飞腾 X100 套片
软件编程手册

(V1.3)

2022 年 11 月

飞腾信息技术有限公司

www.phytium.com.cn

版权声明：

免责声明：

当前版本

文件标识	
当前版本	**1.3**
完成日期	2022.11.2

版本历史

版本	修订时间	修订人	修订内容
V0.5	202104		
V0.8	202105		修改文中个别描述； 在 5.4.5 节中增加原始数据与实际数据的计算关系图； 统一寄存器多种取值时的描述，采用“取值：描述”格式：如果只有 2 种取值，且描述较短时在一行内进行描述，中间用分号隔开；如果有 3 种以上（含）时分多行描述； 将所有手动换行符替换为段落标记。
V1.0	202110		第 2 章：修改 PCI 设备 class 号；增加 2.3 节飞腾 X100 产品形态标识； 第 5 章：删除 HDCP 和 MST 相关内容； 第 6 章：删除 USB_IRQ_HANDLE 寄存器；删除原第 9 章及 LPC 相关描述； 第 9 章：补全寄存器列表；增加时钟配置参考； 第 10 章：补全 NAND Flash Controller 寄存器列表； 第 11 章：增加 CAN 操作说明，删除 CAN_DMA_CTRL 寄存器，修改寄存器说明； 第 12 章：修改 PWM 操作说明，增加 PWM 寄存器说明； 删除原第 14 章及 HDA 相关描述； 第 13 章：增加 I2C 接口频率调整公式； 第 16 章：改为“2 个 32 位引脚 GPIO 控制器”，删除“A 组”描述； 第 17 章：增加 SMBUS 操作说明、寄存器列表、寄存器描述； 第 18 章：增加 SPI 操作说明； 第 19 章：删除 I2S 操作说明中的 Slave 模式，

			补充 I2S 寄存器说明； 修复其它问题。
V1.1	202202		将 3.3V 修改为 2.5V；统一 UART 操作说明与寄存器说明中 UARTIFLS 的偏移地址；修改 MIO 全局控制寄存器基地址。
V1.2	202209		第 2 章：将 X100 产品形态值存储位置由 SS 寄存器的[23:16]位修改为[7:0]位； 第 5 章：修改 5.2.1 地址映射配置流程；修改 5.4.2 光标大小； 第 16 章：删除 16.1.2，删除 GPIO 中断相关寄存器；修改 GPIO_SWPORTA_DDR 寄存器 Port direction Register 位域为 7:0； 第 18 章：修改 TXFTLR 与 RXFTLR 寄存器描述； 第 20 章：新增“信号幅值调试寄存器”章节。
V1.3	202211		新增 NPU 章节。

目录

图目录

表目录

1 概述

飞腾 X100 是一款主 CPU 配套芯片，其主要功能是实现 GPU 系统和 PCIE、USB、SATA 等高速接口扩展，同时实现整机系统的上下电控制等功能，可参考《飞腾 X100 套片数据手册》。

表 1-1 术语和缩略语表

术语	全称	解释
AMBA	Advanced Microcontroller Bus Architecture	高级微控制器总线体系结构
APB	Advanced Peripheral Bus	高级外围总线，AMBA 的慢速总线
ASTC	Adaptive Scalable Texture Compression	自适应扩展纹理压缩
AUX	Auxiliary	音频输入接口
AVS	Audio Video coding Standard	音视频编码标准
AXI	Advanced eXtensible Interface	高级可扩展接口，AMBA 的高速总线
CPU	Central Processing Unit	中央处理器
DC_REQ	Display controller requestor	显示控制器请求
DDR	Double Data Rate SDRAM	双倍速率同步动态随机存储器
DMA	Direct memory access	直接访问内存
DP	DisplayPort	显示接口
eMMC	Embedded Multi Media Card	内嵌式多媒体存储卡
FBDC	Frame buffer decompressor	解压缩模块
GPIO	General-Purpose Input/Output	通用输入/输出接口
GPU	Graphics Processing Unit	图形处理器
HDMI	High Definition Multimedia Interface	高清晰度多媒体接口
HEVC	High Efficiency Video Coding	高效率视频编码
HPD	Hot plug detect	DP 热拔插检测
I2C	Inter-Integrated Circuit	两线式串行总线
I2S	Inter-IC Sound	集成电路内置音频总线
LPDDR	Low Power Double Data Rate SDRAM	低功耗双倍速率同步动态随机存储器
MIO	Multiple Input/Output	一种多功能输入/输出接口
MMD	Multimedia display	多媒体显示
MMCSD	Multi Media Card/ Secure Digital Memory Card	多媒体卡/数字安全记忆卡二合一读卡器
MPEG	Moving Picture Experts Group	动态图像专家组
NandFlash	NAND Flash	NAND 闪存
NC	No Connect	无连接，不使用状态
NPU	Neural Processor Unit	神经加速网络
ONFI	Open NAND Flash Interface	开放式 NAND 闪存接口

PCIE	Peripheral Component Interconnect Express	高速串行计算机扩展总线标准
USB PD	USB Power Delivery	USB 功率传输协议
PS2	PS2	一种计算机输入装置接口，用于连接鼠标和键盘
SATA	Serial Advanced Technology Attachment	一种串行硬件驱动器接口
SD	Secure Digital Memory Card	安全数字存储
SIMD	Single Instruction Multiple Data	单指令多数据流
SIMT	Single Instruction Multiple Threads	单指令多线程
SMBus	System Management Bus	系统管理总线
SPI	Serial Peripheral Interface	串行外设接口
SE	Secure Engine	安全引擎固件
UART	Universal Asynchronous Receiver/Transmitter	通用异步收发器
USB	Universal Serial Bus	通用串行总线
VC1	Video Codec 1	VC1 视讯编解码器

2 飞腾 X100 PCI 功能概述

飞腾 X100 主要实现为多功能 PCI 设备，通过一个 PCI3.0 Switch 分出多个 PCIE 总线，每个 PCIE 总线下实现一个多功能 PCIE 设备。

2.1 PCI 功能号分配

飞腾 X100 各个下行 PCIE 链路总线下的 PCIE 功能号分配如表 2-1 所示，其中每个 EEP 表示一个独立的 PCIE 设备，每个 EEP 下分为多个功能号。

表 2-1 功能描述

EEP	功能	说明	备注
x16	GPU	GPU	
	VPU	VPU 视频解码模块	
	DC	显示控制模块	
	I2S	音频输出接口	
	NPU	NPU	
x4	SATAx4	1 个 SATA 控制器，4 个 SATA 接口	func0 实现 4 个 SATA 接口。 func1 实现 PS 接口。 func2 实现 PS 接口。
	PS2_0	PS2 控制器	
	PS2_1	PS2 控制器	
x8	USBx8	8 个 USB3.1 控制器	按 8 个 function 来实现。
x1	MMCSD0	MMC SD 控制器 0	LSD EEP function 实现一个功能。 地址空间划分需要注意。
	MMCSD1	MMC SD 控制器 1	
	NANDFLASH	NAND FLASH 控制器	
	I2S	I2S 音频控制器	
	SPIM0	SPI master 控制器 0	
	SPIM1	SPI master 控制器 1	
	PWM0	PWM 控制器 0	
	PWM1	PWM 控制器 1	
	PWM2	PWM 控制器 2	
	PWM3	PWM 控制器 3	
	9 线 UART0	9 线串口控制器 0	
	9 线 UART1	9 线串口控制器 1	
	9 线 UART2	9 线串口控制器 2	
	9 线 UART3	9 线串口控制器 3	
	MIO0	UART/I2C/PWM 控制器 0	
	MIO1	UART/I2C/PWM 控制器 1	
	MIO2	UART/I2C/PWM 控制器 2	
	MIO3	UART/I2C/PWM 控制器 3	
	MIO4	UART/I2C/PWM 控制器 4	
	MIO5	UART/I2C/PWM 控制器 5	

	MIO6	UART/I2C/PWM 控制器 6	
	MIO7	UART/I2C/PWM 控制器 7	
	GPIO0~1	2 组 GPIO 控制器	
	CAN0	CAN 控制器 0	
	CAN1	CAN 控制器 1	
	SMBUS0	SMBUS 总线 0	
	SMBUS1	SMBUS 总线 1	
	PS2_0	PS2 控制器 0	
	PS2_1	PS2 控制器 1	
	LDMA	DMA 控制器	
	LDMA_BDL	DMA 控制器	
	LSD_CFG	LSD 全局控制	

2.2 PCI 设备号分配

飞腾 X100 的各个 PCI 功能的厂商 ID（Vendor ID）统一为 1DB7。

各个 PCI 功能的设备 ID（Device ID）如下表所示。

表 2-2　各 PCI 功能号的设备 ID 和 class 号

DEVICE ID	名称	class	sub_class	prog
DC20	GPU	0x0B	0x40	0x00
DC21	VPU	0x04	0x00	0x00
DC22	DC	0x03	0x80	0x00
DC23	I2S/DMA	0x04	0x01	0x00
DC24	NPU	0x0B	0x04	0x00
DC26	SATA	0x01	0x06	0x01
DC27	USB	0x0C	0x03	0x30
DC28	MMCSD	0x08	0x05	0x00
DC29	NANDFLASH	0x01	0x09	0x00
DC2B	I2S	0x04	0x01	0x00
DC2C	SPIM	0x0C	0x80	0x00
DC2D	CAN	0x0C	0x09	0x00
DC2E	UART	0x07	0x00	0x00
DC2F	PWM	0x08	0x80	0x00
DC30	MIO	0x07	0x02	0x00
DC31	GPIO	0x08	0x80	0x00
DC32	SMBUS	0x0C	0x05	0x00
DC34	PS	0x08	0x80	0x00
DC36	LDMA	0x08	0x80	0x00
DC38	LSD_CFG	0x08	0x80	0x00
DC3A	SWITCH	0xFF	0x04	0x00
DC3C	GPU_DMA	0x08	0x80	0x00

2.3 飞腾 X100 产品形态标识

飞腾 X100 有多种产品形态（详见《飞腾 X100 套片数据手册》），产品形态可以根据 X100 中 GPU 设备 PCI header 的 SS（Sub System Identifiers）寄存器（偏移量为 0x2C）[7:0]位的值确定。

表 2-3　飞腾 X100 产品形态标识对照表

值	产品形态
0x00	X100
0x01	X100 标准版
0x02	X100 基础版
0x03	X100 工业版
0x04	X100 工业版(无 GPU)
0x05	X100 移动版(无盖)
0x06	X100 标准版(无盖)
0x07	X100 基础版(无盖)

2.4 PCI 资源分配

本节介绍飞腾 X100 的各个 PCI 设备的总体资源分配，主要是内存映射的寄存器空间大小，各设备 BAR 基址通过在系统中执行 lspci 命令扫描获得。对于包含独立显存的 PCI 设备，如 GPU、DC 等，显存空间在对应的部件章节介绍。

2.4.1　PCIE x16 链路

表 2-4　PCIE x16 设备资源分配

功能号	部件	地址空间大小	相对 BAR 基址的偏移范围
func0	GPU slave	1MB	0x00_0000_0000 ~ 0x00_000f_ffff
	地址变换	4KB	0x00_0010_1000 ~ 0x00_0010_1fff
func1	VPU slave	64KB	0x00_0000_0000 ~ 0x00_0000_ffff
	地址变换	4KB	0x00_0001_0000 ~ 0x00_0001_0fff
	VPU 自定义 slave	4KB	0x00_0001_1000 ~ 0x00_0001_1fff
func2	DC0 slave	8KB	0x00_0000_0000 ~ 0x00_0000_1fff
	DC0_REQ	4KB	0x00_0000_2000 ~ 0x00_0000_2fff
	DP0	4KB	0x00_0000_3000 ~ 0x00_0000_3fff
	地址变换	4KB	0x00_0000_4000 ~ 0x00_0000_4fff
	phy_dc0_st	4KB	0x00_0000_5000 ~ 0x00_0000_5fff
	DC1 slave	8KB	0x00_0000_8000 ~ 0x00_0000_9fff
	DC1_REQ	4KB	0x00_0000_a000 ~ 0x00_0000_afff
	DP1	4KB	0x00_0000_b000 ~ 0x00_0000_bfff
	地址变换	4KB	0x00_0000_c000 ~ 0x00_0000_cfff
	phy_dc1_st	4KB	0x00_0000_d000 ~ 0x00_0000_dfff

	DC2 slave	8KB	0x00_0001_0000 ~ 0x00_0001_1fff
	DC2_REQ	4KB	0x00_0001_2000 ~ 0x00_0001_2fff
	DP2	4KB	0x00_0001_3000 ~ 0x00_0001_3fff
	地址变换	4KB	0x00_0001_4000 ~ 0x00_0001_4fff
	phy_dc2_st	4KB	0x00_0001_5000 ~ 0x00_0001_5fff
func5	DMA_slave	4KB	0x00_0000_0000 ~ 0x00_0000_0fff
	I2S slave	4KB	0x00_0000_1000 ~ 0x00_0000_1fff
	DMA_slave	4KB	0x00_0000_2000 ~ 0x00_0000_2fff
	I2S slave	4KB	0x00_0000_3000 ~ 0x00_0000_3fff
	DMA_slave	4KB	0x00_0000_4000 ~ 0x00_0000_4fff
	I2S slave	4KB	0x00_0000_5000 ~ 0x00_0000_5fff
func6	NPU slave	1MB	0x00_0000_0000 ~ 0x00_000f_ffff
	地址变换	4KB	0x00_0010_0000 ~ 0x00_0010_0fff

2.4.2 PCIE x8 链路

表 2-5 PCIE x8 设备资源分配

功能号	部件	地址空间大小	相对 BAR 基址的偏移范围
func0	USB0	128KB	0x00_0000_0000 ~ 0x00_0001_ffff
func1	USB1	128KB	0x00_0000_0000 ~ 0x00_0001_ffff
func2	USB2	128KB	0x00_0000_0000 ~ 0x00_0001_ffff
func3	USB3	128KB	0x00_0000_0000 ~ 0x00_0001_ffff
func4	USB4	128KB	0x00_0000_0000 ~ 0x00_0001_ffff
func5	USB5	128KB	0x00_0000_0000 ~ 0x00_0001_ffff
func6	USB6	128KB	0x00_0000_0000 ~ 0x00_0001_ffff
func7	USB7	128KB	0x00_0000_0000 ~ 0x00_0001_ffff

2.4.3 PCIE x4 链路

表 2-6 PCIE x4 设备资源分配

功能号	部件	地址空间大小	相对 BAR 基址的偏移范围
func0	SATAx4	16K	0x00_0000_0000 ~ 0x00_0000_3fff
func1	PS2_0	4K	0x00_0000_0000 ~ 0x00_0000_0fff
func2	PS2_1	4K	0x00_0000_0000 ~ 0x00_0000_0fff

2.4.4 PCIx1 链路

表 2-7 PCI x1 设备资源分配

功能号	部件	地址空间大小	相对 BAR 基址的偏移范围
func1	MMCSD0	4K	0x00_0000_0000 ~ 0x00_0000_0fff
func2	MMCSD1	4K	0x00_0000_0000 ~ 0x00_0000_0fff
func3	NANDFLASH	4K	0x00_0000_0000 ~ 0x00_0000_0fff
func5	CAN0	4K	0x00_0000_0000 ~ 0x00_0000_0fff
func6	CAN1	4K	0x00_0000_0000 ~ 0x00_0000_0fff
func7	PWM0	4K	0x00_0000_0000 ~ 0x00_0000_0fff

func8	PWM1	4K	0x00_0000_0000 ~ 0x00_0000_0fff
func9	PWM2	4K	0x00_0000_0000 ~ 0x00_0000_0fff
func10	PWM3	4K	0x00_0000_0000 ~ 0x00_0000_0fff
func11	9 线 UART0	4K	0x00_0000_0000 ~ 0x00_0000_0fff
func12	9 线 UART1	4K	0x00_0000_0000 ~ 0x00_0000_0fff
func13	9 线 UART2	4K	0x00_0000_0000 ~ 0x00_0000_0fff
func14	9 线 UART3	4K	0x00_0000_0000 ~ 0x00_0000_0fff
func15	MIO0	4K	0x00_0000_0000 ~ 0x00_0000_0fff
func16	MIO1	4K	0x00_0000_0000 ~ 0x00_0000_0fff
func17	MIO2	4K	0x00_0000_0000 ~ 0x00_0000_0fff
func18	MIO3	4K	0x00_0000_0000 ~ 0x00_0000_0fff
func19	MIO4	4K	0x00_0000_0000 ~ 0x00_0000_0fff
func20	MIO5	4K	0x00_0000_0000 ~ 0x00_0000_0fff
func21	MIO6	4K	0x00_0000_0000 ~ 0x00_0000_0fff
func22	MIO7	4K	0x00_0000_0000 ~ 0x00_0000_0fff
func23	LSD_CFG	4K	0x00_0000_0000 ~ 0x00_0000_0fff
func24	LDMA	4K	0x00_0000_0000 ~ 0x00_0000_0fff
func25	GPIO0~1	4K	0x00_0000_0000 ~ 0x00_0000_0fff
func26	SMBUS0	4K	0x00_0000_0000 ~ 0x00_0000_0fff
func27	SMBUS1	4K	0x00_0000_0000 ~ 0x00_0000_0fff
func28	SPIM0	4K	0x00_0000_0000 ~ 0x00_0000_0fff
func29	SPIM1	4K	0x00_0000_0000 ~ 0x00_0000_0fff
func30	I2S	4K	0x00_0000_0000 ~ 0x00_0000_0fff

3 GPU

3.1 简介

GPU 是一种通用的图形处理器，基于 SIMD/SIMT 的体系结构设计，使得 GPU 非常适合处理图形图像渲染的计算任务，同时采用统一可编程渲染架构，使 GPU 能够很好地扩展到科学计算领域，特别是对于具有大量并行特征的计算，GPU 能够提供比 CPU 无可比拟的高性价比算力。GPU 主要用于 2D 渲染、3D 渲染、通用计算等。

飞腾 X100 中 GPU 支持通用图形和通用计算 API，包括 OpenGL3.3、OpenGL ES 3.2、OpenCL 3.0 和 Vulkan1.2。

3.2 地址空间

对于 GPU 部件，地址空间包括两部分：

内存映射的寄存器空间：通过 PCI 的 bar0 确定，其总大小为 2MB。根据寄存器功能，进一步细分如下表所示。其中 GPU slave 表示 GPU 核心寄存器，地址变换部分主要包括 GPU 地址映射寄存器。

表 3-1 GPU 寄存器空间分配表

部件	地址空间大小	相对 BAR 基址的偏移范围
GPU Slave	1MB	0x00_0000_0000 ~ 0x00_000f_ffff
地址变换	4KB	0x00_0010_1000 ~ 0x00_0010_1fff

独立显存空间：通过 PCI 的 bar2 确定，其大小由固件配置，如 2GB。

GPU 的独立显存空间可全部映射到系统 64 位内存空间内。CPU 使用系统地址访问显存，GPU 使用显存局部地址访问显存。为使用独立显存空间，软件（驱动）需配置 GPU 的地址映射寄存器，地址映射部件负责将系统内存空间中的显存系统地址转换成显存局部物理地址。

3.2.1 地址映射配置流程

GPU 可使用主存或显存进行渲染。如果要使用显存，需要为 GPU 从 DDR 显存中预分配一定大小（如 2G）专门供 GPU 使用，预分配由飞腾 X100 的固件设置，然后进行地址映射。

地址映射配置过程如下：

- 当 OS 通过 PCI 扫描到 GPU 时，会为其独立显存空间分配 PCI mem64 资源，GPU 驱动获得该 mem64 资源的地址值和大小。
- 将地址值右移 22 位，然后写入 region0_src_addr 寄存器。
- 将大小值左移 22 位，且最高位（31 位）设置为 1 后，写入 region0_size 寄存器。

配置完地址映射后，CPU 就可使用系统分配的 PCI mem64 资源地址访问显存。

3.3 操作原理

本节以 3D 图形渲染为例，介绍飞腾 X100 中 GPU 的操作原理。

3.3.1 3D 渲染基础

3D 渲染的目的是将场景中的 3D 几何模型转换成 2D 图像，然后用于输出。根据应用场景，在系统主存或显存中构建 3D 渲染需要的数据（如顶点、纹理）和命令（如着色器）的数据结构，GPU 驱动负责将这些高层数据结构转换成适合 GPU 执行的命令和数据格式，然后配置 GPU 硬件寄存器，从而启动硬件 3D 渲染流程。

下图是基于 OpenGL 的一个简单的 3D 渲染过程示例。

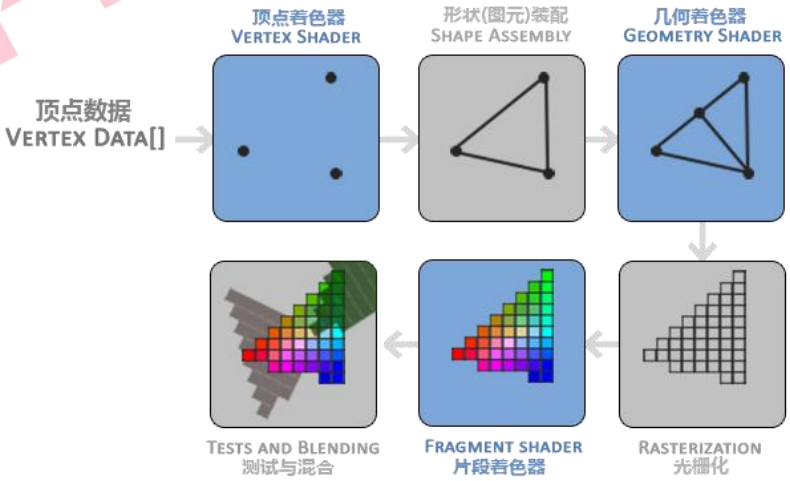

图 3-1　3D 渲染过程示例

在飞腾 X100 的 GPU 中，3D 图渲染主要分两个阶段：几何处理阶段、片元处理阶段。几何处理阶段进行几何处理和顶点操作，如转换和光照；片元处理阶段处理光栅化、纹理和片元着色等。

3.3.2 几何处理阶段

几何处理阶段主要执行下列操作：

- **输入装配**：从缓冲读入图元数据，并将数据装配成其它流水线阶段的图元。
- **顶点着色**：对每个输入顶点运行着色器代码，并产生单个输出顶点。
- **Hull 着色**：对每个包（一组顶点）运行着色器，产生逐顶点的数据和逐包的数据，供 TESSELLATOR（细分）和域着色器使用。
- **Tessellation**：细分，将一个图元细分成一组更小的对象（三角形、顶点、线段），及对象的顶点位置。
- **域着色**：对细分后的对象的每个顶点运行着色器，产生单个输出顶点。
- **几何着色**：对完整的图元（点、线、三角形）运行着色器，根据输入顶点生成输出顶点。
- **视口转换**：将对象转换到视口坐标系统。
- **透视分割**（Perspective Divide）：将 3D 对象投射到 2D 空间（归一化的设备坐标空间），从而较远的对象在屏幕上显示较小。
- **早期图元 culling**：裁剪掉没有位于可视区域内的图元。
- **裁剪 Clipping**: 将位于可视区域边界的图元划分为多个图元。
- **流输出**（Stream Out）：也称转换反馈（transform feedback）。连续输出来自几何处理阶段的顶点数据。使能多遍（multi-pass）渲染操作。
- **生成分块显示列表**（generate tiled display lists）：将与 tile 重叠的图元引用写入到 tile 列表，用于片元处理阶段。

输入装配为后续阶段准备图元数据（通过将图元分割成单个顶点、线和三角形）。输入装配还提交着色器任务到顶点着色和几何着色阶段。对象在视口转换阶段转换成视口坐标系统，然后进入早期图元 culling 阶段，裁剪不必要的图元。转换后的几何图元输出到参数缓冲（PB）。最后的 Tiling 阶段在 PB 创建 tile 列表，在片元处理阶段对 tile 列表逐 tile 执行光栅化。

对于一次具体的几何处理来说，上述流水线阶段不是全部都需要，例如最简单的情景可只配置顶点着色阶段，而 Hull 着色、Tessellation、域着色、几何着色器等阶段全部禁用。

3.3.3 片元处理阶段

片元处理阶段执行 3D 渲染中的下列操作：

- **获取分块的显示列表**：读取几何处理阶段生成的显式列表中的分块（tile）数据。
- **获取几何数据**：读取顶点数据，包括顶点属性和顶点常量。
- **depth biasing**：增加 z-bias。
- **光栅化**：为片元着色器准备图元并确定如何调用片元着色器，基于块进行。
- **隐藏表面消除**：基于块（tile）执行早期隐藏表面消除。
- **深度测试和模板测试**：基于模板测试和深度测试，确定是否绘制一个片元。
- **纹理获取**：为片元着色采样纹理数据。
- **片元着色**：运行着色器代码，逐片元应用着色技术，如光照、后处理。
- **颜色混合（blending）**：调整片元着色器和渲染目标的值。混合将一个或多个片元值组合生成最终的片元颜色。

分块列表获取阶段从 3D 显示列表中一个一个地获取块。几何获取阶段获取转换后的几何数据，根据需要传递给下一阶段。光栅化阶段负责为片元着色器准备图元，该阶段将每个图元分解成多个单独的元素，并确定如何调用片元着色器。隐藏表面消除（HSR）、深度和模板测试使用片上缓冲，通常在片元处理阶段（这部分即 TBDR 的延迟渲染部分）之前执行。

片元处理阶段运行片元着色器，负责获取纹理数据和应用着色技术（如逐片元光照和后处理）。Iterator 负责循环提供片元着色器代码需要的片元数据。片元着色器混合一个或多个片元值创建最终的像素颜色。经片元着色器调整后的像素值存储在片上块缓冲（on-chip tile buffer）中，然后传送到渲染缓冲或直接传递给帧缓冲。

3.4 寄存器说明

GPU 系统寄存器主要分为：GPU 状态与控制寄存器、地址映射寄存器。GPU 系统寄存器的基地址为：PCI Bar0 对应的基址+0x0010_1000。

3.4.1 GPU 状态与控制寄存器

3.4.1.1 gpu_info(0x400)

位	读写	复位值	描述
31:0	RW	0x0	GPU 相关信息，由 SE 在启动时配置和定义

3.4.1.2 gpu_status&ctrl 寄存器(0x404)

位	读写	复位值	描述
31:28	RO	0x0	保留
30:19	RO	0x0	保留
18	RO	0x0	保留
17	RO	0x0	GPU PLL 调频请求响应，拉高时表示 PLL 调频完成
16	RW	0x0	GPU PLL 调频请求中断，输出到 SE，写 1 清零
15:5	RO	0x0	保留
4	RW	0x1	GPU ASTC 功能使能信号
3:2	RO	0x0	保留
1	RW	0x0	Timer 计数器置位使能，高电平使能
0	RW	0x0	Timer 计数器复位使能，高电平有效

3.4.1.3 gpu_pll 寄存器(0x408)

位	读写	复位值	描述
31:7	RO	0x0	保留
6:0	RW	0x0	GPU PLL 参数。 0x0：800； 0x1：600； 0x2：400； 0x3：200

3.4.1.4 gpu_timer_set0 寄存器(0x40C)

位	读写	复位值	描述
31:0	RW	0x0	GPU Timer 计数器置位数值

3.4.1.5 gpu_timer_set1 寄存器(0x410)

位	读写	复位值	描述
31:0	RW	0x0	GPU Timer 计数器置位数值

3.4.1.6 gpu_gpio_input 寄存器(0x414)

位	读写	复位值	描述
31:17	RO	0x0	保留
16	RO	0x0	GPIO 输入响应
15:9	RO	0x0	保留
8	RW	0x0	GPIO 输入请求
7:0	RW	0x0	GPIO 输入数据

3.4.1.7 gpu_gpio_output 寄存器(0x418)

位	读写	复位值	描述
31:17	RO	0x0	保留
16	RW	0x0	GPIO 输出响应
15:9	RO	0x0	保留
8	RO	0x0	GPIO 输出请求

7:0	RO	0x0	GPIO 输出数据

3.4.2 GPU 地址映射寄存器

3.4.2.1 region0_src_addr (0x000)

位	读写	复位值	描述
31:22	RO	0x0	保留
21:0	RW	0x0	region0 映射源地址的比特[43:22]，即源地址需要 4MB 边界对齐

3.4.2.2 region0_size (0x004)

位	读写	复位值	描述
31	RW	0x0	region0 映射使能。1：使能；0：不使能
30:22	RO	0x0	保留
21:0	RW	0x0	reigon0 大小，单位为 4MB

3.4.2.3 region0_dst_addr (0x008)

位	读写	复位值	描述
31:22	RW	0x0	region0 本地映射 memory 资源大小，单位为 16MB
21:0	RW	0x0	region0 映射的目的地址的比特[43:22]，该寄存器在启动时由 SE 配置

4 VPU

4.1 简介

VPU 提供硬件视频解码功能，飞腾 X100 中 VPU 支持的特性有：

- 性能
  - 支持格式
    - HEVC YUV-NV12 / YUY2 / UYVY 8bit
    - H264 YUV-NV12 / YUY2 / UYVY 8bit
    - VC1 YUV-NV12 8bit
    - VP8 YUV-NV12 8bit
    - MPEG2 YUV-NV12 8bit
    - MPEG4 YUV-NV12 8bit
    - AVS YUV-NV12 8bit
    - VP6 YUV-NV12 8bit
    - RealVideo YUV-NV12 8bit
    - Sorenson YUV-NV12 8bit
    - HVEC Still Picture Profile
  - 支持缩小
    - 高宽最小变为原来的四分之一
    - 缩小函数的输入分辨率最小是 64×64
  - 支持旋转（仅 YUV-NV12 8bit 格式）
  - 最大分辨率
    - HEVC：8192×8192
    - 其他格式：4096×4096
    - HEVC 静态：64000×64000
    - 其他格式静态：32000×32000
  - 最小分辨率
    - VC1：80×64
    - 其他：64×64
- 低功耗

- 时钟门控

4.2 地址空间

对于 VPU 部件，地址空间包括两部分：

- 内存映射的寄存器空间：通过 PCI 的 bar0 确定，其总大小为 2MB，进一步细分如下表所示：

表 4-1 VPU 寄存器空间分配表

部件	地址空间大小	相对 BAR 基址的偏移范围
VPU slave	64KB	0x00_0000_0000 ~ 0x00_0000_ffff
地址变换	4KB	0x00_0001_0000 ~ 0x00_0001_0fff
VPU 自定义 slave	4KB	0x00_0001_1000 ~ 0x00_0001_1fff

- 独立显存空间：通过 PCI 的 bar2 确定，其大小由固件配置，如 512MB。

VPU 的独立显存空间可全部映射到系统 64 位内存空间。CPU 使用系统地址访问显存，VPU 使用显存局部地址访问显存。为使用独立显存空间，软件（驱动）需配置 VPU 的地址映射寄存器，地址映射部件负责将系统内存空间中的显存系统地址转换成显存局部物理地址。

4.2.1 地址映射配置流程

VPU 可使用主存或显存。如果要使用显存，需要为 VPU 从 DDR 显存中预分配一定大小（如 512M）专门供 VPU 使用，预分配由飞腾 X100 的固件设置；然后进行地址映射。

地址映射配置过程如下：

- 当 OS 通过 PCI 扫描到 VPU 时，会为其独立显存空间分配 PCI mem64 资源，VPU 驱动获得该 mem64 资源的地址值和大小。
- 将地址值右移 22 位，然后写入 region2_src_addr 寄存器。
- 将大小值左移 22 位，且最高位（31 位）设置为 1 后，写入 region2_size 寄存器。

4.3 操作原理

4.3.1 VPU 视频解码基础

VPU 主要用于视频解码。为在 Linux 系统使用 VPU 硬件，首先介绍 VPU

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

4.4 寄存器说明

4.4.1 VPU 地址映射寄存器

VPU 地址映射寄存器的基地址为：PCI Bar0 对应的基址+ 0x001_0000。

4.4.1.1 region2_src_addr (0x018)

位	读写	复位值	描述
31:22	RW	0x0	保留
21:0	RW	0x0	region2 映射源地址的比特[43:22]，即源地址需要 4MB 边界对齐

4.4.1.2 region2_size (0x01C)

位	读写	复位值	描述
31	RW	0x0	region2 映射使能。1：使能；0：不使能
30:22	RW	0x0	保留
21:0	RW	0x0	reigon2 大小，单位为 4MB

4.4.1.3 region2_dst_addr (0x020)

位	读写	复位值	描述
31:22	RW	0x0	region2 本地映射 memory 资源大小，单位为 16MB
21:0	RW	0x0	region2 映射的目的地址的比特[43:22]，该寄存器在启动时由 SE 配置

4.4.2 控制寄存器

VPU 控制寄存器的基地址为：PCI Bar0 对应的基址+ 0x0001_1000。

4.4.2.1 vpu_config (0x000)

位	读写	复位值	描述
31:14	RW	0x0	保留
13	RW	0x1	该值是可配置的，但是目前必须是高电平 1：关闭寄存器访问保护；0：开启寄存器保护
12	RO	0x0	1：VPU 处于空闲状态；0：VPU 未处于空闲状态
11	RW	0x1	1：开启 vp8 特性；0：关闭 vp8 特性
10	RW	0x1	1：开启 vp6 特性；0：关闭 vp6 特性
9	RW	0x1	1：开启 rv8、rv9 特性；0：关闭 rv8、rv9 特性
8	RW	0x1	1：开启 sorensen 特性；0：关闭 sorensen 特性
7	RW	0x1	1：开启 avs 特性；0：关闭 avs 特性

6	RW	0x1	1：开启 mpeg4、h263 特性；0：关闭 mpeg4、h263 特性
5	RW	0x1	1：开启 mpeg2 特性；0：关闭 mpeg2 特性
4	RW	0x1	1：开启 mpeg1 特性；0：关闭 mpeg1 特性
3	RW	0x1	1：开启 wmv9 特性；0：关闭 wmv9 特性
2	RW	0x1	1：开启 vc1 特性；0：关闭 vc1 特性
1	RW	0x1	1：开启 h264 特性；0：关闭 h264 特性
0	RW	0x1	1：开启 hevc 特性；0：关闭 hevc 特性

5 NPU

5.1 简介

NPU 是一种专门用于神经网络计算加速的部件或设备，通常采用 ASIC 设计，具有较高的算力能耗比。飞腾 X100 中的 NPU 主要用于图像领域神经网络推理加速，支持卷积、转置卷积、池化、全连、激活等 DNN 常用算子，支持图像分类、目标检测、目标跟踪等常用模型。

5.2 地址空间

对于 NPU 部件，地址空间包括两部分：

- 内存映射的寄存器空间：通过 PCI 的 bar0 确定，其总大小为 2MB，进一步细分如下表所示：

表 5-1 NPU 寄存器空间分配表

部件	地址空间大小	相对 BAR 基址的偏移范围
NPU slave	1MB	0x00_0000_0000 ~ 0x00_000f_ffff
地址变换	4KB	0x00_0010_0000 ~ 0x00_0010_0fff

- 独立显存空间：通过 PCI 的 bar2 确定，其大小由固件配置，如 512MB。

NPU 的独立显存空间可全部映射到系统 64 位内存空间。CPU 使用系统地址访问显存，NPU 使用显存局部地址访问显存。为使用独立显存空间，软件（驱动）需配置 NPU 的地址映射寄存器，地址映射部件负责将系统内存空间中的显存系统地址转换成显存局部物理地址。

5.2.1 地址映射配置流程

NPU 可使用主存或显存。如果要使用显存，需要为 NPU 从 DDR 显存中预分配一定大小（如 512MB）专门供 NPU 使用，预分配由飞腾 X100 的固件设置；然后进行地址映射。

地址映射配置过程如下：

- 当 OS 通过 PCI 扫描到 NPU 时，会为其独立显存空间分配 PCI mem64 资源，NPU 驱动获得该 mem64 资源的地址值和大小。
- 将地址值右移 22 位，然后写入 region1_src_addr 寄存器。
- 将大小值左移 22 位，且最高位（31 位）设置为 1 后，写入 region1_size

寄存器。

5.3 操作原理

神经网络计算目前还没有通用标准，飞腾 X100 NPU 的使用需要基于如下定制软件栈。总体上，神经网络模型在飞腾 X100 NPU 上推理执行需要经过两个大的步骤：

1）基于离线工具链的模型转化编译，生成在 NPU 上可部署的模型和代码文件；

2）基于运行时软件栈的模型推理执行。

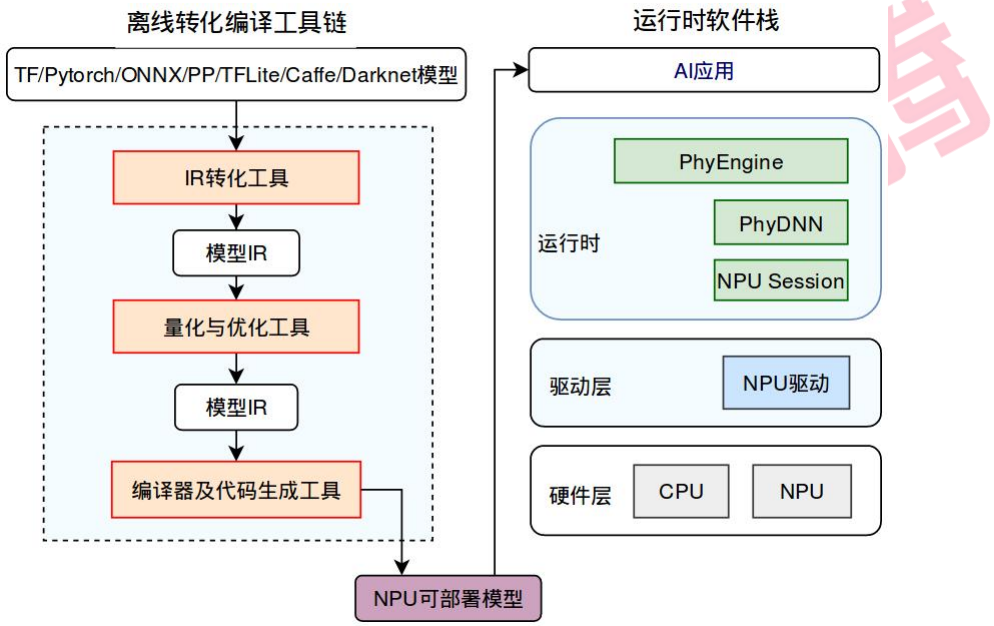

图 5-1　X100 NPU 软件栈

5.3.1　离线转化编译

离线转化编译主要完成以下工作：

(1) 模型转化

将训练好的模型转化为模型 IR。工具链支持如下框架模型的直接转化：

- TensorFlow
- TensorFlow Lite
- ONNX
- Caffe
- PyTorch
- PaddlePaddle

(2) 量化与优化

基于模型 IR 进行模型的量化与计算图优化，输出仍为模型 IR。量化基于设置的量化参数和数据样本进行。计算图优化主要分为两类：通用优化和基于硬件设备特性的优化。

(3) 编译与代码生成

模型 IR 编译为 NPU 可部署模型，可以根据具体情况通过两种途径进行：

- 直接基于编译器。这种编译途径只适合整个模型都能在 NPU 上执行的情况。
- 基于 TVM 代码生成工具。这种编译途径不仅适合整个模型都能在 NPU 上执行的情况，还适合模型中有 NPU 不能运行的 layer 的情况。

5.3.2 在线部署运行

在线部署运行的软件栈分为两大部分：

- NPU 设备驱动
- 运行时库

根据部署模型的不同编译方式，AI 应用中的模型部署运行方式略有差别，在应用层面都可以通过飞腾推理引擎接口实现推理。

基于飞腾推理引擎实现推理的过程如下：

- 创建和初始化推理引擎；
- 准备输入数据，对输入的数据按模型要求进行预处理；
- 输入数据执行推理，返回推理结果；
- 推理结果后处理。

5.4 寄存器说明

5.4.1 NPU 地址映射寄存器

NPU 地址映射寄存器的基地址为：PCI Bar0 对应的基址+0x0010_0000。

5.4.1.1 region1_src_addr (0x00C)

位	读写	复位值	描述
31:22	RW	0x0	保留
21:0	RW	0x0	region1 映射源地址的比特[43:22]，即源地址需要 4MB 边界对齐

5.4.1.2 region1_size (0x010)

位	读写	复位值	描述
31	RW	0x0	region1 映射使能。1：使能；0：不使能
30:22	RW	0x0	保留

21:0	RW	0x0	Reigon1 大小，单位为 4MB

5.4.1.3 region1_dst_addr (0x014)

位	读写	复位值	描述
31:22	RW	0x0	region1 本地映射 memory 资源大小，单位为 16MB
21:0	RW	0x0	region1 映射的目的地址的比特[43:22]，该寄存器在启动时由 SE 配置

6 DC

6.1 简介

显示控制器 DC（Display Controller）主要负责将帧缓冲（framebuffer）中的内容送到显示器输出。飞腾 X100 中 DC、DP、DC_REQ 子部件位于同一 PCI 功能号下，与另一 PCI 功能号 VPU 合称为 MMD 部件。

帧缓冲内容可由渲染部件（如 CPU、GPU、VPU 等）渲染生成，结果存放于 DDR 中。在飞腾 X100 中，GPU 渲染生成的内容可能采用压缩格式存储，因此在送给 DC 显示前，可能需经过帧缓冲解压缩部件（FBDC）解压。图中 DC_REQ 部件用于控制该解压过程。帧缓冲经 DC 输出的内容通过 DP 接口最终送到 DP 接口的显示器输出。

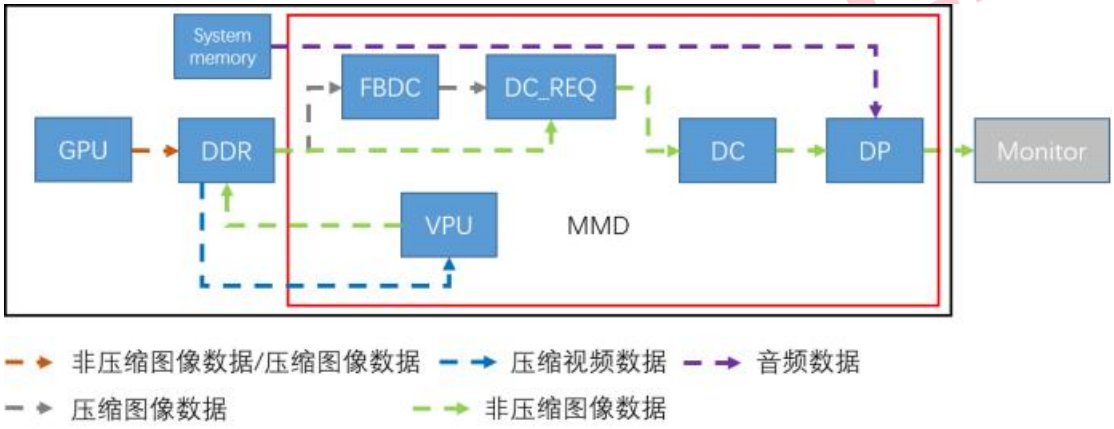

图 6-1　飞腾 X100 中 GPU、VPU 渲染数据流

飞腾 X100 的 DC 支持下列特性：

- 输入格式，支持多种 tile mode 和 color format 的组合，分两种情况，一种是使用 DC_REQ，一种是不使用 DC_REQ
  - 不使用 DCREQ 仅使用 DC，支持 linear 模式，color format 支持 X4R4G4B4、A4R4G4B4、X1R5G5B5、A1R5G5B5、R5G6B5、X8R8G8B8、A8R8G8B8、YUY2、UYVY、YV12、NV16、NV12、NV21、P10；
  - 使用 DCREQ，数据从 GPU 端过来，支持 tilemode0/tilemode3，tilemode0 支持 A1R5G5B5、A4R4G4B4、R5G6B5，tilemode3 支持

