

Q353333N1100 系列 SoC Wi-Fi、BLE 和 SLE Combo 芯片

用户指南

文档版本 05

发布日期 2025-02-07

前言

概述

本文档主要介绍 Q353333N1100 系列芯片的各项基本功能，用户提供 Q353333N1100 系列芯片的应用配置方法。

产品版本

与本文档相对应的产品版本如下。

产品名称	产品版本	描述
Q353333	N1100	常规版本芯片，支持 WiFi、BLE、SLE 等。
Q353333	N1100E	感知版本芯片，基于常规版本芯片，增加感知功能。

读者对象

本文档主要适用于以下工程师：

- 技术支持工程师
- 客户开发工程师

符号约定

在本文中可能出现下列标志，它们所代表的含义如下。

符号	说明
 危险	表示如不避免则将会导致死亡或严重伤害的具有高等级风险的危害。
 警告	表示如不避免则可能导致死亡或严重伤害的具有中等级风险的危害。
 注意	表示如不避免则可能导致轻微或中度伤害的具有低等级风险的危害。
须知	用于传递设备或环境安全警示信息。如不避免则可能会导致设备损坏、数据丢失、设备性能降低或其它不可预知的结果。 “须知”不涉及人身伤害。
 说明	对正文中重点信息的补充说明。 “说明”不是安全警示信息，不涉及人身、设备及环境伤害信息。

修改记录

文档版本	发布日期	修改说明
05	2025-02-07	- 更新 “1.1 概述” 小节内容。 - 更新 “4.3 WiFi PHY” 小节内容。 - 更新 “4.6 感知功能” 小节内容。 - 更新 “6.1.2 软用管脚描述” 小节内容。
04	2024-12-06	新增 “8 包装信息” 章节内容。
03	2024-09-26	更新 “4.1.3 RF 性能” 小节内容。
02	2024-08-28	更新 “2.3.3 上下电时序约束” 小节内容。

文档版本	发布日期	修改说明
		- 更新 “3.3.3 初始化流程” 小节内容。 - 更新 “6.5 SPI” 、 “6.8 I2S” 、 “6.9 QSPI” 和 “6.11 ADC” 章节的工作方式和寄存器概览及寄存器描述。
01	2024-04-10	- 更新 “2.3.3 上下电时序约束” 小节内容。 - 更新 “3.3.3 初始化流程” 小节内容。 - 更新 “6.5 SPI” 、 “6.8 I2S” 、 “6.9 QSPI” 和 “6.11 ADC” 章节的工作方式和寄存器概览及寄存器描述。
00B03	2024-02-22	- 更新 “2.4 处理器子系统” 小节内容。 - 更新 “3.2 功能描述” 中 “AHB Master 接口” 内容。 - 更新 “3.3.1 读写 Flash” 、 “3.3.2 其他操作” 、 “3.3.3 初始化流程” 和 “3.3.8 通过 DMA 方式读写 Flash 操作流程” 小节内容。 - 删除 “软件擦除 Flash 后回读数据操作流程” 小节内容。 - 更新 “4.1.1 概述” 、 “4.1.2 功能描述” 和 “4.1.3 RF 性能” 小节内容。 - 更新 “5.1.1 概述” 小节内容。 - 更新 “5.7.2 功能描述” 中 EFUSE 功能描述。 - 更新 “6.5.2 功能描述” 小节内容。 - 更新 “6.6.3 工作方式” 小节内容。 - 更新 “6.10.2 功能描述” 小节内容。
00B02	2023-12-15	- 更新 “1.1 概述” 小节内容。 - 更新 “4.4.2 功能描述” 小节内容。 - 更新 “4.5.2 功能描述” 小节内容。

文档版本	发布日期	修改说明
		- 更新 “6.9.2 功能描述” 小节内容。 - 更新 “6.11.2 功能描述” 小节内容。
00B01	2023-09-11	第一次临时版本发布。

目 录

前言 i

1 产品概述..... 1

1.1 概述..... 1

1.2 功能描述..... 2

1.2.1 功能特性..... 2

1.2.2 遵从的标准与协议 3

1.3 逻辑框图..... 4

1.4 应用场景..... 6

2 系统..... 7

2.1 复位..... 7

2.1.1 概述..... 7

2.1.2 复位控制..... 7

2.2 时钟..... 8

2.2.1 概述..... 8

2.2.2 时钟分配..... 9

2.2.3 时钟控制..... 9

2.3 电源管理与低功耗模式控制..... 9

2.3.1 概述..... 9

2.3.2 系统工作模式 10

2.3.3 上下电时序约束..... 10

2.4 处理器子系统 11

2.5 存储器空间映射..... 11

2.6 中断系统..... 14

2.6.1 中断分配.....	14
2.6.2 中断结构.....	16
2.6.3 寄存器概览.....	17
2.6.4 寄存器描述.....	17
2.7 RTC.....	21
2.7.1 概述.....	21
2.7.2 功能描述.....	21
2.8 Timer.....	22
2.8.1 概述.....	22
2.8.2 功能描述.....	22
2.9 看门狗.....	22
2.9.1 概述.....	22
2.9.2 功能描述.....	22
3 QSPI Flash 控制器	24
3.1 概述.....	24
3.2 功能描述.....	25
3.3 工作方式.....	26
3.3.1 读写 Flash	26
3.3.2 其他操作.....	26
3.3.3 初始化流程.....	27
3.3.4 通过寄存器方式读 Flash 操作流程.....	27
3.3.5 通过寄存器方式写 Flash 操作流程.....	28
3.3.6 通过寄存器方式其他操作流程.....	29
3.3.7 通过 AHB Slave 直接读写 Flash 操作流程.....	30
3.3.8 通过 DMA 方式读写 Flash 操作流程	30
3.4 寄存器概览.....	31
3.5 寄存器描述.....	33
4 WiFi /BLE & SLE 系统	53
4.1 WiFi / BLE & SLE RF.....	53
4.1.1 概述.....	53
4.1.2 功能描述.....	54

4.1.3 RF 性能.....	54
4.2 WiFi/BLE & SLE ABB	64
4.2.1 概述.....	64
4.2.2 功能描述.....	65
4.2.3 工作方式.....	65
4.3 WiFi PHY	66
4.3.1 概述.....	66
4.3.2 功能描述.....	66
4.3.3 工作方式.....	67
4.4 WiFi MAC.....	68
4.4.1 概述.....	68
4.4.2 功能描述.....	68
4.4.3 工作方式.....	69
4.4.3.1 AP 模式.....	69
4.4.3.2 STA 模式.....	69
4.4.3.3 Monitor 模式.....	69
4.4.3.4 AP 与 STA 共存.....	69
4.4.3.5 CSI 模式.....	69
4.5 BLE/SLE	70
4.5.1 概述.....	70
4.5.2 功能描述.....	70
4.5.3 工作方式.....	73
4.5.3.1 中断.....	73
4.5.3.2 加密.....	73
4.6 感知功能.....	74
4.6.1 概述.....	74
4.6.2 功能描述.....	74
4.6.3 工作方式.....	74
4.6.4 RF 性能.....	75
5 安全系统.....	76
5.1 安全子系统.....	76
5.1.1 概述.....	76

5.1.2 功能描述.....	76
5.2 对称加解密模块.....	77
5.2.1 概述.....	77
5.2.2 功能描述.....	77
5.3 HASH 模块.....	77
5.3.1 概述.....	77
5.3.2 功能描述.....	77
5.4 密钥派生模块	78
5.4.1 概述.....	78
5.4.2 功能描述.....	78
5.5 非对称公钥算法模块.....	78
5.5.1 概述.....	78
5.5.2 功能描述.....	78
5.6 随机数生成模块.....	79
5.6.1 概述.....	79
5.6.2 功能描述.....	79
5.7 EFUSE.....	79
5.7.1 概述.....	79
5.7.2 功能描述.....	79
6 外围设备.....	81
6.1 IO MUX.....	81
6.1.1 概述.....	81
6.1.2 软用管脚描述	81
6.1.3 寄存器概览.....	88
6.1.4 寄存器描述.....	90
6.2 GPIO.....	129
6.2.1 概述.....	129
6.2.2 功能描述.....	129
6.2.3 工作方式.....	130
6.2.3.1 初始化配置.....	130
6.2.3.2 边沿中断配置	130
6.2.3.3 电平中断配置	131

6.2.4 寄存器概览	132
6.2.5 寄存器描述	132
6.3 UART	140
6.3.1 概述	140
6.3.2 功能描述	140
6.3.3 工作方式	141
6.3.4 寄存器概览	144
6.3.5 寄存器描述	145
6.4 I2C	165
6.4.1 概述	165
6.4.2 功能描述	165
6.4.3 工作方式	165
6.4.3.1 不使用 FIFO	166
6.4.3.2 使用 FIFO	168
6.4.4 寄存器概览	170
6.4.5 寄存器描述	171
6.5 SPI	182
6.5.1 概述	182
6.5.2 功能描述	182
6.5.3 工作方式	182
6.5.3.1 接口信号	182
6.5.3.2 传输帧格式	183
6.5.3.3 工作模式	187
6.5.3.4 应用说明	189
6.5.4 寄存器概览	192
6.5.5 寄存器描述	193
6.6 PWM	222
6.6.1 概述	222
6.6.2 功能描述	222
6.6.3 工作方式	223
6.6.4 寄存器概览	223
6.6.5 寄存器描述	225
6.7 Tsensor	234

6.7.1 概述.....	234
6.7.2 功能描述.....	234
6.7.3 工作方式.....	235
6.7.3.1 正常检测温度模式	235
6.7.3.2 高低温门限中断模式	235
6.7.3.3 过温保护中断模式	236
6.7.4 寄存器概览.....	237
6.7.5 寄存器描述	238
6.8 I2S	246
6.8.1 概述.....	246
6.8.2 功能描述.....	246
6.8.3 工作方式.....	246
6.8.3.1 接口信号.....	246
6.8.3.2 操作模式.....	247
6.8.3.3 应用说明.....	248
6.8.4 寄存器概况	252
6.8.5 寄存器描述	254
6.8.6 注意事项.....	273
6.9 QSPI	273
6.9.1 概述.....	273
6.9.2 功能描述.....	274
6.9.3 工作方式.....	274
6.9.4 寄存器概览	276
6.9.5 寄存器描述	278
6.10 DMA.....	307
6.10.1 概述.....	307
6.10.2 功能描述.....	307
6.10.3 工作方式.....	308
6.10.4 寄存器概览.....	309
6.10.5 寄存器描述	311
6.11 ADC.....	337
6.11.1 概述	337

6.11.2 功能描述	338
6.11.3 工作方式	338
6.11.4 寄存器概览	340
6.11.5 寄存器描述	340
7 JTAG	351
7.1 概述	351
7.2 调试接口	351
8 包装信息	352
A 缩略语	357

1 产品概述

- 1.1 概述
- 1.2 功能描述
- 1.3 逻辑框图
- 1.4 应用场景

1.1 概述

Q353333N1100 系列芯片是一款高度集成的 2.4GHz SoC Wi-Fi、BLE 和 SLE 的 Combo 芯片，集成 IEEE 802.11b/g/n/ax 基带和 RF 电路，RF 电路包括功率放大器 PA、低噪声放大器 LNA、RF balun、天线开关以及电源管理等模块；支持 HT 20MHz/40MHz、HE 20MHz 标准带宽，提供最大 150Mbit/s 物理层速率。

Q353333N1100 系列芯片 Wi-Fi 基带支持正交频分多址（OFDMA）技术，正交频分复用（OFDM）技术，并向下兼容直接序列扩频（DSSS）和补码键控（CCK）技术，支持 IEEE 802.11b/g/n 协议的各种数据速率，支持 IEEE 802.11ax 协议的 MCS0~MCS9 速率。

Q353333N1100 系列芯片支持 BLE 1MHz/2MHz 频宽，支持 BLE 5.4 协议，支持 BLE Mesh 和 BLE 网关功能，最大空口速率 2Mbps。

Q353333N1100 系列芯片支持 SLE 1MHz/2MHz/4MHz 频宽，支持 SLE1.0 协议，支持 SLE 网关功能，最大空口速率 12Mbps。

Q353333N1100 系列芯片集成高性能 32bit 微处理器、硬件安全引擎以及丰富的外设接口，外设接口包括 SPI、QSPI、UART、I2C、PWM、GPIO 和多路 ADC；芯片内置 SRAM 和 Flash，可独立运行，并支持在 Flash 上运行程序。

Q353333N1100E 支持感知功能，智能感知房间内是否有人。

Q353333N1100 系列芯片支持 OpenHarmony 和第三方组件，并配套提供开放、易用的开发和调试运行环境。

Q353333N1100 系列芯片适应于智能家电等物联网智能终端领域。

1.2 功能描述

1.2.1 功能特性

Wi-Fi

- 1×1 2.4GHz 频段。
- PHY 支持 IEEE 802.11b/g/n/ax。
MAC 支持 IEEE 802.11d/e/i/k/v/r/w。
- 支持 802.11n 20MHz/40MHz 频宽，支持 802.11ax 20MHz 频宽。
- 支持最大速率：150Mbit/s@HT40 MCS7，114.7Mbit/s@HE20 MCS9。
- 内置 PA 和 LNA，集成 TX/RX Switch、Balun 等。
- 支持 STA 和 SoftAP 形态，作为 SoftAP 时最大支持 6 个 STA 接入。
- 支持 A-MPDU、A-MSDU。
- 支持 Block-ACK。
- 支持 QoS，满足不同业务服务质量需求。
- 支持 WPA/WPA2/WPA3 personal、WPS2.0、WAPI。
- 支持 RF 自校准方案。
- 支持 STBC。
- 支持 LDPC TX。
- 支持雷达感知功能（仅 Q353333N1100E 芯片支持）。

蓝牙

- 低功耗蓝牙 Bluetooth Low Energy (BLE)。
- 支持 BLE 5.4。
- 支持 125Kbit/s、500Kbit/s、1Mbit/s、2Mbit/s 速率。
- 支持多路广播。

- 支持 Class 1。
- 支持高功率 20dBm。
- 支持 BLE Mesh，支持 BLE 网关。

星闪

- 星闪低功耗接入技术 Sparklink Low Energy (SLE)。
- 支持 SLE 1.0。
- 支持 SLE 1MHz/2MHz/4MHz，最大空口速率 12Mbit/s。
- 支持 Polar 信道编码。
- 支持 SLE 网关。

CPU 子系统

- 高性能 32bit 微处理器，最大工作频率 240MHz。
- 内嵌 SRAM 606KB、ROM 300KB。
- 内嵌 4MB Flash。

外围接口

- 1 个 SPI 接口、1 个 QSPI 接口、2 个 I2C 接口、1 个 I2S 接口、3 个 UART 接口、19 个 GPIO 接口、6 路 ADC 输入、8 路 PWM（注：上述接口通过复用实现）。
- 外部晶体时钟频率 24MHz、40MHz。

其他信息

- 电源电压输入：典型值 3.3V/5V。
IO 电源电压支持 1.8V/3.3V，外接 MCU 和调试的 UART 支持 5V tolerant。
- 封装：QFN-40，5mm×5mm。
- 工作温度：-40°C ~ +85°C。

1.2.2 遵从的标准与协议

Q353333N1100 系列芯片支持以下标准协议：

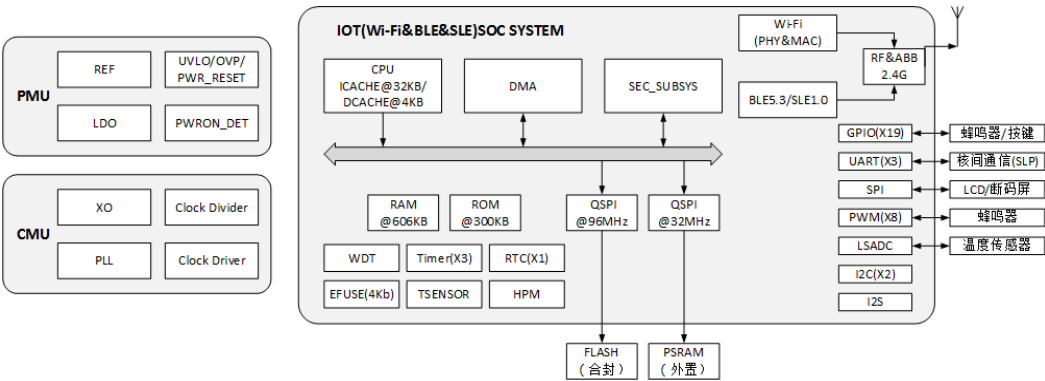
- 802.11-2020 Wireless LAN Medium Access Control (MAC) and Physical Layer (PHY) Specifications.
- Bluetooth Core Specification Version 5.4.

- Sparklink Wireless Communication Low Enery Air-interface Technical Requirements V01.00.

1.3 逻辑框图

Q353333N1100 系列芯片逻辑框图如图 1-1 所示。

图1-1 Q353333N1100 系列芯片逻辑框图



其中，图中的各模块功能描述如表 1-1 所示。

表1-1 模块功能描述

模块名	功能描述
PMU	电源管理单元。
REF	电压参考。
UVLO/OVP/PWR RESET	欠压/过压保护、电源复位。
LDO	低压差线性稳压器。
PWRON DET	上电检测。
CMU	时钟管理单元。
XO	晶体振荡器。
Clock Divider	时钟分频器。
PLL	锁相环。

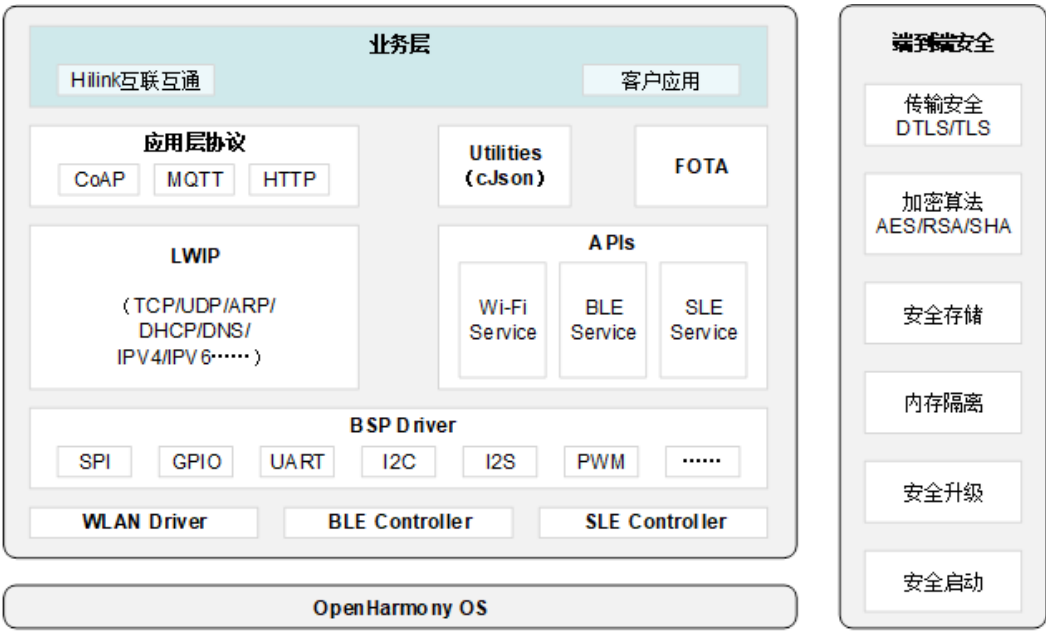
模块名	功能描述
Clock Driver	时钟驱动器。
CPU	中央处理单元。
DMA	直接存储器访问单元。
SEC SUBSYS	安全子系统。
RAM	随机存取存储器。
ROM	只读存储器。
QSPI	4 线 SPI。
WDT	看门狗单元。
Timer	定时器。
RTC	实时时钟单元。
EFUSE	加解密和芯片 ID 存储。
TSensor	温度传感器。
HPM	工艺监视单元。
Wi-Fi	WiFi 通信模块。
BLE	低功耗蓝牙通信模块。
SLE	SLE 通信模块。
PHY	信道调制、解调。
MAC	MAC 层业务处理。
RF&ABB	射频&模拟模块。
GPIO	通用输入输出接口。
UART	通用异步串行接口控制器。
SPI	串行外设接口控制器。
PWM	脉冲宽度调制单元。
LSADC	低速 ADC。

模块名	功能描述
I2C	集成电路互连总线控制器。
I2S	集成电路内置音频总线控制器。
FLASH	闪存。
PSRAM	伪静态存储器。

1.4 应用场景

Q353333N1100 系列芯片适应于智能家电等物联网智能终端领域，典型应用框图如图 1-2 所示。

图1-2 Q353333N1100 系列芯片典型应用框图



2 系统

- 2.1 复位
- 2.2 时钟
- 2.3 电源管理与低功耗模式控制
- 2.4 处理器子系统
- 2.5 存储器空间映射
- 2.6 中断系统
- 2.7 RTC
- 2.8 Timer
- 2.9 看门狗

2.1 复位

2.1.1 概述

复位模块根据输入复位源产生各个模块的复位信号，支持整个芯片的全局复位和各个模块的单独复位。

复位实现为异步复位，同步撤离。

2.1.2 复位控制

复位控制的输入信号分为：

- 全局复位控制。

- 单独模块复位控制。

芯片可以使用的复位控制方式如表 2-1 所示。

表2-1 复位控制

复位方式	来源	复位范围	复位后芯片模式
POR 复位	PMU POR 模块输出。	全芯片，并锁存硬件控制字	WORK 状态
Watch Dog 复位控制	Watch Dog 模块输出，软件可以屏蔽。	全芯片/模块可配置	WORK 状态
全局软复位控制	软件配置。	全芯片	WORK 状态
模块软复位控制	CRG (Clock and Reset Generator) 寄存器，软件配置。	模块	该模块处于复位状态，其余模块处于 WORK 状态

2.2 时钟

2.2.1 概述

时钟管理模块对芯片时钟输入、生成、控制进行统一的管理，其功能包括：

- 时钟输入的管理和控制。
- 时钟高频与低频的切换。
- 时钟分频和控制。
- 时钟的门控控制。
- 各模块工作时钟的生成。

2.2.2 时钟分配

正常工作时，各模块的时钟分配如表 2-2 所示。

表2-2 模块时钟分配

模块名称	时钟频率 (MHz)	模块名称	时钟频率 (MHz)
CPU	240	WDT (WatchDog)	晶体分频
CPU Bus	240	RTC (Real-Time Clock)	0.032
Timer	晶体分频	UART	160
GPIO	120	EFUSE	晶体
WiFi MAC	120	BT MAC	32
WiFi PHY	320	BT PHY	32
晶体	40/24	PMU OSC	0.032
I2C	80	SPI	160
QSPI	64	I2S	8.192 等

2.2.3 时钟控制

时钟管理模块主体包括：

- PLL 频率控制。
- 时钟分频和时钟源选择控制。
- 时钟门控管理。

2.3 电源管理与低功耗模式控制

2.3.1 概述

芯片的低功耗模式用于有效减少芯片的功耗，芯片提供多种低功耗的控制来动态降低芯片的功耗：

- 系统工作模式控制
除了 Work 模式之外，各种模式对功耗都有一定的减少作用，可以根据实际的功耗要求和功能要求选择不同的工作模式。
- 时钟门控和时钟频率调整
 - 提供时钟关断功能，可以关闭没有必要的时钟，减少芯片的功耗。
 - 系统工作的时钟频率可以进行调整，在满足功能的情况下可以调节时钟频率，动态降低芯片功耗。
- 模块级低功耗控制
提供模块级的低功耗控制，可以在某模块不工作的情况下，关断该模块或使模块处于低功耗状态，以减少芯片的功耗。

2.3.2 系统工作模式

系统工作模式分为以下三种模式：

- Work
正常工作状态，所有电源供电均打开，完成正常的 WiFi 收发等业务。
- Light Sleep
浅睡模式为可快速恢复业务收发的睡眠模式，关闭收发时钟以降低功耗。此时 CPU 配置为 WFI (Wait For Interrupt) 模式，等待中断唤醒后恢复收发，Flash、IO 保持供电，系统内发生任意中断均可唤醒 CPU（如定时中断，外设通信中断，GPIO 中断等）；SoC 系统可选晶体或 PLL 时钟，时钟频率和外设的通信速率有关。
- Shutdown
关机模式，芯片通过 PWR_ON 管脚拉低之后进入 Shutdown 模式，整芯片下电。当 PWR_ON 管脚拉高后退出 Shutdown 模式，进入到 Work 模式。

2.3.3 上下电时序约束

下图与下表中出现的硬件控制字 IO/VBAT/POWER_ON/AVDD33/DVDD3318 均为芯片对外的管脚。

表2-3 POWER_ON 管脚

信号名称	含义	复位时间	I/O
POWER_ON	芯片上下电管脚： 1. 拉高 POWER_ON 管	1. 上电要求：POWER_ON 拉高时间要求超过 20ms。	I

信号名称	含义	复位时间	I/O
	脚，芯片完成上电。 2. 拉低 POWER_ON 管脚，芯片完成复位，并且下电。	2. 下电要求：POWER_ON 拉低时间要求超过 5ms（推荐 20ms）。	

2.4 处理器子系统

系统提供一个自研 RISC-V 处理器作为主控 CPU，完成各种系统任务和控制工作。处理器带有 32KB 指令 Cache、4KB 数据 Cache。

该芯片 CPU 具有以下功能特点：

- 处理器的工作频率最高可达 240MHz。
- 支持直接模式和向量模式的中断方式，支持 1 个 nmi 中断，以及 64 个非标准外部中断。
- 支持 Flash Patch 功能，支持 192 个指令比较器和 2 个地址比较器。
- 支持边沿和电平两种中断触发方式。
- 支持 PMP（Physical Memory Protection）功能。
- 支持 JTAG 和 SWD（Serial Wire Debug）调试接口。

2.5 存储器空间映射

地址空间映射如表 2-4 所示。

表2-4 存储器地址空间映射

Slave Name	地址范围 (Start)	地址范围 (End)
CPU_ITCM	0x0010_0000	0x0017_FFFF
CPU_DTCM	0x0018_0000	0x001C_7FFF
-reserved（读取返回 dead_beef）	0x001C_8000	0x001F_FFFF

Slave Name	地址范围 (Start)	地址范围 (End)
NOR_FLASH (读取范围超过 Flash 大小时, 地址卷绕读取)	0x0020_0000	0x009F_FFFF
SHARE_RAM	0x00A0_0000	0x00A9_8FFF
-reserved (读取返回 resp_error)	0x00A9_9000	0x00BF_FFFF
PKE_ROM	0x00C0_0000	0x00C0_0BFF
-reserved (读取返回随机值)	0x00C0_0C00	0x00C0_0FFF
-reserved (读取返回 resp_error)	0x00C0_1000	0x3FFF_FFFF
SYS_CTL0	0x4000_0000	0x4000_3FFF
-reserved (读取返回 dead_dead)	0x4000_4000	0x4000_4FFF
RTC	0x4000_5000	0x4000_50FF
-reserved (读取返回 dead_dead)	0x4000_5100	0x4000_5FFF
WDT	0x4000_6000	0x4000_6FFF
-reserved (读取返回 dead_dead)	0x4000_7000	0x4000_FFFF
-reserved (读取返回 resp_error)	0x4001_0000	0x43FF_FFFF
SYS_CTL1	0x4400_0000	0x4400_1FFF
TIMER	0x4400_2000	0x4400_2FFF
-reserved (读取返回 dead_dead)	0x4400_3000	0x4400_3FFF
SYS_CTL2	0x4400_4000	0x4400_7FFF
EFUSE_CTL	0x4400_8000	0x4400_BFFF
LSADC_CTL	0x4400_C000	0x4400_CFFF
IO_CONFIG	0x4400_D000	0x4400_DFFF
TSensor_CTL	0x4400_E000	0x4400_EFFF
-reserved (读取返回 dead_dead)	0x4400_F000	0x4400_FFFF
UART0	0x4401_0000	0x4401_0FFF
UART1	0x4401_1000	0x4401_1FFF
UART2	0x4401_2000	0x4401_2FFF
-reserved (读取返回 dead_dead)	0x4401_3000	0x4401_7FFF

Slave Name	地址范围 (Start)	地址范围 (End)
I2C0	0x4401_8000	0x4401_80FF
I2C1	0x4401_8100	0x4401_81FF
-reserved (读取返回 dead_dead)	0x4401_8200	0x4401_FFFF
SPI	0x4402_0000	0x4402_00FF
-reserved (读取返回 dead_dead)	0x4402_0100	0x4402_0FFF
QSPI	0x4402_1000	0x4402_10FF
-reserved (读取返回 dead_dead)	0x4402_1100	0x4402_3FFF
PWM0~7	0x4402_4000	0x4402_4FFF
I2S	0x4402_5000	0x4402_50FF
-reserved (读取返回 dead_dead)	0x4402_5100	0x4402_7FFF
GPIO0~7	0x4402_8000	0x4402_8FFF
GPIO8~15	0x4402_9000	0x4402_9FFF
GPIO16~18	0x4402_A000	0x4402_AFFF
-reserved (读取返回 dead_dead)	0x4402_B000	0x440F_FFFF
SEC_APB	0x4410_0000	0x4411_4FFF
-reserved (读取返回 dead_dead)	0x4411_5000	0x4411_FFFF
WIFI_SUB	0x4421_0000	0x4421_3FFF
-reserved (读取返回 resp_error)	0x4421_4000	0x47FF_FFFF
SFC_CFG	0x4800_0000	0x4800_1FFF
-reserved (读取返回 resp_error)	0x4800_2000	0x48FF_FFFF
BSLE_SUB	0x4900_0000	0x4903_FFFF
-reserved (读取返回 resp_error)	0x4904_0000	0x49FF_FFFF
DMA_CFG	0x4A00_0000	0x4A00_0FFF

2.6 中断系统

2.6.1 中断分配

芯片支持向量模式和直接模式的中断方式，支持边沿和电平两种中断触发方式。支持优先级可编程，优先级配置寄存器（共 3bit）可配置 7 级的优先级。

中断系统包括：

- CPU 的内部标准中断：中断编号 0 ~ 25。
- CPU 外部的非标准中断：所支持的非标准中断编号如表 2-5 所示。

表2-5 非标准中断编号列表

Int No.	Int Name	Description
nmi	TEE_NMI_INT	nmi 软中断&WDT 全局中断。
26	TIMER_INT[0]	Timer[0]的全局中断。
27	TIMER_INT[1]	Timer[1]的全局中断。
28	TIMER_INT[2]	Timer[2]的全局中断。
29	RTC_IRQ	RTC 的全局中断。
30	Reserved	保留。
31	I2C0_INT	I2C0 中断。
32	I2C1_INT	I2C1 中断。
33	GPIO_INT[0]	GPIO[7:0]上报的组合中断。
34	GPIO_INT[1]	GPIO[15:8]上报的组合中断。
35	GPIO_INT[2]	GPIO[23:16]上报的组合中断。
36	SOFT_INT[0]	CPU 软中断 0。
37	SOFT_INT[1]	CPU 软中断 1。
38	SOFT_INT[2]	CPU 软中断 2。
39	SOFT_INT[3]	CPU 软中断 3。

Int No.	Int Name	Description
40	COEX_WL_INT	共存 WLAN 软中断。
41	COEX_BT_INT	共存 BT 软中断。
42	COEX_WIFI_RESUME_INT	共存 WLAN 恢复射频中断。
43	SPI_INT	SPI 中断。
44	WLPHY_INT	WLAN PHY 中断。
45	WLMAC_INT	WLAN MAC 中断。
46	BLE_INT	BLE 中断。
47	SLE_INT	SLE 中断。
48	TSensor_INT	TSensor 中断。
49	PMU_CMU_ERR_INT	PMU/CMU 异常中断。
50	DIAG_INT	维测中断。
51	I2S_INT	I2S 中断。
52	QSPI_INT	QSPI 中断。
53	UART0_INT	UART0 中断。
54	UART1_INT	UART1 中断。
55	UART2_INT	UART2 中断。
56	PWM_ABNOR_INT	PWM_ABNOR 中断。
57	PWM_CFG_INT	PWM_CFG 中断。
58	SFC_INT	SFC 中断。
59	DMA_INT	DMA 中断。
60	TIMER_ABNOR_INT	TIMER[2:0]异常中断。
61	I2S_TX_INT	I2S_TX 中断。
62	I2S_RX_INT	I2S_RX 中断。
63	PKE_REE_INT	PKE REE 中断。

Int No.	Int Name	Description
64	SPACC_REE_INT	SPACC REE 中断。
65	RKP_REE_INT	RKP REE 中断。
66	KLAD_REE_INT	KLAD REE 中断。
67	GLP_UART_RX_WAKE_INT	GLP UART RX 唤醒中断。
68	TIMING_GEN_INT	TIMING_GEN 中断。
69	MAC_MONITOR_INT	MAC 维测中断。
70	MEM_MONITOR_INT	MEM 维测中断。
71	TCM_MONITOR_INT	TCM 维测中断。
72	LSADC_INTR	LSADC 中断。
89~73	Reserved	保留。

2.6.2 中断结构

芯片使用 CPU 内部集成的中断控制器，所有的外设或寄存器触发的非标准中断均直接连至 CPU 内部进行处理。

本节主要介绍非 IP 中断，各个 IP 的中断说明请参见各 IP 对应章节。

NMI 中断

须知

TEE_NMI_INT[tee_nmi_int]不会自动清零，在触发中断后需要将该值写回 0。

NMI（Non-Maskable Interrupt）中断用于控制软件，由 WDOG 中断和寄存器触发，通过使能 WDOG 或写 TEE_NMI_INT[tee_nmi_int]为 1 触发 NMI 中断。

软中断

软中断为通过配置寄存器触发中断的中断触发方式，包含 4 个可以通过写寄存器触发的中断（软中断 0 ~ 软中断 3）。

软中断 0 ~ 软中断 3 的触发方式相同，以使用软中断 0 为例，配置步骤如下：

步骤 1 写 SOFT_INT_EN[soft_int0_en]为 1，打开软中断 0 的使能。

步骤 2 写 SOFT_INT_SET[soft_int0_set]为 1，将软中断 0 置位。

此寄存器为自清零寄存器，写 1 后自动回零，写 0 无效。

步骤 3 软件进入软中断 0 处理程序，此时可读取 SOFT_INT_STS[soft_int0_sts]查询中断状态：

- 0：无中断。
- 1：有中断。

步骤 4 写 SOFT_INT_CLR[soft_int0_clr]为 1，清除软中断 0。

此寄存器为自清零寄存器，写 1 后自动回零，写 0 无效。

----结束

2.6.3 寄存器概览

中断系统寄存器概览如表 2-6 所示。

表2-6 中断系统寄存器概览（基址是 0x44000000）

Offset	Address	Register	Description
0x0040		NMI_INT	WDT 中断查询&NMI 中断配置寄存器。
0x0150		SOFT_INT_EN	CPU 软中断使能寄存器。
0x0154		SOFT_INT_SET	CPU 软中断置位寄存器。
0x0158		SOFT_INT_CLR	CPU 软中断清除寄存器。
0x015C		SOFT_INT_STS	CPU 软中断查询寄存器。

2.6.4 寄存器描述

NMI_INT

NMI_INT 为 WDT 中断查询&NMI 中断配置寄存器。

Offset Address: 0x0040 Total Reset Value: 0x00000000

Bits	Access	Name	Description	Reset
31:2	RO	reserved	保留。	0x00000000
1	RO	wdt_int	看门狗中断查询寄存器。 0: WDT 中断无效; 1: WDT 中断有效。	0x0
0	RW	tee_nmi_int	软控 nmi 中断配置寄存器。 0: 拉低 CPU NMI 软中断; 1: 拉高 CPU NMI 软中断。	0x0

SOFT_INT_EN

SOFT_INT_EN 为 CPU 软中断使能寄存器。

Offset Address: 0x0150 Total Reset Value: 0x00000000

Bits	Access	Name	Description	Reset
31:4	RO	reserved	保留。	0x00000000
3	RW	soft_int_3_en	CPU 软中断 3 使能开关寄存器。 0: 使能无效; 1: 使能有效。	0x0
2	RW	soft_int_2_en	CPU 软中断 2 使能开关寄存器。 0: 使能无效; 1: 使能有效。	0x0
1	RW	soft_int_1_en	CPU 软中断 1 使能开关寄存器。 0: 使能无效; 1: 使能有效。	0x0
0	RW	soft_int	CPU 软中断 0 使能开关寄存	0x0

Bits	Access	Name	Description	Reset
		0_en	器。 0：使能无效； 1：使能有效。	

SOFT_INT_SET

SOFT_INT_SETCPU 软中断置位寄存器。

Offset Address: 0x0154 Total Reset Value: 0x00000000

Bits	Access	Name	Description	Reset
31:4	RO	reserved	保留。	0x00000000
3	W1_PU LSE	soft_int3_set	CPU 软中断 3 置位配置寄存器。 0：置位无效； 1：置位有效。	0x0
2	W1_PU LSE	soft_int2_set	CPU 软中断 2 置位配置寄存器。 0：置位无效； 1：置位有效。	0x0
1	W1_PU LSE	soft_int1_set	CPU 软中断 1 置位配置寄存器。 0：置位无效； 1：置位有效。	0x0
0	W1_PU LSE	soft_int0_set	CPU 软中断 0 置位配置寄存器。 0：置位无效； 1：置位有效。	0x0

SOFT_INT_CLR

SOFT_INT_CLR 为 CPU 软中断清除寄存器。

Offset Address: 0x0158 Total Reset Value: 0x00000000

Bits	Access	Name	Description	Reset
31:4	RO	reserved	保留。	0x00000000
3	W1_PULSE	soft_int3_clr	CPU 软中断 3 清除配置寄存器。 0: 清除无效; 1: 清除有效。	0x0
2	W1_PULSE	soft_int2_clr	CPU 软中断 2 清除配置寄存器。 0: 清除无效; 1: 清除有效。	0x0
1	W1_PULSE	soft_int1_clr	CPU 软中断 1 清除配置寄存器。 0: 清除无效; 1: 清除有效。	0x0
0	W1_PULSE	soft_int0_clr	CPU 软中断 0 清除配置寄存器。 0: 清除无效; 1: 清除有效。	0x0

SOFT_INT_STS

SOFT_INT_STS 为 CPU 软中断查询寄存器。

Offset Address: 0x015C Total Reset Value: 0x00000000

Bits	Access	Name	Description	Reset
31:4	RO	reserved	保留。	0x00000000
3	RO	soft_int	CPU 软中断 3 状态查询寄存	0x0

Bits	Access	Name	Description	Reset
		3_sts	器。 0：中断无效； 1：中断有效。	
2	RO	soft_int 2_sts	CPU 软中断 2 状态查询寄存器。 0：中断无效； 1：中断有效。	0x0
1	RO	soft_int 1_sts	CPU 软中断 1 状态查询寄存器。 0：中断无效； 1：中断有效。	0x0
0	RO	soft_int 0_sts	CPU 软中断 0 状态查询寄存器。 0：中断无效； 1：中断有效。	0x0

2.7 RTC

2.7.1 概述

RTC 的功能主要是实现定时、计数功能，可以供操作系统用作系统时钟，也可以供应用程序用作定时和计数。

2.7.2 功能描述

RTC 具有以下功能特点：

- 48bit 位宽的 free running 递加计数器。
- 计数器上电解复位后即开始计数，无需任何使能配置。
- 计数时钟为 32kHz 时钟。
- 支持配置中断产生寄存器阈值，计数器递加到阈值时产生中断。

- 支持计数器值实时读取。

2.8 Timer

2.8.1 概述

芯片中有 3 个相同且可独立配置的定时器 (Timer)，主要实现定时、计数功能的 IP，可供程序用作定时和计数。

2.8.2 功能描述

Timer 具有以下功能特点：

- 3 个 32bit 的可独立配置的 Timer 单元。
- 每个 Timer 单元支持三种工作模式：
 - one-shot 模式：加载配置值到计数器，进行递减计数，递减到 0 后停止计数。
 - periodic 模式：Timer 持续计数，加载配置值到计数器，进行递减计数，递减到 0 后再次载入配置值并继续递减计数。
 - free running 模式：Timer 持续计数，计数器起始值为 0xFFFF_FFFF，进行递减计数，递减到 0 后再次载入 0xFFFF_FFFF 并继续递减计数。
- 每个 Timer 单元支持独立使能。

2.9 看门狗

2.9.1 概述

WatchDog 用于系统异常恢复，如果未得到更新则隔一定时间（可编程）产生一个系统复位信号，当 WatchDog 在此之前关闭工作时钟或更新计数器，复位信号不会产生。

2.9.2 功能描述

WatchDog 内置 1 个可编程 32bit 计数器，具有以下功能特点：

- 内置计数器进行递减计数，当由预设值递减到 0 时产生超时。
- WatchDog 具有 2 种工作方式。
 - 方式一：如果超时，则只产生系统复位。

- 方式二：第一次超时，WatchDog 产生中断；第二次超时，如果中断未被清除，WatchDog 产生系统复位。
- 可配置超时间隔。

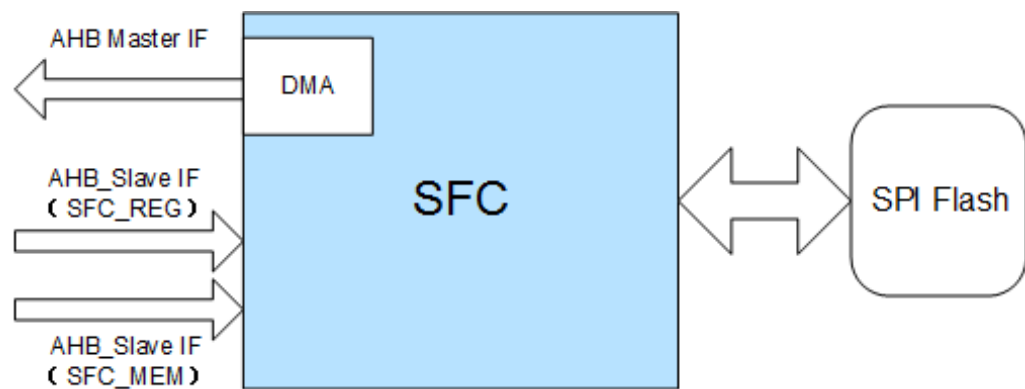
3 QSPI Flash 控制器

- 3.1 概述
- 3.2 功能描述
- 3.3 工作方式
- 3.4 寄存器概览
- 3.5 寄存器描述

3.1 概述

SFC 是一个 SPI Flash 控制器。业务侧提供一个 AHB（Advanced High Performance Bus）Slave 接口，主要完成 AHB 通道对 SPI Flash 的访问控制功能；提供一个 AHB Master 接口，用于 DMA 方式读写 Flash。

图3-1 SFC 应用框图



注：IF (Interface)。

3.2 功能描述

AHB Slave 接口

AHB Slave 接口具有以下特点：

- 提供一个 AHB Slave 接口，可以根据不同的选择信号访问内部配置寄存器或直接访问 SPI Flash Memory。
- 支持 AMBA2.0 协议。
- 仅支持小端 (Little-Endian)。

AHB Master 接口

AHB Master 接口具有以下特点：

- 提供一个 AHB Master 接口，用于 DMA 方式在内存和 Flash 之间搬运数据。
- 支持 AMBA2.0 协议。
- 只支持小端。
- 支持 Single、INCR、INCR4、INCR8、INCR16 传输类型。
- 不支持 Early Termination。
- 支持总线 Lock 传输。

存储器接口

存储器接口具有以下特点：

- 片选的存储空间最大支持到 64Mbit (3Byte 地址模式)。片选映射基地址可配置。只支持片选 1，不支持片选 0。
- 支持 Standard SPI、Dual-Output/Dual-Input SPI、Quad-Output/Quad-Input SPI、Dual-I/O SPI、Quad-I/O SPI 五种接口类型。上电后默认支持 Standard SPI 接口类型，可通过寄存器配置切换接口类型。
- 支持 XIP (Executed In Place)。
- SPI Flash 读写操作支持总线直接读写、寄存器编程读写、DMA 读写三种方式。
- 支持多种写保护操作。

- SFC 模块支持 SPI Mode0 和 Mode3，按协议要求，支持 SPI Mode0 和 Mode3 的 SPI Flash 器件在时钟的上升沿采样数据，在时钟的下降沿输出数据。
- XIP 实现地址 remap，为了软件只编译一个 XIP 镜像，升级镜像时主镜像和备份镜像使用相同地址。

Flash 数据在线解密

Flash 数据在线解密具有以下特点：

- 解密算法为 AES-128-CTR，仅支持 1 个 IV，密钥来源于 KM 派生。
- 支持 4 个解密区域，配置粒度为 256Byte；每个区域支持单独的 IV 解密起始地址可配，配置粒度同为 256Byte，4 个解密区域配置不能存在交叉地址。

说明

支持 AES 在线解密，解密时读取数据量没有 16Byte 的倍数与对齐的约束。

3.3 工作方式

3.3.1 读写 Flash

有三种方式读写 Flash：

- 通过寄存器配置方式发送 SPI Flash Program、Read 等命令来读写 Flash。例如：对寄存器 CMD_CONFIG 写 0x0000_7F8B，对寄存器 CMD_INS 写 0x03，表示通过 Standard SPI 方式发起读 64Byte Flash 数据的操作。

此方式直接控制需要发送的 Flash 命令。

- 通过 AHB Slave 接口以类似读写普通 Memory 的方式读写 Flash。
SFC 模块会自动将 AHB 总线的读写操作时序映射为 SPI Flash 读写命令。
- 通过 DMA 方式在 Flash 和外部 Memory 之间搬移数据。

3.3.2 其他操作

对 Flash 的其他操作如 Erase、进入 Deep Power Down、读 Device ID 等必须通过寄存器访问来实现。需要配置 CMD_INS[REG_INS]为相应的命令，具体请参见 Flash 器件手册。

例如：对寄存器 CMD_CONFIG 写 0x0000_0583，对寄存器 CMD_INS 写 0x0000_009F，表示读器件 ID 的操作。

3.3.3 初始化流程

须知

注意以下初始化流程仅做参考，请根据器件差异进行调整。

初始化流程如下：

步骤 1 进行时钟初始化配置，打开 SFC 需要的工作时钟 192M。

步骤 2 参考 IO_MUX，配置 IO 的驱动能力。

步骤 3（如果需要调整 Timing 参数）配置 TIMING 寄存器。

步骤 4 配置总线操作方式寄存器。

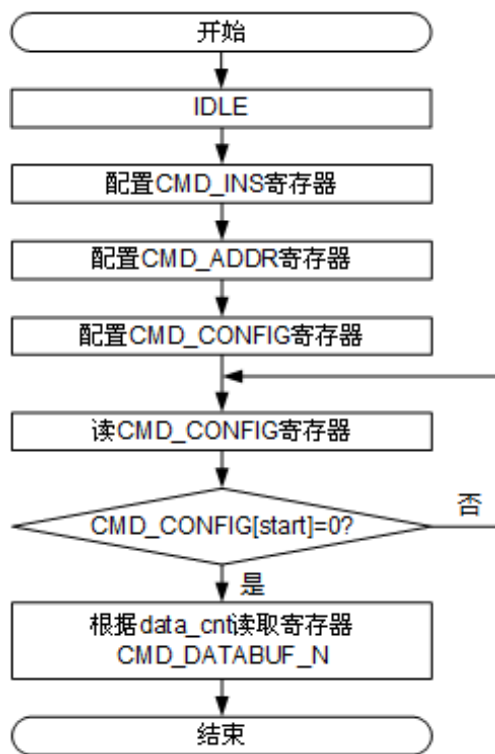
- 根据实际 Flash 大小配置 BUS_FLASH_SIZE[flash_size_cs1]（直接获知器件大小或可通过发 Read ID 命令给 Flash 查询获得）。
- 有些器件要求进入非 Standard SPI 读写时序，需要预先以特殊命令配置 Flash。根据器件需要，对寄存器 CMD_INS 进行写操作，发特定命令配置 Flash。
- 通过 BUS_CONFIG1/BUS_CONFIG2 配置总线读写操作指令和参数。
例如：对寄存器 BUS_CONFIG1 写 0xCC85_EB1E 表示配置的参数为写指令 32h，写方式为 Quad-Input SPI，读指令 EBh、读方式为 Quad I/O SPI。
- 如果需要开启总线写操作，配置 BUS_CONFIG1[wr_enable]为 1，使能总线写。默认关闭总线写功能。

----结束

3.3.4 通过寄存器方式读 Flash 操作流程

通过寄存器读取 Flash 的操作流程（查询方式），如图 3-2 所示。

图3-2 通过寄存器读取 Flash 的操作流程（查询方式）



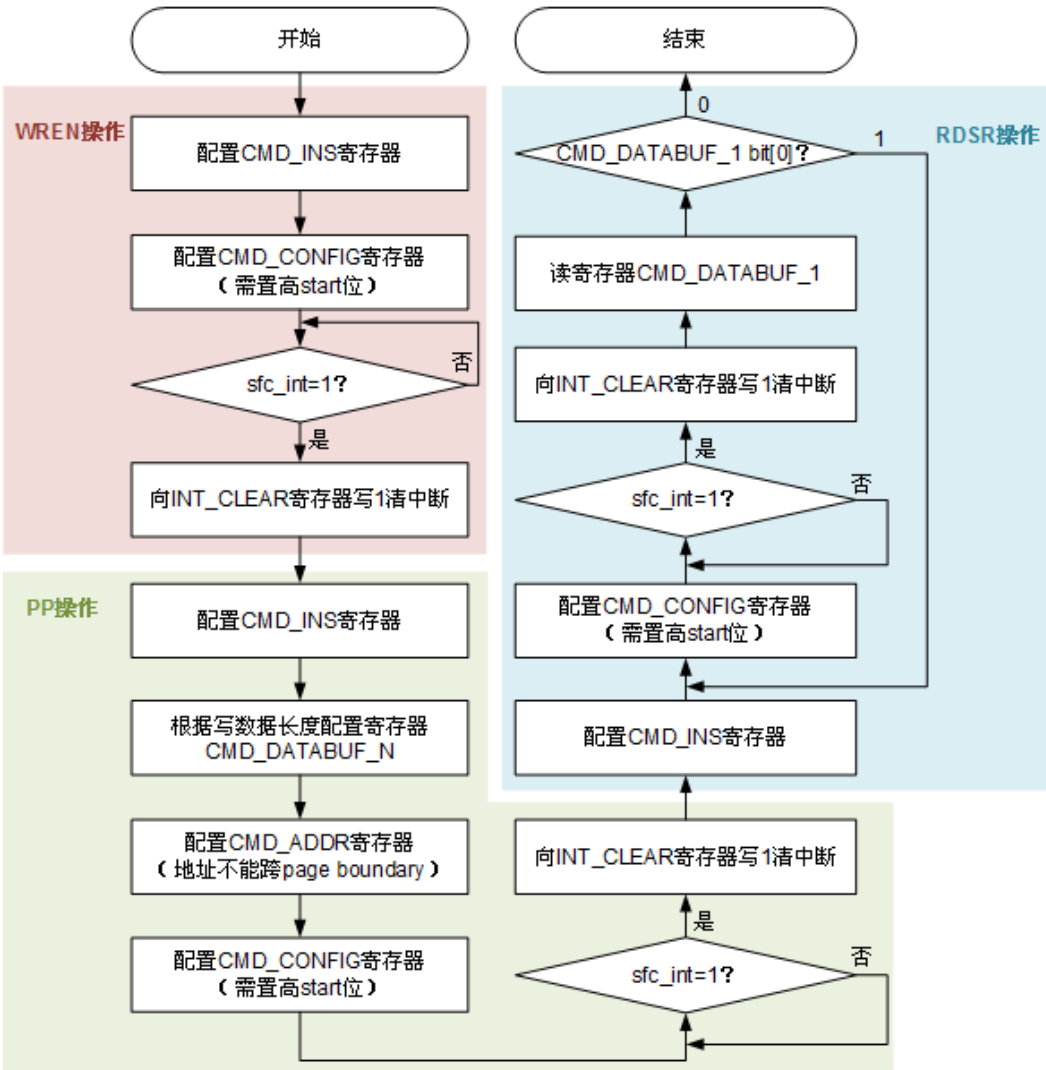
3.3.5 通过寄存器方式写 Flash 操作流程

须知

- 通过寄存器方式写 Flash 数据时，总线和 DMA 不得访问 Flash。
- 单次写 Flash 不能跨越 Page 边界（寄存器写方式没有跨越 Page 边界保护，需要软件保证，如果跨越 256Byte 边界，将会 Wrap 到该 Page 的起始地址，覆盖原来的内容）。

通过寄存器写 Flash 的操作流程（中断方式），如图 3-3 所示。

图3-3 通过寄存器写 Flash 的操作流程（中断方式）

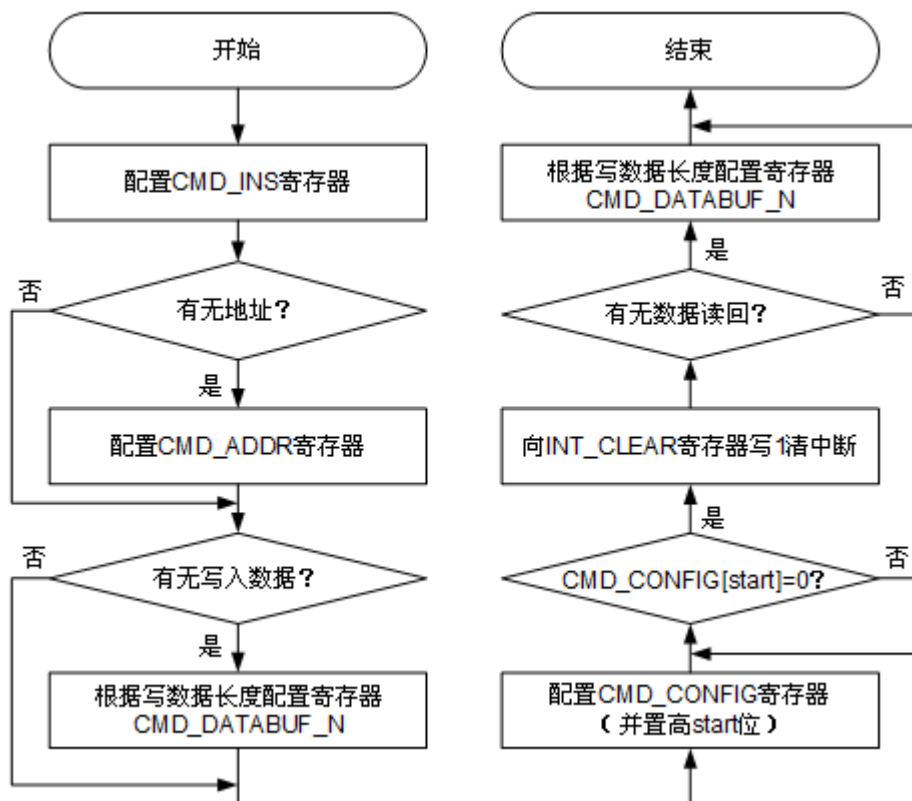


注：WREN (Write Read Enable)，PP (Page Program)，RDSR (Read Status Register)。

3.3.6 通过寄存器方式其他操作流程

通过寄存器方式其他操作流程如图 3-4 所示。

图3-4 通过寄存器方式其他操作流程



说明

SFC 控制器不支持发出“OPCODE (1byte) + DUMMY (3byte 全 0)”组合 SPI 时序，某些 Flash 指令需要这种组合时序时，可以采用“OPCODE (1byte) + ADDR (3byte 全 0)”组合代替。

3.3.7 通过 AHB Slave 直接读写 Flash 操作流程

上电复位后，默认配置为 Standard SPI 时序模式。不需要额外配置，可直接读 Flash。

默认通过 AHB Slave 写 Flash 是禁止的。需要配置 BUS_CONFIG1[wr_enable]为 1，使能总线写操作。

如果需要调整默认配置，请参见“3.3.3 初始化流程”。

3.3.8 通过 DMA 方式读写 Flash 操作流程

DMA 操作流程如下：

步骤 1 如需调整总线操作方式时序配置，请参见“3.3.3 初始化流程”。

- 步骤 2 写 `BUS_DMA_MEM_SADDR`，配置 DMA 操作的内存端起始地址；写 `BUS_DMA_FLASH_SADDR`，配置 Flash 端起始地址（Flash 偏移地址）；写 `BUS_DMA_LEN`，配置数据长度。
- 步骤 3 写 `BUS_DMA_CTRL`，配置读写方向，选择 Flash 片选 1。
- 步骤 4 写 `BUS_DMA_CTRL[start]` 为 1，使能 DMA 操作。
- 步骤 5 等待 `dma_done` 中断触发（中断方式）或轮询 DMA 操作完成（`BUS_DMA_CTRL[start]` 变为 0）。

 说明

- DMA 操作时可以同时 Flash 寄存器读命令操作。
- DMA 操作时可以同时通过 AHB Slave 直接访问 Flash，但需保证中间不修改总线操作相关配置。
- DMA 操作时需要保证首地址 4Byte 对齐。

----结束

3.4 寄存器概览

SFC 寄存器概览如表 3-1 所示。

表3-1 SFC 寄存器概览（基址是 0x4800_0000）

偏移地址	名称	描述
0x0100	<code>GLOBAL_CONFIG</code>	全局配置寄存器。
0x0110	<code>TIMING</code>	Timing 配置寄存器。
0x0120	<code>INT_RAW_STATUS</code>	中断原始状态寄存器。
0x0124	<code>INT_STATUS</code>	经过屏蔽处理的中断状态寄存器。
0x0128	<code>INT_MASK</code>	中断屏蔽寄存器。
0x012C	<code>INT_CLEAR</code>	中断清除寄存器。
0x0130	<code>SOFT_RST_MASK</code>	软复位寄存器屏蔽位。
0x0200	<code>BUS_CONFIG1</code>	总线操作方式配置 1 寄存器。
0x0204	<code>BUS_CONFIG2</code>	总线操作方式配置 2 寄存器。

偏移地址	名称	描述
0x0240	BUS_DMA_CTRL	DMA 操作控制寄存器。
0x0244	BUS_DMA_MEM_SADDR	DMA 操作 DDR 起始地址寄存器。
0x0248	BUS_DMA_FLASH_SADDR	DMA 操作 Flash 起始地址寄存器。
0x024C	BUS_DMA_LEN	DMA 操作搬运数据长度寄存器。
0x0250	BUS_DMA_AHB_CTRL	DMA 操作 AHB 时 burst 操作方式选择控制寄存器。
0x0300	CMD_CONFIG	命令操作方式配置寄存器。
0x0308	CMD_INS	命令操作方式指令寄存器。
0x030C	CMD_ADDR	命令操作方式地址寄存器。
0x0400 + 4 × n	CMD_DATABUF_N	命令操作方式数据 Buffer 寄存器。
0x1000 + 4 × n	APC_CFG_START_ADDR	FAPC 鉴权。
0x1040 + 4 × n	APC_CFG_END_ADDR	FAPC 鉴权。
0x1180	SFC_FAPC_DEC_AUTH_CFG	FAPC 鉴权。
0x1200	SFC_FAPC_SADDR_STATUS	FAPC 鉴权。
0x1204	SFC_FAPC_EADDR_STATUS	FAPC 鉴权。
0x1208	SFC_APC_ERR_INT	FAPC 鉴权。
0x120C	SFC_APC_CLR	FAPC 鉴权。
0x1220	FAPC_ONE_WAY_LOCK	FAPC 鉴权锁定寄存器。
0x1300	LEA_LP_EN	LEA 控制。
0x1304	LEA_DFX_INFO	LEA DFX。
0x1600	LEA_IV_VLD	LEA 控制。
0x1640	LEA_IV_ACPU_START_ADDR_0	LEA IV 解密起始地址寄存器 0。

偏移地址	名称	描述
0x1644	LEA_IV_ACPU_START_ADDR_1	LEA IV 解密起始地址寄存器 1。
0x1648	LEA_IV_ACPU_START_ADDR_2	LEA IV 解密起始地址寄存器 2。
0x164C	LEA_IV_ACPU_START_ADDR_3	LEA IV 解密起始地址寄存器 3。

SFC 寄存器偏移地址中变量的取值范围和含义如表 3-2 所示。

表3-2 SFC 寄存器偏移地址变量表

变量名称	取值范围	描述
n	0 ~ 3	FLASH 解密地址的区间个数。

3.5 寄存器描述

GLOBAL_CONFIG

GLOBAL_CONFIG 为全局配置寄存器。

Offset Address: 0x0100 Total Reset Value: 0x0000_0000

Bits	Access	Name	Description	Reset
[31:6]	-	reserved	保留。	0x000000
[5:3]	RW	rd_delay	SPI 读出数据延迟周期个数。 000: 0.5 ~ 1 个时钟周期 (默认值); 001: 1 ~ 1.5 个时钟周期; 010: 1.5 ~ 2 个时钟周期; 011: 2 ~ 2.5 个时钟周期; 100: 2.5 ~ 3 个时钟周期;	0x0

Bits	Access	Name	Description	Reset
			101: 3~3.5 个时钟周期; 110: 3.5~4 个时钟周期; 111: 不支持, (按照“110”含义处理)。	
[2]	RW	flash_addr_mode	SPI 地址模式。 0: 3byte 寻址模式 (默认值); 1: 4byte 寻址模式。 注意: CMD_CONFIG[start]为 1 时写无效。	0x0
[1]	RW	wp_en	硬件写保护使能 (写保护管脚)。 0: 禁止; 1: 使能。	0x0
[0]	RW	mode	SPI 模式设置。 0: 支持 Mode0; 1: 支持 Mode3。	0x0

TIMING

TIMING 为 Timing 配置寄存器。

Offset Address: 0x0110 Total Reset Value: 0x0000_660F

Bits	Access	Name	Description	Reset
[31:15]	-	reserved	保留。	0x00000
[14:12]	RW	tcsn	片选保持时间。 0x0~0x7: (n+1) 个时钟周期。 例如: 0x6 表示 7 个时钟周期。	0x6
[11]	-	reserved	保留。	0x0

Bits	Access	Name	Description	Reset
[10:8]	RW	tcss	片选建立时间。 0x0 ~ 0x7: (n + 1) 个时钟周期。 例如: 0x6 表示 7 个时钟周期。	0x6
[7:4]	-	reserved	保留。	0x0
[3:0]	RW	tshsl	设置 2 次 Flash 操作之间的时间间隔。 0x0 ~ 0xF: (n + 2) 个时钟周期。 例如: 0xF 表示 17 个时钟周期。	0xF

INT_RAW_STATUS

INT_RAW_STATUS 为中断原始状态寄存器。

Offset Address: 0x0120 Total Reset Value: 0x0000_0000

Bits	Access	Name	Description	Reset
[31:2]	-	reserved	保留。	0x00000000
[1]	RO	dma_done_int_raw_status	DMA 操作完成中断原始状态 (未经过屏蔽)。 0: 未完成; 1: 已完成。	0x0
[0]	RO	cmd_op_end_raw_status	指令操作结束原始中断状态 (未经过屏蔽)。 0: 未完成; 1: 已完成。	0x0

