	0x108C	PCIe 链接能力寄存器
link status	0x1090	(31:16) PCIe 链接状态寄存器
link control		(15:0) PCIe 链接控制寄存器
slot capabilities	0x1094	PCIe 槽位能力寄存器
slot status	0x1098	(31:16) PCIe 槽位状态寄存器
slot control		(15:0) PCIe 槽位控制寄存器
root capabilities	0x109C	(31:16) PCIe RC 能力寄存器
root control		(15:0) PCIe RC 控制寄存器
root status	0x10A0	PCIe RC 状态寄存器
device capabilities 2	0x10A4	PCIe 设备能力寄存器 2
device status 2	0x10A8	(31:16) PCIe 设备状态寄存器 2
device control 2		(15:0) PCIe 设备控制寄存器 2
link capabilities 2	0x10AC	PCIe 链接能力寄存器 2
link status 2	0x10B0	(31:16) PCIe 链接状态寄存器 2
link control 2		(15:0) PCIe 链接控制寄存器 2
slot capabilities 2	0x10B4	PCIe 槽位能力寄存器 2
slot status 2	0x10B8	(31:16) PCIe 槽位状态寄存器 2
slot control 2		(15:0) PCIe 槽位控制寄存器 2
PCI Express Enhanced Capability Header	0x1200	PCIe 增强能力报头
Uncorrectable Error Status Register	0x1204	不可纠正错误状态寄存器
Uncorrectable Error Mask Register	0x1208	不可纠正错误掩码寄存器
Uncorrectable Error Severity Register	0x120C	不可纠正错误严重性寄存器
Correctable Error Status Register	0x1210	可纠正错误状态寄存器
Correctable Error Mask Register	0x1214	可纠正错误掩码寄存器
Advanced Error Capabilities and Control Register	0x1218	高级错误能力和控制寄存器
Header Log Register	0x121C	报头记录寄存器
Root Error Command	0x122C	PCIe RC 错误命令寄存器
Root Error Status	0x1230	PCIe RC 错误状态寄存器

5.1.6 PCIe 控制器寄存器说明

5.1.6.1 LTSSM_ENABLE (0x84)

域	位	读写	复位值	描述
reserved	31:3	RW	0x0	保留
ltssm_enable	2	RW	0x0	该位在正常情况下释放，拉高的时

域	位	读写	复位值	描述
				候 会 阻 止 LTSSM 退 出 Detect. Quite 状态。 0: 使能 ltssm 1: 关闭 ltssm
reserved	1:0	RW	0x0	保留

5.1.6.2 PCIE_VC_CRED (0x90)

PCIE_VC_CRED 寄存器为 64 位寄存器。

域	位	读写	复位值	描述
reserved	63:62	RW	0x0	保留
Completion credit scale	61:60	RW	0x0	2'b00: x1, 2'b01: x4, 2'b10: x16, 2'b11: reserved
Non-Posted credit scale	59:58	RW	0x0	2'b00: x1, 2'b01: x4, 2'b10: x16, 2'b11: reserved
Posted credit scale	57:56	RW	0x0	2'b00: x1, 2'b01: x4, 2'b10: x16, 2'b11: reserved
Completion data credits	55:44	RW	0x0	信用值, 最大 0x7FF
Completion header credits	43:36	RW	0x0	信用值, 最大 0x7F
Non-Posted data credits	35:28	RW	0x0	信用值, 最大 0x7F
Non-Posted header credits	27:20	RW	0x0	信用值, 最大 0x7F
Posted data credits.	19:8	RW	0x0	信用值, 最大 0x7FF
Posted header credits	7:0	RW	0x0	信用值, 最大 0x7F

5.1.6.3 PCIE_BAR_01 (0xE4)

PCIE_BAR_01 寄存器为 64 位寄存器。bar 类型为 memory:

域	位	读写	复位值	描述
Bar size mask	63:32	RW	0x0	如果是 64 位地址(bit[2]=1), bar1 和 bar0 是相同的配置
Bar size mask	31:4	RW	0x0	bar 大小屏蔽
prefetchable	3	RW	0x0	预取
64-bit address space	2	RW	0x0	64 位地址空间 0: 32 位地址 1: 64 位地址
reserved	1	RW	0x0	保留
BAR type	0	RW	0x0	bar 类型, 0: memory

bar 类型为 IO:

域	位	读写	复位值	描述
reserved	63:32	RW	0x0	保留
Bar size mask	31:2	RW	0x0	bar 大小屏蔽
reserved	1	RW	0x0	保留
BAR type	0	RW	0x0	bar 类型, 1: IO

5.1.6.4 PCIE_BAR_23 (0xEC)

PCIE_BAR_23 寄存器为 64 位寄存器。bar 类型为 memory:

域	位	读写	复位值	描述
Bar size mask	63:32	RW	0x0	如果是 64 位地址(bit[2]=1), bar2 和 bar3 是相同的配置
Bar size mask	31:4	RW	0x0	bar 大小屏蔽
prefetchable	3	RW	0x0	预取
64-bit address space	2	RW	0x0	64 位地址空间 0: 32 位地址 1: 64 位地址
reserved	1	RW	0x0	保留
BAR type	0	RW	0x0	bar 类型, 0: memory

bar 类型为 IO:

域	位	读写	复位值	描述
reserved	63:32	RW	0x0	保留
Bar size mask	31:2	RW	0x0	bar 大小屏蔽
reserved	1	RW	0x0	保留
BAR type	0	RW	0x0	bar 类型, 1: IO

5.1.6.5 PCIE_BAR_45 (0xF4)

PCIE_BAR_45 寄存器为 64 位寄存器。bar 类型为 memory:

域	位	读写	复位值	描述
Bar size mask	63:32	RW	0x0	如果是 64 位地址(bit[2]=1), bar4 和 bar5 是相同的配置
Bar size mask	31:4	RW	0x0	bar 大小屏蔽
prefetchable	3	RW	0x0	预取
64-bit address space	2	RW	0x0	64 位地址空间 0: 32 位地址 1: 64 位地址
reserved	1	RW	0x0	保留
BAR type	0	RW	0x0	bar 类型, 0: memory

bar 类型为 IO:

域	位	读写	复位值	描述
reserved	63:32	RW	0x0	保留
Bar size mask	31:2	RW	0x0	bar 大小屏蔽
reserved	1	RW	0x0	保留
BAR type	0	RW	0x0	bar 类型, 1: IO

5.1.6.6 PCIE_WINROM (0xFC)

域	位	读写	复位值	描述
Expansion ROM size mask	31:11	RW	0x0	当控制器为 EP

域	位	读写	复位值	描述
reserved	10:4	RW	0x0	模 式 时 , bit[3:0] 为保 留位; 当控制器为 RC 模 式 时 , bit[31:11] 为 保留位。
Prefetchable memory window 64-bit addressing support	3	RW	0x0	
Prefetchable memory window implemented	2	RW	0x0	
IO window 32-bit addressing support	1	RW	0x0	
IO window implemented	0	RW	0x0	

5.1.6.7 PCIE_EQ_PRESET (0x100)

PCIE_EQ_PRESET 寄存器为 256 位寄存器。

域	位	读写	复位值	描述
reserved	255	R0	0x0	lan 15 255:240 当控制器为 EP 模式时, 只 有 此 时 处 于 loopback master 情况时, 这些值被 发送给 link partner。 bit[255:240] 位 于 地 址 0x011E- 0x011F。
Link partner's receiver preset hint	254:252	R0	0x0	
Link partner's transmitter preset	2511:248	R0	0x0	
reserved	247	R0	0x0	
default receiver preset hint	246:244	R0	0x0	
default transmitter preset	243:240	R0	0x0	
...
reserved	31	R0	0x0	lan 1 31:16 当控制器为 EP 模式时, 只 有 此 时 处 于 loopback master 情况时, 这些值被 发送给 link partner。
Link partner's receiver preset hint	30:28	R0	0x0	
Link partner's transmitter preset	27:24	R0	0x0	
reserved	23	R0	0x0	
default receiver preset hint	22:20	R0	0x0	
default transmitter preset	19:16	R0	0x0	
reserved	15	R0	0x0	lan 0 15:0 当控制器为 EP 模式时, 只 有 此 时 处 于 loopback master 情况时, 这些值被 发送给 link partner。 bit[15:0] 位 于 地 址 0x0100- 0x0101。
Link partner's receiver preset hint	14:12	R0	0x0	
Link partner's transmitter preset	11:8	R0	0x0	
reserved	7	R0	0x0	
default receiver preset hint	6:4	R0	0x0	
default transmitter preset	3:0	R0	0x0	

5.1.6.8 IMASK_LOCAL (0x180)

域	位	读写	复位值	描述
Local Processor Interrupt Mask	31:0	RW	0x0	每一位对应一个中断源; 置 1 使能中断, 置 0 屏蔽中断; 详细中断对应位请参考 ISTATUS_LOCAL 寄存器。

5.1.6.9 ISTATUS_LOCAL (0x184)

域	位	读写	复位值	描述
System error	31	RW1C	0x0	RC only, reserved for EP
PM/LTR/Hotplug event	30	RW1C	0x0	PM/LTR/Hotplug event for Rootport, Legacy power management state change for Endpoint.
AER Event	29	RW1C	0x0	RC only, reserved for EP
MSI received	28	RW1C	0x0	RC only, reserved for EP
interrupt line D	27	RW1C	0x0	RC only, reserved for EP
interrupt line C	26	RW1C	0x0	RC only, reserved for EP
interrupt line B	25	RW1C	0x0	RC only, reserved for EP
interrupt line A	24	RW1C	0x0	RC only, reserved for EP
PCIe Doorbell	23	RW1C	0x0	pcie 请求成功命中地址转换表
PCIe Discard Error	22	RW1C	0x0	pcie read 超时
PCIe Fetch Error	21	RW1C	0x0	pcie read 报错
PCIe Post Error	20	RW1C	0x0	pcie write 报错
AXI Doorbell	19	RW1C	0x0	axi 请求成功命中地址转换表
AXI Discard Error	18	RW1C	0x0	axi read 超时
AXI Fetch Error	17	RW1C	0x0	axi read 报错
AXI Post Error	16	RW1C	0x0	axi write 报错
dma_error	15:8	RW1C	0x0	每一位对应一个 DMA Engine, dma 传输报错。bit 位置 1 表示传输报错。
dma_end	7:0	RW1C	0x0	每一位对应一个 DMA Engine, dma 传输结束。bit 位置 1 表示传输结束。

5.1.6.10 IMASK_HOST (0x188)

域	位	读写	复位值	描述
Host Processor Interrupt Mask	31:0	RW	0x0	每一位对应一个中断源；置 1 使能中断，置 0 屏蔽中断；详细中断对应位请参考 ISTATUS_HOST 寄存器。

5.1.6.11 ISTATUS_HOST (0x18C)

域	位	读写	复位值	描述
interrupt	31:24	RW1C	0x0	ep 产生相应的中断信号
PCIe Doorbell	23	RW1C	0x0	pcie 请求成功命中地址转换表
PCIe Discard Error	22	RW1C	0x0	pcie read 超时
PCIe Fetch Error	21	RW1C	0x0	pcie read 报错
PCIe Post Error	20	RW1C	0x0	pcie write 报错
AXI Doorbell	19	RW1C	0x0	axi 请求成功命中地址转换表
AXI Discard Error	18	RW1C	0x0	axi read 超时
AXI Fetch Error	17	RW1C	0x0	axi read 报错

域	位	读写	复位值	描述
AXI Post Error	16	RW1C	0x0	axi write 报错
dma_error	15:8	RW1C	0x0	每一位对应一个 DMA Engine, dma 传输报错。bit 位置 1 表示传输报错。
dma_end	7:0	RW1C	0x0	每一位对应一个 DMA Engine, dma 传输结束。bit 位置 1 表示传输结束。

5.1.6.12 PHYMAC_CFG (0x33C)

域	位	读写	复位值	描述
reserved	31:2	RW	0x0	保留
RXELECIDLE	1	RW	0x0	置 1, 使用 RXELECIDLE 检测 electrical idle entry。
phases 2/3 equalization	0	RW	0x0	RC 模式下: 置 1, 执行 EQ 的 phase 2/3; 置 0, 不执行 EQ 的 phase 2/3。 EP 模式下: 置 1, 使对端的 RC 在 phase2 阶段执行调整; 置 0, 不执行。

5.1.6.13 PCIE_EQ_TUNING (0x35C)

PCIE_EQ_TUNING 寄存器为 256 位寄存器。

域	位	读写	复位值	描述
reserved	95:34	RW	0x0	保留
Maximum number of fine-tuning iterations	33:28	RW	0x0	Gen3 速率下, fine-tuning 最大迭代次数: 0: 不使用 fine-tune。 1-63: 当 phy 不支持 direction-change 模式下的信号评估反馈时, 这个值必须为 0; 否则为 Fine-tune coefficients 的执行次数。
reserved	27	RW	0x0	保留
preset(s)	26:16	RW	0x0	Gen3 速率下, preset 值: bit[16]: Preset #0 bit[26]: Preset #10
reserved	15:2	RW	0x0	保留
behavior	1:0	RW	0x0	fine-tuning 过程检测到错误时的操作: 2'b00: 使用上一次成功的 coefficients 值, 继续进行。 2'b01: 采用上一次成功的 coefficients 值, 停止这个过程。 2'b10: 采用目前为止最佳的 preset 值, 继续进行。 2'b11: 采用目前为止最佳的 preset 值, 停止这个过程。

5.1.7 HPB 寄存器列表

表 5-6 HPB 寄存器基地址

名称	基地址
psu. HPB 寄存器	0x000_3110_0000
peu. HPB 寄存器	0x000_3110_1000

表 5-7 HPB 寄存器列表

寄存器	偏移	描述
REG_INTERRUPT_STATUS	0x000	中断状态寄存器
REG_INTERRUPT_ENABLE	0x004	INTX 中断使能寄存器
REG_ERR_EVT_STATUS	0x0C0	错误状态寄存器
REG_MSI_EN	0x200	MSI 使能寄存器
REG_MSI64_HI_ADDR	0x208	MSI64 高位地址设置寄存器
REG_MSI64_LO_ADDR	0x20C	MSI64 低位地址设置寄存器
REG_Cx_TEST_IN_ERRINJ	0x308+0x10*x	Cx 错误注入寄存器
REG_Cx_LTSSM	0x540+4*x	链路状态寄存器
REG_VGA_EN	0x700	VGA 使能寄存器
REG_PIO_RE_LIMIT	0x720	限制 pio 读请求数目寄存器
REG_PIO_WR_LIMIT	0x724	限制 pio 写请求数目寄存器
REG_PIO_RD_INTERVAL	0x728	设置 pio 读请求间隔周期寄存器
REG_PIO_WR_INTERVAL	0x72C	设置 pio 写请求间隔周期寄存器
REG_BIF_MODE	0x800	PEU 拆分模式设置寄存器
REG_SPEED_WIDTH	0x83C	控制器链路宽度和速度设置寄存器
REG_CLK_OK	0xA08	时钟状态寄存器
REG_Cx_PREF_BASE_LIMIT	0xA40+0x10*x	设置控制器 EP 模式 pio 请求译码可预存储空间上限与基址低位寄存器
REG_Cx_PREF_BASE_UP32	0xA44+0x10*x	设置控制器 EP 模式 pio 请求译码可预存储空间上限与基址高位寄存器
REG_Cx_MEM_BASE_LIMIT	0xA48+0x10*x	设置控制器 EP 模式 pio 请求译码 mem 空间上限与基址低位寄存器

5.1.8 HPB 寄存器说明

5.1.8.1 REG_INTERRUPT_STATUS (0x000)

peu 内中断状态寄存器：

域	位	读写	复位值	描述
reserved	31:16	R0	0x0	保留
c3_inta	15	R0	0x0	peu. c3 控制器 INTA 中断状态
c3_intb	14	R0	0x0	peu. c3 控制器 INTB 中断状态
c3_intc	13	R0	0x0	peu. c3 控制器 INTC 中断状态
c3_intd	12	R0	0x0	peu. c3 控制器 INTD 中断状态

域	位	读写	复位值	描述
c2_inta	11	R0	0x0	peu. c2 控制器 INTA 中断状态
c2_intb	10	R0	0x0	peu. c2 控制器 INTB 中断状态
c2_intc	9	R0	0x0	peu. c2 控制器 INTC 中断状态
c2_intd	8	R0	0x0	peu. c2 控制器 INTD 中断状态
c1_inta	7	R0	0x0	peu. c1 控制器 INTA 中断状态
c1_intb	6	R0	0x0	peu. c1 控制器 INTB 中断状态
c1_intc	5	R0	0x0	peu. c1 控制器 INTC 中断状态
c1_intd	4	R0	0x0	peu. c1 控制器 INTD 中断状态
c0_inta	3	R0	0x0	peu. c0 控制器 INTA 中断状态
c0_intb	2	R0	0x0	peu. c0 控制器 INTB 中断状态
c0_intc	1	R0	0x0	peu. c0 控制器 INTC 中断状态
c0_intd	0	R0	0x0	peu. c0 控制器 INTD 中断状态

psu 内中断状态寄存器：

域	位	读写	复位值	描述
reserved	31:8	R0	0x0	保留
c1_inta	7	R0	0x0	psu. c1 控制器 INTA 中断状态
c1_intb	6	R0	0x0	psu. c1 控制器 INTB 中断状态
c1_intc	5	R0	0x0	psu. c1 控制器 INTC 中断状态
c1_intd	4	R0	0x0	psu. c1 控制器 INTD 中断状态
c0_inta	3	R0	0x0	psu. c0 控制器 INTA 中断状态
c0_intb	2	R0	0x0	psu. c0 控制器 INTB 中断状态
c0_intc	1	R0	0x0	psu. c0 控制器 INTC 中断状态
c0_intd	0	R0	0x0	psu. c0 控制器 INTD 中断状态

5.1.8.2 REG_INTERRUPT_ENABLE (0x004)

域	位	读写	复位值	描述
reserved	31:4	RW	0x0	保留
intd_en	3	RW	0x0	INTD 中断使能
intc_en	2	RW	0x0	INTC 中断使能
intb_en	1	RW	0x0	INTB 中断使能
inta_en	0	RW	0x0	INTA 中断使能

5.1.8.3 REG_ERR_EVT_STATUS (0x0C0)

域	位	读写	复位值	描述
err_evt_status_reg	31:0	RW	0x0	bit[0]:pio 读地址错 bit[1]:pio 读超时 bit[2]:pio 写地址错 bit[3]:pio 写超时 bit[4]:读响应错 bit[5]:写响应错 其它位为保留位

5.1.8.4 REG_MSI_EN (0x200)

peu 内 MSI 中断使能寄存器：

域	位	读写	复位值	描述
reserved	31:4	RW	0x0	保留
msi_en	3:0	RW	0x0	MSI 中断使能

psu 内 MSI 中断使能寄存器：

域	位	读写	复位值	描述
reserved	31:2	RW	0x0	保留
msi_en	1:0	RW	0x0	msi 中断使能

5.1.8.5 REG_MSI64_HI_ADDR (0x208)

域	位	读写	复位值	描述
msi64_hi_addr	31:0	RW	0x0	MSI64 高位地址

5.1.8.6 REG_MSI64_LO_ADDR (0x20C)

域	位	读写	复位值	描述
msi64_lo_addr	31:16	RW	0x0	MSI64 低位地址
reserved	15:0	RW	0x0	保留

5.1.8.7 REG_Cx_TEST_IN_ERRINJ (0x308+0x10*x)

peu 内寄存器 REG_Cx_TEST_IN_ERRINJ 的 x 取值范围为 0~3。

psu 内寄存器 REG_Cx_TEST_IN_ERRINJ 的 x 取值范围为 0~1。

域	位	读写	复位值	描述
cx_test_in_errinj	31:0	RW	0x0	<p>bit[0]: 造成一次接收器错误（通过假装好像一次 RX 状态上报并在 Lane #0 检测到译码错误）。</p> <p>bit[1]: 触发一次接收器上溢错误（接收器上溢并没有真实发生）。</p> <p>bit[2]: 造成一次突发的断开错误（通过直接控制 LTSSM 造成非预期的 HotReset）。</p> <p>bit[3]: 造成一次坏的 TLP 报文错误（通过在下一个接收到的 TLP 报文的 LCRC 中翻转一个 bit）。</p> <p>bit[4]: 造成一次坏的 DLLP 报文错误（通过在下一个接收到的 DLLP 报文的 DCRC 中翻转一个 bit）。</p> <p>bit[5]: 造成一次重复的 NUM 翻转（通过设置 replay_num=3 并且在假装好像在一个未响应的 TLP 报文发送时收到 NAK）。</p> <p>bit[6]: 造成一次重复的超时（一旦一次未响应的 TLP 发送就使 replay timeout 过期）。</p> <p>bit[7]: 造成一次数据链路协议错误（通过修改下一次接收到的 ACK 中的 sequence# 以致检测到一个 ACK/NAK 错误）。</p>

域	位	读写	复位值	描述
				bit[8]: 造成一次流控协议错误（通过损坏下一次接收到的 UpdateFC DLLP 的 Header FC 导致其比告知的信用数值更大）。 bit[15:9]: 预留。 bit[16]: 将下一次接收到的 TLP 报文头中的 EP bit 置 1。 bit[17]: 造成一次非预期的完成错误（通过将下一次接收到的 Cpl/CpID TLP 报文的 Bus# 设置为 FFh）。 bit[18]: 造成一次缺陷（Malformed）TLP 报文错误（通过翻转下一次收到的 TLP 报文的 TD bit 位）。 bit[19]: 造成一次 ECRC 错误（通过翻转洗一次收到的 TLP 报文头的 Tag9 bit 位）。 bit[20]: 造成一次不支持的请求错误： • 对于 DSP: 在下次接收到的 MRd TLP 替换 TLP 报文类型为 MRdLk。 • 对于 USP: 在下次接收到的 Cpl/CpID TLP 替换 TLP 报文类型为 CplLk/CpldLk。 bit[31:21]: 保留。

5.1.8.8 REG_Cx_LTSSM (0x540+4*x)

peu 内寄存器 REG_Cx_LTSSM 的 x 取值范围为 0~3。

psu 内寄存器 REG_Cx_LTSSM 的 x 取值范围为 0~1。

域	位	读写	复位值	描述
reserved	31:8	RW	0x0	保留
cx_link_state_reg	7:6	RW	0x0	当链路状态机处于 Recovery. Equalization 时，链路均衡的状态： 2'b00: phase 0 2'b01: phase 1 2'b10: phase2 2'b11: phase 3
cx_ltssm_state_reg	5:0	RW	0x0	链路训练状态机 ltssm: • 00h: detect. quiet • 01h: detect. active • 02h: polling. active • 03h: polling. compliance • 04h: polling. configuration • 05h: config. linkwidthstart • 06h: config. linkwidthaccept • 07h: config. lanenumwait • 08h: config. lanenumaccept • 09h: config. complete • 0Ah: config. idle • 0Bh: recovery. receiver lock • 0Ch: recovery. equalization

域	位	读写	复位值	描述
				<ul style="list-style-type: none"> • 0Dh: recovery. speed • 0Eh: recovery. receiverconfig • 0Fh: recovery. idle • 10h: L0 • 11h: L0s • 12h: L1. entry • 13h: L1. idle • 14h: L2. idle/L2. transmitwake • 15h: reserved • 16h: disable • 17h: loopback. entry • 18h: loopback. active • 19h: loopback. exit • 1Ah: hotreset

5.1.8.9 REG_VGA_EN (0x700)

peu 内 VGA 使能寄存器:

域	位	读写	复位值	描述
reserved	31:4	RW	0x0	保留
vga_en	3:0	RW	0x0	bit[3]: peu. c3 控制器 vga 使能 bit[2]: peu. c2 控制器 vga 使能 bit[1]: peu. c1 控制器 vga 使能 bit[0]: peu. c0 控制器 vga 使能

psu 内 VGA 使能寄存器:

域	位	读写	复位值	描述
reserved	31:2	RW	0x0	保留
vga_en	1:0	RW	0x0	bit[1]: peu_spu. c1 控制器 vga 使能 bit[0]: peu_spu. c0 控制器 vga 使能

5.1.8.10 REG_PIO_RE_LIMIT (0x720)

域	位	读写	复位值	描述
reserved	31:5	RW	0x0	保留
pio_rd_limit	4:0	RW	0x10	限制 pio 读请求的数目, 1~16

5.1.8.11 REG_PIO_WR_LIMIT (0x724)

域	位	读写	复位值	描述
reserved	31:5	RW	0x0	保留
pio_wr_limit	4:0	RW	0x10	限制 pio 写请求的数目, 1~16

5.1.8.12 REG_PIO_RD_INTERVAL (0x728)

域	位	读写	复位值	描述
reserved	31:16	RW	0x0	保留
pio_rd_interval	15:0	RW	0x0	设置 pio 读请求的间隔周期

5.1.8.13 REG_PIO_WR_INTERVAL (0x72C)

域	位	读写	复位值	描述
reserved	31:16	RW	0x0	保留
pio_wr_interval	15:0	RW	0x0	设置 pio 写请求的间隔周期

5.1.8.14 REG_BIF_MODE (0x800)

域	位	读写	复位值	描述
reserved	31:2	RW	0x0	保留
bif_mode	1:0	RW	0x0	PEU 拆分模式 2'b00: x4 2'b01: x1 x1 x2 2'b10: x1 x1 x1 x1 2'b11: 保留

5.1.8.15 REG_SPEED_WIDTH (0x83C)

peu 内控制器链路宽度和速度设置寄存器:

域	位	读写	复位值	描述
reserved	31:30	RO	0x0	保留
c3_link_speed	29:28	RO	0x0	c3 的链路速度 2'b01: 2.5G 2'b10: 5G 2'b11: 8G
reserved	27:26	RW	0x0	保留
c3_link_width	25:24	RO	0x0	c2 的链路宽度 2'b10: x4 2'b01: x2 2'b00: x1
reserved	23:22	RW	0x0	保留
c2_link_speed	21:20	RO	0x0	c2 的链路速度 2'b01: 2.5G 2'b10: 5G 2'b11: 8G
reserved	19:18	RW	0x0	保留
c2_link_width	17:16	RO	0x0	c2 的链路宽度

域	位	读写	复位值	描述
				2'b10: x4 2'b01: x2 2'b00: x1
reserved	15:14	RW	0x0	保留
c1_link_speed	13:12	RO	0x0	c1 的链路速度 2'b01: 2.5G 2'b10: 5G 2'b11: 8G
reserved	11:10	RW	0x0	保留
c1_link_width	9:8	RO	0x0	c1 的链路宽度 2'b10: x4 2'b01: x2 2'b00: x1
reserved	7:6	RW	0x0	保留
c0_link_speed	5:4	RO	0x0	c0 的链路速度 2'b01: 2.5G 2'b10: 5G 2'b11: 8G
reserved	3:2	RW	0x0	保留
c0_link_width	1:0	RO	0x0	c0 的链路宽度 2'b10: x4 2'b01: x2 2'b00: x1

psu 内控制器链路宽度和速度设置寄存器:

域	位	读写	复位值	描述
reserved	31:14	RO	0x0	保留
c1_link_speed	13:12	RO	0x0	c1 的链路速度 2'b01: 2.5G 2'b10: 5G 2'b11: 8G
reserved	11:10	RW	0x0	保留
c1_link_width	9:8	RO	0x0	c1 的链路宽度 2'b10: x4 2'b01: x2 2'b00: x1
reserved	7:6	RW	0x0	保留
c0_link_speed	5:4	RO	0x0	c0 的链路速度 2'b01: 2.5G 2'b10: 5G 2'b11: 8G
reserved	3:2	RW	0x0	保留
c0_link_width	1:0	RO	0x0	c0 的链路宽度 2'b10: x4

域	位	读写	复位值	描述
				2'b01: x2 2'b00: x1

5.1.8.16 REG_CLK_OK (0xA08)

域	位	读写	复位值	描述
c3_pipe_clk_ok_reg	7	RO	0x0	为 1 时表示有对应的时钟
c3_core_clk_ok_reg	6	RO	0x0	为 1 时表示有对应的时钟
c2_pipe_clk_ok_reg	5	RO	0x0	为 1 时表示有对应的时钟
c2_core_clk_ok_reg	4	RO	0x0	为 1 时表示有对应的时钟
c1_pipe_clk_ok_reg	3	RO	0x0	为 1 时表示有对应的时钟
c1_core_clk_ok_reg	2	RO	0x0	为 1 时表示有对应的时钟
c0_pipe_clk_ok_reg	1	RO	0x0	为 1 时表示有对应的时钟
c0_core_clk_ok_reg	0	RO	0x0	为 1 时表示有对应的时钟

5.1.8.17 REG_Cx_PREF_BASE_LIMIT (0xA40+0x10*x)

peu 内寄存器 REG_Cx_PREF_BASE_LIMIT 的 x 取值范围为 0~3。

psu 内寄存器 REG_Cx_PREF_BASE_LIMIT 的 x 取值范围为 0~1。

域	位	读写	复位值	描述
cx_ep_pref_limit	31:20	RW	0x0	设置 cx EP 模式 pio 请求译码, 可预存储空间上限低位
reserved	19:16	RW	0x0	保留
cx_ep_pref_base	15:4	RW	0x0	设置 cx EP 模式 pio 请求译码, 可预存储空间基址低位
reserved	3:0	RW	0x0	保留

5.1.8.18 REG_Cx_PREF_BASE_UP32 (0xA44+0x10*x)

peu 内寄存器 REG_Cx_PREF_BASE_UP32 的 x 取值范围为 0~3。

psu 内寄存器 REG_Cx_PREF_BASE_UP32 的 x 取值范围为 0~1。

域	位	读写	复位值	描述
reserved	31:16	RW	0x0	保留
cx_ep_pref_limit_up32	15:8	RW	0x0	设置 cx EP 模式 pio 请求译码, 可预存储空间上限高位
cx_ep_pref_base_up32	7:0	RW	0x0	设置 cx EP 模式 pio 请求译码, 可预存储空间基址高位

5.1.8.19 REG_Cx_MEM_BASE_LIMIT (0xA48+0x10*x)

peu 内寄存器 REG_Cx_MEM_BASE_LIMIT 的 x 取值范围为 0~3。

psu 内寄存器 REG_Cx_MEM_BASE_LIMIT 的 x 取值范围为 0~1。

域	位	读写	复位值	描述
cx_ep_mem_limit	31:20	RW	0x0	设置 cxEP 模式 pio 请求译码, mem 空间上限低位
reserved	19:16	RW	0x0	保留
cx_ep_mem_base	15:4	RW	0x0	设置 cxEP 模式 pio 请求译码, mem 空间基址低位
reserved	3:0	RW	0x0	保留

5.1.9 DMA 引擎技术

PCIe 核心实现了两个完全独立的 DMA 引擎模块,可以编程使用 DMA 直接传输或分散式 DMA (SG-DMA) 传输。每个 DMA 引擎,既支持从主存到 EP 设备搬运数据,也支持从 EP 设备到主存搬运数据;支持任意长度的数据传输 (DMA 直接传输模式最大 4GB; SG DMA 模式每个描述符最大 16MB)。

PCIe DMA 引擎可通过实现以下功能实现最大吞吐量:

- 可配的 outstanding 读请求数 (最多 128 个)
- 可配置的缓冲区大小
- 可配置描述符获取
- 完成重新排序以实现最大吞吐量

DMA 直接传输模式下, DMA 的起始地址指向地址空间的一块连续的数据缓冲区,传输开始后, DMA 引擎顺序的将数据从传输源端,写到传输目标端;单次 DMA 传输,可以支持最大 4GB 的数据传输。



图 5-2 DMA 直接传输模式

PCIE SG-DMA 引擎还提供了高级功能,以实现最大的灵活性:

- 四种分散收集类型,例如单独的源 SG 和目标 SG 链表,可轻松实现位于不同域中的两个主机之间的接口。

- 动态 SG-DMA 配置，例如每个特定 SG 的中断生成或开始和结束条件。
- 增强了主机和网桥 DMA 引擎之间的握手机制，例如描述符中的动态可选报告。

SG-DMA 的整体结构如下图所示，下面的介绍都基于 SG-DMA 展开。在分散收集传输模式下，DMA 源和/或目标起始地址是指向页面描述符链表的指针。每个描述符包含数据块（页面）的地址和大小，以及指向下一个描述符块的指针，以启用循环缓冲区。

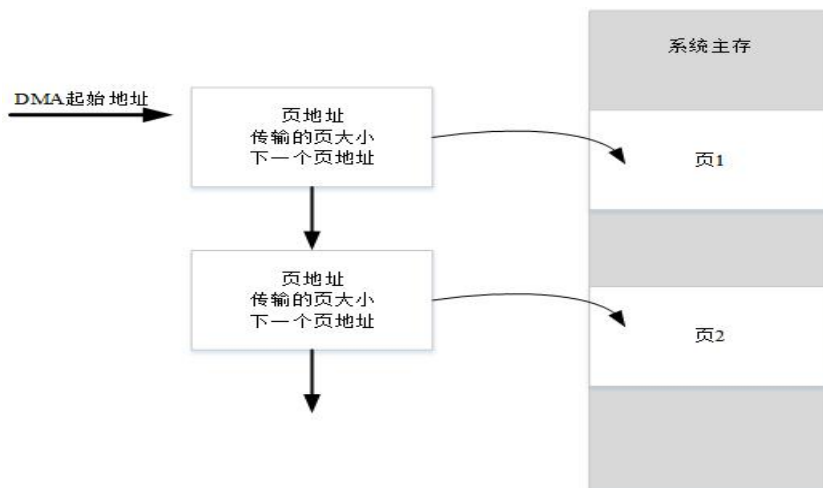


图 5-3 SG-DMA 整体结构图

可以通过配置 DMA_CONTROL 寄存器的[25:24]位，来设置当前 DMA 传输所采用的 SG_TYPE 类型；控制器支持四种 SG_TYPE 类型，每一种类型的传输模式如下所示。

SG_TYPE 00：传输源端和传输目标端的描述符是相互独立的；在这种模式下，传输源端描述符里面的 DESC_DEST_ADDR 字段以及传输目标端描述符里面的 DESC_SRC_ADDR 字段为保留字段。

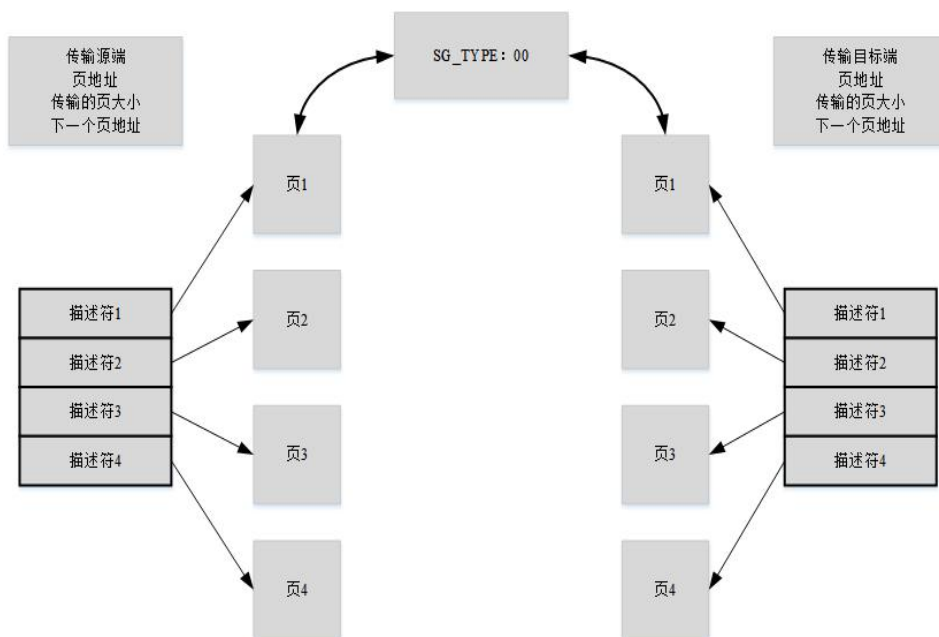


图 5-4 SG_TYPE 00 类型传输模式

SG_TYPE 01: 传输源端的地址是根据描述符来设置的, 传输目标端的地址是递增的; 在这种模式下, 传输源端描述符里面的 DESC_DEST_ADDR 字段为保留字段。

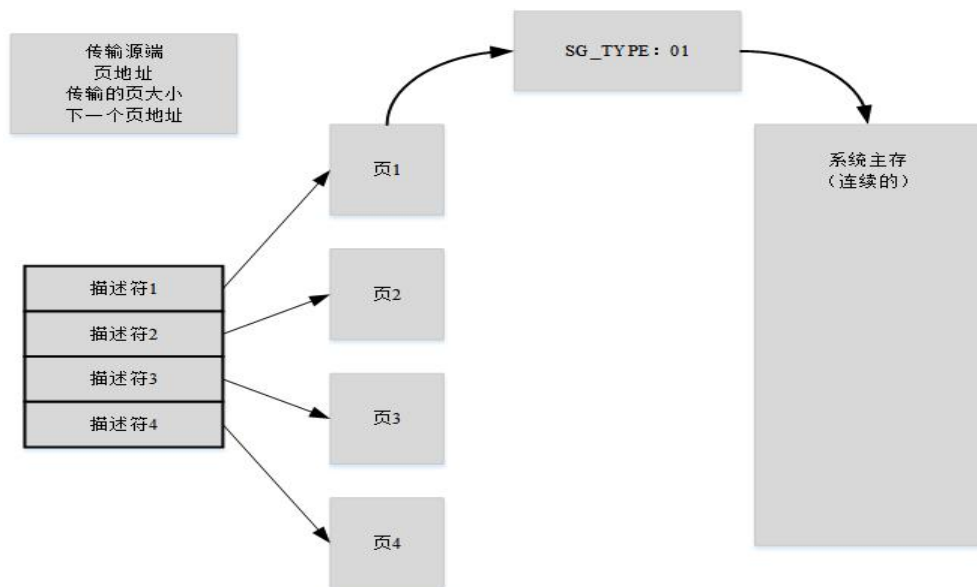


图 5-5 SG_TYPE 01 类型传输模式

SG_TYPE 10: 传输源端的地址是递增的, 传输目标端的地址是根据描述符来设置的; 在此模式下, 传输目标端描述符里面的 DESC_SRC_ADDR 字段为保留字段。

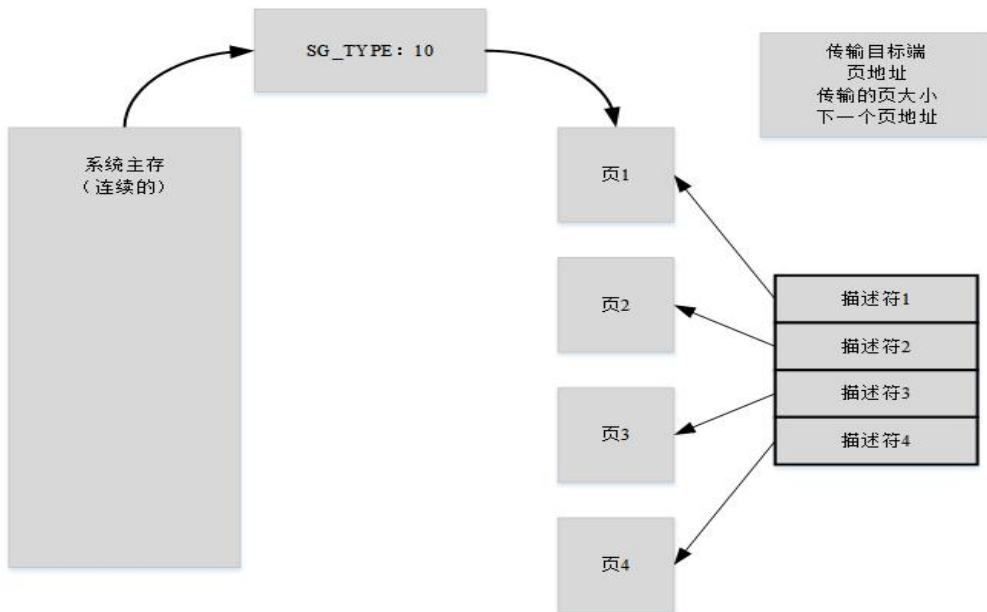


图 5-6 SG_TYPE 10 类型传输模式

SG_TYPE 11: 传输源端和传输目标端的地址是根据传输源端的描述符来设置的。

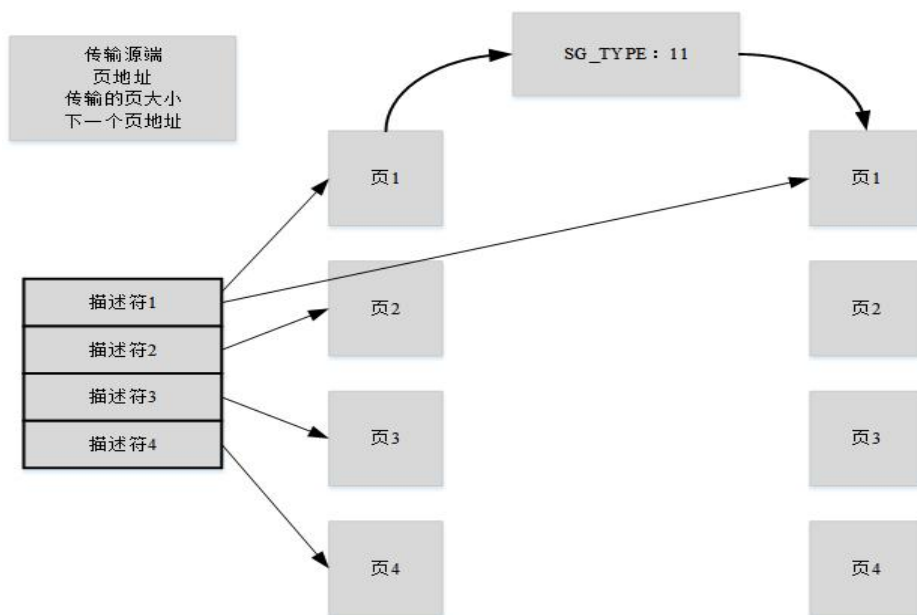


图 5-7 SG_TYPE 11 类型传输模式

5.1.10 DMA Engine 寄存器列表

psu.c0 控制器与 peu.c0 控制器均包含 2 个 DMA 引擎。

表 5-8 DMA Engine 寄存器基地址

名称	基地址
DMA_Engine0 寄存器	c0 控制器寄存器基地址+0x400
DMA_Engine1 寄存器	c0 控制器寄存器基地址+0x440

表 5-9 DMA Engine 寄存器列表

寄存器	偏移	描述
DMA_SRCPARAM	0x000	DMA 传输的源接口参数
DMA_DESTPARAM	0x004	DMA 传输的目标接口参数
DMA_SRCADDR[31:0]	0x008	这些寄存器提供 DMA 传输的起始源地址和目标地址（如果有）。如果实际的源地址和目标地址小于 64 位，则忽略 MSB 位。
DMA_SRCADDR[63:32]	0x00C	
DMA_DESTADDR[31:0]	0x010	
DMA_DESTADDR[63:32]	0x014	<ul style="list-style-type: none"> 当源启用 SG 模式时，DMA_SRCADDR 提供第一个源描述符的地址。 当目标启用 SG 模式时，DMA_DESTADDR 提供第一个 Destination 描述符的地址。 请注意，当 SG 类型字段等于 2'b11 时，DMA_DESTADDR 是无关的，因为它与 DMA_SRCADDR 相同。
DMA_LENGTH	0x018	DMA_LENGTH 寄存器提供应从源传输到目标的数据量（以字节为单位）（最大 4GB）。如果 SE_COND 字段的位 1 被清除，则该值仅是指示性的，将用于计算 DMA_STATUS 寄存器的 STATUS 字段的位 0。
DMA_CONTROL	0x01C	DMA 控制寄存器
DMA_STATUS	0x020	DMA 状态寄存器
DMA_PRC_LENGTH	0x024	DMA 传输数据量寄存器
DMA_SHARE_ACCESS	0x028	DMA 访问控制寄存器
DMA_PASID	0x02C	DMA PASID 寄存器

5.1.11 DMA Engine 寄存器说明

5.1.11.1 DMA_SRCPARAM (0x000)

域	位	读写	复位值	描述
reserved	31:28	RW	0x0	保留
trsf_param	27:16	RW	0x0	<p>提供传输参数，内容取决于 SRC_ID 的值。</p> <ul style="list-style-type: none"> 源接口为 PCIe 接口 <p>bit[18:16]: TLP 类型</p> <p>3'b000: Memory</p> <p>3'b001: Memory Locked</p> <p>3'b011: Translation Request</p> <p>3'b101: Memory Translated Request</p> <p>3'b010: IO</p> <p>3'b100: Message</p> <p>其他值保留。</p> <p>bit[19]: 报文请求 No Write (NW) 标志 (当 TLP 的 type 字段不是 3'b011 时, bit[19] 必须为 0)。</p> <p>bit[22:20]: TLP 属性</p> <p>bit 20: NoSnoop</p> <p>bit 21: Relaxed Ordering</p> <p>bit 22: ID-Bade Ordering</p> <p>bit[23]: ECRC Forward</p> <p>bit[26:24]: Traffic Class</p> <p>bit[27]: 保留</p> <ul style="list-style-type: none"> 源接口为 AXI4-Master 接口 <p>bit[19:16]: ACACHE</p> <p>bit[20]: ALOCK</p> <p>bit[23:21]: APROT</p> <p>bit[27:24]: AQOS</p> <ul style="list-style-type: none"> 源接口为 AXI4-Stream 接口 <p>bit[23:16]: TID</p> <p>bit[27:24]: TDEST</p>
reserved	15:4	RW	0x0	保留
src_id	3:0	RW	0x0	<p>定义 DMA 传输的源接口 ID:</p> <p>0x8-0xB: AXI4-Stream 接口编号 0-3</p> <p>0x4-0x7: AXI4-Master 接口编号 0-3</p> <p>0x0: PCIe 接口</p> <p>当这些寄存器字段未通过内核常量进行硬连线时, 它们将被读/写。</p>

5.1.11.2 DMA_DESTPARAM (0x004)

域	位	读写	复位值	描述
reserved	31:28	RW	0x0	保留
trsf_param	27:16	RW	0x0	<p>提供传输参数，其内容取决于 DEST_ID 的值。</p> <ul style="list-style-type: none"> 目标接口为 PCIe 接口 <p>bit[18:16]: TLP 类型</p> <p>3'b000: Memory</p> <p>3'b001: Memory Locked</p> <p>3'b011: Translation Request</p> <p>3'b101: Memory Translated Request</p> <p>3'b010: IO</p> <p>3'b100: Message</p> <p>其他值保留。</p> <p>bit[19]: 报文请求 No Write (NW) 标志 (当 TLP 的 type 字段不是 3'b011 时, bit[19]必须为 0)</p> <p>bit[22:20]: TLP 属性</p> <p>bit 20: NoSnoop</p> <p>bit 21: Relaxed Ordering</p> <p>bit 22: ID-Bade Ordering</p> <p>bit[23]: ECRC Forward</p> <p>bit[26:24]: Traffic Class</p> <p>bit[27]: 保留</p> <ul style="list-style-type: none"> 目标接口为 AXI4-Master 接口 <p>bit[19:16]: ACACHE</p> <p>bit[20]: ALOCK</p> <p>bit[23:21]: APROT</p> <p>bit[27:24]: AQOS</p> <ul style="list-style-type: none"> 目标接口为 AXI4-Stream 接口 <p>bit[23:16]: TID</p> <p>bit[27:24]: TDEST</p>
reserved	15:4	RW	0x0	保留
dest_id	3:0	RW	0x0	<p>定义目标接口 ID 的 DMA 传输:</p> <p>4'h8-4'hB: AXI4-Stream 接口编号 0-3</p> <p>4'h4-4'h7: AXI4-Master 接口编号 0-3</p> <p>4'h0: PCIe 接口</p> <p>当这些寄存器字段未通过内核常量进行硬连线时, 它们将被读/写。</p>

5.1.11.3 DMA_SRCADDR[31:0] (0x008)

域	位	读写	复位值	描述
source address	31:0	RW	0x0	源地址[31:0]

5.1.11.4 DMA_SRCADDR[63:32] (0x00C)

域	位	读写	复位值	描述
source address	31:0	RW	0x0	源地址[63:32]

5.1.11.5 DMA_DESTADDR[31:0] (0x010)

域	位	读写	复位值	描述
destination address	31:0	RW	0x0	目标地址[31:0]

5.1.11.6 DMA_DESTADDR[63:32] (0x014)

域	位	读写	复位值	描述
destination address	31:0	RW	0x0	目标地址[63:32]

5.1.11.7 DMA_LENGTH (0x018)

域	位	读写	复位值	描述
dma length	31:0	RW	0x0	DMA 长度

5.1.11.8 DMA_CONTROL (0x01C)

域	位	读写	复位值	描述
sg2_id	31:29	RW	0x0	与 SG_ID 相同。 请注意，如果 SG_TYPE 设置为 2'b00，则 SG_ID 连接到源，而 SG2_ID 连接到目标。否则，SG_ID 连接到源和/或目标，而 SG2_ID 不相关。 当 SG_ID 和 SG2_ID 寄存器未由 Core 硬连线为常量，它们是可读写的。
sg_id	28:26	RW	0x0	定义应在哪个接口上读取描述符： • 3'h0: PCIe 接口 • 3'h3: AXI4-主描述符接口 • 3'h4-4'h7: AXI4-主接口编号 0-3
sg_type	25:24	RW	0x0	为以下项定义散布收集类型 DMA（仅在设置了 CTRL 域的第 3 位时才相关）。 • 2'b00: 源和目标均独立的 SG。 • 2'b01: 根据描述符设置源地址，目标地址递增。 • 2'b10: 根据描述符设置目标地址，源地址递增。 • 2'b11: 源地址和目标地址是根据描述符设置的。 当该寄存器字段未通过内核常量进行硬连

域	位	读写	复位值	描述
				线时，将对其进行读/写。
desc_updt	23	RW	0x0	由应用程序设置为 1，以指示 DMA 引擎描述符已更新。仅在启用 SG 模式且 DESC_NEXT_RDY 字段设置为 0 时才有意义（请参阅 DESC_NEXT_RDY）。该位由 DMA 引擎自动清除。
reserved	22:14	RW	0x0	保留
irq_id	13:12	RW	0x0	<p>定义应在哪个接口上报告 DMA 传输事件：</p> <ul style="list-style-type: none"> • bit[12]：向本地处理器（在 AXI 上）发出中断域）。 • bit[13]：向主机处理器（在 PCIe 域上）发出中断。 <p>注意，这两个位不能同时设置为“1”。要在两侧产生中断，必须在 AXI 侧启用中断，并使用 LOCAL_INTERRUPT_IN 0 在 PCIe 侧产生中断。</p> <p>如果发生 DMA 传输事件，则将其报告给 ISTATUS_HOST 和/或 ISTATUS_LOCAL 寄存器（请参见 ISTATUS 寄存器），具体取决于 IRQ_ID 的内容。</p>
irq	11:8	RW	0x0	<p>定义何时发出中断：</p> <ul style="list-style-type: none"> • bit[8]：在 DMA 端发出 IRQ。 • bit[9]：如果发生错误，则发出 IRQ。 • bit[10]：如果传输源报告 EOP 条件，则发出 IRQ。 • bit[11]：保留。 <p>请注意，如果 IRQ 设置为 0，则不会发出中断。</p>
se_cond	7:4	RW	0x0	<p>定义开始和结束 DMA 条件：</p> <ul style="list-style-type: none"> • bit[4]：保留。 • bit[5]：如果达到 DMA_LENGTH，则停止。 • bit[6]：如果 DMA 的源是 AXI-Stream 接口，则在传输的源报告 EOP 条件（接收到 TLAST）的情况下，停止 DMA。 <p>如果 DMA 目标是 AXI-Stream 接口，请在 DMA 的末尾生成 EOP（TLAST 生成）。</p> <p>注意：如果此位等于 0，则内核将每 PCIe maxpayload 字节生成一个 TLAST，以防止可能的 AXI 互连仲裁问题。</p> <ul style="list-style-type: none"> • bit[7]：在错误情况下中止（否则，错误的数据包或描述符被视为已处理，但错误记录在源和/或目标错误字段中）。 <p>请注意，如果 bit[7:5] SE_COND 字段为 3'b000，则仅当用户中止或启用 SG 模式并接收到链末时，DMA 传输才能停止。</p>

域	位	读写	复位值	描述
ctrl	3:0	RW	0x0	提供 DMA 的基本控制： <ul style="list-style-type: none">• bit[0]：启动/中止：设置为 1b 时，它将启动 DMA 传输；之前应该已经设置了适当的寄存器。在 DMA 传输结束时，DMA 引擎自动清除该位。如果传输没有结束并且用户将其设置为 0，则传输中止。• bit[1]：暂停/恢复：设置为 1b 时，DMA 传输被暂停（为更高优先级的传输临时提供更多带宽）。• bit[2]：保留。• bit[3]：启用 SG 模式。当该寄存器字段未通过内核常量进行硬连线时，将对其进行读/写。

5.1.11.9 DMA_STATUS (0x020)

域	位	读写	复位值	描述
desc_error	31:24	RO	0x0	读取时检测到错误描述符： <ul style="list-style-type: none">• bit[24]：源描述符错误。• bit[25]：目标描述符错误。• bit[26]：由于完成超时，描述符读取失败。• bit[27]：由于打开，描述符读取由于接收到 UR 而失败，PCIe 域，如果在 AXI 域上则接收到 DECERR。• bit[28]：由于在 PCIe 域上收到 UR 或 EP，或者在 AXI 域上收到 SLVERR 响应，描述符读取失败。• bit[29]：如果在 PCIe 域，PCIe 控制器或网桥内存错误上，由于接收到 ECRC，描述符读取失败；或 AXI 应用程序报告的描述符错误（如果在 AXI 域上）。• bit[31:30]：保留。
dest_error	23:16	RO	0x0	读取时检测到错误。 目标数据（与 SRC_ERROR 相同的映射）。
src_error	15:8	RO	0x0	读取源时检测到错误数据或描述符： <ul style="list-style-type: none">• bit[8]：由于完成超时，数据读取失败。• bit[9]：由于在 PCIe 域上接收到 CA，在 AXI 域上接收到 DECERR，导致数据读取失败。• bit[10]：由于在 PCIe 域上收到 UR 或 EP，在 AXI 域上收到 SLVERR 响应，导致数据读取失败。• bit[11]：由于在 PCIe 域，PCIe 控制器或网桥内存错误上收到 ECRC，数据读取失

域	位	读写	复位值	描述
				<p>败。或如果在 AXI 域上，则 AXI 应用程序报告的数据错误。</p> <ul style="list-style-type: none"> • bit[15:12]：保留 <p>当控制字段的位 0 设置为 1b 时，将自动清除这些字段。</p>
status	7:0	RO	0x0	<p>状态：</p> <ul style="list-style-type: none"> • bit[0]：已达到 DMA_LENGTH 的 DMA 完成。当用 DMA_LENGTH 等于 0（无限 DMA 传输）编程 DMA 时，DMA_STATUS [0] 不相关。 • bit[1]：DMA 已完成，传输源报告了 EOP 条件。 • bit[2]：DMA 完成，在 DMA 引擎读取的最后一个描述符上收到 EOC。 • bit[3]：DMA 完成，但有错误。 • bit[4]：DMA 完成，传输的数据超过 4GBytes。 • bit[5]：保留。 • bit[6]：用户成功停止 DMA。 • bit[7]：DMA 错误结束（缓冲区或描述符未释放）。 <p>请注意，如果 DMA 因错误而结束，则错误状态字段将不是 8'b0。当 DMA_CONTROL[0] 设置为 1b 时，此字段将自动清除。</p>

5.1.11.10 DMA_PRC_LENGTH (0x024)

域	位	读写	复位值	描述
dma_prc_length	31:0	RO	0x0	<p>此 32 位寄存器提供了从源传输到目标的数据量（以字节为单位）。仅在清除状态字段的位 4 时才有意义。</p>

5.1.11.11 DMA_SHARE_ACCESS (0x028)

域	位	读写	复位值	描述
dma_share_access	31:0	RW/RO	0x0	<p>bit[0]：R / W：DMA 访问已锁定：设置为 1 时，对 DMA 引擎寄存器的写访问仅限于由 bit[10:4] 标识的物理或虚拟功能。否则，所有功能都被允许写访问。</p> <ul style="list-style-type: none"> • bit[1]：RO：授予 DMA 访问权限：当由 bit[10:4] 标识的物理或虚拟功能读取或将 DMA 访问锁定设置为 0 时，返回 1。否则，返回 0。 • bit[3:2]：RO：保留。 • bit[12:4]：RO：虚拟功能编号（1-511）：

域	位	读写	复位值	描述
				如果为 0，则以物理功能为目标。仅当实现了虚拟功能时，这些位才可用。 <ul style="list-style-type: none">• bit[17:13]: R0: 物理功能编号 (0-31)。仅当实现了多个物理功能时，这些位才可用。• bit[31:18]: R0: 保留。 注意: bit[17:4] 由网桥根据相关功能自动设置，并在清除 bit[0] 时清除。

5.1.11.12 DMA_PASID (0x02C)

域	位	读写	复位值	描述
dma_pasid	31:0	RW/R0	0x0	<ul style="list-style-type: none">• 启用位 bit[0] PASID• 请求的位 bit[1] 执行• 请求的位 bit[2] 特权模式• bit[22:3] 进程地址空间 ID (PASID)• bit[31:23] 保留

5.1.12 描述符配置寄存器列表

当使用 Scatter-Gather DMA 时，需对描述符进行配置，每个描述符的地址取决于在 DMA Engine 里面的配置。

表 5-10 描述符配置寄存器列表

寄存器	偏移	描述
DESC_STATUS	0x00	描述符状态寄存器
DESC_CONTROL	0x04	描述符控制寄存器
DESC_NEXT_ADDR[31:0]	0x08	下一个描述符地址
DESC_NEXT_ADDR[63:32]	0x0C	
DESC_SRC_ADDR[31:0]	0x10	源地址
DESC_SRC_ADDR[63:32]	0x14	
DESC_DEST_ADDR[31:0]	0x18	目标地址
DESC_DEST_ADDR[63:32]	0x1C	

5.1.13 描述符配置寄存器说明

5.1.13.1 DESC_STATUS (0x00)

域	位	读写	复位值	描述
desc_prc_page_size	31:8	R0	0x0	提供已处理页面大小，它是实际的写或读的页面大小。如果描述符处理由于错误或 EOP 检测而缩短(参见 SE_COND 描述)，则它与 PAGE_SIZE 不同。
desc_prc_status	7:4	R0	0x0	bit[4]: SG_DMA 已经被处理

域	位	读写	复位值	描述
				bit[5]: 在处理当前 SG_DMA 描述符时发生一个错误 bit[6]: SG_DMA 传输源报告一个 EOP (结束包) 条件 bit[7]: 保留 注意: 建议应用程序清除 desc_status_num, 以便确定何时通过轮询 bit0 来确定更新 STATUS。
desc_status_num	3:0	RO	0x0	提供一个 DMA 引擎的状态号。状态号递增, 能够使应用程序确定最终需要处理的描述符, 这在异步设备之间的流中是关键。 注意: DESC_STATUS 地址偏移量等于 0x00-0x03, 以允许 DMA 引擎在不需要的情况下不读取这个 Dword, 已优化吞吐量。

5.1.13.2 DESC_CONTROL (0x04)

域	位	读写	复位值	描述
desc_page_size	31:8	RW	0x0	提供以字节为单位的页面大小。取值范围为 1-16Mbytes。如果设置为 24'h0, 则为 16Mbytes。
desc_irq	7:4	RW	0x0	定义什么时候发出中断: bit[7]: 保留 bit[6]: 如果传输源报告的一个 EOP 条件, 就会发出 irq bit[5]: 如果错误发生, 就会发出 irq bit[4]: 当 SG_DMA 描述符被处理时, 会发出 irq
reserved	3:1	RW	0x0	保留
desc_stats_req	0	RW	0x0	定义了当前的 SG_DMA 描述符被处理时, DMA 引擎是否通过写入 DESC_STATUS 来提供一个状态报告。这使得应用程序能够轻松地监视 DMA 进程, 而无需轮询 DMA 引擎寄存器。

5.1.13.3 DESC_NEXT_ADDR[31:0] (0x08)

域	位	读写	复位值	描述
DESC_NEXT_ADDR [31:5]	31:5	RW	0x0	下一个描述符地址。该地址必须在 32 字节的边界上对齐。
DESC_NEXT_RDY	4	RW	0x0	表示下一个 SG-DMA 描述符是否准备好 (并且可获取)。应用程序可以将其设置为 0, 表示链表没有结束, 但是稍后会完成。
DESC_SE_COND	3:0	RW	0x0	DESC_SE_COND 为 SG-DMA 描述符处理的结束和开始条件。它由以下字段组成: bit3: 当传输源报告了 SOP 接收时 (在 TLAST 断言之后的数据上, 仅当源是一个 AXI4-stream 接

域	位	读写	复位值	描述
				<p>口才相关)，启动这个 SG-DMA 描述符处理。</p> <p>bit[2]: 如果 SG-DMA 的目标是一个 AXI4-stream 接口，且 SG-type 为 01，则在 SG-DMA 描述符处理之后发出一个 EOP。DMA_CONTROL 参数的 SE_COND[2] 必须被断言以启动该特性。如果 SG-DMA 的目标是一个 AXI4-stream 接口，且 SG-type 为 10，如果传输源报告 EOP 条件，则停止 SG-DMA 描述符处理。</p> <p>bit[1]: 如果发生错误，终止 SG-DMA 描述符处理。</p> <p>bit[0]: SG-DMA 描述符处理之后，结束 DMA 传输（相当于链的末端）。如果 DESC_SE_COND 的 [2:1] 字段被清除，一旦 PAGE_SIZE 字节被处理，DMA 将停止。如果 DMA 传输长度未知，应用程序可以构建一个动态的 SG-DMA 链表。在这种情况下，允许应用程序清除最后一个建立的 SG-DMA 描述符，并且 DESC_NEXT_ADDR 字段和 DESC_SE_COND 的第 0 位被认为和 DMA 引擎无关。</p> <p>一旦应用程序构建了下一个描述符（或者当前的 SG-DMA 描述符是链表中的最后一个描述符），它应当更新当前的 SG-DMA 描述符，设置 DESC_NEXT_ADDR 字段和 DESC_SE_COND 的第 0 位为默认值。</p> <p>DMA 引擎定期查询 SG-DMA 描述符是否被更新（每 1us 到 65us，取决于核的参数配置）。然而应用程序可以将 DMA 引擎的 DMA_CONTROL 寄存器的 DESC_UPDT 字段设置为 1 来表明描述符已经被更新了。</p>

5.1.13.4 DESC_NEXT_ADDR[63:32] (0x0C)

域	位	读写	复位值	描述
DESC_NEXT_ADDR[63:32]	31:0	RW	0x0	下一个描述符地址

5.1.13.5 DESC_SRC_ADDR[31:0] (0x10)

域	位	读写	复位值	描述
DESC_SRC_ADDR [31:0]	31:0	RW	0x0	源地址 0~31 位。如果没有使用（SG_TYPE 字段设置成 2'b10），应用程序设置该制字段为 64'b0。

5.1.13.6 DESC_SRC_ADDR[63:32] (0x14)

域	位	读写	复位值	描述
DESC_SRC_ADDR[63:32]	31:0	RW	0x0	源地址 32~63 位

5.1.13.7 DESC_DEST_ADDR[31:0] (0x18)

域	位	读写	复位值	描述
DESC_DEST_ADDR[31:0]	31:0	RW	0x0	目标地址 0~31 位，如果没有使用（SG_TYPE 字段设置成 2'b10），应用程序设置该字段为 64'b0。

5.1.13.8 DESC_DEST_ADDR[63:32] (0x1C)

域	位	读写	复位值	描述
DESC_DEST_ADDR[63:32]	31:0	RW	0x0	目标地址 32~63 位

5.2 SMMU

SMMU 实现对 PCIe、SATA 外设虚实地址转化，用于外设虚拟化。

5.2.1 操作说明

5.2.1.1 中断

表 5-11 SSMU 中断分配

中断信号	中断含义
smmu_tcu_pri_q_irpt_ns	TCU 非安全中断，指示 smmu PRI queue 中断
smmu_tcu_event_q_irpt_ns	TCU 非安全中断，指示 smmu PRI queue 中断
smmu_tcu_event_q_irpt_s	TCU 非安全中断，指示 smmu Event_Qenen 非空或溢出
smmu_tcu_global_irpt_ns	TCU 非安全中断，指示 smmu globle Non-Secure interrupt
smmu_tcu_global_irpt_s	TCU 安全中断，指示 smmu globle Secure interrupt
smmu_tcu_pmu_irpt	TCU 指示 PMU 中断
smmu_tbu0_pmu_irpt	TBU0 PMU 中断
smmu_tbu1_pmu_irpt	TBU1 PMU 中断
smmu_tbu2_pmu_irpt	TBU2 PMU 中断

5.2.1.2 TBU Bypass 模式配置

SMMU 支持每个 TBU 外围可配置的 bypass 通路。

5.2.1.3 软件 Bypass 模式配置

SMMU 的软件 bypass 模式有两组寄存器需要配置，均为全局配置寄存器：

SMMU_CR0/SMMU_GBPA 只用于 Non-secure streamID 请求，对 Secure streamID 请求不起作用；SMMU_S_CR0/SMMU_S_GBPA 只用于 Secure streamID 请求，对 Non-secure

streamID 请求不起作用。

5.2.1.4 使能 SMMU 翻译模式配置

使能 SMMU 翻译模式需要将 CRO 的 SMMU_EN 置为 1，整个过程 SMMU 的寄存器都将由 SMMU 驱动完成，用户不必做任何配置。

5.2.2 寄存器列表

表 5-12 SMMU 寄存器基地址

名称	基地址
SMMU	0x000_3000_0000

表 5-13 SMMU 寄存器列表

寄存器名称	偏移	描述
SMMU_CRO	0x0020	SMMU_CRO 非安全 stream 控制寄存器（SMMUv3 定义）
SMMU_GBPA	0x0044	SMMU 非安全 stream 全局 bypass 特性控制寄存器（SMMUv3 定义）
SMMU_S_CRO	0x8020	SMMU_S_CRO 安全 stream 控制寄存器（SMMUv3 定义）
SMMU_S_GBPA	0x8044	SMMU 安全 stream 全局 bypass 特性控制寄存器（SMMUv3 定义）
TCU_CTRL	0x8E00	TCU 缓存能力控制寄存器
TCU_NODE_CTRL _x	0x9000+ _x *0x4	TCU 连接点 TBU _x 的 DVM 与优先级控制寄存器
creg_ace_mux_ctl	0x2B3_90A0	tbu 的 ace 通道 bypass 控制寄存器
creg_smmu_tcu_ctl	0x2B3_9100	smmu 翻译请求特性控制寄存器
creg_smmu_pmu_ctl	0x2B3_9104	smmu pmu 快照特性控制寄存器
creg_smmu_tbox_ctl	0x2B3_9108+ _x *0x4	tbox 翻译特性控制寄存器
creg_smmu_wake_ctl	0x2B3_9114	smmu 通道唤醒控制与状态寄存器

注意：表中 _x 取值为 0~3。

5.2.3 寄存器说明

5.2.3.1 SMMU_CRO (0x0020)

域	位	读写	复位值	描述
Reserved, RES0	31:9	RW	0x0	保留
VMW	8:6	RW	0x0	VMID Wildcard 3'b000: TLB invalidations 操作应完全匹配 VMID tags 3'b001: TLB invalidations 操作应匹配 VMID[N:1] 3'b010: TLB invalidations 操作应匹配 VMID[N:2] 3'b011: TLB invalidations 操作应匹配 VMID[N:3] 3'b100: TLB invalidations 操作应匹配 VMID[N:4]

域	位	读写	复位值	描述
Reserved, RES0	5	RW	0x0	保留
ATSchk	4	RW	0x0	ATS behavior 0: Fast mode: ATS 请求经过 SMMU 时不查询 STE 和 TLB 已进行检查 1: Safe mode: ATS 请求经过 SMMU 时应查询 STE. EATS 以决定该 StreamID 是否允许继续翻译
CMDQEN	3	RW	0x0	Enable Command queue processing 0: Non-secure Command queue 中的命令禁止继续执行 1: Non-secure Command queue 中的命令允许继续执行
EVENTQEN	2	RW	0x0	Enable Event queue writes 0: 禁止继续向 Non-secure Event queue 中写 event 1: 允许继续向 Non-secure Event queue 中写 event
PRIQEN	1	RW	0x0	Enable PRI queue writes 0: 禁止继续向 PRI queue 中写 event 1: 允许继续向 PRI queue 中写 event
SMMUEN	0	RW	0x0	Non-secure SMMU enable 0: bypass 所有 Non-secure 请求, 此时具体行为有 SMMU_GBPA 寄存器控制 1: 检查并翻译所有 Non-secure 请求

5.2.3.2 SMMU_GBPA (0x0044)

域	位	读写	复位值	描述
Update completion flag	31	RW1C	0x0	配置 SMMU_GBPA 时写 1, 配置完成后自动清 0
Reserved, RES0	30:21	RW	0x0	保留
ABORT	20	RW	0x0	当 SMMU_CR0.SMMUEN==0 时对请求 Abort 并产生 fault
INSTCFG	19:18	RW	0x0	对于读请求, global bypass 时, 强制覆盖 arprot[2] 的 Instruction/data 属性 2'b00: 不覆盖 2'b01: Reserved, 不覆盖 2'b10: 覆盖 arprot[2]=0, 即 Data 属性 2'b11: 覆盖 arprot[2]=1, 即 Instruction 属性 INSTCFG 只影响读请求, 不影响写请求
PRIVCFG	17:16	RW	0x0	对于读写请求, global bypass 时, 强制覆盖 arprot[0] 的 User/privileged 属性 2'b00: 不覆盖 2'b01: Reserved, 不覆盖 2'b10: 覆盖 axprot[0]=0, 即 Unprivileged 属性 2'b11: 覆盖 axprot[0]=1, 即 Privileged 属性

域	位	读写	复位值	描述
Reserved, RES0	15:14	RW	0x0	保留
SHCFG	13:12	RW	0x0	对于读写请求, global bypass 时, 强制覆盖 ax_domain[1:0]的 Shareability 属性 2'b00: Non-shareable 2'b01: Use incoming 2'b10: Outer Shareable 2'b11: Inner Shareable
ALLOCCFG	11:8	RW	0x0	对于读写请求, global bypass 时, 强制覆盖 ax_cache[3:0]的 Allocation 属性 4'b0xxx use incoming RA/WA/TR allocation/transient hints 4'b1RWT Hints are overridden to given values: - Read Allocate == R - Write Allocate == W - Transient == T
Reserved, RES0	7:5	RW	0x0	保留
MTCFG: Memory type override	4	RW	0x0	0: ax_cache[3:0]不被覆盖 1: ax_cache[3:0]使用 SMMU_ GBPA. MemAttr 覆盖
MemAttr	3:0	RW	0x0	Memory type 用于设置 smmu bypass 时的 MemAttr 属性, 编码与 STE 中的 MemAttr 相同

5.2.3.3 SMMU_S_CRO (0x8020)

域	位	读写	复位值	描述
Reserved, RES0	31:10	RW	0x0	保留
NSSTALLD	9	RW	0x0	Non-secure stall disable 0: Non-secure 配置请求可以使用 Stall mode 1: Non-secure 配置请求禁止使用 Stall mode
Reserved, RES0	8:6	RW	0x0	保留
SIF	5	RW	0x0	Secure Instruction Fetch 0: 允许从标记为非安全状态的内存空间中获取安全状态的指令 1: 禁止从标记为非安全状态的内存空间中获取安全状态的指令
Reserved, RES0	4	RW	0x0	保留
CMDQEN	3	RW	0x0	Enable Command queue processing 0: 非安全命令队列中的命令禁止继续执行 1: 非安全命令队列中的命令允许继续执行
EVENTQEN	2	RW	0x0	Enable Event queue writes 0: 禁止继续向 Non-secure Event queue 中写 event 1: 允许继续向 Non-secure Event queue 中写 event
Reserved, RES0	1	RW	0x0	保留

域	位	读写	复位值	描述
SMMUEN	0	RW	0x0	Non-secure SMMU enable 0: bypass 所有 Non-secure 请求, 此时具体行为有 SMMU_GBPA 寄存器控制。 1: 检查并翻译所有 Non-secure 请求。

5.2.3.4 SMMU_S_GBPA (0x8044)

域	位	读写	复位值	描述
Update completion flag	31	RW1C	0x0	配置 SMMU_S_GBPA 时写 1, 配置完成后自动清 0。
Reserved, RES0	30:21	RW	0x0	保留
ABORT	20	RW	0x0	当 SMMU_S_CR0.SMMUEN==0 时对请求 Abort 并产生 fault
INSTCFG	19:18	RW	0x0	对于读请求, global bypass 时, 强制覆盖 arprot[2] 的 Instruction/data 属性 2'b00: 不覆盖 2'b01: reserved, 不覆盖 2'b10: 覆盖 arprot[2]=0, 即 Data 属性 2'b11: 覆盖 arprot[2]=1, 即 Instruction 属性 INSTCFG 只影响读请求, 不影响写请求
PRIVCFG	17:16	RW	0x0	对于读写请求, global bypass 时, 强制覆盖 arprot[0] 的 User/privileged 属性 2'b00: 不覆盖 2'b01: reserved, 不覆盖 2'b10: 覆盖 axprot[0]=0, 即 Unprivileged 属性 2'b11: 覆盖 axprot[0]=1, 即 Privileged 属性
NSCFG: NS override	15:14	RW	0x0	对于读写请求, global bypass 时, 强制覆盖 arprot[0] 的 Secure/Non-secure 属性 2'b00: Use incoming 2'b01: Reserved, behaves as 2'b00 2'b10: Secure 2'b11: Non-secure
SHCFG	13:12	RW	0x0	对于读写请求, global bypass 时, 强制覆盖 ax_domain[1:0] 的 Shareability 属性 2'b00: Non-shareable 2'b01: Use incoming 2'b10: Outer Shareable 2'b11: Inner Shareable
ALLOCCFG	11:8	RW	0x0	对于读写请求, global bypass 时, 强制覆盖 ax_cache[3:0] 的 Allocation 属性 4'b0xxx use incoming RA/WA/TR allocation/transient hints 4'b1RWT Hints are overridden to given values: - Read Allocate == R

域	位	读写	复位值	描述
				<ul style="list-style-type: none"> - Write Allocate == W - Transient == T
Reserved, RES0	7:5	RW	0x0	保留
MTCFG: Memory type override	4	RW	0x0	0: ax_cache[3:0]不被覆盖 1: ax_cache[3:0]使用 SMMU_S_GBP. MemAttr 覆盖
MemAttr	3:0	RW	0x0	Memory type 用于设置 smmu bypass 时的 MemAttr 属性, 编码与 STE 中的 MemAttr 相同

5.2.3.5 TCU_CTRL (0x8E00)

域	位	读写	复位值	描述
reserved	31:16	RW	0x0	保留
WCS2L3_DIS	15	RW	0x0	禁止 tcu 对 Stage 2 level 3 Walk 过程的缓存: 0: enabled; 1: disabled
WCS2L2_DIS	14	RW	0x0	禁止 tcu 对 Stage 2 level 2 Walk 过程的缓存: 0: enabled; 1: disabled
WCS2L1_DIS	13	RW	0x0	禁止 tcu 对 Stage 2 level 1 Walk 过程的缓存: 0: enabled; 1: disabled
WCS2L0_DIS	12	RW	0x0	禁止 tcu 对 Stage 2 level 0 Walk 过程的缓存: 0: enabled; 1: disabled
WCS1L3_DIS	11	RW	0x0	禁止 tcu 对 Stage 1 level 3 Walk 过程的缓存: 0: enabled; 1: disabled
WCS1L2_DIS	10	RW	0x0	禁止 tcu 对 Stage 1 level 2 Walk 过程的缓存: 0: enabled; 1: disabled
WCS1L1_DIS	9	RW	0x0	禁止 tcu 对 Stage 1 level 1 Walk 过程的缓存: 0: enabled; 1: disabled
WCS1L0_DIS	8	RW	0x0	禁止 tcu 对 Stage 1 level 0 Walk 过程的缓存: 0: enabled; 1: disabled
reserved	7:0	RW	0x0	保留

5.2.3.6 TCU_NODE_CTRLx (0x9000+x*0x4)

域	位	读写	复位值	描述
-	31:5	RW	0x0	保留
DIS_DVM	4	RW	0x0	禁止 DVM。置 1 时, 该节点设备不参与 DVM 操作。 若当前节点响应 DVM 速度慢, 可以对 DIS_DVM 置 1 以提高效率, 只能在初始化时置位。 Note: 直连 DTI 的 ATS 设备不参与 DVM invalidation, 因此忽略该位。
-	3:2	RW	0x0	保留
PRI_LEVEL	1:0	RW	0x0	SMMU 节点优先级, 优先级高的节点请求优先处理。

5.2.3.7 creg_ace_mux_ctl (0x2B3_90A0)

域	位	读写	复位值	描述
ace_mux_sel_2	2	RW	0x1	控制 TBU2 开启 bypass 通道，默认开启 tbu bypass 通道，即不经过 smmu。
ace_mux_sel_1	1	RW	0x0	控制 TBU1 开启 bypass 通道，默认关闭 tbu bypass 通道，即经过 smmu。
ace_mux_sel_0	0	RW	0x0	控制 TBU0 开启 bypass 通道，默认关闭 tbu bypass 通道，即经过 smmu。

5.2.3.8 creg_smmu_tcu_ctl (0x2B3_9100)

域	位	读写	复位值	描述
smmu_tcu_sup_oas	6:4	RW	0x2	设置 TCU 发出访问的地址有效位宽： 3'b000: 32 bits 3'b001: 36 bits 3'b010: 40 bits 3'b011: 42 bits 3'b100: 44 bits 3'b101: 48 bits
smmu_tcu_sec_override	3	RW	0x0	控制 TCU 的安全寄存器能否被非安全请求强制访问。
smmu_tcu_sup_sev	2	RW	0x0	控制 event0 输出是否有效。
smmu_tcu_sup_btm	1	RW	0x1	控制 TCU 是否支持 DVM 请求，关闭后将不支持 DVM。
smmu_tcu_sup_cohacc	0	RW	0x1	控制 TCU 发出的请求是否有 IO 一致性，关闭后 TCU 将不保证数据一致性。

5.2.3.9 creg_smmu_pmu_ctl (0x2B3_9104)

域	位	读写	复位值	描述
smmu_pmusnapshot_ack	1	RO	0x0	smmu pmu 快照 ACK
smmu_pmusnapshot_req	0	RW	0x0	smmu pmu 快照请求

5.2.3.10 creg_smmu_tbox_ctl (0x2B3_9108+x*0x4)

域	位	读写	复位值	描述
smmu_tbox_cmo_disable	10	RW	0x0	控制 TBU 是否接收 CMO 请求：若置 1，CMO 请求将被禁止并返回 slver 响应。
smmu_tbox_utlb_roundrobin	9	RW	0x0	控制 TBU 中 Micro TLB 的替换算法： 0: 伪随机替换 1: 轮询替换
smmu_tbox_sec_overrid	8	RW	0x0	控制 TBU 的安全寄存器能否被非安全请求强制访问

域	位	读写	复位值	描述
smmu_tbox_s_sid_high	7:4	RW	0x0	设置 TBU 安全 streamid 的高 4bits
smmu_tbox_ns_sid_high	3:0	RW	0x0	设置 TBU 非安全 streamid 的高 4bits

5.2.3.11 creg_smmu_wake_ctl (0x2B3_9114)

域	位	读写	复位值	描述
smmu_awakeup_prog	8	RW	0x1	SMMU apb 配置端口唤醒控制信号
smmu_tbu2_awakeup_s	7	RW	0x1	TBU2 slave 端口唤醒控制信号
smmu_tbu2_awakeup_m	6	RO	0x0	TBU2 master 端口唤醒指示
smmu_tbu1_awakeup_s	5	RW	0x1	TBU1 slave 端口唤醒控制信号
smmu_tbu1_awakeup_m	4	RO	0x0	TBU1 master 端口唤醒指示
smmu_tbu0_awakeup_s	3	RW	0x1	TBU0 slave 端口唤醒控制信号
smmu_tbu0_awakeup_m	2	RO	0x0	TBU0 master 端口唤醒指示
smmu_tcu_acwakeup	1	RW	0x1	TCU AC 通道唤醒控制信号
smmu_tcu_awakeup	0	RO	0x0	TCU aw/ar 通道唤醒指示

5.3 以太网控制器

飞腾派以太网控制器包含千兆和万兆以太网接口，主要实现 AXI 总线与以太网接口互相转换。

5.3.1 操作说明

5.3.1.1 中断

MAC0 包含 8 个队列，分别是 q7~q0。MAC1/MAC2/MAC3 包含 4 个队列，分别是 q3~q0。每个队列有各自的中断。

- q0 队列中断

int_status 寄存器各 bit 表示的中断如下：

bit31：发送方向锁定检测中断；

bit30：接收方向锁定检测中断；

bit29：tsu 计数比较中断；

bit28：网络唤醒报文中断；

bit27：接收方向 LPI 状态指示 bit 值改变中断；

bit26：tsu 秒级寄存器增长中断；

bit25：PTP pdelay resp 报文发送中断；

bit24：PTP pdelay req 报文发送中断；

bit23：PTP pdelay resp 报文接收中断；

bit22: PTP pdelay req 报文接收中断;
bit21: PTP sync 报文发送中断;
bit20: PTP delay req 报文发送中断;
bit19: PTP sync 报文接收中断;
bit18: PTP delay req 报文接收中断;
bit17: 设置 PCS 时, 接收到对端 link 中断;
bit16: 设置 PCS 时, 自协商完成中断;
bit15: 输入引脚 ext_interrupt_in 接收到上升沿中断;
bit14: Pause 帧发送中断;
bit13: Pause 帧清除中断;
bit12: 非 0 Pause 帧接收中断;
bit11: bresp 通道不正常中断;
bit10: 接收超限中断;
bit9: link 状态改变中断;
bit7: 发送完成中断;
bit6: 由于 axi 总线造成发送错误中断;
bit5: 超出重新发送限制或延时冲突中断;
bit4: 强制停止发送中断;
bit3: 发送方向读到 used bit 中断;
bit2: 接收方向读到 used bit 中断;
bit1: 接收完成中断;
bit0: 管理报文中断。

● 其它队列中断

q7~q1 队列包含相同的中断如下:

bit7: 发送完成中断;
bit6: 由于 axi 总线造成发送错误中断;
bit5: 超出重新发送限制或延时冲突中断;
bit2: 接收方向读到 used bit 中断;
bit1: 接收完成中断。

5.3.1.2 初始化流程

初始化流程如下图, sgmi i pcs 包含有自协商功能, 可以选择开启或关闭, 如果开启, sgmi i pcs 会进行自协商过程, 软件需要根据协商结果进行速率、模式等进行对应的配置。

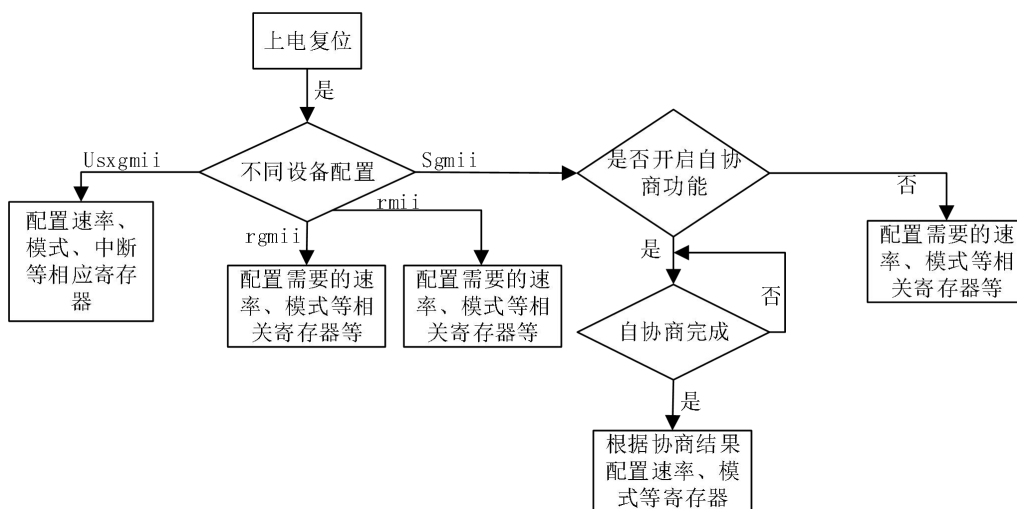


图 5-8 以太网控制器初始化流程图

5.3.1.3 MAC 工作流程

以 1G 速率模式为例介绍下 MAC 的基本工作流程：

- 配置本端 PHY 和对端 PHY 为 1G 速率模式；
- 配置 `hs_mac_config` 的 `hs_mac_speed` 为 1, 向 MAC 外部表明 MAC 的速率为千兆；
- 配置 `int_enable` 为 'hfffffff', 使能所有类型的中断；
- 配置 `pcs_control` 的 `enable_auto_neg` 为 0, 去使能自协商；
- 配置 `network_config` 的 `data_bus_width` 为 'b10', 设置 AXI 总线数据有效位宽为 128bit; 配置 `pcs_select` 为 1, 使能 `sgmii` 接口; 配置 `gigabit_mode_enable` 为 1, 配置 MAC 速率为千兆；
- 配置本端 MAC 的 `transmit_q_ptr` 的 0bit 为 'h0', 使能本端 MAC 的 q0 队列发送功能, 配置 q0 队列发送描述符地址 (`transmit_q_ptr` 的 [31:2]bit); 配置对端 MAC 的 `receive_q_ptr` 的 0bit 为 'h0', 使能对端 MAC 的 q0 队列接收功能, 配置 q0 队列接收描述符地址 `receive_q_ptr` 的 [31:2]bit);
- 配置 `network_control` 的 `enable_transmit` 和 `enable_receive` 为 1, 使能发送通道和接收通道；
- 等待本端 PHY 和对端 PHY 的 `pcs_mac_clk_ln_0`、`pcs_mac_clk_divx0_ln_0`、`pcs_mac_clk_divx1_ln_0`、`pma_rx_rd_clk_ln_0` 时钟按照预定的频率输出；
- 等待本端 MAC 与对端 MAC link；
- 配置 MAC0 的 `network_control` 的 `transmit_start` 为 1, 使能发送开始。

5.3.1.4 各种工作模式对应的配置

1. 10G usxgmii 模式

- `int_disable` 写 `0xFFFFFFFF` //使能中断
- `hs_mac_config` 写 `0x00000004` //hs_mac_speed 为 10G
- `usx_control_register` 写 `0x00011103` //[16:14]: `hs_mac_speed` 为 10G, 与寄存器 `hs_mac_config` 中配置一致; [13:12]: `serdes_rate` 配置为 10G; 1: `tx_datapath_en`; 0: `usxgmii pcs enable`
- `transmit_q_ptr` 写 `0x555555500` //[31:2] 发送方向队列 0 的地址指针; 0: `dma_tx_dis_q`, 队列的 `disable`. 地址根据实际情况配置
- `transmit_q1_ptr` 写 `0x555555500` //[31:2] 发送方向队列 1 的地址指针; 0: `dma_tx_dis_q`, 队列的 `disable`. 地址根据实际情况配置
- `transmit_q2_ptr` 写 `0x555555500` //[31:2] 发送方向队列 2 的地址指针; 0: `dma_tx_dis_q`, 队列的 `disable`. 地址根据实际情况配置
- `transmit_q3_ptr` 写 `0x555555500` //[31:2] 发送方向队列 3 的地址指针; 0: `dma_tx_dis_q`, 队列的 `disable`. 地址根据实际情况配置
- `receive_q_ptr` 写 `0xFFFFFFFF0` //[31:2] 接收方向队列 0 的地址指针; 0: `dma_rx_dis_q`, 队列的 `disable`. 地址根据实际情况配置
- `receive_q1_ptr` 写 `0xFFFFFFFF0` //[31:2] 接收方向队列 0 的地址指针; 0: `dma_rx_dis_q`, 队列的 `disable`. 地址根据实际情况配置
- `receive_q2_ptr` 写 `0xFFFFFFFF0` //[31:2] 接收方向队列 0 的地址指针; 0: `dma_rx_dis_q`, 队列的 `disable`. 地址根据实际情况配置
- `receive_q3_ptr` 写 `0xFFFFFFFF0` //[31:2] 接收方向队列 0 的地址指针; 0: `dma_rx_dis_q`, 队列的 `disable`. 地址根据实际情况配置
- `network_control` 写 `0x0000020c` //[9]: `transmit_start`, [3]: `enable_transmit`; [2]: `enable_receive`
- `network_control` 写 `0x0000040c` //[10]: `transmit_halt`

2. 1G sgmi i

- `network_config` 寄存器写 `0x00000c00` //[11]: `pcs_select`, `sgmii` 模式下需要设置; [10]: `gigabit_mode_enable`, 设置为 1 的时候
- `transmit_q_ptr` 寄存器写 `0x555555500` //[31:2] 发送方向队列 0 的地址指针; 0: `dma_tx_dis_q`, 队列的 `disable` 地址根据实际情况配置
- `transmit_q1_ptr` 寄存器写 `0x555555500` //[31:2] 发送方向队列 1 的地址指针; 0: `dma_tx_dis_q`, 队列的 `disable`. 地址根据实际情况配置

- transmit_q2_ptr 寄存器写 0x555555500 //[31:2] 发送方向队列 2 的地址指针; 0: dma_tx_dis_q, 队列的 disable. 地址根据实际情况配置
- transmit_q3_ptr 寄存器写 0x555555500 //[31:2] 发送方向队列 3 的地址指针; 0: dma_tx_dis_q, 队列的 disable. 地址根据实际情况配置
- receive_q_ptr 寄存器写 0xFFFFFFFF0 //[31:2] 接收方向队列 0 的地址指针; 0: dma_rx_dis_q, 队列的 disable. 地址根据实际情况配置
- receive_q1_ptr 寄存器写 0xFFFFFFFF0 //[31:2] 接收方向队列 0 的地址指针; 0: dma_rx_dis_q, 队列的 disable. 地址根据实际情况配置
- receive_q2_ptr 寄存器写 0xFFFFFFFF0 //[31:2] 接收方向队列 0 的地址指针; 0: dma_rx_dis_q, 队列的 disable 地址根据实际情况配置
- receive_q3_ptr 寄存器写 0xFFFFFFFF0 //[31:2] 接收方向队列 0 的地址指针; 0: dma_rx_dis_q, 队列的 disable. 地址根据实际情况配置
- network_control 寄存器写 0x0000020c //[9]: transmit_start, [3]: enable_transmit; [2]: enable_receive
- network_control 寄存器写 0x0000040c //[10]: transmit_halt

3. rgmii

- network_config 寄存器写 0x00000402 //[10]: gigabit_mode_enable, [2]: full_duplex
- transmit_q_ptr 寄存器写 0x555555500 //[31:2] 发送方向队列 0 的地址指针; 0: dma_tx_dis_q, 队列的 disable. 地址根据实际情况配置
- transmit_q1_ptr 寄存器写 0x555555500 //[31:2] 发送方向队列 1 的地址指针; 0: dma_tx_dis_q, 队列的 disable. 地址根据实际情况配置
- transmit_q2_ptr 寄存器写 0x555555500 //[31:2] 发送方向队列 2 的地址指针; 0: dma_tx_dis_q, 队列的 disable. 地址根据实际情况配置
- transmit_q3_ptr 寄存器写 0x555555500 //[31:2] 发送方向队列 3 的地址指针; 0: dma_tx_dis_q, 队列的 disable. 地址根据实际情况配置
- receive_q_ptr 寄存器写 0xFFFFFFFF0 //[31:2] 接收方向队列 0 的地址指针; 0: dma_rx_dis_q, 队列的 disable. 地址根据实际情况配置
- receive_q1_ptr 寄存器写 0xFFFFFFFF0 //[31:2] 接收方向队列 0 的地址指针; 0: dma_rx_dis_q, 队列的 disable. 地址根据实际情况配置
- receive_q2_ptr 寄存器写 0xFFFFFFFF0 //[31:2] 接收方向队列 0 的地址指针; 0: dma_rx_dis_q, 队列的 disable. 地址根据实际情况配置
- receive_q3_ptr 寄存器写 0xFFFFFFFF0 //[31:2] 接收方向队列 0 的地址指针; 0: dma_rx_dis_q, 队列的 disable. 地址根据实际情况配置

- network_control 寄存器写 0x0000020c //[9]: transmit_start, [3]: enable_transmit; [2]: enable_receive
- network_control 寄存器写 0x0000040c //[10]: transmit_halt

4. rmii

- int_enable 寄存器写 0xFFFFFFFF //使能中断
- network_config 寄存器写 0x00000001 //100m

5.3.2 寄存器列表

表 5-14 以太网控制器寄存器基地址

名称	基地址
MAC0	0x000_3200_C000
MAC1	0x000_3200_E000
MAC2	0x000_3201_0000
MAC3	0x000_3201_2000

表 5-15 以太网控制器寄存器列表

寄存器名称	偏移	描述
network_control	0x0000	网络控制寄存器
network_config	0x0004	网络配置寄存器
network_status	0x0008	网络状态寄存器
dma_config	0x0010	DMA 配置寄存器
transmit_status	0x0014	发送状态寄存器
receive_q_ptr	0x0018	q0 队列接收描述符地址寄存器
transmit_q_ptr	0x001C	q0 队列发送描述符地址寄存器
recevie_status	0x0020	接收状态寄存器
int_status	0x0024	中断状态寄存器
int_enable	0x0028	中断使能寄存器
int_disable	0x002C	中断去使能寄存器
int_mask	0x0030	中断屏蔽寄存器
phy_management	0x0034	Phy 管理寄存器
pause_time	0x0038	接收 pause 量寄存器
tx_pause_quantum	0x003C	发送 pause 量寄存器
jumbo_max_length	0x0048	巨型帧长度寄存器
hs_mac_config	0x0050	高速配置寄存器
axi_max_pipeline	0x0054	AXI 总线 outstanding 配置寄存器
hash_bottom	0x0080	hash 地址的第一个 32bit 寄存器
hash_top	0x0084	hash 地址的其余 32bit 寄存器
spec_add1_bottom	0x0088	第 1 组目的地址寄存器
spec_add1_top	0x008C	第 1 组源地址寄存器
spec_add2_bottom	0x0090	第 2 组目的地址寄存器
spec_add2_top	0x0094	第 2 组源地址寄存器
spec_add3_bottom	0x0098	第 3 组目的地址寄存器

寄存器名称	偏移	描述
spec_add3_top	0x009C	第 3 组源地址寄存器
spec_add4_bottom	0x00A0	第 4 组目的地址寄存器
spec_add4_top	0x00A4	第 4 组源地址寄存器
tsu_ptp_tx_msb_sec	0x00E8	PTP 报文发送秒数寄存器
tsu_ptp_rx_msb_sec	0x00EC	PTP 报文接收秒数寄存器
tsu_peer_tx_msb_sec	0x00F0	PTP PEER 报文发送秒数寄存器
tsu_peer_rx_msb_sec	0x00F4	PTP PEER 报文接收秒数寄存器
octets_txed_bottom	0x0100	正确发送报文数低 32bit
octets_txed_top	0x0104	正确发送报文数高 16bit
frames_txed_ok	0x0108	正确发送报文数
broadcast_txed	0x010C	正确发送 Broadcast 报文数
multicast_txed	0x0110	正确发送 Multicast 报文数
pause_frames_txed	0x0114	正确发送 pause 报文数
frames_txed_64	0x0118	正确发送长度为 64 字节的报文数
frames_txed_65	0x011C	正确发送长度在 65~127 字节之间的报文数
frames_txed_128	0x0120	正确发送长度在 128~255 字节之间的报文数
frames_txed_256	0x0124	正确发送长度在 256~511 字节之间的报文数
frames_txed_512	0x0128	正确发送长度在 512~1023 字节之间的报文数
frames_txed_1024	0x012C	正确发送长度在 1024~1518 字节之间的报文数
frames_txed_1519	0x0130	正确发送长度在 1519~16k 字节之间的报文数
underruns	0x0134	传输欠载寄存器
single_collisions	0x0138	单独冲突报文寄存器
multiple_collisions	0x013C	多冲突报文寄存器
excessive_collisions	0x0140	超载冲突寄存器
late_collisions	0x0144	延迟冲突寄存器
deferred_frames	0x0148	延迟传输报文寄存器
crc_errors	0x014C	载波检测错误寄存器
octets_rxed_bottom	0x0150	正确接收报文数低 32bit
octets_rxed_top	0x0154	正确接收报文数高 16bit
frames_rxed_ok	0x0158	正确接收报文数
broadcast_rxed	0x015C	正确接收 Broadcast 报文数
multicast_rxed	0x0160	正确接收 Multicast 报文数
pause_frames_rxed	0x0164	正确接收 pause 报文数
frames_rxed_64	0x0168	正确接收长度为 64 字节的报文数
frames_rxed_65	0x016C	正确接收长度在 65~127 字节之间的报文数
frames_rxed_128	0x0170	正确接收长度在 128~255 字节之间的报文数
frames_rxed_256	0x0174	正确接收长度在 256~511 字节之间的报文数
frames_rxed_512	0x0178	正确接收长度在 512~1023 字节之间的报文数
frames_rxed_1024	0x017C	正确接收长度在 1024~1518 字节之间的报文数
frames_rxed_1519	0x0180	正确接收长度在 1519~16k 字节之间的报文数
undersize_frames	0x0184	接收报文长度过小报文计数
excessive_rx_length	0x0188	接收报文长度过大报文计数
rx_jabbers	0x018C	接收的报文长度大于 1518 字节计数

寄存器名称	偏移	描述
rx_fcs_errors	0x0190	报文检查序列错误
rx_length_errors	0x0194	报文长度小于从长度字段里提取的报文长度的报文数量
rx_symblo_errors	0x0198	在接收期间, rx_er 有效时接收的报文数量
rx_alignment_errors	0x019C	alignment 错误
rx_resource_errors	0x01A0	接收源错误
rx_overruns	0x01A4	接收溢出, 没复制到存储里的报文数量
rx_ip_ck_errors	0x01A8	IP 头校验错误
rx_tcp_ck_errors	0x01AC	TCP 头校验错误
rx_dup_ck_errors	0x01B0	UDP 头校验错误
tsu_timer_inc_sub_nsec	0x01BC	1588 计时器每个时钟增长的量
tsu_timer_msb_sec	0x01C0	su 计时器值
tsu_timer_sec	0x01D0	T1588 计时器秒寄存器
tsu_timer_nsec	0x01D4	纳秒计时器计数
tsu_timer_adjust	0x01D8	1588 计时器纳秒寄存器纳秒增长或减少数量
tsu_timer_incr	0x01DC	1588 计时器纳秒寄存器的纳秒计数, 随着每拍时钟增长
tsu_ptp_tx_sec	0x01E0	PTP 报文发送秒数
tsu_ptp_tx_nsec	0x01E4	PTP 报文发送纳秒数
tsu_ptp_rx_sec	0x01E8	PTP 报文接收秒数
tsu_ptp_rx_nsec	0x01EC	PTP 报文接收纳秒数
tsu_peer_tx_sec	0x01F0	PTP PEER 报文发送秒数
tsu_peer_tx_nsec	0x01F4	PTP PEER 报文发送纳秒数
tsu_peer_rx_sec	0x01F8	PTP PEER 报文接收秒数
tsu_peer_rx_nsec	0x01FC	PTP PEER 报文接收纳秒数
pcs_control	0x0200	pcs (物理子层编码) 控制寄存器
int_q1_status	0x0400	队列 1 中断状态寄存器
int_q2_status	0x0404	队列 2 中断状态寄存器
int_q3_status	0x0408	队列 3 中断状态寄存器
int_q4_status	0x040C	队列 4 中断状态寄存器
int_q5_status	0x0410	队列 5 中断状态寄存器
int_q6_status	0x0414	队列 6 中断状态寄存器
int_q7_status	0x0418	队列 7 中断状态寄存器
transmit_q1_ptr	0x0440	队列 1 发送描述符地址寄存器
transmit_q2_ptr	0x0444	队列 2 发送描述符地址寄存器
transmit_q3_ptr	0x0448	队列 3 发送描述符地址寄存器
transmit_q4_ptr	0x044C	队列 4 发送描述符地址寄存器
transmit_q5_ptr	0x0450	队列 5 发送描述符地址寄存器
transmit_q6_ptr	0x0454	队列 6 发送描述符地址寄存器
transmit_q7_ptr	0x0458	队列 7 发送描述符地址寄存器
receive_q1_ptr	0x0480	队列 1 接收描述符地址寄存器
receive_q2_ptr	0x0484	队列 2 接收描述符地址寄存器
receive_q3_ptr	0x0488	队列 3 接收描述符地址寄存器
receive_q4_ptr	0x048C	队列 4 接收描述符地址寄存器