A2R10G10B10（有损、无损）、B8R8G8A8；

- 支持输出格式
  - RGB101010
  - RGB888
  - RGB666
  - RGB565
- 显示
  - 支持最大分辨率 3840x2160
  - 支持灰度系数修正（影响亮度）
  - 支持图像 dither（一种以降低分辨率为代价，来增强图像颜色的功能）
  - 支持 BT709 色域
- 图层
  - 支持 hardware cursor
  - 支持旋转（Linear 格式）
  - 支持缩小和放大（Linear 格式）
- 低功耗
  - 时钟门控

飞腾 X100 的 DP 支持特性参考《飞腾 X100 套片数据手册》。

DCREQ 模块有以下特性：

- 支持 tilemode0 和 tilemode3；
- Tilemode0 支持 color format：A1R5G5B5、A4R4G4B4、R5G6B5；
- Tilemode3 支持 color format：A2R10G10B10（有损、无损）、B8R8G8A8；

6.2 地址空间

对于 DC、DP、DC_REQ 部件，地址空间分配如下：

- 内存映射的寄存器空间：通过 PCI 的 bar0 确定，其总大小为 2M，进一步细分如下表所示（注意，由于有 3 路 DC、DP 显示，对应 3 套部件）；

表 6-1　DC 寄存器空间分配表

部件	地址空间大小	相对 BAR 基址的偏移范围
DC0 slave	8KB	0x00_0000_0000 ~ 0x00_0000_1fff
DC0_REQ	4KB	0x00_0000_2000 ~ 0x00_0000_2fff

DP0	4KB	0x00_0000_3000 ~ 0x00_0000_3fff
地址变换	4KB	0x00_0000_4000 ~ 0x00_0000_4fff
phy_dc0_st	4KB	0x00_0000_5000 ~ 0x00_0000_5fff
DC1 slave	8KB	0x00_0000_8000 ~ 0x00_0000_9fff
DC1_REQ	4KB	0x00_0000_a000 ~ 0x00_0000_afff
DP1	4KB	0x00_0000_b000 ~ 0x00_0000_bfff
地址变换	4KB	0x00_0000_c000 ~ 0x00_0000_cfff
phy_dc1_st	4KB	0x00_0000_d000 ~ 0x00_0000_dfff
DC2 slave	8KB	0x00_0001_0000 ~ 0x00_0001_1fff
DC2_REQ	4KB	0x00_0001_2000 ~ 0x00_0001_2fff
DP2	4KB	0x00_0001_3000 ~ 0x00_0001_3fff
地址变换	4KB	0x00_0001_4000 ~ 0x00_0001_4fff
phy_dc2_st	4KB	0x00_0001_5000 ~ 0x00_0001_5fff

- 独立显存空间：通过 PCI 的 bar2 确定，其大小由固件配置，如 512MB。

6.2.1 地址映射配置流程

DC 可使用显存作为帧缓冲的存储（不支持主存）。如果要使用显存，需要为 DC 从 DDR 显存中预分配一定大小（如 512M）专门供 DC 使用，预分配由飞腾 X100 的固件设置；然后进行地址映射。

地址映射配置过程如下：

- 当 OS 通过 PCI 扫描到 DC 时，会为其独立显存空间分配 PCI mem 资源，DC 驱动获得该 mem 资源的地址值和大小。
- 将地址值右移 22 位，然后写入 region3_src_addr 寄存器。
- 将大小值左移 22 位，且最高位（31 位）设置为 1 后，写入 region3_size 寄存器。

6.3 操作说明

6.3.1 整体初始化流程

- 如有必要，进行 DC、DCREQ 模块的软复位；
- 根据所选用的分辨率设置对应的像素时钟；
- DP 初始化；
- DC/DCREQ 初始化；
- 初始化完成后，如果不需要切换分辨率，则 DP 配置及 DC 的大部分配置不需要变化，只需要根据 DC、DCREQ 的图层配置相关的 double buffer

registers 机制进行切帧配置即可正常工作。

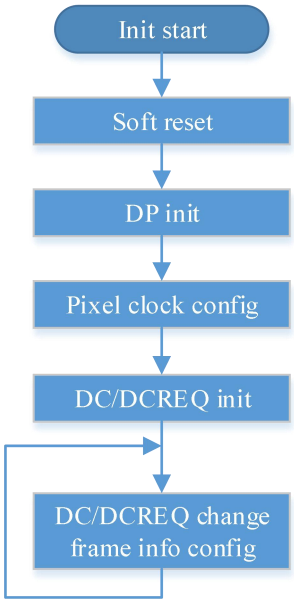

图 6-2　MMD 部件初始化流程

6.3.2　DP 初始化流程

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

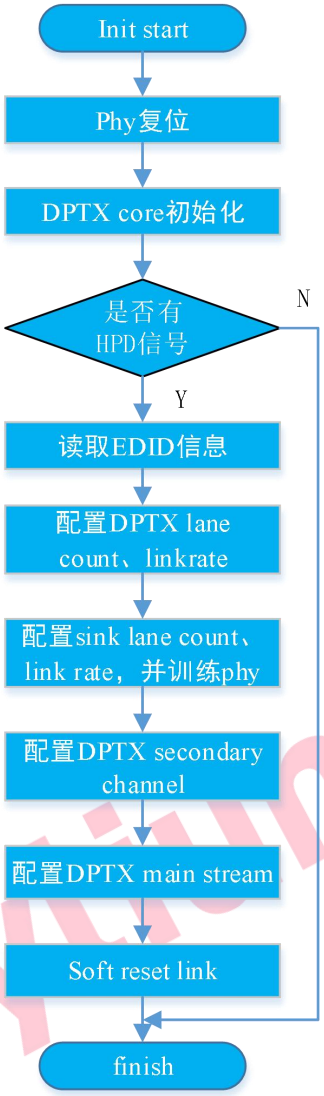

图 6-3　DP 初始化流程

6.3.3　DC/DCREQ 配置流程

- 根据当前分辨率模式及对应像素时钟，配置对应的 timing parameter；
- 输出模式配置，DP 模式，颜色格式为 RGB666/RGB888/RGB101010；
- 配置 cursor、gamma、dither 信息（如果需要）；
- 主图层 framebuffer address 和 stride 配置；
- 配置 panel；
- 配置主图层信息及其对应的 DCREQ 信息；
- 使能图像数据传输；
- 进入帧切换循环过程，帧切换依赖于 DC、DCREQ 两个图层的 double buffer 机制。

6.3.4 DP link training 流程

- 检测到 HPD 后，AUX 读 Sink DPCD 0x00000-0x0000D，读取 sink 设备信息；
- AUX 写 sink DPCD 0x00100-0x00101，配置 bandwidth 和 lane count；
- 关闭 source 端的置扰（Scrambing_disable = 0x01），选择 TPS1（TRAINING_PATTERN_SET = 0X01），source 开始发送 Training Pattern 1，AUX 写 DPCD 0x00102 选择 Training Pattern 1，并选择 disable scrambing；
- 写 0x00103-0x00106 分别配置 lane0、lane1、lane2、lane3 的 Voltage swing 和 Pre-emphasis；
- 读 0x0000E，TRAINING_AUX_RD_INTERVAL 的值；
- 等待由 TRAINING_AUX_RD_INTERVAL 确认的时间间隔后，读 0x00202-0x00207 获取 link status；
- 如果一个或以上的 lane 的 Status 读出的 CR_DONE 为 0，则时钟恢复失败，则通过下述三步处理；
  - 检查 0x00206-0x00207 的值，并调整 0x00103-0x00106 的配置；
  - 若 source 已经发送 5 次 TPS1，则降低 link bandwidth（0x00100），从第 3 步开始重新训练；
  - 若已经降低到最低带宽 1.62Gbps，则训练失败；CR_DONE 通过，则时钟恢复完成后，开始 Training Pattern 2/3/4，source 发送 Training Pattern 1，同时 AUX 写 0x00102 选择 Training Pattern 2/3/4，并选择 disable scrambing（若 source 和 sink 都支持 HBR2，则 TPS2 替换为 TPS3，若都支持 HBR3，则 TPS2 替换成 TPS4）；
- 写 0x00103-0x00106 分别配置 lane 0、lane 1、lane 2、lane 3 的 voltage swing 和 pre-emphasis；
- 等待由 TRAINING_AUX_RD_INTERVAL 确认的时间间隔后，读 0x00202-0x00207 获取 link status；
- 如果一个或以上的 lane 的 Status 读出的 CR_DONE 为 0，即时钟恢复失败，终止 TPS2，重新开始 TPS1；

- 如果CR_DONE通过，检查CHANNEL_EQ_DONE、SYMBOL_LOCKED；
- 若一个及以上的 lane 的 status 读出的 CHANNEL_EQ_DONE、SYMBOL_LOCKED 失败，则通过以下三种方法处理；
  - 检查 0x00206-0x00207 的值，并调整 0x00103-0x00106 的配置；
  - 若已经发送 5 次 TPS2，则降低 link bandwidth（0x00100），从第 3 步开始重新训练；
  - 若已经降低到最低带宽 1.62Gbps，则训练失败；
- 若 CHANNEL_EQ_DONE、SYMBOL_LOCKED 表示训练完成，则写 0x00102 关闭 link training；
- source 端寄存器 Scrambing_Disable = 0x00，TRAINING_PATTERN_SET = 0X0。

6.3.5 Post Link Training Adjust Request

- 训练过程中，如果 DPCD 读 0x002 bit 5 =1，DPCD 0x101 bit5 设置为 1；
- 训练完成，如果 DPCD 0x204 bit 1 = 1，开始 adjust 过程，ADJ_REQ_TIMER 开始计时，5-10ms；
- 读 DPCD 0x206-0x207 统计改变次数：
  - 如果 0x206-0x207 的值改变次数超过 6 次，DPCD 0x101 bit5 设为 0，结束 adjust 过程；
  - 如果改变次数未超过 6 次：
    - 如果 0x206-0x207 中预加重和电压摆幅与预设值不同，根据 0x206-0x207 的值，调整 phy 的预加重和摆幅，以及 DPCD 0x103-0x106 的值。改变次数加 1，ADJ_REQ_TIMER 清 0；
    - 如果 0x206-0x207 中预加重和电压摆幅与预设值相同，则保持 ADJ_REQ_TIMER 超过 200ms，DPCD 0x101 bit5 设为 0，结束 adjust 过程，若保持时间 ADJ_REQ_TIMER 未超过 200ms，重复 2-3 步；
- 在每次第 3 步结束后，读 0x202-0x204 的值，若全部为 0，则终止 adjust 过程，DPCD 0x101 bit5 设为 0，重新开始训练。

6.3.6 DP 视频相关配置

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

Link symbol rate = lane count*link rate*100

Vid symbol rate = (pixel clock*bpc*3)/8

Data per tu = (vid symbol rate/link symbol rate)*TU size

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

6.3.7 音频配置

6.3.7.1 DP 音频时钟同步模式

- SEC_INPUT_SELECT(0x304)设置音频输入模式选择，I2S；
- SEC_CHANNEL_COUNT(0x308)设置声道数；
- SEC_AUDIO_CHANNEL_MAP(0x35c)，设置声道映射；
- SECONDARY_DATA_WINDOW(0x08c)，设置次要数据窗口大小；
- SEC_CS_CATEGORY_CODE(0x344)，设置 8 bit category code；
- SEC_MAUD(0x318) 写入 M 值，SEC_NAUD(0x31C) 写入 N 值，SEC_AUDIO_CLOCK_MODE(0x320)设置同步模式；
- SEC_TIMESTAMP_INTERVAL(0x33C)，设置发送 ATS 的间隔时间，0 表示只在每个消隐期间发送一次；
- SEC_INFOFRAME_SELECT(0x334)选择要写入的 INFORFRAME 类型，0 对应 vender specific information(VSI)，1 对应 AUX Video Information(AVI)，2 对应 Source Product 描述，3 对应 Audio 描述，4 对应 NTSC VBI，在这些类型中 Audio 描述是必选的，其他可选；
- SEC_INFOFRAME_DATA(0x338)写入 information data，依次写入 16 或 32 字节的 payload data。Payload 从低位到高位开始写，每次写入 1 个字

节，AVI 和 AUD 的长度为 16 字节，VSI、SDP、NTSC VBI 长度为 32 字节，不足的补 0。即每种 Infoframe 必须写入 0x338 寄存器 16 或 32 次。每种类型 inforframe 的值和意义参考 CEA 861 -E；

- SEC_CS_CATEGORY_CODE(0x344)配置 Channel Status；
- SECONDARY_STREAM_ENABLE（0x088）使能音频输入 (和 0x084 一起使能)，SEC_INFOFRAME_RATE（0x314）设置各 Infoframe data 发送频率，SEC_INFOFRAME_ENABLE（0x310）使能对应的 Infoframe 类型；
- 如果只发送音频，Software_reg0(0x1C8)写 2。音视频都发送，忽略此步骤；
- 等待 500us；
- SEC_AUDIO_ENABLE（0x300）使能次要数据通道；
- 使能 I2S 数据的输入，开始音频数据传输。

6.3.7.2 音频时钟异步模式

- SEC_INPUT_SELECT（0x304，32'h0）设置音频输入选择，目前只支持 I2S(写 0)；
- SEC_CHANNEL_COUNT（0x308, 8）设置声道数；
- SEC_AUDIO_CHANNEL_MAP（0x35c, 32'h87654321）设置声道映射
- SECONDARY_DATA_WINDOW（0x08c）设置次要数据窗口大小：

Hblank = （hbp+hsw+hfp）*1000 / fpixel_clock

Case(Link rate)

1.62: hblank/24.69

2.7: hblank/14.81

5.4: hblank/7.4

8.1: hblank/4.94

Endcase

Hblank = hblank*0.9

- SEC_CS_CATEGORY_CODE（0x344）设置 8-bit category code；
- 设置时钟模式：SEC_AUDIO_CLOCK_MODE（0x320）（异步）；
- SEC_TIMESTAMP_INTERVAL（0x33c）设置发送 ATS 的时间间隔，单位为 us。设为 0 时，仅在每个场消隐期间发送一次；

- SEC_INFOFRAME_SELECT（0x334）选择要写入的 INFOFRAME 类型，0 对应 vender specific information（VSI），1 对应 AUX Video Information（AVI），2 对应 Source Product 描述（SDP），3 对应 Audio 描述（AUD），4 对应 NTSC VBI，在这些类型中，Audio 描述是发送音频时必需的，其他都是可选的；
- SEC_INFOFRAME_DATA（0x338）写入 Infoframe data， 依次写入 16 或 32 字节的 payload data。payload 从低位到高位开始写，每次写入 1 字节，AVI 和 AUD 的长度为 16 字节，VSI、SDP、NTSC VBI 的长度为 32 字节，不足的补 0，即每种 Infoframe 必须写入 0x338 寄存器 16 或 32 次。每种类型 inforframe 的值和意义参考 CEA 861 -E；
- 0x344 配置 Channel Status；
- SECONDARY_STREAM_ENABLE（0x088）使能音频输入 (和 0x084 一起使能)，SEC_INFOFRAME_RATE（0x314）设置各 Infoframe data 发送频率，SEC_INFOFRAME_ENABLE(0x310)使能对应的 Infoframe 类型；
- 如果只发送音频，Software_reg0(0x1C8)写 2，如果音视频都发送，忽略此步骤；
- SEC_AUDIO_ENABLE（0x300）使能次要数据通道；
- I2S SCLK 时钟开始输入，等待至少 1620us（异步模式，控制器在 1.62Gbps 的速率产生第一个有效的 MAUD 所需的时间，2.7Gbps 需 971us，5.4Gbps 需 485us，8.1Gbps 需 324us）；
- 使能 I2S 数据的输入，开始音频数据传输。

6.3.7.3 音频数据属性发生改变

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

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

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

6.3.8 切帧流程

DP 完成初始化以后，如果不改变分辨率及输出模式，是不需要修改配置的，切帧操作完全是基于 DC/DCREQ 本身的与图像信息相关的寄存器的 double buffered 机制实现的，这种机制保护在当前帧显示的过程中，不会被新的一个配置帧把配置改掉，导致输出异常。基本原理是，在一帧开始输出 vblank 的时候，就已经配置生效，这时候可以在寄存器中配置下一帧的帧信息（如 framebuffer address、stride 等）。

切帧流程如下：

- 完成 DC 初始化；
- 检测 FrameBufferConfig (0x1518)寄存器 bit6 FLIP_IN_PROGRES 是否为 1，若为 1，则执行下一步，若为 0，则重复检测；
- 配置 FrameBufferConfig (0x1518)寄存器 bit3 VALID 为 1；
- 配置下一帧要显示的图像的 framebuffer info、dcreq info，注意这些要配置的信息的寄存器必须是有 double buffered 功能的；
- 配置 FrameBufferConfig (0x1518)寄存器 bit3 VALID 为 0；
- 返回第二步，继续下一次切帧流程。

6.4 详细功能说明

6.4.1 行场同步信号时序参数配置

6.4.1.1 DC 部分

这部分配置是对于不同分辨率标准的行场同步信号 timing 参数配置，具体数据从各种视频标准中得来。

- 设定行显示信息：配置 HDisplay 寄存器的 total 定义行总像素个数，配置 HDisplay 寄存器的 display_end 定义行显示像素个数；
- 设定场同步信号信息：配置 VDisplay 寄存器的 total 定义场总行个数，配置 VDisplay 寄存器的 display_end 定义长显示行个数；
- 设定行同步脉冲信息：配置 HSync 寄存器，定义行脉冲同步信号起始位置、极性和是否启用；
- 设定场同步脉冲信息：配置 VSync 寄存器，定义场脉冲同步信号起始位置、极性和是否启用。

6.4.1.2 DP 部分

CEA 标准中某一分辨率包含以下相关参数：

hTotal = 800，行总像素个数（MAIN_STREAM_HTOTAL.htotal）；

hPorarity = 1（低有效），行同步信号脉冲极性（MAIN_STREAM_POLARITY.vsync_polarity）；

hsWidth = 96，行同步信号宽度（MAIN_STREAM_HSWIDTH.hs_width）；

hRes = 640，行有效信号宽度（MAIN_STREAM_HRES.hres）；

hStart = 144，行同步信号起始位置到行有效信号起始位置之间的宽度（MAIN_STREAM_HSTART. hstart）；

vTotal = 640，场总行数（MAIN_STREAM_VTOTAL.vtotal）；

vPorarity =1（低有效），场同步信号脉冲极性（MAIN_STREAM_POLARITY.hsync_polarity）；

vsWidth = 2，场同步信号脉冲宽度（MAIN_STREAM_VSWIDTH.vs_width）；

vRes = 480，场有效行数（MAIN_STREAM_VRES.vres）；

vStart = 35，场同步信号起始位置到场有效信号起始位置之间的行数。

6.4.2 光标

6.4.2.1 光标大小

不支持修改光标的大小，固定为 32 像素×32 像素大小。如果需要使用其他像素大小的光标，只能是软光标。

6.4.2.2 光标位置

Cursor 的位置由两个因素决定，一个是 hotspot（reg 1468），即 cursor 的热点，一个是 cursor pos（reg 1470），即 cursor 的位置。会表现在屏幕上的假设 hotspot 设置（0,0），则鼠标的整个位置就是 cursor pos(x,y)这个设置决定，假设 hotspot 不为零（x1,y1）, cursor 图像的起始位置会是(x-x1, y-y1), 会比实际设定的 cursor pos 偏左上，意思是将 cursor 内部的这个 hotspot 点设置到 cursor pos（x,y）位置处。

6.4.2.3 光标模式

- RGB 模式

当使用 RGB 模式的时候，cursor buffer 大小为 32×32×4 bytes，每个像素 4 个 bytes。实际 buffer 中的内容就是 cursor 最终显示出来的内容。

- MASK 模式

每个像素 2 位，buffer 大小为 32×32×2/8=256 bytes。MASK 模式下 cursor 的颜色和 background（reg 1474）和 forgeground（reg 1478）有关。硬件在读取 buffer 的时候一次读取 64 位，分两部分 bit31-bit0，bit63-bit32。

例如：ColorValueOf(x0,0) = OP(bit[0],bit[32])

ColorValueOf(x1,0) = OP(bit[1],bit[33])

……..

ColorVauleOf(x31,0) = OP(bit[31],bit[63])

OP（a,b）含义如下表所示：

表 6-2　OP（a,b）含义

low_signal(bit0-bit31)	high_signal(bit32-bit63)	color
0	0	background_color
0	1	foreground_color
1	0	nop
1	1	invert_color

6.4.2.4　光标 buffer

地址起始位置需要有像素对齐和地址偏移对齐，因为 cursor 是硬件固定大小的，所以像素不需要对齐要求，但是地址需要按照 128 字节对齐，目前软件中为了处理方便，地址全部按照 4K 对齐，包括 framebuffer 的地址。

6.4.3　gamma

6.4.3.1　gamma 校正表

gamma 校正查找表包括 3 个查找表：分别为红色、绿色和蓝色。查找表包含在只写寄存器 GammaData 中。

6.4.3.2　使能 gamma 校正表

伽马校正可通过以下方法实现：

- 如果将 gamma 校正描述为（originalColor，newColor），则可以根据颜色值顺序设置 gamma 查找表，从最小值开始，逐步设置为最大值。
- 如果要使用以下序列进行 gamma 校正：（0，0）、（1，2）、（2，5）、（3，6）…，则可以设置寄存器 GammaIndex=0，这意味着 gamma 校正从“0”开始。然后按顺序设置寄存器“GammaData”：0，2，5，6，…，连续设置 256 次以完全填充查找表。

6.4.4 dither

在颜色强度变化缓慢的图像中，像素之间的强度级别可能会有明显的跳跃。dither 可用于在相邻像素间扩散强度。在 dither 模式中，按顺序扫描像素，并且计算像素强度时的误差被分布（即扩散）到相邻像素，以使图像的整体强度更接近输入强度。

6.4.4.1 dither 表

DC 抖动实现需要一个 4 位 4x4 条目的查找表。这些 64 位表位于四个 32 位寄存器 DisplayDitherTableLow 和 DisplayDitherTableHigh 中。抖动查找表需要由显示器的两个最低有效位 x[1:0]和两个最低有效位 y[1:0]索引。

6.4.4.2 配置 dither 表

可以使用以下方法启用抖动：

- 确定要增强的像素颜色位：可以通过设置寄存器显示抖动配置 DisplayDitherConfig 来实现。
- 创建查找表：通过设置寄存器显示抖动表 DisplayDitherTableLow 和 DisplayDitherTableHigh。
  - 查找表包括 16 个条目，每个条目 4 位。
  - 查找表通过索引 X[1:0]和 Y[1:0]提供值 U[3:0]。
  - 颜色值 RedColor[3:0]用于与此 U[3:0]进行比较。
  - 如果 RedColor[3:0]>U[3:0]，并且 RedColor[7:2]不是 6'b111111，那么颜色值为：NewRedColor=RedColor[7:2]+1'b1。
  - 如果 RedColor[3:0]≤U[3:0]，则 NewRedColor=RedColor[7:2]。

6.4.5 address 和 stride

DC 读取的原始图像数据需要通过在 address 位置读取，同时根据 width、height 和 stride 计算出图形有效形状。对于 framebuffer 的 address 中的内容，是要显示的图像的原始数据，有数据格式之分。以 framebuffer 为例进行说明，硬件读取 address（reg 1400）中的内容，通过 width、height（reg 1810）和 stride（reg 1408）算出有效图像数据的大小，形成真实的原始图像数据，供硬件进行其他算法处理。Width 表示实际图像的宽，单位像素，height 表示实际图像的高，单位像素，stride 表示 buffer 中一行的字节数。这样的做法是因为 address 中存的原始图像数据需要有像素对齐限制，所以需要进行 buffer 处理，实际申请的图像数据

buffer 大小会大于等于实际图像的大小，具体每个像素包含多少个字节，会根据 framebuffer 原始数据的 format 和 tilemode 的不同而产生不同。

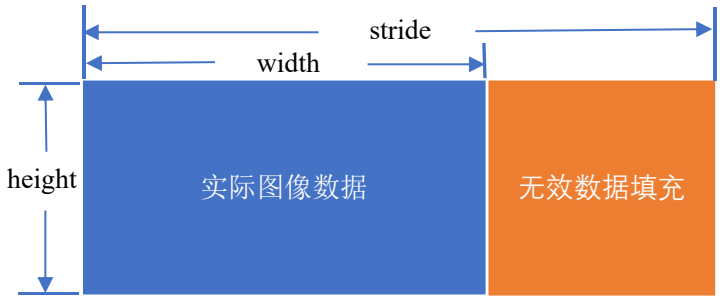

图 6-4　原始数据与有效数据之间的计算关系

当使能 dcreq 时，dcreq 的地址和 DC 的地址配置有对应关系，dcreq 的对应图层起始地址 ＋ 图源 header = DC buffer 起始地址。

header 计算方法如下所示：

- tilemode 0 时，bpc = 16，tilemode 3 时，bpc = 32；
- 计算 width×height×bpc÷4÷256÷256，如果结果不为整数，则整数部分+1，并取最终整数部分作为结果；
- header =上一步计算的结果×256。

6.4.6　Scale

Scale 分为扩大和缩小，framebuffer 可进行此操作。要求缩小后的图像的宽和高分别不能小于原始图形宽和高的三分之一。

横向和纵向的 Scale_tap 均只支持配成 3，缩放因子可以通过原始图像的大小和生成图像的大小计算出来，横向和纵向的 InitialOffset 均只能配成 0x8000。

6.4.7　数据转换

6.4.7.1　ARGB 数据转换

从源或目标读取的像素将扩展为 ARGB 格式，以保持无损像素操作。生成的像素将转换为目标格式。

6.4.7.2　YUV 转换成 RGB

YUV 数据使用 10 位通道转换为每分量 10 位 RGB 格式。一旦转换，就没有办法回到 YUV 格式。DC 支持 BT.2020 和用户可编程系数 YUV 到 RGB 颜色转换标准。硬件还可以支持 8 位的 BT.709。

6.4.7.3 Alpha Blending

一般阿尔法混合方程为：

$$Cd=Fs*Cs'+Fd*Cd' \quad (1)$$

$$Ad=Fs*As''+Fd*Ad'' \quad (2)$$

Cs'：源颜色分量

Cd'：目标颜色分量

As''：修正源α分量

Ad''：目标 alpha 分量

Fs：最终值来源的一部分

Fd：是最终值目标的一部分

Alpha Blending 分为 4 个阶段：

- 透明/不透明转换

在这个阶段中，如果需要匹配内部 alpha 规则，可以反转传入 alpha（独立于源或目标）。在内部，alpha 为 0 表示透明，而 alpha 为“0xFF”表示不透明。外部内容可能遵循相反的规则。块的输出是 As（Ad 表示目的地）或 1-As（1-Ad 表示目的地）。

- 全局值替代

寄存器中的全局 alpha 值可用于替换或缩放传入的 alpha。传入的 alpha As 可以通过，直接由 Ags（全局 alpha）代替，或者由全局 alpha 值（As*Ags）缩放。源和目标具有不同的全局 alpha 值。

- blending 因子生成

在此阶段，将生成混合因子。根据混合模式，每个 alpha 可以取值 0、1、A 或 1-A。

- 最终 blending

根据公式得到混合后的最终值。

表 6-3 混合因子说明

模式	Fs	Fd
clear	0	0
src	1	0
dst	0	1
src_over	1	1- As''

dst_over	1- Ad"	1
src_in	Ad"	0
dst_in	0	As"
src_out	1- Ad"	0

6.5 寄存器说明

6.5.1 DC 寄存器列表

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

表 6-4 DC 寄存器列表

寄存器名称	偏移	描述	double buffered
通用配置寄存器			
GeneralConfig	0x14B0	DC 通用配置	是
HiClockControl	0x0000	时钟控制	否
HIIdle	0x0004	idle 状态	否
AxiStatus	0x000C	AXI 状态	否
Frame Buffer 寄存器			
FrameBufferAddress	0x1400	Frame Buffer 起始地址低 32 位	是
FrameBufferStride	0x1408	Frame Buffer Stride	是
FrameBufferConfig	0x1518	Frame Buffer 属性配置	个别位是
FrameBufferColorKey	0x1508	Frame Buffer Colorkey 起始颜色	是
FrameBufferColorKeyHigh	0x1510	Frame Buffer Colorkey 结束颜色	是
FrameBufferScaleConfig	0x1520	Frame Buffer Scalar 配置	是
FrameBufferBGColor	0x1528	Frame Buffer 背景颜色配置	是
FrameBufferUPlanarAddress	0x1530	FrameBuffer 第二平面数据地址	是
FrameBufferVPalnarAddress	0x1548	FrameBuffer 第三平面数据地址	是
FrameBufferUStride	0x1800	FrameBuffer 第二平面数据 stride	是
FrameBufferVStride	0x1808	FrameBuffer 第三平面数据 stride	是
FrameBufferSize	0x1810	Frame Buffer 大小	是
FrameBufferScaleFactorX	0x1828	Frame Buffer X 轴缩放因子	是
FrameBufferScaleFactorY	0x1830	Frame Buffer Y 轴缩放因子	是
FrameBufferClearValue	0x1A18	Frame Buffer 清屏颜色配置	是

FrameBufferInitialOffset	0x1A20	Frame Buffer scalar 源偏移配置	是
Dither 寄存器			
DisplayDitherConfig	0x1410	dither 配置	否
DisplayDitherTableLow	0x1420	dither 表低 32 位	否
DisplayDitherTableHigh	0x1428	dither 表高 32 位	否
panel 寄存器			
PanelConfig	0x1418	Panel 参数配置	否
HDisplay	0x1430	行 visible 和 total 参数配置	否
HSync	0x1438	行同步脉冲信号配置	个别位是
VDisplay	0x1440	场 visible 和 total 参数配置	否
VSync	0x1448	场同步脉冲信号配置	个别位是
DisplayCurrentLocation	0x1450	当前显示的坐标位置	否
Gamma 校正寄存器			
GammaIndex	0x1458	gamma 表序列	否
GammaData	0x1460	gamma 表数据	否
光标 cursor 寄存器			
CursorConfig	0x1468	光标属性配置	个别位是
CursorAddress	0x146C	光标数据地址配置	是
CursorLocation	0x1470	光标位置	是
CursorBackground	0x1474	光标背景颜色	是
CursorForeground	0x1478	光标近景颜色	是
DP 配置寄存器			
DPConfig	0x1CD0	DP 配置	是
中断和门控寄存器			
DisplayIntr	0x147C	显示中断	否
DisplayIntrEnable	0x1480	显示中断使能	否
IntrAcknowledge	0x0010	中断确认	否
IntrEnbl	0x0014	中断使能	否
CursorModuleClockGatingControl	0x1484	光标的时钟门控寄存器	否
ModuleClockGatingControl	0x1A28	时钟门控寄存器，不使用功能可以关掉时钟，以节省功耗	否
滤波寄存器			
HoriFilterKernelIndex	0x1838	水平滤波索引	否
HoriFilterKernel	0x1A00	水平滤波参数	是
VertiFilterKernelIndex	0x1A08	垂直滤波索引	否
VertiFilterKernel	0x1A10	垂直滤波参数	是
版本管理寄存器			
ChipRev	0x0024	芯片版本信息	否
ChipDate	0x0028	芯片版本信息	否
HIChipPatchRev	0x0098	芯片版本信息	否
ProductId	0x00A8	芯片版本信息	否

6.5.2 DC 寄存器说明

6.5.2.1 通用配置寄存器

通用寄存器，包括一些包括时钟的配置，以及一些 AXI 总线的配置。

域	位	读写	复位值	描述
GeneralConfig				
一些通用配置				
reserved	31:4	RW	28’d0	保留
disable_idle	3	RW	1’d0	禁用 idle 信号 0：禁用；1：使能
stall_output_when_underflow	2	RW	1’d0	如果使能，FIFO 下溢时，输出将停止
endian_control	1:0	RW	2’d0	端交换控制 0：不交换 1：2 字节 2：4 字节 3：8 字节
HiClockControl				
DC 时钟控制				
reserved	31:14	RW	32’h900	保留
disable_ram_power_optimization	13	RW		禁用 ram 电源优化 0：禁用；1：使能
soft_reset	12	RW		DC 软复位 1：复位
disable_debug_register	11	RW		禁用 debug 寄存器 1：使能；0：禁用
disable_ram_clock_gating	10	RW		禁用 ram 的时钟门控
reserved	9:0	RO		保留
HIIdle				
DC idle 状态				
reserved	31:17	RO	32’h7FFFFFFF	保留
idle_dc	16	RO		DC 处于 idle 状态
reserved	15:0	RO		保留
AxiStatus				
AXI 状态寄存器				
reserved	31:10	RO	22’d0	保留
det_rd_err	9	RO	1’d0	1：检测到读错误
reserved	8	RO	1’d0	保留
rd_err_id	7:4	RO	4’d0	读错误 ID
reserved	3:0	RO	4’d0	保留

6.5.2.2 Frame Buffer 寄存器

Frame Buffer 为 DC 的底层图层，单图层情况下，指的是只是用 FrameBuffer

的图层。如果单独使用 DC，则是 linear 模式，如果配合 DCREQ 使用，可以处理 GPU 端的 tilemode0/tilemode3 的数据。

域	位	读写	复位值	描述
FrameBufferAddress				
Frame Buffer 数据起始地址的低 32 位，配合 DCREQ 中的 prefix 8 位地址，构成 Frame Buffer 的 40 位数据起始地址，地址必须 128 字节对齐。				
address	31:0	RW	32’d0	Frame Buffer 起始地址的低 32 位
FrameBufferStride				
Frame Buffer stride，单位字节，描述 Frame Buffer 原始数据每一行数据的字节数。				
reserved	31:17	RO	15’d0	保留
stride	16:0	RW	17’d0	每一行数据的字节数
FrameBufferConfig				
Frame Buffer 属性参数配置				
color format	31:26	RW	6’d0	Frame Buffer 的颜色格式。 0x00：X4R4G4B4 0x01：A4R4G4B4 0x02：X1R5G5B5 0x03：A1R5G5B5 0x04：R5G6B5 0x05：X8R8G8B8 0x06：A8R8G8B8 0x07：YUY2 0x08：UYVY 0x0F：YV12 0x11：NV12 0x12：NV16 0x15：NV12_10BIT 0x16：A2R10G10B10 0x17：NV16_10BIT 0x1B：P010
uv_swizzle	25	RW	1’d0	保留
swizzle	24:23	RW	2’d0	0：ARGB 1：RGBA 2：ABGR 3：BGRA
scale	22	RW	1’d0	scale 使能。 0：禁用；1：使能
tile_mode	21:17	RW	5’d0	tile mode 0x00：linear 0x04：tilemode0 0x07：tilemode3
yuv	16:14	RW	3’d0	yuv 标准

				1：709；3：2020
rot_angle	13:11	RW	3’d0	翻转模式 0：不翻转 1：X 轴翻转 2：Y 轴翻转 3：XY 轴翻转 5：180 度翻转
transparency	10:9	RW	2’d0	透明度 0：非透明模式 1：msk 模式 2：color key 模式
clear	8	RW	1’d0	clear 使能。当使能 clear 模式时，framebuffer 的像素值来自 FrameBufferClearValue 寄存器，禁用 clear 模式时，framebuffer 的像素值来自内存或者显存，地址由 FrameBufferAddress+prefix 配置。 0：禁用；1：使能
reserved	7	RW	1’d0	保留
filp_in_progress	6	RO	1’d0	切帧时配合 double buffered 寄存器实现对帧正确性的保护。当 frame buffer 的地址写入时，filp_in_progress 置 1。当把寄存器的值拷贝如工作寄存器生效后，filp_in_progress 将在 VBLANK 开始后清零。这个时候，可以将下一帧的相关参数配置到具有 double buffered 的寄存器中。请参考章节 2.6 中的切帧流程。 0：no；1：yes
underflow	5	RO	1’d0	当显示 FIFO 下溢时，本位置 1。读取一次该寄存器后，本位将会清零。 0：no；1：yes
reset	4	WO	1’d0	复位 DC（与软复位不同）。 1：复位。将 0 写入该位以重置显示控制器，然后配置其他寄存器，最后将 1 写入该位以启动显示控制器。当显示控制器启动时，它从 VBLANK_START 开始，所有寄存器在 VSYNC_END 跳转到工作集。计数器将重置到 HSYNC 和 VSYNC 的末尾。参考时序参数章节会详细描述。
valid	3	RW	1’d0	valid 字段定义是否可以在下一个 VBLANK 复制一组新的寄存器。这可确保如果该寄存器保存的是最近的

				一次配置，帧将始终以 valid working 开始，从而减少 SW 等待 VBLANK 信号开始的需要，以确保在下一个 VBLANK 之前加载所有状态。请参考 5.3.9 节的切帧流程。 0：working；1：pending
gamma	2	RW	1'd0	gamma 使能 0：禁用；1：使能
reserved	1	RW	1'd0	保留
output	0	RW	1'd0	当输出使能，像素将会显示。当输出禁用，显示的像素为全黑，此时 panel 的时序是正确的，但是没有任何像素。 0：禁用；1：使能
FrameBufferColorKey				
FrameBuffer Color key 范围下限				
alpha	31:24	RW	8'd0	alpha 分量
red	23:16	RW	8'd0	red 分量
green	15:8	RW	8'd0	green 分量
blue	7:0	RW	8'd0	blue 分量
FrameBufferColorKeyHigh				
FrameBuffer Color key 范围上限				
alpha	31:24	RW	8'd0	alpha 分量
red	23:16	RW	8'd0	red 分量
green	15:8	RW	8'd0	green 分量
blue	7:0	RW	8'd0	blue 分量
FrameBufferScaleConfig				
Frame Buffer scale 参数配置				
reserved	31:8	RO	24'd0	保留
horizontal_filter_tap	7:4	RW	4'd0	只支持配置成 3
filter_tap	3:0	RW	4'd0	只支持配置成 3
FrameBufferBGColor				
Frame Buffer 背景颜色，当 color key 使能，且颜色不在 color key 范围内时候，将会替换成本寄存器配置的颜色				
alpha	31:24	RW	8'd0	alpha 分量
red	23:16	RW	8'd0	red 分量
green	15:8	RW	8'd0	green 分量
blue	7:0	RW	8'd0	blue 分量
FrameBufferUPlanarAddress				
FrameBuffer 图层第二平面数据地址				
address	31:0	RW	31'd0	数据地址
FrameBufferVPlanarAddress				
FrameBuffer 图层第三平面数据地址				

address	31:0	RW	31’d0	数据地址
FrameBufferUStride				
FrameBuffer 图层第二平面 stride				
reserved	31:17	RO	15’d0	保留
stride	16:0	RW	17’d0	每一行数据的字节数
FrameBufferVStride				
FrameBuffer 图层第三平面 stride				
reserved	31:17	RO	15’d0	保留
stride	16:0	RW	17’d0	每一行数据的字节数
FrameBufferSize				
Frame Buffer 存在内存中的数据窗口大小，width × height，单位是像素				
reserved	31:30	RO	2’d0	保留
height	29:15	RW	15’d0	数据窗口高
width	14:0	RW	15’d0	数据窗口宽
FrameBufferScaleFactorX				
Frame Buffer X 轴缩放因子				
reserved	31	RW	1’d0	保留
x	30:0	RW	31’d0	x 轴缩放因子
FrameBufferScaleFactorY				
Frame Buffer Y 轴缩放因子				
reserved	31	RW	1’d0	保留
y	30:0	RW	31’d0	y 轴缩放因子
FrameBufferClearValue				
Frame Buffer 图层 clear 值，该层 clear 使能时有效				
alpha	31:24	RW	8’d0	alpha 分量
red	23:16	RW	8’d0	red 分量
green	15:8	RW	8’d0	green 分量
blue	7:0	RW	8’d0	blue 分量
FrameBufferInitialOffset				
缩放功能时，获取帧缓冲区源时使用的初始偏移量。				
y	31:16	RW	16’d0	只支持配成 0x8000
x	15:0	RW	16’d0	只支持配成 0x8000

6.5.2.3 dither 寄存器

域	位	读写	复位值	描述
DisplayDitherConfig				
Dither 功能属性配置				
enable	31	RW	1’d0	使能 dither 模式，允许 R8G8B8 模式显示在每个像素位数较少的面板上显示得更好。 1：使能；0：禁用
reserved	30:0	RO	31’d0	保留
DisplayDitherTableLow				
Dither 表 1				

Y1_X3	31:28	RW	4’d0	(x,y) =(3,1)的阈值
Y1_X2	27:24	RW	4’d0	(x,y) =(2,1)的阈值
Y1_X1	23:20	RW	4’d0	(x,y) =(1,1)的阈值
Y1_X0	19:16	RW	4’d0	(x,y) =(0,1)的阈值
Y0_X3	15:12	RW	4’d0	(x,y) =(3,0)的阈值
Y0_X2	11:8	RW	4’d0	(x,y) =(2,0)的阈值
Y0_X1	7:4	RW	4’d0	(x,y) =(1,0)的阈值
Y0_X0	3:0	RW	4’d0	(x,y) =(0,0)的阈值
DisplayDitherTableHigh				
Dither 表 2				
Y3_X3	31:28	RW	4’d0	(x,y) =(3,3)的阈值
Y3_X2	27:24	RW	4’d0	(x,y) =(2,3)的阈值
Y3_X1	23:20	RW	4’d0	(x,y) =(1,3)的阈值
Y3_X0	19:16	RW	4’d0	(x,y) =(0,3)的阈值
Y2_X3	15:12	RW	4’d0	(x,y) =(3,2)的阈值
Y2_X2	11:8	RW	4’d0	(x,y) =(2,2)的阈值
Y2_X1	7:4	RW	4’d0	(x,y) =(1,2)的阈值
Y2_X0	3:0	RW	4’d0	(x,y) =(0,2)的阈值

