PHYTIUM 飞腾

FT-2000/4
软件编程手册

（V1.4）

2020 年 6 月

天津飞腾信息技术有限公司

www.phytium.com.cn

当前版本

文件标识	**P-U-2004-003**
当前版本	**1.4**
作　　者	SDG
完成日期	2020.6.09

版本历史

版本	修订时间	修订人	修订章节	修订内容
1.0	2019.8.31			第一个正式发布的版本
1.1	2019.9.23			增加 PCIE End Point 模式描述；修正中断控制器基址
1.2	2020.1.20		5.3；5.8；5.9；5.13	修改 QSPI 章节 Flash 操作流程；删除 RTC 描述；修正 WDT1 地址；修正 GPIO 章节延迟配置描述错误
1.3	2020.3.24		5.4；5.8；5.11 ；5.13；	详细描述芯片引脚复用；修复一些笔误，修改音频 codec 错误描述
1.4	2020.6.08		1；5.13	修改存储控制器描述； 新增引脚驱动能力描述；

目录

1 概述 1

2 地址空间分配 2

2.1 总体地址空间划分 3

2.2 中断控制器内部地址空间划分 3

2.3 SOC 设备地址空间 4

3 计算单元 5

3.1 浮点功能部件 5

3.2 SIMD 功能部件 5

4 存储 6

4.1 Cache 6

4.1.1 L1 Cache 6

4.1.2 L2 Cache 7

4.1.3 L3 Cache 7

4.2 Memory 8

5 外设 9

5.1 PCI Express 9

5.1.1 树形拓扑 9

5.1.2 地址空间划分 9

5.1.3 配置空间寻址 10

5.1.4 中断 10

5.1.5 End Point 11

5.2 千兆位以太网 25

5.2.1 操作说明 25

5.2.2 寄存器 27
5.2.3 GMAC 公共配置寄存器 27
5.2.4 GMAC 控制寄存器 34
5.2.5 GMAC DMA 寄存器 64
5.2.6 GMAC 描述符表 95
5.3 QSPI 117
5.3.1 操作说明 117
5.3.2 寄存器说明 119
5.4 SPI 127
5.4.1 寄存器说明 127
5.5 UART 139
5.5.1 操作说明 139
5.5.2 寄存器说明 141
5.6 LPC 接口 156
5.6.1 操作说明 156
5.6.2 寄存器说明 156
5.7 I2C 接口 159
5.7.1 操作说明 160
5.7.2 寄存器说明 162
5.8 GPIO 接口 194
5.8.1 操作说明 194
5.8.2 寄存器说明 195
5.9 WDT 201
5.9.1 操作说明 201
5.9.2 寄存器说明 203
5.10 SD 控制器 205
5.10.1 操作说明 205
5.10.2 寄存器说明 208
5.11 HD Audio 控制器 218

5.11.1 操作说明 219
5.11.2 寄存器说明 219
5.12 CAN 237
5.12.1 基地址 237
5.12.2 寄存器列表 238
5.12.3 寄存器说明 239
5.13 芯片引脚控制 246
5.13.1 寄存器基地址 246
5.13.2 引脚功能配置 246
5.13.3 通用 IO 引脚时延配置 254
5.13.4 时钟引脚延时配置 260
5.13.5 复用功能表 262
5.13.6 引脚驱动能力 265
6 中断管理 271
6.1 PPI 中断 271
6.2 SPI 中断 271
7 功耗管理 275
7.1 CPU 功耗管理 275
7.2 系统功耗管理 275

图目录

图 1-1 FT-2000/4 结构视图 1
图 5-1 PCIE 树 9
图 5-2 32 位小端格式的同端模式下的 Rx / Tx 描述符 95
图 5-3 发送描述符结构 106
图 5-4 接收描述符结构 111
图 5-5 QSPI 在系统中的地址映射 118

表目录

表 1-1 功能描述 2
表 2-1 FT-2000/4 总体地址空间划分 3
表 2-2 中断控制器内部地址空间划分 3
表 2-3 片内 SOC 设备地址空间划分 4
表 5-1 PCIE 地址空间分配 9
表 5-2 PCIE 配置空间寻址格式 10
表 5-3 PCIE 的 SPI 中断号 10
表 5-4 PCIE 的 DMA 描述符 12
表 5-5 PCIE 寄存器基地址 15
表 5-6 PEU 配置寄存器列表 15
表 5-7 PEU 控制器寄存器列表 19
表 5-8 GMAC 寄存器基地址 27
表 5-9 GMAC 配置寄存器列表 27
表 5-10 GMAC 控制寄存器列表 34
表 5-11 GMAC DMA 寄存器列表 64
表 5-12 QSPI 控制器地址空间 119
表 5-13 QSPI 寄存器列表 119
表 5-14 SPI 基地址 127
表 5-15 UART 寄存器基地址 141
表 5-16 UART 寄存器列表 141
表 5-17 LPC 寄存器基地址 156
表 5-18 LPC 寄存器列表 156
表 5-19 I2C 寄存器基地址 162
表 5-20 I2C 寄存器列表 162
表 5-21 GPIO 控制寄存器基地址 195
表 5-22 GPIO 寄存器列表 195
表 5-23 WDT 基地址 203

表 5-24 WDT 寄存器列表 203
表 5-25 SDC 寄存器基地址 208
表 5-26 SDC 寄存器列表 208
表 5-27 HDA 寄存器基地址 219
表 5-28 HDA 寄存器列表 220
表 5-29 CAN 基地址 237
表 5-30 CAN 寄存器列表 238
表 5-31 引脚复用配置寄存器基址 246
表 5-32 引脚功能配置寄存器 247
表 5-33 通用 IO 引脚延时配置寄存器 254
表 5-34 时钟引脚延时配置寄存器 260
表 5-35 引脚复用功能表 262
表 5-36 引脚与驱动寄存器对应表 266
表 6-1 PPI 中断 ID 分配表 271
表 6-2 SPI 中断 ID 分配表 272

1 概述

FT-2000/4 是一款面向桌面应用的高性能通用 4 核处理器。每 2 个核构成 1 个处理器核簇（Cluster），并共享 L2 Cache。处理器核通过片内高速互联网络及相关控制器与存储系统、I/O 系统相连。

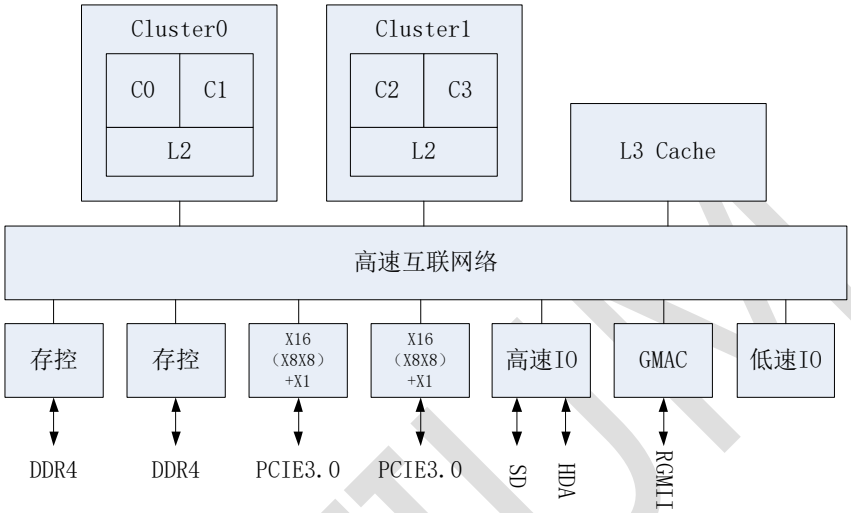

图 1-1 FT-2000/4 结构视图

存储系统包含 Cache 子系统和 DDR，I/O 系统包含 PCIE、高速 IO 子系统、千兆位以太网 GMAC 和低速 IO 子系统。

FT-2000/4 处理器的主要技术特征如下：

- 兼容 ARM v8 64 位指令系统，兼容 32 位指令
- 支持单精度、双精度浮点运算指令
- 支持 ASIMD 处理指令
- 集成 2 个 DDR4 通道，可对 DDR 存储数据进行实时加密
- 集成 34 Lane PCIE3.0 接口：2 个 X16（每个可拆分成 2 个 X8），2 个 X1
- 集成 2 个 GMAC，RGMII 接口，支持 10/100/1000 自适应
- 集成 1 个 SD 卡控制器，兼容 SD 2.0 规范
- 集成 1 个 HDAudio，支持音频输出，可同时支持最多 4 个 Codec
- 集成 SM2、SM3、SM4 模块
- 集成 4 个 UART，1 个 LPC，32 个 GPIO，4 个 I2C，1 个 QSPI，2 个通用 SPI，2 个 WDT，16 个外部中断（和 GPIO 共用 IO）

- 集成温度传感器

功能特征总结如表 1-1 所示。

表 1-1 功能描述

硬件特性		说明
Core	两个 cluster，四个核	FTC663 核
L2Cache	4MB	每两个 core 共享 2M L2 Cache
L3Cache	4MB	分为 8 个 Bank
存储控制器	2 个 72 位 DDR4（其中 8 位 ECC）	支持 DDR3/4 规范，支持 x4(DDR3 不支持)、x8、x16、x32 类型的颗粒，支持 UDIMM、RDIMM、SODIMM、LRDIMM
外设	PCIE3.0	2 个 X16（每个可拆分成 2 个 X8），2 个 X1
	千兆以太网控制器	2 个，RGMII 接口，10/100/1000 自适应
	SD 卡控制器	1 个，兼容 SD 2.0
	HDAudio	1 个，支持 44.1/48/96/192KHz 的音频数据采样率
	UART	4 个串口
	LPC	1 个 Low Pin Count 接口
	GPIO	32 个 GPIO 接口，分为 A、B、C、D 四组，每组 8 位
	I2C	4 个 I2C 接口，支持 Master 或 Slave 模式
	QSPI	1 个 QSPI Flash 接口
	WDT	2 个，一个 Secure，一个 NonSecure
	SPI	2 个通用 SPI master 接口

2 地址空间分配

FT-2000/4 的物理地址宽度为 44bit，地址空间共计 16TB。

2.1 总体地址空间划分

表 2-1 FT-2000/4 总体地址空间划分

地址空间	长度	描述
0x000_00000000~0x000_1FFFFFFF	512MB	QSPI
0x000_20000000~0x000_27FFFFFF	128MB	LPC
0x000_28000000~0x000_2FFFFFFF	128MB	设备寄存器地址，除表中单独列出的寄存器以外，其它所有寄存器
0x000_30000000~0x000_39FFFFFF	160MB	调试寄存器空间
0x000_3A000000~0x000_3AFFFFFF	16MB	片内网络内部寄存器空间
0x000_40000000~0x000_7FFFFFFF	1GB	PCIE 的配置、IO 和 MEM32 空间
0x000_80000000~0x000_FFFFFFFF	2GB	Memory 空间
0x010_00000000~0x01F_FFFFFFFF	64GB	PCIE 的 MEM64 空间
0x020_00000000~0xFFF_FFFFFFFF	15TB768GB	扩展 Memory 空间

2.2 中断控制器内部地址空间划分

中断控制器的寄存器的基址是 0x2990_0000，各部分寄存器的偏移见下表。

表 2-2 中断控制器内部地址空间划分

偏移	大小	描述
0x00_0000~0x00_FFFF	64KB	Distributor 寄存器
0x01_0000~0x01_FFFF	64KB	Distributor 基于消息的 SPI 寄存器
0x02_0000~0x02_FFFF	64KB	ITS 控制寄存器
0x03_0000~0x03_FFFF	64KB	ITS 地址转换寄存器
0x04_0000~0x07_FFFF		保留
0x08_0000~0x08_FFFF	64KB	RD0 控制和物理 LPI 寄存器
0x09_0000~0x09_FFFF	64KB	RD0 SGI 和 PPI 寄存器
0x0A_0000~0x0A_FFFF	64KB	RD1 控制和物理 LPI 寄存器

0x0B_0000~0x0B_FFFF	64KB	RD1 SGI 和 PPI 寄存器
0x0C_0000~0x0C_FFFF	64KB	RD2 控制和物理 LPI 寄存器
0x0D_0000~0x0D_FFFF	64KB	RD2 SGI 和 PPI 寄存器
0x0E_0000~0x0E_FFFF	64KB	RD3 控制和物理 LPI 寄存器
0x0F_0000~0x0F_FFFF	64KB	RD3 SGI 和 PPI 寄存器

2.3 SOC 设备地址空间

表 2-3 片内 SOC 设备地址空间划分

地址空间	长度	描述
0x000_28000000~0x000_28000FFF	4KB	UART0
0x000_28001000~0x000_28001FFF	4KB	UART1
0x000_28002000~0x000_28002FFF	4KB	UART2
0x000_28003000~0x000_28003FFF	4KB	UART3
0x000_28004000~0x000_28004FFF	4KB	GPIO0
0x000_28005000~0x000_28005FFF	4KB	GPIO1
0x000_28006000~0x000_28006FFF	4KB	I2C0
0x000_28007000~0x000_28007FFF	4KB	I2C1
0x000_28008000~0x000_28008FFF	4KB	I2C2
0x000_28009000~0x000_28009FFF	4KB	I2C3
0x000_2800A000~0x000_2800BFFF	8KB	WDT0
0x000_2800C000~0x000_2800CFFF	4KB	SPIM0
0x000_28012000~0x000_28012FFF	4KB	RAS
0x000_28013000~0x000_28013FFF	4KB	SPIM1
0x000_28014000~0x000_28014FFF	4KB	QSPI 寄存器
0x000_28016000~0x000_28017FFF	8KB	WDT1

3 计算单元

FT-2000/4 处理器集成四个自主研发 64 位通用处理核心，兼容 ARMv8-A 处理器架构，可以运行在 32 位工作模式。处理器核采用乱序多发射超标量体系结构，支持动态分支预测和全局历史缓存区。

FT-2000/4 处理器兼容 ARMv8-A 处理器架构，提供如下支持：

1. 支持 AArch32 和 AArch64 两种运行模式；
2. 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；
3. 在每个异常级都支持 Little-Endian 和 Big-Endian 两种字节序；
4. 支持 A32 指令集；
5. 支持 T32 指令集；
6. 支持 A64 指令集；
7. 提供 FPU 浮点计算单元；
8. 提供 SIMD 向量处理单元；
9. 提供加密计算单元；
10. 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，支持浮点精确异常。请参见《ARM Architecture Reference Manual ARMv8 for ARMv8-A architecture profile》[1]。

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，支持浮点精确异常。请参见《ARM Architecture Reference Manual ARMv8 for ARMv8-A architecture profile》[1]。

4 存储

4.1 Cache

FT-2000/4 处理器中，片上存储分为 3 级 Cache 结构。其中一级 Cache 为各个内核私有，每个内核分别有 32KB 的 L1 指令 Cache 和 32KB 的 L1 数据 Cache；二级 Cache 为每个 cluster 内 2 个内核共享，每个 cluster 内 2MB，全片总计 4MB；三级 Cache 为所有内核共享，全片总计 4MB。

4.1.1 L1 Cache

L1 Cache 分为指令 Cache 和数据 Cache。

- L1 指令 Cache 有如下特性：
  1. 容量为 32KB
  2. 64 字节定长 Cache line
  3. 物理地址寻址
  4. 非阻塞
  5. LRU 替换算法
  6. 支持 MBIST 奇偶校验，支持重新取指令
  7. 支持硬件预期
- L1 数据 Cache 有如下特性：
  1. 容量为 32KB
  2. 64 字节定长 Cache line
  3. 物理地址寻址
  4. 非阻塞
  5. LRU 替换算法
  6. 支持 MBIST
  7. 全局同步 monitor
  8. 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：

1. 64 字节定长 Cache line
2. 物理寻址及标识缓存
3. 16 路组相连的 Cache 结构
4. 支持多条流水线并行处理访存请求
5. 包含 L1 数据 Cache
6. 伪随机 Cache 替换策略
7. 支持 128-bit 位宽 CHI 接口
8. 含有 L1 数据 Cache 目录副本
9. 支持 ECC 纠错
10. 支持可选择的硬件预取
11. 支持软件可编程的 RAM 可变响应时间
12. 支持插入寄存器中继以减少路径延迟
13. 支持 MBIST

4.1.3 L3 Cache

L3 Cache 的主要特性包括：

1. 支持 44 位物理地址
2. 采用物理地址作为 index 和 tag
3. Cache line 为 64B
4. 以 cache line 为粒度维护一致性
5. 通过 snoop unit 模块维护一致性
6. L3 及 snoop unit 均为 16 路组相联
7. L3 tag、snoop filter tag 以及 L3 data array 支持 ECC 校验
8. 支持 scrub 以消除单位错误

4.2 Memory

FT-2000/4 处理器集成 2 个 DDR4 内存控制器，其主要功能如下：

- 兼容 JEDEC DDR4 SDRAM 标准协议
- 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
- 多种 RAS 设计

5 外设

5.1 PCI Express

FT-2000/4 内置两个 PCIE 单元（PCI-E Unit，PEU），分别为 PEU0 和 PEU1。每个 PEU 包含 3 个控制器：C0、C1 和 C2。当 PEU 拆分模式为 X16 时，C1 不可见。FT-2000/4 PCIE 接口支持 PCIE 3.0 规范，其特点如下：

- 支持 Root Complex 和 End Point 两种模式；
- 共 34 lane，两路 X16（可拆分为 2 个 X8）和两路 X1；
- 内部集成 DMA 引擎，一读一写两个通道。

5.1.1 树形拓扑

FT-2000/4 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。PCIE 树形结构视图如下所示。

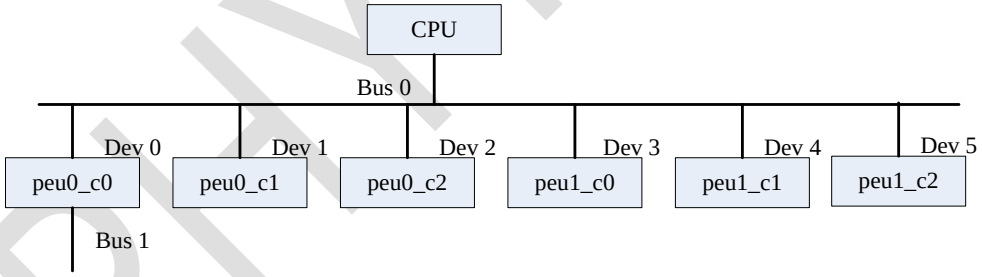

图 5-1 PCIE 树

5.1.2 地址空间划分

表 5-1 PCIE 地址空间分配

地址范围	大小	用途
0x000_40000000~0x000_4FFFFFFF	256MB	配置空间
0x000_50000000~0x000_57FFFFFF	128MB	IO 空间

0x000_58000000~0x000_7FFFFFFF	640MB	MEM32 空间
0x010_00000000~0x01F_FFFFFFFF	64GB	MEM64 空间

5.1.3 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 5-2 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

5.1.4 中断

PCIE 子系统的中断源包括 MSI、INTx、消息以及其它一些中断事件，但不包含错误。错误单独通过错误处理模块报中断。MSI 中断通过中断控制器提供的 MSI 中断方式报上去，所有 INTA 中断共享一个 SPI 中断，INTB 共享一个 SPI 中断，INTC 共享一个 SPI 中断，INTD 共享一个 SPI 中断，消息事件共享一个 SPI，其余杂散中断共享一个 SPI 中断，中断号如下表所示：

表 5-3 PCIE 的 SPI 中断号

SPI 中断号	中断信号	说明
58	peu01_msg_int_spi	接收到一个消息
59	peu01_misc_int_spi	Peu 0/1 内发生特殊事件
60	peu01_inta_spi	Peu 0/1 收到 inta 中断
61	peu01_intb_spi	Peu 0/1 收到 intb 中断
62	peu01_intc_spi	Peu 0/1 收到 intc 中断
63	peu01_intd_spi	Peu 0/1 收到 intd 中断
64	peu01_msi_spi	Peu 0/1 走 spi 通路的 msi 中断

65	peu0_mas_spi	Peu 0 MAS 模块输出的中断
66 -72	RSV 7-bit	
73	peu1_mas_spi	Peu 1 MAS 模块输出的中断

5.1.5 End Point

5.1.5.1 操作说明

PCIE 控制器支持 End Point 模式，End Point 模式的软件初始化流程如下：

- EP 端 BAR 空间的初始化配置：
  - 配置 bar0 对应的本地 memory 地址：例如 BAR0 对应本地 memory 地址 region A(0x23_00000000 – 0x23_003FFFFF)，则写 REG_F0_B0_ATR_L 为 0x00000000，写 REG_F0_B0_ATR_H 为 0x00000023;
  - 配置 bar2 对应的本地 memory 地址：例如 BAR2 对应本地 memory 地址 region B(0x23_00400000 – 0x23_007FFFFF)，则写 REG_F0_B2_ATR_L 为 0x00400000，写 REG_F0_B2_ATR_H 为 0x00000023;
- EP 访问 RC memory 空间的初始化配置：

  EP 通过 region C（0x1100000000-0x11FFFFFFFF)访问 RC memory 空间 region D （0x00000000 – 0xFFFFFFFF)的示例如下：
  - 配置 REG_EP_C0_PREF_BASE_LIMIT 寄存器为 0x00000000；
  - 配置 REG_EP_C0_PREF_BASE_LIMIT_UP32 寄存器为 0x00001211；
  - 配置 REG_OUTBOUND_R0_PATR0 为 0x0000001f；
  - 配置 REG_OUTBOUND_R0_PATR1 为 0x00000000；
  - 配置 REG_OUTBOUND_R0_PHDR0 为 0x00000002；
  - 配置 REG_OUTBOUND_R0_ARBA0 为 0x0000001f；
  - 配置 REG_OUTBOUND_R0_ARBA0 为 0x00000011；
- EP 向 RC 发送 msi 中断的流程如下：
  - 读 REG_MSI_LOW_ADDRESS 获取 msi 地址低 8 位；

- ◆ 读 REG_MSI_HIGH_ADDRESS 获取 msi 地址高 8 位；
- ◆ 读 REG_MSI_DATA 获取事件号；
- ◆ 写事件号到目的地址（msi 地址+region C 基地址）；

■ EP 的 DMA 软件配置流程：

- ◆ 配置 REG_MISC_INT_ENALBE 为 0x7；
- ◆ 配置 REG_DMA_INT_ENABLE 使能中断；
- ◆ 将 dma 描述符基地址写入寄存器 REG_DMA_CH0_SP_L 和 REG_DMA_CH0_SP_H；
- ◆ 写 REG_DMA_CH0_CTRL 寄存器：配置传输方向，并启动 DMA;
- ◆ 当 DMA 传输完成或者出错时，cpu 可以收到 59 号中断，读取 REG_DMA_INT_STATUS 获取中断状态，写 REG_DMA_INT_STATUS 清掉中断状态；
- ◆ DMA 描述符采用链表方式，其格式如下：

表 5-4 PCIE 的 DMA 描述符

偏移	长度(字节)	位	说明
0	8	63:0	源地址
8	4		AXI Ax 通道属性
		31:16	保留位
		15:12	AxREGION[3:0]
		11:8	AxQOS[3:0]
		7	AxLOCK[0]
		6:3	AxCACHE[3:0]
		2:0	AxPROT[2:0]
12	8	63:0	目标地址
20	8		PCIE 报文头属性
		63:45	保留位
		44	TPH Present
		43	保留位

		42:41	TPH ST Hint
		40	保留位
		39:32	TPH Steering Tag
		31:26	保留位
		25:10	Requester ID
		9	Clint req ID
		8:6	保留位
		5:3	PCIE Transfer Class
		2	ID-Based ordering
		1	Relaxed ordering
		0	No Snoop
28	3	23:0	传输长度（0 表示最大长度，即 2^24B）
31	1		控制字
		7:6	保留位
		5	是否该描述符后面还有描述符，即链表模式
		4:3	保留位
		2:1	• 00 - Read and write data for bulk operation • 01 - Prefetch: read-only for scatter/gather operation • 10 - Write data for scatter/gather operation
		0	描述符传输完成后是否产生中断
32	1		AXI 总线状态
		7:3	保留位
		2	Internal data integrity error detected (when generating AXI request)
		1:0	BRESP[1:0] or RRESP[1:0]
33	1		PCIE 总线状态
		7:4	保留位
		3:0	• 0000 - Normal Completion • 0001 - Completion TLP was poisoned

			• 0010 - Request termintaed by a completion with UR, CA or CRS status • 0011 - Request terminated by a completion with no data • 0100 - The current completion has the same tag as an outstanding request but its requester ID, traffic class or Attributes fields do not match • 0101 - The low address bits of the completion TLP header do not match the starting address of the next expected byte. • 0110 - The tag of the current completion does not match that of any outstanding request • 0111 - Request terminated by completion timeout or by an FLR targeted at the function which generated the request • 1000 - Internal error: returning byte count of the completion does not match the expected • 1001 to 1011 - Reserved • 1100 - Internal data integrity error detected (when generating controller request) • 1101 to 1110 - Reserved • 1111 - Multiple error conditions detected
34	1		DMA 通道状态
		7	Buffer not empty
		6	Buffer underflow
		5	Buffer Overflow
		4	描述符是错误的
		3	数据完整性错误
		2	AXI 传输提前结束
		1	PCIE 传输提前结束
		0	传输正常
35	1		保留位

36	8	63:0	链表模式下指向下一个描述符的地址

- RC 向 EP 发送中断的软件配置：
  - EP 端：写 region B 基地址到寄存器 REG_MSI_UP32_ADDR 和 REG_MSI_LOW32_ADDR；
  - EP 端：写 0x1 到 REG_MSI_ENABLE 和 REG_MSI_SPI_ENABLE 使能中断：
  - RC 端：RC 向 BAR2 的基地址写 0 即可向 EP 发送中断,中断号为 64；

5.1.5.2 寄存器

FT-2000/4 PCIE 寄存器基地址划分如下表。

表 5-5 PCIE 寄存器基地址

名称	基地址
PEU0 配置寄存器	0x000_29100000
PEU0 C0 控制寄存器	0x000_29000000
PEU0 C1 控制寄存器	0x000_29010000
PEU0 C2 控制寄存器	0x000_29020000
PEU1 配置寄存器	0x000_29101000
PEU1 C0 控制寄存器	0x000_29030000
PEU1 C1 控制寄存器	0x000_29040000
PEU1 C2 控制寄存器	0x000_29050000

5.1.5.3 PEU 配置寄存器

5.1.5.3.1 配置寄存器列表

表 5-6 PEU 配置寄存器列表

寄存器	偏移	说明
REG_MISC_INT_STATE	0x008	杂散中断状态寄存器
REG_MISC_INT_ENALBE	0x00c	杂散中断是能寄存器

REG_MSI_ENABLE	0x200	MSI 中断使能寄存器
REG_MSI_UP32_ADDR	0x208	匹配 MSI 的高 32 位地址寄存器
REG_MSI_LOW32_ADDR	0x20c	匹配 MSI 的低 32 位地址寄存器
REG_MSI_SPI_ENABLE	0x608	MSI 转 SPI 的使能寄存器
REG_MSI_SPI_DATA	0x60c	MSI 转 SPI 后的 ID 和数据信息寄存器
REG_EP_C0_PREF_BASE_LIMIT	0xa30	EP 模式下 PIO 请求译码到 C0 控制器的可预取空间的 BASE 和 LIMIT 域
REG_EP_C0_PREF_BASE_LIMIT_UP32	0xa34	EP 模式下 PIO 请求译码到 C0 控制器的可预取空间的高 32 位 BASE 和 LIMIT
REG_EP_C1_PREF_BASE_LIMIT	0xa40	EP 模式下 PIO 请求译码到 C1 控制器的可预取空间的 BASE 和 LIMIT 域
REG_EP_C1_PREF_BASE_LIMIT_UP32	0xa44	EP 模式下 PIO 请求译码到 C1 控制器的可预取空间的高 32 位 BASE 和 LIMIT

5.1.5.3.2 REG_MISC_INT_STATE

位	名称	初值	读写	说明
31:19	RSV	0	R	
18	c2_power_state_change	0	R	c2 电源状态变化
17	c2_local_int	0	R	c2 本地中断
16	c2_dma_int	0	R	c2 dma 中断
15:11	RSV	0	R	
10	c1_power_state_change	0	R	c1 电源状态变化
9	c1_local_int	0	R	c1 本地中断
8	c1_dma_int	0	R	c1 dma 中断
7:3	RSV	0	R	
2	c0_power_state_change	0	R	c0 电源状态变化

1	c0_local_int	0	R	c0 本地中断
0	c0_dma_int	0	R	c0 dma 中断

5.1.5.3.3 REG_MISC_INT_ENALBE

位	名称	初值	读写	说明
2	c2_misc_int_en	0	WR	c2 杂散中断使能
1	c1_misc_int_en	0	WR	c1 杂散中断使能
0	c0_misc_int_en	0	WR	c0 杂散中断使能

5.1.5.3.4 REG_MSI_ENABLE

位	名称	初值	读写	说明
0	msi_en	0	WR	MSI 中断使能

5.1.5.3.5 REG_MSI_UP32_ADDR

位	名称	初值	读写	说明
31:0	msi64_hi_addr	0	WR	MSI64 高位地址

5.1.5.3.6 REG_MSI_LOW32_ADDR

位	名称	初值	读写	说明
31:16	msi64_lo_addr	0	WR	MSI64 低位地址
15:0	R	0	R	保留

5.1.5.3.7 REG_MSI_SPI_ENABLE

位	名称	初值	读写	说明
0	msi_spi_en	0	WR	1 表示使能 msi 走 spi 的通路，使能之后 msi 中断 spi 中断

5.1.5.3.8 REG_MSI_SPI_DATA

位	名称	初值	读写	说明
30:16	msi_device_id	0	R	msi 中断的设备 id
15:0	msi_data	0	R	msi 中断的数据

5.1.5.3.9 REG_EP_C0_PREF_BASE_LIMIT

位	名称	初值	读写	说明
31:20	c0_pref_limit[15:4]	0	WR	可预取存储空间上限低位
19:16	保留	0		
15:4	c0_pref_base[15:4]	0	WR	可预取存储空间基址低位
3:0	保留	0		

5.1.5.3.10 REG_EP_C0_PREF_BASE_LIMIT_UP32

位	名称	初值	读写	说明
15:8	c0_pref_limit_up32[7:0]	0	WR	可预取存储空间上限高位
7:0	c0_pref_base_up32[7:0]	0	WR	可预取存储空间基址高位

5.1.5.3.11 REG_EP_C1_PREF_BASE_LIMIT

位	名称	初值	读写	说明
31:20	c1_pref_limit[15:4]	0	WR	可预取存储空间上限低位
19:16	保留	0		
15:4	c1_pref_base[15:4]	0	WR	可预取存储空间基址低位
3:0	保留	0		

5.1.5.3.12 REG_EP_C1_PREF_BASE_LIMIT_UP32

位	名称	初值	读写	说明

15:8	c1_pref_limit_up32[7:0]	0	WR	可预取存储空间上限高位
7:0	c1_pref_base_up32[7:0]	0	WR	可预取存储空间基址高位

5.1.5.4 控制器寄存器

5.1.5.4.1 控制器寄存器列表

表 5-7 PEU 控制器寄存器列表

寄存器	偏移	说明
REG_MSI_LOW_ADDRESS	0x94	MSI 事务的写地址低 32 位
REG_MSI_HIGH_ADDRESS	0x98	MSI 事务的写地址高 32 位
REG_MSI_DATA	0x9c	MSI 事务携带的数据信息
REG_OUTBOUND_R0_PATR0	0x8000	控制器输出方向上 region 0 的转换后地址低 32 位
REG_OUTBOUND_R0_PATR1	0x8004	控制器输出方向上 region 0 的转换后地址高 32 位
REG_OUTBOUND_R0_PHDR0	0x8008	控制器输出方向上 region 0 的转换描述符[31:0]位
REG_OUTBOUND_R0_PHDR1	0x800C	控制器输出方向上 region 0 的转换描述符[63:31]位
REG_OUTBOUND_R0_PHDR2	0x8010	控制器输出方向上 region 0 的转换描述符[95:64]位
REG_OUTBOUND_R0_ARBA0	0x8018	控制器输出方向上 region 0 的转换前的地址低 32 位
REG_OUTBOUND_R0_ARBA1	0x801C	控制器输出方向上 region 0 的转换前的地址高 32 位
REG_F0_B0_ATR_L	0x8840	控制器 FUNC 0 BAR 0 地址转换寄存器低 32 位
REG_F0_B0_ATR_H	0x8844	控制器 FUNC 0 BAR 0 地址转换寄存器高 32 位

REG_F0_B2_ATR_L	0x8850	控制器 FUNC 0 BAR 2 地址转换寄存器低 32 位
REG_F0_B2_ATR_H	0x8854	控制器 FUNC 0 BAR 2 地址转换寄存器高 32 位
REG_DMA_CH0_CTRL	0xC000	DMA channel 0 的控制器寄存器
REG_DMA_CH0_SP_L	0xC004	DMA channel 0 描述符存储的首地址低 32 位寄存器
REG_DMA_CH0_SP_H	0xC008	DMA channel 0 描述符存储的首地址高 32 位寄存器
REG_DMA_CH1_CTRL	0xC014	DMA channel 1 的控制器寄存器
REG_DMA_CH1_SP_L	0xC018	DMA channel 1 描述符存储的首地址低 32 位寄存器
REG_DMA_CH1_SP_H	0xC01C	DMA channel 1 描述符存储的首地址高 32 位寄存器
REG_DMA_INT_STATUS	0xC0A0	DMA 中断状态寄存器
REG_DMA_INT_ENABLE	0xC0A4	DMA 使能寄存器

5.1.5.4.2 REG_MSI_LOW_ADDRESS

位	名称	初值	读写	说明
31:2	Low_addr	0	RW	MSI 事务的写地址低位
1:0	RSV	0	R	保留位，说明地址是字对齐

5.1.5.4.3 REG_MSI_HIGH_ADDRESS

位	名称	初值	读写	说明
31:0	High_addr	0	RW	MSI 事务的写地址高 32 位

5.1.5.4.4 REG_MSI_DATA

位	名称	初值	读写	说明
31:16	RSV	0	R	保留位
15:0	Message_data	0	RW	MSI 事务携带的数据信息

REG_OUTBOUND_R0_PATR0

位	名称	初值	读写	说明
31:8	addr_bits	0	WR	控制器输出方向 region 0 转换后的地址[31:8]位
7:6	RSV	0	R	保留位
5:0	num_bits	0	WR	配置可通过的 AXI 域地址，例如配置为 N，那么 N+1 位地址可通过。

5.1.5.4.5 REG_OUTBOUND_R0_PATR1

位	名称	初值	读写	说明
31:0	addr_bits	0	WR	控制器输出方向 region 0 转换后的地址[63:32]位

5.1.5.4.6 REG_OUTBOUND_R0_PHDR0

位	名称	初值	读写	说明
31:0	descriptor_bits	0	WR	控制器输出方向 region 0 的描述符[31:0]位

5.1.5.4.7 REG_OUTBOUND_R0_PHDR1

位	名称	初值	读写	说明
31:0	descriptor_bits	0	WR	控制器输出方向 region 0 的描述符[63:32]位

5.1.5.4.8 REG_OUTBOUND_R0_PHDR2

位	名称	初值	读写	说明
31:13	RSV	0	R	保留位
12:0	descriptor_bits	0	WR	控制器输出方向 region 0 的描述符[76:64]位

5.1.5.4.9 REG_OUTBOUND_R0_ARBA0

位	名称	初值	读写	说明
31:8	addr_bits	0	WR	控制器输出方向 region 0 转换前的地址[31:8]位
7:6	RSV	0	R	保留位
5:0	lower_mask_bits	0	WR	配置 AXI 域匹配地址时的 mask 位，例如配置为 M，那么 M+1 位地址在匹配时不做比较。

5.1.5.4.10 REG_OUTBOUND_R0_ARBA1

位	名称	初值	读写	说明
31:0	addr_bits	0	WR	控制器输出方向 region 0 转换前的地址[63:32]位

5.1.5.4.11 REG_F0_B0_ATR_L

位	名称	初值	读写	说明
31:0	addr_bits	0	WR	控制器 FUNC 0 BAR 0 地址转换寄存器低 32 位

5.1.5.4.12 REG_F0_B0_ATR_H

位	名称	初值	读写	说明
31:0	addr_bits	0	WR	控制器 FUNC 0 BAR 0 地址转换寄存器高 32 位

5.1.5.4.13 REG_F0_B2_ATR_L

位	名称	初值	读写	说明
31:0	addr_bits	0	WR	控制器 FUNC 0 BAR 2 地址转换寄存器低 32 位

5.1.5.4.14 REG_F0_B2_ATR_H

位	名称	初值	读写	说明
31:0	addr_bits	0	WR	控制器 FUNC 0 BAR 2 地址转换寄存器高 32 位

5.1.5.4.15 REG_DMA_CH0_CTRL

位	名称	初值	读写	说明
31:2	RSV	0	R	保留位
1	ob_not_ib	0	WR	配置 DMA channel 0 是读操作还是写操作
0	Go command bit	0	WR	DMA channel 0 点火位，开始传输

5.1.5.4.16 REG_DMA_CH0_SP_L

位	名称	初值	读写	说明
31:0	start_pointer_lower	0	WR	保存 channel 0 描述符的内存地址低 32 位

5.1.5.4.17 REG_DMA_CH0_SP_H

位	名称	初值	读写	说明
31:0	start_pointer_upper	0	WR	保存 channel 0 描述符的内存地址高 32 位

5.1.5.4.18 REG_DMA_CH1_CTRL

位	名称	初值	读写	说明
31:2	RSV	0	R	保留位

1	ob_not_ib	0	WR	配置 DMA channel 1 是读操作还是写操作
0	Go command bit	0	WR	DMA channel 1 点火位，开始传输

5.1.5.4.19 REG_DMA_CH1_SP_L

位	名称	初值	读写	说明
31:0	start_pointer_lower	0	WR	保存 channel 1 描述符的内存地址低 32 位

5.1.5.4.20 REG_DMA_CH1_SP_H

位	名称	初值	读写	说明
31:0	start_pointer_upper	0	WR	保存 channel 1 描述符的内存地址高 32 位

5.1.5.4.21 REG_DMA_INT_STATUS

位	名称	初值	读写	说明
31:10	RSV	0	R	保留位
9	ch1_err_int	0	R/W1CLR	channel 1 传输出错中断
8	ch0_err_int	0	R/W1CLR	channel 0 传输出错中断
7:2	RSV	0	R	保留位
1	ch1_done_int	0	R/W1CLR	channel 1 传输完成中断
0	ch0_done_int	0	R/W1CLR	channel 0 传输完成中断

5.1.5.4.22 REG_DMA_INT_ENABLE

位	名称	初值	读写	说明
31:10	RSV	0	R	保留位
9	ch1_err_int_en	1	R/W1CLR	使能 channel 1 传输出错后产生中断
8	ch0_err_int_en	1	R/W1CLR	使能 channel 0 传输出错后产生中断
7:2	RSV	0	R	保留位
1	ch1_done_int_en	1	R/W1CLR	使能 channel 1 传输完成后产生中断

0	ch0_done_int_en	1	R/W1CLR	使能 channel 0 传输完成后产生中断

5.2 千兆位以太网

以太网控制器（GMAC）的主要功能是在兼容 IEEE802.3-2005 标准的以太网中发送和接收数据，支持 RGMII 的 PHY 接口。

GMAC 接口特点：

支持速率 1000Mbps/100Mbps/10Mbps

支持 IEEE 802.3-2005 Ethernet MAC，Reduced Gigabit Media Independent Interface (RGMII)

5.2.1 操作说明

■ 初始化 DMA

1. 软件复位，复位内部寄存器和逻辑（Bus_mode_Register-bit[0]）。
2. 等待复位完成（Bus_mode_Register-bit[0]复位操作完成后，这位会被清 0）。
3. 读 AXI 总线的状态位，确认 AXI 的传输完成。（AXI_Status_Register）。
4. 设置下面的域初始化 DMA。（Bus_mode_Register）

   A、 混合突发并且地址对齐(MB AAL)

   B、 固定突发或不定义突发

   C、 突发长度且突发模式值

   D、 描述符长度

   E、 发送、接收 DMA 仲裁调度
5. 设置 AXI 接口选项，如果设置固定的突发长度，选择最大突发长度为 AXI 总线（AXI_Bus_mode_Register）。
6. 设置合适的发送描述符、接收描述符。描述符设置属于 DMA，如果模式设置为 OSF 至少需要 2 个描述符。
7. 确定重用描述符表，在描述符列表中至少有 3 个描述描述符。
8. 把发送、接收描述符地址分别写到发送描述符列表寄存器或接收描述符列表寄存器（Receive_desc_list_Register、Transmit_desc_list_Register）。
9. 初始化操作模式寄存器(Ope_Mode_Register)