寄存器名称	偏移	描述
receive_q5_ptr	0x0490	队列 5 接收描述符地址寄存器
receive_q6_ptr	0x0494	队列 6 接收描述符地址寄存器
receive_q7_ptr	0x0498	队列 7 接收描述符地址寄存器
upper_tx_q_base_addr	0x04C8	发送缓存描述符队列基地址的高 32 位
tx_bd_control	0x04CC	发送描述符时戳插入模式
rx_bd_control	0x04D0	接收描述符时戳插入模式
upper_rx_q_base_addr	0x04D4	接收缓存描述符队列基地址的高 32 位
tx_q_seg_alloc_q_lower	0x05A0	给各队列分配空间。各 MAC 总空间为 32KB。每个分段大小为 2KB，最大可分配 16 个分段（MAC1 除外，MAC1 每个分段大小为 1KB，最大可分配 32 个分段）。
int_qx_enable	0x0600+0x4*(x-1)	队列 x 中断使能寄存器（x 取值位为 1~7）
int_qx_disable	0x0620+0x4*(x-1)	队列 x 中断去使能寄存器（x 取值位为 1~7）
int_qx_mask	0x0640+0x4*(x-1)	队列 x 中断屏蔽寄存器（x 取值位为 1~7）
usx_control_register	0x0A80	usxgmii 控制寄存器
usx_status_register	0x0A88	usxgmii 状态寄存器
mmsl_control	0x0F00	MMSL 控制寄存器
mmsl_status	0x0F04	MMSL 状态寄存器
mmsl_err_stats	0x0F08	MMSL 错误统计寄存器
mmsl_ass_ok_count	0x0F0C	MMSL 帧重组成功寄存器
mmsl_frag_count_rx	0x0F10	MMSL 接收片段计数寄存器
mmsl_frag_count_tx	0x0F14	MMSL 发送片段计数寄存器
mmsl_int_status	0x0F18	MMSL 中断状态寄存器
mmsl_int_enable	0x0F1C	MMSL 中断使能寄存器
mmsl_int_disable	0x0F20	MMSL 中断去使能寄存器
mmsl_int_mask	0x0F24	MMSL 中断屏蔽寄存器
emac_network_control	0x1000	EMAC 网络控制寄存器
emac_network_config	0x1004	EMAC 网络配置寄存器
emac_dma_config	0x1010	EMAC DMA 配置寄存器
emac_receive_q_ptr	0x1018	EMAC q0 队列接收描述符地址寄存器
emac_transmit_q_ptr	0x101C	EMAC q0 队列发送描述符地址寄存器
emac_int_status	0x1024	EMAC 中断状态寄存器
emac_hs_mac_config	0x1050	EMAC 高速配置
emac_axi_max_pipeline	0x1054	EMAC AXI 总线 outstanding 配置
emac_octets_txed_bottom	0x1100	EMAC 正确发送报文数低 32bit
emac_octets_txed_top	0x1104	EMAC 正确发送报文数高 16bit
emac_frames_txed_ok	0x1108	EMAC 正确发送报文数
emac_broadcast_txed	0x110C	EMAC 正确发送 Broadcast 报文数
emac_multicast_txed	0x1110	EMAC 正确发送 Multicast 报文数
emac_pause_frames_txed	0x1114	EMAC 正确发送 pause 报文数
emac_frames_txed_64	0x1118	EMAC 正确发送长度为 64 字节的报文数
emac_frames_txed_65	0x111C	EMAC 正确发送长度在 65~127 字节之间的报文数
emac_frames_txed_128	0x1120	EMAC 正确发送长度在 128~255 字节之间的报文数
emac_frames_txed_256	0x1124	EMAC 正确发送长度在 256~511 字节之间的报文数

寄存器名称	偏移	描述
emac_frames_txed_512	0x1128	EMAC 正确发送长度在 512~1023 字节之间的报文数
emac_frames_txed_1024	0x112C	EMAC 正确发送长度在 1024~1518 字节之间的报文数
emac_frames_txed_1519	0x1130	EMAC 正确发送长度在 1519~16k 字节之间的报文数
emac_octets_rxed_bottom	0x1150	EMAC 正确接收报文数低 32bit
emac_octets_rxed_top	0x1154	EMAC 正确接收报文数高 16bit
emac_frames_rxed_ok	0x1158	EMAC 正确接收报文数
emac_broadcast_rxed	0x115C	EMAC 正确接收 Broadcast 报文数
emac_multicast_rxed	0x1160	EMAC 正确接收 Multicast 报文数
emac_pause_frames_rxed	0x1164	EMAC 正确接收 pause 报文数
emac_frames_rxed_64	0x1168	EMAC 正确接收长度为 64 字节的报文数
emac_frames_rxed_65	0x116C	EMAC 正确接收长度在 65~127 字节之间的报文数
emac_frames_rxed_128	0x1170	EMAC 正确接收长度在 128~255 字节之间的报文数
emac_frames_rxed_256	0x1174	EMAC 正确接收长度在 256~511 字节之间的报文数
emac_frames_rxed_512	0x1178	EMAC 正确接收长度在 512~1023 字节之间的报文数
emac_frames_rxed_1024	0x117C	EMAC 正确接收长度在 1024~1518 字节之间的报文数
emac_frames_rxed_1519	0x1180	EMAC 正确接收长度在 1519~16k 字节之间的报文数

注意：EMAC 寄存器只 mac0 有。

5.3.3 寄存器说明

5.3.3.1 network_control (0x0000)

域	位	读写	复位值	描述
reserved	31	RO	0x0	保留
two_pt_five_gig	29	RW	0x0	值为 1 时 2.5g 速率模式使能；值为 0 时 2.5g 速率模式去使能。
oss_correction_field	27	RW	0x0	值为 1 时，更新 1588 报文时戳值。
ext_rxq_sel_en	26	RW	0x0	值为 1 时，匹配 ext_match1, ext_match2, ext_match3 与 ext_match4 这 4 个值的报文可以通过。
pfc_ctrl	25	RW	0x0	值为 1 时，使能暂停帧。
one_step_sync_mode	24	RW	0x0	值为 1 时，更新 1588 报文头里的时戳值。
ext_tsu_port_enable	23	RW	0x0	值为 1 时，时戳值使用外部 TSU 计数。
store_udp_offset	22	RW	0x0	值为 1 时，每个接收报文的校验值的高 16bit 会被更新，更新的校验值是从 TCP 或 UDP 的头开始计算。
alt_sgmi mode	21	RW	0x0	使能 Alternative sgmi 模式。
ptp_unicast_ena	20	RW	0x0	值为 1 时，可识别 PTP unicast 报文。
tx_lpi_en	19	RW	0x0	值为 1 时发送低功耗报文；值为 0 时不发送低功耗报文。
flush_rx_pkt_pclk	18	WO	0x0	值为 1 时，清除接收方向 ram 里的下一个报文。

域	位	读写	复位值	描述
transmit_pfc_priority_based_pause_frame	17	WO	0x0	值为 1 时，发送 PFC 报文。
pfc_enable	16	RW	0x0	值为 1 时，使能 PFC 自协商功能和可以识别暂停帧。
store_rx_ts	15	RW	0x0	值为 1 时，每个接收报文的校验值会被接收报文时戳值里的非纳秒部分代替。
stats_read_snap	14	RW	0x0	值为 1 时，读取统计寄存器的原始值。
stats_take_snap	13	WO	0x0	值为 1 时，存储统计寄存器的当前值且清除统计寄存器的值。
tx_pause_frame_zero	12	WO	0x0	值为 1 时，发送 0 值暂停帧。
tx_pause_frame_req	11	WO	0x0	值为 1 时，发送暂停帧。
transmit_halt	10	RW	0x0	值为 1 时发送停止；值为 0 时发送不停止。
transmit_start	9	RW	0x0	值为 1 时发送开始；值为 0 时发送不开始。
back_pressure	8	RW	0x0	值为 1 时，在 10M/100M 的半双工模式时，使能背压功能。
stats_write_en	7	RW	0x0	值为 1 时，统计寄存器可用来测试。
inc_all_stats_regs	6	WO	0x0	值为 1 时，所有统计寄存器都会增长，这个功能以测试为目的。
clear_all_stats_regs	5	WO	0x0	值为 1 时清除所有统计寄存器；值为 0 时不清除所有统计寄存器。
man_port_en	4	RW	0x0	值为 1 时使能 MDIO 接口；值为 0 时去使能 MDIO 接口。
enable_transmit	3	RW	0x0	值为 1 时使能发送通道；值为 0 时去使能发送通道。
enable_receive	2	RW	0x0	值为 1 时使能接收通道；值为 0 时去使能接收通道。
loopback_local	1	RW	0x0	值为 1 时使能 local 环回；值为 0 时去使能 local 环回。
loopback	0	RW	0x0	控制对外输出的 loopback 引脚。

5.3.3.2 network_config (0x0004)

域	位	读写	复位值	描述
sgmii_mode_enable	27	RW	0x0	值为 1 时使能 sgmii 模式；值为 0 时去使能 smii 模式。
en_half_duplex_rx	25	RW	0x0	值为 1 时接收方向为半双工模式；值为 0 时去使能收方向半双工模式。
receive_checksum_offload_enable	24	RW	0x0	值为 1 时接收方向检查 IP、TCP、UDP 报文的校验值；值为 0 时接收方向不检查 IP、TCP、UDP 报文的校验值。
disable_copy_of_pause_frames	23	RW	0x0	值为 1 时 pause 帧不写到存储里；值为 0 时 pause 帧写到存储里。
data_bus_width	22:21	RW	0x0	axi 总线数据位宽。值为 0 时数据位宽为 32bit；值为 1 时数据位宽为 64bit；值为 2 时数据位宽

域	位	读写	复位值	描述
				为 128bit。
mdc_clock_division	20:18	RW	0x2	apb 时钟分频系数（apb 时钟频率为 100MHz），分频后的时钟用作 mdc 时钟。值为 0x0 时 8 分频；值为 0x1 时 16 分频；值为 0x2 时 32 分频；值为 0x3 时 48 分频；值为 0x4 时 64 分频；值为 0x5 时 96 分频；值为 0x6 时 128 分频；值为 0x7 时 224 分频。
fcs_remove	17	RW	0x0	值为 1 时接收报文去掉校验再写到存储里；值为 0 时接收报文不去掉校验写到存储里。
pcs_select	11	RW	0x0	值为 1 时使能 TBI 接口的 pcs 功能；值为 0 时使能 GMII/MII 接口的 pcs 功能。
gigabit_mode_enable	10	RW	0x0	值为 0 时当接口类型为 MII 或 TBI 时，MAC 速率配置为 10M/100M；值为 1 时当接口类型为 GMII 或 TBI 时，MAC 速率配置为千兆。 pcs_select 配置 1，gigabit_mode_enable 配置 1，是千兆 sgmiI 接口；pcs_select 配置 1，gigabit_mode_enable 配置 0，是百兆/十兆 sgmiI 接口；pcs_select 配置 0，gigabit_mode_enable 配置 1，是千兆 rgmiI 接口；pcs_select 配置 0，gigabit_mode_enable 配置 0，是百兆/十兆 rmiI 接口。
copy_all_frames	4	RW	0x0	值为 1 时接收方向去掉滤波功能；值为 0 时接收方向使能滤波功能。
jumbo_frames	3	RW	0x0	值为 1 时可处理长度最大 16k 字节的报文。值为 0 时不可处理长度最大 16k 字节报文。
full_duplex	1	RW	0x0	值为 1 时 mac 为全双工模式；值为 0 时 mac 为半双工模式。
speed	0	RW	0x0	值为 1 时配置 mac 速率为 100M；值为 0 时配置 mac 速率为 10M。当 bit10 值为 0 时，bit0 配合 bit10 一起使用。

5.3.3.3 network_status (0x0008)

域	位	读写	复位值	描述
reserved	31:4	R0	0x0	保留
mac_full_duplex	3	R0	0x0	值为 1 时 mac 为全双工模式；值为 0 时 mac 为半双工模式。
Man_done	2	R0	0x1	值为 1 时 phy 的 mdio 为空闲状态；值为 0 时 phy 的 mdio 为非空闲状态。
Mdio_in	1	R0	0x0	Mdio_in 引脚的值。
Pcs_link_state	0	R0	0x0	低速模式 pcs link 状态。值为 1 时 pcs link；值为 0 时 pcs 没 link。

5.3.3.4 dma_config (0x0010)

域	位	读写	复位值	描述
send_bcast_to_all_qs	31	RW	0x0	值为 1 时，可以匹配不止一个内容的广播报文被发送到所有队列。
dma_addr_bus_width_1	30	RW	0x0	dma 的地址位宽。值为 0 时 dma 地址位宽为 32bit；值为 1 时 dma 地址位宽为 64bit。
tx_bd_extended_mode_en	29	RW	0x0	值为 1 时，使能发送扩展描述符模式。
rx_bd_extended_mode_en	28	RW	0x0	值为 1 时，使能接收扩展描述符模式。
reserved	27	RO	0x0	保留
force_max_amba_burst_tx	26	RW	0x0	值为 1 时配置发送方向 dma 的 burst 达到最大值；值为 0 时不配置发送方向 dma 的 burst 达到最大值。
force_max_amba_burst_rx	25	RW	0x0	值为 1 时配置接收方向 dma 的 burst 达到最大值；值为 0 时不配置接收方向 dma 的 burst 达到最大值。
force_discard_on_err	24	RW	0x0	值为 1 时当读到 used bit 值为 1 时，自动丢弃报文；值为 0 时不丢弃报文。
rx_buf_size	23:16	RW	0x02	DMA 接收缓存空间配置。值为 0x01 时，DMA 接收缓存 64 字节；值为 0x02 时，DMA 接收缓存 128 字节；值为 0x18 时，DMA 接收缓存 1536 字节；值为 0xa0 时，DMA 接收缓存 10240 字节；值为 0xff 时，DMA 接收缓存 16320 字节。
reserved	15	RO	0x0	保留
reserved	14	RO	0x0	保留
crc_error_report	13	RW	0x0	值为 1 时，接收缓存描述符的 16bit 表示校验错误。
infinite_last_dbuf_size_en	12	RW	0x0	值为 1 时，表示数据缓存的最后一个描述符是可变的。
tx_pbuf_tcp_en	11	RW	0x0	值为 1 时，可产生发送的 IP，TCP 和 UDP 报文的校验值。
tx_pbuf_size	10	RW	0x1	发送方向报文缓存深度选择。值为 1 时，发送方向报文缓存深度为 4K 字节；值为 0 时，发送方向报文缓存深度为 2K 字节。
rx_pbuf_size	9:8	RW	0x3	接收方向报文缓存深度选择。值为 3 时，接收方向报文缓存深度为 8K 字节；值为 2 时，接收方向报文缓存深度为 2K 字节；值为 1 时，接收方向报文缓存深度为 2K 字节；值为 0 时，接收方向报文缓存深度为 1K 字节。
endian_swap_packet	7	RW	0x0	值为 1 时报文设置为大端；值为 0 时报文设置为小端。
endian_swap_management	6	RW	0x0	值为 1 时描述符设置为大端；值为 0 时描述符设置为小端。
hdr_data_splitting_en	5	RW	0x0	值为 1 时，报文的数据和头分开缓存。
amba_burst_length	4:0	RW	0x4	axi 总线 burst 数量最大值设置。值为

域	位	读写	复位值	描述
				5'b1xxxx 时 burst 最大值为 16；值为 5'b01xxx 时 burst 最大值为 8；值为 5'b001xx 时 burst 最大值为 4；值为 5'b0001x/00001 时 burst 最大值为 1；值为 5'b00000 时 burst 最大值为 256。

5.3.3.5 transmit_status (0x0014)

域	位	读写	复位值	描述
reserved	31:13	R0	0x0	保留
tx_used_bit_read_midframe	12	RW W1toC	0x0	使用多 buffer 操作时，当一个报文在队列里时，和这个报文相关的描述符里的 used bit 需要是 0。如果这个 bit 有效，bit4 的 amba_error 会拉高。写 1 清除
tx_frame_too_large	11	RW	0x0	发送报文太大，不能写到 SRAM 里。如果这个 bit 有效，bit4 的 amba_error 会拉高。写 1 清除
tx_dma_lockup_detected	10	RW	0x0	发送方向 DMA 锁定。写 1 清除
tx_mac_lockup_detected	9	RW	0x0	发送方向 MAC 锁定。写 1 清除
resp_not_ok	8	RW	0x0	总线响应异常错误。写 1 清除
Late_collision_occurred	7	RW	0x0	发生延迟冲突。仅在千兆模式时有效。
Transmit_under_run	6	RW	0x0	发送停止。由于报文数据无效，停止一个正在发送的报文时拉高。这个 bit 有效如果一个发送状态写回没完成，当另一个状态写回要进行时。写 1 清除
Transmit_complete	5	RW	0x0	当报文发送完成时有效。写 1 清除
Amba_error	4	RW	0x0	因为 AXI 错误，发送报文损坏。当从外部存储读取报文时出现错误：BRESP 错误和 buffer 深度不够（因为缓存深度不够导致报文发送停止、FCS 错误、tx_er 有效）。当 DMA 的 buffer 深度配置小于报文长度时也会有效。一些 AXI 错误直接导致 MAC 发送路径复位，在这些情况下，tx_er 不会拉高且这个 bit 不会有效。写 1 清除
Transmit_go	3	R0	0x0	发送进行标志。如果拉高发送正在进行。使用 DMA 接口时，该 bit 代表 tx_go 的值
Retry_limit_exceeded	2	RW	0x0	超出重试限制。写 1 清除
Collision_occurred	1	RW	0x0	出现冲突时有效。在 100/10 模式时，这个 bit 代表出现冲突或延时冲突。在千兆模式时，出现延时冲突时，这个 bit 无效。写 1 清除
Used_bit_read	0	RW	0x0	在读到发送描述符的 used_bit 拉高时有效。写 1 清除

5.3.3.6 receive_q_ptr (0x0018)

域	位	读写	复位值	描述
dma_rx_q_ptr	31:2	RW	0x0	q0 队列的接收描述符缓存地址。
dma_rx_dis_q	0	RW	0x1	值为 1 时 q0 队列接收无效; 值为 0 时 q0 队列接收有效。

5.3.3.7 transmit_q_ptr (0x001C)

域	位	读写	复位值	描述
dma_tx_q_ptr	31:2	RW	0x0	q0 队列的发送描述符缓存地址。
dma_tx_dis_q	0	RW	0x1	值为 1 时 q0 队列发送无效; 值为 0 时 q0 队列发送有效。

5.3.3.8 receive_status (0x0020)

域	位	读写	复位值	描述
reserved	31:6	R0	0x0	保留
rx_dma_lockup_detected	5	RW	0x0	接收方向 DMA 锁定时拉高。写 1 清除
rx_mac_lockup_detected	4	RW	0x0	接收方向 MAC 锁定时拉高。写 1 清除
resp_not_ok	3	RW	0x0	出现 bresp 错误时拉高。写 1 清除
Receive_overrun	2	RW	0x0	接收溢出。外部 sram 满时拉高; 或报文结束时, 没上报报文状态。或者 DMA buffer 溢出时拉高。对于 DMA 操作来说, buffer 会恢复当出现溢出时。写 1 清除
Frame_received	1	RW	0x0	在报文接收和写到存储里时拉高。写 1 清除
Buffer_not_available	0	RW	0x0	buffer 不可用标志。要使用一个 buffer 且指针表明 buffer 已经被使用时拉高。DMA 会在接收报文结束时重新读指针, 直到发现指针可用。写 1 清除

5.3.3.9 int_status (0x0024)

域	位	读写	复位值	描述
wol_interrupt	28	R0	0x0	值为 1 时表示有网络唤醒事件发生; 值为 0 时表示没有网络唤醒事件发生。
receive_lpi_indication_status_bit_change	27	R0	0x0	值为 1 表示接收方向低功耗状态 bit 改变; 值为 0 表示接收方向低功耗状态 bit 没改变。
pcs_auto_negotiation_complete	16	R0	0x0	值为 1 表示 pcs 自协商完成; 值为 0 表示 pcs 自协商未完成。
external_interrupt	15	R0	0x0	值为 1 表示外部中断引脚 ext_interrupt_in 检测到上升沿。值为 0 表示外部中断引脚 ext_interrupt_in 未检测到上升沿。
link_change	9	R0	0x0	值为 1 表示 pcs 的 link 状态改变。值为 0 表示 pcs 的 link 状态没改变。
transmit_complete	7	R0	0x0	值为 1 表示发送完成。值为 0 表示发送没完成。

域	位	读写	复位值	描述
tx_used_bit_read	3	R0	0x0	值为 1 表示发送方向读到 used bit。值为 0 表示发送方向没读到 used bit。
rx_used_bit_read	2	R0	0x0	值为 1 表示接收方向读到 used bit。值为 0 表示接收方向没读到 used bit。
receive_complete	1	R0	0x0	值为 1 表示一个报文已经存在 sram 里。值为 0 表示报文没在 sram 里。
management_frame_sent	0	R0	0x0	值为 1 表示 mdio 已完成操作。值为 0 表示 mdio 未完成操作。

5.3.3.10 int_enable (0x0028)

域	位	读写	复位值	描述
enable_tx_lockup_detected_interrupt	31	WO	0x0	使能发送方向禁闭检测中断
enable_rx_lockup_detected_interrupt	30	WO	0x0	使能接收方向禁闭检测中断
enable_tsu_timer_comparison_interrupt	29	WO	0x0	使能 TSU 计时器比较中断
enable_wol_event_received_interrupt	28	WO	0x0	使能网络唤醒事件接收中断
enable_rx_lpi_indication_interrupt	27	WO	0x0	使能接收方向低功耗指示中断
enable_tsu_seconds_register_increment	26	WO	0x0	使能 TSU 计数秒级寄存器增长中断
enable_ptp_pdelay_resp_frame_transmitted	25	WO	0x0	使能 PTP 延时反馈报文传输中断
enable_ptp_pdelay_req_frame_transmitted	24	WO	0x0	使能 PTP 延时请求报文传输中断
enable_ptp_pdelay_resp_frame_received	23	WO	0x0	使能 PTP 延时反馈报文接收中断
enable_ptp_pdelay_req_frame_received	22	WO	0x0	使能 PTP 延时请求报文接收中断
enable_ptp_sync_frame_transmitted	21	WO	0x0	使能 PTP 同步报文传输中断
enable_ptp_delay_req_frame_transmitted	20	WO	0x0	使能 PTP 延时请求报文传输中断
enable_ptp_sync_frame_received	19	WO	0x0	使能 PTP 同步报文接收中断
enable_ptp_delay_req_frame_received	18	WO	0x0	使能 PTP 延时请求报文接收中断
enable_pcs_link_partner_page_received	17	WO	0x0	使能 PCS 连接对端页接收中断
enable_pcs_auto_negotiation_complete_interrupt	16	WO	0x0	使能 PCS 自协商完成中断
enable_external_interrupt	15	WO	0x0	使能外部中断
enable_pause_frame_transmitted_interrupt	14	WO	0x0	使能暂停帧传输中断
enable_pause_time_zero_interrupt	13	WO	0x0	使能暂停时间值为 0 中断
enable_pause_frame_with_non_zero_pause_quantum_interrupt	12	WO	0x0	使能非零值暂停帧中断
enable_resp_not_ok_interrupt	11	WO	0x0	使能 bresp/hresp 信号没准备好中断
enable_receive_overrun_interrupt	10	WO	0x0	使能接收超时中断
enable_link_change_interrupt	9	WO	0x0	使能连接状态改变中断
enable_usxgmii_interrupt	8	WO	0x0	使能 USXGMII 接口中断

域	位	读写	复位值	描述
enable_transmit_complete_interrupt	7	WO	0x0	使能发送完成中断
enable_transmit_frame_corruption_due_to_amba_error_interrupt	6	WO	0x0	使能由于 AXI 总线错误导致报文错误中断
enable_retry_limit_exceeded_or_late_collision_interrupt	5	WO	0x0	使能超出重试限制或延时冲突中断
enable_transmit_buffer_under_run_interrupt	4	WO	0x0	使能发送缓存欠载中断
enable_transmit_used_bit_read_interrupt	3	WO	0x0	使能发送 used bit 读中断
enable_receive_used_bit_read_interrupt	2	WO	0x0	使能接收 used bit 读中断
enable_receive_complete_interrupt	1	WO	0x0	使能接收完成中断
enable_management_done_interrupt	0	WO	0x0	使能 mdio 接口完成中断

5.3.3.11 int_disable(0x002C)

域	位	读写	复位值	描述
disable_tx_lockup_detected_interrupt	31	WO	0x0	去使能发送方向禁闭检测中断
disable_rx_lockup_detected_interrupt	30	WO	0x0	去使能接收方向禁闭检测中断
disable_tsu_timer_comparison_interrupt	29	WO	0x0	去使能 TSU 计时器比较中断
disable_wol_event_received_interrupt	28	WO	0x0	去使能网络唤醒事件接收中断
disable_rx_lpi_indication_interrupt	27	WO	0x0	去使能接收方向低功耗指示中断
disable_tsu_seconds_register_increment	26	WO	0x0	去使能 TSU 计数秒级寄存器增长中断
disable_ptp_pdelay_resp_frame_transmitted	25	WO	0x0	去使能 PTP 延时反馈报文传输中断
disable_ptp_pdelay_req_frame_transmitted	24	WO	0x0	去使能 PTP 延时请求报文传输中断
disable_ptp_pdelay_resp_frame_received	23	WO	0x0	去使能 PTP 延时反馈报文接收中断
disable_ptp_pdelay_req_frame_received	22	WO	0x0	去使能 PTP 延时请求报文接收中断
disable_ptp_sync_frame_transmitted	21	WO	0x0	去使能 PTP 同步报文传输中断
disable_ptp_delay_req_frame_transmitted	20	WO	0x0	去使能 PTP 延时请求报文传输中断
disable_ptp_sync_frame_received	19	WO	0x0	去使能 PTP 同步报文接收中断
disable_ptp_delay_req_frame_received	18	WO	0x0	去使能 PTP 延时请求报文接收中断
disable_pcs_link_partner_page_received	17	WO	0x0	去使能 PCS 连接对端页接收中断
disable_pcs_auto_negotiation_complete_interrupt	16	WO	0x0	去使能 PCS 自协商完成中断
disable_external_interrupt	15	WO	0x0	去使能外部中断

域	位	读写	复位值	描述
disable_pause_frame_transmitted_interrupt	14	W0	0x0	去使能暂停帧传输中断
disable_pause_time_zero_interrupt	13	W0	0x0	去使能暂停时间值为 0 中断
disable_pause_frame_with_non_zero_pause_quantum_interrupt	12	W0	0x0	去使能非零值暂停帧中断
disable_resp_not_ok_interrupt	11	W0	0x0	去使能 bresp/hresp 信号没准备好中断
disable_receive_overrun_interrupt	10	W0	0x0	去使能接收超时中断
disable_link_change_interrupt	9	W0	0x0	去使能连接状态改变中断
disable_usxgmii_interrupt	8	W0	0x0	去使能 USXGMII 接口中断
disable_transmit_complete_interrupt	7	W0	0x0	去使能发送完成中断
disable_transmit_frame_corruption_due_to_amba_error_interrupt	6	W0	0x0	去使能由于 AXI 总线错误导致报文错误中断
disable_retry_limit_exceeded_or_late_collision_interrupt	5	W0	0x0	去使能超出重试限制或延时冲突中断
disable_transmit_buffer_under_run_interrupt	4	W0	0x0	去使能发送缓存欠载中断
disable_transmit_used_bit_read_interrupt	3	W0	0x0	去使能发送 used bit 读中断
disable_receive_used_bit_read_interrupt	2	W0	0x0	去使能接收 used bit 读中断
disable_receive_complete_interrupt	1	W0	0x0	去使能接收完成中断
disable_management_done_interrupt	0	W0	0x0	去使能 mdio 接口完成中断

5.3.3.12 int_mask (0x0030)

域	位	读写	复位值	描述
tx_lockup_detected_mask	31	R0	0x0	发送方向禁闭检测中断屏蔽
rx_lockup_detected_mask	30	R0	0x0	接收方向禁闭检测中断屏蔽
tsu_timer_comparison_mask	29	R0	0x0	TSU 计时器比较中断屏蔽
wol_event_received_mask	28	R0	0x0	网络唤醒事件接收中断屏蔽
rx_lpi_indication_mask	27	R0	0x0	接收方向低功耗指示中断屏蔽
tsu_seconds_register_increment	26	R0	0x0	TSU 计数秒级寄存器增长中断屏蔽
ptp_pdelay_resp_frame_transmitted	25	R0	0x0	PTP 延时反馈报文传输中断屏蔽
ptp_pdelay_req_frame_transmitted	24	R0	0x0	PTP 延时请求报文传输中断屏蔽
ptp_pdelay_resp_frame_received	23	R0	0x0	PTP 延时反馈报文接收中断屏蔽
ptp_pdelay_req_frame_received	22	R0	0x0	PTP 延时请求报文接收中断屏蔽
ptp_sync_frame_transmitted	21	R0	0x0	PTP 同步报文传输中断屏蔽
ptp_delay_req_frame_transmitted	20	R0	0x0	PTP 延时请求报文传输中断屏蔽
ptp_sync_frame_received	19	R0	0x0	PTP 同步报文接收中断屏蔽
ptp_delay_req_frame_received	18	R0	0x0	PTP 延时请求报文接收中断屏蔽
pcs_link_partner_page_received	17	R0	0x0	PCS 连接对端页接收中断屏蔽
pcs_auto_negotiation_complete_mask	16	R0	0x0	PCS 自协商完成中断屏蔽
external_mask	15	R0	0x0	外部中断屏蔽
pause_frame_transmitted_mask	14	R0	0x0	暂停帧传输中断屏蔽

域	位	读写	复位值	描述
pause_time_zero_mask	13	R0	0x0	暂停时间值为 0 中断屏蔽
pause_frame_with_non_zero_pause_quantum_mask	12	R0	0x0	非零值暂停帧中断屏蔽
resp_not_ok_mask	11	R0	0x0	bresp/hresp 信号没准备好中断屏蔽
receive_overrun_mask	10	R0	0x0	接收超时中断屏蔽
link_change_mask	9	R0	0x0	连接状态改变中断屏蔽
usxgmi_mask	8	R0	0x0	USXGMI 接口中断屏蔽
transmit_complete_mask	7	R0	0x0	发送完成中断屏蔽
transmit_frame_corruption_due_to_amba_error_mask	6	R0	0x0	由于 AXI 总线错误导致报文错误中断屏蔽
retry_limit_exceeded_or_late_collision_mask	5	R0	0x0	超出重试限制或延时冲突中断屏蔽
transmit_buffer_under_run_mask	4	R0	0x0	发送缓存欠载中断屏蔽
transmit_used_bit_read_mask	3	R0	0x0	发送 used bit 读中断屏蔽
receive_used_bit_read_mask	2	R0	0x0	接收 used bit 读中断屏蔽
receive_complete_mask	1	R0	0x0	接收完成中断屏蔽
management_done_mask	0	R0	0x0	mdio 接口完成中断屏蔽

5.3.3.13 phy_management (0x0034)

域	位	读写	复位值	描述
write1	30	RW	0x0	值为 1 时 mdio 传输 Clause 22 frame；值为 0 时 mdio 传输 Clause 45 frame。
operation	29:28	RW	0x0	当 mdio 传输的是 Clause 45 frame 时, 值为 2'h0 代表传地址, 值为 2'h1 代表执行写操作, 值为 2'h2 代表读取后增量, 值为 2'h3 代表读报文。当 mdio 传输的是 Clause 22 frame 时, 值为 2'h2 代表读操作, 值为 2'h1 代表写操作。
phy_address	27:23	RW	0x0	phy 地址
register_address	22:18	RW	0x0	phy 内部寄存器地址
phy_write_read_data	15:0	RW	0x0	mdio 读/写数据

5.3.3.14 pause_time (0x0038)

域	位	读写	复位值	描述
reserved	31:16	R0	0x0	保留字段
quantum	15:0	R0	0x0	接收 pause 量寄存器, 保存当前接收 pause 帧量级寄存器的值。

5.3.3.15 tx_pause_quantum (0x003C)

域	位	读写	复位值	描述
quantum_p1	31:16	Rw	0xFFFF	发送 pause 量-使用优先级 1 的发送 pause 帧的

域	位	读写	复位值	描述
				pause 量值写入。
quantum	15:0	RW	0xFFFF	发送 pause 量-使用发送 pause 帧的 pause 量值写入。

5.3.3.16 jumbo_max_length (0x0048)

域	位	读写	复位值	描述
reserved	31:14	RO	0x0	保留
jumbo_max_length	13:0	RW	0x2800	巨型帧长度, 最大值为 16383 字节。32 字节数据位宽时最大值为 16300 字节。

5.3.3.17 hs_mac_config (0x0050)

域	位	读写	复位值	描述
preamble_in_crc_rx	8:7	RW	0x0	接收方向 preamble 参与校验计算。值为 0 时 preamble 不参与校验计算；值为 1 时 7MS bytes of preamble 参与校验计算；值为 2 时所有 preamble 参与校验计算；值为 3 时 4MS bytes of preamble 参与校验计算。
preamble_in_crc_tx	6:5	RW	0x0	发送方向 preamble 参与校验计算。值为 0 时 preamble 不参与校验计算；值为 1 时 7MS bytes of preamble 参与校验计算；值为 2 时所有 preamble 参与校验计算；值为 3 时 4MS bytes of preamble 参与校验计算。
hs_mac_speed	2:0	RW	0x04	配置 hs_mac_speed 引脚输出表示 mac 速率的值。值为 3'h0 时表示 mac 速率为 100M；值为 3'h1 时表示 mac 速率为 1G；值为 3'h2 时表示 mac 速率为 2.5G；值为 3'h3 时表示 mac 速率为 5G，值为 3'h4 时代表 mac 速率为 10G。

5.3.3.18 axi_max_pipeline (0x0054)

域	位	读写	复位值	描述
aw2b_max_pipeline	15:8	RW	0x1	axi 总线的 aw 通道 outstanding 最大值
ar2r_max_pipeline	7:0	RW	0x1	axi 总线的 ar 通道 outstanding 最大值

5.3.3.19 hash_bottom(0x0080)

域	位	读写	复位值	描述
address	31:0	RW	0x0	hash 地址的第一个 32bit 寄存器

5.3.3.20 hash_top(0x0084)

域	位	读写	复位值	描述
address	31:0	RW	0x0	hash 地址的其余 32bit 寄存器

5.3.3.21 spec_add1_bottom(0x0088)

域	位	读写	复位值	描述
address	31:0	RW	0x0	目的地址的 8 位字节 3 到 0，具体是[31:0]bit。Bit0 代表地址时多播还是单播，且对应于接收到的第一字节的最低有效位

5.3.3.22 spec_add1_top(0x008C)

域	位	读写	复位值	描述
reserved	31:17	RO	0x0	保留
Filter_type	16	RW	0x0	这个控制 bit 选择滤波是否比较接收报文的 MAC 的原地址或 MAC 的目的地址。设置为 0 时，滤波比较目的地址；设置为 1 时，滤波比较源地址
address	15:0	RW	0x0	特定地址 1。比较的目的地址/源地址的 8 位字节 5 到 4，具体是[47:32]bit

5.3.3.23 spec_add2_bottom(0x0090)

域	位	读写	复位值	描述
address	31:0	RW	0x0	目的地址的 8 位字节 3 到 0，具体是[31:0]bit。Bit0 代表地址时多播还是单播，且对应于接收到的第一字节的最低有效位

5.3.3.24 spec_add2_top(0x0094)

域	位	读写	复位值	描述
reserved	31:30	RO	0x0	保留
Filter_byte_mask	29:24	RW	0x0	置高时，不比较相应的特定地址，bit24 控制是否比较接收到的第一个字节。bit29 控制是否比较接收到的最后一个字节。
reserved	23:17	RO	0x0	保留
Filter_type	16	RW	0x0	这个控制 bit 选择滤波是否比较接收报文的 MAC 的原地址或 MAC 的目的地址。设置为 0 时，滤波比较目的地址；设置为 1 时，滤波比较源地址
address	15:0	RW	0x0	特定地址 1。比较的目的地址/源地址的 8 位字节 5 到 4，具体是[47:32]bit

5.3.3.25 spec_add3_bottom(0x0098)

域	位	读写	复位值	描述
address	31:0	RW	0x0	目的地址的 8 位字节 3 到 0，具体是[31:0]bit。Bit0 代表地址时多播还是单播，且对应于接收到的第一字节的最低有效位

5.3.3.26 spec_add3_top (0x009C)

域	位	读写	复位值	描述
reserved	31:30	R0	0x0	保留
Filter_byte_mask	29:24	RW	0x0	置高时，不比较相应的特定地址，bit24 控制是否比较接收到的第一个字节。bit29 控制是否比较接收到的最后一个字节。
reserved	23:17	R0	0x0	保留
Filter_type	16	RW	0x0	这个控制 bit 选择滤波是否比较接收报文的 MAC 的原地址或 MAC 的目的地址。设置为 0 时，滤波比较目的地址；设置为 1 时，滤波比较源地址
address	15:0	RW	0x0	特定地址 1。比较的目的地址/源地址的 8 位字节 5 到 4，具体是 [47:32]bit

5.3.3.27 spec_add4_bottom (0x00A0)

域	位	读写	复位值	描述
address	31:0	RW	0x0	目的地址的 8 位字节 3 到 0，具体是 [31:0]bit。Bit0 代表地址时多播还是单播，且对应于接收到的第一字节的最低有效位

5.3.3.28 spec_add4_top (0x00A4)

域	位	读写	复位值	描述
reserved	31:30	R0	0x0	保留
Filter_byte_mask	29:24	RW	0x0	置高时，不比较相应的特定地址，bit24 控制是否比较接收到的第一个字节。bit29 控制是否比较接收到的最后一个字节。
reserved	23:17	R0	0x0	保留
Filter_type	16	RW	0x0	这个控制 bit 选择滤波是否比较接收报文的 MAC 的源地址或 MAC 的目的地址。设置为 0 时，滤波比较目的地址；设置为 1 时，滤波比较源地址
address	15:0	RW	0x0	特定地址 1。比较的目的地址/源地址的 8 位字节 5 到 4，具体是 [47:32]bit

5.3.3.29 tsu_ptp_tx_msb_sec (0x00E8)

域	位	读写	复位值	描述
reserved	31:16	R0	0x0	保留
Timer_seconds	15:0	R0	0x0	PTP 报文发送秒数。这个寄存器随着 1588 计时器秒寄存器更新，当一个 PTP 发送事件的 SFD 通过 MII 接口时。当 MAC 识别到一个 PTP sync 或 delay_req 报文时，更新发生。寄存器更新时，置有效一个中断。

5.3.3.30 tsu_ptp_rx_msb_sec (0x00EC)

域	位	读写	复位值	描述
reserved	31:16	R0	0x0	保留
Timer_seconds	15:0	R0	0x0	PTP 报文接收秒数。这个寄存器随着 1588 计时器秒寄存器更新, 当一个 PTP 发送事件的 SFD 通过 MII 接口时。当 MAC 识别到一个 PTP sync 或 delay_req 报文时, 更新发生。寄存器更新时, 置有效一个中断。

5.3.3.31 tsu_peer_tx_msb_sec (0x00F0)

域	位	读写	复位值	描述
reserved	31:16	R0	0x0	保留
Timer_seconds	15:0	R0	0x0	PTP PEER 报文发送秒数。这个寄存器随着 1588 计时器秒寄存器更新, 当一个 PTP 发送 peer 事件的 SFD 通过 MII 接口时。当 MAC 识别到一个 PTP pdelay_req 或 pdelay_resp 报文时, 更新发生。寄存器更新时, 置有效一个中断。

5.3.3.32 tsu_peer_rx_msb_sec (0x00F4)

域	位	读写	复位值	描述
reserved	31:16	R0	0x0	保留
Timer_seconds	15:0	R0	0x0	PTP PEER 报文接收秒数。这个寄存器随着 1588 计时器秒寄存器更新, 当一个 PTP 发送 peer 事件的 SFD 通过 MII 接口时。当 MAC 识别到一个 PTP pdelay_req 或 pdelay_resp 报文时, 更新发生。寄存器更新时, 置有效一个中断。

5.3.3.33 octets_txed_bottom (0x0100)

域	位	读写	复位值	描述
count	31:0	R0	0x0	正确发送报文数低 32bit, 8 进制

5.3.3.34 octets_txed_top (0x0104)

域	位	读写	复位值	描述
count	15:0	R0	0x0	正确发送报文数高 16bit, 8 进制

5.3.3.35 frames_txed_ok (0x0108)

域	位	读写	复位值	描述
count	31:0	R0	0x0	正确发送报文数, 10 进制

5.3.3.36 broadcast_txed (0x0100)

域	位	读写	复位值	描述
count	31:0	R0	0x0	正确发送 Broadcast 报文数

5.3.3.37 multicast_txed (0x0110)

域	位	读写	复位值	描述
count	31:0	R0	0x0	正确发送 Multicast 报文数

5.3.3.38 pause_frames_txed (0x0114)

域	位	读写	复位值	描述
count	15:0	R0	0x0	正确发送 pause 报文数

5.3.3.39 frames_txed_64 (0x0118)

域	位	读写	复位值	描述
count	31:0	R0	0x0	正确发送长度为 64 字节的报文数

5.3.3.40 frames_txed_65 (0x011C)

域	位	读写	复位值	描述
count	31:0	R0	0x0	正确发送长度在 65~127 字节之间的报文数

5.3.3.41 frames_txed_128 (0x0120)

域	位	读写	复位值	描述
count	31:0	R0	0x0	正确发送长度在 128~255 字节之间的报文数

5.3.3.42 frames_txed_256 (0x0124)

域	位	读写	复位值	描述
count	31:0	R0	0x0	正确发送长度在 256~511 字节之间的报文数

5.3.3.43 frames_txed_512 (0x0128)

域	位	读写	复位值	描述
count	31:0	R0	0x0	正确发送长度在 512~1023 字节之间的报文数

5.3.3.44 frames_txed_1024 (0x012C)

域	位	读写	复位值	描述
count	31:0	R0	0x0	正确发送长度在 1024~1518 字节之间的报文数

5.3.3.45 frames_txed_1519 (0x0130)

域	位	读写	复位值	描述
count	31:0	R0	0x0	正确发送长度在 1519~16k 字节之间的报文数

5.3.3.46 tx_underruns (0x0134)

域	位	读写	复位值	描述
reserved	31:10	R0	0x0	保留
count	9:0	R0	0x0	传输欠载。计数因为传输欠载没发出报文数量。读清除。

5.3.3.47 single_collisions (0x0138)

域	位	读写	复位值	描述
reserved	31:18	R0	0x0	保留
count	17:0	R0	0x0	单独冲突报文, 计数在成功发送之前遇到单独冲突报文数量。读清除。

5.3.3.48 multiple_collisions (0x013C)

域	位	读写	复位值	描述
reserved	31:18	R0	0x0	保留
count	17:0	R0	0x0	多冲突报文, 计数在成功发送之前遇到 2 到 15 个冲突报文数量。读清除。

5.3.3.49 excessive_collisions (0x0140)

域	位	读写	复位值	描述
reserved	31:10	R0	0x0	保留
count	9:0	R0	0x0	超载冲突, 计数因为遇到 16 个冲突发送失败报文数量。读清除。

5.3.3.50 late_collisions (0x0144)

域	位	读写	复位值	描述
reserved	31:10	R0	0x0	保留
count	9:0	R0	0x0	延迟冲突, 计数在间隙时间后出现的延迟冲突数量。在 100/10 兆模式时, 在一个冲突或延迟冲突出现时, 计数两次; 在千兆模式时, 一个延迟冲突导致传输放弃, 单冲突和多冲突寄存器不更新。读清除。

5.3.3.51 deferred_frames (0x0148)

域	位	读写	复位值	描述
reserved	31:18	R0	0x0	保留
count	17:0	R0	0x0	延迟传输报文。在传输时，如果第一次申请时，载波检测有效，计数此时传输报文数量。涉及遇到冲突或者运行不足的报文时，不计数。读清除。

5.3.3.52 crc_errors (0x014C)

域	位	读写	复位值	描述
reserved	31:10	R0	0x0	保留
count	9:0	R0	0x0	载波检测错误。传输时，没发现载波检测，或者在传输一个没遇到冲突的报文时，载波检测先有效再无效时计数传输的报文数量。仅在半双工模式时有效。读清除。

5.3.3.53 octets_rxed_bottom (0x0150)

域	位	读写	复位值	描述
count	31:0	R0	0x0	正确接收报文数低 32bit，8 进制

5.3.3.54 octets_rxed_top (0x0154)

域	位	读写	复位值	描述
count	15:0	R0	0x0	正确接收报文数高 16bit，8 进制

5.3.3.55 frames_rxed_ok (0x0158)

域	位	读写	复位值	描述
count	31:0	R0	0x0	正确接收报文数，10 进制

5.3.3.56 broadcast_rxed (0x015C)

域	位	读写	复位值	描述
count	31:0	R0	0x0	正确接收 Broadcast 报文数

5.3.3.57 multicast_rxed (0x0160)

域	位	读写	复位值	描述
count	31:0	R0	0x0	正确接收 Multicast 报文数

5.3.3.58 pause_frames_rxed (0x0164)

域	位	读写	复位值	描述
count	15:0	R0	0x0	正确接收 pause 报文数

5.3.3.59 frames_rxed_64 (0x0168)

域	位	读写	复位值	描述
count	31:0	R0	0x0	正确接收长度为 64 字节的报文数

5.3.3.60 frames_rxed_65 (0x016C)

域	位	读写	复位值	描述
count	31:0	R0	0x0	正确接收长度在 65~127 字节之间的报文数

5.3.3.61 frames_rxed_128 (0x0170)

域	位	读写	复位值	描述
count	31:0	R0	0x0	正确接收长度在 128~255 字节之间的报文数

5.3.3.62 frames_rxed_256 (0x0174)

域	位	读写	复位值	描述
count	31:0	R0	0x0	正确接收长度在 255~511 字节之间的报文数

5.3.3.63 frames_rxed_512 (0x0178)

域	位	读写	复位值	描述
count	31:0	R0	0x0	正确接收长度在 512~1023 字节之间的报文数

5.3.3.64 frames_rxed_1024 (0x017C)

域	位	读写	复位值	描述
count	31:0	R0	0x0	正确接收长度在 1024~1518 字节之间的报文数

5.3.3.65 frames_rxed_1519 (0x0180)

域	位	读写	复位值	描述
count	31:0	R0	0x0	正确接收长度在 1519~16k 字节之间的报文数

5.3.3.66 undersize_frames (0x0184)

域	位	读写	复位值	描述
reserved	31:10	R0	0x0	保留

域	位	读写	复位值	描述
count	9:0	R0	0x0	接收报文长度过小报文计数，接收的报文长度小于 64 字节报文数量，接收的报文没有 CRC 错误或对齐错误。读清除。

5.3.3.67 Excessive_rx_length(0x0188)

域	位	读写	复位值	描述
reserved	31:10	R0	0x0	保留
count	9:0	R0	0x0	接收报文长度过大报文计数，接收的报文长度大于 1518 字节(如果配置了 network_config 寄存器的 bit8 和 bit3)报文数量，接收的报文没有 CRC 错误、对齐错误或接收特征错误。读清除。

5.3.3.68 rx_jabbers(0x018C)

域	位	读写	复位值	描述
reserved	31:10	R0	0x0	保留
count	9:0	R0	0x0	接收的报文长度大于 1518 字节(如果配置了 network_config 寄存器的 bit8 和 bit3)报文数量，接收的报文没有 CRC 错误、对齐错误或接收特征错误。读清除。

5.3.3.69 rx_fcs_errors(0x0190)

域	位	读写	复位值	描述
reserved	31:10	R0	0x0	保留
count	9:0	R0	0x0	报文检查序列错误。报文长度为整数字节数的报文数量，接收的报文有 CRC 错误且长度在 64 字节和 1518 字节之间的(如果配置了 network_config 寄存器的 bit8 和 bit3)报文。在出现特征错误、报文长度为有效长度且字节数为整数时，这个计数也会增长。当报文有 FCS 错误时，计数增长，不考虑是否复制到存储里，在使能 network_config 寄存器的 bit26 情况下。读清除。

5.3.3.70 rx_length_errors(0x0194)

域	位	读写	复位值	描述
reserved	31:10	R0	0x0	保留
count	9:0	R0	0x0	报文长度小于从长度字段里提取的报文长度的报文数量。计数增长条件为：报文长度小于 0x0600；报文长度小于从长度字段里提取的报文长度；检查使能。通过配置 network_config 寄存器的 bit16。读清除。

5.3.3.71 rx_symbol_errors(0x0198)

域	位	读写	复位值	描述
reserved	31:10	R0	0x0	保留
count	9:0	R0	0x0	在接收期间, rx_er 有效时接收的报文数量。在百兆/十兆模式时, 特征错误计数不考虑报文长度检查。千兆模式, 为了计数特征错误, 报文要满足时隙时间需求。读清除。

5.3.3.72 rx_alignment_errors(0x019C)

域	位	读写	复位值	描述
reserved	31:10	R0	0x0	保留
count	9:0	R0	0x0	当报文长度不是整数字节、当报文长度截断成一个整数字节且长度在 64 字节到 1518 字节(network_config 寄存器的 bit8 和 bit3 被配置)之间时, 报文有 CRC 错误。计数增长。在遇到特征错误; 报文有有效长度; 且报文的有效长度不是整数字节时, 计数也会增长。读清除。

5.3.3.73 rx_resource_errors(0x01A0)

域	位	读写	复位值	描述
reserved	31:18	R0	0x0	保留
count	17:0	R0	0x0	接收源错误。接收缓存描述符读到 used bit 拉高的报文计数。这个寄存器可以用作检测软件处理和释放接收描述符效率的方式。在一个理想的系统里, 这个计数不会增长。读清除。

5.3.3.74 rx_overruns(0x01A4)

域	位	读写	复位值	描述
reserved	31:10	R0	0x0	保留
count	9:0	R0	0x0	由于接收溢出, 被识别到地址, 但没复制到存储里的报文数量。读清除。

5.3.3.75 rx_ip_ck_errors(0x01A8)

域	位	读写	复位值	描述
reserved	31:8	R0	0x0	保留
count	7:0	R0	0x0	IP 头校验错误。由于 IP 头校验错误被丢弃的报文数量。这些报文长度在 64 字节和 1518 字节(如果配置了 network_config 寄存器的 bit8 和 bit3)之间, 且没有 CRC 错误、没有对齐错误、没有特征错误。读清除。

5.3.3.76 rx_tcp_ck_errors (0x01AC)

域	位	读写	复位值	描述
reserved	31:8	R0	0x0	保留
count	7:0	R0	0x0	TCP 头校验错误。由于 TCP 头校验错误被丢弃的报文数量。这些报文长度在 64 字节和 1518 字节 (如果配置了 network_config 寄存器的 bit8 和 bit3) 之间, 且没有 CRC 错误、没有对齐错误、没有特征错误。读清除。

5.3.3.77 rx_dup_ck_errors (0x01B0)

域	位	读写	复位值	描述
reserved	31:8	R0	0x0	保留
count	7:0	R0	0x0	UDP 头校验错误。由于 UDP 头校验错误被丢弃的报文数量。这些报文长度在 64 字节和 1518 字节 (如果配置了 network_config 寄存器的 bit8 和 bit3) 之间, 且没有 CRC 错误、没有对齐错误、没有特征错误。读清除。

5.3.3.78 tsu_timer_inc_sub_nsec (0x01BC)

域	位	读写	复位值	描述
sub_ns_incr_lsb	31:24	RW	0x0	sub_ns 的低 8bit 值, 1588 计时器每个时钟增长的量
reserved	23:16	R0	0x0	保留
sub_ns_incr	15:0	RW	0x0	sub_ns 的 16bit 值, 1588 计时器每个时钟增长的量。24bit 控制的纳秒精度, 可达到 5.86E-17 秒的精度。

5.3.3.79 tsu_timer_msb_sec (0x01C0)

域	位	读写	复位值	描述
reserved	31:16	R0	0x0	保留
timer	15:0	RW	0x0	tsu 计时器值。重要的秒计时器计数。这个寄存器可写。在 1588 纳秒计数器计到 1 秒时, 这个 48bit 的计数器增长 1。这个寄存器会增长或减少在操作计数器调整寄存器 (如果从 0 开始减少, 这个 48bit 组合的计数会变成 0xfffffffffff) 时。注意: 这个寄存器时可使用的仅当低 32bit 寄存器被写入。这是为了确保 48bit 的秒值是单独更新的。

5.3.3.80 tsu_timer_sec (0x01D0)

域	位	读写	复位值	描述
timer	31:0	RW	0x0	T1588 计时器秒寄存器。TSU 计时器的值。最不重要的 32bit 秒计时器计数。这个寄存器可写。在 1588 纳秒计数器计到 1 秒时, 这个 48bit 的计数器增长 1。这个

域	位	读写	复位值	描述
				寄存器会增长或减少在操作计数器调整寄存器(如果从 0 开始减少, 这个 48bit 组合的计数会变成 0xffffffffffff)时。

5.3.3.81 tsu_timer_nsec (0x01D4)

域	位	读写	复位值	描述
reserved	31:30	RO	0x0	保留
timer	29:0	RW	0x0	纳秒计时器计数。这个寄存器可写。通过操作 1588 计时器调整寄存器可调整这个寄存器。随着 1588 计时器增长寄存器每个时钟(如果这个寄存器值接近 0, 且对于计时器调整寄存器的写操作导致减少, 这个减少为秒寄存器减少如果必须的且纳秒寄存器的值会变为 9999999xx) 增长的值进行增长。

5.3.3.82 tsu_timer_adjust (0x01D8)

域	位	读写	复位值	描述
Add_subtract	31	WO	0x0	写 1 减去 1588 计时器值; 写 0 加上 1588 计时器值;
reserved	30	RO	0x0	保留
sub_ns_incr	29:0	WO	0x0	计时器增长值。1588 计时器纳秒寄存器纳秒增长或减少数量。如果需要, 1588 秒寄存器会增长或减少。

5.3.3.83 tsu_timer_incr (0x01DC)

域	位	读写	复位值	描述
reserved	31:24	RO	0x0	保留
num_incs	23:16	RW	0x0	替代增量使用后增长的数量
Alt_ns_incr	15:8	RW	0x0	1588 计时器纳秒寄存器的纳秒替代计数, 随着每拍时钟增长
ns_increment	7:0	RW	0x0	1588 计时器纳秒寄存器的纳秒计数, 随着每拍时钟增长。这是 32bit 的计数器增长计数最重要的 8bit。tsu_timer_inc_sub_nsec 寄存器包含增长最不重要的 24bit。

5.3.3.84 tsu_ptp_tx_sec (0x01E0)

域	位	读写	复位值	描述
timer	31:0	RO	0x0	PTP 报文发送秒数[31:0]bit。这个寄存器随着 1588 计时器秒寄存器更新, 当一个 PTP 发送事件的 SFD 通过 MII 接口时。当 MAC 识别到一个 PTP sync 或 delay_req 报文时, 更新发生。寄存器更新时, 置有效一个中断。

5.3.3.85 tsu_ptp_tx_nsec (0x01E4)

域	位	读写	复位值	描述
timer	31:0	RO	0x0	PTP 报文发送纳秒数。这个寄存器随着 1588 计时器纳秒寄存器更新，当一个 PTP 发送事件的 SFD 通过 MII 接口时。当 MAC 识别到一个 PTP sync 或 delay_req 报文时，更新发生。寄存器更新时，置有效一个中断。

5.3.3.86 tsu_ptp_rx_sec (0x01E8)

域	位	读写	复位值	描述
timer	31:0	RO	0x0	PTP 报文接收秒数 [31:0]bit。这个寄存器随着 1588 计时器纳秒寄存器更新，当一个 PTP 发送事件的 SFD 通过 MII 接口时。当 MAC 识别到一个 PTP sync 或 delay_req 报文时，更新发生。寄存器更新时，置有效一个中断。

5.3.3.87 tsu_ptp_rx_sec (0x01EC)

域	位	读写	复位值	描述
reserved	31:30	RO	0x0	保留
timer	29:0	RO	0x0	PTP 报文接收纳秒数。这个寄存器随着 1588 计时器纳秒寄存器更新，当一个 PTP 发送事件的 SFD 通过 MII 接口时。当 MAC 识别到一个 PTP sync 或 delay_req 报文时，更新发生。寄存器更新时，置有效一个中断。

5.3.3.88 tsu_peer_tx_sec (0x01F0)

域	位	读写	复位值	描述
timer	31:0	RO	0x0	PTP PEER 报文接收秒数 [31:0]bit。这个寄存器随着 1588 计时器秒寄存器更新，当一个 PTP 发送 peer 事件的 SFD 通过 MII 接口时。当 MAC 识别到一个 PTP pdelay_req 或 pdelay_resp 报文时，更新发生。寄存器更新时，置有效一个中断。

5.3.3.89 tsu_peer_tx_nsec (0x01F4)

域	位	读写	复位值	描述
reserved	31:30	RO	0x0	保留
timer	29:0	RO	0x0	PTP PEER 报文发送纳秒数。这个寄存器随着 1588 计时器纳秒寄存器更新，当一个 PTP 发送 peer 事件的 SFD 通过 MII 接口时。当 MAC 识别到一个 PTP pdelay_req 或 pdelay_resp 报文时，更新发生。寄存器更新时，置有效一个中断。

5.3.3.90 tsu_peer_rx_sec (0x01F8)

域	位	读写	复位值	描述
timer	31:0	R0	0x0	PTP PEER 报文接收秒数 [31:0]bit。这个寄存器随着 1588 计时器秒寄存器更新, 当一个 PTP 接收 peer 事件的 SFD 通过 MII 接口时。当 MAC 识别到一个 PTP pdelay_req 或 pdelay_resp 报文时, 更新发生。寄存器更新时, 置有效一个中断。

5.3.3.91 tsu_peer_rx_nsec (0x01FC)

域	位	读写	复位值	描述
reserved	31:30	R0	0x0	保留
timer	29:0	R0	0x0	PTP PEER 报文接收纳秒数。这个寄存器随着 1588 计时器纳秒寄存器更新, 当一个 PTP 接收 peer 事件的 SFD 通过 MII 接口时。当 MAC 识别到一个 PTP pdelay_req 或 pdelay_resp 报文时, 更新发生。寄存器更新时, 置有效一个中断。

5.3.3.92 pcs_control (0x0200)

域	位	读写	复位值	描述
reserved	31:16	R0	0x0	保留
pcs_software_reset	15	RW	0x1	PCS 软复位: 写 1 复位; 写 0 复位释放。
loopback_mode	14	RW	0x0	控制 phy 环回: 写 1 使能环回; 写 0 去使能环回。
speed_select_bit_1	13	R0	0x0	PCS 运行速率指示: 值为 0 表示 PCS 运行速率为千兆; 值为 1 表示 PCS 运行速率为非千兆。
enable_auto_neg	12	RW	0x1	使能自协商: 写 1 使能自协商; 写 0 去使能自协商。
reserved	11:10	R0	0x0	保留
restart_auto_neg	9	RW	0x0	重新开始自协商: 写 1 重新开始自协商; 写 0 不重新开始自协商。
mac_duplex_state	8	R0	0x0	MAC 的双工模式: 值为 1 时: MAC 为全双工模式; 值为 0 时: MAC 为半双工模式。
collision_test	7	RW	0x0	冲突测试: 写 1, PCS 在发送方向产生冲突; 写 0, PCS 不产生冲突。
speed_select_bit_0	6	R0	0x1	PCS 运行速率指示: 值为 0 表示 PCS 运行速率为千兆; 值为 1 表示 PCS 运行速率为非千兆。
reserved	5:0	R0	0x0	保留

5.3.3.93 int_q1_status (0x0400)

域	位	读写	复位值	描述
reserved	31:12	R0	0x0	保留
resp_not_ok	11	R0	0x0	hresp 没准备好

域	位	读写	复位值	描述
reserved	10:8	R0	0x0	保留
transmit_complete	7	R0	0x0	队列 1 发送完成中断。1 表示一个报文发送完成。
amba_error	6	R0	0x0	由于缓存不足导致发送报文错误中断
try_limit_exceeded_or_late_collision	5	R0	0x0	超出重来限制或延迟冲突中断
reserved	4:3	R0	0x0	保留
rx_used_bit_read	2	R0	0x0	接收方向，读 used bit 中断。1 表示接收方向队列 1 读到 used bit。
receive_complete	1	R0	0x0	队列 1 接收完成中断。1 表示一个报文接收完成。
reserved	0	R0	0x0	保留

5.3.3.94 int_q2_status (0x0404)

域	位	读写	复位值	描述
reserved	31:12	R0	0x0	保留
resp_not_ok	11	R0	0x0	hresp 没准备好
reserved	10:8	R0	0x0	保留
transmit_complete	7	R0	0x0	队列 2 发送完成中断。1 表示一个报文发送完成。
amba_error	6	R0	0x0	由于缓存不足导致发送报文错误中断
try_limit_exceeded_or_late_collision	5	R0	0x0	超出重来限制或延迟冲突中断
reserved	4:3	R0	0x0	保留
rx_used_bit_read	2	R0	0x0	接收方向，读 used bit 中断。1 表示接收方向队列 2 读到 used bit。
receive_complete	1	R0	0x0	队列 2 接收完成中断。1 表示一个报文接收完成。
reserved	0	R0	0x0	保留

5.3.3.95 int_q3_status (0x0408)

域	位	读写	复位值	描述
reserved	31:12	R0	0x0	保留
resp_not_ok	11	R0	0x0	hresp 没准备好，hresp 是 AHB 总线的信号。
reserved	10:8	R0	0x0	保留
transmit_complete	7	R0	0x0	队列 3 发送完成中断。1 表示一个报文发送完成。
amba_error	6	R0	0x0	由于缓存不足导致发送报文错误中断
try_limit_exceeded_or_late_collision	5	R0	0x0	超出重来限制或延迟冲突中断
reserved	4:3	R0	0x0	保留
rx_used_bit_read	2	R0	0x0	接收方向，读 used bit 中断。1 表示接收方向队列 3 读到 used bit。
receive_complete	1	R0	0x0	队列 3 接收完成中断。1 表示一个报文接收完成。
reserved	0	R0	0x0	保留

5.3.3.96 int_q4_status (0x040C)

该寄存器只 mac0 有。

域	位	读写	复位值	描述
reserved	31:12	R0	0x0	保留
resp_not_ok	11	R0	0x0	hresp 没准备好
reserved	10:8	R0	0x0	保留
transmit_complete	7	R0	0x0	队列 4 发送完成中断。1 表示一个报文发送完成。
amba_error	6	R0	0x0	由于缓存不足导致发送报文错误中断
try_limit_exceeded_or_late_collision	5	R0	0x0	超出重来限制或延迟冲突中断
reserved	4:3	R0	0x0	保留
rx_used_bit_read	2	R0	0x0	接收方向，读 used bit 中断。1 表示接收方向队列 4 读到 used bit。
receive_complete	1	R0	0x0	队列 4 接收完成中断。1 表示一个报文接收完成。
reserved	0	R0	0x0	保留

5.3.3.97 int_q5_status (0x0410)

该寄存器只 mac0 有。

域	位	读写	复位值	描述
reserved	31:12	R0	0x0	保留
resp_not_ok	11	R0	0x0	hresp 没准备好
reserved	10:8	R0	0x0	保留
transmit_complete	7	R0	0x0	队列 5 发送完成中断。1 表示一个报文发送完成。
amba_error	6	R0	0x0	由于缓存不足导致发送报文错误中断
try_limit_exceeded_or_late_collision	5	R0	0x0	超出重来限制或延迟冲突中断
reserved	4:3	R0	0x0	保留
rx_used_bit_read	2	R0	0x0	接收方向，读 used bit 中断。1 表示接收方向队列 1 读到 used bit。
receive_complete	1	R0	0x0	队列 5 接收完成中断。1 表示一个报文接收完成。
reserved	0	R0	0x0	保留

5.3.3.98 int_q6_status (0x0414)

该寄存器只 mac0 有。

域	位	读写	复位值	描述
reserved	31:12	R0	0x0	保留
resp_not_ok	11	R0	0x0	hresp 没准备好
reserved	10:8	R0	0x0	保留
transmit_complete	7	R0	0x0	队列 6 发送完成中断。1 表示一个报文发送完成。
amba_error	6	R0	0x0	由于缓存不足导致发送报文错误中断

域	位	读写	复位值	描述
try_limit_exceeded_or_late_collision	5	R0	0x0	超出重来限制或延迟冲突中断
reserved	4:3	R0	0x0	保留
rx_used_bit_read	2	R0	0x0	接收方向, 读 used bit 中断。1 表示接收方向队列 6 读到 used bit。
receive_complete	1	R0	0x0	队列 6 接收完成中断。1 表示一个报文接收完成。
reserved	0	R0	0x0	保留

5.3.3.99 int_q7_status (0x0418)

该寄存器只 mac0 有。

域	位	读写	复位值	描述
reserved	31:12	R0	0x0	保留
resp_not_ok	11	R0	0x0	hresp 没准备好
reserved	10:8	R0	0x0	保留
transmit_complete	7	R0	0x0	队列 7 发送完成中断。1 表示一个报文发送完成。
amba_error	6	R0	0x0	由于缓存不足导致发送报文错误中断
try_limit_exceeded_or_late_collision	5	R0	0x0	超出重来限制或延迟冲突中断
reserved	4:3	R0	0x0	保留
rx_used_bit_read	2	R0	0x0	接收方向, 读 used bit 中断。1 表示接收方向队列 7 读到 used bit。
receive_complete	1	R0	0x0	队列 7 接收完成中断。1 表示一个报文接收完成。
reserved	0	R0	0x0	保留

5.3.3.100 transmit_q1_ptr (0x0440)

域	位	读写	复位值	描述
dma_tx_q_ptr	31:2	RW	0x0	q1 队列的发送描述符缓存地址
dma_tx_dis_q	0	RW	0x1	值为 1 时 q1 队列发送无效; 值为 0 时 q1 队列发送有效。

5.3.3.101 transmit_q2_ptr (0x0444)

域	位	读写	复位值	描述
dma_tx_q_ptr	31:2	RW	0x0	q2 队列的发送描述符缓存地址
dma_tx_dis_q	0	RW	0x1	值为 1 时 q2 队列发送无效; 值为 0 时 q2 队列发送有效。

5.3.3.102 transmit_q3_ptr (0x0448)

域	位	读写	复位值	描述
dma_tx_q_ptr	31:2	RW	0x0	q3 队列的发送描述符缓存地址
dma_tx_dis_q	0	RW	0x1	值为 1 时 q3 队列发送无效; 值为 0 时 q3 队列发送有效。

5.3.3.103 transmit_q4_ptr (0x044C)

该寄存器只 mac0 有。

域	位	读写	复位值	描述
dma_tx_q_ptr	31:2	RW	0x0	q4 队列的发送描述符缓存地址
dma_tx_dis_q	0	RW	0x1	值为 1 时 q4 队列发送无效;值为 0 时 q4 队列发送有效。

5.3.3.104 transmit_q5_ptr (0x0450)

该寄存器只 mac0 有。

域	位	读写	复位值	描述
dma_tx_q_ptr	31:2	RW	0x0	q5 队列的发送描述符缓存地址
dma_tx_dis_q	0	RW	0x1	值为 1 时 q5 队列发送无效;值为 0 时 q5 队列发送有效。

5.3.3.105 transmit_q6_ptr (0x0454)

该寄存器只 mac0 有。

域	位	读写	复位值	描述
dma_tx_q_ptr	31:2	RW	0x0	q6 队列的发送描述符缓存地址
dma_tx_dis_q	0	RW	0x1	值为 1 时 q6 队列发送无效;值为 0 时 q6 队列发送有效。

5.3.3.106 transmit_q7_ptr (0x0458)

该寄存器只 mac0 有。

域	位	读写	复位值	描述
dma_tx_q_ptr	31:2	RW	0x0	q7 队列的发送描述符缓存地址
dma_tx_dis_q	0	RW	0x1	值为 1 时 q7 队列发送无效;值为 0 时 q7 队列发送有效。

5.3.3.107 receive_q1_ptr (0x0480)

域	位	读写	复位值	描述
dma_rx_q_ptr	31:2	RW	0x0	q1 队列的接收描述符缓存地址
dma_rx_dis_q	0	RW	0x1	值为 1 时 q1 队列接收无效;值为 0 时 q1 队列接收有效。

5.3.3.108 receive_q2_ptr (0x0484)

域	位	读写	复位值	描述
dma_rx_q_ptr	31:2	RW	0x0	q2 队列的接收描述符缓存地址
dma_rx_dis_q	0	RW	0x1	值为 1 时 q2 队列接收无效;值为 0 时 q2 队列接收有效。

5.3.3.109 receive_q3_ptr (0x0488)

域	位	读写	复位值	描述
dma_rx_q_ptr	31:2	RW	0x0	q3 队列的接收描述符缓存地址
dma_rx_dis_q	0	RW	0x1	值为 1 时 q3 队列接收无效;值为 0 时 q3 队列接收有效。

5.3.3.110 receive_q4_ptr (0x048C)

该寄存器只 mac0 有。

域	位	读写	复位值	描述
dma_rx_q_ptr	31:2	RW	0x0	q4 队列的接收描述符缓存地址
dma_rx_dis_q	0	RW	0x1	值为 1 时 q4 队列接收无效;值为 0 时 q4 队列接收有效。

5.3.3.111 receive_q5_ptr (0x0490)

该寄存器只 mac0 有。

域	位	读写	复位值	描述
dma_rx_q_ptr	31:2	RW	0x0	q5 队列的接收描述符缓存地址
dma_rx_dis_q	0	RW	0x1	值为 1 时 q5 队列接收无效;值为 0 时 q5 队列接收有效。

5.3.3.112 receive_q6_ptr (0x0494)

该寄存器只 mac0 有。

域	位	读写	复位值	描述
dma_rx_q_ptr	31:2	RW	0x0	q6 队列的接收描述符缓存地址
dma_rx_dis_q	0	RW	0x1	值为 1 时 q6 队列接收无效;值为 0 时 q6 队列接收有效。

5.3.3.113 receive_q7_ptr (0x0498)

该寄存器只 mac0 有。

域	位	读写	复位值	描述
dma_rx_q_ptr	31:2	RW	0x0	q7 队列的接收描述符缓存地址
dma_rx_dis_q	0	RW	0x1	值为 1 时 q7 队列接收无效;值为 0 时 q7 队列接收有效。

5.3.3.114 upper_tx_q_base_addr (0x04C8)

域	位	读写	复位值	描述
upper_tx_q_base_addr	31:0	RW	0x0	发送缓存描述符队列基地址的高 32 位, 启用 64 位寻址时使用。

5.3.3.115 tx_bd_control (0x04C0)

域	位	读写	复位值	描述
reserved	31:6	R0	0x0	保留
tx_bd_ts_mode	5:4	RW	0x0	发送描述符时戳插入模式。当扩展缓存描述符模式使能时,发送缓存描述符word1的bit23:值为'b00时,bit23一直为0;值为'b01时,仅当PTP事件报文时bit23为高;值为'b10时,仅当所有PTP报文时bit23为高;值为'b11时,bit23一直为1。
reserved	3:0	R0	0x0	保留

5.3.3.116 rx_bd_control (0x04D0)

域	位	读写	复位值	描述
reserved	31:6	R0	0x0	保留
rx_bd_ts_mode	5:4	RW	0x0	接收描述符时戳插入模式。值为'b00时,TS插入去使能;值为'b01时,仅当PTP事件报文时TS插入使能;值为'b10时,仅当所有PTP报文时TS插入使能;值为'b11时,所有报文TS插入使能。
reserved	3:0	R0	0x0	保留

5.3.3.117 upper_rx_q_base_addr (0x04D4)

域	位	读写	复位值	描述
upper_rx_q_base_addr	31:0	RW	0x0	接收缓存描述符队列基地址的高32位,启用64位寻址时使用。

5.3.3.118 tx_q_seg_alloc_q_lower (0x05A0)

域	位	读写	复位值	描述
reserved	31	R0	0x0	保留
segment_alloc_q7	30:28	RW	0x0	分配给q7队列的空间。以log2计算。例如:配置为2会得到4个分段,最大可分配16个分段。每个分段大小为2KB。
reserved	27	R0	0x0	保留
segment_alloc_q6	26:24	RW	0x0	分配给q6队列的空间。以log2计算。例如:配置为2会得到4个分段,最大可分配16个分段。每个分段大小为2KB。
reserved	23	R0	0x0	保留
segment_alloc_q5	22:20	RW	0x0	分配给q5队列的空间。以log2计算。例如:配置为2会得到4个分段,最大可分配16个分段。每个分段大小为2KB。
reserved	19	R0	0x0	保留
segment_alloc_q4	18:16	RW	0x0	分配给q4队列的空间。以log2计算。例如:配置

域	位	读写	复位值	描述
				为 2 会得到 4 个分段，最大可分配 16 个分段。每个分段大小为 2KB。
reserved	15	RO	0x0	保留
segment_alloc_q3	14:12	RW	0x0	分配给 q3 队列的空间。以 log2 计算。例如：配置为 2 会得到 4 个分段，最大可分配 16 个分段。每个分段大小为 2KB。
reserved	11	RO	0x0	保留
segment_alloc_q2	10:8	RW	0x0	分配给 q2 队列的空间。以 log2 计算。例如：配置为 2 会得到 4 个分段，最大可分配 16 个分段。每个分段大小为 2KB。
reserved	7	RO	0x0	保留
segment_alloc_q1	6:4	RW	0x0	分配给 q1 队列的空间。以 log2 计算。例如：配置为 2 会得到 4 个分段，最大可分配 16 个分段。每个分段大小为 2KB。
reserved	3	RO	0x0	保留
segment_alloc_q0	2:0	RW	0x0	分配给 q0 队列的空间。以 log2 计算。例如：配置为 2 会得到 4 个分段，最大可分配 16 个分段。每个分段大小为 2KB。

5.3.3.119 int_qx_enable (0x0600+0x4*(x-1))

域	位	读写	复位值	描述
reserved	31:12	RO	0x0	保留
enable_resp_not_ok_interrupt	11	WO	0x0	使能 bresp/hresp 信号没准备好中断
reserved	10:8	RO	0x0	保留
enable_transmit_complete_interrupt	7	WO	0x0	使能发送完成中断
enable_transmit_frame_corruption_due_to_amba_error_interrupt	6	WO	0x0	使能由于 AXI 总线错误导致报文错误中断
enable_retry_limit_exceeded_or_late_collision_interrupt	5	WO	0x0	使能超出重试限制或延时冲突中断
reserved	4:3	RO	0x0	保留
enable_rx_used_bit_read_interrupt	2	WO	0x0	使能接收 used bit 读中断
enable_receive_complete_interrupt	1	WO	0x0	使能接收完成中断
reserved	0	RO	0x0	保留

5.3.3.120 int_qx_disable (0x0620+0x4*(x-1))

域	位	读写	复位值	描述
reserved	31:12	RO	0x0	保留
disable_resp_not_ok_interrupt	11	WO	0x0	去使能 bresp/hresp 信号没准备好中断
reserved	10:8	RO	0x0	保留
disable_transmit_complete_interrupt	7	WO	0x0	去使能发送完成中断

域	位	读写	复位值	描述
disable_transmit_frame_corruption_due_to_amba_error_interrupt	6	WO	0x0	去使能由于 AXI 总线错误导致报文错误中断
disable_retry_limit_exceeded_or_late_collision_interrupt	5	WO	0x0	去使能超出重试限制或延时冲突中断
reserved	4:3	RO	0x0	保留
disable_rx_used_bit_read_interrupt	2	WO	0x0	去使能接收 used bit 读中断
disable_receive_complete_interrupt	1	WO	0x0	去使能接收完成中断
reserved	0	RO	0x0	保留

5.3.3.121 int_qx_mask (0x0640+0x4*(x-1))

域	位	读写	复位值	描述
reserved	31:12	RO	0x0	保留
resp_not_ok_interrupt_mask	11	RO	0x1	bresp/hresp 信号没准备好中断屏蔽 0: 中断已使能 1: 中断已禁止
reserved	10:8	RO	0x0	保留
transmit_complete_interrupt_mask	7	RO	0x1	发送完成中断屏蔽
amba_error_interrupt_mask	6	RO	0x1	AXI 总线错误中断屏蔽
retry_limit_exceeded_or_late_collision_interrupt_mask	5	RO	0x1	超出重试限制或延时冲突中断屏蔽
reserved	4:3	RO	0x0	保留
rx_used_interrupt_mask	2	RO	0x1	接收 used 中断屏蔽
receive_complete_interrupt_mask	1	RO	0x1	接收完成中断屏蔽
reserved	0	RO	0x0	保留

5.3.3.122 usx_control_register (0x0A80)

域	位	读写	复位值	描述
reserved	31:17	RO	0x0	保留
hs_mac_speed	16:14	RW	0x0	值为 3'h0 时代表 mac 速率为 100M; 值为 3'h1 时代表 mac 速率为 1G; 值为 3'h2 时代表 mac 速率为 2.5G; 值为 3'h3 时代表 mac 速率为 2.5G; 值为 3'h4 时代表 mac 速率为 10G
Serdes_rate	13:12	RW	0x1	值为 2'h0 时代表 serdes 速率为 5G; 值为 1'h1 时代表 serdes 速率为 10G
reserved	11:10	RO	0x0	保留
rx_scr_bypass	9	RW	0x0	写 1' b1 时, 接收方向不经过扰码器; 写 1' b0 时, 接收方向经过扰码器
tx_scr_bypass	8	RW	0x0	写 1' b1 时, 发送方向不经过扰码器; 写 1' b0 时, 发送方向经过扰码器
reserved	7:6	RO	0x0	保留

域	位	读写	复位值	描述
Fec_ena_err_ind	5	RW	0x0	FEC 错误指示使能：写 1' b1 时，将 PCS 时钟设置为 FEC 不可纠错的时钟；写 1' b0 时，FEC 不可纠错的时钟对 PCS 同步头不影响
Fec_enable	4	RW	0x0	前向纠错使能：写 1' b1 时，使能前向纠错功能；写 1' b0 时，去使能前向纠错功能
reserved	3	R0	0x0	保留
rx_sync_reset	2	RW	0x0	接收方向复位：写 1' b1 时，接收方向复位；写 1' b0 时，接收方向复位释放
tx_datapath_en	1	RW	0x0	高速 PCS 发送通路使能：写 1' b1 时，使能高速 PCS 发送通路；写 1' b0 时，去使能高速 PCS 发送通路
signal_ok	0	RW	0x0	高速 PCS 接收通路使能：写 1' b1 时，使能高速 PCS 接收通路；写 1' b0 时，去使能高速 PCS 接收通路

5.3.3.123 usx_status_register (0x0A88)

域	位	读写	复位值	描述
reserved	31:29	R0	0x0	保留
tx_fault	28	R0	0x0	值为 1' b1 时代表发送方向编码状态机进入到错误状态；值为 1' b0 时代表发送方向编码状态机没进入到错误状态
rx_fault	27	R0	0x0	值为 1' b1 时代表接收方向译码状态机进入到错误状态；值为 1' b0 时代表接收方向译码状态机没进入到错误状态
reserved	26:1	R0	0x0	保留
block_lock	0	R0	0x0	值为 1' b1 时代表 PCS 进入同步状态状态；值为 1' b0 时代表 PCS 未进入同步状态状态

5.3.3.124 mmsl_control (0x0F00)

该寄存器只 mac0 有。

域	位	读写	复位值	描述
reserved	31:9	R0	0x0	保留
mmsl_bypass	8	R0	0x1	当配置为 1，mmsl 被旁路掉
invert_mrcr	7	R0	0x0	值为 1' b1 时代表接收方向译码状态机进入到错误状态；值为 1' b0 时代表接收方向译码状态机没进入到错误状态
mmsl_debug_mode	6	R0	0x0	保留
route_rx_to_pmac	5	R0	0x0	值为 1' b1 时代表 PCS 进入同步状态状态；值为 1' b0 时代表 PCS 未进入同步状态状态
restart_ver	4	R0	0x0	重启验证，写 1 启动验证流程。 如果 pre_enable=0 或者 verify_disable=1 或验证流程已经在进行时无法发起 restart_ver 请求。

域	位	读写	复位值	描述
				如果激活抢占功能，需要在写 restart_ver 位之前设置 route_rx_to_pmac 为零，否则快速 MAC 帧将被发送到 pmac，并且如果它们散布在 pmac 帧片段之间将会丢失。这个位在读取时总是返回零。”
pre_enable	3	R0	0x0	该 bit 用于启用/禁用抢占操作。如果设置为 0，则不会发生抢占，验证报文将不会被响应或发送。如果 verify_disable 为零，则在用 1 写入该位时开始验证过程，并且在验证过程完成后可以发生抢占。如果 verify_disable 为高值，则可以在设置该位后立即发生抢占。如果该位从 1 重置为 0，并且正在进行抢占，则抢占将完成，不会再发生抢占。”
verify_disable	2	R0	0x0	该位用于启用/禁用验证过程，以确定链路伙伴是否支持 802.3br。如果禁用，则不会验证链接伙伴，并且一旦将 pre_enable 设置为 1，就会启用抢占。无论是否设置了这个位，都要验证数据包是否得到响应。该位是静态的，必须在设置 pre_enable 之前有效。
add_frag_size	1:0	R0	0x0	emac 抢占之前 pmac 发送的最小字节数如果在 eMAC 发送队列上启用了 EnST，它会在 EnST 开始时间之前向 MMSL 发送一个早期保持信号 add_frag_time。 00: 64B 01: 128B 10: 192B 11: 256B 该位域是静态的，必须在 pre_enable 设置或 EnST 控制寄存器启用时间敏感调度之前有效。

5.3.3.125 mmsl_status (0x0F04)

该寄存器只 mac0 有。

域	位	读写	复位值	描述
reserved	31:11	R0	0x0	保留
smd_error	10	RW	0x0	如果收到非法的 SMD，也就是说，如果 SMD 不是 Express、Verify、Response、Start Preemptible 或者 Continuation Preemptible SMD 该 bit 设置为 1。
frer_count_err	9	RW	0x0	帧数错误状态。 如果收到的 SMD-C 表示不同于预期的帧数(即片段属于另一个帧，而不是在此之前已经收到的开始数据包)，或者如果发生了片段错误，这意味着收到的 SMD-C 后面的字段编码的片段数不同于它的期望

域	位	读写	复位值	描述
				值。
smdc_error	8	RW	0x0	SMD-C 错误状态。 如果在等待 SMD-S 时收到 SMD-C, 则将该 bit 置 1。
smds_error	7	RW	0x0	SMD-S 错误状态。 如果在等待 SMD-C 时收到 SMD-S, 则将该 bit 置 1。
rcv_v_error	6	RW	0x0	当收到错误的 m-packet 时置 1
rcv_r_error	5	WO	0x0	如果回复 m-packet 错误该 bit 置 1
verify_status	4:2	RW	0x0	验证状态机状态 000: INIT_VERIFICATION 001: VERIFICATION_IDLE 010: SEND_VERIFY 011: WAIT_FOR_RESPONSE 100: VERIFIED 101: VERIFY_FAIL
respond_status	1	RW	0x0	Response 状态机状态 0:R_IDLE 1:SEND_RESPOND
pre_active	0	RW	0x0	当验证完成后该 bit 置 1, 或者在配置 pre_enable 时 verify_disable 为 1。

5.3.3.126 mmsl_err_stats (0x0F08)

该寄存器只 mac0 有。

域	位	读写	复位值	描述
reserved	31:24	R0	0x0	保留
smd_err_count	23:16	R0	0x0	由于未知 SMD 值或者无帧的 SMD-C 导致 mac 帧被拒绝的个数统计。
reserved	15: 8	R0	0x0	帧数错误状态。 如果收到的 SMD-C 表示不同于预期的帧数(即片段属于另一个帧, 而不是在此之前已经收到的开始数据包), 或者如果发生了片段错误, 这意味着收到的 SMD-C 后面的字段编码的片段数不同于它的期望值。
ass_error_count	7:0	R0	0x0	有重组错误的 MAC 帧的计数。 每当进入 ASSEMBLY_ERROR 状态时加 1

5.3.3.127 mmsl_ass_ok_count (0x0F0C)

该寄存器只 mac0 有。

域	位	读写	复位值	描述
reserved	31:17	R0	0x0	保留
ass_ok_count	16:0	R0	0x0	成功重组并传送到 MAC 的 MAC 帧数统计

5.3.3.128 mmsl_frag_count_rx (0x0F10)

该寄存器只 mac0 有。

域	位	读写	复位值	描述
reserved	31:17	R0	0x0	保留
frag_count_rx	16:0	R0	0x0	由于抢占而收到的额外 m-packet 的计数

5.3.3.129 mmsl_frag_count_tx (0x0F14)

该寄存器只 mac0 有。

域	位	读写	复位值	描述
reserved	31:17	R0	0x0	保留
frag_count_tx	16:0	R0	0x0	由于抢占而发送的额外 m-packet 的计数

5.3.3.130 mmsl_int_status (0x0F18)

该寄存器只 mac0 有。

域	位	读写	复位值	描述
reserved	31:6	R0	0x0	保留
smd_err	5	R0	0x0	如果收到非法的 SMD，也就是说，如果 SMD 不是 Express、Verify、Response、Start Preemptible 或者 Continuation Preemptible SMD。
fr_count_err	4	R0	0x0	帧数错误状态。 如果收到的 SMD-C 表示不同于预期的帧数（即片段属于另一个帧，而不是在此之前已经收到的开始数据包），或者如果发生了片段错误，这意味着收到的 SMD-C 后面的字段编码的片段数不同于它的期望值。
smdc_err	3	R0	0x0	SMD-C 错误状态。
smds_err	2	R0	0x0	如果在等待 SMD-S 时收到 SMD-C，则将该 bit 置 1。
rcv_v_err	1	R0	0x0	收到错误的验证 m-packet
rcv_r_err	0	R0	0x0	收到错误的 response m-packet。如果出现这种情况，则验证过程失败

5.3.3.131 mmsl_int_enable (0x0F1C)

该寄存器只 mac0 有。

域	位	读写	复位值	描述
reserved	31:6	R0	0x0	保留
smd_err_int_en	5	R0	0x0	使能中断
fr_count_err_int_en	4	R0	0x0	使能中断
smdc_err_int_en	3	R0	0x0	使能中断
smds_err_int_en	2	R0	0x0	使能中断
rcv_v_err_int_en	1	R0	0x0	使能中断

域	位	读写	复位值	描述
rcv_r_err_int_en	0	RO	0x0	使能中断

5.3.3.132 mmsl_int_disable (0x0F20)

该寄存器只 mac0 有。

域	位	读写	复位值	描述
reserved	31:6	WO	0x0	保留
smd_err_int_dis	5	WO	0x0	失能中断
fr_count_err_int_dis	4	WO	0x0	失能中断
smdc_err_int_dis	3	WO	0x0	失能中断
smds_err_int_dis	2	WO	0x0	失能中断
rcv_v_err_int_dis	1	WO	0x0	失能中断
rcv_r_err_int_dis	0	WO	0x0	失能中断

5.3.3.133 mmsl_int_mask (0x0F24)

该寄存器只 mac0 有。

域	位	读写	复位值	描述
reserved	31:6	WO	0x0	保留
smd_err_int_mask	5	WO	0x0	屏蔽中断
fr_count_err_int_mask	4	WO	0x0	屏蔽中断
smdc_err_int_mask	3	WO	0x0	屏蔽中断
smds_err_int_mask	2	WO	0x0	屏蔽中断
rcv_v_err_int_mask	1	WO	0x0	屏蔽中断
rcv_r_err_int_mask	0	WO	0x0	屏蔽中断

5.3.3.134 emac_network_control (0x1000)

该寄存器只 mac0 有。

域	位	读写	复位值	描述
reserved	31	RO	0x0	保留
two_pt_five_gig	29	RW	0x0	值为 1 时 2.5g 速率模式使能；值为 0 时 2.5g 速率模式去使能。
sel_mii_on_rgmii	28	RW	0x0	值为 1 时使能 rgmii 接口；值为 0 时使能 rmii 接口。
tx_lpi_en	19	RW	0x0	值为 1 时发送低功耗报文；值为 0 时不发送低功耗报文。
transmit_halt	10	RW	0x0	值为 1 时发送停止；值为 0 时发送不停止。
transmit_start	9	RW	0x0	值为 1 时发送开始；值为 0 时发送不开始。
clear_all_stats_regs	5	WO	0x0	值为 1 时清除所有统计寄存器；值为 0 时不清除所有统计寄存器。
man_port_en	4	RW	0x0	值为 1 时使能 MDIO 接口；值为 0 时去使能 MDIO 接口。

域	位	读写	复位值	描述
enable_transmit	3	RW	0x0	值为 1 时使能发送通道；值为 0 时去使能发送通道。
enable_receive	2	RW	0x0	值为 1 时使能接收通道；值为 0 时去使能接收通道。
loopback_local	1	RW	0x0	值为 1 时使能 local 环回；值为 0 时去使能 local 环回。

5.3.3.135 emac_network_config (0x1004)

该寄存器只 mac0 有。

域	位	读写	复位值	描述
sgmii_mode_enable	27	RW	0x0	sgmii 模式使能，为了满足 sgmii 需求，改变自协商行为和自协商对端能力寄存器，且将链路定时器时间从 10ms 减少到 1.6ms。值为 1 时使能 sgmii 模式；值为 0 时去使能 sgmii 模式。
en_half_duplex_rx	25	RW	0x0	值为 1 时接收方向为半双工模式；值为 0 时去使能收方向半双工模式。
receive_checksum_offload_enable	24	RW	0x0	值为 1 时接收方向检查 IP、TCP、UDP 报文的校验值；值为 0 时接收方向不检查 IP、TCP、UDP 报文的校验值。
disable_copy_of_pause_frames	23	RW	0x0	值为 1 时 pause 帧不写到存储里；值为 0 时 pause 帧写到存储里。
data_bus_width	22:21	RW	0x0	axi 总线数据位宽。值为 0 时数据位宽为 32bit；值为 1 时数据位宽为 64bit；值为 2 时数据位宽为 128bit。
mdc_clock_division	20:18	RW	0x2	apb 时钟分频系数，分频后的时钟用作 mdc 时钟。值为 0x0 时 8 分频；值为 0x1 时 16 分频；值为 0x2 时 32 分频；值为 0x3 时 48 分频；值为 0x4 时 64 分频；值为 0x5 时 96 分频；值为 0x6 时 128 分频；值为 0x7 时 224 分频。
fcs_remove	17	RW	0x0	值为 1 时接收报文去掉校验再写到存储里；值为 0 时接收报文不去掉校验写到存储里。
pcs_select	11	RW	0x0	值为 1 时使能 TBI 接口的 pcs 功能；值为 0 时使能 GMII/MII 接口的 pcs 功能。
gigabit_mode_enable	10	RW	0x0	值为 0 时当接口类型为 MII 或 TBI 时，MAC 速率配置为 10M/100M；值为 1 时当接口类型为 GMII 或 TBI 时，MAC 速率配置为千兆。
copy_all_frames	4	RW	0x0	值为 1 时接收方向去掉滤波功能；值为 0 时接收方向使能滤波功能
jumbo_frames	3	RW	0x0	值为 1 时可处理长度最大 16k 字节的报文。值为 0 时不可处理长度最大 16k 字节的报文。
full_duplex	1	RW	0x0	值为 1 时 mac 为全双工模式；值为 0 时 mac 为半双工模式。
speed	0	RW	0x0	值为 1 时配置 mac 速率为 100M；值为 0 时配置 mac 速率为 10M。这个 bit 配合 bit10 一起使用。

5.3.3.136 emac_dma_config (0x1010)

该寄存器只 mac0 有。

域	位	读写	复位值	描述
dma_addr_bus_width_1	30	RW	0x0	dma 的地址位宽。值为 0 时 dma 地址位宽为 32bit；值为 1 时 dma 地址位宽为 64bit。
force_max_amba_burst_tx	26	RW	0x0	值为 1 时配置发送方向 dma 的 burst 达到最大值；值为 0 时不配置发送方向 dma 的 burst 达到最大值。
force_max_amba_burst_rx	25	RW	0x0	值为 1 时配置接收方向 dma 的 burst 达到最大值；值为 0 时不配置接收方向 dma 的 burst 达到最大值。
force_discard_on_err	24	RW	0x0	值为 1 时当读到 used bit 值为 1 时，自动丢弃报文；值为 0 时不丢弃报文。
endian_swap_packet	7	RW	0x0	值为 1 时报文设置为大端；值为 0 时报文设置为小端。
endian_swap_management	6	RW	0x0	值为 1 时描述符设置为大端；值为 0 时描述符设置为小端。
amba_burst_length	4:0	RW	0x4	axi 总线 burst 数量最大值设置。值为 5'b1xxxx 时 burst 最大值为 16；值为 5'b01xxx 时 burst 最大值为 8；值为 5'b001xx 时 burst 最大值为 4；值为 5'b0001x/00001 时 burst 最大值为 1；值为 5'b00000 时 burst 最大值为 256。

5.3.3.137 emac_receive_q_ptr (0x1018)

该寄存器只 mac0 有。

域	位	读写	复位值	描述
dma_rx_q_ptr	31:2	RW	0x0	q 队列的接收描述符缓存地址
dma_rx_dis_q	0	RW	0x1	值为 1 时 q 队列接收无效；值为 0 时 q 队列接收有效。

5.3.3.138 emac_transmit_q_ptr (0x101C)

该寄存器只 mac0 有。

域	位	读写	复位值	描述
dma_tx_q_ptr	31:2	RW	0x0	q 队列的发送描述符缓存地址
dma_tx_dis_q	0	RW	0x1	值为 1 时 q 队列发送无效；值为 0 时 q 队列发送有效。

5.3.3.139 emac_int_status (0x1024)

该寄存器只 mac0 有。

域	位	读写	复位值	描述
wol_interrupt	28	R0	0x0	值为 1 时表示有网络唤醒事件发生；值为 0 时表示无网络唤醒事件发生。
receive_lpi_indication_status_bit_change	27	R0	0x0	值为 1 表示接收方向低功耗状态 bit 改变；值为 0 表示接收方向低功耗状态 bit 没改变。
pcs_auto_negotiation_complete	16	R0	0x0	值为 1 表示 pcs 自协商完成；值为 0 表示 pcs 自协商未完成。
external_interrupt	15	R0	0x0	值为 1 表示外部中断引脚 ext_interrupt_in 检测到上升沿。值为 0 表示外部中断引脚 ext_interrupt_in 未检测到上升沿。
link_change	9	R0	0x0	值为 1 表示 pcs 的 link 状态改变。值为 0 表示 pcs 的 link 状态没改变。
transmit_complete	7	R0	0x0	值为 1 表示发送完成。值为 0 表示发送没完成。
tx_used_bit_read	3	R0	0x0	值为 1 表示发送方向读到 used bit。值为 0 表示发送方向没读到 used bit。
rx_used_bit_read	2	R0	0x0	值为 1 表示接收方向读到 used bit。值为 0 表示接收方向没读到 used bit。
receive_complete	1	R0	0x0	值为 1 表示一个报文已经存在 sram 里。值为 0 表示报文没在 sram 里。
management_frame_sent	0	R0	0x0	值为 1 表示 mdio 已完成操作。值为 0 表示 mdio 未完成操作。

5.3.3.140 emac_hs_mac_config (0x1050)

该寄存器只 mac0 有。

域	位	读写	复位值	描述
preamble_in_crc_rx	8:7	RW	0x0	接收方向 preamble 参与校验计算。值为 0 时 preamble 不参与校验计算；值为 1 时 7MS bytes of preamble 参与校验计算；值为 2 时所有 preamble 参与校验计算；值为 3 时 4MS bytes of preamble 参与校验计算。
preamble_in_crc_tx	6:5	RW	0x0	发送方向 preamble 参与校验计算。值为 0 时 preamble 不参与校验计算；值为 1 时 7MS bytes of preamble 参与校验计算；值为 2 时所有 preamble 参与校验计算；值为 3 时 4MS bytes of preamble 参与校验计算。
reserved	2:0	R	0x0	保留

5.3.3.141 emac_axi_max_pipeline (0x1054)

该寄存器只 mac0 有。

域	位	读写	复位值	描述
aw2b_max_pipeline	15:8	RW	0x1	axi 总线的 aw 通道 outstanding 最大值
ar2r_max_pipeline	7:0	RW	0x1	axi 总线的 ar 通道 outstanding 最大值

5.3.3.142 emac_octets_txed_bottom (0x1100)

该寄存器只 mac0 有。

域	位	读写	复位值	描述
count	31:0	RO	0x0	正确发送报文数低 32bit, 8 进制

5.3.3.143 emac_octets_txed_top (0x1104)

该寄存器只 mac0 有。

域	位	读写	复位值	描述
count	15:0	RO	0x0	正确发送报文数高 16bit, 8 进制

5.3.3.144 emac_frames_txed_ok (0x1108)

该寄存器只 mac0 有。

域	位	读写	复位值	描述
count	31:0	RO	0x0	正确发送报文数, 10 进制

5.3.3.145 emac_broadcast_txed (0x110C)

该寄存器只 mac0 有。

域	位	读写	复位值	描述
count	31:0	RO	0x0	正确发送 Broadcast 报文数

5.3.3.146 emac_multicast_txed (0x1110)

该寄存器只 mac0 有。

域	位	读写	复位值	描述
count	31:0	RO	0x0	正确发送 Multicast 报文数

5.3.3.147 emac_pause_frames_txed (0x1114)

该寄存器只 mac0 有。

域	位	读写	复位值	描述
count	15:0	RO	0x0	正确发送 pause 报文数

5.3.3.148 emac_frames_txed_64 (0x1118)

该寄存器只 mac0 有。

域	位	读写	复位值	描述
count	31:0	R0	0x0	正确发送长度为 64 字节的报文数

5.3.3.149 emac_frames_txed_65 (0x111C)

该寄存器只 mac0 有。

域	位	读写	复位值	描述
count	31:0	R0	0x0	正确发送长度在 65~127 字节之间的报文数

5.3.3.150 emac_frames_txed_128 (0x1120)

该寄存器只 mac0 有。

域	位	读写	复位值	描述
count	31:0	R0	0x0	正确发送长度在 128~255 字节之间的报文数

5.3.3.151 emac_frames_txed_256 (0x1124)

该寄存器只 mac0 有。

域	位	读写	复位值	描述
count	31:0	R0	0x0	正确发送长度在 255~511 字节之间的报文数

5.3.3.152 emac_frames_txed_512 (0x1128)

该寄存器只 mac0 有。

域	位	读写	复位值	描述
count	31:0	R0	0x0	正确发送长度在 512~1023 字节之间的报文数

5.3.3.153 emac_frames_txed_1024 (0x112C)

该寄存器只 mac0 有。

域	位	读写	复位值	描述
count	31:0	R0	0x0	正确发送长度在 1024~1518 字节之间的报文数

5.3.3.154 emac_frames_txed_1519 (0x1130)

该寄存器只 mac0 有。

域	位	读写	复位值	描述
count	31:0	R0	0x0	正确发送长度在 1519~16k 字节之间的报文数

5.3.3.155 emac_octets_rxed_bottom (0x1150)

该寄存器只 mac0 有。

域	位	读写	复位值	描述
count	31:0	R0	0x0	正确接收报文数低 32bit, 8 进制

5.3.3.156 emac_octets_rxed_top (0x1154)

该寄存器只 mac0 有。

域	位	读写	复位值	描述
count	15:0	R0	0x0	正确接收报文数高 16bit, 8 进制

5.3.3.157 emac_frames_rxed_ok (0x1158)

该寄存器只 mac0 有。

域	位	读写	复位值	描述
count	31:0	R0	0x0	正确接收报文数, 10 进制

5.3.3.158 emac_broadcast_rxed (0x115C)

该寄存器只 mac0 有。

域	位	读写	复位值	描述
count	31:0	R0	0x0	正确接收 Broadcast 报文数

5.3.3.159 emac_multicast_rxed (0x1160)

该寄存器只 mac0 有。

域	位	读写	复位值	描述
count	31:0	R0	0x0	正确接收 Multicast 报文数

5.3.3.160 emac_pause_frames_rxed (0x1164)

该寄存器只 mac0 有。

域	位	读写	复位值	描述
count	15:0	R0	0x0	正确接收 pause 报文数

5.3.3.161 emac_frames_rxed_64 (0x1168)

该寄存器只 mac0 有。

域	位	读写	复位值	描述
count	31:0	R0	0x0	正确接收长度为 64 字节的报文数

5.3.3.162 emac_frames_rxed_65 (0x116C)

该寄存器只 mac0 有。

域	位	读写	复位值	描述
count	31:0	R0	0x0	正确接收长度在 65~127 字节之间的报文数

5.3.3.163 emac_frames_rxed_128 (0x1170)

该寄存器只 mac0 有。

域	位	读写	复位值	描述
count	31:0	R0	0x0	正确接收长度在 128~255 字节之间的报文数

5.3.3.164 emac_frames_rxed_256 (0x1174)

该寄存器只 mac0 有。

域	位	读写	复位值	描述
count	31:0	R0	0x0	正确接收长度在 255~511 字节之间的报文数

5.3.3.165 emac_frames_rxed_512 (0x1178)

该寄存器只 mac0 有。

域	位	读写	复位值	描述
count	31:0	R0	0x0	正确接收长度在 512~1023 字节之间的报文数

5.3.3.166 emac_frames_rxed_1024 (0x117C)

该寄存器只 mac0 有。

域	位	读写	复位值	描述
count	31:0	R0	0x0	正确接收长度在 1024~1518 字节之间的报文数

5.3.3.167 emac_frames_rxed_1519 (0x1180)