6.5.2.4 Panel 配置寄存器

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

域	位	读写	复位值	描述
PanelConfig				
Panel 参数配置				
reserved	31:10	RO	22’d0	保留
clock_polarity	9	RW	1’d0	时钟极性 1：Negtive；0：Positive
clock	8	RW	1’d0	时钟使能 0：禁用；1：使能
reserved	7:6	RO	2’d0	保留
data_polarity	5	RW	1’d0	数据极性 1：Negtive；0：Positive
data_enable	4	RW	1’d0	数据使能 0：禁用；1：使能
reserved	3:2	RO	2’d0	保留
de_polarity	1	RW	1’d0	数据有效时的极性 1：Negtive；0：Positive
de	0	RW	1’d0	数据有效使能 0：禁用；1：使能
HDisplay				
行显示信息配置				

reserved	31	RW	1’d0	保留
total	30:16	RW	15’d0	行总像素个数
reserved	15	RW	1’d0	保留
display_end	14:0	RW	15’d0	行显示像素个数
HSync				
行同步脉冲信息配置				
polarity	31	RW	1’d0	行脉冲同步信号极性 1：Negtive；0：Positive
pulse	30	RW	1’d0	行脉冲同步信号控制 0：禁用；1：使能
end	29:15	RW	15’d0	行脉冲同步信号结束位置（像素）
start	14:0	RW	15’d0	行脉冲同步信号起始位置（像素）
VDisplay				
场同步信号信息配置				
reserved	31	RW	1’d0	保留
total	30:16	RW	15’d0	场总行个数
reserved	15	RW	1’d0	保留
display_end	14:0	RW	15’d0	场显示行个数
VSync				
场同步脉冲信息配置				
polarity	31	RW	1’d0	场脉冲同步信号极性 1：Negtive；0：Positive
pulse	30	RW	1’d0	场脉冲同步信号控制 0：禁用；1：使能
end	29:15	RW	15’d0	场脉冲同步信号结束位置（行）
start	14:0	RW	15’d0	场脉冲同步信号起始位置（行）
DisplayCurrentLocation				
当前位置				
y	31:16	RW	16’d0	y 轴当前位置
x	15:0	RW	16’d0	x 轴当前位置

6.5.2.5 Gamma 校正寄存器

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

域	位	读写	复位值	描述
GammaIndex				
gamma 表序列				
reserved	31:8	RO	24’d0	保留
index	7:0	RW	8’d0	gamma 表序列号
GammaData				
gamma 数据转换。描述 gamma 的转换值。当这个寄存器被写入时，数据被存储在 GammaIndex 指定的索引出的 gamma 表中注册。之后如果寄存器被写入，索引就会递增。				

reserved	31:30	RW	2’d0	保留
red	29:20	RW	10’d0	red 转换值
green	19:10	RW	10’d0	green 转换值
blue	9:0	RW	10’d0	blue 转换值

6.5.2.6 cursor 寄存器

域	位	读写	复位值	描述
CursorConfig				
光标属性配置				
reserved	31:12	RO	20’d0	保留
hot_spot_x	20:16	RW	5’d0	光标热点的水平偏移
reserved	15:13	RO	3’d0	保留
hot_spot_y	12:8	RW	5’d0	光标热点的垂直偏移
reserved	7:5	RO	3’d0	保留
display	4	RW	1’d0	必须配为 0
reserved	3:1	RO	3’d0	保留
format	0	RW	1’d0	光标格式 0：DISABLED 1：MASKED 2：A8R8G8B8
CursorAddress				
光标数据地址低 32 位，配合 prefix 构成 40 位地址。				
address	31:0	RW	31’d0	光标数据地址低 32 位
CursorLocation				
光标热点的位置				
reserved	31	RO	1’d0	保留
y	30:16	RW	15’d0	光标热点垂直位置
reserved	15	RO	1’d0	保留
x	14:0	RW	15’d0	光标热点水平位置
CursorBackground				
光标背景色，MASK 模式时生效				
reserved	31:30	RO	2’d0	保留
red	29:20	RW	10’d0	red 值
green	19:10	RW	10’d0	green 值
blue	9:0	RW	10’d0	blue 值
CursorForeground				
光标前景色，MASK 模式时生效				
reserved	31:30	RO	2’d0	保留
red	29:20	RW	10’d0	red 值
green	19:10	RW	10’d0	green 值
blue	9:0	RW	10’d0	blue 值

MMD 支持 MASK 模式的光标，大小为 32×32 像素。

6.5.2.7 DP 配置寄存器

DP 配置寄存器配置 DC 模块与 DP 模块之间的数据接口格式。

域	位	读写	复位值	描述
DPConfig				
Display Port 输出配置				
reserved	31:4	RO	28'd0	保留
bus_output_sel	3	RW	1'd0	输出总线选择，必须配置成 1。 1：DP
dp_data_format	2:0	RW	3'd0	DP 接口输出格式 0：RGB565 1：RGB666 2：RGB888 3：RGB101010

6.5.2.8 中断和门控寄存器

该寄存器组描述了 MMD 中断相关的信息以及个别模块的时钟门控。

域	位	读写	复位值	描述
DisplayIntr				
DC 中断，置 1 时表示一帧输出完成，读取一次该寄存器后自动清零。				
reserved	31:1	RO	31'd0	保留
disp0	0	RO	1'd0	1：一帧图像输出完成
DisplayIntrEnable				
DC 中断使能				
reserved	31:1	RO	31'd0	保留
disp0	0	RO	1'd0	1：使能；0：禁用
IntrAcknowledge				
中断状态寄存器，每一位表示一个被触发的中断事件				
intr_vec	31:0	RO	32'd0	对于每一个事件 0：清除，1：中断有效。 bit 31： AXI_BUS_ERROR
IntrEnbl				
中断事件使能。每一位表示一个中断事件。bit31 是 AXI_BUS_ERROR 位，30:0 是写入流的平铺状态刷新完成中断				
intr_enbl_vec	31:0	RW	32'd0	
CursorModuleClockGatingControl				
光标的时钟门控				
reserved	31:1	RW	31'd0	保留
disable_module_clock_gating_cursor	0	RW	1'd0	
ModuleClockGatingControl				
各模块的时钟门控				
reserved	31:12	RW	20'd0	

disable_module_clock_gating_overlay_scalar	11	RW	1'd0	
disable_module_clock_gating_overlay	10	RW	1'd0	
reserved	9:2	RW	8'd0	
disable_module_clock_gating_overlay	1	RW	1'd0	
disable_module_clock_gating_video	0	RW	1'd0	

6.5.2.9 Framebuffer 层滤波寄存器

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

域	位	读写	复位值	描述
IndexColorTableIndex				
索引颜色表序列				
reserved	31:8	RO	24'd0	保留
index	7:0	RW	8'd0	索引颜色表序列号
IndexColorTableData				
索引颜色表数据。描述索引颜色表的转换值。当这个寄存器被写入时，数据被存储在 IndexColorTableIndex 指定的索引出的索引颜色表中注册。之后如果寄存器被写入，索引就会递增。				
alpha	31:30	RO	2'd0	
red	29:20	RW	10'd0	
green	19:10	RW	10'd0	
blue	9:0	RW	10'd0	
HoriFilterKernelIndex				
Frame Buffer 层水平滤波参数序列号				
reserved	31:8	RO	24'd0	保留
index	7:0	RW	8'd0	水平滤波参数序列号
HoriFilterKernel				
Frame Buffer 层水平滤波参数				
conefficient1	31:16	RW	16'd0	
conefficient0	15:0	RW	16'd0	
VertiFilterKernelIndex				
Frame Buffer 层水平滤波参数序列号				
reserved	31:8	RO	24'd0	保留
index	7:0	RW	8'd0	垂直滤波参数序列号
VertiFilterKernel				
Frame Buffer 层垂直滤波参数				
conefficient1	31:16	RW	16'd0	
conefficient0	15:0	RW	16'd0	

6.5.3 DP 寄存器列表

包括链路配置、链路控制、控制器性能/ID、AUX 接口、主数据流属性、次

要通道、面板自刷新和数据包直接写入几大类。

寄存器名称	偏移	描述
链路配置		
LINK_BW_SET	0x000	链路速率配置寄存器
LANE_COUNT_SET	0x004	通道数配置寄存器
ENHANCED_FRAME_EN	0x008	Enhanced Framing 模式寄存器
TRAINING_PATTERN_SET	0x00C	训练模式 Traning Pattern 寄存器
LINK_QUAL_PATTERN_SET	0x010	连接质量测试寄存器
SCRAMBLING_DISABLE	0x014	加扰使能寄存器
ALTERNATE_SCRAMBLER_RESET	0x01C	ALSR 寄存器
HBR2_COMPLIANCE_SCRAMBLER_RESET	0x020	定义在传输 HBR2 符合性链路质量模式期间传输加扰器重置模式的间隔
DISPLAYPORT_VERSION	0x024	DP 版本寄存器
PHY_POWER_STATE	0x028	电源状态寄存器
LANE_REMAP_CONTROL	0x02C	通道重映射寄存器
CUSTOM_80BIT_PATTERN_0	0x030	80-bit 自定义数据序列寄存器 0
CUSTOM_80BIT_PATTERN_1	0x034	80-bit 自定义数据序列寄存器 1
CUSTOM_80BIT_PATTERN_2	0x038	80-bit 自定义数据序列寄存器 2
链路控制		
TRANSMITTER_OUTPUT_ENABLE	0x080	主链路输出使能寄存器
MAIN_STREAM_ENABLE	0x084	视频数据使能寄存器
SECONDARY_STREAM_ENABLE	0x088	音频数据使能寄存器
SECONDARY_DATA_WINDOW	0x08C	SDP 窗口设置寄存器
SOFT_RESET	0x090	软复位寄存器
INPUT_SOURCE_ENABLE	0x094	视频输入使能寄存器
FORCE_SCRAMBLER_RESET	0x0C0	加扰器复位
USER_CONTROL_STATUS	0x0C4	时序控制信号状态（高有效）
USER_DATA_CONTROL_0	0x0C8	用户数据控制寄存器 0
USER_DATA_CONTROL_1	0x0CC	用户数据控制寄存器 1
控制器性能和核 ID		
CORE CAPABILITIES	0x0F8	控制器性能寄存器
CORE ID	0x0FC	控制器 ID 寄存器
AUX 接口		
AUX_COMMAND	0x100	AUX 请求配置寄存器
AUX_WIRTE_FIFO	0x104	AUX 写数据寄存器
AUX_ADDRESS	0x108	AUX 地址寄存器
AUX_CLOCK_DIVIDER	0x10C	AUX 时钟分频寄存器
AUX_REPLY_TIMEOUT_INTERVAL	0x110	AUX 超时时间寄存器
SINK_HPD_STATE	0x128	HPD 信号状态寄存器
INTERRUPT_STATE	0x130	中断状态寄存器

AUX_REPLY_DATA	0x134	AUX 读数据寄存器
AUX_REPLY_CODE	0x138	AUX 回复代码寄存器
AUX_REPLY_COUNT	0x13C	AUX 回复数寄存器
INTERRUPT_STATUS	0x140	中断寄存器
INTERRUPT_MASK	0x144	中断屏蔽寄存器
AUX_REPLY_DATA_COUNT	0x148	AUX 接收数据数寄存器
AUX_STATUS	0x14C	AUX 传输状态寄存器
AUX_REPLY_CLOCK_WIDTH	0x150	AUX 时钟宽度寄存器
AUX_WAKE_ACK_DETECTED	0x154	标记从连接的 SINK 设备检测到的 PHY_WAKE_ACK 信号
GP_HOST_TIMER	0x158	通用定时器
主数据流属性		
MAIN_STREAM_HTOTAL	0x180	视频行总长度寄存器
MAIN_STREAM_VTOTAL	0x184	视频场总行数寄存器
MAIN_STREAM_POLARITY	0x188	同步脉冲极性寄存器
MAIN_STREAM_HSWIDTH	0x18C	行同步脉冲宽度寄存器
MAIN_STREAM_VSWIDTH	0x190	场同步脉冲宽度寄存器
MAIN_STREAM_HRES	0x194	水平分辨率寄存器
MAIN_STREAM_VRES	0x198	垂直分辨率寄存器
MAIN_STREAM_HSTART	0x19C	Hstart 寄存器
MAIN_STREAM_VSTART	0x1A0	Vstart 寄存器
MAIN_STREAM_MISC0	0x1A4	MISC0 寄存器
MAIN_STREAM_MISC1	0x1A8	MISC1 的寄存器
MAIN_MVID	0x1AC	MVID 寄存器
TRANSFER_UNIT_CONFIG_SRC_0	0x1B0	传输单元配置寄存器
MAIN_NVID	0x1B4	NVID 寄存器
USER_PIXEL_WIDTH	0x1B8	像素输入模式寄存器
USER_DATA_COUNT	0x1BC	UDC 寄存器
MAIN_STREAM_INTERLACED	0x1C0	扫描类型寄存器
USER_SYNC_POLARITY	0x1C4	时序控制信号极性
USER_CONTROL	0x1C8	Sparse TU 模式寄存器
次要通道		
SEC_AUDIO_ENABLE	0x300	音频使能寄存器
SEC_INPUT_SELECT	0x304	音频输入选择寄存器
SEC_CHANNEL_COUNT	0x308	音频声道数寄存器
SEC_DIRECT_CLKDIV	0x30C	音频时钟分频寄存器
SEC_INFOFRAME_ENABLE	0x310	InfoFrame 类型选择寄存器
SEC_INFOFRAME_RATE	0x314	InfoFram 频率寄存器
SEC_MAUD	0x318	音频 MAUD 寄存器
SEC_NAUD	0x31C	音频 NAUD 寄存器
SEC_AUDIO_CLOCK_MODE	0x320	音频时钟模式寄存器
SEC_3D_VSC_DATA	0x324	3D 视频 VSC 寄存器
SEC_AUDIO_FIFO	0x328	音频数据输入寄存器

SEC_AUDIO_FIFO_DEPTH	0x32C	音频数据 FIFO 深度
SEC_AUDIO_FIFO_READY	0x330	
SEC_INFOFRAME_SELECT	0x334	输入 InfoFrame 类型选择寄存器
SEC_INFOFRAME_DATA	0x338	InfoFrame 数据输入寄存器
SEC_TIMESTAMP_INTERVAL	0x33C	ATS 间隔寄存器
SEC_CS_SOURCE_FORMAT	0x340	通道状态寄存器
SEC_CS_CATEGORY_CODE	0x344	Channel Status byte 1 Category code 寄存器
SEC_CS_LENGTH_ORIG_FREQ	0x348	Channel Status byte 4 寄存器
SEC_CS_FREQ_CLOCK_ACCURACY	0x34C	Channel Status byte 3 寄存器
SEC_CS_COPYRIGHT	0x350	Channel Status Copyright 寄存器
SEC_GTC_COUNT_CONFIG	0x354	GTC 配置寄存器
SEC_GTC_COMMAND_EDGE	0x358	GTC TX 寄存器
SEC_AUDIO_CHANNEL_MAP	0x35C	音频通道映射寄存器
数据包直接写入		
SEC_DB_LANE_SELECT	0x3E0	
SEC_DB_WRITE_INDEX	0x3E4	
SEC_DB_DATA_COUNT	0x3E8	
SEC_DB_DATA	0x3EC	
SEC_DB_READY	0x3F0	
SEC_DB_BUSY	0x3F4	
SEC_DB_ENABLE	0x3F8	

6.5.4 DP 寄存器说明

6.5.4.1 链路配置寄存器

域	位	读写	复位值	描述
LINK_BW_SET				
设置主链路带宽。寄存器使用与接收设备中相同名称的 DPCD 寄存器支持的值相同的值。这个值在一些 PHY 实现中使用，核心的数字部分不使用。支持 1.62、2.7、5.4、8.1Gbps。				
reserved	31:8	RO	24’d0	保留
link_bw_set	7:0	RW	8’d0	主链路带宽设置。该值乘 0.27Gb/s 即为当前设置的链接速度(link rate)。 0x06：1.62Gbps/lane 0x0A：2.7Gbps/lane 0x14：5.4Gbps/lane 0x1E：8.1Gbps/lane
LANE_COUNT_SET				
DP 使用此寄存器设置将用于配置和操作链路的通道数。在训练或视频传输期间，未使用的通道将不会激活				
reserved	31:5	RO	27’d0	保留
lane_count_set	4:0	RW	5’d0	通道数设置，支持 3 种模式 0x01:1 lane (lane 0)

				0x02:2 lanes (lane0 lane1) 0x04:4 lanes (lane0 lane1 lane2 lane3) 该值应与通过 AUX 通道写入接收端 DPCD 的 LANE_COUNT_SET 寄存器的值一致。
ENHANCED_FRAME_EN				
enhanced framing 模式使能，用于 DP1.2a 及更高版本。该值应与通过 AUX 通道写入接收端 DPCD 的 LANE_COUNT_SET 寄存器的值一致。				
reserved	31:1	RO	31'd0	保留
enhanced_framing_en	0	RO	1'd0	控制器不支持除增强帧以外的模式，因此该寄存器为只读寄存器
TRAINING_PATTERN_SET				
将该寄存器设置为非零值将输出设置为指定的训练模式。设置后，所有其他主要链接信息（如视频和音频数据）都将被阻止，以支持训练模式。				
reserved	31:3	RO	29'd0	保留
training_pattern_set	2:0	RW	3'd0	设置链路训练过程的 TPS(Traning pattern sequence) 000：不发送 TPS 001：TPS1 010：TPS2(DP1.1a)，1.62,2.7Gbps 011：TPS3(DP1.2)，5.4Gbps 100：TPS4(DP1.3+)，8.1Gbps TSP1 用于时钟恢复，TPS2、TPS3、TPS4 用于信道均等化
LINK_QUAL_PATTERN_SET				
此配置寄存器用于指示控制器启用以下测试模式之一，以便在所选通道上进行链路质量测量。这些测量是基于测试的目的，链路建立和维护时不需要。				
reserved	31:27	RO	4'd0	保留
lane_3_pattern_set	26:24	RW	3'd0	通道 3 的 pattern 设置，具体见 lane_0_pattern_set
reserved	23:19	RO	5'd0	保留
lane_2_pattern_set	18:16	RW	3'd0	通道 2 的 pattern 设置，具体见 lane_0_pattern_set
reserved	15:11	RO	5'd0	保留
lane_1_pattern_set	10:8	RW	3'd0	通道 1 的 pattern 设置，具体见 lane_0_pattern_set
reserved	7:3	RO	5'd0	保留
lane_1_pattern_set	2:0	RW	3'd0	通道 0 的 pattern 设置

				0x000 =不发送 test pattern 0x001:D10.2 test pattern (unscrambled) 0x010:Symbol Error Rate measurement pattern 011：PRBS7 100：80-bit custom pattern 101：HBR2 eye pattern 110：保留 111：保留
SCRAMBLING_DISABLE				
用于禁用 DisplayPort 发射机的内部加扰功能。该位必须在链路训练过程中设置。				
reserved	31:1	RO	31’d0	保留
scrambling_disable	0	RW	1’d0	加扰功能关闭，置 1 时，控制器不再对输出数据作加扰处理。链路训练时不做加扰处理，其他情况下应置 0
ALTERNATE_SCRAMBLER_RESET				
对于嵌入式 DisplayPort 实现，内核支持使用备用扰码器重置模式。此位只能用于嵌入式应用程序。在逐框显示端口应用程序中设置此位将导致链接失败。				
reserved	31:1	RO	31’d0	保留
alternate_scrambler_reset	0	RW	1’d0	DP 模式必须为 0。需要 Sink 设备的支持
HBR2_COMPLIANCE_SCRAMBLER_RESET				
定义在传输 HBR2 符合性链路质量模式期间传输加扰器重置模式的间隔				
reserved	31:16	RO	16’d0	保留
hbr2_compliance_scrambler_reset	15:0	RW	16’d0	设置链路质量测试时 HBR2 eye pattern 的 SR 间隔时间。该值应与 Sink 端 DPCD 的 0024A-0024B 地址的值一致。
DISPLAYPORT_VERSION				
指定核心实现支持的 DisplayPort 版本。此值用于处理辅助数据包，不控制任何 DisplayPort 功能的使用。				
reserved	31:6	RO	26’d0	保留
version_number	5:0	RW	6’d0	指定控制器支持的 DP 协议版本号 0x15: DP1.5 0x14: DP1.4a 0x13: DP1.3a 0x12: DP1.2a 0x11: DP1.1a
PHY_POWER_STATE				

控制链路上 ML_PHY_SLEEP 和 ML_PHY_STANDBY 数据模式的传输。写入该寄存器将触发模式的单次传输。对于多个模式，每次写入之间的最小间隔为 100 纳秒，可以启动对该寄存器的重复写入。				
reserved	31:2	RO	30’d0	保留
power_state	1:0	RW	2’d0	保留
LANE_REMAP_CONTROL				
用于将 DisplayPort 链接的物理通道映射到控制器的内部符号通道。使用这些寄存器位，四个物理通道中的任何一个都可以映射到任何其他内部通道进行处理。每个车道上传输的 10 位数据也可以反转。此操作发生在将通道符号数据发送到 PHY 层之前。				
reserved	31:20	RO	2’d0	保留
invert_lane_3	19	RW	1’d0	置 1 时，将输出通道 3 的数据反相
invert_lane_2	18	RW	1’d0	置 1 时，将输出通道 2 的数据反相
invert_lane_1	17	RW	1’d0	置 1 时，将输出通道 1 的数据反相
invert_lane_0	16	RW	1’d0	置 1 时，将输出通道 0 的数据反相
reserved	15:9	RO	7’d0	保留
remap_enable	8	RW	1’d0	1：使能通道重映射功能 0:控制器内部的通道与输出通道正常映射，忽略该寄存器 7:0 的值
remap_lane_3	7:6	RW	2’d0	指定控制器内部的某一个通道映射至输出通道 3 00：通道 0 映射到输出通道 3 01：通道 1 映射到输出通道 3 10：通道 2 映射到输出通道 3 11：通道 3 映射到输出通道 3
remap_lane_2	5:4	RW	2’d0	指定控制器内部的某一个通道映射输出通道 2 00：通道 0 映射到输出通道 2 01：通道 1 映射到输出通道 2 10：通道 2 映射到输出通道 2 11：通道 3 映射到输出通道 2
remap_lane_1	3:2	RW	2’d0	指定控制器内部的某一个通道映射至输出通道 1 00：通道 0 映射到输出通道 1 01：通道 1 映射到输出通道 1 10：通道 2 映射到输出通道 1 11：通道 3 映射到输出通道 1
remap_lane_0	1:0	RW	2’d0	指定控制器内部的某一个通道映射至输出通道 0

				00：通道 0 映射到输出通道 0 01：通道 1 映射到输出通道 0 10：通道 2 映射到输出通道 0 11：通道 3 映射到输出通道 0
CUSTOM_80BIT_PATTERN_0				
链路质量测试时，80bit 的自定义数据序列（custom pattern）的 bit [31:0],正在进行链路质量测试时不能改变该寄存器的值				
custom_80bit_pattern_0	31:0	RW	32’d0	80bit 的自定义数据序列（custom pattern）的 bit 31:0
CUSTOM_80BIT_PATTERN_1				
链路质量测试时，80bit 的自定义数据序列（custom pattern）的 bit [63:32],正在进行链路质量测试时不能改变该寄存器的值				
custom_80bit_pattern_1	31:0	RW	32’d0	80bit 的自定义数据序列（custom pattern）的 bit 63:32
CUSTOM_80BIT_PATTERN_2				
链路质量测试时，80bit 的自定义数据序列（custom pattern）的 bit [79:64],正在进行链路质量测试时不能改变该寄存器的值				
custom_80bit_pattern_2	31:0	RW	32’d0	80bit 的自定义数据序列（custom pattern）的 bit 79:64

6.5.4.2 链路控制寄存器

域	位	读写	复位值	描述
TRANSMITTER_OUTPUT_ENABLE				
此位用于禁用主链路成帧逻辑的所有输出。当设置为“0”时，发送器核心将只在链路上输出填充符号。禁用时不传输控制符号或有效链路数据。该位防止链路控制器核心干扰 PHY 加电序列。				
reserved	31:1	RO	31’d0	保留
enable	0	RW	1’d0	控制器主链路输出使能 0：控制器只发送无效的填充符号（Stuffing Symbol），不会发送任何控制符号（Control Symbol）和有效数据。 1：控制器主链路输出使能。 应在完成控制器和 PHY 的配置后，将寄存器置 1。用于防止控制器的输出干扰 PHY 的上电过程
MAIN_STREAM_ENABLE				
一旦链路被正确训练并且准备好开始传输用户视频数据，这些比特中的一个或多个可以基于当前虚拟源配置被写入 1。在相关视频输入端口上接收垂直同步脉冲之前，DisplayPort 发射器将输出所选源的“无视频”模式。当选择 SST 模式时应 Bit 3:1 设置为“0”。				
reserved	31:4	RO	28’d0	保留
source_3_enable	3	RW	1’d0	SST 模式应设为 0
source_2_enable	2	RW	1’d0	SST 模式应设为 0
source_1_enable	1	RW	1’d0	SST 模式应设为 0

sst_source_0_enable	0	RW	1'd0	1：在 SST 模式下，表示主链路数据有效。
SECONDARY_STREAM_ENABLE				
当主系统准备好开始传输包含音频信息的次要数据包时，该位被写入"1"。当设置为"0"时，DisplayPort 发送端的活动通道将不作为流的一部分发送次要数据。				
reserved	31:1	RO	31'd0	保留
secondary_stream_enable	0	RW	1'd0	0：禁用辅助数据，并将 VB-ID 中的 AudioMute 标志设置为"1"。 1：使能次要数据传输。
SECONDARY_DATA_WINDOW				
指定允许传输次要通道数据包的水平消隐窗口的链路符号时钟宽度。此有效数据窗口的值应小于主链接符号之间的水平消隐周期。				
reserved	31:12	RO	20'd0	保留
secondary_data_window	11:0	RW	12'd0	有效数据窗口的宽度。该值由以下公式计算： HBLANK_PERIOD /LINK_SYMBOL_CLOCK_PERIOD * 0.9 HBLANK_PERIOD: 行消隐长度 LINK_SYMBOL_CLOCK_PERIOD：链路时钟周期长度
SOFT_RESET				
执行特定控制器管理功能的软重置。此复位仅适用于控制部分。可编程寄存器组的状态不受软复位的影响。				
reserved	31:2	RO	30'd0	保留
video_soft_reset	1	WO	1'd0	1: 复位视频时钟域 vid_clk 部分模块的功能，异步复位同步释放，复位信号保持 8 个 vid_clk 周期有效
link_soft_reset	0	WO	1'd0	1: 复位链路时钟域 link_clk 部分模块的功能，异步复位同步释放，复位信号保持 8 个 lnk_clk 周期有效
INPUT_SOURCE_ENABLE				
允许从一个或多个虚拟源传输视频和辅助通道音频数据。在 SST 模式下，只有该寄存器的 bit 0 有效。如果未启用视频流，启用虚拟源输入将开始传输 NO_VIDEO 模式。先使能 0x094 寄存器，开始发送 no video pattern，等待发送 5 次 no video pattern 后，再使能 0x084 寄存器。				
reserved	31:4	RO	28'd0	保留
virtual_source_3_enable	3	RW	1'd0	使能 Source3 输入，写 1 后，将开始发送 No Video Pattern
virtual_source_2_enable	2	RW	1'd0	使能 Source2 输入，写 1 后，将开始发送 No Video Pattern
virtual_source_1_enable	1	RW	1'd0	使能 Source1 输入，写 1 后，将开

				始发送 No Video Pattern
virtual_source_0_enable	0	RW	1'd0	使能 Source0 输入，写 1 后，将开始发送 No Video Pattern
FORCE_SCRAMBLER_RESET				
用于 debug。设为 1 时，控制器将会强制将下一次发送的 Blanking Start 符号替换为 Scrambler Reset。（正常情况下，每 512 个 BS 被替换为 SR）。 建议配合 0x84 一起使用，使能 0x84 后，复位一次 SR，以确保开始发送视频数据时，Source 和 Sink 两端的 LFSR 同步				
reserved	31:1	RO	31'd0	保留
force_scrambler_reset	0	RO	1'd0	1: 控制器将会强制将下一次发送的 Blanking Start 符号替换为 Scrambler Reset。
USER_CONTROL_STATUS				
提供来自用户数据接口的极性校正控制信号的直接副本。此寄存器可用于触发主机系统中的特定事件。				
reserved	31:4	RO	28'd0	保留
user_control_oddeven	3	RO	1'd0	当前视频控制信号 vid_oddeven 输入经过极性纠正后的状态
user_control_den	2	RO	1'd0	当前视频控制信号 vid_enable 输入经过极性纠正后的状态
user_control_hsync	1	RO	1'd0	当前视频控制信号 vid_hsync 输入经过极性纠正后的状态
user_control_vsync	0	RO	1'd0	当前视频控制信号 vid_vsync 输入经过极性纠正后的状态
USER_DATA_CONTROL_0				
Source0,Source1 累计延迟控制寄存器 控制内部视频数据 data path， 默认值 0x20042004 不可随意更改。在链路利用率过低或过高时需要手动配置。				
reserved	31	RO	16'h2004	保留
source1_data_accumulation_delay	30:26	RW		Source1：从有效数据准备好到开始传输第一个 TU 之间的延迟
reserved	25:22	RO		保留
source1_fifo_accumulation_depth	21:16	RW		Source1 :user FIFO 需要累计足够的数据，才能开始读并发送到 link 上，该值决定 FIFO 中累积数据的个数，FIFO 中的数据数量达到该值后 fifo_data_ready 置 1
reserved	15	RO	16'h2004	保留
source0_data_accumulation_delay	14:10	RW		Source0：从有效数据准备好到开始传输第一个 TU 之间的延迟
reserved	19:6	RO		保留
source0_fifo_accumulation_depth	5:0	RW		Source0 :user FIFO 需要累计足够的数据，才能开始读并发送到 link

				上，该值决定 FIFO 中累积数据的个数，FIFO 中的数据数量达到该值后 fifo_data_ready 置 1
USER_DATA_CONTROL_1				
Source2,Source3 累计延迟控制寄存器 控制内部视频数据 data path， 默认值 0x20042004 不可随意更改。在链路利用率过低或过高时需要手动配置。				
reserved	31	RO	16’h2004	保留
source3_data_accumulation_delay	30:26	RW		Source3：从有效数据准备好到开始传输第一个 TU 之间的延迟
reserved	25:22	RO		保留
source3_fifo_accumulation_depth	21:16	RW		Source3 :user FIFO 需要累计足够的数据，才能开始读并发送到 link 上，该值决定 FIFO 中累积数据的个数，FIFO 中的数据数量达到该值后 fifo_data_ready 置 1
reserved	15	RO	16’h2004	保留
source2_data_accumulation_delay	14:10	RW		Source2：从有效数据准备好到开始传输第一个 TU 之间的延迟
reserved	19:6	RO		保留
source2_fifo_accumulation_depth	5:0	RW		Source2 :user FIFO 需要累计足够的数据，才能开始读并发送到 link 上，该值决定 FIFO 中累积数据的个数，FIFO 中的数据数量达到该值后 fifo_data_ready 置 1

6.5.4.3 控制器性能/ID 寄存器

域	位	读写	复位值	描述
CORE_CAPABILITIES				
确定控制器中可用的功能。软件可以使用这个寄存器来确定控制器配置。				
reserved	31:19	RO	13’d0	保留
reserved	18:16	RO	3’d4	保留
reserved	15:12	RO	5’d0	保留
reserved	11	RO	1’b1	保留
embedded_present	10	RO	1’b1	保留
secondary_present	9	RO	1’b1	1：当前支持音频传输
reserved	8	RO	1’b1	保留
reserved	7:3	RO	5’d0	保留
lane_count	2:0	RO	3’d4	当前支持最大通道数
CORE_ID				
控制器版本，供软件使用				
core_id	31:16	RO	16’hA	控制器识别 ID，为固定值 0x000A。用于软件配置
core_rev_level	15:0	RO	16’h508	控制器版本号，固定值 0x0508。用于

				软件配置

6.5.4.4 AUX 接口寄存器

域	位	读写	复位值	描述
AUX_COMMAND				
写入时启动指定长度的 AUX 通道命令。该寄存器作为 AUX 通道请求设置过程的一部分最后写入。写入时，内部状态机将开始向接收器设备发送请求。在写入该寄存器之前，必须设置 AUX_ADDRESS 和 AUX_WRITE_FIFO（如适用）。				
reserved	31:14	RO	18'd0	保留
aux_phy_wake	13	RW	1'b0	保留
address_only	12	RW	1'b1	写 1 时控制器将发送仅包含地址（无数据）的 AUX 请求
command	11:8	RW	4'd0	AUX 命令类型 0x8：AUX Write 0x9：AUX Read 0x0：I2C over AUX Write 0x4：I2C over AUX Write, Middle of Transaction bit set（MOT） 0x1：I2C over AUX Read 0x5：I2C over AUX Read, Middle of Transaction bit set（MOT） 0x2：I2C over AUX Write Status
reserved	7:4	RO	4'd0	保留
byte_count	3:0	RW	4'd0	当前 AUX 命令要发送的数据字节数，寄存器的值 0-15 对应 1-16 个字节数据
AUX_WIRTE_FIFO				
在 AUX 通道上启动本机或 I2C 写入请求之前，主机系统必须向映射到此地址的 FIFO 提供写入数据。只有支持当前事务所需的字节数必须写入 FIFO。在清除 REQUEST_IN_PROGRESS 位之前，不得执行对 FIFO 的后续写入。				
reserved	31:8	RO	24'd0	保留
aux_channel_data	7:0	WO	8'd0	存入 AUX_WRITE_FIFO 的数据，主机在发起 AUX 或者 I2C 写请求前，将要发送的数据写入寄存器，写入数据个数不能超过 AUX_COMMAND 中设定的 BYTE_COUNT
AUX_ADDRESS				
对于 AUX 请求，每个 AUX 请求需要一个 20 位地址；对于 I2 转 AUX 请求，每个 AUX 请求需要一个 8 位地址。此寄存器指定当前 AUX 通道的地址命令。这些位用于请求的地址字段而不作修改。				
reserved	31:20	RO	12'd0	保留
aux_address	19:0	WR	20'd0	20-bit 的 AUX 通道命令的起始地址
AUX_CLOCK_DIVIDER				
AUX 通道的时钟频率为固定的 1Mhz，该时钟由 APB 时钟分频产生。寄存器的值为 APB 时钟分频值（只支持整数分频），有效范围 10-400。例如若 APB 时钟为 75Mhz，该值将设				

为 75。				
reserved	31:9	RO	23’d0	保留
devide_value	8:0	RW	9’d0	分频值
AUX_REPLY_TIMEOUT_INTERVAL				
控制器在发送 AUX requeset 后，等待 AUX reply 的时间，若超过这个时间还未收到 relpy，控制器向主机发送超时中断。				
reserved	31:9	RO	23’d0	保留
reply_value	8:0	RW	9’d400	超时时间，单位 ms
SINK_HPD_STATE				
控制器 HPD 输入端口的 raw state				
reserved	31:1	RO	31’d0	保留
hpd_raw_state	0	RO	1’b0	
INTERRUPT_STATE				
保存中断管理逻辑内部状态的状态位。这些信号用于生成中断，并且可以用作不实现中断的系统的轮询状态。				
reserved	31:7	RO	25’d0	保留
reply_error	6	RO	1’b0	AUX reply error
reserved	5	RO	1’b0	保留
gp_timer_event	4	RO	1’b0	通用定时器中断
reply_timeout	3	RO	1’b0	等待 AUX reply 超时
reply_recieved	2	RO	1’b0	接收到 AUX reply
hpd_irq	1	RO	1’b0	hpd irq 中断
hpd_event	0	RO	1’b0	HPD 连接或断开事件 中断
AUX_REPLY_DATA				
此只读地址映射到内部 FIFO，该 FIFO 包含在 AUX 通道应答期间接收的多达 16 字节的信息。从字节 0 开始从 FIFO 读取应答数据。FIFO 中的有效字节数与应答数据计数寄存器指示的接收字节数相对应。				
reserved	31:8	RO	24’d0	保留
aux_reply_data	7:0	RO	8’d0	AUX reply 期间接收到的数据，每次读取，内部 FIFO 读指针加一。有效数据个数与 REPLY_DATA_COUNT 寄存器中的设置有关
AUX_REPLY_CODE				
最近一次发起的 AUX request 时接收到的 reply code。反映 AUX 传输的状态				
reserved	31:4	RO	28’d0	保留
aux_reply_code	3:0	RO	4’d0	0x0:Native AUX ACK 0x1:Native AUX NACK 0x2:Native AUX Defer 0x0:I2C over AUX ACK 0x4:I2C over AUX NACK 0x8:I2C over AUX Defer
AUX_REPLY_COUNT				
统计目前收到的 AUX reply 个数（不包括 reply code）。写 1 清零				

reserved	31:8	RO	24’d0	保留
aux_reply_count	7:0	RO	8’d0	收到的 AUX reply 个数。
INTERRUPT_STATUS				
控制器中断状态寄存器，包括中断原因。可导致中断的特定事件和相关的状态位如下所示。从该寄存器读取的数据将清除所有值。				
reserved	31:7	RO	25’d0	保留
reply_error	6	RW	1’b0	AUX reply error
reserved	5	RW	1’b0	保留
gp_timer_irq	4	RW	1’b0	通用定时器发起中断
reply_timeout	3	RW	1’b0	因等待 AUX reply 超时发起中断
reply_recieved	2	RW	1’b0	接收到 AUX reply 发起中断
hdp_irq	1	RW	1’b0	hpd irq 中断
hpd_event	0	RW	1’b0	HPD 连接或断开事件 中断
INTERRUPT_MASK				
控制器的每个中断源可以单独屏蔽。当此寄存器中的相应位设置为“1”时，不会为事件生成中断。上电后，所有中断源都被屏蔽。				
reserved	31:7	RO	25’d0	保留
reply_error_mask	6	RW	1’b1	AUX reply error
reserved	5	RW	1’b1	保留
gp_timer_irq_mask	4	RW	1’b1	通用定时器发起中断
reply_timeout_mask	3	RW	1’b1	因等待 AUX reply 超时发起中断
reply_recieved_mask	2	RW	1’b1	接收到 AUX reply 发起中断
hdp_irq_mask	1	RW	1’b1	hpd irq 中断
hpd_event_mask	0	RW	1’b1	HPD 连接或断开事件 中断
AUX_REPLY_DATA_COUNT				
最近一次 AUX reply 传输从 sink 端接收到的数据个数。控制器发起 AUX request 会清空该寄存器				
reserved	31:5	RO	27’d0	保留
data_count	4:0	RO	5’d0	从 AUX 通道收到的响应数据数量。
AUX_STATUS				
此寄存器包含内部 AUX 通道控制器的状态。监视请求和应答事务的进度，并检查应答事务是否有错误。这些位总是有效的。				
reserved	31:4	RO	28’d0	保留
reply_error	3	RO	1’b0	1：最近一次的 AUX relpy 传输过程出现错误。Relpy_error 指 reply 过程中，等待 framing start code 超时
request_in_progress	2	RO	1’b0	1：控制器正在发送 AUX requeset；0：AUX request 状态机空闲
reply_in_progress	1	RO	1’b0	1：控制器正在接受 AUX reply
reply_received	0	RO	1’b0	0：控制器正在发送 AUX request，发起 request 将此位清零 1： 控制器已收到完整有效的 AUX reply

AUX_REPLY_CLOCK_WIDTH				
AUX reply 通过同步过程恢复的 AUX reply 时钟宽度（APB 时钟的个数）。该寄存器仅在一个 AUX reply 传输完成后有效				
reserved	31:10	RO	22’d0	保留
aux_reply_clock_width	9:0	RO	10’d0	AUX reply 时钟宽度
AUX_WAKE_ACK_DETECTED				
标记从连接的接收设备检测 AUX 物理唤醒信号。该位在任何 AUX 事务开始时清除，并在检测到 AUX 物理唤醒信号时设置。				
reserved	31:1	RO	31’d0	保留
aux_phy_wake_ack	0	RO	1’b0	1：AUX reply 接收过程中接收到 PHY_WAKE_ACK, 表示 AUX PHY WAKE 请求完成
GP_HOST_TIMER				
控制器使用的通用 20 位定时器。DisplayPort Tx core 不将此计时器用于任何内部功能。计时器的分辨率为 1 usec。				
enable	31	RW	1’b0	写 1 使能通用定时器
reload	30	RW	1’b0	写 1 定时器计到 0 时自动重置，写 0 定时器只运行一次
interrupt	29	RW	1’b0	写 1 定时器计到 0 时产生中断
reserved	28:20	RO	9’d0	保留
timer_value	19:0	RW	20’d0	写操作设置定时器的计数值，读操作返回定时器当前的值

6.5.4.5 主数据流属性寄存器

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

域	位	读写	复位值	描述
MAIN_STREAM_HTOTAL				
指定主流视频信号在水平帧周期内的时钟总数。此值作为主流属性 Htotal 发送。				
reserved	31:16	RO	16’d0	保留
Htotal	15:0	RW	16’d0	Htotal
MAIN_STREAM_VTOTAL				
提供主流视频帧中垂直同步脉冲之间的总行数。此值作为主流属性 Vtotal 提供。				
reserved	31:16	RO	16’d0	保留
vtotal	15:0	RW	16’d0	Vtotal
MAIN_STREAM_POLARITY				
行场同步信号脉冲极性				
reserved	31:2	RO	30’d0	保留
vsync_polarity	1	RO	1’b0	场同步信号极性 0：高有效；1：低有效
hsync_polarity	0	RO	1’b0	行同步型号极性 0：高有效；1：低有效

MAIN_STREAM_HSWIDTH				
行同步信号脉宽，单位：像素时钟周期				
reserved	31:16	RO	16’d0	保留
hs_width	15:0	RW	16’d0	行同步信号脉宽
MAIN_STREAM_VSWIDTH				
场同步信号脉宽，单位：像素时钟周期				
reserved	31:16	RO	16’d0	保留
vs_width	15:0	RW	16’d0	场同步信号脉宽
MAIN_STREAM_HRES				
视频流一行的有效像素个数				
reserved	31:16	RO	16’d0	保留
hres	15:0	RW	16’d0	视频流一行的有效像素个数
MAIN_STREAM_VRES				
视频流一场的有效行数				
reserved	31:16	RO	16’d0	保留
vres	15:0	RW	16’d0	视频流一场的有效行数
MAIN_STREAM_HSTART				
视频流每行的有效开始像素位置				
reserved	31:16	RO	16’d0	保留
hstart	15:0	RW	16’d0	视频流每行的有效开始像素位置
MAIN_STREAM_VSTART				
视频流每场的有效开始行位置				
reserved	31:16	RO	16’d0	保留
vstart	15:0	RW	16’d0	视频流每场的有效开始行位置
MAIN_STREAM_MISC0				
此 8 位值包含有关视频流时钟和颜色的信息陈述。这些位从 DisplayPort 规范 MISC0 寄存器定义映射。				
reserved	31:8	RO	24’d0	保留
bit_depth	7:5	RW	3’b000	每个颜色的位数。 000：6 位 001：8 位 010：10 位 011：12 位 100：16 位 101：保留 110：保留 111：保留
ycbcr_colorimetry	4	RW	1’b0	主链路视频色度。 0：ITU-R BT601-5；1：ITU-R BT709-5
dynamic_range	3	RW	1’b0	颜色范围。 0：VESA range；1：CEA range
component_format	2:1	RW	2’b00	颜色格式 00：RGB