A、 发送、接收存储处理

B、 接收且发送域值控制（RTC、TTC）

C、 硬件流程控制使能

D、 设置 MAC 传输层发送、接收 FIFO 的活动流程控制域值和无效活动流程控制域值（RFA、RFD）。

E、 错误帧、不期望的正确帧处理使能。

F、 OSF 模式

10. 清除中断请求。（Status_Register）

11. 使能中断。（Interrupt_enable_register）

12. 读 AXI 总线状态，确定 AXI 传输完成。（AXI_Status_Register）

13. 启动发送及接收。（Ope_Mode_Register bit[1] bit[13]）

■ 初始化 mac

1. 通过 GMII_Addr_Register 寄存器管理外部 phy。

2. 通过 GMII_Data_Register 读取 phy 的状态如 link up、速度、操作模式等。

3. 提供 MAC 地址通过设置（MAC_Addr0_High、MAC_Addr0_Low）寄存器。

4. 如果使能 hash 过滤，设置 hash 表寄存器。

5. 设置 mac 过滤项。

A、 全部接收

B、 混杂模式

C、 Hash 过滤

D、 单投、多投，广播、控制帧过滤设置

6. 设置合适的流程控制

A、 停止时间、停止帧控制时间

B、 发送、接收流程控制位

C、 流程控制忙、反相活动。

7. 编程中断屏蔽寄存器

8. 设置 MAC_cfg_Register 寄存器，设置 Jabber、模式、端口。

9. 设置发送使能和接收使能（MAC_cfg_Register bit[2] bit[3]）

5.2.2 寄存器

5.2.2.1 基地址

FT-2000/4 包含 2 个 GMAC 控制器，分别称为 GMAC0 和 GMAC1。寄存器基地址的划分如下表。

表 5-8 GMAC 寄存器基地址

名称	基地址
GMAC 公共寄存器	0x000_2820B000
GMAC0 私有寄存器	0x000_2820C000
GMAC1 私有寄存器	0x000_28210000

寄存器分为公共寄存器和每个控制器的私有寄存器两大类。每个控制器的私有寄存器又分为控制寄存器和 DMA 寄存器两类。

5.2.3 GMAC 公共配置寄存器

5.2.3.1 GMAC 公共配置寄存器列表

表 5-9 GMAC 配置寄存器列表

寄存器	偏移	说明
REG_GMAC0_AWCACHE	0x0	Gmac0 AXI 总线写地址通道 cache 域
REG_GMAC0_ARCACHE	0x4	Gmac0 AXI 总线读地址通道 cache 域
REG_GMAC0_AWDOMAIN	0x8	Gmac0 AXI 总线写地址通道 domain 域
REG_GMAC0_ARDOMAIN	0xC	Gmac0 AXI 总线读地址通道 domain 域
REG_GMAC0_AWBAR	0x10	Gmac0 AXI 写地址通道 bar 域
REG_GMAC0_ARBAR	0x14	Gmac0 AXI 读地址通道 bar 域
REG_GMAC0_AWSNOOP	0x18	Gmac0 AXI 写地址通道 snoop 域
REG_GMAC0_ARSNOOP	0x1C	Gmac0 AXI 读地址通道 snoop 域
REG_GMAC0_AWPROT	0x20	Gmac0 AXI 写地址通道 prot 域
REG_GMAC0_ARPROT	0x24	Gmac0 AXI 读地址通道 prot 域

REG_GMAC0_AWBASE_ADDR	0x28	Gmac0 AXI 写地址通道 base 域
REG_GMAC0_ARBASE_ADDR	0x2C	Gmac0 AXI 读地址通道 base 域
REG_GMAC1_AWCACHE	0x100	Gmac1 AXI 写地址通道 cache 域
REG_GMAC1_ARCACHE	0x104	Gmac1 AXI 读地址通道 cache 域
REG_GMAC1_AWDOMAIN	0x108	Gmac1 AXI 写地址通道 domain 域
REG_GMAC1_ARDOMAIN	0x10C	Gmac1 AXI 读地址通道 domain 域
REG_GMAC1_AWBAR	0x110	Gmac1 AXI 写地址通道 bar 域
REG_GMAC1_ARBAR	0x114	Gmac1 AXI 读地址通道 bar 域
REG_GMAC1_AWSNOOP	0x118	Gmac1 AXI 写地址通道 snoop 域
REG_GMAC1_ARSNOOP	0x11C	Gmac1 AXI 读地址通道 snoop 域
REG_GMAC1_AWPROT	0x120	Gmac1 AXI 写地址通道 prot 域
REG_GMAC1_ARPROT	0x124	Gmac1 AXI 读地址通道 prot 域
REG_GMAC1_AWBASE_ADDR	0x128	Gmac1 AXI 写地址通道 base 域
REG_GMAC1_ARBASE_ADDR	0x12C	Gmac1 AXI 读地址通道 base 域

5.2.3.2 REG_GMAC0_AWCACHE

域	位	R/W	初值	说明
Reserved	31:4	RO	0x0	保留
gmac0_awcache	3:0	RW	0x2	gmac0 AXI 总线的 awcache 值

5.2.3.3 REG_GMAC0_ARCACHE

域	位	R/W	初值	说明
Reserved	31:4	RO	0x0	保留
gmac0_arcache	3:0	RW	0x2	gmac0 AXI 总线的 arcache 值

5.2.3.4 REG_GMAC0_AWDOMAIN

域	位	R/W	初值	说明
Reserved	31:4	RO	0x0	保留
gmac0_awdomain	1:0	RW	0x1	gmac0 AXI 总线的 awdomain 值

5.2.3.5 REG_GMAC0_ARDOMAIN

域	位	R/W	初值	说明
Reserved	31:4	RO	0x0	保留
gmac0_ardomain	1:0	RW	0x1	gmac0 AXI 总线的 ardomain 值

5.2.3.6 REG_GMAC0_AWBAR

域	位	R/W	初值	说明
Reserved	31:2	RO	0x0	保留
gmac0_awbar	1:0	RW	0x0	gmac0 AXI 总线的 awbar 值

5.2.3.7 REG_GMAC0_ARBAR

域	位	R/W	初值	说明
Reserved	31:2	RO	0x0	保留
gmac0_arbar	1:0	RW	0x0	gmac0 AXI 总线的 arbar 值

5.2.3.8 REG_GMAC0_AWSNOOP

域	位	R/W	初值	说明
Reserved	31:3	RO	0x0	保留
gmac0_awsnoop	2:0	RW	0x0	gmac0 AXI 总线的 awsnoop 值

5.2.3.9 REG_GMAC0_ARSNOOP

域	位	R/W	初值	说明
Reserved	31:4	RO	0x0	保留
gmac0_arsnoop	3:0	RW	0x0	gmac0 AXI 总线的 arsnoop 值

5.2.3.10 REG_GMAC0_AWPROT

域	位	R/W	初值	说明
Reserved	31:3	RO	0x0	保留
gmac0_awprot	2:0	RW	0x2	gmac0 AXI 总线的 awprot 值

5.2.3.11 REG_GMAC0_ARPROT

域	位	R/W	初值	说明
Reserved	31:3	RO	0x0	保留
gmac0_arprot	2:0	RW	0x2	gmac0 AXI 总线的 arprot 值

5.2.3.12 REG_GMAC0_AWBASE_ADDR

域	位	R/W	初值	说明
Reserved	31:4	RO	0x0	保留
gmac0_awbase_addr	3:0	RW	0x0	gmac0 AXI 总线的 aw 通道基地址值

5.2.3.13 REG_GMAC0_ARBASE_ADDR

域	位	R/W	初值	说明
Reserved	31:4	RO	0x0	保留
gmac0_arbase_addr	3:0	RW	0x0	gmac0 AXI 总线的 ar 通道基地址值

5.2.3.14 REG_GMAC1_AWCACHE

域	位	R/W	初值	说明
Reserved	31:4	RO	0x0	保留
gmac1_awcache	3:0	RW	0x2	gmac1 AXI 总线的 awcache 值

5.2.3.15 REG_GMAC1_ARCACHE

域	位	R/W	初值	说明
Reserved	31:4	RO	0x0	保留
gmac1_arcache	3:0	RW	0x2	gmac1 AXI 总线的 arcache 值

5.2.3.16 REG_GMAC1_AWDOMAIN

域	位	R/W	初值	说明
Reserved	31:2	RO	0x0	保留
gmac1_awdomain	1:0	RW	0x1	gmac1 AXI 总线的 awdomain 值

5.2.3.17 REG_GMAC1_ARDOMAIN

域	位	R/W	初值	说明

Reserved	31:2	RO	0x0	保留
gmac1_ardomain	1:0	RW	0x1	gmac1 AXI 总线的 ardomain 值

5.2.3.18 REG_GMAC1_AWBAR

域	位	R/W	初值	说明
Reserved	31:2	RO	0x0	保留
gmac1_awbar	1:0	RW	0x0	gmac1 AXI 总线的 awbar 值

5.2.3.19 REG_GMAC1_ARBAR

域	位	R/W	初值	说明
Reserved	31:2	RO	0x0	保留
gmac1_arbar	1:0	RW	0x0	gmac1 AXI 总线的 arbar 值

5.2.3.20 REG_GMAC1_AWSNOOP

域	位	R/W	初值	说明
Reserved	31:3	RO	0x0	保留
gmac1_awsnoop	2:0	RW	0x0	gmac1 AXI 总线的 awsnoop 值

5.2.3.21 REG_GMAC1_ARSNOOP

域	位	R/W	初值	说明
Reserved	31:4	RO	0x0	保留
gmac1_arsnoop	3:0	RW	0x0	gmac1 AXI 总线的 arsnoop 值

5.2.3.22 REG_GMAC1_AWPROT

域	位	R/W	初值	说明
Reserved	31:3	RO	0x0	保留
gmac1_awprot	2:0	RW	0x2	gmac1 AXI 总线的 awprot 值

5.2.3.23 REG_GMAC1_ARPROT

域	位	R/W	初值	说明
Reserved	31:3	RO	0x0	保留
gmac1_arprot	2:0	RW	0x2	gmac1 AXI 总线的 arprot 值

5.2.3.24 REG_GMAC1_AWBASE_ADDR

域	位	R/W	初值	说明
Reserved	31:4	RO	0x0	保留
gmac1_awbase_addr	3:0	RW	0x0	gmac1 AXI 总线的 aw 通道基地址值

5.2.3.25 REG_GMAC1_ARBASE_ADDR

域	位	R/W	初值	说明
Reserved	31:4	RO	0x0	保留
gmac1_arbase_addr	3:0	RW	0x0	gmac1 AXI 总线的 ar 通道基地址值

5.2.4 GMAC 控制寄存器

5.2.4.1 GMAC 控制寄存器列表

表 5-10 GMAC 控制寄存器列表

寄存器	偏移	描述
MAC 配置寄存器	0x0000	这是 MAC 的操作模式寄存器。
Mac 帧过滤	0x0004	包含帧过滤控件。
哈希表高位寄存器	0x0008	包含多播哈希表的高 32 位。
哈希表低位寄存器	0x000c	包含多播哈希表的低 32 位。
GMII 地址寄存器	0x0010	控制外部 PHY 的管理周期。
GMII 数据寄存器	0x0014	包含要写入 PHY 寄存器或从 PHY 寄存器读取的数据。
流控寄存器	0x0018	控制控制帧的生成。
VLAN 标记寄存器	0x001C	标识 IEEE 802.1Q VLAN 类型帧。
版本寄存器	0x0020	标识 Core 的版本。
调试寄存器	0x0024	给出各种内部块的状态以进行调试。
LPI 控制和状态寄存器	0x0030	控制低功耗空闲（LPI）操作并提供内核的 LPI 状态。
LPI 定时器控制寄存器	0x0034	控制 LPI 状态中的超时值。
中断状态寄存器	0x0038	包含中断状态。
中断屏蔽寄存器	0x003C	包含用于生成中断的掩码。
MAC 地址 0 高寄存器	0x0040	包含第一个 MAC 地址的高 16 位。
MAC 地址 0 低寄存器	0x0044	包含第一个 MAC 地址的低 32 位。
MAC 地址 1 高寄存器	0x0048	包含第二个 MAC 地址的高 16 位。
MAC 地址 1 低寄存器	0x004C	包含第二个 MAC 地址的低 32 位。
寄存器 54	0X00D8	表示通过 SGMII，RGMII 或从 PHY 接收的状态信号 SMII 界面。

5.2.4.2 MAC 配置寄存器

域	位	R/W	初值	说明
Reserved	31:27	RO	0x0	保留
SFTERR	26	RO	0x0	SMII 强制传输错误 设置时,指示 PHY 强制传输正在传输的 SMII 帧中的传输错误。如果在核心配置期间未选择 SMII PHY 端口,则保留此位。
CST	25	RW	0x0	类型帧的 CRC 剥离 设置时,以瑟类型的所有帧的最后 4 个字节 (FCS)(类型字段大于 0x0600) 在将帧转发到应用程序之前被剥离和丢弃。当 MAC 接收器启用了 IP 校验和引擎(类型 1)时,此功能无效。
TC	24	RW	0x0	在 RGMII/SGMII/SMII 中传输配置设置后,此位支持在 RGMII/SMII/SGMII 端口中传输双工模式、链路速度以及向上/向下信息链接到 PHY。重置此位时,不会将此类信息驱动到 PHY。如果在核心配置期间未选择 RGMII、SMII 或 SGMII PHY 端口,则此位是保留的(和 RO)。
WD	23	RW	0x0	看门狗禁用 该位置 1 时，GMAC 禁止接收器上的看门狗定时器，并可接收最多 16384 字节的帧。当该位复位时，GMAC 允许接收的帧不超过 2048

				字节（如果 JE 设置为高，则为 10240）并切断之后接收的任何字节。
JD	22	RW	0x0	Jabber 禁用 当该位置 1 时，GMAC 禁用发送器上的 jabber 定时器，并可以传输最多 16384 字节的帧。 当该位复位时，如果应用程序在传输过程中发出超过 2048 字节的数据（如果 JE 设置为高电平则为 10240），则 GMAC 会切断发送器。
BE	21	RW	0x0	帧突发启用 当该位置 1 时，GMAC 允许在 GMII 半双工模式下传输期间帧突发。 该位仅保留（和 RO）10/100 Mbps 或仅支持全双工配置。
JE	20	RW	0x0	Jumbo 帧启用 当该位置 1 时，GMAC 允许 Jumbo 帧为 9018 字节（VLAN 为 9022 字节） 标记帧）而不报告接收帧状态中的巨大帧错误。
IFG	19:17	RW	0x0	帧内间隔 000 96bit times 001 88bit times 010 80bit times … 111 40bit times 在半双工模式下，最小 IFG 只能配

				置为 64 位时间（IFG = 100）。 不考虑较低的值。 在 1000-Mbps 模式下，支持的最小 IFG 在 GMAC-CORE 配置中为 64 位时（及以上），在其他系统配置中为 80 位（及以上）。
DCRS	16	RW	0x0	传输过程中禁用载波侦听 设置为高电平时，该位使 MAC 发送器在半双工模式下的帧发送期间忽略（G）MII CRS 信号。 该请求导致在这种传输期间由于载波丢失或无载波而不产生错误。 当该位为低时，MAC 发送器由于载波侦听而产生这样的错误，甚至可以中止传输。 在核心配置期间选择核心进行全双工操作时，此位保留（和 RO）。
PS	15	RW	0x0	端口选择 0 GMII（1000Mbps） 1 MII(10/100Mbps) 该位是只读的，具有 10/100 Mbps 的适当值（始终为 1）或 仅 1000 Mbps（始终为 0）配置，以及默认 10/100/1000 Mbps 配置中的 R_W
FES	14	RW	0x0	速度 1 10Mbps 0 100Mbps 该位默认保留（RO），仅在配置

				期间启用 RMII/SMII/RGMII/SGMII/RevMII 时启用。
DO	13	RW	0x0	不能自接收 设置为 1 时，在半双工模式下 gmii_txen_o 占用时，GMAC 就不接收帧。 设置为 0 时，当发送时 GMAC 接收所有包。 对于全双工模式，该位保留且只读
LM	12	RW	0x0	环回模式 当该位置 1 时，GMAC 在 GMII / MII 上以环回模式工作。 （G）MII 接收时钟输入（clk_rx_i）是环回正常工作所必需的，因为发送时钟不在内部环回。
DM	11	RW	0x0	双工模式 该位置 1 时，GMAC 工作在全双工模式，可以同时发送和接收。该位为 RO，在全双工配置中默认值为 1'b1。
IPC	10	RW	0x0	校验和卸载 当该位置 1 时，GMAC 计算所有接收的以太网帧有效载荷的一个补码和的 16 位补码。 它还检查 IPv4 Header 校验和（假设接收到的以太网帧的字节 25-26 或 29-30（VLAN 标记））对于接收的帧是否正确，并在接收状态字中给出状

				态。 GMAC 核心还附加为 IP 报头数据报有效负载计算的 16 位校验和（IPv4 报头之后的字节），并将其附加到传输到应用程序的以太网帧（当取消选择类型 2 COE 时）。 该位复位时，该功能被禁止。
DR	9	RW	0x0	禁用重试 该位置 1 时，GMAC 仅尝试 1 次传输。当 GMII / MII 发生冲突时，GMAC 忽略当前帧传输并报告帧中止在发送帧状态中具有过多的冲突错误。 当该位复位时，GMAC 会根据 BL 的设置（位[6：5]）尝试重试。 该位仅适用于半双工模式，并且在全双工配置中保留（RO 为默认值）。
LUD	8	RW	0x0	链接 up/down 指示在 RGMII / SGMII / SMII 接口中传输配置期间链路是启动还是关闭: 0 Link Down 1 Link Up 该位保留（RO 为默认值），并在配置期间启用 RGMII / SGMII / SMII 时启用。
ACS	7	RW	0x0	自动 PAD/ CRC 剥线 当该位复位时，GMAC 会根据 BL 的设置（位[6：5]）尝试重试。 该

				位仅适用于半双工模式，并且在全双工配置中保留（只读为默认值）。当该位复位时，GMAC 将所有传入帧传递给未修改的主机。
BL	6:5	RW	0x0	后退限制 后退限制确定了在冲突后重试期间重新安排传输尝试之前 GMAC 等待的时隙延迟的随机整数（r）（1000 Mbps 的 4096 位时间和 10/100 Mbps 的 512 位时间）。 该位仅适用于半双工模式，仅在全双工配置中保留（RO）。 00 k = min (n, 10) 01 k = min (n, 8) 10 k = min (n, 4) 11 k = min (n, 1) 其中 n =重传尝试。 随机整数 r 取该范围内的值 $0 \le r < 2k$
DC	4	RW	0x0	延期检查 设置此位后，GMAC 中将启用延迟检查功能。 当发送状态机在 10/100-Mbps 模式下延迟超过 24288 位时，GMAC 发出帧中止状态以及在发送帧状态中设置的过度延迟错误位。 如果 Core 配置为 1000 Mbps 操作，或者在 10/100-Mbps 模式下启用 Jumbo 帧模式，则延迟阈值为 155680 位。当发送器准备好发送时，延迟开

				始，但由于 GMII / MII 上的活动 CRS（载波侦听）信号而被阻止。延迟时间不是累积的。
TE	3	RW	0x0	发送器启用 当该位置 1 时，GMAC 的发送状态机被启用以在 GMII / MII 上传输。 当该位复位时，GMAC 发送状态机在完成当前帧的发送之后被禁止，并且不再发送任何帧。
RE	2	RW	0x0	接收器启用 当该位置位时，GMAC 的接收器状态机被启用以接收来自 GMII / MII 的帧。 当该位复位时，GMAC 接收状态机在完成当前帧的接收后被禁止，并且不再从 GMII / MII 接收任何帧。
Reserved	1:0	RO	0x0	保留

5.2.4.3 Mac 帧过滤

域	位	R/W	初值	说明
RA	31	RW	0x0	全部接收 当该位置 1 时，GMAC 接收器模块将接收到的所有帧传递给应用程序，而不管它们是否通过地址过滤器。 源地址/目的地址过滤的结果在接收状态字中的相应位中更新（通过或失败）。 当该位复位时，接收模块仅将那些通过源地

				址/ 目的地址过滤器的帧传递给应用程序。
Reserved	30:11	RO	0x0	保留
HPF	10	RW	0x0	hash 或 perfect 过滤器 置位时，如果该位匹配 HMC 或 HUC 位设置的完美滤波或散列滤波，则该位配置地址滤波器以传递帧。当为低电平且 HUC / HMC 位置 1 时，仅当帧与哈希滤波器匹配时才传递帧。 如果在核心配置期间未选择哈希过滤器，则保留此位只读。
SAF	9	RW	0x0	源地址过滤使能 GMAC 内核将接收帧的源地址字段与启用的源地址寄存器中编程的值进行比较。 如果比较匹配，则接收状态的源地址匹配位设置为高。 当该位设置为高电平且源地址滤波器发生故障时，GMAC 将丢帧。 当该位复位时，GMAC Core 会根据源地址比较将接收到的帧转发给应用程序，并使用接收状态的更新源地址匹配位。
SAIF	8	RW	0x0	源地址反相过滤 该位置 1 时，地址校验块以反向滤波模式工作，进行 SA 地址比较。源地址与源地址寄存器匹配的帧被标记为源地址过滤器失败。

				当该位复位时，源地址与源地址寄存器不匹配的帧被标记为源地址过滤器失败。
PCF	7:6	RW	0x0	通过控制帧 这些位控制所有控制帧的转发（包括单播和多播暂停帧）。 请注意，暂停控制帧的处理取决于寄存器 6（流控制寄存器）的全双工模式和第 2 位（RFE）。 00 GMAC 过滤所有控制帧到达应用程序。 01 GMAC 将除暂停控制帧之外的所有控制帧转发到应用程序，即使它们未通过地址过滤器也是如此。 10 即使 GM 地址失败，GMAC 也会将所有控制帧转发给应用程序过滤。 11 GMAC 转发通过地址过滤器的控制帧。
DBF	5	RW	0x0	禁用广播帧 设置此位后，AFM 模块将过滤所有传入的广播帧。 此外，它会覆盖所有其他过滤器设置。 当该位复位时，AFM 模块传递所有接收的广播帧。
PM	4	RW	0x0	通过所有多播 置位时，该位指示所有接收到的具有多播目的地址的帧（目标地址字段中的第一位为"1"）被传递。

				复位时，组播帧的过滤取决于HMC 位。
DAIF	3	RW	0x0	目的地址反向过滤 当该位置 1 时，地址校验块以反向滤波模式工作，用于单播和多播帧的目的地址比较。 复位时，执行帧的正常滤波。
HMC	2	RW	0x0	哈希多播 设置时，MAC 根据哈希表对接收到的组播帧进行目的地址过滤。 复位时，MAC 对多播帧执行完美的目标地址过滤，即将目的地址字段与目的地址寄存器中编程的值进行比较。
HUC	1	RW	0x0	哈希单播 设置时，MAC 根据哈希表执行单播帧的目的地址过滤。 复位时，MAC 对单播帧执行完美的目标地址过滤，即将 DA 字段与 DA 寄存器中编程的值进行比较。
PR	0	RW	0x0	哈希单播 设置时，MAC 根据哈希表执行单播帧的目的地址过滤。 复位时，MAC 对单播帧执行完美的目标地址过滤，即将目的地址字段与目的地址寄存器中编程的值进行比较。

5.2.4.4 哈希表高位寄存器

域	位	R/W	初值	说明
HTH	31:0	RW	0x0	该字段包含 Hash 表的高 32 位。

5.2.4.5 哈希表低位寄存器

域	位	R/W	初值	说明
HTL	31:0	RW	0x0	该字段包含 Hash 表的低 32 位。

5.2.4.6 GMII 地址寄存器

域	位	R/W	初值	说明
Reserved	31:16	RO	0x0	保留
PA	15:11	RW	0x0	物理层地址 该字段告知正在访问 32 个可能的 PHY 设备中的哪一个。 对于 RevMII，此字段提供 RevMII 模块的 PHY 地址。
GR	10:6	RW	0x0	GMII 寄存器 这些位在所选 PHY 器件中选择所需的 GMII 寄存器。 对于 RevMII，这些位在 RevMII 寄存器集中选择所需的 CSR 寄存器。
CR	5:2	RW	0x0	CSR 时钟范围 CSR 时钟范围选择根据设计中使用的 clk_csr_i 频率确定 MDC 时钟的频率。 建议的 clk_csr_i 频率范

				围适用于下面的每个值（当位[5] = 0 时），确保 MDC 时钟大约在 1.0 MHz - 2.5 MHz 的频率范围之间。 0000 R_W 选择 clk_csr_i MDC 时钟 0000 60-100MHz clk_csr_i 0001 100-150 MHz clk_csr_i/62 0010 20-35 MHz clk_csr_i/16 0011 35-60 MHz clk_csr_i/26 0100 150-250 MHz clk_csr_i/102 0101 250-300 MHz clk_csr_i/124 0110, 0111 保留
GW	1	RW	0x0	GMII 写 置 1 时，该位告诉 PHY / RevMII 这是使用 GMII 数据寄存器的写操作。 如果未设置该位，则为读操作，将数据放入 GMII 数据寄存器中。
GB	0	R_W S_SC	0x0	GMII 忙 在写入寄存器 4 和寄存器 5 之前，该位应读为逻辑 0. 在 PHY / RevMII 寄存器访问期间，软件将该位设置为 1'b1 以指示正在进行读或写访问。

				在 GMAC 清除该位之前，寄存器 5 无效。 因此，寄存器 5（GMII 数据）应保持有效，直到 GMAC 在 PHY 写操作期间清除该位为止。 类似地，对于读操作，寄存器 5 的内容在该位清零之前无效。后续的读/写操作应该只在前一个操作完成后才会发生。

5.2.4.7 GMII 数据寄存器

域	位	R/W	初值	说明
Reserved	31:16	RO	0x0	保留
GD	15:0	RW	0x0	GMII 数据 它包含管理读操作后从 PHY / RevMII 读取的 16 位数据值或管理写操作之前写入 PHY / RevMII 的 16 位数据值。

5.2.4.8 流控寄存器

域	位	R/W	初值	说明
PT	31:16	RW	0x0	暂停时间 该字段保存要在传输控制帧中的"暂停时间"字段中使用的值。 如果暂停时间位配置为与（G）MII 时钟域双重同步，则应仅在目标时钟域中至少 4 个时钟周期后才能对该寄存器执行连续写操作。

Reserved	15:8	RO	0x0	保留
DZPQ	7	RW	0x0	置位时，该位禁止在 FIFO 层（MTL 或外部边带流控制信号 sbd_flowctrl_i / mti_flowctrl_i）取消断言流量控制信号时自动生成零暂停控制帧。 当该位复位时，启用自动零暂停帧生成的正常操作。
Reserved	6	RO	0x0	保留
PLT	5:4	RW	0x0	暂停低阈值 该字段配置暂停定时器的阈值，在该阈值处检查输入流控制信号 mti_flowctrl_i（或 sbd_flowctrl_i）以自动重传暂停帧。 阈值应始终小于位 3116 中配置的暂停时间。例如，如果 PT = 100H（256 个时隙），并且 PLT = 01，那么如果在发送第一个暂停帧之后的 228（256-28）个时隙处断言 mti_flowctrl_i 信号，则自动发送第二个暂停帧。 选择 阈值 00 暂停时间减去 4 个时段 01 暂停时间减去 28 个时段 10 暂停时间减去 144 个时段 11 暂停时间减去 256 个时段
UP	3	RW	0x0	单播暂停帧检测 当该位置 1 时，GMAC 会检测到具有 MAC 地址 0 高位寄存器和

				MAC 地址 0 低位寄存器中指定的站点单播地址的暂停帧，以及检测到具有唯一多播地址的暂停帧。当该位复位时，GMAC 仅检测具有 802.3x 标准中指定的唯一多播地址的暂停帧。
RFE	2	RW	0x0	接收流控制启用 当该位置 1 时，GMAC 对接收到的暂停帧进行解码，并在指定的（暂停时间）时间内禁用其发送器。 该位复位时，禁用暂停帧的解码功能。
TFE	1	RW	0x0	传输流控制启用 在全双工模式下，当该位置 1 时，GMAC 使流控制操作能够发送暂停帧。 当该位复位时，GMAC 中的流控制操作被禁用，GMAC 不发送任何暂停帧。 在半双工模式下，当该位置 1 时，GMAC 启用背压操作。 该位复位时，禁用背压功能。
FCB/BPA	0	R_WS_SC (FCB) RW (BPA)	0x0	流量控制忙/背压激活 如果 TFE 位置 1，该位在全双工模式下启动暂停控制帧，并在半双工模式下激活背压功能。 在全双工模式下，在写入流控制寄存器之前，该位应读为 1'b0。 要启动暂停控制帧，应用程序必须将此位设置为 1'b1。 在控制帧的传

				输期间，该位继续被设置为表示帧传输正在进行中。 完成暂停控制帧传输后，GMAC 将该位复位为 1'b0。 在清除该位之前，不应写入流控制寄存器。

5.2.4.9 VLAN 标记寄存器

域	位	R/W	初值	说明
Reserved	31:17	RO	0x0	保留
ETV	16	RW	0x0	启用 12 位 VLAN 标记比较 设置此位时，将使用 12 位 VLAN 标识符而不是完整的 16 位 VLAN 标记进行比较和过滤。 将 VLAN 标记的比特 11：0 与接收的 VLAN 标记帧中的相应字段进行比较。 当该位复位时，接收的 VLAN 帧的第 15 和第 16 字节的所有 16 位用于比较.
VL	15:0	RW	0x0	接收帧的 VLAN 标记标识符 它包含用于标识 VLAN 帧的 802.1Q VLAN 标记，并与 VLAN 帧接收的帧的第 15 和第 16 字节进行比较。 位 1513 是用户优先级，位 12 是规范格式指示符（CFI），位 11：0 是 VLAN 标记的 VLAN 标识符（VID）字段。 当 ETV 位置 1 时，仅使用 VID（位 11：0）进行比较。

5.2.4.10 版本寄存器

域	位	R/W	初值	说明
UDV	15:8	RO	xx	用户定义版本号
SDV	7：0	RO	0x36	硬件定议版本号

5.2.4.11 调试寄存器

域	位	R/W	初值	说明
Reserved	31:26	RO	0x0	保留
	25	RO	0x0	当为高电平时，表示 MTL TxStatus FIFO 已满，因此 MTL 无法再接受任何帧传输。 该位保留在 GMAC-AHB 和 GMAC-DMA 配置中。
	24	RO	0x0	当为高电平时，表示 MTL TxFIFO 不为空并且有一些数据需要传输。
Reserved	23	RO	0x0	保留
	22	RO	0x0	为高电平时，表示 MTL TxFIFO 写入控制器处于活动状态并将数据传输到 TxFIFO。
	21:20	RO	0x0	
	19	RO	0x0	当为高电平时，表示 MAC 发送器处于暂停状态（仅限全双工），因此不会调度任何帧进行传输
	18:17	RO	0x0	这表示 MAC 发送帧控制器模块的状态:

				00 空闲状态 01 等待前一帧或 IFG /退避时段的状态结束 10 生成和发送暂停控制帧（全双工模式） 11 传输输入帧以进行传输
	16	RO	0x0	当为高时，表示 MAC GMII / MII 传输协议引擎正在主动传输数据而不处于 IDLE 状态.t6ytfrdesz
Reserved	15:10	RO	0x0	保留
	9:8	RO	0x0	给的 RXFIFO 状态级: 00 RxFIFO 空 01 RxFIFO 填充水平低于流量控制去激活阈值 10 RxFIFO 填充级别高于流量控制激活阈值 11 RxFIFO 满
Reserved	7	RO	0x0	保留
	6:5	RO	0x0	它给出了 RxFIFO 读控制器的状态: 00 空闲状态 01 读取帧数据 10 读框状态（或时间戳） 11 刷新帧数据和状态
	4	RO	0x0	为高电平时，表示 MTL RxFIFO 写控制器处于活动状态，并将接收到的帧传送到 FIFO。

	3	RO	0x0	保留
	2:1	RO	0x0	当为高电平时，它分别表示 MAC 接收帧控制器模块的小型 FIFO 读写控制器的活动状态。
	0	RO	0x0	当为高时，表示 MAC GMII / MII 接收协议引擎正在主动接收数据而不处于 IDLE 状态。

5.2.4.12 LPI 控制和状态寄存器

域	位	R/W	初值	说明
Reserved	31:20	RO	0x0	保留
LPITXA	19	RW	0x0	LPI 发送自动化 该位控制 MAC 在发送侧进入或退出 LPI 模式时的行为。 该位在 GMAC-CORE 配置中不起作用，其中 Tx 时钟门控在 LPI 模式期间完成。 如果 LPITXA 和 LPIEN 位设置为 1，则只有在所有未完成的帧（在核心中）和待处理的帧（在应用程序接口中）已经发送之后，MAC 才进入 LPI 模式。 当应用程序发送任何帧进行传输时，MAC 将退出 LPI 模式。 此外，GMAC 内核在退出 LPI 状态时自动清除 LPIEN 位。 当该位为 0 时，LPIEN 位直接控制 MAC 进入或退出 LPI 模式时的行

				为。
PLSEN	18	RW	0x0	物理链路状态使能 该位使能在 RGMII，SGMII 或 SMII 接收路径上接收的链路状态（见寄存器 54（SGMII/RGMII/SMII 状态寄存器））用于激活 LPI LS TIMER。 置 1 时，MAC 使用寄存器 54 的链路状态位和 LPI LS 定时器触发的 PLS 位。 清零后，MAC 忽略寄存器 54 的链路状态位。 如果您未选择 RGMII，SGMII 或 SMII PHY，则该位为 RO 并保留
PLS	17	RW	0x0	PHY 链路状态 该位指示 PHY 的链路状态。 仅当链路状态为（好）时，MAC 发送器才会断言 LPI 模式，至少在 LPI LS TIMER 指示的时间内。 设置后，链接被认为是正常的（向上），重置时，链接被认为是关闭的。
LPIEN	16	R_W_SC	0x0	LPI 使能 置位时，该位指示 MAC 发送器进入 LPI 状态。 复位时，该位指示 MAC 退出 LPI 状态并恢复正常传输。 当 LPITXA 位置 1 且 MAC 由于新数据包到达而退出 LPI 状态时，该位清零。

Reserved	15:10	RO	0x0	保留
RLPIST	9	RO	0x0	接收 LPI 状态 置位时，该位指示 MAC 正在 GMII / MII 接口上接收 LPI 模式。
TLPIST	8	RO	0x0	发送 LPI 状态 置位时，该位表示 MAC 正在 GMII / MII 接口上发送 LPI 模式。
Reserved	7:4	RO	0x0	保留
RLPIEX	3	R_SS_RC	0x0	接收 LPI 退出 置位时，该位指示 MAC 接收器已停止接收 GMII / MII 上的 LPI 模式，已退出 LPI 状态，并已恢复正常接收。 通过读入该寄存器来清除该位。
RLPIEN	2	R_SS_RC	0x0	接收 LPI 进入 置位时，该位指示 MAC 接收器已接收到 LPI 模式并进入 LPI 状态。通过读入该寄存器来清除该位。
TLPIEX	1	R_SS_RC	0x0	发送 LPI 退出 置位时，该位指示用户清零 LPIEN 位且 LPI TW 定时器到期后 MAC 发送器已退出 LPI 状态。 通过读入该寄存器来清除该位。
TLPIEN	0	R_SS_RC	0x0	发送 LPI 进入 置 1 时，该位表示由于 LPIEN 位的设置，MAC 发送器已进入 LPI 状态。 通过读入该寄存器来清除该位。

5.2.4.13 LPI 定时器控制寄存器

域	位	R/W	初值	说明
Reserved	31:16	RO	0x0	保留
LIT	25:16	RW	0x3E8	这些位指定在将 LPI 模式发送到 PHY 之前，来自 PHY 的链路状态应该是（可以）的最小时间（以毫秒为单位）。 即使 LPIEN 位置 1，MAC 也不发送 LPI 模式，除非 LPI LS 定时器达到编程的终端计数。 LIT 的默认值为 1000（1 秒），如 IEEE 标准中所定义。
TWT	15:0	RW	0x0	这些位指定 MAC 在停止向 PHY 发送 LPI 模式之后以及在恢复正常传输之前等待的最小时间（以微秒为单位）。 TLPIEX 状态位在该定时器到期后置位。

5.2.4.14 中断状态寄存器

域	位	R/W	初值	说明
Reserved	31:11	RO	0x0	保留
LPIIS	10	RO	0x0	LPI 中断状态 启用节能以太网功能后，此位将设置为 MAC 发送器或接收器中的任何 LPI 状态进入或退出。 读取寄存器 12 的字节 0（LPI 控制和状态寄存器）时，该位清零。 在所有其他模式中，该位保留。

TIS	9	RO/R_SS_RC	0x0	时间戳中断状态 启用高级时间戳功能时，如果满足以下任一条件，则设置此位： •系统时间值等于或超过目标时间中指定的值 高低寄存器 •秒寄存器中有溢出 •声明辅助快照触发器
Reserved	8	RO	0x0	保留
MMCRCOIS	7	RO	0x0	接收校验和卸载中断状态 只要在 MMC 接收校验和卸载中断寄存器中产生中断，该位就会置为高电平。 当该中断寄存器中的所有位清零时，该位清零。 仅当您选择可选的 MMC 模块和 Checksum Offload 时，此位才有效配置期间的引擎（类型 2）。
MMCTIS	6	RO	0x0	MMC 发送中断状态 只要在 MMC 发送中断中产生中断，该位就会置为高电平 寄存器。 当该中断寄存器中的所有位清零时，该位清零。 只有在配置期间选择可选的 MMC 模块时，该位才有效。
MMCRIS	5	RO	0x0	MMC 接收中断状态 只要在 MMC 接收中断中产生中断，该位就会置为高电平 寄存器。 当该中断寄存器中的所有位清零时，该位清零。

				只有在配置期间选择可选的 MMC 模块时，该位才有效
MMCIS	4	RO	0x0	只要位[7：5]中的任何一位设置为高电平，该位就会置为高电平;只有当所有这些位都为低电平时，该位才会清零。 只有在配置期间选择可选的 MMC 模块时，该位才有效。
PMTIS	3	RO	0x0	PMT Interrupt Status 无论何时在掉电模式下接收到 Magic 数据包或 LAN 唤醒帧，此位都会置 1。 由于对 PMT 控制和状态寄存器的读操作，当[6：5]位清零时，该位清零。 只有在配置期间选择可选的 PMT 模块时，该位才有效。
PCSANC	2	RO	0x0	PCS 自协商完成 在 TBI/RTBI/SGMII PHY 接口寄存器 49（AN 状态寄存器）中的位[5]中完成自动协商时，该位置 1。 当用户对 AN 状态寄存器进行读操作时，该位清零。 只有在配置和操作期间选择可选的 TBI/RTBI/SGMII PHY 接口时，该位才有效。
PCSLSC	1	RO	0x0	PCS 链路状态改变 由于 TBI/RTBI/SGMII PHY 接口中的链路状态发生任何变化（寄存器 49（AN 状态寄存器）中的位 2），

				因此该位置 1。当用户对 AN 状态寄存器进行读操作时，该位清零。只有在配置和操作期间选择可选的 TBI/RTBI/SGMII PHY 接口时，该位才有效。
RSIS	0	RO	0x0	RGMII/SMII Interrupt Status 由于 RGMII/SMII 接口的链路状态（寄存器 54 中的位 3（SGMII/RGMII/SMII 状态寄存器））的值发生任何变化，因此该位置 1。当用户对 SGMII/RGMII/SMII 状态寄存器进行读操作时，该位清零。 只有在配置和操作期间选择可选的 RGMII/SMII PHY 接口时，该位才有效。

5.2.4.15 中断屏蔽寄存器

域	位	R/W	初值	说明
Reserved	31:11	RO	0x0	保留
LPIIM	10	RW	0x0	LPI 中断屏蔽 置 1 时，由于寄存器 14（中断状态寄存器）中的 LPI 中断状态位置 1，该位禁止中断信号的置位。 只有在配置期间选择节能以太网功能时，此位才有效。 在所有其他模式中，该位保留。
TIM	9	RW	0x0	时间戳中断屏蔽

				置 1 时，由于设置，该位禁止中断信号的置位 寄存器 14（中断状态寄存器）中的时间戳中断状态位。 该位仅在启用 IEEE1588 时间戳时有效。 在所有其他模式中，该位保留。
Reserved	8:4	RW	0x0	保留
PMTIM	3	RW	0x0	PMT 中断屏蔽 置 1 时，由于设置，该位禁止中断信号的置位 寄存器 14（中断状态寄存器）中的 PMT 中断状态位。
PCSANCIM	2	RW	0x0	PCS AN 完成中断屏蔽 置 1 时，由于自动协商事件完成导致寄存器 14（中断状态寄存器）中的 PCS 自动协商完成位置 1，该位禁止中断信号的置位。
PCSLSIM	1	RW	0x0	PCS 链路状态中断屏蔽 置 1 时，由于链路状态事件发生变化导致寄存器 14（中断状态寄存器）中的 PCS Link 状态改变位置位，该位禁止中断信号置位。
RSIM	0	RW	0x0	RGMII/SMII 中断屏蔽 置 1 时，由于设置，该位禁止中断信号的置位 寄存器 14（中断状态寄存器）中的 RGMII / SMII 中断状态位。