该寄存器只 mac0 有。

域	位	读写	复位值	描述
count	31:0	R0	0x0	正确接收长度在 1519~16k 字节之间的报文数

5.3.4 描述符表

接收缓存描述符如下表所示：

bit	function
Word 0	
31:3	缓冲区起始地址[31:3]

bit	function
2	缓冲区起始地址[2]，在扩展缓存描述符模式(DMA configuration register[28] = 1)，表明描述符包含时戳。
1	wrap 位有效代表最后一个接收描述符
0	ownership 位需要为 0，为了写数据到接收缓存里。当完成写一包报文到存储里时，将 ownership 位置 1。软件需要清除 ownership 位在缓存可以再次使用之前。
Word 1	
31	检测到所有全局广播地址
30	多播哈希匹配
29	单播哈希匹配
28	扩展地址匹配。注意：如果报文缓存模式和特殊地址滤波配置数量大于 4，扩展地址匹配不需要在这个 bit 里上报。
27	找到特殊地址匹配寄存器，bit25 和 bit26 表明引起匹配的特殊地址寄存器。
26:25	特殊地址寄存器匹配如下： 2'b00:特殊地址寄存器 1 匹配； 2'b01:特殊地址寄存器 2 匹配； 2'b10:特殊地址寄存器 3 匹配； 2'b11:特殊地址寄存器 4 匹配； 如果特殊地址不止一个，只匹配一个，按优先级划分寄存器 4 到 1
24	这个 bit 有不同意义，基于接收校验卸载是否使能。如果接收校验卸载去使能，Type ID 寄存器匹配发现，bit22 和 bit23 指示导致匹配的 type ID 寄存器。如果接收校验卸载使能，值为 0 时，报文不是 SNAP 编码和/或包含设置了 CFI bit 的 VLAN 标签。值为 1 时，报文是 SNAP 编码，且包含 VLAN 标签或包含未设置了 CFI bit 的 VLAN 标签。
23:22	这个 bit 有不同意义，基于接收校验卸载是否使能。当接收校验卸载未使能时，Type ID 寄存器匹配，编码如下：2'b00 代表 Type ID 寄存器 1 匹配；2'b01 代表 Type ID 寄存器 2 匹配；2'b10 代表 Type ID 寄存器 3 匹配；2'b11 代表 Type ID 寄存器 4 匹配；如果 Type ID 寄存器不止一个，只匹配一个，按优先级划分寄存器 4 到 1 当接收校验卸载使能时，2'b00 代表 IP 报文头校验或 TCP/UDP 校验检查；2'b01 代表 IP 报文头校验检查正确。TCP 或 UDP 校验检查；2'b10 代表 IP 报文头和 TCP 校验检查且正确；2'b11 代表 IP 报文头和 UDP 校验检查且正确；
21	发现 VLAN 标签，对于结合堆叠 VLAN 处理功能的报文，如果接收到的第二个 VLAN 标签 ID 值为 0x8100，将这个 bit 置位
20	优先级标签检测，对于结合堆叠 VLAN 处理功能的报文，如果接收到的第二个 VLAN 标签 ID 值为 0x8100 且包含空 VLAN 标识符，将这个 bit 置位
19:17	在设置 bit15 和 bit21 时，这些 bit 代表 VLAN 优先级。当时能头部和数据分离时，bit17 代表表明这个描述符指向最后一个头的缓存。
16	这个 bit 有不同含义基于 DMA Configuration 寄存器的 bit13 和 bit5 的状态。当头部/数据分离使能且这个描述符不是报文最后一个描述符时，这个 bit 表明描述符指向包含头部字节数的数据缓存。当这个描述符是报文最后一个描述符时，且设置 DMA Configuration 寄存器的 bit13，这个 bit 代表 FCS/CRC 错误。当这个描述符为报文的最后一个描述符且 DMA Configuration 寄存器的 bit13 清除时，接收报文包含 VLAN 标签，这个 bit 代表 CFI。
15	报文的结尾，当有效时缓存包含一个报文的结尾。如果报文结尾未设置，仅为一个有效状态 bit。如果头部/数据分离使能，bit16 和 bit17 也是有效的状态 bit 当这个 bit 未设置时。
14	报文起始位，当有效时缓存包含一个报文的起始位。如果设置 bit15 和 bit14，缓存包含整个报文。

bit	function
13	这个 bit 有不同含义基于是否巨型帧和忽略 FCS 模式使能。如果都未使能，这个 bit 置 0。如果使能巨型帧或 RSC 模式，这个 bit 和 [12:0] 一起代表报文长度。如果忽略 FCS 使能和巨型帧去使能，这个 bit 代表没和报文的 FCS 状态如下所示：1'b0 时代表 FCS 正确；1'b1 时代表 FCS 错误，但会复制到存储因为忽略 FCS 使能；
12:0	当使能头部/数据分离且设置 bit17 时，这些 bit 代表头部字节数。当设置 bit15 时，这些 bit 代表接收报文长度，报文是否包含 FCS 基于是否使能 FCS 去除模式。如果使能 FCS 丢弃模式，用 13 个 bit 代表包含 FCS 的报文长度。如果使能巨型帧，这 12 个 bit 和 bit13 一起使用。如果使能 FCS 丢弃模式，用 13 个 bit 代表不包含 FCS 的报文长度。如果使能巨型帧，这 12 个 bit 和 bit13 一起使用。

当使能 64bit 地址模式时，下面的表格表明添加的描述符

bit	function
Word 2 (64bit 地址)	
31:0	数据缓存的高 32bit 地址
Word 3 (64bit 地址)	
31:0	未使用

当使能描述符时戳捕获模式时，下面的表格表明添加的描述符

bit	function
Word 2 (32bit 地址) 或 Word 4 (64bit 地址)	
31:30	时戳秒[1:0]
29:0	时戳纳秒[29:0]
Word 3 (32bit 地址) 或 Word 5 (64bit 地址)	
31:10	未使用
9:0	时戳秒[11:2]
	注意：时戳模式由 rx_bd_control 寄存器控制。接收描述符时戳插入模式 bit 定义如下：2'b00 代表时戳插入去使能；2'b01 代表时戳插入仅对 PTP Event 报文；2'b10 代表时戳插入仅对所有 PTP 报文；2'b11 代表时戳插入对所有报文；这些时戳 bit 被写回报文的最后一个缓存描述符里。

非 LS0 报文的发送缓存描述符如下表所示：

bit	function
Word 0	
31:0	缓存区字节地址
Word 1	
31	要设置为 0 为了读数据到发送缓存。设置这个 bit 为 1 为了报文的第一个缓存，如果成功完成传输。软件必须清除这个 bit 在可以再次使用之前。
30	wrap 位，标记发送缓存描述符的最后一个描述符。这个 bit 可以被设置任意 buffer 在一包之内。
29	超过重发限制，发送出现错误
28	传输溢出，当封装的数据一开始写到 fifo 里，或 hresp 错误，或不能及时得到传输数据，或缓存没空间时设置这个 bit。这个 bit 不会设置当 DMA 设置为封包缓存模式时。
27	传输报文出问题由于 AHB 或 AXI 错误，在通过 AHB/AXI 读取数据期间出现错误出现错误会设置这个 bit，包括 HRESP 或 RRESP/BRESP 错误和在一个报文中途出现缓存空间不足。如果报文长度太大大于配置的缓存空间也会设置这个 bit。

bit	function
26	延迟冲突，发现传输错误。延迟冲突仅会强制设置这个状态 bit 在千兆模式。
25:24	保留
23	对于扩展缓存描述符模式，这个 bit 表明已在描述符里捕获到一个时戳。否则保留。
22:20	传输 IP/TCP/UDP 校验生成卸载错误：'b000 代表无错误；'b001 代表报文为一个 VLAN 报文，但头部没充分完成，或头部有错误；'b010 代表报文为一个 SNAP 报文，但头部没充分完成，或头部有错误；'b011 代表报文为一个 IP 报文，或 IP 包为无效短包，或 IP 类型不是 IPv4/IPv6；'b100 代表报文为一个 VLAN/SNAP/IP 报文；'b101 代表不支持封包分段。对于 IPv4，产生且插入 IP 校验；'b110 代表报文类型不为 TCP 或 UDP。TCP/UDP 校验因此不产生。对于 IPv4，产生且插入 IP 校验；'b111 代表封包过早结束被发现且不能产生 TCP/UDP 校验。
19:17	保留。必须设置为 3'b000 来去使能 TS0 和 UF0。
16	不会附加 CRC。当设置时代表缓存里的数据包含有效的 CRC 或填充数据。这个 bit 必须设置对于报文的第一个缓存且会被报文后来的缓存忽略。注意：必须清除这个 bit 当使用传输 IP/TCP/UDP 校验生成卸载时，否则校验生成卸载和置换不会发生。注意：必须清除这个 bit 当发送方向部分存储转发有效时。
15	最后一个缓存。当设置时，这个 bit 表明当前帧达到的最后一个缓存。
14	保留
13:0	缓存长度

TS0 报文头缓存的发送缓存描述符如下表所示：

bit	function
Word 0	
31:0	缓存区字节地址
Word 1	
31	要设置为 0 为了读数据到发送缓存。设置这个 bit 为 1 为了报文的第一个缓存，如果成功完成传输。软件必须清除这个 bit 在可以再次使用之前。
30	wrap 位，标记发送缓存描述符的最后一个描述符。这个 bit 可以被设置任意 buffer 在一包之内。
29	超过重发限制，发送出现错误
28	发送溢出，TS0 时为 0
27	传输报文出问题由于 AHB 或 AXI 错误，在通过 AHB/AXI 读取数据期间出现错误出现错误会设置这个 bit，包括 HRESP 或 RRESP/BRESP 错误和在一个报文中途出现缓存空间不足。如果报文长度太大大于配置的缓存空间也会设置这个 bit。
26	延迟冲突，发现传输错误。延迟冲突仅会强制设置这个状态 bit 在千兆模式。
25:24	TCP 数据流识别。用于选择硬件计数，用于 TCP 序列号生成
23	用于扩展缓存描述符模式，这个 bit 表明在描述符里捕获到一个时戳。否则保留。
22:20	传输 IP/TCP/UDP 校验生成卸载错误：'b000 代表无错误；'b001 代表报文为一个 VLAN 报文，但头部没充分完成，或头部有错误；'b010 代表报文为一个 SNAP 报文，但头部没充分完成，或头部有错误；'b011 代表报文为一个 IP 报文，或 IP 包为无效短包，或 IP 类型不是 IPv4/IPv6；'b100 代表报文为一个 VLAN/SNAP/IP 报文；'b101 代表不支持封包分段。对于 IPv4，产生且插入 IP 校验；'b110 代表报文类型不为 TCP 或 UDP。TCP/UDP 校验因此不产生。对于 IPv4，产生且插入 IP 校验；'b111 代表封包过早结束被发现且不能产生 TCP/UDP 校验。TCP 序列号选择源。'b0 代表使用头部缓存里的序列号值，且硬件生成序列号值对后来的小 TCP 报文。'b0 代表使用硬件生成序列号值对所有小 TCP 报文。
18:17	LS0 控制，设置为 2'b10 或 2'b11 使能 TS0

bit	function
16	以太网不添加 CRC。TS0 运行时必须设置为 0
15	最后缓存。必须清除因为 TS0 需要至少一个 payload 缓存
14	保留
13:0	缓存长度

TS0 报文 payload 缓存的发送缓存描述符如下表所示：

bit	function
Word 0	
31:0	缓存区字节地址
Word 1	
31	要设置为 0 为了读数据到发送缓存。设置这个 bit 为 1 为了报文的第一个缓存，如果成功完成传输。软件必须清除这个 bit 在可以再次使用之前。
30	wrap 位，标记发送缓存描述符的最后一个描述符。这个 bit 可以被设置任意 buffer 在一包之内。
29:16	TCP 最大分段尺寸字节值。TS0 会使用默认值 536 字节如果设置的值为 0。
15	最后一个缓存。当设置时，这个 bit 表明当前帧达到的最后一个缓存。
14	保留
13:0	缓存长度

UF0 报文头缓存的发送缓存描述符如下表所示：

bit	function
Word 0	
31:0	缓存区字节地址
Word 1	
31	要设置为 0 为了读数据到发送缓存。设置这个 bit 为 1 为了报文的第一个缓存，如果成功完成传输。软件必须清除这个 bit 在可以再次使用之前。
30	wrap 位，标记发送缓存描述符的最后一个描述符。这个 bit 可以被设置任意 buffer 在一包之内。
29	超过重发限制，发送出现错误
28	发送溢出，UF0 时为 0
27	传输报文出问题由于 AHB 或 AXI 错误，在通过 AHB/AXI 读取数据期间出现错误出现错误会设置这个 bit，包括 HRESP 或 RRESP/BRESP 错误和在一个报文中途出现缓存空间不足。如果报文长度太大大于配置的缓存空间也会设置这个 bit。
26	延迟冲突，发现传输错误。延迟冲突仅会强制设置这个状态 bit 在千兆模式。
25:24	保留
23	用于扩展缓存描述符模式，这个 bit 表明在描述符里捕获到一个时戳。否则保留。
22:20	传输 IP/TCP/UDP 校验生成卸载错误：'b000 代表无错误；'b001 代表报文为一个 VLAN 报文，但头部没充分完成，或头部有错误；'b010 代表报文为一个 SNAP 报文，但头部没充分完成，或头部有错误；'b011 代表报文为一个 IP 报文，或 IP 包为无效短包，或 IP 类型不是 IPv4/IPv6；'b100 代表报文为一个 VLAN/SNAP/IP 报文；'b101 代表不支持封包分段。对于 IPv4，产生且插入 IP 校验；'b110 代表报文类型不为 TCP 或 UDP。TCP/UDP 校验因此不产生。对于 IPv4，产生且插入 IP 校验；'b111 代表封包过早结束被发现且不能产生 TCP/UDP 校验。TCP 序列号选择源。'b0 代表使用头部缓存里的序列号值，且硬件生成序列号值对后来的小 TCP 报文。'b0 代表使用硬件生成序列号值对所有小 TCP 报文。

bit	function
19	保留
18:17	LS0 控制, 设置为 2'b01 使能 UF0。
16	以太网不添加 CRC。UF0 运行时必须设置为 0。
15	最后缓存。必须清除因为 UF0 需要至少一个 payload 缓存。
14	保留
13:0	缓存长度

UF0 报文 payload 缓存的发送缓存描述符如下表所示:

bit	function
Word 0	
31:0	缓存区字节地址
Word 1	
31	要设置为 0 为了读数据到发送缓存。设置这个 bit 为 1 为了报文的第一个缓存, 如果成功完成传输。软件必须清除这个 bit 在可以再次使用之前。
30	wrap 位, 标记发送缓存描述符的最后一个描述符。这个 bit 可以被设置任意 buffer 在一包之内。
29:16	TCP 最大分段尺寸字节值。UF0 会使用默认值 1518 字节如果设置的值为 0。
15	最后一个缓存。当设置时, 这个 bit 表明当前帧达到的最后一个缓存。
14	保留
13:0	缓存长度

当使能 64bit 地址模式时, 下面的表格表明添加的描述符

bit	function
Word 2 (64bit 地址)	
31:0	数据缓存的高 32bit 地址
Word 3 (64bit 地址)	
31:0	未使用

当描述符时戳捕获模式使能时, 下表表明添加的地址:

bit	function
Word 2 (32bit 地址) 或 Word 4 (64bit 地址)	
31:30	时戳秒[1:0]
29:0	时戳纳秒[29:0]
Word 3 (32bit 地址) 或 Word 5 (64bit 地址)	
31	使用传输发射时间
31:10	未使用
9:0	时戳秒[11:2]
注意: 时戳模式由 tx_bd_control 寄存器控制。发送描述符时戳插入模式 bit 定义如下: 2'b00 代表时戳插入去使能; 2'b01 代表时戳插入仅对 PTP Event 报文; 2'b10 代表时戳插入仅对所有 PTP 报文; 2'b11 代表时戳插入对所有报文; 这些时戳 bit 被写回报文的最后一个缓存描述符里。	

5.4 USB3.0 控制器

飞腾派 USB3.0 控制器兼容 USB3.0 规范，向下兼容 USB2.0 规范，兼容 xHCI1.1 规范。

5.4.1 操作说明

5.4.1.1 主机控制器初始化过程

USB 控制器兼容 xHCI，初始化步骤描述如下：

- 初始化系统 I/O 内存映射；
- 在芯片硬件复位后，等待 USBSTS 的 CNR (Controller Not Ready) 标志为 0；
- 设置 CONFIG 寄存器的 MaxDeviceslotsEnable (MaxSlotEn) 域，使能系统软件将要使用的设备 slots；
- 设置 DCBAAP (设备上下文基地址数组指针) 寄存器，该 64 位地址指向设备上下文基地址数组的位置；
- 配置 CommandRingControl，定义 Command Ring Dequeue Pointer，指向 ComandRing 的第一个 TRB 的开始地址；
- 初始化中断：设置中断相关寄存器，IMOD, IMAN, USBCMD, EventRingRegisters；
- 写 USBCMD 寄存器，将 Run/Stop 位设为 1 打开主机控制器；
- 主控制器打开并运行，Root Hub 端口将开始报告设备连接等信息，系统软件可以开始枚举设备。

5.4.1.2 设备枚举过程

- hub 报告设备连接，通过状态改变管道通知主机这个事件；
- 主机通过查询确定 hub 状态改变的具体信息；
- 主机知道哪个端口有新设备连接；
- 如果主机复位，Hub 执行对应端口复位处理流程。复位完成，端口进入回到使能状态；
- USB 设备进入 reset 状态能从 VBUS 汲取不超过 150mA 的电流，USB 设备所有的寄存器和状态进行复位，响应默认地址；
- 主机为 USB 设备分配一个唯一的地址，设备转换到 Address 状态；
- 在 USB 设备接收到唯一地址之前，其默认控制管道通过默认地址是可访问的，主机读取设备描述符；
- 主机设置同步延时通知设备主机发出一个包到设备接收所需要的延时时间；

- 主机使用 SetSEL 请求通知设备系统退出延时；
- 主机读取设备配置信息；
- 主机设置下游端口 U1/U2 超时；
- 基于配置信息和 USB 设备的用法，主机赋予设备配置值。设备处于 Configured 状态，设备汲取描述符描述的 VBUS 电源值。

5.4.1.3 设备数据传输流程

- 识别端点收集开始传输需要的信息；
- 准备数据缓冲，建立 TRBs；
- 开始传输相关 USB 设备 EP 的 TRBs；
- 等待传输完成。

5.4.1.4 初始化操作说明

- 上电复位；
- 配置域复位释放；
- 将控制器配置为 Host 模式；
- 控制器复位全释放；
- xHCI 驱动初始化控制器。

5.4.2 寄存器列表

表 5-16 USB3.0 寄存器基地址

名称	基地址
USB3_0	0x000_31A0_0000
USB3_1	0x000_31A2_0000

XHCI 协议规范定义寄存器基地址为 USB3.0 寄存器基地址+0x8000。

表 5-17 USB3 寄存器列表

寄存器名称	偏移	描述
USB_RESETN_STATUS	0x1_0000	USB 复位状态
USB_SOC_RESETN	0x1_0004	USB 复位控制
USB_MODE_STRAP	0x1_0008	USB 模式选择
USB_AXI_SIDE_CFG	0x1_000C	AXI 边带信号配置
USB2PHY_REFCLK_MODE	0x1_0020	USB2PHY 参考时钟选择
USB_IRQ_HANDLE	0x1_0028	控制器中断处理寄存器
LPI_CTR_COUNTER0	0x1_002C	LPI 模块控制寄存器
LPI_CTR_COUNTER1	0x1_0030	LPI 模块控制寄存器
LPI_CTR_EN	0x1_0034	LPI 模块控制寄存器

5.4.3 寄存器说明

5.4.3.1 USB_RESETN_STATUS (0x10000)

域	位	读写	复位值	描述
reserved	31:24	RO	0x0	保留
cfg_rstn_cnt_val	23:16	RW	0x64	cfg_rstn 复位释放计数值, 当计数器值等于 cfg_rstn_cnt_val 时复位 cfg_rstn 信号, 用于控制复位该信号的时间。
presetn_cnt_val	15:8	RW	0x4	presetn 复位释放计数值, 当计数器值等于 presetn_cnt_val 时复位 presetn 信号, 用于控制复位该信号的时间。
reserved	7:2	RO	0x0	保留
cfg_rstn_rdy	1	RO	0x0	cfg_rstn 复位释放计数完成位, 计数器值已等于 cfg_rstn_cnt_val 并复位 cfg_rstn 信号。
presetn_rdy	0	RO	0x0	presetn 复位释放计数完成位, 计数器值已等于 presetn_cnt_val 并复位 presetn 信号。

5.4.3.2 USB_SOC_RESETN (0x10004)

域	位	读写	复位值	描述
reserved	31:17	RO	0x0	保留
cfg_rstn	16	RW	0x1	USB2PHY 寄存器配置接口复位信号
reserved	15:9	RO	0x0	保留
preset_n	8	RW	0x1	USBSSP APB 时钟域复位
reserved	7:1	RO	0x0	保留
pwrup_rst_n	0	RW	0x1	USBSSP 上电复位 (除 APB 时钟域)

5.4.3.3 USB_MODE_STRAP (0x10008)

域	位	读写	复位值	描述
reserved	31:2	RO	0x0	保留
mode_strap	1:0	RW	0x0	与 USBSSP 的 mode_strap 信号相连 00: 控制器未初始化配置为主机 01: 控制器初始化配置为主机

5.4.3.4 USB_AXI_SIDE_CFG (0x1000C)

域	位	读写	复位值	描述
awlock	31	RW	0x0	AXI4 awlock 信号
awcache	30:27	RW	0x0	AXI4 awcache 信号
awprot	26:24	RW	0x2	AXI4 awprot 信号
awqos	23:20	RW	0x0	AXI4 awqos 信号
reserved	19:16	RO	0x0	保留

域	位	读写	复位值	描述
arlock	15	RW	0x0	AXI4 arlock 信号
arcache	14:11	RW	0x0	AXI4 arcache 信号
arprot	10:8	RW	0x2	AXI4 arprot 信号
argos	7:4	RW	0x0	AXI4 argos 信号
reserved	3:0	RO	0x0	保留

5.4.3.5 USB2PHY_REFCLK_MODE (0x10020)

域	位	读写	复位值	描述
reserved	31:1	RO	0x0	保留
refclk_mode	0	RW	0x1	连接 USB2PHY 的 REFCLK_MODE 信号 0: 输入时钟 REFCLK 是 25MHz 1: 输入时钟 REFCLK 是 12MHz

5.4.3.6 USB_IRQ_HANDLE (0x10028)

域	位	读写	复位值	描述
reserved	31:25	RO	0x0	保留
irq_mix	24	RW	0x0	1 有效, 将 irq, otgirq, host_system_error, itp 中断合到 xhci_irq 中断信号, 否则中断由 otgirq, host_system_error, itp, xhci_irq 分别给出。
reserved	23:17	RO	0x0	保留
itp_clear	16	W1-1	0x0	itp 发出的电平中断清零, 重新开始计数发出中断, 该域写 1 在下一时钟周期自动返回为 0。
counter_itp_value	15:0	RW	0x0	收集 itp 脉冲转换为电平中断, 设置电平发出间隔时间值。

5.4.3.7 LPI_CTR_COUNTER0 (0x1002C)

域	位	读写	复位值	描述
wait_cnt0	31:0	RW	0x0	连接 LPI_CTR 模块的 wait_cnt[31:0] 信号, 用于 Q-Channel0, 总线空闲时间达到 wait_cnt 值后进行 aclk 时钟关断。

5.4.3.8 LPI_CTR_COUNTER1 (0x10030)

域	位	读写	复位值	描述
wait_cnt1	31:0	RW	0x0	连接 LPI_CTR 模块的 wait_cnt[63:32] 信号, 用于 Q-Channel1, 总线空闲时间达到 wait_cnt 值后进行 clk_sof 时钟关断。

5.4.3.9 LPI_CTR_EN (0x10034)

域	位	读写	复位值	描述
reserved	31:2	R0	0x0	保留
lpi_en	1:0	RW	0x0	连接 LPI_CTR 模块的 lpi_en 信号，用于 Q-channel 0 和 Q-channel 1 的使能。 bit0 用于配置 Q-channel 0，bit1 用于配置 Q-channel 1。 置 1 表示使能，置 0 表示不使能。

5.5 USB2.0 OTG 控制器

飞腾派 USB2.0 OTG 兼容 USB2.0 规范。

5.5.1 操作说明

5.5.1.1 初始化操作说明

初始化为 host 模式：

- 上电复位；写 HCPORCTRL 寄存器为 0x20 复位 port。
- 使能总线，开启扫描设备；写 OTGCTRL 寄存器为 0x81。
- 打开中断使能，设备速度识别完成后清除对应中断；写 HCUSBIEEN 寄存器打开中断使能；等待中断号为 0x10 的 usbrext 中断（全速），写 HCUSBIRQ 寄存器 0x10 清除此中断；等待中断号为 0x14 的 highspeed mode（高速）中断，写 HCUSBIRQ 寄存器 0x20 清除此中断。
- 驱动初始化控制器，开始枚举设备。
- 初始化为 device 模式：
- 上电复位；写 USBCS 寄存器为 0，清除 disconnect 位。
- 打开 OTGIEEN 使能，等待控制器进入设备模式中断，写 OTGIRQ 寄存器 0x10 清除此中断。
- 打开中断使能，设备速度识别完成后清除对应中断；写 USBIEEN 寄存器打开中断使能；等待中断号为 0x10 的 usbrext 中断（全速），写 USBIRQ 寄存器 0x10 清除此中断；等待中断号为 0x14 的 highspeed mode（高速）中断，写 USBIRQ 寄存器 0x20 清除此中断。
- 等待对端发起枚举。

5.5.1.2 数据传输操作说明

- 配置 ep 类型、size、addr、传输方向等；写 HCIN*MAXPCK 寄存器，设置 ep* 的最大包的大小；写 HCOU*STADDRH 和 HCOU*STADDRH 寄存器设置片内数据存放的地址；写 HCOU07IEN 寄存器使能所需的 ep；写 HCOU*TXCON 寄存器设置 ep 类型；写 HCENDPRST 寄存器设置 ep 号和 ep 方向；写 HCFIFOCTRL 清除所需 ep 的 toggle 位和设置 fifoauto 位。
- 配置需要通信的 ep 号；写 HCOU*CTRL 寄存器选择需要数据传输的两个 ep 号。
- 触发传输。

5.5.2 寄存器列表

表 5-18 USB2 寄存器基地址

名称	基地址
USB2_0 控制器寄存器	0x000_3280_0000
USB2_1 控制器寄存器	0x000_3284_0000
USB2_0 UIB 寄存器	0x000_3288_0000
USB2_1 UIB 寄存器	0x000_328C_0000

注：表中 USB2_0 对应数据手册中的 USB2_P3，表中 USB2_1 对应数据手册中的 USB2_P4。

表 5-19 USB2 寄存器列表

寄存器名称	偏移	描述
UIB 寄存器		
SECURE_CTRL_USB2_REG	0x0	USBHS0/USBHS1 控制器安全属性控制器寄存器；仅安全访问下可配
SECSID_ATST_USB2_REG	0x4	SECSID_ATST 为 SMMU 需要的控制信号；仅安全访问下可配
NSAID_SMMUID_USB2_REG	0x8	NSAID 为 LMU 所需要的安全控制信号；NSAID 与 SMMUID 及一些固定值共同组成 SMMU 所需的 streamid；仅安全访问下可配
ACE_USB2_REG	0xC	ACE 是系统需要的一些边带检测信号
USB2_WAKEUP_REG	0x10	USB2 唤醒寄存器
USB2_DEBUG_REG	0x14	USB2 调试寄存器
USB2_RSTN_REG	0x18	复位信号
USB2_HIGH_ADDR	0x1C	复位信号
USB2PHY_REFCLK_MODE	0x2C	USB2PHY 参考时钟选择
LPI_CTR_COUNTER	0x30	LPI 模块控制寄存器，连接 LPI_CTR 模块的 wait_cnt[31:0] 信号。用于 Usb2_ncc 时钟动态关断。该寄存器只有 USB2_PHY0 有。
LPI_CTR_EN	0x34	LPI 模块控制寄存器，连接 LPI_CTR 模块的 lpi_en 信号。用于 usb2_ncc 时钟动态关断。该寄存器只 USB2_PHY0 有。
OVERCURRENTN_CTRL	0x38	OVERCURRENT_N 引脚信号内部选通处理
控制器寄存器		

寄存器名称	偏移	描述
HCUSBCS	0x1A3	控制器控制和状态寄存器
HCPORTCTRL	0x1AB	端口控制寄存器
OTGCTRL	0x1BE	OTG 控制寄存器
HCUSBIEN	0x198	USB 中断使能寄存器
HCUSBIRQ	0x18C	USB 中断请求寄存器
HCINOMAXPCK	0x1E0	OUT 传输中端点 0 最大数据包大小配置寄存器
HCIN _x MAXPCK	0x1E2、0x1E3、 ... 0x1FE、0x1FF	OUT 传输中端点 x 最大数据包大小配置寄存器
HCOUT _x STADDR _L	0x344、0x348、 0x34C、0x350、 ... 0x378、0x37C	IN 传输开始低 8 位地址寄存器
HCOUT _x STADDR _H	0x345、0x349、 0x34D、0x351、 ... 0x379、0x37D	IN 传输开始高 8 位地址寄存器
HCOUT07IEN	0x194	端点 0~7 的中断使能寄存器
HCOUT815IEN	0x195	端点 8~15 的中断使能寄存器
HCOUT _x CON	0x00E、0x016、 0x01E、0x026、 ... 0x07E	IN 传输端点 1~15 控制寄存器
HCENDPRST	0x1A2	端点复位寄存器
HCIFIFOCTRL	0x1A8	控制器 FIFO 控制寄存器
HCOUT _x CTRL	0x0C4、0x0C8、 0x0CC、0x0D0、 0x0D4、... 0x0FC	端点 1~15 控制寄存器

注：表中 x 的取值为 1~15。

5.5.3 寄存器说明

5.5.3.1 SECURE_CTRL_USB2_REG (0x0)

域	位	读写	复位值	描述
Secure_resp	31:30	RW	0x0	从机总线返回状态 pslverr 返回第 31 位。 axi rresp 返回两位配置值
reserved	29:1	RO	0x0	保留
Secure_crtl	0	RW	0x0	0: 安全模式 1: 非安全模式

5.5.3.2 SECSID_ATST_USB2_REG (0x4)

域	位	读写	复位值	描述
reserved	31:2	RO	0x0	保留
Secsid_atst_usb2	1:0	RW	0x0	连接 awuser_usb2[1:0] 和 aruser_usb2 [1:0]

5.5.3.3 NSAID_SMMUID_USB_REG (0x8)

域	位	读写	复位值	描述
reserved	31:8	RO	0x0	保留
SMMUID	7:4	RW	0x5	连接 awuser_usb2[5:2] 和 aruser_usb2 [9:6]
Nsid	3:0	RW	0xE	连接 awuser_usb2[9:6] 和 aruser_usb2 [5:2]

5.5.3.4 ACE_USB2_REG (0xC)

域	位	读写	复位值	描述
reserved	31:28	RO	0x0	保留
arcache	27:24	RW	0xB	连接 ncc_usb2 中的 arcache_usb2
arbar	23:22	RW	0x0	连接 arbar_usb2[17:16]
arsnoop	21:18	RW	0x0	连接 arsnoop_usb2[15:12]
ardomain	17:16	RW	0x1	连接 ardomain_usb2[11:10]
reserved	15:12	RO	0x0	保留
awcache	11:8	RW	0x7	连接 ncc_usb2 中的 awcache_usb2
reserved	7	RO	0x0	保留
awbar	6:5	RW	0x0	连接 awbar_usb2[17:16]
awsnoop	4:2	RW	0x0	连接 awsnoop_usb2[15:12]
awdomain	1:0	RW	0x1	连接 awdomain_usb2[11:10]

5.5.3.5 USB2_WAKEUP_REG (0x10)

域	位	读写	复位值	描述
reserved	31:1	RO	0x0	保留
wakeup	0	RW	0x0	连接 wakeup 信号

5.5.3.6 USB2_DEBUG_REG (0x14)

域	位	读写	复位值	描述
reserved	31:27	RO	0x0	保留
tsmode	26:25	RW	0x0	TEST 模式选择缩短时间
tmodecustom	24	RO	0x0	TEST 模式指示
tmodeselcustom	23:16	RO	0x0	TEST 模式选择
reserved	15:14	RO	0x0	保留
otgstate	13:9	RO	0x0	OTG 状态机

域	位	读写	复位值	描述
downstrstate	8:5	R0	0x0	下游端口状态机
upstrstate	4:0	R0	0x0	上游端口状态机

5.5.3.7 USB2_RSTN_REG (0x18)

域	位	读写	复位值	描述
reserved	31:1	R0	0x0	保留
cfg_rstn_r	0	RW	0x1	usb2phy 配置接口复位，低有效

5.5.3.8 USB2_HIGH_ADDR (0x1C)

域	位	读写	复位值	描述
reserved	31:8	R0	0x0	保留
Addr_h	7:0	RW	0x0	系统 40 位地址，表示 DMA 操作的高 8 位地址

5.5.3.9 USB2PHY_REFCLK_MODE (0x2C)

域	位	读写	复位值	描述
reserved	31:1	R0	0x0	保留
refclk_mode	0	RW	0x1	连接 USB2PHY 的 REFCLK_MODE 信号 0: 输入时钟 REFCLK 是 25MHz 1: 输入时钟 REFCLK 是 12MHz

5.5.3.10 LPI_CTR_COUNTER (0x30)

域	位	读写	复位值	描述
wait_cnt	31:0	RW	0x0	连接 LPI_CTR 模块的 wait_cnt[31:0]

5.5.3.11 LPI_CTR_EN (0x34)

域	位	读写	复位值	描述
reserved	31:1	R0	0x0	保留
lpi_en	0	RW	0x0	连接 LPI_CTR 模块的 lpi_en 信号，用于时钟动态关断

5.5.3.12 OVERCURRENTN_CTRL (0x38)

域	位	读写	复位值	描述
reserved	31:2	R0	0x0	保留
overcurrentn_byps	1	RW	0x0	0: overcurrentn 信号来自 overcurrent_n pad 信号 1: overcurrentn 信号来自 overcurrentn_reg 的值

域	位	读写	复位值	描述
overcurrentn_reg	0	RW	0x0	overcurrentn_byps 为 1 有效, overcurrentn 信号寄存器值

5.5.3.13 HCUSBCS (0x1A3)

Host mode

域	位	读写	复位值	描述
wakesrc	7	RW	0x0	该 bit 为 1 表示唤醒引脚恢复 HC
reserved	6	RO	0x0	保留
sigrsume	5	RW	0x0	Sleep 状态: 写该 bit 为 1, CUSB2 初始化恢复信号。恢复期间硬件自动清除该 bit。 Suspend 状态: 使用 otgctrl[1] (abusdrop) bit, 强制恢复 A_SUPSPEND 状态的信号。
reserved	4:1	RO	0x0	保留
lsmode	0	RO	0x0	1: 设备工作在 LS 模式 0: 设备工作在 FS 或是 HS 模式

Device mode

域	位	读写	复位值	描述
wakesrc	7	RW	0x0	该 bit 为 1 表示唤醒引脚恢复 CUSB2。
discon	6	RW	0x1	软件断开连接 bit, 写 1 强制断开连接。
sigrsume	5	RW	0x0	远程唤醒 bit, 写该 bit 为 1, CUSB2 初始化远程唤醒信号。硬件自动清除该 bit。
reserved	4:2	RO	0x0	保留
lpmnyet	1	RW	0x0	1: CUSB2 发送 NYET 0: CUSB2 发送 ACK
lsmode	0	RO	0x0	1: 设备工作在 LS 模式; 0: 设备工作在 FS 或是 HS 模式

5.5.3.14 HCPORCTRL (0x1AB)

only-Host mode

域	位	读写	复位值	描述
1_6ms - 55ms	7:6	RW	0x1	USB 复位长度控制 00: CUSB2 信号 10ms 复位 01: CUSB2 信号 55ms 复位 (默认) 10: CUSB2 信号 1 到 6ms 复位 11: 无效
portrst	5	RW	0x0	端口复位 写该 bit 为 1, 强制复位端口信号
testm4 - testm0	4:0	RW	0x0	调试模式选择: 00000: 调试模式禁用 ----1: Test_J ---10: Test_K

域	位	读写	复位值	描述
				--100: Test_SE0_NAK -1000: Test_Packet 10000: Test_Force_Enable (发送 S0F 以确定设备是否连接)

5.5.3.15 OTGCTRL (0x1BE)

OTG mode

域	位	读写	复位值	描述
forcebconn	7	RW	0x0	写该 bit 为 1, 探测 B-device 连接, 使用 short debounce interval (TA_BCON_SDB)。
reserved	6	RO	0x0	保留
srpdatdeten	5	RW	0x0	Data 探测使能
srpvbusdeten	4	RW	0x0	VBUS 探测使能
bhnpen	3	RW	0x0	接收到 SetFeature(b_hnp_enable) 命令, 写该 bit 为 1。总线复位和会话结束时清除。
asetbhnpen	2	RO	0x0	当 B-device 设置 bhnpen bit, 写该 bit 为 1。总线复位和会话结束时清除。
abusdrop	1	RW	0x0	强制总线下电
busreq	0	RW	0x0	开始或终止会话

OTG2 mode

域	位	读写	复位值	描述
forcebconn	7	RW	0x0	写该 bit 为 1, 探测 B-device 连接, 使用 short debounce interval (TA_BCON_SDB)。
reserved	6	RO	0x0	保留
srpdatdeten	5	RW	0x0	Data 探测使能
reserved	4	RO	0x0	保留
bhnpen	3	RW	0x0	接收到 SetFeature(b_hnp_enable) 命令, 写该 bit 为 1。总线复位和会话结束时清除。
asetbhnpen	2	RO	0x0	当 B-device 设置 bhnpen bit, 写该 bit 为 1。总线复位和会话结束时清除。
abusdrop	1	RW	0x0	强制总线下电
busreq	0	RW	0x0	开始或终止会话

5.5.3.16 HCUSBIEN (0x198)

Host mode

域	位	读写	复位值	描述
reserved	7:6	RO	0x0	保留
hspeedie	5	RW	0x0	High speed 模式中断使能。该 bit 为 0, 忽略 hspeedir interrupt request。
uresie	4	RW	0x0	USB 复位中断使能。该 bit 为 0, 忽略 USB 复位

域	位	读写	复位值	描述
				的中断请求。
suspie	3	RW	0x0	USB 挂起中断使能。该 bit 为 0，忽略 USB 挂起中断请求。
reserved	2	RO	0x0	保留
sofie	1	RW	0x0	USB 起始帧中断使能。该 bit 为 1，忽略 hcfrmnrirq 中断请求。
reserved	0	RO	0x0	保留

Device mode

域	位	读写	复位值	描述
lpmie	7	RW	0x0	USB 链路电源管理中断使能。该 bit 为 0，忽略 lpmir 中断请求。
reserved	6	RO	0x0	保留
hspeedie	5	RW	0x0	High speed 模式中断使能。该 bit 为 0，忽略 hspeedir interrupt request。
uresie	4	RW	0x0	USB 复位中断使能。该 bit 为 0，忽略 USB 复位的中断请求。
suspie	3	RW	0x0	USB 挂起中断使能。该 bit 为 0，忽略 USB 挂起中断请求。
sutokie	2	RW	0x0	SETUP 令牌中断使能。该 bit 为 0，忽略 SETUP 令牌的中断请求。
sofie	1	RW	0x0	USB 起始帧中断使能。该 bit 为 1，忽略起始帧的中断请求。
sudavie	0	RW	0x0	SETUP 数据有效中断使能。该 bit 为 0，忽略 SETUP 数据的中断请求。

5.5.3.17 HCUSBIRQ (0x18C)

Host mode

域	位	读写	复位值	描述
reserved	7:6	RO	0x0	保留
hspeedir	5	RW	0x0	高速模式中断请求。进入高速模式，HC 置起该 bit 为 1。写该 bit 为 1，清除中断。
uresir	4	RW	0x0	复位中断请求。HC 停止 USB 总线复位型号，core 置起该 bit 为 1。写该 bit 为 1，清除中断。
suspir	3	RW	0x0	挂起中断请求。进入挂起信号，CUSB2 置起该 bit 为 1。写该 bit 为 1，清除中断。
reserved	2	RO	0x0	保留
sofir	1	RW	0x0	起始帧中断请求。SOF 包发送给 USB，HC 置起该 bit 为 1。写该 bit 为 1，清除中断。
reserved	0	RW	0x0	保留

Device mode

域	位	读写	复位值	描述
lpmir	7	RW	0x0	链路电源管理中断请求。在接收到 ACK 或是 NYET, CUSB 置起该 bit 为 1。
reserved	6	RO	0x0	保留
hspeedir	5	RW	0x0	高速模式中断请求。进入高速模式, HC 置起该 bit 为 1。写该 bit 为 1, 清除中断。
uresir	4	RW	0x0	复位中断请求。当探查到开始 USB 总线复位, CUSB2 置起该 bit 为 1。写该 bit 为 1, 清除中断。
suspir	3	RW	0x0	挂起中断请求。探查到挂起信号, CUSB2 置起该 bit 为 1。写该 bit 为 1, 清除中断。
sutokir	2	RW	0x0	SETUP 令牌包中断请求。当接收 SETUP 令牌包, CUSB2 置起该 bit 为 1。写该 bit 为 1, 清除中断。
sofir	1	RW	0x0	起始帧中断请求。当接收到 S0F 包, CUSB2 置起该 bit 为 1。写该 bit 为 1, 清除中断。
sudavir	0	RW	0x0	SETUP 数据有效中断请求。接收到错误的 SETUP 数据包, CUSB2 置起该 bit 为 1。写该 bit 为 1, 清除中断。

5.5.3.18 HCINOMAXPCK (0x1E0)

域	位	读写	复位值	描述
reserved	7	RO	0x0	保留
maxp6-maxp0	6:0	RW	0x0	OUT 0 最大包大小

5.5.3.19 HCIN_xMAXPCKl

($x=1$, 0x1E2) ($x=2$, 0x1E4) ($x=3$, 0x1E6) ... ($x=14$, 0x1FC) ($x=15$, 0x1FE)

域	位	读写	复位值	描述
Maxp7-maxp0	7:0	RW	0x0	OUT x 最大包大小低阶位 ($x=1-15$)

5.5.3.20 HCIN_xMAXPCKh

($x=1$, 0x1E3) ($x=2$, 0x1E5) ($x=3$, 0x1E7) ... ($x=14$, 0x1FD) ($x=15$, 0x1FF)

域	位	读写	复位值	描述
reserved	7:3	RO	0x0	保留
Maxp10-maxp8	2:0	RW	0x0	OUT x 最大包大小高阶位 ($x=1-15$)

5.5.3.21 HCOU_xSTADDRL

($x=1$, 0x344) ($x=2$, 0x348) ($x=3$, 0x34C) ... ($x=14$, 0x378) ($x=15$, 0x37C)

域	位	读写	复位值	描述
Start addr	7:2	R/W	0x0	IN 传输开始地址低位 ($x=1\sim15$)
reserved	1:0	R/W	0x0	保留

5.5.3.22 HCOUT_xSTADDRH

($x=1$, 0x345) ($x=2$, 0x349) ($x=3$, 0x34D) ... ($x=14$, 0x379) ($x=15$, 0x37D)

域	位	读写	复位值	描述
Start addr	7:0	R/W	0x0	IN 传输开始地址高位 ($x=1\sim15$)

5.5.3.23 HCOUT07IEN (0x194)

Host mode

域	位	读写	复位值	描述
Hcout _x ien ($x=7-0$)	7:0	RW	0x0	HCOUT x 端点中断使能 ($x=7-0$)

Device mode

域	位	读写	复位值	描述
In _x ien ($x=7-0$)	7:0	RW	0x0	IN x 端点中断使能 ($x=7-0$)

5.5.3.24 HCOUT815IEN (0x195)

Host mode

域	位	读写	复位值	描述
hcout _x ien ($x=15-8$)	7:0	RW	0x0	HCOUT x 端点中断使能 ($x=15-8$)

Device mode

域	位	读写	复位值	描述
hcout _x ien ($x=15-8$)	7:0	RW	0x0	IN x 端点中断使能 ($x=15-8$)

5.5.3.25 HCOUT_xCON

($x=1$, 0x00E) ($x=2$, 0x016) ($x=3$, 0x01E) ($x=4$, 0x026) ... ($x=15$, 0x07E)

Host mode

域	位	读写	复位值	描述
reserved	7:6	R0	0x0	保留
isod1 - isod0	5:4	RW	0x0	仅支持同步传输。确定每个微帧内传输发的包的数量。 isod1, iso0=00, 每微帧 1 个 ISO 包 isod1, iso0=01, 每微帧 2 个 ISO 包 isod1, iso0=10, 每微帧 3 个 ISO 包
type1 - type0	3:2	RW	quad buffering:2 triple buffering:2	选择端点类型 type1, type0=10, 批量端点

域	位	读写	复位值	描述
			double buffer:2 single buffering:2 默认: 0	type1, type0=01, 同步端点 type1, type0=11, 中断端点
buf1 - buf0	1:0	RW	quad buffering:3 triple buffering:2 double buffer:1 single buffering:0 默认: 0	选择 buffering 类型 buf1, buf0=00, single buffering buf1, buf0=01, double buffering buf1, buf0=10, triple buffering buf1, buf0=11, quad buffering

Device mode

域	位	读写	复位值	描述
val	7	RW	0x0	IN x 端点有效。该 bit 为 1, 表示使能端点, 0 表示禁用端点。
stall	6	RW	0x0	IN x 端点停止。该 bit 为 1, 表示 CUSB2 返回 STALL。
isod1 - isod0	5:4	RW	0x0	仅对高速模式的同步 IN 端点有效, 确定每个微帧内传输发的包的数量。 isod1, iso0=00, 每微帧 1 个 ISO 包 isod1, iso0=01, 每微帧 2 个 ISO 包 isod1, iso0=10, 每微帧 3 个 ISO 包
type1 - type0	3:2	RW	quad buffering:2 triple buffering:2 double buffer:2 single buffering:2 默认: 0	选择端点类型 type1, type0=10, 批量端点 type1, type0=01, 同步端点 type1, type0=11, 中断端点
buf1 - buf0	1:0	RW	quad buffering:3 triple buffering:2 double buffer:1 single buffering:0 默认: 0	选择 buffering 类型 buf1, buf0=00, single buffering buf1, buf0=01, double buffering buf1, buf0=10, triple buffering buf1, buf0=11, quad buffering

5.5.3.26 HCENDPRST (0x1A2)

Host mode

域	位	读写	复位值	描述
togsetq	7	RW	0x0	读请求: 数据 toggle 值 写请求: toggle 设置值
fiforst	6	WO	0x0	Fifo 复位
togrst	5	WO	0x0	Toggle 复位
hcio	4	RW	0x0	端点方向 0=HCIN; 1=HCOUT
ep3, ep2, ep1, ep0	3:0	RW	0x0	端点数量, 有效值 0-15

Device mode

域	位	读写	复位值	描述
togsetq	7	RW	0x0	读请求: 数据 toggle 值 写请求: toggle 设置值
fiforst	6	WO	0x0	Fifo 复位
togrst	5	WO	0x0	Toggle 复位
io	4	RW	0x0	端点方向 0=IN 1=OUT
ep3, ep2, ep1, ep0	3:0	RW	0x0	端点数量, 有效值 0-15

5.5.3.27 HCFIFOCTRL (0x1A8)

Host mode

域	位	读写	复位值	描述
fifoacc	7	RW	0x0	fifoaccess 位
fifocmit	6	WO	0x0	fifo commit 位
fifoauto	5	WO	0x0	fifoauto 位
hcio	4	WO	0x0	端点方向 0=HCIN 1=HCOUT
ep3, ep2, ep1, ep0	3:0	WO	0x0	端点数量, 有效值 0-15

Device mode

域	位	读写	复位值	描述
fifoacc	7	RW	0x0	fifoaccess 位
fifocmit	6	WO	0x0	fifo commit 位
fifoauto	5	WO	0x0	fifoauto 位
io	4	WO	0x0	端点方向 0=IN 1=OUT
ep3, ep2, ep1, ep0	3:0	WO	0x0	端点数量, 有效值 0-15

5.5.3.28 HCOUxCTRL

(x=1, 0x0C4) (x=2, 0x0C8) (x=3, 0x0CC) ... (x=14, 0x0F8) (x=15, 0x0FC)

Only-Host mode

域	位	读写	复位值	描述
reserved	7:4	RO	0x0	保留
endpnr3, endpnr2, endpnr1, endpnr0	3:0	RW	0x0	配置端点号

5.6 USB HUB 控制器

USB HUB 控制器位于 USB P2 接口下。USB HUB 的工作模式示意图如下图所示。USB HUB 控制器有两种工作模式：USB HUB 模式和 HOST 模式。USB HUB 模式下，USB HUB 可以看做有 3 个独立的 Device，分别对应 usb_vhub0、usb_vhub1、usb_vhub2，通过软件模拟成各类 USB 设备使用；HOST 模式下，可以使用 usb_vhub0（对应数据手册中 USB2_P2）作为一个 USB2.0 HOST 控制器。可以在设备初始化时通过配置数据选择器进行模式选择。

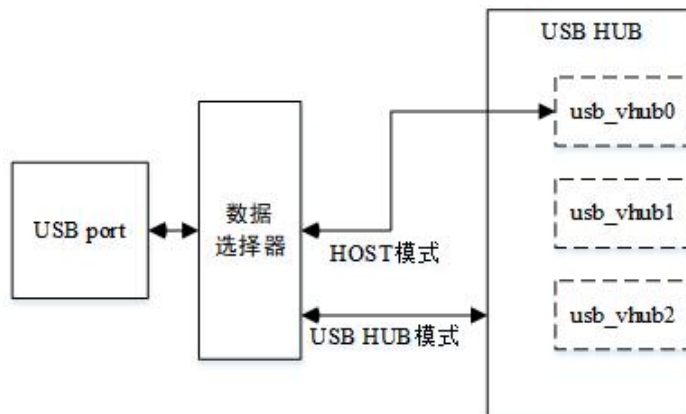


图 5-9 USB HUB 工作模式示意图

5.6.1 操作说明

5.6.1.1 初始化操作说明

USB HUB 控制器的初始化配置主要是用来配置 USB HUB 的两种工作模式，所以有两种初始化配置操作。

HOST 配置模式下的操作如下：

HOST 工作模式需要配置 VHUB_CFG 寄存器以及 DEV1_CFG 空间中的 GEN_CFG、UTMIP_CFG 寄存器。

- 上电复位；将 VHUB_CFG 寄存器中的 host_dev_sel 位 (VHUB_CFG[0]) 配置成 1，其余位保持不变，将 USB HUB 选择 Device1 进行输出。
- 将 UTMIP_CFG 寄存器中的 utmiavalid 位和 utmivbusvalid 位 (UTMIP_CFG[3]、UTMIP_CFG[1]) 配置成 1，其余位保持不变，将 Device1 配置成 HOST 模式。
- 将 VHUB_CFG 控制器中的 phy_cfg_rstn 位 (VHUB_CFG[11]) 以及 GEN_CFG 寄存器中的 dev_reset 位 (GEN_CFG[0]) 配置成 1，其余位保持不变，释放 PHY 和 USB 控制器复位。
- 其余操作与 USB2.0 OTG 控制器一致，请参考 USB2.0 OTG 控制器初始化操作。
- USB HUB 模式下的操作如下：

- 上电复位；分别将 3 个设备的 GEN_CFG 寄存器中的 keep_disconnect 位 (GEN_CFG[7]) 置 1，使得当前状态下 Host 不枚举设备。
- 分别将 3 个设备的 UTMIP_CFG 寄存器中的 utmibvalid 位和 utmivbusvalid 位 (UTMIP_CFG[2]、UTMIP_CFG[1]) 配置成 1，其余位保持不变，将 3 个设备配置成 Device 模式。
- 将 VHUB_CFG 寄存器中的 hub_rstn、phy_cfg_rstn 位 (VHUB_CFG[12]、VHUB_CFG[11]) 以及 3 个设备的 GEN_CFG 寄存器中的 dev_reset 位 (GEN_CFG[0]) 配置成 1，其余位保持不变，释放 PHY 和 USB HUB 控制器复位。
- 其余操作与 USB2.0 OTG 控制器一致，请参考 USB2.0 OTG 控制器初始化操作。操作对应设备时需要配置对应设备的 GEN_CFG 寄存器中的 keep_disconnect 位 (GEN_CFG[7]) 为 0，使得 HOST 能够枚举该设备。

5.6.1.2 数据传输操作说明

数据传输操作同 USB2.0 OTG 控制器，请参考 USB2.0 OTG 控制器操作说明。

5.6.2 寄存器列表

USB HUB 的寄存器空间共有 7 段，基址如下表所示：

表 5-20 USB HUB 寄存器基地址

名称	基地址
USB_VHUB0	0x000_3180_0000
USB_VHUB1	0x000_3188_0000
USB_VHUB2	0x000_3190_0000
VHUB_CFG	0x000_319C_0000
DEV1_CFG	0x000_3199_0000
DEV2_CFG	0x000_319A_0000
DEV3_CFG	0x000_319B_0000

其中 USB_VHUB0、USB_VHUB1、USB_VHUB2 的寄存器与 5.5 章节中 USB2_0、USB2_1 一致，请参考 5.5.3 章节中的寄存器说明。

表 5-21 VHUB_CFG 寄存器列表

寄存器名称	偏移	描述
VHUB_CFG	0x0	配置 USB HUB 使用模式、控制复位以及观测状态的寄存器

表 5-22 DEV(1~3)_CFG 寄存器列表

寄存器名称	偏移	描述
GEN_CFG	0x0	USB 设备 1~3 的通用配置寄存器

寄存器名称	偏移	描述
GEN_STA	0x4	USB 设备 1~3 的通用状态寄存器
DBG_STA	0x10	调试状态寄存器
UTMIP_CFG	0x20	UTMI+信号配置寄存器，用于配置 USB 设备的工作模式
UTMIP_STA	0x24	UTMI+信号状态寄存器

5.6.3 寄存器说明

5.6.3.1 VHUB_CFG (0x0)

域	位	读写	复位值	描述
reserved	31:13	RO	0x0	保留
hub_rstn	12	RW	0x0	USB HUB 的复位信号 0: 复位 1: 释放复位
phy_cfg_rstn	11	RW	0x0	PHY 和配置模块的复位信号 0: 复位 1: 释放复位
host_disconnect	10	RO	0x0	PHY 输出的 host 断开连接指示信号 0: 连接 1: 断开
sessend	9	RO	0x0	PHY 输出的 sessend 信号
sessvalid	8	RO	0x0	PHY 输出的 sessvalid 信号
bvalid	7	RO	0x0	PHY 输出的 bvalid 信号
vbus_valid	6	RO	0x0	PHY 输出的 vbus_valid 信号
iddig	5	RO	0x0	PHY 输出的 iddig 信号
refclk_mode	4	RW	0x0	PHY 参考时钟选择 0: 25MHz 1: 12MHz
pll_en	3	RW	0x1	PHY PLL 使能控制 0: 不使能 1: 使能
reserved	2	RO	0x0	保留
txbitstuff_enable	1	RW	0x0	PHY 的 txbitstuff 使能信号 0: 不使能 1: 使能
host_dev_sel	0	RW	0x0	USB HUB 或 HOST 模式选择信号 0: USB HUB 模式 1: Host 模式

5.6.3.2 GEN_CFG (0x0)

域	位	读写	复位值	描述
reserved	31:8	RO	0x0	保留

域	位	读写	复位值	描述
keep_disconnect	7	RW	0x1	USB HUB 模式下： 上电复位后让 HOST 端不枚举该 USB 设备， 在 USB 设备准备好（如模拟好对应设备）后 释放该信号，使 Host 能够枚举到模拟设备 （该信号只能在 USB HUB 全体复位后生效 1 次）。 0：可被 HOST 检测到 1：不可 HOST 检测到 HOST 模式下：保留
Reserved	6	RO	0x1	保留
bypass_sleepm	5	RW	0x1	旁路输入的 sleepm 信号 0：非旁路 1：旁路
overcurrent_sel	4	RW	0x1	选择输入到设备的 overcurrent 信号来自 PHY 或是来自配置寄存器 0：PHY 1：配置寄存器
overcurrent_cfg	3	RW	0x0	配置寄存器的 overcurrent 信号
wakeup_rst	2	RW	0x0	控制器的唤醒模块的复位信号 0：复位 1：释放复位
send_wakeup	1	RW	0x0	非 USB HUB 模式下发出 wakeup 信号的配置 寄存器 0：不发 1：发出
dev_rst	0	RW	0x0	对应设备的复位控制 0：复位 1：释放复位

5.6.3.3 GEN_STA (0x04)

域	位	读写	复位值	描述
reserved	31:2	RO	0x0	保留
workaroundrst_state	1	RO	0x0	设备对 PHY 的复位状态指示信号 0：复位完成 1：复位中
softrst_state	0	RO	0x0	设备软复位的复位状态指示信号 0：复位完成 1：复位中

5.6.3.4 DBG_STA (0x10)

域	位	读写	复位值	描述
reserved	31:2	RO	0x0	保留

域	位	读写	复位值	描述
tmodecustom	24	R0	0x0	TEST 模式指示
tmodeselcustom	23:16	R0	0x0	TEST 模式选择
reserved	15:14	R0	0x0	保留
otgstate	13:9	R0	0x0	OTG 状态机 0x00: A_IDLE 0x01: A_WAIT_VRISE 0x02: A_WAIT_BCON 0x03: A_HOST 0x04: A_SUSPEND 0x05: A_PERIPHERAL 0x06: A_VBUS_ERR 0x07: A_WAIT_VFALL 0x10: B_IDLE 0x11: B_PERIPHERAL 0x12: B_WAIT_ACON 0x13: B_HOST_1 0x14: B_HOST_2 0x15: B_SRP_INIT1 0x16: B_SRP_INIT2 0x17: B_DISCHRG1 0x18: B_DISCHRG2
downstrstate	8:5	R0	0x0	HOST 模式的端口状态机 0x00: DISABLED 0x01: DDRISE_SE0 0x02: CLR_TIMER1 0x03: RUN_TIMER1 0x04: WAITD 0x05: DEV_DET 0x06: FULL_SPEED 0x07: LOW_SPEED 0x08: DR_CHIRP_J 0x09: DR_CHIRP_K 0x0A: DRIVE_SE0_HS 0x0B: HIGH_SPEED 0x0C: LPM_SUSPEND 0x0D: LPM_DRIVE_RESUME
upstrstate	4:0	R0	0x0	Device 模式下端口状态机 0x00: DISABLED_U 0x01: TIMERRESET 0x02: SE0_STATE 0x03: J_STATE 0x04: HIGH_SPEED_U 0x05: HIGH_SPEED_SE0 0x06: REMOVE_HS 0x07: SUSPENDED

域	位	读写	复位值	描述
				0x08: WAIT_FILT 0x09: DETECT_SE0 0x0A: HANDSHAKE 0x0B: DETECT_HSJ 0x0C: DETECT_HSK 0x0D: INCREASE 0x0E: WAIT_END_RESET 0x0F: DRIVE_RESUME 0x10: RESUME 0x11: SLEEP_RESUME 0x12: SLEEP

5.6.3.5 UTMIP_CFG (0x20)

域	位	读写	复位值	描述
reserved	31:6	RO	0x0	保留
utmiiddig	5	RW	0x0	0: 设备为 HOST 模式 1: 设备为 Device 模式
reserved	4	RO	0x0	保留
utmiavalid	3	RW	0x0	HOST 模式下配置成 1 USB HUB 模式下配置成 0
utmibvalid	2	RW	0x0	HOST 模式下配置成 0 USB HUB 模式下配置成 1
utmivbusvalid	1	RW	0x0	0: 设备非工作模式 1: 设备工作模式
utmisessend	0	RW	0x0	sessend 信号，正常工作模式下配置成 0

5.6.3.6 UTMIP_STA (0x24)

域	位	读写	复位值	描述
reserved	31:7	RO	0x0	保留
utmiidpullup	6	RO	0x0	设备输出的 idpullup 信号
utmidppulldown	5	RO	0x0	设备输出的 dppulldown 信号
utmidmpulldown	4	RO	0x0	设备输出的 dmpulldown 信号
utmidrvvbus	3	RO	0x0	设备输出的 drvbus 信号
utmichrgvbus	2	RO	0x0	设备输出的 chrgvbus 信号
utmidischrgvhus	1	RO	0x0	设备输出的 dischrgvhus 信号
utmisleepm	0	RO	0x0	设备输出的 sleepm 信号

5.7 SATA 控制器

飞腾派实现了两个 SATA 控制器，每个控制器只包含一个 PORT，控制器设计兼容 SATA3.0 规范，寄存器部分兼容 AHCI1.31 规范中的 HBA 内存寄存器（HBA Memory

Registers)，无 PCI Header 部分（HBA Configuration Registers），因此不可当做 pci 设备，只能当平台设备使用，控制器不支持 IDE 模式（SATA Legacy Mode，即不可用原生命令直接操作控制器），控制器不支持硬件 RAID 功能。详细内容可参考飞腾派数据手册、《Serial ATA International Organization: Serial ATA Revision 3.0》和《Serial ATA Advanced Host Controller Interface (AHCI) 1.3.1》。

SATA 寄存器基地址如下表所示。

表 5-23 SATA 寄存器基地址

名称	基地址
SATA0	0x000_31A4_0000
SATA1	0x000_3201_4000

5.8 DC

5.8.1 简介

DC 是一个显示控制器，主要完成将 CPU/VPU 处理后的图像数据，按照 Display 协议处理后送给 DP PHY 接入显示器。

飞腾派 DC 主要特性：

- 支持两路 Display，两路 Display 互相独立；
- 支持的 size 有 640×480、800×600、1024×768、1152×864、1280×720、1280×960、1366×768、1440×900、1600×900、1920×1080，最大帧率为 5.4Gbps/(size*32bit)；
- Display{0, 1} 有单独鼠标图层；
- 支持 Linear 显示格式；
- 支持 Frame Buffer 两个图层单独时钟使能；
- 支持 Hsync, Vsync, DE 可配置，参数配置满足协议；
- 输入图像格式：
 - ARGB2101010, A/XRGB8888, A/XRGB1555, A/XRGB4444, RGB565
 - YUV422 (YUY2, UYVY, NV16)
 - YUV420 (YUY2 (P010) YV12, NV12, NV21)
- 输出像素 DP 接口 48bit, RGB 分别 16bits 有效；
- 输出支持像素格式：RGB2101010, RGB888, RGB666, RGB555；
- 输出支持 RGB Swizzle；
- 支持水平/垂直翻转，不支持 90° 旋转；
- 支持缩放功能，缩放倍数小于 3；

- 支持 Dither, Gamma 图像增强功能。

飞腾派集成的 DP 兼容 DisplayPort1.4/Embedded DisplayPort1.4 协议, 支持特性请参考飞腾派数据手册。

5.8.2 操作说明

5.8.2.1 整体初始化流程

1. 如有必要, 进行 DC 模块的软复位;
2. 根据所选用的分辨率设置对应的像素时钟;
3. DP 初始化;
4. DC 初始化。

初始化完成后, 如果不需要切换分辨率, 则 DP 配置及 DC 的大部分配置不需要变化, 只需要根据 DC 的图层配置相关的 double buffer registers 机制进行切帧配置即可正常工作。

5.8.2.2 DP 初始化流程

1. phy 复位;
2. DPTX core 初始化, 包括 APB 时钟分频, 分频后给 AUX 通道提供时钟, 并使能 DPTX;
3. 检测 HPD 信号, 如果有则继续初始化 DP, 如果没有, 则退出初始化; HPD 的检测应该在软件热插拔机制中同时实现;
1. 通过 AUX 通道读取 DPCD 和 EDID 信息;
2. 配置 DPTX 的 lane count 和 link rate;
3. 配置 RX (sink) 端的 lane count、link rate, 并进行 phy 的链路训练;
4. 如果有音频, 则配置 DPTX 的 secondary channel;
5. 配置 DPTX 的视频信息参数, main stream;
6. 对 link 进行 soft reset。

5.8.2.3 DP link training 流程

1. 检测到 HPD 后, AUX 读 Sink DPCD 0x00000-0x0000D, 读取 sink 设备信息;
2. AUX 写 sink DPCD 0x00100-0x00101, 配置 bandwidth 和 lane count;
3. 关闭 source 端的置扰 (Scrambling_disable = 0x01), 选择 TPS1 (TRAINING_PATTERN_SET = 0x01), source 开始发送 Training Pattern 1,

- AUX 写 DPCD 0x00102 选择 Training Pattern 1, 并选择 disable scrambling;
4. 写 DPCD 0x00103-0x00106 分别配置 lane0、lane1、lane2、lane3 的 Voltage swing 和 Pre-emphasis;
 5. 读 DPCD 0x0000E, TRAINING_AUX_RD_INTERVAL 的值;
 6. 等待由 TRAINING_AUX_RD_INTERVAL 确认的时间间隔后, 读 DPCD 0x00202-0x00207 获取 link status;
 7. 如果一个或以上的 lane 的 Status 读出的 CR_DONE 为 0, 则时钟恢复失败, 则通过下述三步处理:
 - a. 检查 DPCD 0x00206-0x00207 的值, 并调整 DPCD 0x00103-0x00106 的配置;
 - b. 若 source 已经发送 5 次 TPS1, 则降低 link bandwidth (0x00100), 从第 3 步开始重新训练;
 - c. 若已经降低到最低带宽 1.62Gbps, 则训练失败; CR_DONE 通过, 则时钟恢复完成后, 开始 Training Pattern 2/3/4, source 发送 Training Pattern 1, 同时 AUX 写 DPCD 0x00102 选择 Training Pattern 2/3/4, 并选择 disable scrambling (若 source 和 sink 都支持 HBR2, 则 TPS2 替换为 TPS3, 若都支持 HBR3, 则 TPS2 替换成 TPS4)。
 8. 写 DPCD 0x00103-0x00106 分别配置 lane 0、lane 1、lane 2、lane 3 的 voltage swing 和 pre-emphasis;
 9. 等待由 TRAINING_AUX_RD_INTERVAL 确认的时间间隔后, 读 DPCD 0x00202-0x00207 获取 link status;
 10. 如果一个或以上的 lane 的 Status 读出的 CR_DONE 为 0, 即时钟恢复失败, 终止 TPS2, 重新开始 TPS1;
 11. 如果 CR_DONE 通过, 检查 CHANNEL_EQ_DONE、SYMBOL_LOCKED;
 12. 若一个及以上的 lane 的 status 读出的 CHANNEL_EQ_DONE、SYMBOL_LOCKED 失败, 则通过以下三种方法处理:
 13. 检查 DPCD 0x00206-0x00207 的值, 并调整 DPCD 0x00103- 0x00106 的配置;
 14. 若已经发送 5 次 TPS2, 则降低 link bandwidth (0x00100), 从第 3 步开始重新训练;
 15. 若已经降低到最低带宽 1.62Gbps, 则训练失败。
 16. 若 CHANNEL_EQ_DONE、SYMBOL_LOCKED 表示训练完成, 则写 DPCD 0x00102 关闭 link training;
 17. source 端寄存器 Scrambling_Disable = 0x00, TRAINING_PATTERN_SET = 0x0。
- 注意: DPCD 偏移的寄存器详细信息请参考 DP1.4 规范。

5.8.2.4 DP 视频相关配置

1. 根据每个 source 的视频属性、分辨率、视频格式，配置主数据流属性相关寄存器 0x180—0x1C4，其中分辨率的配置已和 DC 的分辨率配置一起说明。
2. 设置 TU 格式，根据显示模式和带宽计算一个 TU 内有效的 symbol 个数，以 ARGB8888 为例：

$\text{Link symbol rate} = \text{lane count} * \text{link rate} * 100$

$\text{Vid symbol rate} = (\text{pixel clock} * \text{bpc} * 3) / 8$

$\text{Data per tu} = (\text{vid symbol rate} / \text{link symbol rate}) * \text{TU size}$

Data per tu 整数副本写入 TRANSFER_UNIT_CONFIG_SRC_0. symbols_per_tu, 小数部分向下取整写入 TRANSFER_UNIT_CONFIG_SRC_0. frac_symbols_per_tu, frac_symbols_per_tu. transfer_unit_size 一般设置为 64。

5.8.2.5 音频配置

5.8.2.5.1 DP 音频时钟同步模式

1. SEC_INPUT_SELECT (0x304) 设置音频输入模式选择 I2S;
2. SEC_CHANNEL_COUNT (0x308) 设置声道数;
3. SEC_AUDIO_CHANNEL_MAP (0x35c), 设置声道映射;
4. SECONDARY_DATA_WINDOW (0x08c), 设置次要数据窗口大小;
5. SEC_CS_CATEGORY_CODE (0x344), 设置 8 bit category code;
6. SEC_MAUD (0x318) 写入 M 值, SEC_NAUD (0x31c) 写入 N 值, SEC_AUDIO_CLOCK_MODE (0x320) 设置同步模式;
7. SEC_TIMESTAMP_INTERVAL (0x33c), 设置发送 ATS 的间隔时间, 0 表示只在每个消隐期间发送一次;
8. SEC_INFOFRAME_SELECT (0x334) 选择要写入的 INFOFRAME 类型, 0 对应 vender specific information (VSI), 1 对应 AUX Video Information (AVI), 2 对应 Source Product 描述, 3 对应 Audio 描述, 4 对应 NTSC VBI, 在这些类型中 Audio 描述是必选的, 其他可选;
9. SEC_INFOFRAME_DATA (0x338) 写入 information data, 依次写入 16 或 32 字节的 payload data。Payload 从低位到高位开始写, 每次写入 1 个字节, AVI 和 AUD 的长度为 16 字节, VSI、SDP、NTSC VBI 长度为 32 字节, 不足的补 0。即每种 Infoframe 必须写入 0x338 寄存器 16 或 32 次。每种类型 inforframe 的值和意义参考 CEA 861 -E;

10. 0x344 配置 Channel Status;
11. SECONDARY_STREAM_ENABLE (0x088) 使能音频输入 (和 0x084 一起使能),
SEC_INFOFRAME_RATE (0x314) 设置各 Infoframe data 发送频率,
SEC_INFOFRAME_ENABLE (0x310) 使能对应的 Infoframe 类型;
12. 如果只发送音频, 0x1c8 写 2。音视频都发送, 忽略此步骤;
13. 等待 500us;
14. SEC_AUDIO_ENABLE (0x300) 使能次要数据通道;
15. 使能 I2S 数据的输入, 开始音频数据传输。

5.8.2.5.2 音频时钟异步模式

1. SEC_INPUT_SELECT (0x304) 设置音频输入模式选择, 目前只支持 I2S (写 0);
2. SEC_CHANNEL_COUNT (0x308) 设置声道数;
3. SEC_AUDIO_CHANNEL_MAP (0x35c) 设置声道映射;
4. SECONDARY_DATA_WINDOW (0x08c) 设置次要数据窗口大小:
不同速率下 SECONDARY_DATA_WINDOW 寄存器配置值的计算方法:
$$Hblank = (hbp + hsw + hfp) * 1000 / fpixel_clock$$

Case (Link rate)

 1. 62: $hblank / 24.69$
 2. 7: $hblank / 14.81$
 5. 4: $hblank / 7.4$
 8. 1: $hblank / 4.94$

Endcase

$$Hblank = hblank * 0.9$$
5. SEC_CS_CATEGORY_CODE (0x344) 设置 8-bit category code;
6. 设置时钟模式: SEC_AUDIO_CLOCK_MODE (0x320) (异步);
7. SEC_TIMESTAMP_INTERVAL (0x33c) 设置发送 ATS 的时间间隔, 单位为 us。设为 0 时, 仅在每个场消隐期间发送一次;
8. SEC_INFOFRAME_SELECT (0x334) 选择要写入的 INFOFRAME 类型, 0 对应 vender specific information (VSI), 1 对应 AUX Video Information (AVI), 2 对应 Source Product 描述 (SDP), 3 对应 Audio 描述 (AUD), 4 对应 NTSC VBI, 在这些类型中, Audio 描述是发送音频时必需的, 其他都是可选的;
9. SEC_INFOFRAME_DATA (0x338) 写入 Infoframe data, 依次写入 16 或 32 字节的 payload data。payload 从低位到高位开始写, 每次写入 1 字节, AVI 和 AUD 的长度为 16 字节, VSI、SDP、NTSC VBI 的长度为 32 字节, 不足的补 0, 即每

- 种 Infoframe 必须写入 0x338 寄存器 16 或 32 次。每种类型 inforframe 的值和意义参考 CEA 861 -E;
10. 0x344 配置 Channel Status;
 11. SECONDARY_STREAM_ENABLE (0x088) 使能音频输入 (和 0x084 一起使能), SEC_INFOFRAME_RATE (0x314) 设置各 Infoframe data 发送频率, SEC_INFOFRAME_ENABLE 0x310 使能对应的 Infoframe 类型;
 12. 如果只发送音频, 0x1c8 写 2, 如果音视频都发送, 忽略此步骤;
 13. SEC_AUDIO_ENABLE (0x300) 使能次要数据通道;
 14. I2S SCLK 时钟开始输入, 等待至少 1620us (异步模式, 控制器在 1.62Gbps 的速率产生第一个有效的 MAUD 所需的时间, 2.7Gbps 需 971us, 5.4Gbps 需 485us, 8.1Gbps 需 324us);
 15. 使能 I2S 数据的输入, 开始音频数据传输。

5.8.2.5.3 音频数据属性发生改变

支持的改变包括采样频率和声道数的变化。若输入的音频发生改变, 0x300 写 0, 在 DP 下一次发送的 VB-ID 中 AudioMute_Flag 将变成 0。

SCLK 异步: 待改变的音频稳定后, 0x300 写 1, 等待至少 1620us (SCLK 频率可能改变, 因此要重新计算 Maud), 使能 I2S 输入。

SCLK 同步: 待改变的音频稳定后, 0x300 写 1, 使能 I2S 输入。

5.8.2.6 正常帧配置流程

1. 配置寄存器 dc0_pixel0_clk_config(dc1_pixel1_clk_config) 像素时钟频率, 等待 1ms 后, 确认 ack 是否为 1, 为 1 表示 PLL 时钟已稳定;
2. 配置 DC0IntrEnable (DC1IntrEnable) 中断使能寄存器 32'h1;
3. 配置 DC0FrameBufferSize (DC1FrameBufferSize) Framebuffer 窗口尺寸, 其中 Height[29:15] 和 Width[14:0], 以 640×480 为例, 配置为 32'h00F00280;
4. 配置 DC0FrameBufferConfig (DC1FrameBufferConfig) 帧缓冲属性配置寄存器, 参考寄存器手册;
5. 配置 DC0FrameBufferStride (DC1FrameBufferStride) 每行字节数大小 FramebufferStride;
6. 配置 DC0FrameBufferAddress (DC1FrameBufferAddress) Framebuffer 的起始地址 FrameAddress, 128Bytes 对齐;
7. 配置 DC0DPConfig (DC1DPConfig) DP 接口数据格式;
8. 参考 CEA 或者 VESA 标准配置行场同步信号;

9. 配置 DC0HSync (DC1HSync) 水平同步计数器;
10. 配置 DC0HDisplay (DC1HDisplay) 水平总数和显示结束计数器;
11. 配置 DC0VSync (DC1VSync) 垂直同步计数器;
12. 配置 DC0VDisplay (DC1VDisplay) 垂直总数和显示结束计数器;
13. 配置 DC0FrameBufferConfig (DC1FrameBufferConfig) 帧缓冲属性配置, 保持第 3 步原有值, 仅将 bit[4] 置 1'b1, 开始工作;
14. 可以配置第二帧数据, 重复 1~8 步骤;
15. 等待 vUpdate 帧完成中断, 配置第三帧数据, 重复 1~8 步骤;
16. 然后收到每一次帧完成中断, 配置下一帧数据...

5.8.2.7 切帧配置流程

1. 原有帧在正常发送过程中;
2. 配置下一帧为 clear 帧;
3. 再一次配置下一帧为 clear 帧;
4. 收到帧结束中断, 间隔 500ns, 配置寄存器 AQHiClockControl 的 bit[18:17] 相应的 core0 或者 core1 进行复位相应的 core 逻辑;
5. 配置新分辨率对应的像素时钟请求及像素时钟频率数值, 即配置寄存器 dc0_pixel0_clk_config(core0) 或 dc1_pixel1_clk_config(core1) 的 bit[30] 为 1'b1 和 bit[29:0];
6. 间隔 1us 读寄存器 dc0_pixel0_clk_config(core0) 或 dc1_pixel1_clk_config(core1) 的 bit[31], 如果为 1'b1, 表明像素时钟已稳定;
7. 按照正常帧配置流程配置下一帧即可。

5.8.3 DC 寄存器列表

下表所示为 DC 寄存器列表, 支持 D1Therd、Gamma、DP 输出模式、scale 滤波等配置, 部分寄存器具有 double buffered 功能。

表 5-24 DC 寄存器基地址

名称	基地址
DC	0x000_3200_0000

表 5-25 DC 寄存器列表

寄存器名称	偏移	描述	double buffered
通用配置寄存器			
DC0GeneralConfig	DC0:0x14B0	通用配置	是
DC1GeneralConfig	DC1:0x24B0		

寄存器名称	偏移	描述	double buffered
AQHiClockControl	0x0000	时钟控制	否
像素时钟配置寄存器			
dc0_pixel0_clk_config	DC0:0x00F0	DC0 像素时钟配置	否
dc1_pixel1_clk_config	DC1:0x00F4	DC1 像素时钟配置	否
Frame Buffer 寄存器			
DC0FrameBufferAddress	DC0:0x1400	Frame Buffer 起始地址低 32 位	是
DC1FrameBufferAddress	DC1:0x2400		
DC0FrameBufferHiAddress	DC0:0x1404	Frame Buffer 起始地址高 8 位	是
DC1FrameBufferHiAddress	DC1:0x2404		
DC0FrameBufferStride	DC0:0x1408	Frame Buffer Stride	是
DC1FrameBufferStride	DC1:0x2408		
DC0FrameBufferConfig	DC0:0x1518	Frame Buffer 属性配置	个别位是
DC1FrameBufferConfig	DC1:0x2518		
DC0FrameBufferColorKey	DC0:0x1508	Frame Buffer Colorkey 起始颜色	是
DC1FrameBufferColorKey	DC1:0x2508		
DC0FrameBufferColorKeyHigh	DC0:0x1510	Frame Buffer Colorkey 结束颜色	是
DC1FrameBufferColorKeyHigh	DC1:0x2510		
DC0FrameBufferScaleConfig	DC0:0x1520	Frame Buffer Scalar 配置	是
DC1FrameBufferScaleConfig	DC1:0x2520		
DC0FrameBufferBGColor	DC0:0x1528	Frame Buffer 背景颜色配置	是
DC1FrameBufferBGColor	DC1:0x2528		
DC0FrameBufferUPlanarAddress	DC0:0x1530	FrameBuffer 第二平面起始地址低 32 位	是
DC1FrameBufferUPlanarAddress	DC1:0x2530		
DC0FrameBufferUPlanarhlAddress	DC0:0x1534	FrameBuffer 第二平面起始地址高 8bit	是
DC1FrameBufferUPlanarhlAddress	DC1:0x2534		
DC0FrameBufferVPInarAddress	DC0:0x1538	FrameBuffer 第三平面起始地址低 32 位	是
DC1FrameBufferVPInarAddress	DC1:0x2538		
DC0FrameBufferVPlanarhlAddress	DC0:0x153C	FrameBuffer 第三平面起始地址高 8bit	是
DC1FrameBufferVPlanarhlAddress	DC1:0x253C		
DC0FrameBufferUStride	DC0:0x1800	FrameBuffer 第二平面数据 stride	是
DC1FrameBufferUStride	DC1:0x2800		
DC0FrameBufferVStride	DC0:0x1808	FrameBuffer 第三平面数据 stride	是
DC1FrameBufferVStride	DC1:0x2808		
DC0FrameBufferSize	DC0:0x1810	Frame Buffer 大小	是
DC1FrameBufferSize	DC1:0x2810		
DC0FrameBufferScaleFactorX	DC0:0x1828	Frame Buffer X 轴缩放因子	是
DC1FrameBufferScaleFactorX	DC1:0x2828		
DC0FrameBufferScaleFactorY	DC0:0x1830	Frame Buffer Y 轴缩放因子	是
DC1FrameBufferScaleFactorY	DC1:0x2830		
DC0FrameBufferClearValue	DC0:0x1A18	Frame Buffer 清屏颜色配置	是
DC1FrameBufferClearValue	DC1:0x2A18		
DC0FrameBufferInitialOffset	DC0:0x1A20	Frame Buffer scalar 源偏移配置	是
DC1FrameBufferInitialOffset	DC1:0x2A20		

寄存器名称	偏移	描述	double buffered
Dither 寄存器			
DC0DisplayDitherConfig DC1DisplayDitherConfig	DC0:0x1410 DC1:0x2410	dither 配置	否
DC0DisplayDitherTableLow DC1DisplayDitherTableLow	DC0:0x1420 DC1:0x2420	dither 表低 32 位	否
DC0DisplayDitherTableHigh DC1DisplayDitherTableHigh	DC0:0x1428 DC1:0x2428	dither 表高 32 位	否
panel 寄存器			
DC0PanelConfig DC1PanelConfig	DC0:0x1418 DC1:0x2418	Panel 参数配置	否
DC0HDisplay DC1HDisplay	DC0:0x1430 DC1:0x2430	行 visible 和 total 参数配置	否
DC0HSync DC1HSync	DC0:0x1438 DC1:0x2438	行同步脉冲信号配置	个别位是
DC0VDisplay DC1VDisplay	DC0:0x1440 DC1:0x2440	场 visible 和 total 参数配置	否
DC0VSync DC1VSync	DC0:0x1448 DC1:0x2448	场同步脉冲信号配置	个别位是
DC0DisplayCurrentLocation DC1DisplayCurrentLocation	DC0:0x1450 DC1:0x2450	当前显示的坐标位置	否
Gamma 校正寄存器			
DC0GammaIndex DC1GammaIndex	DC0:0x1458 DC1:0x2458	gamma 表序列	否
DC0GammaData DC1GammaData	DC0:0x1460 DC1:0x2460	gamma 表数据	否
光标 cursor 寄存器			
DC0CursorConfig DC1CursorConfig	DC0:0x1468 DC1:0x2468	光标属性配置	个别位是
DC0CursorAddress DC1CursorAddress	DC0:0x146C DC1:0x246C	光标数据地址配置	是
DC0CursorLocation DC1CursorLocation	DC0:0x1470 DC1:0x2470	光标位置	是
DC0CursorBackground DC1CursorBackground	DC0:0x1474 DC1:0x2474	光标背景颜色	是
DC0CursorForeground DC1CursorForeground	DC0:0x1478 DC1:0x2478	光标近景颜色	是
DP 配置寄存器			
DC0DPConfig DC1DPConfig	DC0:0x1CD0 DC1:0x2CD0	DP 配置	是
中断和门控寄存器			
DC0Intr DC1Intr	DC0:0x147C DC1:0x247C	中断状态标志位	否
DC0IntrEnable	DC0:0x1480	中断使能	否

寄存器名称	偏移	描述	double buffered
DC1IntrEnable	DC1:0x2480		
DC0CursorModuleClockGatingControl DC1CursorModuleClockGatingControl	DC0:0x1484 DC1:0x2484	光标的时钟门控寄存器	否
DC0ModuleClockGatingControl DC1ModuleClockGatingControl	DC0:0x1A28 DC1:0x2A28	时钟门控寄存器，不使用功能可以关掉时钟，以节省功耗	否
滤波寄存器			
DC0Hor iFilterKernelIndex DC1Hor iFilterKernelIndex	DC0:0x1838 DC1:0x2838	水平滤波索引	否
DC0Hor iFilterKernel DC1Hor iFilterKernel	DC0:0x1A00 DC1:0x2A00	水平滤波参数	是
DC0VertiFilterKernelIndex DC1VertiFilterKernelIndex	DC0:0x1A08 DC1:0x2A08	垂直滤波索引	否
DC0VertiFilterKernel DC1VertiFilterKernel	DC0:0x1A10 DC1:0x2A10	垂直滤波参数	是

5.8.4 DC 寄存器说明

5.8.4.1 通用配置寄存器

域	位	读写	复位值	描述
DC0GeneralConfig DC1GeneralConfig				
通用配置				
reserved	31:4	RW	0x0	保留
disable_idle	3	RW	0x0	禁用 idle 信号 0: 禁用; 1: 使能
stall_output_when_underflow	2	RW	0x0	如果使能, FIFO 下溢时, 输出将停止
endian_control	1:0	RW	0x0	端交换控制 0: 不交换 1: 2 字节 2: 4 字节 3: 8 字节
AQHiClockControl				
DC 时钟控制				
reserved	31:19	—	0x800	保留
DC1_CORE_SOFT_RESET	18	RW		DC1core 域复位。1: 复位
DC0_CORE_SOFT_RESET	17	RW		DC0core 域复位。1: 复位
AXI_SOFT_RESET	16	RW		AXI 数据域复位。1: 复位
disable_ram_power_optimization	13	RW		禁用 ram 电源优化 0: 禁用; 1: 使能

域	位	读写	复位值	描述
Ahb_soft_reset	12	RW		DC 软复位。1：复位
disable_debug_register	11	RW		禁用 debug 寄存器 1：使能；0：禁用
disable_ram_clock_gating	10	RW		禁用 ram 的时钟门控
reserved	9:0	RO		保留

5.8.4.2 像素时钟配置寄存器

域	位	读写	复位值	描述
dc0_pixel0_clk_config				
DC0 像素时钟配置				
ack0	31	RO	0x0	pixel0 像素时钟已经配置完成
dc0_req	30	RW	0x0	请求像素时钟 pixel0_clock 变更
dc0_freq_sel	29:0	RW	0x24414	请求像素时钟 pixel0_clock 变更值, 默认值为 1080p 分辨率下的典型像素时钟配置值 148.5MHz
dc1_pixel1_clk_config				
DC1 像素时钟配置				
ack1	31	RO	0x0	pixel1 像素时钟已经配置完成
dc1_req	30	RW	0x0	请求像素时钟 pixel1_clock 变更
dc1_freq_sel	29:0	RW	0x24414	请求像素时钟 pixel1_clock 变更值, 默认值为 1080p 分辨率下的典型像素时钟配置值 148.5MHz

各分辨率对应的像素时钟频率值如下表所示。

表 5-26 分辨率与像素时钟频率对应关系表

分辨率	像素时钟频率 (MHz)
640×480	25.175
800×600	40
1024×768	65
1280×720	74.25
1366×768	85.5
1920×1080	148.5

5.8.4.3 Frame Buffer 寄存器

Frame Buffer 为 DC0/1 的图层基地址。

域	位	读写	复位值	描述
DC0FrameBufferAddress				
DC1FrameBufferAddress				
Frame Buffer 数据起始地址的低 32 位，配寄存器 DC0FrameBufferHiAddress/ DC1FrameBufferHiAddress 8 位地址，构成 Frame Buffer 的 40 位数据起始地址，地址必须 128 字节对齐。				
address	31:0	RW	0x0	Frame Buffer 起始地址的低 32 位

域	位	读写	复位值	描述
DC0FrameBufferHiAddress				
DC1FrameBufferHiAddress				
Frame Buffer 数据起始地址的高 8 位。				
hi_address	7:0	RW	0x0	Frame Buffer 起始地址的高 8 位
DC0FrameBufferStride				
DC1FrameBufferStride				
Frame Buffer stride, 单位字节, 描述 Frame Buffer 原始数据每一行数据的字节数。				
reserved	31:16	R0	0x0	保留
stride	15:0	RW	0x0	每一行数据的字节数
DC0FrameBufferConfig				
DC1FrameBufferConfig				
Frame Buffer 属性参数配置				
format	31:26	RW	0x0	Frame Buffer 的颜色格式。 0x00: X4R4G4B4 0x01: A4R4G4B4 0x02: X1R5G5B5 0x03: A1R5G5B5 0x04: R5G6B5 0x05: X8R8G8B8 0x06: A8R8G8B8 0x07: YUY2 0x08: UYVY 0x0F: YV12 0x11: NV12 0x12: NV16 0x15: NV12_10BIT 0x16: A2R10G10B10 0x17: NV16_10BIT 0x1B: P010
uv_swizzle	25	RW	0x0	保留
swizzle	24:23	RW	0x0	0: ARGB 1: RGBA 2: ABGR 3: BGRA
scale	22	RW	0x0	scale 使能 0: 禁用; 1: 使能
yuv	16:14	RW	0x0	yuv 标准 1: 709; 3: 2020
rot_angle	13:11	RW	0x0	翻转模式 0: 不翻转 1: X 轴翻转 2: Y 轴翻转 3: XY 轴翻转 5: 180 度翻转

域	位	读写	复位值	描述
transparency	10:9	RW	0x0	透明度 0: 非透明模式 1: msk 模式 2: color key 模式
clear	8	RW	0x0	clear 使能。当使能 clear 模式时，framebuffer 的像素值来自 FrameBuffer ClearValue 寄存器，禁用 clear 模式时，framebuffer 的像素值来自内存或者显存，地址由 FrameBufferAddress+ prefix 配置。 0: 禁用；1: 使能
reserved	7	RW	0x0	保留
filp_in_progress	6	R0	0x0	切帧时配合 double buffered 寄存器实现对帧正确性的保护。当 frame buffer 的地址写入时，filp_in_progress 置 1。当把寄存器的值拷贝如工作寄存器生效后，filp_in_progress 将在 VBLANK 开始后清零。这个时候，可以将下一帧的相关参数配置到具有 double buffered 的寄存器中。 0: no; 1: yes
underflow	5	R0	0x0	当显示 FIFO 下溢时，本位置 1。读取一次该寄存器后，本位将会清零。 0: no; 1: yes
reset	4	W0	0x0	复位 DC（与软复位不同）。 1: 复位。将 0 写入该位以重置显示控制器，然后配置其他寄存器，最后将 1 写入该位以启动显示控制器。当显示控制器启动时，它从 VBLANK_START 开始，所有寄存器在 VSYNC_END 跳转到工作集。计数器将重置到 HSYNC 和 VSYNC 的末尾。
valid	3	RW	0x0	valid 字段定义是否可以在下一个 VBLANK 复制一组新的寄存器。这可确保如果该寄存器保存的是最近的一次配置，帧将始终以 valid working 开始，从而减少 SW 等待 VBLANK 信号开始的需要，以确保在下一个 VBLANK 之前加载所有状态。请参考 5.3.9 节的切帧流程。 0: working; 1: pending
gamma	2	RW	0x0	gamma 使能 0: 禁用；1: 使能
reserved	1	RW	0x0	保留
output	0	RW	0x0	当输出使能，像素将会显示。当输出禁用，显示的像素为全黑，此时 panel 的时序是正确的，但是没有任何像素。 0: 禁用；1: 使能

DC0FrameBufferColorKey

DC1FrameBufferColorKey

域	位	读写	复位值	描述
FrameBuffer Color key 范围下限				
alpha	31:24	RW	0x0	alpha 分量
red	23:16	RW	0x0	red 分量
green	15:8	RW	0x0	green 分量
blue	7:0	RW	0x0	blue 分量
DC0FrameBufferColorKeyHigh				
DC1FrameBufferColorKeyHigh				
FrameBuffer Color key 范围上限				
alpha	31:24	RW	0x0	alpha 分量
red	23:16	RW	0x0	red 分量
green	15:8	RW	0x0	green 分量
blue	7:0	RW	0x0	blue 分量
DC0FrameBufferScaleConfig				
DC1FrameBufferScaleConfig				
Frame Buffer scale 参数配置				
reserved	31:8	RO	0x0	保留
horizontal_filter_tap	7:4	RW	0x0	只支持配置成 3
filter_tap	3:0	RW	0x0	只支持配置成 3
DC0FrameBufferBGColor				
DC1FrameBufferBGColor				
Frame Buffer 背景颜色, 当 color key 使能, 且颜色不在 color key 范围内时候, 将会替换成本寄存器配置的颜色				
alpha	31:24	RW	0x0	alpha 分量
red	23:16	RW	0x0	red 分量
green	15:8	RW	0x0	green 分量
blue	7:0	RW	0x0	blue 分量
DC0FrameBufferUPlanarAddress				
DC1FrameBufferUPlanarAddress				
FrameBuffer 图层第二平面数据起始地址的低 32 位				
address	31:0	RW	0x0	起始地址的低 32 位
DC0FrameBufferUPlanarhlAddress				
DC1FrameBufferUPlanarhlAddress				
FrameBuffer 图层第二平面数据起始地址的高 8 位				
msb_address	7:0	RW	0x0	起始地址的高 8 位
DC0FrameBufferVPlanarAddress				
DC1FrameBufferVPlanarAddress				
FrameBuffer 图层第三平面数据起始地址的低 32 位				
address	31:0	RW	0x0	起始地址的低 32 位
DC0FrameBufferVPlanarhlAddress				
DC1FrameBufferVPlanarhlAddress				
FrameBuffer 图层第三平面数据起始地址的高 8 位				
msb_address	7:0	RW	0x0	起始地址的高 8 位
DC0FrameBufferUStride				

域	位	读写	复位值	描述
DC1FrameBufferUStride				
FrameBuffer 图层第二平面 stride				
reserved	31:17	R0	0x0	保留
stride	16:0	RW	0x0	每一行数据的字节数
DC0FrameBufferVStride				
DC1FrameBufferVStride				
FrameBuffer 图层第三平面 stride				
reserved	31:17	R0	0x0	保留
stride	16:0	RW	0x0	每一行数据的字节数
DC0FrameBufferSize				
DC1FrameBufferSize				
Frame Buffer 存在内存中的数据窗口大小, width × height, 单位是像素				
reserved	31:30	R0	0x0	保留
height	29:15	RW	0x0	数据窗口高
width	14:0	RW	0x0	数据窗口宽
DC0FrameBufferScaleFactorX				
DC1FrameBufferScaleFactorX				
Frame Buffer X 轴缩放因子				
reserved	31	RW	0x0	保留
x	30:0	RW	0x0	x 轴缩放因子
DC0FrameBufferScaleFactorY				
DC1FrameBufferScaleFactorY				
Frame Buffer Y 轴缩放因子				
reserved	31	RW	0x0	保留
y	30:0	RW	0x0	y 轴缩放因子
DC0FrameBufferClearValue				
DC1FrameBufferClearValue				
Frame Buffer 图层 clear 值, 该层 clear 使能时有效				
alpha	31:24	RW	0x0	alpha 分量
red	23:16	RW	0x0	red 分量
green	15:8	RW	0x0	green 分量
blue	7:0	RW	0x0	blue 分量
DC0FrameBufferInitialOffset				
DC1FrameBufferInitialOffset				
缩放功能时, 获取帧缓冲区源时使用的初始偏移量。				
y	31:16	RW	0x0	只支持配成 0x8000
x	15:0	RW	0x0	只支持配成 0x8000

5.8.4.4 dither 寄存器

域	位	读写	复位值	描述
DC0DisplayDitherConfig				
DC1DisplayDitherConfig				
Dither 功能属性配置				

域	位	读写	复位值	描述
enable	31	RW	0x0	使能 dither 模式，允许 R8G8B8 模式显示在每个像素位数较少的面板上显示得更好。 1: 使能; 0: 禁用
reserved	30:0	RO	0x0	保留
DC0DisplayDitherTableLow				
DC1DisplayDitherTableLow				
Dither 表 1				
Y1_X3	31:28	RW	0x0	(x, y) = (3, 1) 的阈值
Y1_X2	27:24	RW	0x0	(x, y) = (2, 1) 的阈值
Y1_X1	23:20	RW	0x0	(x, y) = (1, 1) 的阈值
Y1_X0	19:16	RW	0x0	(x, y) = (0, 1) 的阈值
Y0_X3	15:12	RW	0x0	(x, y) = (3, 0) 的阈值
Y0_X2	11:8	RW	0x0	(x, y) = (2, 0) 的阈值
Y0_X1	7:4	RW	0x0	(x, y) = (1, 0) 的阈值
Y0_X0	3:0	RW	0x0	(x, y) = (0, 0) 的阈值
DC0DisplayDitherTableHigh				
DC1DisplayDitherTableHigh				
Dither 表 2				
Y3_X3	31:28	RW	0x0	(x, y) = (3, 3) 的阈值
Y3_X2	27:24	RW	0x0	(x, y) = (2, 3) 的阈值
Y3_X1	23:20	RW	0x0	(x, y) = (1, 3) 的阈值
Y3_X0	19:16	RW	0x0	(x, y) = (0, 3) 的阈值
Y2_X3	15:12	RW	0x0	(x, y) = (3, 2) 的阈值
Y2_X2	11:8	RW	0x0	(x, y) = (2, 2) 的阈值
Y2_X1	7:4	RW	0x0	(x, y) = (1, 2) 的阈值
Y2_X0	3:0	RW	0x0	(x, y) = (0, 2) 的阈值

5.8.4.5 Panel 配置寄存器

Panel 寄存器包括 panel 配置和行场同步信号的时序参数配置。行场同步信号时序参数配置包括 HDisplay、HSync、VDisplay、VSync。

域	位	读写	复位值	描述
DC0PanelConfig				
DC1PanelConfig				
Panel 参数配置				
reserved	31:10	RO	0x0	保留
pixel_clock_out_enable	8	RW	0x1	时钟极性 1: Negative; 0: Positive
reserved	7:6	RO	0x0	保留
data_polarity	5	RW	0x0	数据极性 1: Negative; 0: Positive
data_enable	4	RW	0x0	数据使能 0: 禁用; 1: 使能

域	位	读写	复位值	描述
reserved	3:2	RO	0x0	保留
de_polarity	1	RW	0x0	数据有效时的极性 1: Negative; 0: Positive
de	0	RW	0x0	数据有效使能 0: 禁用; 1: 使能
DC0HDisplay				
DC1HDisplay				
行显示信息配置				
reserved	31	RW	0x0	保留
total	30:16	RW	0x0	行总像素个数
reserved	15	RW	0x0	保留
display_end	14:0	RW	0x0	行显示像素个数
DC0HSync				
DC1HSync				
行同步脉冲信息配置				
polarity	31	RW	0x0	行脉冲同步信号极性 1: Negative; 0: Positive
pulse	30	RW	0x0	行脉冲同步信号控制 0: 禁用; 1: 使能
end	29:15	RW	0x0	行脉冲同步信号结束位置 (像素)
start	14:0	RW	0x0	行脉冲同步信号起始位置 (像素)
DC0VDisplay				
DC1VDisplay				
场同步信号信息配置				
reserved	31	RW	0x0	保留
total	30:16	RW	0x0	场总行个数
reserved	15	RW	0x0	保留
display_end	14:0	RW	0x0	场显示行个数
DC0VSync				
DC1VSync				
场同步脉冲信息配置				
polarity	31	RW	0x0	场脉冲同步信号极性 1: Negative; 0: Positive
pulse	30	RW	0x0	场脉冲同步信号控制 0: 禁用; 1: 使能
end	29:15	RW	0x0	场脉冲同步信号结束位置 (行)
start	14:0	RW	0x0	场脉冲同步信号起始位置 (行)
DC0DisplayCurrentLocation				
DC1DisplayCurrentLocation				
当前位置				
y	31:16	RW	0x0	y 轴当前位置
x	15:0	RW	0x0	x 轴当前位置

5.8.4.6 Gamma 校正寄存器

MMD 支持 gamma 校正, 通过 GammaData 和 GammaIndex 配置 gamma 数据表来实现该功能。

域	位	读写	复位值	描述
DC0GammaIndex				
DC1GammaIndex				
gamma 表序列				
reserved	31:8	RO	0x0	保留
index	7:0	RW	0x0	gamma 表序列号
DC0GammaData				
DC1GammaData				
gamma 数据转换。描述 gamma 的转换值。当这个寄存器被写入时, 数据被存储在 GammaIndex 指定的索引出的 gamma 表中注册。之后如果寄存器被写入, 索引就会递增。				
reserved	31:30	RW	0x0	保留
red	29:20	RW	0x0	red 转换值
green	19:10	RW	0x0	green 转换值
blue	9:0	RW	0x0	blue 转换值

5.8.4.7 cursor 寄存器

域	位	读写	复位值	描述
DC0CursorConfig				
DC1CursorConfig				
光标属性配置				
reserved	31:21	RO	0x0	保留
hot_spot_x	20:16	RW	0x0	光标热点的水平偏移
reserved	15:13	RO	0x0	保留
hot_spot_y	12:8	RW	0x0	光标热点的垂直偏移
reserved	7:5	RO	0x0	保留
display	4	RW	0x0	必须配为 0
reserved	3:1	RO	0x0	保留
format	0	RW	0x0	光标格式 0: DISABLED 1: MASKED 2: A8R8G8B8
DC0CursorAddress				
DC1CursorAddress				
光标数据地址低 32 位, 配合 prefix 构成 40 位地址。				
address	31:0	RW	0x0	光标数据地址低 32 位
DC0CursorLocation				
光标热点的位置				
reserved	31	RO	0x0	保留
y	30:16	RW	0x0	光标热点垂直位置

域	位	读写	复位值	描述
reserved	15	RO	0x0	保留
x	14:0	RW	0x0	光标热点水平位置
DC0CursorBackground DC1CursorBackground 光标背景色, MASK 模式时生效				
reserved	31:30	RO	0x0	保留
red	29:20	RW	0x0	red 值
green	19:10	RW	0x0	green 值
blue	9:0	RW	0x0	blue 值
DC0CursorForeground DC1CursorForeground 光标前景色, MASK 模式时生效				
reserved	31:30	RO	0x0	保留
red	29:20	RW	0x0	red 值
green	19:10	RW	0x0	green 值
blue	9:0	RW	0x0	blue 值

5.8.4.8 DP 配置寄存器

DP 配置寄存器配置 DC 模块与 DP 模块之间的数据接口格式。

域	位	读写	复位值	描述
DC0DPConfig DC1DPConfig Display Port 输出配置				
reserved	31:4	RO	0x0	保留
dp_data_format	2:0	RW	0x0	DP 接口输出格式 0: RGB565 1: RGB666 2: RGB888 3: RGB101010