				01：YCbCr 4:2:2 10：YCbCr 4:4:4 11：保留
synchronous_clock	0	RW	0'b0	locking mode for the user data 0：asynchronous clock；1：synchronous clock
MAIN_STREAM_MISC1				
这些位表示 DisplayPort 规范中定义的主数据流属性字段 MISC1。这些比特提供隔行扫描和立体视频信息。				
reserved	31:8	RO	24'd0	保留
y_only	7	RW	1'b0	1: Y-only Color format
zero	6:3	RW	4'd0	必须配置为 0
stereo_video_attr	2:1	RW	2'b00	00:立体视频 01:右眼 10:保留 11:左眼
interlaced_total_even	0	RW	1'b0	0 ：每个隔行扫描帧的行数是奇数 1 ：每个隔行扫描帧的行数是偶数
MAIN_MVID				
M 值，异步时钟模式下生效				
reserved	31:24	RO	8'd0	保留
m_vid	23:0	RW	24'd0	像素时钟(Mhz)×100
TRANSFER_UNIT_CONFIG_SRC_0				
传输单元是一个 DisplayPort 包，表示有效的数据符号和填充符号。该寄存器值设置发射机帧逻辑中传输单元的大小。该寄存器只支持 4 的倍数值。当配置为稀疏 TU 模式时，只有该寄存器的 TRANSFER_UNIT_SIZE 字段有效。忽略所有其他字段。				
reserved	31:27	RO	4'd0	保留
frac_symbols_per_tu	27:24	RW	4'd0	一个 TU 中 valid symbol 的个数的小数部分（单位为 1/16th）
symbols_per_tu	23:16	RW	8'd0	一个 TU 中 valid symbol 的个数的整数部分，可被设置为 64 及以下的整数值，关于 valid symbol 个数的计算见 DP 标准的 2.2.1.4.1
reserved	15:7	RW	9'd0	保留
transfer_unit_size	6:0	RW	7'd0	TU 的大小（valid symbol 和 stuff symbol 的总数），必须被设置在 32 和 64 之间
MAIN_NVID				
N 值。基于链路速率设置用于主流属性的第二时钟值。当与 M_VID 值一起使用时，该值允许接收器设备恢复用户数据像素时钟的频率。异步时钟模式下生效。				
reserved	31:24	RW	8'd0	保留
n_vid	23:0	RW	24'd0	1.62Gbps 时，配成 16200； 2.7Gbps 时，配成 27000； 5.4Gbps 时，配成 54000；

				8.1Gbps 时，配成 81000；
USER_PIXEL_WIDTH				
控制器的用户数据接口接受每个时钟周期一个、两个或四个像素。此寄存器选择用户数据输入端口的宽度，应在启用主链路视频之前进行设置。复位时，该寄存器默认为 1。				
reserved	31:3	RW	29'd0	保留
user_pixel_count	2:0	RW	3'd1	1、2、4 分别对应每个视频时钟周期输入 1、2、4 个像素数据
USER_DATA_COUNT				
在发送消隐开始符号之前，确定要从用户 FIFO 读取的发送器帧逻辑的总数据计数。换句话说，这个值是一行活动数据中 symbol 的总数。计算值应向上舍入。				
reserved	31:18	RW	24'd0	保留
user_data_count	17:0	RW	18'd0	SYMBOL_COUNT = ((HRES * 位 per pixel) + 7) / 8 UDC = (SYMBOL_COUNT + lane_count – 1) / lane_count
MAIN_STREAM_INTERLACED				
视频扫描类型				
reserved	31:1	RW	31'd0	保留
main_stream_interlaced	0	RW	1'b0	1：隔行扫描；0：逐行扫描 与 VBID 和 MSA 有关
USER_SYNC_POLARITY				
指示视频源同步信号的极性。				
reserved	31:4	RW	28'd0	保留
user_oddeven_polarity	3	RW	1'b0	odd/even 信号的极性 1：高有效；0：低有效
user_data_enable_polarity	2	RW	1'b0	user data enable 信号的极性 1：高有效；0：低有效
user_vsync_polarity	1	RW	1'b0	vsync 信号极性 1：高有效；0：低有效
user_hsync_polarity	0	RW	1'b0	hsync 信号极性 1：高有效；0：低有效
USER_CONTROL				
控制控制器的每个活动通道中用户数据 FIFO 的行为。				
reserved	31:2	RW	30'd0	保留
user_secondary_immediate	1	RW	1'b0	设置为 1 时，若此时 0x088 寄存器已被设为 1，Secondary channel 立即变为有效，不需要等待下一个场同步信号
user_sparse_mode_enable	0	RW	1'b0	设置为 1 时，TU 中 valid symbol 的数量可变，不再是固定值，TRANSFER_UNIT_CONFIG 寄存器中 SYMBOL_PER_TU 和 FRAC_SYMBOLS_PER_TU 的设置将被忽略

6.5.4.6 次要通道寄存器

次要通道寄存器主要描述音频通道的属性及工作方式。

域	位	读写	复位值	描述
SEC_AUDIO_ENABLE				
次要通道使能				
reserved	31:2	RW	30'd0	保留
sec_audio_mute	1	RW	1'b0	设置为 1 时，VBID 的 audio mute（静音）位被设为 1。SEC_AUDIO_MUTE 与 SEC_AUDIO_ENABLE 不能设为相同值
sec_audio_enable	0	RW	1'b0	1：音频使能；0：音频禁用
SEC_INPUT_SELECT				
选择音频输入来源，目前只支持 I2S 和 Direct Sample FIFO 两种模式				
reserved	31:2	RW	30'd0	保留
audio_source_select	1:0	RW	2'b00	00： I2S 01：Direct Sample FIFO 10：保留 11：保留
SEC_CHANNEL_COUNT				
音频通道数量选择，最多支持 8 声道				
reserved	31:3	RW	29'd0	保留
audio_channel_num	2:0	RW	3'b000	000： Audio mute set in the VBID field. 010：Two channel audio 011：Two channel audio with LFE 110：5.1 channel surround sound 111：7.1 channel surround sound
SEC_DIRECT_CLKDIV				
用于音频源为 Direct write FIFO mode，APB 时钟分频至音频采样时钟的分频值。MMD 中 APB 时钟为 48MHz，分频后的时钟单位为 KHz				
reserved	31:16	RW	16'd0	保留
clk_div	15:0	RW	16'd0	clk_div = (APB Frequency / Audio Sample Rate)
SEC_INFOFRAME_ENABLE				
此寄存器中的每一位将允许传输五个受支持的通用辅助数据包中的一个				
reserved	31:5	RW	16'd0	保留
NTSC_VBI_InfoFrame_Product	4	RW	1'b0	NTSC VBI InfoFrame
Audio_InfoFrame_Source	3	RW	1'b0	Audio InfoFrame
Description _InfoFrame	2	RW	1'b0	Source Product Description InfoFrame
AUX_Video_Information	1	RW	1'b0	AUX Video Information (AVI)

_(AVI) InfoFrame				InfoFrame
vendor_specific_ infoFrame	0	RW	1'b0	Vendor Specific InfoFrame
SEC_INFOFRAME_RATE				
选择每个通用辅助数据分组被发送到接收器设备的频率。包可以以立即模式发送一次，或者以帧模式每帧发送一次。将对应于通用 SDP 的位设置为 1，以启用每帧的连续传输。设置为 0 以在启用位从 0 转换为 1 时传输通用 SDP 一次。连续传输被定义为在垂直消隐间隔期间每帧发送一个包。				
reserved	31:5	RW	16'd0	保留
NTSC_VBI_InfoFrame_Product	4	RW	1'b0	NTSC VBI InfoFrame
Audio_InfoFrame_Source	3	RW	1'b0	Audio InfoFrame
Description _InfoFrame	2	RW	1'b0	Source Product Description InfoFrame
AUX_Video_Information _ InfoFrame	1	RW	1'b0	AUX Video Information (AVI) InfoFrame
vendor_specific_ infoFrame	0	RW	1'b0	Vendor Specific InfoFrame
SEC_MAUD				
音频数据 M 值，音频时钟与链路时钟同步时有效				
reserved	31:24	RW	8'd0	保留
m_aud	23:0	RW	24'd0	M 值
SEC_NAUD				
音频数据 N 值，音频时钟与链路时钟同步时有效				
reserved	31:24	RW	8'd0	保留
n_aud	23:0	RW	24'd0	N 值
SEC_AUDIO_CLOCK_MODE				
音频数据时钟模式，指音频数据的 sample clock 和传输时钟 Link clock				
reserved	31:1	RW	31'd0	保留
sec_audio_clock_mode	0	RW	0	0：异步时钟模式 1：同步时钟模式
SEC_3D_VSC_DATA				
用于在 3D 立体应用中承载附加视频流配置信息的数据字节。DisplayPort 规范的第 2.2.5.6.2 节规定了该字节的内容。				
reserved	31:8	RW	24'd0	保留
simsp	7:4	RW	4'd0	Stereo Interface Method-Specific Parameter
simc	3:0	RW	4'd0	Stereo Interface Method Code
SEC_AUDIO_FIFO				
音频数据输入为 Direct write 时的 FIFO				
sample_fifo_entry	31:0	RW	32'd0	用于音频数据输入为 Direct write FIFO mode 时。写入顺序为 Channel 1，Sample0 最先写入，随后通道数先增加至支持的通道数后，Sample 数再增加，以 2 声道为例：顺序依次为 C1 S0，C2

				S0，C1 S1，C2 S1，C1 S2，C2 S2。每 8 个 sample 为一个 secondary audio channel packet 每个 sample 高 8 位为 control field，低 24 位为音频数据 每次写入后，内部 sample FIFO 指针加一，读操作将返回上一次写入寄存器的值
SEC_AUDIO_FIFO_DEPTH				
包含要存储在音频采样 FIFO 中的最大采样数。				
override	31	RW	1’b0	direct write FIFO override
reserved	30:10	RW	21’d0	保留
sec_audio_fifo_depth	9:0	RW	10’d0	用于音频数据输入为 Direct write FIFO mode 时。内部 sample FIFO 的深度，写入 FIFO 的个数达到设定的深度时，FIFO 的 ready flag 无效
SEC_AUDIO_FIFO_READY				
direct audio sample FIFO 中数据个数少于 SEC_AUDIO_FIFO_DEPTH 寄存器中设定的深度，可以写入 sample 数据。				
reserved	31:1	RW	31’d0	保留
sec_audio_fifo_ready	0	RW	1’b0	1：direct audio sample FIFO 中数据个数少于 SEC_AUDIO_FIFO_DEPTH 寄存器中设定的深度，可以写入 sample 数据
SEC_INFOFRAME_SELECT				
选择要写入内部 SRAM 的 InfoFrame SDP 的类型。不同类型的 InfoFrame 在 SRAM 中有对应的地址范围。				
reserved	31:3	RW	29’d0	保留
sec_info_select	2:0	RW	3’d0	0 : Vendor Specific 1 :AUX Video Information 2 : Source Product Description 3 : Audio Description 4 : NTSC VBI
SEC_INFOFRAME_DATA				
写入的 InfoFrame SDP 数据 ，根据 SEC_INFOFRAME_SELECT 寄存器设置的值，将数据写入 InfoFrame SRAM 对应的地址。读寄存器将返回上一次写入的值。				
reserved	31:8	RW	24’d0	保留
sec_info_data	7:0	RW	8’d0	
SEC_TIMESTAMP_INTERVAL				
发送每个 audio timestamp packet 间等待的时间间隔（us）。若设为 0，每个场消隐期间控制器将只发送一次 audio timestamp packet				
reserved	31:8	RW	24’d0	保留

set_timestamp_interval	7:0	RW	8'd0	发送每个 audio timestamp packet 间等待的时间间隔,单位 us。
SEC_CS_SOURCE_FORMAT				
此寄存器用于定义随每个音频帧发送的信道状态信息中的字段。适当的位会自动插入到音频帧的通道状态字段中。				
reserved	31:8	RW	24'd0	保留
source_num	7:4	RW	4'd0	具有唯一源代码标识符的四位代码
linear_pcm	3	RW	1'd0	0：LPCM samples；1：encoded samples
pcm_audio_format	2:0	RW	3'd0	与通道状态的位 5-3 相对应的三位代码。
SEC_CS_CATEGORY_CODE				
通道状态类别代码				
reserved	31:8	RW	24'd0	保留
sec_cs_category_code	7:0	RW	8'd0	8 位分类码，表示 IEC60958-3 相关附件中定义的数字音频信号的设备类型。
SEC_CS_LENGTH_ORIG_FREQ				
通道状态字长和原始采样频率。这些值被插入到音频流次要数据包的信道状态位内的适当字段中。				
reserved	31:8	RW	24'd0	保留
sample_word_length	7:4	RW	4'd0	IEC 60958-3 规范中信道状态字段位 32-35 中规定的每个采样字的长度。
orig_sampling_freq	3:0	RW	4'd0	携带音频信号的原始采样频率。这些位反映 IEC60958-3 信道状态字段中位 39-36 的值。
SEC_CS_FREQ_CLOCK_ACCURACY				
通道状态类别代码				
reserved	31:8	RW	24'd0	保留
sampling_freq	7:4	RW	4'0	当前音频信号的采样频率，其值反映位 24-27。
clock_accuracy	3:0	RW	4'0	当前采样频率的时钟精度代码。此代码相当于信道状态字段的位 28-29。
SEC_CS_COPYRIGHT				
音频流通道状态位的版权代码				
reserved	31:3	RW	29'd0	保留
cgms-a	2:1	RW	2'd0	复制当前音频流的生成管理系统信息。
copyright	0	RW	1'd0	版权声明的 CP 位。不支持交替此位的值以指示未知状态的模式。

SEC_GTC_COUNT_CONFIG				
配置用于管理全局时间码函数的内部累加器。累加器由 8 位整数部分和 16 位小数部分组成。在每个主机时钟周期，分数和整数部分被添加到各自的计数器。当分数计数器超过 16 位范围时，整数部分会增加一个额外的计数。				
reserved	31:24	RW	8’d0	保留
gtc_count_int	23:16	RW	8’d0	GTC 计数器每次累加值整数部分
gtc_count_frac	15:0	RW	16’d0	GTC 计数器每次累加值小数部分
SEC_GTC_COMMAND_EDGE				
返回最近一次 AUX 命令边沿(完成 同步后，CMD 的第一个上升沿)时的 GTC 计数值，这个值和 GTC update 事件时发送给 sink 端的 GTC 值相同				
gtc_count_value	31:0	RW	32’d0	GTC 计数值
SEC_AUDIO_CHANNEL_MAP				
音频输入通道映射				
channel_8_map	31:28	RW	4’d0	将 channel 8 输入的音频数据与选择的 1-8 任一 channel 映射。 1：映射到输出 channel 1 2：映射到输出 channel 2 依此类推
channel_7_map	27:28	RW	4’d0	同 channel_8_map
channel_6_map	23:20	RW	4’d0	同 channel_8_map
channel_5_map	19:16	RW	4’d0	同 channel_8_map
channel_4_map	15:12	RW	4’d0	同 channel_8_map
channel_3_map	11:8	RW	4’d0	同 channel_8_map
channel_2_map	7:4	RW	4’d0	同 channel_8_map
channel_1_map	3:0	RW	4’d0	同 channel_8_map

6.5.4.7 数据包直接写入寄存器

域	位	读写	复位值	描述
SEC_DB_LANE_SELECT				
选择要将音频数据写入哪条 lane，每条 lane 的音频数据都存入一个深度 16，宽度 32bit 的 buffer。				
reserved	31:2	RW	30’d0	保留
lane_select	1:0	RW	2’d0	00：lane0 01：lane1 10：lane2 11：lane3
SEC_DB_WRITE_INDEX				
允许直接设置缓冲区的写入索引。索引值 0-15 将允许写入第一个缓冲区，而索引值 16-31 将允许写入第二个缓冲区。一个写索引值用于所有四个通道中的直接缓冲区。				
reserved	31:5	RW	27’d0	保留
buffer_select	4	RW	1’d0	buffer0 或者 buffer1
index	3:0	RW	4’d0	用于将来写操作的直接缓冲区的索引。每次写入操作后，索引值将自动递增。
SEC_DB_DATA_COUNT				

写入的 sdp 中 valid link symbol 的个数，应在设置 SEC_DB_READY 前（即开始从 buffer 中读取数据前）设置合适的 data count。Data count 范围 1-16				
reserved	31:6	RW	26'd0	保留
count	5:0	RW	6'd0	写入的 sdp 中 valid link symbol 的个数
SEC_DB_DATA				
用于将来写操作的直接缓冲区的索引。每次写入后，索引值将自动递增操作。链接符号数据对写入辅助数据包直接缓冲区。写入时，此寄存器中的 16 位值将使用 SEC_DB_ WRITE 索引值设置的索引存储在直接缓冲区中。每次写入操作后，索引将自动递增				
sec_db_dat	31:0	RW	32'd0	
SEC_DB_READY				
在正确写入直接缓冲区后，可以通过设置适当的 SEC_DB_READY 位来启动 SDP 的传输。当设置为"1"时，内部直接缓冲逻辑将在主链路上为每个启用的通道传输辅助数据包。在设置这些标志之前，必须正确设置 SEC_DB_DATA_COUNT 寄存器的值。这些寄存器在读取时总是返回"0"。				
reserved	31:2	RO	30'd0	保留
buffer_1_ready	1	RW	1'd0	写 1 开始传输 buffer1 中的数据，从 index32 开始传输
buffer_0_ready	0	RW	1'd0	写 1 开始传输 buffer0 中的数据，从 index0 开始传输
SEC_DB_BUSY				
这些只读状态位表示每个直接缓冲区的状态。当设置为"1"时，当前直接缓冲区正在等待通过主链路传输。当辅助数据包完成传输时，每个位将清除为"0"。在写入 SEC_DB_READY 寄存器后，该位被设置为"1"。				
reserved	31:2	RO	30'd0	保留
buffer_1_busy	1	RW	1'd0	BUFFER1 的传输状态，1 表示 BUFFER1 的数据正在 link 上传输
buffer_0_busy	0	RW	1'd0	BUFFER0 的传输状态，1 表示 BUFFER0 的数据正在 link 上传输
SEC_DB_ENABLE				
direct packet write 接口使能				
reserved	31:1	RO	31'd0	保留
enable	0	RW	1'd0	1：使能 direct packet write 接口，在 direct packet mode 下忽略所有内部产生的 Secondary data packet

6.5.5 DCREQ 寄存器列表

包括 pixel clock 配置寄存器和 MMD AXI 总线地址高位扩展寄存器，与图层信息相关的寄存器具有 double buffered 功能。

寄存器名称	偏移	描述	double buffered
addr_start00	0x00	图层 0 数据起始地址	是
addr_end00	0x04	图层 0 数据结束地址	是

addr_start01	0x08	图层 1 数据起始地址	是
addr_end01	0x0c	图层 1 数据结束地址	是
layer0_cfg	0x10	图层 0 的配置信息	是
layer1_cfg	0x14	图层 1 的配置信息	是
clear_color0_l	0x18	图层 0 clear color 低 32 位	是
clear_color0_h	0x1c	图层 0 clear color 高 32 位	是
clear_color1_l	0x20	图层 1 clear color 低 32 位	是
clear_color1_h	0x24	图层 1 clear color 高 32 位	是
ch0123_val0	0x28	图层 0 ARGB constant color	是
ch0123_val1	0x2c	图层 1 ARGB constant color	是
yuv_val0	0x30	图层 0 YUV constant color	是
yuv_val1	0x34	图层 1 YUV constant color	是
dc_pixel_clk_config	0x38	DC 像素时钟配置	否
fbdc_cr_core_id_bp	0x40	FBDC produce code 和 branch code	否
fbdc_cr_dore_id_nv	0x44	FBDC version 和 scalable core code	否
fbdc_cr_core_id_c	0x48	FBDC configuration code	否
fb_addr_prefix	0x50	DMA 访问地址高 8 位	否
filter	0x58	FBDC filter 信息寄存器	否
dc_req_enable	0x5c	dcreq 使能开关	否
dc_req_wrapper_soft_rstn	0x68	dcreq 软复位	否

6.5.6 DCREQ 寄存器说明

6.5.6.1 addr_startxx 和 addr_endxx

当前图层图像数据起始地址和结束地址的低 32 位，图像数据存储在主存或者显存中，必须是连续的一段空间，地址 128 字节对齐。DCREQ 共支持两个图层，图层 0 对应 addr_start00 和 addr_end00，图层 1 对应 addr_start01 和 addr_end01。

域	位	读写	复位值	描述
addr_start00				
addr_start00	31:0	RW	32’d0	图层 0 图像数据起始地址低 32 位
addr_end00				
addr_end00	31:0	RW	32’d0	图层 0 图像数据结束地址低 32 位
addr_start01				
addr_start01	31:0	RW	32’d0	图层 1 起始地址低 32 位
addr_end01				
addr_end01	31:0	RW	32’d0	图层 1 图像数据结束地址低 32 位

6.5.6.2 layerx_cfg

当前图层信息配置寄存器，主要描述了来自 GPU 端的数据的压缩信息，共有两个图层，图层 0 对应 layer0_cfg，图层 1 对应 layer1_cfg，寄存器结构完全一致。

域	位	读写	复位值	描述
layer0_cfg/ layer1_cfg				
reserved	31:17	RO	15'd0	保留
mode	16	RW	1'd0	图层的模式，DCREQ 只处理 tile 模式的数据 0：linear；1：tile
reserved	15	RO	1'd0	保留
color_format	14:8	RW	7'd0	表示 GPU 端颜色格式参数，必须与 GPU 和 DC 端配置相同。 0x0E ： ARGB2101010 （ 对 应 DC 中 A2R10G10B10） 0x29：BGRA8888（对应 DC 中 A8R8G8B8，DCREQ 输出给 DC 的数据已经把 BGRA 颜色分量顺序转换成了 ARGB 顺序） 0x02：A4R4G4B4（对应 DC 的 A4R4G4B4） 0x04：A1R5G5B5（对应 DC 的 A1R5G5B5） 0x05：R5G6B5（对应 DC 的 R5G6B5） 0x59：YUV422-VYUY（对应 DC 的 YUY2） 0x5B：YUV422-YVYU（对应 DC 的 UYVY）
argb_swizzle	7:4	RW	4'd0	通知 FDBC 输入的 ARGB 颜色分量顺序，需要和 GPU 端与 DC 端使用相同的配置 以下用于 ARGB 和 RGB： 0b0000：ARGB 0b0001：ARBG 0b0010：AGRB 0b0011：AGBR 0b0100：ABGR 0b0110-0b0111：保留 以下仅用于 ARGB： 0b1000：RGBA 0b1001：RBGA 0b1010：GRBA 0b1011：GBRA 0b1100：BGRA 0b1101：BRGA 0b1110-0b1111：保留
reserved	3	RO	1'd0	保留
tile_type	2:1	RW	2'd0	表示 GPU 端使用的 tile 类型 0：保留 1：8×8（tilemode0 时使用） 2：16×4（tilemode3 时使用） 3：32×2
lossy	0	RW	1'd0	表示该图层使用有损还是无损压缩算法 0：无损压缩；1：有损压缩

6.5.6.3 clear_color

Clear color 相关的寄存器共有 4 个，分别是通道 0 的 clear_color0_l、clear_color0_h 和通道 1 的 clear_color1_l、clear_color1_h。clear color 共 64 位，所以分两个寄存器表示一个通道的 clear color，需要和 GPU 配置成相同的值。

域	位	读写	复位值	描述
clear_color0_l				
clear_color0_l	31:0	RW	32'd0	图层 0 的 clear color 低 32 位，需要和 GPU 保持一致
clear_color0_h				
clear_color0_h	31:0	RW	32'd0	图层 0 的 clear color 高 32 位，需要和 GPU 保持一致
clear_color1_l				
clear_color01_l	31:0	RW	32'd0	图层 1 的 clear color 低 32 位，需要和 GPU 保持一致
clear_color1_h				
clear_color1_h	31:0	RW	32'd0	图层 1 的 clear color 高 32 位，需要和 GPU 保持一致

6.5.6.4 ch0123_valx

ch0123_val 表示 ARGB constant color，需要和 GPU 端保持一致。ch0123_valx 共有两个寄存器，图层 0 对应 ch0123_val0，图层 1 对应 ch0123_val1，寄存器结构完全一致。

域	位	读写	复位值	描述
ch0123_val0/ch0123_val1				
alpha	31:24	RW	32'd0	alpha 分量
red	23:16	RW	32'd0	red 分量
green	15:8	RW	32'd0	green 分量
blue	7:0	RW	32'd0	blue 分量

6.5.6.5 yuv_valx

yuv_val 表示 yuv 颜色分量，具体分为 y 分量和 uv 分量，需要和 GPU 端保持一致。yuv_valx 共有两个寄存器，图层 0 对应 yuv_val0，图层 1 对应 yuv_val1，寄存器结构完全一致。

域	位	读写	复位值	描述
yuv_val0/yuv_val1				
reserved	31:26	RO	6'd0	保留
UV	25:18	RW	8'd0	UV 分量的 constant color
reserved	17:10	RO	8'd0	保留

Y	9:2	RW	8’d0	Y 分量的 constant color
reserved	1:0	RO	2’d0	保留

6.5.6.6 dc_pixel_clk_config

此寄存器可以用来进行 DC 模块的像素时钟配置，像素时钟与分辨率、刷新率等因素有关，本寄存器的配置值可以是任意像素时钟的值，单位是 KHz。

域	位	读写	复位值	描述
dc_pixel_clk_config				
ack	31	RO	1’d0	更改像素始终后时钟的稳定情况。 1：时钟已稳定；0：时钟未稳定
request	30	RW	1’d0	配置像素时钟的请求。 1：请求更改像素时钟；0：停止像素时钟更改
freq_sel	29:0	RW	30’d0	要配置的 DC 像素时钟的值，单位 KHz

6.5.6.7 fdbc 版本信息

fbdc 的相关版本信息可以通过寄存器读出。

域	位	读写	复位值	描述
fbdc_cr_core_id_bp				
fbdc_cr_core_id_b	31:16	RO	16’d0	fbdc 编号（branch code）
fbdc_cr_core_id_p	15:0	RO	16’d0	fbdc 编号（product code）
fbdc_cr_dore_id_nv				
fbdc_cr_core_id_n	31:16	RO	16’d0	fbdc 编号（scalable core code）
fbdc_cr_core_id_v	15:0	RO	16’d0	fbdc 编号（version code）
fbdc_cr_core_id_c				
reserved	31:16	RO	16’d0	保留
fbdc_cr_core_id_c	15:0	RO	16’d0	fbdc 编号（configuration code）

6.5.6.8 AXI-DMA 总线地址扩展

fb_addr_prefix 对原有的 32 位总线地址扩展到了 40 位，这个地址扩展同时作用于 DCREQ、DC 中的图像数据地址。

域	位	读写	复位值	描述
fb_addr_prefix				
reserved	31:8	RO	24’d0	保留
prefix	7:0	RW	8’d0	AXI 访问 DMA 数据总线地址扩展，同时作用域 DCREQ 和 DC 中的有关图像数据地址的寄存器。

6.5.6.9 Filter

域	位	读写	复位值	描述
filter				
reserved	31:9	RO	23’d0	保留
filter_status_clear	8	RW	1’d0	该位写入 1 使 filter status 状态位清零
reserved	7:6	RO	2’d0	保留

filter_status	5:4	RO	2'd0	bit 4 对应图层 0，bit5 对应图层 1 1：fbdc 在 filter stage 发现非法帧 0：fbdc 在 filter stage 没有发现非法帧
reserved	3:1	RO	3'd0	保留
filter_enable	0	RW	1'd0	1：启用 fbdc 的 filter stage 0：禁用 fbdc 的 filter stage

6.5.6.10 软复位

dc_req_wrapper_soft_rstn 寄存器用来对 DCREQ 进行软复位。

域	位	读写	复位值	描述
dc_req_wrapper_soft_rstn				
reserved	31:2	RO	30'd0	保留
hold_ready	1	RW	1'd0	1：保持 dcreq 输出的 ready 为 1，arready 为 0 0：停止保持
wrapper_soft_rstn	0	RW	1'd0	1：复位 dc_req_wrapper，配置寄存器不会复位；0：停止复位

6.5.6.11 dcreq 使能

dc_req_enable 控制 DCREQ 模块的使能或禁用，当 DCREQ 使能状态时，处理的图像数据为 tile 压缩模式，当 DCREQ 禁用状态时候，DCREQ 被旁路，处理的数据为 linear 模式。

域	位	读写	复位值	描述
dc_req_enable				
reserved	31:1	RO	31'd0	保留
enable	0	RW	1'd0	1：启用 DCREQ，为 DC 提供数据解压、数据格式处理功能 0：禁用 DCREQ，DC 只能获取 linear 格式的数据

6.5.7 地址转换寄存器列表

寄存器名称	偏移	描述
region3_src_addr	0x24	映射源地址
region3_size	0x28	映射地址大小
region3_dst_addr	0x2C	映射目标地址

6.5.8 地址转换寄存器说明

6.5.8.1 region3_src_addr(0x24)

源地址寄存器，需要配置 pcie 驱动分配后的地址，作为显存地址。

域	位	读写	复位值	描述
reserved	31:22	RO	0x0	保留

addr	21:0	RW	0x0	region3 映射源地址的比特[43:22]，即源地址需要 4MB 边界对齐

6.5.8.2 region3_size(0x28)

映射地址的大小，需要 X100 的固件在初始化阶段配置。

域	位	访问	复位值	描述
enable	31	RW	0x0	region3 映射使能：1：使能；0：不使能
reserved	30:22	RO	0x0	保留
size	21:0	RW	0x00	reigon3 大小，单位为 4MB

6.5.8.3 region3_dst_addr(0x2C)

映射的目标地址，需要 X100 的固件在初始化阶段配置。

域	位	访问	复位值	描述
size	31:22	RW	0x0	region3 本地映射 memory 资源大小，单位为 16MB
addr	21:0	RW	0x0500_0000	region3 映射的目的地址的比特[43:22]，该寄存器在启动时由 SE 配置

7 USB

飞腾 X100 集成了 8 个独立的 USB 控制器。

7.1 操作说明

7.1.1 主机控制器初始化过程

USB 控制器兼容 xHCI，初始化步骤描述如下：

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

主控制器打开并运行，Root Hub 端口将开始报告设备连接等信息，系统软件可以开始枚举设备。

7.1.2 设备枚举过程

7.1.2.1 USB2 总线枚举

- Hub 报告设备连接，通过状态改变管道通知主机这个事件；
- 主机通过查询确定 hub 状态改变的具体信息；
- 主机了解有新设备连接，等待至少 100ms 用于嵌入操作处理以及让设备稳定供电，随后主机向端口发送一个端口使能和复位命令；
- Hub 执行端口复位处理流程，USB 设备进入 reset 状态能从 VBUS 汲取不超过 100mA 的电流，USB 设备所有的寄存器和状态进行复位，响应默认地址；

- 主机为 USB 设备分配一个唯一的地址，设备转换到 Address 状态；
- 在 USB 设备接收到唯一地址之前，其默认控制管道通过默认地址是可访问的，主机读取设备描述符；
- 主机读取设备配置信息；
- 基于配置信息和 USB 设备的用法，主机赋予设备配置值。设备处于 Configured 状态，设备汲取描述符描述的 VBUS 电源值。

7.1.2.2 USB3 总线枚举

- hub 报告设备连接，通过状态改变管道通知主机这个事件；
- 主机通过查询确定 hub 状态改变的具体信息；
- 主机知道哪个端口有新设备连接；
- 如果主机复位，Hub 执行对应端口复位处理流程。复位完成，端口进入回到使能状态；
- USB 设备进入 reset 状态能从 VBUS 汲取不超过 150mA 的电流，USB 设备所有的寄存器和状态进行复位，响应默认地址；
- 主机为 USB 设备分配一个唯一的地址，设备转换到 Address 状态；
- 在 USB 设备接收到唯一地址之前，其默认控制管道通过默认地址是可访问的，主机读取设备描述符；
- 主机设置同步延时通知设备主机发出一个包到设备接收所需要的延时时间；
- 主机使用 SetSEL 请求通知设备系统退出延时；
- 主机读取设备配置信息；
- 主机设置下游端口 U1/U2 超时；
- 基于配置信息和 USB 设备的用法，主机赋予设备配置值。设备处于 Configured 状态，设备汲取描述符描述的 VBUS 电源值。

7.1.3 设备数据传输流程

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

7.1.4 初始化配置流程

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

7.2 寄存器列表

寄存器名称	偏移	功能描述
USB_RESETN_STATUS	0x0	USB 复位状态
USB_SOC_RESETN	0x4	USB 复位控制
USB_MODE_STRAP	0x8	USB 模式选择
USB_AXI_SIDE_CFG	0xC	AXI 边带信号配置
USB_UTMI_SIDE_REG	0x10	USB2.0 UTMI 边带信号配置
USB2PHY_OTG_REG	0x14	OTG 和 HOST 功能使能选择寄存器
USB2PHY_VBUS_REG	0x18	VBUS18 的充放电控制
USB2PHY_PLL_EN	0x1C	USB2PHY PLL 使能选择寄存器
USB2PHY_REFCLK_MODE	0x20	USB2PHY 参考时钟选择
UIB_STATE_COUNTER	0x24	USB2PHY 寄存器访问接口时序配置寄存器
LPI_CTR_COUNTER0	0x2C	LPI 模块控制寄存器
LPI_CTR_COUNTER1	0x30	LPI 模块控制寄存器
LPI_CTR_EN	0x34	LPI 模块控制寄存器
OVERCURRENTN_CTRL	0x38	OVERCURRENT_N 引脚信号内部选通处理

7.3 寄存器说明

7.3.1 直接访问空间映射

UIB 模块接收来自外部的 APB 请求并进行地址译码，其地址空间映射如下图所示，APB 访问空间 128KB，每个空间对应不同处理，处理描述见下表。

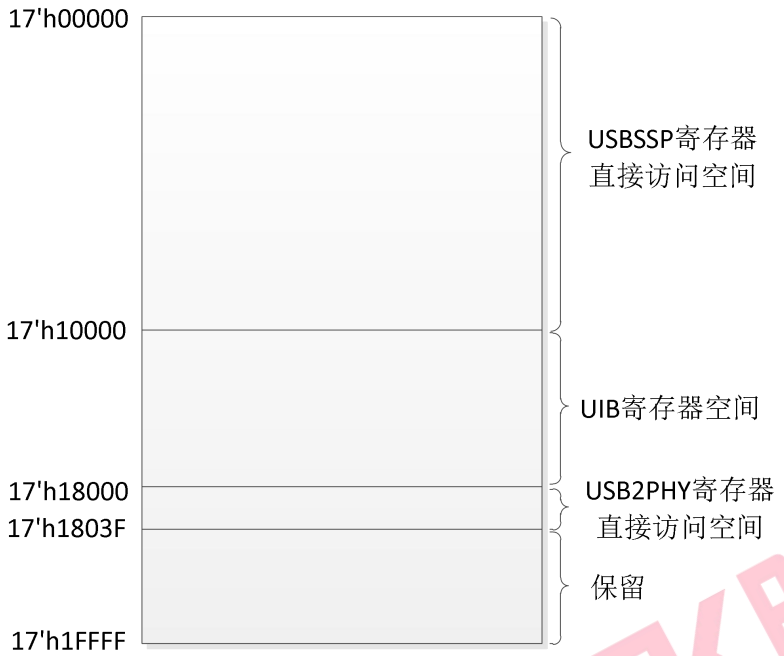

图 7-1　APB 地址空间映射

USBSSP 寄存器直接访问空间	APB 命中此空间直接将 APB 请求转发到 USBSSP 模块
UIB 寄存器空间	APB 命中此空间对 UIB 定义的寄存器进行访问
USB2PHY 寄存器直接访问空间	APB 命中此空间将 APB 请求转换为 USB2PHY 寄存器访问定义的协议的请求

7.3.2　USB_RESETN_STATUS(0x0)

域	位	读写	复位值	描述
reserved	31:24	RW	0x0	保留
cfg_rstn_cnt_val	23:16	RW	0x64	cfg_rstn 复位释放计数值
presetn_cnt_val	15:8	RW	0x4	presetn 复位释放计数值
reserved	7:2	RO	0x0	保留
cfg_rstn_rdy	1	RO	0x0	cfg_rstn 复位释放计数完成
presetn_rdy	0	RO	0x0	presetnn 复位释放计数完成

7.3.3　USB_SOC_RESETN(0x4)

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

7.3.4　USB_MODE_STRAP(0x8)

域	位	读写	复位值	描述
reserved	31:2	RW	0x0	保留
mode_strap	1:0	RW	0x1	与 USBSSP 的 mode_strap 信号相连 00:控制器未初始化配置为主机或设备 01:控制器初始化配置为主机 10:控制器初始化配置为设备

7.3.5 USB_AXI_SIDE_CFG(0xC)

域	位	读写	复位值	描述
awlock	31	RW	0x0	AXI4 awlock 信号
awcache	30:27	RW	0x0	AXI4 awcache 信号
awprot	26:24	RW	0x2	AXI4 awprot 信号
awqos	23:20	RW	0x0	AXI4 awqos 信号
reserved	19:16	RO	0x0	保留
arlock	15	RW	0x0	AXI4 arlock 信号
arcache	14:11	RW	0x0	AXI4 arcache 信号
arprot	10:8	RW	0x2	AXI4 arprot 信号
arqos	7:4	RW	0x0	AXI4 arqos 信号
reserved	3:0	RO	0x0	保留

7.3.6 USB_UTMI_SIDE_REG(0x10)

域	位	读写	复位值	描述
reserved	31:2	RW	0x0	保留
utmi_sleepm_in	1	RW	0x0	连接 USBSSP 的 utmi_sleepm_in 信号，Bypass utmi_sleepm 信号
utmi_suspendm_in	0	RW	0x0	连接 USBSSP 的 utmi_suspendm_in 信号，Bypass utmi_suspendm 信号

7.3.7 USB2PHY_OTG_REG(0x14)

域	位	读写	复位值	描述
reserved	31:2	RW	0x0	保留
otg_suspendm_byps	1	RW	0x0	连接 USB2PHY 的 OTG_SUSPENDM_BYPS 信号。 0:OTG_SUSPENDM 信号由 IDDIG 控制 1:OTG_SUSPENDM 信号由 SOC 控制
otg_suspendm	0	RW	0x0	连接 USB2PHY 的 OTG_SUSPENDM 信号，使能 OTG 和主机功能。OTG_SUSPENDM 信号用于 OTG 应用中的 VBUS 协商和主机应用中的主机断开连接。 OTG 应用中，OTG_SUSPENDM_BYPS 为 0 时，OTG_SUSPENDM 为 IDDIG 的

				相反数，OTG_SUSPENDM_BYPS 为 1 时，OTG_SUSPENDM 为 otg_suspendm 的值 Host 应用中，otg_suspendm_byps 和 otg_suspendm 需设为 1 Device 应用中，otg_suspendm_byps 设为 1，otg_suspendm 设为 0

7.3.8 USB2PHY_VBUS_REG(0x18)

域	位	读写	复位值	描述
reserved	31:2	RW	0x0	保留
discharge_bus	1	RW	0x0	连接 USB2PHY 的 DISCHRG_BUS 信号，VBUS18 放电 0:不通过一个电阻放电 VBUS18 1:通过一个电阻放电 VBUS18（至少持续 50ms 有效）
charge_bus	0	RW	0x0	连接 USB2PHY 的 CHRG_BUS 信号，VBUS18 充电 0:不通过一个电阻充电 VBUS18 1：通过一个电阻充电 VBUS18（至少持续 30ms 有效）

7.3.9 USB2PHY_PLL_EN(0x1C)

域	位	读写	复位值	描述
reserved	31:2	RW	0x0	保留
pll_en_byps	1	RW	0x0	0:PLL_EN 信号来自 utmi_suspendm 信号 1: PLL_EN 信号来自 pll_en 的值
pll_en	0	RW	0x0	pll_en_byps 为 1 有效，连接 USB2PHY 的 PLL_EN 信号

7.3.10 USB2PHY_REFCLK_MODE(0x20)

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

7.3.11 UIB_STATE_COUNTER(0x24)

域	位	读写	复位值	描述
reserved	31:24	RW	0x0	保留
counter_sample_value	23:16	RW	0xF	UIB 协议转换状态机 SAMPLE 状态跳转计数器 counter_sample 初值设置,向下计数
counter_enable_value	15:8	RW	0xF	UIB 协议转换状态机 ENABLE 状态跳

				转计数器 counter_enable 初值设置,向下计数
counter_setup_value	7:0	RW	0xF	UIB 协议转换状态机 SETUP 状态跳转计数器 counter_setup 初值设置,向下计数

7.3.12 LPI_CTR_COUNTER0(0x2C)

域	位	读写	复位值	描述
wait_cnt0	31:0	RW	0x0	连接 LPI_CTR 模块的 wait_cnt[31:0]信号，用于 Q-channel 0 通道

7.3.13 LPI_CTR_COUNTER1(0x30)

域	位	读写	复位值	描述
wait_cnt1	31:0	RW	0x0	连接 LPI_CTR 模块的 wait_cnt[63:32]信号，用于 Q-channel 1 通道

7.3.14 LPI_CTR_EN(0x34)

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

7.3.15 OVERCURRENTN_CTRL(0x38)

域	位	读写	复位值	描述
reserved	31:2	RW	0x0	保留
overcurrentn_byps	1	RW	0x0	0: overcurrentn 信号来自 overcurrent_npad 信号 1: overcurrentn 信号来自 overcurren_reg 的值
overcurrentn_reg	0	RW	0x0	overcurrentn_byps 为 1 有效，overruentn 信号寄存器值

8 SATA

飞腾 X100 中的 SATA 兼容 SATA3.0 规范，兼容 AHCI1.3 规范，不支持 SATA 原生命令操作，详细内容可参考《飞腾 X100 套片数据手册》《Serial ATA International Organization:Serial ATA Revision 3.0》和《Serial ATA Advanced Host Controller Interface (AHCI) 1.3.1》。

9 PS2 Controller

9.1 操作说明

9.1.1 控制器初始化流程

- 配置 CONTROL 寄存器的 bit[0]，触发复位，延时 4ms 后解除复位。
- 配置 CONTROL 寄存器的 bit[3:1]，使能各个中断。
- 配置 TIMER_VAL 寄存器。

9.1.2 中断处理流程

- 读取 STATUS 寄存器，获取 bit[1:0]、bit[12:8]的信息。
- 若 bit[12]、bit[1]、bit[0] 有任意一个不为 0，跳转错误中断处理。
- 若 bit[12]、bit[1]、bit[0] 都为 0，跳转接收中断处理。