INT_STATUS

INT_STATUS 为经过屏蔽处理的中断状态寄存器。

Offset Address: 0x0124 Total Reset Value: 0x0000_0000

Bits	Access	Name	Description	Reset
[31:2]	-	reserved	保留。	0x00000000
[1]	RO	dma_done_int_status	DMA 操作完成中断原始状态（经过屏蔽）。 0: 未完成; 1: 已完成。	0x0
[0]	RO	cmd_op_end_status	指令操作结束中断状态（经过屏蔽）。 0: 未完成; 1: 已完成。	0x0

INT_MASK

INT_MASK 为中断屏蔽寄存器。

Offset Address: 0x0128 Total Reset Value: 0x0000_0000

Bits	Access	Name	Description	Reset
[31:2]	-	reserved	保留。	0x00000000
[1]	RW	dma_done_int_mask	DMA 操作完成中断屏蔽位。 0: 屏蔽; 1: 不屏蔽。	0x0
[0]	RW	cmd_op_end_int_mask	指令操作结束中断屏蔽位。 0: 屏蔽; 1: 不屏蔽。	0x0

INT_CLEAR

INT_CLEAR 为中断清除寄存器。

Offset Address: 0x012C Total Reset Value: 0x0000_0000

Bits	Access	Name	Description	Reset
[31:2]	-	reserved	保留。	0x00000000
[1]	WO	dma_done_int_clr	DMA 操作完成中断清除位，向该位写 1 将清除 INT_STATUS[dma_done_int_status]和 INT_RAW_STATUS[dma_done_int_raw_status]。 0: 不清除; 1: 清除。 注意：清除操作完成后该位自动返回 0。	0x0
[0]	WO	cmd_op_end_int_clr	指令操作结束中断清除位，向该位写 1 将清除 INT_STATUS[cmd_op_end_status]和 INT_RAW_STATUS[cmd_op_end_raw_status]。 0: 不清除; 1: 清除。 注意：清除操作完成后该位自动返回 0。	0x0

SOFT_RST_MASK

SOFT_RST_MASK 为软复位寄存器屏蔽位

Offset Address: 0x0130 Total Reset Value: 0x0000_0001

Bits	Access	Name	Description	Reset
------	--------	------	-------------	-------

Bits	Access	Name	Description	Reset
[31:1]	-	reserved	保留。	0x00000000
[0]	RW	sfc_bus_soft_rst_mask	SFC 总线时钟域数字逻辑软复位屏蔽位： 0：软复位可以正常生效； 1：软复位被屏蔽，不会生效。	0x1

BUS_CONFIG1

BUS_CONFIG1 为总线操作方式配置 1 寄存器。

Offset Address: 0x0200 Total Reset Value: 0x8080_0300

Bits	Access	Name	Description	Reset
[31]	RW	rd_enable	总线读使能。 0：禁止； 1：使能。	0x1
[30]	RW	wr_enable	总线写使能。 0：禁止； 1：使能。	0x0
[29:22]	RW	wr_ins	写指令。	0x02
[21:19]	RW	wr_dummy_bytes	总线写操作 DummyByte。 000：没有 DummyByte； 001：1Byte； 010：2Byte； 111：7Byte。	0x0
[18:16]	RW	wr_mem_if_type	总线写操作指定连接的 SPI FLASH 接口类型。	0x0

Bits	Access	Name	Description	Reset
			000: Standard SPI 接口类型; 001: Dual-Input/Dual-Output SPI; 010: Dual-I/O SPI; 011: Full DIO SPI; 100: 保留; 101: Quad-Input/Dual-Output SPI; 110: Quad-I/O SPI; 111: Full QIO SPI。	
[15:8]	RW	rd_ins	读指令。	0x03
[7:6]	RW	rd_prefetch_cnt	总线访问 Flash 方式 (非定长读)预 取周期。 00: 不预取 (默认值); 01: 预取 1 个时钟周期数据; 10: 预取 2 个时钟周期数据; 11: 预取 3 个时钟周期数据。	0x0
[5:3]	RW	rd_dummy_bytes	总线读操作 DummyByte。 00: 没有 DummyByte; 001: 1Byte 010: 2Byte; 111: 7Byte。	0x0
[2:0]	RW	rd_mem_if_type	总线读操作指定连接的 SPI FLASH 接口类型。 000: Standard SPI 接口类型; 001: Dual-Input/Dual-Output SPI; 010: Dual-I/O SPI;	0x0

Bits	Access	Name	Description	Reset
			101: Quad-Input/Dual-Output SPI; 110: Quad-I/O SPI; 其他: 保留。	

BUS_CONFIG2

BUS_CONFIG2 为总线操作方式配置 2 寄存器。

Offset Address: 0x0204 Total Reset Value: 0x0000_0000

Bits	Access	Name	Description	Reset
[31:3]	-	reserved	保留。	0x00000000
[2:0]	RW	wip_locate	WIP (Write In Progress) 位于 Flash 状态寄存器的位置。 000: WIP 位于 Flash 状态寄存器的 bit[0] (默认值); 001: WIP 位于 Flash 状态寄存器的 bit[1]; 010: WIP 位于 Flash 状态寄存器的 bit[2]; 011: WIP 位于 Flash 状态寄存器的 bit[3]; 100: WIP 位于 Flash 状态寄存器的 bit[4]; 101: WIP 位于 Flash 状态寄存器的 bit[5]; 110: WIP 位于 Flash 状态寄存器的 bit[6]; 111: WIP 位于 Flash 状态寄存器的 bit[7]。	0x0

BUS_DMA_CTRL

BUS_DMA_CTRL 为 DMA 操作控制寄存器。

Offset Address: 0x0240 Total Reset Value: 0x0000_0000

Bits	Access	Name	Description	Reset
[31:5]	-	reserved	保留。	0x00000000
[4]	RW	dma_sel_cs	DMA 操作指定片选。 0: 片选 0; 1: 片选 1。	0x0
[3:2]	-	reserved	保留。	0x0
[1]	RW	dma_rw	DMA 读写指示。 0: 写操作; 1: 读操作。	0x0
[0]	RW	dma_start	DMA 传输使能控制。 0: 无操作; 1: 开始 DMA 操作。 注意: DMA 传输完成自动回 0。	0x0

BUS_DMA_MEM_SADDR

BUS_DMA_MEM_SADDR 为 DMA 操作芯片内存起始地址寄存器。

Offset Address: 0x0244 Total Reset Value: 0x0000_0000

Bits	Access	Name	Description	Reset
[31:0]	RW	dma_mem_saddr	DMA 操作 memory 起始地址。 Q353333 配置值应在 0x0010_0000 ~ 0x00BF_FFFF 之间。	0x00000000

BUS_DMA_FLASH_SADDR

BUS_DMA_FLASH_SADDR 为 DMA 操作 Flash 起始地址寄存器。

Offset Address: 0x0248 Total Reset Value: 0x0000_0000

Bits	Access	Name	Description	Reset
[31:0]	RW	dma_flash_saddr	DMA 操作 Flash 起始地址。	0x00000000

BUS_DMA_LEN

BUS_DMA_LEN 为 DMA 操作搬运数据长度寄存器。

Offset Address: 0x024C Total Reset Value: 0x0000_0000

Bits	Access	Name	Description	Reset
[31:30]	-	reserved	保留。	0x0
[29:0]	RW	dma_len	DMA 操作数据搬运长度 (n+1), 单位: byte。 例如: 6 表示长度为 7byte。	0x00000000

BUS_DMA_AHB_CTRL

BUS_DMA_AHB_CTRL 为 DMA 操作 AHB 时 burst 操作方式选择控制寄存器。

Offset Address: 0x0250 Total Reset Value: 0x0000_0007

Bits	Access	Name	Description	Reset
[31:3]	-	reserved	保留。	0x00000000
[2]	RW	incr16_en	INC16 burst 类型使能。 0: 禁止; 1: 使能。	0x1

Bits	Access	Name	Description	Reset
[1]	RW	incr8_en	INC8 burst 类型使能。 0：禁止； 1：使能。	0x1
[0]	RW	incr4_en	INC4 burst 类型使能。 0：禁止； 1：使能。	0x1

CMD_CONFIG

CMD_CONFIG 为命令操作方式配置寄存器。

Offset Address: 0x0300 Total Reset Value: 0x0000_7E00

Bits	Access	Name	Description	Reset
[31:20]	-	reserved	保留。	0x000
[19:17]	RW	mem_if_type	指定寄存器命令操作方式连接的 SPI FLASH 接口类型。 000: Standard SPI 接口类型; 001: Dual-Input/Dual-Output SPI; 010: Dual-I/O SPI; 101: Quad-Input/Dual-Output SPI; 110: Quad-I/O SPI; 其他: 保留。	0x0
[16:15]	-	reserved	保留。	0x0
[14:9]	RW	data_cnt	读写数据长度 (单位: Byte) 。 0x00 ~ 0x3F: (n+1) Byte。 例如: 0x3F 表示 64Byte。	0x3F

Bits	Access	Name	Description	Reset
[8]	RW	rw	标识此次操作数据读写，需 [data_en] 为 1。 0：写，有发送数据； 1：读，有返回数据。	0x0
[7]	RW	data_en	标识此次操作是否有数据。 0：无数据； 1：有数据。	0x0
[6:4]	RW	dummy_byte_cnt	寄存器命令操作方式 DummyByte。 000：没有 DummyByte； 001：1Byte； 010：2Byte； 111：7Byte。	0x0
[3]	RW	addr_en	此次操作是否有地址。 0：无地址； 1：有地址。	0x0
[2]	-	reserved	保留。	0x0
[1]	RW	sel_cs	片选选择操作。 0：选择片选 0 进行操作； 1：选择片选 1 进行操作。	0x0
[0]	RW	start	标识指令操作开始。 0：结束； 1：开始。 注意：此次操作完成后该位自动回 0。	0x0

CMD_INS

CMD_INS 为命令操作方式指令寄存器。

Offset Address: 0x0308 Total Reset Value: 0x0000_0000

Bits	Access	Name	Description	Reset
[31:8]	-	reserved	保留。	0x000000
[7:0]	RW	reg_ins	寄存器访问 Flash 方式下的指令码。	0x00

CMD_ADDR

CMD_ADDR 为命令操作方式地址寄存器。

Offset Address: 0x030C Total Reset Value: 0x0000_0000

Bits	Access	Name	Description	Reset
[31:30]	-	reserved	保留。	0x0
[29:0]	RW	cmd_addr	寄存器访问 Flash 方式下的操作地址。	0x00000000

CMD_DATABUF_N

CMD_DATABUF_N 为命令操作方式数据 Buffer 寄存器。

Offset Address: 0x0400 + 4 × n Total Reset Value: 0x0000_0000

Bits	Access	Name	Description	Reset
[31:0]	RW	cmd_databuf_n	寄存器访问 Flash 方式下第 n 数据 Buffer (n: 0 ~ 15) 。	0x00000000

APC_CFG_START_ADDR

APC_CFG_START_ADDR 为 FAPC 鉴权。

Offset Address: 0x1000 + 4×n Total Reset Value: 0x0000_0000

Bits	Access	Name	Description	Reset
[31:8]	RW	apc_cfg_start_addr_n	flash 存储区域划分 4 段地址区间，每段起始和截止地址位宽 32bit，该寄存器表示第 n 段起始地址的高 24bit（n：0~3）。 注意：该处为绝对地址，包含总线基地址。	0x000000
[7:0]	-	reserved	保留。	0x00

APC_CFG_END_ADDR

APC_CFG_END_ADDR 为 FAPC 鉴权。

Offset Address: 0x1040 + 4×n Total Reset Value: 0x0000_0000

Bits	Access	Name	Description	Reset
[31:8]	RW	apc_cfg_end_addr_n	flash 存储区域划分 4 段地址区间，每段起始和截止地址位宽 32bit，该寄存器表示第 n 段截止地址的高 24bit（n：0~3） 注：该处为绝对地址，包含总线基地址。	0x000000
[7:0]	-	reserved	保留。	0x00

SFC_FAPC_DEC_AUTH_CFG

SFC_FAPC_DEC_AUTH_CFG 为 FAPC 鉴权。

Offset Address: 0x1180 Total Reset Value: 0x0000_0000

Bits	Access	Name	Description	Reset
[31:0]	RW	sfc_fapc_dec_	flash 存储区域划分 4 段区间，该	0x000000

Bits	Access	Name	Description	Reset
		auth_cfg	寄存器从低位开始，每 2bit (n: 0~3)表示一段区间的数据处理方式： 0b00：透传； 0b10：解密； 其他：保留。	00

SFC_FAPC_SADDR_STATUS

SFC_FAPC_SADDR_STATUS 为 FAPC 鉴权。

Offset Address: 0x1200 Total Reset Value: 0x0000_0000

Bits	Access	Name	Description	Reset
[31:0]	RO	sfc_fapc_saddr_status	APC 状态检测寄存器：基于 burst 起始地址检测 [31:31]：是否出现过鉴权不通过历史态； [30:28]：当前 master 对应 MID 的低 3bit； [27:27]：1 表示当前是写操作，0 表示读； [26:0]：当前访问 flash 地址。	0x00000000

SFC_FAPC_EADDR_STATUS

SFC_FAPC_EADDR_STATUS 为 FAPC 鉴权。

Offset Address: 0x1204 Total Reset Value: 0x0000_0000

Bits	Access	Name	Description	Reset
[31:0]	RO	sfc_fapc_eaddr_status	APC 状态检测寄存器：基于 burst	0x00000000

Bits	Access	Name	Description	Reset
			结束地址检测 [31:31]: 是否出现过鉴权不通过历史态; [30:28]: 当前 master 对应 MID 的低 3bit; [27:27]: 1 表示当前是写操作, 0 表示读; [26:0]: 当前访问 flash 地址。	

SFC_APC_ERR_INT

SFC_APC_ERR_INT 为 FAPC 鉴权

Offset Address: 0x1208 Total Reset Value: 0x0000_0000

Bits	Access	Name	Description	Reset
[31:1]	-	reserved	保留。	0x00000000
[0]	RO	sfc_apc_err_int	鉴权错误中断状态: 0: 鉴权全部通过; 1: 出现鉴权不通过, 上报中断。	0x0

SFC_APC_CLR

SFC_APC_CLR 为 FAPC 鉴权

Offset Address: 0x120C Total Reset Value: 0x0000_0000

Bits	Access	Name	Description	Reset
[31:1]	-	reserved	保留。	0x00000000
[0]	W1_PULSE	sfc_apc_clr	鉴权错误中断状态清除: 0: 无效;	0x0

Bits	Access	Name	Description	Reset
			1: 清除 sfc_apc_err_int 中断。	

FAPC_ONE_WAY_LOCK

FAPC_ONE_WAY_LOCK 为 FAPC 鉴权锁定寄存器

Offset Address: 0x1220 Total Reset Value: 0x0000_0000

Bits	Access	Name	Description	Reset
[31:16]	-	reserved	保留。	0x0000
[15:0]	RW	sfc_fpac_oneyay_lock	鉴权寄存器锁定位, bit0~3 分别表示 16 段地址区间对应的鉴权寄存器锁定位。 0: 对应鉴权寄存器可读可写; 1: 对应鉴权寄存器不可改写, 只能读。 注意: 默认值是 0, 一旦写 1, 只有整 harden 复位才能归 0。	0x0000

LEA_LP_EN

LEA_LP_EN 为 LEA 控制

Offset Address: 0x1300 Total Reset Value: 0x0000_0001

Bits	Access	Name	Description	Reset
[31:1]	-	reserved	保留。	0x00000000
[0]	RW	lea_lp_en	AES 模块低功耗模式配置: 0: 关闭 LEA 低功耗配置; 1: 使能 LEA 低功耗。	0x1

LEA_DFX_INFO

LEA_DFX_INFO is LEA DFX

Offset Address: 0x1304 Total Reset Value: 0x0000_0000

Bits	Access	Name	Description	Reset
[31:0]	RO	lea_dfx_info	AES 模块 DFX 信息观测寄存器。	0x00000000

LEA_IV_VLD

LEA_IV_VLD 为 LEA 控制。

Offset Address: 0x1600 Total Reset Value: 0x0000_0000

Bits	Access	Name	Description	Reset
[31:1]	-	reserved	保留。	0x00000000
[0]	RW	lea_iv_vld	AES IV 值有效寄存器。 配置 AES_IV 之后，需要将该寄存器写 1，是配置同步生效，做完时钟同步之后，该信号自动归 0。	0x0

LEA_IV_ACPU_START_ADDR_0

LEA_IV_ACPU_START_ADDR_0 为 LEA IV 解密起始地址寄存器 0。

Offset Address: 0x1640 Total Reset Value: 0x0000_0000

Bits	Access	Name	Description	Reset
[31:8]	RW	iv_start_addr_0	ACPU IV 解密区域 0 的起始地址，该寄存器表示起始地址的高 24bit。 注意：该处为绝对地址，包含总线基地址。	0x00000000
[7:0]	-	reserved	保留。	0x00

LEA_IV_ACPU_START_ADDR_1

LEA_IV_ACPU_START_ADDR_1 为 LEA IV 解密起始地址寄存器 1。

Offset Address: 0x1644 Total Reset Value: 0x0000_0000

Bits	Access	Name	Description	Reset
[31:8]	RW	iv_start_addr_1	ACPU IV 解密区域 1 的起始地址, 该寄存器表示起始地址的高 24bit。 注意: 该处为绝对地址, 包含总线基地址。	0x000000
[7:0]	-	reserved	保留。	0x00

LEA_IV_ACPU_START_ADDR_2

LEA_IV_ACPU_START_ADDR_2 为 LEA IV 解密起始地址寄存器 2。

Offset Address: 0x1648 Total Reset Value: 0x0000_0000

Bits	Access	Name	Description	Reset
[31:8]	RW	iv_start_addr_2	ACPU IV 解密区域 2 的起始地址, 该寄存器表示起始地址的高 24bit。 注意: 该处为绝对地址, 包含总线基地址。	0x000000
[7:0]	-	reserved	保留。	0x00

LEA_IV_ACPU_START_ADDR_3

LEA_IV_ACPU_START_ADDR_3 为 LEA IV 解密起始地址寄存器 3。

Offset Address: 0x164C Total Reset Value: 0x0000_0000

Bits	Access	Name	Description	Reset
[31:8]	RW	iv_start_addr_3	ACPU IV 解密区域 3 的起始地址，该寄存器表示起始地址的高 24bit。 注意：该处为绝对地址，包含总线基地址。	0x000000
[7:0]	-	reserved	保留。	0x00

4 WiFi /BLE & SLE 系统

- 4.1 WiFi / BLE & SLE RF
- 4.2 WiFi/BLE & SLE ABB
- 4.3 WiFi PHY
- 4.4 WiFi MAC
- 4.5 BLE/SLE
- 4.6 感知功能

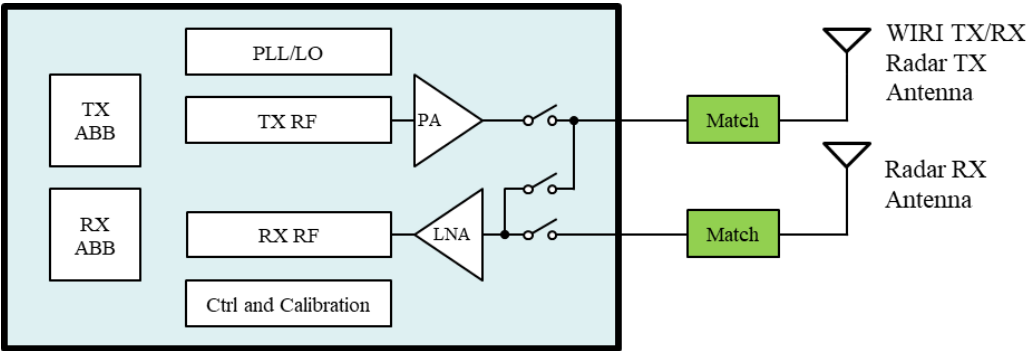
4.1 WiFi / BLE & SLE RF

4.1.1 概述

RF 部分包含 2.4G RX、TX、PLL 三个模块。支持 IEEE 802.11b/g/n/ax 20M 模式。
RF 的电路功能主要包含：

- 集成 TX/RX Switch。
- RX 通路包含 LNA、Mixer、LPF (Low Pass Filter)、VGA (Variable Gain Amplifier)。
- TX 通路包含 LPF、UPC (UP Converter)、PA (Power Amplifier)。
- 集成 PLL/LO (Local Oscillator) 通路，为信号通路提供 LO。
- 集成了 Radar 功能

图4-1 RF 电路模块架构



4.1.2 功能描述

WiFi RF 具有以下功能特点：

- RF 电路提供稳定的 LO 信号，支持收发信号的上下变频功能。
- 支持校准功能，包含：RX DC（Direct Current）校准、TX LO Leakage 校准、TX Power 校准、TRX IQ 校准等。

4.1.3 RF 性能

芯片集成 2.4G WiFi/BLE/SLE 收发机，支持感知功能。除感知在 RFIO 口接收之外，其他的功能都在 RFIO 口测试。

参数	Sub-Item	最小值	典型值	最大值	单位	测试条件
RF 工作频率段	-	2400	-	2500	MHz	2401MHz 以下和 2483.5MHz 以上频点无法满足无委会辐射要求。 信道频率的选择需要遵循协议和法规要求。
WIF RX 11b 灵敏度	1 Mbps DSSS	-	-99	-	dBm	PER 8%,1024 octet PSDU
	2 Mbps DSSS	-	-96	-	dBm	
	5.5 Mbps	-	-94	-	dBm	

参数	Sub-Item	最小值	典型值	最大值	单位	测试条件
	DSSS/CCK					
	11 Mbps DSSS/CCK	-	-91	-	dBm	
WIF RX 11g 灵敏度	BPSK, R=1/2 (6Mbps OFDM)	-	-96	-	dBm	PER 8%, 1000 octets PSDU
	BPSK, R=3/4 (9Mbps OFDM)	-	-94	-	dBm	
	QPSK, R=1/2 (12Mbps OFDM)	-	-93	-	dBm	
	QPSK, R=3/4 (18Mbps OFDM)	-	-90	-	dBm	
	16-QAM, R=1/2 (24Mbps OFDM)	-	-87	-	dBm	
	16-QAM, R=3/4 (36Mbps OFDM)	-	-84	-	dBm	
	16-QAM, R=1/2 (48Mbps OFDM)	-	-80	-	dBm	
	64-QAM, R=3/4 (54Mbps OFDM)	-	-78	-	dBm	
WIF RX 11n HT20-MF 灵敏度	HT20 MCS0	-	-95	-	dBm	BCC Long PER 10%, 4096 octets PSDU
	HT20 MCS1	-	-92	-	dBm	
	HT20 MCS2	-	-90	-	dBm	
	HT20 MCS3	-	-87	-	dBm	

参数	Sub-Item	最小值	典型值	最大值	单位	测试条件
	HT20 MCS4	-	-83	-	dBm	
	HT20 MCS5	-	-79	-	dBm	
	HT20 MCS6	-	-77	-	dBm	
	HT20 MCS7	-	-76	-	dBm	
WIF RX 11n HT40 灵敏度	HT40 MCS0	-	-93	-	dBm	BCC Long PER 10%, 4096 octets PSDU
	HT40 MCS1	-	-90	-	dBm	
	HT40 MCS2	-	-87	-	dBm	
	HT40 MCS3	-	-84	-	dBm	
	HT40 MCS4	-	-81	-	dBm	
	HT40 MCS5	-	-76	-	dBm	
	HT40 MCS6	-	-74	-	dBm	
	HT40 MCS7	-	-73	-	dBm	
WIF RX 11ax HE20 灵敏度	HE20 MCS0	-	-96	-	dBm	BCC 4x3.2us PER 10%, 4096 octets PSDU
	HE20 MCS1	-	-93	-	dBm	
	HE20 MCS2	-	-91	-	dBm	
	HE20 MCS3	-	-88	-	dBm	
	HE20 MCS4	-	-84	-	dBm	
	HE20 MCS5	-	-80	-	dBm	
	HE20 MCS6	-	-79	-	dBm	
	HE20 MCS7	-	-77	-	dBm	
	HE20 MCS8	-	-73	-	dBm	
	HE20 MCS9	-	-71	-	dBm	
WIFI RX 11b 最大解 调电平	1 Mbps DSSS	-	0	-	dBm	PER 8%, 1024 octets PPDU
	2 Mbps DSSS	-	0	-	dBm	
	5.5 Mbps DSSS/CCK	-	0	-	dBm	
	11 Mbps DSSS/CCK	-	0	-	dBm	
WIFI RX 11g 最大解	64-QAM, R=3/4 (54Mbps)	-	0	-	dBm	PER 8%, 1000 octets PPDU

参数	Sub-Item	最小值	典型值	最大值	单位	测试条件
调电平	OFDM)					
WIFI RX 11n HT20 最大解调电平	HT20 MCS7	-	0	-	dBm	PER 10%, 4096 octets PPDU
WIFI RX 11b 邻道抑制比	1Mbps DSSS	-	48	-	dB	有用信号-74dBm, PER 8%, 1024 octets PPDU
	2Mbps DSSS	-	59	-	dB	有用信号-74dBm, PER 8%, 1024 octets PPDU
	5.5Mbps DSSS/CCK	-	44	-	dB	有用信号-70dBm, PER 8%, 1024 octets PPDU
	11Mbps DSSS/CCK	-	44	-	dB	有用信号-70dBm, PER 8%, 1024 octets PPDU
WIFI RX 11g 邻道抑制比	BPSK, R=1/2 (6Mbps OFDM)	-	34	-	dB	有用信号-79dBm, PER 8%, 1000 octets PPDU
	BPSK, R=3/4 (9Mbps OFDM)	-	30	-	dB	有用信号-78dBm, PER 8%, 1000 octets PPDU
	QPSK, R=1/2 (12Mbps OFDM)	-	31	-	dB	有用信号-76dBm, PER 8%, 1000 octets PPDU
	QPSK, R=3/4 (18Mbps OFDM)	-	27	-	dB	有用信号-74dBm, PER 8%, 1000 octets PPDU
	16-QAM, R=1/2 (24Mbps OFDM)	-	27	-	dB	有用信号-71dBm, PER 8%, 1000 octets PPDU
	16-QAM,	-	22	-	dB	有用信号-67dBm,

参数	Sub-Item	最小值	典型值	最大值	单位	测试条件
	R=3/4 (36Mbps OFDM)					PER 8%, 1000 octets PPDU
	16-QAM, R=1/2 (48Mbps OFDM)	-	19	-	dB	有用信号-63dBm, PER 8%, 1000 octets PPDU
	64-QAM, R=3/4 (54Mbps OFDM)	-	20	-	dB	有用信号-62dBm, PER 8%, 1000 octets PPDU
WIFI RX 11n TH20 邻道抑制比	HT20 MCS0	-	30	-	dB	有用信号-79dBm, PER 10%, 4096 octets PPDU
	HT20 MCS1	-	29	-	dB	有用信号-76dBm, PER 10%, 4096 octets PPDU
	HT20 MCS2	-	26	-	dB	有用信号-74dBm, PER 10%, 4096 octets PPDU
	HT20 MCS3	-	24	-	dB	有用信号-71dBm, PER 10%, 4096 octets PPDU
	HT20 MCS4	-	19	-	dB	有用信号-67dBm, PER 10%, 4096 octets PPDU
	HT20 MCS5	-	17	-	dB	有用信号-63dBm, PER 10%, 4096 octets PPDU
	HT20 MCS6	-	15	-	dB	有用信号-62dBm, PER 10%, 4096 octets PPDU
	HT20 MCS7	-	13	-	dB	有用信号-61dBm, PER 10%, 4096 octets PPDU

参数	Sub-Item	最小值	典型值	最大值	单位	测试条件
WIFI RX 11n TH40 邻道抑制比	HT40 MCS0	-	29	-	dB	有用信号-76dBm, PER 10%, 4096 octets PPDU
	HT40 MCS1	-	27	-	dB	有用信号-73dBm, PER 10%, 4096 octets PPDU
	HT40 MCS2	-	24	-	dB	有用信号-71dBm, PER 10%, 4096 octets PPDU
	HT40 MCS3	-	21	-	dB	有用信号-68dBm, PER 10%, 4096 octets PPDU
	HT40 MCS4	-	17	-	dB	有用信号-64dBm, PER 10%, 4096 octets PPDU
	HT40 MCS5	-	13	-	dB	有用信号-60dBm, PER 10%, 4096 octets PPDU
	HT40 MCS6	-	14	-	dB	有用信号-59dBm, PER 10%, 4096 octets PPDU
	HT40 MCS7	-	10	-	dB	有用信号-58dBm, PER 10%, 4096 octets PPDU
WIFI TX11b 最大发射功 率	1Mbps DSSS	-	23	-	dBm	Mask & EVM 满足 协议要求
	2Mbps DSSS	-	23	-	dBm	
	5.5Mbps DSSS/CCK	-	23	-	dBm	
	11Mbps DSSS/CCK	-	23	-	dBm	
WIFI TX11g 最大发射功 率	BPSK, R=1/2 (6Mbps OFDM)	-	21	-	dBm	
	BPSK, R=3/4 (9Mbps OFDM)	-	21	-	dBm	

参数	Sub-Item	最小值	典型值	最大值	单位	测试条件
	QPSK, R=1/2 (12Mbps OFDM)	-	21	-	dBm	
	QPSK, R=3/4 (18Mbps OFDM)	-	21	-	dBm	
	16-QAM, R=1/2 (24Mbps OFDM)	-	21	-	dBm	
	16-QAM, R=3/4 (36Mbps OFDM)	-	21	-	dBm	
	16-QAM, R=1/2 (48Mbps OFDM)	-	20	-	dBm	
	64-QAM, R=3/4 (54Mbps OFDM)	-	19	-	dBm	
WIFI TX HT20-MF 最大发射功率	MCS0	-	20	-	dBm	BCC long GI
	MCS1	-	20	-	dBm	
	MCS2	-	20	-	dBm	
	MCS3	-	19	-	dBm	
	MCS4	-	19	-	dBm	
	MCS5	-	19	-	dBm	
	MCS6	-	19	-	dBm	
	MCS7	-	18	-	dBm	
WIFI TX HT40-MF	MCS0	-	20	-	dBm	BCC long GI
	MCS1	-	20	-	dBm	

参数	Sub-Item	最小值	典型值	最大值	单位	测试条件
最大发射功率	MCS2	-	20	-	dBm	
	MCS3	-	19	-	dBm	
	MCS4	-	19	-	dBm	
	MCS5	-	19	-	dBm	
	MCS6	-	19	-	dBm	
	MCS7	-	18	-	dBm	
WIFI TX HE20 最大发射功率	MCS0	-	20	-	dBm	-
	MCS1	-	20	-	dBm	
	MCS2	-	20	-	dBm	
	MCS3	-	19	-	dBm	
	MCS4	-	19	-	dBm	
	MCS5	-	19	-	dBm	
	MCS6	-	19	-	dBm	
	MCS7	-	18	-	dBm	
	MCS8	-	17	-	dBm	
	MCS9	-	15	-	dBm	
LE RX 灵敏度	LE 1M	-	-99	-	dBm	PER< 30%
	LE 2M	-	-96	-	dBm	
	LR S2 255byte	-	-100	-	dBm	
	LR S8 255byte	-	-105	-	dBm	
	LR S2 37byte	-	-101	-	dBm	
	LR S8 37byte	-	-105	-	dBm	
SLE RX 灵敏度	SLE_1M GFSK	-	-99	-	dBm	PER <10%
	SLE_2M GFSK	-	-96	-	dBm	
	SLE_4M GFSK	-	-93	-	dBm	
	1M QPSK shortHD	-	-101	-	dBm	

参数	Sub-Item	最小值	典型值	最大值	单位	测试条件
	pilot16:1 polar3/4					
	2M QPSK shortHD pilot16:1 polar3/4	-	-98	-	dBm	
	4M QPSK shortHD pilot16:1 polar3/4	-	-95	-	dBm	
	1M 8PSK shortHD pilot16:1 polar3/4	-	-96	-	dBm	
	2M 8PSK shortHD pilot16:1 polar3/4	-	-93	-	dBm	
	4M 8PSK shortHD pilot16:1 polar3/4	-	-90	-	dBm	
	1M QPSK shortHD pilot_no polar1/1	-	-96	-	dBm	
	2M QPSK shortHD pilot_no polar1/1	-	-93	-	dBm	
	4M QPSK shortHD pilot_no polar1/1	-	-89	-	dBm	
	1M 8PSK shortHD pilot_no polar1/1	-	-90	-	dBm	
	2M 8PSK shortHD pilot_no polar1/1	-	-87	-	dBm	

参数	Sub-Item	最小值	典型值	最大值	单位	测试条件
	4M 8PSK shortHD pilot_no polar1/1	-	-82	-	dBm	
LE TX 最大 发射功率	LE 1M	-	20	-	dBm	满足 EVM 和 ACP 要求
	LE 2M	-	20	-	dBm	
	LR S2 500K	-	20	-	dBm	
	LR S8 125K	-	20	-	dBm	
SLE TX 最大 发射功率	SLE_1M GFSK	-	20	-	dBm	
	SLE_2M GFSK	-	20	-	dBm	
	SLE_4M GFSK	-	20	-	dBm	
	1M QPSK shortHD pilot16:1 polar3/4	-	14	-	dBm	
	2M QPSK shortHD pilot16:1 polar3/4	-	14	-	dBm	
	4M QPSK shortHD pilot16:1 polar3/4	-	14	-	dBm	
	1M 8PSK shortHD pilot16:1 polar3/4	-	14	-	dBm	
	2M 8PSK shortHD pilot16:1 polar3/4	-	14	-	dBm	
	4M 8PSK shortHD pilot16:1 polar3/4	-	14	-	dBm	
	1M QPSK shortHD	-	14	-	dBm	

参数	Sub-Item	最小值	典型值	最大值	单位	测试条件
	pilot_no polar1/1					
	2M QPSK shortHD pilot_no polar1/1	-	14	-	dBm	
	4M QPSK shortHD pilot_no polar1/1	-	14	-	dBm	
	1M 8PSK shortHD pilot_no polar1/1	-	14	-	dBm	
	2M 8PSK shortHD pilot_no polar1/1	-	14	-	dBm	
	4M 8PSK shortHD pilot_no polar1/1	-	14	-	dBm	
TX 输出功率精度	-	-2	-	2	dB	-
TX 输出功率分辨率	-	-	1	-	dB	BLE 只能发送固定功率

 说明

以上数据仿真条件为 VBAT=3.3V。

4.2 WiFi/BLE & SLE ABB

4.2.1 概述

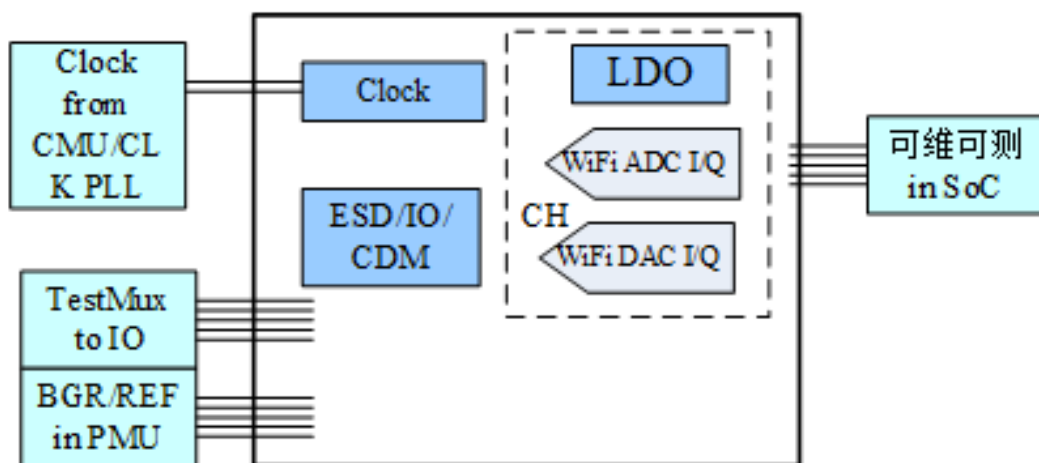
ABB IP 用于 Connectivity SoC 芯片，是支持 WiFi 802.11b/g/n/ax（2.4G mode）系统的模拟数字接口模块，根据功能分为以下 2 个功能模块：

- WiFi IQ-ADC
- WiFi IQ DAC

完成发送时的数模转换及接收时的模数转换功能。

WiFi ADC (1 个通道, 通道有 IQ)、WiFi DAC (1 个通道, 通道有 IQ), 以及时钟 buf 模块和 LDO, 共同包括在 Q353333V1100 WL ABB 中, 时钟 buf 和 LDO 不作为独立功能模块, 不在行为模型中独立体现。

图4-2 ABB 模块组成



注：WADC (WiFi Analog Digital Converter), WDAC (WiFi Digital Analog Converter)。

4.2.2 功能描述

ABB IP 具有以下功能特点：

- 提供 1 路 WiFi IQ ADC、1 路 WiFi IQ DAC。
- 支持 WiFi 802.11b/g/n (2.4G mode)。

4.2.3 工作方式

业务模式寄存器配置为固定一次性配置，业务工作期间无需重复配置，仅在逻辑电源掉电重新上电时需要重新配置；校正算法在温度、电压漂移下受影响，如果温度、电压变化较大，需要重新运行算法并刷新寄存器，除此情况之外无需重复配置。

校准包括：

- WLAN (Wireless Local Area Network) 的 ADC 比较器校准。

- WLAN DC Offset 校准。
- WLAN 电容校准。

校准步骤：

步骤 1 比较器校准。

步骤 2 电容校准。

步骤 3 DC Offset 校准。

----结束

4.3 Wi-Fi PHY

4.3.1 概述

WLAN PHY 实现 802.11 协议定义的物理层功能，包括：

- 802.11b 协议定义的 DSSS、CCK 调制解调。
- 802.11g、802.11n、802.11ax 协议定义的 OFDM 调制解调包括发送的加扰、交织、编码、OFDM 调制等处理；接收方向 OFDM 解调、Viterbi 译码、解交织、解扰等处理；同时实现 AGC (Automatic Gain Control)、CCA (Clear Channel Assessment)、RSSI (Receive Signal Strength Indicator)功能。
- 实现 RF/ABB 校准功能。

4.3.2 功能描述

Wi-Fi PHY 具有以下功能特点：

- 支持 IEEE802.11b/g/n/ax 无线局域网络通信协议，其中 ax 支持 su/ersu 的收发、tb 帧的发送、mu 帧的接收。
- 支持 802.11b 的 DSSS、CCK，802.11g/n/ax 的 BCC (Binary Convolutional Code)编解码，802.11n/ax 的 LDPC (Low Density Parity Check)的编码。
- 支持 2.4G Band，802.11b/g/n/ax 支持 20MHz 信号带宽，802.11n 支持 40MHz 信号带宽，802.11ax (tb/mu) 支持 20MHz-only 信号带宽。
- 支持 4 选 1 多天线分集，最大支持 1 个空间流；802.11n/ax 支持 STBC (Space-Time Block Code)接收；802.11n/ax 支持 4x1 TxBF；802.11ax 最多支持 4 用户识别并支持配置其中任一个用户接收。

- 支持运动物体感知 (Moving object Sensing)。
- 支持 GLP (Green-tooth Low-energy Positioning)辅助同步。
- 支持 PSD (Power Spectral Desity)上报。
- 支持 CSI (Channel State Infomation)上报。
- 支持 ABB/RF 校准功能。

4.3.3 工作方式

PHY 模式初始化配置支持物理带宽为 20MHz 的 WiFi 业务收发，在业务模式下可以根据与 AP 的交互完成不同物理带宽的切换，也可以再测试模式下配置不同的物理带宽用于性能测试或者问题定位。PHY 会根据不同的带宽，自适应驱动配置，完成基带数据发送或者接收。根据上层业务的需求，PHY 主要支持以下几种工作模式。

校准模式

在上电时对 ABB/RF 进行离线校准，RF 配置校准模式，复用 PHY 中部分逻辑通路进行校准计算，校准项主要包括 TXDC、TXIQ、TXPWR、RXDC、RXIQ、RXRC 等，校准完成后将校准结果存储在 PHY 中对应配置寄存器中，在测试或者业务模式下自动线控调用，优化 ABB/RF 性能。

测试模式

测试模式，主要是常发、常收测试。其中常发测试主要是指基于描述符后来者配置寄存器下发 TXVECTOR 来启动 RF 线控及 PHY 内部编码调制等，最终将数字 DAC 数据送给 ABB/RF 输出，多帧连续输出，配合仪器用于测试发送时各种性能指标或者基本问题定位；常收测试主要将进过 ABB/RF 的数字 ADC 数据送给 PHY 进行解调，并将解调后的数据送给 MAC 进行 FCS (Frame Check Sequence)校验，来统计接收数据的是否正确，多帧连续输入，配合仪器用于测试接收时各种性能指标或者基本问题定位。

业务模式

业务模式下，PHY 受上层 MAC 主控，与 AP 进行收发通信。业务发送时，PHY 接收来自 MAC 的 TXVECTOR，启动 RF 线控及 PHY 编码调制等，最终将数字 DAC 送给 ABB/RF 数采。业务接收时，PHY 将来自 ABB/RF 的数字 ADC 数据经过 AGC 控制后进行解调译码，并将解析后数据送给上层 MAC 进一步处理。

感知模式

感知模式下，PHY 受上层 MAC 主控，根据业务需求启动感知使能，PHY 会从 PKTRAM 中读取 DAC 采样率下的感知数据，经过校准后送给 ABB/RF；同时会将经过 ABB/RF 的数字 ADC 数据经过固定增益控制后的 ADC 数据进行校准处理，然后校准后的 ADC 数据储存在 PKTRAM 中，并给出中断信息，CPU 收到中断信息后对存储在 PKTRAM 中的感知数据做进一步处理，满足感知业务需求。

PSD 模式

PSD 模式下，PHY 将来自 ABB/RF 的数字 ADC 数据经过 AGC 控制后进行 FFT 计算等统计，最终将 PSD 存储在内部存储空间，并以中断形式上报 CPU，CPU 收到中断后顺序将 PSD 信息读出。可以通过配置不同信道多次统计，收集数据用于再次开发利用。

GLP 联合测距模式

支持与 GLP 联合测距，WiFi 业务下生成发送/接收初始化同步脉冲，给 GLP 提供精确的脉冲定时以及频偏估计值上报。

4.4 WiFi MAC

4.4.1 概述

DBB (Digital Baseband) MAC 主要完成 WiFi MAC 层的硬件处理，包括信道接入、组解帧、数据收发、加解密、节能控制等功能。

4.4.2 功能描述

WiFi MAC 具有以下功能特点：

- 支持 IEEE802.11b/g/n/ax 无线局域网络通信协议。
- 支持 STA 模式和 AP 模式。
- 支持 2.4G Band、802.11b/a/g 20MHz；802.11n 20MHz/40MHz；802.11ax 20MHz。最大支持 1 流、1 天线。
- 支持 WPA、WPA2、AES 加解密。
- 支持 WPS2.0。

- 支持协议低功耗：PSM (Power Saving Mode)、UAPSD (Unscheduled Automatic Power Save Delivery)、P2P (Peer-to-Peer) Power Save。

4.4.3 工作方式

4.4.3.1 AP 模式

在一个基础 BSS (Basic Service Set) 网络中提供所有接入点的基本功能，包括：

- 发送 Beacon 帧声明 BSS 的存在和能力。
- 为 BSS 中的客户端提供无线关联和认证服务。
- 管理 BSS 网络中与之关联的客户端。
- 芯片支持 1 个 AP。

4.4.3.2 STA 模式

在一个基础 BSS 网络中提供扫描发现网络、加入网络并管理与 AP 的连接以提供数据收发服务的功能。

芯片支持 2 个 STA。

4.4.3.3 Monitor 模式

芯片进入 Monitor 模式，实现网卡的功能，MAC 将接收到的所有帧上报软件。

4.4.3.4 AP 与 STA 共存

芯片支持 1 个 AP 和 1 个 STA 同时工作。

芯片支持 2.4G 下 AP/STA 在相同或不同信道的并发，分别对应同频共存和异频共存。

约束：STA 上电后会进行信道扫描，导致信道切换，因此开启 AP/STA 动态共存时，需要先创建 STA，再创建 AP，否则将会影响 AP 的工作信道。

4.4.3.5 CSI 模式

CSI (Channel State Information) 模式支持将 PHY 上报的信道状态信息 (CSI) 过滤后上报软件：

- 支持 11g/11n/11ax 的 CSI 信息上报，不支持 11b。
- 支持对将提取 CSI 的帧进行源地址过滤，源地址过滤列表 (白名单) 共 6 个 (关联设备使用 LUT (Lookup Table) 中的地址内容)。

- 支持 6 个 CSI 采样周期，CSI 采样周期与白名单绑定，一个白名单对应一个采集周期。
- 支持白名单、采样周期、特定帧类型等匹配条件，满足匹配条件才上报 CSI 信息。
- 支持带宽（20MHz、40MHz）、帧格式（NON-HT、HT-MF）、RSSI（Received Signal Strength Indicator）、SNR（Signal Noise Ratio）随 CSI 信息上报（不支持 STBC 帧上报），上报 L-LTF H 矩阵数据。

4.5 BLE/SLE

4.5.1 概述

BLE/SLE 部分包含 MODEM 和 MAC，MODEM 实现调制解调功能，MAC 部分实现调度、收发控制和组包解包功能。

4.5.2 功能描述

BLE 主要特性如表 4-1 所示。

表4-1 BLE 主要特性

标题	描述
蓝牙协议版本	支持蓝牙核心规范 5.4。
蓝牙模式	仅支持 Low Energy only。
BT4.0 特性	支持蓝牙规范 4.0 特性。
Low Energy Physical	Low Energy Physical Layer。
Low Energy Link	Low Energy Link Layer。
Enhancements to HCI for Low Energy	支持 BLE 模式相关的 HCI 功能。
Low Energy Direct Test Mode	支持 BLE 直接测试模式。
AES Encryption	支持对数据包进行 AES 加解密。
BT4.1 特性	支持蓝牙规范 4.1 特性。
Low duty cycle directed advertising	支持低占空比定向广播。

标题	描述
LE Dual mode topology	BLE 设备可同时为 master 和 Slave。
Fast Advertising interval	支持高占空比定向广播。
LE privacy v1.1	支持 LE 隐私策略 v1.1。
LE Ping	支持 LE Ping 功能。
Private address changes	支持私有地址变更功能。
BT4.2 特性	支持蓝牙规范 4.2 特性。
LE Data Packet Length Extension	支持数据包长度扩展，最大可支持 250Byte。
LE Secure Connections	支持低功耗蓝牙安全连接。
Link Layer privacy	支持低功耗蓝牙链路层隐私策略。
Link Layer Extended Scanner Filter policies	支持扩展扫描过滤机制。
BT5.0 特性	支持蓝牙规范 5.0 特性。
2 Msym/s PHY for LE	支持 2M 传输速率。
LE Channel Selection Algorithm #2	支持自适应跳频算法 2。
High Duty Cycle Non-Connectable Advertising	支持高占空比非连接广播。
LE Long Range	支持 BLE Long Range。
BT5.2 特性	支持蓝牙规范 5.2 特性。
BLE Power Control	支持功率控制功能。
连接个数	支持 4 条 BLE 连接（可选 8 条）。
BLE dual mode	BLE 设备支持的角色。
Master	支持 LE 的 Master role。
Slave	支持 LE 的 Slave Role。
PHY Update	支持选择 PHY 信道。
Data Length Update	支持选择数据包的长度。

标题	描述
白名单个数	白名单个数最大支持 8 条。
BLE RPA 列表	Device 能支持的最大的 BLE RPA 名单数目到 4 个。
RPA 功能	支持私有可解析地址功能。 广播、扫描、Init 支持 RPA 功能。
RPA 名单个数	最大支持 4 条 RPA 条目。
快速信道干扰检测	支持业务间隙扫描蓝牙信道所有频点，以判断空口的干扰程度。
Channel map update	支持信道位图更新功能。
信道扫描	支持扫描所有的蓝牙信道，根据扫描结果评估信道干扰程度。

SLE 主要特性如表 4-2 所示。

表4-2 SLE 主要特性

标题	描述
SLE 协议 1.0	支持 SLE1.0 协议核心规范内容。
SLE 链路管理	支持 SLE 链路管理。
时隙调度	支持系统基础时隙按 125μs 调度。
SLE 广播业务	支持 SLE 广播链路业务。
Channel Scan 业务	支持对通信信道进行扫描，上报信道 rssi。
SLE 帧格式	支持 SLE1.0 协议无线帧类型。 支持 SLE1.0 协议无线帧类型 1。 支持 SLE1.0 协议无线帧类型 2。
白名单个数	白名单个数最大支持 8 条。
SLE 调制模式和物理层带宽	支持调制解调带宽 1M/2M/4M 三种速率。

标题	描述
	支持 SLE 调制方式 GFSK-1M/GFSK-2M/GFSK-4M 三种速率。 支持 QPSK 调制方式 QPSK-1M/QPSK-2M/QPSK-4M 三种速率。 支持 8PSK 调制方式 8PSK-1M/8PSK-2M/8PSK-4M 三种速率。
SLE 码率	支持帧类型 2 下，QPSK 调制 Polar 码率为 3/4。 支持帧类型 2 下，8PSK 调制 Polar 码率为 3/4、1。
SLE 导频插值比例	支持帧类型 2 下，数据信息符号导频比为 4:1、16:1。
信道干扰检测	支持信道扫描业务进行干扰检测。
连接个数	支持默认最大支持 4 条 SLE link（可选 8 条，与 BLE 共享连接数）。

4.5.3 工作方式

4.5.3.1 中断

BLE&SLE CPU 只有 2 个主中断源 ble_irq/SLE_irq，每个中断源由多个子中断源汇和而成，CPU 响应相应中断源，通过查询中断状态寄存器来查询子中断类型。

4.5.3.2 加密

BLE 支持 AES-128 加密方式，SLE 支持 SM4 和 AES-128 加密。

4.6 感知功能

4.6.1 概述

感知模块通过芯片内置的感知信号发送、接收以及信号处理模块，实现近距离的运动物体感知功能。

须知

实际应用时，感知性能与 TRx 天线增益、TRx 通路间的隔离度相关，详细请参考感知模组设计约束相关文档。

4.6.2 功能描述

1. 自动计算硬件 TRx 通路间的隔离度；
2. 根据硬件 TRx 间隔度自动适配接收增益；
3. 提供动目标的测距功能，上报距离模组最近的动目标，不区分角度，只上报距离。

须知

1. 动目标定义为所有在可观测范围内的运动物体。
2. 天线设计需要满足天线隔离度、天线方向性、天线增益及天线净空等要求。
3. 动目标检测功能的精度、最远工作距离、可视角度，受实际应用环境影响(反射物、遮挡物、信道环境、电磁波传播等)，性能发生改变，属于正常现象。
4. 如果出现连续的 WIFI\BLE\SLE 业务切换，如周期性扫描等，感知触发延时会有对应延长，具体延长时间与感知功能不可用的时间占比相关。

4.6.3 工作方式

感知模块与 WIFI\BLE\SLE 业务之间通过分时共存，感知信号本身有如下特点：

- 感知信号间隔 5ms。
- 感知信号占空比 < 2%。
- 最快每 0.32s 上报一次检测结果。

须知

- 1. 当本机 WIFI\BLE\SLE 业务连续工作，导致感知业务不能以 5ms 间隔进行收发工作，感知性能会下降；
- 2. 若 WIFI\BLE\SLE 业务持续工作时间超过 500ms 时，感知功能将暂停工作，等待 WIFI\BLE\SLE 业务停止后，感知功能自动恢复。

4.6.4 RF 性能

感知功能使用芯片的 RFI 口进行接收，RFIO 口进行发射。

参数	Sub-Item	最小值	典型值	最大值	单位	测试条件
感知 RF 工作频率段	-	2400	-	2500	MHz	2401MHz 以下和 2483.5MHz 以上频点无法满足无委会辐射要求。 信道频率的选择需要遵循协议和法规要求。
感知发射功率	-	-	11.5	-	dBm	感知发射功率不可调，为固定的 11.5dBm，边带信道下，如果 WIFI 的发射功率设置低于 11.5dBm，则感知的发射功率以实际 WIFI 发射功率为准。

5 安全系统

- 5.1 安全子系统
- 5.2 对称加解密模块
- 5.3 HASH 模块
- 5.4 密钥派生模块
- 5.5 非对称公钥算法模块
- 5.6 随机数生成模块
- 5.7 EFUSE

5.1 安全子系统

5.1.1 概述

安全子系统包括对称加解密算法模块、非对称算法模块、单向散列算法模块、密钥派生模块、随机数生成器模块等子模块。

5.1.2 功能描述

安全子系统支持以下功能：

- 支持 AES/SM4 对称算法加解密。
- 支持 SHA1/SHA2/SM3 散列算法。
- 支持 PBKDF2 密钥派生算法。
- 支持多种非对称公钥算法。

- 支持随机数生成。

5.2 对称加解密模块

5.2.1 概述

实现 AES-128/AES-192/AES-256/SM4 对称加解密算法的算子模块，由 CPU 配置发起计算。

5.2.2 功能描述

对称加解密模块具有以下功能特点：

- 支持 AES-128/AES-192/AES-256/SM4。
- 支持 ECB/CBC/CTR/CFB/OFB/CCM/GCM/CBC_MAC/CMAC 工作模式。

5.3 HASH 模块

5.3.1 概述

HASH 是一个实现单向散列算法的模块，主要用于数据完整性认证。

5.3.2 功能描述

HASH 模块具有以下功能特点：

- 支持算法 SHA1。
- 支持 SHA-2-224/256/384/512。
- 支持国密 SM3。
- 支持 HMAC-SHA1/HMAC-SHA224/HMAC-SHA256/HMAC-SHA384/HMAC-SHA512 单向散列算法。

5.4 密钥派生模块

5.4.1 概述

生成和配置对称加解密、HASH、FLASH 在线解密模块所需要的软件 key 和硬件 Key。软件 key 由用户传入派生材料，派生算法可选择 PBKDF2 和 HKDF；硬件 key 的派生材料根据用户传入的 root key 类型决定，派生算法固定使用 PBKDF2 算法，得到的 key 软件无法读取，由逻辑层面提供给其他模块使用，适用于对密钥安全性要求较高的场景。

5.4.2 功能描述

密钥派生模块具有以下功能特点：

- 支持调用 HMAC-SHA256/HMAC-SM3 实现 PBKDF2 算法。
- 支持满足 SP800-132 PBKDF2 标准。
- 支持设备密钥和存储密钥派生。

5.5 非对称公钥算法模块

5.5.1 概述

非对称公钥算法模块可以实现非对称密码算法 RSA (Rivest-Shamir-Adleman)、ECC (Elliptic Curve Cryptography)、SM2 的功能。CPU 可以访问 PKE 模块的内部寄存器，可以配置计算参数和读取计算结果或查询模块的工作状态。

5.5.2 功能描述

非对称公钥算法模块具有以下功能特点：

- ECC 支持算法标准：RFC 5639 - BrainpoolP256/384/512；NIST FIPS 186-4 P192/224/256/384/521；RFC 7748 - Curve25519；RFC 8032 - ED25519；
- 支持 RSA1024/2048/3072/4096 公私钥运算。
- 支持国密 SM2。

5.6 随机数生成模块

5.6.1 概述

本模块能够产生真随机数，CPU 可通过系统总线获取随机数。

5.6.2 功能描述

TRNG 模块具有以下功能特点：

- 支持真随机数产生，CPU 通过总线获取，随机数存储 FIFO 的深度为 8。
- 随机数质量符合 AIS-31 标准。

5.7 EFUSE

5.7.1 概述

EFUSE IP 是一种可编程的存储单元，由于其仅可编程一次的特征，多用于芯片保存 Chip ID、密钥或其他一次性存储数据。

5.7.2 功能描述

EFUSE 模块具有以下功能特点：

- 存储容量为 2048bit。
- EFUSE_ARRAY_BIT[2047:0]区域分为 2 个锁定域和 29 个子区域：
 - 锁定区域 0: bit[207:199]: 依次为区域 8 ~ 0 的锁定位。
 - 锁定区域 1: bit[317:304]: 依次为区域 22 ~ 9 的锁定位。
 - 支持进行锁定的子区域有 23 个，子区域 0: bit[303:208] + bit[191:0]; 子区域 1: bit[335:320]; 子区域 2: bit[351:336]; 子区域 3: bit[479:352]; 子区域 4: bit[607:480];
子区域 5: bit[927:672] ; 子区域 6: bit[959:928]; 子区域 7: bit[975:960];
子区域 8: bit[1007:976]; 子区域 9: bit[1023:1008]; 子区域 10:
bit[1039:1024];
子区域 11: bit[1055:1040] ; 子区域 12: bit[1087:1056]; 子区域 13:
bit[1103:1088]; 子区域 14: bit[1119:1104]; 子区域 15: bit[1151:1120]; 子区域 16: bit[1295:1152];

子区域 17: bit[1439:1296]; 子区域 18: bit[1583:1440]; 子区域 19: bit[1631:1584]; 子区域 20: bit[1679:1632]; 子区域 21: bit[1727:1680]; 子区域 22: bit[1775:1728]。

- 若 PG[x]被烧写为 1, 则锁定功能在下次上电后生效, 软件不能再烧写修改对应区域; PG3 和 PG4 被锁定后既不可再烧写, 也不可读取。
- 不支持进行锁定的子区域有 6 个, 子区域 23: bit[1839:1776]; 子区域 24: bit[1903:1840]; 子区域 25: bit[1951:1904]; 子区域 26: bit[198:192]; 子区域 27: bit[319:318]; 子区域 28: bit[671:608]; 子区域 29: bit[2047:1952]; 其中子区域 23 以及 24 为客户预留的区域。

6 外围设备

6.1 IO MUX

6.2 GPIO

6.3 UART

6.4 I2C

6.5 SPI

6.6 PWM

6.7 Tsensor

6.8 I2S

6.9 QSPI

6.10 DMA

6.11 ADC

6.1 IO MUX

6.1.1 概述

芯片数字管脚数量有限，通过 IO 复用的方式丰富管脚功能。

6.1.2 软用管脚描述

须知

ADC 管脚：LSADC 通道与 GPIO 功能只支持其中 1 种功能，使用 ADC 通道管脚时，需要将 GPIO 管脚配置成高阻模式。ADC 通道管脚与 GPIO 管脚的对应关系如表 6-2 所示。

复用配置约束：对于同一个复用信号，只能复用一个 GPIO 管脚，不允许复用到多个 GPIO 管脚。

表6-1 ADC 通道管脚与复用管脚对应关系

复用管脚名称	ADC 管脚
GPIO_07	ADC0
GPIO_08	ADC1
GPIO_09	ADC2
GPIO_10	ADC3
GPIO_11	ADC4
GPIO_12	ADC5

软件复用管脚如表 6-2 所示。

表6-2 软件复用管脚

Pin	Pa d 信 号	复用 控制 寄存 器	复用信 号 0	复用 信号 1	复用 信号 2	复用 信号 3	复用 信号 4	复用 信号 5	复用 信号 6	复 用 信 号 7
4	GP IO_ 00	GPIO _00_S EL	GPIO_ 00	PWM 0	DIA G[0]	SPI1_ CSN	JTAG _TDI	-	-	-
5	GP IO_ 01	GPIO _01_S EL	GPIO_ 01	PWM 1	DIA G[1]	SPI1_ IO0	JTAG _MO DE	BT_ SAM PLE	-	-
6	GP IO_ 02	GPIO _02_S EL	GPIO_ 02	PWM 2	DIA G[2]	SPI1_ IO3	WIFI _TSF _SY _NC	WL_ GLP _SY _NC_ PUL SE	BLE &SL E_G LP_S YNC _PU LSE	-
7	GP IO_ 03	GPIO _03_S EL	GPIO_ 03	PWM 3	PMU _32 K_T EST	SPI1_ IO1	HW_ I D[0]	DIA G[3]	-	-
8	GP IO_ 04	GPIO _04_S	SSI_C LK	PWM 4	GPI O_0	SPI1_ IO1	JTAG _EN	DFT _JT	-	-

Pin	Pad 信号	复用控制寄存器	复用信号 0	复用信号 1	复用信号 2	复用信号 3	复用信号 4	复用信号 5	复用信号 6	复用信号 7
	04	EL			4		ABLE	AG_TMS		
9	GPIO_05	GPIO_05_SEL	SSI_D ATA	PWM 5	UART2_CTS	SPI1_IO2	GPIO_05	SPI0_IN	DFT_JTAG_TCK	-
10	GPIO_06	GPIO_06_SEL	GPIO_06	PWM 6	UART2_RTS	SPI1_SCK	REF_CLK_FREQ_STATUS	DIA G[4]	SPI0_OUT	DFT_JTAG_TDI
11	GPIO_07	GPIO_07_SEL	GPIO_07	PWM 7	UART2_RXD	SPI0_SCK	I2S_MCLK	DIA G[5]	-	-
12	GPIO_08	GPIO_08_SEL	GPIO_08	PWM 0	UART2_TXD	SPI0_CS1_N	DIAG[6]	-	-	-
13	GPIO_09	GPIO_09_SEL	GPIO_09	PWM 1	RADAR_ANT0_SW	SPI0_OUT	I2S_DO	HW_ID[1]	DIAG[7]	JTAG_TDO
14	GPIO_10	GPIO_10_SEL	GPIO_10	PWM 2	ANT0_SW	SPI0_CS0_N	I2S_SCLK	DIA G[0]	-	-
15	GPIO_11	GPIO_11_SEL	GPIO_11	PWM 3	RADAR_ANT1_SW	SPI0_IN	I2S_LRCLK	DIA G[1]	HW_ID[2]	-
16	GPIO_12	GPIO_12_SEL	GPIO_12	PWM 4	ANT1_SW	-	I2S_DI	DIA G[7]	HW_ID[3]	-
24	GPIO_13	GPIO_13_SEL	GPIO_13	UART1_CTS	RADAR_ANT0_SW	DFT_JTAG_TDO	JTAG_TMS	-	-	-
25	GPIO_14	GPIO_14_S	GPIO_14	UART1_RT	RADAR_	DFT_JTAG	JTAG_TCK	-	-	-

Pin	Pa d 信 号	复用 控制 寄存 器	复用信 号 0	复用 信号 1	复用 信号 2	复用 信号 3	复用 信号 4	复用 信号 5	复用 信号 6	复 用 信 号 7
	14	EL		S	ANT 1_S W	_TRS TN				
26	UA RT 1_ TX D	UART 1_TX D_SE L	GPIO_ 15	UART 1_TX D	I2C1 _SD A	-	-	-	-	-
27	UA RT 1_ RX D	UART 1_RX D_SE L	GPIO_ 16	UART 1_RX D	I2C1 _SC L	-	-	-	-	-
28	UA RT 0_ TX D	UART 0_TX D_SE L	GPIO_ 17	UART 0_TX D	I2C0 _SD A	-	-	-	-	-
29	UA RT 0_ RX D	UART 0_RX D_SE L	GPIO_ 18	UART 0_RX D	I2C0 _SC L	-	-	-	-	-
注：Flash 为芯片内置，SFC 管脚无 PIN 序号										