5.8.4.9 中断和门控寄存器

该寄存器组描述了 MMD 中断相关的信息以及个别模块的时钟门控。

域	位	读写	复位值	描述
DC0Intr DC1Intr DC 中断状态标志位				
reserved	31:3	RO	0x0	保留
DP_INT	2	RW	0x0	DP0/DP1 的中断信号
DC_UNDERFLOW_INT	1	RW	0x0	DC0/DC1 underflow 中断信号, 表明数据供应不上
DCx_INTR	0	RW	0x0	x 取值为 0, 1, 分别表示 DC0/DC1 的帧

域	位	读写	复位值	描述
				完成中断信号
DC0IntrEnable DC1IntrEnable				
DC 中断使能				
reserved	31:2	RO	0x0	保留
DCx_UNDERFLOW_INT_ENABLE	1	RW	0x0	x 取值为 0, 1, 分别表示 DC0/DC1 underflow 中断使能
DCx_INTR_ENABLE	0	RW	0x0	x 取值为 0, 1, 分别表示 DC0/DC1 中断使能 1: 使能; 0: 禁用
DC0CursorModuleClockGatingControl DC1CursorModuleClockGatingControl				
光标的时钟门控				
reserved	31:1	RW	0x0	保留
disable_module_clock_gating_cursor	0	RW	0x1	光标的时钟门控
DC0ModuleClockGatingControl DC1ModuleClockGatingControl				
各模块的时钟门控				
reserved	31:11	RW	0x0	
disable_module_clock_gating_scaler	10	RW	0x1	1: 禁止时钟门控 0: 使能时钟门控
reserved	9:1	RW	0x0	保留
disable_module_clock_gating_video	0	RW	0x1	1: 禁止时钟门控 0: 使能时钟门控

5.8.4.10 Framebuffer 层滤波寄存器

该组寄存器的功能是配置 Frame Buffer 层进行扩大和缩小功能时候的相关滤波器参数，包括垂直滤波参数、水平滤波参数。

域	位	读写	复位值	描述
DC0Hor iFilterKernelIndex DC1Hor iFilterKernelIndex				
Frame Buffer 层水平滤波参数序列号				
reserved	31:8	RO	0x0	保留
index	7:0	RW	0x0	水平滤波参数序列号
DC0Hor iFilterKernel DC1Hor iFilterKernel				
Frame Buffer 层水平滤波参数				
coefficent1	31:16	RW	0x0	coefficent1 值
coefficent0	15:0	RW	0x0	coefficent0 值
DC0Vert iFilterKernelIndex DC1Vert iFilterKernelIndex				

域	位	读写	复位值	描述
Frame Buffer 层垂直滤波参数序列号				
reserved	31:8	RO	0x0	保留
index	7:0	RW	0x0	垂直滤波参数序列号
DC0VertiFilterKernel				
DC1VertiFilterKernel				
Frame Buffer 层垂直滤波参数				
coeffcient1	31:16	RW	0x0	coeffcient1 值
coeffcient0	15:0	RW	0x0	coeffcient0 值

5.8.5 DP 寄存器列表

包括链路配置、链路控制、控制器性能/ID、AUX 接口、主数据流属性、次要通道、面板自刷新、数据包直接写入几大类。

表 5-27 DP 寄存器基地址

名称	基地址
DP0	0x000_3200_4000
DP1	0x000_3200_5000

表 5-28 DP 寄存器列表

寄存器名称	偏移	描述
链路配置		
LINK_BW_SET	0x000	链路速率配置寄存器
LANE_COUNT_SET	0x004	通道数配置寄存器
ENHANCED_FRAME_EN	0x008	Enhanced Framing 模式寄存器
TRAINING_PATTERN_SET	0x00C	训练模式 Training Pattern 寄存器
LINK_QUAL_PATTERN_SET	0x010	连接质量测试寄存器
SCRAMBLING_DISABLE	0x014	加扰使能寄存器
ALTERNATE_SCRAMBLER_RESET	0x01C	ALSR 寄存器
HBR2_COMPLIANCE_SCRAMBLER_RESET	0x020	定义在传输 HBR2 符合性链路质量模式期间传输加扰器重置模式的间隔
DISPLAYPORT_VERSION	0x024	DP 版本寄存器
PHY_POWER_STATE	0x028	电源状态寄存器
LANE_REMAP_CONTROL	0x02C	通道重映射寄存器
CUSTOM_80BIT_PATTERN_0	0x030	80-bit 自定义数据序列寄存器 0
CUSTOM_80BIT_PATTERN_1	0x034	80-bit 自定义数据序列寄存器 1
CUSTOM_80BIT_PATTERN_2	0x038	80-bit 自定义数据序列寄存器 2
链路控制		
TRANSMITTER_OUTPUT_ENABLE	0x080	主链路输出使能寄存器
MAIN_STREAM_ENABLE	0x084	视频数据使能寄存器
SECONDARY_STREAM_ENABLE	0x088	音频数据使能寄存器
SECONDARY_DATA_WINDOW	0x08C	SDP 窗口设置寄存器
SOFT_RESET	0x090	软复位寄存器
INPUT_SOURCE_ENABLE	0x094	视频输入使能寄存器

寄存器名称	偏移	描述
FORCE_SCRAMBLER_RESET	0x0C0	加扰器复位
USER_CONTROL_STATUS	0x0C4	时序控制信号状态（高有效）
USER_DATA_CONTROL_0	0x0C8	用户数据控制寄存器 0
USER_DATA_CONTROL_1	0x0CC	用户数据控制寄存器 1
控制器性能和核 ID		
CORE_CAPABILITIES	0x0F8	控制器性能寄存器
CORE_ID	0x0FC	控制器 ID 寄存器
AUX 接口		
AUX_COMMAND	0x100	AUX 请求配置寄存器
AUX_WRITE_FIFO	0x104	AUX 写数据寄存器
AUX_ADDRESS	0x108	AUX 地址寄存器
AUX_CLOCK_DIVIDER	0x10C	AUX 时钟分频寄存器
AUX_REPLY_TIMEOUT_INTERVAL	0x110	AUX 超时时间寄存器
SINK_HPD_STATE	0x128	HPD 信号状态寄存器
INTERRUPT_STATE	0x130	中断状态寄存器
AUX_REPLY_DATA	0x134	AUX 读数据寄存器
AUX_REPLY_CODE	0x138	AUX 回复代码寄存器
AUX_REPLY_COUNT	0x13C	AUX 回复数寄存器
INTERRUPT_STATUS	0x140	中断寄存器
INTERRUPT_MASK	0x144	中断屏蔽寄存器
AUX_REPLY_DATA_COUNT	0x148	AUX 接收数据数寄存器
AUX_STATUS	0x14C	AUX 传输状态寄存器
AUX_REPLY_CLOCK_WIDTH	0x150	AUX 时钟宽度寄存器
AUX_WAKE_ACK_DETECTED	0x154	标记从连接的 SINK 设备检测到的 PHY_WAKE_ACK 信号
GP_HOST_TIMER	0x158	通用定时器
主数据流属性		
MAIN_STREAM_HTOTAL	0x180	视频行总长度寄存器
MAIN_STREAM_VTOTAL	0x184	视频场总行数寄存器
MAIN_STREAM_POLARITY	0x188	同步脉冲极性寄存器
MAIN_STREAM_HSWIDTH	0x18C	行同步脉冲宽度寄存器
MAIN_STREAM_VSWIDTH	0x190	场同步脉冲宽度寄存器
MAIN_STREAM_HRES	0x194	水平分辨率寄存器
MAIN_STREAM_VRES	0x198	垂直分辨率寄存器
MAIN_STREAM_HSTART	0x19C	Hstart 寄存器
MAIN_STREAM_VSTART	0x1A0	Vstart 寄存器
MAIN_STREAM_MISCO	0x1A4	MISCO 寄存器
MAIN_STREAM_MISC1	0x1A8	MISC1 的寄存器
MAIN_MVID	0x1AC	MVID 寄存器
TRANSFER_UNIT_CONFIG_SRC_0	0x1B0	传输单元配置寄存器
MAIN_NVID	0x1B4	NVID 寄存器
USER_PIXEL_WIDTH	0x1B8	像素输入模式寄存器
USER_DATA_COUNT	0x1BC	UDC 寄存器
MAIN_STREAM_INTERLACED	0x1C0	扫描类型寄存器

寄存器名称	偏移	描述
USER_SYNC_POLARITY	0x1C4	时序控制信号极性
USER_CONTROL	0x1C8	Sparse TU 模式寄存器
次要通道		
SEC_AUDIO_ENABLE	0x300	音频使能寄存器
SEC_INPUT_SELECT	0x304	音频输入选择寄存器
SEC_CHANNEL_COUNT	0x308	音频声道数寄存器
SEC_DIRECT_CLKDIV	0x30C	音频时钟分频寄存器
SEC_INFOFRAME_ENABLE	0x310	InfoFrame 类型选择寄存器
SEC_INFOFRAME_RATE	0x314	InfoFrame 频率寄存器
SEC_MAUD	0x318	音频 MAUD 寄存器
SEC_NAUD	0x31C	音频 NAUD 寄存器
SEC_AUDIO_CLOCK_MODE	0x320	音频时钟模式寄存器
SEC_3D_VSC_DATA	0x324	3D 视频 VSC 寄存器
SEC_AUDIO_FIFO	0x328	音频数据输入寄存器
SEC_AUDIO_FIFO_DEPTH	0x32C	音频数据 FIFO 深度
SEC_AUDIO_FIFO_READY	0x330	音频数据 FIFO 是否准备好
SEC_INFOFRAME_SELECT	0x334	输入 InfoFrame 类型选择寄存器
SEC_INFOFRAME_DATA	0x338	InfoFrame 数据输入寄存器
SEC_TIMESTAMP_INTERVAL	0x33C	ATS 间隔寄存器
SEC_CS_SOURCE_FORMAT	0x340	通道状态寄存器
SEC_CS_CATEGORY_CODE	0x344	Channel Status byte 1 Category code 寄存器
SEC_CS_LENGTH_ORIG_FREQ	0x348	Channel Status byte 4 寄存器
SEC_CS_FREQ_CLOCK_ACCURACY	0x34C	Channel Status byte 3 寄存器
SEC_CS_COPYRIGHT	0x350	Channel Status Copyright 寄存器
SEC_GTC_COUNT_CONFIG	0x354	GTC 配置寄存器
SEC_GTC_COMMAND_EDGE	0x358	GTC TX 寄存器
SEC_AUDIO_CHANNEL_MAP	0x35C	音频通道映射寄存器
数据包直接写入		
SEC_DB_LANE_SELECT	0x3E0	选择要将音频数据写入哪条 lane
SEC_DB_WRITE_INDEX	0x3E4	允许直接设置缓冲区的写入索引
SEC_DB_DATA_COUNT	0x3E8	写入的 sdp 中 valid link symbol 的个数
SEC_DB_DATA	0x3EC	用于将来写操作的直接缓冲区的索引
SEC_DB_READY	0x3F0	在正确写入直接缓冲区后，可以通过设置适当的 SEC_DB_READY 位来启动 SDP 的传输
SEC_DB_BUSY	0x3F4	每个直接缓冲区的状态
SEC_DB_ENABLE	0x3F8	direct packet write 接口使能

5.8.6 DP 寄存器说明

5.8.6.1 链路配置寄存器

域	位	读写	复位值	描述
LINK_BW_SET				

域	位	读写	复位值	描述
设置主链路带宽。寄存器使用与接收设备中相同名称的 DPCD 寄存器支持的值相同的值。这个值在一些 PHY 实现中使用，核心的数字部分不使用。支持 1.62、2.7、5.4、8.1Gbps。				
reserved	31:8	R0	0x0	保留
link_bw_set	7:0	RW	0x0	主链路带宽设置。该值乘 0.27Gb/s 即为当前设置的链接速度(link rate)。 0x06: 1.62Gbps/lane 0x0A: 2.7Gbps/lane 0x14: 5.4Gbps/lane 0x1E: 8.1Gbps/lane
LANE_COUNT_SET				
DP 使用此寄存器设置将用于配置和操作链路的通道数。在训练或视频传输期间，未使用的通道将不会激活				
reserved	31:5	R0	0x0	保留
lane_count_set	4:0	RW	0x0	通道数设置，支持 3 种模式 0x01: 1 lane (lane 0) 0x02: 2 lanes (lane0 lane1) 0x04: 4 lanes (lane0 lane1 lane2 lane3) 该值应与通过 AUX 通道写入接收端 DPCD 的 LANE_COUNT_SET 寄存器的值一致。
ENHANCED_FRAME_EN				
enhanced framing 模式使能，用于 DP1.2a 及更高版本。该值应与通过 AUX 通道写入接收端 DPCD 的 LANE_COUNT_SET 寄存器的值一致。				
reserved	31:1	R0	0x0	保留
enhanced_framing_en	0	R0	0x0	控制器不支持除增强帧以外的模式，因此该寄存器为只读寄存器。
TRAINING_PATTERN_SET				
将该寄存器设置为非零值将输出设置为指定的训练模式。设置后，所有其他主要链接信息（如视频和音频数据）都将被阻止，以支持训练模式。				
reserved	31:3	R0	0x0	保留
training_pattern_set	2:0	RW	0x0	设置链路训练过程的 TPS(Traning pattern sequence) 3'b000: 不发送 TPS 3'b001: TPS1 3'b010: TPS2 (DP1.1a), 1.62, 2.7Gbps 3'b011: TPS3 (DP1.2), 5.4Gbps 3'b100: TPS4 (DP1.3+), 8.1Gbps TSP1 用于时钟恢复, TPS2、TPS3、TPS4 用于信道均等化
LINK_QUAL_PATTERN_SET				
此配置寄存器用于指示控制器启用以下测试模式之一，以便在所选通道上进行链路质量测量。这些测量是基于测试的目的，链路建立和维护时不需要。				
reserved	31:27	R0	0x0	保留
lane_3_pattern_set	26:24	RW	0x0	通道 3 的 pattern 设置，具体见 lane_0_pattern_set

域	位	读写	复位值	描述
reserved	23:19	R0	0x0	保留
lane_2_pattern_set	18:16	RW	0x0	通道 2 的 pattern 设置，具体见 lane_0_pattern_set
reserved	15:11	R0	0x0	保留
lane_1_pattern_set	10:8	RW	0x0	通道 1 的 pattern 设置，具体见 lane_0_pattern_set
reserved	7:3	R0	0x0	保留
lane_0_pattern_set	2:0	RW	0x0	通道 0 的 pattern 设置 3'b000 =不发送 test pattern 3'b001:D10.2 test pattern (unscrambled) 3'b010:Symbol Error Rate measurement pattern 3'b011: PRBS7 3'b100: 80-bit custom pattern 3'b101: HBR2 eye pattern 3'b110: 保留 3'b111: 保留

SCRAMBLING_DISABLE

用于禁用 DisplayPort 发射机的内部加扰功能。该位必须在链路训练过程中设置。

reserved	31:1	R0	0x0	保留
scrambling_disable	0	RW	0x0	加扰功能关闭，置 1 时，控制器不再对输出数据作加扰处理。链路训练时不做加扰处理，其他情况下应置 0。

ALTERNATE_SCRAMBLER_RESET

对于嵌入式 DisplayPort 实现，内核支持使用备用扰码器重置模式。此位只能用于嵌入式应用程序。在逐框显示端口应用程序中设置此位将导致链接失败。

reserved	31:1	R0	0x0	保留
alternate_scrambler_reset	0	RW	0x0	DP 模式必须为 0。需要 Sink 设备的支持

HBR2_COMPLIANCE_SCRAMBLER_RESET

定义在传输 HBR2 符合性链路质量模式期间传输加扰器重置模式的间隔

reserved	31:16	R0	0x0	保留
hbr2_compliance_scrambler_reset	15:0	RW	0x0	设置链路质量测试时 HBR2 eye pattern 的 SR 间隔时间。该值应与 Sink 端 DPCD 的 0024A-0024B 地址的值一致。

DISPLAYPORT_VERSION

指定核心实现支持的 DisplayPort 版本。此值用于处理辅助数据包，不控制任何 DisplayPort 功能的使用。

reserved	31:6	R0	0x0	保留
version_number	5:0	RW	0x0	指定控制器支持的 DP 协议版本号 0x15: DP1.5 0x14: DP1.4a 0x13: DP1.3a 0x12: DP1.2a

域	位	读写	复位值	描述
				0x11: DP1.1a
PHY_POWER_STATE				
控制链路上 ML_PHY_SLEEP 和 ML_PHY_STANDBY 数据模式的传输。写入该寄存器将触发模式的单次传输。对于多个模式，每次写入之间的最小间隔为 100 纳秒，可以启动对该寄存器的重复写入。				
reserved	31:2	RO	0x0	保留
power_state	1:0	RW	0x0	保留
LANE_REMAP_CONTROL				
用于将 DisplayPort 链接的物理通道映射到控制器的内部符号通道。使用这些寄存器位，四个物理通道中的任何一个都可以映射到任何其他内部通道进行处理。每个车道上传输的 10 位数据也可以反转。此操作发生在将通道符号数据发送到 PHY 层之前。				
reserved	31:20	RO	0x0	保留
invert_lane_3	19	RW	0x0	置 1 时，将输出通道 3 的数据反相
invert_lane_2	18	RW	0x0	置 1 时，将输出通道 2 的数据反相
invert_lane_1	17	RW	0x0	置 1 时，将输出通道 1 的数据反相
invert_lane_0	16	RW	0x0	置 1 时，将输出通道 0 的数据反相
reserved	15:9	RO	0x0	保留
remap_enable	8	RW	0x0	1: 使能通道重映射功能 0: 控制器内部的通道与输出通道正常映射，忽略该寄存器 7:0 的值
remap_lane_3	7:6	RW	0x0	指定控制器内部的某一个通道映射至输出通道 3 00: 通道 0 映射到输出通道 3 01: 通道 1 映射到输出通道 3 10: 通道 2 映射到输出通道 3 11: 通道 3 映射到输出通道 3
remap_lane_2	5:4	RW	0x0	指定控制器内部的某一个通道映射至输出通道 2 00: 通道 0 映射到输出通道 2 01: 通道 1 映射到输出通道 2 10: 通道 2 映射到输出通道 2 11: 通道 3 映射到输出通道 2
remap_lane_1	3:2	RW	0x0	指定控制器内部的某一个通道映射至输出通道 1 00: 通道 0 映射到输出通道 1 01: 通道 1 映射到输出通道 1 10: 通道 2 映射到输出通道 1 11: 通道 3 映射到输出通道 1
remap_lane_0	1:0	RW	0x0	指定控制器内部的某一个通道映射至输出通道 0 00: 通道 0 映射到输出通道 0 01: 通道 1 映射到输出通道 0 10: 通道 2 映射到输出通道 0 11: 通道 3 映射到输出通道 0
CUSTOM_80BIT_PATTERN_0				

域	位	读写	复位值	描述
链路质量测试时，80bit 的自定义数据序列（custom pattern）的 bit [31:0]，正在进行链路质量测试时不能改变该寄存器的值				
custom_80bit_pattern_0	31:0	RW	0x0	80bit 的自定义数据序列（custom pattern）的 bit 31:0
CUSTOM_80BIT_PATTERN_1				
链路质量测试时，80bit 的自定义数据序列（custom pattern）的 bit [63:32]，正在进行链路质量测试时不能改变该寄存器的值				
custom_80bit_pattern_1	31:0	RW	0x0	80bit 的自定义数据序列（custom pattern）的 bit 63:32
CUSTOM_80BIT_PATTERN_2				
链路质量测试时，80bit 的自定义数据序列（custom pattern）的 bit [79:64]，正在进行链路质量测试时不能改变该寄存器的值				
custom_80bit_pattern_2	31:0	RW	0x0	80bit 的自定义数据序列（custom pattern）的 bit 79:64

5.8.6.2 链路控制寄存器

域	位	读写	复位值	描述
TRANSMITTER_OUTPUT_ENABLE				
此位用于禁用主链路成帧逻辑的所有输出。当设置为“0”时，发送器核心将只在链路上输出填充符号。禁用时不传输控制符号或有效链路数据。该位防止链路控制器核心干扰 PHY 加电序列。				
reserved	31:1	R0	0x0	保留
enable	0	RW	0x0	控制器主链路输出使能。 0：控制器只发送无效的填充符号（Stuffing Symbol），不会发送任何控制符号（Control Symbol）和有效数据。 1：控制器主链路输出使能。 应在完成控制器和 PHY 的配置后，将寄存器置 1。用于防止控制器的输出干扰 PHY 的上电过程。
MAIN_STREAM_ENABLE				
一旦链路被正确训练并且准备好开始传输用户视频数据，这些比特中的一个或多个可以基于当前虚拟源配置被写入 1。在相关视频输入端口上接收垂直同步脉冲之前，DisplayPort 发射器将输出所选源的“无视频”模式。				
reserved	31:1	R0	0x0	保留
sst_mst_source_0_enable	0	RW	0x0	1：主链路数据有效 0：主链路数据无效
SECONDARY_STREAM_ENABLE				
当主系统准备好开始传输包含音频信息的次要数据包时，该位被写入“1”。当设置为“0”时，DisplayPort 发送端的活动通道将不作为流的一部分发送次要数据。				
reserved	31:1	R0	0x0	保留
secondary_stream_enable	0	RW	0x0	0：禁用辅助数据，并将 VB-ID 中的 AudioMute 标志设置为“1”。 1：使能次要数据传输。

域	位	读写	复位值	描述
SECONDARY_DATA_WINDOW				
指定允许传输次要通道数据包的水平消隐窗口的链路符号时钟宽度。此有效数据窗口的值应小于主链接符号之间的水平消隐周期。				
reserved	31:12	R0	0x0	保留
secondary_data_window	11:0	RW	0x0	有效数据窗口的宽度。该值由以下公式计算： $\text{HBLANK_PERIOD} / \text{LINK_SYMBOL_CLOCK_PERIOD} * 0.9$ HBLANK_PERIOD: 行消隐长度 LINK_SYMBOL_CLOCK_PERIOD: 链路时钟周期长度
SOFT_RESET				
执行特定控制器管理功能的软重置。此复位仅适用于控制部分。可编程寄存器组的状态不受软复位的影响。				
reserved	31:2	R0	0x0	保留
video_soft_reset	1	W0	0x0	1: 复位视频时钟域 vid_clk 部分模块的功能, 异步复位同步释放, 复位信号保持 8 个 vid_clk 周期有效。
link_soft_reset	0	W0	0x0	1: 复位链路时钟域 link_clk 部分模块的功能, 异步复位同步释放, 复位信号保持 8 个 link_clk 周期有效。
INPUT_SOURCE_ENABLE				
在 SST 模式下, 只有该寄存器的 bit 0 有效。如果未启用视频流, 启用虚拟源输入将开始传输 NO_VIDEO 模式。先使能 0x094 寄存器, 开始发送 no video pattern, 等待发送 5 次 no video pattern 后, 再使能 0x084 寄存器。				
reserved	31:1	R0	0x0	保留
virtual_source_0_enable	0	RW	0x0	使能 Source0 输入, 写 1 后, 将开始发送 No Video Pattern。
FORCE_SCRAMBLER_RESET				
用于 debug。设为 1 时, 控制器将会强制将下一次发送的 Blanking Start 符号替换为 Scrambler Reset。(正常情况下, 每 512 个 BS 被替换为 SR)。 建议配合 0x84 一起使用, 使能 0x84 后, 复位一次 SR, 以确保开始发送视频数据时, Source 和 Sink 两端的 LFSR 同步				
reserved	31:1	R0	0x0	保留
force_scrambler_reset	0	R0	0x0	1: 控制器将会强制将下一次发送的 Blanking Start 符号替换为 Scrambler Reset。
USER_CONTROL_STATUS				
提供来自用户数据接口的极性校正控制信号的直接副本。此寄存器可用于触发主机系统中的特定事件。				
reserved	31:4	R0	0x0	保留
user_control_oddeven	3	R0	0x0	当前视频控制信号 vid_oddeven 输入经过极性纠正后的状态
user_control_den	2	R0	0x0	当前视频控制信号 vid_enable 输入经

域	位	读写	复位值	描述
				过极性纠正后的状态
user_control_hsync	1	R0	0x0	当前视频控制信号 vid_hsync 输入经过极性纠正后的状态
user_control_vsync	0	R0	0x0	当前视频控制信号 vid_vsync 输入经过极性纠正后的状态

USER_DATA_CONTROL_0

Source0, Source1 累计延迟控制寄存器 控制内部视频数据 data path, 默认值 0x20042004 不可随意更改。在链路利用率过低或过高时需要手动配置。

reserved	31	R0	0x2004	保留
source1_data_accumulation_delay	30:26	RW		Source1: 从有效数据准备好到开始传输第一个 TU 之间的延迟。
reserved	25:22	R0		保留
source1_fifo_accumulation_depth	21:16	RW		Source1: user FIFO 需要累计足够的数据, 才能开始读并发送到 link 上, 该值决定 FIFO 中累积数据的个数, FIFO 中的数据数量达到该值后 fifo_data_ready 置 1。
reserved	15	R0	0x2004	保留
source0_data_accumulation_delay	14:10	RW		Source0: 从有效数据准备好到开始传输第一个 TU 之间的延迟。
reserved	9:6	R0		保留
source0_fifo_accumulation_depth	5:0	RW		Source0: user FIFO 需要累计足够的数据, 才能开始读并发送到 link 上, 该值决定 FIFO 中累积数据的个数, FIFO 中的数据数量达到该值后 fifo_data_ready 置 1。

USER_DATA_CONTROL_1

Source2, Source3 累计延迟控制寄存器 控制内部视频数据 data path, 默认值 0x20042004 不可随意更改。在链路利用率过低或过高时需要手动配置。

reserved	31	R0	0x2004	保留
source3_data_accumulation_delay	30:26	RW		Source3: 从有效数据准备好到开始传输第一个 TU 之间的延迟。
reserved	25:22	R0		保留
source3_fifo_accumulation_depth	21:16	RW		Source3: user FIFO 需要累计足够的数据, 才能开始读并发送到 link 上, 该值决定 FIFO 中累积数据的个数, FIFO 中的数据数量达到该值后 fifo_data_ready 置 1。
reserved	15	R0	0x2004	保留
source2_data_accumulation_delay	14:10	RW		Source2: 从有效数据准备好到开始传输第一个 TU 之间的延迟。
reserved	9:6	R0		保留
source2_fifo_accumulation_depth	5:0	RW		Source2: user FIFO 需要累计足够的数据, 才能开始读并发送到 link 上, 该值

域	位	读写	复位值	描述
				决定 FIFO 中累积数据的个数, FIFO 中的数据数量达到该值后 fifo_data_ready 置 1。

5.8.6.3 控制器性能/ID 寄存器

域	位	读写	复位值	描述
CORE_CAPABILITIES				
确定控制器中可用的功能。软件可以使用这个寄存器来确定控制器配置。				
reserved	31:19	R0	0x0	保留
mst_source_count	18:16	R0	0x4	MST 模式下最多支持的 source 数, 0: 不支持 MST
reserved	15:12	R0	0x0	保留
fec_present	11	R0	0x1	1: 当前支持 FEC
embedded_present	10	R0	0x1	保留
secondary_present	9	R0	0x1	1: 当前支持音频传输
reserved	8	R0	0x1	保留
reserved	7:3	R0	0x0	保留
lane_count	2:0	R0	0x4	当前支持最大通道数
CORE_ID				
控制器版本, 供软件使用				
core_id	31:16	R0	0xA	控制器识别 ID, 为固定值 0x000A。用于软件配置。
core_rev_level	15:0	R0	0x508	控制器版本号, 固定值 0x0508。用于软件配置。

5.8.6.4 AUX 接口寄存器

域	位	读写	复位值	描述
AUX_COMMAND				
写入时启动指定长度的 AUX 通道命令。该寄存器作为 AUX 通道请求设置过程的一部分最后写入。写入时, 内部状态机将开始向接收器设备发送请求。在写入该寄存器之前, 必须设置 AUX_ADDRESS 和 AUX_WRITE_FIFO (如适用)。				
reserved	31:14	R0	0x0	保留
aux_phy_wake	13	RW	0x0	保留
address_only	12	RW	0x1	写 1 时控制器将发送仅包含地址(无数据)的 AUX 请求
command	11:8	RW	0x0	AUX 命令类型 0x8: AUX Write 0x9: AUX Read 0x0: I2C over AUX Write 0x4: I2C over AUX Write, Middle of Transaction bit set (MOT) 0x1: I2C over AUX Read

域	位	读写	复位值	描述
				0x5: I2C over AUX Read, Middle of Transaction bit set (MOT) 0x2: I2C over AUX Write Status
reserved	7:4	R0	0x0	保留
byte_count	3:0	RW	0x0	当前 AUX 命令要发送的数据字节数, 寄存器的值 0-15 对应 1-16 个字节数据

AUX_WRITE_FIFO

在 AUX 通道上启动本机或 I2C 写入请求之前, 主机系统必须向映射到此地址的 FIFO 提供写入数据。只有支持当前事务所需的字节数必须写入 FIFO。在清除 REQUEST_IN_PROGRESS 位之前, 不得执行对 FIFO 的后续写入。

reserved	31:8	R0	0x0	保留
aux_channel_data	7:0	W0	0x0	存入 AUX_WRITE_FIFO 的数据, 主机在发起 AUX 或者 I2C 写请求前, 将要发送的数据写入寄存器, 写入数据个数不能超过 AUX_COMMAND 中设定的 BYTE_COUNT。

AUX_ADDRESS

对于 AUX 请求, 每个 AUX 请求需要一个 20 位地址; 对于 I2 转 AUX 请求, 每个 AUX 请求需要一个 8 位地址。此寄存器指定当前 AUX 通道的地址命令。这些位用于请求的地址字段而不作修改。

reserved	31:20	R0	0x0	保留
aux_address	19:0	WR	0x0	20-bit 的 AUX 通道命令的起始地址

AUX_CLOCK_DIVIDER

AUX 通道的时钟频率为固定的 1MHz, 该时钟由 APB 时钟分频产生。寄存器的值为 APB 时钟分频值 (只支持整数分频), 有效范围 10-400。例如若 APB 时钟为 75MHz, 该值将设为 75。

reserved	31:9	R0	0x0	保留
divide_value	8:0	RW	0x0	分频值

AUX_REPLY_TIMEOUT_INTERVAL

控制器在发送 AUX request 后, 等待 AUX reply 的时间, 若超过这个时间还未收到 reply, 控制器向主机发送超时中断。

reserved	31:9	R0	0x0	保留
reply_value	8:0	RW	0x190	超时时间, 单位 ms, 默认为 400ms

SINK_HPD_STATE

控制器 HPD 输入端口的 raw state

reserved	31:1	R0	0x0	保留
hpd_raw_state	0	R0	0x0	HPD 输入端口的 raw state

INTERRUPT_STATE

保存中断管理逻辑内部状态的状态位。这些信号用于生成中断, 并且可以用作不实现中断的系统的轮询状态。

reserved	31:7	R0	0x0	保留
reply_error	6	R0	0x0	AUX reply error
reserved t	5	R0	0x0	保留
gp_timer_event	4	R0	0x0	通用定时器中断
reply_timeout	3	R0	0x0	等待 AUX reply 超时
reply_recieved	2	R0	0x0	接收到 AUX reply
hpd_irq	1	R0	0x0	hpd irq 中断

域	位	读写	复位值	描述
hpd_event	0	R0	0x0	HPD 连接或断开事件 中断
AUX_REPLY_DATA				
此只读地址映射到内部 FIFO，该 FIFO 包含在 AUX 通道应答期间接收的多达 16 字节的信息。从字节 0 开始从 FIFO 读取应答数据。FIFO 中的有效字节数与应答数据计数寄存器指示的接收字节数相对应。				
reserved	31:8	R0	0x0	保留
aux_reply_data	7:0	R0	0x0	AUX reply 期间接收到的数据，每次读取，内部 FIFO 读指针加一。有效数据个数与 REPLY_DATA_COUNT 寄存器中的设置有关。
AUX_REPLY_CODE				
最近一次发起的 AUX request 时接收到的 reply code。反映 AUX 传输的状态				
reserved	31:4	R0	0x0	保留
aux_reply_code	3:0	R0	0x0	0x0:Native AUX ACK 0x1:Native AUX NACK 0x2:Native AUX Defer 0x0:I2C over AUX ACK 0x4:I2C over AUX NACK 0x8:I2C over AUX Defer
AUX_REPLY_COUNT				
统计目前收到的 AUX reply 个数（不包括 reply code）。写 1 清零				
reserved	31:8	R0	0x0	保留
aux_reply_count	7:0	R0	0x0	收到的 AUX reply 个数。
INTERRUPT_STATUS				
控制器中断状态寄存器，包括中断原因。可导致中断的特定事件和相关的状态位如下所示。从该寄存器读取的数据将清除所有值。				
reserved	31:7	R0	0x0	保留
reply_error	6	RW	0x0	AUX reply error
reserved	5	RW	0x0	保留
gp_timer_irq	4	RW	0x0	通用定时器发起中断
reply_timeout	3	RW	0x0	因等待 AUX reply 超时发起中断
reply_recieved	2	RW	0x0	接收到 AUX reply 发起中断
hdp_irq	1	RW	0x0	hpd irq 中断
hpd_event	0	RW	0x0	HPD 连接或断开事件中断
INTERRUPT_MASK				
控制器的每个中断源可以单独屏蔽。当此寄存器中的相应位设置为“1”时，不会为事件生成中断。上电后，所有中断源都被屏蔽。				
reserved	31:7	R0	0x0	保留
reply_error_mask	6	RW	0x1	AUX reply error
reserved	5	RW	0x1	保留
gp_timer_irq_mask	4	RW	0x1	通用定时器发起中断
reply_timeout_mask	3	RW	0x1	因等待 AUX reply 超时发起中断
reply_recieved_mask	2	RW	0x1	接收到 AUX reply 发起中断
hdp_irq_mask	1	RW	0x1	hpd irq 中断
hpd_event_mask	0	RW	0x1	HPD 连接或断开事件 中断
AUX_REPLY_DATA_COUNT				

域	位	读写	复位值	描述
最近一次 AUX reply 传输从 sink 端接收到的数据个数。控制器发起 AUX request 会清空该寄存器				
reserved	31:5	R0	0x0	保留
data_count	4:0	R0	0x0	从 AUX 通道收到的响应数据数量。
AUX_STATUS				
此寄存器包含内部 AUX 通道控制器的状态。监视请求和应答事务的进度，并检查应答事务是否有错误。这些位总是有效的。				
reserved	31:4	R0	0x0	保留
reply_error	3	R0	0x0	1: 最近一次的 AUX reply 传输过程出现错误。Reply_error 指 reply 过程中，等待 framing start code 超时。
request_in_progress	2	R0	0x0	1: 控制器正在发送 AUX request; 0: AUX request 状态机空闲。
reply_in_progress	1	R0	0x0	1: 控制器正在接受 AUX reply。
reply_received	0	R0	0x0	0: 控制器正在发送 AUX request, 发起 request 将此位清零。 1: 控制器已收到完整有效的 AUX reply。
AUX_REPLY_CLOCK_WIDTH				
AUX reply 通过同步过程恢复的 AUX reply 时钟宽度 (APB 时钟的个数)。该寄存器仅在一个 AUX reply 传输完成后有效				
reserved	31:10	R0	0x0	保留
aux_reply_clock_width	9:0	R0	0x0	AUX reply 时钟宽度
AUX_WAKE_ACK_DETECTED				
标记从连接的接收设备检测 AUX 物理唤醒信号。该位在任何 AUX 事务开始时清除，并在检测到 AUX 物理唤醒信号时设置。				
reserved	31:1	R0	0x0	保留
aux_phy_wake_ack	0	R0	0x0	1: AUX reply 接收过程中接收到 PHY_WAKE_ACK, 表示 AUX PHY WAKE 请求完成。
GP_HOST_TIMER				
控制器使用的通用 20 位定时器。DisplayPort Tx core 不将此计时器用于任何内部功能。计时器的分辨率为 1 usec。				
enable	31	RW	0x0	写 1 使能通用定时器
reload	30	RW	0x0	写 1 定时器计到 0 时自动重置，写 0 定时器只运行一次
interrupt	29	RW	0x0	写 1 定时器计到 0 时产生中断
reserved	28:20	R0	0x0	保留
timer_value	19:0	RW	0x0	写操作设置定时器的计数值，读操作返回定时器当前的值

5.8.6.5 主数据流属性寄存器

主数据流属性寄存器描述的是视频流相关的属性配置以及行场同步信号时序参数配置。

域	位	读写	复位值	描述
MAIN_STREAM_HTOTAL				

域	位	读写	复位值	描述
指定主流视频信号在水平帧周期内的时钟总数。此值作为主流属性 Htotal 发送。				
reserved	31:16	R0	0x0	保留
Htotal	15:0	RW	0x0	Htotal
MAIN_STREAM_VTOTAL				
提供主流视频帧中垂直同步脉冲之间的总行数。此值作为主流属性 Vtotal 提供。				
reserved	31:16	R0	0x0	保留
vtotal	15:0	RW	0x0	Vtotal
MAIN_STREAM_POLARITY				
行场同步信号脉冲极性				
reserved	31:2	R0	0x0	保留
vsync_polarity	1	R0	0x0	场同步信号极性 0: 高有效; 1: 低有效
hsync_polarity	0	R0	0x0	行同步信号极性 0: 高有效; 1: 低有效
MAIN_STREAM_HSWIDTH				
行同步信号脉宽, 单位: 像素时钟周期				
reserved	31:16	R0	0x0	保留
hs_width	15:0	RW	0x0	行同步信号脉宽
MAIN_STREAM_VSWIDTH				
场同步信号脉宽, 单位: 像素时钟周期				
reserved	31:16	R0	0x0	保留
vs_width	15:0	RW	0x0	场同步信号脉宽
MAIN_STREAM_HRES				
视频流一行的有效像素个数				
reserved	31:16	R0	0x0	保留
hres	15:0	RW	0x0	视频流一行的有效像素个数
MAIN_STREAM_VRES				
视频流一场的有效行数				
reserved	31:16	R0	0x0	保留
vres	15:0	RW	0x0	视频流一场的有效行数
MAIN_STREAM_HSTART				
视频流每行的有效开始像素位置				
reserved	31:16	R0	0x0	保留
hstart	15:0	RW	0x0	视频流每行的有效开始像素位置
MAIN_STREAM_VSTART				
视频流每场的有效开始行位置				
reserved	31:16	R0	0x0	保留
vstart	15:0	RW	0x0	视频流每场的有效开始行位置
MAIN_STREAM_MISCO				
此 8 位值包含有关视频流时钟和颜色的信息陈述。这些位从 DisplayPort 规范 MISCO 寄存器定义映射。				
reserved	31:8	R0	0x0	保留
bit_depth	7:5	RW	0x0	每个颜色的位数。 000: 6 位

域	位	读写	复位值	描述
				001: 8 位 010: 10 位 011: 12 位 100: 16 位 101: 保留 110: 保留 111: 保留
yccr_colorimetry	4	RW	0x0	主链路视频色度。 0: ITU-R BT601-5; 1: ITU-R BT709-5
dynamic_range	3	RW	0x0	颜色范围。 0: VESA range; 1: CEA range
component_format	2:1	RW	0x0	颜色格式 00: RGB 01: YCbCr 4:2:2 10: YCbCr 4:4:4 11: 保留
synchronous_clock	0	RW	0x0	locking mode for the user data 0: asynchronous clock; 1: synchronous clock

MAIN_STREAM_MISC1

这些位表示 DisplayPort 规范中定义的主数据流属性字段 MISC1。这些比特提供隔行扫描和立体视频信息。

reserved	31:8	R0	0x0	保留
y_only	7	RW	0x0	1: Y-only Color format
zero	6:3	RW	0x0	必须配置为 0
stereo_video_attr	2:1	RW	0x0	00: 立体视频 01: 右眼 10: 保留 11: 左眼
interlaced_total_even	0	RW	0x0	0: 每个隔行扫描帧的行数是奇数 1: 每个隔行扫描帧的行数是偶数

MAIN_MVID

M 值，异步时钟模式下生效

reserved	31:24	R0	0x0	保留
m_vid	23:0	RW	0x0	像素时钟 (MHz) × 100

TRANSFER_UNIT_CONFIG_SRC_0

传输单元是一个 DisplayPort 包，表示有效的数据符号和填充符号。该寄存器值设置发射机帧逻辑中传输单元的大小。该寄存器只支持 4 的倍数值。当配置为稀疏 TU 模式时，只有该寄存器的 TRANSFER_UNIT_SIZE 字段有效。忽略所有其他字段。

reserved	31:28	R0	0x0	保留
frac_symbols_per_tu	27:24	RW	0x0	一个 TU 中 valid symbol 的个数的小数部分（单位为 1/16th）。
symbols_per_tu	23:16	RW	0x0	一个 TU 中 valid symbol 的个数的整数部分，可被设置为 64 及以下的整数值，关

域	位	读写	复位值	描述
				于 valid symbol 个数的计算见 DP 标准的 2.2.1.4.1。
reserved	15:7	RW	0x0	保留
transfer_unit_size	6:0	RW	0x0	TU 的大小(valid symbol 和 stuff symbol 的总数), 必须被设置在 32 和 64 之间。
MAIN_NVID				
N 值。基于链路速率设置用于主流属性的第二时钟值。当与 M_VID 值一起使用时, 该值允许接收器设备恢复用户数据像素时钟的频率。异步时钟模式下生效。				
reserved	31:24	RW	0x0	保留
n_vid	23:0	RW	0x0	1. 62Gbps 时, 配成 16200; 2. 7Gbps 时, 配成 27000; 5. 4Gbps 时, 配成 54000; 8. 1Gbps 时, 配成 81000;
USER_PIXEL_WIDTH				
控制器的用户数据接口接受每个时钟周期一个、两个或四个像素。此寄存器选择用户数据输入端口的宽度, 应在启用主链路视频之前进行设置。复位时, 该寄存器默认为 1。				
reserved	31:3	RW	0x0	保留
user_pixel_count	2:0	RW	0x1	1、2、4 分别对应每个视频时钟周期输入 1、2、4 个像素数据
USER_DATA_COUNT				
在发送消隐开始符号之前, 确定要从用户 FIFO 读取的发送器帧逻辑的总数据计数。换句话说, 这个值是一行活动数据中 symbol 的总数。计算值应向上舍入。				
reserved	31:18	RW	0x0	保留
user_data_count	17:0	RW	0x0	$\text{SYMBOL_COUNT} = ((\text{HRES} * \text{位 per pixel}) + 7) / 8$ $\text{UDC} = (\text{SYMBOL_COUNT} + \text{lane_count} - 1) / \text{lane_count}$
MAIN_STREAM_INTERLACED				
视频扫描类型				
reserved	31:1	RW	0x0	保留
main_stream_interlaced	0	RW	0x0	1: 隔行扫描; 0: 逐行扫描 与 VBI 和 MSA 有关
USER_SYNC_POLARITY				
指示视频源同步信号的极性。				
reserved	31:4	RW	0x0	保留
user_oddeven_polarity	3	RW	0x0	odd/even 信号的极性 1: 高有效; 0: 低有效
user_data_enable_polarity	2	RW	0x0	user data enable 信号的极性 1: 高有效; 0: 低有效
user_vsync_polarity	1	RW	0x0	vsync 信号极性 1: 高有效; 0: 低有效
user_hsync_polarity	0	RW	0x0	hsync 信号极性 1: 高有效; 0: 低有效
USER_CONTROL				

域	位	读写	复位值	描述
控制控制器的每个活动通道中用户数据 FIFO 的行为。				
reserved	31:2	RW	0x0	保留
user_secondary_immediate	1	RW	0x0	设置为 1 时, 若此时 0x088 寄存器已被设为 1, Secondary channel 立即变为有效, 不需要等待下一个场同步信号。
user_sparse_mode_enable	0	RW	0x0	设置为 1 时, TU 中 valid symbol 数量可变, 不再是固定值, TRANSFER_UNIT_CONFIG 寄存器中 SYMBOL_PER_TU 和 FRAC_SYMBOLS_PER_TU 的设置将被忽略。

5.8.6.6 次要通道寄存器

次要通道寄存器主要描述音频通道的属性及工作方式。

域	位	读写	复位值	描述
SEC_AUDIO_ENABLE				
次要通道使能				
reserved	31:2	RW	0x0	保留
sec_audio_mute	1	RW	0x0	设置为 1 时, VBID 的 audio mute (静音) 位被设为 1。 SEC_AUDIO_MUTE 与 SEC_AUDIO_ENABLE 不能设为相同值
sec_audio_enable	0	RW	0x0	1: 音频使能; 0: 音频禁用
SEC_INPUT_SELECT				
选择音频输入来源, 目前只支持 I2S 和 Direct Sample FIFO 两种模式				
reserved	31:2	RW	0x0	保留
audio_source_select	1:0	RW	0x0	00: I2S 01: Direct Sample FIFO 10: 保留 11: 保留
SEC_CHANNEL_COUNT				
音频通道数量选择, 最多支持 8 声道				
reserved	31:3	RW	0x0	保留
audio_channel_num	2:0	RW	0x0	000: Audio mute set in the VBID field 010: Two channel audio 011: Two channel audio with LFE 110: 5.1 channel surround sound 111: 7.1 channel surround sound
SEC_DIRECT_CLKDIV				
用于音频源为 Direct write FIFO mode, APB 时钟分频至音频采样时钟的分频值。MMD 中 APB 时钟为 48MHz, 分频后的时钟单位为 KHz				
reserved	31:16	RW	0x0	保留
clk_div	15:0	RW	0x0	clk_div = (APB Frequency / Audio Sample Rate)

域	位	读写	复位值	描述
SEC_INFOFRAME_ENABLE				
此寄存器中的每一位将允许传输五个受支持的通用辅助数据包中的一个				
reserved	31:5	RW	0x0	保留
NTSC_VBI_InfoFrame_Product	4	RW	0x0	NTSC VBI InfoFrame
Audio_InfoFrame_Source	3	RW	0x0	Audio InfoFrame
Description_InfoFrame	2	RW	0x0	Source Product Description InfoFrame
AUX_Video_Information_(AVI)_InfoFrame	1	RW	0x0	AUX Video Information (AVI) InfoFrame
vendor_specific_infoFrame	0	RW	0x0	Vendor Specific InfoFrame
SEC_INFOFRAME_RATE				
选择每个通用辅助数据分组被发送到接收器设备的频率。包可以以立即模式发送一次，或者以帧模式每帧发送一次。将对应于通用 SDP 的位设置为 1，以启用每帧的连续传输。设置为 0 以在启用位从 0 转换为 1 时传输通用 SDP 一次。连续传输被定义为在垂直消隐间隔期间每帧发送一个包。				
reserved	31:5	RW	0x0	保留
NTSC_VBI_InfoFrame_Product	4	RW	0x0	NTSC VBI InfoFrame
Audio_InfoFrame_Source	3	RW	0x0	Audio InfoFrame
Description_InfoFrame	2	RW	0x0	Source Product Description InfoFrame
AUX_Video_Information_InfoFrame	1	RW	0x0	AUX Video Information (AVI) InfoFrame
vendor_specific_infoFrame	0	RW	0x0	Vendor Specific InfoFrame
SEC_MAUD				
音频数据 M 值，音频时钟与链路时钟同步时有效				
reserved	31:24	RW	0x0	保留
m_aud	23:0	RW	0x0	M 值
SEC_NAUD				
音频数据 N 值，音频时钟与链路时钟同步时有效				
reserved	31:24	RW	0x0	保留
n_aud	23:0	RW	0x0	N 值
SEC_AUDIO_CLOCK_MODE				
音频数据时钟模式，指音频数据的 sample clock 和传输时钟 Link clock				
reserved	31:1	RW	0x0	保留
sec_audio_clock_mode	0	RW	0x0	0: 异步时钟模式 1: 同步时钟模式
SEC_3D_VSC_DATA				
用于在 3D 立体应用中承载附加视频流配置信息的数据字节。DisplayPort 规范的第 2.2.5.6.2 节规定了该字节的内容。				
reserved	31:8	RW	0x0	保留
sims	7:4	RW	0x0	Stereo Interface Method-Specific Parameter
sime	3:0	RW	0x0	Stereo Interface Method Code

域	位	读写	复位值	描述
SEC_AUDIO_FIFO				
音频数据输入为 Direct write 时的 FIFO				
sample_fifo_entry	31:0	RW	0x0	用于音频数据输入为 Direct write FIFO mode 时。写入顺序为 Channel 1, Sample0 最先写入, 随后通道数先增加至支持的通道数后, Sample 数再增加, 以 2 声道为例: 顺序依次为 C1 S0, C2 S0, C1 S1, C2 S1, C1 S2, C2 S2。 每 8 个 sample 为一个 secondary audio channel packet 每个 sample 高 8 位为 control field, 低 24 位为音频数据 每次写入后, 内部 sample FIFO 指针加一, 读操作将返回上一次写入寄存器的值。
SEC_AUDIO_FIFO_DEPTH				
包含要存储在音频采样 FIFO 中的最大采样数。				
override	31	RW	0x0	direct write FIFO override
reserved	30:10	RW	0x0	保留
sec_audio_fifo_depth	9:0	RW	0x0	用于音频数据输入为 Direct write FIFO mode 时。内部 sample FIFO 的深度, 写入 FIFO 的个数达到设定的深度时, FIFO 的 ready flag 无效。
SEC_AUDIO_FIFO_READY				
direct audio sample FIFO 中数据个数少于 SEC_AUDIO_FIFO_DEPTH 寄存器中设定的深度, 可以写入 sample 数据。				
reserved	31:1	RW	0x0	保留
sec_audio_fifo_ready	0	RW	0x0	1: direct audio sample FIFO 中数据个数少于 SEC_AUDIO_FIFO_DEPTH 寄存器中设定的深度, 可以写入 sample 数据。
SEC_INFOFRAME_SELECT				
选择要写入内部 SRAM 的 InfoFrame SDP 的类型。不同类型的 InfoFrame 在 SRAM 中有对应的地址范围。				
reserved	31:3	RW	0x0	保留
sec_info_select	2:0	RW	0x0	0: Vendor Specific 1: AUX Video Information 2: Source Product Description 3: Audio Description 4: NTSC VBI
SEC_INFOFRAME_DATA				
写入的 InfoFrame SDP 数据, 根据 SEC_INFOFRAME_SELECT 寄存器设置的值, 将数据写入 InfoFrame SRAM 对应的地址。读寄存器将返回上一次写入的值。				
reserved	31:8	RW	0x0	保留
sec_info_data	7:0	RW	0x0	写入的 InfoFrame SDP 数据
SEC_TIMESTAMP_INTERVAL				
发送每个 audio timestamp packet 间等待的时间间隔 (us)。若设为 0, 每个场消隐期间控制器将				

域	位	读写	复位值	描述
只发送一次 audio timestamp packet				
reserved	31:8	RW	0x0	保留
set_timestamp_interval	7:0	RW	0x0	发送每个 audio timestamp packet 间等待的时间间隔, 单位 us
SEC_CS_SOURCE_FORMAT				
此寄存器用于定义随每个音频帧发送的信道状态信息中的字段。适当的位会自动插入到音频帧的通道状态字段中。				
reserved	31:8	RW	0x0	保留
source_num	7:4	RW	0x0	具有唯一源代码标识符的四位代码
linear_pcm	3	RW	0x0	0: LPCM samples; 1: encoded samples
pcm_audio_format	2:0	RW	0x0	与通道状态的位 5-3 相对应的三位代码
SEC_CS_CATEGORY_CODE				
通道状态类别代码				
reserved	31:8	RW	0x0	保留
sec_cs_category_code	7:0	RW	0x0	8 位分类码, 表示 IEC60958-3 相关附件中定义的数字音频信号的设备类型。
SEC_CS_LENGTH_ORIG_FREQ				
通道状态字长和原始采样频率。这些值被插入到音频流次要数据包的信道状态位内的适当字段中。				
reserved	31:8	RW	0x0	保留
sample_word_length	7:4	RW	0x0	IEC 60958-3 规范中信道状态字段位 32-35 中规定的每个采样字的长度。
orig_sampling_freq	3:0	RW	0x0	携带音频信号的原始采样频率。这些位反映 IEC60958-3 信道状态字段中位 39-36 的值。
SEC_CS_FREQ_CLOCK_ACCURACY				
通道状态类别代码				
reserved	31:8	RW	0x0	保留
sampling_freq	7:4	RW	0x0	当前音频信号的采样频率, 其值反映位 24-27。
clock_accuracy	3:0	RW	0x0	当前采样频率的时钟精度代码。此代码相当于信道状态字段的位 28-29。
SEC_CS_COPYRIGHT				
音频流通道状态位的版权代码				
reserved	31:3	RW	0x0	保留
cgms-a	2:1	RW	0x0	复制当前音频流的生成管理系统信息。
copyright	0	RW	0x0	版权声明的 CP 位。不支持交替此位的值以指示未知状态的模式。
SEC_GTC_COUNT_CONFIG				
配置用于管理全局时间码函数的内部累加器。累加器由 8 位整数部分和 16 位小数部分组成。在每个主机时钟周期, 分数和整数部分被添加到各自的计数器。当分数计数器超过 16 位范围时, 整数部分会增加一个额外的计数。				
reserved	31:24	RW	0x0	保留
gtc_count_int	23:16	RW	0x0	GTC 计数器每次累加值的整数部分
gtc_count_frac	15:0	RW	0x0	GTC 计数器每次累加值的小数部分

域	位	读写	复位值	描述
SEC_GTC_COMMAND_EDGE				
返回最近一次 AUX 命令边沿(完成 同步后, CMD 的第一个上升沿)时的 GTC 计数值, 这个值和 GTC update 事件时发送给 sink 端的 GTC 值相同				
gtc_count_value	31:0	RW	0x0	GTC 计数值
SEC_AUDIO_CHANNEL_MAP				
音频输入通道映射				
channel_8_map	31:28	RW	0x0	将 channel 8 输入的音频数据与选择的 1-8 任一 channel 映射。 1: 映射到输出 channel 1 2: 映射到输出 channel 2 依此类推
channel_7_map	27:24	RW	0x0	同 channel_8_map
channel_6_map	23:20	RW	0x0	同 channel_8_map
channel_5_map	19:16	RW	0x0	同 channel_8_map
channel_4_map	15:12	RW	0x0	同 channel_8_map
channel_3_map	11:8	RW	0x0	同 channel_8_map
channel_2_map	7:4	RW	0x0	同 channel_8_map
channel_1_map	3:0	RW	0x0	同 channel_8_map

5.8.6.7 数据包直接写入寄存器

域	位	读写	复位值	描述
SEC_DB_LANE_SELECT				
选择要将音频数据写入哪条 lane, 每条 lane 的音频数据都存入一个深度 16, 宽度 32bit 的 buffer。				
reserved	31:2	RW	0x0	保留
lane_select	1:0	RW	0x0	00: lane0 01: lane1 10: lane2 11: lane3
SEC_DB_WRITE_INDEX				
允许直接设置缓冲区的写入索引。索引值 0-15 将允许写入第一个缓冲区, 而索引值 16-31 将允许写入第二个缓冲区。一个写索引值用于所有四个通道中的直接缓冲区。				
reserved	31:5	RW	0x0	保留
buffer_select	4	RW	0x0	buffer0 或者 buffer1
index	3:0	RW	0x0	用于将来写操作的直接缓冲区的索引。每次写入操作后, 索引值将自动递增。
SEC_DB_DATA_COUNT				
写入的 sdp 中 valid link symbol 的个数, 应在设置 SEC_DB_READY 前(即开始从 buffer 中读取数据前)设置合适的 data count。Data count 范围 1-16。				
reserved	31:6	RW	0x0	保留
count	5:0	RW	0x0	写入的 sdp 中 valid link symbol 的个数
SEC_DB_DATA				
用于将来写操作的直接缓冲区的索引。每次写入后, 索引值将自动递增操作。链接符号数据对写入辅助数据包直接缓冲区。写入时, 此寄存器中的 16 位值将使用 SEC_DB_WRITE 索引值设置的索引存				

域	位	读写	复位值	描述
储在直接缓冲区中。每次写入操作后，索引将自动递增。				
sec_db_dat	31:0	RW	0x0	
SEC_DB_READY				
在正确写入直接缓冲区后，可以通过设置适当的 SEC_DB_READY 位来启动 SDP 的传输。当设置为“1”时，内部直接缓冲逻辑将在主链路上为每个启用的通道传输辅助数据包。在设置这些标志之前，必须正确设置 SEC_DB_DATA_COUNT 寄存器的值。这些寄存器在读取时总是返回“0”。				
reserved	31:2	RO	0x0	保留
buffer_1_ready	1	RW	0x0	写 1 开始传输 buffer1 中的数据，从 index32 开始传输
buffer_0_ready	0	RW	0x0	写 1 开始传输 buffer0 中的数据，从 index0 开始传输
SEC_DB_BUSY				
这些只读状态位表示每个直接缓冲区的状态。当设置为“1”时，当前直接缓冲区正在等待通过主链路传输。当辅助数据包完成传输时，每个位将清除为“0”。在写入 SEC_DB_READY 寄存器后，该位被设置为“1”。				
reserved	31:2	RO	0x0	保留
buffer_1_busy	1	RW	0x0	BUFFER1 的传输状态，1 表示 BUFFER1 的数据正在 link 上传输
buffer_0_busy	0	RW	0x0	BUFFER0 的传输状态，1 表示 BUFFER0 的数据正在 link 上传输
SEC_DB_ENABLE				
direct packet write 接口使能				
reserved	31:1	RO	0x0	保留
enable	0	RW	0x0	1: 使能 direct packet write 接口，在 direct packet mode 下忽略所有内部产生的 Secondary data packet

5.8.7 DP-PHY 寄存器列表

表 5-29 DP-PHY 寄存器基地址

名称	基地址
DP_PHY0	0x000_3230_0000
DP_PHY1	0x000_3240_0000

表 5-30 DP-PHY 寄存器列表

寄存器名称	偏移	描述
TX_TXCC_CTRL	0x10100	TX 系数控制器控制寄存器
TX_TXCC_CPOST_MULT_00	0x10130	post emphasis 乘数值寄存器
TX_TXCC_MGNFS_MULT_000	0x10140	Margin 全摆幅乘数值寄存器
DRV_DIAG_TX_DRV	0x10318	TX 驱动器诊断寄存器
TX_DIAG_ACYA	0x1079C	发送器模拟配置寄存器

5.8.8 DP-PHY 寄存器说明

5.8.8.1 TX_TXCC_CTRL (0x10100)

位	读写	复位值	描述
15:14	RW	0x0	保留
13:12	RW	0x2	Margin 乘法器舍入控制：此字段控制保证 Margin 乘法器的舍入功能。 <ul style="list-style-type: none"> 2'b00：向最近的取整 2'b01：向下 2'b10：向上
11:10	RW	0x2	LF 乘法器舍入控制：此字段控制保证 LF 乘法器的舍入功能。 <ul style="list-style-type: none"> 2'b00：向最近的取整 2'b01：向下 2'b10：向上
9:8	RW	0x2	post-emphasis 乘法器舍入控制：此字段控制保证 post-emphasis 乘法器的舍入功能。 <ul style="list-style-type: none"> 2'b00：向最近的取整 2'b01：向下 2'b10：向上
7:6	RW	0x2	pre-emphasis 乘法器舍入控制：此字段控制保证 pre-emphasis 乘法器的舍入功能。 <ul style="list-style-type: none"> 2'b00：向最近的取整 2'b01：向下 2'b10：向上
5:4	RW	0x2	coefficient 乘法器舍入控制：此字段控制保证 coefficient 乘法器的舍入功能。 <ul style="list-style-type: none"> 2'b00：向最近的取整 2'b01：向下 2'b10：向上
3	RW	0x0	de-emphasis 控制标准模式 3 值：当 xcvr_standard_mode 设置为 2'b11 时，该位控制 de-emphasis 模式。 <ul style="list-style-type: none"> 1'b0：使用 tx_deemphasis 信号的 2 个最低有效位控制。 1'b1：使用 tx_deemphasis 信号所有 18 位的系数值控制。
2	RW	0x1	de-emphasis 控制标准模式 2 值：当 xcvr_standard_mode 设置为 2'b10 时，该位控制 de-emphasis 模式。 <ul style="list-style-type: none"> 1'b0：使用 tx_deemphasis 信号的 2 个最低有效位控制。 1'b1：使用 tx_deemphasis 信号所有 18 位的系数值控制。
1	RW	0x0	de-emphasis 控制标准模式 1 值：当 xcvr_standard_mode 设置为 2'b01 时，该位控制 de-emphasis 模式。 <ul style="list-style-type: none"> 1'b0：使用 tx_deemphasis 信号的 2 个最低有效位控制。 1'b1：使用 tx_deemphasis 信号所有 18 位的系数值控制。
0	RW	0x0	de-emphasis 控制标准模式 0 值：当 xcvr_standard_mode 设置为 2'b00 时，该位控制 de-emphasis 模式。 <ul style="list-style-type: none"> 1'b0：使用 tx_deemphasis 信号的 2 个最低有效位控制。

位	读写	复位值	描述
			•1'b1: 使用 tx_deemphasis 信号所有 18 位的系数值控制。

5.8.8.2 TX_TXCC_CPOST_MULT_00 (0x10130)

位	读写	复位值	描述
15:8	RW	0x0	保留
7:0	RW	0x10	指定用于 post emphasis 的乘数值。以下描述了该字段中每个位对应的乘数值。 •Bit 7 : $128/128 = 1.000000$ •Bit 6 : $64/128 = 0.500000$ •Bit 5 : $32/128 = 0.250000$ •Bit 4 : $16/128 = 0.125000$ •Bit 3 : $8/128 = 0.062500$ •Bit 2 : $4/128 = 0.031250$ •Bit 1 : $2/128 = 0.015625$ •Bit 0 : $1/128 = 0.0078125$

5.8.8.3 TX_TXCC_MGNFS_MULT_000 (0x10140)

位	读写	复位值	描述
15:8	RW	0x0	保留
7:0	RW	0x0	指定用于根据电阻器校准值生成裕度值的乘数值。以下描述了该字段中每个位对应的乘数值。 •Bit 7 : $128/128 = 1.000000$ •Bit 6 : $64/128 = 0.500000$ •Bit 5 : $32/128 = 0.250000$ •Bit 4 : $16/128 = 0.125000$ •Bit 3 : $8/128 = 0.062500$ •Bit 2 : $4/128 = 0.031250$ •Bit 1 : $2/128 = 0.015625$ •Bit 0 : $1/128 = 0.0078125$

5.8.8.4 DRV_DIAG_TX_DRV (0x10318)

位	读写	复位值	描述
15:8	RW	0x0	保留
7	RW	0x1	TX boost enable: 通过控制进入模拟的 txda_drv_boost_en 信号, 增加快速数据转换的发送器幅度。
6	RW	0x0	保留
5:4	RW	0x2	TX boost tune: 当使用此寄存器中的 TX boost enable 位启用发射机升压功能时, 通过控制模拟 txda_drv_boost_tune 信号, 控制发送器升压幅度。以下内容指定了此字段的编码。 •2'b00: 最小增压

位	读写	复位值	描述
			•2'b01: •2'b10: •2'b11: 最大增压
3:2	RW	0x0	保留
1	RW	0x1	TX 预驱动器上拉控制: 当预驱动器被禁用时, 该位通过控制模拟 txda_drv_predv_pullup 信号来控制预驱动器输出的状态。 •1'b0: 拉低 •1'b1: 拉高
0	RW	0x1	TX driver margin-type: 通过控制进入模拟的 txda_drv_margin_type 信号, 选择驱动器将在其中操作的边距类型。 •1'b0: Classical margining -符合驱动器差分 and 共模输出阻抗规格的高功率裕度。 •1'b1: Low-Power margining -符合驱动器差分输出阻抗规范的低功率裕量。

5.8.8.5 TX_DIAG_ACYA (0x1079C)

位	读写	复位值	描述
15:1	RW	0x0	保留
0	RW	0x0	控制与 H 桥驱动器控制器相关锁存器的状态, 该锁存器在 H 桥驱动器编码器逻辑中为发送器模拟中的数字信号, 以及升压启用和电平控制信号。这些锁存器可以关闭以保持这些信号的当前状态。关闭时, 驱动锁存器输入的信号可以通过写入控制它们的寄存器来改变, 而不会影响驱动器的状态。 •1'b0: Latches transparent •1'b1: Latches gated

5.9 VPU 控制器

5.9.1 简介

VPU 提供硬件视频解码功能。Linux 中, 一个主要的视频应用框架是 GStreamer。GStreamer 是一种可配置、可插拔的视频处理框架, 可根据 GStreamer 元素实例, 在运行时动态构建视频处理流水线。每个元素提供视频解码、编码过程中需要的一部分功能。构建流水线时, 一个元素的输出连接到另一个元素的输入端。其中的解码器模块包含多种, 用于支持不同的视频标准, 这些可由软件或硬件实现。VPU 的主要功能就是这些解码器模块的硬件实现。

飞腾派中 VPU 支持的特性有:

- 性能
 - 支持格式: H. 265 (HEVC)、H. 264 (AVC)、H. 263、MPEG-4、MPEG-2、VC-1、

- VP8、FLV;
- 支持缩小
 - ◆ 高宽最小变为原来的四分之一
 - ◆ 缩小函数的输入分辨率最小是 64×64
- 支持旋转（仅 YUV-NV12 8bit 格式）
- 最大分辨率
 - ◆ HEVC: 8192×8192
 - ◆ 其他格式: 4096×4096
 - ◆ HEVC 静态: 64000×64000
 - ◆ 其他格式静态: 32000×32000
- 最小分辨率
 - ◆ VC1: 80×64
 - ◆ 其他: 64×64
- 低功耗
 - 时钟门控

5.9.2 寄存器列表

表 5-31 VPU 寄存器基地址

名称	基地址
VPU	0x000_32B0_0000

表 5-32 VPU 寄存器列表

寄存器名称	偏移	描述
MTX_ENABLE	0x0000	嵌入式处理器（MTX Core）使能寄存器

5.9.3 寄存器说明

5.9.3.1 MTX_ENABLE（0x0000）

域	位	读写	复位值	描述
MTXMajRev	31:24	R0	0x01	嵌入式处理器的 major revision
MTXMinRev	23:16	R0	0x02	嵌入式处理器的 minor revision
MTXTCaps	15:12	R0	0x9	保留
MTXArch	11	R0	0x1	表示处理器是 MTX，不是 META
reserved	10:8	R0	—	保留
MTXStepRev	7:4	R0	0x2	嵌入式处理器的 step revision
reserved	3	R0	—	保留
Tstopped	2	R0	0x0	1 表示 MTXenable 已经设置为 1，master state machine 立即进入 off 状态

域	位	读写	复位值	描述
Toff	1	RO	0x1	高电平表示 master state machine 已处于 off 状态
MTXEnable	0	RW	0x0	1: 嵌入式处理器 (MTX core) 开启 0: 嵌入式处理器 (MTX core) 关闭

5.10 GDMA 控制器

5.10.1 简介

- 最多支持 16 个独立通道，共享一组对外总线，各通道支持独立时钟关断；
- 每个通道读写支持 16 个 Outstanding，全局读写支持 16 个 Outstanding；
- 不同通道发送不同的 AXI 报文 ID，同一个通道发送的相同 AXI 报文 ID；
- 通道间请求仲裁支持 QoS 和轮询两种方式，读写独立，整体 QoS 输出独立配置；
- 支持链表 BDL 模式和直接操作模式，链表模式支持单次/循环操作，各通道独立；
- 各通道独立配置 ACE-Lite 相关的一致性信号的；
- 支持软件任意时刻 Abort 操作，硬件保证总线的正确性；
- 支持任意地址和任意数据量的读写，软件配置报文 size 和 lenth，读写独立；
- AXI 数据位宽 128bits，地址位宽 40bits，读写支持 FIXED 和 INCR 两种传输模式；
- 支持可配置中断上报，各通道可独立控制，对外输出一个中断线。

5.10.2 操作说明

5.10.2.1 直接模式功能

软件实现直接模式功能时，以操作 Channel0 为例，寄存器配置流程如下：

1. 写 dma_c0_mode[0] 为 0，表示工作在 Direct 模式（default 模式）；
2. 当使用 QoS 策略进行仲裁的时候：
 - a. 对于写请求仲裁，写寄存器 dma_c0_mode[8] 使能该通道 QoS 替换功能，同时写寄存器 dma_c0_mode [15:12]，配置该通道写请求的 QoS 值，QoS 值依据不同通道的优先级来确定，值越高，优先级越高；
 - b. 对于读请求仲裁，写寄存器 dma_c0_mode[16] 使能该通道 QoS 替换功能，同时写寄存器 dma_c0_mode [23:20]，配置该通道读请求 QoS 值；
 - c. 如不使能 QoS 的替换功能，则默认读写的 QoS 值为 0。
3. 写 dma_c0_intr_ctl 寄存器，用来控制中断输出使能；
4. 写 dma_c0_ts 寄存器，表示当前 DMA 要搬移的数据总量，总量必须是 size 整数倍，当前传输 size 配置值支持 0, 1, 2, 3 和 4，分别对应 1Byte, 2Byte, 4Byte,

- 8Byte, 16Byte 的数据;
5. 写 dma_c0_xfer_cfg 寄存器, 用来配置读写操作的 AXI 传输类型, 注意: 报文传输的总数据量必须是 axsize 的整数倍, 且报文传输的 axsize 必须和首地址对齐;
 6. 写寄存器 dma_c0_upsaddr, dma_c0_lwsaddr, dma_c0_updaddr, dma_c0_lwdaddr, 分别用来表示数据源地址和目的地址, 注意首地址必须与 size 对齐;
 7. 写寄存器 dma_c0_ctl[0] 为 1, 表示使能该通道;
 8. 写寄存器 dma_intr_ctl 寄存器, 用来控制全局中断输出使能;
 9. 写 dma_c0_aw_cfg 和 dma_c0_ar_cfg 寄存器, 配置与数据一致性有关的 AXI 信号;
 10. 参考步骤 2, 若此时需要在内部仲裁选取 QoS 模式, 则写寄存器 dma_ctrl[5:4] 为 1, 否则采用轮询模式仲裁;
 11. 写 dma1_ctrl[0] 为 1, 随后 DMA 开始进行数据搬移。
- 上述步骤中, 1~9 步骤没有特殊顺序要求。

5.10.2.2 链表模式功能

软件实现链表模式功能时, 需要预先在内存中按照格式要求初始化一个 BDL 表格, 并将该表格的首地址配置到内部源地址寄存器上。该表格的每一个条目包含了实际数据存放源地址和搬移的目的地址, 以及数据长度、是否产生中断等控制信息; 当 DMA 完成对这些数据的搬移后, 自动读取下一个链表条目, 以此类推; 当所有条目读取完成后, 可通过寄存器配置, 设定是否从第一个条目再次获取数据, BDL 基本格式如下表所示。

表 5-33 BDL 基本格式

偏移	描述
0x1C	该条目传输完成中断产生控制位
0x18	传输数据总量, 以 Byte 为基本单位
0x14	数据目标控制 Burst [1:0] Size: [6:4] lenth : [15:8]
0x10	数据源控制 Burst [1:0] Size: [6:4] lenth : [15:8]
0x0C	数据目的地址高位
0x08	数据目的地址低位
0x04	数据源地址高位
0x00	数据源地址低位

以操作 Channel0 为例, 寄存器配置流程如下:

1. 写寄存器 dma_c0_mode[0] 为 1, 表示工作在链表 BDL 模式, 若需要启动 BDL 轮

- 转功能，则还需要写 `dma_c0_mode[4]` 为 1；
2. 当使用 QoS 策略进行仲裁的时候：
 - a. 对于写请求仲裁，写寄存器 `dma_c0_mode[8]` 使能该通道 QoS 替换功能，同时写寄存器 `dma_c0_mode [15:12]`，配置该通道的写请求的 QoS 值；
 - b. 对于读请求仲裁，写寄存器 `dma_c0_mode[16]` 使能该通道 QoS 替换功能，同时写寄存器 `dma_c0_mode [23:20]`，配置该通道的读请求的 QoS 值；
 - c. 如不使能 QoS 的替换功能，则默认读写的 QoS 值为 0。
 3. 写 `dma_c0_intr_ctl` 寄存器，用来控制中断输出使能；
 4. 写寄存器 `dma_c0_upsaddr`, `dma_c0_lwsaddr`，用来表示当前 BDL 列表在内存中的位置，注意：BDL 地址必须是 128Byte 对齐；
 5. 写 `dma_c0_lvi` 寄存器，用来表示当前 BDL 列表中有效的条目，实际有效条目数量为该寄存器值+1；
 6. 写寄存器 `dma_c0_ctl[0]` 为 1，表示使能该通道；
 7. 写寄存器 `dma_intr_ctl` 寄存器，用来控制全局中断输出使能；
 8. 写 `dma_c0_aw_cfg` 和 `dma_c0_ar_cfg` 寄存器，配置与数据一致性有关的 AXI 信号；
 9. 参考步骤 2，若此时需要在内部仲裁选取 QoS 模式，则写寄存器 `dma_ctrl[5:4]` 为 1，否则采用轮询模式仲裁；
 10. 写 `dma_ctrl[0]` 为 1，随后 DMA 开始进行数据搬移。
- 上述步骤中，1~9 步骤没有特殊顺序要求。

5.10.2.3 Abort 处理流程

正常传输过程中，如果软件想要中断当前某个通道，必须遵守如下顺序，以免造成整个系统出现错误，以停止 Channel0 为例：

1. 写寄存器 `dma_c0_ctl[0]` 为 0，即 `disable` 该通道；
2. 软件读取 `dma_c0_state[4]`，只有该状态位是 0 的时候，表示该通道已完成当下 AXI 报文请求；
3. 软件写 `dma_c0_ctl[1]` 为 1，即软复位该通道，重新配置最新的传输参数，然后使能该通道即可；若想继续操作，可不进行软复位操作，再次 `enable` 该通道，操作会延续 `disable` 之前的进行；
4. 若需要再次启动该通道，参考之前的操作说明。

5.10.2.4 中断处理流程

当前所有中断都是可屏蔽的，当 DMA 控制器产生中断后：

1. 软件根据寄存器列表信息，查询 dma_state 寄存器，判断是当前哪个通道产生中断请求；
2. 根据判断，以 channel0 为例，读取当前 dam_c0_state 寄存器，判断中断产生源，软件以此作为后续操作依据。

5.10.3 寄存器列表

表 5-34 GDMA 寄存器基地址

名称	基地址
DMAC	0x000_32B3_4000

表 5-35 GDMA 寄存器列表

寄存器	偏移量	描述
dma_ctl	0x000	全局控制类寄存器
dma_stat	0x004	中断状态寄存器
dma_intr_ctl	0x008	中断使能控制寄存器
dma_lp_ctl	0x00C	每一位对应一个通道时钟开启和关断
dma_qos_cfg	0x010	读写请求的 QoS 配置
dma_Cx_ctl	0x020+x*0x60	channel x 软复位信号与使能控制信号
dma_Cx_mode	0x024+x*0x60	Channel x 模式寄存器
dma_Cx_intr_ctl	0x028+x*0x60	Channel x 中断输出控制寄存器
dma_Cx_state	0x02c+x*0x60	Channel x 状态寄存器
dma_Cx_lvi	0x030+x*0x60	Channel x 在链表模式下 BDL 有效条目
dma_Cx_ts	0x034+x*0x60	channel x 在 direct 模式下操作的总 Byte 数据量
dma_Cx_upsaddr	0x038+x*0x60	通道源地址寄存器，BDL 模式下，该地址为 BDL 列表的地址
dma_Cx_lwsaddr	0x03c+x*0x60	通道源地址寄存器，BDL 模式下，该地址为 BDL 列表的地址
dma_Cx_updaddr	0x040+x*0x60	通道目的地址寄存器，BDL 模式下，该地址无意义
dma_Cx_lwdaddr	0x044+x*0x60	通道目的地址寄存器，BDL 模式下，该地址无意义
dma_Cx_xfer_cfg	0x048+x*0x60	Channel x 读写请求 burst 信息
dma_Cx_lcp	0x04c+x*0x60	只读寄存器，当 BDL 模式下的时候，用来表示当前操作了多少个 BDL 列表
dma_Cx_secctl	0x050+x*0x60	安全控制寄存器
dma_Cx_sec_atst	0x054+x*0x60	secssid 和 atst 信号控制寄存器
dma_Cx_nsid_strmid	0x058+x*0x60	NSAID 和 streamID 控制寄存器
dma_Cx_aw_cfg	0x05c+x*0x60	控制 axi aw 通道配置寄存器
dma_Cx_ar_cfg	0x060+x*0x60	控制 axi ar 通道配置寄存器
dma_Cx_secrsp	0x064+x*0x60	dma 通道 response 安全控制寄存器

注：该表中 x 的取值范围为 0~15。

5.10.4 寄存器说明

下列章节中 x 的取值范围均为 0~15。

5.10.4.1 DMA_CTL (0x000)

域	位	读写	复位值	描述
dam_ot_ctl	11:8	RW	0xF	dma 传输 outstanding 控制, 实际数量为该寄存器值 +1。
dma_rd_arb_mode	5	RW	0x0	dma 读请求仲裁模式。 0: 轮询模式; 1: 采用 QoS 判断模式。
dma_wr_arb_mode	4	RW	0x0	dma 写请求仲裁模式 0: 轮询模式; 1: 采用 QoS 判断模式。
dma_srst	1	RW	0x0	dma 控制器软复位信号。 1 表示进行软复位, 写 0 退出。
dma_enable	0	RW	0x0	dma 控制器使能信号。 1 表示使能, 0 表示禁用。
reserved	Others	R0	0x0	保留

5.10.4.2 DMA_STATE (0x004)

域	位	读写	复位值	描述
channel15_intr_state	15	R0	0x0	channel15 中断状态标志位。 1 表示有中断, 0 表示无中断。
channel14_intr_state	14	R0	0x0	channel14 中断状态标志位。 1 表示有中断, 0 表示无中断。
channel13_intr_state	13	R0	0x0	channel13 中断状态标志位。 1 表示有中断, 0 表示无中断。
channel12_intr_state	12	R0	0x0	channel12 中断状态标志位。 1 表示有中断, 0 表示无中断。
channel11_intr_state	11	R0	0x0	channel11 中断状态标志位。 1 表示有中断, 0 表示无中断。
channel10_intr_state	10	R0	0x0	channel10 中断状态标志位。 1 表示有中断, 0 表示无中断。
channel9_intr_state	9	R0	0x0	channel9 中断状态标志位。 1 表示有中断, 0 表示无中断。
channel8_intr_state	8	R0	0x0	channel8 中断状态标志位。 1 表示有中断, 0 表示无中断。
channel7_intr_state	7	R0	0x0	channel7 中断状态标志位。 1 表示有中断, 0 表示无中断。
channel6_intr_state	6	R0	0x0	channel6 中断状态标志位。 1 表示有中断, 0 表示无中断。
channel5_intr_state	5	R0	0x0	channel5 中断状态标志位。 1 表示有中断, 0 表示无中断。
channel4_intr_state	4	R0	0x0	channel4 中断状态标志位。 1 表示有中断, 0 表示无中断。
channel3_intr_state	3	R0	0x0	channel3 中断状态标志位。 1 表示有中断, 0 表示无中断。
channel2_intr_state	2	R0	0x0	channel2 中断状态标志位。

域	位	读写	复位值	描述
				1 表示有中断, 0 表示无中断。
channel1_intr_state	1	R0	0x0	channel1 中断状态标志位。 1 表示有中断, 0 表示无中断。
channel0_intr_state	0	R0	0x0	channel0 中断状态标志位。 1 表示有中断, 0 表示无中断。
reserved	others	R0	0x0	保留

5.10.4.3 DMA_INTR_CTL (0x008)

域	位	读写	复位值	描述
global_intr_enable	31	RW	0x0	全局中断输出 mask 控制信号, 1 表示允许中断输出, 0 表示不允许。
channel15_intr_enable	15	RW	0x0	channel15 中断使能控制, 1 表示使能, 0 表示不使能。
channel14_intr_enable	14	RW	0x0	channel14 中断使能控制, 1 表示使能, 0 表示不使能。
channel13_intr_enable	13	RW	0x0	channel13 中断使能控制, 1 表示使能, 0 表示不使能。
channel12_intr_enable	12	RW	0x0	channel12 中断使能控制, 1 表示使能, 0 表示不使能。
channel11_intr_enable	11	RW	0x0	channel11 中断使能控制, 1 表示使能, 0 表示不使能。
channel10_intr_enable	10	RW	0x0	channel10 中断使能控制, 1 表示使能, 0 表示不使能。
channel9_intr_enable	9	RW	0x0	channel9 中断使能控制, 1 表示使能, 0 表示不使能。
channel8_intr_enable	8	RW	0x0	channel8 中断使能控制, 1 表示使能, 0 表示不使能。
channel7_intr_enable	7	RW	0x0	channel7 中断使能控制, 1 表示使能, 0 表示不使能。
channel6_intr_enable	6	RW	0x0	channel6 中断使能控制, 1 表示使能, 0 表示不使能。
channel5_intr_enable	5	RW	0x0	channel5 中断使能控制, 1 表示使能, 0 表示不使能。
channel4_intr_enable	4	RW	0x0	channel4 中断使能控制, 1 表示使能, 0 表示不使能。
channel3_intr_enable	3	RW	0x0	channel3 中断使能控制, 1 表示使能, 0 表示不使能。
channel2_intr_enable	2	RW	0x0	channel2 中断使能控制, 1 表示使能, 0 表示不使能。
channel1_intr_enable	1	RW	0x0	channel1 中断使能控制, 1 表示使能, 0 表示不使能。
channel0_intr_enable	0	RW	0x0	channel0 中断使能控制, 1 表示使能, 0 表示不使能。

域	位	读写	复位值	描述
reserved	others	R0	0x0	保留

5.10.4.4 DMA_LP_CTL (0x00C)

域	位	读写	复位值	描述
channel low power control	31:0	RW	0x0	每一位对应一个通道时钟开启和关断，例如 bit[0] 对应第一个通道，bit[15] 对应最后一个通道；1 表示关断，默认开启。

5.10.4.5 DMA_QOS_CFG (0x010)

域	位	读写	复位值	描述
arqos	7:4	RW	0x0	读请求的 QoS 配置
awqos	3:0	RW	0x0	写请求的 QoS 配置
reserved	others	RW	0x0	保留

5.10.4.6 DMA_Cx_CTRL (0x020+x*0x60)

域	位	读写	复位值	描述
cx_srst	4	RW	0x0	channel x 软复位信号，1 表示进行软复位，写 0 退出。
cx_enable	0	RW	0x0	channel x 使能控制信号，1 表示使能该通道，0 表示不使能。
reserved	others	R0	0x0	保留

5.10.4.7 DMA_Cx_MODE (0x024+x*0x60)

域	位	读写	复位值	描述
cx_rd_qos	23:20	RW	0x0	Channel x 读请求 QoS 值配置，取值范围为 0~15。读请求仲裁时，采用各通道的 QoS 值来控制读请求优先级。值越大，优先级越高。
cx_rd_qos_enable	16	RW	0x0	是否用 cx_rd_qos 的值替换该通道的去请求 QoS，1 表示替换，0 不替换。
cx_wr_qos	15:12	RW	0x0	Channel x 写请求 QoS 值配置，取值范围为 0~15。写请求仲裁时，采用各通道的 QoS 值来控制写请求优先级。值越大，优先级越高。
cx_wr_qos_enable	8	RW	0x0	是否用 cx_wr_qos 中的值替换该通道写请求的 QoS：1 表示替换；0 不替换。
cx_bdl_rol1	4	RW	0x0	当配置为 BDL 链表模式的时候，完成当前一个 BDL 列表是否重新开始。1 表示循环使用链表进行数据搬运；0 表示当前链表完成后即停止。
cx_mode	0	RW	0x0	配置当前采用 direct 或者 BDL 链表模式：0：采用 Direct 模式(default) 1：采用 BDL 链表模式
reserved	others	R0	0x0	保留

5.10.4.8 DMA_Cx_INTR_CTL (0x028+x*0x60)

域	位	读写	复位值	描述
channel _x _trans_end_enable	3	RW	0x0	channel _x 所有数据传输完成中断输出控制, 1 表示允许输出, 0 表示不允许。
channel _x _bdl_end_enable	2	RW	0x0	BDL 模式下一个 BDL 条目数据传输完成中断输出控制, 1 表示允许输出, 0 表示不允许。
channel _x _fifo_full_enable	1	RW	0x0	channel _x FIFO 满中断输出控制, 1 表示允许输出, 0 表示不允许。
channel _x _fifo_empty_enable	0	RW	0x0	channel _x FIFO 空中断输出控制, 1 表示允许输出, 0 表示不允许。
reserved	others	RO	0x0	保留

5.10.4.9 DMA_Cx_STATE (0x02C+x*0x60)

域	位	读写	复位值	描述
channel _x _busy	4	RW	0x0	channel _x 处于数据传输中, 写 1 清 0。
channel _x _trans_end	3	RW	0x0	channel _x 传输完成状态, direct 模式时, 表示所有数据传输完成, BDL 模式表示当前所有 BDL 条目传输完成; 写 1 清 0。
channel _x _bdl_end	2	RW	0x0	在 BDL 模式下, 表示当前一个 BDL 条目数据传输完成; Direct 模式下该位始终为; 0 写 1 清 0。
channel _x _fifo_full	1	RW	0x0	channel _x FIFO 满状态, 写 1 清 0。
channel _x _fifo_empty	0	RW	0x0	channel _x FIFO 空状态, 写 1 清 0。
reserved	others	RO	0x0	保留

5.10.4.10 DMA_Cx_LVI (0x030+x*0x60)

域	位	读写	复位值	描述
dma_cx_lvi	31:0	RW	0x0	channel _x last valid index, 即链表模式下 BDL 有效条目, 实际有效条目=该寄存器值+1。

5.10.4.11 DMA_Cx_TS (0x034+x*0x60)

域	位	读写	复位值	描述
dma_cx_ts	31:0	RW	0x0	channel _x 在 direct 模式下操作的总 Byte 数据量

5.10.4.12 DMA_Cx_UPSADDR (0x038+x*0x60)

域	位	读写	复位值	描述
dma_cx_upsaddr	31:0	RW	0x0	Channel _x 源地址高 32bits

5.10.4.13 DMA_Cx_LWSADDR (0x03C+x*0x60)

域	位	读写	复位值	描述
dma_cx_lwsaddr	31:0	RW	0x0	Channel x 源地址低 32bits, 6:0 固定为 0

5.10.4.14 DMA_Cx_UPDADDR (0x040+x*0x60)

域	位	读写	复位值	描述
dma_cx_updaddr	31:0	RW	0x0	Channel x 目的地址高 32bits

5.10.4.15 DMA_Cx_LWDADDR (0x044+x*0x60)

域	位	读写	复位值	描述
dma_cx_lwdaddr	31:0	RW	0x0	Channel x 目的地址低 32bits, 6:0 固定为 0

5.10.4.16 DMA_Cx_XFER_CFG (0x048+x*0x60)

域	位	读写	复位值	描述
dma_cx_arlen	31:24	RW	0x0	Channel x 读请求 burst lenth 大小, 最多为 8。
dma_cx_arsize	22:20	RW	0x0	Channel x 读请求 size 大小, 支持 1,2,4,8,16-Byte。
dma_cx_arburst	17:16	RW	0x0	Channel x 读请求 burst 类型。 2'h0: fix; 2'h1: incr。
dma_cx_awlen	15:8	RW	0x0	Channel x 写请求 burst lenth 大小, 最多为 8。
dma_cx_awsiz	6:4	RW	0x0	Channel x 写请求 size 大小, 支持 1,2,4,8,16-Byte。
dma_cx_awburst	1:0	RW	0x0	Channel x 写请求 burst 类型。 2'h0: fix; 2'h1: incr。

5.10.4.17 DMA_Cx_LCP (0x04C+x*0x60)

域	位	读写	复位值	描述
dma_cx_lcp	31:0	RO	0x0	只读寄存器, 当 BDL 模式下的时候, 用来表示当前操作了多少个 BDL 列表, 直接模式下该寄存器值为 0, 该寄存器读清零。

5.10.4.18 DMA_Cx_SECCTL (0x050+x*0x60)

域	位	读写	复位值	描述
dma_cx_secctl	31:0	RW	0x0	安全控制寄存器, 仅安全态可访问

5.10.4.19 DMA_Cx_SEC_ATST (0x054+x*0x60)

域	位	读写	复位值	描述
dma_cx_sec_atst	31:0	RW	0x0	secssid 和 atst 信号控制寄存器

5.10.4.20 DMA_Cx_NSID_STRMID (0x058+x*0x60)

域	位	读写	复位值	描述
dma_cx_nsid_strmid	31:0	RW	0x0	NSAID 和 streamID 控制寄存器。 3:0 表示 ssmmu_id, 7:4 表示 NASID。

5.10.4.21 DMA_Cx_AW_CFG (0x05C+x*0x60)

域	位	读写	复位值	描述
reserved	others	RW	0x0	保留
dma_cx_awbar	21:20	RW	0x0	控制总线 awbar 的值
dma_cx_awsnoop	18:16	RW	0x0	控制总线 awsnoop 的值
dma_cx_awdomain	13:12	RW	0x0	控制总线 awdomain 的值
dma_cx_awprot	9:8	RW	0x0	控制 axi aw 通道 PORT[0] 和 PORT[2] 位域, PORT[1] 位域由安全控制寄存器决定。
dma_cx_awregion	7:4	RW	0x0	控制总线 awregion 的值
dma_cx_awcache	3:0	RW	0x0	控制总线 awcache 的值

5.10.4.22 DMA_Cx_AR_CFG (0x060+x*0x60)

域	位	读写	复位值	描述
reserved	others	RW	0x0	保留
dma_cx_arbar	21:20	RW	0x0	控制总线 arbar 的值
dma_cx_arsnoop	19:16	RW	0x0	控制总线 arsnoop 的值
dma_cx_ardomain	13:12	RW	0x0	控制总线 ardomain 的值
dma_cx_arprot	9:8	RW	0x0	控制 axi ar 通道的 PORT[0] 和 PORT[2] 位域, PORT[1] 位域由安全控制寄存器决定。
dma_cx_arregion	7:4	RW	0x0	控制总线 arregion 的值
dma_cx_arcache	3:0	RW	0x0	控制总线 arcache 的值

5.10.4.23 DMA_Cx_SECRSP (0x064+x*0x60)

域	位	读写	复位值	描述
dma_cx_secrsp	0	RW	0x0	dma 通道 response 安全控制位, 只有安全请求能访问, 其余位保留。

5.11 DDMA 控制器

5.11.1 简介

- 支持 8 个 channel 工作, 每个 channel 可接收 32 个外设的 DMA 请求信号, 软件配置选择服务任何一个外设请求;
- 支持各通道独立关断功能;

- AXI 总线读写通道分离操作；
- 各通道包含独立缓存 FIFO，且 FIFO 读写数据方向可控；
- 支持外设 dma 请求超时处理机制；
- 支持中断。

5.11.2 操作说明

5.11.2.1 从内存搬移数据到外设

当某个通道配置的是接收 dma_tx_req 信号，即需要 DMA 从内存读取数据到本地，并写入外设，则基本流程如下：

- 通道使能后，上行 AXI 读数据逻辑检测到该通道的属性，并根据配置的 block 尺寸，即 dma_chal_ts 寄存器，开始搬移一定量的数据到本地寄存器，第一次的时候，若 block ≥ 64Byte，则搬移 64Byte，否则搬移 block 的尺寸，后续传输，若剩余等待传输数据 < 64Byte，则余下多少传输多少，若剩余等待传输的数据 ≥ 64Byte，仍传输 64Byte；
- 下行 AXI 通道中，检测到有 dma_tx_req 的请求，即边沿检测，且本地 FIFO 中有数据，则开始发送 4Byte 的数据到外设，发送完成后，将 dma_tx_ack 信号拉高，随后待 dma_tx_req 拉低之后，再次拉低 dma_tx_ack 信号，完成一次请求，并开始等待下一轮的传输。

传输边界情况考虑如下：

本地逻辑判断，例如，当内存中只有 32byte 的数据需要传送，则上行 AXI 这一次只取回 32Byte 的数据到本地，数据读取完成后，暂停上行 AXI 对这个通道的服务；等到下行请求，当所有数据都发送到外设的时候，硬件内部停止该通道，即主动将该通道使能信号拉低，产生中断，等待软件再次启动。

5.11.2.2 从外设搬移数据到内存

当某个通道配置的是接收 dma_rd_req 信号，即需要从外设读取数据到本地，DMA 需要将本地数据发送到内存，则基本流程如下：

- 通道使能后，下行 AXI 检测到 dma_rx_req 的请求，且检测到本地 FIFO 非满，则开始发送 AXI 读请求，读取 4Byte 的数据到本地，并完成 DMA 请求的握手操作；若 FIFO 满了，则暂时不处理这个请求；
- 上行 AXI 检测到本地 FIFO 中可用的数据量，大于下一次传输 len 的时候，则准备发送上行的 AXI 报文，当 block ≥ 64 的时候，等待有 64Byte 的时候载发送，若当前余下不足 64Byte 报文，则发送长度等不到 64Byte 就传输。

传输边界情况考虑如下：

例如，本地逻辑判断，内存中只剩余 24byte 的空间可以存放数据，等到本地计数表明，已经读取到了 24Byte 的数据之后，则不再接收 dma_rx_req 的请求，同时，等待 AXI 将这 24Byte 发送到内存完成后，主动暂停 DMA 这个通道，产生中断，等待软件处理。对于 dma_rx_req 还增加了超时处理机制，即当外设长时间没有数据请求的时候，会主动停止对该通道的服务，释放功能。

注意：

- 软件在配置待传送数据总长度的时候，最小是 4byte；
- 若软件配置的总数据长度是 64Byte，则可能实际应用中，只有 8Byte 是有效数据，但是 DMA 不做数据的处理，数据的填补整合交给软件去完成。

5.11.2.3 DMA 请求信号的握手处理机制

对于外围设备的 req&ack 是一个 4 段握手的机制，每个通道有一个独有的状态机，实现握手机制。当通道使能后，即检测 req 是否有效，若有效，则状态跳转到 WAIT_ACK，等待数据与外设交互完成后，使能 ack 信号，完成后续握手机制。

5.11.2.4 启动配置流程

1. 写 dma_ctl[0] 位，disable 整个 DMA；
2. 写 dma_ctl[1] 位，复位内部状态，随后写 dma_ctl[1] 为 0，跳出复位；
3. （以下配置不区分先后顺序）：
 - 写 dma_chal_config 寄存器，设定各通道 dma_req 选择，注意，软件在配置各通道请求信号线的选择时候后，要保证与该通道配置的属性一致；
 - 写 dma_upaxi_awconfig、dma_upaxi_arconfig 寄存器，配置上行 AXI 读写地址通道部分信号值；
 - 写 dma_chalx_ddr_upaddr 和 dma_chalx_ddr_lwaddr 寄存器，配置每个通道需要访问的内存中的地址；
 - 写 dma_chal_dev_addr 寄存器，配置每个通道访问外设寄存器地址；
 - 写 dma_chal_ts 寄存器，配置每个通道传输数据的总数量(Byte 基本单位，数目是 4Byte 的整数倍，若非整数倍，则可能出现数据丢失的情况)。
4. 写 dma_chal_ctl[2] 位，配置该通道传输模式，选择接收 dma_tx_req 或者 dma_rx_req；注意，通道模式的选择，与 dma_chal_config 的选择要匹配；
5. 若通道配置接收 dma_rx_req，配置是否开启超时处理机制寄存器 dma_chalx_timeout_cnt；
6. 写 dma_chal_ctl[0] 位，使能该通道；

7. 写 dma_ctl[0] 位 1，使能整个 DMA。

5.11.2.5 暂停配置流程

若想暂停某个通道工作，做如下操作：

1. 写 dma_chal_ctl[0] 位为 0，暂停通道；
2. 通道暂停后，等待一段事件，若软件读取 dma_chal_ctl[0] 位为 0，表示该通道已经停止，此时，软件才可以写入 dma_chal_ctl[1] 位，复位整个通道；
3. 按照上述流程，重新配置寄存器，并启动。

5.11.2.6 超时机制

超时机制，主要是针对接收外设 dam_rx_req 的请求，这是因为对于外设控制器，系统很难确 dma 需要读取外设数据的总量，因此，当某个通道绑定好接收 dam_rx_req 请求的时候，若外设长时间没有拉高 dam_rx_req，且本地该 dma 通道仅缓存了部分数据，但数据量尚未达到向 AXI 发送写报文的阈值，因此启动超时机制，即在长时间没有收到 dma 请求后，主动将内部剩余的数据写入 ddr，并暂停该通道。该超时机制主要针对 chal_mode 配置为 1 的情况，且超时机制可以通过写寄存器来控制是否使能。

5.11.2.7 中断说明

DMA 控制器，只有当通道完成当前配置数据所有搬移后，才产生一个中断，做如下说明：

1. 当数据流方向从内存到外设，则当下行 AXI 向外设写完最后 4Byte 数据后，主动暂停该通道工作，并产生中断，等待软件后续配置操作；
2. 当数据流方向从外设到内存，则当上行 AXI 向内存写完最后以此数据后，主动暂停该通道工作。当在收到 bvalid 信号后，置位状态寄存器，并输出中断。

表 5-36 DDMA 通道 slave ID 分配表

slave ID	说明	slave ID	说明
DMA0			
0	can0_tx	13	can0_rx
1	can1_tx	14	can1_rx
2	uart0_tx	15	uart0_rx
3	uart1_tx	16	uart1_rx
4	uart2_tx	17	uart2_rx
5	uart3_tx	18	uart3_rx
6	spim0_tx	19	spim0_rx
7	spim1_tx	20	spim1_rx
8	spim2_tx	21	spim2_rx

slave ID	说明	slave ID	说明
9	spim3_tx	22	spim3_rx
10	–	–	保留
11	–	–	保留
12	smbus_tx	25	smbus_rx
DMA1			
0	mio0_tx	16	mio0_rx
1	mio1_tx	17	mio1_rx
2	mio2_tx	18	mio2_rx
3	mio3_tx	19	mio3_rx
4	mio4_tx	20	mio4_rx
5	mio5_tx	21	mio5_rx
6	mio6_tx	22	mio6_rx
7	mio7_tx	23	mio7_rx
8	mio8_tx	24	mio8_rx
9	mio9_tx	25	mio9_rx
10	mio10_tx	26	mio10_rx
11	mio11_tx	27	mio11_rx
12	mio12_tx	28	mio12_rx
13	mio13_tx	29	mio13_rx
14	mio14_tx	30	mio14_rx
15	mio15_tx	31	mio15_rx

5.11.3 寄存器列表

表 5-37 DDMA 寄存器基地址

名称	基地址
DMA0	0x000_2800_3000
DMA1	0x000_2800_4000

表 5-38 DDMA 寄存器列表

寄存器	偏移量	描述
全局类寄存器		
dma_ctl	0x00	全局控制类寄存器
dma_chal_cfg	0x04	各通道 DMA 请求信号线选择配置信号
dma_stat	0x08	中断状态寄存器
dma_mask_int	0x0C	中断掩码寄存器
dma_upaxi_awcfg	0x10	用来配置上行 AXI 写通道部分信号线值
dma_upaxi_arcfg	0x14	用来配置上行 AXI 读通道部分信号线值
dma_dwnaxi_awcfg	0x18	用来配置下行 AXI 写通道部分信号线值
dma_dwnaxi_arcfg	0x1C	用来配置下行 AXI 读通道部分信号线值
dma_channel_bind	0x20	用来表示，某个通道是否被绑定 DAM 请求信号线，给软件提供参考。
dma_gcap	0x24	只读寄存器，用来告知软件当前设计中支持通道的数目： 0 表示支持 1 个通道

寄存器	偏移量	描述
		1 表示支持 2 个通道 ...
Channel x 寄存器 (x 取值为 0~7)		
<code>dma_chalx_ddr_upaddr</code>	<code>0x40+0x40*x</code>	数据在内存中的高 32 位地址, 可以是数据源地址, 也可以使数据目的地址。
<code>dma_chalx_ddr_lwaddr</code>	<code>0x44+0x40*x</code>	数据在内存中的低 32 位地址, 可以是数据源地址, 也可以使数据目的地址。
<code>dma_chalx_dev_addr</code>	<code>0x48+0x40*x</code>	设备地址, 可以是数据源地址, 也可以是数据目的地址。
<code>dma_chalx_ts</code>	<code>0x4C+0x40*x</code>	软件配置的, 需要传输的总的文件大小, 对应内存中的地址软件申请应该是 64Byte 对齐的。
<code>dma_chalx_crt_upaddr</code>	<code>0x50+0x40*x</code>	当前内存需要读写数据的高 32 位地址
<code>dma_chalx_crt_lwaddr</code>	<code>0x54+0x40*x</code>	当前内存需要读写数据的低 32 位地址
<code>dma_chalx_ctl</code>	<code>0x58+0x40*x</code>	Channel x 的控制寄存器
<code>dma_chalx_sts</code>	<code>0x5C+0x40*x</code>	Channel x 的状态寄存器
<code>dma_chalx_timeout_cnt</code>	<code>0x60+0x40*x</code>	使能超时机制时, 超时等待阈值时间。

5.11.4 寄存器说明

下列章节中 x 的取值范围均为 0~7。

5.11.4.1 dma_ctl (0x00)

域	位	读写	复位值	描述
<code>reserved</code>	31:2	RO	0x0	保留
<code>dma_srst</code>	1	RW	0x0	DMA 全局软复位控制信号 1: 软复位 0: 非软复位
<code>dma_enable</code>	0	RW	0x0	DMA 全局使能控制信号 1: enable 0: disable

5.11.4.2 dma_chal_cfg (0x04)