9.1.2.1 错误中断处理流程

- 读取 RX_BUFFER 寄存器，读取次数为 STATUS 寄存器的 bit[12:8]。
- 配置 INTERRUPT 寄存器，清所有中断。

9.1.2.2 接收中断处理流程

- 读取 RX_BUFFER 寄存器，读取次数为 STATUS 寄存器的 bit[12:8]，将读出的数据返给系统内核。
- 配置 INTERRUPT 寄存器，清接收中断。

9.2 寄存器说明

寄存器名称	读写	偏移	描述
STATUS	RO	0x00	bit[0]:transmit timeout error，1 有效，读清 0 bit[1]:response timeout error，1 有效，读清 0 bit[2]:tx full，1 有效，表示 TX BUFFER 满了，不能继续写入数据 bit[7:4]保留 bit[12:8]:rx counter，表示当前 RX BUFFER 缓存数据的个数 bit[15:13]保留
CONTROL	RW	0x04	bit[0]:复位信号，1 有效，复位所有 FIFO 及 PS2 接口，默认值为 0 bit[1]: transmit timeout err 事件使能，1 使能，0 禁止，默认值为 0 bit[2]: response timeout err 事件使能，1 使能，0 禁止，默认值为 0 bit[3]:rx 中断使能，1 使能，0 禁止，默认值为 1

			bit[15:4]保留
INTERRUPT	RW	0x08	bit[0]:写 1 清 timeout 中断 bit[1]:写 1 清 rx 中断 bit[7:2]保留
TX_BUFFER	WO	0x0C	将数据写入该寄存器，PS2 接口将其发送出去
RX_BUFFER	RO	0x10	接收数据，bit[8]为 parity error，1 为 error；当 RX_BUFFER 读空且继续读取时，读出值为 0x1ff
TIMER_VAL	RW	0x14	表示当前 apb 时钟下计时 5us 所需的计数值，例如：1MHz 时钟下，值为 5；10MHz 时钟下，值为 50（十进制），依此类推。通过配置该值，以适应不同频率的工作时钟

10 MMCSD

10.1 操作说明

10.1.1 MMCSD 寄存器初始化配置

- 电源使能，配置寄存器 pwren；
- 配置 ctrl 寄存器，如全局中断使能等；
- 配置 uhs_reg_ext 寄存器，设置一级时钟分频参数；
- 读 gpio 寄存器，确认时钟分频完成；
- 配置 clkena 寄存器，关断时钟；
- 配置 clkdiv 寄存器，设置二级时钟分频参数；
- 配置 clkena 寄存器，开启时钟；
- 配置 cmd 寄存器，更新时钟，并确定时钟更新完成。

10.1.2 MMCSD 读/写操作流程

- 将数据大小（以字节为单位）写入到 bytcnt 寄存器中。对于多数据块读/写操作，bytcnt 必须是数据块大小的倍数；
- 将数据块大小（以字节为单位）写入到 blksiz 寄存器中；
- 将数据读操作的起始数据地址（卡）写入到 cmdarg 寄存器；
- 写 cmd 参数寄存器中。对于 SD 和 MMC 卡，READ_SINGLE_BLOCK (CMD17)命令用于单数据块读操作，将 READ_MULTIPLE_BLOCK (CMD18)命令用于多数据块读操作。WRITE_SINGLE_BLOCK(CMD24)命令用于单数据块读操作，将 WRITE_MULTIPLE_BLOCK(CMD25)命令用于多数据块读操作。对于 SDIO 卡，将 IO_RW_EXTENDED (CMD53)命令用于单数据块以及多数据块传递。写入 cmd 寄存器后，软件层开始执行命令。发送命令到总线后生成 Command Done 中断；
- 软件必须检查 raw_ints 寄存器中 dcrc、bds、sbe 和 ebe 比特中报告的数据错误中断。如果需要，软件能够发送一个 SD/SDIO STOP 命令来终止数据传递；
- 软件必须检查 raw_ints 寄存器中的软件层超时条件。

软件层操作流程：

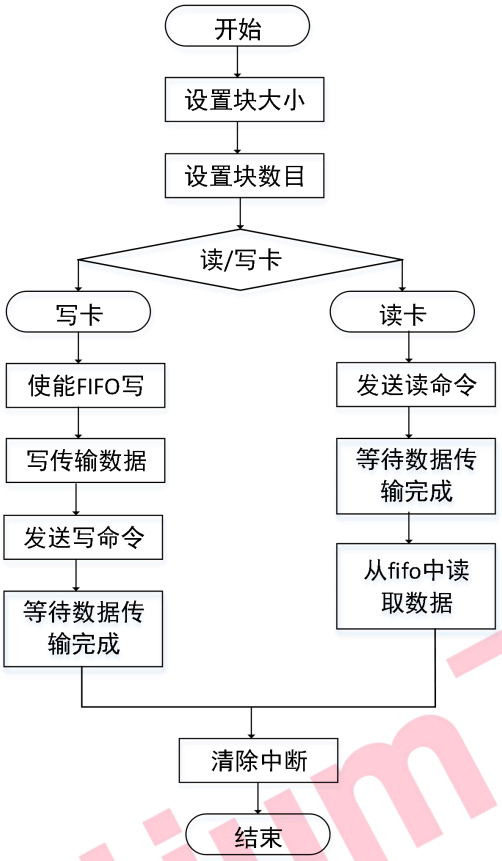

图 10-1 软件层操作流程

10.1.3 SDIO 读操作

10.1.3.1 SDIO 8-bit Read

命令	描述
CMD52	单字节读指令

10.1.3.2 SDIO 16-bit Read

命令	描述
CMD53	双字节读指令

10.1.3.3 SDIO 32-bit Read

命令	描述
CMD53	四字节读指令

10.1.3.4 SDIO FIFO Read

命令	描述
CMD53	FIFO 读指令

10.1.4 SDIO 写操作

10.1.4.1 SDIO 8-bit Write

命令	描述
CMD52	单字节写指令

10.1.4.2 SDIO 16-bit Write

命令	描述
CMD53	双字节写指令

10.1.4.3 SDIO 32-bit Write

命令	描述
CMD53	四字节写指令

10.1.4.4 SDIO FIFO Write

命令	描述
CMD53	FIFO 写指令

数据传输类指令格式如下图，详情可参考《SDIO Simplified Specification Version 3.00》。

S	D	Command Index 110101b	R/W flag	Function Number	Block Mode	OP Code	Register Address	Byte/Block Count	CRC7	E
1	1	6	1	3	1	1	17	9	7	1
			参数							

图 10-2 io_rw_extended 命令-CMD53

10.2 寄存器列表

寄存器名称	偏移	描述
cntrl	0x0	控制寄存器
pwren	0x4	卡供电开关（负载层开关，不在协议范围，作用相当于一个 gpio）
clkdiv	0x8	时钟驱动
clkena	0x10	时钟控制
tmout	0x14	超时
ctype	0x18	卡位宽配置
blksiz	0x1c	块大小
bytcnt	0x20	传输字节数
int_mask_n	0x24	中断使能
cmdarg	0x28	命令参数
cmd	0x2C	命令
resp0	0x30	响应寄存器 0
resp1	0x34	响应寄存器 1
resp2	0x38	响应寄存器 2
resp3	0x3C	响应寄存器 3
masked_ints	0x40	中断状态寄存器
raw_ints	0x44	中断清除（写 1 清除）
status	0x48	卡状态寄存器
fifoth	0x4C	fifo 深度
card_detect_biu	0x50	卡在位情况
card_write_prt_biu	0x54	写保护使能

gpio	0x58	CIU 时钟 ready
tran_crd_cnt_mx	0x5C	CIU 到卡传输的字节数
tran_fifo_cnt_mx	0x60	MEM & FIFO 之间传输的字节数
debnce	0x64	去抖时钟数，参考值 5-25ms
uid	0x68	用户 ID
vid	0x6C	控制器版本
hcon	0x70	保留
uhs_reg	0x74	外部调压器接口电压
card_reset	0x78	复位
status_reg	0x90	状态寄存器
intr_en_reg	0x94	中断使能
cardthctl	0x100	读卡深度
uhs_reg_ext	0x108	一级时钟分频
emmc_ddr_reg	0x10c	eMMC ddr 开始位
enable_shift	0x110	移位使能
data	0x200	数据 FIFO 寄存器

10.3 时钟配置参考

MMCSD 卡控制器时钟结构包括总线接口时钟（APB、AXI）、CIU（DEVICE 适配控制器）时钟、 卡驱动时钟（寄存器输出）；CIU 时钟用于产生 DEVICE 适配时钟，它由控制器外部输入高频时钟源通过一级门控时钟调频模块生成，调频参数可根据 SD 卡工作频率及低功耗需求更改 CIU 时钟单元时钟。DEVICE 驱动时钟由调频之后的 CIU 时钟经过寄存器分频方式产生。注：时钟结构如下图所示，门控调频单元相位调整部分为作为细粒度调整，卡时钟控制单元相位调整部分作为区间粗粒度调整。

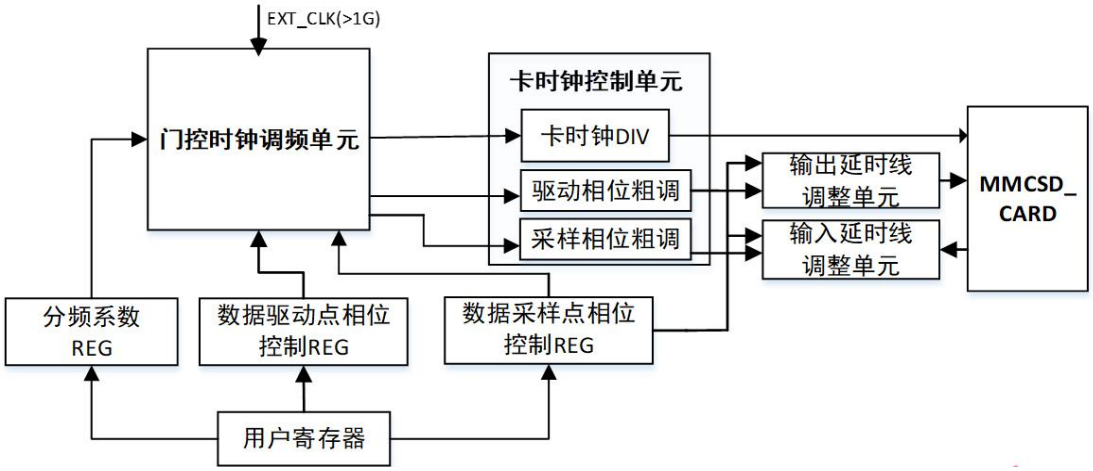

图 10-3 时钟结构

驱动点相位控制：用于驱动卡命令和数据输出更新时刻（相对 SD 卡时钟）。

数据采样点相位控制：用于接收卡命令和数据的采样时刻（相对 SD 卡时钟）。

CARD 时钟与驱动采样相位的调整由 clkdiv(0x8)与 uhs_reg_ext(0x108)配合完成。uhs_reg_ext(0x108)分频参数决定了 CIU 主模块时钟频率，clkdiv(0x8)分频参数决定了 CARD 工作时钟频率。具体配置参照寄存器部分：

- 一级分频

通过 uhs_reg_ext(0x108)寄存器配置 CIU 主时钟分频参数及 drv、samp 相位。

分频参数= CLK_DIV_CTRL+1

注意：CCLK_IN、CCLK_IN_DRV、CCLK_IN_SAMPLE 为一级分频后的时钟，均有 clock gating 生成，占空比非 50%。

CCLK_IN_DRV、CCLK_IN_SAMPLE 均为 CCLK_IN 相对相移，因此必须小于等于 CLK_DIV_CTRL。CCLK_IN_DRV 为控制器一级驱动数据点，CCLK_IN_SAMPLE 为控制器一级采样数据点。

- 二级分频

通过 clkdiv(0x8)寄存器配置 CARD 时钟。

分频参数= 2* CLK_DIVIDER [7:0]

注意：CLK_DRV、CLK_SMPL 必须小于 CLK_DIVIDER，且必须满足 CLK_SMPL= CLK_DRV+1。

最小配置参数：CLK_DIVIDER=1，CLK_DRV=0，CLK_SMPL=1。

cclk_out 为最终生成的卡时钟，cclk_out_en 为控制器二级驱动数据点，cclk_smpl_en 为控制器二级采样数据点。最终的数据驱动时刻和采样时刻由一二级配置参数共同决定。

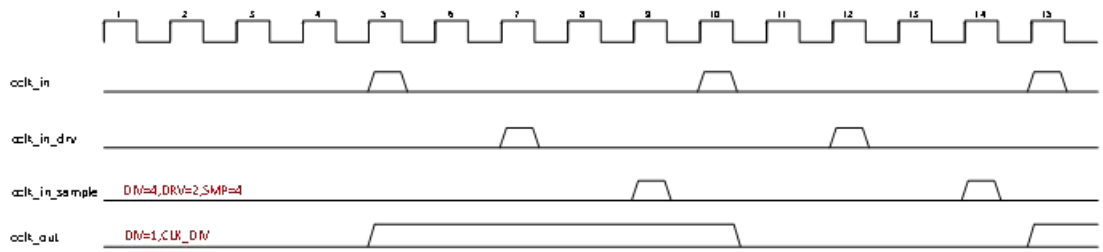

图 10-4　时钟参数设置参考 1

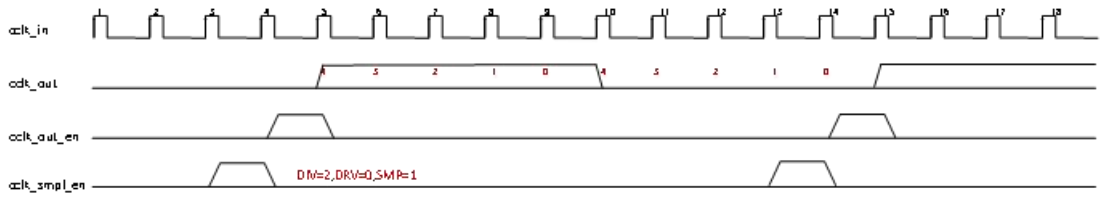

图 10-5　时钟参数设置参考 2

时钟分频配置流程

- uhs_reg_ext(0x108)配置 CIU 时钟及驱动相位和 EXT_CLK_ENABLE；

注意：修改时钟频率前要先禁用 EXT_CLK_ENABLE 位，否则修改不生效，配置完成，再开启使能。

- clkdiv(0x8)配置 CARD 时钟分频系数

注意：配置分频之前要先关断 clkena[0]，配置完成，再开启使能。

- cmd[21][31]配 1，更新时钟，启动命令

注意：USE_HOLD_REG 必须为 1，否则发出的命令会有问题。

时钟配置参数参考值：表中 D2 表示十进制数 2，相位参数[0:2]表示可以选择 0、1、2 这三个十进制数。

表 10-1　分频参数表

频率/Mhz	一级分频参数(uhs_reg_ext(0x108))			二级分频参数(clkdiv(0x8))		
	分频参数 CLK_DIV_CTRL	输出相位参数 CLK_DRV_PHASE_CTRL	采样相位参数 CLK_SMPL_PHASE_CTRL	分频参数 CLK_DIVIDER	输出相位参数 CLK_DRV	采样相位参数 CLK_SMPL
12.5	D2	[0:2]	[0:2]	D16	[15:0]	[15:0]
	D3	[0:3]	[0:3]	D12	[11:0]	[11:0]
	D5	[0:5]	[0:5]	D8	[7:0]	[7:0]
25	D2	[0:2]	[0:2]	D8	[7:0]	[7:0]
	D3	[0:3]	[0:3]	D6	[5:0]	[5:0]
	D5	[0:5]	[0:5]	D4	[3:0]	[3:0]
50	D2	[0:2]	[0:2]	D4	[3:0]	[3:0]
	D3	[0:3]	[0:3]	D3	[2:0]	[3:0]
	D5	[0:5]	[0:5]	D2	[1:0]	[1:0]
100	D2	[0:2]	[0:2]	D2	[1:0]	[1:0]
	D5	[0:5]	[0:5]	D1	0	0

10.4 寄存器说明

10.4.1 cntrl(0x0)

域	位	读写	复位值	描述
reserved	31:26	RW	0x0	保留
reserved	25	RW	0x0	保留
ENABLE_OD_PULLUP	24	RW	0x0	外部开漏输出
CARD_VOLTAGE_B	23:20	RW	0x0	B 电压选择
CARD_VOLTAGE_A	19:16	RW	0x0	A 电压选择
ENDIAN	11	RW	0x0	0: 小端；1: 大端
SEND_AUTO_STOP_CCSD	10	RW	0x0	对应 CCD,自动 STOP(不支持)
SEND_CCSD	9	RW	0x0	发送 CCSD (不支持)
ABORT_READ_DATA	8	RW	0x0	读暂停异常清楚 data FSM
SEND_IRQ_RESPONSE	7	RW	0x0	MMC 中断自动响应配置。
READ_WAIT	6	RW	0x0	SDIO 读等待
reserved	5	RW	0x0	保留
INT_ENABLE	4	RW	0x0	全局中断使能配置
reserved	3	RW	0x0	保留
reserved	2	RW	0x0	保留
FIFO_RESET	1	RW	0x0	复位 FIFO
CONTROLLER_RESET	0	RW	0x0	复位控制器，除 FIFO

10.4.2 pwren(0x4)

域	位	读写	复位值	描述
POWER_ENABLE	31:0	RW	0x0	卡供电开关。0：关；1：开

10.4.3 clkdiv(0x8)

域	位	读写	复位值	描述
reserved	31:24	RW	0x0	保留
CLK_SMPL	23:16	RW	0x0	采样相位区间设置
CLK_DRV	15:8	RW	0x0	输出相位区间设置
CLK_DIVIDER	7:0	RW	0x0	时钟分频参数设置，分频参数=2*CLK_DIVIDER

10.4.4 clkena(0x10)

域	位	读写	复位值	描述
CCLK_LOW_POWER	31:16	RW	0x0	功耗模式控制 0x0：非低功耗；0x1：低功耗
CCLK_ENABLE	15:0	RW	0x0	card 时钟使能控制 0：Clock disabled；1：Clock enabled

10.4.5 tmout(0x14)

域	位	读写	复位值	描述
DATA_TIMEOUT	31:8	RW	0xFFFFFF	读卡超时（以卡时钟为单位）
RESPONSE_TIMEOUT	7:0	RW	0x40	响应超时（以卡时钟为单位）

10.4.6 ctype(0x18)

域	位	读写	复位值	描述
CARD0_WIDTH1	31:16	RW	0x0	0：Non 8-bit mode；1：8-bit mode
CARD0_WIDTH2	15:0	RW	0x0	0：1-bit mode；1：4-bit mode

10.4.7 blksiz(0x1C)

域	位	读写	复位值	描述
BLOCK_SIZE	31:0	RW	0x200	块大小

10.4.8 bytcnt(0x20)

域	位	读写	复位值	描述
BYTE_COUNT	31:0	RW	0x0	传输字节数

10.4.9 int_mask_n(0x24)

域	位	读写	复位值	描述
reserved	31:17	RW	0x0	保留
SDIO_INT_MASK_CARD0	16	RW	0x0	SDIO interrupt 中断 0：屏蔽；1：使能
EBE_INT_MASK	15	RW	0x0	读写结束位错误/写未收到 CRC 中断 0：屏蔽；1：使能
ACD_INT_MASK	14	RW	0x0	Auto command 完成中断 0：屏蔽；1：使能
SBE_BCI_INT_MASK	13	RW	0x0	起始位错误/busy 撤销中断 0：屏蔽；1：使能
HLE_INT_MASK	12	RW	0x0	硬件锁存中断 0：屏蔽；1：使能
FRUN_INT_MASK	11	RW	0x0	FIFO 上下溢中断 0：屏蔽；1：使能
HTO_INT_MASK	10	RW	0x0	数据 starv/电源切换中断 0：屏蔽；1：使能
DRTO_INT_MASK	9	RW	0x0	数据读超时中断 0：屏蔽；1：使能
RTO_INT_MASK	8	RW	0x0	响应超时中断 0：屏蔽；1：使能
DCRC_INT_MASK	7	RW	0x0	数据 CRC 校验错误中断

				0：屏蔽；1：使能
RCRC_INT_MASK	6	RW	0x0	响应 CRC 错误中断. 0：屏蔽；1：使能
RXDR_INT_MASK	5	RW	0x0	接收 FIFO 请求中断 0：屏蔽；1：使能
TXDR_INT_MASK	4	RW	0x0	发送 FIFO 请求中断 0：屏蔽 ； 1：使能
DTO_INT_MASK	3	RW	0x0	Data transfer over (DTO) interrupt enable. 0：屏蔽；1：使能
CMD_INT_MASK	2	RW	0x0	命令传输完成中断 0：屏蔽；1：使能
RE_INT_MASK	1	RW	0x0	响应错误中断 0：屏蔽；1：使能
CD_INT_MASK	0	RW	0x0	卡检测中断 0：屏蔽；1：使能

10.4.10 cmdarg(0x28)

域	位	读写	复位值	描述
CMD_ARG	31:0	RW	0x0	命令参数

10.4.11 cmd(0x2C)

域	位	读写	复位值	描述
START_CMD	31:30	RW	0x0	启动命令
USE_HOLD_REG	29	RW	0x1	0：旁路 HOLD Register 1：使能 HOLD Register
VOLT_SWITCH	28	RW	0x0	0：无电压切换；1：使能电压切换
BOOT_MODE	27	RW	0x0	0： Mandatory Boot； 1：Alternate Boot
DISABLE_BOOT	26	RW	0x0	中止 boot 进程。
EXPECT_BOOT_ACK	25	RW	0x0	Expect book ack
ENABLE_BOOT	24	RW	0x0	使能 boot for mandatory
reserved	23:22	RW	0x0	保留
UPDATE_CLOCK_REGISTERS_ONLY	21	RW	0x0	1：不发送命令，只更新时钟 REG
CARD_NUMBER	20:16	RW	0x0	保留
SEND_INITIALIZATION	15	RW	0x0	在发送命令之前，等待 80cyc 初始时钟序列完成。
STOP_ABORT_CMD	14	RW	0x0	1：stop/abort 操作
WAIT_PRVDATA_COMPLETE	13	RW	0x0	0：立即发送命令
SEND_AUTO_STOP	12	RW	0x0	1：自动发送 stop

TRANSFER_MODE	11	RW	0x0	0：block
READ_WRITE	10	RW	0x0	0：读卡；1：写卡
DATA_EXPECTED	9	RW	0x0	0：DTA 无数据；1：DAT 有数据
CHECK_RESPONSE_CRC	8	RW	0x0	0：不检查 CRC；1：检查 CRC
RESPONSE_LENGTH	7	RW	0x0	0：短响应；1：长响应
RESPONSE_EXPECT	6	RW	0x0	0：无响应；1：有响应
CMD_INDEX	5:0	RW	0x0	命令索引

10.4.12 resp0(0x30)

域	位	读写	复位值	描述
RESPONSE0	31:0	RO	0x0	响应寄存器 0 Bit[31:0]

10.4.13 resp1(0x34)

域	位	读写	复位值	描述
RESPONSE1	31:0	RO	0x0	响应寄存器 1Bit[63:32]

10.4.14 resp2(0x38)

域	位	读写	复位值	描述
RESPONSE2	31:0	RO	0x0	响应寄存器 2Bit[95:64]

10.4.15 resp3(0x3C)

域	位	读写	复位值	描述
RESPONSE3	31:0	RO	0x0	响应寄存器 3Bit[127:96]

10.4.16 masked_ints(0x40)

域	位	读写	复位值	描述
SDIO_INTERRUPT_CARD0	31:16	RO	0x0	ISDIO card 中断
END_BIT_ERROR_INTERRUPT	15	RO	0x0	读写 End-bit 错误/写未收到 CRC
AUTO_COMMAND_DONE_INTERRUPT	14	RO	0x0	自动命令完成 (ACD) .
BUSY_COMPLETE_INTERRUPT_INTERRUPT	13	RO	0x0	起始位错误(SBE)/Busy 完成(BCI)
HARDWARE_LOCKED_WRITE_INTERRUPT	12	RO	0x0	硬件锁存写错误(HLE)
FIFO_UNDER_OVER_RUN_INTERRUPT	11	RO	0x0	FIFO 上/下溢错误(FRUN)
HOST_TIMEOUT_INTERRUPT	10	RO	0x0	数据饥饿超时(HTO)/Volt_switch
DATA_READ_TIMEOUT_INTERRUPT	9	RO	0x0	数据读超时(DRTO)
RESPONSE_TIMEOUT_INTERRUPT	8	RO	0x0	响应超时 (RTO).
DATA_CRC_ERROR_INTERRUPT	7	RO	0x0	数据 CRC 错误(DCRC).
RESPONSE_CRC_ERROR_INTERRUPT	6	RO	0x0	响应 CRC 错误(RCRC)

RECEIVE_FIFO_DATA_REQUEST_INTERRUPT	5	RO	0x0	RX FIFO 数据请求(RXDR)
TRANSMIT_RECEIVE_FIFO_DATA_INTERRUP	4	RO	0x0	TX FIFO 数据请求(TXDR)
DATA_TRANSFER_OVER_INTERRUPT	3	RO	0x0	数据传输完成 (DTO)
COMMAND_DONE_INTERRUPT	2	RO	0x0	命令完成(CD)
RESPONSE_ERROR_INTERRUPT	1	RO	0x0	响应错误(RE)
CARD_DETECT_INTERRUPT	0	RO	0x0	卡检测(CD)

10.4.17 raw_ints(0x44)

域	位	读写	复位值	描述
SDIO_INTERRUPT_CARD0	31:16	RW	0x0	SDIO card 中断
END_BIT_ERROR_STATUS	15	RW	0x0	读写 End-bit 错误/写未收到 CRC
AUTO_COMMAND_DONE_STATUS	14	RW	0x0	自动命令完成 (ACD)
BUSY_COMPLETE_STATUS	13	RW	0x0	起始位错误(SBE)/Busy 完成(BCI)
HARDWARE_LOCKED_WRITE_STATUS	12	RW	0x0	硬件锁存写错误(HLE)
FIFO_UNDER_OVER_RUN_STATUS	11	RW	0x0	FIFO 上/下溢错误(FRUN)
HOST_TIMEOUT_STATUS	10	RW	0x0	数据饥饿超时(HTO)/Volt_switch
DATA_READ_TIMEOUT_STATUS	9	RW	0x0	数据读超时(DRTO)
RESPONSE_TIMEOUT_STATUS	8	RW	0x0	响应超时 (RTO)
DATA_CRC_ERROR_STATUS	7	RW	0x0	数据 CRC 错误(DCRC)
RESPONSE_CRC_ERROR_STATUS	6	RW	0x0	响应 CRC 错误(RCRC)
RECEIVE_FIFO_DATA_REQUEST_INTERRUPT	5	RW	0x0	RX FIFO 数据请求(RXDR) NON-DMA 使用
TRANSMIT_RECEIVE_FIFO_DATA_STATUS	4	RW	0x0	TX FIFO 数据请求(TXDR) NON-DMA 使用
DATA_TRANSFER_OVER_STATUS	3	RW	0x0	数据传输完成 (DTO)
COMMAND_DONE_STATUS	2	RW	0x0	命令完成(CD)
RESPONSE_ERROR_STATUS	1	RW	0x0	响应错误(RE)
CARD_DETECT_STATUS	0	RW	0x0	卡检测(CD)

10.4.18 status(0x48)

域	位	读写	复位值	描述
reserved	31:30	RO	0x0	保留
FIFO_COUNT	29:17	RO	0x0	FIFO 填充计数器
RESPONSE_INDEX	16:11	RO	0x0	响应索引

DATA_STATE_MC_BUSY	10	RO	0x0	DATA TX|RX FSM busy 0：not busy；1：busy
DATA_BUSY	9	RO	0x0	卡 busy 0：not busy；1：busy
DATA_3_STATUS	8	RO	0x0	DATA[3] 卡在位检测 0：不在位；1：在位
COMMAND_FSM_STATES	7:4	RO	0x0	cmd FSM
FIFO_FULL	3	RO	0x0	FIFO full
FIFO_EMPTY	2	RO	0x0	FIFO empty
FIFO_TX_WATERMARK	1	RO	0x0	达到 FIFO_TX 标记
FIFO_RX_WATERMARK	0	RO	0x0	达到 FIFO_RX 标记

10.4.19 fifoth(0x4C)

域	位	读写	复位值	描述
reserved	30:28	RW	0x0	保留
RX_WMark	27:16	RW	0x1ff	FIFO threshold
TX_WMark	11:0	RW	0x0	FIFO threshold

10.4.20 card_detect_biu(0x50)

域	位	读写	复位值	描述
CARD0_DETECT_N	31:0	RO	0x0	1：卡不在位；0：卡在位

10.4.21 card_write_prt_biu(0x54)

域	位	读写	复位值	描述
WRITE_PROTECT_0	31:0	RO	0x0	1：写保护；0：无写保护

10.4.22 gpio(0x58)

域	位	读写	复位值	描述
CCLK_RDY	31:0	RO	0x0	CIU 时钟 ready

10.4.23 tran_crd_cnt_mx(0x5C)

域	位	读写	复位值	描述
TRANS_CARD_BYTE_COUNT	31:0	RO	0x0	CIU 到卡传输的字节数

10.4.24 tran_fifo(0x60)

域	位	读写	复位值	描述
TRANS_FIFO_BYTE_COUNT	31:0	RO	0x0	MEM&FIFO 之间传输的字节数

10.4.25 debnce(0x64)

域	位	读写	复位值	描述
DEBOUNCE_COUNT	31:0	RW	0xFFFFFF	去抖时钟数，参考值 5-25 ms

10.4.26 uid(0x68)

域	位	读写	复位值	描述
UID	31:0	RW	0x59595959	用户 ID

10.4.27 vid(0x6C)

域	位	读写	复位值	描述
VID	31:0	RO	0x6488280A	控制器版本

10.4.28 hcon(0x70)

域	位	读写	复位值	描述
res	31:0	RW	0x00010001	保留

10.4.29 uhs_reg(0x74)

域	位	读写	复位值	描述
DDR_REG_0	31:16	RW	0x0	DDR 模式。
VOLT_REG_0	15:0	RW	0x0	外部调压器接口电压 0：2.5V Vdd；1：1.8V Vdd

10.4.30 card_reset(0x78)

域	位	读写	复位值	描述
CARD0_RESET	31:0	RW	0x0	1：运行；0：复位

10.4.31 status_reg(0x90)

域	位	读写	复位值	描述
reserved	31:13	RO	0x0	保留
EB	12:10	RO	0x0	异常标识 3’b001：发送异常 3’b010：接收异常 bitIDSTS[2]置 1，不产生该中断。
AIS	9	RW	0x0	异常中断汇总 IDSTS[2]：总线错误 Interrupt IDSTS[4]：DU 中断
NIS	8:6	RW	0x0	正常中断汇总 IDSTS[0]：发送 IDSTS[1]：接收
CES	5	RW	0x0	卡错误汇总 EBE：结束位错误 RTO：响应超时/Boot Ack 超时 RCRC：响应 CRC 错误 SBE：起始位错误

				DRTO：数据读超时/BDS 超时 DCRC：数据 CRC 错误 RE：响应错误
DU	4:3	RW	0x0	描述符不可读中断
FBE	2	RW	0x0	1：清除 IDSTS[12:10]中断。
RI	1	RW	0x0	接收完成中断，针对描述符。
TI	0	RW	0x0	发送完成中断，针对描述符。

10.4.32 intr_en_reg(0x94)

域	位	读写	复位值	描述
reserved	31:10	RW	0x0	保留
AISE	9	RW	0x0	异常中断使能
NIE	8	RW	0x0	正常中断使能
CESE	5	RW	0x0	卡错误中断使能
DUE	4	RW	0x0	描述符不可读中断使能
FBEE	2	RW	0x0	总线错误中断使能
RIE	1	RW	0x0	接收中断使能
TIE	0	RW	0x0	发送中断使能

10.4.33 cardthctl(0x100)

域	位	读写	复位值	描述
reserved	31:29	RO	0x0	保留
CARDRDTHRESHOLD	28:16	RW	0x0	读卡 Threshold ： N=27：FIFO_DEPTH is 128 N=26：FIFO_DEPTH is 64 N=25：FIFO_DEPTH is 32 N=24：FIFO_DEPTH is 16 N=23：FIFO_DEPTH is 8
CARDWRTHREN	2	RO	0x0	写卡 Threshold 使能
BUSY_CLR_INT_EN	1	RW	0x0	Busy 清中断
CARDRDTHREN	0	RW	0x0	卡读 Threshold 使能

10.4.34 uhs_reg_ext(0x108)

域	位	读写	复位值	描述
EXT_CLK_MUX_CTRL	31	RW	0x0	保留
CLK_DRV_PHASE_CTRL	30:24	RW	0x0	输出相位参数，相对于 CIU 时钟相位点。
CLK_SMPL_PHASE_CTRL	22:16	RW	0x0	采样相位参数，相对于 CIU 时钟相位点。
CLK_DIV_CTRL	14:8	RW	0x5	分频参数,CIU f= CLK_DIV_CTRL +1，MIN=1
EXT_CLK_ENABLE	1	RW	0x0	外部时钟-CIU 时钟源使能。
MMC_VOLT_REG_0	0	RW	0x0	1.2V 供电选择

10.4.35 emmc_ddr_reg(0x10C)

域	位	读写	复位值	描述
EMMC_DDR_REG	31:0	RW	0x0	0：start bit 一个周期 1：start bit 小于一个周期

10.4.36 enable_shift(0x110)

域	位	读写	复位值	描述
reserved	31:2	RW	0x0	保留
ENABLE_SHIFT	1:0	RW	0x0	00：默认相位 01：移位使能到下一个上升沿 10：移位使能到下一个下降沿

10.4.37 data(0x200)

域	位	读写	复位值	描述
DATA	31:0	RW	0x0	数据 FIFO 寄存器

11 NAND Flash Controller

11.1 操作说明

11.1.1 控制器初始化流程

- 配置 timing mode，异步模式配置 Nf_timing0_reg~Nf_timing5_reg，同步模式配置 Nf_timing6_reg~Nf_timing14_reg，Toggle 模式配置 Nf_timing15~Nf_timing18_reg。具体参数值的选择根据 flash 设备支持的 timing mode。
- 如需配置命令、地址、数据之间的时间间隔，操作寄存器 Nf_interval_reg。
- 如需配置请求之间的时间间隔，操作寄存器 Nf_cmdintval_reg。
- 如需配置 FIFO 超时时间，操作寄存器 Nf_fiftimout_reg。
- 配置 FIFO 的满阈值和空阈值，操作 Nf_fiflevel0_reg 和 Nf_fiflevel1_reg。
- 如需打开相应中断，配置 Nf_intrmask_reg 寄存器使能相应中断。
- onfi 同步、toggle 模式下读数据是通过高频时钟（2GHz）采样得到，采样相位调节，操作 Nf_ctrl1_reg。
- 配置 Nf_ctrl0_reg 寄存器，包括 spare size 大小，spare size 使能，ECC 纠错位数，ECC 使能，接口模式和接口位宽，并使能 nandflash 控制器。

11.1.2 发送请求流程

- 首先将描述符表和数据填入内存中。
- 将描述表地址的低 32 位写入 Nf_maddr0_reg 寄存器。
- 将描述表地址的高 8 位写入 Nf_maddr1_reg 寄存器，并使能 dma 请求。
- 如果没有使能中断，读 Nf_state_reg 寄存器判断命令发送完成。如果使能了中断，检测到中断后，读 Nf_intr_reg 寄存器获得中断类型。
- 如果使能了 ECC，以 512B 为单位进行检验，读 Nf_encodefincnt_reg 判断编码完成，读 Nf_decodefincnt_reg 判断解码完成，然后读 Nf_errlocation1_reg~Nf_errlocation64_reg 判断错误位置。

11.1.3 错误相关操作

读 Nf_state_reg 寄存器发现产生错误后，写 Nf_errclr_reg 寄存器清除相应错误。

11.1.4 FIFO 清空操作

读 Nf_fiforsta_reg 寄存器判断 FIFO 中有数据，需要写 Nf_fifree_reg 寄存器，清空 FIFO，才能继续进行下一操作。

11.1.5 调试相关操作

读 Nf_debug_reg 寄存器获取控制器状态机信息。读 dma、nand 接口相关以及 ECC 剩余数据寄存器（偏移地址在 12'h090~12'h0b0）获得数据信息。

11.1.6 写保护操作

配置 Nf_wp_reg 寄存器，控制写保护信号。

11.2 寄存器列表

寄存器名称	偏移	描述
Nf_ctrl0_reg	0x000	控制寄存器 0
Nf_ctrl1_reg	0x004	控制寄存器 1
Nf_maddr0_reg	0x008	内存中存储的描述符表首地址的低 32 位
Nf_maddr1_reg	0x00C	内存中存储的描述符表首地址的高 8 位和 DMA 使能
Nf_timing0_reg	0x010	Timing0 寄存器
Nf_timing1_reg	0x014	Timing1 寄存器
Nf_timing2_reg	0x018	Timing2 寄存器
Nf_timing3_reg	0x01C	Timing3 寄存器
Nf_timing4_reg	0x020	Timing4 寄存器
Nf_timing5_reg	0x024	Timing5 寄存器
Nf_timing6_reg	0x028	Timing6 寄存器
Nf_timing7_reg	0x02C	Timing7 寄存器
Nf_timing8_reg	0x030	Timing8 寄存器
Nf_timing9_reg	0x034	Timing9 寄存器
Nf_timing10_reg	0x038	Timing10 寄存器
Nf_timing11_reg	0x03C	Timing11 寄存器
Nf_timing12_reg	0x040	Timing12 寄存器
Nf_timing13_reg	0x044	Timing13 寄存器
Nf_timing14_reg	0x048	Timing14 寄存器
Nf_timing15_reg	0x04C	Timing15 寄存器
Nf_timing16_reg	0x050	Timing16 寄存器
Nf_timing17_reg	0x054	Timing17 寄存器
Nf_timing18_reg	0x058	Timing18 寄存器
Nf_fiforsta_reg	0x05C	FIFO 状态寄存器
Nf_interval_reg	0x060	命令、地址、数据之间的间隔时间配置寄存

		器
Nf_cmdintval_reg	0x064	请求之间的间隔时间配置寄存器
Nf_fiftimout_reg	0x068	FIFO 超时计数
Nf_fiflevel0_reg	0x06C	FIFO 阈值选择
Nf_fiflevel1_reg	0x070	FIFO 阈值选择
Nf_wp_reg	0x074	WP 使能寄存器
Nf_fifree_reg	0x078	FIFO 清空寄存器
Nf_state_reg	0x07C	状态寄存器
Nf_intrmask_reg	0x080	中断屏蔽位寄存器
Nf_intr_reg	0x084	中断状态寄存器
Nf_debug_reg	0x088	debug 寄存器
Nf_errclr_reg	0x08C	错误清除寄存器
Nf_dmardcnt_reg	0x090	DMA 读页操作当前的剩余数据
Nf_dmardsparcnt_reg	0x094	DMA 读 spar 操作当前的剩余数据
Nf_dmawrcnt_reg	0x098	DMA 写页操作当前的剩余数据
Nf_dmawrsparcnt_reg	0x09C	DMA 写 spar 操作当前的剩余数据
Nf_intwrcnt_reg	0x0A0	NAND 接口写操作当前剩余数据
Nf_intasyrdcnt_reg	0x0A4	NAND 接口异步读操作当前剩余数据
Nf_intsyntogrdcnt_reg	0x0A8	NAND 接口同步写或 tog 读操作当前剩余数据
Nf_encodefincnt_reg	0x0AC	一次请求中的 ECC 编码完成次数计数器
Nf_encodedatcnt_reg	0x0B0	ECC 编码发送计数器
Nf_decodefincnt_reg	0x0B4	一次请求中的 ECC 校验完成计数器
Nf_errlocation1_reg	0x0B8	错误定位 1 寄存器
Nf_errlocation2_reg	0x0BC	错误定位 2 寄存器
Nf_errlocation3_reg	0x0C0	错误定位 3 寄存器
Nf_errlocation4_reg	0x0C4	错误定位 4 寄存器
Nf_errlocation5_reg	0x0C8	错误定位 5 寄存器
Nf_errlocation6_reg	0x0CC	错误定位 6 寄存器
Nf_errlocation7_reg	0x0D0	错误定位 7 寄存器
Nf_errlocation8_reg	0x0D4	错误定位 8 寄存器
Nf_errlocation9_reg	0x0D8	错误定位 9 寄存器
Nf_errlocation10_reg	0x0DC	错误定位 10 寄存器
Nf_errlocation11_reg	0x0E0	错误定位 11 寄存器
Nf_errlocation12_reg	0x0E4	错误定位 12 寄存器
Nf_errlocation13_reg	0x0E8	错误定位 13 寄存器
Nf_errlocation14_reg	0x0EC	错误定位 14 寄存器
Nf_errlocation15_reg	0x0F0	错误定位 15 寄存器
Nf_errlocation16_reg	0x0F4	错误定位 16 寄存器
Nf_errlocation17_reg	0x0F8	错误定位 17 寄存器
Nf_errlocation18_reg	0x0FC	错误定位 18 寄存器

Nf_errlocation19_reg	0x100	错误定位 19 寄存器
Nf_errlocation20_reg	0x104	错误定位 20 寄存器
Nf_errlocation21_reg	0x108	错误定位 21 寄存器
Nf_errlocation22_reg	0x10C	错误定位 22 寄存器
Nf_errlocation23_reg	0x110	错误定位 23 寄存器
Nf_errlocation24_reg	0x114	错误定位 24 寄存器
Nf_errlocation25_reg	0x118	错误定位 25 寄存器
Nf_errlocation26_reg	0x11C	错误定位 26 寄存器
Nf_errlocation27_reg	0x120	错误定位 27 寄存器
Nf_errlocation28_reg	0x124	错误定位 28 寄存器
Nf_errlocation29_reg	0x128	错误定位 29 寄存器
Nf_errlocation30_reg	0x12C	错误定位 30 寄存器
Nf_errlocation31_reg	0x130	错误定位 31 寄存器
Nf_errlocation32_reg	0x134	错误定位 32 寄存器
Nf_errlocation33_reg	0x138	错误定位 33 寄存器
Nf_errlocation34_reg	0x13C	错误定位 34 寄存器
Nf_errlocation35_reg	0x140	错误定位 35 寄存器
Nf_errlocation36_reg	0x144	错误定位 36 寄存器
Nf_errlocation37_reg	0x148	错误定位 37 寄存器
Nf_errlocation38_reg	0x14C	错误定位 38 寄存器
Nf_errlocation39_reg	0x150	错误定位 39 寄存器
Nf_errlocation40_reg	0x154	错误定位 40 寄存器
Nf_errlocation41_reg	0x158	错误定位 41 寄存器
Nf_errlocation42_reg	0x15C	错误定位 42 寄存器
Nf_errlocation43_reg	0x160	错误定位 43 寄存器
Nf_errlocation44_reg	0x164	错误定位 44 寄存器
Nf_errlocation45_reg	0x168	错误定位 45 寄存器
Nf_errlocation46_reg	0x16C	错误定位 46 寄存器
Nf_errlocation47_reg	0x170	错误定位 47 寄存器
Nf_errlocation48_reg	0x174	错误定位 48 寄存器
Nf_errlocation49_reg	0x178	错误定位 49 寄存器
Nf_errlocation50_reg	0x17C	错误定位 50 寄存器
Nf_errlocation51_reg	0x180	错误定位 51 寄存器
Nf_errlocation52_reg	0x184	错误定位 52 寄存器
Nf_errlocation53_reg	0x188	错误定位 53 寄存器
Nf_errlocation54_reg	0x18C	错误定位 54 寄存器
Nf_errlocation55_reg	0x190	错误定位 55 寄存器
Nf_errlocation56_reg	0x194	错误定位 56 寄存器
Nf_errlocation57_reg	0x198	错误定位 57 寄存器
Nf_errlocation58_reg	0x19C	错误定位 58 寄存器
Nf_errlocation59_reg	0x1A0	错误定位 59 寄存器