5.2.4.16 MAC 地址 0 高寄存器

域	位	R/W	初值	说明
MO	31	RO	0x1	一直为 1
Reserved	30:16	RO	0x0	保留
A[47:32]	15:0	RW	0xFFFF	MAC 地址 0[47:32] 该字段包含 6 字节第一 MAC 地址的高 16 位（4732）。 MAC 使用它来过滤接收的帧并将 MAC 地址插入发送流控制暂停帧中。

5.2.4.17 MAC 地址 0 低寄存器

域	位	R/W	初值	说明
A[31:0]	31:0	RW	0xFFFF_FFFF	MAC 地址 0[31:0] 该字段包含 6 字节第一 MAC 地址的低 32 位。 MAC 使用它来过滤接收的帧并将 MAC 地址插入发送流控制暂停帧中。

5.2.4.18 MAC 地址 1 高寄存器

域	位	R/W	初值	说明
AE	31	RW	0x0	此位置 1 时，地址过滤器模块使用第二个 MAC 地址进行完美过滤。复位时，地址过滤器模块忽略用于过滤的地址。
SA	30	RW	0x0	源地址 当该位置 1 时， MAC 地址 1[47:0]

				用于与接收帧的源地址字段进行比较。 当该位复位时，MAC 地址 1[47:0]用于与接收帧的目的地址字段进行比较。
MBC	29:24	RW	0x0	这些位是掩码控制位，用于比较每个 MAC 地址字节。 设置为高电平时，GMAC 内核不会将接收到的目的地址/源地址的相应字节与 MAC 地址 1 寄存器的内容进行比较。 每个位控制字节的屏蔽，如下所示: 位 29 寄存器 18[15:8] 位 28 寄存器 18[7:0] 位 27 寄存器 19[31:24] ... 位 24 寄存器 19[7:0]
Reserved	23:16	RO	0x0	保留
A[47:32]	15:0	RW	0xFFFF	MAC 地址 1[47:32] 该字段包含 6 字节第二 MAC 地址的高 16 位[47:32]

5.2.4.19 MAC 地址 1 低寄存器

域	位	R/W	初值	说明
A[31:0]	31:0	RW	0xFFFF_FFFF	MAC 地址 1[31:0] 该字段包含 6 字节第二 MAC 地址的低 32 位。 在初始化过程之后由应用程序加载之前，此字段的内容

				是未定义的。

5.2.4.20 寄存器 54

域	位	R/W	初值	说明
Reserved	31:6	RO	0x0	保留
FCD	5	RO	0x0	指示 SMII PHY 是否检测到错误载波（1'b1）。 为核心配置 SGMII / RGMII PHY 接口时保留。
JT	4	RO	0x0	Jabber 超时 指示接收帧中是否存在 jabber 超时错误（1'b1）。为核心配置 SGMII / RGMII PHY 接口时保留。
LS	3	RO	0x0	链路状态 指示链接是 up（1'b1）还是 down（1'b0）。
LSPEED	2:1	RO	0x0	链路速度 显示当前链路速度: 00 2.5 MHz 01 25 MHz 10 125 MHz
LMODE	0	RO	0x0	Link Mode 指示链路的当前操作模式: 1’b0 半双工模式 1’b1 全双工模式

5.2.5 GMAC DMA 寄存器

5.2.5.1 GMAC DMA 寄存器列表

表 5-11 GMAC DMA 寄存器列表

寄存器	偏移	描述
总线模式寄存器	0x1000	控制主机接口模式
发送轮询请求寄存器	0x1004	主机使用它来指示 DMA 轮询传输描述符列表。
发送轮询请求寄存器	0x1008	主机使用它来指示 DMA 轮询接收描述符列表。
接收描述符列表地址寄存器	0x100c	将 DMA 指向接收描述符列表的开头。
发送描述符列表地址寄存器	0x1010	将 DMA 指向传输描述符列表的开头。
状态寄存器	0x1014	软件驱动程序（应用程序）在中断服务程序或轮询期间读取该寄存器以确定 DMA 的状态。
操作模式寄存器	0x1018	建立接收和发送操作模式和命令。
中断使能寄存器	0x101c	启用状态寄存器报告的中断。
丢帧和缓冲区溢出计数器寄存器	0x1020	包含丢弃帧的计数器，因为没有主机接收描述符可用，并且由于接收 FIFO 溢出而丢弃帧。
接收中断看门狗定时器寄存器	0x1024	从 DMA 接收中断（RI）的看门狗超时。
AXI 总线模式寄存器	0x1028	控制 AXI 主行为（主要控制突发分裂和未完成请求的数量）。
AXI 状态寄存器	0X102C	在 GMAC-AHB 配置中给出 AHB 主接口的空闲状态。 在 GMAC-AXI 配置中给出 AXI 主机读或写通道的空闲状态。
当前主机发送描述符寄存器	0X1048	指向 DMA 读取的当前发送描述符的开

		始。
当前主机接收描述符寄存器	0X104C	指向 DMA 读取的当前发送描述符的开始。
当前主机发送缓冲地址寄存器	0X1050	指向 DMA 读取的当前发送缓冲区地址。
当前主机发送缓冲地址寄存器	0x1054	指向 DMA 读取的当前接收缓冲区地址。
硬件功能寄存器	0x1058	表示核心的可选功能的存在。

5.2.5.2 总线模式寄存器

域	位	R/W	初值	说明
Reserved	31:30	RO	0x0	保留
PRWG	29:28	RW	0x0	通道优先权重 在循环仲裁期间设置通道 0 的优先级权重系统总线的 DMA 通道。 值 优先权重 00 1 01 2 10 3 11 4 当您选择 AV 功能时，除 GMAC-AXI 外，所有 GMAC 配置中都会出现这些位。 否则，这些位被保留并且是只读的（RO）。
TXPR	27	RW	0x0	发送优先级 置 1 时，该位表示在系统端总线仲裁期间，发送 DMA 的优先级高于接收 DMA。 在 GMAC-AXI 配置

				中，该位保留为只读（RO）。
MB	26	RW	0x0	混合突发 对于 gmac-axi 配置这个位保留。
AAL	25	RW	0x0	当该位设置为高电平且 FB 位等于 1 时，AHB / AXI 接口会产生与起始地址 LS 位对齐的所有突发。如果 FB 位等于 0，则第一个突发（访问数据缓冲区的起始地址）不对齐，但随后的突发与地址对齐。 该位仅在 GMAC-AHB 和 GMAC-AXI 配置中有效并保留（所有其他配置中的 RO，默认值为 0）。
8xPBL	24	RW	0x0	8xPBL 模式 设置为高电平时，该位将编程的 PBL 值（位[22:17]和位[13：8]）相乘 8 次。 因此，DMA 根据 PBL 值以 8、16、32、64、128 和 256 个节拍传输数据。
USP	23	RW	0x0	使用单独的 PBL 设置为高电平时，该位将 RxDMA 配置为使用位[22:17]中配置的值作为 PBL，而位[13：8]中的 PBL 值仅适用于 TxDMA 操作。 当复位为低电平时，位[13：8]中的 PBL 值适用于两个 DMA 引擎。
RPBL	22:17	RW	0x1	RxDMA PBL 这些位表示在一个 RxDMA 事务

				中要传输的最大节拍数。 这是单个块读/写中使用的最大值。 每次在主机总线上启动突发传输时，RxDMA 总是按照 RPBL 中的规定尝试突发。 可以使用允许的值 1,2,4,8,16 和 32 对 RPBL 进行编程。任何其他值都会导致未定义的行为。 这些位有效且仅在 USP 设置为高时适用。
FB	16	RW	0x0	固定突发 该位控制 AHB / AXI 主接口是否执行固定突发传输。置 1 时，AHB 在正常突发传输开始时仅使用 SINGLE， INCR4， INCR8 或 INCR16。 复位时，AHB / AXI 使用 SINGLE 和 INCR 突发传输操作。 在 GMAC-AXI 配置中，有关详细说明，请参见 AXI 总线模式寄存器的位 0（UNDEF：AXI 未定义突发长度）。
PR	15:14	RW	0x0	这些位控制 RxDMA 和 TxDMA 之间的加权循环仲裁中的优先级比率。 仅当位 1（DA：DMA 仲裁方案）复位时，这些位才有效。 优先级比率为 Rx：Tx 或 Tx：Rx，具体取决于位 27（TXPR：发送优先级）是复位还是置位。

				值 优先比率 00 1:1 01 2:1 10 3:1 11 4:1 在 GMAC-AXI 配置中，这些位保留为只读（RO）。
PBL	13:8	RW	0x0	PBL: 可编程突发长度这些位指示在一个 DMA 事务中要传输的最大节拍数。 这是单个块读/写中使用的最大值。 每次在主机总线上启动突发传输时，DMA 总是按照 PBL 中的指定尝试突发。 可以使用允许的值 1，2，4，8，16 和 32 对 PBL 进行编程。任何其他值都会导致未定义的行为。 当 USP 设置为高时，此 PBL 值仅适用于 TxDMA 事务。
ATDS	7	RW	0x0	交替描述表大小 设置为 1 时，32 字节的增强描述符表 设置为 0 时，16 字节的描述符表
DSL	6:2	RW	0x0	描述符跳跃长度 这些位根据总线的宽度 32 位、64 位、128 位指定多少个字节、双字节、四字节在两个不相连的描述符表跳跃。
DA	1	RW	0x0	1。设置 8xPBL 模式。 2.设置 PBL。 例如，如果要传输的最大

				节拍数为 64，则首先将 8xPBL 设置为 1，然后将 PBL 设置为 8.PBL 值具有以下限制：可能节拍的最大数量（PBL）受大小限制 MTL 层中的 Tx FIFO 和 Rx FIFO 以及 DMA 上的数据总线宽度。除非指定，否则 FIFO 具有约束，即支持的最大差数是 FIFO 深度的一半。对于不同的数据总线宽度和 FIFO 大小，下表中提供了有效的 PBL 范围（包括 x8 模式）。
SWR	0	W1C RS	0x1	软件复位 该位置 1 时，MAC DMA 控制器将复位所有 GMAC 子系统内部寄存器和逻辑。 在所有核心时钟域中完成复位操作后，它会自动清零。在重新编程内核的任何寄存器之前，在该位读取 0 值. 注：仅当所有有效时钟域中的所有复位均置为无效时，才会完成复位操作。 因此，必须存在所有 PHY 输入时钟（适用于所选 PHY 接口）以完成软件复位.

5.2.5.3 发送轮询请求寄存器

域	位	R/W	初值	说明
TPD	31:0	RO_ WT	0x0	当这些位写入任何值时，DMA 读取寄存器 18 指向的当前描述符。

				如果该描述符不可用（由主机拥有），则传输返回到挂起状态，并且 DMA 寄存器 5[2]被置位。 如果描述符可用，则继续传输。

5.2.5.4 发送轮询请求寄存器

域	位	R/W	初值	说明
RPD	31:0	RO_WT	0x0	当这些位写入任何值时，DMA 读取寄存器 19 指向的当前描述符。 如果该描述符不可用（由主机拥有），则接收返回到挂起状态，寄存器 5[7]未置位。 如果描述符可用，则接收 DMA 返回活动状态。

5.2.5.5 接收描述符列表地址寄存器

域	位	R/W	初值	说明
START_REC_LIST	31:4	RW	0x0	该字段包含接收描述符列表中第一个描述符的基址。 忽略 32/64/128 位总线宽度的 LSB 位[1/2/3:0]，并在内部由 DMA 取全零。 因此，这些 LSB 位是只读（RO）.

5.2.5.6 发送描述符列表地址寄存器

域	位	R/W	初值	说明

START_TRA_LIST	31:4	RW	0x0	该字段包含发送描述符列表中第一个描述符的基址。 忽略 32/64/128 位总线宽度的 LSB 位[1/2/3:0]，并在内部由 DMA 取全零。 因此，这些 LSB 位是只读（RO）.

5.2.5.7 状态寄存器

域	位	R/W	初值	说明
reserved	31	RO	0x0	保留
GLPII	30	RO	0x0	GMAC LPI 中断（针对通道 0） 该位指示 GMAC 内核的 LPI 逻辑中的中断事件。 要将此位复位为 1'b0，软件必须读取 GMAC 内核中的相应寄存器，以获得中断的确切原因并清除其来源. 注意：GLPII 状态仅在通道 0 DMA 寄存器中给出，仅在启用节能以太网功能时才适用。 否则，该位保留.
TTI	29	RO	0x0	时间戳触发中断 该位指示 GMAC 内核的时间戳生成器块中的中断事件。 软件必须读取 GMAC 内核中的相应寄存器以获得中断的确切原因并清除其源以将该位复位为 1'b0。 该位为高时，来自 GMAC 子系统（sbd_intr_o）的中断信号为高电

				平.
GPI	28	RO	0x0	GMAC PMT 中断 该位指示 GMAC 内核的 PMT 模块中的中断事件。 软件必须读取 GMAC 内核中的相应寄存器以获得中断的确切原因并清除其源以将该位复位为 1'b0。 当该位为高时，来自 GMAC 子系统（sbd_intr_o）的中断信号为高电平.
GMI	27	RO	0x0	GMAC MMC 中断 该位反映 GMAC 内核的 MMC 模块中的中断事件。 软件必须读取 GMAC 内核中的相应寄存器，以获得中断的确切原因并清除中断源，使该位为 1'b0。 当该位为高时，来自 GMAC 子系统（sbd_intr_o）的中断信号为高电平。
GLI	26	RO	0x0	GMAC 线路接口中断 该位反映 GMAC Core 的 PCS，SMII 或 RGMII 接口模块中的中断事件。 软件必须读取 GMAC 内核中的相应寄存器，以获得中断的确切原因并清除中断源，使该位为 1'b0。 当该位为高时，来自 GMAC 子系统（sbd_intr_o）的中断信号为高电平.
EB	25:23	RO	0x0	这些位指示导致总线错误的错误

				类型（例如，AHB / AXI 接口上的错误响应）。 仅当位 13（FBI：致命总线错误中断）置 1 时，这些位才有效。 该字段不会产生中断. 位 23 1'b1 TxDMA 在数据传输过程中出错 1'b0 RxDMA 在数据传输期间出错 位 24 1'b1 读传输期间出错 1'b0 写传输期间出错 位 25 1'b1 描述符访问期间出错 1'b0 数据缓冲区访问期间出错
TS	22:20	RO	0x0	这些位指示发送 DMA FSM 状态。该字段不会产生中断. • 3’b000: 停止；发出重置或停止发送命令. • 3’b001: 运行；获取发送描述符. • 3’b010: 运行；等待状态. • 3’b011: 运行；从主机内存缓冲区读取数据并将其排队到发送缓冲区（发送 FIFO）. • 3’b100: TIME_STAMP 写状态. • 3’b101: 保留供将来使用. • 3’b110: 暂停；发送描述符不可用或发送缓冲区下溢. • 3’b111: 运行；关闭发送描述符.
RS	19:17	RO	0x0	这些位指示接收 DMA FSM 状态。该字段不会产生中断.

				• 3’b000: 已停止：已发出重置或停止接收命令. • 3’b001: 正在运行：获取接收传输描述符. • 3’b010: 保留供将来使用. • 3’b011: 正在运行：正在等待接收数据包. • 3’b100: 暂停：接收描述符不可用. • 3’b101: 运行：关闭接收描述符. • 3’b110: 时间戳写状态. • 3’b111: 运行：将接收数据包数据从接收缓冲区传输到主机内存.
NIS	16	R_SS_WC	0x0	正常中断摘要 正常中断摘要位值是在 DMA 寄存器 7 中使能相应中断位时的以下逻辑或: • 寄存器 5[0]: 传输中断 • 寄存器 5[2]: 发送缓冲区不可用 • 寄存器 5[6]: 接收中断 • 寄存器 5[14]: 早期接收中断 只有未屏蔽的位才会影响正常中断摘要位. 这是一个粘滞位，每次清除导致 NIS 置位的相应位时，必须将其清零（通过向该位写入 1）.
AIS	15	R_SS_WC	0x0	异常中断摘要 当 DMA 寄存器 7 中使能相应的中断位时，异常中断汇总位值是以下

				逻辑或: • 寄存器 5[1]: 发送进程已停止 • 寄存器 5[3]: 传输 Jabber 超时 • 寄存器 5[4]: 接收 FIFO 溢出 • 寄存器 5[5]: 传输下溢 • 寄存器 5[7]: 接收缓冲区不可用 • 寄存器 5[8]: 接收进程已停止 • 寄存器 5[9]: 接收看门狗超时 • 寄存器 5[10]: 早期传输中断 • 寄存器 5[13]: 致命的总线错误 只有未屏蔽的位会影响异常中断摘要位. 这是一个粘滞位，每次清除导致 AIS 置位的相应位时必须清零.
ERI	14	R_SS_WC	0x0	早期接收中断 该位表示 DMA 已填充数据包的第一个数据缓冲区。 接收中断寄存器 5[6]自动清除该位.
FBI	13	R_SS_WC	0x0	FBI: 致命总线错误中断 该位表示发生了总线错误，详见[25:23]。 设置此位后，相应的 DMA 引擎将禁用其所有总线访问.
Reserved	12:11	RO	0x0	保留
ETI	10	R_SS_WC	0x0	早期发送中断 该位表示要发送的帧已完全传送到 MTL 发送 FIFO.
RWT	9	R_SS_WC	0x0	接收看门狗超时 当接收到长度大于 2048 字节的帧时，该位置位（启用 Jumbo 帧模式

				时为 10,240）.
RPS	8	R_SS_WC	0x0	接收进程已停止 当接收进程进入 Stopped 状态时，该位置位。
RU	7	R_SS_WC	0x0	接收缓冲区不可获取 该位表示接收列表中的下一个描述符由主机拥有，并且不能被 DMA 获取。 接收流程暂停。 要继续处理接收描述符，主机应更改描述符的所有权并发出接收轮询请求命令。 如果未发出接收轮询请求，则在接收到下一个识别的传入帧时，接收进程将恢复。 仅当前一个接收描述符由 DMA 拥有时，才设置寄存器 5 [7]。
RI	6	R_SS_WC	0x0	接收中断 该位表示帧接收完成。 特定帧状态信息已在描述符中发布。 接待处仍处于运行状态。
UNF	5	R_SS_WC	0x0	发送下溢 该位表示发送缓冲区在帧发送期间有下溢。 传输暂停，并设置下溢错误 TDES0[1]。
OVF	4	R_SS_WC	0x0	接收溢出 该位表示接收缓冲区在帧接收期间有溢出。 如果将部分帧传送到应用程序，则在 RDES0[11]中设置溢出状态。
TJT	3	R_SS	0x0	发送 Jabber 超时

		_WC		该位表示发送 Jabber 计时器已到期，这意味着发送器已过度激活。传输过程中止并置于"已停止"状态。 这会导致发送 Jabber 超时 TDES0[14]标志置位。
TU	2	R_SS _WC	0x0	发送缓冲区不可获取 该位表示发送列表中的下一个描述符由主机拥有，并且不能被 DMA 获取。 传输暂停。 位 22:20 解释了传输过程状态转换。 要恢复处理发送描述符，主机应更改描述符位的所有权，然后发出发送轮询请求命令。
TPS	1	R_SS _WC	0x0	发送停止时设置该位。
TI	0	R_SS _WC	0x0	发送中断 该位表示帧传输结束，TDES1[31]在第一个描述符中设置。

5.2.5.8 操作模式寄存器

域	位	R/W	初值	说明
Reserved	31:27	RO	0x0	
DT	26	RW	0x0	DT: 禁用丢弃 TCP / IP 校验和错误帧 设置此位后，内核不会丢弃只有 Receive Checksum Offload 引擎检测到错误的帧。 这些帧在 MAC 接收的以太网帧中没有任何错误

				（包括 FCS 错误），但仅在封装的有效载荷中存在错误。 当该位复位时，如果 FEF 位复位，则所有错误帧都将被丢弃。 如果禁用完整校验和卸载引擎（类型 2），则保留该位（RO 值为 1'b0）。
RSF	25	RW	0x0	接收存储转发 当该位置 1 时，MTL 仅在写入完整帧后从 Rx FIFO 读取一帧，忽略 RTC 位。 当该位复位时，Rx FIFO 以直通模式工作，受 RTC 位指定的阈值限制。
DFF	24	RW	0x0	禁用正在接收的帧的刷新 当该位置 1 时，RxDMA 不会刷新任何帧，因为接收描述符/缓冲区不可用，因为这个位复位时正常。
RFA[2]	23	RW	0x0	激活流量控制的阈值 MSB 如果 GMAC-UNIV 配置为 Rx FIFO 深度为 8 KB 或更高，则该位（设置时）提供额外的阈值电平，用于在半双工和全双工模式下激活流控制。 该位（作为最高有效位）与 RFA（位[10：9]）一起为激活流控制提供以下阈值。 • 100 完全减去 5 KB • 101 完全减去 6 KB • 110 完全减去 7 KB • 111 保留 如果 Rx FIFO 的深度为 4 KB 或更

				小，则该位保留（和 RO）。
RFD[2]	22	RW	0x0	用于停用流量控制的阈值的 MSB 如果 GMAC-UNIV 配置为 Rx FIFO 深度为 8 KB 或更高，则该位（设置时）提供额外的阈值电平，用于在半双工和全双工模式下停用流控制。 该位（作为最高有效位）与 RFD（位 1211）一起给出以下阈值以停用流量控制。 • 100 完全减去 5 KB • 101 完全减去 6 KB • 110 完全减去 7 KB • 111 保留 如果 Rx FIFO 的深度为 4 KB 或更小，则该位保留（和 RO）。
TSF	21	RW	0x0	发送存储和转发 当该位置 1 时，当全帧驻留在 MTL 发送 FIFO 中时，发送开始。 该位置 1 时，忽略寄存器 6[16:14]中指定的 TTC 值。 只有在传输停止时才应更改此位。
FTF	20	R_WS_SC	0x0	刷新发送 FIFO 当该位置 1 时，发送 FIFO 控制器逻辑将复位为其默认值，因此 Tx FIFO 中的所有数据都将丢失/刷新。 当冲洗操作完全完成时，该位在内部被清除。 在该位清零之前，不应写入操作模式寄存器。 MAC 发送器已经接受的数据不会

				被刷新。 它被安排用于传输并导致下溢和欠幅帧传输。 注意：刷新操作仅在清空其内容的 TxFIFO 并且主机接受所传输的帧的所有未决传输状态后才完成。为了完成此刷新操作，需要 PHY 发送时钟（clk_tx_i）处于活动状态。
Reserved	19:17	RO	0x0	保留
TTC	16:14	RW	0x0	传输阈值控制 这三个位控制 MTL 发送 FIFO 的阈值电平。 当 MTL 发送 FIFO 中的帧大小大于阈值时，发送开始。另外，还发送长度小于阈值的全帧。 仅当 TSF 位（位 21）复位时，才使用这些位。 • 000 64 • 001 128 • 010 192 • 011 256 • 100 40 • 101 32 • 110 24 • 111 16
ST	13	RW	0x0	启动/停止传输命令 当该位置 1 时，发送处于运行状态，DMA 检查当前位置的发送列表以寻找要发送的帧。 从列表中的当前位置（即寄存器 4 设置的发

				送列表基址）或先前停止传输时保留的位置尝试获取描述符。 如果当前描述符不归 DMA 所有，则传输进入挂起状态，并且发送缓冲区不可用（寄存器 5 2）置位。 启动传输命令仅在传输停止时有效。如果在设置 DMA 寄存器 4 之前发出命令，则 DMA 行为是不可预测的。 当该位复位时，在完成当前帧的传输之后，传输过程处于停止状态。传输列表中的下一个描述符位置被保存，并在重新开始传输时成为当前位置。 停止传输命令仅在当前帧的传输完成或传输处于暂停状态时有效。
RFD	12:11	RW	0x0	停用流量控制的阈值（HD 和 FD）这些位控制激活后流控制被置为无效的阈值（Rx FIFO 的填充级别）。 • 00 Full minus 1 KB • 01 Full minus 2 KB • 10 Full minus 3 KB • 11 Full minus 4 KB 请注意，取消断言仅在断言流量控制后才有效。 如果 Rx FIFO 为 8 KB 或更多，则将更多位（RFD [2]）用于更多阈值电平，如位[22]中所述。当 RxFIFO 深度小于 4 KB 时，

				这些位保留为只读。
RFA	10:9	RW	0x0	激活流量控制的阈值（HD 和 FD） 这些位控制激活流量控制的阈值（Rx FIFO 的填充水平）。 • 00 Full minus 1 KB • 01 Full minus 2 KB • 10 Full minus 3 KB • 11 Full minus 4 KB 注意，当 EFC 位设置为高时，上述仅适用于 4 KB 或更大的 Rx FIFO。 如果 Rx FIFO 为 8 KB 或更多，则更多位（RFA[2]）用于更多阈值电平，如位[23]中所述。 当 RxFIFO 深度小于 4 KB 时，这些位保留为只读。
EFC	8	RW	0x0	启用 HW 流量控制 该位置 1 时，使能基于 Rx FIFO 填充电平的流量控制信号操作。 复位时，禁用流量控制操作。 当 Rx FIFO 小于 4 KB 时，不使用该位（保留并始终复位）。
FEF	7	RW	0x0	前向错误帧 当该位复位时，Rx FIFO 会丢失具有错误状态的帧（CRC 错误，冲突错误，GMII_ER，巨帧，看门狗超时，溢出）。 但是，如果帧的起始字节（写入）指针已经传输到读取控制器端（在阈值模式下），则不会丢弃帧。

FUF	6	RW	0x0	转发不成熟的好框架 置位时，Rx FIFO 转发 Undersized 帧（没有错误且长度小于 64 字节的帧，包括 pad-bytes 和 CRC）。 复位时，Rx FIFO 会丢弃少于 64 字节的所有帧，除非由于接收阈值的值较低（例如，RTC = 01）而已经传输。
Reserved	5	RO	0x0	保留
RTC	4:3	RW	0x0	接收阀值控制 这两位控制 MTL 接收 FIFO 的阈值电平。 当 MTL 接收 FIFO 中的帧大小大于阈值时，传输（请求）到 DMA 开始。 此外，长度小于阈值的全帧会自动传输。 请注意，如果配置的接收 FIFO 大小为 128 字节，则值 11 不适用。 这些位仅在 RSF 位为零时有效，并在 RSF 位设置为 1 时被忽略。 • 00 64 • 01 32 • 10 96 • 11 128
OSF	2	RW	0x0	在第二帧上操作 当该位置 1 时，即使在获得第一帧的状态之前，该位也指示 DMA 处理第二帧发送数据。
SR	1	RW	0x0	开始/停止接收 设置此位后，接收进程将处于"正

				在运行"状态。 DMA 尝试从接收列表中获取描述符并处理传入的帧。 尝试从列表中的当前位置获取描述符，该位置是 DMA 寄存器 3 设置的地址或接收过程先前停止时保留的位置。 如果 DMA 没有描述符，则暂停接收并设置接收缓冲区不可用（寄存器 5[7]）。 启动接收命令仅在接收停止时有效。如果在设置 DMA 寄存器 3 之前发出命令，则 DMA 行为是不可预测的。 当该位清零时，在传输当前帧后停止 RxDMA 操作。 接收列表中的下一个描述符位置将被保存，并在重新启动接收进程后成为当前位置。 停止命令仅在接收进程处于运行状态（等待接收数据包）或挂起状态时有效。
Reserved	0	RO	0x0	保留

5.2.5.9 中断使能寄存器

域	位	R/W	初值	说明
Reserved	31:17	RO	0x0	保留
NIE	16	RW	0x0	正常中断摘要启用 该位置 1 时，使能正常中断。 该位复位时，禁止正常中断。 该位使能以下位

				• 寄存器 5[0] 发送中断 • 寄存器 5[2] 发送缓冲区不可获取 • 寄存器 5[6] 接收中断 • 寄存器 5[14] 早期接收中断
AIE	15	RW	0x0	异常中断摘要启用 该位置 1 时，使能异常中断。 该位复位时，禁止异常中断。 该位使能以下位 • 寄存器 5[1] 发送过程停止 • 寄存器 5[3] 发送 Jabber 超时 • 寄存器 5[4] 接收溢出 • 寄存器 5[5] 发送下溢 • 寄存器 5[7] 接收缓冲区不可用 • 寄存器 5[8] 接收进程已停止 • 寄存器 5[9] 接收看门狗超时 • 寄存器 5[10] 早期传输中断 • 寄存器 5[13] 致命的总线错误
ERE	14	RW	0x0	早期接收中断使能 当该位设置为正常中断摘要使能（寄存器 7[16]）时，将启用早期接收中断。 该位复位时，禁止早期接收中断。
FBE	13	RW	0x0	致命总线错误启用 当该位设置为异常中断摘要使能（寄存器 7[15]）时，将启用致命总线错误中断。 该位复位时，禁用致命总线错误使能中断。
Reserved	12：11	RO	0x0	保留

ETE	10	RW	0x0	早期发送中断使能 当该位设置为异常中断摘要使能（寄存器 7[15]）时，使能早期发送中断。 该位复位时，禁止早期发送中断。
RWE	9	RW	0x0	接收看门狗超时使能 当该位设置为异常中断摘要使能（寄存器 7[15]）时，将使能接收看门狗超时中断。 该位复位时，禁止接收看门狗超时中断。
RSE	8	RW	0x0	接收已停止启用 当该位设置为异常中断摘要使能（寄存器 7[15]）时，使能接收停止中断。 该位复位时，禁止接收停止中断。
RUE	7	RW	0x0	接收缓冲区不可用启用 当该位设置为异常中断摘要使能（寄存器 7[15]）时，使能接收缓冲区不可用中断。 该位复位时，禁止接收缓冲区不可用中断。
RIE	6	RW	0x0	接收中断使能 当该位设置为正常中断摘要使能（寄存器 7[16]）时，使能接收中断。 该位复位时，禁止接收中断。
UNE	5	RW	0x0	下溢中断使能 当该位设置为异常中断摘要使能（寄存器 7[15]）时，发送下溢中断使能。 该位复位时，将禁止下溢中断。

OVE	4	RW	0x0	溢出中断使能 当该位设置为异常中断摘要使能（寄存器 7[15]）时，使能接收溢出中断。 该位复位时，禁止溢出中断
THE	3	RW	0x0	发送 Jabber 超时启用 当该位设置为异常中断摘要使能（寄存器 7[15]）时，将使能发送 Jabber 超时中断。 该位复位时，发送 Jabber 超时中断被禁止。
TUE	2	RW	0x0	发送缓冲区不可用启用 当该位设置为正常中断摘要使能（寄存器 7[16]）时，使能发送缓冲区不可用中断。 该位复位时，发送缓冲区不可用中断被禁止。
TSE	1	RW	0x0	传输停止启用 当该位设置为异常中断摘要使能（寄存器 7[15]）时，使能发送停止中断。 该位复位时，禁止发送停止中断。
TIE	0	RW	0x0	发送中断使能 当该位设置为正常中断摘要使能（寄存器 7[16]）时，使能发送中断。 该位复位时，禁止发送中断。

5.2.5.10 丢帧和缓冲区溢出计数器寄存器

域	位	R/W	初值	说明
reserved	31:29	RO	0x0	保留

OVFIFO	28	R_SS_RC	0x0	FIFO 溢出计数器的溢出位
CFIFO	27:17	R_SS_RC	0x0	表示应用程序错过的帧数。 每当 MTL 断言边带信号 mtl_rxoverflow_o 时，该计数器就递增。 当使用 mci_be_i 2 在 1'b1 读取该寄存器时，计数器清零。
OVMIS	16	R_SS_RC	0x0	丢帧计数器的溢出位
CMIS	15:0	R_SS_RC	0x0	表示由于主机接收缓冲区不可用而导致控制器丢失的帧数。 每次 DMA 丢弃传入帧时，此计数器都会递增。当使用 mci_be_i 0 在 1'b1 读取该寄存器时，计数器清零。

5.2.5.11 接收中断看门狗定时器寄存器

域	位	R/W	初值	说明
Reserved	31:8	RO	0x0	保留
RIWT	7:0	RIWT	0x0	表示系统时钟周期数乘以 256，其中设置了看门狗定时器。 在 RxDMA 完成由于相应描述符 RDES1[31]中的设置而未设置 RI 状态位的帧的传输之后，看门狗定时器被编程的值触发。 当看门狗定时器用完时，RI 位置位且定时器停止。 当 RI 位设置为高电平时，看门狗定时器复位，因为根据任何接收帧的 RDES1[31]自动设

				置 RI.

5.2.5.12 AXI 总线模式寄存器

域	位	R/W	初值	说明
EN_LPI	31	RW	0x0	启用 LPI（低功耗接口） 设置为 1 时，启用 GMAC-AXI 支持的 LPI（低功耗接口）模式，并接受来自 AXI 系统时钟控制器的 LPI 请求。 设置为 0 时，禁用 LPI 模式并始终拒绝来自 AXI 系统时钟控制器的 LPI 请求。
UNLCK_ON_MGK_RWK	30	RW	0x0	在 Magic 包文或远程唤醒上解锁 当设置为 1 时，只有在收到 Magic 包文或远程唤醒时，GMAC-AXI 才能请求退出低功耗模式。
Reserved	29:24	RO	0x0	保留
WR_OSR_LMT	23:20	RW	0x1	AXI 最大写入未决请求限制此值限制 AXI 写入接口上的最大未完成请求。 最大未完成请求数= WR_OSR_LMT
RD_OSR_LMT	19:16	RW	0x1	XI 最大读取未决请求限制此值限制 AXI 读取接口上的最大未完成请求。 最大未完成请求= RD_OSR_LMT+1。
Reserved	15:13	RO	0x0	保留
AXI_AAL	12	RO	0x0	地址对齐的节拍

				该位是只读位，反映 AAL 位寄存器 0（总线模式寄存器 25）。 当该位设置为 1 时，GMAC-AXI 在读和写通道上执行地址对齐的突发传输。
Reserved	11:8	RO	0x0	保留
BLEN256	7	RW	0x0	AXI 突发长度 256 当该位设置为 1 时，允许 GMAC-AXI 在 AXI 主接口上选择 256 的突发长度。 仅当配置参数 AXI_BL 设置为 256 时，该位才存在。否则，该位保留为只读（RO）。
BLEN128	6	RW	0x0	AXI 突发长度 128 当该位设置为 1 时，允许 GMAC-AXI 在 AXI 主接口上选择 128 的突发长度。 仅当配置参数 AXI_BL 设置为 128 或更高时，才会出现此位。 否则，该位保留为只读（RO）。
BLEN64	5	RW	0x0	AXI 突发长度 64 当该位设置为 1 时，允许 GMAC-AXI 在 AXI 主接口上选择 64 的突发长度. 仅当配置参数 AXI_BL 设置为 64 或更高时，才会出现此位。 否则，该位保留为只读（RO）。
BLEN32	4	RW	0x0	AXI 突发长度 32 当该位设置为 1 时，允许

				GMAC-AXI 在 AXI 主接口上选择 32 的突发长度。仅当配置参数 AXI_BL 设置为 32 或更大时才存在该位。 否则，该位保留为只读（RO）。
BLEN16	3	RW	0x0	AXI 突发长度 16 当该位设置为 1 或 UNDEF 设置为 1 时，允许 GMAC-AXI 在 AXI 主接口上选择 16 的突发长度。
BLEN8	2	RW	0x0	AXI 突发长度 8 当此位设置为 1 或 UNDEF 设置为 1 时，允许 GMAC-AXI 在 AXI 主接口上选择 8 的突发长度。
BLEN4	1	RW	0x0	AXI 突发长度 4 当该位设置为 1 或 UNDEF 设置为 1 时，允许 GMAC-AXI 在 AXI 主接口上选择 4 的突发长度。
UNDEF	0	RO	0x1	AXI 未定义的突发长度 该位为只读位，表示寄存器 0（总线模式寄存器 16）中 FB 位的补码（反相）。 • 当该位设置为 1 时，允许 GMAC-AXI 执行任何等于或低于最大允许突发长度的突发长度，如位 7：1 所示 • 当该位设置为 0 时，允许 GMAC-AXI 仅执行固定的突发长度，如 BLEN256 / 128/64/32/16/8/4 所示，或突发长度为 1。

5.2.5.13 AXI 状态寄存器

域	位	R/W	初值	说明
reserved	31:2	RO	0x0	保留
AXI_R_ACT	1	RO	0x0	当为高电平时，表示 AXI Master 的读通道处于活动状态并传输数据。
AXI_W_ACT	0	RO	0x0	当为高电平时，表示 AXI Master 的写通道处于活动状态，并以 GMAC-AXI 配置传输数据。

5.2.5.14 当前主机发送描述符寄存器

域	位	R/W	初值	说明
HTDAP	31:0	RO	0x0	主机发送描述符地址指针重置时清除。 在操作期间由 DMA 更新指针。

5.2.5.15 当前主机接收描述符寄存器

域	位	R/W	初值	说明
CHRDR	31:0	RO	0x0	主机接收描述符地址指针重置时清除。 在操作期间由 DMA 更新指针。

5.2.5.16 当前主机发送缓冲地址寄存器

域	位	R/W	初值	说明
CHTBAR	31:0	RO	0x0	主机发送缓冲区地址指针重置时清除。 在操作期间由 DMA 更新指针。

5.2.5.17 当前主机发送缓冲地址寄存器

域	位	R/W	初值	说明
CHRBAR	31:0	RO	0x0	主机接收缓冲区地址指针重置时清除。 在操作期间由 DMA 更新指针。

5.2.5.18 硬件功能寄存器

域	位	R/W	初值	说明
PHYIF	31:28	RO		有源或选定的 PHY 接口 RO 当您的配置中有多个 PHY 接口时，这些位指示在复位取消断言期间 phy_intf_sel_i 的采样值 0000 GMII/MII 0001 RGMII 0010 SGMII 0011 TBI 0100 RMII 0101 RTBI 0110 SMII 0111 RevMII

				别的值保留
reserved	27:25	RO		Reserved
	24	RO		替代（增强描述符）
	23:22	RO		额外的 Tx 频道数
	21:20	RO		其他 Rx 频道的数量
	19	RO		RxFIFO > 2048 Bytes
	18	RO		Rx 中的 IP 校验和卸载（类型 2）
	17	RO		Rx 中的 IP 校验和卸载（类型 1）
	16	RO		校验和卸载 Tx
	15	RO		AV 功能
	14	RO		节能以太网功能
	13	RO		IEEE 1588-2008 高级时间戳
	12	RO		IEEE 1588-2002 时间戳
	11	RO		RMON 模块
	10	RO		PMT Magic 包文
	9	RO		PMT 远程唤醒
	8	RO		SMA（MDIO）接口
	7	RO		保留
	6	RO		PCS 寄存器（TBI / SGMII / RTBI PHY 接口）
	5	RO		多个 MAC 地址寄存器
	4	RO		HASH 过滤器
	3	RO		保留
	2	RO		半双工支持
	1	RO		支持 1000 Mbps
	0	RO		10/100 Mbps 支持

5.2.6 GMAC 描述符表

GMAC 描述符分为 2 类：正常描述符和增强描述符。

5.2.6.1 正常描述符格式

以太网子系统中的 DMA 基于链接的描述符列表传输数据，默认描述符格式（接收和发送描述符共用）如图 5-2 所示。

每个描述符包含两个缓冲区，两个字节计数缓冲区和两个地址指针，这使得适配器端口能够与各种类型的内存管理方案兼容。

描述符地址必须与使用的总线宽度对齐（LWord 用于 128 位总线）。 数据总线可以配置为 little-endian 格式。

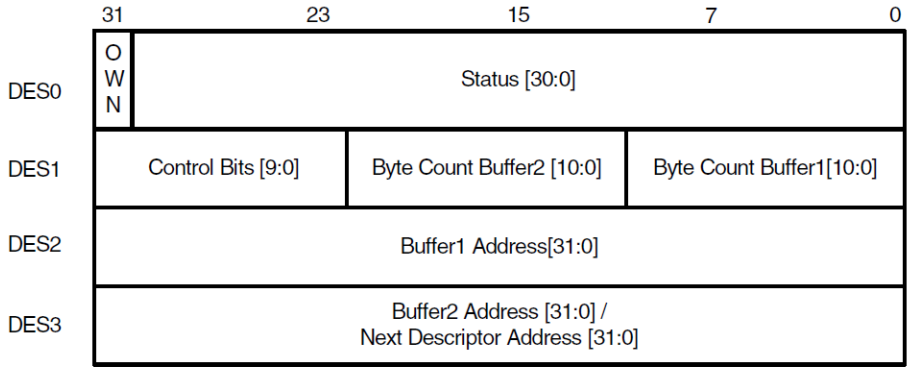

图 5-2 32 位小端格式的同端模式下的 Rx / Tx 描述符

5.2.6.1.1 接收描述符

GMAC 子系统在接收帧时至少需要两个描述符。DMA 的接收状态机（在 GMAC 子系统中）总是尝试在预期输入帧的情况下获取额外的描述符（传入帧的大小未知）。在 RxDMA 关闭描述符之前，即使没有接收到帧，它也会尝试获取下一个描述符。