域	位	读写	复位值	描述
<code>chal3_sel_en</code>	31	RW	0x0	通道 3 的 dma 请求信号源端选择是否有效标志, 只有为 1 的才表示 <code>chal3_sel</code> 配置请求源选择有效。
<code>chal3_sel</code>	30:24	RW	0x0	通道 3 的 dma 请求信号源端选择, 同一时刻, 只有 1 位有效, 实现 32 选 1 的功能。
<code>chal2_sel_en</code>	23	RW	0x0	通道 2 的 dma 请求信号源端选择是否有效标志, 只有为 1 的才表示 <code>chal2_sel</code> 配置请求源选择有效。
<code>chal2_sel</code>	22:16	RW	0x0	通道 2 的 dma 请求信号源端选择, 同一时刻, 只有 1 位有效, 实现 32 选 1 的功能。
<code>chal1_sel_en</code>	15	RW	0x0	通道 1 的 dma 请求信号源端选择是否有效标志, 只有为 1 的才表示 <code>chal1_sel</code> 配置请求源选择有效。
<code>chal1_sel</code>	14:8	RW	0x0	通道 1 的 dma 请求信号源端选择, 同一时刻, 只有

域	位	读写	复位值	描述
				1 位有效，实现 32 选 1 的功能。
chal0_sel_en	7	RW	0x0	通道 0 的 dma 请求信号线源端选择是否有效标志，只有为 1 的才表示 chal0_sel 配置的请求源选择有效。
chal0_sel	6:0	RW	0x0	通道 0 的 dma 请求信号线源端选择，同一时刻，只有 1 位有效，实现 32 选 1 的功能。

5.11.4.3 dma_sta (0x08)

域	位	读写	复位值	描述
reserved	31:29	R0	0x0	保留
chal7_sel	28	RW	0x0	通道 7 的 dma 请求 block 传输完成，该位置 1，软件写 1 清 0。
reserved	27:25	R0	0x0	保留
chal6_sel	24	RW	0x0	通道 6 的 dma 请求 block 传输完成，该位置 1，软件写 1 清 0。
reserved	23:21	R0	0x0	保留
chal5_sel	20	RW	0x0	通道 5 的 dma 请求 block 传输完成，该位置 1，软件写 1 清 0。
reserved	19:17	R0	0x0	保留
chal4_sel	16	RW	0x0	通道 4 的 dma 请求 block 传输完成，该位置 1，软件写 1 清 0。
reserved	15:13	R0	0x0	保留
chal3_sel	12	RW	0x0	通道 3 的 dma 请求 block 传输完成，该位置 1，软件写 1 清 0。
reserved	11:9	R0	0x0	保留
chal2_sel	8	RW	0x0	通道 2 的 dma 请求 block 传输完成，该位置 1，软件写 1 清 0。
reserved	7:5	R0	0x0	保留
chal1_sel	4	RW	0x0	通道 1 的 dma 请求 block 传输完成，该位置 1，软件写 1 清 0。
reserved	3:1	R0	0x0	保留
chal0_sel	0	RW	0x0	通道 0 的 dma 请求 block 传输完成，该位置 1，软件写 1 清 0。

5.11.4.4 dma_mask_int (0x0C)

域	位	读写	复位值	描述
global_en	31	RW	0x0	全局中断使能输出控制位
reserved	30:7	R0	0x0	保留
chal7_mask	7	RW	0x0	通道 7 的 dma 请求传输完成，中断输出 mask 控制信号
chal6_mask	6	RW	0x0	通道 6 的 dma 请求传输完成，中断输出 mask 控制信号
chal5_mask	5	RW	0x0	通道 5 的 dma 请求传输完成，中断输出 mask 控制信号
chal3_mask	4	RW	0x0	通道 4 的 dma 请求传输完成，中断输出 mask 控制信号
chal3_mask	3	RW	0x0	通道 3 的 dma 请求传输完成，中断输出 mask 控制信号

域	位	读写	复位值	描述
chal2_mask	2	RW	0x0	通道 2 的 dma 请求传输完成, 中断输出 mask 控制信号
chal1_mask	1	RW	0x0	通道 1 的 dma 请求传输完成, 中断输出 mask 控制信号
chal0_mask	0	RW	0x0	通道 0 的 dma 请求传输完成, 中断输出 mask 控制信号

5.11.4.5 dma_upaxi_awconfg (0x10)

域	位	读写	复位值	描述
dma_upaxi_awconfg	31:0	RW	0x0	用来配置上行 AXI 写地址通道的部分信号值[3:0]:配置 awcache 信号值 [7:4]: 配置 awregion 信号值 [10:8]: 配置 awsnoop 信号值 [12:11]: 配置 awbar 信号值 [14:13]: 配置 awdomain 信号值 [17:15]: 配置 awprot 信号值

5.11.4.6 dma_upaxi_arconfg (0x14)

域	位	读写	复位值	描述
dma_upaxi_arconfg	31:0	RW	0x0	用来配置上行 AXI 读地址通道的部分信号值 [3:0]: 配置 arcache 信号值 [7:4]: 配置 arregion 信号值 [10:8]: 配置 arsnoop 信号值 [12:11]: 配置 arbar 信号值 [14:13]: 配置 ardomain 信号值 [17:15]: 配置 arprot 信号值

5.11.4.7 dma_dwnaxi_awconfg (0x18)

域	位	读写	复位值	描述
dma_dwnaxi_awconfg	31:0	RW	0x0	用来配置下行 AXI 写地址通道的部分信号值 [3:0]: 配置 awcache 信号值 [7:4]: 配置 awregion 信号值 [10:8]: 配置 awsnoop 信号值 [12:11]: 配置 awbar 信号值 [14:13]: 配置 awdomain 信号值 [17:15]: 配置 awprot 信号值

5.11.4.8 dma_dwnaxi_arconfg (0x1C)

域	位	读写	复位值	描述
dma_dwnaxi_arconfg	31:0	RW	0x0	用来配置下行 AXI 读地址通道的部分信号值 [3:0]: 配置 arcache 信号值 [7:4]: 配置 arregion 信号值 [10:8]: 配置 arsnoop 信号值

域	位	读写	复位值	描述
				[12:11]: 配置 arbar 信号值 [14:13]: 配置 ardomain 信号值 [17:15]: 配置 arprot 信号值

5.11.4.9 dma_channel_bind (0x20)

域	位	读写	复位值	描述
reserved	31:29	R0	0x0	保留
dma_channel_bind	7:0	RW	0x0	状态标志位, 分别用来表示 0~7 通道是否绑定了外设。 1: 该通道已绑定外设 DMA 请求信号线 0: 该通道未绑定外设 DMA 请求信号线

5.11.4.10 dma_gcap (0x24)

域	位	读写	复位值	描述
dma_gcap	31:0	R0	0x0	只读寄存器, 表示当前 DMA 设计总共可利用的有效通道数目

5.11.4.11 dma_chal_cfg1 (0x28)

域	位	读写	复位值	描述
chal7_sel_en	31	RW	0x0	通道 7 的 dma 请求信号线源端选择是否有效标志, 只有为 1 的首才表示通道 7 当前 dma 请求选择有效。
chal7_sel	30:24	RW	0x0	通道 7 的 dma 请求信号线源端选择, 同一时刻, 只有 1 位有效, 实现 32 选 1 的功能。
chal6_sel_en	23	RW	0x0	通道 6 的 dma 请求信号线源端选择是否有效标志, 只有为 1 的首才表示通道 6 当前 dma 请求选择有效。
chal6_sel	22:16	RW	0x0	通道 6 的 dma 请求信号线源端选择, 同一时刻, 只有 1 位有效, 实现 32 选 1 的功能。
chal5_sel_en	15	RW	0x0	通道 5 的 dma 请求信号线源端选择是否有效标志, 只有为 1 的首才表示通道 5 当前 dma 请求选择有效。
chal5_sel	14:8	RW	0x0	通道 5 的 dma 请求信号线源端选择, 同一时刻, 只有 1 位有效, 实现 32 选 1 的功能。
chal4_sel_en	7	RW	0x0	通道 4 的 dma 请求信号线源端选择是否有效标志, 只有为 1 的首才表示通道 4 当前 dma 请求选择有效。
chal4_sel	6:0	RW	0x0	通道 4 的 dma 请求信号线源端选择, 同一时刻, 只有 1 位有效, 实现 32 选 1 的功能。

5.11.4.12 dma_chal_x_ddr_upaddr (0x40+0x40*x)

域	位	读写	复位值	描述
dma_chal_x_ddr_upaddr	31:0	RW	0x0	AXI 操作内存中的高 32bit 的地址信息, 可以向该地址写数据, 也可以从该地址读取数据, 具体形式根据软件配置来决定。

5.11.4.13 dma_chalx_ddr_lwadd (0x44+0x40*x)

域	位	读写	复位值	描述
dma_chalx_ddr_lwadd	31:0	RW	0x0	[31:7] AXI 操作内存中的低 32bit 的地址信息，可以向该地址写数据，也可以从该地址读取数据，具体形式根据软件配置来决定。 [6:0] 默认是 0，即地址是 128Byte 对齐的。

5.11.4.14 dma_chalx_dev_addr (0x48+0x40*x)

域	位	读写	复位值	描述
dma_chalx_dev_addr	31:0	RW	0x0	访问外设 FIFO 地址，是一个固定的，可以向该地址写数据，也可以从该地址读取数据，具体形式根据软件配置决定。

5.11.4.15 dma_chalx_ts (0x4C+0x40*x)

域	位	读写	复位值	描述
dma_chalx_ts	31:0	RW	0x0	需要传输数据的总的数目，以 Byte 作为基本单位，当该通道所有数据传输完成后，硬件主动停止该通道工作，并产生中断。 注意：设定数据的总数目要为 4Byte 的倍数。

5.11.4.16 dma_chalx_crt_upaddr (0x50+0x40*x)

域	位	读写	复位值	描述
dma_chalx_crt_upaddr	31:0	RW	0x0	当前对应内存中需要读写数据的高 32 位地址

5.11.4.17 dma_chalx_crt_lwaddr (0x54+0x40*x)

域	位	读写	复位值	描述
dma_chalx_crt_lwaddr	31:0	RW	0x0	当前对应内存中需要读写数据的低 32 位地址

5.11.4.18 dma_chalx_ctl (0x58+0x40*x)

域	位	读写	复位值	描述
reserved	31:3	RW	0x0	保留
chalx_mode	2	RW	0x0	通道 x 配置为接收外设的哪一种请求。 1: 接收外设 dma_rx_req 0: 接收外设 dma_tx_req 注意：正常工作中，软件不允许对该位进行操作，只有在通道非使能的情况下才允许操作。

域	位	读写	复位值	描述
chalx_srst	1	RW	0x0	通道 x 复位控制寄存器，有效时，该通道相关寄存器和 FIFO 均复位，1 表示有效。 软件若想复位该通道，必须先 disable 该通道。
chalx_en	0	RW	0x0	通道 x 使能控制信号。 1：通道使能，通道工作； 0：通道非使能，通道不工作，但寄存器和 FIFO 数值保持不变。 当软件想要暂停某个通道的时候，先写该位 1'b0，随后读取到该位是 1'b0 的时候，才能复位该通道。

5.11.4.19 dma_chalx_sts (0x5C+0x40*x)

域	位	读写	复位值	描述
reserved	31:2	RW	0x0	保留
fifo_empty	1	RW	0x0	通道 x 对应 FIFO 空状态信号
fifo_full	0	RW	0x0	通道 x 对应 FIFO 满状态信号

5.11.4.20 dma_chalx_timeout_cnt (0x60+0x40*x)

域	位	读写	复位值	描述
timeout_en	31	RW	0x0	超时机制使能位，只有当为 1 的时候才启动超时机制。
reserved	30:28	RW	0x0	保留
timeout_cnt	27:0	RW	0x7ae14	通道配置为接收外设读取请求时，若使能超时机制，本地计数器开始计数，若长时间内外设没有 dma 读请求，当计数值达到该阈值后，将产生超时标志信息。此时 dma 将本地已经缓存的数据发送到系统，并主动暂停该通道，产生相关中断。

5.12 硬件信号量（Semaphore）

5.12.1 简介

飞腾派集成了硬件信号量模块，支持 32 个硬件信号量。每个信号量包括 UNLOCK 和 LOCKED 两个状态，具体状态跳转如下图所示：

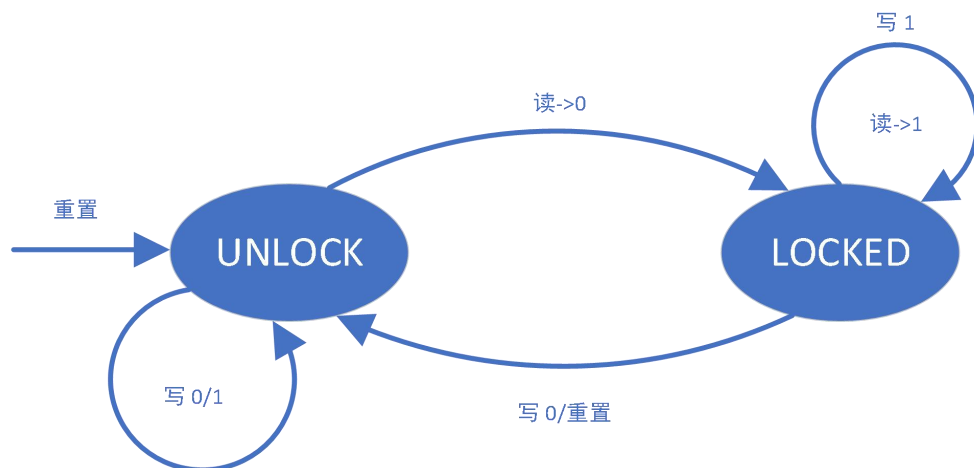


图 5-10 硬件信号量状态转换示意图

5.12.2 操作说明

1. 读锁定方式，发送读操作到对应信号量的读锁定寄存器(rlock_reg)。
2. 读取所有信号量的锁定和解锁状态，读取 state_reg。
3. 单独复位一个信号量，操作 indrst_reg 寄存器。
4. 复位所有信号量，操作 rst_reg 寄存器。

5.12.3 寄存器列表

表 5-39 Semaphore 寄存器基地址

名称	基地址
Semaphore	0x000_32B3_6000

表 5-40 Semaphore 寄存器列表

寄存器	偏移量	描述
Sema_rst_reg	0x000	写 1 复位所有信号量
Sema_indrst_reg	0x004	写信号量独热码为 1，复位对应信号量
Sema_state_reg	0x008	状态寄存器
Sema_rlockx_reg	0x010+x*4	信号量 x 读锁定寄存器

5.12.4 寄存器说明

5.12.4.1 Sema_rst_reg (0x000)

域	位	读写	复位值	描述
rstall	0	RW	0x0	写 1 复位所有信号量

5.12.4.2 Sema_indrst_reg (0x004)

域	位	读写	复位值	描述
indrst	31:0	RW	0x0	写信号量独热码为 1，复位对应信号量

5.12.4.3 Sema_state_reg (0x008)

域	位	读写	复位值	描述
Stat_lock31	31	RO	0x0	信号量 31 的状态 0 表示解锁，1 表示锁定
Stat_lock30	30	RO	0x0	信号量 30 的状态 0 表示解锁，1 表示锁定
Stat_lock29	29	RO	0x0	信号量 29 的状态 0 表示解锁，1 表示锁定
Stat_lock28	28	RO	0x0	信号量 28 的状态 0 表示解锁，1 表示锁定
Stat_lock27	27	RO	0x0	信号量 27 的状态 0 表示解锁，1 表示锁定
Stat_lock26	26	RO	0x0	信号量 26 的状态 0 表示解锁，1 表示锁定
Stat_lock25	25	RO	0x0	信号量 25 的状态 0 表示解锁，1 表示锁定
Stat_lock24	24	RO	0x0	信号量 24 的状态 0 表示解锁，1 表示锁定
Stat_lock23	23	RO	0x0	信号量 23 的状态 0 表示解锁，1 表示锁定
Stat_lock22	22	RO	0x0	信号量 22 的状态 0 表示解锁，1 表示锁定
Stat_lock21	21	RO	0x0	信号量 21 的状态 0 表示解锁，1 表示锁定
Stat_lock20	20	RO	0x0	信号量 20 的状态 0 表示解锁，1 表示锁定
Stat_lock19	19	RO	0x0	信号量 19 的状态 0 表示解锁，1 表示锁定
Stat_lock18	18	RO	0x0	信号量 18 的状态 0 表示解锁，1 表示锁定
Stat_lock17	17	RO	0x0	信号量 17 的状态 0 表示解锁，1 表示锁定
Stat_lock16	16	RO	0x0	信号量 16 的状态 0 表示解锁，1 表示锁定
Stat_lock15	15	RO	0x0	信号量 15 的状态 0 表示解锁，1 表示锁定
Stat_lock14	14	RO	0x0	信号量 14 的状态 0 表示解锁，1 表示锁定
Stat_lock13	13	RO	0x0	信号量 13 的状态

域	位	读写	复位值	描述
				0 表示解锁, 1 表示锁定
Stat_lock12	12	RO	0x0	信号量 12 的状态 0 表示解锁, 1 表示锁定
Stat_lock11	11	RO	0x0	信号量 11 的状态 0 表示解锁, 1 表示锁定
Stat_lock10	10	RO	0x0	信号量 10 的状态 0 表示解锁, 1 表示锁定
Stat_lock9	9	RO	0x0	信号量 9 的状态 0 表示解锁, 1 表示锁定
Stat_lock8	8	RO	0x0	信号量 8 的状态 0 表示解锁, 1 表示锁定
Stat_lock7	7	RO	0x0	信号量 7 的状态 0 表示解锁, 1 表示锁定
Stat_lock6	6	RO	0x0	信号量 6 的状态 0 表示解锁, 1 表示锁定
Stat_lock5	5	RO	0x0	信号量 5 的状态 0 表示解锁, 1 表示锁定
Stat_lock4	4	RO	0x0	信号量 4 的状态 0 表示解锁, 1 表示锁定
Stat_lock3	3	RO	0x0	信号量 3 的状态 0 表示解锁, 1 表示锁定
Stat_lock2	2	RO	0x0	信号量 2 的状态 0 表示解锁, 1 表示锁定
Stat_lock1	1	RO	0x0	信号量 1 的状态 0 表示解锁, 1 表示锁定
Stat_lock0	0	RO	0x0	信号量 0 的状态 0 表示解锁, 1 表示锁定

5.12.4.4 Sema_rlockx_reg (0x010 + x*4)

x 的取值范围为 0~31。

域	位	读写	复位值	描述
reserved	31:1	RW	0x0	保留
rlockx	0	RW	0x0	信号量 x 读锁定寄存器。 读返回 0: 信号量当前未被锁定, 读后被锁定。 读返回 1: 信号量当前已经被锁定, 锁定失败。 写 0: 信号量被解锁。 写 1: 无效。

5.13 看门狗(WDT)

看门狗定时器是一个计数器, 在发生软件问题和程序跑飞后使系统重新启动。

5.13.1 操作说明

1. 喂狗：写 WRR、WCS、WOR、WCV 寄存器。写 WRR、WCS、WOR 寄存器会直接将当前 $\text{sys_cnt} + \text{WOR}$ 寄存器储存的值更新到 WCV 寄存器中；写 WCV 寄存器需要先写高 32 位再写低 32 位的序列完成后，才会一起更新 WCV 寄存器。其中写 WRR 为喂狗操作。写 WCS 的最低位（其他位写操作无效）配置控制器使能位，同时写此寄存器也会有喂狗的效果。写 WOR 寄存器设置看门狗的计时值，如果无对此寄存器配置操作则默认为 $32'h30000000$ 。写寄存器 WCV 则可配置看门狗的比较值，如果不配置看门狗的比较值默认 $\text{sys_cnt} + \text{WOR}$ 。首次配置时可以依次对四个寄存器进行配置，若后续计时到正常喂狗，若不改计数时间，则只需写 WRR 寄存器喂狗即可。
2. 超时： sys_cnt 的计数值大于当前 WCV 寄存器存储的比较值。
3. 复位：一般是拉低 hresetn 信号。
4. 一般操作应遵循：热保护复位和常规复位完成后，先写 WOR 寄存器告知看门狗计数超时值。然后再写 WCS 寄存器最低位为 $1'b1$ ，表示使能看门狗，如果使能之前未写 WOR 寄存器，那么计数超时值默认为 $0x30000000$ ，即 1s。
5. 正常喂狗可以写 WRR 寄存器，如果需要改变计数超时值，可以直接写 WOR 寄存器。如果 WOR 的 32 位寄存器不够计数需求，可以直接写 64 位的 WCV 寄存器。
6. 可以随时读 WCS、WOR、WCV 寄存器来获取所关心的信息。常规复位时，这些寄存器不会复位，以便查错（但是会复位 WCS[0]，即禁用看门狗）。
7. 当 ws0（一次超时）和 ws1（二次超时）同时拉高时，WCV 寄存器不会更新，即连续 2 次超时后（一次超时控制器报中断，二次超时控制器发起复位。一次超时上报中断后，需要进行喂狗，若在计数时间内（WOR）无喂狗操作则会二次超时），仍能读取当前的 WCV 值。

5.13.2 寄存器列表

表 5-41 WDT 寄存器基地址

名称	基地址
WDT0	0x000_2804_0000
WDT1	0x000_2804_2000

表 5-42 WDT 寄存器列表

寄存器	偏移	描述
WDT_WRR	0x0000	看门狗更新寄存器
WDT_W_IIDR	0x0FCC	看门狗接口身份识别寄存器
WDT_WCS	0x1000	看门狗控制和状态寄存器

寄存器	偏移	描述
WDT_WOR	0x1008	看门狗清除寄存器
WDT_WCVL	0x1010	看门狗比较值低 32 位寄存器
WDT_WCVH	0x1014	看门狗比较值高 32 位寄存器

5.13.3 寄存器说明

5.13.3.1 WDT_WRR (0x0000)

域	位	读写	复位值	描述
WDT_WRR	31:0	RW	0x0	Watchdog 更新寄存器。写操作会重新开始看门狗计数, 读返回 0。

5.13.3.2 WDT_W_I IDR (0x0FCC)

域	位	读写	复位值	描述
reserved	31:20	R0	0x0	保留
Watchdog version	19:16	R0	0x0	Watchdog 版本号
reserved	15:12	R0	0x0	保留
JEP 106 continuation code	11:8	R0	0x8	JEP 106 扩展编码
Fix zero	7	R0	0x0	固定为 0
JEP 106 identify code	6:0	R0	0x19	JEP 106 身份编码

5.13.3.3 WDT_WCS (0x1000)

域	位	读写	复位值	描述
reserved	31:3	R0	0x0	保留
Ws1	2	R0	0x0	二次超时, 读返回当前 ws1 的值
Ws0	1	R0	0x0	一次超时, 读返回当前 ws0 的值
Wdt_en	0	RW	0x0	Watchdog 使能信号, 高有效, 常规复位和热保护都会清 0。

5.13.3.4 WDT_WOR (0x1008)

域	位	读写	复位值	描述
WDT_WOR	31:0	RW	0x3000000	Watchdog 清除寄存器

5.13.3.5 WDT_WCVL (0x1010)

WCVL、WCVH 寄存器存储的看门狗计数的比较值。计数超时值=比较值—当前 sys_cnt。
注意：写操作有严格的顺序，必须先写高 32 位，再写低 32 位。

域	位	读写	复位值	描述
WDT_WCVL	31:0	RW	0x0	Watchdog 比较值的低 32 位

5.13.3.6 WDT_WCVH (0x1014)

域	位	读写	复位值	描述
WDT_WCVH	31:0	RW	0x0	Watchdog 比较值的高 32 位

5.14 QSPI 控制器

QSPI 控制器控制主机与 nor flash 的交互，主机访问 flash 的模式包括内存映射模式（将外部 flash 的空间直接映射到控制器内部，此时系统将其视作内部寄存器）、间接模式（使用控制器的寄存器进行全部操作）、状态轮询模式（周期性读取 flash 内部寄存器）。主机向 flash 发送命令（擦除/编程/读取/flash 内部寄存器读写）时，QSPI 控制器将主机发送的命令转变为 qspi flash 支持的接口信号发送至 flash 以达到主机访问 flash 的目的。

5.14.1 操作说明

5.14.1.1 初始化操作

1. 设置外接 FLASH 的容量

默认外接 FLASH 的容量为 16MB，若想外接其他容量的 FLASH，以外接一个 32MB（256Mbit）的 flash 为例，应向 FLASH 容量设置寄存器 (base_addr+0x000) 写 0x0000_0003。

2. 设置频率

pclk 是 apb 总线时钟，也是 QSPI 控制器主时钟，固定为 300MHz。sck 是控制器输出的 qspi 总线时钟，由控制器从 pclk 时钟分频产生。读操作默认是 64 分频，其他操作默认是 128 分频，若要配置为其他分频系数，以二分频为例，需向分频设置寄存器 (base_addr+0x030) 写 0x0000_0101。

5.14.1.2 擦除操作

全芯片擦除需要配置命令端口寄存器和低位数据端口寄存器；其他擦除需要配置命令端口寄存器、地址端口寄存器和低位数据端口寄存器。

以对 flash0 进行 64KB 块擦除命令 (D8) 操作为例：

1. 向命令端口寄存器 (base_addr+0x010) 写入 0x06000000；
2. 向低位数据寄存器 (base_addr+0x01c) 进行写操作，写任意值均可；
3. 向命令端口寄存器 (base_addr+0x010) 写入 0xD8408000；
4. 向地址端口寄存器 (base_addr+0x014) 写入擦除块地址如 0x00010000（擦

- 64kB~128KB 的块，0x00010000 是该块的起始地址）；
5. 向低位数据寄存器 (base_addr+0x01c) 进行写操作，写任意值均可；
 6. 完成。

5.14.1.3 编程操作

256 字节页编程：

1. 向命令端口寄存器 (base_addr+0x010) 写入 0x06000000；
2. 向低位数据寄存器 (base_addr+0x01c) 进行写操作，写任意值均可；
3. 向直接地址写配置寄存器 (base_addr+0x008) 写入 0x02000208；
4. 对数据空间连续写 64 次 4 字节的数据；
5. 对写缓冲 flush 寄存器 (base_addr+0x00c) 写入 0x0000_0001；
6. 完成一次 256 字节页编程。

5.14.1.4 QPI 模式直接地址访问

1. 向直接地址写配置寄存器 (base_addr+0x008) 写入 0x020002c6；
2. 进入 QPI 模式，向命令端口寄存器 (base_addr+0x010) 写入 0x38000006；
3. 向低位数据寄存器 (base_addr+0x01c) 进行写操作，写任意值均可；
4. 打开写使能 wren，向命令端口寄存器 (base_addr+0x010) 写入 0x06060006；
5. 向低位数据寄存器 (base_addr+0x01c) 进行写操作，写任意值均可；
6. 进行直接地址写；
7. 写完后退出 QPI，向命令端口寄存器 (base_addr+0x010) 写入 0xF5000006。

5.14.1.5 读操作

对数据空间直接进行读操作，每次读 4 字节数据，可配置直接访问读配置寄存器提升性能，如连续读取数据块，可开启数据读缓冲。

EC 和 EB 命令为例：

MODE 字节设置：向 (base_addr+0x02c) 地址写 0x0000_F0A0。

四线四字节地址编程：向 (base_addr+0x004) 地址写 0xEC4F_4070。

四线三字节地址编程：向 (base_addr+0x004) 地址写 0xEB47_4070。

6C 和 6B 命令为例：

四线四字节地址编程：向 (base_addr+0x004) 地址写 0x6C2C0079。

四线三字节地址编程：向 (base_addr+0x004) 地址写 0x6B240079。

5.14.1.6 读状态寄存器

1. 向命令端口寄存器 (base_addr+0x010) 写入 0x05002000。
2. 向低位数据寄存器 (base_addr+0x01c) 进行读操作。

5.14.1.7 读 ID 操作

一般来说读取芯片 ID 指令有 5Ah、90h、9Fh、98h、ABh、15h 等，不同芯片支持某一种或几种指令，不同指令返回内容格式有所差异，这不仅与指令有关还与厂商和芯片型号有所关系。为了缩小差异，建议选择 9Fh 指令，其绝大部分芯片都支持且前三字节内容格式都相同，能准确知道厂商、器件接口类型和容量。对 flash0 以 9F 命令读四字节 ID 为例：

1. 向命令端口寄存器 (base_addr+0x010) 写入 0x9F002018；
2. 向低位数据寄存器 (base_addr+0x01c) 进行读操作。

5.14.1.8 错误信号

表 5-43 QSPI 错误信号

信号名	描述
pslverr	APB 访问错误
qspi_ras_addr_err	访问地址不是实际有效地址，此信号单拍高有效
qspi_ras_pstrb_err	写访问请求 pstrb 信号不是控制器支持的组合，此信号单拍高有效
qspi_ras_err_addr	RAS 中断报错地址，qspi_ras_addr_err 或 qspi_ras_pstrb_err 为高有效

5.14.2 寄存器列表

表 5-44 QSPI 寄存器基地址

名称	基地址
QSPI	0x000_2800_8000

表 5-45 QSPI 寄存器列表

寄存器	偏移	描述
flash_capacity	0x000	FLASH 容量设置寄存器
rd_cfg	0x004	直接地址访问读配置寄存器
wr_cfg	0x008	直接地址访问写配置寄存器
flush_reg	0x00C	写缓冲 flush 寄存器
cmd_port	0x010	命令端口寄存器
addr_port	0x014	地址端口寄存器
hd_port	0x018	高位数据端口寄存器
ld_port	0x01C	低位数据端口寄存器
cs_set	0x020	片选设置寄存器

寄存器	偏移	描述
wip_rd	0x024	WIP 读取设置寄存器
wp_reg	0x028	写保护设置寄存器
mode_reg	0x02C	XIP 模式设置寄存器
cycle_reg	0x030	分频系数设置寄存器

0~0x1FFF_FFFF 为 QSPI 直接地址可访问数据空间。

5.14.3 寄存器说明

5.14.3.1 flash_capacity (0x000)

设置所连接的 Flash 容量和数量，目前 QSPI Flash 芯片最大支持 2Gb（256MB）的容量，那么单个 Flash 最大支持地址也就是 32'h0fff_ffff 大小，最大支持连接四个相同容量的 Flash。

域	位	读写	复位值	描述
reserved	31:5	RW	0x0	保留
flash_num	4:3	RW	0x0	支持连接 flash 00: 支持 1 个 flash 01: 支持 2 个 flash 10: 支持 3 个 flash 11: 支持 4 个 flash
flash_capacity	2:0	RW	0x0	支持容量大小 000: 4MB 001: 8MB 010: 16MB 011: 32MB 100: 64MB 101: 128MB 110: 256MB 111: 4MB

5.14.3.2 rd_cfg (0x004)

配置直接地址访问的读命令，每次直接地址访问时控制器将按此寄存器配置参数向 Flash 发送读命令。

域	位	读写	复位值	描述
rd_cmd	31:24	RW	0x0	读命令的指令字节，初始值时会映射到 read (03h) 指令字节。
rd_through	23	RW	0x0	1 表示命令发送不需要等待 Flash 状态寄存器的 WIP 位置为 0。
rd_transfer	22:20	RW	0x0	读命令传输协议 000: 1-1-1

域	位	读写	复位值	描述
				001: 1-1-2 010: 1-1-4 011: 1-2-2 100: 1-4-4 101: 2-2-2 110: 4-4-4 111: 1-1-1
rd_addr_sel	19	RW	0x0	0: 3byte 地址 1: 4byte 地址
rd_latency	18	RW	0x0	1 表示读数据有延迟
mode_byte	17	RW	0x0	1 表示有命令修饰符
cmd_sign	16:9	RW	0x0	当 mode_byte=1 该域有效, 存储命令修饰符。
dummy	8:4	RW	0x0	当 r_latency=1 表示读数据有延迟, 该域有效, 5'h00~5'h1f 表示延时 1~32 个 cycle。
d_buffer	3	RW	0x0	0: 每次读请求直接读数据 1: 每次读请求从缓冲器读数据
rd_sck_sel	2:0	RW	0x0	000: sck 为 pclk 的 128 分频 001: sck 为 pclk 的 2 分频 010: sck 为 pclk 的 4 分频 011: sck 为 pclk 的 8 分频 100: sck 为 pclk 的 16 分频 101: sck 为 pclk 的 32 分频 110: sck 为 pclk 的 64 分频 111: sck 为 pclk 的 128 分频

5.14.3.3 wr_cfg (0x008)

配置直接地址访问的编程命令, 每次直接地址访问时控制器将按此寄存器配置参数向 Flash 发送写命令。

域	位	读写	复位值	描述
wr_cmd	31:24	RW	0x0	写命令的指令字节, 初始值时会映射到 PP (02h) 指令字节
reserved	23:10	RW	0x0	保留
wr_wait	9	RW	0x0	1 表示此命令在 Flash 需要一定的执行时间
wr_through	8	RW	0x0	1 表示命令发送不需要等待 Flash 状态寄存器的 WIP 位置为 0
wr_transfer	7:5	RW	0x0	编程命令传输协议 000: 1-1-1 001: 1-1-2 010: 1-1-4 011: 1-2-2 100: 1-4-4 101: 2-2-2 110: 4-4-4

域	位	读写	复位值	描述
				111: 1-1-1
wr_addr_sel	4	RW	0x0	0: 3byte 地址 1: 4byte 地址
wr_mode	3	RW	0x0	0: 每次写请求直接发编程命令 1: 写数据先放入缓冲, 多次写合并编程
wr_sck_sel	2:0	RW	0x0	000: sck 为 pclk 的 128 分频 001: sck 为 pclk 的 2 分频 010: sck 为 pclk 的 4 分频 011: sck 为 pclk 的 8 分频 100: sck 为 pclk 的 16 分频 101: sck 为 pclk 的 32 分频 110: sck 为 pclk 的 64 分频 111: sck 为 pclk 的 128 分频

5.14.3.4 flush_reg (0x00C)

写 1 将把写缓冲中的数据写入 Flash, 配合地址访问写配置寄存器完成页编程操作。

域	位	读写	复位值	描述
flush	0	WO	0x0	写 1 将产生 flush 操作

5.14.3.5 cmd_port (0x010)

通过寄存器端口访问时的命令, 命令按命令端口寄存器配置参数向 Flash 发送。

域	位	读写	复位值	描述
cmd	31:24	RW	0x0	指令字节
reserved	23	RW	0x0	保留
wait	22	RW	0x0	1 表示此命令在 Flash 需要一定的执行时间
through	21	RW	0x0	1 表示命令发送不需要等待 Flash 状态寄存器的 WIP 位置为 0
cs0_3	20:19	RW	0x0	选择需要发命令的芯片
transfer	18:16	RW	0x0	命令传输协议 000: 1-1-1 001: 1-1-2 010: 1-1-4 011: 1-2-2 100: 1-4-4 101: 2-2-2 110: 4-4-4 111: 1-1-1
cmd_addr	15	RW	0x0	1 表示有地址传输
latency	14	RW	0x0	1 表示读数据有延迟
data_transfer	13	RW	0x0	1 表示有数据传输
addr_sel	12	RW	0x0	0: 3byte 地址

域	位	读写	复位值	描述
				1: 4byte 地址
dummy	11:7	RW	0x0	当 latency=1 表示读数据有延迟, 该域有效, 5'h00~5'h1f 表示延迟 1~32 个 cycle。
p_buffer	6	RW	0x0	0: 命令端口方式不使用缓冲器, 使用数据端口寄存器; 1: 使用缓冲器, 当命令是带数据的读命令时从缓冲器中读数据。
rw_num	5:3	RW	0x0	读写字节数目, data_transfer=1 且 p_buffer=0 有效, 3'h0~3'h7 分别表示从数据端口寄存器读或写数据的 1~8 字节。
sck_sel	2:0	RW	0x0	000: sck 为 pclk 的 128 分频 001: sck 为 pclk 的 2 分频 010: sck 为 pclk 的 4 分频 011: sck 为 pclk 的 8 分频 100: sck 为 pclk 的 16 分频 101: sck 为 pclk 的 32 分频 110: sck 为 pclk 的 64 分频 111: sck 为 pclk 的 128 分频

5.14.3.6 addr_port (0x014)

通过寄存器端口访问时设置的地址。

域	位	读写	复位值	描述
addr	31:0	RW	0x0	地址

5.14.3.7 hd_port (0x018)

通过寄存器端口访问时的高 4 字节数据。

域	位	读写	复位值	描述
data	31:0	RW	0x0	最大支持 4 字节有效数据

5.14.3.8 ld_port (0x01C)

通过寄存器端口访问时的低 4 字节数据。读写该寄存器将触发控制器对 Flash 发送按命令端口寄存器发送的指令。如果命令需要数据, 则按高位数据寄存器、低位数据寄存器的顺序访问。命令端口寄存器设置使用缓冲器时读该寄存器将触发对缓冲器取数据。

域	位	读写	复位值	描述
data	31:0	RW	0x0	与高位数据端口寄存器一起最大支持 8 字节有效数据

5.14.3.9 cs_set (0x020)

设置 csn 有效建立时间、有效保持时间和高电平间隔时间, 参考具体芯片数据手册。

域	位	读写	复位值	描述
cs_hold	31:24	RW	0x5	有效保持时间 计算公式: $time = pclk_cycle * (cs_hold + 1) + sck_cycle / 2$
cs_setup	23:16	RW	0x5	有效建立时间 计算公式: $time = pclk_cycle * (cs_setup + 1) + sck_cycle / 2$
cs_delay	15:0	RW	0x28	高电平间隔时间, 命令结束后的延迟周期, 即指令操作周期间隔 计算公式: $time = pclk_cycle * (cs_delay - cs_hold + 4) \quad (wp_en = 0)$ $time = pclk_cycle * (\max\{cs_delay, wp_hold\} + wp_setup - cs_hold + 4) \quad (wp_en = 1)$

5.14.3.10 wip_rd (0x024)

设置查询 Flash 状态寄存器的命令。

域	位	读写	复位值	描述
w_cmd	31:24	RW	0x5	读状态寄存器指令字节
reserved	23:5	RW	0x0	保留
w_transfer	4:3	RW	0x0	状态寄存器读命令传输协议 00: 1-1-1 01: 2-2-2 1x: 4-4-4
w_sck_sel	2:0	RW	0x0	000: sck 为 pclk 的 128 分频 001: sck 为 pclk 的 2 分频 010: sck 为 pclk 的 4 分频 011: sck 为 pclk 的 8 分频 100: sck 为 pclk 的 16 分频 101: sck 为 pclk 的 32 分频 110: sck 为 pclk 的 64 分频 111: sck 为 pclk 的 128 分频

5.14.3.11 wp_reg (0x028)

1-1-1 命令协议下控制 wp 输出。

域	位	读写	复位值	描述
reserved	31:18	RW	0x0	保留
wp_en	17	RW	0x0	wp 使能信号, 1 使能硬件写保护
wp	16	RW	0x0	写 0: wp/io2 输出 0 写 1: wp/io2 输出 1
wp_hold	15:8	RW	0x50	wp 保持时间 计算公式:

域	位	读写	复位值	描述
				$time = pclk_cycle * (wp_hold + 1)$
wp_setup	7:0	RW	0xF	wp 建立时间 计算公式: $time = pclk_cycle * (wp_setup + 1)$

5.14.3.12 mode_reg (0x02C)

设置 Flash 模式位的值，一般而言同款甚至同一厂商 Flash 的 XIP 模式位的值是固定的。

域	位	读写	复位值	描述
reserved	31:16	RW	0x0	保留
mode_valid	15:8	RW	0xFF	Flash 的 XIP 模式位的有效位，如 Flash 的 XIP 模式位为 Axh，其中低四位为 x 忽略，此时设置 mode_valid=8'hf0。
mode	7:0	RW	0xFF	寄存相关 Flash 模式位的值

5.14.3.13 cycle_reg (0x030)

附加的 SCK 分频系数寄存器，能够实现更细粒度的 2 到 256 整数分频，设置奇数分频系数输出非 50% 占空比 SCK。

域	位	读写	复位值	描述
reserved	31:9	RW	0x0	保留
cycle_sel	8	RW	0x0	0: 使用 rd_cfg/wr_cfg/cmd_port/ wip_reg 设置的分频系数。 1: 当 cycle_sel 不为 0 时使用 cycle_plus 的分频系数，否则使用 rd_cfg/wr_cfg/ cmd_port/ wip_reg 设置的分频系数。
cycle_plus	7:0	RW	0x0	设置分频系数 1: 2 分频 2: 3 分频 ... 255: 256 分频

5.15 SPI 控制器

SPI 是一种高速、全双工、同步的通信总线。飞腾派集成的 SPI Master 控制器用于连接各类 SPI 外设。

5.15.1 操作说明

初始化操作说明：

1. 向使能寄存器 SSIENR (0x08) 写 0;
2. 写控制寄存器 CTRLR0 (0x00) 为 0x74C7;
3. 配置波特率, 写寄存器 BAUDR (0x14) 为 0xF, 即 SCKDV, 计算公式:

$$\frac{F_{ssi_clk}}{SCKDV} = F_{sclk_out}$$

F_{ssi_clk} 为 SPI 控制器的主时钟, 固定为 50MHz;

F_{sclk_out} 为分频后 SCK 总线的时钟频率。

4. 分别写发送和接收 FIFO 域值, TXFTLR (0x018), RXFTLR (0x1C) 为 0x2;
5. 选择哪个从机接收数据, 通过写寄存器 (SER) 0x10 对应位为 1, 共 4 位, 可挂 4 个从机;
6. 使能 SPI 通过向寄存器 SSIENR (0x08) 写 1。

5.15.2 寄存器列表

表 5-46 SPI 寄存器基地址

名称	基地址
SPIM0	0x000_2803_A000
SPIM1	0x000_2803_B000
SPIM2	0x000_2803_C000
SPIM3	0x000_2803_D000

表 5-47 SPI 寄存器列表

寄存器	偏移	描述
CTRLR0	0x00	控制寄存器 0
CTRLR1	0x04	控制寄存器 1
SSIENR	0x08	SPI 使能寄存器
MWCR	0x0C	Microwire 控制
SER	0x10	从机使能寄存器
BAUDR	0x14	波特率选择寄存器
TXFTLR	0x18	发送 FIFO 阈值寄存器
RXFTLR	0x1C	接收 FIFO 阈值寄存器
TXFLR	0x20	发送 FIFO 等级寄存器
RXFLR	0x24	接收 FIFO 等级寄存器
SR	0x28	状态寄存器
IMR	0x2C	中断屏蔽寄存器
ISR	0x30	中断状态寄存器
RISR	0x34	原始中断状态寄存器
TXOICR	0x38	清除发送 FIFO 溢出中断寄存器
RXOICR	0x3C	清除接收 FIFO 溢出中断寄存器
RXUICR	0x40	清除接收 FIFO 下溢中断寄存器
MSTICR	0x44	清除多主机争用中断寄存器
ICR	0x48	中断清除寄存器

寄存器	偏移	描述
DMACR	0x4C	DMA 控制寄存器
DMATDLR	0x50	DMA 发送数据等级寄存器
DMARDLR	0x54	DMA 接收数据等级寄存器
IDR	0x58	识别码
DR	0x60-0xEC	数据寄存器
RX_SAMPLE_DLY	0xF0	接收数据延时寄存器
SPI_CS_REG	0x100	片选控制寄存器

5.15.3 寄存器说明

5.15.3.1 CTRLR0 (0x00)

域	位	读写	复位值	描述
reserved	31:16	RW	0x0	保留
CFS	15:12	RW	0x0	数据大小控制位，用于 Microwire 模式。
SRL	11	RW	0x0	移位寄存器回环。仅用于测试目的。将发送寄存器的输出接入接收寄存器的输入。 0：寄存器正常模式 1：寄存器测试模式 当 SPI 配置为环回模式中的从机时，ss_in_n 和 ssi_clk 必需由外部设备提供。在此模式下，从机无法生成这些信号因为没有可循环的对象。
SLV_OE	10	RW	0x0	从机发送逻辑使能位 0：从机发送使能 1：从机发送禁用
TMOD	9:8	RW	0x0	传输模式控制位，仅指示接收或传输数据是否有效，只有当 spi 配置为主设备时，此传输模式才有效。 00：发送和接收模式 01：仅发送模式 10：仅接收模式 11：读 EEPROM
SCPOL	7	RW	0x0	串行时钟极性。设置为 Motorola SPI 时有效。 0：serial clock 为低时不活跃 1：serial clock 为高时不活跃
SCPH	6	RW	0x0	串行时钟相位。设置为 Motorola SPI 时有效。 0：串行时钟在第一个数据位中间切换 1：串行时钟在第一个数据位开始切换
FRF	5:4	RW	0x0	保留
DFS	3:0	RW	0x7	选择数据长度。当数据大小小于 16 位时，接收数据由接收逻辑自动右对齐。

5.15.3.2 CTRLR1 (0x04)

域	位	读写	复位值	描述
reserved	31:16	RW	0x0	保留
NDF	15:0	RW	0x0	TMOD=10 或 TMOD=11 该字段设置为 SPI 连续接收的数据量。接收数据等于这个寄存器值加 1，可以连续传输接收多达 64 KB 的数据。

5.15.3.3 SSIENR (0x08)

域	位	读写	复位值	描述
reserved	31:1	RW	0x0	保留
SSI_EN	0	RW	0x0	1: 启用 SPI 操作 0: 禁用 SPI 操作

5.15.3.4 MWCR (0x0C)

域	位	读写	复位值	描述
reserved	31:3	RW	0x0	保留
MHS	2	RW	0x0	Microwire 握手，仅当配置为串行主机设备时有效。用于启用和禁用 Microwire 协议。在启用之前清除 SR 寄存器中 BUSY 状态，在最后一个数据/控制位转移之后从目标从设备检查就绪状态。 0: handshaking interface 禁用 1: handshaking interface 使能
MDD	1	RW	0x0	Microwire 控制位。指定使用 Microwire 串行协议时数据字符的方向。 0: 从外部串行设备接收数据 1: 数据发送到外部串行设备
MWMOD	0	RW	0x0	Microwire 传输模式，定义 Microwire 传输是连续的还是非连续的。使用连续模式时，只需要用一个控制字就可以发送或接收数据字块。当使用非连续模式时，必需有一个控制字来控制每一个发送或接收的数据字。 0: 用非连续传输 1: 传连续传输

5.15.3.5 SER (0x10)

域	位	读写	复位值	描述
reserved	31:4	RW	0x0	保留
SER	3:0	RW	0x0	从机选择信号启动标志。该寄存器中的每一个位都对应来自 SPI 主机的从选信号(ss_x_n)，当此寄存器中的某个位被置为 1 时，串行口传输开始时，从主机上相对应的从选行被激活。

域	位	读写	复位值	描述
				1: 选择 0: 不选择

5.15.3.6 BAUDR (0x14)

域	位	读写	复位值	描述
reserved	31:16	RW	0x0	保留
SCKDV	15:0	RW	0x0	SSI 时钟除法器。 SCKDV 为 2 ~ 65534 之间的任何偶数值。 例如: $F_{ssi_clk} = 3.6864\text{MHz}$, $SCKDV = 2$ $F_{sclk_out} = 3.6864/2 = 1.8432\text{MHz}$

5.15.3.7 TXFTLR (0x18)

域	位	读写	复位值	描述
reserved	31:3	RW	0x0	保留
TFT	2:0	RW	0x0	发送 FIFO 阈值。控制发送 FIFO 触发中断的阈值, 取值范围为 0~7, FIFO 深度固定为 8。

5.15.3.8 RXFTLR (0x1C)

域	位	读写	复位值	描述
reserved	31:3	RW	0x0	保留
RFT	2:0	RW	0x0	接收 FIFO 阈值。控制接收 FIFO 触发中断的阈值, 取值范围为 0~7, FIFO 深度固定为 8。

5.15.3.9 TXFLR (0x20)

域	位	读写	复位值	描述
reserved	31:4	R0	0x0	保留
TXTFL	3:0	R0	0x0	发送 FIFO 等级, 包含传输 FIFO 中的有效数据项的数目

5.15.3.10 RXFLR (0x24)

域	位	读写	复位值	描述
reserved	31:4	R0	0x0	保留
RXTFL	3:0	R0	0x0	接收 FIFO 等级, 包含传输 FIFO 中的有效数据项的数目

5.15.3.11 SR (0x28)

域	位	读写	复位值	描述
reserved	31:7	R0	0x0	保留

域	位	读写	复位值	描述
DCOL	6	R0	0x0	传输数据冲突错误。仅当配置 SPI 为主机时才相关。该位通知处理器最后一次传输在完成前已经停止，读时这个位被清除。 0: 没有错误 1: 传输数据冲突错误
TXE	5	R0	0x0	传输错误。如果传输开始时传输 FIFO 为空则设置该位。只有当 SPI 设置为从设备时，才设置该位。读取时将清除此位。 0: 没有错误 1: 传输错误
RFF	4	R0	0x0	接收 FIFO 满。当接收 FIFO 完全被填满时置 1, 当接收 FIFO 包含一个或多个条目是被清除。 0: 接收 FIFO 不满 1: 接收 FIFO 满
RFNE	3	R0	0x0	接收 FIFO 不为空。当接收 FIFO 包含一个或多个条目设置时，当接收 FIFO 为空时清除。这个位可以被软件轮询已完全清空接收 FIFO 的数据。 0: 接收 FIFO 为空 1: 接收 FIFO 不空
TFE	2	R0	0x0	传送 FIFO 为空。发送 FIFO 完全为空时，设置该位。当传输 FIFO 包含一个或多个有效值时将清除该位，位不请求中断。 0: 传输 FIFO 不空 1: 传输 FIFO 为空
TFNF	1	R0	0x0	发送 FIFO 不满时。当传输 FIFO 包含一个或多个有空位时设置，已满时清除。 0: 发送 FIFO 满 1: 发送 FIFO 不满
BUSY	0	R0	0x0	SPI 总线繁忙标志位。当设置时，指示正在进行串行传输；如果以清除，则指示 SPI 为空闲。 0: SPI 空闲 1: SPI 忙

5.15.3.12 IMR (0x2C)

域	位	读写	复位值	描述
reserved	31:6	RW	0x0	保留
MSTIM	5	RW	0x1	多主机抢占中断屏蔽位，如果将 SPI 配置成串行设备，则不存在此位字段。 0: ssi_mst_intr 中断被屏蔽 1: ssi_mst_intr 中断没被屏蔽
RXFIM	4	RW	0x1	接收 FIFO 满中断屏蔽 0: ssi_mst_intr 中断被屏蔽 1: ssi_mst_intr 中断没被屏蔽

域	位	读写	复位值	描述
RXOIM	3	RW	0x1	接收 FIFO 上溢中断屏蔽 0: ssi_mst_intr 中断被屏蔽 1: ssi_mst_intr 中断没有屏蔽
RXUIM	2	RW	0x1	接收 FIFO 下溢中断屏蔽 0: ssi_mst_intr 中断被屏蔽 1: ssi_mst_intr 中断没被屏蔽
TXOIM	1	RW	0x1	发送 FIFO 上溢中断屏蔽 0: ssi_mst_intr 中断被屏蔽 1: ssi_mst_intr 中断没被屏蔽
TXEIM	0	RW	0x1	发送 FIFO 为空中断屏蔽 0: ssi_mst_intr 中断被屏蔽 1: ssi_mst_intr 中断没被屏蔽

5.15.3.13 ISR (0x30)

域	位	读写	复位值	描述
reserved	31:6	R0	0x0	保留
MSTIS	5	R0	0x0	多主机竞争中断状态，如果将 SPI 配置成串行从设备，则不存在此字段。 0: 屏蔽 ssi_mst_intr 后中断不活动 1: 屏蔽 ssi_mst_intr 后中断活动
RXFIS	4	R0	0x0	接收 FIFO 满中断状态 0: 屏蔽 ssi_mst_intr 后中断不活动 1: 屏蔽 ssi_mst_intr 后中断活动
RXOIS	3	R0	0x0	接收 FIFO 上溢中断状态 0: 屏蔽 ssi_mst_intr 后中断不活动 1: 屏蔽 ssi_mst_intr 后中断活动
RXUIS	2	R0	0x0	接收 FIFO 下溢中断状态 0: 屏蔽 ssi_mst_intr 后中断不活动 1: 屏蔽 ssi_mst_intr 后中断活动
TXOIS	1	R0	0x0	发送 FIFO 上溢中断状态 0: 屏蔽 ssi_mst_intr 后中断不活动 1: 屏蔽 ssi_mst_intr 后中断活动
TXEIS	0	R0	0x0	发送 FIFO 空中断状态 0: 屏蔽 ssi_txe_intr 中断后不活动 1: 屏蔽 ssi_txe_intr 中断后活动

5.15.3.14 RISR (0x34)

域	位	读写	复位值	描述
reserved	31:6	R0	0x0	保留
MSTIR	5	R0	0x0	多主机冲突生成中断状态。如果将 SPI 配置成串行从设备，则不存在此字段。 0: 优先屏蔽 ssi_mst_intr 后中断不活动

域	位	读写	复位值	描述
				1: 优先屏蔽 ssi_mst_intr 后中断活动
RXFIR	4	R0	0x0	接收 FIFO 满生成中断状态 0: 优先屏蔽 ssi_mst_intr 后中断不活动 1: 优先屏蔽 ssi_mst_intr 后中断活动
RXOIR	3	R0	0x0	接收 FIFO 上溢生成中断状态 0: 优先屏蔽 ssi_mst_intr 后中断不活动 1: 优先屏蔽 ssi_mst_intr 后中断活动
RXUIR	2	R0	0x0	接收 FIFO 下溢生成中断状态 0: 优先屏蔽 ssi_mst_intr 后中断不活动 1: 优先屏蔽 ssi_mst_intr 后中断活动
TXOIR	1	R0	0x0	传输 FIFO 上溢生成中断状态 0: 优先屏蔽 ssi_mst_intr 后中断不活动 1: 优先屏蔽 ssi_mst_intr 后中断活动
TXEIR	0	R0	0x0	传输 FIFO 空生成中断状态 0: 优先屏蔽 ssi_mst_intr 后中断不活动 1: 优先屏蔽 ssi_mst_intr 后中断活动

5.15.3.15 TXOICR (0x38)

域	位	读写	复位值	描述
reserved	31:1	R0	0x0	保留
TXOICR	0	R0	0x0	清除发送 FIFO 溢出中断。该位反映了中断的状态。从寄存器中读取将清除 ssi_txo_intr 中断。写入无效。

5.15.3.16 RXOICR (0x3C)

域	位	读写	复位值	描述
reserved	31:1	R0	0x0	保留
RXOICR	0	R0	0x0	清除接收 FIFO 溢出中断。该位反映了中断的状态。从寄存器中读取将清除 ssi_rxo_intr 中断，写入无效。

5.15.3.17 RXUICR (0x40)

域	位	读写	复位值	描述
reserved	31:1	R0	0x0	保留
RXUICR	0	R0	0x0	清除接收 FIFO 下溢中断。该位反映了中断的状态，从寄存器中读取将清除 ssi_rxu_intr 中断，写入无效。

5.15.3.18 MSTICR (0x44)

域	位	读写	复位值	描述
reserved	31:1	R0	0x0	保留
MSTICR	0	R0	0x0	清除多主争用中断，register 反映了中断的状态，从寄存器中读取将清除 ssi_mst_intr 中断，写入无效。

5.15.3.19 ICR (0x48)

域	位	读写	复位值	描述
reserved	31:1	R0	0x0	保留
ICR	0	R0	0x0	清除中断。如果下面的中断中的任何一个处于活动状态，则设置此寄存器。读取清除 ssi_txo_intr、ssi_rxu_intr、ssi_rxo_intr 和 ssi_mst_intr 中断。写到这个寄存器是没有效果的。

5.15.3.20 DMACR (0x4C)

域	位	读写	复位值	描述
reserved	31:2	RW	0x0	保留
TDMAE	1	RW	0x0	DMA 发送使能，该位可以启用/禁用 FIFO 的 DMA 传输通道。
RDMAE	0	RW	0x0	DMA 接收使能，该位可以启用/禁用 FIFO DMA 传输通道。

5.15.3.21 DMATDLR (0x50)

域	位	读写	复位值	描述
reserved	31:3	RW	0x0	保留
DMATDL	2:0	RW	0x0	发送数据等级。该位控制 DMA 请求的发送等级。当 dma_tx_req 在发送 FIFO 中的有效数据项的数量等于或低于此字段值时生成的，并且 TDMAE=1。

5.15.3.22 DMARDLR (0x54)

域	位	读写	复位值	描述
reserved	31:3	RW	0x0	保留
DMARDL	2:0	RW	0x0	接收数据等级。该位控制 DMA 请求的接收等级。当接收 FIFO 中的有效值数据的数量等于或高于此字段值+1 和 RDMAE=1 时，将生成 dma_rx_req。

5.15.3.23 IDR (0x58)

域	位	读写	复位值	描述
IDCODE	31:0	R0	0xffffffff	识别码。即外部设备标识代码。

5.15.3.24 DR (0x60-0xEC)

域	位	读写	复位值	描述
reserved	31:16	RW	0x0	保留
DR	15:0	RW	0x0	数据寄存器。写入此寄存器时必须对数据进行右对齐，读取数据时自动右对齐。 读：接收 FIFO 写：发送 FIFO

5.15.3.25 RX_SAMPLE_DLY (0xF0)

域	位	读写	复位值	描述
reserved	31:8	RW	0x0	保留
RSD	7:0	RW	0x0	接收数据延时。这个寄存器用于延时输入信号的采样。每个值表示输入信号采样的单个 ssi_clk 延时。 注意：如果这个寄存器被编程的值超过内部寄存器 (SSI_RX_DLY_SR_DEPTH) 的深度，延时为 0。

5.15.3.26 SPI_CS_REG (0x100)

域	位	读写	复位值	描述
reserved	31:8	RW	0x0	保留
spi_cs3_en	7	RW	0x0	片选 3 使能控制，写 1 使能打开，片选通过 spi_cs3 控制，写 0 不使能，片选由控制器内部控制。
spi_cs2_en	6	RW	0x0	片选 2 使能控制，写 1 使能打开，片选通过 spi_cs2 控制，写 0 不使能，片选由控制器内部控制。
spi_cs1_en	5	RW	0x0	片选 1 使能控制，写 1 使能打开，片选通过 spi_cs1 控制，写 0 不使能，片选由控制器内部控制。
spi_cs0_en	4	RW	0x0	片选 0 使能控制，写 1 使能打开，片选通过 spi_cs0 控制，写 0 不使能，片选由控制器内部控制。
spi_cs3	3	RW	0x0	片选 3 信号，1: cs3 有效（为低）；0: cs3 无效（为高）。
spi_cs2	2	RW	0x0	片选 2 信号，1: cs2 有效（为低）；0: cs2 无效（为高）。
spi_cs1	1	RW	0x0	片选 1 信号，1: cs1 有效（为低）；0: cs1 无效（为高）。
spi_cs0	0	RW	0x0	片选 0 信号，1: cs0 有效（为低）；0: cs0 无效（为高）。

5.16 SD/SDIO/eMMC 控制器

飞腾派集成 2 个 SD 控制器，用于连接外部 SD、eMMC 器件及安全数字 IO (SDIO) 器件。SD 控制器兼容 SD3.0、SDIO3.0 协议规范，支持 UHS-I 模式。

5.16.1 操作说明

5.16.1.1 复位

主要包含三种复位类型：

- 控制器硬复位：卡和控制器通过开机复位（POR）复位。
- 控制器软复位：支持控制器内 DMA_RESET、FIFO_RESET、控制逻辑 RESET，对应寄存器 cntrl 的 bit[2:0]；其中 CONTROLLER_RESET 复位除 DMA、FIFO 之外的控制逻辑，FIFO 复位完成确认需查询等待寄存器 status 的 bit2 fifo_empty 状态位为 1。

- Device 复位：控制器作为 SD/SDIO 模式时，不支持硬件复位输出，SD 存储模式支持 CMD0 软复位命令操作，而 SDIO 模式可通过对通用控制器寄存器 CCCR 的地址 0x06 上的 bit3 RES 功能位写 1 执行 I/O 复位操作；但控制器作为 eMMC 模式时，可同时支持硬复位输出，即通过对寄存器 card_reset bit0 功能位写 0，使得外接信号 rst_n 拉低，将 eMMC 设备端复位到 pre-idle 态。

5.16.1.2 初始化流程

上电初始化流程参考如下：

1. 设备上电之前，确认设备端接口电压正确；
2. 配置 pwren，使设备上电；
3. 配置寄存器 cntrl，执行复位操作；
4. 配置寄存器 debnce，设置卡插拔消抖参数；
5. 配置中断屏蔽寄存器 int_mask，选择中断使能项；
6. 配置寄存器 ctype，设置 bus width 模式；
7. 配置寄存器 fifoth，设置 DMA burst 长度和 FIFO 门限值；
8. 配置寄存器 bus_mode_reg，使能内部 DMA；
9. 写 0xFFFFFFFF 到寄存器 raw_ints，清除所有中断标志；
10. 配置寄存器 tmout，设置超时参数；
11. 配置 cntrl 使能全局中断；
12. 配置分频时钟（具体参考“时钟分频配置”章节），建议设备端初始化工作频率 400KHz；
13. 发送设备初始化命令序列。

5.16.1.3 时钟分频配置

SD/SDIO/eMMC 控制器主时钟包括 clk_lmc(200MHz) 和 clk_samp(1.2GHz)，clk_lmc(200MHz) 是控制器 APB/AXI 总线时钟，clk_samp(1.2GHz) 为 Device 接口时钟源。Device 接口时钟由 clk_samp 时钟通过一级和二级时钟调频模块生成。两级调频结构如下图所示，门控调频单元相位调整部分为作为细粒度调整，卡时钟控制单元相位调整部分作为区间粗粒度调整。

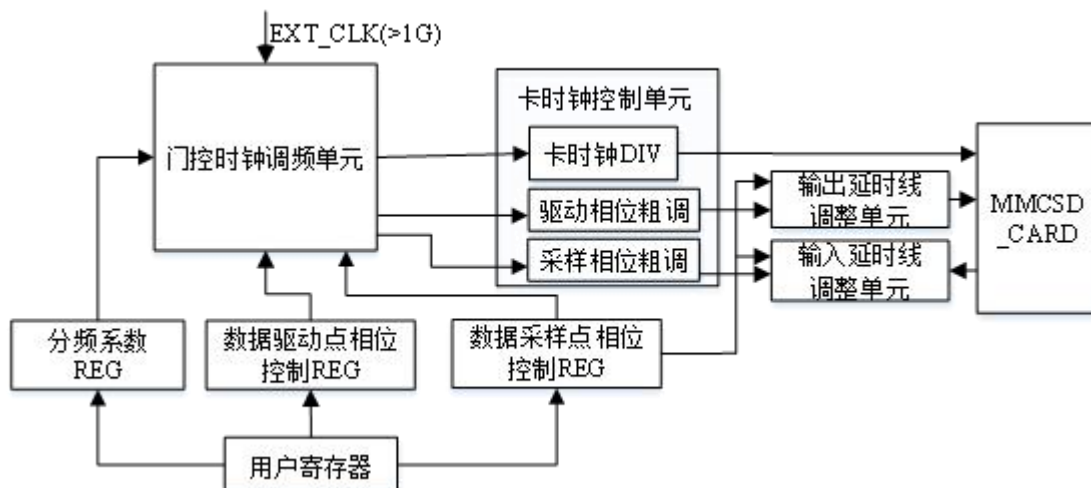


图 5-11 控制器两级时钟调频结构

驱动点相位控制：用于驱动卡命令和数据输出更新时刻（相对 Device 端时钟）。

数据采样点相位控制：用于接收卡命令和数据的采样时刻（相对 Device 端时钟）。

Device 时钟与驱动采样相位的调整由 `clkdiv(0x8)` 与 `clksrc(0x108)` 配合完成。

`clksrc(0x108)` 分频参数决定了控制器内部接口模块时钟频率，`clkdiv(0x8)` 分频参数决定了 CARD 工作时钟频率。

● 一级分频

通过 `clksrc(0x108)` 寄存器配置控制器接口模块时钟分频参数及细调相位 `drv/samp`。

分频参数 = $\text{CLK_DIV_CTRL} + 1$

注意：`CCLK_IN`、`CCLK_IN_DRV`、`CCLK_IN_SAMPLE` 为一级分频后的时钟，均由 `clock gating` 生成，占空比非 50%。

`CCLK_IN_DRV`、`CCLK_IN_SAMPLE` 均为 `CCLK_IN` 相对相移，因此必须小于等于 `CLK_DIV_CTRL`。`CCLK_IN_DRV` 为控制器一级驱动数据点，`CCLK_IN_SAMPLE` 为控制器一级采样数据点。

● 二级分频

通过 `clkdiv(0x8)` 寄存器配置 CARD 时钟。

分频参数 = $2 \times \text{CLK_DIVIDER}[7:0]$

注意：`CLK_DRV`、`CLK_SMPL` 必须小于 `CLK_DIVIDER`，且必须满足 `CLK_SMPL = CLK_DRV + 1`。

最小配置参数：`CLK_DIVIDER=1`，`CLK_DRV=0`，`CLK_SMPL=1`。

`cclk_out` 为最终生成的卡时钟，`cclk_out_en` 为控制器二级驱动数据点，`cclk_smpl_en` 为控制器二级采样数据点。最终的数据驱动时刻和采样时刻由一二级配置参数共同决定。

相位调整关系参考波形如下：

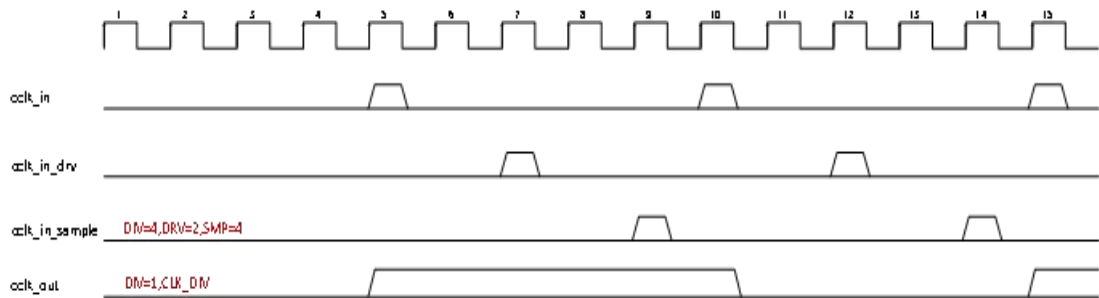


图 5-12 时钟参数设置参考 1

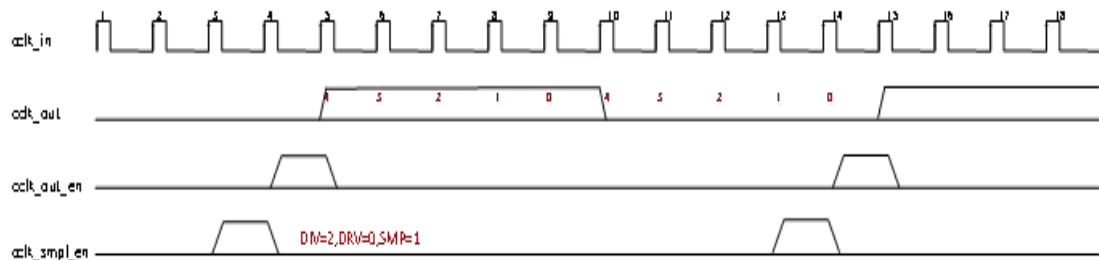


图 5-13 时钟参数设置参考 2

时钟分频配置流程：

1. CLKSRC 配置控制器接口时钟及驱动相位，以及 EXT_CLK_ENABLE。

注意：

- 修改时钟频率前要先禁用 EXT_CLK_ENABL 位，否则修改不生效；
- 确认配置生效，需查询寄存器 CKSTS[0] 确认控制器内部时钟 ready。

2. CLKDIV 配置 CARD 时钟分频系数，分频参数为 $(DIV-1)*2$ 。
3. 注意：配置分频之前要先关断 CLKENA[0]，配置完成，再开启使能。
4. CLKENA[0] 置 1，使能 CARD 时钟。
5. CMD[21][31] 置 1，更新时钟，启动命令。

时钟配置参数参考值：表中 D2 表示十进制数 2，相位参数 [0:2] 表示可以选择 0、1、2 这三个十进制数。

表 5-48 分频参数表

频率 /MHz	一级分频参数 (clksrct (0x108))			二级分频参数 (clkdiv (0x8))		
	分频参数 CLK_DIV_C TRL	输出相位参数 CLK_DRV_PHASE_ CTRL	采样相位参数 CLK_SMPL_PHASE_ CTRL	分频参数 CLK_DIVI DER	输出相位参 数 CLK_DRV	采样相位 参数 CLK_ SMPL
12.5	D2	[0:2]	[0:2]	D16	[15:0]	[15:0]
	D3	[0:3]	[0:3]	D12	[11:0]	[11:0]
	D5	[0:5]	[0:5]	D8	[7:0]	[7:0]
25	D2	[0:2]	[0:2]	D8	[7:0]	[7:0]
	D3	[0:3]	[0:3]	D6	[5:0]	[5:0]
	D5	[0:5]	[0:5]	D4	[3:0]	[3:0]

频 率 /MHz	一级分频参数 (clksrc (0x108))			二级分频参数 (clkdiv (0x8))		
	分 频 参 数 CLK_DIV_C TRL	输出相位参数 CLK_DRV_PHASE_ CTRL	采 样 相 位 参 数 CLK_SMPL_PHASE_ CTRL	分频参数 CLK_DIVI DER	输出相位参 数 CLK_ DRV	采 样 相 位 参 数 CLK_ SMPL
50	D2	[0:2]	[0:2]	D4	[3:0]	[3:0]
	D3	[0:3]	[0:3]	D3	[2:0]	[3:0]
	D5	[0:5]	[0:5]	D2	[1:0]	[1:0]
100	D2	[0:2]	[0:2]	D2	[1:0]	[1:0]
	D5	[0:5]	[0:5]	D1	0	0

5.16.1.4 Block 读/写操作流程-DMA 模式

要实现单数据块或者多数据块 DMA 模式读/写操作，软件需要执行下面的步骤：

1. 将数据块大小 (以字节为单位) 写入到 blksiz 寄存器中；
2. 设置卡块大小长度，需与控制器 blksiz 参数一致；
 - a. 对于 SD/eMMC 卡，配置 SET_BLOCKLEN (CMD16) ；
 - b. 对于 SDIO 卡或 SDCombo 卡的 IO 部分,使用 IO_RW_DIRECT (CMD52) 设置 CCCR 寄存器 (功能 0) 或 FBR 寄存器 (功能 1~7) 中的 I/O 块大小字段。
3. 配置传输字节数寄存器 bytcnt，对于多数据块读/写操作，bytcnt 必须是数据块大小的整数倍；
4. 配置寄存器 bus_mode_reg，选择 DMA 模式；
5. 配置 intr_en_reg 寄存器，开启 DMA 需求中断；
6. 将 存 储 器 构 建 的 第 一 个 描 述 符 链 表 基 址 写 入 到 寄 存 器 (desc_list_star_reg_l/desc_list_star_reg_u)；
7. 将数据读操作的起始数据地址 (卡) 写入到 cmdarg 寄存器；
8. 配置寄存器 cmd，对于 SD 和 eMMC 卡，READ_SINGLE_BLOCK (CMD17) 命令用于单数据块读操作，READ_MULTIPLE_BLOCK (CMD18) 命令用于多数据块读操作；对于 SDIO 卡，IO_RW_EXTENDED (CMD53) 命令用于单数据块以及多数据块传递；cmd 寄存器配置完成后，控制器开始执行读/写命令，命令传输完成后生成 Command Done 中断；
9. 等待 DMA 数据传输完成中断后，读取 raw_ints/status_reg 寄存器，确认是否存在异常状态。

5.16.1.5 中断说明

控制器对应的中断屏蔽寄存器与控制寄存器中断使能后，一旦检测到中断发生，会相应将寄存器对应位置 1，直到软件清 0。

支持的中断说明如下：

控制器-Device 命令传输：

- 命令完成
- 命令响应 CRC 错误
- 命令响应超时
- 命令起始比特错误
- 命令 index 错误

控制器-Device 数据传输：

- 数据传输完成
- FIFO 上溢或下溢，正常传输下不会出现 FIFO 上溢和下溢
- 读卡数据返回超时
- 读/写 CRC 错误
- 读卡操作数据结束位错误
- 写卡操作未收到卡返回的 CRC 状态

5.16.1.6 传输异常处理

传输异常包括 CRC 错误异常、DMA 描述符错误中断异常和 Auto CMD12 错误。

5.16.1.6.1 CRC 错误

在块传输的最后，有可能发生写 CRC 状态错误或读 CRC 错误。对于这种类型的错误，建议丢弃最新的数据块，从损坏的数据块重新传输。

对于多块传输，控制器支持 CMD12 来中止当前进程，并通过一个新的数据命令开始传输。在这种情况下，即使 CMD 寄存器的 SEND_AUTO_STOP 位被设置，mmc_sd 也不会自动发送 CMD12，因为最后一个块没有被传输。另一方面，如果是在最后一个块传输内发生 CRC 错误，mmc_sd 会发送一个自动 CMD12。在这种情况下，控制器驱动可重新发送或重新获得最后一个块的单块传输。

5.16.1.6.2 DMA 描述符错误中断

对于这种错误，建议检索传输上下文，对数据部分进行重置，并从无效的描述符中重新创建描述符链，然后发出新的传输。由于现在要传输的数据可能少于设置值，在新的描述符链中配置的数据长度应该与新的值相匹配。

5.16.1.6.3 Auto CMD12 错误

发送或接收多块传输的最后数据块后，如果控制器启动数据传输时，CMD 寄存器的

SEND_AUTO_STOP 位被设置，mmcscd 会自动向卡发送 CMD12 以停止传输。

当该命令发生错误时，建议控制器驱动程序以下列方式处理：

- Auto CMD12 响应超时：不能确定卡是否接收命令。控制器驱动清除 Auto CMD12 错误状态位，并重新发送 CMD12，直到被卡接收。
- Auto CMD12 响应 CRC 错误：当卡对 CMD12 做出响应时，会中止传输。控制器驱动程序可以忽略该错误并清除错误状态位。
- Auto CMD12 冲突错误或未发送：命令未被发送。控制器驱动程序可手动发送 CMD12。

5.16.1.7 SD 模式配置说明

5.16.1.7.1 SD Bus width 模式配置

1. 控制器端：将寄存器 ctype 中的卡位宽功能位配置为 1bit 或 4bit 模式；
2. Device 端：控制器发送 set bus width command (ACMD6)，配置寄存器 argument bit[1:0] bus width 功能位：
 - Bus_Width=00, 1-bit width
 - Bus_Width=10, 4-bit bus width

5.16.1.7.2 SD 速度模式配置

SD 速度模式配置流程：

1. 发送 CMD6，参数为 0xFFFFFx，R1 响应后读取数据，高速模式的 x=1，SDR50 的 x=2，SDR104 的 x=3，DDR50 的 x=4；
2. 等待数据传输完成位被设置；
3. 检查收到的 512bit 的第 x 位是否设置：
 - 如果 bit 401 为 '0'，表示 SD 卡不支持高速模式并返回；
 - 如果 bit 402 为 '0'，表示 SD 卡不支持 SDR50 模式并返回；
 - 如果 bit 403 为 '0'，表示 SD 卡不支持 SDR104 模式并返回；
 - 如果 bit 403 为 '0'，表示 SD 卡不支持 DDR50 模式并返回。
4. 发送 CMD6，参数为 0x80FFFFx，R1 响应后读取数据，高速模式的 x=1，SDR50 的 x=2，SDR104 的 x=3，DDR50 的 x=4；
5. 检查位域 379:376 是否为 0xF，如果是 0xF 表示功能切换失败并返回；
6. 更改时钟参数，配置卡时钟 CARD_CLK 为 50MHz。

禁用 SD 高速/DDR50/SDR50/SDR104 模式流程：

1. 发送 CMD6，参数为 0x80FFFF0，R1 响应后读取数据；

2. 检查位域 379:376 是否为 0xF，如果是 0xF 表示功能切换失败并返回；
3. 更改时钟参数，配置卡时钟 CARD_CLK 低于 25MHz。

5.16.1.8 eMMC 模式配置

5.16.1.8.1 eMMC Bus width 模式配置

1. 控制器端：将寄存器 ctype 中的卡位宽功能位配置为 1bit/4bit 或 8bit 模式；
2. Device 端：控制器发送 Switch command (CMD6)，配置 BUS_WIDTH[183]：
 - Bus_Width=00, 1-bit width
 - Bus_Width=1, 4-bit bus width
 - Bus_Width=2, 8-bit bus width

5.16.1.8.2 eMMC 速度模式配置

1. 发送 CMD9 获取 eMMC 的 CSD 值；
2. 检查 SPEC_VER 字段的值是否大于等于 4，如果 SPEC_VER 值小于 4，表示 MMC 不支持高速模式并返回；
3. 发送 CMD8 获取 eMMC 的 EXT_CSD 值；
4. 获取 CARD_TYPE 字段的值，判断 eMMC 高速模式时钟是 26MHz 或 52MHz；
5. 发送 CMD6，参数为 0x1B90100；
6. 发送 CMD13，等待卡准备就绪；
7. 发送 CMD8 获取 MMC 的 EXT_CSD 值；
8. 检查 HS_TIMING 是否为 1，如果 HS_TIMING 不是 1，表示 MMC 切换到高速模式失败并返回；
9. 更改时钟参数，配置卡时钟 CARD_CLK，根据 CARD_TYPE 配置为 26MHz 或 52MHz。

5.16.1.9 SDIO 模式配置说明

5.16.1.9.1 SDIO Bus width 模式配置

1. 控制器端：将寄存器 ctype 中的卡位宽功能位配置为 1bit 或 4bit 模式；
2. Device 端：控制器发送 IO_IW_DIRECT (CMD52) 命令配置 CCCR 的地址 0x07 上的 bit[1:0] Bus_Width 功能位。
 - Bus_Width=00, 1-bit width
 - Bus_Width=10, 4-bit bus width

5.16.1.9.2 SDIO 速度模式配置

1. 发送 IO_IW_DIRECT (CMD52) 命令查询卡通用控制器寄存器 (CCCR) 的地址 0x13 上的 bit0 SHS (Support High-speed) 功能位
 - SHS=0, SDIO 卡不支持 High-speed 模式;
 - SHS=1, SDIO 卡支持高速模式, 控制器可通过 CMD52 命令使能 CCCR 的地址 0x13 上的 bit1 EHS 功能位使能 High-speed 模式。
2. 参考 5.17.1.3 章节-时钟分频配置部分描述, 使用寄存器 CLKSRC 和 CLKDIV 配置卡时钟 CARD_CLK 参数。

5.16.2 寄存器列表

表 5-49 SD/SDIO/eMMC 寄存器基地址

名称	基地址
MMCSD0	0x000_2800_0000
MMCSD1	0x000_2800_1000

表 5-50 SD/SDIO/eMMC 寄存器列表

寄存器	偏移量	描述
cntrl	0x0	控制寄存器
pwren	0x4	卡供电开关
clkdiv	0x8	时钟分频配置
clkena	0x10	card 时钟使能寄存器
tmout	0x14	超时寄存器
ctype	0x18	卡类型
blksiz	0x1C	块大小
bytcnt	0x20	传输字节数
int_mask	0x24	中断屏蔽寄存器
cmdarg	0x28	命令参数寄存器
cmd	0x2C	命令寄存器
resp0	0x30	响应寄存器 0
resp1	0x34	响应寄存器 1
resp2	0x38	响应寄存器 2
resp3	0x3C	响应寄存器 3
masked_ints	0x40	已屏蔽中断状态寄存器
raw_ints	0x44	原始中断状态寄存器
status	0x48	控制器状态寄存器
fifoth	0x4C	FIFO 阈值寄存器
card_detect	0x50	卡检测寄存器
card_write_prt	0x54	写保护寄存器
cksts	0x58	时钟 Monitor 寄存器
tran_crd_cnt_mx	0x5C	控制器-卡传输的字节数

寄存器	偏移量	描述
tran_fifo_cnt_mx	0x60	memory-FIFO 之间传输的字节数
debnc	0x64	消抖配置寄存器
uid	0x68	用户 ID
vid	0x6C	控制器版本
uhs_reg	0x74	UHS-1 寄存器
card_reset	0x78	复位寄存器
bus_mode_reg	0x80	总线模式寄存器
desc_list_star_reg_l	0x88	描述符地址寄存器
desc_list_star_reg_u	0x8C	描述符地址寄存器
status_reg	0x90	内部 DMAC 状态寄存器
intr_en_reg	0x94	中断使能寄存器
cur_desc_addr_reg_l	0x98	当前主机描述符地址寄存器
cur_desc_addr_reg_u	0x9C	当前主机描述符地址寄存器
cur_buf_addr_reg_l	0xA0	当前缓存描述符地址寄存器
cur_buf_addr_reg_u	0xA4	当前缓存描述符地址寄存器
cardthctl	0x100	卡阈值控制寄存器
clksrc	0x108	UHS 寄存器扩展
enable-shift	0x110	相移使能寄存器
data	0x200	数据 FIFO 寄存器

5.16.3 寄存器说明

5.16.3.1 cntrl (0x0)

域	位	读写	复位值	描述
reserved	31:26	RW	0x0	保留
USE_INTERNAL_DMAMC	25	RW	0x0	用内部 DMA。1 有效。
ENABLE_OD_PULLUP	24	RW	0x0	外部开漏输出。1 有效。
CARD_VOLTAGE_B	23:20	RW	0x0	B 电压选择。1 有效。
CARD_VOLTAGE_A	19:16	RW	0x0	A 电压选择。1 有效。
ENDIAN	11	RW	0x0	0: 小端; 1: 大端。
SEND_AUTO_STOP_CCSD	10	RW	0x0	对应 CCD, 自动 STOP (不支持)。
SEND_CCSD	9	RW	0x0	发送 CCSD (不支持)。
ABORT_READ_DATA	8	RW	0x0	读暂停异常清除 data FSM。1 有效。
SEND_IRQ_RESPONSE	7	RW	0x0	MMC 中断自动响应配置。1 有效。
READ_WAIT	6	RW	0x0	SDIO 读等待。1 有效。
RES_DMA_ENABLE	5	RW	0x0	保留外部 DMA 模式使能。
INT_ENABLE	4	RW	0x0	全局中断使能配置, 和 int_mask_n 寄存器相与使能对应中断。1: 使能。
RES	3	RW	0x0	保留
DMA_RESET	2	RW	0x0	复位内部 DMA。1 有效。
FIFO_RESET	1	RW	0x0	复位 FIFO。1 有效。
CONTROLLER_RESET	0	RW	0x0	复位控制器, 除 DMA, FIFO。

5.16.3.2 pwren (0x4)

域	位	读写	复位值	描述
POWER_ENABLE	31:0	RW	0x0	卡供电开关。 0: 关 1: 开

5.16.3.3 clkdiv (0x8)

域	位	读写	复位值	描述
reserved	31:24	RW	0x0	保留
CLK_SMPL	23:16	RW	0x0	采样相位区间设置
CLK_DRV	15:8	RW	0x0	输出相位区间设置
CLK_DIVIDER	7:0	RW	0x0	时钟分频参数设置, 分频参数=2*CLK_DIVIDER

5.16.3.4 clkena (0x10)

域	位	读写	复位值	描述
CCLK_LOW_POWER	31:16	RW	0x0	功耗模式控制 0: 非低功耗 1: 低功耗
CCLK_ENABLE	15:0	RW	0x0	card 时钟使能控制 0: Clock disabled 1: Clock enabled

5.16.3.5 tmout (0x14)

域	位	读写	复位值	描述
DATA_TIMEOUT	31:8	RW	0xFFFFFFFF	读卡超时 (以卡时钟为单位)
RESPONSE_TIMEOUT	7:0	RW	0x40	响应超时 (以卡时钟为单位)

5.16.3.6 ctype (0x18)

域	位	读写	复位值	描述
CARD0_WIDTH1	31:16	RW	0x0	Non 8-bit mode 1-8-bit mode
CARD0_WIDTH2	15:0	RW	0x0	0: 1-bit mode; 1: 4-bit mode

5.16.3.7 blksiz (0x1C)

域	位	读写	复位值	描述
BLOCK_SIZE	31:0	RW	0x200	块大小

5.16.3.8 bytcnt (0x20)

域	位	读写	复位值	描述
BYTE_COUNT	31:0	RW	0x0	传输字节数

5.16.3.9 int_mask (0x24)

域	位	读写	复位值	描述
reserved	31:17	RW	0x0	保留
SDIO_INT_MASK_CARD0	16	RW	0x0	SDIO interrupt 中断 0: 屏蔽; 1: 使能
EBE_INT_MASK	15	RW	0x0	读写结束位错误/写未收到 CRC 中断 0: 屏蔽; 1: 使能
ACD_INT_MASK	14	RW	0x0	Auto command 完成中断 0: 屏蔽; 1: 使能
SBE_BCI_INT_MASK	13	RW	0x0	起始位错误/busy 撤销中断 0: 屏蔽; 1: 使能
HLE_INT_MASK	12	RW	0x0	硬件锁存中断 0: 屏蔽; 1: 使能
FRUN_INT_MASK	11	RW	0x0	FIFO 上下溢中断 0: 屏蔽; 1: 使能
HTO_INT_MASK	10	RW	0x0	数据 starv/电源切换中断 0: 屏蔽; 1: 使能
DRT0_INT_MASK	9	RW	0x0	数据读超时中断 0: 屏蔽; 1: 使能
RTO_INT_MASK	8	RW	0x0	响应超时中断 0: 屏蔽; 1: 使能
DCRC_INT_MASK	7	RW	0x0	数据 CRC 校验错误中断 0: 屏蔽; 1: 使能
RCRC_INT_MASK	6	RW	0x0	响应 CRC 错误中断 0: 屏蔽; 1: 使能
RXDR_INT_MASK	5	RW	0x0	接收 FIFO 请求中断 0: 屏蔽; 1: 使能
TXDR_INT_MASK	4	RW	0x0	发送 FIFO 请求中断 0: 屏蔽; 1: 使能
DT0_INT_MASK	3	RW	0x0	Data transfer over (DT0) interrupt enable 0: 屏蔽; 1: 使能
CMD_INT_MASK	2	RW	0x0	命令传输完成中断 0: 屏蔽; 1: 使能
RE_INT_MASK	1	RW	0x0	响应错误中断 0: 屏蔽; 1: 使能
CD_INT_MASK	0	RW	0x0	卡检测中断 0: 屏蔽; 1: 使能

5.16.3.10 cmdarg (0x28)

域	位	读写	复位值	描述
CMD_ARG	31:0	RW	0x0	命令参数

5.16.3.11 cmd (0x2C)

域	位	读写	复位值	描述
START_CMD	31:30	RW	0x0	启动命令
USE_HOLD_REG	29	RW	0x1	与命令的发出相关，此位必须为 1。 0: 旁路 HOLD Register 1: 使能 HOLD Register
VOLT_SWITCH	28	RW	0x0	0: 无电压切换; 1: 使能电压切换
reserved	27:22	RW	0x0	保留
UPDATE_CLOCK_REGISTERS_ONLY	21	RW	0x0	1: 不发送命令，只更新时钟 REG
CARD_NUMBER	20:16	RW	0x0	保留
SEND_INITIALIZATION	15	RW	0x0	在发送命令之前，等待 80 个 cycle 初始时钟序列完成
STOP_ABORT_CMD	14	RW	0x0	1: stop/abort 操作
WAIT_PRVDATA_COMPLETE	13	RW	0x0	0: 立即发送命令
SEND_AUTO_STOP	12	RW	0x0	1: 自动发送 stop
reserved	11	RW	0x0	保留
READ_WRITE	10	RW	0x0	0: 读卡; 1: 写卡
DATA_EXPECTED	9	RW	0x0	0: DTA 无数据; 1: DAT 有数据
CHECK_RESPONSE_CRC	8	RW	0x0	0: 不检查 CRC; 1: 检查 CRC
RESPONSE_LENGTH	7	RW	0x0	0: 短响应; 1: 长响应
RESPONSE_EXPECT	6	RW	0x0	0: 无响应; 1: 有响应
CMD_INDEX	5:0	RW	0x0	命令索引

5.16.3.12 resp0 (0x30)

域	位	读写	复位值	描述
RESPONSE0	31:0	R0	0x0	响应寄存器 0 bit[31:0]

5.16.3.13 resp1 (0x34)

域	位	读写	复位值	描述
RESPONSE1	31:0	R0	0x0	响应寄存器 1 bit[63:32]

5.16.3.14 resp2 (0x38)

域	位	读写	复位值	描述
RESPONSE2	31:0	R0	0x0	响应寄存器 2 bit[95:64]

5.16.3.15 resp3 (0x3C)

域	位	读写	复位值	描述
RESPONSE3	31:0	R0	0x0	响应寄存器 3 bit[127:96]

5.16.3.16 masked_ints (0x40)

域	位	读写	复位值	描述
reserved	31:17	R0	0x0	保留
SDIO_INTERRUPT_CARD0	16	R0	0x0	SDIO card 中断
END_BIT_ERROR_INTERRUPT	15	R0	0x0	读写 End-bit 错误/写未收到 CRC
AUTO_COMMAND_DONE_INTERRUPT	14	R0	0x0	自动命令完成 (ACD)
BUSY_COMPLETE_INTERRUPT_INTERRUPT	13	R0	0x0	起始位错误 (SBE)/Busy 完成 (BCI)
HARDWARE_LOCKED_WRITE_INTERRUPT	12	R0	0x0	硬件锁存写错误 (HLE)
FIFO_UNDER_OVER_RUN_INTERRUPT	11	R0	0x0	FIFO 上/下溢错误 (FRUN)
HOST_TIMEOUT_INTERRUPT	10	R0	0x0	数据饥饿超时 (HTO)/ Volt_switch
DATA_READ_TIMEOUT_INTERRUPT	9	R0	0x0	数据读超时 (DRT0)
RESPONSE_TIMEOUT_INTERRUPT	8	R0	0x0	响应超时 (RTO)
DATA_CRC_ERROR_INTERRUPT	7	R0	0x0	数据 CRC 错误 (DCRC)
RESPONSE_CRC_ERROR_INTERRUPT	6	R0	0x0	响应 CRC 错误 (RCRC)
RECEIVE_FIFO_DATA_REQUEST_INTERRUPT	5	R0	0x0	RX FIFO 数据请求 (RXDR)
TRANSMIT_RECEIVE_FIFO_DATA_INTERRUPT	4	R0	0x0	TX FIFO 数据请求 (TXDR)
DATA_TRANSFER_OVER_INTERRUPT	3	R0	0x0	数据传输完成 (DTO)
COMMAND_DONE_INTERRUPT	2	R0	0x0	命令完成 (CD)
RESPONSE_ERROR_INTERRUPT	1	R0	0x0	响应错误 (RE)
CARD_DETECT_INTERRUPT	0	R0	0x0	卡检测 (CD)

5.16.3.17 raw_ints (0x44)

域	位	读写	复位值	描述
reserved	31:17	RW	0x0	保留
SDIO_INTERRUPT_CARD0	16	RW	0x0	SDIO card 中断
END_BIT_ERROR_STATUS	15	RW	0x0	读写 End-bit 错误/写未收到 CRC
AUTO_COMMAND_DONE_STATUS	14	RW	0x0	自动命令完成 (ACD)
BUSY_COMPLETE_STATUS	13	RW	0x0	起始位错误 (SBE)/Busy 完成 (BCI)
HARDWARE_LOCKED_WRITE_STATUS	12	RW	0x0	硬件锁存写错误 (HLE)
FIFO_UNDER_OVER_RUN_STATUS	11	RW	0x0	FIFO 上/下溢错误 (FRUN)
HOST_TIMEOUT_STATUS	10	RW	0x0	数据饥饿超时 (HTO)/Volt_switch
DATA_READ_TIMEOUT_STATUS	9	RW	0x0	数据读超时 (DRT0)
RESPONSE_TIMEOUT_STATUS	8	RW	0x0	响应超时 (RTO)
DATA_CRC_ERROR_STATUS	7	RW	0x0	数据 CRC 错误 (DCRC)
RESPONSE_CRC_ERROR_STATUS	6	RW	0x0	响应 CRC 错误 (RCRC)
RECEIVE_FIFO_DATA_REQUEST_INTERRUPT	5	RW	0x0	RX FIFO 数据请求 (RXDR) NON-DMA 使用
TRANSMIT_RECEIVE_FIFO_DATA_STATUS	4	RW	0x0	TX FIFO 数据请求 (TXDR) NON-DMA 使用

域	位	读写	复位值	描述
DATA_TRANSFER_OVER_STATUS	3	RW	0x0	数据传输完成 (DT0)
COMMAND_DONE_STATUS	2	RW	0x0	命令完成 (CD)
RESPONSE_ERROR_STATUS	1	RW	0x0	响应错误 (RE)
CARD_DETECT_STATUS	0	RW	0x0	卡检测 (CD)

5.16.3.18 status (0x48)

域	位	读写	复位值	描述
DMA_REQ	31	R0	0x0	DMA 请求
DMA_ACK	30	R0	0x0	DMA 确认
FIFO_COUNT	29:17	R0	0x0	FIFO 填充计数器
RESPONSE_INDEX	16:11	R0	0x0	响应索引
DATA_STATE_MC_BUSY	10	R0	0x0	DATA TX/RX FSM busy 0: not busy; 1: busy
DATA_BUSY	9	R0	0x0	卡 busy 0: not busy; 1: busy
DATA_3_STATUS	8	R0	0x0	DATA[3] 卡在位检测 0: 不在位; 1: 在位
COMMAND_FSM_STATES	7:4	R0	0x0	cmd FSM
FIFO_FULL	3	R0	0x0	FIFO full
FIFO_EMPTY	2	R0	0x0	FIFO empty
FIFO_TX_WATERMARK	1	R0	0x0	达到 FIFO_TX 标记
FIFO_RX_WATERMARK	0	R0	0x0	达到 FIFO_RX 标记

5.16.3.19 fifoth (0x4C)

域	位	读写	复位值	描述
DMA_Multiple_Transaction_Size	30:28	RW	0x0	SRC/DEST_MSIZE 000: 1 001: 4 010: 8 011: 16 100: 32 101: 64 110: 128 111: 256
RX_WMark	27:16	RW	0x1ff	FIFO threshold
TX_WMark	11:0	RW	0x0	FIFO threshold

5.16.3.20 card_detect (0x50)

域	位	读写	复位值	描述
CARD0_DETECT_N	31:0	R0	0x0	1: 卡不在位; 0: 卡在位

5.16.3.21 card_write_prt (0x54)

域	位	读写	复位值	描述
WRITE_PROTECT_0	31:0	RO	0x0	1: 写保护; 0: 无写保护

5.16.3.22 cksts (0x58)

域	位	读写	复位值	描述
CCLK_RDY	31:0	RO	0x0	Device 接口模块时钟 ready

5.16.3.23 tran_crd_cnt_mx (0x5C)

域	位	读写	复位值	描述
TRANS_CARD_BYTE_COUNT	31:0	RO	0x0	Device 接口模块到 Device 传输的字节数

5.16.3.24 tran_fifo (0x60)

域	位	读写	复位值	描述
TRANS_FIFO_BYTE_COUNT	31:0	RO	0x0	MEM&FIFO 之间传输的字节数

5.16.3.25 debnce (0x64)

域	位	读写	复位值	描述
DEBOUNCE_COUNT	31:0	RW	0xFFFFFFFF	去抖时钟数, 参考值 5-25 ms

5.16.3.26 uid (0x68)

域	位	读写	复位值	描述
UID	31:0	RW	0x59595959	用户 ID

5.16.3.27 vid (0x6C)

域	位	读写	复位值	描述
VID	31:0	RO	0x6488280A	控制器版本

5.16.3.28 uhs_reg (0x74)

域	位	读写	复位值	描述
reserved	31:16	RW	0x0	保留
VOLT_REG_0	15:0	RW	0x0	外部调压器接口电压 0: 3.3V Vdd; 1: 1.8V Vdd

5.16.3.29 card_reset (0x78)

域	位	读写	复位值	描述
CARD0_RESET	31:0	RW	0x0	1: 运行; 0: 复位

5.16.3.30 bus_mode_reg (0x80)

域	位	读写	复位值	描述
reserved	31:11	RO	0x0	保留
PBL	10:8	RO	0x0	burst LEN 000: 1 transfers 001: 4 transfers 010: 8 transfers 011: 16 transfers 100: 32 transfers 101: 64 transfers 110: 128 transfers 111: 256 transfers
DE	7	RW	0x1	IDMAC 使能
reserved	6:2	RW	0x0	保留
FB	1	RW	0x0	固定 Burst 0: SINGLE & INCR 1: 自动选择 SINGLE, INCR4, INCR8, 或者 INCR16
SWR	0	RW	0x0	软复位 IDMA 复位内部 REG, 自动清 0

5.16.3.31 desc_list_star_reg_l (0x88)

域	位	读写	复位值	描述
DBADDRL	31:0	RW	0x0	desc_list_star_reg[31:0]

5.16.3.32 desc_list_star_reg_u (0x8C)

域	位	读写	复位值	描述
DBADDRU	31:0	RW	0x0	desc_list_star_reg[63:32]

5.16.3.33 status_reg (0x90)

域	位	读写	复位值	描述
FSM	31:13	RO	0x0	DMAC FSM 状态
EB	12:10	RO	0x0	异常标识 3'b001: 发送异常 3'b010: 接收异常

域	位	读写	复位值	描述
				bit[2]置1, 不产生该中断
AIS	9	RW	0x0	异常中断汇总
NIS	8:6	RW	0x0	正常中断汇总
CES	5	RW	0x0	卡错误汇总 EBE: 结束位错误 RT0: 响应超时/Boot Ack 超时 RCRC: 响应 CRC 错误 SBE: 起始位错误 DRT0: 数据读超时/BDS 超时 DCRC: 数据 CRC 错误 RE: 响应错误
DU	4:3	RW	0x0	描述符不可读中断
FBE	2	RW	0x0	总线错误中断 1: 清除 bit[12:10] 中断
RI	1	RW	0x0	接收完成中断, 针对描述符
TI	0	RW	0x0	发送完成中断, 针对描述符

5.16.3.34 intr_en_reg (0x94)

域	位	读写	复位值	描述
reserved	31:10	RW	0x0	保留
AISE	9	RW	0x0	异常中断使能
NIE	8	RW	0x0	正常中断使能
CESE	5	RW	0x0	卡错误中断使能
DUE	4	RW	0x0	描述符不可读中断使能
FBEE	2	RW	0x0	总线错误中断使能
RIE	1	RW	0x0	接收中断使能
TIE	0	RW	0x0	发送中断使能

5.16.3.35 cur_desc_addr_reg_l (0x98)

域	位	读写	复位值	描述
DSCADDR_L	31:0	RW	0x0	cur_desc_addr_reg[31:0]

5.16.3.36 cur_desc_addr_reg_u (0x9C)

域	位	读写	复位值	描述
DSCADDR_U	31:0	RW	0x0	cur_desc_addr_reg[63:32]

5.16.3.37 cur_buf_addr_reg_l (0xA0)

域	位	读写	复位值	描述
BUFADDR_L	31:0	RW	0x0	cur_buf_addr_reg[31:0]

5.16.3.38 cur_buf_addr_reg_u (0xA4)

域	位	读写	复位值	描述
BUFADDRU	31:0	RW	0x0	cur_buf_addr_reg[63:32]

5.16.3.39 cardthctl (0x100)

域	位	读写	复位值	描述
reserved	31:N+1	RO	0x0	保留
CARDRDTHRESHOLD	N:16	RW	0x0	读卡阈值大小，只有在接收 FIFO 中有满足该值定义的空间可用时，才进行传输。建议该配置值大于或等于读取传输的 BLOCK 大小，以确保在数据块之间卡时钟不会停止。 N 取值范围为 23~28。 N=28: FIFO_DEPTH 为 256 N=27: FIFO_DEPTH 为 128 N=26: FIFO_DEPTH 为 64 N=25: FIFO_DEPTH 为 32 N=24: FIFO_DEPTH 为 16 N=23: FIFO_DEPTH 为 8
CARDWRTHREN	2	RO	0x0	写卡 Threshold 使能
BUSY_CLR_INT_EN	1	RW	0x0	Busy 清中断
CARDRDTHREN	0	RW	0x0	卡读 Threshold 使能

5.16.3.40 clksrc (0x108)

域	位	读写	复位值	描述
reserved	31	RW	0x0	保留
CLK_DRV_PHASE_CTRL	30:24	RW	0x0	输出相位参数，相对于控制器端时钟相位点。
CLK_SMPL_PHASE_CTRL	22:16	RW	0x0	采样相位参数，相对于控制器端时钟相位点。
CLK_DIV_CTRL	14:8	RW	0x5	分频参数，CIU f= CLK_DIV_CTRL +1, MIN=1。
EXT_CLK_ENABLE	1	RW	0x0	外部时钟—控制器内部设备接口模块时钟源使能。
reserved	0	RW	0x0	保留

5.16.3.41 enable_shift (0x110)

域	位	读写	复位值	描述
reserved	31:2	RW	0x0	保留
ENABLE_SHIFT	1:0	RW	0x0	00: 默认相位；01: 移位使能到下一个上升沿；10: 移位使能到下一个下降沿

5.16.3.42 data (0x200)

域	位	读写	复位值	描述
DATA	31:0	RW	0x0	数据 FIFO 寄存器

5.17 NAND Flash 控制器

飞腾派集成的 Nand Flash 控制器支持 ONFI 2.2 及以下接口协议。NAND Flash 是一种非易失性存储器，具有存储密度高、编程擦除速度快、成本低、寿命高等特点，是存储芯片的主流。NAND Flash 控制器实现芯片与 NAND Flash 存储设备之间的数据传输。