Nf_errlocation60_reg	0x1A4	错误定位 60 寄存器
Nf_errlocation61_reg	0x1A8	错误定位 61 寄存器
Nf_errlocation62_reg	0x1AC	错误定位 62 寄存器
Nf_errlocation63_reg	0x1B0	错误定位 63 寄存器
Nf_errlocation64_reg	0x1B4	错误定位 65 寄存器
Software_reg0	0x1C8	软件配置寄存器 0
Software_reg1	0x1CC	软件配置寄存器 1
PIDR4	0xFD0	外设识别寄存器 4
PIDR5	0xFD4	外设识别寄存器 5
PIDR6	0xFD8	外设识别寄存器 6
PIDR7	0xFDC	外设识别寄存器 7
PIDR0	0xFE0	外设识别寄存器 0
PIDR1	0xFE4	外设识别寄存器 1
PIDR2	0xFE8	外设识别寄存器 2
PIDR3	0xFEc	外设识别寄存器 3
CIDR0	0xFF0	组件识别寄存器 0
CIDR1	0xFF4	组件识别寄存器 1
CIDR2	0xFF8	组件识别寄存器 2
CIDR3	0xFFC	组件识别寄存器 3

11.3 寄存器说明

11.3.1 Nf_ctrl0_reg(0x000)

域	位	读写	复位值	描述
nf_spare_size	31:9	RW	23'h0	spare_size 配置。
nf_spare_size_en	8	RW	1'h0	spare_size 使能位
ecc_correct	7:5	RW	3'h0	ECC 纠错个数： 2：2 位；4：4 位
ecc_enable	4	RW	1'h0	ECC 使能位
nf_inter_mode	3:2	RW	2'h0	NAND Flash 接口模式： 2'b00：Asyn 2'b01：ONFI Syn 2'b10：Toggle Asyn
df_width	1	RW	1'h0	DQ 信号位宽设置，仅在 ONFI 异步模式下有效： 1’b0：8 位宽；1'b1：16 位宽。
nf_enable	0	RW	1'h0	NAND flash 控制器使能位，写 1 表示控制器打开

11.3.2 Nf_ctrl1_reg(0x004)

域	位	读写	复位值	描述
reserved	31:16	RW	16'h0	保留

sampl_phase	15:0	RW	16'h2	异步模式下相位调节，每增加 1，增加 1.6ns。onfi 同步、toggle 模式下接收数据采样相位调节周期，每增加 1，增加 0.5ns。该值不能为 0。

11.3.3 Nf_maddr0_reg(0x008)

域	位	读写	复位值	描述
dt_addr0	31:0	RW	32'h0	内存中存储的描述符表首地址的低 32 位

11.3.4 Nf_maddr1_reg(0x00C)

域	位	读写	复位值	描述
dma_wlens	31:24	RW	8'h40	设置 dma 写数据时的 lens 长度。
dma_rlens	23:16	RW	8'h40	设置 dma 读数据时的 lens 长度。
dma_empty	15:9	RW	7'h0	保留
dma_en	8	WO	1'h0	DMA 传输使能位，此位为 1 控制器开始进行 DMA 传输
dt_addr1	7:0	RW	8'h0	内存中存储的描述符表首地址的高 8 位

11.3.5 Nf_timing0_reg(0x010)

域	位	读写	复位值	描述
asy_timpara1	31:16	RW	16'h3	tCLS-tWP
asy_timpara0	15:0	RW	16'h3	tCS-tCLS

11.3.6 Nf_timing1_reg(0x014)

域	位	读写	复位值	描述
asy_timpara3	31:16	RW	16'h28	tWH
asy_timpara2	15:0	RW	16'h28	tWP

11.3.7 Nf_timing2_reg(0x018)

域	位	读写	复位值	描述
asy_timpara5	31:16	RW	16'h3	tCLH-tWH
asy_timpara4	15:0	RW	16'h3	tCH-tCLH

11.3.8 Nf_timing3_reg(0x01C)

域	位	读写	复位值	描述
asy_timpara7	31:16	RW	16'h6	tCH-tWH
asy_timpara6	15:0	RW	16'h6	tDQ_en，自定义

11.3.9 Nf_timing4_reg(0x020)

域	位	读写	复位值	描述
asy_timpara9	31:16	RW	16'h28	tREH
asy_timpara8	15:0	RW	16'h70	tWHR-(s10-s9)-(s3+s4+s5+h'1), >120ns,

				//tCCS>200ns

11.3.10 Nf_timing5_reg(0x024)

域	位	读写	复位值	描述
asy_timpara11	31:16	RW	16'h30	tADL-(s2+s3+s4+s5+s6-h'1), //tCCS>200ns
asy_timpara10	15:0	RW	16'h50	tRC

11.3.11 Nf_timing6_reg(0x028)

域	位	读写	复位值	描述
syn_timpara16	31:16	RW	16'h20	tCALS+tCH(建议值 tCK)
syn_timpara1	15:0	RW	16'h41	tCAD+tCS-s3-s5

11.3.12 Nf_timing7_reg(0x02C)

域	位	读写	复位值	描述
syn_timpara3	31:16	RW	16'h05	tDQ_en，自定义
syn_timpara2	15:0	RW	16'h20	tCK

11.3.13 Nf_timing8_reg(0x030)

域	位	读写	复位值	描述
syn_timpara5	31:16	RW	16'h10	0.5tCK
syn_timpara4	15:0	RW	16'h19	tCAD-tCK-s3-h'2

11.3.14 Nf_timing9_reg(0x034)

域	位	读写	复位值	描述
syn_timpara7	31:16	RW	16'h62	tCCS(n*tCK>500ns)-s3-s5-tCALS-h'2
syn_timpara6	15:0	RW	16'h40	tWHR>80ns

11.3.15 Nf_timing10_reg(0x038)

域	位	读写	复位值	描述
syn_timpara9	31:16	RW	16'h38	>1.5tCK ,(n+0.5)tCK+tCH, n>=1
syn_timpara8	15:0	RW	16'h20	tCK(0.75~1.25 倍)

11.3.16 Nf_timing11_reg(0x03C)

域	位	读写	复位值	描述
reserved	31:16	RW	16'h00	保留
syn_timpara11	15:0	RW	16'h09	tCK-tCALS

11.3.17 Nf_timing12_reg(0x040)

域	位	读写	复位值	描述
syn_timpara13	31:16	RW	16'h50	tCKWR =n*tCK -tCALS(n>=2)(建 议值 1.5tCK)
syn_timpara12	15:0	RW	16'h20	tWRCK>20ns (n*tCK)(建议值 tCK)

11.3.18 Nf_timing13_reg(0x044)

域	位	读写	复位值	描述
tog_timpara11	31:16	RW	16'h14	tWPST
tog_timpara10	15:0	RW	16'h0a	tWPRE

11.3.19 Nf_timing14_reg(0x048)

域	位	读写	复位值	描述
tog_timpara1	31:16	RW	16'h08	tCLS-tWP
tog_timpara0	15:0	RW	16'h08	tCS-tCLS

11.3.20 Nf_timing15_reg(0x04C)

域	位	读写	复位值	描述
tog_timpara3	31:16	RW	16'hc8	tWHR/tWHR2（120/300ns）
tog_timpara2	15:0	RW	16'hc8	tADL（300ns）

11.3.21 Nf_timing16_reg(0x050)

域	位	读写	复位值	描述
tog_timpara5	31:16	RW	16'h08	tCLH-tWH
tog_timpara4	15:0	RW	16'h08	tCH-tCLH

11.3.22 Nf_timing17_reg(0x054)

域	位	读写	复位值	描述
tog_timpara7	31:16	RW	16'h20	tRPST||tWPST（25ns+0.5tDSC）
tog_timpara6	15:0	RW	16'h0a	tRPRE||tWPRE

11.3.23 Nf_timing18_reg(0x058)

域	位	读写	复位值	描述
tog_timpara9	31:16	RW	16'h14	tRPSTH
tog_timpara8	15:0	RW	16'h08	0.5tDSC

11.3.24 Nf_fiforsta_reg(0x05C)

域	位	读写	复位值	描述
reserved	31:12	RO	20'h0	保留
fifo_full_real	11	RO	1'h0	FIFO 满数据
fifo_empty_real	10	RO	1'h1	FIFO 无数据
fifo_count	9:0	RO	10'h0	FIFO 当前的数据深度，一个深度是 4

11.3.25 Nf_interval_reg(0x060)

域	位	读写	复位值	描述
reserved	31:16	RO	16'h0	保留
interval_timpara	15:0	RW	16'h00	命令，地址，数据之间的时间间隔，写入值每增加 1，超时时间增加 2ns

11.3.26 Nf_cmdintval_reg(0x064)

域	位	读写	复位值	描述
cmd_interval_param	31:0	RW	32'h30	请求之间的时间间隔，写入值每增加 1，超时时间增加 2ns

11.3.27 Nf_fiftimout_reg(0x068)

域	位	读写	复位值	描述
fifo_timout_param	31:0	RW	32'h800	FIFO 超时计数器，写入值每增加 1，超时时间增加 2ns

11.3.28 Nf_fiflevel0_reg(0x06C)

域	位	读写	复位值	描述
reserved	31:4	RW	28'h0	保留
fiflevel0_sel	3:0	RW	4'h7	fifo 阈值选择： 4'b0000：FIFO 有数据 4'b0001：FIFO >= 1/8 full 4'b0010：FIFO >= 1/4 full 4'b0011：FIFO >= 3/8 full 4'b0100：FIFO >= 1/2 full 4'b0101：FIFO >= 5/8 full 4'b0110：FIFO >= 3/4 full 4'b0111：FIFO >= 7/8 full 4'b1000：FIFO 为满

11.3.29 Nf_fiflevel1_reg(0x070)

域	位	读写	复位值	描述
reserved	31:4	RW	28'h0	保留
fiflevel1_sel	3:0	RW	4'h1	fifo 阈值选择 4'b0000：FIFO 无数据 4'b0001：FIFO <= 1/8 empty 4'b0010：FIFO <= 1/4 empty 4'b0011：FIFO <= 3/8 empty 4'b0100：FIFO <= 1/2 empty 4'b0101：FIFO <= 5/8 empty 4'b0110：FIFO <= 3/4 empty 4'b0111：FIFO <= 7/8 empty

11.3.30 Nf_wp_reg(0x074)

域	位	读写	复位值	描述
reserved	31:1	RW	31'h0	保留
nf_wp	0	RW	1'h1	NAND Falsh 接口 WP 使能位，0 表示写保护打开

11.3.31 Nf_fifree_reg(0x078)

域	位	读写	复位值	描述
reserved	31:1	WO	31'h0	保留
fifo_free	0	WO	1'h0	清空 FIFO 操作，高有效。

11.3.32 Nf_state_reg(0x07C)

域	位	读写	复位值	描述
reserved	31:24	RO	8'h0	保留
axi_wr_err_sta	23	RO	1'h0	axi 写错误
axi_rd_err_sta	22	RO	1'h0	axi 读错误
axi_dsp_err_sta	21	RO	1'h0	描述符错误
ecc_errover_sta	20	RO	1'h0	错误超过可校验范围
ecc_error_sta	19	RO	1'h0	ECC 校验有错
ecc_right_sta	18	RO	1'h1	ECC 校验正确
ecc_finish_sta	17	RO	1'h0	ECC 校验完成
ecc_busy_sta	16	RO	1'h0	ECC 校验忙
rb_sta	15	RO	1'h1	RB_N 接口的状态
dqs_gate_sta	14	RO	1'h0	dqs 门控状态
re_gate_sta	13	RO	1'h0	re_n 门控状态
page_finish_sta	12	RO	1'h0	nand 接口数据操作完成（有 next_cmd 时，所有命令完成后，才显示）
cmd_finish_sta	11	RO	1'h0	nand 接口命令操作完成（有 next_cmd 时，所有命令完成后，才显示）
cs_sta	10:7	RO	4'hf	片选状态
fifo_timeout_sta	6	RO	1'h0	fifo 超时
fifo_full_sta	5	RO	1'h0	fifo 满
fifo_empty_sta	4	RO	1'h1	fifo 空
dma_finish_sta	3	RO	1'h0	dma 完成
dma_pgfinish_sta	2	RO	1'h0	dma 数据操作完成（有 next_cmd 时，所有命令完成后，才显示）
dma_busy_sta	1	RO	1'h0	dma 控制器忙
nf_busy_sta	0	RO	1'h0	nandflash 控制器忙

11.3.33 Nf_intrmask_reg(0x080)

域	位	读写	复位值	描述
reserved	31:14	RW	18'h0	保留
ecc_right_mask	13	RW	1'h1	错误信号中断屏蔽位
ecc_finish_mask	12	RW	1'h1	ecc 完成中断屏蔽位
rb_state_mask	11	RW	1'h1	rb_n 信号 busy 中断屏蔽位

dqs_gate_mask	10	RW	1'h1	dqs 门控打开中断屏蔽位
re_gate_mask	9	RW	1'h1	re_n 门控打开中断屏蔽位
page_finish_mask	8	RW	1'h1	nand 接口页操作完成中断屏蔽位
cmd_finish_mask	7	RW	1'h1	nand 接口命令完成中断屏蔽位
fifo_timeout_mask	6	RW	1'h1	fifo 超时中断屏蔽位
fifo_full_mask	5	RW	1'h1	fifo 为满中断屏蔽位
fifo_empty_mask	4	RW	1'h1	fifo 为空中断屏蔽位
dma_finish_mask	3	RW	1'h1	dma 操作完成中断完成中断屏蔽位
dma_pgfinish_mask	2	RW	1'h1	dma 页操作完成中断屏蔽位
dma_busy_mask	1	RW	1'h1	dma 控制器忙状态中断屏蔽位
nf_busy_mask	0	RW	1'h1	nandflash 控制器忙状态中断屏蔽位

11.3.34 Nf_intr_reg(0x084)

域	位	读写	复位值	描述
reserved	31:14	WO	18'h0	保留
ecc_right_intr	13	WO	1'h0	错误中断状态位（ecc_err,ecc_over,dsp_err）,写 1 清除中断
ecc_finish_intr	12	WO	1'h0	ecc 完成中断状态位，写 1 清除中断
rb_state_intr	11	WO	1'h0	rb_n 信号 busy 中断状态位，写 1 清除中断
dqs_gate_intr	10	WO	1'h0	dqs 门控打开中断状态位，写 1 清除中断
re_gate_intr	9	WO	1'h0	re_n 门控打开中断状态位，写 1 清除中断
page_finish_intr	8	WO	1'h0	nand 接口页操作完成中断状态位，写 1 清除中断
cmd_finish_intr	7	WO	1'h0	nand 接口命令完成中断状态位，写 1 清除中断
fifo_timeout_intr	6	WO	1'h0	fifo 超时中断状态位，写 1 清除中断
fifo_full_intr	5	WO	1'h0	fifo 为满中断状态位，写 1 清除中断
fifo_empty_intr	4	WO	1'h0	fifo 为空中断状态位，写 1 清除中断
dma_finish_intr	3	WO	1'h0	dma 操作完成中断状态位，写 1 清除中断
dma_pgfinish_intr	2	WO	1'h0	dma 页操作完成中断状态位，写 1 清除中断
dma_busy_intr	1	WO	1'h0	dma 控制器忙状态中断状态位，写 1 清除中断
nf_busy_intr	0	WO	1'h0	nandflash 控制器忙状态中断状态位，写 1 清除中断

11.3.35 Nf_debug_reg(0x088)

域	位	读写	复位值	描述
reserved	31:12	RO	20'h0	保留
decoder_fsm	11:8	RO	4'h0	校验的当前状态
main_fsm	7:4	RO	4'h0	指令传输的当前状态
dam_fsm	3:0	RO	4'h0	DMA 传输的当前状态

11.3.36 Nf_errclr_reg(0x08C)

域	位	读写	复位值	描述
reserved	31:4	WO	28'h0	保留
ecc_error_clr	3	WO	1'h0	ecc 错误清除
axi_wr_err_clr	2	WO	1'h0	axi 写错误清除
axi_rd_err_clr	1	WO	1'h0	axi 读错误清除
dsp_err_clr	0	WO	1'h0	描述符错误清除

11.3.37 Nf_dmardcnt_reg(0x090)

域	位	读写	复位值	描述
dma_rd_cnt	31:0	RO	32'h0	dma 读页操作当前的剩余数据

11.3.38 Nf_dmardsparcnt_reg(0x094)

域	位	读写	复位值	描述
dma_rd_spar_cnt	31:0	RO	32'h0	dma 读 spar 操作当前的剩余数据

11.3.39 Nf_dmawrcnt_reg(0x098)

域	位	读写	复位值	描述
dma_wr_cnt	31:0	RO	32'h0	dma 写页操作当前的剩余数据

11.3.40 Nf_dmawrsparcnt_reg(0x09C)

域	位	读写	复位值	描述
dma_wr_spar_cnt	31:0	RO	32'h0	dma 写 spar 操作当前的剩余数据

11.3.41 Nf_intwrcnt_reg(0x0A0)

域	位	读写	复位值	描述
int_wr_data_cnt	31:0	RO	32'h0	nand 接口写操作当前剩余数据

11.3.42 Nf_intwrcnt_reg(0x0A4)

域	位	读写	复位值	描述
int_asy_rd_data_cnt	31:0	RO	32'h0	nand 接口异步读操作当前剩余数据

11.3.43 Nf_intwrcnt_reg(0x0A8)

域	位	读写	复位值	描述
int_syn_tog_rd_data_cnt	31:0	RO	32'h0	nand 接口同步写或 tog 读操作当

				前剩余数据

11.3.44 Nf_encodefincnt_reg(0x0AC)

域	位	读写	复位值	描述
reserved	31:8	RO	24'h0	保留
int_encode_finish_count	7:0	RO	8'h0	一次请求中的ECC编码完成次数计数器

11.3.45 Nf_encodedatcnt_reg(0x0B0)

域	位	读写	复位值	描述
reserved	31:8	RO	24'h0	保留
int_encode_data_cnt	7:0	RO	8'h0	ECC 编码发送计数器

11.3.46 Nf_decodefincnt_reg(0x0B4)

域	位	读写	复位值	描述
reserved	31:17	RO	15'h0	保留
int_decoder_finish_cnt	16:0	RO	17'h0	一次请求中的ECC校验完成计数器

11.3.47 Nf_errlocation1_reg(0x0B8)

域	位	读写	复位值	描述
int_err_location1	31:0	RO	32'h0	第 1 个 512B 字节的错误地址： [31:16]：第 2 个错误地址 [15:0]：第 1 个错误地址 注：为 0 表示没有错误。

11.3.48 Nf_errlocation2_reg(0x0BC)

域	位	读写	复位值	描述
int_err_location2	31:0	RO	32'h0	第 1 个 512B 字节的错误地址： [31:16]：第 4 个错误地址 [15:0]：第 3 个错误地址 注：纠错能力为 4 的时候此寄存器有效。

11.3.49 Nf_errlocation3_reg(0x0C0)

域	位	读写	复位值	描述
int_err_location3	31:0	RO	32'h0	第 2 个 512B 字节的错误地址： [31:16]：第 2 个错误地址 [15:0]：第 1 个错误地址 注：为 0 表示没有错误。

11.3.50 Nf_errlocation4_reg(0x0C4)

域	位	读写	复位值	描述

int_err_location4	31:0	RO	32'h0	第 2 个 512B 字节的错误地址： [31:16]：第 4 个错误地址 [15:0]：第 3 个错误地址 注：纠错能力为 4 的时候此寄存器有效。

11.3.51 Nf_errlocation4_reg(0x0C8)

域	位	读写	复位值	描述
int_err_location5	31:0	RO	32'h0	第 3 个 512B 字节的错误地址： [31:16]：第 2 个错误地址 [15:0]：第 1 个错误地址 注：为 0 表示没有错误。

11.3.52 Nf_errlocation6_reg(0x0CC)

域	位	读写	复位值	描述
int_err_location6	31:0	RO	32'h0	第 3 个 512B 字节的错误地址： [31:16]：第 4 个错误地址 [15:0]：第 3 个错误地址 注：纠错能力为 4 的时候此寄存器有效。

11.3.53 Nf_errlocation7_reg(0x0D0)

域	位	读写	复位值	描述
int_err_location7	31:0	RO	32'h0	第 4 个 512B 字节的错误地址： [31:16]：第 2 个错误地址 [15:0]：第 1 个错误地址 注：为 0 表示没有错误。

11.3.54 Nf_errlocation8_reg(0x0D8)

域	位	读写	复位值	描述
int_err_location8	31:0	RO	32'h0	第 4 个 512B 字节的错误地址： [31:16]：第 4 个错误地址 [15:0]：第 3 个错误地址 注：纠错能力为 4 的时候此寄存器有效。

11.3.55 Nf_errlocation10_reg(0x0DC)

域	位	读写	复位值	描述
int_err_location10	31:0	RO	32'h0	第 5 个 512B 字节的错误地址： [31:16]：第 4 个错误地址 [15:0]：第 3 个错误地址 注：纠错能力为 4 的时候此寄存器有效。

11.3.56 Nf_errlocation11_reg(0x0E0)

域	位	读写	复位值	描述
int_err_location11	31:0	RO	32'h0	第 6 个 512B 字节的错误地址： [31:16]：第 2 个错误地址 [15:0]：第 1 个错误地址 注：为 0 表示没有错误。

11.3.57 Nf_errlocation12_reg(0x0E4)

域	位	读写	复位值	描述
int_err_location12	31:0	RO	32'h0	第 6 个 512B 字节的错误地址： [31:16]：第 4 个错误地址 [15:0]：第 3 个错误地址 注：纠错能力为 4 的时候此寄存器有效。

11.3.58 Nf_errlocation13_reg(0x0E8)

域	位	读写	复位值	描述
int_err_location13	31:0	RO	32'h0	第 7 个 512B 字节的错误地址： [31:16]：第 2 个错误地址 [15:0]：第 1 个错误地址 注：为 0 表示没有错误。

11.3.59 Nf_errlocation14_reg(0x0EC)

域	位	读写	复位值	描述
int_err_location14	31:0	RO	32'h0	第 7 个 512B 字节的错误地址： [31:16]：第 4 个错误地址 [15:0：第 3 个错误地址 注：纠错能力为 4 的时候此寄存器有效。

11.3.60 Nf_errlocation15_reg(0x0F0)

域	位	读写	复位值	描述
int_err_location15	31:0	RO	32'h0	第 8 个 512B 字节的错误地址： [31:16]：第 2 个错误地址 [15:0]：第 1 个错误地址 注：为 0 表示没有错误。

11.3.61 Nf_errlocation16_reg(0x0F4)

域	位	读写	复位值	描述
int_err_location16	31:0	RO	32'h0	第 8 个 512B 字节的错误地址： [31:16]：第 4 个错误地址 [15:0]：第 3 个错误地址 注：纠错能力为 4 的时候此寄存器

				有效。

11.3.62 Nf_errlocation17_reg(0x0F8)

域	位	读写	复位值	描述
int_err_location17	31:0	RO	32'h0	第 9 个 512B 字节的错误地址： [31:16]：第 2 个错误地址 [15:0]：第 1 个错误地址 注：为 0 表示没有错误。

11.3.63 Nf_errlocation18_reg(0x0FC)

域	位	读写	复位值	描述
int_err_location18	31:0	RO	32'h0	第 9 个 512B 字节的错误地址： [31:16]：第 4 个错误地址 [15:0]：第 3 个错误地址 注：纠错能力为 4 的时候此寄存器有效。

11.3.64 Nf_errlocation19_reg(0x100)

域	位	读写	复位值	描述
int_err_location19	31:0	RO	32'h0	第 10 个 512B 字节的错误地址： [31:16]：第 2 个错误地址 [15:0]：第 1 个错误地址 注：为 0 表示没有错误。

11.3.65 Nf_errlocation20_reg(0x104)

域	位	读写	复位值	描述
int_err_location20	31:0	RO	32'h0	第 10 个 512B 字节的错误地址： [31:16]：第 4 个错误地址 [15:0]：第 3 个错误地址 注：纠错能力为 4 的时候此寄存器有效。

11.3.66 Nf_errlocation21_reg(0x108)

域	位	读写	复位值	描述
int_err_location21	31:0	RO	32'h0	第 11 个 512B 字节的错误地址： [31:16]：第 2 个错误地址 [15:0]：第 1 个错误地址 注：为 0 表示没有错误。

11.3.67 Nf_errlocation22_reg(0x10C)

域	位	读写	复位值	描述
int_err_location22	31:0	RO	32'h0	第 11 个 512B 字节的错误地址： [31:16]：第 4 个错误地址

				[15:0]：第 3 个错误地址 注：纠错能力为 4 的时候此寄存器有效。

11.3.68 Nf_errlocation23_reg(0x110)

域	位	读写	复位值	描述
int_err_location23	31:0	RO	32'h0	第 12 个 512B 字节的错误地址： [31:16]：第 2 个错误地址 [15:0]：第 1 个错误地址 注：为 0 表示没有错误。

11.3.69 Nf_errlocation24_reg(0x114)

域	位	读写	复位值	描述
int_err_location24	31:0	RO	32'h0	第 12 个 512B 字节的错误地址： [31:16]：第 4 个错误地址 [15:0]：第 3 个错误地址 注：纠错能力为 4 的时候此寄存器有效。

11.3.70 Nf_errlocation25_reg(0x118)

域	位	读写	复位值	描述
int_err_location25	31:0	RO	32'h0	第 13 个 512B 字节的错误地址： [31:16]：第 2 个错误地址 [15:0]：第 1 个错误地址 注：为 0 表示没有错误。

11.3.71 Nf_errlocation26_reg(0x11C)

域	位	读写	复位值	描述
int_err_location26	31:0	RO	32'h0	第 13 个 512B 字节的错误地址： [31:16]：第 4 个错误地址 [15:0]：第 3 个错误地址 注：纠错能力为 4 的时候此寄存器有效。

11.3.72 Nf_errlocation27_reg(0x120)

域	位	读写	复位值	描述
int_err_location27	31:0	RO	32'h0	第 14 个 512B 字节的错误地址： [31:16]：第 2 个错误地址 [15:0]：第 1 个错误地址 注：为 0 表示没有错误。

11.3.73 Nf_errlocation28_reg(0x124)

域	位	读写	复位值	描述

int_err_location28	31:0	RO	32'h0	第 14 个 512B 字节的错误地址： [31:16]：第 4 个错误地址 [15:0]：第 3 个错误地址 注：纠错能力为 4 的时候此寄存器有效。

11.3.74 Nf_errlocation29_reg(0x128)

域	位	读写	复位值	描述
int_err_location29	31:0	RO	32'h0	第 15 个 512B 字节的错误地址： [31:16]：第 2 个错误地址 [15:0]：第 1 个错误地址 注：为 0 表示没有错误。

11.3.75 Nf_errlocation30_reg(0x12C)

域	位	读写	复位值	描述
int_err_location30	31:0	RO	32'h0	第 15 个 512B 字节的错误地址： [31:16]：第 4 个错误地址 [15:0]：第 3 个错误地址 注：纠错能力为 4 的时候此寄存器有效。

11.3.76 Nf_errlocation31_reg(0x130)

域	位	读写	复位值	描述
int_err_location31	31:0	RO	32'h0	第 16 个 512B 字节的错误地址： [31:16]：第 2 个错误地址 [15:0]：第 1 个错误地址 注：为 0 表示没有错误。

11.3.77 Nf_errlocation32_reg(0x134)

域	位	读写	复位值	描述
int_err_location32	31:0	RO	32'h0	第 16 个 512B 字节的错误地址： [31:16]：第 4 个错误地址 [15:0]：第 3 个错误地址 注：纠错能力为 4 的时候此寄存器有效。

11.3.78 Nf_errlocation33_reg(0x138)

域	位	读写	复位值	描述
int_err_location33	31:0	RO	32'h0	第 17 个 512B 字节的错误地址： [31:16]：第 2 个错误地址 [15:0]：第 1 个错误地址 注：为 0 表示没有错误。

11.3.79 Nf_errlocation34_reg(0x13C)

域	位	读写	复位值	描述
int_err_location34	31:0	RO	32'h0	第 17 个 512B 字节的错误地址： [31:16]：第 4 个错误地址 [15:0]：第 3 个错误地址 注：纠错能力为 4 的时候此寄存器有效。

11.3.80 Nf_errlocation35_reg(0x140)

域	位	读写	复位值	描述
int_err_location35	31:0	RO	32'h0	第 18 个 512B 字节的错误地址： [31:16]：第 2 个错误地址 [15:0]：第 1 个错误地址 注：为 0 表示没有错误。

11.3.81 Nf_errlocation36_reg(0x144)

域	位	读写	复位值	描述
int_err_location36	31:0	RO	32'h0	第 18 个 512B 字节的错误地址： [31:16]：第 4 个错误地址 [15:0]：第 3 个错误地址 注：纠错能力为 4 的时候此寄存器有效。

11.3.82 Nf_errlocation37_reg(0x148)

域	位	读写	复位值	描述
int_err_location37	31:0	RO	32'h0	第 19 个 512B 字节的错误地址： [31:16]：第 2 个错误地址 [15:0]：第 1 个错误地址 注：为 0 表示没有错误。

11.3.83 Nf_errlocation38_reg(0x14C)

域	位	读写	复位值	描述
int_err_location38	31:0	RO	32'h0	第 19 个 512B 字节的错误地址： [31:16]：第 4 个错误地址 [15:0]：第 3 个错误地址 注：纠错能力为 4 的时候此寄存器有效。

11.3.84 Nf_errlocation39_reg(0x150)

域	位	读写	复位值	描述
int_err_location39	31:0	RO	32'h0	第 20 个 512B 字节的错误地址： [31:16]：第 2 个错误地址 [15:0]：第 1 个错误地址

				注：为 0 表示没有错误。

11.3.85 Nf_errlocation40_reg(0x154)

域	位	读写	复位值	描述
int_err_location40	31:0	RO	32'h0	第 20 个 512B 字节的错误地址： [31:16]：第 4 个错误地址 [15:0]：第 3 个错误地址 注：纠错能力为 4 的时候此寄存器有效。

11.3.86 Nf_errlocation41_reg(0x158)

域	位	读写	复位值	描述
int_err_location41	31:0	RO	32'h0	第 21 个 512B 字节的错误地址： [31:16]：第 2 个错误地址 [15:0]：第 1 个错误地址 注：为 0 表示没有错误。

11.3.87 Nf_errlocation42_reg(0x15C)

域	位	读写	复位值	描述
int_err_location42	31:0	RO	32'h0	第 21 个 512B 字节的错误地址： [31:16]：第 4 个错误地址 [15:0]：第 3 个错误地址 注：纠错能力为 4 的时候此寄存器有效。

11.3.88 Nf_errlocation43_reg(0x160)

域	位	读写	复位值	描述
int_err_location43	31:0	RO	32'h0	第 22 个 512B 字节的错误地址： [31:16]：第 2 个错误地址 [15:0]：第 1 个错误地址 注：为 0 表示没有错误。

11.3.89 Nf_errlocation44_reg(0x164)

域	位	读写	复位值	描述
int_err_location44	31:0	RO	32'h0	第 22 个 512B 字节的错误地址： [31:16]：第 4 个错误地址 [15:0]：第 3 个错误地址 注：纠错能力为 4 的时候此寄存器有效。

11.3.90 Nf_errlocation45_reg(0x168)

域	位	读写	复位值	描述
int_err_location45	31:0	RO	32'h0	第 23 个 512B 字节的错误地址：

				[31:16]：第 2 个错误地址 [15:0]：第 1 个错误地址 注：为 0 表示没有错误。

11.3.91 Nf_errlocation46_reg(0x16C)

域	位	读写	复位值	描述
int_err_location46	31:0	RO	32'h0	第 23 个 512B 字节的错误地址： [31:16]：第 4 个错误地址 [15:0]：第 3 个错误地址 注：纠错能力为 4 的时候此寄存器有效。

11.3.92 Nf_errlocation47_reg(0x170)

域	位	读写	复位值	描述
int_err_location47	31:0	RO	32'h0	第 24 个 512B 字节的错误地址： [31:16]：第 2 个错误地址 [15:0]：第 1 个错误地址 注：为 0 表示没有错误。

11.3.93 Nf_errlocation48_reg(0x174)

域	位	读写	复位值	描述
int_err_location48	31:0	RO	32'h0	第 24 个 512B 字节的错误地址： [31:16]：第 4 个错误地址 [15:0]：第 3 个错误地址 注：纠错能力为 4 的时候此寄存器有效。

11.3.94 Nf_errlocation49_reg(0x178)

域	位	读写	复位值	描述
int_err_location49	31:0	RO	32'h0	第 25 个 512B 字节的错误地址： [31:16]：第 2 个错误地址 [15:0]：第 1 个错误地址 注：为 0 表示没有错误。

11.3.95 Nf_errlocation50_reg(0x17C)

域	位	读写	复位值	描述
int_err_location50	31:0	RO	32'h0	第 25 个 512B 字节的错误地址： [31:16]：第 4 个错误地址 [15:0]：第 3 个错误地址 注：纠错能力为 4 的时候此寄存器有效。

11.3.96 Nf_errlocation51_reg(0x180)

域	位	读写	复位值	描述
int_err_location51	31:0	RO	32'h0	第 26 个 512B 字节的错误地址： [31:16]：第 2 个错误地址 [15:0]：第 1 个错误地址 注：为 0 表示没有错误。

11.3.97 Nf_errlocation52_reg(0x184)

域	位	读写	复位值	描述
int_err_location52	31:0	RO	32'h0	第 26 个 512B 字节的错误地址： [31:16]：第 4 个错误地址 [15:0]：第 3 个错误地址 注：纠错能力为 4 的时候此寄存器有效。

11.3.98 Nf_errlocation53_reg(0x188)

域	位	读写	复位值	描述
int_err_location53	31:0	RO	32'h0	第 27 个 512B 字节的错误地址： [31:16]：第 2 个错误地址 [15:0]：第 1 个错误地址 注：为 0 表示没有错误。

11.3.99 Nf_errlocation54_reg(0x18C)

域	位	读写	复位值	描述
int_err_location54	31:0	RO	32'h0	第 27 个 512B 字节的错误地址： [31:16]：第 4 个错误地址 [15:0]：第 3 个错误地址 注：纠错能力为 4 的时候此寄存器有效。

11.3.100Nf_errlocation55_reg(0x190)

域	位	读写	复位值	描述
int_err_location55	31:0	RO	32'h0	第 28 个 512B 字节的错误地址： [31:16]：第 2 个错误地址 [15:0]：第 1 个错误地址 注：为 0 表示没有错误。

11.3.101Nf_errlocation56_reg(0x194)

域	位	读写	复位值	描述
int_err_location56	31:0	RO	32'h0	第 28 个 512B 字节的错误地址： [31:16]：第 4 个错误地址 [15:0]：第 3 个错误地址 注：纠错能力为 4 的时候此寄存器

				有效。

11.3.102Nf_errlocation57_reg(0x198)

域	位	读写	复位值	描述
int_err_location57	31:0	RO	32'h0	第 29 个 512B 字节的错误地址： [31:16]：第 2 个错误地址 [15:0]：第 1 个错误地址 注：为 0 表示没有错误。

11.3.103Nf_errlocation58_reg(0x19C)

域	位	读写	复位值	描述
int_err_location58	31:0	RO	32'h0	第 29 个 512B 字节的错误地址： [31:16]：第 4 个错误地址 [15:0]：第 3 个错误地址 注：纠错能力为 4 的时候此寄存器有效。

11.3.104Nf_errlocation59_reg(0x1A0)

域	位	读写	复位值	描述
int_err_location59	31:0	RO	32'h0	第 30 个 512B 字节的错误地址： [31:16]：第 2 个错误地址 [15:0]：第 1 个错误地址 注：为 0 表示没有错误。

11.3.105Nf_errlocation60_reg(0x1A4)

域	位	读写	复位值	描述
int_err_location60	31:0	RO	32'h0	第 30 个 512B 字节的错误地址： [31:16]：第 4 个错误地址 [15:0]：第 3 个错误地址 注：纠错能力为 4 的时候此寄存器有效。

11.3.106Nf_errlocation61_reg(0x1A8)

域	位	读写	复位值	描述
int_err_location61	31:0	RO	32'h0	第 31 个 512B 字节的错误地址： [31:16]：第 2 个错误地址 [15:0]：第 1 个错误地址 注：为 0 表示没有错误。

11.3.107Nf_errlocation62_reg(0x1AC)

域	位	读写	复位值	描述
int_err_location62	31:0	RO	32'h0	第 31 个 512B 字节的错误地址： [31:16]：第 4 个错误地址

				[15:0]：第 3 个错误地址 注：纠错能力为 4 的时候此寄存器有效。

11.3.108Nf_errlocation63_reg(0x1B0)

域	位	读写	复位值	描述
int_err_location63	31:0	RO	32'h0	第 32 个 512B 字节的错误地址： [31:16]：第 2 个错误地址 [15:0]：第 1 个错误地址 注：为 0 表示没有错误。

11.3.109Nf_errlocation64_reg(0x1B4)

域	位	读写	复位值	描述
int_err_location64	31:0	RO	32'h0	第 32 个 512B 字节的错误地址： [31:16]：第 4 个错误地址 [15:0]：第 3 个错误地址 注：纠错能力为 4 的时候此寄存器有效。

11.3.110Software_reg0(0x1C8)

域	位	读写	复位值	描述
soft_reg0	31:0	RW	32'h0	软件预留寄存器，用于软件操作。

11.3.111Software_reg1(0x1CC)

域	位	读写	复位值	描述
soft_reg1	31:0	RW	32'h0	软件预留寄存器，用于软件操作。

11.3.112PIDR4(0xFD0)

域	位	读写	复位值	描述
reserved	31:8	RO	24'h0	保留
size	7:0	RO	8'h08	[3:0]：Size [7:4]：Designer[18:15]

11.3.113PIDR5(0xFD4)

域	位	读写	复位值	描述
reserved	31:8	RO	24'h0	保留
reserved	7:0	RO	8'h0	保留

11.3.114PIDR6(0xFD8)

域	位	读写	复位值	描述
reserved	31:8	RO	24'h0	保留
reserved	7:0	RO	8'h0	保留

11.3.115PIDR7(0xFDC)

域	位	读写	复位值	描述
reserved	31:8	RO	24'h0	保留
reserved	7:0	RO	8'h0	保留

11.3.116PIDR0(0xFE0)

域	位	读写	复位值	描述
reserved	31:8	RO	24'h0	保留
part_number	7:0	RO	8'h02	Part_number[7:0]

11.3.117PIDR1(0xFE4)

域	位	读写	复位值	描述
reserved	31:8	RO	24'h0	保留
designer	7:0	RO	8'h93	[3:0]：Part_number[11:8] [7:4]：Designer[3:0]

11.3.118PIDR2(0xFE8)

域	位	读写	复位值	描述
reserved	31:8	RO	24'h0	保留
revision	7:0	RO	8'h09	[2:0]：Designer[6:4] [3]：JEDEC [7:4]：Revision

11.3.119PIDR3(0xFEC)

域	位	读写	复位值	描述
reserved	31:8	RO	24'h0	保留
cmod_revand	7:0	RO	8'h0	[3:0]：Customer Modified [7:4]：Revand

11.3.120CIDR0(0xFF0)

域	位	读写	复位值	描述
reserved	31:8	RO	24'h0	保留
preamble0	7:0	RO	8'h0d	Preamble[7:0]

11.3.121CIDR1(0xFF4)

域	位	读写	复位值	描述
reserved	31:8	RO	24'h0	保留
preamble1	7:0	RO	8'ha0	[3:0]：Preamble[11:8] [7:4]：Component Class

11.3.122CIDR2(0xFF8)

域	位	读写	复位值	描述

reserved	31:8	RO	24'h0	保留
preamble2	7:0	RO	8'h05	Preamble[19:12]

11.3.123CIDR2(0xFF8)

域	位	读写	复位值	描述
reserved	31:8	RO	24'h0	保留
preamble3	7:0	RO	8'hb1	Preamble[27:20]

12 CAN

12.1 操作说明

12.1.1 传输初始化

- 配置 CAN_CTRL[0]为 0，传输不使能；
- 配置 CAN_CTRL[7]为 1，复位内部状态；
- 配置 CAN_CTRL 选择传输模式；
- 配置 CAN_ARB_RATE_CTRL 和 CAN_DAT_RATE_CTRL 寄存器，设置传输速率；
- 配置 CAN_ACC_ID0/1/2/3 寄存器和 CAN_ACC_ID0/1/2/3_MASK 寄存器，设置接收 ID；
- 如果需要发送数据，向 CAN_TX_FIFO 写入数据；
- 配置 CAN_INTR 寄存器，选择使能的中断类型；
- 配置 CAN_CTRL[0]为 1，传输使能。

12.1.2 协同工作

- CPU 或系统其他 Master 通过中断或读 CAN_FIFO_CNT 判断缓存空间状态；
- 通过访问 CAN_RX_FIFO 或 CAN_TX_FIFO 读写数据。

12.1.3 终止传输

- 如果 TX_FIFO 不为空，想要停止发送数据，需要设置 CAN_CTRL[7]为 1，设置之后检查 CAN_XFER_STS[8],如果值为 0，表示停止发送成功；
- 如果想要停止传输，需要设置 CAN_CTRL[0]为 0，等到当前一笔数据传输完成以后，CAN_XFER_STS[10]的值为 IDLE，表示传输停止。

12.2 寄存器列表

寄存器名称	偏移	描述
CAN_CTRL	0x000	全局控制寄存器
CAN_INTR	0x004	中断寄存器
CAN_ARB_RATE_CTRL	0x008	仲裁段速率控制寄存器
CAN_DAT_RATE_CTRL	0x00C	数据段速率控制寄存器
CAN_ACC_ID0	0x010	可接收识别符 0 寄存器