在单个描述符（接收）系统中，如果接收缓冲区不能容纳传入帧并且下一个描述符不归 DMA 所有，则子系统会生成描述符错误。 因此，强制主机增加其描述符池或缓冲区大小。否则，子系统开始丢弃所有传入的帧。

■ **接收描述符 0（RDES0）**

RDES0 包含接收的帧状态，帧长度和描述符所有权信息。 下表中的描述适

用于具有相同 Endian 描述符的小端 32 位数据总线的默认模式。

域	位	描述
OWN	31	置位时，该位表示描述符归 GMAC 子系统的 DMA 所有。 该位复位时，该位表示描述符归主机所有。 DMA 在完成帧接收或与此描述符关联的缓冲区已满时清除此位。
AFM	30	置位时，该位表示 GMAC 中目的地址滤波器出现故障的帧。
FL	29:16	这些位指示传输到主机存储器（包括 CRC）的接收帧的字节长度。当最后描述符（RDES0[8]）置 1 且描述符错误（RDES0 [14]）或溢出错误位复位时，该字段有效。 当启用 IP 校验和计算（类型 1）并且接收的帧不是 MAC 控制帧时，帧长度还包括附加到以太网帧的两个字节。 设置最后描述符（RDES0[8]）时，该字段有效。 如果未设置 LS 和 ES 位，则此字段指示已为当前帧传输的累计字节数。
ES	15	表示以下位的逻辑或： RDES0 [0]：有效负载校验和错误 RDES0 [1]：CRC 错误 RDES0 [3]：接收错误 RDES0 [4]：看门狗超时 RDES0 [6]：后期冲突 RDES0 [7]：IPC 校验和（类型 2）/巨帧 RDES0[11]：溢出错误 RDES0[14]：描述符错误 该字段仅在设置了最后描述符（RDES0[8]）时有效。
DE	14	置位时，该位表示由不适合当前描述符缓冲区的帧引起的帧截断，并且 DMA 不拥有下一个描述符。 框架被截断。 该字段仅在设置了最后描述符（RDES0[8]）时有效。
SAF	13	置位时，该位表示帧的源地址字段未通过 GMAC 核心中的源地址过滤器。
LE	12	置位时，该位表示接收到的帧的实际长度以及长度/类型字段不匹

		配。 仅当帧类型（RDES0[5]）位复位时，该位才有效。 存在 CRC 错误时，长度错误状态无效。
OE	11	置位时，该位表示由于 MTL 中的缓冲区溢出而导致接收帧被损坏。 注意：仅当 DMA 将部分帧传输到应用程序时才设置此位。 仅当 RxFIFO 在阈值模式下运行时才会发生这种情况。 在存储转发模式下，所有部分帧都完全丢弃在 RxFIFO 中。
VLAN	10	置位时，该位指示该描述符指向的帧是由 GMAC Core 标记的 VLAN 帧。
FS	9	置位时，该位指示该描述符包含帧的第一个缓冲区。 如果第一个缓冲区的大小为 0，则第二个缓冲区包含帧的开头。 如果第二个缓冲区的大小也是 0，则下一个描述符包含帧的开头。
LS	8	置位时，该位指示此描述符指向的缓冲区是帧的最后一个缓冲区。
IPCCE	7	启用校验和引擎（类型 1）时，该位置 1 时表示核心计算的 16 位 IPv4 报头校验和与接收到的校验和字节不匹配。 在此模式下设置此位时，不会设置错误摘要位 15。 如果在启用完整校验和卸载引擎（类型 2）时设置此位，则表示 IPv4 或 IPv6 标头中存在错误。 此错误可能是由于以太网类型字段和 IP 标头版本字段值不一致，IPv4 中的标头校验和不匹配，或者缺少预期数量的 IP 标头字节的以太网帧。 如果在核心配置期间未选择校验和模块，则此位在设置时表示接收的帧是巨帧。 当启用巨型帧处理时，巨帧大于 1518 字节（或 VLAN 为 1522 字节）正常帧和大于 9018 字节（VLAN 为 9022 字节）帧。
LC	6	置位时，该位表示在半双工模式下接收帧时发生了晚期冲突。
FT	5	置位时，该位表示接收帧是以太网类型帧（LT 字段大于或等于 16h0600）。 当该位复位时，表示接收到的帧是 IEEE802.3 帧。 该位对于小于 14 字节的 Runt 帧无效。
RWT	4	置 1 时，该位表示接收看门狗定时器在接收当前帧时已过期，当前帧在看门狗超时后被截断。

RE	3	置位时，该位指示 gmii_rxer_i 信号被置位，而 gmii_rxdv_i 在帧接收期间被置位。 此错误还包括 GMII 和半双工模式下的载波扩展错误。 扩展期间错误可以是更少/没有扩展或错误（rxd 0f）。
DE	2	置位时，该位表示接收到的帧具有非整数倍的字节（奇数半字节）。该位仅在 MII 模式下有效。
CE	1	置位时，该位表示接收帧上发生了循环冗余校验（CRC）错误。 该字段仅在设置了最后描述符（RDES0 8）时有效。
RxMAPCE	0	置位时，该位表示 Rx MAC 地址寄存器值（1 到 15）与帧目的地址字段匹配。 复位时，该位表示 Rx MAC 地址寄存器 0 值与目的地址字段匹配。 如果启用了完整校验和卸载引擎，则此位在设置时指示核心计算的 TCP，UDP 或 ICMP 校验和与接收的封装的 TCP，UDP 或 ICMP 段校验和字段不匹配。 当接收到的有效负载字节数与接收到的以太网帧中封装的 IPv4 或 IPv6 数据报的长度字段中指示的值不匹配时，也设置该位。

启用完整校验和卸载引擎（类型 2）时，位 5，7 和 0 的排列反映了讨论的条件。

位[5] FT	位[7] IPCCE	位[0] RxMAPCE	描述
0	0	0	IEEE 802.3 类型帧（长度字段值小于 0x0600）。
1	0	0	IPv4 / IPv6 类型帧，未检测到校验和错误。
1	0	1	检测到具有有效负载校验和错误（如针对 PCE 所述）的 IPv4 / IPv6 类型帧
1	1	0	检测到具有 IP 报头校验和错误（如针对 IPC CE 所述）的 IPv4 / IPv6 类型帧
1	1	1	检测到 IP 报头和有效负载校验和错误的 IPv4 / IPv6 类型帧。
0	0	1	IPv4 / IPv6 类型帧，由于不支持的有效负载，没有 IP 头校验和错误并且有效负载

			检查被绕过。
0	1	1	类型既不是 IPv4 也不是 IPv6 的类型（检测引擎完全绕过校验）。
0	1	0	保留

■ **接收描述符 1（RDES1）**

RDES1 包含缓冲区大小和控制描述符链/环的其他位。

位	域	描述
31	DIOC	置 1 时，该位防止为该描述符指向的缓冲区中结束的接收帧设置状态寄存器的 RI（CSR5 [6]）位。 反过来，由于该帧的 RI，这会禁止中断到主机的断言。
30:26	Reserved	保留
25	RER	置位时，该位指示描述符列表到达其最终描述符。 DMA 返回列表的基地址，创建描述符环。
24	RCH	置位时，该位指示描述符中的第二个地址是下一个描述符地址而不是第二个缓冲区地址。 当 RDES1[24]置位时，RBS2（RDES1 [21]-[11]）是一个不关心的值。 RDES1 [25]优先于 RDES1 [24]。
23:22	Reserved	保留
21:11	RBS2	这些位表示第二个数据缓冲区大小，以字节为单位 缓冲区大小必须是 16 的倍数，具体取决于总线宽度（128），即使 RDES3（缓冲区 2 地址指针）的值未与总线宽度对齐也是如此。 在缓冲区大小不是 16 的倍数的情况下，结果行为是不确定的。 如果设置了 RDES1[24]，则该字段无效。
10:0	RBS1	以字节为单位表示第一个数据缓冲区大小 缓冲区大小必须是 16 的倍数，具体取决于总线宽度（128），即使 RDES2（缓冲区 1 地址指针）的值未对齐。 在缓冲区大小不是 16 的倍数的情况下，结果行为是不确定的。 如果此字段为 0，则 DMA 忽略此缓冲区并使用缓冲区 2 或下一个描述符，具体取决于 RCH 的值（位[24]）。

■ **接收描述符 2（RDES2）**

RDES2 包含指向描述符中第一个数据缓冲区的地址指针。

位	域	描述
31:0	B1AP	这些位指示缓冲区 1 的物理地址。除以下条件外，缓冲区地址对齐没有限制：当 RDES2 值用于存储帧的起始时，DMA 使用配置的值来生成地址。 请注意，DMA 在帧开始传输期间执行写操作，RDES2 [3：0]位为 0，但帧数据按实际缓冲区地址指针移位。 如果地址指针指向存储帧的中间或最后部分的缓冲区，则 DMA 忽略 RDES2 [3：0]（对应于总线宽度 128）。

■ **接收描述符 3（RDES3）**

RDES3 包含地址指针，该指针指向描述符中的第二个数据缓冲区或下一个描述符。

位	字段	描述
31:0	B2AP	当使用描述符环结构时，这些位指示缓冲器 2 的物理地址。 如果第二个地址链接（RDES1 [24]）位置 1，则该地址包含指向下一个描述符所在物理存储器的指针。 如果设置 RDES1 [24]，则缓冲区（下一个描述符）地址指针必须与总线宽度对齐（RDES3 [3:0] = 0，对应于 128 的总线宽度。 内部忽略 LSB。）但是，当 RDES1[24]复位时，RDES3 值没有限制，除了以下条件：当 RDES3 值用于存储时，DMA 使用配置的值生成缓冲区地址 框架的开始。 如果地址指针指向存储帧的中间或最后部分的缓冲区，则 DMA 忽略 RDES3 [3:0]（对应于 128 的总线宽度）。

5.2.6.1.2 发送描述符

描述符地址必须与使用的总线宽度对齐（/128）。采用 32 位数据总线的小端模式下的发送描述符格式。

■ **发送描述符 0（TDES0）**

TDES0 包含发送的帧状态和描述符所有权信息。

位	字段	描述
31	OWN	置位时，该位表示描述符归 DMA 所有。 该位复位时，该位表示描述符归主机所有。 DMA 在完成帧传输或在描述符中分配的缓冲区为空时清除该位。 应该在设置了属于同一帧的所有后续描述符之后设置帧的第一描述符的所有权位。 这避免了在获取描述符和驱动程序设置所有权位之间可能存在竞争条件。
30:18	reserved	保留
17	TTSS	该状态位指示已捕获相应发送帧的时间戳。 当该位置 1 时，TDES2 和 TDES3 具有为发送帧捕获的时间戳值。 仅当设置了描述符中的最后一个段控制位（TDES1 30）时，该字段才有效。 该位仅在启用 IEEE1588 时间戳功能时有效; 否则，保留。
16	IHE	置位时，该位表示校验和卸载引擎检测到 IP 头错误，因此没有修改任何校验和插入的传输帧。 该位仅在启用完整校验和卸载时有效。否则，保留。
15	ES	表示以下位的逻辑或： TDES0 16：IP 标头错误 TDES0 14：Jabber 超时 TDES0 13：帧刷新 TDES0 12：有效负载校验和错误 TDES0 11：承运人的损失 TDES0 10：没有运营商 TDES0 9：后期冲突 TDES0 8：过度冲突 TDES0 2：过度延期 TDES0 1：下溢错误
14	JT	置位时，该位表示 GMAC 发送器经历了 jabber 超时。 仅当未设置 GMAC 配置寄存器 JD 位时才设置该位。
13	FF	置位时，该位表示由于 CPU 给出的 SW 刷新命令，DMA / MTL 刷新了帧。

12	PCE	此位置 1 时，表示 Checksum Offload 引擎发生故障，未将任何校验和插入封装的 TCP，UDP 或 ICMP 有效负载。 这种失败可能是由于 IP 报头有效载荷长度字段所指示的字节不足，或者 MTL 在存储转发模式下开始将帧转发到 MAC 发送器而尚未计算校验和。 仅当发送 FIFO 深度小于正在发送的以太网帧的长度时，才会出现第二个错误情况：为了避免死锁，即使在存储转发模式下，MTL 也会在 FIFO 满时开始转发帧。 在配置期间未启用完整校验和卸载引擎时，此位保留。
11	LC	当置位时，该位指示在帧传输期间发生载波丢失（即，在帧传输期间 gmii_crs_i 信号在一个或多个传输时钟周期内无效）。 这仅适用于无冲突传输的帧以及 GMAC 以半双工模式运行时的帧。
10	NC	置位时，该位指示在传输期间未形成 PHY 的载波侦听信号。
9	LC	置位时，该位表示由于冲突窗口之后发生冲突导致帧传输中止（64 字节时间包括 MII 模式中的前导码和 512 字节时间，包括 GMII 模式中的前导码和载波扩展）。 如果设置了下溢错误，则无效。
8	EC	置位时，该位表示在尝试发送当前帧时，在 16 次连续冲突后中止传输。 如果 GMAC 配置寄存器中的 DR（禁止重试）位置 1，则在第一次冲突后置位该位，并中止帧的传输。
7	VF	置位时，该位表示发送的帧是 VLAN 类型的帧。
6:3	CC	该 4 位计数器值表示在发送帧之前发生的冲突的数量。 当设置过度冲突位（TDES0 [8]）时，计数无效。
2	ED	置位时，如果延迟检查位设置为高电平，则该位表示由于过度延迟超过 24288 位时间（在 1000 Mbps 模式下为 155680 位，或在 Jumbo 帧使能模式下）传输已结束 GMAC 控制寄存器。
1	UF	置位时，该位指示 GMAC 中止帧，因为数据从主机存储器到达较晚。下溢错误表示 DMA 在发送帧时遇到空的发送缓冲区。 传输过程进入暂停状态并设置传输下溢（寄存器 5 [5]）和发送中断（寄存器 5 [0]）。
0	DB	置位时，该位表示由于载波的存在，GMAC 在传输之前推迟。 该

		位仅在半双工模式下有效。

■ **发送描述符 1（TDES1）**

TDES1 包含缓冲区大小和控制描述符链/环和正在传输的帧的其他位。

位	字段	描述
31	IC	置位时，该位在发送当前帧后设置发送中断。
30	LS	置位时，该位表示缓冲区包含帧的最后一段。 设置此位时， TBS1：发送缓冲区 1 大小或 TBS2：发送缓冲区 2 大小字段应具有非零值。
29	FS	置位时，该位表示缓冲区包含帧的第一段。
28:27	CIC	这些位控制在以太网帧中插入校验和，封装 TCP，UDP 或 ICMP IPv4 或 IPv6 如下所述。 2'b00：什么都不做。 校验引擎被绕过 2'b01：插入 IPv4 标头校验和。 当帧封装 IPv4 数据报时，使用此值插入 IPv4 标头校验和。 2'b10：插入 TCP / UDP / ICMP 校验和。 校验和仅通过 TCP，UDP 或 ICMP 段计算，并且假定 TCP，UDP 或 ICMP 伪报头校验和存在于相应的输入帧校验和字段中。 如果封装的数据报符合 IPv4，则还会插入 IPv4 头校验和。 2'b11：插入在此引擎中完全计算的 TCP / UDP / ICMP 校验和。 换句话说，TCP，UDP 或 ICMP 伪标头包含在校验和计算中，并且对应校验和字段的输入帧具有全零值。 如果封装的数据报符合 IPv4，则还会插入 IPv4 标头校验和。 检验引擎检测 TCP，UDP 或 ICMP 段是封装在 IPv4 还是 IPv6 中，并相应地处理其数据。
26	DC	置位时，GMAC 不会将循环冗余校验（CRC）附加到发送帧的末尾。 这仅在第一段（TDES1[29]）时有效。
25	TER	置位时，该位指示描述符列表到达其最终描述符。 返回列表的基址，创建描述符环。

24	TCH	置位时，该位指示描述符中的第二个地址是下一个描述符地址而不是第二个缓冲区地址。 当 TDES1 [24]置位时，TBS2（TDES1 [21:11]）不关心值。 TDES1 [25]优先于 TDES1 [24]。
23	DP	设置后，GMAC 不会自动向短于 64 字节的帧添加填充。当该位复位时，DMA 会自动将填充和 CRC 添加到短于 64 字节的帧中，并且尽管存在 DC（TDES1[26]）位的状态，仍会添加 CRC 字段。 仅在设置了第一个段（TDES1 [29]）时才有效。
22	TTSE	置 1 时，该位使能描述符引用的发送帧的 IEEE1588 硬件时间戳。 仅当第一段控制位（TDES1 [29]）置 1 时，该字段才有效。
21:11	TBS2	这些位以字节为单位指示第二个数据缓冲区。 如果设置了 TDES1 [24]，则该字段无效。
10:0	TBS1	这些位表示第一个数据缓冲区的字节大小。 如果此字段为 0，则 DMA 忽略此缓冲区并使用 Buffer 2 或下一个描述符取决于 TCH 的值（位 [24]）。

■ **发送描述符 2（TDES2）**

TDES2 包含指向描述符的第一个缓冲区的地址指针。

位	字段	描述
31:0	B1AP	这些位指示缓冲区 1 的物理地址。缓冲区地址对齐没有限制。

■ **发送描述符 3(TDES3)**

TDES3 包含指向描述符的第二个缓冲区或下一个描述符的地址指针。

位	字段	描述
31:0	B2AP	当使用描述符环结构时，表示缓冲区 2 的物理地址。 如果第二个地址链接（TDES1 [24]）位置 1，则该地址包含指向下一个描述符所在物理存储器的指针。 仅当 TDES1 [24]置 1 时，缓冲区地址指针必须与总线宽度对齐。

5.2.6.2 增强描述符

备用（或增强）描述符结构可以具有 8 个 DWORDS（32 个字节）而不是 4 个 DWORDS，如在普通描述符格式的情况下。 当启用 IEEE 1588-2008 高级时间戳功能或 AV 功能时，备用（或增强）描述符结构是唯一受支持的格式。 备

用描述符结构的特征是：

普通描述符结构允许最多 2048 字节的数据缓冲区。已实现替代描述符结构以支持最大 8 KB 的缓冲区（对于巨型帧有用）。

在 TDES0，TDES1，RDES0（高级时间戳或 IPC 完全卸载配置）和 RDES1 中重新分配控制和状态位。

选择"高级"时，发送描述符将时间戳存储在 TDES6 和 TDES7 中时间戳。

该接收描述符结构还用于存储扩展状态（RDES4）和时间戳（RDES6 和 RDES7）选择高级时间戳功能或 IPC 完全卸载时。

选择备用描述符模式并启用时间戳功能时，软件需要为每个描述符分配 32 字节（8 个 DWORDS）的内存。 如果未启用时间戳或接收 IPC 引擎，则不需要扩展描述符，SW 可以使用默认大小为 16 字节的备用描述符。 使用寄存器 0（总线模式寄存器）的第 7 位（ATDS：备用描述符大小），还需要为此更改配置内核。

如果选择备用描述符而没有时间戳或全 IPC 功能，则描述符大小始终为 4 DWORD（DES0-DES3）。

5.2.6.2.1 发送描述符

发送描述符结构如图 5-3 所示。 应用软件必须在描述符初始化期间对控制位 TDES0 3120 进行编程。 当 DMA 更新描述符时，它会回写除 OWN 位（它清除）之外的所有控制位并更新状态位[19：0]。 发送器描述符字 0（TDES0）到字 3（TDES3）的内容分别在后面给出。

通过提前时间戳支持，可以通过设置"TTSE：发送时间戳启用"（TDES0 的第 25 位）为给定帧启用要采用的时间戳的快照。 当描述符被关闭时（即，当 OWN 位被清除时），时间戳被写入 TDES6 和 TDES7。 这由状态位"TTSS：发送时间戳状态"（TDES0 的第 17 位）指示。

当启用高级时间戳或 IPC 卸载（类型 2）功能时，SW 应设置 DMA 总线模式寄存器 7，以便 DMA 以扩展的描述符大小运行。 当该控制位复位时，TDES4-TDES7 描述符空间无效。

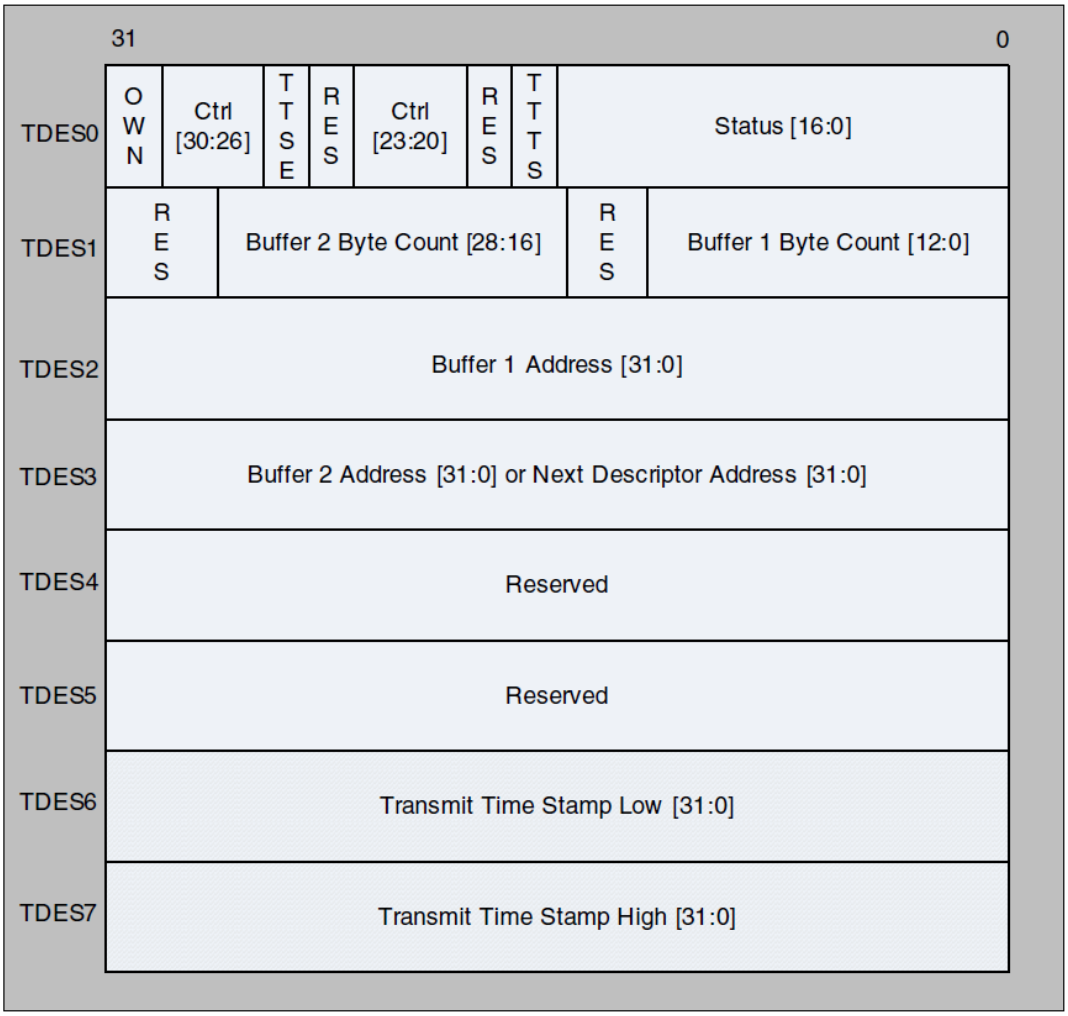

图 5-3 发送描述符结构

■ **发送描述符字 0（TDES0）**

位	字段	描述
31	OWN	置位时，该位表示描述符归 DMA 所有。 重置此位时，表示描述符归主机所有。 DMA 完成帧传输或完全读取描述符中分配的缓冲区时，DMA 清除此位。 必须在设置了属于同一帧的所有后续描述符之后设置帧第一描述符的所有权位。 这避免了在获取描述符和驱动程序设置所有权位之间可能存在竞争条件。
30	IC	置位时，该位在发送当前帧后设置发送中断。
29	LS	置位时，该位表示缓冲区包含帧的最后一段。该位置 1 时，TDES1 中的 TBS1：发送缓冲区 1 大小或 TBS2：发送缓冲区 2 大小字段应具有非零值。
28	FS	置位时，该位表示缓冲区包含帧的第一段。

27	DC	当该位置 1 时，GMAC 不会在发送帧的末尾附加循环冗余校验（CRC）。 仅在设置了第一个段（TDES0 [28]）时才有效。
26	DP	设置后，GMAC 不会自动向短于 64 字节的帧添加填充。 当该位复位时，DMA 会自动将填充和 CRC 添加到短于 64 字节的帧中，并且尽管存在 DC（TDES0 [27]）位的状态，仍会添加 CRC 字段。仅在设置了第一个段（TDES0 [28]）时才有效。
25	TTSE	置 1 时，该位使能描述符引用的发送帧的 IEEE1588 硬件时间戳。该字段仅在第一段控制位（TDES0 [28]）置 1 时有效。
24	reserved	保留
23:22	CIC	这些位控制校验和计算和插入。 位编码如下所示。 2b00：校验和插入已禁用。 2b01：仅启用 IP 标头校验和计算和插入。 2b10：启用 IP 报头校验和和有效负载校验和计算和插入，但不在硬件中计算伪报头校验和。 2b11：启用 IP 报头校验和校验和计算和插入，并在硬件中计算伪报头校验和。
21	TER	置位时，该位指示描述符列表到达其最终描述符。 DMA 返回列表的基址，创建描述符环。
20	TCH	置位时，该位指示描述符中的第二个地址是下一个描述符地址而不是第二个缓冲区地址。 当设置 TDES0 [20]时，TBS2（TDES1 [28:16]）是一个不关心的值。 TDES0 [21]优先于 TDES0 [20]。
19:18	Reserved	保留
17	TTSS	该字段用作状态位以指示针对所描述的发送帧捕获时间戳。 当该位置 1 时，TDES2 和 TDES3 具有为发送帧捕获的时间戳值。该字段仅在设置描述符控制位（TDES0 [29]）时有效。
16	IHE	置位时，该位指示 GMAC 发送器检测到 IP 数据报报头中的错误。发送器检查 IPv4 数据包中的报头长度与从应用程序接收的报头

		字节数，并指示错误状态是否存在不匹配。 对于 IPv6 帧，如果主报头长度不是 40 字节，则报告报头错误。 此外，IPv4 或 IPv6 帧的以太网长度/类型字段值必须与随数据包一起接收的 IP 报头版本匹配。 对于 IPv4 帧，如果"标头长度"字段的值小于 0x5，则还会指示错误状态。
15	ES	表示以下位的逻辑或： TDES0 14：Jabber 超时 TDES0 13：帧刷新 TDES0 11：承运人的损失 TDES0 10：没有运营商 TDES0 9：后期冲突 TDES0 8：过度冲突 TDES0 2：过度延期 TDES0 1：下溢错误 TDES0 16：IP 标头错误 TDES0 12：IP 有效载荷错误
14	JT	置位时，该位表示 GMAC 发送器经历了 jabber 超时。 仅当未设置 GMAC 配置寄存器 JD 位时才设置该位。
13	FF	置位时，该位表示由于 CPU 给出的软件 Flush 命令，DMA / MTL 刷新了帧。
12	IPE	置位时，该位指示 GMAC 发送器检测到 TCP，UDP 或 ICMP IP 数据报有效负载中的错误。 发送器检查在 IPv4 或 IPv6 报头中接收的有效负载长度与从应用程序接收的 TCP，UDP 或 ICMP 数据包字节的实际数量，并在出现不匹配时发出错误状态。
11	LC	当置位时，该位指示在帧传输期间发生载波丢失（即，在帧传输期间 gmii_crs_i 信号在一个或多个传输时钟周期内无效）。 当 GMAC 以半双工模式运行时，这仅适用于无冲突传输的帧。
10	NC	置位时，该位表示由于冲突窗口之后发生冲突而导致帧传输中止

		（在 GII 模式下为 64 字节时，包括前导码，在 MII 模式下为 512 字节时，包括前导码和载波扩展，在 GMII 模式下）。 如果设置了下溢错误位，则该位无效。
9	LC	置位时，该位表示由于冲突窗口之后发生冲突而导致帧传输中止（在 GII 模式下为 64 字节时，包括前导码，在 MII 模式下为 512 字节时，包括前导码和载波扩展，在 GMII 模式下）。 如果设置了下溢错误位，则该位无效。
8	EC	置位时，该位表示在尝试发送当前帧时，在 16 次连续冲突后中止传输。 如果 GMAC 配置寄存器中的 DR（禁止重试）位置 1，则在第一次冲突后置位该位，并中止帧的传输。
7	VF	置位时，该位表示发送的帧是 VLAN 类型的帧。
6:3	CC	这些状态位表示在帧发送之前发生的冲突数。 当设置过多冲突位（TDES0 8]时，此计数无效。 核心仅在半双工模式下更新此状态字段。
2	ED	置位时，如果 GMAC 控制寄存器中的延迟检查位为 2，则该位表示由于过度延迟超过 24288 位时间（在 1000-Mbps 模式下为 155680 位，或者如果启用了 Jumbo 帧），传输已结束 高高举起。
1	UF	置位时，该位指示 GMAC 中止帧，因为数据从主机存储器到达较晚。 下溢错误表示 DMA 在发送帧时遇到空发送缓冲区。 传输过程进入挂起状态并设置传输下溢。
0	DB	置位时，该位表示由于载波的存在，GMAC 在传输之前推迟。 该位仅在半双工模式下有效。

■ **发送描述符 1 （TDES1）**

位	字段	描述
31:29	reserved	保留
28:16	TBS2	这些位表示第二个数据缓冲区大小，以字节为单位 如果设置了 TDES0[20]，则该字段无效。

15:13	Reserved	保留
12:0	TBS1	这些位指示第一个数据缓冲区字节大小（以字节为单位）。 如果此字段为 0，则 DMA 忽略此缓冲区并使用缓冲区 2 或下一个描述符，具体取决于 TCH 的值（TDES0 [20]）。

■ **发送描述符 2（TDES2）**

位	字段	描述
31:0	B1AP	这些位指示缓冲区 1 的物理地址。缓冲区地址对齐没有限制。

■ **发送描述符 6（TDES6）**

位	字段定义	描述
31:0	TTSL	该字段由 DMA 更新，其中对应的发送帧捕获的时间戳的最低有效 32 位。 仅当设置了描述符中的最后一个位（LS）并设置了时间戳状态（TTSS）位时，此字段才具有时间戳。

■ **传输描述符 7（TDES7）**

位	字段定义	描述
31:0	TTSH	该字段由 DMA 更新，其中对应的接收帧捕获的时间戳的最高 32 位。 仅当描述符中的 Last Segment 位（LS）置位且时间戳状态（TTSS）位置 1 时，此字段才具有时间戳。

5.2.6.2.2 接收描述符

接收描述符的结构如下图 5-4 所示。 这可以有 32 个字节的描述符数据（8 DWORDs）选择高级时间戳或 IPC 完全卸载功能时。

当启用这些功能中的任何一个时，SW 应设置 DMA 总线模式寄存器 7，以便 DMA 以扩展的描述符大小运行。当该控制位复位时，RDES0 [7]和 RDES0 [0]始终清零，RDES4-RDES7 描述符空间无效。

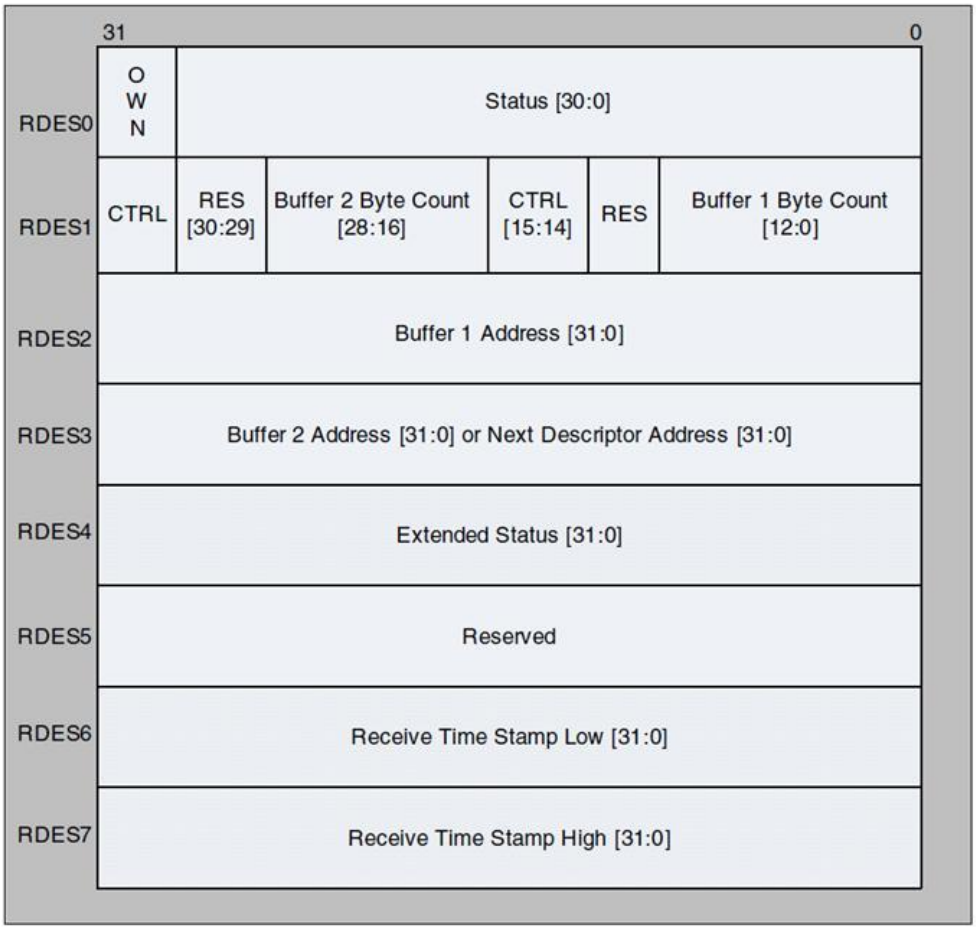

图 5-4 接收描述符结构

■ **接收描述符字段 0（RDES0）**

位	字段定义	描述
31	OWN	置位时，该位表示描述符归 GMAC 子系统的 DMA 所有。 该位复位时，该位表示描述符归主机所有。 DMA 在完成帧接收或与此描述符关联的缓冲区已满时清除此位。
30	AFM	置位时，该位表示 GMAC 中目的地址滤波器出现故障的帧。
29:16	FL	这些位指示传输到主机存储器（包括 CRC）的接收帧的字节长度。当最后描述符（RDES0 [8]）置 1 且描述符错误（RDES0 [14]）或溢出错误位复位时，该字段有效。 当启用 IP 校验和计算（类型 1）并且接收的帧不是 MAC 控制帧时，帧长度还包括附加到以太网帧的两个字节。
15	ES	表示以下位的逻辑或： RDES0 1：CRC 错误

		RDES0 3：接收错误 RDES0 4：看门狗超时 RDES0 6：后期冲突 RDES0 7：巨型报文 RDES4 4：3：IP 标头/有效负载错误 RDES0 11：溢出错误 RDES0 14：描述符错误 该字段仅在设置了最后描述符（RDES0 8）时有效。
14	DE	置位时，该位表示由不适合当前描述符缓冲区的帧引起的帧截断，并且 DMA 不拥有下一个描述符。 框架被截断。 该字段仅在设置了最后描述符（RDES0[8]）时有效。
13	SAF	置位时，该位表示帧的源地址字段未通过 GMAC 核心中的源地址过滤器。
12	LE	置位时，该位表示接收到的帧的实际长度以及长度/类型字段不匹配。 仅当帧类型（RDES0 [5]）位复位时，该位才有效。
11	OE	置位时，该位表示由于 MTL 中的缓冲区溢出而导致接收帧被损坏。
10	VLAN	置位时，该位指示该描述符指向的帧是由 GMAC 标记的 VLAN 帧。
9	FS	置位时，该位指示该描述符包含帧的第一个缓冲区。 如果第一个缓冲区的大小为 0，则第二个缓冲区包含帧的开头。 如果第二个缓冲区的大小也是 0，则下一个描述符包含帧的开头。
8	LS	置位时，该位指示此描述符指向的缓冲区是帧的最后一个缓冲区。
7	TAICE	当存在高级时间戳功能时，如果置位，该位表示时间戳的快照写入描述符字 6（RDES6）和 7（RDES7）。仅在最后描述符位（RDES0 [8]）置 1 时有效。 选择 IP 校验和引擎（类型 1）时，该位置 1 时表示核心计算的 16 位 IPv4 报头校验和与接收到的校验和字节不匹配。 否则，该位置位时表示巨帧状态。 巨帧大于 1518 字节（或

		Jumbo 时，正常帧为 1522 字节，大于 9018 字节（VLAN 为 9022 字节）帧处理已启用。
6	LC	置位时，该位表示在半双工模式下接收帧时发生了晚期冲突。
5	FT	置位时，该位表示接收帧是以太网类型帧（LT 字段大于或等于 16h0600）。 当该位复位时，表示接收到的帧是 IEEE802.3 帧。 该位对于小于 14 字节的 Runt 帧无效。
4	RWT	置 1 时，该位表示接收看门狗定时器在接收当前帧时已过期，当前帧在看门狗超时后被截断。
3	RE	置位时，该位指示 gmii_rxer_i 信号被置位，而 gmii_rxdv_i 在帧接收期间被置位。 此错误还包括 GMII 和半双工模式下的载波扩展错误。 扩展期间错误可以是更少/没有扩展或错误（rxd 0f）。
2	DE	置位时，该位表示接收到的帧具有非整数倍的字节（奇数半字节）。该位仅在 MII 模式下有效。
1	CE	置位时，该位表示接收帧上发生了循环冗余校验（CRC）错误。 该字段仅在设置了最后描述符（RDES0 8）时有效。
0	ESA/RMA	当存在高级时间戳或 IP 校验和卸载（类型 2）时，该位置 1 时表示扩展状态在描述符字 4（RDES4）中可用。 仅在最后描述符位（RDES0 [8]）置 1 时有效。 如果未选择"高级时间戳功能"或"IPC 完全卸载"，则该位指示 Rx MAC 地址状态。 置位时，该位表示 Rx MAC 地址寄存器值（1 到 15）与帧目的地址字段匹配。 复位时，该位表示 Rx MAC 地址寄存器 0 值与目的地址字段匹配。

■ **接收描述符字段 1（RDES1）**

位	字段定义	描述
31	DIC	置 1 时，该位防止为接收帧设置状态寄存器 RI 位（CSR5 6），该帧在此描述符指示的缓冲区中结束。 反过来，由于该帧的 RI，这会禁止中断到主机的断言。
30:29	Reserved	保留
28:16	RBS2	这些位指示第二个数据缓冲区大小（以字节为单位）。 缓冲区

		大小必须是 48 或 16 的倍数，具体取决于总线宽度（分别为 3264 或 128），即使 RDES3（缓冲区 2 地址指针）的值未与总线宽度对齐也是如此。 如果缓冲区大小不是 48 或 16 的适当倍数，则结果行为是未定义的。
15	RER	置位时，该位指示描述符列表到达其最终描述符。 DMA 返回列表的基址，创建描述符环。
14	RCH	置位时，该位指示描述符中的第二个地址是下一个描述符地址而不是第二个缓冲区地址。 设置此位时，RBS2（RDES1 [28:16]）是一个不关心的值。 RDES1 [15]优先于 RDES1 [14]。
13	Reserved	保留
12:0	RBS1	以字节为单位表示第一个数据缓冲区大小 缓冲区大小必须是 48 或 16 的倍数，具体取决于总线宽度（3264 或 128），即使 RDES2（缓冲区 1 地址指针）的值未对齐也是如此。 当缓冲区大小不是 48 或 16 的倍数时，结果行为是不确定的。 如果该字段为 0，则 DMA 忽略此缓冲区并使用缓冲区 2 或下一个描述符，具体取决于 RCH 的值（位 14）。

■ **接收描述符字段 2（RDES2）**

位	字段定义	描述
31:0	B1AP	这些位指示缓冲区 1 的物理地址。除以下条件外，缓冲区地址对齐没有限制：当 RDES2 值用于存储帧的起始时，DMA 使用配置的值来生成地址。 请注意，DMA 在帧开始传输期间执行写操作，RDES2 [3:0]位为 0，但帧数据按实际缓冲区地址指针移位。 如果地址指针指向存储帧的中间或最后部分的缓冲区，则 DMA 忽略 RDES2 [3:0]（对应于总线宽度 128）。

■ **接收描述符字段 3（RDES3）**

位	字段定义	描述
31:0	B2AP	当使用描述符环结构时，这些位指示缓冲器 2 的物理地址。 如果第二个地址链接（RDES1[24]）位置 1，则该地址包含指向下一个描述符所在物理存储器的指针。

		如果设置了 RDES1[24]，则缓冲区（下一个描述符）地址指针必须与总线宽度对齐（RDES3 [3:0] = 0，对应总线宽度 128。但是，当 RDES1 [24]复位时，RDES3 值没有限制，除了以下条件 ：当 RDES3 值用于存储帧的开始时，DMA 使用配置的值来生成缓冲区地址。 如果地址指针指向存储帧的中间或最后部分的缓冲区，则 DMA 忽略 RDES3 [3:0]（对应于 128 的总线宽度）。