5.17.1 操作说明

5.17.1.1 控制器初始化流程

1. 配置 timing mode，异步模式配置 Nf_timing0_reg~Nf_timing5_reg，同步模式配置 Nf_timing6_reg~Nf_timing14_reg，Toggle 模式配置 Nf_timing15~Nf_timing18_reg。具体参数值的选择根据 flash 设备支持的 timing mode。
2. 如需配置命令、地址、数据之间的时间间隔，操作寄存器 Nf_interval_reg。
3. 如需配置请求之间的时间间隔，操作寄存器 Nf_cmdintval_reg。
4. 如需配置 FIFO 超时时间，操作寄存器 Nf_fiftimeout_reg。
5. 配置 FIFO 的满阈值和空阈值，操作 Nf_fiflevel0_reg 和 Nf_fiflevel1_reg。
6. 如需打开相应中断，配置 Nf_intrmask_reg 寄存器使能相应中断。
7. onfi 同步、toggle 模式下读数据是通过高频时钟（2GHz）采样得到，采样相位调节，操作 Nf_ctrl1_reg。
8. 配置 Nf_ctrl0_reg 寄存器，包括 spare size 大小，spare size 使能，ECC 纠错位数，ECC 使能，接口模式和接口位宽，并使能 nandflash 控制器。

5.17.1.2 发送请求流程

1. 首先将描述符表和数据填入内存中。
2. 将描述表地址的低 32 位写入 Nf_maddr0_reg 寄存器。
3. 将描述表地址的高 8 位写入 Nf_maddr1_reg 寄存器，并使能 dma 请求。
4. 如果没有使能中断，读 Nf_state_reg 寄存器判断命令发送完成。如果使能了中断，检测到中断后，读 Nf_intr_reg 寄存器获得中断类型。
5. 如果使能了 ECC，以 512B 为单位进行检验，读 Nf_encodefincnt_reg 判断编码完成，读 Nf_decodefincnt_reg 判断解码完成，然后读 Nf_errlocation1_reg~

Nf_errlocation64_reg 判断错误位置。

5.17.1.3 错误相关操作

读 Nf_state_reg 寄存器发现产生错误后，写 Nf_errclr_reg 寄存器清除相应错误。

5.17.1.4 FIFO 清空操作

读 Nf_fiforsta_reg 寄存器判断 FIFO 中有数据，需要写 Nf_fifree_reg 寄存器，清空 FIFO，才能继续进行下一操作。

5.17.1.5 调试相关操作

读 Nf_debug_reg 寄存器获取控制器状态机信息。读 dma、nand 接口相关以及 ECC 剩余数据寄存器（偏移地址在 0x090~0x0b0）获得数据信息。

5.17.1.6 写保护操作

配置 Nf_wp_reg 寄存器，控制写保护信号。

5.17.1.7 ECC 编码和解码数据长度（page_count）

表 5-51 ECC 编解码数据长度表

pagesize	Ecc_correct=2	Ecc_correct=4
512B	4	7
2KB	d	1a
4KB	1a	34
8KB	34	68
16KB	68	d0

5.17.1.8 错误状态以及处理相关操作

Nf_intr_reg 寄存器 bit[13] 为几种错误统一的中断状态使能位。当该位使能后，产生中断后可读 Nf_state_reg 寄存器 bit[20] 至 [23] 获取当前是什么错误，再通过 Nf_errclr_reg 寄存器写 1 清除掉错误后继续工作。

5.17.1.9 时钟频率配置

NAND Flash 控制器主时钟包括 clk_nand(600MHz) 和 clk_samp(1.2GHz)，clk_samp(1.2GHz) 为同步模式输入数据的采样时钟，输出给设备的时钟信号由

clk_lsd(600MHz)分频得到，异步模式下由于没有时钟，频率由异步时序参数决定，各参数调节寄存器为 Nf_timing0~5reg, Nf_timing13~18reg；同步模式下，对应的频率配置寄存器为 Nf_timing7~8reg，其中 tCK 参数表示时钟频率。

5.17.2 寄存器列表

表 5-52 NAND Flash 寄存器基地址

名称	基地址
NANDFlash	0x000_2800_2000

表 5-53 NAND Flash 寄存器列表

寄存器名称	偏移	描述
Nf_ctrl0_reg	0x000	控制寄存器 0
Nf_ctrl1_reg	0x004	控制寄存器 1
Nf_maddr0_reg	0x008	内存中存储的描述符表首地址的低 32 位
Nf_maddr1_reg	0x00C	内存中存储的描述符表首地址的高 8 位和 DMA 使能
Nf_timing0_reg	0x010	Timing0 寄存器
Nf_timing1_reg	0x014	Timing1 寄存器
Nf_timing2_reg	0x018	Timing2 寄存器
Nf_timing3_reg	0x01C	Timing3 寄存器
Nf_timing4_reg	0x020	Timing4 寄存器
Nf_timing5_reg	0x024	Timing5 寄存器
Nf_timing6_reg	0x028	Timing6 寄存器
Nf_timing7_reg	0x02C	Timing7 寄存器
Nf_timing8_reg	0x030	Timing8 寄存器
Nf_timing9_reg	0x034	Timing9 寄存器
Nf_timing10_reg	0x038	Timing10 寄存器
Nf_timing11_reg	0x03C	Timing11 寄存器
Nf_timing12_reg	0x040	Timing12 寄存器
Nf_timing13_reg	0x044	Timing13 寄存器
Nf_timing14_reg	0x048	Timing14 寄存器
Nf_timing15_reg	0x04C	Timing15 寄存器
Nf_timing16_reg	0x050	Timing16 寄存器
Nf_timing17_reg	0x054	Timing17 寄存器
Nf_timing18_reg	0x058	Timing18 寄存器
Nf_fiforsta_reg	0x05C	FIFO 状态寄存器
Nf_interval_reg	0x060	命令、地址、数据之间的间隔时间配置寄存器
Nf_cmdintval_reg	0x064	请求之间的间隔时间配置寄存器
Nf_fiftimeout_reg	0x068	FIFO 超时计数
Nf_fiflevel0_reg	0x06C	FIFO 阈值选择
Nf_fiflevel1_reg	0x070	FIFO 阈值选择
Nf_wp_reg	0x074	WP 使能寄存器

寄存器名称	偏移	描述
Nf_fifree_reg	0x078	FIFO 清空寄存器
Nf_state_reg	0x07C	状态寄存器
Nf_intrmask_reg	0x080	中断屏蔽位寄存器
Nf_intr_reg	0x084	中断状态寄存器
Nf_debug_reg	0x088	debug 寄存器
Nf_errclr_reg	0x08C	错误清除寄存器
Nf_dmardcnt_reg	0x090	DMA 读页操作当前的剩余数据
Nf_dmardsparcnt_reg	0x094	DMA 读 spar 操作当前的剩余数据
Nf_dmawrcnt_reg	0x098	DMA 写页操作当前的剩余数据
Nf_dmawrsparcnt_reg	0x09C	DMA 写 spar 操作当前的剩余数据
Nf_intwrcnt_reg	0x0A0	NAND 接口写操作当前剩余数据
Nf_intasyrdcnt_reg	0x0A4	NAND 接口异步读操作当前剩余数据
Nf_intsyntogrdcnt_reg	0x0A8	NAND 接口同步写或 tog 读操作当前剩余数据
Nf_encodefincnt_reg	0x0AC	一次请求中的 ECC 编码完成次数计数器
Nf_encodedatcnt_reg	0x0B0	ECC 编码发送计数器
Nf_decodefincnt_reg	0x0B4	一次请求中的 ECC 校验完成计数器
Nf_errlocation(4*(x-1)+1)_reg	0x0B8+0x10*(x-1)	错误定位(4*(x-1)+1) 寄存器
Nf_errlocation(4*(x-1)+2)_reg	0x0BC+0x10*(x-1)	错误定位(4*(x-1)+2) 寄存器
Nf_errlocation(4*(x-1)+3)_reg	0x0C0+0x10*(x-1)	错误定位(4*(x-1)+3) 寄存器
Nf_errlocation(4*(x-1)+4)_reg	0x0C4+0x10*(x-1)	错误定位(4*(x-1)+4) 寄存器
prot_space0_l	0x2D0	写保护地址空间段 0 下限低 32 位地址
prot_space0_h	0x2D4	写保护地址空间段 0 下限高 8 位地址
prot_space1_l	0x2D8	写保护地址空间段 0 上限低 32 位地址
prot_space1_h	0x2DC	写保护地址空间段 0 上限高 8 位地址
prot_space2_l	0x2E0	写保护地址空间段 1 下限低 32 位地址
prot_space2_h	0x2E4	写保护地址空间段 1 下限高 8 位地址
prot_space3_l	0x2E8	写保护地址空间段 1 上限低 32 位地址
prot_space3_h	0x2EC	写保护地址空间段 1 上限高 8 位地址

注意：表中 x 取值为 1~32。

5.17.3 寄存器说明

5.17.3.1 Nf_ctrl0_reg (0x000)

域	位	读写	复位值	描述
nf_spare_size	31:9	RW	0x0	spare_size 配置
nf_spare_size_en	8	RW	0x0	spare_size 使能位
ecc_correct	7:5	RW	0x7	ECC 纠错个数 3'h1: 2 位 3'h3: 4 位 3'h7: 8 位
ecc_enable	4	RW	0x1	ECC 使能位

域	位	读写	复位值	描述
nf_inter_mode	3:2	RW	0x0	NAND Flash 接口模式 2'b00: Asyn 2'b01: ONFI Syn 2'b10: Toggle Asyn
df_width	1	RW	0x0	DQ 信号位宽设置, 仅在 ONFI 异步模式下有效 0: 8 位宽 1: 16 位宽
nf_enable	0	RW	0x1	NAND flash 控制器使能位, 写 1 表示控制器打开

5.17.3.2 Nf_ctrl1_reg (0x004)

域	位	读写	复位值	描述
reserved	31:19	RW	0x0	保留
ecc_bypass	18	RW	0x1	当接收到 ecc 编码地址数据为 13'hff 时, ecc 校验功能 bypass, 1 表示该功能使能。
rb_share_en	17	RW	0x0	4 路设备共用 1 为 rb_n 信号使能, 写 1 为使能。
ecc_data_first_en	16	RW	0x1	先读入 ECC 数据, 然后读入对应数据。
sampl_phase	15:0	RW	0x2	异步模式下相位调节, 每增加 1, 增加 1.6ns。onfi 同步、toggle 模式下接收数据采样相位调节周期, 每增加 1, 增加 0.5ns。该值不能为 0。

5.17.3.3 Nf_maddr0_reg (0x008)

域	位	读写	复位值	描述
dt_addr0	31:0	RW	0x0	内存中存储的描述符表首地址的低 32 位

5.17.3.4 Nf_maddr1_reg (0x00c)

域	位	读写	复位值	描述
dma_wlens	31:24	RW	0x40	设置 dma 写数据时的 lens 长度。默认为 0x40, 单位为 4 字节, 即 lens 长度默认为 64*4 字节。
dma_rlens	23:16	RW	0x40	设置 dma 读数据时的 lens 长度
dma_empty	15:9	RW	0x0	保留
dma_en	8	WO	0x0	DMA 传输使能位, 值为 1 控制器开始进行 DMA 传输。
dt_addr1	7:0	RW	0x0	内存中存储的描述符表首地址的高 8 位

5.17.3.5 Nf_timing0_reg (0x010)

域	位	读写	复位值	描述
asy_timpara1	31:16	RW	0x3	tCLS-tWP
asy_timpara0	15:0	RW	0x3	tCS-tCLS

5.17.3.6 Nf_timing1_reg (0x014)

域	位	读写	复位值	描述
asy_timpara3	31:16	RW	0x28	tWH
asy_timpara2	15:0	RW	0x28	tWP

5.17.3.7 Nf_timing2_reg (0x018)

域	位	读写	复位值	描述
asy_timpara5	31:16	RW	0x3	tCLH-tWH
asy_timpara4	15:0	RW	0x3	tCH-tCLH

5.17.3.8 Nf_timing3_reg (0x01C)

域	位	读写	复位值	描述
asy_timpara7	31:16	RW	0x6	tCH-tWH
asy_timpara6	15:0	RW	0x6	tDQ_en, 自定义

5.17.3.9 Nf_timing4_reg (0x020)

域	位	读写	复位值	描述
asy_timpara9	31:16	RW	0x28	tREH
asy_timpara8	15:0	RW	0xa0	$tWHR - (s10 - s9) - (s3 + s4 + s5 + h'1)$, >120ns, //tCCS>200ns

5.17.3.10 Nf_timing5_reg (0x024)

域	位	读写	复位值	描述
asy_timpara11	31:16	RW	0x30	$tADL - (s2 + s3 + s4 + s5 + s6 - h'1)$, //tCCS> 200ns
asy_timpara10	15:0	RW	0x50	tRC

5.17.3.11 Nf_timing6_reg (0x028)

域	位	读写	复位值	描述
syn_timpara16	31:16	RW	0x20	tCALS+tCH(建议值 tCK)
syn_timpara1	15:0	RW	0x41	tCAD+tCS-s3-s5

5.17.3.12 Nf_timing7_reg (0x02C)

域	位	读写	复位值	描述
syn_timpara3	31:16	RW	0x05	tDQ_en, 自定义
syn_timpara2	15:0	RW	0x20	tCK

5.17.3.13 Nf_timing8_reg (0x030)

域	位	读写	复位值	描述
syn_timpara5	31:16	RW	0x10	0.5tCK
syn_timpara4	15:0	RW	0x19	tCAD-tCK-s3-h'2

5.17.3.14 Nf_timing9_reg (0x034)

域	位	读写	复位值	描述
syn_timpara7	31:16	RW	0x62	tCCS (n*tCK>500ns) -s3-s5-tCALS-h'2
syn_timpara6	15:0	RW	0x40	tWHR>80ns

5.17.3.15 Nf_timing10_reg (0x038)

域	位	读写	复位值	描述
syn_timpara9	31:16	RW	0x38	>1.5tCK, (n+0.5)tCK+tCH, n>=1
syn_timpara8	15:0	RW	0x20	tCK (0.75~1.25 倍)

5.17.3.16 Nf_timing11_reg (0x03C)

域	位	读写	复位值	描述
reserved	31:16	RW	0x00	保留
syn_timpara11	15:0	RW	0x09	tCK-tCALS

5.17.3.17 Nf_timing12_reg (0x040)

域	位	读写	复位值	描述
syn_timpara13	31:16	RW	0x50	tCKWR =n*tCK -tCALS (n>=2) (建议值 1.5tCK)
syn_timpara12	15:0	RW	0x20	tWRCK>20ns (n*tCK) (建议值 tCK)

5.17.3.18 Nf_timing13_reg (0x044)

域	位	读写	复位值	描述
tog_timpara11	31:16	RW	0x14	tWPST
tog_timpara10	15:0	RW	0x0a	tWPRE

5.17.3.19 Nf_timing14_reg (0x048)

域	位	读写	复位值	描述
tog_timpara1	31:16	RW	0x08	tCLS-tWP
tog_timpara0	15:0	RW	0x08	tCS-tCLS

5.17.3.20 Nf_timing15_reg (0x04C)

域	位	读写	复位值	描述
tog_timpara3	31:16	RW	0xc8	tWHR/tWHR2 (120/300ns)
tog_timpara2	15:0	RW	0xc8	tADL (300ns)

5.17.3.21 Nf_timing16_reg (0x050)

域	位	读写	复位值	描述
tog_timpara5	31:16	RW	0x08	tCLH-tWH
tog_timpara4	15:0	RW	0x08	tCH-tCLH

5.17.3.22 Nf_timing17_reg (0x054)

域	位	读写	复位值	描述
tog_timpara7	31:16	RW	0x20	tRPST tWPST (25ns+0.5tDSC)
tog_timpara6	15:0	RW	0x0a	tRPRE tWPRE

5.17.3.23 Nf_timing18_reg (0x058)

域	位	读写	复位值	描述
tog_timpara9	31:16	RW	0x14	tRPSTH
tog_timpara8	15:0	RW	0x08	0.5tDSC

5.17.3.24 Nf_fiforsta_reg (0x05C)

域	位	读写	复位值	描述
reserved	31:12	RO	0x0	保留
fifo_full_real	11	RO	0x0	FIFO 满数据
fifo_empty_real	10	RO	0x1	FIFO 无数据
fifo_count	9:0	RO	0x0	FIFO 当前的数据深度，一个深度是 4

5.17.3.25 Nf_interval_reg (0x060)

域	位	读写	复位值	描述
reserved	31:16	RO	0x0	保留
interval_timpara	15:0	RW	0x0	命令，地址，数据之间的时间间隔，写入值每增加 1，超时时间增加 2ns。

5.17.3.26 Nf_cmdintval_reg (0x064)

域	位	读写	复位值	描述
cmd_interval_param	31:0	RW	0x30	请求之间的时间间隔，写入值每增加 1，超时时间增加 2ns。

5.17.3.27 Nf_fifo_timeout_reg (0x068)

域	位	读写	复位值	描述
fifo_timeout_param	31:0	RW	0x800	FIFO 超时计数器，写入值每增加 1，超时时间增加 2ns。

5.17.3.28 Nf_fifo_level0_reg (0x06C)

域	位	读写	复位值	描述
reserved	31:4	RW	0x0	保留
fifo_level0_sel	3:0	RW	0x2	FIFO 阈值选择 4'b0000: FIFO 有数据 4'b0001: FIFO >= 1/8 full 4'b0010: FIFO >= 1/4 full 4'b0011: FIFO >= 3/8 full 4'b0100: FIFO >= 1/2 full 4'b0101: FIFO >= 5/8 full 4'b0110: FIFO >= 3/4 full 4'b0111: FIFO >= 7/8 full 4'b1000: FIFO 为满

5.17.3.29 Nf_fifo_level1_reg (0x070)

域	位	读写	复位值	描述
reserved	31:4	RW	0x0	保留
fifo_level1_sel	3:0	RW	0x1	FIFO 阈值选择 4'b0000: FIFO 无数据 4'b0001: FIFO <= 1/8 empty 4'b0010: FIFO <= 1/4 empty 4'b0011: FIFO <= 3/8 empty 4'b0100: FIFO <= 1/2 empty 4'b0101: FIFO <= 5/8 empty 4'b0110: FIFO <= 3/4 empty 4'b0111: FIFO <= 7/8 empty

5.17.3.30 Nf_wp_reg (0x074)

域	位	读写	复位值	描述
reserved	31:1	RW	0x0	保留
nf_wp	0	RW	0x1	NAND Flash 接口 WP 使能位。 0 表示写保护打开。

5.17.3.31 Nf_fifree_reg (0x078)

域	位	读写	复位值	描述
reserved	31:1	WO	0x0	保留
fifo_free	0	WO	0x0	清空 FIFO 操作，高有效。

5.17.3.32 Nf_state_reg (0x07C)

域	位	读写	复位值	描述
reserved	31:30	RO	0x0	保留
eccbypass_sta	29	RO	0x0	ecc_bypass 状态有效。
prot_err	28	RO	0x0	保护空间非法访问状态
rb_sta	27:24	RO	0xF	RB_N 接口的状态
axi_wr_err_sta	23	RO	0x0	axi 写错误
axi_rd_err_sta	22	RO	0x0	axi 读错误
axi_dsp_err_sta	21	RO	0x0	描述符错误
ecc_erover_sta	20	RO	0x0	错误超过可校验范围
ecc_error_sta	19	RO	0x0	ECC 校验有错
ecc_right_sta	18	RO	0x1	ECC 校验正确
ecc_finish_sta	17	RO	0x0	ECC 校验完成
ecc_busy_sta	16	RO	0x0	ECC 校验忙
reserved	15	RO	0x0	保留
dqs_gate_sta	14	RO	0x0	dqs 门控状态
re_gate_sta	13	RO	0x0	re_n 门控状态
page_finish_sta	12	RO	0x0	nand 接口数据操作完成（有 next_cmd 时，所有命令完成后，才显示）
cmd_finish_sta	11	RO	0x0	nand 接口命令操作完成（有 next_cmd 时，所有命令完成后，才显示）
cs_sta	10:7	RO	0xF	片选状态
fifo_timeout_sta	6	RO	0x0	fifo 超时
fifo_full_sta	5	RO	0x0	fifo 满
fifo_empty_sta	4	RO	0x1	fifo 空
dma_finish_sta	3	RO	0x0	dma 完成
dma_pgfinish_sta	2	RO	0x0	dma 数据操作完成（有 next_cmd 时，所有命令完成后，才显示）
dma_busy_sta	1	RO	0x0	dma 控制器忙
nf_busy_sta	0	RO	0x0	nandflash 控制器忙

5.17.3.33 Nf_intrmask_reg (0x080)

域	位	读写	复位值	描述
reserved	31:19	RW	0x0	保留
prot_err_mask	18	RW	0x1	保护空间非法访问中断屏蔽位
rb_state_mask	17:14	RW	0xF	rb_n 信号 busy 中断屏蔽位

域	位	读写	复位值	描述
ecc_right_mask	13	RW	0x1	错误信号中断屏蔽位
ecc_finish_mask	12	RW	0x1	ecc 完成中断屏蔽位
reserved	11	RW	0x0	保留
dqs_gate_mask	10	RW	0x1	dqs 门控打开中断屏蔽位
re_gate_mask	9	RW	0x1	re_n 门控打开中断屏蔽位
page_finish_mask	8	RW	0x1	nand 接口页操作完成中断屏蔽位
cmd_finish_mask	7	RW	0x1	nand 接口命令完成中断屏蔽位
fifo_timeout_mask	6	RW	0x1	fifo 超时中断屏蔽位
fifo_full_mask	5	RW	0x1	fifo 为满中断屏蔽位
fifo_empty_mask	4	RW	0x1	fifo 为空中断屏蔽位
dma_finish_mask	3	RW	0x0	dma 操作完成中断完成中断屏蔽位
dma_pgfinish_mask	2	RW	0x1	dma 页操作完成中断屏蔽位
dma_busy_mask	1	RW	0x1	dma 控制器忙状态中断屏蔽位
nf_busy_mask	0	RW	0x1	nandflash 控制器忙状态中断屏蔽位

5.17.3.34 Nf_intr_reg (0x084)

域	位	读写	复位值	描述
prot_err_intr	18	W1C	0x0	保护空间非法访问中断状态位，写 1 清除中断。
rb_state_intr	17:14	W1C	0x0	rb_n 信号 busy 中断状态位，写 1 清除中断。
ecc_right_intr	13	W1C	0x0	错误中断状态位 (ecc_err, ecc_over, dsp_err)，写 1 清除中断。
ecc_finish_intr	12	W1C	0x0	ecc 完成中断状态位，写 1 清除中断。
reserved	11	W1C	0x0	保留
dqs_gate_intr	10	W1C	0x0	dqs 门控打开中断状态位，写 1 清除中断。
re_gate_intr	9	W1C	0x0	re_n 门控打开中断状态位，写 1 清除中断。
page_finish_intr	8	W1C	0x0	nand 接口页操作完成中断状态位，写 1 清除中断。
cmd_finish_intr	7	W1C	0x0	nand 接口命令完成中断状态位，写 1 清除中断。
fifo_timeout_intr	6	W1C	0x0	fifo 超时中断状态位，写 1 清除中断。
fifo_full_intr	5	W1C	0x0	fifo 为满中断状态位，写 1 清除中断。
fifo_empty_intr	4	W1C	0x0	fifo 为空中断状态位，写 1 清除中断。
dma_finish_intr	3	W1C	0x0	dma 操作完成中断完成中断状态位，写 1 清除中断。
dma_pgfinish_intr	2	W1C	0x0	dma 页操作完成中断状态位，写 1 清除中断。
dma_busy_intr	1	W1C	0x0	dma 控制器忙状态中断状态位，写 1 清除中断。
nf_busy_intr	0	W1C	0x0	nandflash 控制器忙状态中断状态位，写 1 清除中断。

5.17.3.35 Nf_debug_reg (0x088)

域	位	读写	复位值	描述
reserved	31:12	RO	0x0	保留
decoder_fsm	11:8	RO	0x0	校验的当前状态
main_fsm	7:4	RO	0x0	指令传输的当前状态

域	位	读写	复位值	描述
dam_fsm	3:0	R0	0x0	DMA 传输的当前状态

5.17.3.36 Nf_errclr_reg (0x08C)

域	位	读写	复位值	描述
reserved	31:5	W1-1	0x0	保留
prot_err_clr	4	W1-1	0x0	保护地址非法访问错误清除
ecc_error_clr	3	W1-1	0x0	ecc 错误清除
axi_wr_err_clr	2	W1-1	0x0	axi 写错误清除
axi_rd_err_clr	1	W1-1	0x0	axi 读错误清除
dsp_err_clr	0	W1-1	0x0	描述符错误清除

5.17.3.37 Nf_dmardcnt_reg (0x090)

域	位	读写	复位值	描述
dma_rd_cnt	31:0	R0	0x0	dma 读页操作当前的剩余数据

5.17.3.38 Nf_dmardsparcnt_reg (0x094)

域	位	读写	复位值	描述
dma_rd_spar_cnt	31:0	R0	0x0	dma 读 spar 操作当前的剩余数据

5.17.3.39 Nf_dmawrcnt_reg (0x098)

域	位	读写	复位值	描述
dma_wr_cnt	31:0	R0	0x0	dma 写页操作当前的剩余数据

5.17.3.40 Nf_dmawrsparcnt_reg (0x09C)

域	位	读写	复位值	描述
dma_wr_spar_cnt	31:0	R0	0x0	dma 写 spar 操作当前的剩余数据

5.17.3.41 Nf_intwrcnt_reg (0x0A0)

域	位	读写	复位值	描述
int_wr_data_cnt	31:0	R0	0x0	nand 接口写操作当前剩余数据

5.17.3.42 Nf_intwrcnt_reg (0x0A4)

域	位	读写	复位值	描述
int_asy_rd_data_cnt	31:0	R0	0x0	nand 接口异步读操作当前剩余数据

5.17.3.43 Nf_intwrcnt_reg (0x0A8)

域	位	读写	复位值	描述
int_syn_tog_rd_data_cnt	31:0	R0	0x0	nand 接口同步写或 tog 读操作当前剩余数据

5.17.3.44 Nf_encodefincnt_reg (0x0AC)

域	位	读写	复位值	描述
reserved	31:8	R0	0x0	保留
int_encode_finish_count	7:0	R0	0x0	一次请求中的 ECC 编码完成次数计数

5.17.3.45 Nf_encodedatcnt_reg (0x0B0)

域	位	读写	复位值	描述
reserved	31:8	R0	0x0	保留
int_encode_data_cnt	7:0	R0	0x0	ECC 编码发送计数器

5.17.3.46 Nf_decodefincnt_reg (0x0B4)

域	位	读写	复位值	描述
reserved	31:17	R0	0x0	保留
int_decoder_finish_cnt	16:0	R0	0x0	一次请求中的 ECC 校验完成计数器

5.17.3.47 Nf_errlocation (4*(x-1)+1)_reg

域	位	读写	复位值	描述
int_err_location (4*(x-1)+1)	31:0	R0	0x0	第 1+(x-1)/4 个 512B 字节的错误地址： 31:16: 第 2 个错误地址 15:0: 第 1 个错误地址 注：为 0 表示没有错误。 x 取值为 1~32。

5.17.3.48 Nf_errlocation (4*(x-1)+2)_reg

域	位	读写	复位值	描述
int_err_location (4*(x-1)+2)	31:0	R0	0x0	第 1+(x-1)/4 个 512B 字节的错误地址： 31:16: 第 4 个错误地址 15:0 : 第 3 个错误地址 注：纠错能力为 4/8 的时候此寄存器有效。 x 取值为 1~32。

5.17.3.49 Nf_err_location(4*(x-1)+3)_reg

域	位	读写	复位值	描述
int_err_location (4*(x-1)+3)	31:0	RO	0x0	第 1+(x-1) /4 个 512B 字节的错误地址： 31:16：第 6 个错误地址 15:0：第 5 个错误地址 注：纠错能力为 8 的时候此寄存器有效。 x 取值为 1~32。

5.17.3.50 Nf_err_location (4*(x-1)+4)_reg

域	位	读写	复位值	描述
int_err_location (4*(x-1)+4)	31:0	RO	0x0	第 1+(x-1) /4 个 512B 字节的错误地址： 31:16：第 8 个错误地址 15:0：第 7 个错误地址 注：纠错能力为 8 的时候此寄存器有效。 x 取值为 1~32。

5.17.3.51 prot_space0_l (0x2D0)

域	位	读写	复位值	描述
prot_addr0_l	31:0	RW	0x0	写保护地址空间段 0 下限的低 32 位地址，先写 32'5555_6666，再写入保护空间的低 32 位地址，写保护空间地址有效。写 32'h7777_8888，再写 32'h1234_5678，解锁写保护空间，可以正常读写。

5.17.3.52 prot_space0_h (0x2D4)

域	位	读写	复位值	描述
reserved	31:8	RW	0x0	保留
prot_addr0_h	7:0	RW	0x0	写保护地址空间段 0 下限的高 8 位地址，操作流程先写高位寄存器，再写低位寄存器。

5.17.3.53 prot_space1_l (0x2D8)

域	位	读写	复位值	描述
prot_addr1_l	31:0	RW	0x0	写保护地址空间段 0 上限的低 32 位地址，先写 32'5555_6666，再写入保护空间的低 32 位地址，写保护空间地址有效。写 32'h7777_8888，再写 32'h1234_5678，解锁写保护空间，可以正常读写。

5.17.3.54 prot_space1_h (0x2DC)

域	位	读写	复位值	描述
reserved	31:8	RW	0x0	保留
prot_addr1_h	7:0	RW	0x0	写保护地址空间段 0 上限的高 8 位地址，操作流程先写高位寄存器，再写低位寄存器。

5.17.3.55 prot_space2_l (0x2E0)

域	位	读写	复位值	描述
prot_addr2_l	31:0	RW	0x0	写保护地址空间段 1 下限的低 32 位地址，先写 32'h5555_6666，再写入保护空间的低 32 位地址，写保护空间地址有效。写 32'h7777_8888，再写 32'h1234_5678，解锁写保护空间，可以正常读写。

5.17.3.56 prot_space2_h (0x2E4)

域	位	读写	复位值	描述
reserved	31:8	RW	0x0	保留
prot_addr2_h	7:0	RW	0x0	写保护地址空间段 1 下限的高 8 位地址，操作流程先写高位寄存器，再写低位寄存器。

5.17.3.57 prot_space3_l (0x2E8)

域	位	读写	复位值	描述
prot_addr3_l	31:0	RW	0x0	写保护地址空间段 1 上限的低 32 位地址，先写 32'h5555_6666，再写入保护空间的低 32 位地址，写保护空间地址有效。写 32'h7777_8888，再写 32'h1234_5678，解锁写保护空间，可以正常读写。

5.17.3.58 prot_space3_h (0x2EC)

域	位	读写	复位值	描述
reserved	31:8	RW	0x0	保留
prot_addr3_h	7:0	RW	0x0	写保护地址空间段 3 上限的高 8 位地址，操作流程先写高位寄存器，再写低位寄存器。

5.18 I2S 控制器

I2S 是针对数字音频设备之间的音频数据传输而制定的一种总线标准，I2S 控制器主要实现音频数据的发送与接收。

5.18.1 操作说明

以 Channel0 配置为例。

- Transmitter 模式

1. 使能整个 I2S 控制器，即写 IER 寄存器为 1；
2. 通过 APB 接口，向 LTHR 和 RTHR 寄存器中写入将要发送的数据，填满整个 FIFO（超过阈值，阈值通过 RFCR0 和 TFCR0 两个寄存器设置）；
3. 设置 TCR0 寄存器，设定发送数据的 resolution；
4. 设定中断使能开关；
5. 设定 ITER 为 1，使能 I2S 控制器的发送功能；
6. 设定 TER0 为 1，将该通道的发送使能打开，开始工作。

- Receiver 模式

1. 使能整个 I2S 控制器，即写 IER 寄存器为 1；
2. 设置 RCR0 寄存器，设定接收数据的 resolution；
3. 设定中断使能开关；
4. 设定 IRER 为 1，使能 I2S 控制器的接收功能；
5. 使能该通道的接收模块，即写 RER0 寄存器为 1；
6. 等待中断到来，即接收数据 FIFO 到达默认的 trigger 值；
7. 通过 APB 接口，读取 LRBR 和 RRBR 两个寄存器，将收到的数据读出。

- I2S 控制器的 DMA 模式

若 I2S 控制器需要工作在 DMA 模式下，基本流程如下：

1. 写全局配置寄存器，将带描述符列表的 DMA 和 I2S 控制器跳出复位；
2. I2S 控制器主时钟包括 clk_nand(600MHz) 和 clk_sio(50MHz)，输出给设备的时钟信号由 clk_nand(600MHz) 分频得到。通过配置 FRAC_DIV, EVEN_DIV 寄存器实现 MCLK 和 SCLK 的分频；
3. 使能 I2S 发送或者传输模块；
4. 配置 DMA 通道相关参数，具体参考 DMA 设计文档；
5. 使能 I2S 时钟信号，使得多通道 I2S 能同时发送。

5.18.2 寄存器列表

表 5-54 I2S 寄存器基地址

名称	基地址
I2S	0x000_2800_9000

表 5-55 I2S 寄存器列表

寄存器名称	偏移	描述
IER	0x000	控制使能全局
IRER	0x004	控制接收模块使能
ITER	0x008	控制发送模块使能
CER	0x00C	全局时钟使能控制
CCR	0x010	时钟配置寄存器
RXFFR	0x014	接收 FIFO 清除寄存器
TXRRF	0x018	发送 FIFO 清除寄存器
LRBR _x	0x020+x*0x40	提供给上层接口的读取接收的左声道数据的寄存器
LTHR _x	0x020+x*0x40	提供给上层接口的写入发送的左声道数据的寄存器
RRBR _x	0x024+x*0x40	提供给上层接口的读取接收的右声道数据的寄存器
RTHR _x	0x024+x*0x40	提供给上层接口的写入发送的右声道数据的寄存器
RER _x	0x028+x*0x40	接收通道 x 使能控制位
TER _x	0x02C+x*0x40	发送通道 x 使能控制位
RGR _x	0x030+x*0x40	控制接收数据的 resolution, 在控制信号使能情况下, 写无效
TCR _x	0x034+x*0x40	控制发送数据的 resolution, 在控制信号使能情况下, 写无效
ISR _x	0x038+x*0x40	中断状态寄存器
IMR _x	0x03C+x*0x40	中断掩码寄存器
ROR _x	0x040+x*0x40	接收溢出状态寄存器, 读取该位, 就会清除接收数据溢出中断, 同时清除状态寄存器相关位
TOR _x	0x044+x*0x40	发送溢出状态寄存器, 读取该位, 就会清除发送数据溢出中断, 同时清除状态寄存器相关位
RF _{CR} _x	0x048+x*0x40	设置接收 FIFO 产生满中断的阈值, 高于该阈值, 触发中断; 在控制信号使能情况下, 写该寄存器无效
TF _{CR} _x	0x04C+x*0x40	设置发送 FIFO 产生空中断的阈值, 低于该阈值, 触发中断; 在控制信号都处于使能情况下, 写该寄存器无效
RFF _x	0x050+x*0x40	写 1 清除该通道接收 FIFO
TFF _x	0x054+x*0x40	写 1 清除该通道发送 FIFO
RXDMA	0x01C0	读 DMA 数据寄存器
RRXDMA	0x01C4	读 DMA 复位寄存器
TXDMA	0x01C8	写 DMA 数据寄存器
RTXDMA	0x01CC	写 DMA 复位寄存器
FRAC_DIV	0x0C00	小数分频寄存器
EVEN_DIV	0x0C04	I2S 二次分频参数寄存器

注意：表中 x 的取值范围为 0~3。

5.18.3 寄存器说明

章节 5.17.3.8~5.17.3.23 的寄存器偏移地址中 x 表示通道号, 取值范围均为 0~3。

5.18.3.1 IER (0x0000)

域	位	读写	复位值	描述
reserved	31:1	RO	0x0	保留
IEN	0	RW	0x0	I2S 模块使能：1 为使能；0 为关闭。

5.18.3.2 IRER (0x0004)

域	位	读写	复位值	描述
reserved	31:1	RO	0x0	保留
RXEN	0	RW	0x0	接收模块使能：1 为使能；0 为关闭。

5.18.3.3 ITER (0x0008)

域	位	读写	复位值	描述
reserved	31:1	RO	0x0	保留
TXEN	0	RW	0x0	发送模块使能：1 为使能；0 为关闭。

5.18.3.4 CER (0x000C)

域	位	读写	复位值	描述
reserved	31:1	RO	0x0	保留
CLKEN	0	RW	0x0	时钟生成使能：1 为使能；0 为关闭。

5.18.3.5 CCR (0x0010)

域	位	读写	复位值	描述
reserved	31:5	RO	0x0	保留
WSS	4:3	RW	0x0	WS 对应 SCLK 的个数。 00: 16 个 01: 24 个 10: 32 个
SCLKG	2:0	RW	0x0	时钟门控信号。 000: 无时钟门控； 001: 12 周期后门控； 010: 16 周期后门控； 011: 20 周期后门控； 100: 24 周期后门控。

5.18.3.6 RXFFR (0x0014)

域	位	读写	复位值	描述
reserved	31:1	RO	0x0	保留
RXFFR	0	RW	0x0	接收队列复位：写 1 复位；自动归 0。

5.18.3.7 TXRRF (0x0018)

域	位	读写	复位值	描述
reserved	31:1	RO	0x0	保留
TXFFR	0	RW	0x0	发送队列复位：写 1 复位；自动归 1。

5.18.3.8 LRBR_x (0x0020+x*0x40)

域	位	读写	复位值	描述
LRBR	31:0	RO	0x0	左接收缓存

5.18.3.9 LTHR_x (0x0020+x*0x40)

域	位	读写	复位值	描述
LTHR	31:0	WO	0x0	左发送缓存

5.18.3.10 RRBR_x (0x0024+x*0x40)

域	位	读写	复位值	描述
RRBR	31:0	RO	0x0	右接收缓存

5.18.3.11 RTHR_x (0x0024+x*0x40)

域	位	读写	复位值	描述
RTHR	31:0	WO	0x0	右发送缓存

5.18.3.12 RER_x (0x0028+x*0x40)

域	位	读写	复位值	描述
reserved	31:1	RO	0x0	保留
RXCHEN	0	RW	0x1	接收通道使能：1 为使能；0 为关闭。

5.18.3.13 TER_x (0x002C+x*0x40)

域	位	读写	复位值	描述
reserved	31:1	RO	0x0	保留
TXCHEN	0	RW	0x1	发送通道使能：1 为使能；0 为关闭。

5.18.3.14 RCR_x (0x0030+x*0x40)

域	位	读写	复位值	描述
reserved	31:3	RO	0x0	保留
WLEN	2:0	RW	0x5	字长配置。 000：忽略字长； 001：12 位；

域	位	读写	复位值	描述
				010: 16 位; 011: 20 位; 100: 24 位; 101: 32 位。

5.18.3.15 TCR_x (0x0034+x*0x40)

域	位	读写	复位值	描述
reserved	31:3	RO	0x0	保留
WLEN	2:0	RW	0x5	字长配置。 000: 忽略字长; 001: 12 位; 010: 16 位; 011: 20 位; 100: 24 位; 101: 32 位。

5.18.3.16 ISR_x (0x0038+x*0x40)

域	位	读写	复位值	描述
reserved	31:6	RO	0x0	保留
TXF0	5	RO	0x0	发送溢出。0 为有效; 1 为溢出。
TXFE	4	RO	0x1	发送空闲。0 为未空闲; 1 为空闲。
reserved	3:2	RO	0x0	保留
RXF0	1	RO	0x0	接收溢出。0 为有效; 1 为溢出。
RXFE	0	RO	0x0	接收可搬出。0 为不可搬出; 1 为可搬出。

5.18.3.17 IMR_x (0x003C+x*0x40)

域	位	读写	复位值	描述
reserved	31:6	RO	0x0	保留
TXFOM	5	RW	0x1	发送溢出屏蔽。0 为不屏蔽; 1 为屏蔽。
TXFEM	4	RW	0x1	发送空闲屏蔽。0 为不屏蔽; 1 为屏蔽。
reserved	3:2	RO	0x0	保留
RXFOM	1	RW	0x1	接收溢出屏蔽。0 为不屏蔽; 1 为屏蔽。
RXFEM	0	RW	0x1	接收可搬出屏蔽。0 为不屏蔽; 1 为屏蔽。

5.18.3.18 ROR_x (0x0040+x*0x40)

域	位	读写	复位值	描述
reserved	31:1	RO	0x0	保留
RXCH0	0	RO	0x0	接收是否溢出并清除接收相关中断。0 为有效; 1 为溢出。

5.18.3.19 TOR_x (0x0044+x*0x40)

域	位	读写	复位值	描述
reserved	31:1	RO	0x0	保留

域	位	读写	复位值	描述
TXCHO	0	RO	0x0	发送是否溢出并清除发送相关中断。0 为有效；1 为溢出。

5.18.3.20 RFCR_x (0x0048+x*0x40)

域	位	读写	复位值	描述
reserved	31:4	RO	0x0	保留
RXCHDT	3:0	RW	0x3	接收通道可搬出触发等级。

5.18.3.21 TFCR_x (0x004C+x*0x40)

域	位	读写	复位值	描述
reserved	31:4	RO	0x0	保留
TXCHET	3:0	RW	0x3	发送通道空闲触发等级。

5.18.3.22 RFF_x (0x0050+x*0x40)

域	位	读写	复位值	描述
reserved	31:1	RO	0x0	保留
RXCHFR	0	WO	0x0	接收通道复位。写 1 复位；自动归 0。

5.18.3.23 TFF_x (0x0054+x*0x40)

域	位	读写	复位值	描述
reserved	31:1	RO	0x0	保留
TXCHFR	0	WO	0x0	发送通道复位。写 1 复位；自动归 1。

5.18.3.24 RXDMA (0x01C0)

域	位	读写	复位值	描述
RXDMA	31:0	RO	0x0	供 DMA 控制器读出数据

5.18.3.25 RRXDMA (0x01C4)

域	位	读写	复位值	描述
reserved	31:1	RO	0x0	保留
RRXDMA	0	WO	0x0	复位接收 DMA 入口（通道归 0）

5.18.3.26 TXDMA (0x01C8)

域	位	读写	复位值	描述
TXDMA	31:0	RO	0x0	供 DMA 控制器写入数据

5.18.3.27 RTXDMA (0x01CC)

域	位	读写	复位值	描述
reserved	31:1	RO	0x0	保留
RTXDMA	0	WO	0x0	复位发送 DMA 入口 (通道归 0)

5.18.3.28 FRAC_DIV (0x0C00)

域	位	读写	复位值	描述
reserved	31:24	RO	0x0	保留
INT_DIV	23:16	RW	0x0	小数分频的整数部分, 最大为 255。
FRAC_DIV	15:0	RW	0x0	小数分频的小数部分, 最大为 9999。

5.18.3.29 EVEN_DIV (0x0C04)

域	位	读写	复位值	描述
EVEN_DIV	31:0	RW	0x0	偶数分频, 进行此寄存器+1 倍的偶数分频。

5.19 CAN 控制器

CAN 控制器是 CAN 总线系统中的一个节点, 用于收发 CAN 总线数据。发送数据时将并行数据生成 CAN 帧, 以二进制码流的方式进行发送, 在此过程中进行位填充、添加 crc 校验、应答检测等操作; 接收数据帧时, 将收到的二进制码流进行解析并接收, 在此过程中进行收发数据对比、去位填充、执行 crc 校验位等操作; 此外还要进行仲裁以及错误处理等操作。飞腾派集成的 CAN 控制器兼容 CAN2.0 标准协议和 ISO 11898-1 (2015) CAN FD 标准协议。

5.19.1 操作说明

5.19.1.1 传输初始化

1. 配置 CAN_CTRL[0] 为 0, 传输不使能;
2. 配置 CAN_CTRL[7] 为 1, 复位内部状态;
(3~7 次序不分先后)
3. 配置 CAN_CTRL 选择传输模式, CAN_CTRL[11] 为 0 表示 CAN 模式, 为 1 表示 CAN_FD 模式;
4. CAN 控制器主时钟为 200MHz, 配置 CAN_ARB_RATE_CTRL 和 CAN_DAT_RATE_CTRL 寄存器, 设置传输速率;
5. 配置 CAN_ACC_ID0/1/2/3 寄存器和 CAN_ACC_ID0/1/2/3_MASK 寄存器, 设置接收

ID;

6. 如果需要发送数据, 向 CAN_TX_FIFO 写入数据;
7. 配置 CAN_INTR 寄存器, 选择使能的中断类型;
8. 配置 CAN_CTRL[0] 为 1, 传输使能。

5.19.1.2 协同工作

CAN 模式和 CAN FD 模式操作相同:

1. CPU/系统其他 Master 通过中断或读 CAN_FIFO_CNT 判断缓存空间状态;
2. 通过访问 CAN_RX_FIFO 或 CAN_TX_FIFO 读写数据。

5.19.1.3 终止传输

CAN 模式和 CAN FD 模式操作相同:

1. 如果 TX_FIFO 不为空, 想要停止发送数据, 需要设置 CAN_CTRL[7] 为 1, 设置之后检查 CAN_XFER_STS[8], 如果值为 0, 表示停止发送成功;
2. 如果要停止传输, 需要设置 CAN_CTRL[0] 为 0, 等到当前一笔数据传输完成以后, CAN_XFER_STS[10] 的值为 IDLE, 表示传输停止。

5.19.1.4 复位

CAN 控制器有两种复位类型:

1. 硬件复位 rst_n, 即全局复位;
2. 软复位, 通过向 CAN_CTRL(0x00) 的 RST 位写 1 来实现, 可实现对状态机以及寄存器进行复位。

5.19.2 寄存器列表

表 5-56 CAN 寄存器基地址

名称	基地址
CAN0	0x000_2800_A000
CAN1	0x000_2800_B000

表 5-57 CAN 寄存器列表

寄存器名称	偏移	描述
CAN_CTRL	0x000	全局控制寄存器
CAN_INTR	0x004	中断寄存器
CAN_ARB_RATE_CTRL	0x008	仲裁段速率控制寄存器
CAN_DAT_RATE_CTRL	0x00C	数据段速率控制寄存器
CAN_ACC_IDx	0x010+x*0x4	可接收识别符 x 寄存器, x 取值为 0~3

寄存器名称	偏移	描述
CAN_ACC_ID _x _MASK	0x020+ <i>x</i> *0x4	可接收识别符 <i>x</i> 掩码寄存器, <i>x</i> 取值为 0~3
CAN_XFER_STS	0x030	传输状态寄存器
CAN_ERR_CNT	0x034	错误计数寄存器
CAN_FIFO_CNT	0x038	FIFO 计数寄存器
CAN_DMA_CTRL	0x03C	DMA 控制寄存器
CAN_XFER_EN	0x040	传输使能寄存器
CAN_INTR1	0x044	中断寄存器 1
CAN_FRM_INFO	0x048	接收 FIFO 中有效帧信息寄存器
CAN_TIME_OUT	0x04C	接收 FIFO 读取数据超时阈值寄存器
CAN_TIME_OUT_CNT	0x050	接收 FIFO 读取数据超时计数寄存器
CAN_INTR2	0x054	中断寄存器 2
CAN_TX_FIFO	0x100~0x1FF	发送 FIFO 影子寄存器
CAN_RX_FIFO	0x200~0x2FF	接收 FIFO 影子寄存器
CAN_RX_INFO_FIFO	0x300~0x3FF	当前帧状态寄存器

5.19.3 寄存器说明

5.19.3.1 CAN_CTRL (0x00)

域	位	读写	复位值	描述
reserved	31:12	RO	0x0	保留
FDCRC	11	RW	0x0	配置 CAN 控制器 CRC 校验模式 0: CAN 校验模式 1: ISO CAN FD 校验模式
IOF	10	RW	0x0	配置 CAN 控制器过载帧是否发送 0: 接收 FIFO 满时发送过载帧 1: 不发送过载帧
RFEDT	9	RW	0x0	配置 CAN 控制器在发送帧时是否产生接收帧完成中断 0: 不产生接收帧完成中断 1: 产生接收帧完成中断
RFEIDF	8	RW	0x0	配置 CAN 控制器在过滤帧时是否产生接收帧完成中断 0: 当帧 ID 匹配时产生接收帧完成中断 1: 当帧 ID 不匹配时不产生接收帧完成中断
RST	7	RW	0x0	软复位 0: 不产生软复位信号 1: 产生软复位信号, CAN 控制器中的 can_clk 时钟域逻辑复位
reserved	6	RO	0x0	保留
TTS	5	RW	0x0	配置 CAN 控制器发送帧模式 0: 发送 FIFO 收到完整的一帧后进行发送, 可以预防发送 FIFO 空的情况, 发送帧时更安全 1: 发送 FIFO 有数据后进行发送, 可能会造成发送 FIFO 下溢, 发送帧时更快

域	位	读写	复位值	描述
reserved	4:3	R0	0x0	保留
AIME	2	RW	0x0	配置 CAN 控制器接收 ID 掩码使能 0: 不使能 1: 使能
TXREQ	1	RW	0x0	配置 CAN 控制器接收发送模式 0: 监听模式 1: 正常模式
XFER	0	RW	0x0	配置 CAN 控制器传输使能 0: 不使能 1: 使能

5.19.3.2 CAN_INTR (0x04)

域	位	读写	复位值	描述
ERIS	31	R0	0x0	错误中断原始状态 0: 不生效 1: 生效
TERIS	30	R0	0x0	发送帧完成中断原始状态 0: 不生效 1: 生效
RERIS	29	R0	0x0	接收完成中断原始状态 0: 不生效 1: 生效
TFRIS	28	R0	0x0	发送 FIFO 空中断原始状态 0: 不生效 1: 生效
RFRIS	27	R0	0x0	接收 FIFO 满中断原始状态 0: 不生效 1: 生效
PERIS	26	R0	0x0	隐性错误中断原始状态 0: 不生效 1: 生效
PWRIS	25	R0	0x0	隐性警告中断原始状态 0: 不生效 1: 生效
BORIS	24	R0	0x0	总线挂起中断原始状态 0: 不生效 1: 生效
EIC	23	WO	0x0	错误中断清除 0: 不清除 1: 清除
TEIC	22	WO	0x0	发送帧完成中断清除 0: 不清除 1: 清除
REIC	21	WO	0x0	接收完成中断清除 0: 不清除 1: 清除
TFIC	20	WO	0x0	发送 FIFO 空中断清除 0: 不清除 1: 清除
RFIC	19	WO	0x0	接收 FIFO 满中断清除 0: 不清除 1: 清除
PEIC	18	WO	0x0	隐性错误中断清除 0: 不清除 1: 清除
PWIC	17	WO	0x0	隐性警告中断清除 0: 不清除 1: 清除

域	位	读写	复位值	描述
BOIC	16	WO	0x0	总线挂起中断清除 0: 不清除 1: 清除
EIE	15	RW	0x0	错误中断使能 0: 不使能 1: 使能
TEIE	14	RW	0x0	发送帧完成中断使能 0: 不使能 1: 使能
REIE	13	RW	0x0	接收完成中断使能 0: 不使能 1: 使能
TFIE	12	RW	0x0	发送 FIFO 空中断使能 0: 不使能 1: 使能
RFIE	11	RW	0x0	接收 FIFO 满中断使能 0: 不使能 1: 使能
PEIE	10	RW	0x0	隐性错误中断使能 0: 不使能 1: 使能
PWIE	9	RW	0x0	隐性警告中断使能 0: 不使能 1: 使能
BOIE	8	RW	0x0	总线挂起中断使能 0: 不使能 1: 使能
EIS	7	RO	0x0	错误中断状态 0: 不生效 1: 生效
TEIS	6	RO	0x0	发送帧完成中断状态 0: 不生效 1: 生效
REIS	5	RO	0x0	接收完成中断状态 0: 不生效 1: 生效
TFIS	4	RO	0x0	发送 FIFO 空中断状态 0: 不生效 1: 生效
RFIS	3	RO	0x0	接收 FIFO 满中断状态 0: 不生效 1: 生效
PEIS	2	RO	0x0	隐性错误中断状态 0: 不生效 1: 生效
PWIS	1	RO	0x0	隐性警告中断状态 0: 不生效 1: 生效
BOIS	0	RO	0x0	总线挂起中断状态 0: 不生效 1: 生效

5.19.3.3 CAN_ARB_RATE_CTRL (0x08)

域	位	读写	复位值	描述
reserved	31:29	RO	0x0	保留
APD	28:16	RW	0x0	仲裁场分频器。0~8191 表示分频范围 1~8192
reserved	15:11	RO	0x0	保留
APH2S	10:8	RW	0x0	仲裁场相位 2 段宽度。0~7 表示长度范围 1~8
APH1S	7:5	RW	0x0	仲裁场相位 1 段宽度。0~7 表示长度范围 1~8
APRS	4:2	RW	0x0	仲裁场传播段宽度。0~7 表示长度范围 1~8

域	位	读写	复位值	描述
ARJW	1:0	RW	0x0	仲裁场同步跳转宽度。0~3 表示宽度范围 1~4

5.19.3.4 CAN_DAT_RATE_CTRL (0x0C)

域	位	读写	复位值	描述
reserved	31:29	R0	0x0	保留
DPD	28:16	RW	0x0	数据场分频器。0~8191 表示分频范围 1~8192
reserved	15:11	R0	0x0	保留
DPH2S	10:8	RW	0x0	数据场相位 2 段宽度。0~7 表示长度范围 1~8
DPH1S	7:5	RW	0x0	数据场相位 1 段宽度。0~7 表示长度范围 1~8
DPRS	4:2	RW	0x0	数据场传播段宽度。0~7 表示长度范围 1~8
DRJW	1:0	RW	0x0	数据场同步跳转宽度。0~3 表示宽度范围 1~4

5.19.3.5 CAN_ACC_ID_x (0x10+x*0x4)

x 取值为 0~3。

域	位	读写	复位值	描述
reserved	31:29	R0	0x0	保留
AID _x	28:0	RW	0x0	配置 CAN 控制器可接收的帧 ID 值，只有 ID 值匹配上的帧才会被写入接收 FIFO。

5.19.3.6 CAN_ACC_ID_x_MASK (0x20+x*0x4)

x 取值为 0~3。

域	位	读写	复位值	描述
reserved	31:29	R0	0x0	保留
AID _x M	28:0	RW	0x0	配置 CAN 控制器接收 ID 掩码，如果与对应可接收的 ID 位的位值为 1，则忽略此位与接收帧对应位 ID 的匹配情况。

5.19.3.7 CAN_XFER_STS (0x30)

域	位	读写	复位值	描述
reserved	31:11	R0	0x0	保留
XFERS	10	R0	0x0	传输状态 0: 空闲 1: 繁忙
RS	9	R0	0x0	接收状态 0: 未接收 1: 正在接收
TS	8	R0	0x0	发送状态 0: 未发送 1: 正在发送
FIES	7:3	R0	0x0	CAN 控制器状态机当前状态 00000: 处于总线空闲状态 00001: 处于仲裁场状态

域	位	读写	复位值	描述
				00010: 处于发送控制场状态 00011: 处于发送数据场状态 00100: 处于发送 CRC 校验场状态 00101: 处于发送帧完成状态 00110: 处于接收控制场状态 00111: 处于接收控制场状态 01000: 处于接收 CRC 校验场状态 01001: 处于接收帧完成状态 01010: 处于帧间隔状态 01011: 处于错误被动发送延迟状态 01100: 处于总线空闲状态（等到一个显性电平即可恢复工作） 01101: 处于过载帧发送状态 01110: 处于过载帧发送标志完成状态 01111: 处于错误帧发送状态 10000: 处于错误帧发送标志完成状态 10001: 处于总线挂起状态 其余值无意义。
FRAS	2:0	R0	0x0	帧标志 000: 数据帧 001: 远程帧 010: 错误帧 011: 过载帧 100: 帧间隔

5.19.3.8 CAN_ERR_CNT (0x34)

域	位	读写	复位值	描述
reserved	31:25	R0	0x0	保留
TEC	24:16	R0	0x0	CAN 控制器发送错误计数器，错误计数范围 0~256。
reserved	15:9	R0	0x0	保留
REC	8:0	R0	0x0	CAN 控制器接收错误计数器，错误计数范围 0~256。

5.19.3.9 CAN_FIFO_CNT (0x38)

域	位	读写	复位值	描述
reserved	31:23	R0	0x0	保留
TFN	22:16	R0	0x0	CAN 控制器发送 FIFO 中有效数据个数，数据个数范围 0~64。
reserved	15:7	R0	0x0	保留
RFN	6:0	R0	0x0	CAN 控制器接收 FIFO 中有效数据个数，数据个数范围 0~64。

5.19.3.10 CAN_DMA_CTRL (0x3C)

域	位	读写	复位值	描述
reserved	31:23	R0	0x0	保留
TFRE	22	RW	0x0	CAN 控制器发送端发送 DMA 请求使能 0: 不使能 1: 使能
TFTH	21:16	RW	0x0	发送 FIFO 发送 DMA 的请求使能阈值, 阈值范围 0~64。
reserved	15:9	R0	0x0	保留
RFRE	6	RW	0x0	CAN 控制器接收端发送 DMA 请求使能 0: 不使能 1: 使能
RFTH	5:0	RW	0x0	接收 FIFO 发送 DMA 的请求使能阈值, 阈值范围 0~64。

5.19.3.11 CAN_XFER_EN (0x40)

域	位	读写	复位值	描述
reserved	31:1	R0	0x0	保留
XFER	0	RW	0x0	配置 CAN 控制器传输使能 0: 不使能 1: 使能

5.19.3.12 CAN_INTR1 (0x44)

域	位	读写	复位值	描述
TF0IS	31	R0	0x0	发送 FIFO 空中断原始状态 0: 不生效 1: 生效
TF1IS	30	R0	0x0	发送 FIFO 空间大小为 1/4 时中断原始状态 0: 不生效 1: 生效
TF2IS	29	R0	0x0	发送 FIFO 空间大小为 1/2 时中断原始状态 0: 不生效 1: 生效
TF3IS	28	R0	0x0	发送 FIFO 空间大小为 3/4 时中断原始状态 0: 不生效 1: 生效
RF4IS	27	R0	0x0	接收 FIFO 满中断原始状态 0: 不生效 1: 生效
RF3IS	26	R0	0x0	接收 FIFO 空间大小为 1/4 时中断原始状态 0: 不生效 1: 生效
RF2IS	25	R0	0x0	接收 FIFO 空间大小为 1/2 时中断原始状态 0: 不生效 1: 生效
RF1IS	24	R0	0x0	接收 FIFO 空间大小为 3/4 时中断原始状态 0: 不生效 1: 生效
TF0IS	23	W0	0x0	发送 FIFO 空中断清零 0: 不清零 1: 清零
TF1IS	22	W0	0x0	发送 FIFO 空间大小为 1/4 时中断清零 0: 不清零 1: 清零
TF2IS	21	W0	0x0	发送 FIFO 空间大小为 1/2 时中断清零 0: 不清零 1: 清零

域	位	读写	复位值	描述
TF3IS	20	WO	0x0	发送 FIFO 空间大小为 3/4 时中断清零 0: 不清零 1: 清零
RF4IS	19	WO	0x0	接收 FIFO 满中断清零 0: 不清零 1: 清零
RF3IS	18	WO	0x0	接收 FIFO 空间大小为 1/4 时中断清零 0: 不清零 1: 清零
RF2IS	17	WO	0x0	接收 FIFO 空间大小为 1/2 时中断清零 0: 不清零 1: 清零
RF1IS	16	WO	0x0	接收 FIFO 空间大小为 3/4 时中断清零 0: 不清零 1: 清零
TF0IE	15	RW	0x0	发送 FIFO 空中断清零 0: 不清零 1: 清零
TF1IE	14	RW	0x0	发送 FIFO 空间大小为 1/4 时中断清零 0: 不清零 1: 清零
TF2IE	13	RW	0x0	发送 FIFO 空间大小为 1/2 时中断清零 0: 不清零 1: 清零
TF3IE	12	RW	0x0	发送 FIFO 空间大小为 3/4 时中断清零 0: 不清零 1: 清零
RF4IE	11	RW	0x0	接收 FIFO 满中断清零 0: 不清零 1: 清零
RF3IE	10	RW	0x0	接收 FIFO 空间大小为 1/4 时中断清零 0: 不清零 1: 清零
RF2IE	9	RW	0x0	接收 FIFO 空间大小为 1/2 时中断清零 0: 不清零 1: 清零
RF1IE	8	RW	0x0	接收 FIFO 空间大小为 3/4 时中断清零 0: 不清零 1: 清零
TF0IS	7	RO	0x0	发送 FIFO 空中断状态 0: 不生效 1: 生效
TF1IS	6	RO	0x0	发送 FIFO 空间大小为 1/4 时中断状态 0: 不生效 1: 生效
TF2IS	5	RO	0x0	发送 FIFO 空间大小为 1/2 时中断状态 0: 不生效 1: 生效
TF3IS	4	RO	0x0	发送 FIFO 空间大小为 3/4 时中断状态 0: 不生效 1: 生效
RF4IS	3	RO	0x0	接收 FIFO 满中断状态 0: 不生效 1: 生效
RF3IS	2	RO	0x0	接收 FIFO 空间大小为 1/4 时中断状态 0: 不生效 1: 生效
RF2IS	1	RO	0x0	接收 FIFO 空间大小为 1/2 时中断状态 0: 不生效 1: 生效
RF1IS	0	RO	0x0	接收 FIFO 空间大小为 3/4 时中断状态 0: 不生效 1: 生效

5.19.3.13 CAN_FRM_INFO (0x48)

域	位	读写	复位值	描述
SSPD	31:16	R0	0x0	二次采样点延迟：发送器从发送 fdf 位下降沿，到接收到 fdf 下降沿之间 can_clk (200MHz) 的周期数。 二次采样点延迟时间 = SSPD * 5ns
reserved	15:6	R0	0x0	保留
RXFC	5:0	R0	0x0	CAN 控制器接收 FIFO 中有效帧个数 数据帧数范围 0~64

5.19.3.14 CAN_TIME_OUT (0x4C)

域	位	读写	复位值	描述
T0	31:0	W0	0x0	接收 FIFO 读取数据超时阈值。若接收 FIFO 中深度大于所设的阈值，则会产生超时中断，超时中断对应寄存器 can_intr2 bit0。 实际阈值=写入值 * 5ns

5.19.3.15 CAN_TIME_OUT_CNT (0x50)

域	位	读写	复位值	描述
T0C	31:0	R0	0x0	接收 FIFO 读取数据超时计数器

5.19.3.16 CAN_INTR2 (0x54)

域	位	读写	复位值	描述
reserved	31:25	R0	0x0	保留
TORIS	24	R0	0x0	接收 FIFO 读取超时原始中断状态 0: 不生效 1: 生效
reserved	23:17	R0	0x0	保留
T0IC	16	R0	0x0	接收 FIFO 读取超时中断清零 0: 不清零 1: 清零
reserved	15:9	R0	0x0	保留
T0IM	8	R0	0x0	接收 FIFO 读取超时中断使能 0: 不使能 1: 使能
reserved	7:1	R0	0x0	保留
T0IS	0	R0	0x0	接收 FIFO 读取超时中断状态 0: 不生效 1: 生效

5.19.3.17 CAN_TX_FIFO (0x100~0x1FF)

域	位	读写	复位值	描述
TF	31:0	W0	0x0	发送 FIFO 影子寄存器

5.19.3.18 CAN_RX_FIFO (0x200~0x2FF)

域	位	读写	复位值	描述
RF	31:0	R0	0x0	接收 FIFO 影子寄存器

5.19.3.19 CAN_RX_INFO_FIFO (0x300~0x3FF)

域	位	读写	复位值	描述
reserved	31:8	R0	0x0	保留
IORF	7	R0	0x0	读取接收 FIFO 时, 当前帧是否为扩展帧 1: 是 0: 否
FORF	6	R0	0x0	读取接收 FIFO 时, 当前帧是否为 CAN FD 帧 1: 是 0: 否
RORF	5	R0	0x0	读取接收 FIFO 时, 当前帧是否为远程帧 1: 是 0: 否
WNORF	4:0	R0	0x0	读取接收 FIFO 时, 当前接收帧的字数 (4byte)

5.20 JTAG Master 控制器

飞腾派集成的 JTAG master 控制器涉及 5 个 JTAG 总线接口信号, 遵循 JTAG 总线协议, JTAG 协议被广泛应用于各类硬件调试接口以及下载接口, 由于不同厂家不同场景的从机设备支持的指令 IR 不同, 所以不能兼容使用。JTAG master 控制器支持 IR 和 DR 级别的配置, 而不是通过对 GPIO 信号线的编程模拟 JTAG 信号线, 提高软件配置效率的同时, 灵活适用于多个厂商的设备。

5.20.1 操作说明

5.20.1.1 复位控制

控制 jtag 从机复位有两种方式:

1. 写寄存器 rst_ctl 的 ntrst_set 位, 通过对该位写 1 或写 0, 实现 JTAG 总线上的 jtgm_ntrst 信号直接控制。
2. 写寄存器 rst_ctl 的 tms_rst_set 位, 写 1 后, 模块驱动 jtgm_tms 在 5 个 jtgm_tck 时钟上升沿为高, 保证从机状态机处于 Test-Logic-Reset 状态。

寄存器名	地址	字段	字段宽度	复位值	描述
rst_ctl	0x000	mst_rst_set	2	0x0	mst 的状态机强制跳转至复位, 并清空输入输出 fifo 内容。
		tms_rst_set	1	0x0	驱动 tms 信号为 1 持续 5 个 tck 时钟周期, 实现对从机状态机的复位, 写 1 进行复位, 变为 0 表示复位完成。

寄存器名	地址	字段	字段宽度	复位值	描述
		ntrst_set	0	0x1	JTAG 总线上 ntrst 信号的软件控制位。当外接 jtag 总线为低复位有效时，先写 0 再写 1 可以触发一次 jtag 总线 ntrst 硬件复位，当 jtag 总线为高复位有效时，写 1 再写 0 可以触发一次。默认模块复位完成后，该信号由 0 变为 1。

5.20.1.2 时钟控制

JTAG Master 控制器主时钟为 50MHz。jtagm_tck 时钟是由 50M 时钟进行分频所得。分频系数可以通过 16 位的寄存器配置，范围为 2-65536，占空比不固定为 50%。

当 jtag master 处于空闲时，即 IR 和 DR 队列为空时，可以配置 jtagm_tck 时钟持续翻转或者保持为低。

对时钟的控制通过配置 tck_ctl 寄存器实现。

5.20.1.3 采样点调节

jtag-master 只在 jtagm_tck 驱动下，采样 jtagm_tdi 信号，JTAG 总线要求信号在下降沿输出，上升沿采样，减少板级等延迟影响，但是为了保证采样点的有效性，jtag-master 支持除了上升沿以外，提供一共四个采样点的配置选择。通过 fsm_ctl (0x008) 中 jtagm_tdi_sample 寄存器位域配置选择对 jtagm_tdi 进行采样。

当出现奇数分频时，分频时钟的占空比不为 50%，对应二倍频也有变化。3 分频的 tck 的 2 倍频为 1 分频，即不分频。5 分频 tck 的 2 倍频为 2 分频。

当分频系数较小时，为 2 或 3，由于其 2 倍频为不分频，为了避免在主时钟下降沿驱动时序逻辑，所以只支持两到三个采样点的调节。

在四及以上分频系数的 jtagm_tck 的一个时钟周期内，提供四个采样点。

在二分频时，因分频系数限制，且不考虑倍频电路，提供两个采样点 0，1。

在三分频时，提供三个采样点 0，1，2。

对信号的驱动和采样的控制通过配置 tdi_ctl 寄存器实现。

5.20.1.4 IR 传输

如果发送的 IR 的位宽小于等于 16 位：

1. 配置 ir_16_cofig 寄存器，[24]和[20]置 1，[19:16]写要发送 IR 的长度，[15:0]写 IR 内容。

如果发送的 IR 的位宽大于 16 位，小于等于 32 位：

1. 配置 ir_out_reg 寄存器，[31:0]写 IR 的内容。

2. 配置 `ir_config` 寄存器, [8]置 1, [7:5]写 3'b0-withexit, [4:0]写要发送 IR 的长度。

如果发送的 IR 的位宽大于 32 位:

1. 配置 `ir_out_reg` 寄存器, [31:0]写 IR 的内容。
2. 配置 `ir_config` 寄存器, [8]置 1, [7:5]写 3'b001-withoutexit, [4:0]写要发送 IR 的长度。
3. 重复 1~2 操作,直到剩余 IR 长度小于 32 位。配置完 `ir_out_reg` 寄存器后,配置 `ir_config` 寄存器, [8]置 1, [7:5]写 3'b0-withexit, [4:0]写要发送 IR 的长度。

5.20.1.5 DR 传输

如果发送的 DR 的位宽小于等于 32 位:

1. 配置 `dr_out_reg` 寄存器, [31:0]写 DR 的内容。
2. 配置 `dr_config` 寄存器, [8]置 1, [7:5]根据是否需要从机返回内容写 3'b010-DR-norsp-withexit 表示不需要返回, 或者 3'b110-DR-rsp-withexit 表示需要返回, [4:0]写要发送的 DR 长度。
3. 如果需要从机返回,可以读取 `dr_in_reg` 寄存器,支持 16 深度 fifo 缓存。

如果发送的 DR 的位宽大于 32 位:

1. 配置 `dr_out_reg` 寄存器, [31:0]写 DR 的内容。
2. 配置 `dr_config` 寄存器, [8]置 1, [7:5]根据是否需要从机返回内容写 3'b011-DR-norsp-withoutexit 表示不需要返回, 或者 3'b111-DR-rsp-withoutexit 表示需要返回, [4:0]写要发送的 DR 长度。
3. 重复 1-2 操作,直到剩余未发送的 DR 长度小于 32 位。配置完 `dr_out_reg` 寄存器后,配置 `dr_config` 寄存器, [8]置 1, [7:5]根据是否需要从机返回内容写 3'b010-DR-norsp-withexit 表示不需要返回, 或者 3'b110-DR-rsp-withexit 表示需要返回, [4:0]写要发送 IR 的长度。
4. 如果需要从机返回,可以读取 `dr_in_reg` 寄存器,支持 16 深度 fifo 缓存。

注意:默认从最低有效位开始驱动和采样 `jtgm_tdo` 和 `jtgm_tdi`,如果需要从最高有效位开始驱动和采样,需要在配置 `dr_config` 时同时设置[12]为 1。

5.20.1.6 中断处理

面向软件使用场景提供如下五个中断控制:

1. 指令输出 `out_fifo` 预空中断,预空与实际空的差值可以由 `int_cfg` 寄存器配置,默认值为 7。

2. 从机数据返回 in_fifo 预满中断，预满与实际满的差值可以由 int_cfg 寄存器配置，默认值为 7。
3. 从机数据返回 in_fifo 内容超时未读走中断，超时的时钟周期可以由 int_cfg 寄存器配置，默认值为 50 个时钟周期。
4. 从机数据返回 in_fifo 数据被覆盖报中断。
5. JTAG 总线状态机在扫描链模式下，请求未返回有效 1 的 tck 时钟周期数超过配置数量。

每个中断都受掩膜控制，默认不掩膜，采样为高电平有效。操作的寄存器为 int_cfg 和 int_ctl，可以根据需求自行配置。

5.20.2 寄存器列表

表 5-58 JTAG Master 寄存器基地址

名称	基地址
JTAG Master	0x000_2804_4000

表 5-59 JTAG Master 寄存器列表

寄存器	偏移	描述
jtgm_ctl	0x000	通过 tms 或 ntrst 控制从机复位
tck_ctl	0x004	配置 tck 的分频与空闲时是否翻转
fsm_ctl	0x008	状态机观察与超时控制，sjtg 相关控制
int_cfg	0x00C	fifo 空满上报中断相关配置
dr_gen_ctl	0x010	产生配置长度的固定 0 或 1 的 DR 内容输出
int_ctl	0x014	中断的相关掩膜与状态
dr_config	0x018	配置 DR 传输设置
dr_out_reg	0x01C	配置输出 DR 内容
dr_in_reg	0x020	读取从机返回 DR 内容
ir_config	0x024	配置输出 IR 设置
ir_out_reg	0x028	配置输出 IR 内容
ir_16_config	0x02C	配置输出 IR 设置与内容（IR 长度小于等于 16）

5.20.3 寄存器说明

5.20.3.1 jtgm_ctl (0x000)

域	位	读写	复位值	描述
tms_rst_set	1	W1-1	0x0	驱动 tms 信号为 1 持续 5 个 tck 时钟周期，实现对从机状态机的复位，写 1 进行复位，同时 mst 的状态机强制跳转至复位，并清空输入输出 fifo 内容。
ntrst_set	0	RW	0x0	JTAG 总线上 ntrst 信号的软件控制位。jtag 总线 ntrst 的电平可以通过该位进行配置控制，一般需要对该位写 0 后写 1 实现对从机 ntrst 复位控制。

5.20.3.2 tck_ctl (0x004)

域	位	读写	复位值	描述
tck_reset_disable	17	RW	0x1	在连续处于 RESET 状态超过 fsm_ctl (0x008) 寄存器中 fsm_idletout_fg 配置值个时钟周期时, 输出 tck 是否要保持翻转, 默认为 1 不翻转。
tck_idle_disable	16	RW	0x1	在连续处于 IDLE 状态超过 fsm_ctl (0x008) 寄存器中 fsm_idletout_cfg 配置值个时钟周期时, 输出 tck 是否要保持翻转, 默认为 1 不翻转。
tck_div	15:0	RW	0x5	分频系数配置, 默认配置 10M 的频率, 对应分频系数复位值为 16'h5, 至少为 2 分频, 否则设置不生效。

5.20.3.3 fsm_ctl (0x008)

域	位	读写	复位值	描述
jtagm_state	18:16	RO	0x0	jtagmaster 处于的状态观察: 3'h0: RESET 3'h1: IDLE 3'h2: SELECT_DR 3'h3: SHIFT_DR 3'h4: EXIT1_DR 3'h5: SELECT_IR 3'h6: SHIFT_IR 3'h7: EXIT1_IR
reserved	15:10	RO	0x0	保留
fsm_idletout_cfg	9:4	RW	0x32	当 fsm 超过配置值个 tck_cycles, 处于 IDLE 态时, tck 在 tck_reset_disable 和 tck_idle_disable 控制下不输出 tck 翻转, 默认 50 个。
jtg_m_tdi_turn_en	3	RW	0x0	jtg_m_tdi 输入内容进模块处理前, 寄存器可配置进行反向操作, 1 表示数据内容取反。
tdo_equal_tdi_en	2	RW	0x0	输出 jtg_m_tdo 信号可以直接是输入 jtg_m_tdi 信号值, 1 表示输出 tdo 即输入 tdi 值。
jtg_m_tdi_sample	1:0	RW	0x0	jtg_m_tdi 输入信号一个 tck 时钟中四个采样点的调节, 默认 tck 的上升沿。

5.20.3.4 int_cfg (0x00C)

域	位	读写	复位值	描述
outf_empty_cfg	19:16	RW	0x7	out_fifo 还剩配置值个数时, 报空状态。
inf_full_cfg	15:12	RW	0x7	in_fifo 还剩配置值个数时, 报满状态。
inf_timeout_cfg	11:0	RW	0x32	in_fifo 不为空时, 没有读操作的超时阈值配置, 默认为 50*1024 个参考时钟周期。

5.20.3.5 int_ctl (0x014)

域	位	读写	复位值	描述
outf_cnt	25:20	R0	0x0	out_fifo 内有效内容个数。
reserved	19:18	R0	0x0	保留
outf_empty_mask	17	RW	0x0	outf_empty 中断的掩膜，默认不掩膜。
outf_empty	16	R0	0x0	outf_empty 的真实中断状态，不空则为 0，不需要清中断操作。
reserved	15:13	R0	0x0	保留
inf_cnt	12:8	R0	0x0	in_fifo 内有效内容个数。
inf_full_mask	7	RW	0x0	inf_full 中断的掩膜，默认不掩膜。
inf_timeout_mask	6	RW	0x0	inf_timeout 中断的掩膜，默认不掩膜。
inf_full	5	R0	0x0	inf_full 的真实中断状态，不满则为 0，不需要清中断操作。
inf_timeout	4	R0	0x0	inf_timeout 的真实中断，有读 in_fifo 操作时，则为 0，不需要清中断操作。
inf_wfull_mask	3	RW	0x0	inf_wfull 中断的掩膜，默认不掩膜。
inf_wfull	2	R0	0x0	inf_wfull 真实中断状态，读则清中断信号。
sjtag_readtout_mask	1	RW	0x0	sjtag_readtout 中断掩膜，默认不掩膜。
sjtag_readtout	0	RC	0x0	sjtag_readtout 真实中断，sjtag 读操作超时未收到响应有效位，错误中断，读则清。

5.20.3.6 dr_config (0x18)

域	位	读写	复位值	描述
sb_set	12	RW	0x0	为 1 表示 msb，从最高有效位开始驱动和采样 jtgm_tdo 和 jtgm_tdi，为 0 表示 lsb，从最低有效位开始驱动和采样。
reserved	11:9	RW	0x0	保留
dr_valid	8	W1-1	0x0	写 1，表示写入 dr 内容有效。

5.20.3.7 dr_out_reg (0x010)

域	位	读写	复位值	描述
dr_out_data	31:0	RW	0x0	配置输出队列，DR 寄存器的内容，写一次，有效一次，只能在 int_ctl (0x014) 中 outf_full 为 0 时写。

5.20.3.8 dr_in_reg (0x020)

域	位	读写	复位值	描述
DR_in_data	31:0	R0	0x0	读取输入队列，DR 寄存器的内容，读一次，更新一次，只能在 int_ctl (0x014) 中 outf_empty 为 0 时读。

5.20.3.9 ir_config (0x024)

域	位	读写	复位值	描述
ir_valid	8	W1-1	0x0	写 1，表示写入 ir 内容有效。
ir_state	7:5	W1-1	0x0	000 表示 IR-withExit; 001 表示 IR-withoutExit。
ir_bitstrb	4:0	RW	0x0	设置 IR 数据的有效位，如果有效位个数一致，0 表示 1 位，最高可表示 32 位有效。

5.20.3.10 ir_out_reg (0x028)

域	位	读写	复位值	描述
ir_out_data	31:0	RW	0x0	配置输出队列 IR 寄存器的内容，写一次，有效一次，只能在 int_ctl (0x014) 中 outf_full 为 0 时写。

5.20.3.11 ir_16_config (0x02C)

域	位	读写	复位值	描述
ir_16	24	RW	0x1	为 1 表示 IR 小于 16 位，ir_out_reg 和 ir_config 内容无效，反之表示 ir_16_config 内容无效。
ir_16_state	23:21	W1-1	0x0	000 表示 IR-withExit; 001 表示 IR-withoutExit。
ir_16_valid	20	W1-1	0x0	写 1，表示写入 ir 内容有效。
ir_16_bitstrb	19:16	RW	0x0	设置 IR 数据的有效位，0 表示 1 位，最高可表示 16 位有效。
ir_16_out_data	15:0	RW	0x0	当 IR 位宽小于等于 16 位时，写该 16 位寄存器，进行 ir 内容的写入。
ir_16	24	RW	0x1	为 1 表示 IR 小于 16 位，ir_out_reg 和 ir_config 内容无效，反之表示 ir_16_config 内容无效。

5.21 MIO 控制器

MIO 是一个包含多种控制器功能的多路选择控制器。

5.21.1 操作说明

飞腾派的每个 MIO 均可单独当做 UART/I2C。端口功能的选择，可以通过配置 creg_mio_func_sel 寄存器来实现，配置为 00 选择 I2C，配置为 01 选择 UART。

MIO 控制器主时钟为 50MHz，UART/I2C 模式下主时钟都为该频率。

5.21.2 寄存器列表

表 5-60 MIO 寄存器基地址

名称	基地址
MIO0	0x000_2801_4000
MIO1	0x000_2801_6000
MIO2	0x000_2801_8000
MIO3	0x000_2801_A000
MIO4	0x000_2801_C000
MIO5	0x000_2801_E000
MIO6	0x000_2802_0000
MIO7	0x000_2802_2000
MIO8	0x000_2802_4000
MIO9	0x000_2802_6000
MIO10	0x000_2802_8000
MIO11	0x000_2802_A000
MIO12	0x000_2802_C000
MIO13	0x000_2802_E000
MIO14	0x000_2803_0000
MIO15	0x000_2803_2000

MIO 寄存器空间为 8KB，其中低 4KB 空间是 2 个控制器的共享空间，当选择不同功能时，该 4KB 空间属于相关控制器的寄存器空间；高 4KB 空间为 MIO 全局控制寄存器空间，具体定义如下表。

表 5-61 MIO 全局控制寄存器列表

寄存器	偏移	描述
creg_mio_func_sel	0x1000	控制当前 MIO 功能选择
mio_func_sel_state	0x1004	当前 MIO 功能状态
mio_version	0x1100	MIO 版本号

注意：表中寄存器的实际地址为 MIO 基地址+偏移地址，无需另外再+0x1000。

5.21.3 寄存器说明

5.21.3.1 creg_mio_func_sel (0x1000)

域	位	读写	复位值	描述
mfs	[1:0]	RW	0x0	控制当前 mio 功能选择 'h00: I2C 'h01: UART
reserved	[31:2]	RW	0x0	保留

5.21.3.2 mio_func_sel_state (0x1004)

域	位	读写	复位值	描述
mfst	[1:0]	RO	0x0	当前 mio 功能状态
reserved	[31:2]	RW	0x0	保留

5.21.3.3 mio_version (0x1100)

域	位	读写	复位值	描述
mv	[31:0]	RO	0x1	当前 mio 版本号

5.22 UART 控制器

飞腾派集成了 9 线 UART 控制器、5 线 UART 控制器和 3 线 UART 控制器，其中 3 线 UART 控制器均由 MIO 配置实现。

5.22.1 操作说明

采用 MIO 配置实现 3 线 UART 控制器时，需要首先配置 MIO 寄存器 `creg_mio_func_sel` 为 0x01，选择 UART 模式。其他操作请参考下述配置流程。需要注意，3 线 UART 控制器无流控功能。

5.22.1.1 初始化配置

1. 配置前需要先关闭 UART：向 0x30 (UARTCR) 地址的 bit[0] 写 0。
2. 配置波特率：3 线 UART 控制器 (MIO 配置) 主时钟为 50MHz，其他 UART 控制器 (非 MIO 配置) 主时钟为 100MHz。向 0x24 (UARTIBRD) 地址写入 divisor 整数，向 0x28 (UARTFBRD) 地址写入 divisor 小数 (变换后)。

公式： $\text{divisor} = \text{uartclk} / (16 \times \text{波特率})$

例如：uartclk 为 100MHZ，波特率为 115200 时：

$\text{divisor} = (100 \times 10^6) / (16 \times 115200) = 54.253$

整数位 BRDI=54，小数位 BRDf=0.253

$m = \text{integer}((0.253 \times 64) + 0.5) = 16$

因此向 0x24 (UARTIBRD) 地址写入 0x36 (54 转换成十六进制)，向 0x28 (UARTFBRD) 地址写入 0x10。

3. 配置位宽、校验、停止、使能 FIFO：向 0x2C (UARTLCR_H) 地址写入相应数值。
例如：位宽为 8bit，没有校验位，1 拍停止位，使能 FIFO，向 0x2C (UARTLCR_H) 地址写入 0x70。

4. 如果需要使能中断，向 0x38 (UARTIMSC) 地址相应位写 1，打开中断。
5. 如果使能 FIFO，并且使能中断，需要配置产生中断的 FIFO 阈值，即向 0x34 (UARTIFLS) 地址写入相应数值，传输和接受 FIFO 深度都是 32 个字节。
6. 使能 UART、loopback、发送/接收、hardware flow control：向 0x30 (UARTCR) 地址写入相应值。

例如：如果使能 UART，使能发送和接受数据，不使能 loopback、hardware flow control 相关功能，向 0x30 (UARTCR) 地址写入 0x0301。

5.22.1.2 发送数据操作流程

使用轮询方式：

判断发送 FIFO 不满：读 0x18 (UARTFR) 地址，判断 bit[5] 为 0 时，即发送 FIFO 不满。

写入数据：向 0x00 (UARTDR) 写入数据，根据配置一次可以写入 5-8bit。

使用中断方式：

判断是否产生发送中断：读 0x3C (UARTRIS) 地址，判断 bit[5] 为 1 时，即产生发送中断。如果使能 FIFO，当传输 FIFO 里的数据小于等于设置的 FIFO 阈值时，产生中断；如果没有使能 FIFO，相当于传输 FIFO 深度为 1 字节，当数据寄存器的数据发送后，产生中断。

写入数据：向 0x00 (UARTDR) 写入数据，当发送 FIFO 的数据大于设置的 FIFO 阈值时，中断清除，或者向 0x44 (UARTICR) 中断清除寄存器写 0x20，清除中断。

5.22.1.3 接收数据操作流程

使用轮询方式：

判断接收 FIFO 不空：读 0x18 (UARTFR) 地址，判断 bit[4] 为 0 时，即接收 FIFO 不空。

读出数据：读 0x00 (UARTDR) 数据寄存器的数据。

使用中断方式：

判断是否产生接收中断：读 0x3C (UARTRIS) 地址，判断 bit[4] 为 1 时，即产生接收中断。如果使能 FIFO，当接收 FIFO 里的数据大于等于设置的 FIFO 阈值时，产生中断；如果没有使能 FIFO，相当于接收 FIFO 深度为 1 字节，当接收到 1 字节数据后，产生中断。

读出数据：读 0x00 (UARTDR) 数据寄存器，当接收 FIFO 里的数据小于设置的 FIFO 阈值时，中断清除，或者向 0x44 (UARTICR) 中断清除寄存器写 0x10，清除中断。

5.22.1.4 Flow control 相关操作

RTS flow control: 向 0x30 (UARTCR) 地址的 bit[14] 写 1, 使能 RTS flow control。

CTS flow control: 向 0x30 (UARTCR) 地址的 bit[15] 写 1, 使能 CTS flow control。

5.22.2 寄存器列表

MIO 配置实现的 3 线 UART 控制器寄存器基地址请参考 MIO 寄存器基地址。

表 5-62 UART 寄存器基地址

名称	基地址
UART0	0x000_2800_C000
UART1	0x000_2800_D000
UART2	0x000_2800_E000
UART3	0x000_2800_F000

表 5-63 UART 寄存器列表

寄存器名称	偏移	描述
UARTDR	0x000	数据寄存器
UARTRSR/UARTECR	0x004	接收状态寄存器/错误清除寄存器
UARTFR	0x018	标志寄存器
UARTILPR	0x020	低功耗计数寄存器
UARTIBRD	0x024	波特率整数值配置寄存器
UARTFBRD	0x028	波特率小数配置寄存器
UARTLCR_H	0x02C	线控寄存器
UARTCR	0x030	控制寄存器
UARTIFLS	0x034	FIFO 阈值选择寄存器
UARTIMSC	0x038	中断屏蔽选择/清除寄存器
UARTRIS	0x03C	中断状态寄存器
UARTMIS	0x040	中断屏蔽状态寄存器
UARTICR	0x044	中断清除寄存器
UARTDMACR	0x048	DMA 控制寄存器

5.22.3 寄存器说明

5.22.3.1 UARTDR (0x000)

域	位	读写	复位值	描述
reserved	31:12	RW	0x0	保留
OE	11	RW	0x0	溢出错误。 如果接收到数据并且接收 FIFO 已满, 该位设置为 1。 一旦 FIFO 中有一个空位, 并且可以向其写入一个新字符, 则此项清除为 0。
BE	10	RW	0x0	突发错误。

域	位	读写	复位值	描述
				如果检测到突发条件，则该位设置为 1，表示接收到的数据输入保持在较低的状态超过一个完整的字节传输时间（定义为起始位、数据位、奇偶校验位和停止位）。 在 FIFO 模式下，此错误与 FIFO 顶部的字符相关联。当突发产生时，只有一个 0 字符加载到 FIFO 中。只有在接收数据输入为 1（标记状态）并且接收到下一个有效起始位后，才启用下一个字符。
PE	9	RW	0x0	奇偶校验错误。 当设置为 1 时，表示接收的数据字符的奇偶性与 UARTLCR_H 寄存器中 EPS 和 SPS 位不匹配。 在 FIFO 模式下，此错误与 FIFO 顶部的字符相关。
FE	8	RW	0x0	帧错误。 当设置为 1 时，表示接收字符没有有效的停止位（有效的停止位为 1）。 在 FIFO 模式下，此错误与 FIFO 顶部的字符相关。
DATA	7:0	RW	0x0	接收（读）数据，传输（写）数据。

5.22.3.2 UARTCCR (0x004)

域	位	读写	复位值	描述
reserved	31:8	RW	0x0	保留
uartccr	7:0	RW	0x0	写入该寄存器将清除帧错误、奇偶校验错误、突发错误和溢出错误。 可写入任意值。

5.22.3.3 UARTSR (0x004)

域	位	读写	复位值	描述
reserved	31:4	RW	0x0	保留
OE	3	RW	0x0	溢出错误。 如果接收到数据并且此时 FIFO 已满，此位设置为 1。该位通过写入 UARTCCR 清除为 0。FIFO 的内容保持有效，因为当 FIFO 满时不再写入数据，只覆盖移位寄存器的内容。 CPU 现在必须读数据，以清空 FIFO。
BE	2	RW	0x0	突发错误。 如果检测到突发条件，则该位设置为 1，表示接收到的数据输入保持在较低的状态超过一个完整的字节传输时间（定义为起始位、数据位、奇偶校验位和停止位）。在写入 UARTCCR 后该位清除为 0。在 FIFO 模式下，该错误与 FIFO 顶部的字符相关联。只有在接收数据输入变为 1（标记状态）并且接收到下一个有效起始位后，才启用下一个字符。
PE	1	RW	0x0	奇偶校验错误。 当设置为 1，表示接收到数据字符的奇偶性与 UARTLCR_H

域	位	读写	复位值	描述
				寄存器的 EPS 和 SPS 不匹配, 通过写入 UARTECR 将该位清除为 0。在 FIFO 模式下, 该错误与 FIFO 顶部字符相关联。
FE	0	RW	0x0	帧错误。 当设置为 1, 表示接收字符没有有效的停止位 (有效停止位为 1)。通过写入 UARTECR 将该位清除为 0 在 FIFO 模式, 该错误与 FIFO 顶部的字符相关联。

5.22.3.4 UARTFR (0x018)

域	位	读写	复位值	描述
reserved	31:9	RO	0x0	保留, 读取为零。
RI	8	RO	0x0	Ring 指示信号。该位表示 UART Ring 指示信号、nUARTRI、调制解调器状态输入。当 nUARTRI 低电位时该位为 1。
TXFE	7	RO	0x1	发送 FIFO 空。该位由寄存器 UARTLCR 中 FEN 位的状态决定。如果 FIFO 被禁用, 当发送保持寄存器为空时该位为 1。如果 FIFO 可使用, 当发送 FIFO 位为空时设置 TXFE 位。该位不表示发送移位寄存器是否有数据。
RXFF	6	RO	0x0	接收 FIFO 满。该位取决于 UARTLCR_H 寄存器中 FEN 位的状态。如果 FIFO 被禁用。当接收保持寄存器满时设置该位。如果 FIFO 可使用, 当接收 FIFO 满时设置 RXFF 位。
TXFF	5	RO	0x0	发送 FIFO 已满。该位取决于 UARTLCR_H 寄存器中 FEN 位的状态。如果 FIFO 被禁用, 当发送保持寄存器已满时设置该位。如果 FIFO 能使用, 当传输 FIFO 已满时设置 TXFF 位。
RXFE	4	RO	0x1	接收 FIFO 为空。该位取决于 UARTLCR_H 寄存器中 FEN 位的状态。如果 FIFO 被禁用, 当接收保持寄存器为空时设置该位。如果 FIFO 能使用, 当接收 FIFO 为空时设置 RXFE 位。
BUSY	3	RO	0x0	UART 繁忙。如果该位设置为 1, UART 正忙于传输数据。该位保持设置, 直到从移位寄存器发送完整字节 (包括所以停止位) 为止。当发送 FIFO 变成不空时该位被立刻置已, 无论 UART 是否被使能。
DCD	2	RO	0x0	数据载波检测。该位表示 UART 数据载波、nUARTDCD、调制解调状态输入。当 nUARTDCD 为低电平时该位为 1。
DSR	1	RO	0x1	数据准备完成。该位表示 UART 数据准备完成、nUARTDSR、调制解调状态输入。也就是说, 当 uUARTDSR 为低电平时该位为 1。
CTS	0	RO	0x1	清除发送。该位表示 UART 清除发送、nUARTCTS、调制解调状态输入的补码。当 nUARTCTS 为低电平时该位为 1。

5.22.3.5 UARTILPR (0x020)

域	位	读写	复位值	描述
reserved	31:9	RO	0x0	保留

域	位	读写	复位值	描述
ILPDVSR	7:0	RW	0x0	8 位低功耗计数器。在复位时这些位清零。

5.22.3.6 UARTIBRD (0x024)

域	位	读写	复位值	描述
reserved	31:16	RO	0x0	保留
BAUD DIVINT	15:0	RW	0x0	波特率计算因子的整数值。在复位时这些位被清除为 0。

5.22.3.7 UARTFBRD (0x028)

域	位	读写	复位值	描述
reserved	31:6	RO	0x0	保留
BAUD DIVFRAC	5:0	RW	0x0	波特率计算因子的小数值。在复位时这些位清除为 0。

5.22.3.8 UARTLCR_H (0x02C)

域	位	读写	复位值	描述
reserved	31:8	RO	0x0	保留，读取为零。
SPS	7	RW	0x0	奇偶校验位。 0: 奇偶校验被禁用 1: 奇偶校验两者之一 如果 EPS 位为 0 那么奇偶校验位被传输并且在数据为 1 时检查。如果 EPS 位为 1 那么奇偶校验位被传输并且在数据为 0 时检查。当 PEN 位禁用奇偶校验检测和生成时，该位没有效果。
WLEN	6:5	RW	0x0	数据长度。表示在一次发送或接收的数据位的数量，如下所示： b11: 8 位 b10: 7 位 b01: 6 位 b00: 5 位
FEN	4	RW	0x0	FIFO 使能位。 0: FIFOs 是禁用的（字符模式），FIFOs 是一个字节深度的保持寄存器。 1: 传输与接收 FIFO 缓冲区使能（FIFO 模式）。
STP2	3	RW	0x0	两个停止位选择。如果该位设置为 1，两个停止位正在帧末尾传输。接收逻辑不检查接收到的两个停止位。
EPS	2	RW	0x0	奇偶类型选择，在传输和接收期间控制 UART 使用的奇偶校验类型。 0: 奇数校验 1: 偶数校验

域	位	读写	复位值	描述
				当 PEN 位禁用奇偶校验检测和生成时该位无效。
PEN	1	RW	0x0	奇偶校验使能。 0: 奇偶校验被禁止 1: 奇偶校验使能
BRK	0	RW	0x0	发送突发命令, 如果该位设置为 1, 则在完成当前字符传输后, UARTTXD 会持续输出一个低电平。为了正确执行突发命令, 软件必须将该位设置为至少两个完整的帧传输。对于正常使用, 该位必须清除为 0。

5.22.3.9 UARTCR (0x030)

域	位	读写	复位值	描述
reserved	31:16	RO	0x0	保留
CTSEn	15	RW	0x0	CTS 硬件流控使能端。如果该位设置为 1, CTS 硬件流控使能为可用。数据仅在 nUARTCTS 信号有效时传输。
RTSEn	14	RW	0x0	RTS 硬件流控使能端。如果该位设置为 1, RTS 硬件流控可用。数据仅在接收 FIFO 不满时接收它的请求。
Out2	13	RW	0x0	该位是 nUATROut2 调制解调状态输出。当该位被编程为 1 时, 输出端为 0。对于 DTE 这可以作为 RI。
Out1	12	RW	0x0	该位是 nUARTOut1 调制解调状态输出, 当该位被编程为 1 时, 输出为 0。对于 DTE 这可以作为 DCD。
RTS	11	RW	0x0	该位是 UART 发送请求、nUARTRTS、调制解调状态输出。当该位被编程为 1 时, nUARTRTS 为低。
DTR	10	RW	0x0	该位是 UART 数据传输准备完成、nUARTDTR、调制解调状态输出。当该位被编程为 1 时, nUARTDTR 为低。
RXE	9	RW	0x1	接收使能端。如果该位设置为 1, UART 的接收部分能用。具体是 UART 信号还是 SIR 信号的数据接收发生取决于 SIREN 位的设置。当在接收数据中 UART 禁用, UART 会先完成当前的传输。
TXE	8	RW	0x1	发送使能位。如果该位设置为 1, UART 的发送部分能使用。具体是 UART 信号或 SIR 信号数据传输的发生取决于 SIREN 位的设置。当在发送数据中 UART 禁用, UART 会先完成当前的传输。
reserved	7:3	RO	0x0	保留, 读取为零。
SIRLP	2	RW	0x0	SIR low-power IrDA 模式。该位用于选择 IrDA 编码模式。如果该位被清除为 0, 则低电平位作为有效的高脉冲传输, 脉冲宽度为位周期的 3/16。如果该位被设置为 1, 则脉冲宽度为 IrLPBAUD16 输入信号周期的 3 倍。设置该位使用较少的功耗, 但可能会缩短传输距离。
SIREN	1	RW	0x0	SIR 使能位。 0: IrDA SIR ENDEC 被禁用。nSITOUT 保持为低 (无光脉冲产生), 在 SIRIN 上的信号传输无效。 1: IrDA SIR ENDEC 能使用。数据在 nSIROUT 和 SIRIN 上发送与接收。UARTTXD 保持为高。信号在 UARTRXD 传输或

域	位	读写	复位值	描述
				调制解调状态输入无效。 如果 UARTEN 位禁用 UART 那么该位无效。
UARTEN	0	RW	0x0	UART 使能位。 0: UART 被禁用。如果 UART 在传输或接收中被禁用, 它在停止前完成当前字符。 1: UART 使能。UART 信号或 SIR 信号的数据发送或接收取决于 SIREN 位的设置。

5.22.3.10 UARTIFLS (0x034)

域	位	读写	复位值	描述
reserved	31:6	RO	0x0	保留
RXIFLSEL	5:3	RW	0x2	接收 FIFO 中断的阈值选择。中断的触发点如下: b000: 接收 FIFO $\geq 1/8$ full b001: 接收 FIFO $\geq 1/4$ full b010: 接收 FIFO $\geq 1/2$ full b011: 接收 FIFO $\geq 3/4$ full b100: 接收 FIFO $\geq 7/8$ full b101-b111: 保留
TXIFLSEL	2:0	RW	0x2	发送 FIFO 中断的阈值选择。中断的触发点如下: b000: 发送 FIFO $\leq 1/8$ full b001: 发送 FIFO $\leq 1/4$ full b010: 发送 FIFO $\leq 1/2$ full b011: 发送 FIFO $\leq 3/4$ full b100: 发送 FIFO $\leq 7/8$ full b101-b111: 保留

5.22.3.11 UARTIMSC (0x038)

域	位	读写	复位值	描述
reserved	31:11	RO	0x0	保留, 读取为零。
OEIM	10	RW	0x0	溢出错误中断屏蔽。读取返回 UARTOEINTR 中断的当前屏蔽。 1: 设置 UARTOEINTR 中断屏蔽; 0: 清除屏蔽。
BEIM	9	RW	0x0	突发错误中断屏蔽。读操作返回当前的 UARTBEINTR 中断屏蔽。 1: 设置 UARTBEINTR 中断屏蔽; 0: 清除屏蔽。
PEIM	8	RW	0x0	奇偶校验错误中断屏蔽。读操作返回当前的 UARTPEINTR 中断屏蔽。 1: 设置 UARTPEINTR 中断屏蔽; 0: 清除屏蔽。
FEIM	7	RW	0x0	帧错误中断屏蔽。读操作返回当前的 UARTFEINTR 中断屏

域	位	读写	复位值	描述
				蔽。 1: 设置 UARTFEINTR 中断屏蔽; 0: 清除屏蔽。
RTIM	6	RW	0x0	接收超时中断屏蔽。读操作返回当前的 UARTRTINTR 中断屏蔽。 1: 设置 UARTRTINTR 中断屏蔽; 0: 清除屏蔽。
TXIM	5	RW	0x0	发送中断屏蔽。读操作返回当前的 UARTTXINTR 中断屏蔽。 1: 设置 UARTTXINTR 中断屏蔽; 0: 清除屏蔽。
RXIM	4	RW	0x0	接收中断屏蔽。读操作返回当前的 UARTRXINTR 中断屏蔽。 1: 设置 UARTRXINTR 中断屏蔽; 0: 清除屏蔽。
DSRMIM	3	RW	0x0	nUARTDSR 调制解调中断屏蔽。读操作返回当前 UARTDSRINTR 中断屏蔽。 1: 设置 UARTDSTINTR 中断屏蔽; 0: 清除屏蔽。
DCDMIM	2	RW	0x0	nUARTDCD 调制解调中断屏蔽。读操作返回当前 UARTDCDINTR 中断屏蔽。 1: 设置 UARTDCDINTR 中断屏蔽; 0: 清除屏蔽。
CTSMIM	1	RW	0x0	nUARTCTS 调制解调中断屏蔽。读操作返回当前的 UARTCTSINTR 中断屏蔽。 写 1 时, 设置 UARTCTSINTR 中断屏蔽; 写 0 时, 清除屏蔽。
RIMIM	0	RW	0x0	nUARTRI 调制解调中断屏蔽。读操作返回当前的 UATTRIINTR 中断屏蔽。 写 1 时, 设置 UARTRIINTR 中断屏蔽; 写 0 时, 清除屏蔽。