GPIO 的软件复用管脚说明如表 6-3 所示。

表6-3 GPIO 的软件复用管脚说明

信号名	方向	说明
REFCLK_FREQ_STATU S	I	晶体时钟频率的指示信号： 1'b0:40M。 1'b1:24M。
JTAG_ENABLE	I	jtag 使能： 1'b0:普通 IO。

信号名	方向	说明
		1'b1:jtag 使能。
JTAG_MODE	I	DFT 使能： 1'b0:正常功能模式。 1'b1:DFT (Design For Testability) 测试模式。
HW_ID	I	HW_ID (上电硬件控制字)
DFT_JTAG_TDI	I	DFT_JTAG 数据输入。
DFT_JTAG_TRSTN	I	DFT_JTAG 复位输入，低电平有效，默认状态为复位。
DFT_JTAG_TCK	I	DFT_JTAG 时钟输入。
DFT_JTAG_TMS	I	DFT_JTAG 模式选择输入。
DFT_JTAG_TDO	B	DFT_JTAG 数据输出。
JTAG_TDI	I	JTAG 数据输入。
JTAG_TCK	I	JTAG 时钟输入。
JTAG_TMS	B	JTAG 模式选择输入。
JTAG_TDO	B	JTAG 数据输出。
UART0_RXD	I	UART0 RX。
UART0_TXD	O	UART0 TX。
UART1_RXD	I	UART1 RX。
UART1_TXD	O	UART1 TX。
UART1_RTS	O	UART1 流控信号。
UART1_CTS	I	UART1 流控信号。
UART2_RXD	I	UART2 RX。
UART2_TXD	O	UART2 TX。
UART2_RTS	O	UART2 流控信号。

信号名	方向	说明
UART2_CTS	I	UART2 流控信号。
PWM0	B	PWM0 输出。
PWM1	B	PWM1 输出。
PWM2	B	PWM2 输出。
PWM3	B	PWM3 输出。
PWM4	B	PWM4 输出。
PWM5	B	PWM5 输出。
PWM6	B	PWM6 输出。
PWM7	B	PWM7 输出。
GPIO_00	B	GPIO
GPIO_01	B	GPIO
GPIO_02	B	GPIO
GPIO_03	B	GPIO
GPIO_04	B	GPIO
GPIO_05	B	GPIO
GPIO_06	B	GPIO
GPIO_07	B	GPIO
GPIO_08	B	GPIO
GPIO_09	B	GPIO
GPIO_10	B	GPIO
GPIO_11	B	GPIO
GPIO_12	B	GPIO
GPIO_13	B	GPIO
GPIO_14	B	GPIO
GPIO_15	B	GPIO
GPIO_16	B	GPIO
GPIO_17	B	GPIO
GPIO_18	B	GPIO

信号名	方向	说明
SSI_CLK	I	SSI 时钟
SSI_DATA	B	SSI 数据
SPI0_SCK	B	SPI0 时钟信号
SPI0_CS0_N	B	SPI0 片选信号 0 chip select0, active low
SPI0_CS1_N	B	SPI0 片选信号 1 chip select1, active low
SPI0_IN	I	SPI0 数据接收信号 data input
SPI0_OUT	B	SPI0 数据发送信号 data output, with tri-state output
SPI1_SCK	O	QSPI (SPI1 时钟信号)
SPI1_CSN	O	QSPI (SPI1 片选信号)
SPI1_IO0	B	QSPI (SPI1)数据 0。
SPI1_IO1	B	QSPI (SPI1)数据 1。
SPI1_IO2	B	QSPI (SPI1)数据 2。
SPI1_IO3	B	QSPI (SPI1)数据 3。
SFC_CLK	O	Flash 控制信号，不支持双沿。 Flash 时钟范围：CMU 中 PLL 源头时钟二分频产生 96M 或 80M 的时钟。上电使用晶体时钟二分频：20M 或 12M。
SFC_CSN	O	Flash 控制信号，不支持双沿，默认上拉。
SFC_IO0	B	Flash 数据信号，不支持双沿，默认上拉。
SFC_IO1	B	Flash 数据信号，不支持双沿，默认上拉。
SFC_IO2	B	Flash 数据信号，不支持双沿，默认上拉。
SFC_IO3	B	Flash 数据信号，不支持双沿，默认上拉。
I2C0_SCL	B	I2C 时钟
I2C0_SDA	B	I2C 数据

信号名	方向	说明
I2C1_SCL	B	I2C 时钟
I2C1_SDA	B	I2C 数据
I2S_MCLK	O	I2S MCLK
I2S_SCLK	B	I2S CLK
I2S_LRCLK	B	I2S WS
I2S_DI	I	I2S RX
I2S_DO	O	I2S TX
WIFI_TSF_SYNC	O	WiFi 输出的音频同步信号。
WL_GLP_SYNC_PULSE	O	WiFi 输出的 GLP 同步信号。
BLE&SLE_GLP_SYNC_PULSE	O	BLE&SLE 输出的 GLP 同步信号。
BT_SAMPLE	I	BT 维测信号。
DIAG	O	内部时钟、信号的维测，端口观测。
PMU_32K_TEST	O	PMU 32k 时钟观测。
ANT0_SW	O	WiFi 输出的物理天线选择信号 0。
ANT1_SW	O	WiFi 输出的物理天线选择信号 1。
RADAR_ANT0_SW	O	感知输出的物理天线选择信号 0。
RADAR_ANT1_SW	O	感知输出的物理天线选择信号 1。

6.1.3 寄存器概览

IO_CONFIG 寄存器概览如表 6-4 所示。

表6-4 IO_CONFIG 寄存器概览（基址是 0x4400_d000）

偏移地址	名称	描述
0x0	GPIO_00_SEL	GPIO_00 管脚复用控制寄存器。
0x4	GPIO_01_SEL	GPIO_01 管脚复用控制寄存器。

偏移地址	名称	描述
0x8	GPIO_02_SEL	GPIO_02 管脚复用控制寄存器。
0xc	GPIO_03_SEL	GPIO_03 管脚复用控制寄存器。
0x10	GPIO_04_SEL	GPIO_04 管脚复用控制寄存器。
0x14	GPIO_05_SEL	GPIO_05 管脚复用控制寄存器。
0x18	GPIO_06_SEL	GPIO_06 管脚复用控制寄存器。
0x1c	GPIO_07_SEL	GPIO_07 管脚复用控制寄存器。
0x20	GPIO_08_SEL	GPIO_08 管脚复用控制寄存器。
0x24	GPIO_09_SEL	GPIO_09 管脚复用控制寄存器。
0x28	GPIO_10_SEL	GPIO_10 管脚复用控制寄存器。
0x2c	GPIO_11_SEL	GPIO_11 管脚复用控制寄存器。
0x30	GPIO_12_SEL	GPIO_12 管脚复用控制寄存器。
0x34	GPIO_13_SEL	GPIO_13 管脚复用控制寄存器。
0x38	GPIO_14_SEL	GPIO_14 管脚复用控制寄存器。
0x3c	UART1_TXD_SEL	UART1_TXD 管脚复用控制寄存器。
0x40	UART1_RXD_SEL	UART1_RXD 管脚复用控制寄存器。
0x44	UART0_TXD_SEL	UART0_TXD 管脚复用控制寄存器。
0x48	UART0_RXD_SEL	UART0_RXD 管脚复用控制寄存器。
0x800	PAD_GPIO_00_CTRL	GPIO_00 功能管脚控制寄存器。
0x804	PAD_GPIO_01_CTRL	GPIO_01 功能管脚控制寄存器。
0x808	PAD_GPIO_02_CTRL	GPIO_02 功能管脚控制寄存器。
0x80c	PAD_GPIO_03_CTRL	GPIO_03 功能管脚控制寄存器。
0x810	PAD_GPIO_04_CTRL	GPIO_04 功能管脚控制寄存器。
0x814	PAD_GPIO_05_CTRL	GPIO_05 功能管脚控制寄存器。
0x818	PAD_GPIO_06_CTRL	GPIO_06 功能管脚控制寄存器。

偏移地址	名称	描述
0x81c	PAD_GPIO_07_CTRL	GPIO_07 功能管脚控制寄存器。
0x820	PAD_GPIO_08_CTRL	GPIO_08 功能管脚控制寄存器。
0x824	PAD_GPIO_09_CTRL	GPIO_09 功能管脚控制寄存器。
0x828	PAD_GPIO_10_CTRL	GPIO_10 功能管脚控制寄存器。
0x82c	PAD_GPIO_11_CTRL	GPIO_11 功能管脚控制寄存器。
0x830	PAD_GPIO_12_CTRL	GPIO_12 功能管脚控制寄存器。
0x834	PAD_GPIO_13_CTRL	GPIO_13 功能管脚控制寄存器。
0x838	PAD_GPIO_14_CTRL	GPIO_14 功能管脚控制寄存器。
0x83c	PAD_UART1_TXD_CTRL	UART1_TXD 功能管脚控制寄存器。
0x840	PAD_UART1_RXD_CTRL	UART1_RXD 功能管脚控制寄存器。
0x844	PAD_UART0_TXD_CTRL	UART0_TXD 功能管脚控制寄存器。
0x848	PAD_UART0_RXD_CTRL	UART0_RXD 功能管脚控制寄存器。
0x868	PAD_SFC_CLK_CTRL	SFC_CLK 功能管脚控制寄存器。
0x86c	PAD_SFC_CSN_CTRL	SFC_CSN 功能管脚控制寄存器。
0x870	PAD_SFC_IO0_CTRL	SFC_IO0 功能管脚控制寄存器。
0x874	PAD_SFC_IO1_CTRL	SFC_IO1 功能管脚控制寄存器。
0x878	PAD_SFC_IO2_CTRL	SFC_IO2 功能管脚控制寄存器。
0x87c	PAD_SFC_IO3_CTRL	SFC_IO3 功能管脚控制寄存器。

6.1.4 寄存器描述

GPIO_00_SEL

GPIO_00_SEL 为 GPIO_00 复用关系配置。

Offset Address: 0x0 Total Reset Value: 0x0000_0000

Bits	Access	Name	Description	Reset
[31:3]	-	reserved	保留。	0x00000000
[2:0]	RW	gpio_00_sel	GPIO_00 管脚复用: 0: GPIO_00; 1: PWM0; 2: DIAG[0]; 3: SPI1_CSN; 4: JTAG_TDI; 其他: 保留。	0x0

GPIO_01_SEL

GPIO_01_SEL 为 GPIO_01 复用关系配置。

Offset Address: 0x4 Total Reset Value: 0x0000_0000

Bits	Access	Name	Description	Reset
[31:3]	-	reserved	保留。	0x00000000
[2:0]	RW	gpio_01_sel	GPIO_01 管脚复用: 0: GPIO_01; 1: PWM1; 2: DIAG[1]; 3: SPI1_IO0; 4: JTAG_MODE; 5: BT_SAMPLE; 其他: 保留。	0x0

GPIO_02_SEL

GPIO_02_SEL 为 GPIO_02 复用关系配置。

Offset Address: 0x8 Total Reset Value: 0x0000_0000

Bits	Access	Name	Description	Reset
[31:3]	-	reserved	保留。	0x00000000
[2:0]	RW	gpio_02_sel	GPIO_02 管脚复用： 0: GPIO_02; 1: PWM2; 2: DIAG[2]; 3: SPI1_IO3; 4: WIFI_TSF_SYNC; 5: WL_GLP_SYNC_PULSE; 6: BSLE_GLP_SYNC_PULSE; 其他：保留。	0x0

GPIO_03_SEL

GPIO_03_SEL 为 GPIO_03 复用关系配置。

Offset Address: 0xc Total Reset Value: 0x0000_0000

Bits	Access	Name	Description	Reset
[31:3]	-	reserved	保留。	0x00000000
[2:0]	RW	gpio_03_sel	GPIO_03 管脚复用： 0: GPIO_03; 1: PWM3; 2: PMU_32K_TEST; 3: SPI1_IO1; 4: HW_ID[0];	0x0

Bits	Access	Name	Description	Reset
			5: DIAG[3]; 其他: 保留。	

GPIO_04_SEL

GPIO_04_SEL 为 GPIO_04 复用关系配置。

Offset Address: 0x10 Total Reset Value: 0x0000_0000

Bits	Access	Name	Description	Reset
[31:3]	-	reserved	保留。	0x00000000
[2:0]	RW	gpio_04_sel	GPIO_04 管脚复用: 0: SSI_CLK; 1: PWM4; 2: GPIO_04; 3: SPI1_IO1; 4: JTAG_ENABLE; 5: DFT_JTAG_TMS; 其他: 保留。	0x0

GPIO_05_SEL

GPIO_05_SEL 为 GPIO_05 复用关系配置。

Offset Address: 0x14 Total Reset Value: 0x0000_0000

Bits	Access	Name	Description	Reset
[31:3]	-	reserved	保留。	0x00000000
[2:0]	RW	gpio_05_sel	GPIO_05 管脚复用: 0: SSI_DATA;	0x0

Bits	Access	Name	Description	Reset
			1: PWM5; 2: UART2_CTS; 3: SPI1_IO2; 4: GPIO_05; 5: SPI0_IN; 6: DFT_JTAG_TCK; 其他: 保留。	

GPIO_06_SEL

GPIO_06_SEL 为 GPIO_06 复用关系配置。

Offset Address: 0x18 Total Reset Value: 0x0000_0000

Bits	Access	Name	Description	Reset
[31:3]	-	reserved	保留。	0x00000000
[2:0]	RW	gpio_06_sel	GPIO_06 管脚复用: 0: GPIO_06; 1: PWM6; 2: UART2_RTS; 3: SPI1_SCK; 4: REFCLK_FREQ_STATUS; 5: DIAG[4]; 6: SPI0_OUT; 7: DFT_JTAG_TDI; 其他: 保留。	0x0

GPIO_07_SEL

GPIO_07_SEL 为 GPIO_07 复用关系配置。

Offset Address: 0x1c Total Reset Value: 0x0000_0000

Bits	Access	Name	Description	Reset
[31:3]	-	reserved	保留。	0x00000000
[2:0]	RW	gpio_07_sel	GPIO_07 管脚复用： 0: GPIO_07; 1: PWM7; 2: UART2_RXD; 3: SPI0_SCK; 4: I2S_MCLK; 5: DIAG[5]; 其他：保留。	0x0

GPIO_08_SEL

GPIO_08_SEL 为 GPIO_08 复用关系配置。

Offset Address: 0x20 Total Reset Value: 0x0000_0000

Bits	Access	Name	Description	Reset
[31:3]	-	reserved	保留。	0x00000000
[2:0]	RW	gpio_08_sel	GPIO_08 管脚复用： 0: GPIO_08; 1: PWM0; 2: UART2_TXD; 3: SPI0_CS1_N; 4: DIAG[6]; 其他：保留。	0x0

GPIO_09_SEL

GPIO_09_SEL 为 GPIO_09 复用关系配置

Offset Address: 0x24 Total Reset Value: 0x0000_0000

Bits	Access	Name	Description	Reset
[31:3]	-	reserved	保留。	0x00000000
[2:0]	RW	gpio_09_sel	GPIO_09 管脚复用： 0: GPIO_09; 1: PWM1; 2: RADAR_ANT0_SW; 3: SPI0_OUT; 4: I2S_DO; 5: HW_ID[1]; 6: DIAG[7]; 7: JTAG_TDO; 其他：保留。	0x0

GPIO_10_SEL

GPIO_10_SEL 为 GPIO_10 复用关系配置。

Offset Address: 0x28 Total Reset Value: 0x0000_0000

Bits	Access	Name	Description	Reset
[31:3]	-	reserved	保留。	0x00000000
[2:0]	RW	gpio_10_sel	GPIO_10 管脚复用： 0: GPIO_10; 1: PWM2; 2: ANT0_SW; 3: SPI0_CS0_N;	0x0

Bits	Access	Name	Description	Reset
			4: I2S_SCLK; 5: DIAG[0]; 其他: 保留。	

GPIO_11_SEL

GPIO_11_SEL 为 GPIO_11 复用关系配置。

Offset Address: 0x2c Total Reset Value: 0x0000_0000

Bits	Access	Name	Description	Reset
[31:3]	-	reserved	保留。	0x00000000
[2:0]	RW	gpio_11_sel	GPIO_11 管脚复用: 0: GPIO_11; 1: PWM3; 2: RADAR_ANT1_SW; 3: SPI0_IN; 4: I2S_LRCLK; 5: DIAG[1]; 6: HW_ID[2]; 其他: 保留。	0x0

GPIO_12_SEL

GPIO_12_SEL 为 GPIO_12 复用关系配置。

Offset Address: 0x30 Total Reset Value: 0x0000_0000

Bits	Access	Name	Description	Reset
[31:3]	-	reserved	保留。	0x00000000

Bits	Access	Name	Description	Reset
[2:0]	RW	gpio_12_sel	GPIO_12 管脚复用： 0: GPIO_12; 1: PWM4; 2: ANT1_SW; 4: I2S_DI; 6: HW_ID[3]; 其他：保留。	0x0

GPIO_13_SEL

GPIO_13_SEL 为 GPIO_13 复用关系配置。

Offset Address: 0x34 Total Reset Value: 0x0000_0000

Bits	Access	Name	Description	Reset
[31:3]	-	reserved	保留。	0x00000000
[2:0]	RW	gpio_13_sel	GPIO_13 管脚复用： 0: GPIO_13; 1: UART1_CTS; 2: RADAR_ANT0_SW; 3: DFT_JTAG_TDO; 4: JTAG_TMS; 其他：保留。	0x0

GPIO_14_SEL

GPIO_14_SEL 为 GPIO_14 复用关系配置。

Offset Address: 0x38 Total Reset Value: 0x0000_0000

Bits	Access	Name	Description	Reset
------	--------	------	-------------	-------

Bits	Access	Name	Description	Reset
[31:3]	-	reserved	保留。	0x00000000
[2:0]	RW	gpio_14_sel	GPIO_14 管脚复用： 0: GPIO_14; 1: UART1_RTS; 2: RADAR_ANT1_SW; 3: DFT_JTAG_TRSTN; 4: JTAG_TCK; 其他：保留。	0x0

UART1_TXD_SEL

UART1_TXD_SEL 为 UART1_TXD 复用关系配置。

Offset Address: 0x3c Total Reset Value: 0x0000_0000

Bits	Access	Name	Description	Reset
[31:2]	-	reserved	保留。	0x00000000
[1:0]	RW	uart1_txd_sel	UART1_TXD 管脚复用： 0: GPIO_15; 1: UART1_TXD; 2: I2C1_SDA; 其他：保留。	0x0

UART1_RXD_SEL

UART1_RXD_SEL 为 UART1_RXD 复用关系配置。

Offset Address: 0x40 Total Reset Value: 0x0000_0000

Bits	Access	Name	Description	Reset
------	--------	------	-------------	-------

Bits	Access	Name	Description	Reset
[31:2]	-	reserved	保留。	0x00000000
[1:0]	RW	uart1_rxd_sel	UART1_RXD 管脚复用： 0: GPIO_16; 1: UART1_RXD; 2: I2C1_SCL; 其他：保留。	0x0

UART0_TXD_SEL

UART0_TXD_SEL 为 UART0_TXD 复用关系配置。

Offset Address: 0x44 Total Reset Value: 0x0000_0000

Bits	Access	Name	Description	Reset
[31:2]	-	reserved	保留。	0x00000000
[1:0]	RW	uart0_txd_sel	UART0_TXD 管脚复用： 0: GPIO_17; 1: UART0_TXD; 2: I2C0_SDA; 其他：保留。	0x0

UART0_RXD_SEL

UART0_RXD_SEL 为 UART0_RXD 复用关系配置。

Offset Address: 0x48 Total Reset Value: 0x0000_0000

Bits	Access	Name	Description	Reset
[31:2]	-	reserved	保留。	0x00000000

Bits	Access	Name	Description	Reset
[1:0]	RW	uart0_rxd_sel	UART0_RXD 管脚复用： 0: GPIO_18; 1: UART0_RXD; 2: I2C0_SCL; 其他：保留。	0x0

PAD_GPIO_00_CTRL

PAD_GPIO_00_CTRL 为 GPIO_00 控制寄存器。

Offset Address: 0x800 Total Reset Value: 0x0000_0800

Bits	Access	Name	Description	Reset
[31:12]	-	reserved	保留。	0x00000
[11]	RW	pad_gpio_00_ctrl_ie	GPIO_00.IE 管脚控制： 0: 输入禁止； 1: 输入使能。	0x1
[10]	RW	pad_gpio_00_ctrl_ps	GPIO_00.PS 管脚控制： 需与 GPIO_00.PE 管脚控制结合使用。 PE/PS 对应上下拉能力控制如下： 00: 无上下拉； 01: 强上拉； 10: 下拉； 11: 上拉。	0x0
[9]	RW	pad_gpio_00_ctrl_pe	GPIO_00.PE 管脚控制。	0x0
[8:7]	-	reserved	保留。	0x0
[6]	RW	pad_gpio_00_ctrl_ds2	GPIO_00.DS2 管脚控制：	0x0

Bits	Access	Name	Description	Reset
			需与 GPIO_00.DS1/GPIO_00.DS0 管脚控制结合使用。 DS2~DS0 对应驱动能力调节： 111~000，驱动能力依次减弱。	
[5]	RW	pad_gpio_00_ctrl_ds1	GPIO_00.DS1 管脚控制。	0x0
[4]	RW	pad_gpio_00_ctrl_ds0	GPIO_00.DS0 管脚控制。	0x0
[3]	RW	pad_gpio_00_ctrl_st	GPIO_00.ST 管脚控制： 0: No Schmitt; 1: Schmitt Enable。	0x0
[2:0]	-	reserved	保留。	0x0

PAD_GPIO_01_CTRL

PAD_GPIO_01_CTRL 为 GPIO_01 控制寄存器。

Offset Address: 0x804 Total Reset Value: 0x0000_0A00

Bits	Access	Name	Description	Reset
[31:12]	-	reserved	保留。	0x00000
[11]	RW	pad_gpio_01_ctrl_ie	GPIO_01.IE 管脚控制： 0: 输入禁止； 1: 输入使能。	0x1
[10]	RW	pad_gpio_01_ctrl_ps	GPIO_01.PS 管脚控制： 需与 GPIO_01.PE 管脚控制结合使用。 PE/PS 对应上下拉能力控制如下： 00: 无上下拉；	0x0

Bits	Access	Name	Description	Reset
			01: 强上拉; 10: 下拉; 11: 上拉。	
[9]	RW	pad_gpio_01_ctrl_pe	GPIO_01.PE 管脚控制。	0x1
[8:7]	-	reserved	保留。	0x0
[6]	RW	pad_gpio_01_ctrl_ds2	GPIO_01.DS2 管脚控制: 需与 GPIO_01.DS1/GPIO_01.DS0 管脚控制结合使用。 DS2~DS0 对应驱动能力调节: 111 至 000, 驱动能力依次减弱。	0x0
[5]	RW	pad_gpio_01_ctrl_ds1	GPIO_01.DS1 管脚控制。	0x0
[4]	RW	pad_gpio_01_ctrl_ds0	GPIO_01.DS0 管脚控制。	0x0
[3]	RW	pad_gpio_01_ctrl_st	GPIO_01.ST 管脚控制: 0: No Schmitt; 1: Schmitt Enable。	0x0
[2:0]	-	reserved	保留。	0x0

PAD_GPIO_02_CTRL

PAD_GPIO_02_CTRL 为 GPIO_02 控制寄存器。

Offset Address: 0x808 Total Reset Value: 0x0000_0800

Bits	Access	Name	Description	Reset
[31:12]	-	reserved	保留。	0x00000
[11]	RW	pad_gpio_02_ctrl_ie	GPIO_02.IE 管脚控制:	0x1

Bits	Access	Name	Description	Reset
			0: 输入禁止; 1: 输入使能。	
[10]	RW	pad_gpio_02_ctrl_ps	GPIO_02.PS 管脚控制: 需与 GPIO_02.PE 管脚控制结合使用。 PE/PS 对应上下拉能力控制如下: 00: 无上下拉; 01: 强上拉; 10: 下拉; 11: 上拉。	0x0
[9]	RW	pad_gpio_02_ctrl_pe	GPIO_02.PE 管脚控制。	0x0
[8:7]	-	reserved	保留。	0x0
[6]	RW	pad_gpio_02_ctrl_ds2	GPIO_02.DS2 管脚控制: 需与 GPIO_02.DS1/GPIO_02.DS0 管脚控制结合使用。 DS2~DS0 对应驱动能力调节: 111~000, 驱动能力依次减弱。	0x0
[5]	RW	pad_gpio_02_ctrl_ds1	GPIO_02.DS1 管脚控制。	0x0
[4]	RW	pad_gpio_02_ctrl_ds0	GPIO_02.DS0 管脚控制。	0x0
[3]	RW	pad_gpio_02_ctrl_st	GPIO_02.ST 管脚控制: 0: No Schmitt; 1: Schmitt Enable。	0x0
[2:0]	-	reserved	保留。	0x0

PAD_GPIO_03_CTRL

PAD_GPIO_03_CTRL 为 GPIO_03 控制寄存器。

Offset Address: 0x80c Total Reset Value: 0x0000_0A00

Bits	Access	Name	Description	Reset
[31:12]	-	reserved	保留。	0x00000
[11]	RW	pad_gpio_03_ctrl_ie	GPIO_03.IE 管脚控制： 0：输入禁止； 1：输入使能。	0x1
[10]	RW	pad_gpio_03_ctrl_ps	GPIO_03.PS 管脚控制： 需与 GPIO_03.PE 管脚控制结合使用。 PE/PS 对应上下拉能力控制如下： 00：无上下拉； 01：强上拉； 10：下拉； 11：上拉。	0x0
[9]	RW	pad_gpio_03_ctrl_pe	GPIO_03.PE 管脚控制。	0x1
[8:7]	-	reserved	保留。	0x0
[6]	RW	pad_gpio_03_ctrl_ds2	GPIO_03.DS2 管脚控制： 需与 GPIO_03.DS1/GPIO_03.DS0 管脚控制结合使用。 DS2~DS0 对应驱动能力调节： 111 至 000，驱动能力依次减弱。	0x0
[5]	RW	pad_gpio_03_ctrl_ds1	GPIO_03.DS1 管脚控制。	0x0
[4]	RW	pad_gpio_03_ctrl_ds0	GPIO_03.DS0 管脚控制。	0x0
[3]	RW	pad_gpio_03_c	GPIO_03.ST 管脚控制：	0x0

Bits	Access	Name	Description	Reset
		trl_st	0: No Schmitt; 1: Schmitt Enable。	
[2:0]	-	reserved	保留。	0x0

PAD_GPIO_04_CTRL

PAD_GPIO_04_CTRL 为 GPIO_04 控制寄存器。

Offset Address: 0x810 Total Reset Value: 0x0000_0A00

Bits	Access	Name	Description	Reset
[31:12]	-	reserved	保留。	0x00000
[11]	RW	pad_gpio_04_ctrl_ie	GPIO_04.IE 管脚控制： 0: 输入禁止； 1: 输入使能。	0x1
[10]	RW	pad_gpio_04_ctrl_ps	GPIO_04.PS 管脚控制： 需与 GPIO_04.PE 管脚控制结合使用。 PE/PS 对应上下拉能力控制如下： 00: 无上下拉； 01: 强上拉； 10: 下拉； 11: 上拉。	0x0
[9]	RW	pad_gpio_04_ctrl_pe	GPIO_04.PE 管脚控制。	0x1
[8:7]	-	reserved	保留。	0x0
[6]	RW	pad_gpio_04_ctrl_ds2	GPIO_04.DS2 管脚控制： 需与 GPIO_04.DS1/GPIO_04.DS0 管	0x0

Bits	Access	Name	Description	Reset
			脚控制结合使用。 DS2~DS0 对应驱动能力调节： 111~000，驱动能力依次减弱。	
[5]	RW	pad_gpio_04_ctrl_ds1	GPIO_04.DS1 管脚控制。	0x0
[4]	RW	pad_gpio_04_ctrl_ds0	GPIO_04.DS0 管脚控制。	0x0
[3]	RW	pad_gpio_04_ctrl_st	GPIO_04.ST 管脚控制： 0：No Schmitt； 1：Schmitt Enable。	0x0
[2:0]	-	reserved	保留。	0x0

PAD_GPIO_05_CTRL

PAD_GPIO_05_CTRL 为 GPIO_05 控制寄存器。

Offset Address: 0x814 Total Reset Value: 0x0000_0820

Bits	Access	Name	Description	Reset
[31:12]	-	reserved	保留。	0x00000
[11]	RW	pad_gpio_05_ctrl_ie	GPIO_05.IE 管脚控制： 0：输入禁止； 1：输入使能。	0x1
[10]	RW	pad_gpio_05_ctrl_ps	GPIO_05.PS 管脚控制： 需与 GPIO_05.PE 管脚控制结合使用。 PE/PS 对应上下拉能力控制如下： 00：无上下拉； 01：强上拉； 10：下拉；	0x0

Bits	Access	Name	Description	Reset
			11: 上拉。	
[9]	RW	pad_gpio_05_ctrl_pe	GPIO_05.PE 管脚控制。	0x0
[8:7]	-	reserved	保留。	0x0
[6]	RW	pad_gpio_05_ctrl_ds2	GPIO_05.DS2 管脚控制： 需与 GPIO_05.DS1/GPIO_05.DS0 管脚控制结合使用。 DS2~DS0 对应驱动能力调节： 111~000，驱动能力依次减弱。	0x0
[5]	RW	pad_gpio_05_ctrl_ds1	GPIO_05.DS1 管脚控制。	0x1
[4]	RW	pad_gpio_05_ctrl_ds0	GPIO_05.DS0 管脚控制。	0x0
[3]	RW	pad_gpio_05_ctrl_st	GPIO_05.ST 管脚控制： 0: No Schmitt; 1: Schmitt Enable。	0x0
[2:0]	-	reserved	保留。	0x0

PAD_GPIO_06_CTRL

PAD_GPIO_06_CTRL 为 GPIO_06 控制寄存器。

Offset Address: 0x818 Total Reset Value: 0x0000_0A00

Bits	Access	Name	Description	Reset
[31:12]	-	reserved	保留。	0x00000
[11]	RW	pad_gpio_06_ctrl_ie	GPIO_06.IE 管脚控制： 0: 输入禁止； 1: 输入使能。	0x1

Bits	Access	Name	Description	Reset
[10]	RW	pad_gpio_06_ctrl_ps	GPIO_06.PS 管脚控制： 需与 GPIO_06.PE 管脚控制结合使用。 PE/PS 对应上下拉能力控制如下： 00：无上下拉； 01：强上拉； 10：下拉； 11：上拉。	0x0
[9]	RW	pad_gpio_06_ctrl_pe	GPIO_06.PE 管脚控制。	0x1
[8:7]	-	reserved	保留。	0x0
[6]	RW	pad_gpio_06_ctrl_ds2	GPIO_06.DS2 管脚控制： 需与 GPIO_06.DS1/GPIO_06.DS0 管脚控制结合使用。 DS2~DS0 对应驱动能力调节： 111~000，驱动能力依次减弱。	0x0
[5]	RW	pad_gpio_06_ctrl_ds1	GPIO_06.DS1 管脚控制。	0x0
[4]	RW	pad_gpio_06_ctrl_ds0	GPIO_06.DS0 管脚控制。	0x0
[3]	RW	pad_gpio_06_ctrl_st	GPIO_06.ST 管脚控制： 0：No Schmitt； 1：Schmitt Enable。	0x0
[2:0]	-	reserved	保留。	0x0

PAD_GPIO_07_CTRL

PAD_GPIO_07_CTRL 为 GPIO_07 控制寄存器。

Offset Address: 0x81c Total Reset Value: 0x0000_0800

Bits	Access	Name	Description	Reset
[31:12]	-	reserved	保留。	0x00000
[11]	RW	pad_gpio_07_ctrl_ie	GPIO_07.IE 管脚控制： 0：输入禁止； 1：输入使能。	0x1
[10]	RW	pad_gpio_07_ctrl_ps	GPIO_07.PS 管脚控制： 需与 GPIO_07.PE 管脚控制结合使用。 PE/PS 对应上下拉能力控制如下： 00：无上下拉； 01：强上拉； 10：下拉； 11：上拉。	0x0
[9]	RW	pad_gpio_07_ctrl_pe	GPIO_07.PE 管脚控制。	0x0
[8:7]	-	reserved	保留。	0x0
[6]	RW	pad_gpio_07_ctrl_ds2	GPIO_07.DS2 管脚控制： 需与 GPIO_07.DS1/GPIO_07.DS0 管脚控制结合使用。 DS2~DS0 对应驱动能力调节： 111~000，驱动能力依次减弱。	0x0
[5]	RW	pad_gpio_07_ctrl_ds1	GPIO_07.DS1 管脚控制。	0x0
[4]	RW	pad_gpio_07_ctrl_ds0	GPIO_07.DS0 管脚控制。	0x0
[3]	RW	pad_gpio_07_ctrl_st	GPIO_07.ST 管脚控制： 0：No Schmitt； 1：Schmitt Enable。	0x0
[2:0]	-	reserved	保留。	0x0

PAD_GPIO_08_CTRL

PAD_GPIO_08_CTRL 为 GPIO_08 控制寄存器。

Offset Address: 0x820 Total Reset Value: 0x0000_0800

Bits	Access	Name	Description	Reset
[31:12]	-	reserved	保留。	0x00000
[11]	RW	pad_gpio_08_ctrl_ie	GPIO_08.IE 管脚控制： 0：输入禁止； 1：输入使能。	0x1
[10]	RW	pad_gpio_08_ctrl_ps	GPIO_08.PS 管脚控制： 需与 GPIO_08.PE 管脚控制结合使用。 PE/PS 对应上下拉能力控制如下： 00：无上下拉； 01：强上拉； 10：下拉； 11：上拉。	0x0
[9]	RW	pad_gpio_08_ctrl_pe	GPIO_08.PE 管脚控制。	0x0
[8:7]	-	reserved	保留。	0x0
[6]	RW	pad_gpio_08_ctrl_ds2	GPIO_08.DS2 管脚控制： 需与 GPIO_08.DS1/GPIO_08.DS0 管脚控制结合使用。 DS2~DS0 对应驱动能力调节： 111~000，驱动能力依次减弱。	0x0
[5]	RW	pad_gpio_08_ctrl_ds1	GPIO_08.DS1 管脚控制。	0x0
[4]	RW	pad_gpio_08_ctrl_ds0	GPIO_08.DS0 管脚控制。	0x0

Bits	Access	Name	Description	Reset
[3]	RW	pad_gpio_08_ctrl_st	GPIO_08.ST 管脚控制: 0: No Cchmitt; 1: Schmitt tEnable。	0x0
[2:0]	-	reserved	保留。	0x0

PAD_GPIO_09_CTRL

PAD_GPIO_09_CTRL 为 GPIO_09 控制寄存器。

Offset Address: 0x824 Total Reset Value: 0x0000_0A00

Bits	Access	Name	Description	Reset
[31:12]	-	reserved	保留。	0x00000
[11]	RW	pad_gpio_09_ctrl_ie	GPIO_09.IE 管脚控制: 0: 输入禁止; 1: 输入使能。	0x1
[10]	RW	pad_gpio_09_ctrl_ps	GPIO_09.PS 管脚控制: 需与 GPIO_09.PE 管脚控制结合使用。 PE/PS 对应上下拉能力控制如下: 00: 无上下拉; 01: 强上拉; 10: 下拉; 11: 上拉。	0x0
[9]	RW	pad_gpio_09_ctrl_pe	GPIO_09.PE 管脚控制。	0x1
[8:7]	-	reserved	保留。	0x0
[6]	RW	pad_gpio_09_ctrl_ds2	GPIO_09.DS2 管脚控制: 需与	0x0

Bits	Access	Name	Description	Reset
			GPIO_09.DS1/GPIO_09.DS0 管脚控制结合使用。 DS2~DS0 对应驱动能力调节： 111~000，驱动能力依次减弱。	
[5]	RW	pad_gpio_09_ctrl_ds1	GPIO_09.DS1 管脚控制。	0x0
[4]	RW	pad_gpio_09_ctrl_ds0	GPIO_09.DS0 管脚控制。	0x0
[3]	RW	pad_gpio_09_ctrl_st	GPIO_09.ST 管脚控制： 0: No Schmitt; 1: Schmitt Enable。	0x0
[2:0]	-	reserved	保留。	0x0

PAD_GPIO_10_CTRL

PAD_GPIO_10_CTRL 为 GPIO_10 控制寄存器。

Offset Address: 0x828 Total Reset Value: 0x0000_0800

Bits	Access	Name	Description	Reset
[31:12]	-	reserved	保留。	0x00000
[11]	RW	pad_gpio_10_ctrl_ie	GPIO_10.IE 管脚控制： 0: 输入禁止； 1: 输入使能。	0x1
[10]	RW	pad_gpio_10_ctrl_ps	GPIO_10.PS 管脚控制： 需与 GPIO_10.PE 管脚控制结合使用。 PE/PS 对应上下拉能力控制如下： 00: 无上下拉； 01: 强上拉；	0x0

Bits	Access	Name	Description	Reset
			10: 下拉; 11: 上拉。	
[9]	RW	pad_gpio_10_ctrl_pe	GPIO_10.PE 管脚控制。	0x0
[8:7]	-	reserved	保留。	0x0
[6]	RW	pad_gpio_10_ctrl_ds2	GPIO_10.DS2 管脚控制: 需与 GPIO_10.DS1/GPIO_10.DS0 管脚控制结合使用。 DS2~DS0 对应驱动能力调节: 111~000, 驱动能力依次减弱。	0x0
[5]	RW	pad_gpio_10_ctrl_ds1	GPIO_10.DS1 管脚控制。	0x0
[4]	RW	pad_gpio_10_ctrl_ds0	GPIO_10.DS0 管脚控制。	0x0
[3]	RW	pad_gpio_10_ctrl_st	GPIO_10.ST 管脚控制: 0: No Schmitt; 1: Schmitt Enable。	0x0
[2:0]	-	reserved	保留。	0x0

PAD_GPIO_11_CTRL

PAD_GPIO_11_CTRL 为 GPIO_11 控制寄存器。

Offset Address: 0x82c Total Reset Value: 0x0000_0A00

Bits	Access	Name	Description	Reset
[31:12]	-	reserved	保留。	0x00000
[11]	RW	pad_gpio_11_ctrl_ie	GPIO_11.IE 管脚控制: 0: 输入禁止;	0x1

Bits	Access	Name	Description	Reset
			1: 输入使能。	
[10]	RW	pad_gpio_11_ctrl_ps	GPIO_11.PS 管脚控制: 需与 GPIO_11.PE 管脚控制结合使用。 PE/PS 对应上下拉能力控制如下: 00: 无上下拉; 01: 强上拉; 10: 下拉; 11: 上拉。	0x0
[9]	RW	pad_gpio_11_ctrl_pe	GPIO_11.PE 管脚控制。	0x1
[8:7]	-	reserved	保留。	0x0
[6]	RW	pad_gpio_11_ctrl_ds2	GPIO_11.DS2 管脚控制: 需与 GPIO_11.DS1/GPIO_11.DS0 管脚控制结合使用。 DS2~DS0 对应驱动能力调节: 111~000, 驱动能力依次减弱。	0x0
[5]	RW	pad_gpio_11_ctrl_ds1	GPIO_11.DS1 管脚控制。	0x0
[4]	RW	pad_gpio_11_ctrl_ds0	GPIO_11.DS0 管脚控制。	0x0
[3]	RW	pad_gpio_11_ctrl_st	GPIO_11.ST 管脚控制: 0: No Schmitt; 1: Schmitt Enable。	0x0
[2:0]	-	reserved	保留。	0x0

PAD_GPIO_12_CTRL

PAD_GPIO_12_CTRL 为 GPIO_12 控制寄存器。

Offset Address: 0x830 Total Reset Value: 0x0000_0A00

Bits	Access	Name	Description	Reset
[31:12]	-	reserved	保留。	0x00000
[11]	RW	pad_gpio_12_ctrl_ie	GPIO_12.IE 管脚控制： 0：输入禁止； 1：输入使能。	0x1
[10]	RW	pad_gpio_12_ctrl_ps	GPIO_12.PS 管脚控制： 需与 GPIO_12.PE 管脚控制结合使用。 PE/PS 对应上下拉能力控制如下： 00：无上下拉； 01：强上拉； 10：下拉； 11：上拉。	0x0
[9]	RW	pad_gpio_12_ctrl_pe	GPIO_12.PE 管脚控制。	0x1
[8:7]	-	reserved	保留。	0x0
[6]	RW	pad_gpio_12_ctrl_ds2	GPIO_12.DS2 管脚控制： 需与 GPIO_12.DS1/GPIO_12.DS0 管脚控制结合使用。 DS2~DS0 对应驱动能力调节： 111~000，驱动能力依次减弱。	0x0
[5]	RW	pad_gpio_12_ctrl_ds1	GPIO_12.DS1 管脚控制。	0x0
[4]	RW	pad_gpio_12_ctrl_ds0	GPIO_12.DS0 管脚控制。	0x0
[3]	RW	pad_gpio_12_ctrl_st	GPIO_12.ST 管脚控制： 0：No Schmitt； 1：Schmitt Enable。	0x0

Bits	Access	Name	Description	Reset
[2:0]	-	reserved	保留。	0x0

PAD_GPIO_13_CTRL

PAD_GPIO_13_CTRL 为 GPIO_13 控制寄存器。

Offset Address: 0x834 Total Reset Value: 0x0000_0800

Bits	Access	Name	Description	Reset
[31:12]	-	reserved	保留。	0x00000
[11]	RW	pad_gpio_13_ctrl_ie	GPIO_13.IE 管脚控制： 0：输入禁止； 1：输入使能。	0x1
[10]	RW	pad_gpio_13_ctrl_ps	GPIO_13.PS 管脚控制： 需与 GPIO_13.PE 管脚控制结合使用。 PE/PS 对应上下拉能力控制如下： 00：无上下拉； 01：强上拉； 10：下拉； 11：上拉。	0x0
[9]	RW	pad_gpio_13_ctrl_pe	GPIO_13.PE 管脚控制。	0x0
[8:7]	-	reserved	保留。	0x0
[6]	RW	pad_gpio_13_ctrl_ds2	GPIO_13.DS2 管脚控制： 需与 GPIO_13.DS1/GPIO_13.DS0 管脚控制结合使用。 DS2~DS0 对应驱动能力调节：	0x0

Bits	Access	Name	Description	Reset
			111~000, 驱动能力依次减弱。	
[5]	RW	pad_gpio_13_ctrl_ds1	GPIO_13.DS1 管脚控制。	0x0
[4]	RW	pad_gpio_13_ctrl_ds0	GPIO_13.DS0 管脚控制。	0x0
[3]	RW	pad_gpio_13_ctrl_st	GPIO_13.ST 管脚控制: 0: No Schmitt; 1: Schmitt Enable。	0x0
[2:0]	-	reserved	保留。	0x0

PAD_GPIO_14_CTRL

PAD_GPIO_14_CTRL 为 GPIO_14 控制寄存器。

Offset Address: 0x838 Total Reset Value: 0x0000_0800

Bits	Access	Name	Description	Reset
[31:12]	-	reserved	保留。	0x00000
[11]	RW	pad_gpio_14_ctrl_ie	GPIO_14.IE 管脚控制: 0: 输入禁止; 1: 输入使能。	0x1
[10]	RW	pad_gpio_14_ctrl_ps	GPIO_14.PS 管脚控制: 需与 GPIO_14.PE 管脚控制结合使用。 PE/PS 对应上下拉能力控制如下: 00: 无上下拉; 01: 强上拉; 10: 下拉; 11: 上拉。	0x0
[9]	RW	pad_gpio_14_c	GPIO_14.PE 管脚控制。	0x0

Bits	Access	Name	Description	Reset
		trl_pe		
[8:7]	-	reserved	保留。	0x0
[6]	RW	pad_gpio_14_ctrl_ds2	GPIO_14.DS2 管脚控制： 需与 GPIO_14.DS1/GPIO_14.DS0 管脚控制结合使用。 DS2~DS0 对应驱动能力调节： 111~000，驱动能力依次减弱。	0x0
[5]	RW	pad_gpio_14_ctrl_ds1	GPIO_14.DS1 管脚控制。	0x0
[4]	RW	pad_gpio_14_ctrl_ds0	GPIO_14.DS0 管脚控制。	0x0
[3]	RW	pad_gpio_14_ctrl_st	GPIO_14.ST 管脚控制： 0: No Schmitt; 1: Schmitt Enable。	0x0
[2:0]	-	reserved	保留。	0x0

PAD_UART1_TXD_CTRL

PAD_UART1_TXD_CTRL 为 UART1_TXD 控制寄存器。

Offset Address: 0x83c Total Reset Value: 0x0000_0800

Bits	Access	Name	Description	Reset
[31:12]	-	reserved	保留。	0x00000
[11]	RW	pad_uart1_txd_ctrl_ie	UART1_TXD.IE 管脚控制： 0: 输入禁止； 1: 输入使能。	0x1
[10:4]	-	reserved	保留。	0x00
[3]	RW	pad_uart1_txd	UART1_TXD.ST 管脚控制：	0x0

Bits	Access	Name	Description	Reset
		_ctrl_st	0: No Schmitt; 1: Schmitt Enable。	
[2:0]	-	reserved	保留。	0x0

PAD_UART1_RXD_CTRL

PAD_UART1_RXD_CTRL 为 UART1_RXD 控制寄存器。

Offset Address: 0x840 Total Reset Value: 0x0000_0800

Bits	Access	Name	Description	Reset
[31:12]	-	reserved	保留。	0x00000
[11]	RW	pad_uart1_rxd_ctrl_ie	UART1_RXD.IE 管脚控制: default: 1 0: 禁止; 1: 使能。	0x1
[10:4]	-	reserved	保留。	0x00
[3]	RW	pad_uart1_rxd_ctrl_st	UART1_RXD.ST 管脚控制: 0: No Schmitt; 1: Schmitt Enable。	0x0
[2:0]	-	reserved	保留。	0x0

PAD_UART0_TXD_CTRL

PAD_UART0_TXD_CTRL 为 UART0_TXD 控制寄存器。

Offset Address: 0x844 Total Reset Value: 0x0000_0800

Bits	Access	Name	Description	Reset
[31:12]	-	reserved	保留。	0x00000

Bits	Access	Name	Description	Reset
]				
[11]	RW	pad_uart0_txd_ctrl_ie	UART0_TXD.IE 管脚控制： 0：输入禁止； 1：输入使能。	0x1
[10:4]	-	reserved	保留。	0x00
[3]	RW	pad_uart0_txd_ctrl_st	UART0_TXD.ST 管脚控制： 0：No Schmitt； 1：Schmitt Enable。	0x0
[2:0]	-	reserved	保留。	0x0

PAD_UART0_RXD_CTRL

PAD_UART0_RXD_CTRL 为 UART0_RXD 控制寄存器。

Offset Address: 0x848 Total Reset Value: 0x0000_0800

Bits	Access	Name	Description	Reset
[31:12]	-	reserved	保留。	0x00000
[11]	RW	pad_uart0_rxd_ctrl_ie	UART0_RXD.IE 管脚控制： 0：输入禁止； 1：输入使能。	0x1
[10:4]	-	reserved	保留。	0x00
[3]	RW	pad_uart0_rxd_ctrl_st	UART0_RXD.ST 管脚控制： 0：No Schmitt； 1：Schmitt Enable。	0x0
[2:0]	-	reserved	保留。	0x0

PAD_SFC_CLK_CTRL

PAD_SFC_CLK_CTRL 为 SFC_CLK 控制寄存器。

Offset Address: 0x868 Total Reset Value: 0x0000_0810

Bits	Access	Name	Description	Reset
[31:12]	-	reserved	保留。	0x00000
[11]	RW	pad_sfc_clk_ctrl_ie	SFC_CLK.IE 管脚控制： 0：输入禁止； 1：输入使能。	0x1
[10]	RW	pad_sfc_clk_ctrl_ps	SFC_CLK.PS 管脚控制： 需与 SFC_CLK.PE 管脚控制结合使用。 PE/PS 对应上下拉能力控制如下： 00：无上下拉； 01：强上拉； 10：下拉； 11：上拉。	0x0
[9]	RW	pad_sfc_clk_ctrl_pe	SFC_CLK.PE 管脚控制。	0x0
[8:7]	-	reserved	保留。	0x0
[6]	RW	pad_sfc_clk_ctrl_ds2	SFC_CLK.DS2 管脚控制： 需与 SFC_CLK.DS1/SFC_CLK.DS0 管脚控制结合使用。 DS2~DS0 对应驱动能力调节： 111~000，驱动能力依次减弱。	0x0
[5]	RW	pad_sfc_clk_ctrl_ds1	SFC_CLK.DS1 管脚控制。	0x0
[4]	RW	pad_sfc_clk_ctrl_ds0	SFC_CLK.DS0 管脚控制。	0x1
[3]	RW	pad_sfc_clk_ctrl	SFC_CLK.ST 管脚控制：	0x0

Bits	Access	Name	Description	Reset
		rl_st	0: No Schmitt; 1: Schmitt Enable。	
[2:0]	-	reserved	保留。	0x0

PAD_SFC_CSN_CTRL

PAD_SFC_CSN_CTRL 为 SFC_CSN 控制寄存器。

Offset Address: 0x86c Total Reset Value: 0x0000_0E10

Bits	Access	Name	Description	Reset
[31:12]	-	reserved	保留。	0x00000
[11]	RW	pad_sfc_csn_ctrl_ie	SFC_CSN.IE 管脚控制： 0: 输入禁止； 1: 输入使能。	0x1
[10]	RW	pad_sfc_csn_ctrl_ps	SFC_CSN.PS 管脚控制： 需与 SFC_CSN.PE 管脚控制结合使用。 PE/PS 对应上下拉能力控制如下： 00: 无上下拉； 01: 强上拉； 10: 下拉； 11: 上拉。	0x1
[9]	RW	pad_sfc_csn_ctrl_pe	SFC_CSN.PE 管脚控制。	0x1
[8:7]	-	reserved	保留。	0x0
[6]	RW	pad_sfc_csn_ctrl_ds2	SFC_CSN.DS2 管脚控制： 需与 SFC_CSN.DS1/SFC_CSN.DS0 管	0x0

Bits	Access	Name	Description	Reset
			脚控制结合使用。 DS2~DS0 对应驱动能力调节： 111~000，驱动能力依次减弱。	
[5]	RW	pad_sfc_csn_ctrl_ds1	SFC_CSN.DS1 管脚控制。	0x0
[4]	RW	pad_sfc_csn_ctrl_ds0	SFC_CSN.DS0 管脚控制。	0x1
[3]	RW	pad_sfc_csn_ctrl_st	SFC_CSN.ST 管脚控制： 0：No Schmitt； 1：Schmitt Enable。	0x0
[2:0]	-	reserved	保留。	0x0

PAD_SFC_IO0_CTRL

PAD_SFC_IO0_CTRL 为 SFC_IO0 控制寄存器。

Offset Address: 0x870 Total Reset Value: 0x0000_0E10

Bits	Access	Name	Description	Reset
[31:12]	-	reserved	保留。	0x00000
[11]	RW	pad_sfc_io0_ctrl_ie	SFC_IO0.IE 管脚控制： 0：输入禁止； 1：输入使能。	0x1
[10]	RW	pad_sfc_io0_ctrl_ps	SFC_IO0.PS 管脚控制： 需与 SFC_IO0.PE 管脚控制结合使用。 PE/PS 对应上下拉能力控制如下： 00：无上下拉； 01：强上拉； 10：下拉；	0x1

Bits	Access	Name	Description	Reset
			11: 上拉。	
[9]	RW	pad_sfc_io0_ctrl_pe	SFC_IO0.PE 管脚控制。	0x1
[8:7]	-	reserved	保留。	0x0
[6]	RW	pad_sfc_io0_ctrl_ds2	SFC_IO0.DS2 管脚控制： 需与 SFC_IO0.DS1/SFC_IO0.DS0 管脚控制结合使用。 DS2~DS0 对应驱动能力调节： 111~000，驱动能力依次减弱。	0x0
[5]	RW	pad_sfc_io0_ctrl_ds1	SFC_IO0.DS1 管脚控制。	0x0
[4]	RW	pad_sfc_io0_ctrl_ds0	SFC_IO0.DS0 管脚控制。	0x1
[3]	RW	pad_sfc_io0_ctrl_st	SFC_IO0.ST 管脚控制： 0: No Schmitt; 1: Schmitt Enable。	0x0
[2:0]	-	reserved	保留。	0x0

PAD_SFC_IO1_CTRL

PAD_SFC_IO1_CTRL 为 SFC_IO1 控制寄存器。

Offset Address: 0x874 Total Reset Value: 0x0000_0E10

Bits	Access	Name	Description	Reset
[31:12]	-	reserved	保留。	0x00000
[11]	RW	pad_sfc_io1_ctrl_ie	SFC_IO1.IE 管脚控制： 0: 输入禁止； 1: 输入使能。	0x1

Bits	Access	Name	Description	Reset
[10]	RW	pad_sfc_io1_ctrl_ps	SFC_IO1.PS 管脚控制： 需与 SFC_IO1.PE 管脚控制结合使用。 PE/PS 对应上下拉能力控制如下： 00：无上下拉； 01：强上拉； 10：下拉； 11：上拉。	0x1
[9]	RW	pad_sfc_io1_ctrl_pe	SFC_IO1.PE 管脚控制。	0x1
[8:7]	-	reserved	保留。	0x0
[6]	RW	pad_sfc_io1_ctrl_ds2	SFC_IO1.DS2 管脚控制： 需与 SFC_IO1.DS1/SFC_IO1.DS0 管脚控制结合使用。 DS2~DS0 对应驱动能力调节： 111~000，驱动能力依次减弱。	0x0
[5]	RW	pad_sfc_io1_ctrl_ds1	SFC_IO1.DS1 管脚控制。	0x0
[4]	RW	pad_sfc_io1_ctrl_ds0	SFC_IO1.DS0 管脚控制。	0x1
[3]	RW	pad_sfc_io1_ctrl_st	SFC_IO1.ST 管脚控制： 0：No Schmitt； 1：Schmitt Enable。	0x0
[2:0]	-	reserved	保留。	0x0

PAD_SFC_IO2_CTRL

PAD_SFC_IO2_CTRL 为 SFC_IO2 控制寄存器。

Offset Address: 0x878 Total Reset Value: 0x0000_0E10

Bits	Access	Name	Description	Reset
[31:12]	-	reserved	保留。	0x00000
[11]	RW	pad_sfc_io2_ctl_ie	SFC_IO2.IE 管脚控制： 0：输入禁止； 1：输入使能。	0x1
[10]	RW	pad_sfc_io2_ctl_ps	SFC_IO2.PS 管脚控制： 需与 SFC_IO2.PE 管脚控制结合使用。 PE/PS 对应上下拉能力控制如下： 00：无上下拉； 01：强上拉； 10：下拉； 11：上拉。	0x1
[9]	RW	pad_sfc_io2_ctl_pe	SFC_IO2.PE 管脚控制。	0x1
[8:7]	-	reserved	保留。	0x0
[6]	RW	pad_sfc_io2_ctl_ds2	SFC_IO2.DS2 管脚控制： 需与 SFC_IO2.DS1/SFC_IO2.DS0 管脚控制结合使用。 DS2~DS0 对应驱动能力调节： 111 至 000，驱动能力依次减弱。	0x0
[5]	RW	pad_sfc_io2_ctl_ds1	SFC_IO2.DS1 管脚控制。	0x0
[4]	RW	pad_sfc_io2_ctl_ds0	SFC_IO2.DS0 管脚控制。	0x1
[3]	RW	pad_sfc_io2_ctl_st	SFC_IO2.ST 管脚控制： 0：No Schmitt； 1：Schmitt Enable。	0x0
[2:0]	-	reserved	保留。	0x0

PAD_SFC_IO3_CTRL

PAD_SFC_IO3_CTRL 为 SFC_IO3 控制寄存器。

Offset Address: 0x87c Total Reset Value: 0x0000_0E10

Bits	Access	Name	Description	Reset
[31:12]	-	reserved	保留。	0x00000
[11]	RW	pad_sfc_io3_ctrl_ie	SFC_IO3.IE 管脚控制： 0：输入禁止； 1：输入使能。	0x1
[10]	RW	pad_sfc_io3_ctrl_ps	SFC_IO3.PS 管脚控制： 需与 SFC_IO3.PE 管脚控制结合使用。 PE/PS 对应上下拉能力控制如下： 00：无上下拉； 01：强上拉； 10：下拉； 11：上拉。	0x1
[9]	RW	pad_sfc_io3_ctrl_pe	SFC_IO3.PE 管脚控制。	0x1
[8:7]	-	reserved	保留。	0x0
[6]	RW	pad_sfc_io3_ctrl_ds2	SFC_IO3.DS2 管脚控制： 需与 SFC_IO3.DS1/SFC_IO3.DS0 管脚控制结合使用。 DS2~DS0 对应驱动能力调节： 111~000，驱动能力依次减弱。	0x0
[5]	RW	pad_sfc_io3_ctrl_ds1	SFC_IO3.DS1 管脚控制。	0x0
[4]	RW	pad_sfc_io3_ctrl_ds0	SFC_IO3.DS0 管脚控制。	0x1

Bits	Access	Name	Description	Reset
[3]	RW	pad_sfc_io3_ctl_st	SFC_IO3.ST 管脚控制: 0: No Schmitt; 1: Schmitt Enable。	0x0
[2:0]	-	reserved	保留。	0x0

6.2 GPIO

6.2.1 概述

GPIO (General Purpose Programable Input/Output) 为通用可编程输入输出外设，用于生成和采集特定应用的输入或输出信号，实现系统和外设之间的通信，方便系统对外设的控制。

6.2.2 功能描述

GPIO 接口具有以下功能特点：

- 共 3 组 GPIO (GPIO0/1/2)，其中 GPIO0/1 各有 8 个独立的可配置管脚，GPIO2 有 3 个独立的可配置管脚，总共 19 个独立的可配置管脚。
- 第一组 GPIO0 对应管脚为 7:0，第二组 GPIO1 对应管脚为 15:8，第三组 GPIO2 对应管脚为 18:16；
- 每个 GPIO 管脚都可单独控制传输方向。
- 每个 GPIO 可以被配置为外部中断源。
- GPIO 用作中断时有 5 种中断触发方式，中断时触发方式可配：
 - 上升沿触发
 - 下降沿触发
 - 高电平触发
 - 低电平触发
 - 双沿触发
- 每组 GPIO 上报一个中断，共三个中断号，CPU 查询上报的 GPIO 编号。
- 每个中断支持独立屏蔽功能，边沿中断支持可清除功能。

6.2.3 工作方式

6.2.3.1 初始化配置

每个 GPIO 可以单独配置为输入或者输出，具体步骤如下：

步骤 1 参考 IO_MUX 进行 GPIO 的 IO 复用初始化配置。

步骤 2 配置 `GPIO_SW_OEN[n]` 数据方向寄存器，数据方向按 bit 单独控制，写入值为 1 表示该 bit 对应的数据方向是输入，写入值为 0 表示该 bit 对应的数据方向是输出；

步骤 3 写 `GPIO_SW_OUT[n]` 数据寄存器，如果对应的 bit 数据方向是输出，写入该 bit 的值是对应 I/O 信号的输出，回读值等于最后一次写入的值。如果对应的 bit 数据方向是输入，写入值不起作用；

步骤 4 读 `GPIO_SW_OUT[n]` 数据寄存器，如果对应的 bit 数据方向是输入，读该寄存器为外部端口的输入值，软件写入值无效。

----结束

须知

当 GPIO 用作输出时，建议禁止 GPIO 控制器的中断功能，否则当输出信号符合中断产生条件时，会产生 GPIO 中断。

6.2.3.2 边沿中断配置

假设低 8bit GPIO 输入中断，中断类型均为上升沿有效，并且输入中断需要去毛刺。

步骤 1 写中断使能寄存器 `GPIO_INT_EN[7:0]=8'h0`，配置 8 个 GPIO 中断禁止。

步骤 2 写双沿中断使能寄存器 `GPIO_INT_DEEDGE[7:0]=8'h0`，配置不使能双沿中断。

步骤 3 写中断类型寄存器 `GPIO_INT_TYPE[7:0]=8'hFF`，配置 8 个输入中断为边沿有效。

步骤 4 写中断极性寄存器 `GPIO_INT_POLARITY[7:0]=8'hFF`，配置为 8 个输入中断上升沿有效。

步骤 5 写中断屏蔽寄存器 `GPIO_INT_MASK[7:0]=8'h0`，配置不屏蔽中断。

步骤 6 写中断去毛刺控制寄存器 `GPIO_INT_DEBOUNCE[7:0]=8'hFF`，配置为去毛刺使能。

步骤 7 写中断使能寄存器 `GPIO_INT_EN[7:0]=8'hFF`，使能 8 个输入端口的中断功能。

步骤 8 当有上升沿有效（经过去毛刺）的中断发生后，CPU 收到 GPIO 中断，可以通过查询中断状态寄存器 GPIO_INTR 来判断哪个端口发生了中断。

步骤 9 写中断清除寄存器 GPIO_INT_EOI 对应 bit 为 1，可以清除掉相应的中断状态。

----结束

说明

中断相关寄存器的配置均独立生效，建议先取消中断使能，再配置相应的中断相关寄存器，最后配置中断使能。

当配置的是边沿中断时，可以通过写中断清除寄存器 GPIO_INT_EOI 来清中断，如果是电平中断写此寄存器无效。

当配置双沿中断使能寄存器 GPIO_INT_DEDGE，双沿中断有效时，配置中断类型寄存器和中断极性寄存器均无效。

对于边沿中断（包括双沿或单沿），同步功能是默认有效的，不能配置。

双沿中断使能后，建议软件配置中断去毛刺控制寄存器 GPIO_INT_DEBOUNCE 为不使能。

6.2.3.3 电平中断配置

假设低 2 个 GPIO 输入中断，中断类型均为高电平有效，并且输入中断需要去毛刺。

步骤 1 写中断使能寄存器 GPIO_INT_EN[1:0]=2'h0，配置低 2 个 GPIO 中断不使能。

步骤 2 写双沿中断使能寄存器 GPIO_INT_DEDGE[1:0]=2'h0，配置不使能双沿中断。

步骤 3 写中断类型寄存器 GPIO_INT_TYPE[1:0]=2'h0，配置低 2 个输入中断为电平有效。

步骤 4 写中断极性寄存器 GPIO_INT_POLARITY[1:0]=2'h3，配置为低 2 个输入中断高电平有效。

步骤 5 写中断屏蔽寄存器 GPIO_INT_MASK[1:0]=2'h0，配置不屏蔽中断。

步骤 6 写中断去毛刺控制寄存器 GPIO_INT_DEBOUNCE[1:0]=2'h3，配置低 2 个输入中断为去毛刺使能。

步骤 7 写中断使能寄存器 GPIO_INT_EN[1:0]=2'h3，使能低 2 个输入端口的中断功能。

步骤 8 当有高电平有效的中断发生后，经过去毛刺处理，CPU 收到 GPIO 中断，可以通过查询中断状态寄存器 GPIO_INTR 来判断哪个端口发生了中断。

----结束

说明

当设置电平中断时，中断同步功能默认有效，不可配置。

电平中断无法通过写 GPIO_INT_EOI 中断清除寄存器来清除。
由于需要同步，因此电平中断必须在 GPIO 有时钟时才能输出。

6.2.4 寄存器概览

GPIO 寄存器概览如表 6-5 所示。

表6-5 GPIO 寄存器概览（基址是 GPIO0:0x44028000、GPIO1:0x44029000、GPIO2:0x4402A000）

偏移地址	名称	描述
0x0000	GPIO_SW_OUT	GPIO 数据寄存器。
0x0004	GPIO_SW_OEN	GPIO 数据方向寄存器。
0x000C	GPIO_INT_EN	GPIO 中断使能寄存器。
0x0010	GPIO_INT_MASK	GPIO 中断屏蔽寄存器。
0x0014	GPIO_INT_TYPE	GPIO 中断类型寄存器。
0x0018	GPIO_INT_POLARITY	GPIO 中断极性寄存器。
0x001C	GPIO_INT_DEEDGE	GPIO 双沿中断使能寄存器。
0x0020	GPIO_INT_DEBOUNCE	GPIO 中断去毛刺控制寄存器。
0x0024	GPIO_INT_RAW	GPIO 原始中断状态寄存器。
0x0028	GPIO_INTR	GPIO 中断状态寄存器。
0x002C	GPIO_INT_EOI	GPIO 中断清除寄存器。
0x0030	GPIO_DATA_SET	GPIO 数据设置寄存器。
0x0034	GPIO_DATA_CLR	GPIO 数据清除寄存器。