CAN_ACC_ID1	0x014	可接收识别符 1 寄存器
CAN_ACC_ID2	0x018	可接收识别符 2 寄存器
CAN_ACC_ID3	0x01C	可接收识别符 3 寄存器
CAN_ACC_ID0_MASK	0x020	可接收识别符 0 掩码寄存器
CAN_ACC_ID1_MASK	0x024	可接收识别符 1 掩码寄存器
CAN_ACC_ID2_MASK	0x028	可接收识别符 2 掩码寄存器
CAN_ACC_ID3_MASK	0x02C	可接收识别符 3 掩码寄存器
CAN_XFER_STS	0x030	传输状态寄存器
CAN_ERR_CNT	0x034	错误计数寄存器
CAN_FIFO_CNT	0x038	FIFO 计数寄存器
CAN_INTR1	0x044	中断 1 寄存器
CAN_TX_FIFO	0x100~0x1FF	发送 FIFO 寄存器
CAN_RX_FIFO	0x200~0x2FF	接收 FIFO 寄存器

12.3 寄存器说明

12.3.1 CAN_CTRL(0x000)

位	读写	复位值	描述
31:11	RO	0x0	保留
10	RW	0x0	过载帧是否发送 0：接收 FIFO 满时发送过载帧； 1：接收 FIFO 满时不发送过载帧
9:8	RO	0x0	保留
7	RW	0x0	软复位 0：不使能；1：使能
6:3	RO	0x0	保留
2	RW	0x0	可接收识别符掩码使能 0：不使能；1：使能
1	RW	0x0	收发请求 0：只接收；1：发送和接收
0	RW	0x0	传输使能 0：不使能；1：使能

12.3.2 CAN_INTR(0x004)

位	读写	复位值	描述
31:24	RO	0x0	保留
23	RW	0x0	错误中断清除 0：不清除；1：清除
22	RW	0x0	发送帧结束中断清除 0：不清除；1：清除
21	RW	0x0	接收帧结束中断清除 0：不清除；1：清除

20	RW	0x0	发送 FIFO 空中断清除 0：不清除；1：清除
19	RW	0x0	接收 FIFO 满中断清除 0：不清除；1：清除
18	RW	0x0	隐性错误中断清除 0：不清除；1：清除
17	RW	0x0	隐性警告中断清除 0：不清除；1：清除
16	RW	0x0	总线关闭中断清除 0：不清除；1：清除
15	RW	0x0	错误中断使能 0：不使能；1：使能
14	RW	0x0	发送帧结束中断使能 0：不使能；1：使能
13	RW	0x0	接收帧结束中断使能 0：不使能；1: 使能
12	RW	0x0	发送 FIFO 空中断使能 0：不使能；1：使能
11	RW	0x0	接收 FIFO 满中断使能 0：不使能；1：使能
10	RW	0x0	隐性错误中断使能 0：不使能；1：使能
9	RW	0x0	隐性警告中断使能 0：不使能；1：使能
8	RW	0x0	总线关闭中断使能 0：不使能；1：使能
7	RO	0x0	错误中断状态 0：不生效；1：生效
6	RO	0x0	发送帧结束中断状态 0：不生效；1：生效
5	RO	0x0	接收帧结束中断状态 0：不生效；1：生效
4	RO	0x0	发送 FIFO 空中断状态 0：不生效；1：生效
3	RO	0x0	接收 FIFO 空中断状态 0：不生效；1：生效
2	RO	0x0	隐性错误中断状态 0：不生效；1：生效
1	RO	0x0	隐性警告中断状态 0：不生效；1：生效
0	RO	0x0	总线关闭中断状态 0：不生效 1：生效

12.3.3 CAN_ARB_RATE_CTRL(0x008)

位	读写	复位值	描述
31:29	RW	0x0	保留
28:16	RW	0x0	前级分频器 0~8191 表示 1~8192
15:11	RO	0x0	保留位
10:8	RW	0x0	相位 2 段计数值 0~7 表示 1~8
7:5	RW	0x0	相位 1 段计数值 0~7 表示 1~8
4:2	RW	0x0	传播段计数值 0~7 表示 1~8
1:0	RW	0x0	同步跳转宽度 0~3 表示 1~4

12.3.4 CAN_DAT_RATE_CTRL(0x00C)

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

12.3.5 CAN_ACC_ID0(0x010)

位	读写	复位值	描述
31:1	RW	0x0	保留
0	RW	0x0	只有携带可接收识别符的帧才会被写入 FIFO

12.3.6 CAN_ACC_ID1(0x014)

位	读写	复位值	描述
31:1	RW	0x0	保留
0	RW	0x0	只有携带可接收识别符的帧才会被写入 FIFO

12.3.7 CAN_ACC_ID2(0x018)

位	读写	复位值	描述

31:1	RW	0x0	保留
0	RW	0x0	只有携带可接收识别符的帧才会被写入 FIFO

12.3.8 CAN_ACC_ID3(0x01C)

位	读写	复位值	描述
31:1	RW	0x0	保留
0	RW	0x0	只有携带可接收识别符的帧才会被写入 FIFO

12.3.9 CAN_ACC_ID0_MASK(0x020)

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

12.3.10 CAN_ACC_ID1_MASK(0x024)

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

12.3.11 CAN_ACC_ID2_MASK(0x028)

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

12.3.12 CAN_ACC_ID3_MASK(0x02C)

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

12.3.13 CAN_XFER_STS(0x030)

位	读写	复位值	描述
31:11	RO	0x0	保留位
10	RO	0x0	传输状态 0：闲；1：忙
9	RO	0x0	接收状态 0：未接收；1：接收
8	RO	0x0	发送状态 0：未发送；1：发送
7:3	RO	0x0	有限状态机当前状态编号

2:0	RO	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

12.3.14 CAN_ERR_CNT(0x034)

位	读写	复位值	功能描述
31:25	RO	0x0	保留
24:16	RO	0x0	发送错误计数器 0~256
15:9	RO	0x0	保留
8:0	RO	0x0	接收错误计数器 0~256

12.3.15 CAN_FIFO_CNT(0x038)

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

12.3.16 CAN_INTR1(0x044)

位	读写	复位值	描述
31:24	RO	0x0	保留
23	RW	0x0	发送 FIFO 空中断清除 0：不清除；1：清除
22	RW	0x0	发送 FIFO 空间大小为 1/4 时中断清除 0：不清除；1：清除
21	RW	0x0	发送 FIFO 空间大小为 1/2 时中断清除 0：不清除；1：清除
20	RW	0x0	发送 FIFO 空间大小为 3/4 时中断清除 0：不清除；1：清除
19	RW	0x0	接收 FIFO 满中断清除 0：不清除；1：清除
18	RW	0x0	接收 FIFO 空间大小为 3/4 时中断清除 0：不清除；1：清除
17	RW	0x0	接收 FIFO 空间大小为 1/2 时中断清除 0：不清除；1：清除
16	RW	0x0	接收 FIFO 空间大小为 1/4 时中断清除 0：不清除；1：清除
15	RW	0x0	发送 FIFO 空中断使能 0：不使能；1：使能

14	RW	0x0	发送 FIFO 空间大小为 1/4 时中断使能 0：不使能；1：使能
13	RW	0x0	发送 FIFO 空间大小为 1/2 时中断使能 0：不使能；1：使能
12	RW	0x0	发送 FIFO 空间大小为 3/4 时中断使能 0：不使能；1：使能
11	RW	0x0	接收 FIFO 满中断使能 0：不使能；1：使能
10	RW	0x0	接收 FIFO 空间大小为 3/4 时中断使能 0：不使能；1：使能
9	RW	0x0	接收 FIFO 空间大小为 1/2 时中断使能 0：不使能；1：使能
8	RW	0x0	接收 FIFO 空间大小为 1/4 时中断使能 0：不使能；1：使能
7	RO	0x0	发送 FIFO 空中断状态 0：不生效；1：生效
6	RO	0x0	发送 FIFO 空间大小为 1/4 时中断状态 0：不生效；1：生效
5	RO	0x0	发送 FIFO 空间大小为 1/2 时中断状态 0：不生效；1：生效
4	RO	0x0	发送 FIFO 空间大小为 3/4 时中断状态 0：不生效；1：生效
3	RO	0x0	接收 FIFO 满中断状态 0：不生效；1：生效
2	RO	0x0	接收 FIFO 空间大小为 3/4 时中断状态 0：不生效；1：生效
1	RO	0x0	接收 FIFO 空间大小为 1/2 时中断状态 0：不生效；1：生效
0	RO	0x0	接收 FIFO 空间大小为 1/4 时中断状态 0：不生效；1：生效

12.3.17 CAN_TX_FIFO(0x100~0x1FF)

位	读写	复位值	功能描述
31:0	WO	0x0	发送 FIFO 寄存器

12.3.18 CAN_RX_FIFO(0x200~0x2FF)

位	属性	复位值	功能描述
31:0	RW	0x0	接收 FIFO 寄存器

13 PWM

PWM 支持典型的 Timer 和 PWM 功能。

13.1 操作说明

PWM（pulse width modulation）支持输入 capture 和输出 compare 两种功能。

当配置为 capture 模式后，输入管脚出现上升沿或下降沿时，会出发本地的一个 counter 值的递增，同时产生一个脉冲信号置位状态寄存器。当计数到最大值 16‘hffff 时，计数器清零，并重新开始计数。capture 模式下，每接收到一个输入边沿（上升或下降），产生一个中断。

Compare 功能需要用到一个 16 位宽计数器，该计数器的工作时钟是输入时钟经过 1 至 4096 分频系数分频获得，计数器固定为 modulo 模式，计数器从 0 开始递增计数直到 TIM_PERIOD 最大值，当计数值等于 TIM_PERIOD 的时候，STAT 寄存器的 OVFIF 位置位，随后再次从 0 开始计数。

13.2 寄存器列表

寄存器名称	偏移	描述
TIM_CNT	0x0000	计数器
TIM_CTRL	0x0004	控制寄存器
STAT	0x0008	状态寄存器
TIM_PERIOD	0x000C	Timer Period 寄存器，类似一个边界点寄存器用来控制输出信号波形变化的周期
PWM_CTRL	0x0010	PWM 功能控制寄存器
PWM_CCR	0x0014	Compare 模式下，控制 PWM 的输出信号的占空比，Capture 模式下，存放捕获输入信号的计数值

13.3 寄存器说明

13.3.1 TIM_CNT

域	位	读写	复位值	描述
Reversed	31:16	RW	0x0	保留
TIM_CNT	15:0	RW	0x0	计数值

13.3.2 TIM_CTL

域	位	读写	复位值	描述
Reversed	31:28	RW	4'h0	保留
DIV	27:16	RW	12'h0	分频参数 0：1 分频； 1：2 分频；

				… 支持分频范围 1~4096
Reversed	15:6	RW	10'h0	保留
GIE	5	RW	1'h0	全局中断输出使能控制位
OVFIF_EN	4	RW	1'h0	计数中断使能控制位
Reversed	3:2	RW	1'h0	保留
ENABLE	1	RW	1'h0	全局使能位。1 表示全局使能，0 表示不使能。若需要暂停 Timer/PWM 工作，可通过该位控制。
SW_RST	0	RW	1'h0	全局软复位信号，软件写 1，部件实现全局软复位，复位完成后，自动跳出软复位。

13.3.3 STAT

域	位	读写	复位值	描述
Reversed	31:4	RW	28'h0	保留
FIFO_FULL	3	RW	1'h0	该位显示当前 FIFO 满的情况，但是不作为中断输出，在软件向 FIFO 中写数据的时候作为前提，若检测到该位是 1，则此时不允许写数据
FIFO_EMPTY	2	RW	1'h0	只有在选择使用 FIFO 作为 duty 控制，且 FIFO 为空的时候，产生中断
OVFIF	1	RW	1'h0	当计数值达到 TIM_PERIOD 寄存器值的时候，产生中断
CHIF	0	RW	1'h0	Capture 或 Compare 中断标志。写 1 清除。

13.3.4 TIM_PERIOD

域	位	读写	复位值	描述
Reversed	31:16	RW	16'h0	保留
PERIOD	15:0	RW	16'h1	输出波形周期控制位，即 period 寄存器

13.3.5 PWM_CTL

域	位	读写	复位值	描述
Reversed	31:10	RW	22'h0	保留
FIFO_EMPTY_EN	9	RW	20'h0	在 Compare 模式下，FIFO 出现空的时候，中断输出使能控制位，1 表示使能，0 表示非使能
DUTY_SEL	8	RW	1'h0	Compare 模式下，控制 duty 的比较值的来源 0：来自寄存器 PWM_CCR；1：来自 FIFO
ICOV	7	RW	1'h0	在 Compare 模式下，配置 PWM 输出的初始值
CMP	6:4	RW	3'h0	比较操作配置

				000：比较匹配时输出置 1； 001：比较匹配时输出清 0； 010：比较匹配时输出翻转； 011：向上计数且比较匹配时输出置 1；100：向上计数且比较匹配时输出清 0。
IE	3	RW	1'h0	中断使能位，用来控制是否产生 compare/capture 中断
Mode	2	RW	1'h0	0：捕获模式；1：比较模式
CAP	1:0	RW	2'h0	捕获操作配置 00：无捕获操作； 01：捕获关联信号的上升沿； 10：捕获关联信号的下降沿； 11：捕获关联信号的上升沿和下降沿

13.3.6 PWM_CCR

域	位	读写	复位值	描述
Reversed	31:17	RW	16'h0	保留
GPIO	16	RW	1'h0	输出 GPIO 控制位
CCR	15:0	RW	16'h0	Channel0 的 Capture/Compare 寄存器，该寄存器在不同模式下具有不同的功能： Compare 模式：该寄存器的值属于控制 duty cycle 的，此时，该值最小为 1，最大不能超过 TCCR 的值； Capture 模式：该寄存器的 32 位都要使用到，[15:0]存放的是输入信号下降沿采集到的次数，[31:16]存放的是输入信号上升沿采集到的次数

14 I2C

14.1 操作说明

14.1.1 配置为 master

- 配置寄存器 0x6c（IC_ENABLE）为 0；
- 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；
- 将设备地址写入寄存器 0x04（IC_TAR）；
- 使能 I2C，配置寄存器 0x6c（IC_ENABLE）为 1。

14.1.2 配置为 slave

- 配置寄存器 0x6c（IC_ENABLE）为 0；
- 写寄存器 0x00（IC_CON），例如，配置为 standard speed 的从机，该寄存器写 0x02；
- 将设备地址写入寄存器 0x08（IC_SAR）；
- 使能 I2C，配置寄存器 0x6c（IC_ENABLE）为 1。

14.1.3 master 模式发送和接收数据流程

14.1.3.1 发送数据

- 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。
- 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。
- 支持写入多字节数据，重复 1、2 步骤即可。
- 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。

14.1.3.2 接受数据

- 发送读数据命令：向 0x10（IC_DATA_CMD）bit[8]写 1，表示命令为读操作。
- 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。

- 读取数据：读 0x10（IC_DATA_CMD）地址。
- 支持读多字节数据，重复前三步即可。
- 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。

14.1.4 slave 模式发送和接收数据流程

14.1.4.1 发送数据

- 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。
- 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。
- 读 0x50（IC_CLR_RD_REQ）地址，清除中断。

14.1.4.2 接收数据

- 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。
- 读取数据：读 0x10（IC_DATA_CMD）地址。

14.1.5 接口频率调整

正常模式：$SCL\_FREQ\_SS = \frac{IC\_CLK\_FREQ}{IC\_SS\_SCL\_HCNT+IC\_SS\_SCL\_LCNT}$;

快速模式：$SCL\_FREQ\_FS = \frac{IC\_CLK\_FREQ}{IC\_FS\_SCL\_HCNT+IC\_FS\_SCL\_LCNT}$;

高速模式：$SCL\_FREQ\_HS = \frac{IC\_CLK\_FREQ}{IC\_HS\_SCL\_HCNT+IC\_HS\_SCL\_LCNT}$。

其中，IC_CLK_FREQ 为 48Mhz。

14.2 寄存器列表

寄存器名称	偏移	描述
IC_CON	0x00	I2C 控制寄存器
IC_TAR	0x04	I2C 主机地址寄存器
IC_SAR	0x08	I2C 从机地址寄存器
IC_HS_MADDR	0x0C	I2C 高速主机模式编码地址寄存器
IC_DATA_CMD	0x10	I2C 数据寄存器
IC_SS_SCL_HCNT	0x14	标准模式 I2C 时钟信号 SCL 的高电平计数寄存器
IC_SS_SCL_LCNT	0x18	标准模式 I2C 时钟信号 SCL 的低电平计数

		寄存器
IC_FS_SCL_HCNT	0x1C	快速模式 I2C 时钟信号 SCL 的高电平计数寄存器
IC_FS_SCL_LCNT	0x20	快速模式 I2C 时钟信号 SCL 的低电平计数寄存器
IC_HS_SCL_HCNT	0x24	高速模式 I2C 时钟信号 SCL 的高电平计数寄存器
IC_HS_SCL_LCNT	0x28	高速模式 I2C 时钟信号 SCL 的低电平计数寄存器
IC_INTR_STAT	0x2C	I2C 中断状态寄存器
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

14.3 寄存器说明

14.3.1 IC_CON(0x00)

域	位	读写	复位值	描述
reserved	31:7	RO	0x0	保留
IC_SLAVE_DISABLE	6	RW	0x1	I2C Slave 功能是否关闭的控制位。即在使用 I2C 功能时通过配置此参数控制 I2C Slave 功能是打开还是关闭。软件驱动可以在系统复位后配置此参数，即通过软件配置 Slave 的使能或关闭并不是必需的。在默认状态下和复位状态下 I2C 的 Slave 功能均是使能的。如果此位设置为 1，则 I2C 控制器只能作为 Master 使用，不能响应反向 Slave 的请求。 0：使能 I2C Slave 功能；1：关闭 I2C Slave 功能
IC_RESTART_EN	5	RW	0x1	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 researt 功能时，I2C 的 Master 控制器不发送起始字节、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	0x1	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	RW	0x1	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	RW	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	RW	0x1	I2C Master 的使能位。0：关闭 master 功能；1：使能 master 功能

14.3.2 IC_TAR(0x04)

域	位	读写	复位值	描述
reserved	31:13	RO	0x0	保留
IC_10BITADDR_MASTER	12	RW	1	选择工作在 I2C Master 时使用 7 位地址模式还是 10 位地址模式进行通信。 0：7 位地址模式；1：10 位地址模式 声明：此位只有在 I2C_DYNAMIC_TAR_UPDATE 为“Yes”时才有效。
SPECIAL	11	RW	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	RW	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	RW	0x55	存放 Master 通信的目的地址。使用广播呼叫地址格式时此参数可以忽略，使用 START BYTE 格式时只需 CPU 向此处进行一次写操作。

14.3.3 IC_SAR(0x08)

域	位	读写	复位值	描述
reserved	31:10	RO	0x0	保留
IC_SAR	9:0	RW	0x55	IC_SAR 存放 I2C 工作在 Slave 模式下的 Slave 地址。7 位地址模式下只使用 IC_SAR[6:0]。只有在关闭 I2C 接口功能时（IC_ENABLE=0）才能更新 IC_SAR 的值，在 I2C 接口处于使能状态时不能改变 IC_SAR 的值。

14.3.4 IC_HS_MADDR(0x0C)

域	位	读写	复位值	描述
reserved	31:3	RO	0x0	保留
IC_HS_MAR	2:0	RW	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

14.3.5 IC_DATA_CMD(0x10)

域	位	读写	复位值	描述
reserved	31:11	RO	0x0	保留
RESTART	10	WO	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 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。

14.3.6 IC_SS_SCL_HCNT(0x14)

域	位	读写	复位值	描述
reserved	31:16	RO	0x0	保留
IC_SS_SCL_HCNT	15:0	RW	0x190	该寄存器必须在 I2C 总线传输之前进行设计，用于明确正确的 I/O 时序。。该寄存器用于设置标准速率下 SCL 高电平持续时间的计数值。 该寄存器仅当 I2C 接口在不使能情况下（当 IC_ENABLE=0 时）可写。其他情况下的写操作无效。 寄存器最小取值为 6，比 6 小的值无法设置，若设置值小于 6，则硬件将寄存器值设置为 6。当 APB_DATA_WIDTH=8 时，寄存器设置的顺序尤为关键，此时，首先应配置计数器的低 32 位数据，之后再配置高 32 位。 当 IC_HC_COUNT_VALUES 为 1 时，该寄存器只读。

14.3.7 IC_SS_SCL_LCNT(0x18)

域	位	读写	复位值	描述
reserved	31:16	RO	0x0	保留
IC_SS_SCL_LCNT	15:0	RW	0x1d6	该寄存器必须在 I2C 总线传输之前进行设计，用于明确正确的 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。

14.3.8 IC_FS_SCL_HCNT(0x1C)

域	位	读写	复位值	描述
reserved	31:16	RO	0x0	保留
IC_FS_SCL_LCNT	15:0	RW	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 高电平持续时间的计数值。用于发送高速模式下的 Mater Code 和 START BYTE 或 General Call。 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。

14.3.9 IC_FS_SCL_LCNT(0x20)

域	位	读写	复位值	描述
reserved	31:16	RO	0x0	保留
IC_FS_SCL_LCNT	15:0	RW	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。

14.3.10 IC_HS_SCL_HCNT(0x24)

域	位	读写	复位值	描述
reserved	31:16	RO	0x0	保留
IC_HS_SCL_HCNT	15:0	RW	0x06	该寄存器必须在 I2C 总线传输之前进行设计，用于明确正确的 I/O 时序。该寄存器用于设置高速模式下 SCL 高电平持续时间的计数值。 SCL 高电平时间依赖于总线的负载情况。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。

14.3.11 IC_HS_SCL_LCNT(0x28)

域	位	读写	复位值	描述
reserved	31:16	RO	0x0	保留
IC_HS_SCL_LCNT	15:0	RW	0x10	该寄存器必须在 I2C 总线传输之前进行设计，用于明确正确的 I/O 时序。该寄存器用于设置高速模式下 SCL 低电平持续时间的计数值。 SCL 低电平时间依赖于总线的负载情况。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。

14.3.12 IC_INTR_STAT(0x2C)

域	位	读写	复位值	描述
reserved	31:12	RO	0x0	保留
R_GEN_CALL	11	RO	0x0	此寄存器为中断状态寄存器。寄存器的每位中断状态在 IC_INTR_MASK 寄存器中都有相应的中断屏蔽位，这些中断状态位可以通过读对应的中断清除寄存器进行清除。未被屏蔽的原始中断可以参考 IC_RAW_INTR_STAT 寄存器。
R_START_DET	10	RO	0x0	
R_STOP_DET	9	RO	0x0	
R_ACTIVITY	8	RO	0x0	
R_RX_DONE	7	RO	0x0	
R_TX_ABRT	6	RO	0x0	
R_RD_REQ	5	RO	0x0	
R_TX_EMPTY	4	RO	0x0	
R_TX_OVER	3	RO	0x0	
R_RX_FULL	2	RO	0x0	
R_RX_OVER	1	RO	0x0	
R_RX_UNDER	0	RO	0x0	

14.3.13 IC_INTR_MASK(0x30)

域	位	读写	复位值	描述
reserved	31:12	RO	0x0	保留
M_GEN_CALL	11	RW	0x1	此寄存器为中断屏蔽寄存器。寄存器的每一位都可以屏蔽 IC_INTR_STAT 寄存器中对应的中断位。此寄存器是低有效：0 表示中断屏蔽，1 表示中断不屏蔽。 例如： bit[11]：M_GEN_CALL 中断事件标志屏蔽控制。置 0 时，如果对应中断事件发生，不会置位 IC_INTR_STAT 寄存器中对应的中断标志位。
M_START_DET	10	RW	0x0	
M_STOP_DET	9	RW	0x0	
M_ACTIVITY	8	RW	0x0	
M_RX_DONE	7	RW	0x1	
M_TX_ABRT	6	RW	0x1	
M_RD_REQ	5	RW	0x1	
M_TX_EMPTY	4	RW	0x1	
M_TX_OVER	3	RW	0x1	
M_RX_FULL	2	RW	0x1	
M_RX_OVER	1	RW	0x1	
M_RX_UNDER	0	RW	0x1	

14.3.14 IC_RAW_INTR_STAT(0x34)

域	位	读写	复位值	描述
reserved	31:12	RO	0x0	保留
GEN_CALL	11	RO	0x0	只有接收并识别到 General Call 格式时才会被置位。一旦 GEN_CALL 置位，则只有通过关闭 I2C 控制器或 CPU 读取 IC_CLR_GEN_CALL 寄存器中的第 0 位，

				GEN_CALL 位才能被清 0。I2C 控制器会把接收到的数据存放在 Rx 缓冲区中。
START_DET	10	RO	0x0	此位状态表示在 I2C 总线接口上是否产生了 START 或 RESTART。与控制器工作在 Master 模式还是 Slave 模式无关。
STOP_DET	9	RO	0x0	此位状态表示在 I2C 总线接口上是否产生了 STOP。与控制器工作在 Master 模式还是 Slave 模式无关。
ACTIVITY	8	RO	0x0	此位表示 I2C 控制器的活动状态。有 4 种方法可以清除 ACTIVITY 标志：关闭 DW_apb_i2c；读取 IC_CLR_ACTIVITY 寄存器；读取 IC_CLR_INTR 寄存器；系统复位。一旦被置位则会一直保持置位，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	0x0	I2C 控制器工作在 Slave 发送模式下，发送完数据的最后一个字节后，在规定时间内没有收到 Master 端的回应（ACK），RX_DONE 将会被置位表示结束。
TX_ABRT	6	RO	0x0	该位表明如果 DW_apb_i2c 不能完成所期望的对传输 FIFO 内容的操作。这种情况可能发生在 I2C 作为主机或从机上，叫作“transmit abort”。置 1 时，IC_TX_ABRT_SOURCE 寄存器将指出 ransmit abort 发生的原因。
RD_REQ	5	RO	0x0	读请求标志。当 I2C 控制器工作在 Slave 模式下，且有 Master 尝试从 DW_apb_i2c 中读取数据时，RD_REQ 被置位。I2C 控制器在处理 RD_REQ 请求期间会将 SCL 保持低电平。RD_REQ 是处理器必须响应的中断请求，并在请求处理完成时把 Master 所要的数据放到 IC_DATA_CMD 寄存器中。读取 IC_CLR_RD_REQ 寄存器的值可以将 RD_REQ 标志清 0。
TX_EMPTY	4	RO	0x0	当发送缓冲区小于等于 IC_TX_TL 寄存器中设定的门限值时将置位 TX_EMPTY。当缓冲区大于门限值时，硬件会自动把 TX_EMPTY 清 0。IC_ENABLE bit0=0 时，TXFIFO 被刷新复位，TXFIFO 可以认为为空，此时 TX_EMPTY 被置为 1。当总线处于非活动状态时 ic_en=0，TX_EMPTY=0。
TX_OVER	3	RO	0x0	在发送过程中，如果发送缓冲区大小达到 IC_TX_BUFFER_DEPTH 且处理器还在尝

				试通过向 IC_DATA_CMD 中写数据来发起另一个 I2C 命令时，TX_OVER 被置位。即使在控制器功能被关闭的情况下（IC_ENABLE[0]=0）RX_OVER 状态也会一直保持置位，直到总线进入空闲状态。ic_en =0 时，TX_OVER 被清 0。
RX_FULL	2	RO	0x0	当接收缓冲区大于等于 IC_RX_TL 中设定的门限值（RX_TL)时，RX_FULL 置位。当缓冲区小于门限值时，硬件会自动把 RX_FULL 清 0。IC_ENABLE bit0=0 时，RXFIFO 被刷新复位，RXFIFO 为空，此时 RX_FULL 被清 0。
RX_OVER	1	RO	0x0	当接收缓冲区大小达到 IC_RX_BUFFER_DEPTH ，且还继续从外部接收数据时，RX_OVER 置位。TX_OVER 事件会被 I2C 控制器响应，且在缓冲区满后接收到的所有数据均被丢弃。即使在控制器功能被关闭的情况下（IC_ENABLE[0]=0）RX_OVER 状态也会一直保持置位，直到总线进入空闲状态。ic_en= 0 时，RX_OVER 被清 0。
RX_UNDER	0	RO	0x0	处理器通过访问 IC_DATA_CMD 寄存器获取接收缓冲区的数据时，若接收缓冲区为空，RX_UNDER 被置位。即使在控制器功能被关闭的情况下（IC_ENABLE[0]=0）RX_UNDER 状态也会一直保持置位，直到总线进入空闲状态。ic_en =0 时，RX_UNDER 被清 0。

14.3.15 IC_RX_TL(0x38)

域	位	读写	复位值	描述
reserved	31:8	RO	0x0	保留
RX_TL	7:0	RW	0x0	接收缓冲区满中断（RX_FULL）触发门限控制。有效范围 0~255，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。

14.3.16 IC_TX_TL(0x3C)

域	位	读写	复位值	描述
reserved	31:8	RO	0x0	保留
TX_TL	7:0	RW	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。

14.3.17 IC_CLR_INTR(0x40)

域	位	读写	复位值	描述
reserved	31:1	RO	0x0	保留
CLR_INTR	0	RO	0x0	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。

14.3.18 IC_CLR_RX_UNDER(0x44)

域	位	读写	复位值	描述
reserved	31:1	RO	0x0	保留
CLR_RX_UNDER	0	RO	0x0	读取这个寄存器以清除 IC_RAW_INTR_STAT 寄存器的 RX_UNDER 中断 (bit 0)

14.3.19 IC_CLR_RX_OVER(0x48)

域	位	读写	复位值	描述
reserved	31:1	RO	0x0	保留
CLR_RX_OVER	0	RO	0x0	读取这个寄存器以清除 IC_RAW_INTR_STAT 寄存器的 RX_UNDER 中断 (bit 1)

14.3.20 IC_CLR_TX_OVER(0x4C)

域	位	读写	复位值	描述
reserved	31:1	RO	0x0	保留
CLR_TX_OVER	0	RO	0x0	读取这个寄存器以清除 IC_RAW_INTR_STAT 寄存器的 RX_OVER 中断 (bit3)

14.3.21 IC_CLR_RD_REQ(0x50)

域	位	读写	复位值	描述
reserved	31:1	RO	0x0	保留
CLR_RD_REQ	0	RO	0x0	读取这个寄存器以清除 IC_RAW_INTR_STAT 寄存器的 RD_REQ 中断 (bit5)

14.3.22 IC_CLR_TX_ABRT(0x54)

域	位	读写	复位值	描述
reserved	31:1	RO	0x0	保留
CLR_TX_ABRT	0	RO	0x0	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

14.3.23 IC_CLR_RX_DONE(0x58)

域	位	读写	复位值	描述
reserved	31:1	RO	0x0	保留
CLR_RX_DONE	0	RO	0x0	读取这个寄存器以清除 IC_RAW_INTR_STAT 寄存器的 RX_DONE 中断 (bit7)

14.3.24 IC_CLR_ACTIVITY(0x5C)

域	位	读写	复位值	描述
reserved	31:1	RO	0x0	保留
CLR_ACTIVITY	0	RO	0x0	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。

14.3.25 IC_CLR_STOP_DET(0x60)

域	位	读写	复位值	描述
reserved	31:1	RO	0x0	保留
CLR_STOP_DET	0	RO	0x0	读这个寄存器来清除 IC_RAW_INTR_STAT 状态寄存器的 STOP_DET 中断 (bit 9)

14.3.26 IC_CLR_START_DET(0x64)

域	位	读写	复位值	描述
reserved	31:1	RO	0x0	保留
CLR_START_DET	0	RO	0x0	读这个寄存器来清除 IC_RAW_INTR_STAT 状态寄存器的 START_DET 中断 (bit 10)

14.3.27 IC_CLR_GEN_CALL(0x68)

域	位	读写	复位值	描述
reserved	31:1	RO	0x0	保留
CLR_START_DET	0	RO	0x0	读这个寄存器来清除 IC_RAW_INTR_STAT 状态寄存器的 GEN_CALL 中断 (bit 11)

14.3.28 IC_ENABLE(0x6C)

域	位	读写	复位值	描述
reserved	31:1	RO	0x0	保留
ENABLE	0	RO	0x0	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。

14.3.29 IC_STATUS(0x70)

域	位	读写	复位值	描述
reserved	31:7	RO	0x0	保留
SLV_ACTIVITY	6	RO	0x0	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	0x0	Master FSM 活动状态标志。Master FSM(Master Finite State Machine) 处于非 Idle 状态时被置位。 0：Master FSM 处于 Idle 状态，此时 I2C 控制器的 Master 功能处于非活动状态 1：Master FSM 处于非 Idle 状态，此时 I2C 控制器的 Master 功能处于活动状态。
RFF	4	RO	0x0	接收 FIFO 全满标志。当接收 FIFO 全满时

				置位;FIFO 中有一个或一个以上为空时 0。 0：接收 FIFO 未满；1：接收 FIFO 全满
RFNE	3	RO	0x0	接收 FIFO 不为空标志。当接收 FIFO 不为空时置位，为空时清 0。 0：接收 FIFO 为空；1：接收 FIFO 不为空
TFE	2	RO	0x1	发送 FIFO 全空标志。发送 FIFO 全空时置位；发送 FIFO 有一个或一个以上不为空的值时清 0。此标志的产生不伴随有中断发生。 0：发送 FIFO 不为空；1：发送 FIFO 为空
TFNF	1	RO	0x1	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	0x0	I2C 控制器活动状态标志

14.3.30 IC_TXFLR(0x74)

域	位	读写	复位值	描述
reserved	[31:TX_ABW+1]	RO	0x0	保留
TXFLR	[TX_ABW:0]	RO	0x0	发送 FIFO 中的有效数据量。

14.3.31 IC_RXFLR(0x78)

域	位	读写	复位值	描述
reserved	[31:RX_ABW+1]	RO	0x0	保留
RXFLR	[RX_ABW:0]	RO	0x0	接收 FIFO 中的有效数据量。

14.3.32 IC_SDA_HOLD(0x7C)

域	位	读写	复位值	描述
reserved	31:16	RO	0x0	保留
IC_SDA_HOLD	15:0	RW	0x1	设置所需的 SDA 保持时间以 48Mhz 周期为单位

14.3.33 IC_TX_ABRT_SOURCE(0x80)

域	位	读写	复位值	描述
reserved	31:16	RO	0x0	保留
ABRT_SLVRD_INTX	15	RO	0x0	1：当处理器端响应从模式请求将数据传送到远程主机并且用户在 IC_DATA_CMD 寄存器写入 1。
ABRT_SLV_ARBLOST	14	RO	0x0	1：从机在传输数据给远程主机时丢失总线占用，同时 IC_TX_ABRT_SOURCE[12]被设置。
ABRT_SLVFLUSH_TXFIFO	13	RO	0x0	1：从设备接收到一个读命令并且发

				送 FIFO 有数据，从设备发出 TX_ABRT 中断来刷新发送 FIFO 的数据。
ARB_LOST	12	RO	0x0	1：主机失去了仲裁，或者 IC_TX_ABRT_SOURCE[14] 被设置，从机发送丢失仲裁。
ABRT_MASTER_DIS	11	RO	0x0	1：用户试图禁用主模式的情况下禁用主操作。适用主机发送或从机发送模式
ABRT_10B_RD_NORSTRT	10	RO	0x0	1：Restart 被禁用 (IC_RESTART_EN bit (IC_CON[5]) = 0)并且主机以 10 位寻址模式发出命令。适用于主机接收模式。
ABRT_SBYTE_NORSTRT	9	RO	0x0	要清除该位，ABRT_SBYTE_NORSTRT 必须是确定的，且 IC_CON[5]=1，且 SPECIAL 位 (IC_TAR[11]) 必须清除，或 GC_OR_STARTt(IC_TAR[10])清除。 1：Restart 位被禁用，即 IC_Restart_en(IN_CON[5]=0)时，用户试图发送起始字节。 适用于主机模式
ABRT_HS_NORSTRT	8	RO	0x0	1：Restart 被禁用，即 (IC_RESTART_EN bit (IC_CON[5]) = 0)，用户尝试使用主机以高速模式传输数据。适用于主机发送和接收模式
ABRT_SBYTE_ACKDET	7	RO	0x0	1：主机发送一个 START 字节，但是 start 字节已被发送（错误行为）。适用于主机模式。
ABRT_HS_ACKDET	6	RO	0x0	1：处于高速模式，但是高速主编码已被识别 (错误行为)。适用于主机模式。
ABRT_GCALL_READ	5	RO	0x0	1：I2C 在主模式下发送了一个 General Call，但用户在 General Call 之后的字节被编程为读 (IC_DATA_CMD[9]置 1)适用于主发送模式
ABRT_GCALL_NOACK	4	RO	0x0	1：DW_apb_i2c 在主模式下发送了一个 General Call 并且没有从机在总线上承认
ABRT_TXDATA_NOACK	3	RO	0x0	1：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。适用于主发送模式。
ABRT_10ADDR2_NOACK	2	RO	0x0	1：主设备处于 10 位地址模式，10 位地址的第二个地址字节未被任何从机承认。适用于主机的发送和接收
ABRT_10ADDR1_NOACK	1	RO	0x0	1：主设备处于 10 位地址模式，10 位地址的第 1 个地址字节未被任何从机承认。适用于主机发送和接收。
ABRT_7B_ADDR_NOACK	0	RO	0x0	1：主设备处于 7 位地址模式，地址发送未被任何从机承认。适用于主机的发送和接收。

14.3.34 IC_SLV_DATA_NACK_ONLY(0x84)

域	位	读写	复位值	描述
reserved	31:1	RO	0x0	保留
NACK	0	RW	0x0	生成 NACK。NACK 只发生在当 I2C 作为从机接收时。如果将这个寄存器置 1，那么它只能在接收数据字节之后生成一个 NACK 。因此数据传输被终止，接收到的数据不会被推送到接收缓冲区。当寄存器设置为 0 时，它将根据正常条件生 NACK/ACK。

14.3.35 IC_DMA_CR(0x88)

域	位	读写	复位值	描述
reserved	31:2	RO	0x0	保留
TDMAE	1	RW	0x0	DMA 传输使能位。这个位可以启用/禁用发送 FIFO 的 DMA 通道 0：发送 DMA 禁用；1：发送 DMA 启用
RDMAE	0	RW	0x0	接收 DMA 使能位。这个位可以启用/禁用接收 FIFO DMA 通道 0：接收 DMA 禁用；1：接收 DMA 启用

14.3.36 IC_DMA_TDLR(0x8C)

域	位	读写	复位值	描述
reserved	31:TX_ABW	RO	0x0	保留
DMATDL	TX_ABW–1:0	RW	0x0	传输数据层。该位控制传输逻辑发出 DMA 请求的中有效项的数量等于或低于此字段值，且 TDMAE = 1。

14.3.37 IC_DMA_RDLR(0x90)

域	位	读写	复位值	描述
reserved	31:RX_ABW	RO	0x0	保留
DMARDL	[RX_ABW–1:0]	RW	0x0	接收数据阈值。该位控制接收逻辑中的一个 DMA 请求的阈值。

14.3.38 IC_SDA_SETUP(0x94)

域	位	读写	复位值	描述
reserved	31:8	RO	0x0	保留
SDA_SETUP	7:0	RW	0x64	建议如果所需延时为 1000ns，对于频率为 10MHz 的 ic_clk，IC_SDA_SETUP 编程为 0x11。IC_SDA_SETUP 必须以最小值 0x2 来编程。

14.3.39 IC_ACK_GENERAL_CALL(0x98)

域	位	读写	复位值	描述
reserved	31:1	RO	0x0	保留
ACK_GEN_CALL	0	RW	0x1	ACK General Call。 1：当 I2C 收到 General Call 时，I2C 以 ACK 响应；0：I2C 不生成 General Call 中断。

14.3.40 IC_ENABLE_STATUS(0x9C)

域	位	读写	复位值	描述
reserved	31:3	RO	0x0	保留
SLV_RX_DATA_LOST	2	RO	0x0	从机收到的数据丢失。
SLV_DISABLED_WHILE_BUSY	1	RO	0x0	从机在忙时禁用（发送、接收）。该位表示 I2C 从机在忙时，IC_ENABLE 寄存器由 1 设置成 0。
IC_EN	0	RO	0x0	ic_en 状态。

14.3.41 IC_FS_SPKLEN(0xA0)

域	位	读写	复位值	描述
reserved	31:8	RO	0x0	保留
IC_FS_SPKLEN	7:0	RW	0x5	FS 模式下， 在任何 I2C 总线事务发送之前，必须设置寄存器，保证稳定运行。此寄存器在设置时间，在 IC CLK 周期中过滤掉的 SCL 或 SDA 线路中的尖峰。只有当 I2C 接口被禁用时，才能写该寄存器。其他时间的写入没有效果。最小有效值是 1。

14.3.42 IC_HS_SPKLEN(0xA4)

域	位	读写	复位值	描述
reserved	31:8	RO	0x0	保留
IC_HS_SPKLEN	7:0	RW	0x2	HS 模式下，在任何 I2C 总线事务发送之前，必须设置寄存器，保证稳定运行。此寄存器在设置时间，在 IC CLK 周期中过滤掉的 SCL 或 SDA 线路中的尖峰。只有

				当 I2C 接口被禁用时，才能写如该寄存器，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。

14.3.43 IC_COMP_PARAM_1(0xF4)

域	位	读写	复位值	描述
reserved	31:24	RO	0x0	保留
TX_BUFFER_DEPTH	23:16	RO	0x0	IC_TX_BUFFER_DEPTH 0x00：保留 0x01：2 0x02：3 ... 0xFF：256
RX_BUFFER_DEPTH	15:8	RO	0x3	IC_RX_BUFFER_DEPTH 0x00：保留 0x01：2 0x02：3 ... 0xFF：256
ADD_ENCODED_PARAMS	7	RO	0x1	IC_ADD_ENCODED_PARAMS 0：错误；1：正确
HAS_DMA	6	RO	0x0	IC_HAS_DMA 0：错误；1：正确
INTR_IO	5	RO	0x1	IC_INTR_IO 0：个体 1：联合
HC_COUNT_VALUES	4	RO	0x0	IC_HC_COUNT_VALUES 0：错误 1：正确
MAX_SPEED_MODE	3:2	RO	0x3	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 位 0x1：16 位 0x2：32 位 0x3：保留

15 UART

UART（Universal Asynchronous Receiver/Transmitter）是一种用于异步通信的通用串行数据总线，符合 ARM 协议规范，作为一个从设备挂在 APB2 总线上。

15.1 操作说明

15.1.1 初始化配置

- 配置前需要先关闭 UART：向 0x30（UARTCR）地址的 bit[0]写 0。
- 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((0.042*64)+0.5)=3

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。

- 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。

- 如果需要使能中断，向 0x38（UARTIMSC）地址相应位写 1，打开中断。
- 如果使能 FIFO，并且使能中断，需要配置产生中断的 FIFO 阈值，即向 0x34（UARTIFLS）地址写入相应数值，传输和接受 FIFO 深度都是 32 个字节。
- 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。

15.1.2 发送数据操作流程

使用轮询方式：

- 判断发送 FIFO 不满：读 0x18(UARTFR)地址，判断 bit[5]为 0 时，即发送 FIFO 不满。
- 写入数据：向 0x00（UARTDR）写入数据，根据配置一次可以写入 5-8bit。