5.22.3.12 UARTRIS (0x03C)

域	位	读写	复位值	描述
reserved	31:11	R0	0x0	保留, 读取为零。
OERIS	10	R0	0x0	溢出错误中断状态。反馈 UARTOEINTR 中断的原始中断状态。
BERIS	9	R0	0x0	突发错误中断状态。反馈 UARTBEINTR 中断的原始中断状态。
PERIS	8	R0	0x0	奇偶校验错误中断状态。反馈 UARTPEINTR 中断的原始中断状态。
FERIS	7	R0	0x0	帧错误中断状态。反馈 UARTFEINTR 中断的原始中断状态。
RTRIS	6	R0	0x0	接收超时中断状态。反馈 UATRTRINTR 中断的原始中断状态。
TXRIS	5	R0	0x0	发送中断状态。反馈 UARTTXINTR 中断的原始中断状态。
RXRIS	4	R0	0x0	接收中断状态。反馈 UARTRXINTR 中断的原始中断状态。
DSRRMIS	3	R0	0x0	nUARTDSR 调制解调中断状态。反馈 UARTDSRINTR 中断的原始中断状态。

域	位	读写	复位值	描述
DCDRMIS	2	R0	0x1	nUARTDCD 调制解调中断状态。反馈 UARTDCDINTR 中断的原始中断状态。
CTSRMIS	1	R0	0x0	nUARTCTS 调制解调中断状态。反馈 UARTCTSINTR 中断的原始中断状态。
RIRMIS	0	R0	0x1	nUARTRI 调制解调中断状态。反馈 UARTRIINTR 中断的原始中断状态。

5.22.3.13 UARTMIS (0x040)

域	位	读写	复位值	描述
reserved	31:11	R0	0x0	保留，读取为零。
OEMIS	10	R0	0x0	溢出错误屏蔽中断状态。反馈 UARTORINTR 的中断屏蔽状态。
BEMIS	9	R0	0x0	突发错误屏蔽中断状态。反馈 UARTBEINTR 的中断屏蔽状态。
PEMIS	8	R0	0x0	奇偶校验错误屏蔽中断状态。反馈 UARTREINTR 的中断屏蔽状态。
FEMIS	7	R0	0x0	帧错误屏蔽中断状态。反馈 UARTRRINTR 里的中断屏蔽状态。
RTMIS	6	R0	0x0	接收超时屏蔽中断状态。反馈 UARTRTINTR 的中断屏蔽状态。
TXMIS	5	R0	0x0	发送屏蔽中断状态。反馈 UARCTXINTR 里的中断屏蔽状态。
RXMIS	4	R0	0x0	接收屏蔽中断状态。反馈 UARTRXINTR 里的中断屏蔽状态。
DSRMMIS	3	R0	0x0	nUARTDSR 调制解调屏蔽中断状态。反馈 UARTDSRINTR 里的中断屏蔽状态。
DCDMMIS	2	R0	0x0	nUARTDCD 调制解调屏蔽中断状态。反馈 UARTDCDINTR 里的中断屏蔽状态。
CTSMMS	1	R0	0x0	nUARTCTS 调制解调屏蔽中断状态。反馈 UARTCTSINTR 里的中断屏蔽状态。
RIMMS	0	R0	0x0	nUARTRI 调制解调屏蔽中断状态。反馈 UARTRIINTR 里的中断屏蔽状态。

5.22.3.14 UARTICR (0x044)

域	位	读写	复位值	描述
reserved	31:11	R0	0x0	保留，读取为零。
OEIC	10	W0	0x0	溢出错误中断清除。 清除 UARTOEINTR 中断。
BEIC	9	W0	0x0	突发错误中断清除。 清除 UARTBEINTR 中断。
PEIC	8	W0	0x0	奇偶校验错误中断清除。 清除 UARTPEINTR 中断。
FEIC	7	W0	0x0	帧错误中断清除。 清除 UARTFEINTR 中断。
RTIC	6	W0	0x0	接收超时中断清除。 清除 UARTRTINTR 中断。
TXIC	5	W0	0x0	传输中断清除。 清除 UARCTXINTR 中断。

域	位	读写	复位值	描述
RXIC	4	WO	0x0	接收中断清除。 清除 UARTRXINTR 中断。
DSRMIC	3	WO	0x0	nUARTDSR 调制解调中断清除。 清除 UARTDSRINTR 中断。
DCDMIC	2	WO	0x0	nUARTDCD 调制解调中断清除。 清除 UARTDCDINTR 中断。
CTSMIC	1	WO	0x0	nUARTCTS 调制解调中断清除。 清除 UARTCTSINTR 中断。
RIMIC	0	WO	0x0	nUARTRI 调制解调中断清除。 清除 UARTRIINTR 中断。

5.22.3.15 UARTDMACR (0x048)

域	位	读写	复位值	描述
reserved	31:11	RO	0x0	保留，读取为零。
DMAONERR	2	RW	0x0	DMA 错误。 如果该位设置为 1，DMA 接收请求输出，当 UART 产生错误中断时，UARTRXDMASREQ 或 UARTRXDABREQ 被禁用。
TXDMAE	1	RW	0x0	发送 DMA 使能端。 如果该位设置为 1，传输 FIFO 的 DMA 启用。
RXDMAE	0	RW	0x0	接收 DMA 使能位。 如果该位设置为 1，接收 FIFO 的 DMA 启用。

5.23 I2C/SMBUS 控制器

I2C 总线是一种简单、双向二进制同步串行总线。它只需要两根线即可在连接于总线上的器件之间传输信息。SMBUS 是一种基于 I2C 扩展出来的两线接口，在 I2C 基础上定义了一些复杂的操作，为系统和电源管理相关的任务提供控制总线，原理还是基于 I2C。

飞腾派的 I2C 控制器均由 MIO 配置实现，当 MIO 作为 I2C 功能使用时，只支持 I2C 功能，不支持 SMBUS 功能。

5.23.1 操作说明

5.23.1.1 模式配置

配置 MIO 寄存器 `creg_mio_func_sel` 为 0x00，选择 I2C 模式。

5.23.1.2 配置为 master (I2C/SMBUS)

1. 配置寄存器 0x6c (IC_ENABLE) 为 0；
2. 写寄存器 0x00 (IC_CON)，配置主从、speed、设备地址宽度。例如，配置 I2C

- 为主机、7 位设备地址、standard speed，该寄存器写 0x63；
3. 将设备地址写入寄存器 0x04 (IC_TAR)；
 4. 使能控制器，配置寄存器 0x6c (IC_ENABLE) 为 1。

5.23.1.3 配置为 slave (I2C/SMBUS)

1. 配置寄存器 0x6c (IC_ENABLE) 为 0；
2. 写寄存器 0x00 (IC_CON)，例如，配置为 standard speed 的从机，该寄存器写 0x02；
3. 将设备地址写入寄存器 0x08 (IC_SAR)；
4. 使能控制器，配置寄存器 0x6c (IC_ENABLE) 为 1。

5.23.1.4 master 模式发送和接收数据流程 (I2C/SMBUS)

5.23.1.4.1 发送数据

1. 判断发送 FIFO 不满：读 0x70 (IC_STATUS) 地址，判断 bit[1] 为 1 时，即发送 FIFO 不满。
2. 发送写数据命令：向 0x10 (IC_DATA_CMD) 的 bit[7:0] 写入数据，向 bit[8] 写入 0。
3. 支持写入多字节数据，重复 1、2 步骤即可。
4. 写入最后一个字节数据时要加上停止信号，即除了向 0x10 (IC_DATA_CMD) 的 bit[7:0] 写数据，bit[8] 写 0 表示写以外，向 bit[9] 写 1 表示停止。

5.23.1.4.2 接收数据

1. 发送读数据命令：向 0x10 (IC_DATA_CMD) bit[8] 写 1，表示命令为读操作。
2. 判断接收 FIFO 不空：读 0x70 (IC_STATUS) 地址，判断 bit[3] 为 1 时，即接收 FIFO 不空。
3. 读取数据：读 0x10 (IC_DATA_CMD) 地址。
4. 支持读多字节数据，重复前三步即可。
5. 读最后一个字节数据时要加上停止信号，即除了向 0x10 (IC_DATA_CMD) 的 bit[8] 仍写 1 表示读以外，向 bit[9] 写 1 表示停止。

5.23.1.5 slave 模式发送和接收数据流程（I2C/SMBUS）

5.23.1.5.1 发送数据

1. 当接收到的地址匹配上后，读 0x34 (IC_RAW_INTR_STAT) 地址，判断 bit[5] 为 1 时，表示从机将 scl 拉低，准备好发送数据。
2. 发送写数据命令：向 0x10 (IC_DATA_CMD) 的 bit[7:0] 写入数据，向 bit[8] 写入 0。
3. 读 0x50 (IC_CLR_RD_REQ) 地址，清除中断。

5.23.1.5.2 接收数据

1. 判断接收 FIFO 不空：读 0x70 (IC_STATUS) 地址，判断 bit[3] 为 1 时，即接收 FIFO 不空。
2. 读取数据：读 0x10 (IC_DATA_CMD) 地址。

5.23.1.6 接口频率调整（I2C/SMBUS）

I2C/SMBUS 控制器主时钟为 50MHz。

$$\text{正常模式: } \text{SCL_FREQ_SS} = \frac{\text{IC_CLK_FREQ}}{(\text{IC_SS_SCL_LCNT}+1) + (\text{IC_SS_SCL_HCNT}+7 + \text{IC_FS_SPKLEN})}$$

$$\text{快速模式: } \text{SCL_FREQ_FS} = \frac{\text{IC_CLK_FREQ}}{(\text{IC_FS_SCL_LCNT}+1) + (\text{IC_FS_SCL_HCNT}+7 + \text{IC_FS_SPKLEN})}$$

$$\text{高速模式: } \text{SCL_FREQ_HS} = \frac{\text{IC_CLK_FREQ}}{(\text{IC_HS_SCL_LCNT}+1) + (\text{IC_HS_SCL_HCNT}+7 + \text{IC_HS_SPKLEN})}$$

5.23.1.7 设备超时处理（I2C/SMBUS）

T_{timeout} 参数允许主机或者从机断定设备故障，正无限期的拉低时钟，或者主设备想要让设备不驱动总线。SMBUS 协议建议从设备在单个时钟周期内在小于 T_{timeout} 最小值的时间内释放总线，设备检测到此情况要重置接口，能够在 T_{timeout} 最大值的时间内可以接受新的命令请求。

ic_smbclk_low_timeout 参数寄存器用于配置超时时间，ic_enable 寄存器使能相应控制位 smbus_mst_scktimeout_en/smbus_slv_scktimeout_en，当达到超时值时（可通过查询状态或者检测中断的方式），主机通过使能 smbus_mst_release，发送停止信号，结束传输。

5.23.1.8 主设备时钟扩展（I2C/SMBUS）

$T_{low}: MEXT$ ，定义允许主设备在消息中的一个字节内延长其时钟周期的累计时间：

START to ACK

ACK to ACK

ACK to STOP

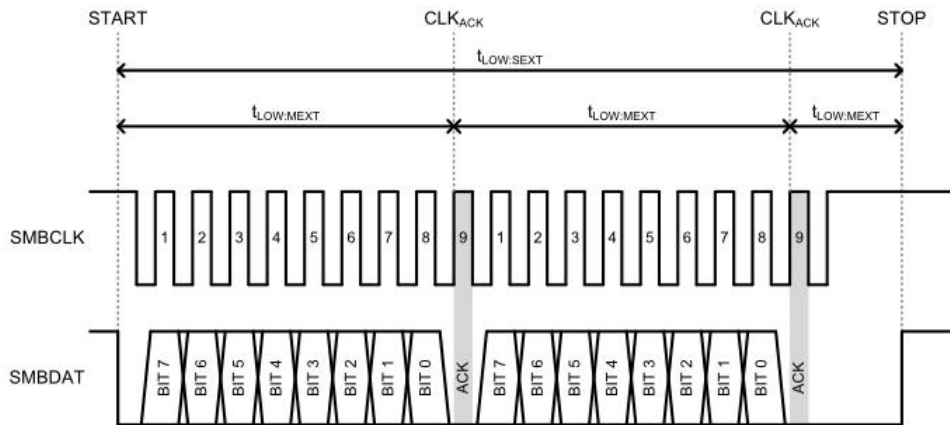


图 5-14 时钟扩展

`ic_smbclk_low_mext` 寄存器用于配置时钟扩展超时时间，`ic_enable` 寄存器使能控制位 `smbus_mst_sckextend_en` 后，检测到主设备时钟扩展超时产生中断，主机可以通过使能 `smbus_mst_release`，结束任何时钟超时传输，发送停止信号。

5.23.1.9 SMBDAT 超时处理（I2C/SMBUS）

从设备需要符合 $T_{low}: SEXT$ 。 $T_{low}: SEXT$ 是一个给定从设备允许在一条消息中从初始开始到停止延长时钟周期的累计时间。

将 `SMBDAT TIMEOUT` 值写入 `ic_smbdat_stuck_timeout` 寄存器，`ic_enable` 寄存器使能 `smbus_mst_sdatimout_en` 位后，如果 `SMBDAT` 线为低超时后，则生成 `smbus_mst_sda_low_timeout` 中断，软件可以启用 `IC_ENABLE` 寄存器的 `smbus_mst_clkreset_en` 位保持 `SCL` 为低直到 `ic_smbclk_low_timeout`，然后复位总线上所有设备的 `SMBUS` 接口。

5.23.1.10 报警信号（SMBUS）

`alert` 信号（`SMBALERT#`）是 `SMBUS` 标准指定的其他可选信号。它可以被简单的设备用来请求主机的注意。设备可以使用 `SMBALERT` 信号请求具有主功能的主机注意。这个有效的低信号被输入到主机设备并从所有其他设备输出。由于多个设备可能实现 `SMBALERT`，因此需要有线和信号。当检测到 `SMBALERT#` 信号时，主机必须发送一个警报

响应地址，该地址通过向设备发出警报而得到确认，并将该地址发送给主机并解除警报信号。如果主机仍然检测到断言的警报信号，它将重复发送警报响应地址。

5.23.2 I2C 寄存器列表

I2C 寄存器基地址请参考 MIO 寄存器基地址。

表 5-64 I2C 寄存器列表

寄存器	偏移	描述
IC_CON	0x00	I2C 控制寄存器
IC_TAR	0x04	I2C 主机地址寄存器
IC_SAR	0x08	I2C 从机地址寄存器
IC_HS_MADDR	0x0C	I2C 高速主机模式编码地址寄存器
IC_DATA_CMD	0x10	I2C 数据寄存器
IC_SS_SCL_HCNT	0x14	标准模式 I2C 时钟信号 SCL 的高电平计数寄存器
IC_SS_SCL_LCNT	0x18	标准模式 I2C 时钟信号 SCL 的低电平计数寄存器
IC_FS_SCL_HCNT	0x1C	快速模式 I2C 时钟信号 SCL 的高电平计数寄存器
IC_FS_SCL_LCNT	0x20	快速模式 I2C 时钟信号 SCL 的低电平计数寄存器
IC_HS_SCL_HCNT	0x24	高速模式 I2C 时钟信号 SCL 的高电平计数寄存器
IC_HS_SCL_LCNT	0x28	高速模式 I2C 时钟信号 SCL 的低电平计数寄存器
IC_INTR_STAT	0x2C	I2C 中断状态（中断屏蔽配置作用后的中断状态）寄存器
IC_INTR_MASK	0x30	I2C 中断屏蔽寄存器
IC_RAW_INTR_STAT	0x34	I2C 原始中断状态（真实的中断状态）寄存器
IC_RX_TL	0x38	I2C 接收 FIFO 阈值寄存器
IC_TX_TL	0x3C	I2C 发送 FIFO 阈值寄存器
IC_CLR_INTR	0x40	I2C 清除组合和单独中断寄存器
IC_CLR_RX_UNDER	0x44	清除 RX_UNDER 中断寄存器
IC_CLR_RX_OVER	0x48	清除 RX_OVER 中断寄存器
IC_CLR_TX_OVER	0x4C	清除 TX_OVER 中断寄存器
IC_CLR_RD_REQ	0x50	清除 RD_REQ 中断寄存器
IC_CLR_TX_ABRT	0x54	清除 TX_ABRT 中断寄存器
IC_CLR_RX_DONE	0x58	清除 RX_DONE 中断寄存器
IC_CLR_ACTIVITY	0x5C	清除 ACTIVITY 中断寄存器
IC_CLR_STOP_DET	0x60	清除 STOP_DET 中断寄存器
IC_CLR_START_DET	0x64	清除 START_DET 中断寄存器
IC_CLR_GEN_CALL	0x68	清除 GEN_CALL 中断寄存器
IC_ENABLE	0x6C	I2C 使能寄存器
IC_STATUS	0x70	I2C 状态寄存器
IC_TXFLR	0x74	发送 FIFO 等级寄存器
IC_RXFLR	0x78	接收 FIFO 等级寄存器
IC_SDA_HOLD	0x7C	SDA 保持时间寄存器
IC_TX_ABRT_SOURCE	0x80	I2C 发送异常状态寄存器
IC_SLV_DATA_NACK_ONLY	0x84	产生 SLV_DATA_NACK 寄存器
IC_DMA_CR	0x88	DMA 控制寄存器

寄存器	偏移	描述
IC_DMA_TDLR	0x8C	DMA 发送数据阈值
IC_DMA_RDLR	0x90	DMA 接收数据阈值
IC_SDA_SETUP	0x94	I2CSDA 建立时间寄存器
IC_ACK_GENERAL_CALL	0x98	I2CACK_Gen_Call 寄存器
IC_ENABLE_STATUS	0x9C	I2C 使能状态寄存器
IC_FS_SPKLEN	0xA0	FS 模式尖峰滤波寄存器
IC_HS_SPKLEN	0xA4	HS 模式尖峰滤波寄存器

5.23.3 I2C 寄存器说明

5.23.3.1 IC_CON (0x00)

域	位	读写	复位值	描述
reserved	31:7	RO	0x0	保留
IC_SLAVE_DISABLE	6	RW	0x1	<p>I2C Slave 功能是否关闭的控制位。即在使用 I2C 功能时通过配置此参数控制 I2C Slave 功能是打开还是关闭。软件驱动可以在系统复位后配置此参数，即通过软件配置 Slave 的使能或关闭并不是必需的。在默认状态下和复位状态下 I2C 的 Slave 功能均是使能的。如果此位设置为 1，则 I2C 控制器只能作为 Master 使用，不能响应反向 Slave 的请求。</p> <p>0：使能 I2C Slave 功能 1：关闭 I2C Slave 功能</p>
IC_RESTART_EN	5	RW	0x1	<p>配置作为 I2C Master 使用时是否支持 restart 功能。某些 I2C Slave 设备不能处理 restart 信号，但多数 I2C Slave 设备均能处理 restart 信号。</p> <p>0：不支持 restart 1：支持 restart</p> <p>当设备不支持 RESTART 功能时，I2C 的 Master 控制器支持以下功能：</p> <ul style="list-style-type: none"> 不发送起始字节 不支持 Hs 工作模式 不能进行 10 位地址读操作 <p>在不支持 restart 功能时进行以上操作，IC_RAW_INTR_STAT 寄存器中的 TX_BART 标志会被置起。</p>
IC_10BITADDR_MASTER	4	RO	0x1	<p>当 I2C_DYNAMIC_TAR_UPDATE 参数为 0（“No”）时，此位为 IC_10BITADDR_MASTER，控制其作为 I2C Master 时使用 7 位地址模式还是 10 位地址模式进行通信。</p> <p>当 I2C_DYNAMIC_TAR_UPDATE 参数为 1（“Yes”）时，此位为 IC_10BITADDR_MASTER_rd_only，读写类型为只读状态，从此处读取的值为 IC_TAR 的第 12 位所设置的值，其含义为：</p> <p>0：7 位地址模式 1：10 位地址模式</p>

域	位	读写	复位值	描述
IC_10BITADDR_SLAVE	3	RW	0x1	<p>当工作在 slave 模式时，此位用来选择 I2C 控制器响应 7 位地址访问模式还是响应 10 位地址访问请求模式。</p> <p>0: 7 位地址模式</p> <p>此模式下，对于 10 位地址访问请求，I2C 控制器忽略请求，不响应；对于 7 位地址访问请求，I2C 控制器将请求中的 7 位地址与 IC_SAR 寄存器中的 7 位地址值进行对比，若两者一致则响应，若不一致则不响应。</p> <p>1: 10 位地址模式</p> <p>此模式下，I2C 控制器只响应与 IC_SAR 寄存器中的 10 位地址相匹配的 10 位地址访问请求。</p>
SPEED	2:1	RW	0x3	<p>这个参数用来设定 I2C 控制器工作在 Master 模式时的速率。此参数值的范围为 1~IC_MAX_SPEED_MODE。如果软件设定的值不在 1~IC_MAX_SPEED_MODE 范围内，硬件会将其更改为 IC_MAX_SPEED_MODE，以起到保护作用。</p> <p>01: 标准模式 (0 to 100 Kbit/s)</p> <p>10: 快速模式 (≤ 400 Kbit/s)</p> <p>11: 高速模式 (≤ 3.4 Mbit/s)</p>
MASTER_MODE	0	RW	0x1	<p>I2C Master 的使能位。</p> <p>0: 关闭 master 功能</p> <p>1: 使能 master 功能</p>

5.23.3.2 IC_TAR (0x04)

域	位	读写	复位值	描述
reserved	31:13	RO	0x0	保留
IC_10BITADDR_MASTER	12	RW	0x1	<p>选择工作在 I2C Master 时使用 7 位地址模式还是 10 位地址模式进行通信。</p> <p>0: 7 位地址模式</p> <p>1: 10 位地址模式</p> <p>声明：此位只有在 I2C_DYNAMIC_TAR_UPDATE 为 “Yes” 时才有效。</p>
SPECIAL	11	RW	0x0	<p>选择 I2C 通信使用广播呼叫地址格式还是使用 START BYTE 格式。</p> <p>0: 使用 IC_TAR 地址格式，忽略 GC_OR_START 设置；</p> <p>1: 使用 GC_OR_START 设定的格式。</p>
GC_OR_START	10	RW	0x0	<p>如果 bit[11] (SPECIAL) 为 1，该位设定 DW_apb_i2c 使用广播呼叫地址格式还是 START BYTE 格式。</p> <p>0: 使用广播呼叫地址格式</p> <p>此模式下只能进行写操作。如果尝试在此模式下进行读操作，则 IC_RAW_INTR_STAT 寄存器中的第 6 位 (TX_ABORT) 将会被置位。如果 SPECIAL 位一直为 1，I2C 控制器则会一直工作在这种模式下。</p> <p>1: START BYTE 格式</p>
IC_TAR	9:0	RW	0x55	<p>存放 Master 通信的目的地址。使用广播呼叫地址格式时此参数可以忽略，使用 START BYTE 格式时只需 CPU 向此</p>

域	位	读写	复位值	描述
				处进行一次写操作。

5.23.3.3 IC_SAR (0x08)

域	位	读写	复位值	描述
reserved	31:10	RO	0x0	保留
IC_SAR	9:0	RW	0x55	IC_SAR 存放 I2C 工作在 Slave 模式下的 Slave 地址。7 位地址模式下只使用 IC_SAR[6:0]。只有在关闭 I2C 接口功能时 (IC_ENABLE=0) 才能更新 IC_SAR 的值, 在 I2C 接口处于使能状态时不能改变 IC_SAR 的值。

5.23.3.4 IC_HS_MADDR (0x0C)

域	位	读写	复位值	描述
reserved	31:3	RO	0x0	保留
IC_HS_MAR	2:0	RW	0x1	I2C HS 模式主机编码。HS 模式主机代码保留 6 位 (00001xxx) 不用于从机寻址或其他用途。每一个主机都有一个特殊的主代码; 在相同的 I2C 总线下可以出现多达 8 个高速主机模式。有效值在 0~7 之间。如果将 IC_MAX_SPEED_MODE 配置的参数设置为 Standard (1) or Fast (2)。该寄存器值为 0。

5.23.3.5 IC_DATA_CMD (0x10)

域	位	读写	复位值	描述
reserved	31:11	RO	0x0	保留
RESTART	10	WO	0x0	该位设置是否在发送或接收一个字节数据前发起 RESTART, 且只有在 IC_EMPTYFIFO_HOLD_MASTER_EN 为 1 时有效。 1: 如果 IC_RESTART_EN =1, 不管传输方向与上次传输一致还是相反, 在发送或接收数据前会发起一个 RESTART; 如果 IC_RESTART_EN =0, 则使用 START/Stop 配对模式, 每次以 START 作为一次传输的开始, 以 Stop 结束一次传输。 0: 如果 IC_RESTART_EN =1, 则只有在传输方向与上次发生改变时发起一个 RESTART; 如果 IC_RESTART_EN =0, 则使用 START/Stop 配对模式, 每次以 START 作为一次传输的开始, 以 Stop 结束一次传输。
STOP	9	WO	0x0	此位设置是否在发送或接收到一个字节数据后发起 STOP, 且只有在 IC_EMPTYFIFO_HOLD_MASTER_EN 为 1 时有效。 1: 不管 Tx FIFO 是否为空, 在发送或接收数据后都会发起一个 STOP。如果 Tx FIFO 不为空, 则在发送或接收数据后, 总线的 Master 端会立即通过产生 START 和申请总

域	位	读写	复位值	描述
				线仲裁的方式开始一次新的通信。 0: 不管 Tx FIFO 是否为空, 在发送或接收数据后都不发起 STOP。如果 Tx FIFO 不为空, 则继续发送或接收当前通信的其他数据字节 (由 CMD 位决定是发送还是接收); 如果 Tx FIFO 为空, 总线的 Master 端会持续拉低 SCL 信号线并将总线挂起, 直到 Tx FIFO 中有新的有效值。
CMD	8	WO	0x0	此位是 I2C 控制器工作在 Master 模式时进行读写操作的控制位。控制器工作在 Slave 模式时, 此位值无效。 1: 读 0: 写 工作在 Slave 接收模式时不需要考虑 CMD 位的设定。工作在 Slave 发送模式时, CMD=0 表示 IC_DATA_CMD 中的数据将被发送。 在对 CMD 位进行操作时需要考虑以下情况: 无论 IC_RAW_INTR_STAT 中的 SPECIAL 位 (第 11 位) 是否被清 0, 在发送广播呼叫地址格式后进行读操作都会导致 TX_ABRT 中断被置位 (IC_RAW_INTR_STAT 寄存器中的第 6 位); 如果在收到 RD_REQ 中断后软件置 CMD 位为 1 也同样会导致 TX_ABRT 中断事件的发生, 即 TX_ABRT 位被置 1。
DAT	7:0	WO	0x0	DAT 中存放用来发送的数据或从 I2C 总线上接收到的数据。在开始一次读操作时向 DAT 中写入数据将被 DW_apb_i2c 忽略, 但此时从 DAT 读取的数据则是从 I2C 总线接口接收到的数据。

5.23.3.6 IC_SS_SCL_HCNT (0x14)

域	位	读写	复位值	描述
reserved	31:16	RW	0x0	保留
IC_SS_SCL_HCNT	15:0	RW	0x190	该寄存器必须在 I2C 总线传输之前进行设计, 用于明确正确的 I/O 时序。该寄存器用于设置标准速率下 SCL 高电平持续时间的计数值。 该寄存器仅当 I2C 接口在不使能情况下 (当 IC_ENABLE=0 时) 可写, 其他情况下的写操作无效。寄存器最小取值为 6, 比 6 小的值无法设置, 若设置值小于 6, 则硬件将寄存器值设置为 6。当 APB_DATA_WIDTH=8 时, 寄存器设置的顺序尤为关键, 此时, 首先应配置计数器的低 32 位数据, 之后再配置高 32 位。当 IC_HC_COUNT_VALUES 为 1 时, 该寄存器只读。

5.23.3.7 IC_SS_SCL_LCNT (0x18)

域	位	读写	复位值	描述
reserved	31:16	RW	0x0	保留
IC_SS_SCL_LCNT	15:0	RW	0x1d6	该寄存器必须在 I2C 总线传输之前进行设计, 用于明确

域	位	读写	复位值	描述
				<p>正确的 I/O 时序。该寄存器用于设置标准速率下 SCL 低电平持续时间的计数值。</p> <p>该寄存器仅当 I2C 接口在不使能情况下(当 IC_ENABLE=0 时)可写, 其他情况下的写操作无效。寄存器最小取值为 8, 比 8 小的值无法设置, 若设置值小于 8, 则硬件将寄存器值设置为 8。当 APB_DATA_WIDTH=8 时, 寄存器设置的顺序尤为关键, 此时, 首先应配置计数器的低 32 位数据, 之后再配置高 32 位。当 IC_HC_COUNT_VALUES 为 1 时, 该寄存器只读。</p>

5.23.3.8 IC_FS_SCL_HCNT (0x1C)

域	位	读写	复位值	描述
reserved	31:16	RW	0x0	保留
IC_FS_SCL_HCNT	15:0	RW	0x3c	<p>该寄存器必须在 I2C 总线传输之前进行设计, 用于明确正确的 I/O 时序。该寄存器用于设置快速模式下 SCL 高电平持续时间的计数值。用于发送高速模式下的 Mater Code 和 START BYTE 或 General Call。</p> <p>当 IC_MAX_SPEED_MODE=standard, 此寄存器为只读且返回值为全 0。该寄存器仅当 I2C 接口在不使能情况下(当 IC_ENABLE=0 时)可写。其他情况下的写操作无效。寄存器最小取值为 6, 比 6 小的值无法设置, 若设置值小于 6, 则硬件将寄存器值设置为 6。当 APB_DATA_WIDTH =8 时, 寄存器设置的顺序尤为关键, 此时, 首先应配置计数器的低字节(8 位)数据, 之后再配置高字节(8 位)。当 IC_HC_COUNT_VALUES 为 1 时, 该寄存器只读。</p>

5.23.3.9 IC_FS_SCL_LCNT (0x20)

域	位	读写	复位值	描述
reserved	31:16	RW	0x0	保留
IC_FS_SCL_LCNT	15:0	RW	0x82	<p>该寄存器必须在 I2C 总线传输之前进行设计, 用于明确正确的 I/O 时序。该寄存器用于设置快速模式下 SCL 低电平持续时间的计数值。用于发送高速模式下的 Mater Code 和 START BYTE 或 General Call。</p> <p>当 IC_MAX_SPEED_MODE=standard, 此寄存器为只读且返回值为全 0。该寄存器仅当 I2C 接口在不使能情况下(当 IC_ENABLE=0 时)可写。其他情况下的写操作无效。寄存器最小取值为 8, 比 8 小的值无法设置, 若设置值小于 8, 则硬件将寄存器值设置为 8。当 APB_DATA_WIDTH=8 时, 寄存器设置的顺序尤为关键, 此时, 首先应配置计数器的低字节(8 位)数据, 之后再配置高 32 位字节(8 位)。当 IC_HC_COUNT_VALUES 为 1 时, 该寄存器只读。</p>

5.23.3.10 IC_HS_SCL_HCNT (0x24)

域	位	读写	复位值	描述
reserved	31:16	RW	0x0	保留
IC_HS_SCL_HCNT	15:0	RW	0x06	该寄存器必须在 I2C 总线传输之前进行设计，用于明确正确的 I/O 时序。该寄存器用于设置高速模式下 SCL 高电平持续时间的计数值。 SCL 高电平时间依赖于总线负载情况。接 100pF 负载时，高电平时间为 60ns；接 400pF 负载时，高电平时间为 120ns。IC_MAX_SPEED_MODE != high 时，此寄存器为只读且返回值为全 0。该寄存器仅当 I2C 接口在不使能情况下（当 IC_ENABLE=0 时）可写。其他情况下写操作无效。寄存器最小取值为 6，比 6 小的值无法设置，若设置值小于 6，则硬件将寄存器值设置为 6。当 APB_DATA_WIDTH=8 时，寄存器设置的顺序尤为关键，此时，首先应配置计数器的低字节（8 位）数据，之后再配置高字节（8 位）。当 IC_HC_COUNT_VALUES 为 1 时，该寄存器只读。

5.23.3.11 IC_HS_SCL_LCNT (0x28)

域	位	读写	复位值	描述
reserved	31:16	RW	0x0	保留
IC_HS_SCL_LCNT	15:0	RW	0x10	该寄存器必须在 I2C 总线传输之前进行设计，用于明确正确的 I/O 时序。该寄存器用于设置高速模式下 SCL 低电平持续时间的计数值。 SCL 低电平时间依赖于总线负载情况。接 100pF 负载时，低电平时间为 160ns；接 400pF 负载时，低电平时间为 320ns。IC_MAX_SPEED_MODE != high 时，此寄存器为只读且返回值为全 0。该寄存器仅当 I2C 接口在不使能情况下（当 IC_ENABLE=0 时）可写。其他情况下写操作无效。当 APB_DATA_WIDTH=8 时，寄存器设置的顺序尤为关键，此时，首先应配置计数器的低字节（8 位）数据，之后再配置高字节（8 位）。寄存器最小取值为 8，比 8 小的值无法设置，若设置值小于 8，则硬件将寄存器值设置为 8。

5.23.3.12 IC_INTR_STAT (0x2C)

域	位	读写	复位值	描述
reserved	31:12	R0	0x0	保留
R_GEN_CALL	11	R0	0x0	中断屏蔽配置作用后的中断状态。当寄存器 IC_INTR_MASK 的某一位置 0，如果有相应中断事件发生，IC_INTR_STAT 对应的中断标志位就不会置位（值为 0），而 IC_RAW_INTR_STAT 的中断标志位仍正常置位（值为 1）。
R_START_DET	10	R0	0x0	
R_STOP_DET	9	R0	0x0	
R_ACTIVITY	8	R0	0x0	
R_RX_DONE	7	R0	0x0	
R_TX_ABRT	6	R0	0x0	

R_RD_REQ	5	R0	0x0
R_TX_EMPTY	4	R0	0x0
R_TX_OVER	3	R0	0x0
R_RX_FULL	2	R0	0x0
R_RX_OVER	1	R0	0x0
R_RX_UNDER	0	R0	0x0

5.23.3.13 IC_INTR_MASK (0x30)

域	位	读写	复位值	描述
reserved	31:12	R0	0x0	保留
M_GEN_CALL	11	RW	0x1	中断事件标志屏蔽控制。置 0 时，如果对应中断事件发生，不会置位 IC_INTR_STAT 寄存器中对应的中断标志位。
M_START_DET	10	RW	0x0	
M_STOP_DET	9	RW	0x0	
M_ACTIVITY	8	RW	0x0	
M_RX_DONE	7	RW	0x1	
M_TX_ABRT	6	RW	0x1	
M_RD_REQ	5	RW	0x1	
M_TX_EMPTY	4	RW	0x1	
M_TX_OVER	3	RW	0x1	
M_RX_FULL	2	RW	0x1	
M_RX_OVER	1	RW	0x1	
M_RX_UNDER	0	RW	0x1	

5.23.3.14 IC_RAW_INTR_STAT (0x34)

域	位	读写	复位值	描述
reserved	31:12	R0	0x0	保留
GEN_CALL	11	R0	0x0	只有接收并识别到 General Call 格式时才会被置位。一旦 GEN_CALL 置位，则只有通过关闭 I2C 控制器或 CPU 读取 IC_CLR_GEN_CALL 寄存器中的第 0 位，GEN_CALL 位才能被清 0。I2C 控制器会把接收到的数据存放在 rx 缓冲区中。
START_DET	10	R0	0x0	此位状态表示在 I2C 总线接口上是否产生了 START 或 RESTART。与控制器工作在 Master 还是 Slave 模式无关。 1：总线产生了 START 或 RESTART 0：总线没有 START 或 RESTART 产生
STOP_DET	9	R0	0x0	此位状态表示在 I2C 总线接口上是否产生了 STOP。与控制器工作在 Master 模式还是 Slave 模式无关。 1：总线产生了 STOP 0：总线没有 STOP 产生
ACTIVITY	8	R0	0x0	此位表示 I2C 控制器的活动状态。 有 4 种方法可以清除 ACTIVITY 标志： •关闭 i2c •读取 IC_CLR_ACTIVITY 寄存器

域	位	读写	复位值	描述
				<ul style="list-style-type: none"> •读取 IC_CLR_INTR 寄存器 •系统复位 一旦被置位则会一致保持置位，直到通过以上四种方式中的一种将其标志清 0。即使在 Idle 状态下如果采取清 0 动作的话也会一直保持置位。
RX_DONE	7	R0	0x0	I2C 控制器工作在 Slave 发送模式下，发送完数据的最后一个字节后，在规定时间内没有收到 Master 端的回应（ACK），RX_DONE 将会被置位表示结束。
TX_ABRT	6	R0	0x0	该位表明如果 DW_apb_i2c 不能完成所期望的对传输 FIFO 内容的操作。这种情况可能发生在 I2C 作为主机或从机上，称作“transmit abort”。当位置 1，IC_TX_ABRT_SOURCE 寄存器将指出 transmit abort 发生的原因。
RD_REQ	5	R0	0x0	读请求标志。当 I2C 控制器工作在 Slave 模式下，且有 Master 尝试从 DW_apb_i2c 中读取数据时，RD_REQ 被置位。I2C 控制器在处理 RD_REQ 请求期间会将 SCL 保持低电平。RD_REQ 是处理器必须响应的中断请求，并在请求处理完成时把 Master 所要的数据放到 IC_DATA_CMD 寄存器中。读取 IC_CLR_RD_REQ 寄存器的值可以将 RD_REQ 标志清 0。
TX_EMPTY	4	R0	0x0	当发送缓冲区小于等于 IC_TX_TL 寄存器中设定的门限值时将置位 TX_EMPTY。当缓冲区大于门限值时，硬件会自动把 TX_EMPTY 清 0。IC_ENABLE bit0=0 时，TXFIFO 被刷新复位，TXFIFO 可以认为为空，此时 TX_EMPTY 被置为 1。当总线处于非活动状态时 ic_en=0，TX_EMPTY=0。
TX_OVER	3	R0	0x0	在发送过程中，如果发送缓冲区大小达到 IC_TX_BUFFER_DEPTH 且处理器还在尝试通过向 IC_DATA_CMD 中写数据来发起另一个 I2C 命令时，TX_OVER 被置位。即使在控制器功能被关闭的情况下（IC_ENABLE[0]=0）RX_OVER 状态也会一直保持置位，直到总线进入空闲状态。ic_en=0 时，TX_OVER 被清 0。
RX_FULL	2	R0	0x0	当接收缓冲区大于等于 IC_RX_TL 中设定的门限值（RX_TL）时，RX_FULL 置位。当缓冲区小于门限值时，硬件会自动把 RX_FULL 清 0。IC_ENABLE bit0=0 时，RXFIFO 被刷新复位，RXFIFO 为空，此时 RX_FULL 被清 0。
RX_OVER	1	R0	0x0	当接收缓冲区大小达到 IC_RX_BUFFER_DEPTH，且还继续从外部接收数据时，RX_OVER 置位。TX_OVER 事件会被 I2C 控制器响应，且在缓冲区满后接收到的所有数据均被丢弃。即使在控制器功能被关闭的情况下（IC_ENABLE[0]=0）RX_OVER 状态也会一直保持置位，直到总线进入空闲状态。ic_en= 0 时，RX_OVER 被清 0。
RX_UNDER	0	R0	0x0	处理器通过访问 IC_DATA_CMD 寄存器获取接收缓冲区的数据时，若接收缓冲区为空，RX_UNDER 被置位。即使在控制器功能被关闭的情况下（IC_ENABLE[0]=0）RX_UNDER 状态也会一直保持置位，直到总线进入空闲状态。ic_en=0 时，RX_UNDER 被清 0。

5.23.3.15 IC_RX_TL (0x38)

域	位	读写	复位值	描述
reserved	31:8	R0	0x0	保留
RX_TL	7:0	RW	0x0	接收缓冲区满中断 (RX_FULL) 触发门限控制。有效范围 0~7, 但最大值不能超出缓冲区的深度。如果设定值超出缓冲区的最大深度, 其实际设置的有效大小为缓冲区的最大深度值。0 表示接收缓冲区大于等于 1 时触发中断, 7 表示接收缓冲区大于等于 8 时触发中断。

5.23.3.16 IC_TX_TL (0x3C)

域	位	读写	复位值	描述
reserved	31:8	R0	0x0	保留
TX_TL	7:0	RW	0x0	发送缓冲区空中断 (TX_EMPTY) 触发门限控制。有效范围 0~7, 但最大值不能超出缓冲区的深度。如果设定值超出缓冲区的最大深度, 其实际设置的有效大小为缓冲区的最大深度值。0 表示发送缓冲区小于等于 0 时触发中断, 7 表示发送缓冲区小于等于 7 时触发中断。

5.23.3.17 IC_CLR_INTR (0x40)

域	位	读写	复位值	描述
reserved	31:1	R0	0x0	保留
CLR_INTR	0	R0	0x0	读取此寄存器以清除组合中断, 即所有的个别中断, 和 IC_TX_ABRT_SOURCE 寄存器。不清除硬件可以清除的中断但清除软件可以清除的中断。

5.23.3.18 IC_CLR_RX_UNDER (0x44)

域	位	读写	复位值	描述
reserved	31:1	R0	0x0	保留
CLR_RX_UNDER	0	R0	0x0	读取这个寄存器以清除 IC_RAW_INTR_STAT 寄存器的 RX_UNDER 中断 (bit 0)。

5.23.3.19 IC_CLR_RX_OVER (0x48)

域	位	读写	复位值	描述
reserved	31:1	R0	0x0	保留
CLR_RX_OVER	0	R0	0x0	读取这个寄存器以清除 IC_RAW_INTR_STAT 寄存器的 RX_OVER 中断 (bit1)。

5.23.3.20 IC_CLR_TX_OVER (0x4C)

域	位	读写	复位值	描述
reserved	31:1	R0	0x0	保留
CLR_TX_OVER	0	R0	0x0	读取这个寄存器以清除 IC_RAW_INTR_STAT 寄存器的 TX_OVER 中断 (bit3)。

5.23.3.21 IC_CLR_RD_REQ (0x50)

域	位	读写	复位值	描述
reserved	31:1	R0	0x0	保留
CLR_RD_REQ	0	R0	0x0	读取这个寄存器以清除 IC_RAW_INTR_STAT 寄存器的 RD_REQ 中断 (bit5)。

5.23.3.22 IC_CLR_TX_ABRT (0x54)

域	位	读写	复位值	描述
reserved	31:1	R0	0x0	保留
CLR_TX_ABRT	0	R0	0x0	读这个寄存器来清除 IC_RAW_INTR_STAT 寄存器和 IC_TX_ABRT_SOURCE 寄存器的中断 TX_ABRT (bit 6)。这还会从刷新/重置状态释放 TX FIFO，从而准许更多的写入 TX FIFO。

5.23.3.23 IC_CLR_RX_DONE (0x58)

域	位	读写	复位值	描述
reserved	31:1	R0	0x0	保留
CLR_RX_DONE	0	R0	0x0	读取这个寄存器以清除 IC_RAW_INTR_STAT 寄存器的 RX_DONE 中断 (bit7)。

5.23.3.24 IC_CLR_ACTIVITY (0x5C)

域	位	读写	复位值	描述
reserved	31:1	R0	0x0	保留
CLR_ACTIVITY	0	R0	0x0	如果 I2C 不处于活动状态，读取这个寄存器将清除 ACTIVITY 中断。如果 I2C 模块仍然在总线上处于活动状态，则继续设置 ACTIVITY 中断位。如果模块被禁用并且总线上没有其他活动则由硬件自动清除模块。该寄存器中的读取的值，以获取 IC_RAW_INTR_STAT 寄存器在 ACTIVITY 中断 (bit 8) 的状态。

5.23.3.25 IC_CLR_STOP_DET (0x60)

域	位	读写	复位值	描述
reserved	31:1	R0	0x0	保留
CLR_STOP_DET	0	R0	0x0	读这个寄存器来清除 IC_RAW_INTR_STAT 状态寄存器的 STOP_DET 中断(bit 9)。

5.23.3.26 IC_CLR_START_DET (0x64)

域	位	读写	复位值	描述
reserved	31:1	R0	0x0	保留
CLR_START_DET	0	R0	0x0	读这个寄存器来清除 IC_RAW_INTR_STAT 状态寄存器的 START_DET 中断(bit 10)。

5.23.3.27 IC_CLR_GEN_CALL (0x68)

域	位	读写	复位值	描述
reserved	31:1	R0	0x0	保留
CLR_GEN_CALL	0	R0	0x0	读这个寄存器来清除 IC_RAW_INTR_STAT 状态寄存器的 GEN_CALL 中断(bit 11)。

5.23.3.28 IC_ENABLE (0x6C)

域	位	读写	复位值	描述
reserved	31:1	R0	0x0	保留
ENABLE	0	RW	0x0	<p>I2C 控制器使能或关闭控制位。</p> <p>0：关闭 I2C 控制器功能；</p> <p>1：使能 I2C 控制器功能。</p> <p>以下现象会在 I2C 控制器功能关闭时出现：</p> <ul style="list-style-type: none"> • TXFIFO 和 RXFIFO 被刷新。 • IC_INTR_STAT 寄存器中的状态保持不变。 <p>在控制器发送数据过程中关闭 I2C 控制器功能，则在当前发送操作完成后，清空发送缓冲区中的内容。</p> <p>在控制器接收数据过程中关闭 I2C 控制器功能，通信将在接收完当前字节后停止，且不响应使用 asynchronous pclk and ic_clk 的系统（IC_CLK_TYPE=1）。在使能或关闭控制器时有 2 个 ic_clk 的延迟。</p>

5.23.3.29 IC_STATUS (0x70)

域	位	读写	复位值	描述
reserved	31:7	R0	0x0	保留
SLV_ACTIVITY	6	R0	0x0	Slave FSM 活动状态标志。Slave FSM(Slave Finite State Machine 不在 Idle 状态时被置位。

域	位	读写	复位值	描述
				0: Slave FSM 处于 Idle 状态, 此时 I2C 控制器的 Slave 功能处于非活动状态; 1: Slave FSM 处于非 Idle 状态, 此时 I2C 控制器的 Slave 功能处于活动状态。
MST_ACTIVITY	5	R0	0x0	Master FSM 活动状态标志。Master FSM (Master Finite State Machine) 处于非 Idle 状态时被置位。 0: Master FSM 处于 Idle 状态, 此时 I2C 控制器的 Master 功能处于非活动状态; 1: Master FSM 处于非 Idle 状态, 此时 I2C 控制器 Master 功能处于活动状态。
RFF	4	R0	0x0	接收 FIFO 全满标志。当接收 FIFO 全满时置位; FIFO 中有一个或一个以上为空时 0。 0: 接收 FIFO 未滿 1: 接收 FIFO 全滿
RFNE	3	R0	0x0	接收 FIFO 不为空标志。当接收 FIFO 不为空时置位, 为空时清 0。 0: 接收 FIFO 为空 1: 接收 FIFO 不为空
TFE	2	R0	0x1	发送 FIFO 全空标志。发送 FIFO 全空时置位; 发送 FIFO 有一个或一个以上不为空的值时清 0。此标志的产生不伴随有中断发生。 0: 发送 FIFO 不为空 1: 发送 FIFO 为空
TFNF	1	R0	0x1	发送 FIFO 未滿标志。发送 FIFO 中有一个或一个以上位置为空时置位; 发送 FIFO 满时清 0。 0: 发送 FIFO 已滿 1: 发送 FIFO 未滿
ACTIVITY	0	R0	0x0	I2C 控制器活动状态标志。 1: 表示控制器正在进行数据传输, 主从模式皆有效。 0: 表示控制器处于非活动状态。

5.23.3.30 IC_TXFLR (0x74)

域	位	读写	复位值	描述
reserved	31:3	R0	0x0	保留
TXFLR	2:0	R0	0x0	发送 FIFO 阈值。发送 FIFO 中的有效数据量。

5.23.3.31 IC_RXFLR (0x78)

域	位	读写	复位值	描述
reserved	31:3	R0	0x0	保留
RXFLR	2:0	R0	0x0	接收 FIFO 阈值。接收 FIFO 中的有效数据量。

5.23.3.32 IC_SDA_HOLD (0x7C)

域	位	读写	复位值	描述
reserved	31:16	R0	0x0	保留
IC_SDA_HOLD	15:0	RW	0x1	设置所需的 SDA 保持时间以 ic_clk 周期为单位。

5.23.3.33 IC_TX_ABRT_SOURCE (0x80)

域	位	读写	复位值	描述
reserved	31:16	R0	0x0	保留
ABRT_SLVRD_INTX	15	R0	0x0	1: 当处理器端响应从模式请求将数据传送到远程主机并且用户在 IC_DATA_CMD 寄存器写入 1。
ABRT_SLV_ARBLOST	14	R0	0x0	1: 从机在传输数据给远程主机时丢失总线占用, 同时 IC_TX_ABRT_SOURCE[12] 被设置。
ABRT_SLVFLUSH_TXFIFO	13	R0	0x0	1: 从设备接收到一个读命令并且发送 FIFO 有数据, 从设备发出 TX_ABRT 中断来刷新发送 FIFO 的数据。
ARB_LOST	12	R0	0x0	1: 主机失去了仲裁, 或者 IC_TX_ABRT_SOURCE[14] 被设置, 从机发送丢失仲裁。
ABRT_MASTER_DIS	11	R0	0x0	1: 用户试图禁用主模式的情况下禁用主操作。适用主机发送或从机发送模式。
ABRT_10B_RD_NORSTRT	10	R0	0x0	1: Restart 被禁用 (IC_RESTART_EN bit (IC_CON[5]) = 0) 并且主机已 10 位寻址模式发出命令。适用于主机接收模式。
ABRT_SBYTE_NORSTRT	9	R0	0x0	要清除该位, ABRT_SBYTE_NORSTRT 必须是确定的, 且 IC_CON[5]=1, 且 SPECIAL 位 (IC_TAR[11]) 必须清除, 或 GC_OR_STARTt (IC_TAR[10]) 清除。 1: Restart 位被禁用, 即 IC_Restart_en (IN_CON[5]=0) 时, 用户试图发送起始字节。适用于主机模式。
ABRT_HS_NORSTRT	8	R0	0x0	1: restart 被禁用, 即 (IC_RESTART_EN bit (IC_CON[5]) = 0), 用户尝试使用主机以高速模式传输数据。适用于主机发送和接收模式。
ABRT_SBYTE_ACKDET	7	R0	0x0	1: 主机发送一个 start 字节, 但是 start 字节已被发送 (错误行为)。适用于主机模式。
ABRT_HS_ACKDET	6	R0	0x0	1: 处于高速模式, 但是高速主编码已被识别 (错误行为)。适用于主机模式。
ABRT_GCALL_READ	5	R0	0x0	1: i2c 在主模式下发送了一个 General Call, 但用户在 General Call 之后的字节被编程为读 (IC_DATA_CMD[9] 置 1)。适用于主发送模式。
ABRT_GCALL_NOACK	4	R0	0x0	1: DW_apb_i2c 在主模式下发送了一个 General Call 并且没有从机在总线上承认。
ABRT_TXDATA_NOACK	3	R0	0x0	1: 是一个主模式位, 主机已收到地址的确认但是当他发送地址后面的数据字节时, 并没有收到

域	位	读写	复位值	描述
				来自远程从机的确认。适用于主机模式。
ABRT_10ADDR2_NOACK	2	RO	0x0	1: 主设备处于 10 位地址模式, 10 位地址的第二个地址字节未被任何从机承认。适用于主机的发送和接收。
ABRT_10ADDR1_NOACK	1	RO	0x0	1: 主设备处于 10 位地址模式, 10 位地址的第一个地址字节未被任何从机承认。适用于主机的发送和接收。
ABRT_7B_ADDR_NOACK	0	RO	0x0	1: 主设备处于 7 位地址模式, 地址发送未被任何从机承认。适用于主机的发送和接收。

5.23.3.34 IC_SLV_DATA_NACK_ONLY (0x84)

域	位	读写	复位值	描述
reserved	31:1	RO	0x0	保留
NACK	0	RW	0x0	生成 NACK。NACK 只发生在当 i2c 作为从机接收时。如果将这个寄存器置 1, 那么它只能在接收数据字节之后生成一个 NACK。因此数据传输被终止, 接收到的数据不会被推送到接收缓冲区。 当寄存器设置为 0 时, 它将根据正常条件产生 NACK/ACK。 1: 产生 NACK 0: 产生正常条件的 NACK/ACK

5.23.3.35 IC_DMA_CR (0x88)

域	位	读写	复位值	描述
reserved	31:2	RO	0x0	保留
TDMAE	1	RW	0x0	DMA 传输使能位。这个位可以启用/禁用发送 FIFO 的 DMA 通道。 0: 发送 DMA 禁用 1: 发送 DMA 启用
RDMAE	0	RW	0x0	接收 DMA 使能位。这个位可以启用/禁用接收 FIFO DMA 通道。 0: 接收 DMA 禁用 1: 接收 DMA 启用

5.23.3.36 IC_DMA_TDLR (0x8C)

域	位	读写	复位值	描述
reserved	31:2	RO	0x0	保留
DMATDL	1:0	RW	0x0	传输数据层。该位控制传输逻辑发出 DMA 请求的中有效项的数量等于或低于此字段值, 且 TDMAE = 1。

5.23.3.37 IC_DMA_RDLR (0x90)

域	位	读写	复位值	描述
reserved	31:2	R0	0x0	保留
DMARDL	1:0	RW	0x0	接收数据阈值。该位控制接收逻辑中的一个 DMA 请求的阈值。

5.23.3.38 IC_SDA_SETUP (0x94)

域	位	读写	复位值	描述
reserved	31:8	R0	0x0	保留
SDA_SETUP	7:0	RW	0x64	SDA 建立。建议如果所需延时为 1000ns，对于频率为 10MHz 的 ic_clk, IC_SDA_SETUP 编程为 11。IC_SDA_SETUP 必须以最小值 2 来编程。

5.23.3.39 IC_ACK_GENERAL_CALL (0x98)

域	位	读写	复位值	描述
reserved	31:1	R0	0x0	保留
ACK_GEN_CALL	0	RW	0x1	ACK General Call。置为 1 时，当 i2c 收到 General Call 时，i2c 以 ACK 响应。置 0 时 i2c 不生成 General Call 中断。

5.23.3.40 IC_ENABLE_STATUS (0x9C)

域	位	读写	复位值	描述
reserved	31:3	R0	0x0	保留
SLV_RX_DATA_LOST	2	R0	0x0	从机收到的数据丢失。
SLV_DISABLED_WHILE_BUSY	1	R0	0x0	从机在忙时禁用（发送、接收）。该位表示 I2C 从机在忙时，IC_ENABLE 寄存器由 1 设置成 0。
IC_EN	0	R0	0x0	ic_en 状态：I2C 控制器是否开启或关闭，默认 I2C 控制器功能关闭。 0：I2C 控制器功能关闭 1：I2C 控制器功能开启

5.23.3.41 IC_FS_SPKLEN (0xA0)

域	位	读写	复位值	描述
reserved	31:8	R0	0x0	保留
IC_FS_SPKLEN	7:0	RW	0x5	FS 模式下，在任何 I2C 总线事务发送之前，必须设置寄存器，保证稳定运行。此寄存器在设置时间，在 IC_CLK 周期中过滤掉的 SCL 或 SDA 线路中的尖峰。只有当 I2C 接口被禁用时，才能写该寄存器。其他时间的写入没有效果。最小有效值是 1。

5.23.3.42 IC_HS_SPKLEN (0xA4)

域	位	读写	复位值	描述
reserved	31:8	R0	0x0	保留
IC_HS_SPKLEN	7:0	RW	0x2	HS 模式下，在任何 I2C 总线事务发送之前，必须设置寄存器，保证稳定运行。此寄存器在设置时间，在 IC CLK 周期中过滤掉的 SCL 或 SDA 线路中的尖峰。只有当 I2C 接口被禁用时，才能写入该寄存器，该寄存器也对应于被设置的 IC 启用寄存器。其他时间的写入没有效果。最小有效值是 1。IC_MAX_SPEED_MODE 参数为 3 时该寄存器才有效。

5.23.4 SMBUS 寄存器列表

表 5-65 SMBUS 寄存器基地址

名称	基地址
SMBUS	0x000_2801_3000

表 5-66 SMBUS 寄存器列表

寄存器	偏移	描述
IC_CON	0x00	I2C 控制寄存器
IC_TAR	0x04	I2C 主机地址寄存器
IC_SAR	0x08	I2C 从机地址寄存器
IC_HS_MADDR	0x0C	I2C 高速主机模式编码地址寄存器
IC_DATA_CMD	0x10	I2C 数据寄存器
IC_SS_SCL_HCNT	0x14	标准模式 I2C 时钟信号 SCL 的高电平计数寄存器
IC_SS_SCL_LCNT	0x18	标准模式 I2C 时钟信号 SCL 的低电平计数寄存器
IC_FS_SCL_HCNT	0x1C	快速模式 I2C 时钟信号 SCL 的高电平计数寄存器
IC_FS_SCL_LCNT	0x20	快速模式 I2C 时钟信号 SCL 的低电平计数寄存器
IC_HS_SCL_HCNT	0x24	高速模式 I2C 时钟信号 SCL 的高电平计数寄存器
IC_HS_SCL_LCNT	0x28	高速模式 I2C 时钟信号 SCL 的低电平计数寄存器
IC_INTR_STAT	0x2C	I2C 中断状态（中断屏蔽配置作用后的中断状态）寄存器
IC_INTR_MASK	0x30	I2C 中断屏蔽寄存器
IC_RAW_INTR_STAT	0x34	I2C 原始中断状态（真实的中断状态）寄存器
IC_RX_TL	0x38	I2C 接收 FIFO 阈值寄存器
IC_TX_TL	0x3C	I2C 发送 FIFO 阈值寄存器
IC_CLR_INTR	0x40	I2C 清除组合和单独中断寄存器
IC_CLR_RX_UNDER	0x44	清除 RX_UNDER 中断寄存器
IC_CLR_RX_OVER	0x48	清除 RX_OVER 中断寄存器
IC_CLR_TX_OVER	0x4C	清除 TX_OVER 中断寄存器
IC_CLR_RD_REQ	0x50	清除 RD_REQ 中断寄存器
IC_CLR_TX_ABRT	0x54	清除 TX_ABRT 中断寄存器
IC_CLR_RX_DONE	0x58	清除 RX_DONE 中断寄存器
IC_CLR_ACTIVITY	0x5C	清除 ACTIVITY 中断寄存器

寄存器	偏移	描述
IC_CLR_STOP_DET	0x60	清除 STOP_DET 中断寄存器
IC_CLR_START_DET	0x64	清除 START_DET 中断寄存器
IC_CLR_GEN_CALL	0x68	清除 GEN_CALL 中断寄存器
IC_ENABLE	0x6C	I2C 使能寄存器
IC_STATUS	0x70	I2C 状态寄存器
IC_TXFLR	0x74	发送 FIFO 等级寄存器
IC_RXFLR	0x78	接收 FIFO 等级寄存器
IC_SDA_HOLD	0x7C	SDA 保持时间寄存器
IC_TX_ABRT_SOURCE	0x80	I2C 发送异常状态寄存器
IC_SLV_DATA_NACK_ONLY	0x84	产生 SLV_DATA_NACK 寄存器
IC_DMA_CR	0x88	DMA 控制寄存器
IC_DMA_TDLR	0x8C	DMA 发送数据阈值
IC_DMA_RDLR	0x90	DMA 接收数据阈值
IC_SDA_SETUP	0x94	I2CSDA 建立时间寄存器
IC_ACK_GENERAL_CALL	0x98	I2CACK_Gen_Call 寄存器
IC_ENABLE_STATUS	0x9C	I2C 使能状态寄存器
IC_FS_SPKLEN	0xA0	FS 模式尖峰滤波寄存器
IC_HS_SPKLEN	0xA4	HS 模式尖峰滤波寄存器
IC_SMBCLK_LOW_MEXT	0xA8	时钟 MEXT 参数寄存器
IC_SMBCLK_LOW_TIMEOUT	0xAC	时钟 TIMEOUT 参数寄存器
IC_SMBCLK_STUCK_TIMEOUT	0xB0	时钟扩展超时参数寄存器
IC_SMBDAT_STUCK_TIMEOUT	0xB4	数据扩展超时参数寄存器
IC_SMBCLK_LOW_SEXT	0xB8	SEXT 参数设置寄存器
CLR_SMMST_SCL_EXT_LOW_TIMEOUT	0xBC	清除 SMMST_SCL_EXT_LOW_TIMEOUT 中断寄存器
CLR_SMMST_SCL_TMO_LOW_TIMEOUT	0xC0	清除 SMMST_SCL_TMO_LOW_TIMEOUT 中断寄存器
CLR_SMMST_SDA_LOW_TIMEOUT	0xC4	清除 SMMST_SDA_LOW_TIMEOUT 中断寄存器
CLR_SMSLV_SCL_EXT_LOW_TIMEOUT	0xC8	清除 SMSLV_SCL_EXT_LOW_TIMEOUT 中断寄存器
CLR_SMSLV_SCL_TMO_LOW_TIMEOUT	0xCC	清除 SMSLV_SCL_TMO_LOW_TIMEOUT 中断寄存器
CLR_SMBALERT_IN_N	0xD0	清除 SMBALERT_IN_N 中断寄存器

5.23.5 SMBUS 寄存器说明

本章节仅描述发生位域更改的寄存器，以及新增的寄存器，其他寄存器请参考 5.25.3 I2C 寄存器说明章节。

5.23.5.1 IC_TAR (0x04)

域	位	读写	复位值	描述
reserved	31:14	R0	0x0	保留
smbus_mst_quick_cmd_en	13	RW	0x0	主机模式下，快速命令使能信号。
IC_10BITADDR_MASTER	12	RW	0x1	选择工作在 I2C Master 时使用 7 位地址模式还是 10 位地址模式进行通信。

域	位	读写	复位值	描述
				0: 7 位地址模式 1: 10 位地址模式 声明: 此位只有在 I2C_DYNAMIC_TAR_UPDATE 为 “Yes” 时才有效。
SPECIAL	11	RW	0x0	选择 I2C 通信使用广播呼叫地址格式还是使用 START BYTE 格式。 0: 使用 IC_TAR 地址格式, 忽略 GC_OR_START 设置 1: 使用 GC_OR_START 设定的格式
GC_OR_START	10	RW	0x0	如果位 11 (SPECIAL) 为 1, 该位设定 DW_apb_i2c 使用广播呼叫地址格式还是 START BYTE 格式。 0: 使用广播呼叫地址格式。此模式下只能进行写操作。如果尝试在此模式下进行读操作, 则 IC_RAW_INTR_STAT 寄存器中的第 6 位 (TX_ABRT) 将会被置位。如果 SPECIAL 位一直为 1, I2C 控制器则会一直工作在这种模式下。 1: START BYTE 格式
IC_TAR	9:0	RW	0x55	存放 Master 通信的目的地址。使用广播呼叫地址格式时此参数可以忽略, 使用 START BYTE 格式时只需 CPU 向此处进行一次写操作。

5.23.5.2 IC_INTR_STAT (0x2C)

域	位	读写	复位值	描述
reserved	31:18	R0	0x0	保留
r_smbalert_in_n	17	R0	0x0	警报中断
r_smslv_scl_tmo_low_timeout	16	R0	0x0	仅在从机模式, 时钟超时中断
r_smslv_scl_ext_low_timeout	15	R0	0x0	从机时钟扩展超时中断, 主机和从机模式下
r_smmst_sda_low_timeout	14	R0	0x0	数据超时中断
r_smmst_scl_tmo_low_timeout	13	R0	0x0	仅在主机模式下, 时钟超时中断
r_smmst_scl_ext_low_timeout	12	R0	0x0	仅在主机模式下, 时钟扩展超时中断
R_GEN_CALL	11	R0	0x0	GEN_CALL 中断状态
R_START_DET	10	R0	0x0	START_DET 中断状态
R_STOP_DET	9	R0	0x0	STOP_DET 中断状态
R_ACTIVITY	8	R0	0x0	ACTIVITY 中断状态
R_RX_DONE	7	R0	0x0	RX_DONE 中断状态
R_TX_ABRT	6	R0	0x0	TX_ABRT 中断状态
R_RD_REQ	5	R0	0x0	RD_REQ 中断状态
R_TX_EMPTY	4	R0	0x0	TX_EMPTY 中断状态
R_TX_OVER	3	R0	0x0	TX_OVER 中断状态
R_RX_FULL	2	R0	0x0	RX_FULL 中断状态
R_RX_OVER	1	R0	0x0	RX_OVER 中断状态
R_RX_UNDER	0	R0	0x0	RX_UNDER 中断状态

5.23.5.3 IC_INTR_MASK (0x30)

域	位	读写	复位值	描述
reserved	31:18	RW	0x0	保留
m_smbalert_in_n	17	RW	0x1	屏蔽警报中断
m_smslv_scl_tmo_low_timeout	16	RW	0x1	屏蔽从机模式时钟超时中断
m_smslv_scl_ext_low_timeout	15	RW	0x1	屏蔽从机时钟扩展超时中断
m_smmst_sda_low_timeout	14	RW	0x1	屏蔽数据超时中断
m_smmst_scl_tmo_low_timeout	13	RW	0x1	屏蔽主机时钟超时中断
m_smmst_scl_ext_low_timeout	12	RW	0x1	屏蔽主机时钟扩展超时中断
M_GEN_CALL	11	RW	0x1	中断事件标志屏蔽控制。置 0 时，如果对应中断事件发生，不会置位 IC_INTR_STAT 寄存器中对应的中断标志位。
M_START_DET	10	RW	0x0	
M_STOP_DET	9	RW	0x0	
M_ACTIVITY	8	RW	0x0	
M_RX_DONE	7	RW	0x1	
M_TX_ABRT	6	RW	0x1	
M_RD_REQ	5	RW	0x1	
M_TX_EMPTY	4	RW	0x1	
M_TX_OVER	3	RW	0x1	
M_RX_FULL	2	RW	0x1	
M_RX_OVER	1	RW	0x1	
M_RX_UNDER	0	RW	0x1	

5.23.5.4 IC_RAW_INTR_STAT (0x34)

域	位	读写	复位值	描述
reserved	31:18	R0	0x0	保留
smbalert_in_n	17	R0	0x0	警报中断
smslv_scl_tmo_low_timeout	16	R0	0x0	仅在从机模式，时钟超时中断
smslv_scl_ext_low_timeout	15	R0	0x0	从机时钟扩展超时中断，主机和从机模式下
smmst_sda_low_timeout	14	R0	0x0	数据超时中断
smmst_scl_tmo_low_timeout	13	R0	0x0	仅在主机模式下，时钟超时中断
smmst_scl_ext_low_timeout	12	R0	0x0	仅在主机模式下，时钟扩展超时中断
GEN_CALL	11	R0	0x0	只有接收并识别到 General Call 格式时才会被置位。一旦 GEN_CALL 置位，则只有通过关闭 I2C 控制器或 CPU 读取 IC_CLR_GEN_CALL 寄存器中的第 0 位，GEN_CALL 位才能被清 0。I2C 控制器会把接收到的数据存放在 Rx 缓冲区中。
START_DET	10	R0	0x0	此位状态表示在 I2C 总线接口上是否产生了 START 或

域	位	读写	复位值	描述
				RESTART。与控制器工作在 Master 模式还是 Slave 模式无关。
STOP_DET	9	R0	0x0	此位状态表示在 I2C 总线接口上是否产生了 STOP。与控制器工作在 Master 模式还是 Slave 模式无关。
ACTIVITY	8	R0	0x0	<p>此位表示 I2C 控制器的活动状态。</p> <p>有 4 种方法可清除 ACTIVITY 标志：</p> <ul style="list-style-type: none"> •关闭 i2c •读取 IC_CLR_ACTIVITY 寄存器 •读取 IC_CLR_INTR 寄存器 •系统复位 <p>一旦被置位则会一致保持置位，直到通过以上四种方式中的一种将其标志清 0。即使在 Idle 状态下如果采取清 0 动作的话也会一直保持置位。</p>
RX_DONE	7	R0	0x0	I2C 控制器工作在 Slave 发送模式下，发送完数据的最后一个字节后，在规定时间内没有收到 Master 端的回应（ACK），RX_DONE 将会被置位表示结束。
TX_ABRT	6	R0	0x0	该位表明如果 DW_apb_i2c 不能完成所期望的对传输 FIFO 内容的操作。这种情况可能发生在 I2C 作为主机或从机上，叫作“transmit abort”。当位置 1，IC_TX_ABRT_SOURCE 寄存器将指出 ransmit abort 发生的原因。
RD_REQ	5	R0	0x0	读请求标志。当 I2C 控制器工作在 Slave 模式下，且有 Master 尝试从 DW_apb_i2c 中读取数据时，RD_REQ 被置位。I2C 控制器在处理 RD_REQ 请求期间会将 SCL 保持低电平。RD_REQ 是处理器必须响应的中断请求，并在请求处理完成时把 Master 所要的数据放到 IC_DATA_CMD 寄存器中。读取 IC_CLR_RD_REQ 寄存器的值可以将 RD_REQ 标志清 0。
TX_EMPTY	4	R0	0x0	当发送缓冲区小于等于 IC_TX_TL 寄存器中设定的门限值时将置位 TX_EMPTY。当缓冲区大于门限值时，硬件会自动把 TX_EMPTY 清 0。IC_ENABLE bit0=0 时，TXFIFO 被刷新复位，TXFIFO 可以认为为空，此时 TX_EMPTY 被置为 1。当总线处于非活动状态时 ic_en=0，TX_EMPTY=0。
TX_OVER	3	R0	0x0	在发送过程中，如果发送缓冲区大小达到 IC_TX_BUFFER_DEPTH 且处理器还在尝试通过向 IC_DATA_CMD 中写数据来发起另一个 I2C 命令时，TX_OVER 被置位。即使在控制器功能被关闭的情况下（IC_ENABLE[0]=0）RX_OVER 状态也会一直保持置位，直到总线进入空闲状态。ic_en =0 时，TX_OVER 被清 0。
RX_FULL	2	R0	0x0	当接收缓冲区大于等于 IC_RX_TL 中设定的门限值（RX_TL）时，RX_FULL 置位。当缓冲区小于门限值时，硬件会自动把 RX_FULL 清 0。IC_ENABLE bit0=0 时，RXFIFO 被刷新复位，RXFIFO 为空，此时 RX_FULL 被清 0。
RX_OVER	1	R0	0x0	当接收缓冲区大小达到 IC_RX_BUFFER_DEPTH，且还继续从外部接收数据时，RX_OVER 置位。TX_OVER 事件会被 I2C 控制器响应，且在缓冲区满后接收到的所有数据均被丢

域	位	读写	复位值	描述
				弃。即使在控制器功能被关闭的情况下 (IC_ENABLE[0]=0) RX_OVER 状态也会一直保持置位，直到总线进入空闲状态。ic_en=0 时，RX_OVER 被清 0。
RX_UNDER	0	RO	0x0	处理器通过访问 IC_DATA_CMD 寄存器获取接收缓冲区的数据时，若接收缓冲区为空，RX_UNDER 被置位。即使在控制器功能被关闭的情况下 (IC_ENABLE[0]=0) RX_UNDER 状态也会一直保持置位，直到总线进入空闲状态。ic_en=0 时，RX_UNDER 被清 0。

5.23.5.5 IC_ENABLE (0x6C)

域	位	读写	复位值	描述
reserved	31:11	RO	0x0	保留
smbus_slv_release	10	RW	0x0	从机模式下，释放总线使能信号。
smbus_slv_alert_en	9	RW	0x0	从机模式下，alert_en 信号配置。
smbus_slv_sckextend_en	8	RW	0x0	从机时钟扩展超时检测使能信号，主机模式下，产生停止信号，结束传输，从机模式下，释放总线。
smbus_slv_scktimeout_en	7	RW	0x0	时钟超时检测使能信号，从机模式下释放总线。
smbus_mst_clkreset_en	6	RW	0x0	主机模式下，将时钟拉低超时后，终止传输，恢复总线使能信号。
smbus_mst_alert_en	5	RW	0x0	主机模式下，alert_en 信号检测使能信号。
smbus_mst_release	4	RW	0x0	主机模式下发送停止信号，结束传输使能信号。
smbus_mst_sdatimout_en	3	RW	0x0	主机模式下，数据超时检测使能信号。
smbus_mst_sckextend_en	2	RW	0x0	主机模式下，时钟扩展超时检测使能信号。
smbus_mst_scktimeout_en	1	RW	0x0	主机模式下，时钟超时检测使能信号
ENABLE	0	RO	0x0	<p>I2C 控制器使能或关闭控制位。</p> <p>0：关闭 I2C 控制器功能</p> <p>1：使能 I2C 控制器功能</p> <p>以下现象会在 I2C 控制器功能关闭时出现：</p> <ul style="list-style-type: none"> • TXFIFO 和 RXFIFO 被刷新 • IC_INTR_STAT 寄存器中的状态保持不变 <p>在控制器发送数据过程中关闭 I2C 控制器功能，则在当前发送操作完成后，清空发送缓冲区中的内容。在控制器接收数据过程中关闭 I2C 控制器功能，通信将在接收完当前字节后停止，且不响应使用 asynchronous pclk and ic_clk 的系统 (IC_CLK_TYPE=1)。在使能或关闭控制器时有 2 个 ic_clk 的延迟。</p>

5.23.5.6 IC_SMBCLK_LOW_MEXT (0xA8)

域	位	读写	复位值	描述
IC_SMBCLK_LOW_MEXT	31:0	RW	0xFFFFFFFF	时钟 MEXT 参数设置寄存器

5.23.5.7 IC_SMBCLK_LOW_TIMEOUT (0xAC)

域	位	读写	复位值	描述
IC_SMBCLK_LOW_TIMEOUT	31:0	RW	0xFFFFFFFF	时钟 TIMEOUT 参数设置寄存器

5.23.5.8 IC_SMBCLK_STUCK_TIMEOUT (0xB0)

域	位	读写	复位值	描述
IC_SMBCLK_STUCK_TIMEOUT	31:0	RW	0xFFFFFFFF	时钟扩展超时参数设置寄存器

5.23.5.9 IC_SMBDAT_STUCK_TIMEOUT (0xB4)

域	位	读写	复位值	描述
IC_SMBDAT_STUCK_TIMEOUT	31:0	RW	0xFFFFFFFF	数据扩展超时参数设置寄存器

5.23.5.10 IC_SMBCLK_LOW_SEXT (0xB8)

域	位	读写	复位值	描述
IC_SMBCLK_LOW_SEXT	31:0	RW	0xFFFFFFFF	SEXT 参数设置寄存器

5.23.5.11 CLR_SMMST_SCL_EXT_LOW_TIMEOUT (0xBC)

域	位	读写	复位值	描述
reserved	31:1	R0	0x0	保留
clr_smmst_scl_ext_low_timeout	0	R0	0x0	读这个寄存器来清除 IC_RAW_INTR_STAT 状态寄存器的 smmst_scl_ext_low_timeout 中断(bit12)。

5.23.5.12 CLR_SMMST_SCL_TMO_LOW_TIMEOUT (0xC0)

域	位	读写	复位值	描述
reserved	31:1	R0	0x0	保留
clr_smmst_scl_tmo_low_timeout	0	R0	0x0	读这个寄存器来清除 IC_RAW_INTR_STAT 状态寄存器的 smmst_scl_tmo_low_timeout 中断(bit 13)。

5.23.5.13 CLR_SMMST_SDA_LOW_TIMEOUT (0xC4)

域	位	读写	复位值	描述
reserved	31:1	R0	0x0	保留
clr_smmst_sda_low_timeout	0	R0	0x0	读这个寄存器来清除 IC_RAW_INTR_STAT 状态寄存器的 smmst_sda_low_timeout 中断(bit 14)。

5.23.5.14 CLR_SMSLV_SCL_EXT_LOW_TIMEOUT (0xC8)

域	位	读写	复位值	描述
reserved	31:1	R0	0x0	保留
clr_smslv_scl_ext_low_timeout	0	R0	0x0	读这个寄存器来清除 IC_RAW_INTR_STAT 状态寄存器的 smslv_scl_ext_low_timeout 中断 (bit 15)。

5.23.5.15 CLR_SMSLV_SCL_TMO_LOW_TIMEOUT (0xCC)

域	位	读写	复位值	描述
reserved	31:1	R0	0x0	保留
clr_smslv_scl_tmo_low_timeout	0	R0	0x0	读这个寄存器来清除 IC_RAW_INTR_STAT 状态寄存器的 smslv_scl_tmo_low_timeout 中断 (bit 16)。

5.23.5.16 CLR_SMBALERT_IN_N (0xD0)

域	位	读写	复位值	描述
reserved	31:1	R0	0x0	保留
clr_smbalert_in_n	0	R0	0x0	读这个寄存器来清除 IC_RAW_INTR_STAT 状态寄存器的 smbalert_in_n 中断 (bit 17)。

5.24 PWM 控制器

PWM 是一种对模拟信号电平进行数字编码的方法。飞腾派集成的 PWM 控制器支持典型的 PWM 功能，有 2 个完全独立的 compare 输出通道。

5.24.1 操作说明

使用 PWM 功能前，需要先配置相关 PAD 复用寄存器，将对应 PAD 配置到对应功能上，即可使用 PWM 功能。飞腾派 PWM 不支持 capture 模式。

5.24.1.1 compare 输出模式说明

在此模式下，Timer 启动后，将 PWM_CCR 寄存器的值与 counter 值不断进行比较，当二者匹配时，相关联的 IO 管脚输出 0、1 或翻转（由对应的 PWM_CTRL.CMP 决定），输出信号是 glitch-free 的（寄存后输出）。

注意，在 compare 模式下，输出 Duty 的值，可以直接来自 PWM_CCR 寄存器，也可以来自 FIFO，通过 FIFO 可实现对 duty 值的动态控制，减少总线频繁操作寄存器。当 FIFO 空的时候，pwm_cmp_out 信号输出保持原值，并置位状态寄存器相关位，输出中断（使能开关打开）。计数器正常计数，当 FIFO 不空了，会在下一轮的计数开始后，加载

最新的 Duty 值。最终选择 FIFO 还是 PWM_CCR 寄存器，是通过 PWM_CTRL 寄存器配置的。

5.24.1.2 中断

PWM 有两种中断源，对外只有一根中断线，具体中断置位的情况如下所述，但是否将中断输出，要参考全局中断控制寄存器相关位是否使能：

- 计数中断，会在不同模式下产生中断：
 - Modulo 模式时，当计数值达到 period 的时候，产生中断，若配置的 period 的值始终为 0，则不产生中断；
 - Up and down 模式时，当计数值递减为 0 的时候，产生中断，但最开始计数值为 0 不产生中断
- 配置为 compare 模式时，有不同中断触发情况：
 - compare 模式时，当 Timer 计数器等于 Duty 值的时候，产生一个中断；
 - 若采用 FIFO 存放 Duty 的值，则在每次计数刚开始的时候，检测到此时 FIFO 空，中断不会产生；即使在计数过程中，FIFO 变为空，也要等下一个计数周期开始产生中断；
 - 当寄存器配置的 Period 值为 0 的时候，不会产生中断。
- FIFO 空中断，该中断只有在使用 FIFO 作为存放 DUTY 时候才可能触发。

STATE 寄存器包含有各 channel 的上述中断标志，任意一个使能的中断标志为 1 时，输出到全芯片中断部件的中断线置位。

5.24.1.3 死区功能配置

在上电复位完成后，进行寄存器配置，从而启动 dead_band 功能：

1. APB 写 DBDLY 寄存器，配置上升沿&下降沿延迟周期；
2. APB 写 DBCTRL 寄存器，设定 PWM 波输入源或 bypass 死区模块，是否上升沿延迟，下降沿延迟或两者同时应用于输入信号、上升沿延迟信号或/和下降沿延迟信号在输出后是否反转。

5.24.1.4 内部 pwm 子模块启动说明

在上电复位完成后，进行寄存器配置，从而启动 Timer 和 PWM 功能：

1. APB 写 TIM_CTRL 寄存器，配置分频系数（PWM 控制器主时钟为 50MHz），计数模式，全局中断使能开关，但是 TIM_CTRL.ENABLE 保持 0，即不使能 PWM。
2. APB 写 PWM_PERIOD 寄存器，设定计数周期。
3. APB 写 PWM_CTRL 寄存器，设定 PWM 工作模式(compare)，并根据工作模式，设

定相关模式工作参数。

4. 选择 compare 模式，则 APB 写 PWM_CCR 寄存器中写入 Duty 值(或向 FIFO 中)，配置输出初始值（默认为 0）。
5. 配置中断使能位。

APB 写 TIM_CTRL. ENABLE 值 1，该 PWM 可以工作。若要实现多 PWM 同时工作，可等多 PWM 配置完成后，上层同时将 pwm_gectl 信号拉高（通过配置 0x2807_e020 寄存器，该寄存器低 8 位控制使能 8 个 PWM 控制器，其中 bit0 使能 PWM0 控制器，bit1 使能 PWM1 控制器，以此类推），此时，PWM 开始工作。

5.24.1.5 内部 pwm 子模块暂停说明

工作中，若想暂停 PWM，则直接写 TIM_CTRL. ENABLE 值 0，暂停整个 PWM。暂停的时候，会有如下结果：

1. 计数器清零，输出分频时钟变为 pclk(控制器时钟为 50MHz)，即没有分频。
2. PWM 功能停止，若是 compare 模式，则输出保持默认值。

5.24.1.6 内部 pwm 子模块模式切换

1. 工作中，若想改变时钟分频系数，硬件支持在 PWM 处于 enable 下进行更改，不会产生毛刺；
2. 工作中，若想改变 counter 计数模式，必须在 PWM 处于 disable 下进行更改，否则会出现不确定情况；
3. 在工作中，若需要改变 period 或通道的 duty，硬件支持在 enable 下进行更改，但新的值只有在 counter=0 的时候才重新加载；
4. 在 run 过程中，若想改变 PWM 初始值，是没有作用的，只有在 disable 状态下进行修改。

5.24.1.7 PWM DUTY 的修改

在 compare 模式下，pwm 输出 duty 控制的数据，可来自 PWM_CCR 寄存器(default)，也可以通过模式配置，来自 FIFO(动态变化)。切换模式的时候，必须在 PWM 处于 disable 状态下进行。

● FIFO 模式

1. 在配置 duty 之前，要先确定使用 FIFO 模式。
2. APB 开始向 PWM_CCR 寄存器写数据，因为是 FIFO 模式，数据会同时写入到 FIFO 中，PWM_CCR 寄存器只保留当前最新的写入值。

3. 当 APB 读取到 STAT[3] 为 1，表示 FIFO 满了，要停止向 FIFO 中写入数据，若 APB 仍然向 PWM_CCR 寄存器写数据，内部逻辑控制，此时 pready 信号为 0，使该次 APB 写操作暂时不能完成。会存在一定风险，即 APB 总线会被占据，此时，若写其它寄存器是可以的。
4. 当 FIFO 为空的时候，产生一个中断，此时 PWM 输出保持，计数器正常计数，计数过程中，若 FIFO 非空了，会在下一个计数循环开始新的输出。

注意，在使用 FIFO 模式的时候：

- 上电复位后，enable 之前，FIFO 中必须有数据，否则 enable 后，立刻出中断；
- 在工作中，FIFO 可以出现空，pwm 输出保持；
- 工作中若 disable PWM，再次 enable 之前，FIFO 中必须包含数据；

● 寄存器模式

在使用寄存器模式的时候，Duty 的值可以随时变化，此时，APB 写操作时候，不再向 FIFO 中存放数据，当 pwm 计数器为 0 的时候，从寄存器中会加载新的 duty 值，此时要考虑 duty 为 0 的边界情况。注意：配置寄存器时，不允许软件将配置的 duty 值大于 period 值。

5.24.2 寄存器列表

每个 PWM 控制器有 4K 地址空间，划分为 3 块。以 PWM0 控制器为例：

0~1k: pwm_db;

1~2k: pwm0;

2~3k: pwm1;

PWM apb 接口宽度为 12bit，地址高两位为 2'b00 时访问 pwm_db 寄存器空间，地址高两位为 2'b01 时访问 pwm0 寄存器空间，地址高两位为 2'b10 时访问 pwm1 寄存器空间。

表 5-67 PWM 寄存器基地址

名称	基地址
PWM0	0x000_2804_A000
PWM1	0x000_2804_B000
PWM2	0x000_2804_C000
PWM3	0x000_2804_D000
PWM4	0x000_2804_E000
PWM5	0x000_2804_F000
PWM6	0x000_2805_0000
PWM7	0x000_2805_1000

注意：每个 PWM 控制器对应两路 pwm，因此外接 pin 脚共 16 个 pwm。二者对应关系为控制器 PWM0 分别与外接 pin 脚的 pwm0 和 pwm1 相接，控制器 PWM1 分别与外接 pin 脚的 pwm2 和 pwm3 相接，以此类

推。

表 5-68 PWM 寄存器列表

寄存器	偏移	描述
DBCTRL	0x00	计数相关控制寄存器
DBDLY	0x04	pwm 输出 period 寄存器
TIM_CNT	0x400/0x800	计数器，用来表示当前内部 timer 计数信息，计数采用内部分频后的时钟
TIM_CTRL	0x404/0x804	计数相关控制寄存器
STATE	0x408/0x808	pwm 内部状态寄存器
PWM_PERIOD	0x40C/0x80C	pwm 输出 period 寄存器
PWM_CTRL	0x410/0x810	pwm 工作模式控制寄存器
PWM_CCR	0x414/0x814	通道的捕获和比较寄存器

5.24.3 寄存器说明

5.24.3.1 DBCTRL (0x00)

域	位	读写	复位值	描述
reserved	31:6	RW	0x0	保留
OUT_MODE	5:4	RW	0x0	死带输出模式控制： 00: bypass 01: 禁用上升沿延迟，pwm0_out 直接输出，下降沿延迟作用在 pwm1_out 10: 禁用下降沿延迟，pwm1_out 直接输出，上升沿延迟作用在 pwm0_out 11: pwm0_out 上升沿和 pwm1_out 下降沿都完全启用死带
POLSEL	3:2	RW	0x0	极性选择控制： 00: active high(AH) pwm0_out 和 pwm1_out 都不翻转 01: active low complementary(ALC) pwm0_out 翻转 10: active high complementary(AHC) pwm1_out 翻转 11: active low(AL) pwm0_out 和 pwm1_out 都翻转
IN_MODE	1	RW	0x0	死区输入 pwm 源选择： 0: pwm0 输入 1: pwm1 输入
DB_SW_RST	0	RW	0x0	全局软复位信号，软件写 1，部件实现全局软复位，复位完成后，自动跳出软复位。 软件读取到 1 的时候，表示处于复位中 软件读取到 0 的时候，表示跳出复位，可以进行功能操作。

5.24.3.2 DBDLY (0x04)

域	位	读写	复位值	描述
reserved	31:20	RW	0x0	保留
DBFED	19:10	RW	0x0	下降沿延迟周期

域	位	读写	复位值	描述
DBRED	9:0	RW	0x0	上升沿延迟周期

5.24.3.3 tim_cnt (0x400/0x800)

域	位	读写	复位值	描述
reserved	31:16	RW	0x0	保留
cnt	15:0	RW	0x0	计数标志

5.24.3.4 tim_ctrl (0x404/0x804)

域	位	读写	复位值	描述
reversed	31:28	RW	0x0	保留
DIV	27:16	RW	0x0	分频参数 0: 1 分频 1: 2 分频 ... 支持分频范围: 1~4096
reversed	15:6	RW	0x0	保留
GIE	5	RW	0x0	全局中断输出使能控制位
OVFIF_ENABLE	4	RW	0x0	counter-overflow 中断使能控制位
reversed	3	RW	0x0	保留
Mode	2	RW	0x0	0: modulo, 在 0~pwm_period.ccr 之间循环计数; 1: up-and-down, 在 0~pwm_period.ccr-1 之间循环计数。 工作中, 计数模式的修改, 需要先 disable 全局, 修改后再 enable。
ENABLE	1	RW	0x0	全局使能位。 1 表示全局使能, 0 表示不使能。 0: disable, 计数器清零, compare 停止工作, 输出复位值, 寄存器内容保持原值; 1: enable, Timer&PWM 正常工作 若需要暂停 Timer/PWM 工作, 可通过该位控制。
SW_RST	0	RW	0x0	全局软复位信号, 软件写 1, 部件实现全局软复位, 复位完成后, 自动跳出软复位。 软件读取到 1 的时候, 表示处于复位中 软件读取到 0 的时候, 表示跳出复位, 可以进行功能操作。

5.24.3.5 state (0x408/0x808)

域	位	读写	复位值	描述
reserved	31:4	RW	0x0	保留
FIFO_FULL	3	RW	0x0	该位显示当前 FIFO 满的情况, 但是不作为中断输出, 在软件向 FIFO 中写数据的时候作为前提, 若检测到该位是 1, 则此时不允许写数据, 否则 APB 总线可能会被堵塞, 但是 FIFO full 不会输出中断。

域	位	读写	复位值	描述
FIFO_EMPTY	2	RW1c	0x0	只有在选择使用 FIFO 作为 duty 控制，且 FIFO 为空的时候，产生中断。
OVFIF	1	RW1c	0x0	计数中断，不同计数模式下有不同定义。 modulo: 当计数值达到 PWM_PERIOD 寄存器值时，产生中断。 up and down: 当 PWM_PERIOD 非 0，此时，若计数值递减为 0 的时候，产生中断（但整体处于复位时候不产生），若 PWM_PERIOD 为 0，无中断。 注意：计数是在分频后的时钟域，产生的中断脉冲信号相对于 clk_fio 会很宽，为了避免重复对状态寄存器赋值，使用上升沿检测来置位。
CHIF	0	RW1c	0x0	compare 中断标志，写 1 清除。

5.24.3.6 pwm_period (0x40C/0x80C)

域	位	读写	复位值	描述
reversed	31:16	RW	0x0	保留
CCR	15:0	RW	0x1	计数控制功能，在非 Free-running 模式时，该值是用来控制 counter 归零等情况的，即 period 寄存器；该寄存器的值支持工作中动态调整（compare 模式下，实际输出 period 周期数为此值+1）。

5.24.3.7 pwm_ctrl (0x410/0x810)

域	位	读写	复位值	描述
reserved	31:10	RW	0x0	保留
FIFO_EMPTY_ENABLE	9	RW	0x0	在 compare 模式下，FIFO 出现空的时候，中断输出使能控制位，1 表示使能，0 表示非使能，即使非使能，FIFO 空的时候也要置位状态寄存器相关位，但不输出中断。
DUTY_SEL	8	RW	0x0	compare 模式下，控制 duty 的比较值的来源。 0: 来自寄存器 PWM_CCR; 1: 来自 FIFO, fifo 值由 PWM_CCR.ccr 写入。 duty 值来自寄存器的时候，只在上一个 pwm 周期结束的时候 load 当前 pwm_ccr 寄存器的值，比如说在一个 pwm 周期内多次写这个寄存器，那么只有最后一个值有效；来自 FIFO 时，当 FIFO 非满时，这些值会存储在 FIFO 内，硬件在每个 pwm 周期结束时自动读出做为下一次的 duty 值。
ICOV	7	RW	0x0	在 compare 模式下，配置 PWM 输出的初始值，跳出全局复位后，初始值根据当前设定，在 enable 信号使能之前，需先配置好该值。
CMP	6:4	RW	0x0	比较操作配置。 000: 比较匹配时输出置 1。 001: 比较匹配时输出清 0。 010: 比较匹配时输出翻转。 011: 向上计数且比较匹配时输出置 1。然后在 up-and-down

域	位	读写	复位值	描述
				<p>模式向下计数且比较匹配时输出清 0，其他模式则在计数到 0 时输出清 0。</p> <p>100：向上计数且比较匹配时输出清 0。然后在 up-and-down 模式向下计数且比较匹配时输出置 1，其他模式则在计数到 0 时输出置 1。</p> <p>101：计数值等于 PWM_CCR.ccr 时输出清 0，等于 pwm_period.ccr 时输出置 1。</p> <p>110：计数值等于 PWM_CCR.ccr 时输出置 1，等于 pwm_period.ccr 时输出清 0。</p> <p>111：输出管脚初始化。</p> <p>说明：对于“010：比较匹配时输出翻转”，比较匹配是指 counter 值变化后匹配；如果初始值相同，不算比较匹配上。对于“000：比较匹配时输出置 1”“001：比较匹配时输出清 0”“101”“110”不涉及一直翻转的模式，比较匹配是指 counter 值初次匹配。</p> <p>对于“011”“100”比较匹配也指 counter 值初次匹配，但是否跳变，还要看当前计数器的计数方向。</p>
IE	3	RW	0x0	<p>中断使能位，用来控制是否产生 compare 中断。</p> <p>该位为 0 表示 compare 中断使能无效，1 表示 compare 中断使能有效。</p>
Mode	2	RW	0x0	1：比较模式
reserved	1:0	RW	0x0	保留

5.24.3.8 pwm_ccr (0x414/0x814)

域	位	读写	复位值	描述
reversed	31:17	RW	0x0	保留
GPIO	16	RW	0x0	输出 GPIO 控制位
CCR	15:0	RW	0x0	compare 模式：该寄存器的值属于控制 duty cycle 的，此时，该值最小为 1，最大不能超过 pwm_period.ccr 的值。

5.25 Timer

5.25.1 简介

飞腾派支持典型的 Timer 和 Fan Tachometer 功能。有一个独立的输入捕获(capture)通道，主要支持特性如下：

- 支持全局软复位功能，全局使能控制；
- 支持 restart 或 free_run 计数模式；
- 支持可配置计数器 32 位或 64 位；
- 支持上升、下降边沿输入 capture 检测；

- 支持可配转速周期内电机转速检测，并故障检查；
- 支持可配置重复定时或一次定时，计数器开始计数值可配；
- 支持一次定时并可以动态修改定时参数；
- 支持计数器捕获、比较或翻转事件触发中断。

5.25.2 操作说明

5.25.2.1 启动及定时功能说明

在上电复位完成后，进行寄存器配置，从而启动 Timer 或 Tachometer 功能：

1. APB 写 `intr_mask_n` 寄存器使能相关的中断；
2. APB 写 `comp_value_u` 与 `comp_value_l` 寄存器设置定时计数值；
3. APB 写 `ctrl_reg` 寄存器，配置计数模式，时钟使能，计数器使能或 tach 输入边沿检测模式等配置信息；
4. 要实现一次定时功能，除了使能相关的中断位，软件需要先读 `cnt_value_u` 与 `cnt_value_l` 寄存器的值，将这个值加上需要计数的时间后，将新值写入 `comp_value_u` 与 `comp_value_l` 寄存器中。要注意不同计数模式下计数器的翻转和重置。

另外，如果在一次定时周期内，软件修改了 `comp_value_u` 与 `comp_value_l` 寄存器的值，如果修改后的值比之前大，控制器会分别定时到两次然后给出两个一次定时中断，如果想第一次写入值无效，软件写 `ctrl_reg` 寄存器 `force_load` 位为 1。如果修改后的值比之前小，软件必须写 `ctrl_reg` 寄存器 `force_load` 位为 1，那么定时器将会在计数到第二次写入的数值时给出一次定时中断，第一次写入的值无效；如果依旧想要在第一次计数周期报出中断，软件需要重新写 `comp_value_u` 与 `comp_value_l` 寄存器。

5. Tachometer 功能使能后，软件读取 `tach_result_reg` 寄存器获取转速计数，`bit[31]` 为 1 表示 `tach_value` 有效。该值在 tach 使能后每计算一次立即更新。

5.25.2.2 中断说明

定时器共有七种中断，全部中断都可以通过配置寄存器选择使能或屏蔽：

- 输入捕获中断 (`tach_cap`)：tach_in 信号检测到上升沿/下降沿，边沿选择由 `ctrl_reg` 寄存器的 `tach_mode` 域确定，在计数到设定 `cap_cnt` 值后产生中断。
- 重复定时中断 (`cyc_comp`)：每隔一定时间产生一个定时中断；
- 一次定时中断 (`once_comp`)：软件可以随时写 `comp_value_u` 与 `comp_value_l` 寄存器，在计数达到 `comp_value_u` 与 `comp_value_l` 寄存器的值后产生一次定

- 时中断；
- 翻转中断（rollover）：计数器在 free_run 模式下，从满刻度翻转到 0x0 时产生一次翻转中断；
 - 超转速中断（tach_over）：检测到 tach_value 值超出寄存器设定的 tach_over_limit 值后，产生一个超转速中断；
 - 低转速中断（tach_under）：检测到 tach_value 值低于寄存器设定的 tach_under_limit 值后，产生一个低转速中断。

5.25.2.3 复位说明

若在工作中，需要重新配置所有寄存器，则首先进行全局软复位操作，软件 SWR 写 1，部件实现全局软复位，复位完成后，自动跳出软复位。软件读取到 0 时，表示跳出软复位；复位后，所有寄存器信息为初始态。

软件开始重新配置寄存器信息和使能位重新开始工作。

5.25.2.4 计数器暂停说明

工作中若想暂停计数器，可以直接写 ctrl_reg.counter_en 位为 0，计时器暂停计数，计数值被冻结。当软件写 ctrl_reg.counter_en 位为 1 重新使能计数器时，计数器将从冻结值继续开始计数。若想重新使能后计数器从 0 开始重新计数，需要在使能前软件写 ctrl_reg.counter_clr 位为 1。

5.25.2.5 模式切换说明

进行计数模式切换时，软件必须先配置寄存器 disable 计数器并将计数器清零，即 ctrl_reg 寄存器 counter_en 置 0 和 counter_clr 置 1，然后在进行模式切换。

5.25.2.6 timer 功能寄存器配置序列

1. APB 写 ctrl_reg 寄存器，配置 mode 位为 00，选择定时器功能；然后是计数模式、计数器 32/64 位选择、定时模式选择、时钟使能配置。
2. APB 写 intr_mask_n 寄存器使能对应的中断；
3. APB 写寄存器设置 comp_value_u 与 comp_value_l 定时计数值。注意，在选择 64 位计时器时，读写值时必须先读写低 32 位寄存器 comp_value_l，再读写高 32 位 comp_value_u 寄存器。
4. APB 写寄存器 ctrl_reg，使能计数器。

注意：控制器支持计数开始值可配，可以通过 APB 写 start_value_reg 寄存器改变

初始计数值，这个值将在下一个计数周期有效。如果配置了 `ctrl_reg.force_load` 值为 1，那么计数器将立即从 `start_value_reg` 开始计数。

使用一次定时，计数器模式应该设置为 `free_run`。

5.25.2.7 tacho 功能寄存器配置序列

1. APB 写 `ctrl_reg` 寄存器，配置 `mode` 位为 01，选择 tachometer 功能；然后是计数模式、计数器 32/64 位选择、tach 输入模式选择、tach 计数周期、时钟使能配置、`tach_in` 消抖级数；
2. APB 写寄存器 `ctrl_reg`，使能计数器；
3. APB 写 `intr_mask_n` 寄存器使能对应的中断；
4. APB 写 `tach_under_reg` 和 `tach_over_reg` 值，设置合理转速范围；
5. APB 写寄存器 `ctrl_reg`，使能 `tach_in` 输入；
6. APB 读 `tach_result_reg` 寄存器，`bit[31]` 为 1 表示此时 `bit[30:0]` 有效，表示此时在设置的转速周期内的时钟计数。

5.25.2.8 输入 capture 功能寄存器配置序列

1. APB 写 `ctrl_reg` 寄存器，配置 `mode` 位为 10，选择 capture 功能；然后、输入模式选择、时钟使能配置、`tach_in` 消抖级数，`cap_cnt` 等配置；
2. APB 写 `intr_mask_n` 寄存器使能对应的中断；
3. APB 写寄存器 `ctrl_reg`，使能 `tach_in` 输入。

5.25.3 寄存器列表

表 5-69 Timer 寄存器基地址

名称	基地址
Tacho0	0x000_2805_4000
Tacho1	0x000_2805_5000
Tacho2	0x000_2805_6000
...	...
Tacho37	0x000_2807_9000

注意：控制器 Tacho0~Tacho15 包含 timer 控制器所有功能，Tacho16~Tacho37 仅有 timer 功能（无 tacho 能和输入 capture 功能）。

表 5-70 Timer 寄存器列表

寄存器	偏移	描述
<code>ctrl_reg</code>	0x00	Tachometer 控制寄存器
<code>tach_result_reg</code>	0x04	一个转速周期内的时钟周期计数结果
<code>comp_value_u</code>	0x08	定时计数值高 32 位

寄存器	偏移	描述
comp_value_l	0x1C	timer 模式下：定时计数值低 32 位 tacho 模式下：配置 tach_value 在几个转速周期= pulse_num
cnt_value_u	0x20	计数器当前计数值高 32 位
cnt_value_l	0x24	计数器当前计数值低 32 位
intr_mask_n	0x28	中断使能寄存器
intr_status	0x2C	中断状态寄存器
tacho_over_reg	0x30	超转速中断寄存器
tacho_under_reg	0x34	低转速中断寄存器
start_value_reg	0x38	计数器初始值

5.25.4 寄存器说明

5.25.4.1 ctrl_reg (0x00)