6.2.5 寄存器描述

GPIO_SW_OUT

GPIO_SW_OUT 为 GPIO 数据寄存器。

 说明

GPIO2 的 BIT[7:3]未使用。

Offset Address: 0x0000 Total Reset Value: 0x0000_0000

Bits	Access	Name	Description	Reset
[31:8]	RO	reserved	保留。	0x000000
[7:0]	RW	gpio_sw_out	GPIO 信号的数据配置。 软件模式输出：写入该寄存器的值是该端口 I/O 信号的输出，回读值等于最后一次写入寄存器的值。 软件模式输入：读该寄存器为外部端口的输入值，写该寄存器无效。 硬件模式：写无意义，读回为写入值。 注意：由于端口可以独立控制，因此，上述描述是针对该寄存器的每一位而言。	0x00

GPIO_SW_OEN

GPIO_SW_OEN 为 GPIO 数据方向寄存器。

 说明

GPIO2 的 BIT[7:3]未使用。

Offset Address: 0x0004 Total Reset Value: 0x0000_00FF

Bits	Access	Name	Description	Reset
[31:8]	RO	reserved	保留。	0x000000
[7:0]	RW	gpio_sw_oen	GPIO 信号的数据方向控制。 写入寄存器的值独立控制相应端口的方向： 0: output;	0xFF

Bits	Access	Name	Description	Reset
			1: input (默认值)。 读回表示各端口的输入输出属性： 0: output; 1: input。	

GPIO_INT_EN

GPIO_INT_EN 为 GPIO 中断使能寄存器。

说明

GPIO2 的 BIT[7:3]未使用。

Offset Address: 0x000C Total Reset Value: 0x0000_0000

Bits	Access	Name	Description	Reset
[31:8]	RO	reserved	保留。	0x000000
[7:0]	RW	gpio_int_en	GPIO 信号的中断使能配置。 写入寄存器的值可以独立控制相应端口的中断使能： 0: 普通 GPIO 端口 (默认值); 1: 中断端口。 读回表示各端口的中断使能状态： 0: 普通 GPIO 端口; 1: 中断端口。 注意：中断功能必须在对应端口为输入属性下才会有效。	0x00

GPIO_INT_MASK

GPIO_INT_MASK 为 GPIO 中断屏蔽寄存器。

说明

GPIO2 的 BIT[7:3]未使用。

Offset Address: 0x0010 Total Reset Value: 0x0000_0000

Bits	Access	Name	Description	Reset
[31:8]	RO	reserved	保留。	0x000000
[7:0]	RW	gpio_int_mask	GPIO 信号的中断屏蔽控制。 写入寄存器的值可以独立控制相应端口的中断屏蔽： 0：不屏蔽（默认值）； 1：屏蔽。 读回表示各端口的中断屏蔽状态： 0：不屏蔽； 1：屏蔽。	0x00

GPIO_INT_TYPE

GPIO_INT_TYPE 为 GPIO 中断类型寄存器。

说明

GPIO2 的 BIT[7:3]未使用。

Offset Address: 0x0014 Total Reset Value: 0x0000_0000

Bits	Access	Name	Description	Reset
[31:8]	RO	reserved	保留。	0x000000
[7:0]	RW	gpio_int_type	GPIO 信号的中断类型配置。 写入寄存器的值可以独立控制相应端口的中断类型： 0：电平中断（默认值）； 1：边沿中断。 读回表示各端口的中断类型属性：	0x00

Bits	Access	Name	Description	Reset
			0：电平中断， 1：边沿中断。	

GPIO_INT_POLARITY

GPIO_INT_POLARITY 为 GPIO 中断极性寄存器。

说明

GPIO2 的 BIT[7:3]未使用。

Offset Address：0x0018 Total Reset Value：0x0000_0000

Bits	Access	Name	Description	Reset
[31:8]	RO	reserved	保留。	0x000000
[7:0]	RW	gpio_int_polarity	GPIO 信号的中断极性配置。 写入寄存器的值可以独立控制相应端口的中断极性： 0：低电平或者下降沿（默认值）； 1：高电平或者上升沿。 读回表示各端口的中断极性属性： 0：低电平或者下降沿； 1：高电平或者上升沿。	0x00

GPIO_INT_DEEDGE

GPIO_INT_DEEDGE 为 GPIO 双沿中断使能寄存器。

说明

GPIO2 的 BIT[7:3]未使用。

Offset Address：0x001C Total Reset Value：0x0000_0000

Bits	Access	Name	Description	Reset
------	--------	------	-------------	-------

Bits	Access	Name	Description	Reset
[31:8]	RO	reserved	保留。	0x000000
[7:0]	RW	gpio_int_dedge	GPIO 信号的双沿中断使能配置。 写入寄存器的值可以独立控制相应端口的双沿中断使能： 0：禁止双沿中断； 1：使能双沿中断。 读回表示各端口的双沿中断使能状态： 0：未使能双沿中断； 1：已使能双沿中断。	0x00

GPIO_INT_DEBOUNCE

GPIO_INT_DEBOUNCE 为 GPIO 中断去毛刺控制寄存器。

说明

GPIO2 的 BIT[7:3]未使用。

Offset Address: 0x0020 Total Reset Value: 0x0000_0000

Bits	Access	Name	Description	Reset
[31:8]	RO	reserved	保留。	0x000000
[7:0]	RW	gpio_int_debounce	GPIO 信号的中断输入去毛刺控制配置。 写入寄存器的值可以独立控制相应端口是否使能中断输入去毛刺功能： 0：禁止去毛刺（默认值）； 1：使能去毛刺。 读回表示各端口的 Debounce 属性：	0x00

Bits	Access	Name	Description	Reset
			0: 禁止去毛刺; 1: 使能去毛刺。	

GPIO_INT_RAW

GPIO_INT_RAW 为 GPIO 原始中断状态寄存器。

说明

GPIO2 的 BIT[7:3]未使用。

Offset Address: 0x0024 Total Reset Value: 0x0000_0000

Bits	Access	Name	Description	Reset
[31:8]	RO	reserved	保留。	0x000000
[7:0]	RO	gpio_int_raw	GPIO 信号的原始中断状态查询。 屏蔽之前的中断状态: 0: 未产生中断; 1: 产生中断。	0x00

GPIO_INTR

GPIO_INTR 为 GPIO 中断状态寄存器。

说明

GPIO2 的 BIT[7:3]未使用。

Offset Address: 0x0028 Total Reset Value: 0x0000_0000

Bits	Access	Name	Description	Reset
[31:8]	RO	reserved	保留。	0x000000
[7:0]	RO	gpio_intr	GPIO 信号的中断状态查询。屏蔽 之后的中断状态: 0: 未产生中断;	0x00

Bits	Access	Name	Description	Reset
			1：产生中断。	

GPIO_INT_EOI

GPIO_INT_EOI 为 GPIO 中断清除寄存器。

 说明

GPIO2 的 BIT[7:3]未使用。

Offset Address: 0x002C Total Reset Value: 0x0000_0000

Bits	Access	Name	Description	Reset
[31:8]	RO	reserved	保留。	0x000000
[7:0]	RO	gpio_int_eoi	GPIO 信号的中断清除控制。 写入寄存器的值可以独立控制相应边沿中断端口的中断清除： 0：不清除中断（默认值）； 1：清除中断。	0x00

GPIO_DATA_SET

GPIO_DATA_SET 为 GPIO 数据设置寄存器。

 说明

GPIO2 的 BIT[7:3]未使用。

Offset Address: 0x0030 Total Reset Value: 0x0000_0000

Bits	Access	Name	Description	Reset
[31:8]	RO	reserved	保留。	0x000000
[7:0]	WO	gpio_data_set	写 GPIO 数据请求，高有效。 软件模式输出时，写入寄存器的值可以将	0x00

Bits	Access	Name	Description	Reset
			GPIO_SW_OUT[gpio_sw_out]相应 bit 位置 1。	

GPIO_DATA_CLR

GPIO_DATA_CLR 为 GPIO 数据清除寄存器。

说明

GPIO2 的 BIT[7:3]未使用。

Offset Address: 0x0034 Total Reset Value: 0x0000_0000

Bits	Access	Name	Description	Reset
[31:8]	RO	reserved	保留。	0x000000
[7:0]	WO	gpio_data_clr	清 GPIO 数据请求，高有效。 软件模式输出时，写入寄存器的值 可以将 GPIO_SW_OUT[gpio_sw_out]相应 bit 位清零。	0x00

6.3 UART

6.3.1 概述

通用异步收发器 UART (universal asynchronous receiver/transmitter) 是一个异步串行的通信接口，UART 主要用于和外部芯片的 UART 进行对接，实现两芯片间的通信。

芯片提供 3 个 UART 单元 (UART0/1/2)，UART0 仅支持两线模式，UART1/2 支持流控功能。

6.3.2 功能描述

UART 具有以下功能特点：

- 支持 64x8bit 的发送 FIFO 和 64x10bit 的接收 FIFO (First In First Out)。
- 支持可编程数据位宽 5/6/7/8 bit。
- 支持可编程停止位宽 1/1.5/2 bit。
- 支持奇/偶校验或者无校验位，支持校验位为软件设定值。
- 支持波特率可编程（整数分频和小数分频，小数分频的参数 DLF_SIZE 为 6）。
- 支持接收 FIFO 中断、发送 FIFO 中断、接收超时中断、错误中断。
- 支持中断状态查询。
- UART0 不支持硬件流控，UART1 和 UART2 支持硬件流控。
- 支持三种数据搬运方式：
 - 支持 DMA 方式。
 - 支持中断方式。
 - 支持软件查询方式。

6.3.3 工作方式

接口信号

UART 接口信号描述如表 UART 接口信号描述所示。

表6-6 UART 接口信号描述

信号名	宽度 (bit)	方向	功能描述
RXD	1	I	输入数据。
TXD	1	O	输出数据。
CTS	1	I	清除发送信号，用于硬件流控，低有效。
RTS	1	O	请求发送信号，用于硬件流控，低有效。

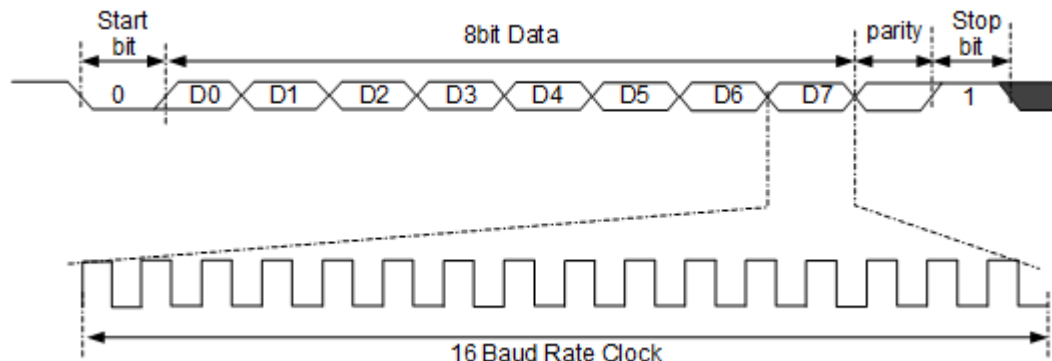
UART 数据格式

须知

整数波特率寄存器和小数波特率寄存器的值必须等到当前数据发送和接收完毕才更新。

UART 的数据帧格式如图 6-1 所示。其中数据帧长度、停止位位数和奇偶检验可配置。

图6-1 UART 数据帧格式



中断或查询方式下的数据传输

初始化步骤如下：

- 步骤 1 进行 UART 时钟初始化配置，将 UART 工作时钟配置成 160M。
- 步骤 2 配置 IO 复用关系，IO 暂不复用为 UART，防止初始化过程中 UART 被对端影响。
- 步骤 3 配置 `HALT_TX[halt_tx]=1`，停止 TX 功能，防止初始化过程中触发 TX。
- 步骤 4 配置 `INTR_EN=0`，屏蔽 UART 所有中断。
- 步骤 5 向 `FIFO_CTL[rx_fifo_rst]`和 `FIFO_CTL[tx_fifo_rst]`写 1，分别复位 RX FIFO 和 TX FIFO。
- 步骤 6 配置 `BAUD_CTL[baud_div]`，根据需要配置波特率过采样倍数。
- 步骤 7 配置 `UART_CTL[div_en]=1`，准备开始配置波特率。
- 步骤 8 配置 `DIV_H`、`DIV_L`、`DIV_FRA`，确定波特率。配置值计算方法为：分频值=工作时钟频率/（波特率×波特率过采样倍数），{`DIV_H`, `DIV_L`}组成分频值的整数部分，`DIV_FRA` = 分频率的小数部分×64。
- 步骤 9 配置 `UART_CTL[stp]`、`UART_CTL[pen]`、`UART_CTL[eps]`、`UART_CTL[dlen]`，根据需要配置帧格式，以上寄存器分别控制停止位位宽，奇偶校验使能，奇偶校验选择和数据位长度。
- 步骤 10 配置 `MODEM_CTL[afc_en]`和 `MODEM_CTL[rts]`，根据需要配置自动流控。

表6-7 自动流控配置参考

场景	MODEM_CTL[afc_en]配置值	MODEM_CTL[rts]配置值
打开自动流控	1	1
关闭自动流控，反压对端	0	0
关闭自动流控，正常 RX	0	1

- 步骤 11 配置 `FIFO_CTL[rx_empty_trig]`和 `FIFO_CTL[tx_empty_trig]`，设定发送及接收 FIFO 水线。
- 步骤 12 配置 `FIFO_CTL[fifo_en]=1`，使能 TX FIFO 和 RX FIFO。
- 步骤 13 配置 `INTR_EN=0x1f`，恢复中断使能。
- 步骤 14 恢复 IO 复用，将 IO 复用为 UART，以及配置对应 IO 的驱动能力。
- 步骤 15 配置 `HALT_TX[halt_tx]=0`，使能 TX 功能。

----结束

数据发送步骤如下：

- 步骤 1 将发送数据写入 `DATA[data]`步骤，启动数据发送。若为查询模式则跳转至**步骤 2**，若为中断模式则跳转至**步骤 3**。
- 步骤 2 查询方式下，如果进行连续数据发送，需要通过读取 `FIFO_STATUS[tx_fifo_full]`检测 TX FIFO 状态。如果 `FIFO_STATUS[tx_fifo_full]`为 0，即 TX FIFO 未滿，则可以向 TX FIFO 中发送数据。直到无数据需要发送，跳转至**步骤 4**。
- 步骤 3 中断方式下，在中断服务程序中查询 `INTR_STATUS[thre_intr_status]`发送中断状态位，决定是否向 TX FIFO 中发送数据。当 `INTR_STATUS[thre_intr_status]`置 1，此时 TX FIFO 内数据量小于发送数据水线，可以向 TX FIFO 中发送数据。直到无数据需要发送，跳转至**步骤 4**。
- 步骤 4 通过检测 `INTR_STATUS[tx_fifo_empty]`是否为 1，判断 UART 是否完成全部数据发送。

----结束

数据接收的处理方式如下：

- 步骤 1 等待数据接收，若为查询模式则跳转至步骤 2。若为中断模式则跳转至步骤 3。
- 步骤 2 查询方式下，进行数据接收时通过读取 INTR_STATUS[rx_fifo_empty]检测 RX_FIFO 状态，如果 INTR_STATUS[rx_fifo_empty]为 0，则 RX_FIFO 非空，可以读取 RX_FIFO 中的数据，跳转至步骤 4。
- 步骤 3 中断方式下，则检测 INTR_STATUS[data_avail_intr_status]接收中断状态位，决定是否读取 RX_FIFO 中的数据。当 INTR_STATUS[data_avail_intr_status]置 1，此时 RX_FIFO 内数据量大于接收 FIFO 水线，可以读取 RX_FIFO 中数据，跳转至步骤 4。
- 步骤 4 回读 DATA[data]，读出数据即为 RX 数据。

----结束

6.3.4 寄存器概览

UART 寄存器概览如表 6-8 所示。

表6-8 UART 寄存器概览（基址是 UART0: 0x44010000、UART1: 0x44011000、UART2: 0x44012000）

偏移地址	名称	描述
0x00	INTR_ID	中断 ID 寄存器。
0x4	DATA	数据寄存器。
0x8	UART_CTL	UART 控制寄存器。
0xC	DIV_H	分频系数（高位）寄存器。
0x10	DIV_L	分频系数（低位）寄存器。
0x14	DIV_FRA	分频系数（小数部分）寄存器。
0x18	INTR_EN	中断使能寄存器。
0x1C	INTR_STATUS	中断状态寄存器。
0x24	FIFO_CTL	FIFO 控制寄存器。
0x28	FAR	FIFO 存取模式使能寄存器。
0x2C	MODEM_CTL	Modem 控制寄存器。

偏移地址	名称	描述
0x30	MODEM_STATUS	Modem 状态寄存器。
0x34	LINE_STATUS	Line 状态寄存器。
0x38	UART_GP_REG	Uart 通用寄存器。
0x3C	TX_FIFO_READ	发送 FIFO 读取寄存器。
0x40	RX_FIFO_WRITE	接收 FIFO 写入寄存器。
0x44	FIFO_STATUS	FIFO 状态寄存器。
0x48	TX_FIFO_CNT	发送 FIFO 数据计数器。
0x4C	RX_FIFO_CNT	接收 FIFO 数据计数器。
0x50	HALT_TX	传输挂起寄存器。
0x54	DMA_SW_ACK	DMA 应答寄存器。
0x58	BAUD_CTL	波特率控制寄存器。
0x5C	STP_CTL	停止位控制寄存器。
0x60	UART_PARAMETER	UART 参数寄存器。

6.3.5 寄存器描述

INTR_ID

INTR_ID 为中断 ID 寄存器。

Offset Address: 0x00 Total Reset Value: 0x0001

Bits	Access	Name	Description	Reset
[15:5]	-	reserved	保留。	0x000
[4]	RO	fifo_en_s	FIFO 使能控制。 0: FIFO 禁用; 1: FIFO 使能。	0x0

Bits	Access	Name	Description	Reset
[3:0]	RO	intr_id	中断 ID。 0x0：modem 状态； 0x1：无中断请求； 0x2：THR 空标志； 0x4：接收数据到达； 0x6：接收数据线状态； 0x7：busy 状态； 0xc：字符超时； others：未定义。	0x1

DATA

DATA 为数据寄存器。

Offset Address：0x4 Total Reset Value：0x0000

Bits	Access	Name	Description	Reset
[15:8]	-	reserved	保留。	0x00
[7:0]	RW	data	寄存器写入： 若使能 FIFO_CTL[fifo_en]，写入该寄存器的数据将被发送至 tx fifo；若未使能 FIFO_CTL[fifo_en]，写入该寄存器的数据将被存储于 transmitter holding 寄存器（简称 thr，tx fifo 的底端存储位置）。 寄存器读取： 该寄存器包含从串行输入端口 (sin)接收的数据。若 FIFO_CTL[fifo_en]未使能，当前数据必须在下一个接收数据到来前读	0x00

Bits	Access	Name	Description	Reset
			取，否则将被覆盖，并产生 over-run 错误；若使能 FIFO_CTL[fifo_en]，通过该寄存器可访问 rx fifo。若 rx fifo 已满，且在下一接收数据到来前没有读取该寄存器，虽然 rx fifo 内已有的数据不受影响，但后续到来的数据将丢失，并产生 over-run 错误。	

UART_CTL

UART_CTL 为 UART 控制寄存器。

Offset Address: 0x8 Total Reset Value: 0x0000

Bits	Access	Name	Description	Reset
[15:8]	-	reserved	保留。	0x00
[7]	RW	stp	停止位宽。 0: 1bit 停止位; 1: UART_CTL[dlen]为零时 1.5bit 停止位，否则 2bit 停止位。 注意：只有当 STP_CTL[stp_mode]==0 时,该寄存器才生效。	0x0
[6]	RW	sps	粘性奇偶校验位选择。 0: 禁用粘性奇偶校验; 1: 使能粘性就校验。	0x0
[5]	RW	pen	奇偶校验使能。 0: 禁用奇偶校验; 1: 使能奇偶校验。	0x0
[4]	RW	eps	奇偶校验模式选择。	0x0

Bits	Access	Name	Description	Reset
			0：选择奇校验模式； 1：选择偶校验模式。	
[3:2]	RW	dlen	数据长度。 00：每个字符 5bit 数据； 01：每个字符 6bit 数据； 10：每个字符 7bit 数据； 11：每个字符 8bit 数据。	0x0
[1]	RW	xbreak	Break 控制位。 0：串行输出已经释放可用于数据传输； 1：串行输出强制置于空白状态。	0x0
[0]	RW	div_en	UART 分频器使能寄存器。 0：DIV_L 与 DIV_H 两寄存器仅在 UART 非 Busy 状态可写入； 1：DIV_L 与 DIV_H 两寄存器可在任意时刻读写。 注意：本芯片 UART 始终处于非 Busy 状态。	0x0

DIV_H

DIV_H 为分频系数（高位）寄存器。

Offset Address：0xC Total Reset Value：0x0000

Bits	Access	Name	Description	Reset
[15:8]	-	reserved	保留。	0x00
[7:0]	RW	div_h	分频器分频系数整数部分高 8bit。 该寄存器仅在 UART_CTL[div_en] 置位且 UART 非忙状态下可以存	0x00

Bits	Access	Name	Description	Reset
			取。	

DIV_L

DIV_L 为分频系数（低位）寄存器。

Offset Address: 0x10 Total Reset Value: 0x0000

Bits	Access	Name	Description	Reset
[15:8]	-	reserved	保留。	0x00
[7:0]	RW	div_l	分频器分频系数整数部分低 8bit。 该寄存器仅在 UART_CTL[div_en] 置位且 UART 非忙状态下可以存 取。	0x00

DIV_FRA

DIV_FRA 为分频系数（小数部分）寄存器。

Offset Address: 0x14 Total Reset Value: 0x0000

Bits	Access	Name	Description	Reset
[15:6]	-	reserved	保留。	0x000
[5:0]	RW	div_fra	分频器分频系数小数部分。分频器 所用实际分频系数的小数部分为该 值除以 2 ⁶ 。	0x00

INTR_EN

INTR_EN 为中断使能寄存器。

Offset Address: 0x18 Total Reset Value: 0x0000

Bits	Access	Name	Description	Reset
[15:5]	-	reserved	保留。	0x000
[4]	RW	ptim_en	可编程 THRE 中断模式使能。 0: 禁用可编程 THRE 中断模式; 1: 使能可编程 THRE 中断模式。	0x0
[3]	RW	tran_em_intr_en	发送数据为空中断使能。 0: 禁用发送为空中断; 1: 使能发送为空中断。	0x0
[2]	RW	rece_data_intr_en	接收数据到达中断使能。 0: 禁用接收数据到达中断; 1: 使能接收数据到达中断。	0x0
[1]	RW	modem_intr_en	Modem 状态中断使能。 0 (DISABLED) : 禁用 Modem 状态中断; 1 (ENABLED) : 使能 Modem 状态中断。	0x0
[0]	RW	rece_line_stat_intr_en	接收数据线状态中断使能。 0: 禁用接收数据线状态中断; 1: 使能接收数据线状态中断。	0x0

INTR_STATUS

INTR_STATUS 为中断状态寄存器。

Offset Address: 0x1C Total Reset Value: 0x0000

Bits	Access	Name	Description	Reset
[15:6]	-	reserved	保留。	0x000
[5]	RO	line_intr_status	接收数据线中断。 1: 中断有效;	0x0

Bits	Access	Name	Description	Reset
			0: 无中断。 注意：需要使能 INTR_EN[rece_line_stat_intr_en]，否则本中断保持为 0。	
[4]	RO	data_avail_intr_status	接收数据到达中断。 1: 中断有效; 0: 无中断。 注意：需要使能 INTR_EN[rece_data_intr_en]，否则本中断保持为 0。	0x0
[3]	RO	char_to_intr_status	字符超时中断。 1: 中断有效; 0: 无中断。 注意：需要使能 FIFO_CTL[fifo_en]和 INTR_EN[rece_data_intr_en]，否则本中断保持为 0。	0x0
[2]	RO	thre_intr_status	THR 空中断。 1: 中断有效; 0: 无中断。 注意：需要使能 INTR_EN[tran_em_intr_en]，否则本中断保持为 0。	0x0
[1]	RO	modem_intr_status	Modem 状态中断。 1: 中断有效; 0: 无中断。 注意：需要使能 INTR_EN[modem_intr_en]，否则本中断保持为 0。	0x0
[0]	RO	busy_det_intr	忙状态监测中断。	0x0

Bits	Access	Name	Description	Reset
			1: 中断有效; 0: 无中断。 注意: 本芯片 UART 始终处于非 Busy 状态, 即该寄存器保持 0。	

FIFO_CTL

FIFO_CTL 为 FIFO 控制寄存器。

Offset Address: 0x24 Total Reset Value: 0x0000

Bits	Access	Name	Description	Reset
[15:7]	-	reserved	保留。	0x000
[6]	WO	rx_fifo_rst	接收 FIFO 复位请求。 0: 不复位; 1: 复位。	0x0
[5]	WO	tx_fifo_rst	发送 FIFO 复位请求。 0: 不复位; 1: 复位。	0x0
[4]	WO	fifo_en	FIFO 使能。 0: 禁用 FIFO; 1: 使能 FIFO。	0x0
[3:2]	WO	rx_empty_trig	接收空中断触发条件设置。 00: FIFO 内存在 1 个字符; 01: FIFO 1/4 满; 10: FIFO 1/2 满; 11: 距 FIFO 满状态少 2 个字符以内。	0x0
[1:0]	WO	tx_empty_trig	发送空中断触发条件设置。	0x0

Bits	Access	Name	Description	Reset
			00: FIFO 为空; 01: FIFO 内存在 2 个字符; 10: FIFO 1/4 满; 11: FIFO 1/2 满。	

FAR

FAR 为 FIFO 存取模式使能寄存器。

Offset Address: 0x28 Total Reset Value: 0x0000

Bits	Access	Name	Description	Reset
[15:1]	-	reserved	保留。	0x0000
[0]	RW	far	使能 FIFO 存取模式。出于测试目的，FIFO 存取模式允许 Master 写入接收 FIFO、读取发送 FIFO。 0: 禁用; 1: 使能。	0x0

MODEM_CTL

MODEM_CTL 为 Modem 控制寄存器。

Offset Address: 0x2C Total Reset Value: 0x0000

Bits	Access	Name	Description	Reset
[15:6]	-	reserved	保留。	0x000
[5]	RW	out2	可编程接口 OUT2。 0: out2_n 端口撤离; 1: out2_n 端口有效。 注意: 该寄存器无用。	0x0

Bits	Access	Name	Description	Reset
[4]	RW	out1	可编程接口 OUT1。 0: out1_n 端口撤离; 1: out1_n 端口有效。 注意: 该寄存器无用。	0x0
[3]	RW	dtr	数据终端 ready。 0: dtr_n 端口撤离; 1: dtr_n 端口有效。 注意: 该寄存器无用。	0x0
[2]	RW	rts	自动流控 RTS 软控信号。 0: 反压对端, rts_n=1; 1: 自动流控 RTS 软控信号。 注意: 当 MODEL_CTL[afc_en]==1 且 FIFO_CTL[fifo_en]==1 时, 自动 流控 RTS 由硬件逻辑接管, 此时 本寄存器无效。	0x0
[1]	RW	lb_mode	环回模式控制。 0: 禁用; 1: 使能。	0x0
[0]	RW	afc_en	自动流控模式使能。 0: 禁用; 1: 使能。	0x0

MODEM_STATUS

MODEM_STATUS 为 Modem 状态寄存器。

Offset Address: 0x30 Total Reset Value: 0x0000

Bits	Access	Name	Description	Reset
------	--------	------	-------------	-------

Bits	Access	Name	Description	Reset
[15:8]	-	reserved	保留。	0x00
[7]	RO	dsr	数据已准备指示寄存器。用于指示调制解调器控制线 dsr_n 的当前状态。在环回模式 (MODEM_CTL[lb_mode==1])下, 本寄存器值取自与 MODEM_CTL[dtr]。 0: dsr_n 信号线高电平; 1: dsr_n 信号线为低电平。 注意: 本芯片 UART 的 dsr_n 恒为 1。	0x0
[6]	RO	ddsr	dsr_n 跳变指示寄存器, 指示 dsr 是否有跳变。 0: dsr_n 信号线无变化; 1: dsr_n 信号线存在变化。	0x0
[5]	RO	ri	响铃指示寄存器。用于指示调制解调器控制线 ri_n 的当前状态。在环回模式 (MODEM_CTL[lb_mode==1])下, 本寄存器值取自 MODEM_CTL[out1]。 0: ri_n 信号线为高电平; 1: ri_n 信号线为低电平。 注意: 本芯片 UART 的 ri_n 恒为 1。	0x0
[4]	RO	teri	ri_n 上升沿指示寄存器。 0: 未检测到 ri_n 上升沿; 1: 检测到 ri_n 的上升沿。	0x0
[3]	RO	dcd	数据载波检测寄存器。用于指示 modem 控制线 dcd_n 的当前状	0x0

Bits	Access	Name	Description	Reset
			态。在环回模式 (MODEM_CTL[lb_mode==1])下, 本寄存器值取自 MODEL_CTL[out2]。 0: dcd_n 信号线为高电平; 1: dcd_n 信号线为低电平。 注意: 本芯片 UART 的 dcd_n 恒为 1。	
[2]	RO	ddcd	dcd_n 跳变指示寄存器, 指示 dcd 是否有跳变。 0: dcd_n 信号线无变化; 1: dcd_n 信号线存在变化。	0x0
[1]	RO	cts	CTS 信号状态。 0: cts_n 信号线为高电平; 1: cts_n 信号线为低电平。	0x0
[0]	RO	dcts	该 bit 位用于指示 Modem 的 CTS_N 数据线从上一次读取 MODEM_STATUS 寄存器后是否发生过变化。 0: 无变化; 1: 存在变化。	0x0

LINE_STATUS

LINE_STATUS 为 Line 状态寄存器。

Offset Address: 0x34 Total Reset Value: 0x00C0

Bits	Access	Name	Description	Reset
[15:8]	-	reserved	保留。	0x00

Bits	Access	Name	Description	Reset
[7]	RO	tx_empty_s	发射器空标志。 若 FIFO 使能, 该 bit 在发送器的移位寄存器与发送 FIFO 同时为空时置位; 当 FIFO 禁用时, 该 bit 在 thr 寄存器与移位寄存器同时位空时置位。 0: 发送器非空; 1: 发送器为空。	0x1
[6]	RO	thre_s	thr 寄存器为空标志。在 INTR_EN[ptim_en]使能的前提下, 如果 FIFO_CTL[fifo_en]也使能, 该 bit 位在发送 FIFO 位满时置位; 当 FIFO_CTL[fifo_en]未使能, 该 bit 位在 thr 寄存器为空时置位。 0: THRE 中断控制禁用; 1: THRE 中断控制使能。	0x1
[5]	RO	data_available	指示 RBR 或接收 FIFO 内至少存在 1 个字符。非 FIFO 模式下该 bit 位在 RBR 倍读取后清零, 在 FIFO 模式下该 bit 位在接收 FIFO 为空时清零。 0: 数据未准备好; 1: 数据已准备好。	0x0
[4]	RO	break_intr	指示串行输入数据中是否检测到 break 序列。 0: 无 break 序列; 1: 检测到 break 序列。	0x0
[3]	RO	overrun_err	Overrun 错误, 读取 LINE_STATUS 寄存器清零该 bit 位。	0x0

Bits	Access	Name	Description	Reset
			0: 无 overrun 错误; 1: 发送 overrun 错误。	
[2]	RO	parity_err	奇偶校验错误, 读取 LINE_STATUS 寄存器清零该 bit 位。 0: 无奇偶校验错误; 1: 奇偶校验报错。	0x0
[1]	RO	frame_err	帧错误, 读取 LINE_STATUS 寄存器清零该 bit 位。 0: 无帧错误; 1: 帧错误。	0x0
[0]	RO	rx_fifo_err	接收 FIFO 错误状态指示。该 bit 位仅在 FIFO 使能时生效。当存在错误的字符位于接收 FIFO 顶部且后续数据无错误, 那么在读取 LSR 后该 bit 位清零。 0: 接收 FIFO 无错误; 1: 接收 FIFO 报错。	0x0

UART_GP_REG

UART_GP_REG 为 Uart 通用寄存器。

Offset Address: 0x38 Total Reset Value: 0x0000

Bits	Access	Name	Description	Reset
[15:8]	-	reserved	保留。	0x00
[7:0]	RW	uart_gp_reg	为开发者提供临时存储空间, 在 UART ip 内无明确定义。	0x00

TX_FIFO_READ

TX_FIFO_READ 为发送 FIFO 读取寄存器。

Offset Address: 0x3C Total Reset Value: 0x0000

Bits	Access	Name	Description	Reset
[15:8]	-	reserved	保留。	0x00
[7:0]	RO	tx_fifo_read	tx fifo 读数据寄存器。当 FIFO_CTL[fifo_en]被使能，读取该寄存器将返回 tx fifo 顶部数据，每个连续的读取会对 tx fifo 进行 POP 操作，并返回当前在 tx fifo 顶部的数据；当 FIFO_CTL[fifo_en]未被使能，读取该寄存器将返回 DR 中的数据。	0x00

RX_FIFO_WRITE

RX_FIFO_WRITE 为接收 FIFO 写入寄存器。

Offset Address: 0x40 Total Reset Value: 0x0000

Bits	Access	Name	Description	Reset
[15:10]	-	reserved	保留。	0x00
[9]	WO	rx_fifo_fe	接收 FIFO 帧错误。 0: 无错误; 1: 帧错误。	0x0
[8]	WO	rx_fifo_pe	接收 FIFO 奇偶校验错误。 0: 校验位错误; 1: 无错误。	0x0
[7:0]	RW	rx_fifo_write	写操作: 向 rx fifo 写数据。该寄存器仅在 FIFO 存取模式使能 (FAR[far]==1)时生效。当	0x00

Bits	Access	Name	Description	Reset
			FIFO_CTL[fifo_en]使能时，写入该寄存器的数据将被压入 rx fifo。每个连续的写操作会连续压入新数据至 rx fifo 的下一写入位置；当 FIFO_CTL[fifo_en]未被使能，写入该寄存器的数据将被压入 DR。 读操作：返回 rx_fifo_level[6:0]，指示 rx fifo 内的数据量。	

FIFO_STATUS

FIFO_STATUS 为 FIFO 状态寄存器。

Offset Address: 0x44 Total Reset Value: 0x0002

Bits	Access	Name	Description	Reset
[15:4]	-	reserved	保留。	0x000
[3]	RO	rx_fifo_empty	rx fifo 空标志。 0: 接收 FIFO 非空; 1: 接收 FIFO 为空。	0x1
[2]	RO	rx_fifo_full	rx fifo 满标志。 0: 接收 FIFO 非满; 1: 接收 FIFO 为满。	0x0
[1]	RO	tx_fifo_empty	tx fifo 空标志。 0: 发送 FIFO 非空; 1: 发送 FIFO 为空状态。	0x1
[0]	RO	tx_fifo_full	tx fifo 满标志。 0: 发送 FIFO 非满; 1: 发送 FIFO 为满。	0x0

TX_FIFO_CNT

TX_FIFO_CNT 为发送 FIFO 数据计数器。

Offset Address: 0x48 Total Reset Value: 0x0000

Bits	Access	Name	Description	Reset
[15:7]	-	reserved	保留。	0x000
[6:0]	RO	tx_fifo_level	tx fifo 数据计数器。用于指示 tx fifo 内数据量。	0x00

RX_FIFO_CNT

RX_FIFO_CNT 为接收 FIFO 数据计数器。

Offset Address: 0x4C Total Reset Value: 0x0000

Bits	Access	Name	Description	Reset
[15:7]	-	reserved	保留。	0x000
[6:0]	RO	reserved	保留。	0x00

HALT_TX

HALT_TX 为传输挂起寄存器。

Offset Address: 0x50 Total Reset Value: 0x0000

Bits	Access	Name	Description	Reset
[15:1]	-	reserved	保留。	0x0000
[0]	RW	halt_tx	TX 功能挂起使能。 0: 禁用传输挂起功能; 1: 使能传输挂起功能。	0x0

DMA_SW_ACK

DMA_SW_ACK 为 DMA 应答寄存器。

Offset Address: 0x54 Total Reset Value: 0x0000

Bits	Access	Name	Description	Reset
[15:1]	-	reserved	保留。	0x0000
[0]	WC	dma_sw_ack	DMA 软件应答。 0: 未应答; 1: DMA 软件应答。	0x0

BAUD_CTL

BAUD_CTL 为波特率控制寄存器。

Offset Address: 0x58 Total Reset Value: 0x007F

Bits	Access	Name	Description	Reset
[15:8]	-	reserved	保留。	0x00
[7:4]	RW	sample_phase	接收采样相位。	0x7
[3:0]	RW	baud_div	波特率过采样倍数。 0x7: 8 倍波特率采样; 0xF: 16 倍波特率采样; 其他: 不支持。	0xF

STP_CTL

STP_CTL 为停止位控制寄存器。

Offset Address: 0x5C Total Reset Value: 0x0000

Bits	Access	Name	Description	Reset
[15:3]	-	reserved	保留。	0x0000

Bits	Access	Name	Description	Reset
[2]	RW	stp_mode	停止位控制模式。 0: 接收与发送的停止位位宽由 UART_CTL[stp]控制; 1: 发送停止位位宽受 STP_CTL[tx_sp]控制,接收停止位位宽受 STP_CTL[rx_sp]控制。	0x0
[1]	RW	tx_sp	发送停止位位宽。 0: 1bit 停止位; 1: 当 UART_CTL[dlen]为零, 1.5bit 停止位, 否则 2bit 停止位。 注意: 只有当 stp_mode 置位, STP_CTL[tx_sp]才生效。	0x0
[0]	RW	rx_sp	接收停止位位宽。 0: 1bit 停止位; 1: 当 UART_CTL[dlen]为零, 1.5bit 停止位, 否则 2bit 停止位。 注意: 只有当 stp_mode 置位, STP_CTL[rx_sp]才生效。	0x0

UART_PARAMETER

UART_PARAMETER 为 UART 参数寄存器。

Offset Address: 0x60 Total Reset Value: 0x0D04

Bits	Access	Name	Description	Reset
[15:13]	-	reserved	保留。	0x0
[12]	RO	shadow	SHADOW 功能使能。 0: 禁用; 1: 使能。	0x0

Bits	Access	Name	Description	Reset
[11]	RO	dma_mode	DMA 模式查询。 0: DMA_EXTRA 禁用; 1: DMA_EXTRA 使能。	0x1
[10]	RO	afce_mode	AFCE 模式查询。 0: AFCE 模式禁用; 1: AFCE 模式使能。	0x1
[9:8]	RO	apb_data_width	总线接口宽度查询寄存器。 00: APB 数据位宽 8bit; 01: APB 数据位宽 16bit; 10: APB 数据位宽 32bit; 11: 未定义。 注意: 本芯片 UART APB 数据位宽固定为 16bit, 故该寄存器值固定为 0x1。	0x1
[7:0]	RO	fifo_depth	UART 接收发送 FIFO 深度。 0x0: FIFO 深度为 0; 0x1: FIFO 深度为 16; 0x2: FIFO 深度为 32; 0x4: FIFO 深度为 64; 0x8: FIFO 深度为 128; 0x10: FIFO 深度为 256; 0x20: FIFO 深度为 512; 0x40: FIFO 深度为 1024; 0x80: FIFO 深度为 2048; 其他: 未定义。 注意: 本芯片 UART FIFO 深度固定为 64, 故该寄存器值固定为 0x4。	0x04

6.4 I2C

6.4.1 概述

I2C 模块是 APB 总线上的从设备，是 I2C 总线上的主设备。I2C 模块的作用是完成 CPU 对 I2C 总线上从设备的数据读写，CPU 可以连续配置多个发送的数据和接收多个数据。I2C 总线上可挂载多个从设备，芯片支持 2 个 I2C 模块 (I2C0 和 I2C1)。

6.4.2 功能描述

I2C 具有以下功能特点：

- 2.0 版本的 I2C 总线协议，只支持 Master 模式。
- I2C 模块在 APB 总线上执行 APB Slave 的功能，在 I2C 总线上作为 Master，支持多主设备时的总线仲裁。
- I2C 主机可以向从机写入数据，也可以接收从机发来的数据。
- 支持 Clock synchronization 和 Bit and Byte waiting。
- 支持中断或轮询操作。
- I2C 模块支持标准地址 (7bit) 和扩展地址 (10bit)。
- 可以工作在两种速度模式下：标准模式 (100Kbit/s)、快速模式 (400Kbit/s)。
- I2C 模块支持 General Call 和 Start Byte 功能。
- I2C 总线上不支持 CBUS 器件。
- 对接收到的 SDA (Serial Data and Address) 和 SCL (Serial Clock Line) 信号进行滤波。
- 内部包含 1 个 32×8bit 的发送 FIFO 和 1 个 32×8bit 的接收 FIFO。
- 支持硬件检测 FIFO 数据深度并发出相应中断。
- 兼容不使用 FIFO 和使用 FIFO 两种工作方式

6.4.3 工作方式

I2C 包含以下两种工作场景：

- 主机仅对单个数据发送和接收 (不使用 FIFO)。
- 主机连续发送多个数据、连续接收多个数据 (使用 FIFO)。

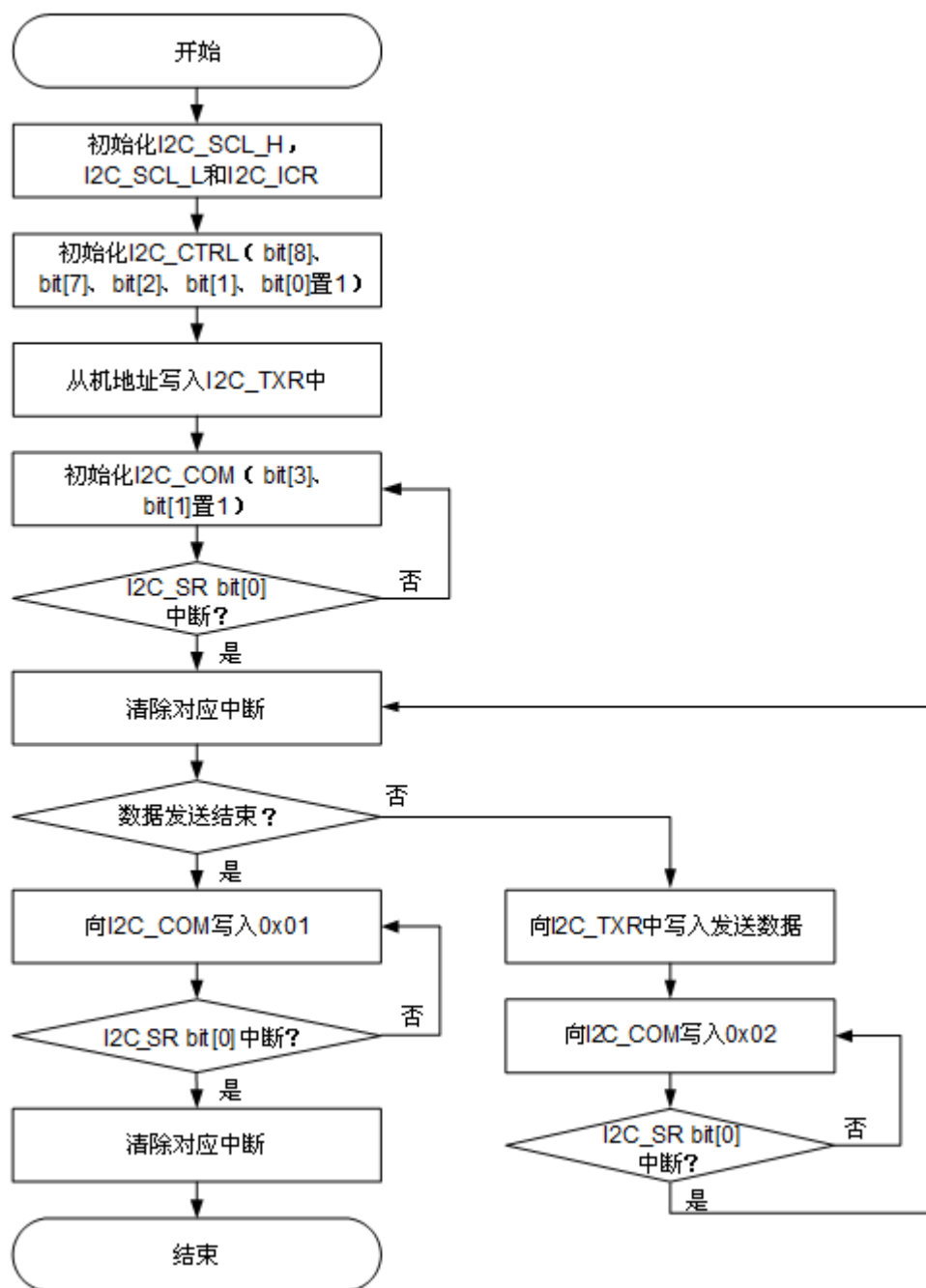
6.4.3.1 不使用 FIFO

I2C 初始化配置前，需要先将 I2C 的工作时钟配置为 80M，并参考 IO_MUX 配置 I2C 对应的 IO 复用。

I2C 主机发送数据流程

I2C 主机发送数据流程如图 6-2 所示。

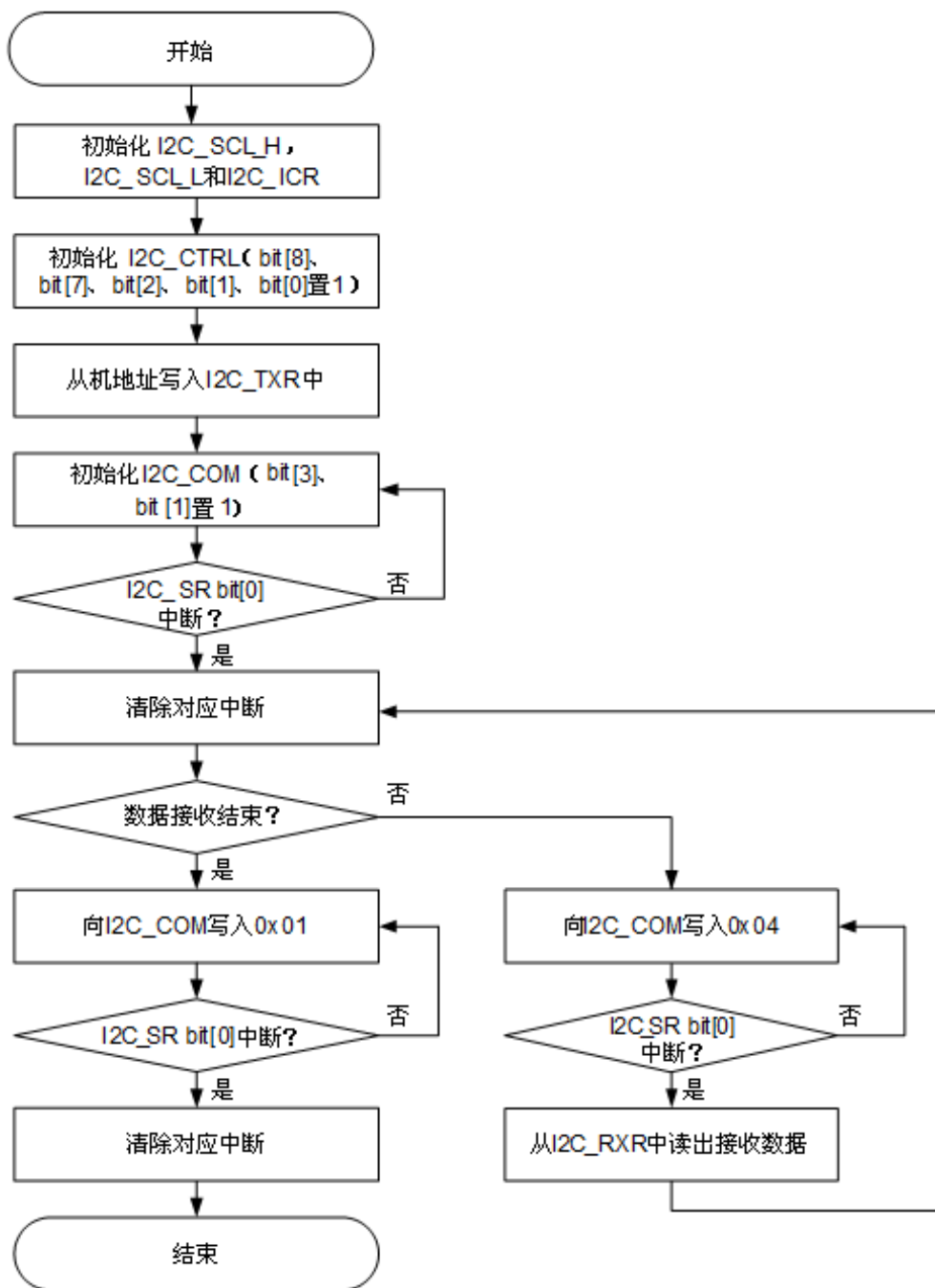
图6-2 I2C 主机发送数据（不使用 FIFO）流程图



I2C 主机接收数据流程

I2C 主机接收数据流程如图 6-3 所示。

图6-3 I2C 主机接收数据（不使用 FIFO）流程图



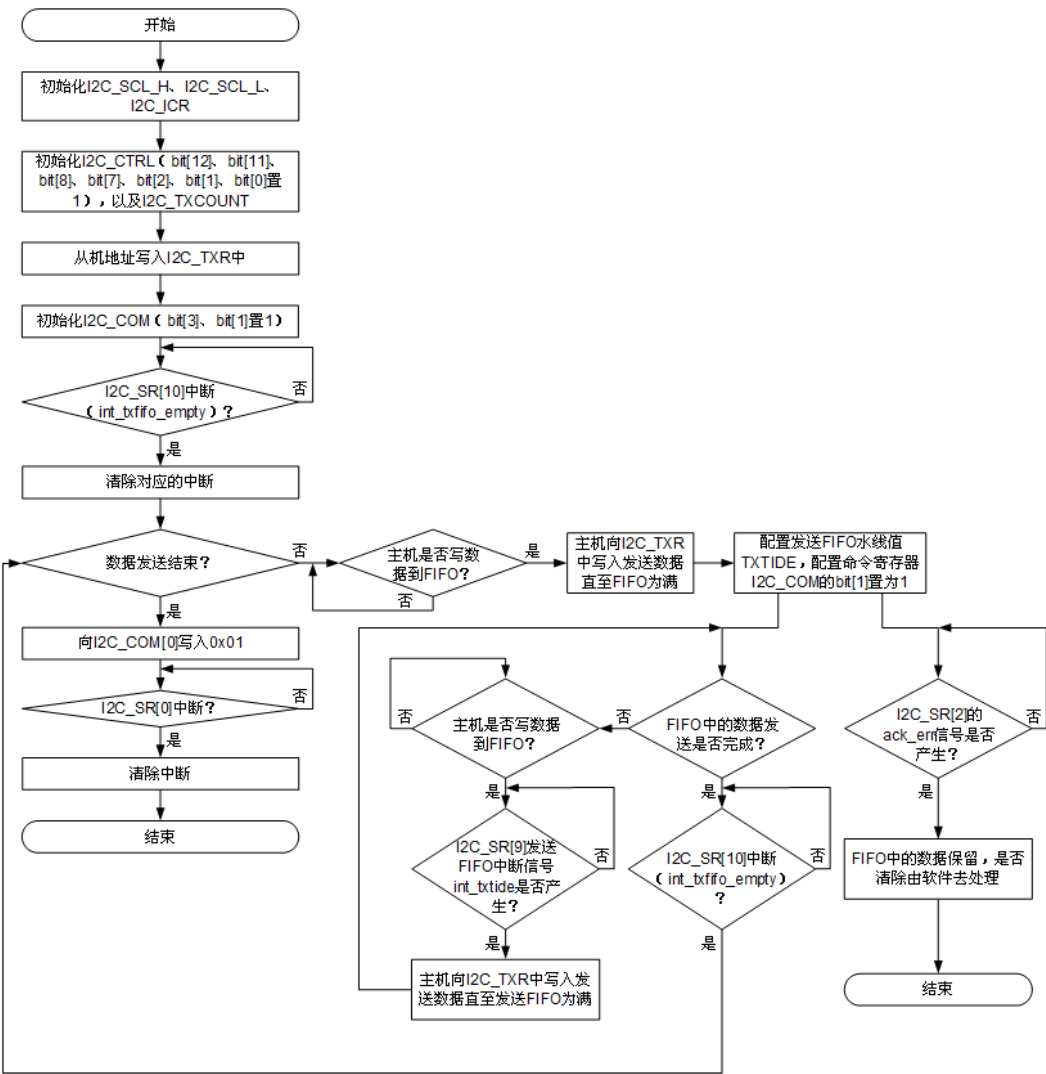
6.4.3.2 使用 FIFO

I2C 初始化配置前，需要先将 I2C 的工作时钟配置为 80M，并参考 IO_MUX 配置 I2C 对应的 IO 复用。

I2C 主机发送数据流程

I2C 主机发送数据流程图如图 6-4 所示。

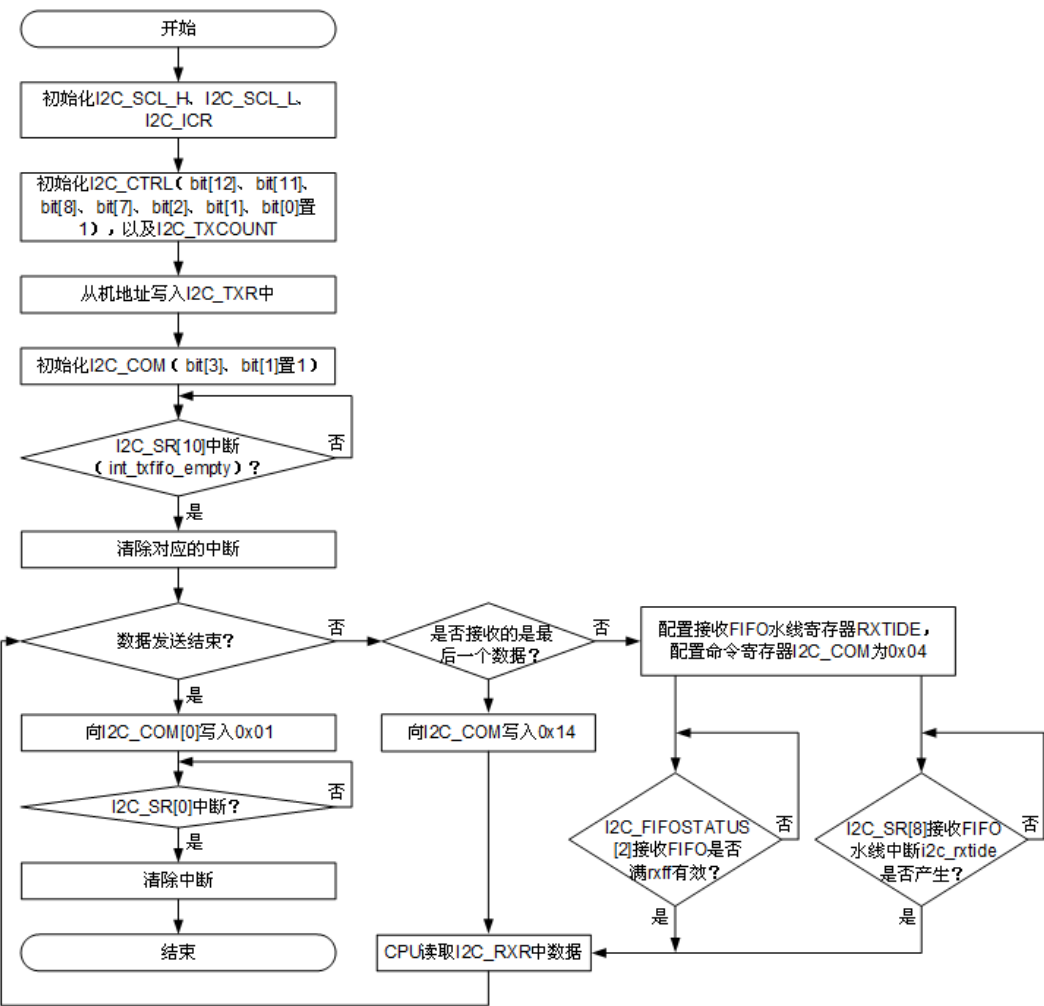
图6-4 I2C 主机发送数据（使用 FIFO）流程图



I2C 主机接收数据流程

I2C 主机接收数据流程图如图 6-5 所示。

图6-5 I2C 主机接收数据（使用 FIFO）流程图



6.4.4 寄存器概览

I2C 寄存器概览如表 6-9 所示。

表6-9 I2C 寄存器概览（基址是 I2C0：0x44018000、I2C1：0x44008100）

偏移地址	名称	描述
0x00	I2C_CTRL	I2C 控制寄存器。
0x04	I2C_COM	I2C 模块的命令寄存器。
0x08	I2C_ICR	I2C 模块的中断清除寄存器。
0x0C	I2C_SR	I2C 模块状态寄存器。

偏移地址	名称	描述
0x10	I2C_SCL_H	I2C 总线 SCL 信号高电平周期数寄存器。
0x14	I2C_SCL_L	I2C 总线 SCL 信号低电平周期数寄存器。
0x18	I2C_TXR	I2C 发送数据寄存器。
0x1C	I2C_RXR	I2C 接收数据寄存器。
0x20	I2C_FIFOSTATUS	FIFO 状态寄存器。
0x24	I2C_TXCOUNT	发送 FIFO 数据个数寄存器。
0x28	I2C_RXCOUNT	接收 FIFO 数据个数寄存器。
0x2C	I2C_RXTIDE	接收 FIFO 的溢出阈值寄存器。
0x30	I2C_TXTIDE	发送 FIFO 的溢出阈值寄存器。
0x34	I2C_FTRPER	毛刺过滤配置寄存器。

6.4.5 寄存器描述

I2C_CTRL

I2C_CTRL 为 I2C 控制寄存器。

Offset Address: 0x00 Total Reset Value: 0x0000_0000

Bits	Access	Name	Description	Reset
[31:13]	-	reserved	保留。	0x00000
[12]	RW	int_txfifo_over_mask	发送 FIFO 数据发送完成中断屏蔽。 0: 屏蔽; 1: 不屏蔽。	0x0
[11]	RW	mode_ctrl	I2C 工作模式选择。 0: 不使用 FIFO 传输模式; 1: 使用 FIFO 传输模式。	0x0

Bits	Access	Name	Description	Reset
[10]	RW	int_txide_mask	发送 FIFO 溢出中断屏蔽。 0: 屏蔽; 1: 不屏蔽。	0x0
[9]	RW	int_rxtide_mask	接收 FIFO 溢出中断屏蔽。 0: 屏蔽; 1: 不屏蔽。	0x0
[8]	RW	i2c_en	I2C 使能。 0: 禁止; 1: 使能。	0x0
[7]	RW	int_mask	I2C 中断总屏蔽。 0: 屏蔽; 1: 不屏蔽。	0x0
[6]	RW	int_start_mask	主机开始条件发送结束中断屏蔽。 0: 屏蔽; 1: 不屏蔽。	0x0
[5]	RW	int_stop_mask	主机停止条件发送结束中断屏蔽。 0: 屏蔽; 1: 不屏蔽。	0x0
[4]	RW	int_tx_mask	主机发送中断屏蔽。 0: 屏蔽; 1: 不屏蔽。	0x0
[3]	RW	int_rx_mask	主机接收中断屏蔽。 0: 屏蔽; 1: 不屏蔽。	0x0
[2]	RW	int_ack_err_mask	从机 ACK 错误中断屏蔽。 0: 屏蔽; 1: 不屏蔽。	0x0

Bits	Access	Name	Description	Reset
[1]	RW	int_arb_loss_mask	总线仲裁失败中断屏蔽。 0：屏蔽； 1：不屏蔽。	0x0
[0]	RW	int_done_mask	总线传输完成中断屏蔽。 0：屏蔽； 1：不屏蔽。	0x0

I2C_COM

I2C_COM 为 I2C 模块的命令寄存器。

 说明

在系统初始化时配置或配置前，需要清除对应中断标志。I2C_COM bit[3:0]在操作结束后将自动清0。

Offset Address: 0x04 Total Reset Value: 0x0000_0000

Bits	Access	Name	Description	Reset
[31:5]	-	reserved	保留。	0x00000000
[4]	RW	op_ack	主机作为接收器是否发送 ACK。 0：发送； 1：不发送。	0x0
[3]	RW	op_start	产生开始条件操作。 0：操作结束； 1：操作有效。	0x0
[2]	RW	op_rd	产生读操作。 0：操作结束； 1：操作有效。	0x0
[1]	RW	op_we	产生写操作。	0x0

Bits	Access	Name	Description	Reset
			0: 操作结束; 1: 操作有效。	
[0]	RW	op_stop	产生停止条件操作。 0: 操作结束; 1: 操作有效。	0x0