使用中断方式：

- 判断是否产生发送中断：读 0x3C(UARTRIS)地址，判断 bit[5]为 1 时，即产生发送中断。如果使能 FIFO，当传输 FIFO 里的数据小于等于设置的 FIFO 阈值时，产生中断；如果没有使能 FIFO，相当于传输 FIFO 深度为 1 字节，当数据寄存器的数据发送后，产生中断。
- 写入数据：向 0x00（UARTDR）写入数据，当发送 FIFO 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。

15.1.3 接收数据操作流程

使用轮询方式：

- 判断接收 FIFO 不空：读 0x18(UARTFR)地址，判断 bit[4]为 0 时，即接收 FIFO 不空。
- 读出数据：读 0x00（UARTDR）数据寄存器的数据。

使用中断方式：

- 判断是否产生接收中断：读 0x3C(UARTRIS)地址，判断 bit[4]为 1 时，即产生接收中断。如果使能 FIFO，当接收 FIFO 里的数据大于等于设置的 FIFO 阈值时，产生中断；如果没有使能 FIFO，相当于接收 FIFO 深度为 1 字节，当接收到 1 字节数据后，产生中断。
- 读出数据：读 0x00（UARTDR）数据寄存器，当接收 FIFO 里的数据小于设置的 FIFO 阈值时，中断清除，或者向 0x44（UARTICR）中断清除寄存器写 0x10，清除中断。

15.1.4 Flow control 相关操作

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

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

15.2 寄存器列表

寄存器名称	偏移	描述
UARTDR	0x000	数据寄存器
UARTRSR/ UARTECR	0x004	接收状态寄存器/错误清除寄存器
UARTFR	0x018	标志寄存器
UARTILPR	0x020	低功耗计数寄存器
UARTIBRD	0x024	波特率整数值配置寄存器
UARTFBRD	0x028	波特率小数值配置寄存器
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

15.3 寄存器说明

15.3.1 UARTDR(0x000)

域	位	读写	复位值	描述
reserved	31:12	RW	0x0	保留
OE	11	RW	0x0	溢出错误。如果接收到数据并且接收的 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	0x0	突发错误。如果检测到突发条件，则该位设置为 1，表示接收到的数据输入保持在较低的状态超过一个完整的字节传输时间（定义为起始位、数据位、奇偶校验位和停止位）。 在 FIFO 模式下，此错误与 FIFO 顶部的字符相关联。当突发产生时，只有一个 0 字符加载到 FIFO 中。只有在接收数据输入为 1（标记状态）并且接收到下一个有效起始位后，才启用下一个字符。
PE	9	RW	0x0	奇偶校验错误。当设置为 1，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	0x0	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	0x0	接收（读）数据。传输（写）数据。

15.3.2 UARTRSR(0x004)

域	位	读写	复位值	描述
reserved	31:8	RW	0x0	保留
uartrsr	7:0	RW	0x0	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。

15.3.3 UARTECR(0x004)

域	位	读写	复位值	描述
reserved	31:4	RW	0x0	保留，读取值不可预测
OE	3	RW	0x0	溢出错误。如果接收到数据并且此时 FIFO 已满，此位设置为 1。该位通过写入 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	0x0	突发错误。如果检测到突发条件，则该位设置为 1，表示接收到的数据输入保持在较低的状态超过一个完整的字节传输时间（定义为起始位、数据位、奇偶校验位和停止位）。在写入 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	0x0	奇偶校验错误。当设置为 1，表示接收到数据字符的奇偶性与 UARTLCR_H 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	0x0	帧错误。当设置为 1，表示接收字符没有有效的停止位（有效停止位为 1）。通过写入 UARTECR 将该位清除为 0 在 FIFO 模式，该错误与 FIFO 顶部的字符相关联。

15.3.4 UARTFR(0x018)

域	位	读写	复位值	描述
reserved	31:9	RO	0x0	保留，读取为零。
RI	8	RO	0x0	Ring 指示信号。该位表示 UART Ring 指示信号、nUARTRI、调制解调器状态输入。当 nUARTRI 低电位时该位为 1。
TXFE	7	RO	0x1	发送 FIFO 空。该位由寄存器 UARTLCR 中 FEN 位的状态决定。如果 FIFO 被禁用，当发送保持寄存器为空时该位为 1。如果 FIFO 可使用，当发送 FIFO

				位为空时设置 TXFE 位。该位不表示发送移位寄存器是否有数据。
RXFF	6	RO	0x0	接收 FIFO 满。该位取决于 UARTLCR_H 寄存器中 FEN 位的状态。如果 FIFO 被禁用。当接收保持寄存器满时设置该位。如果 FIFO 可使用，当接收 FIFO 满时设置 RXFT 位。
TXFF	5	RO	0x0	发送 FIFO 已满。该位取决于 UARTLCR_H 寄存器中 FEN 位的状态。如果 FIFO 被禁用，当发送保持寄存器已满时设置该位。如果 FIFO 能使用，当传输 FIFO 已满时设置 TXFF 位。
RXFE	4	RO	0x1	接收 FIFO 为空。该位取决于 UARTLCR_H 寄存器中 FEN 位的状态。如果 FIFO 被禁用，当接收保持寄存器为空时设置该位。如果 FIFO 能使用，当接收 FIFO 为空时设置 RXFE 位。
BUSY	3	RO	0x0	UART 繁忙。如果该位设置为 1，UART 正忙于传输数据。该位保持设置，直到从移位寄存器发送完整字节（包括所以停止位）为止。当发送 FIFO 变成不空时该位被立刻置已，无论 UART 是否被使能。
DCD	2	RO	0x0	数据载波检测，该位表示 UART 数据载波、nUARTDCD、调制解调状态输入。当 nUARTDCD 为低电平时该为为 1。
DSR	1	RO	0x1	数据准备完成。该位表示 UART 数据准备完成、nUARTDSR、调制解调状态输入。也就是说，当 uUARTDSR 为低电平时该位为 1。
CTS	0	RO	0x1	清除发送。该位表示 UART 清除发送、nUARTCTS、调制解调状态输入的补码。当 nUARTCTS 为低电平时该位为 1。

15.3.5 UARTILPR(0x020)

域	位	读写	复位值	描述
reserved	31:9	RO	0x0	保留
ILPDVSR	7:0	RW	0x0	8 位低功耗计数器。在复位时这些位清零。

15.3.6 UARTIBRD(0x024)

域	位	读写	复位值	描述
reserved	31:16	RO	0x0	保留
BAUD DIVINT	15:0	RW	0x0	波特率计算因子的整数值。在复位时这些位被清除为 0。

15.3.7 UARTFBRD(0x028)

域	位	读写	复位值	描述
reserved	31:6	RO	0x0	保留
BAUD DIVFRAC	5:0	RW	0x0	波特率计算因子的小数值。在复位时这些位清除为 0。

15.3.8 UARTLCR_H(0x02C)

域	位	读写	复位值	描述
reserved	31:8	RO	0x0	保留，读取为零。
SPS	7	RW	0x0	奇偶校验位 0：奇偶校验被禁用；1：奇偶校验两者之一 如果 EPS 位为 0 那么奇偶校验位被传输并且在数据为 1 时检查。如果 EPS 位为 1 那么奇偶校验位被传输并且在数据为 0 时检查。当 PEN 位禁用奇偶校验检测和生成时，该位没有效果。
WLEN	6:5	RW	0x0	数据长度。表示在一次发送或接收的数据位的数量，如下所示： b11：8 位 b10：7 位 b01：6 位 b00：5 位
FEN	4	RW	0x0	FIFO 使能位。 0：FIFOs 是禁用的（字符模式），FIFOs 是一个字节深度的保持寄存器。 1：传输与接收 FIFO 缓冲区使能（FIFO 模式）。
STP2	3	RW	0x0	两个停止位选择。如果该位设置为 1，两个停止位正在帧末尾传输。接收逻辑不检查接收到的两个停止位。
EPS	2	RW	0x0	奇偶类型选择。在传输和接收期间控制 UART 使用的奇偶校验类型。 0：奇数校验；1：偶数校验。 当 PEN 位禁用奇偶校验检测和生成时该位无效。
PEN	1	RW	0x0	奇偶校验使能。 0：奇偶校验被禁止；1 = 奇偶校验使能。
BRK	0	RW	0x0	发送突发命令，如果该位设置为 1，则在完成当前字符传输后，UARTTXD 会持续输出一个低电平。为了正确执行突发命令，软件必须将该位设置为至少两个完整的帧传输。 对于正常使用，该位必须清除为 0。

15.3.9 UARTCR(0x030)

域	位	读写	复位值	描述
reserved	31:16	RO	0x0	保留
CTSEn	15	RW	0x0	CTS 硬件流控使能端。如果该位设置为 1，CTS 硬件流控使能为可用。数据仅在 nUARTCTS 信号有效时传输。
RTSEn	14	RW	0x0	RTS 硬件流控使能端。如果该位设置为 1，RTS 硬件流控可用。数据仅在接收 FIFO 不满时接收它的请求。

Out2	13	RW	0x0	该位是 nUATROut2 调制解调状态输出。当该位被编程为 1 时，输出端为 0。对于 DTE 这可以作为 RI。
Out1	12	RW	0x0	该位是 nUARTOut1 调制解调状态输出，当该位被编程为 1 时，输出为 0。对于 DTE 这可以作为 DCD。
RTS	11	RW	0x0	该位是 UART 发送请求、nUARTRTS、调制解调状态输出。当该位被编程为 1 时，nUARTRTS 为低。
DTR	10	RW	0x0	该位是 UART 数据传输准备完成、nUARTDTR、调制解调状态输出。当该位被编程为 1 时，nUARTDTR 为低。
RXE	9	RW	0x1	接收使能端。如果该位设置为 1，UART 的接收部分能用。具体是 UART 信号还是 SIR 信号的数据接收发生取决与 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	0x1	发送使能位。如果该位设置为 1，UART 的发送部分能使用。具体是 UART 信号或 SIR 信号数据传输的发生取决于 SIREN 位的设置。当在发送数据中 UART 禁用，UART 会先完成当前的传输。
LBE	7	RW	0x0	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。
reserved	6:3	RO	0x0	保留，读取为零。
SIRLP	2	RW	0x0	SIR low-power IrDA 模式。该位用于选择 IrDA 编码模式。如果该位被清除为 0，则低电平位作为有效的高脉冲传输，脉冲宽度为位周期的 3/16。如果该位被设置为 1，则脉冲宽度为 IrLPBAUD16 输入信号周期的 3 倍。设置该位使用较少的功耗，但可能会缩短传输距离。
SIREN	1	RW	0x0	SIR 使能位： 0：IrDA SIR ENDEC 被禁用。nSITOUT 保持为低（无光脉冲产生），在 SIRIN 上的信号传输无效。 1：IrDA SIR ENDEC 能使用。数据在 nSIROUT 和 SIRIN 上发送与接收。UARTTXD 保持为高。信号在 UARTRXD 传输或调制解调状态输入无效。 如果 UARTEN 位禁用 UART 那么该位无效。
UARTEN	0	RW	0x0	UART 使能端： 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。

15.3.10 UARTIFLS(0x034)

域	位	读写	复位值	描述
reserved	31:6	RO	0x0	保留
RXIFLSEL	5:3	RW	0x2	接收 FIFO 中断的阈值选择。中断的触发点如下： b000：接收 FIFO ≥ 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	0x2	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

15.3.11 UARTIMSC(0x038)

域	位	读写	复位值	描述
reserved	31:11	RO	0x0	保留，读取为零。
OEIM	10	RW	0x0	溢出错误中断屏蔽。读取返回 UARTOEINTR 中断的当前屏蔽。 1：设置 UARTOEINTR 中断屏蔽；0：清除屏蔽。
BEIM	9	RW	0x0	突发错误中断屏蔽。读操作返回当前的 UARTBEINTR 中断屏蔽。 1：设置 UARTBEINTR 中断屏蔽；0：清除屏蔽。
PEIM	8	RW	0x0	奇偶校验错误中断屏蔽。读操作返回当前的 UARTPEINTR 中断屏蔽。 1：设置 UARTPEINTR 中断屏蔽；0：清除屏蔽。
FEIM	7	RW	0x0	帧错误中断屏蔽。读操作返回当前的 UARTFEINTR 中断屏蔽。 1：设置 UARTFEINTR 中断屏蔽；0：清除屏蔽。
RTIM	6	RW	0x0	接收超时中断屏蔽。读操作返回当前的 UARTRTINTR 中断屏蔽。 1：设置 UARTRTINTR 中断屏蔽；0：清除屏蔽。
TXIM	5	RW	0x0	发送中断屏蔽。读操作返回当前的 UARTTXINTR 中断屏蔽。 1：设置 UARTTXINTR 中断屏蔽；0：清除屏蔽。
RXIM	4	RW	0x0	接收中断屏蔽。读操作返回当前的

				UARTRXINTR 中断屏蔽。 1：设置 UARTRXINTR 中断屏蔽；0：清除屏蔽。
DSRMIM	3	RW	0x0	nUARTDSR 调制解调中断屏蔽。读操作返回当前的 UARTDSRINTR 中断屏蔽。 1：设置 UARTDSTINTR 中断屏蔽；0：清除屏蔽。
DCDMIM	2	RW	0x0	nUARTDCD 调制解调中断屏蔽。读操作返回当前的 UARTDCDINTR 中断屏蔽。 1：设置 UARTDCDINTR 中断屏蔽；0：清除屏蔽。
CTSMIM	1	RW	0x0	nUARTCTS 调制解调中断屏蔽。读操作返回当前的 UARTCTSINTR 中断屏蔽。 写 1 时，设置 UARTCTSINTR 中断屏蔽。写 0 时，清除屏蔽。
RIMIM	0	RW	0x0	nUARTRI 调制解调中断屏蔽。读操作返回当前的 UATTRIINTR 中断屏蔽。写 1 时，设置 UARTRIINTR 中断屏蔽。写 0 时，清除屏蔽。

15.3.12 UARTRIS(0x03C)

域	位	读写	复位值	描述
reserved	31:11	RO	0x0	保留，读取为零。
OERIS	10	RO	0x0	溢出错误中断状态。反馈 UARTOEINTR 中断的原始中断状态。
BERIS	9	RO	0x0	突发错误中断状态。反馈 UARTBEINTR 中断的原始中断状态。
PERIS	8	RO	0x0	奇偶校验错误中断状态。反馈 UARTPEINTR 中断的原始中断状态。
FERIS	7	RO	0x0	帧错误中断状态。反馈 UARTFEINTR 中断的原始中断状态。
RTRIS	6	RO	0x0	接收超时中断状态。反馈 UATRTRINTR 中断的原始中断状态。
TXRIS	5	RO	0x0	发送中断状态。反馈 UARTTXINTR 中断的原始中断状态。
RXRIS	4	RO	0x0	接收中断状态。反馈 UARTTXINTR 中断的原始中断状态。
DSRRMIS	3	RO	0x0	nUARTDSR 调制解调中断状态。反馈 UARTDSRINTR 中断的原始中断状态。
DCDRMIS	2	RO	0x1	nUARTDCD 调制解调中断状态。反馈 UARTDCDINTR 中断的原始中断状态。
CTSRMIS	1	RO	0x0	nUARTCTS 调制解调中断状态。反馈 UARTCTSINTR 中断的原始中断状态。
RIRMIS	0	RO	0x1	nUARTRI 调制解调中断状态。反馈 UARTRIINTR 中断的原始中断状态。

15.3.13 UARTMIS(0x040)

域	位	读写	复位值	描述
reserved	31:11	RO	0x0	保留，读取为零。
OEMIS	10	RO	0x0	溢出错误屏蔽中断状态。返回 UARTORINTR 里的中断屏蔽状态。
BEMIS	9	RO	0x0	突发错误屏蔽中断状态。反馈 UARTBEINTR 的中断屏蔽状态。
PEMIS	8	RO	0x0	奇偶校验错误屏蔽中断状态。反馈 UARTREINTR 的中断屏蔽状态。
FEMIS	7	RO	0x0	帧错误屏蔽中断状态。反馈 UARTRRINTR 里的中断屏蔽状态。
RTMIS	6	RO	0x0	接收超时屏蔽中断状态。反馈 UARTRTINTR 里的中断屏蔽状态。
TXMIS	5	RO	0x0	发送屏蔽中断状态。反馈 UARTTXINTR 里的中断屏蔽状态。
RXMIS	4	RO	0x0	接收屏蔽中断状态。反馈 UARTRXINTR 里的中断屏蔽状态。
DSRMMIS	3	RO	0x0	nUARTDSR 调制解调屏蔽中断状态。反馈 UARTDSRINTR 里的中断屏蔽状态。
DCDMMIS	2	RO	0x0	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	0x0	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	0x0	nUARTRI 调制解调屏蔽中断状态。反馈 UARTIINTR 里的中断屏蔽状态。

15.3.14 UARTICR(0x044)

域	位	读写	复位值	描述
reserved	31:11	RO	0x0	保留，读取为零。
OEIC	10	WO	0x0	溢出错误中断清除。清除 UARTOEINTR 中断。
BEIC	9	WO	0x0	突发错误中断清除。清除 UARTBEINTR 中断。
PEIC	8	WO	0x0	奇偶校验错误中断清除。清除 UARTPEINTR 中断。
FEIC	7	WO	0x0	帧错误中断清除。清除 UARTFEINTR 中断。
RTIC	6	WO	0x0	接收超时中断清除。清除 UARTRTINTR 中断。
TXIC	5	WO	0x0	传输中断清除。清除 UARTTXINTR 中断。
RXIC	4	WO	0x0	接收中断清除。清除 UARTRXINTR 中断。
DSRMIC	3	WO	0x0	nUARTDSR 调制解调中断清除。清除 UARTDSRINTR 中断。
DCDMIC	2	WO	0x0	nUARTDCD 调制解调中断清除。清除 UARTDCDINTR 中断。
CTSMIC	1	WO	0x0	nUARTCTS 调制解调中断清除。清除 UARTCTSINTR 中断。

RIMIC	0	WO	0x0	nUARTRI 调制解调中断清除。清除 UARTRIINTR 中断。

15.3.15 UARTDMACR(0x048)

域	位	读写	复位值	描述
reserved	31:11	RO	0x0	保留，读取为零。
DMAONERR	2	RW	0x0	DMA 错误。如果该位设置为 1，DMA 接收请求输出，当 UART 产生错误 中断时，UARTRXDMASREQ 或 UARTRXDMABREQ 被禁用。
TXDMAE	1	RW	0x0	发送 DMA 使能端。如果该位设置为 1，传输 FIFO 的 DMA 启用。
RXDMAE	0	RW	0x0	接收 DMA 使能位。如果该位设置为 1，接收 FIFO 的 DMA 启用。

16 MIO

16.1 操作说明

端口功能的选择，可以通过配置寄存器配置来实现，配置为 0x00 选择 I2C，配置为 0x01 选择 UART，配置为 0x02 选择 PWM。

16.2 寄存器说明

MIO 模块寄存器空间为 8KB，其中低 4KB 空间是 3 个控制器的共享空间，当选择不同功能时，该 4KB 空间属于相关控制器的寄存器空间；高 4KB 空间为 MIO 全局控制寄存器空间。MIO 全局控制寄存器基地址为 PCI Bar0 对应的基址 + 0x1000 具体定义如下：

寄存器名称	偏移	读写	描述
MFS	0x000	RW	控制当前 MIO 功能选择 0x00：I2C（默认） 0x01：UART 0x02：PWM
MFST	0x004	RO	当前 MIO 功能状态
MV	0x100	RO	MIO 版本号

17 GPIO

共 2 个 GPIO 控制器，每个控制器提供 32 位 GPIO 引脚。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。模块不支持硬件控制，数据源与方向通过软件配置。

17.1 操作说明

17.1.1 作为数据传输信号

将对应 pin 脚的 PAD 复用类型设置为 GPIO。

- 写数据
  - 配置方向寄存器（gpio_swporta_ddr、gpio_swportb_ddr）为输出（写 1）；
  - 数据写入寄存器（gpio_swporta_dr、gpio_swportb_dr）。
- 读数据
  - 配置方向寄存器（gpio_swporta_ddr、gpio_swportb_ddr）为输入（写 0）
  - 从寄存器（gpio_ext_porta、gpio_ext_portb）中读出数据。

17.2 寄存器列表

寄存器	偏移	描述
GPIO_SWPORTA_DR	0x00	端口输出寄存器
GPIO_SWPORTA_DDR	0x04	端口方向控制寄存器
GPIO_EXT_PORTA	0x08	端口输入寄存器

17.3 寄存器说明

17.3.1 GPIO_SWPORTA_DR(0x00)

域	位	读写	复位值	描述
Port Data Register	31:0	RW	0x0	如果端口的数据方向位设置为输出模式，则写入该寄存器的值在端口的 I/O 信号线上输出。读取的值等于写入该寄存器的最后一个值。

17.3.2 GPIO_SWPORTA_DDR(0x04)

域	位	读写	复位值	描述

reserved	31:8	RO	0x0	保留
Port direction Register	7:0	RW	0x0	写入该寄存器的值独立控制端口中相应数据位的方向。默认方向配置为输入。 0：输入；1：输出

17.3.3 GPIO_EXT_PORTA(0x08)

域	位	读写	复位值	描述
External Port	31:0	RW	0x0	当端口被配置为输入，读取该位置将读取信号的值。当端口数据方向设置为输出，读取该位置将读取端口的数据寄存器。

18 SMBUS

系统管理总线（SMBus）是一种基于 I2C 扩展出来的两线接口，在 I2C 基础上定义了一些复杂的操作，为系统和电源管理相关的任务提供控制总线，本章仅描述新增的寄存器，其它寄存器描述请参考第 13 章节。

18.1 操作说明

18.1.1 设备超时处理

ic_smbclk_low_timeout 参数寄存器用于配置超时时间，ic_enable 寄存器使能相应控制位 smbus_mst_scktimout_en/smbus_slv_scktimout_en，当达到超时值时（可通过查询状态或者检测中断的方式），主机通过使能 smbus_mst_release，发送停止信号，结束传输。

18.1.2 SMBDAT 超时处理

将 SMBDAT TIMEOUT 值写入 ic_smbdat_stuck_timeout 寄存器，ic_enable 寄存器使能 smbus_mst_sdatimout_en 位后，如果 SMBDAT 线为低超时后，则生成 smbus_mst_sda_low_timeout 中断，软件可以启用 IC_ENABLE 寄存器的 smbus_mst_clkreset_en 位保持 SCL 为低直到 ic_smbclk_low_timeout，然后复位总线上所有设备的 SMBUS 接口。

18.2 寄存器列表

寄存器名称	偏移	描述
IC_SMBCLK_LOW_MEXT	0xA8	时钟 MEXT 参数寄存器
IC_SMBCLK_LOW_TIMEOUT	0xAC	时钟 TIMEOUT 参数设置寄存器
IC_SMBCLK_STUCK_TIMEOUT	0xB0	标志寄存器
IC_SMBDAT_STUCK_TIMEOUT	0xB4	低功耗计数寄存器
IC_SMBCLK_LOW_SEXT	0xB8	SEXT 参数设置寄存器
CLR_SMMST_SCL_EXT_LOW_TIMEOUT	0xBC	清除 SMMST_SCL_EXT_LOW_TIMEOUT 中断寄存器
CLR_SMMST_SCL_TMO_LOW_TIMEOUT	0xC0	清除 SMMST_SCL_TMO_LOW_TIMEOUT 中断寄存器
CLR_SMMST_SDA_LOW_TIMEOUT	0xC4	清除 SMMST_SDA_LOW_TIMEOUT 中断寄存器
CLR_SMSLV_SCL_EXT_LOW_TIMEOUT	0xC8	清除 SMSLV_SCL_EXT_LOW_TIMEOUT 中断寄存器
CLR_SMSLV_SCL_TMO_LOW_TIMEOUT	0xCC	清除 SMSLV_SCL_TMO_LOW_

		TIMEOUT 中断寄存器
CLR_SMBALERT_IN_N	0xD0	清除 SMBALERT_IN_N 中断寄存器

18.3 寄存器说明

18.3.1 IC_SMBCLK_LOW_MEXT(0xA8)

域	位	读写	复位值	描述
IC_SMBCLK_LOW_MEXT	31:0	RW	0xFFFFFFFF	时钟 MEXT 参数设置寄存器

18.3.2 IC_SMBCLK_LOW_TIMEOUT (0xAC)

域	位	读写	复位值	描述
IC_SMBCLK_LOW_TIMEOUT	31:0	RW	0xFFFFFFFF	时钟 TIMEOUT 参数设置寄存器

18.3.3 IC_SMBCLK_STUCK_TIMEOUT (0xB0)

域	位	读写	复位值	描述
IC_SMBCLK_STUCK_TIMEOUT	31:0	RW	0xFFFFFFFF	时钟扩展超时参数设置寄存器

18.3.4 IC_SMBDAT_STUCK_TIMEOUT(0xB4)

域	位	读写	复位值	描述
IC_SMBDAT_STUCK_TIMEOUT	31:0	RW	0xFFFFFFFF	数据扩展超时参数设置寄存器

18.3.5 IC_SMBCLK_LOW_SEXT(0xB8)

域	位	读写	复位值	描述
IC_SMBCLK_LOW_SEXT	31:0	RW	0xFFFFFFFF	SEXT 参数设置寄存器

18.3.6 CLR_SMMST_SCL_EXT_LOW_TIMEOUT(0xBC)

域	位	读写	复位值	描述
reserved	31:1	RO	31'h0	保留
clr_smmst_scl_ext_low_timeout	0	RO	1'h0	读这个寄存器来清除 IC_RAW_INTR_STAT 状态寄存器的 SMMST_SCL_EXT_LOW_TIMEOUT 中断 (bit 12)

18.3.7 CLR_SMMST_SCL_TMO_LOW_TIMEOUT(0xC0)

域	位	读写	复位值	描述
reserved	31:1	RO	31'h0	保留
clr_smmst_scl_tmo_low_timeout	0	RO	1'h0	读这个寄存器来清除 IC_RAW_INTR_STAT 状态寄存器的 SMMST_SCL_TMO_LOW_TIMEO

				UT 中断 (bit 13)

18.3.8 CLR_SMMST_SDA_LOW_TIMEOUT(0xC4)

域	位	读写	复位值	描述
reserved	31:1	RO	31'h0	保留
clr_smmst_scl_sda_low_timeout	0	RO	1'h0	读这个寄存器来清除 IC_RAW_INTR_STAT 状态寄存器的 SMMST_SCL_SDA_LOW_TIMEOUT 中断 (bit 14)

18.3.9 CLR_SMSLV_SCL_EXT_LOW_TIMEOUT(0xC8)

域	位	读写	复位值	描述
reserved	31:1	RO	31'h0	保留
clr_smslv_scl_ext_low_timeout	0	RO	1'h0	读这个寄存器来清除 IC_RAW_INTR_STAT 状态寄存器的 SMSLV_SCL_EXT_LOW_TIMEOUT 中断 (bit 15)

18.3.10 CLR_SMSLV_SCL_TMO_LOW_TIMEOUT(0xCC)

域	位	读写	复位值	描述
reserved	31:1	RO	31'h0	保留
clr_smslv_scl_tmo_low_timeout	0	RO	1'h0	读这个寄存器来清除 IC_RAW_INTR_STAT 状态寄存器的 SMSLV_SCL_EXT_LOW_TIMEOUT 中断 (bit 16)

18.3.11 CLR_SMBALERT_IN_N (0xD0)

域	位	读写	复位值	描述
reserved	31:1	RO	31'h0	保留
clr_smbalert_in_n	0	RO	1'h0	读这个寄存器来清除 IC_RAW_INTR_STAT 状态寄存器的 SMBALERT_IN_N 中断 (bit 17)

19 SPI

共 2 个 SPI master 接口，每个接口可以接 4 个 spi 设备。

19.1 操作说明

初始化配置流程：

- 向使能寄存器(SSIENR)0x08 写 0；
- 写控制寄存器（CTRLR0）0x00 为 0x74C7；
- 配置波特率，写寄存器（BAUDR）0x14 为 0x6，即 SCKDV，计算公式：$F_{sclk\_out} = \frac{F_{ssi\_clk}}{SCKDV}$，其中$F_{ssi_clk} = 48Mhz$。
- 分别写发送和接收 FIFO 域值，TXFTLR（0x01c），RXFTLR（0x18）为 0x2；
- 选择哪个从机接收数据，通过写寄存器（SER）0x10 对应位为 1，共 4 位，可挂 4 个从机；
- 使能 SPI 通过向寄存器（SSIENR）0x08 写 1。

19.2 寄存器列表

寄存器	偏移	描述
CTRLR0	0x00	控制寄存器 0
CTRLR1	0x04	控制寄存器 1
SSIENR	0x08	SPI 使能寄存器
MWCR	0x0C	Microwire 控制
SER	0x10	从机使能寄存器
BAUDR	0x14	波特率选择寄存器
TXFTLR	0x18	发送 FIFO 阈值寄存器
RXFTLR	0x1C	接收 FIFO 阈值寄存器
TXFLR	0x20	发送 FIFO 等级寄存器
RXFLR	0x24	接收 FIFO 等级寄存器
SR	0x28	状态寄存器
IMR	0x2C	中断屏蔽寄存器
RISR	0x34	中断状态寄存器
TXOICR	0x38	清除发送 FIFO 溢出中断寄存器
RXOICR	0x3C	清除接收 FIFO 溢出中断寄存器
RXUICR	0x40	清除发送 FIFO 下溢中断寄存器
MSTICR	0x44	清除多主机争用中断寄存器
ICR	0x48	中断清除寄存器
DMACR	0x4C	DMA 控制寄存器
DMATDLR	0x50	DMA 发送数据等级寄存器

DMARDLR	0x54	DMA 接收数据等级寄存器
IDR	0x58	识别码
DR	0x60-0xEC	数据寄存器
RX_SAMPLE_DLY	0xFC	接收数据延时寄存器

19.3 寄存器说明

19.3.1 CTRLR0(0x00)

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

19.3.2 CTRLR1(0x04)

域	位	读写	复位值	描述
reserved	31:16	RW	0x0	保留
NDF	15:0	RW	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。

19.3.3 SSIENR(0x08)

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

19.3.4 MWCR(0x0C)

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

19.3.5 SER(0x10)

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

19.3.6 BAUDR(0x14)

域	位	读写	复位值	描述
reserved	31:16	RW	0x0	保留
SCKDV	15:0	RW	0x0	SSI 时钟除法器。 SCKDV 为 2～65534 之间的任何偶数值。例如: Fssi_clk = 48MHz，SCKDV =2 Fsclk_out = 48/2 = 24MHz

19.3.7 TXFTLR(0x18)

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

19.3.8 RXFTLR(0x1C)

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

19.3.9 TXFLR(0x20)

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

19.3.10 RXFLR(0x24)

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

19.3.11 SR(0x28)

域	位	读写	复位值	描述
reserved	31:7	RO	0x0	保留
DCOL	6	RO	0x0	传输数据冲突错误。仅当配置 SPI 为主机时才相关。该位通知处理器最后一次传输在完成前已经停止，读时这个位被清除。 0：没有错误；1：传输数据冲突错误。
TXE	5	RO	0x0	传输错误。如果传输开始时传输 FIFO 为空则设置该位。只有当 SPI 设置为从设备时，才设置该位。读取时将清除此位。 0：没有错误；1：传输错误
RFF	4	RO	0x0	接收 FIFO 满。当接收 FIFO 完全被填满时置 1，当接收 FIFO 包含一个或多个条目是被清除。 0：接收 FIFO 不满；1：接收 FIFO 满
RFNE	3	RO	0x0	接收 FIFO 不为空。当接收 FIFO 包含一个或多个条目设置时，当接收 FIFO 为空时清除。这个位可以被软件轮询已完全清空接收 FIFO 的数据。 0：接收 FIFO 为空；1：接收 FIFO 不空
TFE	2	RO	0x0	传送FIFO为空. 发送FIFO完全为空时，设置该位. 当

				传输 FIFO 包含一个或多个有效值时将清除该位,位不请求中断 0：传输 FIFO 不空；1：传输 FIFO 为空
TFNF	1	RO	0x0	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	0x0	SPI 总线繁忙标志位。当设置时，指示正在进行串行传输；如果以清除，则指示 SPI 为空闲。 0：SPI 空闲；1： SPI 忙

19.3.12 IMR(0x2C)

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

19.3.13 RISR(0x34)

域	位	读写	复位值	描述
reserved	31:6	RO	0x0	保留
MSTIR	5	RO	0x0	多主机冲突生成中断状态。如果将 SPI 配置成串行从设备，则不存在此字段。 0：优先屏蔽 ssi_mst_intr 后中断不活动 1：优先屏蔽 ssi_mst_intr 后中断活动
RXFIR	4	RO	0x0	接收 FIFO 满生成中断状态 0：优先屏蔽 ssi_mst_intr 后中断不活动 1：优先屏蔽 ssi_mst_intr 后中断活动
RXOIR	3	RO	0x0	接收 FIFO 上溢生成中断状态

				0：优先屏蔽 ssi_mst_intr 后中断不活动 1：优先屏蔽 ssi_mst_intr 后中断活动
RXUIR	2	RO	0x0	接收 FIFO 下溢生成中断状态 0：优先屏蔽 ssi_mst_intr 后中断不活动 1：优先屏蔽 ssi_mst_intr 后中断活动
TXOIR	1	RO	0x0	传输 FIFO 上溢生成中断状态 0：优先屏蔽 ssi_mst_intr 后中断不活动 1：优先屏蔽 ssi_mst_intr 后中断活动
TXEIR	0	RO	0x0	传输 FIFO 空生成中断状态 0：优先屏蔽 ssi_mst_intr 后中断不活动 1：优先屏蔽 ssi_mst_intr 后中断活动

19.3.14 TXOICR(0x38)

域	位	读写	复位值	描述
reserved	31:1	RO	0x0	保留
TXOICR	0	RO	0x0	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。

19.3.15 RXOICR(0x3C)

域	位	读写	复位值	描述
reserved	31:1	RO	0x0	保留
RXOICR	0	RO	0x0	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。

19.3.16 RXUICR(0x40)

域	位	读写	复位值	描述
reserved	31:1	RO	0x0	保留
RXUICR	0	RO	0x0	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。

19.3.17 MSTICR(0x44)

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

19.3.18 ICR(0x48)

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

19.3.19 DMACR(0x4C)

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

19.3.20 DMATDLR(0x50)

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

19.3.21 DMARDLR(0x54)

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

19.3.22 IDR(0x58)

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

19.3.23 DR（0x60-0xEC）

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

19.3.24 RX_SAMPLE_DLY(0xFC)

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

20 I2S

I2S 控制器主要实现音频数据的发送与接收。

20.1 操作说明

20.1.1 Transmitter 模式

- 使能整个 I2S 控制器，即写 IER 寄存器为 1；
- 通过 APB 接口，向 LTHR 和 RTHR 两个寄存器中写入将要发送的数据，填满整个 FIFO（超过阈值，阈值通过 RFCR0 和 TFCR0 两个寄存器设置）；
- 设置 TCR0 寄存器，设定发送数据的分辨率；
- 设定中断使能开关；
- 设定 ITER 为 1，使能 I2S 控制器的发送；
- 设定 TER0 为 1，将 channel0 发送使能打开；
- 写 CER 寄存器，打开 I2S 时钟，开始工作。

20.1.2 Receiver 模式

- 使能整个 I2S 控制器，即写 IER 寄存器为 1；
- 设定 IRER 为 1，使能 I2S 控制器的发送功能；
- 设置 RCR0 寄存器，设定接收数据的分辨率；
- 设定中断使能开关；
- 设定 IRER 寄存器，开启接收功能；
- 设定 RER0 为 1，将 channel0 发送使能打开；
- 写 CER 寄存器，打开 I2S 时钟；
- 等待中断到来，即接收数据 FIFO 到达默认的 trigger 值；
- 通过 APB 接口，读取 LRBR 和 RRBR 两个寄存器，将收到的数据读出；
- 在 APB 向 FIFO 中填充需要发送的数据时候，若此时 FIFO 已经满了，则后续数据会丢失，FIFO 中原始数据保持；在 I2S 接收数据填充本地 FIFO 的时候，若 APB 没有及时读取导致接收 FIFO 满了，则后续接收的数据也丢失。

20.2 寄存器说明

寄存器名称	偏移	读写	复位值	描述
IER	0x000	RW	0x0	I2S Enable Register（控制使能全局）
IRER	0x004	RW	0x0	I2S Receiver Block Enable Register（控制接收模块使能）
ITER	0x008	RW	0x0	I2S Transmitter Block Enable Register（控制发送模块使能）
CER	0x00C	RW	0x0	Clock Enable Register（全局时钟使能控制）
CCR	0x010	RW	依赖于配置	Clock Configuration Register（配置时钟） [4:3]对应 WSS，控制 ws_out 持续周期： 0：16 sclk cycles；（sclk 为 i2s 传输数据的时钟） 1：24 sclk cycles； 2：32 sclk cycles
RXFFR	0x014	WO	0x0	Receiver Block FIFO Register（写 1 将清除所有接收相关的 FIFO）
TXRRF	0x018	WO	0x0	Transmitter Block FIFO Register（写 1 将清除所有发送相关的 FIFO）
LRBR0	0x020	RO	0x0	Left Receive Buffer 0（提供给上层接口的读取接收的左声道数据的寄存器）
LTHR0	0x020	WO	0x0	Left Transmit Holding Register 0（提供给上层接口的写入发送的左声道数据的寄存器）
RRBR0	0x024	RO	0x0	Right Receive Buffer 0（提供给上层接口的读取接收的右声道数据的寄存器）
RTHR0	0x024	WO	0x0	Right Transmit Holding Register 0（提供给上层接口的写入发送的右声道数据的寄存器）
RER0	0x028	RW	0x1	Receive Enable Register 0（接收通道使能控制位）
TER0	0x02C	RW	0x1	Transmit Enable Register 0（发送通道使能控制位）
RCR0	0x030	RW	依赖于配置	Receive Configuration Register 0（控制接收数据的 resolution，在控制信号使能情况下，APB 写该寄存器无效） 000：Ignore word length； 001：12 bit resolution； 010：16 bit resolution； 011：20 bit resolution； 100：24 bit resolution； 101：32 bit resolution
TCR0	0x034	RW	依赖于配置	Receive Configuration Register 0（控制发送数据的 resolution，在控制信号使能情况下，APB 写该寄存器无效） 000：Ignore word length；

				001：12 bit resolution； 010：16 bit resolution； 011：20 bit resolution； 100：24 bit resolution： 101：32 bit resolution
ISR0	0x038	RO	0x10	中断状态寄存器 [0:0]：接收数据有效中断； [1:1]：接收 FIFO 溢出中断； [4:4]：发送 FIFO 小于阈值中断； [5:5]：发送 FIFO 溢出中断。
IMR0	0x03C	RW	0x33	中断屏蔽寄存器 1：mask；0：unmask。 各位域与 ISR0 寄存器对应
ROR0	0x040	RO	0x0	Receive Overrun Register 0（读取该位，就会清除接收数据溢出中断，同时清除状态寄存器相关位）
TOR0	0x044	RO	0x0	Transmit Overrun Register 0（读取该位，就会清除发送数据溢出中断，同时清除状态寄存器相关位）
RFCR0	0x048	RW	依赖于配置	Receive FIFO Configuration Register 0（设置接收 FIFO 产生满中断的阈值，高于该阈值，触发中断；在控制信号使能情况下，APB 写该寄存器无效）
TFCR0	0x04C	RW	依赖于配置	Transmit FIFO Configuration Register 0（设置发送 FIFO 产生空中断的阈值，低于该阈值，触发中断；在控制信号都处于使能情况下，APB 写该寄存器无效）
RFF0	0x050	WO	0x0	Receive FIFO Flush 0（写 1 清除该通道接收 FIFO）
TFF0	0x054	WO	0x0	Transmit FIFO Flush 0（写 1 清除该通道发送 FIFO）
RXDMA	0x1C0	RO	0x0	DMA 模式时候，读取数据入口
RRXDMA	0x1C4	WO	0x0	复位接收 DMA 寄存器
TXDMA	0x1C8	WO	0x0	DMA 模式时候，发送数据入口
RTXDMA	0x1CC	WO	0x0	复位发送 DMA 寄存器
FDP	0xC00	RW	0x0	Fractional Division Parameter 产生 mclk（codec 内部工作时钟）和 sclk 的小数分频参数
EDP	0xC04	RW	0x0	Even Division Parameter 产生 sclk 的偶数分频参数

21 信号幅值调试寄存器

21.1 操作说明

1. cmn_diag_bias_ovrd 寄存器写 0x7700；
2. cmn_txpucal_tune 寄存器写 0x000d；
3. cmn_txpdcal_tune 寄存器写 0x000d。

通过配置上述寄存器，理论上可以对每条 lane 的电压幅值增加约 100mV，不同的项目以实测为准。

21.2 寄存器列表

USB 信号幅值调试寄存器基地址（8 port 相同）为：0x200_0000。

SATA 信号幅值调试寄存器基地址（4 port）分别为：0x1C0_0000，0x1C4_0000，0x1C8_0000，0x1CC_0000。

DP 信号幅值调试寄存器基地址 DP0 和 DP1 相同为：0x140_0000，DP2 为：0x1C8_0000。

寄存器	偏移	描述
cmn_diag_bias_ovrd	0x784	偏置覆盖寄存器
cmn_txpucal_tune	0x40C	TX 上拉电阻校准调节寄存器
cmn_txpdcal_tune	0x42C	TX 下拉电阻校准调节寄存器

21.3 寄存器说明

21.3.1 cmn_diag_bias_ovrd(0x784)

域	位	读写	复位值	描述
reserved	15	RO	0x0	保留
bias_rx_rescal	14:12	RW	0x3	接收电阻校准电流调节，其编码对应电流如下： 3’b000：110.72uA 3’b001：112.50uA 3’b010：114.29uA 3’b011：116.07uA 3’b100：117.86uA 3’b101：119.64uA 3’b110：121.43uA 3’b111：123.22uA
reserved	11	RO	0x0	保留
bias_tx_rescal	10:8	RW	0x3	发送电阻校准电流调节，其编码对应电流如下：

				3’b000：110.72uA 3’b001：112.50uA 3’b010：114.29uA 3’b011：116.07uA 3’b100：117.86uA 3’b101：119.64uA 3’b110：121.43uA 3’b111：123.22uA
reserved	7:4	RW	0x0	保留
reserved	3:1	RO	0x0	保留
reserved	0	RW	0x0	保留

21.3.2 cmn_txpucal_tune(0x40C)

域	位	读写	复位值	描述
reserved	15:7	RO	0x0	保留
txpucal	6:0	RW	0x0	TX 上拉电阻校准调节值

21.3.3 cmn_txpdcal_tune(0x42C)

域	位	读写	复位值	描述
reserved	15:7	RO	0x0	保留
txpdcal	6:0	RW	0x0	TX 下拉电阻校准调节值