■ **接收描述符字段 4（RDES4）**

位	字段定义	描述
31:21	Reserved	保留
20:18	VTPV	这些位为接收的数据包中的VLAN标记提供用户值。仅当RDES4位[16]和[17]置 1 时，这些位才有效。
17	ATPR	置位时，该位表示接收到 AV 标记的数据包。 否则，该位指示接收到未标记的 AV 分组。 当位 16（AV 包接收）置位时，该位有效。
16	AVPR	置位时，该位表示接收到 AV 包。
15	Reserved	保留
14	TD	置位时，该位指示该帧捕获的时间戳，但由于溢出而在 MTL RxFIFO 中被丢弃。 仅当您选择"高级时间戳"功能时，此位才可用。 否则，该位保留。
13	PTPV	置位时，该位表示接收的 PTP 消息具有 IEEE 1588 版本 2 格式。重置时，它具有版本 1 格式。 仅在消息类型为非零时才有效。 仅当选择了高级时间戳功能时，此位才可用，否则保留。
12	PTPFT	置位时，该位表示 PTP 消息直接通过以太网发送。 如果未设置此位且消息类型不为零，则表示 PTP 消息是通过 UDP-IPv4 或 UDP-IPv6 发送的。 可以从位 6 和 7 获取有关 IPv4 或 IPv6 的信息。仅当选择了高级时间戳功能时，此位才可用。
11:8		对这些位进行编码以给出所接收消息的类型。 0000：未收到 PTP 消息 0001：SYNC（所有时钟类型）

		0010：Follow_Up（所有时钟类型） 0011：Delay_Req（所有时钟类型） 0100：Delay_Resp（所有时钟类型） 0101：Pdelay_Req（对等透明时钟） 0110：Pdelay_Resp（对等透明时钟） 0111：Pdelay_Resp_Follow_Up（在对等透明时钟中） 1000：宣布 1001：管理 1010：信令 1011-1110：保留 1111：具有保留消息类型的 PTP 分组 仅当您选择"高级时间戳"功能时，这些位才有效。
7	IPv6PR	置位时，该位表示接收的数据包是 IPv6 数据包。
6	IPv4PR	置位时，该位表示接收的数据包是 IPv4 数据包。
5	IPCB	置位时，该位表示绕过校验和卸载引擎。
4	IPPE	置位时，该位指示核心计算的 16 位 IP 有效负载校验和（即 TCP，UDP 或 ICMP 校验和）与接收段中的相应校验和字段不匹配。 当 TCP，UDP 或 ICMP 段长度与 IP 标头字段中的有效负载长度值不匹配时，也会设置它。
3	IPHE	置位时，该位指示核心计算的 16 位 IPv4 报头校验和与接收的校验和字节不匹配，或者 IP 数据报版本与以太网类型值不一致。
2:0	IPPT	这些位指示由接收校验和卸载引擎（COE）处理的 IP 数据报中封装的有效负载类型。 如果由于 IP 报头错误或分段 IP 而未处理 IP 数据报有效负载，则 COE 还将这些位设置为 2'b00。 3'b000：未知或未处理 IP 有效负载 3'b001：UDP 3'b010：TCP 3'b011：ICMP 3b1xx：保留

■ **接收描述符字段 6（RDES6）**

位	字段定义	描述
31:0	RTSL	该字段由 DMA 更新，其中对应的接收帧捕获的时间戳的最低有效 32 位。 该字段仅由 DMA 更新，用于接收帧的最后一个描述符，由最后描述符状态位（RDES0 [8]）指示。

■ **接收描述符字段 7（RDES7）**

位	字段定义	描述
31:0	RTSH	该字段由 DMA 更新，其中对应的接收帧捕获的时间戳的最高 32 位。 该字段仅由 DMA 更新，用于接收帧的最后一个描述符，由最后描述符状态位（RDES0 [8]）指示。

5.3 QSPI

FT-2000/4 包含 1 个 QSPI（Quad Serial Peripheral Interface）接口控制器。

5.3.1 操作说明

QSPI Flash 的命令操作可以分为两种形式：直接地址访问、寄存器端口访问。直接地址访问是一种自动触发方式，通过读写控制信号和地址按照寄存器配置的命令对 Flash array 进行数据读写操作；寄存器端口访问是通过对控制内部的寄存器按照命令协议形式进行参数设置，然后控制器通过低位数据端口寄存器读写来触发向 Flash 发送指令（一般是非 flash array 读写的指令）。

■ QSPI-Flash 操作

以操作 flash 为例：

- Flash 读（pbf 默认设置，一般不需要修改）

1、写 FLASH_CAPACITY 寄存器，设置正确需要读取的 flash 容量及个数。

2、写 RD_CFG 寄存器，仅修改[31:24]位，填入对应 flash 的读取命令。

3、直接读取 0x0 ---- 0x1FFFFFFF 的地址空间，flash 地址空间的内容按照顺序一一映射到此地址空间。

- Flash 擦除

1、写 CMD_PORT 寄存器，仅修改[31:24]，填入对应 flash 写使能命令，并且置位 bit22。

2、写 LD_PORT，填入 0x1,发送命令。

3、写 CMD_PORT 寄存器，仅修改[31:24]，填入对应 flash 擦除命令，并且置位 bit22 和 bit15。

4、写 ADDR_PORT 寄存器，填入需要擦除的地址。

5、写 LD_PORT，填入 0x1,发送命令。

- Flash 写

1、写 CMD_PORT 寄存器，仅修改[31:24]，填入对应 flash 写使能命令，并且置位 bit22。

2、写 LD_PORT，填入 0x1,发送命令。

3、写 WR_CFG 寄存器，仅修改[31:24]，填入对应 flash 编程命令，并且置位 bit9 及 bit3。

4、直接写 0x0 ---- 0x1FFFFFFF 的地址空间，可以连续写入，且必须保证每次写都是四字节写。

5、写 LD_PORT，填入 0x1,发送命令。

6、写 WR_CFG 寄存器为 0 关闭写使能。

- QSPI 模式切换

1、QSPI 模式切换主要在于将 flash 通过写命令的方式切换为 QSPI 模式。

2、将 flash 切换为 QSPI 模式后，写 RD_CFG 的[22: 20]位，切换成 QSPI 模式 4-4-4 即可让控制器发出 QSPI 模式访问。

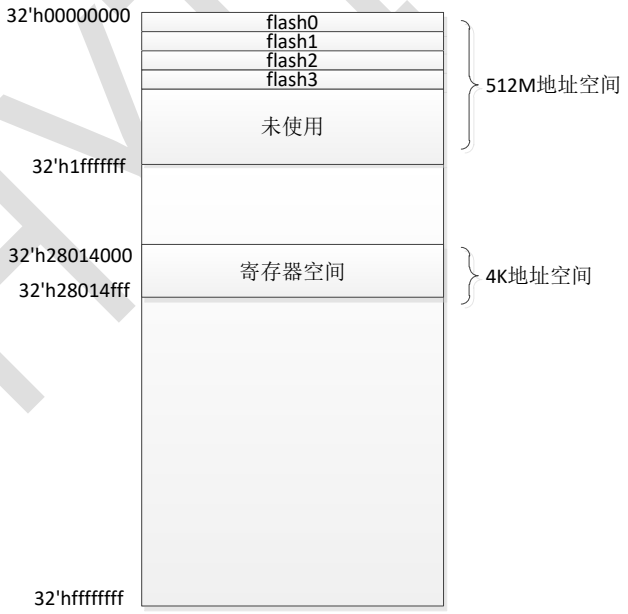

图 5-5 QSPI 在系统中的地址映射

5.3.2 寄存器说明

5.3.2.1 寄存器基地址

表 5-12 QSPI 控制器地址空间

名称	基地址
直接地址可访问的数据空间	0~0x1fff_ffff
QSPI 配置寄存器基址	0x000_28014000

5.3.2.2 寄存器列表

表 5-13 QSPI 寄存器列表

偏移	标识	说明
0x000	FLASH_CAPACITY	FLASH 容量设置寄存器
0x004	RD_CFG	地址访问读配置寄存器
0x008	WR_CFG	地址访问写配置寄存器
0x00C	FLUSH_REG	写缓冲 flush 寄存器
0x010	CMD_PORT	命令端口寄存器
0x014	ADDR_PORT	地址端口寄存器
0x018	HD_PORT	高位数据端口寄存器
0x1C	LD_PORT	低位数据端口寄存器
0x20	FUN_SET	CS 设置寄存器
0x24	WIP_RD	WIP 读取设置寄存器
0x28	WP_REG	WP 寄存器
0x2C	MODE_REG	Mode 设置寄存器

5.3.2.3 FLASH_CAPACITY

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 0x0fff_ffff 大小，最大支持连接四个相同容量的 Flash。

域	位	初值	R/W	说明
reserved	31:5	0	rw	未使用
flash_num	4:3	0	rw	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	0	rw	支持容量大小 000：4MB 001：8MB 010：16MB 011：32MB 100：64MB 101：128MB 110：256MB 111：4MB

5.3.2.4 RD_CFG

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	初值	R/W	说明
rd_cmd	31:24	0	rw	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	0	rw	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	0	rw	读命令传输协议 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
rd_addr_sel	19	0	rw	0：3byte 地址 1：4byte 地址
rd_latency	18	0	rw	1 表示读数据有延迟
mode_byte	17	0	rw	1 表示有命令修饰符
cmd_sign	16:9	0	rw	当 mode_byte=1 该域有效，存储命令修饰符
dummy	8:4	0	rw	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	0	rw	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	0	rw	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。

5.3.2.5 WR_CFG

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	初值	R/W	说明
wr_cmd	31:24	0	rw	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	0	rw	未使用
wr_wait	9	0	rw	1 表示此命令在 Flash 需要一定的执行时间
wr_through	8	0	rw	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	0	rw	编程命令传输协议 000：1-1-1 001：1-1-2 010：1-1-4 011：1-2-2 100：1-4-4 101：2-2-2 110：4-4-4 111：1-1-1
wr_addr_sel	4	0	rw	0：3byte 地址 1：4byte 地址
wr_mode	3	0	rw	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	0	rw	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。

5.3.2.6 FLUSH_REG

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	初值	R/W	说明
flush	0	0	wo	写 1 将产生 flush 操作

5.3.2.7 CMD_PORT

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	初值	R/W	说明
cmd	31:24	0	rw	指令字节
reserved	23	0	rw	未使用
wait	22	0	rw	1 表示此命令在 Flash 需要一定的执行时间
through	21	0	rw	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	0	rw	选择需要发命令的芯片
transfer	18:16	0	rw	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	0	rw	1 表示有地址传输
latency	14	0	rw	1 表示读数据有延迟
data_transfer	13	0	rw	1 表示有数据传输
addr_sel	12	0	rw	0：3byte 地址 1：4byte 地址
dummy	11:7	0	rw	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	0	rw	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	0	rw	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	0	rw	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。

5.3.2.8 ADDR_PORT

通过寄存器端口访问时设置的地址。

域	位	初值	R/W	说明

addr	31:0	0	rw	地址

5.3.2.9 HD_PORT

通过寄存器端口访问时的高 4 字节数据。

域	位	初值	R/W	说明
data	31:0	0	rw	最大支持 4 字节有效数据

5.3.2.10 LD_PORT

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	初值	R/W	说明
data	31:0	0	rw	与高位数据端口寄存器一起最大支持 8 字节有效数据

5.3.2.11 FUN_SET

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

域	位	初值	R/W	说明
cs_hold	31:24	5	rw	有效保持时间 计算公式：time=pclk_cycle*(cs_hold+1)+sck_cycle/2
cs_setup	23:16	5	rw	有效建立时间 计算公式：time=pclk_cycle*(cs_setup+1)+sck_cycle/2
cs_delay	15:0	40	rw	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) (wp_en=0) time = pclk_cycle*(max{cs_delay,wp_hold}+ wp_setup-cs_hold+4) (wp_en=1)

5.3.2.12 WIP_RD

设置查询 Flash 状态寄存器的命令。

域	位	初值	R/W	说明
w_cmd	31:24	5	rw	读状态寄存器指令字节
reserved	23:5	0	rw	未使用
w_transfer	4:3	0	rw	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	0	rw	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。

5.3.2.13 WP_REG

1-1-1 命令协议下控制 wp 输出。

域	位	初值	R/W	说明
reserved	31:1	0	rw	未使用
wp_en	17	0	rw	wp 使能信号，1 使能硬件写保护

wp	16	0	rw	写 0 wp/io2 输出 0，写 1 wp/io2 输出 1
wp_hold	15:8	80	rw	wp 保持时间 计算公式：time=pclk_cycle*(wp_hold+1)
wp_setup	7:0	15	rw	wp 建立时间 计算公式：time=pclk_cycle*(wp_setup+1)

5.3.2.14 MODE_REG

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	初值	R/W	说明
reserved	31:16	0	rw	未使用
mode_valid	15:8	255	rw	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=0xf0
mode	7:0	255	rw	寄存相关 Flash 模式位的值

5.4 SPI

FT-2000/4 包含 2 个通用 SPI 接口，仅作为通用 SPI master 使用，用于发起和控制传输，以及产生时钟，外部最多挂载 4 个 SPI slave。

5.4.1 寄存器说明

5.4.1.1 基地址

表 5-14 SPI 基地址

名称	基地址
SPIM0	0x2800_c000
SPIM1	0x2801_3000

5.4.1.2 寄存器列表

寄存器	偏移（0x）	description
CTRLR0	00	控制寄存器 0
CTRLR1	04	控制寄存器 1
SSIENR	08	SPI 使能寄存器
MWCR	0c	Microwire 控制
SER	10	从机使能寄存器
BAUDR	14	波特率选择寄存器
TXFTLR	18	发送 FIFO 阈值寄存器
RXFTLR	1c	接收 FIFO 阈值寄存器
TXFLR	20	发送 FIFO 等级寄存器
RXFLR	24	接收 FIFO 等级寄存器
SR	28	状态寄存器
IMR	2c	中断屏蔽寄存器
RISR	34	中断状态寄存器
TXOICR	38	清除发送 FIFO 溢出中断寄存器
RXOICR	3c	清除接收 FIFO 溢出中断寄存器
RXUICR	40	清除发送 FIFO 下溢中断寄存器
MSTICR	44	清除多主机争用中断寄存器
ICR	48	中断清除寄存器
DMACR	4c	DMA 控制寄存器
DMATDLR	50	DMA 发送数据等级寄存器
DMARDLR	54	DMA 接收数据等级寄存器
IDR	58	识别码
DR	60-ec	数据寄存器
RX_SAMPLE_DLY	fc	接收数据延时寄存器

5.4.1.3 CTRLR0

域	位	读写	复位值	描述
CFS	15:12	WR	0	数据大小控制位。 用于 Microwire 模式中。
SRL	11	WR	0	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 存器正常模式 1 式器测试模式 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.再此模式下，从机无法生成这些信号因为没有可循环的对象。
SLV_OE	10	WR	0	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	WR	0	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	WR	0	串行时钟极性。设置为 Motorola SPI 时有效。 0 – serial clock 为低时不活跃 1 不活跃 erial clock 为高时不活跃
SCPH	6	WR	0	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	WR	0	帧格式，选择传输模式。00 器的输入。en=0)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。0 rs Microwire 11 rs Microwire=0)序访问。命令端口寄存器设置使
DFS	3:0	WR	7	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。

5.4.1.4 CTRLR1

域	位	读写	复位值	描述
NDF	15:0	WR	0	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。

5.4.1.5 SSIENR

域	位	读写	复位值	描述
SSI_EN	0	WR	0	SPI 使能。 启用和禁用所有 SPI 操作。

5.4.1.6 MWCR

域	位	读写	复位值	描述

MHS	2	WR	0	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 协议的“议的“owire 行主机设握手接口。在启用之前清除 SR 寄存器中的 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	WR	0	Microwire 控制位。指定使用 Microwire 串行协议时数据字符的方向。当这个为置 0 时表示从外部串行设备接收数据，置 1 时，数据发送到外部串行设备 。
MW MOD	0	WR	0	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

5.4.1.7 SER

域	位	读写	复位值	描述
SER	3:0	WR	0	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。 1: 选择 0: 不选择

5.4.1.8 BAUDR

域	位	读写	复位值	描述
SCKDV	15:0	WR	0	SSI 时钟除法器。 SCKDV 为 2 ~ 65534 之间的任何偶数值。例如: Fssi_clk = 3.6864MHz，SCKDV =2 Fsclk_out = 3.6864/2 = 1.8432MHz

5.4.1.9 TXFTLR

域	位	读写	复位值	描述
TFT	2:0	WR	0	发送 FIFO 阙值。控制发送 FIFO 触发中断的阈值。FIFO 深度可以在范围 2-256 之间配置。

5.4.1.10 RXFTLR

域	位	读写	复位值	描述
RFT	3:0	WR	0	接收 FIFO 阙值. 控制接收 FIFO 触发中断的阈值。FIFO 深度范围 2-256 之间配置。

5.4.1.11 TXFLR

域	位	读写	复位值	描述
TXTFL	3:0	RO	0	发送 FIFO 等级，包含传输 FIFO 中的有效数据项的数目

5.4.1.12 RXFLR

域	位	读写	复位值	描述
RXTFL	3:0	RO	0	接收 FIFO 等级，包含传输 FIFO 中的有效数据项的数目

5.4.1.13 SR

域	位	读写	复位值	描述
DCOL	6	RO	0	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	RO	0	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	RO	0	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	RO	0	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	RO	0	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,位域不请求中断 0 – 传输 FIFO 不空 1 – 传输 FIFO 为空
TFNF	1	RO	0	发送 FIFO 不满时。当传输 FIFO 包含一个或多个有空位时设置，以满时清除。 0 – 发送 FIFO 满 1 – 发送 FIFO 不满
BUSY	0	RO	0	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

5.4.1.14 ISR

域	位	读写	复位值	描述
MSTIS	5	RO	0	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	RO	0	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	RO	0	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	RO	0	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	RO	0	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	RO	0	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。

5.4.1.15 RISR

域	位	读写	复位值	描述
MSTIR	5	RO	0	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 1 = 优先屏蔽 ssi_mst_intr 后中断活动
RXFIR	4	RO	0	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	RO	0	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	RO	0	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	RO	0	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	RO	0	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

5.4.1.16 TXOICR

域	位	读写	复位值	描述
TXOICR	0	RO	0	清除传输 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。

5.4.1.17 RXOICR

域	位	读写	复位值	描述
RXOICR	0	RO	0	清除传输 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。

5.4.1.18 RXUICR

域	位	读写	复位值	描述
RXUICR	0	RO	0	清除传输 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_txo_intr 中断，写入无效.

5.4.1.19 MSTICR

域	位	读写	复位值	描述
MSTICR	0	RO	0	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_txo_intr 中断，写入无效。

5.4.1.20 ICR

域	位	读写	复位值	描述
ICR	0	RO	0	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 the ssi_mst_intr 中断. 写到这个寄存器是没有效果的。

5.4.1.21 DMACR

域	位	读写	复位值	描述
TDMAE	1	WR	0	DMA 发送使能。该位可以启用/禁用 FIFO 的 DMA 传输通道。
RDMAE	0	WR	0	DMA 接收使能，该位可以启用/禁用 FIFO DMA 传输通道

5.4.1.22 DMATDLR

域	位	读写	复位值	描述
DMATDL	2:0	WR	3'h0	发送数据等级。该位控制 DMA 请求的发送等级。当 dma_tx_req s 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。

5.4.1.23 DMARDLR

域	位	读写	复位值	描述

DMARDL	2:0	WR	3'h0	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。

5.4.1.24 IDR

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

5.4.1.25 DR

域	位	读写	复位值	描述
DR	15:0	WR	0x0	数据寄存器.写入此寄存器时必须对数据进行右对齐,读取数据时自动的右对齐。 读 = 接收 FIFO 写 = 发送 FIFO

5.4.1.26 RX_SAMPLE_DLY

域	位	读写	复位值	描述
RSD	7:0	WR	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。

5.5 UART

FT-2000/4 包含 4 个 UART（Universal Asynchronous Receiver/Transmitter，通用异步接收/发送装置）控制器。其中，三个 3 线制接口，只提供输入、输出和中断线，一个 9 线制接口，提供握手相关接口。

5.5.1 操作说明

5.5.1.1 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：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：divisor=uartclk/(16*波特率)

例如：uartclk 为 48MHZ，波特率为 115200。

divisor=(48*10^6)/(16*115200)=26.042

整数位 BRDI=26，小数位 BRDf=0.042。

m=integer((0.04215200)=26.0

因此向 0x24（UARTIBRD）地址写入 0x1a(26 转换成十六进制)，向 0x28（UARTFBRD）地址写入 0x3。

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 0x44（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 loppback、hardware flow

control 相关功能，向 0x30（UARTCR）地址写入 0x0301。

5.5.1.2 发送数据操作流程

使用轮询方式：

1、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。

2、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：

1、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。

2、写入数据：向 0x00（UARTDR）写入数据，当发送 FIFO 里的数据大于设置的 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。

5.5.1.3 接收数据操作流程

使用轮询方式：

1、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。

2、读出数据：读 0x00（UARTDR）数据寄存器的数据。

使用中断方式：

1、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。

2、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。

5.5.1.4 Flow control 相关操作

1、RTS flow control：向 0x30（UARTCR）地址的 bit[14]写 1，使能 RTS flow control。

2、CTS flow control：向 0x30（UARTCR）地址的 bit[15]写 1，使能 CTS flow control。

5.5.2 寄存器说明

5.5.2.1 基地址

表 5-15 UART 寄存器基地址

名称	基地址
UART0	0x000_28000000
UART1	0x000_28001000
UART2	0x000_28002000
UART3	0x000_28003000

5.5.2.2 寄存器列表

表 5-16 UART 寄存器列表

名称	偏移地址	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
	0x008~0x014	Reserved
UARTFR	0x018	标志寄存器
	0x01c	Reserved

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	波特率小数值配置寄存器
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

5.5.2.3 UARTDR

域	位	读写	复位值	描述
	15:12	RW	0	保留
OE	11	RW	0	溢出错误。如果接收到数据并且接收的 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	0	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 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	0	奇偶校验错误。当设置为 1，表示接收的数据字符的奇偶性与 UARTLCR_H 寄存器中的 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	0	帧错误。此设置为 1 是，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	0	接收（读）数据。 传输（写）数据。

5.5.2.4 UARTRSR/UARTECR

域	位	读写	复位值	描述
	7:0	RW	0	写入该寄存器将清除帧、奇偶校验、中断和溢出错误。可写入任意值。
	7:4	RW	0	保留，读取值不可预测
OE	3	RW	0	溢出错误。如果接收到数据并且此时 FIFO 已满，此位设置设置为 1。该位通过写入 UARTECR 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	0	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 UARTECR 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	0	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 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 顶部的字符相关联。
FE	0	RW	0	帧错误。此设置为 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。

5.5.2.5 UARTFR

域	位	读写	复位值	描述
	15:9	RO	0	保留，读取为零。
RI	8	RO	0	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。 也就是说，当 nUARTRI 低电位时该位为 1。
TXFE	7	RO	1	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。

				该位不表示如果发送移位寄存器有数据。
RXFF	6	RO	0	接收 FIFO 满。该位取决与 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 RXFT 位。
TXFF	5	RO	0	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	1	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	0	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	0	数据载波检测，该位表示 UART 数据载波、nUARTDCD、调制解调状态输入。也就是说，当 nUARTDCD 为低电平时该为为 1。
DSR	1	RO	1	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	1	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。

5.5.2.6 UARTILPR

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。

5.5.2.7 UARTIBRD

域	位	读写	复位值	描述
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。

5.5.2.8 UARTFBRD

域	位	读写	复位值	描述
BAUD DIVFRAC	5 : 0	RW	6'h0	波特率计算因子的小数值。在复位时这些位清除为 0。

5.5.2.9 UARTLCR_H

域	位	读写	复位值	描述
	15:8			保留，读取为零。
SPS	7	RW	0	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 bits b10 = 7 bits b01 = 6 bits b00 = 5 bits
FEN	4	RW	0	FIFO 使能位。 0 = FIFOs 是禁用的（字符模式），也就是说，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	0	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	0	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。 当 PEN 位禁用奇偶校验检测和生成时该位无效。
PEN	1	RW	0	奇偶校验使能。 0 = 奇偶校验被禁止。

				1 = 奇偶校验使能。
BRK	0	RW	0	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。

5.5.2.10 UARTCR

域	位	读写	复位值	描述
CTSEn	15	RW	0	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	0	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 不满师接收它的请求。
Out2	13	RW	0	该位是 nUATROut2 调制解调状态输出。也就是说，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	0	该位是 nUARTOut1 调制解调状态输出，也就是说，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	0	该位是 UART 发送请求、nUARTRTS、调制解调状态输出 。也就是说，当该位被编程为 1 时，nUARTRTS 为低。
DTR	10	RW	0	该位是 UART 数据传输准备完成、nUARTDTR、调制解调状态输出。也就是说，当该位被编程为 1 时，nUARTDTR 为低。
RXE	9	RW	1	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与 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	1	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。
LBE	7	RW	0	回环使能位。如果该位、SIREN 位和 UARTTCR 的 SIRTEST 位为 1 时，反转 nSIROUT 路径，并将其输入 SIRIN 路径中。测试寄存器中的 SIRTEST 必须设置为 1，以覆盖正常的 half_duplex SIR 操作。在回环结束后 SIRTEST 必须清除为 0。此功能减少了系统测试期间所需的外部耦合数量。 如果该位设置为 1，SIRTEST 位设置为 0，将 UARTTXD 路径输入 UARTRXD 路径中。 在 SIR 或 UART 模式下，当此为被设置时，将调制解调的输出输入到调制解调的输入中。 在重置时该位清除为 0，以此禁用回环功能。
	6:3			保留，读取为零。
SIRLP	2	RW	0	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。如果在位被设置为 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	0	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 传输或调制解调状态输入无效。 This bit has no effect if the UARTEN bit disables the UART。 如果 UARTEN 位禁用 UART 那么该位无效。
UARTEN	0	RW	0	UART 使能端： 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。

5.5.2.11 UARTIFLS

域	位	读写	复位值	描述
RXIFLSEL	5:3	RW	3'h2	接收 FIFO 中断的阈值选择。中断的触发点如下： b000 = 接收 FIFO ≥ 1/8 full b001 = 接收 FIFO≥ 1/4 full b010 = 接收 FIFO ≥ 1/2 full b011 = 接收 FIFO ≥ 3/4 full b100 = 接收 FIFO ≥ 7/8 full b101-b111 = 保留
TXIFLSEL	2:0	RW	3'h2	发送 FIFO 中断的阈值选择。中断的触发点如下： b000 = 发送 FIFO ≤ 1/8 full b001 = 发送 FIFO ≤ 1/4 full b010 = 发送 FIFO ≤ 1/2 full b011 = 发送 FIFO ≤ 3/4 full

				b100 = 发送 FIFO ≤ 7/8 full b101-b111 = 保留

5.5.2.12 UARTIMSC

域	位	读写	复位值	描述
	15:11			保留，读取为零。
OEIM	10	RW	0	溢出错误中断屏蔽。读取返回 UARTOEINTR 中断的当前屏蔽。 写 1 时，设置 UARTOEINTR 中断屏蔽。写 0 时，清除屏蔽。
BEIM	9	RW	0	突发错误中断屏蔽。读操作返回当前的 UARTBEINTR 中断屏蔽。 写 1 时，设置 UARTBEINTR 中断屏蔽。写 0 时，清除屏蔽。
PEIM	8	RW	0	奇偶校验错误中断屏蔽。读操作返回当前的 UARTPEINTR 中断屏蔽。 写 1 时，设置 UARTPEINTR 中断屏蔽。写 0 时，清除屏蔽。
FEIM	7	RW	0	帧错误中断屏蔽。读操作返回当前的 UARTFEINTR 中断屏蔽。 写 1 时，设置 UARTFEINTR 中断屏蔽。写 0 时，清除屏蔽。
RTIM	6	RW	0	接收超时中断屏蔽。读操作返回当前的 UARTRTINTR 中断屏蔽。 写 1 时，设置 UARTRTINTR 中断屏蔽。写 0 时，清除屏蔽。
TXIM	5	RW	0	发送中断屏蔽。读操作返回当前的 UARTTXINTR 中断屏蔽。

				写 1 时，设置 UARTTXINTR 中断屏蔽。写 0 时，清除屏蔽。
RXIM	4	RW	0	接收中断屏蔽。读操作返回当前的 UARTRXINTR 中断屏蔽。 写 1 时，设置 UARTRXINTR 中断屏蔽。写 0 时，清除屏蔽。
DSRMIM	3	RW	0	nUARTDSR 调制解调中断屏蔽。读操作返回当前的 UARTDSRINTR 中断屏蔽。 写 1 时，设置 UARTDSTINTR 中断屏蔽。写 0 时，清除屏蔽。
DCDMIM	2	RW	0	nUARTDCD 调制解调中断屏蔽。读操作返回当前的 UARTDCDINTR 中断屏蔽。 写 1 时，设置 UARTDCDINTR 中断屏蔽。写 0 时，清除屏蔽。
CTSMIM	1	RW	0	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	0	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。

5.5.2.13 UARTRIS

域	位	读写	复位值	描述
	15:11			保留，读取为零。
OERIS	10	RO	0	溢出错误中断状态。反馈 UARTOEINTR 中断的原始中断状态。

BERIS	9	RO	0	突发错误中断状态。反馈 UARTBEINTR 中断的原始中断状态。
PERIS	8	RO	0	奇偶校验错误中断状态。反馈 UARTPEINTR 中断的原始中断状态。
FERIS	7	RO	0	帧错误中断状态。反馈 UARTFEINTR 中断的原始中断状态。
RTRIS	6	RO	0	接收超时中断状态。反馈 UATRTRINTR 中断的原始中断状态。
TXRIS	5	RO	0	发送中断状态。反馈 UARTTXINTR 中断的原始中断状态。
RXRIS	4	RO	0	接收中断状态。反馈 UARTTXINTR 中断的原始中断状态。
DSRRMIS	3	RO	0	nUARTDSR 调制解调中断状态。反馈 UARTDSRINTR 中断的原始中断状态。
DCDRMIS	2	RO	1	nUARTDCD 调制解调中断状态。反馈 UARTDCDINTR 中断的原始中断状态。
CTSRMIS	1	RO	0	nUARTCTS 调制解调中断状态。反馈 UARTCTSINTR 中断的原始中断状态。
RIRMIS	0	RO	1	nUARTRI 调制解调中断状态。反馈 UARTRIINTR 中断的原始中断状态。

5.5.2.14 UARTMIS

域	位	读写	复位值	描述
	15:11			保留，读取为零。
OEMIS	10	RO	0	溢出错误屏蔽中断状态。返回 UARTORINTR 里的中断屏蔽状态。
BEMIS	9	RO	0	突发错误屏蔽中断状态。反馈 UARTBEINTR 的中断屏蔽状态。

PEMIS	8	RO	0	奇偶校验错误屏蔽中断状态。反馈 UARTREINTR 的中断屏蔽状态。
FEMIS	7	RO	0	帧错误屏蔽中断状态。反馈 UARTRRINTR 里的中断屏蔽状态。
RTMIS	6	RO	0	接收超时屏蔽中断状态。反馈 UARTRTINTR 里的中断屏蔽状态。
TXMIS	5	RO	0	发送屏蔽中断状态。反馈 UARTTXINTR 里的中断屏蔽状态。
RXMIS	4	RO	0	接收屏蔽中断状态。反馈 UARTRXINTR 里的中断屏蔽状态。
DSRMMIS	3	RO	0	nUARTDSR 调制解调屏蔽中断状态。反馈 UARTDSRINTR 里的中断屏蔽状态。
DCDMMIS	2	RO	0	nUARTDCD 调制解调屏蔽中断状态。反馈 UARTDCDINTR 里的中断屏蔽状态。
CTSMMIS	1	RO	0	nUARTCTS 调制解调屏蔽中断状态。反馈 UARTCTSINTR 里的中断屏蔽状态。
RIMMIS	0	RO	0	nUARTRI 调制解调屏蔽中断状态。反馈 UARTIINTR 里的中断屏蔽状态。

5.5.2.15 UARTICR

域	位	读写	复位值	描述
	15:11	WO		保留，读取为零。
OEIC	10	WO		溢出错误中断清除。清除 UARTOEINTR 中断。
BEIC	9	WO		突发错误中断清除。清除 UARTBEINTR 中断。
PEIC	8	WO		奇偶校验错误中断清除。清除 UARTPEINTR 中断。

FEIC	7	WO		帧错误中断清除。清除 UARTFEINTR 中断。
RTIC	6	WO		接收超时中断清除。清除 UARTRTINTR 中断。
TXIC	5	WO		传输中断清除。清除 UARTTXINTR 中断。
RXIC	4	WO		接收中断清除。清除 UARTRXINTR 中断。
DSRMIC	3	WO		nUARTDSR 调制解调中断清除。清除 UARTDSRINTR 中断。
DCDMIC	2	WO		nUARTDCD 调制解调中断清除。清除 UARTDCDINTR 中断。
CTSMIC	1	WO		nUARTCTS 调制解调中断清除。清除 UARTCTSINTR 中断。
RIMIC	0	WO		nUARTRI 调制解调中断清除。清除 UARTRIINTR 中断。

5.5.2.16 UARTDMACR

域	位	读写	复位值	描述
	15：3			保留，读取为零
DMAONERR	2	RW	0	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 UARTRXDMABREQ 被禁用。
TXDMAE	1	RW	0	发送 DMA 使能端。如果该位设置为 1，传输 FIFO 的 DMA 启用。
RXDMAE	0	RW	0	接收 DMA 使能位。如果该位设置为 1，接收 FIFO 的 DMA 启用。

5.6 LPC 接口

LPC（Low Pin Count）接口是 Intel 推出的一种低 IO 数目的外设接口，用于连接 SuperIO、Flash 等 LPC 设备。

5.6.1 操作说明

使用 LPC 功能前，需要首先配置相关 PAD 复用寄存器，将对应 PAD 配置到对应 LPC 功能上，才可使用 LPC 功能。

1、判断想要访问设备的接口访问类型，然后通过配置 APB 接口地址的设备类型寄存器发送对应的 IO、FIRMWARE Memory、Memory、DMA 请求。

2、通过 nu_serirq_config[31]判断是 4 字节读取还是单字节读取。

5.6.2 寄存器说明

5.6.2.1 基地址

表 5-17 LPC 寄存器基地址

名称	基地址
LPC	0x000_20000000

5.6.2.2 寄存器列表

表 5-18 LPC 寄存器列表

寄存器	偏移地址	说明
INT_APB_SPCE_CONF	0x7FF_FFFC	配置 APB 接口地址的设备类型寄存器
REG_LONG_TIMEOUT	0x7FF_FFF8	长等待超时控制寄存器
INT_STATE	0x7FF_FFF4	中断状态寄存器
CLR_INT	0x7FF_FFF0	中断清除寄存器

FIRMWARE_LEN_CONFG	0x7FF_FFEC	firmware memory 类型的报文长度配置寄存器（暂未使用）
NU_SERIRQ_CONFIG	0x7FF_FFE8	配置寄存器
CLK_LPC_RSTN_O	0x7FF_FFE4	控制外设复位寄存器
FIRMWR_ID_CONF_STRTB	0x7FF_FFE0	firmware 设备 ID 选择配置寄存器
DMA_CHNNLNU_CONF	0x7FF_FFDC	DMA 设备 ID 配置寄存器
INT_MASK	0x7FF_FFD8	中断屏蔽寄存器
START_CYCLE	0x7FF_FFD4	配置启动周期寄存器
MEM_HIGHBIT_ADDR	0x7FF_FFD0	Memory 访问的高 5 位地址

5.6.2.3 INT_APB_SPCE_CONF

域	位	读写	复位值	描述
Int_apb_spce_conf	7:0	RW	8'b11100100	配置 APB 接口地址的设备类型

5.6.2.4 REG_LONG_TIMEOUT

域	位	读写	复位值	描述
reg_long_timeout	31:0	RW	0	长等待超时控制寄存器

5.6.2.5 INT_STATE

域	位	读写	复位值	描述
int_state	31:0	RO	0	中断状态（串行中断)bit29~0:串行中断，bit30DMA 请求中断

5.6.2.6 CLR_INT

域	位	读写	复位值	描述
clr_int	31:0	RW	0	清除中断寄存器

5.6.2.7 NU_SERIRQ_CONFIG

域	位	读写	复位值	描述
nu_serirq_config	31:0	RW	0x8000_0000	配置寄存器（bit31：针对读数据每次读 4 bytes 数据使能标志（1’b1：读 1byte）； bit1~0:起始周期配置(2’b11：8；2’b10:6；否则 4，默认 4 clk)， bit2：串行中断模式配置默认连续模式 默认为连续模式）， bit3~4：支持的串行中断设备数量（2‘b01 代表 32 否则 16 默认 16）

5.6.2.8 CLK_LPC_RSTN_O

域	位	读写	复位值	描述
clk_lpc_rstn_o	0	RW	0	控制外设复位寄存器（默认 0）

5.6.2.9 FIRMWR_ID_CONF_STRTB

域	位	读写	复位值	描述
Firmwr_id_conf_strtb	2:0	RW	0	firmware memory 设备 ID 选择配置寄存器: 3’b001 22：19 作为 ID，

				3’b010 23：20 作为 ID, 3’b011 24：21 作为 ID, 默认 30：27 作为 ID

5.6.2.10 DMA_CHNNLNU_CONF

域	位	读写	复位值	描述
Dma_chnnlnu_conf	2:0	RW	0x6	DMA 设备 ID 配置寄存器

5.6.2.11 INT_MASK

域	位	读写	复位值	描述
int_mask	1:0	RW	0x3	中断屏蔽寄存器

5.6.2.12 START_CYCLE

域	位	读写	复位值	描述
start_cycle_reg	4:0	RW	0	配置启动周期

5.6.2.13 MEM_HIGHBIT_ADDR

域	位	读写	复位值	描述
mem_highbit_addr	4:0	RW	0	mem 访问的高 5bit 地址

5.7 I2C 接口

I2C（Inter-Integrated Circuit）是一种两线式串行总线，用于连接微控制器及其外围设备。

I2C 只要求两条线路：串行数据总线 SDA 和串行时钟线 SCL。I2C 标准定义了三种传输速度：Standard(100kb/s)，Fast(400kb/s)，Hign speed(3.4Mb/s)。连接在 I2C 总线的设备通过 SCL 和 SDA 进行通信，每个设备都通过一个特定的地址进行识别，并且可以作为一个发送器或者接收器。当设备向总线发送数据时是一个发送器；当设备接收来自总线的数据时是一个接收器。发送器需要产生时钟信号 SCL，控制数据的发送，和产生 START 和 STOP 状态。接收器需要根据发送器发送的 START 和 STOP 状态开始和结束数据的读写操作。接收器还需要在接收到数据后发送一个应答信号给发送器。

5.7.1 操作说明

5.7.1.1 配置为 master

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 i2c，配置寄存器 0x6c（IC_ENABLE）为 1

5.7.1.2 配置为 slave

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 i2c，配置寄存器 0x6c（IC_ENABLE）为 1

5.7.1.3 master 模式发送和接收数据流程

- 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。

- 接受数据：

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 1、2、3 步骤即可。

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。

5.7.1.4 slave 模式发送和接收数据流程

- 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。

- 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。

5.7.2 寄存器说明

5.7.2.1 基地址

表 5-19 I2C 寄存器基地址

名称	基地址
I2C0	0x000_28006000
I2C1	0x000_28007000
I2C2	0x000_28008000
I2C3	0x000_28009000

5.7.2.2 寄存器列表

表 5-20 I2C 寄存器列表

寄存器	偏移	说明
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
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_COMP_PARAM_1	0xf4	I2C 版本信息寄存器

5.7.2.3 IC_CON

域	位	读写	复位值	描述
	15:7		-	
IC_SLAVE_DISABLE	6	WR	1	I2C Slave 功能是否关闭的控制位。即在使用 I2 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。 0: 使能 I2C Slave 功能 1: 关闭 I2C Slave 功能
IC_RESTART_EN	5	WR	1	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。 0: 不支持 Restart 1: 支持 Restart 当设备不支持 RESTART 功能时，I2C 的 Master 控制器支持以下功能： •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。 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。
IC_10BITADDR_MASTER	4	RO	1	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。 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： 0: 7 位地址模式 1: 10 位地址模式
IC_10BITADDR_SLAVE	3	WR	1	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 0: 7 位地址模式。 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 位地址值进行比对，若两者一致则响应，若不一致则不响应。 1: 10 位地址模式。 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。
SPEED	2:1	WR	0x3	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。 1: 标准模式 (0 to 100 Kbit/s) 2: 快速模式 (≤ 400 Kbit/s) 3: 高速模式 (≤ 3.4 Mbit/s)
MASTER_MODE	0	WR	1	I2C Master 的使能位。 0: 关闭 master 功能 1: 使能 master 功能