I2C_ICR

I2C_ICR 为 I2C 模块的中断清除寄存器。

说明

新中断到来时，I2C 模块会自动将 I2C_ICR 相应位清 0。

Offset Address: 0x08 Total Reset Value: 0x0000_0000

Bits	Access	Name	Description	Reset
[31:10]	-	reserved	保留。	0x000000
[9]	WC	clr_int_txfifo_over	发送 FIFO 数据发送完成中断标志清除。 0: 不清除; 1: 清除。	0x0
[8]	WC	clr_int_txtide	发送 FIFO 溢出中断标志清除。 0: 不清除; 1: 清除。	0x0
[7]	WC	clr_int_rxtide	接收 FIFO 溢出中断标志清除。 0: 不清除; 1: 清除。	0x0
[6]	WC	clr_int_start	主机开始条件发送结束中断标志清除。 0: 不清除;	0x0

Bits	Access	Name	Description	Reset
			1：清除。	
[5]	WC	clr_int_stop	主机停止条件发送结束中断标志清除。 0：不清除； 1：清除。	0x0
[4]	WC	clr_int_tx	主机发送中断标志清除。 0：不清除； 1：清除。	0x0
[3]	WC	clr_int_rx	主机接收中断标志清除。 0：不清除； 1：清除。	0x0
[2]	WC	clr_int_ack_err	从机 ACK 错误中断标志清除。 0：不清除； 1：清除。	0x0
[1]	WC	clr_int_arb_los	总线仲裁失败中断标志清除。 0：不清除； 1：清除。	0x0
[0]	WC	clr_int_done	总线传输完成中断标志清除。 0：不清除； 1：清除。	0x0

I2C_SR

I2C_SR 为 I2C 模块状态寄存器。

 说明

I2C_SR bit[1]表示 I2C 总线仲裁失败。当 I2C_SR bit[1]有效时，当前操作失败。在清 I2C_SR bit[1]之前，需要清除其他中断标志，然后清除 I2C_COM 或向 I2C_COM 写入新的操作命令，最后清除 I2C_SR bit[1]。

Offset Address: 0x0C Total Reset Value: 0x0000_0000

Bits	Access	Name	Description	Reset
[31:11]	-	reserved	保留。	0x000000
[10]	RO	int_txfifo_over	发送 FIFO 数据发送完成中断标志。 0: 无中断标志产生; 1: 中断标志产生。	0x0
[9]	RO	int_txtide	发送 FIFO 溢出中断标志。 0: 无中断标志产生; 1: 中断标志产生。	0x0
[8]	RO	int_rxtide	接收 FIFO 溢出中断标志。 0: 无中断标志产生; 1: 中断标志产生。	0x0
[7]	RO	bus_busy	总线忙。 0: 空闲; 1: 忙。	0x0
[6]	RO	int_start	主机开始条件发送结束中断标志。 0: 无中断标志产生; 1: 中断标志产生。	0x0
[5]	RO	int_stop	主机停止条件发送结束中断标志。 0: 无中断标志产生; 1: 中断标志产生。	0x0
[4]	RO	int_tx	主机发送中断标志。 0: 无中断标志产生; 1: 中断标志产生。	0x0
[3]	RO	int_rx	主机接收中断标志。 0: 无中断标志产生;	0x0

Bits	Access	Name	Description	Reset
			1: 中断标志产生。	
[2]	RO	int_ack_err	从机 ACK 错误中断标志。 0: 无中断标志产生; 1: 中断标志产生。	0x0
[1]	RO	int_arb_loss	总线仲裁失败中断标志。 0: 无中断标志产生; 1: 中断标志产生。	0x0
[0]	RO	int_done	总线传输完成中断标志。 0: 无中断标志产生; 1: 中断标志产生。	0x0

I2C_SCL_H

I2C_SCL_H 为 I2C 总线 SCL 信号高电平周期数寄存器。

说明

在系统初始化时配置或配置前使 I2C_CTRL bit[7]=0。

Offset Address: 0x10 Total Reset Value: 0x0000_0000

Bits	Access	Name	Description	Reset
[31:16]	-	reserved	保留。	0x0000
[15:0]	RW	scl_h	I2C 总线 SCL 信号高电平周期数寄存器。用于配置 I2C 模块工作时 SCL 高电平周期数，配置数值乘 2 等于 SCL 高电平周期数。	0x0000

I2C_SCL_L

I2C_SCL_L 为 I2C 总线 SCL 信号低电平周期数寄存器。

 说明

在系统初始化时配置或配置前使 I2C_CTRL bit[7]=0。

Offset Address: 0x14 Total Reset Value: 0x0000_0000

Bits	Access	Name	Description	Reset
[31:16]	-	reserved	保留。	0x0000
[15:0]	RW	scl_l	I2C 总线 SCL 信号低电平周期数寄存器。用于配置 I2C 模块工作时 SCL 低电平周期数，配置数值乘 2 等于 SCL 低电平周期数。	0x0000

I2C_TXR

I2C_TXR 为 I2C 发送数据寄存器。

 说明

不使用 FIFO 模式下，发送结束后，I2C 模块不会修改 I2C_TXR 内容；使用 FIFO 模式下，写入的数据会自动载入到发送 FIFO 中保存直到该数据发送结束。

Offset Address: 0x18 Total Reset Value: 0x0000_0000

Bits	Access	Name	Description	Reset
[31:8]	-	reserved	保留。	0x000000
[7:0]	RW	i2c_txr	主机发送数据。用于配置 I2C 模块工作时发送数据。	0x00

I2C_RXR

I2C_RXR 为 I2C 接收数据寄存器。

 说明

不使用 FIFO 模式下，I2C_RXR 数据在 I2C_SR bit[3]=1 时，数据有效。同时数据将保持到下一个读操作前。使用 FIFO 模式下，读取 I2C_RXR 会直接从接收 FIFO 中取数据。

Offset Address: 0x1C Total Reset Value: 0x0000_0000

Bits	Access	Name	Description	Reset
[31:8]	-	reserved	保留。	0x000000
[7:0]	RO	i2c_rxr	主机接收数据。用于主机接收从机数据。	0x00

I2C_FIFOSTATUS

I2C_FIFOSTATUS 为 FIFO 状态寄存器。

Offset Address: 0x20 Total Reset Value: 0x0000_0000

Bits	Access	Name	Description	Reset
[31:4]	-	reserved	保留。	0x00000000
[3]	RO	rxfe	接收 FIFO 空状态。 0: 非空; 1: 空。	0x0
[2]	RO	rxff	接收 FIFO 满状态。 0: 未滿; 1: 满。	0x0
[1]	RO	txfe	发送 FIFO 空状态。 0: 非空; 1: 空。	0x0
[0]	RO	txff	发送 FIFO 满状态。 0: 未滿; 1: 满。	0x0

I2C_TXCOUNT

I2C_TXCOUNT 为发送 FIFO 数据个数寄存器。

Offset Address: 0x24 Total Reset Value: 0x0000_0000

Bits	Access	Name	Description	Reset
[31:6]	-	reserved	保留。	0x00000000
[5:0]	WC	txcount	读该寄存器返回发送 FIFO 中的字符数，写该寄存器（任意值）将清空发送 FIFO。	0x00

I2C_RXCOUNT

I2C_RXCOUNT 为接收 FIFO 数据个数寄存器。

Offset Address: 0x28 Total Reset Value: 0x0000_0000

Bits	Access	Name	Description	Reset
[31:6]	-	reserved	保留。	0x00000000
[5:0]	WC	rxcount	读该寄存器返回接收 FIFO 中的字符数，写该寄存器（任意值）将清空接收 FIFO。	0x00

I2C_RXTIDE

I2C_RXTIDE 为接收 FIFO 的溢出阈值寄存器。

Offset Address: 0x2C Total Reset Value: 0x0000_0001

Bits	Access	Name	Description	Reset
[31:6]	-	reserved	保留。	0x00000000
[5:0]	RW	rx tide	设置 int_rxtide 中断的触发值。 RXFIFO 中的字符个数 ≥ I2C_RXTIDE[rx tide]时会触发接收 FIFO 溢出中断。	0x01

I2C_TXTIDE

I2C_TXTIDE 为发送 FIFO 的溢出阈值寄存器。

说明

TXFIFO 中的字符只有在成功发送后才会被移除。

Offset Address: 0x30 Total Reset Value: 0x0000_0001

Bits	Access	Name	Description	Reset
[31:6]	-	reserved	保留。	0x00000000
[5:0]	RW	txtide	设置 int_txtide 中断的触发值。 TXFIFO 中的字符个数≤ I2C_TXTIDE[txtide]时会触发发送 FIFO 溢出中断。	0x01

I2C_FTRPER

I2C_FTRPER 为毛刺过滤配置寄存器。

Offset Address: 0x34 Total Reset Value: 0x0000_000F

Bits	Access	Name	Description	Reset
[31:4]	-	reserved	保留。	0x00000000
[3:0]	RW	ftper	毛刺过滤配置寄存器。默认 15 个 时钟周期，需要 SDA 需要再 SCL 为高时，电平保持的时间。scl 为 高时，判断 sda 电平持续时间，单 位为 ic_clk 的时钟个数，持续时间 大于该值才认为该电平为接收值。 注意：配置值需要≤ ((fic_clk/ (8*fsc1))-2)，其中 fic_clk 为 I2C 工 作时钟频率，fsc1 为 I2C 总线传输 速率。	0xF

6.5 SPI

6.5.1 概述

SPI 实现数据的串并、并串转换，可以作为 Master 或 Slave 与外部设备进行同步串行通信（外围设备必须支持 SPI 帧格式）。芯片的 SPI 工作参考时钟为 240MHz。

6.5.2 功能描述

SPI 具有以下功能特点：

- 支持接口时钟频率可编程。
 - 作为主设备：最大支持 10M 接口频率工作。
 - 作为从设备：最大支持 32M 接口频率工作。
- 支持 SPI 帧格式，分为以下三种：
 - Motorola 帧格式
 - TI (Texas Instruments) 帧格式
 - National Microwire 帧格式
- 串行数据帧长度可编程：4bit ~ 16bit。
- 支持发送 FIFO 中断、接收 FIFO 中断独立屏蔽。
- 内部提供环回测试模式。
- 支持 DMA 操作，但不支持作为 DMA 的流控设备。

6.5.3 工作方式

6.5.3.1 接口信号

Master 模式

SPI 接口信号描述如表 6-10 所示。

表6-10 SPI Master 接口信号描述

信号名	宽度 (bit)	方向	功能描述
SPI_C LK	1	O	SPI 串行时钟信号，由 Master 产生并控制。
SPI_C	1	O	SPI 片选信号，低电平有效。

信号名	宽度 (bit)	方向	功能描述
S_N			
SPI_DI	1	I	输入数据。
SPI_DO	1	O	输出数据。

Slave 模式

SPI 接口信号描述如表 6-11 所示。

表6-11 SPI Slave 接口信号描述

信号名	宽度 (bit)	方向	功能描述
SPI_CLK	1	I	SPI 串行时钟信号，由 Master 产生并控制。
SPI_CS_N	1	I	SPI 片选信号，低电平有效。
SPI_DI	1	I	输入数据。
SPI_DO	1	O	输出数据。

6.5.3.2 传输帧格式

SPI 可以与支持 SPI 帧格式的任意串行 Master 或 Slave 器件进行通信。

说明

图中的缩略语含义为：MSB (Most Significant Bit)，LSB (Least Significant Bit)，Q (Q is an undefined signal)。

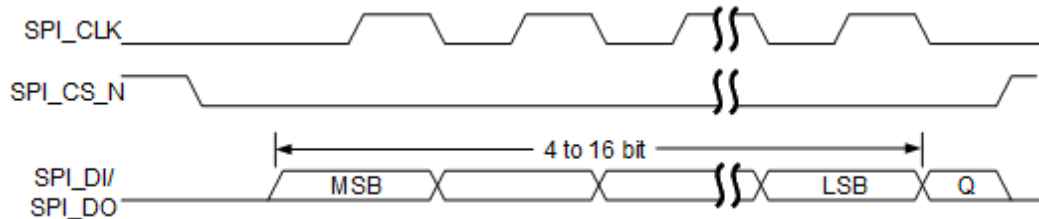
SPO 表示 SPICKOUT 极性，SPH 表示 SPICKOUT 相位。

Motorola 帧格式

【SPO=0、SPH=0】

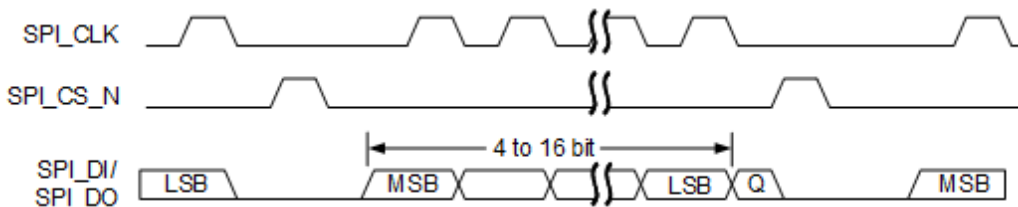
SPI 单帧帧格式如图 6-6 所示。

图6-6 SPI 单帧格式 (SPO=0、SPH=0)



SPI 连续帧格式如图 6-7 所示。

图6-7 SPI 连续帧格式 (SPO=0、SPH=0)



在该模式下，当 SPI 处于空闲状态时：

- SPI_CLK 信号设置为低电平。
- SPI_CS_N 信号设置为高电平。
- 发送数据线 SPI_DO 强制为低电平。

当 SPI 处于使能状态，而且发送 FIFO 内有有效数据时，设置 SPI_CS_N 信号为低电平表示开始传输数据。此时使能 Slave 的数据放入 Master 的输入 SPI_DI。半个 SPI_CLK 时钟周期之后，有效的 Master 数据传输到 SPI_DO。此时 Master 和 Slave 数据都已经有效，SPI_CLK 管脚在接下来的半个 SPI_CLK 时钟周期之后变为高电平。数据在 SPI_CLK 时钟的上升沿被捕获，在时钟的下降沿被传送。

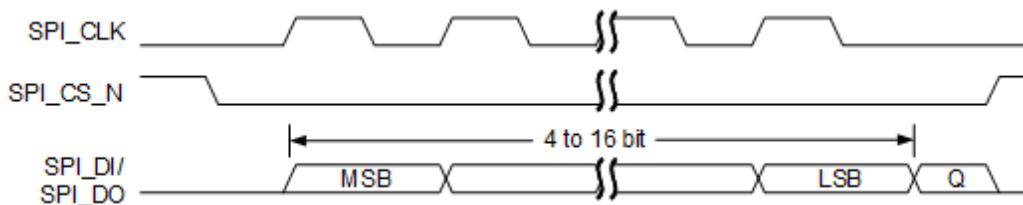
如果传输单个 word (4 ~ 16 bit)，当捕捉到最后 1bit 数据时，SPI_CS_N 在接下来的 1 个 SPI_CLK 时钟周期之后恢复为高电平。

如果是连续传输，SPI_CS_N 信号在每个 word (4 ~ 16 bit) 传输之间会拉高一个时钟周期。这是因为 SPH 为 0 时，Slave 选择管脚会固定其内部串行设备寄存器的数据，使它不会变化。因此在连续传输时，主设备必须在每个 word (4 ~ 16 bit) 传输之间将 SPI_CS_N 信号拉高。连续传输结束时，SPI_CS_N 在捕捉到最后 1bit 之后的 1 个 SPI_CLK 时钟周期之后恢复为高电平。

【SPO=0、SPH=1】

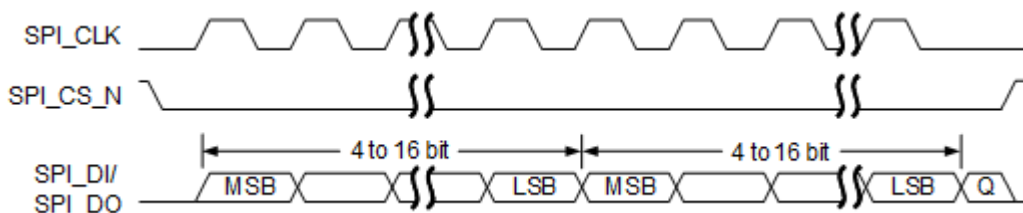
SPI 单帧格式如图 6-8 所示。

图6-8 SPI 单帧格式 (SPO=0、SPH=1)



SPI 连续帧格式如图 6-9 所示。

图6-9 SPI 连续帧格式 (SPO=0、SPH=1)



在该模式下，当 SPI 处于空闲状态时：

- SPI_CLK 信号设置为低电平。
- SPI_CS_N 信号设置为高电平。
- 发送数据线 SPI_DO 强制为低电平。

当 SPI 为使能状态，且发送 FIFO 内有有效数据时，设置 SPI_CS_N 信号为低电平表示开始传输数据。此时 Slave 的数据立刻发送到 Master 的接收数据线 SPI_DI。半个 SPI_CLK 时钟周期之后，Master 的有效数据在自身的传输线上有效，同时 SPI_CLK 产生第一个上升沿。数据在 SPI_CLK 时钟的下降沿被捕获，在时钟的上升沿被传送。

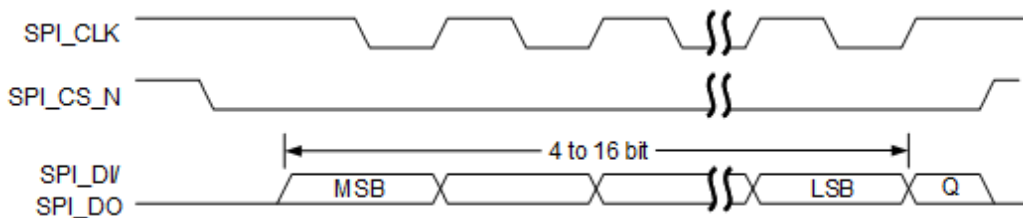
如果传输单个 word (4 ~ 16 bit)，当捕捉到最后 1bit 数据时，SPI_CS_N 在接下来的 1 个 SPI_CLK 时钟之后恢复为高电平。

当连续传输时，在传输数据 word (4 ~ 16 bit) 之间 SPI_CS_N 保持为低电平。连续传输结束时，SPI_CS_N 在最后 1bit 捕捉之后的 1 个 SPI_CLK 时钟之后恢复为高电平。

【SPO=1、SPH=0】

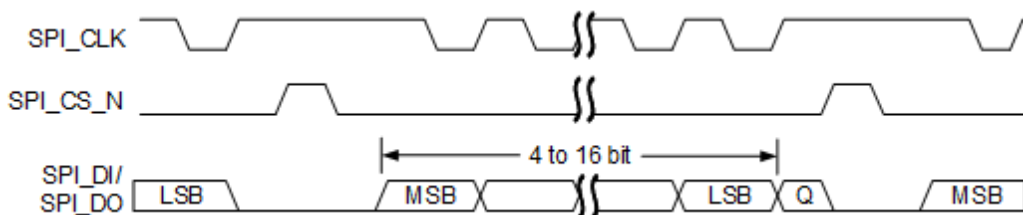
SPI 单帧格式如图 6-10 所示。

图6-10 SPI 单帧格式 (SPO=1、SPH=0)



SPI 连续帧格式如图 6-11 所示。

图6-11 SPI 连续帧格式 (SPO=1、SPH=0)



在该配置下，当 SPI 处于空闲状态时：

- SPI_CLK 信号设置为高电平。
- SPI_CS_N 信号设置为高电平。
- 发送数据线 SPI_DO 强制为低电平。

当 SPI 为使能状态，且发送 FIFO 内有有效数据时，设置 SPI_CS_N 信号为低电平表示开始传输数据。此时 Slave 的数据立刻发送到 Master 的接收数据线 SPI_DI。半个 SPI_CLK 周期之后，Master 的有效数据传送到 SPI_DO。再过半个 SPI_CLK 时钟周期之后，SPI_CLK Master 管脚设置为低电平，这表示数据在 SPI_CLK 时钟的下降沿被捕获，在 SPI_CLK 时钟的上升沿被传送。

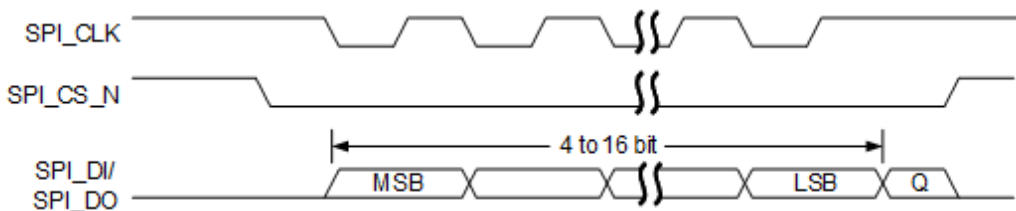
如果传输单个 word (4 ~ 16 bit)，当捕捉到最后 1bit 数据时，SPI_CS_N 在接下来的 1 个 SPI_CLK 时钟之后恢复为高电平。

如果是连续的传输，SPI_CS_N 信号在每个 word (4 ~ 16 bit) 传输之间必须拉高。这是因为当 SPH 为 0 时，Slave 选择管脚固定其内部串行设备寄存器的数据，使它不会变化。SPI_CS_N 在捕捉到最后 1bit 数据之后的 1 个 SPI_CLK 时钟周期之后恢复为高电平。

【SPO=1、SPH=1】

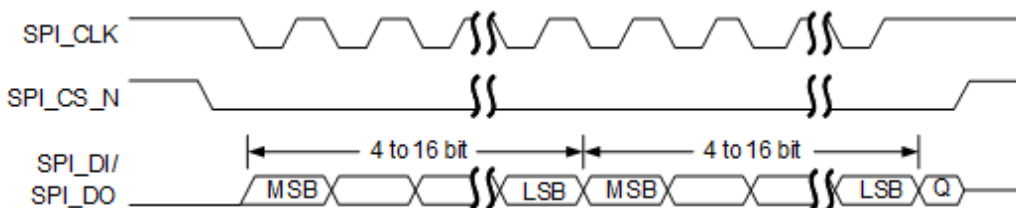
SPI 单帧格式如图 6-12 所示。

图6-12 SPI 单帧格式 (SPO=1、SPH=1)



SPI 连续帧格式如图 6-13 所示。

图6-13 SPI 连续帧格式 (SPO=1、SPH=1)



在该模式下，当 SPI 处于空闲状态时：

- SPI_CLK 信号设置为高电平。
- SPI_CS_N 信号设置为高电平。
- 发送数据线 SPI_DO 强制为低电平。

当 SPI 为使能状态，且发送 FIFO 内有有效数据时，设置 SPI_CS_N Master 信号为低电平表示开始传输数据。此时 Slave 的数据立刻发送到 Master 的接收数据线 SPI_DI。半个 SPI_CLK 时钟周期后，Master 数据在自身的传输线上有效，同时时钟 SPI_CLK 产生第一个下降沿。数据在 SPI_CLK 时钟的上升沿被捕获，在时钟的下降沿被传送。

当传输单个 word (4 ~ 16 bit) 时，SPI_CS_N 在传输的最后 1bit 捕获之后的 1 个 SPI_CLK 时钟周期之后恢复为高电平。

如果是连续传输，SPI_CS_N 信号始终保持为低电平。SPI_CS_N 在捕获到最后 1bit 之后的 1 个 SPI_CLK 时钟周期之后恢复到高状态。对于连续传输来说，SPI_CS_N 在传输过程中一直保持为低电平，结束方式与单个传输方式相同。

6.5.3.3 工作模式

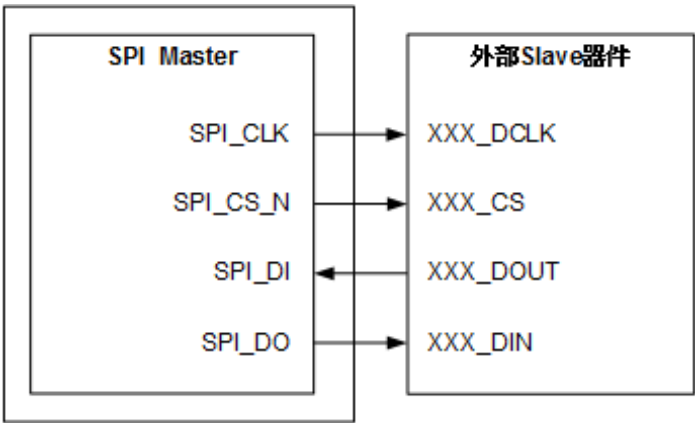
SPI 的工作方式分为中断或查询方式下的数据传输。

SPI 模块支持 Master 以及 Slave 两种基本工作模式。

Master 模式

在 Master 模式下，SPI 模块初始化并直接控制所有的外围设备。配置为 Master 的 SPI 与外围 Slave 器件通信示意图如图 6-14 所示。

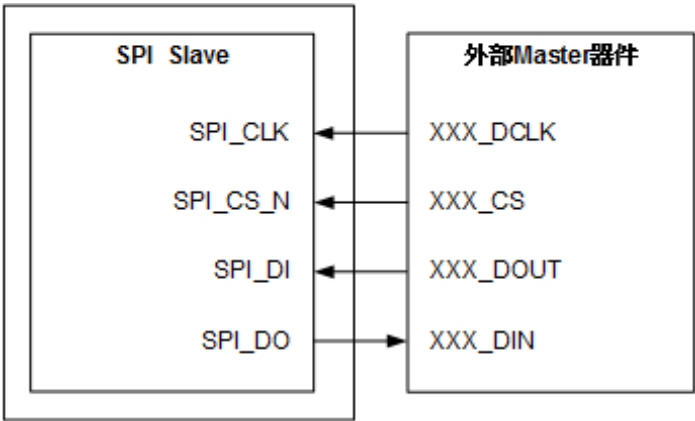
图6-14 SPI 接单 Slave 时的应用



Slave 模式

在 Slave 模式下，SPI 由串行总线上的 Master 进行初始化和控制。配置为 Slave 的 SPI 与外围 Master 器件通信示意图如图 6-15 所示。

图6-15 SPI 接 Master 时的应用



6.5.3.4 应用说明

Master TX 模式

1. 初始化

步骤 1 进行时钟的初始化配置，将 SPI 的工作时钟切换到 240M。

步骤 2 参考 IO_MUX，配置好 SPI 对应的 IO 复用，以及 IO 的驱动能力。

步骤 3 写地址 0x4400_0250 为 0x1，设置 SPI 为 Master 模式。

步骤 4 写 SPI_ER[start_en]为 0x0，关闭 SPI 的 start 使能信号。

步骤 5 写 SPI_CTRA[prs]，SPI_CTRA[scpol]，SPI_CTRA[scpoh]，配置 SPI 通信协议模式。

步骤 6 写 SPI_CTRA[trsm]为 0x1，配置 SPI 为只发送模式。

步骤 7 写 SPI_CTRA[dfs32]为 0x1f，配置 SPI 的串行数据长度为 32bit。

步骤 8 写 SPI_BRS[frdv]，配置 SPI 内部的分频能力。

步骤 9 写 SPI_RSDR[rxdsd]为 0x1，配置 SPI 内部的采样点位置。

步骤 10 写 SPI_CTRB[nrdf]，配置 SPI 需要接收的帧长度。

步骤 11 写 SPI_SLENR[ssef]为 0x1，打开的 SPI 的片选能力。

步骤 12 写 SPI_TWLR[twl]，配置 SPI 发送 FIFO 水线阈值。

步骤 13 写 SPI_ER[start_en]为 0x1，打开 SPI 的 start 使能信号。

----结束

2. 数据发送

步骤 1 CPU 接收到中断，读取 SPI_INSR[tfeis]为 1，表征此时 SPI 的 TXFIFO 为空。

步骤 2 写 SPI_DRNM[datar]，将目标数据写到了 SPI 的 TXFIFO 中。

----结束

Master RX 模式

1. 初始化

步骤 1 进行时钟的初始化配置，将 SPI 的工作时钟切换到 240M。

- 步骤 2 参考 IO_MUX，配置好 SPI 对应的 IO 复用，以及 IO 的驱动能力。
- 步骤 3 写地址 0x4400_0250 为 0x1，设置 SPI 为 Master 模式
- 步骤 4 写 SPI_ER[start_en]为 0x0，关闭 SPI 的 start 使能信号。
- 步骤 5 写 SPI_CTRA[prs]，SPI_CTRA[scpol]，SPI_CTRA[scpoh]，配置 SPI 通信协议模式。
- 步骤 6 写 SPI_CTRA[trsm]为 0x2，配置 SPI 为只接收模式。
- 步骤 7 写 SPI_CTRA[dfs32]为 0x1f，配置 SPI 的串行数据长度为 32bit。
- 步骤 8 写 SPI_BRS[frdv]，配置 SPI 内部的分频能力。
- 步骤 9 写 SPI_RSDR[rxdsd]为 0x1，配置 SPI 内部的采样点位置。
- 步骤 10 写 SPI_CTRB[nrdf]，配置 SPI 需要接收的帧长度。
- 步骤 11 写 SPI_SLENR[ssef]为 0x1，打开的 SPI 的片选能力。
- 步骤 12 写 SPI_RWLR[rlw]，配置 SPI 接收 FIFO 水线阈值。
- 步骤 13 写 SPI_ER[start_en]为 0x1，打开 SPI 的 start 使能信号。

----结束

2. 数据接收

- 步骤 1 CPU 接收到中断，读取 SPI_INSR[rffis]为 1，表征此时 QSPI 的 RXFIFO 中数据量达到了水线值。
- 步骤 2 读 SPI_DRNM[datar]，将需要的数据从 SPI 的 RXFIFO 中读取出来。

----结束

Slave TX 模式

1. 初始化

- 步骤 1 进行时钟的初始化配置，将 SPI 的工作时钟切换到 240M。
- 步骤 2 参考 IO_MUX，配置好 SPI 对应的 IO 复用，以及 IO 的驱动能力。
- 步骤 3 写地址 0x4400_0250 为 0x0，设置 SPI 为 Slave 模式
- 步骤 4 写 SPI_ER[start_en]为 0x0，关闭 SPI 的 start 使能信号。
- 步骤 5 写 SPI_CTRA[prs]，SPI_CTRA[scpol]，SPI_CTRA[scpoh]，配置 SPI 通信协议模式。

步骤 6 写 `SPI_CTRA[trsm]` 为 0x1，配置 SPI 为只发送模式。

步骤 7 写 `SPI_CTRA[dfs32]` 为 0x1f，配置 SPI 的串行数据长度为 32bit。

步骤 8 写 `SPI_BRS[frdv]`，配置 SPI 内部的分频能力。

步骤 9 写 `SPI_TWLR[twl]`，配置 SPI 发送 FIFO 水线阈值。

步骤 10 写 `SPI_ER[start_en]` 为 0x1，打开 SPI 的 start 使能信号。

----结束

2. 数据发送

步骤 1 CPU 接收到中断，读取 `SPI_INSR[tfeis]` 为 1，表征此时 SPI 的 TXFIFO 为空。

步骤 2 写 `SPI_DRNM[datar]`，将目标数据写进 SPI 的 TXFIFO 中。

----结束

Slave RX 模式

1. 初始化

步骤 1 进行时钟的初始化配置，将 SPI 的工作时钟切换到 240M。

步骤 2 参考 IO_MUX，配置好 SPI 对应的 IO 复用，以及 IO 的驱动能力。

步骤 3 写地址 0x4400_0250 为 0x0，设置 SPI 为 Slave 模式

步骤 4 写 `SPI_ER[start_en]` 为 0x0，关闭 SPI 的 start 使能信号。

步骤 5 写 `SPI_CTRA[prs]`，`SPI_CTRA[scpol]`，`SPI_CTRA[scpoh]`，配置 SPI 通信协议模式。

步骤 6 写 `SPI_CTRA[trsm]` 为 0x2，配置 SPI 为只接收模式。

步骤 7 写 `SPI_CTRA[dfs32]` 为 0x1f，配置 SPI 的串行数据长度为 32bit。

步骤 8 写 `SPI_BRS[frdv]`，配置 SPI 内部的分频能力。

步骤 9 写 `SPI_RWLR[rwl]`，配置 SPI 接收 FIFO 水线阈值。

步骤 10 写 `SPI_ER[start_en]` 为 0x1，打开 SPI 的 start 使能信号。

----结束

2. 数据接收

步骤 1 CPU 接收到中断，读取 `SPI_INSR[rffis]` 为 1，表征此时 QSPI 的 RXFIFO 中数据量达到了水线值。

步骤 2 读 `SPI_DRNM[datar]`，将需要的数据从 SPI 的 RXFIFO 中读取出来。

----结束

6.5.4 寄存器概览

SPI_CFG0_REGBANK 寄存器概览如表 6-12 所示。

表6-12 SPI_CFG0_REGBANK 寄存器概览（基址是 0x044020000）

偏移地址	名称	描述
0x0000	<code>SPI_ER</code>	SPI 使能寄存器。
0x0004	<code>SPI_CTRA</code>	SPI 控制寄存器 A。
0x0008	<code>SPI_CTRB</code>	SPI 控制寄存器 B。
0x000C	<code>SPI_ENHCTL</code>	SPI Enhanced 控制寄存器。
0x0010	<code>SPI_MCR</code>	microwire 控制寄存器。
0x0014	<code>SPI_BRS</code>	波特率控制寄存器。
0x0018	<code>SPI_DCR</code>	DMA 控制寄存器。
0x001C	<code>SPI_DRDL</code>	DMA 接收水线寄存器。
0x0020	<code>SPI_DTDL</code>	DMA 发送水线寄存器。
0x0024	<code>SPI_RSDR</code>	RXD 采样延迟寄存器。
0x0028	<code>SPI_TDER</code>	TXD 驱动边沿寄存器。
0x002C	<code>SPI_DRNM</code>	SPI 数据寄存器。
0x00BC	<code>SPI_RAINSR</code>	原始中断状态寄存器。
0x00C0	<code>SPI_INSR</code>	中断状态寄存器。
0x00C4	<code>SPI_INMAR</code>	中断屏蔽寄存器。
0x00C8	<code>SPI_SLENR</code>	Slave 使能寄存器。

偏移地址	名称	描述
0x00CC	SPI_TWLR	发送水线寄存器。
0x00D0	SPI_TLR	发送数据量寄存器。
0x00D4	SPI_TFOFIER	tx fifo 中断清除寄存器。
0x00D8	SPI_RWLR	接收水线寄存器。
0x00DC	SPI_RLR	接收数据量寄存器。
0x00E0	SPI_RFFICR	rx fifo Underflow 中断清除寄存器。
0x00E4	SPI_WSR	状态寄存器。
0x00E8	SPI_MSTICR	多 master 中断清除寄存器。
0x00EC	SPI_ID	版本信息寄存器。
0x00F0	SPI_RSVD	预留寄存器。
0x00F4	SPI_RXOICR	rx fifo Overflow 中断清除寄存器。
0x00F8	SPI_ICR	中断清除寄存器。
0x00FC	SPI_APB_CG	APB 时钟门控控制寄存器。

6.5.5 寄存器描述

SPI_ER

SPI_ER 为 SPI 使能寄存器。

Offset Address: 0x0000 Total Reset Value: 0x0000_0000

Bits	Access	Name	Description	Reset
[31:1]	-	reserved	保留。	0x00000000
[0]	RW	start_en	SPI 使能。 使能或禁用全部 SPI 操作。禁用时，所有串行传输立即挂起，tx fifo 和 rx fifo 缓存被清空。使能	0x0

Bits	Access	Name	Description	Reset
			时，某些 SPI 控制寄存器内容不可更改。禁用时，ssi_sleep 输出被置位（延迟后），通知系统此时可安全地停止 SPI 时钟，以节省功耗。 0：禁止串行传输； 1：使能串行传输。	

SPI_CTRA

SPI_CTRA 为 SPI 控制寄存器 A。

Offset Address: 0x0004 Total Reset Value: 0x0000_E004

Bits	Access	Name	Description	Reset
[31:24]	-	reserved	保留。	0x00
[23:22]	RW	prs	帧格式。选择传输数据所使用的串行协议： 00 (MOTOROLA_SPI)：摩托罗拉协议； 01 (TEXAS_SSP)：TI 协议； 10 (NS_MICROWIRE)：National Microwire 协议； 11 (RESERVED)：保留。	0x0
[21:20]	RO	enhff	SPI 帧格式。选择发送与接收数据的数据帧格式，该寄存器仅在 SSI_SPI_MODE 设为 "Dual/Quad/Octal" 之一时才有效。 当 SSI_SPI_MODE 设为 "Dual"，10 与 11 组合无效。	0x0

Bits	Access	Name	Description	Reset
			当 SSI_SPI_MODE 设为 "Quad", 11 组合无效。 2'b00 (STD_ENHFF) : 标准 SPI 帧格式; 2'b01 (DUAL_ENHFF) : 双线 SPI 帧格式; 2'b10 (QUAD_ENHFF) : 4 线 SPI 帧格式; 2'b11 (OCTAL_ENHFF) : 8 线 SPI 帧格式。 注意: 本芯片 SPI 为单线, 故本寄存器恒为 2'b00。	
[19:18]	RW	trsm	传输模式。 选择串行通信传输模式。该寄存器不影响传输本身, 只用于指示接收或发送数据是否有效。 在 transmit-only 模式, SPI 接收到的外部器件发送的数据无效, 数据不会存储在 rx fifo 内, 且会被下个传输所覆盖。 在 receive-only 模式, 发送的数据无效。第一次写入 tx fifo 后, 相同的数据会在传输传输持续世间内重复发送。 在 transmit&receive 模式, 发送与接收数据均有效。传输会持续到 tx fifo 为空。接收到了外部器件数据被存储于可被 host 处理器读取的 rx fifo 内。 在 eeprom-read 模式, 当发送控制数据时, 接收数据无效。当全部控制数据发送至 EEPROM 后, 接收	0x0

Bits	Access	Name	Description	Reset
			数据开始有效而发送数据变为无效。该模式下，tx fifo 内全部数据均被视为控制数据。该模式仅在 SPI 被配置为主器件时才有效。 00：收发模式（Transmit & Receive）； 01：只发模式（Transmit Only）； 10：只收模式（Receive Only）； 11：EEPROM 读取模式（EEPROM Read）。	
[17:13]	RW	dfs32	32bit 传输模式的数据帧长度。该寄存器仅在 SSI_MAX_XFER_SIZE 配置为 32 时生效。 当数据帧长度配置为小于 32bit，接收逻辑自动将数据向右调整，rx fifo 的高位 bit 被零填充。设计者需保证发送数据写入 tx fifo 前同样经过了高位零填充处理。发送逻辑在发送数据时会忽略未使用的高位 bits。 0x3（FRAME_04BITS）：4bit 串行数据传输； 0x4（FRAME_05BITS）：5bit 串行数据传输； 0x5（FRAME_06BITS）：6bit 串行数据传输； 0x6（FRAME_07BITS）：7bit 串行数据传输； 0x7（FRAME_08BITS）：8bit 串行数据传输；	0x07

Bits	Access	Name	Description	Reset
			0x8 (FRAME_09BITS) : 9bit 串行数据传输; 0x9 (FRAME_10BITS) : 10bit 串行数据传输; 0xa (FRAME_11BITS) : 11bit 串行数据传输; 0xb (FRAME_12BITS) : 12bit 串行数据传输; 0xc (FRAME_13BITS) : 13bit 串行数据传输; 0xd (FRAME_14BITS) : 14bit 串行数据传输; 0xe (FRAME_15BITS) : 15bit 串行数据传输; 0xf (FRAME_16BITS) : 16bit 串行数据传输; 0x10 (FRAME_17BITS) : 17bit 串行数据传输; 0x11 (FRAME_18BITS) : 18bit 串行数据传输; 0x12 (FRAME_19BITS) : 19bit 串行数据传输; 0x13 (FRAME_20BITS) : 20bit 串行数据传输; 0x14 (FRAME_21BITS) : 21bit 串行数据传输; 0x15 (FRAME_22BITS) : 22bit 串行数据传输; 0x16 (FRAME_23BITS) : 23bit 串行数据传输; 0x17 (FRAME_24BITS) : 24bit	

Bits	Access	Name	Description	Reset
			串行数据传输; 0x18 (FRAME_25BITS) : 25bit 串行数据传输; 0x19 (FRAME_26BITS) : 26bit 串行数据传输; 0x1a (FRAME_27BITS) : 27bit 串行数据传输; 0x1b (FRAME_28BITS) : 28bit 串行数据传输; 0x1c (FRAME_29BITS) : 29bit 串行数据传输; 0x1d (FRAME_30BITS) : 30bit 串行数据传输; 0x1e (FRAME_31BITS) : 31bit 串行数据传输; 0x1f (FRAME_32BITS) : 32bit 串行数据传输。	
[12:9]	RO	dfs16	数据帧长度。该寄存器仅在 SSI_MAX_XFER_SIZE 配置为 16 时生效。若 SSI_MAX_XFER_SIZE 配置为 32, 写入该寄存器无任何影响。 选择数据帧长度, 当数据帧长度配 置为小于 16bit, 接收逻辑自动将 数据向右调整, rx fifo 的高位 bit 被 零填充。设计者需保证发送数据写 入 tx fifo 前同样经过了高位零填充 处理。发送逻辑在发送数据时会忽 略未使用的高位。 0x3 (FRAME_04BITS) : 4bit 串 行数据传输; 0x4 (FRAME_05BITS) : 5bit 串	0x0

Bits	Access	Name	Description	Reset
			行数据传输； 0x5 (FRAME_06BITS) : 6bit 串行数据传输； 0x6 (FRAME_07BITS) : 7bit 串行数据传输； 0x7 (FRAME_08BITS) : 8bit 串行数据传输； 0x8 (FRAME_09BITS) : 9bit 串行数据传输； 0x9 (FRAME_10BITS) : 10bit 串行数据传输； 0xa (FRAME_11BITS) : 11bit 串行数据传输； 0xb (FRAME_12BITS) : 12bit 串行数据传输； 0xc (FRAME_13BITS) : 13bit 串行数据传输； 0xd (FRAME_14BITS) : 14bit 串行数据传输； 0xe (FRAME_15BITS) : 15bit 串行数据传输； 0xf (FRAME_16BITS) : 16bit 串行数据传输。 注意：本芯片 SSI_MAX_XFER_SIZE 配置为 32，故此寄存器无效。	
[8:5]	RW	cfs16	控制帧长度。选择 Microwire 协议格式的控制字长度。 0x0 (SIZE_01_BIT) : 1bit 控制帧长度； 0x1 (SIZE_02_BIT) : 2bit 控制	0x0

Bits	Access	Name	Description	Reset
			帧长度; 0x2 (SIZE_03_BIT) : 3bit 控制 帧长度; 0x3 (SIZE_04_BIT) : 4bit 控制 帧长度; 0x4 (SIZE_05_BIT) : 5bit 控制 帧长度; 0x5 (SIZE_06_BIT) : 6bit 控制 帧长度; 0x6 (SIZE_07_BIT) : 7bit 控制 帧长度; 0x7 (SIZE_08_BIT) : 8bit 控制 帧长度; 0x8 (SIZE_09_BIT) : 9bit 控制 帧长度; 0x9 (SIZE_10_BIT) : 10bit 控制 帧长度; 0xa (SIZE_11_BIT) : 11bit 控制 帧长度; 0xb (SIZE_12_BIT) : 12bit 控制 帧长度; 0xc (SIZE_13_BIT) : 13bit 控制 帧长度; 0xd (SIZE_14_BIT) : 14bit 控制 帧长度; 0xe (SIZE_15_BIT) : 15bit 控制 帧长度; 0xf (SIZE_16_BIT) : 16bit 控制 帧长度。	
[4]	RW	scpol	选择时钟极性。 仅当帧格式 (SPI_CTRA[prs]) 设	0x0

Bits	Access	Name	Description	Reset
			为摩托罗拉协议时有效。用于选择非激活状态下串行时钟的极性（当 SPI master 未传输数据时）。 0 (SCLK_LOW)：非激活状态时钟为低； 1 (SCLK_HIGH)：非激活状态时钟为高。	
[3]	RW	scph	串行时钟相位。 仅当帧格式 (SPI_CTRA[prs]) 设为摩托罗拉协议时有效。该寄存器控制串行时钟与 slave 选择信号之间的关系。 当 SCPH = 0, 数据在串行时钟第一个边沿被采样, 当 SCPH=1, 串行时钟在 slave 选择信号激活后开始翻转, 数据在第二个时钟边沿被采样。 0 (SCPH_MIDDLE)：串行时钟在第一个数据位中间翻转； 1 (SCPH_START)：串行时钟在第一个数据位的起始位置翻转。	0x0
[2]	RW	ssn_te	Slave 选择信号翻转使能。 当时钟相位 (SPI_CTRA[scph]) 设为 0, 该寄存器控制 slave 选择信号 (ss*_n) 数据传输间隙的表现。若该 1: ss*_n 信号线将在连续的数据帧间翻转且串行时钟 (sclk) 会在 ss*_n 为高时保持默认值； 0: ss*_n 信号将在数据传输期间保持为低, sclk 信号也将持续翻转。	0x1

Bits	Access	Name	Description	Reset
			注意：该寄存器仅在 SSI_SCPH0_SSTOGGLE 设为 1 时生效。	
[1]	RW	srlt	移位寄存器环。仅用于测试。若使能，发送移位寄存器的输出将被连接至接收以为寄存器的输入。当 SPI 配置为从器件，回环模式所使用的 ss_in_n 与 ssi_clk 必须由外部器件提供。 0 (NORMAL_MODE)：常规模式。 1 (TESTING_MODE)：测试模式，Tx & Rx 移位寄存器互联；	0x0
[0]	RW	soe	Slave 输出使能。仅在 SPI 配置为从器件有效。当配置为主器件，该寄存器无任何功能。该寄存器使能或禁止 slave 器件输出的 ssi_oe_n 信号的设置。该功能在主器件以广播模式发送数据时非常有效，仅有一个从器件会在主器件的 rxd 数据线上回送数据。 0 (ENABLED)：Slave 输出使能； 1 (DISABLED)：禁止 Slave 输出，ssi_oe_n 不会被激活，当 ssi_oe_n 控制 slave 器件 txd 信号的三态 buffer 时，txd 输出将呈现高阻态。	0x0

SPI_CTRB

SPI_CTRB 为 SPI 控制寄存器 B。

Offset Address: 0x0008 Total Reset Value: 0x0000_0000

Bits	Access	Name	Description	Reset
[31:16]	-	reserved	保留。	0x0000
[15:0]	RW	nrdf	数据帧数量。 当 SPI_CTRA[trsm]==2'b10 或 2'b11 时，该寄存器设置 SPI 连续接收的数据帧数量。SPI 持续接收数据直到数据帧个数等于该寄存器加 1，最大允许 64KB 的连续数据接收。 注意：当 SPI 配置为从器件，传输将在 slave 被选中时一直进行，此时本寄存器无任何作用。	0x0000

SPI_ENHCTL

SPI_ENHCTL 为 SPI Enhanced 控制寄存器。

说明

此寄存器值恒为 0。

Offset Address: 0x000C Total Reset Value: 0x0000_0000

Bits	Access	Name	Description	Reset
[31:16]	-	reserved	保留。	0x0000
[15]	RW	ddren	SPI DDR 使能位。 0：使能； 1：关闭。 该寄存器允许 SPI 在 Dual/Quad/Octal 帧格式时使用 DDR 数据模式。	0x0
[14]	RW	iddren	指令阶段 DDR 使能位。	0x0

Bits	Access	Name	Description	Reset
			0: 使能; 1: 关闭。 该寄存器允许 SPI 传输的指令阶段使用 DDR 数据模式。	
[13:12]	RW	ilen	指令长度。Dual/Quad/Octal 模式下指令 bit 位宽。 00 (INST_L_0) : 0bit 无指令; 01 (INST_L_1) : 4bit 指令; 10 (INST_L_2) : 8bit 指令; 11 (INST_L_3) : 16bit 指令。 注意: 本芯片 SPI 为单线模式, 故此寄存器值恒为 0。	0x0
[11:8]	RW	addrlen	地址长度。该寄存器定义传输地址长度。只有当该长度的 bit 位发送至 FIFO 后传输才会开始。 0x0 (ADDR_L_0) : 0bit 地址宽度; 0x1 (ADDR_L_1) : 4bit 地址宽度; 0x2 (ADDR_L_2) : 8bit 地址宽度; 0x3 (ADDR_L_3) : 12bit 地址宽度; 0x4 (ADDR_L_4) : 16bit 地址宽度; 0x5 (ADDR_L_5) : 20bit 地址宽度; 0x6 (ADDR_L_6) : 24bit 地址宽度; 0x7 (ADDR_L_7) : 28bit 地址宽度;	0x0

Bits	Access	Name	Description	Reset
			度; 0x8 (ADDR_L_8) : 32bit 地址宽度; 0x9 (ADDR_L_9) : 36bit 地址宽度; 0xa (ADDR_L_10) : 40bit 地址宽度; 0xb (ADDR_L_11) : 44bit 地址宽度; 0xc (ADDR_L_12) : 48bit 地址宽度; 0xd (ADDR_L_13) : 52bit 地址宽度; 0xe (ADDR_L_14) : 56bit 地址宽度; 0xf (ADDR_L_15) : 60bit 地址宽度。	
[7]	RW	rdsen	读数据选通使能位。一旦该 bit 位设为 1, SPI 将使用读数据选通 (rxds) 采样 DDR 模式的读数据。 0: 使能; 1: 关闭。	0x0
[6:5]	RW	aaitf	地址和指令传输模式。选择 SPI 发送指令及地址数据的模式。 00: 以标志 SPI 模式发送指令及地址; 01: 指令以标志 SPI 模式发送, 地址以 CTRLRO.ENHFF 定义的模式发送; 10: 指令及地址均以 ENHFF 定义的模式发送;	0x0

Bits	Access	Name	Description	Reset
			11: 保留。	
[4:0]	RW	waitnum	等待周期。在 Dual/Quad/Octal 模式下，定义控制帧发送与数据接收之间的等待时间。寄存器的值以 SPI 时钟周期为单位。	0x00

SPI_MCR

SPI_MCR 为 microwire 控制寄存器。

Offset Address: 0x0010 Total Reset Value: 0x0000_0000

Bits	Access	Name	Description	Reset
[31:3]	-	reserved	保留。	0x00000000
[2]	RW	mtrc	Microwire 控制，定义 Microwire 串行协议的数据传输方向。 0: 接收数据，SPI MacroCell 从外部串行器件接收数据； 1: 发送数据，SPI MacroCell 向外部串行器件发送数据。	0x0
[1]	RW	mss	Microwire 传输模式。定义 Microwire 传输是否是连续的。 0: 非连续 Microwire 传输，发送或接收每个数均需要 1 个控制字； 1: 连续 Microwire 传输，一个数据块发送或接收仅需要 1 个控制字。	0x0
[0]	RW	mh	Microwire 握手使能。仅在 SPI 被配置为 Master 器件时有效，当被配置为 Slave 器件时该寄存器无意义。用于使能或禁止 Microwire 协	0x0

Bits	Access	Name	Description	Reset
			议的 busy/ready 握手接口。当使能时，SPI 在最后 1bit 数据传输完成之后，SR 寄存器的 Busy 状态清除之前，检查目标 Slave 器件的 ready 状态。 0：握手接口禁用； 1：握手接口使能。	

SPI_BRS

SPI_BRS 为波特率控制寄存器。

Offset Address: 0x0014 Total Reset Value: 0x0000_0000

Bits	Access	Name	Description	Reset
[31:16]	-	reserved	保留。	0x0000
[15:0]	RW	frdv	时钟分频器。若该寄存器设为 0，SPI 时钟输出将被禁止。SPI 时钟频率按如下公式计算： $F_{sclk_out} = F_{ssi_clk} / FRDV$ ，其中 FRDV 是 2 到 65534 之间的任意偶数。例如： $F_{ssi_clk} = 3.6864\text{MHz}$ 、 $FRDV = 2$ $F_{sclk_out} = 3.6864 / 2 = 1.8432\text{MHz}$ 注意：该寄存器最低 bit 位被强制设为 0，不受寄存器写入的影响，这保证该寄存器的值永远是偶数。	0x0000

SPI_DCR

SPI_DCR 为 DMA 控制寄存器。

Offset Address: 0x0018 Total Reset Value: 0x0000_0000

Bits	Access	Name	Description	Reset
[31:2]	-	reserved	保留。	0x00000000
[1]	RW	rden	接收 DMA 使能。 0: 接收 DMA 禁用; 1: 接收 DMA 使能。	0x0
[0]	RW	tden	发送 DMA 使能。 0: 发送 DMA 禁用; 1: 接收 DMA 使能。	0x0

SPI_DRDL

SPI_DRDL 为 DMA 接收流水线寄存器。

Offset Address: 0x001C Total Reset Value: 0x0000_0000

Bits	Access	Name	Description	Reset
[31:6]	-	reserved	保留。	0x00000000
[5:0]	RW	rdl	接收流水线。该寄存器控制接收逻辑在何时发起 DMA 请求。 dma_rx_req 将在 rx fifo 内数据量大于等于该寄存器加 1 时产生（且要求 SPI_DCR[rden]=1）。	0x00

SPI_DTDL

SPI_DTDL 为 DMA 发送流水线寄存器。

Offset Address: 0x0020 Total Reset Value: 0x0000_0000

Bits	Access	Name	Description	Reset
------	--------	------	-------------	-------

Bits	Access	Name	Description	Reset
[31:6]	-	reserved	保留。	0x0000000
[5:0]	RW	tdl	发送水线。该寄存器控制发送逻辑在何时发起 DMA 请求。 dma_tx_req 将在 tx fifo 内有效数据量等于或小于该值时产生（且要求 SPI_DCR[tden]=1）。	0x00

SPI_RSDR

SPI_RSDR 为 RXD 采样延迟寄存器。

Offset Address: 0x0024 Total Reset Value: 0x0000_0000

Bits	Access	Name	Description	Reset
[31:8]	-	reserved	保留。	0x0000000
[7:0]	RW	rxdsd	Rxd 采样延迟。该寄存器用于延迟 rxd 输入端口的采样时间，延迟单位为 ssi_clk 周期。 注意：若该寄存器所配置的值超出内部移位寄存器深度 (SSI_RX_DLY_SR_DEPTH=8)，零延迟采样将作用在 rxd 端口。	0x00

SPI_TDER

SPI_TDER 为 TXD 驱动边沿寄存器。

Offset Address: 0x0028 Total Reset Value: 0x0000_0000

Bits	Access	Name	Description	Reset
[31:8]	-	reserved	保留。	0x0000000
[7:0]	RO	txdde	TXD 驱动沿。该值决定了发送数据	0x00

Bits	Access	Name	Description	Reset
			的驱动沿。 注意：本芯片 SPI 的该寄存器值恒为 0。	

SPI_DRNM

SPI_DRNM 为 SPI 数据寄存器。

Offset Address: 0x002C Total Reset Value: 0x0000_0000

Bits	Access	Name	Description	Reset
[31:0]	RW	datar	写入/读出数据。写入该寄存器必须将数据右对齐，读取的数据已被自动右对齐。 读操作：读出 rx fifo 缓存的数据； 写操作：数据存入 tx fifo。	0x00000000

SPI_RAINSR

SPI_RAINSR 为原始中断状态寄存器。

Offset Address: 0x00BC Total Reset Value: 0x0000_0000

Bits	Access	Name	Description	Reset
[31:16]	-	reserved	保留。	0x0000
[15]	RO	mmcris	多 Master 连接原始中断状态。该寄存器在 SPI 被配置为 Slave 器件时不可见： 0：中断未产生 (SPI_INMAR[mmcim]作用前)； 1：中断产生 (SPI_INMAR[mmcim]作用前)。	0x0

Bits	Access	Name	Description	Reset
[14:9]	RO	reserved	保留。	0x00
[8]	RO	tfofris	tx fifo 溢出原始中断状态： 0：中断未产生 (SPI_INMAR[tfofim]作用前)； 1：中断产生 (SPI_INMAR[tfofim] 作用前)。	0x0
[7]	RO	tferis	tx fifo 为空原始中断： 0：中断未产生 (SPI_INMAR[tfeim]作用前)； 1：中断产生 (SPI_INMAR[tfeim] 作用前)。	0x0
[6:3]	RO	reserved	保留。	0x0
[2]	RO	rfufri	rx fifo 下溢原始中断状态： 0：中断未产生 (SPI_INMAR[rfufim]作用前)； 1：中断产生 (SPI_INMAR[rfufim] 作用前)。	0x0
[1]	RO	rfofri	rx fifo 溢出原始中断状态： 0：中断未产生 (SPI_INMAR[rfofim]作用前)； 1：中断产生 (SPI_INMAR[rfofim]作用前)。	0x0
[0]	RO	rffri	rx fifo 为满原始中断状态： 0：中断未产生 (SPI_INMAR[rffim]作用前)； 1：中断产生 (SPI_INMAR[rffim] 作用前)。	0x0

SPI_INSR

SPI_INSR 为中断状态寄存器。

Offset Address: 0x00C0 Total Reset Value: 0x0000_0000

Bits	Access	Name	Description	Reset
[31:16]	-	reserved	保留。	0x0000
[15]	RO	mmcis	多 Master 连接中断状态。该寄存器在 SPI 被配置为 Slave 器件时不可见： 0：中断未产生 (SPI_INMAR[mmcim]作用后)； 1：中断产生 (SPI_INMAR[mmcim]作用后)。	0x0
[14:9]	RO	reserved	保留。	0x00
[8]	RO	tfofis	tx fifo 溢出中断状态： 0：中断未产生 (SPI_INMAR[tfofim]作用后)； 1：中断产生 (SPI_INMAR[tfofim]作用后)。	0x0
[7]	RO	tfeis	tx fifo 为空中断： 0：中断未产生 (SPI_INMAR[tfeim]作用后)； 1：中断产生 (SPI_INMAR[tfeim]作用后)。	0x0
[6:3]	RO	reserved	保留。	0x0
[2]	RO	rfufis	rx fifo 下溢中断状态： 0：中断未产生 (SPI_INMAR[rfufim]作用后)； 1：中断产生 (SPI_INMAR[rfufim]作用后)。	0x0

Bits	Access	Name	Description	Reset
[1]	RO	rfofis	rx fifo 溢出中断状态： 0：中断未产生 (SPI_INMAR[rfofim]作用后)； 1：中断产生 (SPI_INMAR[rfofim] 作用后)。	0x0
[0]	RO	ffis	rx fifo 为满中断状态： 0：中断未产生 (SPI_INMAR[rffim]作用后)； 1：中断产生 (SPI_INMAR[rffim] 作用后)。	0x0

SPI_INMAR

SPI_INMAR 为中断屏蔽寄存器。

Offset Address: 0x00C4 Total Reset Value: 0x0000_8187

Bits	Access	Name	Description	Reset
[31:16]	-	reserved	保留。	0x0000
[15]	RW	mmcim	多 Master 连接中断屏蔽。该寄存器在 SPI 被配置为 Slave 器件时不可见。 0：中断屏蔽； 1：中断未屏蔽。	0x1
[14:9]	RO	reserved	保留。	0x00
[8]	RW	tfofim	tx fifo 溢出中断屏蔽： 0：中断屏蔽； 1：中断未屏蔽。	0x1
[7]	RW	tfeim	tx fifo 为空中断屏蔽：	0x1

Bits	Access	Name	Description	Reset
			0：中断屏蔽； 1：中断未屏蔽。	
[6:3]	RO	reserved	保留。	0x0
[2]	RW	rfufim	rx fifo 下溢原始中断屏蔽： 0：中断屏蔽； 1：中断未屏蔽。	0x1
[1]	RW	rfofim	rx fifo 溢出中断屏蔽： 0：中断屏蔽； 1：中断未屏蔽。	0x1
[0]	RW	rffim	rx fifo 为满中断屏蔽： 0：中断屏蔽； 1：中断未屏蔽。	0x1

SPI_SLENR

SPI_SLENR 为 Slave 使能寄存器。

Offset Address: 0x00C8 Total Reset Value: 0x0000_0000

Bits	Access	Name	Description	Reset
[31:2]	-	reserved	保留。	0x00000000
[1:0]	RW	ssef	Slave 选择使能标志。该寄存器的每一 bit 代表 1 个 Slave 选择信号 (ss_x_n)。若该寄存器某 bit 位设为 1，相应的 Slave 选择信号被激活。启动一个传输前，用户需要使能相应 Slave 器件所对应的 bit 位。在非广播模式，该寄存器中仅应有 1bit 被置位。 0：无从器件选择；	0x0

Bits	Access	Name	Description	Reset
			1: 从器件被选择。 注意：置位或清零该寄存器并不影响相应的 Slave 选择输出信号，直到一个传输开始。	

SPI_TWLR

SPI_TWLR 为发送水线寄存器。

Offset Address: 0x00CC Total Reset Value: 0x0000_0000

Bits	Access	Name	Description	Reset
[31:6]	-	reserved	保留。	0x00000000
[5:0]	RW	twl	tx fifo 水线配置寄存器。控制 tx fifo 产生中断的水线，FIFO 深度被配置为 2-64。当 tx fifo 内数据量小于等于该寄存器值，tx fifo 空中断将被触发。 注意：若用户向该寄存器写入的值大于或等于 fifo 深度（64），则该寄存器将不可写入并保持当前值。	0x00

SPI_TLR

SPI_TLR 为发送数据量寄存器。

Offset Address: 0x00D0 Total Reset Value: 0x0000_0000

Bits	Access	Name	Description	Reset
[31:7]	-	reserved	保留。	0x00000000
[6:0]	RO	tfl	tx fifo 内有效数据个数指示寄存器。	0x00

SPI_TFOFIER

SPI_TFOFIER 为 tx fifo 中断清除寄存器。

Offset Address: 0x00D4 Total Reset Value: 0x0000_0000

Bits	Access	Name	Description	Reset
[31:1]	-	reserved	保留。	0x00000000
[0]	RC	ctfofi	清除 tx fifo 溢出中断寄存器。该寄存器反映中断状态，读该寄存器将清除 ssi_txo_intr 中断，写该寄存器无效。 0: tx fifo 溢出中断无效； 1: tx fifo 溢出中断有效。	0x0

SPI_RWLR

SPI_RWLR 为接收水线寄存器。

Offset Address: 0x00D8 Total Reset Value: 0x0000_0000

Bits	Access	Name	Description	Reset
[31:6]	-	reserved	保留。	0x00000000
[5:0]	RW	rwl	rx fifo 水线配置寄存器。控制 rx fifo 产生中断的水线位置。当 rx fifo 内数据量大于或等于该寄存器值加 1，rx fifo 的满中断将被触发。 注意：若用户向该寄存器写入的值大于或等于 FIFO 深度，则该寄存器将不可写入并保持当前值。	0x00

SPI_RLR

SPI_RLR 为接收数据量寄存器。

Offset Address: 0x00DC Total Reset Value: 0x0000_0000

Bits	Access	Name	Description	Reset
[31:7]	-	reserved	保留。	0x0000000
[6:0]	RO	rfl	rx fifo 内有效数据量指示寄存器。	0x00