域	位	读写	复位值	描述
tacho_en	31	RW	0x0	tachometer 输入使能位
reserved	30:28	RW	0x0	保留
tim_mode	27	RW	0x0	定时模式选择位 0: 重复定时 1: 一次定时
counter_clr	26	W1-1	0x0	清空计数器 1: 清空 0: 不清空
counter_en	25	RW	0x0	计数器使能位
counter_series	24	RW	0x0	计数器 32/64 位选择 0: 32 位 1: 64 位
reserved	23	RW	0x0	保留
tim_counter_mode	22	RW	0x0	计数器计数模式: 1: restart mode 0: free_run mode
tach_mode	21:20	RW	0x0	tach 输入模式选择位 00: 下降沿 01: 上升沿 10: 下降沿&上升沿 11: 保留
anti_jitter_number	19:18	RW	0x0	tach_in 输入消抖寄存器级数= N+1
reserved	17:12	RW	0x0	保留
cap_cnt	11:5	RW	0x7F	在输入 capture 模式下，边沿检测计数到该值的时候，会产生一个中断，并重新计数
cap_en	4	RW	0x0	输入 capture 计数使能
force_load	3	W1-1	0x0	强制更新位 1: 立即更新

域	位	读写	复位值	描述
tim_swr	2	RW	0x0	控制器复位信号 1: 在复位状态 0: 不在复位状态
mode	1:0	RW	0x0	模式选择 00: 定时器功能 01: tachometer 功能 10: 输入 capture 功能 11: 保留

5.25.4.2 tach_result_reg (0x04)

域	位	读写	复位值	描述
tachvalue_vld	31	R0	0x0	tach_result_reg 值有效
tach_value	30: 0	R0	0x0	一个转速周期内的时钟周期计数结果

5.25.4.3 comp_value_u (0x08)

域	位	读写	复位值	描述
cmp_value_u	31:0	RW	0x0	定时计数值高 32 位

5.25.4.4 comp_value_l (0x010)

域	位	读写	复位值	描述
cmp_value_l	31:0	RW	0x0	timer 模式下: 定时计数值低 32 位 tacho 模式下: 配置 tach_value 在几个转速周期 = pulse_num

5.25.4.5 cnt_value_u (0x20)

域	位	读写	复位值	描述
cnt_value_u_in	31:0	R0	0x0	计数器当前计数值高 32 位

5.25.4.6 cnt_value_l (0x24)

域	位	读写	复位值	描述
cnt_value_l_in	31:0	R0	0x0	计数器当前计数值低 32 位。cap mode: 边沿检测计数值

5.25.4.7 intr_mask_n (0x28)

域	位	读写	复位值	描述
reserved	31:6	RW	0x0	保留
tach_cap_en	5	RW	0x0	tach 输入捕获中断使能

域	位	读写	复位值	描述
cyc_comp_en	4	RW	0x0	重复定时输出中断使能
once_comp_en	3	RW	0x0	一次定时输出中断使能
rollover_en	2	RW	0x0	计数器翻转中断使能
tach_under_en	1	RW	0x0	tach 低于转速中断使能
tach_over_en	0	RW	0x0	tach 超转速中断使能

5.25.4.8 intr_status (0x2C)

域	位	读写	复位值	描述
reserved	31:6	W1C	0x0	保留
tach_cap_intr_in	5	W1C	0x0	tach 输入捕获中断
cyc_intr_in	4	W1C	0x0	重复定时输出中断
once_intr_in	3	W1C	0x0	一次定时输出中断
rollover_intr_in	2	W1C	0x0	计数器翻转中断
tachunder_intr_in	1	W1C	0x0	tach 低于转速中断
tachover_intr_in	0	W1C	0x0	tach 超转速中断

5.25.4.9 tacho_over_reg (0x30)

域	位	读写	复位值	描述
reserved	31	RW	0x0	保留
tach_over_limit	30:0	RW	0x7FFF_FFFF	预设的 tach 最大值, 超出此值时出 tach_over 出中断。

5.25.4.10 tacho_under_reg (0x34)

域	位	读写	复位值	描述
reserved	31	RW	0x0	保留
tach_under_limit	30:0	RW	0xf	预设的 tach 最小值, 低于此值时出 tach_under 出中断。

5.25.4.11 start_value_reg (0x38)

域	位	读写	复位值	描述
start_value	31:0	RW	0x0	计数器初始值。如果 force_load=1, 立即从初始值开始重新计数, 否则下一个计数周期有效。

5.26 矩阵键盘 (Keypad)

Keypad 是一组排列在模块或垫上的按钮, 上面有数字、符号或字母。键盘主要用于需要数字输入的设备。许多设备的配置都遵循 E. 161 标准。

5.26.1 操作说明

5.26.1.1 正常工作流程

- 典型键盘配置及扫描顺序

通过以下步骤配置键盘：

1. 启用键盘的行数 (KPP_KPCR[KRE])。
2. 配置列为输出 0。
3. 清除 KPKD 状态标志和同步器链。
4. 设置 KDIE (按键按下中断使能) 控制位, 清除 KRIE (按键释放中断使能) 控制位 (避免错误释放事件)。
5. (系统现在处于待机状态, 等待按下一个键。)

- 键盘扫描程序的操作步骤

1. 禁用 (按下和释放) 键盘中断。
2. 写 0 到 KPP_KDDR [KCD], 设置列的方向为输入, 此时列的输出由 pad 得上拉电阻拉高。
3. 将第 0 列配置成输出, 并输出 0, 其他列为输入 (上拉到 1)。
4. 采样每行的输入并保存数据。在单个列上可以检测到多个键的按下。
5. 将下一列配置成输出, 并输出 0, 其他列为输入 (上拉到 1)。
6. 采样每行的输入并保存数据。在单个列上可以检测到多个键的按下。
7. 对其余的列重复步骤 5-6。
8. 在准备待机模式时, 将所有列输出写为 0。
9. 通过写入 1 来清除 KPKD 和 KPKR 状态位; 通过向 KPP_KRSS 寄存器写入一个 “1” 来置位 KPKR 同步器链; 通过写一个 “1” 到 KDSC 寄存器来清除 KPKD 同步器链。
10. 重新启用适当的键盘中断, 以便 KDIE 检测到一个键保持条件, 或者 KRIE 检测到一个键释放事件。

5.26.1.2 keypad 使用注意事项

- 矩阵按键的鬼键问题

在使用带有双触点开关的简单键盘矩阵的情况下, 当按下三个或更多键时, 就有可能检测到 “幽灵” 键。这是这种键盘矩阵所施加的限制。

同时按下三个键会导致当前被软件 “扫描” 的一列与另一列之间发生短路。根据第三个键按下的位置, 可能会检测到 “幽灵” 键按下。但这可以通过使用提供 “幽灵” 键

保护的键盘矩阵来纠正。这样的矩阵在行和列之间的所有键盘点上加入了单向“二极管”。这样，多次按下三个键不会导致第四个键短路。

- 矩阵按键的多按键检测问题

1. 无法在有按键未释放时，再产生到另一个按键动作的中断信号。所以当有按键未被释放时，需要软件定时来进行扫描。
2. 释放中断只能在所有按键都释放的情况下产生。

5.26.2 寄存器列表

表 5-71 Keypad 寄存器基地址

名称	基地址
Keypad	0x000_2807_A000

表 5-72 Keypad 寄存器列表

寄存器	偏移	描述
Keypad Control Register (KPCR)	0x0000	控制寄存器
Keypad Status Register (KPSR)	0x0004	状态寄存器
Keypad Data Direction Register (KDDR)	0x0008	数据方向寄存器
Keypad Data Register (KPDR)	0x000C	数据寄存器

5.26.3 寄存器说明

5.26.3.1 KPCR (0x000)

域	位	读写	复位值	描述
reserved	31:24	RO	0x0	保留
KCO	23:16	RW	0x0	列有效的使能信号，KCO[23:16]对应 col[7:0] 1：使能 0：不使能
reserved	15:8	RO	0x0	保留
KRE	7:0	RW	0x0	行有效的使能信号，KRE[7:0]对应 row[7:0] 1：使能 0：不使能

5.26.3.2 KPSR (0x004)

域	位	读写	复位值	描述
reserved	31:10	RO	0x0	保留
KRIE	9	RW	0x0	按键释放中断使能信号 1：使能 0：不使能
KDIE	8	RW	0x0	按键按下中断使能信号 1：使能 0：不使能
reserved	7:4	RO	0x0	保留
KRSS	3	W1-1	0x0	释放同步器清 0 信号

域	位	读写	复位值	描述
				写 1 有效，保存一拍自动清 0
KDSC	2	W1-1	0x0	按下同步器清 0 信号 写 1 有效，保存一拍自动清 0
KPKR	1	W1C	0x0	按键释放检测信号 硬件检测信号，写 1 清 0
KPKD	0	W1C	0x0	按键按下检测信号 硬件检测信号，写 1 清 0

5.26.3.3 KDDR (0x008)

域	位	读写	复位值	描述
reserved	31:24	RO	0x0	保留
KCDD	23:16	RW	0x0	列输出方式信号，KCDD[23:16]对应 col[7:0] 0：开漏输出 1：推挽输出
reserved	15:0	RO	0x0	保留

5.26.3.4 KPDR (0x00c)

域	位	读写	复位值	描述
reserved	31:24	RO	0x0	保留
KCD	23:16	RW	0x0	列输出状态，配合 KCDD 寄存器使用，当 KCDD 相应位为高时输出有效。 KCD[23:16]对应 col[7:0]。 0：输出低电平 1：输出高电平
reserved	15:8	RO	0x0	保留
KRD	7:0	RW	0x0	行输入状态，KRD[7:0]对应 row[7:0] 寄存器值随着硬件输入信号而改变。 0：输入低电平 1：输入高电平

5.27 GPIO 控制器

5.27.1 简介

飞腾派集成 6 个 GPIO 控制器提供 96 位 GPIO 信号。96 位 GPIO 信号，支持外部中断功能。每位中断信号（默认没有优先级区分）产生一个中断报送到全芯片的中断管理模块。在中断管理模块内可针对 GPIO0~5 六路中断设置不同的优先级。支持中断单独屏蔽和清除。控制器 GPIO0~2 的每位中断单独上报。控制器 GPIO3~5 的中断由模块内合成一个中断上报（共 3 个中断，每个控制器 16 个引脚合并产生一个中断）。

飞腾派各 GPIO 控制器主时钟为 50MHz。

5.27.2 操作说明

首先将对应 pin 脚的 PAD 复用类型设置为 GPIO。

5.27.2.1 GPIO 输出功能

1. 配置端口方向控制寄存器 GPIO_SWPORT_DDR 方向为输出。例如配置 GPIO_SWPORT_DDR 寄存器为 16'hFFFF，那么 16 位 GPIO 信号为输出模式；
2. 配置端口输出寄存器 GPIO_SWPORT_DR。写入该寄存器的值在端口的 I/O 信号线上输出。读取的值等于写入该寄存器的最后一个值。

5.27.2.2 GPIO 输入功能

1. 配置端口方向控制寄存器 GPIO_SWPORT_DDR 方向输入。例如配置 GPIO_SWPORT_DDR 寄存器为 16'h0000，那么 16 位 GPIO 信号为输入模式；
2. 通过读取 GPIO_EXT_PORT 寄存器，读取出 GPIO 信号的值。

5.27.2.3 GPIO 中断功能

1. 配置方向控制寄存器方向为输入。例如配置 GPIO_SWPORT_DDR 为 16'h0000，那么 16 位 GPIO 信号为输入模式；
2. 配置寄存器 GPIO_INTEN 写 1 使能中断功能；
3. 配置中断类型寄存器 GPIO_INTTYPE_LEVEL 设置端口中断敏感类型，每当 0 写入此寄存器的某位时，它将中断配置为电平敏感型；否则，它将是边沿敏感型；
4. 配置中断极性寄存器 GPIO_INT_POLARITY 设置端口输入端可能出现的边沿或电平极性。每当 0 写入此寄存器的某个位时，它将中断类型配置为下降沿或低电平敏感；否则，它将是上升沿或高电平敏感。

5.27.3 寄存器列表

表 5-73 GPIO 寄存器基地址

名称	基地址
GPIO0	0x000_2803_4000
GPIO1	0x000_2803_5000
GPIO2	0x000_2803_6000
GPIO3	0x000_2803_7000
GPIO4	0x000_2803_8000
GPIO5	0x000_2803_9000

表 5-74 GPIO 寄存器列表

寄存器	偏移	描述
GPIO_SWPORT_DR	0x00	端口输出寄存器
GPIO_SWPORT_DDR	0x04	端口方向控制寄存器
GPIO_EXT_PORT	0x08	端口输入寄存器
GPIO_INTEN	0x18	端口中断使能寄存器
GPIO_INTMASK	0x1C	端口中断屏蔽寄存器
GPIO_INTTYPE_LEVEL	0x20	端口中断等级寄存器
GPIO_INT_POLARITY	0x24	端口中断极性寄存器
GPIO_INTSTATUS	0x28	端口中断状态寄存器
GPIO_RAW_INTSTATUS	0x2C	端口原始中断状态寄存器
GPIO_DEBOUNCE	0x34	防反跳配置寄存器
GPIO_PORT_EOI	0x38	端口中断清除寄存器

5.27.4 寄存器说明

5.27.4.1 GPIO_SWPORT_DR (0x00)

域	位	读写	复位值	描述
reserved	31:16	RW	0x0	保留
Port Data Register	15:0	RW	0x0	如果端口的数据方向位设置为输出模式，则写入该寄存器的值在端口的 I/O 信号线上输出。读取的值等于写入该寄存器的最后一个值。 数据位 0~15 分别对应 GPIO 引脚的 0~15。

5.27.4.2 GPIO_SWPORT_DDR (0x04)

域	位	读写	复位值	描述
reserved	31:16	RW	0x0	保留
Port direction Register	15:0	RW	0x0	写入该寄存器的值独立控制端口中相应数据位的方向。默认方向配置为输入。 0: 输入; 1: 输出 bit0~15 分别代表数据位 0~15 的方向。

5.27.4.3 GPIO_EXT_PORT (0x08)

域	位	读写	复位值	描述
reserved	31:16	RW	0x0	保留
External Port	15:0	RW	0x0	当端口被配置为输入，读取该位置将读取信号的值。 当端口数据方向设置为输出，读取该位置将读取端口的数据寄存器。

5.27.4.4 GPIO_INTEN (0x18)

域	位	读写	复位值	描述
reserved	31:16	RW	0x0	保留
Port Interrupt enable	15:0	RW	0x0	使能端口的每一位配置为中断。默认情况下，中断被禁用。每当将 1 写入该寄存器的某一位时，它将端口上的对应位配置为中断；否则，端口作为正常的 GPIO 信号运行。如果相应的数据方向寄存器设置为输出，则在端口的相应位上中断被禁用。 0：配置端口位为正常 GPIO 信号 1：配置端口位为中断

5.27.4.5 GPIO_INTMASK (0x1C)

域	位	读写	复位值	描述
reserved	31:16	RW	0x0	保留
Port Interrupt mask	15:0	RW	0x0	控制端口上的中断是否屏蔽。默认情况下，所有中断位是未屏蔽的。每当将 1 写入该寄存器的某一位时，它会屏蔽该信号的中断生成。否则，允许通过中断。 0：中断位未屏蔽 1：屏蔽中断位

5.27.4.6 GPIO_INTTYPE_LEVEL (0x20)

域	位	读写	复位值	描述
reserved	31:16	RW	0x0	保留
Port Interrupt level	15:0	RW	0x0	控制端口上可能发送的中断类型。每当 0 写入此寄存器的某个位时，它将中断配置为电平敏感型；否则，它将是边沿敏感型。 0：电平敏感型 1：边沿敏感型

5.27.4.7 GPIO_INT_POLARITY (0x24)

域	位	读写	复位值	描述
reserved	31:16	RW	0x0	保留
Port Interrupt polarity	15:0	RW	0x0	控制端口输入端可能出现的边沿或电平极性。每当 0 写入此寄存器的某个位时，它将中断类型配置为下降沿或低电平敏感；否则，它将是上升沿或高电平敏感。 0：下降沿或低电平 1：上升沿或高电平

5.27.4.8 GPIO_INTSTATUS (0x28)

域	位	读写	复位值	描述
reserved	31:16	R0	0x0	保留
Port Interrupt status	15:0	R0	0x0	端口中断状态

5.27.4.9 GPIO_RAW_INTSTATUS (0x2C)

域	位	读写	复位值	描述
reserved	31:16	R0	0x0	保留
Port Raw Interrupt status	15:0	R0	0x0	端口原始的中断状态

5.27.4.10 GPIO_DEBOUNCE (0x34)

域	位	读写	复位值	描述
reserved	31:16	RW	0x0	保留
Debounce clk config	15:8	RW	0x0	防反跳时钟分频配置。参考时钟 50MHz。 $\text{dbounce_clk} = \text{pclk} / (\text{debounce}[15:8] + 1)$
Debounce enable	7:0	RW	0x0	控制作为中断源的外部信号是否需要消抖以解除任何虚假故障。在该寄存器中写入 1 到某位中将启动消抖电路。信号在内部处理之前必须在外部时钟的两个周期内有效。 0: 禁用防反跳 1: 使能防反跳

5.27.4.11 GPIO_PORT_EOI (0x38)

域	位	读写	复位值	描述
reserved	31:16	W0	0x0	保留
Port Clear interrupt	15:0	W0	0x0	控制端口清除中断。当 1 写入该寄存器的相应位时，中断清除。当端口未配置为中断时，所有中断被清除。 0: 不清除中断 1: 清除中断

5.28 PAD 复用

5.28.1 简介

飞腾派支持 IO PAD 复用，用户可以通过配置控制寄存器来完成复用、上下拉电阻、驱动能力的调整以及延迟的选择。没有以上功能的 PAD 不分配相应的寄存器空间。

5.28.2 操作说明

每个引脚（驱动能力固定的专用 pad 不分配寄存器）使用两个寄存器： x_reg0 和

`x_reg1`。

寄存器 `x_reg0` 用于控制复用功能、驱动能力、上下拉电阻，其中[2:0]用于配置复用功能，[7:4]用于控制驱动能力，[9:8]用于配置上下拉电阻，其他域位为保留位。注意：寄存器 `x_reg0` 请严格参照飞腾派数据手册第 2.1.4 章低速引脚复用表进行配置，禁止将各 IO 引脚的寄存器配置到不存在的功能上。

寄存器 `x_reg1` 用于配置实现 IO 引脚延时调节，延时调节分粗调和细调。粗调每级延时均值约 360ps，精调每级延时均值约 100ps。粗调和细调各有 8 级档位，两种方式可以串联组合使用，最大延时时间约为 3.7ns。每个引脚占用 16 位，高 8 位控制输出延时，低 8 位控制输入延时。

bit[0]：输入延时功能使能；bit[8]：输出延时功能使能

0：不使能；1：使能

bit[3:1]：输入延时精调档位选择；bit[11:9]：输出延时精调档位选择

假设值为 m ，则表示 $(m+1)*100ps$ 延时

bit[6:4]：输入延时粗调档位选择；bit[14:12]：输出延时粗调档位选择

假设值为 n ，则表示 $(n+1)*366ps$ 延时

根据需求配置相应的寄存器即可。PAD 复用功能表请参见飞腾派数据手册。

5.28.3 寄存器列表

表 5-75 PAD 寄存器基地址

名称	基地址
PAD	0x000_32B3_0000

表 5-76 PAD 寄存器列表

寄存器名称	偏移	描述
AN59_reg0	0x0000	上下拉使能配置：默认下拉； 复用功能配置：默认选择功能 0
AW47_reg0	0x0004	驱动能力配置：默认 4 档驱动
AR55_reg0	0x0020	上下拉使能配置：默认上拉； 驱动能力配置：默认 4 档驱动； 复用功能配置：默认选择功能 0
AJ55_reg0	0x0024	上下拉使能配置：默认既不上拉，也不下拉； 驱动能力配置：默认 4 档驱动； 复用功能配置：默认选择功能 0
AL55_reg0	0x0028	上下拉使能配置：默认既不上拉，也不下拉； 驱动能力配置：默认 4 档驱动； 复用功能配置：默认选择功能 0
AL53_reg0	0x002C	上下拉使能配置：默认既不上拉，也不下拉； 驱动能力配置：默认 4 档驱动； 复用功能配置：默认选择功能 0

寄存器名称	偏移	描述
AN51_reg0	0x0030	上下拉使能配置：默认上拉； 驱动能力配置：默认 4 档驱动； 复用功能配置：默认选择功能 0
AR51_reg0	0x0034	上下拉使能配置：默认上拉； 驱动能力配置：默认 4 档驱动； 复用功能配置：默认选择功能 0
BA57_reg0	0x0038	上下拉使能配置：默认既不上拉，也不下拉； 驱动能力配置：默认 4 档驱动； 复用功能配置：默认选择功能 0
BA59_reg0	0x003C	上下拉使能配置：默认既不上拉，也不下拉； 驱动能力配置：默认 4 档驱动； 复用功能配置：默认选择功能 0
AW57_reg0	0x0040	上下拉使能配置：默认既不上拉，也不下拉； 驱动能力配置：默认 4 档驱动； 复用功能配置：默认选择功能 0
AW59_reg0	0x0044	上下拉使能配置：默认既不上拉，也不下拉； 驱动能力配置：默认 4 档驱动； 复用功能配置：默认选择功能 0
AU55_reg0	0x0048	上下拉使能配置：默认既不上拉，也不下拉； 驱动能力配置：默认 4 档驱动； 复用功能配置：默认选择功能 0
AN57_reg0	0x004C	上下拉使能配置：默认下拉； 驱动能力配置：默认 4 档驱动； 复用功能配置：默认选择功能 0
AL59_reg0	0x0050	上下拉使能配置：默认下拉； 驱动能力配置：默认 4 档驱动； 复用功能配置：默认选择功能 0
AJ59_reg0	0x0054	上下拉使能配置：默认下拉； 驱动能力配置：默认 4 档驱动； 复用功能配置：默认选择功能 0
AJ57_reg0	0x0058	上下拉使能配置：默认下拉； 驱动能力配置：默认 4 档驱动； 复用功能配置：默认选择功能 0
AG59_reg0	0x005C	上下拉使能配置：默认下拉； 驱动能力配置：默认 4 档驱动； 复用功能配置：默认选择功能 0
AG57_reg0	0x0060	上下拉使能配置：默认下拉； 驱动能力配置：默认 4 档驱动； 复用功能配置：默认选择功能 0
AE59_reg0	0x0064	上下拉使能配置：默认下拉； 驱动能力配置：默认 4 档驱动； 复用功能配置：默认选择功能 0
AC59_reg0	0x0068	上下拉使能配置：默认下拉； 驱动能力配置：默认 4 档驱动；

寄存器名称	偏移	描述
		复用功能配置：默认选择功能 0
AC57_reg0	0x006C	上下拉使能配置：默认下拉； 驱动能力配置：默认 4 档驱动； 复用功能配置：默认选择功能 0
AR49_reg0	0x0070	上下拉使能配置：默认上拉； 驱动能力配置：默认 4 档驱动； 复用功能配置：默认选择功能 0
BA55_reg0	0x0074	上下拉使能配置：默认上拉； 驱动能力配置：默认 4 档驱动； 复用功能配置：默认选择功能 0
BA53_reg0	0x0078	上下拉使能配置：默认上拉； 驱动能力配置：默认 4 档驱动； 复用功能配置：默认选择功能 0
AR59_reg0	0x007C	上下拉使能配置：默认上拉； 驱动能力配置：默认 4 档驱动； 复用功能配置：默认选择功能 0
AU59_reg0	0x0080	上下拉使能配置：默认上拉； 驱动能力配置：默认 4 档驱动； 复用功能配置：默认选择功能 0
AR57_reg0	0x0084	上下拉使能配置：默认上拉； 驱动能力配置：默认 4 档驱动； 复用功能配置：默认选择功能 0
BA49_reg0	0x0088	上下拉使能配置：默认上拉； 驱动能力配置：默认 4 档驱动； 复用功能配置：默认选择功能 0
AW55_reg0	0x008C	上下拉使能配置：默认既不上拉也不下拉； 驱动能力配置：默认 4 档驱动； 复用功能配置：默认选择功能 0
A35_reg0	0x0090	上下拉使能配置：默认既不上拉也不下拉； 驱动能力配置：默认 4 档驱动； 复用功能配置：默认选择功能 0
R57_reg0	0x0094	上下拉使能配置：默认既不上拉也不下拉； 驱动能力配置：默认 4 档驱动； 复用功能配置：默认选择功能 0
R59_reg0	0x0098	上下拉使能配置：默认既不上拉也不下拉； 驱动能力配置：默认 4 档驱动； 复用功能配置：默认选择功能 0
U59_reg0	0x009C	上下拉使能配置：默认既不上拉也不下拉； 驱动能力配置：默认 4 档驱动； 复用功能配置：默认选择功能 0
W59_reg0	0x00A0	上下拉使能配置：默认既不上拉也不下拉； 驱动能力配置：默认 4 档驱动； 复用功能配置：默认选择功能 0
U57_reg0	0x00A4	上下拉使能配置：默认上拉；

寄存器名称	偏移	描述
		驱动能力配置：默认 4 档驱动； 复用功能配置：默认选择功能 0
AA57_reg0	0x00A8	上下拉使能配置：默认既不上拉也不下拉； 驱动能力配置：默认 4 档驱动； 复用功能配置：默认选择功能 0
AA59_reg0	0x00AC	上下拉使能配置：默认既不上拉也不下拉； 驱动能力配置：默认 4 档驱动； 复用功能配置：默认选择功能 0
AW51_reg0	0x00B0	上下拉使能配置：默认上拉； 驱动能力配置：默认 4 档驱动； 复用功能配置：默认选择功能 0
AU51_reg0	0x00B4	上下拉使能配置：默认既不上拉也不下拉； 驱动能力配置：默认 4 档驱动； 复用功能配置：默认选择功能 0
A39_reg0	0x00B8	上下拉使能配置：默认既不上拉也不下拉； 驱动能力配置：默认 4 档驱动； 复用功能配置：默认选择功能 0
C39_reg0	0x00BC	上下拉使能配置：默认既不上拉也不下拉； 驱动能力配置：默认 4 档驱动； 复用功能配置：默认选择功能 0
C37_reg0	0x00C0	上下拉使能配置：默认下拉； 驱动能力配置：默认 4 档驱动； 复用功能配置：默认选择功能 0
A37_reg0	0x00C4	上下拉使能配置：默认下拉； 驱动能力配置：默认 4 档驱动； 复用功能配置：默认选择功能 0
A41_reg0	0x00C8	上下拉使能配置：默认上拉； 驱动能力配置：默认 4 档驱动； 复用功能配置：默认选择功能 0
A43_reg0	0x00CC	上下拉使能配置：默认上拉； 驱动能力配置：默认 4 档驱动； 复用功能配置：默认选择功能 0
A45_reg0	0x00D0	上下拉使能配置：默认上拉； 驱动能力配置：默认 4 档驱动； 复用功能配置：默认选择功能 0
C45_reg0	0x00D4	上下拉使能配置：默认上拉； 驱动能力配置：默认 4 档驱动； 复用功能配置：默认选择功能 0
A47_reg0	0x00D8	上下拉使能配置：默认上拉； 驱动能力配置：默认 4 档驱动； 复用功能配置：默认选择功能 0
A49_reg0	0x00DC	上下拉使能配置：默认既不上拉也不下拉； 驱动能力配置：默认 4 档驱动； 复用功能配置：默认选择功能 0

寄存器名称	偏移	描述
C49_reg0	0x00E0	上下拉使能配置：默认上拉； 驱动能力配置：默认 4 档驱动； 复用功能配置：默认选择功能 0
A51_reg0	0x00E4	上下拉使能配置：默认上拉； 驱动能力配置：默认 4 档驱动； 复用功能配置：默认选择功能 0
A33_reg0	0x00E8	上下拉使能配置：默认既不上拉也不下拉； 驱动能力配置：默认 4 档驱动； 复用功能配置：默认选择功能 0
C33_reg0	0x00EC	上下拉使能配置：默认既不上拉也不下拉； 驱动能力配置：默认 4 档驱动； 复用功能配置：默认选择功能 0
C31_reg0	0x00F0	上下拉使能配置：默认上拉； 驱动能力配置：默认 4 档驱动； 复用功能配置：默认选择功能 0
A31_reg0	0x00F4	上下拉使能配置：默认上拉； 驱动能力配置：默认 4 档驱动； 复用功能配置：默认选择功能 0
AJ53_reg0	0x00F8	上下拉使能配置：默认上拉； 驱动能力配置：默认 4 档驱动； 复用功能配置：默认选择功能 0
AL49_reg0	0x00FC	上下拉使能配置：默认既不上拉也不下拉； 驱动能力配置：默认 4 档驱动； 复用功能配置：默认选择功能 0
AL47_reg0	0x0100	上下拉使能配置：默认下拉； 驱动能力配置：默认 4 档驱动； 复用功能配置：默认选择功能 0
AN49_reg0	0x0104	上下拉使能配置：默认上拉； 驱动能力配置：默认 4 档驱动； 复用功能配置：默认选择功能 0
AG51_reg0	0x0108	上下拉使能配置：默认上拉； 驱动能力配置：默认 4 档驱动； 复用功能配置：默认选择功能 0
AJ51_reg0	0x010C	上下拉使能配置：默认上拉； 驱动能力配置：默认 4 档驱动； 复用功能配置：默认选择功能 0
AG49_reg0	0x0110	上下拉使能配置：默认上拉； 驱动能力配置：默认 4 档驱动； 复用功能配置：默认选择功能 0
AE55_reg0	0x0114	上下拉使能配置：默认上拉； 驱动能力配置：默认 4 档驱动； 复用功能配置：默认选择功能 0
AE53_reg0	0x0118	上下拉使能配置：默认上拉； 驱动能力配置：默认 4 档驱动；

寄存器名称	偏移	描述
		复用功能配置：默认选择功能 0
AG55_reg0	0x011C	上下拉使能配置：默认上拉； 驱动能力配置：默认 4 档驱动； 复用功能配置：默认选择功能 0
AJ49_reg0	0x0120	上下拉使能配置：默认既不上拉也不下拉； 驱动能力配置：默认 4 档驱动； 复用功能配置：默认选择功能 0
AC55_reg0	0x0124	上下拉使能配置：默认上拉； 驱动能力配置：默认 4 档驱动； 复用功能配置：默认选择功能 0
AC53_reg0	0x0128	上下拉使能配置：默认既不上拉也不下拉； 驱动能力配置：默认 4 档驱动； 复用功能配置：默认选择功能 0
AE51_reg0	0x012C	上下拉使能配置：默认下拉； 驱动能力配置：默认 4 档驱动； 复用功能配置：默认选择功能 0
W51_reg0	0x0130	上下拉使能配置：默认既不上拉也不下拉； 驱动能力配置：默认 4 档驱动； 复用功能配置：默认选择功能 0
W55_reg0	0x0134	上下拉使能配置：默认既不上拉也不下拉； 驱动能力配置：默认 4 档驱动； 复用功能配置：默认选择功能 0
W53_reg0	0x0138	上下拉使能配置：默认上拉； 驱动能力配置：默认 4 档驱动； 复用功能配置：默认选择功能 0
U55_reg0	0x013C	上下拉使能配置：默认上拉； 驱动能力配置：默认 4 档驱动； 复用功能配置：默认选择功能 0
U53_reg0	0x0140	上下拉使能配置：默认上拉； 驱动能力配置：默认 4 档驱动； 复用功能配置：默认选择功能 0
AE49_reg0	0x0144	上下拉使能配置：默认上拉； 驱动能力配置：默认 4 档驱动； 复用功能配置：默认选择功能 0
AC49_reg0	0x0148	上下拉使能配置：默认上拉； 驱动能力配置：默认 4 档驱动； 复用功能配置：默认选择功能 0
AE47_reg0	0x014C	上下拉使能配置：默认上拉； 驱动能力配置：默认 4 档驱动； 复用功能配置：默认选择功能 0
AA47_reg0	0x0150	上下拉使能配置：默认上拉； 驱动能力配置：默认 4 档驱动； 复用功能配置：默认选择功能 0
AA49_reg0	0x0154	上下拉使能配置：默认上拉；

寄存器名称	偏移	描述
		驱动能力配置：默认 4 档驱动； 复用功能配置：默认选择功能 0
W49_reg0	0x0158	上下拉使能配置：默认上拉； 驱动能力配置：默认 4 档驱动； 复用功能配置：默认选择功能 0
AA51_reg0	0x015C	上下拉使能配置：默认上拉； 驱动能力配置：默认 4 档驱动； 复用功能配置：默认选择功能 0
U49_reg0	0x0160	上下拉使能配置：默认上拉； 驱动能力配置：默认 4 档驱动； 复用功能配置：默认选择功能 0
G59_reg0	0x0164	上下拉使能配置：默认上拉； 驱动能力配置：默认 4 档驱动； 复用功能配置：默认选择功能 0
J59_reg0	0x0168	上下拉使能配置：默认上拉； 驱动能力配置：默认 4 档驱动； 复用功能配置：默认选择功能 0
L57_reg0	0x016C	上下拉使能配置：默认上拉； 驱动能力配置：默认 4 档驱动； 复用功能配置：默认选择功能 0
C59_reg0	0x0170	上下拉使能配置：默认上拉； 驱动能力配置：默认 4 档驱动； 复用功能配置：默认选择功能 0
E59_reg0	0x0174	上下拉使能配置：默认上拉； 驱动能力配置：默认 4 档驱动； 复用功能配置：默认选择功能 0
J57_reg0	0x0178	上下拉使能配置：默认既不上拉也不下拉； 驱动能力配置：默认 4 档驱动； 复用功能配置：默认选择功能 0
L59_reg0	0x017C	上下拉使能配置：默认上拉； 驱动能力配置：默认 4 档驱动； 复用功能配置：默认选择功能 0
N59_reg0	0x0180	上下拉使能配置：默认下拉； 驱动能力配置：默认 4 档驱动； 复用功能配置：默认选择功能 0
C57_reg0	0x0184	上下拉使能配置：默认下拉； 驱动能力配置：默认 4 档驱动； 复用功能配置：默认选择功能 0
E57_reg0	0x0188	上下拉使能配置：默认下拉； 驱动能力配置：默认 4 档驱动； 复用功能配置：默认选择功能 0
E31_reg0	0x018C	上下拉使能配置：默认既不上拉也不下拉； 驱动能力配置：默认 4 档驱动； 复用功能配置：默认选择功能 0

寄存器名称	偏移	描述
G31_reg0	0x0190	上下拉使能配置：默认既不上拉也不下拉； 驱动能力配置：默认 4 档驱动； 复用功能配置：默认选择功能 0
N41_reg0	0x0194	上下拉使能配置：默认既不上拉也不下拉； 驱动能力配置：默认 4 档驱动； 复用功能配置：默认选择功能 0
N39_reg0	0x0198	上下拉使能配置：默认既不上拉也不下拉； 驱动能力配置：默认 4 档驱动； 复用功能配置：默认选择功能 0
J33_reg0	0x019C	上下拉使能配置：默认上拉； 驱动能力配置：默认 4 档驱动； 复用功能配置：默认选择功能 0
N33_reg0	0x01A0	上下拉使能配置：默认既不上拉也不下拉； 驱动能力配置：默认 4 档驱动； 复用功能配置：默认选择功能 0
L33_reg0	0x01A4	上下拉使能配置：默认既不上拉也不下拉； 驱动能力配置：默认 4 档驱动； 复用功能配置：默认选择功能 0
N45_reg0	0x01A8	上下拉使能配置：默认既不上拉也不下拉； 驱动能力配置：默认 4 档驱动； 复用功能配置：默认选择功能 0
N43_reg0	0x01AC	上下拉使能配置：默认既不上拉也不下拉； 驱动能力配置：默认 4 档驱动； 复用功能配置：默认选择功能 0
L31_reg0	0x01B0	上下拉使能配置：默认既不上拉也不下拉； 驱动能力配置：默认 4 档驱动； 复用功能配置：默认选择功能 0
J31_reg0	0x01B4	上下拉使能配置：默认既不上拉也不下拉； 驱动能力配置：默认 4 档驱动； 复用功能配置：默认选择功能 0
J29_reg0	0x01B8	上下拉使能配置：默认既不上拉也不下拉； 驱动能力配置：默认 4 档驱动； 复用功能配置：默认选择功能 0
E29_reg0	0x01BC	上下拉使能配置：默认既不上拉也不下拉； 驱动能力配置：默认 4 档驱动； 复用功能配置：默认选择功能 0
G29_reg0	0x01C0	上下拉使能配置：默认既不上拉也不下拉； 驱动能力配置：默认 4 档驱动； 复用功能配置：默认选择功能 0
N27_reg0	0x01C4	上下拉使能配置：默认既不上拉也不下拉； 驱动能力配置：默认 4 档驱动； 复用功能配置：默认选择功能 0
L29_reg0	0x01C8	上下拉使能配置：默认既不上拉也不下拉； 驱动能力配置：默认 4 档驱动；

寄存器名称	偏移	描述
		复用功能配置：默认选择功能 0
J37_reg0	0x01CC	上下拉使能配置：默认既不上拉也不下拉； 驱动能力配置：默认 4 档驱动； 复用功能配置：默认选择功能 0
J39_reg0	0x01D0	上下拉使能配置：默认既不上拉也不下拉； 驱动能力配置：默认 4 档驱动； 复用功能配置：默认选择功能 0
G41_reg0	0x01D4	上下拉使能配置：默认既不上拉也不下拉； 驱动能力配置：默认 4 档驱动； 复用功能配置：默认选择功能 0
E43_reg0	0x01D8	上下拉使能配置：默认既不上拉也不下拉； 驱动能力配置：默认 4 档驱动； 复用功能配置：默认选择功能 0
L43_reg0	0x01DC	上下拉使能配置：默认既不上拉也不下拉； 驱动能力配置：默认 4 档驱动； 复用功能配置：默认选择功能 0
C43_reg0	0x01E0	上下拉使能配置：默认既不上拉也不下拉； 驱动能力配置：默认 4 档驱动； 复用功能配置：默认选择功能 0
E41_reg0	0x01E4	上下拉使能配置：默认既不上拉也不下拉； 驱动能力配置：默认 4 档驱动； 复用功能配置：默认选择功能 0
L45_reg0	0x01E8	上下拉使能配置：默认既不上拉也不下拉； 驱动能力配置：默认 4 档驱动； 复用功能配置：默认选择功能 0
J43_reg0	0x01EC	上下拉使能配置：默认既不上拉也不下拉； 驱动能力配置：默认 4 档驱动； 复用功能配置：默认选择功能 0
J41_reg0	0x01F0	上下拉使能配置：默认既不上拉也不下拉； 驱动能力配置：默认 4 档驱动； 复用功能配置：默认选择功能 0
L39_reg0	0x01F4	上下拉使能配置：默认既不上拉也不下拉； 驱动能力配置：默认 4 档驱动； 复用功能配置：默认选择功能 0
E37_reg0	0x01F8	上下拉使能配置：默认既不上拉也不下拉； 驱动能力配置：默认 4 档驱动； 复用功能配置：默认选择功能 0
E35_reg0	0x01FC	上下拉使能配置：默认既不上拉也不下拉； 驱动能力配置：默认 4 档驱动； 复用功能配置：默认选择功能 0
G35_reg0	0x0200	上下拉使能配置：默认既不上拉也不下拉； 驱动能力配置：默认 4 档驱动； 复用功能配置：默认选择功能 0
J35_reg0	0x0204	上下拉使能配置：默认既不上拉也不下拉；

寄存器名称	偏移	描述
		驱动能力配置：默认 4 档驱动； 复用功能配置：默认选择功能 0
L37_reg0	0x0208	上下拉使能配置：默认既不上拉也不下拉； 驱动能力配置：默认 4 档驱动； 复用功能配置：默认选择功能 0
N35_reg0	0x020C	上下拉使能配置：默认既不上拉也不下拉； 驱动能力配置：默认 4 档驱动； 复用功能配置：默认选择功能 0
R51_reg0	0x0210	上下拉使能配置：默认既不上拉也不下拉； 驱动能力配置：默认 4 档驱动； 复用功能配置：默认选择功能 0
R49_reg0	0x0214	上下拉使能配置：默认既不上拉也不下拉； 驱动能力配置：默认 4 档驱动； 复用功能配置：默认选择功能 0
N51_reg0	0x0218	上下拉使能配置：默认既不上拉也不下拉； 驱动能力配置：默认 4 档驱动； 复用功能配置：默认选择功能 0
N55_reg0	0x021C	上下拉使能配置：默认既不上拉也不下拉； 驱动能力配置：默认 4 档驱动； 复用功能配置：默认选择功能 0
L55_reg0	0x0220	上下拉使能配置：默认既不上拉也不下拉； 驱动能力配置：默认 4 档驱动； 复用功能配置：默认选择功能 0
J55_reg0	0x0224	上下拉使能配置：默认既不上拉也不下拉； 驱动能力配置：默认 4 档驱动； 复用功能配置：默认选择功能 0
J45_reg0	0x0228	上下拉使能配置：默认既不上拉也不下拉； 驱动能力配置：默认 4 档驱动； 复用功能配置：默认选择功能 0
E47_reg0	0x022C	上下拉使能配置：默认既不上拉也不下拉； 驱动能力配置：默认 4 档驱动； 复用功能配置：默认选择功能 0
G47_reg0	0x0230	上下拉使能配置：默认既不上拉也不下拉； 驱动能力配置：默认 4 档驱动； 复用功能配置：默认选择功能 0
J47_reg0	0x0234	上下拉使能配置：默认既不上拉也不下拉； 驱动能力配置：默认 4 档驱动； 复用功能配置：默认选择功能 0
J49_reg0	0x0238	上下拉使能配置：默认既不上拉也不下拉； 驱动能力配置：默认 4 档驱动； 复用功能配置：默认选择功能 0
N49_reg0	0x023C	上下拉使能配置：默认既不上拉也不下拉； 驱动能力配置：默认 4 档驱动； 复用功能配置：默认选择功能 0

寄存器名称	偏移	描述
L51_reg0	0x0240	上下拉使能配置：默认既不上拉也不下拉； 驱动能力配置：默认 4 档驱动； 复用功能配置：默认选择功能 0
L49_reg0	0x0244	上下拉使能配置：默认既不上拉也不下拉； 驱动能力配置：默认 4 档驱动； 复用功能配置：默认选择功能 0
N53_reg0	0x0248	上下拉使能配置：默认既不上拉也不下拉； 驱动能力配置：默认 4 档驱动； 复用功能配置：默认选择功能 0
J53_reg0	0x024C	上下拉使能配置：默认既不上拉也不下拉； 驱动能力配置：默认 4 档驱动； 复用功能配置：默认选择功能 0
AJ55_reg1	0x1024	输入输出延时配置：默认不使能输入输出延时
AL55_reg1	0x1028	输入输出延时配置：默认不使能输入输出延时
AL53_reg1	0x102C	输入输出延时配置：默认不使能输入输出延时
AN51_reg1	0x1030	输入输出延时配置：默认不使能输入输出延时
AR51_reg1	0x1034	输入输出延时配置：默认不使能输入输出延时
AJ57_reg1	0x1058	输入输出延时配置：默认不使能输入输出延时
AG59_reg1	0x105C	输入输出延时配置：默认不使能输入输出延时
AG57_reg1	0x1060	输入输出延时配置：默认不使能输入输出延时
AE59_reg1	0x1064	输入输出延时配置：默认不使能输入输出延时
BA55_reg1	0x1074	输入输出延时配置：默认不使能输入输出延时
BA53_reg1	0x1078	输入输出延时配置：默认不使能输入输出延时
AR59_reg1	0x107C	输入输出延时配置：默认不使能输入输出延时
AU59_reg1	0x1080	输入输出延时配置：默认不使能输入输出延时
A45_reg1	0x10D0	输入输出延时配置：默认不使能输入输出延时
C45_reg1	0x10D4	输入输出延时配置：默认不使能输入输出延时
A47_reg1	0x10D8	输入输出延时配置：默认不使能输入输出延时
A49_reg1	0x10DC	输入输出延时配置：默认不使能输入输出延时
C49_reg1	0x10E0	输入输出延时配置：默认不使能输入输出延时
A51_reg1	0x10E4	输入输出延时配置：默认不使能输入输出延时
A33_reg1	0x10E8	输入输出延时配置：默认不使能输入输出延时
C33_reg1	0x10EC	输入输出延时配置：默认不使能输入输出延时
C31_reg1	0x10F0	输入输出延时配置：默认不使能输入输出延时
A31_reg1	0x10F4	输入输出延时配置：默认不使能输入输出延时
AJ53_reg1	0x10F8	输入输出延时配置：默认不使能输入输出延时
AL49_reg1	0x10FC	输入输出延时配置：默认不使能输入输出延时
AL47_reg1	0x1100	输入输出延时配置：默认不使能输入输出延时
AN49_reg1	0x1104	输入输出延时配置：默认不使能输入输出延时
AG51_reg1	0x1108	输入输出延时配置：默认不使能输入输出延时
AJ51_reg1	0x110C	输入输出延时配置：默认不使能输入输出延时
AG49_reg1	0x1110	输入输出延时配置：默认不使能输入输出延时
AE55_reg1	0x1114	输入输出延时配置：默认不使能输入输出延时
AE53_reg1	0x1118	输入输出延时配置：默认不使能输入输出延时

寄存器名称	偏移	描述
AG55_reg1	0x111C	输入输出延时配置：默认不使能输入输出延时
AJ49_reg1	0x1120	输入输出延时配置：默认不使能输入输出延时
AC55_reg1	0x1124	输入输出延时配置：默认不使能输入输出延时
AC53_reg1	0x1128	输入输出延时配置：默认不使能输入输出延时
AE51_reg1	0x112C	输入输出延时配置：默认不使能输入输出延时
W51_reg1	0x1130	输入输出延时配置：默认不使能输入输出延时
W53_reg1	0x1138	输入输出延时配置：默认不使能输入输出延时
U55_reg1	0x113C	输入输出延时配置：默认不使能输入输出延时
U53_reg1	0x1140	输入输出延时配置：默认不使能输入输出延时
AE49_reg1	0x1144	输入输出延时配置：默认不使能输入输出延时
AC49_reg1	0x1148	输入输出延时配置：默认不使能输入输出延时
AE47_reg1	0x114C	输入输出延时配置：默认不使能输入输出延时
AA47_reg1	0x1150	输入输出延时配置：默认不使能输入输出延时
AA49_reg1	0x1154	输入输出延时配置：默认不使能输入输出延时
W49_reg1	0x1158	输入输出延时配置：默认不使能输入输出延时
AA51_reg1	0x115C	输入输出延时配置：默认不使能输入输出延时
U49_reg1	0x1160	输入输出延时配置：默认不使能输入输出延时
J59_reg1	0x1168	输入输出延时配置：默认不使能输入输出延时
L57_reg1	0x116C	输入输出延时配置：默认不使能输入输出延时
C59_reg1	0x1170	输入输出延时配置：默认不使能输入输出延时
E59_reg1	0x1174	输入输出延时配置：默认不使能输入输出延时
J57_reg1	0x1178	输入输出延时配置：默认不使能输入输出延时
L59_reg1	0x117C	输入输出延时配置：默认不使能输入输出延时
N59_reg1	0x1180	输入输出延时配置：默认不使能输入输出延时
E31_reg1	0x118C	输入输出延时配置：默认不使能输入输出延时
G31_reg1	0x1190	输入输出延时配置：默认不使能输入输出延时
N41_reg1	0x1194	输入输出延时配置：默认不使能输入输出延时
N39_reg1	0x1198	输入输出延时配置：默认不使能输入输出延时
J33_reg1	0x119C	输入输出延时配置：默认不使能输入输出延时
N33_reg1	0x11A0	输入输出延时配置：默认不使能输入输出延时
L33_reg1	0x11A4	输入输出延时配置：默认不使能输入输出延时
N45_reg1	0x11A8	输入输出延时配置：默认不使能输入输出延时
N43_reg1	0x11AC	输入输出延时配置：默认不使能输入输出延时
L31_reg1	0x11B0	输入输出延时配置：默认不使能输入输出延时
J31_reg1	0x11B4	输入输出延时配置：默认不使能输入输出延时
J29_reg1	0x11B8	输入输出延时配置：默认不使能输入输出延时
E29_reg1	0x11BC	输入输出延时配置：默认不使能输入输出延时
G29_reg1	0x11C0	输入输出延时配置：默认不使能输入输出延时
J37_reg1	0x11CC	输入输出延时配置：默认不使能输入输出延时
J39_reg1	0x11D0	输入输出延时配置：默认不使能输入输出延时
G41_reg1	0x11D4	输入输出延时配置：默认不使能输入输出延时
E43_reg1	0x11D8	输入输出延时配置：默认不使能输入输出延时
L43_reg1	0x11DC	输入输出延时配置：默认不使能输入输出延时

寄存器名称	偏移	描述
C43_reg1	0x11E0	输入输出延时配置：默认不使能输入输出延时
E41_reg1	0x11E4	输入输出延时配置：默认不使能输入输出延时
L45_reg1	0x11E8	输入输出延时配置：默认不使能输入输出延时
J43_reg1	0x11EC	输入输出延时配置：默认不使能输入输出延时
J41_reg1	0x11F0	输入输出延时配置：默认不使能输入输出延时
L39_reg1	0x11F4	输入输出延时配置：默认不使能输入输出延时
E37_reg1	0x11F8	输入输出延时配置：默认不使能输入输出延时
E35_reg1	0x11FC	输入输出延时配置：默认不使能输入输出延时
G35_reg1	0x1200	输入输出延时配置：默认不使能输入输出延时
L55_reg1	0x1220	输入输出延时配置：默认不使能输入输出延时
J55_reg1	0x1224	输入输出延时配置：默认不使能输入输出延时
J45_reg1	0x1228	输入输出延时配置：默认不使能输入输出延时
E47_reg1	0x122C	输入输出延时配置：默认不使能输入输出延时
G47_reg1	0x1230	输入输出延时配置：默认不使能输入输出延时
J47_reg1	0x1234	输入输出延时配置：默认不使能输入输出延时
J49_reg1	0x1238	输入输出延时配置：默认不使能输入输出延时
N49_reg1	0x123C	输入输出延时配置：默认不使能输入输出延时
L51_reg1	0x1240	输入输出延时配置：默认不使能输入输出延时
L49_reg1	0x1244	输入输出延时配置：默认不使能输入输出延时
N53_reg1	0x1248	输入输出延时配置：默认不使能输入输出延时
J53_reg1	0x124C	输入输出延时配置：默认不使能输入输出延时

5.28.4 寄存器说明

5.28.4.1 x_reg0

x 为引脚名。

域	位	读写	复位值	描述
reserved	31:10	—	复位值参照 PAD 寄存器列表对应寄存器描述的默认配置	保留
res	9:8	RW		res[9:8]=2'b00：上下拉电阻都不使能 res[9:8]=2'b01：使能下拉电阻 res[9:8]=2'b10：使能上拉电阻 res[9:8]=2'b11：保留 上下拉电阻取值范围为 $45 \pm 15k\Omega$
drive	7:4	RW		配置驱动能力，驱动能力分为 16 级 drive[7:4]=4'b0000：0 档驱动能力（最弱） drive[7:4]=4'b0001：1 档驱动能力 drive[7:4]=4'b0010：2 档驱动能力 ... drive[7:4]=4'b1111：15 档驱动能力（最强）
reserved	3	—		保留
func_sel	2:0	RW		配置复用功能 func_sel[2:0]=n：选择第 n 个 func

域	位	读写	复位值	描述
				3'b000: 选择 func0 3'b001: 选择 func1 3'b010: 选择 func2 3'b011: 选择 func3 3'b100: 选择 func4 3'b101: 选择 func5 3'b110: 选择 func6

5.28.4.2 x_reg1

x 为引脚名。

域	位	读写	复位值	描述
reserved	31:16	—	0x0	保留
output_delay	15:8	RW	0x0	out_delay[8]=1: 使能输出延时 out_delay[11:9]: 输出延时精调 out_delay[14:12]: 输出延时粗调 默认不使能输出延时
input_delay	7:0	RW	0x0	in_delay[0]=1: 使能输入延时 in_delay[3:1]: 输入延时精调 in_delay[6:4]: 输入延时粗调 默认不使能输入延时

5.29 RAS

5.29.1 错误的分类与上报

飞腾派检测的错误类型分以下两种：

- 可纠错误：可以立刻纠正的错误，如 ECC 单位错；
- 不可纠错误：无法纠正的错误，如 ECC 多位错或者奇偶校验错。

支持以下两种报错机制：

- 错误处理中断（Fault Handling Interrupt）
- 错误恢复中断（Error Recovery Interrupt）

不同的错误类型可以通过不同的中断报错，可纠错误只能通过错误处理中断报错，而不可纠错误既可以走错误处理中断，也可以走错误恢复中断，或者同时触发两种中断。

目前软件对于这两种中断没有明确区分处理流程，建议每种错误只走一种中断。为了验证软件处理流程的正确性，可使用错误注入寄存器模拟错误中断上报。

对于不可纠错误，是检测即上报。所以往错误注入寄存器写入对应的错误编号，可以产生错误恢复中断上报。

对于可纠错误，是溢出后上报。有两种方式：

一是写对应错误编号的错误记录杂项寄存器 0 (ERR<n>MISC0) 为 0xFF00000000，然后往错误注入寄存器写入对应的错误编号，可以产生错误处理中断上报。

二是重复操作 255 次，往错误注入寄存器写入对应的错误编号，可以产生错误处理中断上报。

一个模块里面是安全还是非安全是可配置的，如果配置为安全，那么这个错误记录的状态和相关地址、计数只能由安全态去访问。修改模块安全属性时，需要同时修改对应模块报告错误的安全属性寄存器 err_sec_set 对应的 bit 位，1 为安全，0 为非安全。

5.29.2 错误记录

飞腾派一共有三组 ras 记录，分别为 ras_soc、ras_peu_psu 和 ras_peu 错误记录，用于记录片上各模块的错误中断信息和 PCIe 控制器内错误中断信息。

5.29.2.1 ras_soc 错误记录

表 5-77 ras_soc 错误信号统计表

错误编号	信号名	类型	含义	软件处理建议	地址
0	lsd_nfc_ras_err	不可纠	nandflash 模块错误汇总信号	软件清除中断，查询具体错误状态	lsd_nfc_ras_err_addr
4	lsd_lbc_ras_err	不可纠	LocalBus 的外接设备响应超时错误	查询 LBC_EVTER_STAT 寄存器查询错误类型，并清除错误，然后进行复位操作	NA
5	usb3_err_0	不可纠	USB3 控制器 0 总线错误的中断信号	清除错误可继续发送请求	NA
6	usb3_err_1	不可纠	USB3 控制器 1 总线错误的中断信号	清除错误可继续发送请求	NA
7	gsd_gmu_mac0_asf_nonfatal_int	不可纠	MAC0 控制器一般错误中断信号	控制器内部有相关寄存器记录了一些信息，可供查询	NA
8	gsd_gmu_mac0_asf_fatal_int	不可纠	MAC0 控制器致命错误中断信号	控制器内部有相关寄存器记录了一些信息，可供查询	NA
9	gsd_gmu_mac0_asf_trans_to_err	不可纠	MAC 控制器总线错误中断信号	清除错误可继续发送请求	NA
10	gsd_gmu_mac0_asf_protocol_err	不可纠	MAC 协议级错误中断信号	清除错误可继续发送请求	NA
11	gsd_gmu_mac1_asf_nonfatal_int	不可纠	MAC1 控制器一般错误中断信号	控制器内部有相关寄存器记录了一些信息，可供查询	NA
12	gsd_gmu_mac1_asf_fatal_int	不可纠	MAC1 控制器致命错误中断信号	控制器内部有相关寄存器记录了一些信息，可供查询	NA
13	gsd_gmu_mac1_asf_trans_to_err	不可纠	MAC 控制器总线错误中断信号	清除错误可继续发送请求	NA
14	gsd_gmu_mac1_as	不可	MAC 协议级错误中	清除错误可继续发送请求	NA

错误编号	信号名	类型	含义	软件处理建议	地址
	f_protocol_err	纠	断信号		
15	gsd_gmu_mac2_asf_nonfatal_int	不可纠	MAC2 控制器一般错误中断信号	控制器内部有相关寄存器记录了一些信息，可供查询	NA
16	gsd_gmu_mac2_asf_fatal_int	不可纠	MAC2 控制器致命错误中断信号	控制器内部有相关寄存器记录了一些信息，可供查询	NA
17	gsd_gmu_mac2_asf_trans_to_err	不可纠	MAC 控制器总线错误中断信号	清除错误可继续发送请求	NA
18	gsd_gmu_mac2_asf_protocol_err	不可纠	MAC 协议级错误中断信号	清除错误可继续发送请求	NA
19	gsd_gmu_mac3_asf_nonfatal_int	不可纠	MAC3 控制器一般错误中断信号	控制器内部有相关寄存器记录了一些信息，可供查询	NA
20	gsd_gmu_mac3_asf_fatal_int	不可纠	MAC3 控制器致命错误中断信号	控制器内部有相关寄存器记录了一些信息，可供查询	NA
21	gsd_gmu_mac3_asf_trans_to_err	不可纠	MAC 控制器总线错误中断信号	清除错误可继续发送请求	NA
22	gsd_gmu_mac3_asf_protocol_err	不可纠	MAC 协议级错误中断信号	清除错误可继续发送请求	NA
23	dmu_ras_ecc_corrected_err	可纠	从 DIMM 读回的数据校验出现单位错，自动纠正，AXI 总线读响应正常	可不处理	NA
24	dmu_ras_ecc_uncorrected_err	不可纠	从 DIMM 读回数据校验出现多位错，无法纠正，AXI 总线的读响应带错误标记	控制器内部有相关寄存器记录了一些信息，可供查询	NA
25	cci_ras_nERRIRQ	不可纠	指示互联某端口存在 imprecise error	查询 Imprecise Error 寄存器和 EventBus，解决后对 Imprecise Error 寄存器相应端口 bit 写 1'b1 以清除中断	NA
26	smmu_tcu_ras_irq	不可纠	TCU 指示 RAS 中断	查询 SMMU_TCU_ERRSTATUS 寄存器	NA
27	smmu_tbu0_ras_irq	不可纠	TBU0 RAS 中断	查询 SMMU_TBU0_ERRSTATUS 寄存器	NA
28	smmu_tbu1_ras_irq	不可纠	TBU1 RAS 中断	查询 SMMU_TBU1_ERRSTATUS 寄存器	NA
29	smmu_tbu2_ras_irq	不可纠	TBU2 RAS 中断	查询 SMMU_TBU2_ERRSTATUS 寄存器	NA
30	ocm_sram_ue	不可纠	OCM RAM 出现不可纠 ECC 错	建议软件处理	NA
31	ocm_sram_ce	可纠	OCM RAM 出现可纠的 ECC 错	可不处理	NA
32	int_axim_err	不可纠	int 模块 axi 主接口错误	进行 int 复位操作	NA

错误编号	信号名	类型	含义	软件处理建议	地址
33	int_fatal_error	不可纠	int 模块内 ECC 校验错误	进行 int 复位操作	NA
34	nEXTERRIRQ_cluster0	不可纠	cluster0 中 I2c 以外总线错误	对 L2ECTLR 的 [29] 写 0 清错误标志, 不影响使用可不作处理, 影响系统功能时对 cluster0 进行复位操作	NA
35	nINTERRIRQ_cluster0	不可纠	cluster0 中 I2c 内部 ECC 校验错误	对 L2ECTLR 的 [30] 写 0 清错误标志, 不影响使用可不作处理, 影响系统功能时对 cluster0 进行复位操作	NA
36	nEXTERRIRQ_cluster1	不可纠	cluster0 中 I2c 以外总线错误	对 L2ECTLR 的 [29] 写 0 清错误标志, 不影响使用可不作处理, 影响系统功能时对 cluster1 进行复位操作	NA
37	nINTERRIRQ_cluster1	不可纠	cluster0 中 I2c 内部 ECC 校验错误	对 L2ECTLR 的 [30] 写 0 清错误标志, 不影响使用可不作处理, 影响系统功能时对 cluster1 进行复位操作	NA
38	nEXTERRIRQ_cluster2	不可纠	cluster0 中 I2c 以外总线错误	对 L2ECTLR 的 [29] 写 0 清错误标志, 不影响使用可不作处理, 影响系统功能时对 cluster2 进行复位操作	NA
39	nINTERRIRQ_cluster2	不可纠	cluster0 中 I2c 内部 ECC 校验错误	对 L2ECTLR 的 [30] 写 0 清错误标志, 不影响使用可不作处理, 影响系统功能时对 cluster2 进行复位操作	NA

5.29.2.2 ras_peu_psu 错误记录

表 5-78 ras_peu_psu 错误信号统计表

错误编号	错误相关信号名	错误类别	错误详细说明	除错误信号外, 是否还有地址信号	可纠(错误处理), 不可纠(错误恢复)	软件处理建议	地址
0	pio_rd_addr_error	不可纠	从 cluster 流出的未命中读地址。	有	错误恢复中断	读取 ras_peu_psu 模块 ERR<0>ADDR 确认软件发出不正确的地址, 定位原因。	pio_rd_addr_error_addr
1	pio_wr_addr_error	不可纠	从 cluster 流出的未命中写地址	有	错误恢复中断	读取 ras_peu_psu 模块 ERR<1>ADDR 确认软件发出不正确的地址, 定位原因。	pio_wr_addr_error_addr
2	pio_rd_ti	不可	pio 通路读	有	错误恢复	不处理, 不一定是	pio_rd_time

错误编号	错误相关信号名	错误类别	错误详细说明	除错误信号外，是否还有地址信号	可纠(错误处理)，不可纠(错误恢复)	软件处理建议	地址
	meout	纠	超时请求		中断	真正的问题，可能是设计的超时时间过短的原因。	out_addr
3	pio_wr_timeout	不可纠	pio 通路写超时请求	有	错误恢复中断	不处理，不一定是真正的问题，可能是设计的超时时间过短的原因。	pio_wr_timeout_addr
4	axi_b_rsp_error	不可纠	pio 写响应错误	有	错误恢复中断	不处理	axi_b_rsp_error_addr
5	axi_r_rsp_error	不可纠	pio 读响应错误	有	错误恢复中断	不处理	axi_r_rsp_error_addr

5.29.2.3 ras_peu 错误记录

表 5-79 ras_peu 错误信号统计表

错误编号	错误相关信号名	错误类别	错误详细说明	除错误信号外，是否还有地址信号	可纠(错误处理)，不可纠(错误恢复)	软件处理建议	地址
0	pio_rd_addr_error	不可纠	从 cluster 流出的未命中读地址。	有	错误恢复中断	读取 ras_peu 模块 ERR<0> ADDR 确认软件发出不正确的地址，定位原因。	pio_rd_addr_error_addr
1	pio_wr_addr_error	不可纠	从 cluster 流出的未命中写地址	有	错误恢复中断	读取 ras_peu 模块 ERR<1>ADDR 确认软件发出不正确的地址，定位原因。	pio_wr_addr_error_addr
2	pio_rd_timeout	不可纠	pio 通路读超时请求	有	错误恢复中断	不处理，不一定是真正的问题，可能是设计的超时时间过短的原因。	pio_rd_timeout_addr
3	pio_wr_timeout	不可纠	pio 通路写超时请求	有	错误恢复中断	不处理，不一定是真正的问题，可能是设计的超时时间过短的原因。	pio_wr_timeout_addr
4	axi_b_rsp_error	不可纠	pio 写响应错误	有	错误恢复中断	不处理	axi_b_rsp_error_addr
5	axi_r_rsp_error	不可纠	pio 读响应错误	有	错误恢复中断	不处理	axi_r_rsp_error_addr

5.29.3 寄存器说明

表 5-80 RAS 错误记录寄存器基地址

名称	基地址
ras_soc	0x000_32B2_8000
ras_peu_psu	0x000_3140_0000
ras_peu	0x000_3140_1000

每一组错误记录的寄存器如下表所示（ n 为错误编号）。

表 5-81 RAS 错误记录寄存器列表

页内偏移地址	位宽	名称	说明
$0x000+64 \times n$	64	ERR< n >FR	错误记录属性寄存器
$0x008+64 \times n$	64	ERR< n >CTLR	错误记录控制寄存器
$0x010+64 \times n$	64	ERR< n >STATUS	错误记录状态寄存器
$0x018+64 \times n$	64	ERR< n >ADDR	错误记录地址寄存器
$0x020+64 \times n$	64	ERR< n >MISC0	错误记录杂项寄存器 0
0xE00	64	ERRGSR	错误分组状态寄存器
0xFBC	32	ERRDEVARCH	设备体系结构寄存器
0xFC8	32	ERRIDR	错误记录 ID 寄存器
0x030	32	err_def_set0	错误类型设置寄存器 0
0x034	32	err_def_set1	错误类型设置寄存器 1
0x038	32	err_def_set2	错误类型设置寄存器 2
0x03C	32	err_def_set3	错误类型设置寄存器 3
0x070	32	err_def_set4	错误类型设置寄存器 4
0x074	32	err_def_set5	错误类型设置寄存器 5
0x078	32	err_def_set6	错误类型设置寄存器 6
0x07C	32	err_inject	错误注入寄存器
0x0B0	64	err_sec_set	安全属性配置寄存器

5.29.3.1 错误记录属性寄存器（ERR< n >FR）

该寄存器定义实现了哪些属性，哪些是软件可编程的。可与 ERR< n >CTLR 寄存器对照起来看。

域	位	读写	复位值	描述
IMP	63:32	RO	0	实现决定
RSV	31:20	RO	0	保留
CEO	19:18	RO	2'b00	定义已记录一个可纠错而又检测到第二个可纠错时的行为。 00：不覆盖上个错误的记录信息。如果实现了计数器，就计数；如果没有，或者计数器溢出，则报告错误（ERR< n >STATUS.OF 置 1） 01：如果 ERR< n >STATUS.OF 已经置 1，则保留上个错误的记录信息，否则覆盖；如果实现了计数器，则计数；如果溢出，则报告错误（ERR< n >STATUS.OF 置 1）

域	位	读写	复位值	描述
DUI	17:16	RO	2'b10	是否实现了 DUI 功能。 00: 没有实现, ERR<n>CTLR. DUI 保留; 10: 实现, 通过 ERR<n>CTLR. DUI 控制
RP	15	RO	0	是否实现 repeat 计数器, 决定 ERR_MISC0 的实现方式 0: 没有 repeat 计数器, 只有单个计数器 1: 两个计数器, 包括 repeat 计数器
CEC	14:12	RO	3'b010	可纠错计数器类型。 000: 没有实现 010: 实现 8 位计数器, 在 ERR<n>MISC0[39:32] 100: 实现 16 位计数器, 在 ERR<n>MISC0[47:32]
CFI	11:10	RO	2'b10	是否实现了 CFI 功能。 00: 没有实现, ERR<n>CTLR. CFI 保留 10: 实现, 通过 ERR<n>CTLR. CFI 控制
UE	9:8	RO	2'b00	是否实现 UE 功能。建议除核以外其它模块均不实现。 00: 没有实现, ERR<n>CTLR. UE 保留 01: 实现且该功能总是使能, ERR<n>CTLR. UE 保留 10: 实现, 由 ERR<n>CTLR. UE 控制
FI	7:6	RO	2'b10	是否实现了 FI 功能。 00: 没有实现, ERR<n>CTLR. FI 保留 01: 实现且该功能总是使能, ERR<n>CTLR. FI 保留 10: 实现, 由 ERR<n>CTLR. FI 控制
UI	5:4	RO	2'b10	是否实现了 UI 功能。 00: 没有实现, ERR<n>CTLR. UI 保留 01: 实现且该功能总是使能, ERR<n>CTLR. UI 保留 10: 实现, 由 ERR<n>CTLR. UI 控制
IMP	3:2	RO	0	实现决定
ED	1:0	RO	2'b10	错误报告和记录方式 01: 总是使能, ERR<n>CTLR. ED 保留 10: 由 ERR<n>CTLR. ED 控制

5.29.3.2 错误记录控制寄存器 (ERR<n>CTLR)

该寄存器设置错误记录的相关控制位。

域	位	读写	复位值	描述
RE	63	RW	0x1	用于实现对每个错误记录的错误报告使能
IMP	62:32	RW	0x0	实现决定, 我们实现为只读 (RAZ, 只读且读出为 0)
RSV	31:12	RO	0x0	保留
RSV	11	RO	0x0	保留
DUI	10	RW	0x0	延迟 (Deferred) 错误的错误恢复中断使能
RSV	9	RO	0x0	保留
CFI	8	RW	0x0	可纠错错误处理中断使能, 为 1 时, 如果实现了可纠错计数器, 那么计数器溢出时产生错误处理中断; 否则立刻产生。
RSV	7:5	RO	0x0	保留
UE	4	RW	0x0	带内不可纠错报告使能。使能之后, 当某个响应里面检测到

域	位	读写	复位值	描述
				不可纠错且不可延迟时，将通过带内错误报告机制报错 (in-band error signaling)，也称为 external abort，响应将被丢弃。 目前没有实现，只读 (RAZ)。
FI	3	RW	0x0	错误处理中断使能，为 1 时，所有检测到的不可纠错都会产生错误处理中断；如果 CF1 位没有实现，那么可纠错也会产生错误处理中断
UI	2	RW	0x1	不可纠错恢复中断使能，为 1 时，一旦检测到不可纠错，立刻产生错误恢复中断
IMP	1	RW	0x0	实现决定，我们实现为只读 (RAZ)。
ED	0	RW	0x1	错误报告和记录使能。为 0 时不报告错误，但是否记录错误由实现决定。我们实现为该位只控制错误报告，错误记录不受影响。

5.29.3.3 错误记录状态寄存器 (ERR<n>STATUS)

该寄存器反映错误状态，区分安全与非安全访问。

域	位	读写	复位值	描述
RSV	63:32	RO	0x0	保留
AV	31	R/W1C	0x0	指明 ERR<n>ADDR 记录的地址是否有效 清 0 前必须先将 UE、DE 或 CE 清 0，或者同时清 0
V	30	R/W1C	0x0	指明 ERR<n>STATUS 寄存器是否有效。 0：没有检测到错误，ERR_STATUS 其它位都无效 1：至少检测并记录一个错误 清 0 前必须先将 UE、DE 或 CE 清 0，或者同时清 0
UE	29	R/W1C	0x0	是否检测到不可纠错。 0：没有检测到错误，或者检测到的错误是可纠错或者被延迟了 (Deferred Error) 1：至少检测到一个不可纠错 清 0 前必须先将 OF 清 0，或者同时清 0
ER	28	R/W1C	0x0	错误报告标识。建议除核以外其它模块均不实现。 0：没有报告过带内错误 (external abort) 1：报告过 external abort，原因包括： 实现了 ERR<n>CTLR. UE 且被置 1； 没有实现 ERR<n>CTLR. UE 且总是报告错误 目前实现为只读 (RAZ)。
OF	27	R/W1C	0x0	溢出标志，表示检测到多个错误。
MV	26	R/W1C	0x0	指明 ERR_MISC0 和 ERR_MISC1 是否有效
CE	25:24	R/W1C	0x0	可纠错标志。 00：没有检测到可纠错； 01：至少检测到一个瞬时的可纠错； 10：至少有一个错误被纠正； 11：至少检测到一个持久的可纠错 清 0 前必须先将 OF 清 0，或者同时清 0；并且这两位必须同

域	位	读写	复位值	描述
				时清 0
DE	23	R/W1C	0x0	延迟错误标志。 0: 没有检测到错误; 1: 至少有一个错误没有纠正或者延迟 清 0 前必须先将 0F 清 0, 或者同时清 0
PN	22	R/W1C	0x0	Poison。 0: 由于检测到错误数据而记录了没有纠正或者被延迟的错误; 1: 由于检测到 poison 数据而记录了没有纠正或者被延迟的错误 目前实现为只读 (RAZ)。
UET	21:20	R/W1C	–	不可纠错类型。 00: Uncontainable 错误 01: 不可恢复错误 10: 潜在或者可重启错误 11: 被触发的 (Signaled) 或者可恢复错误 这两位必须同时清 0
RSV	19:16	RO	0x0	保留。
IERR	15:8	RO	–	实现决定的错误编码
SERR	7:0	RO	–	体系结构定义的主错误类型。 0: 没有错误 1: 实现定义的错误 2: 内部存储器数据错, 如片上 SRAM ECC 错 3: 实现定义的引脚, 如 nSEI 4: 断言失效, 如一致性失败 5: 内部数据路径错误, 如 ALU 结果奇偶错 6: 相联存储器数据错, 如 Cache 数据 ECC 错 7: 相联存储器地址或控制信号错, 如 Cache 标识 ECC 错 8: TLB 数据错 9: TLB 地址或者控制信号错 10: 生产者数据错, 如写数据总线上的奇偶错 11: 生产者地址控制信号错, 如地址总线上的奇偶错 12: 外部存储器数据错, 如 SDRAM ECC 错 13: 非法地址, 如访问没有分配的存储空间 14: 非法访问, 如对字寄存器进行字节写操作 15: 非法状态, 例如设备没有准备好 16: 内部数据寄存器错误, 如 SIMD&FP 寄存器奇偶错 17: 内部控制寄存器错误, 如系统寄存器奇偶错 18: 从设备错误响应 19: 外部超时, 如访问其它模块时超时 20: 内部超时, 如自己模块内部超时 21: 主设备收到来自从设备的不支持的延迟错误, 如收到从设备的 poisoned 数据, 而主设备不能够再延迟该错误

5.29.3.4 错误记录地址寄存器 (ERR<n>ADDR)

如果错误关联了一个地址，将地址写入该寄存器。该寄存器区分安全与非安全访问。

域	位	读写	复位值	描述
NS	63	R0	IMP	安全属性，0 表示安全。
SI	62	R0	0x0	指明 NS 域的值与程序员视图是否一致 0：一致，NS 域正确 1：不一致，NS 域可能不正确 目前实现为只读（RAZ）。
AI	61	R0	0x0	地址是否正确 0：PADDR 是一个有效的物理地址 1：PADDR 可能不是一个有效的物理地址 目前实现为只读（RAZ）。
RSV	60:56	R0	0x0	保留
PADDR[55:0]	55:0	R0	0x0	错误相关的物理地址

5.29.3.5 错误记录杂项寄存器 0 (ERR<n>MISC0)

该寄存器记录错误相关信息，可以有 5 种形式。在当前实现中，固定实现第三种，即标准 8 位计数器。该寄存器区分安全与非安全访问。

域	位	读写	复位值	描述
IMP	63:40	R0	0x0	实现决定（RAZ）
OF	39	RW	0x0	溢出标志
CEC	38:32	RW	0x0	可纠错错误的计数器
IMP	31:0	R0	0x0	实现决定（RAZ）

5.29.3.6 错误分组状态寄存器 (ERRGSR)

该寄存器记录所有分组的错误状态信息。每个寄存器对应一个分组，每一位对应组内一个错误记录的 ERR<n>STATUS.V 位。

域	位	读写	复位值	说明
S[63:0]	63:0	R0	0x0	错误状态的位向量

5.29.3.7 设备体系结构寄存器 (ERRDEVARCH)

该寄存器为程序员提供体系结构信息。

域	位	读写	复位值	说明
ARCHITECT	31:21	R0	0x23B	固定含义：JEP106 扩展编码 0x4，ID 编码 0x3B
PRESENT	20	R0	0x1	指明该寄存器是否提供体系结构信息
REVISION	19:16	R0	0x0	RAS 修订版本，0 表示 RAS v1.0
ARCHVER	15:12	R0	0x0	RAS 版本，0 表示 RAS v1
ARCHPART	11:0	R0	0xA00	模块体系结构，0xA00 表示 RAS

5.29.3.8 错误记录 ID 寄存器 (ERRIDR)

该寄存器定义本组内最大的错误记录索引。

域	位	读写	复位值	描述
RSV	31:16	R0	0	保留
NUM	15:0	R0	n	本组内实现的错误记录数目

5.29.3.9 错误注入寄存器

该寄存器用于注入错误, 供调试使用。将要注入错误的错误记录编号写入该寄存器, 指定的错误记录就会出现一个错误。

域	位	读写	复位值	描述
RSV	31:6	R0	0x0	保留
err_inject_num	5:0	W0	0x0	要注入错误的错误记录编号

5.29.3.10 错误类型设置寄存器 0~6

该寄存器用于设置错误的类型。有的时候设计人员可能不确定一种错误的类型, 例如开始将某个错误设置为 corrected 类型, 后来觉得作为 deferred 错误更合适, 可以通过这一组寄存器来修改。每个错误记录使用 3 位的错误类型编码, 见表 5-96。

域	位	读写	复位值	描述
RSV	31	RW	0x0	保留
error type	30:28	RW	0x0	错误记录 (8*寄存器编号+7) 的错误类型编码
RSV	27	RW	0x0	保留
error type	26:24	RW	0x0	错误记录 (8*寄存器编号+6) 的错误类型编码
RSV	23	RW	0x0	保留
error type	22:20	RW	0x0	错误记录 (8*寄存器编号+5) 的错误类型编码
RSV	19	RW	0x0	保留
error type	18:16	RW	0x0	错误记录 (8*寄存器编号+4) 的错误类型编码
RSV	15	RW	0x0	保留
error type	14:12	RW	0x0	错误记录 (8*寄存器编号+3) 的错误类型编码
RSV	11	RW	0x0	保留
error type	10:8	RW	0x0	错误记录 (8*寄存器编号+2) 的错误类型编码
RSV	7	RW	0x0	保留
error type	6:4	RW	0x0	错误记录 (8*寄存器编号+1) 的错误类型编码
RSV	3	RW	0x0	保留
error type	2:0	RW	0x0	错误记录 (8*寄存器编号+0) 的错误类型编码

表 5-82 错误类型编码

错误类型	编码
Uncorrected Uncontainable Error	000

错误类型	编码
Uncorrected Unrecoverable Error	001
Uncorrected Latent or Restartable Error	010
Uncorrected Signaled or Recoverable Error	011
Deferred Error	100
Corrected Transient Error	101
Corrected Error	110
Corrected Persistent Error	111

5.29.3.11 安全属性配置寄存器

该寄存器设置每个错误记录的安全属性，默认值由配置决定。

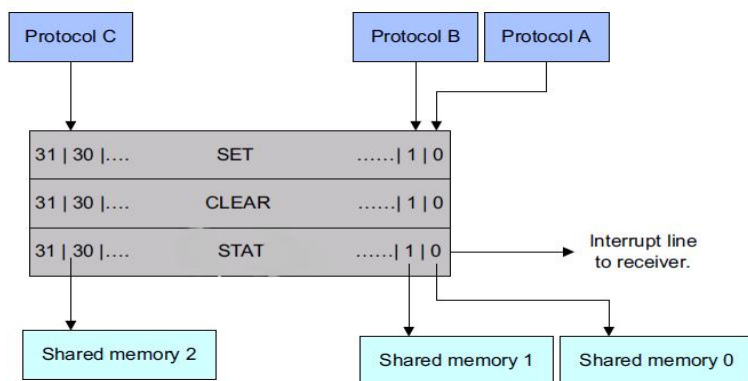
域	位	读写	复位值	描述
RSV	63:RecordNum	R0	0x0	保留
Sec	RecordNum-1:0	RW	IMP	对应错误记录的安全属性，1 为安全，0 为非安全

5.30 MHU

SCMI 消息通过 MHU 作为命令通道和数据通道。

5.30.1 操作说明

MHU 模块物理通道的结构图如下。



假设由 SE 向 AP 发送消息并且采用 STAT[1] 所代表的虚拟通道来传输消息，以此为例详细介绍消息的发送与接收过程。

5.30.1.1 发送消息

1. 消息的发送方即 SE 检测 STAT[1] 是否为低电平，来确定 STAT[1] 所代表的虚拟通道是否空闲，若 STAT[1] 为高电平，则 SE 必须等到 CLEAR 寄存器对 STAT[1]

清零之后才能进行消息的传递。

2. 消息的发送方即 SE 向 STAT[1]代表的虚拟通道所对应的共享内存中写入要传输的消息, 消息的内容和大小由具体的设备决定, SE 必须确保写入的消息对消息的接收方即 AP 是可见的。
3. 消息的发送方即 SE 对 SET[1]写 1, 从而使 STAT[1]置位, 表示 STAT[1]代表的虚拟通道正在传输消息, 与此同时报一个中断信号给消息的接收方即 AP, 通知 AP 接收相应的消息, 需要注意的是中断信号是否发出需要根据 CONFIG[0]来决定是否使能。

5.30.1.2 接收消息

1. 消息的接收方即 AP 接收到物理通道发送过来的中断信号后, 读 STAT 寄存器, 确定消息传输所使用的虚拟通道, STAT 哪一位为高电平, 就是使用的该位所对应的虚拟通道, 此处 STAT[1]为高电平。
2. 消息的接收方即 AP 访问 STAT[1]代表的虚拟通道所对应的共享内存, 将其中的消息数据拷贝一份读出, 该共享内存中的消息数据在通道被释放之前都是有效的, 当该通道再次被占用时, 共享内存中的数据就会被覆盖。
3. 消息的接收方即 AP 对 CLEAR[1]写 1, 从而使 STAT[1]清零, 表示 STAT[1]代表的虚拟通道被释放, SE 所发送的消息被成功接收, 但是并不一定被处理, STAT[1]所代表的虚拟通道已经具备了接收下一条消息的条件。

5.30.2 寄存器列表

表 5-83 MHU 寄存器基地址

名称	基地址
MHU	0x000_32A0_0000

表 5-84 MHU 寄存器列表

寄存器名称	偏移	描述
SE_OS_STAT	0x00	SE_OS 通道状态寄存器
SE_OS_CLEAR	0x10	SE_OS 通道清零寄存器
SE_UBOOT_STAT	0x20	SE_UBOOT 通道状态寄存器
SE_UBOOT_CLEAR	0x30	SE_UBOOT 通道清零寄存器
AP_OS_STAT	0x100	AP_OS 通道状态寄存器
AP_OS_SET	0x108	AP_OS 通道置位寄存器
AP_OS_CLEAR	0x110	AP_OS 通道清零寄存器
AP_UBOOT_STAT	0x120	AP_UBOOT 通道状态寄存器
AP_UBOOT_SET	0x128	AP_UBOOT 通道置位寄存器
AP_UBOOT_CLEAR	0x130	AP_UBOOT 通道清零寄存器

寄存器名称	偏移	描述
AP_OS_CONFIG	0x508	AP_OS 通道中断使能
AP_UBOOT_CONFIG	0x50C	AP_UBOOT 通道中断使能

5.30.3 寄存器说明

5.30.3.1 SE_OS_STAT (0x00)

域	位	读写	复位值	描述
access_type	31	R0	0x0	上一次的读写安全属性 0: 上一次是非安全读写 1: 上一次是安全读写
slot	30:0	R0	0x0	虚通道状态 0: 虚通道空闲 1: 虚通道忙

5.30.3.2 SE_OS_CLEAR (0x10)

域	位	读写	复位值	描述
reserved	31	W0	0x0	保留位
Set	30:0	W0	0x0	发起虚通道通信, 将 stat 寄存器对应位置 1 0: 无效 1: 将 stat 对应位置 1, 发起对应虚通道通信

5.30.3.3 SE_UBOOT_STAT (0x20)

域	位	读写	复位值	描述
access_type	31	R0	0x0	上一次的读写安全属性 0: 上一次是非安全读写 1: 上一次是安全读写
slot	30:0	R0	0x0	虚通道状态 0: 虚通道空闲 1: 虚通道忙

5.30.3.4 SE_UBOOT_CLEAR (0x30)

域	位	读写	复位值	描述
reserved	31	W0	0x0	保留位
Set	30:0	W0	0x0	发起虚通道通信, 将 stat 寄存器对应位置 1 0: 无效 1: 将 stat 对应位置 1, 发起对应虚通道通信

5.30.3.5 AP_OS_STAT (0x100)

域	位	读写	复位值	描述
access_type	31	R0	0x0	上一次的读写安全属性

域	位	读写	复位值	描述
				0: 上一次是非安全读写 1: 上一次是安全读写
slot	30:0	RO	0x0	虚通道状态 0: 虚通道空闲 1: 虚通道忙

5.30.3.6 AP_OS_SET (0x108)

域	位	读写	复位值	描述
reserved	31	WO	0x0	保留位
Set	30:0	WO	0x0	发起虚通道通信, 将 stat 寄存器对应位置 1 0: 无效 1: 将 stat 对应位置 1, 发起对应虚通道通信

5.30.3.7 AP_OS_CLEAR (0x110)

域	位	读写	复位值	描述
reserved	31	WO	0x0	保留位
Set	30:0	WO	0x0	发起虚通道通信, 将 stat 寄存器对应位置 1 0: 无效 1: 将 stat 对应位置 1, 发起对应虚通道通信

5.30.3.8 AP_UBOOT_STAT (0x120)

域	位	读写	复位值	描述
access_type	31	RO	0x0	上一次的读写安全属性 0: 上一次是非安全读写 1: 上一次是安全读写
slot	30:0	RO	0x0	虚通道状态 0: 虚通道空闲 1: 虚通道忙

5.30.3.9 AP_UBOOT_SET (0x128)

域	位	读写	复位值	描述
reserved	31	WO	0x0	保留位
Set	30:0	WO	0x0	发起虚通道通信, 将 stat 寄存器对应位置 1 0: 无效 1: 将 stat 对应位置 1, 发起对应虚通道通信

5.30.3.10 AP_UBOOT_CLEAR (0x130)

域	位	读写	复位值	描述
reserved	31	WO	0x0	保留位

域	位	读写	复位值	描述
Set	30:0	WO	0x0	发起虚通道通信，将 stat 寄存器对应位置 1 0: 无效 1: 将 stat 对应位置 1，发起对应虚通道通信

5.30.3.11 AP_OS_CONFIG (0x508)

域	位	读写	复位值	描述
reserved	31:1	RW	0x0	保留位
en	0	RW	0x0	MHU 通道 0 中断使能信号 0: 使能中断 1: 屏蔽中断

5.30.3.12 AP_UBOOT_CONFIG (0x50C)

域	位	读写	复位值	描述
reserved	31:1	RW	0x0	保留位
en	0	RW	0x0	MHU 通道 0 中断使能信号 0: 使能中断 1: 屏蔽中断

6 中断管理

在 ARMv8 平台上，中断分为三类 PPI、SPI 和 LPI。其中，PPI 和 SPI 的中断号分配由系统的中断管理模块定义，LPI 则由系统软件负责管理与分配。

6.1 PPI 中断

PPI 中断中断号对应的中断含义如下表所示。

表 6-1 PPI 中断 ID 分配表

中断 ID	信号名	含义
16	–	保留
17	–	保留
18	–	保留
19	–	保留
20	–	保留
21	–	保留
22	COMMIRQ	DCC 中断
23	PMUIRQ	PMU 溢出中断
24	CTIIRQ	CTI 中断
25	VCPUMNTIRQ	虚拟 CPU 接口管理中断
26	CNTHPIRQ	虚拟机监控器定时器中断
27	CNTVIRQ	虚拟定时器中断
28	–	保留
29	CNTPSIRQ	安全物理定时器中断
30	CNTPNSIRQ	非安全物理定时器中断
31	–	保留

6.2 SPI 中断

SPI 中断中断号对应的中断含义如下表所示。

表 6-2 SPI 中断 ID 分配表

中断 ID	信号名	含义
32	sys_ras_er_spi	系统错误可恢复中断
33	sys_ras_fh_spi	系统错误可恢复安全中断
34	sys_ras_er_spi_s	系统故障中断
35	sys_ras_fh_spi_s	系统故障安全中断
36	pcie_inta	Pcie inta 中断
37	pcie_intb	Pcie intb 中断
38	pcie_intc	Pcie intc 中断
39	pcie_intd	Pcie intd 中断
40	peu_pcie_misc_int	peu0 杂项中断
41	peu_pcie_msi_int	peu0 消息中断
42	psu_pcie_misc_int	peu1 杂项中断
43	psu_pcie_msi_int	peu1 消息中断
46	usb2_0_int	USB20TG 控制器 0 的中断信号
47	usb2_1_int	USB20TG 控制器 1 的中断信号
48	psu_usb3_0_int	USB3 控制器 0 的中断信号
49	psu_usb3_1_int	USB3 控制器 1 的中断信号
50	usb2_0_wakeup_int	USB20TG 控制器 0 的中断唤醒信号
51	usb2_1_wakeup_int	USB20TG 控制器 1 的中断唤醒信号
60	gsd_gmu_mac0_eth_queue_int[4]	mac0 中断信号
61	gsd_gmu_mac0_eth_queue_int[5]	mac0 中断信号
62	gsd_gmu_mac0_eth_queue_int[6]	mac0 中断信号
63	gsd_gmu_mac0_eth_queue_int[7]	mac0 中断信号
64	-	保留
65	-	保留
66	-	保留
67	-	保留
68	-	保留
69	-	保留
70	sriov_int	PEU 中断
71	-	保留
72	-	保留
73	-	保留
74	psu_sata_int	sata0 中断
75	gsd_sata_int	sata1 中断
76	gsd_dcdp_int	DCDP 中断
77	gsd_i2s0_int	I2S0 的中断信号
78	gsd_i2s1_int	I2S1 的中断信号
79	gsd_i2sdma0_int	I2SDMA0 的中断信号
80	gsd_i2sdma1_int	I2SDMA1 的中断信号
81	gsd_gmu_emac_ethernet_int	eMAC 中断
82	gsd_gmu_mmsl_int	MAC 合并模块中断。
83	gsd_mac_hotplug_int[0]	mac0~3 控制器的热插拔中断

中断 ID	信号名	含义
84	gsd_mac_hotplug_int[1]	mac0~3 控制器的热插拔中断
85	gsd_mac_hotplug_int[2]	mac0~3 控制器的热插拔中断
86	gsd_mac_hotplug_int[3]	mac0~3 控制器的热插拔中断
87	gsd_gmu_mac0_eth_queue_int[0]	mac0 中断信号
88	gsd_gmu_mac0_eth_queue_int[1]	mac0 中断信号
89	gsd_gmu_mac0_eth_queue_int[2]	mac0 中断信号
90	gsd_gmu_mac0_eth_queue_int[3]	mac0 中断信号
91	gsd_gmu_mac1_eth_queue_int[0]	mac1 中断信号
92	gsd_gmu_mac1_eth_queue_int[1]	mac1 中断信号
93	gsd_gmu_mac1_eth_queue_int[2]	mac1 中断信号
94	gsd_gmu_mac1_eth_queue_int[3]	mac1 中断信号
95	-	保留
96	gsd_gmu_mac2_eth_queue_int[0]	mac2 中断信号
97	gsd_gmu_mac2_eth_queue_int[1]	mac2 中断信号
98	gsd_gmu_mac2_eth_queue_int[2]	mac2 中断信号
99	gsd_gmu_mac2_eth_queue_int[3]	mac2 中断信号
100	gsd_gmu_mac3_eth_queue_int[0]	mac3 中断信号
101	gsd_gmu_mac3_eth_queue_int[1]	mac3 中断信号
102	gsd_gmu_mac3_eth_queue_int[2]	mac3 中断信号
103	gsd_gmu_mac3_eth_queue_int[3]	mac3 中断信号
104	lsd_mmcsd_int[0]	MMCS0 的中断信号
105	lsd_mmcsd_int[1]	MMCS1 的中断信号
106	lsd_nfc_int	NANDFlash
107	lsd_dma_int[0]	DMA0
108	lsd_dma_int[1]	DMA1
109	lsd_dma_bdl_int	DDMA1
112	lsd_i2s_int	I2S
113	lsd_can_int[0]	CAN0
114	lsd_can_int[1]	CAN1
115	lsd_uart_int[0]	UART0
116	lsd_uart_int[1]	UART1
117	lsd_uart_int[2]	UART2
118	lsd_uart_int[3]	UART3
123	lsd_smbus_int	SMBUS
124	lsd_mio_int[0]	MI00
125	lsd_mio_int[1]	MI01
126	lsd_mio_int[2]	MI02
127	lsd_mio_int[3]	MI03
128	lsd_mio_int[4]	MI04
129	lsd_mio_int[5]	MI05
130	lsd_mio_int[6]	MI06
131	lsd_mio_int[7]	MI07
132	lsd_mio_int[8]	MI08

中断 ID	信号名	含义
133	lsd_mio_int[9]	MI09
134	lsd_mio_int[10]	MI010
135	lsd_mio_int[11]	MI011
136	lsd_mio_int[12]	MI012
137	lsd_mio_int[13]	MI013
138	lsd_mio_int[14]	MI014
139	lsd_mio_int[15]	MI015
140	lsd_gpio_int[0]	GPIO0
141	lsd_gpio_int[1]	GPIO1
142	lsd_gpio_int[2]	GPIO2
143	lsd_gpio_int[3]	GPIO3
144	lsd_gpio_int[4]	GPIO4
145	lsd_gpio_int[5]	GPIO5
146	lsd_gpio_int[6]	GPIO6
147	lsd_gpio_int[7]	GPIO7
148	lsd_gpio_int[8]	GPIO8
149	lsd_gpio_int[9]	GPIO9
150	lsd_gpio_int[10]	GPIO10
151	lsd_gpio_int[11]	GPIO11
152	lsd_gpio_int[12]	GPIO12
153	lsd_gpio_int[13]	GPIO13
154	lsd_gpio_int[14]	GPIO14
155	lsd_gpio_int[15]	GPIO15
156	lsd_gpio_int[16]	GPIO16
157	lsd_gpio_int[17]	GPIO17
158	lsd_gpio_int[18]	GPIO18
159	lsd_gpio_int[19]	GPIO19
160	lsd_gpio_int[20]	GPIO20
161	lsd_gpio_int[21]	GPIO21
162	lsd_gpio_int[22]	GPIO22
163	lsd_gpio_int[23]	GPIO23
164	lsd_gpio_int[24]	GPIO24
165	lsd_gpio_int[25]	GPIO25
166	lsd_gpio_int[26]	GPIO26
167	lsd_gpio_int[27]	GPIO27
168	lsd_gpio_int[28]	GPIO28
169	lsd_gpio_int[29]	GPIO29
170	lsd_gpio_int[30]	GPIO30
171	lsd_gpio_int[31]	GPIO31
172	lsd_gpio_int[32]	GPIO32
173	lsd_gpio_int[33]	GPIO33
174	lsd_gpio_int[34]	GPIO34
175	lsd_gpio_int[35]	GPIO35

中断 ID	信号名	含义
176	lzd_gpio_int[36]	GPIO36
177	lzd_gpio_int[37]	GPIO37
178	lzd_gpio_int[38]	GPIO38
179	lzd_gpio_int[39]	GPIO39
180	lzd_gpio_int[40]	GPIO40
181	lzd_gpio_int[41]	GPIO41
182	lzd_gpio_int[42]	GPIO42
183	lzd_gpio_int[43]	GPIO43
184	lzd_gpio_int[44]	GPIO44
185	lzd_gpio_int[45]	GPIO45
186	lzd_gpio_int[46]	GPIO46
187	lzd_gpio_int[47]	GPIO47
188	lzd_gpio_int[48]	GPIO48
189	lzd_gpio_int[49]	GPIO49
190	lzd_gpio_int[50]	GPIO50
191	lzd_spim_int[0]	SPIM0
192	lzd_spim_int[1]	SPIM1
193	lzd_spim_int[2]	SPIM2
194	lzd_spim_int[3]	SPIM3
196	lzd_wdt_int[0]	WDT0
197	lzd_wdt_int[1]	WDT1
198	–	保留
199	lzd_jtgm_int	JTAG MASTER
200	–	保留
201	–	保留
202	–	保留
203	–	保留
205	lzd_pwm_int[0]	PWM0
206	lzd_pwm_int[1]	PWM1
207	lzd_pwm_int[2]	PWM2
208	lzd_pwm_int[3]	PWM3
209	lzd_pwm_int[4]	PWM4
210	lzd_pwm_int[5]	PWM5
211	lzd_pwm_int[6]	PWM6
212	lzd_pwm_int[7]	PWM7
221	lzd_keypad_int	KEYPAD
222	–	保留
223	–	保留
224	–	保留
225	–	保留
226	lzd_timer_int[0]	TIMER0
227	lzd_timer_int[1]	TIMER1
228	lzd_timer_int[2]	TIMER2

中断 ID	信号名	含义
229	lzd_timer_int[3]	TIMER3
230	lzd_timer_int[4]	TIMER4
231	lzd_timer_int[5]	TIMER5
232	lzd_timer_int[6]	TIMER6
233	lzd_timer_int[7]	TIMER7
234	lzd_timer_int[8]	TIMER8
235	lzd_timer_int[9]	TIMER9
236	lzd_timer_int[10]	TIMER10
237	lzd_timer_int[11]	TIMER11
238	lzd_timer_int[12]	TIMER12
239	lzd_timer_int[13]	TIMER13
240	lzd_timer_int[14]	TIMER14
241	lzd_timer_int[15]	TIMER15
242	lzd_timer_int[16]	TIMER16
243	lzd_timer_int[17]	TIMER17
244	lzd_timer_int[18]	TIMER18
245	lzd_timer_int[19]	TIMER19
246	lzd_timer_int[20]	TIMER20
247	lzd_timer_int[21]	TIMER21
248	lzd_timer_int[22]	TIMER22
249	lzd_timer_int[23]	TIMER23
250	lzd_timer_int[24]	TIMER24
251	lzd_timer_int[25]	TIMER25
252	lzd_timer_int[26]	TIMER26
253	lzd_timer_int[27]	TIMER27
254	lzd_timer_int[28]	TIMER28
255	lzd_timer_int[29]	TIMER29
256	lzd_timer_int[30]	TIMER30
257	lzd_timer_int[31]	TIMER31
258	lzd_timer_int[32]	TIMER32
259	lzd_timer_int[33]	TIMER33
260	lzd_timer_int[34]	TIMER34
261	lzd_timer_int[35]	TIMER35
262	lzd_timer_int[36]	TIMER36
263	lzd_timer_int[37]	TIMER37
266	dmac0_intr	GDMA 控制器中断

7

功耗管理

飞腾派采用多种低功耗设计技术，包括动态调频、温度控制等。

7.1 动态调频

飞腾派支持 SCMI, SCMI 请参考 arm 官网《system control and management interface v3.0》中的“4.5 Performance domain management protocol”章节。

7.2 温度控制

飞腾派向 SE 发送 SCMI 消息设置超温阈值, SE 检测到超温后向飞腾派发送超温通知。具体细节请参考 arm 官网《system control and management interface v3.0》中的“4.7 Sensor management protocol”章节。

8

术语和缩略语

表 8-1 术语和缩略语

术语	全称	解释
AP	Application Processor	应用处理器
APB	Advanced Peripheral Bus	高级外围总线，AMBA 的慢速总线
AXI	Advanced eXtensible Interface	高级可扩展接口，AMBA 的高速总线
BDL	Buffer Description List	缓存描述符链表
BT	Block Transfer	块传输
CAN	Controller Area Network	控制器局域网
CAN-FD	CAN with Flexible Data-Rate	具有灵活数据速率的控制器局域网
CMD	Command descriptor	命令描述符
CPU	Central Processing Unit	中央处理器
CPPC	Collaborative Processor Performance Control	协同处理器性能控制
DC	Display Controller	显示控制器
DDMA	Device Direct Memory Access	设备 DMA
DDR	Double Data Rate SDRAM	双倍速率同步动态随机存储器
DE	Data Enable	数据使能信号
DMA	Direct Memory Access	直接访问内存
DP	DisplayPort	显示接口
DPCD	DisplayPort Configuration Data	显示端口配置数据
DVFS	Dynamic Voltage and Frequency Scaling	动态电压频率调整
EDID	Extended Display Identification Data	扩展显示标识数据
eMMC	Embedded Multi Media Card	内嵌式多媒体存储卡
GDMA	General Direct Memory Access	通用 DMA
GPIO	General-Purpose Input/Output	通用输入/输出接口
HPB	Host PCIe Bridge	PCIe 主桥
Hsync	Horizontal Synchronization	水平同步
I2C	Inter-Integrated Circuit	集成电路总线
I2S	Inter-IC Sound	集成电路内置音频总线
JTAG	Joint Test Action Group	联合测试工作组
KCS	Keyboard Controller Style	键盘控制器方式

术语	全称	解释
LPI_CTL	Low Power Interface Controller	自定义模块，第二级时钟门控处理逻辑
MHU	Message Handling Unit	消息处理单元
MIO	Multiple Input/Output	多功能输入/输出接口
MMD	Multimedia Display	多媒体显示
MMSL	MAC Merge Sublayer	MAC 融合子层
NAND Flash	NAND Flash	NAND 闪存
ONFI	Open NAND Flash Interface	开放式 NAND 闪存接口
PCIE	Peripheral Component Interconnect Express	高速串行计算机扩展总线标准
PWM	Pulse-Width Modulation	脉冲宽度调制
QSPI	Quad Serial Peripheral Interface	四线式串行外设接口
SATA	Serial Advanced Technology Attachment	串行硬件驱动器接口
SCMI	System Control and Management Interface	系统控制和管理接口协议
SD	Secure Digital Memory Card	安全数字存储
SDIO	Serial Digital Input/Output	串行数字输入输出
SE	Secure Element	安全引擎
SMBus	System Management Bus	系统管理总线
SMMU	System Memory Management Unit	系统内存管理单元
SPI	Serial Peripheral Interface	串行外设接口
SRIOV	Single Root Input/Output Virtualization	单根 IO 虚拟化
UART	Universal Asynchronous Receiver/Transmitter	通用异步收发器
UIB	USB Integration Bridge	自定义模块，APB 地址译码
USB	Universal Serial Bus	通用串行总线
USB OTG	USB On-The-Go	USB2.0 规格的补充标准，允许设备在主机模式与设备模式之间切换，以使设备可以充当 USB 主机或 USB 设备
USB PD	USB Power Delivery	USB 功率传输协议
USBSSP	USB super speed plus	表示 USB3.0 控制器
VPU	Video Processing Unit	视频处理器
Vsync	Vertical Synchronization	垂直同步，场同步
WDT	Watch Dog Timer	看门狗定时器
W1C	Write One to Clear	写 1 清 0，写 0 时无影响，读取时无影响
W1-1	Write one is valid, automatically clear after one clock cycle	写 1 有效，1 拍后自动清零