5.7.2.4 IC_TAR

域	位	读写	复位值	描述
-	15:13			
IC_10BITADDR_	12	WR	1	选择工作在 I2C Master 时使用 7 位地址

MASTER				模式还是 10 位地址模式进行通信。 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	WR	0	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	WR	0	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	WR	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。

5.7.2.5 IC_SAR

域	位	读写	复位值	描述
	15:10			
IC_SAR	9:0		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。

5.7.2.6 IC_HS_MADDR

域	位	读写	复位值	描述
	15:3			
IC_HS_MAR	2:0	WR	1	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

5.7.2.7 IC_DATA_CMD

域	位	读写	复位值	描述
	15:11			
RESTART	10	WO	0	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	0	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 TxFIFO 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 和申请总线仲裁的方式开始一次新的通信。 •0 –不管 TxFIFO 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	0	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 In slave-receiver mode， this bit is a “don’t care”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	0	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。

5.7.2.8 IC_SS_SCL_HCNT

域	位	读写	复位值	描述
IC_SS_SCL_HCNT	15:0	WR	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。

5.7.2.9 IC_SS_SCL_LCNT

域	位	读写	复位值	描述
IC_SS_SCL_LCNT	15:0	WR	0x1d6	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 低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 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。

5.7.2.10 IC_FS_SCL_HCNT

域	位	读写	复位值	描述
IC_FS_SCL_LCNT	15:0	WR	0x82	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。 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。

5.7.2.11 IC_FS_SCL_LCNT

域	位	读写	复位值	描述
IC_FS_SCL_LCNT	15:0	WR	0x82	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。 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。

5.7.2.12 IC_HS_SCL_HCNT

域	位	读写	复位值	描述
IC_HS_SCL_HCNT	15:0	WR	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的负载情况。接 100pF 的负载时，高电平时间为 60ns;接 400pF 的负载时，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。

5.7.2.13 IC_HS_SCL_LCNT

域	位	读写	复位值	描述
IC_HS_SCL_LCN	15:0	WR	0x10	该寄存器必须在 I2C 总线传输之前进行

T				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的负载情况。接 100pF 的负载时，低电平时间为 160ns;接 400pF 的负载时，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。其他情况下的写操作无效。 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。

5.7.2.14 IC_INTR_STAT

域	位	读写	复位值	描述
	15:12			
R_GEN_CALL	11	RO	0	
R_START_DET	10	RO	0	
R_STOP_DET	9	RO	0	
R_ACTIVITY	8	RO	0	
R_RX_DONE	7	RO	0	
R_TX_ABRT	6	RO	0	

R_RD_REQ	5	RO	0	
R_TX_EMPTY	4	RO	0	
R_TX_OVER	3	RO	0	
R_RX_FULL	2	RO	0	
R_RX_OVER	1	RO	0	
R_RX_UNDER	0	RO	0	

5.7.2.15 IC_INTR_MASK

域	位	读写	复位值	描述
	15:12			
M_GEN_CALL	11	WR	1	M_GEN_CALL 中断事件标志屏蔽控制。置位时，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	WR	0	
M_STOP_DET	9	WR	0	
M_ACTIVITY	8	WR	0	
M_RX_DONE	7	WR	1	
M_TX_ABRT	6	WR	1	
M_RD_REQ	5	WR	1	
M_TX_EMPTY	4	WR	1	
M_TX_OVER	3	WR	1	
M_RX_FULL	2	WR	1	
M_RX_OVER	1	WR	1	
M_RX_UNDER	0	WR	1	

5.7.2.16 IC_RAW_INTR_STAT

域	位	读写	复位值	描述
	15:12			
GEN_CALL	11	RO	0	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	RO	0	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 模式还是 Slave 模式无关。
STOP_DET	9	RO	0	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	RO	0	此位表示 I2C 控制器的活动状态。 有 4 种方法可以清楚 ACTIVITY 标志： •关闭 DW_apb_i2c •读取 IC_CLR_ACTIVITY 寄存器 •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	RO	0	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	RO	0	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	RO	0	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	RO	0	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	RO	0	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	RO	0	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	RO	0	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	RO	0	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。

5.7.2.17 IC_RX_TL

域	位	读写	复位值	描述
	15:8			
RX_TL	7:0	WR	0x0	接收缓冲区满中断（RX_FULL）触发门限控制。有效范围 0~255，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， 255 表示接收缓冲区大于等于 256 时触发中断。

5.7.2.18 IC_TX_TL

域	位	读写	复位值	描述
	15:8			
TX_TL	7:0	WR	0x0	发送缓冲区满中断（TX_EMPTY）触发门限控制。有效范围 0~255，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， 255 表示发送缓冲区小于等于 255 时触发中断。

5.7.2.19 IC_CLR_INTR

域	位	读写	复位值	描述
	15:1			
CLR_INTR	0	RO	0	读取次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。

5.7.2.20 IC_CLR_RX_UNDER

域	位	读写	复位值	描述
	15:1			
CLR_RX_UNDER	0	RO	0	读取这个寄存器以清除 IC_RAW_INTR_STAT 寄存器的 RX_UNDER 中断 (bit 0)

5.7.2.21 IC_CLR_RX_OVER

域	位	读写	复位值	描述
	15:1			
CLR_RX_OVER	0	RO	0	读取这个寄存器以清除 IC_RAW_INTR_STAT 寄存器的 RX_UNDER 中断 (bit 1)

5.7.2.22 IC_CLR_TX_OVER

域	位	读写	复位值	描述
	15:1			
CLR_TX_OVER	0	RO	0	读取这个寄存器以清除 IC_RAW_INTR_STAT 寄存器的 RX_OVER 中断 (bit3)

5.7.2.23 IC_CLR_RD_REQ

域	位	读写	复位值	描述
	15:1			
CLR_RD_REQ	0	RO	0	读取这个寄存器以清除 IC_RAW_INTR_STAT 寄存器的 RD_REQ 中断 (bit5)

5.7.2.24 IC_CLR_TX_ABRT

域	位	读写	复位值	描述
	15:1			
CLR_TX_ABRT	0	RO	0	读这个寄存器来清除 IC_RAW_INTR_STAT register 和 IC_TX_ABRT_SOURCE register 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

5.7.2.25 IC_CLR_RX_DONE

域	位	读写	复位值	描述
	15:1			
CLR_RX_DONE	0	RO	0	读取这个寄存器以清除 IC_RAW_INTR_STAT 寄存器的 RX_DONE 中断 (bit7)

5.7.2.26 IC_CLR_ACTIVITY

域	位	读写	复位值	描述

	15:1			
CLR_ACTIVITY	0	RO	0	如果 I2C 不在处于活动状态，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。

5.7.2.27 IC_CLR_STOP_DET

域	位	读写	复位值	描述
	15:1			
CLR_STOP_DET	0	RO	0	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)

5.7.2.28 IC_CLR_START_DET

域	位	读写	复位值	描述
	15:1			
CLR_START_DET	0	RO	0	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)

5.7.2.29 IC_CLR_GEN_CALL

域	位	读写	复位值	描述
	15:1			
CLR_START_DET	0	RO	0	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)

5.7.2.30 IC_ENABLE

域	位	读写	复位值	描述
	15:1			
ENABLE	0	RO	0	I2C 控制器使能或关闭控制位。 0: 关闭 I2C 控制器功能 1: 使能 I2C 控制器功能 以下现象会在 I2C 控制器功能关闭时出现: • TXFIFO 和 RXFIFO 被刷新 • IC_INTR_STAT 寄存器中的状态保持不变。 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。

5.7.2.31 IC_STATUS

域	位	读写	复位值	描述
	31:7			
SLV_ACTIVITY	6	RO	0	Slave FSM 活动状态标志。Slave FSM(Slave Finite State Machine 不在 Idle 状态时被置位 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	RO	0	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 控制器的 Master 功能处于活动状态。
RFF	4	RO	0	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 未满 1: 接收 FIFO 全满
RFNE	3	RO	0	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	RO	1	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	RO	1	发送 FIFO 未满标志。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 已满 1: 发送 FIFO 未满
ACTIVITY	0	RO	0	I2C 控制器活动状态标志

5.7.2.32 IC_TXFLR

域	位	读写	复位值	描述
	[31:TX_ABW+1]			
TXFLR	[TX_ABW:0]	RO	0	发送 FIFO 阈值。发送 FIFO 中的有效数据量。

5.7.2.33 IC_RXFLR

域	位	读写	复位值	描述
	[31:RX_A			

	BW+1]			
RXFLR	[RX_ABW:0]	RO	0	接收 FIFO 阈值。接收 FIFO 中的有效数据量。

5.7.2.34 IC_SDA_HOLD

域	位	读写	复位值	描述
	31:16			
IC_SDA_HOLD	15:0	WR	0x1	设置所需的 SDA 保持时间以 ic_clk 周期为单位

5.7.2.35 IC_TX_ABRT_SOURCE

域	位	读写	复位值	描述
ABRT_SLVRD_INTX	15	RO	0	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	RO	0	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	RO	0	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	RO	0	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	RO	0	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	RO	0	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	RO	0	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	RO	0	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	RO	0	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	RO	0	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	RO	0	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_N	4	RO	0	1:DW_apb_i2c 在主模式下发送了一个

OACK				General Call 并且没有从机在总线上承认
ABRT_TXDATA_NOACK	3	RO	0	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，并没有收到来自远程从机的确认。Master-Transmitter
ABRT_10ADDR2_NOACK	2	RO	0	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	0	1：主设备处于 10 位地址模式，10 位地址的第以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	0	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。

5.7.2.36 IC_SLV_DATA_NACK_ONLY

域	位	读写	复位值	描述
	31:1			
NACK	0	WR	0	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，它将根据正常条件生 NACK/ACK 。 1 = 产生 NACK 0 = 产生正常条件的 NACK/ACK

5.7.2.37 IC_DMA_CR

域	位	读写	复位值	描述
	31:2			
TDMAE	1	WR	0	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	WR	0	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

5.7.2.38 IC_DMA_TDLR

域	位	读写	复位值	描述
	31:TX_ABW			
DMATDL	TX_ABW–1:0	WR	0	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。

5.7.2.39 IC_DMA_RDLR

域	位	读写	复位值	描述
	31:RX_ABW			
DMARDL	[RX_ABW–1:0]	WR	0	接收数据阈值。该位控制接收逻辑中的一个 DMA 请求的阈值。

5.7.2.40 IC_SDA_SETUP

域	位	读写	复位值	描述
	31:8			
SDA_SETUP	7:0	WR	0x64	SDA 建立。 建议如果所需延时为 1000ns,对于频率为 10 MHz 的 ic_clk，IC_SDA_SETUP 编程为 11 。IC_SDA_SETUP 必须以最小值 2 来编程。

5.7.2.41 IC_ACK_GENERAL_CALL

域	位	读写	复位值	描述
	31:1			
ACK_GEN_CALL	0	WR	1	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。

5.7.2.42 IC_ENABLE_STATUS

域	位	读写	复位值	描述
	31:3			
SLV_RX_DATA_LOST	2	RO	0	从机收到的数据丢失。
SLV_DISABLED_WHILE_BUSY	1	RO	0	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	RO	0	ic_en 状态。

5.7.2.43 IC_FS_SPKLEN

域	位	读写	复位值	描述
	31:8			
IC_FS_SPKLEN	7:0	WR	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。

5.7.2.44 IC_HS_SPKLEN

域	位	读写	复位值	描述
	31:8			
IC_HS_SPKLEN	7:0	WR	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，才能写如该寄存器，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。

5.7.2.45 IC_COMP_PARAM_1

域	位	读写	复位值	描述
	31:24			
TX_BUFFER_DEPTH	23:16	RO	0x0	IC_TX_BUFFER_DEPTH 0x00 = Reserved 0x01 = 2 0x02 = 3 ... 0xFF = 256
RX_BUFFER_DEPTH	15:8	RO	0x3	IC_RX_BUFFER_DEPTH 0x00 = Reserved 0x01 = 2 0x02 = 3 ... 0xFF = 256
ADD_ENCODED_PARAMS	7	RO	1	IC_ADD_ENCODED_PARAMS 0: 错误 1: 正确
HAS_DMA	6	RO	0	IC_HAS_DMA 0: 错误 1: 正确
INTR_IO	5	RO	1	IC_INTR_IO 0: 个体 1: 联合
HC_COUNT_VALUES	4	RO	0	IC_HC_COUNT_VALUES 0: 错误 1: 正确
MAX_SPEED_MO	3:2	RO	0x3	

DE				IC_MAX_SPEED_MODE 0x0 = 保留 0x1 = 标准 0x2 = 快速 0x3 = 高
APB_DATA_WIDTH	1:0	RO	0x2	APB_DATA_WIDTH 0x0 = 8 bits 0x1 = 16 bits 0x2 = 32 bits 0x3 = 保留

5.8 GPIO 接口

FT-2000/4 提供两个 GPIO（General Purpose Programming I/O）模块，每个 GPIO 模块有 16b 接口，每 8b 位一组。GPIO 可以控制外部 IO pad 的输入输出方向，当 IO pad 为输出时，内部寄存器中的数据输出到片外；当 IO pad 为输入时，pad 上的数据被锁存到内部寄存器。其中 A 组的 8b 接口支持复用为外部中断信号。

5.8.1 操作说明

5.8.1.1 作为数据传输信号

首先，需要将对应 pin 脚的 PAD 复用类型设置为 GPIO。

- 写数据

  1）配置方向寄存器（gpio_swporta_ddr 、gpio_swportb_ddr）为输出（写 1）

  2）数据写入寄存器（gpio_swporta_dr、gpio_swportb_dr）

- 读数据

  1）配置方向寄存器（gpio_swporta_ddr 、gpio_swportb_ddr）为输入（写 0）

  2）从寄存器（gpio_ext_porta、gpio_ext_portb）中读出数据

5.8.1.2 复用为中断信号

1）配置方向寄存器（gpio_swporta_ddr ）为输入（写 0）

2）写中断屏蔽寄存器（gpio_intmask）为 0

3）配置中断类型寄存器（gpio_inttype_level）

4）配置中断极性寄存器（gpio_int_polarity）

5）使能中断，寄存器（gpio_inten）写 1

5.8.2 寄存器说明

5.8.2.1 基地址

表 5-21 GPIO 控制寄存器基地址

名称	基地址
GPIO 0 控制寄存器	0x000_28004000
GPIO 1 控制寄存器	0x000_28005000

5.8.2.2 寄存器列表

表 5-22 GPIO 寄存器列表

寄存器	偏移	描述
GPIO_SWPORTA_DR	0x00	A 组端口输出寄存器
GPIO_SWPORTA_DDR	0x04	A 组端口方向控制寄存器
GPIO_EXT_PORTA	0x08	A 组端口输入寄存器
GPIO_SWPORTB_DR	0x0c	B 组端口输出寄存器
GPIO_SWPORTB_DDR	0x10	B 组端口方向控制寄存器
GPIO_EXT_PORTB	0x14	B 组端口输入寄存器
GPIO_INTEN	0x18	A 组端口中断使能寄存器
GPIO_INTMASK	0x1c	A 组端口中断屏蔽寄存器
GPIO_INTTYPE_LEVEL	0x20	A 组端口中断等级寄存器
GPIO_INT_POLARITY	0x24	A 组端口中断极性寄存器

GPIO_INTSTATUS	0x28	A 组端口中断状态寄存器
GPIO_RAW_INTSTATUS	0x2c	A 组端口原始中断状态寄存器
GPIO_LS_SYNC	0x30	配置中断同步寄存器
GPIO_DEBOUNCE	0x34	防反跳配置寄存器
GPIO_PORTA_EOI	0x38	A 组端口中断清除寄存器

5.8.2.3 GPIO_SWPORTA_DR

域	位	读写	复位值	描述
Reserved	31:8			保留
Port A Data Register	7:0	WR	0x0	如果 A 组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 A 组端口的 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。

5.8.2.4 GPIO_SWPORTA_DDR

域	位	读写	复位值	描述
Reserved	31:8			保留
Port A direction Register	7:0	WR	0x0	写入该寄存器的值独立控制 A 组端口中相应数据位的方向。默认方向配置为输入。 0 – Input 1 – Output

5.8.2.5 GPIO_EXT_PORTA

域	位	读写	复位值	描述

Reserved	31:8			保留
External Port A	7:0	RO	0x0	当 A 组端口被配置为输入，读取该位置将读取信号的值。当 A 组端口数据方向设置为输出，读取该位置将读取端口 A 的数据寄存器。

5.8.2.6 GPIO_SWPORTB_DR

域	位	读写	复位值	描述
Reserved	31:8			保留
Port B Data Register	7:0	WR	0x0	如果 B 组端口的相应数据方向位设置为输出模式，则写入该寄存器的值在 B 组端口的 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。

5.8.2.7 GPIO_SWPORTB_DDR

域	位	读写	复位值	描述
Reserved	31:8			保留
Port B direction Register	7:0	WR	0x0	写入该寄存器的值独立控制 B 组端口中相应数据位的方向。默认方向配置为输入。 0 – Input 1 – Output

5.8.2.8 GPIO_EXT_PORTB

域	位	读写	复位值	描述

Reserved	31:8			保留
External Port B	7:0	RO	0x0	当 B 组端口被配置为输入，读取该位置将读取信号的值。当 B 组端口数据方向设置为输出，读取该位置将读取 B 组端口的数据寄存器。

5.8.2.9 GPIO_INTEN

域	位	读写	复位值	描述
Reserved	31:8			保留
Port A Interrupt enable	7:0	WR	0x0	使能 A 组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 A 组端口上的对应位配置为中断；否则，A 组端口作为正常的 GPIO 信号运行。如果相应的数据方向寄存器设置为输出，则在 A 组端口的相应位上中断被禁用。 0- 配置 A 组端口位为正常 GPIO 信号 1- 配置 A 组端口位为中断

5.8.2.10 GPIO_INTMASK

域	位	读写	复位值	描述
Reserved	31:8			保留
Port A Interrupt mask	7:0	WR	0x0	控制 A 组端口上的中断是否屏蔽。默认情况下，所以中断位是为屏蔽的。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

5.8.2.11 GPIO_INTTYPE_LEVEL

域	位	读写	复位值	描述
Reserved	31:8			保留
Port A Interrupt level	7:0	WR	0x0	控制 A 组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

5.8.2.12 GPIO_INT_POLARITY

域	位	读写	复位值	描述
Reserved	31:8			保留
Port A Interrupt polarity	7:0	WR	0x0	控制 A 组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

5.8.2.13 GPIO_INTSTATUS

域	位	读写	复位值	描述
Reserved	31:8			保留
Port A Interrupt status	7:0	RO	0x0	A 组端口中断状态

5.8.2.14 GPIO_RAW_INTSTATUS

域	位	读写	复位值	描述
Reserved	31:8			保留
Port A Raw Interrupt status	7:0	RO	0x0	A 组端口原始的中断状态

5.8.2.15 GPIO_LS_SYNC

域	位	读写	复位值	描述
Reserved	31:8			保留
gpio_ls_sync	7:0	WR	0x0	将 1 写入该寄存器将导致所有电平敏感的中断被同步到 pclk_intr。 0 – 不同步到 pclk_intr 1 – 同步到 pclk_intr

5.8.2.16 GPIO_DEBOUNCE

域	位	读写	复位值	描述
Reserved	31:8			保留
Debounce clk config	15:8	WR	0x0	防反跳时钟分频配置 dbounce_clk=pclk/(debounce[15:8]+1)

				复位值: 0x0
Debounce enable	7:0	WR	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 复位值: 0x0

5.8.2.17 GPIO_PORTA_EOI

域	位	读写	复位值	描述
Reserved	31:8			保留
Port A Clear interrupt	7:0	WO	0x0	控制 A 组端口清除边缘型中断。当 1 写入该寄存器的相应位时，中断清除。当端口 A 没有配置为中断时，所有的中断被清除。 0 – 不清除中断 1 – 清除中断

5.9 WDT

FT-2000/4 的 WDT（Watchdog Timer）设计符合 ARM 的协议规范，作为 APB2 总线的一个从设备。

5.9.1 操作说明

操作 WDT 在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 0x3000000。

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 寄存器不够计数需求，可以直接写 64 位的 WCV 寄存器。

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 (但是会复位 WCS[0]，即 disable 看门狗) 。

5.9.1.1 初始化

1、喂狗：写 WRR、WCS、WOR、WCV 寄存器。（写 WRR、WCS、WOR 寄存器会直接将当前 sys_cnt + 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）

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。

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 disable 看门狗）。

7、当达到两次超时后，WCV 寄存器不会更新，仍能读取当前的 WCV 值。

5.9.2 寄存器说明

5.9.2.1 基地址

表 5-23 WDT 基地址

名称	基地址
WDT0	0x000_2800A000
WDT1	0x000_28016000

5.9.2.2 寄存器列表

WDT 寄存器空间为 8K，前 4K 为 Refresh 空间，后 4K 为 WatchDog Control 空间。

表 5-24 WDT 寄存器列表

寄存器	偏移	说明
WDT_WRR	0x0000	Watchdog 更新寄存器
WDT_W_IIR	0x0FCC	Watchdog 接口身份识别寄存器
WDT_W_IIR	0x1FCC	Watchdog 接口身份识别寄存器
WDT_WCS	0x1000	Watchdog 控制和状态寄存器
WDT_WOR	0x1008	Watchdog 清除寄存器
WDT_WCVL	0x1010	Watchdog 比较值的低 32bits 寄存器
WDT_WCVH	0x1014	Watchdog 比较值的高 32bits 寄存器

5.9.2.3 WDT_WRR

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

5.9.2.4 WDT_IIDR

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

5.9.2.5 WDT_WCS

域	位	读写	复位值	描述
reserved	31:3	RO	0	保留
ws1	2	RO	0	读返回当前 ws1 的值
ws0	1	RO	0	读返回当前 ws0 的值
wdt_en	0	WR	0	Watchdog 使能信号，高有效，常规复位和热保护复位都会清 0。

5.9.2.6 WDT_WOR

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

5.9.2.7 WDT_WCVL

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，在写低 32 位。

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

5.9.2.8 WDT_WCVH

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

5.10 SD 控制器

FT-2000/4 内置 SD 控制器（SDC），该控制器具有如下技术特征：

- 兼容 SD-2.0 版本（支持 SDSC\SDHC 类型卡）；
- 支持 SD 通信模式，不支持 SPI 模式；
- 支持时钟频率: 0-50MHZ
- 支持 4BIT 数据总线模式
- 集成的基于描述符的直接存储器访问（DMA）
- 内部集成可配置的接收和发送 FIFO 缓存
- 卡在位检测（插入/拔出）
- 支持单块读写操作，暂不支持多块读写
- 硬件支持 CRC 错误检测和中断产生

5.10.1 操作说明

SD 卡总线通信基于命令/数据的传输，有三种数据格式：命令包，响应包，数据包；利用 7-CRC 命令校验码与 16- CRC 数据校验码保证传输交互的正确性。控制器实现功能一方面是提供上层管理控制读写 SD 卡设备内部寄存器，获取设备信息；另一方面是管理 SD 卡设备与系统外部 MEMORY 的数据交互任务；同时将处理过程生成的正常异常中断传送到系统通用中断单元，供应用层读取。

■ 命令交互流程

SD 卡 CMD 控制总线的基本事务是命令/响应事务。内核可将 SD 控制器内部寄存器映射到统一的地址空间，使内核可以通过地址访问，读写 SD 卡控制寄存器，直接在命令与响应结构中进行控制并随时获取外设状态传输信息。内核对 SD 卡的指令操作需要设置两个寄存器：command 和 argument ，通过调用 SD_Controller_apb 实现。其中，command 作为命令 index 寄存器，argument 作为命令参数寄存器。当检查到 argument 寄存器发生更新时，SD_CMD_Master 状态机跳转，读取相关命令寄存器参数编码产生 40bits 命令包，通过 SD_CMD Serial_Host 模块打包成 48bits 串行移位发送到 SD 卡等待响应。接收到 SD 响应后返送回 SD_CMD_Master 模块执行错误校验，响应信息同时返回用户可读写寄存器，供应用层调用，CMD 命令线的整个执行传输进程由 NORMAL_INT_REG 寄存器跟踪检测。

■ 数据交互流程

SD 数据总线交互传输支持 burst 传输模式的 AXI4 总线，支持 DMA 模块读写模式，通过一个 DMA 接口提供独立的 FIFO 缓存存取，完成 FIFO 与 DMA 主端口之间的数据交互。DMA 寄存器可被内核访问来控制 DMA 操作。用户具体执行 SD 卡读写操作，通过写 bd_tx/rx 寄存器，设置需要搬移的数据 memory 地址和 SD 卡数据块起始位地址参数，启动 DMA 传输。SD_Data_Master 发送 block 读写指令到 SD_CMD_Master，SD_CMD Serial_Host 利用 st_dat_t 裁定 SD_DATA Serial_Host 传输开始时的读写状态跳转。SD_DATA Serial_Host 负责 tx & rx FIFO 数据发送及接收缓存控制。当发送完成时，SD_Data_Master 负责数据发送错误校验及中断查询 Dat_int_status 状态标识判定数据发送成功与否。需注意的是，SD 数据传输要求以块为单位，有单块或多块操作，目前设计验证均基于单块传输进行，按照协议块写操作使用简单的 busy 信号来表示 DAT0 数据线上的持续写操作（不管使用几线传输）。但多块操作模式在快速写操作时更有优势，对多块的读写操作将再后续升级中实现设计验证，本设计不再提及。

■ DMA

主机 CPU 发出读/写命令，访问核中的 DMA 相关控制寄存器。控制器核自动处理所有的 SD 卡协议，包括数据移位和 CRC 生成,使数据可以自动地在外部存储器和 SD 卡之间传送而无需 CPU 干预。

■ 中断

当 SD 卡完成某项操作或者工作异常，中断控制器会产生相应的中断，软件可对不同的中断做出相应的处理。中断包括读/写缓冲器已满/空、读/写缓冲器读写出错、命令发送完毕、数据传输完成、响应超时、读写数据 CRC 出错等。软件可通过配置中断控制器屏蔽或使能中断

5.10.1.1 控制器初始化

软件初始化 SDC 的步骤：

1. 设置 timeout 超时参数；
2. disable 控制器
3. 设置时钟分频参数
4. 使能控制器

复位后，SD 卡通过主机信息来决定使用何种模式（SD/SPI;SD 卡默认使用 DAT0 来传输数据，初始化后通过主机改变总线宽度；

5.10.1.2 命令发送及响应

发送命令步骤：

1. 设置命令寄存器
2. 设置命令参数寄存器
3. 等待响应
4. check 响应

5.10.1.3 数据传输及响应

数据传输及等待响应的步骤：

1. 控制器初始化流程
2. SD 卡参数配置流程；
3. 读写控制流程

5.10.2 寄存器说明

5.10.2.1 基地址

表 5-25 SDC 寄存器基地址

名称	基地址
SDC	0x000_28207C00

5.10.2.2 寄存器列表

表 5-26 SDC 寄存器列表

寄存器	偏移	说明
CONTROLL_SETTING_REG	0x00	控制器配置寄存器
ARGUMENT_REG	0x04	参数寄存器
CMD_SETTING_REG	0x08	命令寄存器
CLOCK_DIV_REG	0x0C	时钟分频寄存器
SOFTWARE_RESET_REG	0x10	复位控制寄存器
POWER_CONTROLL_REG	0x14	电源控制寄存器
TIMEOUT_CMD_REG	0x18	cmd 超时设置寄存器
TIMEOUT_DATA_REG	0x1C	数据超时设置寄存器
NORMAL_INT_EN_REG	0x20	中断使能寄存器
ERROR_INT_EN_REG	0x24	error 中断使能寄存器
BD_ISR_EN_REG	0x28	数据传输中断使能寄存器
CAPABILIES_REG	0x2c	状态寄存器
SD_DRV_REG	0x30	SD 卡驱动相位寄存器
SD_SAMP_REG	0x34	SD 卡采样相位寄存器
SD_SEN_REG	0x38	卡检测控制器
HDS_AXI_REG_CONF1	0x3c	AXI 边界配置寄存器 1

DAT_IN_M_RX_BD	0x40	SD BD RX 地址寄存器
DAT_IN_M_TX_BD	0x60	SD BD TX 地址寄存器
BLK_CNT_REG	0x80	块读写配置寄存器
HDS_AXI_REG_CONF2	0xa8	AXI 边界配置寄存器 2
NORMAL_INT_STATUS_REG	0xc0	中断状态寄存器
ERROR_INT_STATUS_REG	0xc4	error 中断寄存器
BD_ISR_REG	0xc8	数据传输中断状态寄存器
BD_STATUS	0xcc	bd 描述符寄存器
STATUS_REG	0xd0	状态寄存器
BLOCK	0xd4	块长度寄存器
CMD_RESP_1	0xe0	命令响应寄存器 1
CMD_RESP_2	0xe4	命令响应寄存器 2
CMD_RESP_3	0xe8	命令响应寄存器 3
CMD_RESP_4	0xec	命令响应寄存器 4

5.10.2.3 CONTROLL_SETTING_REG

域	位	读写	复位值	描述
Reserved	31:11	RW	0x0	
PERMDW	11:10	RW	0x0	写操作对应的大小端选择： 00： 小端对齐 01： 大端对齐 10： SD 协议方式
PERMDR	9:8	RW	0x0	读操作对应的大小端选择： 00： 小端对齐 01： 大端对齐 10： SD 协议方式
Reserved	7:0	RW	0x0	

5.10.2.4 ARGUMENT_REG

域	位	读写	复位值	描述
argument_reg	31:0	RW	0x0	命令参数寄存器

5.10.2.5 CMD_SETTING_REG

域	位	读写	复位值	描述
Reserved	31：16	RW	0x0	
TRTY	15：14	RW	0x0	10： adtc 指令 其它： 读写操作
CMDI	13:8	RW	0x0	命令索引
CMDW	7:6	RW	0x0	
Reserved	5	RW	0x0	
CICE	4	RW	0x0	0：CMD 响应时，不执行索引检查 1：CMD 响应时，执行索引检查
CRCE	3	RW	0x0	0 : CMD 响应时，不执行 CRC 检查 1 : CMD 响应时，执行 CRC 检查
Reserved	2	RW	0x0	
RTS	1:0	RW	0x0	0：不响应 01：响应字节长度 136 10: 响应字节长度 48 11：响应字节长度 48

5.10.2.6 CLOCK_DIV_REG

域	位	读写	复位值	描述
clock_div_reg	31:0	RW	299	CLKD-时钟分频系数

				SD_frequency= 600M/ (2*(clock_d+1))

5.10.2.7 SOFTWARE_RESET_REG

域	位	读写	复位值	描述
Reserved	31:4	RW	0x0	
CFCLF	3	RW	0x0	卡插入拔出状态触发标志清 0
BDRST	2	RW	0x0	DMA BD 清 0
Reserved	1	RW	0x0	
SRST	0	RW	0x0	控制器软复位

5.10.2.8 POWER_CONTROLL_REG

域	位	读写	复位值	描述
Reserved	31:0	RW	0x0	

5.10.2.9 TIMEOUT_CMD_REG

域	位	读写	复位值	描述
CTO	31：0	RW	0x0	command 超时参数

5.10.2.10 TIMEOUT_DATA_REG

域	位	读写	复位值	描述
DTO	31：0	RW	0x0	data 超时参数

5.10.2.11 NORMAL_INT_EN_REG

域	位	读写	复位值	描述
Reserved	31：16	RW	0x0	
EEI	15	RW	0x0	错误中断使能
Reserved	14：2	RW	0x0	
ECR	1	RW	0x0	卡拔出中断使能
ECC	0	RW	0x0	命令完成中断使能

5.10.2.12 ERROR_INT_EN_REG

域	位	读写	复位值	描述
Reserved	31:4	RW	0x0	
ECIE	3	RW	0x0	命令索引错误中断使能
Reserved	2	RW	0x0	
ECCRCE	1	RW	0x0	命令 CRC 错误中断使能
ECTE	0	RW	0x0	命令超时中断使能

5.10.2.13 BD_ISR_EN_REG

域	位	读写	复位值	描述
Reserved	31:16	RW	0x0	
EDAISE	15	RW	0x0	DMA 错误中断使能
Reserved	14:9	RW	0x0	
RESPE	8	RW	0x0	读 SD 卡操作，AXI BR 通道完成中断
EDATFRAXE	7	RW	0x0	AXI 总线错误中断使能
ENRCRCE	6	RW	0x0	CRC 校验错误中断使能
ETRE	5	RW	0x0	传输错误中断使能

ECMDE	4	RW	0x0	命令响应错误中断使能
EDTE	3	RW	0x0	数据超时中断使能
Reserved	2:1	RW	0x0	
ETRS	0	RW	0x0	DMA 传输完成中断使能

5.10.2.14 CAPABILIES_REG

域	位	读写	复位值	描述
Reserved	31:0	RW	0x0	

5.10.2.15 SD_DRV_REG

域	位	读写	复位值	描述
sd_drv_reg	31:0	RW	0x0	卡驱动相位配置参数

5.10.2.16 SD_SAMP_REG

域	位	读写	复位值	描述
sd_samp_reg	31:0	RW	150	卡采样相位配置参数

5.10.2.17 SD_SEN_REG

域	位	读写	复位值	描述
DEBNCE	31:8	RW	0x0	去抖时钟分频参数
Reserved	7:3	RW	0x0	
CRES	2	RW	0x0	CARD 在位状态标志选择 0： 卡在位-0，不在位-1 1: 卡在位-1，不在位-0

CREFR	1	RW	0x0	卡拔出时自动释放 AXI 总线选择
Reserved	0		0x0	

5.10.2.18 HDS_AXI_REG_CONF1

域	位	读写	复位值	描述
Reserved	31:8	RW	0x0	AXI 边界信号配置 1
Reserved	27:24	RW		
awregion_hds_m	22:19	RW		
awsnoop_hds_m	18:16	RW		
arbar_hds_m	15:14	RW		
ardomain_hds_m	13:12	RW		
arregion_hds_m	11:8	RW		
arsnoop_hds_m	7:4	RW		
awbar_hds_m	3:2	RW		
awdomain_hds_m	1:0	RW		

5.10.2.19 DAT_IN_M_RX_BD

域	位	读写	复位值	描述
dat_in_m_rx_bd	31:0	RW	0x0	dma 读卡地址配置：4 个 cycle 系统低 4B-系统高 4B-SD 低 4B- SD 高 4B

5.10.2.20 DAT_IN_M_TX_BD

域	位	读写	复位值	描述
dat_in_m_tx_bd	31:0	RW	0x0	dma 写卡地址配置：4 个 cycle 系统低 4B-系统高 4B-SD 低 4B- SD 高 4B

5.10.2.21 BLK_CNT_REG

域	位	读写	复位值	描述
blk_cnt_reg	31:0	RW	0x1	dma block num setting

5.10.2.22 HDS_AXI_REG_CONF2

域	位	读写	复位值	描述
Reserved	31:30	RW	0x0	AXI 边界信号配置 2
sd_arprot	29:27	RW		
sd_awprot	26:24	RW		
sd_arcache_m	23:20	RW		
sd_awcache_m	19:16	RW		
reserved	15:14	RW		
hda_arpro	13:11	RW		
hda_awprot	10:8	RW		
hda_arcache_m	7:4	RW		
hda_awcache_m	3:0	RW		

5.10.2.23 NORMAL_INT_STATUS_REG

域	位	读写	复位值	描述
Reserved	31:16	RW1C	0x0	
EI	15		0x0	命令错误中断
Reserved	14:2		0x0	
CR	1		0x0	卡移除中断
CC	0		0x0	命令完成中断

5.10.2.24 ERROR_INT_STATUS_REG

域	位	读写	复位值	描述
Reserved	31:5	RW1C	0x0	
CNR	4		0x0	命令响应错误中断
CIR	3		0x0	命令索引错误中断
reserved	2		0x0	
CCRCE	1		0x0	命令 CRC 错误中断
CTE	0		0x0	命令超时错误中断

5.10.2.25 BD_ISR_REG

域	位	读写	复位值	描述
Reserved	31:16	RW1C	0x0	
DAIS	15		0x0	DMA 错误中断
Reserved	14:9		0x0	
RESPE	8		0x0	读 SD 卡操作，AXI BR 通道完成中断
DATFRAX	7		0x0	axi 总线强制释放中断
NRCRC	6		0x0	无 CRC 响应中断
TRE	5		0x0	CRC 响应错误中断
CMDE	4		0x0	命令响应错误中断
DTE	3		0x0	超时中断
Reserved	2:1		0x0	
TRS	0		0x0	DMA 传输完成中断

5.10.2.26 BD_STATUS

域	位	读写	复位值	描述
bd_status	31:0	RO	0x0	bd 描述符

5.10.2.27 STATUS_REG

域	位	读写	复位值	描述
Reserved	31	RO	0x0	
DATMAST	30:27	RO	0x0	data_master 状态机
CDIF	26	RO	0x0	卡在位标志
CDRF	25	RO	0x0	卡不在位标志
CLSL	24	RO	0x1	命令闲信号
DLSL	23:20	RO	0xf	dat[3:0] 线信号
CDSL	19	RO	0x0	卡检测管脚信号
Reserved	18:16	RO	0x0	
CST	15:12	RO		cmd_host state 状态机
CSM	11:7	RO	0x0	
DAT_AVA	6	RO	0x0	DAT_AVA 当前命令状态流程运转完
CRC_VALID	5	RO	0x0	
Reserved	4:1	RO	0x0	
CICMD	0	RO	0x0	CMD 总线状态

5.10.2.28 BLOCK

域	位	读写	复位值	描述
block	31:0	RO	0x200	设置块长度

注：按照协议：对于 SDSC 卡，blcok size 不固定，默认 512B; 对 SDHC

卡 blcok size 固定为 512B。本设计中 blcok size 固定为 512B。

5.10.2.29 CMD_RESP_1

域	位	读写	复位值	描述
cmd_resp_1	31:0	RO	0x0	命令响应[31-0]

5.10.2.30 CMD_RESP_2

域	位	读写	复位值	描述
cmd_resp_2	31:0	RO	0x0	命令响应[63-32]

5.10.2.31 CMD_RESP_3

域	位	读写	复位值	描述
cmd_resp_3	31:0	RO	0x0	命令响应[95-64]

5.10.2.32 CMD_RESP_4

域	位	读写	复位值	描述
cmd_resp_4	31:0	RO	0x0	命令响应[127-96]

5.11 HD Audio 控制器

FT-2000/4 集成了 1 个 HD Audio（HDA）控制器。该控制器的作用是音频输入输出。

5.11.1 操作说明

CPU 从内存中读取音频文件信息，并完成音频文件的解码工作后，将解码后的音频数据和指令信息放入内存的指定位置。HDAudio 控制器通过内部 DMA 控制器将内存中的音频数据缓存到本地 FIFO，并负责实现本地 FIFO 与外部音频解码器 codec 之间的数据传输。最多可以连接 4 个 codec，每个 codec 可以根据 StreamID 号来提取自身所需要的数据流，从而实现多 codec 的工作。Codec 还可以将外部的模拟信号转换成数字信号，发送到控制器，从而实现录音功能。

HDAudio 控制器具有如下功能特性：

- 支持 44.1/48/96/192KHz 的音频数据采样率；
- 支持 16/20/24/32-bit 的声道采样率，单个音频流，最多支持 16 个声道数目；
- 最多支持 4 个 codec 同时连接工作，最大支持 4 个 Input Stream 和 4 个 Output Stream；
- 内部包含一个 DMAC，实现内存与 HD Audio 控制器之间的数据搬移；
- 支持中断处理机制，主要包括 Response 类型中断、codec 唤醒中断，描述符类型中断，可控制将中断输出至系统；
- 支持耳机热插拔机制。

5.11.2 寄存器说明

5.11.2.1 基地址

表 5-27 HDA 寄存器基地址

名称	基地址
HDA	0x000_28206000

5.11.2.2 寄存器列表

表 5-28 HDA 寄存器列表

寄存器	偏移(0x)	描述
GCAP	00	全局性能寄存器，用来说明一个 frame 中支持多少 Stream、是否支持 64 位寻址
VMIN	02	版本编号低位寄存器
VMAJ	03	版本编号高位寄存器
OUTPAY	04	用来说明输入输出 frame 中可以最多包含多少 word 的数据
INPAY	06	
GCTL	08	全局控制类寄存器，说明是否接受 unsolicited 响应以及 Flush 操作 CRST 信号可进行 controller 的软复位
WAKEEN	0C	标志 SDIx 线是否允许产生 wake 中断
STATESTS	0E	标志着哪个 SDIx 线有 codec 连接
GSTS	10	标志开始了 Flush 操作
OUTSTRMPAY	18	表示实际中，一个输出/输入 frame 中的 word 数量
INSTRMPAY	1A	
INTCTL	20	总的中断控制寄存器，控制三大类型的中断，判断是否将中断信号输出
INTSTS	24	总的中断状态寄存器，与中断控制寄存器相配合，控制中断信号的输出
INT_MODEL	230	配置中断模式 [0]表示对 rirb 类型的中断模式控制 [4:7]：InputStream 中断模式控制 [11:8]:Output Stream 中断模式控制 配置为 1，表示采用中断采用 delay 输出的方式；