SPI_RFFICR

SPI_RFFICR 为 rx fifo Underflow 中断清除寄存器。

Offset Address: 0x00E0 Total Reset Value: 0x0000_0000

Bits	Access	Name	Description	Reset
[31:1]	-	reserved	保留。	0x00000000
[0]	RC	crfufi	清除 rx fifo 下溢中断。该寄存器反映中断状态，读该寄存器将清除 ssi_rxu_intr 中断，写该寄存器无效。 0: rx fifo 下溢中断无效; 1: rx fifo 下溢中断有效。	0x0

SPI_WSR

SPI_WSR 为状态寄存器。

Offset Address: 0x00E4 Total Reset Value: 0x0000_1800

Bits	Access	Name	Description	Reset
[31:16]	-	reserved	保留。	0x0000

Bits	Access	Name	Description	Reset
[15]	RO	sbfb	SSI 忙标志。 0: SPI 空闲或禁用; 1: SPI 正在传输数据。	0x0
[14:13]	RO	reserved	保留。	0x0
[12]	RO	tfe	tx fifo 空标志。当 tx fifo 彻底为空时, 该 bit 位置位。当 tx fifo 内存在一个或更多的有效数据, 该 bit 位清零。该 bit 位并不产生中断。 0: tx fifo 非空; 1: tx fifo 为空。	0x1
[11]	RO	tfnf	tx fifo 非满标志。当 tx fifo 内存在一个或多个空存储空间时, 该 bit 位置位; 当 FIFO 为满时该 bit 位清零。 0: tx fifo 为满; 1: tx fifo 非满。	0x1
[10:6]	RO	reserved	保留。	0x00
[5]	RO	rffe	rx fifo 满标志。当 rx fifo 彻底满时, 该 bit 位置位。当 rx fifo 仍有 1 个或更多的存储空间该 bit 位清零。 0: rx fifo 非满; 1: rx fifo 满。	0x0
[4]	RO	rfne	rx fifo 非空标志。当 rx fifo 内存在 1 个或更多有效数据, 该 bit 位置位; 当 rx fifo 为空时, 该 bit 位清零。该 bit 位可被软件轮询来确定 rx fifo 彻底变空。 0: rx fifo 空;	0x0

Bits	Access	Name	Description	Reset
			1: rx fifo 非空。	
[3:1]	RO	reserved	保留。	0x0
[0]	RO	dcfe_tfee	MASTER 模式下: 数据冲突错误。 当 SPI master 在执行数据传输, ss_in_n 信号被其他 master 驱动时, 该 bit 位置位, 通知处理器最后一次数据传输在完成前被挂起。该 bit 位在读取后自动清零。 0: 无数据冲突; 1: 数据冲突错误产生。 SLAVE 模式下: 发送错误。 当 tx fifo 为空时, 若一个传输开始, 该 bit 位置位。该 bit 位仅在 SPI 配置为 Slave 器件时才可置位。上次传输的数据将在 txd 数据线上重复发送。该 bit 位在读取时自动清零。 0: 无错误; 1: 发送错误产生。	0x0

SPI_MSTICR

SPI_MSTICR 为多 master 中断清除寄存器。

Offset Address: 0x00E8 Total Reset Value: 0x0000_0000

Bits	Access	Name	Description	Reset
[31:1]	-	reserved	保留。	0x00000000
[0]	RC	cmmci	清除多 Master 连接中断。该寄存器反应中断状态, 读该寄存器将清	0x0

Bits	Access	Name	Description	Reset
			除 ssi_mst_intr 中断，写该寄存器无任何影响。 1：多 Master 连接中断有效； 0：无中断。	

SPI_ID

SPI_ID 为版本信息寄存器。

Offset Address: 0x00EC Total Reset Value: 0x2021_0228

Bits	Access	Name	Description	Reset
[31:0]	RO	id	版本信息寄存器。存储十六进制格式的 IP 版本信息。	0x20210228

SPI_RSVD

SPI_RSVD 为预留寄存器。

Offset Address: 0x00F0 Total Reset Value: 0x0000_0000

Bits	Access	Name	Description	Reset
[31:11]	-	reserved	保留。	0x000000
[10]	RW	wait_bypass_en	等待状态旁路信号。 0：不等待； 1：等待。 注意：仅在 master 模式下有效。	0x0
[9:0]	RW	wait_number	指令发送后等待接收数据的时间。 单位为 SPI 工作时钟 cycles。 注意：仅在 master 模式下有效。	0x000

SPI_RXOICR

SPI_RXOICR 为 rx fifo Overflow 中断清除寄存器。

Offset Address: 0x00F4 Total Reset Value: 0x0000_0000

Bits	Access	Name	Description	Reset
[31:1]	-	reserved	保留。	0x00000000
[0]	RC	crfofi	清除 rx fifo 溢出中断。 该寄存器反应中断状态，读该寄存器清除 ssi_rxo_intr 中断，写该寄存器无影响。	0x0

SPI_ICR

SPI_ICR 为中断清除寄存器。

Offset Address: 0x00F8 Total Reset Value: 0x0000_0000

Bits	Access	Name	Description	Reset
[31:1]	-	reserved	保留。	0x00000000
[0]	RC	cis	清除中断。该寄存器在 ssi_txo_intr, ssi_rxu_intr, ssi_rxo_intr 与 the ssi_mst_intr 四个中断中任意一个有效时置位。读该寄存器将清除这四个中断。写该寄存器无影响。 0: tx fifo 溢出中断,rx fifo 下溢中断, rx fifo 溢出中断, 多 master 中断均无效; 1: tx fifo 溢出中断,rx fifo 下溢中断, rx fifo 溢出中断, 多 master 中断, 任意一个生效。	0x0

SPI_APB_CG

SPI_APB_CG 为 APB 时钟门控控制寄存器。

Offset Address: 0x00FC Total Reset Value: 0x0000_0000

Bits	Access	Name	Description	Reset
[31:16]	-	reserved	保留。	0x0000
[15:0]	RW	spi_apb_cg	总线时钟自动 CG 开关寄存器。控制 APB 总线强制门控信号，该寄存器用作关闭 APB 总线时钟自动门控。 0xdead: 关闭自动 CG，APB 总线时钟常开； 0xbeef: 恢复自动 CG； others: 不影响自动 CG 状态。	0x0000

6.6 PWM

6.6.1 概述

PWM 模块用于生成 PWM 信号。

6.6.2 功能描述

PWM 模式具有以下功能特点：

- 支持 8 路 PWM。
- 支持将 PWM 分组绑定，实现互补输出，最多支持 4 组分组。
- 支持 PWM 输出恒为 0、1 或高阻。
- PWM 占空比支持 0~100%（256 档）。
- 支持 PWM 不同配置间的平滑切换。

6.6.3 工作方式

PWM0~7 模块支持 2 个时钟源：晶振时钟；PLL 时钟。

以配置 PWM0 信号对 80M 时钟分频、分频倍数为 25 倍、占空比为 1/5 为例，配置步骤如下：

- 步骤 1 写 CLDO_CRG_CLK_SEL[pwm_cksel]为 0x1，将 PWM 时钟置为 80M。
- 步骤 2 参考 IO_MUX，配置好 PWM 对应的 IO 复用以及 IO 驱动能力。
- 步骤 3 写 PWM_ABNOR_STATE_CLR0[pwm_abnor_state_clr0]为 0xFFFF，写 PWM_ABNOR_STATE_CLR1[pwm_abnor_state_clr1]为 0xFFFF，清除异常状态。
- 步骤 4 写 PWM_EN[pwm_en_0]为 1，使能 PWM0。
- 步骤 5 写 PWM_FREQ_L[pwm_freq_l_0]为 0x19，配置分频倍数为 25。
- 步骤 6 写 PWM_DUTY_L[pwm_duty_l_0]为 0x5，配出占空比 1/5。
- 步骤 7 写 PWM_PORTITY[pwm_poarity_0]为 0，PWM 输出极性为正向极性。
- 步骤 8 写 PWM_SEL[pwm_sel_0]为 0x1，配置第一组使用的 PWM0。
- 步骤 9 写 PWM_STARTCLRCNT_EN[pwm_startclrcnt_en_0]为 1，在 start 时刻对 PWM 内部寄存器载入配置并清零计数器。
- 步骤 10 写 PWM_START[pwm_start_0]为 1，配置 PWM0 新配置生效。

----结束

6.6.4 寄存器概览

PWM 寄存器概览如表 6-13 所示。

表6-13 PWM 寄存器概览（基址是 0x4402_4000）

偏移地址	名称	描述
0x000 + 0x10×i	PWM_SEL	pwm 分组选择控制寄存器。
0x004 + 0x10×i	PWM_STARTCLRCNT_EN	pwm 配置生效计数清零使能寄存器。

偏移地址	名称	描述
0x008 + 0x10×i	PWM_START	PWM 配置生效寄存器。
0x0100 + 0x40×j	PWM_EN	PWM 使能寄存器。
0x0104 + 0x40×j	PWM_PORTITY	PWM 的输出极性配置寄存器。
0x0108 + 0x40×j	PWM_OEN_CFG	Pwm 无效电平高阻使能配置寄存器。
0x010C + 0x40×j	PWM_OFFSET_L	PWM_START 为 PWM 相位控制计数值低 16bit 寄存器。
0x0110 + 0x40×j	PWM_OFFSET_H	PWM_START 为 PWM 相位控制计数值高 16bit 寄存器。
0x0114 + 0x40×j	PWM_FREQ_L	PWM_FREQ 为 PWM 频率控制计数值低 16bit 寄存器。
0x0118 + 0x40×j	PWM_FREQ_H	PWM_FREQ 为 PWM 频率控制计数值高 16bit 寄存器。
0x011C + 0x40×j	PWM_DUTY_L	PWM_DUTY 为 PWM 占空比计数值寄存器。
0x0120 + 0x40×j	PWM_DUTY_H	PWM_DUTY 为 PWM 占空比计数值寄存器。
0x0124 + 0x40×j	PWM_PERIODLOAD_FLAG	PWM 周期配置允许标志寄存器。
0x0128 + 0x40×j	PWM_PERIOD_VAL	PWM 周期脉冲周期值寄存器。
0x012C + 0x40×j	PWM_PERIODCNT	PWM 周期脉冲周期值寄存器。
0x500	PWM_ABNOR_STAT E0	pwm 异常状态寄存器 0。
0x504	PWM_ABNOR_STAT	pwm 异常状态寄存器 1。

偏移地址	名称	描述
	E1	
0x508	PWM_ABNOR_STAT E_CLR0	pwm 异常状态清除寄存器 0。
0x50C	PWM_ABNOR_STAT E_CLR1	pwm 异常状态清除寄存器 1。
0x510	PWM_INT_MASK	pwm 中断屏蔽寄存器。
0x0514	PWM_DMA_EN	pwm dma 使能寄存器。
0x0518	PWM_CFG_INT_CLR 0	pwm 步进模式循环结束中断清除。

PWM 寄存器偏移地址中变量的取值范围和含义如表 6-14 所示。

表6-14 PWM 寄存器偏移地址变量表

变量名称	取值范围	描述
i	0 ~ 3	PWM 分组
j	0 ~ 7	pwm 个数

6.6.5 寄存器描述

PWM_SEL

PWM_SEL 为 pwm 分组选择控制寄存器。

说明

本 IP 仅支持 8 个 PWM，最大可以支持分组数为 4；

即下述寄存器 i 支持 0~3，j 支持 0~7

Offset Address: 0x000 + 0x10×i Total Reset Value: 0x0000_0000

Bits	Access	Name	Description	Reset
[31:16]	-	reserved	保留。	0x0000
[15:0]	RW	pwm_sel_i	分组 pwm 选择信号，最多支持 4	0x0000

Bits	Access	Name	Description	Reset
			组，每组对应 8bit pwm_sel，每 bit 对应一路 pwm。	

PWM_STARTCLRCNT_EN

PWM_STARTCLRCNT_EN 为 pwm 配置生效计数清零使能寄存器。

Offset Address: 0x004 + 0x10×i Total Reset Value: 0x0000_0000

Bits	Access	Name	Description	Reset
[31:1]	-	reserved	保留。	0x00000000
[0]	RW	pwm_startclrnt_en_i	star 时刻对 pwm 内部寄存器清零使能。每组对应 1bit pwm_startclrnt_en。	0x0

PWM_START

PWM_START 为 PWM 配置生效寄存器。

Offset Address: 0x008 + 0x10×i Total Reset Value: 0x0000_0000

Bits	Access	Name	Description	Reset
[31:1]	-	reserved	保留。	0x00000000
[0]	W1_PULSE	pwm_start_i	pwm 配置生效寄存器。配成 1 时，此前赋值的 pwm 相关寄存器生效。每组对应 1bit pwm_start。 注意：因逻辑内部模块需要对相关寄存器配置值进行同步和处理，建议该寄存器的配置间隔时间至少 10μs 以上。	0x0

PWM_EN

PWM_EN 为 PWM 使能寄存器。

Offset Address: $0x0100 + 0x40 \times j$ Total Reset Value: 0x0000_0000

Bits	Access	Name	Description	Reset
[31:1]	-	reserved	保留	0x00000000
[0]	RW	pwm_en_j	pwm 功能使能, 每路 pwm 对应一个 pwm_en。 0: pwm 关闭, pwm_out 输出持续为 0; 1: pwm 使能打开。	0x0

PWM_PORTITY

PWM_PORTITY 为 PWM 的输出极性配置寄存器。

Offset Address: $0x0104 + 0x40 \times j$ Total Reset Value: 0x0000_0000

Bits	Access	Name	Description	Reset
[31:1]	-	reserved	保留	0x00000000
[0]	RW	pwm_poarity_j	Pwm 输出极性配置: 0: 正向极性; 1: 反向极性。	0x0

PWM_OEN_CFG

PWM_OEN_CFG 为 Pwm 无效电平高阻使能配置寄存器。

Offset Address: $0x0108 + 0x40 \times j$ Total Reset Value: 0x0000_0000

Bits	Access	Name	Description	Reset
[31:1]	-	reserved	保留	0x00000000

Bits	Access	Name	Description	Reset
				00
[0]	RW	pwm_oen_cfg_j	Pwm 无效电平高阻使能配置： 0：无效电平跟随极性等其他配置； 1：无效电平时 Pwm 输出高阻。	0x0

PWM_OFFSET_L

PWM_OFFSET_L 为 PWM_START 为 PWM 相位控制计数值低 16bit 寄存器。

Offset Address: 0x010C + 0x40×j Total Reset Value: 0x0000_0000

Bits	Access	Name	Description	Reset
[31:16]	-	reserved	保留。	0x0000
[15:0]	RW	pwm_offset_l_j	Pwm 相位控制计数值的低 16bit，控制 pwm 在一个周期内为 1 的位置，对 PWM，取值范围 1~65535。每路 pwm 对应一个 offset 参数。	0x0000

PWM_OFFSET_H

PWM_OFFSET_H 为 PWM_START 为 PWM 相位控制计数值高 16bit 寄存器。

Offset Address: 0x0110 + 0x40×j Total Reset Value: 0x0000_0000

Bits	Access	Name	Description	Reset
[31:16]	-	reserved	保留。	0x0000
[15:0]	RW	pwm_offset_h_j	Pwm 相位控制计数值的高 16bit，在 pwm_offset_l 计满后+1，控制 pwm 在一个周期内为 1 的位置，对 PWM，取值范围 1~65535。每	0x0000

Bits	Access	Name	Description	Reset
			路 pwm 对应一个 offset 参数。	

PWM_FREQ_L

PWM_FREQ_L 为 PWM_FREQ 为 PWM 频率控制计数值低 16bit 寄存器。

Offset Address: $0x0114 + 0x40 \times j$ Total Reset Value: 0x0000_0000

Bits	Access	Name	Description	Reset
[31:16]	-	reserved	保留。	0x0000
[15:0]	RW	pwm_freq_l_j	pwm 频率控制计数值的低 16bit, 含义为对 PWM 时钟的分频倍数, 取值范围 1~65535。每路 pwm 对应一个 freq 参数。	0x0000

PWM_FREQ_H

PWM_FREQ_H 为 PWM_FREQ 为 PWM 频率控制计数值高 16bit 寄存器。

Offset Address: $0x0118 + 0x40 \times j$ Total Reset Value: 0x0000_0000

Bits	Access	Name	Description	Reset
[31:16]	-	reserved	保留。	0x0000
[15:0]	RW	pwm_freq_h_j	pwm 频率控制计数值的高 16bit, 在 pwm_freq_l 计满后+1, 含义为对 PWM 时钟的分频倍数, 取值范围 1~65535。每路 pwm 对应一个 freq 参数。	0x0000

PWM_DUTY_L

PWM_DUTY_L 为 PWM_DUTY 为 PWM 占空比计数值寄存器。

Offset Address: 0x011C + 0x40×j Total Reset Value: 0x0000_0000

Bits	Access	Name	Description	Reset
[31:16]	-	reserved	保留。	0x0000
[15:0]	RW	pwm_duty_l_j	pwm 占空比控制计数值的低 16bit, 取值范围 1~65535。 pwm_duty 与 pwm_freq 的比值为占空比。每路 pwm 对应一个 duty 参数。	0x0000

PWM_DUTY_H

PWM_DUTY_H 为 PWM_DUTY 为 PWM 占空比计数值寄存器。

Offset Address: 0x0120 + 0x40×j Total Reset Value: 0x0000_0000

Bits	Access	Name	Description	Reset
[31:16]	-	reserved	保留。	0x0000
[15:0]	RW	pwm_duty_h_j	pwm 占空比控制计数值的高 16bit, 在 pwm_duty_l 计满后+1, 取值范围 1~65535。pwm_duty 与 pwm_freq 的比值为占空比。每路 pwm 对应一个 duty 参数。	0x0000

PWM_PERIODLOAD_FLAG

PWM_PERIODLOAD_FLAG 为 PWM 周期配置允许标志寄存器。

Offset Address: 0x0124 + 0x40×j Total Reset Value: 0x0000_0000

Bits	Access	Name	Description	Reset
[31:1]	-	reserved	保留。	0x00000000
[0]	RO	pwm_periodloa	pwm 平滑切换周期结束, 可进行	0x0

Bits	Access	Name	Description	Reset
		d_flag_j	新参数配置标志寄存器。周期计数结束产生。 配置 start 清除/或者配置 pwm_cfg_int_clr0 寄存可清除。	

PWM_PERIOD_VAL

PWM_PERIOD_VAL 为 PWM 周期脉冲周期值寄存器。

Offset Address: 0x0128 + 0x40×j Total Reset Value: 0x0000_0000

Bits	Access	Name	Description	Reset
[31:16]	-	reserved	保留。	0x0000
[15:0]	RW	pwm_period_val_j	Pwm 输出脉冲数值，非零时触发 pwm 输出脉冲计数功能，输出脉冲个数大于该值时会加载新参数继续计数。	0x0000

PWM_PERIODCNT

PWM_PERIODCNT 为 PWM 周期脉冲周期值寄存器。

Offset Address: 0x012C + 0x40×j Total Reset Value: 0x0000_0000

Bits	Access	Name	Description	Reset
[31:16]	-	reserved	保留。	0x0000
[15:0]	RO	pwm_periodcnt_j	Pwm 输出脉冲计数值。	0x0000

PWM_ABNOR_STATE0

PWM_ABNOR_STATE0 为 pwm 异常状态寄存器 0。

Offset Address: 0x500 Total Reset Value: 0x0000_0000

Bits	Access	Name	Description	Reset
[31:16]	-	reserved	保留。	0x0000
[15:0]	RO	pwm_abnor_state0	bit[15:0]为 Pwm 多路使用配置异常, 每路 pwm 对应 1bit。	0x0000

PWM_ABNOR_STATE1

PWM_ABNOR_STATE1 为 pwm 异常状态寄存器 1。

Offset Address: 0x504 Total Reset Value: 0x0000_0000

Bits	Access	Name	Description	Reset
[31:16]	RO	reserved	保留。	0x0000
[15:0]	RO	pwm_abnor_state1	bit[15:0]为 Pwm 计数值配置异常, 每路 pwm 对应 1bit。	0x0000

PWM_ABNOR_STATE_CLR0

PWM_ABNOR_STATE_CLR0 为 pwm 异常状态清除寄存器 0。

Offset Address: 0x508 Total Reset Value: 0x0000_0000

Bits	Access	Name	Description	Reset
[31:16]	RO	reserved	保留。	0x0000
[15:0]	W1_PULSE	pwm_abnor_state_clr0	Pwm 异常清除寄存器 0, 对应寄存器 pwm_abnor_state0。	0x0000

PWM_ABNOR_STATE_CLR1

PWM_ABNOR_STATE_CLR1 为 pwm 异常状态清除寄存器 1。

Offset Address: 0x50C Total Reset Value: 0x0000_0000

Bits	Access	Name	Description	Reset
[31:16]	RO	reserved	保留。	0x0000
[15:0]	W1_PU LSE	pwm_abnor_state_clr1	Pwm 异常清除寄存器，对应寄存器 pwm_abnor_state1。	0x0000

PWM_INT_MASK

PWM_INT_MASK 为 pwm 中断屏蔽寄存器。

Offset Address: 0x510 Total Reset Value: 0x0000_0000

Bits	Access	Name	Description	Reset
[31:2]	-	reserved	保留。	0x00000000
[1:0]	RW	pwm_int_mask	pwm 中断屏蔽寄存器， 0：屏蔽； 1：不屏蔽，默认屏蔽： bit[1]：步进模式一轮周期循环结束中断屏蔽； bit[0]：异常中断屏蔽。	0x0

PWM_DMA_EN

PWM_DMA_EN 为 pwm dma 使能寄存器。

Offset Address: 0x0514 Total Reset Value: 0x0000_0000

Bits	Access	Name	Description	Reset
[31:1]	-	reserved	保留。	0x00000000
[0]	RW	pwm_dma_en	dma 功能使能： 0：禁止。 1：使能；	0x0

PWM_CFG_INT_CLR0

PWM_CFG_INT_CLR0 为 pwm 步进模式循环结束中断清除。

Offset Address: 0x0518 Total Reset Value: 0x0000_0000

Bits	Access	Name	Description	Reset
[31:16]	RO	reserved	保留。	0x0000
[15:0]	W1_P ULSE	pwm_cfg_int_clr0	Pwmpwm 步进模式循环结束中断清除寄存器。 对应 pwm_cfg_int 的中断和 pwm_periodload_flag_j 寄存器。	0x0000

6.7 Tsensor

6.7.1 概述

Tsensor 是模拟温度检测 IP，检测芯片的节温并以二进制形式输出温度信息。

检查温度范围：-40°C ~ +125°C的温度检测，10bit SARADC 量化温度，分辨率 0.208°C /LSB。

支持 IP 校准后温度精度在±2°C以内。

6.7.2 功能描述

- 支持 Tsensor 三种测温模式：单次 16 点平均测温模式，周期 16 点平均测温模式，单点测温模式。
- 支持 Tsensor 测温完成中断上报。
- 支持 Tsensor 测温 overtemp 中断上报。
- 支持高温低温门限的使用。
- 支持软件分别可配高温门限和低温门限。
- 支持周期采样温度上报。
- 支持软件可配周期采样间隔。

- 支持 Tsensor 测温门限中断上报。

6.7.3 工作方式

Tsensor 模块的工作模式分为以下 3 种：

- 正常检测温度模式。
- 高低温门限中断模式。
- 过温保护中断模式。

说明

以上 3 种模式均为在检测温度值基础上进行，检测温度的模式有多种，此处检测温度的模式均为周期采样配合 16 次单点平均计算。

6.7.3.1 正常检测温度模式

正常检测温度模式配置步骤如下：

- 步骤 1 写 `TSENSOR_STS`[`tsensor_clr`]为 0x1，清除所有模式的状态信息。
- 步骤 2 写 `TSENSOR_TEMP_INT_CLR`[`tsensor_int_clr`]为 0x1，清除 Tsensor 中断信号。
- 步骤 3 写 `TSENSOR_TEMP_INT_EN`[`tsensor_done_int_en`]为 0x1，打开 Tsensor 的采集温度完成中断使能。
- 步骤 4 写 `TSENSOR_CTRL`[`tsensor_mode`]为 0x0，选择 16 次平均值的单次上报方式。
- 步骤 5 写 `TSENSOR_CTRL`[`tsensor_enable`]为 0x1，开启 Tsensor 的使能信号。
- 步骤 6 写 `TSENSOR_AUTO_REFRESH_PERIOD`[`tsensor_auto_refresh_period`]，设置合适的周期采样的时间间隔。
- 步骤 7 写 `TSENSOR_AUTO_REFRESH_CFG`[`tsensor_auto_refresh_enable`]为 0x1，开启周期采样的使能信号。
- 步骤 8 读 `TSENSOR_TEMP_INT_STS`[`tsensor_done_int_sts`]为 0x1，等待 16 点平均计算温度模式下的中断信号的产生。
- 步骤 9 读 `TSENSOR_STS`[`tsensor_data`]，获取 16 点平均单次上报模式下的温度值。

----结束

6.7.3.2 高低温门限中断模式

高低温门限中断模式配置步骤如下：

- 步骤 1 写 `TSENSOR_STS`[`tsensor_clr`]为 0x1, 清除自动模式下产生的 rdy 信号。
- 步骤 2 写 `TSENSOR_TEMP_INT_CLR`[`tsensor_int_clr`]为 0x1, 清除 Tsensor 中断信号。
- 步骤 3 写 `TSENSOR_TEMP_INT_EN`[`tsensor_out_thresh_int_en`]为 0x1, 打开 Tsensor 的超门限范围中断使能。
- 步骤 4 写 `TSENSOR_TEMP_HIGH_LIMIT`[`tsensor_temp_high_limit`]、`TSENSOR_TEMP_LOW_LIMIT`[`tsensor_temp_low_limit`], 设置合适的高低温门限值。
- 步骤 5 写 `TSENSOR_CTRL`[`tsensor_mode`]为 0x0, 选择 16 次平均值的单次上报方式。
- 步骤 6 写 `TSENSOR_CTRL`[`tsensor_enable`]为 0x1, 开启 Tsensor 的使能信号。
- 步骤 7 写 `TSENSOR_AUTO_REFRESH_PERIOD`[`tsensor_auto_refresh_period`], 设置合适的周期采样的时间间隔。
- 步骤 8 写 `TSENSOR_AUTO_REFRESH_CFG`[`tsensor_auto_refresh_enable`]为 0x1, 开启周期采样的使能信号。
- 步骤 9 读 `TSENSOR_TEMP_INT_STS`[`tsensor_out_thresh_int_sts`]是否为 1, 如果为 1, 则当前的温度值高于高温门限或低于低温门限。

----结束

6.7.3.3 过温保护中断模式

过温保护中断模式配置步骤如下:

- 步骤 1 写 `TSENSOR_STS`[`tsensor_clr`]为 0x1, 清除自动模式下产生的 rdy 信号。
- 步骤 2 写 `TSENSOR_TEMP_INT_CLR`[`tsensor_int_clr`]为 0x1, 清除 Tsensor 中断信号。
- 步骤 3 写 `TSENSOR_TEMP_INT_EN`[`tsensor_overtemp_int_en`]为 0x1, 打开 Tsensor 的过温中断使能。
- 步骤 4 写 `TSENSOR_OVER_TEMP`[`tsensor_overtemp_thresh`], 设置合适的过温门限值。
- 步骤 5 写 `TSENSOR_OVER_TEMP`[`tsensor_overtemp_thresh_en`]为 0x1, 打开过温保护使能信号。
- 步骤 6 写 `TSENSOR_CTRL`[`tsensor_mode`]为 0x0, 选择 16 次平均值的单次上报方式。
- 步骤 7 写 `TSENSOR_CTRL`[`tsensor_enable`]为 0x1, 开启 Tsensor 的使能信号。

步骤 8 写 `TSENSOR_AUTO_REFRESH_PERIOD`[`tsensor_auto_refresh_period`]，设置合适的周期采样的时间间隔。

步骤 9 写 `TSENSOR_AUTO_REFRESH_CFG`[`tsensor_auto_refresh_enable`]为 0x1，开启周期采样的使能信号。

读 `TSENSOR_TEMP_INT_STS`[`tsensor_overtemp_int_sts`]是否为 1，如果为 1，则当前的温度值超过了设置的过温门限值。

----结束

6.7.4 寄存器概览

Tsensor 寄存器概览如表 6-15 所示。

表6-15 Tsensor 寄存器概览（基址是 0x4000_0000）

偏移地址	名称	描述
0x0000 + 0x1000 × CH_NUM	TSENSOR_CTL_ID	TSENSOR CTL ID 寄存器。
0x0010 + 0x1000 × CH_NUM	TSENSOR_REG0	通用寄存器。
0x0014 + 0x1000 × CH_NUM	TSENSOR_REG1	通用寄存器。
0x0018 + 0x1000 × CH_NUM	TSENSOR_REG2	通用寄存器。
0x001C + 0x1000 × CH_NUM	TSENSOR_REG3	通用寄存器。
0x0300 + 0x1000 × CH_NUM	TSENSOR_START	TSENSOR 启动寄存器。
0x0304 + 0x1000 × CH_NUM	TSENSOR_CTRL	TSENSOR 控制寄存器。
0x0308 + 0x1000 × CH_NUM	TSENSOR_STS	TSENSOR 状态寄存器。
0x0310 + 0x1000 × CH_NUM	TSENSOR_CTRL1	TSENSOR 控制寄存器 1。

偏移地址	名称	描述
0x0314 + 0x1000×CH_NUM	TSENSOR_TEMP_HIGH_LIMIT	TSENSOR 温度门限上限。
0x0318 + 0x1000×CH_NUM	TSENSOR_TEMP_LOW_LIMIT	TSENSOR 温度门限下限。
0x031C + 0x1000×CH_NUM	TSENSOR_OVER_TEMP	TSENSOR 过温控制寄存器。
0x0320 + 0x1000×CH_NUM	TSENSOR_TEMP_INT_EN	TSENSOR 中断控制寄存器。
0x0324 + 0x1000×CH_NUM	TSENSOR_TEMP_INT_CLR	TSENSOR 中断控制寄存器。
0x0328 + 0x1000×CH_NUM	TSENSOR_TEMP_INT_STS	TSENSOR 中断控制寄存器。
0x0330 + 0x1000×CH_NUM	TSENSOR_AUTO_REFRESH_PERIOD	TSENSOR 自动检测控制寄存器。
0x0334 + 0x1000×CH_NUM	TSENSOR_AUTO_REFRESH_CFG	TSENSOR 自动检测控制寄存器。

Tsensor 寄存器偏移地址中变量的取值范围和含义如表 6-16 所示。

表6-16 Tsensor 寄存器偏移地址变量表

变量名称	取值范围	描述
CH_NUM	0	Tsensor 组数

6.7.5 寄存器描述

TSENSOR_CTL_ID

TSENSOR_CTL_ID 为 TSENSOR CTL ID 寄存器

Offset Address: 0x0000 + 0x1000×CH_NUM Total Reset Value: 0x0106

Bits	Access	Name	Description	Reset
[15:0]	RO	tsensor_ctl_id	TSensor CTL ID 寄存器。	0x0106

TSensor_REG0

TSensor_REG0 为通用寄存器。

Offset Address: 0x0010 + 0x1000×CH_NUM Total Reset Value: 0x0000

Bits	Access	Name	Description	Reset
[15:0]	RW	tsensor_reg0	Tsensor 通用寄存器 0。	0x0000

TSensor_REG1

TSensor_REG1 为通用寄存器。

Offset Address: 0x0014 + 0x1000×CH_NUM Total Reset Value: 0x0000

Bits	Access	Name	Description	Reset
[15:0]	RW	tsensor_reg1	Tsensor 通用寄存器 1。	0x0000

TSensor_REG2

TSensor_REG2 为通用寄存器。

Offset Address: 0x0018 + 0x1000×CH_NUM Total Reset Value: 0x0000

Bits	Access	Name	Description	Reset
[15:0]	RW	tsensor_reg2	Tsensor 通用寄存器 2。	0x0000

TSensor_REG3

TSensor_REG3 为通用寄存器。

Offset Address: 0x001C + 0x1000×CH_NUM Total Reset Value: 0x0000

Bits	Access	Name	Description	Reset
[15:0]	RW	tsensor_reg3	Tsensor 通用寄存器 3。	0x0000

TSENSOR_START

TSENSOR_START 为 TSENSOR 启动寄存器。

Offset Address: 0x0300 + 0x1000×CH_NUM Total Reset Value: 0x0000

Bits	Access	Name	Description	Reset
[15:1]	-	reserved	保留。	0x0000
[0]	W1_PULSE	tsensor_start	自动模式下，写 1 刷新一次温度码，回读 tsensor_data_auto 获取当前温度值，当 tsensor_rdy_auto 为 1 时，表明温度值有效。 写 0 无效。	0x0

TSENSOR_CTRL

TSENSOR_CTRL 为 TSENSOR 控制寄存器。

Offset Address: 0x0304 + 0x1000×CH_NUM Total Reset Value: 0x0000

Bits	Access	Name	Description	Reset
[15:3]	-	reserved	保留。	0x0000
[2:1]	RW	tsensor_mode	模式选择 2'b00: 16 点平均单次上报模式; 2'b01: 16 点平均循环上报模式; 2'b10、2'b11: 单点循环上报模式 (该模式不比较阈值，仅上报温度码)。	0x0

Bits	Access	Name	Description	Reset
[0]	RW	tsensor_enable	TSENSOR_CTRL 开关 0: 关闭 TENSOR_CTRL; 1: 打开 TENSOR_CTRL。	0x0

TSENSOR_STS

TSENSOR_STS 为 TSENSOR 状态寄存器。

Offset Address: 0x0308 + 0x1000×CH_NUM Total Reset Value: 0x0000

Bits	Access	Name	Description	Reset
[15:12]	-	reserved	保留。	0x0
[11:2]	RO	tsensor_data	所有模式下获取到的温度值。 10bit 温度区间码值输出，线性分布; -40C~dec'114 125C~dec'896 即是 tsensor 直接输出码值对温度°C单位换算公式 $T_{\text{°C}} = [\text{BIN2DEC}(\text{temp_out} \text{ <9:0>}) - 114] / (896 - 114) * [125 - (-40)] + (-40)$ 。	0x000
[1]	RO	tsensor_rdy	所有模式下。 0: 检测未启动或手动检测中; 1: tsensor_data 值为有效的温度值。	0x0
[0]	W1_PULSE	tsensor_clr	清除所有模式的状态。 0: 无效; 1: 清除。	0x0

TSENSOR_CTRL1

TSENSOR_CTRL1 为 TSENSOR 控制寄存器 1。

Offset Address: 0x0310 + 0x1000×CH_NUM Total Reset Value: 0x0000

Bits	Access	Name	Description	Reset
[15:5]	-	reserved	保留。	0x000
[4]	RW	temp_scan_dft	DFT 使能 0: 功能输出; 1: DFT 输出。	0x0
[3]	RW	temp_set	0: 功能模式下 temp_out 正常输出 (默认) ; 1: 功能模式下 temp_out 可以通过数模接口输入来配置输出值。	0x0
[2:1]	RW	temp_ct_sel	对于输入 1M 精准时钟情况下: 00: 0.512ms; 01: 0.256ms; 10: 1.024ms; 11: 2.048ms。	0x0
[0]	RW	temp_calib	0: 选择开启校准算法 (默认) ; 1: 不开启校准算法。	0x0

TSENSOR_TEMP_HIGH_LIMIT

TSENSOR_TEMP_HIGH_LIMIT 为 TSENSOR 温度门限上限。

Offset Address: 0x0314 + 0x1000×CH_NUM Total Reset Value: 0x0000

Bits	Access	Name	Description	Reset
[15:10]	-	reserved	保留。	0x00
[9:0]	RW	tsensor_temp_high_limit	过高温度阈值门限。	0x000

TSENSOR_TEMP_LOW_LIMIT

TSENSOR_TEMP_LOW_LIMIT 为 TSENSOR 温度门限下限。

Offset Address: $0x0318 + 0x1000 \times CH_NUM$ Total Reset Value: 0x0000

Bits	Access	Name	Description	Reset
[15:10]	-	reserved	保留。	0x00
[9:0]	RW	tsensor_temp_low_limit	过低温度阈值门限。	0x000

TSENSOR_OVER_TEMP

TSENSOR_OVER_TEMP 为 TSENSOR 过温控制寄存器。

Offset Address: $0x031C + 0x1000 \times CH_NUM$ Total Reset Value: 0x03FF

Bits	Access	Name	Description	Reset
[15:11]	-	reserved	保留。	0x00
[10]	RW	tsensor_overtemp_thresh_en	16 点平均单次上报模式或 16 点平均循环上报模式下过温 PA 保护使能。 0: PA 保护使能关闭; 1: PA 保护使能打开。	0x0
[9:0]	RW	tsensor_overtemp_thresh	过温保护阈值门限。	0x3FF

TSENSOR_TEMP_INT_EN

TSENSOR_TEMP_INT_EN 为 TSENSOR 中断控制寄存器

Offset Address: $0x0320 + 0x1000 \times CH_NUM$ Total Reset Value: 0x0000

Bits	Access	Name	Description	Reset
------	--------	------	-------------	-------

Bits	Access	Name	Description	Reset
[15:3]	-	reserved	保留。	0x0000
[2]	RW	tsensor_overtemp_int_en	TSENSOR 过温保护中断使能。	0x0
[1]	RW	tsensor_out_threshold_int_en	TSENSOR 温度超门限范围中断使能。 0: 禁止; 1: 使能。	0x0
[0]	RW	tsensor_done_int_en	TSENSOR 温度采集完毕中断使能。 0: 禁止; 1: 使能。	0x0

TSENSOR_TEMP_INT_CLR

TSENSOR_TEMP_INT_CLR 为 TSENSOR 中断控制寄存器。

Offset Address: 0x0324 + 0x1000×CH_NUM Total Reset Value: 0x0000

Bits	Access	Name	Description	Reset
[15:1]	-	reserved	保留。	0x0000
[0]	W1_PULSE	tsensor_int_clr	0: 无效; 1: 清除中断。	0x0

TSENSOR_TEMP_INT_STS

TSENSOR_TEMP_INT_STS 为 TSENSOR 中断控制寄存器。

Offset Address: 0x0328 + 0x1000×CH_NUM Total Reset Value: 0x0000

Bits	Access	Name	Description	Reset
[15:3]	-	reserved	保留。	0x0000

Bits	Access	Name	Description	Reset
[2]	RO	tsensor_overtemp_int_sts	TSENSOR 过温保护中断状态。	0x0
[1]	RO	tsensor_out_thresh_int_sts	TSENSOR 温度超门限范围中断状态。	0x0
[0]	RO	tsensor_done_int_sts	TSENSOR 温度采集完毕中断状态。	0x0

TSENSOR_AUTO_REFRESH_PERIOD

TSENSOR_AUTO_REFRESH_PERIOD 为 TSENSOR 自动检测控制寄存器。

Offset Address: 0x0330 + 0x1000×CH_NUM Total Reset Value: 0xFFFF

Bits	Access	Name	Description	Reset
[15:0]	RW	tsensor_auto_refresh_period	TSENSOR 自动检测周期, 32k 时钟周期数。	0xFFFF

TSENSOR_AUTO_REFRESH_CFG

TSENSOR_AUTO_REFRESH_CFG 为 TSENSOR 自动检测控制寄存器。

Offset Address: 0x0334 + 0x1000×CH_NUM Total Reset Value: 0x0000

Bits	Access	Name	Description	Reset
[15:1]	-	reserved	保留。	0x0000
[0]	RW	tsensor_auto_refresh_enable	16 点平均单次上报模式下周期检测使能。 0: 定时周期检测关闭; 1: 定时周期检测打开。	0x0

6.8 I2S

6.8.1 概述

I2S 模块是 APB 总线上的从设备、I2S 总线上的主/从设备。

6.8.2 功能描述

- 支持 Master/Slave 模式。
- 支持 TRX 的左右声道各自独立 FIFO，FIFO 规格为 32X8。
- 支持 8kHz/16kHz/32kHz/44.1kHz/48kHz/96kHz/128kHz。
- 支持 16/24/32 位的传输工作模式。
- 支持 I2S 协议/PCM-TDM 协议（PCM-TDM 只支持 RX 方向的 16 位传输工作模式）。

6.8.3 工作方式

6.8.3.1 接口信号

I2S 接口信号描述如表 6-17 所示。

表6-17 I2S 接口信号

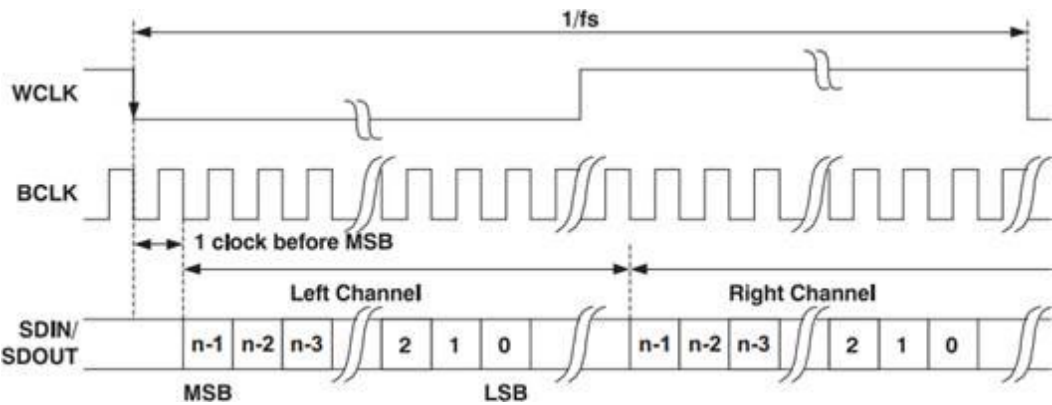
信号名	宽度 (bit)	方向	功能描述
AUDIO_AIAO_BCLK	1	O	串行时钟 SCLK，也称位时钟（BCLK），即对应数字音频的每 1bit 数据，SCLK 都有 1 个脉冲。 SCK（Serial Clock singal）的频率=2×采样频率×采样位数。 由 Master 产生并控制。
AUDIO_AIAO_WS	1	O	左右声道选择信号 Word Select。 WS 为声道选择信号，即数据发送端所选择的声道： 0：选择左声道； 1：选择右声道。

信号名	宽度 (bit)	方向	功能描述
			WS 也称帧时钟，即 LRCLK (Left Right Clock)。WS 频率等于声音的采样率。由 Master 产生并控制。
AUDIO_AIAO_M_CLK	1	O	Master Clock。在 I2S 接口的 ADC/DAC 系统中，除 SCK 和 WS 外，CODEC (Coder/Decoder) 还需要控制器提供 MCLK (Master Clock)。由 Master 产生并控制。
AUDIO_AIAO_TX_SD	1	O	输出数据。
AUDIO_AIAO_RX_SD	1	I	输入数据。

6.8.3.2 操作模式

I2S 操作模式如图 6-16 所示。

图6-16 I2S 操作模式



SD 是串行数据，在 I2S 中以二进制补码的形式在数据线上传输。在 WS 变化后的第 1 个 SCK 脉冲，先传输最高位 (MSB)。左声道的数据 MSB 在 WS 下降沿之后第 2 个 SCK/BCLK 上升沿有效，右声道数据的 MSB 在 WS 上升沿之后第 2 个 SCK/BCLK 上升沿有效。

6.8.3.3 应用说明

Master TX 模式

1. 初始化

- 步骤 1 进行时钟的初始化配置，打开 I2S 需要的源头 960M 时钟。
- 步骤 2 参考 IO_MUX，配置好 I2S 对应的 IO 复用，以及 IO 的驱动能力。
- 步骤 3 设置 `SIO_CT_SET[rst_n]` 为 0x1，解除 I2S 控制器的复位。
- 步骤 4 设置 `SIO_CT_SET[intr_en]` 为 0x1，打开 I2S 的全局中断使能。
- 步骤 5 设置 `SIO_CT_SET[tx_enable]` 为 0x1，打开 I2S TX 发送使能。
- 步骤 6 设置 `SIO_MODE[cfg_i2s_ms_mode_sel]` 为 0x0，设置为 Master 模式。
- 步骤 7 设置 `SIO_MODE[ext_rec_en]`，`SIO_MODE[sio_mode]`，`SIO_DATA_WIDTH_SET[rx_mode]`，设置 I2S 进行通信的协议模式以及位宽。
- 步骤 8 设置 `SIO_FIFO_THRESHOLD[tx_fifo_threshold]`，设置发送 FIFO 的水线阈值。
- 步骤 9 设置 `SIO_INTMASK[tx_intr]` 为 0x0，设置 TX_FIFO 中断不屏蔽。
- 步骤 10 设置地址 `0x4400_1100[12]` 为 0x1，打开 I2S 的工作时钟门控。
- 步骤 11 设置 `CFG_I2S_CRG[cfg_i2s_bclk_div_en]` 为 0x0，关闭分频器使能开关。
- 步骤 12 设置 `CFG_I2S_BCLK_DIV_NUM[cfg_i2s_bclk_div_num]`，配置 MCLK -> BCLK 的分频值。
- 步骤 13 设置 `CFG_I2S_CRG[cfg_i2s_bclk_div_en]` 为 0x1，打开分频器使能开关。
- 步骤 14 设置 `CFG_I2S_FS_DIV_NUM[cfg_i2s_fs_div_num]`，配置 BCLK -> WS 的分频值。
- 步骤 15 设置 `CFG_I2S_FS_DIV_RATIO_NUM[cfg_i2s_fs_div_ratio_num]`，配置 WS 输出占空比。
- 步骤 16 设置 `CFG_I2S_CRG[cfg_i2s_crg_clken]` 为 0x1，打开时钟门控。

----结束

2. 发送数据

- 步骤 1 CPU 接收到中断，或者读取寄存器 `SIO_INTSTATUS[tx_intr]` 为 0x1 时，此时表征 I2S TX_FIFO 中的数据不足水线阈值。

步骤 2 写寄存器 `SIO_I2S_LEFT_XD[tx_left_data]` 以及 `SIO_I2S_RIGHT_XD[tx_right_data]`, 向 I2S 的左右声道 FIFO 中写入数据。

步骤 3 设置 `SIO_INTCLR[tx_intr]` 为 0x1, 清除发送 FIFO 中断。

步骤 4 设置 `SIO_CT_SET[tx_enable]` 为 0x0, 关闭 I2S TX 发送使能。

----结束

Master RX 模式

1. 初始化

步骤 1 进行时钟的初始化配置, 打开 I2S 需要的源头 960M 时钟。

步骤 2 参考 IO_MUX, 配置好 I2S 对应的 IO 复用, 以及 IO 的驱动能力。

步骤 3 设置 `SIO_CT_SET[rst_n]` 为 0x1, 解除 I2S 控制器的复位。

步骤 4 设置 `SIO_CT_SET[intr_en]` 为 0x1, 打开 I2S 的全局中断使能。

步骤 5 设置 `SIO_CT_SET[rx_enable]` 为 0x1, 打开 I2S RX 接收使能。

步骤 6 设置 `SIO_MODE[cfg_i2s_ms_mode_sel]` 为 0x0, 设置为 Master 模式。

步骤 7 设置 `SIO_MODE[ext_rec_en]`, `SIO_MODE[sio_mode]`, `SIO_DATA_WIDTH_SET[rx_mode]`, 设置 I2S 进行通信的协议模式以及位宽。

步骤 8 设置 `SIO_FIFO_THRESHOLD[rx_fifo_threshold]`, 设置接收 FIFO 的水线阈值。

步骤 9 设置 `SIO_INTMASK[rx_intr]` 为 0x0, 设置 RX_FIFO 中断不屏蔽。

步骤 10 设置地址 `0x4400_1100[12]` 为 0x1, 打开 I2S 的工作时钟门控。

步骤 11 设置 `CFG_I2S_CRG[cfg_i2s_bclk_div_en]` 为 0x0, 关闭分频器使能开关。

步骤 12 设置 `CFG_I2S_BCLK_DIV_NUM[cfg_i2s_bclk_div_num]`, 配置 MCLK -> BCLK 的分频值。

步骤 13 设置 `CFG_I2S_CRG[cfg_i2s_bclk_div_en]` 为 0x1, 打开分频器使能开关。

步骤 14 设置 `CFG_I2S_FS_DIV_NUM[cfg_i2s_fs_div_num]`, 配置 BCLK -> WS 的分频值。

步骤 15 设置 `CFG_I2S_FS_DIV_RATIO_NUM[cfg_i2s_fs_div_ratio_num]`, 配置 WS 输出占空比。

步骤 16 设置 `CFG_I2S_CRG[cfg_i2s_crg_clken]` 为 0x1, 打开时钟门控。

----结束**2. 接收数据**

- 步骤 1 CPU 接收到中断，或者读取寄存器 `SIO_INTSTATUS`[rx_intr]为 0x1 时，此时表征 I2S RX_FIFO 中数据量达到了水线阈值。
- 步骤 2 读取寄存器 `SIO_I2S_LEFT_RD`[rx_left_data]与 `SIO_I2S_RIGHT_RD`[rx_right_data]，将左右声道接收 FIFO 中的数据读取出来。
- 步骤 3 设置 `SIO_INTCLR`[rx_intr]为 0x1，清除接收 FIFO 中断。
- 步骤 4 设置 `SIO_CT_SET`[rx_enable]为 0x0，关闭 I2S RX 接收使能。

----结束**Slave TX 模式****1. 初始化**

- 步骤 1 进行时钟的初始化配置，打开 I2S 需要的源头 960M 时钟。
- 步骤 2 参考 IO_MUX，配置好 I2S 对应的 IO 复用，以及 IO 的驱动能力。
- 步骤 3 设置 `SIO_CT_SET`[rst_n]为 0x1，解除 I2S 控制器的复位。
- 步骤 4 设置 `SIO_CT_SET`[intr_en]为 0x1，打开 I2S 的全局中断使能。
- 步骤 5 设置 `SIO_CT_SET`[tx_enable]为 0x1，打开 I2S TX 发送使能。
- 步骤 6 设置 `SIO_MODE`[cfg_i2s_ms_mode_sel]为 0x1，设置为 Slave 模式。
- 步骤 7 设置 `SIO_MODE`[ext_rec_en]，`SIO_MODE`[sio_mode]，`SIO_DATA_WIDTH_SET`[rx_mode]，设置 I2S 进行通信的协议模式以及位宽。
- 步骤 8 设置 `SIO_FIFO_THRESHOLD`[tx_fifo_threshold]，设置发送 FIFO 的水线阈值。
- 步骤 9 设置 `SIO_INTMASK`[tx_intr]为 0x0，设置 TX_FIFO 中断不屏蔽。
- 步骤 10 设置地址 0x4400_1100[12]为 0x1，打开 I2S 的工作时钟门控。
- 步骤 11 设置 `CFG_I2S_CRG`[cfg_i2s_bclk_div_en]为 0x0，关闭分频器使能开关。
- 步骤 12 设置 `CFG_I2S_BCLK_DIV_NUM`[cfg_i2s_bclk_div_num]，配置 MCLK -> BCLK 的分频值。
- 步骤 13 设置 `CFG_I2S_CRG`[cfg_i2s_bclk_div_en]为 0x1，打开分频器使能开关。

步骤 14 设置 `CFG_I2S_FS_DIV_NUM`[`cfg_i2s_fs_div_num`]，配置 BCLK -> WS 的分频值。

步骤 15 设置 `CFG_I2S_FS_DIV_RATIO_NUM`[`cfg_i2s_fs_div_ratio_num`]，配置 WS 输出占空比。

步骤 16 设置 `CFG_I2S_CRG`[`cfg_i2s_crg_clken`]为 0x1，打开时钟门控。

----结束

2. 发送数据

步骤 1 CPU 接收到中断，或者读取寄存器 `SIO_INTSTATUS`[`tx_intr`]为 0x1 时，此时表征 I2S TXFIFO 中的数据不足水线阈值。

步骤 2 写寄存器 `SIO_I2S_LEFT_XD`[`tx_left_data`]以及 `SIO_I2S_RIGHT_XD`[`tx_right_data`]，向 I2S 的左右声道 FIFO 中写入数据。

步骤 3 设置 `SIO_INTCLR`[`tx_intr`]为 0x1，清除发送 FIFO 中断。

步骤 4 设置 `SIO_CT_SET`[`tx_enable`]为 0x0，关闭 I2S TX 发送使能。

----结束

Slave RX 模式

1. 初始化

步骤 1 进行时钟的初始化配置，打开 I2S 需要的源头 960M 时钟。

步骤 2 参考 IO_MUX，配置好 I2S 对应的 IO 复用，以及 IO 的驱动能力。

步骤 3 设置 `SIO_CT_SET`[`rst_n`]为 0x1，解除 I2S 控制器的复位。

步骤 4 设置 `SIO_CT_SET`[`intr_en`]为 0x1，打开 I2S 的全局中断使能。

步骤 5 设置 `SIO_CT_SET`[`rx_enable`]为 0x1，打开 I2S RX 接收使能。

步骤 6 设置 `SIO_MODE`[`cfg_i2s_ms_mode_sel`]为 0x1，设置为 Slave 模式。

步骤 7 设置 `SIO_MODE`[`ext_rec_en`]，`SIO_MODE`[`sio_mode`]，`SIO_DATA_WIDTH_SET`[`rx_mode`]，设置 I2S 进行通信的协议模式以及位宽。

步骤 8 设置 `SIO_FIFO_THRESHOLD`[`rx_fifo_threshold`]，设置接收 FIFO 的水线阈值。

步骤 9 设置 `SIO_INTMASK`[`rx_intr`]为 0x0，设置 RX_FIFO 中断不屏蔽。

步骤 10 设置地址 0x4400_1100[12]为 0x1，打开 I2S 的工作时钟门控。

- 步骤 11 设置 `CFG_I2S_CRG[cfg_i2s_bclk_div_en]` 为 0x0，关闭分频器使能开关。
- 步骤 12 设置 `CFG_I2S_BCLK_DIV_NUM[cfg_i2s_bclk_div_num]`，配置 MCLK -> BCLK 的分频值。
- 步骤 13 设置 `CFG_I2S_CRG[cfg_i2s_bclk_div_en]` 为 0x1，打开分频器使能开关。
- 步骤 14 设置 `CFG_I2S_FS_DIV_NUM[cfg_i2s_fs_div_num]`，配置 BCLK -> WS 的分频值。
- 步骤 15 设置 `CFG_I2S_FS_DIV_RATIO_NUM[cfg_i2s_fs_div_ratio_num]`，配置 WS 输出占空比。
- 步骤 16 设置 `CFG_I2S_CRG[cfg_i2s_crg_clken]` 为 0x1，打开时钟门控。

----结束

2. 接收数据

- 步骤 1 CPU 接收到中断，或者读取寄存器 `SIO_INTSTATUS[rx_intr]` 为 0x1 时，此时表征 I2S RX_FIFO 中数据量达到了水线阈值。
- 步骤 2 读取寄存器 `SIO_I2S_LEFT_RD[rx_left_data]` 与 `SIO_I2S_RIGHT_RD[rx_right_data]`，将左右声道接收 FIFO 中的数据读取出来。
- 步骤 3 设置 `SIO_INTCLR[rx_intr]` 为 0x1，清除接收 FIFO 中断。
- 步骤 4 设置 `SIO_CT_SET[rx_enable]` 为 0x0，关闭 I2S RX 接收使能。

----结束

6.8.4 寄存器概况

I2S_DIO 寄存器概览如表 6-18 所示。

表6-18 I2S_DIO 寄存器概览（基址是 0x4402_5000）

偏移地址	名称	描述
0x003C	<code>SIO_VERSION</code>	版本号与测试模式配置。
0x0040	<code>SIO_MODE</code>	模式配置寄存器。
0x0044	<code>SIO_INTSTATUS</code>	中断状态查询寄存器。
0x0048	<code>SIO_INTCLR</code>	中断清除寄存器。

偏移地址	名称	描述
0x004C	SIO_I2S_LEFT_XD	左声道发送数据。
0x0050	SIO_I2S_RIGHT_XD	右声道发送数据。
0x0054	SIO_I2S_LEFT_RD	左声道接收数据。
0x0058	SIO_I2S_RIGHT_RD	右声道接收数据。
0x005C	SIO_CT_SET	I2S_CT 置位寄存器。
0x0060	SIO_CT_CLR	I2S_CT 清除寄存器。
0x0064	SIO_FIFO_THRESHO LD	FIFO 阈值配置寄存器。
0x0068	SIO_RX_STA	RX_FIFO 深度指示。
0x006C	SIO_TX_STA	TX_FIFO 深度指示。
0x0078	SIO_DATA_WIDTH_S ET	传输数据长度配置。
0x007C	SIO_I2S_START_PO S	声道访问配置寄存器。
0x0080	I2S_POS_FLAG	声道访问状态寄存器。
0x0084	SIO_SIGNED_EXT	高位数据符号扩展使能寄存器。
0x0088	SIO_I2S_POS_MERG E_EN	左右声道数据的操作地址合并使能寄存器。
0x008C	SIO_INTMASK	中断屏蔽寄存器。
0x0090	CFG_I2S_CRG	时钟配置寄存器。
0x0094	CFG_I2S_BCLK_DIV_ NUM	BCLK 分频比配置寄存器。
0x0098	CFG_I2S_FS_DIV_N UM	FSCLK 分频器的分频比配置寄存器。
0x009C	CFG_I2S_FS_DIV_R ATIO_NUM	FSCLK 分频器的占空比配置寄存器
0x00A0	SIO_I2S_DUAL_RX_ CHN	交替读取接收 fifo 功能寄存器。
0x00C0	SIO_I2S_DUAL_TX_C HN	交替写入发送 fifo 功能寄存器。

6.8.5 寄存器描述

SIO_VERSION

SIO_VERSION 为版本号与测试模式配置。

Offset Address: 0x003C Total Reset Value: 0x0000_0013

Bits	Access	Name	Description	Reset
[31:9]	-	reserved	保留。	0x000000
[8]	RW	sio_loop	SIO 循环和正常模式选择。 0: 正常模式; 1: SIO 发送和接收数据环回模式。用于 SIO 的自测试。在该模式下, 在 SIO 的对外接口处, SIO 接收串行数据线与 SIO 发送串行数据线直接相连。	0x0
[7:0]	RO	vesion	SIO 的版本号。	0x13

SIO_MODE

SIO_MODE 为模式配置寄存器。

Offset Address: 0x0040 Total Reset Value: 0x0000_0080

Bits	Access	Name	Description	Reset
[31:8]	-	reserved	保留。	0x000000
[7]	RW	cfg_i2s_ms_mode_sel	SIO 模块 MASTER / SLAVE mode 选择寄存器 1'b1: master 模式; 1'b0: slave 模式。	0x1
[6]	RW	clk_edge	PCM 多路接收模式下, 采样数据时钟边沿选择。	0x0

Bits	Access	Name	Description	Reset
			0: 下降沿有效; 1: 上升沿有效。	
[5:4]	RW	chn_num	多路接收的通道数选择。 00: 2chn; 01: 4chn; 10: 8chn; 11: 16chn。	0x0
[3]	RW	ext_rec_en	标准模式下, I2S 接收两个声道的数据, 即左声道和右声道。PCM 只接收一个声道的数据。 多路接收模式下, 对于 I2S 或 PCM, 接收通道数可以配置。该模式下, 通道的数据宽度必须为 8bit 或 16bit。 0: 标准的 I2S 或 PCM 接收模式; 1: 扩展的 I2S 或 PCM 多路接收模式。	0x0
[2]	-	rxws_select	选择 RX 模块的 ws 来自于 txws_dac 还是 rxws_adc	0x0
[1]	RW	pcm_mode	PCM 时序模式。 0: 标准模式; 1: 自定义模式。	0x0
[0]	RW	sio_mode	PCM/I2S 模式选择。 0: I2S 模式; 1: PCM 模式。	0x0

SIO_INTSTATUS

SIO_INTSTATUS 为中断状态查询寄存器。

Offset Address: 0x0044 Total Reset Value: 0x0000_0000

Bits	Access	Name	Description	Reset
[31:6]	-	reserved	保留。	0x00000000
[5]	RO	tx_left_fifo_under	I2S 模式下, 左声道发送 FIFO 下溢中断状态。PCM 模式下无效。 0: 未产生中断; 1: 已产生中断。	0x0
[4]	RO	tx_right_fifo_under	I2S 模式下, 右声道发送 FIFO 下溢中断状态。PCM 模式下, PCM 发送 FIFO 下溢标志。 0: 未产生中断; 1: 已产生中断。	0x0
[3]	RO	rx_left_fifo_overflow	I2S 模式下, 左声道接收 FIFO 上溢中断状态。PCM 模式下无效。 0: 未产生中断; 1: 已产生中断。	0x0
[2]	RO	rx_right_fifo_overflow	I2S 模式下, 右声道接收 FIFO 上溢中断状态。PCM 模式下, PCM 接收 FIFO 上溢标志。 0: 未产生中断; 1: 已产生中断。	0x0
[1]	RO	tx_intr	发送 FIFO 低于阈值中断状态。 0: 未产生中断; 1: 已产生中断。	0x0
[0]	RO	rx_intr	接收 FIFO 高于阈值中断状态。 0: 未产生中断; 1: 已产生中断。	0x0

SIO_INTCLR

SIO_INTCLR 为中断清除寄存器。

Offset Address: 0x0048 Total Reset Value: 0x0000_0000

Bits	Access	Name	Description	Reset
[31:6]	-	reserved	保留。	0x00000000
[5]	WO	tx_left_fifo_under_clr	I2S 模式下, 左声道发送 FIFO 下溢中断清除。PCM 模式下无效。 0: 不清除; 1: 清除。	0x0
[4]	WO	tx_right_fifo_under_clr	I2S 模式下, 右声道发送 FIFO 下溢中断清除。PCM 模式下, PCM 发送 FIFO 下溢中断清除。 0: 不清除; 1: 清除。	0x0
[3]	WO	rx_left_fifo_overflow_clr	I2S 模式下, 左声道接收 FIFO 上溢中断清除。PCM 模式下无效。 0: 不清除; 1: 清除。	0x0
[2]	WO	rx_right_fifo_overflow_clr	I2S 模式下, 右声道接收 FIFO 上溢中断清除。PCM 模式下, PCM 接收 FIFO 上溢中断清除。 0: 不清除; 1: 清除。	0x0
[1]	WO	tx_intr_clr	发送 FIFO 低于阈值中断清除。 0: 不清除; 1: 清除。	0x0
[0]	WO	rx_intr_clr	接收 FIFO 高于阈值中断清除。 0: 不清除;	0x0

Bits	Access	Name	Description	Reset
			1: 清除。	

SIO_I2S_LEFT_XD

SIO_I2S_LEFT_XD 为左声道发送数据。

Offset Address: 0x004C Total Reset Value: 0x0000_0000

Bits	Access	Name	Description	Reset
[31:0]	WO	tx_left_data	I2S 左声道发送数据。	0x00000000

SIO_I2S_RIGHT_XD

SIO_I2S_RIGHT_XD 为右声道发送数据。

Offset Address: 0x0050 Total Reset Value: 0x0000_0000

Bits	Access	Name	Description	Reset
[31:0]	WO	tx_right_data	I2S 模式[31:0]为右声道发送数据; PCM 模式[15:0]为 PCM 发送数据。	0x00000000