		配置为 0，表示中断采用 bvalid 计数的方式 [11:8]固定配置为全 1；
STRM1_DELY	240	针对 4 个 Input stream，若配置为 0 模式的时候，dely 周期数值的配置； 4 个 Output stream 的 dely 固定是 0
STRM2_DELY	244	
STRM3_DELY	248	
STRM4_DELY	24c	
Wall Clock Counter	30	用来针对用一个系统中不同 controller 传输数据时候的同步
SSYNC	38	配合 RUN 位，实现对同一个 controller 中的 stream 之间数据接收与发送的同步
CORBLBA	40	CORB 在 DDR 中的基地址信息
CORBUBA	44	
CORBWP	48	CORB 的写指针，由软件写入
CORBRP	4A	CORB 的读指针，由硬件写入
CORBCTL	4C	控制着 DMAC 是否开始从 CORB 中取数据，以及 CMEI 中断允许与否的控制
CORBSTS	4D	状态标志位，是否产生 CMEI 中断
CORBSIZE	4E	CORB 允许设定尺寸以及实际尺寸
RIRBLBA	50	RIRB 在 DDR 中的基地址信息
RIRBUBA	54	
RIRBWP	58	RIRB 的写指针，硬件写入，软件读取
RINTCNT	5A	记录接收的 Rsponses 数量，用 N 表示
RIRBCTL	5C	RIRB 控制寄存器，控制中断的产生以及是否允许 DMAC 操作 RUN 位
RIRBSTS	5D	存放中断的状态标志信息
RIRBSIZE	5E	RIRB 允许设定尺寸以及实际尺寸
DPLBASE	70	内容为 DMA Position Buffer 在 DDR 中的地址信息，Controller 会更新 Buffer 中内容
DPUBASE	74	
SD1CTL	80	Stream Descriptor 控制寄存器，含 Stream ID,

		SDO 数量，RUN 和 SRST 位
SD1STS	83	Stream Descriptor 状态寄存器，包括 FIFO 中断的状态，IOC 中断状态等
SD1LPIB	84	Link Position Buffer 寄存器，表明当前从已经取出多少 Byte 数据到 DDR 中
SD1CBL	88	Cyclic Buffer Length，用来说明当前 cyclic buffer 的 byte 数目，由软件写入
SD1LVI	8C	Last Valid Index 寄存器，表示 BDL 列表中最后一个可引用的条目，由软件写入
SD1FIFOS	90	表示当前描述符对应 FIFO 的尺寸
SD1FMT	92	为 Stream Descriptor 的格式信息保存的寄存器
SD1BDPL	98	内容位 BDL 基地址信息
SD1BDPU	9C	
SD5CTL	100	Stream Descriptor 控制寄存器，含 Stream ID, SDO 数量，RUN 和 SRST 位
SD5STS	103	Stream Descriptor 状态寄存器，包括 FIFO 中断的状态，IOC 中断状态等
SD5LPIB	104	Link Position Buffer 寄存器，表明当前从已经取出多少 Byte 数据到 DDR 中
SD5CBL	108	Cyclic Buffer Length，用来说明当前 cyclic buffer 的 byte 数目，由软件写入
SD5LVI	10C	Last Valid Index 寄存器，表示 BDL 列表中最后一个可引用的条目，由软件写入
SD5FIFOS	110	表示当前描述符对应 FIFO 的尺寸
SD5FMT	112	为 Stream Descriptor 的格式信息保存的寄存器
SD5BDPL	118	内容位 BDL 基地址信息
SD5BDPU	11C	

SD2CTL	A0	Stream Descriptor 控制寄存器，含 Stream ID, SDO 数量，RUN 和 SRST 位
SD2STS	A3	Stream Descriptor 状态寄存器，包括 FIFO 中断的状态，IOC 中断状态等
SD2LPIB	A4	Link Position Buffer 寄存器，表明当前从已经取出多少 Byte 数据到 DDR 中
SD2CBL	A8	Cyclic Buffer Length，用来说明当前 cyclic buffer 的 byte 数目，由软件写入
SD2LVI	AC	Last Valid Index 寄存器，表示 BDL 列表中最后一个可引用的条目，由软件写入
SD2FIFOS	B0	表示当前描述符对应 FIFO 的尺寸
SD2FMT	B2	为 Stream Descriptor 的格式信息保存的寄存器
SD2BDPL	B8	内容位 BDL 基地址信息
SD2BDPU	BC	
SD6CTL	120	Stream Descriptor 控制寄存器，含 Stream ID, SDO 数量，RUN 和 SRST 位
SD6STS	123	Stream Descriptor 状态寄存器，包括 FIFO 中断的状态，IOC 中断状态等
SD6LPIB	124	Link Position Buffer 寄存器，表明当前从已经取出多少 Byte 数据到 DDR 中
SD6CBL	128	Cyclic Buffer Length，用来说明当前 cyclic buffer 的 byte 数目，由软件写入
SD6LVI	12C	Last Valid Index 寄存器，表示 BDL 列表中最后一个可引用的条目，由软件写入
SD6FIFOS	130	表示当前描述符对应 FIFO 的尺寸
SD6FMT	132	为 Stream Descriptor 的格式信息保存的寄存器
SD6BDPL	138	内容位 BDL 基地址信息

SD6BDPU	13C	
SD3CTL	C0	Stream Descriptor 控制寄存器，含 Stream ID, SDO 数量，RUN 和 SRST 位
SD3STS	C3	Stream Descriptor 状态寄存器，包括 FIFO 中断的状态，IOC 中断状态等
SD3LPIB	C4	Link Position Buffer 寄存器，表明当前从已经取出多少 Byte 数据到 DDR 中
SD3CBL	C8	Cyclic Buffer Length，用来说明当前 cyclic buffer 的 byte 数目，由软件写入
SD3LVI	CC	Last Valid Index 寄存器，表示 BDL 列表中最后一个可引用的条目，由软件写入
SD3FIFOS	D0	表示当前描述符对应 FIFO 的尺寸
SD3FMT	D2	为 Stream Descriptor 的格式信息保存的寄存器
SD3BDPL	D8	内容位 BDL 基地址信息
SD3BDPU	DC	
SD7CTL	140	Stream Descriptor 控制寄存器，含 Stream ID, SDO 数量，RUN 和 SRST 位
SD7STS	143	Stream Descriptor 状态寄存器，包括 FIFO 中断的状态，IOC 中断状态等
SD7LPIB	144	Link Position Buffer 寄存器，表明当前从已经取出多少 Byte 数据到 DDR 中
SD7CBL	148	Cyclic Buffer Length，用来说明当前 cyclic buffer 的 byte 数目，由软件写入
SD7LVI	14C	Last Valid Index 寄存器，表示 BDL 列表中最后一个可引用的条目，由软件写入
SD7FIFOS	150	表示当前描述符对应 FIFO 的尺寸
SD7FMT	152	为 Stream Descriptor 的格式信息保存的寄存器

SD7BDPL	158	内容位 BDL 基地址信息
SD7BDPU	15C	
SD4CTL	E0	Stream Descriptor 控制寄存器，含 Stream ID, SDO 数量，RUN 和 SRST 位
SD4STS	E3	Stream Descriptor 状态寄存器，包括 FIFO 中断的状态，IOC 中断状态等
SD4LPIB	E4	Link Position Buffer 寄存器，表明当前从已经取出多少 Byte 数据到 DDR 中
SD4CBL	E8	Cyclic Buffer Length，用来说明当前 cyclic buffer 的 byte 数目，由软件写入
SD4LVI	EC	Last Valid Index 寄存器，表示 BDL 列表中最后一个可引用的条目，由软件写入
SD4FIFOS	F0	表示当前描述符对应 FIFO 的尺寸
SD4FMT	F2	为 Stream Descriptor 的格式信息保存的寄存器
SD4BDPL	F8	内容位 BDL 基地址信息
SD4BDPU	FC	
SD8CTL	160	Stream Descriptor 控制寄存器，含 Stream ID, SDO 数量，RUN 和 SRST 位
SD8STS	163	Stream Descriptor 状态寄存器，包括 FIFO 中断的状态，IOC 中断状态等
SD8LPIB	164	Link Position Buffer 寄存器，表明当前从已经取出多少 Byte 数据到 DDR 中
SD8CBL	168	Cyclic Buffer Length，用来说明当前 cyclic buffer 的 byte 数目，由软件写入
SD8LVI	16C	Last Valid Index 寄存器，表示 BDL 列表中最后一个可引用的条目，由软件写入
SD8FIFOS	170	表示当前描述符对应 FIFO 的尺寸
SD8FMT	172	为 Stream Descriptor 的格式信息保存的寄存

		器
SD8BDPL	178	内容位 BDL 基地址信息
SD8BDPU	17C	

5.11.2.3 GCAP

域	位	读写	复位值	描述
Output stream support	15：12	RO	0x4	设计支持最大输出 stream 数量
Intput stream support	11：8	RO	0x4	设计支持最大输入 stream 数量
Bidirectional Streams Support	7:3	RO	0x0	设计支持双向 stream 数量，本设计不支持双向 stream
Data Out Signals	2:1	RO	0x0	设计支持数据输出线数量
64Bit address Support	0	RO	0x1	1-设计支持 64bits 地址输出 0-设计不支持 64bits 地址输出

5.11.2.4 VMIN

域	位	读写	复位值	描述
Minor Version	7:0	RO	0x00	当前设计版本的低位编号

5.11.2.5 VMAJ

域	位	读写	复位值	描述
Major Version	7:0	RO	0x00	当前设计版本的高位编号

5.11.2.6 OUTPAY

域	位	读写	复位值	描述
OUTPAY	15:0	RO	0x3C	当前设计 SDO 输出线支持最大有效数据量
INPAY	15:0	RO	0x1D	当前设计 SDI 输入线支持最大有效数据量

5.11.2.7 INPAY

域	位	读写	复位值	描述

5.11.2.8 GCTL

域	位	读写	复位值	描述
Reserved	31:9	RO	0x0	保留
UNSO	8	RW	0x0	是否接受主动响应 1-接受 condec 主动响应 0-不接受 codec 主动响应
Reserved	7:2	RO	0x0	Reserved
FCNTR	1	RO	0x0	设计不支持该位操作
CRST	0	W1C	0x0	全局软复位 1-没有进入软复位

				0-进行控制器软复位操作 当跳出软复位之后，只有软件读取到该位为 1 的时候，才能操作其他寄存器

5.11.2.9 WAKEEN

域	位	读写	复位值	描述
Reserved	15	RO	0x0	Reserved
SDIWEN	14:0	RW	0x0	控制 SDI 对应的 codec 是否可以产生唤醒中断，每一位对应一个 codec 写 1 表示对应 SDI 可以产生中断

5.11.2.10 STATESTS

域	位	读写	复位值	描述
Reserved	15	RO	0x0	Reserved
SDIWAK	14:0	RW	0x0	状态寄存器，写 1 表示当前该位对应的 sdi 线上有 codec 连接，或者状态请求

5.11.2.11 GSTS

域	位	读写	复位值	描述
Reserved	15:2	RO	0x0	Reserved
FSTS	1	RO	0x0	设计不支持该位操作
Reserved	0	RO	0x0	保留

5.11.2.12 OUTSTRMPAY

域	位	读写	复位值	描述
OUTSTRMPAY	15:0	RO	0x0	单个 stream 输出负载最大数据量

5.11.2.13 INSTRMPAY

域	位	读写	复位值	描述
INSTRMPAY	15:0	RO	0x0	单个 stream 输入负载最大数据量

5.11.2.14 INTCTL

域	位	读写	复位值	描述
GIE	31	RW	0x0	全局中断输出使能控制位
CIE	30	RW	0x0	一般中断使能控制位；当 esponse 接受完成，或者 overrun 的时候，若该位置 1，则输出中断，否则不输出
SIE	29:0	RW	0x0	对应各个 stream 中断使能控制位，写 1 的时候，表示对应 strem 允许产生中断，否则不允许

5.11.2.15 INTSTS

域	位	读写	复位值	描述
GIS	31	RO	0x0	全局状态标识位，是所有位的或结果
CIS	30	RO	0x0	通用类型状态标识位，例如 response 发送完成，response 溢出
SIS	29:0	RO	0x0	单独 stream 状态寄存器位

5.11.2.16 WCCT

域	位	读写	复位值	描述
WCCT	31:0	RO	0x0	计数寄存器，以 24MHz 进行计数的一个寄存器，为软件提供时间参数

5.11.2.17 SSYNC

域	位	读写	复位值	描述
Reserved	31:30	RO	0x0	保留
SSYNC	29:0	RW	0x0	stream 同步控制位，每一位对应一个 stream，该位用来控制同步的发送或者接收数据 stream。设定该位后，阻止数据的发送和接收，等待所有 streams 都准备好了，通过写该位为 0，数据就可以一起发送/接受

5.11.2.18 CORBLBASE

域	位	读写	复位值	描述
CORBLBASE	31:7	RW	0x0	CORB 的低 25bits 地址寄存器
Unimplemented Bits	6:0	RO	0x0	Hardwired to 0，保证 corb 地址是 128Byte 对齐

5.11.2.19 CORBUBASE

域	位	读写	复位值	描述
CORBUBASE	31:0	RW	0x0	CORB 的高 32bits 地址寄存器

5.11.2.20 CORBWP

域	位	读写	复位值	描述
Reserved	15:8	RO	0x0	保留
CORBWP	7:0	RW	0x0	CORB 的写指针寄存器，软件写入

5.11.2.21 CORBRP

域	位	读写	复位值	描述
CORBRPRST	15	RW	0x0	CORB 的读指针复位控制位，写 1 进行 CORB 读指针的复位
Reserved	14:8	RO	0x0	保留
CORBRP	7:0	RO	0x0	CORB 的读指针，软件读取该位，用来判定当前还可以向 CORB 中写入多少指令数据，而不至于溢出

5.11.2.22 CORBCTL

域	位	读写	复位值	描述
Reserved	7:2	RO	0x0	保留
CORBRUN	1	RW	0x0	CORB 使能位，1 表示可以开始从内存取指令
CMEIE	0	RW	0x0	设计中未使用该控制位

5.11.2.23 CORBSTS

域	位	读写	复位值	描述
Reserved	7:1	RO	0x0	保留
CMEI	0	RO	0x0	设计未使用到该位

5.11.2.24 CORBSIZE

域	位	读写	复位值	描述
CORBSZCAP	7:4	RO	ox4	表示设计中，CORB 支持的数据长度有哪些，本设计只支持 256byte 的数据
Reserved	3:2	RO	0x0	Reserved
CORBSIZE	1:0	RO	0x2	实际 CORB 对应的空间大小，该设计中，此位是只读，即只支持 256byte 的大小 0001 8 B = 2 entries 0010 64 B = 16 entries 0100 1024 B = 256 entries

5.11.2.25 RIRBLBASE

域	位	读写	复位值	描述
RIRBLBASE	31:7	RW	0x0	RIRB 的低 25bits 地址
Unimplemented Bits	6:0	RO	0x0	硬件接 0，实现 128byte 地址对其

5.11.2.26 RIRBUBASE

域	位	读写	复位值	描述

RIRBUBASE	31:0	RW	0x0	RIRB 的高 32bits 地址

5.11.2.27 RIRBWP

域	位	读写	复位值	描述
RIRBWPRST	15	W	0x0	RIRB 的写指针复位
Reserved	14:8	RO	0x0	Reserved
RIRBWP	7:0	RO	0x0	硬件操作的，当前 RIRB 写指针，当前硬件写入 RIRB 的最后一个有效的位置

5.11.2.28 RINTCNT

域	位	读写	复位值	描述
Reserved	15:8	RO	0x0	Reserved
RINTCNT	7:0	RW	0x0	用来设定当前收到多少个 response 后产生中断

5.11.2.29 RIRBCTL

域	位	读写	复位值	描述
Reserved	7:3	RO	0x0	保留
RIRBOIC	2	RW	0x0	设计中未使用该位
RIRBDMAEN	1	RW	0x0	RIRB 使能控制位
RINTCTL	0	RW	0x0	当硬件接收到 N 个 response 后，该位控制是否产生中断

5.11.2.30 RIRBSTS

域	位	读写	复位值	描述

Reserved	7:3	RO	0x0	保留
RIRBOIS	2	RW	0x0	当 RIRB 写入溢出的时候，将该位置 1
Reserved	1	RO	0x0	Reserved
RINTFL	0	W1C	0x0	当硬件接收到 N 个 response 后，将该位置 1

5.11.2.31 RIRBSIZE

域	位	读写	复位值	描述
RIRBSZCAP	7:4	RO	ox4	只读寄存器位，用来表示当前 RIRB 支持条目的数据量可以是多少 0001 8 B = 2 entries 0010 64 B = 16 entries 0100 1024 B = 256 entries
Reserved	3:2	RO	0x0	Reserved
RIRBSIZE	1:0	RO	0x2	只读寄存器，即当前 RIRB 条目数量只支持 256Byte 大小

5.11.2.32 SDnCTL

域	位	读写	复位值	描述
STRM	23:20	RW	0x0	当前 stream 的标识号，给 codec 提供识别信息
DIR	19	RW	0x0	当前 stream 的方向控制，软件不应该写该位,默认 0～3 表示输入 stream，4～7 是输出 stream

TP	18	RW	0x0	控制优先级，设计未实现该功能
STRIPE	17:16	RW	0x0	控制可以将输出数据拆分成在多个 SDO 线上输出，因为设计只支持 1 个 SDO，故该位无效
Reserved	15:5	RO	0x0	保留
DEIE	4	RW	0x0	中断状态控制位，当取指针出现错误的时候，控制是否产生中断
FEIE	3	RW	0x0	控制本地 FIFO 出错的时候，是否产生中断，该设计未使用
IOCE	2	RW	0x0	当 BDL 列表的 IOC 位包含有效的时候，该位控制是否产生中断
RUN	1	RW	0x0	Stream Run 开始标志位
SRST	0	RW	0x0	stream 的软复位控制，写 1 进行软复位，写 0 跳出软复位

5.11.2.33 SDnSTS

域	位	读写	复位值	描述
Reserved	7:6	RO	0x0	保留
FIFORDY	5	RO	0x0	FIFO 准备好标志位，设计未使用
DESE	4	W1C	0x0	stream 指针出现错误状态位
FIFOE	3	W1C	0x0	FIFO 错误状态位
BCIS	2	W1C	0x0	当该 Buffer 的 IOC 标志位有效，buffer 完成后，该位置 1
Reserved	7:6	RO	0x0	保留

5.11.2.34 SDnLPIB

域	位	读写	复位值	描述

LPIB	31:0	RO	0x0	表明当前有多少数据从 BUFFER 中取回来了，当该值达到 CBL 的时候，清零

5.11.2.35 SDnCBL

域	位	读写	复位值	描述
CBL	31:0	RW	0x0	表明当前这个 BDL 条目中，所有 Byte 的数据

5.11.2.36 ISDnLVI

域	位	读写	复位值	描述
Reserved	15:8	RO	0x0	Reserved
LVI	7:0	RW	0x0	用来指示当前 BDL 列表的有效条目，实际条目值为该寄存器值+1

5.11.2.37 SDnFIFOS

域	位	读写	复位值	描述
FIFOS	15:0	RO	0x7F	用来指示当前本地 FIFO 可以存放的最大数目

5.11.2.38 SDnFMT

域	位	读写	复位值	描述
Reserved	15	RO	0x0	保留
BASE	14	RW	0x0	采样基本频率 0-48KHz 1-44.1KHz
MULT	13:11	RW	0x0	基本采样率的倍数

DIV	10:8	RW	0x0	基本采样率的分频
Reserved	7	RO	0x0	保留
BITS	6:4	RW	0x0	采样分辨率 000 = 8 bits. 001 = 16 bits. 010 = 20 bits. 011 = 24 bits.100 = 32 bits. 101-111 = Reserved
CHAN	3:0	RW	0x0	采样声道数量配置

5.11.2.39 SDnBDPL

域	位	读写	复位值	描述
BDLLBASE	31:7	RW	0x0	当前 stream 的 BDL 低 25bit 地址
Reserved	6:0	RO	0x0	硬件接 0

5.11.2.40 SDnBDPU

域	位	读写	复位值	描述
BDLUBASE	31:0	RW	0x0	当前 stream 的 BDL 高 32bits 地址

5.12 CAN

5.12.1 基地址

表 5-29 CAN 基地址

名称	基地址
CAN0	0x000_28207000
CAN1	0x000_28207400

CAN2	0x000_28207800

5.12.2 寄存器列表

表 5-30 CAN 寄存器列表

寄存器名称	偏移	功能描述
CAN_CTRL	0x0000	全局控制寄存器
CAN_INTR	0x0004	中断寄存器
CAN_ARB_RATE_CTRL	0x0008	仲裁段速率控制寄存器
CAN_DAT_RATE_CTRL	0x000c	数据段速率控制寄存器
CAN_ACC_ID0	0x0010	可接收识别符 0 寄存器
CAN_ACC_ID1	0x0014	可接收识别符 1 寄存器
CAN_ACC_ID2	0x0018	可接收识别符 2 寄存器
CAN_ACC_ID3	0x001c	可接收识别符 3 寄存器
CAN_ACC_ID0_MASK	0x0020	可接收识别符 0 掩码寄存器
CAN_ACC_ID1_MASK	0x0024	可接收识别符 1 掩码寄存器
CAN_ACC_ID2_MASK	0x0028	可接收识别符 2 掩码寄存器
CAN_ACC_ID3_MASK	0x002C	可接收识别符 3 掩码寄存器
CAN_XFER_STS	0x0030	传输状态寄存器
CAN_ERR_CNT	0x0034	错误计数寄存器
CAN_FIFO_CNT	0x0038	FIFO 计数寄存器
CAN_DMA_CTRL	0x003c	DMA 请求控制寄存器
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

5.12.3 寄存器说明

5.12.3.1 CAN_CTRL

位域	读写	复位值	功能描述
31:3	只读	0x0	保留位
2	读写	0x0	可接收识别符掩码使能 0:不使能 1:使能
1	读写	0x0	收发请求 0:只接收 1:发送和接收
0	读写	0x0	传输使能 0:不使能 1:使能

5.12.3.2 CAN_INTR

位域	读写	复位值	功能描述
31:24	只读	0x0	保留位
23	写 1 清 0	0x0	错误中断清除 0: 不清除 1: 清除
22	写 1 清 0	0x0	发送帧结束中断清除 0: 不清除 1: 清除
21	写 1 清 0	0x0	接收帧结束中断清除 0: 不清除 1: 清除

位域	读写	复位值	功能描述
20	写 1 清 0	0x0	发送 FIFO 空中断清除 0: 不清除 1: 清除
19	写 1 清 0	0x0	接收 FIFO 满中断清除 0: 不清除 1: 清除
18	写 1 清 0	0x0	隐性错误中断清除 0: 不清除 1: 清除
17	写 1 清 0	0x0	隐性警告中断清除 0: 不清除 1: 清除
16	写 1 清 0	0x0	总线关闭中断清除 0: 不清除 1: 清除
15	读写	0x0	错误中断使能 0: 不使能 1: 使能
14	读写	0x0	发送帧结束中断使能 0: 不使能 1: 使能
13	读写	0x0	接收帧结束中断使能 0: 不使能 1: 使能
12	读写	0x0	发送 FIFO 空中断使能 0: 不使能 1: 使能

位域	读写	复位值	功能描述
11	读写	0x0	接收 FIFO 满中断使能 0: 不使能 1: 使能
10	读写	0x0	隐性错误中断使能 0: 不使能 1: 使能
9	读写	0x0	隐性警告中断使能 0: 不使能 1: 使能
8	读写	0x0	总线关闭中断使能 0: 不使能 1: 使能
7	只读	0x0	错误中断状态 0: 不生效 1: 生效
6	只读	0x0	发送帧结束中断状态 0: 不生效 1: 生效
5	只读	0x0	接收帧结束中断状态 0: 不生效 1: 生效
4	只读	0x0	发送 FIFO 空中断状态 0: 不生效 1: 生效
3	只读	0x0	接收 FIFO 空中断状态 0: 不生效 1: 生效

位域	读写	复位值	功能描述
2	只读	0x0	隐性错误中断状态 0: 不生效 1: 生效
1	只读	0x0	隐性警告中断状态 0: 不生效 1: 生效
0	只读	0x0	总线关闭中断状态 0: 不生效 1: 生效

5.12.3.3 CAN_ARB_RATE_CTRL

位域	读写	复位值	功能描述
31:25	只读	0x0	保留位
24:16	读写	0x0	前级分频器 0~511 表示 1~512
15:11	只读	0x0	保留位
10:8	读写	0x0	相位 2 段计数值 0~7 表示 1~8
7:5	读写	0x0	相位 1 段计数值 0~7 表示 1~8
4:2	读写	0x0	传播段计数值 0~7 表示 1~8
1:0	读写	0x0	同步跳转宽度 0~3 表示 1~4

5.12.3.4 CAN_DAT_RATE_CTRL

位域	读写	复位值	功能描述
31:21	只读	0x0	保留位
20:16	读写	0x0	前级分频器 0~31 表示 1~32
15:11	只读	0x0	保留位
10:8	读写	0x0	相位 2 段计数值 0~7 表示 1~8
7:5	读写	0x0	相位 1 段计数值 0~7 表示 1~8
4:2	读写	0x0	传播段计数值 0~7 表示 1~8
1:0	读写	0x0	同步跳转宽度 0~3 表示 1~4

5.12.3.5 CAN_ACC_ID0/1/2/3

位域	读写	复位值	功能描述
31:1	只读	0x0	保留位
0	读写	0x0	只有携带可接收识别符的帧才会被写入 FIFO

5.12.3.6 CAN_ACC_ID0/1/2/3_MASK

位域	读写	复位值	功能描述
31:29	只读	0x0	保留位
28:0	读写	0x0	如果对应与可接收识别符的位域值为 1，则忽略此位是否与接收帧对应位域识别符的匹配情况

5.12.3.7 CAN_XFER_STS

位域	读写	复位值	功能描述
31:10	只读	0x0	保留位
9	只读	0x0	传输状态 0:闲 1:忙
8	只读	0x0	接收状态 0:未接收 1:接收
7	只读	0x0	发送状态 0:未发送 1:发送
6:3	只读	0x0	有限状态机当前状态编号
2:0	只读	0x0	帧状态 000:数据帧 001:遥控帧 010:错误帧 011:过载帧 100:帧间空白

5.12.3.8 CAN_ERR_CNT

位域	读写	复位值	功能描述
31:25	只读	0x0	保留位
24:16	只读	0x0	发送错误计数器 0~256
15:9	只读	0x0	保留位

位域	读写	复位值	功能描述
8:0	只读	0x0	接收错误计数器 0~256

5.12.3.9 CAN_FIFO_CNT

位域	读写	复位值	功能描述
31:17	只读	0x0	保留位
22:16	只读	0x0	发送 FIFO 有效数据个数 0~64
15:9	只读	0x0	保留位
6:0	只读	0x0	接收 FIFO 有效数据个数 0~64

5.12.3.10 CAN_DMA_CTRL

位域	读写	复位值	功能描述
31:25	只读	0x0	保留位
24	读写	0x0	发送 FIFO DMA 请求使能
23:16	读写	0x0	发送 FIFO DMA 请求阈值 0~255
15:9	读写	0x0	保留位
8	读写	0x0	接收 FIFO DMA 请求使能
7:0	读写	0x0	接收 FIFO DMA 请求阈值 0~255

5.12.3.11 CAN_TX_FIFO

位域	读写	复位值	功能描述
31:0	W	0x0	发送 FIFO 影子寄存器

5.12.3.12 CAN_RX_FIFO

位域	属性	复位值	功能描述
31:0	只读	0x0	接收 FIFO 影子寄存器

5.13 芯片引脚控制

FT-2000/4 中的某些 IO 之间存在共用同一物理引脚情况，即引脚复用。用户可以通过配置相关寄存器来设置复用引脚的当前功能。按配置的功能不同，可以将寄存器分为引脚功能配置寄存器、通用 IO 引脚延迟配置寄存器和时钟引脚延时配置寄存器。

5.13.1 寄存器基地址

表 5-31 引脚复用配置寄存器基址

名称	基地址
引脚复用控制	0x28180000

5.13.2 引脚功能配置

5.13.2.1 操作说明

每个复用引脚的控制信号有有 4 位。其中，bit[1:0]表示功能选择，2’b00：选择 func0；2’b01：选择 func1；2’b10：选择 func2；2’b11：选择 func3。bit[3:2]表示上下拉电阻选择，2’b00：上下拉电阻都不使能；2’b01：使能下拉电阻；2’b10：使能上拉电阻；2’b11：使能上拉电阻（原则上不会出现上下拉电阻都使能的情

况，如果这样配置，即使能上拉电阻）。

5.13.2.2 寄存器说明

表 5-32 引脚功能配置寄存器

偏移	位域	说明	复位值
0x0200	[31 : 30]	控制 all_pll_lock_pad 的上下拉	2'b00
	[29 : 28]	控制 all_pll_lock_pad 的复用功能	2'b00
	[27 : 26]	控制 cru_clk_obv_pad 的上下拉	2'b00
	[25 : 24]	控制 cru_clk_obv_pad 的复用功能	2'b00
	[23 : 22]	控制 sjtag_tdi_pad 的上下拉	2'b10
	[21 : 20]	控制 sjtag_tdi_pad 的复用功能	2'b00
	[19 : 18]	控制 sjtag_tms_pad 的上下拉	2'b10
	[17 : 16]	控制 sjtag_tms_pad 的复用功能	2'b00
	[15 : 14]	控制 sjtag_ntrst_pad 的上下拉	2'b01
	[13 : 12]	控制 sjtag_ntrst_pad 的复用功能	2'b00
	[11 : 10]	控制 sjtag_tdo_pad 的上下拉	2'b01
	[9 : 8]	控制 sjtag_tdo_pad 的复用功能	2'b00
	[7 : 6]	控制 tjtag_tdo_pad 的上下拉	2'b01
	[5 : 4]	控制 tjtag_tdo_pad 的复用功能	2'b00
	[3 : 2]	控制 tjtag_ntrst_pad 的上下拉	2'b01
	[1 : 0]	控制 tjtag_ntrst_pad 的复用功能	2'b00
0x0204	[31 : 30]	控制 tjtag_tms_pad 的上下拉	2'b10
	[29 : 28]	控制 tjtag_tms_pad 的复用功能	2'b00
	[27 : 26]	控制 tjtag_tdi_pad 的上下拉	2'b10
	[25 : 24]	控制 tjtag_tdi_pad 的复用功能	2'b00
	[23 : 22]	控制 ntrst_swj_pad 的上下拉	2'b01
	[21 : 20]	控制 ntrst_swj_pad 的复用功能	2'b00
	[19 : 18]	控制 tdi_swj_pad 的上下拉	2'b10
	[17 : 16]	控制 tdi_swj_pad 的复用功能	2'b00

	[15 : 14]	控制 swditms_swj_pad 的上下拉	2'b10
	[13 : 12]	控制 swditms_swj_pad 的复用功能	2'b00
	[11 : 10]	控制 swdo_swj_pad 的上下拉	2'b01
	[9 : 8]	控制 swdo_swj_pad 的复用功能	2'b00
	[7 : 6]	控制 tdo_swj_pad 的上下拉	2'b01
	[5 : 4]	控制 tdo_swj_pad 的复用功能	2'b00
	[3 : 2]	控制 hdt_mb_done_state_pad 的上下拉	2'b00
	[1 : 0]	控制 hdt_mb_done_state_pad 的复用功能	2'b00
0x0208	[31 : 30]	控制 hdt_mb_fail_state_pad 的上下拉	2'b00
	[29 : 28]	控制 hdt_mb_fail_state_pad 的复用功能	2'b00
	[27 : 26]	控制 i2c_0_scl_pad 的上下拉	2'b10
	[25 : 24]	控制 i2c_0_scl_pad 的复用功能	2'b00
	[23 : 22]	控制 i2c_0_sda_pad 的上下拉	2'b10
	[21 : 20]	控制 i2c_0_sda_pad 的复用功能	2'b00
	[19 : 18]	控制 spi0_csn0_pad 的上下拉	2'b00
	[17 : 16]	控制 spi0_csn0_pad 的复用功能	2'b00
	[15 : 14]	控制 spi0_sck_pad 的上下拉	2'b00
	[13 : 12]	控制 spi0_sck_pad 的复用功能	2'b00
	[11 : 10]	控制 spi0_so_pad 的上下拉	2'b00
	[9 : 8]	控制 spi0_so_pad 的复用功能	2'b00
	[7 : 6]	控制 spi0_si_pad 的上下拉	2'b01
	[5 : 4]	控制 spi0_si_pad 的复用功能	2'b00
	[3 : 2]	控制 sd_cmd_pad 的上下拉	2'b10
	[1 : 0]	控制 sd_cmd_pad 的复用功能	2'b00
0x020c	[31 : 30]	控制 sd_clk_pad 的上下拉	2'b00
	[29 : 28]	控制 sd_clk_pad 的复用功能	2'b00
	[27 : 26]	控制 sd_dat0_pad 的上下拉	2'b10
	[25 : 24]	控制 sd_dat0_pad 的复用功能	2'b00
	[23 : 22]	控制 sd_dat1_pad 的上下拉	2'b10

	[21 : 20]	控制 sd_dat1_pad 的复用功能	2'b00
	[19 : 18]	控制 sd_dat2_pad 的上下拉	2'b10
	[17 : 16]	控制 sd_dat2_pad 的复用功能	2'b00
	[15 : 14]	控制 sd_dat3_pad 的上下拉	2'b10
	[13 : 12]	控制 sd_dat3_pad 的复用功能	2'b00
	[11 : 10]	控制 sd_detect_pad 的上下拉	2'b10
	[9 : 8]	控制 sd_detect_pad 的复用功能	2'b00
	[7 : 6]	控制 hda_bclk_pad 的上下拉	2'b00
	[5 : 4]	控制 hda_bclk_pad 的复用功能	2'b00
	[3 : 2]	控制 hda_rst_pad 的上下拉	2'b00
	[1 : 0]	控制 hda_rst_pad 的复用功能	2'b00
0x0210	[31 : 30]	控制 hda_sync_pad 的上下拉	2'b00
	[29 : 28]	控制 hda_sync_pad 的复用功能	2'b00
	[27 : 26]	控制 hda_sdo_pad 的上下拉	2'b00
	[25 : 24]	控制 hda_sdo_pad 的复用功能	2'b00
	[23 : 22]	控制 hda_sdi0_pad 的上下拉	2'b10
	[21 : 20]	控制 hda_sdi0_pad 的复用功能	2'b00
	[19 : 18]	控制 uart_0_rxd_pad 的上下拉	2'b10
	[17 : 16]	控制 uart_0_rxd_pad 的复用功能	2'b00
	[15 : 14]	控制 uart_0_txd_pad 的上下拉	2'b00
	[13 : 12]	控制 uart_0_txd_pad 的复用功能	2'b00
	[11 : 10]	控制 uart_1_rxd_pad 的上下拉	2'b10
	[9 : 8]	控制 uart_1_rxd_pad 的复用功能	2'b00
	[7 : 6]	控制 uart_1_txd_pad 的上下拉	2'b00
	[5 : 4]	控制 uart_1_txd_pad 的复用功能	2'b00
	[3 : 2]	控制 uart_2_rxd_pad 的上下拉	2'b10
	[1 : 0]	控制 uart_2_rxd_pad 的复用功能	2'b00
0x0214	[31 : 30]	控制 uart_2_txd_pad 的上下拉	2'b00
	[29 : 28]	控制 uart_2_txd_pad 的复用功能	2'b00

	[27 : 26]	控制 uart_3_rxd_pad 的上下拉	2'b10
	[25 : 24]	控制 uart_3_rxd_pad 的复用功能	2'b00
	[23 : 22]	控制 uart_3_txd_pad 的上下拉	2'b00
	[21 : 20]	控制 uart_3_txd_pad 的复用功能	2'b00
	[19 : 18]	控制 qspi_csn0_pad 的上下拉	2'b00
	[17 : 16]	控制 qspi_csn0_pad 的复用功能	2'b00
	[15 : 14]	控制 qspi_csn1_pad 的上下拉	2'b00
	[13 : 12]	控制 qspi_csn1_pad 的复用功能	2'b00
	[11 : 10]	控制 qspi_csn2_pad 的上下拉	2'b00
	[9 : 8]	控制 qspi_csn2_pad 的复用功能	2'b00
	[7 : 6]	控制 qspi_csn3_pad 的上下拉	2'b00
	[5 : 4]	控制 qspi_csn3_pad 的复用功能	2'b00
	[3 : 2]	控制 qspi_sck_pad 的上下拉	2'b00
	[1 : 0]	控制 qspi_sck_pad 的复用功能	2'b00
0x0218	[31 : 30]	控制 qspi_so_io0_pad 的上下拉	2'b01
	[29 : 28]	控制 qspi_so_io0_pad 的复用功能	2'b00
	[27 : 26]	控制 qspi_si_io1_pad 的上下拉	2'b01
	[25 : 24]	控制 qspi_si_io1_pad 的复用功能	2'b00
	[23 : 22]	控制 qspi_wp_io2_pad 的上下拉	2'b01
	[21 : 20]	控制 qspi_wp_io2_pad 的复用功能	2'b00
	[19 : 18]	控制 qspi_hold_io3_pad 的上下拉	2'b01
	[17 : 16]	控制 qspi_hold_io3_pad 的复用功能	2'b00
	[15 : 14]	控制 ext_lpc_lad_0_pad 的上下拉	2'b01
	[13 : 12]	控制 ext_lpc_lad_0_pad 的复用功能	2'b00
	[11 : 10]	控制 ext_lpc_lad_1_pad 的上下拉	2'b01
	[9 : 8]	控制 ext_lpc_lad_1_pad 的复用功能	2'b00
	[7 : 6]	控制 ext_lpc_lad_2_pad 的上下拉	2'b01
	[5 : 4]	控制 ext_lpc_lad_2_pad 的复用功能	2'b00
	[3 : 2]	控制 ext_lpc_lad_3_pad 的上下拉	2'b01

	[1 : 0]	控制 ext_lpc_lad_3_pad 的复用功能	2'b00
0x021c	[31 : 30]	控制 peu0_linkup0_pad 的上下拉	2'b00
	[29 : 28]	控制 peu0_linkup0_pad 的复用功能	2'b00
	[27 : 26]	控制 peu0_linkup1_pad 的上下拉	2'b00
	[25 : 24]	控制 peu0_linkup1_pad 的复用功能	2'b00
	[23 : 22]	控制 peu0_linkup2_pad 的上下拉	2'b00
	[21 : 20]	控制 peu0_linkup2_pad 的复用功能	2'b00
	[19 : 18]	控制 peu0_c0_clkreq_pad 的上下拉	2'b10
	[17 : 16]	控制 peu0_c0_clkreq_pad 的复用功能	2'b00
	[15 : 14]	控制 peu0_c1_clkreq_pad 的上下拉	2'b10
	[13 : 12]	控制 peu0_c1_clkreq_pad 的复用功能	2'b00
	[11 : 10]	控制 peu0_c2_clkreq_pad 的上下拉	2'b10
	[9 : 8]	控制 peu0_c2_clkreq_pad 的复用功能	2'b00
	[7 : 6]	控制 peu1_c0_clkreq_pad 的上下拉	2'b10
	[5 : 4]	控制 peu1_c0_clkreq_pad 的复用功能	2'b00
	[3 : 2]	控制 peu1_c1_clkreq_pad 的上下拉	2'b10
	[1 : 0]	控制 peu1_c1_clkreq_pad 的复用功能	2'b00
0x0220	[31 : 30]	控制 peu1_c2_clkreq_pad 的上下拉	2'b10
	[29 : 28]	控制 peu1_c2_clkreq_pad 的复用功能	2'b00
	[27 : 26]	控制 phy_gmac0_clk_rx_pad 的上下拉	2'b01
	[25 : 24]	控制 phy_gmac0_clk_rx_pad 的复用功能	2'b00
	[23 : 22]	控制 gmac0_phy_clk_tx_pad 的上下拉	2'b00
	[21 : 20]	控制 gmac0_phy_clk_tx_pad 的复用功能	2'b00
	[19 : 18]	控制 phy_gmac0_rxd0_pad 的上下拉	2'b01
	[17 : 16]	控制 phy_gmac0_rxd0_pad 的复用功能	2'b00
	[15 : 14]	控制 phy_gmac0_rxd1_pad 的上下拉	2'b01
	[13 : 12]	控制 phy_gmac0_rxd1_pad 的复用功能	2'b00
	[11 : 10]	控制 phy_gmac0_rxd2_pad 的上下拉	2'b01
	[9 : 8]	控制 phy_gmac0_rxd2_pad 的复用功能	2'b00