SIO_I2S_LEFT_RD

SIO_I2S_LEFT_RD 为左声道接收数据。

Offset Address: 0x0054 Total Reset Value: 0x0000_0000

Bits	Access	Name	Description	Reset
[31:0]	RO	rx_left_data	I2S 左声道接收数据。	0x00000000

SIO_I2S_RIGHT_RD

SIO_I2S_RIGHT_RD 为右声道接收数据。

Offset Address: 0x0058 Total Reset Value: 0x0000_0000

Bits	Access	Name	Description	Reset
[31:0]	RO	rx_right_data	I2S 模式[31:0]为右声道接收数据； PCM 模式[15:0]为 PCM 接收数据。	0x00000000

SIO_CT_SET

SIO_CT_SET 为 I2S_CT 置位寄存器。

Offset Address: 0x005C Total Reset Value: 0x0000_8000

Bits	Access	Name	Description	Reset
[31:16]	-	reserved	保留。	0x0000
[15]	RW	rst_n	I2S/PCM 通道复位，低电平有效。 它对 I2S/PCM 接收和发送模块（包括 FIFO）进行复位，因此发送和接收 FIFO 状态寄存器会变为 0。但不会复位 CPU 接口寄存器模块。 复位置位寄存器： 1：复位寄存器置位，即复位不生效； 0：复位寄存器不置位，复位保持原始状态。 和 SIO_CT_CLR 寄存器搭配使用。	0x1
[14]	RW	intr_en	中断全局使能置位寄存器。 0：不置位（保持原状态）；	0x0

Bits	Access	Name	Description	Reset
			1: 置位 (使能)。 和 SIO_CT_CLR 寄存器搭配使用。	
[13]	RW	rx_enable	接收通道使能置位寄存器。 0: 不置位 (保持原状态) ; 1: 置位 (使能)。 和 SIO_CT_CLR 寄存器搭配使用。	0x0
[12]	RW	tx_enable	发送通道使能置位寄存器。 0: 不置位 (保持原状态) ; 1: 置位 (使能)。 和 SIO_CT_CLR 寄存器搭配使用。	0x0
[11]	RW	rx_fifo_disable	接收 FIFO 禁止置位寄存器。 0: 不置位 (保持原状态) ; 1: 置位 (禁止)。 和 SIO_CT_CLR 寄存器搭配使用。	0x0
[10]	RW	tx_fifo_disable	发送 FIFO 禁止置位寄存器。 0: 不置位 (保持原状态) ; 1: 置位 (禁止)。 和 SIO_CT_CLR 寄存器搭配使用。	0x0
[9]	RW	rx_data_merge_en	数据接收合并使能置位寄存器, 只在 I2S 模式下且数据位宽为 16bit 时有效。 0: 不置位 (保持原状态) ; 1: 置位 (使能)。 和 SIO_CT_CLR 寄存器搭配使用。	0x0

Bits	Access	Name	Description	Reset
			用。 数据接收合并使能寄存器使能时，左右声道数据拼成一个 32bit 数据在 FIFO 中进行存储，左声道 16bit 数据占据高 16 位，右声道 16bit 数据占据低 16 位。这样可以提高 FIFO 的利用率，以提供更长时间的缓冲能力。 CPU 从接收 FIFO 中读数据时，按照如下顺序：从左声道 FIFO 读出一个 32bit 数据（左右声道各 16bit 合并成的一个 32bit 数据），再从右声道 FIFO 读出一个 32bit 数据，如此反复。	
[8]	RW	tx_data_merge_en	数据发送合并使能置位寄存器，只在 I2S 模式下且数据位宽为 16bit 时有效。 0：不置位（保持原状态）； 1：置位（使能）。 和 SIO_CT_CLR 寄存器搭配使用。 数据发送使能寄存器使能时，左右声道数据拼成一个 32bit 数据在 FIFO 中进行存储，左声道 16bit 数据占据高 16 位，右声道 16bit 数据占据低 16 位。这样可以提高 FIFO 的利用率，以提供更长时间的缓冲能力。 在这种情况下，CPU 向发送 FIFO 中写入数据时，按照如下顺序：向左声道 FIFO 写入一个 32bit 数据（左右声道各 16bit 合并成的一个 32bit 数据），再向右声道 FIFO 写	0x0

Bits	Access	Name	Description	Reset
			入一个 32bit 数据，如此反复。	
[7:4]	-	reserved	保留。	0x0
[3:0]	-	reserved	保留。	0x0

SIO_CT_CLR

SIO_CT_CLR 为 I2S_CT 清除寄存器。

Offset Address: 0x0060 Total Reset Value: 0x0000_8000

Bits	Access	Name	Description	Reset
[31:16]	-	reserved	保留。	0x0000
[15]	RW	rst_n_clr	I2S/PCM 通道复位，低电平有效。 它对 I2S/PCM 接收和发送模块（包括 FIFO）进行复位，因此发送和接收 FIFO 状态寄存器会变为 0。但不会复位 CPU 接口寄存器模块。 复位清除寄存器： 1：复位寄存器清除，即复位生效； 0：复位寄存器不清除，复位保持原始状态。	0x1
[14]	RW	intr_en_clr	中断全局使能清除寄存器。 1：清除（禁止）； 0：不清除（保持原始状态）。	0x0
[13]	RW	rx_enable_clr	接收通道使能清除寄存器。 1：清除（禁止）； 0：不清除（保持原始状态）。	0x0

Bits	Access	Name	Description	Reset
[12]	RW	tx_enable_clr	发送通道使能清除寄存器。 1: 清除 (禁止) ; 0: 不清除 (保持原始状态) 。	0x0
[11]	RW	rx_fifo_disable_clr	接收 FIFO 禁止清除寄存器。 1: 清除 (使能) ; 0: 不清除 (保持原始状态) 。	0x0
[10]	RW	tx_fifo_disable_clr	发送 FIFO 禁止清除寄存器。 1: 清除 (使能) ; 0: 不清除 (保持原始状态) 。	0x0
[9]	RW	rx_data_merge_en_clr	数据接收合并使能清除寄存器, 只在 I2S 模式下且数据位宽为 16bit 时有效。 1: 清除 (禁止) ; 0: 不清除 (保持原始状态) 。	0x0
			数据接收合并使能寄存器使能时, 左右声道数据拼成一个 32bit 数据在 FIFO 中进行存储, 左声道 16bit 数据占据高 16 位, 右声道 16bit 数据占据低 16 位。这样可以提高 FIFO 的利用率, 以提供更长时间的缓冲能力。 CPU 从接收 FIFO 中读数据时, 按照如下顺序: 从左声道 FIFO 读出一个 32bit 数据 (左右声道各 16bit 合并成的一个 32bit 数据), 再从右声道 FIFO 读出一个 32bit 数据, 如此反复。	
[8]	RW	tx_data_merge_en_clr	数据发送合并使能清除寄存器, 只在 I2S 模式下且数据位宽为 16bit 时有效。	0x0

Bits	Access	Name	Description	Reset
			1: 清除 (禁止) ; 0: 不清除 (保持原始状态)。 数据发送合并使能寄存器使能时, 左右声道数据拼成一个 32bit 数据在 FIFO 中进行存储, 左声道 16bit 数据占据高 16 位, 右声道 16bit 数据占据低 16 位。这样可以提高 FIFO 的利用率, 以提供更长时间的缓冲能力。 在这种情况下, CPU 向发送 FIFO 中写入数据时, 按照如下顺序: 向左声道 FIFO 写入一个 32bit 数据 (左右声道各 16bit 合并成的一个 32bit 数据), 再向右声道 FIFO 写入一个 32bit 数据, 如此反复。	
[7:4]	-	reserved	保留。	0x0
[3:0]	-	reserved	保留。	0x0

SIO_FIFO_THRESHOLD

SIO_FIFO_THRESHOLD 为 FIFO 阈值配置寄存器。

Offset Address: 0x0064 Total Reset Value: 0x0000_0000

Bits	Access	Name	Description	Reset
[31:16]	-	reserved	保留。	0x0000
[15:8]	RW	rx_fifo_threshold	接收 FIFO 阈值。 当 rx_right_depth >= (rx_fifo_threshold + 1) 时, 报接收中断和 DMA 请求。	0x00
[7:0]	RW	tx_fifo_threshold	发送 FIFO 阈值。	0x00

Bits	Access	Name	Description	Reset
			当 tx_right_depth < (tx_fifo_threshold+1) 时, 报发送中断和 DMA 请求。	

SIO_RX_STA

SIO_RX_STA 为 RX_FIFO 深度指示。

Offset Address: 0x0068 Total Reset Value: 0x0000_0000

Bits	Access	Name	Description	Reset
[31:16]	-	reserved	保留。	0x0000
[15:8]	RO	rx_left_depth	左声道接收 FIFO 深度指示。 只在 I2S 模式下有效。	0x00
[7:0]	RO	rx_right_depth	I2S 模式下, 为右声道接收 FIFO 深度指示。 PCM 模式下, 为 PCM 接收 FIFO 深度指示。	0x00

SIO_TX_STA

SIO_TX_STA 为 TX_FIFO 深度指示。

Offset Address: 0x006C Total Reset Value: 0x0000_0000

Bits	Access	Name	Description	Reset
[31:16]	-	reserved	保留。	0x0000
[15:8]	RO	tx_left_depth	左声道发送 FIFO 深度指示。 只在 I2S 模式下有效。	0x00
[7:0]	RO	tx_right_depth	I2S 模式下, 为右声道发送 FIFO	0x00

Bits	Access	Name	Description	Reset
			深度指示。 PCM 模式下，为 PCM 发送 FIFO 深度指示。	

SIO_DATA_WIDTH_SET

SIO_DATA_WIDTH_SET 为传输数据长度配置

Offset Address: 0x0078 Total Reset Value: 0x0000_0009

Bits	Access	Name	Description	Reset
[31:6]	-	reserved	保留。	0x00000000
[5:3]	RW	rx_mode	接收数据长度配置位。 000: 8bit; 001: 16bit; 010: 18bit; 011: 20bit; 100: 24bit; 101: 32bit; 110~111: 保留。 对于单路接收场景： 1. I2S 模式，支持 16/18/20/24/32bit; 2. PCM 模式，支持 8/16bit。 对于多路接收，I2S 与 PCM 两种模式只支持 8/16bit。	0x1
[2:0]	RW	tx_mode	发送数据长度配置位。 对于 I2S 模式 000: 保留; 001: 16bit;	0x1

Bits	Access	Name	Description	Reset
			010: 18bit; 011: 20bit; 100: 24bit; 101: 32bit; 110~111: 保留。 对于 PCM 模式: 000: 8bit 001: 16bit; 010: 32bit; 011: 64bit; 100: 128bit; 101~111: 保留。	

SIO_I2S_START_POS

SIO_I2S_START_POS 为声道访问配置寄存器

Offset Address: 0x007C Total Reset Value: 0x0000_0000

Bits	Access	Name	Description	Reset
[31:2]	-	reserved	保留。	0x00000000
[1]	RW	start_pos_write	写发送 FIFO 时: 0: 从左声道开始访问; 1: 从右声道开始访问。 仅用于交替写入发送 FIFO 的场景。	0x0
[0]	RW	start_pos_read	读接收 FIFO 时: 0: 从左声道开始访问; 1: 从右声道开始访问。	0x0

Bits	Access	Name	Description	Reset
			仅用于交替读取接受 FIFO 的场景。	

I2S_POS_FLAG

I2S_POS_FLAG 为声道访问状态寄存器。

Offset Address: 0x0080 Total Reset Value: 0x0000_0000

Bits	Access	Name	Description	Reset
[31:2]	-	reserved	保留。	0x00000000
[1]	RO	next_pos_write	写发送 FIFO 时: 0: 下一次从左声道开始访问; 1: 下一次从右声道开始访问。 仅用于交替写入发送 FIFO 的场景。	0x0
[0]	RO	next_pos_read	读接收 FIFO 时: 0: 下一次从左声道开始访问; 1: 下一次从右声道开始访问。 仅用于交替读取接受 FIFO 的场景。	0x0

SIO_SIGNED_EXT

SIO_SIGNED_EXT 为高位数据符号扩展使能寄存器。

Offset Address: 0x0084 Total Reset Value: 0x0000_0000

Bits	Access	Name	Description	Reset
[31:1]	-	reserved	保留。	0x00000000

Bits	Access	Name	Description	Reset
[0]	RW	signed_ext_en	高位数据符号扩展使能。 0：禁止； 1：使能。	0x0

SIO_I2S_POS_MERGE_EN

SIO_I2S_POS_MERGE_EN 为左右声道数据的操作地址合并使能寄存器。

Offset Address: 0x0088 Total Reset Value: 0x0000_0000

Bits	Access	Name	Description	Reset
[31:1]	-	reserved	保留。	0x00000000
[0]	RW	merge_en	在 I2S 模式下，对左右声道数据的操作地址合并使能。 0：禁止； 1：使能。	0x0

SIO_INTMASK

SIO_INTMASK 为中断屏蔽寄存器。

Offset Address: 0x008C Total Reset Value: 0x0000_0000

Bits	Access	Name	Description	Reset
[31:6]	-	reserved	保留。	0x00000000
[5]	RW	tx_left_fifo_under_mask	I2S 模式下，左声道发送 FIFO 下溢中断屏蔽。PCM 模式下无效。 0：不屏蔽； 1：屏蔽。	0x0
[4]	RW	tx_right_fifo_under_mask	I2S 模式下，右声道发送 FIFO 下	0x0

Bits	Access	Name	Description	Reset
			溢中断屏蔽。PCM 模式下，PCM 发送 FIFO 下溢中断屏蔽。 0：不屏蔽； 1：屏蔽。	
[3]	RW	rx_left_fifo_overflow_mask	I2S 模式下，左声道接收 FIFO 上溢中断屏蔽。PCM 模式下无效。 0：不屏蔽； 1：屏蔽。	0x0
[2]	RW	rx_right_fifo_overflow_mask	I2S 模式下，右声道接收 FIFO 上溢中断屏蔽。PCM 模式下，PCM 接收 FIFO 上溢中断屏蔽。 0：不屏蔽； 1：屏蔽。	0x0
[1]	RW	tx_intr_mask	发送 FIFO 低于阈值中断屏蔽。 0：不屏蔽； 1：屏蔽。	0x0
[0]	RW	rx_intr_mask	接收 FIFO 高于阈值中断屏蔽。 0：不屏蔽； 1：屏蔽。	0x0

CFG_I2S_CRG

CFG_I2S_CRG 为时钟配置寄存器。

Offset Address: 0x0090 Total Reset Value: 0x0000

Bits	Access	Name	Description	Reset
[15:5]	-	reserved	保留。	0x000
[4]	RW	cfg_i2s_fs_sel	SIO_FSClk 时钟相位需静态配置；	0x0

Bits	Access	Name	Description	Reset
			1'b1: FSCLK 反相, 波形取反; 1'b0: FSCLK 原相位。	
[3]	-	reserved	无效。	0x0
[2]	RW	cfg_i2s_bclk_sel	SIO_BCLK 时钟相位,需静态配置; 1'b1: BCLK 反相, 波形取反; 1'b0: BCLK 原相位。	0x0
[1]	RW	cfg_i2s_crg_clk_en	SIO_BCLK 时钟门控寄存器; 1: 开启 bclk; 0: 关闭 bclk。	0x0
[0]	RW	cfg_i2s_bclk_div_en	SIO 模块 bclk_div_en, Bclk 分频器使能信号。	0x0

CFG_I2S_BCLK_DIV_NUM

CFG_I2S_BCLK_DIV_NUM 为 BCLK 分频比配置寄存器。

Offset Address: 0x0094 Total Reset Value: 0x0001

Bits	Access	Name	Description	Reset
[15:7]	-	reserved	保留。	0x000
[6:0]	RW	cfg_i2s_bclk_div_num	SIO 模块 bclk 分频器的分频比。 注意：在配置分频比之前需要先配置 CFG_I2S_CRG<cfg_i2s_bclk_div_en>为 0，配置完分频比后，再配置 CFG_I2S_CRG<cfg_i2s_bclk_div_en>为 1。	0x01

CFG_I2S_FS_DIV_NUM

CFG_I2S_FS_DIV_NUM 为 FSCLK 分频器的分频比配置寄存器。

Offset Address: 0x0098 Total Reset Value: 0x0080

Bits	Access	Name	Description	Reset
[15:10]	-	reserved	保留。	0x00
[9:0]	RW	cfg_i2s_fs_div_num	SIO 模块 fsclk 分频器的分频比。	0x080

CFG_I2S_FS_DIV_RATIO_NUM

CFG_I2S_FS_DIV_RATIO_NUM 为 FSCLK 分频器的占空比配置寄存器。

Offset Address: 0x009C Total Reset Value: 0x0001

Bits	Access	Name	Description	Reset
[15:10]	-	reserved	保留。	0x00
[9:0]	RW	cfg_i2s_fs_div_ratio_num	SIO 模块 fsclk 分频器的占空比。	0x001

SIO_I2S_DUAL_RX_CHN

SIO_I2S_DUAL_RX_CHN 为交替读取接收 fifo 功能寄存器。

Offset Address: 0x00A0 Total Reset Value: 0x0000_0000

Bits	Access	Name	Description	Reset
[31:0]	RO	rx_data	接收到的数据。 结合 SIO_I2S_START_POS<start_pos_read>配置使用，写入此寄存器内容，相当于交替读取左右接收 fifo。	0x00000000

SIO_I2S_DUAL_TX_CHN

SIO_I2S_DUAL_TX_CHN 为交替写入发送 fifo 功能寄存器。

Offset Address: 0x00C0 Total Reset Value: 0x0000_0000

Bits	Access	Name	Description	Reset
[31:0]	WO	tx_data	发送的数据。 结合 SIO_I2S_START_POS<start_pos _write>配置使用，写入此寄存器内 容，相当于交替写入左右发送 fifo。	0x000000 00

6.8.6 注意事项

WS 从 MCLK 经过两个分频比 (aiao_fsclk_div 和 aiao_bclk_div) 分频而来，
 $WS = MCLK / (FSCLK_DIV \times BCLK_DIV)$

BCLK 从 MCLK 经过 aiao_bclk_div 分频而来， $BCLK = MCLK / BCLK_DIV$ 。

两个分频比的配置要注意：

- 一定要保证 WS 与采样频率一致。
- 尽量满足 “ $BCLK = 2 \times \text{采样频率} \times \text{采样位数}$ ” 的公式，如不满足，则需要保证以下
两点要求：
 - $BCLK / (2 \times \text{采样频率})$ 要大于采样位数。
 - CODEC 的 BCLK 也要与 IIS 输出的 BCLK 保持一致。

6.9 QSPI

6.9.1 概述

QSPI 是 Quad SPI 的简写，一共有 6 线组成
(SPI_CLK/SPI_CSN/SPI_D0/SPI_D1/SPI_D2/SPI_D3)，常用来对接
FLASH/PSRAM 等器件。

6.9.2 功能描述

QSPI 具有以下功能特点：

- 仅支持 Master 模式。
- 支持 1 线/4 线模式。
- 支持一个 32x16 的 TXFIFO，以及一个 32x16 的 RXFIFO。
- 1 线模式下，支持三种帧格式：
 - Motorola 帧格式。
 - TI (Texas Instruments) 帧格式。
 - National Microwire 帧格式。
- 1 线模式下，支持串行帧长度可编程。
- 支持 DMA 搬移操作。
- 支持接口时钟频率可编程：
 - 1 线模式下，最大支持 10M 的接口频率工作。
 - 4 线模式下，最大支持 32M 的接口频率工作。

6.9.3 工作方式

芯片集成了一个 QSPI 控制器，用于对接外部的 PSRAM/FLASH 等器件进行通信。

Master TX 模式

下面以对接 PSRAM 场景，举例配置流程。

1. 初始化

步骤 1 进行时钟的初始化配置，将 QSPI 的工作时钟切换到 64M。

步骤 2 参考 IO_MUX，配置好 QSPI 对应的 IO 复用，以及 IO 的驱动能力。

步骤 3 写 `SPI_ER[start_en]` 为 0x0，关闭 QSPI 的 start 使能信号。

步骤 4 写 `SPI_ENHCTL[ilen]` 为 0x2，配置 QSPI 的通信指令模式为 8bit 指令。

步骤 5 写 `SPI_ENHCTL[addrlen]` 为 0x8，配置 QSPI 的通信地址长度为 32bit。

步骤 6 写 `SPI_ENHCTL[aaif]` 为 0x2，配置 QSPI 指令/地址均以 ENHFF 定义的模式发送。

步骤 7 写 `SPI_CTR[prs]` 为 0x0，配置 QSPI 的帧格式为摩托罗拉协议。

步骤 8 写 `SPI_CTRA[scpol:scph]` 为 0x1，配置 QSPI 为 Motorola 对应的帧格式。

步骤 9 写 `SPI_CTRA[enhff]` 为 0x2，配置 QSPI 为四线通信模式。

步骤 10 写 `SPI_CTRA[trsm]` 为 0x1，配置 QSPI 为只发送模式。

步骤 11 写 `SPI_CTRA[dfs32]` 为 0x1f，配置 QSPI 的串行数据长度为 32bit。

步骤 12 写 `SPI_BRS[frdv]` 为 0x2，配置 QSPI 内部 2 分频。

步骤 13 写 `SPI_RSDR[rxdsd]` 为 0x1，配置 QSPI 内部的采样点位置。

步骤 14 写 `SPI_SLENR[ssef]` 为 0x1，打开的 QSPI 的片选能力。

步骤 15 写 `SPI_TWLR[twl]`，配置 QSPI 发送 FIFO 水线阈值。

步骤 16 写 `SPI_ER[start_en]` 为 0x1，打开 QSPI 的 start 使能信号。

----结束

2. 数据发送

步骤 1 CPU 接收到中断，读取 `SPI_INSR[tfeis]` 为 1，表征此时 QSPI 的 TXFIFO 为空。

步骤 2 写 `SPI_DRNM[datar]`，将目标数据写进 QSPI 的 TXFIFO 中。

----结束

Master RX 模式

下面以对接 PSRAM 场景，举例配置流程。

1. 初始化

步骤 1 进行时钟的初始化配置，将 QSPI 的工作时钟切换到 64M。

步骤 2 参考 IO_MUX，配置好 QSPI 对应的 IO 复用，以及 IO 的驱动能力。

步骤 3 写 `SPI_ER[start_en]` 为 0x0，关闭 QSPI 的 start 使能信号。

步骤 4 写 `SPI_ENHCTL[waitnum]`，配置 QSPI 控制帧与数据接收之间的等待时间。

步骤 5 写 `SPI_CTRA[prsr]` 为 0x0，配置 QSPI 的帧格式为摩托罗拉协议。

步骤 6 写 `SPI_CTRA[scpol:scph]` 为 0x1，配置 QSPI 为 Motorola 对应的帧格式。

步骤 7 写 `SPI_CTRA[enhff]` 为 0x2，配置 QSPI 为四线通信模式。

步骤 8 写 `SPI_CTRA[trsm]` 为 0x2，配置 QSPI 为只接收模式。

步骤 9 写 `SPI_CTRA[dfs32]` 为 0x1f，配置 QSPI 的串行数据长度为 32bit。

步骤 10 写 `SPI_CTRB[nrdf]`，配置 QSPI 需要接收的帧长度。

步骤 11 写 `SPI_BRS[frdv]` 为 0x2，配置 QSPI 内部 2 分频。

步骤 12 写 `SPI_RSDR[rxdsd]` 为 0x1，配置 QSPI 内部的采样点位置。

步骤 13 写 `SPI_SLENR[ssef]` 为 0x1，打开的 QSPI 的片选能力。

步骤 14 写 `SPI_RWLR[rwl]`，配置 QSPI 接收 FIFO 水线阈值。

步骤 15 写 `SPI_ER[start_en]` 为 0x1，打开 QSPI 的 start 使能信号。

----结束

2. 数据接收

步骤 1 CPU 接收到中断，读取 `SPI_INSR[rffis]` 为 1，表征此时 QSPI 的 RXFIFO 达到了水线。

步骤 2 读 `SPI_DRNM[datar]`，将需要的数据从 QSPI RXFIFO 中读取出来。

----结束

6.9.4 寄存器概览

SPI_CFG0_REGBANK 寄存器概览如表 6-19 所示。

表6-19 SPI_CFG0_REGBANK 寄存器概览（基址是 0x044021000）

偏移地址	名称	描述
0x0000	<code>SPI_ER</code>	SPI 使能寄存器。
0x0004	<code>SPI_CTRA</code>	SPI 控制寄存器 A。
0x0008	<code>SPI_CTRB</code>	SPI 控制寄存器 B。
0x000C	<code>SPI_ENHCTL</code>	SPI Enhanced 控制寄存器。
0x0010	<code>SPI_MCR</code>	microwire 控制寄存器。
0x0014	<code>SPI_BRS</code>	波特率控制寄存器。
0x0018	<code>SPI_DCR</code>	DMA 控制寄存器。
0x001C	<code>SPI_DRDL</code>	DMA 接收水线寄存器。

偏移地址	名称	描述
0x0020	SPI_DTDL	DMA 发送水线寄存器。
0x0024	SPI_RSDR	RXD 采样延迟寄存器。
0x0028	SPI_TDER	TXD 驱动边沿寄存器。
0x002C	SPI_DRNM	SPI 数据寄存器。
0x00BC	SPI_RAINSR	原始中断状态寄存器。
0x00C0	SPI_INSR	中断状态寄存器。
0x00C4	SPI_INMAR	中断屏蔽寄存器。
0x00C8	SPI_SLENR	Slave 使能寄存器。
0x00CC	SPI_TWLR	发送水线寄存器。
0x00D0	SPI_TLR	发送数据量寄存器。
0x00D4	SPI_TFOFIER	tx fifo 中断清除寄存器。
0x00D8	SPI_RWLR	接收水线寄存器。
0x00DC	SPI_RLR	接收数据量寄存器。
0x00E0	SPI_RFFICR	rx fifo Underflow 中断清除寄存器。
0x00E4	SPI_WSR	状态寄存器。
0x00E8	SPI_MSTICR	多 master 中断清除寄存器。
0x00EC	SPI_ID	版本信息寄存器。
0x00F0	SPI_RSVD	预留寄存器。
0x00F4	SPI_RXOICR	rx fifo Overflow 中断清除寄存器。
0x00F8	SPI_ICR	中断清除寄存器。
0x00FC	SPI_APB_CG	APB 时钟门控控制寄存器。

6.9.5 寄存器描述

SPI_ER

SPI_ER 为 SPI 使能寄存器。

Offset Address: 0x0000 Total Reset Value: 0x0000_0000

Bits	Access	Name	Description	Reset
[31:1]	-	reserved	保留。	0x00000000
[0]	RW	start_en	SPI 使能。 使能或禁用全部 SPI 操作。禁用时，所有串行传输立即挂起，tx fifo 和 rx fifo 缓存被清空。使能时，某些 SPI 控制寄存器内容不可更改。禁用时，ssi_sleep 输出被置位（延迟后），通知系统此时可安全地停止 SPI 时钟，以节省功耗。 0：禁止串行传输； 1：使能串行传输。	0x0

SPI_CTRA

SPI_CTRA 为 SPI 控制寄存器 A。

Offset Address: 0x0004 Total Reset Value: 0x0000_E004

Bits	Access	Name	Description	Reset
[31:24]	RO	spi_ctra_rsvd	保留。	0x00
[23:22]	RW	prs	帧格式。选择传输数据所使用的串行协议： 00 (MOTOROLA_SPI)：摩托罗拉协议；	0x0

Bits	Access	Name	Description	Reset
			01 (TEXAS_SSP) : TI 协议; 10 (NS_MICROWIRE) : National Microwire 协议; 11 (RESERVED) : 保留。	
[21:20]	RW	enhff	SPI 帧格式。选择发送与接收数据的数据帧格式，该寄存器仅在 SSI_SPI_MODE 设为 "Dual/Quad/Octal" 之一时才有效。 当 SSI_SPI_MODE 设为 "Dual", 10 与 11 组合无效。 当 SSI_SPI_MODE 设为 "Quad", 11 组合无效。 00 (STD_ENHFF) : 标准 SPI 帧格式; 01 (DUAL_ENHFF) : 双线 SPI 帧格式; 10 (QUAD_ENHFF) : 4 线 SPI 帧格式; 11 (OCTAL_ENHFF) : 8 线 SPI 帧格式。 注意：本芯片 QSPI 为四线，即 SSI_SPI_MODE 设为 "Quad"，故本寄存器可被配置为 2'b00 或 2'b01 或 2'b10，当配置为 2'b11 时，本寄存器将保持原值不变。	0x0
[19:18]	RW	trsm	传输模式。 选择串行通信传输模式。该寄存器不影响传输本身，只用于指示接收或发送数据是否有效。 在 transmit-only 模式，SPI 接收到	0x0

Bits	Access	Name	Description	Reset
			的外部器件发送的数据无效，数据不会存储在 rx fifo 内，且会被下个传输所覆盖。 在 receive-only 模式，发送的数据无效。第一次写入 tx fifo 后，相同的数据会在传输传输持续世间内重复发送。 在 transmit&receive 模式，发送与接收数据均有效。传输会持续到 tx fifo 为空。接收到了外部器件数据被存储于可被 host 处理器读取的 rx fifo 内。 在 eeprom-read 模式，当发送控制数据时，接收数据无效。当全部控制数据发送至 EEPROM 后，接收数据开始有效而发送数据变为无效。该模式下，tx fifo 内全部数据均被视为控制数据。该模式仅在 SPI 被配置为主器件时才有效。 00：收发模式（Transmit & Receive）； 01：只发模式（Transmit Only）； 10：只收模式（Receive Only）； 11：EEPROM 读取模式（EEPROM Read）。	
[17:13]	RW	dfs32	32bit 传输模式的数据帧长度。该寄存器仅在 SSI_MAX_XFER_SIZE 配置为 32 时生效。 当数据帧长度配置为小于 32bit，接收逻辑自动将数据向右调整，rx fifo 的高位 bit 被零填充。设计者需	0x07

Bits	Access	Name	Description	Reset
			保证发送数据写入 tx fifo 前同样经过了高位零填充处理。发送逻辑在发送数据时会忽略未使用的高位 bit。 0x3 (FRAME_04BITS) : 4bit 串行数据传输; 0x4 (FRAME_05BITS) : 5bit 串行数据传输; 0x5 (FRAME_06BITS) : 6bit 串行数据传输; 0x6 (FRAME_07BITS) : 7bit 串行数据传输; 0x7 (FRAME_08BITS) : 8bit 串行数据传输; 0x8 (FRAME_09BITS) : 9bit 串行数据传输; 0x9 (FRAME_10BITS) : 10bit 串行数据传输; 0xa (FRAME_11BITS) : 11bit 串行数据传输; 0xb (FRAME_12BITS) : 12bit 串行数据传输; 0xc (FRAME_13BITS) : 13bit 串行数据传输; 0xd (FRAME_14BITS) : 14bit 串行数据传输; 0xe (FRAME_15BITS) : 15bit 串行数据传输; 0xf (FRAME_16BITS) : 16bit 串行数据传输; 0x10 (FRAME_17BITS) : 17bit 串行数据传输;	

Bits	Access	Name	Description	Reset
			0x11 (FRAME_18BITS) : 18bit 串行数据传输; 0x12 (FRAME_19BITS) : 19bit 串行数据传输; 0x13 (FRAME_20BITS) : 20bit 串行数据传输; 0x14 (FRAME_21BITS) : 21bit 串行数据传输; 0x15 (FRAME_22BITS) : 22bit 串行数据传输; 0x16 (FRAME_23BITS) : 23bit 串行数据传输; 0x17 (FRAME_24BITS) : 24bit 串行数据传输; 0x18 (FRAME_25BITS) : 25bit 串行数据传输; 0x19 (FRAME_26BITS) : 26bit 串行数据传输; 0x1a (FRAME_27BITS) : 27bit 串行数据传输; 0x1b (FRAME_28BITS) : 28bit 串行数据传输; 0x1c (FRAME_29BITS) : 29bit 串行数据传输; 0x1d (FRAME_30BITS) : 30bit 串行数据传输; 0x1e (FRAME_31BITS) : 31bit 串行数据传输; 0x1f (FRAME_32BITS) : 32bit 串行数据传输。	
[12:9]	RO	dfs16	数据帧长度。该寄存器仅在	0x0

Bits	Access	Name	Description	Reset
			SSI_MAX_XFER_SIZE 配置为 16 时生效。若 SSI_MAX_XFER_SIZE 配置为 32，写入该寄存器无任何影响。 选择数据帧长度，当数据帧长度配置为小于 16bit，接收逻辑自动将数据向右调整，rx fifo 的高位 bit 被零填充。设计者需保证发送数据写入 tx fifo 前同样经过了高位零填充处理。发送逻辑在发送数据时会忽略未使用的高位。 0x3 (FRAME_04BITS) : 4bit 串行数据传输； 0x4 (FRAME_05BITS) : 5bit 串行数据传输； 0x5 (FRAME_06BITS) : 6bit 串行数据传输； 0x6 (FRAME_07BITS) : 7bit 串行数据传输； 0x7 (FRAME_08BITS) : 8bit 串行数据传输； 0x8 (FRAME_09BITS) : 9bit 串行数据传输； 0x9 (FRAME_10BITS) : 10bit 串行数据传输； 0xa (FRAME_11BITS) : 11bit 串行数据传输； 0xb (FRAME_12BITS) : 12bit 串行数据传输； 0xc (FRAME_13BITS) : 13bit 串行数据传输； 0xd (FRAME_14BITS) : 14bit	

Bits	Access	Name	Description	Reset
			串行数据传输; 0xe (FRAME_15BITS) : 15bit 串行数据传输; 0xf (FRAME_16BITS) : 16bit 串行数据传输。 注意: 本芯片 SSI_MAX_XFER_SIZE 配置为 32, 故此寄存器无效。	
[8:5]	RW	cfs16	控制帧长度。选择 Microwire 协议格式的控制字长度。 0x0 (SIZE_01_BIT) : 1bit 控制帧长度; 0x1 (SIZE_02_BIT) : 2bit 控制帧长度; 0x2 (SIZE_03_BIT) : 3bit 控制帧长度; 0x3 (SIZE_04_BIT) : 4bit 控制帧长度; 0x4 (SIZE_05_BIT) : 5bit 控制帧长度; 0x5 (SIZE_06_BIT) : 6bit 控制帧长度; 0x6 (SIZE_07_BIT) : 7bit 控制帧长度; 0x7 (SIZE_08_BIT) : 8bit 控制帧长度; 0x8 (SIZE_09_BIT) : 9bit 控制帧长度; 0x9 (SIZE_10_BIT) : 10bit 控制帧长度; 0xa (SIZE_11_BIT) : 11bit 控制	0x0

Bits	Access	Name	Description	Reset
			帧长度; 0xb (SIZE_12_BIT) : 12bit 控制 帧长度; 0xc (SIZE_13_BIT) : 13bit 控制 帧长度; 0xd (SIZE_14_BIT) : 14bit 控制 帧长度; 0xe (SIZE_15_BIT) : 15bit 控制 帧长度; 0xf (SIZE_16_BIT) : 16bit 控制 帧长度。	
[4]	RW	scpol	选择时钟极性。 仅当帧格式 (SPI_CTRA[prs]) 设 为摩托罗拉协议时有效。用于选择 非激活状态下串行时钟的极性 (当 SPI master 未传输数据时)。 0 (SCLK_LOW) : 非激活状态时 钟为低; 1 (SCLK_HIGH) : 非激活状态时 钟为高。	0x0
[3]	RW	scph	串行时钟相位。 仅当帧格式 (SPI_CTRA[prs]) 设 为摩托罗拉协议时有效。该寄存器 控制串行时钟与 slave 选择信号之 间的关系。 当 SCPH=0, 数据在串行时钟第一 一个边沿被采样, 当 SCPH=1, 串行 时钟在 slave 选择信号激活后开始 翻转, 数据在第二个时钟边沿被采 样。 0 (SCPH_MIDDLE) : 串行时钟	0x0

Bits	Access	Name	Description	Reset
			在第一个数据位中间翻转； 1 (SCPH_START)：串行时钟在第一个数据位的起始位置翻转。	
[2]	RW	ssn_te	Slave 选择信号翻转使能。 当时钟相位 (SPI_CTRA[scph]) 设为 0，该寄存器控制 slave 选择信号 (ss*_n) 数据传输间隙的表现。若该 1：ss*_n 信号线将在连续的数据帧间翻转且串行时钟 (sclk) 会在 ss*_n 为高时保持默认值； 0：ss*_n 信号将在数据传输期间保持为低，sclk 信号也将持续翻转。 注意：该寄存器仅在 SSI_SCPH0_SSTOGGLE 设为 1 时生效。	0x1
[1]	RW	srlt	移位寄存器环。仅用于测试。若使能，发送移位寄存器的输出将被连接至接收以为寄存器的输入。当 SPI 配置为从器件，回环模式所使用的 ss_in_n 与 ssi_clk 必须由外部器件提供。 1 (TESTING_MODE)：测试模式，Tx&Rx 移位寄存器互联； 0 (NORMAL_MODE)：常规模式。	0x0
[0]	RW	soe	Slave 输出使能。仅在 SPI 配置为从器件有效。当配置为主器件，该寄存器无任何功能。该寄存器使能或禁止 slave 器件输出的 ssi_oe_n 信号的设置。该功能在主器件以广	0x0

Bits	Access	Name	Description	Reset
			播模式发送数据时非常有效，仅有一个从器件会在主器件的 rxd 数据线上回送数据。 1 (DISABLED)：禁止 Slave 输出，ssi_oe_n 不会被激活，当 ssi_oe_n 控制 slave 器件 txd 信号的三态 buffer 时，txd 输出将呈现高阻态； 0 (ENABLED)：Slave 输出使能。	

SPI_CTRB

SPI_CTRB 为 SPI 控制寄存器 B。

Offset Address: 0x0008 Total Reset Value: 0x0000_0000

Bits	Access	Name	Description	Reset
[31:16]	RO	spi_ctrb_rsvd	保留。	0x0000
[15:0]	RW	nrdf	数据帧数量。 当 SPI_CTRA[trsm]==2'b10 或 2'b11 时，该寄存器设置 SPI 连续接收的数据帧数量。SPI 持续接收数据直到数据帧个数等于该寄存器加 1，最大允许 64KB 的连续数据接收。 注意：当 SPI 配置为从器件，传输将在 slave 被选中时一直进行，此时本寄存器无任何作用。	0x0000

SPI_ENHCTL

SPI_ENHCTL 为 SPI Enhanced 控制寄存器。

<CH>此寄存器值恒为 0。

Offset Address: 0x000C Total Reset Value: 0x0000_0000

Bits	Access	Name	Description	Reset
[31:16]	RW	spi_enhctl_rsvd	保留。	0x0000
[15]	RW	ddren	SPI DDR 使能位。该寄存器允许 SPI 在 Dual/Quad/Octal 帧格式时使用 DDR 数据模式。 0: 使能; 1: 关闭。	0x0
[14]	RW	iddren	指令阶段 DDR 使能位。该寄存器允许 SPI 传输的指令阶段使用 DDR 数据模式。 0: 使能; 1: 关闭。	0x0
[13:12]	RW	ilen	指令长度。Dual/Quad/Octal 模式下指令 bit 位宽。 00 (INST_L_0) : 0bit 无指令; 01 (INST_L_1) : 4bit 指令; 10 (INST_L_2) : 8bit 指令; 11 (INST_L_3) : 16bit 指令。	0x0
[11:8]	RW	addrlen	地址长度。该寄存器定义传输地址长度。只有当该长度的 bit 位发送至 FIFO 后传输才会开始。 0x0 (ADDR_L_0) : 0bit 地址宽度; 0x1 (ADDR_L_1) : 4bit 地址宽度;	0x0

Bits	Access	Name	Description	Reset
			0x2 (ADDR_L_2) : 8bit 地址宽度; 0x3 (ADDR_L_3) : 12bit 地址宽度; 0x4 (ADDR_L_4) : 16bit 地址宽度; 0x5 (ADDR_L_5) : 20bit 地址宽度; 0x6 (ADDR_L_6) : 24bit 地址宽度; 0x7 (ADDR_L_7) : 28bit 地址宽度; 0x8 (ADDR_L_8) : 32bit 地址宽度; 0x9 (ADDR_L_9) : 36bit 地址宽度; 0xa (ADDR_L_10) : 40bit 地址宽度; 0xb (ADDR_L_11) : 44bit 地址宽度; 0xc (ADDR_L_12) : 48bit 地址宽度; 0xd (ADDR_L_13) : 52bit 地址宽度; 0xe (ADDR_L_14) : 56bit 地址宽度; 0xf (ADDR_L_15) : 60bit 地址宽度。	
[7]	RW	rdsen	读数据选通使能位。一旦该 bit 位设为 1, SPI 将使用读数据选通 (rxds) 采样 DDR 模式的读数据。	0x0

Bits	Access	Name	Description	Reset
			0：使能； 1：关闭。	
[6:5]	RW	aaitf	地址和指令传输模式。选择 SPI 发送指令及地址数据的模式。 00：以标志 SPI 模式发送指令及地址； 01：指令以标志 SPI 模式发送，地址以 CTRLRO.ENHFF 定义的模式发送； 10：指令及地址均以 ENHFF 定义的模式发送； 11：保留。	0x0
[4:0]	RW	waitnum	等待周期。在 Dual/Quad/Octal 模式下，定义控制帧发送与数据接收之间的等待时间。寄存器的值以 SPI 时钟周期为单位。	0x00

SPI_MCR

SPI_MCR 为 microwire 控制寄存器。

Offset Address：0x0010 Total Reset Value：0x0000_0000

Bits	Access	Name	Description	Reset
[31:3]	-	reserved	保留。	0x00000000
[2]	RW	mtrc	Microwire 控制，定义 Microwire 串行协议的数据传输方向。 0：接收数据，SPI MacroCell 从外部串行器件接收数据； 1：发送数据，SPI MacroCell 向外部串行器件发送数据。	0x0

Bits	Access	Name	Description	Reset
[1]	RW	mss	Microwire 传输模式。定义 Microwire 传输是否是连续的。 0: 非连续 Microwire 传输, 发送或接收每个数均需要 1 个控制字; 1: 连续 Microwire 传输, 一个数据块发送或接收仅需要 1 个控制字。	0x0
[0]	RW	mh	Microwire 握手使能。仅在 SPI 被配置为 Master 器件时有效, 当被配置为 Slave 器件时该寄存器无意义。用于使能或禁止 Microwire 协议的 busy/ready 握手接口。当使能时, SPI 在最后 1bit 数据传输完成之后, SR 寄存器的 Busy 状态清除之前, 检查目标 Slave 器件的 ready 状态。 0: 握手接口禁用; 1: 握手接口使能。	0x0

SPI_BRS

SPI_BRS 为波特率控制寄存器。

Offset Address: 0x0014 Total Reset Value: 0x0000_0000

Bits	Access	Name	Description	Reset
[31:16]	-	reserved	保留。	0x0000
[15:0]	RW	frdv	时钟分频器。若该寄存器设为 0, SPI 时钟输出将被禁止。SPI 时钟频率按如下公式计算: $F_{sclk_out} = F_{ssi_clk} / FRDV$, 其中 FRDV 是 2 到 65534 之间的任意	0x0000

Bits	Access	Name	Description	Reset
			偶数。例如： $F_{ssi_clk}=3.6864\text{MHz}$ 、 $FRDV=2$ $F_{sclk_out}=3.6864/2 = 1.8432\text{MHz}$ 注意：该寄存器最低 bit 位被强制设为 0，不受寄存器写入的影响，这保证该寄存器的值永远是偶数。	

SPI_DCR

SPI_DCR 为 DMA 控制寄存器。

Offset Address: 0x0018 Total Reset Value: 0x0000_0000

Bits	Access	Name	Description	Reset
[31:2]	-	reserved	保留。	0x00000000
[1]	RW	rden	接收 DMA 使能。 0: 接收 DMA 禁用; 1: 接收 DMA 使能。	0x0
[0]	RW	tden	发送 DMA 使能。 0: 发送 DMA 禁用; 1: 接收 DMA 使能。	0x0

SPI_DRDL

SPI_DRDL 为 DMA 接收水线寄存器。

Offset Address: 0x001C Total Reset Value: 0x0000_0000

Bits	Access	Name	Description	Reset
[31:4]	-	reserved	保留。	0x00000000
[3:0]	RW	rdl	接收水线。该寄存器控制接收逻辑	0x0

Bits	Access	Name	Description	Reset
			在何时发起 DMA 请求。 dma_rx_req 将在 rx fifo 内数据量大于等于该寄存器加 1 时产生（且要求 SPI_DCR[rden]=1）。	

SPI_DTDL

SPI_DTDL 为 DMA 发送流水线寄存器。

Offset Address: 0x0020 Total Reset Value: 0x0000_0000

Bits	Access	Name	Description	Reset
[31:4]	-	reserved	保留。	0x00000000
[3:0]	RW	tdl	发送流水线。该寄存器控制发送逻辑在何时发起 DMA 请求。 dma_tx_req 将在 tx fifo 内有效数据量等于或小于该值时产生（且要求 SPI_DCR[tden]=1）。	0x0

SPI_RSDR

SPI_RSDR 为 RXD 采样延迟寄存器。

Offset Address: 0x0024 Total Reset Value: 0x0000_0000

Bits	Access	Name	Description	Reset
[31:8]	-	reserved	保留。	0x00000000
[7:0]	RW	rxdsd	Rxd 采样延迟。该寄存器用于延迟 rxd 输入端口的采样时间，延迟单位为 ssi_clk 周期。 注意：若该寄存器所配置的值超出内部移位寄存器深度	0x00

Bits	Access	Name	Description	Reset
			(SSI_RX_DLY_SR_DEPTH=8) , 零延迟采样将作用在 rxd 端口; 注意: 只能在 SPI_ER[start_en]=1 时配置。	

SPI_TDER

SPI_TDER 为 TXD 驱动边沿寄存器。

Offset Address: 0x0028 Total Reset Value: 0x0000_0000

Bits	Access	Name	Description	Reset
[31:8]	-	reserved	保留。	0x000000
[7:0]	RW	txdde	TXD 驱动沿。该值决定了发送数据的驱动沿。 注意: 本芯片 SPI 的该寄存器值恒为 0。	0x00

SPI_DRNM

SPI_DRNM 为 SPI 数据寄存器。

Offset Address: 0x002C Total Reset Value: 0x0000_0000

Bits	Access	Name	Description	Reset
[31:0]	RW	datar	写入/读出数据。写入该寄存器必须将数据右对齐, 读取的数据已被自动右对齐。 读操作: 读出 rx fifo 缓存的数据; 写操作: 数据存入 tx fifo。	0x00000000

SPI_RAINSR

SPI_RAINSR 为原始中断状态寄存器。

Offset Address: 0x00BC Total Reset Value: 0x0000_0000

Bits	Access	Name	Description	Reset
[31:16]	-	reserved	保留。	0x0000
[15]	RO	mmcris	多 Master 连接原始中断状态。该寄存器在 SPI 被配置为 Slave 器件时不可见： 0：中断未产生（SPI_INMAR[mmcim]作用前）； 1：中断产生（SPI_INMAR[mmcim]作用前）。	0x0
[14:9]	RO	reserved	保留。	0x00
[8]	RO	tfofris	tx fifo 溢出原始中断状态： 0：中断未产生（SPI_INMAR[tfofim]作用前）； 1：中断产生（SPI_INMAR[tfofim]作用前）。	0x0
[7]	RO	tferis	tx fifo 为空原始中断： 0：中断未产生（SPI_INMAR[tfeim]作用前）； 1：中断产生（SPI_INMAR[tfeim]作用前）。	0x0
[6:3]	RO	reserved	保留。	0x0
[2]	RO	rfufri	rx fifo 下溢原始中断状态： 0：中断未产生（SPI_INMAR[rfufim]作用前）； 1：中断产生（SPI_INMAR[rfufim]作用前）。	0x0

Bits	Access	Name	Description	Reset
[1]	RO	rfofris	rx fifo 溢出原始中断状态： 0：中断未产生 (SPI_INMAR[rfofim]作用前)； 1：中断产生 (SPI_INMAR[rfofim] 作用前)。	0x0
[0]	RO	rffris	rx fifo 为满原始中断状态： 0：中断未产生 (SPI_INMAR[rffim]作用前)； 1：中断产生 (SPI_INMAR[rffim] 作用前)。	0x0

SPI_INSR

SPI_INSR 为中断状态寄存器。

Offset Address: 0x00C0 Total Reset Value: 0x0000_0000

Bits	Access	Name	Description	Reset
[31:16]	RO	spi_insr_rsvd0	保留。	0x0000
[15]	RO	mmcis	多 Master 连接中断状态。该寄存器在 SPI 被配置为 Slave 器件时不可见： 0：中断未产生 (SPI_INMAR[mmcim]作用后)； 1:中断产生 (SPI_INMAR[mmcim] 作用后)。	0x0
[14:9]	RO	reserved	保留。	0x00
[8]	RO	tfofis	tx fifo 溢出中断状态： 0：中断未产生 (SPI_INMAR[tfofim]作用后)；	0x0

Bits	Access	Name	Description	Reset
			1: 中断产生 (SPI_INMAR[tfofim]作用后)。	
[7]	RO	tfeis	tx fifo 为空中断: 0: 中断未产生 (SPI_INMAR[tfeim]作用后) ; 1: 中断产生 (SPI_INMAR[tfeim]作用后)。	0x0
[6:3]	RO	reserved	保留。	0x0
[2]	RO	rfufis	rx fifo 下溢中断状态: 0: 中断未产生 (SPI_INMAR[rfufim]作用后) ; 1: 中断产生 (SPI_INMAR[rfufim]作用后)。	0x0
[1]	RO	rfofis	rx fifo 溢出中断状态: 0: 中断未产生 (SPI_INMAR[rfofim]作用后) ; 1: 中断产生 (SPI_INMAR[rfofim]作用后)。	0x0
[0]	RO	rffis	rx fifo 为满中断状态: 0: 中断未产生 (SPI_INMAR[rffim]作用后) ; 1: 中断产生 (SPI_INMAR[rffim]作用后)。	0x0

SPI_INMAR

SPI_INMAR 为中断屏蔽寄存器。

Offset Address: 0x00C4 Total Reset Value: 0x0000_8187

Bits	Access	Name	Description	Reset
[31:16]	-	reserved	保留。	0x0000
[15]	RW	mmcim	多 Master 连接中断屏蔽。该寄存器在 SPI 被配置为 Slave 器件时不可见。 0：中断屏蔽； 1：中断未屏蔽。	0x1
[14:9]	RO	reserved	保留。	0x00
[8]	RW	tfofim	tx fifo 溢出中断屏蔽： 0：中断屏蔽； 1：中断未屏蔽。	0x1
[7]	RW	tfeim	tx fifo 为空中断屏蔽： 0：中断屏蔽； 1：中断未屏蔽。	0x1
[6:3]	RO	reserved	保留。	0x0
[2]	RW	rfufim	rx fifo 下溢原始中断屏蔽： 0：中断屏蔽； 1：中断未屏蔽。	0x1
[1]	RW	rfofim	rx fifo 溢出中断屏蔽： 0：中断屏蔽； 1：中断未屏蔽。	0x1
[0]	RW	rffim	rx fifo 为满中断屏蔽： 0：中断屏蔽； 1：中断未屏蔽。	0x1

SPI_SLENR

SPI_SLENR 为 Slave 使能寄存器。

Offset Address: 0x00C8 Total Reset Value: 0x0000_0000

Bits	Access	Name	Description	Reset
[31:1]	-	reserved	保留。	0x00000000
[0]	RW	ssef	Slave 选择使能标志。该寄存器代表 Slave 选择信号。若该寄存器设为 1，相应的 Slave 选择信号被激活。启动一个传输前，用户需要使能该寄存器。 0：无从器件选择； 1：从器件被选择。 注意：置位或清零该寄存器并不影响相应的 Slave 选择输出信号，直到一个传输开始。	0x0

SPI_TWLR

SPI_TWLR 为发送水线寄存器。

Offset Address: 0x00CC Total Reset Value: 0x0000_0000

Bits	Access	Name	Description	Reset
[31:4]	-	reserved	保留。	0x00000000
[3:0]	RW	twl	tx fifo 水线配置寄存器。控制 tx fifo 产生中断的水线，FIFO 深度被配置为 2-64。当 tx fifo 内数据量小于等于该寄存器值，tx fifo 空中断将被触发。 注意：若用户向该寄存器写入的值大于或等于 fifo 深度（64），则该寄存器将不可写入并保持当前值；	0x0

SPI_TLR

SPI_TLR 为发送数据量寄存器。

Offset Address: 0x00D0 Total Reset Value: 0x0000_0000

Bits	Access	Name	Description	Reset
[31:5]	-	reserved	保留。	0x0000000
[4:0]	RO	tfl	tx fifo 内有效数据个数指示寄存器。	0x00

SPI_TFOFIER

SPI_TFOFIER 为 tx fifo 中断清除寄存器。

Offset Address: 0x00D4 Total Reset Value: 0x0000_0000

Bits	Access	Name	Description	Reset
[31:1]	-	reserved	保留。	0x00000000
[0]	RC	ctfofi	清除 tx fifo 溢出中断寄存器。该寄存器反映中断状态，读该寄存器将清除 ssi_txo_intr 中断，写该寄存器无效。 0: tx fifo 溢出中断无效; 1: tx fifo 溢出中断有效。	0x0

SPI_RWLR

SPI_RWLR 为接收水线寄存器。

Offset Address: 0x00D8 Total Reset Value: 0x0000_0000

Bits	Access	Name	Description	Reset
[31:4]	-	reserved	保留。	0x0000000

Bits	Access	Name	Description	Reset
[3:0]	RW	rwl	rx fifo 水线配置寄存器。控制 rx fifo 产生中断的水线位置。当 rx fifo 内数据量大于或等于该寄存器值加 1，rx fifo 的满中断将被触发。 注意：若用户向该寄存器写入的值大于或等于 FIFO 深度，则该寄存器将不可写入并保持当前值。	0x0

SPI_RLR

SPI_RLR 为接收数据量寄存器。

Offset Address: 0x00DC Total Reset Value: 0x0000_0000

Bits	Access	Name	Description	Reset
[31:5]	-	reserved	保留。	0x00000000
[4:0]	RO	rfl	rx fifo 内有效数据量指示寄存器。	0x00

SPI_RFFICR

SPI_RFFICR 为 rx fifo Underflow 中断清除寄存器。

Offset Address: 0x00E0 Total Reset Value: 0x0000_0000

Bits	Access	Name	Description	Reset
[31:1]	-	reserved	保留。	0x00000000
[0]	RC	crfufi	清除 rx fifo 下溢中断。该寄存器反映中断状态，读该寄存器将清除 ssi_rxu_intr 中断，写该寄存器无效。 0: rx fifo 下溢中断无效；	0x0

Bits	Access	Name	Description	Reset
			1: rx fifo 下溢中断有效。	

SPI_WSR

SPI_WSR 为状态寄存器。

Offset Address: 0x00E4 Total Reset Value: 0x0000_1800

Bits	Access	Name	Description	Reset
[31:16]	-	reserved	保留。	0x0000
[15]	RO	sbfi	SSI 忙标志。 0: SPI 空闲或禁用; 1: SPI 正在传输数据。	0x0
[14:13]	RO	reserved	保留。	0x0
[12]	RO	tfe	tx fifo 空标志。当 tx fifo 彻底为空时, 该 bit 位置位。当 tx fifo 内存存在一个或更多的有效数据, 该 bit 位清零。该 bit 位并不产生中断。 0: tx fifo 非空; 1: tx fifo 为空。	0x1
[11]	RO	tfnf	tx fifo 非满标志。当 tx fifo 内存在一个或多个空存储空间时, 该 bit 位置位; 当 FIFO 为满时该 bit 位清零。 0: tx fifo 为满; 1: tx fifo 非满。	0x1
[10:6]	RO	reserved	保留。	0x00
[5]	RO	rffe	rx fifo 满标志。当 rx fifo 彻底满时, 该 bit 位置位。当 rx fifo 仍有 1 个或更多的存储空间该 bit 位清	0x0

Bits	Access	Name	Description	Reset
			零。 0 (NOT_FULL) : rx fifo 非满; 1 (FULL) : rx fifo 满。	
[4]	RO	rfne	rx fifo 非空标志。当 rx fifo 内存在 1 个或更多有效数据, 该 bit 位置位; 当 rx fifo 为空时, 该 bit 位清零。该 bit 位可被软件轮询来确定 rx fifo 彻底变空。 0 : rx fifo 空; 1: rx fifo 非空。	0x0
[3:1]	RO	reserved	保留。	0x0
[0]	RO	dcfe_tfee	MASTER 模式下: 数据冲突错误。 当 SPI master 在执行数据传输, ss_in_n 信号被其他 master 驱动时, 该 bit 位置位, 通知处理器最后一次数据传输在完成前被挂起。该 bit 位在读取后自动清零。 0: 无数据冲突; 1: 数据冲突错误产生。 SLAVE 模式下: 发送错误。 当 tx fifo 为空时, 若一个传输开始, 该 bit 位置位。该 bit 位仅在 SPI 配置为 Slave 器件时才可置位。上次传输的数据将在 txd 数据线上重复发送。该 bit 位在读取时自动清零。 0: 无错误; 1: 发送错误产生。	0x0

SPI_MSTICR

SPI_MSTICR 为多 master 中断清除寄存器。

Offset Address: 0x00E8 Total Reset Value: 0x0000_0000

Bits	Access	Name	Description	Reset
[31:1]	-	reserved	保留。	0x00000000
[0]	RC	cmmci	清除多 Master 连接中断。该寄存器反应中断状态，读该寄存器将清除 ssi_mst_intr 中断，写该寄存器无任何影响。 1: 多 Master 连接中断有效; 0: 无中断。	0x0

SPI_ID

SPI_ID 为版本信息寄存器。

Offset Address: 0x00EC Total Reset Value: 0x2021_0228

Bits	Access	Name	Description	Reset
[31:0]	RO	id	版本信息寄存器。存储十六进制格式的 IP 版本信息。	0x20210228

SPI_RSVD

SPI_RSVD 为预留寄存器。

Offset Address: 0x00F0 Total Reset Value: 0x0000_0000

Bits	Access	Name	Description	Reset
[31:11]	-	reserved	保留。	0x00000000
[10]	RO	wait_bypass_en	等待状态旁路信号。	0x0

Bits	Access	Name	Description	Reset
			0: 不等待; 1: 等待。 注意: 仅在 master 模式下有效。	
[9:0]	RO	wait_number	指令发送后等待接收数据的时间。 单位为 SPI 工作时钟 cycles。 注意: 仅在 master 模式下有效。	0x000

SPI_RXOICR

SPI_RXOICR 为 rx fifo Overflow 中断清除寄存器。

Offset Address: 0x00F4 Total Reset Value: 0x0000_0000

Bits	Access	Name	Description	Reset
[31:1]	-	reserved	保留。	0x00000000
[0]	RC	crfofi	清除 rx fifo 溢出中断。该寄存器反应中断状态，读该寄存器清除 ssi_rxo_intr 中断，写该寄存器无影响。 0: rx fifo 溢出中断无效; 1: rx fifo 溢出中断有效。	0x0

SPI_ICR

SPI_ICR 为中断清除寄存器。

Offset Address: 0x00F8 Total Reset Value: 0x0000_0000

Bits	Access	Name	Description	Reset
[31:1]	-	reserved	保留。	0x00000000
[0]	RC	cis	清除中断。该寄存器在	0x0

Bits	Access	Name	Description	Reset
			ssi_txo_intr, ssi_rxu_intr, ssi_rxo_intr 与 the ssi_mst_intr 四个中断中任意一个有效时置位。读该寄存器将清除这四个中断。写该寄存器无影响。 0: tx fifo 溢出中断, rx fifo 下溢中断, rx fifo 溢出中断, 多 master 中断均无效; 1: tx fifo 溢出中断, rx fifo 下溢中断, rx fifo 溢出中断, 多 master 中断, 任意一个生效。	

SPI_APB_CG

SPI_APB_CG 为 APB 时钟门控控制寄存器。

Offset Address: 0x00FC Total Reset Value: 0x0000_0000

Bits	Access	Name	Description	Reset
[31:16]	-	reserved	保留。	0x0000
[15:0]	RW	spi_apb_cg	总线时钟自动 CG 开关寄存器。控制 APB 总线强制门控信号, 该寄存器用作关闭 APB 总线时钟自动门控。 0xdead: 关闭自动 CG, APB 总线时钟常开; 0xbeef: 恢复自动 CG; others: 不影响自动 CG 状态。	0x0000

6.10 DMA

6.10.1 概述

直接存储器访问（DMA）方式，是一种完全由硬件执行 I/O 交换的工作方式。在这种方式中，直接存储器访问控制器（DMAC）直接在存储器和外设、外设和外设、存储器和存储器之间进行数据传输，减少处理器的干涉和开销。

DMA（Directory Memory Access）方式一般用于高速传输成组的数据。DMAC（Directory Memory Access Controller）在收到 DMA 传输请求后根据 CPU 对通道的配置启动总线主控制器，向存储器和外设发出地址和控制信号，对传输数据的个数计数，并且以中断方式向 CPU 报告传输操作的结束或错误。

6.10.2 功能描述

DMA 控制器有如下特点：

- 支持单 MASTER，支持 4 通道，每个通道可配置用于一种单向传输。
- 支持 UART0/UART1/UART2/SPI/QSPI/I2S 硬握手通道，可通过配置设为传输的源端请求或目的端请求。
- 支持四种方向的搬移场景：
 - MEMORY 到 MEMORY。
 - 外设到外设。
 - 外设到 MEMORY。
 - MEMORY 到外设。
- DMA 通道优先级固定，优先级从高到低对应的通道号依次为 0 ~ 3。
- DMAC 通道 0 ~ 通道 3 中各包含 1 个 16×32bit 的 FIFO。
- 支持总线位宽为 32bit 的 AHB 总线接口；支持一组 AHB SLAVE 接口和一组 AHB MASTER 接口。
- 外设可使用单次传输（single）和连续传输（burst）2 种 DMA 请求。
- 支持软件控制的 DMA 请求。
- 支持源地址和目的地址可分别配置为自动递增或不递增，递增步长取决于传输位宽。
- 支持分别配置源端和目的端的传输位宽：8/16/32bit。
- 支持链表 DMA 传输。

- 提供 1 个可屏蔽电平中断输出，中断可清除。
- 支持 DMA 错误和 DMA 传输完成中断屏蔽前后状态查询，及两者的组合中断状态查询。

6.10.3 工作方式

DMA 初始化配置步骤如下：

- 步骤 1 读 [DMAC_EN_CHNS](#)[en_chns]，获取空闲的通道编号 ch_num，以通道 0 为例。
- 步骤 2 写 [DMAC_CONFIG](#)[dmac_en]为 0x1，使能 DMAC。
- 步骤 3 写 [DMAC_CHN_CONTROL_0](#)[dmac_chn_en_0]为 0x0，关闭通道 0 使能。
- 步骤 4 如果需要通过软请求方式进行 DMA 搬数，则需要根据表 1 中的保留编号配置 [DMAC_BURST_REQ](#)[burst_req]、[DMAC_SINGLE_REQ](#)[single_req]。