	[7 : 6]	控制 phy_gmac0_rxd3_pad 的上下拉	2'b01
	[5 : 4]	控制 phy_gmac0_rxd3_pad 的复用功能	2'b00
	[3 : 2]	控制 phy_gmac0_rxdv_pad 的上下拉	2'b01
	[1 : 0]	控制 phy_gmac0_rxdv_pad 的复用功能	2'b00
0x0224	[31 : 30]	控制 gmac0_phy_txd0_pad 的上下拉	2'b00
	[29 : 28]	控制 gmac0_phy_txd0_pad 的复用功能	2'b00
	[27 : 26]	控制 gmac0_phy_txd1_pad 的上下拉	2'b00
	[25 : 24]	控制 gmac0_phy_txd1_pad 的复用功能	2'b00
	[23 : 22]	控制 gmac0_phy_txd2_pad 的上下拉	2'b00
	[21 : 20]	控制 gmac0_phy_txd2_pad 的复用功能	2'b00
	[19 : 18]	控制 gmac0_phy_txd3_pad 的上下拉	2'b00
	[17 : 16]	控制 gmac0_phy_txd3_pad 的复用功能	2'b00
	[15 : 14]	控制 gmac0_phy_txen_pad 的上下拉	2'b00
	[13 : 12]	控制 gmac0_phy_txen_pad 的复用功能	2'b00
	[11 : 10]	控制 gmac0_phy_mdc_pad 的上下拉	2'b00
	[9 : 8]	控制 gmac0_phy_mdc_pad 的复用功能	2'b00
	[7 : 6]	控制 gmac0_phy_mdio_pad 的上下拉	2'b10
	[5 : 4]	控制 gmac0_phy_mdio_pad 的复用功能	2'b00
	[3 : 2]	控制 phy_gmac1_clk_rx_pad 的上下拉	2'b01
	[1 : 0]	控制 phy_gmac1_clk_rx_pad 的复用功能	2'b00
0x0228	[31 : 30]	控制 gmac1_phy_clk_tx_pad 的上下拉	2'b00
	[29 : 28]	控制 gmac1_phy_clk_tx_pad 的复用功能	2'b00
	[27 : 26]	控制 ckobv_sel0_pad 的上下拉	2'b01
	[25 : 24]	控制 ckobv_sel0_pad 的复用功能	2'b00
	[23 : 22]	控制 ckobv_sel1_pad 的上下拉	2'b01
	[21 : 20]	控制 ckobv_sel1_pad 的复用功能	2'b00
	[19 : 18]	控制 ckobv_sel2_pad 的上下拉	2'b01
	[17 : 16]	控制 ckobv_sel2_pad 的复用功能	2'b00
	[15 : 14]	控制 ckobv_sel3_pad 的上下拉	2'b01

	[13 : 12]	控制 ckobv_sel3_pad 的复用功能	2'b00
	[11 : 10]	控制 ckobv_sel4_pad 的上下拉	2'b01
	[9 : 8]	控制 ckobv_sel4_pad 的复用功能	2'b00
	[7 : 6]	控制 gmac1_phy_txd0_pad 的上下拉	2'b00
	[5 : 4]	控制 gmac1_phy_txd0_pad 的复用功能	2'b00
	[3 : 2]	控制 gmac1_phy_txd1_pad 的上下拉	2'b00
	[1 : 0]	控制 gmac1_phy_txd1_pad 的复用功能	2'b00
0x022c	[31 : 30]	reserved	2'b00
	[29 : 28]	reserved	2'b00
	[27 : 26]	控制 gmac1_phy_txd2_pad 的上下拉	2'b00
	[25 : 24]	控制 gmac1_phy_txd2_pad 的复用功能	2'b00
	[23 : 22]	控制 gmac1_phy_txd3_pad 的上下拉	2'b00
	[21 : 20]	控制 gmac1_phy_txd3_pad 的复用功能	2'b00
	[19 : 18]	控制 gmac1_phy_txen_pad 的上下拉	2'b00
	[17 : 16]	控制 gmac1_phy_txen_pad 的复用功能	2'b00
	[15 : 14]	控制 gmac1_phy_mdc_pad 的上下拉	2'b00
	[13 : 12]	控制 gmac1_phy_mdc_pad 的复用功能	2'b00
	[11 : 10]	控制 gmac1_phy_mdio_pad 的上下拉	2'b10
	[9 : 8]	控制 gmac1_phy_mdio_pad 的复用功能	2'b00
	[7 : 0]	预留	
0x0488	[23 : 20]	控制 pad_drive_strength5（DS5）驱动能力	4'h5
	[19 : 16]	控制 pad_drive_strength4（DS4）驱动能力	4'h4
	[15 : 12]	控制 pad_drive_strength3（DS3）驱动能力	4'h3
	[11 : 8]	控制 pad_drive_strength2（DS2）驱动能力	4'h2
	[7 : 4]	控制 pad_drive_strength1（DS1）驱动能力	4'h1
	[3: 0]	控制 pad_drive_strength0（DS0）驱动能力	4'h0

5.13.3 通用 IO 引脚时延配置

5.13.3.1 操作说明

FT-2000/4 的 IO 引脚可通过寄存器配置实现延时调节。数据路径的延时调节分粗条和精调，粗调每级延时约 530ps，精调级延时约 70ps，每种延时结构共有八级，两种结构串联组合使用，模块最大延时时间约为 5ns。

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]：输入延迟功能使能

Bit[8]：输出延迟功能使能

0：不使能 1：使能

Bit[3:1]：输入延迟精调档位选择

Bit[11:9]：输出延迟精调档位选择

假设值为 m，则表示(m+1)*70ps 延迟

Bit [6:4]：输入延迟粗调档位选择

Bit [14:12]：输出延迟粗调档位选择

假设值为 n，则表示(n+1)*530ps 延迟

Bit [7]：保留

Bit [15]：保留

5.13.3.2

5.13.3.3 寄存器说明

表 5-33 通用 IO 引脚延时配置寄存器

偏移地址	位域	说明	复位值
0x0300	[31 : 24]	控制 all_pll_lock_pad 的输出延时	0

	[23 : 16]	控制 all_pll_lock_pad 的输入延时	0
	[15 : 8]	控制 cru_clk_obv_pad 的输出延时	0
	[7 : 0]	控制 cru_clk_obv_pad 的输入延时	0
0x0304	[31 : 24]	控制 tjtag_tdi_pad 的输出延时	0
	[23 : 16]	控制 tjtag_tdi_pad 的输入延时	0
	[15 : 8]	控制 ntrst_swj_pad 的输出延时	0
	[7 : 0]	控制 ntrst_swj_pad 的输入延时	0
0x0308	[31 : 24]	控制 tdi_swj_pad 的输出延时	0
	[23 : 16]	控制 tdi_swj_pad 的输入延时	0
	[15 : 8]	控制 swditms_swj_pad 的输出延时	0
	[7 : 0]	控制 swditms_swj_pad 的输入延时	0
0x030c	[31 : 24]	控制 swdo_swj_pad 的输出延时	0
	[23 : 16]	控制 swdo_swj_pad 的输入延时	0
	[15 : 8]	控制 tdo_swj_pad 的输出延时	0
	[7 : 0]	控制 tdo_swj_pad 的输入延时	0
0x0310	[31 : 24]	控制 hdt_mb_done_state_pad 的输出延时	0
	[23 : 16]	控制 hdt_mb_done_state_pad 的输入延时	0
	[15 : 8]	控制 hdt_mb_fail_state_pad 的输出延时	0
	[7 : 0]	控制 hdt_mb_fail_state_pad 的输入延时	0
0x0314	[31 : 24]	控制 i2c_0_scl_pad 的输出延时	0
	[23 : 16]	控制 i2c_0_scl_pad 的输入延时	0
	[15 : 8]	控制 i2c_0_sda_pad 的输出延时	0
	[7 : 0]	控制 i2c_0_sda_pad 的输入延时	0
0x0318	[31 : 24]	控制 spi0_csn0_pad 的输出延时	0
	[23 : 16]	控制 spi0_csn0_pad 的输入延时	0
	[15 : 8]	控制 spi0_sck_pad 的输出延时	0
	[7 : 0]	控制 spi0_sck_pad 的输入延时	0
0x031c	[31 : 24]	控制 spi0_so_pad 的输出延时	0
	[23 : 16]	控制 spi0_so_pad 的输入延时	0

	[15 : 8]	控制 spi0_si_pad 的输出延时	0
	[7 : 0]	控制 spi0_si_pad 的输入延时	0
0x0320	[31 : 24]	控制 sd_cmd_pad 的输出延时	0
	[23 : 16]	控制 sd_cmd_pad 的输入延时	0
	[15 : 8]	控制 sd_clk_pad 的输出延时	0
	[7 : 0]	控制 sd_clk_pad 的输入延时	0
0x0324	[31 : 24]	控制 sd_dat0_pad 的输出延时	0
	[23 : 16]	控制 sd_dat0_pad 的输入延时	0
	[15 : 8]	控制 sd_dat1_pad 的输出延时	0
	[7 : 0]	控制 sd_dat1_pad 的输入延时	0
0x0328	[31 : 24]	控制 sd_dat2_pad 的输出延时	0
	[23 : 16]	控制 sd_dat2_pad 的输入延时	0
	[15 : 8]	控制 sd_dat3_pad 的输出延时	0
	[7 : 0]	控制 sd_dat3_pad 的输入延时	0
0x032c	[31 : 24]	控制 hda_bclk_pad 的输出延时	0
	[23 : 16]	控制 hda_bclk_pad 的输入延时	0
	[15 : 8]	控制 hda_rst_pad 的输出延时	0
	[7 : 0]	控制 hda_rst_pad 的输入延时	0
0x0330	[31 : 24]	控制 hda_sync_pad 的输出延时	0
	[23 : 16]	控制 hda_sync_pad 的输入延时	0
	[15 : 8]	控制 hda_sdo_pad 的输出延时	0
	[7 : 0]	控制 hda_sdo_pad 的输入延时	0
0x0334	[31 : 24]	控制 hda_sdi0_pad 的输出延时	0
	[23 : 16]	控制 hda_sdi0_pad 的输入延时	0
	[15 : 8]	控制 uart_2_rxd_pad 的输出延时	0
	[7 : 0]	控制 uart_2_rxd_pad 的输入延时	0
0x0338	[31 : 24]	控制 uart_2_txd_pad 的输出延时	0
	[23 : 16]	控制 uart_2_txd_pad 的输入延时	0
	[15 : 8]	控制 uart_3_rxd_pad 的输出延时	0

	[7 : 0]	控制 uart_3_rxd_pad 的输入延时	0
0x033c	[31 : 24]	控制 uart_3_txd_pad 的输出延时	0
	[23 : 16]	控制 uart_3_txd_pad 的输入延时	0
	[15 : 8]	控制 qspi_csn0_pad 的输出延时	0
	[7 : 0]	控制 qspi_csn0_pad 的输入延时	0
0x0340	[31 : 24]	控制 qspi_csn1_pad 的输出延时	0
	[23 : 16]	控制 qspi_csn1_pad 的输入延时	0
	[15 : 8]	控制 qspi_csn2_pad 的输出延时	0
	[7 : 0]	控制 qspi_csn2_pad 的输入延时	0
0x0344	[31 : 24]	控制 qspi_csn3_pad 的输出延时	0
	[23 : 16]	控制 qspi_csn3_pad 的输入延时	0
	[15 : 8]	控制 qspi_sck_pad 的输出延时	0
	[7 : 0]	控制 qspi_sck_pad 的输入延时	0
0x0348	[31 : 24]	控制 qspi_so_io0_pad 的输出延时	0
	[23 : 16]	控制 qspi_so_io0_pad 的输入延时	0
	[15 : 8]	控制 qspi_si_io1_pad 的输出延时	0
	[7 : 0]	控制 qspi_si_io1_pad 的输入延时	0
0x034c	[31 : 24]	控制 qspi_wp_io2_pad 的输出延时	0
	[23 : 16]	控制 qspi_wp_io2_pad 的输入延时	0
	[15 : 8]	控制 qspi_hold_io3_pad 的输出延时	0
	[7 : 0]	控制 qspi_hold_io3_pad 的输入延时	0
0x0350	[31 : 24]	控制 clk_lpc_in_pad 的输出延时	0
	[23 : 16]	控制 clk_lpc_in_pad 的输入延时	0
	[15 : 8]	控制 ext_lpc_ldrq_n_pad 的输出延时	0
	[7 : 0]	控制 ext_lpc_ldrq_n_pad 的输入延时	0
0x0354	[31 : 24]	控制 ext_lpc_irq_n_pad 的输出延时	0
	[23 : 16]	控制 ext_lpc_irq_n_pad 的输入延时	0
	[15 : 8]	控制 ext_lpc_lad_0_pad 的输出延时	0
	[7 : 0]	控制 ext_lpc_lad_0_pad 的输入延时	0

0x0358	[31 : 24]	控制 ext_lpc_lad_1_pad 的输出延时	0
	[23 : 16]	控制 ext_lpc_lad_1_pad 的输入延时	0
	[15 : 8]	控制 ext_lpc_lad_2_pad 的输出延时	0
	[7 : 0]	控制 ext_lpc_lad_2_pad 的输入延时	0
0x035c	[31 : 24]	控制 ext_lpc_lad_3_pad 的输出延时	0
	[23 : 16]	控制 ext_lpc_lad_3_pad 的输入延时	0
	[15 : 8]	控制 phy_gmac0_clk_rx_pad 的输出延时	0
	[7 : 0]	控制 phy_gmac0_clk_rx_pad 的输入延时	0
0x0360	[31 : 24]	控制 gmac0_phy_clk_tx_pad 的输出延时	0
	[23 : 16]	控制 gmac0_phy_clk_tx_pad 的输入延时	0
	[15 : 8]	控制 phy_gmac0_rxd0_pad 的输出延时	0
	[7 : 0]	控制 phy_gmac0_rxd0_pad 的输入延时	0
0x0364	[31 : 24]	控制 phy_gmac0_rxd1_pad 的输出延时	0
	[23 : 16]	控制 phy_gmac0_rxd1_pad 的输入延时	0
	[15 : 8]	控制 phy_gmac0_rxd2_pad 的输出延时	0
	[7 : 0]	控制 phy_gmac0_rxd2_pad 的输入延时	0
0x0368	[31 : 24]	控制 phy_gmac0_rxd3_pad 的输出延时	0
	[23 : 16]	控制 phy_gmac0_rxd3_pad 的输入延时	0
	[15 : 8]	控制 phy_gmac0_rxdv_pad 的输出延时	0
	[7 : 0]	控制 phy_gmac0_rxdv_pad 的输入延时	0
0x036c	[31 : 24]	控制 gmac0_phy_txd0_pad 的输出延时	0
	[23 : 16]	控制 gmac0_phy_txd0_pad 的输入延时	0
	[15 : 8]	控制 gmac0_phy_txd1_pad 的输出延时	0
	[7 : 0]	控制 gmac0_phy_txd1_pad 的输入延时	0
0x0370	[31 : 24]	控制 gmac0_phy_txd2_pad 的输出延时	0
	[23 : 16]	控制 gmac0_phy_txd2_pad 的输入延时	0
	[15 : 8]	控制 gmac0_phy_txd3_pad 的输出延时	0
	[7 : 0]	控制 gmac0_phy_txd3_pad 的输入延时	0
0x0374	[31 : 24]	控制 gmac0_phy_txen_pad 的输出延时	0

	[23 : 16]	控制 gmac0_phy_txen_pad 的输入延时	0
	[15 : 8]	控制 gmac0_phy_mdc_pad 的输出延时	0
	[7 : 0]	控制 gmac0_phy_mdc_pad 的输入延时	0
0x0378	[31 : 24]	控制 gmac0_phy_mdio_pad 的输出延时	0
	[23 : 16]	控制 gmac0_phy_mdio_pad 的输入延时	0
	[15 : 8]	控制 phy_gmac1_clk_rx_pad 的输出延时	0
	[7 : 0]	控制 phy_gmac1_clk_rx_pad 的输入延时	0
0x037c	[31 : 24]	控制 gmac1_phy_clk_tx_pad 的输出延时	0
	[23 : 16]	控制 gmac1_phy_clk_tx_pad 的输入延时	0
	[15 : 8]	控制 ckobv_sel0_pad 的输出延时	0
	[7 : 0]	控制 ckobv_sel0_pad 的输入延时	0
0x0380	[31 : 24]	控制 ckobv_sel1_pad 的输出延时	0
	[23 : 16]	控制 ckobv_sel1_pad 的输入延时	0
	[15 : 8]	控制 ckobv_sel2_pad 的输出延时	0
	[7 : 0]	控制 ckobv_sel2_pad 的输入延时	0
0x0384	[31 : 24]	控制 ckobv_sel3_pad 的输出延时	0
	[23 : 16]	控制 ckobv_sel3_pad 的输入延时	0
	[15 : 8]	控制 ckobv_sel4_pad 的输出延时	0
	[7 : 0]	控制 ckobv_sel4_pad 的输入延时	0
0x0388	[31 : 24]	控制 gmac1_phy_txd0_pad 的输出延时	0
	[23 : 16]	控制 gmac1_phy_txd0_pad 的输入延时	0
	[15 : 8]	控制 gmac1_phy_txd1_pad 的输出延时	0
	[7 : 0]	控制 gmac1_phy_txd1_pad 的输入延时	0
0x038c	[31 : 24]	控制 gmac1_phy_txd2_pad 的输出延时	0
	[23 : 16]	控制 gmac1_phy_txd2_pad 的输入延时	0
	[15 : 8]	控制 gmac1_phy_txd3_pad 的输出延时	0
	[7 : 0]	控制 gmac1_phy_txd3_pad 的输入延时	0
0x0390	[31 : 24]	控制 gmac1_phy_txen_pad 的输出延时	0
	[23 : 16]	控制 gmac1_phy_txen_pad 的输入延时	0

	[15 : 8]	控制 gmac1_phy_mdc_pad 的输出延时	0
	[7 : 0]	控制 gmac1_phy_mdc_pad 的输入延时	0
0x0394	[31 : 24]	reserved	0
	[23 : 16]	reserved	0
	[15 : 8]	控制 gmac1_phy_mdio_pad 的输出延时	0
	[7 : 0]	控制 gmac1_phy_mdio_pad 的输入延时	0
0x0398	[31 : 24]	控制 SWDIO_pad 的输出延时	0
	[23 : 16]	控制 SWDIO_pad 的输入延时	0
	[15 : 8]	控制 SWCLK_pad 的输出延时	0
	[7 : 0]	控制 SWCLK_pad 的输入延时	0

5.13.4 时钟引脚延时配置

5.13.4.1 操作说明

时钟信号可实现 8 个档位延时调节，每级约为 20ps，每个时钟 pad 占用 4 位。

时钟延迟寄存器的位域含义如下：

Bit 0 是使能延迟功能，取值含义如下：

0：不使用时钟延迟功能

1：使能时钟延时功能

Bit [3:1]是延时档位选择位，假设值为 m，则表示(m+1)*20ps 延迟。

5.13.4.2 寄存器说明

表 5-34 时钟引脚延时配置寄存器

偏移	位域	说明	复位值
0x0400	[31 : 29]	控制 tck_swj 的延迟时间档位选择	0
	[28]	控制 tck_swj 的使能延迟功能	0
	[27 : 25]	控制 tjtag_tck 的延迟时间档位选择	0
	[24]	控制 tjtag_tck 的使能延迟功能	0

	[23 : 21]	控制 clk_mpc_clust0 的延迟时间档位选择	0
	[20]	控制 clk_mpc_clust0 的使能延迟功能	0
	[19 : 17]	控制 clk_mpc_clust1 的延迟时间档位选择	0
	[16]	控制 clk_mpc_clust1 的使能延迟功能	0
	[15 : 13]	控制 clk_noc 的延迟时间档位选择	0
	[12]	控制 clk_noc 的使能延迟功能	0
	[11 : 9]	控制 clk_lmu0 的延迟时间档位选择	0
	[8]	控制 clk_lmu0 的使能延迟功能	0
	[7 : 5]	控制 clk_lmu1 的延迟时间档位选择	0
	[4]	控制 clk_lmu1 的使能延迟功能	0
	[3 : 1]	控制 clk_fio0 的延迟时间档位选择	0
	[0]	控制 clk_fio0 的使能延迟功能	0
0x0404	[31 : 29]	reserved	0
	[28]	reserved	0
	[27 : 25]	控制 clk_fio1 的延迟时间档位选择	0
	[24]	控制 clk_fio1 的使能延迟功能	0
	[23 : 21]	控制 clk_gmu 的延迟时间档位选择	0
	[20]	控制 clk_gmu 的使能延迟功能	0
	[19 : 17]	控制 clk_1200 的延迟时间档位选择	0
	[16]	控制 clk_1200 的使能延迟功能	0
	[15 : 13]	控制 clk_sm 的延迟时间档位选择	0
	[12]	控制 clk_sm 的使能延迟功能	0
	[11 : 9]	控制 clk_aux 的延迟时间档位选择	0
	[8]	控制 clk_aux 的使能延迟功能	0
	[7 : 5]	控制 clk_lpc_gated 的延迟时间档位选择	0
	[4]	控制 clk_lpc_gated 的使能延迟功能	0
	[3 : 1]	控制 clk_sio 的延迟时间档位选择	0

	[0]	控制 clk_sio 的使能延迟功能	0

5.13.5 复用功能表

表 5-35 引脚复用功能表

控制域	Func 0	Func 1	Func 2
all_pll_lock_pad	all_pll_lock		i2c_1_scl
cru_clk_obv_pad	cru_clk_obv	gpio0_porta_0	i2c_1_sda
sjtag_tdi_pad		gpio0_porta_1	uart_0_cts_n
sjtag_tms_pad		gpio0_porta_2	uart_0_dcd_n
sjtag_ntrst_pad		gpio0_porta_3	uart_0_dsr_n
sjtag_tdo_pad		gpio0_porta_4	uart_0_ri_n
tjtag_tdo_pad		gpio0_porta_5	uart_0_rts_n
tjtag_ntrst_pad		gpio0_porta_6	uart_0_dtr_n
tjtag_tms_pad		gpio0_porta_7	peu1_linkup_0
tjtag_tdi_pad		can_txd_0	peu1_linkup_1
ntrst_swj_pad	ntrst_swj	can_txd_1	peu1_linkup_2
tdi_swj_pad	tdi_swj	can_txd_2	
swditms_swj_pad	swditms_swj	can_rxd_0	
swdo_swj_pad	swdo_swj	can_rxd_1	i2c_2_scl
tck_swj_in_pad	tck_swj_in		
tdo_swj_pad	tdo_swj	can_rxd_2	i2c_2_sda
hdt_mb_done_state_pad		lpc_ext_irq_out en	i2c_3_scl
hdt_mb_fail_state_pad		lpc_ext_lad_ou ten	i2c_3_sda
i2c_0_scl_pad	i2c_0_scl		
i2c_0_sda_pad	i2c_0_sda		
spi0_csn0_pad	spi0_csn0	gpio1_porta_5	

spi0_sck_pad	spi0_sck	gpio1_porta_6	
spi0_so_pad	spi0_so	gpio1_porta_7	
spi0_si_pad	spi0_si	gpio1_portb_0	
sd_cmd_pad	sd_cmd	gpio1_portb_1	
sd_clk_pad	sd_clk	gpio1_portb_2	
sd_dat0_pad	sd_dat0	gpio1_portb_3	
sd_dat1_pad	sd_dat1	gpio1_portb_4	
sd_dat2_pad	sd_dat2	gpio1_portb_5	
sd_dat3_pad	sd_dat3	gpio1_portb_6	
sd_detect_pad	sd_detect	status_jtagnsw	
hda_bclk_pad	hda_bclk		
hda_rst_pad	hda_rst_n		
hda_sync_pad	hda_sync		
hda_sdo_pad	hda_sdo		
hda_sdi0_pad	hda_sdi0		
uart_0_rxd_pad	uart_0_rxd		
uart_0_txd_pad	uart_0_txd		
uart_1_rxd_pad	uart_1_rxd		
uart_1_txd_pad	uart_1_txd		
uart_2_rxd_pad	uart_2_rxd	spi1_csn0	gpio0_portb_5
uart_2_txd_pad	uart_2_txd	spi1_sck	hda_sdi1
uart_3_rxd_pad	uart_3_rxd	spi1_so	hda_sdi2
uart_3_txd_pad	uart_3_txd	spi1_si	hda_sdi3
qspi_csn0_pad	qspi_csn0		
qspi_csn1_pad	qspi_csn1	gpio1_portb_7	
qspi_csn2_pad	qspi_csn2	spi1_csn1	gpio0_portb_6
qspi_csn3_pad	qspi_csn3	spi1_csn2	gpio0_portb_7
qspi_sck_pad	qspi_sck		
qspi_so_io0_pad	qspi_so_io0		

qspi_si_io1_pad	qspi_si_io1		
qspi_wp_io2_pad	qspi_wp_io2		
qspi_hold_io3_pad	qspi_hold_io3		
clk_lpc_in_pad	clk_lpc_in		
ext_lpc_ldrq_n_pad	ext_lpc_ldrq_n		
ext_lpc_irq_n_pad	ext_lpc_irq_n		
ext_lpc_lad_0_pad	ext_lpc_lad_0	gpio1_porta_3	
ext_lpc_lad_1_pad	ext_lpc_lad_1	gpio1_porta_4	
ext_lpc_lad_2_pad	ext_lpc_lad_2	spi1_csn3	
ext_lpc_lad_3_pad	ext_lpc_lad_3	spi0_csn3	
peu0_linkup0_pad	peu0_linkup_0	pvm0_fout	o_trng_bits_0
peu0_linkup1_pad	peu0_linkup_1	pvm1_fout	o_trng_bits_1
peu0_linkup2_pad	peu0_linkup_2	pvm2_fout	o_trng_bits_2
peu0_c0_clkreq_pad	peu0_c0_clkreq		o_trng_bits_3
peu0_c1_clkreq_pad	peu0_c1_clkreq		o_trng_vld_0
peu0_c2_clkreq_pad	peu0_c2_clkreq		o_trng_vld_1
peu1_c0_clkreq_pad	peu1_c0_clkreq		o_trng_vld_2
peu1_c1_clkreq_pad	peu1_c1_clkreq		o_trng_vld_3
peu1_c2_clkreq_pad	peu1_c2_clkreq		o_trng_samclk_0
phy_gmac0_clk_rx_pad	phy_gmac0_clk_rx		o_trng_samclk_1
gmac0_phy_clk_tx_pad	gmac0_phy_clk_tx		o_trng_samclk_2
phy_gmac0_rxd0_pad	phy_gmac0_rxd0		o_trng_samclk_3
phy_gmac0_rxd1_pad	phy_gmac0_rxd1		trng0_src
phy_gmac0_rxd2_pad	phy_gmac0_rxd2		trng1_src
phy_gmac0_rxd3_pad	phy_gmac0_rxd3		trng2_src
phy_gmac0_rxdv_pad	phy_gmac0_rxdv		trng3_src
gmac0_phy_txd0_pad	gmac0_phy_txd0		trng0_valid
gmac0_phy_txd1_pad	gmac0_phy_txd1		trng1_valid
gmac0_phy_txd2_pad	gmac0_phy_txd2		trng2_valid

gmac0_phy_txd3_pad	gmac0_phy_txd3		trng3_valid
gmac0_phy_txen_pad	gmac0_phy_txen	efuse_prerr_may_repair	
gmac0_phy_mdc_pad	gmac0_phy_mdc	efuse_prerr_not_repair	
gmac0_phy_mdio_pad	gmac0_phy_mdio		
phy_gmac1_clk_rx_pad	phy_gmac1_clk_rx		
gmac1_phy_clk_tx_pad	gmac1_phy_clk_tx		
ckobv_sel0_pad		phy_gmac1_rxd0	gpio0_portb_0
ckobv_sel1_pad		phy_gmac1_rxd1	gpio0_portb_1
ckobv_sel2_pad		phy_gmac1_rxd2	gpio0_portb_2
ckobv_sel3_pad		phy_gmac1_rxd3	gpio1_porta_0
ckobv_sel4_pad		phy_gmac1_rxdv	gpio1_porta_1
gmac1_phy_txd0_pad	gmac1_phy_txd0	peu0_obv_clk	gpio1_porta_2
gmac1_phy_txd1_pad	gmac1_phy_txd1	peu1_obv_clk	spi0_csn1
gmac1_phy_txd2_pad	gmac1_phy_txd2	pcru_state_0	spi0_csn2
gmac1_phy_txd3_pad	gmac1_phy_txd3	pcru_state_1	gpio0_portb_3
gmac1_phy_txen_pad	gmac1_phy_txen	pcru_state_2	gpio0_portb_4
gmac1_phy_mdc_pad	gmac1_phy_mdc	pcru_state_3	
gmac1_phy_mdio_pad	gmac1_phy_mdio	pcru_state_4	

5.13.6 引脚驱动能力

引脚驱动能力决定由 pad_drive_strength N（偏移 0x488）来决定，数值越大去能力越高，而 pin 脚与 DSN 的对应关系如下表所示，DS N 对应着

pad_drive_strength N（例如 DS0 对应 pad_drive_strength 0）

表 5-36 引脚与驱动寄存器对应表

PadName	PAD_CFG
clk_ref_pad	ds0
por_n_pad	ds0
cru_rst_ok_pad	ds1
all_pll_lock_pad	ds1
cru_clk_obv_pad	ds1
SCP_EN_pad	ds0
SB_EN_pad	ds0
TEST_EN_pad	ds0
PWR_CTR0_pad	ds1
PWR_CTR1_pad	ds1
sjtag_tdi_pad	ds1
sjtag_tms_pad	ds1
sjtag_ntrst_pad	ds1
sjtag_tdo_pad	ds1
sjtag_tck_pad	ds0
tjtag_tdo_pad	ds1
tjtag_tck_in_pad	ds0
tjtag_ntrst_pad	ds1
tjtag_tms_pad	ds1
tjtag_tdi_pad	ds2
ntrst_swj_pad	ds2
tdi_swj_pad	ds2
swditms_swj_pad	ds0
swdo_swj_pad	ds1
tck_swj_in_pad	ds0
tdo_swj_pad	ds1

force_mb_start_pad	ds0
hdt_mb_done_state_pad	ds1
hdt_mb_fail_state_pad	ds1
DFT_TEST_CLK1_pad	ds0
DFT_TEST_CLK2_pad	ds0
DFT_TEST_CLK3_pad	ds0
DFT_TEST_CLK4_pad	ds0
DFT_TEST_CLK5_pad	ds0
DFT_TEST_CLK6_pad	ds0
DFT_TEST_CLK7_pad	ds0
DFT_TEST_CLK8_pad	ds0
DFT_TEST_CLK9_pad	ds0
edt_clock_pad	ds0
DFT_TMS_pad	ds0
DFT_TDI_pad	ds0
DFT_TRST_pad	ds0
DFT_TDO_pad	ds2
IP_TEST_pad	ds0
burn_in_pad	ds0
i2c_0_scl_pad	ds1
i2c_0_sda_pad	ds1
spi0_csn0_pad	ds1
spi0_sck_pad	ds4
spi0_so_pad	ds1
spi0_si_pad	ds1
sd_cmd_pad	ds2
sd_clk_pad	ds4
sd_dat0_pad	ds2
sd_dat1_pad	ds2

sd_dat2_pad	ds2
sd_dat3_pad	ds2
sd_detect_pad	ds2
hda_bclk_pad	ds2
hda_rst_pad	ds2
hda_sync_pad	ds2
hda_sdo_pad	ds2
hda_sdi0_pad	ds2
uart_0_rxd_pad	ds0
uart_0_txd_pad	ds1
uart_1_rxd_pad	ds0
uart_1_txd_pad	ds1
uart_2_rxd_pad	ds1
uart_2_txd_pad	ds4
uart_3_rxd_pad	ds2
uart_3_txd_pad	ds2
qspi_csn0_pad	ds3
qspi_csn1_pad	ds3
qspi_csn2_pad	ds3
qspi_csn3_pad	ds3
qspi_sck_pad	ds4
qspi_so_io0_pad	ds3
qspi_si_io1_pad	ds3
qspi_wp_io2_pad	ds3
qspi_hold_io3_pad	ds2
clk_lpc_in_pad	ds0
lpc_ext_rstn_o_pad	ds1
ext_lpc_ldrq_n_pad	ds0
lpc_ext_lframe_n_pad	ds1

ext_lpc_irq_n_pad	ds1
ext_lpc_lad_0_pad	ds2
ext_lpc_lad_1_pad	ds2
ext_lpc_lad_2_pad	ds2
ext_lpc_lad_3_pad	ds2
peu0_linkup0_pad	ds2
peu0_linkup1_pad	ds2
peu0_linkup2_pad	ds2
peu0_c0_clkreq_pad	ds2
peu0_c1_clkreq_pad	ds2
peu0_c2_clkreq_pad	ds2
peu1_c0_clkreq_pad	ds2
peu1_c1_clkreq_pad	ds2
peu1_c2_clkreq_pad	ds2
phy_gmac0_clk_rx_pad	ds2
gmac0_phy_clk_tx_pad	ds4
phy_gmac0_rxd0_pad	ds2
phy_gmac0_rxd1_pad	ds2
phy_gmac0_rxd2_pad	ds2
phy_gmac0_rxd3_pad	ds2
phy_gmac0_rxdv_pad	ds2
gmac0_phy_txd0_pad	ds3
gmac0_phy_txd1_pad	ds3
gmac0_phy_txd2_pad	ds3
gmac0_phy_txd3_pad	ds3
gmac0_phy_txen_pad	ds3
gmac0_phy_mdc_pad	ds3
gmac0_phy_mdio_pad	ds3
phy_gmac1_clk_rx_pad	ds0

gmac1_phy_clk_tx_pad	ds4
ckobv_sel0_pad	ds2
ckobv_sel1_pad	ds2
ckobv_sel2_pad	ds2
ckobv_sel3_pad	ds2
ckobv_sel4_pad	ds2
gmac1_phy_txd0_pad	ds3
gmac1_phy_txd1_pad	ds3
gmac1_phy_txd2_pad	ds3
gmac1_phy_txd3_pad	ds3
gmac1_phy_txen_pad	ds3
gmac1_phy_mdc_pad	ds3
gmac1_phy_mdio_pad	ds3
SWDIO_pad	ds1
SWCLK_pad	ds0
DFT_wrp_clk_pad	ds0
peu01_phy01_jtag_tdi_pad	ds0
peu01_phy01_jtag_tms_pad	ds0
peu01_phy01_jtag_trst_n_pad	ds0
peu01_phy01_jtag_tck_pad	ds0
peu0_phy0_jtag_tdo_pad	ds1
peu0_phy1_jtag_tdo_pad	ds1
peu1_phy0_jtag_tdo_pad	ds1
peu1_phy1_jtag_tdo_pad	ds1
cru_scan_clk_pad	ds0
cru_clk_sel_pad	ds0

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 中断均为低电平有效，相应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
31		保留
30	CNTPNSIRQ	非安全的物理定时器
29	CNTPSIRQ	安全的物理定时器
28		保留
27	CNTVIRQ	虚拟定时器
26	CNTHPIRQ	hypervisor 定时器
25	VCPUMNTIRQ	虚拟 CPU 接口管理中断
24	CTIIRQ	CTI 中断
23	PMUIRQ	PMU 溢出中断
22	COMMIRQ	DCC 中断
21		保留
20		保留
19		保留
18		保留
17		保留
16		保留

6.2 SPI 中断

SPI 中断均为高电平有效，相应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

ID	信号名	含义
32	misc_ras_er_spi	l3c, lmu 和 peu er_spi 的或
33	misc_ras_fh_spi	l3c, lmu 和 peu fi_spi 的或
34	misc_ras_er_spi_s	l3c, lmu 和 peu er_spi_s 的或
35	misc_ras_fh_spi_s	l3c, lmu 和 peu fi_spi_s 的或
37	lpc_ICU_int	收到 lpc 设备串行中断
38	uart_0_Intr_irq	串口 0 中断
39	uart_1_Intr_irq	串口 1 中断
40	uart_2_Intr_irq	串口 2 中断
41	uart_3_Intr_irq	串口 3 中断
42	gpio0_Intr_irq	GPIO0 中断
43	gpio0_Intr_irq	GPIO 1 中断
44	i2c_0_Intr_irq	i2c 0 中断
45	i2c_1_Intr_irq	i2c 1 中断
46	i2c_2_Intr_irq	i2c 2 中断
47	i2c_3_Intr_irq	i2c 3 中断
48	wdt_0_ICU_Intr_irq	看门狗内部计数器计数到零
49	wdt_1_ICU_Intr_irq	看门狗内部计数器计数到零
50	spim0_ssi_Intr_irq	SPIM0 中断
51	spim1_ssi_Intr_irq	SPIM 1 中断
52	sd_dma_int	SD 控制器 DMA 相关中断
53	sd_cmd_int	SD 控制器命令处理相关中断，包括卡检测中断
54	sd_err_int	SD 控制器 err 相关中断
55	hda_int	HDAudio 控制器相关中断
56	lmu0_intr	LMU0 中断，Scrub 完成等中断信息
57	Lmu1_intr	LMU 1 中断，Scrub 完成等中断信息
58	peu01_msg_int_spi	接收到一个消息

59	peu01_misc_int_spi	Peu 0/1 内发生特殊事件
60	peu01_inta_spi	Peu 0/1 收到 inta 中断
61	peu01_intb_spi	Peu 0/1 收到 intb 中断
62	peu01_intc_spi	Peu 0/1 收到 intc 中断
63	peu01_intd_spi	Peu 0/1 收到 intd 中断
64	peu01_msi_spi	Peu 0/1 走 spi 通路的 msi 中断
65	peu0_mas_spi	Peu 0 MAS 模块输出的中断
66 -72	RSV 7-bit	
73	peu1_mas_spi	Peu 1 MAS 模块输出的中断
74	trng0_int	真随机数 0 中断
75	trng1_int	真随机数 1 中断
76	trng2_int	真随机数 2 中断
77	trng3_int	真随机数 3 中断
78	int_secure_scp	SCP 向 CPU 传输消息，Secure 通道中断
79	int_uboot_scp	SCP 向 CPU 传输消息，Uboot 通道中断
80	int_os_scp	SCP 向 CPU 传输消息，OS 通道中断
81	gmac0_ieu_sbd_intr_o	Gmac0 中断
82	gmac1_ieu_sbd_intr_o	Gmac 1 中断
83	cru_ICU_temp_sensor0_int	温度传感器 0 超出阈值
84	cru_ICU_temp_sensor1_int	温度传感器 1 超出阈值
85	vc_irq	TDBG vc 组件中断
86	DBGWATCHTRIGREQ	
118-87	STMHWEVENT[31:0]	

119	hds_can0_intr	Can0 组件中断
120	~wdt_0_cru_sys_rst_n	Wdt0 复位
121	~wdt_1_cru_sys_rst_n	Wdt1 复位
122		
123	hds_can1_intr	Can1 组件中断
124	hds_can2_intr	Can2 组件中断
159-125	RSV 35-bit	

7 功耗管理

FT-2000/4 处理器将采用多种低功耗设计技术，包括时钟门控、DFS、电源关断等。

7.1 CPU 功耗管理

FT-2000/4 支持处理器的多种功耗管理技术，并通过 ARM 定义的 SCPI（System Control and Power Interface）[2]接口和 PSCI（Power State Corodination Interface）[3] 供系统功耗管理软件调用。

- 动态调频

实现 core 运行频率的动态调节。通过 SCPI 接口，可以查询 CPU 支持的频率点集合，以及实现频率的动态切换。FT-2000/4 包含 4 个 core，每 2 个 core 为 1 个 cluster。Core 的动态调频以 cluster 为操作对象，即 cluster 中的两个 core 互相关联，修改某个 core 的运行频率，同一个 cluster 内的另外一个 core 也将保持运行频率同步。

- Core 挂起

FT-2000/4 支持时钟门控技术。Core 执行 wfi 指令，即可实现当前 core 的暂停执行，进入低功，耗状态。通过中断或事件，可以唤醒挂起的 core。

- Core 动态开关

FT-2000/4 包含 4 个 core，支持以 core 为单位进行动态关闭（OFF）与唤醒（ON）。功耗管理软件通过调用 PSCI 接口实现 Core 的动态 OFF/ON。

- 温度传感器

FT-2000/4 内置 2 个温度传感器。系统软件通过 SCPI 接口查询并控制温度传感器。

7.2 系统功耗管理

FT-2000/4 支持系统级动态功耗管理，并通过飞腾基础固件（Phytium Base Firmware，PBF）提供标准 PSCI 接口，实现整个系统的待机、休眠、关机、重启等功能。上层系统软件，比如，操作系统，可以调用 PSCI 接口实现相关功能，

无需关注主板或 CPU 的实现细节。

- 待机

系统运行状态保存在 DDR 中。除唤醒源和 DDR 外，主板上所有部件断电，DDR 进入自刷新节电模式。

- 休眠、重启与关机

休眠（Suspend to Disk）、重启、关机与传统操作系统的概念一致，只是对系统电源状态的控制（重启、关机）由标准 PSCI 接口来实现。

参考文档

[1] ARM, Architecture Reference Manual ARMv8, for ARMv8-A architecture profile, version D.

[2] ARM, ARM Compute Subsystem SCP Message Interface Protocols, version 1.2.

[3] ARM, ARM Power State Coordination Interface Platform Design Document, version 1.1.