 说明

如果通过硬请求方式进行 DMA 搬数或传输方向为 MEMORY 到 MEMORY，则忽略此步骤。

表6-20 DMA 请求接口信号描述

外设编号	外设端口	功能描述
0	reserved	保留。
1	uart0_tx	UART0 的发送信号。
2	uart0_rx	UART0 的接收信号。
3	uart1_tx	UART1 的发送信号。
4	uart1_rx	UART1 的接收信号。
5	uart2_tx	UART2 的发送信号。
6	uart2_rx	UART2 的接收信号。
7	spi_tx	SPI 的发送信号。
8	spi_rx	SPI 的接收信号。
9	qspi_tx	QSPI 的发送信号。
10	qspi_rx	QSPI 的接收信号。

外设编号	外设端口	功能描述
11	i2s_tx	I2S 的发送信号。
12	i2s_rx	I2S 的接收信号。
13~15	reserved	保留。

- 步骤 5 配置通道 0 的源地址 [DMAC_S_ADDR_0](#)[dmac_s_addr_0]和目的地址 [DMAC_D_ADDR_0](#)[dmac_d_addr_0]。
- 步骤 6 根据具体需求配置 DMA 通道 0 控制寄存器 [DMAC_CHN_CONTROL_0](#)，例如传输位宽、传输长度、Burst 长度等。
- 步骤 7 如果需要进行链表传输，则配置链式地址 [DMAC_LLI_0](#)[dmac_lli_0]。
- 步骤 8 根据具体需求配置 DMA 通道 0 配置寄存器 [DMAC_CHN_CONTROL_0](#)：
- 写[dmac_flow_ctl_0]，配置流控和传输类型。
 - 写[dmac_d_peripheral_0]，配置目的设备，配置值为表 1 中的外设编号。
 - 写[dmac_s_peripheral_0]，配置源设备，配置值为表 1 中的外设编号。
 - 写[dmac_int_tc_0]，配置完成中断屏蔽位。
 - 写[dmac_int_en_0]，配置错误中断屏蔽位。
 - 写[dmac_chn_en_0]为 0x1，启动通道 0。
- 步骤 9 若 [DMAC_CHN_CONTROL_0](#)[dmac_int_tc_0]未屏蔽，则当 DMA 通道 0 传输完成后上报完成中断，或轮询读取 [DMAC_ORI_INT_ST](#)[ori_int_trans_st]查询完成状态。
- 结束

6.10.4 寄存器概览

AHB_DMA_RB 寄存器概览如表 6-21 所示。

表6-21 AHB_DMA_RB 寄存器概览（基址是 0x4A000000）

偏移地址	名称	描述
0x0004	DMAC_INT_ST	中断状态寄存器。
0x0008	DMAC_INT_CLR	传输中断寄存器。

偏移地址	名称	描述
0x000C	DMAC_ORI_INT_ST	原始中断状态寄存器。
0x0010	DMAC_EN_CHNS	通道使能查询寄存器。
0x0014	DMAC_BURST_REQ	BURST 软件配置寄存器。
0x0018	DMAC_SINGLE_REQ	SINGLE 软件配置寄存器。
0x001C	DMAC_CONFIG	配置寄存器。
0x0020	DMAC_SYNC	同步寄存器。
0x0100	DMAC_LLI_0	通道 0 链表寄存器。
0x0120	DMAC_LLI_1	通道 1 链表寄存器。
0x0140	DMAC_LLI_2	通道 2 链表寄存器。
0x0160	DMAC_LLI_3	通道 3 链表寄存器。
0x0110	DMAC_S_ADDR_0	通道 0 源地址寄存器。
0x0130	DMAC_S_ADDR_1	通道 1 源地址寄存器。
0x0150	DMAC_S_ADDR_2	通道 2 源地址寄存器。
0x0170	DMAC_S_ADDR_3	通道 3 源地址寄存器。
0x0104	DMAC_D_ADDR_0	通道 0 目的地址寄存器。
0x0124	DMAC_D_ADDR_1	通道 1 目的地址寄存器。
0x0144	DMAC_D_ADDR_2	通道 2 目的地址寄存器。
0x0164	DMAC_D_ADDR_3	通道 3 目的地址寄存器。
0x0114	DMAC_CHN_CONTR OL_0	通道 0 控制寄存器。
0x0134	DMAC_CHN_CONTR OL_1	通道 1 控制寄存器。
0x0154	DMAC_CHN_CONTR OL_2	通道 2 控制寄存器。
0x0174	DMAC_CHN_CONTR OL_3	通道 3 控制寄存器。
0x0108	DMAC_CHN_CONFIG	通道 0 配置寄存器。

偏移地址	名称	描述
	_0	
0x0128	DMAC_CHN_CONFIG_1	通道 1 配置寄存器。
0x0148	DMAC_CHN_CONFIG_2	通道 2 配置寄存器。
0x0168	DMAC_CHN_CONFIG_3	通道 3 配置寄存器。

6.10.5 寄存器描述

DMAC_INT_ST

DMAC_INT_ST 为中断状态寄存器。

 说明

本 IP 仅支持 4 通道，i=0'3 有效，i=4'7 无效。

Offset Address: 0x0004 Total Reset Value: 0x00000000

Bits	Access	Name	Description	Reset
[31:24]	RO	reserved	保留。	0x00
[23:16]	RO	int_err_st	i=0~7 int_err_st[i]=1'b0: 通道 i 未产生错误中断（经过中断屏蔽）； int_err_st[i]=1'b1: 通道 i 产生错误中断（经过中断屏蔽）。	0x00
[15:8]	RO	int_trans_st	i=0~7 int_trans_st[i]=1'b0: 通道 i 未产生传输中断（经过中断屏蔽）； int_trans_st[i]=1'b1: 通道 i 产生传输中断（经过中断屏蔽）。	0x00
[7:0]	RO	int_st	i=0~7	0x00

Bits	Access	Name	Description	Reset
			int_st[i]=1'b0: 通道 i 未产生中断 (经过中断屏蔽); int_st[i]=1'b1: 通道 i 产生中断 (经过中断屏蔽)。	

DMAC_INT_CLR

DMAC_INT_CLR 为传输中断寄存器。

说明

本 IP 仅支持 4 通道, i=0'3 有效, i=4'7 无效。

Offset Address: 0x0008 Total Reset Value: 0x00000000

Bits	Access	Name	Description	Reset
[31:16]	RO	reserved	保留。	0x0000
[15:8]	RW	int_err_clr	i=0~7 int_err_clr[i]=1'b0: 不清除通道 i 的错误中断; int_err_clr[i]=1'b1: 清除通道 i 的错误中断。	0x00
[7:0]	RW	int_trans_clr	i=0~7 int_trans_clr[i]=1'b0: 不清除通道 i 的传输中断; int_trans_clr[i]=1'b1: 清除通道 i 的传输中断。	0x00

DMAC_ORI_INT_ST

DMAC_ORI_INT_ST 为原始中断状态寄存器。

说明

本 IP 仅支持 4 通道, i=0'3 有效, i=4'7 无效。

Offset Address: 0x000C Total Reset Value: 0x00000000

Bits	Access	Name	Description	Reset
31:16	RO	reserved	保留。	0x0000
15:8	RO	ori_int_err_st	i=0~7 ori_int_err_st[i]=1'b0: 通道 i 未产生错误中断（未经中断屏蔽）； ori_int_err_st[i]=1'b1: 通道 i 产生错误中断（未经中断屏蔽）。	0x00
7:0	RO	ori_int_trans_st	i=0~7 ori_int_trans_st[i]=1'b0: 通道 i 未产生传输中断（未经中断屏蔽）； ori_int_trans_st[i]=1'b1: 通道 i 产生传输中断（未经中断屏蔽）。	0x00

DMAC_EN_CHNS

DMAC_EN_CHNS 为通道使能查询寄存器。

说明

本 IP 仅支持 4 通道，i=0~3 有效，i=4~7 无效。

Offset Address: 0x0010 Total Reset Value: 0x00000000

Bits	Access	Name	Description	Reset
31:8	RO	reserved	保留。	0x0000000
7:0	RO	en_chns	i=0~7 en_chns[i]=1'b0: 通道 i 未使能； en_chns[i]=1'b1: 通道 i 使能。	0x00

DMAC_BURST_REQ

DMAC_BURST_REQ 为 BURST 软件配置寄存器。

Offset Address: 0x0014 Total Reset Value: 0x00000000

Bits	Access	Name	Description	Reset
31:16	RW	burst_req	i=0~15 burst_req[i]=1'b0: 无影响; burst_req[i]=1'b1: 产生 DMA burst 传输请求, 当传输结束时该寄存器中的相应位被清零。	0x0000
15:0	RW	last_burst_req	i=0~15 last_burst_req[i]=1'b0: 无影响; last_burst_req[i]=1'b1: 产生 DMA last burst 传输请求, 当传输结束时该寄存器中的相应位被清零。	0x0000

DMAC_SINGLE_REQ

DMAC_SINGLE_REQ 为 SINGLE 软件配置寄存器。

Offset Address: 0x0018 Total Reset Value: 0x00000000

Bits	Access	Name	Description	Reset
31:16	RW	single_req	i=0~15 single_req[i]=1'b0: 无影响 single_req[i]=1'b1: 产生 DMA single 传输请求, 当传输结束时该寄存器中的相应位被清零	0x0000
15:0	RW	last_single_req	i=0~15 last_single_req[i]=1'b0: 无影响	0x0000

Bits	Access	Name	Description	Reset
			last_single_req[i]=1'b1: 产生 DMA last single 传输请求, 当传输结束时该寄存器中的相应位被清零	

DMAC_CONFIG

DMAC_CONFIG 为配置寄存器。

说明

本 IP 仅支持单 master, master2 配置无效。

Offset Address: 0x001C Total Reset Value: 0x00000000

Bits	Access	Name	Description	Reset
31:3	RO	reserved	保留。	0x00000000
2	RW	dmac_m2	Master 2 endianness 配置位。 0: little endian 模式; 1: big endian 模式。	0x0
1	RW	dmac_m1	Master 1 endianness 配置位。 0: little endian 模式; 1: big endian 模式。	0x0
0	RW	dmac_en	DMAC 使能。 0: 禁止 DMAC; 1: 使能 DMAC。	0x0

DMAC_SYNC

DMAC_SYNC 为同步寄存器。

Offset Address: 0x0020 Total Reset Value: 0x00000000

Bits	Access	Name	Description	Reset
31:16	RO	reserved	保留。	0x0000
15:0	RW	damc_sync	控制是否需要请求线进行同步。 0：使能对应外设的 DMA 请求信号同步逻辑； 1：禁止对应外设的 DMA 请求信号同步逻辑。	0x0000

DMAC_LLI_0

DMAC_LLI_0 为通道 0 链表寄存器。

 说明

本 IP 仅支持单 master，master2 相关配置无效。

Offset Address: 0x0100 Total Reset Value: 0x00000000

表6-22 Table9 DMAC_LLI_0 Register Field Description

Bits	Access	Name	Description	Reset
31:2	RW	dmac_lli_0	链式地址[31:2]	0x00000000
1	RO	reserved	保留。	0x0
0	RW	dmac_lm_0	用于载入下一个链表结点的 Master。 0: Master1; 1: Master2。	0x0

DMAC_LLI_1

DMAC_LLI_1 为通道 1 链表寄存器。

 说明

本 IP 仅支持单 master，master2 相关配置无效。

Offset Address: 0x0120 Total Reset Value: 0x00000000

Bits	Access	Name	Description	Reset
31:2	RW	dmac_lli_1	链式地址[31:2]	0x00000000
1	RO	reserved	保留。	0x0
0	RW	dmac_lm_1	用于载入下一个链表结点的 Master。 0: Master1; 1: Master2。	0x0

DMAC_LLI_2

DMAC_LLI_2 为通道 2 链表寄存器。

 说明

本 IP 仅支持单 master，master2 相关配置无效。

Offset Address: 0x0140 Total Reset Value: 0x00000000

Bits	Access	Name	Description	Reset
31:2	RW	dmac_lli_2	链式地址[31:2]	0x00000000
1	RO	reserved	保留。	0x0
0	RW	dmac_lm_2	用于载入下一个链表结点的 Master。 0: Master1; 1: Master2。	0x0

DMAC_LLI_3

DMAC_LLI_3 为通道 3 链表寄存器。

 说明

本 IP 仅支持单 master，master2 相关配置无效。

Offset Address: 0x0160 Total Reset Value: 0x00000000

Bits	Access	Name	Description	Reset
31:2	RW	dmac_lli_3	链式地址[31:2]	0x00000000
1	RO	reserved	保留。	0x0
0	RW	dmac_lm_3	用于载入下一个链表结点的 Master。 0: Master1; 1: Master2。	0x0

DMAC_S_ADDR_0

DMAC_S_ADDR_0 为通道 0 源地址寄存器。

Offset Address: 0x0110 Total Reset Value: 0x00000000

Bits	Access	Name	Description	Reset
31:0	RW	dmac_s_addr_0	DMAC 通道 0 源地址寄存器。	0x00000000

DMAC_S_ADDR_1

DMAC_S_ADDR_1 为通道 1 源地址寄存器。

Offset Address: 0x0130 Total Reset Value: 0x00000000

Bits	Access	Name	Description	Reset
31:0	RW	dmac_s_addr_1	DMAC 通道 1 源地址寄存器。	0x00000000

DMAC_S_ADDR_2

DMAC_S_ADDR_2 为通道 2 源地址寄存器。

Offset Address: 0x0150 Total Reset Value: 0x00000000

Bits	Access	Name	Description	Reset
31:0	RW	dmac_s_addr_2	DMAC 通道 2 源地址寄存器。	0x00000000

DMAC_S_ADDR_3

DMAC_S_ADDR_3 为通道 3 源地址寄存器。

Offset Address: 0x0170 Total Reset Value: 0x00000000

Bits	Access	Name	Description	Reset
31:0	RW	dmac_s_addr_3	DMAC 通道 3 源地址寄存器。	0x00000000

DMAC_D_ADDR_0

DMAC_D_ADDR_0 为通道 0 目的地址寄存器。

Offset Address: 0x0104 Total Reset Value: 0x00000000

Bits	Access	Name	Description	Reset
31:0	RW	dmac_d_addr_0	DMAC 通道 0 目的地址寄存器。	0x00000000

DMAC_D_ADDR_1

DMAC_D_ADDR_1 为通道 1 目的地址寄存器。

Offset Address: 0x0124 Total Reset Value: 0x00000000

Bits	Access	Name	Description	Reset
31:0	RW	dmac_d_addr_1	DMAC 通道 1 目的地址寄存器。	0x00000000

DMAC_D_ADDR_2

DMAC_D_ADDR_2 为通道 2 目的地址寄存器。

Offset Address: 0x0144 Total Reset Value: 0x00000000

Bits	Access	Name	Description	Reset
31:0	RW	dmac_d_addr_2	DMAC 通道 2 目的地址寄存器。	0x00000000

DMAC_D_ADDR_3

DMAC_D_ADDR_3 为通道 3 目的地址寄存器。

Offset Address: 0x0164 Total Reset Value: 0x00000000

Bits	Access	Name	Description	Reset
31:0	RW	dmac_d_addr_3	DMAC 通道 3 目的地址寄存器。	0x00000000

DMAC_CHN_CONTROL_0

DMAC_CHN_CONTROL_0 为通道 0 控制寄存器。

 说明

本 IP 仅支持单 master，master2 相关配置无效。

Offset Address: 0x0114 Total Reset Value: 0x00000000

Bits	Access	Name	Description	Reset
31	RW	dmac_trans_int_0	传输完成中断使能位。该位用于决定当前链表结点是否触发传输完成中断。 0：当前链表结点不触发传输完成中断； 1：当前链表结点触发传输完成中断。	0x0
30:28	RW	dmac_prot_0	master 发出的访问保护 HPROT[2:0]信号。	0x0
27	RW	dmac_d_inc_0	目的地址递增。	0x0

Bits	Access	Name	Description	Reset
			0: 目的地址不递增; 1: 目的地址每传一个数就递增一次。 目的设备为外设时目的地址不递增; 目的设备为存储器时目的地址递增。	
26	RW	dmac_s_inc_0	源地址递增。 0: 源地址不递增; 1: 源地址每传一个数就递增一次。 源设备为外设时源地址不递增; 源设备为存储器时源地址递增。	0x0
25	RW	dmac_d_master_0	设置访问目的设备的 master。 0: 使用 Master1 作为目的设备传输; 1: 使用 Master2 作为目的设备传输。	0x0
24	RW	dmac_s_master_0	设置访问源设备的 master。 0: 使用 Master1 作为源设备传输; 1: 使用 Master2 作为源设备传输。	0x0
23:21	RW	dmac_d_width_0	目的设备传输位宽。 000: Byte (8bit) 001: Halfword (16bit) 010: Word (32bit)	0x0
20:18	RW	dmac_s_width_0	源设备传输位宽。 000: Byte (8bit) 001: Halfword (16bit)	0x0

Bits	Access	Name	Description	Reset
			010: Word (32bit)	
17:15	RW	dmac_d_bsize_0	目的设备 burst 长度。 000: 1 001: 4 010: 8 011: 16 100: 32 101: 64 110: 128 111: 256	0x0
14:12	RW	dmac_s_bsize_0	源设备 burst 长度。 000: 1 001: 4 010: 8 011: 16 100: 32 101: 64 110: 128 111: 256	0x0
11:0	RW	dmac_trans_size_0	当 DMAC 是流控制器时，通过写该寄存器可设定 DMA 传输的长度。	0x000

DMAC_CHN_CONTROL_1

DMAC_CHN_CONTROL_1 为通道 1 控制寄存器。

 说明

本 IP 仅支持单 master，master2 相关配置无效。

Offset Address: 0x0134 Total Reset Value: 0x00000000

Bits	Access	Name	Description	Reset
31	RW	dmac_trans_int_1	传输完成中断使能位。该位用于决定当前链表结点是否触发传输完成中断。 0: 当前链表结点不触发传输完成中断; 1: 当前链表结点触发传输完成中断。	0x0
30:28	RW	dmac_prot_1	master 发出的访问保护 HPROT[2:0]信号。	0x0
27	RW	dmac_d_inc_1	目的地址递增。 0: 目的地址不递增; 1: 目的地址每传一个数就递增一次。 目的设备为外设时目的地址不递增; 目的设备为存储器时目的地址递增。	0x0
26	RW	dmac_s_inc_1	源地址递增。 0: 源地址不递增; 1: 源地址每传一个数就递增一次。 源设备为外设时源地址不递增; 源设备为存储器时源地址递增。	0x0
25	RW	dmac_d_master_1	设置访问目的设备的 master。 0: 使用 Master1 作为目的设备传输; 1: 使用 Master2 作为目的设备传输。	0x0
24	RW	dmac_s_master	设置访问源设备的 master。	0x0

Bits	Access	Name	Description	Reset
		_1	0: 使用 Master1 作为源设备传输; 1: 使用 Master2 作为源设备传输。	
23:21	RW	dmac_d_width_1	目的设备传输位宽。 000: Byte (8bit) ; 001: Halfword (16bit) ; 010: Word (32bit) 。	0x0
20:18	RW	dmac_s_width_1	源设备传输位宽。 000: Byte (8bit) 001: Halfword (16bit) 010: Word (32bit)	0x0
17:15	RW	dmac_d_bsize_1	目的设备 burst 长度。 000: 1 001: 4 010: 8 011: 16 100: 32 101: 64 110: 128 111: 256	0x0
14:12	RW	dmac_s_bsize_1	源设备 burst 长度。 000: 1 001: 4 010: 8 011: 16 100: 32 101: 64 110: 128	0x0

Bits	Access	Name	Description	Reset
			111: 256	
11:0	RW	dmac_trans_size_1	当 DMAC 是流控制器时，通过写该寄存器可设定 DMA 传输的长度。	0x000

DMAC_CHN_CONTROL_2

DMAC_CHN_CONTROL_2 为通道 2 控制寄存器。

说明

本 IP 仅支持单 master，master2 相关配置无效。

Offset Address: 0x0154 Total Reset Value: 0x00000000

Bits	Access	Name	Description	Reset
31	RW	dmac_trans_int_2	传输完成中断使能位。该位用于决定当前链表结点是否触发传输完成中断。 0: 当前链表结点不触发传输完成中断; 1: 当前链表结点触发传输完成中断。	0x0
30:28	RW	dmac_prot_2	master 发出的访问保护 HPROT[2:0]信号。	0x0
27	RW	dmac_d_inc_2	目的地址递增。 0: 目的地址不递增; 1: 目的地址每传一个数就递增一次。 目的设备为外设时目的地址不递增; 目的设备为存储器时目的地址递增。	0x0
26	RW	dmac_s_inc_2	源地址递增。 0: 源地址不递增;	0x0

Bits	Access	Name	Description	Reset
			1: 源地址每传一个数就递增一次。 源设备为外设时源地址不递增; 源设备为存储器时源地址递增。	
25	RW	dmac_d_master_2	设置访问目的设备的 master。 0: 使用 Master1 作为目的设备传输; 1: 使用 Master2 作为目的设备传输。	0x0
24	RW	dmac_s_master_2	设置访问源设备的 master。 0: 使用 Master1 作为源设备传输; 1: 使用 Master2 作为源设备传输。	0x0
23:21	RW	dmac_d_width_2	目的设备传输位宽。 000: Byte (8bit) 001: Halfword (16bit) 010: Word (32bit)	0x0
20:18	RW	dmac_s_width_2	源设备传输位宽。 000: Byte (8bit) 001: Halfword (16bit) 010: Word (32bit)	0x0
17:15	RW	dmac_d_bsize_2	目的设备 burst 长度。 000: 1 001: 4 010: 8 011: 16	0x0

Bits	Access	Name	Description	Reset
			100: 32 101: 64 110: 128 111: 256	
14:12	RW	dmac_s_bsize_2	源设备 burst 长度。 000: 1 001: 4 010: 8 011: 16 100: 32 101: 64 110: 128 111: 256	0x0
11:0	RW	dmac_trans_size_2	当 DMAC 是流控制器时，通过写该寄存器可设定 DMA 传输的长度。	0x000

DMAC_CHN_CONTROL_3

DMAC_CHN_CONTROL_3 为通道 3 控制寄存器。

说明

本 IP 仅支持单 master，master2 相关配置无效。

Offset Address: 0x0174 Total Reset Value: 0x00000000

Bits	Access	Name	Description	Reset
31	RW	dmac_trans_int_3	传输完成中断使能位。该位用于决定当前链表结点是否触发传输完成中断。 0: 当前链表结点不触发传输完成中断;	0x0

Bits	Access	Name	Description	Reset
			1: 当前链表结点触发传输完成中断。	
30:28	RW	dmac_prot_3	master 发出的访问保护 HPROT[2:0]信号。	0x0
27	RW	dmac_d_inc_3	目的地址递增。 0: 目的地址不递增; 1: 目的地址每传一个数就递增一次。 目的设备为外设时目的地址不递增; 目的设备为存储器时目的地址递增。	0x0
26	RW	dmac_s_inc_3	源地址递增。 0: 源地址不递增; 1: 源地址每传一个数就递增一次。 源设备为外设时源地址不递增; 源设备为存储器时源地址递增。	0x0
25	RW	dmac_d_master_3	设置访问目的设备的 master。 0: 使用 Master1 作为目的设备传输; 1: 使用 Master2 作为目的设备传输。	0x0
24	RW	dmac_s_master_3	设置访问源设备的 master。 0: 使用 Master1 作为源设备传输; 1: 使用 Master2 作为源设备传输。	0x0
23:21	RW	dmac_d_width_3	目的设备传输位宽。	0x0

Bits	Access	Name	Description	Reset
		3	000: Byte (8bit) 001: Halfword (16bit) 010: Word (32bit)	
20:18	RW	dmac_s_width_3	源设备传输位宽。 000: Byte (8bit) 001: Halfword (16bit) 010: Word (32bit)	0x0
17:15	RW	dmac_d_bsize_3	目的设备 burst 长度。 000: 1 001: 4 010: 8 011: 16 100: 32 101: 64 110: 128 111: 256	0x0
14:12	RW	dmac_s_bsize_3	源设备 burst 长度 000: 1 001: 4 010: 8 011: 16 100: 32 101: 64 110: 128 111: 256	0x0
11:0	RW	dmac_trans_size_3	当 DMAC 是流控制器时，通过写该寄存器可设定 DMA 传输的长度。	0x000

DMAC_CHN_CONFIG_0

DMAC_CHN_CONFIG_0 为通道 0 配置寄存器。

Offset Address: 0x0108 Total Reset Value: 0x00000000

Bits	Access	Name	Description	Reset
31:17	RO	reserved	保留。	0x0000
16	RW	dmac_halt_0	Halt 位。 0: 允许 DMA 请求; 1: 忽略后来的 DMA 请求, 通道 FIFO 中的内容都被传完。	0x0
15	RO	dmac_active_0	Active 位。 0: 通道 FIFO 中没有数据; 1: 通道 FIFO 中有数据。	0x0
14	RW	dmac_lock_0	Lock 位。 0: 禁止总线上 lock 传输; 1: 使能总线上 lock 传输。	0x0
13	RW	dmac_int_tc_0	传输完成中断屏蔽位。 0: 屏蔽本通道的传输完成中断; 1: 不屏蔽本通道的传输完成中断。	0x0
12	RW	dmac_int_en_0	错误中断屏蔽位。 0: 屏蔽本通道的错误中断; 1: 不屏蔽本通道的错误中断。	0x0
11:9	RW	dmac_flow_ctl_0	流控及传输类型字段。 000: 存储器至存储器 DMAC; 001: 存储器至外设 DMAC;	0x0

Bits	Access	Name	Description	Reset
			010: 外设至存储器 DMAC; 011: 源设备至目的设备 DMAC; 100: 源设备至目的设备目的设备; 101: 存储器至外设目的设备; 110: 外设至存储器源设备; 111: 源设备至目的设备源设备。	
8:5	RW	dmac_d_peripheral_0	目的设备。该字段用于选择一个外设请求信号作为本通道的 DMA 目的设备的请求信号。 如果 DMA 传输的目的设备是存储器则该字段被忽略。	0x0
4:1	RW	dmac_s_peripheral_0	源设备。该字段用于选择一个外设请求信号作为本通道的 DMA 源设备的请求信号。 如果 DMA 传输的源设备是存储器则该字段被忽略。	0x0
0	RW	dmac_chn_en_0	通道使能位。 0: 关闭通道; 1: 启动通道。	0x0

DMAC_CHN_CONFIG_1

DMAC_CHN_CONFIG_1 为通道 1 配置寄存器。

Offset Address: 0x0128 Total Reset Value: 0x00000000

Bits	Access	Name	Description	Reset
31:17	RO	reserved	保留。	0x0000

Bits	Access	Name	Description	Reset
16	RW	dmac_halt_1	Halt 位。 0: 允许 DMA 请求; 1: 忽略后来的 DMA 请求, 通道 FIFO 中的内容都被传完。	0x0
15	RO	dmac_active_1	Active 位。 0: 通道 FIFO 中没有数据; 1: 通道 FIFO 中有数据。	0x0
14	RW	dmac_lock_1	Lock 位。 0: 禁止总线上 lock 传输; 1: 使能总线上 lock 传输。	0x0
13	RW	dmac_int_tc_1	传输完成中断屏蔽位。 0: 屏蔽本通道的传输完成中断; 1: 不屏蔽本通道的传输完成中断。	0x0
12	RW	dmac_int_en_1	错误中断屏蔽位。 0: 屏蔽本通道的错误中断; 1: 不屏蔽本通道的错误中断。	0x0
11:9	RW	dmac_flow_ctl_1	流控及传输类型字段。 000: 存储器至存储器 DMAC; 001: 存储器至外设 DMAC; 010: 外设至存储器 DMAC; 011: 源设备至目的设备 DMAC; 100: 源设备至目的设备目的设备;	0x0

Bits	Access	Name	Description	Reset
			101: 存储器至外设目的设备; 110: 外设至存储器源设备; 111: 源设备至目的设备源设备。	
8:5	RW	dmac_d_peripheral_1	目的设备。该字段用于选择一个外设请求信号作为本通道的 DMA 目的设备的请求信号。 如果 DMA 传输的目的设备是存储器则该字段被忽略。	0x0
4:1	RW	dmac_s_peripheral_1	源设备。该字段用于选择一个外设请求信号作为本通道的 DMA 源设备的请求信号。 如果 DMA 传输的源设备是存储器则该字段被忽略。	0x0
0	RW	dmac_chn_en_1	通道使能位 0: 关闭通道; 1: 启动通道。	0x0

DMAC_CHN_CONFIG_2

DMAC_CHN_CONFIG_2 为通道 2 配置寄存器。

Offset Address: 0x0148 Total Reset Value: 0x00000000

Bits	Access	Name	Description	Reset
31:17	RO	reserved	保留。	0x0000
16	RW	dmac_halt_2	Halt 位。 0: 允许 DMA 请求; 1: 忽略后来的 DMA 请求, 通道 FIFO 中的内容都被传	0x0

Bits	Access	Name	Description	Reset
			完。	
15	RO	dmac_active_2	Active 位。 0: 通道 FIFO 中没有数据; 1: 通道 FIFO 中有数据。	0x0
14	RW	dmac_lock_2	Lock 位。 0: 禁止总线上 lock 传输; 1: 使能总线上 lock 传输。	0x0
13	RW	dmac_int_tc_2	传输完成中断屏蔽位。 0: 屏蔽本通道的传输完成中断; 1: 不屏蔽本通道的传输完成中断。	0x0
12	RW	dmac_int_en_2	错误中断屏蔽位。 0: 屏蔽本通道的错误中断; 1: 不屏蔽本通道的错误中断。	0x0
11:9	RW	dmac_flow_ctl_2	流控及传输类型字段 000: 存储器至存储器 DMAC; 001: 存储器至外设 DMAC; 010: 外设至存储器 DMAC; 011: 源设备至目的设备 DMAC; 100: 源设备至目的设备目的设备; 101: 存储器至外设目的设备; 110: 外设至存储器源设备; 111: 源设备至目的设备源设	0x0

Bits	Access	Name	Description	Reset
			备。	
8:5	RW	dmac_d_peripheral_2	目的设备。该字段用于选择一个外设请求信号作为本通道的 DMA 目的设备的请求信号。 如果 DMA 传输的目的设备是存储器则该字段被忽略。	0x0
4:1	RW	dmac_s_peripheral_2	源设备。该字段用于选择一个外设请求信号作为本通道的 DMA 源设备的请求信号。 如果 DMA 传输的源设备是存储器则该字段被忽略。	0x0
0	RW	dmac_chn_en_2	通道使能位 0: 关闭通道; 1: 启动通道。	0x0

DMAC_CHN_CONFIG_3

DMAC_CHN_CONFIG_3 为通道 3 配置寄存器。

Offset Address: 0x0168 Total Reset Value: 0x00000000

Bits	Access	Name	Description	Reset
31:17	RO	reserved	保留。	0x0000
16	RW	dmac_halt_3	Halt 位。 0: 允许 DMA 请求; 1: 忽略后来的 DMA 请求, 通道 FIFO 中的内容都被传完。	0x0
15	RO	dmac_active_3	Active 位。 0: 通道 FIFO 中没有数据; 1: 通道 FIFO 中有数据。	0x0

Bits	Access	Name	Description	Reset
14	RW	dmac_lock_3	Lock 位。 0: 禁止总线上 lock 传输; 1: 使能总线上 lock 传输。	0x0
13	RW	dmac_int_tc_3	传输完成中断屏蔽位。 0: 屏蔽本通道的传输完成中断; 1: 不屏蔽本通道的传输完成中断。	0x0
12	RW	dmac_int_en_3	错误中断屏蔽位。 0: 屏蔽本通道的错误中断; 1: 不屏蔽本通道的错误中断。	0x0
11:9	RW	dmac_flow_ctl_3	流控及传输类型字段 000: 存储器至存储器 DMAC; 001: 存储器至外设 DMAC; 010: 外设至存储器 DMAC; 011: 源设备至目的设备 DMAC; 100: 源设备至目的设备目的设备; 101: 存储器至外设目的设备; 110: 外设至存储器源设备; 111: 源设备至目的设备源设备。	0x0
8:5	RW	dmac_d_peripheral_3	目的设备。该字段用于选择一个外设请求信号作为本通道的 DMA 目的设备的请求信号。 如果 DMA 传输的目的设备是	0x0

Bits	Access	Name	Description	Reset
			存储器则该字段被忽略。	
4:1	RW	dmac_s_peripheral_3	源设备。该字段用于选择一个外设请求信号作为本通道的 DMA 源设备的请求信号。 如果 DMA 传输的源设备是存储器则该字段被忽略。	0x0
0	RW	dmac_channel_en_3	通道使能位。 0: 关闭通道; 1: 启动通道。	0x0

6.11 ADC

6.11.1 概述

LSADC 为一款 SAR (Successive Approximations Register) ADC (逐次逼近型数模转换设备), 实现将模拟信号转变成数字信号的功能。

参数	最小值	典型值	最大值	单位	描述
Power supply					
AVDD18	1.71	1.8	1.89	V	模拟 1.8V 电压
AVDD3P3	2.97	3.3	3.63	V	模拟 3.3V 电压
DVDD	0.99	1.1	1.21	V	数字电源电压
LSADC 规范					
Full Scale Input	0.3	-	3.3	V	ADC 输入范围
DNL	-	±1.5	±3	LSB	差分非线性
INL	-	±2	±4	LSB	积分非线性
Resolution	-	12	-	bit	精度
Power Dissipation	-	1.3	1.5	mA	工作时功耗

参数	最小值	典型值	最大值	单位	描述
	-	32	500	μA	关机时功耗 (DVDD,AVDD33 及 AVDD18 均在 位)
	-	3.6	4.32	μA	关机时功耗 (DVDD 掉 电,AVDD33 及 AVDD18 在位)
Clock 规范					
f _{CLK}	-	-	32	MHz	输入时钟频率
Duty-Cycle	45	50	55	%	占空比
f _s	-	-	1000	Ksps	采样率

6.11.2 功能描述

LSADC 模块具有以下功能特点：

- 输入时钟 32MHz，12bit 分辨率，单通道采样率最大为 1Msps。
- 共 6 个通道，支持软件配置 0~5 任意通道使能，逻辑按通道编号先低后高发起切换，完成单通道采样并完成平均值滤波后自动进行通道切换。。
- 支持 128×17bit FIFO 用于数据缓存，数据存储格式：高 3bit 为通道编号，低 14bit 为有效数据。
- 支持对 ADC 采样数据进行平均滤波处理，平均次数支持 1（不进行平均）、2、4、8；多通道时，每个通道接收 N 个数据（平均滤波个数）再切换通道。
- 支持 FIFO 水线中断、满中断上报,ADC 忙状态、控制器 FIFO 空满状态查询。

6.11.3 工作方式

须知

1. 在配置过程中 LSADC_CTRL8[lsadc_start]和 LSADC_CTRL8[lsadc_stop]的配置必须组合配置，即在开始采样时配置 LSADC_CTRL8[lsadc_start]为 0x1，在结束本次 adc 采样时要配置 LSADC_CTRL8[lsadc_stop]为 0x1，如果在进行第一次转换结束后没有配置 LSADC_CTRL8[lsadc_stop]，在第二次 adc 转化时直接配置 LSADC_CTRL8[lsadc_start]将导致数据错误；即不支持 start->start->stop 的操作。
2. 请保证本次采样结束后（配置完 LSADC_CTRL8[stop]为 1）将 fifo 中的数据读空，否则下次采样时将有上次 adc 采样的残留数据。
3. 单通道采样率最大为 1Msps，当开启平均值 n 次滤波时，单通道采样率为 1Msps/n，此时当有 m 个通道轮询时，每个通道的采样率为 1Msps/ (n×m)。
4. 从 LSADC_CTRL9[dr]寄存器回读的数据格式：bit[16:14]代表数据对应的通道编号；bit[13:0]的值 code 为采样数据。采样数据与电压对应关系如下：
 读 EFUSE<1135>的值记为 s，读 EFUSE<1134,1120>的值记为 b，读 EFUSE<1151,1136>的值记为 k。
 如果 s 为 0，则 $V = (\text{code}/k) \times 3.6 - (b/2^{15}) \times 3.6$ ；
 如果 s 为 1，则 $V = (\text{code}/k) \times 3.6 + (b/2^{15}) \times 3.6$ 。

配置 ADC 的步骤如下：

- 步骤 1 参考 IO_MUX，将需要使用的 ADC 复用的 GPIO 管脚配置成高阻。
- 步骤 2 配置 LSADC_CTRL11[da_lsadc_rstn]为 0x1，解除模拟逻辑复位。
- 步骤 3 确认 LSADC_CTRL1[rne]寄存器是否为 0，确保 FIFO 已经为空，若不为空可从 LSADC_CTRL9[dr]寄存器将多余数据读空。以保证上次数据不会影响到这次的采样。
- 步骤 4 根据使用需求配置 LSADC_CTRL0[equ_model_sel]寄存器的值。
- 步骤 5 配置 fifo 的中断和水线：LSADC_CTRL1[rxintsize]、LSADC_CTRL2[rxim]、LSADC_CTRL2[rorm]；
- 步骤 6 配置扫描通道号 LSADC_CTRL0[ch_vld]：使能 1 个或者多个通道。
- 步骤 7 开始 adc 采样：写 LSADC_CTRL8[lsadc_start]为 1 开始 ADC 采样。
- 步骤 8 等待中断上报。
- 步骤 9 循环读取 LSADC_CTRL1[rne]的值，若为 1，则从 LSADC_CTRL9[dr]寄存器中读取 ADC 数据；若为 0，则跳转步骤 7；（可一直在步骤 7、步骤 8 循环读取 ADC 数据，若想停止采样，则跳转至步骤 9）。

- 步骤 10 停止采样：配置 [LSADC_CTRL8](#)[lsadc_stop]信号写 1。
- 步骤 11 循环读取 [LSADC_CTRL1](#)[rne]的值，若为 1，则从 [LSADC_CTRL9](#)[dr]寄存器中读取 ADC 数据；若为 0，则停止。

----结束

6.11.4 寄存器概览

I2S_DIO 寄存器概览如表 6-23 所示。

表6-23 ADC_CTL_REG 寄存器概览（基址是 0x4400C000）

偏移地址	名称	描述
0x000	LSADC_CTRL0	ADC 控制寄存器。
0x004	LSADC_CTRL1	ADC FIFO 寄存器。
0x008	LSADC_CTRL2	ADC 中断寄存器。
0x00C	LSADC_CTRL3	ADC 中断清除寄存器。
0x010	LSADC_CTRL4	monitor1 中断寄存器。
0x014	LSADC_CTRL6	monitor2 中断寄存器。
0x018	LSADC_CTRL7	门限告警使能与上报寄存器。
0x01C	LSADC_CTRL8	ADC 开始与停止扫描控制寄存器。
0x020	LSADC_CTRL9	ADC FIFO 读数据控制寄存器。
0x0024	LSADC_CTRL11	ADC 使能寄存器。
0x0028	LSADC_CTRL12	channel 手动控制选择寄存器。

6.11.5 寄存器描述

LSADC_CTRL0

LSADC_CTRL0 为 ADC 控制寄存器。

Offset Address: 0x000 Total Reset Value: 0x0401E500

Bits	Access	Name	Description	Reset
[31:28]	RO	reserved	保留。	0x0
[27:21]	RW	cast_cnt	切换通道后丢弃的采样次数。 0x0: 0 次; 0x1: 1 次; 0x7E: 126 次; 0x7F: 127 次; 注意: 单通道模式时需要配置为 0x0。	0x20
[20:13]	RW	start_cnt	从 start 拉高到发起下一次采样的延迟。 0x0: 保留。 0x1: 保留。 0x2: 保留。 0x3: 保留。 0x4: 4 周期延迟; 0x5: 5 周期延迟; 0xFE: 254 周期延迟; 0xFF: 255 周期延迟; 注意: 配置不能小于 0x4。	0x0F
[12:8]	RW	sample_cnt	从 sample 拉高到 start 拉高的时间计数。 0x0: 保留。 0x1: 保留。 0x2: 保留。 0x3: 保留。 0x4: 4 周期延迟;	0x05

Bits	Access	Name	Description	Reset
			0x5: 5 周期延迟; 0x1E: 30 周期延迟; 0x1F: 31 周期延迟; 注意: 配置不能小于 0x4。	
[7:6]	RW	equ_model_sel	平均算法模式选择。 00: 1 次平均 (即不进行平均) ; 01: 2 次平均算法模式; 10: 4 次平均算法模式; 11: 8 次平均算法模式。	0x0
[5:0]	RW	ch_vld	每个通道是否有效。 bit[0] ~ bit[5]分别对应通道 A ~ 通道 F。 每 bit 对应的取值含义如下: 0: 无效; 1: 有效。	0x00

LSADC_CTRL1

LSADC_CTRL1 为 ADC FIFO 寄存器。

Offset Address: 0x004 Total Reset Value: 0x00000002

Bits	Access	Name	Description	Reset
[31:6]	RO	reserved	保留。	0x00000000
[5]	RO	bsy	ADC 忙标记。 0: 空闲; 1: 忙。	0x0
[4]	RO	rff	FIFO 是否已满。	0x0

Bits	Access	Name	Description	Reset
			0: 未滿; 1: 已滿。	
[3]	RO	rne	FIFO 是否未空。 0: 已空; 1: 未空。	0x0
[2:0]	RW	rxintsize	FIFO 水线设置。 当接收 FIFO 中数据个数≥配置的字数时, RXRIS 有效。 000: 127; 001: 124; 010: 64; 011: 32; 100: 16; 101: 8; 110: 4; 111: 1。	0x2

LSADC_CTRL2

LSADC_CTRL2 为 ADC 中断寄存器。
Offset Address: 0x008 Total Reset Value: 0x00000000

Bits	Access	Name	Description	Reset
[31:12]	RO	reserved	保留。	0x00000
[11]	RO	monitor_int2_high_alarm_st	monitor_int2 高门限警告中断状态。 0: 无中断; 1: 有中断。	0x0
[10]	RO	monitor_int2_low_alarm	monitor_int2 低门限警告中断	0x0

Bits	Access	Name	Description	Reset
		m_st	状态。 0: 无中断; 1: 有中断。	
[9]	RO	monitor_int1_high_alarm_st	monitor_int1 高门限警告中断状态。 0: 无中断; 1: 有中断。	0x0
[8]	RO	monitor_int1_low_alarm_st	monitor_int1 低门限警告中断状态。 0: 无中断; 1: 有中断。	0x0
[7]	RW	monitor_int2_mask	monitor_int2 中断屏蔽位。 0: 屏蔽; 1: 不屏蔽。	0x0
[6]	RW	monitor_int1_mask	monitor_int1 中断屏蔽位。 0: 屏蔽; 1: 不屏蔽。	0x0
[5]	RO	rxris	接收 FIFO 水线中断的原始中断状态。 0: 无中断; 1: 有中断。	0x0
[4]	RO	rorris	接收溢出中断的原始中断状态。 0: 无中断; 1: 有中断。	0x0
[3]	RO	rxmis	接收 FIFO 水线中断屏蔽后的状态。 0: 无中断;	0x0

Bits	Access	Name	Description	Reset
			1: 有中断。	
[2]	RO	rormis	接收溢出中断屏蔽后的状态。 0: 无中断; 1: 有中断。	0x0
[1]	RW	rxim	水线中断屏蔽位。 0: 屏蔽; 1: 不屏蔽。	0x0
[0]	RW	rorim	FIFO 溢出中断屏蔽位。 0: 屏蔽; 1: 不屏蔽。	0x0

LSADC_CTRL3

LSADC_CTRL3 为 ADC 中断清除寄存器。

Offset Address: 0x00C Total Reset Value: 0x00000000

Bits	Access	Name	Description	Reset
31:3	RO	reserved	保留。	0x00000000
2	W1_PULSE	monitor_int2_clear	monitor_int2 中断清除位。 0: 不清除; 1: 清除。	0x0
1	W1_PULSE	monitor_int1_clear	monitor_int1 中断清除位。 0: 不清除; 1: 清除。	0x0
0	W1_PULSE	roric	FIFO 溢出中断清除位。 0: 不清除; 1: 清除。	0x0

LSADC_CTRL4

LSADC_CTRL4 为 monitor1 中断寄存器。

Offset Address: 0x010 Total Reset Value: 0x7FE00078

Bits	Access	Name	Description	Reset
31	RO	reserved	保留。	0x0
30:17	RW	monitor_int1_th_high	monitor_int1 监测告警上限,14bit 位宽,满 code 与基准电压相关(3.6V), 最终对应电压计算公式: $\text{code}/2^{14} \times 3.6$ 。	0x3FF0
16:3	RW	monitor_int1_th_low	monitor_int1 监测告警下限,14bit 位宽,满 code 与基准电压相关(3.6V), 最终对应电压计算公式: $\text{code}/2^{14} \times 3.6$ 。	0x000F
2:0	RW	monitor_int1_ch_num	monitor_int1 监测通道 num。	0x0

LSADC_CTRL6

LSADC_CTRL6 为 monitor2 中断寄存器。

Offset Address: 0x014 Total Reset Value: 0x7FE00079

Bits	Access	Name	Description	Reset
[31]	RO	reserved	保留。	0x0
[30:17]	RW	monitor_int2_th_high	monitor_int2 监测告警上限,14bit 位宽,满 code 与基准电压相关(3.6V), 最终对应电压计算公式: $\text{code}/2^{14} \times 3.6$ 。	0x3FF0
[16:3]	RW	monitor_int2_th_low	monitor_int2 监测告警下限,14bit 位宽,满 code 与基准	0x000F

Bits	Access	Name	Description	Reset
			电压相关(3.6V), 最终对应电压计算公式: $\text{code}/2^{14} \times 3.6$ 。	
[2:0]	RW	monitor_int2_c h_num	monitor_int2 监测通道 num。	0x1

LSADC_CTRL7

LSADC_CTRL7 为门限告警使能与上报寄存器。

Offset Address: 0x018 Total Reset Value: 0x00000000

Bits	Access	Name	Description	Reset
[31:30]	RO	reserve d	保留。	0x0
[29]	RW	monitor_int2_e n	monitor_int2 中断使能。 0: 禁止; 1: 使能。	0x0
[28]	RW	monitor_int1_e n	monitor_int1 中断使能。 0: 禁止; 1: 使能。	0x0
[27:14]	RO	monitor_int2_al arm_data	monitor_int2 监测告警上报 值, 14bit 位宽, 满 code 与基准 电压相关(3.6V), 最终对应电 压计算公式: $\text{code}/2^{14} \times 3.6$ 。	0x0000
[13:0]	RO	monitor_int1_al arm_data	monitor_int1 监测告警上报 值, 14bit 位宽, 满 code 与基准 电压相关(3.6V), 最终对应电 压计算公式: $\text{code}/2^{14} \times 3.6$ 。	0x0000

LSADC_CTRL8

LSADC_CTRL8 为 ADC 开始与停止扫描控制寄存器。

Offset Address: 0x01C Total Reset Value: 0x00000000

Bits	Access	Name	Description	Reset
[31:2]	RO	reserved	保留。	0x00000000
[1]	W1_PU LSE	lsadc_stop	停止自动扫描。 写 0: 无效; 写 1: 停止 LSADC 的扫描功能 (需再写 LSADC_CTRL8[lsadc_start] 才可以重新启动扫描)。	0x0
[0]	W1_PU LSE	lsadc_start	LSADC 启动信号。 写 0: 无效; 写 1: 启动 LSADC。	0x0

LSADC_CTRL9

LSADC_CTRL9 为 ADC FIFO 读数据控制寄存器。

Offset Address: 0x020 Total Reset Value: 0x00000000

Bits	Access	Name	Description	Reset
[31:17]	RO	reserved	保留。	0x0000
[16:0]	RO	dr	读取 ADC 数据。 bit[16:14]: 当前数据对应的通道编号 (通道编号 0x0 ~ 0x7 分别代表通道 A ~ 通道 H) ; bit[13:0]: 有效数据, 14bit 位宽, 值记为 code。 Code 与电压的计算公式: 读 EFUSE<1135>的值记为 s, 读 EFUSE<1134,1120>的	0x00000

Bits	Access	Name	Description	Reset
			值记为 b, 读 EFUSE<1151,1136>的值记 为 k。 如果 s 为 0, 则 $V = (code/k) \times 3.6 - (b/2^{15}) \times 3.6$; 如果 s 为 1, 则 $V = (code/k) \times 3.6 + (b/2^{15}) \times 3.6$ 。	

LSADC_CTRL11

LSADC_CTRL11 为 ADC 使能寄存器。

Offset Address: 0x0024 Total Reset Value: 0x00000000

Bits	Access	Name	Description	Reset
[31:17]	RO	reserve d	保留。	0x0000
[16]	RW	da_lsad c_rstn	adc 逻辑电路独立复位信号, 默认 0。 0: adc 逻辑复位; 1: 逻辑不复位。	0x0
[15:0]	RW	da_lsad c_en	adc 使能。	0x0000

LSADC_CTRL12

LSADC_CTRL12 为 channel 手动控制选择寄存器。

Offset Address: 0x0028 Total Reset Value: 0x00000000

Bits	Access	Name	Description	Reset
[31:4]	RO	reserve d	保留。	0x00000000
[3]	RW	lsadc_s can_ch _man_s el	adc 扫描通道号自动模式与手 动模式选择。 0: auto 模式;	0x0

Bits	Access	Name	Description	Reset
			1：手动模式。	
[2:0]	RW	lsadc_scan_ch_man	adc 手动配置模式时的扫描通道号。 000：通道 0； 001：通道 1； 010：通道 2； 011：通道 3； 100：通道 4； 101：通道 5； 110：保留， 111：保留。	0x0

7 JTAG

7.1 概述

7.2 调试接口

7.1 概述

芯片内部集成一个自研 CPU，在内部集成 Coresight 调试架构，支持基于 Coresight 的 JTAG 和 SWD（Serial Wire Debug）调试接口，通过 Coresight 的 JTAG 或 SWD 调试接口实现调试，支持 Lauterbach 仿真器和 JLINK 仿真器。

7.2 调试接口

须知

芯片调试接口默认复用为其他功能，如果需要使用调试功能，则管脚 GPIO_04 在上电时置高电平，系统复位解除后，对应管脚（GPIO_00、GPIO_09、GPIO_13、GPIO_14）即可复位为调试接口，此后 GPIO_04 管脚功能不受影响，可以作为正常管脚使用。

8 包装信息

表8-1 包装信息必填项

Device Name	Q353333N1100E&Q353333N1100
Package Type	WMQFN(HF)
Body size	5×5
出货类别	测试厂出货方式
MSL level	3
是否 ESD 敏感产品	否
是否无铅 Pb-Free	Y
是否无卤 Halogen-Free	Y
包装材料	T&R
特殊需求	Tray

表8-2 TR 必填项

满卷数量 MPQ (PCS)	4000
满数箱数量 MOQ (PCS)	20000
内圈空格 (PCS)	75
外圈空格 (PCS)	75

覆盖胶带材质	-
Hub size 内径 (intch)	7
Hub size 外径 (intch)	13
带宽 (mm)	12
Pitch (mm)	8
Pin1 方向	1

表8-3 Tray 必填项

满卷数量 MPQ (PCS)	-
满数箱数量 MOQ (PCS)	-
Tray 堆叠方式	-
Tray 阵列	-
Tray 材质 PPE/PPE+GF/MPPO	-
支撑方式 (球面/斜边/ 基板)	-
Tray 长度	-
Tray 宽度	-
Pitch X (mm)	-
Pitch Y (mm)	-

表8-4 物料信息

净重 Net weight of Single	0.06
-------------------------	------

Device (g)	
开孔方式 Feeder hole type	单边 Single

表8-5 小盒最小整数包装信息 Info of inner packing

毛重 Gross weight(KG)	1.03
包装件长 Length (MM)	357
包装件宽 Width (MM)	338
包装件高 Hight (MM)	50
包装材质 Material	纸箱

表8-6 大箱整数外包装信息 Info of outer packing

毛重 Gross weight(KG)	5.77
包装件长 Length (MM)	375
包装件宽 Width (MM)	285
包装件高 Hight (MM)	362
包装材质 Material	纸箱

图8-2 芯片卷带设计

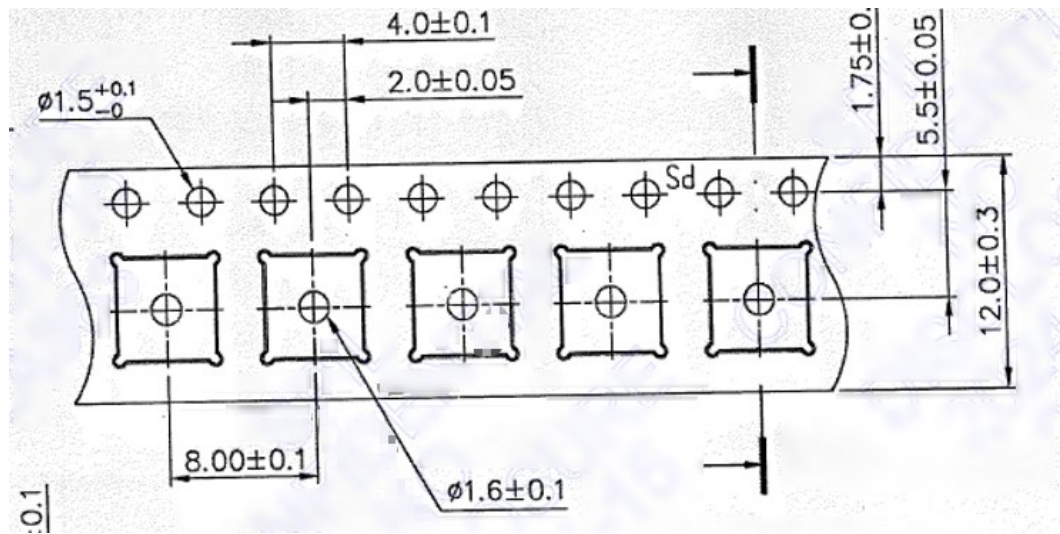


图8-3 Pin1 对应载带定位孔位置

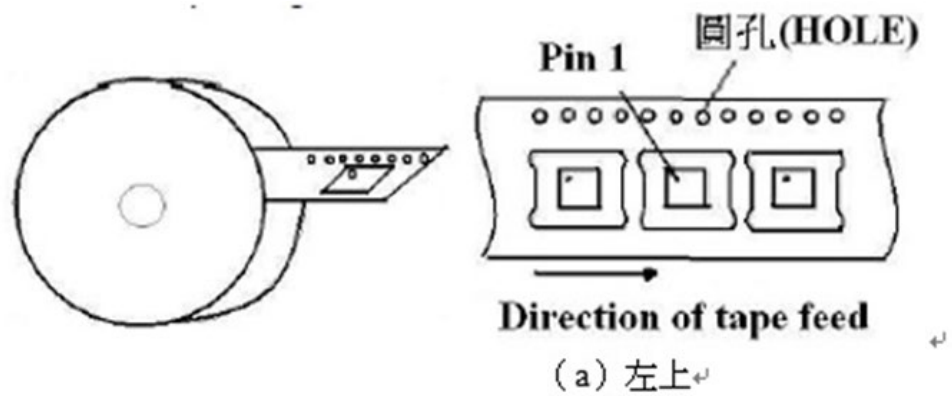


图8-4 芯片卷带包装过程 1

Tape & Reel

- 1. Transfer parts from tray into carrier tape
- 2. Combine the partial parts if customer require

Manufacture Flow :

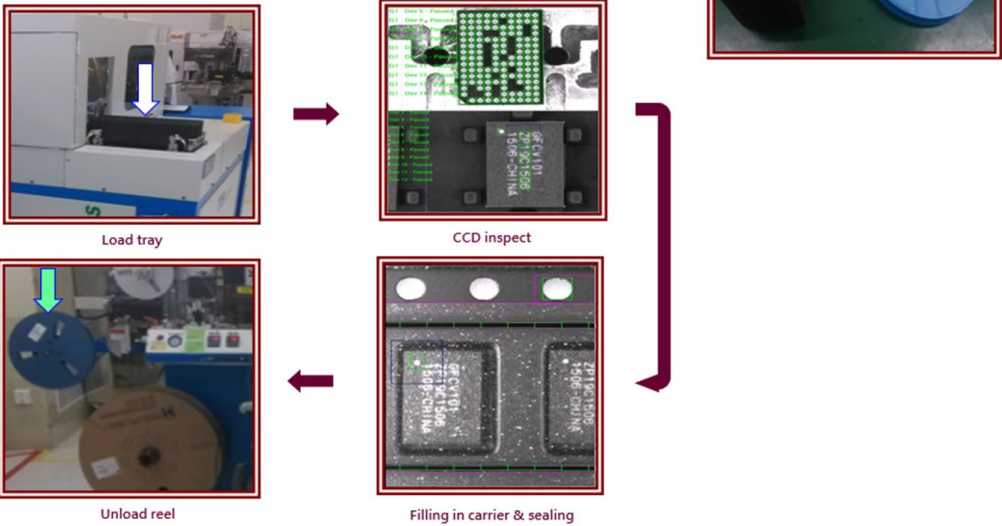


图8-5 芯片卷带包装过程 2

Packing & PSLA

- 1. Filling into MBB and Vacuuming
- 2. Packing with box and labeling
- 3. QC Check packing & label

Manufacture Flow :



图8-6 芯片卷带包装过程 3



A 缩略语

A		
ACK	Acknowledgement	通信协议中的肯定应答
ADC	Analog to Digital Converter	模数转换器
AES	Advanced Encryption Standard	高级加密标准
AGC	Automatic Gain Control	自动增益控制
A-MPDU	Aggregate MAC Protocol Data Unit	MAC 协议数据单元聚合
APB	Advanced Peripheral Bus	高级外设总线
ASIC	Application-Specific Integrated Circuit	专用集成电路
B		
BCC	Binary Convolutional Code	二进制卷积代码
BLE	Bluetooth Low Energy	低功耗蓝牙
BPSK	Binary Phase Shift Keying	二进制相移键控
BSS	Basic Service Set	基本服务集
BT	Bluetooth	蓝牙
C		
CBB	Common Building Block	通用基础模块
CBC	Cipher Block Chaining	密码分组链接

CCA	Clear Channel Assessment	空闲信道评估
CCK	Complementary Code Keying	补码键控
CMU	Clock Managing Unit	时钟管理单元
CPU	Central Processing Unit	中央处理器
CRG	Clock and Reset Generator	时钟和复位生成器
CSI	Channel State Information	信道状态信息
CTR	Counter	计算器
D		
DAC	Digital Analog Converter	数字模拟转换器
DC	Direct Current	直流电
DC	Duty Cycle	占空比
DBAC	Dual Band Adaptive Concurrent	双频自适应并发
DBB	Digital Baseband	数字基带
DMA	Direct Memory Access	直接存储器存取
DS	Drive Strength	驱动强度
DSSS	Direct Sequence Spread Spectrum	直接序列扩频
E		
EFUSE	Electrical FUSE	一次性的电可编程的 FUSE
G		
GI	Guard Interval	保护间隔
GPIO	General Purpose Input/Output	通用目的输入输出接口
H		
HT	High Throughput	高吞吐量

I

ID	Identifier	标识符
IEEE	Institute of Electrical and Electronics Engineers	电气和电子工程师学会 [美]
IO	Input Output	输入输出

J

JTAG	Joint Test Action Group	联合测试行为组织，是一种国际标准的测试协议
-------------	-------------------------	-----------------------

L

LDO	Low Dropout Regulator	低压差调节器
LNA	Low Noise Amplifier	低噪声放大器
LO	Local Oscillator	本地振荡器
LPF	Low Pass Filter	低通滤波器
LSB	Least Significant Bit	最低位
LUT	Lookup Table	查找表

M

MAC	Media Access Control	媒体接入控制
MCU	Main Control Unit	主控制器单元
MSB	Most Significant Bit	最高位

N

NMI	Non-Maskable Interrupt	不可屏蔽中断
------------	------------------------	--------

O

OFDM	Orthogonal Frequency Division	正交频分复用
-------------	-------------------------------	--------

	Multiplex	
OSC	Oscillator	振荡器
OTT	Over The Top	OTT 解决方案
P		
P2P	Peer-to-Peer	点到点
PA	Power Amplifier	功率放大器
PD	Power Down	下电
PHY	Physical Layer	物理层
PLL	Phase-Locked Loop	锁相环
PMP	Physical Memory Protection	物理内存保护
PMU	Power Management Unit	电源管理模块
PSM	Power Saving Mode	节电模式
PTA	Packet Traffic Arbitration	包流量仲裁
Q		
QFN	Quad Flat Non-leaded package	四侧无引脚扁平封装
QoS	Quality of Service	业务质量
QPSK	Quadrature Phase Shift Keying	正交相移键控
R		
RAM	Random Access Memory	随机存取存储器
RF	Radio Frequency	射频
RISC	Reduced Instruction Set Computing	精简指令集计算
RO	Ring Oscillator	振荡
ROM	Read-Only Memory	只读存储器
RSA	Rivest-Shamir-Adleman	RSA 加密算法

RSSI	Received Signal Strength Indicator	接收信号强度指示
RTC	Real-Time Clock	实时时钟
RX	Receiver	接收器
S		
SAR	Successive Approximations Register	逐次逼近寄存器
SD	Serial Data	串行数据
SD	Secure Digital	安全数字
SDIO	Secure Digital Input/Output	安全数字输入输出接口
SNR	Signal Noise Ratio	信噪比
SoC	System On Chip	片上系统
SPI	Synchronous Peripheral Interface	同步串行接口
SRAM	Static Random Access Memory	静态随机存储器
SSI	Synchronous Serial Interface	同步串行接口
STA	Station	站点
STBC	Space-Time Block Coding	空时分组编码
SWD	Serial Wire Debug	串行线调试
T		
TPC	Transmit Power Control	发射功率控制
TX	Transmitter	发送器
U		
UAPSD	Unscheduled Automatic Power Save Delivery	非调度自动节能发送
UART	Universal Asynchronous Receiver & Transmitter	通用异步收发器
UPC	UP Converter	升压模块

V		
VGA	Variable Gain Amplifier	可变增益放大器
VSWR	Voltage Standing Wave Ratio	电压驻波比
W		
WADC	WiFi Analog Digital Converter	无线模拟数字转换器
WDAC	WiFi digital analog converter	无线数字模拟转换器
WDT	Watch Dog Timer	看门狗定时器
WFI	Wait For Interrupt	等待中断
WiFi	Wireless Fidelity	无线保真
WLAN	Wireless Local Area Network	无线局域网
WPA	Wi-Fi Protected Access	Wi-Fi 保护访问
X		
XIP	Executed In Place	芯片内执行